```
1.import pandas as pd
 import numpy as np
 data = pd.read_csv(r"C:\Users\HPR\Desktop\ML Syllabus\2.csv")
 data
concepts = np.array(data)[:,:-1]
concepts
target = np.array(data)[:,-1]
target
def train(con, tar):
 for i, val in enumerate(tar):
 if val == 'yes':
 specific_h = con[i].copy()
 break
 for i, val in enumerate(con):
 if tar[i] == 'yes':
 for x in range(len(specific_h)):
 if val[x] != specific_h[x]:
 specific_h[x] = '?'
 else:
 pass
 return specific_h
 print(train(concepts, target))
2.import pandas as pd
 import numpy as np
 data = pd.read_csv(r"C:\Users\HPR\Desktop\ML Syllabus\2.csv")
 concepts = np.array(data.iloc[:,0:-1])
 target = np.array(data.iloc[:,-1])
def learn(concepts, target):
specific_h = concepts[0].copy()
print("initialization of specific_h \n",specific_h)
general_h = [["?" for i in range(len(specific_h))] for i in range(len(specific_h))]
```

```
print("initialization of general_h \n", general_h)
for i, h in enumerate(concepts):
 if target[i] == "yes":
 print("If instance is Positive ")
 for x in range(len(specific_h)):
if h[x]!= specific_h[x]:
specific_h[x] ='?'
general_h[x][x] ='?'
 if target[i] == "no":
print("If instance is Negative ")
 for x in range(len(specific_h)):
if h[x]!= specific_h[x]:
general_h[x][x] = specific_h[x]
else:
general_h[x][x] = '?'
print(" step {}".format(i+1))
 print(specific_h)
print(general_h)
 print("\n")
 print("\n")
 indices = [i for i, val in enumerate(general_h) if val == ['?', '?', '?', '?', '?', '?']]
for i in indices:
general_h.remove(['?', '?', '?', '?', '?', '?'])
return specific_h, general_h
s_final, g_final = learn(concepts, target)
 print("Final Specific_h:", s_final, sep="\n")
 print("Final General_h:", g_final, sep="\n")
```

3.
```
import numpy as np
 import pandas as pd
 from sklearn import metrics
 df=pd.read_csv(r"C:\Users\HPR\Desktop\ML Syllabus\Play Tennis.csv")
 value=['Outlook','Temprature','Humidity','Wind']
 df
len(df)
df.shape
df.head()
df.tail()
df.describe()
from sklearn import preprocessing
 string_to_int= preprocessing.LabelEncoder()
df=df.apply(string_to_int.fit_transform)
df
feature_cols = ['Outlook','Temprature','Humidity','Wind']
 X = df[feature_cols ]
y = df.Play_Tennis
from sklearn.model_selection
import train_test_split
 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
from sklearn.tree import DecisionTreeClassifier
classifier =DecisionTreeClassifier(criterion="entropy", random_state=100)
classifier.fit(X_train, y_train)
y_pred= classifier.predict(X_test)
from sklearn.metrics import accuracy_score
 print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
data_p=pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
data_p
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

5. import numpy as np

```
 import pandas as pd
 from sklearn import metrics
df=pd.read_csv(r"C:\Users\HPR\Desktop\ML Syllabus\Play_Tennis_reg.csv")
 len(df)
df.shape
x = df.drop("Golf Players", axis=1)
y = df['Golf Players']
x
y
from sklearn.preprocessing import LabelEncoder
from sklearn import preprocessing
 string_to_int= preprocessing.LabelEncoder()
X=X.apply(string_to_int.fit_transform)
X
from sklearn.tree import DecisionTreeRegressor
 reg = DecisionTreeRegressor()
 reg = reg.fit(X, y)
 y_pred = reg.predict([[2,1,0,1]])
print("Result is: ", y_pred)
y_pred = reg.predict([[2,1,0,0]])
print("Result is: ", y_pred)
y_pred = reg.predict([[1,2,0,0]])
print("Result is: ", y_pred)
```