UFC Machine Learning Prediction Project

*All Post-Scraping Work

- Author: Travis Royce
- traviscroyce@gmail.com
- linkedin.com/in/travis-royce

Project Overview

The purpose of this project is to create a model to predict the outcome of UFC (Ultimate Fighting Championship) events. The UFC was created in 1993 with the expressed purpose of identifying the most effective martial art(s). This project is an extension of my curiosity into the original purpose of the UFC; what makes an effective martial artist, relative to an opponent?

Business Overview

This project's end result will be an application which can help identify descrepancies between the "Vegas" odds of a fight and the model-predicted (i.e., theoretically more accurate) odds of that same fight. This could be used by bookmakers to increase revenues by offering slightly "better" odds than competitor bookmakers, given the model indicates it is appropriate. Alternatively, it could be used by bookmakers to create more accurate odds than previously, as the favorite only wins around 60% of the time in MMA.

Data Overview

Data Description

The majority of the data I used for this project I scraped from UFCStats.com. This website contains more statistics than any other website, but does not have some key metrics such as fighter sizes, bios, and odds.

Thus, the odds were scraped from bestfightodds.com, while the fighter sizes and bios were scraped from ufc.com.

Further, individual events and fights, and much from the final streamlit application, are scraped from ufc.com.

The data itself is fight-by-fight based data, originally from over 8,000 fights (which, after dropping for lack of data, decreased to around 5,000 by the final testing dataframe).

There are 350 final tested features in this dataset, and over 450 features in the non-tested version of the dataset, as many of the features are in-match statistics that are unknowable before a fight, so they had to be removed from the testing section.

Data Limitations

I would really like to be able to categorize these martial artists based on their initial skillset and studies, but this is not possible with any dataset that I have seen. For instance, if a martial artist is a blackbelt in Jiujitsu since he was 6, I would like to know! Ideally, in further iterations of this project, I would be able to add historical, contextual data such as this.

Define Target Variable

The target variable in this project is if a fighter won an individual fight or not.

Define Scoring Metric

Because our data is evenly split between wins and losses, and for each fight there is only the choice of Fighter A or Fighter B, it renders recall and precision equivalent. Therefore, I chose to use accuracy.

Project Structure

As there was no up-to-date database available, a good portion of this notebook is scraping and saving data using various methods with beautiful soup and selenium.

Feature Addition and Prediction

In [ ]:
```
# Load Packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sqlite3
import seaborn as sns
from bs4 import BeautifulSoup
import time
import shutil
import datetime
from scipy.stats import norm
import warnings
warnings.filterwarnings('ignore')
from random import randint
import  random
import os
```

```
os.chdir('C:/Users/Travis/OneDrive/Data Science/Personal_Projects/Sports/UFC_Predic
from cmath import nan
import pandas as pd
import numpy as np
%matplotlib inline
import matplotlib.ticker as mtick
import sqlite3
import seaborn as sns
from matplotlib.pyplot import figure
from bs4 import BeautifulSoup
import time
import shutil
import datetime
from scipy.stats import norm
import requests
import json
import xgboost
from xgboost import XGBClassifier
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from bs4 import BeautifulSoup
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from sklearn.pipeline import Pipeline, make_pipeline, FeatureUnion
from sklearn.tree import plot_tree
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.compose import make_column_selector as selector, ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler, OrdinalEncoder
import pickle
from sklearn.metrics import fbeta_score
import winsound
from sklearn.linear_model import LinearRegression
from sklearn import tree, preprocessing
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier, RandomForestClassifier, ExtraTreesC
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from matplotlib.offsetbox import OffsetImage, AnnotationBbox
from sklearn.pipeline import Pipeline, make_pipeline, FeatureUnion
from sklearn.tree import plot_tree
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.compose import make_column_selector as selector, ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler, OrdinalEncoder
from sklearn.metrics import roc_curve, auc, f1_score, make_scorer, recall_score
from sklearn.svm import SVC
```

Load fights_df, which was created in the scraping notebook.

In [ ]:
```
# Load fights df
fights_df = pd.read_csv('data/final/aggregates/Fight_DF.csv')
```

```
print(fights_df.shape)
fights_df.head()
```

```
(8359, 78)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
1	Joshua Burkman	Josh Neer	1	0	35	88	19	
2	Paddy Pimblett	Kazula Vargas	0	0	3	6	7	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

5 rows × 78 columns

Add fighter odds to dataframe

In [ ]:

```
# Load Fighter Odds
odds_by_fighter = pd.read_csv('data/final/odds/All_Odds_by_Fighter_WithChange.csv')

print(odds_by_fighter.shape)
odds_by_fighter.head()
```

```
(22954, 8)
```

Out[ ]:

	fighter	5D	Ref	event_odds_url	event_ufcstats_url	event
0	Brock Lesnar	-225.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-1
1	Frank Mir	205.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-1
2	Georges St-Pierre	-275.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-1
3	Thiago Alves	250.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-1
4	Dan Henderson	-200.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-1

In [ ]:

```
# Clean up
odds_by_fighter['event_code'] = odds_by_fighter['event_ufcstats_url'].str.split('/'
odds_by_fighter['fighter'] = odds_by_fighter['fighter'].str.strip()
odds_by_fighter.head(3)
```

Out[ ]:

	fighter	5D	Ref	event_odds_url	event_ufcstats_url	event_i
0	Brock Lesnar	-225.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13
1	Frank Mir	205.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13
2	Georges St-Pierre	-275.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13

Load event data, also created in the scraping notebook.

In [ ]:
```
event_data = pd.read_csv('data/final/events/All_Events_Fights_and_FightUrls.csv')
```

In [ ]:
```
# Clean up
event_data['Fighter1'] = event_data['Fighter1'].str.strip()
event_data['Fighter2'] = event_data['Fighter2'].str.strip()
```

In [ ]:
```
event_data.head(3)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	W/L	Weight class	Method	Round	Time	Fighter1	Fighter2	F1_Kc
0	0	0	win	Heavyweight	KO/TKO Punches	3	4:23	Ciryl Gane	Tai Tuivasa	1
1	1	1	win	Middleweight	U-DEC	3	5:00	Robert Whittaker	Marvin Vettori	0
2	2	2	win	Middleweight	U-DEC	3	5:00	Nassourdine Imavov	Joaquin Buckley	0

In [ ]:
```
# Make sure the formatting is the same for both dfs

event_data['event_id'] = event_data['event_id'].astype(str)
odds_by_fighter['event_code'] = odds_by_fighter['event_code'].astype(str)
```

In [ ]:
```
# pick random row from odds_by_fighter to test that it works
rand_row = odds_by_fighter.sample(1)
rand_row
```

Out[ ]:

	fighter	5D	Ref	event_odds_url	event_ufcstats_url	ev
11742	Jamie Pickett	275.0	268.0	https://www.bestfightodds.com/events/ufc-fight...	http://ufcstats.com/event-details/e49c2db95e57...	uf nigl thor v

In [ ]:
```
# Checking out the data so far, just a test
test_data = event_data[event_data['event_id'] == rand_row['event_code'].values[0]]
test_data.head(4)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	W/L	Weight class	Method	Round	Time	Fighter1	Fighter2	F1_
5142	5142	0	win	Welterweight	U-DEC	5	5:00	Stephen Thompson	Geoff Neal	
5143	5143	1	win	Bantamweight	U-DEC	3	5:00	Jose Aldo	Marlon Vera	
5144	5144	2	win	Welterweight	U-DEC	3	5:00	Michel Pereira	Khaos Williams	
5145	5145	3	win	Bantamweight	KO/TKO Punches	1	3:47	Rob Font	Marlon Moraes	

In [ ]:
```
def grab_fight_url(fighter, event_code):
        # This function takes in a fighter name and event code and returns the figh

        try:
                data = event_data[event_data['event_id'] == event_code]
                data1 = data[data['Fighter1'] == fighter]
                data2 = data[data['Fighter2'] == fighter]
                data3 = pd.concat([data1, data2])

                return data3['fight_link'].values[0]
        except:
                return np.nan
```

In [ ]:
```
# test
grab_fight_url('Robert Whittaker', '00a905a4a4a2b071')
```

Out[ ]:
```
'http://www.ufcstats.com/fight-details/b8ca1acdf3dc2f58'
```

Now, we add fight urls to odds_by_fighter, so that we can connect the two dataframes.

In [ ]:
```
# add fight urls to odds by fighter
odds_by_fighter['fight_url'] = odds_by_fighter.apply(lambda row: grab_fight_url(row
```

In [ ]:
```
odds_by_fighter.head(4)
```

Out[ ]:

	fighter	5D	Ref	event_odds_url	event_ufcstats_url	event_i
0	Brock Lesnar	-225.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13
1	Frank Mir	205.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13
2	Georges St-Pierre	-275.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13
3	Thiago Alves	250.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100 13

In [ ]:
```
# add columns 'odds' to odds by fighter, where odds is equal to the 5D column unles
odds_by_fighter['odds'] = odds_by_fighter.apply(lambda row: row['5D'] if np.isnan(r
```

In [ ]:
```
# save odds by fighter with fight urls
odds_by_fighter.to_csv('data/final/odds/All_Odds_by_Fighter_With_Fight_Urls.csv', i
```

Double the Fights_DF

In this dataset, we currently have one row for each fighter in a fight. We need to double the dataframe so that we have one row for each fighter in a fight. This will allow us to A) balance the dataset, and B) have a more accurate representation of the fight from the perspective of each fighter.

In [ ]:
```
# Create the double fights df (switching A and B)
fights_df2 = fights_df.copy()

# switch all _A and A_ to _C and C_
fights_df2.columns = fights_df2.columns.str.replace('Fighter_A', 'Fighter_C')
fights_df2.columns = fights_df2.columns.str.replace('A_', 'C_')

# switch all _B and B_ to _A and A_
fights_df2.columns = fights_df2.columns.str.replace('Fighter_B', 'Fighter_A')
fights_df2.columns = fights_df2.columns.str.replace('B_', 'A_')

# switch all _C and C_ to _B and B_
fights_df2.columns = fights_df2.columns.str.replace('Fighter_C', 'Fighter_B')
fights_df2.columns = fights_df2.columns.str.replace('C_', 'B_')

fights_df2.head()
```

Out[ ]:

	Fighter_B	Fighter_A	B_Kd	A_Kd	B_Sig_strike_land	B_Sig_strike_att	A_Sig_strike_land	A_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
1	Joshua Burkman	Josh Neer	1	0	35	88	19	
2	Paddy Pimblett	Kazula Vargas	0	0	3	6	7	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

5 rows × 78 columns

In [ ]:
```
# add the new columns to the original df
double_fights_df = pd.concat([fights_df, fights_df2], axis=0)
double_fights_df.head(4)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
1	Joshua Burkman	Josh Neer	1	0	35	88	19	
2	Paddy Pimblett	Kazula Vargas	0	0	3	6	7	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	

4 rows × 78 columns

In [ ]:
```
double_fights_df.columns
```

Out[ ]:
```
Index(['Fighter_A', 'Fighter_B', 'A_Kd', 'B_Kd', 'A_Sig_strike_land',
       'A_Sig_strike_att', 'B_Sig_strike_land', 'B_Sig_strike_att',
       'A_Sig_strike_percent', 'B_Sig_strike_percent', 'A_Total_Strikes',
       'B_Total_Strikes', 'A_Total_Strikes_land', 'A_Total_Strikes_att',
       'B_Total_Strikes_land', 'B_Total_Strikes_att',
       'A_Total_Strikes_percent', 'B_Total_Strikes_percent',
       'A_Takedowns_land', 'A_Takedowns_att', 'B_Takedowns_land',
       'B_Takedowns_att', 'A_Takedown_percent', 'B_Takedown_percent',
       'A_Sub_Attempts_land', 'A_Sub_Attempts_att', 'B_Sub_Attempts_land',
       'B_Sub_Attempts_att', 'A_Rev', 'B_Rev', 'A_Ctrl_time_min',
       'A_Ctrl_time_sec', 'B_Ctrl_time_min', 'B_Ctrl_time_sec',
       'A_Ctrl_time_tot', 'B_Ctrl_time_tot', 'details', 'event_title',
       'event_url', 'date', 'Winner', 'fight_id', 'A_Head_Strikes_land',
       'A_Head_Strikes_att', 'B_Head_Strikes_land', 'B_Head_Strikes_att',
       'A_Head_Strikes_percent', 'B_Head_Strikes_percent',
       'A_Body_Strikes_land', 'A_Body_Strikes_att', 'B_Body_Strikes_land',
       'B_Body_Strikes_att', 'A_Body_Strikes_percent',
       'B_Body_Strikes_percent', 'A_Leg_Strikes_land', 'A_Leg_Strikes_att',
       'B_Leg_Strikes_land', 'B_Leg_Strikes_att', 'A_Leg_Strikes_percent',
       'B_Leg_Strikes_percent', 'A_Distance_Strikes_land',
       'A_Distance_Strikes_att', 'B_Distance_Strikes_land',
       'B_Distance_Strikes_att', 'A_Distance_Strikes_percent',
       'B_Distance_Strikes_percent', 'A_Clinch_Strikes_land',
       'A_Clinch_Strikes_att', 'B_Clinch_Strikes_land', 'B_Clinch_Strikes_att',
       'A_Clinch_Strikes_percent', 'B_Clinch_Strikes_percent',
       'A_Ground_Strikes_land', 'A_Ground_Strikes_att',
       'B_Ground_Strikes_land', 'B_Ground_Strikes_att',
       'A_Ground_Strikes_percent', 'B_Ground_Strikes_percent'],
      dtype='object')
```

In [ ]:
```
double_fights_df.to_csv('data/final/aggregates/Fight_DF_V2.csv', index=False)
```

In [ ]:
```
double_fights_df['event_url'].unique()
```

Out[ ]:
```
array(['http://www.ufcstats.com/event-details/805ad1801eb26abb',
       'http://www.ufcstats.com/event-details/f70144caea5c4c80',
       'http://www.ufcstats.com/event-details/1d00756835ca67c9', ...,
       'http://www.ufcstats.com/event-details/14a433bccba87016',
       'http://www.ufcstats.com/event-details/8dc46e9a7895ce1a',
       'http://www.ufcstats.com/event-details/68d76fd9ecf7431d'],
      dtype=object)
```

In [ ]:
```
double_fights_df.head()
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
1	Joshua Burkman	Josh Neer	1	0	35	88	19	
2	Paddy Pimblett	Kazula Vargas	0	0	3	6	7	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

5 rows × 78 columns

Add Odds by Fighter to FightsDf

Load Point for OddsByFighter

In [ ]:
```
# load
odds_by_fighter = pd.read_csv('data/final/odds/All_Odds_by_Fighter_With_Fight_Urls.
```

Note: We want to use ufcstats Urls as our connector

We could also just use the event code if that isnt working for some reason

In [ ]:
```
odds_by_fighter.head(2)
```

Out[ ]:

	fighter	5D	Ref	event_odds_url	event_ufcstats_url	event_id
0	Brock Lesnar	-225.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100-137
1	Frank Mir	205.0	NaN	https://www.bestfightodds.com/events/ufc-100-137	http://ufcstats.com/event-details/0ee783aa00e4...	ufc-100-137

In [ ]:
```
# make sure formatting is the same (string)
odds_by_fighter['event_ufcstats_url'] = odds_by_fighter['event_ufcstats_url'].astyp
odds_by_fighter['event_code'] = odds_by_fighter['event_code'].astype(str)

double_fights_df['event_url'] = double_fights_df['event_url'].astype(str)
```

In [ ]:
```
# make sure name formatting is same
odds_by_fighter['fighter'] = odds_by_fighter['fighter'].str.strip()
```

```
double_fights_df['Fighter_A'] = double_fights_df['Fighter_A'].str.strip()
double_fights_df['Fighter_B'] = double_fights_df['Fighter_B'].str.strip()
```

In [ ]:
```
# Add column "Event Code", which is event code for ufcstats

double_fights_df['event_code'] = double_fights_df['event_url'].str.split('/').str[-
```

In [ ]:
```
double_fights_df['event_code'] = double_fights_df['event_code'].astype(str)
```

In [ ]:
```
# check the differences between double_fights_df['event_code'] and odds_by_fighter[
dfights = double_fights_df['event_code'].unique()
odds = odds_by_fighter['event_code'].unique()

# check differences in list
diff = [x for x in dfights if x not in odds]
```

In [ ]:
```
diff2 = [x for x in odds if x not in dfights]
diff2
```

Out[ ]:
```
['885e7f70dcac0007',
 'nan',
 '49c29e57d4e2be6f',
 'be5aab761c40ef35',
 '756f45905fb20cb5',
 '4f670b7972fa0a2e']
```

Note: The double fights DF has lots of OUTSIDE of UFC fights, whereas the odds does not. As we want to focus on the UFC events, this is okay. In summation, it is fine that we are losing some non-UFC fight data here.

In [ ]:
```
# Check Dtypes of odds_by_fighter
odds_by_fighter.dtypes
```

Out[ ]:
```
fighter               object
5D                   float64
Ref                  float64
event_odds_url        object
event_ufcstats_url    object
event_id              object
event_name            object
odds_change          float64
event_code            object
fight_url             object
odds                 float64
dtype: object
```

In [ ]:
```
odds_by_fighter['event_code'] = odds_by_fighter['event_code'].astype(str)
odds_by_fighter['event_ufcstats_url'] = odds_by_fighter['event_ufcstats_url'].astyp
```

In [ ]:
```
def get_odds_from_obf(event_code, fighter):
        # Function takes in event code and fighter name and returns the odds ]
        # for that fighter in that event
        try:
                data = odds_by_fighter[odds_by_fighter['event_code'] == event_code]
```

```
                data = data[data['fighter'] == fighter]
                return data['odds'].values[0]
        except:
                return np.nan
```

In [ ]:
```
# test
get_odds_from_obf('805ad1801eb26abb', 'Holly Holm')
```

Out[ ]:
```
-125.0
```

In [ ]:
```
double_fights_df['Fighter_A_Odds'] = double_fights_df.apply(lambda row: get_odds_fr
double_fights_df['Fighter_B_Odds'] = double_fights_df.apply(lambda row: get_odds_fr
```

In [ ]:
```
# check missing
len(double_fights_df[double_fights_df['Fighter_A_Odds'].isna()])
```

Out[ ]:
```
6913
```

In [ ]:
```
# check not missing
len(double_fights_df[double_fights_df['Fighter_A_Odds'].notna()])
```

Out[ ]:
```
9805
```

Add Odds Change to FightsDf

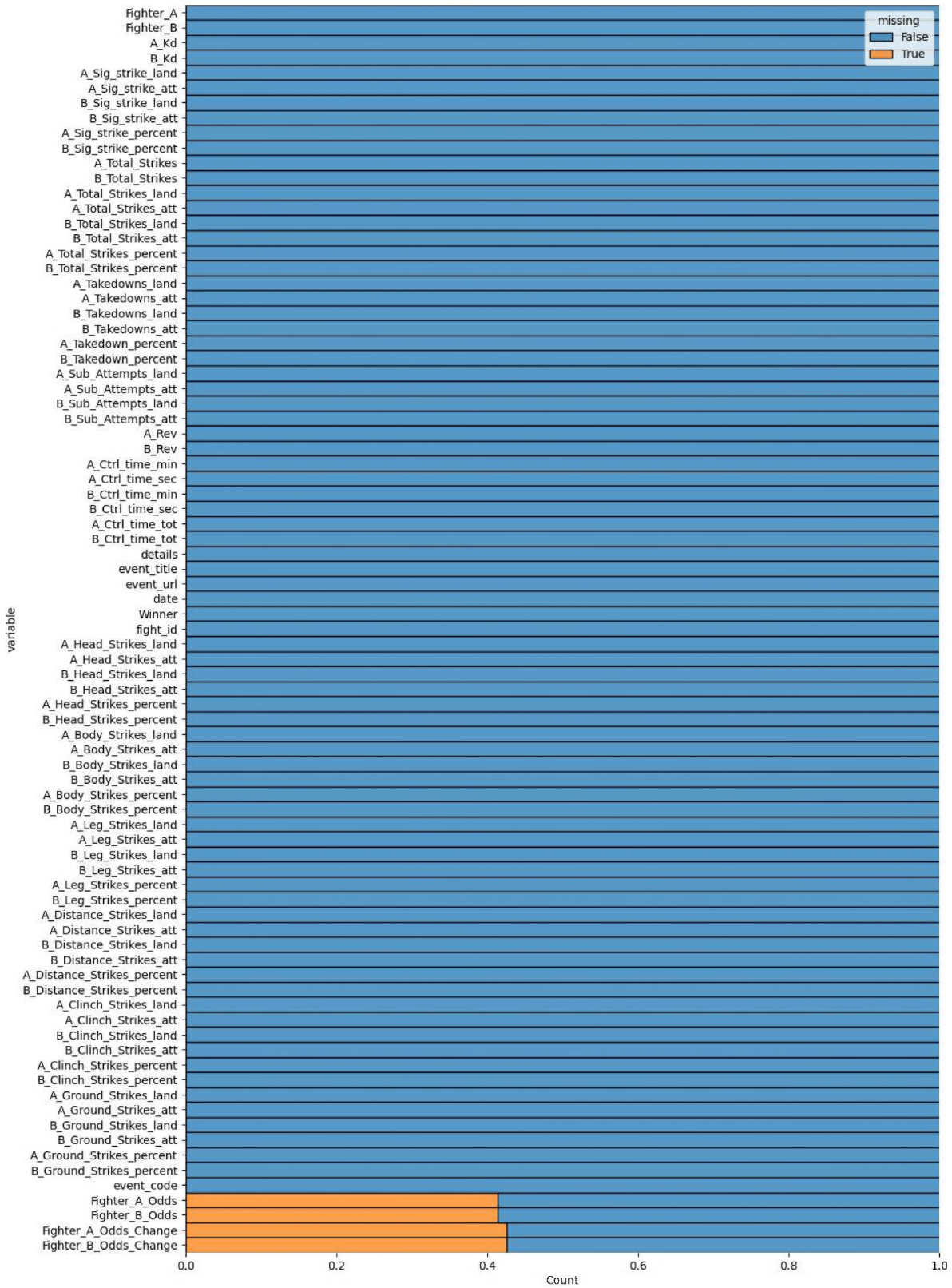
In [ ]:
```
def get_odds_change_from_obf(event_code, fighter):
        # Function takes in event code and fighter name and returns the odds change
        try:
                data = odds_by_fighter[odds_by_fighter['event_code'] == event_code]
                data = data[data['fighter'] == fighter]
                return data['odds_change'].values[0]
        except:
                return np.nan
```

In [ ]:
```
# add odds change
double_fights_df['Fighter_A_Odds_Change'] = double_fights_df.apply(lambda row: get_
double_fights_df['Fighter_B_Odds_Change'] = double_fights_df.apply(lambda row: get_
```

Check the missing values in the dataset

In [ ]:
```
f, ax = plt.subplots(figsize = (12,20))

sns.despine(f, left=True, bottom=True)
sns.histplot(
    data = double_fights_df.isna().melt(value_name='missing'),
    y= 'variable',
    hue = 'missing',
    multiple = 'fill', ax = ax)
```

Out[ ]:
```
<AxesSubplot:xlabel='Count', ylabel='variable'>
```



As we know from earlier, we do not have many of these odds because these events were outside of the UFC. Thus, we can drop them.

In [ ]:
```
double_fights_df = double_fights_df.dropna()
double_fights_df.shape
```

Out[ ]: (8694, 83)

In [ ]:
```
# Save point for safety
double_fights_df.to_csv('data/final/aggregates/Double_Fights_DF_V1.csv', index=Fals
```

In [ ]:
```
double_fights_df.head()
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_s
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	
5	Alistair Overeem	Stefan Struve	0	0	17	25	2	
10	Nordine Taleb	Kyle Prepolec	0	0	90	201	52	

5 rows × 83 columns

In [ ]:
```
# find all objects in df
double_fights_df.select_dtypes(include=['object']).columns
```

Out[ ]:
```
Index(['Fighter_A', 'Fighter_B', 'A_Total_Strikes', 'B_Total_Strikes',
       'details', 'event_title', 'event_url', 'date', 'Winner', 'fight_id',
       'event_code'],
      dtype='object')
```

In [ ]:
```
# drop A_Total_Strikes and B_Total_Strikes, they were not meant to be here.

double_fights_df = double_fights_df.drop(['A_Total_Strikes', 'B_Total_Strikes'], ax
```

Add Feature: In-Fight Statistic Differentials

This is

Calculate differences in (in-match) fight statistics.

In [ ]:
```
for col in double_fights_df.columns:
    if col.startswith('A_'):
        new_col = col.replace('A_', 'Dif_')
        double_fights_df[new_col] = double_fights_df[col] - double_fights_df[col.re
    else:
        continue

double_fights_df
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Si
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	
5	Alistair Overeem	Stefan Struve	0	0	17	25	2	
10	Nordine Taleb	Kyle Prepolec	0	0	90	201	52	
...	...	...	...	...	...	...	...	
8346	Marcos Rogerio	Blagoy Ivanov	0	0	68	133	62	
8347	Marcus Davis	Mike Swick	0	0	31	55	32	
8351	Lucas Martins	Darren Elkins	0	0	26	82	47	
8352	Krzysztof Jotko	Sean Strickland	0	0	37	196	84	
8356	Gleison Tibau	Tony Ferguson	0	0	2	20	11	

8694 rows × 115 columns

In [ ]:
```
# add difference in odds
double_fights_df['Dif_Odds'] = double_fights_df['Fighter_A_Odds'] - double_fights_d
```

Add Distribution Stat Features

Here, double_fights_df becomes fights, just for ease.

In [ ]:
```
fights = double_fights_df
```

In [ ]:
```
def get_fighter_running_dist_stats(fighter, date, col_to_get, stat_to_calc):
    # for each column in all_metric_cols, get the mean, std, etc. for each fighter

    # filter data
    data = fights[(fights['Fighter_A'] == fighter) | (fights['Fighter_B'] == fighte

    # only get fights before the date
    datey = pd.to_datetime(date)
    data['date'] = pd.to_datetime(data['date'])
    data = data[data['date'] < datey]
```

```
    # fighter could be either fighter A or fighter B
    fighter_data = pd.DataFrame()
    # when fighter is fighter A, get all fighter A data and copy it to fighter_data

    # fighterA df
    fighterA_df = fights[fights['Fighter_A'] == fighter]
    fighterB_df = fights[fights['Fighter_B'] == fighter]

    # keep only the fighters columns date, FighterA, and the col_to_get,
    # do same for B, change col names from B to A, and concat
    fighterA_df = fighterA_df[['date', 'Fighter_A', 'A_' + col_to_get]]
    fighterA_df.rename(columns={'A_' + col_to_get: col_to_get, 'Fighter_A': 'fighte
    fighterB_df = fighterB_df[['date', 'Fighter_B', 'B_' + col_to_get]]
    fighterB_df.rename(columns={'B_' + col_to_get: col_to_get, 'Fighter_B': 'fighte

    # append the dataframes on fighter
    fighter_data = fighter_data.append(fighterA_df)
    fighter_data = fighter_data.append(fighterB_df)

    # get the chosen statistic
    if stat_to_calc == 'mean':
        x = fighter_data[col_to_get].mean()
    elif stat_to_calc == 'std':
        x = fighter_data[col_to_get].std()
    elif stat_to_calc == 'median':
        x = fighter_data[col_to_get].median()
    return x
```

Get Non-specific column names for Feature Creation

In [ ]:
```
dif_cols = [n for n in fights.columns if 'Dif_' in n]
A_cols = [n for n in fights.columns if n.startswith('A_')]
all_metric_cols = A_cols + dif_cols
A_cols2 = pd.DataFrame(A_cols)
A_cols2['nonspecific'] = A_cols2[0].str[2:]
the_cols = list(A_cols2['nonspecific'].unique())
```

In [ ]:
```
# test get_fighter_running_dist_stats
get_fighter_running_dist_stats('Khabib Nurmagomedov', '2020-01-18', 'Sig_strike_lan
```

Out[ ]:
```
55.75
```

Get the mean, standard deviation, and median of each feature.

These features I call 'Rolling' features, because they are rolling metrics based on fighter's previous fights. Further, these features are calculated as a distribution of the fighter's previous fights.

In [ ]:
```
for col in the_cols:
    for stat in ['mean', 'std', 'median']:
        fights['A_Rolling_' + col + '_' + stat] = fights.apply(lambda row: get_figh
        fights['B_Rolling_' + col + '_' + stat] = fights.apply(lambda row: get_figh
```

In [ ]:
```
# Save for safety
fights.to_csv('data/final/aggregates/Double_Fights_DF_V2.csv', index=False)
```

Feature: Top-Down Averages (typical UFC Style)

In [ ]:
```
def get_top_down_averages(fighter, date, col_to_get, dataframe):

    # for each column in all_metric_cols, calculate the top-down averages, as oppos

    data = dataframe[(dataframe['Fighter_A'] == fighter) | (dataframe['Fighter_B']

    # only get fights before the date

    datey = pd.to_datetime(date)
    data['date'] = pd.to_datetime(data['date'])
    data = data[data['date'] < datey]

    # fighter could be either fighter A or fighter B

    fighter_data = pd.DataFrame()

    # when fighter is fighter A, get all fighter A data and copy it to fighter_data

    # fighterA df

    fighterA_df = dataframe[dataframe['Fighter_A'] == fighter]
    fighterB_df = dataframe[dataframe['Fighter_B'] == fighter]

    # keep only the fighters columns date, FighterA, and the col_to_get,
    # do same for B, change col names from B to A, and concat

    fighterA_df = fighterA_df[['date', 'Fighter_A', 'A_' + col_to_get]]
    fighterA_df.rename(columns={'A_' + col_to_get: col_to_get, 'Fighter_A': 'fighte
    fighterB_df = fighterB_df[['date', 'Fighter_B', 'B_' + col_to_get]]
    fighterB_df.rename(columns={'B_' + col_to_get: col_to_get, 'Fighter_B': 'fighte
    fighter_data = fighter_data.append(fighterA_df)
    fighter_data = fighter_data.append(fighterB_df)

    # append the dataframes on fighter

    fighter_data = fighter_data.append(fighterA_df)
    fighter_data = fighter_data.append(fighterB_df)

    # get the average

    tot = fighter_data[col_to_get].sum()
    num = fighter_data[col_to_get].count()
    x = tot / num
    return x
```

In [ ]:
```
# delete rolling cols from the_cols
the_cols = [n for n in the_cols if 'Rolling' not in n]
the_cols
```

Out[ ]:
```
['Kd',
 'Sig_strike_land',
 'Sig_strike_att',
 'Sig_strike_percent',
 'Total_Strikes_land',
 'Total_Strikes_att',
 'Total_Strikes_percent',
 'Takedowns_land',
 'Takedowns_att',
 'Takedown_percent',
 'Sub_Attempts_land',
 'Sub_Attempts_att',
 'Rev',
 'Ctrl_time_min',
 'Ctrl_time_sec',
 'Ctrl_time_tot',
 'Head_Strikes_land',
 'Head_Strikes_att',
 'Head_Strikes_percent',
 'Body_Strikes_land',
 'Body_Strikes_att',
 'Body_Strikes_percent',
 'Leg_Strikes_land',
 'Leg_Strikes_att',
 'Leg_Strikes_percent',
 'Distance_Strikes_land',
 'Distance_Strikes_att',
 'Distance_Strikes_percent',
 'Clinch_Strikes_land',
 'Clinch_Strikes_att',
 'Clinch_Strikes_percent',
 'Ground_Strikes_land',
 'Ground_Strikes_att',
 'Ground_Strikes_percent']
```

In [ ]:
```
for col in the_cols:
    fights['A_topdown_Avg_' + col ] = fights.apply(lambda row: get_top_down_average
    fights['B_topdown_Avg_' + col ] = fights.apply(lambda row: get_top_down_average
```

In [ ]:
```
fights.to_csv('data/final/aggregates/Double_Fights_DF_V3.csv')
```

Add Opponent Stats

Now, we add the opponent top-down averages to the dataframe.

In [ ]:
```
def get_opponent_averages(datafrm, date, col_to_get, fighter):
    col_to_get = 'B_' + col_to_get
    data = datafrm[(datafrm['Fighter_A'] == fighter) | (datafrm['Fighter_B'] == fig
    datey = pd.to_datetime(date)
    data['date'] = pd.to_datetime(data['date'])
    data = data[data['date'] < datey]

    fighter_data = pd.DataFrame()
```

```
    fighterA_df = datafrm[datafrm['Fighter_A'] == fighter]
    fighterB_df = datafrm[datafrm['Fighter_B'] == fighter]

    # switch A_ and B_ to B_ and A_ in fighterB_df
    fighterB_df.columns = [n.replace('A_', 'C_') if n.startswith('A_') else n for n
    fighterB_df.columns = [n.replace('B_', 'D_') if n.startswith('B_') else n for n
    fighterB_df.columns = [n.replace('C_', 'B_') if n.startswith('C_') else n for n
    fighterB_df.columns = [n.replace('D_', 'A_') if n.startswith('D_') else n for n

    # copy fighter_A and fighter_B columns to new dataframe, then switch fighter_A

    fighterB_df.rename(columns={'Fighter_A': 'Fighter_A2', 'Fighter_B': 'Fighter_B2
    fighterB_df.rename(columns={'Fighter_B2': 'Fighter_A', 'Fighter_A2': 'Fighter_B

    fighter_data = fighter_data.append(fighterA_df)
    fighter_data = fighter_data.append(fighterB_df)

    # get sum of col to get
    col_sum = fighter_data[col_to_get].sum()
    # get number of fights
    num_fights = len(fighter_data)
    # get average
    avg = col_sum / num_fights

    return avg
```

In [ ]:
```
# test
get_opponent_averages(fights, '2017-01-01', 'Sig_strike_land', 'Conor McGregor')
```

Out[ ]:
```
37.42857142857143
```

In [ ]:
```
# Determine columns to calculate opp averages for
cols = [n for n in fights.columns if n.startswith('B_')]
cols = [n for n in cols if 'topdown_' not in n]
cols = [n for n in cols if 'Rolling_' not in n]
cols = [n for n in cols if 'Dif_' not in n]
```

In [ ]:
```
# delete the first 2 characters of each in col
cols2 = pd.DataFrame(cols)
cols2['nonspecific'] = cols2[0].str[2:]
cols2

# get unique values
the_cols = list(cols2['nonspecific'].unique())
the_cols
```

Out[ ]:
```
['Kd',
 'Sig_strike_land',
 'Sig_strike_att',
 'Sig_strike_percent',
 'Total_Strikes_land',
 'Total_Strikes_att',
 'Total_Strikes_percent',
 'Takedowns_land',
 'Takedowns_att',
 'Takedown_percent',
 'Sub_Attempts_land',
 'Sub_Attempts_att',
 'Rev',
 'Ctrl_time_min',
 'Ctrl_time_sec',
 'Ctrl_time_tot',
 'Head_Strikes_land',
 'Head_Strikes_att',
 'Head_Strikes_percent',
 'Body_Strikes_land',
 'Body_Strikes_att',
 'Body_Strikes_percent',
 'Leg_Strikes_land',
 'Leg_Strikes_att',
 'Leg_Strikes_percent',
 'Distance_Strikes_land',
 'Distance_Strikes_att',
 'Distance_Strikes_percent',
 'Clinch_Strikes_land',
 'Clinch_Strikes_att',
 'Clinch_Strikes_percent',
 'Ground_Strikes_land',
 'Ground_Strikes_att',
 'Ground_Strikes_percent']
```

In [ ]:
```
n=0

for col in the_cols:
    fights['A_Opp_Avg_' + col] = fights.apply(lambda row: get_opponent_averages(fig
    fights['B_Opp_Avg_' + col] = fights.apply(lambda row: get_opponent_averages(fig
    n = n+1
    print(f' {col}, #{n} / {len(cols)} done.')
```

```
Kd, #1 / 34 done.
Sig_strike_land, #2 / 34 done.
Sig_strike_att, #3 / 34 done.
Sig_strike_percent, #4 / 34 done.
Total_Strikes_land, #5 / 34 done.
Total_Strikes_att, #6 / 34 done.
Total_Strikes_percent, #7 / 34 done.
Takedowns_land, #8 / 34 done.
Takedowns_att, #9 / 34 done.
Takedown_percent, #10 / 34 done.
Sub_Attempts_land, #11 / 34 done.
Sub_Attempts_att, #12 / 34 done.
Rev, #13 / 34 done.
Ctrl_time_min, #14 / 34 done.
Ctrl_time_sec, #15 / 34 done.
Ctrl_time_tot, #16 / 34 done.
Head_Strikes_land, #17 / 34 done.
Head_Strikes_att, #18 / 34 done.
Head_Strikes_percent, #19 / 34 done.
Body_Strikes_land, #20 / 34 done.
Body_Strikes_att, #21 / 34 done.
Body_Strikes_percent, #22 / 34 done.
Leg_Strikes_land, #23 / 34 done.
Leg_Strikes_att, #24 / 34 done.
Leg_Strikes_percent, #25 / 34 done.
Distance_Strikes_land, #26 / 34 done.
Distance_Strikes_att, #27 / 34 done.
Distance_Strikes_percent, #28 / 34 done.
Clinch_Strikes_land, #29 / 34 done.
Clinch_Strikes_att, #30 / 34 done.
Clinch_Strikes_percent, #31 / 34 done.
Ground_Strikes_land, #32 / 34 done.
Ground_Strikes_att, #33 / 34 done.
Ground_Strikes_percent, #34 / 34 done.
```

In [ ]:
```
# save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V4.csv')
```

Feature: Rolling Career Stat Differentials

This is the difference between fighter_A and fighter_B's rolling career stats.

1) Mean (Average)

In [ ]:
```
# get columns with Rolling and Mean
rolling_mean_cols= [n for n in fights.columns if 'Rolling' in n and 'mean' in n]
rolling_mean_cols = pd.DataFrame(rolling_mean_cols)
rolling_mean_cols['nonspecific'] = rolling_mean_cols[0].str[2:]
rmc = rolling_mean_cols['nonspecific'].unique()
```

In [ ]:
```
for col in rmc:
    fights['Dif_' + col] = fights['A_' + col] - fights['B_' + col]
```

```
fights.head()
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_s
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	
5	Alistair Overeem	Stefan Struve	0	0	17	25	2	
10	Nordine Taleb	Kyle Prepolec	0	0	90	201	52	

5 rows × 490 columns

2) Median (Average)

In [ ]:
```
rolling_median_cols= [n for n in fights.columns if 'Rolling' in n and 'median' in n
rolling_median_cols = pd.DataFrame(rolling_median_cols)
rolling_median_cols['nonspecific'] = rolling_median_cols[0].str[2:]
rmc = rolling_median_cols['nonspecific'].unique()
```

In [ ]:
```
for col in rmc:
    fights['Dif_' + col] = fights['A_' + col] - fights['B_' + col]

fights.head()
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_s
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	
5	Alistair Overeem	Stefan Struve	0	0	17	25	2	
10	Nordine Taleb	Kyle Prepolec	0	0	90	201	52	

5 rows × 524 columns

3) Standard Deviations

In [ ]:
```
rolling_std_cols= [n for n in fights.columns if 'Rolling' in n and 'std' in n]
rolling_std_cols = pd.DataFrame(rolling_std_cols)
rolling_std_cols['nonspecific'] = rolling_std_cols[0].str[2:]
rsc = rolling_std_cols['nonspecific'].unique()
rsc
```

Out[ ]:
```
array(['Rolling_Kd_std', 'Rolling_Sig_strike_land_std',
       'Rolling_Sig_strike_att_std', 'Rolling_Sig_strike_percent_std',
       'Rolling_Total_Strikes_land_std', 'Rolling_Total_Strikes_att_std',
       'Rolling_Total_Strikes_percent_std', 'Rolling_Takedowns_land_std',
       'Rolling_Takedowns_att_std', 'Rolling_Takedown_percent_std',
       'Rolling_Sub_Attempts_land_std', 'Rolling_Sub_Attempts_att_std',
       'Rolling_Rev_std', 'Rolling_Ctrl_time_min_std',
       'Rolling_Ctrl_time_sec_std', 'Rolling_Ctrl_time_tot_std',
       'Rolling_Head_Strikes_land_std', 'Rolling_Head_Strikes_att_std',
       'Rolling_Head_Strikes_percent_std',
       'Rolling_Body_Strikes_land_std', 'Rolling_Body_Strikes_att_std',
       'Rolling_Body_Strikes_percent_std', 'Rolling_Leg_Strikes_land_std',
       'Rolling_Leg_Strikes_att_std', 'Rolling_Leg_Strikes_percent_std',
       'Rolling_Distance_Strikes_land_std',
       'Rolling_Distance_Strikes_att_std',
       'Rolling_Distance_Strikes_percent_std',
       'Rolling_Clinch_Strikes_land_std',
       'Rolling_Clinch_Strikes_att_std',
       'Rolling_Clinch_Strikes_percent_std',
       'Rolling_Ground_Strikes_land_std',
       'Rolling_Ground_Strikes_att_std',
       'Rolling_Ground_Strikes_percent_std'], dtype=object)
```

In [ ]:
```
for col in rsc:
    fights['Dif_' + col] = fights['A_' + col] - fights['B_' + col]

fights.head()
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_s
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	
5	Alistair Overeem	Stefan Struve	0	0	17	25	2	
10	Nordine Taleb	Kyle Prepolec	0	0	90	201	52	

5 rows × 558 columns

In [ ]:
```
# Save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V5.csv')
```

In [ ]:
```
#Identify columns with missing values
nothere = fights.isna().sum()
nothere = pd.DataFrame(nothere)
nothere = nothere.loc[nothere[0] > 0]

if len(nothere) > 0:
    cols = nothere.index

    f, ax = plt.subplots(figsize = (12,12))

    sns.despine(f, left=True, bottom=True)
    sns.histplot(
        data = fights[cols].isna().melt(value_name='missing'),
        y= 'variable',
        hue = 'missing',
        multiple = 'fill', ax = ax)

    ax.set_title('Missing Data by Column', fontsize = 20)
    ax.set_xlabel('Count', fontsize = 16)
    ax.set_ylabel('Column', fontsize = 16)
    ax.tick_params(labelsize = 14)

    plt.show()

else:
    print('No missing values')
```

```
No missing values
```

Feature: UFC.Com Bio Data

UFC Bio data includes fighter height, weight, reach, and stance, among other important features.

In [ ]:
```
fighter_bios = pd.read_csv('data/final/aggregates/All_Fighter_Bios.csv')
fighter_bios.head(1)
```

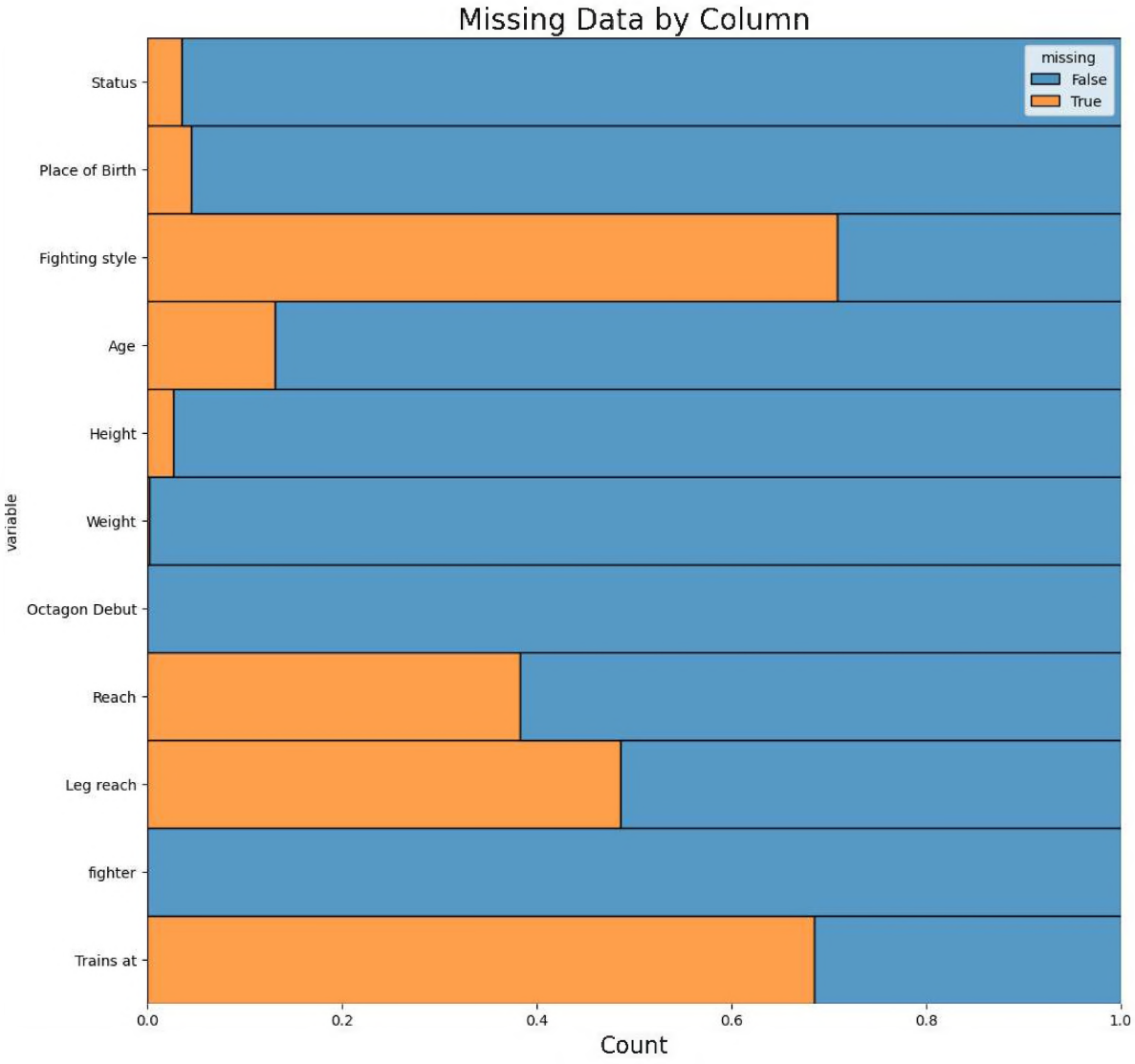
Out[ ]:

	Unnamed: 0.1	Unnamed: 0	Status	Place of Birth	Fighting style	Age	Height	Weight	Octagon Debut	Reach	Leg reach
0	0	1	Not Fighting	Parrish, United States	MMA	32.0	72.0	155.0	Jul. 30, 2019	78.0	42.(

In [ ]:
```
# drop unnamed column
fighter_bios.drop(columns=['Unnamed: 0', 'Unnamed: 0.1'], inplace=True)
```

In [ ]:
```
# Check NaNs

f, ax = plt.subplots(figsize = (12,12))

sns.despine(f, left=True, bottom=True)
sns.histplot(
    data = fighter_bios.isna().melt(value_name='missing'),
    y= 'variable',
    hue = 'missing',
    multiple = 'fill', ax = ax)

ax.set_title('Missing Data by Column', fontsize = 20)
ax.set_xlabel('Count', fontsize = 16)
```

Out[ ]:
```
Text(0.5, 0, 'Count')
```



In [ ]:
```
# Check active fighter NANs
active = fighter_bios.loc[fighter_bios['Status'] == "Active"]
active
```

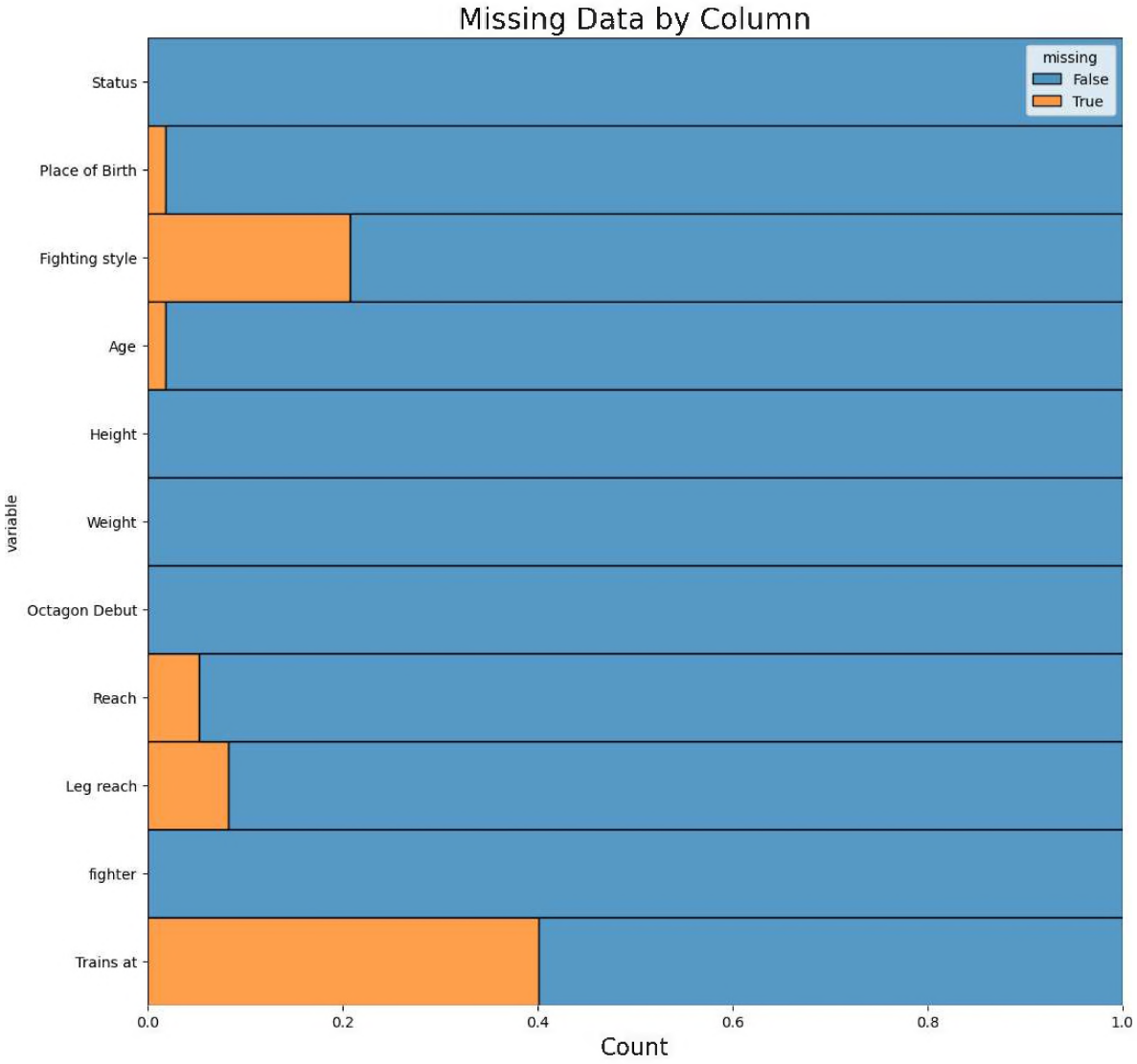
Out[ ]:

	Status	Place of Birth	Fighting style	Age	Height	Weight	Octagon Debut	Reach	Leg reach	fighter
3	Active	Houston, United States	Muay Thai	32.0	69.0	135.0	May. 24, 2014	71.0	40.0	Aaron Phillips
11	Active	Rabat, Morocco	Striker	36.0	69.0	185.5	Jul. 22, 2018	76.0	41.0	Abu Azaitar
12	Active	Republic of Dagestan, Russia	NaN	33.0	71.0	170.5	Nov. 09, 2019	72.0	39.5	Abubakar Nurmagomedov
13	Active	Argun, Russia	MMA	32.0	74.0	186.0	Sep. 03, 2022	78.0	43.0	Abus Magomedov
14	Active	San Jose, United States	NaN	40.0	65.0	125.0	Aug. 29, 2017	NaN	NaN	Adam Antolin
...	...	...	...	...	...	...	...	...	...	...
2408	Active	Irving, United States	Jiu-Jitsu	37.0	72.0	185.0	Aug. 29, 2013	75.0	40.0	Zak Cummings
2411	Active	Paris, France	MMA	38.0	68.0	147.0	Oct. 05, 2019	72.0	42.0	Zarah Fairn
2413	Active	Kazakhstan	Freestyle	34.0	64.0	139.8	Jul. 11, 2020	66.5	36.0	Zhalgas Zhumagulov
2415	Active	Hebei, China	Muay Thai	33.0	64.0	114.8	Aug. 04, 2018	63.0	36.0	Zhang Weili
2416	Active	USSR, Russia	Striker	31.0	68.0	167.0	Feb. 15, 2014	68.0	38.5	Zubaira Tukhugov

641 rows × 11 columns

In [ ]:
```
# Check NaNs

f, ax = plt.subplots(figsize = (12,12))

sns.despine(f, left=True, bottom=True)
sns.histplot(
    data = active.isna().melt(value_name='missing'),
    y= 'variable',
    hue = 'missing',
    multiple = 'fill', ax = ax)

ax.set_title('Missing Data by Column', fontsize = 20)
ax.set_xlabel('Count', fontsize = 16)
```

Out[ ]: Text(0.5, 0, 'Count')



Dealing with Initial Missing Values

Initial missing values are those such as Age, Status, Fighting Style (categorical variables) which I can impute unknown categories to without much issue. The others (height, weight, reach, etc) must be dealt with after the merge.

In [ ]:
```
# if missing place of birth, replace with 'Unknown'
fighter_bios['Place of Birth'].fillna('Unknown', inplace=True)

# if missing fighting style, replace with 'Unknown'
fighter_bios['Fighting style'].fillna('Unknown', inplace=True)

# if missing Trains at, replace with 'Unknown'
fighter_bios['Trains at'].fillna('Unknown', inplace=True)

# if missing status, replace with 'Unknown'
fighter_bios['Status'].fillna('Unknown', inplace=True)

# if missing age, replace with median
```

```
fighter_bios['Age'].fillna(fighter_bios['Age'].median(), inplace=True)

# if missing height, replace with median -- thankfully not many missing, as
# this is a somewhat important feature
fighter_bios['Height'].fillna(fighter_bios['Height'].median(), inplace=True)

# if missing weight, replace with median -- thankfully not many missing
fighter_bios['Weight'].fillna(fighter_bios['Weight'].median(), inplace=True)
```

In [ ]:
```
def get_bio_data(fighter, stat):
    try:
        data = fighter_bios[fighter_bios['fighter'] == fighter]
        d = data[stat].values[0]
        d = pd.to_numeric(d)
        return d
    except:
        return nan
```

In [ ]:
```
# test
weight = get_bio_data('Khabib Nurmagomedov', 'Height')
weight
```

Out[ ]:
```
70.0
```

Grab data from ufc bios, add to fights

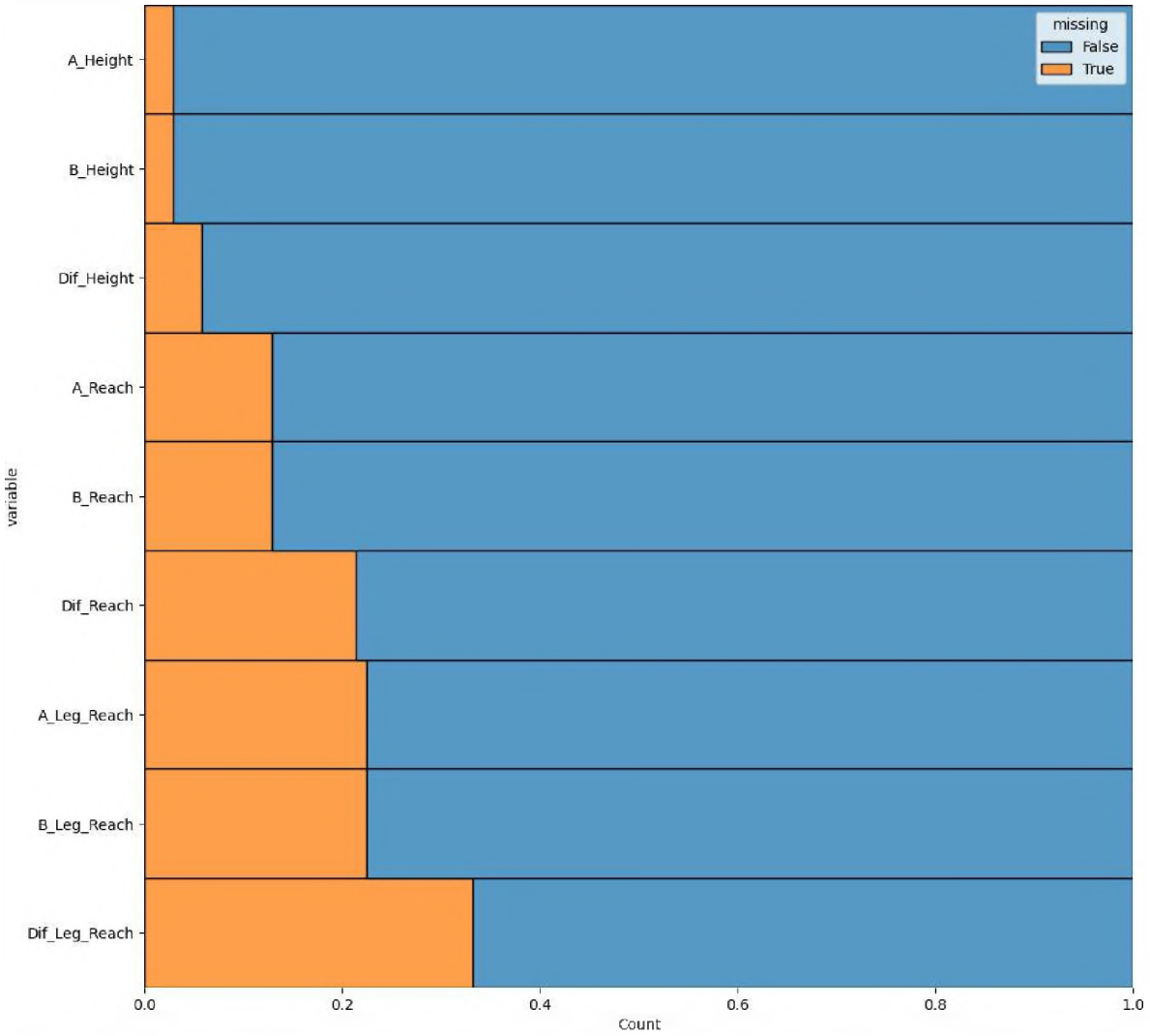
In [ ]:
```
fights['A_Height'] = fights.apply(lambda row: get_bio_data(row['Fighter_A'], 'Heigh
fights['B_Height'] = fights.apply(lambda row: get_bio_data(row['Fighter_B'], 'Heigh
fights['Dif_Height'] = fights['A_Height'] - fights['B_Height']
```

In [ ]:
```
fights['A_Reach'] = fights.apply(lambda row: get_bio_data(row['Fighter_A'], 'Reach'
fights['B_Reach'] = fights.apply(lambda row: get_bio_data(row['Fighter_B'], 'Reach'
fights['Dif_Reach'] = fights['A_Reach'] - fights['B_Reach']
```

In [ ]:
```
fights['A_Leg_Reach'] = fights.apply(lambda row: get_bio_data(row['Fighter_A'], 'Le
fights['B_Leg_Reach'] = fights.apply(lambda row: get_bio_data(row['Fighter_B'], 'Le
fights['Dif_Leg_Reach'] = fights['A_Leg_Reach'] - fights['B_Leg_Reach']
```

In [ ]:
```
#Identify columns with missing values
nothere = fights.isna().sum()
nothere = pd.DataFrame(nothere)
nothere = nothere.loc[nothere[0] > 0]
cols = nothere.index

f, ax = plt.subplots(figsize = (12,12))

sns.despine(f, left=True, bottom=True)
sns.histplot(
    data = fights[cols].isna().melt(value_name='missing'),
    y= 'variable',
    hue = 'missing',
    multiple = 'fill', ax = ax)
```

Out[ ]:
```
<AxesSubplot:xlabel='Count', ylabel='variable'>
```



In [ ]:
```
# drop rows missing height data
fights.dropna(subset=['A_Height', 'B_Height'], inplace=True)
len(fights)
```

Out[ ]: 8194

In [ ]:
```
# New column indicating if we have the reach data for fighter A
fights['A_Reach_NA'] = fights['A_Reach'].isna()

# New column indicating if we have the reach data for fighter B
fights['B_Reach_NA'] = fights['B_Reach'].isna()

# New column indicating if we have the reach data for either fighter
fights['Reach_NA'] = fights['A_Reach_NA'] | fights['B_Reach_NA']

# New column indicating if we have leg reach data for fighter A
fights['A_Leg_Reach_NA'] = fights['A_Leg_Reach'].isna()

# New column indicating if we have leg reach data for fighter B
fights['B_Leg_Reach_NA'] = fights['B_Leg_Reach'].isna()
```

```
# New column indicating if we have leg reach data for either fighter
fights['Leg_Reach_NA'] = fights['A_Leg_Reach_NA'] | fights['B_Leg_Reach_NA']
```

Add Weightclass

Weightclass data is in the All_Events_Fights_and_FightUrls file, which we created in the scraping notebook.

In [ ]:
```
all_events = pd.read_csv('data/final/events/All_Events_Fights_and_FightUrls.csv')
```

In [ ]:
```
all_events.head()
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	W/L	Weight class	Method	Round	Time	Fighter1	Fighter2	F1_K
0	0	0	win	Heavyweight	KO/TKO Punches	3	4:23	Ciryl Gane	Tai Tuivasa	
1	1	1	win	Middleweight	U-DEC	3	5:00	Robert Whittaker	Marvin Vettori	
2	2	2	win	Middleweight	U-DEC	3	5:00	Nassourdine Imavov	Joaquin Buckley	
3	3	3	win	Middleweight	KO/TKO Punches	3	1:09	Roman Kopylov	Alessio Di Chirico	
4	4	4	win	Featherweight	U-DEC	3	5:00	William Gomis	Jarno Errens	

In [ ]:
```
def find_typical_weightclass(fighter):
    try:
        data = all_events[all_events['Fighter1'] == fighter]
        data2 = all_events[all_events['Fighter2'] == fighter]
        data = pd.concat([data, data2])
        d = data['Weight class'].value_counts().index[0]
        return d
    except:
        return nan
```

In [ ]:
```
# test
weight = find_typical_weightclass('Khabib Nurmagomedov')
weight
```

Out[ ]:
```
'Lightweight'
```

In [ ]:
```
fights['A_Typical_Weightclass'] = fights.apply(lambda row: find_typical_weightclass
fights['B_Typical_Weightclass'] = fights.apply(lambda row: find_typical_weightclass
```

In [ ]:
```
fights.head(3)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

3 rows × 575 columns

In [ ]:
```
# save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V6.csv', index=False)
```

In [ ]:
```
def find_weightclass(fight_id):

    # use fight_id to find weightclass of fight

    try:
        fight_url = 'http://www.ufcstats.com/fight-details/' + str(fight_id)
        data = all_events[all_events['fight_link'] == fight_url]
        d = data['Weight class'].values[0]
        return d
    except:
        return nan
```

In [ ]:
```
# test
weight = find_weightclass('ff6c8dab41efcc09')
weight
```

Out[ ]:
```
'Heavyweight'
```

In [ ]:
```
fights['fight_weightclass'] = fights.apply(lambda row: find_weightclass(row['fight_
```

In [ ]:
```
fights.head(3)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

3 rows × 576 columns

In [ ]:
```
fights['A_Fight_in_Typical_Weightclass'] = fights.apply(lambda row: row['A_Typical_
fights['B_Fight_in_Typical_Weightclass'] = fights.apply(lambda row: row['B_Typical_
```

In [ ]:
```
fights.head(3)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

3 rows × 578 columns

In [ ]:
```
fights.to_csv('data/final/aggregates/Double_Fights_DF_V7.csv', index=False)
```

Feature: Method

The main methods of fight finishing are:

- Decision (DEC)
- KO/TKO
- Submission (SUB)

In [ ]:
```
all_events.head(2)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	W/L	Weight class	Method	Round	Time	Fighter1	Fighter2	F1_Kd
0	0	0	win	Heavyweight	KO/TKO Punches	3	4:23	Ciryl Gane	Tai Tuivasa	1
1	1	1	win	Middleweight	U-DEC	3	5:00	Robert Whittaker	Marvin Vettori	0

In [ ]:
```
all_events.Method.value_counts()[:20]
```

Out[ ]:
```
U-DEC                    2083
KO/TKO  Punches           673
KO/TKO  Punch             650
S-DEC                     586
SUB  Rear Naked Choke     421
SUB  Guillotine Choke     199
KO/TKO  Kick              147
SUB  Armbar               113
KO/TKO                     95
KO/TKO  Elbows             85
SUB  Arm Triangle          77
KO/TKO  Knee               71
M-DEC                      64
SUB  Triangle Choke        60
SUB  D'Arce Choke          31
KO/TKO  Elbow              30
SUB  Kimura                28
KO/TKO  Flying Knee        27
Overturned                 26
SUB  Anaconda Choke        23
Name: Method, dtype: int64
```

In [ ]:
```
# split Method into Method and Method Detail
all_events['Method_Primary'] = all_events['Method'].str.split('  ', expand=True)[0]
all_events['Method_Detail'] = all_events['Method'].str.split('  ', expand=True)[1]
all_events.head(3)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	W/L	Weight class	Method	Round	Time	Fighter1	Fighter2	F1_Kc
0	0	0	win	Heavyweight	KO/TKO Punches	3	4:23	Ciryl Gane	Tai Tuivasa	1
1	1	1	win	Middleweight	U-DEC	3	5:00	Robert Whittaker	Marvin Vettori	0
2	2	2	win	Middleweight	U-DEC	3	5:00	Nassourdine Imavov	Joaquin Buckley	0

3 rows × 22 columns

Get fight end method

In [ ]:
```
def get_method(fight_id):
    try:
        fight_url = 'http://www.ufcstats.com/fight-details/' + str(fight_id)
        data = all_events[all_events['fight_link'] == fight_url]
        d = data['Method_Primary'].values[0]
        return d
    except:
        return nan
```

In [ ]:
```
def get_details(fight_id):
    try:
```

```
        fight_url = 'http://www.ufcstats.com/fight-details/' + str(fight_id)
        data = all_events[all_events['fight_link'] == fight_url]
        d = data['Method_Detail'].values[0]
        return d
    except:
        return nan
```

In [ ]:
```
# test
method = get_method('ff6c8dab41efcc09')
method
```

Out[ ]:
```
'U-DEC'
```

In [ ]:
```
fights['InFightData__Method_Primary'] = fights.apply(lambda row: get_method(row['fi
fights['InFightData__Method_Detail'] = fights.apply(lambda row: get_details(row['fi
```

Feature: Round & Time

In [ ]:
```
def get_round_time(fight_id, round_or_time):
    try:
        fight_url = 'http://www.ufcstats.com/fight-details/' + str(fight_id)
        data = all_events[all_events['fight_link'] == fight_url]
        if round_or_time == 'round':
            d = data['Round'].values[0]
        elif round_or_time == 'time':
            d = data['Time'].values[0]
        return d
    except:
        return nan
```

In [ ]:
```
# test
round = get_round_time('ff6c8dab41efcc09', 'round')
round
```

Out[ ]:
```
3
```

In [ ]:
```
fights['InFightData__Round'] = fights.apply(lambda row: get_round_time(row['fight_i
fights['InFightData__Time'] = fights.apply(lambda row: get_round_time(row['fight_id
```

In [ ]:
```
fights.head(3)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

3 rows × 582 columns

In [ ]:
```
fights.to_csv('data/final/aggregates/Double_Fights_DF_V8.csv', index=False)
```

Dealing with Remaining Missing Values

For the height and reach, we can impute the median height and reach of the fighter's weightclass.

In [ ]:
```
# get missing values, sorted by column
missing_values = fights.isnull().sum().sort_values(ascending=False)
# get rid of 0 values
missing_values = missing_values[missing_values > 0]
# get percentage of missing values
missing_values = missing_values / len(fights) * 100
missing_values
```

Out[ ]:
```
InFightData__Method_Detail    51.598731
Dif_Leg_Reach                 29.118867
A_Leg_Reach                   20.234318
B_Leg_Reach                   20.234318
Dif_Reach                     16.597510
A_Reach                       10.361240
B_Reach                       10.361240
B_Typical_Weightclass          0.085428
A_Typical_Weightclass          0.085428
dtype: float64
```

In [ ]:
```
# replace nan values in "InFightData__Method_Detail" with "None"
fights['InFightData__Method_Detail'] = fights['InFightData__Method_Detail'].fillna(
```

In [ ]:
```
# replace nan values in Typical Weightclass with current weightclass
fights['A_Typical_Weightclass'] = fights.apply(lambda row: row['fight_weightclass']
fights['B_Typical_Weightclass'] = fights.apply(lambda row: row['fight_weightclass']
```

In [ ]:
```
# get median reach per height
reach_by_height = fights.groupby('A_Height')['A_Reach'].median()
reach_by_height
```

Out[ ]:
```
A_Height
0.0      NaN
60.0    60.0
61.0    62.0
61.5    62.0
62.0    64.0
62.5    62.5
63.0    64.5
63.5    65.5
64.0    64.0
64.5    68.0
65.0    66.0
66.0    67.0
66.5    66.0
67.0    69.0
67.5    69.0
68.0    70.0
68.5    71.0
69.0    71.0
70.0    72.0
70.5    76.0
71.0    73.0
71.5    73.0
72.0    74.0
72.5    78.0
73.0    75.0
73.5    77.0
74.0    75.5
74.5    78.0
75.0    77.0
75.5    76.0
76.0    80.0
77.0    79.0
78.0    81.0
78.5    79.0
79.0    80.0
80.0    80.0
84.0    84.5
Name: A_Reach, dtype: float64
```

In [ ]:
```
# replace nan values in reach with median reach per height
fights['A_Reach'] = fights.apply(lambda row: reach_by_height[row['A_Height']] if pd
fights['B_Reach'] = fights.apply(lambda row: reach_by_height[row['B_Height']] if pd
```

In [ ]:
```
# get median leg reach per height
leg_reach_by_height = fights.groupby('A_Height')['A_Leg_Reach'].median()
leg_reach_by_height
```

Out[ ]:
```
A_Height
0.0       NaN
60.0    34.00
61.0    35.00
61.5    35.00
62.0    36.00
62.5    38.00
63.0    36.00
63.5    38.00
64.0    37.00
64.5    38.00
65.0    37.50
66.0    38.00
66.5    38.00
67.0    38.00
67.5    37.25
68.0    39.00
68.5    38.50
69.0    39.50
70.0    40.00
70.5    42.00
71.0    40.50
71.5    43.00
72.0    41.00
72.5    40.00
73.0    41.50
73.5    41.00
74.0    42.00
74.5    44.00
75.0    43.50
75.5    43.00
76.0    44.50
77.0    46.00
78.0    44.50
78.5    45.50
79.0    47.50
80.0      NaN
84.0    44.00
Name: A_Leg_Reach, dtype: float64
```

In [ ]:
```
# replace nan values in leg reach with median leg reach per height
fights['A_Leg_Reach'] = fights.apply(lambda row: leg_reach_by_height[row['A_Height'
fights['B_Leg_Reach'] = fights.apply(lambda row: leg_reach_by_height[row['B_Height'
```

In [ ]:
```
# recalculate leg reach dif and reach dif
fights['A_Leg_Reach_Dif'] = fights['A_Leg_Reach'] - fights['B_Leg_Reach']
fights['A_Reach_Dif'] = fights['A_Reach'] - fights['B_Reach']
```

In [ ]:
```
fights.dtypes[-40:]
```

Out[ ]:
```
Dif_Rolling_Body_Strikes_att_std            float64
Dif_Rolling_Body_Strikes_percent_std        float64
Dif_Rolling_Leg_Strikes_land_std            float64
Dif_Rolling_Leg_Strikes_att_std             float64
Dif_Rolling_Leg_Strikes_percent_std         float64
Dif_Rolling_Distance_Strikes_land_std       float64
Dif_Rolling_Distance_Strikes_att_std        float64
Dif_Rolling_Distance_Strikes_percent_std    float64
Dif_Rolling_Clinch_Strikes_land_std         float64
Dif_Rolling_Clinch_Strikes_att_std          float64
Dif_Rolling_Clinch_Strikes_percent_std      float64
Dif_Rolling_Ground_Strikes_land_std         float64
Dif_Rolling_Ground_Strikes_att_std          float64
Dif_Rolling_Ground_Strikes_percent_std      float64
A_Height                                   float64
B_Height                                   float64
Dif_Height                                 float64
A_Reach                                    float64
B_Reach                                    float64
Dif_Reach                                  float64
A_Leg_Reach                                float64
B_Leg_Reach                                float64
Dif_Leg_Reach                              float64
A_Reach_NA                                    bool
B_Reach_NA                                    bool
Reach_NA                                      bool
A_Leg_Reach_NA                                bool
B_Leg_Reach_NA                                bool
Leg_Reach_NA                                  bool
A_Typical_Weightclass                       object
B_Typical_Weightclass                       object
fight_weightclass                           object
A_Fight_in_Typical_Weightclass                bool
B_Fight_in_Typical_Weightclass                bool
InFightData__Method_Primary                 object
InFightData__Method_Detail                  object
InFightData__Round                           int64
InFightData__Time                           object
A_Leg_Reach_Dif                            float64
A_Reach_Dif                                float64
dtype: object
```

In [ ]:
```
# get missing values, sorted by column
missing_values = fights.isnull().sum().sort_values(ascending=False)
# get rid of 0 values
missing_values = missing_values[missing_values > 0]
# get percentage of missing values
missing_values = missing_values / len(fights) * 100
missing_values
```

Out[ ]:
```
Dif_Leg_Reach      29.118867
Dif_Reach          16.597510
A_Leg_Reach_Dif     0.683427
A_Reach_Dif         0.610203
A_Leg_Reach         0.353918
B_Leg_Reach         0.353918
A_Reach             0.317305
B_Reach             0.317305
dtype: float64
```

In [ ]:
```
# get median reach per weightclass
reach_by_weightclass = fights.groupby('fight_weightclass')['A_Reach'].median()
reach_by_weightclass
```

Out[ ]:
```
fight_weightclass
Bantamweight             69.0
Catch Weight             71.0
Featherweight            71.0
Flyweight                66.0
Heavyweight              78.0
Light Heavyweight        76.0
Lightweight              71.0
Middleweight             75.0
Welterweight             74.0
Women's Bantamweight     67.5
Women's Featherweight    68.5
Women's Flyweight        66.5
Women's Strawweight      64.0
Name: A_Reach, dtype: float64
```

In [ ]:
```
# get median height per weightclass
height_by_weightclass = fights.groupby('fight_weightclass')['A_Height'].median()
height_by_weightclass
```

Out[ ]:
```
fight_weightclass
Bantamweight             67.0
Catch Weight             70.0
Featherweight            69.0
Flyweight                65.0
Heavyweight              75.0
Light Heavyweight        74.0
Lightweight              70.0
Middleweight             73.0
Welterweight             71.0
Women's Bantamweight     66.0
Women's Featherweight    67.0
Women's Flyweight        66.0
Women's Strawweight      64.0
Name: A_Height, dtype: float64
```

In [ ]:
```
# replace nan values in "A_Reach" with median for weight class
fights['A_Reach'] = fights.apply(lambda row: reach_by_weightclass[row['fight_weight
fights['B_Reach'] = fights.apply(lambda row: reach_by_weightclass[row['fight_weight

# replace nan values in "A_Height" with median for weight class
```

```
fights['A_Height'] = fights.apply(lambda row: height_by_weightclass[row['fight_weig
fights['B_Height'] = fights.apply(lambda row: height_by_weightclass[row['fight_weig
```

In [ ]:
```
# Once again, re-run the Difs
fights['Dif_Leg_Reach'] = fights['A_Leg_Reach'] - fights['B_Leg_Reach']
fights['Dif_Reach'] = fights['A_Reach'] - fights['B_Reach']
```

In [ ]:
```
# get missing
missing_values = fights.isnull().sum().sort_values(ascending=False)
missing_values = missing_values[missing_values > 0]
missing_values = missing_values / len(fights) * 100
missing_values
```

Out[ ]:
```
Dif_Leg_Reach      0.683427
A_Leg_Reach_Dif    0.683427
A_Reach_Dif        0.610203
B_Leg_Reach        0.353918
A_Leg_Reach        0.353918
dtype: float64
```

In [ ]:
```
# drop nan values
fights = fights.dropna()
```

In [ ]:
```
# drop A_Reach_Dif and A_Leg_Reach_Dif
fights = fights.drop(['A_Reach_Dif', 'A_Leg_Reach_Dif'], axis=1)
```

Feature: Size Ratios

In [ ]:
```
fights['A_Ape_Index'] = fights['A_Reach'] / fights['A_Height']
fights['B_Ape_Index'] = fights['B_Reach'] / fights['B_Height']

# Leg_Index = Leg_Reach / Height
fights['A_Leg_Index'] = fights['A_Leg_Reach'] / fights['A_Height']
fights['B_Leg_Index'] = fights['B_Leg_Reach'] / fights['B_Height']

# Leg_to_Wing_Index = Leg_Reach / Reach
fights['A_Leg_to_Wing_Index'] = fights['A_Leg_Reach'] / fights['A_Reach']
fights['B_Leg_to_Wing_Index'] = fights['B_Leg_Reach'] / fights['B_Reach']
```

In [ ]:
```
# save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V9.csv', index=False)
```

Feature: Winner

In [ ]:
```
# split Winner columns by "
fights['Winner'] = fights['Winner'].str.split('"').str[0].str.strip()
fights.head(2)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	

2 rows × 588 columns

In [ ]:
```
# add column "win?", if column winner is equal to fighter_A
fights['win?'] = fights.apply(lambda row: 1 if row['Winner'] == row['Fighter_A'] el
fights.head(30)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_S
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	
5	Alistair Overeem	Stefan Struve	0	0	17	25	2	
10	Nordine Taleb	Kyle Prepolec	0	0	90	201	52	
11	Matt Brown	Jordan Mein	0	1	40	84	24	
12	Beneil Dariush	Rashid Magomedov	0	0	60	120	41	
15	Andy Ogle	Cole Miller	0	0	27	51	21	
16	Urijah Faber	Michael McDonald	1	0	29	46	9	
20	Li Jingliang	Dhiego Lima	1	0	13	39	6	
22	Joe Lauzon	Marcin Held	0	0	20	62	23	
25	Chad Laprise	Thibault Gouti	2	0	17	35	4	
29	Pat Barry	Stefan Struve	0	0	30	45	31	
31	Alexandre Pantoja	Eric Shelton	0	0	37	99	32	
32	Holly Holm	Bethe Correia	1	0	25	53	15	
35	Katlyn Chookagian	Jessica Eye	0	0	65	201	81	
36	Chas Skelly	Maximo Blanco	0	0	1	1	0	
39	Anthony Smith	Jimmy Crute	0	0	26	39	18	
41	Jimi Manuwa	Corey Anderson	1	0	6	13	6	
42	John Teixeira	Hugo Viana	0	0	52	140	65	
43	Roy Nelson	Dave Herman	1	0	2	4	3	
48	Chris Camozzi	Tom Watson	0	0	87	146	82	
50	Cheick Kongo	Travis Browne	0	0	58	109	22	

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_S
52	Phil Hawes	Kyle Daukaus	0	0	66	101	28	
61	Thales Leites	Jack Hermansson	0	0	11	17	35	
63	Carlos Felipe	Justin Tafa	0	0	77	142	86	
65	Tim Elliott	Brandon Royval	0	0	17	32	16	
68	Thiago Moises	Kurt Holobaugh	0	0	66	102	39	
69	Raquel Pennington	Irene Aldana	0	0	47	137	52	
70	Warlley Alves	Colby Covington	0	0	2	4	2	

In [ ]:
```
# check number of wins vs losses
fights['win?'].value_counts()
```

Out[ ]:
```
1    4069
0    4069
Name: win?, dtype: int64
```

In [ ]:
```
fights.to_csv('data/final/aggregates/Double_Fights_DF_V10.csv', index=False)
```

Feature: Favorite

In [ ]:
```
# get columns in fight df with odds in the name
odds_columns = [col for col in fights.columns if 'Odds' in col]
odds_columns
```

Out[ ]:
```
['Fighter_A_Odds',
 'Fighter_B_Odds',
 'Fighter_A_Odds_Change',
 'Fighter_B_Odds_Change',
 'Dif_Odds']
```

In [ ]:
```
fights['favorite?'] = np.where(fights['Fighter_A_Odds'] < fights['Fighter_B_Odds'],
fights['favorite?'].value_counts()
```

Out[ ]:
```
0    4139
1    3999
Name: favorite?, dtype: int64
```

In [ ]:
```
fight_check_cols = ['Fighter_A', 'Fighter_B', 'Winner', 'win?', 'Fighter_A_Odds', '
fights[fight_check_cols].head(30)
```

Out[ ]:

	Fighter_A	Fighter_B	Winner	win?	Fighter_A_Odds	Fighter_B_Odds	favorite?
0	Holly Holm	Irene Aldana	Holly Holm	1	-125.0	105.0	1
3	Greg Hardy	Ben Sosoli	Ben Sosoli	0	-400.0	325.0	1
4	Jared Rosholt	Josh Copeland	Jared Rosholt	1	-310.0	280.0	1
5	Alistair Overeem	Stefan Struve	Alistair Overeem	1	-200.0	185.0	1
10	Nordine Taleb	Kyle Prepolec	Nordine Taleb	1	-420.0	335.0	1
11	Matt Brown	Jordan Mein	Matt Brown	1	290.0	-320.0	0
12	Beneil Dariush	Rashid Magomedov	Beneil Dariush	1	-115.0	-105.0	1
15	Andy Ogle	Cole Miller	Cole Miller	0	185.0	-200.0	0
16	Urijah Faber	Michael McDonald	Urijah Faber	1	-150.0	130.0	1
20	Li Jingliang	Dhiego Lima	Li Jingliang	1	130.0	-140.0	0
22	Joe Lauzon	Marcin Held	Joe Lauzon	1	-140.0	120.0	1
25	Chad Laprise	Thibault Gouti	Chad Laprise	1	-320.0	290.0	1
29	Pat Barry	Stefan Struve	Stefan Struve	0	-165.0	155.0	1
31	Alexandre Pantoja	Eric Shelton	Alexandre Pantoja	1	100.0	-120.0	0
32	Holly Holm	Bethe Correia	Holly Holm	1	-570.0	435.0	1
35	Katlyn Chookagian	Jessica Eye	Jessica Eye	0	-235.0	195.0	1
36	Chas Skelly	Maximo Blanco	Chas Skelly	1	-150.0	130.0	1
39	Anthony Smith	Jimmy Crute	Anthony Smith	1	210.0	-250.0	0
41	Jimi Manuwa	Corey Anderson	Jimi Manuwa	1	-145.0	125.0	1
42	John Teixeira	Hugo Viana	Hugo Viana	0	185.0	-200.0	0
43	Roy Nelson	Dave Herman	Roy Nelson	1	-162.0	152.0	1
48	Chris Camozzi	Tom Watson	Chris Camozzi	1	-135.0	115.0	1
50	Cheick Kongo	Travis Browne	Travis Browne	0	-178.0	167.0	1
52	Phil Hawes	Kyle Daukaus	Phil Hawes	1	132.0	-152.0	0
61	Thales Leites	Jack Hermansson	Jack Hermansson	0	110.0	-130.0	0

	Fighter_A	Fighter_B	Winner	win?	Fighter_A_Odds	Fighter_B_Odds	favorite?
63	Carlos Felipe	Justin Tafa	Carlos Felipe	1	-185.0	160.0	1
65	Tim Elliott	Brandon Royval	Brandon Royval	0	-138.0	118.0	1
68	Thiago Moises	Kurt Holobaugh	Thiago Moises	1	-125.0	105.0	1
69	Raquel Pennington	Irene Aldana	Raquel Pennington	1	115.0	-135.0	0
70	Warlley Alves	Colby Covington	Warlley Alves	1	105.0	-125.0	0

In [ ]:
```
# save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V11.csv', index=False)
```

Features: Win/Loss Details

In [ ]:
```
# get columns with date in the name
date_columns = [col for col in fights.columns if 'date' in col]
date_columns
```

Out[ ]:
```
['date']
```

In [ ]:
```
# add a formatted date column
fights['datetime'] = pd.to_datetime(fights['date'])
fights['date_formatted'] = fights['datetime'].dt.date
```

In [ ]:
```
def get_number_UFC_fights(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    return len(data)
```

In [ ]:
```
# test the function
get_number_UFC_fights('Conor McGregor', '2020-01-01')
```

Out[ ]:
```
10
```

In [ ]:
```
fights['A_Total_UFC_Fights'] = fights.apply(lambda row: get_number_UFC_fights(row['
fights['B_Total_UFC_Fights'] = fights.apply(lambda row: get_number_UFC_fights(row['

# add diff
fights['Dif_Total_UFC_Fights'] = fights['A_Total_UFC_Fights'] - fights['B_Total_UFC
```

In [ ]:
```
fights.head(3)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	
4	Jared Rosholt	Josh Copeland	0	0	22	45	9	

3 rows × 595 columns

Wins

In [ ]:
```
# add number of wins
def get_number_UFC_wins(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    return data['win?'].sum()
```

In [ ]:
```
fights['A_UFC_Wins'] = fights.apply(lambda row: get_number_UFC_wins(row['Fighter_A'
fights['B_UFC_Wins'] = fights.apply(lambda row: get_number_UFC_wins(row['Fighter_B'

# add diff
fights['Dif_UFC_Wins'] = fights['A_UFC_Wins'] - fights['B_UFC_Wins']
```

Losses

In [ ]:
```
def get_number_UFC_losses(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    return len(data) - data['win?'].sum()
```

In [ ]:
```
fights['A_UFC_Losses'] = fights.apply(lambda row: get_number_UFC_losses(row['Fighte
fights['B_UFC_Losses'] = fights.apply(lambda row: get_number_UFC_losses(row['Fighte

# add diff
fights['Dif_UFC_Losses'] = fights['A_UFC_Losses'] - fights['B_UFC_Losses']
```

W/L Percentages

In [ ]:
```
fights['A_UFC_Win_Percentage'] = fights['A_UFC_Wins'] / fights['A_Total_UFC_Fights'
fights['B_UFC_Win_Percentage'] = fights['B_UFC_Wins'] / fights['B_Total_UFC_Fights'

# add diff
fights['Dif_UFC_Win_Percentage'] = fights['A_UFC_Win_Percentage'] - fights['B_UFC_W
```

Last 5 Games Win Percentage

In [ ]:
```
def get_last5_win_percentage(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    data = data.sort_values(by='datetime', ascending=False)
    data = data.head(5)
    return data['win?'].sum() / len(data)
```

In [ ]:
```
fights['A_Last5_Win_Percentage'] = fights.apply(lambda row: get_last5_win_percentag
fights['B_Last5_Win_Percentage'] = fights.apply(lambda row: get_last5_win_percentag

# add diff
fights['Dif_Last5_Win_Percentage'] = fights['A_Last5_Win_Percentage'] - fights['B_L
```

Last 3 Win Percentage

In [ ]:
```
def get_last3_win_percentage(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    data = data.sort_values(by='datetime', ascending=False)
    data = data.head(3)
    return data['win?'].sum() / len(data)
```

In [ ]:
```
fights['A_Last3_Win_Percentage'] = fights.apply(lambda row: get_last3_win_percentag
fights['B_Last3_Win_Percentage'] = fights.apply(lambda row: get_last3_win_percentag

# add diff
fights['Dif_Last3_Win_Percentage'] = fights['A_Last3_Win_Percentage'] - fights['B_L
```

Win_by / Loss_by Features

In [ ]:
```
fights['InFightData__Method_Primary'].value_counts()
```

Out[ ]:
```
U-DEC          3018
KO/TKO         2584
SUB            1472
S-DEC           872
M-DEC            78
Overturned       64
CNC              30
DQ               18
Other             2
Name: InFightData__Method_Primary, dtype: int64
```

In [ ]:
```
def get_general_method(fight_method):
    if 'DEC' in fight_method:
        return 'DEC'
    elif 'SUB' in fight_method:
        return 'SUB'
    elif 'KO' in fight_method:
        return 'KO'
    else:
        return 'OTHER'
```

In [ ]:
```
# make general method column
fights['InFightData__General_Method'] = fights['InFightData__Method_Primary'].apply
```

In [ ]:
```
fights.head(2)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	

2 rows × 611 columns

Identify the method of win/loss for each fighter

In [ ]:
```
def get_win_by_ko_percent(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    wins = data[data['win?'] == 1]
    wins = wins[wins['InFightData__General_Method'] == 'KO']
    return data['win?'].sum() / len(data)

def get_loss_by_ko_percent(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    losses = data[data['win?'] == 0]
    losses = losses[losses['InFightData__General_Method'] == 'KO']
    return data['win?'].sum() / len(data)
```

In [ ]:
```
fights['A_Win_By_KO_Percentage'] = fights.apply(lambda row: get_win_by_ko_percent(r
fights['B_Win_By_KO_Percentage'] = fights.apply(lambda row: get_win_by_ko_percent(r

# add diff
fights['Dif_Win_By_KO_Percentage'] = fights['A_Win_By_KO_Percentage'] - fights['B_w
```

In [ ]:
```
fights['A_Loss_By_KO_Percentage'] = fights.apply(lambda row: get_loss_by_ko_percent
fights['B_Loss_By_KO_Percentage'] = fights.apply(lambda row: get_loss_by_ko_percent

# add diff
fights['Dif_Loss_By_KO_Percentage'] = fights['A_Loss_By_KO_Percentage'] - fights['B
```

Win/Loss by DEC

In [ ]:
```
def get_win_by_decision_percent(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    wins = data[data['win?'] == 1]
    wins = wins[wins['InFightData__General_Method'] == 'DEC']
    return data['win?'].sum() / len(data)
```

```
def get_loss_by_decision_percent(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    losses = data[data['win?'] == 0]
    losses = losses[losses['InFightData__General_Method'] == 'DEC']
    return data['win?'].sum() / len(data)
```

In [ ]:
```
fights['A_Win_By_Decision_Percentage'] = fights.apply(lambda row: get_win_by_decisi
fights['B_Win_By_Decision_Percentage'] = fights.apply(lambda row: get_win_by_decisi

# add diff
fights['Dif_Win_By_Decision_Percentage'] = fights['A_Win_By_Decision_Percentage'] -
```

In [ ]:
```
fights['A_Loss_By_Decision_Percentage'] = fights.apply(lambda row: get_loss_by_deci
fights['B_Loss_By_Decision_Percentage'] = fights.apply(lambda row: get_loss_by_deci

# add diff
fights['Dif_Loss_By_Decision_Percentage'] = fights['A_Loss_By_Decision_Percentage']
```

In [ ]:
```
# save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V12.csv', index=False)
```

Round & Time Statistics

In [ ]:
```
# find round columns
round_cols = [col for col in fights.columns if 'Round' in col]
round_cols
```

Out[ ]:
```
['InFightData__Round']
```

In [ ]:
```
time_cols = [col for col in fights.columns if 'Time' in col]
time_cols
```

Out[ ]:
```
['InFightData__Time']
```

In [ ]:
```
fights.InFightData__Round.dtypes
```

Out[ ]:
```
dtype('int64')
```

In [ ]:
```
fights.InFightData__Time
```

Out[ ]:
```
0       5:00
3       5:00
4       3:12
5       4:13
10      5:00
        ...
8345    1:10
8347    5:00
8351    5:00
8352    5:00
8356    2:37
Name: InFightData__Time, Length: 8138, dtype: object
```

I want time in seconds

In [ ]:
```
# convert time to seconds
fights['final_round_seconds'] = fights.InFightData__Time.apply(lambda x: int(x.spli
```

In [ ]:
```
fights['InFightData__Total_Fight_Time_Seconds'] = (fights['InFightData__Round'] - 1
```

Career Fight Time

In [ ]:
```
def get_career_fight_time_seconds(fighter, date):
    data = fights[fights['Fighter_A'] == fighter]
    data = data[data['datetime'] < date]
    return data['InFightData__Total_Fight_Time_Seconds'].sum()
```

In [ ]:
```
fights['A_UFC_Fight_Time_Seconds'] = fights.apply(lambda row: get_career_fight_time
fights['B_UFC_Fight_Time_Seconds'] = fights.apply(lambda row: get_career_fight_time

