

“When the Facts Change, I Change My Mind. What Do You Do, Sir?” - Attributed to Keynes or Samuelson

BAYESIAN MACHINE LEARNING

- NAÏVE BAYES

- NAÏVE BAYES



PyData | Silicon Valley 2013

Krishna Sankar,

<http://doubleclix.wordpress.com>

March 18, 2013

THE ROAD LIES PLAIN BEFORE
ME;--TIS A THEME
SINGLE AND OF
DETERMINED BOUNDS;...
- WORDSWORTH, THE PRELUDE

*Bayesian
Topics*

Applications

Problems

*Classification
Naïve Bayes*

*Non
parametric
Bayesian*

*Bayesian
Signal
Processing*

*Reversible
Jump Markov
Chain Monte
Carlo*

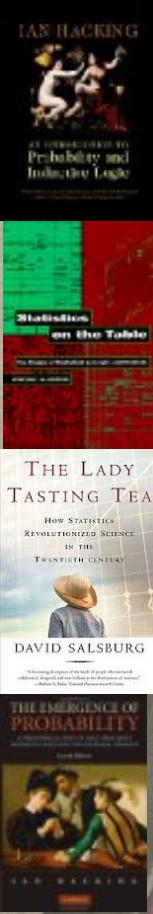
*Bayesian
Hypothesis
testing*

*Bayesian
Networks*

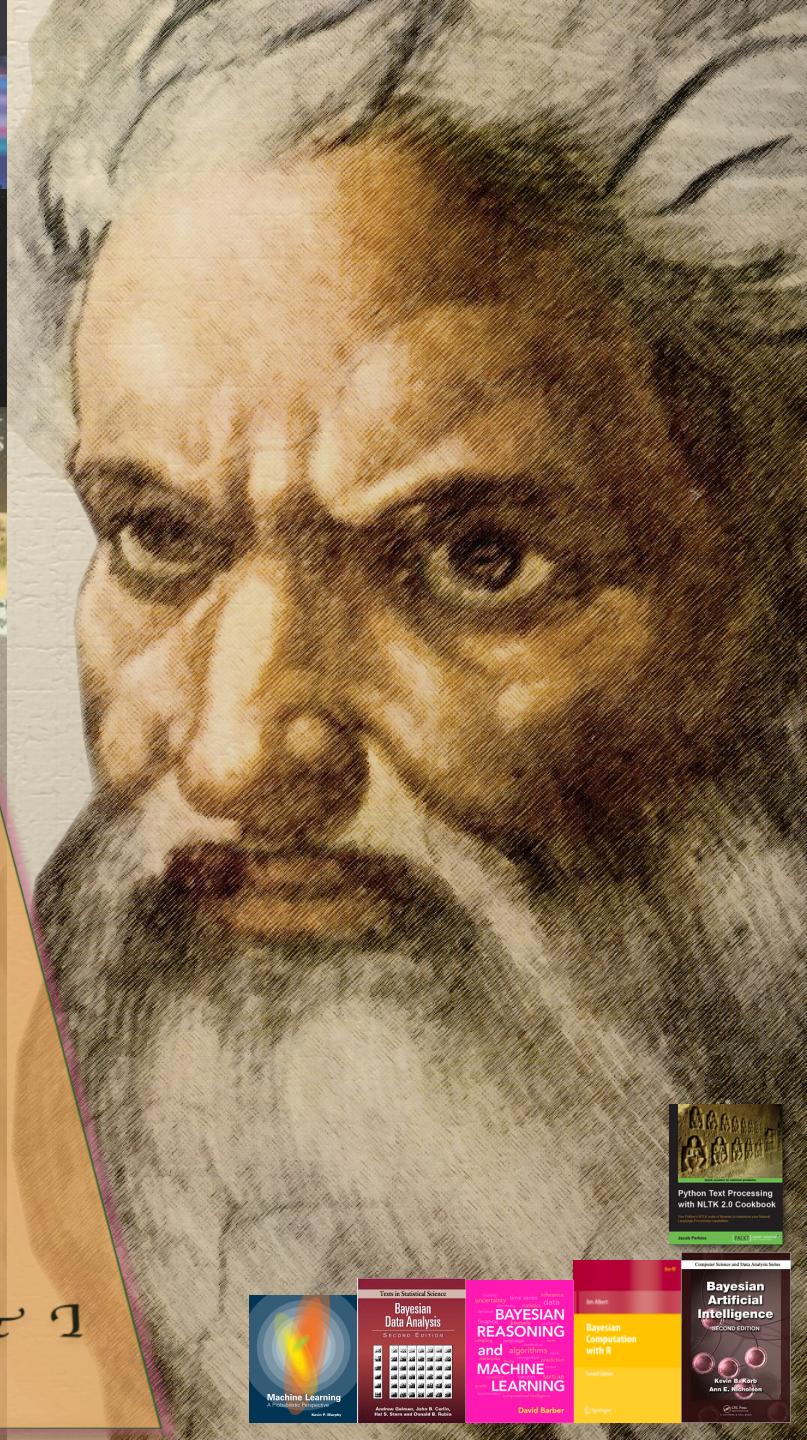
Agenda

- Focus on a small slice of the Bayesian Domain
 - Classification
- Work on a couple of problems using Python libraries

*"Learn you a Bayesian
for Great Good!"*



THANKS TO ... THE GIANTS WHOSE SHOULDERS I AM STANDING ON



- Prof. Allen Downey
 - Pycon Bayesian tutorial as well as his excellent books
- Prof. Andrew Moore
- Prof. Tom Mitchell/CMU
- Prof. Andrew Gelman/Columbia
- Corey Chivers/Montreal R Users Group
- Oscar Bonilla
- Calum Miller
- Olivier Grisel
- Paul Graham
- Prof. Todd Ebert
- Jacob Perkins a.k.a. StreamHacker for his excellent code
- Ref. URLs at the end
- Any omission is inadvertent & I will correct them ...



étude

An étude (/ 'erju:d /; French pronunciation: [e'tydy], a French word meaning *study*) is an instrumental musical composition, usually short and of considerable difficulty, usually designed to provide practice material for perfecting a particular musical skill. The

We will focus on
“short”, “acquiring
skill” & “having
fun”!



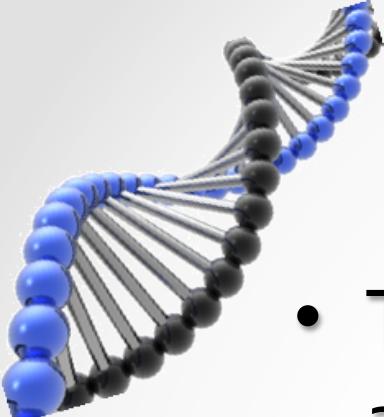
Frédéric Chopin's Étude Op. 10, No. 2: a rapid chromatic scale in the right hand is used to develop the weaker fingers of the right hand. Most études are written to perfect a particular technical skill.





Agenda

- Bayesian Fundamentals (25) 1:15-1:40
 - Bayesian Theorem, Probability (15)
 - Solve 3 problems by hand (10)
 - Python – Libraries + Install(5) 1:35-1:40
 - Classification Problems using Bayes – Intro (10) 1:40-1:50
 - Dr.Seus Classification (10) Hands on – 1:50-2:00
 - Break (10) 2:00-2:10
 - Spam Filtering – Naïve Bayes (30) 2:10-2:40
 - Explanation(10)
 - Hands on – Bayes Spam Classifier (10)
 - Hands-on – Evaluation, Precision & Recall (10)
 - Q&A(10) 2:40-2:50
- 



Pragmatix (1 of 3)

- The hands-on programs use Python 2.x and the nltk stack
 - [Installing epd free is a good idea](#)
 - http://www.enthought.com/products/epd_free.php
 - But just pip install nltk should work
- Did you attend Prof. Downey's tutorial last week ?

PyCon 2013: Bayesian Statistics Made Simple

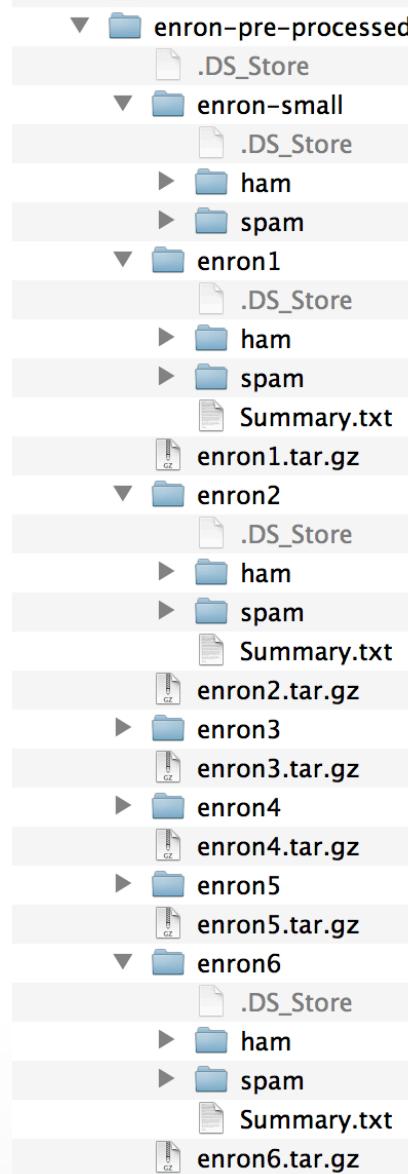
Bayesian Statistics Made Simple

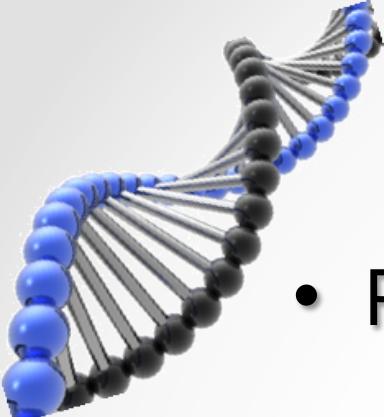
Allen B. Downey
Olin College



Pragmatix (2 of 3)

- Data
 - We will use the Enron spam dataset
 - Background : <http://csmining.org/index.php/enron-spam-datasets.html>
 - The files are also available at <http://labs-repos.iit.demokritos.gr/skel/i-config/>
- We will work with the data in the enron-pre-processed folder
- FYI, full Enron dataset is available at <http://www.cs.cmu.edu/~enron/>
- With attachments is available at <http://www.edrm.net/resources/data-sets/edrm-enron-email-data-set-v2>
 - It is ~210 GB, aws public dataset





Pragmatix (3 of 3)

- **Programs**
 - The python code is in Github
 - <git@github.com:xsankar/pydata.git>
 - <https://github.com/xsankar/pydata>
 - Three files
 - cat_in_the_hat-xx.py
 - spam-01.py
 - Spam-02.py
- **This slide set is in slideshare**
 - <http://www.slideshare.net/ksankar/pydata-19>)
 - as well as in the github repository



