In [1]:
```
import numpy as np
import pandas as pd
```

In [3]:
```
# Importing the required libraries
# Loading the train/test data
# The lowercase alphabets are categorical variables

train = pd.read_csv(r'C:\Users\ushar\OneDrive\Desktop\ML_SL\Mercedes-main\MERCtra
train.head()
```

Out[3]:

	ID	y	X0	X1	X2	X3	X4	X5	X6	X8	...	X375	X376	X377	X378	X379	X380	X382
0	0	130.81	k	v	at	a	d	u	j	o	...	0	0	1	0	0	0	0
1	6	88.53	k	t	av	e	d	y	l	o	...	1	0	0	0	0	0	0
2	7	76.26	az	w	n	c	d	x	j	x	...	0	0	0	0	0	0	1
3	9	80.62	az	t	n	f	d	x	l	e	...	0	0	0	0	0	0	0
4	13	78.02	az	v	n	f	d	h	d	n	...	0	0	0	0	0	0	0

5 rows × 378 columns

In [4]:
```
 train.info()
print('Size of training set')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4209 entries, 0 to 4208
Columns: 378 entries, ID to X385
dtypes: float64(1), int64(369), object(8)
memory usage: 12.1+ MB
Size of training set
```

In [5]:
```
train.shape
```

Out[5]:
```
(4209, 378)
```

In [6]:
```
# Separating y column as this is for pediction output
y_train = train['y'].values
y_train
```

Out[6]:
```
array([130.81,  88.53,  76.26, ..., 109.22,  87.48, 110.85])
```

In [7]:
```
# A lot of columns that have an X
# Let's check for the same
# 376 features with X
colums_x = [c for c in train.columns if 'X' in c]
# colums_x
print(len(colums_x))
print(train[colums_x].dtypes.value_counts())
```

```
376
int64     368
object      8
dtype: int64
```

In [8]:
```
# Looking at the test datset for simiilar features
test = pd.read_csv(r'C:\Users\ushar\OneDrive\Desktop\ML_SL\Mercedes-main\MERCtest
test.head()
```

Out[8]:

	ID	X0	X1	X2	X3	X4	X5	X6	X8	X10	...	X375	X376	X377	X378	X379	X380	X382	X3
0	1	az	v	n	f	d	t	a	w	0	...	0	0	0	1	0	0	0	
1	2	t	b	ai	a	d	b	g	y	0	...	0	0	1	0	0	0	0	
2	3	az	v	as	f	d	a	j	j	0	...	0	0	0	1	0	0	0	
3	4	az	l	n	f	d	z	l	n	0	...	0	0	0	1	0	0	0	
4	5	w	s	as	c	d	y	i	m	0	...	1	0	0	0	0	0	0	

5 rows × 377 columns

In [9]:
```
train.info()
print('Size of training set')
train.shape
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4209 entries, 0 to 4208
Columns: 378 entries, ID to X385
dtypes: float64(1), int64(369), object(8)
memory usage: 12.1+ MB
Size of training set
```

Out[9]:
```
(4209, 378)
```

In [10]:
```
# Creating the final dataset
# Removing unwanted columns (ID); y has been removed earlier


final_column = list(set(train.columns) - set(['ID', 'y']))
```

In [11]:
```
# x_train
x_train = train[final_column]
```

In [12]:
```
# x_test
x_test = test[final_column]
```

In [13]:
```
# Searching for null values
# Creating a function for the same
# There are no missin values

def detect(df):
    if df.isnull().any().any():
        print("Yes")
    else:
        print("No")

detect(x_train)
detect(x_test)
```

```
No
No
```

In [14]:
```
# Removal of columns with a variance of 0
# Column with a variance of 1 is irrelevant so we drop it

for column in final_column:
    check = len(np.unique(x_train[column]))
    if check == 1:
        x_train.drop(column, axis = 1)
        x_test.drop(column, axis = 1)
    if check > 2: # Column is categorical; hence mapping to ordinal measure of va
        mapit = lambda x: sum([ord(digit) for digit in x])
        x_train[column] = x_train[column].apply(mapit)
        x_test[column] = x_test[column].apply(mapit)

x_train.head()
```

```
C:\Users\ushar\Anaconda3\lib\site-packages\ipykernel_launcher.py:8: SettingWith
CopyWarning:
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/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)

C:\Users\ushar\Anaconda3\lib\site-packages\ipykernel_launcher.py:9: SettingWith
CopyWarning:
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/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  if __name__ == '__main__':
```

Out[14]:

	X97	X245	X205	X232	X242	X13	X309	X278	X4	X327	...	X248	X200	X12	X145	X266
0	0	0	0	0	0	1	0	0	100	1	...	0	0	0	0	1
1	0	0	1	0	0	0	0	0	100	0	...	0	0	0	0	0
2	0	0	1	1	0	0	0	0	100	0	...	0	0	0	0	0
3	0	0	1	1	0	0	0	0	100	0	...	0	0	0	0	0
4	0	0	1	1	0	0	0	0	100	0	...	0	0	0	0	0

5 rows × 376 columns

In [15]:
```
# Performing dimensionality reduction with principal components analysis

from sklearn.decomposition import PCA
n_comp = 12
pca = PCA(n_components = n_comp, random_state = 42)
pca_result_train = pca.fit_transform(x_train)
pca_result_test = pca.transform(x_test)
print(pca_result_train)
print(pca_result_test)
```

```
[[-49.08156207  -4.90948084 -17.25085325 ...   1.65805483   0.93297078
    1.67828164]
 [-48.94680383  -7.22674339 -13.7631947  ...  -0.21430578   0.10898115
    0.44857658]
 [ 92.62761708  31.9940341  -26.17503456 ...  -0.62195332   2.92609764
   -0.52644626]
 ...
 [ 89.47970814  20.44554421  48.11999819 ...  -1.27198168  -0.28714485
    2.00813053]
 [ 96.97110845  31.50977186  49.20059282 ...   0.14365998  -0.97975702
    0.99236468]
 [-17.21024322 -14.22166025  55.38091289 ...  -0.28905878  -0.3164053
    0.69151798]]
[[ 9.22615149e+01  3.29260839e+01 -3.01130736e+01 ... -4.11415419e-01
   3.62100818e+00 -1.20772260e+00]
 [-3.48622379e+01  6.87132606e+00 -3.74760829e+01 ...  6.09269838e-01
  -6.95851705e-01 -4.25039849e-01]
 [ 4.36560426e+01 -5.05939489e+01 -6.10591086e+01 ... -3.20468534e-01
   2.60154270e+00 -1.53740953e+00]
 ...
 [-2.52437784e+01 -2.63794193e+01  5.40742341e+01 ...  6.03525945e-01
   2.61354717e-02  3.67582205e-02]
 [ 4.53823778e+01 -6.38062446e+01  3.58666036e+01 ... -9.15190402e-01
  -6.72300519e-01  5.15203104e-01]
 [-4.23807477e+01 -2.52862351e+01  6.10815522e+01 ... -2.98847068e-01
  -9.77130348e-01  5.35160865e-02]]
```

In [20]:
```
pip install xgboost
```

```
Collecting xgboost
  Downloading https://files.pythonhosted.org/packages/3d/1b/83e5dc0021d12884e99
98999945e156cf3628a79dacecaed2ede9f3107cb/xgboost-1.3.3-py3-none-win_amd64.whl
 (https://files.pythonhosted.org/packages/3d/1b/83e5dc0021d12884e9998999945e156
cf3628a79dacecaed2ede9f3107cb/xgboost-1.3.3-py3-none-win_amd64.whl) (95.2MB)
Requirement already satisfied: scipy in c:\users\ushar\anaconda3\lib\site-packa
ges (from xgboost) (1.3.1)
Requirement already satisfied: numpy in c:\users\ushar\anaconda3\lib\site-packa
ges (from xgboost) (1.16.5)
Installing collected packages: xgboost
Successfully installed xgboost-1.3.3
Note: you may need to restart the kernel to use updated packages.
```

In [21]:
```
# ML Modeling with XGboost

import xgboost as xgb
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
```

In [22]:
```
# Splitting the data by 80/20
x_train, x_valid, y_train, y_valid = train_test_split(pca_result_train, y_train,
```

In [29]:
```
# Building the final feature set
f_train = xgb.DMatrix(x_train, label = y_train)
f_valid = xgb.DMatrix(x_valid, label = y_valid)
f_test = xgb.DMatrix(x_test)
f_test = xgb.DMatrix(pca_result_test)
f_train = xgb.DMatrix(x_train, label = y_train)
f_valid = xgb.DMatrix(x_valid, label = y_valid)
f_test = xgb.DMatrix(x_test)
f_test = xgb.DMatrix(pca_result_test)
```

```
<xgboost.core.DMatrix object at 0x00000156D75361C8>
```

In [24]:
```
# Setting the parameters for XGB
params = {}
params['objective'] = 'reg:linear'
params['eta'] = 0.02
params['max_depth'] = 4
```

In [25]:
```
# Predicting the score
# Creating a function for the same

def scorer(m, w):
    labels = w.get_label()
    return 'r2', r2_score(labels, m)

final_set = [(f_train, 'train'), (f_valid, 'valid')]

P = xgb.train(params, f_train, 1000, final_set, early_stopping_rounds=50, feval=
```

```
[16:38:53] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.3.
0/src/objective/regression_obj.cu:170: reg:linear is now deprecated in favor of
reg:squarederror.
[0]     train-rmse:98.99695     train-r2:-59.49733      valid-rmse:98.88884
valid-r2:-61.82690
[10]    train-rmse:81.14409     train-r2:-39.64492      valid-rmse:81.07848
valid-r2:-41.23399
[20]    train-rmse:66.59753     train-r2:-26.37844      valid-rmse:66.55611
valid-r2:-27.45948
[30]    train-rmse:54.75786     train-r2:-17.50911      valid-rmse:54.73430
valid-r2:-18.24731
[40]    train-rmse:45.13932     train-r2:-11.57774      valid-rmse:45.13527
valid-r2:-12.08830
[50]    train-rmse:37.34557     train-r2:-7.60935       valid-rmse:37.36176
valid-r2:-7.96821
[60]    train-rmse:31.05138     train-r2:-4.95188       valid-rmse:31.08614
valid-r2:-5.20847
[70]    train-rmse:25.98506     train-r2:-3.16812       valid-rmse:26.03608
valid-r2:-3.35514
[80]    train-rmse:21.94074     train-r2:-1.97163       valid-rmse:22.00434
valid-r2:-2.11077
[90]    train-rmse:18.73487     train-r2:-1.16667       valid-rmse:18.82125
valid-r2:-1.27587
[100]   train-rmse:16.22054     train-r2:-0.62414       valid-rmse:16.33785
valid-r2:-0.71491
[110]   train-rmse:14.28336     train-r2:-0.25937       valid-rmse:14.42612
valid-r2:-0.33706
[120]   train-rmse:12.81045     train-r2:-0.01303       valid-rmse:12.97342
valid-r2:-0.08134
[130]   train-rmse:11.69324     train-r2:0.15596        valid-rmse:11.89772
valid-r2:0.09055
[140]   train-rmse:10.86476     train-r2:0.27133        valid-rmse:11.11423
valid-r2:0.20639
[150]   train-rmse:10.24847     train-r2:0.35165        valid-rmse:10.54874
valid-r2:0.28509
[160]   train-rmse:9.79641      train-r2:0.40758        valid-rmse:10.15288
valid-r2:0.33774
[170]   train-rmse:9.45956      train-r2:0.44762        valid-rmse:9.87473
valid-r2:0.37353
[180]   train-rmse:9.22368      train-r2:0.47483        valid-rmse:9.68225
valid-r2:0.39771
[190]   train-rmse:9.04458      train-r2:0.49502        valid-rmse:9.54260
valid-r2:0.41496
[200]   train-rmse:8.91474      train-r2:0.50942        valid-rmse:9.44861
valid-r2:0.42643
```

```
[210]   train-rmse:8.81488      train-r2:0.52035        valid-rmse:9.38823
valid-r2:0.43374
[220]   train-rmse:8.74025      train-r2:0.52844        valid-rmse:9.34325
valid-r2:0.43915
[230]   train-rmse:8.66565      train-r2:0.53645        valid-rmse:9.30757
valid-r2:0.44343
[240]   train-rmse:8.61011      train-r2:0.54238        valid-rmse:9.28263
valid-r2:0.44640
[250]   train-rmse:8.55749      train-r2:0.54795        valid-rmse:9.26350
valid-r2:0.44868
[260]   train-rmse:8.52122      train-r2:0.55178        valid-rmse:9.24905
valid-r2:0.45040
[270]   train-rmse:8.48235      train-r2:0.55585        valid-rmse:9.23762
valid-r2:0.45176
[280]   train-rmse:8.44120      train-r2:0.56015        valid-rmse:9.22816
valid-r2:0.45288
[290]   train-rmse:8.40780      train-r2:0.56363        valid-rmse:9.22413
valid-r2:0.45336
[300]   train-rmse:8.36885      train-r2:0.56766        valid-rmse:9.21558
valid-r2:0.45437
[310]   train-rmse:8.34404      train-r2:0.57022        valid-rmse:9.21221
valid-r2:0.45477
[320]   train-rmse:8.31110      train-r2:0.57361        valid-rmse:9.21144
valid-r2:0.45486
[330]   train-rmse:8.28332      train-r2:0.57645        valid-rmse:9.20841
valid-r2:0.45522
[340]   train-rmse:8.25624      train-r2:0.57922        valid-rmse:9.20450
valid-r2:0.45568
[350]   train-rmse:8.22516      train-r2:0.58238        valid-rmse:9.20661
valid-r2:0.45543
[360]   train-rmse:8.19375      train-r2:0.58556        valid-rmse:9.20486
valid-r2:0.45564
[370]   train-rmse:8.16996      train-r2:0.58797        valid-rmse:9.20428
valid-r2:0.45571
[380]   train-rmse:8.14136      train-r2:0.59085        valid-rmse:9.20379
valid-r2:0.45577
[390]   train-rmse:8.11330      train-r2:0.59366        valid-rmse:9.20236
valid-r2:0.45594
[400]   train-rmse:8.08825      train-r2:0.59617        valid-rmse:9.20062
valid-r2:0.45614
[410]   train-rmse:8.06259      train-r2:0.59873        valid-rmse:9.19947
valid-r2:0.45628
[420]   train-rmse:8.04465      train-r2:0.60051        valid-rmse:9.20055
valid-r2:0.45615
[430]   train-rmse:8.02389      train-r2:0.60257        valid-rmse:9.19911
valid-r2:0.45632
[440]   train-rmse:7.99466      train-r2:0.60546        valid-rmse:9.19710
valid-r2:0.45656
[450]   train-rmse:7.97439      train-r2:0.60746        valid-rmse:9.19722
valid-r2:0.45654
[460]   train-rmse:7.95370      train-r2:0.60949        valid-rmse:9.19724
valid-r2:0.45654
[470]   train-rmse:7.92327      train-r2:0.61247        valid-rmse:9.19759
valid-r2:0.45650
[480]   train-rmse:7.90695      train-r2:0.61407        valid-rmse:9.19671
valid-r2:0.45660
[490]   train-rmse:7.87775      train-r2:0.61691        valid-rmse:9.19139
```

```
valid-r2:0.45723
[500]   train-rmse:7.86077      train-r2:0.61856        valid-rmse:9.19009
valid-r2:0.45739
[510]   train-rmse:7.84321      train-r2:0.62026        valid-rmse:9.19076
valid-r2:0.45731
[520]   train-rmse:7.82734      train-r2:0.62180        valid-rmse:9.18926
valid-r2:0.45748
[530]   train-rmse:7.80564      train-r2:0.62389        valid-rmse:9.19036
valid-r2:0.45735
[540]   train-rmse:7.78404      train-r2:0.62597        valid-rmse:9.18689
valid-r2:0.45776
[550]   train-rmse:7.76428      train-r2:0.62787        valid-rmse:9.18781
valid-r2:0.45766
[560]   train-rmse:7.74140      train-r2:0.63006        valid-rmse:9.18302
valid-r2:0.45822
[570]   train-rmse:7.70738      train-r2:0.63330        valid-rmse:9.17475
valid-r2:0.45920
[580]   train-rmse:7.67793      train-r2:0.63610        valid-rmse:9.17692
valid-r2:0.45894
[590]   train-rmse:7.65826      train-r2:0.63796        valid-rmse:9.17833
valid-r2:0.45877
[600]   train-rmse:7.63126      train-r2:0.64051        valid-rmse:9.17547
valid-r2:0.45911
[610]   train-rmse:7.60100      train-r2:0.64336        valid-rmse:9.17197
valid-r2:0.45952
[620]   train-rmse:7.57856      train-r2:0.64546        valid-rmse:9.16943
valid-r2:0.45982
[630]   train-rmse:7.55252      train-r2:0.64789        valid-rmse:9.16842
valid-r2:0.45994
[640]   train-rmse:7.53341      train-r2:0.64967        valid-rmse:9.16989
valid-r2:0.45977
[650]   train-rmse:7.50746      train-r2:0.65208        valid-rmse:9.17039
valid-r2:0.45971
[660]   train-rmse:7.48546      train-r2:0.65412        valid-rmse:9.17440
valid-r2:0.45924
[670]   train-rmse:7.45836      train-r2:0.65662        valid-rmse:9.17437
valid-r2:0.45924
[680]   train-rmse:7.44151      train-r2:0.65817        valid-rmse:9.17481
valid-r2:0.45919
[681]   train-rmse:7.43931      train-r2:0.65837        valid-rmse:9.17555
valid-r2:0.45910
```

In [26]:
```
# Predicting on test set
p_test = P.predict(f_test)
p_test
```

Out[26]:
```
array([ 79.95739 ,  96.367615,  81.17032 , ...,  98.88627 , 106.84664 ,
        95.02746 ], dtype=float32)
```

In [27]:
```
Predicted_Data = pd.DataFrame()
Predicted_Data['y'] = p_test
Predicted_Data.head()
```

Out[27]:

	y
0	79.957390
1	96.367615
2	81.170319
3	77.602242
4	109.650452

In [ ]:
```

```