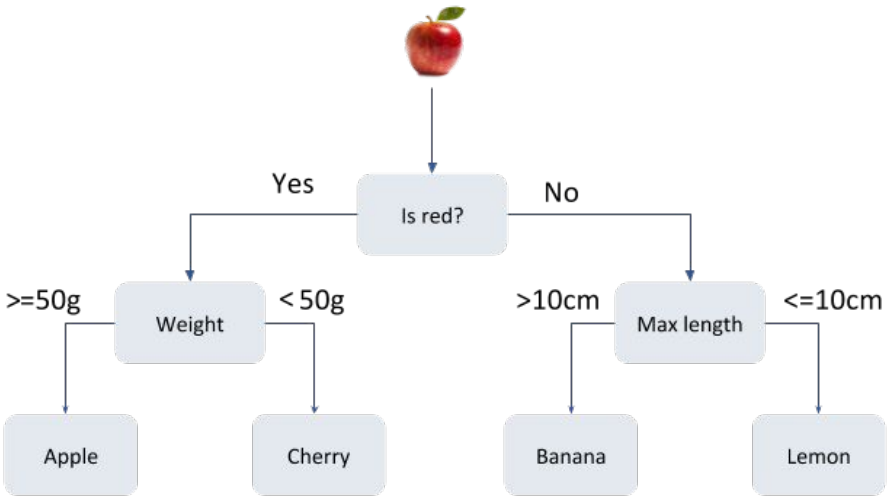
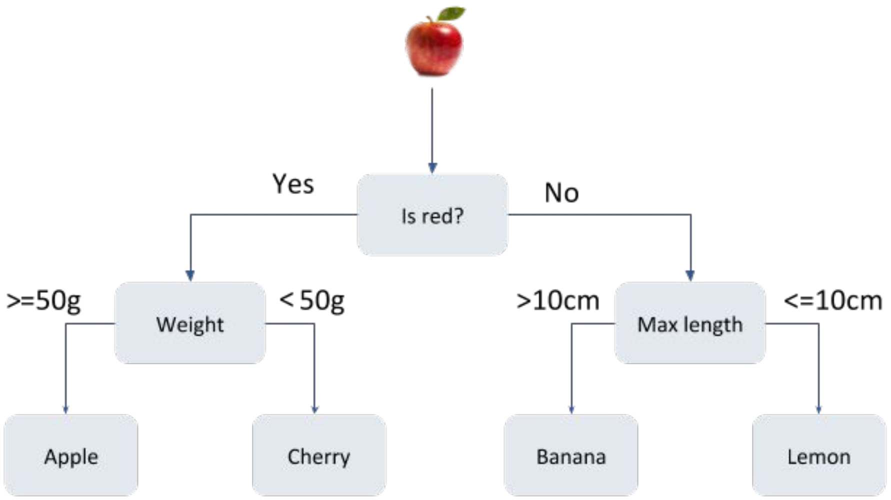
Decision Trees

Decision Trees



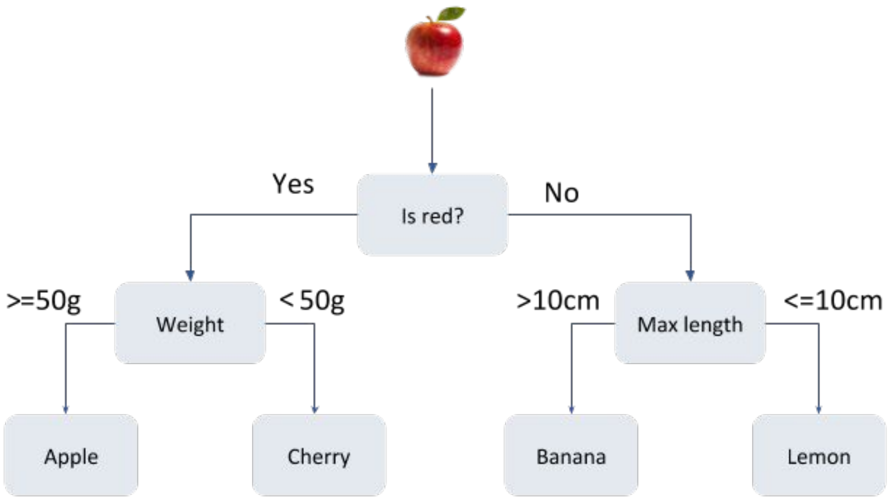
- A **hierarchical** structure

Decision Trees



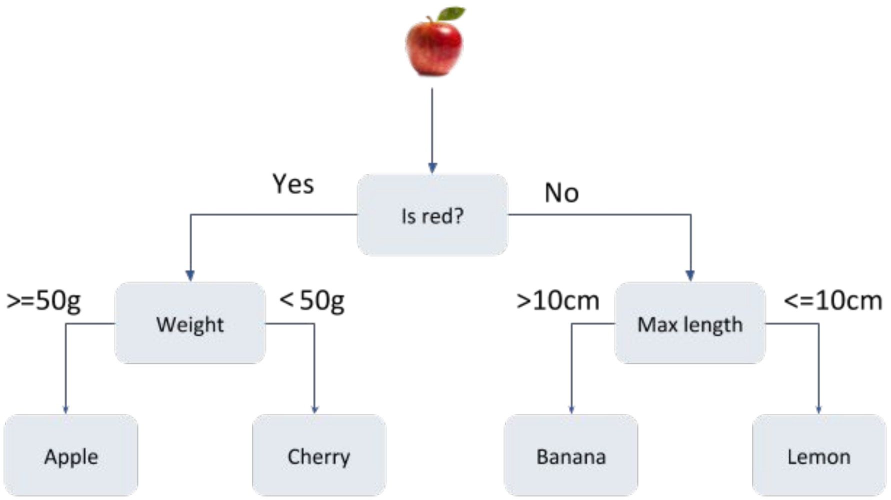
- A **hierarchical** structure
- Each **node** corresponds to a **binary rule**

Decision Trees



- A **hierarchical** structure
- Each **node** corresponds to a **binary rule**
- Only **one feature** is considered per node

Decision Trees



- A **hierarchical** structure
- Each **node** corresponds to a **binary rule**
- Only **one feature** is considered per node
- The end of a decision path is called a **leaf**

CAMBRIDGE SPARK

Decision Tree Classifier (DTC)

How can we fit a DTC?

1. Start from the **first node**. For each feature find the splitting point that **best separates** the two classes.

CAMBRIDGE SPARK

Decision Tree Classifier (DTC)

How can we fit a DTC?

1. Start from the **first node**. For each feature find the splitting point that **best separates** the two classes.
2. Take the **feature** and the **splitting point** that offers the best separation

Decision Tree Classifier (DTC)

How can we fit a DTC?

1. Start from the **first node**. For each feature find the splitting point that **best separates** the two classes.
2. Take the **feature** and the **splitting point** that offers the best separation
3. Move to **next** node

Decision Tree Classifier (DTC)

How can we fit a DTC?

1. Start from the **first node**. For each feature find the splitting point that **best separates** the two classes.
2. Take the **feature** and the **splitting point** that offers the best separation
3. Move to **next** node

Two remaining issues:

- How do we quantify the “best separation”?
- When do we stop creating nodes?

CAMBRIDGE SPARK

Quantifying the best separation of classes

Different metrics can be used to measure “separation”.

Common ones are:

- GINI impurity
- Information gain

Quantifying the best separation of classes

Different metrics can be used to measure “separation”.

Common ones are:

- GINI impurity
- Information gain

Broadly speaking they lead to very similar results, we will focus on GINI impurity.

GINI Impurity

- Assume there are $\boldsymbol{k}$ classes: 1, 2, …, k
- At a given node P, there is a proportion $\boldsymbol{p_1}$ of elements in class 1, $\boldsymbol{p_2}$ of elements in class 2, etc...

CAMBRIDGE SPARK

GINI Impurity

- Assume there are ***k*** classes: 1, 2, …, *k*
- At a given node P, there is a proportion $\boldsymbol{p_1}$ of elements in class 1, $\boldsymbol{p_2}$ of elements in class 2, etc...
- GINI Impurity at that node:

$$G(P) = 1 - \sum_{i=1}^{k} p_i^2$$

CAMBRIDGE SPARK

GINI Impurity - An example

For binary classification, $\boldsymbol{k = 2}$. And at a given node $\mathbf{G(P) = 1 - (p_1^2 + p_2^2)}$

GINI Impurity - An example

For binary classification, $\boldsymbol{k = 2}$. And at a given node $\mathbf{G(P) = 1 - (p_1^2 + p_2^2)}$

So that, If P is a **pure node** with only one class represented, say class 1:

$$p_1=1,\ p_2=0 \quad \text{so: } \mathbf{G(P) = 0}$$

CAMBRIDGE SPARK

GINI Impurity - An example

For binary classification, $\boldsymbol{k}$ **= 2**. And at a given node $\mathbf{G(P) = 1 - (p_1^2 + p_2^2)}$

So that, If P is a **pure node** with only one class represented, say class 1:

p_1=1, p_2=0 so: **G(P) =0**

If P is **impure** and has 50% of each class:

p_1 = ½, p_2 = ½ so: **G(P) = .5**

That can be interpreted as the likelihood of misclassification of a new point at this node if we classify it at random.

CAMBRIDGE SPARK

Fitting a tree with GINI Impurity

- At each node, the parent node ***P*** has a GINI Impurity of **G(P)**

CAMBRIDGE SPARK

Fitting a tree with GINI Impurity

- At each node, the parent node ***P*** has a GINI Impurity of **G(P)**
- We want to split it into **two children nodes** with GINI Impurity **G(C_1)** and **G(C_2)**

Fitting a tree with GINI Impurity

- At each node, the parent node ***P*** has a GINI Impurity of **G(P)**
- We want to split it into **two children nodes** with GINI Impurity **$G(C_1)$** and **$G(C_2)$**
- We pick the children for which the weighted average of **$G(C_1)$** and **$G(C_2)$** offers the largest gain compared with **G(P)**
  - That's referred to as **GINI Gain**

CAMBRIDGE SPARK

When to stop splitting?

If we keep splitting forever, we'd get one sample per leaf **(overfitting)**

- one decision path for every single sample

When to stop splitting?

If we keep splitting forever, we'd get one sample per leaf **(overfitting)**

- one decision path for every single sample

To prevent that:

- Pick a **maximum depth**
  - Controls the maximum complexity of our tree

When to stop splitting?

If we keep splitting forever, we'd get one sample per leaf **(overfitting)**

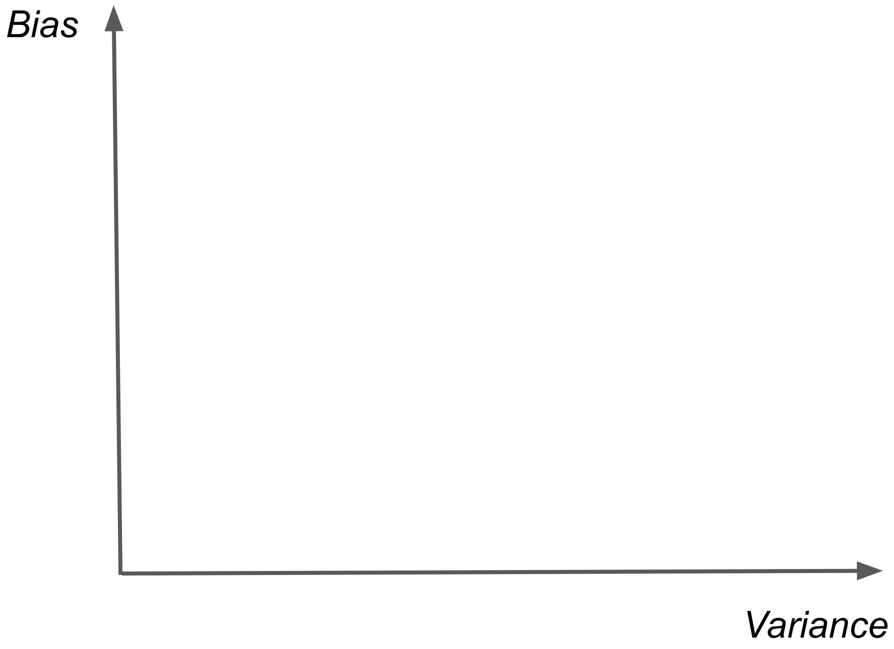
- one decision path for every single sample

