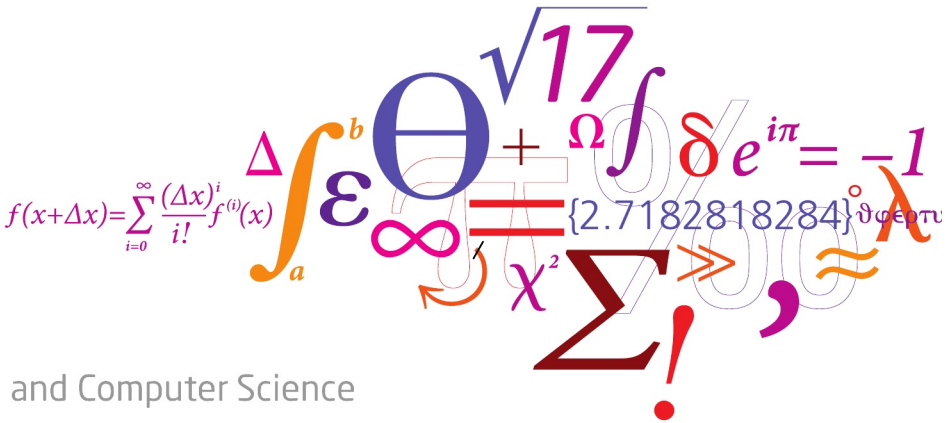
02450: Introduction to Machine Learning and Data Mining

Decision trees and linear regression

Tue Herlau

DTU Compute, Technical University of Denmark (DTU)



DTU Compute
Department of Applied Mathematics and Computer Science

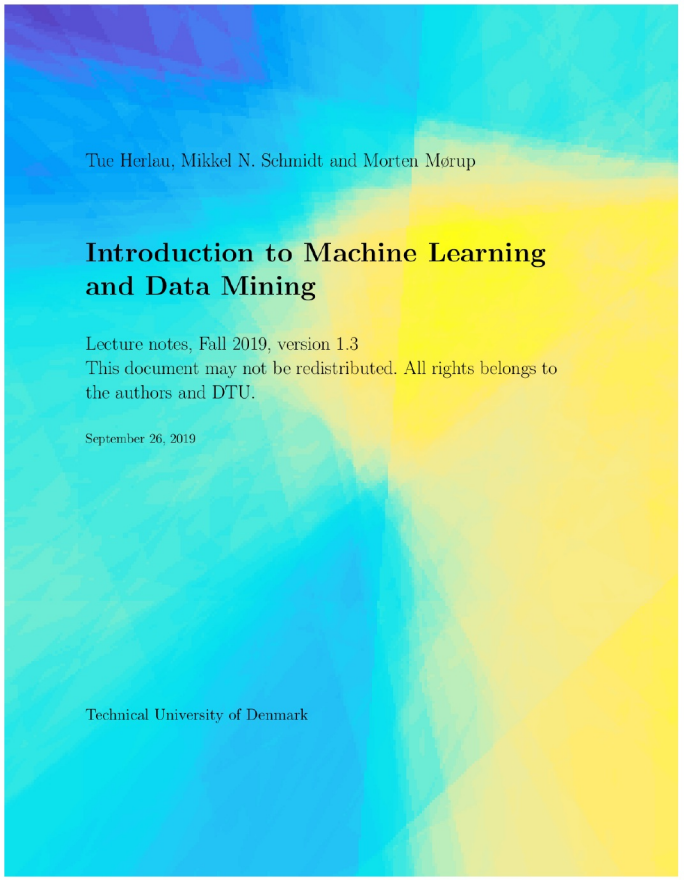
Today

Feedback Groups of the day:

Mathias Fager, Magnus Hansen, Sebastian Bilde, Oliver Storm Køppen, Ahad Imtiaz, Gustav Ohlendorff Brønd, Alexander Mizrahi-Werner, Anders Juhl Jørgensen, Anders Bredgaard Thuesen, Kathrine Schultz-Nielsen, Peter Grønning, David Ib Frederik Sørensen, Andreas Råskov Madsen, Sebastian Roel Hjorth, Clara Deniers, Anton Jakob Sørensen, Patrik Zori, Sophie Gasser, Thomas Johan Sebastiaan Gertsen, Eleonora Girardini, Rikke Toft Grabski, Kasper Lynghøj Grønvang, Benjamin Fogstrup Grundahl, Gísli Tómas Gudjónsson, Katarina Mary Gunter, Tom Haider, Malte Bjørn Hallgren, Christian Kjølhede Hallgren, Alex Hämäläinen, Johan Weiss Hansen, Mads Frederik Hansen, Andy Dünnweber Hansen, Sara Perlt Hansen, Cilie Werner Feldager Hansen, Ilian Oleg Haralampiev, Mikkel Mosbæk Qvist Harteg, Nathaniel Stephen Frost Hartwig, Wanli HE, Camilla Berg Hedberg, Freja Dahl Hede, Troels Kraft Hedelund, Line Heide, Anna Heiselberg, Hanne Maren Helgedagsrud, Niels Svend Helsø, Jonas Christian Henriksen, Anders Henriksen, Tue Herlau, Néstor Hernández Ramos, Sebastian Emil Kokholm Hersbøll, Theis Ferré Hjortkjær, Mohamad Muwfak Hlal, Marin Holi

Reading material:

Chapter 8, Chapter 9



Lecture Schedule

1. Introduction
   3 September: C1

   Data: Feature extraction, and visualization
2. Data, feature extraction and PCA
   10 September: C2, C3
3. Measures of similarity, summary statistics and probabilities
   17 September: C4, C5
4. Probability densities and data Visualization
   24 September: C6, C7

   Supervised learning: Classification and regression
5. **Decision trees and linear regression**
   1 October: C8, C9 **(Project 1 due before 13:00)**
6. Overfitting, cross-validation and Nearest Neighbor
   8 October: C10, C12
7. Performance evaluation, Bayes, and Naive Bayes
   22 October: C11, C13
8. Artificial Neural Networks and Bias/Variance
   29 October: C14, C15
9. AUC and ensemble methods
   5 November: C16, C17

   Unsupervised learning: Clustering and density estimation
10. K-means and hierarchical clustering
    12 November: C18 **(Project 2 due before 13:00)**
11. Mixture models and density estimation
    19 November: C19, C20
12. Association mining
    26 November: C21

    Recap
13. Recap and discussion of the exam
    3 December: C1-C21 **(Project 3 due before 13:00)**

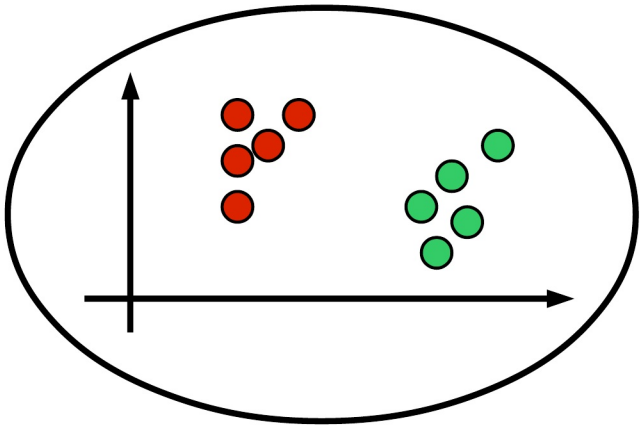
Piazza online help: `https://piazza.com/dtu.dk/fall2019/02450`
Videos/streaming of lectures: `https://video.dtu.dk`

Sorry streaming. The software is broken.

Learning Objectives

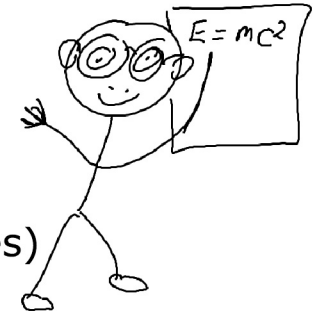
- Explain what supervised learning is
- Explain the difference between classification and regression
- Be able to evaluate classifiers in terms of the confusion matrix, error rate and accuracy
- Understand the principle behind decision trees and Hunt's algorithm
- Apply and interpret decision trees, linear regression and logistic regression

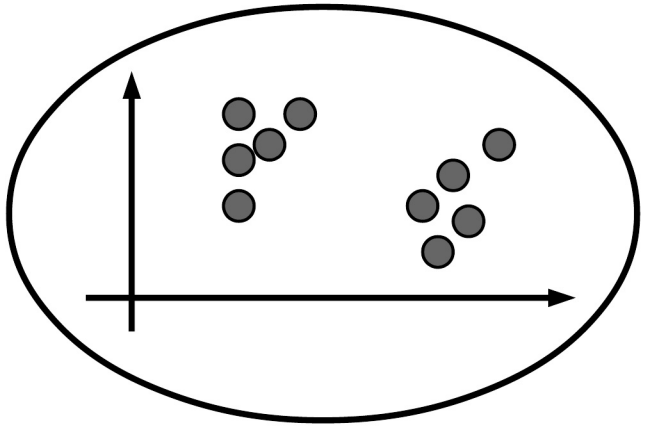
Supervised and Unsupervised learning



Supervised Learning
Input data $\mathbf{x}_n$ and output y_n

(Generalize from known examples)

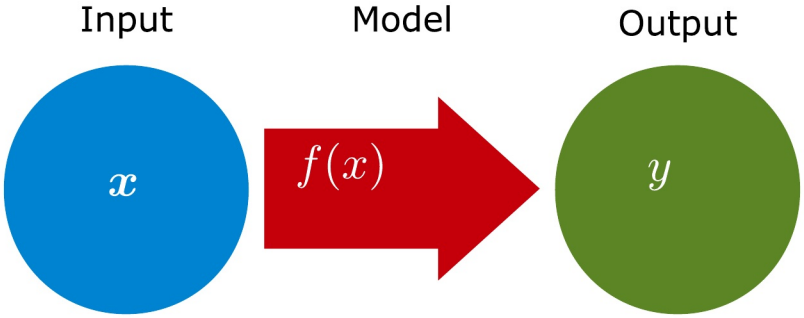




Unsupervised Learning
Input data $\mathbf{x}_n$ alone

(Exploratory analysis)

Supervised learning



- **Data**
  - Inputs and outputs *(this is what we are given)*

$$\{\boldsymbol{x}_n, y_n\}_{n=1}^{N}$$

- **Model**
  - Function that maps inputs to outputs *(what we are trying to determine)*

$$f(\boldsymbol{x})$$

- **Cost function**
  - Dissimilarity measure between observation and prediction *(how we tell if a model is good or bad)*

$$d\big(y, f(\boldsymbol{x})\big)$$

- **Types of supervised learning**
  - Regression: Continuous output **y**
  - Classification: Discrete output **y**

Classification

- **Definition:** Learning a function that maps a data object to a discrete class

- **Why classify?**
  - Descriptive modeling
    - Explain / understand the relation between attributes and class
  - Predictive modeling
    - Predict the class of a new data object

Confusion matrix

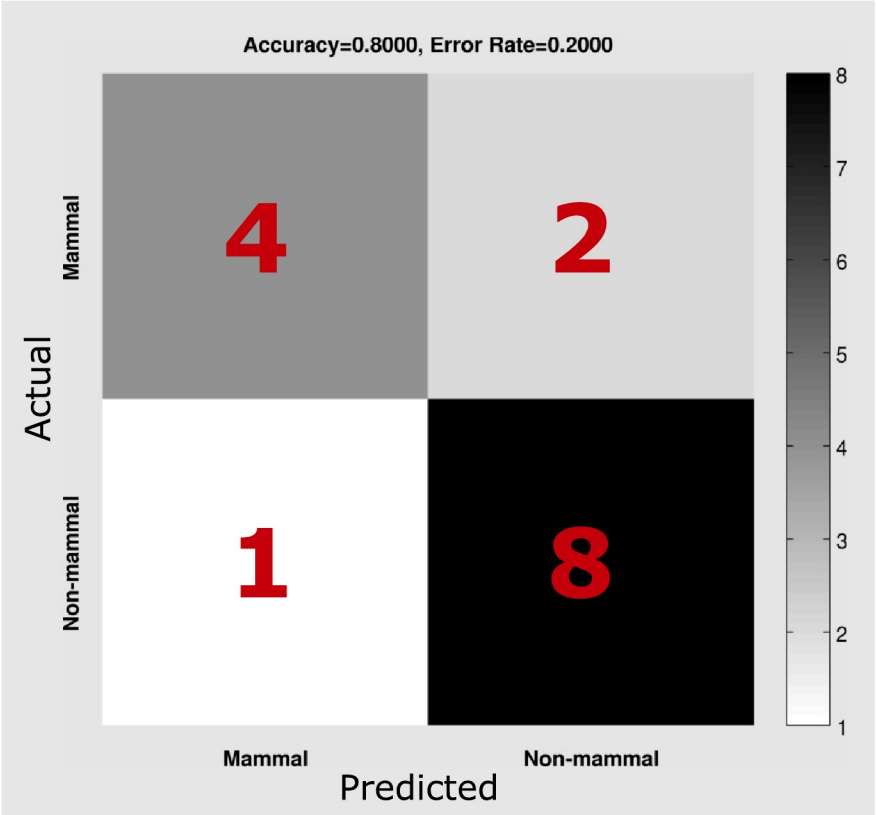
- Visualization of actual versus predicted class labels

- **Accuracy**
  (Number of correctly predicted observations divided by the total number of observations)

$$\frac{4+8}{4+2+1+8} = 80\%$$

- **Error rate**
  (Number of in-correctly predicted observations divided by the total number of observations)

$$\frac{2+1}{4+2+1+8} = 20\%$$



Decision trees

- Remember the game "20 questions to the professor"? (see also www.20q.new)

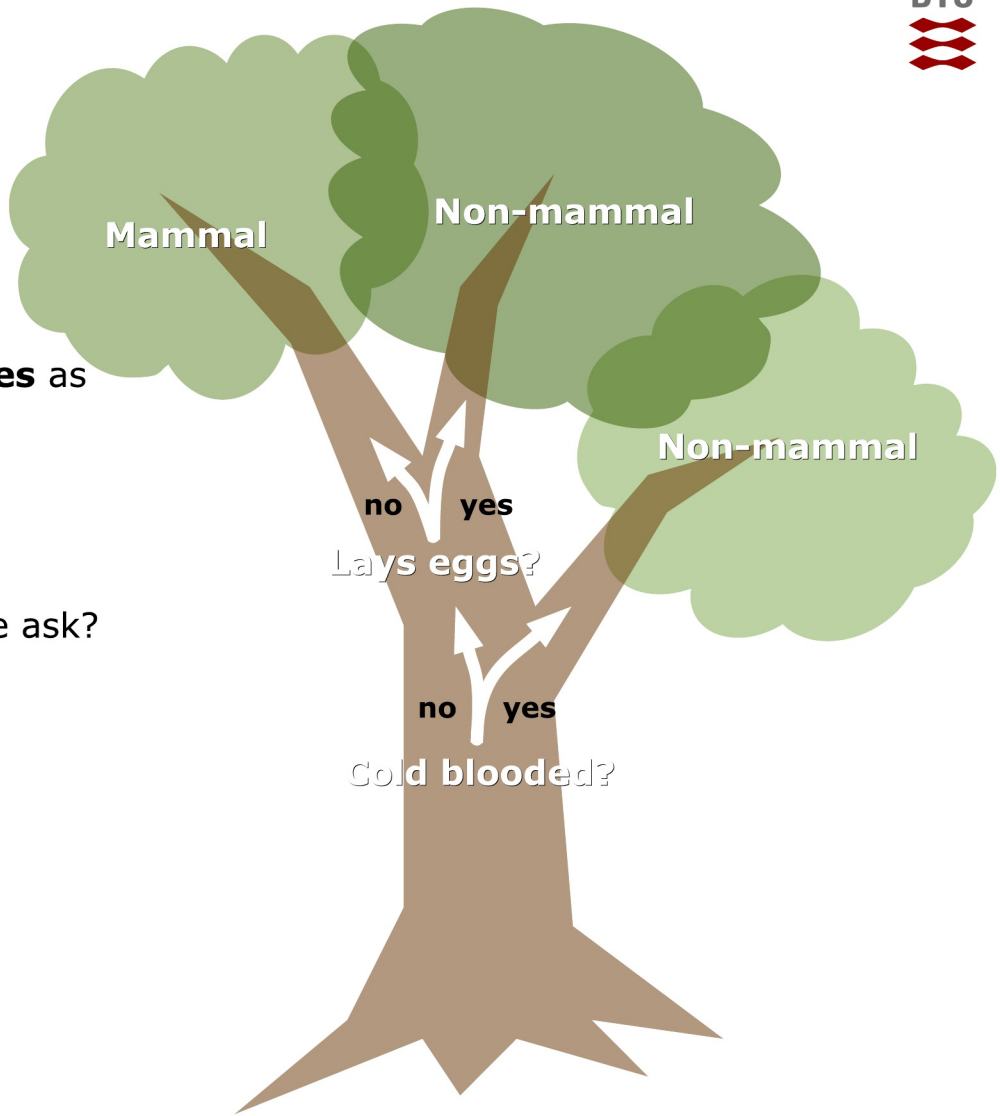
Q1.	Is it an Animal? Yes.
Q2.	Can you hold it? No.
Q3.	Does it live in groups (gregarious)? Yes.
Q4.	Are there many different sorts of it? No.
Q5.	Can it jump? Yes.
Q6.	Does it eat seeds? No.
Q7.	Is it white? Sometimes.
Q8.	Is it black and white? No.
Q9.	Does it have paws? Yes.
Q10.	Can you see it in a zoo? Yes.
Q11.	Does it roar? Yes.
Q12.	Is it worth a lot of money? Yes.
Q13.	Does it have spots? Yes.
Q14.	Is it multicoloured? Yes.
Q15.	Can you make money by selling it? Yes.
Q16.	Does it live in the jungle? Yes.
Q17.	I guessed that it was a leopard? Wrong.
Q18.	Does it like to play? Yes.
Q19.	I guessed that it was a cheetah? Wrong.
Q20.	I am guessing that it is a siberian tiger? Correct.

Decision trees

- Ask a series of questions until a conclusion is reached

- **Example**: Classify **vertebrates** as
  - **Mammal** or
  - **Non-mammal**

- **Learning task**
  - Which questions should we ask?



Hunts algorithm

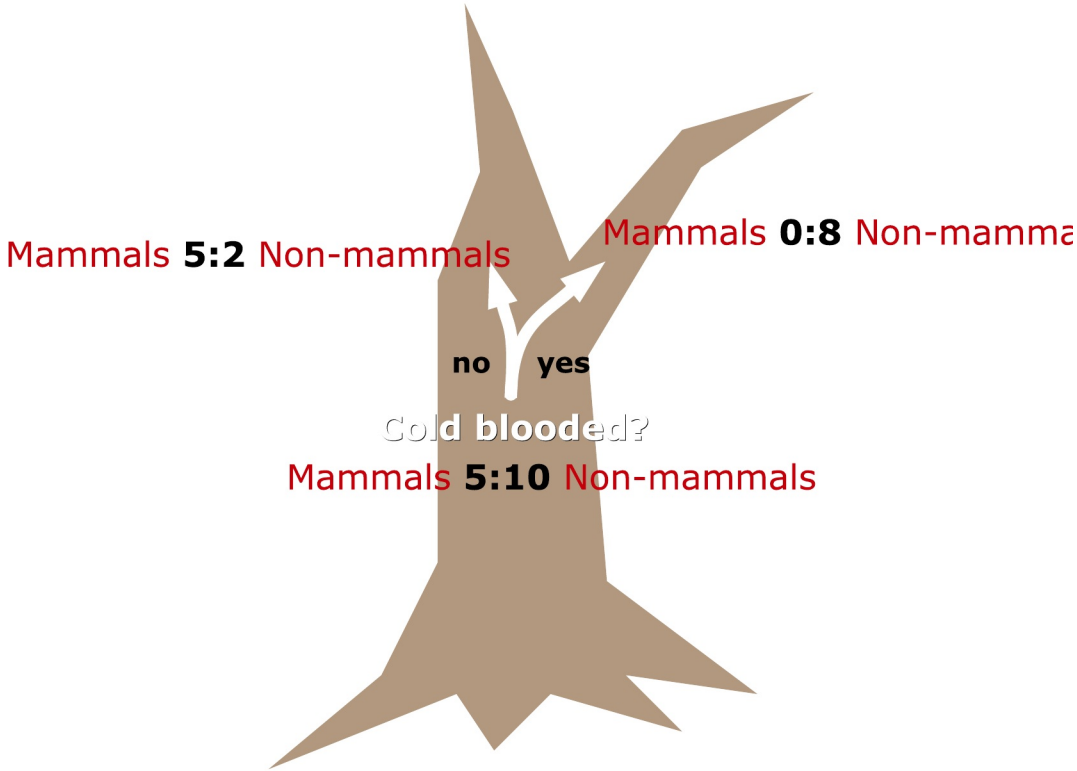
- Assign all data objects to the root

Hunts algorithm

- Select an attribute test condition
  - Find a good question to ask

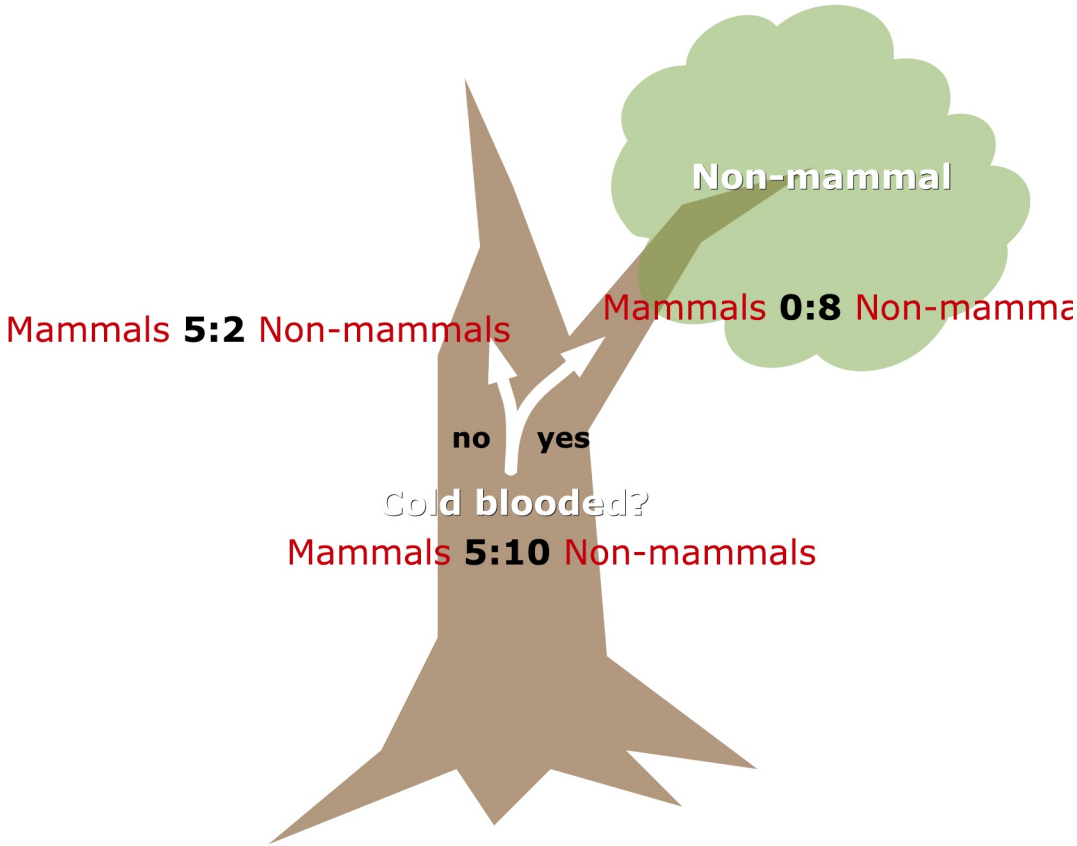
Hunt's Algorithm

- Partition the data objects into subsets according to the test condition



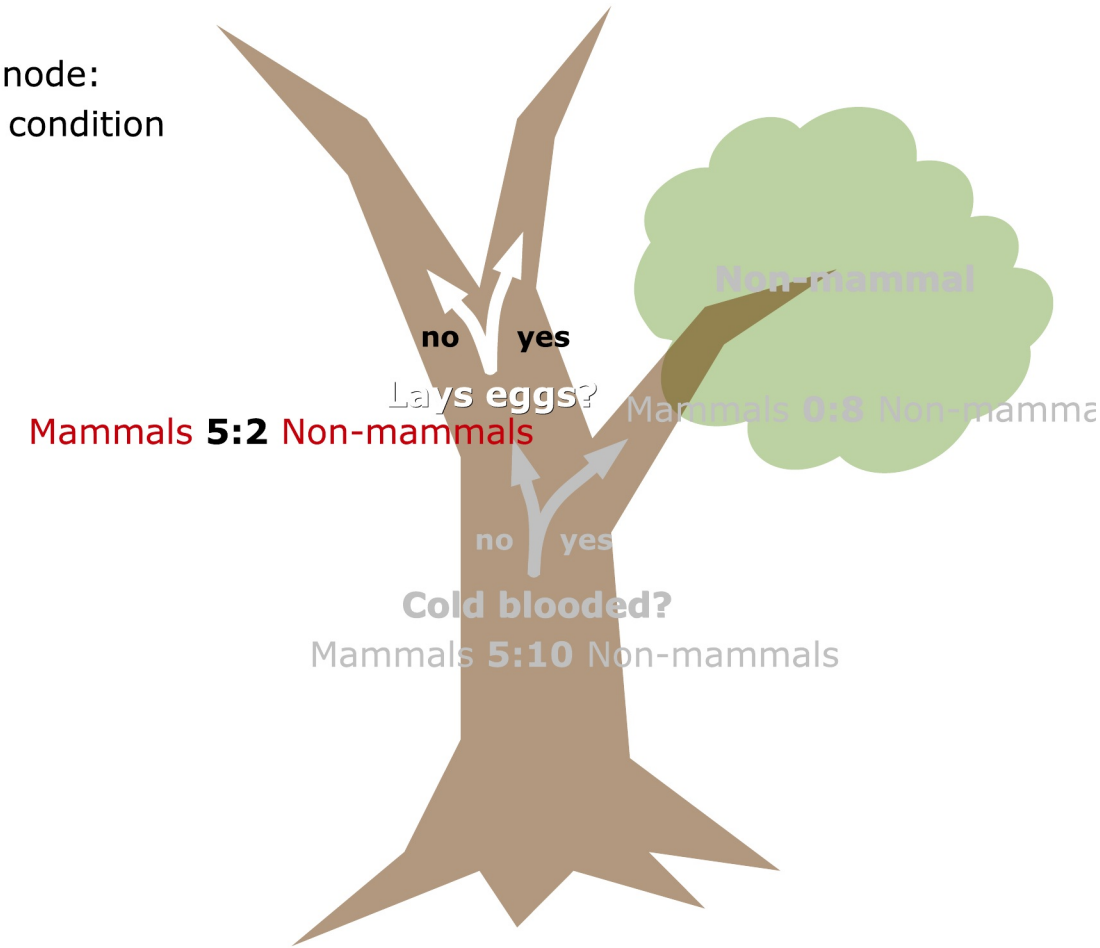
Hunts algorithm

- If all data objects belong to the same class
  - Create a leaf node



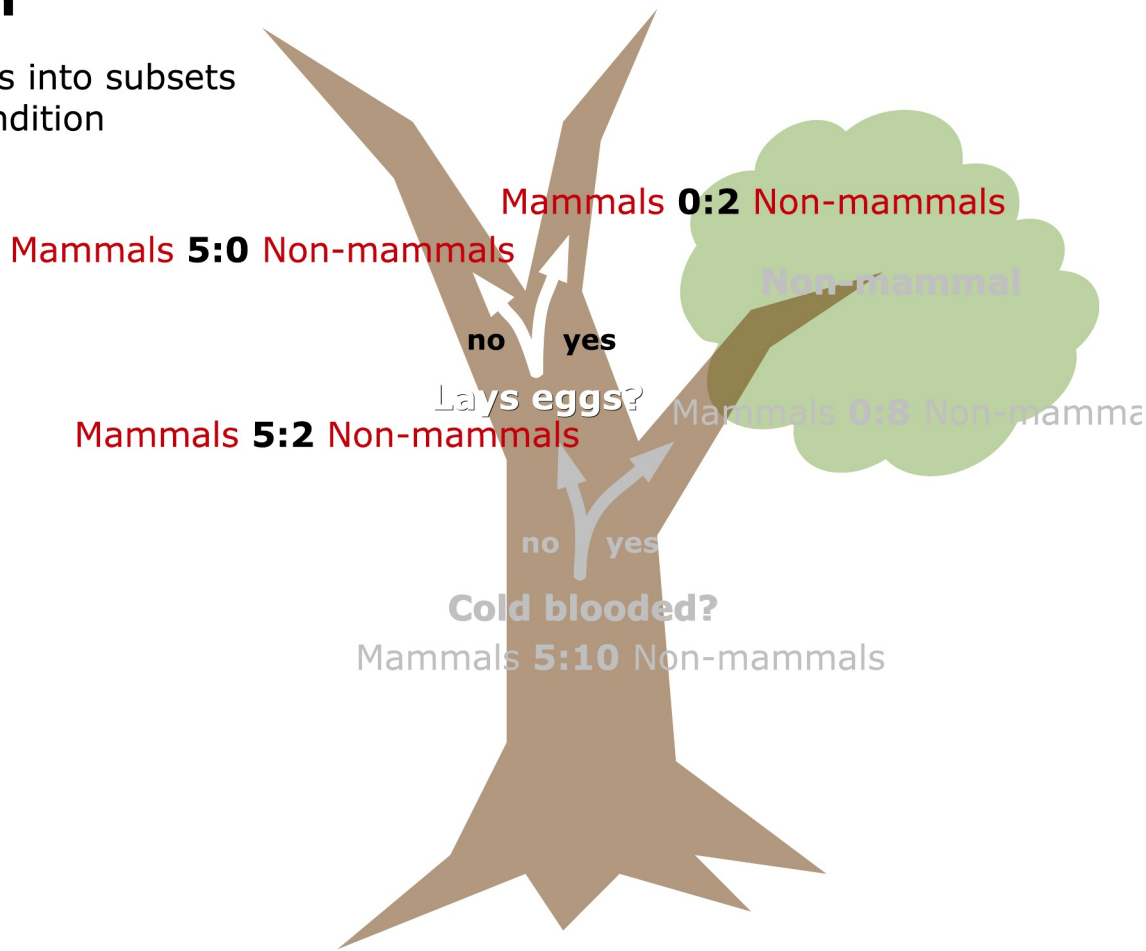
Hunts algorithm

- Repeat for each non-leave node:
  - Select an attribute test condition



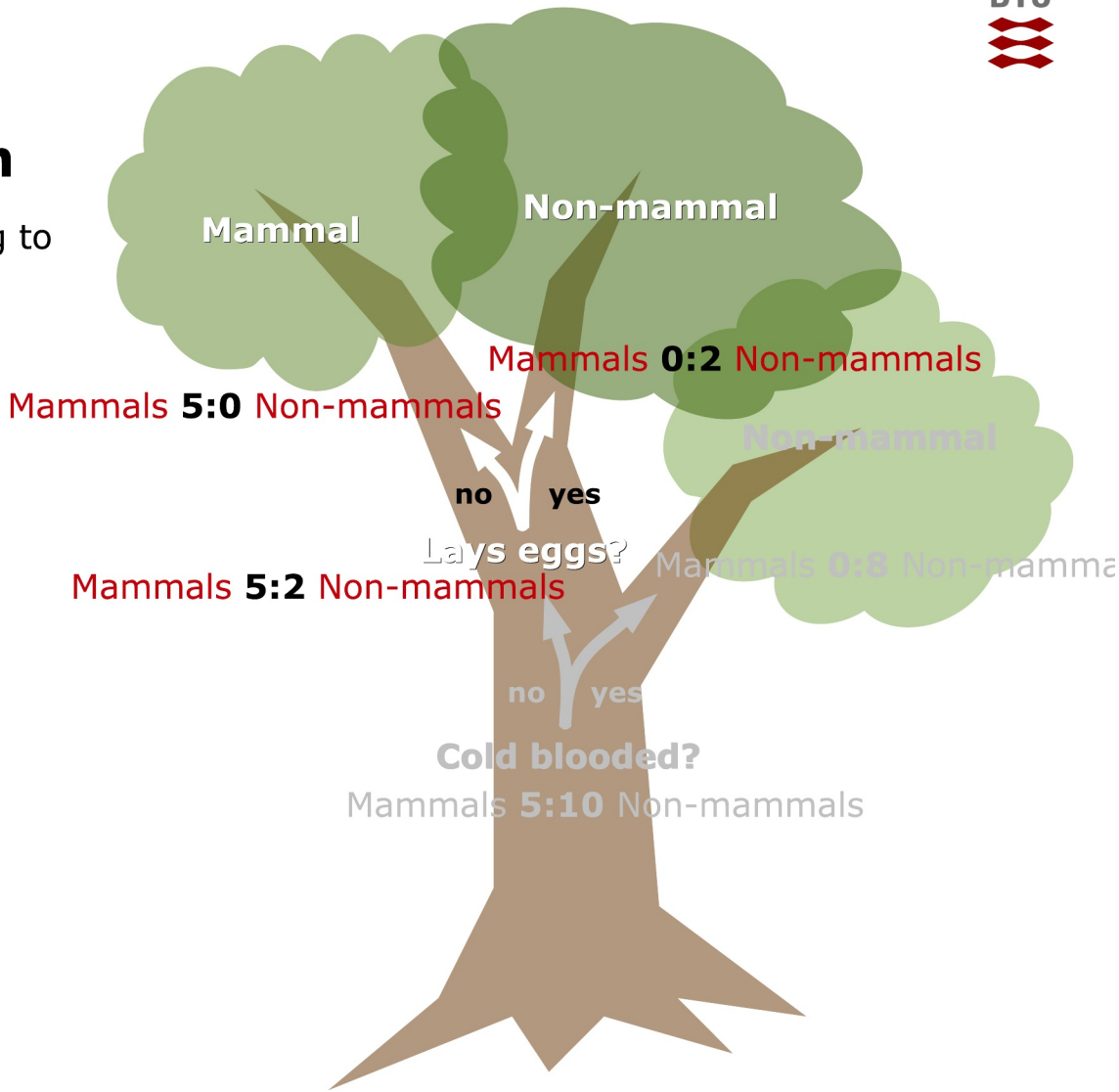
Hunts algorithm

- Partition the data objects into subsets according to the test condition



Hunts algorithm

- If all data objects belong to the same class
  - Create a leaf node



Hunts algorithm

- But how do we find the **best question** at each step?

Algorithm 2: Hunt's algorithm for decision trees

Require: Initial tree T only containing the root node

Require: D_r : Dataset associated with the current branch. Initially just the full dataset

if The **stop criterion** is met **then**

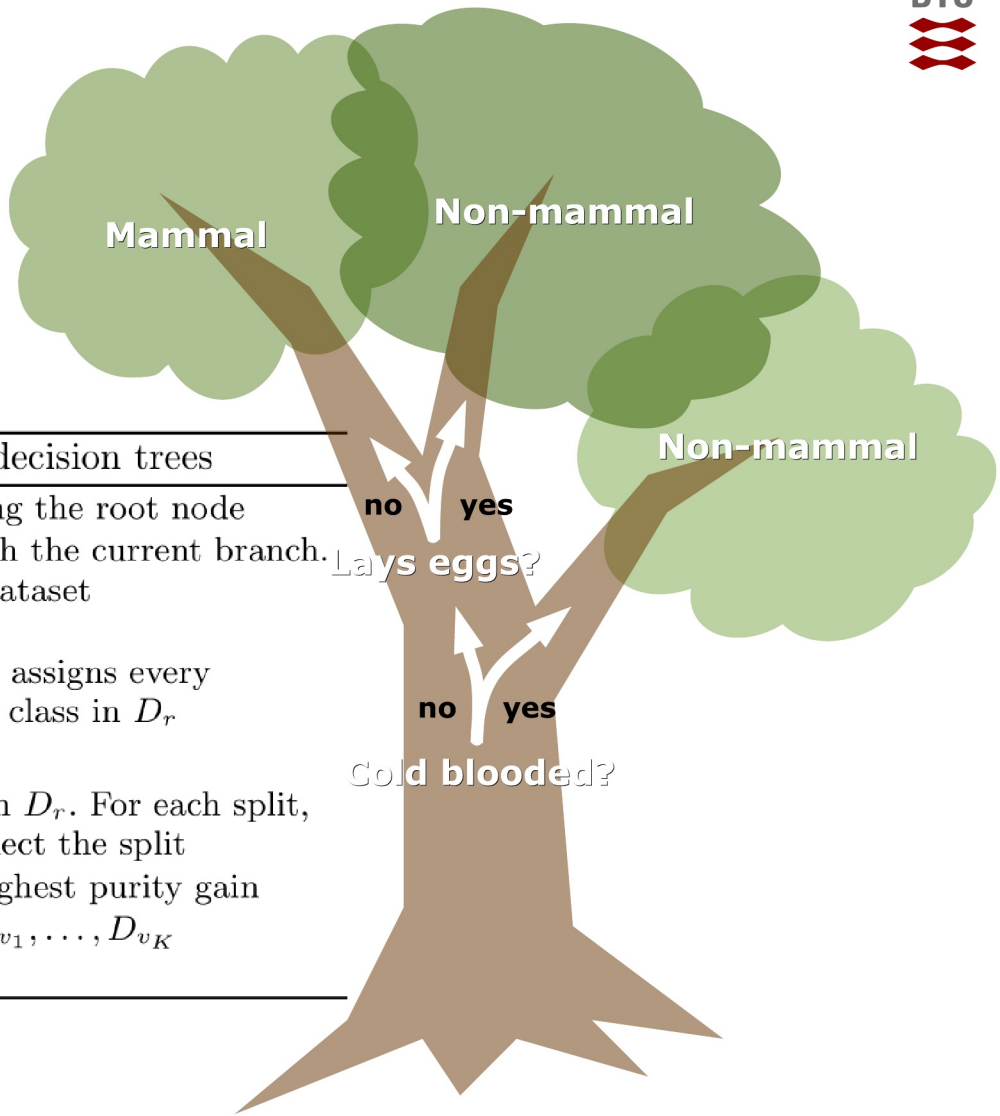
Add a leaf node to the tree which assigns every observation to the most prevalent class in D_r

else

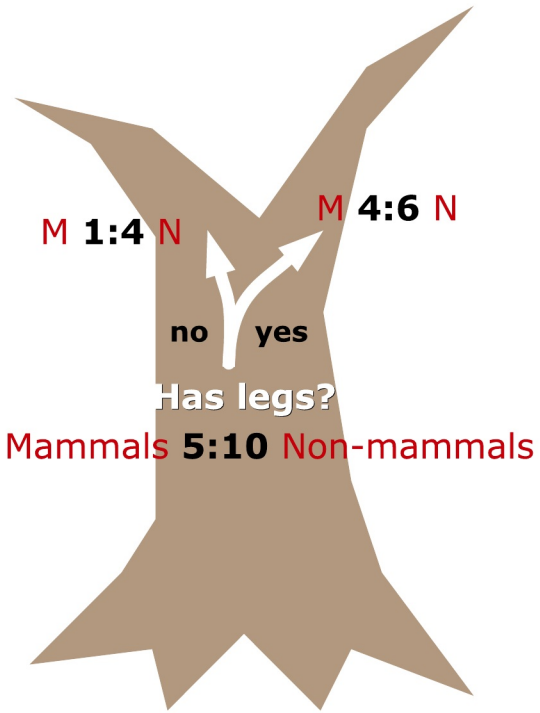
Try a number of different splits on D_r. For each split, compute the **purity gain** and select the split $D_r = \{D_{v_1}, \ldots, D_{v_K}\}$ with the highest purity gain

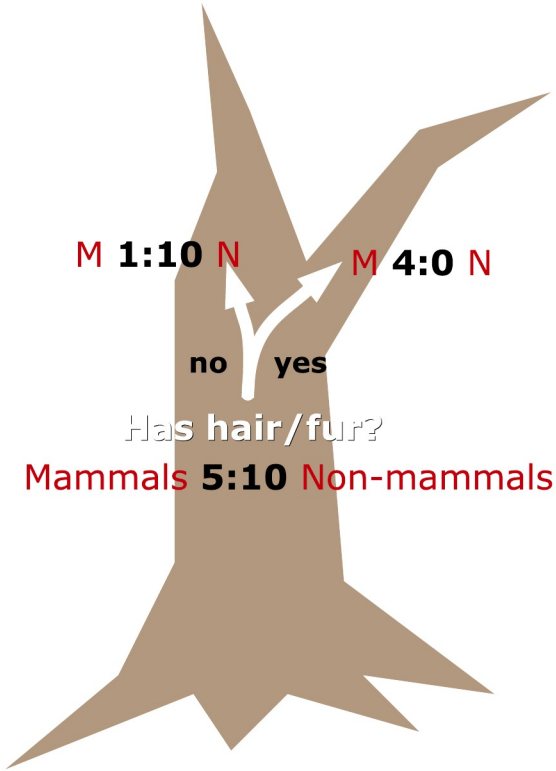
Recursively call the method on $D_{v_1}, \ldots, D_{v_K}$

end if



Which split is best?

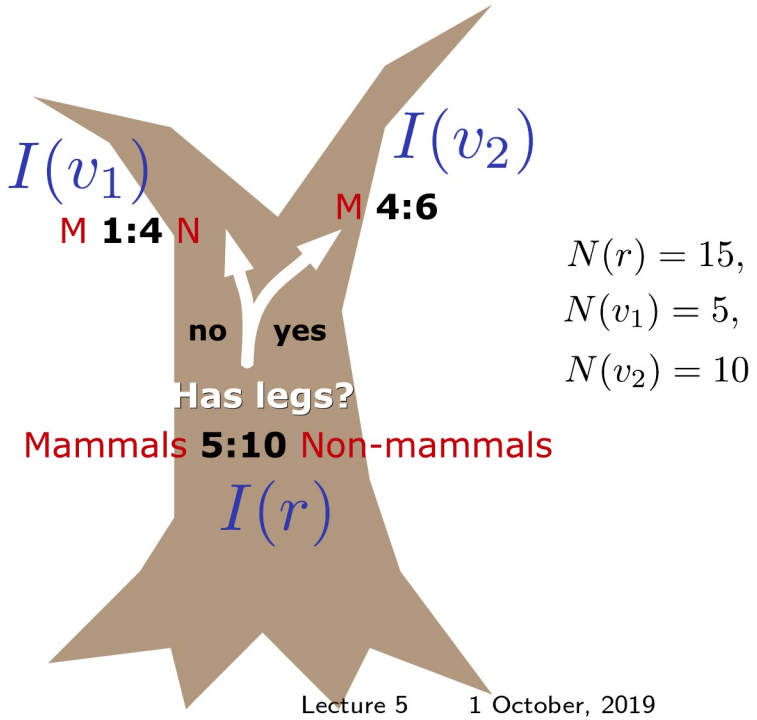




Which split is best?

- Create a measure Δ (**the purity gain**) of how good a split is
- A binary split creates 3 partitions: the root r and the right/left branches v_1, v_2.
- For each partition, we compute $I(r), I(v_1), I(v_2)$ (the **impurity**)
- Purity gain is the **weighted reduction in impurity**:

$$\Delta = I(r) - \sum_{k=1}^{K=2} \frac{N(v_k)}{N(r)} I(v_k)$$



Which split is best?

- Create a measure Δ (**the purity gain**) of how good a split is
- A binary split creates 3 partitions: the root r and the right/left branches v_1, v_2.
- For each partition, we compute $I(r), I(v_1), I(v_2)$ (the **impurity**) of each partition
- Purity gain is the **weighted reduction in impurity**:

$$\Delta = I(r) - \sum_{k=1}^{K=2} \frac{N(v_k)}{N(r)} I(v_k)$$

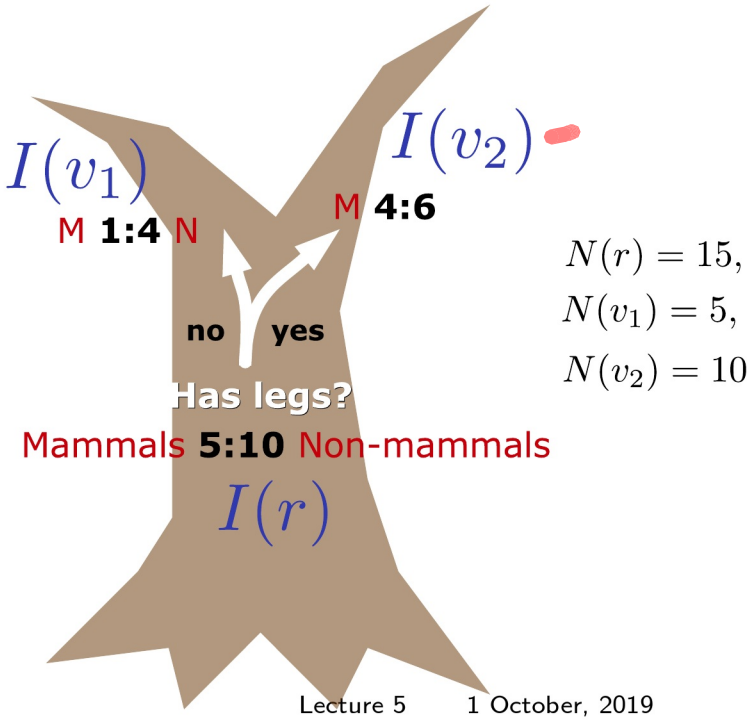
The impurity measure $I()$ can be one of the following

$$\text{Entropy}(v) = -\sum_{c=1}^{C} p(c|v) \log_2 p(c|v),$$

$$\text{Gini}(v) = 1 - \sum_{c=1}^{C} p(c|v)^2,$$

$$\text{ClassError}(v) = 1 - \max_c p(c|v).$$

$$p(c|v) = \frac{\{\text{Nr. in class } c \text{ in branch } v\}}{N(v)}$$



Quiz 1, Impurity gain

If we use the Gini index as impurity measure I, what is the purity gain Δ for the split indicated by the tree?

$$\Delta = I(r) - \sum_{k=1}^{K=2} \frac{N(v_k)}{N(r)} I(v_k)$$

I(root) = 1-(1/3)^2 - (2/3)^2 = 4/9

A. ≈ 0.0177

B. ≈ 0.104

C. ≈ 0.129

D. ≈ 0.2

E. Don't know.

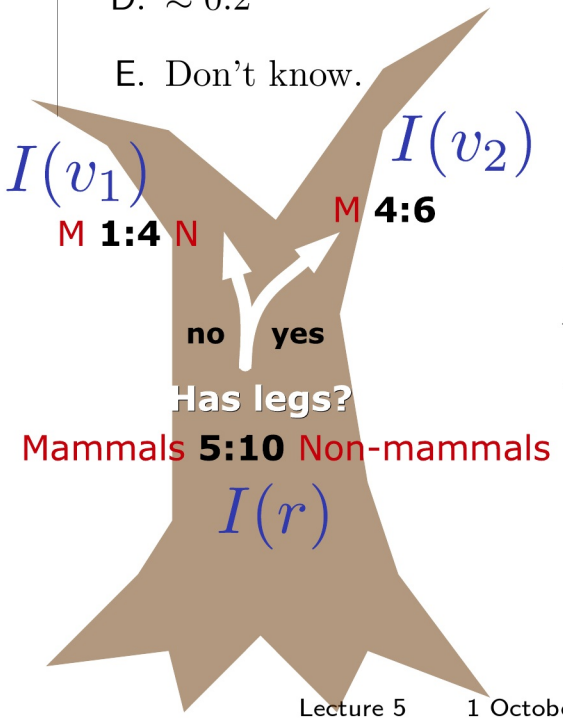
The impurity measure $I()$ can be one of the following

$$\text{Entropy}(v) = -\sum_{c=1}^{C} p(c|v) \log_2 p(c|v),$$

$$\text{Gini}(v) = 1 - \sum_{c=1}^{C} p(c|v)^2,$$

$$\text{ClassError}(v) = 1 - \max_c p(c|v).$$

$$p(c|v) = \frac{\{\text{Nr. in class } c \text{ in branch } v\}}{N(v)}$$



$N(r) = 15,$
$N(v_1) = 5,$
$N(v_2) = 10$

Using Gini impurity we get:

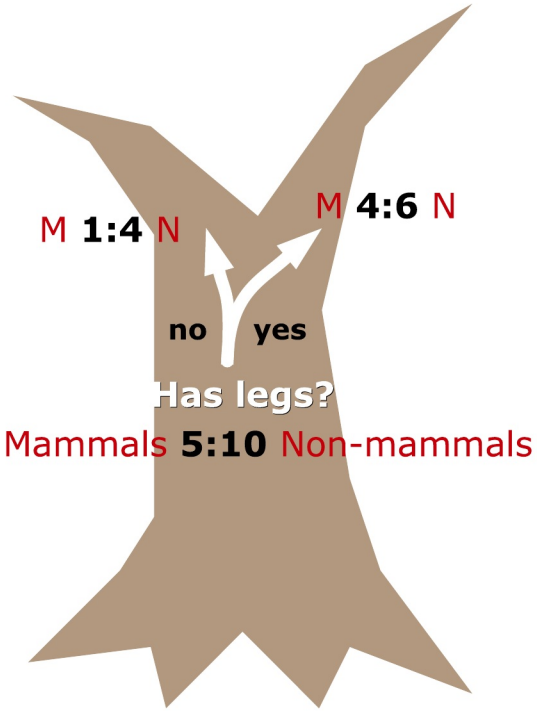
$$I(r) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2, I(v_1) = 1 - \left(\frac{1}{5}\right)^2 - \left(\frac{4}{5}\right)^2, I(v_2) = 1 - \left(\frac{4}{10}\right)^2 - \left(\frac{6}{10}\right)^2$$

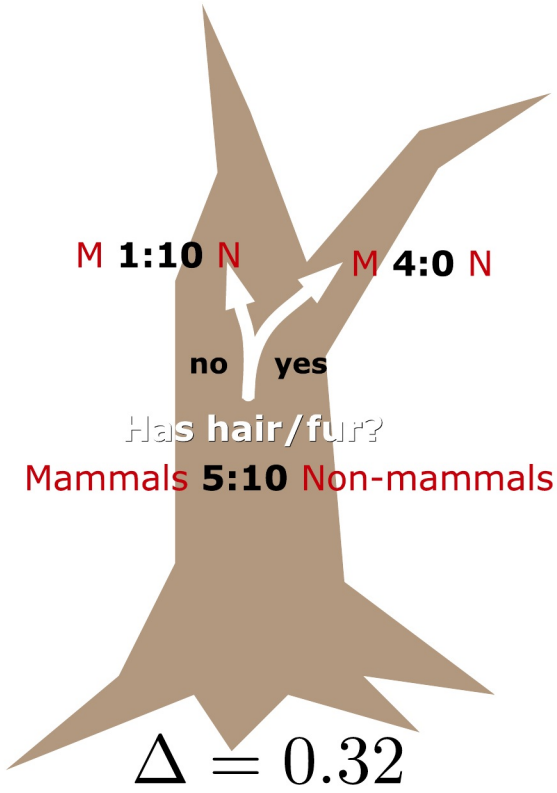
and finally

$$\Delta = I(r) - \frac{5}{15} I(v_1) - \frac{10}{15} I(v_2) \approx 0.0177$$

Selecting the best split

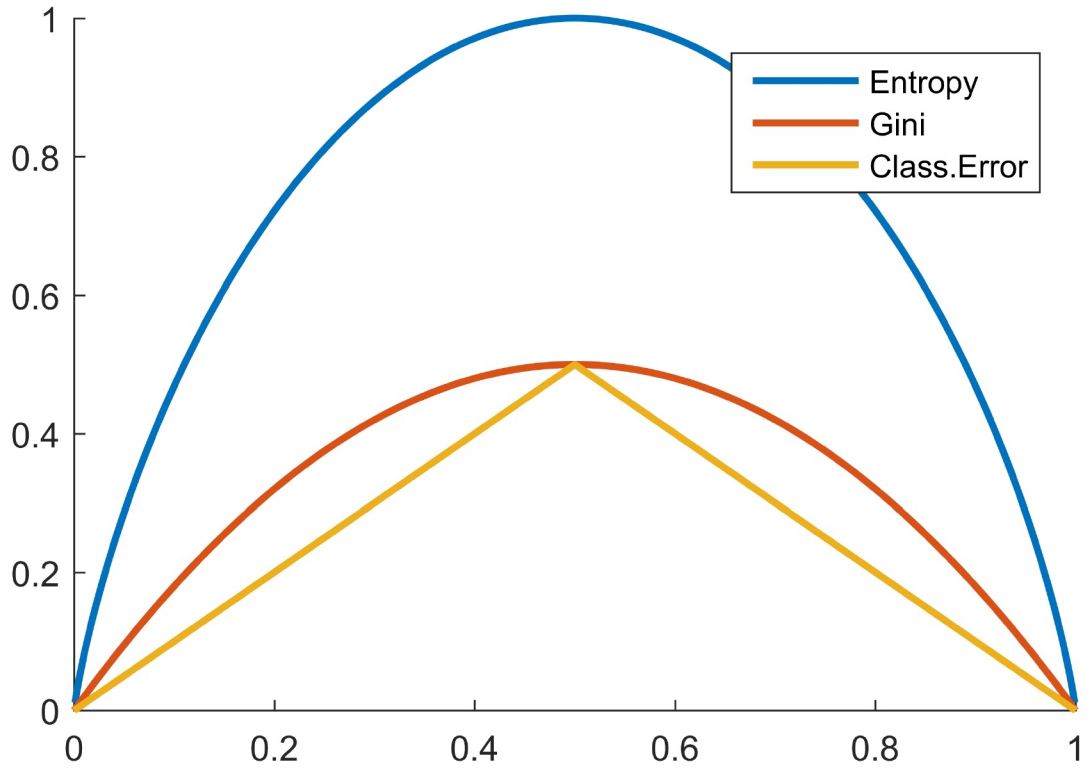
- Consider a large number of possible splits
- Compute a measure of impurity after the proposed split
  - For each new branch of the tree
  - Compute weighted average impurity
- Choose split that reduces impurity most





$$\Delta = 0.32$$

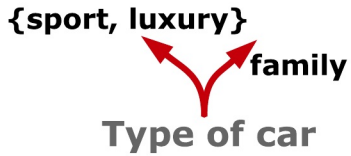
For a two class problem



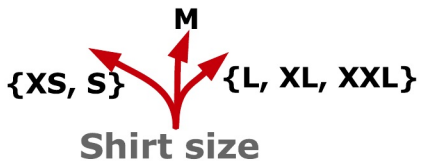
Which splits to consider

- **Binary**

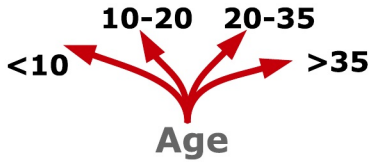
- **Nominal**

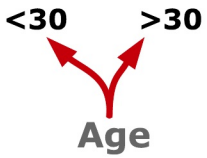


- **Ordinal**

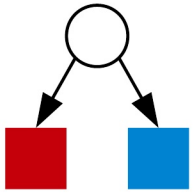


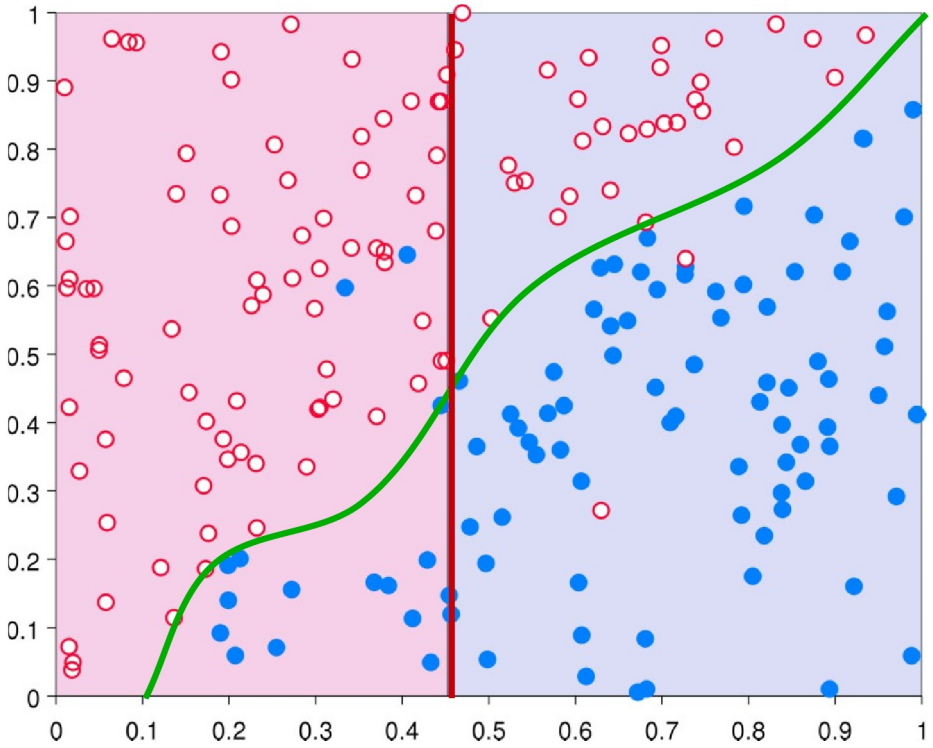
- **Continuous**



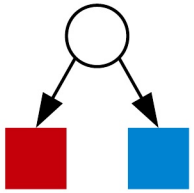


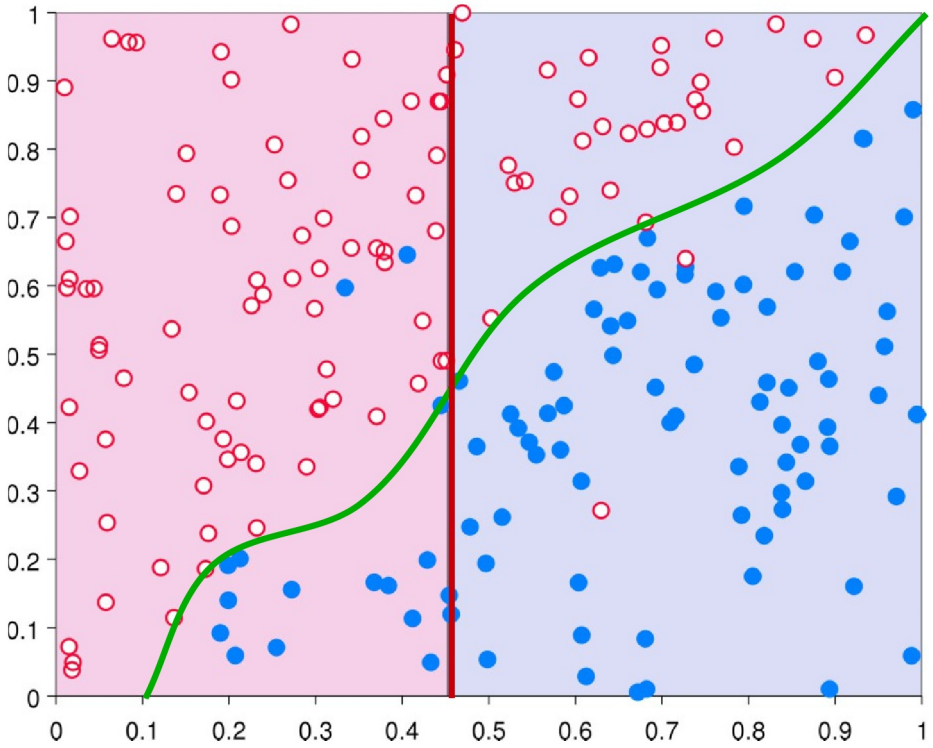
Classification Trees



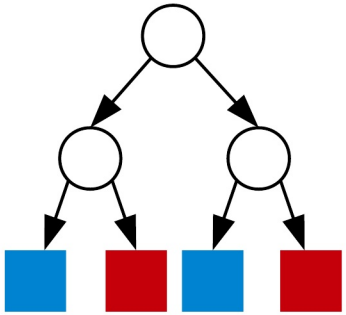


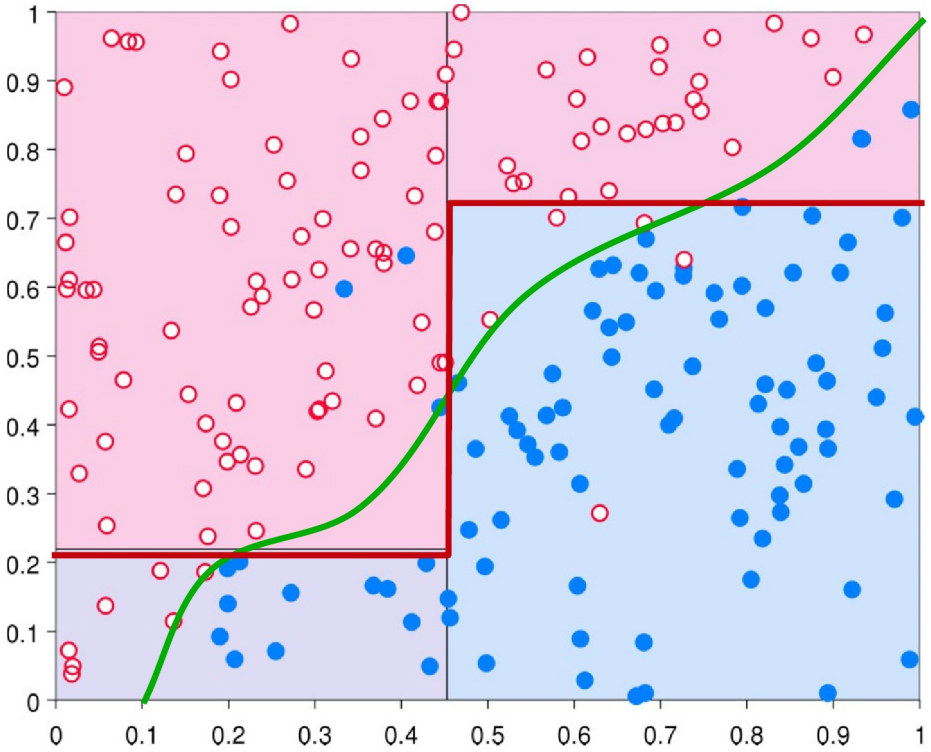
Classification Trees



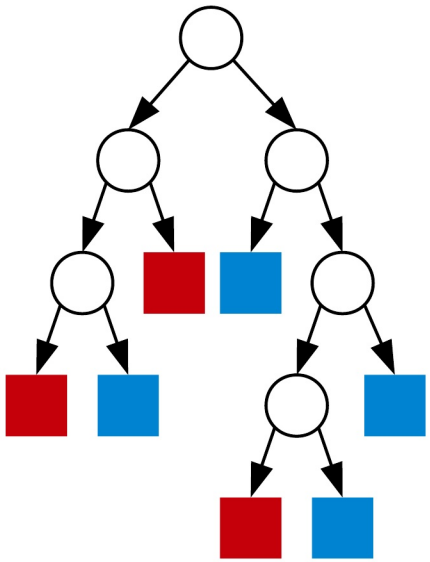


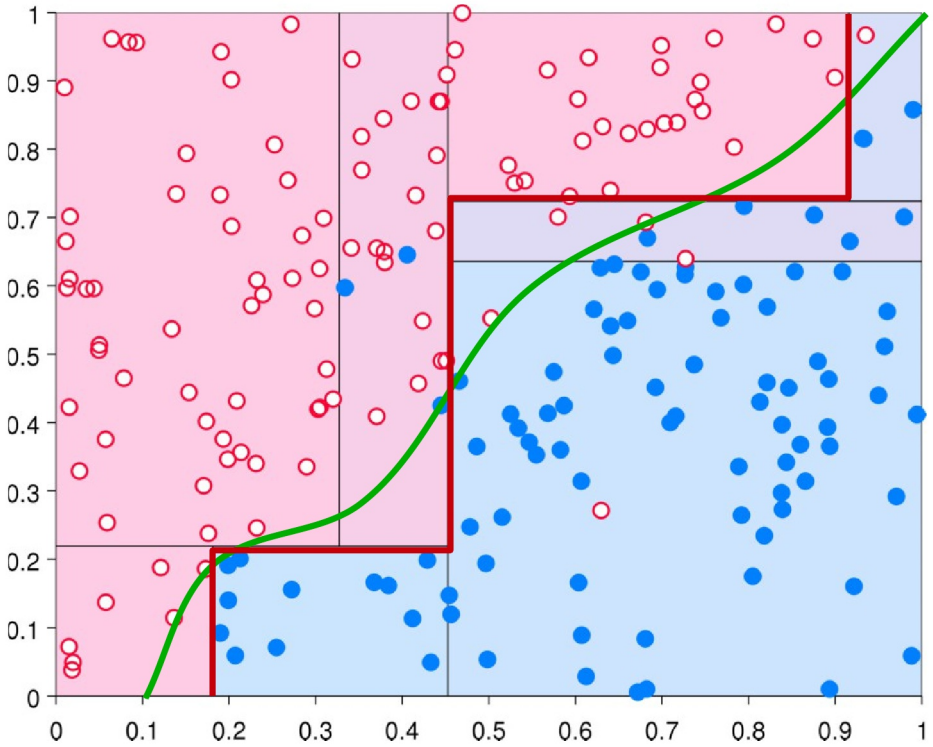
Classification trees





Classification trees



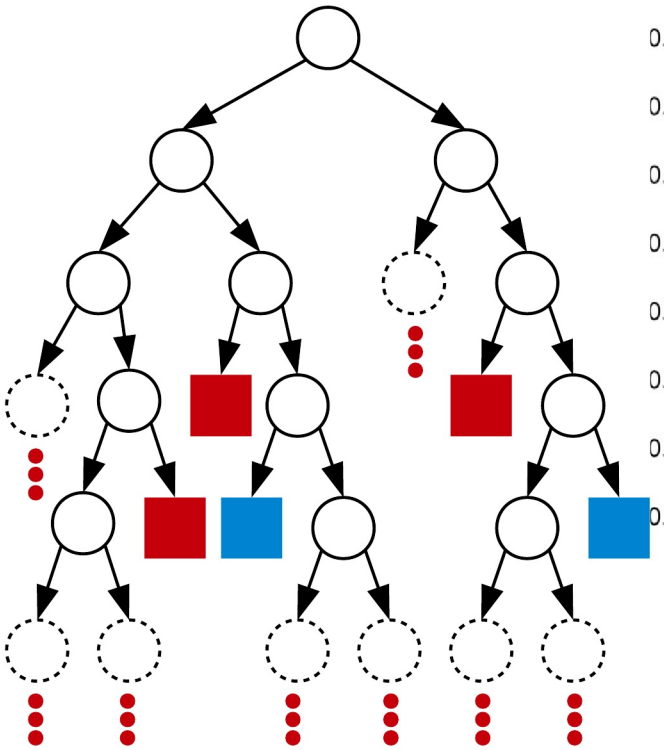


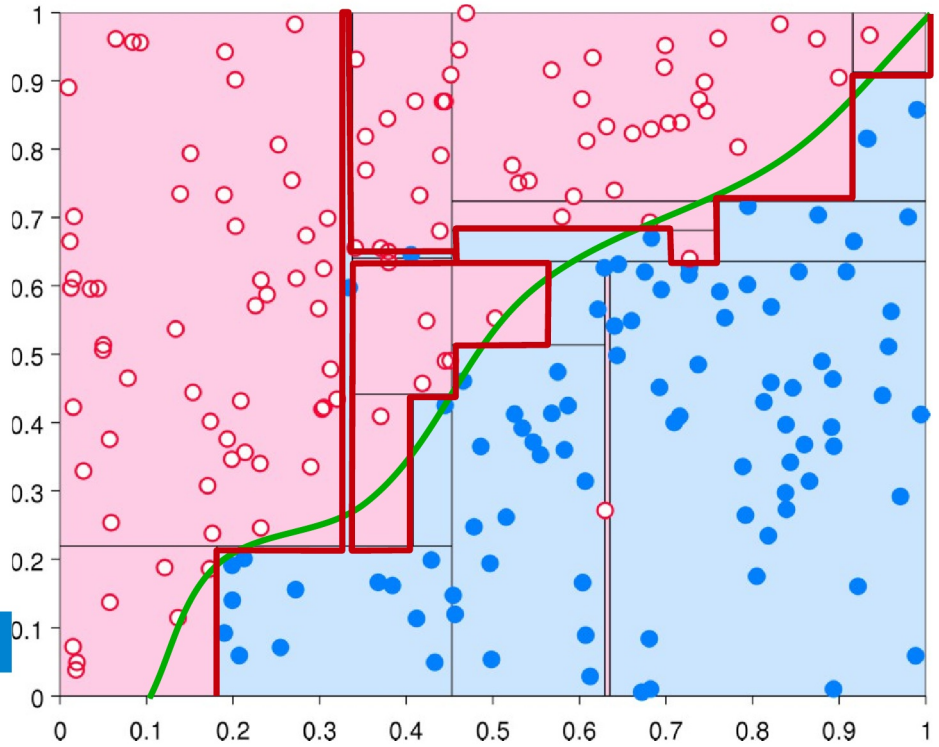
Classification trees

Common stopping criteria:

All records have the same class label

The number of observations have fallen below some minimum treshold



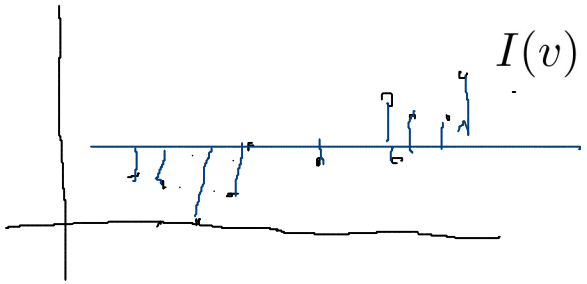


Regression trees

Algorithm 4: Hunt's algorithm for regression trees

Require: Initial tree T only containing the root node
Require: D_r : Dataset associated with the current branch. Initially just the full dataset
if The **stop criterion** is met **then**
Add a leaf node to the tree which assigns every observation the mean value of the nodes in D_r: $y(r) = \frac{1}{N(r)} \sum_{i \in r} y_i$
else
Try a number of different splits on D_r. For each split, compute the **purity gain** using the sum-of-squares impurity measure and select the split $D_r = \{D_{v_1}, \ldots, D_{v_K}\}$ with the highest purity gain
Recursively call the method on $D_{v_1}, \ldots, D_{v_K}$
end if

Use mean square error as purity gain



$$I(v) = \frac{1}{N(v)} \sum_{i \in v} (y_i - \hat{y}_v)^2, \quad \hat{y}_v = \frac{1}{N(v)} \sum_{i \in v} y_i$$

mean

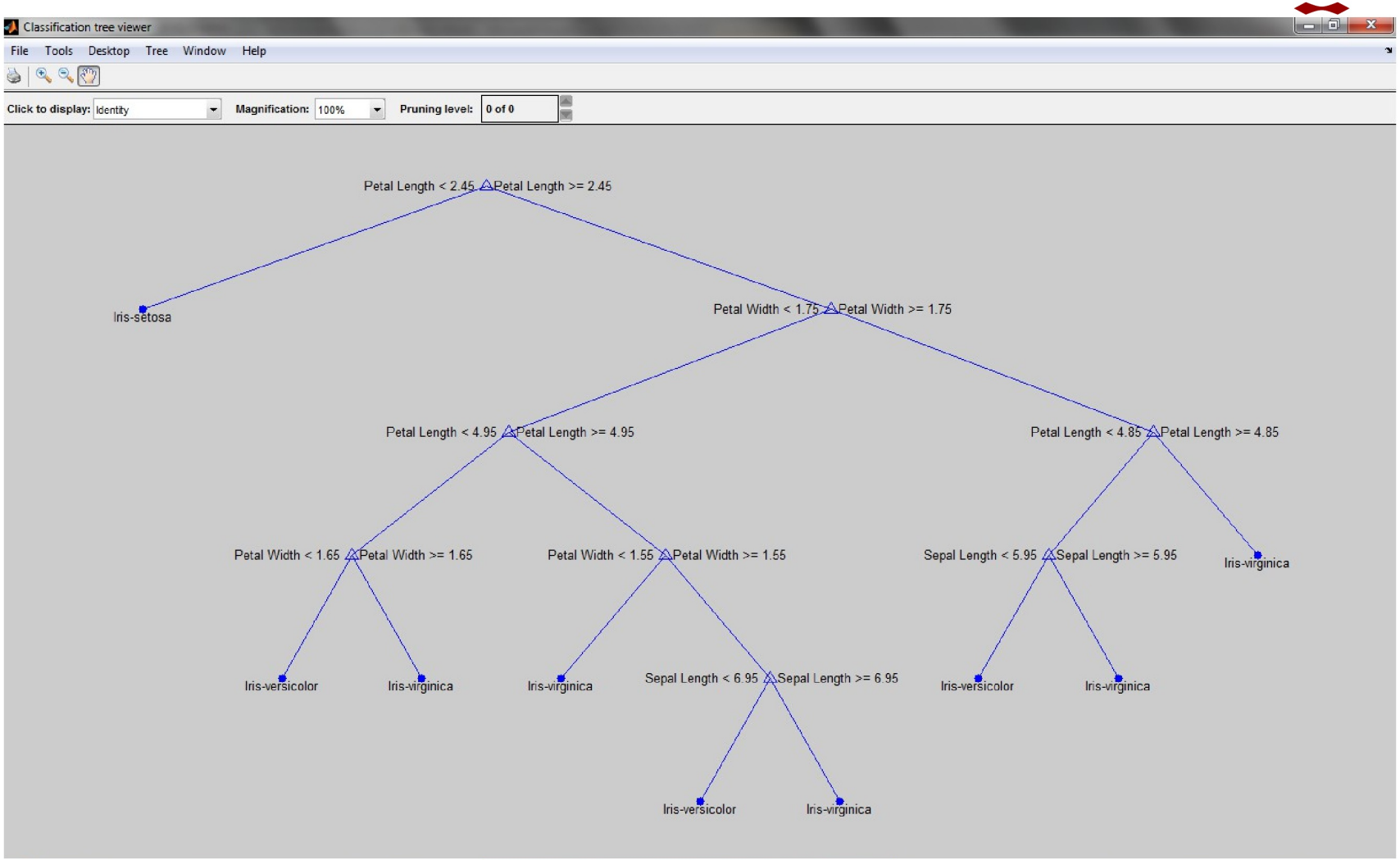
The iris data set

- **Three flowers**
  - 50 instances of each class, 150 in total

- **Attributes**
  - Sepal (outermost leaves)
    - length in cm
    - width in cm
  - Petal (innermost leaves)
    - length in cm
    - width in cm
  - Class of flower
    - Iris Setosa
    - Iris Versicolour
    - Iris Virginica

Fower ID	Attribute			
	Sepal Length	Sepal Width	Petal Length	Petal Width
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
.	.	.	.	.
.	.	.	.	.
150	5.9	3.0	5.1	1.8

X Observation x Attribute



What would the following iris flower be classsified as?

Sepal Length	Sepal Width	Petal Length	Petal Width
4.0	3.5	3.0	2.0

Example: Real data

Classification data: Presence of breast cancer and Forrest cover type

	N	M	Classes
Breast Cancer	569.0	30.0	2.0
Covertype	581012.0	54.0	7.0

Regression data: Price of houses

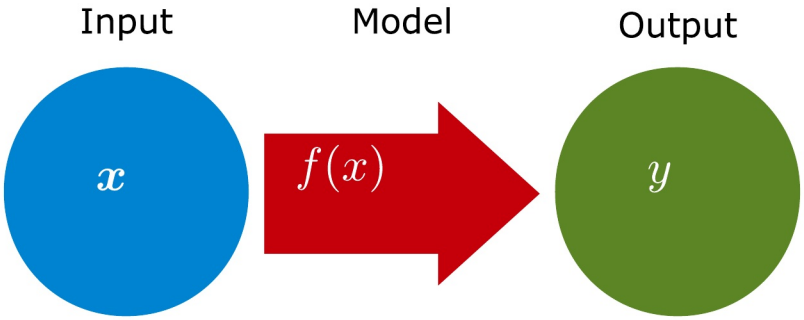
	N	M
Boston Cancer	569.0	30.0
Covertype	581012.0	54.0

Example: Real data

	Breast Cancer		Covertype	
	Acc (train)	Acc (test)	Acc (train)	Acc (test)
Tree Classification	1.0	0.937	1.0	0.938
Tree Classification (minsplit=15)	0.986	0.951	0.974	0.93
Tree Classification (minsplit=30)	0.965	0.888	0.956	0.919

	Boston				California Housing			
	L1 (train)	L1 (test)	L2 (train)	L2 (test)	L1 (train)	L1 (test)	L2 (train)	L2 (test)
Tree Regression	0.0	2.847	0.0	14.299	0.0	0.46	0.0	0.522
Tree Regression (Minsplit 15)	1.18	3.131	2.641	26.49	0.194	0.423	0.097	0.44
Tree Regression (Minsplit 30)	1.851	2.592	7.359	11.068	0.262	0.409	0.16	0.406

Supervised learning



- **Mapping between domains**
  - Classification: Discrete (nominal) output
  - Regression: Continuous output

Supervised learning

- **Data**
  - Inputs and outputs $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$

- **Model**
  - Function that maps inputs to outputs

$$f(\boldsymbol{x})$$

- **Cost function**
  - Dissimilarity measure between data and model

$$d\big(y, f(\boldsymbol{x})\big)$$

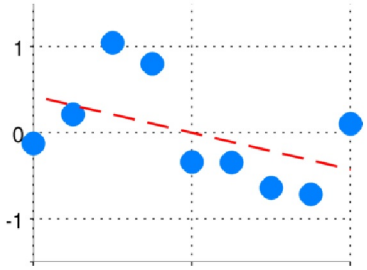
Regression

- **Definition:** Learning a function that maps a data object to a continuous-valued output

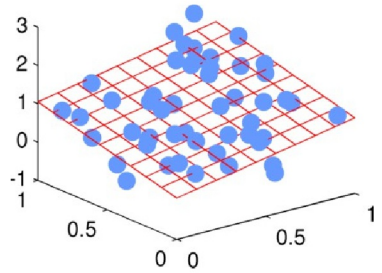
- **Why Regression?**
  - Descriptive modeling
    - Explain / understand the relation between attributes and continous-valued output
  - Predictive modeling
    - Predict the output value of a new data object

Linear regression

- 1-dimensional inputs
  $f(x) = w_0 + w_1 x$



- 2-dimensional inputs
  $f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2$



- K-dimensional inputs
  $f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_K x_K$

Linear regression

- K-dimensional inputs

$$f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_K x_K$$

- Non-linearly transformed inputs

$$f(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_K x^K$$

$$f(x) = w_0 + w_1 sin(x) + w_2 cos(x)$$

Linear regression

- K-dimensional inputs

$$f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_K x_K$$

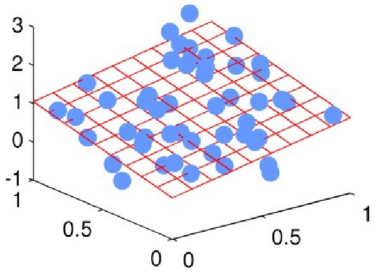
- Non-linearly transformed inputs

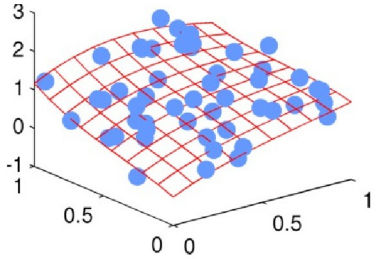
$$f(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_K x^K$$

$$f(x) = w_0 + w_1 sin(x) + w_2 cos(x)$$

- **Example**

$$f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2$$





$$f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_6 x_1^3 + w_7 x_1^2 x_2 + w_8 x_1 x_2^2 + w_9 x_2^3$$

Vector notation

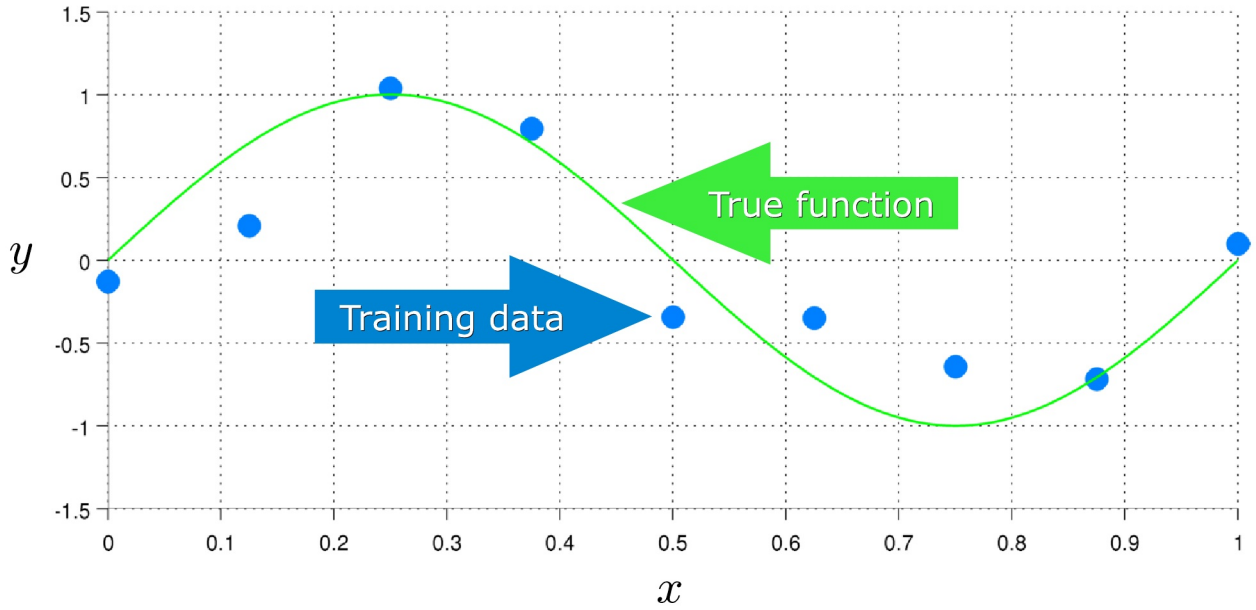
- The linear model can be written compactly using vector notation

$$f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_K x_K$$

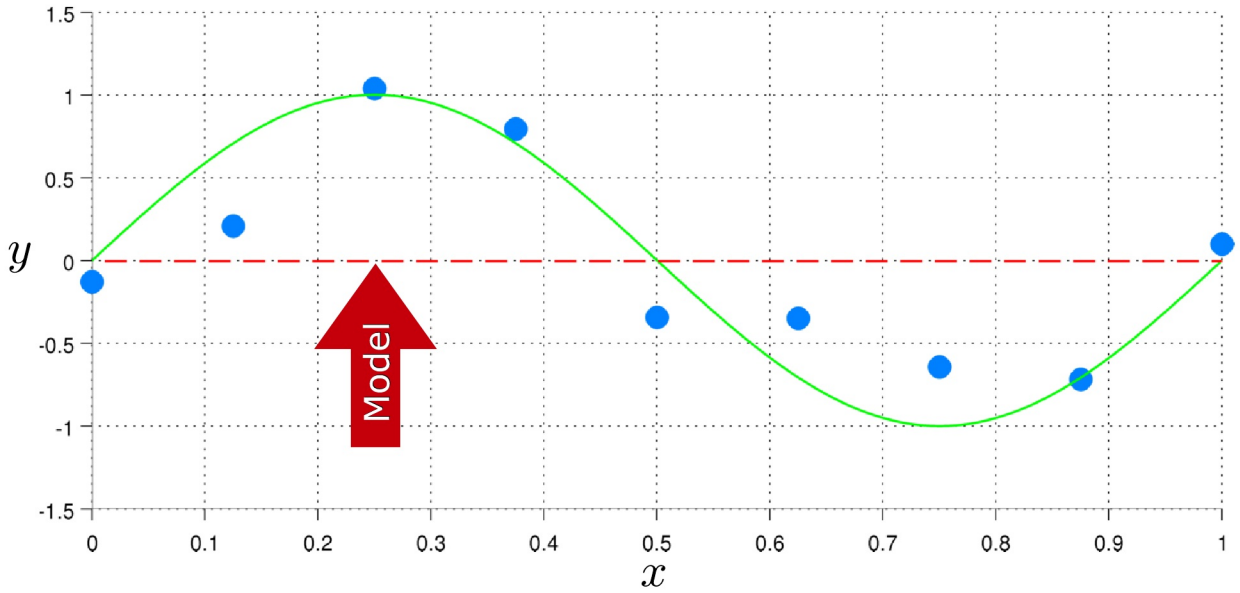
$$= \sum_{k=0}^{K} w_k x_k = \boxed{\boldsymbol{x}^\top \boldsymbol{w}}$$

  - where $x_0 = 1$

Linear regresssion



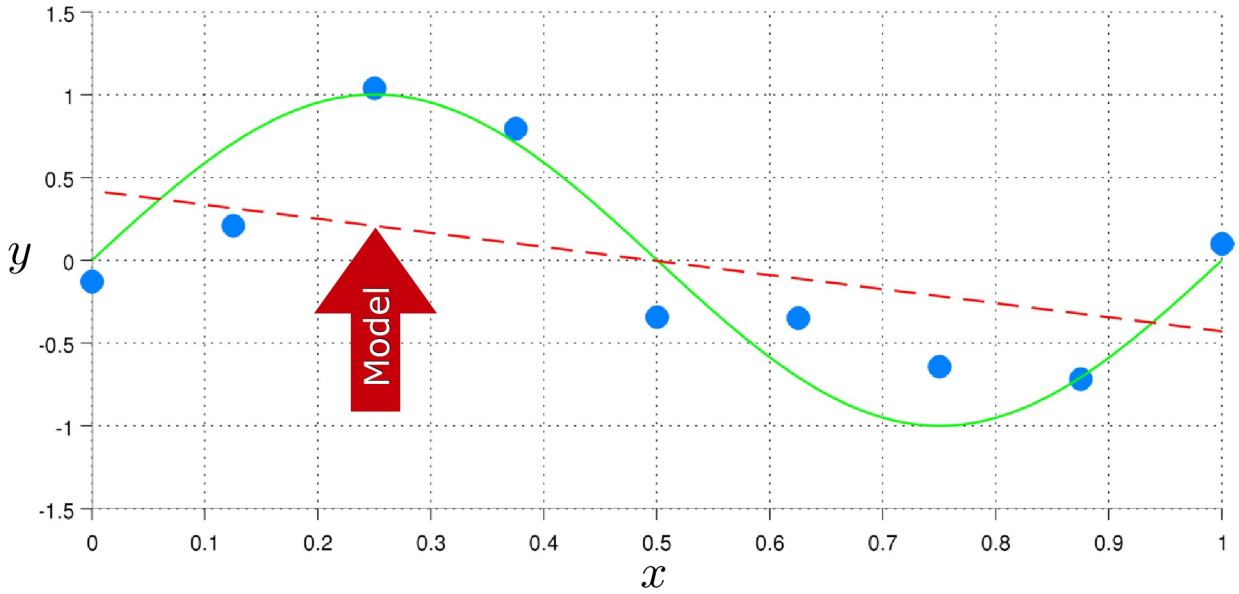
Linear regresssion



Model

$f(x) = w_0$

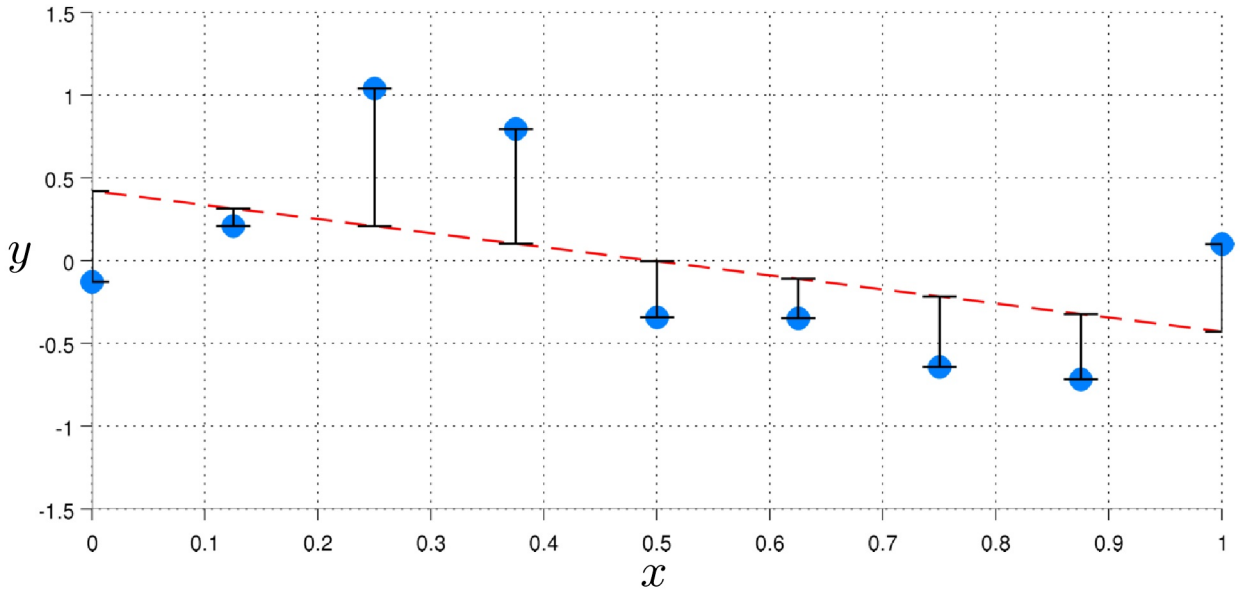
Linear regresssion



Model

$$f(x) = w_0 + w_1 x$$

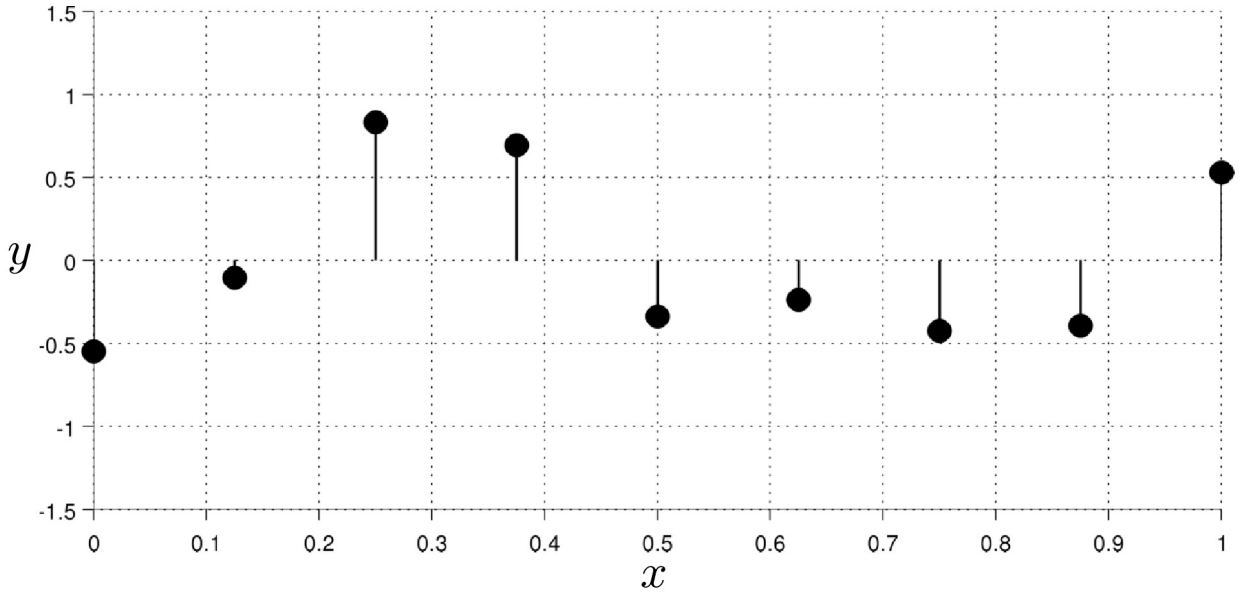
Residual error



Model

$$f(x) = w_0 + w_1 x$$

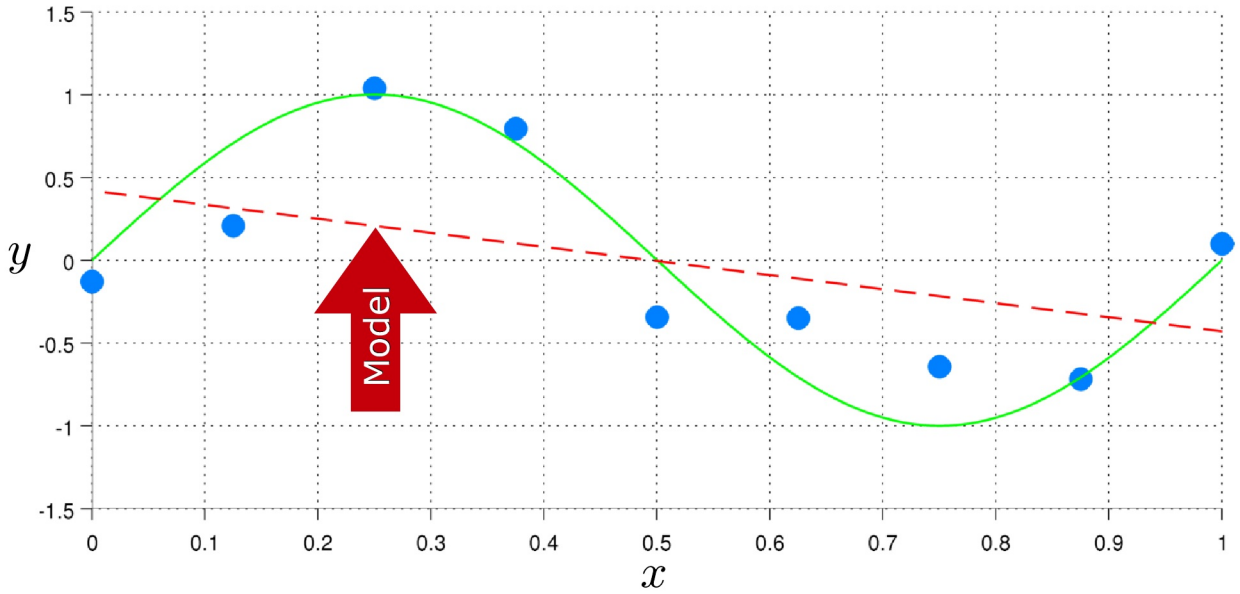
Residual error



Model

$$f(x) = w_0 + w_1 x$$

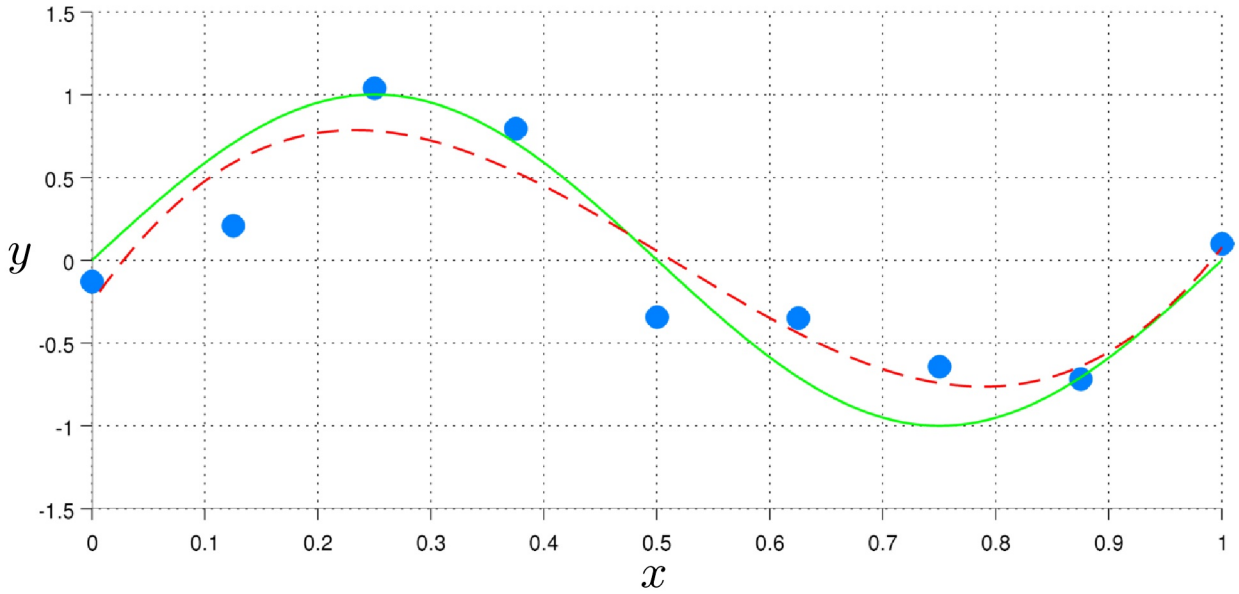
Linear regression



Model

$$f(x) = w_0 + w_1 x$$

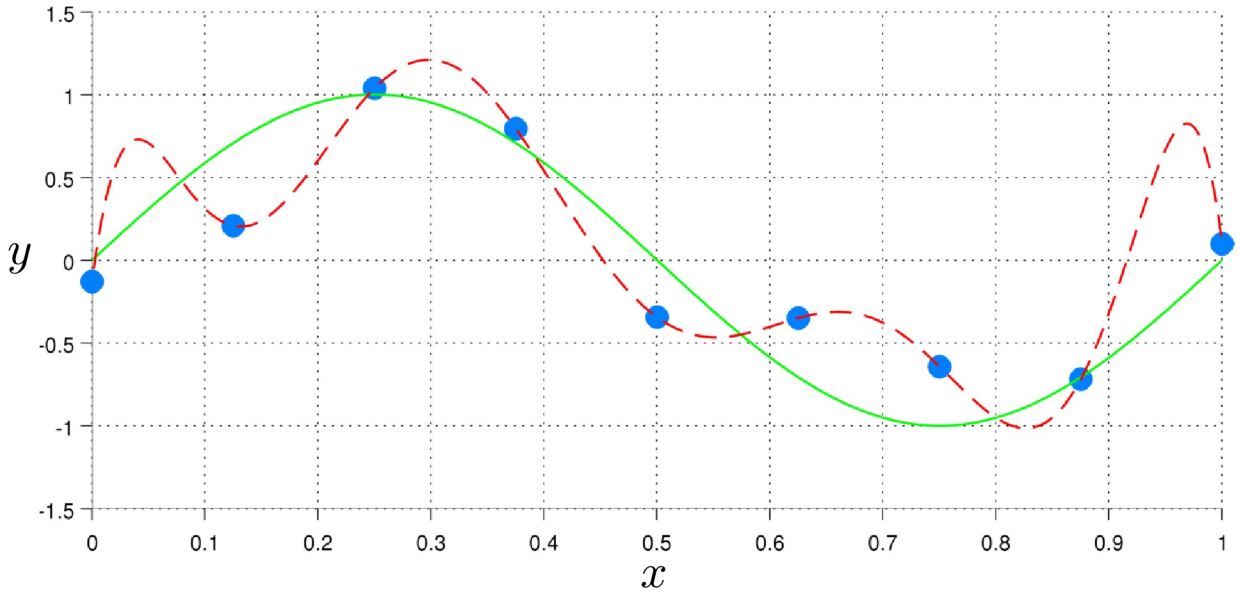
Linear regression



Model

$$f(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

Linear regression

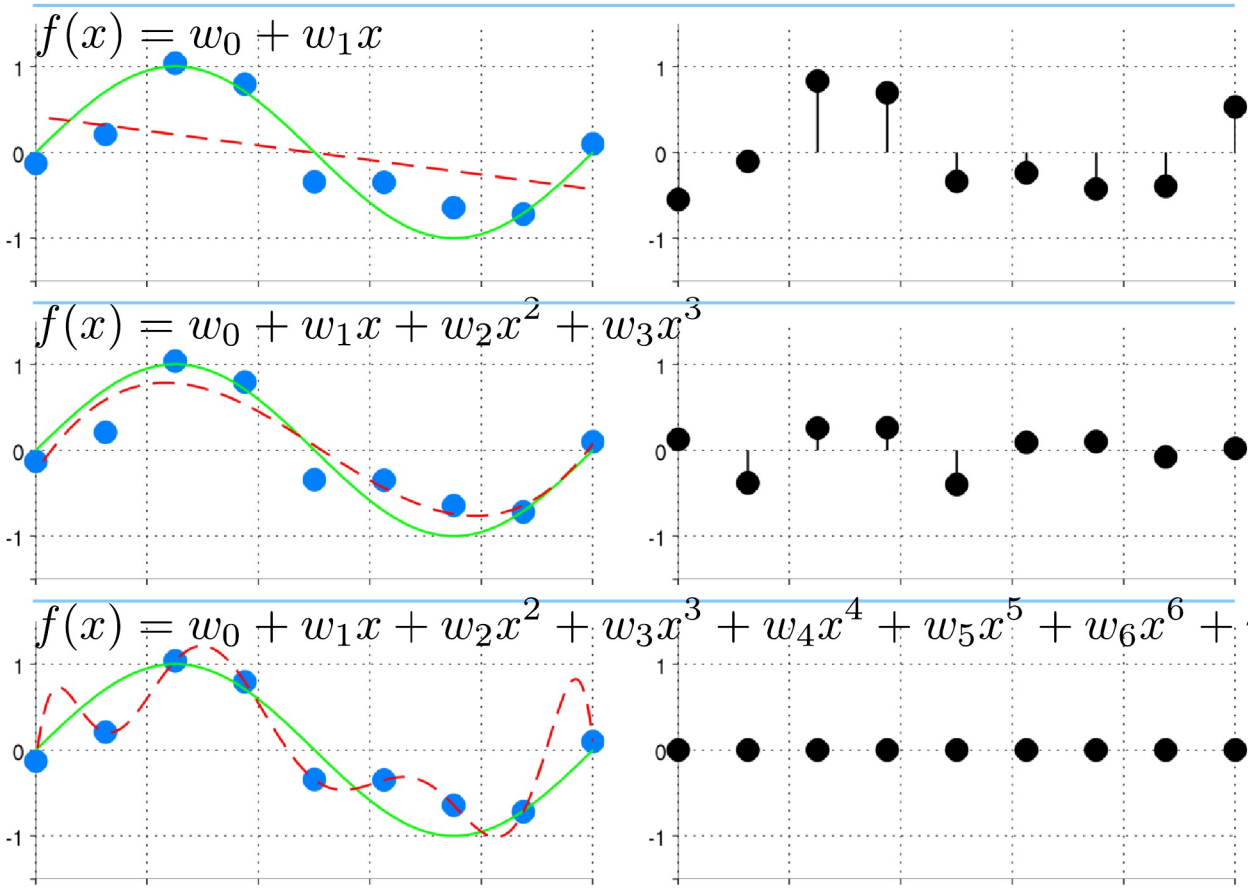


Model

$$f(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 + w_5 x^5 + w_6 x^6 + w_7 x^7 + w_8 x^8$$

Model order

- Which model order
  - Gives the best fit?
  - Do you think is most "correct"?



Learning

Setup: We got some data $(\boldsymbol{y}, \boldsymbol{X})$. We have a way to define a function

$$f(\boldsymbol{x}; \boldsymbol{w}) = \tilde{\boldsymbol{x}}^T \boldsymbol{w} = w_0 + \sum_{k=1}^{M} x_k w_k$$

Learning

Setup: We got some data $(\boldsymbol{y}, \boldsymbol{X})$. We have a way to define a function

$$f(\boldsymbol{x}; \boldsymbol{w}) = \tilde{\boldsymbol{x}}^T \boldsymbol{w} = w_0 + \sum_{k=1}^{M} x_k w_k$$

- **Question:** How do we learn the correct $\boldsymbol{w}$?

Learning

Setup: We got some data $(\boldsymbol{y}, \boldsymbol{X})$. We have a way to define a function

$$f(\boldsymbol{x}; \boldsymbol{w}) = \tilde{\boldsymbol{x}}^T \boldsymbol{w} = w_0 + \sum_{k=1}^{M} x_k w_k$$

- **Question:** How do we learn the correct $\boldsymbol{w}$?
- Answer: For each observation, assume:

$$y_i = f(\boldsymbol{x}_i; \boldsymbol{w}) + \varepsilon_i$$

where ε_i is a normally distributed noise term $\mathcal{N}(\varepsilon_i | \mu = 0, \sigma^2)$. Recall:

$$\mathcal{N}(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Learning

Setup: We got some data $(\boldsymbol{y}, \boldsymbol{X})$. We have a way to define a function

$$f(\boldsymbol{x}; \boldsymbol{w}) = \tilde{\boldsymbol{x}}^T \boldsymbol{w} = w_0 + \sum_{k=1}^{M} x_k w_k$$

- **Question:** How do we learn the correct $\boldsymbol{w}$?
- Answer: For each observation, assume:

$$y_i = f(\boldsymbol{x}_i; \boldsymbol{w}) + \varepsilon_i$$

where ε_i is a normally distributed noise term $\mathcal{N}(\varepsilon_i | \mu = 0, \sigma^2)$. Recall:

$$\mathcal{N}(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- This means that

$$p(y_i | \boldsymbol{x}_i, \boldsymbol{w}) =$$

Learning

Setup: We got some data $(\boldsymbol{y}, \boldsymbol{X})$. We have a way to define a function

$$f(\boldsymbol{x}; \boldsymbol{w}) = \tilde{\boldsymbol{x}}^T \boldsymbol{w} = w_0 + \sum_{k=1}^{M} x_k w_k$$

- **Question:** How do we learn the correct $\boldsymbol{w}$?
- Answer: For each observation, assume:

$$y_i = f(\boldsymbol{x}_i; w) + \varepsilon_i$$

where ε is a noise term from a normal distribution $\mathcal{N}(\varepsilon_i | \mu = 0, \sigma^2)$. Recall:

$$\mathcal{N}(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- This means that (since $\varepsilon_i = y_i - f(\tilde{\boldsymbol{x}}_i, \boldsymbol{w})$)

$$p(y_i | \boldsymbol{x}_i, \boldsymbol{w}) = p(\varepsilon_i | \tilde{\boldsymbol{x}}_i, \boldsymbol{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{((y_i - f(\tilde{\boldsymbol{x}}_i, \boldsymbol{w})) - 0)^2}{2\sigma^2}} = \mathcal{N}(y_i | \mu = f(\tilde{\boldsymbol{x}}_i, \boldsymbol{w}), \sigma^2)$$

Recall from last time: Maximum likelihood learning

- Consider some data $\boldsymbol{X} = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ and $\boldsymbol{y} = y_1, \ldots, y_N$
- Suppose $\boldsymbol{x}_i$ relates to y_i by some parameters $\boldsymbol{w}$
- **Assume**

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) = \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i, \boldsymbol{w}), \quad p(\boldsymbol{w}|\boldsymbol{X}) = p(\boldsymbol{w})$$

- Then

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{X})}{p(\boldsymbol{y}|\boldsymbol{X})} = \frac{\prod_{i=1}^{N} p(\boldsymbol{y}_i|\boldsymbol{w}, \boldsymbol{x}_i)p(\boldsymbol{w})}{p(\boldsymbol{X})}$$

- And maximizing: $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$ is equivalent to

$$\textit{Minimize:} \quad \boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E(\boldsymbol{w})$$

$$E(\boldsymbol{w}) = \frac{1}{N}\left[-\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) - \log p(\boldsymbol{w})\right]$$

Back to the linear model

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w})^2}{2\sigma^2}}$$

Optimal $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$ found as $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E(\boldsymbol{w})$

$$E(\boldsymbol{w}) = \frac{1}{N}\left[-\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) - \log p(\boldsymbol{w})\right]$$

Back to the linear model

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \mathcal{N}(y_i|f(\boldsymbol{x}_i, \boldsymbol{w}), \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - f(\boldsymbol{x}_i;\boldsymbol{w}))^2}{2\sigma^2}}$$

Optimal $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$ found as $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E(\boldsymbol{w})$

$$\begin{aligned} E(\boldsymbol{w}) &= \frac{1}{N}\left[-\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) - \log p(\boldsymbol{w})\right] \\ &= \frac{1}{N}\sum_{i=1}^{N} \frac{(y_i - \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w})^2}{2\sigma^2} = \frac{1}{N}\frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w})^2 \propto \|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}\|^2 \\ \frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} &= 2\tilde{\boldsymbol{X}}^\top(\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}) = 0 \\ &\Rightarrow \boldsymbol{w}^* = (\tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}})^{-1}\tilde{\boldsymbol{X}}^\top \boldsymbol{y} \end{aligned}$$

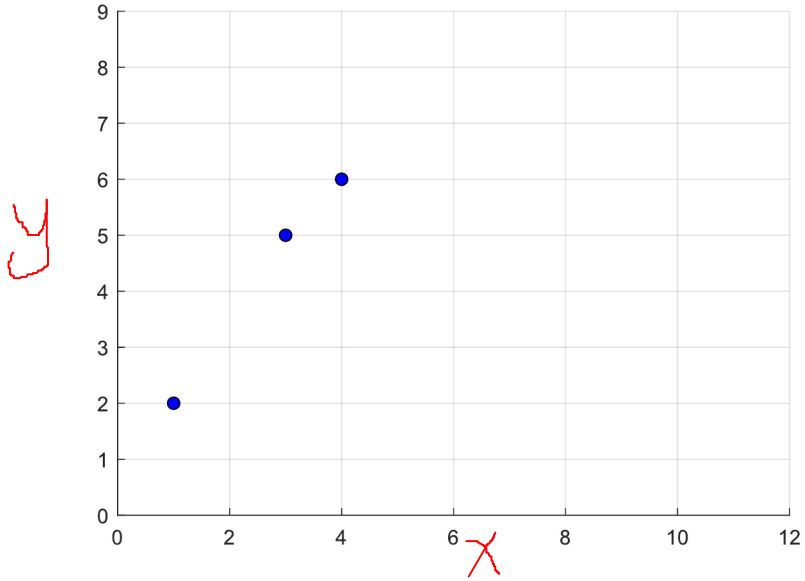
Quiz 2, The linear model

Suppose you observe three points:

$$(x, y) = (1, 2), (3, 5), (4, 6)$$

Knowing what you have learned so far, you first bring these points to the standard format:

$$\boldsymbol{X} = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 2 \\ 5 \\ 6 \end{bmatrix}$$



You wish to train a linear model of the form $y = ax+b$ on this dataset. What is $\boldsymbol{w} = \begin{bmatrix} b \\ a \end{bmatrix}$ Then, compute the prediction of the model at $x = 5$? (the prediction is given as: $y = \tilde{\boldsymbol{x}}^\top \boldsymbol{w}^*$)

A. 6.5

B. 7

C. 7.5

D. 8

E. Don't know.

Recall $\boldsymbol{w}^* = (\tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}})^{-1} \tilde{\boldsymbol{X}}^\top \boldsymbol{y}$

The solution is found by first computing $\boldsymbol{w}$ using the standard formula, and remembering to add a column of ones to $\boldsymbol{X}$ to account for the offset. We get:

$$\boldsymbol{w} \approx \begin{bmatrix} 0.7143 \\ 1.3571 \end{bmatrix}$$

Evaluating the model gives $f(5) = y = 7.5$.

Logistic regression

- Assume we are given $(\boldsymbol{X}, \boldsymbol{y})$, but assume y is *binary*: $y_i = 0, 1$
- An idea is to use the Bernoulli distribution

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \text{Bernouilli}(y_i|\theta_i) = \theta_i^{y_i}(1-\theta_i)^{1-y_i}$$

  Where θ_i depends on $\boldsymbol{w}$ and $\boldsymbol{x}_i$.

- **Problem:** θ_i must belong to the unit interval, but $f(\boldsymbol{x}_i, \boldsymbol{w}) = \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}$ won't
- **Solution:** Assume

$$\theta_i = \sigma(f(\boldsymbol{x}, \boldsymbol{w})), \quad \text{where } \sigma(z) = \frac{1}{1+e^{-z}} \text{ is the } \textbf{logistic sigmoid}$$

  Then

$$-\log p(y_i|\boldsymbol{x}_i, \boldsymbol{w})) =$$

Logistic regression

- Assume we are given $(\boldsymbol{X}, \boldsymbol{y})$, but assume y is *binary*: $y_i = 0, 1$
- An idea is to use the Bernoulli distribution

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \text{Bernouilli}(y_i|\theta_i) = \theta_i^{y_i}(1-\theta_i)^{1-y_i}$$

  Where θ_i depends on $\boldsymbol{w}$ and $\boldsymbol{x}_i$.

- **Problem:** θ_i must belong to the unit interval, but $f(\boldsymbol{x}_i; \boldsymbol{w}) = \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}$ won't
- **Solution:** Assume

$$\theta_i = \sigma(f(\boldsymbol{x}, \boldsymbol{w})), \quad \text{where } \sigma(z) = \frac{1}{1+e^{-z}} \text{ is the } \textbf{logistic sigmoid}$$

  Then

$$\begin{aligned} -\log p(y_i|\boldsymbol{x}_i, \boldsymbol{w})) &= -\log\left[\text{Bern}(y_i|\theta_i = \sigma(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}))\right] = -\log\left[\theta_i^{y_i}(1-\theta_i)^{1-y_i}\right] \\ &= -y_i \log(\theta_i) - (1-y_i)\log(1-\theta_i) \end{aligned}$$

Recall from 10 minutes ago: Maximum likelihood learning

- Consider some data $\boldsymbol{X} = \boldsymbol{x}_1, \ldots, x_N$ and $\boldsymbol{y} = y_1, \ldots, y_N$
- **Assume**

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) = \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i, \boldsymbol{w}), \quad p(\boldsymbol{w}|\boldsymbol{X}) = p(\boldsymbol{w})$$

- Then

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) = \frac{\prod_{i=1}^{N} p(\boldsymbol{y}_i|\boldsymbol{w}, \boldsymbol{x}_i) p(\boldsymbol{w})}{p(\boldsymbol{X})}$$

- Maximizing: $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$ is equivalent to $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E(\boldsymbol{w})$

$$E(\boldsymbol{w}) = \frac{1}{N}\left[-\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) - \log p(\boldsymbol{w})\right]$$

$$= \frac{1}{N}\sum_{i=1}^{N} \left[-y_i \log(\theta_i) - (1 - y_i)\log(1 - \theta_i)\right]$$

where: $\theta_i = \sigma(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}) = \dfrac{1}{1 + e^{-\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}}}$

Quiz 3, Logistic regression

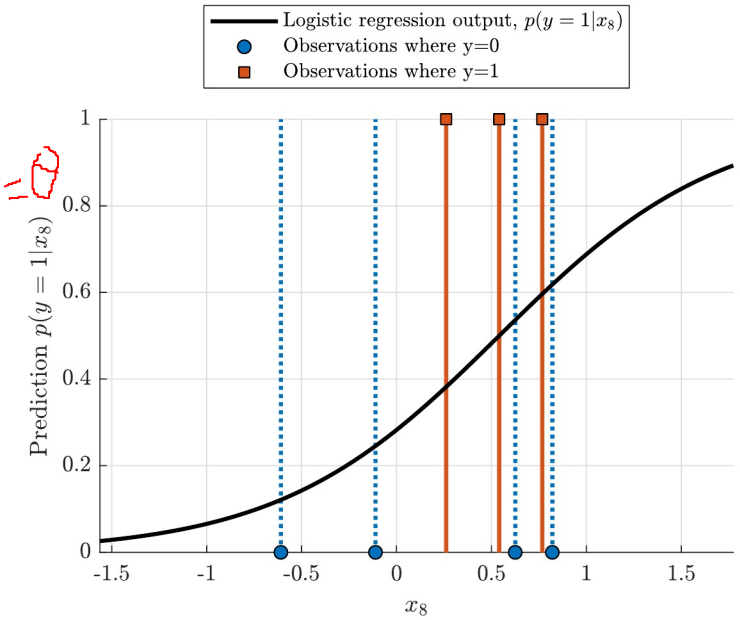


Figure 1: Output of a logistic regression classifier trained on 7 observations from the dataset.

Consider the Avila Bible dataset. We are particularly interested in predicting whether a bible copy was written by copyist 1, and we therefore wish to train a logistic regression classifier to distinguish between copyist one vs. copyist two and three.

To simplify the setup further, we select just 7 observations and train a logistic regression classifier using only the feature x_8 as input (as usual, we apply a simple feature transformation to the inputs to add a constant feature in the first coordinate to handle the intercept term). To be consistent with the lecture notes, we label the output as $y = 0$ (corresponding to copyist one) and $y = 1$ (corresponding to copyist two and three).

In Figure 1 is shown the predicted output probability an observation belongs to the positive class, $p(y = 1|x_8)$. What are the weights?

A. $\begin{bmatrix} -0.93 \\ 1.72 \end{bmatrix}$

B. $\begin{bmatrix} -2.82 \\ 0.0 \end{bmatrix}$

C. $\begin{bmatrix} 1.36 \\ 0.4 \end{bmatrix}$

D. $\begin{bmatrix} -0.65 \\ 0.0 \end{bmatrix}$

E. Don't know.

The solution is easily found by simply computing the predicted $\hat{y} = p(y = 1|x_8)$-value for an appropriate choice of x_8. Notice that

$$p(y = 1|x_8) = \sigma(x_8^T \boldsymbol{w})$$

If we select $x_8 = 1$ and select the weights as in option A we find $p(y = 1|x_8) = 0.69$, in good agreement with the figure. On the other hand, for the weights in option C we obtain $\hat{y} = 0.85$, for D that $\hat{y} = 0.34$ and finally for B that $\hat{y} = 0.06$. We can therefore conclude that A is correct.

Example: Real data

	Breast Cancer		Covertype	
	Acc (train)	Acc (test)	Acc (train)	Acc (test)
Tree Classification	1.0	0.937	1.0	0.939
Tree Classification (minsplit=15)	0.986	0.944	0.974	0.93
Tree Classification (minsplit=30)	0.965	0.881	0.956	0.919
Logistic Regression	0.958	0.951	-inf	-inf
Baseline	0.631	0.615	0.488	0.487

	Boston				California Housing			
	L1 (train)	L1 (test)	L2 (train)	L2 (test)	L1 (train)	L1 (test)	L2 (train)	L2 (test)
Tree Regression	0.0	2.866	0.0	14.349	0.0	0.467	0.0	0.538
Tree Regression (Minsplit 15)	1.18	2.602	2.641	11.954	0.194	0.422	0.097	0.44
Tree Regression (Minsplit 30)	1.851	2.569	7.359	10.955	0.262	0.409	0.16	0.403
Linear Regression	3.253	3.573	22.484	21.889	0.531	0.535	0.521	0.536
Baseline	6.487	7.005	79.373	99.621	0.915	0.91	1.336	1.317

General linear model

$$\text{Linear regr.: } E(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} \|y_i - \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}\|^2$$

$$\text{Logistic regr.: } E(\boldsymbol{w}) = \frac{-1}{N}\sum_{i=1}^{N} \left[y_i \log(\theta_i) + (1-y_i)\log(1-\theta_i)\right], \ \theta_i = \sigma(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w})$$

$$\text{GLM } E(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} d(y_i, g(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}))$$

General linear model

$$\text{Linear regr.: } E(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} \|y_i - \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}\|^2$$

$$\text{Logistic regr.: } E(\boldsymbol{w}) = \frac{-1}{N}\sum_{i=1}^{N} \left[y_i \log(\theta_i) + (1-y_i)\log(1-\theta_i)\right], \ \theta_i = \sigma(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w})$$

$$\text{GLM } E(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} d(y_i, g(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}))$$

We call d the cost function and g the link function. In our examples:

$$\text{Lin.reg. : } d(y,z) = \|y-z\|^2, \qquad z = g(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}) = \tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}$$

$$\text{Log.reg. : } d(y,z) = -y\log z - (1-y)\log(1-z), \quad z = g(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w}) = \sigma(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{w})$$

Resources

http://www2.imm.dtu.dk Our interactive regression demo

(http://www2.imm.dtu.dk/courses/02450/DemoComplexityRegression.html)