Andrés Morones

Adrián Landaverde Nava

Cristian Gonzaga López

Joel Erick Martínez Espinosa

00. Introduction

In the R Shiny Dashboard made previously: (https://adrian-landaverde.shinyapps.io/CrystalCaseStudy/) we made the EDA. In which we could see how the energy from 2005 and 2006 is very different despite the variables are very alike.

So it will be attempted to predict the energy using 2 approaches:

In the fisrt one, it will be implemented one model for each year using the humidity and the temperature

In the second one, it will be used a time series analysis to predict the energy based on the trend and seasonality of the data

01. Libraries and Data

```
1 !pip install statsmodels
```

```
Requirement already satisfied: statsmodels in /usr/local/lib/python3.7/dist-packages (0
Requirement already satisfied: scipy>=0.18 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: patsy>=0.4.0 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from pats
```

```
1 import pandas as pd
2 import numpy as np
```

```
import datetime as dt
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.pyplot import figure
import seaborn as sns
from scipy import stats
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LinearRegression
from sklearn.neural_network import MLPRegressor
from sklearn.preprocessing import MinMaxScaler
import statistics
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.statespace.sarimax import SARIMAX
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning:
  import pandas.util.testing as tm
```

```
import warnings
warnings.filterwarnings("ignore")
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df_energy= pd.read_csv("/content/drive/MyDrive/Universidad/Cuarto Semestre/Crystal/energy_
df_energy
```

	Hour	energy_consumpt_2005	energy_consumpt_2006	full_temp_2005	full_humid_2005
0	1	631.623161	1246.300847	-0.400000	64.000000
1	2	534.397104	1062.500558	-0.733333	65.333333
2	3	453.538785	884.586887	-1.066667	66.666667
3	4	400.699718	786.564121	-1.400000	68.000000
4	5	378.171092	742.669614	-1.666667	60.333333

```
df_energy["Year"]=2005
df_energy["Day"]= (((df_energy["Hour"]-1)/24)+1).apply(np.floor)
df_energy["Week"]= (((df_energy["Day"]-1)/7)+1).apply(np.floor)
df_energy
```

	Hour	energy_consumpt_2005	energy_consumpt_2006	full_temp_2005	full_humid_2005
0	1	631.623161	1246.300847	-0.400000	64.000000
1	2	534.397104	1062.500558	-0.733333	65.333333
2	3	453.538785	884.586887	-1.066667	66.666667
3	4	400.699718	786.564121	-1.400000	68.000000
4	5	378.171092	742.669614	-1.666667	60.333333
...	...	...	...	...	..
8779	8780	950.369306	0.000000	3.333333	64.000000
8780	8781	880.138770	0.000000	2.666667	68.000000
8781	8782	792.754026	0.000000	2.000000	72.000000
8782	8783	740.446668	0.000000	1.333333	76.000000
8783	8784	706.176769	0.000000	0.666667	80.000000

8784 rows × 10 columns

```
df_energy["Date"]=(np.asarray(df_energy['Year'], dtype='datetime64[Y]')-1970)+(np.asarray(
df_energy["Month"]=pd.DatetimeIndex(df_energy['Date']).month
df_energy
```

	Hour	energy_consumpt_2005	energy_consumpt_2006	full_temp_2005	full_humid_2005
0	1	631.623161	1246.300847	-0.400000	64.000000
1	2	534.397104	1062.500558	-0.733333	65.333333
2	3	453.538785	884.586887	-1.066667	66.666667
3	4	400.699718	786.564121	-1.400000	68.000000
4	5	378.171092	742.669614	-1.666667	60.333333
...	...	...	...	...	...
8779	8780	950.369306	0.000000	3.333333	64.000000
8780	8781	880.138770	0.000000	2.666667	68.000000
8781	8782	792.754026	0.000000	2.000000	72.000000

```
df_energy.isna().sum()
```

```
Hour                    0
energy_consumpt_2005    34
energy_consumpt_2006    42
full_temp_2005          0
full_humid_2005         0
full_temp_2006          24
full_humid_2006         24
Year                    0
Day                     0
Week                    0
Date                    0
Month                   0
dtype: int64
```

```
df_energy["energy_consumpt_2005"].fillna(value=df_energy["energy_consumpt_2005"].mean(), i
df_energy["energy_consumpt_2006"].fillna(value=df_energy["energy_consumpt_2006"].mean(), i
df_energy=df_energy.iloc[:-24]
df_energy.isna().sum()
```

```
Hour                    0
energy_consumpt_2005    0
energy_consumpt_2006    0
full_temp_2005          0
full_humid_2005         0
full_temp_2006          0
full_humid_2006         0
```

```
Year                  0
Day                   0
Week                  0
Date                  0
Month                 0
dtype: int64
```

```
variables=["energy_consumpt_2005","energy_consumpt_2006","full_temp_2005","full_temp_2006'
```

02. Machine Learning Regression

Based on the dashboard plots, the energy is more correlated to the mean temperature per day, and to the humidity per hour

```
meanEnergy2005= df_energy.groupby("Day")['full_temp_2005'].mean()
meanEnergy2006= df_energy.groupby("Day")['full_temp_2006'].mean()
df_energy["temp_per_day_2005"]=(np.repeat(meanEnergy2005,24)).values
df_energy["temp_per_day_2006"]=(np.repeat(meanEnergy2006,24)).values
df_energy["HourOfDay"]=df_energy["Hour"]-((df_energy['Day']-1)*24)
df_energy
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
  This is separate from the ipykernel package so we can avoid doing imports until
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
  after removing the cwd from sys.path.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
  """
```

	Hour	energy_consumpt_2005	energy_consumpt_2006	full_temp_2005	full_humid_200
0	1	631.623161	1246.300847	-0.400000	64.000000
1	2	534.397104	1062.500558	-0.733333	65.333333

```
def plotRgresion(y,y_predict,ax,limites,ymin,ymax):
  ax.plot(df_energy["Hour"],y,label="y Real",color="blue")
  ax.plot(df_energy["Hour"],y_predict,label="y Predicted",color="red")
  if(limites):
    ax.set_ylim(ymin,ymax)
  return(ax)
```

```
X_2005= df_energy[['temp_per_day_2005','HourOfDay','full_humid_2005']]
y_2005= df_energy['energy_consumpt_2005']

X_train, X_test, y_train, y_test= train_test_split(X_2005,y_2005,random_state=21)
```

02.1 Random Forest

```
parameters= {"n_estimators":list(range(10,101,10))}
model = RandomForestRegressor(random_state=0)

grid = GridSearchCV(model, param_grid = parameters,scoring="r2",verbose=3)
grid.fit(X_train,y_train)
print('Grid best parameter (max. R2): ', grid.best_params_)
print('Grid best score (R2): ', grid.best_score_)
print("R2 Score on test:",r2_score(y_test, grid.predict(X_test)))
fig, ax= plt.subplots(figsize=(20,8))
ax= plotRgresion(y_2005,grid.predict(X_2005),ax,True,0,1000)
```

```
plt.legend()
plt.show()
print("Final R2 Score:",r2_score(y_2005,grid.predict(X_2005)))
```

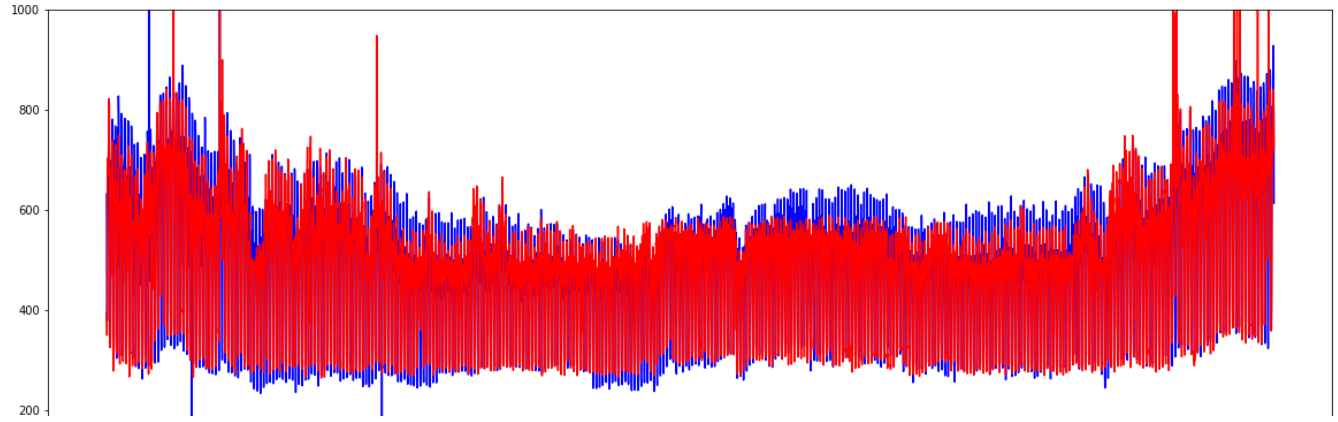
```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
[CV 1/5] END ..................n_estimators=10;, score=0.662 total time=   0.1s
[CV 2/5] END ..................n_estimators=10;, score=0.235 total time=   0.1s
[CV 3/5] END ..................n_estimators=10;, score=0.924 total time=   0.1s
[CV 4/5] END ..................n_estimators=10;, score=0.377 total time=   0.3s
[CV 5/5] END ..................n_estimators=10;, score=0.118 total time=   0.2s
[CV 1/5] END ..................n_estimators=20;, score=0.670 total time=   0.5s
[CV 2/5] END ..................n_estimators=20;, score=0.651 total time=   0.5s
[CV 3/5] END ..................n_estimators=20;, score=0.896 total time=   0.4s
[CV 4/5] END ..................n_estimators=20;, score=0.365 total time=   0.5s
[CV 5/5] END ..................n_estimators=20;, score=0.235 total time=   0.6s
[CV 1/5] END ..................n_estimators=30;, score=0.670 total time=   0.7s
[CV 2/5] END ..................n_estimators=30;, score=0.625 total time=   0.8s
[CV 3/5] END ..................n_estimators=30;, score=0.915 total time=   0.8s
[CV 4/5] END ..................n_estimators=30;, score=0.371 total time=   0.6s
[CV 5/5] END ..................n_estimators=30;, score=0.202 total time=   0.6s
[CV 1/5] END ..................n_estimators=40;, score=0.672 total time=   0.9s
[CV 2/5] END ..................n_estimators=40;, score=0.701 total time=   0.8s
[CV 3/5] END ..................n_estimators=40;, score=0.916 total time=   0.8s
[CV 4/5] END ..................n_estimators=40;, score=0.379 total time=   0.8s
[CV 5/5] END ..................n_estimators=40;, score=0.266 total time=   1.0s
[CV 1/5] END ..................n_estimators=50;, score=0.672 total time=   1.2s
[CV 2/5] END ..................n_estimators=50;, score=0.705 total time=   1.7s
[CV 3/5] END ..................n_estimators=50;, score=0.924 total time=   1.6s
[CV 4/5] END ..................n_estimators=50;, score=0.380 total time=   1.2s
[CV 5/5] END ..................n_estimators=50;, score=0.266 total time=   1.3s
```

02.2 K-Nearet Neighbours

```
[CV 4/5] END ..................n_estimators=60;, score=0.377 total time=   1.3s
```

```
parametrosKNN= {"n_neighbors":list(range(3,19,2))}
knn = KNeighborsRegressor()

grid = GridSearchCV(knn, param_grid = parametrosKNN,scoring="r2",verbose=3)
grid.fit(X_train,y_train)
print('Grid best parameter (max. R2): ', grid.best_params_)
print('Grid best score (R2): ', grid.best_score_)
print("R2 Score on test:",r2_score(y_test, grid.predict(X_test)))
fig, ax= plt.subplots(figsize=(20,8))
ax= plotRgresion(y_2005,grid.predict(X_2005),ax,True,0,1000)
plt.show()
print("Final R2 Score:",r2_score(y_2005,grid.predict(X_2005)))
```

```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
[CV 1/5] END .....................n_neighbors=3;, score=0.638 total time=   0.0s
[CV 2/5] END .....................n_neighbors=3;, score=0.515 total time=   0.0s
[CV 3/5] END .....................n_neighbors=3;, score=0.722 total time=   0.0s
[CV 4/5] END .....................n_neighbors=3;, score=0.322 total time=   0.0s
[CV 5/5] END .....................n_neighbors=3;, score=0.382 total time=   0.0s
[CV 1/5] END .....................n_neighbors=5;, score=0.639 total time=   0.0s
[CV 2/5] END .....................n_neighbors=5;, score=0.716 total time=   0.0s
[CV 3/5] END .....................n_neighbors=5;, score=0.821 total time=   0.0s
[CV 4/5] END .....................n_neighbors=5;, score=0.362 total time=   0.0s
[CV 5/5] END .....................n_neighbors=5;, score=0.337 total time=   0.0s
[CV 1/5] END .....................n_neighbors=7;, score=0.632 total time=   0.0s
[CV 2/5] END .....................n_neighbors=7;, score=0.721 total time=   0.0s
[CV 3/5] END .....................n_neighbors=7;, score=0.802 total time=   0.0s
[CV 4/5] END .....................n_neighbors=7;, score=0.370 total time=   0.0s
[CV 5/5] END .....................n_neighbors=7;, score=0.358 total time=   0.0s
[CV 1/5] END .....................n_neighbors=9;, score=0.616 total time=   0.0s
[CV 2/5] END .....................n_neighbors=9;, score=0.773 total time=   0.0s
[CV 3/5] END .....................n_neighbors=9;, score=0.832 total time=   0.0s
[CV 4/5] END .....................n_neighbors=9;, score=0.371 total time=   0.0s
[CV 5/5] END .....................n_neighbors=9;, score=0.360 total time=   0.0s
[CV 1/5] END ....................n_neighbors=11;, score=0.624 total time=   0.0s
[CV 2/5] END ....................n_neighbors=11;, score=0.779 total time=   0.0s
[CV 3/5] END ....................n_neighbors=11;, score=0.849 total time=   0.0s
[CV 4/5] END ....................n_neighbors=11;, score=0.372 total time=   0.0s
[CV 5/5] END ....................n_neighbors=11;, score=0.339 total time=   0.0s
[CV 1/5] END ....................n_neighbors=13;, score=0.627 total time=   0.0s
[CV 2/5] END ....................n_neighbors=13;, score=0.785 total time=   0.0s
[CV 3/5] END ....................n_neighbors=13;, score=0.844 total time=   0.0s
[CV 4/5] END ....................n_neighbors=13;, score=0.374 total time=   0.0s
[CV 5/5] END ....................n_neighbors=13;, score=0.337 total time=   0.0s
[CV 1/5] END ....................n_neighbors=15;, score=0.627 total time=   0.0s
[CV 2/5] END ....................n_neighbors=15;, score=0.790 total time=   0.0s
[CV 3/5] END ....................n_neighbors=15;, score=0.850 total time=   0.0s
[CV 4/5] END ....................n_neighbors=15;, score=0.374 total time=   0.0s
[CV 5/5] END ....................n_neighbors=15;, score=0.338 total time=   0.0s
[CV 1/5] END ....................n_neighbors=17;, score=0.625 total time=   0.0s
[CV 2/5] END ....................n_neighbors=17;, score=0.796 total time=   0.0s
[CV 3/5] END ....................n_neighbors=17;, score=0.843 total time=   0.0s
[CV 4/5] END ....................n_neighbors=17;, score=0.371 total time=   0.0s
[CV 5/5] END ....................n_neighbors=17;, score=0.340 total time=   0.0s
Grid best parameter (max. R2):  {'n_neighbors': 15}
Grid best score (R2):  0.5958735892803727
R2 Score on test: 0.8486469601628195
```



02.3 Linear Regresion

```
parametrosLR= {"fit_intercept":[True,False],"normalize":[True,False]}
linear = LinearRegression()

grid = GridSearchCV(linear, param_grid = parametrosLR,scoring="r2",verbose=3)
grid.fit(X_train,y_train)
print('Grid best parameter (max. R2): ', grid.best_params_)
print('Grid best score (R2): ', grid.best_score_)
print("R2 Score on test:",r2_score(y_test, grid.predict(X_test)))
fig, ax= plt.subplots(figsize=(20,8))
ax= plotRgresion(y_2005,grid.predict(X_2005),ax,True,0,1000)
plt.show()
print("Final R2 Score:",r2_score(y_2005,grid.predict(X_2005)))
```

```
Fitting 5 folds for each of 4 candidates, totalling 20 fits
[CV 1/5] END fit_intercept=True, normalize=True;, score=0.413 total time=   0.0s
[CV 2/5] END fit_intercept=True, normalize=True;, score=0.540 total time=   0.0s
[CV 3/5] END fit_intercept=True, normalize=True;, score=0.583 total time=   0.0s
[CV 4/5] END fit_intercept=True, normalize=True;, score=0.224 total time=   0.0s
[CV 5/5] END fit_intercept=True, normalize=True;, score=0.235 total time=   0.0s
[CV 1/5] END fit_intercept=True, normalize=False;, score=0.413 total time=   0.0s
[CV 2/5] END fit_intercept=True, normalize=False;, score=0.540 total time=   0.0s
[CV 3/5] END fit_intercept=True, normalize=False;, score=0.583 total time=   0.0s
[CV 4/5] END fit_intercept=True, normalize=False;, score=0.224 total time=   0.0s
[CV 5/5] END fit_intercept=True, normalize=False;, score=0.235 total time=   0.0s
[CV 1/5] END fit_intercept=False, normalize=True;, score=-0.037 total time=   0.0s
```

02.4 Neural Networks

```
[CV 5/5] END fit_intercept=False, normalize=True;, score=-0.040 total time=   0.0s
```

```
layers=[]
for i in range(1,11):
  for j in range(1,11):
    layers.append((i,j))
parametrosMLP= {"hidden_layer_sizes":layers}
mlp = MLPRegressor(max_iter=5000)

grid = GridSearchCV(mlp, param_grid = parametrosMLP,scoring="r2",verbose=3)
grid.fit(X_train,y_train)
print('Grid best parameter (max. R2): ', grid.best_params_)
print('Grid best score (R2): ', grid.best_score_)
print("R2 Score on test:",r2_score(y_test, grid.predict(X_test)))
fig, ax= plt.subplots(figsize=(20,8))
ax= plotRgresion(y_2005,grid.predict(X_2005),ax,True,0,1000)
plt.show()
print("Final R2 Score:",r2_score(y_2005,grid.predict(X_2005)))
```

```
Fitting 5 folds for each of 100 candidates, totalling 500 fits
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 1/5] END ........hidden_layer_sizes=(1, 1);, score=-5.158 total time=  52.0s
[CV 2/5] END ........hidden_layer_sizes=(1, 1);, score=-0.000 total time=  42.0s
[CV 3/5] END ........hidden_layer_sizes=(1, 1);, score=0.582 total time=   8.7s
[CV 4/5] END ........hidden_layer_sizes=(1, 1);, score=0.221 total time=  12.9s
[CV 5/5] END ........hidden_layer_sizes=(1, 1);, score=0.234 total time=  22.0s
[CV 1/5] END ........hidden_layer_sizes=(1, 2);, score=0.414 total time=   9.4s
[CV 2/5] END ........hidden_layer_sizes=(1, 2);, score=0.540 total time=   9.4s
[CV 3/5] END ........hidden_layer_sizes=(1, 2);, score=0.583 total time=  10.2s
[CV 4/5] END ........hidden_layer_sizes=(1, 2);, score=0.223 total time=   7.2s
[CV 5/5] END ........hidden_layer_sizes=(1, 2);, score=0.235 total time=   9.8s
[CV 1/5] END ........hidden_layer_sizes=(1, 3);, score=0.410 total time=   6.5s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 2/5] END ........hidden_layer_sizes=(1, 3);, score=-6.794 total time=  51.9s
[CV 3/5] END ........hidden_layer_sizes=(1, 3);, score=0.583 total time=   7.8s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 4/5] END ........hidden_layer_sizes=(1, 3);, score=-2.699 total time=  58.2s
[CV 5/5] END ........hidden_layer_sizes=(1, 3);, score=0.235 total time=   5.8s
[CV 1/5] END ........hidden_layer_sizes=(1, 4);, score=0.414 total time=  10.6s
[CV 2/5] END ........hidden_layer_sizes=(1, 4);, score=0.540 total time=   7.1s
[CV 3/5] END ........hidden_layer_sizes=(1, 4);, score=0.581 total time=   7.8s
[CV 4/5] END ........hidden_layer_sizes=(1, 4);, score=0.221 total time=   5.8s
[CV 5/5] END ........hidden_layer_sizes=(1, 4);, score=0.235 total time=   9.0s
[CV 1/5] END ........hidden_layer_sizes=(1, 5);, score=-0.000 total time=   5.0s
[CV 2/5] END ........hidden_layer_sizes=(1, 5);, score=0.540 total time=  29.0s
[CV 3/5] END ........hidden_layer_sizes=(1, 5);, score=0.580 total time=   7.9s
[CV 4/5] END ........hidden_layer_sizes=(1, 5);, score=0.221 total time=   5.5s
[CV 5/5] END ........hidden_layer_sizes=(1, 5);, score=0.235 total time=   7.0s
[CV 1/5] END ........hidden_layer_sizes=(1, 6);, score=0.424 total time=  10.3s
[CV 2/5] END ........hidden_layer_sizes=(1, 6);, score=0.540 total time=   9.1s
[CV 3/5] END ........hidden_layer_sizes=(1, 6);, score=0.587 total time=  10.0s
[CV 4/5] END ........hidden_layer_sizes=(1, 6);, score=-0.000 total time=   4.4s
[CV 5/5] END ........hidden_layer_sizes=(1, 6);, score=0.235 total time=   5.9s
[CV 1/5] END ........hidden_layer_sizes=(1, 7);, score=0.421 total time=   9.4s
[CV 2/5] END ........hidden_layer_sizes=(1, 7);, score=0.546 total time=   6.8s
[CV 3/5] END ........hidden_layer_sizes=(1, 7);, score=0.582 total time=   8.4s
[CV 4/5] END ........hidden_layer_sizes=(1, 7);, score=0.223 total time=   9.1s
[CV 5/5] END ........hidden_layer_sizes=(1, 7);, score=0.234 total time=   5.8s
[CV 1/5] END ........hidden_layer_sizes=(1, 8);, score=0.430 total time=   9.3s
[CV 2/5] END ........hidden_layer_sizes=(1, 8);, score=0.540 total time=   5.3s
[CV 3/5] END ........hidden_layer_sizes=(1, 8);, score=0.583 total time=   6.6s
[CV 4/5] END ........hidden_layer_sizes=(1, 8);, score=-0.000 total time=   3.7s
[CV 5/5] END ........hidden_layer_sizes=(1, 8);, score=0.235 total time=   7.0s
[CV 1/5] END ........hidden_layer_sizes=(1, 9);, score=0.412 total time=   6.1s
[CV 2/5] END ........hidden_layer_sizes=(1, 9);, score=0.540 total time=   7.2s
[CV 3/5] END ........hidden_layer_sizes=(1, 9);, score=0.581 total time=   6.5s
[CV 4/5] END ........hidden_layer_sizes=(1, 9);, score=0.222 total time=   7.8s
[CV 5/5] END ........hidden_layer_sizes=(1, 9);, score=-0.001 total time=   4.6s
[CV 1/5] END ........hidden_layer_sizes=(1, 10);, score=0.413 total time=   7.8s
[CV 2/5] END ........hidden_layer_sizes=(1, 10);, score=0.546 total time=  10.4s
[CV 3/5] END ........hidden_layer_sizes=(1, 10);, score=0.580 total time=   6.5s
[CV 4/5] END ........hidden_layer_sizes=(1, 10);, score=-0.000 total time=   4.8s
```

```
[CV 5/5] END .......hidden_layer_sizes=(1, 10);, score=-0.001 total time=   4.2s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 1/5] END ........hidden_layer_sizes=(2, 1);, score=-5.159 total time=  56.3s
[CV 2/5] END ........hidden_layer_sizes=(2, 1);, score=0.539 total time=   9.8s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 3/5] END ........hidden_layer_sizes=(2, 1);, score=-6.517 total time=  55.8s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 4/5] END ........hidden_layer_sizes=(2, 1);, score=-2.723 total time=  56.2s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 5/5] END ........hidden_layer_sizes=(2, 1);, score=-2.859 total time=  54.2s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 1/5] END ........hidden_layer_sizes=(2, 2);, score=-5.149 total time=  56.6s
[CV 2/5] END ........hidden_layer_sizes=(2, 2);, score=-0.000 total time=   8.7s
[CV 3/5] END ........hidden_layer_sizes=(2, 2);, score=0.581 total time=   7.4s
[CV 4/5] END ........hidden_layer_sizes=(2, 2);, score=0.223 total time=  10.8s
[CV 5/5] END ........hidden_layer_sizes=(2, 2);, score=0.234 total time=   6.4s
[CV 1/5] END ........hidden_layer_sizes=(2, 3);, score=0.480 total time=   5.7s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 2/5] END ........hidden_layer_sizes=(2, 3);, score=-6.754 total time= 1.1min
[CV 3/5] END ........hidden_layer_sizes=(2, 3);, score=0.635 total time=   6.8s
[CV 4/5] END ........hidden_layer_sizes=(2, 3);, score=0.223 total time=  28.4s
[CV 5/5] END ........hidden_layer_sizes=(2, 3);, score=0.237 total time=   7.6s
[CV 1/5] END ........hidden_layer_sizes=(2, 4);, score=0.414 total time=   7.2s
[CV 2/5] END ........hidden_layer_sizes=(2, 4);, score=0.539 total time=  10.0s
[CV 3/5] END ........hidden_layer_sizes=(2, 4);, score=0.676 total time=  10.3s
[CV 4/5] END ........hidden_layer_sizes=(2, 4);, score=0.230 total time=  13.8s
[CV 5/5] END ........hidden_layer_sizes=(2, 4);, score=0.277 total time=   5.1s
[CV 1/5] END ........hidden_layer_sizes=(2, 5);, score=0.537 total time=  13.3s
[CV 2/5] END ........hidden_layer_sizes=(2, 5);, score=0.540 total time=   7.6s
[CV 3/5] END ........hidden_layer_sizes=(2, 5);, score=0.583 total time=   9.0s
[CV 4/5] END ........hidden_layer_sizes=(2, 5);, score=0.232 total time=  10.9s
[CV 5/5] END ........hidden_layer_sizes=(2, 5);, score=0.236 total time=   6.2s
[CV 1/5] END ........hidden_layer_sizes=(2, 6);, score=0.476 total time=   5.4s
[CV 2/5] END ........hidden_layer_sizes=(2, 6);, score=0.580 total time=   5.3s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 3/5] END ........hidden_layer_sizes=(2, 6);, score=-6.489 total time= 1.1min
[CV 4/5] END ........hidden_layer_sizes=(2, 6);, score=0.220 total time=  12.7s
[CV 5/5] END ........hidden_layer_sizes=(2, 6);, score=0.237 total time=   6.9s
[CV 1/5] END ........hidden_layer_sizes=(2, 7);, score=0.412 total time=   7.8s
[CV 2/5] END ........hidden_layer_sizes=(2, 7);, score=0.540 total time=   8.6s
[CV 3/5] END ........hidden_layer_sizes=(2, 7);, score=0.584 total time=   7.2s
[CV 4/5] END ........hidden_layer_sizes=(2, 7);, score=0.222 total time=   5.2s
[CV 5/5] END ........hidden_layer_sizes=(2, 7);, score=0.236 total time=   7.0s
[CV 1/5] END ........hidden_layer_sizes=(2, 8);, score=0.414 total time=   6.6s
[CV 2/5] END ........hidden_layer_sizes=(2, 8);, score=0.541 total time=   7.2s
[CV 3/5] END ........hidden_layer_sizes=(2, 8);, score=0.582 total time=   6.5s
[CV 4/5] END ........hidden_layer_sizes=(2, 8);, score=0.255 total time=   6.9s
[CV 5/5] END ........hidden_layer_sizes=(2, 8);, score=0.235 total time=   6.9s
[CV 1/5] END ........hidden_layer_sizes=(2, 9);, score=0.488 total time=   6.4s
[CV 2/5] END ........hidden_layer_sizes=(2, 9);, score=0.539 total time=   9.9s
```

```
[CV 2/5] END .........hidden_layer_sizes=(2, 9);, score=0.559 total time=   9.9s
[CV 3/5] END .........hidden_layer_sizes=(2, 9);, score=0.581 total time=   7.3s
[CV 4/5] END .........hidden_layer_sizes=(2, 9);, score=0.224 total time=   7.6s
[CV 5/5] END .........hidden_layer_sizes=(2, 9);, score=0.272 total time=   5.8s
[CV 1/5] END ........hidden_layer_sizes=(2, 10);, score=0.416 total time=   5.9s
[CV 2/5] END ........hidden_layer_sizes=(2, 10);, score=0.540 total time=   6.1s
[CV 3/5] END ........hidden_layer_sizes=(2, 10);, score=0.582 total time=   6.6s
[CV 4/5] END ........hidden_layer_sizes=(2, 10);, score=0.282 total time=   9.6s
[CV 5/5] END ........hidden_layer_sizes=(2, 10);, score=0.237 total time=   8.2s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 1/5] END ........hidden_layer_sizes=(3, 1);, score=-5.116 total time=  58.3s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 2/5] END ........hidden_layer_sizes=(3, 1);, score=-6.727 total time=  59.8s
[CV 3/5] END .........hidden_layer_sizes=(3, 1);, score=0.582 total time=   8.5s
[CV 4/5] END .........hidden_layer_sizes=(3, 1);, score=0.223 total time=   9.9s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 5/5] END ........hidden_layer_sizes=(3, 1);, score=-2.856 total time= 1.0min
[CV 1/5] END .........hidden_layer_sizes=(3, 2);, score=0.413 total time=  12.7s
[CV 2/5] END .........hidden_layer_sizes=(3, 2);, score=0.539 total time=   8.9s
[CV 3/5] END .........hidden_layer_sizes=(3, 2);, score=0.578 total time=   7.3s
[CV 4/5] END .........hidden_layer_sizes=(3, 2);, score=0.244 total time=   9.2s
[CV 5/5] END .........hidden_layer_sizes=(3, 2);, score=0.237 total time=   7.1s
[CV 1/5] END .........hidden_layer_sizes=(3, 3);, score=0.415 total time=   5.5s
[CV 2/5] END .........hidden_layer_sizes=(3, 3);, score=0.695 total time=  12.5s
[CV 3/5] END .........hidden_layer_sizes=(3, 3);, score=0.722 total time=   8.7s
[CV 4/5] END .........hidden_layer_sizes=(3, 3);, score=0.239 total time=   7.0s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 5/5] END ........hidden_layer_sizes=(3, 3);, score=-2.861 total time= 1.1min
[CV 1/5] END .........hidden_layer_sizes=(3, 4);, score=0.414 total time=   5.3s
[CV 2/5] END .........hidden_layer_sizes=(3, 4);, score=0.540 total time=   5.8s
[CV 3/5] END .........hidden_layer_sizes=(3, 4);, score=0.587 total time=   6.3s
[CV 4/5] END .........hidden_layer_sizes=(3, 4);, score=0.308 total time=   8.6s
[CV 5/5] END .........hidden_layer_sizes=(3, 4);, score=0.236 total time=   7.6s
[CV 1/5] END .........hidden_layer_sizes=(3, 5);, score=0.418 total time=   6.7s
[CV 2/5] END .........hidden_layer_sizes=(3, 5);, score=0.544 total time=   6.6s
[CV 3/5] END .........hidden_layer_sizes=(3, 5);, score=0.588 total time=   4.6s
[CV 4/5] END .........hidden_layer_sizes=(3, 5);, score=0.220 total time=   8.1s
[CV 5/5] END .........hidden_layer_sizes=(3, 5);, score=0.277 total time=   6.7s
[CV 1/5] END .........hidden_layer_sizes=(3, 6);, score=0.415 total time=   5.2s
[CV 2/5] END .........hidden_layer_sizes=(3, 6);, score=0.546 total time=   6.0s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 3/5] END ........hidden_layer_sizes=(3, 6);, score=-6.489 total time= 1.2min
[CV 4/5] END .........hidden_layer_sizes=(3, 6);, score=0.343 total time=   7.9s
[CV 5/5] END .........hidden_layer_sizes=(3, 6);, score=0.236 total time=   3.7s
[CV 1/5] END .........hidden_layer_sizes=(3, 7);, score=0.553 total time=  10.7s
[CV 2/5] END .........hidden_layer_sizes=(3, 7);, score=0.591 total time=   7.5s
[CV 3/5] END .........hidden_layer_sizes=(3, 7);, score=0.631 total time=   7.6s
[CV 4/5] END .........hidden_layer_sizes=(3, 7);, score=0.261 total time=   5.6s
[CV 5/5] END .........hidden_layer_sizes=(3, 7);, score=0.357 total time=  10.1s
[CV 1/5] END .........hidden_layer_sizes=(3, 8);, score=0.483 total time=   6.0s
[CV 2/5] END .........hidden_layer_sizes=(3, 8);, score=0.603 total time=   8.2s
[CV 3/5] END .........hidden_layer_sizes=(3, 8);, score=0.770 total time=   7.9s
```

```
[CV 4/5] END .........hidden_layer_sizes=(3, 8);, score=0.222 total time=   6.9s
[CV 5/5] END .........hidden_layer_sizes=(3, 8);, score=0.359 total time=  12.5s
[CV 1/5] END .........hidden_layer_sizes=(3, 9);, score=0.415 total time=   8.9s
[CV 2/5] END .........hidden_layer_sizes=(3, 9);, score=0.695 total time=  13.3s
[CV 3/5] END .........hidden_layer_sizes=(3, 9);, score=0.630 total time=   7.2s
[CV 4/5] END .........hidden_layer_sizes=(3, 9);, score=0.265 total time=   9.0s
[CV 5/5] END .........hidden_layer_sizes=(3, 9);, score=0.361 total time=  13.2s
[CV 1/5] END ........hidden_layer_sizes=(3, 10);, score=0.433 total time=  14.0s
[CV 2/5] END ........hidden_layer_sizes=(3, 10);, score=0.548 total time=   5.9s
[CV 3/5] END ........hidden_layer_sizes=(3, 10);, score=0.704 total time=   9.7s
[CV 4/5] END ........hidden_layer_sizes=(3, 10);, score=0.312 total time=   7.2s
[CV 5/5] END ........hidden_layer_sizes=(3, 10);, score=0.294 total time=  19.0s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 1/5] END ........hidden_layer_sizes=(4, 1);, score=-5.131 total time=  54.6s
[CV 2/5] END .........hidden_layer_sizes=(4, 1);, score=0.601 total time=   9.0s
[CV 3/5] END .........hidden_layer_sizes=(4, 1);, score=0.632 total time=  30.9s
[CV 4/5] END .........hidden_layer_sizes=(4, 1);, score=0.221 total time=  11.3s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 5/5] END ........hidden_layer_sizes=(4, 1);, score=-2.818 total time=  54.6s
[CV 1/5] END .........hidden_layer_sizes=(4, 2);, score=0.414 total time=   6.3s
[CV 2/5] END .........hidden_layer_sizes=(4, 2);, score=0.543 total time=   6.8s
[CV 3/5] END .........hidden_layer_sizes=(4, 2);, score=0.591 total time=   7.7s
[CV 4/5] END .........hidden_layer_sizes=(4, 2);, score=0.222 total time=   8.2s
[CV 5/5] END .........hidden_layer_sizes=(4, 2);, score=0.335 total time=   8.0s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 1/5] END ........hidden_layer_sizes=(4, 3);, score=-5.169 total time= 1.0min
[CV 2/5] END .........hidden_layer_sizes=(4, 3);, score=0.606 total time=   7.7s
[CV 3/5] END .........hidden_layer_sizes=(4, 3);, score=0.642 total time=   5.3s
[CV 4/5] END .........hidden_layer_sizes=(4, 3);, score=0.239 total time=   5.8s
[CV 5/5] END .........hidden_layer_sizes=(4, 3);, score=0.274 total time=   4.9s
[CV 1/5] END .........hidden_layer_sizes=(4, 4);, score=0.414 total time=   5.1s
[CV 2/5] END .........hidden_layer_sizes=(4, 4);, score=0.538 total time=   4.8s
[CV 3/5] END .........hidden_layer_sizes=(4, 4);, score=0.634 total time=   8.2s
[CV 4/5] END .........hidden_layer_sizes=(4, 4);, score=0.340 total time=   6.5s
[CV 5/5] END .........hidden_layer_sizes=(4, 4);, score=0.237 total time=   4.9s
[CV 1/5] END .........hidden_layer_sizes=(4, 5);, score=0.507 total time=  14.3s
[CV 2/5] END .........hidden_layer_sizes=(4, 5);, score=0.768 total time=  12.7s
[CV 3/5] END .........hidden_layer_sizes=(4, 5);, score=0.804 total time=  22.5s
[CV 4/5] END .........hidden_layer_sizes=(4, 5);, score=0.232 total time=   4.8s
[CV 5/5] END .........hidden_layer_sizes=(4, 5);, score=0.277 total time=   5.6s
[CV 1/5] END .........hidden_layer_sizes=(4, 6);, score=0.406 total time=   4.4s
[CV 2/5] END .........hidden_layer_sizes=(4, 6);, score=0.604 total time=   5.9s
[CV 3/5] END .........hidden_layer_sizes=(4, 6);, score=0.810 total time=   7.4s
[CV 4/5] END .........hidden_layer_sizes=(4, 6);, score=0.233 total time=   7.6s
[CV 5/5] END .........hidden_layer_sizes=(4, 6);, score=0.278 total time=   5.5s
[CV 1/5] END .........hidden_layer_sizes=(4, 7);, score=0.419 total time=   7.1s
[CV 2/5] END .........hidden_layer_sizes=(4, 7);, score=0.625 total time=   9.5s
[CV 3/5] END .........hidden_layer_sizes=(4, 7);, score=0.589 total time=   4.5s
[CV 4/5] END .........hidden_layer_sizes=(4, 7);, score=0.223 total time=   5.3s
[CV 5/5] END .........hidden_layer_sizes=(4, 7);, score=0.238 total time=   4.7s
[CV 1/5] END .........hidden_layer_sizes=(4, 8);, score=0.442 total time=   8.5s
[CV 2/5] END .........hidden_layer_sizes=(4, 8);, score=0.591 total time=   7.9s
[CV 3/5] END .........hidden_layer_sizes=(4, 8);, score=0.641 total time=   5.2s
[CV 4/5] END .........hidden_layer_sizes=(4, 8);, score=0.258 total time=   9.2s
```

```
[CV 4/5] END .........hidden_layer_sizes=(4, 8);, score=0.358 total time=   9.3s
[CV 5/5] END .........hidden_layer_sizes=(4, 8);, score=0.238 total time=   5.0s
[CV 1/5] END .........hidden_layer_sizes=(4, 9);, score=0.446 total time=   3.9s
[CV 2/5] END .........hidden_layer_sizes=(4, 9);, score=0.784 total time=   5.8s
[CV 3/5] END .........hidden_layer_sizes=(4, 9);, score=0.587 total time=   8.1s
[CV 4/5] END .........hidden_layer_sizes=(4, 9);, score=0.243 total time=   5.4s
[CV 5/5] END .........hidden_layer_sizes=(4, 9);, score=0.238 total time=   4.1s
[CV 1/5] END ........hidden_layer_sizes=(4, 10);, score=0.581 total time=   6.4s
[CV 2/5] END ........hidden_layer_sizes=(4, 10);, score=0.545 total time=   6.1s
[CV 3/5] END ........hidden_layer_sizes=(4, 10);, score=0.801 total time=   6.9s
[CV 4/5] END ........hidden_layer_sizes=(4, 10);, score=0.235 total time=  15.9s
[CV 5/5] END ........hidden_layer_sizes=(4, 10);, score=0.277 total time=   5.7s
[CV 1/5] END .........hidden_layer_sizes=(5, 1);, score=0.480 total time=   6.9s
[CV 2/5] END .........hidden_layer_sizes=(5, 1);, score=0.544 total time=   5.5s
[CV 3/5] END .........hidden_layer_sizes=(5, 1);, score=0.588 total time=  18.9s
[CV 4/5] END .........hidden_layer_sizes=(5, 1);, score=0.220 total time=   6.1s
[CV 5/5] END .........hidden_layer_sizes=(5, 1);, score=0.275 total time=   9.3s
[CV 1/5] END .........hidden_layer_sizes=(5, 2);, score=0.472 total time=   6.7s
[CV 2/5] END .........hidden_layer_sizes=(5, 2);, score=0.543 total time=  21.1s
[CV 3/5] END .........hidden_layer_sizes=(5, 2);, score=0.581 total time=   6.8s
[CV 4/5] END .........hidden_layer_sizes=(5, 2);, score=0.220 total time=   6.2s
[CV 5/5] END .........hidden_layer_sizes=(5, 2);, score=0.235 total time=  22.1s
[CV 1/5] END .........hidden_layer_sizes=(5, 3);, score=0.474 total time=   5.3s
[CV 2/5] END .........hidden_layer_sizes=(5, 3);, score=0.786 total time=  10.8s
[CV 3/5] END .........hidden_layer_sizes=(5, 3);, score=0.624 total time=   7.7s
[CV 4/5] END .........hidden_layer_sizes=(5, 3);, score=0.345 total time=  17.4s
[CV 5/5] END .........hidden_layer_sizes=(5, 3);, score=0.236 total time=   4.4s
[CV 1/5] END .........hidden_layer_sizes=(5, 4);, score=0.412 total time=   4.6s
[CV 2/5] END .........hidden_layer_sizes=(5, 4);, score=0.768 total time=   8.1s
[CV 3/5] END .........hidden_layer_sizes=(5, 4);, score=0.819 total time=   5.7s
[CV 4/5] END .........hidden_layer_sizes=(5, 4);, score=0.222 total time=   5.6s
[CV 5/5] END .........hidden_layer_sizes=(5, 4);, score=0.355 total time=  14.0s
[CV 1/5] END .........hidden_layer_sizes=(5, 5);, score=0.565 total time=   9.8s
[CV 2/5] END .........hidden_layer_sizes=(5, 5);, score=0.602 total time=   6.1s
[CV 3/5] END .........hidden_layer_sizes=(5, 5);, score=0.641 total time=   6.0s
[CV 4/5] END .........hidden_layer_sizes=(5, 5);, score=0.212 total time=   5.7s
[CV 5/5] END .........hidden_layer_sizes=(5, 5);, score=0.352 total time=  13.4s
[CV 1/5] END .........hidden_layer_sizes=(5, 6);, score=0.458 total time=   5.4s
[CV 2/5] END .........hidden_layer_sizes=(5, 6);, score=0.603 total time=   6.4s
[CV 3/5] END .........hidden_layer_sizes=(5, 6);, score=0.808 total time=  12.1s
[CV 4/5] END .........hidden_layer_sizes=(5, 6);, score=0.361 total time=  12.6s
[CV 5/5] END .........hidden_layer_sizes=(5, 6);, score=0.237 total time=   3.9s
[CV 1/5] END .........hidden_layer_sizes=(5, 7);, score=0.562 total time=   8.5s
[CV 2/5] END .........hidden_layer_sizes=(5, 7);, score=0.604 total time=   6.5s
[CV 3/5] END .........hidden_layer_sizes=(5, 7);, score=0.698 total time=   7.6s
[CV 4/5] END .........hidden_layer_sizes=(5, 7);, score=0.321 total time=  13.4s
[CV 5/5] END .........hidden_layer_sizes=(5, 7);, score=0.272 total time=   5.1s
[CV 1/5] END .........hidden_layer_sizes=(5, 8);, score=0.590 total time=   7.6s
[CV 2/5] END .........hidden_layer_sizes=(5, 8);, score=0.595 total time=   6.0s
[CV 3/5] END .........hidden_layer_sizes=(5, 8);, score=0.827 total time=  13.8s
[CV 4/5] END .........hidden_layer_sizes=(5, 8);, score=0.331 total time=   5.4s
[CV 5/5] END .........hidden_layer_sizes=(5, 8);, score=0.238 total time=   3.7s
[CV 1/5] END .........hidden_layer_sizes=(5, 9);, score=0.416 total time=   4.8s
[CV 2/5] END .........hidden_layer_sizes=(5, 9);, score=0.776 total time=   9.6s
[CV 3/5] END .........hidden_layer_sizes=(5, 9);, score=0.621 total time=   5.5s
[CV 4/5] END .........hidden_layer_sizes=(5, 9);, score=0.249 total time=   4.1s
[CV 5/5] END .........hidden_layer_sizes=(5, 9);, score=0.367 total time=   8.5s
```

```
[CV 1/5] END ........hidden_layer_sizes=(5, 10);, score=0.592 total time=   6.6s
[CV 2/5] END ........hidden_layer_sizes=(5, 10);, score=0.592 total time=   5.2s
[CV 3/5] END ........hidden_layer_sizes=(5, 10);, score=0.582 total time=   5.7s
[CV 4/5] END ........hidden_layer_sizes=(5, 10);, score=0.269 total time=  13.7s
[CV 5/5] END ........hidden_layer_sizes=(5, 10);, score=0.343 total time=  10.4s
[CV 1/5] END .........hidden_layer_sizes=(6, 1);, score=0.481 total time=  23.6s
[CV 2/5] END .........hidden_layer_sizes=(6, 1);, score=0.595 total time=   7.8s
[CV 3/5] END .........hidden_layer_sizes=(6, 1);, score=0.586 total time=  27.8s
[CV 4/5] END .........hidden_layer_sizes=(6, 1);, score=0.233 total time=  22.7s
[CV 5/5] END .........hidden_layer_sizes=(6, 1);, score=0.237 total time=   5.8s
[CV 1/5] END .........hidden_layer_sizes=(6, 2);, score=0.478 total time=   6.3s
[CV 2/5] END .........hidden_layer_sizes=(6, 2);, score=0.778 total time=  18.9s
[CV 3/5] END .........hidden_layer_sizes=(6, 2);, score=0.643 total time=   6.4s
[CV 4/5] END .........hidden_layer_sizes=(6, 2);, score=0.242 total time=   6.4s
[CV 5/5] END .........hidden_layer_sizes=(6, 2);, score=0.359 total time=  11.0s
[CV 1/5] END .........hidden_layer_sizes=(6, 3);, score=0.419 total time=   5.1s
[CV 2/5] END .........hidden_layer_sizes=(6, 3);, score=0.767 total time=  17.8s
[CV 3/5] END .........hidden_layer_sizes=(6, 3);, score=0.797 total time=  28.8s
[CV 4/5] END .........hidden_layer_sizes=(6, 3);, score=0.222 total time=   4.9s
[CV 5/5] END .........hidden_layer_sizes=(6, 3);, score=0.279 total time=   5.0s
[CV 1/5] END .........hidden_layer_sizes=(6, 4);, score=0.597 total time=  10.0s
[CV 2/5] END .........hidden_layer_sizes=(6, 4);, score=0.607 total time=   5.0s
[CV 3/5] END .........hidden_layer_sizes=(6, 4);, score=0.587 total time=   4.4s
[CV 4/5] END .........hidden_layer_sizes=(6, 4);, score=0.359 total time=  10.3s
[CV 5/5] END .........hidden_layer_sizes=(6, 4);, score=0.277 total time=   5.1s
[CV 1/5] END .........hidden_layer_sizes=(6, 5);, score=0.405 total time=   5.2s
[CV 2/5] END .........hidden_layer_sizes=(6, 5);, score=0.874 total time=  14.5s
[CV 3/5] END .........hidden_layer_sizes=(6, 5);, score=0.817 total time=   6.8s
[CV 4/5] END .........hidden_layer_sizes=(6, 5);, score=0.244 total time=   5.3s
[CV 5/5] END .........hidden_layer_sizes=(6, 5);, score=0.233 total time=   4.5s
[CV 1/5] END .........hidden_layer_sizes=(6, 6);, score=0.573 total time=   9.2s
[CV 2/5] END .........hidden_layer_sizes=(6, 6);, score=0.597 total time=   4.8s
[CV 3/5] END .........hidden_layer_sizes=(6, 6);, score=0.574 total time=   4.1s
[CV 4/5] END .........hidden_layer_sizes=(6, 6);, score=0.240 total time=   6.9s
[CV 5/5] END .........hidden_layer_sizes=(6, 6);, score=0.237 total time=   5.1s
[CV 1/5] END .........hidden_layer_sizes=(6, 7);, score=0.590 total time=   9.5s
[CV 2/5] END .........hidden_layer_sizes=(6, 7);, score=0.785 total time=   8.5s
[CV 3/5] END .........hidden_layer_sizes=(6, 7);, score=0.571 total time=   4.7s
[CV 4/5] END .........hidden_layer_sizes=(6, 7);, score=0.346 total time=  11.5s
[CV 5/5] END .........hidden_layer_sizes=(6, 7);, score=0.276 total time=   4.3s
[CV 1/5] END .........hidden_layer_sizes=(6, 8);, score=0.412 total time=   4.6s
[CV 2/5] END .........hidden_layer_sizes=(6, 8);, score=0.869 total time=  12.7s
[CV 3/5] END .........hidden_layer_sizes=(6, 8);, score=0.788 total time=   8.4s
[CV 4/5] END .........hidden_layer_sizes=(6, 8);, score=0.220 total time=   6.3s
[CV 5/5] END .........hidden_layer_sizes=(6, 8);, score=0.238 total time=   4.3s
[CV 1/5] END .........hidden_layer_sizes=(6, 9);, score=0.589 total time=   8.6s
[CV 2/5] END .........hidden_layer_sizes=(6, 9);, score=0.534 total time=   5.2s
[CV 3/5] END .........hidden_layer_sizes=(6, 9);, score=0.817 total time=  11.7s
[CV 4/5] END .........hidden_layer_sizes=(6, 9);, score=0.222 total time=   4.9s
[CV 5/5] END .........hidden_layer_sizes=(6, 9);, score=0.238 total time=   3.9s
[CV 1/5] END ........hidden_layer_sizes=(6, 10);, score=0.587 total time=   8.1s
[CV 2/5] END ........hidden_layer_sizes=(6, 10);, score=0.539 total time=   3.3s
[CV 3/5] END ........hidden_layer_sizes=(6, 10);, score=0.580 total time=   6.7s
[CV 4/5] END ........hidden_layer_sizes=(6, 10);, score=0.225 total time=   4.0s
[CV 5/5] END ........hidden_layer_sizes=(6, 10);, score=0.275 total time=   4.8s
[CV 1/5] END .........hidden_layer_sizes=(7, 1);, score=0.470 total time=  26.1s
```

```
[CV 2/5] END .........hidden_layer_sizes=(7, 1);, score=0.596 total time=   6.5s
[CV 3/5] END .........hidden_layer_sizes=(7, 1);, score=0.588 total time=  24.9s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 4/5] END .........hidden_layer_sizes=(7, 1);, score=-2.734 total time=  59.9s
[CV 5/5] END .........hidden_layer_sizes=(7, 1);, score=0.276 total time=   5.9s
[CV 1/5] END .........hidden_layer_sizes=(7, 2);, score=0.476 total time=   6.5s
[CV 2/5] END .........hidden_layer_sizes=(7, 2);, score=0.598 total time=   5.4s
[CV 3/5] END .........hidden_layer_sizes=(7, 2);, score=0.804 total time=  10.0s
[CV 4/5] END .........hidden_layer_sizes=(7, 2);, score=0.242 total time=   5.5s
[CV 5/5] END .........hidden_layer_sizes=(7, 2);, score=0.236 total time=   5.1s
[CV 1/5] END .........hidden_layer_sizes=(7, 3);, score=0.479 total time=  16.2s
[CV 2/5] END .........hidden_layer_sizes=(7, 3);, score=0.603 total time=   6.1s
[CV 3/5] END .........hidden_layer_sizes=(7, 3);, score=0.640 total time=   8.4s
[CV 4/5] END .........hidden_layer_sizes=(7, 3);, score=0.349 total time=  12.6s
[CV 5/5] END .........hidden_layer_sizes=(7, 3);, score=0.275 total time=   5.7s
[CV 1/5] END .........hidden_layer_sizes=(7, 4);, score=0.471 total time=   6.2s
[CV 2/5] END .........hidden_layer_sizes=(7, 4);, score=0.544 total time=   6.2s
[CV 3/5] END .........hidden_layer_sizes=(7, 4);, score=0.639 total time=   5.4s
[CV 4/5] END .........hidden_layer_sizes=(7, 4);, score=0.240 total time=   6.1s
[CV 5/5] END .........hidden_layer_sizes=(7, 4);, score=0.279 total time=   5.0s
[CV 1/5] END .........hidden_layer_sizes=(7, 5);, score=0.479 total time=   4.2s
[CV 2/5] END .........hidden_layer_sizes=(7, 5);, score=0.599 total time=   4.9s
[CV 3/5] END .........hidden_layer_sizes=(7, 5);, score=0.637 total time=   4.9s
[CV 4/5] END .........hidden_layer_sizes=(7, 5);, score=0.246 total time=   5.7s
[CV 5/5] END .........hidden_layer_sizes=(7, 5);, score=0.359 total time=  10.6s
[CV 1/5] END .........hidden_layer_sizes=(7, 6);, score=0.485 total time=   6.5s
[CV 2/5] END .........hidden_layer_sizes=(7, 6);, score=0.605 total time=   6.4s
[CV 3/5] END .........hidden_layer_sizes=(7, 6);, score=0.576 total time=   5.1s
[CV 4/5] END .........hidden_layer_sizes=(7, 6);, score=0.252 total time=   7.7s
[CV 5/5] END .........hidden_layer_sizes=(7, 6);, score=0.279 total time=   5.3s
[CV 1/5] END .........hidden_layer_sizes=(7, 7);, score=0.408 total time=   5.0s
[CV 2/5] END .........hidden_layer_sizes=(7, 7);, score=0.607 total time=   5.4s
[CV 3/5] END .........hidden_layer_sizes=(7, 7);, score=0.644 total time=   4.8s
[CV 4/5] END .........hidden_layer_sizes=(7, 7);, score=0.262 total time=   5.7s
[CV 5/5] END .........hidden_layer_sizes=(7, 7);, score=0.236 total time=   4.6s
[CV 1/5] END .........hidden_layer_sizes=(7, 8);, score=0.485 total time=   4.8s
[CV 2/5] END .........hidden_layer_sizes=(7, 8);, score=0.542 total time=   3.6s
[CV 3/5] END .........hidden_layer_sizes=(7, 8);, score=0.579 total time=   4.5s
[CV 4/5] END .........hidden_layer_sizes=(7, 8);, score=0.224 total time=   4.5s
[CV 5/5] END .........hidden_layer_sizes=(7, 8);, score=0.274 total time=   4.1s
[CV 1/5] END .........hidden_layer_sizes=(7, 9);, score=0.480 total time=   4.5s
[CV 2/5] END .........hidden_layer_sizes=(7, 9);, score=0.775 total time=   8.0s
[CV 3/5] END .........hidden_layer_sizes=(7, 9);, score=0.758 total time=   8.4s
[CV 4/5] END .........hidden_layer_sizes=(7, 9);, score=0.331 total time=   7.0s
[CV 5/5] END .........hidden_layer_sizes=(7, 9);, score=0.357 total time=   8.5s
[CV 1/5] END ........hidden_layer_sizes=(7, 10);, score=0.474 total time=   4.7s
[CV 2/5] END ........hidden_layer_sizes=(7, 10);, score=0.761 total time=   5.6s
[CV 3/5] END ........hidden_layer_sizes=(7, 10);, score=0.769 total time=  10.3s
[CV 4/5] END ........hidden_layer_sizes=(7, 10);, score=0.344 total time=   9.4s
[CV 5/5] END ........hidden_layer_sizes=(7, 10);, score=0.302 total time=   5.7s
[CV 1/5] END .........hidden_layer_sizes=(8, 1);, score=0.414 total time=  12.0s
[CV 2/5] END .........hidden_layer_sizes=(8, 1);, score=0.602 total time=   6.0s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 3/5] END ........hidden_layer_sizes=(8, 1);, score=-6.472 total time=  58.1s
[CV 4/5] END .........hidden_layer_sizes=(8, 1);, score=0.225 total time=  13.3s
```

```
[CV 5/5] END .........hidden_layer_sizes=(8, 1);, score=0.237 total time=   6.0s
[CV 1/5] END .........hidden_layer_sizes=(8, 2);, score=0.503 total time=   9.8s
[CV 2/5] END .........hidden_layer_sizes=(8, 2);, score=0.586 total time=  26.6s
[CV 3/5] END .........hidden_layer_sizes=(8, 2);, score=0.629 total time=   9.1s
[CV 4/5] END .........hidden_layer_sizes=(8, 2);, score=0.331 total time=  11.9s
[CV 5/5] END .........hidden_layer_sizes=(8, 2);, score=0.276 total time=   5.3s
[CV 1/5] END .........hidden_layer_sizes=(8, 3);, score=0.482 total time=   4.9s
[CV 2/5] END .........hidden_layer_sizes=(8, 3);, score=0.781 total time=   8.2s
[CV 3/5] END .........hidden_layer_sizes=(8, 3);, score=0.802 total time=   9.3s
[CV 4/5] END .........hidden_layer_sizes=(8, 3);, score=0.237 total time=   5.1s
[CV 5/5] END .........hidden_layer_sizes=(8, 3);, score=0.352 total time=  13.1s
[CV 1/5] END .........hidden_layer_sizes=(8, 4);, score=0.591 total time=   8.8s
[CV 2/5] END .........hidden_layer_sizes=(8, 4);, score=0.734 total time=   9.5s
[CV 3/5] END .........hidden_layer_sizes=(8, 4);, score=0.812 total time=   9.5s
[CV 4/5] END .........hidden_layer_sizes=(8, 4);, score=0.239 total time=   5.7s
[CV 5/5] END .........hidden_layer_sizes=(8, 4);, score=0.360 total time=  10.8s
[CV 1/5] END .........hidden_layer_sizes=(8, 5);, score=0.479 total time=   5.2s
[CV 2/5] END .........hidden_layer_sizes=(8, 5);, score=0.783 total time=   9.0s
[CV 3/5] END .........hidden_layer_sizes=(8, 5);, score=0.785 total time=   6.1s
[CV 4/5] END .........hidden_layer_sizes=(8, 5);, score=0.243 total time=   5.1s
[CV 5/5] END .........hidden_layer_sizes=(8, 5);, score=0.353 total time=  10.4s
[CV 1/5] END .........hidden_layer_sizes=(8, 6);, score=0.414 total time=   5.4s
[CV 2/5] END .........hidden_layer_sizes=(8, 6);, score=0.769 total time=  11.1s
[CV 3/5] END .........hidden_layer_sizes=(8, 6);, score=0.814 total time=  12.2s
[CV 4/5] END .........hidden_layer_sizes=(8, 6);, score=0.239 total time=   5.5s
[CV 5/5] END .........hidden_layer_sizes=(8, 6);, score=0.237 total time=   4.3s
[CV 1/5] END .........hidden_layer_sizes=(8, 7);, score=0.591 total time=   6.7s
[CV 2/5] END .........hidden_layer_sizes=(8, 7);, score=0.543 total time=   6.4s
[CV 3/5] END .........hidden_layer_sizes=(8, 7);, score=0.589 total time=   3.6s
[CV 4/5] END .........hidden_layer_sizes=(8, 7);, score=0.344 total time=   5.5s
[CV 5/5] END .........hidden_layer_sizes=(8, 7);, score=0.269 total time=   6.8s
[CV 1/5] END .........hidden_layer_sizes=(8, 8);, score=0.412 total time=   4.0s
[CV 2/5] END .........hidden_layer_sizes=(8, 8);, score=0.803 total time=   8.2s
[CV 3/5] END .........hidden_layer_sizes=(8, 8);, score=0.584 total time=   4.1s
[CV 4/5] END .........hidden_layer_sizes=(8, 8);, score=0.224 total time=   4.3s
[CV 5/5] END .........hidden_layer_sizes=(8, 8);, score=0.360 total time=   7.6s
[CV 1/5] END .........hidden_layer_sizes=(8, 9);, score=0.442 total time=   4.2s
[CV 2/5] END .........hidden_layer_sizes=(8, 9);, score=0.600 total time=   5.0s
[CV 3/5] END .........hidden_layer_sizes=(8, 9);, score=0.637 total time=   4.6s
[CV 4/5] END .........hidden_layer_sizes=(8, 9);, score=0.332 total time=   8.5s
[CV 5/5] END .........hidden_layer_sizes=(8, 9);, score=0.236 total time=   4.6s
[CV 1/5] END ........hidden_layer_sizes=(8, 10);, score=0.403 total time=   3.9s
[CV 2/5] END ........hidden_layer_sizes=(8, 10);, score=0.538 total time=   4.2s
[CV 3/5] END ........hidden_layer_sizes=(8, 10);, score=0.795 total time=   7.3s
[CV 4/5] END ........hidden_layer_sizes=(8, 10);, score=0.318 total time=   7.8s
[CV 5/5] END ........hidden_layer_sizes=(8, 10);, score=0.357 total time=  10.6s
[CV 1/5] END .........hidden_layer_sizes=(9, 1);, score=0.415 total time=  35.6s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 2/5] END ........hidden_layer_sizes=(9, 1);, score=-6.754 total time= 1.1min
[CV 3/5] END .........hidden_layer_sizes=(9, 1);, score=0.587 total time=  13.4s
[CV 4/5] END .........hidden_layer_sizes=(9, 1);, score=0.222 total time=   5.7s
[CV 5/5] END .........hidden_layer_sizes=(9, 1);, score=0.276 total time=   6.0s
[CV 1/5] END .........hidden_layer_sizes=(9, 2);, score=0.588 total time=  11.9s
[CV 2/5] END .........hidden_layer_sizes=(9, 2);, score=0.544 total time=  33.1s
[CV 3/5] END .........hidden_layer_sizes=(9, 2);, score=0.627 total time=   6.3s
```

```
[CV 4/5] END .........hidden_layer_sizes=(9, 2);, score=0.225 total time=   5.7s
[CV 5/5] END .........hidden_layer_sizes=(9, 2);, score=0.277 total time=   6.5s
[CV 1/5] END .........hidden_layer_sizes=(9, 3);, score=0.416 total time=  31.8s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 2/5] END .........hidden_layer_sizes=(9, 3);, score=-6.757 total time= 1.3min
[CV 3/5] END .........hidden_layer_sizes=(9, 3);, score=0.646 total time=  19.1s
[CV 4/5] END .........hidden_layer_sizes=(9, 3);, score=0.223 total time=  37.2s
[CV 5/5] END .........hidden_layer_sizes=(9, 3);, score=0.266 total time=   4.7s
[CV 1/5] END .........hidden_layer_sizes=(9, 4);, score=0.411 total time=   5.2s
[CV 2/5] END .........hidden_layer_sizes=(9, 4);, score=0.601 total time=  25.0s
[CV 3/5] END .........hidden_layer_sizes=(9, 4);, score=0.792 total time=   9.0s
[CV 4/5] END .........hidden_layer_sizes=(9, 4);, score=0.333 total time=  14.3s
[CV 5/5] END .........hidden_layer_sizes=(9, 4);, score=0.277 total time=   4.4s
[CV 1/5] END .........hidden_layer_sizes=(9, 5);, score=0.621 total time=  11.8s
[CV 2/5] END .........hidden_layer_sizes=(9, 5);, score=0.545 total time=   5.9s
[CV 3/5] END .........hidden_layer_sizes=(9, 5);, score=0.639 total time=   4.9s
[CV 4/5] END .........hidden_layer_sizes=(9, 5);, score=0.307 total time=   9.5s
[CV 5/5] END .........hidden_layer_sizes=(9, 5);, score=0.366 total time=   8.5s
[CV 1/5] END .........hidden_layer_sizes=(9, 6);, score=0.412 total time=   5.4s
[CV 2/5] END .........hidden_layer_sizes=(9, 6);, score=0.597 total time=   4.4s
[CV 3/5] END .........hidden_layer_sizes=(9, 6);, score=0.812 total time=   9.4s
[CV 4/5] END .........hidden_layer_sizes=(9, 6);, score=0.224 total time=   4.7s
[CV 5/5] END .........hidden_layer_sizes=(9, 6);, score=0.277 total time=   3.9s
[CV 1/5] END .........hidden_layer_sizes=(9, 7);, score=0.559 total time=   7.2s
[CV 2/5] END .........hidden_layer_sizes=(9, 7);, score=0.602 total time=   5.6s
[CV 3/5] END .........hidden_layer_sizes=(9, 7);, score=0.639 total time=   4.4s
[CV 4/5] END .........hidden_layer_sizes=(9, 7);, score=0.222 total time=   4.0s
[CV 5/5] END .........hidden_layer_sizes=(9, 7);, score=0.363 total time=   7.6s
[CV 1/5] END .........hidden_layer_sizes=(9, 8);, score=0.593 total time=   5.9s
[CV 2/5] END .........hidden_layer_sizes=(9, 8);, score=0.774 total time=  10.6s
[CV 3/5] END .........hidden_layer_sizes=(9, 8);, score=0.854 total time=  11.4s
[CV 4/5] END .........hidden_layer_sizes=(9, 8);, score=0.337 total time=   9.7s
[CV 5/5] END .........hidden_layer_sizes=(9, 8);, score=0.237 total time=   3.6s
[CV 1/5] END .........hidden_layer_sizes=(9, 9);, score=0.582 total time=   6.9s
[CV 2/5] END .........hidden_layer_sizes=(9, 9);, score=0.783 total time=   7.3s
[CV 3/5] END .........hidden_layer_sizes=(9, 9);, score=0.814 total time=   6.2s
[CV 4/5] END .........hidden_layer_sizes=(9, 9);, score=0.342 total time=   8.5s
[CV 5/5] END .........hidden_layer_sizes=(9, 9);, score=0.363 total time=   9.3s
[CV 1/5] END ........hidden_layer_sizes=(9, 10);, score=0.470 total time=   4.5s
[CV 2/5] END ........hidden_layer_sizes=(9, 10);, score=0.542 total time=   4.0s
[CV 3/5] END ........hidden_layer_sizes=(9, 10);, score=0.582 total time=   6.8s
[CV 4/5] END ........hidden_layer_sizes=(9, 10);, score=0.324 total time=   5.9s
[CV 5/5] END ........hidden_layer_sizes=(9, 10);, score=0.336 total time=   9.7s
[CV 1/5] END ........hidden_layer_sizes=(10, 1);, score=0.463 total time=  22.6s
[CV 2/5] END ........hidden_layer_sizes=(10, 1);, score=0.599 total time=  32.6s
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 3/5] END .......hidden_layer_sizes=(10, 1);, score=-6.533 total time= 1.2min
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 4/5] END .......hidden_layer_sizes=(10, 1);, score=-2.708 total time= 1.1min
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
[CV 5/5] END .......hidden_layer_sizes=(10, 1);, score=-2.824 total time= 1.0min
[CV 1/5] END ........hidden_layer_sizes=(10, 2);, score=0.467 total time=   5.0s
[CV 2/5] END ........hidden_layer_sizes=(10, 2);, score=0.543 total time=   5.2s
```

```
[CV 3/5] END ........hidden_layer_sizes=(10, 2);, score=0.631 total time=   6.0s
[CV 4/5] END ........hidden_layer_sizes=(10, 2);, score=0.248 total time=   6.5s
[CV 5/5] END ........hidden_layer_sizes=(10, 2);, score=0.355 total time=  12.6s
[CV 1/5] END ........hidden_layer_sizes=(10, 3);, score=0.600 total time=   9.9s
[CV 2/5] END ........hidden_layer_sizes=(10, 3);, score=0.600 total time=   4.9s
[CV 3/5] END ........hidden_layer_sizes=(10, 3);, score=0.812 total time=  11.1s
[CV 4/5] END ........hidden_layer_sizes=(10, 3);, score=0.246 total time=   8.3s
[CV 5/5] END ........hidden_layer_sizes=(10, 3);, score=0.238 total time=  36.0s
[CV 1/5] END ........hidden_layer_sizes=(10, 4);, score=0.585 total time=  11.2s
[CV 2/5] END ........hidden_layer_sizes=(10, 4);, score=0.749 total time=   6.2s
[CV 3/5] END ........hidden_layer_sizes=(10, 4);, score=0.635 total time=   6.4s
[CV 4/5] END ........hidden_layer_sizes=(10, 4);, score=0.238 total time=   4.5s
[CV 5/5] END ........hidden_layer_sizes=(10, 4);, score=0.360 total time=  10.9s
[CV 1/5] END ........hidden_layer_sizes=(10, 5);, score=0.584 total time=   6.9s
[CV 2/5] END ........hidden_layer_sizes=(10, 5);, score=0.754 total time=   7.0s
[CV 3/5] END ........hidden_layer_sizes=(10, 5);, score=0.585 total time=   4.1s
[CV 4/5] END ........hidden_layer_sizes=(10, 5);, score=0.298 total time=   9.4s
[CV 5/5] END ........hidden_layer_sizes=(10, 5);, score=0.278 total time=   4.9s
[CV 1/5] END ........hidden_layer_sizes=(10, 6);, score=0.573 total time=   8.6s
[CV 2/5] END ........hidden_layer_sizes=(10, 6);, score=0.794 total time=   6.7s
[CV 3/5] END ........hidden_layer_sizes=(10, 6);, score=0.799 total time=   8.3s
[CV 4/5] END ........hidden_layer_sizes=(10, 6);, score=0.330 total time=   5.9s
[CV 5/5] END ........hidden_layer_sizes=(10, 6);, score=0.279 total time=   4.2s
[CV 1/5] END ........hidden_layer_sizes=(10, 7);, score=0.558 total time=   9.2s
[CV 2/5] END ........hidden_layer_sizes=(10, 7);, score=0.783 total time=   6.1s
[CV 3/5] END ........hidden_layer_sizes=(10, 7);, score=0.821 total time=   9.0s
[CV 4/5] END ........hidden_layer_sizes=(10, 7);, score=0.319 total time=   5.3s
[CV 5/5] END ........hidden_layer_sizes=(10, 7);, score=0.272 total time=   3.6s
[CV 1/5] END ........hidden_layer_sizes=(10, 8);, score=0.566 total time=   7.1s
[CV 2/5] END ........hidden_layer_sizes=(10, 8);, score=0.538 total time=   5.2s
[CV 3/5] END ........hidden_layer_sizes=(10, 8);, score=0.851 total time=   9.7s
[CV 4/5] END ........hidden_layer_sizes=(10, 8);, score=0.292 total time=   4.7s
[CV 5/5] END ........hidden_layer_sizes=(10, 8);, score=0.279 total time=   3.6s
[CV 1/5] END ........hidden_layer_sizes=(10, 9);, score=0.590 total time=  12.5s
[CV 2/5] END ........hidden_layer_sizes=(10, 9);, score=0.779 total time=   7.1s
[CV 3/5] END ........hidden_layer_sizes=(10, 9);, score=0.631 total time=   4.5s
[CV 4/5] END ........hidden_layer_sizes=(10, 9);, score=0.316 total time=   7.3s
[CV 5/5] END ........hidden_layer_sizes=(10, 9);, score=0.350 total time=   5.4s
[CV 1/5] END .......hidden_layer_sizes=(10, 10);, score=0.416 total time=   3.3s
[CV 2/5] END .......hidden_layer_sizes=(10, 10);, score=0.575 total time=   5.3s
[CV 3/5] END .......hidden_layer_sizes=(10, 10);, score=0.571 total time=   4.5s
[CV 4/5] END .......hidden_layer_sizes=(10, 10);, score=0.247 total time=   4.5s
[CV 5/5] END .......hidden_layer_sizes=(10, 10);, score=0.365 total time=   7.5s
Grid best parameter (max. R2):  {'hidden_layer_sizes': (9, 9)}
Grid best score (R2):  0.5767142177992696
R2 Score on test: 0.7896391321596805
```

Based on the models above, we obtained that the best method to predict the energy was a Random Forest with 70 Trees using the following variables:

'temp_per_day_2005','HourOfDay','full_humid_2005'

- Mean energy per Day
- Hour of the Day
- Humidity of the Hour

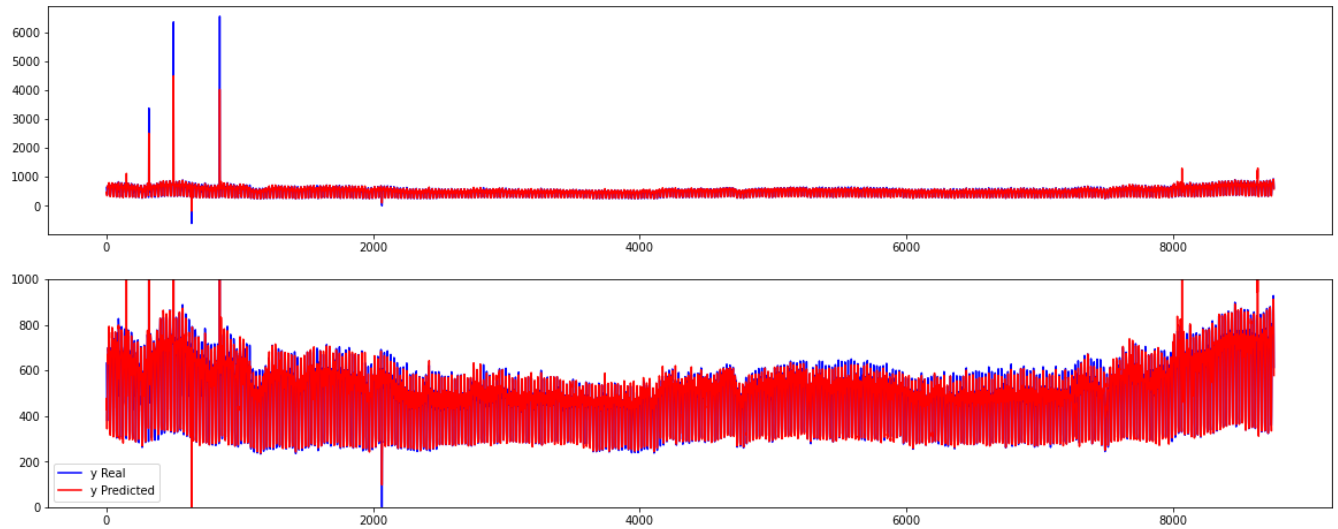
Hence, we obtained 2 different models to predict energy, each one for a different year

02.5 Final Model for 2005

```
X_2005= df_energy[['temp_per_day_2005','HourOfDay','full_humid_2005']]
y_2005= df_energy['energy_consumpt_2005']

X_train, X_test, y_train, y_test= train_test_split(X_2005,y_2005,random_state=21)
```

```
parameters= {"n_estimators":list(range(10,101,10))}
rf = RandomForestRegressor(random_state=0, n_estimators=70)
rf.fit(X_train,y_train)

print("R2 Score on test:",r2_score(y_test, rf.predict(X_test)))
fig, axs= plt.subplots(2,1,figsize=(20,8))
axs[0]= plotRgresion(y_2005,rf.predict(X_2005),axs[0],False,0,1000)
axs[1]= plotRgresion(y_2005,rf.predict(X_2005),axs[1],True,0,1000)
plt.legend()
plt.show()
print("Final R2 Score:",r2_score(y_2005,rf.predict(X_2005)))
```

```
R2 Score on test: 0.6105817867373805
```



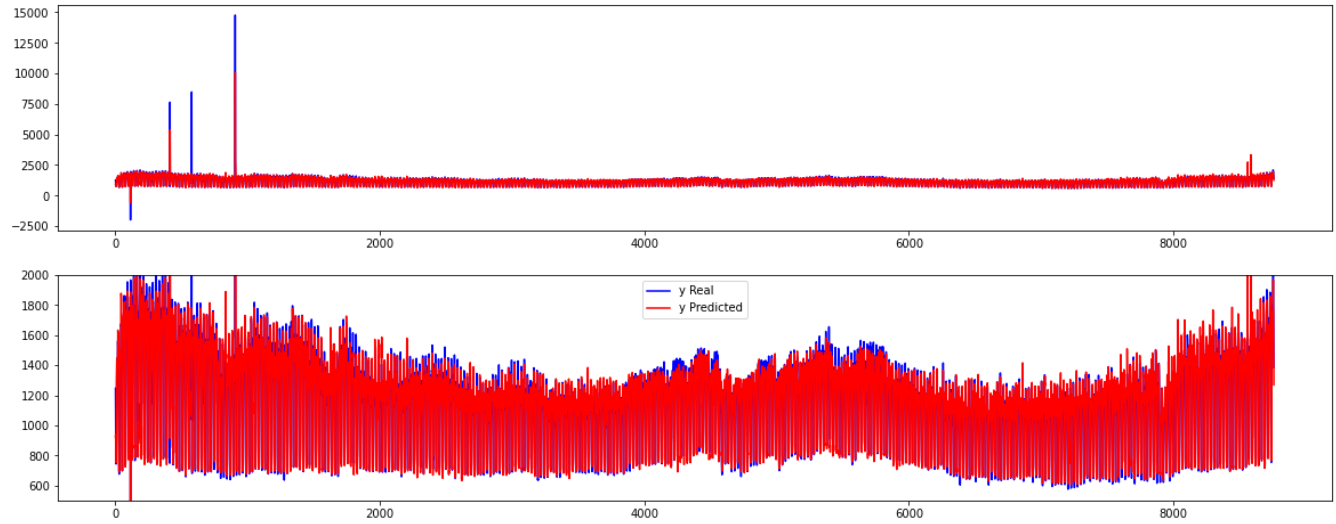

```
Final R2 Score: 0.8696535631662909
```

02.6 Final Model for 2006

```
X_2006= df_energy[['temp_per_day_2006','HourOfDay','full_humid_2006']]
y_2006= df_energy['energy_consumpt_2006']

X_train, X_test, y_train, y_test= train_test_split(X_2006,y_2006,random_state=21)
```

```
parameters= {"n_estimators":list(range(10,101,10))}
rf = RandomForestRegressor(random_state=0, n_estimators=70)
rf.fit(X_train,y_train)

print("R2 Score on test:",r2_score(y_test, rf.predict(X_test)))
fig, axs= plt.subplots(2,1,figsize=(20,8))
axs[0]= plotRgresion(y_2006,rf.predict(X_2006),axs[0],False,0,1000)
axs[1]= plotRgresion(y_2006,rf.predict(X_2006),axs[1],True,500,2000)
plt.legend()
plt.show()
print("Final R2 Score:",r2_score(y_2006,rf.predict(X_2006)))
```

```
R2 Score on test: 0.7109454039566507
```




```
Final R2 Score: 0.8857925371269807
```

03. Time Series Forecast

The previous models can predcit the energy of the hour based on some variables, however, we can see that both years have a similar pattern in the energy, but it has a different scale.

Hence, if we apply a tiem series forecast we might get a better model to predict the energy based on time

```
df_energy= pd.read_csv("/content/energy__data-2 (1).csv")
df_energy
```

	Hour	energy_consumpt_2005	energy_consumpt_2006	full_temp_2005	full_humid_200
0	1	631.623161	1246.300847	-0.400000	64.00000
1	2	534.397104	1062.500558	-0.733333	65.33333
2	3	453.538785	884.586887	-1.066667	66.66666
3	4	400.699718	786.564121	-1.400000	68.00000
4	5	378.171092	742.669614	-1.666667	60.33333
...	...	...	...	...	..
8779	8780	950.369306	0.000000	3.333333	64.00000
8780	8781	880.138770	0.000000	2.666667	68.00000
8781	8782	792.754026	0.000000	2.000000	72.00000
8782	8783	740.446668	0.000000	1.333333	76.00000
8783	8784	706.176769	0.000000	0.666667	80.00000

8784 rows × 7 columns

03.1 Na & Outliers

We will deal with Na and outliers so that we can get a better representation of the time series

```
#We will check for Na and interpolate it to get a close prediction of what that value coul
df_energy.isna().sum()
```

```
Hour                    0
energy_consumpt_2005    34
energy_consumpt_2006    42
full_temp_2005          0
full_humid_2005         0
full_temp_2006          24
full_humid_2006         24
dtype: int64
```

```
df_energy["energy_consumpt_2005"].fillna(value=df_energy["energy_consumpt_2005"].interpola
df_energy["energy_consumpt_2006"].fillna(value=df_energy["energy_consumpt_2006"].interpola
df_energy.isna().sum()
```

```
Hour                    0
energy_consumpt_2005    0
energy_consumpt_2006    0
full_temp_2005          0
full_humid_2005         0
full_temp_2006         24
full_humid_2006        24
dtype: int64
```

```
df_energy["energy_consumpt_2005"].plot()
df_energy["energy_consumpt_2006"].plot()
```

```
#Eliminate outliers by year since there is a big difference in values each year,
#The graphs for each year will be affected by this couple big outliers
up_limit06 = df_energy["energy_consumpt_2006"].mean() + 3*df_energy["energy_consumpt_2006"
low_limit06 = df_energy["energy_consumpt_2006"].mean() - 3*df_energy["energy_consumpt_2006
up_limit05 = df_energy["energy_consumpt_2005"].mean() + 3*df_energy["energy_consumpt_2005"
low_limit05 = df_energy["energy_consumpt_2005"].mean() - 3*df_energy["energy_consumpt_2005
print(up_limit05,low_limit05)
print(up_limit06,low_limit06)
print(df_energy[(df_energy["energy_consumpt_2006"] < low_limit06) | (df_energy["energy_cor
print(df_energy[(df_energy["energy_consumpt_2005"] < low_limit05) | (df_energy["energy_cor
df_energy = df_energy[(df_energy["energy_consumpt_2006"] > low_limit06) & (df_energy["ener
df_energy = df_energy[(df_energy["energy_consumpt_2005"] > low_limit05) & (df_energy["ener

```

```
991.0855739940082 -13.475213269908352
2207.6933705625834 84.29042652372891
      Hour  energy_consumpt_2005  energy_consumpt_2006  full_temp_2005  \
8779  8780            950.369306                   0.0        3.333333
8780  8781            880.138770                   0.0        2.666667
8781  8782            792.754026                   0.0        2.000000
8782  8783            740.446668                   0.0        1.333333
8783  8784            706.176769                   0.0        0.666667

      full_humid_2005  full_temp_2006  full_humid_2006
8779             64.0             NaN              NaN
8780             68.0             NaN              NaN
8781             72.0             NaN              NaN
8782             76.0             NaN              NaN
8783             80.0             NaN              NaN
      Hour  energy_consumpt_2005  energy_consumpt_2006  full_temp_2005  \
317    318           3378.521357            850.109412        4.933333
502    503           6364.455548           1325.593826        3.633333
638    639           -614.175490           1601.455847       16.933333
849    850           6560.013773           1369.922506        8.000000
8777  8778           1008.915444              0.000000        6.666667
```

```
      full_humid_2005  full_temp_2006  full_humid_2006
317         80.666667        6.600000        72.666667
502         88.000000        6.933333        82.666667
638         43.000000       14.733333        39.000000
849         30.000000        9.400000        53.000000
8777        50.333333             NaN             NaN
```

03.2 Time series

```
#We create a time series by adding 2006 data after 2005 data
df_energy_original = df_energy
df_energy06 = df_energy[["energy_consumpt_2006", "full_temp_2006", "full_humid_2006"]]
df_energy05 = df_energy[["energy_consumpt_2005", "full_temp_2005", "full_humid_2005"]]
df_energy_merged = df_energy05.append(df_energy06, ignore_index = True)
df_energy_merged.fillna(0, inplace=True)
df_energy_merged["Energy_05-06"] = df_energy_merged["energy_consumpt_2005"] + df_energy_me
df_energy_merged["Temp_05-06"] = df_energy_merged["full_temp_2005"] + df_energy_merged["fu
df_energy_merged["Humid_05-06"] = df_energy_merged["full_humid_2005"] + df_energy_merged['
df_energy_merged["Hour"] = df_energy_merged.index + 1
df_energy = df_energy_merged[["Hour","Energy_05-06", "Temp_05-06", "Humid_05-06"]]
df_energy.drop(df_energy.tail(24).index,inplace = True)
df_energy

```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:4913: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
```

```
# We create a time index dividing each value in days, weeks, month and years
df_energy["Year"]=2005
df_energy["Day"]= (((df_energy["Hour"]-1)/24)+1).apply(np.floor)
df_energy["Date"]=(np.asarray(df_energy['Year'], dtype='datetime64[Y]')-1970)+(np.asarray(
df_energy["Month"]=pd.DatetimeIndex(df_energy['Date']).month
df_energy.set_index("Date", inplace=True)
df_energy = df_energy[["Hour","Energy_05-06", "Temp_05-06", "Humid_05-06"]]
df_energy
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
```

```
# One table for each year
df_energy_05 = df_energy["2005":"2005"]
df_energy_05.rename(columns={"Energy_05-06":"Energy_05","Temp_05-06":"Temp_05","Humid_05-0
df_energy_06 = df_energy["2006":"2006"]
df_energy_06.rename(columns={"Energy_05-06":"Energy_06","Temp_05-06":"Temp_06","Humid_05-0
df_energy_06
```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:5047: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
  errors=errors,
```

Date	Hour	Energy_06	Temp_06	Humid_06
2006-01-01	8761	1227.253515	9.800000	59.000000
2006-01-01	8762	1374.030850	10.200000	56.000000
2006-01-01	8763	1444.229449	11.066667	53.000000
2006-01-01	8764	1443.072200	11.933333	50.000000
2006-01-01	8765	1421.878892	12.800000	47.000000
...	...	...	...	...
2006-12-30	17476	1875.387704	6.200000	69.333333
2006-12-30	17477	1854.660964	5.000000	74.666667
2006-12-30	17478	1754.937960	3.800000	80.000000
2006-12-30	17479	1522.868907	3.266667	84.000000
2006-12-30	17480	1254.040360	2.733333	88.000000

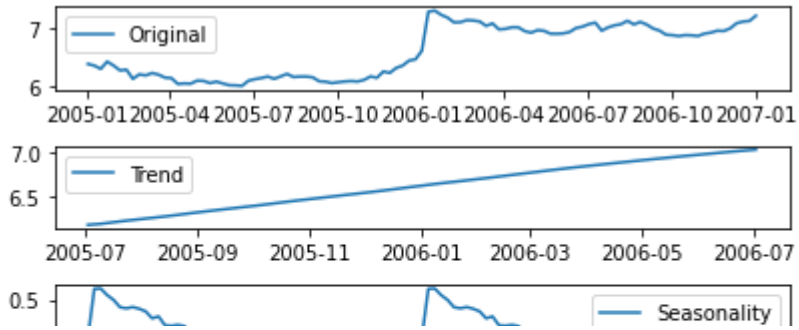
8720 rows × 4 columns

```
# Avarage the values from hourly to day, week and month
df_energyDay=df_energy.resample("D").mean()
df_energyWeek=df_energy.resample("W").mean()
df_energyMont=df_energy.resample("M").mean()
df_energyWeek
```

Date	Hour	Energy_05-06	Temp_05-06	Humid_05-06
2005-01-02	24.5	597.812417	5.268750	52.229167
2005-01-09	132.5	580.957300	11.107341	76.662698
2005-01-16	300.5	548.721333	10.871230	72.688492
2005-01-23	468.5	622.625542	2.938095	54.722222
2005-01-30	636.5	581.283938	8.793254	68.250000
...	...	...	...	...
2006-12-03	16764.5	1107.905150	9.755952	74.684524
2006-12-10	16932.5	1211.112521	8.797619	70.130952
2006-12-31	17416.5	1378.978959	5.507031	70.898438

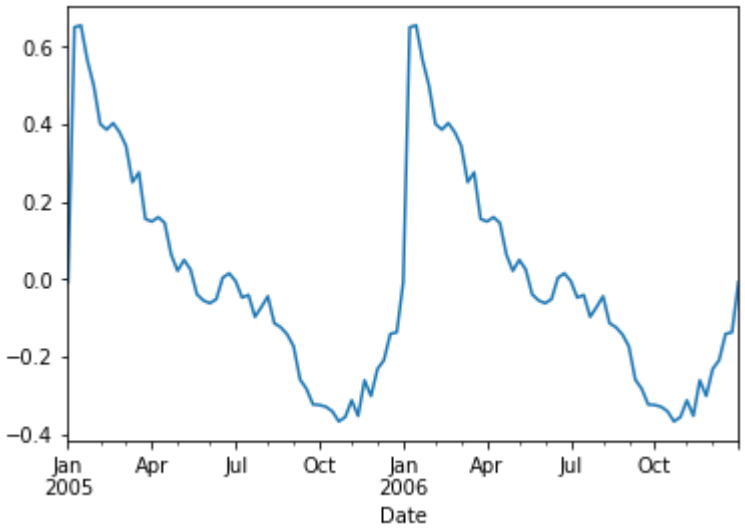
03.3 Forecast

```
#We selected per week since is a more accurate representation of the mean but still enough
#We analize our data dividing it by trend, seasonality and residuals
ts_log = np.log(df_energyWeek["Energy_05-06"])
decomposition = seasonal_decompose(ts_log)

trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid

plt.subplot(411)
plt.plot(ts_log, label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend, label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonality')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual, label='Residuals')
plt.legend(loc='best')
plt.tight_layout()
```




```
seasonal.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fefebefa910>
```




```
#Algorithm to find optimal SARIMAX values while we can asses for every value of q, d, p ar
#This will be a very long task so we can shrink this parameters based on a quick analisis
#Like setting seasonality for 52
#You can shrink the parameters the more quicker you want the code to run
#Or enlarge them the most thourough you want it to be
from math import sqrt
from multiprocessing import cpu_count
from joblib import Parallel
from joblib import delayed
from warnings import catch_warnings
from warnings import filterwarnings
from statsmodels.tsa.statespace.sarimax import SARIMAX
from sklearn.metrics import mean_squared_error
from pandas import read_csv
# one-step sarima forecast
def sarima_forecast(history, config):
    order, sorder, trend = config
    # define model
    model = SARIMAX(history, order=order, seasonal_order=sorder, trend=trend, enforce_stat
    # fit model
    model_fit = model.fit(disp=False)
    # make one step forecast
    yhat = model_fit.predict(len(history), len(history))
```

```
    return yhat[0]

# root mean squared error or rmse
def measure_rmse(actual, predicted):
    return sqrt(mean_squared_error(actual, predicted))

# split a univariate dataset into train/test sets
def train_test_split(data, n_test):
    return data[:-n_test], data[-n_test:]

# walk-forward validation for univariate data
def walk_forward_validation(data, n_test, cfg):
    predictions = list()
    # split dataset
    train, test = train_test_split(data, n_test)
    # seed history with training dataset
    history = [x for x in train]
    # step over each time-step in the test set
    for i in range(len(test)):
        # fit model and make forecast for history
        yhat = sarima_forecast(history, cfg)
        # store forecast in list of predictions
        predictions.append(yhat)
        # add actual observation to history for the next loop
        history.append(test[i])
    # estimate prediction error
    error = measure_rmse(test, predictions)
    return error

# score a model, return None on failure
def score_model(data, n_test, cfg, debug=False):
    result = None
    # convert config to a key
    key = str(cfg)
    # show all warnings and fail on exception if debugging
    if debug:
        result = walk_forward_validation(data, n_test, cfg)
    else:
        # one failure during model validation suggests an unstable config
        try:
            # never show warnings when grid searching, too noisy
            with catch_warnings():
                filterwarnings("ignore")
                result = walk_forward_validation(data, n_test, cfg)
        except:
            error = None
    # check for an interesting result
    if result is not None:
        print(' > Model[%s] %.3f' % (key, result))
    return (key, result)

```

```
# grid search configs
def grid_search(data, cfg_list, n_test, parallel=True):
    scores = None
    if parallel:
        # execute configs in parallel
        executor = Parallel(n_jobs=cpu_count(), backend='multiprocessing')
        tasks = (delayed(score_model)(data, n_test, cfg) for cfg in cfg_list)
        scores = executor(tasks)
    else:
        scores = [score_model(data, n_test, cfg) for cfg in cfg_list]
    # remove empty results
    scores = [r for r in scores if r[1] != None]
    # sort configs by error, asc
    scores.sort(key=lambda tup: tup[1])
    return scores

# create a set of sarima configs to try
def sarima_configs(seasonal=[0]):
    models = list()
    # define config lists
    p_params = [0, 1, 2]
    d_params = [0, 1]
    q_params = [0, 1, 2]
    t_params = ['n','c','t','ct']
    P_params = [0, 1, 2]
    D_params = [0, 1]
    Q_params = [0, 1, 2]
    m_params = seasonal
    # create config instances
    for p in p_params:
        for d in d_params:
            for q in q_params:
                for t in t_params:
                    for P in P_params:
                        for D in D_params:
                            for Q in Q_params:
                                for m in m_params:
                                    cfg = [(p,d,q), (P,D,Q,m), t]
                                    models.append(cfg)
    return models

if __name__ == '__main__':
    # load dataset
    data = df_energyWeek["Energy_05-06"]
    print(data.shape)
    # data split
    n_test = 12
    # model configs
    cfg_list = sarima_configs(seasonal=[13,52])
    # grid search
    scores = grid_search(data, cfg_list, n_test)
```

```
    print('done')
    # list top 3 configs
    for cfg, error in scores[:3]:
        print(cfg, error)
```

```
(105,)
 > Model[[(0, 0, 0), (0, 0, 0, 52), 'n']] 1116.328
 > Model[[(0, 0, 0), (0, 0, 0, 13), 'n']] 1116.328
 > Model[[(0, 0, 0), (0, 0, 1, 13), 'n']] 494.988
 > Model[[(0, 0, 0), (0, 1, 0, 13), 'n']] 199.506
 > Model[[(0, 0, 0), (0, 1, 0, 52), 'n']] 569.083
 > Model[[(0, 0, 0), (0, 1, 1, 13), 'n']] 176.467
 > Model[[(0, 0, 0), (0, 0, 2, 13), 'n']] 12553.307
 > Model[[(0, 0, 0), (1, 0, 0, 13), 'n']] 209.060
 > Model[[(0, 0, 0), (1, 0, 0, 52), 'n']] 188.948
 > Model[[(0, 0, 0), (1, 0, 1, 13), 'n']] 294.836
 > Model[[(0, 0, 0), (1, 1, 0, 13), 'n']] 167.702
 > Model[[(0, 0, 0), (1, 0, 2, 13), 'n']] 697652242.139
 > Model[[(0, 0, 0), (1, 1, 1, 13), 'n']] 165.412
 > Model[[(0, 0, 0), (2, 0, 0, 13), 'n']] 213.899
 > Model[[(0, 0, 0), (2, 0, 1, 13), 'n']] 327.606
 > Model[[(0, 0, 0), (2, 1, 0, 13), 'n']] 158.717
 > Model[[(0, 0, 0), (2, 0, 2, 13), 'n']] 309690394.119
 > Model[[(0, 0, 0), (2, 1, 1, 13), 'n']] 178.670
 > Model[[(0, 0, 0), (0, 0, 0, 13), 'c']] 337.229
 > Model[[(0, 0, 0), (0, 0, 0, 52), 'c']] 337.229
 > Model[[(0, 0, 0), (0, 0, 1, 13), 'c']] 239.540
 > Model[[(0, 0, 0), (0, 1, 0, 13), 'c']] 229.072
 > Model[[(0, 0, 0), (0, 0, 2, 13), 'c']] 4900.259
 > Model[[(0, 0, 0), (0, 1, 0, 52), 'c']] 111.675
 > Model[[(0, 0, 0), (0, 1, 1, 13), 'c']] 239.682
 > Model[[(0, 0, 0), (1, 0, 0, 13), 'c']] 175.479
 > Model[[(0, 0, 0), (1, 0, 0, 52), 'c']] 107.532
 > Model[[(0, 0, 0), (1, 0, 1, 13), 'c']] 155.214
 > Model[[(0, 0, 0), (0, 1, 2, 13), 'c']] 165154.750
 > Model[[(0, 0, 0), (1, 1, 0, 13), 'c']] 223.793
 > Model[[(0, 0, 0), (1, 1, 1, 13), 'c']] 249.031
 > Model[[(0, 0, 0), (1, 0, 2, 13), 'c']] 8662762181.486
 > Model[[(0, 0, 0), (2, 0, 0, 13), 'c']] 152.464
 > Model[[(0, 0, 0), (2, 0, 1, 13), 'c']] 156.991
 > Model[[(0, 0, 0), (2, 0, 2, 13), 'c']] 1219936709.460
 > Model[[(0, 0, 0), (1, 1, 2, 13), 'c']] 164258.123
 > Model[[(0, 0, 0), (2, 1, 0, 13), 'c']] 344.780
 > Model[[(0, 0, 0), (2, 1, 1, 13), 'c']] 329.567
 > Model[[(0, 0, 0), (0, 0, 0, 13), 't']] 337.429
 > Model[[(0, 0, 0), (0, 0, 0, 52), 't']] 337.429
 > Model[[(0, 0, 0), (0, 0, 1, 13), 't']] 332.561
 > Model[[(0, 0, 0), (0, 0, 2, 13), 't']] 2455.257
 > Model[[(0, 0, 0), (0, 1, 0, 13), 't']] 257.410
 > Model[[(0, 0, 0), (2, 1, 2, 13), 'c']] 21246831.379
 > Model[[(0, 0, 0), (0, 1, 1, 13), 't']] 328.568
 > Model[[(0, 0, 0), (0, 1, 0, 52), 't']] 272.550
 > Model[[(0, 0, 0), (1, 0, 0, 13), 't']] 328.520
 > Model[[(0, 0, 0), (1, 0, 0, 52), 't']] 203.527
 > Model[[(0, 0, 0), (1, 0, 1, 13), 't']] 361.099
 > Model[[(0, 0, 0), (0, 1, 2, 13), 't']] 107573.187
 > Model[[(0, 0, 0), (1, 1, 0, 13), 't']] 266.798
 > Model[[(0, 0, 0), (1, 0, 2, 13), 't']] 3599957634.992
 > Model[[(0, 0, 0), (1, 1, 1, 13), 't']] 337.244
 > Model[[(0, 0, 0), (2, 0, 0, 13), 't']] 349.716
 > Model[[(0, 0, 0), (2, 0, 1, 13), 't']] 342.303
```

```
> Model[[(0, 0, 0), (2, 0, 2, 13), 't']] 231431042.070
> Model[[(0, 0, 0), (1, 1, 2, 13), 't']] 180963.284
> Model[[(0, 0, 0), (2, 1, 0, 13), 't']] 432.453
> Model[[(0, 0, 0), (2, 1, 1, 13), 't']] 446.198
> Model[[(0, 0, 0), (0, 0, 0, 13), 'ct']] 193.305
> Model[[(0, 0, 0), (0, 0, 0, 52), 'ct']] 193.305
> Model[[(0, 0, 0), (0, 0, 1, 13), 'ct']] 233.009
> Model[[(0, 0, 0), (0, 0, 2, 13), 'ct']] 1238.514
> Model[[(0, 0, 0), (0, 1, 0, 13), 'ct']] 244.498
> Model[[(0, 0, 0), (2, 1, 2, 13), 't']] 27416303.783
> Model[[(0, 0, 0), (0, 1, 1, 13), 'ct']] 221.505
> Model[[(0, 0, 0), (0, 1, 0, 52), 'ct']] 55.252
> Model[[(0, 0, 0), (1, 0, 0, 13), 'ct']] 243.677
> Model[[(0, 0, 0), (0, 1, 2, 13), 'ct']] 145419.055
> Model[[(0, 0, 0), (1, 0, 1, 13), 'ct']] 253.025
> Model[[(0, 0, 0), (1, 0, 0, 52), 'ct']] 71.908
> Model[[(0, 0, 0), (1, 1, 0, 13), 'ct']] 203.114
> Model[[(0, 0, 0), (1, 0, 2, 13), 'ct']] 5915097769.754
> Model[[(0, 0, 0), (1, 1, 1, 13), 'ct']] 231.674
> Model[[(0, 0, 0), (2, 0, 0, 13), 'ct']] 278.532
> Model[[(0, 0, 0), (2, 0, 1, 13), 'ct']] 124.329
> Model[[(0, 0, 0), (2, 0, 2, 13), 'ct']] 678445174.000
> Model[[(0, 0, 0), (1, 1, 2, 13), 'ct']] 160979.153
> Model[[(0, 0, 0), (2, 1, 0, 13), 'ct']] 222.120
> Model[[(0, 0, 0), (2, 1, 1, 13), 'ct']] 284.668
> Model[[(0, 0, 1), (0, 0, 0, 13), 'n']] 588.629
> Model[[(0, 0, 1), (0, 0, 0, 52), 'n']] 588.629
> Model[[(0, 0, 1), (0, 0, 1, 13), 'n']] 331.839
> Model[[(0, 0, 1), (0, 0, 2, 13), 'n']] 836772280.684
> Model[[(0, 0, 1), (0, 1, 0, 13), 'n']] 134.559
> Model[[(0, 0, 1), (0, 1, 0, 52), 'n']] 292.771
> Model[[(0, 0, 1), (0, 1, 1, 13), 'n']] 111.154
> Model[[(0, 0, 0), (2, 1, 2, 13), 'ct']] 25545550.006
> Model[[(0, 0, 1), (1, 0, 0, 13), 'n']] 136.418
> Model[[(0, 0, 1), (0, 1, 2, 13), 'n']] 136076164.819
> Model[[(0, 0, 1), (1, 0, 1, 13), 'n']] 145.533
> Model[[(0, 0, 1), (1, 0, 2, 13), 'n']] 1477371372.464
> Model[[(0, 0, 1), (1, 1, 0, 13), 'n']] 104.297
> Model[[(0, 0, 1), (1, 0, 0, 52), 'n']] 131.469
> Model[[(0, 0, 1), (1, 1, 1, 13), 'n']] 113.798
> Model[[(0, 0, 1), (2, 0, 0, 13), 'n']] 118.243
> Model[[(0, 0, 1), (1, 1, 2, 13), 'n']] 1019158134.208
> Model[[(0, 0, 1), (2, 0, 1, 13), 'n']] 151.587
> Model[[(0, 0, 1), (2, 0, 2, 13), 'n']] 263058262.763
> Model[[(0, 0, 1), (2, 1, 0, 13), 'n']] 103.172
> Model[[(0, 0, 1), (2, 1, 1, 13), 'n']] 112.557
> Model[[(0, 0, 1), (0, 0, 0, 13), 'c']] 192.470
> Model[[(0, 0, 1), (0, 0, 0, 52), 'c']] 192.470
> Model[[(0, 0, 1), (0, 0, 1, 13), 'c']] 159.144
> Model[[(0, 0, 1), (2, 1, 2, 13), 'n']] 118169566.241
> Model[[(0, 0, 1), (0, 1, 0, 13), 'c']] 144.291
> Model[[(0, 0, 1), (0, 0, 2, 13), 'c']] 3609.089
> Model[[(0, 0, 1), (0, 1, 0, 52), 'c']] 64.744
> Model[[(0, 0, 1), (0, 1, 1, 13), 'c']] 127.705
> Model[[(0, 0, 1), (1, 0, 0, 13), 'c']] 119.875
> Model[[(0, 0, 1), (0, 1, 2, 13), 'c']] 542665441.979
> Model[[(0, 0, 1), (1, 0, 1, 13), 'c']] 109.865
```

```
> Model[[(0, 0, 1), (1, 0, 1, 13), 'c']] 109.865
> Model[[(0, 0, 1), (1, 0, 0, 52), 'c']] 56.643
> Model[[(0, 0, 1), (1, 1, 0, 13), 'c']] 124.333
> Model[[(0, 0, 1), (1, 0, 2, 13), 'c']] 6614710114.165
> Model[[(0, 0, 1), (1, 1, 1, 13), 'c']] 106.132
> Model[[(0, 0, 1), (2, 0, 0, 13), 'c']] 99.179
> Model[[(0, 0, 1), (2, 0, 1, 13), 'c']] 107.714
> Model[[(0, 0, 1), (1, 1, 2, 13), 'c']] 88192.619
> Model[[(0, 0, 1), (2, 0, 2, 13), 'c']] 1163229571.398
> Model[[(0, 0, 1), (2, 1, 0, 13), 'c']] 165.922
> Model[[(0, 0, 1), (2, 1, 1, 13), 'c']] 136.996
> Model[[(0, 0, 1), (0, 0, 0, 13), 't']] 177.299
> Model[[(0, 0, 1), (0, 0, 0, 52), 't']] 177.299
> Model[[(0, 0, 1), (0, 0, 1, 13), 't']] 191.527
> Model[[(0, 0, 1), (2, 1, 2, 13), 'c']] 3095317.174
> Model[[(0, 0, 1), (0, 0, 2, 13), 't']] 1510.649
> Model[[(0, 0, 1), (0, 1, 0, 13), 't']] 155.231
> Model[[(0, 0, 1), (0, 1, 1, 13), 't']] 171.026
> Model[[(0, 0, 1), (0, 1, 0, 52), 't']] 145.354
> Model[[(0, 0, 1), (1, 0, 0, 13), 't']] 177.103
> Model[[(0, 0, 1), (0, 1, 2, 13), 't']] 579128504.588
> Model[[(0, 0, 1), (1, 0, 1, 13), 't']] 187.613
> Model[[(0, 0, 1), (1, 0, 2, 13), 't']] 3412880724.893
> Model[[(0, 0, 1), (1, 0, 0, 52), 't']] 147.716
> Model[[(0, 0, 1), (1, 1, 0, 13), 't']] 146.659
> Model[[(0, 0, 1), (1, 1, 1, 13), 't']] 123.616
> Model[[(0, 0, 1), (2, 0, 0, 13), 't']] 186.200
> Model[[(0, 0, 1), (1, 1, 2, 13), 't']] 565854558.454
> Model[[(0, 0, 1), (2, 0, 1, 13), 't']] 188.295
> Model[[(0, 0, 1), (2, 0, 2, 13), 't']] 349956789.744
> Model[[(0, 0, 1), (2, 1, 0, 13), 't']] 204.561
> Model[[(0, 0, 1), (2, 1, 1, 13), 't']] 215.657
> Model[[(0, 0, 1), (0, 0, 0, 13), 'ct']] 106.827
> Model[[(0, 0, 1), (0, 0, 0, 52), 'ct']] 106.827
> Model[[(0, 0, 1), (0, 0, 1, 13), 'ct']] 119.856
> Model[[(0, 0, 1), (2, 1, 2, 13), 't']] 199740017.843
> Model[[(0, 0, 1), (0, 1, 0, 13), 'ct']] 149.275
> Model[[(0, 0, 1), (0, 0, 2, 13), 'ct']] 992.390
> Model[[(0, 0, 1), (0, 1, 1, 13), 'ct']] 124.021
> Model[[(0, 0, 1), (0, 1, 0, 52), 'ct']] 40.587
> Model[[(0, 0, 1), (1, 0, 0, 13), 'ct']] 117.855
> Model[[(0, 0, 1), (0, 1, 2, 13), 'ct']] 5974062.236
> Model[[(0, 0, 1), (1, 0, 1, 13), 'ct']] 112.140
> Model[[(0, 0, 1), (1, 0, 2, 13), 'ct']] 4724818353.961
> Model[[(0, 0, 1), (1, 0, 0, 52), 'ct']] 82.902
> Model[[(0, 0, 1), (1, 1, 0, 13), 'ct']] 129.362
> Model[[(0, 0, 1), (1, 1, 1, 13), 'ct']] 103.392
> Model[[(0, 0, 1), (2, 0, 0, 13), 'ct']] 118.239
> Model[[(0, 0, 1), (1, 1, 2, 13), 'ct']] 279629919.305
> Model[[(0, 0, 1), (2, 0, 1, 13), 'ct']] 94.871
> Model[[(0, 0, 1), (2, 0, 2, 13), 'ct']] 1168206371.487
> Model[[(0, 0, 1), (2, 1, 0, 13), 'ct']] 149.027
> Model[[(0, 0, 1), (2, 1, 1, 13), 'ct']] 181.417
> Model[[(0, 0, 2), (0, 0, 0, 13), 'n']] 357.652
> Model[[(0, 0, 2), (0, 0, 0, 52), 'n']] 357.652
> Model[[(0, 0, 2), (0, 0, 1, 13), 'n']] 238.735
> Model[[(0, 0, 1), (2, 1, 2, 13), 'ct']] 29741860.686
```

```
> Model[[(0, 0, 2), (0, 1, 0, 13), 'n']] 131.729
> Model[[(0, 0, 2), (0, 0, 2, 13), 'n']] 20289297.384
> Model[[(0, 0, 2), (0, 1, 1, 13), 'n']] 107.275
> Model[[(0, 0, 2), (0, 1, 0, 52), 'n']] 192.810
> Model[[(0, 0, 2), (1, 0, 0, 13), 'n']] 125.361
> Model[[(0, 0, 2), (0, 1, 2, 13), 'n']] 136468278577.467
> Model[[(0, 0, 2), (1, 0, 1, 13), 'n']] 101.284
> Model[[(0, 0, 2), (1, 0, 2, 13), 'n']] 102741600923006.234
> Model[[(0, 0, 2), (1, 1, 0, 13), 'n']] 99.739
> Model[[(0, 0, 2), (1, 0, 0, 52), 'n']] 136.213
> Model[[(0, 0, 2), (1, 1, 1, 13), 'n']] 103.888
> Model[[(0, 0, 2), (2, 0, 0, 13), 'n']] 100.678
> Model[[(0, 0, 2), (2, 0, 1, 13), 'n']] 114.895
> Model[[(0, 0, 2), (1, 1, 2, 13), 'n']] 3341061.033
> Model[[(0, 0, 2), (2, 0, 2, 13), 'n']] 33228543133129076.000
> Model[[(0, 0, 2), (2, 1, 0, 13), 'n']] 98.178
