Unit 3.2 Graded Assignment = Implementation of classification algorithm

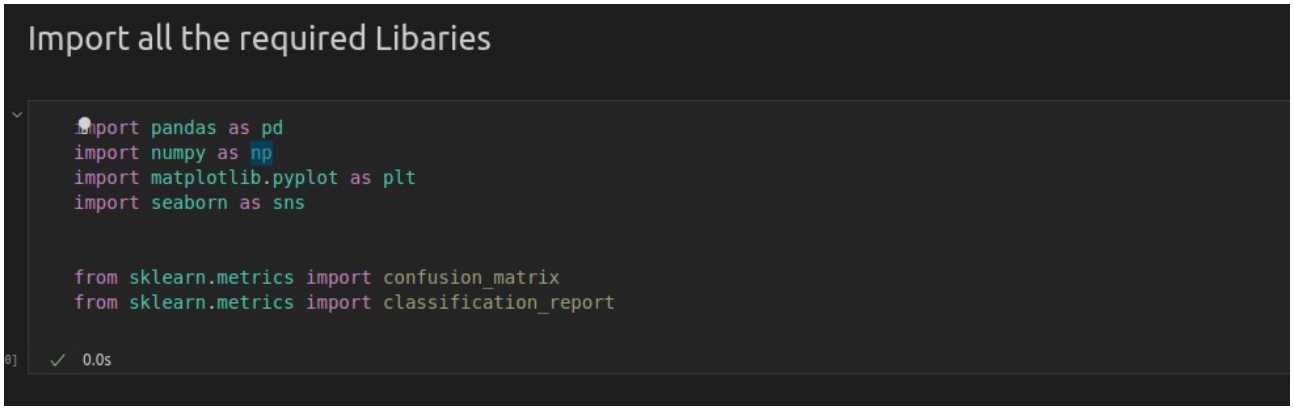
Instructions:
Implement a single classification model of your choice and try to achieve at least an 80% F1 score on the wine dataset.

Submitted by:

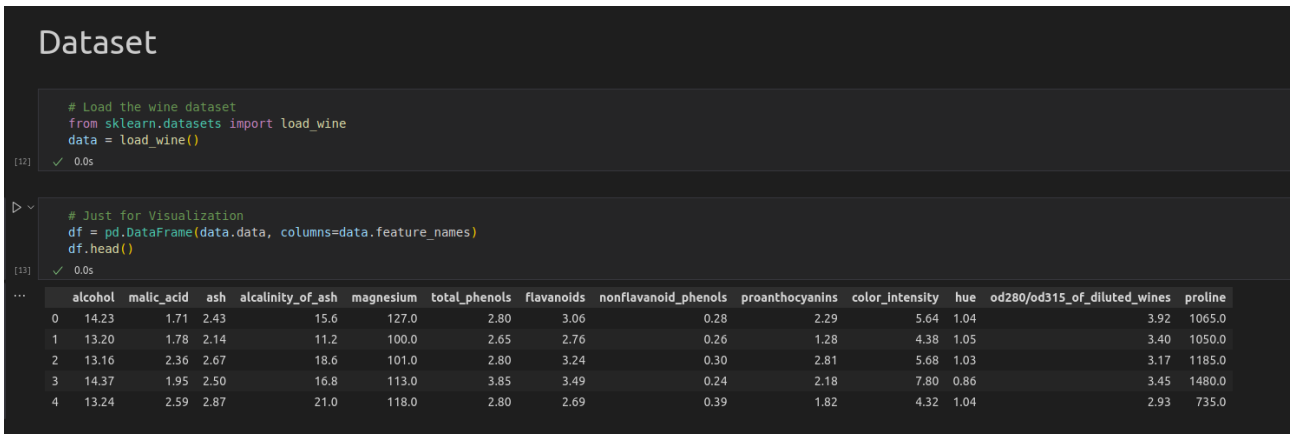
1. **Ali Nasir (2303.KHI.DEG.012)**
2. **Saif ur Rehman (2303.KHI.DEG.007)**

Solution:

