Following actions should be performed:

If for any column(s), the variance is equal to zero, then you need to remove those variable(s). ¶

Check for null and unique values for test and train sets.

Apply label encoder.

Perform dimensionality reduction.

Predict your test_df values using XGBoost.

In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
df_train = pd.read_csv('./Dataset/train.csv')
print('Size of training set: {} rows and {} columns'
      .format(*df_train.shape))
df_train.head()
```

```
Size of training set: 4209 rows and 378 columns
```

Out[2]:

	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	9	80.62	az	t	n	f	d	x	l	e	...	0	0	0	0	0	0	
4	13	78.02	az	v	n	f	d	h	d	n	...	0	0	0	0	0	0	

5 rows × 378 columns

In [3]:

```
y_train = df_train['y'].values
```

In [4]:

```
df_test = pd.read_csv('./Dataset/test.csv')
```

In [5]:

```
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[5]:

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

In [6]:

```
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) for c in counts]))
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 [7]:

```
usable_columns = list(set(df_train.columns) - set(['ID', 'y']))
y_train = df_train['y'].values
id_test = df_test['ID'].values

x_train = df_train[usable_columns]
x_test = df_test[usable_columns]
```

In [8]:

```
def check_missing_values(df):
    if df.isnull().any().any():
        print("There are missing values in the dataframe")
    else:
        print("There are no missing values in the dataframe")
check_missing_values(x_train)
check_missing_values(x_test)
```

```
There are no missing values in the dataframe
There are no missing values in the dataframe
```

In [9]:

```
for column in usable_columns:
    cardinality = len(np.unique(x_train[column]))
    if cardinality == 1:
        x_train.drop(column, axis=1)
        x_test.drop(column, axis=1)
    if cardinality > 2:
        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:\users\aditya\appdata\local\programs\python\python37\lib\site-packages\ipy
kernel_launcher.py:9: 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_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  if __name__ == '__main__':
c:\users\aditya\appdata\local\programs\python\python37\lib\site-packages\ipy
kernel_launcher.py:10: 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_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  # Remove the CWD from sys.path while we load stuff.
```

Out[9]:

	X20	X195	X312	X354	X241	X348	X112	X109	X19	X141	...	X266	X99	X244	X153
0	0	0	0	1	0	0	0	0	0	0	...	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0
2	0	0	0	1	1	1	0	0	0	0	...	0	0	1	0
3	0	0	0	0	1	1	0	0	0	0	...	0	0	1	0
4	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0

5 rows × 376 columns

In [10]:

```
print('Feature types:')
x_train[cols].dtypes.value_counts()
```

```
Feature types:
```

Out[10]:

```
int64    376
dtype: int64
```

In [11]:

```
from sklearn.decomposition import PCA
n_comp = 12
pca = PCA(n_components=n_comp, random_state=420)
pca2_results_train = pca.fit_transform(x_train)
pca2_results_test = pca.transform(x_test)
```

In [12]:

```
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=0.25,
        random_state=0)

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

params = {}
params['objective'] = 'reg:linear'
params['eta'] = 0.01
params['max_depth'] = 2

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)


p_test = clf.predict(d_test)

sub = pd.DataFrame()
sub['ID'] = id_test
sub['y'] = p_test

sub.head()
```

```
[00:26:43] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release
_1.1.0/src/objective/regression_obj.cu:170: reg:linear is now deprecated i
n favor of reg:squarederror.
[0]     train-rmse:100.03477    valid-rmse:99.78257     train-r2:-65.23608
valid-r2:-51.54268
Multiple eval metrics have been passed: 'valid-r2' will be used for early
stopping.

Will train until valid-r2 hasn't improved in 50 rounds.
[10]    train-rmse:90.59826     valid-rmse:90.37879     train-r2:-53.32911
valid-r2:-42.10581
[20]    train-rmse:82.07540     valid-rmse:81.89900     train-r2:-43.58808
valid-r2:-34.39647
[30]    train-rmse:74.37967     valid-rmse:74.24620     train-r2:-35.61856
valid-r2:-28.09050
[40]    train-rmse:67.43328     valid-rmse:67.33838     train-r2:-29.09826
valid-r2:-22.92919
[50]    train-rmse:61.16568     valid-rmse:61.10699     train-r2:-23.76330
valid-r2:-18.70537
```

In [ ]:

```

```