MARUTI SUZUKI

Importing Required Libraries

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
sns.set(context="notebook", palette="Spectral", style = 'darkgrid' ,font_scale = 1.5, c
from sklearn import preprocessing
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
```

Loading Data

In [2]:
```
data = pd.read_csv("C:/Users/Galaxy star Gopi/Downloads/MARUTI.NS.csv")
```

In [3]:
```
data.head()
```

Out[3]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-01	7377.000000	7409.950195	7282.049805	7311.700195	7153.508789	634725
1	2020-01-02	7327.600098	7368.000000	7312.000000	7329.850098	7171.266113	616838
2	2020-01-03	7328.950195	7332.000000	7226.000000	7254.250000	7097.301758	571967
3	2020-01-06	7200.000000	7210.000000	7026.500000	7042.399902	6890.035156	748516
4	2020-01-07	7100.000000	7165.600098	7026.399902	7073.600098	6920.560059	590500

In [4]:
```
data.tail()
```

Out[4]:

	Date	Open	High	Low	Close	Adj Close	Volume
725	2022-12-01	9019.049805	9025.000000	8930.000000	8958.150391	8958.150391	694489
726	2022-12-02	8820.000000	8928.900391	8754.750000	8815.849609	8815.849609	971612
727	2022-12-05	8844.000000	8853.200195	8765.000000	8792.049805	8792.049805	519621
728	2022-12-06	8777.200195	8800.000000	8694.950195	8717.349609	8717.349609	551379
729	2022-12-07	8720.000000	8747.849609	8641.000000	8659.150391	8659.150391	546450

In [5]:
```
data.isnull().sum()
```

```
Date         0
```

Out[5]:
```
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

In [6]:
```
data = data.dropna()
```

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 730 entries, 0 to 729
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       730 non-null    object 
 1   Open       730 non-null    float64
 2   High       730 non-null    float64
 3   Low        730 non-null    float64
 4   Close      730 non-null    float64
 5   Adj Close  730 non-null    float64
 6   Volume     730 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 45.6+ KB
```

In [8]:
```
data.describe()
```

Out[8]:

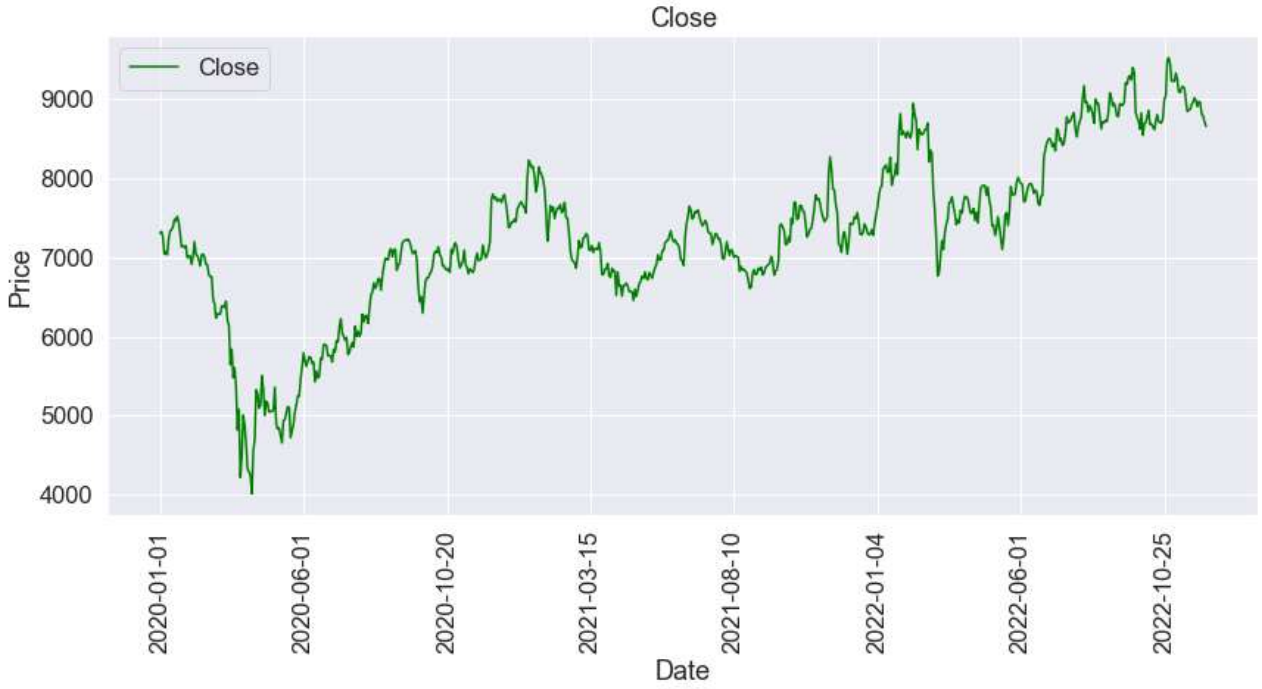
	Open	High	Low	Close	Adj Close	Volume
count	730.000000	730.000000	730.000000	730.000000	730.000000	7.300000e+02
mean	7305.206925	7399.294397	7204.439981	7297.882948	7223.059621	9.564028e+05
std	1042.092313	1035.984537	1048.706347	1039.174036	1067.424183	5.825795e+05
min	4150.000000	4267.000000	4001.100098	4011.500000	3924.709961	3.005200e+04
25%	6852.399902	6919.724976	6775.087402	6832.462524	6757.993286	5.490685e+05
50%	7278.525147	7349.899902	7178.649902	7272.675049	7187.944824	7.989480e+05
75%	7843.000000	7954.625122	7748.750000	7853.862549	7797.151489	1.175805e+06
max	9650.000000	9769.000000	9451.099609	9527.599609	9527.599609	3.846792e+06

Visualize close value from 2020 to 2022

In [9]:
```
plt.figure(figsize=[12, 5]); # Set dimensions for figure
data.plot(x='Date', y='Close', figsize = (14, 6), legend = True, color='g')
plt.title('Close')
plt.ylabel('Price')
plt.xlabel('Date')
plt.xticks(rotation=90)
```

```
plt.grid(True)
plt.show()
```

```
<Figure size 864x360 with 0 Axes>
```

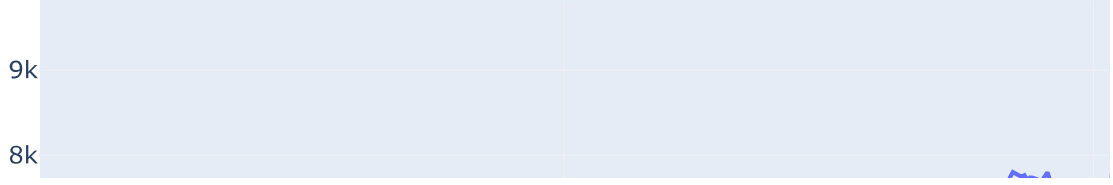


In [10]:

```
import plotly.graph_objs as go
```

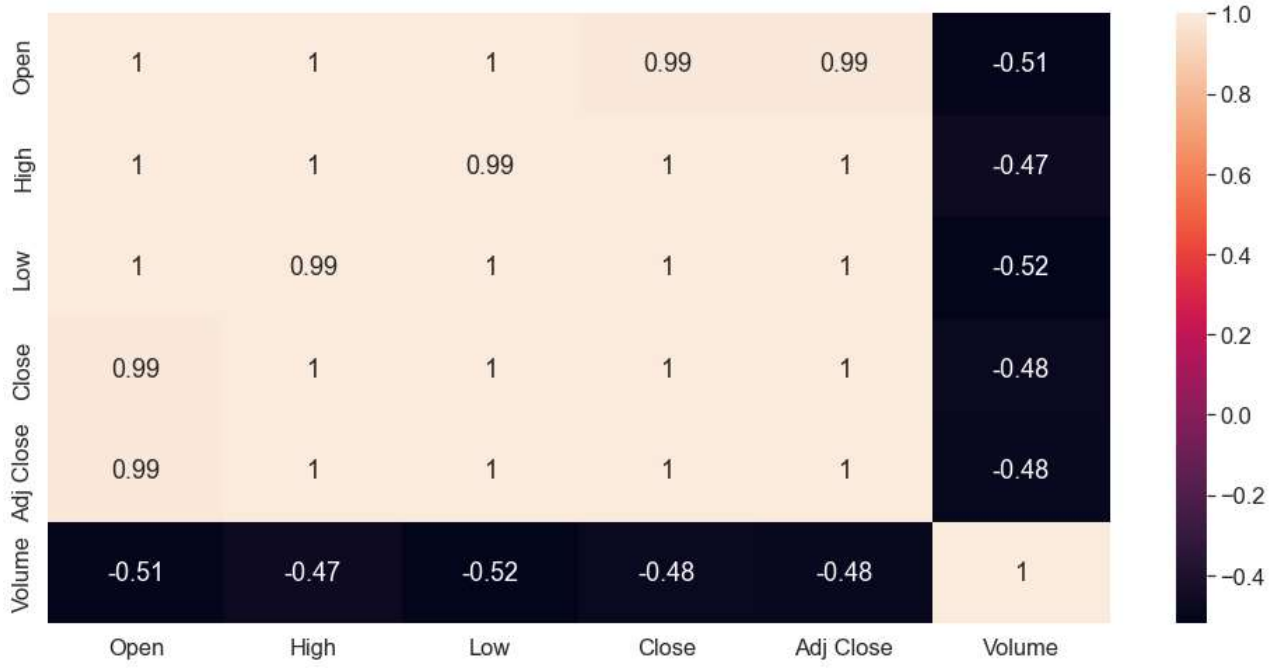
In [11]:

```
#Declare figure
fig = go.Figure()

#add a trace
fig.add_trace(go.Scatter(x=data.Date, y=data['Close']))
#Update X and Y axis with title
fig.update_xaxes(
    title = 'Date',rangeslider_visible=True
)

fig.update_yaxes(
    title = 'Stock price'
)

#Show
fig.show()
```



Now see some correalations between data

In [12]:
```
import seaborn as sns
plt.figure(1 , figsize = (17 , 8))
cor = sns.heatmap(data.corr(), annot = True)
```



Separate the Input and Output Columns

In [13]:
```
X = data[['High','Low','Open','Volume']].values
y = data['Close'].values
```

In [14]:
```
X
```

Out[14]:
```
array([[  7409.950195,   7282.049805,   7377.      , 634725.      ],
       [  7368.      ,   7312.      ,   7327.600098, 616838.      ],
       [  7332.      ,   7226.      ,   7328.950195, 571967.      ],
       ...,
       [  8853.200195,   8765.      ,   8844.      , 519621.      ],
       [  8800.      ,   8694.950195,   8777.200195, 551379.      ],
       [  8747.849609,   8641.      ,   8720.      , 546450.      ]])
```

In [15]:
```
y
```

Out[15]:
```
array([7311.700195, 7329.850098, 7254.25    , 7042.399902, 7073.600098,
       7035.200195, 7227.899902, 7330.5     , 7352.700195, 7386.799805,
       7482.950195, 7462.649902, 7520.149902, 7449.600098, 7302.649902,
       7135.600098, 7147.100098, 7128.450195, 7148.799805, 6997.049805,
       7010.299805, 7020.450195, 6913.5     , 7011.299805, 7199.600098,
       7040.899902, 7021.600098, 6971.75    , 6892.899902, 7033.75    ,
       7043.299805, 7005.049805, 6913.799805, 6906.75    , 6781.100098,
       6756.600098, 6757.600098, 6470.399902, 6414.549805, 6234.850098,
       6289.799805, 6283.100098, 6285.149902, 6386.049805, 6384.350098,
       6365.950195, 6445.950195, 6201.100098, 6146.299805, 5643.149902,
       5838.600098, 5480.25    , 5603.950195, 5352.649902, 4819.549805,
       5079.200195, 4220.350098, 4486.450195, 5005.950195, 4878.      ,
       4646.100098, 4328.450195, 4288.299805, 4246.350098, 4011.5     ,
       4553.649902, 4698.100098, 5326.649902, 5283.149902, 5094.299805,
       5130.799805, 5505.      , 5331.350098, 5001.100098, 5180.850098,
       5156.350098, 5045.649902, 5056.350098, 5052.700195, 5068.450195,
       5358.799805, 4886.299805, 4829.850098, 4843.25    , 4749.299805,
       4654.149902, 4937.799805, 4951.299805, 5036.100098, 5114.049805,
       5100.399902, 4720.950195, 4805.299805, 4891.950195, 5050.100098,
       5134.299805, 5246.149902, 5244.399902, 5468.350098, 5610.799805,
       5793.600098, 5690.100098, 5624.      , 5690.200195, 5746.200195,
       5734.799805, 5662.299805, 5675.600098, 5426.149902, 5561.75    ,
       5475.75    , 5502.200195, 5726.200195, 5714.399902, 5897.350098,
       5896.      , 5886.5     , 5755.350098, 5762.299805, 5754.899902,
       5678.700195, 5838.299805, 5803.100098, 5948.450195, 5932.100098,
       6123.600098, 6226.75    , 6044.399902, 6002.350098, 5955.649902,
       5988.850098, 5771.75    , 5801.299805, 5875.350098, 5919.5     ,
       5867.5     , 6130.850098, 6003.399902, 6065.950195, 6000.700195,
       6043.299805, 6282.799805, 6185.149902, 6265.399902, 6262.75    ,
       6165.200195, 6358.950195, 6526.25    , 6555.75    , 6679.      ,
       6608.899902, 6649.200195, 6730.299805, 6731.649902, 6588.200195,
       6768.399902, 6893.100098, 6990.299805, 6974.700195, 6973.      ,
       7082.5     , 7109.950195, 7008.649902, 7098.      , 7104.100098,
       6839.950195, 6905.649902, 6920.75    , 7072.600098, 7190.100098,
       7209.149902, 7224.      , 7215.25    , 7231.649902, 7193.600098,
       7128.850098, 7052.799805, 7057.950195, 7085.850098, 6964.75    ,
       6626.950195, 6440.549805, 6501.299805, 6297.700195, 6497.700195,
       6703.      , 6742.75    , 6743.450195, 6796.299805, 6818.850098,
       6892.399902, 7046.899902, 7079.850098, 7062.399902, 7136.649902,
       7032.899902, 7002.350098, 6892.600098, 6894.899902, 6851.950195,
       6844.600098, 6861.850098, 6811.549805, 7103.25    , 7054.799805,
       7158.399902, 7186.049805, 7117.700195, 6965.149902, 6868.350098,
       6913.350098, 6954.049805, 7092.799805, 6907.600098, 6872.149902,
       6793.450195, 6857.350098, 6836.850098, 6809.25    , 6852.25    ,
       6981.450195, 7054.299805, 6963.299805, 6965.850098, 6987.75    ,
       7158.399902, 7055.700195, 6994.25    , 7035.799805, 7101.700195,
       7206.5     , 7739.100098, 7803.149902, 7743.399902, 7762.700195,
```

