CSE 4/587

Data Intensive Computing

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Day 17

Naive Bayes (continued)

Announcements and Feedback

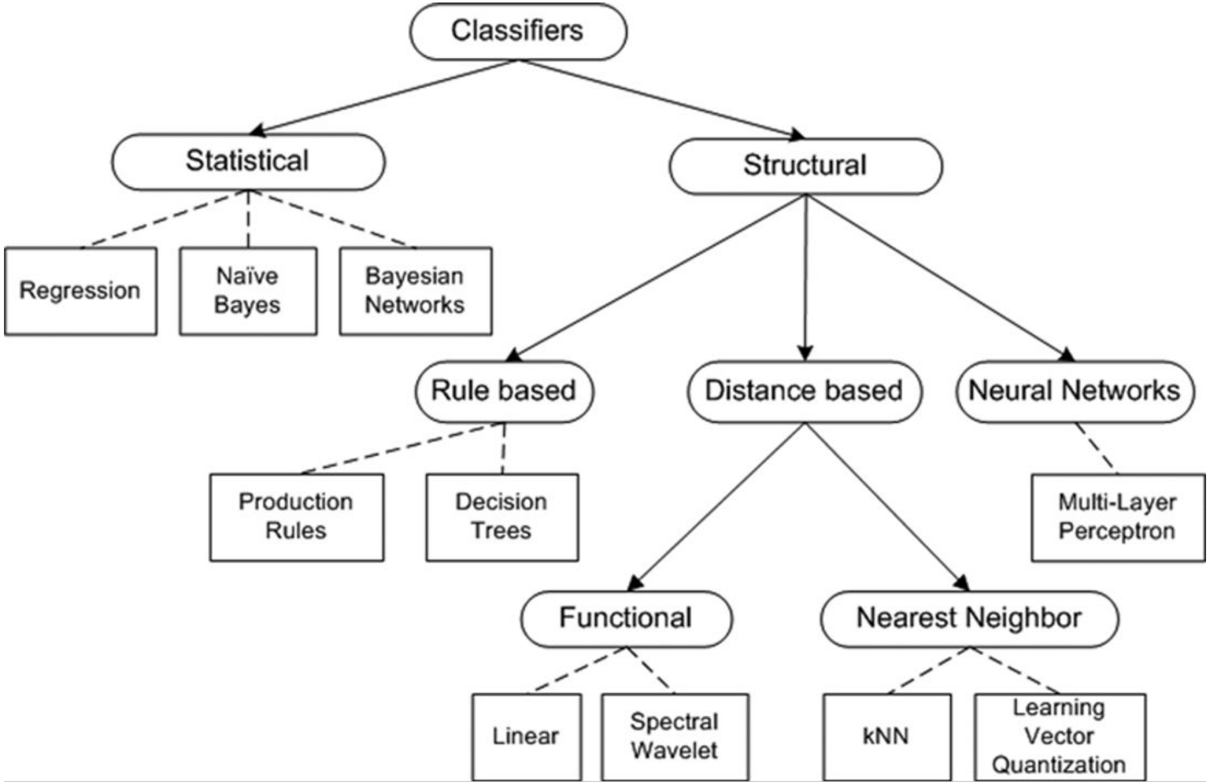
- Read Doing Data Science Chapter 4

Classification of Classification Algorithms

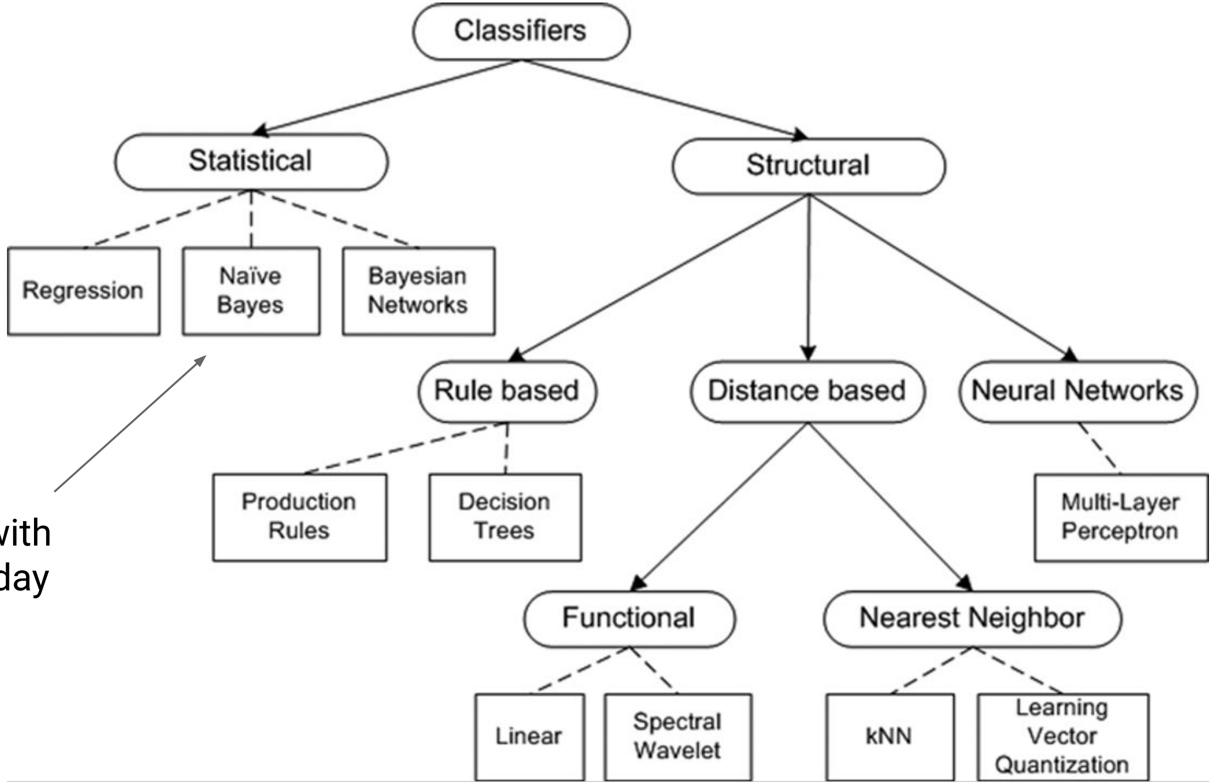
Classification algorithms can be divided into two broad categories:

- Statistical algorithms
  - Regression
  - Probability based classification: Bayes
- Structural algorithms
  - Rule-based algorithms: if-else, decision trees
  - Distance-based algorithm: similarity, nearest neighbor
  - Neural networks

