ML LAB

ASSIGNMENT 1

NAME: SWAPNIL GHOSH
ROLL: 001911001067
DEPT: IT
YEAR: 4TH

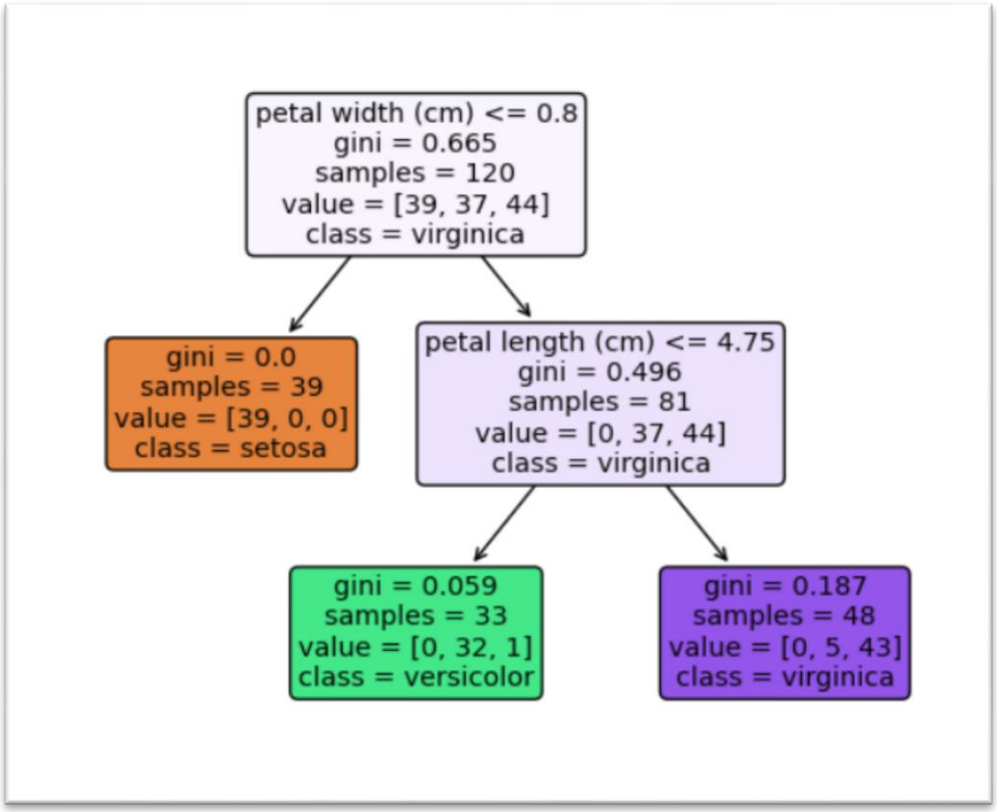
IRIS - CLASSIFICATION

DECISION TREE CLASSIFIER - CODE & EVALUATION

```python
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

iris = load_iris()
X, y = iris.data, iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=0)

classifier = DecisionTreeClassifier(criterion="gini", max_depth=2,
    splitter="best", max_features = 2, max_leaf_nodes=3, random_state=1)
model = classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))

tree.plot_tree(decision_tree = model, feature_names = iris.feature_names,
    class_names = iris.target_names, filled=True, rounded=True)
plt.savefig("iris.png")
```

```
Confusion Matrix:
[[11  0  0]
 [ 0 12  1]
 [ 0  0  6]]
--------------------------------------------------
Performance Evaluation:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      0.92      0.96        13
           2       0.86      1.00      0.92         6

    accuracy                           0.97        30
   macro avg       0.95      0.97      0.96        30
weighted avg       0.97      0.97      0.97        30
```

DECISION TREE CLASSIFIER - TREE PLOT



NAIVE BAYES - MULTINOMIALNB CLASSIFIER

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix

iris = load_iris()
X, y = iris.data, iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=0)

classifier = MultinomialNB(alpha=1.0, fit_prior=False)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))
```

```
onfusion matrix:
[11  0  0]
[ 0 13  0]
[ 0  0  6]]
--------------------------------------------------
erformance Evaluation:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00         6

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
eighted avg       1.00      1.00      1.00        30
```

NAIVE BAYES - BERNOULLINB CLASSIFIER

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import BernoulliNB
from sklearn.metrics import classification_report, confusion_matrix

iris = load_iris()
X, y = iris.data, iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=1)

classifier = BernoulliNB(alpha=1.0, binarize=1.7,fit_prior=False)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))
```

```
onfusion Matrix:
[10  1  0]
[ 0 13  0]
[ 0  0  6]]
--------------------------------------------------
erformance Evaluation:
              precision    recall  f1-score   support

           0       1.00      0.91      0.95        11
           1       0.93      1.00      0.96        13
           2       1.00      1.00      1.00         6

    accuracy                           0.97        30
   macro avg       0.98      0.97      0.97        30
eighted avg       0.97      0.97      0.97        30
```

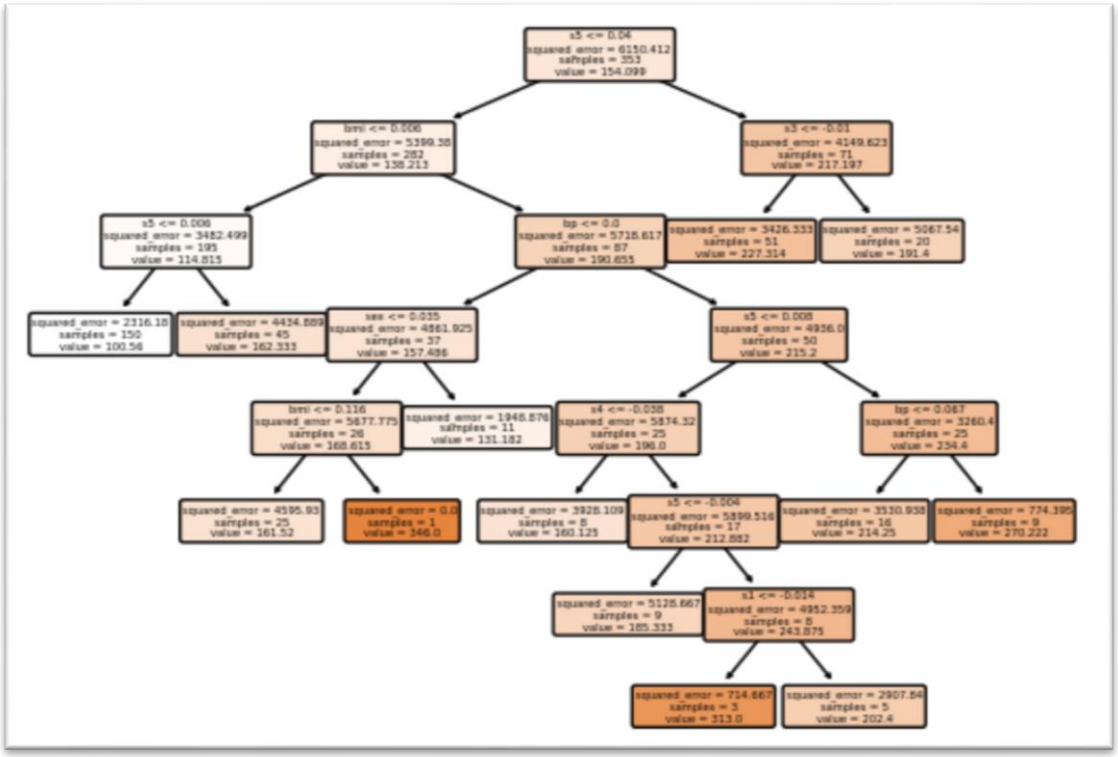
DIABETES - REGRESSION

DECISION TREE REGRESSOR - CODE & EVALUATION

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_diabetes
from sklearn import tree
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor


diabetes = load_diabetes()
X, y = diabetes.data, diabetes.target
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=7)

regressor = DecisionTreeRegressor(criterion="squared_error",
    splitter = "random", max_features="sqrt", ccp_alpha=41,
    random_state=46)
model = regressor.fit(X_train, y_train)
y_pred = regressor.predict(X_test)


print("Performance Evaluation:\n")

print("RMSE: ", np.sqrt(np.mean((y_pred - y_test)**2)))
print("MAE: ", np.mean(np.abs(y_pred - y_test)))
print("R2: ", regressor.score(X_test, y_test))

tree.plot_tree(regressor, feature_names = diabetes.feature_names,
    filled=True, rounded=True)
plt.savefig("diabetes.png")
```

```
abetes % python dt1.py
Performance Evaluation:

RMSE:  51.79107344514936
MAE:  40.67246383197474
R2:  0.4612850990313666
```

DECISION TREE REGRESSOR - TREE PLOT



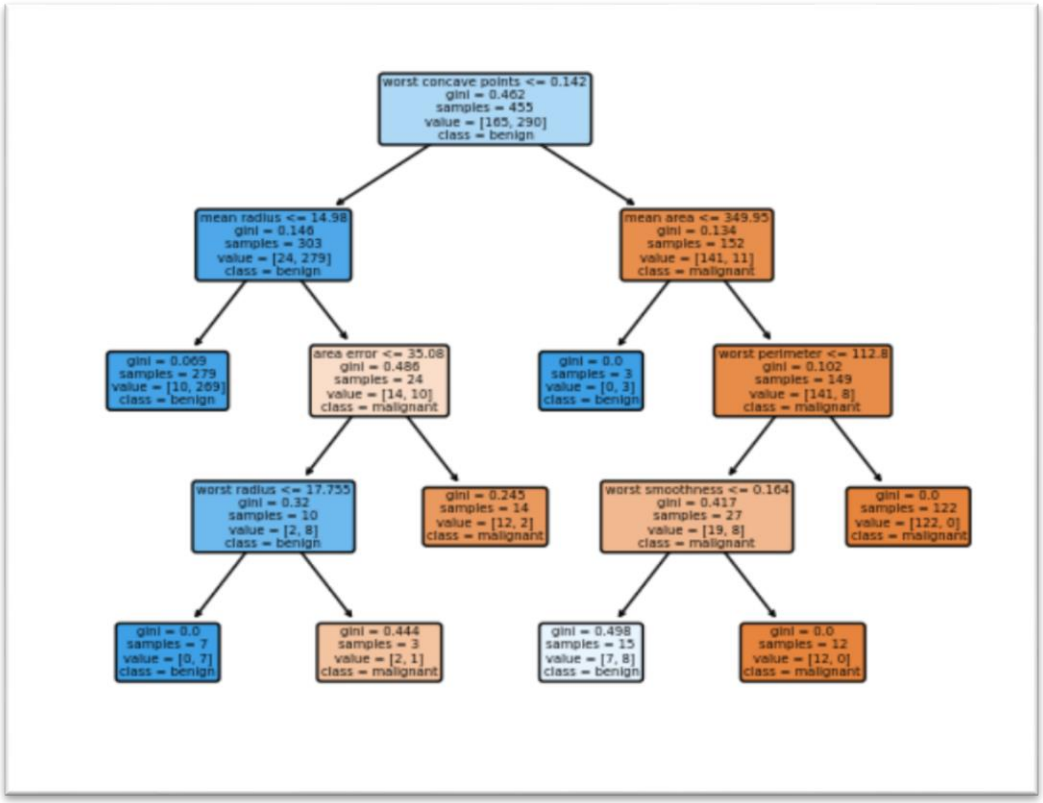
BREAST CANCER WISCONSIN - CLASSIFICATION

DECISION TREE CLASSIFIER - CODE & EVALUATION

```python
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split

cancer = load_breast_cancer()
X, y = cancer.data, cancer.target
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=0)

classifier = DecisionTreeClassifier(criterion="gini", max_depth=4,
    splitter="best", max_features = "sqrt", max_leaf_nodes=8,
    random_state=5)
model = classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))

tree.plot_tree(decision_tree = model, feature_names =
    cancer.feature_names, class_names = cancer.target_names,
    filled=True, rounded=True)
plt.savefig("breast_cancer_wisconsin.png")
```

```
Confusion Matrix:
[[46  1]
 [ 1 66]]
--------------------------------------------------
Performance Evaluation:
              precision    recall  f1-score   support

           0       0.98      0.98      0.98        47
           1       0.99      0.99      0.99        67

    accuracy                           0.98       114
   macro avg       0.98      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114
```

DECISION TREE CLASSIFIER - TREE PLOT



NAIVE BAYES - MULTINOMIALNB CLASSIFIER

```python
from sklearn.datasets import load_breast_cancer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split

cancer = load_breast_cancer()
X, y = cancer.data, cancer.target
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2)

classifier = MultinomialNB(alpha=0.5, fit_prior=True,
    class_prior=[0.45, 0.55])
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))
```

```
Confusion Matrix:
[[34  4]
 [ 3 73]]
--------------------------------------------------
Performance Evaluation:
              precision    recall  f1-score   support

           0       0.92      0.89      0.91        38
           1       0.95      0.96      0.95        76

    accuracy                           0.94       114
   macro avg       0.93      0.93      0.93       114
weighted avg       0.94      0.94      0.94       114
```

NAIVE BAYES - GAUSSIANNB CLASSIFIER

```python
from sklearn.datasets import load_breast_cancer
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split

cancer = load_breast_cancer()
X, y = cancer.data, cancer.target
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=0)

classifier = GaussianNB(var_smoothing=1e-8)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))
```

```
Confusion Matrix:
[[44  3]
 [ 2 65]]
--------------------------------------------------
Performance Evaluation:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95        47
           1       0.96      0.97      0.96        67

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114
```

NAIVE BAYES - BERNOULLINB CLASSIFIER

```python
from sklearn.datasets import load_breast_cancer
from sklearn.naive_bayes import BernoulliNB
from sklearn.metrics import classification_report,
    confusion_matrix
from sklearn.model_selection import train_test_split

cancer = load_breast_cancer()
X, y = cancer.data, cancer.target
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=0)

classifier = BernoulliNB(alpha=1.0,
    binarize=16.8,fit_prior=False)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("--------------------------------------------------")

print("Performance Evaluation:")
print(classification_report(y_test, y_pred))
```

```
Confusion Matrix:
[[42  5]
 [ 1 66]]
--------------------------------------------------
Performance Evaluation:
              precision    recall  f1-score   support

           0       0.98      0.89      0.93        47
           1       0.93      0.99      0.96        67

    accuracy                           0.95       114
   macro avg       0.95      0.94      0.94       114
weighted avg       0.95      0.95      0.95       114
```