# add diff
fights['Dif_UFC_Fight_Time_Seconds'] = fights['A_UFC_Fight_Time_Seconds'] - fights[
```

Fix NANs

In [ ]:
```
missing = fights.isna().sum()
missing[missing > 0]
missing = missing[missing > 0]
missing
```

Out[ ]:
```
A_UFC_Win_Percentage              1613
B_UFC_Win_Percentage              1613
Dif_UFC_Win_Percentage            2624
A_Last5_Win_Percentage            1613
B_Last5_Win_Percentage            1613
Dif_Last5_Win_Percentage          2624
A_Last3_Win_Percentage            1613
B_Last3_Win_Percentage            1613
Dif_Last3_Win_Percentage          2624
A_Win_By_KO_Percentage            1613
B_Win_By_KO_Percentage            1613
Dif_Win_By_KO_Percentage          2624
A_Loss_By_KO_Percentage           1613
B_Loss_By_KO_Percentage           1613
Dif_Loss_By_KO_Percentage         2624
A_Win_By_Decision_Percentage      1613
B_Win_By_Decision_Percentage      1613
Dif_Win_By_Decision_Percentage    2624
A_Loss_By_Decision_Percentage     1613
B_Loss_By_Decision_Percentage     1613
Dif_Loss_By_Decision_Percentage   2624
dtype: int64
```

In [ ]:
```
# fill nans with 0
fights = fights.fillna(0)
```

In [ ]:
```
fights.to_csv('data/final/aggregates/Double_Fights_DF_V13.csv', index=False)
```

Stats / Round Features

For these, use 1) Top down, and 2) Opponent averages.

First, we have to identify the correct columns to work with.

In [ ]:
```
topdown_cols = [n for n in fights.columns.to_list() if 'topdown' in n]
# A TOPDOWN
a_topdown_cols = [n for n in topdown_cols if n.startswith('A_')]
# get rid of any percent columns
a_topdown_cols = [n for n in a_topdown_cols if 'percent' not in n]

# B TOPDOWN
b_topdown_cols = [n for n in topdown_cols if n.startswith('B_')]
# get rid of any percent columns
b_topdown_cols = [n for n in b_topdown_cols if 'percent' not in n]
```

In [ ]:
```
# get total number of rounds in ufc
fights['A_UFC_Fight_Rounds'] = fights['A_UFC_Fight_Time_Seconds'] / 300
fights['B_UFC_Fight_Rounds'] = fights['B_UFC_Fight_Time_Seconds'] / 300
```

In [ ]:
```
# add per-round averages using topdown columns
for col in a_topdown_cols:
    fights[f'{col}_per_round'] = fights[col] / fights['A_UFC_Fight_Rounds']
```

```
for col in b_topdown_cols:
    fights[f'{col}_per_round'] = fights[col] / fights['B_UFC_Fight_Rounds']
```

In [ ]:
```
fights.head(2)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	

2 rows × 680 columns

In [ ]:
```
fights.columns.to_list()
```

Out[ ]:
```
['Fighter_A',
 'Fighter_B',
 'A_Kd',
 'B_Kd',
 'A_Sig_strike_land',
 'A_Sig_strike_att',
 'B_Sig_strike_land',
 'B_Sig_strike_att',
 'A_Sig_strike_percent',
 'B_Sig_strike_percent',
 'A_Total_Strikes_land',
 'A_Total_Strikes_att',
 'B_Total_Strikes_land',
 'B_Total_Strikes_att',
 'A_Total_Strikes_percent',
 'B_Total_Strikes_percent',
 'A_Takedowns_land',
 'A_Takedowns_att',
 'B_Takedowns_land',
 'B_Takedowns_att',
 'A_Takedown_percent',
 'B_Takedown_percent',
 'A_Sub_Attempts_land',
 'A_Sub_Attempts_att',
 'B_Sub_Attempts_land',
 'B_Sub_Attempts_att',
 'A_Rev',
 'B_Rev',
 'A_Ctrl_time_min',
 'A_Ctrl_time_sec',
 'B_Ctrl_time_min',
 'B_Ctrl_time_sec',
 'A_Ctrl_time_tot',
 'B_Ctrl_time_tot',
 'details',
 'event_title',
 'event_url',
 'date',
 'Winner',
 'fight_id',
 'A_Head_Strikes_land',
 'A_Head_Strikes_att',
 'B_Head_Strikes_land',
 'B_Head_Strikes_att',
 'A_Head_Strikes_percent',
 'B_Head_Strikes_percent',
 'A_Body_Strikes_land',
 'A_Body_Strikes_att',
 'B_Body_Strikes_land',
 'B_Body_Strikes_att',
 'A_Body_Strikes_percent',
 'B_Body_Strikes_percent',
 'A_Leg_Strikes_land',
 'A_Leg_Strikes_att',
 'B_Leg_Strikes_land',
 'B_Leg_Strikes_att',
```

```
'A_Leg_Strikes_percent',
'B_Leg_Strikes_percent',
'A_Distance_Strikes_land',
'A_Distance_Strikes_att',
'B_Distance_Strikes_land',
'B_Distance_Strikes_att',
'A_Distance_Strikes_percent',
'B_Distance_Strikes_percent',
'A_Clinch_Strikes_land',
'A_Clinch_Strikes_att',
'B_Clinch_Strikes_land',
'B_Clinch_Strikes_att',
'A_Clinch_Strikes_percent',
'B_Clinch_Strikes_percent',
'A_Ground_Strikes_land',
'A_Ground_Strikes_att',
'B_Ground_Strikes_land',
'B_Ground_Strikes_att',
'A_Ground_Strikes_percent',
'B_Ground_Strikes_percent',
'event_code',
'Fighter_A_Odds',
'Fighter_B_Odds',
'Fighter_A_Odds_Change',
'Fighter_B_Odds_Change',
'Dif_Kd',
'Dif_Sig_strike_land',
'Dif_Sig_strike_att',
'Dif_Sig_strike_percent',
'Dif_Total_Strikes_land',
'Dif_Total_Strikes_att',
'Dif_Total_Strikes_percent',
'Dif_Takedowns_land',
'Dif_Takedowns_att',
'Dif_Takedown_percent',
'Dif_Sub_Attempts_land',
'Dif_Sub_Attempts_att',
'Dif_Rev',
'Dif_Ctrl_time_min',
'Dif_Ctrl_time_sec',
'Dif_Ctrl_time_tot',
'Dif_Head_Strikes_land',
'Dif_Head_Strikes_att',
'Dif_Head_Strikes_percent',
'Dif_Body_Strikes_land',
'Dif_Body_Strikes_att',
'Dif_Body_Strikes_percent',
'Dif_Leg_Strikes_land',
'Dif_Leg_Strikes_att',
'Dif_Leg_Strikes_percent',
'Dif_Distance_Strikes_land',
'Dif_Distance_Strikes_att',
'Dif_Distance_Strikes_percent',
'Dif_Clinch_Strikes_land',
'Dif_Clinch_Strikes_att',
'Dif_Clinch_Strikes_percent',
```

```
'Dif_Ground_Strikes_land',
'Dif_Ground_Strikes_att',
'Dif_Ground_Strikes_percent',
'Dif_Odds',
'A_Rolling_Kd_mean',
'B_Rolling_Kd_mean',
'A_Rolling_Kd_std',
'B_Rolling_Kd_std',
'A_Rolling_Kd_median',
'B_Rolling_Kd_median',
'A_Rolling_Sig_strike_land_mean',
'B_Rolling_Sig_strike_land_mean',
'A_Rolling_Sig_strike_land_std',
'B_Rolling_Sig_strike_land_std',
'A_Rolling_Sig_strike_land_median',
'B_Rolling_Sig_strike_land_median',
'A_Rolling_Sig_strike_att_mean',
'B_Rolling_Sig_strike_att_mean',
'A_Rolling_Sig_strike_att_std',
'B_Rolling_Sig_strike_att_std',
'A_Rolling_Sig_strike_att_median',
'B_Rolling_Sig_strike_att_median',
'A_Rolling_Sig_strike_percent_mean',
'B_Rolling_Sig_strike_percent_mean',
'A_Rolling_Sig_strike_percent_std',
'B_Rolling_Sig_strike_percent_std',
'A_Rolling_Sig_strike_percent_median',
'B_Rolling_Sig_strike_percent_median',
'A_Rolling_Total_Strikes_land_mean',
'B_Rolling_Total_Strikes_land_mean',
'A_Rolling_Total_Strikes_land_std',
'B_Rolling_Total_Strikes_land_std',
'A_Rolling_Total_Strikes_land_median',
'B_Rolling_Total_Strikes_land_median',
'A_Rolling_Total_Strikes_att_mean',
'B_Rolling_Total_Strikes_att_mean',
'A_Rolling_Total_Strikes_att_std',
'B_Rolling_Total_Strikes_att_std',
'A_Rolling_Total_Strikes_att_median',
'B_Rolling_Total_Strikes_att_median',
'A_Rolling_Total_Strikes_percent_mean',
'B_Rolling_Total_Strikes_percent_mean',
'A_Rolling_Total_Strikes_percent_std',
'B_Rolling_Total_Strikes_percent_std',
'A_Rolling_Total_Strikes_percent_median',
'B_Rolling_Total_Strikes_percent_median',
'A_Rolling_Takedowns_land_mean',
'B_Rolling_Takedowns_land_mean',
'A_Rolling_Takedowns_land_std',
'B_Rolling_Takedowns_land_std',
'A_Rolling_Takedowns_land_median',
'B_Rolling_Takedowns_land_median',
'A_Rolling_Takedowns_att_mean',
'B_Rolling_Takedowns_att_mean',
'A_Rolling_Takedowns_att_std',
'B_Rolling_Takedowns_att_std',
```

```
'A_Rolling_Takedowns_att_median',
'B_Rolling_Takedowns_att_median',
'A_Rolling_Takedown_percent_mean',
'B_Rolling_Takedown_percent_mean',
'A_Rolling_Takedown_percent_std',
'B_Rolling_Takedown_percent_std',
'A_Rolling_Takedown_percent_median',
'B_Rolling_Takedown_percent_median',
'A_Rolling_Sub_Attempts_land_mean',
'B_Rolling_Sub_Attempts_land_mean',
'A_Rolling_Sub_Attempts_land_std',
'B_Rolling_Sub_Attempts_land_std',
'A_Rolling_Sub_Attempts_land_median',
'B_Rolling_Sub_Attempts_land_median',
'A_Rolling_Sub_Attempts_att_mean',
'B_Rolling_Sub_Attempts_att_mean',
'A_Rolling_Sub_Attempts_att_std',
'B_Rolling_Sub_Attempts_att_std',
'A_Rolling_Sub_Attempts_att_median',
'B_Rolling_Sub_Attempts_att_median',
'A_Rolling_Rev_mean',
'B_Rolling_Rev_mean',
'A_Rolling_Rev_std',
'B_Rolling_Rev_std',
'A_Rolling_Rev_median',
'B_Rolling_Rev_median',
'A_Rolling_Ctrl_time_min_mean',
'B_Rolling_Ctrl_time_min_mean',
'A_Rolling_Ctrl_time_min_std',
'B_Rolling_Ctrl_time_min_std',
'A_Rolling_Ctrl_time_min_median',
'B_Rolling_Ctrl_time_min_median',
'A_Rolling_Ctrl_time_sec_mean',
'B_Rolling_Ctrl_time_sec_mean',
'A_Rolling_Ctrl_time_sec_std',
'B_Rolling_Ctrl_time_sec_std',
'A_Rolling_Ctrl_time_sec_median',
'B_Rolling_Ctrl_time_sec_median',
'A_Rolling_Ctrl_time_tot_mean',
'B_Rolling_Ctrl_time_tot_mean',
'A_Rolling_Ctrl_time_tot_std',
'B_Rolling_Ctrl_time_tot_std',
'A_Rolling_Ctrl_time_tot_median',
'B_Rolling_Ctrl_time_tot_median',
'A_Rolling_Head_Strikes_land_mean',
'B_Rolling_Head_Strikes_land_mean',
'A_Rolling_Head_Strikes_land_std',
'B_Rolling_Head_Strikes_land_std',
'A_Rolling_Head_Strikes_land_median',
'B_Rolling_Head_Strikes_land_median',
'A_Rolling_Head_Strikes_att_mean',
'B_Rolling_Head_Strikes_att_mean',
'A_Rolling_Head_Strikes_att_std',
'B_Rolling_Head_Strikes_att_std',
'A_Rolling_Head_Strikes_att_median',
'B_Rolling_Head_Strikes_att_median',
```

```
'A_Rolling_Head_Strikes_percent_mean',
'B_Rolling_Head_Strikes_percent_mean',
'A_Rolling_Head_Strikes_percent_std',
'B_Rolling_Head_Strikes_percent_std',
'A_Rolling_Head_Strikes_percent_median',
'B_Rolling_Head_Strikes_percent_median',
'A_Rolling_Body_Strikes_land_mean',
'B_Rolling_Body_Strikes_land_mean',
'A_Rolling_Body_Strikes_land_std',
'B_Rolling_Body_Strikes_land_std',
'A_Rolling_Body_Strikes_land_median',
'B_Rolling_Body_Strikes_land_median',
'A_Rolling_Body_Strikes_att_mean',
'B_Rolling_Body_Strikes_att_mean',
'A_Rolling_Body_Strikes_att_std',
'B_Rolling_Body_Strikes_att_std',
'A_Rolling_Body_Strikes_att_median',
'B_Rolling_Body_Strikes_att_median',
'A_Rolling_Body_Strikes_percent_mean',
'B_Rolling_Body_Strikes_percent_mean',
'A_Rolling_Body_Strikes_percent_std',
'B_Rolling_Body_Strikes_percent_std',
'A_Rolling_Body_Strikes_percent_median',
'B_Rolling_Body_Strikes_percent_median',
'A_Rolling_Leg_Strikes_land_mean',
'B_Rolling_Leg_Strikes_land_mean',
'A_Rolling_Leg_Strikes_land_std',
'B_Rolling_Leg_Strikes_land_std',
'A_Rolling_Leg_Strikes_land_median',
'B_Rolling_Leg_Strikes_land_median',
'A_Rolling_Leg_Strikes_att_mean',
'B_Rolling_Leg_Strikes_att_mean',
'A_Rolling_Leg_Strikes_att_std',
'B_Rolling_Leg_Strikes_att_std',
'A_Rolling_Leg_Strikes_att_median',
'B_Rolling_Leg_Strikes_att_median',
'A_Rolling_Leg_Strikes_percent_mean',
'B_Rolling_Leg_Strikes_percent_mean',
'A_Rolling_Leg_Strikes_percent_std',
'B_Rolling_Leg_Strikes_percent_std',
'A_Rolling_Leg_Strikes_percent_median',
'B_Rolling_Leg_Strikes_percent_median',
'A_Rolling_Distance_Strikes_land_mean',
'B_Rolling_Distance_Strikes_land_mean',
'A_Rolling_Distance_Strikes_land_std',
'B_Rolling_Distance_Strikes_land_std',
'A_Rolling_Distance_Strikes_land_median',
'B_Rolling_Distance_Strikes_land_median',
'A_Rolling_Distance_Strikes_att_mean',
'B_Rolling_Distance_Strikes_att_mean',
'A_Rolling_Distance_Strikes_att_std',
'B_Rolling_Distance_Strikes_att_std',
'A_Rolling_Distance_Strikes_att_median',
'B_Rolling_Distance_Strikes_att_median',
'A_Rolling_Distance_Strikes_percent_mean',
'B_Rolling_Distance_Strikes_percent_mean',
```

```
'A_Rolling_Distance_Strikes_percent_std',
'B_Rolling_Distance_Strikes_percent_std',
'A_Rolling_Distance_Strikes_percent_median',
'B_Rolling_Distance_Strikes_percent_median',
'A_Rolling_Clinch_Strikes_land_mean',
'B_Rolling_Clinch_Strikes_land_mean',
'A_Rolling_Clinch_Strikes_land_std',
'B_Rolling_Clinch_Strikes_land_std',
'A_Rolling_Clinch_Strikes_land_median',
'B_Rolling_Clinch_Strikes_land_median',
'A_Rolling_Clinch_Strikes_att_mean',
'B_Rolling_Clinch_Strikes_att_mean',
'A_Rolling_Clinch_Strikes_att_std',
'B_Rolling_Clinch_Strikes_att_std',
'A_Rolling_Clinch_Strikes_att_median',
'B_Rolling_Clinch_Strikes_att_median',
'A_Rolling_Clinch_Strikes_percent_mean',
'B_Rolling_Clinch_Strikes_percent_mean',
'A_Rolling_Clinch_Strikes_percent_std',
'B_Rolling_Clinch_Strikes_percent_std',
'A_Rolling_Clinch_Strikes_percent_median',
'B_Rolling_Clinch_Strikes_percent_median',
'A_Rolling_Ground_Strikes_land_mean',
'B_Rolling_Ground_Strikes_land_mean',
'A_Rolling_Ground_Strikes_land_std',
'B_Rolling_Ground_Strikes_land_std',
'A_Rolling_Ground_Strikes_land_median',
'B_Rolling_Ground_Strikes_land_median',
'A_Rolling_Ground_Strikes_att_mean',
'B_Rolling_Ground_Strikes_att_mean',
'A_Rolling_Ground_Strikes_att_std',
'B_Rolling_Ground_Strikes_att_std',
'A_Rolling_Ground_Strikes_att_median',
'B_Rolling_Ground_Strikes_att_median',
'A_Rolling_Ground_Strikes_percent_mean',
'B_Rolling_Ground_Strikes_percent_mean',
'A_Rolling_Ground_Strikes_percent_std',
'B_Rolling_Ground_Strikes_percent_std',
'A_Rolling_Ground_Strikes_percent_median',
'B_Rolling_Ground_Strikes_percent_median',
'A_topdown_Avg_Kd',
'B_topdown_Avg_Kd',
'A_topdown_Avg_Sig_strike_land',
'B_topdown_Avg_Sig_strike_land',
'A_topdown_Avg_Sig_strike_att',
'B_topdown_Avg_Sig_strike_att',
'A_topdown_Avg_Sig_strike_percent',
'B_topdown_Avg_Sig_strike_percent',
'A_topdown_Avg_Total_Strikes_land',
'B_topdown_Avg_Total_Strikes_land',
'A_topdown_Avg_Total_Strikes_att',
'B_topdown_Avg_Total_Strikes_att',
'A_topdown_Avg_Total_Strikes_percent',
'B_topdown_Avg_Total_Strikes_percent',
'A_topdown_Avg_Takedowns_land',
'B_topdown_Avg_Takedowns_land',
```

```
'A_topdown_Avg_Takedowns_att',
'B_topdown_Avg_Takedowns_att',
'A_topdown_Avg_Takedown_percent',
'B_topdown_Avg_Takedown_percent',
'A_topdown_Avg_Sub_Attempts_land',
'B_topdown_Avg_Sub_Attempts_land',
'A_topdown_Avg_Sub_Attempts_att',
'B_topdown_Avg_Sub_Attempts_att',
'A_topdown_Avg_Rev',
'B_topdown_Avg_Rev',
'A_topdown_Avg_Ctrl_time_min',
'B_topdown_Avg_Ctrl_time_min',
'A_topdown_Avg_Ctrl_time_sec',
'B_topdown_Avg_Ctrl_time_sec',
'A_topdown_Avg_Ctrl_time_tot',
'B_topdown_Avg_Ctrl_time_tot',
'A_topdown_Avg_Head_Strikes_land',
'B_topdown_Avg_Head_Strikes_land',
'A_topdown_Avg_Head_Strikes_att',
'B_topdown_Avg_Head_Strikes_att',
'A_topdown_Avg_Head_Strikes_percent',
'B_topdown_Avg_Head_Strikes_percent',
'A_topdown_Avg_Body_Strikes_land',
'B_topdown_Avg_Body_Strikes_land',
'A_topdown_Avg_Body_Strikes_att',
'B_topdown_Avg_Body_Strikes_att',
'A_topdown_Avg_Body_Strikes_percent',
'B_topdown_Avg_Body_Strikes_percent',
'A_topdown_Avg_Leg_Strikes_land',
'B_topdown_Avg_Leg_Strikes_land',
'A_topdown_Avg_Leg_Strikes_att',
'B_topdown_Avg_Leg_Strikes_att',
'A_topdown_Avg_Leg_Strikes_percent',
'B_topdown_Avg_Leg_Strikes_percent',
'A_topdown_Avg_Distance_Strikes_land',
'B_topdown_Avg_Distance_Strikes_land',
'A_topdown_Avg_Distance_Strikes_att',
'B_topdown_Avg_Distance_Strikes_att',
'A_topdown_Avg_Distance_Strikes_percent',
'B_topdown_Avg_Distance_Strikes_percent',
'A_topdown_Avg_Clinch_Strikes_land',
'B_topdown_Avg_Clinch_Strikes_land',
'A_topdown_Avg_Clinch_Strikes_att',
'B_topdown_Avg_Clinch_Strikes_att',
'A_topdown_Avg_Clinch_Strikes_percent',
'B_topdown_Avg_Clinch_Strikes_percent',
'A_topdown_Avg_Ground_Strikes_land',
'B_topdown_Avg_Ground_Strikes_land',
'A_topdown_Avg_Ground_Strikes_att',
'B_topdown_Avg_Ground_Strikes_att',
'A_topdown_Avg_Ground_Strikes_percent',
'B_topdown_Avg_Ground_Strikes_percent',
'A_Opp_Avg_Kd',
'B_Opp_Avg_Kd',
'A_Opp_Avg_Sig_strike_land',
'B_Opp_Avg_Sig_strike_land',
```

```
'A_Opp_Avg_Sig_strike_att',
'B_Opp_Avg_Sig_strike_att',
'A_Opp_Avg_Sig_strike_percent',
'B_Opp_Avg_Sig_strike_percent',
'A_Opp_Avg_Total_Strikes_land',
'B_Opp_Avg_Total_Strikes_land',
'A_Opp_Avg_Total_Strikes_att',
'B_Opp_Avg_Total_Strikes_att',
'A_Opp_Avg_Total_Strikes_percent',
'B_Opp_Avg_Total_Strikes_percent',
'A_Opp_Avg_Takedowns_land',
'B_Opp_Avg_Takedowns_land',
'A_Opp_Avg_Takedowns_att',
'B_Opp_Avg_Takedowns_att',
'A_Opp_Avg_Takedown_percent',
'B_Opp_Avg_Takedown_percent',
'A_Opp_Avg_Sub_Attempts_land',
'B_Opp_Avg_Sub_Attempts_land',
'A_Opp_Avg_Sub_Attempts_att',
'B_Opp_Avg_Sub_Attempts_att',
'A_Opp_Avg_Rev',
'B_Opp_Avg_Rev',
'A_Opp_Avg_Ctrl_time_min',
'B_Opp_Avg_Ctrl_time_min',
'A_Opp_Avg_Ctrl_time_sec',
'B_Opp_Avg_Ctrl_time_sec',
'A_Opp_Avg_Ctrl_time_tot',
'B_Opp_Avg_Ctrl_time_tot',
'A_Opp_Avg_Head_Strikes_land',
'B_Opp_Avg_Head_Strikes_land',
'A_Opp_Avg_Head_Strikes_att',
'B_Opp_Avg_Head_Strikes_att',
'A_Opp_Avg_Head_Strikes_percent',
'B_Opp_Avg_Head_Strikes_percent',
'A_Opp_Avg_Body_Strikes_land',
'B_Opp_Avg_Body_Strikes_land',
'A_Opp_Avg_Body_Strikes_att',
'B_Opp_Avg_Body_Strikes_att',
'A_Opp_Avg_Body_Strikes_percent',
'B_Opp_Avg_Body_Strikes_percent',
'A_Opp_Avg_Leg_Strikes_land',
'B_Opp_Avg_Leg_Strikes_land',
'A_Opp_Avg_Leg_Strikes_att',
'B_Opp_Avg_Leg_Strikes_att',
'A_Opp_Avg_Leg_Strikes_percent',
'B_Opp_Avg_Leg_Strikes_percent',
'A_Opp_Avg_Distance_Strikes_land',
'B_Opp_Avg_Distance_Strikes_land',
'A_Opp_Avg_Distance_Strikes_att',
'B_Opp_Avg_Distance_Strikes_att',
'A_Opp_Avg_Distance_Strikes_percent',
'B_Opp_Avg_Distance_Strikes_percent',
'A_Opp_Avg_Clinch_Strikes_land',
'B_Opp_Avg_Clinch_Strikes_land',
'A_Opp_Avg_Clinch_Strikes_att',
'B_Opp_Avg_Clinch_Strikes_att',
```

```
'A_Opp_Avg_Clinch_Strikes_percent',
'B_Opp_Avg_Clinch_Strikes_percent',
'A_Opp_Avg_Ground_Strikes_land',
'B_Opp_Avg_Ground_Strikes_land',
'A_Opp_Avg_Ground_Strikes_att',
'B_Opp_Avg_Ground_Strikes_att',
'A_Opp_Avg_Ground_Strikes_percent',
'B_Opp_Avg_Ground_Strikes_percent',
'Dif_Rolling_Kd_mean',
'Dif_Rolling_Sig_strike_land_mean',
'Dif_Rolling_Sig_strike_att_mean',
'Dif_Rolling_Sig_strike_percent_mean',
'Dif_Rolling_Total_Strikes_land_mean',
'Dif_Rolling_Total_Strikes_att_mean',
'Dif_Rolling_Total_Strikes_percent_mean',
'Dif_Rolling_Takedowns_land_mean',
'Dif_Rolling_Takedowns_att_mean',
'Dif_Rolling_Takedown_percent_mean',
'Dif_Rolling_Sub_Attempts_land_mean',
'Dif_Rolling_Sub_Attempts_att_mean',
'Dif_Rolling_Rev_mean',
'Dif_Rolling_Ctrl_time_min_mean',
'Dif_Rolling_Ctrl_time_sec_mean',
'Dif_Rolling_Ctrl_time_tot_mean',
'Dif_Rolling_Head_Strikes_land_mean',
'Dif_Rolling_Head_Strikes_att_mean',
'Dif_Rolling_Head_Strikes_percent_mean',
'Dif_Rolling_Body_Strikes_land_mean',
'Dif_Rolling_Body_Strikes_att_mean',
'Dif_Rolling_Body_Strikes_percent_mean',
'Dif_Rolling_Leg_Strikes_land_mean',
'Dif_Rolling_Leg_Strikes_att_mean',
'Dif_Rolling_Leg_Strikes_percent_mean',
'Dif_Rolling_Distance_Strikes_land_mean',
'Dif_Rolling_Distance_Strikes_att_mean',
'Dif_Rolling_Distance_Strikes_percent_mean',
'Dif_Rolling_Clinch_Strikes_land_mean',
'Dif_Rolling_Clinch_Strikes_att_mean',
'Dif_Rolling_Clinch_Strikes_percent_mean',
'Dif_Rolling_Ground_Strikes_land_mean',
'Dif_Rolling_Ground_Strikes_att_mean',
'Dif_Rolling_Ground_Strikes_percent_mean',
'Dif_Rolling_Kd_median',
'Dif_Rolling_Sig_strike_land_median',
'Dif_Rolling_Sig_strike_att_median',
'Dif_Rolling_Sig_strike_percent_median',
'Dif_Rolling_Total_Strikes_land_median',
'Dif_Rolling_Total_Strikes_att_median',
'Dif_Rolling_Total_Strikes_percent_median',
'Dif_Rolling_Takedowns_land_median',
'Dif_Rolling_Takedowns_att_median',
'Dif_Rolling_Takedown_percent_median',
'Dif_Rolling_Sub_Attempts_land_median',
'Dif_Rolling_Sub_Attempts_att_median',
'Dif_Rolling_Rev_median',
'Dif_Rolling_Ctrl_time_min_median',
```

```
'Dif_Rolling_Ctrl_time_sec_median',
'Dif_Rolling_Ctrl_time_tot_median',
'Dif_Rolling_Head_Strikes_land_median',
'Dif_Rolling_Head_Strikes_att_median',
'Dif_Rolling_Head_Strikes_percent_median',
'Dif_Rolling_Body_Strikes_land_median',
'Dif_Rolling_Body_Strikes_att_median',
'Dif_Rolling_Body_Strikes_percent_median',
'Dif_Rolling_Leg_Strikes_land_median',
'Dif_Rolling_Leg_Strikes_att_median',
'Dif_Rolling_Leg_Strikes_percent_median',
'Dif_Rolling_Distance_Strikes_land_median',
'Dif_Rolling_Distance_Strikes_att_median',
'Dif_Rolling_Distance_Strikes_percent_median',
'Dif_Rolling_Clinch_Strikes_land_median',
'Dif_Rolling_Clinch_Strikes_att_median',
'Dif_Rolling_Clinch_Strikes_percent_median',
'Dif_Rolling_Ground_Strikes_land_median',
'Dif_Rolling_Ground_Strikes_att_median',
'Dif_Rolling_Ground_Strikes_percent_median',
'Dif_Rolling_Kd_std',
'Dif_Rolling_Sig_strike_land_std',
'Dif_Rolling_Sig_strike_att_std',
'Dif_Rolling_Sig_strike_percent_std',
'Dif_Rolling_Total_Strikes_land_std',
'Dif_Rolling_Total_Strikes_att_std',
'Dif_Rolling_Total_Strikes_percent_std',
'Dif_Rolling_Takedowns_land_std',
'Dif_Rolling_Takedowns_att_std',
'Dif_Rolling_Takedown_percent_std',
'Dif_Rolling_Sub_Attempts_land_std',
'Dif_Rolling_Sub_Attempts_att_std',
'Dif_Rolling_Rev_std',
'Dif_Rolling_Ctrl_time_min_std',
'Dif_Rolling_Ctrl_time_sec_std',
'Dif_Rolling_Ctrl_time_tot_std',
'Dif_Rolling_Head_Strikes_land_std',
'Dif_Rolling_Head_Strikes_att_std',
'Dif_Rolling_Head_Strikes_percent_std',
'Dif_Rolling_Body_Strikes_land_std',
'Dif_Rolling_Body_Strikes_att_std',
'Dif_Rolling_Body_Strikes_percent_std',
'Dif_Rolling_Leg_Strikes_land_std',
'Dif_Rolling_Leg_Strikes_att_std',
'Dif_Rolling_Leg_Strikes_percent_std',
'Dif_Rolling_Distance_Strikes_land_std',
'Dif_Rolling_Distance_Strikes_att_std',
'Dif_Rolling_Distance_Strikes_percent_std',
'Dif_Rolling_Clinch_Strikes_land_std',
'Dif_Rolling_Clinch_Strikes_att_std',
'Dif_Rolling_Clinch_Strikes_percent_std',
'Dif_Rolling_Ground_Strikes_land_std',
'Dif_Rolling_Ground_Strikes_att_std',
'Dif_Rolling_Ground_Strikes_percent_std',
'A_Height',
'B_Height',
```

```
'Dif_Height',
'A_Reach',
'B_Reach',
'Dif_Reach',
'A_Leg_Reach',
'B_Leg_Reach',
'Dif_Leg_Reach',
'A_Reach_NA',
'B_Reach_NA',
'Reach_NA',
'A_Leg_Reach_NA',
'B_Leg_Reach_NA',
'Leg_Reach_NA',
'A_Typical_Weightclass',
'B_Typical_Weightclass',
'fight_weightclass',
'A_Fight_in_Typical_Weightclass',
'B_Fight_in_Typical_Weightclass',
'InFightData__Method_Primary',
'InFightData__Method_Detail',
'InFightData__Round',
'InFightData__Time',
'A_Ape_Index',
'B_Ape_Index',
'A_Leg_Index',
'B_Leg_Index',
'A_Leg_to_Wing_Index',
'B_Leg_to_Wing_Index',
'win?',
'favorite?',
'datetime',
'date_formatted',
'A_Total_UFC_Fights',
'B_Total_UFC_Fights',
'Dif_Total_UFC_Fights',
'A_UFC_Wins',
'B_UFC_Wins',
'Dif_UFC_Wins',
'A_UFC_Losses',
'B_UFC_Losses',
'Dif_UFC_Losses',
'A_UFC_Win_Percentage',
'B_UFC_Win_Percentage',
'Dif_UFC_Win_Percentage',
'A_Last5_Win_Percentage',
'B_Last5_Win_Percentage',
'Dif_Last5_Win_Percentage',
'A_Last3_Win_Percentage',
'B_Last3_Win_Percentage',
'Dif_Last3_Win_Percentage',
'InFightData__General_Method',
'A_Win_By_KO_Percentage',
'B_Win_By_KO_Percentage',
'Dif_Win_By_KO_Percentage',
'A_Loss_By_KO_Percentage',
'B_Loss_By_KO_Percentage',
```

```
'Dif_Loss_By_KO_Percentage',
'A_Win_By_Decision_Percentage',
'B_Win_By_Decision_Percentage',
'Dif_Win_By_Decision_Percentage',
'A_Loss_By_Decision_Percentage',
'B_Loss_By_Decision_Percentage',
'Dif_Loss_By_Decision_Percentage',
'final_round_seconds',
'InFightData__Total_Fight_Time_Seconds',
'A_UFC_Fight_Time_Seconds',
'B_UFC_Fight_Time_Seconds',
'Dif_UFC_Fight_Time_Seconds',
'A_UFC_Fight_Rounds',
'B_UFC_Fight_Rounds',
'A_topdown_Avg_Kd_per_round',
'A_topdown_Avg_Sig_strike_land_per_round',
'A_topdown_Avg_Sig_strike_att_per_round',
'A_topdown_Avg_Total_Strikes_land_per_round',
'A_topdown_Avg_Total_Strikes_att_per_round',
'A_topdown_Avg_Takedowns_land_per_round',
'A_topdown_Avg_Takedowns_att_per_round',
'A_topdown_Avg_Sub_Attempts_land_per_round',
'A_topdown_Avg_Sub_Attempts_att_per_round',
'A_topdown_Avg_Rev_per_round',
'A_topdown_Avg_Ctrl_time_min_per_round',
'A_topdown_Avg_Ctrl_time_sec_per_round',
'A_topdown_Avg_Ctrl_time_tot_per_round',
'A_topdown_Avg_Head_Strikes_land_per_round',
'A_topdown_Avg_Head_Strikes_att_per_round',
'A_topdown_Avg_Body_Strikes_land_per_round',
'A_topdown_Avg_Body_Strikes_att_per_round',
'A_topdown_Avg_Leg_Strikes_land_per_round',
'A_topdown_Avg_Leg_Strikes_att_per_round',
'A_topdown_Avg_Distance_Strikes_land_per_round',
'A_topdown_Avg_Distance_Strikes_att_per_round',
'A_topdown_Avg_Clinch_Strikes_land_per_round',
'A_topdown_Avg_Clinch_Strikes_att_per_round',
'A_topdown_Avg_Ground_Strikes_land_per_round',
'A_topdown_Avg_Ground_Strikes_att_per_round',
'B_topdown_Avg_Kd_per_round',
'B_topdown_Avg_Sig_strike_land_per_round',
'B_topdown_Avg_Sig_strike_att_per_round',
'B_topdown_Avg_Total_Strikes_land_per_round',
'B_topdown_Avg_Total_Strikes_att_per_round',
'B_topdown_Avg_Takedowns_land_per_round',
'B_topdown_Avg_Takedowns_att_per_round',
'B_topdown_Avg_Sub_Attempts_land_per_round',
'B_topdown_Avg_Sub_Attempts_att_per_round',
'B_topdown_Avg_Rev_per_round',
'B_topdown_Avg_Ctrl_time_min_per_round',
'B_topdown_Avg_Ctrl_time_sec_per_round',
'B_topdown_Avg_Ctrl_time_tot_per_round',
'B_topdown_Avg_Head_Strikes_land_per_round',
'B_topdown_Avg_Head_Strikes_att_per_round',
'B_topdown_Avg_Body_Strikes_land_per_round',
'B_topdown_Avg_Body_Strikes_att_per_round',
```

```
 'B_topdown_Avg_Leg_Strikes_land_per_round',
 'B_topdown_Avg_Leg_Strikes_att_per_round',
 'B_topdown_Avg_Distance_Strikes_land_per_round',
 'B_topdown_Avg_Distance_Strikes_att_per_round',
 'B_topdown_Avg_Clinch_Strikes_land_per_round',
 'B_topdown_Avg_Clinch_Strikes_att_per_round',
 'B_topdown_Avg_Ground_Strikes_land_per_round',
 'B_topdown_Avg_Ground_Strikes_att_per_round']
```

In [ ]:
```
# get opponent average columns
opp_avg_cols = [n for n in fights.columns.to_list() if 'Opp_Avg' in n]
# A OPP AVG
a_opp_avg_cols = [n for n in opp_avg_cols if n.startswith('A_')]
# get rid of any percent columns
a_opp_avg_cols = [n for n in a_opp_avg_cols if 'percent' not in n]

# B OPP AVG
b_opp_avg_cols = [n for n in opp_avg_cols if n.startswith('B_')]
# get rid of any percent columns
b_opp_avg_cols = [n for n in b_opp_avg_cols if 'percent' not in n]

b_opp_avg_cols
```

Out[ ]:
```
['B_Opp_Avg_Kd',
 'B_Opp_Avg_Sig_strike_land',
 'B_Opp_Avg_Sig_strike_att',
 'B_Opp_Avg_Total_Strikes_land',
 'B_Opp_Avg_Total_Strikes_att',
 'B_Opp_Avg_Takedowns_land',
 'B_Opp_Avg_Takedowns_att',
 'B_Opp_Avg_Sub_Attempts_land',
 'B_Opp_Avg_Sub_Attempts_att',
 'B_Opp_Avg_Rev',
 'B_Opp_Avg_Ctrl_time_min',
 'B_Opp_Avg_Ctrl_time_sec',
 'B_Opp_Avg_Ctrl_time_tot',
 'B_Opp_Avg_Head_Strikes_land',
 'B_Opp_Avg_Head_Strikes_att',
 'B_Opp_Avg_Body_Strikes_land',
 'B_Opp_Avg_Body_Strikes_att',
 'B_Opp_Avg_Leg_Strikes_land',
 'B_Opp_Avg_Leg_Strikes_att',
 'B_Opp_Avg_Distance_Strikes_land',
 'B_Opp_Avg_Distance_Strikes_att',
 'B_Opp_Avg_Clinch_Strikes_land',
 'B_Opp_Avg_Clinch_Strikes_att',
 'B_Opp_Avg_Ground_Strikes_land',
 'B_Opp_Avg_Ground_Strikes_att']
```

In [ ]:
```
# add per-round averages using opponent averages
for col in a_opp_avg_cols:
    fights[f'{col}_per_round'] = fights[col] / fights['A_UFC_Fight_Rounds']

for col in b_opp_avg_cols:
    fights[f'{col}_per_round'] = fights[col] / fights['B_UFC_Fight_Rounds']
```

Add Per-Round Difs

In [ ]:
```
# add difference in per-round averages

# get per-round topdown columns
topdown_per_round_cols = [n for n in fights.columns.to_list() if 'topdown' and 'per

topdown_per_round_cols = pd.DataFrame(topdown_per_round_cols)
topdown_per_round_cols.columns = ['col']
topdown_per_round_cols['non-specific'] = topdown_per_round_cols['col'].str[2:]
# get unique values
topdown_per_round_cols = topdown_per_round_cols['non-specific'].unique().tolist()
```

In [ ]:
```
for col in topdown_per_round_cols:
    fights[f'Dif_{col}'] = fights[f'A_{col}'] - fights[f'B_{col}']
```