Classification of Classification Algorithms

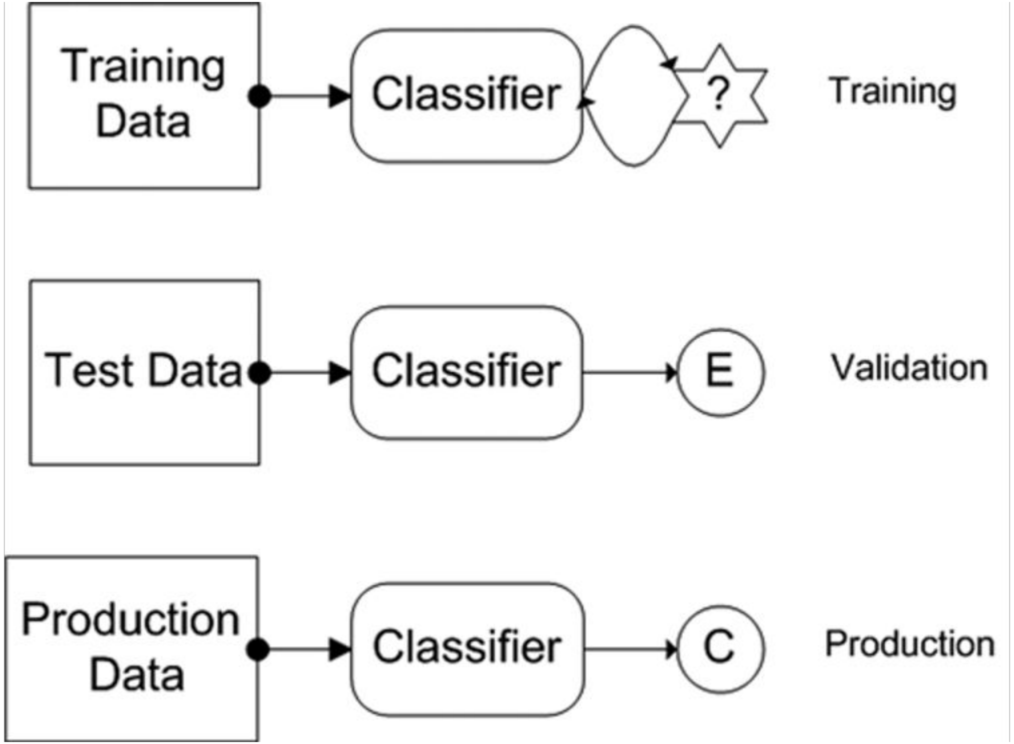


Classification of Classification Algorithms



We'll continue with Naive Bayes today

Life Cycle of Classifiers



Training Stage

- Provide classifier with data points for which we have already assigned an appropriate class
- Purpose of this stage is to determine the parameters of our model

Validation Stage

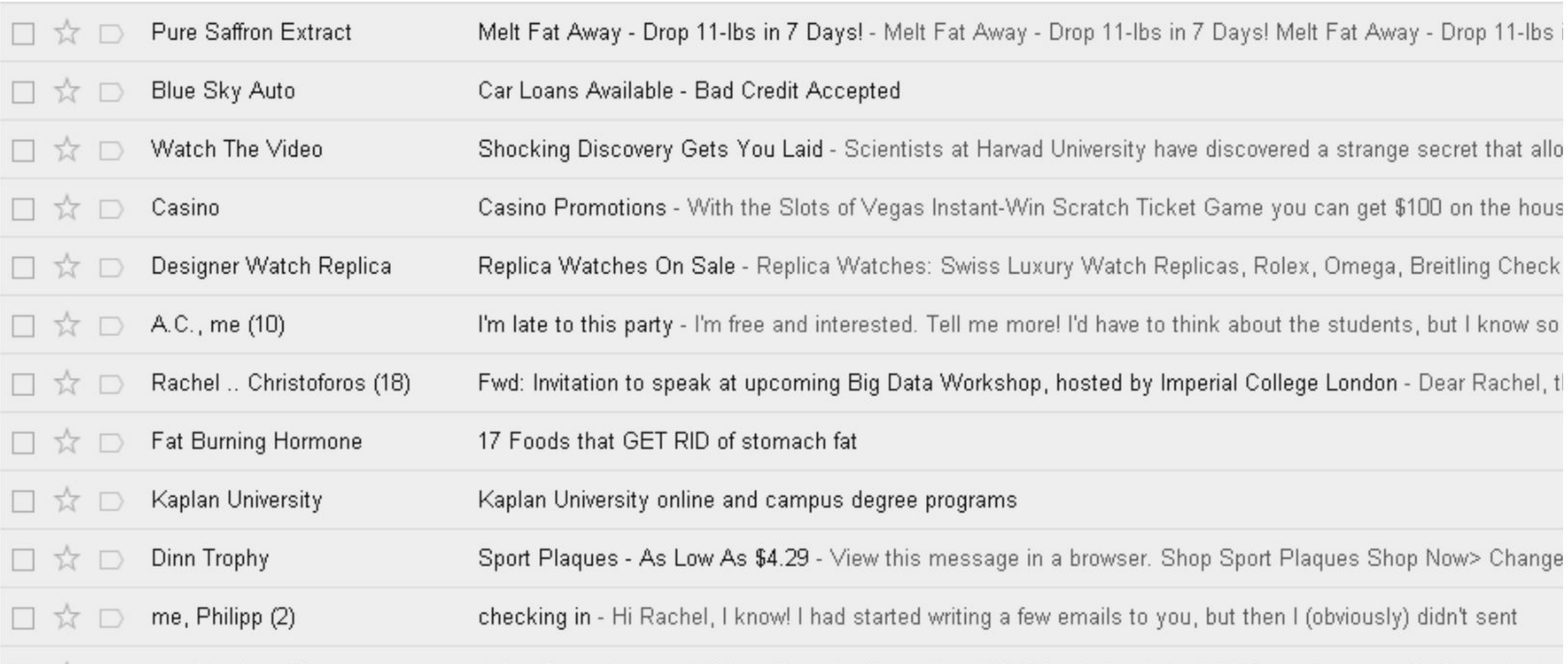
- In the validation stage we validate the classifier to ensure credibility
- Primary goal of this stage is to determine the classification errors
- Quality of the results should be evaluated using various metrics
- Training and testing stages ***may be repeated several times*** before a classifier transitions to the production stage
  - We could evaluate several types of classifiers and pick one or combine all classifiers into a meta-classifier scheme

Production Stage

- Now our classifier(s) are ready for use in a live production system
- We can enhance the results by allowing human-in-the-loop feedback

All steps are repeated as we get more data from the production system.

Motivating Example: Spam Classification



Motivating Example: Spam Classification

Goal: Classify email into spam and not spam (binary classification)

Motivating Example: Spam Classification

Goal: Classify email into spam and not spam (binary classification)

Let's say you get an email saying "You've won the lottery!"

How do we know right away that this email is spam?

Motivating Example: Spam Classification

Goal: Classify email into spam and not spam (binary classification)

Let's say you get an email saying "You've won the lottery!"

How do we know right away that this email is spam?

Idea: The use of certain words, ie lottery, can indicate an email is spam.

What about previous techniques?

So, our features in this problem are individual words…

Can we use linear regression or k-NN to detect spam?

What about previous techniques?

So, our features in this problem are individual words…

Can we use linear regression or k-NN to detect spam?

- Linear regression deals with continuous variables
    - We could use a heuristic to convert a continuous range into a binary range…but we are dealing with a huge number of features

What about previous techniques?

So, our features in this problem are individual words…

Can we use linear regression or k-NN to detect spam?

- Linear regression deals with continuous variables
  - We could use a heuristic to convert a continuous range into a binary range…but we are dealing with a huge number of features
- k-NN works well for low dimensionality…but again, we have a huge number of features (potentially thousands of words).
  - Curse of Dimensionality…

What about previous techniques?

So, our features in this problem are individual words…

Can we use linear regression or k-NN to detect spam?

- Linear regression deals with continuous variables
  - We could use a heuristic to convert a continuous range into a binary range…but we are dealing with a huge number of features
- k-NN works well for low dimensionality…but again, we have a huge number of features (potentially thousands of words).
  - Curse of Dimensionality…

So what do we do?

Naive Bayes

Basic Idea: Make a probabilistic model – have many ***simple rules***, and aggregate those rules together to provide a probability.

Bayes Law and Probability Theory

Basic principle: $P(H \mid E) = P(E \mid H) * P(H) / P(E)$

Posterior probability is proportional to likelihood times prior

- H – hypothesis E – evidence
- **Prior** = probability of the E given H; $P(E \mid H)$
- **Likelihood** = $P(H) / P(E)$
- **Posterior** = Probability of H given E; $P(H \mid E)$

Bayes Law - Spam Classification

Given Bayes Law, how can we start classifying emails as spam?

Bayes Law - Spam Classification

Given Bayes Law, how can we start classifying emails as spam?

Let's start one word at a time:

$$P(spam|word) = P(word|spam) * P(spam) / P(word)$$

Bayes Law - Spam Classification

Given Bayes Law, how can we start classifying emails as spam?

Let's start one word at a time:

Probability that the given word appears in an email

$$P(\mathit{spam}|\mathit{word}) = P(\mathit{word}|\mathit{spam}) * P(\mathit{spam}) / P(\mathit{word})$$

Probability that an email is spam if it contains a given word

Probability that the given word appears in an email known to be spam

Probability that an email is spam

Bayes Law - Spam Classification

We've now boiled our classification problem down to a counting problem:

Given a set of emails that have been classified as spam or not spam (ham):

1. Count number of spam vs ham emails to compute **P(*spam*)**
2. Count number of times the given word, ie lottery, appears in emails to compute **P(*word*)**
3. Count number of times the given word appears in spam emails to compute **P(*word*|*spam*)**

Enron Email Example - DDS Chapter 4

- **Input:** Enron data set containing employee emails
- A small subset chosen for EDA
- 1500 spam, 3672 ham
- Test word is “meeting”
- Running a simple shell script reveals that there are 16 spam emails containing “meeting” and 153 ham emails containing "meeting"
- **Output:** What is the probability that an email containing "meeting" is spam? What is your intuition? Now prove it using Bayes Law…

Enron Email Example - DDS Chapter 4

Enron Email Example - DDS Chapter 4

P(*spam*) = 1500 / (1500+3672) = 0.29

Enron Email Example - DDS Chapter 4

P(*spam*) = 1500 / (1500+3672) = 0.29

P(*ham*) = 1 - **P(*spam*)** = 0.71

Enron Email Example - DDS Chapter 4

P(*spam*) = 1500 / (1500+3672) = 0.29

P(*ham*) = 1 - **P(*spam*)** = 0.71

P(*meeting*|*spam*) = 16/1500 = 0.0106

Enron Email Example - DDS Chapter 4

P(*spam*) = 1500 / (1500+3672) = 0.29

P(*ham*) = 1 - **P(*spam*)** = 0.71

P(*meeting|spam*) = 16/1500 = 0.0106

P(*meeting|ham*) = 153/3672 = 0.0416

Enron Email Example - DDS Chapter 4

P(*spam*) = 1500 / (1500+3672) = 0.29

P(*ham*) = 1 - **P(*spam*)** = 0.71

P(*meeting|spam*) = 16/1500 = 0.0106

P(*meeting|ham*) = 153/3672 = 0.0416

P(*meeting*) = (16+153) / (1500+3672) = 0.0326

Enron Email Example - DDS Chapter 4

P(*spam*) = 1500 / (1500+3672) = 0.29

P(*ham*) = 1 - **P(*spam*)** = 0.71

P(*meeting|spam*) = 16/1500 = 0.0106

P(*meeting|ham*) = 153/3672 = 0.0416

P(*meeting*) = (16+153) / (1500+3672) = 0.0326

P(*spam|meeting*) = **P(*meeting|spam*)*P(*spam*)/P(*meeting*)** = 0.094 (9.4%)

Further Examples

"money": 80% chance of being spam

"viagra": 100% chance

"enron": 0% chance

With one word, we end up overfitting…

Naive Bayes

Basic Idea: Make a probabilistic model – have many ***simple rules***, and aggregate those rules together to provide a probability.

Naive Bayes

Bayes law for each word

Basic Idea: Make a probabilistic model – have many ***simple rules***, and aggregate those rules together to provide a probability.

Naive Bayes

Bayes law for each word

Basic Idea: Make a probabilistic model – have many ***simple rules***, and aggregate those rules together to provide a probability.

Putting It All Together - Naive Bayes

So we've counted and computed probabilities for all words in our input

Putting It All Together - Naive Bayes

So we've counted and computed probabilities for all words in our input

Let's say we have $\boldsymbol{i}$ words. Let $\boldsymbol{x}$ be a vector of size $\boldsymbol{i}$,

where $\boldsymbol{x_j} = \mathbf{1}$ if the $\boldsymbol{j^{th}}$ word is present in an email, **0** otherwise.

Putting It All Together - Naive Bayes

So we've counted and computed probabilities for all words in our input

Let's say we have $\boldsymbol{i}$ words. Let $\boldsymbol{x}$ be a vector of size $\boldsymbol{i}$,

where $\boldsymbol{x_j}$ **= 1** if the $\boldsymbol{j^{th}}$ word is present in an email, **0** otherwise.

Now how do we compute P(*x*|*spam*)?

Once we do this, we can apply Bayes Law to find P(*spam*|*x*)

Naive Bayes

Naive Bayes

Let ***c*** represent the condition that an email is spam

Naive Bayes

Let ***c*** represent the condition that an email is spam

Let $\boldsymbol{x_j} = \mathbf{1}$ if the $\boldsymbol{j^{th}}$ word is in the email

Naive Bayes

Let $\boldsymbol{c}$ represent the condition that an email is spam

Let $\boldsymbol{x_j = 1}$ if the $\boldsymbol{j^{th}}$ word is in the email

Let $\boldsymbol{\theta_{jc}}$ be the probability that an email is spam if it has the $\boldsymbol{j^{th}}$ word

Naive Bayes

Let ***c*** represent the condition that an email is spam

Let $\boldsymbol{x_j}$ **= 1** if the $\boldsymbol{j^{th}}$ word is in the email

Let $\boldsymbol{\theta_{jc}}$ be the probability that an email is spam if it has the $\boldsymbol{j^{th}}$ word

$$p(x|c) = \prod_j \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)}$$

Naive Bayes

Let ***c*** represent the condition that an email is spam

Let $\boldsymbol{x_j}$ **= 1** if the $\boldsymbol{j^{th}}$ word is in the email

Let $\boldsymbol{\theta_{jc}}$ be the probability that an email is spam if it has the $\boldsymbol{j^{th}}$ word

$$p(x|c) = \prod_j \theta_{jc}^{x_j} (1 - \theta_{jc})^{(1-x_j)}$$

$\boldsymbol{\theta_{jc}}$ if the $\boldsymbol{j^{th}}$ word is in the email

Naive Bayes

Let ***c*** represent the condition that an email is spam

Let $\boldsymbol{x_j}$ **= 1** if the $\boldsymbol{j^{th}}$ word is in the email

Let $\boldsymbol{\theta_{jc}}$ be the probability that an email is spam if it has the $\boldsymbol{j^{th}}$ word

$$p(x|c) = \prod_j \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)}$$

$\boldsymbol{\theta_{jc}}$ if the $\boldsymbol{j^{th}}$ word is in the email

$\boldsymbol{1-\theta_{jc}}$ if the $\boldsymbol{j^{th}}$ word is not in the email

Example

"meeting": 1% chance of being in a spam email

"money": 10% chance of being in a spam email

"viagra": 4% chance of being in a spam email

"enron": 0% chance of being in a spam email

What is the probability that a spam email contains "meeting" and "money"?

(but not "viagra" or "enron")

Example

$x = [1,1,0,0]$ $\theta_{1c} = 0.01$ $\theta_{2c} = 0.10$ $\theta_{3c} = 0.04$ $\theta_{4c} = 0.0$

Example

$\boldsymbol{x}$ = [1,1,0,0] $\theta_{1c} = 0.01$ $\theta_{2c} = 0.10$ $\theta_{3c} = 0.04$ $\theta_{4c} = 0.0$

$$p(\boldsymbol{x}|c) = \theta_{1c}\theta_{2c}(1 - \theta_{3c})(1 - \theta_{4c})$$

Example

$\boldsymbol{x} = [1,1,0,0]$ $\theta_{1c} = 0.01$ $\theta_{2c} = 0.10$ $\theta_{3c} = 0.04$ $\theta_{4c} = 0.0$

$$p(\boldsymbol{x}|c) = \theta_{1c}\theta_{2c}(1 - \theta_{3c})(1 - \theta_{4c})$$

$$p(\boldsymbol{x}|c) = 0.01 * 0.1 * 0.96 * 1.0 = 0.00096$$

Example

$x = [1,1,0,0]$ $\theta_{1c} = 0.01$ $\theta_{2c} = 0.10$ $\theta_{3c} = 0.04$ $\theta_{4c} = 0.0$

$$p(x|\mathrm{c}) = \theta_{1c}\theta_{2c}(1 - \theta_{3c})(1 - \theta_{4c})$$

$$p(x|\mathrm{c}) = 0.01 * 0.1 * 0.96 * 1.0 = 0.00096$$

There is a 0.09% chance that this exact vector *x* appears in a spam email

Cleaning it up...

- Multiplying many small probabilities can result in numerical issues
- A common method for avoiding this is to take the log of both side

$$log(p(x|c)) = \sum_j x_j log(\theta_j/(1-\theta_j)) + \sum_j log(1-\theta_j)$$

Cleaning it up...

Many of these terms don't depend on the email and can be precomputed

$$log(p(x|c)) = \sum_j x_j log(\theta_j/(1-\theta_j)) + \sum_j log(1-\theta_j)$$

Cleaning it up...

Many of these terms don't depend on the email and can be precomputed

$$log(p(x|c)) = \sum_j x_j \boxed{log(\theta_j/(1-\theta_j))} + \sum_j log(1-\theta_j)$$

Call this $\boldsymbol{w_j}$

Cleaning it up...

Many of these terms don't depend on the email and can be precomputed

$$log(p(x|c)) = \sum_j x_j \boxed{log(\theta_j/(1-\theta_j))} + \boxed{\sum_j log(1-\theta_j)}$$

Call this $\boldsymbol{w_j}$

Call this $\boldsymbol{w_0}$

Cleaning it up...

Many of these terms don't depend on the email and can be precomputed

$$log(p(x|c)) = \sum_j x_j w_j + w_0$$

The Final Formula

Now given ***p*(*x*|*spam*)** we can use Baye's Law we can compute ***p*(*spam*|*x*):**

$$p(spam|x) = p(x|spam) * p(spam) / p(x)$$

The Final Formula

Now given $p(x|spam)$ we can use Baye's Law we can compute $p(spam|x)$:

$$p(spam|x) = p(x|spam) * p(spam) / p(x)$$

These other two terms are pretty straightforward to compute, and $p(spam)$ is independent of the input email

Naive Bayes

A few notes:

- Occurrences of words are considered independent events
  - Don't care how many times a word appears
  - Don't care about combinations of words
  - This is why it's called "naive"

Extending our Model: Laplace Smoothing

From the previous formula, $\boldsymbol{\theta}_{jc}$ is just a ratio of counts: $\boldsymbol{n}_{jc} / \boldsymbol{n}_j$

Where $\boldsymbol{n}_{jc}$ is the number of times the word appears in a spam email

and $\boldsymbol{n}_j$ is the number of times the word appears in any email

Extending our Model: Laplace Smoothing

From the previous formula, $\boldsymbol{\theta_{jc}}$ is just a ratio of counts: $\boldsymbol{n_{jc}} / \boldsymbol{n_j}$

Where $\boldsymbol{n_{jc}}$ is the number of times the word appears in a spam email

and $\boldsymbol{n_j}$ is the number of times the word appears in any email

This is just an estimate based on our dataset…what if $\boldsymbol{\theta_{jc}}$ = 1 (or 0)?

Extending our Model: Laplace Smoothing

Laplace Smoothing is a technique to avoid these extreme probabilities

Introduce parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ to our computation of $\boldsymbol{\theta}_{jc}$

$$\theta_{jc} = \frac{n_{jc} + \alpha}{n_j + \beta}$$

Extending our Model: Laplace Smoothing

$\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are parameters of your model (just like $\boldsymbol{k}$ for k-NN)

Extending our Model: Laplace Smoothing

$\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are parameters of your model (just like $\boldsymbol{k}$ for k-NN)

Small values for $\boldsymbol{\alpha}, \boldsymbol{\beta}$ will ensure that the distribution of $\boldsymbol{\theta}$ vanishes at 0, 1

Extending our Model: Laplace Smoothing

$\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are parameters of your model (just like $\boldsymbol{k}$ for k-NN)

Small values for $\boldsymbol{\alpha}, \boldsymbol{\beta}$ will ensure that the distribution of $\boldsymbol{\theta}$ vanishes at 0, 1

Larger values will squeeze the distribution even more into the middle

Extending our Model: Laplace Smoothing

$\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are parameters of your model (just like $\boldsymbol{k}$ for k-NN)

Small values for $\boldsymbol{\alpha}, \boldsymbol{\beta}$ will ensure that the distribution of $\boldsymbol{\theta}$ vanishes at 0, 1

Larger values will squeeze the distribution even more into the middle

More data allows you to relax the values of $\boldsymbol{\alpha}, \boldsymbol{\beta}$

Extending our Model: Multiple Classes

What if we want more than two classes?

Example from DDS: Classifying NYTimes articles based on section

Extending our Model: Multiple Classes

What if we want more than two classes?

Example from DDS: Classifying NYTimes articles based on section

Idea: For a given article, compute the probabilities for each class (section), and then classify the article as the one with the highest probability