# MULTIPLE LINEAR REGRESSION

## STEP-1

```
import pandas as pd
df = pd.read_csv("ml_data_salary.csv")
df.head()
```

| | age | distance | YearsExperience | Salary |
|---|---|---|---|---|
| 0 | 31.1 | 77.75 | 1.1 | 39343 |
| 1 | 31.3 | 78.25 | 1.3 | 46205 |
| 2 | 31.5 | 78.75 | 1.5 | 37731 |
| 3 | 32.0 | 80.00 | 2.0 | 43525 |
| 4 | 32.2 | 80.50 | 2.2 | 39891 |

```
X = df[["age","distance","YearsExperience"]]
y=df["Salary"]
```

## STEP-3 Fit linear regression model

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model = model.fit(X,y)
model
```

```
LinearRegression()
```

## STEP-4 Evaluating model fitness

```
# model fittness
print("Score for data =" , model.score(X,y))
```

```
Score for data = 0.9569960750337954
```

## STEP-5 PREDICTION OF UNKNOWN VALUES

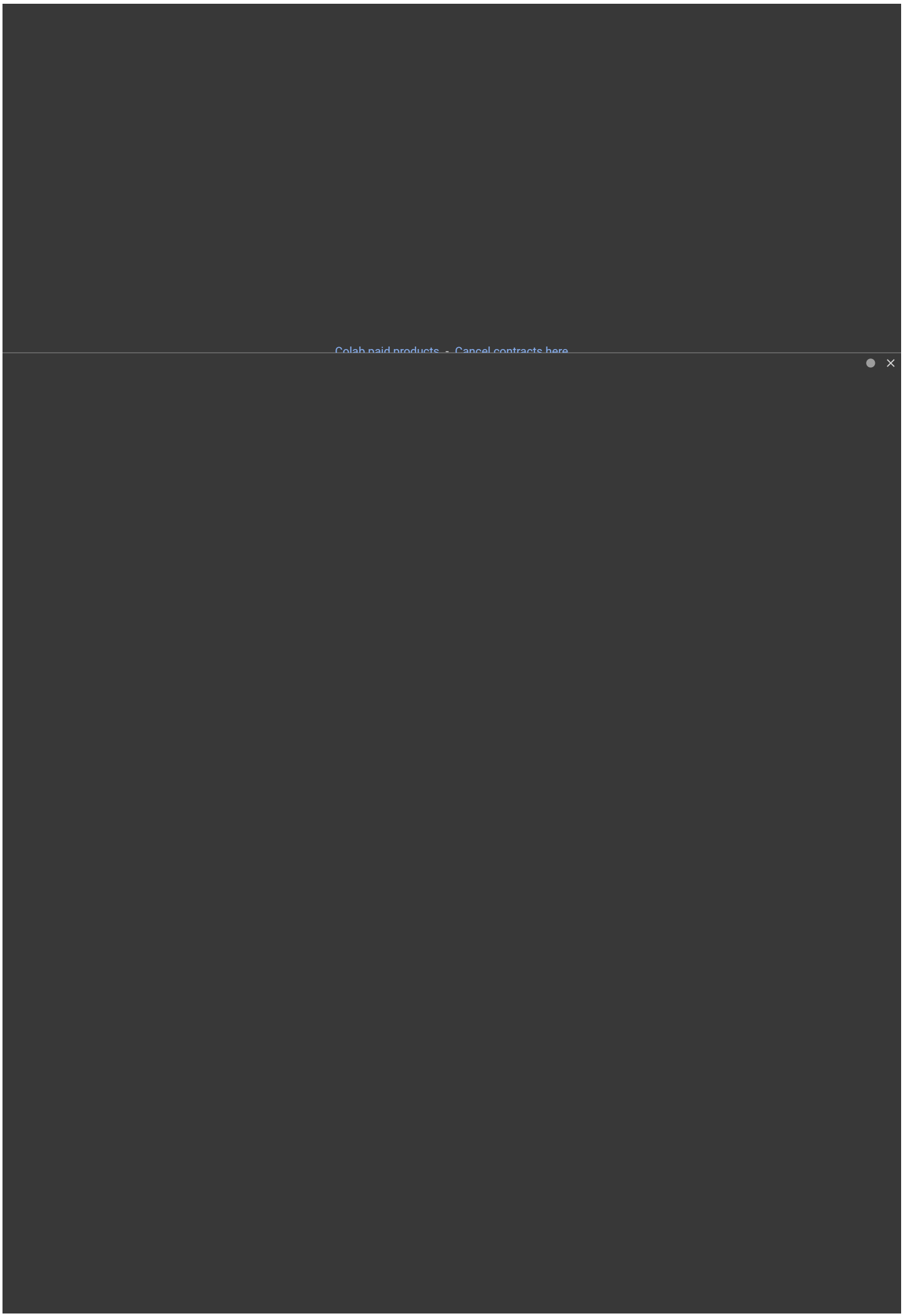
```
model.predict([[31.1,77.75,1.1]])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LinearRegression
  warnings.warn(
array([36209.375])
```

## STEP-6 TO CHECK THE ACCURACY SCORE AND SPLIT DATA IN 80/20 RATIO

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score

model = LinearRegression()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

accuracy = r2_score(y_test, y_pred)
print("Accuracy score: {:.2f}".format(accuracy))
```

```
Accuracy score: 0.99
```


Colab paid products - Cancel contracts here

# DECISION TREE CLASSIFIER

## STEP-1 IMPORT DATA

```
import pandas as pd
df = pd.read_csv("mldata1.csv")
df.head()
```

| | age | height | weight | gender | likeness |
|---|---|---|---|---|---|
| 0 | 27 | 170.688 | 76.0 | Male | Biryani |
| 1 | 41 | 165 | 70.0 | Male | Biryani |
| 2 | 29 | 171 | 80.0 | Male | Biryani |
| 3 | 27 | 173 | 102.0 | Male | Biryani |
| 4 | 29 | 164 | 67.0 | Male | Biryani |

## Step-2 Making input and Output Variable

```
df["gender"] = df["gender"].replace("Male",1)
df["gender"] = df["gender"].replace("Female",0)
```

```
X = df[["weight","gender"]]
y = df["likeness"]
```

## Step-3 Making Machine Learning Model

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier().fit(X,y)
model.predict([[50,1]])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but DecisionTreeClas
  warnings.warn(
array(['Samosa'], dtype=object)
```

## Step-4 Checking machine learning model performance

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)
model = DecisionTreeClassifier().fit(X_train,y_train)
predicted_values = model.predict(X_test)
predicted_values
```

```
array(['Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Pakora',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Pakora', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Pakora', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Pakora', 'Biryani', 'Biryani',
       'Samosa'], dtype=object)
```

```
score = accuracy_score(y_test, predicted_values)
score
```

```
0.6122448979591837
```

## Step-5 Making Visualization

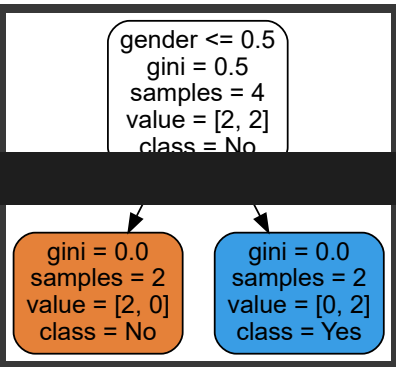
```
from sklearn import tree
model = DecisionTreeClassifier().fit(X,y)
```

```
tree.export_graphviz(model,out_file= "foodie.dot",
feature_names=["age","gender"],
class_names=sorted(y.unique()),
label="all",rounded=True,filled=True)
```

```
from sklearn import tree
from sklearn.datasets import load_iris
import graphviz

# Create a sample dataset (replace this with your own data)
X = [[30, 0], [25, 1], [35, 1], [40, 0]]
y = ['No', 'Yes', 'Yes', 'No']

# Train the decision tree model
model = tree.DecisionTreeClassifier()
model.fit(X, y)

# Export the decision tree as a DOT file
dot_data = tree.export_graphviz(
    model,
    out_file=None,
    feature_names=["age", "gender"],
    class_names=sorted(set(y)),
    label="all",
    rounded=True,
    filled=True
)

# Save the DOT file
with open("foodie.dot", "w") as f:
    f.write(dot_data)

# Convert the DOT file to a visual representation (e.g., PDF, PNG, or SVG)
graph = graphviz.Source(dot_data)
graph.render(filename="foodie", format="pdf")
```

```
'foodie.pdf'
```

```
import graphviz
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier

# Create a sample dataset
X = [[30, 0], [25, 1], [35, 1], [40, 0]]
y = ['No', 'Yes', 'Yes', 'No']

# Train the decision tree model
model = DecisionTreeClassifier()
model.fit(X, y)

# Export the decision tree as a DOT file
dot_data = tree.export_graphviz(
    model,
    out_file=None,
    feature_names=["age", "gender"],
    class_names=sorted(set(y)),
    label="all",
    rounded=True,
    filled=True
)

# Save the DOT file
with open("foodie.dot", "w") as f:
    f.write(dot_data)

# Convert the DOT file to a visual representation
graph = graphviz.Source(dot_data)
graph.render("foodie", format="png")

# Display the decision tree
graph
```