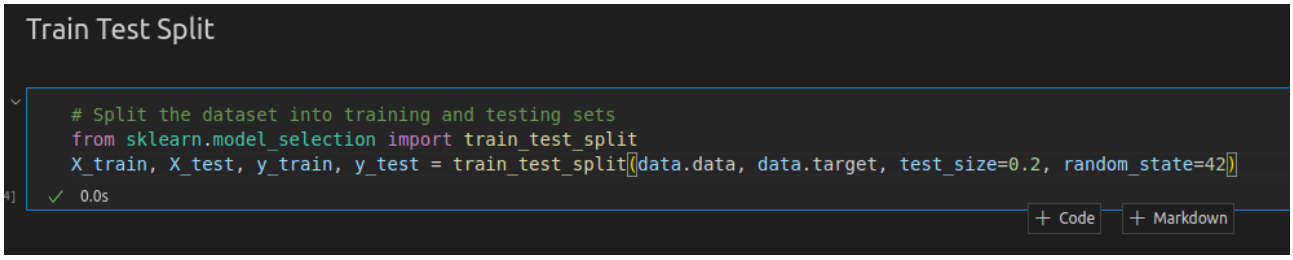
1. First we will import all the required libraries.



2. Import the builtin data from the sklearn library.



3. We have split our dataset into test size 0.2 which means 80 % training data and 20% testing data and a random state of 42 so that any bias ness could be removed.



4. Here we have implemented the decision tree with max depth 3.

Decision Tree Classifier Implementation

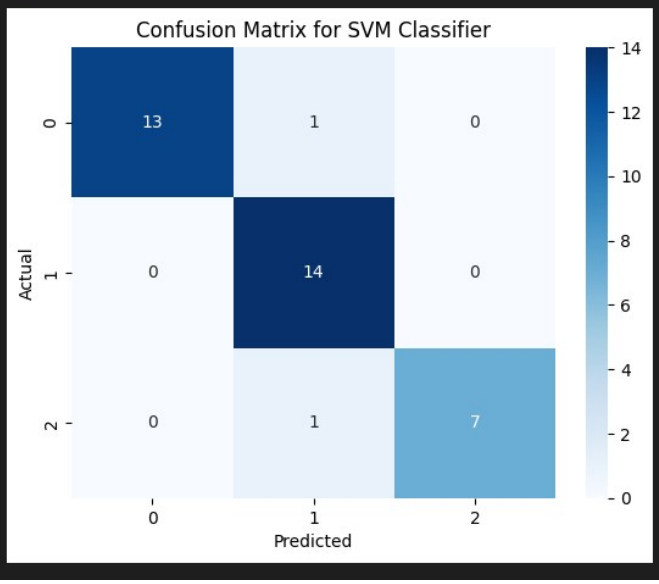
```
from sklearn.tree import DecisionTreeClassifier, plot_tree

# Create a decision tree classifier with max depth of 3
dt = DecisionTreeClassifier(max_depth=3)

# Train the model using the training data
dt.fit(X_train, y_train)

# Use the trained model to predict the labels of the test data
y_pred = dt.predict(X_test)
```

✓ 0.0s

In our classification report, it can be seen that our f1 score is 0.96 with a complete decision tree.

```
matrix = confusion_m  (parameter) annot: Any | None
# Plot the confusion
sns.heatmap(matrix, annot=True, cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix for SVM Classifier')
plt.show()
```

✓ 0.2s




```
from sklearn.metrics import classification_report
# Print the classification report
report = classification_report(y_test, y_pred, target_names=data.target_names)
print(report)
```

✓ 0.0s

```
              precision    recall  f1-score   support

     class_0       1.00      0.93      0.96        14
     class_1       0.88      1.00      0.93        14
     class_2       1.00      0.88      0.93         8

    accuracy                           0.94        36
   macro avg       0.96      0.93      0.94        36
weighted avg       0.95      0.94      0.94        36
```

```python
# Plot the decision tree
plot_tree(dt, filled=True, feature_names=data.feature_names, class_names=data.target_names)
plt.title("Decision Tree")
plt.show()
```

✓ 0.3s