Decision Tree Classifiers

In [74]:
```
# import the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [75]:
```
# import the data set
df = pd.read_csv("bayani.csv")
df.head()
```

Out[75]:

	age	height	weight	gender	likeness
0	27	170.688	76.0	Male	Biryani
1	41	165.000	70.0	Male	Biryani
2	29	171.000	80.0	Male	Biryani
3	27	173.000	102.0	Male	Biryani
4	29	164.000	67.0	Male	Biryani

In [76]:
```
# Changing the data from string to numeric in gender
df["gender"]=df ['gender'].replace("Male",1)
df["gender"]=df ['gender'].replace("Female",0)
df.head()
```

Out[76]:

	age	height	weight	gender	likeness
0	27	170.688	76.0	1	Biryani
1	41	165.000	70.0	1	Biryani
2	29	171.000	80.0	1	Biryani
3	27	173.000	102.0	1	Biryani
4	29	164.000	67.0	1	Biryani

Selection of the input and Output variables

In [77]:
```
x=df [['weight','gender']]
y=df ['likeness']

x.head()
```

Out[77]:

	weight	gender
0	76.0	1

	weight	gender
1	70.0	1
2	80.0	1
3	102.0	1
4	67.0	1

In [78]:
```
y.head()
```

Out[78]:
```
0    Biryani
1    Biryani
2    Biryani
3    Biryani
4    Biryani
Name: likeness, dtype: object
```

In [80]:
```
# import the library deciesion tree and train the model
from sklearn.tree import DecisionTreeClassifier
# graph
from sklearn import tree
#create the model and fit it
model = DecisionTreeClassifier().fit(x,y)
model.predict([[80,1]])
```

```
C:\Users\Asfandyar\AppData\Roaming\Python\Python310\site-packages\sklearn\base.py:450: U
serWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted w
ith feature names
  warnings.warn(
```

Out[80]:
```
array(['Biryani'], dtype=object)
```

Let run the Graph of the Decision Tree

In [82]:
```
tree.export_graphviz(model,
                     out_file="foodies.dot",
                     feature_names=['age','gender'],
                     class_names=sorted(y.unique()),
                     label='all',
                     rounded=True,
                     filled=True)
```

In [19]:
```
# how to measure the accuracy  of out model
# split the data into test and train (80/20)
from sklearn.model_selection import  train_test_split
from sklearn.metrics import accuracy_score
# Split the weights
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2 ,random_state=0)
# create the model and train on the x_train and y_train
model1= DecisionTreeClassifier().fit (x_train,y_train)
# NOw do prediction of the x_test and then we have to compare the actual result of the
# prediction
```

```
predicted=model1.predict(x_test)
print ("Prediction of test data are as below",predicted)
```

```
Prediction of test data are as below ['Biryani' 'Biryani' 'Pakora' 'Biryani' 'Samosa' 'B
iryani' 'Pakora'
 'Biryani' 'Biryani' 'Biryani' 'Samosa' 'Samosa' 'Samosa' 'Pakora'
 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Pakora' 'Biryani'
 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Samosa'
 'Biryani' 'Samosa' 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani'
 'Biryani' 'Biryani' 'Biryani' 'Samosa' 'Biryani' 'Biryani' 'Biryani'
 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani' 'Biryani']
```

Now Compare the prediction and the actual values

In [24]:
```
score = accuracy_score(y_test , predicted)
score
```

Out[24]:
```
0.6122448979591837
```

FInd the accuracy of the Simple linear Regression

Step 1: Import the DataSet

In [58]:
```
# import the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score
```

In [59]:
```
import pandas as pd
df = pd.read_csv("ml_data_salary.csv")
df.head()
```

Out[59]:

	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

Step 2: Splitting Dataset into Training Data and testing Data

In [60]:
```
x= df [["YearsExperience"]]
y= df ["Salary"]
```

In [61]:
```
# import library
from sklearn.linear_model import  LinearRegression
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)
```

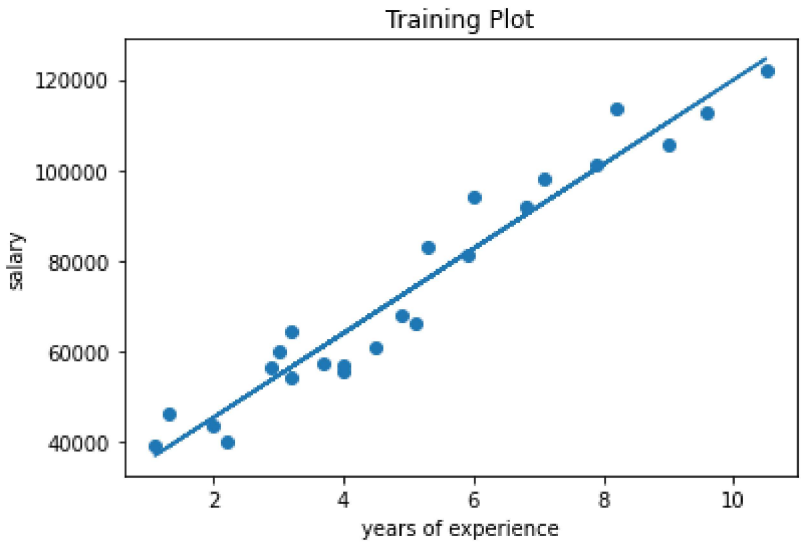
Step-3 Fit the Model

In [62]:
```
from sklearn.linear_model import  LinearRegression
model=LinearRegression().fit(x_train,y_train)
model
```

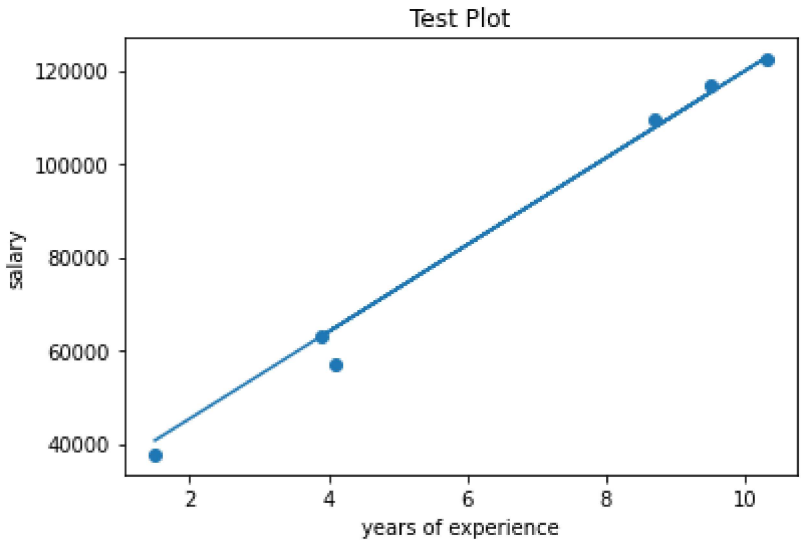
Out[62]:
```
LinearRegression()
```

Step-4 Ploting

In [49]:
```
import matplotlib.pyplot as pt
pt.scatter(x_train,y_train)
# make plot with prediction
pt.plot(x_train,model.predict(x_train))
# Labeling the plot
pt.xlabel("years of experience")
pt.ylabel("salary")
pt.title("Training Plot")
pt.show()
```



In [50]:
```
pt.scatter(x_test,y_test)
# make plot with prediction
pt.plot(x_test,model.predict(x_test))
# Labeling the plot
pt.xlabel("years of experience")
pt.ylabel("salary")
pt.title("Test Plot")
pt.show()
```



Step-5 Evaluating

In [52]:
```
from sklearn.metrics import accuracy_score
print ("Score of train model",model.score(x_test,y_test))
print ("Score of test model",model.score(x_train,y_train))
# predicted_R=model.predict(x_test)
# score = accuracy_score(y_test , predicted_R)
# print ("Score of test model",score)
```

```
Score of train model 0.988169515729126
Score of test model 0.9411949620562126
```

In [54]:
```
predicted_R=model.predict(x_test)
predicted_R
```

Out[54]:
```
array([ 40748.96184072, 122699.62295594,  64961.65717022,  63099.14214487,
       115249.56285456, 107799.50275317])
```

In [ ]:
```
```

In [57]:
```
score1 = accuracy_score(y_test , predicted_R)
score1
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [57], in <module>
----> 1 score1 = accuracy_score(y_test , predicted_R)
      2 score1

File ~\AppData\Roaming\Python\Python310\site-packages\sklearn\metrics\_classification.py:211, in accuracy_score(y_true, y_pred, normalize, sample_weight)
    145 """Accuracy classification score.
    146
    147 In multilabel classification, this function computes subset accuracy:
   (...)
    207 0.5
    208 """
```

```
    210 # Compute accuracy for each possible representation
--> 211 y_type, y_true, y_pred = _check_targets(y_true, y_pred)
    212 check_consistent_length(y_true, y_pred, sample_weight)
    213 if y_type.startswith("multilabel"):

File ~\AppData\Roaming\Python\Python310\site-packages\sklearn\metrics\_classification.py:93, in _check_targets(y_true, y_pred)
     90     y_type = {"multiclass"}
     92 if len(y_type) > 1:
---> 93     raise ValueError(
     94         "Classification metrics can't handle a mix of {0} and {1} targets".format(
     95             type_true, type_pred
     96         )
     97     )
     99 # We can't have more than one value on y_type => The set is no more needed
    100 y_type = y_type.pop()

ValueError: Classification metrics can't handle a mix of multiclass and continuous targets
```

Why the accuracy_score is not working

> for classification we use metric structure and for Simple Linear Regression we use RMSE and MSE \
>
> ### Function used
>
> sklearn.metrics.mean_squared_error(y_true, y_pred, *, sample_weight=None, multioutput='uniform_average', squared=True)\ squared = If **True** returns MSE value, if False returns RMSE value.

Finding MSE

In [66]:
```
# import library
from sklearn.metrics import mean_squared_error
#find MSE
mse=mean_squared_error(y_test, predicted_R, squared=False)
print ("MSE for the model is :",mse)
```

```
MSE for the model is : 3580.979237321343
```

In [68]:
```
print (y_test)
```

```
2      37731
28    122391
13     57081
10     63218
26    116969
24    109431
Name: Salary, dtype: int64
```

In [69]:
```
print (predicted_R)
```

```
[ 40748.96184072 122699.62295594  64961.65717022  63099.14214487
 115249.56285456 107799.50275317]
```

Find out RMSE

In [70]:
```
RMSE=mean_squared_error(y_test, predicted_R, squared=True)
print ("RMSE for the model is :",RMSE)
```

```
RMSE for the model is : 12823412.298126549
```

Train and Save the model

In [90]:
```
# import library
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LinearRegression

import joblib
# fit the model
models= DecisionTreeClassifier()
joblib.dump(model,"foodie.joblib")
```

Out[90]:
```
['foodie.joblib']
```

Take the data and load it

In [84]:
```
# import the data
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
# loading the data
dfs=pd.read_csv("bayani.csv")
dfs.head()
```

Out[84]:

	age	height	weight	gender	likeness
0	27	170.688	76.0	Male	Biryani
1	41	165.000	70.0	Male	Biryani
2	29	171.000	80.0	Male	Biryani
3	27	173.000	102.0	Male	Biryani
4	29	164.000	67.0	Male	Biryani

In [85]:
```
# change the male to 1 and female
dfs['gender']=dfs['gender'].replace('Male',1)
dfs['gender']=dfs['gender'].replace('Female',0)

dfs.head()
```

Out[85]:

	age	height	weight	gender	likeness
0	27	170.688	76.0	1	Biryani
1	41	165.000	70.0	1	Biryani
2	29	171.000	80.0	1	Biryani
3	27	173.000	102.0	1	Biryani
4	29	164.000	67.0	1	Biryani

In [91]:
```
# input and outputs!!
xx=dfs [['weight','gender']]
yy=dfs ['likeness']
```

Load the Model which is Saved

In [92]:
```
modelss=joblib.load("foodie.joblib")
```

Now pass the Data

In [94]:
```
#assign the model that is fit to xx,yy in mods
x_train1,x_test1,y_train1,y_test1=train_test_split(xx,yy,test_size=0.2 ,random_state=0)
mods=modelss.fit(x_train1,y_train1)
# now prediction on the test and than accuracy
pred=mods.predict(x_test1)
pred
```

Out[94]:
```
array(['Biryani', 'Biryani', 'Pakora', 'Biryani', 'Samosa', 'Biryani',
       'Pakora', 'Biryani', 'Biryani', 'Biryani', 'Samosa', 'Samosa',
       'Samosa', 'Pakora', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Pakora', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Samosa', 'Biryani', 'Samosa',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Samosa', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani'], dtype=object)
```

Finally we found Accuracy from Loading the Model !!

In [95]:
```
from sklearn.metrics import accuracy_score
score = accuracy_score(y_test1 , pred)
score
```

Out[95]:
```
0.6122448979591837
```