```
7709.25    , 7736.      , 7733.549805, 7696.799805, 7765.25    ,
7796.350098, 7694.      , 7559.75    , 7376.100098, 7395.399902,
7449.950195, 7446.049805, 7483.      , 7452.350098, 7612.899902,
7649.600098, 7691.299805, 7702.299805, 7655.450195, 7628.600098,
7566.049805, 8014.899902, 8232.75    , 8188.049805, 8139.850098,
8149.450195, 8024.75    , 7830.649902, 7922.600098, 8144.950195,
8076.75    , 8048.850098, 7981.5     , 7870.899902, 7588.5     ,
7206.649902, 7399.799805, 7654.700195, 7589.200195, 7640.      ,
7494.149902, 7574.600098, 7627.049805, 7621.700195, 7667.350098,
7568.5     , 7596.649902, 7697.049805, 7503.200195, 7497.549805,
7323.      , 7086.450195, 6970.5     , 6952.950195, 6930.350098,
6866.149902, 7015.      , 7214.100098, 7124.700195, 7131.600098,
7249.      , 7259.5     , 7301.549805, 7272.100098, 7096.200195,
7089.299805, 7150.299805, 7064.649902, 7113.75    , 7113.549805,
7101.25    , 7186.25    , 7066.200195, 6786.850098, 6784.450195,
6851.799805, 6859.200195, 6923.899902, 6768.549805, 6751.299805,
6865.      , 6826.850098, 6827.100098, 6520.600098, 6815.799805,
6644.25    , 6648.100098, 6511.75    , 6646.649902, 6650.100098,
6676.100098, 6638.899902, 6568.75    , 6573.799805, 6565.649902,
6455.649902, 6597.850098, 6508.25    , 6590.100098, 6666.649902,
6703.049805, 6764.      , 6736.649902, 6817.549805, 6736.399902,
6717.850098, 6811.100098, 6775.899902, 6737.850098, 6814.649902,
6871.600098, 6911.799805, 7034.299805, 6968.899902, 6970.      ,
7086.299805, 7091.149902, 7184.700195, 7209.5     , 7214.700195,
7275.649902, 7336.75    , 7239.      , 7199.899902, 7223.799805,
7177.850098, 7166.25    , 7114.25    , 6969.350098, 6959.350098,
6899.899902, 7265.399902, 7432.700195, 7527.450195, 7649.      ,
7596.25    , 7487.5     , 7515.899902, 7584.399902, 7573.850098,
7599.450195, 7514.950195, 7449.399902, 7401.200195, 7425.700195,
7470.100098, 7430.350098, 7326.75    , 7307.      , 7303.549805,
7165.549805, 7232.700195, 7299.549805, 7293.850098, 7235.799805,
7240.25    , 7165.049805, 6993.5     , 6977.700195, 7076.950195,
7199.399902, 7102.899902, 7027.549805, 7100.799805, 7081.75    ,
7001.950195, 7021.5     , 7012.      , 7002.200195, 6826.850098,
6885.399902, 6840.100098, 6850.899902, 6825.899902, 6803.200195,
6711.450195, 6608.600098, 6624.850098, 6796.899902, 6846.100098,
6784.899902, 6792.899902, 6863.100098, 6846.899902, 6877.149902,
6781.25    , 6802.25    , 6873.700195, 6894.450195, 6909.      ,
6930.950195, 7014.450195, 6950.399902, 6777.      , 6831.      ,
6846.700195, 6952.25    , 7403.450195, 7423.850098, 7384.049805,
7338.049805, 7162.299805, 7170.299805, 7258.25    , 7199.25    ,
7492.5     , 7430.      , 7700.799805, 7693.799805, 7482.149902,
7496.899902, 7657.      , 7655.649902, 7590.75    , 7575.25    ,
7408.899902, 7260.600098, 7297.350098, 7356.25    , 7369.700195,
7482.399902, 7615.549805, 7791.75    , 7734.25    , 7747.399902,
7684.600098, 7577.549805, 7511.649902, 7453.450195, 7479.850098,
7503.100098, 8050.350098, 8274.75    , 8117.149902, 7864.399902,
7854.      , 7669.450195, 7572.5     , 7170.5     , 7149.5     ,
7067.799805, 7273.25    , 7324.950195, 7208.700195, 7042.850098,
7186.600098, 7434.899902, 7421.75    , 7425.649902, 7516.399902,
7493.149902, 7567.950195, 7450.799805, 7296.600098, 7286.600098,
7315.149902, 7423.75    , 7387.149902, 7317.100098, 7289.5     ,
7297.450195, 7350.049805, 7282.25    , 7426.450195, 7523.899902,
7630.100098, 7775.350098, 7882.100098, 7906.      , 8125.600098,
8143.850098, 8170.75    , 8077.049805, 8084.200195, 8265.5     ,
7915.149902, 8015.700195, 8036.350098, 8189.600098, 8052.299805,
8602.599609, 8820.200195, 8550.950195, 8597.299805, 8559.400391,
8515.25    , 8593.650391, 8550.549805, 8511.650391, 8599.450195,
8949.450195, 8805.450195, 8737.150391, 8366.400391, 8622.700195,
8582.950195, 8552.450195, 8569.099609, 8612.700195, 8622.799805,
```

```
8698.849609, 8210.150391, 8356.099609, 8314.150391, 7814.200195,
7595.549805, 7247.299805, 6769.049805, 6806.299805, 7026.850098,
7211.799805, 7102.549805, 7316.75    , 7420.100098, 7485.799805,
7693.      , 7705.549805, 7766.649902, 7644.850098, 7556.950195,
7415.450195, 7477.350098, 7439.149902, 7598.600098, 7561.299805,
7700.049805, 7774.799805, 7761.850098, 7744.200195, 7640.5     ,
7557.950195, 7565.399902, 7618.350098, 7469.5     , 7573.5     ,
7435.100098, 7666.700195, 7879.200195, 7904.5     , 7912.350098,
7906.75    , 7788.399902, 7888.149902, 7717.799805, 7634.75    ,
7397.899902, 7407.649902, 7279.25    , 7360.299805, 7515.899902,
7396.450195, 7253.649902, 7101.950195, 7247.600098, 7532.799805,
7566.549805, 7404.549805, 7586.600098, 7897.450195, 7802.649902,
7787.25    , 7805.      , 7942.950195, 8013.899902, 7966.350098,
7939.399902, 7925.399902, 7705.049805, 7710.549805, 7809.399902,
7892.450195, 7929.850098, 7935.100098, 7901.600098, 7811.649902,
7853.450195, 7833.600098, 7688.149902, 7660.950195, 7780.600098,
7782.75    , 8271.      , 8363.200195, 8448.75    , 8489.700195,
8508.900391, 8470.75    , 8402.599609, 8443.349609, 8348.700195,
8630.049805, 8606.400391, 8475.950195, 8507.650391, 8419.700195,
8440.200195, 8567.75    , 8778.049805, 8704.799805, 8715.849609,
8749.049805, 8797.      , 8830.950195, 8621.599609, 8525.849609,
8658.099609, 8721.049805, 8773.549805, 9003.400391, 9173.049805,
8959.450195, 8966.700195, 8841.200195, 8921.950195, 8879.700195,
8813.849609, 8699.200195, 9003.700195, 8944.349609, 8947.700195,
8778.950195, 8628.      , 8720.5     , 8704.      , 8735.900391,
8720.599609, 8834.200195, 9082.25    , 9024.5     , 8919.599609,
8950.049805, 8888.75    , 8784.900391, 8790.650391, 8946.      ,
8927.650391, 8926.5     , 8967.200195, 9209.75    , 9189.650391,
9280.849609, 9300.849609, 9247.900391, 9401.849609, 9343.450195,
8834.950195, 8774.049805, 8726.75    , 8624.849609, 8828.150391,
8547.      , 8690.5     , 8703.      , 8779.099609, 8862.400391,
8684.150391, 8686.599609, 8649.549805, 8617.299805, 8723.799805,
8807.349609, 8719.349609, 8707.400391, 8703.299805, 8765.450195,
9005.      , 9041.950195, 9492.549805, 9527.599609, 9454.599609,
9229.400391, 9239.099609, 9229.5     , 9333.299805, 9253.5     ,
9097.25    , 9087.549805, 9152.349609, 9162.200195, 9134.950195,
8986.450195, 8848.049805, 8865.200195, 8873.75    , 8934.5     ,
8967.099609, 9019.5     , 8985.400391, 8910.200195, 8974.150391,
8958.150391, 8815.849609, 8792.049805, 8717.349609, 8659.150391])
```

Spliting the Train and Test data

In [16]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=1)
```

Linear Regression Model

In [17]:
```
regressor = LinearRegression()
```

Fitting the data in the Model

In [18]:
```
regressor.fit(X_train, y_train)
```

Out[18]:
```
▾ LinearRegression
LinearRegression()
```

In [19]:
```
print(regressor.coef_)
```

```
[ 8.52668406e-01  8.05394707e-01 -6.63234937e-01  4.20515576e-07]
```

In [20]:
```
print(regressor.intercept_)
```

```
29.570629497657137
```

Prediction

In [21]:
```
predicted = regressor.predict(X_test)
```

In [22]:
```
print(predicted)
```

```
[7450.31296312 7313.54665853 6977.90036148 8736.60499262 7973.97719918
 8767.31475906 8105.81610571 7062.45208598 9130.02700768 7135.9729919
 5725.65497265 5132.10266859 7314.23643825 6543.55016797 8685.55977246
 7287.68472774 6675.90768947 7241.01059359 8475.37171994 7848.14354675
 4889.02979801 8485.9024683  7047.61243491 7684.23976649 7138.22694321
 7410.11320244 9305.13832793 7984.75327361 7746.6372014  6182.20722371
 7200.82730547 7953.58517068 5457.36092731 7241.86796922 7672.64424952
 7101.99395438 7000.75991672 7925.49800693 7277.87490364 6795.18904888
 5909.00910859 7574.48432914 6898.83652022 6789.17471997 7121.29937664
 7590.56945939 8738.33708435 8488.72613235 6723.70656871 7569.49023869
 5702.87121489 8074.30735503 8934.77141744 5536.84003223 7495.40629879
 7075.23785936 4415.33615363 7874.54307946 7214.58394368 7032.44248281
 7771.09549899 4695.9398574  7056.93888527 4922.97072396 6608.78694921
 6603.43267537 6733.44342953 6950.38174504 4383.03030566 5491.86156252
 7696.23487518 7418.00751128 7959.7753438  7061.43896075 9168.30079139
 8792.01222663 8797.11673603 7177.45194183 4529.77625781 9120.8910407
 8866.58604099 7560.23533283 7405.66379597 4293.63745341 7178.45239924
 8565.7122329  5731.04283288 6960.31965452 7894.13624284 7399.32640313
 6667.26340847 7675.86144456 8920.46905848 6588.11050138 7519.27707001
 6287.36174215 7183.95271499 7837.43200585 6665.67569271 8772.26805932
 8883.36937063 7320.30820036 7077.33187556 8602.44433168 6847.61479134
 5203.00626586 6967.08780845 7678.52412447 4856.56650772 6806.79911507
 5769.87164129 7340.94028459 8738.52410623 7711.81553994 6493.65987841
 8788.62487034 8171.39020404 7079.63805904 7038.56641537 9013.13135469
 4802.45582236 6935.8089049  8755.98910486 4692.91941425 7537.13692645
 7675.66166501 7438.4273529  6791.84431393 8935.62084015 7674.77986022
 4827.40465236 7533.7084843  7783.31296527 7674.31593431 7194.32651015
 7107.90787912 7300.92498725 7336.91141343 5825.26596971 7894.00004981
 6993.24870666 8742.92378904 7948.30039812 8679.69542543 6269.20740309
 8443.18402451 5109.22307633 5081.37372459 7814.39150535 9135.42271241
 5977.92865131 7641.67147855 7186.31540455 5801.58530862 6566.21739114
 6902.91713746 6770.98756681 6823.1924467  8540.11074262 8924.12719563
 9371.96956241 8276.53746716 5723.98132136 4871.8053974  6528.61735413
 6707.29098726 6388.4302556  5696.18729132 7163.61069421 8823.62418382
 6826.40876745 8473.46953973 6807.26392083 6879.45446474 7171.21246658
 7201.53180375 6387.2045     8839.63423064 6773.51288708 7426.4666094
```

```
7474.34297591 5098.89808785 6733.30804312 8471.61346607 6557.25595707
6808.51005721 7796.27067087 7096.1169289  7111.13185707 7967.14541139
7086.55210361 6798.68954286 7209.28054412 5801.62788831 5629.64920043
6947.85504523 7460.03888352 7043.81339208 7018.82652978 8687.31155589
7015.349418   9014.68150524 8603.63118311 5960.88880785 7533.07748509
8935.86758098 5070.01990245 7739.32333967 6675.65758214 8166.44266897
6941.36488441 7390.72749032 7606.48225441 4279.55179396 7048.95737958
7576.48408283 7588.48218554 7365.50591143 7632.25916309]
```

Combine the Actual and Predicted data

In [23]:
```
data1 = pd.DataFrame({'Actual': y_test.flatten(), 'Predicted' : predicted.flatten()})
```

In [24]:
```
data1.head(20)
```

Out[24]:

	Actual	Predicted
0	7452.350098	7450.312963
1	7272.100098	7313.546659
2	6973.000000	6977.900361
3	8720.599609	8736.604993
4	8015.700195	7973.977199
5	8726.750000	8767.314759
6	8036.350098	8105.816106
7	7102.899902	7062.452086
8	9189.650391	9130.027008
9	7124.700195	7135.972992
10	5678.700195	5725.654973
11	5094.299805	5132.102669
12	7352.700195	7314.236438
13	6520.600098	6543.550168
14	8720.500000	8685.559772
15	7317.100098	7287.684728
16	6626.950195	6675.907689
17	7289.500000	7241.010594
18	8448.750000	8475.371720
19	7791.750000	7848.143547

Mean Absolute Error

In [25]:
```
import math
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test,predicted))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,predicted))
print('Root Mean Squared Error:', math.sqrt(metrics.mean_squared_error(y_test,predicted
```

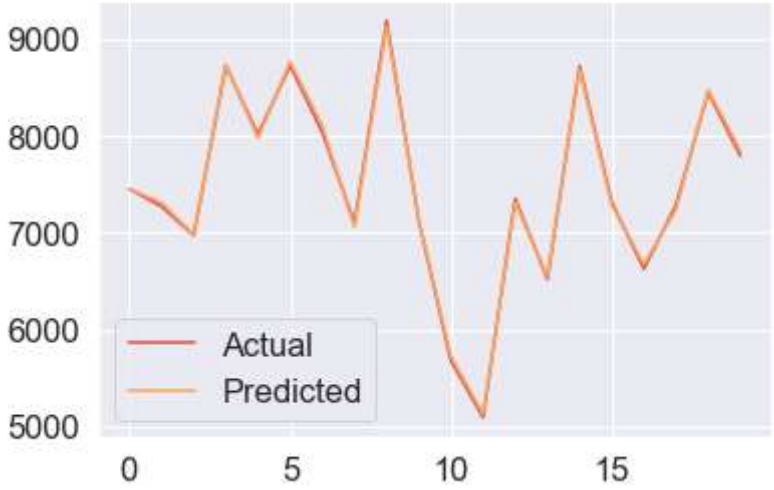
```
Mean Absolute Error: 36.39733935781136
Mean Squared Error: 2727.242127802716
Root Mean Squared Error: 52.22300381826687
```

Plotting Graph

In [26]:
```
graph = data1.head(20)
```

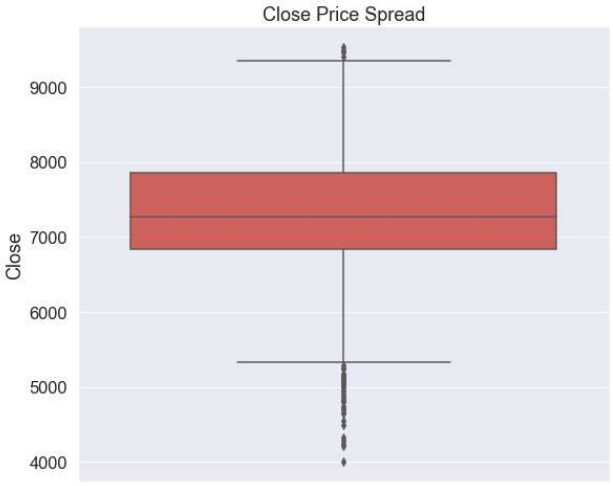
In [28]:
```
graph.plot(kind='line')
```

Out[28]:
```
<AxesSubplot:>
```

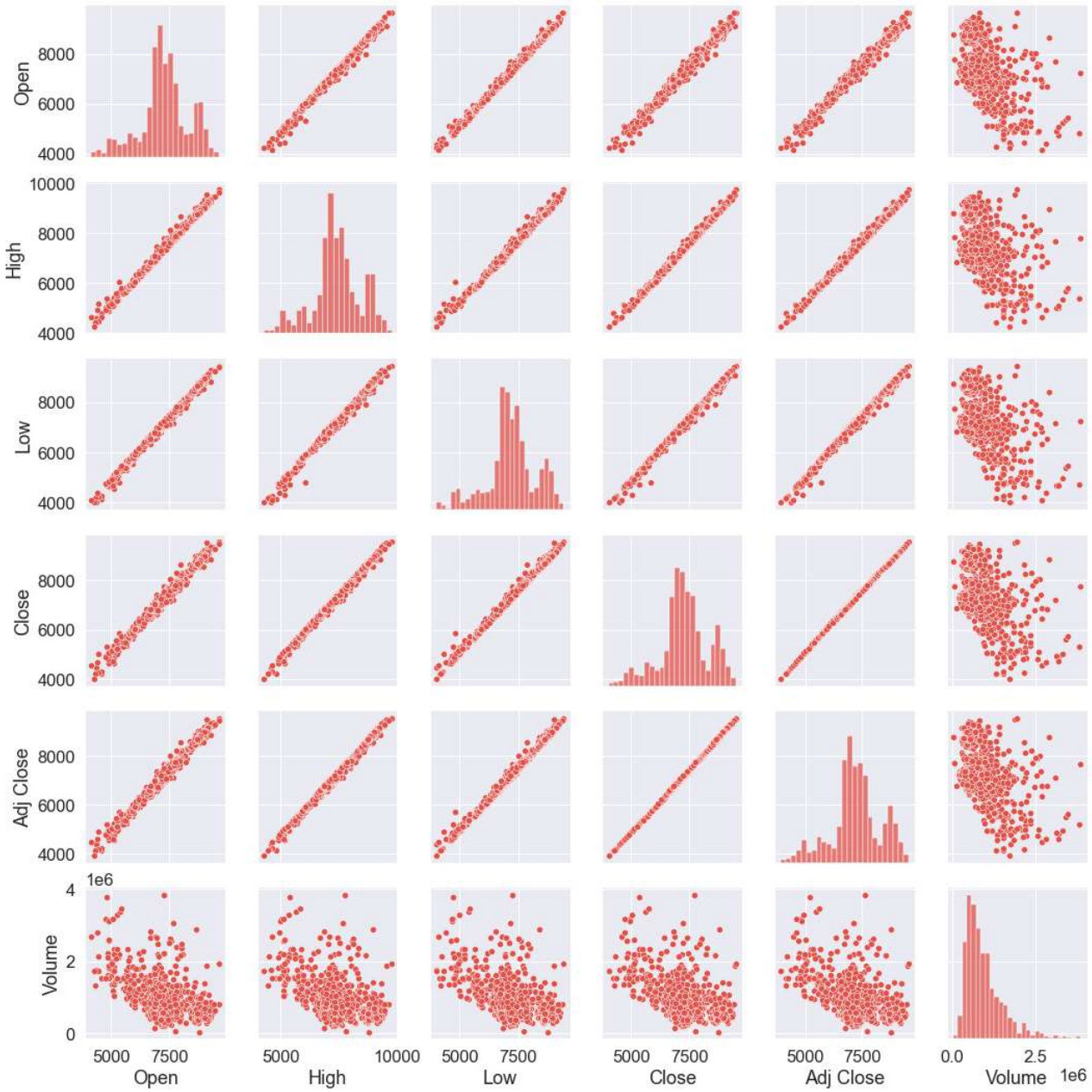


In [29]:
```
plt.figure(figsize=(20,8))

plt.subplot(1,2,1)
plt.title('Close Price ')
sns.distplot(data.Close)

plt.subplot(1,2,2)
plt.title('Close Price Spread')
sns.boxplot(y=data.Close)

plt.show()
```

```
C:\Users\Galaxy star Gopi\anaconda3\lib\site-packages\seaborn\distributions.py:2619: Fut
ureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
```



In [30]:
```
data = sns.pairplot(data)
```

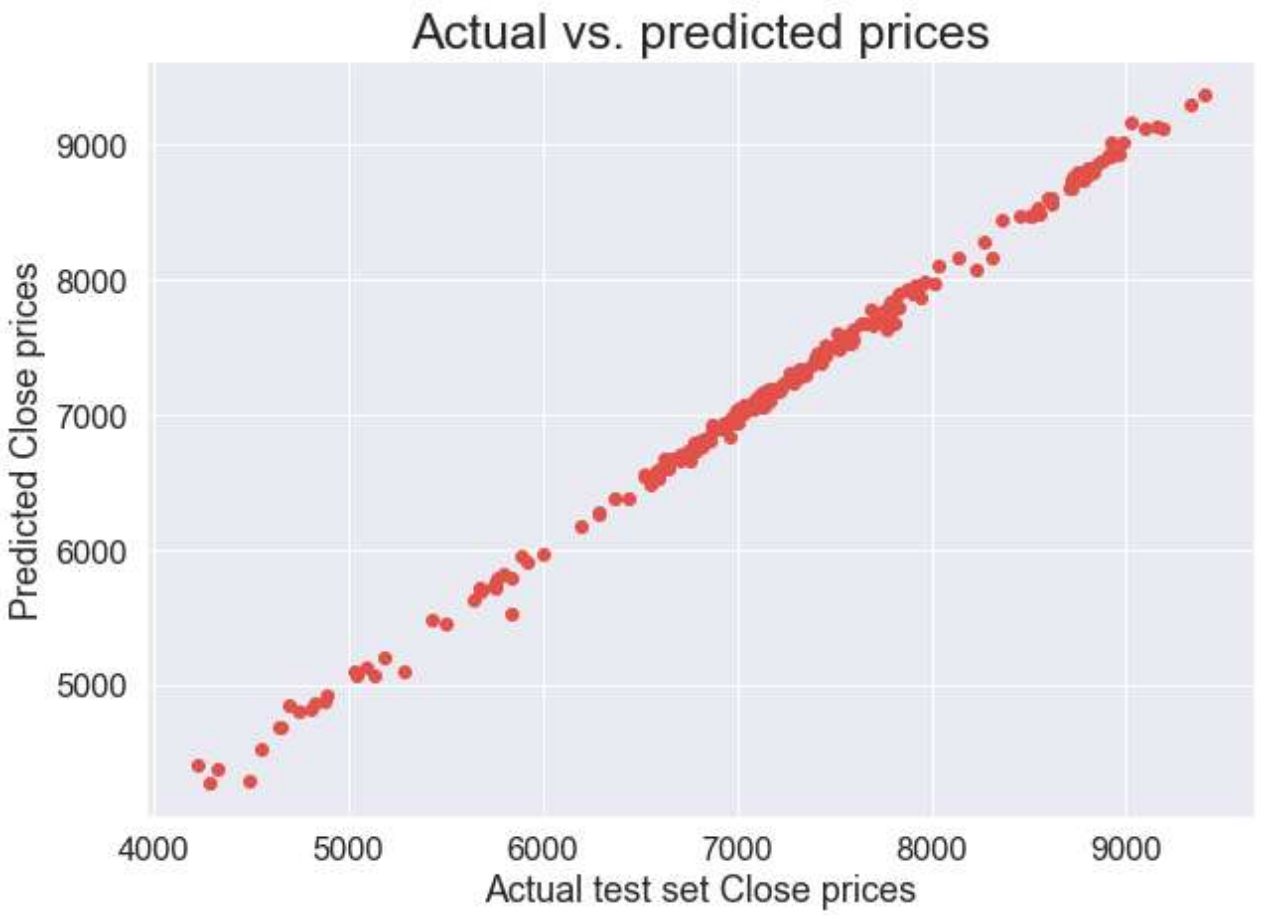


In [38]:
```
predictions = regressor.predict(X_test)
print ("Type of the predicted object:", type(predictions))
print ("Size of the predicted object:", predictions.shape)
```

```
Type of the predicted object: <class 'numpy.ndarray'>
Size of the predicted object: (219,)
```

In [40]:
```
plt.figure(figsize=(10,7))
plt.title("Actual vs. predicted prices",fontsize=25)
plt.xlabel("Actual test set Close prices",fontsize=18)
plt.ylabel("Predicted Close prices", fontsize=18)
plt.scatter(x=y_test,y=predictions)
```

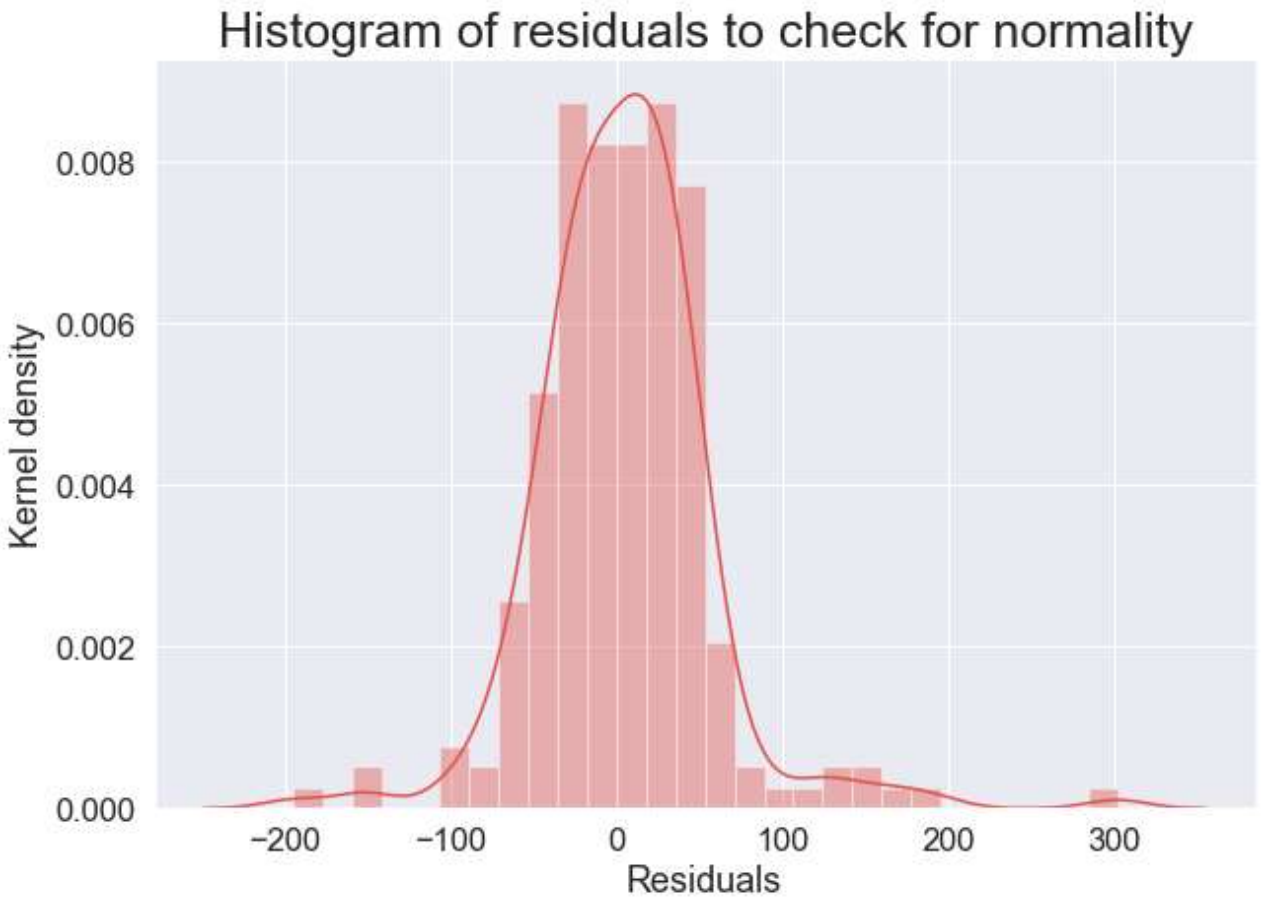
Out[40]:
```
<matplotlib.collections.PathCollection at 0x28d0a307880>
```



In [44]:
```
plt.figure(figsize=(10,7))
plt.title("Histogram of residuals to check for normality",fontsize=25)
plt.xlabel("Residuals",fontsize=18)
plt.ylabel("Kernel density", fontsize=18)
sns.distplot([y_test-predictions])
```

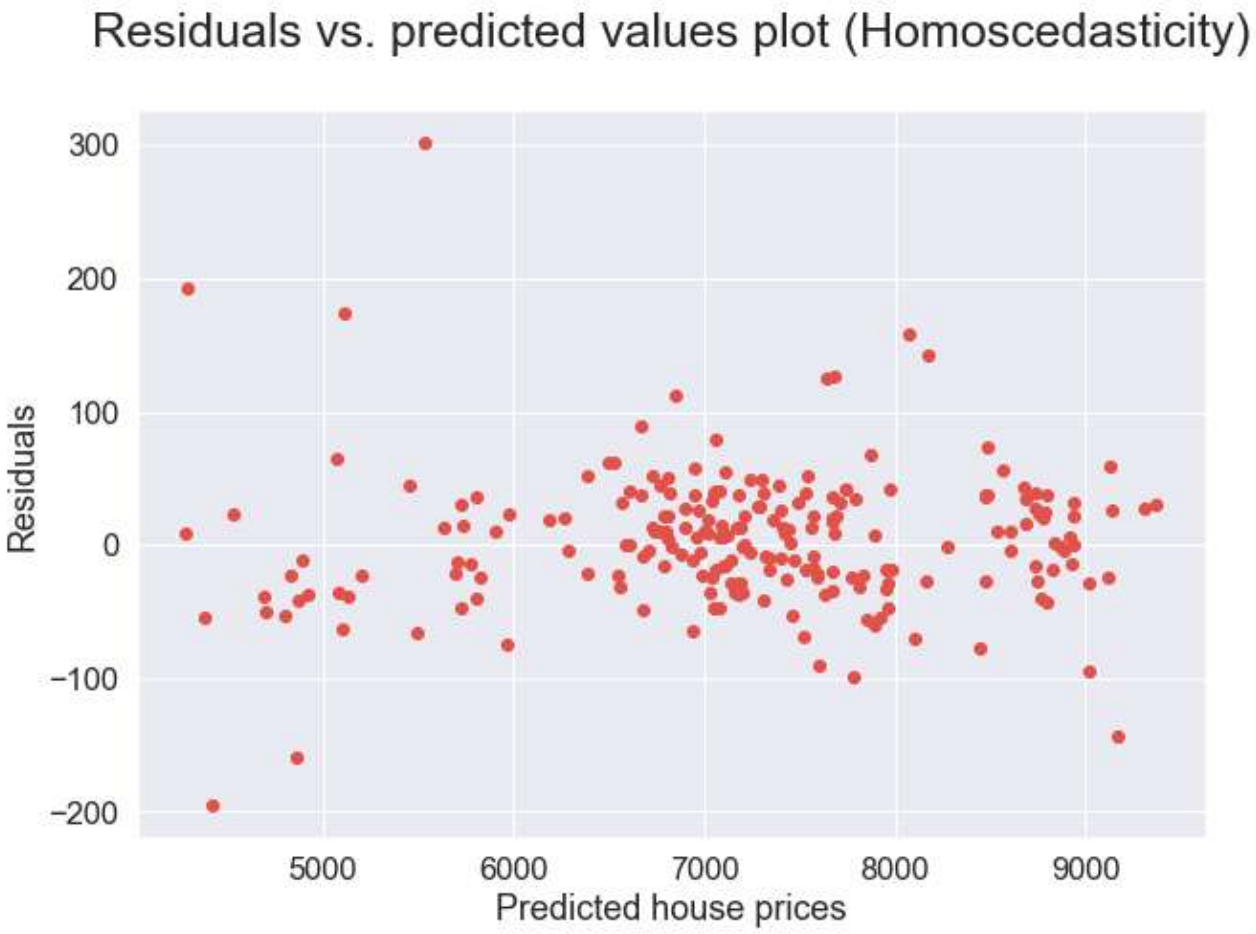
```
C:\Users\Galaxy star Gopi\anaconda3\lib\site-packages\seaborn\distributions.py:2619: Fut
ureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
```

