```
import numpy as nm
import matplotlib.pyplot as pt
import pandas as pd
import seaborn as sns
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics

import matplotlib.pyplot as plt
%matplotlib inline

data = pd.read_csv("/content/tinder.csv")
data.head()
```

	Srno	Texting	Outings	Common Intrest	Like
0	1	2	2	1	Didntlike
1	2	3	15	5	LargeDoses
2	3	3	7	2	SmallDoses
3	4	3	4	3	SmallDoses
4	5	1	1	0	Didntlike

```
data.describe()
data.isnull().sum()
```

```
Srno              0
Texting           0
Outings           0
Common Intrest    0
Like              0
dtype: int64
```

```
dataset_n = data.drop(["Srno"],axis=1)
dataset_n.head()
```

	Texting	Outings	Common Intrest	Like
0	2	2	1	Didntlike
1	3	15	5	LargeDoses
2	3	7	2	SmallDoses
3	3	4	3	SmallDoses
4	1	1	0	Didntlike

```
x = dataset_n.drop("Like",axis=1)
x.head()
```

	Texting	Outings	Common Intrest
0	2	2	1
1	3	15	5
2	3	7	2
3	3	4	3
4	1	1	0

```
y = pd.DataFrame(dataset_n.iloc[:,-1].values)
y.head()
```

	0
0	Didntlike
1	LargeDoses
2	SmallDoses
3	SmallDoses
4	Didntlike

```
feature_cols = [ 'Texting','Outings','Common Intrest']
X = data[feature_cols]
y = data.Like
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

```
X_train.shape
```

```
(40, 3)
```

```
X_test.shape
```

```
(10, 3)
```

```
y_train.head()
```

	0
32	LargeDoses
39	SmallDoses
21	LargeDoses
36	Didntlike

Fitting Logistic Regression Model

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:993: DataConversic
  y = column_or_1d(y, warn=True)
LogisticRegression(random_state=0)
```

```
prediction = classifier.predict(X_test)
prediction
```

```
array(['SmallDoses', 'LargeDoses', 'Didntlike ', 'SmallDoses',
       'LargeDoses', 'SmallDoses', 'SmallDoses', 'Didntlike ',
       'LargeDoses', 'SmallDoses'], dtype=object)
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, prediction)
cm
```

```
array([[2, 0, 0],
       [0, 2, 1],
       [0, 1, 4]])
```

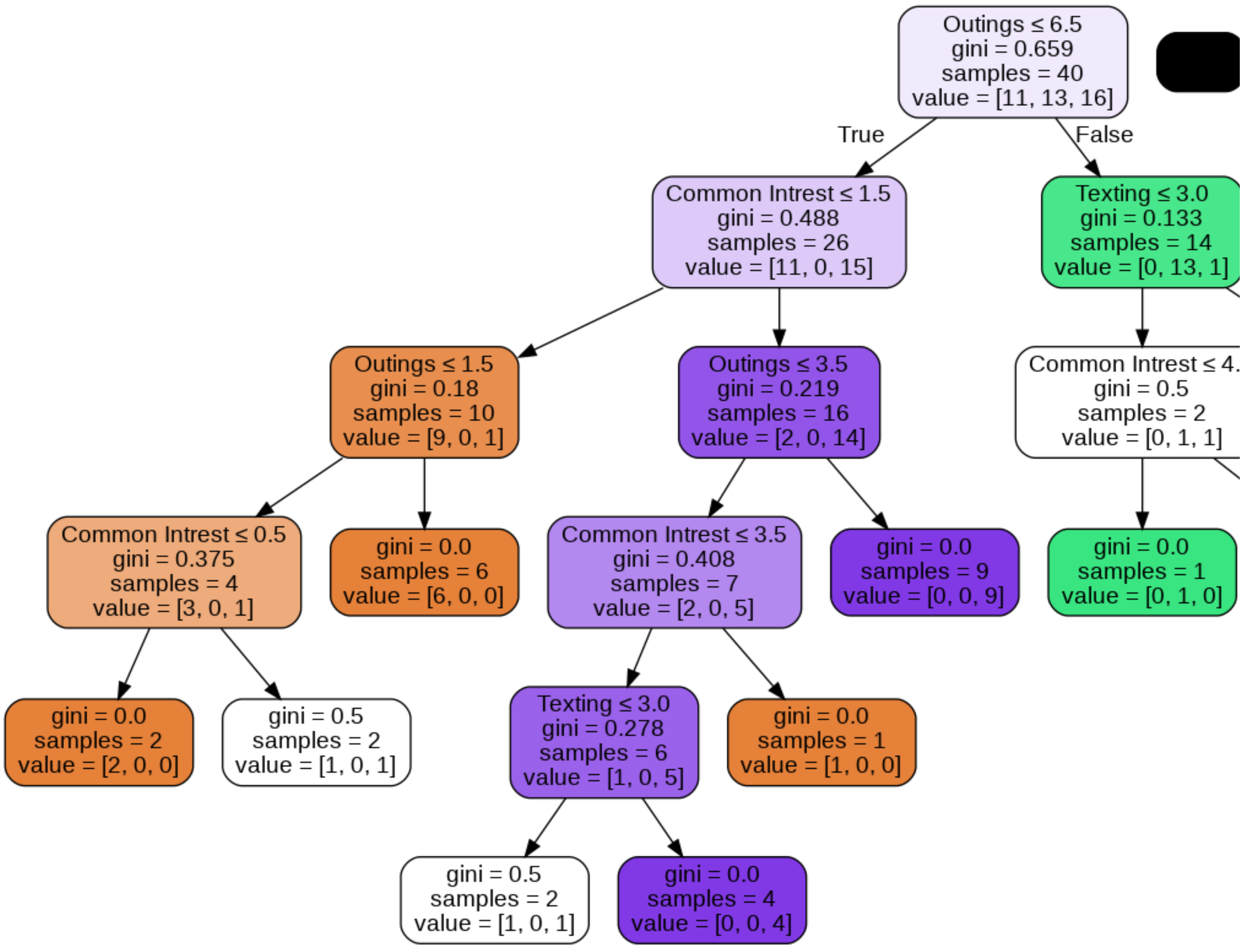
```
from sklearn.metrics import accuracy_score
acc=accuracy_score(y_test, prediction)
acc
```

```
0.8
```

```
from sklearn.tree import export_graphviz
from io import StringIO
from IPython.display import Image
import pydotplus

dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,feature_names = feature_cols)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
```

```
graph.write_png('Tinder Matches.png')
Image(graph.create_png())
```