In [ ]:
```
fights.head(2)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
3	Greg Hardy	Ben Sosoli	0	0	54	105	26	

2 rows × 780 columns

In [ ]:
```
# save to csv
fights.to_csv('data/final/aggregates/Double_Fights_DF_V14.csv', index=False)
```

EDA

In [ ]:
```
fights = pd.read_csv('data/final/aggregates/Double_Fights_DF_V14.csv')
```

In [ ]:
```
fights.columns
```

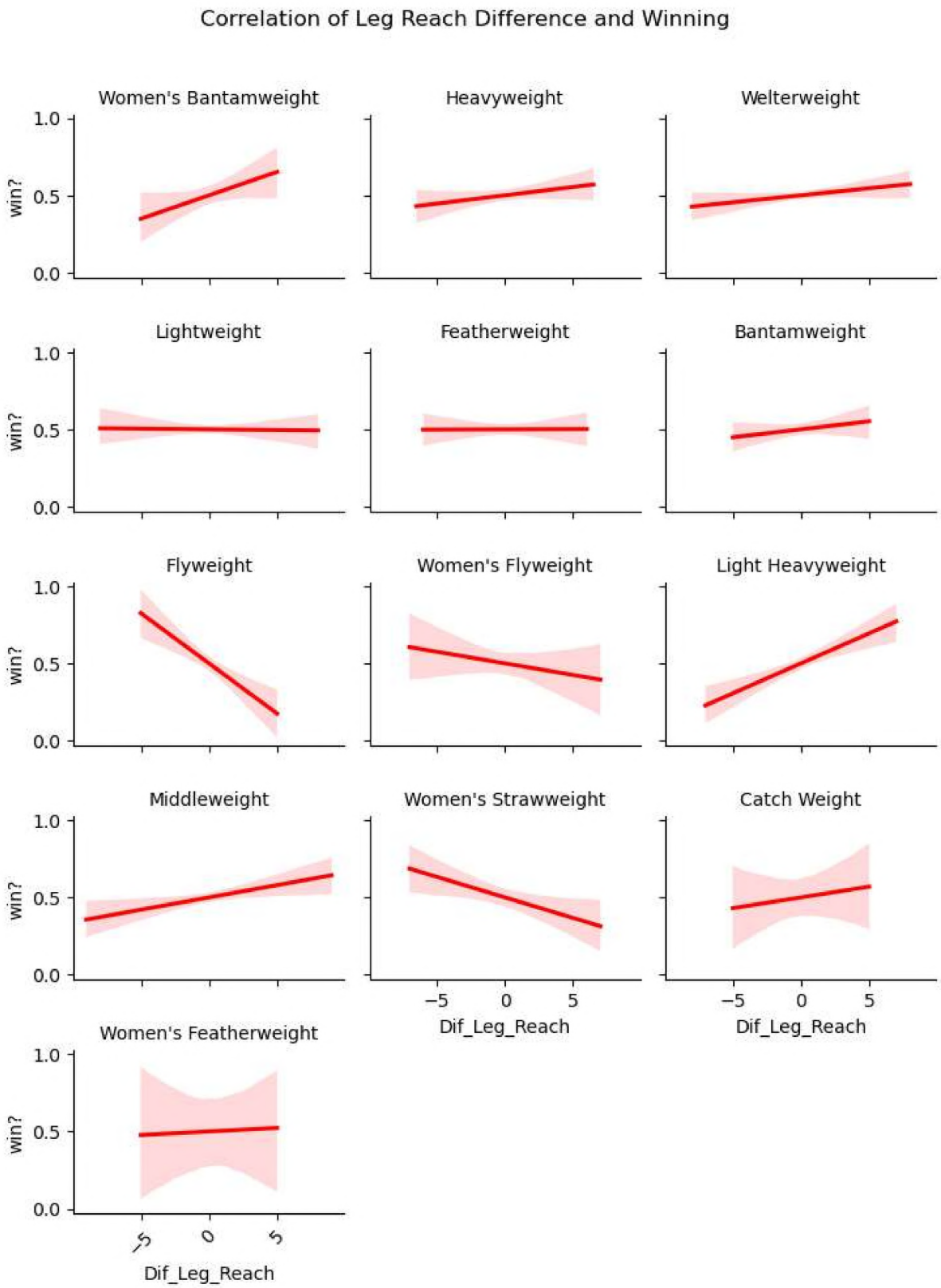
Out[ ]:
```
Index(['Fighter_A', 'Fighter_B', 'A_Kd', 'B_Kd', 'A_Sig_strike_land',
       'A_Sig_strike_att', 'B_Sig_strike_land', 'B_Sig_strike_att',
       'A_Sig_strike_percent', 'B_Sig_strike_percent',
       ...
       'Dif_Opp_Avg_Body_Strikes_land_per_round',
       'Dif_Opp_Avg_Body_Strikes_att_per_round',
       'Dif_Opp_Avg_Leg_Strikes_land_per_round',
       'Dif_Opp_Avg_Leg_Strikes_att_per_round',
       'Dif_Opp_Avg_Distance_Strikes_land_per_round',
       'Dif_Opp_Avg_Distance_Strikes_att_per_round',
       'Dif_Opp_Avg_Clinch_Strikes_land_per_round',
       'Dif_Opp_Avg_Clinch_Strikes_att_per_round',
       'Dif_Opp_Avg_Ground_Strikes_land_per_round',
       'Dif_Opp_Avg_Ground_Strikes_att_per_round'],
      dtype='object', length=780)
```

First, I want to look at what wins fights using in-fight data, which I will not be using to solve with.

In [ ]:
```
unnamed = [n for n in fights.columns.to_list() if 'Unnamed' in n]
# drop unnamed columns
fights = fights.drop(columns=unnamed)
```

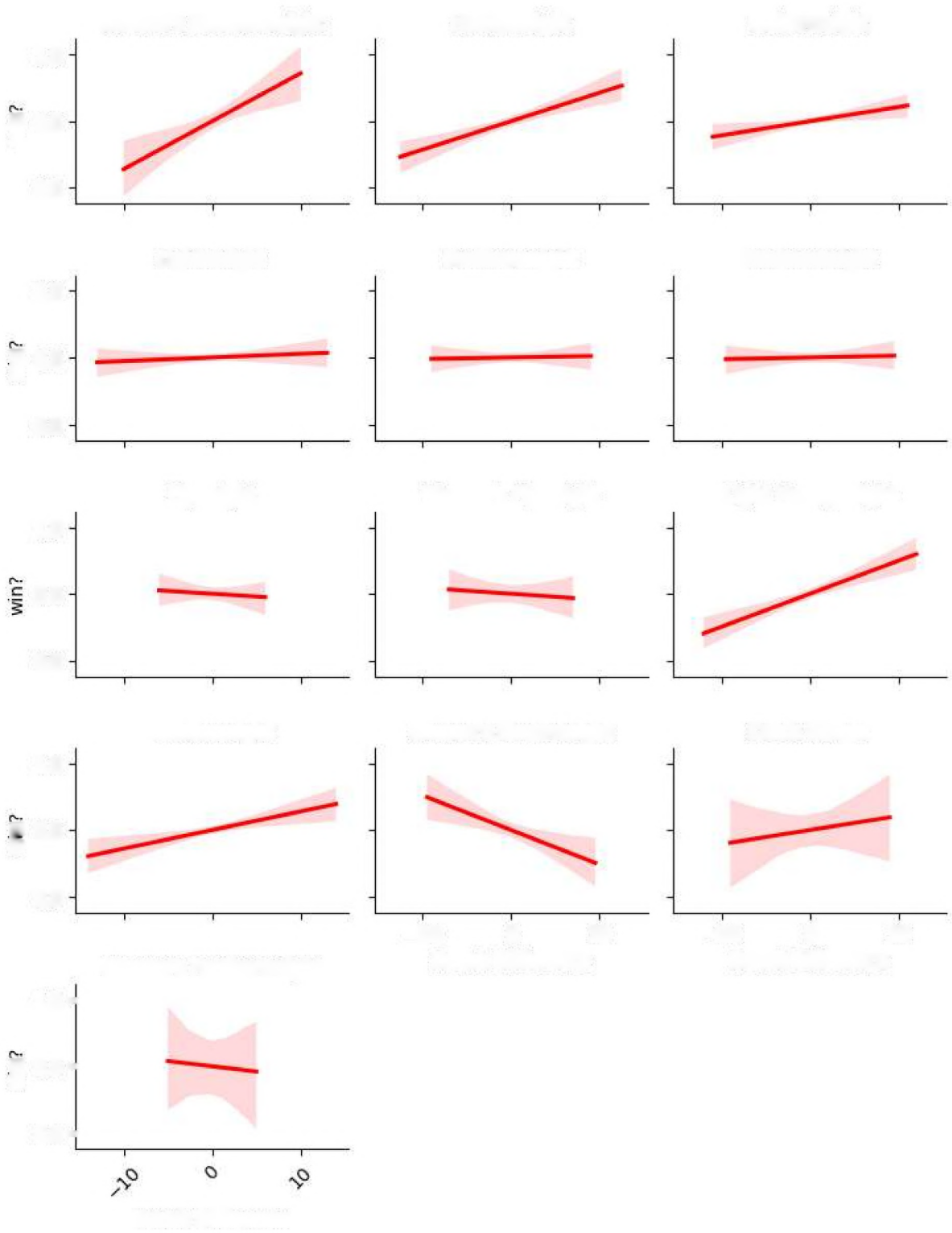
In [ ]:
```
# drop first 115 columns
fight_viz = fights.iloc[:, 115:]
```

In [ ]:
```
# plot correlation coefficients
g = sns.FacetGrid(fight_viz, col='fight_weightclass', col_wrap=3, height=2, aspect=
g.map_dataframe(sns.regplot, x='Dif_Leg_Reach', y='win?', scatter=False, color='red
g.set_axis_labels('Dif_Leg_Reach', 'win?')
g.set_titles('{col_name}')
g.fig.suptitle('Correlation of Leg Reach Difference and Winning')
g.fig.subplots_adjust(top=0.9)
# plot x-axis ticks and labels
plt.xticks(rotation=45)
plt.show()
# show y-axis ticks
```



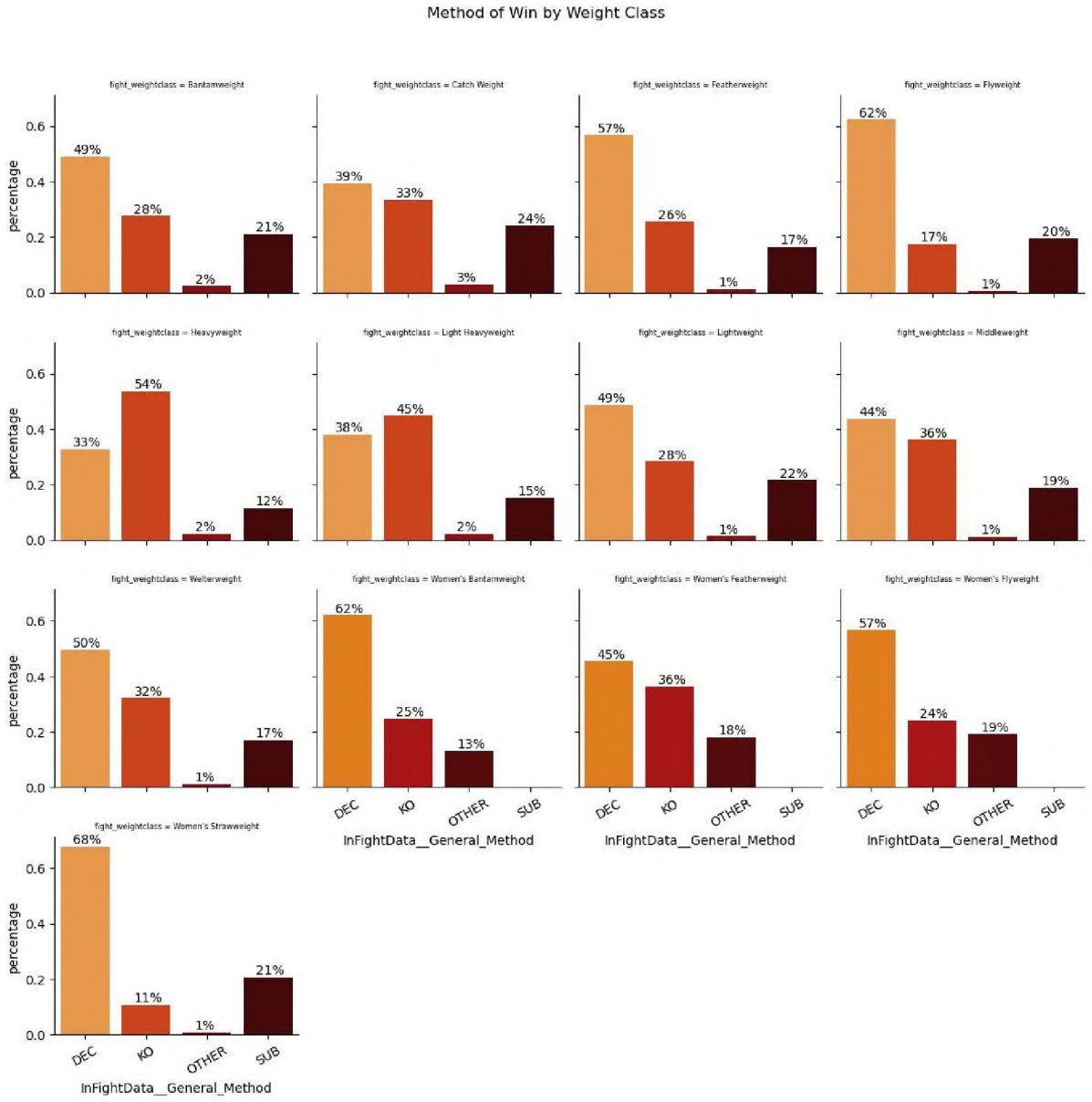
This is interesting, as it shows that in some weightclasses, having a leg reach disadvantage may actually be an advantage, such as with the Flyweight class. This would seem to indicate that the smaller the athlete, the more important compactness (of body) becomes. Let's look further into this by taking a look at reach.

In [ ]:
```
# plot correlation coefficients
g = sns.FacetGrid(fight_viz, col='fight_weightclass', col_wrap=3, height=2, aspect=
g.map_dataframe(sns.regplot, x='Dif_Reach', y='win?', scatter=False, color='red')
g.set_axis_labels('Reach Difference', 'win?')
g.set_titles('{col_name}')
g.fig.suptitle('Correlation of Reach Difference and Winning')
g.fig.subplots_adjust(top=0.9)
# plot x-axis ticks and labels
plt.xticks(rotation=45)
plt.show()
```



Once again, it appears that as the weightclass decreases, the importance of reach decreases, and vice versa. This makes sense when it comes to heavyweights, as being able to reach your opponent AND having 'knockout power' is a large advantage. Let's check the KO/TKO rate for each weightclass.

In [ ]:
```
grouped = fights.groupby(['InFightData__General_Method', 'fight_weightclass']).size
grouped['percentage'] = grouped.groupby('fight_weightclass')['counts'].apply(lambda
g = sns.FacetGrid(grouped, col='fight_weightclass', col_wrap=4, height=3)
g = g.map(sns.barplot, 'InFightData__General_Method', 'percentage', palette='gist_h

# make titles smaller
for ax in g.axes.flat:
    ax.title.set_fontsize(6)
# show x-axis ticks and labels
# add data labels
for ax in g.axes.flat:
    for p in ax.patches:
        ax.annotate('{:.0%}'.format(p.get_height()), (p.get_x()+0.2, p.get_height()

# show x-axis ticks and labels
for ax in g.axes.flat:
    ax.set_xticklabels(ax.get_xticklabels(), rotation=30)

# add overall title
g.fig.suptitle('Method of Win by Weight Class')
g.fig.subplots_adjust(top=0.9)

plt.show()
```



In [ ]:
```
data_pred = fight_viz
df=data_pred.corr().abs().stack().reset_index().sort_values(0, ascending=False)
df['pairs'] = list(zip(df.level_0, df.level_1))
df.set_index(['pairs'], inplace = True)
df.drop(columns=['level_1', 'level_0'], inplace = True)
df.columns = ['cc']
df.drop_duplicates(inplace=True)

# only keep pairs that include win?
df = df[df.index.map(lambda x: 'win?' in x[0] or 'win?' in x[1])]
# make cc not scientific notation
df['cc'] = df['cc'].apply(lambda x: '{:.2f}'.format(x))
df['cc'] = df['cc'].astype(float)
# only keep pairs with cc > 0.2
df = df[df['cc'] > 0.2]

df
```

Out[ ]:

pairs	cc
(Dif_Odds, win?)	0.36
(win?, favorite?)	0.31
(win?, Dif_Rolling_Head_Strikes_land_mean)	0.28
(win?, Dif_Rolling_Head_Strikes_percent_mean)	0.27
(win?, Dif_Rolling_Sig_strike_land_mean)	0.27
(win?, Dif_Rolling_Head_Strikes_percent_median)	0.26
(Dif_Rolling_Ground_Strikes_land_mean, win?)	0.26
(Dif_Rolling_Kd_std, win?)	0.25
(win?, Dif_Rolling_Total_Strikes_land_mean)	0.25
(Dif_Rolling_Ground_Strikes_att_mean, win?)	0.25
(Dif_Rolling_Total_Strikes_land_std, win?)	0.23
(win?, Dif_Rolling_Head_Strikes_land_std)	0.23
(win?, Dif_Rolling_Sig_strike_land_std)	0.23
(Dif_Rolling_Sig_strike_percent_mean, win?)	0.23
(win?, Dif_Rolling_Ground_Strikes_land_std)	0.23
(Dif_Rolling_Head_Strikes_land_median, win?)	0.23
(win?, Dif_Rolling_Ground_Strikes_att_std)	0.23
(win?, Dif_Rolling_Distance_Strikes_land_std)	0.23
(Dif_Rolling_Kd_mean, win?)	0.23
(Dif_Rolling_Ground_Strikes_percent_mean, win?)	0.23
(win?, Dif_Rolling_Ctrl_time_tot_std)	0.22
(Dif_Rolling_Sig_strike_land_median, win?)	0.22
(win?, Dif_Rolling_Sig_strike_percent_median)	0.22
(win?, Dif_Rolling_Ctrl_time_min_std)	0.22
(Dif_Rolling_Total_Strikes_land_median, win?)	0.21
(Dif_Rolling_Takedowns_land_std, win?)	0.21
(Dif_Rolling_Total_Strikes_att_std, win?)	0.21
(win?, Dif_Rolling_Total_Strikes_att_mean)	0.21

Plot histogram and PDF of each column

In [ ]:
```
fight_viz_num = fight_viz.select_dtypes(include=['float64', 'int64'])
# get rid of any infinite values
fight_viz_num = fight_viz_num.replace([np.inf, -np.inf], np.nan)
```

```
# drop any rows with NaN values
fight_viz_num = fight_viz_num.dropna()
```

In [ ]:
```
fight_viz_rolling_dif_columns= [n for n in fight_viz_num.columns.to_list() if 'Dif'
fight_viz_rolling_dif_columns
```

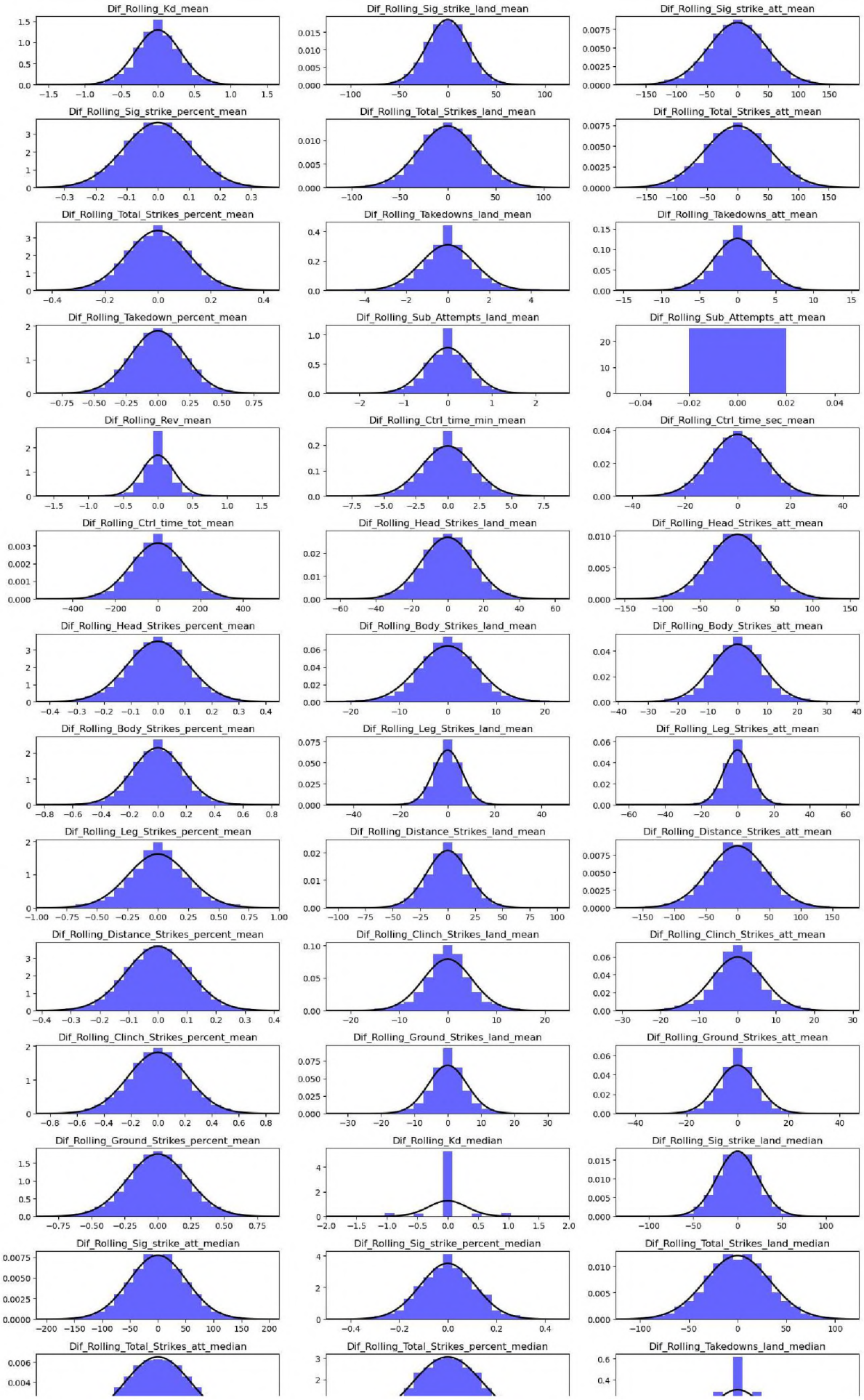
Out[ ]:
```
['Dif_Rolling_Kd_mean',
 'Dif_Rolling_Sig_strike_land_mean',
 'Dif_Rolling_Sig_strike_att_mean',
 'Dif_Rolling_Sig_strike_percent_mean',
 'Dif_Rolling_Total_Strikes_land_mean',
 'Dif_Rolling_Total_Strikes_att_mean',
 'Dif_Rolling_Total_Strikes_percent_mean',
 'Dif_Rolling_Takedowns_land_mean',
 'Dif_Rolling_Takedowns_att_mean',
 'Dif_Rolling_Takedown_percent_mean',
 'Dif_Rolling_Sub_Attempts_land_mean',
 'Dif_Rolling_Sub_Attempts_att_mean',
 'Dif_Rolling_Rev_mean',
 'Dif_Rolling_Ctrl_time_min_mean',
 'Dif_Rolling_Ctrl_time_sec_mean',
 'Dif_Rolling_Ctrl_time_tot_mean',
 'Dif_Rolling_Head_Strikes_land_mean',
 'Dif_Rolling_Head_Strikes_att_mean',
 'Dif_Rolling_Head_Strikes_percent_mean',
 'Dif_Rolling_Body_Strikes_land_mean',
 'Dif_Rolling_Body_Strikes_att_mean',
 'Dif_Rolling_Body_Strikes_percent_mean',
 'Dif_Rolling_Leg_Strikes_land_mean',
 'Dif_Rolling_Leg_Strikes_att_mean',
 'Dif_Rolling_Leg_Strikes_percent_mean',
 'Dif_Rolling_Distance_Strikes_land_mean',
 'Dif_Rolling_Distance_Strikes_att_mean',
 'Dif_Rolling_Distance_Strikes_percent_mean',
 'Dif_Rolling_Clinch_Strikes_land_mean',
 'Dif_Rolling_Clinch_Strikes_att_mean',
 'Dif_Rolling_Clinch_Strikes_percent_mean',
 'Dif_Rolling_Ground_Strikes_land_mean',
 'Dif_Rolling_Ground_Strikes_att_mean',
 'Dif_Rolling_Ground_Strikes_percent_mean',
 'Dif_Rolling_Kd_median',
 'Dif_Rolling_Sig_strike_land_median',
 'Dif_Rolling_Sig_strike_att_median',
 'Dif_Rolling_Sig_strike_percent_median',
 'Dif_Rolling_Total_Strikes_land_median',
 'Dif_Rolling_Total_Strikes_att_median',
 'Dif_Rolling_Total_Strikes_percent_median',
 'Dif_Rolling_Takedowns_land_median',
 'Dif_Rolling_Takedowns_att_median',
 'Dif_Rolling_Takedown_percent_median',
 'Dif_Rolling_Sub_Attempts_land_median',
 'Dif_Rolling_Sub_Attempts_att_median',
 'Dif_Rolling_Rev_median',
 'Dif_Rolling_Ctrl_time_min_median',
 'Dif_Rolling_Ctrl_time_sec_median',
 'Dif_Rolling_Ctrl_time_tot_median',
 'Dif_Rolling_Head_Strikes_land_median',
 'Dif_Rolling_Head_Strikes_att_median',
 'Dif_Rolling_Head_Strikes_percent_median',
 'Dif_Rolling_Body_Strikes_land_median',
 'Dif_Rolling_Body_Strikes_att_median',
 'Dif_Rolling_Body_Strikes_percent_median',
```

```
 'Dif_Rolling_Leg_Strikes_land_median',
 'Dif_Rolling_Leg_Strikes_att_median',
 'Dif_Rolling_Leg_Strikes_percent_median',
 'Dif_Rolling_Distance_Strikes_land_median',
 'Dif_Rolling_Distance_Strikes_att_median',
 'Dif_Rolling_Distance_Strikes_percent_median',
 'Dif_Rolling_Clinch_Strikes_land_median',
 'Dif_Rolling_Clinch_Strikes_att_median',
 'Dif_Rolling_Clinch_Strikes_percent_median',
 'Dif_Rolling_Ground_Strikes_land_median',
 'Dif_Rolling_Ground_Strikes_att_median',
 'Dif_Rolling_Ground_Strikes_percent_median',
 'Dif_Rolling_Kd_std',
 'Dif_Rolling_Sig_strike_land_std',
 'Dif_Rolling_Sig_strike_att_std',
 'Dif_Rolling_Sig_strike_percent_std',
 'Dif_Rolling_Total_Strikes_land_std',
 'Dif_Rolling_Total_Strikes_att_std',
 'Dif_Rolling_Total_Strikes_percent_std',
 'Dif_Rolling_Takedowns_land_std',
 'Dif_Rolling_Takedowns_att_std',
 'Dif_Rolling_Takedown_percent_std',
 'Dif_Rolling_Sub_Attempts_land_std',
 'Dif_Rolling_Sub_Attempts_att_std',
 'Dif_Rolling_Rev_std',
 'Dif_Rolling_Ctrl_time_min_std',
 'Dif_Rolling_Ctrl_time_sec_std',
 'Dif_Rolling_Ctrl_time_tot_std',
 'Dif_Rolling_Head_Strikes_land_std',
 'Dif_Rolling_Head_Strikes_att_std',
 'Dif_Rolling_Head_Strikes_percent_std',
 'Dif_Rolling_Body_Strikes_land_std',
 'Dif_Rolling_Body_Strikes_att_std',
 'Dif_Rolling_Body_Strikes_percent_std',
 'Dif_Rolling_Leg_Strikes_land_std',
 'Dif_Rolling_Leg_Strikes_att_std',
 'Dif_Rolling_Leg_Strikes_percent_std',
 'Dif_Rolling_Distance_Strikes_land_std',
 'Dif_Rolling_Distance_Strikes_att_std',
 'Dif_Rolling_Distance_Strikes_percent_std',
 'Dif_Rolling_Clinch_Strikes_land_std',
 'Dif_Rolling_Clinch_Strikes_att_std',
 'Dif_Rolling_Clinch_Strikes_percent_std',
 'Dif_Rolling_Ground_Strikes_land_std',
 'Dif_Rolling_Ground_Strikes_att_std',
 'Dif_Rolling_Ground_Strikes_percent_std']
```

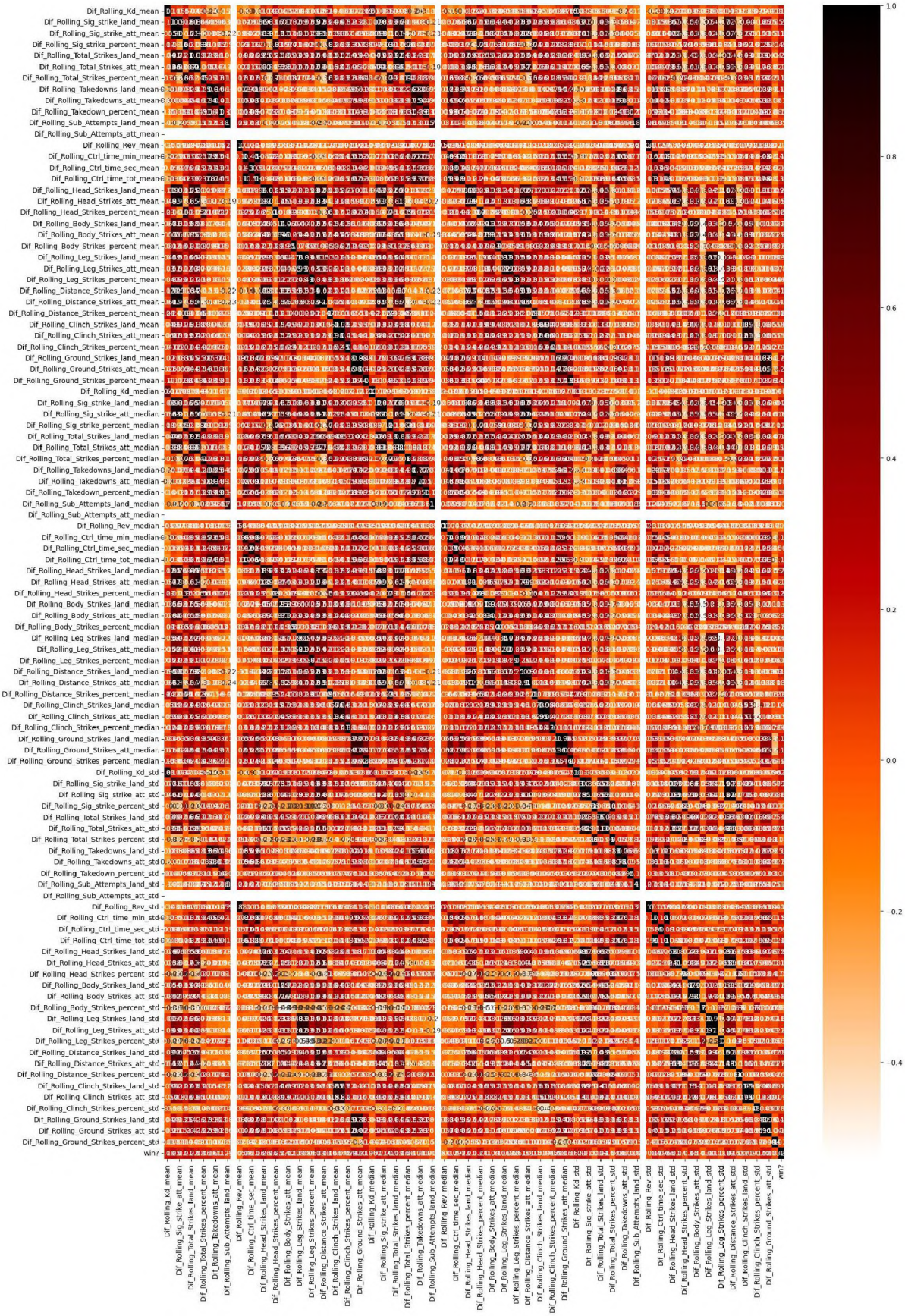
In [ ]:
```
fv_rolling_dif_cols = fight_viz_num[fight_viz_rolling_dif_columns]
```

In [ ]:
```
ncols = 3
nrows = int(np.ceil(len(fv_rolling_dif_cols.columns) / ncols))

fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=(15, 60))
ax = ax.ravel()

for i, column in enumerate(fv_rolling_dif_cols.columns):
```

```
    data = fv_rolling_dif_cols[column]
    mu, std = norm.fit(data)
    x = np.linspace(np.min(data), np.max(data), 100)
    p = norm.pdf(x, mu, std)
    ax[i].hist(data, bins=25, density=True, alpha=0.6, color='b')
    ax[i].plot(x, p, 'k', linewidth=2)
    ax[i].set_title(column)
    ax[i].set_xlim(np.min(data), np.max(data))

plt.tight_layout()
plt.show()
```

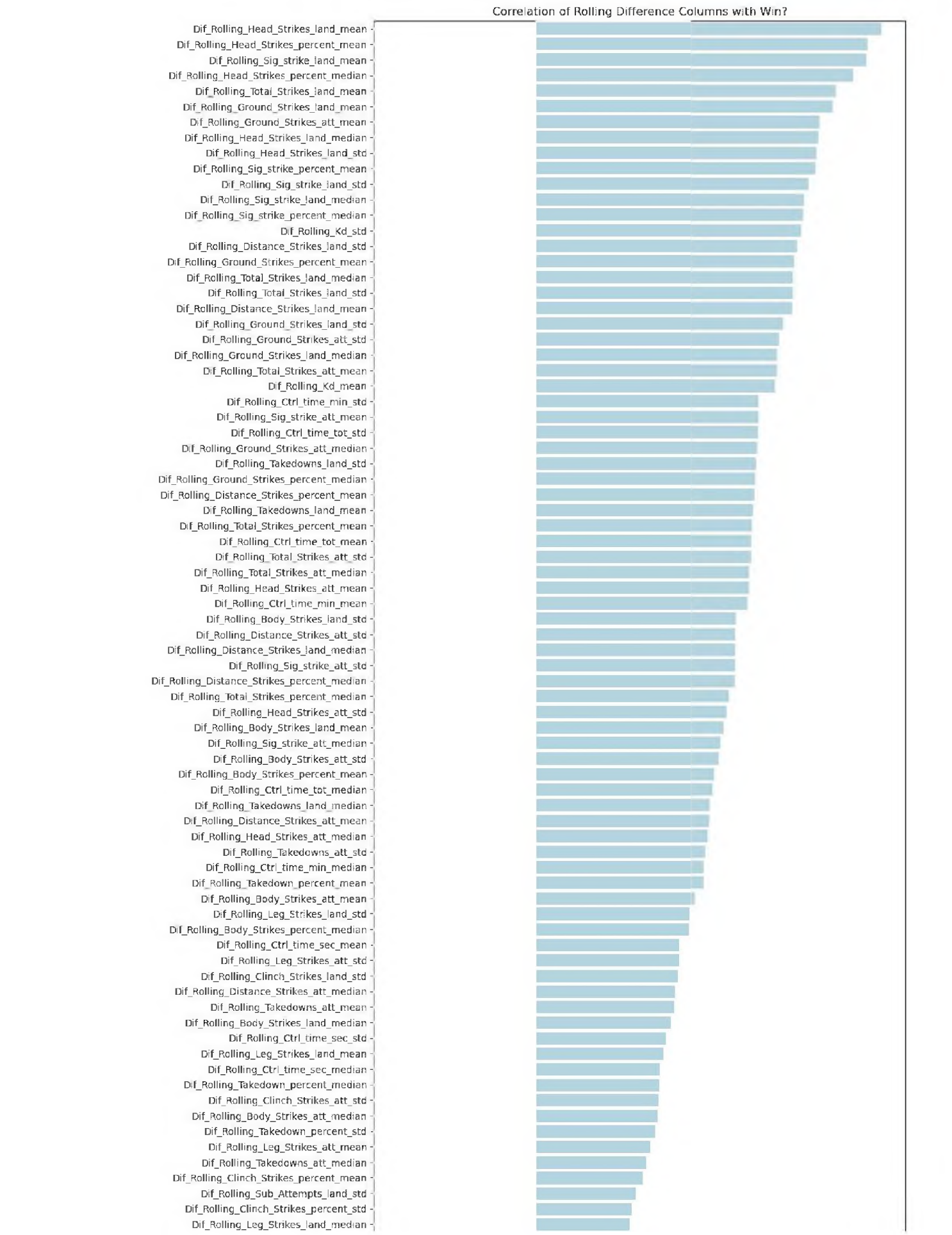

Dif_Rolling_Kd_mean
Dif_Rolling_Sig_strike_land_mean
Dif_Rolling_Sig_strike_att_mean
Dif_Rolling_Sig_strike_percent_mean
Dif_Rolling_Total_Strikes_land_mean
Dif_Rolling_Total_Strikes_att_mean
Dif_Rolling_Total_Strikes_percent_mean
Dif_Rolling_Takedowns_land_mean
Dif_Rolling_Takedowns_att_mean
Dif_Rolling_Takedown_percent_mean
Dif_Rolling_Sub_Attempts_land_mean
Dif_Rolling_Sub_Attempts_att_mean
Dif_Rolling_Rev_mean
Dif_Rolling_Ctrl_time_min_mean
Dif_Rolling_Ctrl_time_sec_mean
Dif_Rolling_Ctrl_time_tot_mean
Dif_Rolling_Head_Strikes_land_mean
Dif_Rolling_Head_Strikes_att_mean
Dif_Rolling_Head_Strikes_percent_mean
Dif_Rolling_Body_Strikes_land_mean
Dif_Rolling_Body_Strikes_att_mean
Dif_Rolling_Body_Strikes_percent_mean
Dif_Rolling_Leg_Strikes_land_mean
Dif_Rolling_Leg_Strikes_att_mean
Dif_Rolling_Leg_Strikes_percent_mean
Dif_Rolling_Distance_Strikes_land_mean
Dif_Rolling_Distance_Strikes_att_mean
Dif_Rolling_Distance_Strikes_percent_mean
Dif_Rolling_Clinch_Strikes_land_mean
Dif_Rolling_Clinch_Strikes_att_mean
Dif_Rolling_Clinch_Strikes_percent_mean
Dif_Rolling_Ground_Strikes_land_mean
Dif_Rolling_Ground_Strikes_att_mean
Dif_Rolling_Ground_Strikes_percent_mean
Dif_Rolling_Kd_median
Dif_Rolling_Sig_strike_land_median
Dif_Rolling_Sig_strike_att_median
Dif_Rolling_Sig_strike_percent_median
Dif_Rolling_Total_Strikes_land_median
Dif_Rolling_Total_Strikes_att_median
Dif_Rolling_Total_Strikes_percent_median
Dif_Rolling_Takedowns_land_median

In [ ]:
```
# check correlation coefficients for rolling difference columns with win?
fvrd_and_wins = fv_rolling_dif_cols.join(fight_viz_num['win?'])
# correlation heatmap
corr = fvrd_and_wins.corr()
plt.figure(figsize=(20, 30))
sns.heatmap(corr, annot=True, cmap='gist_heat_r')
plt.show()
```



In [ ]:
```
# correlation coefficients for rolling difference columns with win?
# drop win? from the list
rol_cors = fvrd_and_wins.corr()['win?'].sort_values(ascending=False)
rol_cors = rol_cors.drop('win?')
```

```
# plot correlation coefficients in a horizontal bar chart
plt.figure(figsize=(10, 30))
sns.barplot(x=rol_cors.values, y=rol_cors.index, color='lightblue')
plt.title('Correlation of Rolling Difference Columns with Win?')
plt.show()
```


Correlation of Rolling Difference Columns with Win?
Dif_Rolling_Head_Strikes_land_mean
Dif_Rolling_Head_Strikes_percent_mean
Dif_Rolling_Sig_strike_land_mean
Dif_Rolling_Head_Strikes_percent_median
Dif_Rolling_Total_Strikes_land_mean
Dif_Rolling_Ground_Strikes_land_mean
Dif_Rolling_Ground_Strikes_att_mean
Dif_Rolling_Head_Strikes_land_median
Dif_Rolling_Head_Strikes_land_std
Dif_Rolling_Sig_strike_percent_mean
Dif_Rolling_Sig_strike_land_std
Dif_Rolling_Sig_strike_land_median
Dif_Rolling_Sig_strike_percent_median
Dif_Rolling_Kd_std
Dif_Rolling_Distance_Strikes_land_std
Dif_Rolling_Ground_Strikes_percent_mean
Dif_Rolling_Total_Strikes_land_median
Dif_Rolling_Total_Strikes_land_std
Dif_Rolling_Distance_Strikes_land_mean
Dif_Rolling_Ground_Strikes_land_std
Dif_Rolling_Ground_Strikes_att_std
Dif_Rolling_Ground_Strikes_land_median
Dif_Rolling_Total_Strikes_att_mean
Dif_Rolling_Kd_mean
Dif_Rolling_Ctrl_time_min_std
Dif_Rolling_Sig_strike_att_mean
Dif_Rolling_Ctrl_time_tot_std
Dif_Rolling_Ground_Strikes_att_median
Dif_Rolling_Takedowns_land_std
Dif_Rolling_Ground_Strikes_percent_median
Dif_Rolling_Distance_Strikes_percent_mean
Dif_Rolling_Takedowns_land_mean
Dif_Rolling_Total_Strikes_percent_mean
Dif_Rolling_Ctrl_time_tot_mean
Dif_Rolling_Total_Strikes_att_std
Dif_Rolling_Total_Strikes_att_median
Dif_Rolling_Head_Strikes_att_mean
Dif_Rolling_Ctrl_time_min_mean
Dif_Rolling_Body_Strikes_land_std
Dif_Rolling_Distance_Strikes_att_std
Dif_Rolling_Distance_Strikes_land_median
Dif_Rolling_Sig_strike_att_std
Dif_Rolling_Distance_Strikes_percent_median
Dif_Rolling_Total_Strikes_percent_median
Dif_Rolling_Head_Strikes_att_std
Dif_Rolling_Body_Strikes_land_mean
Dif_Rolling_Sig_strike_att_median
Dif_Rolling_Body_Strikes_att_std
Dif_Rolling_Body_Strikes_percent_mean
Dif_Rolling_Ctrl_time_tot_median
Dif_Rolling_Takedowns_land_median
Dif_Rolling_Distance_Strikes_att_mean
Dif_Rolling_Head_Strikes_att_median
Dif_Rolling_Takedowns_att_std
Dif_Rolling_Ctrl_time_min_median
Dif_Rolling_Takedown_percent_mean
Dif_Rolling_Body_Strikes_att_mean
Dif_Rolling_Leg_Strikes_land_std
Dif_Rolling_Body_Strikes_percent_median
Dif_Rolling_Ctrl_time_sec_mean
Dif_Rolling_Leg_Strikes_att_std
Dif_Rolling_Clinch_Strikes_land_std
Dif_Rolling_Distance_Strikes_att_median
Dif_Rolling_Takedowns_att_mean
Dif_Rolling_Body_Strikes_land_median
Dif_Rolling_Ctrl_time_sec_std
Dif_Rolling_Leg_Strikes_land_mean
Dif_Rolling_Ctrl_time_sec_median
Dif_Rolling_Takedown_percent_median
Dif_Rolling_Clinch_Strikes_att_std
Dif_Rolling_Body_Strikes_att_median
Dif_Rolling_Takedown_percent_std
Dif_Rolling_Leg_Strikes_att_mean
Dif_Rolling_Takedowns_att_median
Dif_Rolling_Clinch_Strikes_percent_mean
Dif_Rolling_Sub_Attempts_land_std
Dif_Rolling_Clinch_Strikes_percent_std
Dif_Rolling_Leg_Strikes_land_median

Prediction

In [ ]:
```
df  = pd.read_csv('data/final/aggregates/Double_Fights_DF_V14.csv')
print(df.shape)
df.head(3)
```

```
(8138, 780)
```

Out[ ]:

	Fighter_A	Fighter_B	A_Kd	B_Kd	A_Sig_strike_land	A_Sig_strike_att	B_Sig_strike_land	B_Sig_st
0	Holly Holm	Irene Aldana	0	0	154	301	69	
1	Greg Hardy	Ben Sosoli	0	0	54	105	26	
2	Jared Rosholt	Josh Copeland	0	0	22	45	9	

3 rows × 780 columns

In [ ]:
```
unnamed = [n for n in df.columns if 'Unnamed' in n]
df.drop(columns=unnamed, inplace=True)
#Identify columns with missing values
nothere = df.isna().sum()
nothere = pd.DataFrame(nothere)
nothere = nothere.loc[nothere[0] > 0]
nothere
```

Out[ ]:

	0
A_topdown_Avg_Kd_per_round	892
A_topdown_Avg_Sig_strike_land_per_round	14
A_topdown_Avg_Sig_strike_att_per_round	2
A_topdown_Avg_Total_Strikes_land_per_round	8
A_topdown_Avg_Total_Strikes_att_per_round	2
...	...
Dif_Opp_Avg_Distance_Strikes_att_per_round	604
Dif_Opp_Avg_Clinch_Strikes_land_per_round	732
Dif_Opp_Avg_Clinch_Strikes_att_per_round	698
Dif_Opp_Avg_Ground_Strikes_land_per_round	778
Dif_Opp_Avg_Ground_Strikes_att_per_round	762

150 rows × 1 columns

In [ ]:
```
# get cols with InFightData in the name
InFightData = [n for n in df.columns if 'InFightData' in n]
InFightData
```

Out[ ]:
```
['InFightData__Method_Primary',
 'InFightData__Method_Detail',
 'InFightData__Round',
 'InFightData__Time',
 'InFightData__General_Method',
 'InFightData__Total_Fight_Time_Seconds']
```

In [ ]:
```
# drop them from df
df.drop(columns=InFightData, inplace=True)
```

In [ ]:
```
df.columns[:50]
```

Out[ ]:
```
Index(['Fighter_A', 'Fighter_B', 'A_Kd', 'B_Kd', 'A_Sig_strike_land',
       'A_Sig_strike_att', 'B_Sig_strike_land', 'B_Sig_strike_att',
       'A_Sig_strike_percent', 'B_Sig_strike_percent', 'A_Total_Strikes_land',
       'A_Total_Strikes_att', 'B_Total_Strikes_land', 'B_Total_Strikes_att',
       'A_Total_Strikes_percent', 'B_Total_Strikes_percent',
       'A_Takedowns_land', 'A_Takedowns_att', 'B_Takedowns_land',
       'B_Takedowns_att', 'A_Takedown_percent', 'B_Takedown_percent',
       'A_Sub_Attempts_land', 'A_Sub_Attempts_att', 'B_Sub_Attempts_land',
       'B_Sub_Attempts_att', 'A_Rev', 'B_Rev', 'A_Ctrl_time_min',
       'A_Ctrl_time_sec', 'B_Ctrl_time_min', 'B_Ctrl_time_sec',
       'A_Ctrl_time_tot', 'B_Ctrl_time_tot', 'details', 'event_title',
       'event_url', 'date', 'Winner', 'fight_id', 'A_Head_Strikes_land',
       'A_Head_Strikes_att', 'B_Head_Strikes_land', 'B_Head_Strikes_att',
       'A_Head_Strikes_percent', 'B_Head_Strikes_percent',
       'A_Body_Strikes_land', 'A_Body_Strikes_att', 'B_Body_Strikes_land',
       'B_Body_Strikes_att'],
      dtype='object')
```

In [ ]:
```
df.columns[51:85]
```

Out[ ]:
```
Index(['B_Body_Strikes_percent', 'A_Leg_Strikes_land', 'A_Leg_Strikes_att',
       'B_Leg_Strikes_land', 'B_Leg_Strikes_att', 'A_Leg_Strikes_percent',
       'B_Leg_Strikes_percent', 'A_Distance_Strikes_land',
       'A_Distance_Strikes_att', 'B_Distance_Strikes_land',
       'B_Distance_Strikes_att', 'A_Distance_Strikes_percent',
       'B_Distance_Strikes_percent', 'A_Clinch_Strikes_land',
       'A_Clinch_Strikes_att', 'B_Clinch_Strikes_land', 'B_Clinch_Strikes_att',
       'A_Clinch_Strikes_percent', 'B_Clinch_Strikes_percent',
       'A_Ground_Strikes_land', 'A_Ground_Strikes_att',
       'B_Ground_Strikes_land', 'B_Ground_Strikes_att',
       'A_Ground_Strikes_percent', 'B_Ground_Strikes_percent', 'event_code',
       'Fighter_A_Odds', 'Fighter_B_Odds', 'Fighter_A_Odds_Change',
       'Fighter_B_Odds_Change', 'Dif_Kd', 'Dif_Sig_strike_land',
       'Dif_Sig_strike_att', 'Dif_Sig_strike_percent'],
      dtype='object')
```

In [ ]:
```
to_drop= ['A_Kd', 'B_Kd', 'A_Sig_strike_land',
       'A_Sig_strike_att', 'B_Sig_strike_land', 'B_Sig_strike_att',
       'A_Sig_strike_percent', 'B_Sig_strike_percent', 'A_Total_Strikes_land',
       'A_Total_Strikes_att', 'B_Total_Strikes_land', 'B_Total_Strikes_att',
       'A_Total_Strikes_percent', 'B_Total_Strikes_percent',
       'A_Takedowns_land', 'A_Takedowns_att', 'B_Takedowns_land',
       'B_Takedowns_att', 'A_Takedown_percent', 'B_Takedown_percent',
       'A_Sub_Attempts_land', 'A_Sub_Attempts_att', 'B_Sub_Attempts_land',
       'B_Sub_Attempts_att', 'A_Rev', 'B_Rev', 'A_Ctrl_time_min',
       'A_Ctrl_time_sec', 'B_Ctrl_time_min', 'B_Ctrl_time_sec',
       'A_Ctrl_time_tot', 'B_Ctrl_time_tot', 'details','A_Head_Strikes_land',
       'A_Head_Strikes_att', 'B_Head_Strikes_land', 'B_Head_Strikes_att',
       'A_Head_Strikes_percent', 'B_Head_Strikes_percent',
       'A_Body_Strikes_land', 'A_Body_Strikes_att', 'B_Body_Strikes_land',
       'B_Body_Strikes_att', 'B_Body_Strikes_percent', 'A_Leg_Strikes_land', 'A_Leg
       'B_Leg_Strikes_land', 'B_Leg_Strikes_att', 'A_Leg_Strikes_percent',
       'B_Leg_Strikes_percent', 'A_Distance_Strikes_land',
       'A_Distance_Strikes_att', 'B_Distance_Strikes_land',
       'B_Distance_Strikes_att', 'A_Distance_Strikes_percent',
       'B_Distance_Strikes_percent', 'A_Clinch_Strikes_land',
```

```
       'A_Clinch_Strikes_att', 'B_Clinch_Strikes_land', 'B_Clinch_Strikes_att',
       'A_Clinch_Strikes_percent', 'B_Clinch_Strikes_percent',
       'A_Ground_Strikes_land', 'A_Ground_Strikes_att',
       'B_Ground_Strikes_land', 'B_Ground_Strikes_att',
       'A_Ground_Strikes_percent', 'B_Ground_Strikes_percent',  'A_Body_Strikes_per

df.drop(columns=to_drop, inplace=True)
```

In [ ]:
```
# drop all Dif Rows
dif_rows = ['Dif_Kd', 'Dif_Sig_strike_land', 'Dif_Sig_strike_att', 'Dif_Sig_strike_
 'Dif_Total_Strikes_land', 'Dif_Total_Strikes_att', 'Dif_Total_Strikes_percent', 'D
 'Dif_Takedowns_att', 'Dif_Takedown_percent', 'Dif_Sub_Attempts_land', 'Dif_Sub_Att
 'Dif_Rev', 'Dif_Ctrl_time_min', 'Dif_Ctrl_time_sec', 'Dif_Ctrl_time_tot', 'Dif_Hea
 'Dif_Head_Strikes_att', 'Dif_Head_Strikes_percent', 'Dif_Body_Strikes_land', 'Dif_
 'Dif_Body_Strikes_percent', 'Dif_Leg_Strikes_land', 'Dif_Leg_Strikes_att', 'Dif_Le
 'Dif_Distance_Strikes_land', 'Dif_Distance_Strikes_att', 'Dif_Distance_Strikes_per
 'Dif_Clinch_Strikes_att', 'Dif_Clinch_Strikes_percent', 'Dif_Ground_Strikes_land',
 'Dif_Ground_Strikes_percent']

df.drop(columns=dif_rows, inplace=True)
```

Check NAN

In [ ]:
```
# Check NAN
missing = df.isna().sum()
missing = pd.DataFrame(missing)
missing = missing.loc[missing[0] > 0]
missing
```

Out[ ]:

	0
A_topdown_Avg_Kd_per_round	892
A_topdown_Avg_Sig_strike_land_per_round	14
A_topdown_Avg_Sig_strike_att_per_round	2
A_topdown_Avg_Total_Strikes_land_per_round	8
A_topdown_Avg_Total_Strikes_att_per_round	2
...	...
Dif_Opp_Avg_Distance_Strikes_att_per_round	604
Dif_Opp_Avg_Clinch_Strikes_land_per_round	732
Dif_Opp_Avg_Clinch_Strikes_att_per_round	698
Dif_Opp_Avg_Ground_Strikes_land_per_round	778
Dif_Opp_Avg_Ground_Strikes_att_per_round	762

150 rows × 1 columns

In [ ]:
```
# replace NAN with 0
df.fillna(0, inplace=True)
```

In [ ]:
```
# replace and INF with 0
df.replace([np.inf, -np.inf], 0, inplace=True)
```

In [ ]:
```
all_cols = df.columns.to_list()
all_cols
```

Out[ ]:
```
['Fighter_A',
 'Fighter_B',
 'event_title',
 'event_url',
 'date',
 'Winner',
 'fight_id',
 'event_code',
 'Fighter_A_Odds',
 'Fighter_B_Odds',
 'Fighter_A_Odds_Change',
 'Fighter_B_Odds_Change',
 'Dif_Odds',
 'A_Rolling_Kd_mean',
 'B_Rolling_Kd_mean',
 'A_Rolling_Kd_std',
 'B_Rolling_Kd_std',
 'A_Rolling_Kd_median',
 'B_Rolling_Kd_median',
 'A_Rolling_Sig_strike_land_mean',
 'B_Rolling_Sig_strike_land_mean',
 'A_Rolling_Sig_strike_land_std',
 'B_Rolling_Sig_strike_land_std',
 'A_Rolling_Sig_strike_land_median',
 'B_Rolling_Sig_strike_land_median',
 'A_Rolling_Sig_strike_att_mean',
 'B_Rolling_Sig_strike_att_mean',
 'A_Rolling_Sig_strike_att_std',
 'B_Rolling_Sig_strike_att_std',
 'A_Rolling_Sig_strike_att_median',
 'B_Rolling_Sig_strike_att_median',
 'A_Rolling_Sig_strike_percent_mean',
 'B_Rolling_Sig_strike_percent_mean',
 'A_Rolling_Sig_strike_percent_std',
 'B_Rolling_Sig_strike_percent_std',
 'A_Rolling_Sig_strike_percent_median',
 'B_Rolling_Sig_strike_percent_median',
 'A_Rolling_Total_Strikes_land_mean',
 'B_Rolling_Total_Strikes_land_mean',
 'A_Rolling_Total_Strikes_land_std',
 'B_Rolling_Total_Strikes_land_std',
 'A_Rolling_Total_Strikes_land_median',
 'B_Rolling_Total_Strikes_land_median',
 'A_Rolling_Total_Strikes_att_mean',
 'B_Rolling_Total_Strikes_att_mean',
 'A_Rolling_Total_Strikes_att_std',
 'B_Rolling_Total_Strikes_att_std',
 'A_Rolling_Total_Strikes_att_median',
 'B_Rolling_Total_Strikes_att_median',
 'A_Rolling_Total_Strikes_percent_mean',
 'B_Rolling_Total_Strikes_percent_mean',
 'A_Rolling_Total_Strikes_percent_std',
 'B_Rolling_Total_Strikes_percent_std',
 'A_Rolling_Total_Strikes_percent_median',
 'B_Rolling_Total_Strikes_percent_median',
 'A_Rolling_Takedowns_land_mean',
```

```
'B_Rolling_Takedowns_land_mean',
'A_Rolling_Takedowns_land_std',
'B_Rolling_Takedowns_land_std',
'A_Rolling_Takedowns_land_median',
'B_Rolling_Takedowns_land_median',
'A_Rolling_Takedowns_att_mean',
'B_Rolling_Takedowns_att_mean',
'A_Rolling_Takedowns_att_std',
'B_Rolling_Takedowns_att_std',
'A_Rolling_Takedowns_att_median',
'B_Rolling_Takedowns_att_median',
'A_Rolling_Takedown_percent_mean',
'B_Rolling_Takedown_percent_mean',
'A_Rolling_Takedown_percent_std',
'B_Rolling_Takedown_percent_std',
'A_Rolling_Takedown_percent_median',
'B_Rolling_Takedown_percent_median',
'A_Rolling_Sub_Attempts_land_mean',
'B_Rolling_Sub_Attempts_land_mean',
'A_Rolling_Sub_Attempts_land_std',
'B_Rolling_Sub_Attempts_land_std',
'A_Rolling_Sub_Attempts_land_median',
'B_Rolling_Sub_Attempts_land_median',
'A_Rolling_Sub_Attempts_att_mean',
'B_Rolling_Sub_Attempts_att_mean',
'A_Rolling_Sub_Attempts_att_std',
'B_Rolling_Sub_Attempts_att_std',
'A_Rolling_Sub_Attempts_att_median',
'B_Rolling_Sub_Attempts_att_median',
'A_Rolling_Rev_mean',
'B_Rolling_Rev_mean',
'A_Rolling_Rev_std',
'B_Rolling_Rev_std',
'A_Rolling_Rev_median',
'B_Rolling_Rev_median',
'A_Rolling_Ctrl_time_min_mean',
'B_Rolling_Ctrl_time_min_mean',
'A_Rolling_Ctrl_time_min_std',
'B_Rolling_Ctrl_time_min_std',
'A_Rolling_Ctrl_time_min_median',
'B_Rolling_Ctrl_time_min_median',
'A_Rolling_Ctrl_time_sec_mean',
'B_Rolling_Ctrl_time_sec_mean',
'A_Rolling_Ctrl_time_sec_std',
'B_Rolling_Ctrl_time_sec_std',
'A_Rolling_Ctrl_time_sec_median',
'B_Rolling_Ctrl_time_sec_median',
'A_Rolling_Ctrl_time_tot_mean',
'B_Rolling_Ctrl_time_tot_mean',
'A_Rolling_Ctrl_time_tot_std',
'B_Rolling_Ctrl_time_tot_std',
'A_Rolling_Ctrl_time_tot_median',
'B_Rolling_Ctrl_time_tot_median',
'A_Rolling_Head_Strikes_land_mean',
'B_Rolling_Head_Strikes_land_mean',
'A_Rolling_Head_Strikes_land_std',
```

```
'B_Rolling_Head_Strikes_land_std',
'A_Rolling_Head_Strikes_land_median',
'B_Rolling_Head_Strikes_land_median',
'A_Rolling_Head_Strikes_att_mean',
'B_Rolling_Head_Strikes_att_mean',
'A_Rolling_Head_Strikes_att_std',
'B_Rolling_Head_Strikes_att_std',
'A_Rolling_Head_Strikes_att_median',
'B_Rolling_Head_Strikes_att_median',
'A_Rolling_Head_Strikes_percent_mean',
'B_Rolling_Head_Strikes_percent_mean',
'A_Rolling_Head_Strikes_percent_std',
'B_Rolling_Head_Strikes_percent_std',
'A_Rolling_Head_Strikes_percent_median',
'B_Rolling_Head_Strikes_percent_median',
'A_Rolling_Body_Strikes_land_mean',
'B_Rolling_Body_Strikes_land_mean',
'A_Rolling_Body_Strikes_land_std',
'B_Rolling_Body_Strikes_land_std',
'A_Rolling_Body_Strikes_land_median',
'B_Rolling_Body_Strikes_land_median',
'A_Rolling_Body_Strikes_att_mean',
'B_Rolling_Body_Strikes_att_mean',
'A_Rolling_Body_Strikes_att_std',
'B_Rolling_Body_Strikes_att_std',
'A_Rolling_Body_Strikes_att_median',
'B_Rolling_Body_Strikes_att_median',
'A_Rolling_Body_Strikes_percent_mean',
'B_Rolling_Body_Strikes_percent_mean',
'A_Rolling_Body_Strikes_percent_std',
'B_Rolling_Body_Strikes_percent_std',
'A_Rolling_Body_Strikes_percent_median',
'B_Rolling_Body_Strikes_percent_median',
'A_Rolling_Leg_Strikes_land_mean',
'B_Rolling_Leg_Strikes_land_mean',
'A_Rolling_Leg_Strikes_land_std',
'B_Rolling_Leg_Strikes_land_std',
'A_Rolling_Leg_Strikes_land_median',
'B_Rolling_Leg_Strikes_land_median',
'A_Rolling_Leg_Strikes_att_mean',
'B_Rolling_Leg_Strikes_att_mean',
'A_Rolling_Leg_Strikes_att_std',
'B_Rolling_Leg_Strikes_att_std',
'A_Rolling_Leg_Strikes_att_median',
'B_Rolling_Leg_Strikes_att_median',
'A_Rolling_Leg_Strikes_percent_mean',
'B_Rolling_Leg_Strikes_percent_mean',
'A_Rolling_Leg_Strikes_percent_std',
'B_Rolling_Leg_Strikes_percent_std',
'A_Rolling_Leg_Strikes_percent_median',
'B_Rolling_Leg_Strikes_percent_median',
'A_Rolling_Distance_Strikes_land_mean',
'B_Rolling_Distance_Strikes_land_mean',
'A_Rolling_Distance_Strikes_land_std',
'B_Rolling_Distance_Strikes_land_std',
'A_Rolling_Distance_Strikes_land_median',
```

```
'B_Rolling_Distance_Strikes_land_median',
'A_Rolling_Distance_Strikes_att_mean',
'B_Rolling_Distance_Strikes_att_mean',
'A_Rolling_Distance_Strikes_att_std',
'B_Rolling_Distance_Strikes_att_std',
'A_Rolling_Distance_Strikes_att_median',
'B_Rolling_Distance_Strikes_att_median',
'A_Rolling_Distance_Strikes_percent_mean',
'B_Rolling_Distance_Strikes_percent_mean',
'A_Rolling_Distance_Strikes_percent_std',
'B_Rolling_Distance_Strikes_percent_std',
'A_Rolling_Distance_Strikes_percent_median',
'B_Rolling_Distance_Strikes_percent_median',
'A_Rolling_Clinch_Strikes_land_mean',
'B_Rolling_Clinch_Strikes_land_mean',
'A_Rolling_Clinch_Strikes_land_std',
'B_Rolling_Clinch_Strikes_land_std',
'A_Rolling_Clinch_Strikes_land_median',
'B_Rolling_Clinch_Strikes_land_median',
'A_Rolling_Clinch_Strikes_att_mean',
'B_Rolling_Clinch_Strikes_att_mean',
'A_Rolling_Clinch_Strikes_att_std',
'B_Rolling_Clinch_Strikes_att_std',
'A_Rolling_Clinch_Strikes_att_median',
'B_Rolling_Clinch_Strikes_att_median',
'A_Rolling_Clinch_Strikes_percent_mean',
'B_Rolling_Clinch_Strikes_percent_mean',
'A_Rolling_Clinch_Strikes_percent_std',
'B_Rolling_Clinch_Strikes_percent_std',
'A_Rolling_Clinch_Strikes_percent_median',
'B_Rolling_Clinch_Strikes_percent_median',
'A_Rolling_Ground_Strikes_land_mean',
'B_Rolling_Ground_Strikes_land_mean',
'A_Rolling_Ground_Strikes_land_std',
'B_Rolling_Ground_Strikes_land_std',
'A_Rolling_Ground_Strikes_land_median',
'B_Rolling_Ground_Strikes_land_median',
'A_Rolling_Ground_Strikes_att_mean',
'B_Rolling_Ground_Strikes_att_mean',
'A_Rolling_Ground_Strikes_att_std',
'B_Rolling_Ground_Strikes_att_std',
'A_Rolling_Ground_Strikes_att_median',
'B_Rolling_Ground_Strikes_att_median',
'A_Rolling_Ground_Strikes_percent_mean',
'B_Rolling_Ground_Strikes_percent_mean',
'A_Rolling_Ground_Strikes_percent_std',
'B_Rolling_Ground_Strikes_percent_std',
'A_Rolling_Ground_Strikes_percent_median',
'B_Rolling_Ground_Strikes_percent_median',
'A_topdown_Avg_Kd',
'B_topdown_Avg_Kd',
'A_topdown_Avg_Sig_strike_land',
'B_topdown_Avg_Sig_strike_land',
'A_topdown_Avg_Sig_strike_att',
'B_topdown_Avg_Sig_strike_att',
'A_topdown_Avg_Sig_strike_percent',
```

```
'B_topdown_Avg_Sig_strike_percent',
'A_topdown_Avg_Total_Strikes_land',
'B_topdown_Avg_Total_Strikes_land',
'A_topdown_Avg_Total_Strikes_att',
'B_topdown_Avg_Total_Strikes_att',
'A_topdown_Avg_Total_Strikes_percent',
'B_topdown_Avg_Total_Strikes_percent',
'A_topdown_Avg_Takedowns_land',
'B_topdown_Avg_Takedowns_land',
'A_topdown_Avg_Takedowns_att',
'B_topdown_Avg_Takedowns_att',
'A_topdown_Avg_Takedown_percent',
'B_topdown_Avg_Takedown_percent',
'A_topdown_Avg_Sub_Attempts_land',
'B_topdown_Avg_Sub_Attempts_land',
'A_topdown_Avg_Sub_Attempts_att',
'B_topdown_Avg_Sub_Attempts_att',
'A_topdown_Avg_Rev',
'B_topdown_Avg_Rev',
'A_topdown_Avg_Ctrl_time_min',
'B_topdown_Avg_Ctrl_time_min',
'A_topdown_Avg_Ctrl_time_sec',
'B_topdown_Avg_Ctrl_time_sec',
'A_topdown_Avg_Ctrl_time_tot',
'B_topdown_Avg_Ctrl_time_tot',
'A_topdown_Avg_Head_Strikes_land',
'B_topdown_Avg_Head_Strikes_land',
'A_topdown_Avg_Head_Strikes_att',
'B_topdown_Avg_Head_Strikes_att',
'A_topdown_Avg_Head_Strikes_percent',
'B_topdown_Avg_Head_Strikes_percent',
'A_topdown_Avg_Body_Strikes_land',
'B_topdown_Avg_Body_Strikes_land',
'A_topdown_Avg_Body_Strikes_att',
'B_topdown_Avg_Body_Strikes_att',
'A_topdown_Avg_Body_Strikes_percent',
'B_topdown_Avg_Body_Strikes_percent',
'A_topdown_Avg_Leg_Strikes_land',
'B_topdown_Avg_Leg_Strikes_land',
'A_topdown_Avg_Leg_Strikes_att',
'B_topdown_Avg_Leg_Strikes_att',
'A_topdown_Avg_Leg_Strikes_percent',
'B_topdown_Avg_Leg_Strikes_percent',
'A_topdown_Avg_Distance_Strikes_land',
'B_topdown_Avg_Distance_Strikes_land',
'A_topdown_Avg_Distance_Strikes_att',
'B_topdown_Avg_Distance_Strikes_att',
'A_topdown_Avg_Distance_Strikes_percent',
'B_topdown_Avg_Distance_Strikes_percent',
'A_topdown_Avg_Clinch_Strikes_land',
'B_topdown_Avg_Clinch_Strikes_land',
'A_topdown_Avg_Clinch_Strikes_att',
'B_topdown_Avg_Clinch_Strikes_att',
'A_topdown_Avg_Clinch_Strikes_percent',
'B_topdown_Avg_Clinch_Strikes_percent',
'A_topdown_Avg_Ground_Strikes_land',
```

```
'B_topdown_Avg_Ground_Strikes_land',
'A_topdown_Avg_Ground_Strikes_att',
'B_topdown_Avg_Ground_Strikes_att',
'A_topdown_Avg_Ground_Strikes_percent',
'B_topdown_Avg_Ground_Strikes_percent',
'A_Opp_Avg_Kd',
'B_Opp_Avg_Kd',
'A_Opp_Avg_Sig_strike_land',
'B_Opp_Avg_Sig_strike_land',
'A_Opp_Avg_Sig_strike_att',
'B_Opp_Avg_Sig_strike_att',
'A_Opp_Avg_Sig_strike_percent',
'B_Opp_Avg_Sig_strike_percent',
'A_Opp_Avg_Total_Strikes_land',
'B_Opp_Avg_Total_Strikes_land',
'A_Opp_Avg_Total_Strikes_att',
'B_Opp_Avg_Total_Strikes_att',
'A_Opp_Avg_Total_Strikes_percent',
'B_Opp_Avg_Total_Strikes_percent',
'A_Opp_Avg_Takedowns_land',
'B_Opp_Avg_Takedowns_land',
'A_Opp_Avg_Takedowns_att',
'B_Opp_Avg_Takedowns_att',
'A_Opp_Avg_Takedown_percent',
'B_Opp_Avg_Takedown_percent',
'A_Opp_Avg_Sub_Attempts_land',
'B_Opp_Avg_Sub_Attempts_land',
'A_Opp_Avg_Sub_Attempts_att',
'B_Opp_Avg_Sub_Attempts_att',
'A_Opp_Avg_Rev',
'B_Opp_Avg_Rev',
'A_Opp_Avg_Ctrl_time_min',
'B_Opp_Avg_Ctrl_time_min',
'A_Opp_Avg_Ctrl_time_sec',
'B_Opp_Avg_Ctrl_time_sec',
'A_Opp_Avg_Ctrl_time_tot',
'B_Opp_Avg_Ctrl_time_tot',
'A_Opp_Avg_Head_Strikes_land',
'B_Opp_Avg_Head_Strikes_land',
'A_Opp_Avg_Head_Strikes_att',
'B_Opp_Avg_Head_Strikes_att',
'A_Opp_Avg_Head_Strikes_percent',
'B_Opp_Avg_Head_Strikes_percent',
'A_Opp_Avg_Body_Strikes_land',
'B_Opp_Avg_Body_Strikes_land',
'A_Opp_Avg_Body_Strikes_att',
'B_Opp_Avg_Body_Strikes_att',
'A_Opp_Avg_Body_Strikes_percent',
'B_Opp_Avg_Body_Strikes_percent',
'A_Opp_Avg_Leg_Strikes_land',
'B_Opp_Avg_Leg_Strikes_land',
'A_Opp_Avg_Leg_Strikes_att',
'B_Opp_Avg_Leg_Strikes_att',
'A_Opp_Avg_Leg_Strikes_percent',
'B_Opp_Avg_Leg_Strikes_percent',
'A_Opp_Avg_Distance_Strikes_land',
```

```
'B_Opp_Avg_Distance_Strikes_land',
'A_Opp_Avg_Distance_Strikes_att',
'B_Opp_Avg_Distance_Strikes_att',
'A_Opp_Avg_Distance_Strikes_percent',
'B_Opp_Avg_Distance_Strikes_percent',
'A_Opp_Avg_Clinch_Strikes_land',
'B_Opp_Avg_Clinch_Strikes_land',
'A_Opp_Avg_Clinch_Strikes_att',
'B_Opp_Avg_Clinch_Strikes_att',
'A_Opp_Avg_Clinch_Strikes_percent',
'B_Opp_Avg_Clinch_Strikes_percent',
'A_Opp_Avg_Ground_Strikes_land',
'B_Opp_Avg_Ground_Strikes_land',
'A_Opp_Avg_Ground_Strikes_att',
'B_Opp_Avg_Ground_Strikes_att',
'A_Opp_Avg_Ground_Strikes_percent',
'B_Opp_Avg_Ground_Strikes_percent',
'Dif_Rolling_Kd_mean',
'Dif_Rolling_Sig_strike_land_mean',
'Dif_Rolling_Sig_strike_att_mean',
'Dif_Rolling_Sig_strike_percent_mean',
'Dif_Rolling_Total_Strikes_land_mean',
'Dif_Rolling_Total_Strikes_att_mean',
'Dif_Rolling_Total_Strikes_percent_mean',
'Dif_Rolling_Takedowns_land_mean',
'Dif_Rolling_Takedowns_att_mean',
'Dif_Rolling_Takedown_percent_mean',
'Dif_Rolling_Sub_Attempts_land_mean',
'Dif_Rolling_Sub_Attempts_att_mean',
'Dif_Rolling_Rev_mean',
'Dif_Rolling_Ctrl_time_min_mean',
'Dif_Rolling_Ctrl_time_sec_mean',
'Dif_Rolling_Ctrl_time_tot_mean',
'Dif_Rolling_Head_Strikes_land_mean',
'Dif_Rolling_Head_Strikes_att_mean',
'Dif_Rolling_Head_Strikes_percent_mean',
'Dif_Rolling_Body_Strikes_land_mean',
'Dif_Rolling_Body_Strikes_att_mean',
'Dif_Rolling_Body_Strikes_percent_mean',
'Dif_Rolling_Leg_Strikes_land_mean',
'Dif_Rolling_Leg_Strikes_att_mean',
'Dif_Rolling_Leg_Strikes_percent_mean',
'Dif_Rolling_Distance_Strikes_land_mean',
'Dif_Rolling_Distance_Strikes_att_mean',
'Dif_Rolling_Distance_Strikes_percent_mean',
'Dif_Rolling_Clinch_Strikes_land_mean',
'Dif_Rolling_Clinch_Strikes_att_mean',
'Dif_Rolling_Clinch_Strikes_percent_mean',
'Dif_Rolling_Ground_Strikes_land_mean',
'Dif_Rolling_Ground_Strikes_att_mean',
'Dif_Rolling_Ground_Strikes_percent_mean',
'Dif_Rolling_Kd_median',
'Dif_Rolling_Sig_strike_land_median',
'Dif_Rolling_Sig_strike_att_median',
'Dif_Rolling_Sig_strike_percent_median',
'Dif_Rolling_Total_Strikes_land_median',
```

```
'Dif_Rolling_Total_Strikes_att_median',
'Dif_Rolling_Total_Strikes_percent_median',
'Dif_Rolling_Takedowns_land_median',
'Dif_Rolling_Takedowns_att_median',
'Dif_Rolling_Takedown_percent_median',
'Dif_Rolling_Sub_Attempts_land_median',
'Dif_Rolling_Sub_Attempts_att_median',
'Dif_Rolling_Rev_median',
'Dif_Rolling_Ctrl_time_min_median',
'Dif_Rolling_Ctrl_time_sec_median',
'Dif_Rolling_Ctrl_time_tot_median',
'Dif_Rolling_Head_Strikes_land_median',
'Dif_Rolling_Head_Strikes_att_median',
'Dif_Rolling_Head_Strikes_percent_median',
'Dif_Rolling_Body_Strikes_land_median',
'Dif_Rolling_Body_Strikes_att_median',
'Dif_Rolling_Body_Strikes_percent_median',
'Dif_Rolling_Leg_Strikes_land_median',
'Dif_Rolling_Leg_Strikes_att_median',
'Dif_Rolling_Leg_Strikes_percent_median',
'Dif_Rolling_Distance_Strikes_land_median',
'Dif_Rolling_Distance_Strikes_att_median',
'Dif_Rolling_Distance_Strikes_percent_median',
'Dif_Rolling_Clinch_Strikes_land_median',
'Dif_Rolling_Clinch_Strikes_att_median',
'Dif_Rolling_Clinch_Strikes_percent_median',
'Dif_Rolling_Ground_Strikes_land_median',
'Dif_Rolling_Ground_Strikes_att_median',
'Dif_Rolling_Ground_Strikes_percent_median',
'Dif_Rolling_Kd_std',
'Dif_Rolling_Sig_strike_land_std',
'Dif_Rolling_Sig_strike_att_std',
'Dif_Rolling_Sig_strike_percent_std',
'Dif_Rolling_Total_Strikes_land_std',
'Dif_Rolling_Total_Strikes_att_std',
'Dif_Rolling_Total_Strikes_percent_std',
'Dif_Rolling_Takedowns_land_std',
'Dif_Rolling_Takedowns_att_std',
'Dif_Rolling_Takedown_percent_std',
'Dif_Rolling_Sub_Attempts_land_std',
'Dif_Rolling_Sub_Attempts_att_std',
'Dif_Rolling_Rev_std',
'Dif_Rolling_Ctrl_time_min_std',
'Dif_Rolling_Ctrl_time_sec_std',
'Dif_Rolling_Ctrl_time_tot_std',
'Dif_Rolling_Head_Strikes_land_std',
'Dif_Rolling_Head_Strikes_att_std',
'Dif_Rolling_Head_Strikes_percent_std',
'Dif_Rolling_Body_Strikes_land_std',
'Dif_Rolling_Body_Strikes_att_std',
'Dif_Rolling_Body_Strikes_percent_std',
'Dif_Rolling_Leg_Strikes_land_std',
'Dif_Rolling_Leg_Strikes_att_std',
'Dif_Rolling_Leg_Strikes_percent_std',
'Dif_Rolling_Distance_Strikes_land_std',
'Dif_Rolling_Distance_Strikes_att_std',
```

```
'Dif_Rolling_Distance_Strikes_percent_std',
'Dif_Rolling_Clinch_Strikes_land_std',
'Dif_Rolling_Clinch_Strikes_att_std',
'Dif_Rolling_Clinch_Strikes_percent_std',
'Dif_Rolling_Ground_Strikes_land_std',
'Dif_Rolling_Ground_Strikes_att_std',
'Dif_Rolling_Ground_Strikes_percent_std',
'A_Height',
'B_Height',
'Dif_Height',
'A_Reach',
'B_Reach',
'Dif_Reach',
'A_Leg_Reach',
'B_Leg_Reach',
'Dif_Leg_Reach',
'A_Reach_NA',
'B_Reach_NA',
'Reach_NA',
'A_Leg_Reach_NA',
'B_Leg_Reach_NA',
'Leg_Reach_NA',
'A_Typical_Weightclass',
'B_Typical_Weightclass',
'fight_weightclass',
'A_Fight_in_Typical_Weightclass',
'B_Fight_in_Typical_Weightclass',
'A_Ape_Index',
'B_Ape_Index',
'A_Leg_Index',
'B_Leg_Index',
'A_Leg_to_Wing_Index',
'B_Leg_to_Wing_Index',
'win?',
'favorite?',
'datetime',
'date_formatted',
'A_Total_UFC_Fights',
'B_Total_UFC_Fights',
'Dif_Total_UFC_Fights',
'A_UFC_Wins',
'B_UFC_Wins',
'Dif_UFC_Wins',
'A_UFC_Losses',
'B_UFC_Losses',
'Dif_UFC_Losses',
'A_UFC_Win_Percentage',
'B_UFC_Win_Percentage',
'Dif_UFC_Win_Percentage',
'A_Last5_Win_Percentage',
'B_Last5_Win_Percentage',
'Dif_Last5_Win_Percentage',
'A_Last3_Win_Percentage',
'B_Last3_Win_Percentage',
'Dif_Last3_Win_Percentage',
'A_Win_By_KO_Percentage',
```

```
'B_Win_By_KO_Percentage',
'Dif_Win_By_KO_Percentage',
'A_Loss_By_KO_Percentage',
'B_Loss_By_KO_Percentage',
'Dif_Loss_By_KO_Percentage',
'A_Win_By_Decision_Percentage',
'B_Win_By_Decision_Percentage',
'Dif_Win_By_Decision_Percentage',
'A_Loss_By_Decision_Percentage',
'B_Loss_By_Decision_Percentage',
'Dif_Loss_By_Decision_Percentage',
'final_round_seconds',
'A_UFC_Fight_Time_Seconds',
'B_UFC_Fight_Time_Seconds',
'Dif_UFC_Fight_Time_Seconds',
'A_UFC_Fight_Rounds',
'B_UFC_Fight_Rounds',
'A_topdown_Avg_Kd_per_round',
'A_topdown_Avg_Sig_strike_land_per_round',
'A_topdown_Avg_Sig_strike_att_per_round',
'A_topdown_Avg_Total_Strikes_land_per_round',
'A_topdown_Avg_Total_Strikes_att_per_round',
'A_topdown_Avg_Takedowns_land_per_round',
'A_topdown_Avg_Takedowns_att_per_round',
'A_topdown_Avg_Sub_Attempts_land_per_round',
'A_topdown_Avg_Sub_Attempts_att_per_round',
'A_topdown_Avg_Rev_per_round',
'A_topdown_Avg_Ctrl_time_min_per_round',
'A_topdown_Avg_Ctrl_time_sec_per_round',
'A_topdown_Avg_Ctrl_time_tot_per_round',
'A_topdown_Avg_Head_Strikes_land_per_round',
'A_topdown_Avg_Head_Strikes_att_per_round',
'A_topdown_Avg_Body_Strikes_land_per_round',
'A_topdown_Avg_Body_Strikes_att_per_round',
'A_topdown_Avg_Leg_Strikes_land_per_round',
'A_topdown_Avg_Leg_Strikes_att_per_round',
'A_topdown_Avg_Distance_Strikes_land_per_round',
'A_topdown_Avg_Distance_Strikes_att_per_round',
'A_topdown_Avg_Clinch_Strikes_land_per_round',
'A_topdown_Avg_Clinch_Strikes_att_per_round',
'A_topdown_Avg_Ground_Strikes_land_per_round',
'A_topdown_Avg_Ground_Strikes_att_per_round',
'B_topdown_Avg_Kd_per_round',
'B_topdown_Avg_Sig_strike_land_per_round',
'B_topdown_Avg_Sig_strike_att_per_round',
'B_topdown_Avg_Total_Strikes_land_per_round',
'B_topdown_Avg_Total_Strikes_att_per_round',
'B_topdown_Avg_Takedowns_land_per_round',
'B_topdown_Avg_Takedowns_att_per_round',
'B_topdown_Avg_Sub_Attempts_land_per_round',
'B_topdown_Avg_Sub_Attempts_att_per_round',
'B_topdown_Avg_Rev_per_round',
'B_topdown_Avg_Ctrl_time_min_per_round',
'B_topdown_Avg_Ctrl_time_sec_per_round',
'B_topdown_Avg_Ctrl_time_tot_per_round',
'B_topdown_Avg_Head_Strikes_land_per_round',
```

```
'B_topdown_Avg_Head_Strikes_att_per_round',
'B_topdown_Avg_Body_Strikes_land_per_round',
'B_topdown_Avg_Body_Strikes_att_per_round',
'B_topdown_Avg_Leg_Strikes_land_per_round',
'B_topdown_Avg_Leg_Strikes_att_per_round',
'B_topdown_Avg_Distance_Strikes_land_per_round',
'B_topdown_Avg_Distance_Strikes_att_per_round',
'B_topdown_Avg_Clinch_Strikes_land_per_round',
'B_topdown_Avg_Clinch_Strikes_att_per_round',
'B_topdown_Avg_Ground_Strikes_land_per_round',
'B_topdown_Avg_Ground_Strikes_att_per_round',
'A_Opp_Avg_Kd_per_round',
'A_Opp_Avg_Sig_strike_land_per_round',
'A_Opp_Avg_Sig_strike_att_per_round',
'A_Opp_Avg_Total_Strikes_land_per_round',
'A_Opp_Avg_Total_Strikes_att_per_round',
'A_Opp_Avg_Takedowns_land_per_round',
'A_Opp_Avg_Takedowns_att_per_round',
'A_Opp_Avg_Sub_Attempts_land_per_round',
'A_Opp_Avg_Sub_Attempts_att_per_round',
'A_Opp_Avg_Rev_per_round',
'A_Opp_Avg_Ctrl_time_min_per_round',
'A_Opp_Avg_Ctrl_time_sec_per_round',
'A_Opp_Avg_Ctrl_time_tot_per_round',
'A_Opp_Avg_Head_Strikes_land_per_round',
'A_Opp_Avg_Head_Strikes_att_per_round',
'A_Opp_Avg_Body_Strikes_land_per_round',
'A_Opp_Avg_Body_Strikes_att_per_round',
'A_Opp_Avg_Leg_Strikes_land_per_round',
'A_Opp_Avg_Leg_Strikes_att_per_round',
'A_Opp_Avg_Distance_Strikes_land_per_round',
'A_Opp_Avg_Distance_Strikes_att_per_round',
'A_Opp_Avg_Clinch_Strikes_land_per_round',
'A_Opp_Avg_Clinch_Strikes_att_per_round',
'A_Opp_Avg_Ground_Strikes_land_per_round',
'A_Opp_Avg_Ground_Strikes_att_per_round',
'B_Opp_Avg_Kd_per_round',
'B_Opp_Avg_Sig_strike_land_per_round',
'B_Opp_Avg_Sig_strike_att_per_round',
'B_Opp_Avg_Total_Strikes_land_per_round',
'B_Opp_Avg_Total_Strikes_att_per_round',
'B_Opp_Avg_Takedowns_land_per_round',
'B_Opp_Avg_Takedowns_att_per_round',
'B_Opp_Avg_Sub_Attempts_land_per_round',
'B_Opp_Avg_Sub_Attempts_att_per_round',
'B_Opp_Avg_Rev_per_round',
'B_Opp_Avg_Ctrl_time_min_per_round',
'B_Opp_Avg_Ctrl_time_sec_per_round',
'B_Opp_Avg_Ctrl_time_tot_per_round',
'B_Opp_Avg_Head_Strikes_land_per_round',
'B_Opp_Avg_Head_Strikes_att_per_round',
'B_Opp_Avg_Body_Strikes_land_per_round',
'B_Opp_Avg_Body_Strikes_att_per_round',
'B_Opp_Avg_Leg_Strikes_land_per_round',
'B_Opp_Avg_Leg_Strikes_att_per_round',
'B_Opp_Avg_Distance_Strikes_land_per_round',
```

```
'B_Opp_Avg_Distance_Strikes_att_per_round',
'B_Opp_Avg_Clinch_Strikes_land_per_round',
'B_Opp_Avg_Clinch_Strikes_att_per_round',
'B_Opp_Avg_Ground_Strikes_land_per_round',
'B_Opp_Avg_Ground_Strikes_att_per_round',
'Dif_topdown_Avg_Kd_per_round',
'Dif_topdown_Avg_Sig_strike_land_per_round',
'Dif_topdown_Avg_Sig_strike_att_per_round',
'Dif_topdown_Avg_Total_Strikes_land_per_round',
'Dif_topdown_Avg_Total_Strikes_att_per_round',
'Dif_topdown_Avg_Takedowns_land_per_round',
'Dif_topdown_Avg_Takedowns_att_per_round',
'Dif_topdown_Avg_Sub_Attempts_land_per_round',
'Dif_topdown_Avg_Sub_Attempts_att_per_round',
'Dif_topdown_Avg_Rev_per_round',
'Dif_topdown_Avg_Ctrl_time_min_per_round',
'Dif_topdown_Avg_Ctrl_time_sec_per_round',
'Dif_topdown_Avg_Ctrl_time_tot_per_round',
'Dif_topdown_Avg_Head_Strikes_land_per_round',
'Dif_topdown_Avg_Head_Strikes_att_per_round',
'Dif_topdown_Avg_Body_Strikes_land_per_round',
'Dif_topdown_Avg_Body_Strikes_att_per_round',
'Dif_topdown_Avg_Leg_Strikes_land_per_round',
'Dif_topdown_Avg_Leg_Strikes_att_per_round',
'Dif_topdown_Avg_Distance_Strikes_land_per_round',
'Dif_topdown_Avg_Distance_Strikes_att_per_round',
'Dif_topdown_Avg_Clinch_Strikes_land_per_round',
'Dif_topdown_Avg_Clinch_Strikes_att_per_round',
'Dif_topdown_Avg_Ground_Strikes_land_per_round',
'Dif_topdown_Avg_Ground_Strikes_att_per_round',
'Dif_Opp_Avg_Kd_per_round',
'Dif_Opp_Avg_Sig_strike_land_per_round',
'Dif_Opp_Avg_Sig_strike_att_per_round',
'Dif_Opp_Avg_Total_Strikes_land_per_round',
'Dif_Opp_Avg_Total_Strikes_att_per_round',
'Dif_Opp_Avg_Takedowns_land_per_round',
'Dif_Opp_Avg_Takedowns_att_per_round',
'Dif_Opp_Avg_Sub_Attempts_land_per_round',
'Dif_Opp_Avg_Sub_Attempts_att_per_round',
'Dif_Opp_Avg_Rev_per_round',
'Dif_Opp_Avg_Ctrl_time_min_per_round',
'Dif_Opp_Avg_Ctrl_time_sec_per_round',
'Dif_Opp_Avg_Ctrl_time_tot_per_round',
'Dif_Opp_Avg_Head_Strikes_land_per_round',
'Dif_Opp_Avg_Head_Strikes_att_per_round',
'Dif_Opp_Avg_Body_Strikes_land_per_round',
'Dif_Opp_Avg_Body_Strikes_att_per_round',
'Dif_Opp_Avg_Leg_Strikes_land_per_round',
'Dif_Opp_Avg_Leg_Strikes_att_per_round',
'Dif_Opp_Avg_Distance_Strikes_land_per_round',
'Dif_Opp_Avg_Distance_Strikes_att_per_round',
'Dif_Opp_Avg_Clinch_Strikes_land_per_round',
'Dif_Opp_Avg_Clinch_Strikes_att_per_round',
'Dif_Opp_Avg_Ground_Strikes_land_per_round',
'Dif_Opp_Avg_Ground_Strikes_att_per_round']
```

In [ ]:
```
# find weightclass columns
weightclass = [n for n in all_cols if 'Weightclass' in n]
weightclass
```

Out[ ]:
```
['A_Typical_Weightclass',
 'B_Typical_Weightclass',
 'A_Fight_in_Typical_Weightclass',
 'B_Fight_in_Typical_Weightclass']
```

In [ ]:
```
# check Winner column
df['Winner'].value_counts()
```

Out[ ]:
```
Jim Miller             46
Donald Cerrone         38
Charles Oliveira       38
Dustin Poirier         34
Jon Jones              34
                       ..
Daniel Spitz            2
Chris Kelades           2
Michihiro Omigawa       2
Christian Morecraft     2
Royston Wee             2
Name: Winner, Length: 1144, dtype: int64
```

In [ ]:
```
#df.drop(columns=in_fight_cols, inplace=True)
#df.drop(columns=in_fight_difs, inplace=True)
colz = list(df.columns)
# get favorite
#df['favorite?'] = np.where(df['Fighter_A_Odds_obf'] < 0, 1, 0)
```

In [ ]:
```
# check value counts
df['win?'].value_counts()
```

Out[ ]:
```
1    4069
0    4069
Name: win?, dtype: int64
```

In [ ]:
```
fav_check = df.groupby('favorite?')['win?'].value_counts()
fav_check
```

Out[ ]:
```
favorite?  win?
0          0       2696
           1       1443
1          1       2626
           0       1373
Name: win?, dtype: int64
```

In [ ]:
```
# Check how often winner is favorite
fav = df.loc[df['favorite?'] == 1]
fav['win?'].value_counts()
```

Out[ ]:
```
1    2626
0    1373
Name: win?, dtype: int64
```

Assign Target and Split

In [ ]:
```
target_name = "win?"
y = df[target_name]
X = df.drop(columns=[target_name])

# drop columns you dont want to use
to_drop = ['event_title','event_url','date', 'fight_id', 'Fighter_A', 'Fighter_B',
           'Winner', 'event_code', 'A_Typical_Weightclass', 'B_Typical_Weightclass
X = X.drop(columns=to_drop)

# Scoring Metric
class_metric = 'accuracy'
```

In [ ]:
```
numerical_columns_selector = selector(dtype_exclude=object)
categorical_columns_selector = selector(dtype_include=object)

numerical_columns = numerical_columns_selector(X)
categorical_columns = categorical_columns_selector(X)

categorical_columns
```

Out[ ]:
```
['fight_weightclass']
```

In [ ]:
```
numerical_columns
```

Out[ ]:
```
['Fighter_A_Odds',
 'Fighter_B_Odds',
 'Fighter_A_Odds_Change',
 'Fighter_B_Odds_Change',
 'Dif_Odds',
 'A_Rolling_Kd_mean',
 'B_Rolling_Kd_mean',
 'A_Rolling_Kd_std',
 'B_Rolling_Kd_std',
 'A_Rolling_Kd_median',
 'B_Rolling_Kd_median',
 'A_Rolling_Sig_strike_land_mean',
 'B_Rolling_Sig_strike_land_mean',
 'A_Rolling_Sig_strike_land_std',
 'B_Rolling_Sig_strike_land_std',
 'A_Rolling_Sig_strike_land_median',
 'B_Rolling_Sig_strike_land_median',
 'A_Rolling_Sig_strike_att_mean',
 'B_Rolling_Sig_strike_att_mean',
 'A_Rolling_Sig_strike_att_std',
 'B_Rolling_Sig_strike_att_std',
 'A_Rolling_Sig_strike_att_median',
 'B_Rolling_Sig_strike_att_median',
 'A_Rolling_Sig_strike_percent_mean',
 'B_Rolling_Sig_strike_percent_mean',
 'A_Rolling_Sig_strike_percent_std',
 'B_Rolling_Sig_strike_percent_std',
 'A_Rolling_Sig_strike_percent_median',
 'B_Rolling_Sig_strike_percent_median',
 'A_Rolling_Total_Strikes_land_mean',
 'B_Rolling_Total_Strikes_land_mean',
 'A_Rolling_Total_Strikes_land_std',
 'B_Rolling_Total_Strikes_land_std',
 'A_Rolling_Total_Strikes_land_median',
 'B_Rolling_Total_Strikes_land_median',
 'A_Rolling_Total_Strikes_att_mean',
 'B_Rolling_Total_Strikes_att_mean',
 'A_Rolling_Total_Strikes_att_std',
 'B_Rolling_Total_Strikes_att_std',
 'A_Rolling_Total_Strikes_att_median',
 'B_Rolling_Total_Strikes_att_median',
 'A_Rolling_Total_Strikes_percent_mean',
 'B_Rolling_Total_Strikes_percent_mean',
 'A_Rolling_Total_Strikes_percent_std',
 'B_Rolling_Total_Strikes_percent_std',
 'A_Rolling_Total_Strikes_percent_median',
 'B_Rolling_Total_Strikes_percent_median',
 'A_Rolling_Takedowns_land_mean',
 'B_Rolling_Takedowns_land_mean',
 'A_Rolling_Takedowns_land_std',
 'B_Rolling_Takedowns_land_std',
 'A_Rolling_Takedowns_land_median',
 'B_Rolling_Takedowns_land_median',
 'A_Rolling_Takedowns_att_mean',
 'B_Rolling_Takedowns_att_mean',
 'A_Rolling_Takedowns_att_std',
```

```
'B_Rolling_Takedowns_att_std',
'A_Rolling_Takedowns_att_median',
'B_Rolling_Takedowns_att_median',
'A_Rolling_Takedown_percent_mean',
'B_Rolling_Takedown_percent_mean',
'A_Rolling_Takedown_percent_std',
'B_Rolling_Takedown_percent_std',
'A_Rolling_Takedown_percent_median',
'B_Rolling_Takedown_percent_median',
'A_Rolling_Sub_Attempts_land_mean',
'B_Rolling_Sub_Attempts_land_mean',
'A_Rolling_Sub_Attempts_land_std',
'B_Rolling_Sub_Attempts_land_std',
'A_Rolling_Sub_Attempts_land_median',
'B_Rolling_Sub_Attempts_land_median',
'A_Rolling_Sub_Attempts_att_mean',
'B_Rolling_Sub_Attempts_att_mean',
'A_Rolling_Sub_Attempts_att_std',
'B_Rolling_Sub_Attempts_att_std',
'A_Rolling_Sub_Attempts_att_median',
'B_Rolling_Sub_Attempts_att_median',
'A_Rolling_Rev_mean',
'B_Rolling_Rev_mean',
'A_Rolling_Rev_std',
'B_Rolling_Rev_std',
'A_Rolling_Rev_median',
'B_Rolling_Rev_median',
'A_Rolling_Ctrl_time_min_mean',
'B_Rolling_Ctrl_time_min_mean',
'A_Rolling_Ctrl_time_min_std',
'B_Rolling_Ctrl_time_min_std',
'A_Rolling_Ctrl_time_min_median',
'B_Rolling_Ctrl_time_min_median',
'A_Rolling_Ctrl_time_sec_mean',
'B_Rolling_Ctrl_time_sec_mean',
'A_Rolling_Ctrl_time_sec_std',
'B_Rolling_Ctrl_time_sec_std',
'A_Rolling_Ctrl_time_sec_median',
'B_Rolling_Ctrl_time_sec_median',
'A_Rolling_Ctrl_time_tot_mean',
'B_Rolling_Ctrl_time_tot_mean',
'A_Rolling_Ctrl_time_tot_std',
'B_Rolling_Ctrl_time_tot_std',
'A_Rolling_Ctrl_time_tot_median',
'B_Rolling_Ctrl_time_tot_median',
'A_Rolling_Head_Strikes_land_mean',
'B_Rolling_Head_Strikes_land_mean',
'A_Rolling_Head_Strikes_land_std',
'B_Rolling_Head_Strikes_land_std',
'A_Rolling_Head_Strikes_land_median',
'B_Rolling_Head_Strikes_land_median',
'A_Rolling_Head_Strikes_att_mean',
'B_Rolling_Head_Strikes_att_mean',
'A_Rolling_Head_Strikes_att_std',
'B_Rolling_Head_Strikes_att_std',
'A_Rolling_Head_Strikes_att_median',
```

