In [70]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
```

In [71]:
```python
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,r2_score,classification_report,confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest,SelectPercentile,mutual_info_classif,mutual_info_regression
```

In [72]:
```python
data = pd.read_csv('CarPrice_Assignment.csv')
```

In [73]:
```python
data.head()
```

Out[73]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsy
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	130	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	130	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	152	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	109	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	136	

5 rows × 26 columns

In [74]:
```python
data.tail()
```

Out[74]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsyste
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	front	109.1	...	141	m
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	front	109.1	...	141	m
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	front	109.1	...	173	m
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	front	109.1	...	145	
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	front	109.1	...	141	m

5 rows × 26 columns

In [75]:
```python
data.columns
```

Out[75]:
```
Index(['car_ID', 'symboling', 'CarName', 'fueltype', 'aspiration',
       'doornumber', 'carbody', 'drivewheel', 'enginelocation', 'wheelbase',
       'carlength', 'carwidth', 'carheight', 'curbweight', 'enginetype',
       'cylindernumber', 'enginesize', 'fuelsystem', 'boreratio', 'stroke',
       'compressionratio', 'horsepower', 'peakrpm', 'citympg', 'highwaympg',
       'price'],
      dtype='object')
```

In [76]:
```python
data.isnull().sum(
)
```

Out[76]:
```
car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
price               0
dtype: int64
```

In [77]:
```python
data.nunique()
```

Out[77]:
```
car_ID              205
symboling             6
CarName             147
fueltype              2
aspiration            2
doornumber            2
carbody               5
drivewheel            3
enginelocation        2
wheelbase            53
carlength            75
carwidth             44
carheight            49
curbweight          171
enginetype            7
cylindernumber        7
enginesize           44
fuelsystem            8
boreratio            38
stroke               37
compressionratio     32
horsepower           59
peakrpm              23
citympg              29
highwaympg           30
price               189
dtype: int64
```

In [78]:
```python
data.iloc[:,20:30]
```

Out[78]:

	compressionratio	horsepower	peakrpm	citympg	highwaympg	price
0	9.0	111	5000	21	27	13495.0
1	9.0	111	5000	21	27	16500.0
2	9.0	154	5000	19	26	16500.0
3	10.0	102	5500	24	30	13950.0
4	8.0	115	5500	18	22	17450.0
...	...	...	...	...	...	...
200	9.5	114	5400	23	28	16845.0
201	8.7	160	5300	19	25	19045.0
202	8.8	134	5500	18	23	21485.0
203	23.0	106	4800	26	27	22470.0
204	9.5	114	5400	19	25	22625.0

205 rows × 6 columns

In [79]:
```python
data.drop(columns=['car_ID'],inplace=True)
```

In [80]:
```python
data['CarName'].value_counts().sort_values(ascending=False).head(10)
```

Out[80]:
```
toyota corona           6
toyota corolla          6
peugeot 504             6
subaru dl               4
mitsubishi outlander    3
mazda 626               3
toyota mark ii          3
mitsubishi mirage g4    3
mitsubishi g4           3
honda civic             3
Name: CarName, dtype: int64
```

In [81]:
```python
label = LabelEncoder()
```

In [82]:
```python
data['CarName'] = label.fit_transform(data['CarName'])
```

In [83]:
```python
data['fueltype'].value_counts().sort_values(ascending=False).head(10)
```

Out[83]:
```
gas       185
diesel     20
Name: fueltype, dtype: int64
```

In [84]:
```python
data['fueltype'] = label.fit_transform(data['fueltype'])
```

In [85]:
```python
data['aspiration'].value_counts().sort_values(ascending=False).head(10)
```

Out[85]:
```
std      168
turbo     37
Name: aspiration, dtype: int64
```

In [86]:
```python
data['aspiration'] = label.fit_transform(data['aspiration'])
```

In [87]:
```python
data['doornumber'].value_counts().sort_values(ascending=False).head(10)
```

Out[87]:
```
four    115
two      90
Name: doornumber, dtype: int64
```

In [88]:
```python
data['doornumber'] = label.fit_transform(data['doornumber'])
```

In [89]:
```python
data['drivewheel'].value_counts().sort_values(ascending=False).head(10)
```

Out[89]:
```
fwd    120
rwd     76
4wd      9
Name: drivewheel, dtype: int64
```

In [90]:
```python
df = pd.get_dummies(data['drivewheel'],drop_first=True)
```

In [91]:
```python
data['enginelocation'].value_counts().sort_values(ascending=False).head(10)
```

Out[91]:
```
front    202
rear       3
Name: enginelocation, dtype: int64
```

In [92]:
```python
data['enginelocation'] = label.fit_transform(data['enginelocation'])
```

In [93]:
```python
data['enginetype'].value_counts().sort_values(ascending=False).head(10)
```

Out[93]:
```
ohc      148
ohcf      15
ohcv      13
l         12
dohc      12
rotor      4
dohcv      1
Name: enginetype, dtype: int64
```

In [94]:
```python
df2 = pd.get_dummies(data['enginetype'],drop_first=True)
```

In [95]:
```python
data['cylindernumber'].value_counts().sort_values(ascending=False).head(10)
```

Out[95]:
```
four      159
six        24
five       11
eight       5
two         4
three       1
twelve      1
Name: cylindernumber, dtype: int64
```

In [96]:
```python
data['cylindernumber'] = label.fit_transform(data['cylindernumber'])
```

In [97]:
```python
data['fuelsystem'].value_counts().sort_values(ascending=False).head(10)
```

Out[97]:
```
mpfi    94
2bbl    66
idi     20
1bbl    11
spdi     9
4bbl     3
spfi     1
mfi      1
Name: fuelsystem, dtype: int64
```

In [98]:
```python
data['fuelsystem'] = label.fit_transform(data['fuelsystem'])
```

In [99]:
```python
data1 = pd.concat([data,df],axis=1)
```

In [100…
```python
data1.drop(columns=['drivewheel'],inplace=True)
```

In [103…
```python
data1.drop(columns=['enginetype'],inplace=True)
```

In [105…
```python
data1['carbody'] = label.fit_transform(data1['carbody'])
```

In [106…
```python
len(data1.columns)
```

Out[106… 25

In [107…
```python
data1.head()
```

Out[107…

	symboling	CarName	fueltype	aspiration	doornumber	carbody	enginelocation	wheelbase	carlength	carwidth	...	boreratio	stroke	co
0	3	2	1	0	1	0	0	88.6	168.8	64.1	...	3.47	2.68	
1	3	3	1	0	1	0	0	88.6	168.8	64.1	...	3.47	2.68	
2	1	1	1	0	1	2	0	94.5	171.2	65.5	...	2.68	3.47	
3	2	4	1	0	0	3	0	99.8	176.6	66.2	...	3.19	3.40	
4	2	5	1	0	0	3	0	99.4	176.6	66.4	...	3.19	3.40	

5 rows × 25 columns

In [108…
```python
data1.tail()
```

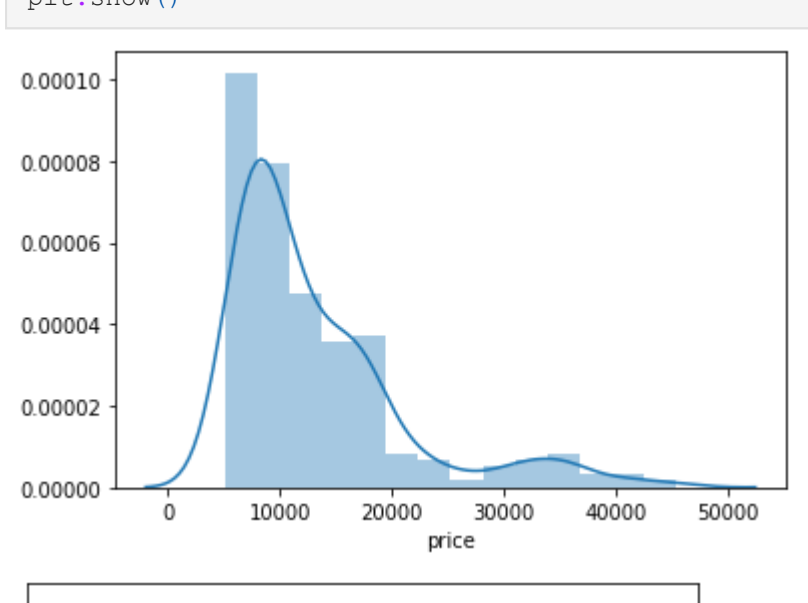
Out[108…

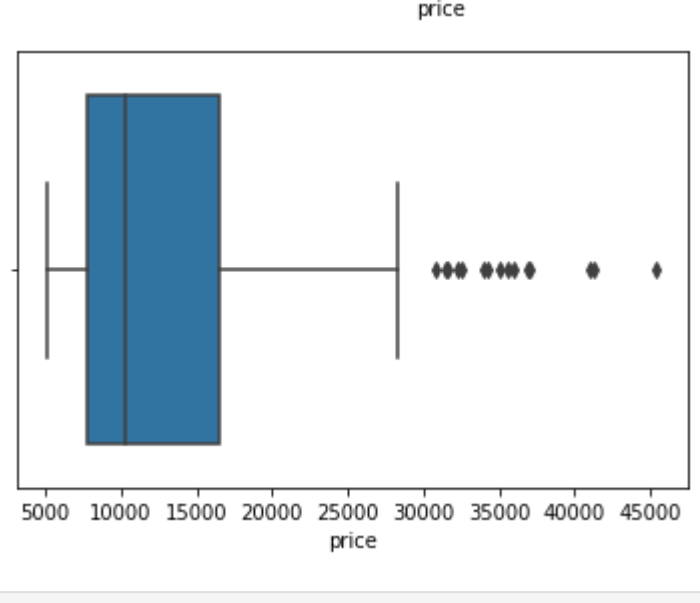
	symboling	CarName	fueltype	aspiration	doornumber	carbody	enginelocation	wheelbase	carlength	carwidth	...	boreratio	stroke
200	-1	139	1	0	0	3	0	109.1	188.8	68.9	...	3.78	3.15
201	-1	138	1	1	0	3	0	109.1	188.8	68.8	...	3.78	3.15
202	-1	140	1	0	0	3	0	109.1	188.8	68.9	...	3.58	2.87
203	-1	142	0	1	0	3	0	109.1	188.8	68.9	...	3.01	3.40
204	-1	143	1	1	0	3	0	109.1	188.8	68.9	...	3.78	3.15

5 rows × 25 columns

Feature selction on the basis of Mutual Information Gain

In [109…
```python
sns.distplot(data1['price'])
plt.show()
sns.boxplot(data1['price'])
plt.show()
```





In [121…
```python
Q1 = data1['price'].quantile(0.25)
Q3 = data1['price'].quantile(0.75)

IQR = Q3-Q1
IQR
```

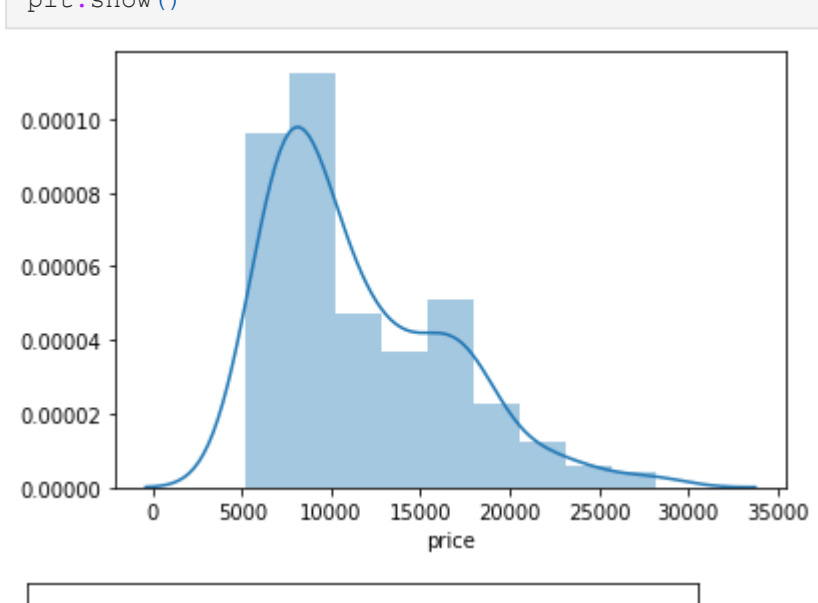
Out[121… 8715.0

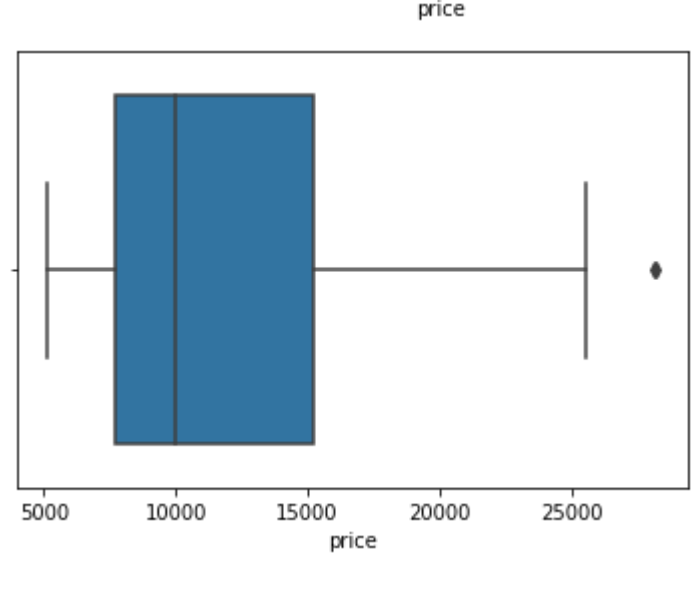
In [122…
```python
dataiqr = data1[~((data1['price']<Q1-1.5*IQR)|(data1['price']>Q3+1.5*IQR))]
```

In [123…
```python
dataiqr.shape
```

Out[123… (190, 25)

In [124…
```python
sns.distplot(dataiqr['price'])
plt.show()
sns.boxplot(dataiqr['price'])
plt.show()
```





Implement the IG

In [125…
```python
x = dataiqr.drop(columns=['price'])
y = dataiqr['price']
```

In [126…
```python
x.shape,y.shape
```

Out[126… ((190, 24), (190,))

In [127…
```python
x_train,x_test,y_train,y_test = train_test_split(x,y,train_size=0.80,random_state=42)
```

In [128…
```python
x_train.shape,x_test.shape
```

Out[128… ((152, 24), (38, 24))

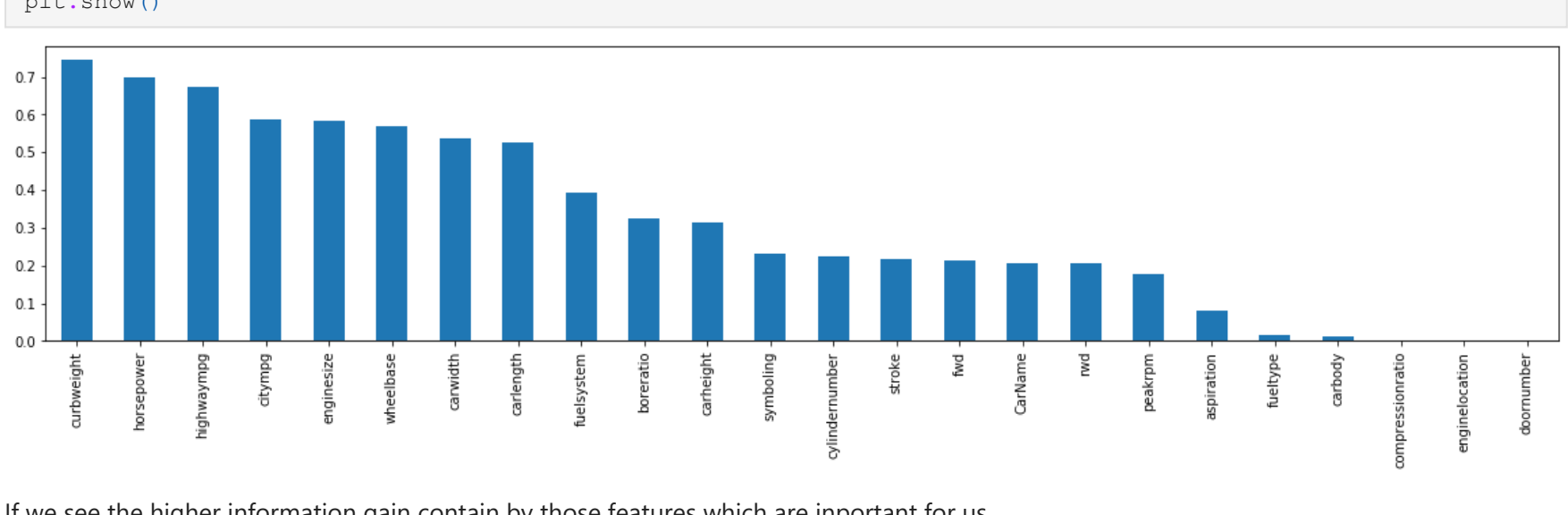
In [129…
```python
mi = mutual_info_regression(x_train,y_train,n_neighbors=3)
mi
```

Out[129…
```
array([0.23099133, 0.20735529, 0.01525175, 0.08124114, 0.        ,
       0.01140548, 0.        , 0.57003122, 0.52634203, 0.53831904,
       0.31366507, 0.74377656, 0.22382044, 0.58440239, 0.39176277,
       0.32568874, 0.21814842, 0.        , 0.69713576, 0.17722742,
       0.58536724, 0.6733307 , 0.21442722, 0.205485  ])
```

In [130…
```python
mi = pd.Series(mi,index=x_train.columns)
```

In [131…
```python
mi.sort_values(ascending=False,inplace=True)
```

In [134…
```python
plt.figure(figsize=(20,4))
mi.plot(kind='bar')
plt.show()
```



If we see the higher information gain contain by those features which are inportant for us.

In [178…
```python
sel = SelectPercentile(mutual_info_regression,percentile=70)
```

In [179…
```python
sel.fit(x_train,y_train)
```

Out[179…
```
SelectPercentile(percentile=70,
                 score_func=<function mutual_info_regression at 0x0DABC030>)
```

In [180…
```python
features = x_train.columns[sel.get_support()]
```

In [181…
```python
features
```

Out[181…
```
Index(['symboling', 'CarName', 'wheelbase', 'carlength', 'carwidth',
       'carheight', 'curbweight', 'cylindernumber', 'enginesize', 'fuelsystem',
       'boreratio', 'stroke', 'horsepower', 'citympg', 'highwaympg', 'fwd',
       'rwd'],
      dtype='object')
```

In [182…
```python
x_train_mi = sel.transform(x_train)
x_test_mi  = sel.transform(x_test)
```

In [183…
```python
x_train_mi.shape,x_test_mi.shape
```

Out[183… ((152, 17), (38, 17))

In [184…
```python
from sklearn.linear_model import LinearRegression
```

In [185…
```python
def Linearmodel(x_train,x_test,y_train,y_test):
    model = LinearRegression()
    model.fit(x_train_mi,y_train)
    y_pred = model.predict(x_test_mi)
    print('Accuracy :',r2_score(y_test,y_pred))
```

In [186…
```python
%%time
Linearmodel(x_train_mi,x_test_mi,y_train,y_test)
```

```
Accuracy : 0.811258225578604
Wall time: 0 ns
```

Mutual Infomation Classification Problem.

In [187…
```python
data = pd.read_csv('dataset01_eurusd4h.csv')
```

In [188…
```python
data.head()
```

Out[188…

	rsi1	rsi2	rsi3	rsi4	rsi5	rsi6	stoch1	stoch2	stoch3	stoch4	...	WPR6	close1	close2	close3	close4	close5	close6	hour	d
0	28.90	31.07	40.01	40.51	39.95	41.98	13.53	29.27	46.80	43.52	...	-56.09	1.3132	1.3132	1.3132	1.3132	1.3132	1.3132	0	
1	27.39	28.90	31.07	40.01	40.51	39.95	3.93	13.53	29.27	46.80	...	-65.93	1.3125	1.3125	1.3125	1.3125	1.3125	1.3125	4	
2	28.41	27.39	28.90	31.07	40.01	40.51	4.27	3.93	13.53	29.27	...	-68.07	1.3127	1.3127	1.3127	1.3127	1.3127	1.3127	8	
3	34.48	28.41	27.39	28.90	31.07	40.01	12.99	4.27	3.93	13.53	...	-74.34	1.3138	1.3138	1.3138	1.3138	1.3138	1.3138	12	
4	33.35	34.48	28.41	27.39	28.90	31.07	24.48	12.99	4.27	3.93	...	-92.30	1.3134	1.3134	1.3134	1.3134	1.3134	1.3134	16	

5 rows × 99 columns

In [189…
```python
data.tail()
```

Out[189…

	rsi1	rsi2	rsi3	rsi4	rsi5	rsi6	stoch1	stoch2	stoch3	stoch4	...	WPR6	close1	close2	close3	close4	close5	close6	hour
4474	62.40	61.40	69.84	77.07	81.66	76.33	26.52	38.85	60.16	74.48	...	-1.88	1.1974	1.1974	1.1974	1.1974	1.1974	1.1974	4
4475	59.95	62.40	61.40	69.84	77.07	81.66	19.21	26.52	38.85	60.16	...	-10.44	1.1964	1.1964	1.1964	1.1964	1.1964	1.1964	8
4476	56.54	59.95	62.40	61.40	69.84	77.07	17.90	19.21	26.52	38.85	...	-15.83	1.1950	1.1950	1.1950	1.1950	1.1950	1.1950	12
4477	48.39	56.54	59.95	62.40	61.40	69.84	15.75	17.90	19.21	26.52	...	-24.95	1.1912	1.1912	1.1912	1.1912	1.1912	1.1912	16
4478	45.79	48.39	56.54	59.95	62.40	61.40	12.88	15.75	17.90	19.21	...	-49.51	1.1898	1.1898	1.1898	1.1898	1.1898	1.1898	20

5 rows × 99 columns

In [190…
```python
data.columns
```

Out[190…
```
Index(['rsi1', 'rsi2', 'rsi3', 'rsi4', 'rsi5', 'rsi6', 'stoch1', 'stoch2',
       'stoch3', 'stoch4', 'stoch5', 'stoch6', 'ema20Slope1', 'ema20Slope2',
       'ema20Slope3', 'ema20Slope4', 'ema20Slope5', 'ema20Slope6',
       'ema50Slope1', 'ema50Slope2', 'ema50Slope3', 'ema50Slope4',
       'ema50Slope5', 'ema50Slope6', 'ema100Slope1', 'ema100Slope2',
       'ema100Slope3', 'ema100Slope4', 'ema100Slope5', 'ema100Slope6',
       'ema200Slope1', 'ema200Slope2', 'ema200Slope3', 'ema200Slope4',
       'ema200Slope5', 'ema200Slope6', 'std1', 'std2', 'std3', 'std4', 'std5',
       'std6', 'mom1', 'mom2', 'mom3', 'mom4', 'mom5', 'mom6', 'BB_up_percen1',
       'BB_up_percen2', 'BB_up_percen3', 'BB_up_percen4', 'BB_up_percen5',
       'BB_up_percen6', 'cci1', 'cci2', 'cci3', 'cci4', 'cci5', 'cci6',
       'force1', 'force2', 'force3', 'force4', 'force5', 'force6', 'macd1',
       'macd2', 'macd3', 'macd4', 'macd5', 'macd6', 'bearsPower1',
       'bearsPower2', 'bearsPower3', 'bearsPower4', 'bearsPower5',
       'bearsPower6', 'bullsPower1', 'bullsPower2', 'bullsPower3',
       'bullsPower4', 'bullsPower5', 'bullsPower6', 'WPR1', 'WPR2', 'WPR3',
       'WPR4', 'WPR5', 'WPR6', 'close1', 'close2', 'close3', 'close4',
       'close5', 'close6', 'hour', 'dayOfWeek', 'tipo'],
      dtype='object')
```

In [191…
```python
len(data.columns)
```

Out[191… 99

In [195…
```python
data.dtypes.head(20)
```

Out[195…
```
rsi1           float64
rsi2           float64
rsi3           float64
rsi4           float64
rsi5           float64
rsi6           float64
stoch1         float64
stoch2         float64
stoch3         float64
stoch4         float64
stoch5         float64
stoch6         float64
ema20Slope1    float64
ema20Slope2    float64
ema20Slope3    float64
ema20Slope4    float64
ema20Slope5    float64
ema20Slope6    float64
ema50Slope1    float64
ema50Slope2    float64
dtype: object
```

In [198…
```python
data.isnull().sum().sort_values(ascending=False).head(10)
```

Out[198…
```
tipo            0
ema100Slope1    0
ema100Slope3    0
ema100Slope4    0
ema100Slope5    0
ema100Slope6    0
ema200Slope1    0
ema200Slope2    0
ema200Slope3    0
ema200Slope4    0
dtype: int64
```

In [200…
```python
data.nunique()
```

Out[200…
```
rsi1         2918
rsi2         2928
rsi3         2929
rsi4         2930
rsi5         2926
             ...
close5       1536
close6       1536
hour           10
dayOfWeek       6
tipo            2
Length: 99, dtype: int64
```

In [201…
```python
x = data.drop(columns='tipo')
y = data['tipo']
```

In [202…
```python
x.shape,y.shape
```

Out[202… ((4479, 98), (4479,))

In [203…
```python
x_train,x_test,y_train,y_test = train_test_split(x,y,train_size=0.80,stratify=y,random_state=42)
```

In [204…
```python
x_train.shape,x_test.shape
```

Out[204… ((3583, 98), (896, 98))

In [205…
```python
from sklearn.feature_selection import VarianceThreshold
```

In [206…
```python
var = VarianceThreshold(threshold=0.01)
x_train_unique = var.fit_transform(x_train)
x_test_unique  = var.transform(x_test)
```

In [214…
```python
x_train_unique = pd.DataFrame(x_train_unique)
x_test_unique   = pd.DataFrame(x_test_unique)
```

In [208…
```python
x_train_T = x_train_unique.T
```

In [209…
```python
x_train_T = pd.DataFrame(x_train_T)
```

In [210…
```python
x_train_T.duplicated().sum()
```

Out[210… 0

In [237…
```python
y_train.value_counts()
```

Out[237…
```
0    1817
1    1766
Name: tipo, dtype: int64
```

In [248…
```python
scaler = StandardScaler()
x_train_unique = scaler.fit_transform(x_train_unique)
x_test_unique  = scaler.transform(x_test_unique)
```

In [249…
```python
x_train_unique = pd.DataFrame(x_train_unique)
x_test_unique = pd.DataFrame(x_test_unique)
```

There is no need find the duplicated.

In [250…
```python
mi =mutual_info_classif(x_train_unique,y_train,n_neighbors=3,random_state=42)
```

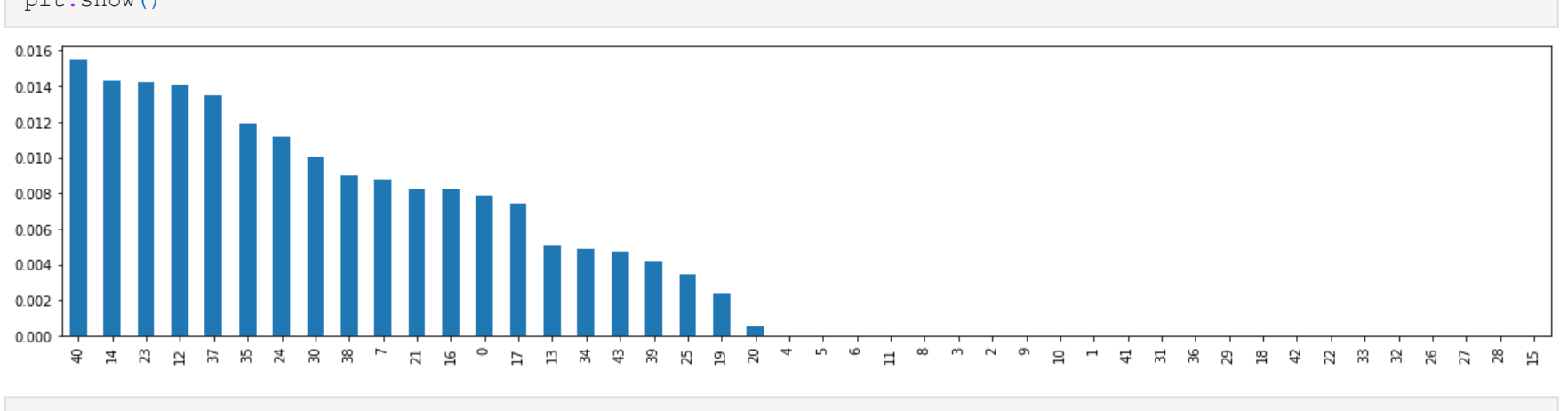
In [251…
```python
mi
```

Out[251…
```
array([0.00787631, 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.00874687, 0.        , 0.        ,
       0.        , 0.        , 0.01411222, 0.00506768, 0.01428357,
       0.        , 0.00821254, 0.00745152, 0.        , 0.00243273,
       0.00055892, 0.00826799, 0.        , 0.01421783, 0.01113077,
       0.00346849, 0.        , 0.        , 0.        , 0.        ,
       0.01006846, 0.        , 0.        , 0.        , 0.00484005,
       0.01190763, 0.        , 0.01351362, 0.00897282, 0.00416849,
       0.01550984, 0.        , 0.        , 0.0047066 ])
```

In [252…
```python
mi = pd.Series(mi,index=x_train_unique.columns)
```

In [253…
```python
mi.sort_values(ascending=False,inplace=True)
```

In [254…
```python
plt.figure(figsize=(20,4))
mi.plot(kind='bar')
plt.show()
```



In [265…
```python
sel = SelectKBest(mutual_info_classif,k=30)
```

In [266…
```python
sel.fit(x_train_unique,y_train)
```

Out[266…
```
SelectKBest(k=30, score_func=<function mutual_info_classif at 0x0DABC078>)
```

In [267…
```python
features = x_train_unique.columns[sel.get_support()]
```

In [268…
```python
mi
```

Out[268…
```
40    0.015510
14    0.014284
23    0.014218
12    0.014112
37    0.013514
35    0.011908
24    0.011131
30    0.010068
38    0.008973
7     0.008747
21    0.008268
16    0.008213
0     0.007876
17    0.007452
13    0.005068
34    0.004840
43    0.004707
39    0.004168
25    0.003468
19    0.002433
20    0.000559
4     0.000000
5     0.000000
6     0.000000
11    0.000000
8     0.000000
3     0.000000
2     0.000000
9     0.000000
10    0.000000
1     0.000000
41    0.000000
31    0.000000
36    0.000000
29    0.000000
18    0.000000
42    0.000000
22    0.000000
33    0.000000
32    0.000000
26    0.000000
27    0.000000
28    0.000000
15    0.000000
dtype: float64
```

In [269…
```python
features
```

Out[269…
```
Int64Index([ 0,  6,  7, 12, 13, 14, 15, 19, 20, 21, 23, 24, 25, 27, 28, 29, 30,
            31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43],
           dtype='int64')
```

In [270…
```python
x_train_sel = sel.transform(x_train_unique)
x_test_sel  = sel.transform(x_test_unique)
```

In [271…
```python
x_train_sel.shape,x_test_sel.shape
```

Out[271… ((3583, 30), (896, 30))

In [281…
```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
```

In [279…
```python
def SVclassifier(x_train,x_test,y_train,y_test):
    clf = SVC(C=0.01,kernel='sigmoid',degree=1)
    clf.fit(x_train,y_train)
    y_pred = clf.predict(x_test)
    print('Accuracy :-',accuracy_score(y_test,y_pred))
```

In [280…
```python
%%time
SVclassifier(x_train_sel,x_test_sel,y_train,y_test)
```

```
Accuracy :- 0.5066964285714286
Wall time: 4.26 s
```

In [282…
```python
def Logistics(x_train,x_test,y_train,y_test):
    clf = LogisticRegression(random_state=42,max_iter=200,solver='lbfgs',verbose=1)
    clf.fit(x_train,y_train)
    y_pred = clf.predict(x_test)
    print('Accuracy :-',accuracy_score(y_test,y_pred))
```

In [400…
```python
%%time
Logistics(x_train_sel,x_test_sel,y_train,y_test)
```

```
Accuracy :- 0.5022321428571429
Wall time: 92 ms
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s finished
```

In [ ]: