In [ ]:
```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sb
import plotly.express as px
plt.style.use('default')
```

> 1. Read dataset

In [ ]:
```python
boston = pd.read_csv("D:\\PROGRAMMING\\Datasets\\Boston.csv")
boston.head()
```

Out[ ]:

| | Unnamed: 0 | crim | zn | indus | chas | nox | rm | age | dis | rad | tax | ptratio | black | lstat | medv |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0.00632 | 18.0 | 2.31 | 0 | 0.538 | 6.575 | 65.2 | 4.0900 | 1 | 296 | 15.3 | 396.90 | 4.98 | 24.0 |
| **1** | 2 | 0.02731 | 0.0 | 7.07 | 0 | 0.469 | 6.421 | 78.9 | 4.9671 | 2 | 242 | 17.8 | 396.90 | 9.14 | 21.6 |
| **2** | 3 | 0.02729 | 0.0 | 7.07 | 0 | 0.469 | 7.185 | 61.1 | 4.9671 | 2 | 242 | 17.8 | 392.83 | 4.03 | 34.7 |
| **3** | 4 | 0.03237 | 0.0 | 2.18 | 0 | 0.458 | 6.998 | 45.8 | 6.0622 | 3 | 222 | 18.7 | 394.63 | 2.94 | 33.4 |
| **4** | 5 | 0.06905 | 0.0 | 2.18 | 0 | 0.458 | 7.147 | 54.2 | 6.0622 | 3 | 222 | 18.7 | 396.90 | 5.33 | 36.2 |

> 2. Find dependent and independent variables

In [ ]:
```python
X = pd.DataFrame(boston.iloc[:, :-1])
y = pd.DataFrame(boston.iloc[:, -1])
```

> 3. Check for the significance

In [ ]:
```python
# The inclusion of a constant allows the regression line to have an intercept point with
# the y-axis, even when all independent variables are zero.
```

In [ ]:
```python
##level of significance
alpha = 0.05

## Add constant to the independent variable
X = sm.add_constant(X)

sig_est = sm.OLS(y, X)## OLS( Ordinary Least Square )
result = sig_est.fit()
print(result.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   medv   R-squared:                       0.741
Model:                            OLS   Adj. R-squared:                  0.734
Method:                 Least Squares   F-statistic:                     100.6
Date:                Tue, 11 Jul 2023   Prob (F-statistic):         3.44e-134
Time:                        19:26:22   Log-Likelihood:                -1498.0
No. Observations:                 506   AIC:                             3026.
Df Residuals:                     491   BIC:                             3089.
Df Model:                          14
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         36.4614      5.101      7.148      0.000      26.439      46.484
Unnamed: 0    -0.0025      0.002     -1.215      0.225      -0.007       0.002
crim          -0.1088      0.033     -3.310      0.001      -0.173      -0.044
zn             0.0480      0.014      3.484      0.001       0.021       0.075
indus          0.0199      0.061      0.324      0.746      -0.101       0.141
chas           2.7052      0.861      3.141      0.002       1.013       4.398
nox          -17.5416      3.822     -4.589      0.000     -25.052     -10.031
rm             3.8392      0.418      9.175      0.000       3.017       4.661
age           -0.0019      0.013     -0.145      0.885      -0.028       0.024
dis           -1.4933      0.200     -7.471      0.000      -1.886      -1.101
rad            0.3249      0.068      4.771      0.000       0.191       0.459
tax           -0.0116      0.004     -3.046      0.002      -0.019      -0.004
ptratio       -0.9480      0.131     -7.246      0.000      -1.205      -0.691
black          0.0094      0.003      3.485      0.001       0.004       0.015
lstat         -0.5262      0.051    -10.377      0.000      -0.626      -0.427
==============================================================================
Omnibus:                      175.545   Durbin-Watson:                   1.084
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              760.925
Skew:                           1.502   Prob(JB):                    5.85e-166
Kurtosis:                       8.202   Cond. No.                     1.68e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.68e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [ ]:

```python
##Checking for the simple linear regression
check = sm.OLS(boston['medv'] , boston['age']).fit()
check.summary()
```

Out[ ]:

OLS Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | medv | **R-squared (uncentered):** | 0.644 |
| **Model:** | OLS | **Adj. R-squared (uncentered):** | 0.644 |
| **Method:** | Least Squares | **F-statistic:** | 915.1 |
| **Date:** | Tue, 11 Jul 2023 | **Prob (F-statistic):** | 1.85e-115 |
| **Time:** | 19:26:22 | **Log-Likelihood:** | -2071.5 |
| **No. Observations:** | 506 | **AIC:** | 4145. |
| **Df Residuals:** | 505 | **BIC:** | 4149. |
| **Df Model:** | 1 | | |
| **Covariance Type:** | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| **age** | 0.2636 | 0.009 | 30.250 | 0.000 | 0.246 | 0.281 |

| | | | |
|---|---|---|---|
| **Omnibus:** | 27.739 | **Durbin-Watson:** | 0.357 |
| **Prob(Omnibus):** | 0.000 | **Jarque-Bera (JB):** | 19.564 |
| **Skew:** | 0.369 | **Prob(JB):** | 5.65e-05 |
| **Kurtosis:** | 2.380 | **Cond. No.** | 1.00 |

Notes:
[1] $R^2$ is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

> 4. Dropping the necessary columns

In [ ]:
```
## unnamed:0,Indus and Age are statistically insignificant for we accept null hypothesis in this
## case and reject the alternate hypothsis, and we will drop those attributes from our model

X = X.drop(["Unnamed: 0", "indus", "age"], axis = 1)
X.head()
```

Out[ ]:

| | const | crim | zn | chas | nox | rm | dis | rad | tax | ptratio | black | lstat |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | 0.00632 | 18.0 | 0 | 0.538 | 6.575 | 4.0900 | 1 | 296 | 15.3 | 396.90 | 4.98 |
| **1** | 1.0 | 0.02731 | 0.0 | 0 | 0.469 | 6.421 | 4.9671 | 2 | 242 | 17.8 | 396.90 | 9.14 |
| **2** | 1.0 | 0.02729 | 0.0 | 0 | 0.469 | 7.185 | 4.9671 | 2 | 242 | 17.8 | 392.83 | 4.03 |
| **3** | 1.0 | 0.03237 | 0.0 | 0 | 0.458 | 6.998 | 6.0622 | 3 | 222 | 18.7 | 394.63 | 2.94 |
| **4** | 1.0 | 0.06905 | 0.0 | 0 | 0.458 | 7.147 | 6.0622 | 3 | 222 | 18.7 | 396.90 | 5.33 |

> 5. Find the coefficients | p_values | Confidence Interval

In [ ]:
```
print("\nThe Coefficiencts are : ")
result.params
```

```
The Coefficiencts are :
```

Out[ ]:
```
const          36.461352
Unnamed: 0     -0.002526
crim           -0.108762
zn              0.048031
indus           0.019932
chas            2.705245
nox           -17.541602
rm              3.839225
age            -0.001938
dis            -1.493304
rad             0.324925
tax            -0.011598
ptratio        -0.947985
black           0.009357
lstat          -0.526184
dtype: float64
```

In [ ]:
```
print("The P-Values are: ")
result.pvalues
```

```
The P-Values are:
```

Out[ ]:
```
const         3.209691e-12
Unnamed: 0    2.250457e-01
crim          1.000250e-03
zn            5.375059e-04
indus         7.458713e-01
chas          1.785946e-03
nox           5.658365e-06
rm            1.245587e-18
age           8.848664e-01
dis           3.682773e-13
rad           2.426287e-06
tax           2.443267e-03
ptratio       1.670700e-12
black         5.364596e-04
lstat         6.050328e-23
dtype: float64
```

In [ ]:
```
print("Confidence Intervals are: ")
result.conf_int()
```

```
Confidence Intervals are:
```

Out[ ]:

| | 0 | 1 |
|---|---|---|
| **const** | 26.438882 | 46.483822 |
| **Unnamed: 0** | -0.006612 | 0.001560 |
| **crim** | -0.173316 | -0.044209 |
| **zn** | 0.020946 | 0.075115 |
| **indus** | -0.100841 | 0.140705 |
| **chas** | 1.012960 | 4.397531 |
| **nox** | -25.051861 | -10.031344 |
| **rm** | 3.017106 | 4.661344 |
| **age** | -0.028227 | 0.024350 |
| **dis** | -1.886054 | -1.100554 |
| **rad** | 0.191101 | 0.458750 |
| **tax** | -0.019078 | -0.004117 |
| **ptratio** | -1.205026 | -0.690945 |
| **black** | 0.004081 | 0.014632 |
| **lstat** | -0.625808 | -0.426560 |

> 6. Train Test Split

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 100)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(404, 12)
(102, 12)
(404, 1)
(102, 1)
```

7. Training Our Model

In [ ]:
```
mlr = LinearRegression()
mlr.fit(X_train, y_train)
```

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

8. Predicting Value

In [ ]:
```
y_predict = mlr.predict(X_test)


final = pd.DataFrame({'Actual':y_test.values.flatten(), 'Predicted':y_predict.flatten()})
final
```

Out[ ]:

| | Actual | Predicted |
|---|---|---|
| **0** | 34.6 | 34.496490 |
| **1** | 31.5 | 30.868682 |
| **2** | 20.6 | 22.304769 |
| **3** | 14.5 | 18.131193 |
| **4** | 16.2 | 20.541658 |
| **...** | ... | ... |
| **97** | 50.0 | 36.370316 |
| **98** | 7.2 | 18.015547 |
| **99** | 50.0 | 23.490485 |
| **100** | 14.0 | 13.702219 |
| **101** | 11.0 | 14.314579 |

102 rows × 2 columns

In [ ]:
```
px.scatter(final, 'Actual', 'Predicted', trendline = 'ols', trendline_color_override='blue')
```

9. Necessary observations

In [ ]:
```
mae = mean_absolute_error(y_test, y_predict)
mse = mean_squared_error(y_test, y_predict)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_predict)

n = len(y_test)     ## no of samples
p = X_test.shape[1] ## no of predictors
adjusted_r2 = 1 - ( 1 - r2 )*( n - 1) / n - p - 1


print("mean_absolute_error is: ", mae)
print("mean_squared_error is : ", mse)
print("root_mean_squared_error is : ", rmse)
print("r square is  : ", r2)
print("adjusted r square is  : ", adjusted_r2)## it decreases as  features/predictors increase
```

```
mean_absolute_error is:  3.2518545636225586
mean_squared_error is :  23.425938278313655
root_mean_squared_error is :  4.840034945980623
r square is  :  0.7574812283240356
adjusted r square is  :  -12.240141136659533
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js