In [62]:

```
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression
from sklearn.model_selection  import train_test_split
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
plt.style.use('bmh')
```

In [63]:

```
df=pd.read_csv("TSLA.csv")
df.head(6)
```

Out[63]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-10-23	421.839996	422.890015	407.380005	420.630005	420.630005	33717000
1	2020-10-26	411.630005	425.760010	410.000000	420.279999	420.279999	28239200
2	2020-10-27	423.760010	430.500000	420.100006	424.679993	424.679993	22686500
3	2020-10-28	416.480011	418.600006	406.000000	406.019989	406.019989	25451400
4	2020-10-29	409.959991	418.059998	406.459991	410.829987	410.829987	22655300
5	2020-10-30	406.899994	407.589996	379.109985	388.040009	388.040009	42511300

In [64]:

```
df.shape
```

Out[64]: (252, 7)

In [65]:

```
plt.figure(figsize=(16,8))
plt.title('TESLA')
plt.xlabel('Days')
plt.ylabel('Close price USD ($)')
plt.plot(df['Close'])
plt.show()
```

In [66]:

```
df=df[['Close']]
df.head(4)
```

Out[66]:

	Close
0	420.630005
1	420.279999
2	424.679993
3	406.019989

In [67]:

```
future_days=25
df['Prediction']=df[['Close']].shift(-future_days)
df.tail(4)
```

Out[67]:

	Close	Prediction
248	864.270020	NaN
249	865.799988	NaN
250	894.000000	NaN
251	909.679993	NaN

In [68]:

```
X=np.array(df.drop(['Prediction'],1))[:-future_days]
print(X)
```

```
[[420.630005]
 [420.279999]
 [424.679993]
 [406.019989]
 [410.829987]
 [388.040009]
 [400.51001 ]
 [423.899994]
 [420.980011]
 [438.089996]
 [429.950012]
 [421.26001 ]
 [410.359985]
 [417.130005]
 [411.76001 ]
 [408.5     ]
 [408.089996]
 [441.609985]
 [486.640015]
 [499.269989]
 [489.609985]
 [521.849976]
 [555.380005]
 [574.      ]
 [585.76001 ]
 [567.599976]
 [584.76001 ]
 [568.820007]
 [593.380005]
 [599.039978]
 [641.76001 ]
 [649.880005]
 [604.47998 ]
 [627.070007]
 [609.98999 ]
 [639.830017]
 [633.25    ]
 [622.77002 ]
 [655.900024]
 [695.      ]
 [649.859985]
 [640.340027]
 [645.97998 ]
 [661.77002 ]
 [663.690002]
 [665.98999 ]
 [694.780029]
 [705.669983]
 [729.77002 ]
 [735.109985]
 [755.97998 ]
 [816.039978]
 [880.02002 ]
 [811.190002]
 [849.440002]
 [854.409973]
 [845.      ]
 [826.159973]
 [844.549988]
 [850.450012]
 [844.98999 ]
 [846.640015]
 [880.799988]
 [883.090027]
 [864.159973]
 [835.429993]
 [793.530029]
 [839.809998]
 [872.789978]
 [854.690002]
 [849.98999 ]
 [852.22998 ]
 [863.419983]
 [849.460022]
 [804.820007]
 [811.659973]
 [816.119995]
 [796.219971]
 [798.150024]
 [787.380005]
 [781.299988]
 [714.5     ]
 [698.840027]
 [742.02002 ]
 [682.219971]
 [675.5     ]
 [718.429993]
 [686.440002]
 [653.200012]
 [621.440002]
 [597.950012]
 [563.      ]
 [673.580017]
 [668.059998]
 [699.599976]
 [693.72998 ]
 [707.940002]
 [676.880005]
 [701.809998]
 [653.159973]
 [654.869995]
 [670.      ]
 [662.159973]
 [630.27002 ]
 [640.390015]
 [618.710022]
 [611.289978]
 [635.619995]
 [667.929993]
 [661.75    ]
 [691.049988]
 [691.619995]
 [670.969971]
 [683.799988]
 [677.02002 ]
 [701.97998 ]
 [762.320007]
 [732.22998 ]
 [738.849976]
 [739.780029]
 [714.630005]
 [718.98999 ]
 [744.119995]
 [719.690002]
 [729.400024]
 [738.200012]
 [704.73999 ]
 [694.400024]
 [677.      ]
 [709.440002]
 [684.900024]
 [673.599976]
 [670.940002]
 [663.539978]
 [672.369995]
 [629.039978]
 [617.200012]
 [589.890015]
 [571.690002]
 [589.73999 ]
 [576.830017]
 [577.869995]
 [563.460022]
 [586.780029]
 [580.880005]
 [606.440002]
 [604.690002]
 [619.130005]
 [630.849976]
 [625.219971]
 [623.900024]
 [605.119995]
 [572.840027]
 [599.049988]
 [605.130005]
 [603.590027]
 [598.780029]
 [610.119995]
 [609.890015]
 [617.690002]
 [599.359985]
 [604.869995]
 [616.599976]
 [623.309998]
 [620.830017]
 [623.710022]
 [656.570007]
 [679.820007]
 [671.869995]
 [688.719971]
 [680.76001 ]
 [679.700012]
 [677.919983]
 [678.900024]
 [659.580017]
 [644.650024]
 [652.809998]
 [656.950012]
 [685.700012]
 [668.539978]
 [653.380005]
 [650.599976]
 [644.219971]
 [646.219971]
 [660.5     ]
 [655.289978]
 [649.26001 ]
 [643.380005]
 [657.619995]
 [644.780029]
 [646.97998 ]
 [677.349976]
 [687.200012]
 [709.669983]
 [709.73999 ]
 [710.919983]
 [714.630005]
 [699.099976]
 [713.76001 ]
 [709.98999 ]
 [707.820007]
 [722.25    ]
 [717.169983]
 [686.169983]
 [665.710022]
 [688.98999 ]
 [673.469971]
 [680.26001 ]
 [706.299988]
 [708.48999 ]
 [711.200012]
 [701.159973]
 [711.919983]
 [730.909973]
 [735.719971]
 [734.090027]
 [732.390015]
 [733.570007]
 [752.919983]
 [753.869995]
 [754.859985]
 [736.27002 ]
 [743.      ]
 [744.48999 ]
 [755.830017]
 [756.98999 ]
 [759.48999 ]]
```

```
<ipython-input-68-0522a47ccbe7>:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  X=np.array(df.drop(['Prediction'],1))[:-future_days]
```

In [69]:

```
y=np.array(df.drop(['Prediction'],1))[:-future_days]
print(y)
```

```
[[420.630005]
 [420.279999]
 [424.679993]
 [406.019989]
 [410.829987]
 [388.040009]
 [400.51001 ]
 [423.899994]
 [420.980011]
 [438.089996]
 [429.950012]
 [421.26001 ]
 [410.359985]
 [417.130005]
 [411.76001 ]
 [408.5     ]
 [408.089996]
 [441.609985]
 [486.640015]
 [499.269989]
 [489.609985]
 [521.849976]
 [555.380005]
 [574.      ]
 [585.76001 ]
 [567.599976]
 [584.76001 ]
 [568.820007]
 [593.380005]
 [599.039978]
 [641.76001 ]
 [649.880005]
 [604.47998 ]
 [627.070007]
 [609.98999 ]
 [639.830017]
 [633.25    ]
 [622.77002 ]
 [655.900024]
 [695.      ]
 [649.859985]
 [640.340027]
 [645.97998 ]
 [661.77002 ]
 [663.690002]
 [665.98999 ]
 [694.780029]
 [705.669983]
 [729.77002 ]
 [735.109985]
 [755.97998 ]
 [816.039978]
 [880.02002 ]
 [811.190002]
 [849.440002]
 [854.409973]
 [845.      ]
 [826.159973]
 [844.549988]
 [850.450012]
 [844.98999 ]
 [846.640015]
 [880.799988]
 [883.090027]
 [864.159973]
 [835.429993]
 [793.530029]
 [839.809998]
 [872.789978]
 [854.690002]
 [849.98999 ]
 [852.22998 ]
 [863.419983]
 [849.460022]
 [804.820007]
 [811.659973]
 [816.119995]
 [796.219971]
 [798.150024]
 [787.380005]
 [781.299988]
 [714.5     ]
 [698.840027]
 [742.02002 ]
 [682.219971]
 [675.5     ]
 [718.429993]
 [686.440002]
 [653.200012]
 [621.440002]
 [597.950012]
 [563.      ]
 [673.580017]
 [668.059998]
 [699.599976]
 [693.72998 ]
 [707.940002]
 [676.880005]
 [701.809998]
 [653.159973]
 [654.869995]
 [670.      ]
 [662.159973]
 [630.27002 ]
 [640.390015]
 [618.710022]
 [611.289978]
 [635.619995]
 [667.929993]
 [661.75    ]
 [691.049988]
 [691.619995]
 [670.969971]
 [683.799988]
 [677.02002 ]
 [701.97998 ]
 [762.320007]
 [732.22998 ]
 [738.849976]
 [739.780029]
 [714.630005]
 [718.98999 ]
 [744.119995]
 [719.690002]
 [729.400024]
 [738.200012]
 [704.73999 ]
 [694.400024]
 [677.      ]
 [709.440002]
 [684.900024]
 [673.599976]
 [670.940002]
 [663.539978]
 [672.369995]
 [629.039978]
 [617.200012]
 [589.890015]
 [571.690002]
 [589.73999 ]
 [576.830017]
 [577.869995]
 [563.460022]
 [586.780029]
 [580.880005]
 [606.440002]
 [604.690002]
 [619.130005]
 [630.849976]
 [625.219971]
 [623.900024]
 [605.119995]
 [572.840027]
 [599.049988]
 [605.130005]
 [603.590027]
 [598.780029]
 [610.119995]
 [609.890015]
 [617.690002]
 [599.359985]
 [604.869995]
 [616.599976]
 [623.309998]
 [620.830017]
 [623.710022]
 [656.570007]
 [679.820007]
 [671.869995]
 [688.719971]
 [680.76001 ]
 [679.700012]
 [677.919983]
 [678.900024]
 [659.580017]
 [644.650024]
 [652.809998]
 [656.950012]
 [685.700012]
 [668.539978]
 [653.380005]
 [650.599976]
 [644.219971]
 [646.219971]
 [660.5     ]
 [655.289978]
 [649.26001 ]
 [643.380005]
 [657.619995]
 [644.780029]
 [646.97998 ]
 [677.349976]
 [687.200012]
 [709.669983]
 [709.73999 ]
 [710.919983]
 [714.630005]
 [699.099976]
 [713.76001 ]
 [709.98999 ]
 [707.820007]
 [722.25    ]
 [717.169983]
 [686.169983]
 [665.710022]
 [688.98999 ]
 [673.469971]
 [680.26001 ]
 [706.299988]
 [708.48999 ]
 [711.200012]
 [701.159973]
 [711.919983]
 [730.909973]
 [735.719971]
 [734.090027]
 [732.390015]
 [733.570007]
 [752.919983]
 [753.869995]
 [754.859985]
 [736.27002 ]
 [743.      ]
 [744.48999 ]
 [755.830017]
 [756.98999 ]
 [759.48999 ]]
```

```
<ipython-input-69-7f05d2eedbf2>:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  y=np.array(df.drop(['Prediction'],1))[:-future_days]
```

