Decision Trees

▼ Imports

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

✓ 7.6s Python

```python
df = pd.read_csv("penguins_size.csv")
```

✓ 0.4s Python

```python
df.head()
```

✓ 0.6s Python

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass
0	Adelie	Torgersen	39.1	18.7	181.0	3750
1	Adelie	Torgersen	39.5	17.4	186.0	3800
2	Adelie	Torgersen	40.3	18.0	195.0	3250
3	Adelie	Torgersen	NaN	NaN	NaN	Na
4	Adelie	Torgersen	36.7	19.3	193.0	3450

▼ Missing Data

Missing Data

```
1  df.isnull().sum()
```

✓ 0.1s

```
species              0
island               0
culmen_length_mm     2
culmen_depth_mm      2
flipper_length_mm    2
body_mass_g          2
sex                 10
dtype: int64
```

```
1  df.info()
```

✓ 0.1s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   species            344 non-null    object
 1   island             344 non-null    object
 2   culmen_length_mm   342 non-null    float64
 3   culmen_depth_mm    342 non-null    float64
 4   flipper_length_mm  342 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                334 non-null    object
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
```

```
1  df = df.dropna()
```
✓ 0.4s

```
1  df.info()
```
✓ 0.3s

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 334 entries, 0 to 343
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   species            334 non-null    object 
 1   island             334 non-null    object 
 2   culmen_length_mm   334 non-null    float64
 3   culmen_depth_mm    334 non-null    float64
 4   flipper_length_mm  334 non-null    float64
 5   body_mass_g        334 non-null    float64
 6   sex                334 non-null    object 
dtypes: float64(4), object(3)
```

```
1 df["island"].unique()
```

✓ 0.7s

```
array(['Torgersen', 'Biscoe', 'Dream'], dtype=object)
```

```
1 df["sex"].unique()
```

✓ 0.5s

```
array(['MALE', 'FEMALE', '.'], dtype=object)
```

```
1 df[df["sex"]=="."]
```

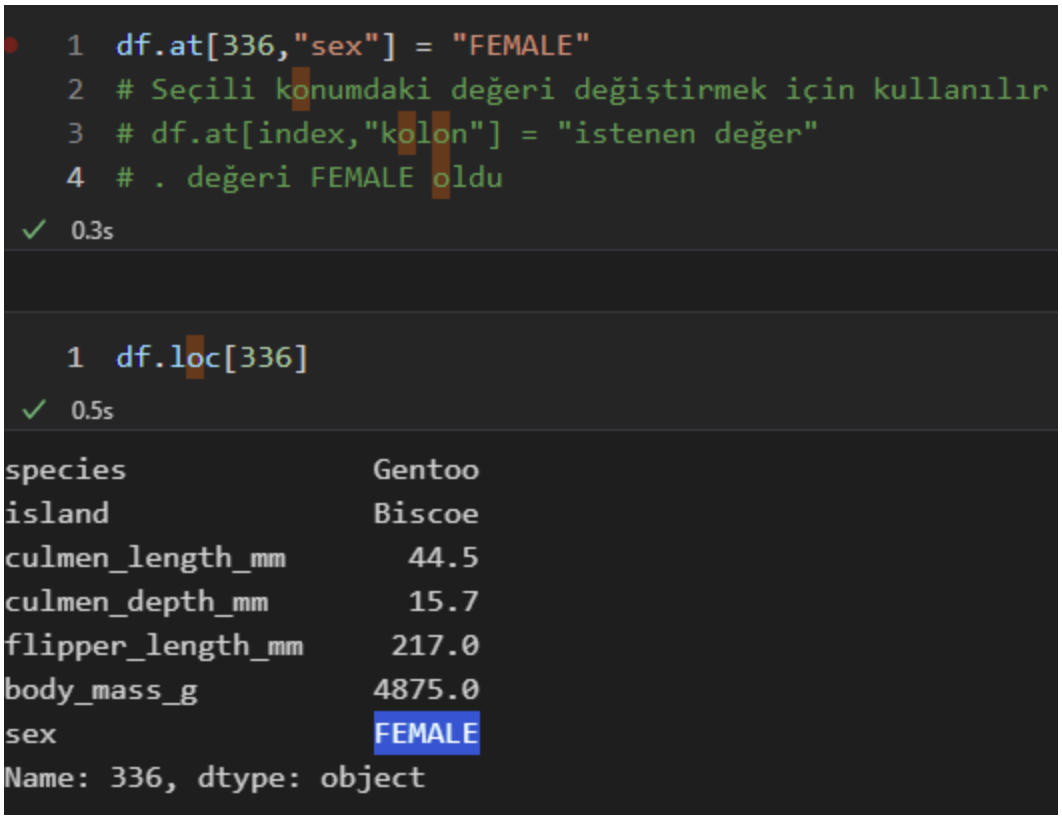
✓ 0.8s

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
336	Gentoo	Biscoe	44.5	15.7	217.0	4875.0	.

```
1 df[df["species"]=="Gentoo"].groupby("sex").describe().transpose()
```