```
'B_Rolling_Head_Strikes_att_median',
'A_Rolling_Head_Strikes_percent_mean',
'B_Rolling_Head_Strikes_percent_mean',
'A_Rolling_Head_Strikes_percent_std',
'B_Rolling_Head_Strikes_percent_std',
'A_Rolling_Head_Strikes_percent_median',
'B_Rolling_Head_Strikes_percent_median',
'A_Rolling_Body_Strikes_land_mean',
'B_Rolling_Body_Strikes_land_mean',
'A_Rolling_Body_Strikes_land_std',
'B_Rolling_Body_Strikes_land_std',
'A_Rolling_Body_Strikes_land_median',
'B_Rolling_Body_Strikes_land_median',
'A_Rolling_Body_Strikes_att_mean',
'B_Rolling_Body_Strikes_att_mean',
'A_Rolling_Body_Strikes_att_std',
'B_Rolling_Body_Strikes_att_std',
'A_Rolling_Body_Strikes_att_median',
'B_Rolling_Body_Strikes_att_median',
'A_Rolling_Body_Strikes_percent_mean',
'B_Rolling_Body_Strikes_percent_mean',
'A_Rolling_Body_Strikes_percent_std',
'B_Rolling_Body_Strikes_percent_std',
'A_Rolling_Body_Strikes_percent_median',
'B_Rolling_Body_Strikes_percent_median',
'A_Rolling_Leg_Strikes_land_mean',
'B_Rolling_Leg_Strikes_land_mean',
'A_Rolling_Leg_Strikes_land_std',
'B_Rolling_Leg_Strikes_land_std',
'A_Rolling_Leg_Strikes_land_median',
'B_Rolling_Leg_Strikes_land_median',
'A_Rolling_Leg_Strikes_att_mean',
'B_Rolling_Leg_Strikes_att_mean',
'A_Rolling_Leg_Strikes_att_std',
'B_Rolling_Leg_Strikes_att_std',
'A_Rolling_Leg_Strikes_att_median',
'B_Rolling_Leg_Strikes_att_median',
'A_Rolling_Leg_Strikes_percent_mean',
'B_Rolling_Leg_Strikes_percent_mean',
'A_Rolling_Leg_Strikes_percent_std',
'B_Rolling_Leg_Strikes_percent_std',
'A_Rolling_Leg_Strikes_percent_median',
'B_Rolling_Leg_Strikes_percent_median',
'A_Rolling_Distance_Strikes_land_mean',
'B_Rolling_Distance_Strikes_land_mean',
'A_Rolling_Distance_Strikes_land_std',
'B_Rolling_Distance_Strikes_land_std',
'A_Rolling_Distance_Strikes_land_median',
'B_Rolling_Distance_Strikes_land_median',
'A_Rolling_Distance_Strikes_att_mean',
'B_Rolling_Distance_Strikes_att_mean',
'A_Rolling_Distance_Strikes_att_std',
'B_Rolling_Distance_Strikes_att_std',
'A_Rolling_Distance_Strikes_att_median',
'B_Rolling_Distance_Strikes_att_median',
'A_Rolling_Distance_Strikes_percent_mean',
```

```
'B_Rolling_Distance_Strikes_percent_mean',
'A_Rolling_Distance_Strikes_percent_std',
'B_Rolling_Distance_Strikes_percent_std',
'A_Rolling_Distance_Strikes_percent_median',
'B_Rolling_Distance_Strikes_percent_median',
'A_Rolling_Clinch_Strikes_land_mean',
'B_Rolling_Clinch_Strikes_land_mean',
'A_Rolling_Clinch_Strikes_land_std',
'B_Rolling_Clinch_Strikes_land_std',
'A_Rolling_Clinch_Strikes_land_median',
'B_Rolling_Clinch_Strikes_land_median',
'A_Rolling_Clinch_Strikes_att_mean',
'B_Rolling_Clinch_Strikes_att_mean',
'A_Rolling_Clinch_Strikes_att_std',
'B_Rolling_Clinch_Strikes_att_std',
'A_Rolling_Clinch_Strikes_att_median',
'B_Rolling_Clinch_Strikes_att_median',
'A_Rolling_Clinch_Strikes_percent_mean',
'B_Rolling_Clinch_Strikes_percent_mean',
'A_Rolling_Clinch_Strikes_percent_std',
'B_Rolling_Clinch_Strikes_percent_std',
'A_Rolling_Clinch_Strikes_percent_median',
'B_Rolling_Clinch_Strikes_percent_median',
'A_Rolling_Ground_Strikes_land_mean',
'B_Rolling_Ground_Strikes_land_mean',
'A_Rolling_Ground_Strikes_land_std',
'B_Rolling_Ground_Strikes_land_std',
'A_Rolling_Ground_Strikes_land_median',
'B_Rolling_Ground_Strikes_land_median',
'A_Rolling_Ground_Strikes_att_mean',
'B_Rolling_Ground_Strikes_att_mean',
'A_Rolling_Ground_Strikes_att_std',
'B_Rolling_Ground_Strikes_att_std',
'A_Rolling_Ground_Strikes_att_median',
'B_Rolling_Ground_Strikes_att_median',
'A_Rolling_Ground_Strikes_percent_mean',
'B_Rolling_Ground_Strikes_percent_mean',
'A_Rolling_Ground_Strikes_percent_std',
'B_Rolling_Ground_Strikes_percent_std',
'A_Rolling_Ground_Strikes_percent_median',
'B_Rolling_Ground_Strikes_percent_median',
'A_topdown_Avg_Kd',
'B_topdown_Avg_Kd',
'A_topdown_Avg_Sig_strike_land',
'B_topdown_Avg_Sig_strike_land',
'A_topdown_Avg_Sig_strike_att',
'B_topdown_Avg_Sig_strike_att',
'A_topdown_Avg_Sig_strike_percent',
'B_topdown_Avg_Sig_strike_percent',
'A_topdown_Avg_Total_Strikes_land',
'B_topdown_Avg_Total_Strikes_land',
'A_topdown_Avg_Total_Strikes_att',
'B_topdown_Avg_Total_Strikes_att',
'A_topdown_Avg_Total_Strikes_percent',
'B_topdown_Avg_Total_Strikes_percent',
'A_topdown_Avg_Takedowns_land',
```

```
'B_topdown_Avg_Takedowns_land',
'A_topdown_Avg_Takedowns_att',
'B_topdown_Avg_Takedowns_att',
'A_topdown_Avg_Takedown_percent',
'B_topdown_Avg_Takedown_percent',
'A_topdown_Avg_Sub_Attempts_land',
'B_topdown_Avg_Sub_Attempts_land',
'A_topdown_Avg_Sub_Attempts_att',
'B_topdown_Avg_Sub_Attempts_att',
'A_topdown_Avg_Rev',
'B_topdown_Avg_Rev',
'A_topdown_Avg_Ctrl_time_min',
'B_topdown_Avg_Ctrl_time_min',
'A_topdown_Avg_Ctrl_time_sec',
'B_topdown_Avg_Ctrl_time_sec',
'A_topdown_Avg_Ctrl_time_tot',
'B_topdown_Avg_Ctrl_time_tot',
'A_topdown_Avg_Head_Strikes_land',
'B_topdown_Avg_Head_Strikes_land',
'A_topdown_Avg_Head_Strikes_att',
'B_topdown_Avg_Head_Strikes_att',
'A_topdown_Avg_Head_Strikes_percent',
'B_topdown_Avg_Head_Strikes_percent',
'A_topdown_Avg_Body_Strikes_land',
'B_topdown_Avg_Body_Strikes_land',
'A_topdown_Avg_Body_Strikes_att',
'B_topdown_Avg_Body_Strikes_att',
'A_topdown_Avg_Body_Strikes_percent',
'B_topdown_Avg_Body_Strikes_percent',
'A_topdown_Avg_Leg_Strikes_land',
'B_topdown_Avg_Leg_Strikes_land',
'A_topdown_Avg_Leg_Strikes_att',
'B_topdown_Avg_Leg_Strikes_att',
'A_topdown_Avg_Leg_Strikes_percent',
'B_topdown_Avg_Leg_Strikes_percent',
'A_topdown_Avg_Distance_Strikes_land',
'B_topdown_Avg_Distance_Strikes_land',
'A_topdown_Avg_Distance_Strikes_att',
'B_topdown_Avg_Distance_Strikes_att',
'A_topdown_Avg_Distance_Strikes_percent',
'B_topdown_Avg_Distance_Strikes_percent',
'A_topdown_Avg_Clinch_Strikes_land',
'B_topdown_Avg_Clinch_Strikes_land',
'A_topdown_Avg_Clinch_Strikes_att',
'B_topdown_Avg_Clinch_Strikes_att',
'A_topdown_Avg_Clinch_Strikes_percent',
'B_topdown_Avg_Clinch_Strikes_percent',
'A_topdown_Avg_Ground_Strikes_land',
'B_topdown_Avg_Ground_Strikes_land',
'A_topdown_Avg_Ground_Strikes_att',
'B_topdown_Avg_Ground_Strikes_att',
'A_topdown_Avg_Ground_Strikes_percent',
'B_topdown_Avg_Ground_Strikes_percent',
'A_Opp_Avg_Kd',
'B_Opp_Avg_Kd',
'A_Opp_Avg_Sig_strike_land',
```

```
'B_Opp_Avg_Sig_strike_land',
'A_Opp_Avg_Sig_strike_att',
'B_Opp_Avg_Sig_strike_att',
'A_Opp_Avg_Sig_strike_percent',
'B_Opp_Avg_Sig_strike_percent',
'A_Opp_Avg_Total_Strikes_land',
'B_Opp_Avg_Total_Strikes_land',
'A_Opp_Avg_Total_Strikes_att',
'B_Opp_Avg_Total_Strikes_att',
'A_Opp_Avg_Total_Strikes_percent',
'B_Opp_Avg_Total_Strikes_percent',
'A_Opp_Avg_Takedowns_land',
'B_Opp_Avg_Takedowns_land',
'A_Opp_Avg_Takedowns_att',
'B_Opp_Avg_Takedowns_att',
'A_Opp_Avg_Takedown_percent',
'B_Opp_Avg_Takedown_percent',
'A_Opp_Avg_Sub_Attempts_land',
'B_Opp_Avg_Sub_Attempts_land',
'A_Opp_Avg_Sub_Attempts_att',
'B_Opp_Avg_Sub_Attempts_att',
'A_Opp_Avg_Rev',
'B_Opp_Avg_Rev',
'A_Opp_Avg_Ctrl_time_min',
'B_Opp_Avg_Ctrl_time_min',
'A_Opp_Avg_Ctrl_time_sec',
'B_Opp_Avg_Ctrl_time_sec',
'A_Opp_Avg_Ctrl_time_tot',
'B_Opp_Avg_Ctrl_time_tot',
'A_Opp_Avg_Head_Strikes_land',
'B_Opp_Avg_Head_Strikes_land',
'A_Opp_Avg_Head_Strikes_att',
'B_Opp_Avg_Head_Strikes_att',
'A_Opp_Avg_Head_Strikes_percent',
'B_Opp_Avg_Head_Strikes_percent',
'A_Opp_Avg_Body_Strikes_land',
'B_Opp_Avg_Body_Strikes_land',
'A_Opp_Avg_Body_Strikes_att',
'B_Opp_Avg_Body_Strikes_att',
'A_Opp_Avg_Body_Strikes_percent',
'B_Opp_Avg_Body_Strikes_percent',
'A_Opp_Avg_Leg_Strikes_land',
'B_Opp_Avg_Leg_Strikes_land',
'A_Opp_Avg_Leg_Strikes_att',
'B_Opp_Avg_Leg_Strikes_att',
'A_Opp_Avg_Leg_Strikes_percent',
'B_Opp_Avg_Leg_Strikes_percent',
'A_Opp_Avg_Distance_Strikes_land',
'B_Opp_Avg_Distance_Strikes_land',
'A_Opp_Avg_Distance_Strikes_att',
'B_Opp_Avg_Distance_Strikes_att',
'A_Opp_Avg_Distance_Strikes_percent',
'B_Opp_Avg_Distance_Strikes_percent',
'A_Opp_Avg_Clinch_Strikes_land',
'B_Opp_Avg_Clinch_Strikes_land',
'A_Opp_Avg_Clinch_Strikes_att',
```

```
'B_Opp_Avg_Clinch_Strikes_att',
'A_Opp_Avg_Clinch_Strikes_percent',
'B_Opp_Avg_Clinch_Strikes_percent',
'A_Opp_Avg_Ground_Strikes_land',
'B_Opp_Avg_Ground_Strikes_land',
'A_Opp_Avg_Ground_Strikes_att',
'B_Opp_Avg_Ground_Strikes_att',
'A_Opp_Avg_Ground_Strikes_percent',
'B_Opp_Avg_Ground_Strikes_percent',
'Dif_Rolling_Kd_mean',
'Dif_Rolling_Sig_strike_land_mean',
'Dif_Rolling_Sig_strike_att_mean',
'Dif_Rolling_Sig_strike_percent_mean',
'Dif_Rolling_Total_Strikes_land_mean',
'Dif_Rolling_Total_Strikes_att_mean',
'Dif_Rolling_Total_Strikes_percent_mean',
'Dif_Rolling_Takedowns_land_mean',
'Dif_Rolling_Takedowns_att_mean',
'Dif_Rolling_Takedown_percent_mean',
'Dif_Rolling_Sub_Attempts_land_mean',
'Dif_Rolling_Sub_Attempts_att_mean',
'Dif_Rolling_Rev_mean',
'Dif_Rolling_Ctrl_time_min_mean',
'Dif_Rolling_Ctrl_time_sec_mean',
'Dif_Rolling_Ctrl_time_tot_mean',
'Dif_Rolling_Head_Strikes_land_mean',
'Dif_Rolling_Head_Strikes_att_mean',
'Dif_Rolling_Head_Strikes_percent_mean',
'Dif_Rolling_Body_Strikes_land_mean',
'Dif_Rolling_Body_Strikes_att_mean',
'Dif_Rolling_Body_Strikes_percent_mean',
'Dif_Rolling_Leg_Strikes_land_mean',
'Dif_Rolling_Leg_Strikes_att_mean',
'Dif_Rolling_Leg_Strikes_percent_mean',
'Dif_Rolling_Distance_Strikes_land_mean',
'Dif_Rolling_Distance_Strikes_att_mean',
'Dif_Rolling_Distance_Strikes_percent_mean',
'Dif_Rolling_Clinch_Strikes_land_mean',
'Dif_Rolling_Clinch_Strikes_att_mean',
'Dif_Rolling_Clinch_Strikes_percent_mean',
'Dif_Rolling_Ground_Strikes_land_mean',
'Dif_Rolling_Ground_Strikes_att_mean',
'Dif_Rolling_Ground_Strikes_percent_mean',
'Dif_Rolling_Kd_median',
'Dif_Rolling_Sig_strike_land_median',
'Dif_Rolling_Sig_strike_att_median',
'Dif_Rolling_Sig_strike_percent_median',
'Dif_Rolling_Total_Strikes_land_median',
'Dif_Rolling_Total_Strikes_att_median',
'Dif_Rolling_Total_Strikes_percent_median',
'Dif_Rolling_Takedowns_land_median',
'Dif_Rolling_Takedowns_att_median',
'Dif_Rolling_Takedown_percent_median',
'Dif_Rolling_Sub_Attempts_land_median',
'Dif_Rolling_Sub_Attempts_att_median',
'Dif_Rolling_Rev_median',
```

```
'Dif_Rolling_Ctrl_time_min_median',
'Dif_Rolling_Ctrl_time_sec_median',
'Dif_Rolling_Ctrl_time_tot_median',
'Dif_Rolling_Head_Strikes_land_median',
'Dif_Rolling_Head_Strikes_att_median',
'Dif_Rolling_Head_Strikes_percent_median',
'Dif_Rolling_Body_Strikes_land_median',
'Dif_Rolling_Body_Strikes_att_median',
'Dif_Rolling_Body_Strikes_percent_median',
'Dif_Rolling_Leg_Strikes_land_median',
'Dif_Rolling_Leg_Strikes_att_median',
'Dif_Rolling_Leg_Strikes_percent_median',
'Dif_Rolling_Distance_Strikes_land_median',
'Dif_Rolling_Distance_Strikes_att_median',
'Dif_Rolling_Distance_Strikes_percent_median',
'Dif_Rolling_Clinch_Strikes_land_median',
'Dif_Rolling_Clinch_Strikes_att_median',
'Dif_Rolling_Clinch_Strikes_percent_median',
'Dif_Rolling_Ground_Strikes_land_median',
'Dif_Rolling_Ground_Strikes_att_median',
'Dif_Rolling_Ground_Strikes_percent_median',
'Dif_Rolling_Kd_std',
'Dif_Rolling_Sig_strike_land_std',
'Dif_Rolling_Sig_strike_att_std',
'Dif_Rolling_Sig_strike_percent_std',
'Dif_Rolling_Total_Strikes_land_std',
'Dif_Rolling_Total_Strikes_att_std',
'Dif_Rolling_Total_Strikes_percent_std',
'Dif_Rolling_Takedowns_land_std',
'Dif_Rolling_Takedowns_att_std',
'Dif_Rolling_Takedown_percent_std',
'Dif_Rolling_Sub_Attempts_land_std',
'Dif_Rolling_Sub_Attempts_att_std',
'Dif_Rolling_Rev_std',
'Dif_Rolling_Ctrl_time_min_std',
'Dif_Rolling_Ctrl_time_sec_std',
'Dif_Rolling_Ctrl_time_tot_std',
'Dif_Rolling_Head_Strikes_land_std',
'Dif_Rolling_Head_Strikes_att_std',
'Dif_Rolling_Head_Strikes_percent_std',
'Dif_Rolling_Body_Strikes_land_std',
'Dif_Rolling_Body_Strikes_att_std',
'Dif_Rolling_Body_Strikes_percent_std',
'Dif_Rolling_Leg_Strikes_land_std',
'Dif_Rolling_Leg_Strikes_att_std',
'Dif_Rolling_Leg_Strikes_percent_std',
'Dif_Rolling_Distance_Strikes_land_std',
'Dif_Rolling_Distance_Strikes_att_std',
'Dif_Rolling_Distance_Strikes_percent_std',
'Dif_Rolling_Clinch_Strikes_land_std',
'Dif_Rolling_Clinch_Strikes_att_std',
'Dif_Rolling_Clinch_Strikes_percent_std',
'Dif_Rolling_Ground_Strikes_land_std',
'Dif_Rolling_Ground_Strikes_att_std',
'Dif_Rolling_Ground_Strikes_percent_std',
'A_Height',
```

```
'B_Height',
'Dif_Height',
'A_Reach',
'B_Reach',
'Dif_Reach',
'A_Leg_Reach',
'B_Leg_Reach',
'Dif_Leg_Reach',
'A_Reach_NA',
'B_Reach_NA',
'Reach_NA',
'A_Leg_Reach_NA',
'B_Leg_Reach_NA',
'Leg_Reach_NA',
'A_Fight_in_Typical_Weightclass',
'B_Fight_in_Typical_Weightclass',
'A_Ape_Index',
'B_Ape_Index',
'A_Leg_Index',
'B_Leg_Index',
'A_Leg_to_Wing_Index',
'B_Leg_to_Wing_Index',
'favorite?',
'A_Total_UFC_Fights',
'B_Total_UFC_Fights',
'Dif_Total_UFC_Fights',
'A_UFC_Wins',
'B_UFC_Wins',
'Dif_UFC_Wins',
'A_UFC_Losses',
'B_UFC_Losses',
'Dif_UFC_Losses',
'A_UFC_Win_Percentage',
'B_UFC_Win_Percentage',
'Dif_UFC_Win_Percentage',
'A_Last5_Win_Percentage',
'B_Last5_Win_Percentage',
'Dif_Last5_Win_Percentage',
'A_Last3_Win_Percentage',
'B_Last3_Win_Percentage',
'Dif_Last3_Win_Percentage',
'A_Win_By_KO_Percentage',
'B_Win_By_KO_Percentage',
'Dif_Win_By_KO_Percentage',
'A_Loss_By_KO_Percentage',
'B_Loss_By_KO_Percentage',
'Dif_Loss_By_KO_Percentage',
'A_Win_By_Decision_Percentage',
'B_Win_By_Decision_Percentage',
'Dif_Win_By_Decision_Percentage',
'A_Loss_By_Decision_Percentage',
'B_Loss_By_Decision_Percentage',
'Dif_Loss_By_Decision_Percentage',
'A_UFC_Fight_Time_Seconds',
'B_UFC_Fight_Time_Seconds',
'Dif_UFC_Fight_Time_Seconds',
```

```
'A_UFC_Fight_Rounds',
'B_UFC_Fight_Rounds',
'A_topdown_Avg_Kd_per_round',
'A_topdown_Avg_Sig_strike_land_per_round',
'A_topdown_Avg_Sig_strike_att_per_round',
'A_topdown_Avg_Total_Strikes_land_per_round',
'A_topdown_Avg_Total_Strikes_att_per_round',
'A_topdown_Avg_Takedowns_land_per_round',
'A_topdown_Avg_Takedowns_att_per_round',
'A_topdown_Avg_Sub_Attempts_land_per_round',
'A_topdown_Avg_Sub_Attempts_att_per_round',
'A_topdown_Avg_Rev_per_round',
'A_topdown_Avg_Ctrl_time_min_per_round',
'A_topdown_Avg_Ctrl_time_sec_per_round',
'A_topdown_Avg_Ctrl_time_tot_per_round',
'A_topdown_Avg_Head_Strikes_land_per_round',
'A_topdown_Avg_Head_Strikes_att_per_round',
'A_topdown_Avg_Body_Strikes_land_per_round',
'A_topdown_Avg_Body_Strikes_att_per_round',
'A_topdown_Avg_Leg_Strikes_land_per_round',
'A_topdown_Avg_Leg_Strikes_att_per_round',
'A_topdown_Avg_Distance_Strikes_land_per_round',
'A_topdown_Avg_Distance_Strikes_att_per_round',
'A_topdown_Avg_Clinch_Strikes_land_per_round',
'A_topdown_Avg_Clinch_Strikes_att_per_round',
'A_topdown_Avg_Ground_Strikes_land_per_round',
'A_topdown_Avg_Ground_Strikes_att_per_round',
'B_topdown_Avg_Kd_per_round',
'B_topdown_Avg_Sig_strike_land_per_round',
'B_topdown_Avg_Sig_strike_att_per_round',
'B_topdown_Avg_Total_Strikes_land_per_round',
'B_topdown_Avg_Total_Strikes_att_per_round',
'B_topdown_Avg_Takedowns_land_per_round',
'B_topdown_Avg_Takedowns_att_per_round',
'B_topdown_Avg_Sub_Attempts_land_per_round',
'B_topdown_Avg_Sub_Attempts_att_per_round',
'B_topdown_Avg_Rev_per_round',
'B_topdown_Avg_Ctrl_time_min_per_round',
'B_topdown_Avg_Ctrl_time_sec_per_round',
'B_topdown_Avg_Ctrl_time_tot_per_round',
'B_topdown_Avg_Head_Strikes_land_per_round',
'B_topdown_Avg_Head_Strikes_att_per_round',
'B_topdown_Avg_Body_Strikes_land_per_round',
'B_topdown_Avg_Body_Strikes_att_per_round',
'B_topdown_Avg_Leg_Strikes_land_per_round',
'B_topdown_Avg_Leg_Strikes_att_per_round',
'B_topdown_Avg_Distance_Strikes_land_per_round',
'B_topdown_Avg_Distance_Strikes_att_per_round',
'B_topdown_Avg_Clinch_Strikes_land_per_round',
'B_topdown_Avg_Clinch_Strikes_att_per_round',
'B_topdown_Avg_Ground_Strikes_land_per_round',
'B_topdown_Avg_Ground_Strikes_att_per_round',
'A_Opp_Avg_Kd_per_round',
'A_Opp_Avg_Sig_strike_land_per_round',
'A_Opp_Avg_Sig_strike_att_per_round',
'A_Opp_Avg_Total_Strikes_land_per_round',
```

```
'A_Opp_Avg_Total_Strikes_att_per_round',
'A_Opp_Avg_Takedowns_land_per_round',
'A_Opp_Avg_Takedowns_att_per_round',
'A_Opp_Avg_Sub_Attempts_land_per_round',
'A_Opp_Avg_Sub_Attempts_att_per_round',
'A_Opp_Avg_Rev_per_round',
'A_Opp_Avg_Ctrl_time_min_per_round',
'A_Opp_Avg_Ctrl_time_sec_per_round',
'A_Opp_Avg_Ctrl_time_tot_per_round',
'A_Opp_Avg_Head_Strikes_land_per_round',
'A_Opp_Avg_Head_Strikes_att_per_round',
'A_Opp_Avg_Body_Strikes_land_per_round',
'A_Opp_Avg_Body_Strikes_att_per_round',
'A_Opp_Avg_Leg_Strikes_land_per_round',
'A_Opp_Avg_Leg_Strikes_att_per_round',
'A_Opp_Avg_Distance_Strikes_land_per_round',
'A_Opp_Avg_Distance_Strikes_att_per_round',
'A_Opp_Avg_Clinch_Strikes_land_per_round',
'A_Opp_Avg_Clinch_Strikes_att_per_round',
'A_Opp_Avg_Ground_Strikes_land_per_round',
'A_Opp_Avg_Ground_Strikes_att_per_round',
'B_Opp_Avg_Kd_per_round',
'B_Opp_Avg_Sig_strike_land_per_round',
'B_Opp_Avg_Sig_strike_att_per_round',
'B_Opp_Avg_Total_Strikes_land_per_round',
'B_Opp_Avg_Total_Strikes_att_per_round',
'B_Opp_Avg_Takedowns_land_per_round',
'B_Opp_Avg_Takedowns_att_per_round',
'B_Opp_Avg_Sub_Attempts_land_per_round',
'B_Opp_Avg_Sub_Attempts_att_per_round',
'B_Opp_Avg_Rev_per_round',
'B_Opp_Avg_Ctrl_time_min_per_round',
'B_Opp_Avg_Ctrl_time_sec_per_round',
'B_Opp_Avg_Ctrl_time_tot_per_round',
'B_Opp_Avg_Head_Strikes_land_per_round',
'B_Opp_Avg_Head_Strikes_att_per_round',
'B_Opp_Avg_Body_Strikes_land_per_round',
'B_Opp_Avg_Body_Strikes_att_per_round',
'B_Opp_Avg_Leg_Strikes_land_per_round',
'B_Opp_Avg_Leg_Strikes_att_per_round',
'B_Opp_Avg_Distance_Strikes_land_per_round',
'B_Opp_Avg_Distance_Strikes_att_per_round',
'B_Opp_Avg_Clinch_Strikes_land_per_round',
'B_Opp_Avg_Clinch_Strikes_att_per_round',
'B_Opp_Avg_Ground_Strikes_land_per_round',
'B_Opp_Avg_Ground_Strikes_att_per_round',
'Dif_topdown_Avg_Kd_per_round',
'Dif_topdown_Avg_Sig_strike_land_per_round',
'Dif_topdown_Avg_Sig_strike_att_per_round',
'Dif_topdown_Avg_Total_Strikes_land_per_round',
'Dif_topdown_Avg_Total_Strikes_att_per_round',
'Dif_topdown_Avg_Takedowns_land_per_round',
'Dif_topdown_Avg_Takedowns_att_per_round',
'Dif_topdown_Avg_Sub_Attempts_land_per_round',
'Dif_topdown_Avg_Sub_Attempts_att_per_round',
'Dif_topdown_Avg_Rev_per_round',
```

```
 'Dif_topdown_Avg_Ctrl_time_min_per_round',
 'Dif_topdown_Avg_Ctrl_time_sec_per_round',
 'Dif_topdown_Avg_Ctrl_time_tot_per_round',
 'Dif_topdown_Avg_Head_Strikes_land_per_round',
 'Dif_topdown_Avg_Head_Strikes_att_per_round',
 'Dif_topdown_Avg_Body_Strikes_land_per_round',
 'Dif_topdown_Avg_Body_Strikes_att_per_round',
 'Dif_topdown_Avg_Leg_Strikes_land_per_round',
 'Dif_topdown_Avg_Leg_Strikes_att_per_round',
 'Dif_topdown_Avg_Distance_Strikes_land_per_round',
 'Dif_topdown_Avg_Distance_Strikes_att_per_round',
 'Dif_topdown_Avg_Clinch_Strikes_land_per_round',
 'Dif_topdown_Avg_Clinch_Strikes_att_per_round',
 'Dif_topdown_Avg_Ground_Strikes_land_per_round',
 'Dif_topdown_Avg_Ground_Strikes_att_per_round',
 'Dif_Opp_Avg_Kd_per_round',
 'Dif_Opp_Avg_Sig_strike_land_per_round',
 'Dif_Opp_Avg_Sig_strike_att_per_round',
 'Dif_Opp_Avg_Total_Strikes_land_per_round',
 'Dif_Opp_Avg_Total_Strikes_att_per_round',
 'Dif_Opp_Avg_Takedowns_land_per_round',
 'Dif_Opp_Avg_Takedowns_att_per_round',
 'Dif_Opp_Avg_Sub_Attempts_land_per_round',
 'Dif_Opp_Avg_Sub_Attempts_att_per_round',
 'Dif_Opp_Avg_Rev_per_round',
 'Dif_Opp_Avg_Ctrl_time_min_per_round',
 'Dif_Opp_Avg_Ctrl_time_sec_per_round',
 'Dif_Opp_Avg_Ctrl_time_tot_per_round',
 'Dif_Opp_Avg_Head_Strikes_land_per_round',
 'Dif_Opp_Avg_Head_Strikes_att_per_round',
 'Dif_Opp_Avg_Body_Strikes_land_per_round',
 'Dif_Opp_Avg_Body_Strikes_att_per_round',
 'Dif_Opp_Avg_Leg_Strikes_land_per_round',
 'Dif_Opp_Avg_Leg_Strikes_att_per_round',
 'Dif_Opp_Avg_Distance_Strikes_land_per_round',
 'Dif_Opp_Avg_Distance_Strikes_att_per_round',
 'Dif_Opp_Avg_Clinch_Strikes_land_per_round',
 'Dif_Opp_Avg_Clinch_Strikes_att_per_round',
 'Dif_Opp_Avg_Ground_Strikes_land_per_round',
 'Dif_Opp_Avg_Ground_Strikes_att_per_round']
```

In [ ]:
```
numerical_columns.remove('favorite?')
categorical_columns.append('favorite?')
```

In [ ]:
```
col_list = X.columns.to_list()
```

Split

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)
```

In [ ]:
```
catcols = []

for col in categorical_columns:
    ind = col_list.index(col)
    catcols.append(ind)
```

```
catcols
```

Out[ ]:
```
[462, 471]
```

In [ ]:
```
cont_cols_index = [n for n in range(len(X_train.columns)) if n not in catcols]
```

In [ ]:
```
# To get the column names from onehotencoder
ohe = OneHotEncoder(sparse=False, handle_unknown='ignore')
checker = ohe.fit_transform(X_train[categorical_columns])
```

In [ ]:
```
feature_names_categorical = ohe.get_feature_names(categorical_columns)
```

Preprocessing

In [ ]:
```
# pipeline for categorical data
cat_preprocessing = make_pipeline(
    OneHotEncoder(handle_unknown="ignore", sparse=False),
)
# pipeline for numerical data
num_preprocessing = make_pipeline(StandardScaler())

# combine both pipeline using a columnTransformer
preprocessing = ColumnTransformer(
    [("num", num_preprocessing, cont_cols_index), ("cat", cat_preprocessing, catcol
)

preprocessing
```

Out[ ]:
```
ColumnTransformer(transformers=[('num',
                                 Pipeline(steps=[('standardscaler',
                                                  StandardScaler())]),
                                 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
                                  14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
                                  25, 26, 27, 28, 29, ...]),
                                ('cat',
                                 Pipeline(steps=[('onehotencoder',
                                                  OneHotEncoder(handle_unknown='ig
nore',
                                                                sparse=False))]),
                                 [462, 471])])
```

Function Additions

In [ ]:
```
# SOURCE: The origin of this confusion matrix code was found on medium, '
# from https://medium.com/@dtuk81/confusion-matrix-visualization-fc31e3f30fea
def make_confusion_matrix(cf,
                          group_names=None,
                          categories='auto',
                          count=True,
                          percent=True,
                          cbar=True,
                          xyticks=True,
```

```
                      xyplotlabels=True,
                      sum_stats=True,
                      figsize=None,
                      cmap='Blues',
                      title=None):

    # CODE TO GENERATE SUMMARY STATISTICS & TEXT FOR SUMMARY STATS
    if sum_stats:
        #Accuracy is sum of diagonal divided by total observations
        accuracy  = np.trace(cf) / float(np.sum(cf))

        #if it is a binary confusion matrix, show some more stats
        if len(cf)==2:
            #Metrics for Binary Confusion Matrices
            a = cf[0,0]
            b = cf[0,1]
            c = cf[1,0]
            d = cf[1,1]
            tn = ((a / (a+b))*100).round(2).astype(str) + '%'
            fp = ((b / (a+b))*100).round(2).astype(str) + '%'
            fn = ((c / (c+d))*100).round(2).astype(str) + '%'
            tp = ((d / (c+d))*100).round(2).astype(str) + '%'
            precision = cf[1,1] / sum(cf[:,1])
            recall    = cf[1,1] / sum(cf[1,:])
            f1_score  = 2*precision*recall / (precision + recall)
            rwf_score = (1+(2**2)) * ((precision * recall) / (((2**2) * precision)
            stats_text = "\n\nAccuracy={:0.3f}\nPrecision={:0.3f}\nRecall={:0.3f}\n
                accuracy,precision,recall,f1_score, rwf_score)
        else:
            stats_text = "\n\nAccuracy={:0.3f}".format(accuracy)
    else:
        stats_text = ""

    # CODE TO GENERATE TEXT INSIDE EACH SQUARE
    blanks = ['' for i in range(cf.size)]

    if group_names and len(group_names)==cf.size:
        group_labels = ["{}\n".format(value) for value in group_names]
    else:
        group_labels = blanks

    if count:
        group_counts = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        group_counts = blanks

    if percent:
        group_percentages =  [tn,fp,fn,tp]
        # old = group_percentages = ["{0:.2%}".format(value) for value in cf.flatte
    else:
        group_percentages = blanks

    box_labels = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in zip(group_labels,group_
    box_labels = np.asarray(box_labels).reshape(cf.shape[0],cf.shape[1])

    # SET FIGURE PARAMETERS ACCORDING TO OTHER ARGUMENTS
```

```
    if figsize==None:
        #Get default figure size if not set
        figsize = plt.rcParams.get('figure.figsize')

    if xyticks==False:
        #Do not show categories if xyticks is False
        categories=False


    # MAKE THE HEATMAP VISUALIZATION
    plt.figure(figsize=figsize)
    sns.heatmap(cf,annot=box_labels,fmt="",cmap=cmap,cbar=cbar,xticklabels=categori

    if xyplotlabels:
        plt.ylabel('True label', weight = 'bold')
        plt.xlabel('Predicted label' + stats_text, weight = 'bold')
    else:
        plt.xlabel(stats_text)

    if title:
        plt.title(title,size = 20, weight = 'bold')
```

In [ ]:
```
dfcols = ['Model', 'RWF Score', 'F1',  'Recall',  'Precision',  'Accuracy']
model_summary = pd.DataFrame(columns=dfcols)
model_summary
```

Out[ ]:

Model	RWF Score	F1	Recall	Precision	Accuracy

In [ ]:
```
# Define Result Saving Initial Function
def save_result_w_matrix(cf,cv_mean_accuracy, cv_std_accuracy, model_name):
            global model_summary
            accuracy  = np.trace(cf) / float(np.sum(cf))
            precision = cf[1,1] / sum(cf[:,1])
            recall    = cf[1,1] / sum(cf[1,:])
            f1_score  = 2*precision*recall / (precision + recall)
            f_beta = (1+(.5**2)) * ((precision * recall) / (((.5**2) * precision) +
            row = [(model_name, f_beta, f1_score, recall, precision, accuracy, cv_m
            res = pd.DataFrame(columns = dfcols, data = row)
            yeep = [model_summary, res]
            model_summary = pd.concat(yeep)
            model_summary = model_summary.sort_values('Accuracy', ascending = False
            model_summary = model_summary.drop_duplicates()
            return model_summary.round(3)
```

In [ ]:
```
def save_result(cv_mean_accuracy, cv_std_accuracy, model_name):
            global model_summary2
            row = [(model_name, cv_mean_accuracy, cv_std_accuracy)]
            res = pd.DataFrame(columns = dfcols2, data = row)
            yeep = [model_summary2, res]
            model_summary2 = pd.concat(yeep)
            model_summary2 = model_summary2.sort_values('Cv_Mean_Accuracy', ascendi
            model_summary2 = model_summary2.drop_duplicates()
            return model_summary2.round(3)
```

In [ ]:
```
# Function runs model, fits it, and saves the results
def run_model(model, model_name):
    model.fit(X_train, y_train)
    model_prediction = model.predict(X_test)
    cf_matrix = confusion_matrix(y_test, model_prediction)
    save_result(cf_matrix, model_name)
    cf = make_confusion_matrix(cf_matrix)
    return model_summary
```

In [ ]:
```
dfcols2 = ['Model', 'Cv_Mean_Accuracy', 'Cv_Std']
model_summary2 = pd.DataFrame(columns=dfcols2)
model_summary2
```

Out[ ]:

Model	Cv_Mean_Accuracy	Cv_Std

In [ ]:
```
def make_cf(model):
    #load pickle file
    with open(model, 'rb') as f:
        model = pickle.load(f)
    #get predictions
    y_pred = model.predict(X_test)
    cf = plot_confusion_matrix(model, y_test, y_pred)
    return cf
```

In [ ]:
```
# Function creates a pipeline, runs it, saves the result, and saves a pickle file

def create_fullpipe(preprocessing, model, model_name):
    fullpipe = Pipeline(steps=[('preprocess', preprocessing), ('model', model)])
    fullpipe.fit(X_train, y_train)
    # cross validation
    cv = cross_val_score(fullpipe, X_test, y_test, cv=3, scoring='accuracy')
    cv_mean = cv.mean()
    cv_std = cv.std()
    res = save_result(cv_mean, cv_std, model_name)
    # pickle model
    pickle.dump(fullpipe, open(f'models/{model_name}.pkl', 'wb'))
    return res
```

Run Vanilla Models

In [ ]:
```
create_fullpipe(preprocessing, LogisticRegression(), 'Logistic_Regression')
create_fullpipe(preprocessing, RandomForestClassifier(), 'Random_Forest')
create_fullpipe(preprocessing, DecisionTreeClassifier(), 'Decision_Tree')
create_fullpipe(preprocessing, BaggingClassifier(), 'Bagged_Trees')
create_fullpipe(preprocessing, ExtraTreesClassifier(), 'Extra_Trees')
create_fullpipe(preprocessing, KNeighborsClassifier(), 'K_Neighbors')
create_fullpipe(preprocessing, XGBClassifier(eval_metric = 'logloss'), 'XGBoost')
```

Out[ ]:

	Model	Cv_Mean_Accuracy	Cv_Std
0	Random_Forest	0.710	0.005
0	Extra_Trees	0.709	0.011
0	XGBoost	0.705	0.013
0	Logistic_Regression	0.691	0.013
0	Bagged_Trees	0.672	0.005
0	K_Neighbors	0.629	0.005
0	Decision_Tree	0.600	0.011

In [ ]:
```
# Function cross validates a model and saves the result and a pickle file
def gridsearched_model(model, model_name):
    cv = cross_val_score(model, X_test, y_test, cv=3, scoring='accuracy')
    cv_mean = cv.mean()
    cv_std = cv.std()
    res = save_result(cv_mean, cv_std, model_name)
    # pickle model
    pickle.dump(model, open(f'models/{model_name}.pkl', 'wb'))
    # make confusion matrix
    return res
```

In [ ]:
```
create_fullpipe(preprocessing, RandomForestClassifier(n_estimators=500), 'Random_Fo
create_fullpipe(preprocessing, RandomForestClassifier(n_estimators=1000), 'Random_F
```

Out[ ]:

	Model	Cv_Mean_Accuracy	Cv_Std
0	Random_Forest_500	0.714	0.012
0	Random_Forest_1000	0.713	0.016
0	Random_Forest	0.710	0.005
0	Extra_Trees	0.709	0.011
0	XGBoost	0.705	0.013
0	Logistic_Regression	0.691	0.013
0	Bagged_Trees	0.672	0.005
0	K_Neighbors	0.629	0.005
0	Decision_Tree	0.600	0.011

Estimators Best at 1000

Now max features

In [ ]:
```
create_fullpipe(preprocessing, RandomForestClassifier(n_estimators=1000, max_featur
create_fullpipe(preprocessing, RandomForestClassifier(n_estimators=1000, max_featur
```

Out[ ]:

	Model	Cv_Mean_Accuracy	Cv_Std
0	Random_Forest_mf_auto	0.718	0.010
0	Random_Forest_mf_sqrt	0.716	0.014
0	Random_Forest_500	0.714	0.012
0	Random_Forest_1000	0.713	0.016
0	Random_Forest	0.710	0.005
0	Extra_Trees	0.709	0.011
0	XGBoost	0.705	0.013
0	Logistic_Regression	0.691	0.013
0	Bagged_Trees	0.672	0.005
0	K_Neighbors	0.629	0.005
0	Decision_Tree	0.600	0.011

In [ ]:

```
# This function is an edited (to fit my purposes) version of a function found on me
def GridSearch_table_plot(grid_clf, param_name,
                          num_results=15,
                          negative=False,
                          graph=True,
                          display_all_params=True):

    from matplotlib      import pyplot as plt
    from IPython.display import display

    clf = grid_clf.best_estimator_
    clf_params = grid_clf.best_params_
    clf_score = grid_clf.best_score_
    clf_stdev = grid_clf.cv_results_['std_test_score'][grid_clf.best_index_]
    cv_results = grid_clf.cv_results_

    print("best parameters: {}".format(clf_params))
    print("best score:      {:0.5f} (+/-{:0.5f})".format(clf_score, clf_stdev))
    if display_all_params:
        import pprint
        pprint.pprint(clf.get_params())

    # pick out the best results
    scores_df = pd.DataFrame(cv_results).sort_values(by='rank_test_score')

    new_param = 'param_' + param_name
    best_row = scores_df.iloc[0, :]
    best_mean = best_row['mean_test_score']
    best_stdev = best_row['std_test_score']
    best_param = best_row[new_param]

    # display the top 'num_results' results
    display(pd.DataFrame(cv_results) \
            .sort_values(by='rank_test_score').head(num_results))
```

```
    # PLOT
    scores_df = scores_df.sort_values(by=new_param)

    means = scores_df['mean_test_score']
    stds = scores_df['std_test_score']
    params = scores_df[new_param]

    # plot
    if graph:
        plt.figure(figsize=(8, 8))
        try:
            plt.errorbar(params, means, yerr=stds)
            plt.axhline(y=best_mean + best_stdev, color='red')
            plt.axhline(y=best_mean - best_stdev, color='red')
            plt.plot(best_param, best_mean, 'or')
            plt.title(param_name + " vs Score\nBest Score {:0.5f}".format(clf_score
            plt.xlabel(param_name)
            plt.ylabel('Score')
            plt.show()
        except:
            plt.plot(best_param, best_mean, 'or')
            plt.title(param_name + " vs Score\nBest Score {:0.5f}".format(clf_score
            plt.xlabel(param_name)
            plt.ylabel('Score')
            plt.show()
```

Instread of doing a full gridsearch, which would take days with this dataset, I will do a gridsearch for each parameter, adding the best parameter to the next gridsearch. In testing, this was the most efficient way to find the best parameters.

In [ ]:
```
def random_forest_gridsearch(model):
    init_grid = {'model__n_estimators' : [50, 100, 150, 200, 350, 500, 750, 1000],
                 'model__max_depth' : [None],
                 'model__max_features' : ['auto'],
                 'model__criterion' : ['gini'],
                 'model__min_samples_split' : [2],
                 'model__min_samples_leaf' : [1],
                 'model__bootstrap' :  [True],
                 'model__max_leaf_nodes' : [None],
                 'model__min_impurity_decrease' : [0.0],
                 'model__ccp_alpha' : [0.0]}

    # grid search #1  - n_estimators
    gridsearch_1 = GridSearchCV(model, init_grid, cv = 4, scoring = 'accuracy', n_j
    gridsearch_1.fit(X_train, y_train)
    gridsearched_model(gridsearch_1.best_estimator_, 'Random_Forest_Gridsearched_1'
    bp = gridsearch_1.best_params_
    bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

    # get param values
    n_est = bp2[bp2['index'] == 'model__n_estimators']
    n_est = n_est[0].values[0]

    # if estimators or depth is min or max, keep pushing boundaries
```

```
    if n_est == 1000:
        # go higher!
        grid2 = {'model__n_estimators' : [500, 650, 800, 1000, 1500, 2000],
                    'model__max_depth' : [None],
                    'model__max_features' : ['auto'],
                    'model__criterion' : ['gini', 'entropy'],
                    'model__min_samples_split' : [2],
                    'model__min_samples_leaf' : [1],
                    'model__bootstrap' :  [True],
                    'model__max_leaf_nodes' : [None],
                    'model__min_impurity_decrease' : [0.0],
                    'model__ccp_alpha' : [0.0]}
        gridsearch_2 = GridSearchCV(model, grid2, cv = 4, scoring = 'accuracy', n_j
        gridsearch_2.fit(X_train, y_train)
        gridsearched_model(gridsearch_2.best_estimator_, 'Random_Forest_Gridsearche
        bp = gridsearch_2.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

    if n_est == 50:
        # go lower
        grid2 = {'model__n_estimators' : [20, 30, 40, 50, 60, 75, 100],
                    'model__max_depth' : [None],
                    'model__max_features' : ['auto'],
                    'model__criterion' : ['gini'],
                    'model__min_samples_split' : [2],
                    'model__min_samples_leaf' : [1],
                    'model__bootstrap' :  [True],
                    'model__max_leaf_nodes' : [None],
                    'model__min_impurity_decrease' : [0.0],
                    'model__ccp_alpha' : [0.0]}
        gridsearch_2 = GridSearchCV(model, grid2, cv = 4, scoring = 'accuracy', n_j
        gridsearch_2.fit(X_train, y_train)
        gridsearched_model(gridsearch_2.best_estimator_, 'Random_Forest_Gridsearche
        bp = gridsearch_2.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

    else:
        print(f'Best n_estimators: {n_est}')
        maxdepth_grid = {'model__n_estimators' : [n_est],
                    'model__max_depth' : [None, 2,3,4,5,6,7,8,9,10, 15, 20, 25, 30,
                    'model__max_features' : ['auto'],
                    'model__criterion' : ['gini'],
                    'model__min_samples_split' : [2],
                    'model__min_samples_leaf' : [1],
                    'model__bootstrap' :  [True],
                    'model__max_leaf_nodes' : [None],
                    'model__min_impurity_decrease' : [0.0],
                    'model__ccp_alpha' : [0.0]}

        ##### GRID SEARCH 2 -- MAX DEPTH #####
        gridsearch_2 = GridSearchCV(model, maxdepth_grid, cv = 4, scoring = 'accura
        gridsearch_2.fit(X_train, y_train)
        gridsearched_model(gridsearch_2.best_estimator_, 'Random_Forest_Gridsearche
        bp = gridsearch_2.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        max_depth = bp2[bp2['index'] == 'model__max_depth']
```

```
        max_depth = max_depth[0].values[0]
        print(f'Best max_depth: {max_depth}')

        ##### GRID SEARCH 3 -- MAX FEATURES #####
        max_features_grid = {'model__n_estimators' : [n_est],
                             'model__max_depth' : [max_depth],
                             'model__max_features' : ['auto', 'sqrt', 'log2', .4
                             'model__criterion' : ['gini'],
                             'model__min_samples_split' : [2],
                             'model__min_samples_leaf' : [1],
                             'model__bootstrap' :  [True],
                             'model__max_leaf_nodes' : [None],
                             'model__min_impurity_decrease' : [0.0],
                             'model__ccp_alpha' : [0.0]}
        gridsearch_3 = GridSearchCV(model, max_features_grid, cv = 4, scoring = 'ac

        gridsearch_3.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_3.best_estimator_, 'Random_Forest_Gridse
        bp = gridsearch_3.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        max_features = bp2[bp2['index'] == 'model__max_features']
        max_features = max_features[0].values[0]
        print(f'Best max_features: {max_features}')

        ##### GRID SEARCH 4 -- MIN SAMPLES SPLIT #####
        min_samples_split_grid = {'model__n_estimators' : [n_est],
                             'model__max_depth' : [max_depth],
                             'model__max_features' : [max_features],
                             'model__criterion' : ['gini'],
                             'model__min_samples_split' : [2, 3, 4, 5, 6, 7, 8,
                             'model__min_samples_leaf' : [1],
                             'model__bootstrap' :  [True],
                             'model__max_leaf_nodes' : [None],
                             'model__min_impurity_decrease' : [0.0],
                             'model__ccp_alpha' : [0.0]}

        gridsearch_4 = GridSearchCV(model, min_samples_split_grid, cv = 4, scoring
        gridsearch_4.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_4.best_estimator_, 'Random_Forest_Gridse
        bp = gridsearch_4.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        min_samples_split = bp2[bp2['index'] == 'model__min_samples_split']
        min_samples_split = min_samples_split[0].values[0]
        print(f'Best min_samples_split: {min_samples_split}')

        ##### GRID SEARCH 5 -- MIN SAMPLES LEAF #####
        min_samples_leaf_grid = {'model__n_estimators' : [n_est],
                             'model__max_depth' : [max_depth],
                             'model__max_features' : [max_features],
                             'model__criterion' : ['gini'],
                             'model__min_samples_split' : [min_samples_split],
                             'model__min_samples_leaf' : [1, 2, 3, 4, 5, 6, 7, 8
                             'model__bootstrap' :  [True],
                             'model__max_leaf_nodes' : [None],
                             'model__min_impurity_decrease' : [0.0],
                             'model__ccp_alpha' : [0.0]}
```

```
gridsearch_5 = GridSearchCV(model, min_samples_leaf_grid, cv = 4, scoring =
gridsearch_5.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_5.best_estimator_, 'Random_Forest_Gridse
bp = gridsearch_5.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
min_samples_leaf = bp2[bp2['index'] == 'model__min_samples_leaf']
min_samples_leaf = min_samples_leaf[0].values[0]
print(f'Best min_samples_leaf: {min_samples_leaf}')

##### GRID SEARCH 6 -- MAX LEAF NODES #####
max_leaf_nodes_grid = {'model__n_estimators' : [n_est],
                       'model__max_depth' : [max_depth],
                       'model__max_features' : [max_features],
                       'model__criterion' : ['gini'],
                       'model__min_samples_split' : [min_samples_split],
                       'model__min_samples_leaf' : [min_samples_leaf],
                       'model__bootstrap' :  [True],
                       'model__max_leaf_nodes' : [None, 2, 3, 4, 5, 6, 7,
                       'model__min_impurity_decrease' : [0.0],
                       'model__ccp_alpha' : [0.0]}
gridsearch_6 = GridSearchCV(model, max_leaf_nodes_grid, cv = 4, scoring = '
gridsearch_6.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_6.best_estimator_, 'Random_Forest_Gridse
bp = gridsearch_6.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
max_leaf_nodes = bp2[bp2['index'] == 'model__max_leaf_nodes']
max_leaf_nodes = max_leaf_nodes[0].values[0]
print(f'Best max_leaf_nodes: {max_leaf_nodes}')

##### GRID SEARCH 7 -- MIN IMPURITY DECREASE #####
min_impurity_decrease_grid = {'model__n_estimators' : [n_est],
                       'model__max_depth' : [max_depth],
                       'model__max_features' : [max_features],
                       'model__criterion' : ['gini'],
                       'model__min_samples_split' : [min_samples_split],
                       'model__min_samples_leaf' : [min_samples_leaf],
                       'model__bootstrap' :  [True],
                       'model__max_leaf_nodes' : [max_leaf_nodes],
                       'model__min_impurity_decrease' : [0.0, 0.1, 0.2, 0.
                       'model__ccp_alpha' : [0.0]}
gridsearch_7 = GridSearchCV(model, min_impurity_decrease_grid, cv = 4, scor
gridsearch_7.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_7.best_estimator_, 'Random_Forest_Gridse
bp = gridsearch_7.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
min_impurity_decrease = bp2[bp2['index'] == 'model__min_impurity_decrease']
min_impurity_decrease = min_impurity_decrease[0].values[0]
print(f'Best min_impurity_decrease: {min_impurity_decrease}')

##### GRID SEARCH 8 -- CCP ALPHA #####
ccp_alpha_grid = {'model__n_estimators' : [n_est],
                       'model__max_depth' : [max_depth],
                       'model__max_features' : [max_features],
                       'model__criterion' : ['gini'],
                       'model__min_samples_split' : [min_samples_split],
```

```
                                'model__min_samples_leaf' : [min_samples_leaf],
                                'model__bootstrap' :  [True],
                                'model__max_leaf_nodes' : [max_leaf_nodes],
                                'model__min_impurity_decrease' : [min_impurity_decr
                                'model__ccp_alpha' : [0.0, 0.1, 0.2, 0.3, 0.4, 0.5]

        gridsearch_8 = GridSearchCV(model, ccp_alpha_grid, cv = 4, scoring = 'accur
        gridsearch_8.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_8.best_estimator_, 'Random_Forest_Gridse
        bp = gridsearch_8.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        ccp_alpha = bp2[bp2['index'] == 'model__ccp_alpha']
        ccp_alpha = ccp_alpha[0].values[0]
        print(f'Best ccp_alpha: {ccp_alpha}')

        return gm, gridsearch_1.best_params_, gridsearch_2.best_params_, gridsearch
```

In [ ]:
```
random_forest = pickle.load(open('models/random_forest.pkl', 'rb'))
random_forest_gridsearch(random_forest)
```

```
Fitting 4 folds for each of 8 candidates, totalling 32 fits
Best n_estimators: 500
Fitting 4 folds for each of 26 candidates, totalling 104 fits
Best max_depth: 8
Fitting 4 folds for each of 10 candidates, totalling 40 fits
Best max_features: 0.8
Fitting 4 folds for each of 9 candidates, totalling 36 fits
Best min_samples_split: 2
Fitting 4 folds for each of 10 candidates, totalling 40 fits
Best min_samples_leaf: 4
Fitting 4 folds for each of 10 candidates, totalling 40 fits
Best max_leaf_nodes: None
Fitting 4 folds for each of 6 candidates, totalling 24 fits
Best min_impurity_decrease: 0.0
Fitting 4 folds for each of 6 candidates, totalling 24 fits
Best ccp_alpha: 0.0
```

Out[ ]: (

	Model	Cv_Mean_Accuracy	Cv_Std
0	Random_Forest_Gridsearched_2	0.719	0.009
0	Random_Forest_mf_auto	0.718	0.010
0	Random_Forest_mf_sqrt	0.716	0.014
0	Random_Forest_500	0.714	0.012
0	Random_Forest_1000	0.713	0.016
0	Random_Forest_Gridsearched_4	0.713	0.006
0	Random_Forest_Gridsearched_6	0.711	0.009
0	Random_Forest_Gridsearched_8	0.710	0.006
0	Random_Forest	0.710	0.005
0	Random_Forest_Gridsearched_1	0.710	0.006
0	Random_Forest_Gridsearched_7	0.709	0.008
0	Random_Forest_Gridsearched_5	0.709	0.005
0	Extra_Trees	0.709	0.011
0	Random_Forest_Gridsearched_3	0.706	0.015
0	XGBoost	0.705	0.013
0	Logistic_Regression	0.691	0.013
0	Bagged_Trees	0.672	0.005
0	K_Neighbors	0.629	0.005
0	Decision_Tree	0.600	0.011


```
{'model__bootstrap': True,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'gini',
 'model__max_depth': None,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 2,
 'model__n_estimators': 500},
{'model__bootstrap': True,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'gini',
 'model__max_depth': 8,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 2,
 'model__n_estimators': 500},
{'model__bootstrap': True,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'gini',
 'model__max_depth': 8,
 'model__max_features': 0.8,
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 2,
 'model__n_estimators': 500},
{'model__bootstrap': True,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'gini',
 'model__max_depth': 8,
 'model__max_features': 0.8,
 'model__max_leaf_nodes': None,
```

```
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 1,
  'model__min_samples_split': 2,
  'model__n_estimators': 500},
 {'model__bootstrap': True,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'gini',
  'model__max_depth': 8,
  'model__max_features': 0.8,
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 4,
  'model__min_samples_split': 2,
  'model__n_estimators': 500},
 {'model__bootstrap': True,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'gini',
  'model__max_depth': 8,
  'model__max_features': 0.8,
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 4,
  'model__min_samples_split': 2,
  'model__n_estimators': 500},
 {'model__bootstrap': True,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'gini',
  'model__max_depth': 8,
  'model__max_features': 0.8,
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 4,
  'model__min_samples_split': 2,
  'model__n_estimators': 500},
 {'model__bootstrap': True,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'gini',
  'model__max_depth': 8,
  'model__max_features': 0.8,
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 4,
  'model__min_samples_split': 2,
  'model__n_estimators': 500})
```

Model 1: XGBoost

In [ ]:
```
xg_boost = pickle.load(open('models/XGBoost.pkl', 'rb'))
keys = xg_boost.get_params().keys()
model_keys = [key for key in keys if 'model' in key]
xg_boost[1].get_params
```

Out[ ]:
```
<bound method XGBModel.get_params of XGBClassifier(base_score=0.5, booster='gbtre
e', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
              eval_metric='logloss', gamma=0, gpu_id=-1, importance_type=None,
              interaction_constraints='', learning_rate=0.300000012,
              max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
              monotone_constraints='()', n_estimators=100, n_jobs=32,
              num_parallel_tree=1, predictor='auto', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)>
```

In [ ]:
```
def xgboost_gridsearch(model):
        ##### GRID SEARCH 1 -- N ESTIMATORS #####
        n_est_grid = {'model__n_estimators' : [50, 100, 200, 300, 500, 600,  800,
                                'model__max_depth' : [None],
                                'model__learning_rate' : [0.1],
                                'model__min_child_weight' : [1],
                                'model__gamma' : [0],
                                'model__subsample' : [1],
                                'model__colsample_bytree' : [1],
                                'model__colsample_bylevel' : [1],
                                'model__colsample_bynode' : [1],
                                'model__reg_alpha' : [0],
                                'model__reg_lambda' : [1],
                                'model__scale_pos_weight' : [1],
                                'model__base_score' : [0.5],
                                'model__n_jobs' : [20],
                                'model__tree_method' : ['exact']}


        gridsearch_1 = GridSearchCV(model, n_est_grid, cv = 4, scoring = 'accuracy'
        gridsearch_1.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_1.best_estimator_, 'XGBoost_Gridsearched
        bp = gridsearch_1.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        n_est = bp2[bp2['index'] == 'model__n_estimators']
        n_est = n_est[0].values[0]
        print(f'Best n_estimators: {n_est}')

        ##### GRID SEARCH 2 -- MAX DEPTH #####
        max_depth_grid = {'model__n_estimators' : [n_est],
                                'model__max_depth' : [3, 4, 5, 6, 7, 8, 9, 10, 11,
                                'model__learning_rate' : [0.1],
                                'model__min_child_weight' : [1],
                                'model__gamma' : [0],
                                'model__subsample' : [1],
                                'model__colsample_bytree' : [1],
                                'model__colsample_bylevel' : [1],
                                'model__colsample_bynode' : [1],
                                'model__reg_alpha' : [0],
                                'model__reg_lambda' : [1],
                                'model__scale_pos_weight' : [1],
                                'model__base_score' : [0.5],
                                'model__n_jobs' : [20],
                                'model__tree_method' : ['exact']}
```

```
gridsearch_2 = GridSearchCV(model, max_depth_grid, cv = 4, scoring = 'accur
gridsearch_2.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_2.best_estimator_, 'XGBoost_Gridsearched
bp = gridsearch_2.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
max_depth = bp2[bp2['index'] == 'model__max_depth']
max_depth = max_depth[0].values[0]
print(f'Best max_depth: {max_depth}')

##### GRID SEARCH 3 -- LEARNING RATE #####
learning_rate_grid = {'model__n_estimators' : [n_est],
                      'model__max_depth' : [max_depth],
                      'model__learning_rate' : [0.01, 0.05, 0.1, 0.15, 0.
                      'model__min_child_weight' : [1],
                      'model__gamma' : [0],
                      'model__subsample' : [1],
                      'model__colsample_bytree' : [1],
                      'model__colsample_bylevel' : [1],
                      'model__colsample_bynode' : [1],
                      'model__reg_alpha' : [0],
                      'model__reg_lambda' : [1],
                      'model__scale_pos_weight' : [1],
                      'model__base_score' : [0.5],
                      'model__n_jobs' : [20],
                      'model__tree_method' : ['exact']}

gridsearch_3 = GridSearchCV(model, learning_rate_grid, cv = 4, scoring = 'a
gridsearch_3.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_3.best_estimator_, 'XGBoost_Gridsearched
bp = gridsearch_3.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
learning_rate = bp2[bp2['index'] == 'model__learning_rate']
learning_rate = learning_rate[0].values[0]
print(f'Best learning_rate: {learning_rate}')

##### GRID SEARCH 4 -- MIN CHILD WEIGHT #####
min_child_weight_grid = {'model__n_estimators' : [n_est],
                      'model__max_depth' : [max_depth],
                      'model__learning_rate' : [learning_rate],
                      'model__min_child_weight' : [1, 2, 3, 4, 5, 6, 7, 8
                      'model__gamma' : [0],
                      'model__subsample' : [1],
                      'model__colsample_bytree' : [1],
                      'model__colsample_bylevel' : [1],
                      'model__colsample_bynode' : [1],
                      'model__reg_alpha' : [0],
                      'model__reg_lambda' : [1],
                      'model__scale_pos_weight' : [1],
                      'model__base_score' : [0.5],
                      'model__n_jobs' : [20],
                      'model__tree_method' : ['exact']}

gridsearch_4 = GridSearchCV(model, min_child_weight_grid, cv = 4, scoring =
gridsearch_4.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_4.best_estimator_, 'XGBoost_Gridsearched
```

```
        bp = gridsearch_4.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        min_child_weight = bp2[bp2['index'] == 'model__min_child_weight']
        min_child_weight = min_child_weight[0].values[0]
        print(f'Best min_child_weight: {min_child_weight}')

        ##### GRID SEARCH 5 -- GAMMA #####
        gamma_grid = {'model__n_estimators' : [n_est],
                      'model__max_depth' : [max_depth],
                      'model__learning_rate' : [learning_rate],
                      'model__min_child_weight' : [min_child_weight],
                      'model__gamma' : [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6,
                      'model__subsample' : [1],
                      'model__colsample_bytree' : [1],
                      'model__colsample_bylevel' : [1],
                      'model__colsample_bynode' : [1],
                      'model__reg_alpha' : [0],
                      'model__reg_lambda' : [1],
                      'model__scale_pos_weight' : [1],
                      'model__base_score' : [0.5],
                      'model__n_jobs' : [20],
                      'model__tree_method' : ['exact']}

        gridsearch_5 = GridSearchCV(model, gamma_grid, cv = 4, scoring = 'accuracy'
        gridsearch_5.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_5.best_estimator_, 'XGBoost_Gridsearched
        bp = gridsearch_5.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        gamma = bp2[bp2['index'] == 'model__gamma']
        gamma = gamma[0].values[0]
        print(f'Best gamma: {gamma}')

        ##### GRID SEARCH 6 -- SUBSAMPLE #####
        subsample_grid = {'model__n_estimators' : [n_est],
                      'model__max_depth' : [max_depth],
                      'model__learning_rate' : [learning_rate],
                      'model__min_child_weight' : [min_child_weight],
                      'model__gamma' : [gamma],
                      'model__subsample' : [0.5, 0.6, 0.7, 0.8, 0.9, 1],
                      'model__colsample_bytree' : [1],
                      'model__colsample_bylevel' : [1],
                      'model__colsample_bynode' : [1],
                      'model__reg_alpha' : [0],
                      'model__reg_lambda' : [1],
                      'model__scale_pos_weight' : [1],
                      'model__base_score' : [0.5],
                      'model__n_jobs' : [20],
                      'model__tree_method' : ['exact']}

        gridsearch_6 = GridSearchCV(model, subsample_grid, cv = 4, scoring = 'accur
        gridsearch_6.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_6.best_estimator_, 'XGBoost_Gridsearched
        bp = gridsearch_6.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
        subsample = bp2[bp2['index'] == 'model__subsample']
        subsample = subsample[0].values[0]
```

```
print(f'Best subsample: {subsample}')

##### GRID SEARCH 7 -- COLSAMPLE BYTREE #####
colsample_bytree_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
                         'model__gamma' : [gamma],
                         'model__subsample' : [subsample],
                         'model__colsample_bytree' : [0.5, 0.6, 0.7, 0.8, 0.
                         'model__colsample_bylevel' : [1],
                         'model__colsample_bynode' : [1],
                         'model__reg_alpha' : [0],
                         'model__reg_lambda' : [1],
                         'model__scale_pos_weight' : [1],
                         'model__base_score' : [0.5],
                         'model__n_jobs' : [20],
                         'model__tree_method' : ['exact']}

gridsearch_7 = GridSearchCV(model, colsample_bytree_grid, cv = 4, scoring =
gridsearch_7.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_7.best_estimator_, 'XGBoost_Gridsearched
bp = gridsearch_7.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
colsample_bytree = bp2[bp2['index'] == 'model__colsample_bytree']
colsample_bytree = colsample_bytree[0].values[0]
print(f'Best colsample_bytree: {colsample_bytree}')

##### GRID SEARCH 8 -- COLSAMPLE BYLEVEL #####
colsample_bylevel_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
                         'model__gamma' : [gamma],
                         'model__subsample' : [subsample],
                         'model__colsample_bytree' : [colsample_bytree],
                         'model__colsample_bylevel' : [0.5, 0.6, 0.7, 0.8, 0
                         'model__colsample_bynode' : [1],
                         'model__reg_alpha' : [0],
                         'model__reg_lambda' : [1],
                         'model__scale_pos_weight' : [1],
                         'model__base_score' : [0.5],
                         'model__n_jobs' : [20],
                         'model__tree_method' : ['exact']}

gridsearch_8 = GridSearchCV(model, colsample_bylevel_grid, cv = 4, scoring
gridsearch_8.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_8.best_estimator_, 'XGBoost_Gridsearched
bp = gridsearch_8.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
colsample_bylevel = bp2[bp2['index'] == 'model__colsample_bylevel']
colsample_bylevel = colsample_bylevel[0].values[0]

print(f'Best colsample_bylevel: {colsample_bylevel}')

##### GRID SEARCH 9 -- COLSAMPLE BYNODE #####
```

```
colsample_bynode_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
                         'model__gamma' : [gamma],
                         'model__subsample' : [subsample],
                         'model__colsample_bytree' : [colsample_bytree],
                         'model__colsample_bylevel' : [colsample_bylevel],
                         'model__colsample_bynode' : [0.5, 0.6, 0.7, 0.8, 0.
                         'model__reg_alpha' : [0],
                         'model__reg_lambda' : [1],
                         'model__scale_pos_weight' : [1],
                         'model__base_score' : [0.5],
                         'model__n_jobs' : [20],
                         'model__tree_method' : ['exact']}

gridsearch_9 = GridSearchCV(model, colsample_bynode_grid, cv = 4, scoring =
gridsearch_9.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_9.best_estimator_, 'XGBoost_Gridsearched
bp = gridsearch_9.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
colsample_bynode = bp2[bp2['index'] == 'model__colsample_bynode']
colsample_bynode = colsample_bynode[0].values[0]
print(f'Best colsample_bynode: {colsample_bynode}')

##### GRID SEARCH 10 -- REG ALPHA #####
reg_alpha_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
                         'model__gamma' : [gamma],
                         'model__subsample' : [subsample],
                         'model__colsample_bytree' : [colsample_bytree],
                         'model__colsample_bylevel' : [colsample_bylevel],
                         'model__colsample_bynode' : [colsample_bynode],
                         'model__reg_alpha' : [0, 0.001, 0.005, 0.01, 0.05,
                         'model__reg_lambda' : [1],
                         'model__scale_pos_weight' : [1],
                         'model__base_score' : [0.5],
                         'model__n_jobs' : [20],
                         'model__tree_method' : ['exact']}

gridsearch_10 = GridSearchCV(model, reg_alpha_grid, cv = 4, scoring = 'accu
gridsearch_10.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_10.best_estimator_, 'XGBoost_Gridsearche
bp = gridsearch_10.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()
reg_alpha = bp2[bp2['index'] == 'model__reg_alpha']
reg_alpha = reg_alpha[0].values[0]
print(f'Best reg_alpha: {reg_alpha}')

##### GRID SEARCH 11 -- REG LAMBDA #####
reg_lambda_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
```

```
                            'model__gamma' : [gamma],
                            'model__subsample' : [subsample],
                            'model__colsample_bytree' : [colsample_bytree],
                            'model__colsample_bylevel' : [colsample_bylevel],
                            'model__colsample_bynode' : [colsample_bynode],
                            'model__reg_alpha' : [reg_alpha],
                            'model__reg_lambda' : [0, 0.001, 0.005, 0.01, 0.05,
                            'model__scale_pos_weight' : [1],
                            'model__base_score' : [0.5],
                            'model__n_jobs' : [20],
                            'model__tree_method' : ['exact']}


gridsearch_11 = GridSearchCV(model, reg_lambda_grid, cv = 4, scoring = 'acc
gridsearch_11.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_11.best_estimator_, 'XGBoost_Gridsearche
bp = gridsearch_11.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

reg_lambda = bp2[bp2['index'] == 'model__reg_lambda']
reg_lambda = reg_lambda[0].values[0]
print(f'Best reg_lambda: {reg_lambda}')

##### GRID SEARCH 12 -- SCALE POS WEIGHT #####
scale_pos_weight_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
                         'model__gamma' : [gamma],
                         'model__subsample' : [subsample],
                         'model__colsample_bytree' : [colsample_bytree],
                         'model__colsample_bylevel' : [colsample_bylevel],
                         'model__colsample_bynode' : [colsample_bynode],
                         'model__reg_alpha' : [reg_alpha],
                         'model__reg_lambda' : [reg_lambda],
                         'model__scale_pos_weight' : [1, 2, 3, 4, 5, 6, 7, 8
                         'model__base_score' : [0.5],
                         'model__n_jobs' : [20],
                         'model__tree_method' : ['exact']}

gridsearch_12 = GridSearchCV(model, scale_pos_weight_grid, cv = 4, scoring
gridsearch_12.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_12.best_estimator_, 'XGBoost_Gridsearche
bp = gridsearch_12.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

scale_pos_weight = bp2[bp2['index'] == 'model__scale_pos_weight']
scale_pos_weight = scale_pos_weight[0].values[0]

print(f'Best scale_pos_weight: {scale_pos_weight}')

##### GRID SEARCH 13 -- BASE SCORE #####
base_score_grid = {'model__n_estimators' : [n_est],
                         'model__max_depth' : [max_depth],
                         'model__learning_rate' : [learning_rate],
                         'model__min_child_weight' : [min_child_weight],
```

```
                                'model__gamma' : [gamma],
                                'model__subsample' : [subsample],
                                'model__colsample_bytree' : [colsample_bytree],
                                'model__colsample_bylevel' : [colsample_bylevel],
                                'model__colsample_bynode' : [colsample_bynode],
                                'model__reg_alpha' : [reg_alpha],
                                'model__reg_lambda' : [reg_lambda],
                                'model__scale_pos_weight' : [scale_pos_weight],
                                'model__base_score' : [0.1, 0.2, 0.3, 0.4, 0.5, 0.6
                                'model__n_jobs' : [20],
                                'model__tree_method' : ['exact']}

        gridsearch_13 = GridSearchCV(model, base_score_grid, cv = 4, scoring = 'acc
        gridsearch_13.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_13.best_estimator_, 'XGBoost_Gridsearche
        bp = gridsearch_13.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

        base_score = bp2[bp2['index'] == 'model__base_score']
        base_score = base_score[0].values[0]

        print(f'Best base_score: {base_score}')

        ##### GRID SEARCH 14 -- MAX DELTA STEP #####
        max_delta_step_grid = {'model__n_estimators' : [n_est],
                                'model__max_depth' : [max_depth],
                                'model__learning_rate' : [learning_rate],
                                'model__min_child_weight' : [min_child_weight],
                                'model__gamma' : [gamma],
                                'model__subsample' : [subsample],
                                'model__colsample_bytree' : [colsample_bytree],
                                'model__colsample_bylevel' : [colsample_bylevel],
                                'model__colsample_bynode' : [colsample_bynode],
                                'model__reg_alpha' : [reg_alpha],
                                'model__reg_lambda' : [reg_lambda],
                                'model__scale_pos_weight' : [scale_pos_weight],
                                'model__base_score' : [base_score],
                                'model__n_jobs' : [20],
                                'model__tree_method' : ['exact']}

        gridsearch_14 = GridSearchCV(model, max_delta_step_grid, cv = 4, scoring =
        gridsearch_14.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_14.best_estimator_, 'XGBoost_Gridsearche

        ## RETURN

        return gm, gridsearch_1.best_params_, gridsearch_2.best_params_, gridsearch
```

In [ ]:
```
xgboost_gridsearch(xg_boost)
```

```
Fitting 4 folds for each of 8 candidates, totalling 32 fits
Best n_estimators: 100
Fitting 4 folds for each of 14 candidates, totalling 56 fits
Best max_depth: 5
Fitting 4 folds for each of 11 candidates, totalling 44 fits
Best learning_rate: 0.1
Fitting 4 folds for each of 10 candidates, totalling 40 fits
Best min_child_weight: 5
Fitting 4 folds for each of 11 candidates, totalling 44 fits
Best gamma: 0.7
Fitting 4 folds for each of 6 candidates, totalling 24 fits
Best subsample: 1
Fitting 4 folds for each of 6 candidates, totalling 24 fits
Best colsample_bytree: 0.9
Fitting 4 folds for each of 6 candidates, totalling 24 fits
Best colsample_bylevel: 1
Fitting 4 folds for each of 6 candidates, totalling 24 fits
Best colsample_bynode: 1
Fitting 4 folds for each of 8 candidates, totalling 32 fits
Best reg_alpha: 0.001
Fitting 4 folds for each of 8 candidates, totalling 32 fits
Best reg_lambda: 1
Fitting 4 folds for each of 10 candidates, totalling 40 fits
Best scale_pos_weight: 1
Fitting 4 folds for each of 9 candidates, totalling 36 fits
Best base_score: 0.5
Fitting 4 folds for each of 1 candidates, totalling 4 fits
```

Out[ ]: (

	Model	Cv_Mean_Accuracy	Cv_Std
0	Random_Forest_Gridsearched_2	0.719	0.009
0	Random_Forest_mf_auto	0.718	0.010
0	Random_Forest_mf_sqrt	0.716	0.014
0	Random_Forest_500	0.714	0.012
0	Random_Forest_1000	0.713	0.016
0	Random_Forest_Gridsearched_4	0.713	0.006
0	Random_Forest_Gridsearched_6	0.711	0.009
0	Random_Forest_Gridsearched_8	0.710	0.006
0	XGBoost_Gridsearched_1	0.710	0.014
0	Random_Forest	0.710	0.005
0	Random_Forest_Gridsearched_1	0.710	0.006
0	Random_Forest_Gridsearched_7	0.709	0.008
0	Random_Forest_Gridsearched_5	0.709	0.005
0	Extra_Trees	0.709	0.011
0	Random_Forest_Gridsearched_3	0.706	0.015
0	XGBoost_Gridsearched_11	0.705	0.015
0	XGBoost_Gridsearched_13	0.705	0.015
0	XGBoost_Gridsearched_10	0.705	0.015
0	XGBoost_Gridsearched_12	0.705	0.015
0	XGBoost_Gridsearched_14	0.705	0.015
0	XGBoost	0.705	0.013
0	XGBoost_Gridsearched_5	0.702	0.011
0	XGBoost_Gridsearched_6	0.702	0.011
0	XGBoost_Gridsearched_4	0.701	0.013
0	XGBoost_Gridsearched_7	0.701	0.016
0	XGBoost_Gridsearched_8	0.701	0.016
0	XGBoost_Gridsearched_9	0.701	0.016
0	XGBoost_Gridsearched_2	0.693	0.018
0	XGBoost_Gridsearched_3	0.693	0.018
0	Logistic_Regression	0.691	0.013
0	Bagged_Trees	0.672	0.005
0	K_Neighbors	0.629	0.005
0	Decision_Tree	0.600	0.011,


```
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
 'model__colsample_bytree': 1,
 'model__gamma': 0,
 'model__learning_rate': 0.1,
 'model__max_depth': None,
 'model__min_child_weight': 1,
 'model__n_estimators': 100,
 'model__n_jobs': 20,
 'model__reg_alpha': 0,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'},
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
 'model__colsample_bytree': 1,
 'model__gamma': 0,
 'model__learning_rate': 0.1,
 'model__max_depth': 5,
```

```
 'model__min_child_weight': 1,
 'model__n_estimators': 100,
 'model__n_jobs': 20,
 'model__reg_alpha': 0,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'},
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
 'model__colsample_bytree': 1,
 'model__gamma': 0,
 'model__learning_rate': 0.1,
 'model__max_depth': 5,
 'model__min_child_weight': 1,
 'model__n_estimators': 100,
 'model__n_jobs': 20,
 'model__reg_alpha': 0,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'},
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
 'model__colsample_bytree': 1,
 'model__gamma': 0,
 'model__learning_rate': 0.1,
 'model__max_depth': 5,
 'model__min_child_weight': 5,
 'model__n_estimators': 100,
 'model__n_jobs': 20,
 'model__reg_alpha': 0,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'},
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
 'model__colsample_bytree': 1,
 'model__gamma': 0.7,
 'model__learning_rate': 0.1,
 'model__max_depth': 5,
 'model__min_child_weight': 5,
 'model__n_estimators': 100,
 'model__n_jobs': 20,
 'model__reg_alpha': 0,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'},
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
```

```
  'model__colsample_bytree': 1,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
```

```
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0.001,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0.001,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
  'model__reg_alpha': 0.001,
  'model__reg_lambda': 1,
  'model__scale_pos_weight': 1,
  'model__subsample': 1,
  'model__tree_method': 'exact'},
 {'model__base_score': 0.5,
  'model__colsample_bylevel': 1,
  'model__colsample_bynode': 1,
  'model__colsample_bytree': 0.9,
  'model__gamma': 0.7,
  'model__learning_rate': 0.1,
  'model__max_depth': 5,
  'model__min_child_weight': 5,
  'model__n_estimators': 100,
  'model__n_jobs': 20,
```

```
 'model__reg_alpha': 0.001,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'},
{'model__base_score': 0.5,
 'model__colsample_bylevel': 1,
 'model__colsample_bynode': 1,
 'model__colsample_bytree': 0.9,
 'model__gamma': 0.7,
 'model__learning_rate': 0.1,
 'model__max_depth': 5,
 'model__min_child_weight': 5,
 'model__n_estimators': 100,
 'model__n_jobs': 20,
 'model__reg_alpha': 0.001,
 'model__reg_lambda': 1,
 'model__scale_pos_weight': 1,
 'model__subsample': 1,
 'model__tree_method': 'exact'})
```

Extra Trees

In [ ]:
```
extra_trees = pickle.load(open('models/Extra_Trees.pkl', 'rb'))
```

In [ ]:
```
keys = extra_trees.get_params().keys()
model_keys = [key for key in keys if 'model' in key]
model_keys
```

Out[ ]:
```
['model',
 'model__bootstrap',
 'model__ccp_alpha',
 'model__class_weight',
 'model__criterion',
 'model__max_depth',
 'model__max_features',
 'model__max_leaf_nodes',
 'model__max_samples',
 'model__min_impurity_decrease',
 'model__min_samples_leaf',
 'model__min_samples_split',
 'model__min_weight_fraction_leaf',
 'model__n_estimators',
 'model__n_jobs',
 'model__oob_score',
 'model__random_state',
 'model__verbose',
 'model__warm_start']
```

In [ ]:
```
def extra_trees_gridsearch(model):
        """
        Gridsearches the Extra Trees model
        """
        # GRID SEARCH 1 -- N ESTIMATORS
        n_estimators_grid = {'model__n_estimators' : [100, 200, 300, 400, 500, 600,
```

```
                        'model__criterion' : ['gini'],
                        'model__max_depth' : [None,],
                        'model__min_samples_split' : [2 ],
                        'model__min_samples_leaf' : [1, ],
                        'model__min_weight_fraction_leaf' : [0.0],
                        'model__max_features' : ['auto'],
                        'model__max_leaf_nodes' : [None],
                        'model__min_impurity_decrease' : [0.0],
                        'model__bootstrap' : [False],
                        'model__oob_score' : [False],
                        'model__n_jobs' : [20],
                        'model__ccp_alpha' : [0.0]}

gridsearch_1 = GridSearchCV(model, n_estimators_grid, cv = 4, scoring = 'ac
gridsearch_1.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_1.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_1.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

# save the best n_estimators
n_est = bp2[bp2['index'] == 'model__n_estimators']
n_est = n_est[0].values[0]

print(f'Best n_estimators: {n_est}')

# GRID SEARCH 2 -- CRITERION
criterion_grid = {'model__n_estimators' : [n_est],
                  'model__criterion' : ['gini', 'entropy'],
                  'model__max_depth' : [None],
                  'model__min_samples_split' : [2 ],
                  'model__min_samples_leaf' : [1 ],
                  'model__min_weight_fraction_leaf' : [0.0],
                  'model__max_features' : ['auto'],
                  'model__max_leaf_nodes' : [None],
                  'model__min_impurity_decrease' : [0.0],
                  'model__bootstrap' : [False],
                  'model__oob_score' : [False],
                  'model__n_jobs' : [20],
                  'model__ccp_alpha' : [0.0]}
gridsearch_2 = GridSearchCV(model, criterion_grid, cv = 4, scoring = 'accur
gridsearch_2.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_2.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_2.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

criterion = bp2[bp2['index'] == 'model__criterion']
criterion = criterion[0].values[0]

print(f'Best criterion: {criterion}')

# GRID SEARCH 3 -- MAX DEPTH
max_depth_grid = {'model__n_estimators' : [n_est],
                  'model__criterion' : [criterion],
                  'model__max_depth' : [None, 1, 2, 3, 4, 5, 6, 7, 8, 9,
                  'model__min_samples_split' : [2 ],
                  'model__min_samples_leaf' : [1 ],
```

```
                        'model__min_weight_fraction_leaf' : [0.0],
                        'model__max_features' : ['auto'],
                        'model__max_leaf_nodes' : [None],
                        'model__min_impurity_decrease' : [0.0],
                        'model__bootstrap' : [False],
                        'model__oob_score' : [False],
                        'model__n_jobs' : [20],
                        'model__ccp_alpha' : [0.0]}
gridsearch_3 = GridSearchCV(model, max_depth_grid, cv = 4, scoring = 'accur
gridsearch_3.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_3.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_3.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

max_depth = bp2[bp2['index'] == 'model__max_depth']
max_depth = max_depth[0].values[0]

print(f'Best max_depth: {max_depth}')

# GRID SEARCH 4 -- MIN SAMPLES SPLIT
min_samples_split_grid = {'model__n_estimators' : [n_est],
                        'model__criterion' : [criterion],
                        'model__max_depth' : [max_depth],
                        'model__min_samples_split' : [2,3,4,5,6,7,8,9,10],
                        'model__min_samples_leaf' : [1 ],
                        'model__min_weight_fraction_leaf' : [0.0],
                        'model__max_features' : ['auto'],
                        'model__max_leaf_nodes' : [None],
                        'model__min_impurity_decrease' : [0.0],
                        'model__bootstrap' : [False],
                        'model__oob_score' : [False],
                        'model__n_jobs' : [20],
                        'model__ccp_alpha' : [0.0]}
gridsearch_4 = GridSearchCV(model, min_samples_split_grid, cv = 4, scoring
gridsearch_4.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_4.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_4.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

min_samples_split = bp2[bp2['index'] == 'model__min_samples_split']
min_samples_split = min_samples_split[0].values[0]

print(f'Best min_samples_split: {min_samples_split}')

# GRID SEARCH 5 -- MIN SAMPLES LEAF
min_samples_leaf_grid = {'model__n_estimators' : [n_est],
                        'model__criterion' : [criterion],
                        'model__max_depth' : [max_depth],
                        'model__min_samples_split' : [min_samples_split],
                        'model__min_samples_leaf' : [1,2,3,4,5,6,7,8,9,10],
                        'model__min_weight_fraction_leaf' : [0.0],
                        'model__max_features' : ['auto'],
                        'model__max_leaf_nodes' : [None],
                        'model__min_impurity_decrease' : [0.0],
                        'model__bootstrap' : [False],
                        'model__oob_score' : [False],
```

```
                    'model__n_jobs' : [20],
                    'model__ccp_alpha' : [0.0]}

gridsearch_5 = GridSearchCV(model, min_samples_leaf_grid, cv = 4, scoring =
gridsearch_5.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_5.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_5.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

min_samples_leaf = bp2[bp2['index'] == 'model__min_samples_leaf']
min_samples_leaf = min_samples_leaf[0].values[0]

print(f'Best min_samples_leaf: {min_samples_leaf}')

# GRID SEARCH 6 -- MIN WEIGHT FRACTION LEAF
min_weight_fraction_leaf_grid = {'model__n_estimators' : [n_est],
                    'model__criterion' : [criterion],
                    'model__max_depth' : [max_depth],
                    'model__min_samples_split' : [min_samples_split],
                    'model__min_samples_leaf' : [min_samples_leaf],
                    'model__min_weight_fraction_leaf' : [0.0, 0.1, 0.2, 0.3
                    'model__max_features' : ['auto'],
                    'model__max_leaf_nodes' : [None],
                    'model__min_impurity_decrease' : [0.0],
                    'model__bootstrap' : [False],
                    'model__oob_score' : [False],
                    'model__n_jobs' : [20],
                    'model__ccp_alpha' : [0.0]}

gridsearch_6 = GridSearchCV(model, min_weight_fraction_leaf_grid, cv = 4, s
gridsearch_6.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_6.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_6.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

min_weight_fraction_leaf = bp2[bp2['index'] == 'model__min_weight_fraction_
min_weight_fraction_leaf = min_weight_fraction_leaf[0].values[0]

print(f'Best min_weight_fraction_leaf: {min_weight_fraction_leaf}')

# GRID SEARCH 7 -- MAX FEATURES
max_features_grid = {'model__n_estimators' : [n_est],
                    'model__criterion' : [criterion],
                    'model__max_depth' : [max_depth],
                    'model__min_samples_split' : [min_samples_split],
                    'model__min_samples_leaf' : [min_samples_leaf],
                    'model__min_weight_fraction_leaf' : [min_weight_fractio
                    'model__max_features' : ['auto', 'sqrt', 'log2'],
                    'model__max_leaf_nodes' : [None],
                    'model__min_impurity_decrease' : [0.0],
                    'model__bootstrap' : [False],
                    'model__oob_score' : [False],
                    'model__n_jobs' : [20],
                    'model__ccp_alpha' : [0.0]}
gridsearch_7 = GridSearchCV(model, max_features_grid, cv = 4, scoring = 'ac
gridsearch_7.fit(X_train, y_train)
```

```
gm = gridsearched_model(gridsearch_7.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_7.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

max_features = bp2[bp2['index'] == 'model__max_features']
max_features = max_features[0].values[0]

print(f'Best max_features: {max_features}')

# GRID SEARCH 8 -- MAX LEAF NODES
max_leaf_nodes_grid = {'model__n_estimators' : [n_est],
                       'model__criterion' : [criterion],
                       'model__max_depth' : [max_depth],
                       'model__min_samples_split' : [min_samples_split],
                       'model__min_samples_leaf' : [min_samples_leaf],
                       'model__min_weight_fraction_leaf' : [min_weight_fractio
                       'model__max_features' : [max_features],
                       'model__max_leaf_nodes' : [None, 2, 4, 6, 8, 10, 12, 14
                       'model__min_impurity_decrease' : [0.0],
                       'model__bootstrap' : [False],
                       'model__oob_score' : [False],
                       'model__n_jobs' : [20],
                       'model__ccp_alpha' : [0.0]}
gridsearch_8 = GridSearchCV(model, max_leaf_nodes_grid, cv = 4, scoring = '
gridsearch_8.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_8.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_8.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

max_leaf_nodes = bp2[bp2['index'] == 'model__max_leaf_nodes']
max_leaf_nodes = max_leaf_nodes[0].values[0]

print(f'Best max_leaf_nodes: {max_leaf_nodes}')

# GRID SEARCH 9 -- MIN IMPURITY DECREASE
min_impurity_decrease_grid = {'model__n_estimators' : [n_est],
                       'model__criterion' : [criterion],
                       'model__max_depth' : [max_depth],
                       'model__min_samples_split' : [min_samples_split],
                       'model__min_samples_leaf' : [min_samples_leaf],
                       'model__min_weight_fraction_leaf' : [min_weight_fractio
                       'model__max_features' : [max_features],
                       'model__max_leaf_nodes' : [max_leaf_nodes],
                       'model__min_impurity_decrease' : [0.0, 0.1, 0.2, 0.3, 0
                       'model__bootstrap' : [False],
                       'model__oob_score' : [False],
                       'model__n_jobs' : [20],
                       'model__ccp_alpha' : [0.0]}

gridsearch_9 = GridSearchCV(model, min_impurity_decrease_grid, cv = 4, scor
gridsearch_9.fit(X_train, y_train)
gm = gridsearched_model(gridsearch_9.best_estimator_, 'Extra_Trees_Gridsear
bp = gridsearch_9.best_params_
bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

min_impurity_decrease = bp2[bp2['index'] == 'model__min_impurity_decrease']
```

```
        min_impurity_decrease = min_impurity_decrease[0].values[0]

        print(f'Best min_impurity_decrease: {min_impurity_decrease}')

        # GRID SEARCH 10 -- BOOTSTRAP
        bootstrap_grid = {'model__n_estimators' : [n_est],
                          'model__criterion' : [criterion],
                          'model__max_depth' : [max_depth],
                          'model__min_samples_split' : [min_samples_split],
                          'model__min_samples_leaf' : [min_samples_leaf],
                          'model__min_weight_fraction_leaf' : [min_weight_fractio
                          'model__max_features' : [max_features],
                          'model__max_leaf_nodes' : [max_leaf_nodes],
                          'model__min_impurity_decrease' : [min_impurity_decrease
                          'model__bootstrap' : [False, True],
                          'model__oob_score' : [False],
                          'model__n_jobs' : [20],
                          'model__ccp_alpha' : [0.0]}
        gridsearch_10 = GridSearchCV(model, bootstrap_grid, cv = 4, scoring = 'accu
        gridsearch_10.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_10.best_estimator_, 'Extra_Trees_Gridsea
        bp = gridsearch_10.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

        bootstrap = bp2[bp2['index'] == 'model__bootstrap']
        bootstrap = bootstrap[0].values[0]

        print(f'Best bootstrap: {bootstrap}')

        # GRID SEARCH 11 -- OOB SCORE
        oob_score_grid = {'model__n_estimators' : [n_est],
                          'model__criterion' : [criterion],
                          'model__max_depth' : [max_depth],
                          'model__min_samples_split' : [min_samples_split],
                          'model__min_samples_leaf' : [min_samples_leaf],
                          'model__min_weight_fraction_leaf' : [min_weight_fractio
                          'model__max_features' : [max_features],
                          'model__max_leaf_nodes' : [max_leaf_nodes],
                          'model__min_impurity_decrease' : [min_impurity_decrease
                          'model__bootstrap' : [bootstrap],
                          'model__oob_score' : [False, True],
                          'model__n_jobs' : [20],
                          'model__ccp_alpha' : [0.0]}
        gridsearch_11 = GridSearchCV(model, oob_score_grid, cv = 4, scoring = 'accu
        gridsearch_11.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_11.best_estimator_, 'Extra_Trees_Gridsea
        bp = gridsearch_11.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

        oob_score = bp2[bp2['index'] == 'model__oob_score']
        oob_score = oob_score[0].values[0]

        print(f'Best oob_score: {oob_score}')

        # GRID SEARCH 12 -- CCP ALPHA
        ccp_alpha_grid = {'model__n_estimators' : [n_est],
```

```
                            'model__criterion' : [criterion],
                            'model__max_depth' : [max_depth],
                            'model__min_samples_split' : [min_samples_split],
                            'model__min_samples_leaf' : [min_samples_leaf],
                            'model__min_weight_fraction_leaf' : [min_weight_fractio
                            'model__max_features' : [max_features],
                            'model__max_leaf_nodes' : [max_leaf_nodes],
                            'model__min_impurity_decrease' : [min_impurity_decrease
                            'model__bootstrap' : [bootstrap],
                            'model__oob_score' : [oob_score],
                            'model__n_jobs' : [20],
                            'model__ccp_alpha' : [0.0, 0.1, 0.2, 0.3, 0.4, 0.5]}
        gridsearch_12 = GridSearchCV(model, ccp_alpha_grid, cv = 4, scoring = 'accu
        gridsearch_12.fit(X_train, y_train)
        gm = gridsearched_model(gridsearch_12.best_estimator_, 'Extra_Trees_Gridsea
        bp = gridsearch_12.best_params_
        bp2 = pd.DataFrame.from_dict(bp, orient='index').reset_index()

        ccp_alpha = bp2[bp2['index'] == 'model__ccp_alpha']
        ccp_alpha = ccp_alpha[0].values[0]

        print(f'Best ccp_alpha: {ccp_alpha}')

        return gm, gridsearch_1.best_params_, gridsearch_2.best_params_, gridsearch
```