> Model[[(0, 0, 2), (2, 1, 1, 13), 'n']] 95.549
> Model[[(0, 0, 2), (0, 0, 0, 13), 'c']] 137.818
> Model[[(0, 0, 2), (0, 0, 0, 52), 'c']] 137.818
> Model[[(0, 0, 2), (0, 0, 1, 13), 'c']] 116.844
> Model[[(0, 0, 2), (2, 1, 2, 13), 'n']] 20160386.507
> Model[[(0, 0, 2), (0, 1, 0, 13), 'c']] 132.259
> Model[[(0, 0, 2), (0, 0, 2, 13), 'c']] 1823.161
> Model[[(0, 0, 2), (0, 1, 1, 13), 'c']] 93.913
> Model[[(0, 0, 2), (0, 1, 0, 52), 'c']] 57.290
> Model[[(0, 0, 2), (1, 0, 0, 13), 'c']] 105.473
> Model[[(0, 0, 2), (0, 1, 2, 13), 'c']] 36238503.472
> Model[[(0, 0, 2), (1, 0, 1, 13), 'c']] 91.867
> Model[[(0, 0, 2), (1, 0, 0, 52), 'c']] 50.359
> Model[[(0, 0, 2), (1, 0, 2, 13), 'c']] 10054251401.421
> Model[[(0, 0, 2), (1, 1, 0, 13), 'c']] 101.038
> Model[[(0, 0, 2), (1, 1, 1, 13), 'c']] 101.488
> Model[[(0, 0, 2), (2, 0, 0, 13), 'c']] 82.664
> Model[[(0, 0, 2), (2, 0, 1, 13), 'c']] 89.288
> Model[[(0, 0, 2), (1, 1, 2, 13), 'c']] 452217177154.043
> Model[[(0, 0, 2), (2, 0, 2, 13), 'c']] 8000418709518639169536.000
> Model[[(0, 0, 2), (2, 1, 0, 13), 'c']] 105.133
> Model[[(0, 0, 2), (2, 1, 1, 13), 'c']] 89.916
> Model[[(0, 0, 2), (0, 0, 0, 13), 't']] 123.161
> Model[[(0, 0, 2), (0, 0, 0, 52), 't']] 123.161
> Model[[(0, 0, 2), (0, 0, 1, 13), 't']] 127.972
> Model[[(0, 0, 2), (0, 0, 2, 13), 't']] 435.297
> Model[[(0, 0, 2), (0, 1, 0, 13), 't']] 137.999
> Model[[(0, 0, 2), (2, 1, 2, 13), 'c']] 88702095.172
> Model[[(0, 0, 2), (0, 1, 1, 13), 't']] 117.619
> Model[[(0, 0, 2), (0, 1, 0, 52), 't']] 110.336
> Model[[(0, 0, 2), (1, 0, 0, 13), 't']] 123.956
> Model[[(0, 0, 2), (0, 1, 2, 13), 't']] 155411178215771.062
> Model[[(0, 0, 2), (1, 0, 1, 13), 't']] 129.134
> Model[[(0, 0, 2), (1, 0, 2, 13), 't']] 3924256161.449
> Model[[(0, 0, 2), (1, 1, 0, 13), 't']] 112.206
> Model[[(0, 0, 2), (1, 0, 0, 52), 't']] 148.812
> Model[[(0, 0, 2), (1, 1, 1, 13), 't']] 119.871
> Model[[(0, 0, 2), (2, 0, 0, 13), 't']] 130.351
> Model[[(0, 0, 2), (2, 0, 1, 13), 't']] 113.772
> Model[[(0, 0, 2), (1, 1, 2, 13), 't']] 142762.371
> Model[[(0, 0, 2), (2, 0, 2, 13), 't']] 150520080.042
```

```
> Model[[(0, 0, 2), (2, 0, 2, 13), 't']] 150529089.042
> Model[[(0, 0, 2), (2, 1, 0, 13), 't']] 136.534
> Model[[(0, 0, 2), (2, 1, 1, 13), 't']] 141.521
> Model[[(0, 0, 2), (0, 0, 0, 13), 'ct']] 82.330
> Model[[(0, 0, 2), (0, 0, 0, 52), 'ct']] 82.330
> Model[[(0, 0, 2), (0, 0, 1, 13), 'ct']] 99.946
> Model[[(0, 0, 2), (2, 1, 2, 13), 't']] 55726716.427
> Model[[(0, 0, 2), (0, 1, 0, 13), 'ct']] 136.695
> Model[[(0, 0, 2), (0, 0, 2, 13), 'ct']] 1281.892
> Model[[(0, 0, 2), (0, 1, 1, 13), 'ct']] 82.651
> Model[[(0, 0, 2), (0, 1, 0, 52), 'ct']] 38.533
> Model[[(0, 0, 2), (1, 0, 0, 13), 'ct']] 88.539
> Model[[(0, 0, 2), (0, 1, 2, 13), 'ct']] 573788781.207
> Model[[(0, 0, 2), (1, 0, 1, 13), 'ct']] 85.382
> Model[[(0, 0, 2), (1, 0, 2, 13), 'ct']] 54809111.240
> Model[[(0, 0, 2), (1, 1, 0, 13), 'ct']] 106.191
> Model[[(0, 0, 2), (1, 0, 0, 52), 'ct']] 138.769
> Model[[(0, 0, 2), (1, 1, 1, 13), 'ct']] 95.654
> Model[[(0, 0, 2), (2, 0, 0, 13), 'ct']] 88.562
> Model[[(0, 0, 2), (2, 0, 1, 13), 'ct']] 73.950
> Model[[(0, 0, 2), (2, 0, 2, 13), 'ct']] 6455827983240107130 88.000
> Model[[(0, 0, 2), (1, 1, 2, 13), 'ct']] 2050993543388071424.000
> Model[[(0, 0, 2), (2, 1, 0, 13), 'ct']] 128.219
> Model[[(0, 0, 2), (2, 1, 1, 13), 'ct']] 121.612
> Model[[(0, 1, 0), (0, 0, 0, 13), 'n']] 51.878
> Model[[(0, 1, 0), (0, 0, 0, 52), 'n']] 51.878
> Model[[(0, 1, 0), (0, 0, 1, 13), 'n']] 52.274
> Model[[(0, 1, 0), (0, 0, 2, 13), 'n']] 49026.228
> Model[[(0, 1, 0), (0, 1, 0, 13), 'n']] 104.136
> Model[[(0, 1, 0), (0, 1, 0, 52), 'n']] 27.552
> Model[[(0, 1, 0), (0, 1, 1, 13), 'n']] 53.917
> Model[[(0, 1, 0), (1, 0, 0, 13), 'n']] 52.242
> Model[[(0, 1, 0), (1, 0, 0, 52), 'n']] 63.652
> Model[[(0, 1, 0), (1, 0, 1, 13), 'n']] 52.188
> Model[[(0, 0, 2), (2, 1, 2, 13), 'ct']] 167696225649.585
> Model[[(0, 1, 0), (1, 1, 0, 13), 'n']] 65.283
> Model[[(0, 1, 0), (1, 1, 1, 13), 'n']] 52.117
> Model[[(0, 1, 0), (1, 0, 2, 13), 'n']] 8102121.785
> Model[[(0, 1, 0), (2, 0, 0, 13), 'n']] 51.242
> Model[[(0, 1, 0), (2, 0, 1, 13), 'n']] 51.127
> Model[[(0, 1, 0), (2, 0, 2, 13), 'n']] 79360899.393
> Model[[(0, 1, 0), (2, 1, 0, 13), 'n']] 64.554
> Model[[(0, 1, 0), (2, 1, 1, 13), 'n']] 49.531
> Model[[(0, 1, 0), (0, 0, 0, 13), 'c']] 48.381
> Model[[(0, 1, 0), (0, 0, 0, 52), 'c']] 48.381
> Model[[(0, 1, 0), (0, 0, 1, 13), 'c']] 46.993
> Model[[(0, 1, 0), (0, 0, 2, 13), 'c']] 37866.392
> Model[[(0, 1, 0), (0, 1, 0, 13), 'c']] 103.470
> Model[[(0, 1, 0), (0, 1, 1, 13), 'c']] 51.307
> Model[[(0, 1, 0), (0, 1, 0, 52), 'c']] 30.824
> Model[[(0, 1, 0), (1, 0, 0, 13), 'c']] 47.336
> Model[[(0, 1, 0), (1, 0, 0, 52), 'c']] 58.882
> Model[[(0, 1, 0), (1, 0, 1, 13), 'c']] 46.947
> Model[[(0, 1, 0), (1, 0, 2, 13), 'c']] 4749531.669
> Model[[(0, 1, 0), (1, 1, 0, 13), 'c']] 64.744
> Model[[(0, 1, 0), (0, 1, 2, 13), 'c']] 31916189.465
> Model[[(0, 1, 0), (1, 1, 1, 13), 'c']] 49.786
```

```
> Model[[(0, 1, 0), (2, 0, 0, 13), 'c']] 45.711
> Model[[(0, 1, 0), (2, 0, 1, 13), 'c']] 45.690
> Model[[(0, 1, 0), (2, 0, 2, 13), 'c']] 63191871.446
> Model[[(0, 1, 0), (2, 1, 0, 13), 'c']] 64.454
> Model[[(0, 1, 0), (1, 1, 2, 13), 'c']] 4530457.702
> Model[[(0, 1, 0), (2, 1, 1, 13), 'c']] 50.732
> Model[[(0, 1, 0), (0, 0, 0, 13), 't']] 45.664
> Model[[(0, 1, 0), (0, 0, 0, 52), 't']] 45.664
> Model[[(0, 1, 0), (0, 0, 1, 13), 't']] 45.721
> Model[[(0, 1, 0), (0, 0, 2, 13), 't']] 36732.963
> Model[[(0, 1, 0), (0, 1, 0, 13), 't']] 105.656
> Model[[(0, 1, 0), (2, 1, 2, 13), 'c']] 8415.708
> Model[[(0, 1, 0), (0, 1, 1, 13), 't']] 54.109
> Model[[(0, 1, 0), (0, 1, 0, 52), 't']] 31.468
> Model[[(0, 1, 0), (1, 0, 0, 13), 't']] 45.754
> Model[[(0, 1, 0), (1, 0, 0, 52), 't']] 59.976
> Model[[(0, 1, 0), (1, 0, 1, 13), 't']] 45.587
> Model[[(0, 1, 0), (0, 1, 2, 13), 't']] 32551318.018
> Model[[(0, 1, 0), (1, 0, 2, 13), 't']] 11019361.486
> Model[[(0, 1, 0), (1, 1, 0, 13), 't']] 67.940
> Model[[(0, 1, 0), (1, 1, 1, 13), 't']] 52.734
> Model[[(0, 1, 0), (2, 0, 0, 13), 't']] 45.194
> Model[[(0, 1, 0), (2, 0, 1, 13), 't']] 45.104
> Model[[(0, 1, 0), (2, 0, 2, 13), 't']] 71767514.535
> Model[[(0, 1, 0), (1, 1, 2, 13), 't']] 5130922.223
> Model[[(0, 1, 0), (2, 1, 0, 13), 't']] 69.353
> Model[[(0, 1, 0), (2, 1, 1, 13), 't']] 54.588
> Model[[(0, 1, 0), (0, 0, 0, 13), 'ct']] 46.226
> Model[[(0, 1, 0), (0, 0, 0, 52), 'ct']] 46.226
> Model[[(0, 1, 0), (0, 0, 1, 13), 'ct']] 49.854
> Model[[(0, 1, 0), (0, 0, 2, 13), 'ct']] 24592.342
> Model[[(0, 1, 0), (0, 1, 0, 13), 'ct']] 111.130
> Model[[(0, 1, 0), (2, 1, 2, 13), 't']] 372.905
> Model[[(0, 1, 0), (0, 1, 1, 13), 'ct']] 72.659
> Model[[(0, 1, 0), (0, 1, 0, 52), 'ct']] 28.226
> Model[[(0, 1, 0), (1, 0, 0, 13), 'ct']] 49.088
> Model[[(0, 1, 0), (0, 1, 2, 13), 'ct']] 20340779.628
> Model[[(0, 1, 0), (1, 0, 0, 52), 'ct']] 73.718
> Model[[(0, 1, 0), (1, 0, 1, 13), 'ct']] 49.906
> Model[[(0, 1, 0), (1, 1, 0, 13), 'ct']] 78.217
> Model[[(0, 1, 0), (1, 1, 1, 13), 'ct']] 72.062
> Model[[(0, 1, 0), (1, 0, 2, 13), 'ct']] 84078380.273
> Model[[(0, 1, 0), (2, 0, 0, 13), 'ct']] 52.663
> Model[[(0, 1, 0), (2, 0, 1, 13), 'ct']] 52.948
> Model[[(0, 1, 0), (1, 1, 2, 13), 'ct']] 4656971.962
> Model[[(0, 1, 0), (2, 0, 2, 13), 'ct']] 72797217.786
> Model[[(0, 1, 0), (2, 1, 0, 13), 'ct']] 95.086
> Model[[(0, 1, 0), (2, 1, 1, 13), 'ct']] 81.224
> Model[[(0, 1, 1), (0, 0, 0, 13), 'n']] 49.744
> Model[[(0, 1, 1), (0, 0, 0, 52), 'n']] 49.744
> Model[[(0, 1, 1), (0, 0, 1, 13), 'n']] 50.081
> Model[[(0, 1, 1), (0, 0, 2, 13), 'n']] 48177.606
> Model[[(0, 1, 1), (0, 1, 0, 13), 'n']] 104.421
> Model[[(0, 1, 0), (2, 1, 2, 13), 'ct']] 263.381
> Model[[(0, 1, 1), (0, 1, 0, 52), 'n']] 28.160
> Model[[(0, 1, 1), (0, 1, 1, 13), 'n']] 53.090
```

```
> Model[[(0, 1, 1), (1, 0, 0, 13), 'n']] 50.133
> Model[[(0, 1, 1), (1, 0, 0, 52), 'n']] 53.624
> Model[[(0, 1, 1), (1, 0, 1, 13), 'n']] 50.004
> Model[[(0, 1, 1), (0, 1, 2, 13), 'n']] 23298407.785
> Model[[(0, 1, 1), (1, 1, 0, 13), 'n']] 65.150
> Model[[(0, 1, 1), (1, 0, 2, 13), 'n']] 10887506.403
> Model[[(0, 1, 1), (1, 1, 1, 13), 'n']] 51.620
> Model[[(0, 1, 1), (2, 0, 0, 13), 'n']] 48.901
> Model[[(0, 1, 1), (2, 0, 1, 13), 'n']] 48.752
> Model[[(0, 1, 1), (2, 0, 2, 13), 'n']] 83113292.996
> Model[[(0, 1, 1), (2, 1, 0, 13), 'n']] 66.022
> Model[[(0, 1, 1), (1, 1, 2, 13), 'n']] 7075330.590
> Model[[(0, 1, 1), (2, 1, 1, 13), 'n']] 48.700
> Model[[(0, 1, 1), (0, 0, 0, 13), 'c']] 46.646
> Model[[(0, 1, 1), (0, 0, 0, 52), 'c']] 46.646
> Model[[(0, 1, 1), (0, 0, 1, 13), 'c']] 45.871
> Model[[(0, 1, 1), (0, 0, 2, 13), 'c']] 40718.374
> Model[[(0, 1, 1), (0, 1, 0, 13), 'c']] 104.018
> Model[[(0, 1, 1), (2, 1, 2, 13), 'n']] 235661.183
> Model[[(0, 1, 1), (0, 1, 1, 13), 'c']] 53.524
> Model[[(0, 1, 1), (0, 1, 0, 52), 'c']] 33.228
> Model[[(0, 1, 1), (1, 0, 0, 13), 'c']] 46.231
> Model[[(0, 1, 1), (0, 1, 2, 13), 'c']] 112540764.254
> Model[[(0, 1, 1), (1, 0, 0, 52), 'c']] 50.424
> Model[[(0, 1, 1), (1, 0, 1, 13), 'c']] 45.822
> Model[[(0, 1, 1), (1, 1, 0, 13), 'c']] 64.817
> Model[[(0, 1, 1), (1, 1, 1, 13), 'c']] 52.087
> Model[[(0, 1, 1), (1, 0, 2, 13), 'c']] 6497188.599
> Model[[(0, 1, 1), (2, 0, 0, 13), 'c']] 44.586
> Model[[(0, 1, 1), (2, 0, 1, 13), 'c']] 44.530
> Model[[(0, 1, 1), (2, 0, 2, 13), 'c']] 545943394.958
> Model[[(0, 1, 1), (1, 1, 2, 13), 'c']] 5704142.805
> Model[[(0, 1, 1), (2, 1, 0, 13), 'c']] 65.986
> Model[[(0, 1, 1), (2, 1, 1, 13), 'c']] 49.404
> Model[[(0, 1, 1), (0, 0, 0, 13), 't']] 44.630
> Model[[(0, 1, 1), (0, 0, 0, 52), 't']] 44.630
> Model[[(0, 1, 1), (0, 0, 1, 13), 't']] 44.882
> Model[[(0, 1, 1), (0, 0, 2, 13), 't']] 4418.591
> Model[[(0, 1, 1), (0, 1, 0, 13), 't']] 105.959
> Model[[(0, 1, 1), (2, 1, 2, 13), 'c']] 6766.096
> Model[[(0, 1, 1), (0, 1, 0, 52), 't']] 34.629
> Model[[(0, 1, 1), (0, 1, 1, 13), 't']] 55.707
> Model[[(0, 1, 1), (1, 0, 0, 13), 't']] 44.989
> Model[[(0, 1, 1), (1, 0, 0, 52), 't']] 50.427
> Model[[(0, 1, 1), (1, 0, 1, 13), 't']] 44.770
> Model[[(0, 1, 1), (0, 1, 2, 13), 't']] 425877212.245
> Model[[(0, 1, 1), (1, 1, 0, 13), 't']] 67.229
> Model[[(0, 1, 1), (1, 0, 2, 13), 't']] 13482601.827
> Model[[(0, 1, 1), (1, 1, 1, 13), 't']] 54.385
> Model[[(0, 1, 1), (2, 0, 0, 13), 't']] 44.247
> Model[[(0, 1, 1), (2, 0, 1, 13), 't']] 44.119
> Model[[(0, 1, 1), (2, 0, 2, 13), 't']] 73396031.425
> Model[[(0, 1, 1), (1, 1, 2, 13), 't']] 6791810.399
> Model[[(0, 1, 1), (2, 1, 0, 13), 't']] 70.012
> Model[[(0, 1, 1), (2, 1, 1, 13), 't']] 52.636
> Model[[(0, 1, 1), (0, 0, 0, 13), 'ct']] 45.496
> Model[[(0, 1, 1), (0, 0, 0, 52), 'ct']] 45.496
```

```
> Model[[(0, 1, 1), (0, 0, 1, 13), 'ct']] 48.357
> Model[[(0, 1, 1), (0, 0, 2, 13), 'ct']] 45780.054
> Model[[(0, 1, 1), (2, 1, 2, 13), 't']] 6780.934
> Model[[(0, 1, 1), (0, 1, 0, 13), 'ct']] 110.090
> Model[[(0, 1, 1), (0, 1, 1, 13), 'ct']] 66.570
> Model[[(0, 1, 1), (0, 1, 0, 52), 'ct']] 29.806
> Model[[(0, 1, 1), (1, 0, 0, 13), 'ct']] 47.780
> Model[[(0, 1, 1), (0, 1, 2, 13), 'ct']] 18224079.884
> Model[[(0, 1, 1), (1, 0, 1, 13), 'ct']] 48.364
> Model[[(0, 1, 1), (1, 0, 0, 52), 'ct']] 69.190
> Model[[(0, 1, 1), (1, 1, 0, 13), 'ct']] 73.878
> Model[[(0, 1, 1), (1, 1, 1, 13), 'ct']] 65.198
> Model[[(0, 1, 1), (1, 0, 2, 13), 'ct']] 6088413.737
> Model[[(0, 1, 1), (2, 0, 0, 13), 'ct']] 50.445
> Model[[(0, 1, 1), (2, 0, 1, 13), 'ct']] 50.408
> Model[[(0, 1, 1), (2, 0, 2, 13), 'ct']] 78953292.879
> Model[[(0, 1, 1), (1, 1, 2, 13), 'ct']] 4851492.732
> Model[[(0, 1, 1), (2, 1, 0, 13), 'ct']] 90.213
> Model[[(0, 1, 1), (2, 1, 1, 13), 'ct']] 76.301
> Model[[(0, 1, 2), (0, 0, 0, 13), 'n']] 50.405
> Model[[(0, 1, 2), (0, 0, 0, 52), 'n']] 50.405
> Model[[(0, 1, 2), (0, 0, 1, 13), 'n']] 50.320
> Model[[(0, 1, 2), (0, 0, 2, 13), 'n']] 47602.011
> Model[[(0, 1, 2), (0, 1, 0, 13), 'n']] 103.881
> Model[[(0, 1, 1), (2, 1, 2, 13), 'ct']] 6009.878
> Model[[(0, 1, 2), (0, 1, 1, 13), 'n']] 52.806
> Model[[(0, 1, 2), (0, 1, 0, 52), 'n']] 27.873
> Model[[(0, 1, 2), (1, 0, 0, 13), 'n']] 50.272
> Model[[(0, 1, 2), (1, 0, 0, 52), 'n']] 56.170
> Model[[(0, 1, 2), (1, 0, 1, 13), 'n']] 50.099
> Model[[(0, 1, 2), (0, 1, 2, 13), 'n']] 67393869.148
> Model[[(0, 1, 2), (1, 1, 0, 13), 'n']] 64.029
> Model[[(0, 1, 2), (1, 1, 1, 13), 'n']] 51.110
> Model[[(0, 1, 2), (1, 0, 2, 13), 'n']] 10158041.120
> Model[[(0, 1, 2), (2, 0, 0, 13), 'n']] 48.723
> Model[[(0, 1, 2), (2, 0, 1, 13), 'n']] 48.469
> Model[[(0, 1, 2), (2, 0, 2, 13), 'n']] 8710075494224.834
> Model[[(0, 1, 2), (2, 1, 0, 13), 'n']] 64.997
> Model[[(0, 1, 2), (1, 1, 2, 13), 'n']] 7456759.909
> Model[[(0, 1, 2), (2, 1, 1, 13), 'n']] 47.938
> Model[[(0, 1, 2), (0, 0, 0, 13), 'c']] 47.397
> Model[[(0, 1, 2), (0, 0, 0, 52), 'c']] 47.397
> Model[[(0, 1, 2), (0, 0, 1, 13), 'c']] 45.448
> Model[[(0, 1, 2), (0, 0, 2, 13), 'c']] 7129.073
> Model[[(0, 1, 2), (0, 1, 0, 13), 'c']] 103.414
> Model[[(0, 1, 2), (2, 1, 2, 13), 'n']] 6064.905
> Model[[(0, 1, 2), (0, 1, 1, 13), 'c']] 53.167
> Model[[(0, 1, 2), (0, 1, 0, 52), 'c']] 31.563
> Model[[(0, 1, 2), (1, 0, 0, 13), 'c']] 45.797
> Model[[(0, 1, 2), (0, 1, 2, 13), 'c']] 51569334.335
> Model[[(0, 1, 2), (1, 0, 0, 52), 'c']] 60.046
> Model[[(0, 1, 2), (1, 0, 1, 13), 'c']] 45.299
> Model[[(0, 1, 2), (1, 1, 0, 13), 'c']] 63.796
> Model[[(0, 1, 2), (1, 1, 1, 13), 'c']] 51.466
> Model[[(0, 1, 2), (1, 0, 2, 13), 'c']] 1389330753.954
> Model[[(0, 1, 2), (2, 0, 0, 13), 'c']] 43.653
```

```
> Model[[(0, 1, 2), (2, 0, 1, 13), 'c']] 43.534
> Model[[(0, 1, 2), (2, 0, 2, 13), 'c']] 70989464.497
> Model[[(0, 1, 2), (1, 1, 2, 13), 'c']] 150806156.637
> Model[[(0, 1, 2), (2, 1, 0, 13), 'c']] 64.959
> Model[[(0, 1, 2), (2, 1, 1, 13), 'c']] 48.744
> Model[[(0, 1, 2), (0, 0, 0, 13), 't']] 44.832
> Model[[(0, 1, 2), (0, 0, 0, 52), 't']] 44.832
> Model[[(0, 1, 2), (0, 0, 1, 13), 't']] 44.340
> Model[[(0, 1, 2), (0, 0, 2, 13), 't']] 29262.846
> Model[[(0, 1, 2), (0, 1, 0, 13), 't']] 105.176
> Model[[(0, 1, 2), (0, 1, 0, 52), 't']] 32.818
> Model[[(0, 1, 2), (2, 1, 2, 13), 'c']] 7120.073
> Model[[(0, 1, 2), (0, 1, 1, 13), 't']] 55.671
> Model[[(0, 1, 2), (1, 0, 0, 13), 't']] 44.314
> Model[[(0, 1, 2), (0, 1, 2, 13), 't']] 129921302.908
> Model[[(0, 1, 2), (1, 0, 0, 52), 't']] 51.285
> Model[[(0, 1, 2), (1, 0, 1, 13), 't']] 44.023
> Model[[(0, 1, 2), (1, 1, 0, 13), 't']] 66.329
> Model[[(0, 1, 2), (1, 1, 1, 13), 't']] 54.127
> Model[[(0, 1, 2), (1, 0, 2, 13), 't']] 1297236695.154
> Model[[(0, 1, 2), (2, 0, 0, 13), 't']] 43.249
> Model[[(0, 1, 2), (2, 0, 1, 13), 't']] 42.993
> Model[[(0, 1, 2), (2, 0, 2, 13), 't']] 37295873.872
> Model[[(0, 1, 2), (2, 1, 0, 13), 't']] 69.151
> Model[[(0, 1, 2), (1, 1, 2, 13), 't']] 7211999.340
> Model[[(0, 1, 2), (2, 1, 1, 13), 't']] 52.174
> Model[[(0, 1, 2), (0, 0, 0, 13), 'ct']] 44.915
> Model[[(0, 1, 2), (0, 0, 0, 52), 'ct']] 44.915
> Model[[(0, 1, 2), (0, 0, 1, 13), 'ct']] 48.607
> Model[[(0, 1, 2), (0, 0, 2, 13), 'ct']] 2842.591
> Model[[(0, 1, 2), (0, 1, 0, 13), 'ct']] 110.163
> Model[[(0, 1, 2), (2, 1, 2, 13), 't']] 10710105041.452
> Model[[(0, 1, 2), (0, 1, 1, 13), 'ct']] 68.437
> Model[[(0, 1, 2), (0, 1, 0, 52), 'ct']] 31.532
> Model[[(0, 1, 2), (1, 0, 0, 13), 'ct']] 47.463
> Model[[(0, 1, 2), (0, 1, 2, 13), 'ct']] 29064401.499
> Model[[(0, 1, 2), (1, 0, 1, 13), 'ct']] 48.432
> Model[[(0, 1, 2), (1, 0, 2, 13), 'ct']] 9586554157096.320
> Model[[(0, 1, 2), (1, 0, 0, 52), 'ct']] 50.474
> Model[[(0, 1, 2), (1, 1, 0, 13), 'ct']] 75.342
> Model[[(0, 1, 2), (1, 1, 1, 13), 'ct']] 67.304
> Model[[(0, 1, 2), (2, 0, 0, 13), 'ct']] 50.639
> Model[[(0, 1, 2), (2, 0, 1, 13), 'ct']] 50.474
> Model[[(0, 1, 2), (2, 0, 2, 13), 'ct']] 81054666.739
> Model[[(0, 1, 2), (1, 1, 2, 13), 'ct']] 6331638.534
> Model[[(0, 1, 2), (2, 1, 0, 13), 'ct']] 91.651
> Model[[(0, 1, 2), (2, 1, 1, 13), 'ct']] 78.124
> Model[[(1, 0, 0), (0, 0, 0, 13), 'n']] 50.763
> Model[[(1, 0, 0), (0, 0, 0, 52), 'n']] 50.763
> Model[[(1, 0, 0), (0, 0, 1, 13), 'n']] 51.255
> Model[[(1, 0, 0), (0, 0, 2, 13), 'n']] 384.492
> Model[[(1, 0, 0), (0, 1, 0, 13), 'n']] 103.996
> Model[[(1, 0, 0), (0, 1, 0, 52), 'n']] 29.919
> Model[[(0, 1, 2), (2, 1, 2, 13), 'ct']] 1009.716
> Model[[(1, 0, 0), (0, 1, 1, 13), 'n']] 59.474
> Model[[(1, 0, 0), (1, 0, 0, 13), 'n']] 50.254
> Model[[(1, 0, 0), (1, 0, 0, 52), 'n']] 178.681
```

```
> Model[[(1, 0, 0), (1, 0, 1, 13), 'n']] 50.251
> Model[[(1, 0, 0), (0, 1, 2, 13), 'n']] 28758.774
> Model[[(1, 0, 0), (1, 1, 0, 13), 'n']] 65.453
> Model[[(1, 0, 0), (1, 0, 2, 13), 'n']] 546317182.809
> Model[[(1, 0, 0), (1, 1, 1, 13), 'n']] 58.273
> Model[[(1, 0, 0), (2, 0, 0, 13), 'n']] 49.739
> Model[[(1, 0, 0), (2, 0, 1, 13), 'n']] 49.659
> Model[[(1, 0, 0), (2, 0, 2, 13), 'n']] 115253835.566
> Model[[(1, 0, 0), (1, 1, 2, 13), 'n']] 52947.853
> Model[[(1, 0, 0), (2, 1, 0, 13), 'n']] 63.758
> Model[[(1, 0, 0), (2, 1, 1, 13), 'n']] 55.623
> Model[[(1, 0, 0), (0, 0, 0, 13), 'c']] 54.434
> Model[[(1, 0, 0), (0, 0, 0, 52), 'c']] 54.434
> Model[[(1, 0, 0), (0, 0, 1, 13), 'c']] 54.546
> Model[[(1, 0, 0), (0, 0, 2, 13), 'c']] 609.859
> Model[[(1, 0, 0), (0, 1, 0, 13), 'c']] 100.916
> Model[[(1, 0, 0), (2, 1, 2, 13), 'n']] 39743640.729
> Model[[(1, 0, 0), (0, 1, 0, 52), 'c']] 87.577
> Model[[(1, 0, 0), (0, 1, 1, 13), 'c']] 45.836
> Model[[(1, 0, 0), (1, 0, 0, 13), 'c']] 54.345
> Model[[(1, 0, 0), (0, 1, 2, 13), 'c']] 24882.488
> Model[[(1, 0, 0), (1, 0, 1, 13), 'c']] 53.998
> Model[[(1, 0, 0), (1, 0, 0, 52), 'c']] 133.870
> Model[[(1, 0, 0), (1, 1, 0, 13), 'c']] 59.313
> Model[[(1, 0, 0), (1, 0, 2, 13), 'c']] 1009329138.986
> Model[[(1, 0, 0), (1, 1, 1, 13), 'c']] 44.417
> Model[[(1, 0, 0), (2, 0, 0, 13), 'c']] 52.916
> Model[[(1, 0, 0), (2, 0, 1, 13), 'c']] 53.076
> Model[[(1, 0, 0), (1, 1, 2, 13), 'c']] 34953.284
> Model[[(1, 0, 0), (2, 0, 2, 13), 'c']] 87485065.656
> Model[[(1, 0, 0), (2, 1, 0, 13), 'c']] 53.029
> Model[[(1, 0, 0), (2, 1, 1, 13), 'c']] 39.975
> Model[[(1, 0, 0), (0, 0, 0, 13), 't']] 41.164
> Model[[(1, 0, 0), (0, 0, 0, 52), 't']] 41.164
> Model[[(1, 0, 0), (0, 0, 1, 13), 't']] 43.486
> Model[[(1, 0, 0), (2, 1, 2, 13), 'c']] 304347236.941
> Model[[(1, 0, 0), (0, 1, 0, 13), 't']] 101.194
> Model[[(1, 0, 0), (0, 0, 2, 13), 't']] 302.802
> Model[[(1, 0, 0), (0, 1, 1, 13), 't']] 44.506
> Model[[(1, 0, 0), (0, 1, 0, 52), 't']] 69.491
> Model[[(1, 0, 0), (1, 0, 0, 13), 't']] 41.730
> Model[[(1, 0, 0), (0, 1, 2, 13), 't']] 32523.212
> Model[[(1, 0, 0), (1, 0, 0, 52), 't']] 179.726
> Model[[(1, 0, 0), (1, 0, 1, 13), 't']] 41.713
> Model[[(1, 0, 0), (1, 1, 0, 13), 't']] 59.927
> Model[[(1, 0, 0), (1, 1, 1, 13), 't']] 43.237
> Model[[(1, 0, 0), (1, 0, 2, 13), 't']] 717377453.352
> Model[[(1, 0, 0), (2, 0, 0, 13), 't']] 41.074
> Model[[(1, 0, 0), (2, 0, 1, 13), 't']] 40.884
> Model[[(1, 0, 0), (1, 1, 2, 13), 't']] 185033.084
> Model[[(1, 0, 0), (2, 0, 2, 13), 't']] 99100583.562
> Model[[(1, 0, 0), (2, 1, 0, 13), 't']] 56.621
> Model[[(1, 0, 0), (2, 1, 1, 13), 't']] 42.233
> Model[[(1, 0, 0), (0, 0, 0, 13), 'ct']] 43.490
> Model[[(1, 0, 0), (0, 0, 0, 52), 'ct']] 43.490
> Model[[(1, 0, 0), (0, 0, 1, 13), 'ct']] 45.043
```

```
> Model[[(1, 0, 0), (0, 0, 2, 13), 'ct']] 527.804
> Model[[(1, 0, 0), (0, 1, 0, 13), 'ct']] 105.869
> Model[[(1, 0, 0), (2, 1, 2, 13), 't']] 4017193.505
> Model[[(1, 0, 0), (0, 1, 1, 13), 'ct']] 63.826
> Model[[(1, 0, 0), (0, 1, 0, 52), 'ct']] 49.768
> Model[[(1, 0, 0), (1, 0, 0, 13), 'ct']] 43.412
> Model[[(1, 0, 0), (0, 1, 2, 13), 'ct']] 23331.749
> Model[[(1, 0, 0), (1, 0, 1, 13), 'ct']] 43.375
> Model[[(1, 0, 0), (1, 0, 2, 13), 'ct']] 960345473.870
> Model[[(1, 0, 0), (1, 0, 0, 52), 'ct']] 67.852
> Model[[(1, 0, 0), (1, 1, 0, 13), 'ct']] 70.594
> Model[[(1, 0, 0), (1, 1, 1, 13), 'ct']] 61.879
> Model[[(1, 0, 0), (2, 0, 0, 13), 'ct']] 43.081
> Model[[(1, 0, 0), (2, 0, 1, 13), 'ct']] 43.243
> Model[[(1, 0, 0), (1, 1, 2, 13), 'ct']] 100145.616
> Model[[(1, 0, 0), (2, 0, 2, 13), 'ct']] 77249912.350
> Model[[(1, 0, 0), (2, 1, 0, 13), 'ct']] 79.740
> Model[[(1, 0, 0), (2, 1, 1, 13), 'ct']] 69.670
> Model[[(1, 0, 1), (0, 0, 0, 13), 'n']] 49.653
> Model[[(1, 0, 1), (0, 0, 0, 52), 'n']] 49.653
> Model[[(1, 0, 1), (0, 0, 1, 13), 'n']] 50.309
> Model[[(1, 0, 1), (0, 0, 2, 13), 'n']] 367.692
> Model[[(1, 0, 1), (0, 1, 0, 13), 'n']] 103.880
> Model[[(1, 0, 1), (0, 1, 0, 52), 'n']] 31.769
> Model[[(1, 0, 0), (2, 1, 2, 13), 'ct']] 25230685.470
> Model[[(1, 0, 1), (0, 1, 1, 13), 'n']] 59.563
> Model[[(1, 0, 1), (1, 0, 0, 13), 'n']] 49.583
> Model[[(1, 0, 1), (1, 0, 0, 52), 'n']] 152.748
> Model[[(1, 0, 1), (1, 0, 1, 13), 'n']] 49.300
> Model[[(1, 0, 1), (1, 0, 2, 13), 'n']] 618997752.385
```

```
# SARIMA forecast
# fit model
model = SARIMAX(df_energyWeek["Energy_05-06"], order=(0, 1, 0), seasonal_order=(0, 1, 0, 5
model_fit = model.fit(disp=False)
model_fit.aic
```

```
646.7513135401384
```

```
> Model[[(1, 0, 1), (2, 1, 1, 13), 'n']] 55.327
```

```
mf = model_fit.get_forecast(steps=52)
model_forecast = mf.predicted_mean
df_energyWeek["Energy_05-06"].plot()
model_forecast.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fefe8d6af50>
```


```
1 model_forecast
```

```
2007-01-07    2104.619289
2007-01-14    2137.176809
2007-01-21    2030.074824
2007-01-28    1963.327585
2007-02-04    1858.010737
2007-02-11    1862.040388
2007-02-18    1905.868554
2007-02-25    1898.763338
2007-03-04    1876.732437
2007-03-11    1785.250854
2007-03-18    1835.575928
2007-03-25    1716.913667
2007-04-01    1726.827026
2007-04-08    1756.786790
2007-04-15    1757.156198
2007-04-22    1684.739601
2007-04-29    1656.645021
2007-05-06    1701.374196
2007-05-13    1690.732066
2007-05-20    1639.102263
2007-05-27    1637.089553
2007-06-03    1643.556327
2007-06-10    1668.233854
2007-06-17    1742.780041
2007-06-24    1769.984096
2007-07-01    1820.432125
2007-07-08    1851.751259
2007-07-15    1693.089736
2007-07-22    1755.423379
2007-07-29    1795.967351
2007-08-05    1819.964328
2007-08-12    1891.786792
2007-08-19    1814.582797
2007-08-26    1867.287032
2007-09-02    1812.169791
2007-09-09    1733.542704
2007-09-16    1693.867007
2007-09-23    1621.839741
2007-09-30    1612.882244
2007-10-07    1597.700633
2007-10-14    1614.090983
2007-10-21    1609.534539
2007-10-28    1598.909953
2007-11-04    1639.509980
2007-11-11    1657.973459
2007-11-18    1692.334938
2007-11-25    1686.101927
```

```
2007-12-02    1735.942355
2007-12-09    1839.149726
2007-12-16    1875.289023
2007-12-23    1888.392004
2007-12-30    2007.016163
Freq: W-SUN, dtype: float64
```

✓ 0 s completado a las 0:04