MACHINE LEARNING

SIMPLE LINEAR REGREEION

STEP-0 Import libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Step 1 Import Dataset

```
df = pd.read_csv("salary_data - Copy.csv")
df.head()
```

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

Step 2 Spliting dataset into training and testing data

```
X = df[["YearsExperience"]]
y = df["Salary"]
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=0)
```

Step 3 Fit Linear Regression Model

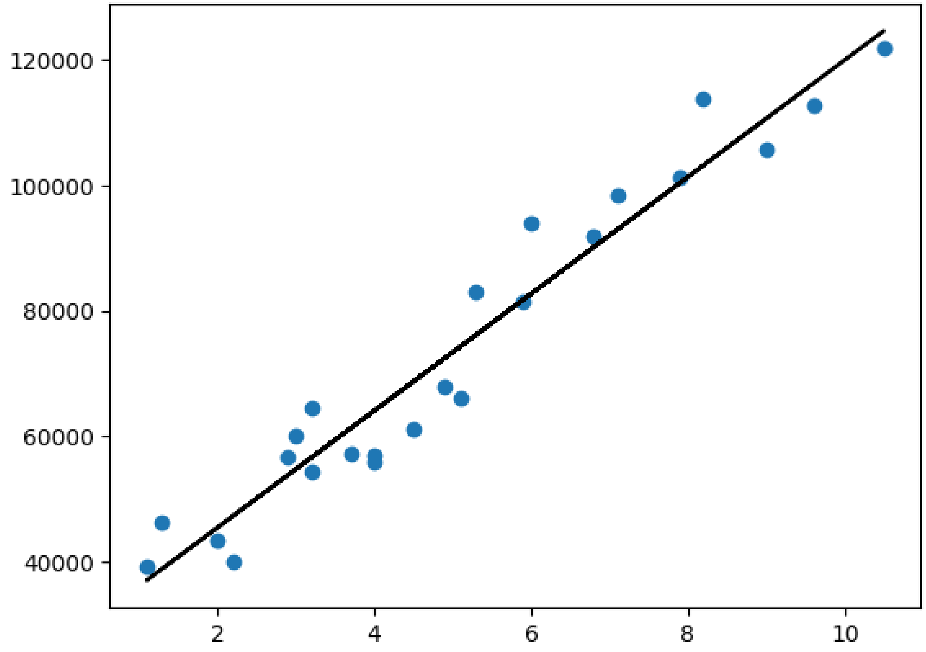
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model = model.fit(X_train, y_train)
model
```

```
▾ LinearRegression
LinearRegression()
```

Step 4 Plotting

```
import matplotlib.pyplot as plt
plt.scatter(X_train,y_train)
plt.plot(X_train.values, model.predict(X_train), color="black")
```

```
    [<matplotlib.lines.Line2D at 0x7f26286aea40>]
```



Step 5 Evaluating Model Fitness

```
# Model Fitness
print("Score for training data =" ,model.score(X_train, y_train))
print("Score for test data =" ,model.score(X_test, y_test))
```

```
    Score for training data = 0.9411949620562126
    Score for test data = 0.988169515729126
```

Step 6 Prediction of unknown values

```
model.predict([[20],[25],[30]])
```

```
    /usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names
      warnings.warn(
    array([213031.60168521, 259594.47731886, 306157.3529525 ])
```

Step 7 Measure the Accuracy

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error

# Generate sample data
x = np.array([1, 2, 3, 4, 5])
y = np.array([2, 4, 5, 4, 5])

# Fit the linear regression model
# Fit the linear regression model
regression_model = LinearRegression()
regression_model.fit(x.reshape(-1, 1), y)

# Predict the target values
y_pred = regression_model.predict(x.reshape(-1, 1))

# Calculate R-squared
r2 = r2_score(y, y_pred)
print("R-squared:", r2)

# Calculate RMSE
rmse = np.sqrt(mean_squared_error(y, y_pred))
print("RMSE:", rmse)
```

```
R-squared: 0.6000000000000001
RMSE: 0.6928203230275508
```

✓ 0s completed at 5:48 AM ● ×