✓ 0s completed at 19:09

# MACHINE LEARNING

## 1-SIMPLE LINEAR REGRESSION

```
pip install scikit-learn
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.22.4
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.10.1
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.2.0
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn)
```

+ Code + Text

## STEP-1 IMPORT LIBRARIES

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

## STEP-2 IMPORT DATA

```
df = pd.read_csv('salary_data.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-3 SELECTING INPUT AND OUTPUT VARIABLES

```
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-4 MAKING LINEAR REGRESSION MODEL

```
from sklearn.linear_model import LinearRegression
model= LinearRegression()
```

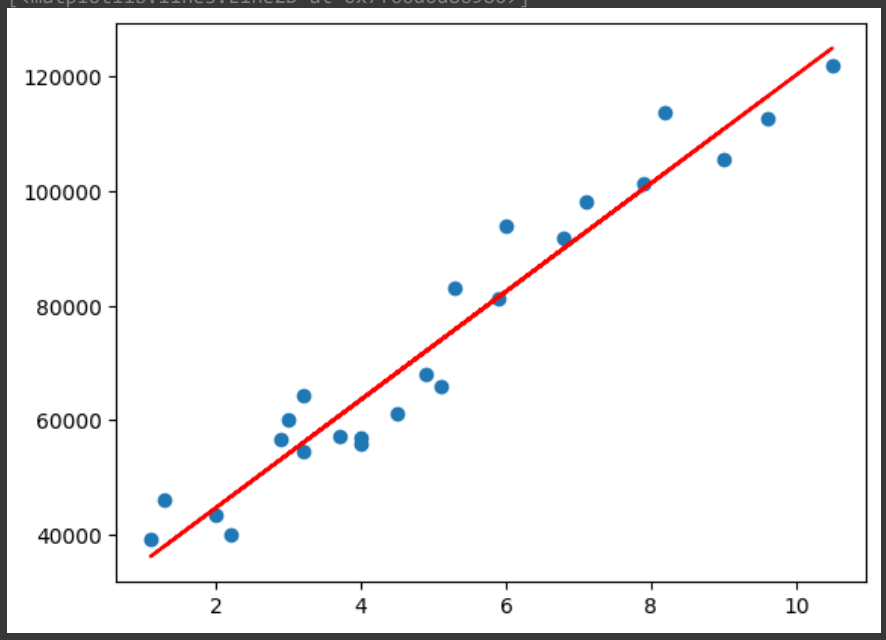
## STEP-5 FITTING THE MODEL

```
model = model.fit(X,y)
model
```

## STEP-6 PLOTTING

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

```
[<matplotlib.lines.Line2D at 0x7f66d0d86980>]
```




## STEP-7 PREDICTING THE MODEL

```
model.predict([[10]])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but
  warnings.warn(
array([120291.82341322])
```

## STEP-8 EVALUATING THE MODEL

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score

model = LinearRegression()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

accuracy = r2_score(y_test, y_pred)
print("Accuracy score: {:.2f}".format(accuracy))
```

```
Accuracy score: 0.99
```

## STEP-9 SPLIT AND COMPUTE ACCURACY 80/20 AND AVERAGE

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

```
# Splitting the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Creating and fitting the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predicting on the test set
y_pred = model.predict(X_test)

# Computing the accuracy (R-squared score)
accuracy = r2_score(y_test, y_pred)

# Printing the accuracy
print("Accuracy:", accuracy)
```

```
Accuracy: 0.988169515729126
```



✓ 0s completed at 17:58