Regression on Household Power Consumption Dataset

Life Cycle of machine learning Project

1 Underwent the problem statement 2 Data collection 3 Data cleaning 4 EDA 5 Preprocessing 6 K-Look Training 7 Cross Validation

Measures taken to improve model accuracy are included

1. Problem Statement

2. Data Collection

a. Importing required packages

```
# Import numpy for array computations, numerical and meaningful indexes
import numpy as np
import numpy as np, np
# Import pandas for data manipulation
import pandas as pd
import pandas as pd, pd
# Import matplotlib for plots
import matplotlib
import matplotlib, mpl
# Import seaborn for more plots
import seaborn as sns
import seaborn, sns
# Import matplotlib.pyplot for plots
import matplotlib.pyplot as plt
import matplotlib.pyplot, plt
# Import numpy.linalg for linear algebra
import numpy.linalg as LA
import numpy.linalg, LA
# Import Counter for counting
from collections import Counter
from collections import Counter, Counter
# Set the width and the height of the plotted
rcParams['figure.figure_size'] = 15, 6
rcParams['figure.figure_size'], 15, 6
# Ignore warnings
import warnings
import warnings, warnings
```

b. reading the dataset

```
household_power_consumption = pd.read_csv("household_power_consumption.txt", sep=";")
```

```
data.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
0	16/12/2006	17:24:00	4.216	0.418	234.84	18.400	0.000	1.000	17.000
1	16/12/2006	17:25:00	5.374	0.498	233.29	23.000	0.000	1.000	16.000
2	16/12/2006	17:26:00	5.268	0.492	233.74	23.000	0.000	2.000	17.000
3	16/12/2006	17:27:00	4.054	0.290	235.68	15.800	0.000	1.000	17.000
4	16/12/2006	17:28:00	3.384	0.290	236.39	15.000	0.000	2.000	17.000

3. Data Profiling and Data cleaning

```
data.shape
```

```
(2075259, 9)
```

```
data.isnull().sum(axis=0).sort_values(ascending=False)
```

```
Sub_metering_3         25979
Sub_metering_2         25979
Sub_metering_1         25979
Global_reactive_power  25979
Global_intensity       25979
Voltage                25979
Global_active_power    25979
Time                   25979
Date                   25979
dtype: int64
```

```
data.info()
```

```
<class pandas.core.frame.DataFrame>
RangeIndex: 2075259 entries, 0 to 2075258
Data columns (total 9 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   Date                   object 
 1   Time                   object 
 2   Global_active_power    float64
 3   Global_reactive_power  float64
 4   Voltage                float64
 5   Global_intensity       float64
 6   Sub_metering_1         float64
 7   Sub_metering_2         float64
 8   Sub_metering_3         float64
dtype: float64
```

```
data.info(),
```

```
<class pandas.core.frame.DataFrame>
RangeIndex: 2075259 entries, 0 to 2075258
Data columns (total 9 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   Date                   object 
 1   Time                   object 
 2   Global_active_power    float64
 3   Global_reactive_power  float64
 4   Voltage                float64
 5   Global_intensity       float64
 6   Sub_metering_1         float64
 7   Sub_metering_2         float64
 8   Sub_metering_3         float64
dtype: float64,
```

a. separating the date into date month year

```
data['Date'] = pd.to_datetime(data['Date'])
data['year'] = pd.DatetimeIndex(data['Date'])
data['month'] = pd.DatetimeIndex(data['Date'])
data['day'] = pd.DatetimeIndex(data['Date'])
```

```
data.head(1)
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3	year	month	day
0	2006-12-16	17:24:00	4.216	0.418	234.84	18.400	0.000	1.000	17.000	2006	12	16

```
data.dtypes,
```

```
Date                   object
Time                   object
Global_active_power    float64
Global_reactive_power  float64
Voltage                float64
Global_intensity       float64
Sub_metering_1         float64
Sub_metering_2         float64
Sub_metering_3         float64
year                   int64
month                  int64
day                    int64
dtype: object,
```

b. check null values

```
# dataset.isnull() dataset.notnull() dataset.shape dataset.to_numpy()
```

```
Date                   0
Time                   0
Global_active_power    0
Global_reactive_power  0
Voltage                0
Global_intensity       0
Sub_metering_1         0
Sub_metering_2         0
Sub_metering_3         0
year                   0
month                  0
day                    0
dtype: int64
```

c. check all

```
# dataset
```

```
Date                   object
Time                   object
Global_active_power    float64
Global_reactive_power  float64
Voltage                float64
Global_intensity       float64
Sub_metering_1         float64
Sub_metering_2         float64
Sub_metering_3         float64
year                   int64
month                  int64
day                    int64
dtype: object
```

d. check all zero column for uniqueness

```
# dataset.shape(axis=1).value_counts()
```

```
2    2075259
dtype: int64
```

```
# dataset.shape(axis=1).value_counts(ascending=True) dtypes nunique
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3	year	month	day
dtypes	object	object	float64	float64	float64	float64	float64	float64	float64	int64	int64	int64
nunique	1442	1440	4187	527	2838	222	89	82	32	4	12	31

NOW data = 75 columns

```
# dataset.shape(axis=1).value_counts(ascending=True) dtypes nunique
```

```
2    2075259
dtype: int64
```

```
# dataset.shape(axis=1).value_counts(ascending=False) dtypes nunique
```

```
2    2075259
dtype: int64
```

e. check

```
# dataset
```

```
Date                   object
Time                   object
Global_active_power    float64
Global_reactive_power  float64
Voltage                float64
Global_intensity       float64
Sub_metering_1         float64
Sub_metering_2         float64
Sub_metering_3         float64
year                   int64
month                  int64
day                    int64
dtype: object
```

handle special character in rows

```
# dataset[dataset['Global_active_power'].isin(['?'])].shape
dataset[~dataset.isin(['?'])].shape dropna(axis=0)
# dataset[dataset['Global_active_power'].isin(['?'])].shape
dataset[~dataset.isin(['?'])].shape
# dataset[~dataset.isin(['?'])].shape
dataset.shape
```

```
(2075259, 12)
```

```
# dataset[~dataset.isin(['?'])].shape
```

```
(2075259, 12)
```

```
# dataset[~dataset.isin(['?'])].shape
```

```
(2075259, 12)
```

f. convert datatype into float

```
# dataset.info()
```

```
<class pandas.core.frame.DataFrame>
RangeIndex: 2075259 entries, 0 to 2075258
Data columns (total 12 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   Date                   object 
 1   Time                   object 
 2   Global_active_power    float64
 3   Global_reactive_power  float64
 4   Voltage                float64
 5   Global_intensity       float64
 6   Sub_metering_1         float64
 7   Sub_metering_2         float64
 8   Sub_metering_3         float64
 9   year                   int64  
 10  month                  int64  
 11  day                    int64  
dtype: float64
```

```
# dataset[~dataset.isin(['?'])].shape dropna(axis=0)
```

```
(2075259, 12)
```

```
# dataset[~dataset.isin(['?'])].shape
```

```
(2075259, 12)
```

```
# dataset[~dataset.isin(['?'])].shape
```

```
(2075259, 12)
```

g. handling null value with mean

```
# dataset.isnull().sum(axis=0).sort_values(ascending=False)
```

```
Sub_metering_3         25979
Sub_metering_2         25979
Sub_metering_1         25979
Global_reactive_power  25979
Global_intensity       25979
Voltage                25979
Global_active_power    25979
Time                   25979
Date                   25979
year                   0
month                  0
day                    0
dtype: int64
```

```
# dataset[~dataset.isin(['?'])].shape dropna(axis=0).sum(axis=0)
```

```
Date                   object
Time                   object
Global_active_power    float64
Global_reactive_power  float64
Voltage                float64
Global_intensity       float64
Sub_metering_1         float64
Sub_metering_2         float64
Sub_metering_3         float64
year                   int64
month                  int64
day                    int64
dtype: object
```

```
# dataset[~dataset.isin(['?'])].shape dropna(axis=0).sum(axis=0).mean()
```

```
(2075259, 12)
```

```
# dataset
```

```
Date                   object
Time                   object
Global_active_power    float64
Global_reactive_power  float64
Voltage                float64
Global_intensity       float64
Sub_metering_1         float64
Sub_metering_2         float64
Sub_metering_3         float64
year                   int64
month                  int64
day                    int64
dtype: object
```

i. removing unnecessary rows and adding target features

```
# dataset
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3	year	month	day
0	2006-12-16	17:24:00	4.216	0.418	234.84	18.400	0.000	1.000	17.000	2006	12	16
1	2006-12-16	17:25:00	5.374	0.498	233.29	23.000	0.000	1.000	16.000	2006	12	16
2	2006-12-16	17:26:00	5.268	0.492	233.74	23.000	0.000	2.000	17.000	2006	12	16
3	2006-12-16	17:27:00	4.054	0.290	235.68	15.800	0.000	1.000	17.000	2006	12	16
4	2006-12-16	17:28:00	3.384	0.290	236.39	15.000	0.000	2.000	17.000	2006	12	16

```
# dataset[~dataset.isin(['?'])].shape dropna(axis=0).sum(axis=0).mean().mean()
```

```
(2075259, 12)
```

```
# dataset
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3	year	month	day
0	2006-12-16	17:24:00	4.216	0.418	234.84	18.400	0.000	1.000	17.000	2006	12	16
1	2006-12-16	17:25:00	5.374	0.498	233.29	23.000	0.000	1.000	16.000	2006	12	16
2	2006-12-16	17:26:00	5.268	0.492	233.74	23.000	0.000	2.000	17.000	2006	12	16
3	2006-12-16	17:27:00	4.054	0.290	235.68	15.800	0.000	1.000	17.000	2006	12	16
4	2006-12-16	17:28:00	3.384	0.290	236.39	15.000	0.000	2.000	17.000	2006	12	16

```
# dataset[~dataset.isin(['?'])].shape dropna(axis=0).sum(axis=0).mean().mean().mean()
```

```
(2075259, 12)
```

```
# dataset
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3	year	month	day
0	2006-12-16	17:24:00	4.216	0.418	234.84	18.400	0.000	1.000	17.000	2006	12	16
1	2006-12-16	17:25:00	5.374	0.498	233.29	23.000	0.000	1.000	16.000	2006	12	16
2	2006-12-16	17:26:00	5.268	0.492	233.74	23.000	0.000	2.000	17.000	2006	12	16
3	2006-12-16	17:27:00	4.054	0.290	235.68	15.800	0.000	1.000	17.000	2006	12	16
4	2006-12-16	17:28:00	3.384	0.290	236.39	15.000	0.000	2.000	17.000	2006	12	16

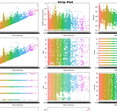
4. Statistical Analysis

4.1 Numerical and categorical features:

```
# dataset.describe(include=['O']).T
# dataset.describe(include=['O']).T dtypes nunique
```

```
Date                   object
Time                   object
dtype: object
```

```
```



4.4 Multivariant Analysis

```
In [108]: df.corr()
```

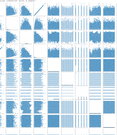
	Hotel_price_per	Hotel_stars_per	Voltage	Hotel_internet_	city	month	year	hour	minute	Total_ticketing_
Hotel_price_per	1.000000	0.249011	-0.007419	-0.008266	-0.010174	-0.000473	-0.010000	-0.000000	-0.000000	-0.000000
Hotel_stars_per	0.249011	1.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000
Voltage	-0.007419	-0.000000	1.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000
Hotel_internet_	-0.008266	-0.000000	-0.000000	1.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000
city	-0.010174	-0.000000	-0.000000	-0.000000	1.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000
month	-0.000473	-0.000000	-0.000000	-0.000000	-0.000000	1.000000	-0.000000	-0.000000	-0.000000	-0.000000
year	-0.010000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	1.000000	-0.000000	-0.000000	-0.000000
hour	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	1.000000	-0.000000	-0.000000
minute	0.007174	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	1.000000	-0.000000
Total_ticketing_	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	-0.000000	1.000000

Pair Plot

```
In [109]: sns.pairplot(df)
Out[109]: <seaborn.axisgrid.PairGrid at 0x1a1a1a1a>
```

Heat Map

```
In [110]: plt.figure(figsize=(20,20))
sns.heatmap(df.corr(),annot=True)
Out[110]: <matplotlib.axes._subplots.AxesSubplot>, <matrix>
```



5. Graph Analysis [Visualization]

Q. HOur for peak power consumption?

```
In [111]: df.groupby('hour').sum()
Out[111]: <DataFrame_hour_>
```

Q month having least power consumption

```
In [112]: df.groupby('month').sum()
Out[112]: <DataFrame_month_>
```

Q year vs consumption trend.

```
In [113]: df.groupby('year').sum()
Out[113]: <DataFrame_year_>
```

Variance Inflation Factor (VIF)

```
In [114]: df.head()
```

	Hotel_price_per	Hotel_stars_per	Voltage	Hotel_internet_	city	month	year	hour	minute	Total_ticketing_
1347718	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1348968	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349148	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349151	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349179	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0

Preprocessing [handling Outliers]

```
In [115]: df.describe()
Out[115]: <DataFrame_Hotel_price_per_Hotel_stars_per_Voltage_Hotel_internet__city_month_year_hour_minute_Total_ticketing_>
```

	Hotel_price_per	Hotel_stars_per	Voltage	Hotel_internet_	city	month	year	hour	minute	Total_ticketing_
count	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
mean	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
std	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
min	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
25%	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
50%	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
75%	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
max	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0

```
In [116]: df.shape
Out[116]: <tuple>
```

```
In [117]: df.dtypes
Out[117]: <series_object_>
```

	Hotel_price_per	Hotel_stars_per	Voltage	Hotel_internet_	city	month	year	hour	minute	Total_ticketing_
1347718	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1348968	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349148	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349151	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349179	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0

```
In [118]: df.dtypes
Out[118]: <series_object_>
```

```
In [119]: df.head()
```

	Hotel_price_per	Hotel_stars_per	Voltage	Hotel_internet_	city	month	year	hour	minute	Total_ticketing_
1347718	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1348968	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349148	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349151	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0
1349179	0.00	0.00	240.47	1.0	0	2020	12	0	0	1.0

```
In [120]: pip install Yellowbrick
Collecting Yellowbrick...
  Downloading Yellowbrick-1.3.post1-py2.7-any.whl (271 KB)
     |********************************| 271 KB 1.1MB/s
Requirement already satisfied: matplotlib!=3.0.3,!=3.1.0,!=3.1.1,!=3.1.2,!=3.1.3,!=3.2.0,!=3.2.1,!=3.2.2,!=3.2.3,!=3.2.4,!=3.3.0,!=3.3.1,!=3.3.2,!=3.3.3,!=3.3.4,!=3.4.0,!=3.4.1,!=3.4.2,!=3.4.3,!=3.5.0,!=3.5.1,!=3.5.2,!=3.5.3,!=3.6.0,!=3.6.1,!=3.6.2,!=3.6.3,!=3.7.0,!=3.7.1,!=3.7.2,!=3.7.3,!=3.7.4,!=3.8.0,!=3.8.1,!=3.8.2,!=3.8.3,!=3.8.4,!=3.9.0,!=3.9.1,!=3.9.2,!=3.9.3,!=3.9.4,!=3.10.0,!=3.10.1,!=3.10.2,!=3.10.3,!=3.10.4,!=3.11.0,!=3.11.1,!=3.11.2,!=3.11.3,!=3.11.4,!=3.12.0,!=3.12.1,!=3.12.2,!=3.12.3,!=3.12.4,!=3.13.0,!=3.13.1,!=3.13.2,!=3.13.3,!=3.13.4,!=3.14.0,!=3.14.1,!=3.14.2,!=3.14.3,!=3.14.4,!=3.15.0,!=3.15.1,!=3.15.2,!=3.15.3,!=3.15.4,!=3.16.0,!=3.16.1,!=3.16.2,!=3.16.3,!=3.16.4,!=3.17.0,!=3.17.1,!=3.17.2,!=3.17.3,!=3.17.4,!=3.18.0,!=3.18.1,!=3.18.2,!=3.18.3,!=3.18.4,!=3.19.0,!=3.19.1,!=3.19.2,!=3.19.3,!=3.19.4,!=3.20.0,!=3.20.1,!=3.20.2,!=3.20.3,!=3.20.4,!=3.21.0,!=3.21.1,!=3.21.2,!=3.21.3,!=3.21.4,!=3.22.0,!=3.22.1,!=3.22.2,!=3.22.3,!=3.22.4,!=3.23.0,!=3.23.1,!=3.23.2,!=3.23.3,!=3.23.4,!=3.24.0,!=3.24.1,!=3.24.2,!=3.24.3,!=3.24.4,!=3.25.0,!=3.25.1,!=3.25.2,!=3.25.3,!=3.25.4,!=3.26.0,!=3.26.1,!=3.26.2,!=3.26.3,!=3.26.4,!=3.27.0,!=3.27.1,!=3.27.2,!=3.27.3,!=3.27.4,!=3.28.0,!=3.28.1,!=3.28.2,!=3.28.3,!=3.28.4,!=3.29.0,!=3.29.1,!=3.29.2,!=3.29.3,!=3.29.4,!=3.30.0,!=3.30.1,!=3.30.2,!=3.30.3,!=3.30.4,!=3.31.0,!=3.31.1,!=3.31.2,!=3.31.3,!=3.31.4,!=3.32.0,!=3.32.1,!=3.32.2,!=3.32.3,!=3.32.4,!=3.33.0,!=3.33.1,!=3.33.2,!=3.33.3,!=3.33.4,!=3.34.0,!=3.34.1,!=3.34.2,!=3.34.3,!=3.34.4,!=3.35.0,!=3.35.1,!=3.35.2,!=3.35.3,!=3.35.4,!=3.36.0,!=3.36.1,!=3.36.2,!=3.36.3,!=3.36.4,!=3.37.0,!=3.37.1,!=3.37.2,!=3.37.3,!=3.37.4,!=3.38.0,!=3.38.1,!=3.38.2,!=3.38.3,!=3.38.4,!=3.39.0,!=3.39.1,!=3.39.2,!=3.39.3,!=3.39.4,!=3.40.0,!=3.40.1,!=3.40.2,!=3.40.3,!=3.40.4,!=3.41.0,!=3.41.1,!=3.41.2,!=3.41.3,!=3.41.4,!=3.42.0,!=3.42.1,!=3.42.2,!=3.42.3,!=3.42.4,!=3.43.0,!=3.43.1,!=3.43.2,!=3.43.3,!=3.43.4,!=3.44.0,!=3.44.1,!=3.44.2,!=3.44.3,!=3.44.4,!=3.45.0,!=3.45.1,!=3.45.2,!=3.45.3,!=3.45.4,!=3.46.0,!=3.46.1,!=3.46.2,!=3.46.3,!=3.46.4,!=3.47.0,!=3.47.1,!=3.47.2,!=3.47.3,!=3.47.4,!=3.48.0,!=3.48.1,!=3.48.2,!=3.48.3,!=3.48.4,!=3.49.0,!=3.49.1,!=3.49.2,!=3.49.3,!=3.49.4,!=3.50.0,!=3.50.1,!=3.50.2,!=3.50.3,!=3.50.4,!=3.51.0,!=3.51.1,!=3.51.2,!=3.51.3,!=3.51.4,!=3.52.0,!=3.52.1,!=3.52.2,!=3.52.3,!=3.52.4,!=3.53.0,!=3.53.1,!=3.53.2,!=3.53.3,!=3.53.4,!=3.54.0,!=3.54.1,!=3.54.2,!=3.54.3,!=3.54.4,!=3.55.0,!=3.55.1,!=3.55.2,!=3.55.3,!=3.55.4,!=3.56.0,!=3.56.1,!=3.56.2,!=3.56.3,!=3.56.4,!=3.57.0,!=3.57.1,!=3.57.2,!=3.57.3,!=3.57.4,!=3.58.0,!=3.58.1,!=3.58.2,!=3.58.3,!=3.58.4,!=3.59.0,!=3.59.1,!=3.59.2,!=3.59.3,!=3.59.4,!=3.60.0,!=3.60.1,!=3.60.2,!=3.60.3,!=3.60.4,!=3.61.0,!=3.61.1,!=3.61.2,!=3.61.3,!=3.61.4,!=3.62.0,!=3.62.1,!=3.62.2,!=3.62.3,!=3.62.4,!=3.63.0,!=3.63.1,!=3.63.2,!=3.63.3,!=3.63.4,!=3.64.0,!=3.64.1,!=3.64.2,!=3.64.3,!=3.64.4,!=3.65.0,!=3.65.1,!=3.65.2,!=3.65.3,!=3.65.4,!=3.66.0,!=3.66.1,!=3.66.2,!=3.66.3,!=3.66.4,!=3.67.0,!=3.67.1,!=3.67.2,!=3.67.3,!=3.67.4,!=3.68.0,!=3.68.1,!=3.68.2,!=3.68.3,!=3.68.4,!=3.69.0,!=3.69.1,!=3.69.2,!=3.69.3,!=3.69.4,!=3.70.0,!=3.70.1,!=3.70.2,!=3.70.3,!=3.70.4,!=3.71.0,!=3.71.1,!=3.71.2,!=3.71.3,!=3.71.4,!=3.72.0,!=3.72.1,!=3.72.2,!=3.72.3,!=3.72.4,!=3.73.0,!=3.73.1,!=3.73.2,!=3.73.3,!=3.73.4,!=3.74.0,!=3.74.1,!=3.74.2,!=3.74.3,!=3.74.4,!=3.75.0,!=3.75.1,!=3.75.2,!=3.75.3,!=3.75.4,!=3.76.0,!=3.76.1,!=3.76.2,!=3.76.3,!=3.76.4,!=3.77.0,!=3.77.1,!=3.77.2,!=3.77.3,!=3.77.4,!=3.78.0,!=3.78.1,!=3.78.2,!=3.78.3,!=3.78.4,!=3.79.0,!=3.79.1,!=3.79.2,!=3.79.3,!=3.79.4,!=3.80.0,!=3.80.1,!=3.80.2,!=3.80.3,!=3.80.4,!=3.81.0,!=3.81.1,!=3.81.2,!=3.81.3,!=3.81.4,!=3.82.0,!=3.82.1,!=3.82.2,!=3.82.3,!=3.82.4,!=3.83.0,!=3.83.1,!=3.83.2,!=3.83.3,!=3.83.4,!=3.84.0,!=3.84.1,!=3.84.2,!=3.84.3,!=3.84.4,!=3.85.0,!=3.85.1,!=3.85.2,!=3.85.3,!=3.85.4,!=3.86.0,!=3.86.1,!=3.86.2,!=3.86.3,!=3.86.4,!=3.87.0,!=3.87.1,!=3.87.2,!=3.87.3,!=3.87.4,!=3.88.0,!=3.88.1,!=3.88.2,!=3.88.3,!=3.88.4,!=3.89.0,!=3.89.1,!=3.89.2,!=3.89.3,!=3.89.4,!=3.90.0,!=3.90.1,!=3.90.2,!=3.90.3,!=3.90.4,!=3.91.0,!=3.91.1,!=3.91.2,!=3.91.3,!=3.91.4,!=3.92.0,!=3.92.1,!=3.92.2,!=3.92.3,!=3.92.4,!=3.93.0,!=3.93.1,!=3.93.2,!=3.93.3,!=3.93.4,!=3.94.0,!=3.94.1,!=3.94.2,!=3.94.3,!=3.94.4,!=3.95.0,!=3.95.1,!=3.95.2,!=3.95.3,!=3.95.4,!=3.96.0,!=3.96.1,!=3.96.2,!=3.96.3,!=3.96.4,!=3.97.0,!=3.97.1,!=3.97.2,!=3.97.3,!=3.97.4,!=3.98.0,!=3.98.1,!=3.98.2,!=3.98.3,!=3.98.4,!=3.99.0,!=3.99.1,!=3.99.2,!=3.99.3,!=3.99.4,!=3.100.0,!=3.100.1,!=3.100.2,!=3.100.3,!=3.100.4,!=3.101.0,!=3.101.1,!=3.101.2,!=3.101.3,!=3.101.4,!=3.102.0,!=3.102.1,!=3.102.2,!=3.102.3,!=3.102.4,!=3.103.0,!=3.103.1,!=3.103.2,!=3.103.3,!=3.103.4,!=3.104.0,!=3.104.1,!=3.104.2,!=3.104.3,!=3.104.4,!=3.105.0,!=3.105.1,!=3.105.2,!=3.105.3,!=3.105.4,!=3.106.0,!=3.106.1,!=3.106.2,!=3.106.3,!=3.106.4,!=3.107.0,!=3.107.1,!=3.107.2,!=3.107.3,!=3.107.4,!=3.108.0,!=3.108.1,!=3.108.2,!=3.108.3,!=3.108.4,!=3.109.0,!=3.109.1,!=3.109.2,!=3.109.3,!=3.109.4,!=3.110.0,!=3.110.1,!=3.110.2,!=3.110.3,!=3.110.4,!=3.111.0,!=3.111.1,!=3.111.2,!=3.111.3,!=3.111.4,!=3.112.0,!=3.112.1,!=3.112.2,!=3.112.3,!=3.112.4,!=3.113.0,!=3.113.1,!=3.113.2,!=3.113.3,!=3.113.4,!=3.114.0,!=3.114.1,!=3.114.2,!=3.114.3,!=3.114.4,!=3.115.0,!=3.115.1,!=3.115.2,!=3.115.3,!=3.115.4,!=3.116.0,!=3.116.1,!=3.116.2,!=3.116.3,!=3.116.4,!=3.117.0,!=3.117.1,!=3.117.2,!=3.117.3,!=3.117.4,!=3.118.0,!=3.118.1,!=3.118.2,!=3.118.3,!=3.118.4,!=3.119.0,!=3.119.1,!=3.119.2,!=3.119.3,!=3.119.4,!=3.120.0,!=3.120.1,!=3.120.2,!=3.120.3,!=3.120.4,!=3.121.0,!=3.121.1,!=3.121.2,!=3.121.3,!=3.121.4,!=3.122.0,!=3.122.1,!=3.122.2,!=3.122.3,!=3.122.4,!=3.123.0,!=3.123.1,!=3.123.2,!=3.123.3,!=3.123.4,!=3.124.0,!=3.124.1,!=3.124.2,!=3.124.3,!=3.124.4,!=3.125.0,!=3.125.1,!=3.125.2,!=3.125.3,!=3.125.4,!=3.126.0,!=3.126.1,!=3.126.2,!=3.126.3,!=3.126.4,!=3.127.0,!=3.127.1,!=3.127.2,!=3.127.3,!=3.127.4,!=3.128.0,!=3.128.1,!=3.128.2,!=3.128.3,!=3.128.4,!=3.129.0,!=3.129.1,!=3.129.2,!=3.129.3,!=3.129.4,!=3.130.0,!=3.130.1,!=3.130.2,!=3.130.3,!=3.130.4,!=3.131.0,!=3.131.1,!=3.131.2,!=3.131.3,!=3.131.4,!=3.132.0,!=3.132.1,!=3.132.2,!=3.132.3,!=3.132.4,!=3.133.0,!=3.133.1,!=3.133.2,!=3.133.3,!=3.133.4,!=3.134.0,!=3.134.1,!=3.134.2,!=3.134.3,!=3.134.4,!=3.135.0,!=3.135.1,!=3.135.2,!=3.135.3,!=3.135.4,!=3.136.0,!=3.136.1,!=3.136.2,!=3.136.3,!=3.136.4,!=3.137.0,!=3.137.1,!=3.137.2,!=3.137.3,!=3.137.4,!=3.138.0,!=3.138.1,!=3.138.2,!=3.138.3,!=3.138.4,!=3.139.0,!=3.139.1,!=3.139.2,!=3.139.3,!=3.139.4,!=3.140.0,!=3.140.1,!=3.140.2,!=3.140.3,!=3.140.4,!=3.141.0,!=3.141.1,!=3.141.2,!=3.141.3,!=3.141.4,!=3.142.0,!=3.142.1,!=3.142.2,!=3.142.3,!=3.142.4,!=3.143.0,!=3.143.1,!=3.143.2,!=3.143.3,!=3.143.4,!=3.144.0,!=3.144.1,!=3.144.2,!=3.144.3,!=3.144.4,!=3.145.0,!=3.145.1,!=3.145.2,!=3.145.3,!=3.145.4,!=3.146.0,!=3.146.1,!=3.146.2,!=3.146.3,!=3.146.4,!=3.147.0,!=3.147.1,!=3.147.2,!=3.147.3,!=3.147.4,!=3.148.0,!=3.148.1,!=3.148.2,!=3.148.3,!=3.148.4,!=3.149.0,!=3.149.1,!=3.149.2,!=3.149.3,!=3.149.4,!=3.150.0,!=3.150.1,!=3.150.2,!=3.150.3,!=3.150.4,!=3.151.0,!=3.151.1,!=3.151.2,!=3.151.3,!=3.151.4,!=3.152.0,!=3.152.1,!=3.152.2,!=3.152.3,!=3.152.4,!=3.153.0,!=3.153.1,!=3.153.2,!=3.153.3,!=3.153.4,!=3.154.0,!=3.154.1,!=3.154.2,!=3.154.3,!=3.154.4,!=3.155.0,!=3.155.1,!=3.155.2,!=3.155.3,!=3.155.4,!=3.156.0,!=3.156.1,!=3.156.2,!=3.156.3,!=3.156.4,!=3.157.0,!=3.157.1,!=3.157.2,!=3.157.3,!=3.157.4,!=3.158.0,!=3.158.1,!=3.158.2,!=3.158.3,!=3.158.4,!=3.159.0,!=3.159.1,!=3.159.2,!=3.159.3,!=3.159.4,!=3.160.0,!=3.160.1,!=3.160.2,!=3.160.3,!=3.160.4,!=3.161.0,!=3.161.1,!=3.161.2,!=3.161.3,!=3.161.4,!=3.162.0,!=3.162.1,!=3.162.2,!=3.162.3,!=3.162.4,!=3.163.0,!=3.163.1,!=3.163.2,!=3.163.3,!=3.163.4,!=3.164.0,!=3.164.1,!=3.164.2,!=3.164.3,!=3.164.4,!=3.165.0,!=3.165.1,!=3.165.2,!=3.165.3,!=3.165.4,!=3.166.0,!=3.166.1,!=3.166.2,!=3.166.3,!=3.166.4,!=3.167.0,!=3.167.1,!=3.167.2,!=3.167.3,!=3.167.4,!=3.168.0,!=3.168.1,!=3.168.2,!=3.168.3,!=3.168.4,!=3.169.0,!=3.169.1,!=3.169.2,!=3.169.3,!=3.169.4,!=3.170.0,!=3.170.1,!=3.170.2,!=3.170.3,!=3.170.4,!=3.171.0,!=3.171.1,!=3.171.2,!=3.171.3,!=3.171.4,!=3.172.0,!=3.172.1,!=3.172.2,!=3.172.3,!=3.172.4,!=3.173.0,!=3.173.1,!=3.173.2,!=3.173.3,!=3.173.4,!=3.174.0,!=3.174.1,!=3.174.2,!=3.174.3,!=3.174.4,!=3.175.0,!=3.175.1,!=3.175.2,!=3.175.3,!=3.175.4,!=3.176.0,!=3.176.1,!=3.176.2,!=3.176.3,!=3.176.4,!=3.177.0,!=3.177.1,!=3.177.2,!=3.177.3,!=3.177.4,!=3.178.0,!=3.178.1,!=3.178.2,!=3.178.3,!=3.178.4,!=3.179.0,!=3.179.1,!=3.179.2,!=3.179.3,!=3.179.4,!=3.180.0,!=3.180.1,!=3.180.2,!=3.180.3,!=3.180.4,!=3.181.0,!=3.181.1,!=3.181.2,!=3.181.3,!=3.181.4,!=3.182.0,!=3.182.1,!=3.182.2,!=3.182.3,!=3.182.4,!=3.183.0,!=3.183.1,!=3.183.2,!=3.183.3,!=3.183.4,!=3.184.0,!=3.184.1,!=3.184.2,!=3.184.3,!=3.184.4,!=3.185.0,!=3.185.1,!=3.185.2,!=3.185.3,!=3.185.4,!=3.186.0,!=3.186.1,!=3.186.2,!=3.186.3,!=3.186.4,!=3.187.0,!=3.187.1,!=3.187.2,!=3.187.3,!=3.187.4,!=3.188.0,!=3.188.1,!=3.188.2,!=3.188.3,!=3.188.4,!=3.189.0,!=3.189.1,!=3.189.2,!=3.189.3,!=3.189.4,!=3.190.0,!=3.190.1,!=3.190.2,!=3.190.3,!=3.190.4,!=3.191.0,!=3.191.1,!=3.191.2,!=3.191.3,!=3.191.4,!=3.192.0,!=3.192.1,!=3.192.2,!=3.192.3,!=3.192.4,!=3.193.0,!=3.193.1,!=3.193.2,!=3.193.3,!=3.193.4,!=3.194.0,!=3.194.1,!=3.194.2,!=3.194.3,!=3.194.4,!=3.195.0,!=3.195.1,!=3.195.2,!=3.195.3,!=3.195.4,!=3.196.0,!=3.196.1,!=3.196.2,!=3.196.3,!=3.196.4,!=3.197.0,!=3.197.1,!=3.197.2,!=3.197.3,!=3.197.4,!=3.198.0,!=3.198.1,!=3.198.2,!=3.198.3,!=3.198.4,!=3.199.0,!=3.199.1,!=3.199.2,!=3.199.3,!=3.199.4,!=3.200.0,!=3.200.1,!=3.200.2,!=3.200.3,!=3.200.4,!=3.201.0,!=3.201.1,!=3.201.2,!=3.201.3,!=3.201.4,!=3.202.0,!=3.202.1,!=3.202.2,!=3.202.3,!=3.202.4,!=3.203.0,!=3.203.1,!=3.203.2,!=3.203.3,!=3.203.4,!=3.204.0,!=3.204.1,!=3.204.2,!=3.204.3,!=3.204.4,!=3.205.0,!=3.205.1,!=3.205.2,!=3.205.3,!=3.205.4,!=3.206.0,!=3.206.1,!=3.206.2,!=3.206.3,!=3.206.4,!=3.207.0,!=3.207.1,!=3.207.2,!=3.207.3,!=3.207.4,!=3.208.0,!=3.208.1,!=3.208.2,!=3.208.3,!=3.208.4,!=3.209.0,!=3.209.1,!=3.209.2,!=3.209.3,!=3.209.4,!=3.210.0,!=3.210.1,!=3.210.2,!=3.210.3,!=3.210.4,!=3.211.0,!=3.211.1,!=3.211.2,!=3.211.3,!=3.211.4,!=3.212.0,!=3.212.1,!=3.212.2,!=3.212.3,!=3.212.4,!=3.213.0,!=3.213.1,!=3.213.2,!=3.213.3,!=3.213.4,!=3.214.0,!=3.214.1,!=3.214.2,!=3.214.3,!=3.214.4,!=3.215.0,!=3.215.1,!=3.215.2,!=3.215.3,!=3.215.4,!=3.216.0,!=3.216.1,!=3.216.2,!=3.216.3,!=3.216.4,!=3.217.0,!=3.217.1,!=3.217.2,!=3.217.3,!=3.217.4,!=3.218.0,!=3.218.1,!=3.218.2,!=3.218.3,!=3.218.4,!=3.219.0,!=3.219.1,!=3.219.2,!=3.219.3,!=3.219.4,!=3.220.0,!=3.220.1,!=3.220.2,!=3.220.3,!=3.220.4,!=3.221.0,!=3.221.1,!=3.221.2,!=3.221.3,!=3.221.4,!=3.222.0,!=3.222.1,!=3.222.2,!=3.222.3,!=3.222.4,!=3.223.0,!=3.223.1,!=3.223.2,!=3.223.3,!=3.223.4,!=3.224.0,!=3.224.1,!=3.224.2,!=3.224.3,!=3.224.4,!=3.225.0,!=3.225.1,!=3.225.2,!=3.225.3,!=3.225.4,!=3.226.0,!=3.226.1,!=3.226.2,!=3.226.3,!=3.226.4,!=3.227.0,!=3.227.1,!=3.227.2,!=3.227.3,!=3.227.4,!=3.228.0,!=3.228.1,!=3.228.2,!=3.228.3,!=3.228.4,!=3.229.0,!=3.229.1,!=3.229.2,!=3.229.3,!=3.229.4,!=3.230.0,!=3.230.1,!=3.230.2,!=3.230.3,!=3.230.4,!=3.231.0,!=3.231.1,!=3.231.2,!=3.231.3,!=3.231.4,!=3.232.0,!=3.232.1,!=3.232.2,!=3.232.3,!=3.232.4,!=3.233.0,!=3.233.1,!=3.233.2,!=3.233.3,!=3.233.4,!=3.234.0,!=3.234.1,!=3.234.2,!=3.234.3,!=3.234.4,!=3.235.0,!=3.235.1,!=3.235.2,!=3.235.3,!=3.235.4,!=3.236.0,!=3.236.1,!=3.236.2,!=3.236.3,!=3.236.4,!=3.237.0,!=3.237.1,!=3.237.2,!=3.237.3,!=3.237.4,!=3.238.0,!=3.238.1,!=3.238.2,!=3.238.3,!=3.238.4,!=3.239.0,!=3.239.1,!=3.239.2,!=3.239.3,!=3.239.4,!=3.240.0,!=3.240.1,!=3.240.2,!=3.240.3,!=3.240.4,!=3.241.0,!=3.241.1,!=3.241.2,!=3.241.3,!=3.241.4,!=3.242.0,!=3.242.1,!=3.242.2,!=3.242.3,!=3.242.4,!=3.243.0,!=3.243.1,!=3.243.2,!=3.243.3,!=3.243.4,!=3.244.0,!=3.244.1,!=3.244.2,!=3.244.3,!=3.244.4,!=3.245.0,!=3.245.1,!=3.245.2,!=3.245.3,!=3.245.4,!=3.246.0,!=3.246.1,!=3.246.2,!=3.246.3,!=3.246.4,!=3.247.0,!=3.247.1,!=3.247.2,!=3.247.3,!=3.247.4,!=3.248.0,!=3.248.1,!=3.248.2,!=3.248.3,!=3.248.4,!=3.249.0,!=3.249.1,!=3.249.2,!=3.249.3,!=3.249.4,!=3.250.0,!=3.250.1,!=3.250.2,!=3.250.3,!=3.250.4,!=3.251.0,!=3.251.1,!=3.251.2,!=3.251.3,!=3.251.4,!=3.252.0,!=3.252.1,!=3.252.2,!=3.252.3,!=3.252.4,!=3.253.0,!=3.253.1,!=3.253.2,!=3.253.3,!=3.253.4,!=3.254.0,!=3.254.1,!=3.254.2,!=3.254.3,!=3.254.4,!=3.255.0,!=3.255.1,!=3.255.2,!=3.255.3,!=3.255.4,!=3.256.0,!=3.256.1,!=3.256.2,!=3.256.3,!=3.256.4,!=3.257.0,!=3.257.1,!=3.257.2,!=3.257.3,!=3.257.4,!=3.258.0,!=3.258.1,!=3.258.2,!=3.258.3,!=3.258.4,!=3.259.0,!=3.259.1,!=3.259.2,!=3.259.3,!=3.259.4,!=3.260.0,!=3.260.1,!=3.260.2,!=3.260.3,!=3.260.4,!=3.261.0,!=3.261.1,!=3.261.2,!=3.261.3,!=3.261.4,!=3.262.0,!=3.262.1,!=3.262.2,!=3.262.3,!=3.262.4,!=3.263.0,!=3.263.1,!=3.263.2,!=3.263.3,!=3.263.4,!=3.264.0,!=3.264.1,!=3.264.2,!=3.264.3,!=3.264.4,!=3.265.0,!=3.265.1,!=3.265.2,!=3.265.3,!=3.265.4,!=3.266.0,!=3.266.1,!=3.266.2,!=3.266.3,!=3.266.4,!=3.267.0,!=3.267.1,!=3.267.2,!=3.267.3,!=3.267.4,!=3.268.0,!=3.268.1,!=3.268.2,!=3.268.3,!=3.268.4,!=3.269.0,!=3.269.1,!=3.269.2,!=3.269.3,!=3.269.4,!=3.270.0,!=3.270.1,!=3.270.2,!=3.270.3,!=3.270.4,!=3.271.0,!=3.271.1,!=3.271.2,!=3.271.3,!=3.271.4,!=3.272.0,!=3.272.1,!=3.272.2,!=3.272.3,!=3.272.4,!=3.273.0,!=3.273.1,!=3.273.2,!=3.273.3,!=3.273.4,!=3.274.0,!=3.274.1,!=3.274.2,!=3.274.3,!=3.274.4,!=3.275.0,!=3.275.1,!=3.275.2,!=3.275.3,!=3.275.4,!=3.276.0,!=3.276.1,!=3.276.2,!=3.276.3,!=3.276.4,!=3.277.0,!=3.277.1,!=3.277.2,!=3.277.3,!=3.277.4,!=3.278.0,!=3.278.1,!=3.278.2,!=3.278.3,!=3.278.4,!=3.279.0,!=3.279.1,!=3.279.2,!=3.279.3,!=3.279.4,!=3.280.0,!=3.280.1,!=3.280.2,!=3.280.3,!=3.280.4,!=3.281.0,!=3.281.1,!=3.281.2,!=3.281.3,!=3.281.4,!=3.282.0,!=3.282.1,!=3.282.2,!=3.282.3,!=3.282.4,!=3.283.0,!=3.283.1,!=3.283.2,!=3.283.3,!=3.283.4,!=3.284.0,!=3.284.1,!=3.284.2,!=3.284.3,!=3.284.4,!=3.285.0,!=3.285.1,!=3.285.2,!=3.285.3,!=3.285.4,!=3.286.0,!=3.286.1,!=3.286.2,!=3.286.3,!=3.286.4,!=3.287.0,!=3.287.1,!=3.287.2,!=3.287.3,!=3.287.4,!=3.288.0,!=3.288.1,!=3.288.2,!=3.288.3,!=3.288.4,!=3.289.0,!=3.289.1,!=3.289.2,!=3.289.3,!=3.289.4,!=3.290.0,!=3.290.1,!=3.290.2,!=3.290.3,!=3.290.4,!=3.291.0,!=3.291.1,!=3.291.2,!=3.291.3,!=3.291.4,!=3.292.0,!=3.292.1,!=3.292.2,!=3.292.3,!=3.292.4,!=3.293.0,!=3.293.1,!=3.293.2,!=3.293.3,!=3.293.4,!=3.294.0,!=3.294.1,!=3.294.2,!=3.294.3,!=3.294.4,!=3.295.0,!=3.
```

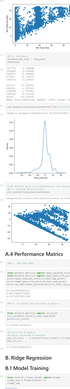
A.2 Prediction1

```
cbg_predictpossum(n_predict=3,n_lag=
                  TRUE,
                  FALSE)
```

```
##  [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

A.3 Assumption

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_residuals,main="Plot of cbg residuals")
qqnorm(cbg$cbg_residuals, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_residuals,col="red",lwd=2)
acf(cbg$cbg_residuals,lwd=2,ylim=c(-1,1))
```




```
cbg$Ljung_Box_test
```

```
##  [1] "Ljung-Box test"
##  [2] "data:  cbg$cbg_residuals"
##  [3] "Q = 1.1e-04, df = 1, p-value = 0.9917"
```

```
cbg$Jarque_Bera_test
```

```
##  [1] "Jarque Bera Test"
##  [2] "data:  cbg$cbg_residuals"
##  [3] "X-squared = 1.2, df = 2, p-value = 0.5488"
```

```
cbg$Shapiro_Wilk_test
```

```
##  [1] "Shapiro-Wilk normality test"
##  [2] "data:  cbg$cbg_residuals"
##  [3] "W = 0.9996, p-value = 0.9000"
```

```
cbg$Ljung_Box_test$p.value[1]
```

```
## [1] 0.9917
```

```
cbg$Jarque_Bera_test$p.value
```

```
## [1] 0.5488
```

```
cbg$Shapiro_Wilk_test$p.value
```

```
## [1] 0.9000
```

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_residuals,main="Plot of cbg residuals")
qqnorm(cbg$cbg_residuals, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_residuals,col="red",lwd=2)
acf(cbg$cbg_residuals,lwd=2,ylim=c(-1,1))
```

A.4 Performance Matrics

```
cbg$cbg_perf_mat
```

```
##  [1] "Error Measures:"
##  [2] "  mnum.error mnum.rmse mnum.mpe mnum.mape"
##  [3] "1  0.0000000  0.0000000 0.0000000 0.0000000"
```

```
cbg$cbg_perf_mat[2,2]
```

```
## [1] 0.0000000
```

```
cbg$cbg_perf_mat[2,4]
```

```
## [1] 0.0000000
```

```
cbg$cbg_perf_mat[2,3]
```

```
## [1] 0.0000000
```

```
cbg$cbg_perf_mat[2,1]
```

```
## [1] 0.0000000
```

B. Ridge Regression

B.1 Model Training

```
cbg$cbg_ridge_model_lambda$ridge_model$lambda
```

```
##  [1] 0.01000000 0.00977237 0.00954993 0.00933254 0.00911991 0.00891251
##  [7] 0.00870964 0.00851138 0.00831763 0.00812831 0.00794328 0.00776247
## [13] 0.00758578 0.00741310 0.00724436 0.00707946 0.00691831 0.00676083
## [19] 0.00660693 0.00645654 0.00630957 0.00616595 0.00602559 0.00588844
## [25] 0.00575440 0.00562341 0.00549540 0.00537032 0.00524807 0.00512861
## [31] 0.00501187 0.00489779 0.00478630 0.00467735 0.00457088 0.00446683
## [37] 0.00436516 0.00426580 0.00416869 0.00407380 0.00398107 0.00389045
## [43] 0.00380189 0.00371535 0.00363078 0.00354813 0.00346737 0.00338844
## [49] 0.00331131 0.00323594 0.00316228 0.00309030 0.00301995 0.00295121
## [55] 0.00288403 0.00281838 0.00275423 0.00269153 0.00263027 0.00257040
## [61] 0.00251189 0.00245471 0.00239883 0.00234422 0.00229087 0.00223872
## [67] 0.00218776 0.00213796 0.00208930 0.00204173 0.00199526 0.00194984
## [73] 0.00190546 0.00186209 0.00181970 0.00177828 0.00173780 0.00169824
## [79] 0.00165959 0.00162181 0.00158489 0.00154882 0.00151356 0.00147910
## [85] 0.00144544 0.00141254 0.00138038 0.00134896 0.00131826 0.00128825
## [91] 0.00125893 0.00123027 0.00120226 0.00117489 0.00114815 0.00112202
## [97] 0.00109648 0.00107152 0.00104713 0.00102329 0.00100000
```