In [70]:

```
#splitting the data into 75% training set and 25% testing
x_train , x_test , y_train , y_test = train_test_split(X ,y , random_state = 0)
```

In [71]:

```
tree=DecisionTreeRegressor().fit(x_train,y_train)
lr=LinearRegression().fit(x_train,y_train)
```

In [72]:

```
x_future=df.drop(['Prediction'],1)[:-future_days]
x_future=x_future.tail(future_days)
x_future=np.array(x_future)
x_future
```

```
<ipython-input-72-6675df72f1ab>:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  x_future=df.drop(['Prediction'],1)[:-future_days]
```

Out[72]:

```
array([[717.169983],
       [686.169983],
       [665.710022],
       [688.98999 ],
       [673.469971],
       [680.26001 ],
       [706.299988],
       [708.48999 ],
       [711.200012],
       [701.159973],
       [711.919983],
       [730.909973],
       [735.719971],
       [734.090027],
       [732.390015],
       [733.570007],
       [752.919983],
       [753.869995],
       [754.859985],
       [736.27002 ],
       [743.      ],
       [744.48999 ],
       [755.830017],
       [756.98999 ],
       [759.48999 ]])
```

In [73]:

```
tree_prediction=tree.predict(x_future)
print(tree_prediction)
print()
lr_prediction=lr.predict(x_future)
print(lr_prediction)
print()
```

```
[717.169983 686.440002 665.710022 688.98999  673.599976 680.26001
 705.669983 707.820007 711.200012 701.159973 711.919983 730.909973
 736.27002  734.090027 732.390015 733.570007 752.919983 753.869995
 754.859985 736.27002  743.       744.48999  755.830017 756.98999
 759.48999 ]

[[717.169983]
 [686.169983]
 [665.710022]
 [688.98999 ]
 [673.469971]
 [680.26001 ]
 [706.299988]
 [708.48999 ]
 [711.200012]
 [701.159973]
 [711.919983]
 [730.909973]
 [735.719971]
 [734.090027]
 [732.390015]
 [733.570007]
 [752.919983]
 [753.869995]
 [754.859985]
 [736.27002 ]
 [743.      ]
 [744.48999 ]
 [755.830017]
 [756.98999 ]
 [759.48999 ]]
```

In [74]:

```
predictions=tree_prediction


valid=df[X.shape[0]:]
valid['Predictions']=predictions
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Days')
plt.ylabel('Close price USD ($)')
plt.plot(df['Close'])
plt.plot(valid[['Close','Predictions']])
plt.legend('Orig','Val','Pred')
plt.show()
```

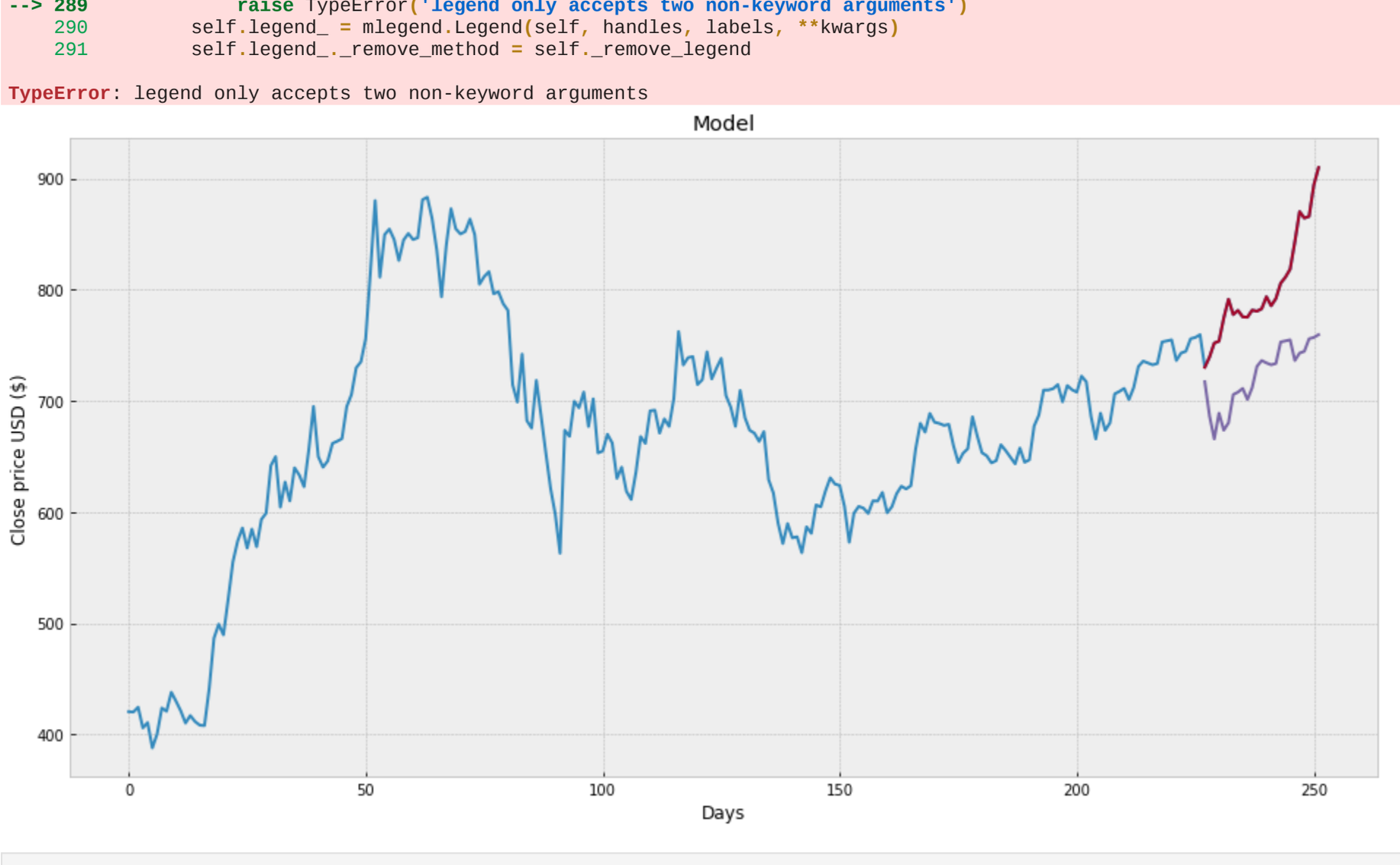
```
c:\users\invest 360\appdata\local\programs\python\python39-32\lib\site-packages\pandas\core\frame.py:3607: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self._set_item(key, value)
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-74-40b903f5f904> in <module>
     10 plt.plot(df['Close'])
     11 plt.plot(valid[['Close','Predictions']])
---> 12 plt.legend('Orig','Val','Pred')
     13 plt.show()

~\AppData\Roaming\Python\Python39\site-packages\matplotlib\pyplot.py in legend(*args, **kwargs)
   2915 @_copy_docstring_and_deprecators(Axes.legend)
   2916 def legend(*args, **kwargs):
-> 2917     return gca().legend(*args, **kwargs)
   2918
   2919

~\AppData\Roaming\Python\Python39\site-packages\matplotlib\axes\_axes.py in legend(self, *args, **kwargs)
    287                 **kwargs)
    288         if len(extra_args):
--> 289             raise TypeError('legend only accepts two non-keyword arguments')
    290         self.legend_ = mlegend.Legend(self, handles, labels, **kwargs)
    291         self.legend_._remove_method = self._remove_legend

TypeError: legend only accepts two non-keyword arguments
```



In [ ]:

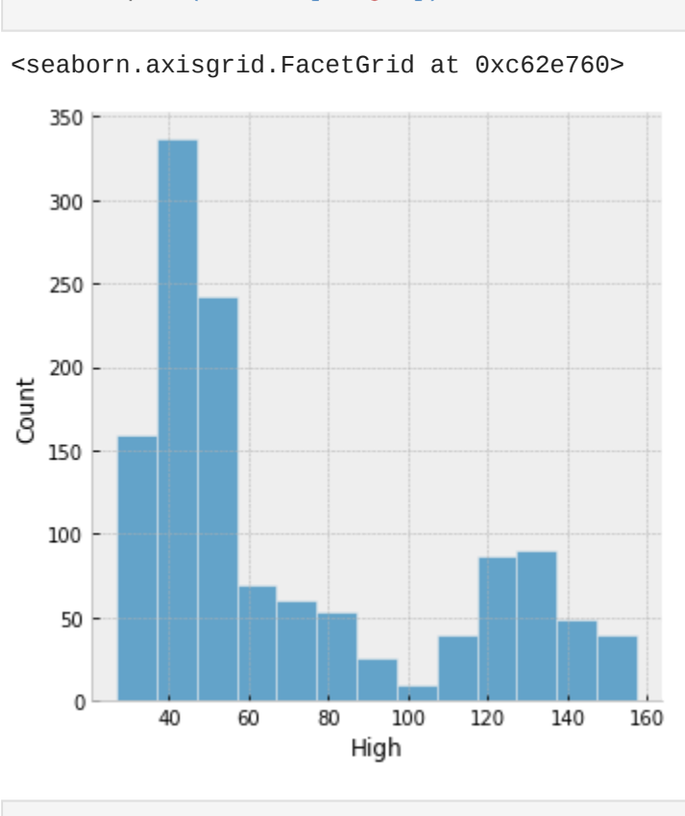
```
predictions=lr_prediction


valid=df[X.shape[0]:]
valid['Predictions']=predictions
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Days')
plt.ylabel('Close price USD ($)')
plt.plot(df['Close'])
plt.plot(valid[['Close','Predictions']])
plt.legend('Orig','Val','Pred')
plt.show()
```

