Tools & Models for Data Science

Intro to Supervised Learning

Chris Jermaine & Risa Myers

Rice University

Machine Learning (Review)

- Predicting / classifying new data based on what we have learned from existing data
- Requires the training data to be representative of the data we want to predict / classify
- Tasks
  - 1. Choosing the model—choose family, complexity, hyperparameters
  - 2. Learning the model—“fit” model to data by adjusting parameters
  - 3. Validating the model—make sure model matches data
  - 4. Applying the model—use the model to explain past/present make predictions on future

"Supervised" Learning

- One of the most fundamental problems in data science
  - Given a bunch of (x_i, y_i) pairs
  - Where x_i is a feature vector for data i
  - And y_i is the label for x_i
  - Goal: learn how to predict value of y from x
  - Called "supervised" because have examples of correct labeling

Problem Examples

- From research done at Rice:
  - Given a text clinical note, label “breast cancer” or not
  - Given a document (email) in a court case, figure which subjects pertain
  - Given information about a patient surgery, predict death
  - Given head trauma patient info, predict intracranial pressure crisis
  - Given an set of surgical vital signs, label “good surgery” or not
  - Predict location and damage from a hurricane
  - Many others!

Two Most Common Examples of Supervised Learning

- Classification and regression
- Classification:
  - Outcome to predict is in $\{+1, -1\}$ (“yes” or “no”), (“True” or “False”)
  - Ex: Given a text clinical note, label it as “breast cancer” or not
- Regression:
  - Outcome to predict is a real number
  - Ex: Given an ad, predict number of clickthrus per hour

What Models Are Used?

- Many!
  - We will cover a number of them
  - Simplest, most common: linear regression. From x_i, predict y_i as:

$$\sum_j x_{i,j} r_j$$

  - Where $x_{i,j}$ is a matrix of rows of feature values, one for each data point
  - $\langle r_1, r_2, \ldots, r_m \rangle$ are called regression coefficients
    - The relative magnitude of the regression coefficient tells you how important each feature is
  - Other common ones: kNN, support vector machines

What do we Mean by Features?

- Music
  - Artist name
  - Song duration
  - Key
  - Loudness
  - Tempo
  - Year
  - . . .

songId	artist name	duration	key	loudness	tempo	year
SONHOTT12A8C13493C	Adam Ant	233	0	-9.013	119.293	1982
SOGNQWU12A8AE4868F	Eurythmics	207	0	-6.629	84.164	1985
SOZWHVZ12A6D4F90E7	U2	485	3	-7.614	134.083	2007
SOGOQGE12AB0182907	The Killers	284	4	-6.546	151.953	2008
SONCHNB12AB01849A2	Wade Ray	136	3	-9.713	171.527	2005

Labels

- Some outcome / value / output pertaining to the data

Familiarity	Familiarity $\geq$ 0.8	songId	artist name	duration	key	loudness	tempo	year
0.63	FALSE	SONHOTT12A8C13493C	Adam Ant	233	0	-9.013	119.293	1982
0.75	FALSE	SOGNQWU12A8AE4868F	Eurythmics	207	0	-6.629	84.164	1985
0.84	TRUE	SOZWHVZ12A6D4F90E7	U2	485	3	-7.614	134.083	2007
0.92	TRUE	SOGOQGE12AB0182907	The Killers	284	4	-6.546	151.953	2008
0.05	FALSE	SONCHNB12AB01849A2	Wade Ray	136	3	-9.713	171.527	2005

A Model to Predict Music Familiarity: Regression Coeffiecients

$$\sum_j x_{i,j} r_j$$

1. Learn the regression coefficients
2. Use the model
   1. Plug in the value of each feature for each song
   2. Get a real valued output
   3. Apply a function to convert it to a binary classification

Feature	Regression Coefficient
artist.hotttnesss	-25.5347
artist.mbtags_count	-0.18
bars_confidence	-0.3755
bars_start	0.0893
beats_confidence	0.407
beats_start	0.1196
duration	0.0047
end_of_fade_in	0.0261
key	-0.0076
key_confidence	0.3139
latitude	0.0022
longitude	0.0019
loudness	-0.1131
mode	-0.0399
mode_confidence	0.3354
release.id	0
song_hotttnesss	-1.6807
start_of_fade_out	-0.0043
tatums_confidence	0.7883
tatums_start	-0.2762
tempo	0.0015
terms_freq	0.0006
time_signature	-0.0579
time_signature_confidence	-0.203
year	0.0001
Intercept	13.7194

Measuring Classification Accuracy: TP, FP, FN, TN

- True Positive (TP) – model predicts TRUE when label is TRUE
- False Positive (FP) – model predicts TRUE when label is FALSE
- True Negative (TN) – model predicts FALSE when label is FALSE
- False Negative (FN) – model predicts FALSE when label is TRUE

Measuring Classification Accuracy: Confusion Matrix

- Tabular representation of results

		Predicted		
		Negative	Positive	
Actual	Negative	TN	FP	Total Negative
	Positive	FN	TP	Total Positive
Total		Total Predicted Negative	Total Predicted Positive	

Measuring Classification Accuracy: Confusion Matrix

- Tabular representation of results

		Predicted		
		Negative	Positive	
Actual	Negative	8954	178	9132
	Positive	335	533	868
Total		9289	711	10000

Measuring Classification Accuracy: % Correct

- Simplest: % correct
- $\frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} = \frac{533+8954}{868+9132} = 0.95$

? Pros and cons?

		Predicted		
		Negative	Positive	
Actual	Negative	8954	178	9132
	Positive	335	533	868
Total		9289	711	10000

Measuring Classification Accuracy: % Correct

- Simplest: % correct
  - Pros
    - Easy to interpret
    - Easy to compute
    - Easy to compare
  - Cons
    - Certain classes may be more important than others
    - E.g. Male breast cancer
    - Can achieve high accuracy just by saying “No”
    - In this case, better to find all the cases and some that aren’t

Measuring Classification Accuracy: FP & FN

		Predicted Negative	Predicted Positive	
Actual	Negative	TN	FP	Total Negative
	Positive	FN	TP	Total Positive
Total		Total Predicted Negative	Total Predicted Positive	

		Predicted Negative	Predicted Positive	
Actual	Negative	8954	178	9132
	Positive	335	533	868
Total		9289	711	10000

- False positive: % of those we say are "yes" that are not really "yes"

$$\frac{FP}{N} = \frac{FP}{FP+TN} = \frac{178}{8954+178} = 0.19$$

- False negative: % of those we say are "no" that are not really "no"

$$\frac{FN}{P} = \frac{FN}{FN+TP} = \frac{335}{335+533} = 0.39$$

- Simple concept, easy to get wrong
- Male breast cancer: 50% FP rate is okay: 3 real cases + 3 extra cases
- Alert fatigue: 72 - 99% of alerts are not actual causes for alert [1]

[1] Sendelbach S, Funk M. Alarm fatigue: a patient safety concern. AACN advanced critical care. 2013;24(4):378-86.

Measuring Classification Accuracy: Recall & Precision

- Recall (Sensitivity): % of those that are really "yes" that we say are "yes"

$$\frac{TP}{P} = \frac{TP}{TP+FN} = \frac{533}{533+335} = 0.61$$

- Precision: % of those that we say are "yes" that are really "yes"

$$\frac{TP}{TP+FP} = \frac{533}{533+178} = 0.75$$

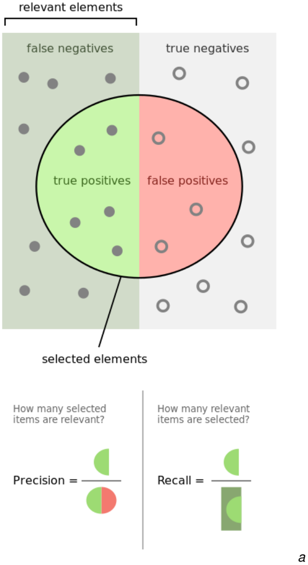


Image credit: `https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg`

Measuring Classification Accuracy: Recall & Precision

		Predicted		
		Negative	Positive	
Actual	Negative	0	9132	9132
	Positive	0	868	868
Total		0	868	10000

TN	FP
FN	TP

? What is our Recall if we call all of the data points Positive?

$$\frac{TP}{TP+FN} =$$

Measuring Classification Accuracy: Recall & Precision

		Predicted		
		Negative	Positive	
Actual	Negative	0	9132	9132
	Positive	0	868	868
Total		0	868	10000

TN	FP
FN	TP

- What is our Recall if we call all of the data points Positive?

$$\frac{TP}{TP+FN} = \frac{868}{868+0} = 1$$

? What is our Precision if we call all of the data points Positive?

$$\frac{TP}{TP+FP} =$$

Measuring Classification Accuracy: Recall & Precision

		Predicted		
		Negative	Positive	
Actual	Negative	0	9132	9132
	Positive	0	868	868
Total		0	868	10000

TN	FP
FN	TP

- What is our Recall if we call all of the data points Positive?

$$\frac{TP}{TP+FN} = \frac{868}{868+0} = 1$$

- What is our Precision if we call all of the data points Positive?

$$\frac{TP}{TP+FP} = \frac{868}{868+9132} = 0.09$$

Measuring Classification Accuracy: Recall & Precision

- Recall and precision
  - Pros
    - More information than accuracy
    - Tolerance for TP and FN can be different, based on the situation
  - Cons
    - Can be confusing
    - There are two numbers (Precision and Recall) to compare

Measuring Classification Accuracy $F1$-score

- F_1
  - Puts recall and precision into single number

$$F_1 = \frac{2 \times \text{ precision } \times \text{ recall}}{\text{precision } + \text{ recall}}$$

  - Ranges from 0 to 1
  - 1 is best
  - Also called the harmonic average of precision and recall

Digression: Mean Blood Pressure

- Blood Pressure has two components: Systolic and Diastolic
- Systolic – 1st number; pressure when heart is beating
- Diastolic – 2nd number; pressure when heart is resting
- Mean BP is NOT

$$\frac{\text{Systolic} + \text{Diastolic}}{2}$$

- Instead it is

$$\frac{\text{Systolic} + 2 * \text{Diastolic}}{3}$$

Measuring Classification Accuracy: F-scores

- F_1

$$F_1 = \frac{2 \times \text{ precision } \times \text{ recall}}{\text{precision } + \text{ recall}}$$

- Pros
  - Reasonable way to combine precision and recall
- Cons
  - Somewhat arbitrary
  - Could use F_2, F_3, etc.

$$F_\beta = (1+\beta^2) \times \frac{\text{precision } \times \text{ recall}}{(\beta^2 \times \text{ precision }) + \text{ recall}}$$

Measuring Classification Accuracy: F-scores on Our Model

- $F_1 = 0.68$
- $F_2 = 0.64$
- $F_3 = 0.63$

AUC ROC

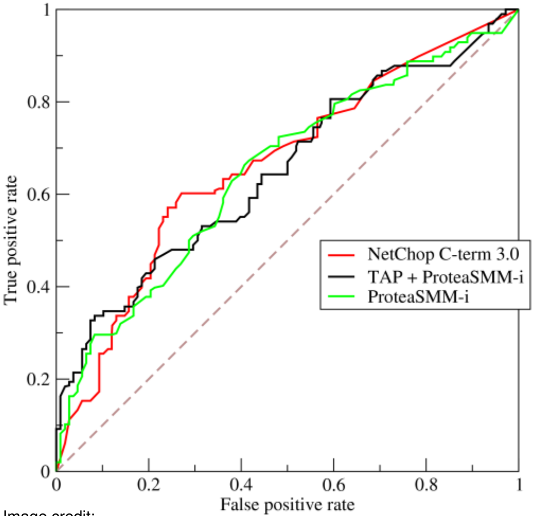


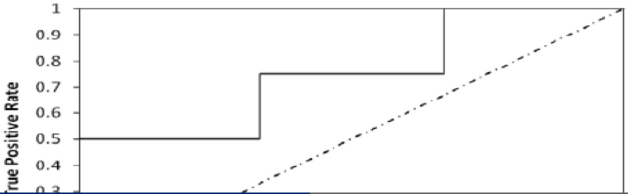
Image credit:

`https://commons.wikimedia.org/wiki/File:Roccurves.png`

- ROC = “Receiver operating characteristic”
- AUC = “Area under curve”
- Measure of how well the classes are separated
- Use for “tunable” classifiers (with cut-offs, like Logistic Regression)
- Gives single number ≤ 1.0
- Less than 0.5 means “actively bad”
- ? Often shown with the diagonal. Why?
- ? What does it mean if you have an AUC < 0.5?

Drawing an ROC curve

Rank		1	2	3	4	5	6	7	8	9	10
Actual class		+	+	–	–	+	–	–	+	–	–
TP	0	1	2	2	2	3	3	3	4	4	4
FP	0	0	0	1	2	2	3	4	4	5	6
TN	6	6	6	5	4	4	3	2	2	1	0
FN	4	3	2	2	2	1	1	1	0	0	0
TPR	0	0.25	0.5	0.5	0.5	0.75	0.75	0.75	1	1	1
FPR	0	0	0	0.17	0.33	0.33	0.50	0.67	0.67	0.83	1



AUC ROC: Pros and Cons

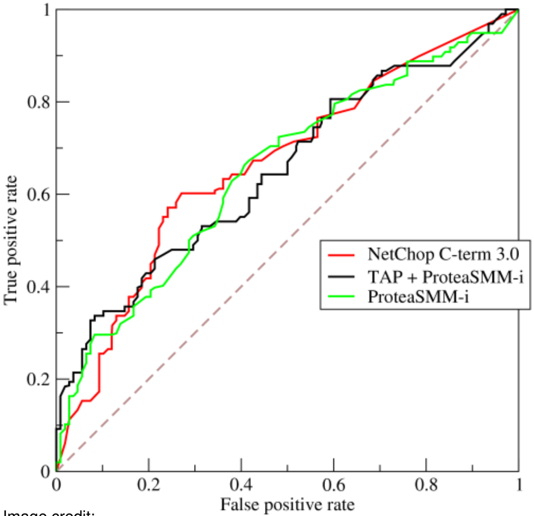


Image credit:

https://commons.wikimedia.org/wiki/File:Roccurves.png

- Pros
  - Single number
  - Well known
  - Immune to classification threshold
- Cons
  - Only works on binary classification

AUC ROC: Intuition

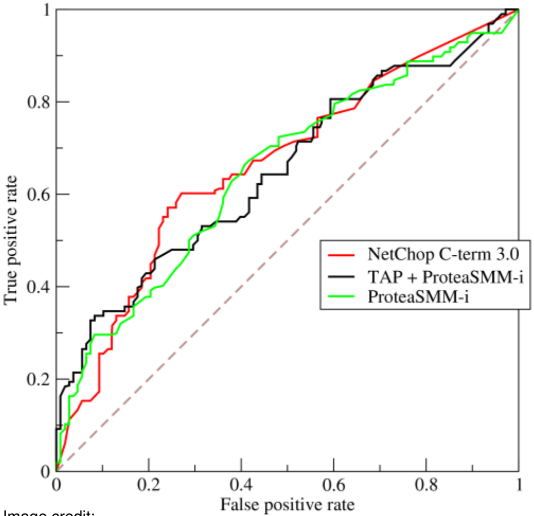
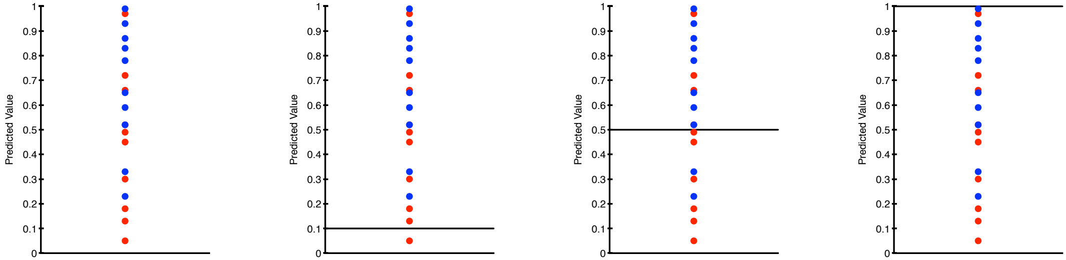


Image credit:

https://commons.wikimedia.org/wiki/File:Roccurves.png

- Requires us to have a numeric value for the results that are mapped to a binary value
- Depending on where we set the threshold, we get different results

AUC ROC



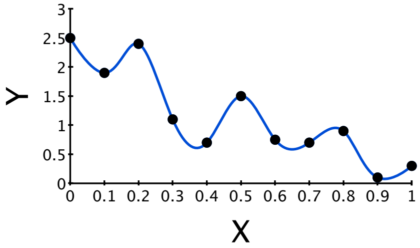
- Red = Negative cases
- Blue = Positive cases
- Start at the bottom (or top)
- Sweep up (down)
- Compute the TPR and FPR at each step
- Plot on ROC
- Compute AUC

Supervised Learning Methodology / Experimental Design

- Important to divide available data into
  - Training–used to learn model
  - Validation–used to see if model useful
  - Repeat these two, experimenting
  - Testing–used to evaluate useful models
- Don't touch testing until ready to eval
  - Evaluation on testing must be very last step!
  - ? Why?

Why Test Only Once?

- Overfitting
  - It's easy to build a model that exactly fits your data
  - But doesn't work on new data
  - Maybe the world has changed
    - Can't do much about this
  - Overfitting
    - You don't want to engineer something that only works on your test data



How To Perform Testing

- One-off
  - Divide data into Training, Validation, Testing
  - Apply validated model(s), get results
  - ? What if you don't have a lot of data?

How To Perform Testing

- One-off
  - Divide data into Training, Validation, Testing
  - Apply validated model(s), get results
  - What if you don’t have a lot of data?
    - You need to “reuse” you data

k-Fold Cross-Validation

- Break data into k random subsets ("folds")

```
For i = 1 to k do:
  Train on all folds except i;
  Evaluate learned model on fold i;
Report average results;
```

- Benefits
  - Works for small test sets
  - k can be large
  - 10 is commonly used
  - Needed when the data set is not very large
  - and / or when there are many parameters

Other techniques for Generating Test Sets

- Bootstrap – random sampling with replacement
- Jackknife resampling (Leave out one)

Bootstrap

- Random sampling with replacement
  1. Pick a sample size, n
  2. Choose n samples from the dataset, WITH replacement
  3. Train on the selected samples
  4. Test on the non-selected samples in the dataset
  5. Average the results
- Norms
  - n is typically chosen to be the size of the original dataset
  - Repeat the process MANY times (at least 20 or 30, usually hundreds of times)
  - Can also be used to estimate statistics such as a population mean

Bootstrap: Pros and Cons

- Pros
  - Works well with small datasets
  - Can produce confidence intervals for this approach
  - Non-parametric
- Cons
  - Sample sets are not all the same size
  - Assumes data are independent
  - Not repeatable (samples are different each time)

Jackknife

- Also know as Leave-one-out
- Sampling WITHOUT replacement
- Older method
  1. Pick a one sample
  2. Train on the remaining samples
  3. Test on the selected sample in the dataset
  4. Repeat for each sample
  5. Average the results
- Norms
  - Can also be used to estimate statistics such as a population mean

Jackknife: Pros and Cons

- Pros
  - Works well with small datasets
  - Can produce confidence intervals for this approach
  - Non-parametric
  - Uses less computing resources
- Cons
  - Repeatable
  - Assumes data are independent

How to Split Data?

- Random
- By time
- Representative of the data (think about Rice's College system)

Classification or Regression?

- Regression
- “Real” value (may be integral)
- Classification
- 1 of a limited number of classes

? How can we convert a regression problem to a classification problem?

- e.g. Familiarity of a song

? How can we make restrict a classification problem to a binary classification problem?

- E.g. How can we predict if someone is going to get an A, B, or C in this class into a binary classification problem?

Rare Class Problems

- Problem
  - Not enough examples to learn how to classify the rare class
- Approaches
  - Oversample the rare class (SMOTE)
  - Undersample the common class
  - Create new rare cases based on the minority class

Data Preparation

- Compute values using the Training set ONLY
- Apply the EXACT same transformation to the VALIDATION and TEST sets

Types of Data Preparation / Feature Engineering

- Mean normalization - mean 0

$$x' = \frac{x - \text{average}(x)}{\max(x) - \min(x)}$$

- Standardization - mean 0, std dev = 1

$$x' = \frac{x - \bar{x}}{\sigma}$$

- Scaling - min -1, max 1

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- Scale feature vector to unit length across the features

$$x' = \frac{x}{||x||}$$

Measuring Regression Accuracy

- View the list of prediction errors as a vector
- Can have many loss functions, corresponding to norms
- Given a vector of errors $\langle \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \rangle$, l_p norm defined as:

$$\left(\sum_{i=1}^{n} |\varepsilon_i|^p \right)^{1/p}$$

- Common loss functions correspond to various norms:
  - l_1 corresponds to mean absolute error
  - l_2 to mean squared error/least squares
    - Most commonly used
    - Convex
    - Easy to compute
  - l_∞ corresponds to minimax

Feature Selection

- Lots of focus in supervised learning on models
  - Linear regression, SVM, kNN, etc.
- Almost always **less** important than feature engineering
  - That is, most simple models accept $x_i = \langle x_{i,1}, x_{i,2}, \ldots, x_{i,m} \rangle$
  - Do not accept your raw data!
  - How you “vectorize” is often the most important question!
- Let’s consider feature engineering thru an example...

Web Page Link Feature Selection

- Web page link
  - Location
  - Font
  - Size
  - Color
  - ...
- vs. Deep learning
  - Use the raw data
  - E.g. Take a screen shot of the web page
  - Skip the feature engineering phase

Feature Selection Email

I NEED YOUR ASSISTANCE PLEASE Spam ×

Mr Khim Leang khl100@foxmail.com via [redacted]
to [redacted]

Mon, Oct 15, 5:02 PM (1 day ago)

Why is this message in spam? It is similar to messages that were identified as spam in the past.

Report not spam

Dear Friend ,

I am Mr Khim Leang and a personal Accountant/Executive board of Directors with Foreign Trade Bank of Cambodia (FTB).
it is with good spirit of heart i opened up this great opportunity to you A deceased client of mine that shares almost the same name as yours died as a result of heart-related condition on march 2005.His heart condition was duo to the death of the members of his family in the tsunami disaster on the 26 December 2004 in Sumatra Indonesia where they all lost their lives..{More info: http://en.wikipedia.org/wiki/2004_Indian_Ocean_earthquake_and_tsunami}

There is a draft account opened in my bank in 1999 by a long-time client our bank,a national of your country.he was a CEO/a textile company owner,business man,a miner at kruger mining company here in Cambodia. he was a geologist and consultant to several other mining conglomerates operating in Cambodia,China,Taiwan,Japan,Indonesia,Pakistan,Vietnam all in Asia,before he passed away on 12th march 2005 leaving nobody as the next of kin of his account after his death.

The amount in this account is currently $32,640,000 (Thirty Two Million Six Hundred and Forty Thousand United States Dollars) I want to present you as a beneficiary,I will use my position and influence in our bank to make they release this money to you for us to share.If i wait for days and i do not hear from you,I shall look for another person.

Kindly get back to me for more details

Yours sincerely

...

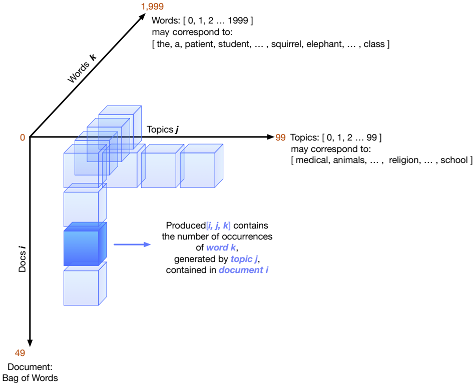
Mr Khim Leang
Board member
Foreign Trade Bank of Cambodia
Phnom Penh

"Bag of Words"

- Might build a dictionary
  - That is, map from each of m unique words in corpus
  - To a number from $\{1 ... m\}$
  - Then, each email is a vector $\langle 1, 0, 2, 1, 0, 0, ... \rangle$
  - jth entry is num occurrences of word j
  - Latent Dirichlet Allocation (LDA) uses this approach

LDA - Latent Dirichlet Allocation

- Recall Lab - Numpy Arrays
- LDA - Topic modeling
  - Each document in a collection is represented as a mixture of topics
  - Each topic is represented as a mixture of words.



“Bag of Words”

- Might build a dictionary
  - That is, map from each of m unique words in corpus
  - To a number from $\{1...m\}$
  - Then, each email is a vector $\langle 1,0,2,1,0,0,...\rangle$
  - jth entry is num occurrences of word j
  - Latent Dirichlet Allocation (LDA) uses this approach
  - ? Are there issues with this approach?

“Bag of Words Issues”

1. Sequence information is lost
   - Perhaps it’s important to know which words appear early on
   - ... or at the end
2. Word importance is lost
3. Some words are equivalent, but look different

1. Preserving Sequence Information

- Use N-grams
- N consecutive items
- *The cow jumped over the moon*
- ? What are the 2-grams in this sentence?
- ? What are the 3-grams in this sentence?

N-Grams for Spam Detection

- Words in an email might not be suspicious
- Might be how they are put together
  - “great sorrow”
  - “heavy tears”
  - “financial institution”
- Idea: also include all 2-grams, 3-grams, 4-grams, etc. as features

N-gram Example

- *The cow jumped over the moon*

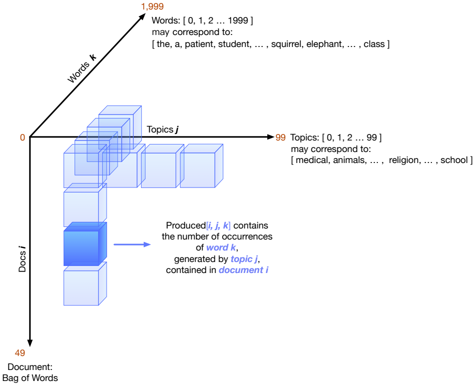
- 2-grams / bigrams
  1. *The cow*
  2. *cow jumped*
  3. *jumped over*
  4. *over the*
  5. *the moon*

  ? Issues with N-grams?

- 3-grams / trigrams
  1. *The cow jumped*
  2. *cow jumped over*
  3. *jumped over the*
  4. *over the moon*

Issues with N-grams

- # of combinations explodes
- The data dimensionality can get to billions
- But the feature vector is typically sparse
- Again, recall the numpy array lab - many entries in the matrix were zero



"Bag of Words Issues"

1. Sequence information is lost
2. Word importance is lost
   - Some words are more significant than others
3. Some words are equivalent, but look different

2. Word importance is lost

1. Eliminate “Stop Words”
   - Common words are often filtered out
   - ... a, an, the, is, with ...
2. TF-IDF

TF-IDF

- Term Frequency, Inverse Document Frequency
- Two components
  1. “Term Frequency” – frequency of each word in the document

$$TF = \frac{\text{num occurs of word in doc}}{\text{num words in doc}}$$

  2. “Inverse Document Frequency” – rareness of the word in the corpus

$$IDF = \log \frac{\text{num of docs}}{\text{num of docs having the word + 1}}$$

? Why “+1”?

- TD-IDF defined as $TF \times IDF$

TF-IDF Intuition

$$TF = \frac{\text{num occurs of word in doc}}{\text{num words in doc}}$$

$$IDF = \log \frac{\text{num of docs}}{\text{num of docs having the word + 1}}$$

$$TD - IDF = TF \times IDF$$

- Words that appear more often have a higher value
- Longer documents are expected to have more words, so we TF uses the number of words in the document
- Words that are rare should have a higher value

"Bag of Words Issues"

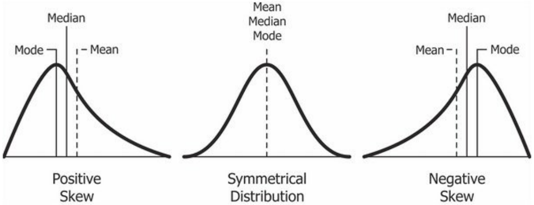
1. Sequence information is lost
2. Word importance is lost
3. Some words are equivalent, but look different
   - Stemming - reduce a word to its base
   - Remove punctuation?
   - Ignore capitalization?

Other Features for Spam Detection

- Country of sender
- Number of words in email
- Time of day sent
- Was the email sent previously? Does it include an email I sent?
- Recipient list disclosed? If not, indicative of spam
- Capitalization

Feature Engineering

- Applying domain knowledge to create useful features
- Examples
  - Spam email features (previous slide)
  - Predicting intracranial pressure crises: Time since last crisis
  - “Binning” values: Low / Medium / High
  - Time series features
    - Window
    - Min / Max / Mean / Median
    - Variance
    - Skew / Kurtosis



Feature Engineering: Don’t Cheat!

- You can only use the information available at the prediction time
- Examples
  - Predicting Intracranial pressure crises: Whether or not the patient died

Feature Engineering: Avoid / Minimize Bias

- If our examples of spam all come from certain countries, we are more likely to categorize ANY email from those countries as spam

Feature Selection

- Finding the helpful features

  Why not use them all?

Feature Selection

- Finding the helpful features
- Why not use them all?
  1. To avoid overfitting
  2. To reduce complexity
  3. To enable faster training
  4. To avoid having a sparse feature space where $n \ll p$ (Curse of dimensionality)

Techniques for Feature Selection

1. Embedded methods
   1. LASSO Regularization (last lecture)
2. Filter methods
   1. Statistical measures of relevance (e.g. Chi squared, information gain, correlation coefficient)
   2. Often univariate
3. Wrapper methods
   1. Search problem
   2. Try different combinations of features
   3. Compare results
   4. Can be Greedy, Stochastic, Exhaustive, Forward, Backward

A4

- kNN classifier to classify text documents
- Most frequent label in the k nearest neighbors
- Using the L_2 norm
- Implemented in PySpark
- Data
  - "20 newsgroups" data set
  - Old-school blog
  - 19,997 documents
- Labels
  - 20 categories
  - Which newsgroup the document was posted in
  - ... comp.graphics, rec.autos, sci.space, ...

Questions?

- What do we know now that we didn't know before?
- How can we use what we learned today?