```
import csv
num_attributes = 6
a = []

with open('EnjoySport.csv', 'r') as csvfile:
    reader = csv.reader(csvfile)
    for row in reader:
        a.append(row)

print("\n Given Training Data: \n")
print(a)

hypothesis = ['0'] * num_attributes
print("\n The initial hypothesis: ", hypothesis)

for j in range(0,num_attributes):
    hypothesis[j] = a[0][j];
    print (hypothesis)

print("\n Find S: Finding a Maximally Specific Hypothesis: \n")

for i in range(0,len(a)):
    if a[i][num_attributes] == 'yes':
        for j in range(0,num_attributes):
```

```
print("\n Find S: Finding a Maximally Specific Hypothesis: \n")

for i in range(0,len(a)):
    if a[i][num_attributes] == 'yes':
        for j in range(0,num_attributes):
            if a[i][j]!=hypothesis[j]:
                hypothesis[j]='?'
    print("For Training instance No.",i,", the Maximally Specific Hypothesis is ",hypothesis)

print("\n The Maximally Specific Hypothesis for the given given set of training examples is : \n")

print (hypothesis)
```

```
 Given Training Data:

[['sunny', 'warm', 'normal', 'strong', 'warm', 'same', 'yes'], ['sunny', 'warm', 'high', 'strong', 'warm', 'same', 'yes'], ['ra
iny', 'cold', 'high', 'strong', 'warm', 'change', 'no'], ['sunny', 'warm', 'high', 'strong', 'cool', 'change', 'yes']]

 The initial hypothesis:  ['0', '0', '0', '0', '0', '0']
['sunny', '0', '0', '0', '0', '0']
['sunny', 'warm', '0', '0', '0', '0']
['sunny', 'warm', 'normal', '0', '0', '0']
['sunny', 'warm', 'normal', 'strong', '0', '0']
['sunny', 'warm', 'normal', 'strong', 'warm', '0']
['sunny', 'warm', 'normal', 'strong', 'warm', 'same']

 Find S: Finding a Maximally Specific Hypothesis:

For Training instance No. 0 , the Maximally Specific Hypothesis is  ['sunny', 'warm', 'normal', 'strong', 'warm', 'same']
For Training instance No. 1 , the Maximally Specific Hypothesis is  ['sunny', 'warm', '?', 'strong', 'warm', 'same']
For Training instance No. 2 , the Maximally Specific Hypothesis is  ['sunny', 'warm', '?', 'strong', 'warm', 'same']
For Training instance No. 3 , the Maximally Specific Hypothesis is  ['sunny', 'warm', '?', 'strong', '?', '?']

 The Maximally Specific Hypothesis for the given given set of training examples is :

['sunny', 'warm', '?', 'strong', '?', '?']
```

```
In [ ]:
```

In [ ]:
```
from pandas import read_csv
from sklearn.model_selection import KFold
df=read_csv('iris.csv')
array=df.values
X=array[:,0:-1]
Y=array[:,-1]
kfold=KFold(n_splits=10,shuffle=True,random_state=0)
print(X)
print(Y)
```

In [5]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
model=DecisionTreeClassifier(criterion="entropy",random_state=0)
result=cross_val_score(model,X,Y,cv=kfold,scoring='accuracy')
print("Accuracy={:.2f}%,SD={:.2f})".format(result.mean()*100,result.std()))
```

```
Accuracy=94.67%,SD=0.06)
```

In [6]:
```
model=model.fit(X,Y)
y_pred=model.predict([[5.1,3.7,1.5,0.4]])
print(y_pred)
```

```
[0.]
```

In [8]:
```
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
y_pred=cross_val_predict(model,X,Y,cv=kfold)
conf_mat=confusion_matrix(Y,y_pred)
print('\n',conf_mat)
report=classification_report(Y,y_pred)
print('\n',report)
```

```
[[50  0  0]
 [ 0 46  4]
 [ 0  4 46]]

              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        50
         1.0       0.92      0.92      0.92        50
         2.0       0.92      0.92      0.92        50

    accuracy                           0.95       150
   macro avg       0.95      0.95      0.95       150
weighted avg       0.95      0.95      0.95       150
```

In [10]:
```
from sklearn import tree
import graphviz
fn=['sepal length(cm)','sepal width(cm)','petal length(cm)','petal width(cm)']
cn=['0','1','2']
tree.export_graphviz(model,
                     out_file="tree.dot",
                     feature_names=fn,
                     class_names=cn,
                     filled=True)
```

jupyter lab3 Last Checkpoint: 02/13/2023 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
from pandas import read_csv
from sklearn.model_selection import KFold
df=read_csv('iris.csv')
array=df.values
x=array[:,0:-1]
y=array[:,-1]
kfold=KFold(n_splits=10,shuffle=True,random_state=0)
```

In [17]:
```
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.model_selection import cross_val_score

model=BaggingClassifier(n_estimators=20,random_state=0)
#model=RandomForestClassifier(random_state=0,n_estimators=100,criterion="entropy")
#model=ExtraTreesClassifier(n_estimators=100,random_state=0)
result=cross_val_score(model,x,y,cv=kfold,scoring='accuracy')
print("(accuracy={:.2f}%,sd={:.2f})".format(result.mean()*100,result.std()))
```

```
(accuracy=95.33%,sd=0.07)
```

jupyter lab3 Last Checkpoint: 02/13/2023 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted

Run Code

In [15]:
```
model=model.fit(x,y)
y_perd=model.predict([[5.1,3.7,1.5,0.4]])
print(y_perd)
```

```
[0.]
```

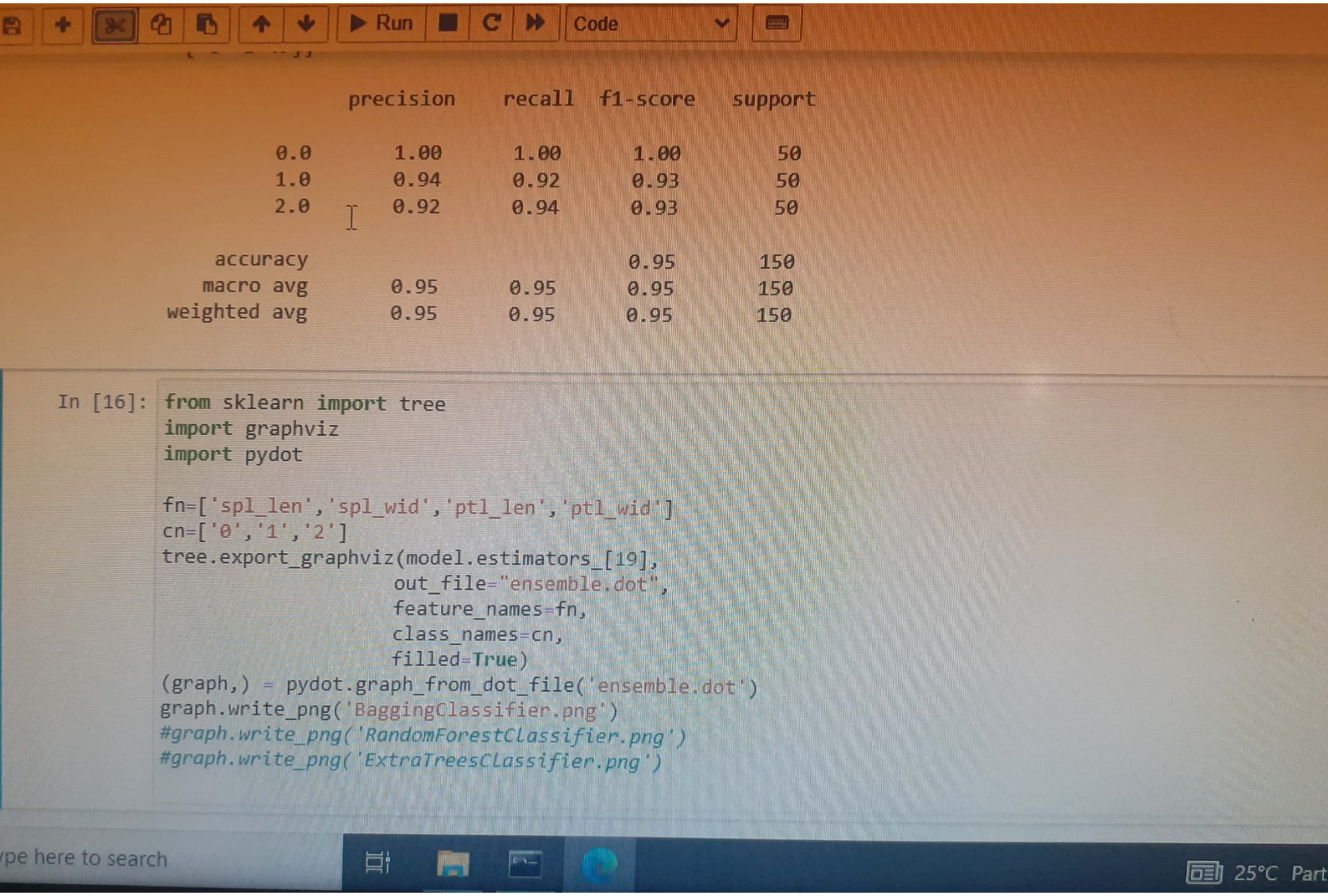
In [4]:
```
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
y_pred =cross_val_predict(model,x,y,cv=kfold)
conf_mat=confusion_matrix(y,y_pred)
print('\n',conf_mat)
report=classification_report(y,y_pred)
print('\n',report)
```

```
 [[50  0  0]
 [ 0 46  4]
 [ 0  3 47]]

              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        50
         1.0       0.94      0.92      0.93        50
         2.0       0.92      0.94      0.93        50
```

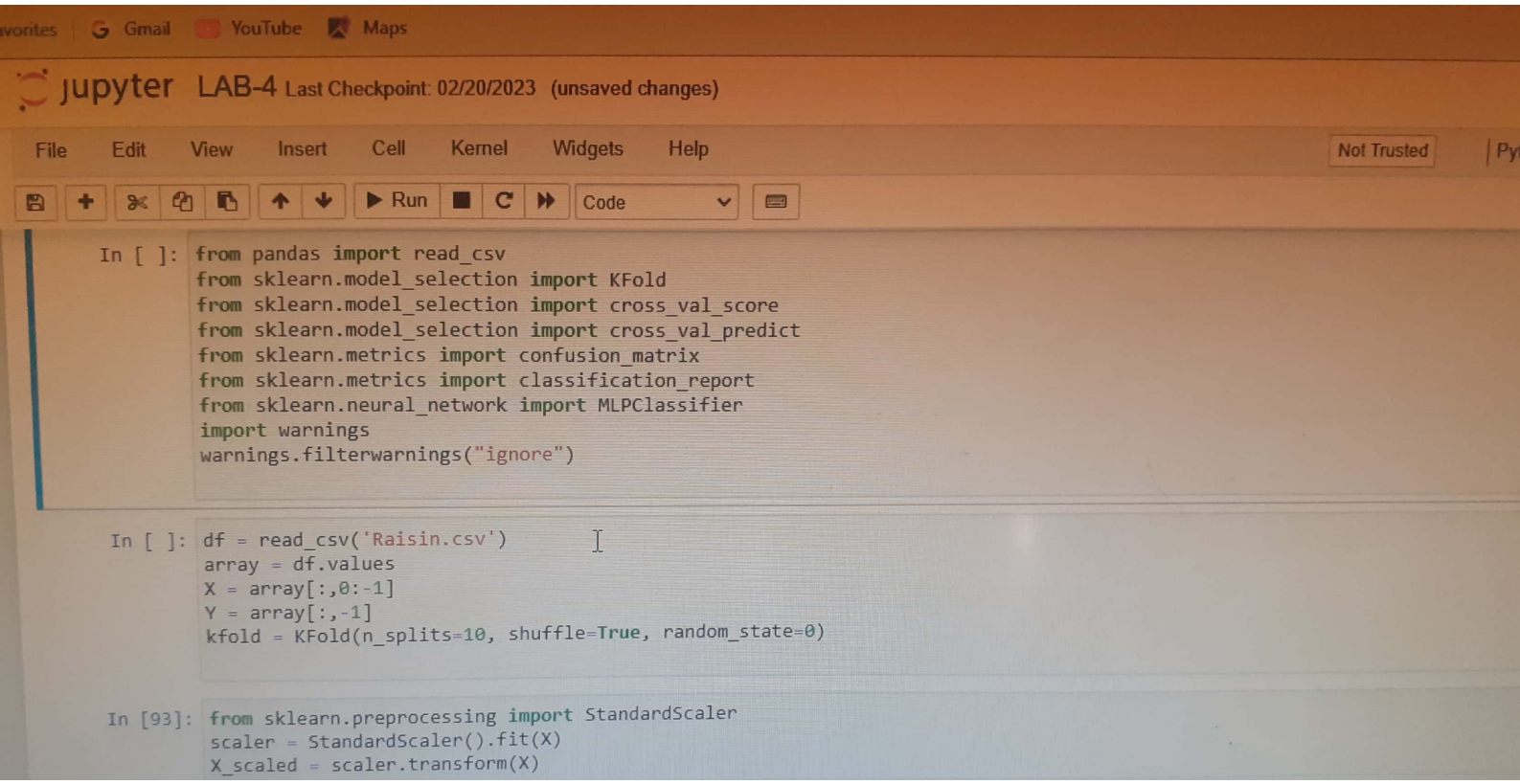
Type here to search 25°C Partly cloudy



	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	50
1.0	0.94	0.92	0.93	50
2.0	0.92	0.94	0.93	50
accuracy			0.95	150
macro avg	0.95	0.95	0.95	150
weighted avg	0.95	0.95	0.95	150

In [16]:
```
from sklearn import tree
import graphviz
import pydot

fn=['spl_len','spl_wid','ptl_len','ptl_wid']
cn=['0','1','2']
tree.export_graphviz(model.estimators_[19],
                     out_file="ensemble.dot",
                     feature_names=fn,
                     class_names=cn,
                     filled=True)
(graph,) = pydot.graph_from_dot_file('ensemble.dot')
graph.write_png('BaggingClassifier.png')
#graph.write_png('RandomForestClassifier.png')
#graph.write_png('ExtraTreesClassifier.png')
```



jupyter LAB-4 Last Checkpoint: 02/20/2023 (unsaved changes)

Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
scaler = StandardScaler().fit(X)
X_scaled = scaler.transform(X)
```

In [105]:
```
model = MLPClassifier(hidden_layer_sizes=(100,100), activation='relu', shuffle=True, random_state=0, max_iter=100)
result = cross_val_score(model, X_scaled, Y, cv=kfold, scoring='accuracy')
print("Accuracy = {:.2f}%".format(result.mean()*100))
Y_pred = cross_val_predict(model, X_scaled, Y, cv=kfold)
conf_mat = confusion_matrix(Y, Y_pred)
print('\n',conf_mat)
report = classification_report(Y, Y_pred)
print('\n',report)
```

```
Accuracy = 86.89%

 [[378  72]
 [ 46 404]]

              precision    recall  f1-score   support

       Besni       0.89      0.84      0.86       450
     Kecimen       0.85      0.90      0.87       450

    accuracy                           0.87       900
   macro avg       0.87      0.87      0.87       900
weighted avg       0.87      0.87      0.87       900
```

pe here to search 25°C Partly cloudy

```
In [100]: model = model.fit(X,Y)
          Y_pred=model.predict([[79408,352.1907699,290.8275329,0.56401133,81463,0.792771926,1073.251]])
          print(Y_pred)

          ['Kecimen']

In [96]: print(X)

         [[87524 442.2460114 253.291155 ... 90546 0.758650579 1184.04]
          [75166 406.690687 243.0324363 ... 78789 0.68412957 1121.786]
          [90856 442.2670483 266.3283177 ... 93717 0.637612812 1208.575]
          ...
          [99657 431.7069809 298.8373229 ... 106264 0.741098519 1292.828]
          [93523 476.3440939 254.1760536 ... 97653 0.658798253 1258.548]
          [85609 512.0817743 215.2719758 ... 89197 0.632019963 1272.862]]

In [ ]: print(X_scaled)
```

jupyter LAB_5 Last Checkpoint: Last Monday at 3:16 PM (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
from pandas import read_csv
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.naive_bayes import GaussianNB
import warnings
warnings.filterwarnings("ignore")
```

In [4]:
```
df = read_csv('Pima_Indian.csv')
array = df.values
X = array[:,0:-1]
Y = array[:,-1]
kfold = KFold(n_splits=10, shuffle=True, random_state=0)
```

In [ ]:
```
model = GaussianNB()
result = cross_val_score(model, X, Y, cv=kfold, scoring='accuracy')
print("Accuracy = {:.2f}%".format(result.mean()*100))
Y_pred = cross_val_predict(model, X, Y, cv=kfold)
conf_mat = confusion_matrix(Y, Y_pred)
print('\n',conf_mat)
report = classification_report(Y, Y_pred)
print('\n',report)
```

Type here to search 25°C Partly cloudy

In [ ]:
```
model = GaussianNB()
result = cross_val_score(model, X, Y, cv=kfold, scoring='accuracy')
print("Accuracy = {:.2f}%".format(result.mean()*100))
Y_pred = cross_val_predict(model, X, Y, cv=kfold)
conf_mat = confusion_matrix(Y, Y_pred)
print('\n',conf_mat)
report = classification_report(Y, Y_pred)
print('\n',report)
```

In [11]:
```
model = model.fit(X,Y)
Y_pred=model.predict([[10,115,0,0,0,35.3,0.134,29]])
print(Y_pred)
```

```
[1.]
```

In [ ]: