

ARCHER Training Courses

Naïve Bayes Classification

EPSRC

NERC SCIENCE OF THE ENVIRONMENT



CRAY
THE SUPERCOMPUTER COMPANY

epcc



Reusing this material



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

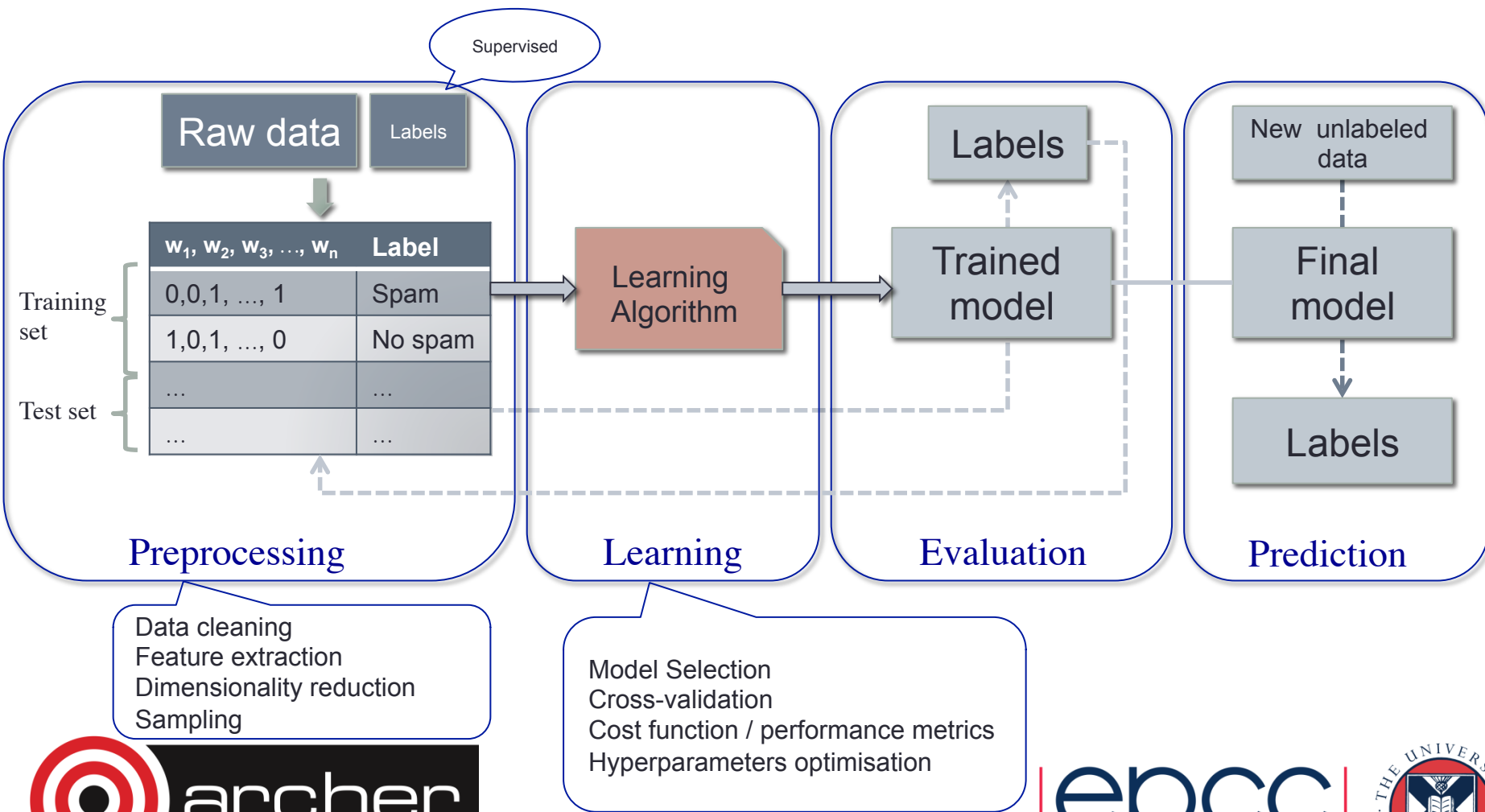
http://creativecommons.org/licenses/by-nc-sa/4.0/deed.en_US

This means you are free to copy and redistribute the material and adapt and build on the material under the following terms: You must give appropriate credit, provide a link to the license and indicate if changes were made. If you adapt or build on the material you must distribute your work under the same license as the original.

Note that this presentation contains images owned by others. Please seek their permission before reusing these images.



Supervised Machine Learning Classification



Revision and question

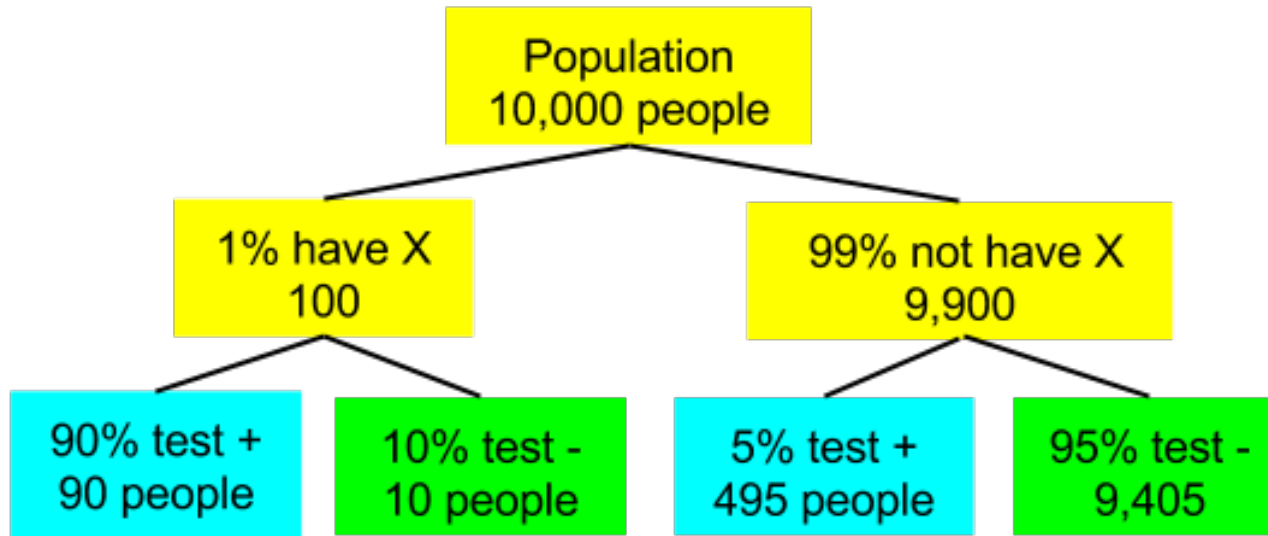
- Test available for condition X
- Sensitivity is 90%
 - For 90% of people with condition X the test will be positive
- Specificity is 95%
 - For 95% of people without condition X the test will be negative



Q. If I take the test and the result is positive what is the probability that I have condition X?

Answer

- It depends on the rate at which condition X occurs in the population



- Thus $90 + 495 = 585$ people test positive, of these 90 have condition X. This is 15.4%

Bayes' Law

- Definitions:

- $p(x)$: probability of event x
- $p(x, y)$: probability of event x and event y (independent or otherwise)
- $p(x|y)$: probability of event x given event y

$$p(x, y) = p(x|y)p(y) = p(y|x)p(x)$$

- Bayes' Law:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} = \frac{p(x|y)p(y)}{p(x|y)p(y) + p(x|\neg y)p(\neg y)}$$

- Bayes' Law shows importance of overall event probability
- Allows to measure $p(x|y)$ and calculate $p(y|x)$

Bayes' Law applied to example

- Bayes' Law

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} = \frac{p(x|y)p(y)}{p(x|y)p(y) + p(x|\neg y)p(\neg y)}$$

- Applied to example:

$$p(X|+) = \frac{p(+|X)p(X)}{p(+|X)p(X) + p(+|\neg X)p(\neg X)}$$

Test sensitivity

Probability of X in whole population

$$p(X|+) = \frac{0.90 \times 0.01}{0.90 \times 0.01 + 0.05 \times 0.99} = 0.154$$

1 - Test specificity

1 - Probability of X in whole population

SPAM classification

- Wish to classify emails as SPAM or not SPAM
- Data for each email:
 - Sender
 - Subject
 - Message contents
 - Lots of other metadata :Sender's IP, time, ...
- Simple method:
 - Classify on presence or absence of keywords
 - Viagra, HPC, purchase, cash etc.
- Is K-NN suitable?
 - No.
 - With 1000s of words there are too many dimensions
 - Dimensions not weighted to relevance



Image courtesy of Nemo under CC0 license.

Probabilistic Classification Model

- We wish to build (train) a statistical model that we can use to classify instances of observed data

- C is set of classes

- e.g. $C = \{\text{Spam}, \text{NotSpam}\}$
- e.g. $C = \{\text{Setosa}, \text{Versicolor}, \text{Virginica}\}$



Setosa



Versicolor



Virginica

- Observed data vector x for n features

$$x = [x_1 \quad x_2 \quad \dots \quad x_n]$$

- Want a model that allows us to calculate $p(c_i|x)$ for all $c_i \in C$
- Then simply choose the class that gives the highest value

Applying Bayes' Th. and being naive

- How do we measure $p(c_i|x)$?

- Bayes' Theorem!

$$p(c_i|x) = \frac{p(c_i)p(x|c_i)}{p(x)}$$

- $p(x)$ is same of all c_i so we can ignore that:

$$p(c_i|x) \propto p(c_i)p(x|c_i)$$

- Now we can be naïve and assume independence between all features:

$$p(c_i|x) \propto p(c_i)p(x_1|c_i)p(x_2|c_i)\dots p(x_n|c_i)$$



Naïve Bayes Classifier and SPAM

- Recall:

$$p(c_i|x) \propto p(c_i)p(x_1|c_i)p(x_2|c_i)\dots p(x_n|c_i)$$

$$p(c_i) = \frac{\text{number of emails of class } c_i}{\text{number of emails}}$$

- Assume the features are the top 10,000 words

$$x_j = \begin{cases} 1, & \text{if the email contains word } w_j \\ 0, & \text{otherwise} \end{cases}$$

$$p(x_j|c_i) = x_j p(w_j|c_i) + (1 - x_j)(1 - p(w_j|c_i))$$

$$p(w_j|c_i) = \frac{\text{number of emails of class } c_i \text{ that contain word } w_j}{\text{number of emails of class } c_i}$$



SPAM example

- Training set 500 spam, 800 non-spam
- “viagra” occurs in 234/500 spam, 10/800 non-spam
- “epcc” occurs in 100/500 spam, 300/800 non-spam
- “Bayes” occurs in 2/500 spam, 56/800 non-spam

$$p(\text{Spam}|\{\text{viagra} = 1, \text{epcc} = 0, \text{Bayes} = 0\})$$

$$\begin{aligned} &\propto p(\text{Spam})p(\text{viagra}|\text{Spam})(1 - p(\text{epcc}|\text{Spam}))(1 - p(\text{Bayes}|\text{Spam})) \\ &= \frac{500}{500 + 800} \cdot \frac{234}{500} \cdot \left(1 - \frac{100}{500}\right) \cdot \left(1 - \frac{2}{500}\right) = 0.143 \end{aligned}$$

The email is
classified as spam

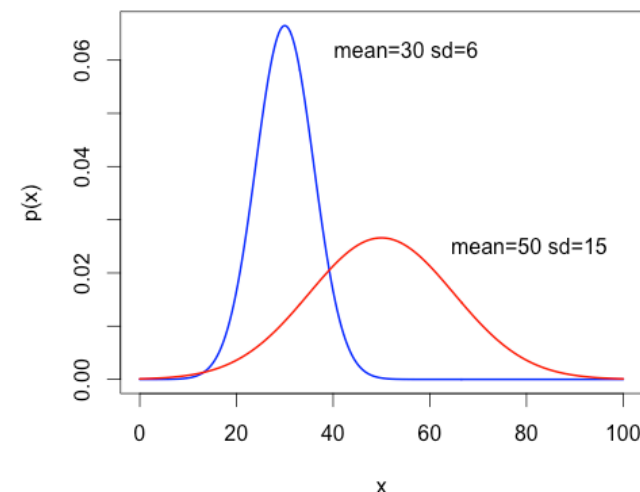
$$p(\text{NonSpam}|\{\text{viagra} = 1, \text{epcc} = 0, \text{Bayes} = 0\})$$

$$\begin{aligned} &\propto p(\text{NonSpam})p(\text{viagra}|\text{NonSpam})(1 - p(\text{epcc}|\text{NonSpam}))(1 - p(\text{Bayes}|\text{NonSpam})) \\ &= \frac{800}{500 + 800} \cdot \frac{10}{800} \cdot \left(1 - \frac{300}{800}\right) \cdot \left(1 - \frac{56}{800}\right) = 0.0045 \end{aligned}$$

Gaussian Naïve Bayes

If we have continuous values rather than discrete then simply model with appropriate distribution

- Gaussian (normal) distribution
 - Mean (μ) and variance (σ^2)
 - σ is called standard deviation
 - 95% of data lies within 1.96 x standard deviations of the mean

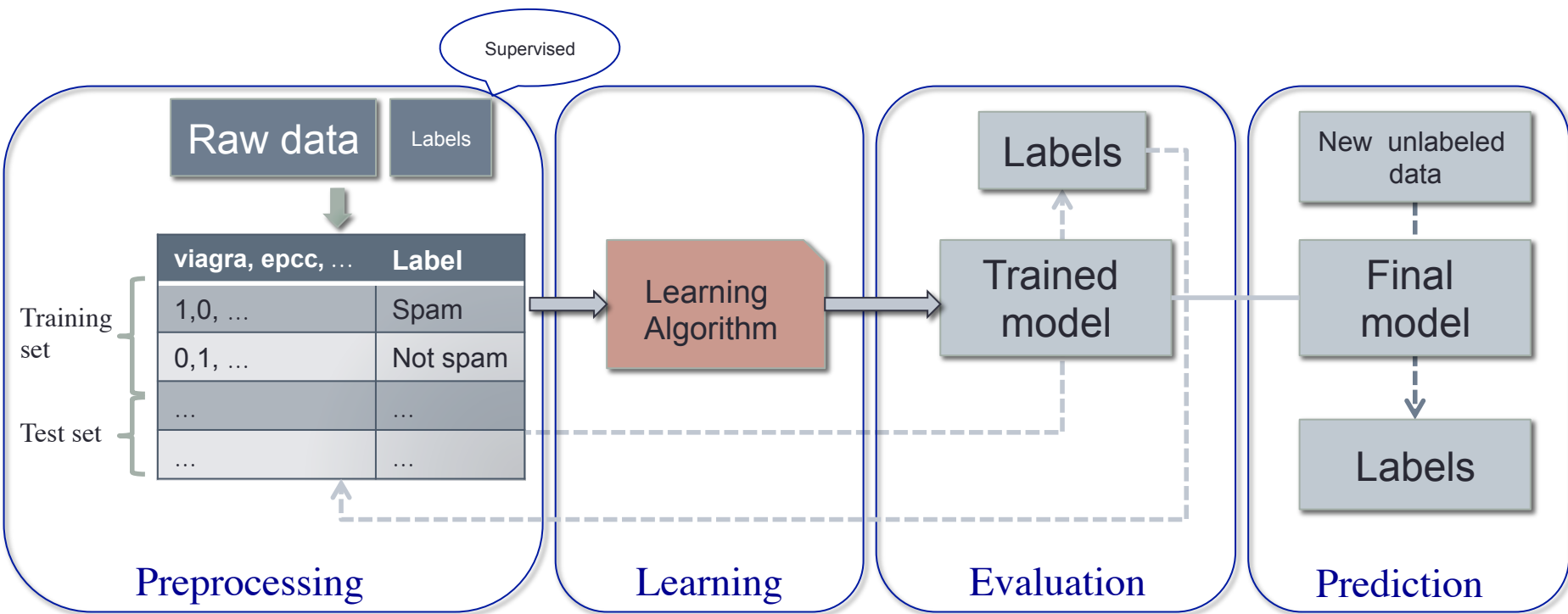


- For each feature calculate mean and variance for each class then:

$$p(x_j|c_i) = \frac{1}{\sigma_{i,j}\sqrt{2\pi}} e^{-\frac{(x_j - \mu_{i,j})^2}{2\sigma_{i,j}^2}}$$

- Mean and variance can be calculated in a single pass through the data.

Supervised Machine Learning Classification



Scan through data building up counts or simple statistical models, e.g. mean and variance for a Gaussian model

Combine per feature probabilities assuming independence. Choose class with maximum score.

Naïve Bayes implementation details

- Discrete values:
 - Learning is just a matter of counting so can be done in a single pass and easy to do in parallel
- Continuous values:
 - Single pass algorithm to compute mean and variance
 - Parallel algorithms to compute mean and variance over distributed datasets
- Map/Reduce
 - Discrete: classic counting problem
 - Map: Key=<class>:<featureName>:<featureValue> Value=1
 - Reduce: standard count reducer and combiner
 - Continuous:
 - Map: Key=<class>:<featureName> Value=<featureValue>
 - Reduce: single pass compute of mean and variance



Applying the model

- Classification stage quick
- Can use logs to convert multiplications into additions
 - Very useful if classifying a large number of items, e.g. voxels in a 3D scan

$$p(\text{NonSpam} | \{\text{viagra} = 1, \text{epcc} = 0, \text{Bayes} = 0\})$$

$$\propto p(\text{NonSpam})p(\text{viagra}|\text{NonSpam})(1 - p(\text{epcc}|\text{NonSpam}))(1 - p(\text{Bayes}|\text{NonSpam}))$$

$$= \frac{800}{500 + 800} \cdot \frac{10}{800} \cdot \left(1 - \frac{300}{800}\right) \cdot \left(1 - \frac{56}{800}\right) = 0.0045$$



Naïve Bayes classification: summary

- Supervised classification
- Model
 - Fairly small, a few numbers for each feature and class combination
- Learning
 - Basic counting to build up model
 - Can often be done in a single pass, so scales well
 - Easily parallelisable
- Naïve assumption does not cause too much harm in practice
- Good first approach to get a base-line performance
- Easy to adjust weighing to get desired balance between sensitivity and specificity

