CST4050 - Comp. 2

Student:

- Name: Iqra
- Surname: Ilyas
- Student number: M00909152
- Campus: Dubai

Driver at fault in car accidents classifier

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
sns.set_style('darkgrid')
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report, f1_score, r2_score
from sklearn import metrics
from sklearn.preprocessing import OrdinalEncoder
import lightgbm as lgb
from lazypredict.Supervised import LazyClassifier
from sklearn.dummy import DummyClassifier
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

1.1 - Introduction

Commuting is an essential part of daily human life and people use different means like personal cars, bikes or public transport for traveling. With time cars are becoming essential instead of luxury and adoption of personal cars have increased rapidly for middle class people and even lower middle class people with help of personal or car loans. With the high number of cars on the road, traffic accidents have also increased among which a very large number of accidents are done by cars. Car accidents involve hitting cars on domestic property, colliding with other cars, bikes, pedestrians etc. damages done by car collisions and accidents vary from no injury and only damage to car or someone's property, mild injury, severe injury and in most of the cases fatal injury to disability or death. Every year, around 1.3 million individuals lose their lives due to road traffic accidents. Many reason can be there behind the car accidents and intensity and severity of damage done by these accidents also vary. Some of the reasons can be the direct fault of the driver such as no or poor driving skills, under age drivers, speeding, not following signs, signals and road instructions, tailgating, reckless driving, mobile usage and distraction; at other times there couldn't be any fault of driver, like weather conditions, car system failure, poor road conditions, unavailability of street lights at night etc. Road traffic crashes cost most countries 3% of their gross domestic product[1]. Reducing road accidents is challenging task, governments and different departments like traffic and police try their best to reduce accidents and damage done by those accidents by improving road and traffic conditions, installing signals and signals, running campaigns for road awareness and reminding people about consequences of speeding, rash/drunk driving, using mobile phones while driving. Still a lot of drivers make mistakes again and again. Accidental insurance claims done after road accidents require a lot of time, analysis and study to reach correct and better results.

The dataset contains comprehensive information about traffic collisions that have taken place on county and local roadways within Montgomery County. The main objective is to develop a predictive machine learning model capable of analyzing and determining whether the driver involved in the accident is at fault or not. The dataset encompasses traffic accident data spanning from 2015 to 2023, providing a wide range of information to build and train the predictive model.

1.2 - Objectives and Dataset

Data available about traffic accidents can be analyzed to understand different factors like severity of damage done, fatality of incident, profile and behavior of driver etc. It is always a tricky task to find who was actually at fault in vehicle collisions/accidents. There are different techniques suggested to find out who to blame for a car accident [8]. Our focus is to predict machine learning models which will analyze and determine whether the driver was at fault or not. This will help understand driver's behaviour in different situations, different time or year and day so that different techniques can be used to teach drivers about serious faults prior to happening and build a safer environment on the road. Insurance companies can also get help from this prediction model to solve cases of accidental claims by drivers. Main objective of this project is to analyze all the available factors in the accidents dataset like time of accidents, weather conditions, lighting conditions, speed, influence on driver etc. and predict the fault of the driver in incidents.

The data set used in this research is "Crash Reporting - Drivers Dataset" obtained from data gov [2] and original data is provided by Montgomery county gov on their website [3]. The data is collected through the Automated Crash Reporting System (ACRS) of the Maryland State Police. The ACRS system is designed to capture and record relevant data related to traffic accidents, providing a comprehensive dataset for analysis and further study. Dataset contains 43 columns and 20,993 rows, from which we use 12 feature variables and 1 target variable. Target value is "Driver Fault" where Yes is "1" and No is "0".

2 - Literature review

There is a great deal of research out there that studies different factors related to accidents and effects and damage done by those accidents. Majority of studies are related to analysis and prediction of severity of accidents. Different machine learning modeling techniques can be used to propose predictive road accident models to increase road safety. Jonathan assessed different human factors that influence the severity of crash outcomes [4]. Age of the driver has an effect on his/her driving behavior, young drivers take more risks while middle aged drivers with substance abuse frequently lead to accidents.

Accidents reported involve different factors some of them are directly related to drivers experience and skill and some are its behaviour at that time. Police officers also try to investigate these things at the place of incident. Sometimes young drivers over-speeding for thrill and have peer pressure when driving in groups or anger and impulsivity contribute to accidents [5].

There are multiple studies done on severity of accidents and different factors affect severity. (Saeid et al.) Used classification models to predict injury severity of accidents, they analyzed different external factors such as light conditions, weather conditions etc. More serious accidents happen in absence of street lights compared to if street lights are present on the road. In the same way, the severity of accidents is far more if the weather is foggy [6]. On other hand accidents occur due to irresponsible behaviour of drivers, (Daniel J. et al. 2018) did a cohort study, in which they told drivers who are habitual of sleeping less at night have higher crash risk. Sleep deficiency or insufficient sleep duration is strongly associated with motor vehicle crashes [7]. Substance abuse or drunk driving is also among common cases where drivers show irresponsibility and put his and other people's life at risk, (Swapnil K. 2020) in this paper analyzed US accidents and found Texas has the highest number of accidents caused by alcoholic drivers and California is at second with not a big difference[9].

Car accidents can be attributed to two main factors: traffic conditions and human error. Human errors encompass a wide range of behaviors, including incorrect decision-making, drowsiness, tailgating, consumption of alcohol and drugs, mobile phone usage, and more. Research by (Gicquel, Ordonneau, Blot, Toillon, Ingrand, and Romo, 2017) highlights the significant role of human error in car accidents. According to a study by (Abu Jadayil, Khraisat, and Shakoor, 2020), human error is identified as the primary cause of accidents, accounting for 96.8% of 97,981 reported accidents. This emphasizes the importance of addressing human behaviors and actions to prevent accidents. Over-speeding is another major contributing factor to accidents, as highlighted by (Fan, 2015). Fan suggests that accidents occur when drivers are unable to avoid them due to limited time and challenging decision-making situations. Furthermore, (Waylen and McKenna, 2008) state that driving under the influence of drugs or alcohol significantly increases the risk of car accidents. Such substances can impair drivers' concentration, reflexes, and awareness levels, leading to a higher likelihood of accidents occurring.

The study conducted by Saifuzzaman, Haque, Zheng, and Washington (2015) reveals that young drivers are highly prone to distraction from mobile phones, which negatively impacts their driving performance and increases the risk of car crashes. When drivers use their phones, they often fail to pay attention to the vehicle in front of them, especially when it slows down. Additionally, drivers may neglect road signs, such as traffic signals or stop signs. These distractions can significantly impair a driver's ability to respond effectively to changing road conditions and lead to accidents. Accidents can also occur due to various external factors, including adverse weather conditions, poor road conditions, or mechanical failures. Fan (2015) suggests that factors like brake failure, steering system failure, car light failure, or tire bursts can contribute to accidents, although the likelihood of such occurrences is relatively rare. Furthermore, Chen, Zhao, Liu, Ren, and Liu (2019) argue that weather and road conditions

have a significant impact on driver behavior and can put drivers in critical positions that may lead to accidents. Conditions such as snow and dense fog can reduce visibility, causing drivers to encounter difficulties in maneuvering their vehicles safely.

3 - Machine Learning pipeline

3.1 - Exploratory Data Analysis

After extracting the file from the data source, the dataset underwent pre-processing to remove redundant information, ensuring data consistency and quality before conducting further analysis. The pre-processing steps involved the following:

In [2]:
```
# Read dataset into a dataframe. And start exploring the dataset
df = pd.read_csv('US_Crash_Drivers_Data.csv')
# print the dataframe
display(df.head())
# print the number of rows and columns.
display(df.shape)
```

	Report Number	Local Case Number	Agency Name	ACRS Report Type	Crash Date	Route Type	Road Name	Cross-Street Type	Cross-Street Name	Off-Road Description	...	Speed Limit
0	DD5641000P	200016622	Rockville Police Departme	Property Damage Crash	4/21/2020 6:45	Municipality	WOOTTON PKWY	Municipality	TOWER OAKS BLVD	NaN	...	40
1	MCP1119008K	200013792	Montgomery County Police	Property Damage Crash	3/24/2020 7:13	County	CASHELL RD	County	MUSIC GROVE CT	NaN	...	35
2	MCP1477000G	190023261	Montgomery County Police	Property Damage Crash	5/17/2019 5:40	US (State)	COLUMBIA PIKE	County	FAIRLAND RD	NaN	...	50
3	MCP2699001H	200016424	Montgomery County Police	Property Damage Crash	4/19/2020 4:55	Maryland (State)	VEIRS MILL RD	County	TWINBROOK PKWY	NaN	...	45
4	MCP1182002G	190025740	Montgomery County Police	Property Damage Crash	5/30/2019 10:28	County	SEVEN LOCKS RD	Maryland (State)	BRADLEY BLVD	NaN	...	35

5 rows × 43 columns

```
(20933, 43)
```

Data Cleaning

Data cleaning is an essential step in the data preprocessing phase. It involves identifying and handling missing values, dealing with outliers, addressing inconsistent or incorrect data, and preparing the data for further analysis.Use the isnull() or isna() function to identify missing values in your dataset. Drop missing values: Use the dropna() function to remove rows or columns with missing values.

In [3]:
```
#calculates the total number of missing values (NaN or null values)
#in each column and then sums up the results
df.isna().any().sum()
#Removes missing value row
df.dropna
df.shape
```

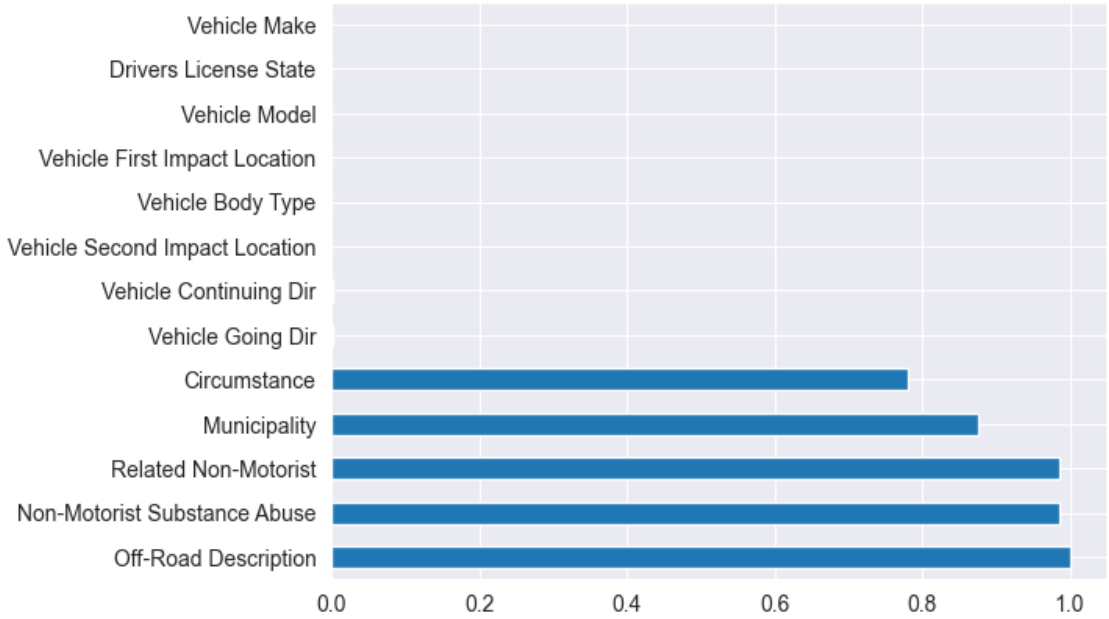
Out[3]:
```
(20933, 43)
```

In [4]:
```
missing_percentages = df.isna().sum().sort_values(ascending = False)/len(df)
```

Missing values in various columns can introduce bias or skewness to their distributions, making the data analysis more challenging. It is advisable to address and filter out these missing values at the initial stages of data processing. By doing so, we ensure that the dataset is cleaner and more suitable for analysis, improving the accuracy and reliability of the subsequent data analysis tasks.

In [5]:
```
missing_percentages[missing_percentages != 0].plot(kind = 'barh')
```

Out[5]:
```
<Axes: >
```



We can observe that several columns have empty values. Off-Road Description, Non-Motorist Substance Abuse, and Related Non-Motorist have approximately 98% missing data. Additionally, Circumstance has around 78% missing data, and Municipality has around 84% missing data. Other columns may have a few unknown values. It would be advisable to exclude these columns from further analysis since they have missing data and will cause issue in model accuracy.

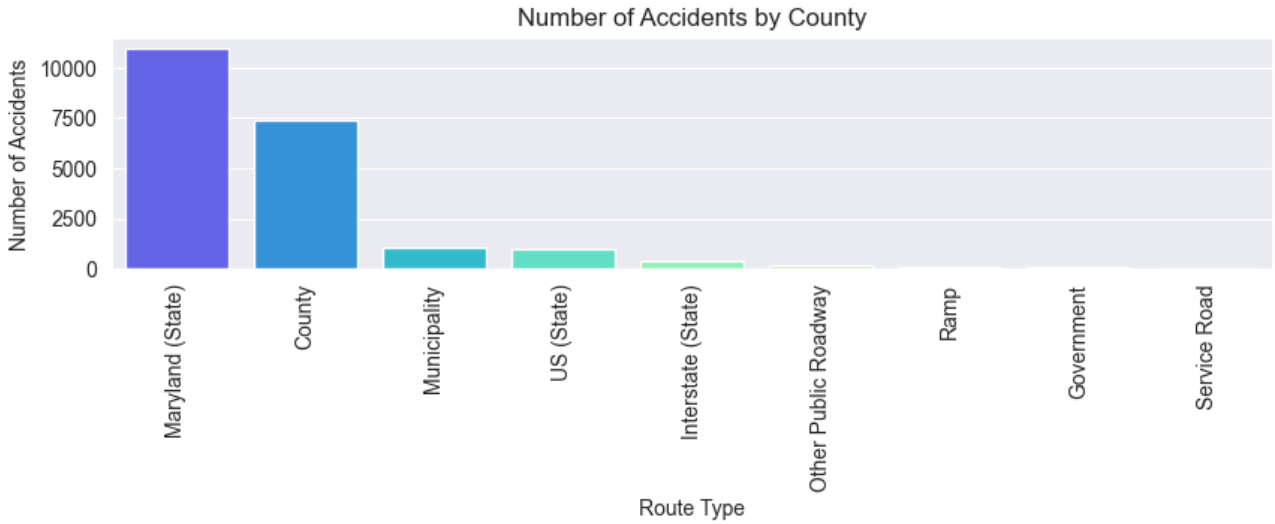
Visualizations

In [6]:
```
#Target Variable
df['Driver At Fault'].value_counts()
```

Out[6]:
```
Driver At Fault
Yes    12390
No      8543
Name: count, dtype: int64
```

Observation: Number of Drivers at fault are more than the one with no apparent fault.

In [7]:
```
## Observing which county have most accidents
route_type_counts = pd.DataFrame(df['Route Type'].value_counts())
z = route_type_counts.values.flatten()
x = route_type_counts.index.to_list()
fig,axs = plt.subplots(figsize = (10,2))
x = route_type_counts[0:15].index.to_list()
y = route_type_counts[0:15].values.flatten()
sns.barplot(x=x, y = y, palette='rainbow')
axs.tick_params(axis = 'x', rotation = 90)
axs.set_ylabel("Number of Accidents")
axs.set_xlabel("Route Type")
plt.title("Number of Accidents by County")
plt.savefig("County_Accidents.png",bbox_inches = 'tight', dpi = 300)
plt.show()
```

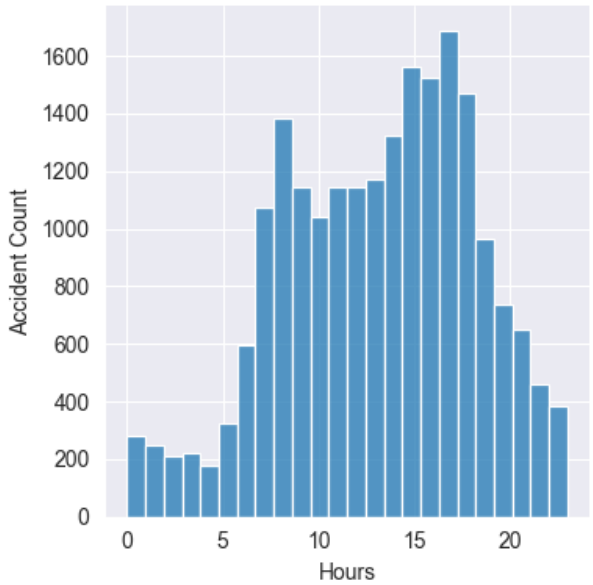


Observation: Maryland (State) and County (Route Type) have a higher number of accidents compared to other roads. This raises the question of why these specific roads have more accidents than others. Potential factors contributing to this could include traffic control issues, poor road conditions, speed limit enforcement problems, or inadequate lighting during nighttime.

In [8]:
```
##Converting to datetime stamp for further date/time analysis
df['Crash Date'] = pd.to_datetime(df['Crash Date'], errors='coerce')
```

In [9]:
```
#Distribution of percentage of accidents in 24hrs
ax=sns.displot(df['Crash Date'].dt.hour, bins = 24, kde=False, height=4)
ax.set(xlabel='Hours', ylabel='Accident Count')
```

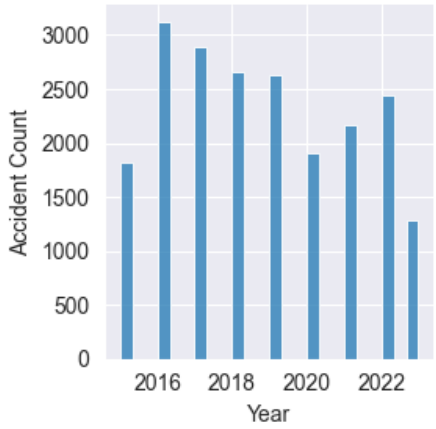
Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x13947585760>
```



The data reveals two distinct peaks in time: one during the morning hours between 7 am and 9 am, and another between 2 pm and 6 pm. This pattern aligns with the common understanding that these periods correspond to rush hours, where increased traffic congestion and higher volumes of vehicles on the road could potentially contribute to a higher likelihood of accidents occurring.

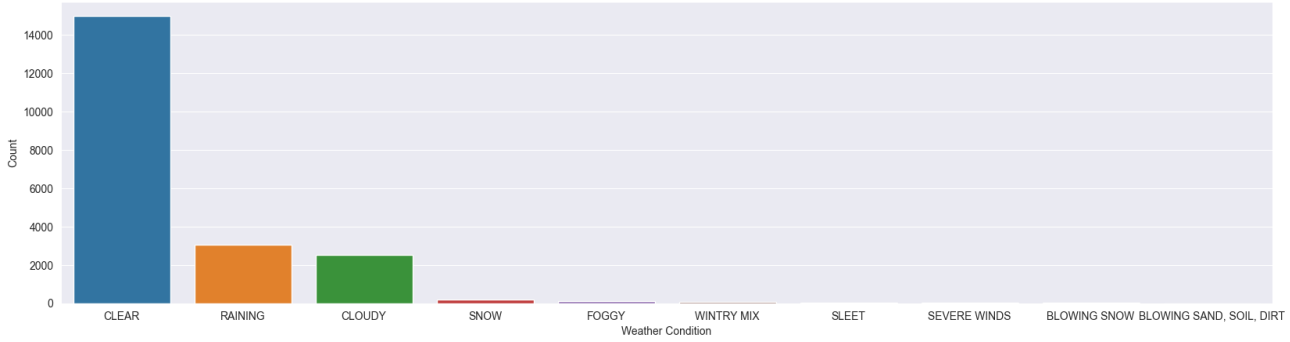
In [10]:
```
#Distribution of percentage of accidents from 2015 onwards
ax=sns.displot(df['Crash Date'].dt.year, bins = 24, kde=False, height=3)
ax.set(xlabel='Year', ylabel='Accident Count')
```

Out[10]:
```
<seaborn.axisgrid.FacetGrid at 0x1390c173fd0>
```



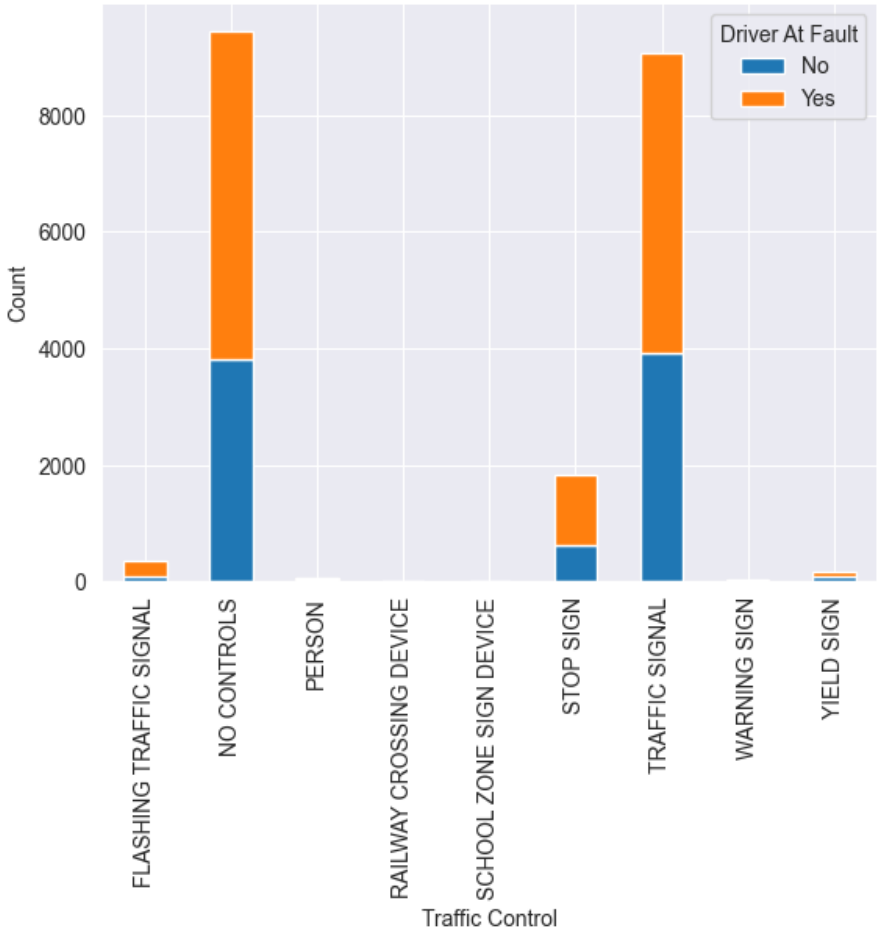
We can observe that 2016 was the peak year for accidents. However, starting from 2019, there has been a slight reduction in accidents. It is important to note that since we do not have the complete dataset for 2023, we cannot accurately predict whether there will be an increase in future or not.

In [11]:
```
#Weather Distribution by accident count
counts = df["Weather"].value_counts()
plt.figure(figsize=(20, 5))
sns.barplot(x = counts.index,y= counts.values)
plt.xlabel("Weather Condition")
plt.ylabel("Count")
plt.savefig("Weather_Accident_Distribution.png",bbox_inches = 'tight', dpi = 300)
plt.show()
```



We can observe that the majority of accidents happened on clear days. Now, the question arises as to how a majority of the recorded accidents occurred under the clear weather condition? However, it is also evident that accidents occurred during rainy and cloudy days.

In [12]:
```
#Observing Driver fault by Traffic Control
grouped_df = df.groupby([df['Traffic Control'],
df['Driver At Fault']]).size().unstack()
# Plot the stacked bar chart
stacked_plot = grouped_df.plot(kind='bar', stacked=True)
# Add labels and title
plt.xlabel('Traffic Control')
plt.ylabel('Count')
# Show the plot
plt.show()
```



As we can observe, even when there is a traffic signal or a stop sign present, the percentage of driver faults is higher compared to cases where the driver is not at fault. There could be two possible reasons for this: either the driver was distracted or under the influence of alcohol. However, weather conditions could also be a contributing factor but we have seen earlier mostly weather is clear.

3.2 - Feature Engineering

Correlation analysis

One method of feature selection is correlation analysis, which examines the relationships between variables. the goal of this step is to enhance the accuracy and performance of the model by eliminating redundant features that may introduce bias or noise.

Correlation coefficients range from -1 to 1, with a perfect positive correlation indicated when a variable is compared with itself. In the pipeline, the variable "Target" exhibits a perfect positive correlation with itself. Based on the correlation matrix, variables with the highest correlation to the target variable are selected, indicating a strong linear relationship. These variables possess the ability to explain significant changes in the prediction when there are changes in the predictor feature. In our correlation analysis, we can see Surface Condition and Weather has 0.7 coefficient.This means that as one variable increases, the other variable tends to increase as well, and vice versa. The magnitude of 0.7 indicates a relatively high degree of association between the variables.

Data Features

From the above columns, we will take only take 13 variables for our data analysis and machine learning pipeline. Our Target variable is "Driver at fault" and rest are feature variables. Following are the columns which we will use for further analysis and to predict our target variable:

1. ACRS Report Type: This column can be a crucial feature for predicting fault in car crashes and assessing the severity of the incident.
2. Driver Substance Abuse: Information on whether the driver was under the influence of substances can indicate impaired driving behavior.

3. Traffic Control: This column indicates the type of traffic control present at the crash location (e.g., stop sign, traffic signal), which can help determine if any traffic violations occurred.
4. Crash Date/Time: The timestamp of the crash can provide temporal information that may be relevant to understanding the circumstances surrounding the event.
5. Route Type: The specific road where the crash occurred can help identify potential road design or infrastructure issues.
6. Collision Type: It can help in understanding the dynamics of the crash and determining fault.
7. Surface Condition: The condition of the road surface (e.g., dry, wet, icy) can impact vehicle handling and braking performance.
8. Driver Distracted By: This column captures any distractions that might have diverted the driver's attention from the road, potentially contributing to the crash.
9. Light: The lighting conditions (e.g., daylight, darkness, streetlights) may play a role in visibility and driver perception.
10. Weather: Weather conditions at the time of the crash could influence driving conditions and potentially contribute to the accident.
11. Vehicle Movement: It can help identify specific maneuvers or actions performed by the driver leading up to the crash.
12. Speed Limit: Assess whether the driver was driving within the legally defined speed limit at the crash location.

Feature selection involves choosing a subset of relevant features from a dataset. The variable features is a list containing the names of the features to be used to predict the model.

In [13]:
```
# Categorical features to encode using OrdinalEncoder
features=['Driver At Fault','ACRS Report Type','Route Type','Collision Type',
'Weather','Surface Condition','Light','Traffic Control','Driver Substance Abuse',
'Driver Distracted By','Vehicle Movement','Speed Limit','Crash Date']
# Columns assigned to the variable X.
X = df[features]
```

To work with categorical features, the code uses the OrdinalEncoder from scikit-learn to transform the categorical columns in X into numerical values. It creates a new DataFrame called df_encoded with the transformed values.

In [14]:
```
encoder = OrdinalEncoder().set_output(transform="pandas")
df_encoded = encoder.fit_transform(X)
```

The target variable y is assigned with the column 'Driver At Fault' from df_encoded. The feature variables X are assigned with all the columns from df_encoded except for 'Driver At Fault'.

In [15]:
```
#Target Variable
y=df_encoded['Driver At Fault']
#Feature Variables
X=df_encoded.drop('Driver At Fault', axis=1)
```

The code prepares the data by selecting the desired features, encoding categorical variables, and separating the target variable from the feature variables. This processed data can then be used for training a machine learning model.

Scaling

Scaling the features can be beneficial for many machine learning algorithms as it helps in handling features with different scales and avoids any particular feature dominating the learning process. However, we only apply scaling to the input features and not the target variable.

In [16]:
```
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
```

Split Test and Train

The dataset is divided into training and testing datasets using the train_test_split() function. In this prediction task, a test size of 20% of the dataset is utilized for evaluating the performance of the model.

In [17]:
```
X_train, X_test, y_train, y_test = train_test_split(X_std, y, test_size=0.2
                                                    , random_state=42)
```

In [18]:
```
#Lazy Predict Classifier run all models with accuracy score
clf = LazyClassifier(verbose=0,ignore_warnings=True, custom_metric=None)
models,predictions = clf.fit(X_train, X_test, y_train, y_test)
models
```

```
100%|██████████████████████████████████████████████████████████████████████████████████████████████████████████| 29/29 [01:15<00:00,  2.62s/it]
```

Out[18]:

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
LGBMClassifier	0.85	0.85	0.85	0.85	0.32
XGBClassifier	0.84	0.84	0.84	0.84	1.14
AdaBoostClassifier	0.81	0.82	0.82	0.81	0.64
BaggingClassifier	0.82	0.81	0.81	0.82	0.56
RandomForestClassifier	0.82	0.81	0.81	0.82	2.19
ExtraTreesClassifier	0.81	0.79	0.79	0.80	1.84
DecisionTreeClassifier	0.79	0.78	0.78	0.79	0.07
SVC	0.77	0.76	0.76	0.77	11.61
NuSVC	0.77	0.76	0.76	0.77	17.35
KNeighborsClassifier	0.76	0.75	0.75	0.76	0.77
QuadraticDiscriminantAnalysis	0.70	0.74	0.74	0.69	0.06
GaussianNB	0.70	0.74	0.74	0.69	0.04
ExtraTreeClassifier	0.75	0.74	0.74	0.75	0.03
LabelSpreading	0.74	0.73	0.73	0.74	21.36
LabelPropagation	0.74	0.73	0.73	0.74	11.40
SGDClassifier	0.73	0.72	0.72	0.73	0.09
LogisticRegression	0.74	0.72	0.72	0.74	0.08
LinearSVC	0.74	0.71	0.71	0.73	1.28
NearestCentroid	0.72	0.71	0.71	0.72	0.36
CalibratedClassifierCV	0.73	0.71	0.71	0.73	4.18
BernoulliNB	0.73	0.71	0.71	0.72	0.04
LinearDiscriminantAnalysis	0.72	0.70	0.70	0.71	0.11
RidgeClassifier	0.72	0.70	0.70	0.71	0.05
RidgeClassifierCV	0.72	0.70	0.70	0.71	0.05
PassiveAggressiveClassifier	0.63	0.65	0.65	0.64	0.07
Perceptron	0.65	0.64	0.64	0.65	0.05
DummyClassifier	0.59	0.50	0.50	0.44	0.04

LazyPredict allows us to compare multiple classifiers with minimal code. It automates the process of training and testing multiple models, saving time and effort in manually setting up each classifier individually.

Based on the results obtained, we can observe that the LGBM Classifier and XGB Classifier achieved i.e. 85% and 84% the highest accuracy scores among the evaluated models. For further analysis, we will select the LGBM Classifier as it demonstrated the highest accuracy. Conversely, the dummy classifier attained the lowest accuracy score compared to the other models.

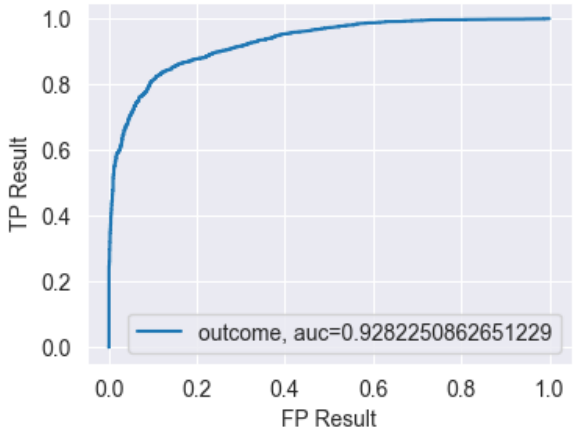
In [19]:
```
#LGBM Classifier
clf = lgb.LGBMClassifier()
clf.fit(X_train, y_train)
y_pred=clf.predict(X_test)
```

Now we have an LGBM Classifier object that can be utilized to predict whether drivers are primarily at fault in the reported accidents within US counties. By using the Fit method, the classifier is trained using the independent variables (x - 'Features') and

the dependent variable (y - 'Driver at Fault'). This process establishes a relationship between the driver's fault and the number of reported accidents within the classification object.

The AUC (Area Under the Curve) is employed to visualize the performance of the binary classifier pair. It demonstrates the distinction between the True Positive and False Positive rates at various classification thresholds. With a trade-off of 0.85 between the False Positive and True Positive rates, the model exhibits excellent discriminatory ability. The curve is positioned close to the top-left section of the graph, indicating the model's capacity to effectively distinguish between the two classes.

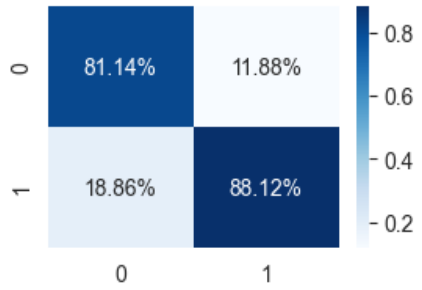
In [20]:
```
# The AUC curve
y_pred_auc = clf.predict_proba(X_test)[::,1]
FPR, TPR, _ = metrics.roc_curve(y_test, y_pred_auc)
roc = metrics.roc_auc_score(y_test, y_pred_auc)
#size of plot
f = plt.figure()
f.set_figwidth(4)
f.set_figheight(3)
#plot result
plt.plot(FPR,TPR,label="outcome, auc="+str(roc))
plt.legend(loc=4)
plt.xlabel('FP Result')
plt.ylabel('TP Result')
plt.show()
```



Based on the AUC curve presented, the model demonstrates a high level of sophistication. With an AUC value of 0.928, it indicates that the model is capable of making predictions for "Driver fault" with near-perfect accuracy.

In [21]:
```
#Confusion Matrix
fig, ax = plt.subplots(figsize=(3, 2))
matrix_cnf_m = metrics.confusion_matrix(y_test, y_pred)
sns.heatmap(matrix_cnf_m/sum(matrix_cnf_m), annot=True,
fmt='.2%', cmap='Blues',ax=ax)
```

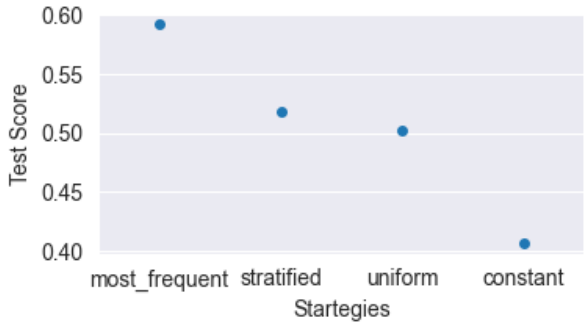
Out[21]: <Axes: >



The obtained matrix shows the following, 81.14% true negative predictions "Driver not at fault" predicted as "Driver not at fault", 18.86% false negative predictions; therefore 18.86% was wrongly predicted as 'Driver not at fault' instead of "Driver at fault". 11.88% false positive prediction,that is "not fault" when there is "Driver fault" and 88.12% true positive "Driver fault" when it was "Driver fault".

Dummy Classifier baseline technique

In [22]:
```
## It serves as a simple baseline to compare against other more complex classifiers.
# Maximum test score of the baseline classifier comes out be: 0.593
strategies = ['most_frequent', 'stratified', 'uniform', 'constant']
test_scores = []
for s in strategies:
    if s =='constant':
        dclf = DummyClassifier(strategy = s, random_state = 0, constant =0)
    else:
        dclf = DummyClassifier(strategy = s, random_state = 0)
    dclf.fit(X_train, y_train)
    score = dclf.score(X_train, y_train)
    test_scores.append(score)
y = test_scores
x = strategies
rounded = []
for value in y:
    rounded.append(round(value, 3))
# Create a strip plot
f = plt.figure()
f.set_figwidth(4)
f.set_figheight(2)
sns.stripplot(x=x, y=rounded)
# Set axis labels and title
plt.xlabel('Startegies')
plt.ylabel('Test Score')
# Show the plot
plt.show()
print("Maximum test score of the baseline classifier: ", max(rounded))
```




```
Maximum test score of the baseline classifier:  0.593
```

The baseline technique used in this analysis is the dummy classifier, which aims to predict whether a driver is at fault or not at fault during accidents. The accuracy score of 0.59 indicates that the pipeline correctly predicted the class for 59% of the instances in the dataset.

However, this accuracy score suggests that the future predictions made by the pipeline would be inaccurate. This can have several negative impacts, including the following:

Undeliberate Bias: Biased models can result in unfair determination of fault, potentially leading to unjust treatment or incorrect assignment of responsibility in accidents. This could harm the organization's reputation and may result in legal implications.

Misclassification: Misclassifying drivers as not at fault when they are actually at fault, or vice versa, can lead to incorrect assessments of accident causes and contribute to incorrect determination of liability. This may result in poor decision-making and potential legal disputes.

Lack of Transparency: Failure to disclose the poor accuracy of the predictive model may lead to mistrust from stakeholders, including insurance providers, law enforcement agencies, and affected individuals. This lack of transparency can damage the organization's reputation and hinder cooperation.

Constant Updates and Maintenance: The pipeline would require regular reviews and updates to improve its accuracy and maintain relevance. This ongoing effort would necessitate additional resources in terms of time and cost.

In this case, the accuracy score of 0.59 suggests that the pipeline has limited ability to accurately distinguish between drivers at fault and drivers not at fault during accidents. Therefore, using more advanced modeling techniques, such as LGBM Classifier or random forest, would likely result in more accurate predictions.

Deep Learning Modeling (Neural Network)

The Dense layer represents a fully connected layer in the neural network. The 22 parameter specifies the number of units (neurons) in the layer, and input_dim=X_train.shape[1] sets the input shape of the layer based on the number of features in the training data. The activation='sigmoid' sets the activation function for this layer to the sigmoid function.

In [23]:
```
# Create a sequential model
model = Sequential()
# Add hidden layers
model.add(Dense(22, input_dim=X_train.shape[1], activation='sigmoid'))
model.add(Dense(10, activation='tanh'))
model.add(Dense(10, activation='relu'))
model.add(Dense(10, activation='softmax'))
# Print the model summary
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 22)                286

 dense_1 (Dense)             (None, 10)                230

 dense_2 (Dense)             (None, 10)                110

 dense_3 (Dense)             (None, 10)                110

=================================================================
Total params: 736
Trainable params: 736
Non-trainable params: 0
_________________________________________________________________
```

The output summary provides a concise overview of the model architecture, the output shapes of each layer, and the number of parameters in the model, allowing us to understand the structure and complexity of the neural network.

Now the dataset is split into train and test size of 20%. Considering 5 epochs and 32 batch size, following result is obtained: Please note that Deep Learning requires high-end machines contrary to traditional Machine Learning algorithms. Due to the limitations of local system, the process is slow otherwise with the growing trend of high performing processors and GPU training the model would have been faster and efficient.

In [24]:
```
# Compile the model with sparse_categorical_crossentropy loss
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=5, batch_size=32, validation_data=(X_test, y_test))
y_predicted=model.predict(X_test)
```

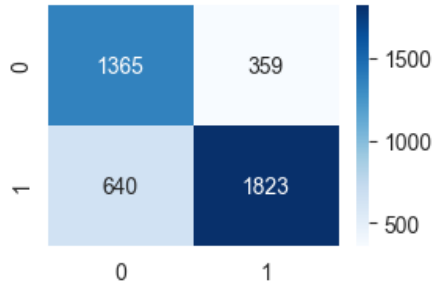
```
Epoch 1/5
524/524 [==============================] - 2s 2ms/step - loss: 0.7875 - accuracy: 0.6430 - val_loss: 0.5292 - val_a
ccuracy: 0.7390
Epoch 2/5
524/524 [==============================] - 1s 2ms/step - loss: 0.4717 - accuracy: 0.7593 - val_loss: 0.4796 - val_a
ccuracy: 0.7444
Epoch 3/5
524/524 [==============================] - 1s 2ms/step - loss: 0.4524 - accuracy: 0.7659 - val_loss: 0.4567 - val_a
ccuracy: 0.7573
Epoch 4/5
524/524 [==============================] - 1s 1ms/step - loss: 0.4372 - accuracy: 0.7757 - val_loss: 0.4499 - val_a
ccuracy: 0.7621
Epoch 5/5
524/524 [==============================] - 1s 1ms/step - loss: 0.4319 - accuracy: 0.7782 - val_loss: 0.4498 - val_a
ccuracy: 0.7614
131/131 [==============================] - 0s 914us/step
```

In [25]:
```
for i in range(2):
    y_predicted[i]
    np.argmax(y_predicted[i])
```

In [26]:
```
y_predicted_labels = [np.argmax(i) for i in y_predicted]
```

In [27]:
```
#Confusion Matrix heatmap
fig, ax = plt.subplots(figsize=(3, 2))
cm = tf.math.confusion_matrix(labels=y_test, predictions=y_predicted_labels)
sns.heatmap(cm, annot=True,
fmt='0', cmap='Blues',ax=ax)
```

Out[27]: <Axes: >



In [28]:
```
#Classification Report
print(classification_report(y_test, y_predicted_labels))
```

```
              precision    recall  f1-score   support

         0.0       0.68      0.79      0.73      1724
         1.0       0.84      0.74      0.78      2463

    accuracy                           0.76      4187
   macro avg       0.76      0.77      0.76      4187
weighted avg       0.77      0.76      0.76      4187
```

Based on the above results, the f1 score obtained for the model is 76%, which is lower than that of the light_GBM model, which achieved an f1 score of 85%. The f1 score is a measure of the model's overall accuracy, taking into account both precision and recall.

The recall value, which indicates the ability of the classifier model to accurately predict the occurrence of driver fault in accidents, is 0.79 and 0.74 in the given scenario. A higher recall score suggests that the results of the search were more relevant. Conversely, a lower recall score suggests that the classifier has a larger number of false negatives, meaning it is failing to identify instances of driver fault correctly.

This lower recall score can be attributed to various factors, such as untuned model hyperparameters or an imbalanced class distribution in the dataset. In real-life datasets, class imbalances are often observed, where one class has significantly more instances than the other. Addressing these issues through hyperparameter tuning and handling class imbalances can potentially improve the recall score and overall performance of the classifier model.

Conclusion & Insights

In conclusion, determining driver's fault after a car accident is a complex and multifaceted process that requires careful investigation, analysis, and consideration of various factors. While it is tempting to assign blame immediately based on initial impressions or assumptions, it is essential to approach such assessments with objectivity and reliance on factual evidence. This project is aimed to generate a prediction model to understand the fault of the driver in any accident with analysis of determinants provided in the dataset. Dataset has records of accidents in Maryland state, US. We explored different machine learning based modeling techniques but continued with LGBM classification models. The highest accuracy is achieved by light-GBM models with an accuracy of up to 85%. We dropped unnecessary columns that were not needed for the prediction model from this dataset, we also performed some data cleanup to overcome missing data and preprocessing to convert some of the variables to meaningful format. Analysis in the project showed insights about different factors contributing, either driver related or external factors. Many comparisons based on these factors show interesting insights, like variation in accidents based on time, weather, light condition,

also injury severity of accidents and state of driver when accident happened. Insurance companies can also get meaningful insights from results of this prediction model to be used along their investigation and analysis to solve claims made by either parties after an accident. Finally analyzing factors which make drivers make mistakes can be pointed out and better campaigns and policies can be created around them.

Utilizing the LGBM Classifier for predicting driver fault represents a significant improvement over the baseline approach of employing a dummy classifier. Unlike the dummy classifier, which simply predicts the most common class without considering the relationship between the features and the target variable, the LGBM Classifier takes into account the underlying relationship between the features and the target variable. This distinction contributes to improved performance in terms of recall, precision, and accuracy compared to the dummy classifier. Furthermore, the LGBM Classifier demonstrates superior performance when compared to the deep learning model. The LGBM Classifier outperforms the deep learning model in various evaluation metrics, indicating its effectiveness in capturing the patterns and associations within the data to make accurate predictions for driver fault.

References

1. WHO (World Health Organization), Road Traffic Injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries
2. Crash Reporting Dataset - Data gov https://catalog.data.gov/dataset/crash-reporting-drivers-data
3. Crash Reporting - Drivers Data - Data Montgomery Website https://data.montgomerycountymd.gov/Public-Safety/Crash-Reporting-Drivers-Data/mmzv-x632
4. Jonathan J. Rolison et al. (June 2018) What are the factors that contribute to road accidents? An assessment of law enforcement views, ordinary drivers' opinions, and road accident records [https://www.sciencedirect.com/science/article/pii/S0001457518300873]
5. Shaneel et al. The influence of anger, impulsivity, sensation seeking and driver attitudes on risky driving behaviour among post [https://www.sciencedirect.com/science/article/abs/pii/S0001457513000626]
6. Saeid et al. Road Car Accident Prediction Using a Machine-Learning-Enabled Data Analysis [https://www.mdpi.com/2221386]
7. Daniel J. et al. Sleep deficiency and motor vehicle crash risk in the general population: a prospective cohort study [https://rdcu.be/df28T]
8. The Tricky Business Of Determining Fault After A Car Accident [https://www.forbes.com/advisor/car-insurance/determining-fault-after-accident/]
9. Swapnil Kisan. Analysis of US accidents and solutions [https://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?article=2085&context=etd]
10. Jadayil, W. A., Khraisat, W., & Shakoor, M. (2020). Statistical analysis for the main fac- tors causing car accidents. ARPN Journal of Engineering and Applied Sciences, 15(5), 696–715. Retrieved from: http://www.arpnjournals.org/jeas/research_papers/rp_2020/jeas_0320_8150.pdf
11. Fan, F. (2018). Study on the Cause of Car Accidents at Intersections. Open Access Library Journal, 5: e4578. https://doi.org/10.4236/oalib.1104578
12. Chen, C., Zhao, X., Liu, H., Ren, G., & Liu, X. (2019). Influence of adverse weather on drivers' perceived risk during car following based on driving simulations. Journal of Modern Transportation, 27(4), 282–292. https://doi.org/10.1007/s40534-019-00197-4