In [75]:

```
sns.displot(Dataset["High"])
```

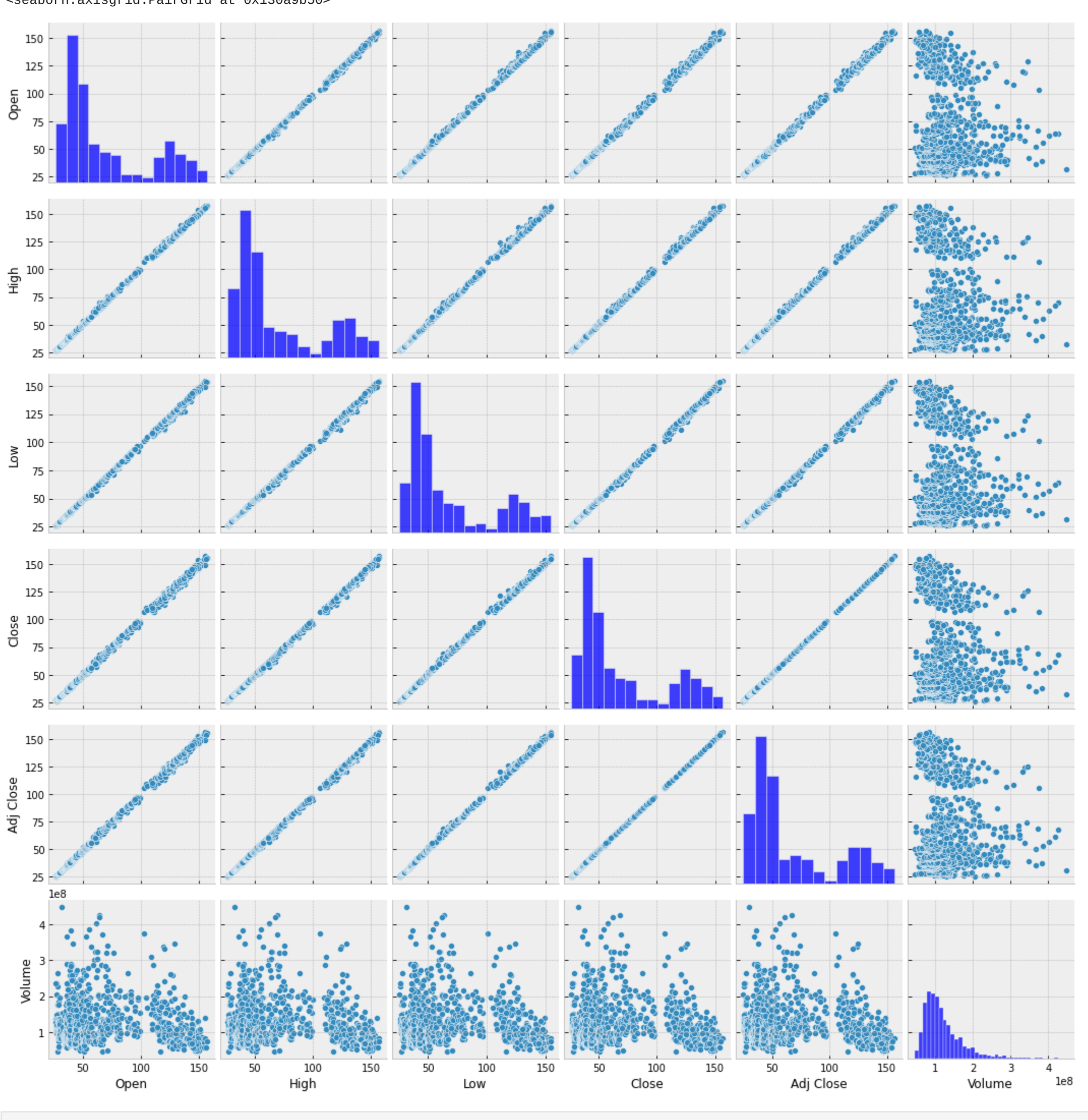
Out[75]: <seaborn.axisgrid.FacetGrid at 0xc62e760>



In [76]:

```
sns.pairplot(Dataset)
```

Out[76]: <seaborn.axisgrid.PairGrid at 0x130a9b50>



In [ ]:

In [ ]:

In [ ]:

In [ ]: