```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


train_ds = pd.read_csv("C:/Users/Nabee/Downloads/Machine-Learning--
Projects-master/Machine-Learning--Projects-master/Projects/Projects for
Submission/Project 1 - Mercedes-Benz Greener Manufacturing/Dataset for
the project/train/train.csv")

test_ds = pd.read_csv("C:/Users/Nabee/Downloads/Machine-Learning--
Projects-master/Machine-Learning--Projects-master/Projects/Projects for
Submission/Project 1 - Mercedes-Benz Greener Manufacturing/Dataset for
the project/test/test.csv")


train_ds.head()


test_ds.head()


print(train_ds.shape)
print(test_ds.shape)


train_ds.describe()


train_ds.var()


train_ds.var()==0


zero_variance = train_ds.var()[train_ds.var()==0].index.values


zero_variance


train_ds = train_ds.drop(zero_variance, axis=1)


train_ds.shape


train_ds = train_ds.drop(['ID'], axis=1)


train_ds.shape


train_ds.head()


train_ds.isnull().sum().values
```

```
test_ds.isnull().sum().values
```

```
train_ds.nunique().values
```

```
test_ds.nunique().values
```

```
from sklearn import preprocessing
```

```
label_encoder = preprocessing.LabelEncoder()
```

```
object_datatypes = train_ds.select_dtypes(include=[object]).columns
```

```
object_datatypes
```

```
train_ds['X0'] = label_encoder.fit_transform(train_ds['X0'])
```

```
train_ds['X1'] = label_encoder.fit_transform(train_ds['X1'])
train_ds['X2'] = label_encoder.fit_transform(train_ds['X2'])
train_ds['X3'] = label_encoder.fit_transform(train_ds['X3'])
train_ds['X4'] = label_encoder.fit_transform(train_ds['X4'])
train_ds['X5'] = label_encoder.fit_transform(train_ds['X5'])
train_ds['X6'] = label_encoder.fit_transform(train_ds['X6'])
train_ds['X8'] = label_encoder.fit_transform(train_ds['X8'])
```

```
train_ds.head()
```

```
train_ds.select_dtypes(include=[object]).columns
```

```
test_ds.head()
```

```
test_ds.select_dtypes(include=[object]).columns
```

```
test_ds['X0'] = label_encoder.fit_transform(test_ds['X0'])
test_ds['X1'] = label_encoder.fit_transform(test_ds['X1'])
test_ds['X2'] = label_encoder.fit_transform(test_ds['X2'])
test_ds['X3'] = label_encoder.fit_transform(test_ds['X3'])
test_ds['X4'] = label_encoder.fit_transform(test_ds['X4'])
test_ds['X5'] = label_encoder.fit_transform(test_ds['X5'])
test_ds['X6'] = label_encoder.fit_transform(test_ds['X6'])
test_ds['X8'] = label_encoder.fit_transform(test_ds['X8'])
```

```
test_ds.select_dtypes(include=[object]).columns


test_ds = test_ds.drop('ID', axis=1)


test_ds.head()


from sklearn.model_selection import train_test_split


x=train_ds.drop('y', axis=1)
y=train_ds.y


xtrain, xtest, ytrain, ytest = train_test_split(x, y, random_state=0)


print(xtrain.shape)
print(xtest.shape)
print(ytrain.shape)
print(ytest.shape)


from sklearn.decomposition import PCA


pca_xtrain = PCA(n_components=0.95)
pca_xtrain.fit(xtrain)


pca_xtrain_transformed = pca_xtrain.transform(xtrain)
print(pca_xtrain_transformed.shape)


pca_xtest = PCA(n_components=0.95)
pca_xtest.fit(xtest)


pca_xtest_transformed = pca_xtest.transform(xtest)
print(pca_xtest_transformed.shape)


pca_xtrain.explained_variance_ratio_.shape


np.sum(pca_xtrain.explained_variance_ratio_)


# PCA of train_ds


pca_train_ds = PCA(n_components=0.95)
pca_train_ds.fit(train_ds)
```

```
x_train_transformed = pca_train_ds.transform(train_ds)


print(x_train_transformed.shape)


# PCA of test_ds


pca_test_ds = PCA(n_components=0.95)
pca_test_ds.fit(test_ds)


pca_test_ds_transformed = pca_test_ds.transform(test_ds)
print(pca_test_ds_transformed.shape)


pca_test_ds.explained_variance_ratio_.shape


np.sum(pca_test_ds.explained_variance_ratio_)


import xgboost as xgb


my_model = xgb.XGBRegressor()


my_model.fit(pca_xtrain_transformed, ytrain)


ypred = my_model.predict(pca_test_ds_transformed)


ypred
```