In [51]:

```
#importing the libraries
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA # for dementionality reduction
```

In [52]:

```
# Reading dataset

df_train = pd.read_csv("train.csv")
df_train.shape
df_test = pd.read_csv('test.csv')
```

In [53]:

```
df_train.head(3)
```

Out[53]:

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

3 rows × 378 columns

In [54]:

```
# Collecting Y column values in the array
y_train = df_train['y'].values
```

In [55]:

```
y_train
# this we will use to learn the prediction output
```

Out[55]:

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

In [56]:

```
cols = [c for c in df_train.columns if 'X' in c]
print('Number of features: {}'.format(len(cols)))
print('Feature types:')
df_train[cols].dtypes.value_counts()
```

```
Number of features: 376
Feature types:
```

Out[56]:

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

In [57]:

```
# Understanding the data types by iterating all the columns having X in the name of the Col

counts = [[], [], []]
for c in cols:
    typ = df_train[c].dtype
    uniq = len(np.unique(df_train[c]))
    if uniq == 1:
        counts[0].append(c)
    elif uniq == 2 and typ == np.int64:
        counts[1].append(c)
    else:
        counts[2].append(c)

print('Constant features: {} Binary features: {} Categorical features: {}\n'.format(*[len(c
print('Constant features:', counts[0])
print('Categorical features:', counts[2])
```

```
Constant features: 12 Binary features: 356 Categorical features: 8

Constant features: ['X11', 'X93', 'X107', 'X233', 'X235', 'X268', 'X289', 'X
290', 'X293', 'X297', 'X330', 'X347']
Categorical features: ['X0', 'X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X8']
```

In [58]:

```
# removing the Id and Y data set from train and test data set
Columns_new = list(set(df_train.columns) - set(['ID','y']))
```

In [59]:

```
y_train = df_train['y'].values
id_test = df_test['ID'].values
x_train = df_train[Columns_new]
x_test = df_test[Columns_new]
```

In [60]:

```
#Check for null and unique values for test and train sets.
def CHK(df):
    if df.isnull().any().any():
        print("no missing values")
    else:
        print("no missing values")
CHK(x_train)
CHK(x_test)
```

```
no missing values
no missing values
```

In [61]:

```
##
#If for any column(s), the variance is equal to zero, then you need to remove those variabl

for column in Columns_new:
    cardinality = len(np.unique(x_train[column]))
    if cardinality == 1:
        x_train.drop(column, axis=1) # Column with only one
    # value is useless so we drop it
    x_test.drop(column, axis=1)
    if cardinality > 2: # Column is categorical
        mapper = lambda x: sum([ord(digit) for digit in x])
        x_train[column] = x_train[column].apply(mapper)
        x_test[column] = x_test[column].apply(mapper)
x_train.head()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:12: Setting
WithCopyWarning:
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: http://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (http://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
rsus-a-copy)
  if sys.path[0] == '':
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:13: Setting
WithCopyWarning:
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: http://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (http://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
rsus-a-copy)
  del sys.path[0]
```

Out[61]:

	X361	X374	X224	X244	X10	X205	X348	X240	X261	X263	...	X34	X300	X168	X238
0	1	0	0	0	0	0	0	0	0	1	...	0	0	0	0
1	1	0	0	0	0	1	1	0	0	1	...	0	0	0	1
2	1	0	1	1	0	1	1	0	0	0	...	0	0	0	0
3	1	0	0	1	0	1	1	0	0	0	...	0	0	0	0
4	1	0	0	0	0	1	1	0	0	0	...	0	0	0	0

5 rows × 376 columns

In [62]:

```
x_train[cols].dtypes.value_counts()
#Perform dimensionality reduction
# Linear dimensionality reduction using Singular Value Decomposition of
# the data to project it to a lower dimensional space.
n_comp = 12
pca = PCA(n_components=n_comp, random_state=20)
pca2_results_train = pca.fit_transform(x_train)
pca2_results_test = pca.transform(x_test)
```

In [63]:

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

x_train, x_valid, y_train, y_valid = train_test_split(pca2_results_train,y_train, test_size

d_train = xgb.DMatrix(x_train, label=y_train)
d_valid = xgb.DMatrix(x_valid, label=y_valid)
#d_test = xgb.DMatrix(x_test)
d_test = xgb.DMatrix(pca2_results_test)

params = {}
params['objective'] = 'reg:linear'
params['eta'] = 0.02
params['max_depth'] = 4

def xgb_r2_score(preds, dtrain):
    labels = dtrain.get_label()
    return 'r2', r2_score(labels, preds)

watchlist = [(d_train, 'train'), (d_valid, 'valid')]

clf = xgb.train(params, d_train,1000, watchlist, early_stopping_rounds=50,
                feval=xgb_r2_score, maximize=True, verbose_eval=10)

# Predict your test_df values using XGBoost.

p_test = clf.predict(d_test)

sub = pd.DataFrame()
sub['ID'] = id_test
sub['y'] = p_test
sub.to_csv('xgb.csv', index=False)

sub.head()
```

```
[15:43:43] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_
1.1.0/src/objective/regression_obj.cu:170: reg:linear is now deprecated in f
avor of reg:squarederror.
[0]     train-rmse:99.04150     valid-rmse:98.70239     train-r2:-59.47071
valid-r2:-61.96931
Multiple eval metrics have been passed: 'valid-r2' will be used for early st
opping.

Will train until valid-r2 hasn't improved in 50 rounds.
[10]    train-rmse:81.18310     valid-rmse:80.84457     train-r2:-39.62955
valid-r2:-41.24502
[20]    train-rmse:66.63157     valid-rmse:66.31034     train-r2:-26.36975
valid-r2:-27.42080
[30]    train-rmse:54.79031     valid-rmse:54.48565     train-r2:-17.50624
valid-r2:-18.18837
[40]    train-rmse:45.17240     valid-rmse:44.87848     train-r2:-11.57931
valid-r2:-12.01817
[50]    train-rmse:37.38147     valid-rmse:37.10155     train-r2:-7.61437
valid-r2:-7.89729
[60]    train-rmse:31.08475     valid-rmse:30.81077     train-r2:-4.95669
valid-r2:-5.13591
[70]    train-rmse:26.02225     valid-rmse:25.75129     train-r2:-3.17446
```

```
valid-r2:-3.28620
[80]    train-rmse:21.97581     valid-rmse:21.70392     train-r2:-1.97715
valid-r2:-2.04474
[90]    train-rmse:18.76038     valid-rmse:18.51956     train-r2:-1.16967
valid-r2:-1.21684
[100]   train-rmse:16.23822     valid-rmse:16.03058     train-r2:-0.62550
valid-r2:-0.66101
[110]   train-rmse:14.28560     valid-rmse:14.12169     train-r2:-0.25808
valid-r2:-0.28898
[120]   train-rmse:12.79751     valid-rmse:12.68434     train-r2:-0.00963
valid-r2:-0.03994
[130]   train-rmse:11.66980     valid-rmse:11.61849     train-r2:0.16047
valid-r2:0.12748
[140]   train-rmse:10.82814     valid-rmse:10.84801     train-r2:0.27720
valid-r2:0.23937
[150]   train-rmse:10.19603     valid-rmse:10.29407     train-r2:0.35913
valid-r2:0.31507
[160]   train-rmse:9.74740      valid-rmse:9.91110      train-r2:0.41428
valid-r2:0.36508
[170]   train-rmse:9.42132      valid-rmse:9.64146      train-r2:0.45282
valid-r2:0.39916
[180]   train-rmse:9.17672      valid-rmse:9.45435      train-r2:0.48086
valid-r2:0.42225
[190]   train-rmse:8.98255      valid-rmse:9.33140      train-r2:0.50260
valid-r2:0.43718
[200]   train-rmse:8.84733      valid-rmse:9.24928      train-r2:0.51746
valid-r2:0.44705
[210]   train-rmse:8.74914      valid-rmse:9.19070      train-r2:0.52811
valid-r2:0.45403
[220]   train-rmse:8.67841      valid-rmse:9.14922      train-r2:0.53571
valid-r2:0.45894
[230]   train-rmse:8.61728      valid-rmse:9.11569      train-r2:0.54223
valid-r2:0.46290
[240]   train-rmse:8.57079      valid-rmse:9.09691      train-r2:0.54715
valid-r2:0.46511
[250]   train-rmse:8.52579      valid-rmse:9.08043      train-r2:0.55190
valid-r2:0.46705
[260]   train-rmse:8.49246      valid-rmse:9.06914      train-r2:0.55539
valid-r2:0.46837
[270]   train-rmse:8.46504      valid-rmse:9.05737      train-r2:0.55826
valid-r2:0.46975
[280]   train-rmse:8.44270      valid-rmse:9.05279      train-r2:0.56059
valid-r2:0.47029
[290]   train-rmse:8.41492      valid-rmse:9.04791      train-r2:0.56347
valid-r2:0.47086
[300]   train-rmse:8.38943      valid-rmse:9.04598      train-r2:0.56611
valid-r2:0.47109
[310]   train-rmse:8.36044      valid-rmse:9.04093      train-r2:0.56911
valid-r2:0.47168
[320]   train-rmse:8.33151      valid-rmse:9.03678      train-r2:0.57208
valid-r2:0.47216
[330]   train-rmse:8.30380      valid-rmse:9.03222      train-r2:0.57493
valid-r2:0.47269
[340]   train-rmse:8.27550      valid-rmse:9.03083      train-r2:0.57782
valid-r2:0.47286
[350]   train-rmse:8.25459      valid-rmse:9.02774      train-r2:0.57995
valid-r2:0.47322
[360]   train-rmse:8.22770      valid-rmse:9.02454      train-r2:0.58268
valid-r2:0.47359
[370]   train-rmse:8.20476      valid-rmse:9.02231      train-r2:0.58500
valid-r2:0.47385
```

```
[380]   train-rmse:8.17463      valid-rmse:9.02108      train-r2:0.58805
valid-r2:0.47399
[390]   train-rmse:8.14605      valid-rmse:9.01614      train-r2:0.59092
valid-r2:0.47457
[400]   train-rmse:8.11499      valid-rmse:9.01335      train-r2:0.59404
valid-r2:0.47490
[410]   train-rmse:8.08766      valid-rmse:9.01116      train-r2:0.59677
valid-r2:0.47515
[420]   train-rmse:8.06851      valid-rmse:9.01002      train-r2:0.59867
valid-r2:0.47528
[430]   train-rmse:8.04146      valid-rmse:9.01051      train-r2:0.60136
valid-r2:0.47523
[440]   train-rmse:8.01826      valid-rmse:9.00841      train-r2:0.60366
valid-r2:0.47547
[450]   train-rmse:7.99330      valid-rmse:9.00465      train-r2:0.60612
valid-r2:0.47591
[460]   train-rmse:7.96281      valid-rmse:9.00382      train-r2:0.60912
valid-r2:0.47601
[470]   train-rmse:7.92897      valid-rmse:9.00038      train-r2:0.61244
valid-r2:0.47641
[480]   train-rmse:7.90437      valid-rmse:9.00123      train-r2:0.61484
valid-r2:0.47631
[490]   train-rmse:7.87299      valid-rmse:8.99948      train-r2:0.61789
valid-r2:0.47651
[500]   train-rmse:7.85183      valid-rmse:8.99876      train-r2:0.61994
valid-r2:0.47659
[510]   train-rmse:7.82470      valid-rmse:8.99847      train-r2:0.62256
valid-r2:0.47663
[520]   train-rmse:7.80249      valid-rmse:8.99821      train-r2:0.62470
valid-r2:0.47666
[530]   train-rmse:7.78171      valid-rmse:8.99747      train-r2:0.62670
valid-r2:0.47674
[540]   train-rmse:7.76155      valid-rmse:8.99624      train-r2:0.62863
valid-r2:0.47689
[550]   train-rmse:7.74025      valid-rmse:8.99394      train-r2:0.63067
valid-r2:0.47715
[560]   train-rmse:7.71530      valid-rmse:8.99372      train-r2:0.63304
valid-r2:0.47718
[570]   train-rmse:7.69898      valid-rmse:8.99432      train-r2:0.63459
valid-r2:0.47711
[580]   train-rmse:7.68065      valid-rmse:8.99173      train-r2:0.63633
valid-r2:0.47741
[590]   train-rmse:7.66357      valid-rmse:8.99054      train-r2:0.63795
valid-r2:0.47755
[600]   train-rmse:7.63724      valid-rmse:8.99124      train-r2:0.64043
valid-r2:0.47747
[610]   train-rmse:7.62298      valid-rmse:8.99168      train-r2:0.64177
valid-r2:0.47742
[620]   train-rmse:7.60773      valid-rmse:8.99306      train-r2:0.64320
valid-r2:0.47726
[630]   train-rmse:7.59036      valid-rmse:8.99265      train-r2:0.64483
valid-r2:0.47730
[640]   train-rmse:7.57303      valid-rmse:8.99169      train-r2:0.64645
valid-r2:0.47741
Stopping. Best iteration:
[593]   train-rmse:7.65176      valid-rmse:8.98975      train-r2:0.63906
valid-r2:0.47764

[15:43:45] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_
1.1.0/src/objective/regression_obj.cu:170: reg:linear is now deprecated in f
avor of reg:squarederror.
```

Out[63]:

	ID	y
0	1	79.049286
1	2	96.350510
2	3	81.374931
3	4	77.107979
4	5	111.473167

In [ ]: