# Decision Trees

Concept Module 9

Template by: Laurent Lessard

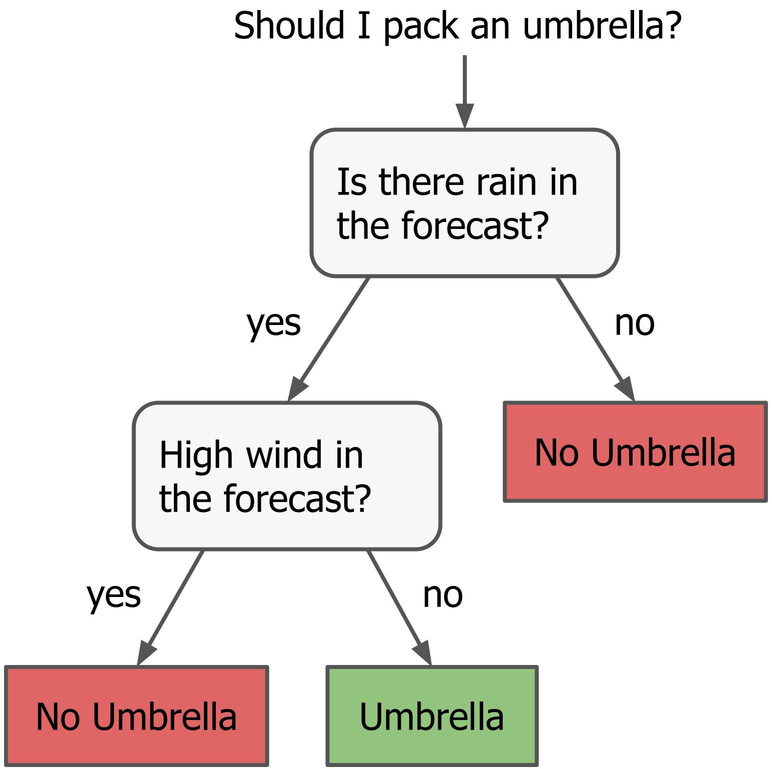
# What is a decision tree?

A **decision tree** is a classifier.

Similar to K-Nearest-Neighbors (KNN):

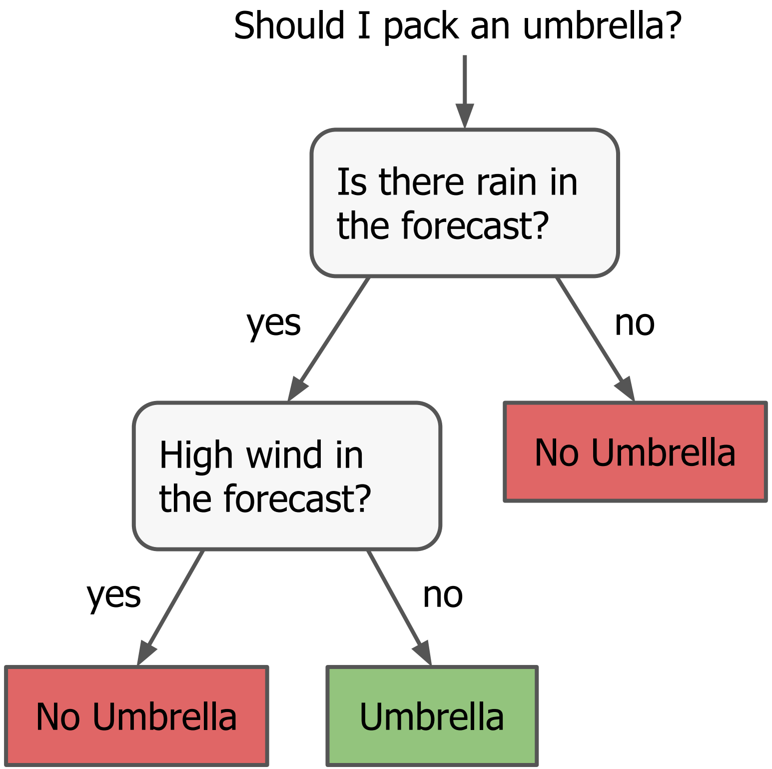
- Trained on labeled data.
- Predicts the labels of the (unlabeled) test data.
- Has its own set of parameters to be chosen (like K in KNN)

# What is a decision tree?




- The single top node is called the root node
- The terminal nodes are called leaf nodes
- Each non-leaf node has exactly two children
- Start at the root and work your way to a leaf

# What is a decision tree?




Labeled data corresponding to this decision tree:

| | wind? | rain? | cold? | umbrella? |
|---|---|---|---|---|
| **0** | yes | yes | no | no |
| **1** | no | yes | yes | yes |
| **2** | yes | no | yes | no |
| **3** | no | yes | no | yes |
| **4** | yes | yes | yes | no |
| **5** | no | no | no | no |

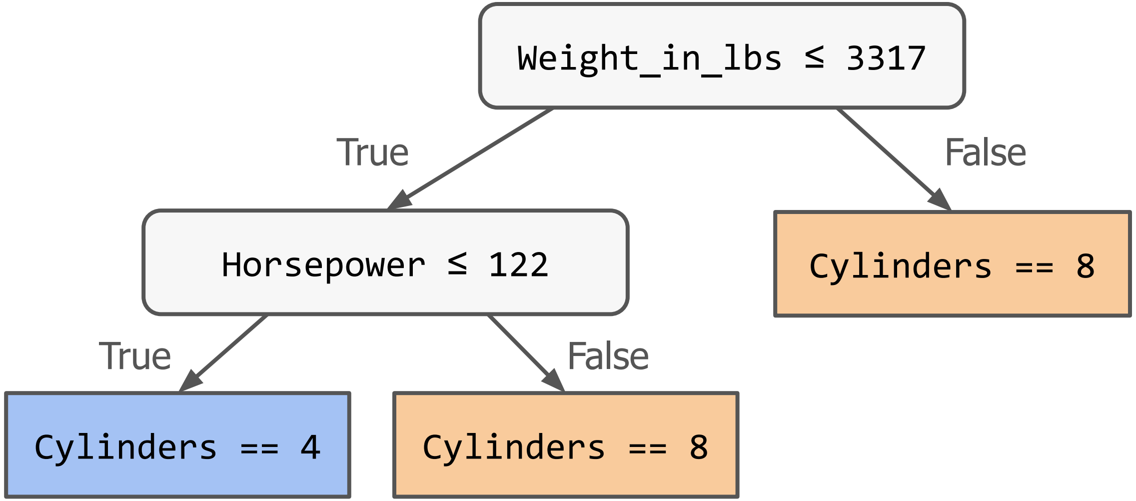
# Why decision trees?

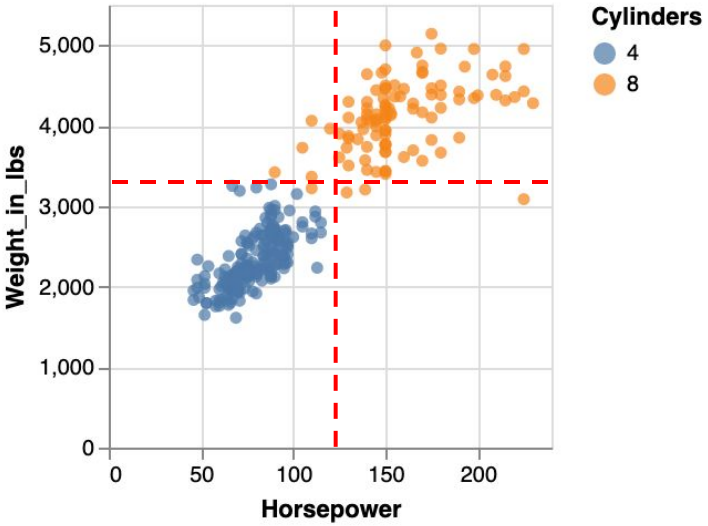
- Decision trees mirror a human-like decision process.
- Leads to a highly interpretable classifier.
- Works with categorical, ordinal, and numerical features.
- It can perform very well!

Contests won by a particular decision tree implementation:
https://github.com/dmlc/xgboost/blob/master/demo/README.md

# Example: car gas mileage

**Goal:** predict number of cylinders by making decisions on car weight and horsepower.

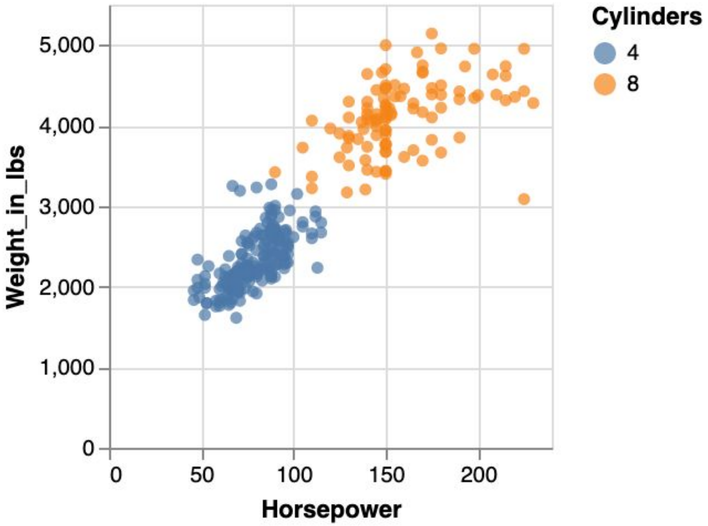



The **depth** of a tree is the longest path from the root to any of the leaf nodes. This tree has a depth of 2.

# How does it work?

1. Pick the feature that separates the data into classes the most accurately/effectively.
2. Create a node for that feature. Repeat process on each child.
3. Stop when…
   a. node contain exactly one class (make it a leaf!)
   b. tree has reached the maximum allowable depth



# Decision Trees in Python

```
from sklearn.tree import DecisionTreeClassifier

dtree = DecisionTreeClassifier(max_depth=2)
dtree.fit(Xtrain,Ytrain)
```

Create decision tree using labeled training data.

Max tree depth

Prediction

```
# predict labels of test data
Ypred = dtree.predict(Xtest)

# measure accuracy
from sklearn.metrics import accuracy_score
accuracy_score(Ytest,Ypred)
```

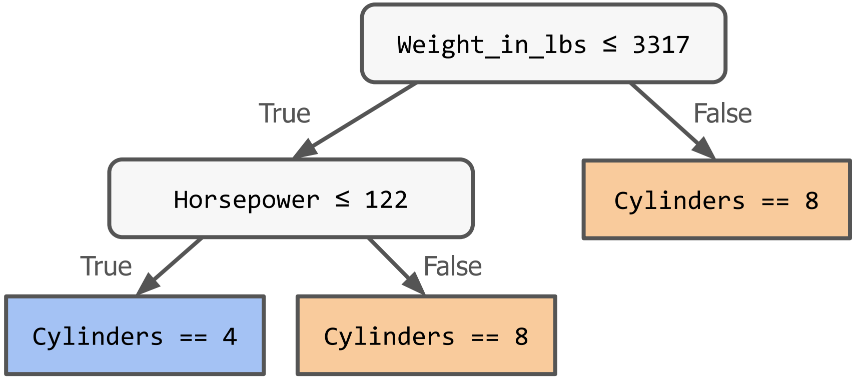
`(Xtrain,Ytrain)` and `(Xtest,Ytest)` are defined in a similar way to when we used K-Nearest-Neighbors.

# Feature importance

```python
#Calculate feature importance for this tree
dtree.feature_importances_
```

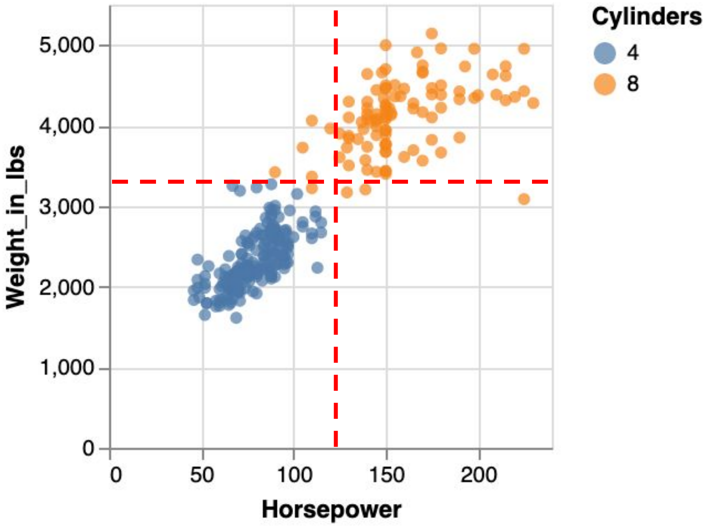
- Array [ $i_1$, $i_2$, …, $i_N$ ] of importances for features 1,…,N.
- Importances are nonnegative and sum to 1.
- Measures how involved/influential each feature is. Depends on various factors!
- Importance of zero means the feature is not used at all.

# Feature importance




```
X = df[["Horsepower", "Weight_in_lbs"]]
y = df["Cylinders"]
dtree = DecisionTreeClassifier(max_depth=2)
dtree.fit(X,y)
dtree.feature_importances_
```

```
array([0.04375543, 0.95624457])
```




Importance of a feature can depend strongly on the tree!

In this example, the feature picked as a root node will get a very high importance score.

# Warning!

Do not include the variable to predict in the training data!

It leads to a *very* short tree that only makes one decision:

```
from sklearn.tree import DecisionTreeClassifier

est = DecisionTreeClassifier(max_depth=2)
est.fit(df, df["Cylinders"])
est.feature_importances_
```

```
array([0., 0., 1.])
```

| | Horsepower | Weight_in_lbs | Cylinders |
|---|---|---|---|
| **387** | 70.0 | 2125 | 4 |
| **247** | 78.0 | 2190 | 4 |
| **354** | 65.0 | 1975 | 4 |
| **97** | 198.0 | 4952 | 8 |
| **327** | 92.0 | 2434 | 4 |

# What else can go wrong?

- Using `max_depth` that is too large can lead to overfitting. We will discuss this concept soon!
- There are lots of parameters you can set:
  `criterion, splitter, max_depth, min_samples_split, min_samples_leaf, min_weight_fraction_leaf, max_features, max_leaf_nodes, min_impurity_decrease, min_impurity_split,…`
- Sometimes the decision tree isn’t intuitive at all.

# Summary

- Decision Trees consist of nodes that ask yes/no questions about features. The leaf nodes contain label predictions.
- A useful parameter is the maximum tree depth. But there are many other parameters!
- Feature Importance is a way of gauging how useful a feature is in the decision tree.