To prevent that:

- Pick a **maximum depth**
  - Controls the maximum complexity of our tree
- Pick a **minimum number of samples** needed in a new node/leaf
  - Controls the overfitting to a few samples

CAMBRIDGE SPARK

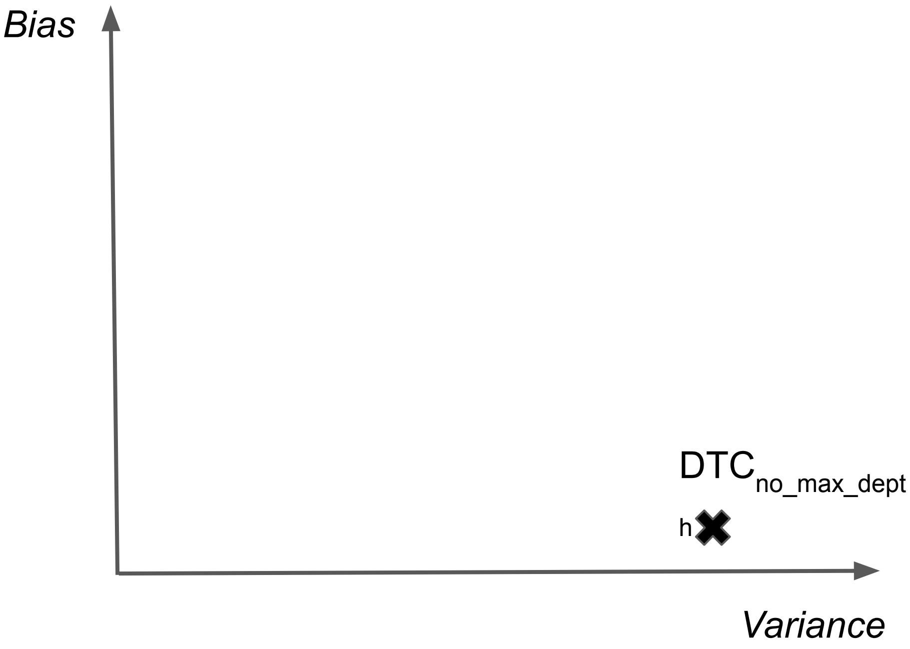
Bias vs Variance



- Where would we place a DTC with *max_depth=None*?

CAMBRIDGE SPARK

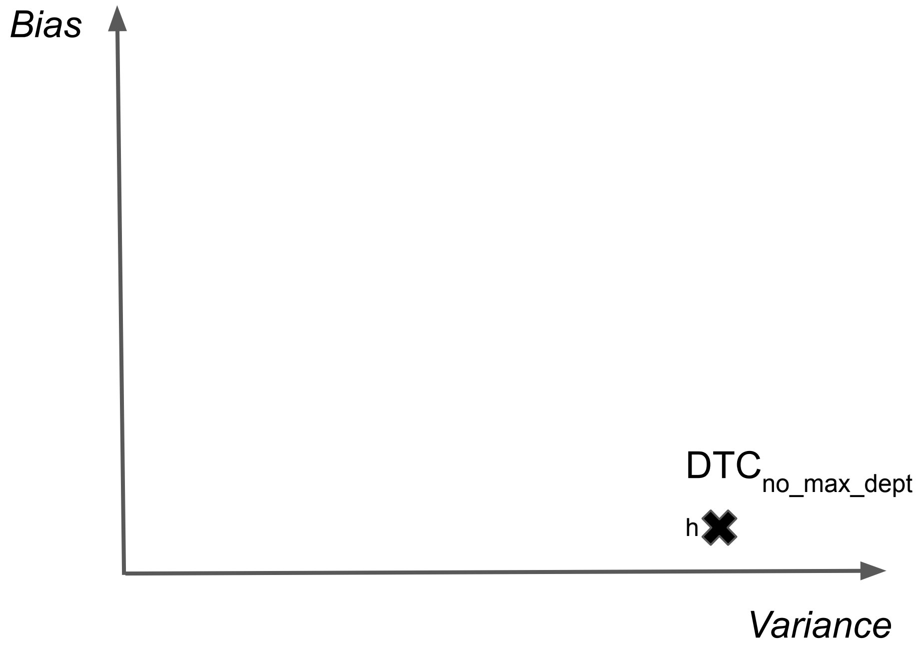
Bias vs Variance



- Where would we place a DTC with *max_depth=None*?

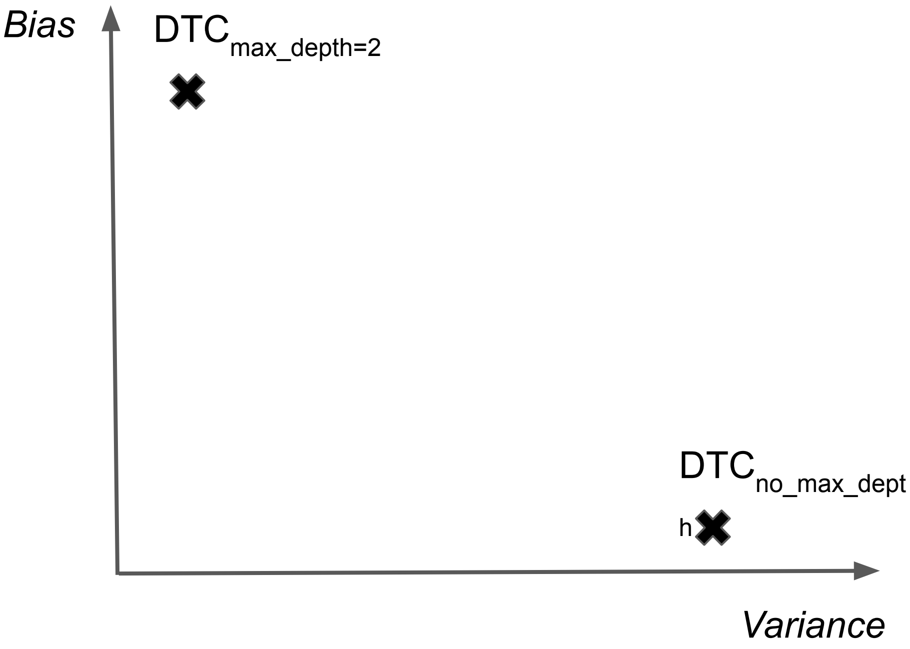
CAMBRIDGE SPARK

Bias vs Variance



- Where would we place a DTC with *max_depth=2*?

Bias vs Variance



- Where would we place a DTC with *max_depth=2*?

CAMBRIDGE SPARK

Generating predictions

Once a DTC is fitted, we can generate predictions from new data:

- Just need to follow the decision path

Decision trees, recap

Pros:

- Very easy to **interpret**

CAMBRIDGE SPARK

Decision trees, recap

Pros:

- Very easy to **interpret**
- No need to **scale** data
  - No comparisons between features

Decision trees, recap

Pros:

- Very easy to **interpret**
- No need to **scale** data
  - No comparisons between features
- It can handle **categorical** data
  - but sklearn’s implementation does not

Decision trees, recap

Pros:

- Very easy to **interpret**
- No need to **scale** data
  - No comparisons between features
- It can handle **categorical** data
  - but sklearn's implementation does not
- We can get **features' importance**
  - Computed from the average decrease in GINI Impurity of each feature

CAMBRIDGE SPARK

Decision trees, recap

Pros:

- Very easy to **interpret**
- No need to **scale** data
  - No comparisons between features
- It can handle **categorical** data
  - But sklearn's implementation does not
- We can get **features' importance**
  - Computed from the average decrease in GINI Impurity of each feature

Cons:

- Can **overfit** easily

Hands-on session

01-decision_tree.ipynb

CAMBRIDGE SPARK