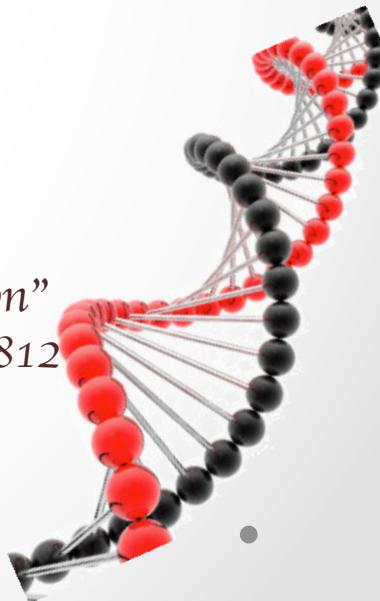


<http://www.google.com/doodles/douglas-adams-61st-birthday>

Bayesian Fundamentals

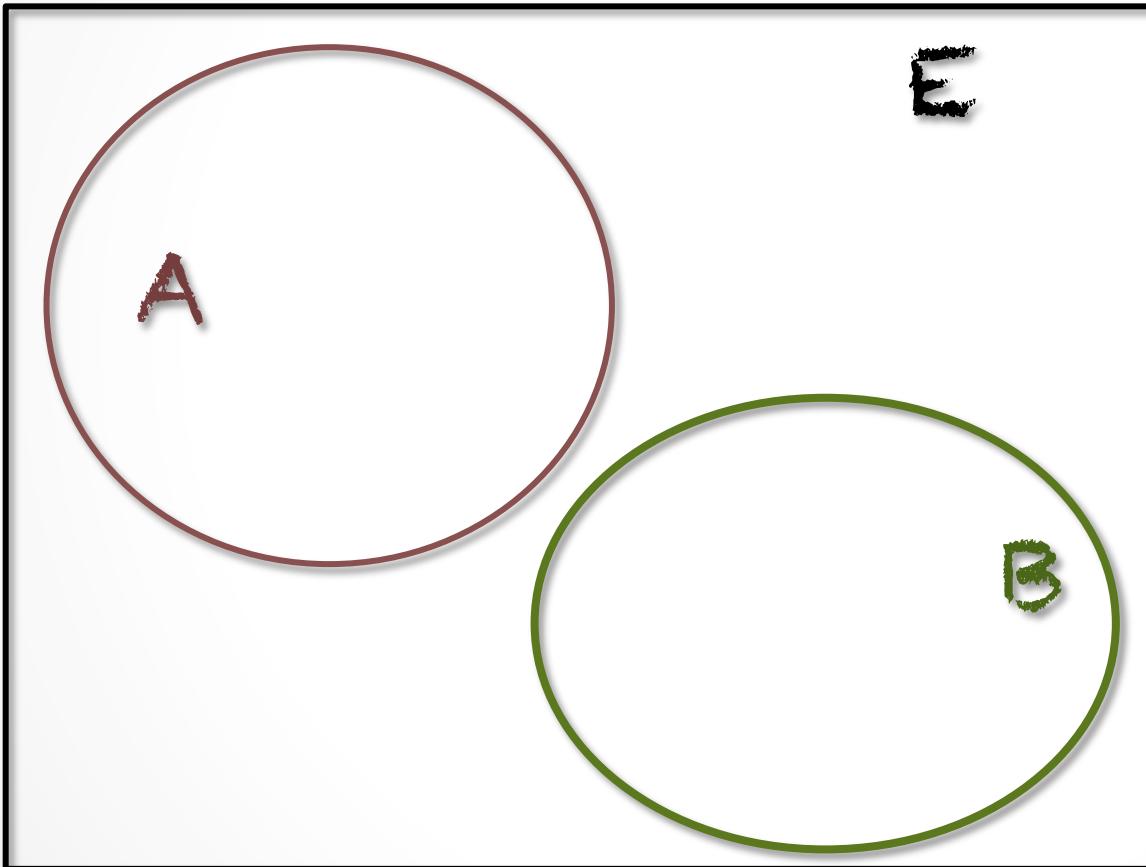
...

“Probability theory is nothing but common sense reduced to calculation”
- Pierre Laplace, 1812





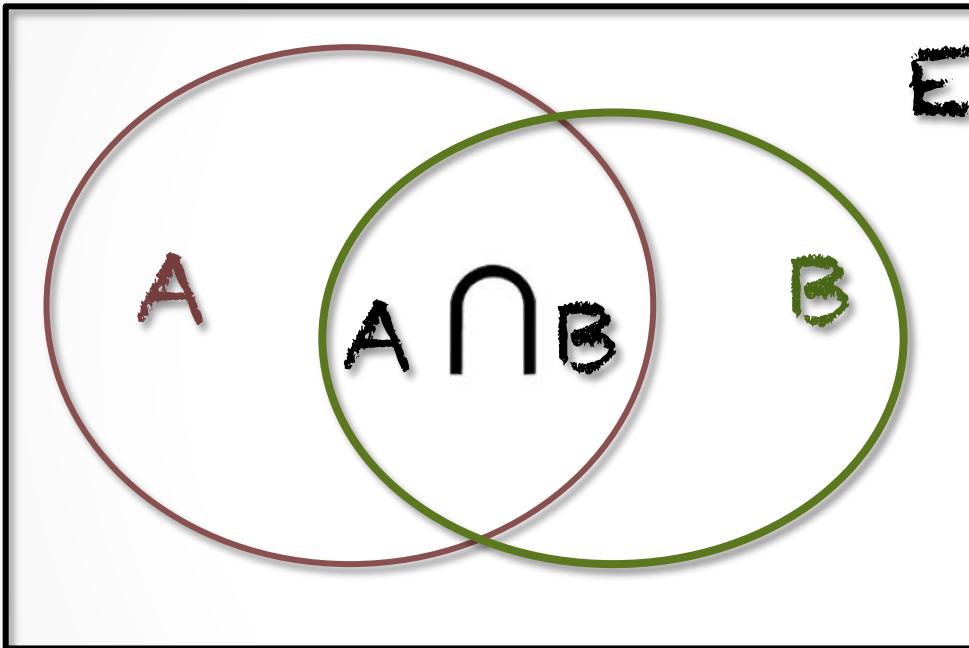
Independent Events



- E = Event Space
- A,B = Event sets
- No possibility of both A & B occurring



Overlapping Event Sets



- E = Event Space
- A, B = Event sets
- Area $A \cap B$ where both occur





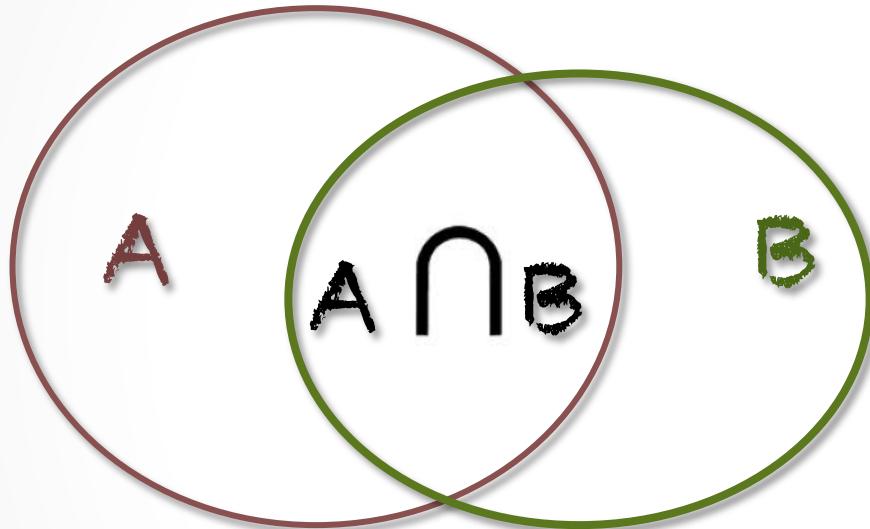
Basics

- $P(A)$ = Probability of event A
 - $P(B)$ = Probability of event B
 - $P(A \wedge B)$ = Probability of events A & B
 - $P(A|B)$ = Conditional probability of event A given that B has happened
 - Definitions (Very important to internalize)
 - Joint Probability - $P(A \wedge B)$
 - Conditional Probability - $P(A|B)$
 - Marginal Probability – Unconditional Probability of one event occurring – $\sum P(A|B) * P(B)$
- 



Overlapping Event Sets

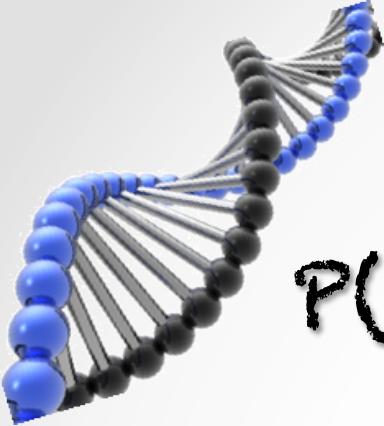
- E = Event Space
- A,B = Event sets



$$\begin{aligned} P(A \cap B) &= P(A) \cdot P(B|A) \\ &= P(B) \cdot P(A|B) \end{aligned}$$

1





$$P(A) \cdot P(B|A) = P(B) \cdot P(A|B)$$

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{P(B)}$$

2

Thomas Bayes



Portrait used of Bayes in the 1936 book *History of Life Insurance*; it is dubious whether it actually depicts Bayes.^[1] No earlier portrait or claimed portrait survived.

Born c. 1701
London, England

Died 7 April 1761 (aged 59)
Tunbridge Wells, Kent, England

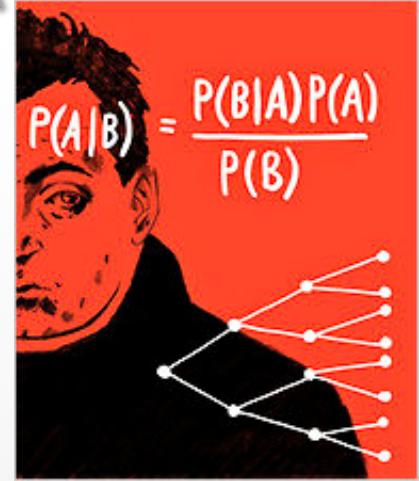
Nationality English

Signature

T. Bayes.

Bayes Theorem

"Inverse Inference Rule"





The paper
that started
it all ...

LII. *An Essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr. Bayes, F. R. S. communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S.*

Dear Sir,

Read Dec. 23,
1763. I Now send you an essay which I have found among the papers of our deceased friend Mr. Bayes, and which, in my opinion, has great merit, and well deserves to be preserved. Experimental philosophy, you will find, is nearly interested in the subject of it; and on this account there seems to be particular reason for thinking that a communication of it to the Royal Society cannot be im-

PROBLEM.

Given the number of times in which an unknown event has happened and failed: *Required* the chance that the probability of its happening in a single trial lies somewhere between any two degrees of probability that can be named.

<http://rstl.royalsocietypublishing.org/content/53/370.short>

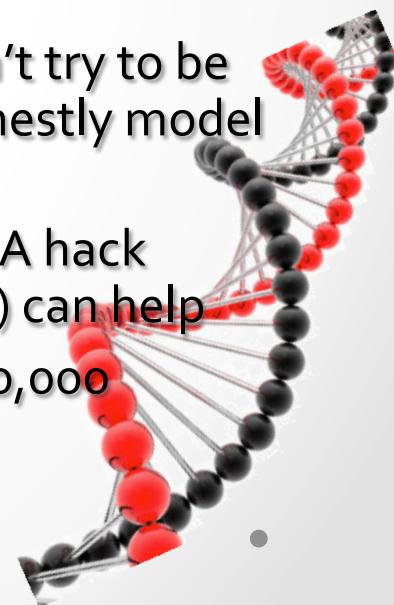
http://en.wikipedia.org/wiki/An_Essay_towards_solving_a_Problem_in_the_Doctrine_of_Chances



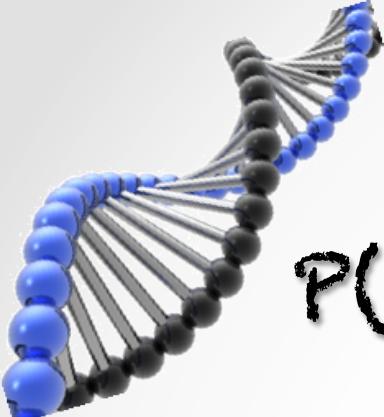


Utility Of Bayes Theorem

- “Inference is a Big deal”¹
 - I have this evidence, what is the probability that this conclusion is true (based on previous experience)
 - Establish
 - Probabilities & Relationships
 - For
 - Inference & Prediction
- Bayes Classifiers¹
 - Rather Technical Complaint: Bayes Classifiers don't try to be maximally discriminative---they merely try to honestly model what's going on
 - Zero probabilities are painful for Joint and Naïve. A hack (justifiable with the magic words “Dirichlet Prior”) can help
 - Naïve Bayes is wonderfully cheap. And survives 10,000 attributes cheerfully!



¹<http://www.autonlab.org/tutorials/prob18.pdf>, Prof. Andrew Moore



Bayes Theorem - Formal

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

2

Assume we have observed occurrences of X & θ in an event set. Later we want to know given X (Data), what is the probability that θ (Hypothesis) has occurred.

Posterior

$$P(\theta|x)$$

Likelihood

$$P(x|\theta)$$

Prior

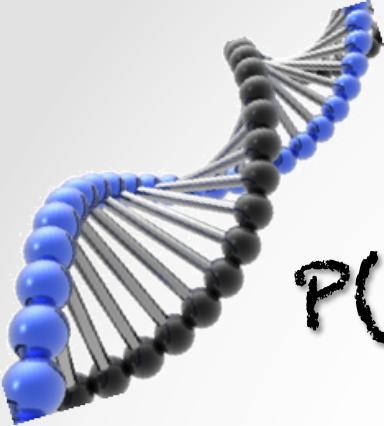
$$P(\theta)$$

$$P(x)$$

3

Normalizing
Constant (a.k.a
Marginal
Probability)





Bayes Theorem – 2nd form

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x)}$$

3

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x|\theta) \cdot P(\theta) + P(x|\sim\theta) \cdot P(\sim\theta)}$$

4

- Marginal Probability ie unconditional probability
- Properties
 - Exclusive & Exhaustive





Bayes Theorem – 2nd (extended) form

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x)}$$

3

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{\sum_m P(x|\theta_m) \cdot P(\theta_m)}$$

5

- When $m = 2$, we get equation 4.

In many cases (for eg.argmax), $P(X)$ can be avoided

$$P(\theta|x) \propto P(x|\theta) \cdot P(\theta)$$

6





Understanding Uncertainty

Home | Blog | Articles | Videos | Animations | Guest Articles | Links | About Us

Home » Blogs » david's blog

Court of Appeal bans Bayesian probability (and Sherlock Holmes)



Submitted by david on Mon, 02/25/2013 - 09:26

 ...when you have eliminated the impossible, whatever remains, however improbable, must be the truth
(Sherlock Holmes in *The Sign of the Four*, ch. 6, 1890) //

In a [recent judgement](#) the English Court of Appeal has not only rejected the Sherlock Holmes doctrine shown above, but also denied that probability can be used as an expression of uncertainty for events that have either happened or not.

- Featured Content -

Main menu

- Post your coincidence
- Read coincidence stories
- Contact us

“..Whilst the chance of a future event happening may be expressed in percentage terms the same is not true of past events. It cannot be said that there is a particular percentage chance that a particular event happened. The event either happened or it did not...”

“.. the “balance of probabilities” standard when expressed mathematically as “50 + % probability” carried with it a “danger of pseudo-mathematics ..”

Ref : <http://understandinguncertainty.org/court-appeal-bans-bayesian-probability-and-sherlock-holmes>
<http://www.lexology.com/library/detail.aspx?g=471d7904-20d2-4fdb-a061-8d49b10de60d&l=7HfVPC65>
<http://www.12kbw.co.uk/cases/commentary/id/161/>





Good References

- <http://yudkowsky.net/rational/bayes> has an excellent write-up on Bayes theorem.
 - Branden Fitelson's PhD thesis "Studies in Bayesian Confirmation Theory [<http://fitelson.org/thesis.pdf>] is a good read
 - <http://plato.stanford.edu/entries/bayes-theorem/>
 - <http://plato.stanford.edu/entries/bayes-theorem/supplement.html>
- 



Let us solve some problems by hand

- Strategy
 - Choose representation of hypothesis
 - Choose representation for Data
 - Write likelihood function
 - Calculate
- I don't like cancer problems, let us try to get some happier problems



Strategy : <http://www.greenteapress.com/thinkbayes/thinkbayes.pdf>

Problems : http://people.hofstra.edu/Stefan_Waner/RealWorld/tutorialsf3/frames6_5.html

http://people.hofstra.edu/Stefan_Waner/RealWorld/tutorialsf3/unit6_6.html



Problem #1

"Of Colossal Conglomerate's 16,000 clients, 3200 own their own business, 1600 are "gold class" customers, and 800 own their own business and are also "gold class" customers. What is the probability that a randomly chosen client **who owns his or her own business is a "gold class" customer?**"



Hint : Create a 4×4 Matrix



Problem #1

"Of Colossal Conglomerate's 16,000 clients, 3200 own their own business, 1600 are "gold class" customers, and 800 own their own business and are also "gold class" customers. What is the probability that a randomly chosen client who owns his or her own business is a "gold class" customer?"

	Owner	!Owner	
Gold	800		1600
! Gold			
	3200		16,000





Solution – Intuition i.e.Table form

	Owner	!Owner	
Gold	800	800	1600
! Gold	2400	12,000	
	3200	12,800	16,000

$$P(O \wedge G) = 800/3200 = 0.25$$





Solution – Bayes

	Owner	!Owner	
Gold	800	800	1600
! Gold	2400	12,000	
	3200	12,800	16,000

$$\begin{aligned} P(G|O) &= \frac{P(O|G) \cdot P(G)}{P(O)} \\ &= \frac{(800/1600)(1600/16000)}{(3200/16000)} \\ &= 800/3200 \\ &= 0.25 \end{aligned}$$


3



Problem #2

A test for a disorder has the following properties. If you have the disorder, the probability that the test returns positive is 0.999. If you don't have the disorder the probability that the test returns positive is 0.004. Assume that 3% of the population has the disorder. If a person is randomly chosen from the population and tested, and the **test returns positive, what is the probability that the person has the disorder?**

Hint : Use Bayes 2nd form

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x|\theta) \cdot P(\theta) + P(x|\sim\theta) \cdot P(\sim\theta)}$$



4

θ = Has disorder,

x = Positive result





Solution – Problem #2

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x|\theta) \cdot P(\theta) + P(x|\sim\theta) \cdot P(\sim\theta)}$$

4

θ = Has disorder, x = Positive result

$$0.999 * 0.03$$

$$= 0.885$$

$$(0.999 * 0.03) * (0.004 * 0.97)$$

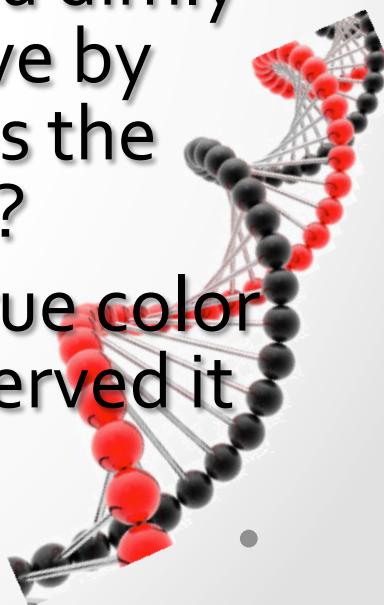
Intuition:

- Initial Prior Belief = 0.03 same as general population
- New data/observation = The test ;o(
- After test, increased to 0.885, given the likelihood 0.999 & factor 0.004 * 0.97





Problem #3

- When observing a **blue** vehicle driving by on a dimly-lit night, there is a 25% chance that its true color is **green**. The same can be said when observing a **green** vehicle under the same conditions: there is a 25% chance that its true color is **blue**. Suppose that 90% of taxis on the streets have **blue** color, while the other 10% have **green** color. On a dimly-lit night you see a taxi randomly drive by and observe its color as **blue**. What is the probability that its true color is **blue**?
 - Hint: condition the event that the true color is **blue** on the event that you've observed it to have **blue** color.
- 



Solution – Problem #3

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x|\theta) \cdot P(\theta) + P(x|\sim\theta) \cdot P(\sim\theta)}$$

4

θ = Color is Blue, x = Observation is blue

$$(0.75 * 0.9)$$

$$= 0.964$$

$$(0.75 * 0.9) + (0.25 * 0.1)$$



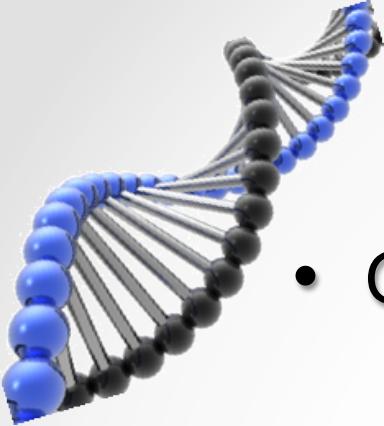


<http://www.google.com/doodles/nicolaus-copernicus-540th-birthday>

Classification

• • •

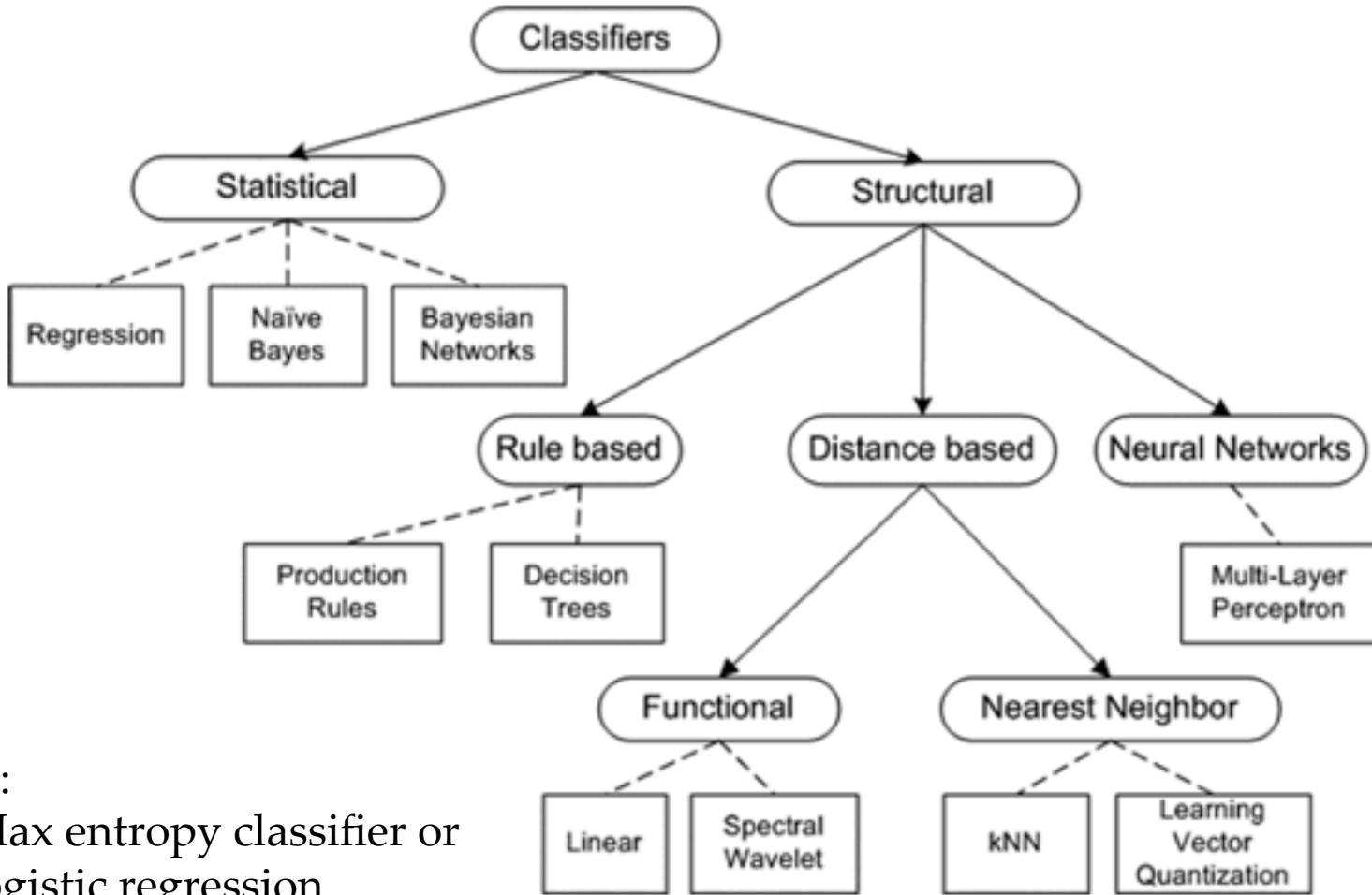




Classification

- Classifiers
 - Label Documents – Binary classification or multi-class classification
 - Mechanics
 - Learn from labeled feature sets
 - Then during classification extract feature set from an unknown document & map to known label-features
- Uses
 - Categorization – Topics, Languages, Type, News Feed...
 - Spam Filtering
 - Sentiment Analysis
 - Security
 - Topic Modeling – LDA 3 level Bayesian (Research and a few words)
 - Recommender Systems
 - Ad placement based on web page-ad
 - Authorship – The Federalist Papers





Add:

- Max entropy classifier or logistic regression
- SVM



Bayes Classifier

Definitions :

d = Document, \mathcal{D} = Document space

c = class, C = Class space

t = token, \mathcal{T} = Token space

$$d \in D \quad c \in \{c_1, c_2, \dots, c_n\} \quad t \in \{t_1, t_2, \dots, t_n\}$$



Problem:

Given a document d ,

Map it to a class in c



Bayes Classifier

Bayes Theorem says that

$$\hat{P}(c|d) \propto P(c) P(d/c)$$

6

Our problem reduces to finding $P(c)$ & $P(d/c)$

If we have a training set of N' documents, without any prior knowledge our estimate would be

$$P(c) = N_c/N'$$

7

where N_c = Number of document in class c





Bayes Classifier

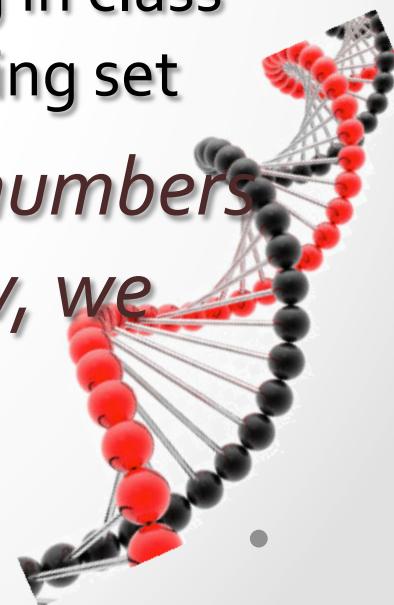
We can define a document in terms of a set of features, here the tokens it has; and then

$$\mathcal{P}(d/c) = \prod_{1 \rightarrow k} \mathcal{P}(t_k|c)$$

----- 8

- k = Number of tokens &
- $\mathcal{P}(t_k|c)$ = The probability of t_k occurring in class c , likelihood as inferred from the training set

Note : Since multiplication of small numbers can result in floating point underflow, we take log & add them





Bayes Classifier

We still are left with calculating the likelihood probability $P(t_k|c)$:

$$\hat{P}(t_k|c) = \frac{N_{tc}}{\sum N_{tc}}$$


Which is the relative frequency of t occurring in class c in the training set

In order to make this work for unseen tokens, ie if $N = 0$; we add the Laplace Smoothing as follows:

$$\hat{P}(t_k|c) = \frac{N_{tc} + 0.5}{\sum N_{tc} + (N_t/2)}$$



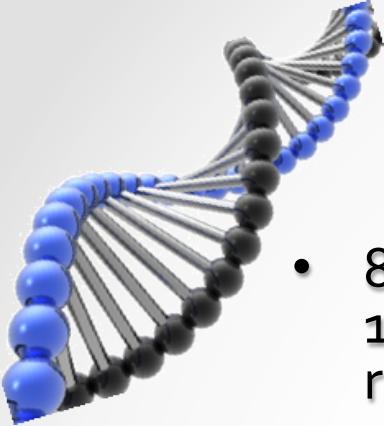



Bayes Classifier

This is exactly what nltk does!

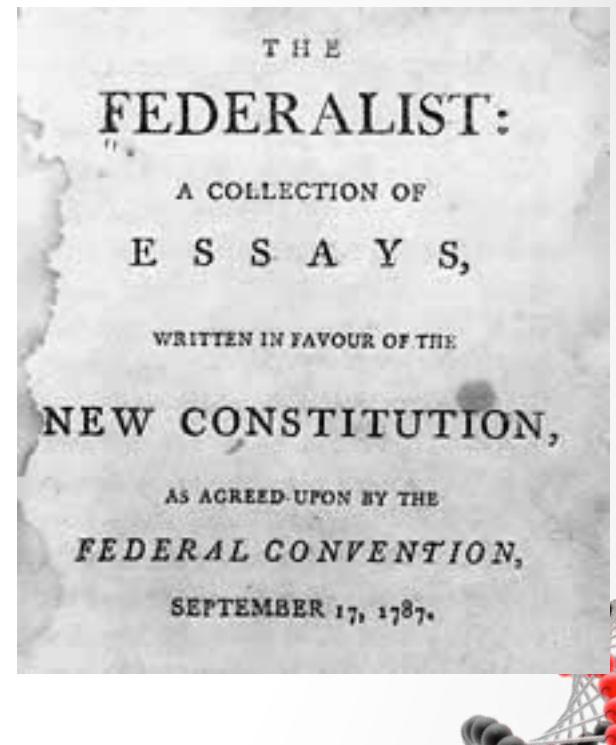
```
- class ELEProbDist(LidstoneProbDist):
    """
    The expected likelihood estimate for the probability distribution
    of the experiment used to generate a frequency distribution.  The
    X{expected likelihood estimate} approximates the probability of a
    sample with count M{c} from an experiment with M{N} outcomes and
    M{B} bins as M{(c+0.5)/(N+B/2)}.  This is equivalent to adding 0.5
    to the count for each bin, and taking the maximum likelihood
    estimate of the resulting frequency distribution.
    """
```





The Federalist Papers

- 85 Essays published anonymously 1787-88 to persuade New Yorkers to ratify the US constitution
- Authorship of 12 papers was not clear – between Hamilton and Madison
- In 1962, Mosteller & Wallace undertook a study to fingerprint the authorship based on word frequencies in previous writings
- Their work revived Bayesian Classification Methods



...the DNA helix continues across the bottom of the slide.

Education: Madison's Avenue

Sep 21, 1962

Oh Hamilton, Poor Hamilton, Madison Wrote 'Em and You're Feeling So Sad. That, in effect, was the title of the story, sketched out last week before a joint-meeting of the American Statistical Association, the Biometric Society and the Institute of Mathematical Statistics. Plotting the story were two smart mathematicians, Harvard's Frederick Mosteller, and the University ...



- <http://www.freerepublic.com/focus/f-news/1386274/posts>

...the DNA helix continues across the bottom of the slide.

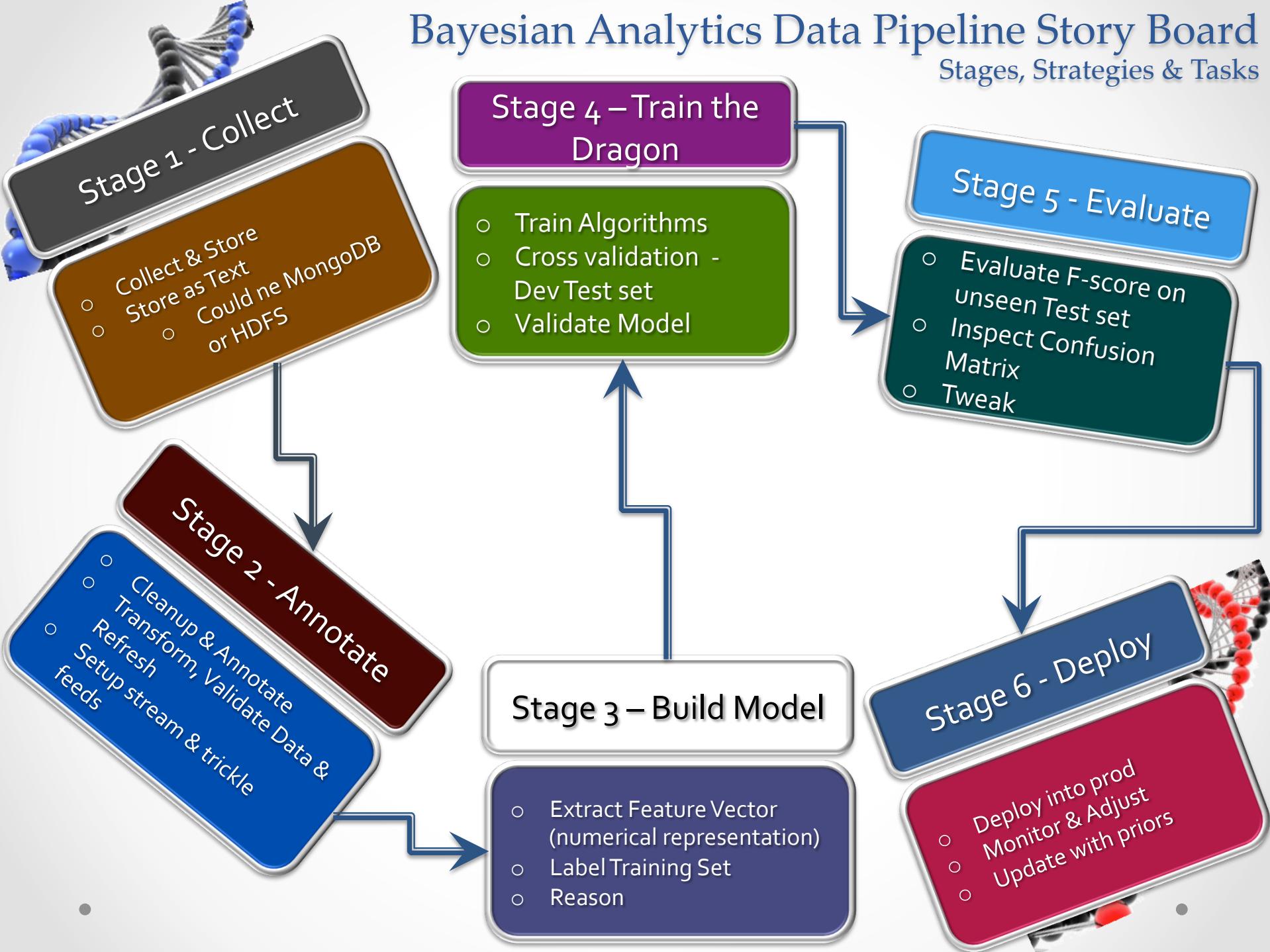


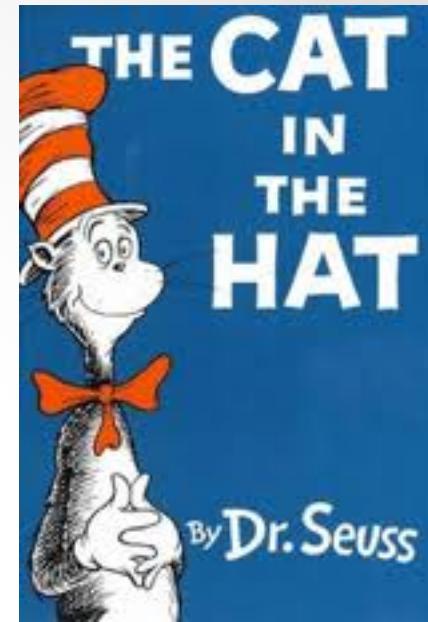
Feature Extraction

- **Bag Of Words**
 - Tokenize into unigrams (split et al)
 - WordGramAnalyzer()
 - Binary Occurrence (True/False)
 - Frequencies
 - TF-IDF
 - Bigram of words (Better : e.g. "New York", "Very Bad")
 - N-gram characters (for language classification)
 - CharNGramAnalyzer()
 - **Cleanup**
 - Lowercase
 - Take out accents
 - At least 2 characters
 - Remove stop words
 - Order doesn't matter
 - Number of occurrences do not matter (We will see later how we can improve)
- 

Bayesian Analytics Data Pipeline Story Board

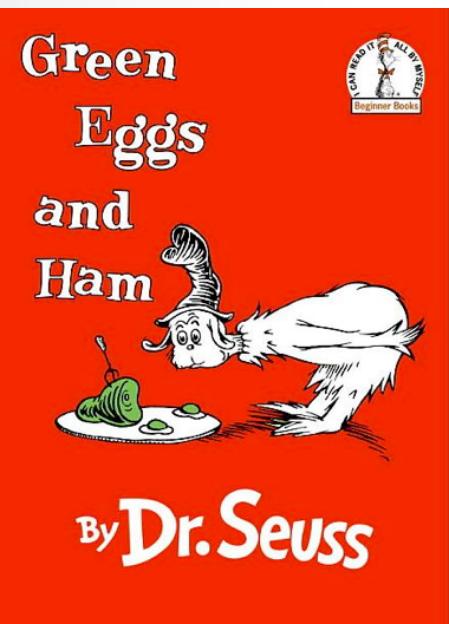
Stages, Strategies & Tasks





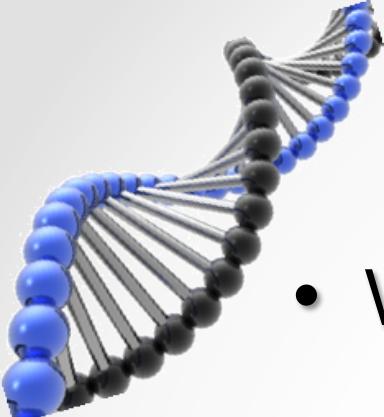
A Classification of the Seuss Kind

• • •



Ham I am
Thing₁ & Thing₂





Problem Statement

- We have two paragraphs
 - One from “Cat In The Hat” &
 - Another from “Green Eggs and Ham”
 - We train the classifiers with them
 - Then we test our classifier with other lines from the books
 - Gives us a starting point for Bayesian Classifier
 - Load “cat_in_the_hat-xx.py”
- 



Code Walk thru (1 of 2)

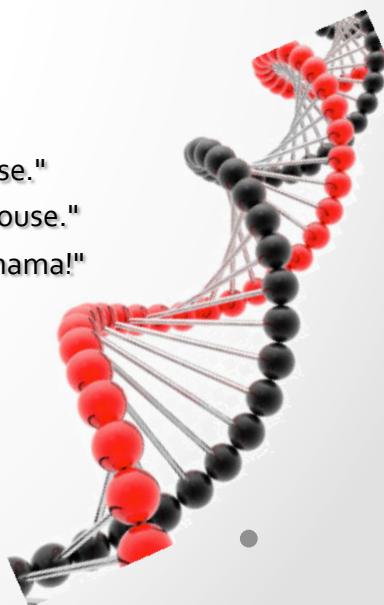
```
#!/usr/bin/env python
#
# pydata Tutorial March 18, 2013
# Thanks to StreamHacker a.k.a. Jacob Perkins
# Thanks to Prof.Todd Ebert
#
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk import tokenize
#
# Data
#
label_1 = "Cat In The Hat"

train_text_1 = "So we sat in the house all that cold, wet day. \
And we saw him! The Cat in the Hat! \
Your mother will not mind. \
He should not be here when your mother is out. \
With a cake on the top of my hat! \
I can hold the ship."

test_text_1 = "That cat is a bad one, \
That Cat in the Hat. \
He plays lots of bad tricks. \
Don't you let him come near."
```

```
label_2 = "Green Eggs and Ham"
train_text_2 = "I am Sam. \
Sam I am.\
I do not like green eggs and ham. \
I would not like them anywhere. \
Would you like them in a house? \
I will not eat them on a train. \
Thank you, thank you, Sam I am."

test_text_2 = "I would not eat them\
here or there.\
I would not eat them anywhere.\
I would not eat green eggs and ham."
#
# For testing classification
#
classify_cith = "We saw the cat in the house."
classify_geah = "And I will eat them in a house."
classify_other = "A man a plan a canal Panama!"
```



Code Walk thru (2 of 2)



```
#  
# Take a list of words and turn it into a Bag Of Words  
#  
def bag_of_words(words):  
    return dict([(word, True) for word in words])  
#  
# Program Starts Here  
#  
#  
# Step 1 : Feature Extraction  
#  
train_words_1 = tokenize.word_tokenize(train_text_1)  
train_words_2 = tokenize.word_tokenize(train_text_2)  
train_cith = [(bag_of_words(train_words_1), label_1)]  
train_geah = [(bag_of_words(train_words_2), label_2)]  
  
test_words_1 = tokenize.word_tokenize(test_text_1)  
test_words_2 = tokenize.word_tokenize(test_text_2)  
test_cith = [(bag_of_words(test_words_1), label_1)]  
test_geah = [(bag_of_words(test_words_2), label_2)]  
  
train_features = train_cith + train_geah  
test_features = test_cith + test_geah
```

```
#  
# Step 2: Train The Dragon er ... classifier ;o)  
#  
classifier = NaiveBayesClassifier.train(train_features)  
print 'Accuracy : %d' %  
nltk.classify.util.accuracy(classifier, test_features)  
#print classifier.most_informative_features()  
#print classifier.labels()  
#  
# Step 3 :Test Classification  
#  
print  
classifier.classify(bag_of_words(tokenize.word_tokenize  
(classify_cith)))  
print  
classifier.classify(bag_of_words(tokenize.word_tokenize  
(classify_geah)))  
print  
classifier.classify(bag_of_words(tokenize.word_tokenize  
(classify_other)))  
#  
# That's all Folks!  
#
```



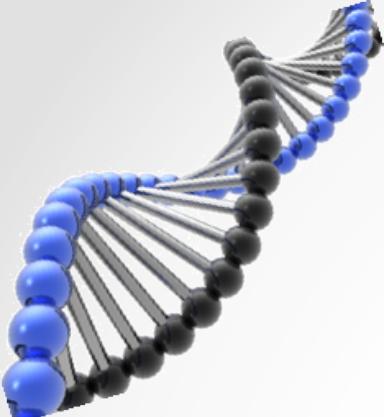
Run the program

Aftermath/Postmortem

...