Out[44]: <AxesSubplot:title={'center':'Histogram of residuals to check for normality'}, xlabel='Residuals', ylabel='Kernel density'>



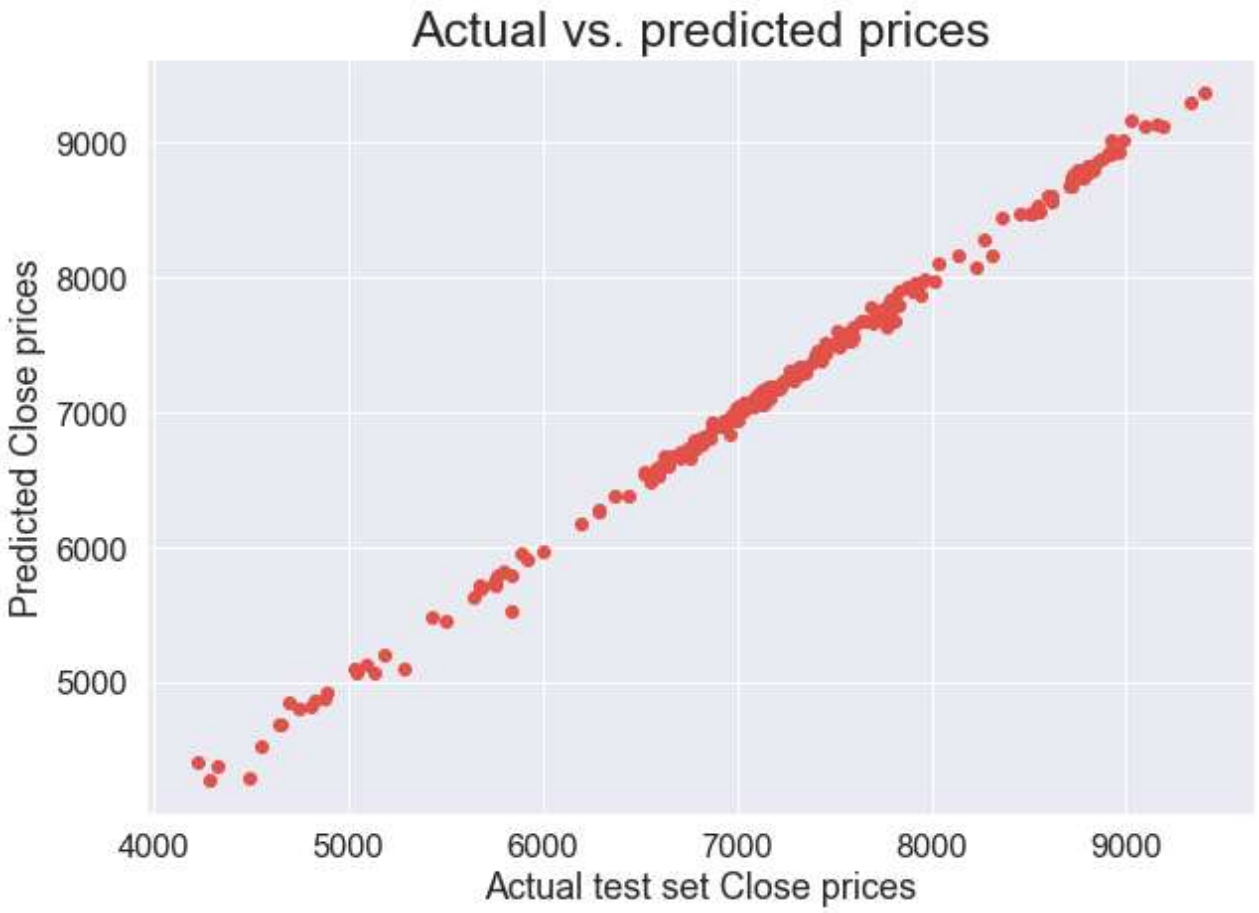
In [42]:
```
plt.figure(figsize=(10,7))
plt.title("Residuals vs. predicted values plot (Homoscedasticity)\n",fontsize=25)
plt.xlabel("Predicted house prices",fontsize=18)
plt.ylabel("Residuals", fontsize=18)
plt.scatter(x=predictions,y=y_test-predictions)
```

Out[42]: <matplotlib.collections.PathCollection at 0x28d0a4d4610>



In [45]:
```
plt.figure(figsize=(10,7))
plt.title("Actual vs. predicted prices",fontsize=25)
plt.xlabel("Actual test set Close prices",fontsize=18)
plt.ylabel("Predicted Close prices", fontsize=18)
plt.scatter(x=y_test,y=predictions)
```

Out[45]:
```
<matplotlib.collections.PathCollection at 0x28d0aa1a640>
```



In [31]:
```
data1.head(20)
```

Out[31]:

	Actual	Predicted
0	7452.350098	7450.312963
1	7272.100098	7313.546659
2	6973.000000	6977.900361
3	8720.599609	8736.604993
4	8015.700195	7973.977199
5	8726.750000	8767.314759
6	8036.350098	8105.816106
7	7102.899902	7062.452086
8	9189.650391	9130.027008
9	7124.700195	7135.972992
10	5678.700195	5725.654973
11	5094.299805	5132.102669
12	7352.700195	7314.236438
13	6520.600098	6543.550168
14	8720.500000	8685.559772

	Actual	Predicted
15	7317.100098	7287.684728
16	6626.950195	6675.907689
17	7289.500000	7241.010594
18	8448.750000	8475.371720
19	7791.750000	7848.143547

In [ ]: