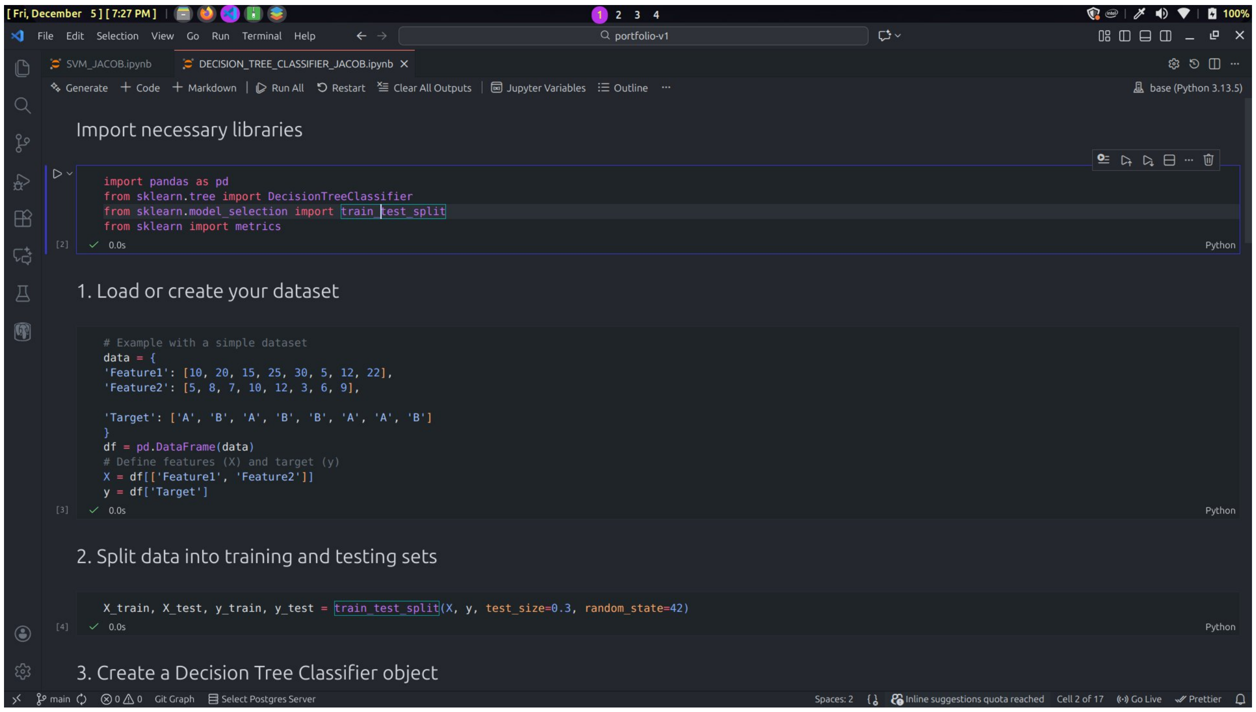
Republic of the Philippines
Partido State University
Goa, Camarines Sur

Name: Troy Tristan Jacob
Course/Year/Section: BSCS-3A

Decision Tree Classifier



Import necessary libraries

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
```
[2] ✓ 0.0s Python

1. Load or create your dataset

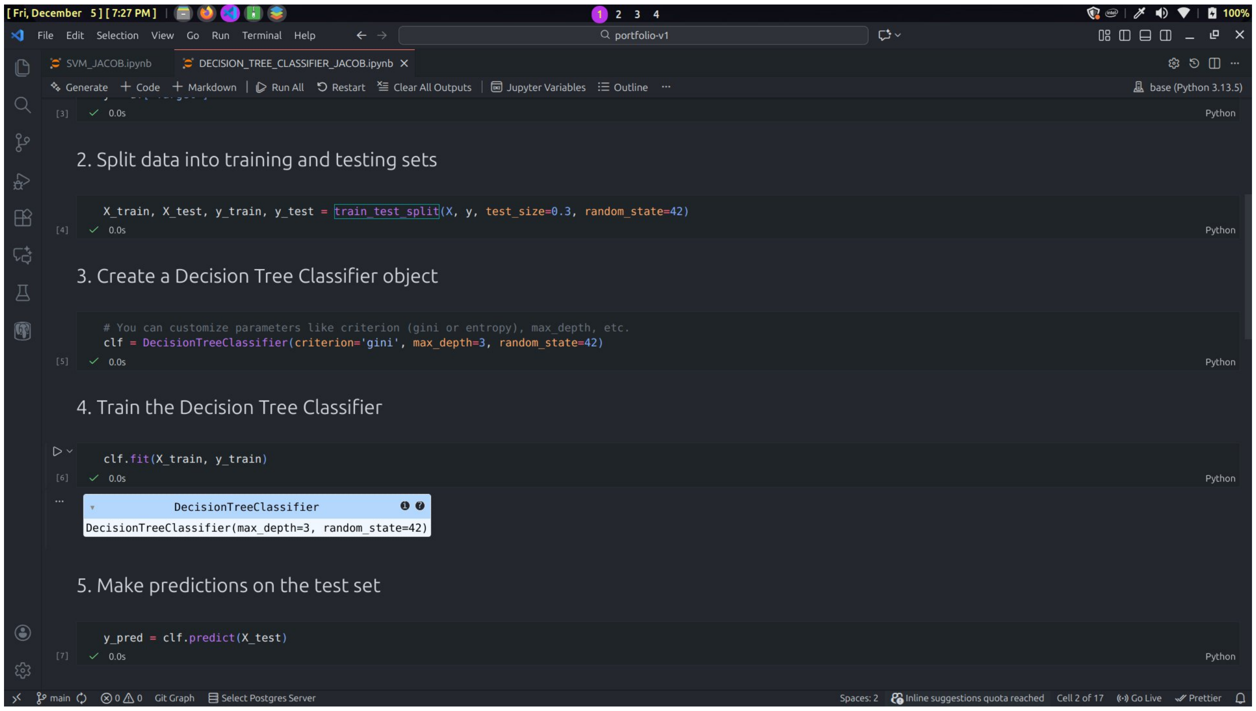
```
# Example with a simple dataset
data = {
'Feature1': [10, 20, 15, 25, 30, 5, 12, 22],
'Feature2': [5, 8, 7, 10, 12, 3, 6, 9],

'Target': ['A', 'B', 'A', 'B', 'B', 'A', 'A', 'B']
}
df = pd.DataFrame(data)
# Define features (X) and target (y)
X = df[['Feature1', 'Feature2']]
y = df['Target']
```
[3] ✓ 0.0s Python

2. Split data into training and testing sets

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```
[4] ✓ 0.0s Python

3. Create a Decision Tree Classifier object



2. Split data into training and testing sets

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

3. Create a Decision Tree Classifier object

```
# You can customize parameters like criterion (gini or entropy), max_depth, etc.
clf = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=42)
```

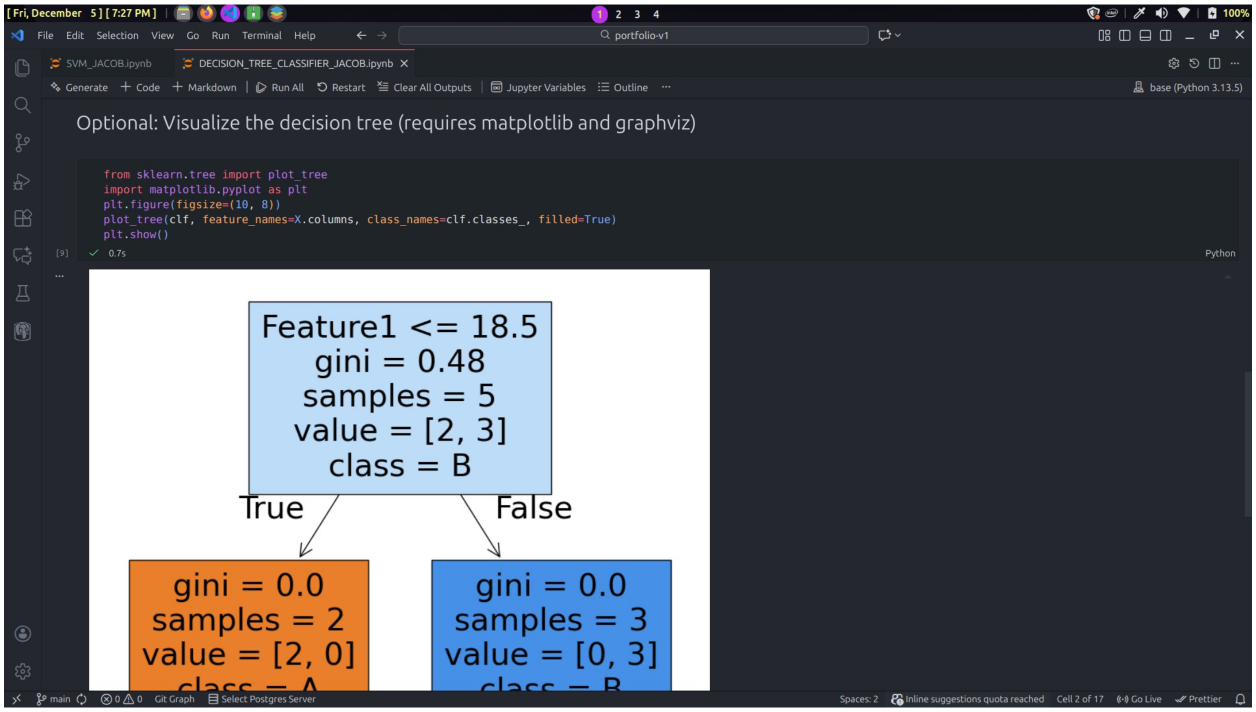
4. Train the Decision Tree Classifier

```
clf.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=3, random_state=42)
```

5. Make predictions on the test set

```
y_pred = clf.predict(X_test)
```



Optional: Visualize the decision tree (requires matplotlib and graphviz)

```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt
plt.figure(figsize=(10, 8))
plot_tree(clf, feature_names=X.columns, class_names=clf.classes_, filled=True)
plt.show()
```

Interpretation of Results

The decision tree visualization demonstrates a simple yet highly effective classification model that achieves perfect accuracy (1.0) on the dataset. The tree structure consists of a single decision node (root) that splits into two pure leaf nodes, indicating an optimal classification scenario.

The root node uses Feature1 with a threshold of 18.5 as the sole splitting criterion, containing all 5 samples with a Gini impurity of 0.48 and a distribution of [2, 3] between classes A and B. This split demonstrates that Feature1 is sufficiently powerful to completely separate the two classes. Samples with Feature1 ≤ 18.5 are directed to the left branch, while those with Feature1 > 18.5 go to the right branch.

Both leaf nodes achieve a Gini impurity of 0.0, indicating perfect homogeneity. The left leaf (orange) contains 2 samples, all belonging to class A, with a value distribution of [2, 0]. The right leaf (blue) contains 3 samples, all belonging to class B, with a value distribution of [0, 3]. This perfect separation demonstrates that the decision boundary at Feature1 = 18.5 completely distinguishes between the two classes without any misclassifications.

The shallow depth of the tree (only one level) suggests that the classification problem is relatively simple, with a clear linear separability along a single feature dimension. The absence of further splits indicates that no additional features or more complex decision rules are necessary for perfect classification.

Conclusion

The decision tree classifier achieves optimal performance with remarkable simplicity, requiring only a single split to perfectly classify all samples. The 100% accuracy reflects ideal conditions where Feature1 alone provides complete discriminatory power at the threshold of 18.5. This exceptional result suggests either a well-curated dataset, a naturally well-separated problem, or potentially a small sample size that may not fully represent real-world complexity. While this perfect performance is impressive, it's important to validate the model on unseen data to ensure it generalizes beyond the training set and doesn't simply reflect overfitting to a limited sample. The tree's simplicity is advantageous for interpretability and computational efficiency, making it an ideal model if this performance translates to new data. However, the small sample size (n=5) warrants caution—additional testing with larger, more diverse datasets would provide greater confidence in the model's robustness and real-world applicability.

Interpretation of Results

The decision tree visualization demonstrates a simple yet highly effective classification model that achieves perfect accuracy (1.0) on the dataset. The tree structure consists of a single decision node (root) that splits into two pure leaf nodes, indicating an optimal classification scenario.

The root node uses Feature1 with a threshold of 18.5 as the sole splitting criterion, containing all 5 samples with a Gini impurity of 0.48 and a distribution of [2, 3] between classes A and B. This split demonstrates that Feature1 is sufficiently powerful to completely separate the two classes. Samples with Feature1 ≤ 18.5 are directed to the left branch, while those with Feature1 > 18.5 go to the right branch.

Both leaf nodes achieve a Gini impurity of 0.0, indicating perfect homogeneity. The left leaf (orange) contains 2 samples, all belonging to class A, with a value distribution of [2, 0]. The right leaf (blue) contains 3 samples, all belonging to class B, with a value distribution of [0, 3]. This perfect separation demonstrates that the decision boundary at Feature1 = 18.5 completely distinguishes between the two classes without any misclassifications.

The shallow depth of the tree (only one level) suggests that the classification problem is relatively simple, with a clear linear separability along a single feature dimension. The absence of further splits indicates that no additional features or more complex decision rules are necessary for perfect classification.

Conclusion

The decision tree classifier achieves optimal performance with remarkable simplicity, requiring only a single split to perfectly classify all samples. The 100% accuracy reflects ideal conditions where Feature1 alone provides complete discriminatory power at the threshold of 18.5. This exceptional result suggests either a well-curated dataset, a naturally well-separated problem, or potentially a small sample size that may not fully represent real-world complexity. While this perfect performance is impressive, it's important to validate the model on unseen data to ensure it generalizes beyond the training set and doesn't simply reflect overfitting to a limited sample. The tree's simplicity is advantageous for interpretability and computational efficiency, making it an ideal model if this performance translates to new data. However, the small sample size (n=5) warrants caution—additional testing with larger, more diverse datasets would provide greater confidence in the model's robustness and real-world applicability.