```
cbg$cbg_ridge_model_lambda$ridge_model$coefficients
```

```
##  [1] "Error in cbg$cbg_ridge_model_lambda$ridge_model$coefficients: subscript out of bounds"
```

```
cbg$cbg_ridge_model_lambda$ridge_model$lambda[1]
```

```
## [1] 0.01000000
```

```
cbg$cbg_ridge_model_lambda$ridge_model$coefficients[,1]
```

```
##  [1] "Error in cbg$cbg_ridge_model_lambda$ridge_model$coefficients: subscript out of bounds"
```

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_ridge_model_lambda$ridge_model$lambda,main="Plot of cbg ridge model lambda")
qqnorm(cbg$cbg_ridge_model_lambda$ridge_model$lambda, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_ridge_model_lambda$ridge_model$lambda,col="red",lwd=2)
acf(cbg$cbg_ridge_model_lambda$ridge_model$lambda,lwd=2,ylim=c(-1,1))
```

A.4 Performance Matrics

```
cbg$cbg_ridge_perf_mat
```

```
##  [1] "Error Measures:"
##  [2] "  mnum.error mnum.rmse mnum.mpe mnum.mape"
##  [3] "1  0.0000000  0.0000000 0.0000000 0.0000000"
```

```
cbg$cbg_ridge_perf_mat[2,2]
```

```
## [1] 0.0000000
```

```
cbg$cbg_ridge_perf_mat[2,4]
```

```
## [1] 0.0000000
```

```
cbg$cbg_ridge_perf_mat[2,3]
```

```
## [1] 0.0000000
```

```
cbg$cbg_ridge_perf_mat[2,1]
```

```
## [1] 0.0000000
```

C. Lasso Regression

C.1 Model Training

```
cbg$cbg_lasso_model_lambda$lambda[1]
```

```
## [1] 0.01000000
```

```
cbg$cbg_lasso_model_lambda$lambda[1]
```

```
## [1] 0.01000000
```

```
cbg$cbg_lasso_model_lambda$lambda[1]
```

```
## [1] 0.01000000
```

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_lasso_model_lambda$lambda,main="Plot of cbg lasso model lambda")
qqnorm(cbg$cbg_lasso_model_lambda$lambda, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_lasso_model_lambda$lambda,col="red",lwd=2)
acf(cbg$cbg_lasso_model_lambda$lambda,lwd=2,ylim=c(-1,1))
```

C.2 Prediction

```
cbg$cbg_lasso_model_lambda$p.lasso
```

```
##  [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

C.3 Assumption

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_lasso_model_lambda$lambda,main="Plot of cbg lasso model lambda")
qqnorm(cbg$cbg_lasso_model_lambda$lambda, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_lasso_model_lambda$lambda,col="red",lwd=2)
acf(cbg$cbg_lasso_model_lambda$lambda,lwd=2,ylim=c(-1,1))
```



```
cbg$cbg_lasso_model_lambda$p.lasso
```

```
##  [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

```
cbg$cbg_lasso_model_lambda$p.lasso[1]
```

```
## [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000"
```

```
cbg$cbg_lasso_model_lambda$p.lasso[2]
```

```
## [1] "Predicted 2 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000"
```

```
cbg$cbg_lasso_model_lambda$p.lasso[3]
```

```
## [1] "Predicted 3 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000"
```

C.4 Performance Matrics

```
cbg$cbg_lasso_perf_mat
```

```
##  [1] "Error Measures:"
##  [2] "  mnum.error mnum.rmse mnum.mpe mnum.mape"
##  [3] "1  0.0000000  0.0000000 0.0000000 0.0000000"
```

```
cbg$cbg_lasso_perf_mat[2,2]
```

```
## [1] 0.0000000
```

```
cbg$cbg_lasso_perf_mat[2,4]
```

```
## [1] 0.0000000
```

```
cbg$cbg_lasso_perf_mat[2,3]
```

```
## [1] 0.0000000
```

```
cbg$cbg_lasso_perf_mat[2,1]
```

```
## [1] 0.0000000
```

D. Elastic Net

D.1 Model training

```
cbg$cbg_elastic_model_lambda$lambda[1]
```

```
## [1] 0.01000000
```

```
cbg$cbg_elastic_model_lambda$lambda[1]
```

```
## [1] 0.01000000
```

```
cbg$cbg_elastic_model_lambda$lambda[1]
```

```
## [1] 0.01000000
```

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_elastic_model_lambda$lambda,main="Plot of cbg elastic model lambda")
qqnorm(cbg$cbg_elastic_model_lambda$lambda, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_elastic_model_lambda$lambda,col="red",lwd=2)
acf(cbg$cbg_elastic_model_lambda$lambda,lwd=2,ylim=c(-1,1))
```



```
cbg$cbg_elastic_model_lambda$p.elastic
```

```
##  [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

D.3 Assumptions

```
par(mfrow=c(2,2))
plot.ts(cbg$cbg_elastic_model_lambda$lambda,main="Plot of cbg elastic model lambda")
qqnorm(cbg$cbg_elastic_model_lambda$lambda, cex=1.5,pch=19,col="blue")
qqline(cbg$cbg_elastic_model_lambda$lambda,col="red",lwd=2)
acf(cbg$cbg_elastic_model_lambda$lambda,lwd=2,ylim=c(-1,1))
```



```
cbg$cbg_elastic_model_lambda$p.elastic
```

```
##  [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

```
cbg$cbg_elastic_model_lambda$p.elastic[1]
```

```
## [1] "Predicted 1 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000"
```

```
cbg$cbg_elastic_model_lambda$p.elastic[2]
```

```
## [1] "Predicted 2 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000"
```

```
cbg$cbg_elastic_model_lambda$p.elastic[3]
```

```
## [1] "Predicted 3 - [1,] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000"
```

D.4 Performance Matrics

```
cbg$cbg_elastic_perf_mat
```

```
##  [1] "Error Measures:"
##  [2] "  mnum.error mnum.rmse mnum.mpe mnum.mape"
##  [3] "1  0.0000000  0.0000000 0.0000000 0.0000000"
```

```
cbg$cbg_elastic_perf_mat[2,2]
```

```
## [1] 0.0000000
```

```
cbg$cbg_elastic_perf_mat[2,4]
```

```
## [1] 0.0000000
```

```
cbg$cbg_elastic_perf_mat[2,3]
```

```
## [1] 0.0000000
```

```
cbg$cbg_elastic_perf_mat[2,1]
```

```
## [1] 0.0000000
```

E. SVR : support vector regression

E.1 Model Training

```
cbg$cbg_svr_model
```

```
##  [1] "Error in cbg$cbg_svr_model: object 'cbg$cbg_svr_model' not found"
```

	Model error per	Model scaling per	Kernel	Model intercepts	Gamma	Cost	Epsilon	Degree	Coef0	Number of support vectors	Total scaling
cbg$cbg_svr_model	1.000	1.000	2.000	1.000	0.500	1.000	0.100	3.000	0.000	1.000	1.000