Loading and predicting Test dataset !!

```
test_data = pd.read_csv('/content/tinder.csv')
```

```
test_data.head()
```

	Srno	Texting	Outings	Common Intrest	Like
0	1	2	2	1	Didntlike
1	2	3	15	5	LargeDoses
2	3	3	7	2	SmallDoses

```
test_data.isnull().sum()
```

```
Srno              0
Texting           0
Outings           0
Common Intrest    0
Like              0
dtype: int64
```

```
test_data = test_data.drop(["Srno"],axis=1)
test_data.head()
```

	Texting	Outings	Common Intrest	Like
0	2	2	1	Didntlike
1	3	15	5	LargeDoses
2	3	7	2	SmallDoses
3	3	4	3	SmallDoses
4	1	1	0	Didntlike

```
 from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
test_data['Like']= label_encoder.fit_transform(test_data['Like'])
print(test_data.head())
```

```
   Texting  Outings  Common Intrest  Like
0        2        2               1     0
1        3       15               5     1
2        3        7               2     2
3        3        4               3     2
4        1        1               0     0
```

```
from sklearn.preprocessing import OneHotEncoder
ohe = OneHotEncoder()
t_d = pd.DataFrame(ohe.fit_transform(x[["Texting"]]).toarray())
testdata_new=pd.concat([x,t_d],axis=1)
testdata_new
```

```
    Texting  Outings  Common Intrest    0    1    2    3
 0        2        2               1  0.0  0.0  1.0  0.0
 1        3       15               5  0.0  0.0  0.0  1.0
 2        3        7               2  0.0  0.0  0.0  1.0
 3        3        4               3  0.0  0.0  0.0  1.0
 4        1        1               0  0.0  1.0  0.0  0.0
 5        3        5               2  0.0  0.0  0.0  1.0
 6        1        1               3  0.0  1.0  0.0  0.0
 7        3       14               6  0.0  0.0  0.0  1.0
 8        2        5               4  0.0  0.0  1.0  0.0
 9        2        8               4  0.0  0.0  1.0  0.0
10        2        7               5  0.0  0.0  1.0  0.0
11        3        6               2  0.0  0.0  0.0  1.0
12        3        9               5  0.0  0.0  0.0  1.0
13        2        4               2  0.0  0.0  1.0  0.0
14        0        2               0  1.0  0.0  0.0  0.0
15        2        3               2  0.0  0.0  1.0  0.0
16        2        1               1  0.0  0.0  1.0  0.0
17        2        5               3  0.0  0.0  1.0  0.0
18        2        2               0  0.0  0.0  1.0  0.0
19        3        8               4  0.0  0.0  0.0  1.0
20        3        7               5  0.0  0.0  0.0  1.0
21        2        9               6  0.0  0.0  1.0  0.0
22        1        2               2  0.0  1.0  0.0  0.0
23        2        1               1  0.0  0.0  1.0  0.0
24        3        5               3  0.0  0.0  0.0  1.0
25        3        5               2  0.0  0.0  0.0  1.0
26        2        2               3  0.0  0.0  1.0  0.0
27        2        2               2  0.0  0.0  1.0  0.0
```

```
test = testdata_new.drop("Texting",axis=1)
test.head()
```

	Outings	Common Intrest	0	1	2	3
0	2	1	0.0	0.0	1.0	0.0
1	15	5	0.0	0.0	0.0	1.0
2	7	2	0.0	0.0	0.0	1.0

```
test = test_data.drop(["Like"],axis=1)
test.head()
```

	Texting	Outings	Common Intrest
0	2	2	1
1	3	15	5
2	3	7	2
3	3	4	3
4	1	1	0

```
45        2        8                4   0.0   0.0   1.0   0.0
```

```
prediction_new = classifier.predict(test)
prediction_new.shape
```

```
(50,)
```

```
res = pd.DataFrame(prediction_new)
res.head()
```

	0
0	Didntlike
1	LargeDoses
2	SmallDoses
3	SmallDoses
4	Didntlike

```
res.columns = ["prediction"]
```