Discussions ?
Questions ?



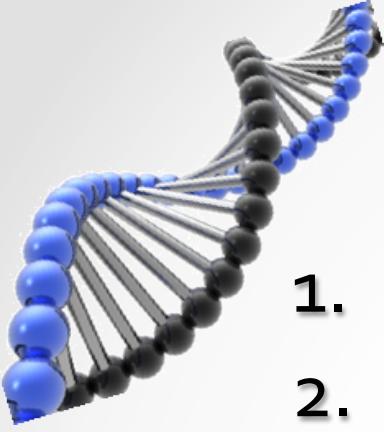


Spam Classification



Spam I am
I do not like them
Not in my Outlook
Nor in my Gmail
I will not take them in my Inbox
I like them trashed sooner they arrive





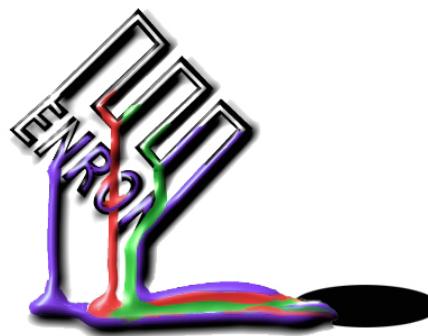
Problem Statement

1. Train the classifier with ham & spam
 2. Then classify unseen data set
 3. Evaluate the model
 - Programs:
 - spam-01.py
 - spam-02.py
 - Dataset
 - Enron
 - Following the work by Metsis,
Androulakos & Palioras
 - <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.5542>
- 



Background

- Spam & Ham data are hard to come by
- In 2005 the Enron investigations made public e-mails of 150 employees from Enron
- That gave impetus to research work
- The data we are using are the processed e-mails of 6 employees, mixing with 3 spam collections



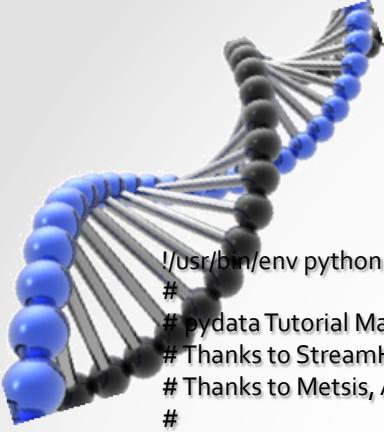


Data Prep

- Unzip the data
- Each user has ham & spam directory
- The program `spam-01.py` runs over these files.
- You should change the directory name [`root_path` variable] to where you downloaded the data
- ***Caution :***
 - *The program takes 10 minutes to run over the entire dataset.*
 - *I created a directory enron-small with 20 ham & spam e-mails to test during development*

▼	enron-pre-processed
	.DS_Store
▼	enron-small
	.DS_Store
▶	ham
▶	spam
▼	enron1
	.DS_Store
▶	ham
▶	spam
	Summary.txt
	enron1.tar.gz
▼	enron2
	.DS_Store
▶	ham
▶	spam
	Summary.txt
	enron2.tar.gz
▶	enron3
	enron3.tar.gz
▶	enron4
	enron4.tar.gz
▶	enron5
	enron5.tar.gz
▼	enron6
	.DS_Store
▶	ham
▶	spam
	Summary.txt
	enron6.tar.gz

Code walk-thru

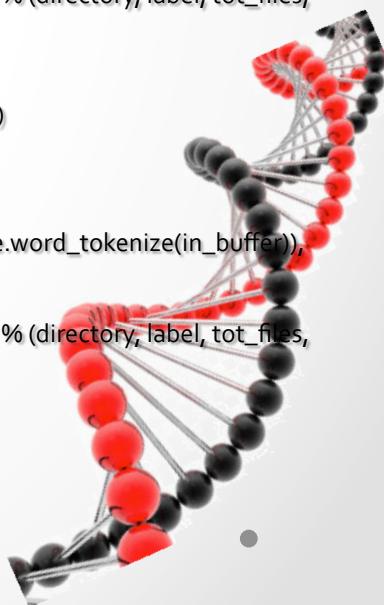


```
#!/usr/bin/env python
#
# pydata Tutorial March 18, 2013
# Thanks to StreamHacker a.k.a. Jacob Perkins
# Thanks to Metsis, Androutsopoulos & Palouras
#
import os
import sys
import time
#
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk import tokenize

#
# Take a list of words and turn it into a Bag Of Words
#
def bag_of_words(words):
    return dict([(word, True) for word in words])

start_time = time.clock()

#
# Data
#
label_1 = "ham"
label_2 = "spam"
ham_list=[]
spam_list=[]
#
root_path = '/Users/ksankar/Documents/Erlang/Bayesian/spam/enron-spam/enron-pre-processed'
directories = ['enron5']
#[‘enron1’,’enron2’,’enron3’,’enron4’,’enron5’,’enron6’]
labels=[‘ham’,’spam’]
```



```
#
# Need separate lists so that we can vary the proportions
#
ham_features=[]
spam_features=[]
tot_files=0
for directory in directories:
    label = labels[0] # ham
    dir_name = os.path.join(root_path,directory,label)
    #print dir_name
    for file in os.listdir(dir_name):
        file_name_full = os.path.join(dir_name,file)
        #print 'Reading file %s' %(file_name_full)
        #print '!',
        in_buffer = open(file_name_full,'r').read()
        ham_features.append((bag_of_words(tokenize.word_tokenize(in_buffer)),label)) # ham
        tot_files += 1
    print "%s %s Total Files = %d Feature Count = %d" %(directory, label, tot_files, len(ham_features))
    #
    label = labels[1] # spam
    dir_name = os.path.join(root_path,directory,label)
    for file in os.listdir(dir_name):
        file_name_full = os.path.join(dir_name,file)
        in_buffer = open(file_name_full,'r').read()
        spam_features.append((bag_of_words(tokenize.word_tokenize(in_buffer)),label)) # spam
        tot_files += 1
    print "%s %s Total Files = %d Feature Count = %d" %(directory, label, tot_files, len(spam_features))
read_time = time.clock()
print "Read Time : %1.3f" %(read_time-start_time)
```



```
#  
# Now ceate the training and test data sets  
#  
• ham_cutoff=len(ham_features) * 9/10 #3/4  
• spam_cutoff=len(spam_features) * 9/10 # 3/4  
#  
• print ham_features[0]  
• print spam_features[0]  
• print ham_features[1]  
• print spam_features[1]  
• trainfeats = ham_features[:ham_cutoff] + spam_features[:spam_cutoff]  
• testfeats = ham_features[ham_cutoff:] + spam_features[spam_cutoff:]  
• print 'train on %d instances, test on %d instances' % (len(trainfeats), len(testfeats))  
• classifier = NaiveBayesClassifier.train(trainfeats)  
• print classifier.labels()  
• print 'Accuracy:', nltk.classify.util.accuracy(classifier, testfeats)  
• classifier.show_most_informative_features()  
#  
• end_time = time.clock()  
• print "Run Time : %1.3f" % (end_time-start_time)  
• print "That's all Folks! All Good Things have to come to an end !"  
• #sys.exit()  
#  
• # That's all Folks!  
#
```



Hands-On

• • •

Run `spam-01.py`

Inspect variables

Try changing Training Set vs. Validation Set





```
enron5 ham Total Files = 1500 Feature Count = 1500
enron5 spam Total Files = 5175 Feature Count = 3675
Read Time : 78.967
, 'remove': True, '2002': True, 'i': True, 'write': True, '2020': True, 'have': True, 'wish': True, 'at': True,
'the': True, 'inc': True, 'or': True, 'Subject': True}, 'spam')
({'operations': True, 'help': True, 'global': True, 'already': True, 'causey': True, 'harrison': True, 'still':
True, 'staff': True, 'had': True, ',': True, 'send': True, 'should': True, 'to': True, 'going': True, 'rick':
True, 'mail': True, 'sent': True, '@': True, 'possible': True, 'early': True, 'know': True, 'shared': True, 'not':
True, 'regarding': True, 'notes': True, 'leave': True, 'glover': True, 'Subject': True, 'energy': True, 'some':
True, 'apologies': True, 'brent': True, 'subject': True, '01': True, '06': True, 'for': True, 'away': True, '/':
True, 're': True, 'new': True, 'before': True, 'shults': True, 'be': True, 'we': True, 'beck': True, 'ect': True,
'corp': True, 'wanted': True, 'by': True, 'enron': True, 'on': True, 'addtional': True, 'of': True, 'bob': True,
'create': True, 'first': True, 'sheila': True, 'via': True, 'solmonson': True ... 'my': True, 'and': True, 'an':
True, 'vp': True, 'as': True, 'at': True, 'in': True, 'any': True, '!': True, 'again': True, 'functions': True,
'end': True, 'provide': True, 'personnel': True, 'discussion': True, 'hou': True, 'role': True, 'responsibility':
True, 'you': True, 'ena': True, 'price': True, 'yesterday': True, 'validate': True, 'america': True, 'sally':
True, 'a': True, '04': True, 'susan': True, 'i': True, 'more': True, '2000': True, 'so': True, 'organization':
True, 'the': True, 'having': True}, 'ham')
({'through': True, 'edt': True, 'now': True, 'll': True, ',': True, 'to': True, 'usage': True, 'checking': True,
'unlimited': True, 'one': True, 'name': True, 'list': True, 'email': True, 'try': True, 'Subject': True,
'activate': True, 'rate': True, 'expect': True, 'our': True, 'click': True, 'what': True, 'for': True, 'please':
True, 'daytime': True, 'state': True, 'toll': True, '529': True, '?': True, 'be': True, 'we': True, 'who': True,
'friday': True, 'free': True, 'here': True, 'by': True, 'on': True, 'of': True, 'trial': True, 'connection': True,
'or': True, 'period': True ... 'is': True, 'received': True, 'am': True, 'it': True, 'have': True, 'if': True, '!':
True, 'information': True, '877': True, 'offer': True, '-': True, '1': True, 'valid': True, '25711': True, '9':
True, 'you': True, 'week': True, 'monday': True, 'after': True, 'never': True, 'a': True, '""': True, 'clear':
True, 'nighttime': True, 'the': True, 'mailing': True, 'spend': True}, 'spam')
```

train on 4657 instances, test on 518 instances

['spam', 'ham']

Accuracy: 0.994208494208

Most Informative Features

beck = True	ham : spam	=	485.6	:	1.0
ect = True	ham : spam	=	427.5	:	1.0
sally = True	ham : spam	=	359.2	:	1.0
713 = True	ham : spam	=	167.3	:	1.0
causey = True	ham : spam	=	154.3	:	1.0
sheila = True	ham : spam	=	136.3	:	1.0
cc = True	ham : spam	=	123.3	:	1.0
ted = True	ham : spam	=	120.0	:	1.0
mike = True	ham : spam	=	107.5	:	1.0
leslie = True	ham : spam	=	100.4	:	1.0

Run Time : 103.863

That's all Folks! All Good Things have to come to an end !