In [ ]:
```
extra_trees_gridsearch(extra_trees)
```

```
Fitting 4 folds for each of 10 candidates, totalling 40 fits
[CV 1/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_jobs
=20, model__oob_score=False;, score=0.698 total time=   0.7s
[CV 2/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_jobs
=20, model__oob_score=False;, score=0.704 total time=   0.7s
[CV 3/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_jobs
=20, model__oob_score=False;, score=0.707 total time=   0.6s
[CV 4/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=100, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   0.6s
[CV 1/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_jobs
=20, model__oob_score=False;, score=0.687 total time=   0.6s
[CV 2/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_j
obs=20, model__oob_score=False
```

```
[CV 2/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_jobs
=20, model__oob_score=False;, score=0.708 total time=   0.6s
[CV 3/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_jobs
=20, model__oob_score=False;, score=0.702 total time=   0.6s
[CV 4/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=200, model__n_jobs
=20, model__oob_score=False;, score=0.711 total time=   0.6s
[CV 1/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_jobs
=20, model__oob_score=False;, score=0.685 total time=   0.8s
[CV 2/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   0.8s
[CV 3/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   0.8s
[CV 4/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
```

```
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=300, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   0.7s
[CV 1/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_jobs
=20, model__oob_score=False;, score=0.691 total time=   0.9s
[CV 2/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   0.9s
[CV 3/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_jobs
=20, model__oob_score=False;, score=0.699 total time=   0.8s
[CV 4/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=400, model__n_jobs
=20, model__oob_score=False;, score=0.708 total time=   0.9s
[CV 1/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_jobs
=20, model__oob_score=False;, score=0.691 total time=   1.0s
[CV 2/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_jobs
=20, model__oob_score=False;, score=0.704 total time=   0.9s
[CV 3/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_jobs
=20, model__oob_score=False;, score=0.700 total time=   0.9s
[CV 4/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=500, model__n_jobs
=20, model__oob_score=False;, score=0.719 total time=   0.9s
[CV 1/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_jobs
=20, model__oob_score=False;, score=0.688 total time=   1.0s
[CV 2/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_jobs
=20, model__oob_score=False;, score=0.716 total time=   1.0s
[CV 3/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
```

```
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.0s
[CV 4/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=600, model__n_jobs
=20, model__oob_score=False;, score=0.709 total time=   1.0s
[CV 1/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_jobs
=20, model__oob_score=False;, score=0.693 total time=   1.2s
[CV 2/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.3s
[CV 3/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.3s
[CV 4/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=700, model__n_jobs
```

```
=20, model__oob_score=False;, score=0.712 total time=   1.2s
[CV 1/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.689 total time=   1.3s
[CV 2/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.720 total time=   1.3s
[CV 3/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.701 total time=   1.3s
[CV 4/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.721 total time=   1.2s
[CV 1/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_jobs
=20, model__oob_score=False;, score=0.697 total time=   1.5s
[CV 2/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_j
obs=20, model__oob_score=False
```

```
[CV 2/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.8s
[CV 3/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_jobs
=20, model__oob_score=False;, score=0.703 total time=   1.4s
[CV 4/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=900, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.5s
[CV 1/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n_
jobs=20, model__oob_score=False;, score=0.689 total time=   1.5s
[CV 2/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n_
jobs=20, model__oob_score=False;, score=0.706 total time=   1.5s
[CV 3/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n_
jobs=20, model__oob_score=False;, score=0.699 total time=   1.5s
[CV 4/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
```

```
on=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=1000, model__n_
jobs=20, model__oob_score=False;, score=0.711 total time=   1.4s
Best n_estimators: 800
Fitting 4 folds for each of 2 candidates, totalling 8 fits
[CV 1/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=g
ini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.689 total time=   1.5s
[CV 2/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=g
ini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.705 total time=   1.5s
[CV 3/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=g
ini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.703 total time=   1.4s
[CV 4/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=gini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=g
ini, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.4s
[CV 1/4; 2/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
```

```
[CV 1/4; 2/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.689 total time=   1.3s
[CV 2/4; 2/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 2/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.714 total time=   1.3s
[CV 3/4; 2/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 2/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.701 total time=   1.5s
[CV 4/4; 2/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 2/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.713 total time=   1.5s
Best criterion: entropy
Fitting 4 folds for each of 30 candidates, totalling 120 fits
[CV 1/4; 1/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 1/4; 1/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.690 total time=   1.4s
[CV 2/4; 1/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 2/4; 1/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
```

```
jobs=20, model__oob_score=False;, score=0.707 total time=   1.4s
[CV 3/4; 1/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 3/4; 1/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.704 total time=   1.5s
[CV 4/4; 1/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 4/4; 1/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=None, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.711 total time=   1.4s
[CV 1/4; 2/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 2/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.710 total time=   1.1s
[CV 2/4; 2/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 2/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.699 total time=   1.0s
[CV 3/4; 2/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 2/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.688 total time=   1.3s
[CV 4/4; 2/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
```

```
[CV 4/4; 2/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=1, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.692 total time=   1.0s
[CV 1/4; 3/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 3/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.704 total time=   1.0s
[CV 2/4; 3/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 3/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.707 total time=   1.1s
[CV 3/4; 3/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 3/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.689 total time=   1.1s
[CV 4/4; 3/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 3/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=2, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.699 total time=   1.0s
[CV 1/4; 4/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 4/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   0.9s
[CV 2/4; 4/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
```

```
n=entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 4/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.703 total time=   1.0s
[CV 3/4; 4/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 4/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.693 total time=   1.0s
[CV 4/4; 4/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 4/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=3, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.703 total time=   1.0s
[CV 1/4; 5/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 5/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   1.0s
[CV 2/4; 5/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 5/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.0s
[CV 3/4; 5/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 5/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.691 total time=   1.0s
[CV 4/4; 5/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 5/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=4, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.705 total time=   1.1s
[CV 1/4; 6/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 6/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.0s
[CV 2/4; 6/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 6/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.708 total time=   0.9s
[CV 3/4; 6/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 6/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.693 total time=   0.9s
[CV 4/4; 6/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 6/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=5, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.705 total time=   1.1s
[CV 1/4; 7/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
```

```
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 7/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.0s
[CV 2/4; 7/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 7/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.711 total time=   1.1s
[CV 3/4; 7/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 7/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.697 total time=   1.0s
[CV 4/4; 7/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 7/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=6, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.708 total time=   1.1s
[CV 1/4; 8/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 8/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 2/4; 8/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 8/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
```

```
=20, model__oob_score=False;, score=0.713 total time=   1.0s
[CV 3/4; 8/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 8/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.697 total time=   1.1s
[CV 4/4; 8/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 8/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=7, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 1/4; 9/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 9/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.1s
[CV 2/4; 9/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 9/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.717 total time=   1.1s
[CV 3/4; 9/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 9/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.2s
[CV 4/4; 9/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
```

```
[CV 4/4; 9/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 1/4; 10/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 10/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.712 total time=   1.2s
[CV 2/4; 10/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 10/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.716 total time=   1.1s
[CV 3/4; 10/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 10/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.699 total time=   1.2s
[CV 4/4; 10/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 10/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=9, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 1/4; 11/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 11/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 2/4; 11/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
```

```
on=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 11/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.720 total time=   1.1s
[CV 3/4; 11/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 11/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.697 total time=   1.1s
[CV 4/4; 11/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 11/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=10, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.709 total time=   1.2s
[CV 1/4; 12/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 12/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.698 total time=   1.4s
[CV 2/4; 12/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 12/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.718 total time=   1.3s
[CV 3/4; 12/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 12/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=Non
```

```
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.702 total time=   1.2s
[CV 4/4; 12/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 12/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=15, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.711 total time=   1.2s
[CV 1/4; 13/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 13/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.692 total time=   1.3s
[CV 2/4; 13/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 13/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.717 total time=   1.4s
[CV 3/4; 13/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 13/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.706 total time=   1.4s
[CV 4/4; 13/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 13/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=20, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.713 total time=   1.4s
[CV 1/4; 14/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
```

```
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 14/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.690 total time=   1.4s
[CV 2/4; 14/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 14/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.713 total time=   1.3s
[CV 3/4; 14/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 14/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.708 total time=   1.3s
[CV 4/4; 14/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 14/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=25, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.704 total time=   1.4s
[CV 1/4; 15/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 15/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.690 total time=   1.3s
[CV 2/4; 15/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 15/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
```

```
obs=20, model__oob_score=False;, score=0.711 total time=   1.4s
[CV 3/4; 15/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 15/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.703 total time=   1.5s
[CV 4/4; 15/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 15/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=30, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.713 total time=   1.4s
[CV 1/4; 16/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 16/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.690 total time=   1.4s
[CV 2/4; 16/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 16/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.717 total time=   1.4s
[CV 3/4; 16/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 16/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.701 total time=   1.4s
[CV 4/4; 16/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
```

```
[CV 4/4; 16/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=35, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.710 total time=   1.4s
[CV 1/4; 17/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 17/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.690 total time=   1.4s
[CV 2/4; 17/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 17/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.706 total time=   1.4s
[CV 3/4; 17/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 17/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.704 total time=   1.4s
[CV 4/4; 17/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 17/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=40, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.708 total time=   1.8s
[CV 1/4; 18/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 18/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.691 total time=   1.4s
[CV 2/4; 18/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
```

```
on=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 18/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.715 total time=   1.4s
[CV 3/4; 18/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 18/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.705 total time=   1.3s
[CV 4/4; 18/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 18/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=45, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.713 total time=   1.4s
[CV 1/4; 19/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 19/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.693 total time=   1.3s
[CV 2/4; 19/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 19/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.710 total time=   1.4s
[CV 3/4; 19/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 19/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=Non
```

```
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.705 total time=   1.5s
[CV 4/4; 19/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 19/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=50, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.712 total time=   1.4s
[CV 1/4; 20/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 20/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.694 total time=   1.4s
[CV 2/4; 20/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 20/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.704 total time=   1.4s
[CV 3/4; 20/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 20/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.705 total time=   1.6s
[CV 4/4; 20/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 20/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=55, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.708 total time=   1.4s
[CV 1/4; 21/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
```

```
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 21/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.694 total time=   1.5s
[CV 2/4; 21/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 21/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.714 total time=   1.3s
[CV 3/4; 21/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 21/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.693 total time=   1.5s
[CV 4/4; 21/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 21/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=60, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.719 total time=   1.4s
[CV 1/4; 22/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 22/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.691 total time=   1.3s
[CV 2/4; 22/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 22/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
```

```
obs=20, model__oob_score=False;, score=0.713 total time=   1.4s
[CV 3/4; 22/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 22/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.706 total time=   1.4s
[CV 4/4; 22/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 22/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=65, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.710 total time=   1.4s
[CV 1/4; 23/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 23/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.693 total time=   1.4s
[CV 2/4; 23/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 23/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.711 total time=   1.6s
[CV 3/4; 23/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 23/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.702 total time=   1.3s
[CV 4/4; 23/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
```

```
[CV 4/4; 23/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=70, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.710 total time=   1.5s
[CV 1/4; 24/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 24/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.687 total time=   1.3s
[CV 2/4; 24/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 24/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.708 total time=   1.4s
[CV 3/4; 24/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 24/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.702 total time=   1.4s
[CV 4/4; 24/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 24/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=75, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.711 total time=   1.5s
[CV 1/4; 25/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 25/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.683 total time=   1.4s
[CV 2/4; 25/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
```

```
on=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 25/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.711 total time=   1.4s
[CV 3/4; 25/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 25/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.696 total time=   1.9s
[CV 4/4; 25/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 25/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=80, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.702 total time=   1.4s
[CV 1/4; 26/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 26/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.686 total time=   1.3s
[CV 2/4; 26/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 26/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.705 total time=   1.4s
[CV 3/4; 26/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 26/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=Non
```

```
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.704 total time=   1.4s
[CV 4/4; 26/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 26/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=85, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.713 total time=   1.3s
[CV 1/4; 27/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 27/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.683 total time=   1.3s
[CV 2/4; 27/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 27/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.714 total time=   1.3s
[CV 3/4; 27/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 27/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.701 total time=   1.3s
[CV 4/4; 27/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 27/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=90, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.705 total time=   1.3s
[CV 1/4; 28/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
```

```
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 1/4; 28/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.695 total time=   1.3s
[CV 2/4; 28/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 2/4; 28/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.709 total time=   1.4s
[CV 3/4; 28/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 3/4; 28/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.704 total time=   1.3s
[CV 4/4; 28/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=N
one, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sampl
es_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False
[CV 4/4; 28/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=95, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False;, score=0.711 total time=   1.4s
[CV 1/4; 29/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 1/4; 29/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.681 total time=   1.5s
[CV 2/4; 29/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 2/4; 29/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
```

```
jobs=20, model__oob_score=False;, score=0.716 total time=   1.3s
[CV 3/4; 29/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 3/4; 29/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.706 total time=   1.3s
[CV 4/4; 29/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 4/4; 29/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=100, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.713 total time=   1.4s
[CV 1/4; 30/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 1/4; 30/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.699 total time=   1.3s
[CV 2/4; 30/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 2/4; 30/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.718 total time=   1.3s
[CV 3/4; 30/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 3/4; 30/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.708 total time=   1.5s
[CV 4/4; 30/30] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=
None, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samp
les_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
```

```
[CV 4/4; 30/30] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=150, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_sample
s_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False;, score=0.713 total time=   1.5s
Best max_depth: 8
Fitting 4 folds for each of 9 candidates, totalling 36 fits
[CV 1/4; 1/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 1/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.716 total time=   1.1s
[CV 2/4; 1/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 1/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 3/4; 1/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 1/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.699 total time=   1.1s
[CV 4/4; 1/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 1/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=2, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 1/4; 2/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 2/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
```

```
=20, model__oob_score=False;, score=0.713 total time=   1.1s
[CV 2/4; 2/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 2/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 3/4; 2/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 2/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.696 total time=   1.1s
[CV 4/4; 2/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 2/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=3, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.5s
[CV 1/4; 3/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 3/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.716 total time=   1.2s
[CV 2/4; 3/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 3/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.716 total time=   1.1s
[CV 3/4; 3/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
```

```
[CV 3/4; 3/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.692 total time=   1.1s
[CV 4/4; 3/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 3/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=4, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 1/4; 4/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 4/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 2/4; 4/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 4/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 3/4; 4/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 4/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.700 total time=   1.1s
[CV 4/4; 4/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 4/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=5, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.1s
[CV 1/4; 5/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
```

```
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 5/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 2/4; 5/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 5/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.1s
[CV 3/4; 5/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 5/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.701 total time=   1.1s
[CV 4/4; 5/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 5/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=6, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   1.2s
[CV 1/4; 6/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 6/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 2/4; 6/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 6/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 3/4; 6/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 6/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.698 total time=   1.1s
[CV 4/4; 6/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 6/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=7, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.710 total time=   1.1s
[CV 1/4; 7/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 7/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 7/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 7/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.717 total time=   1.1s
[CV 3/4; 7/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 7/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.700 total time=   1.1s
[CV 4/4; 7/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
```

```
_split=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 7/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=8, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.713 total time=   1.1s
[CV 1/4; 8/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 1/4; 8/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 2/4; 8/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 2/4; 8/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 3/4; 8/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 3/4; 8/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.696 total time=   1.1s
[CV 4/4; 8/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_j
obs=20, model__oob_score=False
[CV 4/4; 8/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=9, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.0s
[CV 1/4; 9/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 9/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
```

```
s=20, model__oob_score=False;, score=0.718 total time=   1.2s
[CV 2/4; 9/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 9/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.721 total time=   1.1s
[CV 3/4; 9/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 9/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.696 total time=   1.1s
[CV 4/4; 9/9] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 9/9] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.1s
Best min_samples_split: 10
Fitting 4 folds for each of 10 candidates, totalling 40 fits
[CV 1/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 2/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 3/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
```

```
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.702 total time=   1.1s
[CV 4/4; 1/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 1/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=1, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.1s
[CV 1/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.722 total time=   1.2s
[CV 2/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 3/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.696 total time=   1.1s
[CV 4/4; 2/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 2/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=2, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
```

```
s=20, model__oob_score=False;, score=0.711 total time=   1.3s
[CV 1/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 2/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.719 total time=   1.1s
[CV 3/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.692 total time=   1.1s
[CV 4/4; 3/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 3/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=3, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.2s
[CV 1/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
```

```
[CV 2/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.2s
[CV 3/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.697 total time=   1.2s
[CV 4/4; 4/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 4/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=4, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.2s
[CV 1/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 2/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 3/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.696 total time=   1.1s
[CV 4/4; 5/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
```

```
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 5/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=5, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.2s
[CV 1/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 2/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.716 total time=   1.1s
[CV 3/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.692 total time=   1.0s
[CV 4/4; 6/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 6/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=6, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.710 total time=   1.2s
[CV 1/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 2/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.716 total time=   1.1s
[CV 3/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.695 total time=   1.2s
[CV 4/4; 7/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 7/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=7, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 1/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.3s
[CV 2/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.719 total time=   1.2s
[CV 3/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
```

```
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.701 total time=   1.1s
[CV 4/4; 8/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 8/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.713 total time=   1.2s
[CV 1/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 3/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.697 total time=   1.1s
[CV 4/4; 9/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 9/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=9, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
```

```
s=20, model__oob_score=False;, score=0.713 total time=   1.3s
[CV 1/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sampl
es_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 1/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sample
s_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False;, score=0.718 total time=   1.2s
[CV 2/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sampl
es_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 2/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sample
s_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 3/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sampl
es_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 3/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sample
s_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False;, score=0.697 total time=   1.1s
[CV 4/4; 10/10] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=No
ne, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sampl
es_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__
n_jobs=20, model__oob_score=False
[CV 4/4; 10/10] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=10, model__min_sample
s_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n
_jobs=20, model__oob_score=False;, score=0.712 total time=   1.2s
Best min_samples_leaf: 8
Fitting 4 folds for each of 6 candidates, totalling 24 fits
[CV 1/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.2s
[CV 2/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
```

```
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.716 total time=   1.1s
[CV 3/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.698 total time=   1.1s
[CV 4/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.2s
[CV 1/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.709 total time=   1.0s
[CV 2/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.701 total time=   1.1s
[CV 3/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_job
```

```
s=20, model__oob_score=False;, score=0.688 total time=   1.0s
[CV 4/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.1, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.694 total time=   1.1s
[CV 1/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.695 total time=   1.0s
[CV 2/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.699 total time=   0.9s
[CV 3/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.688 total time=   1.1s
[CV 4/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.2, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.687 total time=   0.9s
[CV 1/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
```

```
[CV 1/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.693 total time=   1.1s
[CV 2/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.694 total time=   1.1s
[CV 3/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.683 total time=   1.0s
[CV 4/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.3, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.677 total time=   1.2s
[CV 1/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.676 total time=   1.0s
[CV 2/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.675 total time=   1.1s
[CV 3/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
```

```
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.659 total time=   1.1s
[CV 4/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.4, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.658 total time=   1.0s
[CV 1/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.1s
[CV 2/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 3/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 4/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.5, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
Best min_weight_fraction_leaf: 0.0
Fitting 4 folds for each of 3 candidates, totalling 12 fits
[CV 1/4; 1/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 1/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 1/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 1/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 3/4; 1/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 1/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.698 total time=   1.2s
[CV 4/4; 1/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 1/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 1/4; 2/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 2/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.712 total time=   1.1s
[CV 2/4; 2/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
```

```
=entropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 2/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 3/4; 2/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 2/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.698 total time=   1.1s
[CV 4/4; 2/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 2/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=sqrt, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.1s
[CV 1/4; 3/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 3/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.699 total time=   1.1s
[CV 2/4; 3/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 3/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.1s
[CV 3/4; 3/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 3/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.706 total time=   0.9s
[CV 4/4; 3/3] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 3/3] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=log2, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.2s
Best max_features: auto
Fitting 4 folds for each of 11 candidates, totalling 44 fits
[CV 1/4; 1/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 1/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 2/4; 1/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 1/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.3s
[CV 3/4; 1/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 1/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.697 total time=   1.1s
[CV 4/4; 1/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 1/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.709 total time=   1.2s
[CV 1/4; 2/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
```

```
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 2/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.705 total time=   1.0s
[CV 2/4; 2/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 2/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.700 total time=   1.0s
[CV 3/4; 2/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 2/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.683 total time=   1.0s
[CV 4/4; 2/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 2/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.702 total time=   1.1s
[CV 1/4; 3/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 3/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.712 total time=   1.0s
[CV 2/4; 3/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 3/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4, mo
```

```
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.706 total time=   1.0s
[CV 3/4; 3/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 3/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.690 total time=   1.1s
[CV 4/4; 3/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 3/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=4, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.699 total time=   1.1s
[CV 1/4; 4/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 4/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.717 total time=   1.0s
[CV 2/4; 4/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 4/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.699 total time=   1.1s
[CV 3/4; 4/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 4/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.687 total time=   1.2s
[CV 4/4; 4/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
```

```
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 4/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=6, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.700 total time=   1.1s
[CV 1/4; 5/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 5/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.718 total time=   1.0s
[CV 2/4; 5/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 5/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.704 total time=   0.9s
[CV 3/4; 5/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 5/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.689 total time=   1.0s
[CV 4/4; 5/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 5/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=8, mo
del__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spli
t=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs=
20, model__oob_score=False;, score=0.704 total time=   1.1s
[CV 1/4; 6/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 6/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
```

```
=20, model__oob_score=False;, score=0.711 total time=   1.0s
[CV 2/4; 6/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 6/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.1s
[CV 3/4; 6/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 6/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.692 total time=   1.1s
[CV 4/4; 6/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 6/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=10, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.1s
[CV 1/4; 7/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 7/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 2/4; 7/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 7/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.706 total time=   1.1s
[CV 3/4; 7/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
```

```
[CV 3/4; 7/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.686 total time=   0.9s
[CV 4/4; 7/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 7/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=12, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.705 total time=   1.1s
[CV 1/4; 8/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 8/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.714 total time=   1.1s
[CV 2/4; 8/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 8/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.702 total time=   1.0s
[CV 3/4; 8/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 8/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.693 total time=   0.9s
[CV 4/4; 8/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 8/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=14, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.707 total time=   1.1s
[CV 1/4; 9/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
```

```
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 9/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 9/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 9/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.708 total time=   1.1s
[CV 3/4; 9/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 9/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.694 total time=   1.1s
[CV 4/4; 9/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterio
n=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 9/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=16, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.708 total time=   1.1s
[CV 1/4; 10/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=1
8, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 10/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=18,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.720 total time=   1.1s
[CV 2/4; 10/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=1
8, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 10/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=18,
```

```
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.706 total time=   1.1s
[CV 3/4; 10/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=1
8, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 10/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=18,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.693 total time=   1.2s
[CV 4/4; 10/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=1
8, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 10/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=18,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.708 total time=   1.2s
[CV 1/4; 11/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2
0, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 11/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=20,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 11/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2
0, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 11/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=20,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.707 total time=   1.1s
[CV 3/4; 11/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2
0, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 11/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=20,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.693 total time=   1.1s
[CV 4/4; 11/11] START model__bootstrap=False, model__ccp_alpha=0.0, model__criteri
on=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=2
0, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
```

```
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 11/11] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=20,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.707 total time=   1.2s
Best max_leaf_nodes: None
Fitting 4 folds for each of 6 candidates, totalling 24 fits
[CV 1/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.718 total time=   1.1s
[CV 2/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.717 total time=   1.1s
[CV 3/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.698 total time=   1.2s
[CV 4/4; 1/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 1/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.2s
[CV 1/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
```

```
model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   0.9s
[CV 2/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 3/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 4/4; 2/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 2/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.1, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 1/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 2/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 3/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples
```

```
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 4/4; 3/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 3/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.2, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 1/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.1s
[CV 2/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 3/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 4/4; 4/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 4/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.3, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
```

```
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 1/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.1s
[CV 2/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 3/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 4/4; 5/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 5/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.4, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 1/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 2/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
```

```
[CV 2/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.501 total time=   1.0s
[CV 3/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
[CV 4/4; 6/6] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 6/6] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.5, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.502 total time=   1.0s
Best min_impurity_decrease: 0.0
Fitting 4 folds for each of 2 candidates, totalling 8 fits
[CV 1/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 1/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.711 total time=   1.1s
[CV 2/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 2/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.721 total time=   1.1s
[CV 3/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 3/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
```

```
s=20, model__oob_score=False;, score=0.696 total time=   1.3s
[CV 4/4; 1/2] START model__bootstrap=False, model__ccp_alpha=0.0, model__criterion
=entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=Non
e, model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples
_split=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_
jobs=20, model__oob_score=False
[CV 4/4; 1/2] END model__bootstrap=False, model__ccp_alpha=0.0, model__criterion=e
ntropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False;, score=0.708 total time=   1.1s
[CV 1/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.717 total time=   1.2s
[CV 3/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.696 total time=   1.1s
[CV 4/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.710 total time=   1.1s
Best bootstrap: True
Fitting 4 folds for each of 2 candidates, totalling 8 fits
[CV 1/4; 1/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
```

```
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 1/4; 1/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.1s
[CV 2/4; 1/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 2/4; 1/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.715 total time=   1.2s
[CV 3/4; 1/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 3/4; 1/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.699 total time=   1.1s
[CV 4/4; 1/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=False
[CV 4/4; 1/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=False;, score=0.709 total time=   1.1s
[CV 1/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.716 total time=   2.2s
[CV 2/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
```

```
=20, model__oob_score=True;, score=0.718 total time=   2.2s
[CV 3/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 3/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.701 total time=   2.3s
[CV 4/4; 2/2] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 2/2] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.711 total time=   2.4s
Best oob_score: True
Fitting 4 folds for each of 6 candidates, totalling 24 fits
[CV 1/4; 1/6] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 1/6] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.714 total time=   2.2s
[CV 2/4; 1/6] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 1/6] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.717 total time=   2.2s
[CV 3/4; 1/6] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 3/4; 1/6] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.702 total time=   2.2s
[CV 4/4; 1/6] START model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
```

```
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 1/6] END model__bootstrap=True, model__ccp_alpha=0.0, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.710 total time=   2.2s
[CV 1/4; 2/6] START model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 2/6] END model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.1s
[CV 2/4; 2/6] START model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 2/6] END model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.0s
[CV 3/4; 2/6] START model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 3/4; 2/6] END model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.1s
[CV 4/4; 2/6] START model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 2/6] END model__bootstrap=True, model__ccp_alpha=0.1, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.1s
[CV 1/4; 3/6] START model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 3/6] END model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
```

```
=20, model__oob_score=True;, score=0.501 total time=   2.2s
[CV 2/4; 3/6] START model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 3/6] END model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.0s
[CV 3/4; 3/6] START model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 3/4; 3/6] END model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.0s
[CV 4/4; 3/6] START model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 3/6] END model__bootstrap=True, model__ccp_alpha=0.2, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.1s
[CV 1/4; 4/6] START model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 4/6] END model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.1s
[CV 2/4; 4/6] START model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 4/6] END model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.1s
[CV 3/4; 4/6] START model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
```

```
[CV 3/4; 4/6] END model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.0s
[CV 4/4; 4/6] START model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 4/6] END model__bootstrap=True, model__ccp_alpha=0.3, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.1s
[CV 1/4; 5/6] START model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 5/6] END model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.0s
[CV 2/4; 5/6] START model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 5/6] END model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.0s
[CV 3/4; 5/6] START model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 3/4; 5/6] END model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.0s
[CV 4/4; 5/6] START model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 5/6] END model__bootstrap=True, model__ccp_alpha=0.4, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.1s
[CV 1/4; 6/6] START model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=
```

```
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 1/4; 6/6] END model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.0s
[CV 2/4; 6/6] START model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 2/4; 6/6] END model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.501 total time=   2.0s
[CV 3/4; 6/6] START model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 3/4; 6/6] END model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.0s
[CV 4/4; 6/6] START model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=
entropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None,
model__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_sp
lit=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_job
s=20, model__oob_score=True
[CV 4/4; 6/6] END model__bootstrap=True, model__ccp_alpha=0.5, model__criterion=en
tropy, model__max_depth=8, model__max_features=auto, model__max_leaf_nodes=None, m
odel__min_impurity_decrease=0.0, model__min_samples_leaf=8, model__min_samples_spl
it=10, model__min_weight_fraction_leaf=0.0, model__n_estimators=800, model__n_jobs
=20, model__oob_score=True;, score=0.502 total time=   2.1s
Best ccp_alpha: 0.0
```

Out[ ]: (

	Model	Cv_Mean_Accuracy	Cv_Std
0	Extra_Trees_Gridsearched_2	0.726	0.018
0	Extra_Trees_Gridsearched_1	0.720	0.009
0	Random_Forest_Gridsearched_2	0.719	0.009
0	Random_Forest_mf_auto	0.718	0.010
0	Random_Forest_mf_sqrt	0.716	0.014
0	Random_Forest_500	0.714	0.012
0	Random_Forest_1000	0.713	0.016
0	Random_Forest_Gridsearched_4	0.713	0.006
0	Random_Forest_Gridsearched_6	0.711	0.009
0	Extra_Trees_Gridsearched_11	0.711	0.016
0	Random_Forest_Gridsearched_8	0.710	0.006
0	XGBoost_Gridsearched_1	0.710	0.014
0	Random_Forest_Gridsearched_1	0.710	0.006
0	Random_Forest	0.710	0.005
0	Random_Forest_Gridsearched_7	0.709	0.008
0	Random_Forest_Gridsearched_5	0.709	0.005
0	Extra_Trees_Gridsearched_10	0.709	0.015
0	Extra_Trees	0.709	0.011
0	Extra_Trees_Gridsearched_4	0.708	0.014
0	Extra_Trees_Gridsearched_7	0.708	0.015
0	Extra_Trees_Gridsearched_8	0.707	0.012
0	Extra_Trees_Gridsearched_3	0.707	0.011
0	Extra_Trees_Gridsearched_9	0.707	0.015
0	Random_Forest_Gridsearched_3	0.706	0.015
0	Extra_Trees_Gridsearched_5	0.706	0.015
0	Extra_Trees_Gridsearched_6	0.706	0.018
0	XGBoost_Gridsearched_10	0.705	0.015
0	XGBoost_Gridsearched_12	0.705	0.015
0	XGBoost_Gridsearched_14	0.705	0.015
0	XGBoost_Gridsearched_11	0.705	0.015
0	XGBoost_Gridsearched_13	0.705	0.015
0	XGBoost	0.705	0.013
0	XGBoost_Gridsearched_6	0.702	0.011
0	XGBoost_Gridsearched_5	0.702	0.011
0	Extra_Trees_Gridsearched_12	0.702	0.016
0	XGBoost_Gridsearched_4	0.701	0.013
0	XGBoost_Gridsearched_9	0.701	0.016
0	XGBoost_Gridsearched_7	0.701	0.016
0	XGBoost_Gridsearched_8	0.701	0.016
0	XGBoost_Gridsearched_2	0.693	0.018
0	XGBoost_Gridsearched_3	0.693	0.018
0	Logistic_Regression	0.691	0.013
0	Bagged_Trees	0.672	0.005
0	K_Neighbors	0.629	0.005
0	Decision_Tree	0.600	0.011,


```
{'model__bootstrap': False,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'gini',
 'model__max_depth': None,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 2,
 'model__min_weight_fraction_leaf': 0.0,
```

```
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': False},
{'model__bootstrap': False,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'entropy',
 'model__max_depth': None,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 2,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': False},
{'model__bootstrap': False,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'entropy',
 'model__max_depth': 8,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 2,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': False},
{'model__bootstrap': False,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'entropy',
 'model__max_depth': 8,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 1,
 'model__min_samples_split': 10,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': False},
{'model__bootstrap': False,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'entropy',
 'model__max_depth': 8,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 8,
 'model__min_samples_split': 10,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': False},
{'model__bootstrap': False,
```

```
  'model__ccp_alpha': 0.0,
  'model__criterion': 'entropy',
  'model__max_depth': 8,
  'model__max_features': 'auto',
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 8,
  'model__min_samples_split': 10,
  'model__min_weight_fraction_leaf': 0.0,
  'model__n_estimators': 800,
  'model__n_jobs': 20,
  'model__oob_score': False},
 {'model__bootstrap': False,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'entropy',
  'model__max_depth': 8,
  'model__max_features': 'auto',
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 8,
  'model__min_samples_split': 10,
  'model__min_weight_fraction_leaf': 0.0,
  'model__n_estimators': 800,
  'model__n_jobs': 20,
  'model__oob_score': False},
 {'model__bootstrap': False,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'entropy',
  'model__max_depth': 8,
  'model__max_features': 'auto',
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 8,
  'model__min_samples_split': 10,
  'model__min_weight_fraction_leaf': 0.0,
  'model__n_estimators': 800,
  'model__n_jobs': 20,
  'model__oob_score': False},
 {'model__bootstrap': False,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'entropy',
  'model__max_depth': 8,
  'model__max_features': 'auto',
  'model__max_leaf_nodes': None,
  'model__min_impurity_decrease': 0.0,
  'model__min_samples_leaf': 8,
  'model__min_samples_split': 10,
  'model__min_weight_fraction_leaf': 0.0,
  'model__n_estimators': 800,
  'model__n_jobs': 20,
  'model__oob_score': False},
 {'model__bootstrap': True,
  'model__ccp_alpha': 0.0,
  'model__criterion': 'entropy',
  'model__max_depth': 8,
  'model__max_features': 'auto',
```

```
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 8,
 'model__min_samples_split': 10,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': False},
{'model__bootstrap': True,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'entropy',
 'model__max_depth': 8,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 8,
 'model__min_samples_split': 10,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': True},
{'model__bootstrap': True,
 'model__ccp_alpha': 0.0,
 'model__criterion': 'entropy',
 'model__max_depth': 8,
 'model__max_features': 'auto',
 'model__max_leaf_nodes': None,
 'model__min_impurity_decrease': 0.0,
 'model__min_samples_leaf': 8,
 'model__min_samples_split': 10,
 'model__min_weight_fraction_leaf': 0.0,
 'model__n_estimators': 800,
 'model__n_jobs': 20,
 'model__oob_score': True})
```

Best Model

In [ ]:
```
best_model = pickle.load(open('models/Extra_Trees_Gridsearched_2.pkl', 'rb'))
```

In [ ]:
```
best_model['model'].get_params()
```

Out[ ]:
```
{'bootstrap': False,
 'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'entropy',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 800,
 'n_jobs': 20,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```

In [ ]:
```
cat_list= list(feature_names_categorical)
```

In [ ]:
```
all_cats = cat_list + numerical_columns
```

In [ ]:
```
f_imp = best_model.steps[1][1].feature_importances_
f_imp_df = pd.DataFrame(f_imp, index = all_cats, columns = ['Importance'])
f_imp_df = f_imp_df.sort_values('Importance', ascending = False)
f_imp_df = f_imp_df.round(3)
f_imp_df
```

Out[ ]:

	Importance
Dif_Opp_Avg_Ground_Strikes_land_per_round	0.015
Dif_Opp_Avg_Ground_Strikes_att_per_round	0.014
fight_weightclass_Heavyweight	0.007
fight_weightclass_Catch Weight	0.007
fight_weightclass_Bantamweight	0.006
...	...
Dif_Rolling_Clinch_Strikes_percent_mean	0.000
B_Rolling_Takedowns_att_std	0.000
A_Rolling_Takedowns_att_median	0.000
B_Opp_Avg_Leg_Strikes_att_per_round	0.000
A_Rolling_Takedown_percent_std	0.000

670 rows × 1 columns

Show the feature importances, top 50

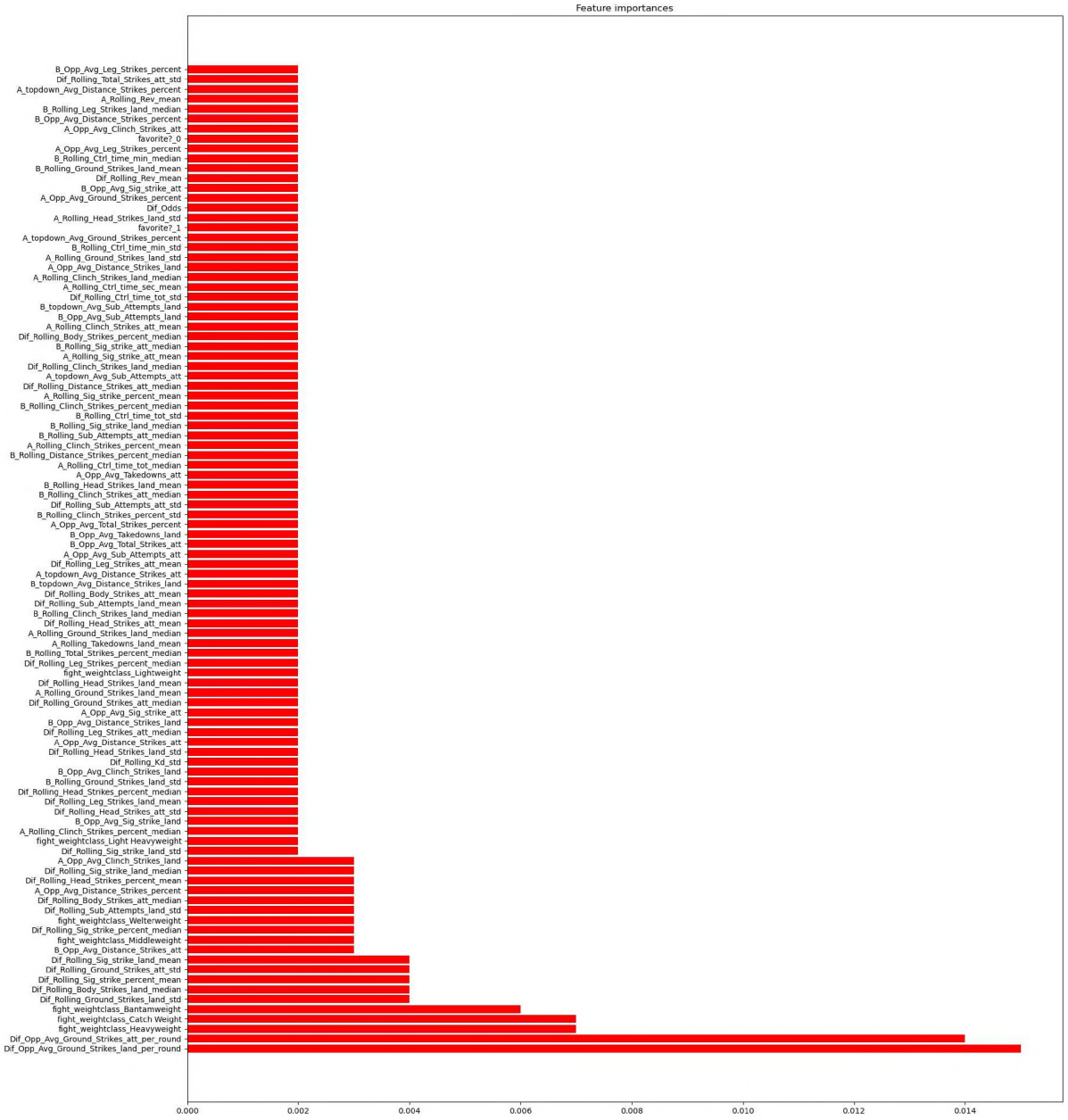
In [ ]:
```
f_imp_df[:50]
```

Out[ ]:

	Importance
Dif_Opp_Avg_Ground_Strikes_land_per_round	0.015
Dif_Opp_Avg_Ground_Strikes_att_per_round	0.014
fight_weightclass_Heavyweight	0.007
fight_weightclass_Catch Weight	0.007
fight_weightclass_Bantamweight	0.006
Dif_Rolling_Ground_Strikes_land_std	0.004
Dif_Rolling_Body_Strikes_land_median	0.004
Dif_Rolling_Sig_strike_percent_mean	0.004
Dif_Rolling_Ground_Strikes_att_std	0.004
Dif_Rolling_Sig_strike_land_mean	0.004
B_Opp_Avg_Distance_Strikes_att	0.003
fight_weightclass_Middleweight	0.003
Dif_Rolling_Sig_strike_percent_median	0.003
fight_weightclass_Welterweight	0.003
Dif_Rolling_Sub_Attempts_land_std	0.003
Dif_Rolling_Body_Strikes_att_median	0.003
A_Opp_Avg_Distance_Strikes_percent	0.003
Dif_Rolling_Head_Strikes_percent_mean	0.003
Dif_Rolling_Sig_strike_land_median	0.003
A_Opp_Avg_Clinch_Strikes_land	0.003
Dif_Rolling_Sig_strike_land_std	0.002
fight_weightclass_Light Heavyweight	0.002
A_Rolling_Clinch_Strikes_percent_median	0.002
B_Opp_Avg_Sig_strike_land	0.002
Dif_Rolling_Head_Strikes_att_std	0.002
Dif_Rolling_Leg_Strikes_land_mean	0.002
Dif_Rolling_Head_Strikes_percent_median	0.002
B_Rolling_Ground_Strikes_land_std	0.002
B_Opp_Avg_Clinch_Strikes_land	0.002
Dif_Rolling_Kd_std	0.002
Dif_Rolling_Head_Strikes_land_std	0.002
A_Opp_Avg_Distance_Strikes_att	0.002
Dif_Rolling_Leg_Strikes_att_median	0.002

	Importance
B_Opp_Avg_Distance_Strikes_land	0.002
A_Opp_Avg_Sig_strike_att	0.002
Dif_Rolling_Ground_Strikes_att_median	0.002
A_Rolling_Ground_Strikes_land_mean	0.002
Dif_Rolling_Head_Strikes_land_mean	0.002
fight_weightclass_Lightweight	0.002
Dif_Rolling_Leg_Strikes_percent_median	0.002
B_Rolling_Total_Strikes_percent_median	0.002
A_Rolling_Takedowns_land_mean	0.002
A_Rolling_Ground_Strikes_land_median	0.002
Dif_Rolling_Head_Strikes_att_mean	0.002
B_Rolling_Clinch_Strikes_land_median	0.002
Dif_Rolling_Sub_Attempts_land_mean	0.002
Dif_Rolling_Body_Strikes_att_mean	0.002
B_topdown_Avg_Distance_Strikes_land	0.002
A_topdown_Avg_Distance_Strikes_att	0.002
Dif_Rolling_Leg_Strikes_att_mean	0.002

In [ ]:
```
#plot 100 most important features, sorted by largest to smallest
f_imp_df
# plot the feature importances of the forest
plt.figure(figsize=(20, 25))
plt.title("Feature importances")
# bar plot of feature importances on the y axis
plt.barh(range(100), f_imp_df['Importance'][:100], color="r", align="center")
# x axis labels
plt.yticks(range(100), f_imp_df.index[:100])
# show the plot
plt.show()
```



Conclusion

Interestingly enough, the most important feature of this model was the difference between a fighters' ground strikes per round. While most other models that I trained included the odds in the top 5 features, this model did not. This is interesting, as it implies that the odds may not be as important as the in-fight statistics.

The final model achieved accuracy of .726.

The model's most important features included:

- Difference between fighter's average ground strikes attempted per round
- Weightclass
- Difference between median ground strikes landed per match

- Difference in submission attempts per match (on average)

The betting favorite wins approximately 62 percent of the time, indicating the lack of "market" (as in, the betting market) knowledge about what makes a winning fighter. That being said, it is our most accurate single statistical metric for predicting a fight.

Weightclass was a major factor in predicting a fight. This makes sense, as we saw (in the EDA section) that the weightclass of a fighter dramatically changes the importance of certain statistics, such as reach.

Ground strikes are strikes thrown from a ground position, meaning the martial artists are likely utilizing wrestling, sambo, or jujitsu. The ability to utilize these is indoubtedly important.

Takedown percentage is the percentage of successful takedowns a martial artist has divided by the number attempted. A takedown typically involves utilizing wrestling skills, although variations with sambo, jujitsu, and judo also occur. It may be important to note that utilizing ground strikes is typically only possible after a successful takedown.

The number of head strikes is also important, albeit less so than the factors above. Head strikes - or the number of strikes to an opponents head, using boxing, muay thai, kickboxing, etc. - is often thought of as the most valuable technique for success, but we find it just to be among them.

Finally, significant strikes percentage. This can be thought of as the accuracy of the striker, but does not reflect total output.

Future Improvements

There are many improvements that I could think to make, the first being the historical basis of a martial artists' skills. For instance, it seems that those with a Sambo background fare increasingly well against those with a Jujitsu background, but I was unable to find enough data to quantify this.

One idea that I am leaning towards would be creating a model to identify martial artist's backgrounds through scraping their wikipedia pages. Getting the coaches, trainers, locations, and some methods may help improve the general model.

Final Product

The final product can predict the winner of a UFC match with 73% accuracy. The product could be a web or phone application. Further, the product can give some indication to the viewer of WHY one martial artist may win over another martial artist, by reviewing the respective fighter's most important features. A version of this final product is available and working on streamlit.