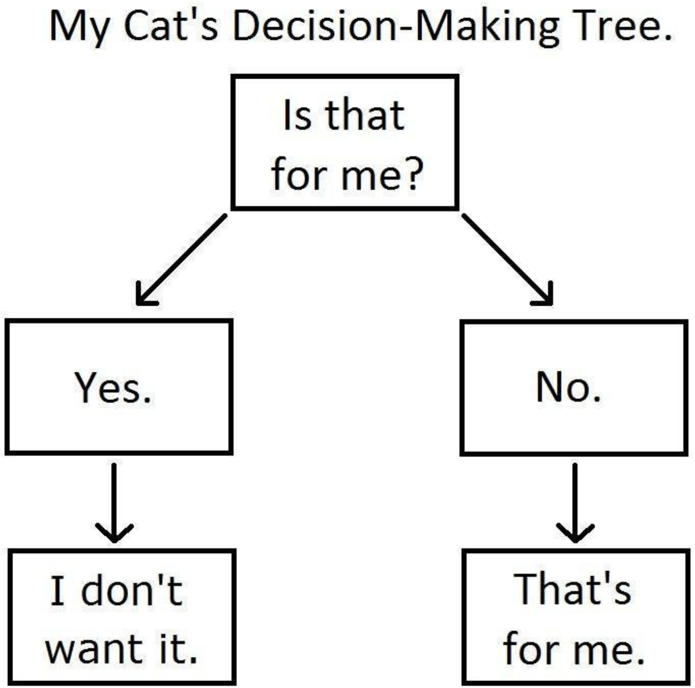
Machine Learning for Networking

Seminar 2

Marc Carrascosa Zamacois: marc.carrascosa@upf.edu

Decision trees

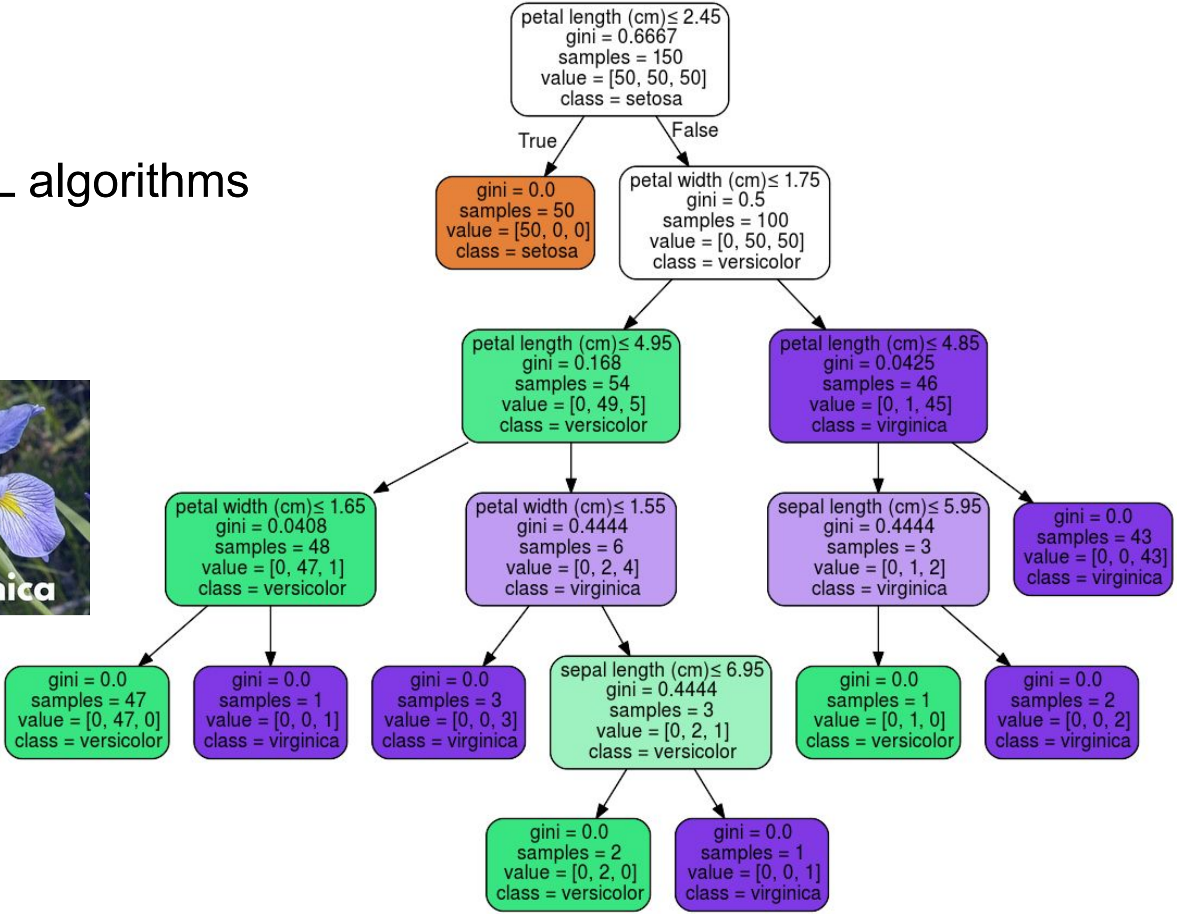
- Models that split the input data into branches with leaves at the end
- Branches are outcomes from tests
- Leaves are decisions



Decision trees

One of the most popular ML algorithms

- Easy to interpret
- Works with little data



Decision tree uses

- Can be used for regression and classification
- Regression uses Mean Square Error

$$MSE = \frac{\sum_{n=1}^{N} (\hat{y}_n - y_n)^2}{N}$$

- Classification uses Entropy

$$H(x) = -\sum_{n=1}^{N} p(x_n) \log p(x_n)$$

- Both cases maximize Information Gain

Small example

f1	f2	y
0	0	0
0	1	0
1	0	1
1	1	1

- We start by checking the entropy of y, splitting based on if y = 0 or y = 1
- Since it is uniformly distributed, the entropy is 1
- We split the dataset based on y, and then look at the other features

$$H(y) = -p(y=0)\log_2 p(y=0) - p(y=1)\log_2 p(y=1) =$$

$$= -\tfrac{2}{4}\log_2 \tfrac{2}{4} - \tfrac{2}{4}\log_2 \tfrac{2}{4} = 1$$

Small example

- Now we look at the entropy of the features based on how they impact y

$$H(\mathrm{f1} = 0, y) = -p(0,0)\log_2 p(0,0) - p(0,1)\log_2 p(0,1) =$$

$$= -\tfrac{2}{2}\log_2 \tfrac{2}{2} - \tfrac{0}{2}\log_2 \tfrac{0}{2} = 0$$

$$H(\mathrm{f1} = 1, y) = -\tfrac{0}{2}\log_2 \tfrac{0}{2} - \tfrac{2}{2}\log_2 \tfrac{2}{2} = 0$$

$$H(\mathrm{f2} = 0, y) = -\tfrac{1}{2}\log_2 \tfrac{1}{2} - \tfrac{1}{2}\log_2 \tfrac{1}{2} = 1$$

$$H(\mathrm{f2} = 1, y) = -\tfrac{1}{2}\log_2 \tfrac{1}{2} - \tfrac{1}{2}\log_2 \tfrac{1}{2} = 1$$

Small example

- Then, we calculate the weighted average of the entropy for each feature

$$H(\mathrm{f1}, y) = \frac{2}{4}H(\mathrm{f1} = 0, y) + \frac{2}{4}H(\mathrm{f1} = 1, y) = 0$$

$$H(\mathrm{f2}, y) = \frac{2}{4}H(\mathrm{f2} = 0, y) + \frac{2}{4}H(\mathrm{f2} = 1, y) = 1$$

- Finally, we calculate the information gain by subtracting each of these entropies to the initial entropy of y, and we find that there is no more information to gain once we split based on feature 1

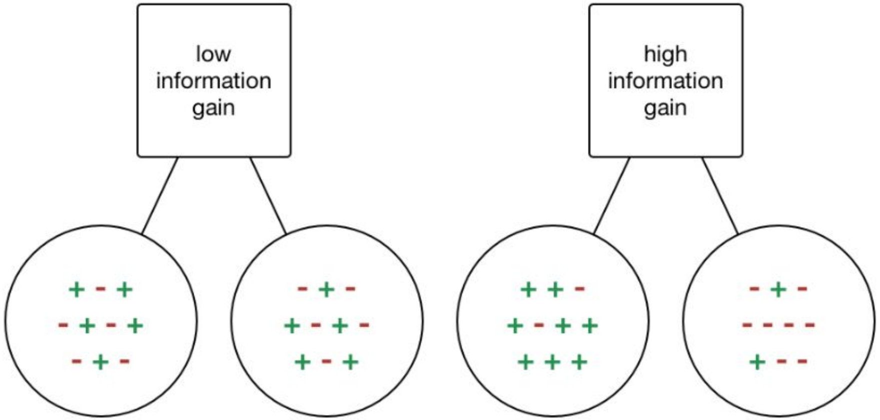
$$\mathrm{IG(y,\ f1)} = 1 - 0 = 1$$

$$\boxed{\mathrm{IG(y,\ f2)} = 1 - 1 = 0}$$

Feature 2 offers no new information!

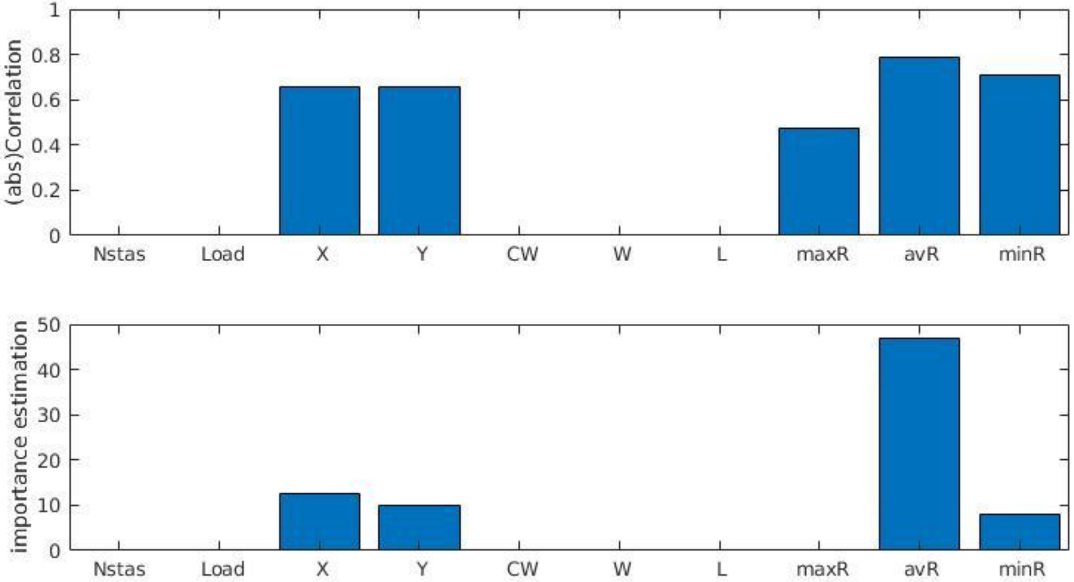
Maximizing homogeneity

- Splitting should give us more information
- Uniform distributions are avoided
- Variables are compared in terms of how much information they give
- Higher nodes in the tree have the highest impact
- At some point, we reach entropy/variance = 0 and there are no more splits

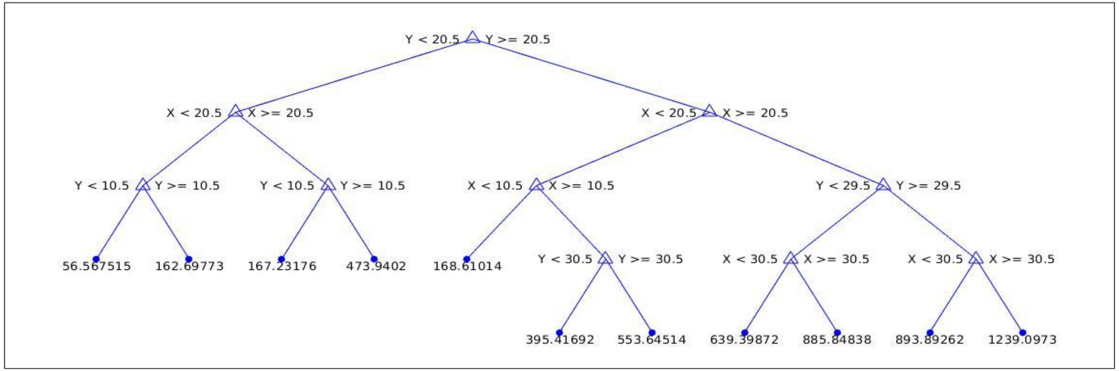


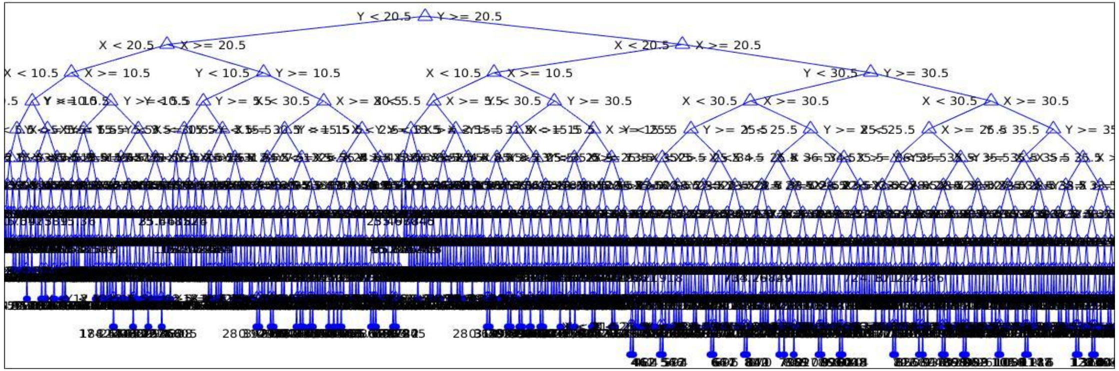
Feature importance

- Since higher nodes are more important, decision trees allow us to check feature importance easily



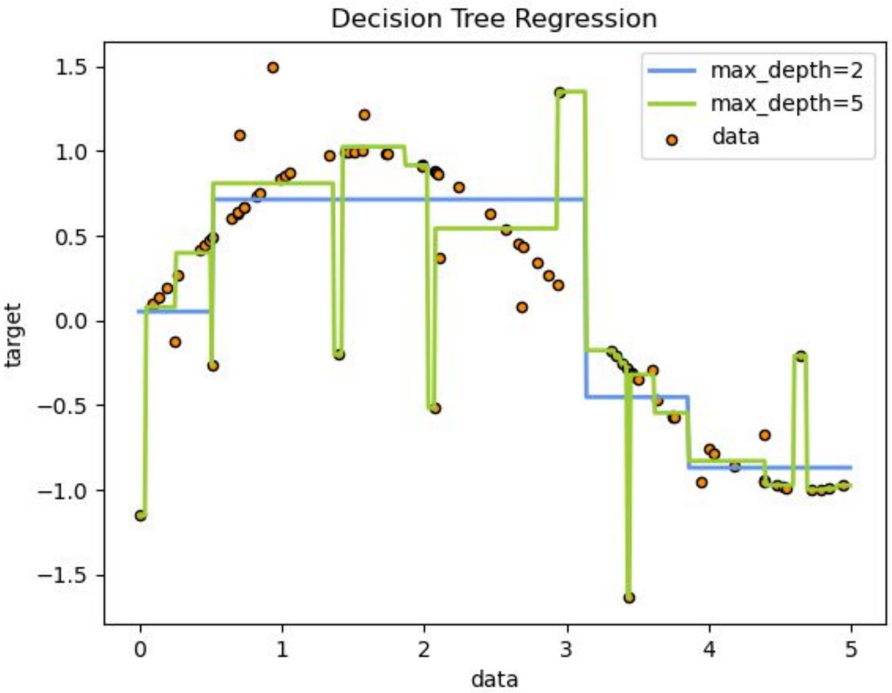
Tree depth - 10 splits vs. 889 (unlimited)





- The deeper we go, the higher our precision
- Is more depth always better?

Overfitting



- Decision trees tend to overfit
- In this example, we are trying to approximate a curve, but if we use a depth of 5, the tree learns outliers that do not represent the general shape of the line
- We want our model to generalize well
- Depth always needs to be considered

“Converting” a dataset for classification

- We take a real variable (throughput) and convert it into a binary/finite set of values
- For our purposes, if $\frac{\text{Throughput}}{\text{Load}} \geq 0.975$ we substitute the value of the throughput by 1
- Otherwise we use 0
- This way, we simplify the data set into satisfied networks and unsatisfied networks
- Error then is calculated by comparing the values directly and calculating the proportion of correct predictions

Tips

Some MATLAB:

- Fitrtree (for regression trees)
  - Model = **fitrtree**(xtrain,ytrain,'MaxNumSplits',1000)
- Predict (for predicting the testing set)
  - Yt = **predict**(xtest,ytest)
- PredictorImportance (for feature importance)
  - Imp = **predictorImportance**(Model)
- Fitctree (for classification trees)
  - Model = **fitctree**(xtrain,ytrain,'MaxNumSplits',1000)
- RMSE (regression error)
  - Err = sqrt(mean((ytest-ypredicted).^2))
- Error (classification error)
  - Err = 100-(sum(ytest==ypredicted)/length(ypredicted)*100)
- View (to see your decision tree)
  - **view**(Model,'mode','graph')
- Bar (for bar graphs)
  - **bar**(imp)