Summary

Dan Jurafsky



Summary: Naive Bayes is Not So Naive

- Very Fast, low storage requirements
- Robust to Irrelevant Features
 - Irrelevant Features cancel each other without affecting results
- Very good in domains with many equally important features
 - Decision Trees suffer from *fragmentation* in such cases – especially if little data
- Optimal if the independence assumptions hold: If assumed independence is correct, then it is the Bayes Optimal Classifier for problem
- A good dependable baseline for text classification
 - **But we will see other classifiers that give better accuracy**

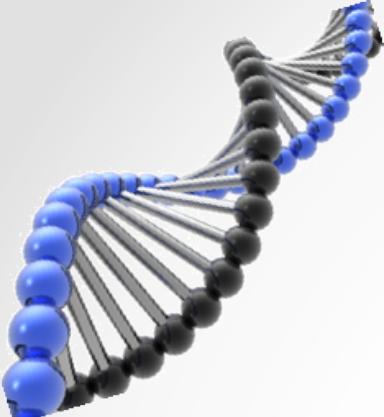


Model Evaluation - Accuracy

	Correct	Not Correct
Selected	True+ (tp)	False+ (fp)
Not Selected	False- (fn)	True- (tn)

- Accuracy =
$$\frac{tp + tn}{tp + fp + fn + tn}$$
- For cases where tn is large compared tp tp, a degenerate return(false) will be very accurate !
- Hence the F-measure is a better reflection of the model strength





Model Evaluation – Precision & Recall

	Correct	Not Correct
Selected	True +ve - tp	False +ve - fp
Not Selected	False -ve - fn	True -ve - tn

- Precision
- Accuracy
- Relevancy

$$\frac{tp}{tp+fp}$$

- Recall
- True +ve Rate
- Coverage
- Sensitivity

$$\frac{tp}{tp+fn}$$

- Precision = How many items we identified are relevant
- Recall = How many relevant items did we identify
- Inverse relationship – Tradeoff depends on situations
 - Legal – Coverage is important than correctness
 - Search – Accuracy is more important
 - Fraud
 - Support cost (high fp) vs. wrath of credit card co.(high fn)





Model Evaluation : F-Measure

	Correct	Not Correct
Selected	True+ (tp)	False+ (fp)
Not Selected	False- (fn)	True- (tn)

Precision = $tp / (tp+fp)$: Recall = $tp / (tp+fn)$

F-Measure = Balanced, Combined, Weighted
Harmonic Mean, measures effectiveness

$$\frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} = \frac{(\beta^2 + 1) PR}{\beta^2 P + R}$$

Common Form : $\beta=1$ $\alpha = 1/2$

$$F_1 = 2PR / P+R$$





Confusion Matrix

Actual	Predicted			
	C ₁	C ₂	C ₃	C ₄
C ₁	10	5	9	3
C ₂	4	20	3	7
C ₃	6	4	13	3
C ₄	2	1	4	15

Correct Ones (c_{ii})

$$\text{Precision} = \frac{C_{ii}}{\sum_i C_{ij}}$$

$$\text{Recall} = \frac{C_{ii}}{\sum_j C_{ij}}$$



Confusion matrix c

- For each pair of classes <c₁, c₂> how many documents from c₁ were incorrectly assigned to c₂?
- c_{3,2}: 90 wheat documents incorrectly assigned to poultry

Docs in test set	Assigned	Assigned poultry	Assigned wheat	Assigned coffee	Assigned interest	Assigned trade
True UK	95	1	13	0	1	0
True poultry	0	1	0	0	0	0
True wheat	10	90	0	1	0	0
True coffee	0	0	0	34	3	7
True interest	-	1	2	13	26	5
True trade	0	0	2	14	5	10

Video 6-8 Text Classification Evaluation Stanford NLP has very good discussion
<http://www.youtube.com/watch?v=OwwdYHWRB5E>





Hands On

• • •

Let us try Precision & Recall for our classifier

`setup-o2.py`

•





Code Walk-Thru

..... Classifier & above – same as before

```
#  
# Now create the data structure for model evaluation  
#  
refsets = collections.defaultdict(set)  
testsets = collections.defaultdict(set)  
for i, (feats, label) in enumerate(testfeats):  
    refsets[label].add(i)  
    observed = classifier.classify(feats)  
    testsets[observed].add(i)  
#print len(refsets)  
#print len(testsets)  
#print refsets  
precisions = {}  
recalls = {}  
for label in classifier.labels():  
    precisions[label] = metrics.precision(refsets[label], testsets[label])  
    recalls[label] = metrics.recall(refsets[label], testsets[label])  
#  
# Let us calculate Precision & Recall and compare with nltk  
#  
# Luckily the data structures are symmetric  
#  
c_00=len(refsets[labels[0]].intersection(testsets[labels[0]]))  
c_01=len(refsets[labels[0]].intersection(testsets[labels[1]]))  
c_10=len(refsets[labels[1]].intersection(testsets[labels[0]]))  
c_11=len(refsets[labels[1]].intersection(testsets[labels[1]]))
```



Code Walk-Thru

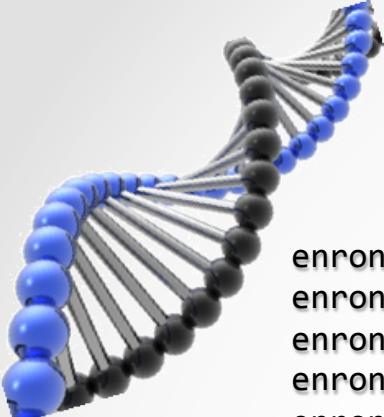
```
c_oo=len(refsets[labels[0]].intersection(testsets[labels[0]]))  
c_01=len(refsets[labels[0]].intersection(testsets[labels[1]]))  
c_10=len(refsets[labels[1]].intersection(testsets[labels[0]]))  
c_11=len(refsets[labels[1]].intersection(testsets[labels[1]]))  
  
#  
print '| H | S |'  
print '--|-----|-----|'  
print 'H | %5d | %5d |' % (c_oo,c_01)  
print '--|-----|-----|'  
print 'S | %5d | %5d |' % (c_10,c_11)  
print '--|-----|-----|'  
  
#  
ham_p = float(c_oo) / (c_oo + c_10)  
ham_r = float(c_oo) / (c_oo + c_01)   
spam_p = float(c_11) / (c_01 + c_11)  
spam_r = float(c_11) / (c_10 + c_11)  
  
#  
print 'P:{ham:%f,spam:%f}' % (ham_p,spam_p)  
print 'R:{ham:%f,spam:%f}' % (ham_r,spam_r)  
  
#  
print '(nltk)P:',precisions  
print '(nltk)R:',recalls
```

H	S
H	>
S	>

$$\text{Precision} = \frac{C_{ii}}{\sum_i C_{ij}}$$

$$\text{Recall} = \frac{C_{ii}}{\sum_j C_{ij}}$$





Run Capture & Discussion

```
enron1 ham Total Files = 3672 Feature Count = 3672
enron1 spam Total Files = 5172 Feature Count = 1500
enron2 ham Total Files = 9533 Feature Count = 8033
enron2 spam Total Files = 11029 Feature Count = 2996
enron3 ham Total Files = 15041 Feature Count = 12045
enron3 spam Total Files = 16541 Feature Count = 4496
enron4 ham Total Files = 18041 Feature Count = 13545
enron4 spam Total Files = 22541 Feature Count = 8996
enron5 ham Total Files = 24041 Feature Count = 15045
enron5 spam Total Files = 27716 Feature Count = 12671
enron6 ham Total Files = 29216 Feature Count = 16545
enron6 spam Total Files = 33716 Feature Count = 17171
```

Read Time : 574.077

Train on 30343 instances, Test on 3373 instances

	H	S
H	1630	25
S	38	1680

P:{ham:0.977218,spam:0.985337}

R:{ham:0.984894,spam:0.977881}

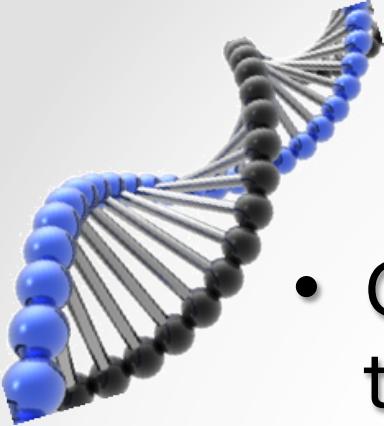
(nltk)P: {'ham': 0.9772182254196643, 'spam': 0.9853372434017595}

(nltk)R: {'ham': 0.9848942598187311, 'spam': 0.9778812572759022}

Run Time : 694.238

That's all Folks! All Good Things have to come to an end !



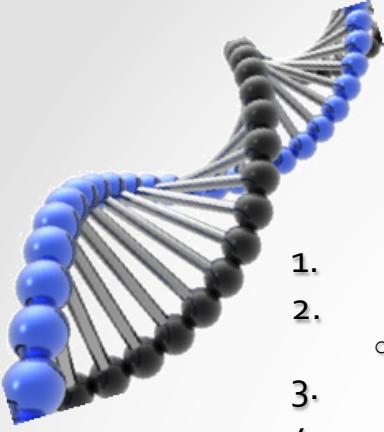


Future Vectors to Follow ...

- Calculate for spam with different sizes of training & testing datasets
- Delete words with low information gain
- Keep only high information gain words
- Stop words elimination (even though spam has lot more stop words)
- Sequence classifier
- Bigram & Trigram models, collocations

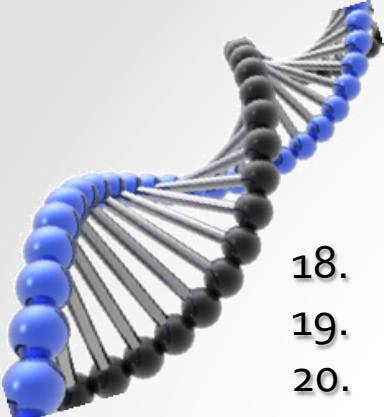
“Run multiple variants & measure relative lift” -- Paco Nathan





References

1. http://lesswrong.com/lw/774/a_history_of_bayes_theorem/
 2. [Prof. Allen Downey's pycon presentations – Must Read](#)
 - o <https://sites.google.com/site/simplebayes/home>
 3. <http://www.autonlab.org/tutorials/prob18.pdf>
 4. [Original Bayes Essay : http://rstl.royalsocietypublishing.org/content/53/370.short](#)
 - o http://en.wikipedia.org/wiki/An_Essay_towards_solving_a_Problem_in_the Doctrine_of_Chances
 5. <http://bayes.wustl.edu>
 6. <http://plato.stanford.edu/entries/bayes-theorem/>
 7. <http://yudkowsky.net/rational/bayes>
 8. <http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html>
 9. <http://oscarbonilla.com/2009/05/visualizing-bayes-theorem>
 10. <http://plato.stanford.edu/entries/bayes-theorem/supplement.html>
 11. <http://dovetheology.com/2012/01/25/derivation-of-bayes-theorem/>
 12. <http://quasar.as.utexas.edu/courses/stat295.2009/%203.%20Bayes%27%20Theorem.pdf>
 13. <http://quasar.as.utexas.edu/courses/stat295.2009/%203.%20Bayes%27%20Theorem.pdf>
 14. <http://www.nytimes.com/2011/08/07/books/review/the-theory-that-would-not-die-by-sharon-bertsch-mcgrayne-book-review.html>
 15. <http://quoteinvestigator.com/2011/07/22/keynes-change-mind/>
 16. <http://www.amstat.org/about/statisticiansinhistory/index.cfm?fuseaction=biosinfo&BioID=10>
 17. Images
 - 1. <http://410bridge.org/>
 - 2. http://en.wikipedia.org/wiki/The_Cat_in_the_Hat
- 

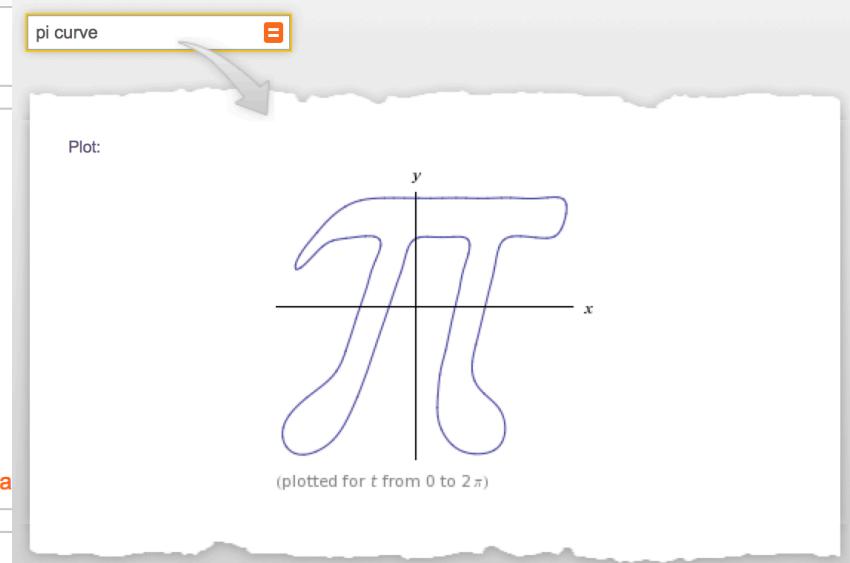
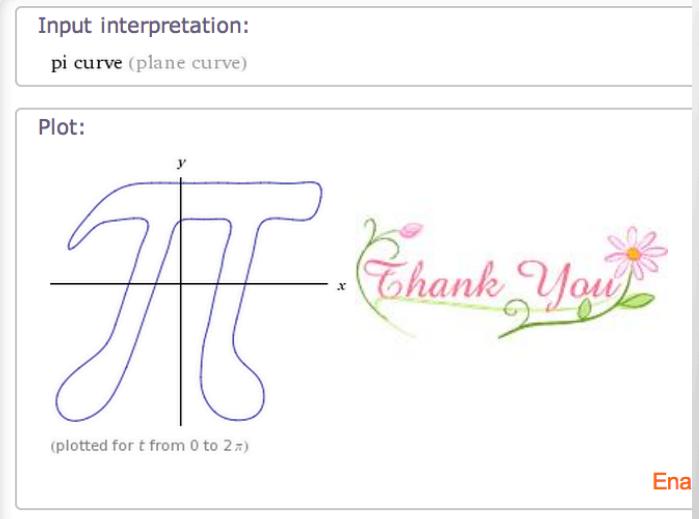


References

18. <https://www.math.umass.edu/~lavine/whatisbayes.pdf>
 19. <http://paulgraham.com/antispam.html>
 20. <http://www.cognitionis.com/nlp/>
 21. <http://computersciencesource.wordpress.com/2010/01/07/year-2-machine-learning-confusion-matrix/>
 22. <http://www.cecs.csulb.edu/~ebert/teaching/lectures/551/bayes/bayes.pdf>
 23. <http://dbacl.sourceforge.net/tutorial.html>
 - o <http://dbacl.sourceforge.net/dbacl.ps>
 24. <http://www.cs.ubc.ca/~murphyk/Teaching/CS340-Fallo6/reading/NB.pdf>
 25. <http://www.jstor.org/discover/10.2307/1403452?uid=3739560&uid=2&uid=4&uid=3739256&sid=21101854646871>
 26. <http://nlp.stanford.edu/IR-book/pdf/13bayes.pdf>
 27. <http://www.quora.com/How-does-multinomial-Naive-Bayes-work>
 28. <http://www.cs.cmu.edu/~knigam/papers/multinomial-aaaiws98.pdf>
 29. <http://hernan.amiune.com/teaching/from-a-bayes-theorem-naive-example-to-a-naive-bayes-example.html>
- 



Gracias



Equations:

Parametric equations:

$$x(t) = \frac{17}{31} \sin\left(\frac{235}{57} - 32t\right) + \frac{19}{17} \sin\left(\frac{192}{55} - 30t\right) + \frac{47}{32} \sin\left(\frac{69}{25} - 29t\right) + \frac{35}{26} \sin\left(\frac{75}{34} - 27t\right) + \frac{6}{31} \sin\left(\frac{23}{10} - 26t\right) + \frac{35}{43} \sin\left(\frac{10}{33} - 25t\right) + \frac{126}{43} \sin\left(\frac{421}{158} - 24t\right) + \frac{143}{57} \sin\left(\frac{35}{22} - 22t\right) + \frac{106}{27} \sin\left(\frac{84}{29} - 21t\right) + \frac{88}{25} \sin\left(\frac{23}{27} - 20t\right) + \frac{74}{27} \sin\left(\frac{53}{22} - 19t\right) + \frac{44}{53} \sin\left(\frac{117}{25} - 18t\right) + \frac{126}{25} \sin\left(\frac{88}{49} - 17t\right) + \frac{79}{11} \sin\left(\frac{43}{26} - 16t\right) + \frac{43}{12} \sin\left(\frac{41}{17} - 15t\right) + \frac{47}{27} \sin\left(\frac{244}{81} - 14t\right) + \frac{8}{5} \sin\left(\frac{79}{19} - 13t\right) + \frac{373}{46} \sin\left(\frac{109}{38} - 12t\right) + \frac{1200}{31} \sin\left(\frac{133}{74} - 11t\right) + \frac{67}{24} \sin\left(\frac{157}{61} - 10t\right) + \frac{583}{28} \sin\left(\frac{13}{8} - 8t\right) + \frac{772}{35} \sin\left(\frac{59}{16} - 7t\right) + \frac{3705}{46} \sin\left(\frac{117}{50} - 6t\right) + \frac{862}{13} \sin\left(\frac{19}{8} - 5t\right) + \frac{6555}{34} \sin\left(\frac{157}{78} - 3t\right) + \frac{6949}{13} \sin\left(\frac{83}{27} - t\right) - \frac{6805}{54} \sin\left(2t + \frac{1}{145}\right) - \frac{5207}{37} \sin\left(4t + \frac{49}{74}\right) - \frac{1811}{58} \sin\left(9t + \frac{55}{43}\right) - \frac{63}{20} \sin\left(23t + \frac{2}{23}\right) - \frac{266}{177} \sin\left(28t + \frac{13}{18}\right) - \frac{2}{21} \sin\left(31t + \frac{7}{16}\right)$$

$$y(t) = \frac{70}{37} \sin\left(\frac{65}{32} - 32t\right) + \frac{11}{12} \sin\left(\frac{98}{41} - 31t\right) + \frac{26}{29} \sin\left(\frac{35}{12} - 30t\right) + \frac{54}{41} \sin\left(\frac{18}{7} - 29t\right) + \frac{177}{71} \sin\left(\frac{51}{19} - 27t\right) + \frac{59}{34} \sin\left(\frac{125}{33} - 26t\right) + \frac{49}{29} \sin\left(\frac{18}{11} - 25t\right) + \frac{151}{75} \sin\left(\frac{59}{22} - 24t\right) + \frac{52}{9} \sin\left(\frac{118}{45} - 22t\right) + \frac{52}{33} \sin\left(\frac{133}{52} - 21t\right) + \frac{37}{45} \sin\left(\frac{61}{14} - 20t\right) + \frac{143}{46} \sin\left(\frac{144}{41} - 19t\right) + \frac{254}{47} \sin\left(\frac{19}{52} - 18t\right) + \frac{246}{35} \sin\left(\frac{92}{25} - 17t\right) + \frac{722}{111} \sin\left(\frac{176}{67} - 16t\right) + \frac{136}{23} \sin\left(\frac{3}{19} - 15t\right) + \frac{273}{25} \sin\left(\frac{32}{21} - 13t\right) + \frac{229}{33} \sin\left(\frac{117}{28} - 12t\right) + \frac{19}{4} \sin\left(\frac{43}{11} - 11t\right) + \frac{135}{8} \sin\left(\frac{23}{10} - 10t\right) + \frac{205}{6} \sin\left(\frac{33}{23} - 8t\right) + \frac{679}{45} \sin\left(\frac{55}{12} - 7t\right) + \frac{101}{8} \sin\left(\frac{11}{12} - 6t\right) + \frac{2760}{59} \sin\left(\frac{40}{11} - 5t\right) + \frac{1207}{18} \sin\left(\frac{21}{23} - 4t\right) + \frac{8566}{27} \sin\left(\frac{39}{28} - 3t\right) + \frac{12334}{29} \sin\left(\frac{47}{37} - 2t\right) + \frac{15410}{39} \sin\left(\frac{185}{41} - t\right) - \frac{596}{17} \sin\left(9t + \frac{3}{26}\right) - \frac{247}{28} \sin\left(14t + \frac{25}{21}\right) - \frac{458}{131} \sin\left(23t + \frac{21}{37}\right) - \frac{41}{36} \sin\left(28t + \frac{7}{8}\right)$$

Was it a vision, or a
waking dream?
Fled is that music:-do
I wake or sleep?

-Keats, Ode to a
Nightingale