✓ 0.3s

	sex	.	FEMALE	MALE
culmen_length_mm	count	1.0	58.000000	61.000000
	mean	44.5	45.563793	49.473770
	std	NaN	2.051247	2.720594
	min	44.5	40.900000	44.400000
	25%	44.5	43.850000	48.100000
	50%	44.5	45.500000	49.500000
	75%	44.5	46.875000	50.500000
	max	44.5	50.500000	59.600000
culmen_depth_mm	count	1.0	58.000000	61.000000
	mean	15.7	14.237931	15.718033
	std	NaN	0.540249	0.741060

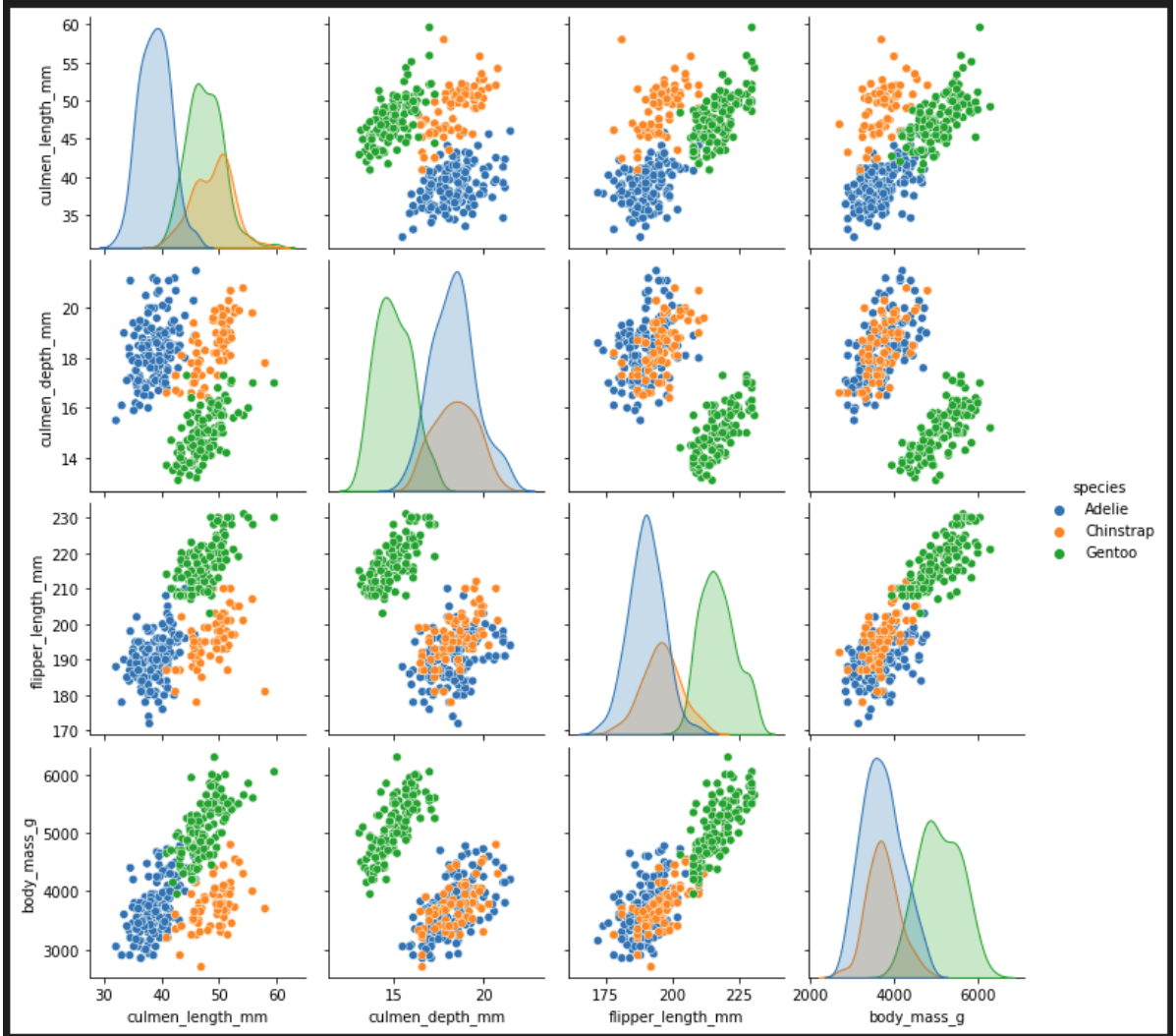


▼ **Data, Viz & Dummy Var**

```
1  sns.pairplot(df,hue="species")
```

✓ 8.9s

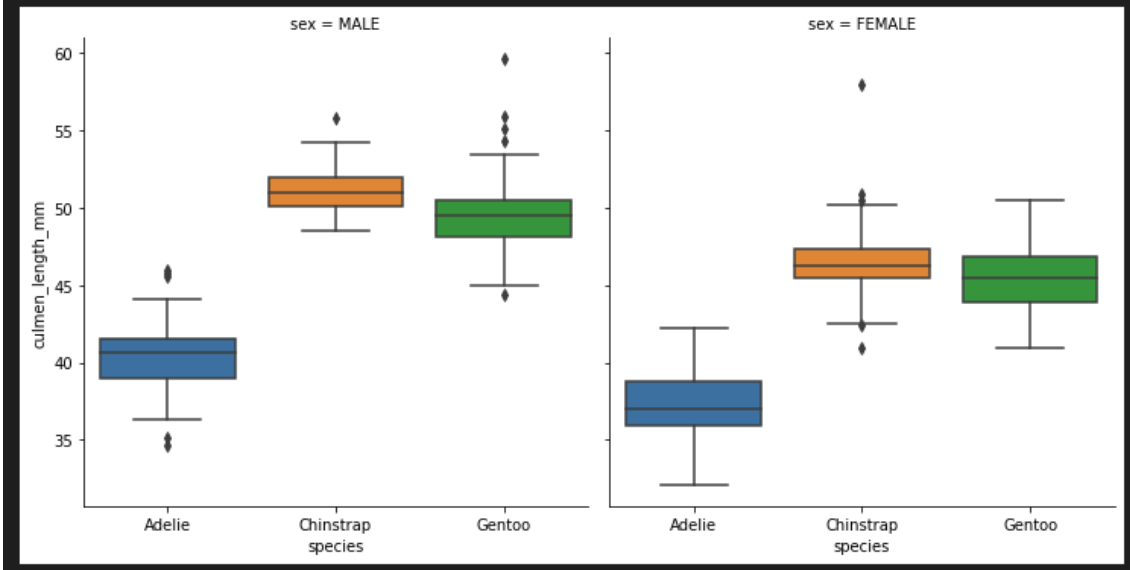
```
<seaborn.axisgrid.PairGrid at 0x1ccf5ba05b0>
```



```
1  sns.catplot(data=df, x="species", y="culmen_length_mm", kind="box", col="sex")
```

✓ 1.1s

```
<seaborn.axisgrid.FacetGrid at 0x1ccf5ba0d00>
```



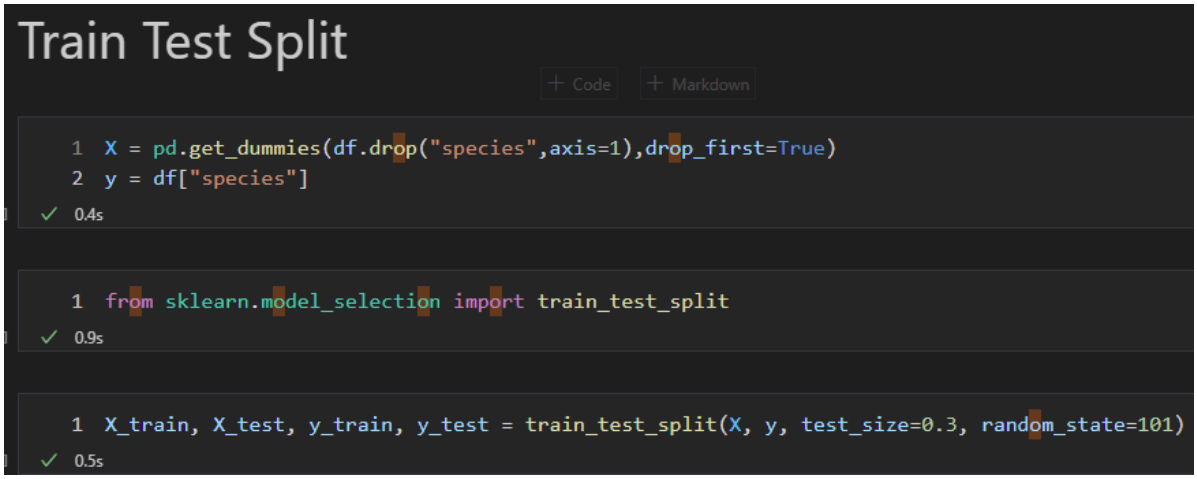

```
1  pd.get_dummies(df.drop("species",axis=1),drop_first=True)
```

✓ 0.1s Python

culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	island_Dream	island_Torgersen	sex_MALE
39.1	18.7	181.0	3750.0	0	1	1
39.5	17.4	186.0	3800.0	0	1	0
40.3	18.0	195.0	3250.0	0	1	0
36.7	19.3	193.0	3450.0	0	1	0
39.3	20.6	190.0	3650.0	0	1	1
...	...	...	...	...	...	...
47.2	13.7	214.0	4925.0	0	0	0
46.8	14.3	215.0	4850.0	0	0	0
50.4	15.7	222.0	5750.0	0	0	1
45.2	14.8	212.0	5200.0	0	0	0
49.9	16.1	213.0	5400.0	0	0	1

rows × 7 columns

▼ Train Test Split



▼ Decision Tree Classifier

Decision Tree Classifier

```
1 from sklearn.tree import DecisionTreeClassifier
```
✓ 0.9s

```
1 model = DecisionTreeClassifier()
```
✓ 0.1s

```
1 model.fit(X_train, y_train)
```
✓ 0.1s

```
DecisionTreeClassifier()
```

```
1 base_pred = model.predict(X_test)
```
✓ 0.1s

```
1 base_pred
```
✓ 0.9s

```
array(['Chinstrap', 'Gentoo', 'Adelie', 'Chinstrap', 'Gentoo',
       'Chinstrap', 'Adelie', 'Gentoo', 'Chinstrap', 'Gentoo', 'Adelie',
       'Adelie', 'Adelie', 'Gentoo', 'Gentoo', 'Adelie', 'Gentoo',
       'Adelie', 'Adelie', 'Adelie', 'Gentoo', 'Chinstrap', 'Adelie',
       'Adelie', 'Adelie', 'Adelie', 'Chinstrap', 'Gentoo', 'Adelie',
       'Chinstrap', 'Gentoo', 'Adelie', 'Gentoo', 'Adelie', 'Adelie',
       'Chinstrap', 'Adelie', 'Gentoo', 'Chinstrap', 'Gentoo', 'Adelie',
       'Adelie', 'Gentoo', 'Adelie', 'Adelie', 'Chinstrap', 'Chinstrap',
       'Chinstrap', 'Chinstrap', 'Chinstrap', 'Adelie', 'Adelie',
       'Gentoo', 'Gentoo', 'Adelie', 'Adelie', 'Chinstrap', 'Chinstrap',
       'Gentoo', 'Adelie', 'Chinstrap', 'Gentoo', 'Adelie', 'Adelie',
       'Chinstrap', 'Gentoo', 'Chinstrap', 'Chinstrap', 'Gentoo',
       'Gentoo', 'Gentoo', 'Gentoo', 'Gentoo', 'Gentoo', 'Gentoo',
       'Gentoo', 'Gentoo', 'Gentoo', 'Adelie', 'Gentoo', 'Adelie',
```

▼ Evaluation

Evaluation

```
1 from sklearn.metrics import classification_report, plot_confusion_matrix
```

✓ 0.1s

```
1 print(classification_report(y_test, base_pred))
```

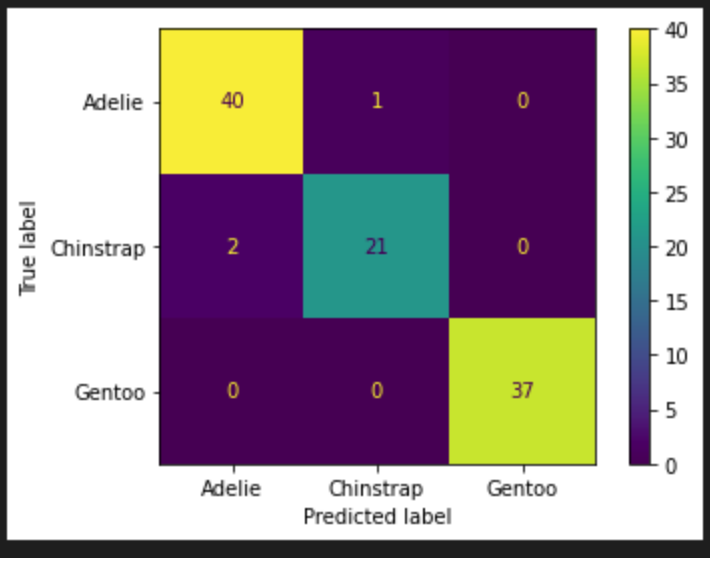
✓ 0.3s

```
              precision    recall  f1-score   support

      Adelie       0.95      0.98      0.96        41
   Chinstrap       0.95      0.91      0.93        23
      Gentoo       1.00      1.00      1.00        37

    accuracy                           0.97       101
   macro avg       0.97      0.96      0.97       101
weighted avg       0.97      0.97      0.97       101
```

```
1 plot_confusion_matrix(model,X_test,y_test)
```

✓ 1.1s

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMat
```



```
1  model.feature_importances_
```
✓ 0.9s

```
array([0.33754639, 0.05221421, 0.542054  , 0.        , 0.0681854 ,
       0.        , 0.        ])
```

```
1  X.columns
```
✓ 0.1s

```
Index(['culmen_length_mm', 'culmen_depth_mm', 'flipper_length_mm',
       'body_mass_g', 'island_Dream', 'island_Torgersen', 'sex_MALE'],
      dtype='object')
```

```
1  pd.DataFrame(index=X.columns, data=model.feature_importances_,
2  columns=["Feature Importance"]).sort_values("Feature Importance")
3  # .sort_values("sıralanacak olan değer") <<< şeklinde yazılır
```
✓ 0.1s

	Feature Importance
body_mass_g	0.000000
island_Torgersen	0.000000
sex_MALE	0.000000
culmen_depth_mm	0.052214
island_Dream	0.068185
culmen_length_mm	0.337546
flipper_length_mm	0.542054

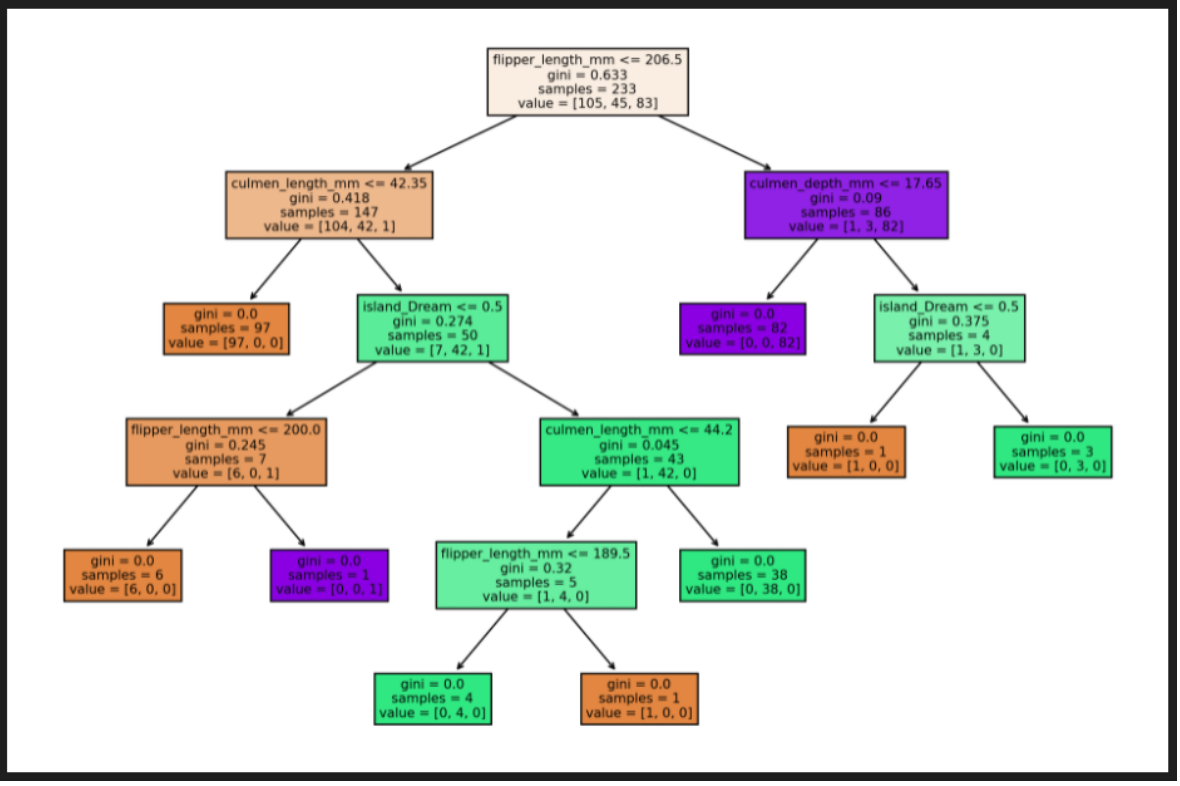
▼ Tree Vizualisation

```
from sklearn.tree import plot_tree
```

0.1s

```
plt.figure(figsize=(12,8), dpi=200)
plot_tree(model, feature_names=X.columns, filled=True);
```

1.2s

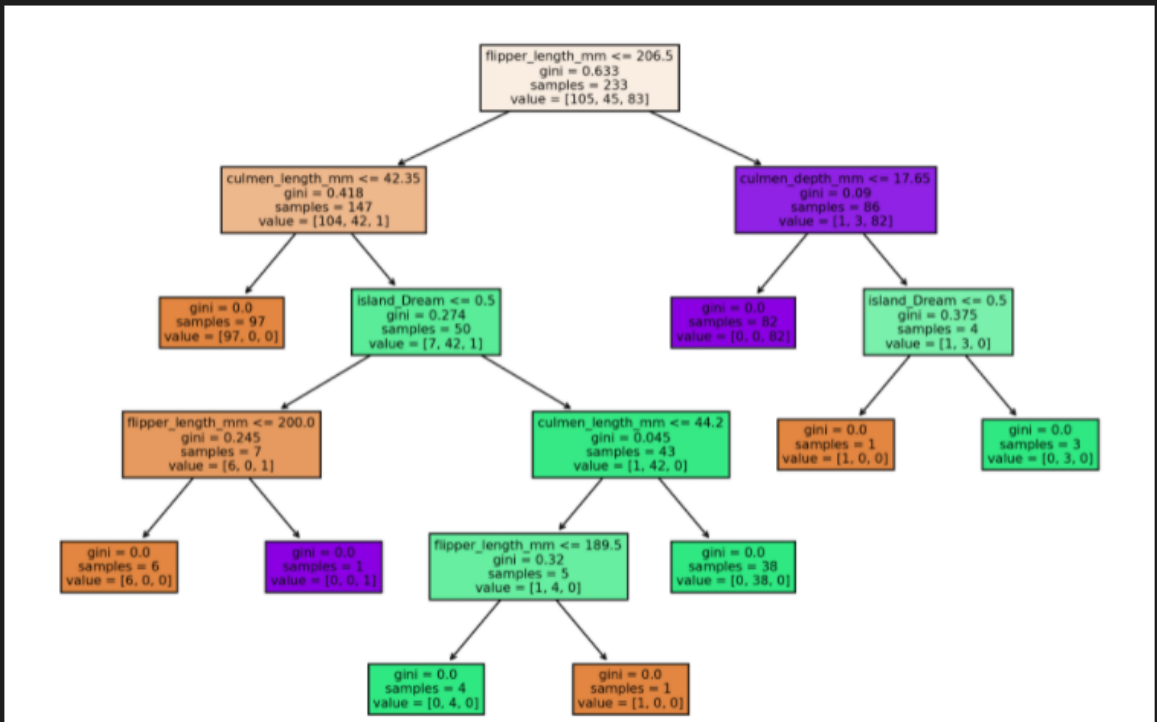



```
def report_model(model):
    model_preds = model.predict(X_test)
    print(classification_report(y_test, model_preds))
    print("\n")
    plt.figure(figsize=(12,8), dpi=200)
    plot_tree(model, feature_names=X.columns, filled=True);
```

0.5s

```
1  report_model(model)
✓ 1.2s                                                    Python
```

```
              precision    recall  f1-score   support

      Adelie       0.95      0.98      0.96        41
   Chinstrap       0.95      0.91      0.93        23
      Gentoo       1.00      1.00      1.00        37

    accuracy                           0.97       101
   macro avg       0.97      0.96      0.97       101
weighted avg       0.97      0.97      0.97       101
```



- iki basamaklı ama daha hızlı fakat nispeden daha az doğruluk oranına sahip
- burada TRADE yap

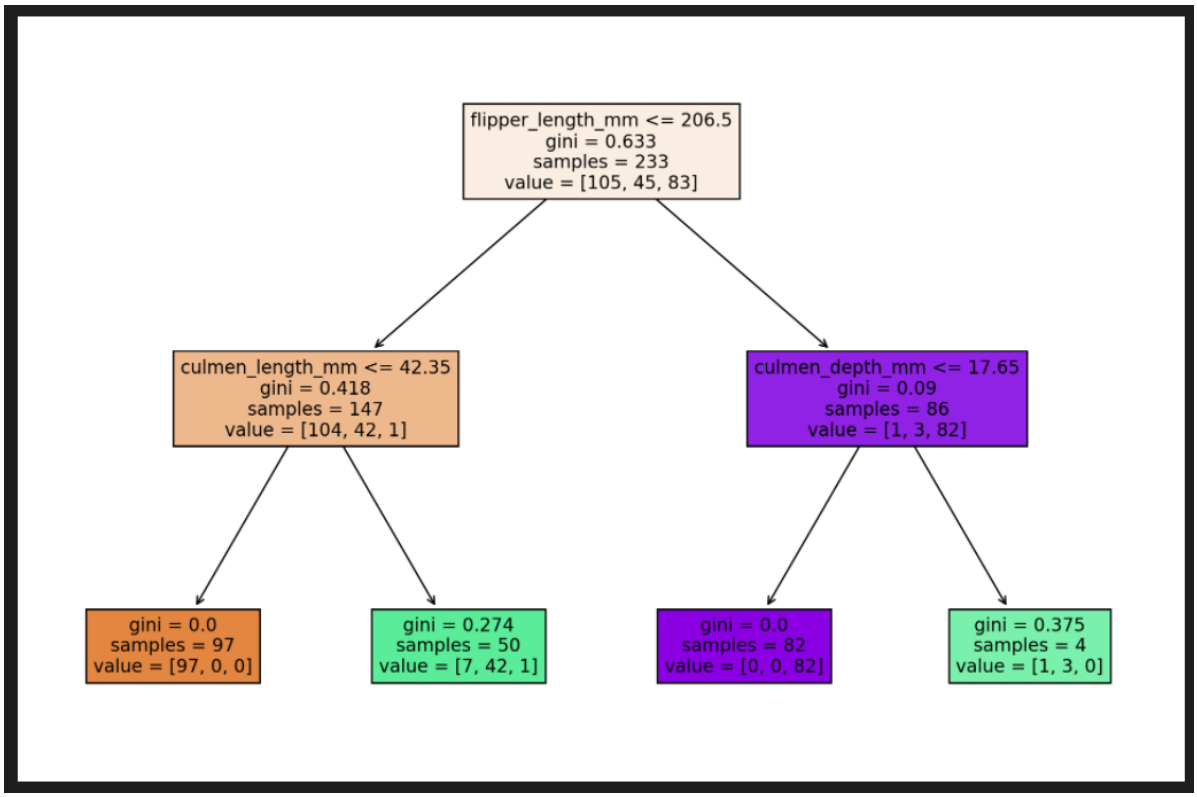
```
1  pruned_tree = DecisionTreeClassifier(max_depth=2)
```
✓ 0.5s

```
1  pruned_tree.fit(X_train, y_train)
```
✓ 0.6s

```
DecisionTreeClassifier(max_depth=2)
```

```
1  report_model(pruned_tree)
```
✓ 0.7s

```
              precision    recall  f1-score   support

      Adelie       0.97      0.88      0.92        41
   Chinstrap       0.81      0.96      0.88        23
      Gentoo       1.00      1.00      1.00        37

    accuracy                           0.94       101
   macro avg       0.93      0.94      0.93       101
weighted avg       0.95      0.94      0.94       101
```



- max leaf sayısı 3 olarak ayarlanırsa

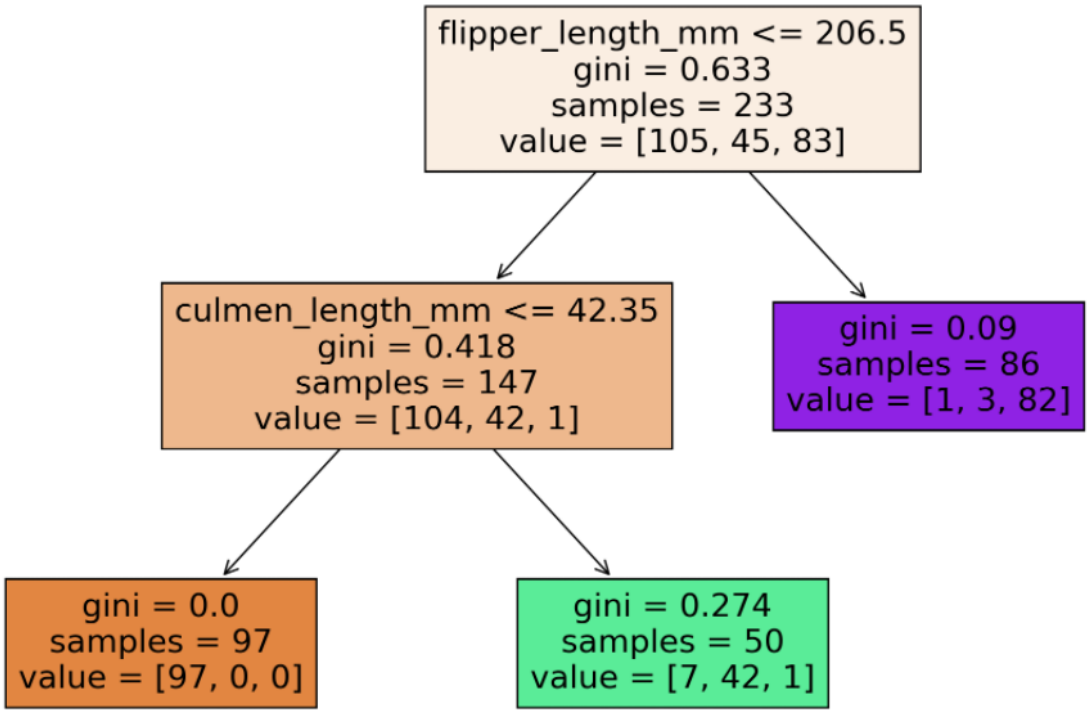
```
1  max_leaf_tree = DecisionTreeClassifier(max_leaf_nodes=3)
```
✓ 0.4s

```
1  max_leaf_tree.fit(X_train, y_train)
```
✓ 0.4s

```
DecisionTreeClassifier(max_leaf_nodes=3)
```

```
1  report_model(max_leaf_tree)
```
✓ 0.5s

```
              precision    recall  f1-score   support

      Adelie       0.97      0.88      0.92        41
   Chinstrap       0.83      0.87      0.85        23
      Gentoo       0.93      1.00      0.96        37

    accuracy                           0.92       101
   macro avg       0.91      0.92      0.91       101
weighted avg       0.92      0.92      0.92       101
```



- entropy tree

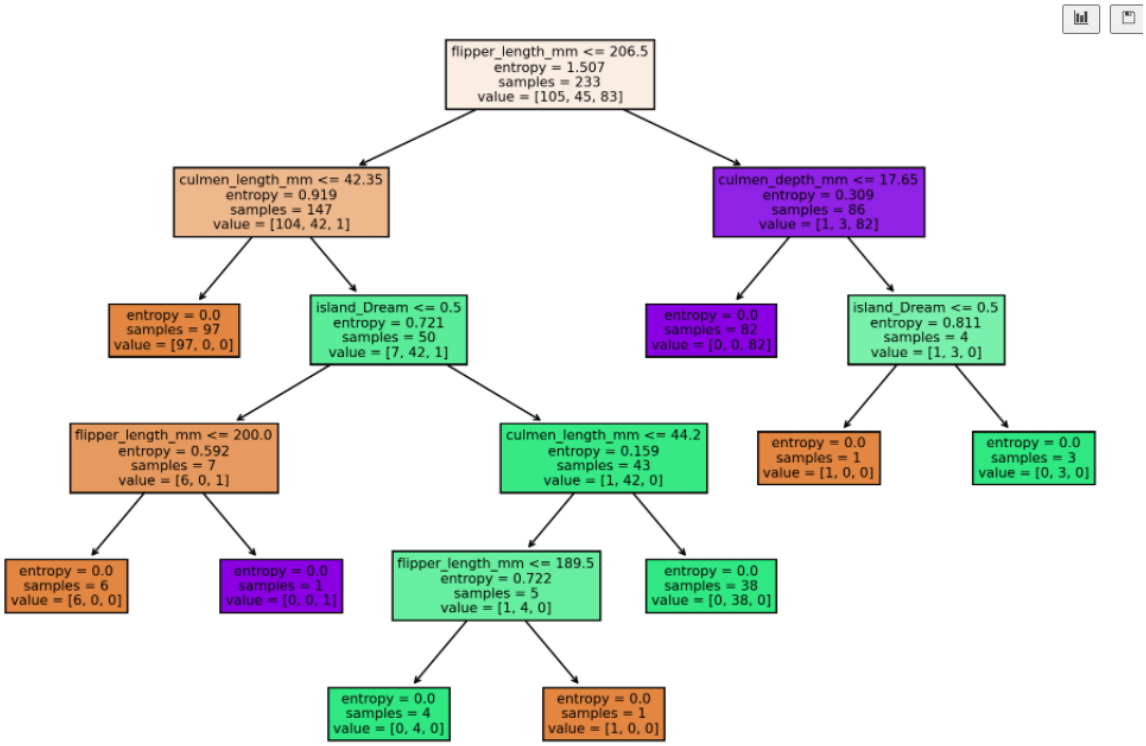
```
1  entropy_tree = DecisionTreeClassifier(criterion="entropy")
```
✓ 0.3s

```
1  entropy_tree.fit(X_train,y_train)
```
✓ 0.1s

```
DecisionTreeClassifier(criterion='entropy')
```

```
1  report_model(entropy_tree)
```
✓ 1.1s

```
              precision    recall  f1-score   support

      Adelie       0.95      0.98      0.96        41
   Chinstrap       0.95      0.91      0.93        23
      Gentoo       1.00      1.00      1.00        37

    accuracy                           0.97       101
   macro avg       0.97      0.96      0.97       101
weighted avg       0.97      0.97      0.97       101
```


flipper_length_mm <= 206.5
entropy = 1.507
samples = 233
value = [105, 45, 83]
culmen_length_mm <= 42.35
entropy = 0.919
samples = 147
value = [104, 42, 1]
culmen_depth_mm <= 17.65
entropy = 0.309
samples = 86
value = [1, 3, 82]
entropy = 0.0
samples = 97
value = [97, 0, 0]
island_Dream <= 0.5
entropy = 0.721
samples = 50
value = [7, 42, 1]
entropy = 0.0
samples = 82
value = [0, 0, 82]
island_Dream <= 0.5
entropy = 0.811
samples = 4
value = [1, 3, 0]
flipper_length_mm <= 200.0
entropy = 0.592
samples = 7
value = [6, 0, 1]
culmen_length_mm <= 44.2
entropy = 0.159
samples = 43
value = [1, 42, 0]
entropy = 0.0
samples = 1
value = [1, 0, 0]
entropy = 0.0
samples = 3
value = [0, 3, 0]
entropy = 0.0
samples = 6
value = [6, 0, 0]
entropy = 0.0
samples = 1
value = [0, 0, 1]
flipper_length_mm <= 189.5
entropy = 0.722
samples = 5
value = [1, 4, 0]
entropy = 0.0
samples = 38
value = [0, 38, 0]
entropy = 0.0
samples = 4
value = [0, 4, 0]
entropy = 0.0
samples = 1
value = [1, 0, 0]