The Decision Tree of Cat Wisdom

Classification Trees

Prompts

- Train a Classification Tree with "yammy" as the target
- Plot the Decision Tree

Strategy

- Prepare the data
- Train the Classification Tree
- Check the Accuracy
- Visualize the Decision Tree

Code & Results

1. Prepare the data

This code checks the "star_rating" and puts a 1 in the new "yammy" column if it is greater than or equal to 4.5.

```
import pandas as pd

# Assuming you already have a DataFrame named df
data['yammy'] = (data['star_rating'] >= 4.5).astype(int)
data.head()
```

This code puts features between 0 and 1.

```
from sklearn.preprocessing import MinMaxScaler

# Select only numeric columns (excluding 'yammy' if it's a label)
features = data.drop(columns=['yammy'])  # Drop target column if needed
numeric_cols = features.select_dtypes(include=['float64', 'int64']).columns

scaler = MinMaxScaler()
data[numeric_cols] = scaler.fit_transform(data[numeric_cols])

data[numeric_cols].head()
```

This code first splits the teacher from the features and then splits the data into training and testing sets.

```
from sklearn.model_selection import train_test_split

# Separate features and target
X = data.drop(columns=['yammy','star_rating'])
y = data['yammy']

# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

2. Train the Model

This code creates a classification tree and trains it using the training set.

```
from sklearn.tree import DecisionTreeClassifier, plot_tree

# Initialize and train classifier
tree_clf = DecisionTreeClassifier(max_depth=4, random_state=42)  # limit depth to prevent overfitting
tree_clf.fit(X_train, y_train)
```

3. Check the Accuracy & Confusion Matrix

This code evaluates model accuracy and calculates a confusion matrix.

```
y_pred = tree_clf.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

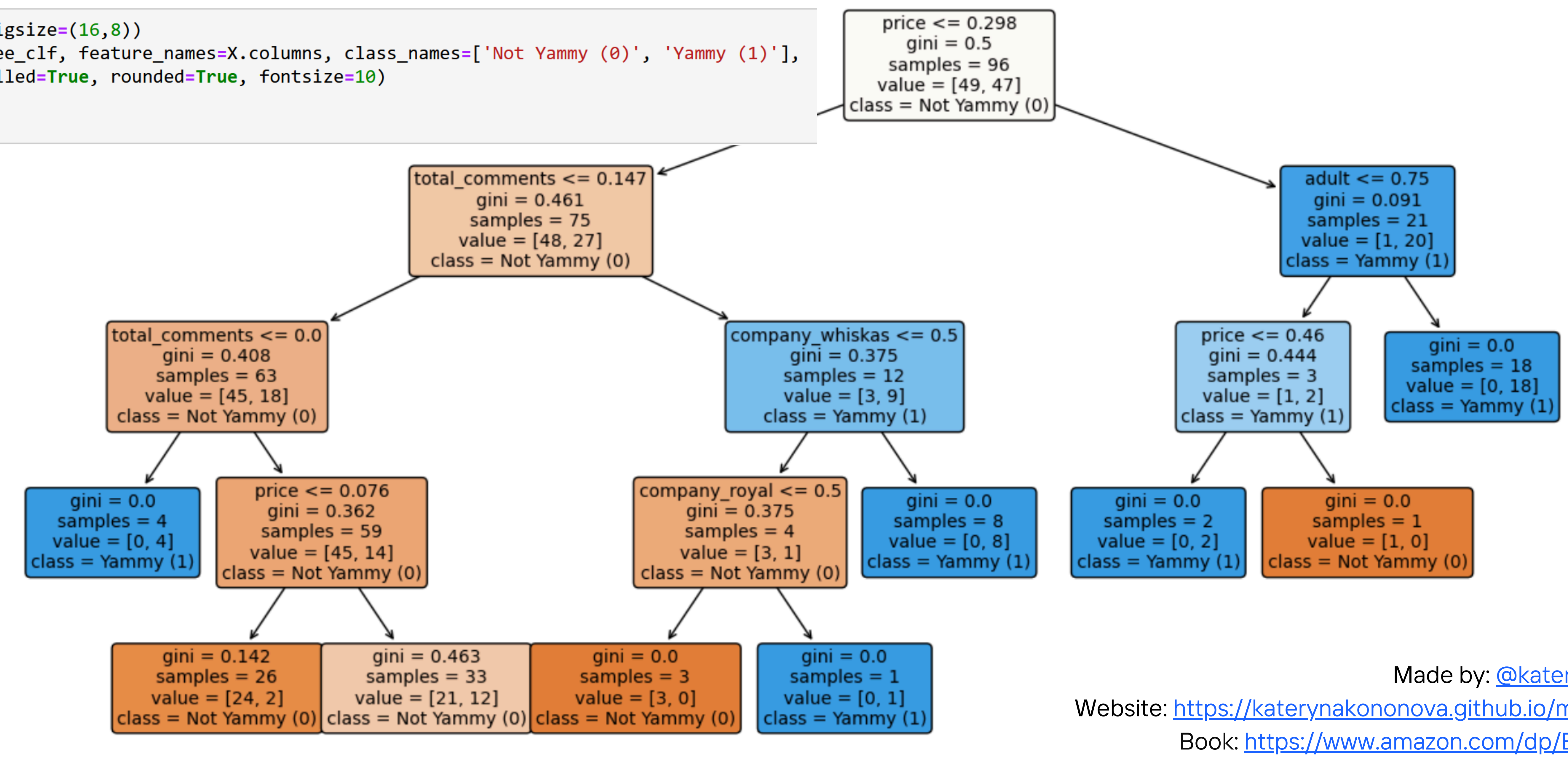
```
Accuracy: 0.64
Confusion Matrix:
 [[12  3]
 [ 6  4]]
```

✅ The model got 64% of the tins right.
❌ It mislabeled 3 out of 15 meh tins as yammy
❌ It mislabeled 6 out of 10 yammy tins as meh

4. Visualize the Decision Tree

This code visualizes the decision tree.

```
plt.figure(figsize=(16,8))
plot_tree(tree_clf, feature_names=X.columns, class_names=['Not Yammy (0)', 'Yammy (1)'],
          filled=True, rounded=True, fontsize=10)
plt.show()
```



price <= 0.298 / gini = 0.5 / samples = 96 / value = [49, 47] / class = Not Yammy (0)
total_comments <= 0.147 / gini = 0.461 / samples = 75 / value = [48, 27] / class = Not Yammy (0)
adult <= 0.75 / gini = 0.091 / samples = 21 / value = [1, 20] / class = Yammy (1)
total_comments <= 0.0 / gini = 0.408 / samples = 63 / value = [45, 18] / class = Not Yammy (0)
company_whiskas <= 0.5 / gini = 0.375 / samples = 12 / value = [3, 9] / class = Yammy (1)
price <= 0.46 / gini = 0.444 / samples = 3 / value = [1, 2] / class = Yammy (1)
gini = 0.0 / samples = 18 / value = [0, 18] / class = Yammy (1)
gini = 0.0 / samples = 4 / value = [0, 4] / class = Yammy (1)
price <= 0.076 / gini = 0.362 / samples = 59 / value = [45, 14] / class = Not Yammy (0)
company_royal <= 0.5 / gini = 0.375 / samples = 4 / value = [3, 1] / class = Not Yammy (0)
gini = 0.0 / samples = 8 / value = [0, 8] / class = Yammy (1)
gini = 0.0 / samples = 2 / value = [0, 2] / class = Yammy (1)
gini = 0.0 / samples = 1 / value = [1, 0] / class = Not Yammy (0)
gini = 0.142 / samples = 26 / value = [24, 2] / class = Not Yammy (0)
gini = 0.463 / samples = 33 / value = [21, 12] / class = Not Yammy (0)
gini = 0.0 / samples = 3 / value = [3, 0] / class = Not Yammy (0)
gini = 0.0 / samples = 1 / value = [0, 1] / class = Yammy (1)

Made by: @katerynakononova
Website: https://katerynakononova.github.io/meowlearning/
Book: https://www.amazon.com/dp/B0CW9SFYXF