Devanshu Surana
PC-23, Panel C
1032210755

ML Lab Assignment 3

FAQ's

Q1) What is Decision Tree Classifier?
→ A decision tree classifier is a supervised learning algorithm that creates a classification model by building a decision tree. A decision tree has a hierarchical tree structure with root node, branches, internal nodes and leaf nodes. Each node in the tree specifies a test on an attribute, and each branch descending from that node corresponds to one of the possible values for that attribute.

Q2) What are some advantages of decision trees?
→ Advantages:
1) Compared to other algo's, decision tree requires less effort for data preparation during pre-processing.
2) Decision tree does not require normalization of data
3) Decision tree does not require scaling of data.
4) Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.

Q3) How does a decision tree work?

→ ① It all starts with root node containing all data.

② Choose the feature and threshold that best splits the data into two subsets, minimizing that variance of target variable within each subset.

③ Repeat steps ② and ③ recursively for each child node until a stopping extension is met.

④ Each leaf node represents a region with low variance in the target variable. Assign the avg value of target variable to the leaf node.

Q4) How do you prevent overfitting in a decision tree?

→ 1) Pruning:
This method involves ~~remaining~~ removing branches or nodes that don't contributes much to the accuracy or complexity of the tree. Pruning reduces complexity of the tree and prevents it from overfitting.

2) Dimensionality Reduction:
This method reduces dimensions of feature sets. As the number of feature increases, the model becomes more complex and increases the chances of overfitting.

Q5) What is pruning in decision trees?

→ Pruning in Decision trees involves removing bra-
-nches or nodes that don't contribute much to
the accuracy ~~of~~ or complexity of the tree.
Pruning reduces complexity of the tree and
prevents it from overfitting.

# ml-lab3A

February 28, 2024

[11]:
```
#importing the libraries
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn import tree
from sklearn.datasets import load_iris
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import cross_val_predict, KFold, train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier, plot_tree
```

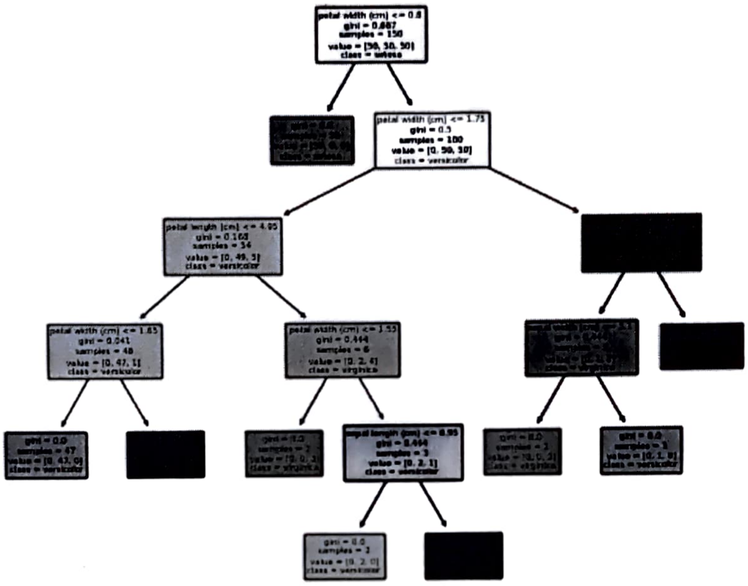
[2]:
```
#loading the dataset
iris = load_iris()
X = iris.data
y = iris.target
```

[3]:
```
X_train,y_train,X_test,y_test = train_test_split(X,y ,train_size=0.
  ↪3,random_state=42)
```

[4]:
```
# Initialize the DecisionTreeClassifier with Gini impurity criterion
clf = DecisionTreeClassifier(criterion="gini")
```

[5]:
```
#performing kfold operation
kf = KFold(n_splits=6, shuffle=True, random_state=42)
```

[6]:
```
#plot the decision tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X,y)
tree.plot_tree(clf,filled=True,feature_names=iris.feature_names,␣
  ↪class_names=iris.target_names)
plt.show()
```

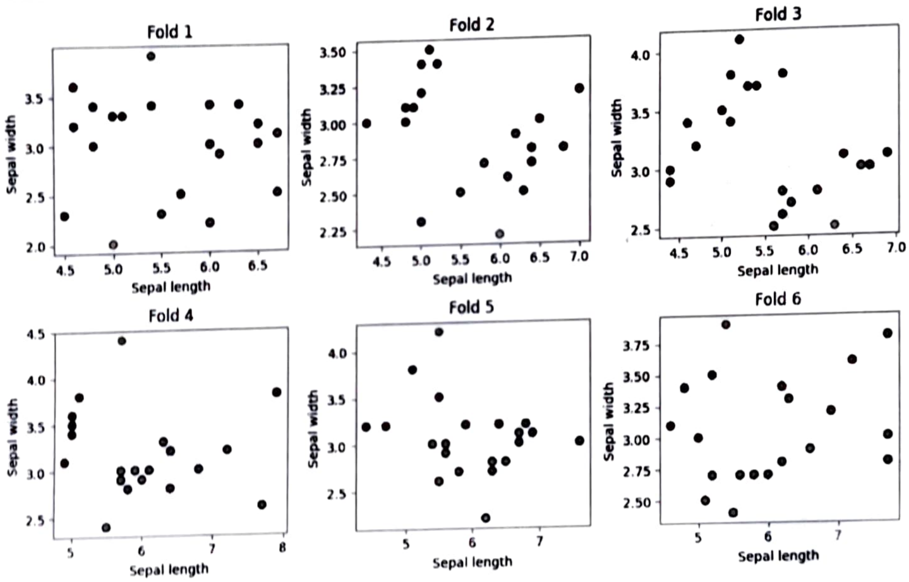



[7]:
```
#preform kfold cross validation
for fold_idx, (train_index, test_index) in enumerate(kf.split(X)):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

print(f"Fold {fold_idx + 1}:")
print(f"  Training samples: {len(X_train)}")
print(f"  Test samples: {len(X_test)}")

y_pred_test = cross_val_predict(clf, X_test, y_test, cv=5)
conf_mat = confusion_matrix(y_test, y_pred_test)
class_report = classification_report(y_test, y_pred_test)
```

```
Fold 6:
  Training samples: 125
  Test samples: 25
```

[8]:
```
plt.figure(figsize=(10, 6))
for i, (train_index, test_index) in enumerate(kf.split(X_train)):
    X_train_kf, X_val_kf = X_train[train_index], X_train[test_index]
    y_train_kf, y_val_kf = y_train[train_index], y_train[test_index]

    clf.fit(X_train_kf, y_train_kf)
```

```
    y_pred_kf = clf.predict(X_val_kf)

    plt.subplot(2, 3, i+1)
    plt.scatter(X_val_kf[:, 0], X_val_kf[:, 1], c=y_pred_kf, cmap=plt.cm.Set1,␣
↪edgecolor='k')
    plt.xlabel('Sepal length')
    plt.ylabel('Sepal width')
    plt.title(f'Fold {i+1}')

plt.tight_layout()
plt.show()
```




[9]:
```
# Generate and print the classification report
#print("Classification Report:")
#print(classification_report(y_test, y_pred_test, target_names=iris.
↪target_names))
```

[10]:
```
#printing confusion and classification matrix
print("Confusion Matrix:")
print(conf_mat)
print("\nClassification Report:")
print(class_report)
```

```
Confusion Matrix:
[[ 5  0  0]
 [ 0  7  1]
```

```
[ 0  1 11]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         5
           1       0.88      0.88      0.88         8
           2       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.93      0.93      0.93        25
weighted avg       0.92      0.92      0.92        25
```

Pankaj
01/03/24.

# ml-lab~~4~~ 3B

February 28, 2024

[22]:
```python
#importing libraries
from sklearn.datasets import load_diabetes
#warning.filterwarning('ignore')
from sklearn.model_selection import KFold
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

[23]:
```python
diabetes = load_diabetes()
X = diabetes.data
y = diabetes.target
```

[24]:
```python
kf = KFold(n_splits=6, shuffle=True, random_state=42)
```

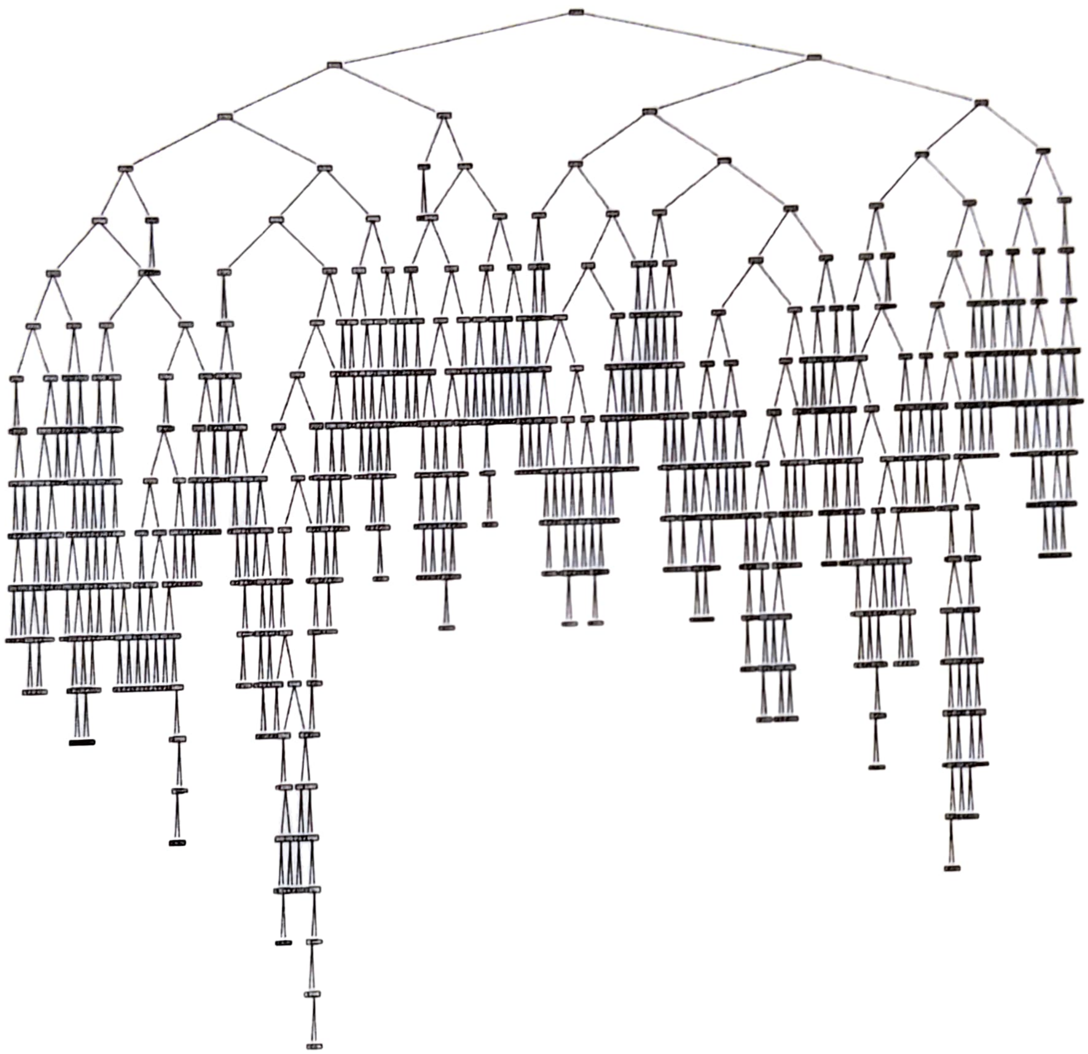
[30]:
```python
for fold_idx, (train_index, test_index) in enumerate(kf.split(X)):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

print(f"Fold {fold_idx + 1}:")
print(f"  Training samples: {len(X_train)}")
print(f"  Test samples: {len(X_test)}")
```

```
Fold 6:
  Training samples: 369
  Test samples: 73
```

[35]:
```python
regressor = DecisionTreeRegressor()
regressor.fit(X_train, y_train)
```

[35]: DecisionTreeRegressor()

[29]:
```python
# Plot the decision tree
plt.figure(figsize=(20,20))
plot_tree(regressor, filled=True)
plt.show()
```



[36]:
```
y_pred = regressor.predict(X_test)
```

[40]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

[44]:
```
print("Mean Absolute Error:", mae)
print("Mean Squared Error:", mse)
print("R-squared:", r2)
```

```
Mean Absolute Error: 71.04109589041096
Mean Squared Error: 7882.328767123287
```

**R-squared:** -0.7273596458834859

Pankaj
01/03/24.