Mercedes-Benz Greener Manufacturing .

November 21, 2022

1 Import required libraries

[1]:
```
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
```

2 Read the data from train.csv

[2]:
```
df_train = pd.read_csv('train1.csv')
```

[3]:
```
df_train.head()
```

[3]:

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

	X380	X382	X383	X384	X385
0	0	0	0	0	0
1	0	0	0	0	0
2	0	1	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

[5 rows x 378 columns]

[4]:
```
df_train.shape
```

[4]: (4209, 378)

[5]:
```
df_train.size
```

[5]: 1591002

[6]:
```
df_train.describe()
```

[6]:

	ID	y	X10	X11	X12	\
count	4209.000000	4209.000000	4209.000000	4209.0	4209.000000	
mean	4205.960798	100.669318	0.013305	0.0	0.075077	
std	2437.608688	12.679381	0.114590	0.0	0.263547	
min	0.000000	72.110000	0.000000	0.0	0.000000	
25%	2095.000000	90.820000	0.000000	0.0	0.000000	
50%	4220.000000	99.150000	0.000000	0.0	0.000000	
75%	6314.000000	109.010000	0.000000	0.0	0.000000	
max	8417.000000	265.320000	1.000000	0.0	1.000000	

	X13	X14	X15	X16	X17	…	\
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000	…	
mean	0.057971	0.428130	0.000475	0.002613	0.007603	…	
std	0.233716	0.494867	0.021796	0.051061	0.086872	…	
min	0.000000	0.000000	0.000000	0.000000	0.000000	…	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	…	
50%	0.000000	0.000000	0.000000	0.000000	0.000000	…	
75%	0.000000	1.000000	0.000000	0.000000	0.000000	…	
max	1.000000	1.000000	1.000000	1.000000	1.000000	…	

	X375	X376	X377	X378	X379	\
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000	
mean	0.318841	0.057258	0.314802	0.020670	0.009503	
std	0.466082	0.232363	0.464492	0.142294	0.097033	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	0.000000	0.000000	
75%	1.000000	0.000000	1.000000	0.000000	0.000000	
max	1.000000	1.000000	1.000000	1.000000	1.000000	

	X380	X382	X383	X384	X385
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000
mean	0.008078	0.007603	0.001663	0.000475	0.001426
std	0.089524	0.086872	0.040752	0.021796	0.037734
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000

[8 rows x 370 columns]

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

[7]:
```
df_train.var()
```

[7]:
```
ID      5.941936e+06
y       1.607667e+02
X10     1.313092e-02
X11     0.000000e+00
X12     6.945713e-02
            …
X380    8.014579e-03
X382    7.546747e-03
X383    1.660732e-03
X384    4.750593e-04
X385    1.423823e-03
Length: 370, dtype: float64
```

[8]:
```
df_train.var()==0
```

[8]:
```
ID      False
y       False
X10     False
X11      True
X12     False
        …
X380    False
X382    False
X383    False
X384    False
X385    False
Length: 370, dtype: bool
```

[9]:
```
(df_train.var()==0).values
```

[9]:
```
array([False, False, False,  True, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False,  True, False, False, False, False, False, False,
       False, False, False, False, False, False, False,  True, False,
```

```
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False,  True, False,  True, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False,  True, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False,  True,  True, False, False,
        True, False, False, False,  True, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
        True, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False,  True,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False])
```

[10]:
```
variance_with_zero = df_train.var()[df_train.var()==0].index.values
variance_with_zero
```

[10]:
```
array(['X11', 'X93', 'X107', 'X233', 'X235', 'X268', 'X289', 'X290',
       'X293', 'X297', 'X330', 'X347'], dtype=object)
```

[11]:
```
df_train = df_train.drop(variance_with_zero, axis=1)
```

[12]:
```
print(df_train.shape)
```

```
(4209, 366)
```

[13]:
```
df_train = df_train.drop(['ID'], axis=1)
```

[14]:
```
df_train.head()
```

[14]:

	y	X0	X1	X2	X3	X4	X5	X6	X8	X10	…	X375	X376	X377	X378	X379	\
0	130.81	k	v	at	a	d	u	j	o	0	…	0	0	1	0	0	
1	88.53	k	t	av	e	d	y	l	o	0	…	1	0	0	0	0	
2	76.26	az	w	n	c	d	x	j	x	0	…	0	0	0	0	0	
3	80.62	az	t	n	f	d	x	l	e	0	…	0	0	0	0	0	
4	78.02	az	v	n	f	d	h	d	n	0	…	0	0	0	0	0	

	X380	X382	X383	X384	X385
0	0	0	0	0	0
1	0	0	0	0	0
2	0	1	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

[5 rows x 365 columns]

4 Read the data from test.csv

[15]:
```
df_test = pd.read_csv('test1.csv')
```

[16]:
```
df_test.head()
```

[16]:

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

	X382	X383	X384	X385
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

[5 rows x 377 columns]

[17]:
```
df_test.shape
```

[17]: (4209, 377)

[18]:
```
df_test.size
```

[18]: 1586793

[19]:
```
df_test.describe()
```

[19]:

	ID	X10	X11	X12	X13	\
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000	
mean	4211.039202	0.019007	0.000238	0.074364	0.061060	
std	2423.078926	0.136565	0.015414	0.262394	0.239468	
min	1.000000	0.000000	0.000000	0.000000	0.000000	
25%	2115.000000	0.000000	0.000000	0.000000	0.000000	
50%	4202.000000	0.000000	0.000000	0.000000	0.000000	
75%	6310.000000	0.000000	0.000000	0.000000	0.000000	
max	8416.000000	1.000000	1.000000	1.000000	1.000000	

	X14	X15	X16	X17	X18	…	\
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000	…	
mean	0.427893	0.000713	0.002613	0.008791	0.010216	…	
std	0.494832	0.026691	0.051061	0.093357	0.100570	…	
min	0.000000	0.000000	0.000000	0.000000	0.000000	…	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	…	
50%	0.000000	0.000000	0.000000	0.000000	0.000000	…	
75%	1.000000	0.000000	0.000000	0.000000	0.000000	…	
max	1.000000	1.000000	1.000000	1.000000	1.000000	…	

	X375	X376	X377	X378	X379	\
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000	
mean	0.325968	0.049656	0.311951	0.019244	0.011879	
std	0.468791	0.217258	0.463345	0.137399	0.108356	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	0.000000	0.000000	
75%	1.000000	0.000000	1.000000	0.000000	0.000000	
max	1.000000	1.000000	1.000000	1.000000	1.000000	

	X380	X382	X383	X384	X385
count	4209.000000	4209.000000	4209.000000	4209.000000	4209.000000
mean	0.008078	0.008791	0.000475	0.000713	0.001663
std	0.089524	0.093357	0.021796	0.026691	0.040752
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000

[8 rows x 369 columns]

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

[20]:
```
df_train.isnull().sum().values
```

[20]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

[21]:
```
df_train.isnull().any()
```

[21]:
```
y        False
X0       False
X1       False
X2       False
X3       False
         …
X380     False
X382     False
X383     False
X384     False
X385     False
Length: 365, dtype: bool
```

[22]:
```
df_test.isnull().sum().values
```

[22]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0])
```

[132]:
```
df_train.nunique()
```

[132]:
```
y        2545
X0         47
X1         27
X2         44
X3          7
         …
X380        2
X382        2
X383        2
X384        2
X385        2
Length: 365, dtype: int64
```

6 Filter out the columns having object datatype

[133]:
```
object_datatypes = df_train.select_dtypes(include=[object])
object_datatypes
```

[133]:
```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79,
80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99,
…]

[4209 rows x 0 columns]
```

[134]:
```
object_datatype_columns = object_datatypes.columns
object_datatype_columns
```

[134]:
```
Index([], dtype='object')
```

7 Apply label encoder

[24]:
```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
df_train['X0'].unique()
```

[24]:
```
array(['k', 'az', 't', 'al', 'o', 'w', 'j', 'h', 's', 'n', 'ay', 'f', 'x',
       'y', 'aj', 'ak', 'am', 'z', 'q', 'at', 'ap', 'v', 'af', 'a', 'e',
       'ai', 'd', 'aq', 'c', 'aa', 'ba', 'as', 'i', 'r', 'b', 'ax', 'bc',
       'u', 'ad', 'au', 'm', 'l', 'aw', 'ao', 'ac', 'g', 'ab'],
      dtype=object)
```

[25]:
```
df_train['X0'] = le.fit_transform(df_train['X0'])
df_train['X0'].unique()
```

[25]:
```
array([32, 20, 40,  9, 36, 43, 31, 29, 39, 35, 19, 27, 44, 45,  7,  8, 10,
       46, 37, 15, 12, 42,  5,  0, 26,  6, 25, 13, 24,  1, 22, 14, 30, 38,
       21, 18, 23, 41,  4, 16, 34, 33, 17, 11,  3, 28,  2])
```

[26]:
```
df_train['X1'] = le.fit_transform(df_train['X1'])
df_train['X2'] = le.fit_transform(df_train['X2'])
df_train['X3'] = le.fit_transform(df_train['X3'])
df_train['X4'] = le.fit_transform(df_train['X4'])
df_train['X5'] = le.fit_transform(df_train['X5'])
df_train['X6'] = le.fit_transform(df_train['X6'])
df_train['X8'] = le.fit_transform(df_train['X8'])
```

[27]:
```
df_train.head()
```

[27]:

	y	X0	X1	X2	X3	X4	X5	X6	X8	X10	…	X375	X376	X377	X378	\
0	130.81	32	23	17	0	3	24	9	14	0	…	0	0	1	0	
1	88.53	32	21	19	4	3	28	11	14	0	…	1	0	0	0	
2	76.26	20	24	34	2	3	27	9	23	0	…	0	0	0	0	
3	80.62	20	21	34	5	3	27	11	4	0	…	0	0	0	0	
4	78.02	20	23	34	5	3	12	3	13	0	…	0	0	0	0	

	X379	X380	X382	X383	X384	X385
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0

```
[5 rows x 365 columns]
```

8 Perform dimensionality reduction

[28]:
```
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
```

[29]:
```
# PCA with 95%
sklearn_pca = PCA(n_components=0.95)
sklearn_pca.fit(df_train)
```

[29]:
```
PCA(n_components=0.95)
```

[30]:
```
sklearn_pca.fit(df_train)
```

[30]:
```
PCA(n_components=0.95)
```

[31]:
```
x_train_transformed = sklearn_pca.transform(df_train)
print(x_train_transformed.shape)
```

```
(4209, 6)
```

[32]:
```
# PCA with 98%
sklearn_pca_98 = PCA(n_components=0.98)
sklearn_pca_98.fit(df_train)
```

[32]:
```
PCA(n_components=0.98)
```

[33]:
```
x_train_transformed_98 = sklearn_pca_98.transform(df_train)
print(x_train_transformed_98.shape)
```

```
(4209, 12)
```

[34]:
```
df_train.y
```

[34]:
```
0       130.81
1        88.53
2        76.26
3        80.62
4        78.02
         ...
4204    107.39
4205    108.77
4206    109.22
4207     87.48
```

```
4208    110.85
Name: y, Length: 4209, dtype: float64
```

9 Train and Test split on Train dataset

[35]:
```
X = df_train.drop('y', axis=1)
y = df_train.y
xtrain,xtest,ytrain,ytest = train_test_split(X,y,test_size=0.3,random_state=42)
```

[36]:
```
print(xtrain)
print(xtrain.shape)
```

```
      X0  X1  X2  X3  X4  X5  X6  X8  X10  X12  …  X375  X376  X377  X378  \
370   35  13  16   1   3   9   6  19    0    0  …     0     0     0     0
3392  15  10  16   2   3  23   9  16    0    0  …     0     0     1     0
2208  31   3  16   2   3  15   2  21    0    0  …     0     0     1     0
3942  35  20   8   6   3  26   6  14    0    1  …     1     0     0     0
1105  36  13  16   5   3   1   6   0    0    0  …     0     0     0     0
…   ..  ..  ..  ..  ..  ..  ..  ..  …  …  …   …   …   …   …
3444  31  10  16   2   3  22  11  17    0    0  …     0     0     1     0
466   20  25  25   2   3   9   9   9    0    0  …     0     0     0     0
3092  45  24   3   2   3  21   8   2    0    0  …     1     0     0     0
3772  45  19   8   5   3  25   8   1    0    1  …     0     0     0     0
860   22   1   7   2   3   5   9  17    0    0  …     1     0     0     0

      X379  X380  X382  X383  X384  X385
370      0     0     0     0     0     0
3392     0     0     0     0     0     0
2208     0     0     0     0     0     0
3942     0     0     0     0     0     0
1105     0     0     0     0     0     0
…    …   …   …   …   …   …
3444     0     0     0     0     0     0
466      0     0     1     0     0     0
3092     0     0     0     0     0     0
3772     0     0     0     0     0     0
860      0     0     0     0     0     0

[2946 rows x 364 columns]
(2946, 364)
```

[37]:
```
print(ytrain)
print(ytrain.shape)
```

```
370      95.13
```

```
3392    117.36
2208    109.01
3942     93.77
1105    103.41
         …
3444    109.42
466      78.25
3092     92.18
3772     91.92
860      87.71
Name: y, Length: 2946, dtype: float64
(2946,)
```

[38]:
```
print(xtest)
print(xtest.shape)
```

```
      X0  X1  X2  X3  X4  X5  X6  X8  X10  X12  …  X375  X376  X377  X378  \
1073   9  16   7   5   3   6   9  11    0    0  …     0     0     0     0
144   27  13   3   5   3  13   8  22    0    0  …     0     0     0     0
2380  31   1  21   2   3  18  11  14    1    0  …     1     0     0     0
184   20  25  22   2   3  13   9  11    0    0  …     0     0     0     0
2587   8  23   8   3   3  17   8  17    0    0  …     0     0     0     0
…    ..  ..  ..  ..  ..  ..  ..  ..  …  …  …   …   …   …   …
2493  27  20  16   2   3  18  10   5    0    0  …     0     0     1     0
3388  40  19  24   5   3  23   3  19    0    0  …     0     0     0     0
3997  22   3   7   0   3  26   6  18    0    0  …     0     0     1     0
383   40   1  16   6   3   9   8   0    0    0  …     1     0     0     0
3364  27   4  33   2   3  23   6  24    0    0  …     0     0     1     0

      X379  X380  X382  X383  X384  X385
1073     0     0     0     0     0     0
144      0     0     0     0     0     0
2380     0     0     0     0     0     0
184      0     0     1     0     0     0
2587     0     0     0     0     0     0
…    …   …   …   …   …   …
2493     0     0     0     0     0     0
3388     0     0     0     0     0     0
3997     0     0     0     0     0     0
383      0     0     0     0     0     0
3364     0     0     0     0     0     0

[1263 rows x 364 columns]
(1263, 364)
```

[104]:
```
# PCA with 95% for xtrain
```

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

[104]:
```
PCA(n_components=0.95)
```

[105]:
```
pca_xtrain_transformed = pca_xtrain.transform(xtrain)
print(pca_xtrain_transformed.shape)
```

```
(2946, 6)
```

[106]:
```
# PCA with 95% for xtest

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

[106]:
```
PCA(n_components=0.95)
```

[107]:
```
pca_xtest_transformed = pca_xtest.transform(xtest)
print(pca_xtest_transformed.shape)
```

```
(1263, 6)
```

[108]:
```
print(pca_xtest.explained_variance_)
print(pca_xtest.explained_variance_ratio_)
```

```
[206.79524961 120.24273955  67.64680756  61.94375666  48.08214872
   8.7271811 ]
[0.38517942 0.22396563 0.12599979 0.11537722 0.08955841 0.01625536]
```

10 Predict your test_df values using XGBoost

[109]:
```
df_test
```

[109]:

	ID	X0	X1	X2	X3	X4	X5	X6	X8	X10	…	X375	X376	X377	X378	\
0	1	21	23	34	5	3	26	0	22	0	…	0	0	0	1	
1	2	42	3	8	0	3	9	6	24	0	…	0	0	1	0	
2	3	21	23	17	5	3	0	9	9	0	…	0	0	0	1	
3	4	21	13	34	5	3	31	11	13	0	…	0	0	0	1	
4	5	45	20	17	2	3	30	8	12	0	…	1	0	0	0	
…	…	..	..	..	..	..	..	..	..	…	…	…	…	…	…	
4204	8410	6	9	17	5	3	1	9	4	0	…	0	0	0	0	
4205	8411	42	1	8	3	3	1	9	24	0	…	0	1	0	0	
4206	8413	47	23	17	5	3	1	3	22	0	…	0	0	0	0	
4207	8414	7	23	17	0	3	1	2	16	0	…	0	0	1	0	
4208	8416	42	1	8	2	3	1	6	17	0	…	1	0	0	0	

```
      X379  X380  X382  X383  X384  X385
0        0     0     0     0     0     0
1        0     0     0     0     0     0
2        0     0     0     0     0     0
3        0     0     0     0     0     0
4        0     0     0     0     0     0
…    …   …   …   …   …   …
4204     0     0     0     0     0     0
4205     0     0     0     0     0     0
4206     0     0     0     0     0     0
4207     0     0     0     0     0     0
4208     0     0     0     0     0     0

[4209 rows x 377 columns]
```

[110]:
```
test_object_datatypes = df_test.select_dtypes(include=[object])
test_object_datatypes
```

[110]:
```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79,
80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99,
…]

[4209 rows x 0 columns]
```

[111]:
```
df_test['X0'] = le.fit_transform(df_test['X0'])
df_test['X1'] = le.fit_transform(df_test['X1'])
df_test['X2'] = le.fit_transform(df_test['X2'])
df_test['X3'] = le.fit_transform(df_test['X3'])
df_test['X4'] = le.fit_transform(df_test['X4'])
df_test['X5'] = le.fit_transform(df_test['X5'])
df_test['X6'] = le.fit_transform(df_test['X6'])
df_test['X8'] = le.fit_transform(df_test['X8'])
```

[112]:
```
print(df_test)
print(df_test.shape)
```

```
      ID  X0  X1  X2  X3  X4  X5  X6  X8  X10  …  X375  X376  X377  X378  \
0      1  21  23  34   5   3  26   0  22    0  …     0     0     0     1
1      2  42   3   8   0   3   9   6  24    0  …     0     0     1     0
2      3  21  23  17   5   3   0   9   9    0  …     0     0     0     1
3      4  21  13  34   5   3  31  11  13    0  …     0     0     0     1
4      5  45  20  17   2   3  30   8  12    0  …     1     0     0     0
```

```
…    …  ..  ..  ..  ..  ..  ..  ..  ..  …  …   …   …   …   …
4204  8410   6   9  17   5   3   1   9   4    0  …     0     0     0     0
4205  8411  42   1   8   3   3   1   9  24    0  …     0     1     0     0
4206  8413  47  23  17   5   3   1   3  22    0  …     0     0     0     0
4207  8414   7  23  17   0   3   1   2  16    0  …     0     0     1     0
4208  8416  42   1   8   2   3   1   6  17    0  …     1     0     0     0

      X379  X380  X382  X383  X384  X385
0        0     0     0     0     0     0
1        0     0     0     0     0     0
2        0     0     0     0     0     0
3        0     0     0     0     0     0
4        0     0     0     0     0     0
…    …   …   …   …   …   …
4204     0     0     0     0     0     0
4205     0     0     0     0     0     0
4206     0     0     0     0     0     0
4207     0     0     0     0     0     0
4208     0     0     0     0     0     0

[4209 rows x 377 columns]
(4209, 377)
```

[113]:
```
pca_df_test = PCA(n_components=0.95)
pca_df_test.fit(df_test)
```

[113]:
```
PCA(n_components=0.95)
```

[114]:
```
pca_df_test_transformed = pca_df_test.transform(df_test)
print(pca_df_test_transformed.shape)
```

```
(4209, 1)
```

[115]:
```
print(pca_df_test.explained_variance_)
print(pca_df_test.explained_variance_ratio_)
```

```
[5871343.86260106]
[0.99990882]
```

[116]:
```
y
```

[116]:
```
0       130.81
1        88.53
2        76.26
3        80.62
4        78.02
         …
4204    107.39
```

```
4205    108.77
4206    109.22
4207     87.48
4208    110.85
Name: y, Length: 4209, dtype: float64
```

[135]:
```
from sklearn import svm
from sklearn import model_selection
import xgboost as xgb
```

[162]:
```
model = xgb.XGBRegressor
model.fit=(pca_xtrain, ytrain)
print(model)
```

```
<class 'xgboost.sklearn.XGBRegressor'>
```