COMP5310 - Week 5

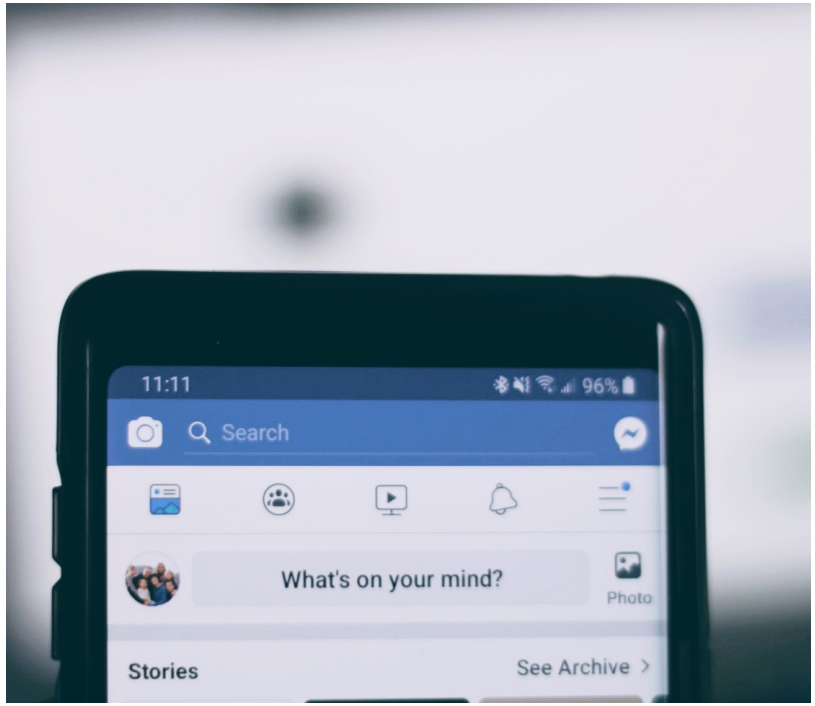
Analysing Unstructured Data

Social Media Data

http://hubpages.com/technology/Big-Data-Understanding-New-Insights#

Scenario: Social Media Site of an Organisation

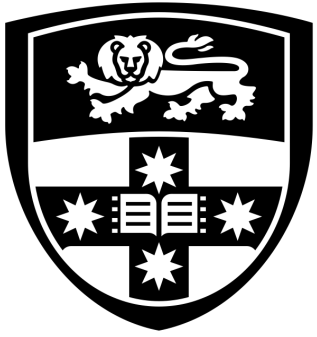
- Nowadays, every large organization, government, or business maintains own social network pages
- Need to keep track of the user feedback and comments
  - Too much traffic to do manually
- Can we use machine learning to
  - automatically filter SPAM?
  - identify common themes of posts?
  - do a sentiment analysis of posts?



Credit: Unsplash

Analysing Social Media Data?

- Social Media Data is **unstructured**
- consists mainly of text, images, videos
- Interesting questions
  - Classification of texts or images
  - Information Extraction
  - Building ML models based on unstructured data



THE UNIVERSITY OF
SYDNEY

Unstructured Data

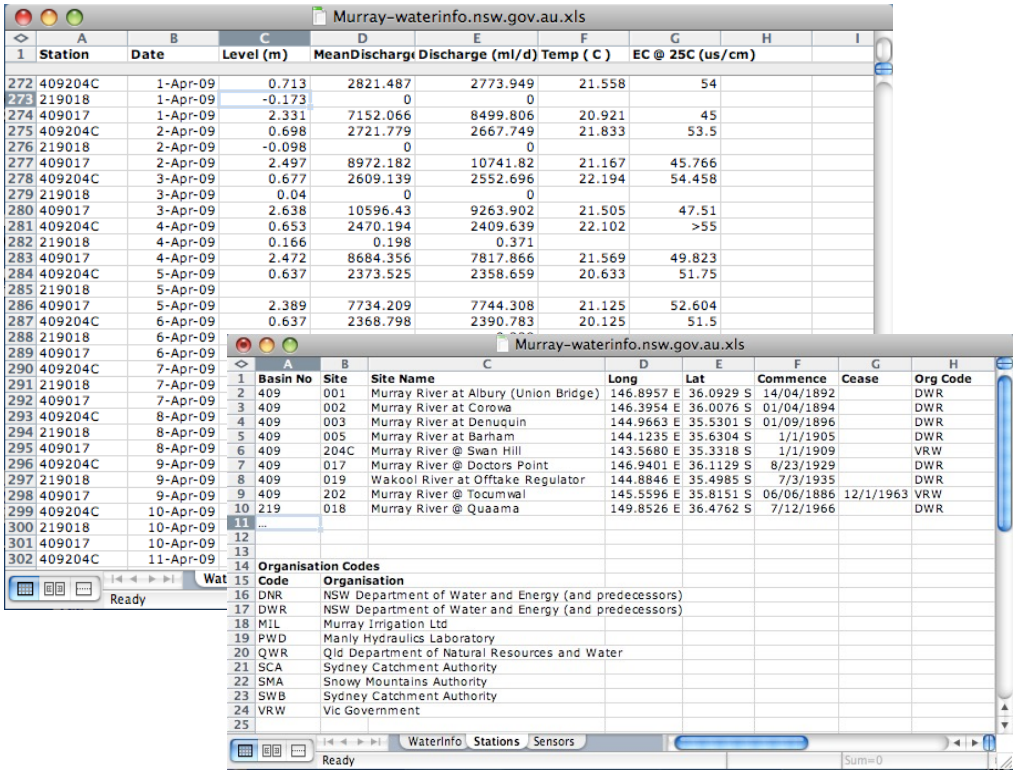
Unstructured Data

Unstructured data refers to information that usually does not have a pre-defined data model, such as Text, Images, Videos, …

Unstructured (text) data is typically text-heavy, but may contain dates, numbers and facts as well as meta-data.

This results in ambiguities that make it more difficult to understand than data in structured databases.

Structured Data



Murray-waterinfo.nsw.gov.au.xls

	A	B	C	D	E	F	G	H	I
1	Station	Date	Level (m)	MeanDischarge	Discharge (ml/d)	Temp (C)	EC @ 25C (us/cm)		
272	409204C	1-Apr-09	0.713	2821.487	2773.949	21.558	54		
273	219018	1-Apr-09	-0.173	0	0				
274	409017	1-Apr-09	2.331	7152.066	8499.806	20.921	45		
275	409204C	2-Apr-09	0.698	2721.779	2667.749	21.833	53.5		
276	219018	2-Apr-09	-0.098	0	0				
277	409017	2-Apr-09	2.497	8972.182	10741.82	21.167	45.766		
278	409204C	3-Apr-09	0.677	2609.139	2552.696	22.194	54.458		
279	219018	3-Apr-09	0.04	0	0				
280	409017	3-Apr-09	2.638	10596.43	9263.902	21.505	47.51		
281	409204C	4-Apr-09	0.653	2470.194	2409.639	22.102	>55		
282	219018	4-Apr-09	0.166	0.198	0.371				
283	409017	4-Apr-09	2.472	8684.356	7817.866	21.569	49.823		
284	409204C	5-Apr-09	0.637	2373.525	2358.659	20.633	51.75		
285	219018	5-Apr-09							
286	409017	5-Apr-09	2.389	7734.209	7744.308	21.125	52.604		
287	409204C	6-Apr-09	0.637	2368.798	2390.783	20.125	51.5		
288	219018	6-Apr-09							
289	409017	6-Apr-09							
290	409204C	7-Apr-09							
291	219018	7-Apr-09							
292	409017	7-Apr-09							
293	409204C	8-Apr-09							
294	219018	8-Apr-09							
295	409017	8-Apr-09							
296	409204C	9-Apr-09							
297	219018	9-Apr-09							
298	409017	9-Apr-09							
299	409204C	10-Apr-09							
300	219018	10-Apr-09							
301	409017	10-Apr-09							
302	409204C	11-Apr-09							

Ready

Murray-waterinfo.nsw.gov.au.xls

	A	B	C	D	E	F	G	H
1	Basin No	Site	Site Name	Long	Lat	Commence	Cease	Org Code
2	409	001	Murray River at Albury (Union Bridge)	146.8957 E	36.0929 S	14/04/1892		DWR
3	409	002	Murray River at Corowa	146.3954 E	36.0076 S	01/04/1894		DWR
4	409	003	Murray River at Denuquin	144.9663 E	35.5301 S	01/09/1896		DWR
5	409	005	Murray River at Barham	144.1235 E	35.6304 S	1/1/1905		DWR
6	409	204C	Murray River @ Swan Hill	143.5680 E	35.3318 S	1/1/1909		VRW
7	409	017	Murray River @ Doctors Point	146.9401 E	36.1129 S	8/23/1929		DWR
8	409	019	Wakool River at Offtake Regulator	144.8846 E	35.4985 S	7/3/1935		DWR
9	409	202	Murray River @ Tocumwal	145.5596 E	35.8151 S	06/06/1886	12/1/1963	VRW
10	219	018	Murray River @ Quaama	149.8526 E	36.4762 S	7/12/1966		DWR
11	...							
12								
13								
14	Organisation Codes							
15	Code	Organisation						
16	DNR	NSW Department of Water and Energy (and predecessors)						
17	DWR	NSW Department of Water and Energy (and predecessors)						
18	MIL	Murray Irrigation Ltd						
19	PWD	Manly Hydraulics Laboratory						
20	QWR	Qld Department of Natural Resources and Water						
21	SCA	Sydney Catchment Authority						
22	SMA	Snowy Mountains Authority						
23	SWB	Sydney Catchment Authority						
24	VRW	Vic Government						
25								

WaterInfo | Stations | Sensors

Ready

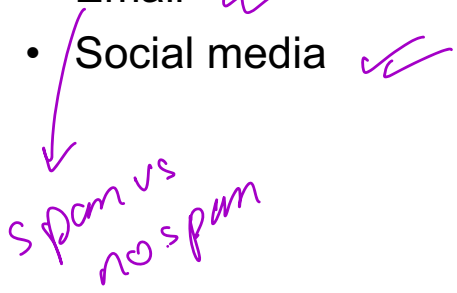
- Data in fields
- Easily stored in databases
- E.g.:
  - Sensor data
  - Financial data
  - Click streams
  - Measurements

Text is Unstructured Data

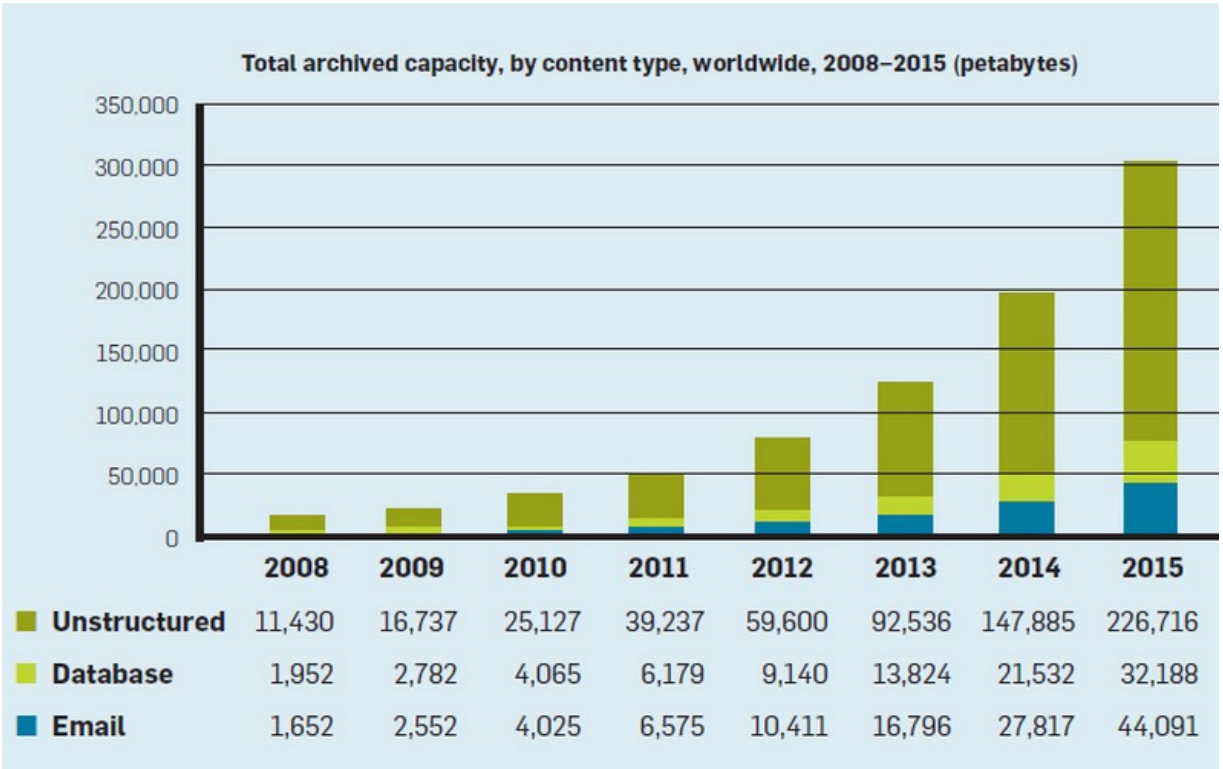
"In the history of cinematic mustaches, few have been as disgusting as that of Rye Gerhardt (Kieran Culkin), the youngest scion of North Dakota's reigning crime family and the stray spark that sets off the powder-keg second season of Fargo."

(From a Slate review of Fargo Season 2)

- 80% - 90% of all potentially usable business information
- E.g.:
  - Images
  - Video
  - Email
  - Social media

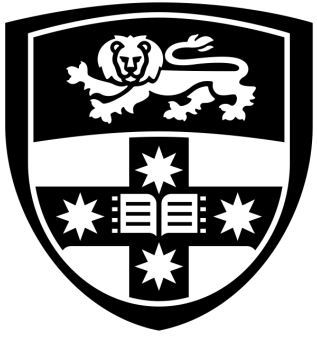


Information Overload



	2008	2009	2010	2011	2012	2013	2014	2015
Unstructured	11,430	16,737	25,127	39,237	59,600	92,536	147,885	226,716
Database	1,952	2,782	4,065	6,179	9,140	13,824	21,532	32,188
Email	1,652	2,552	4,025	6,575	10,411	16,796	27,817	44,091

http://bhavnaober.blogspot.com.au/2015_02_01_archive.html



THE UNIVERSITY OF
SYDNEY

Text Classification

Part 1: Feature Extraction from Text Documents

Example: Legitimate eMail – or SPAM?

Administrative Support, workshop

mail@ncoa.com.au
06/04/2019 at 08:59:15
To: Uwe Roehm <uwe.roehm@sydney.edu.au> Details

2 Attachment(s) Total 768.8 KB View

To: Human Resources

Please find attached our workshop brochure, Administrative Support.

Kindest regards,

Kathryn

Kathryn

Customer Service Representative

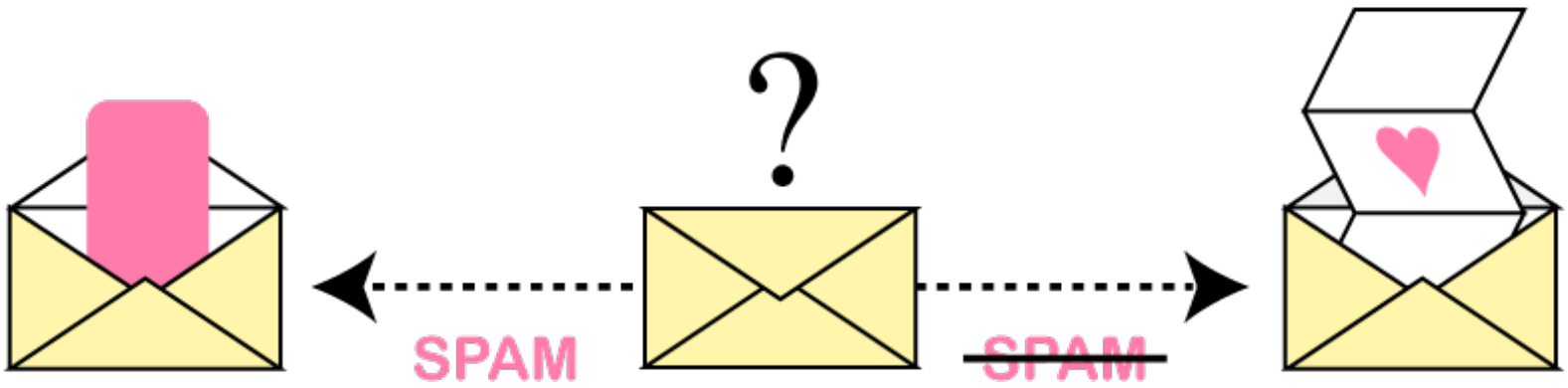
National College of Administration

Telephone: 1300 796 551

Email: [redacted]

www. collegeofadministration.com.au

Motivation Task: Spam/Not-spam Detection

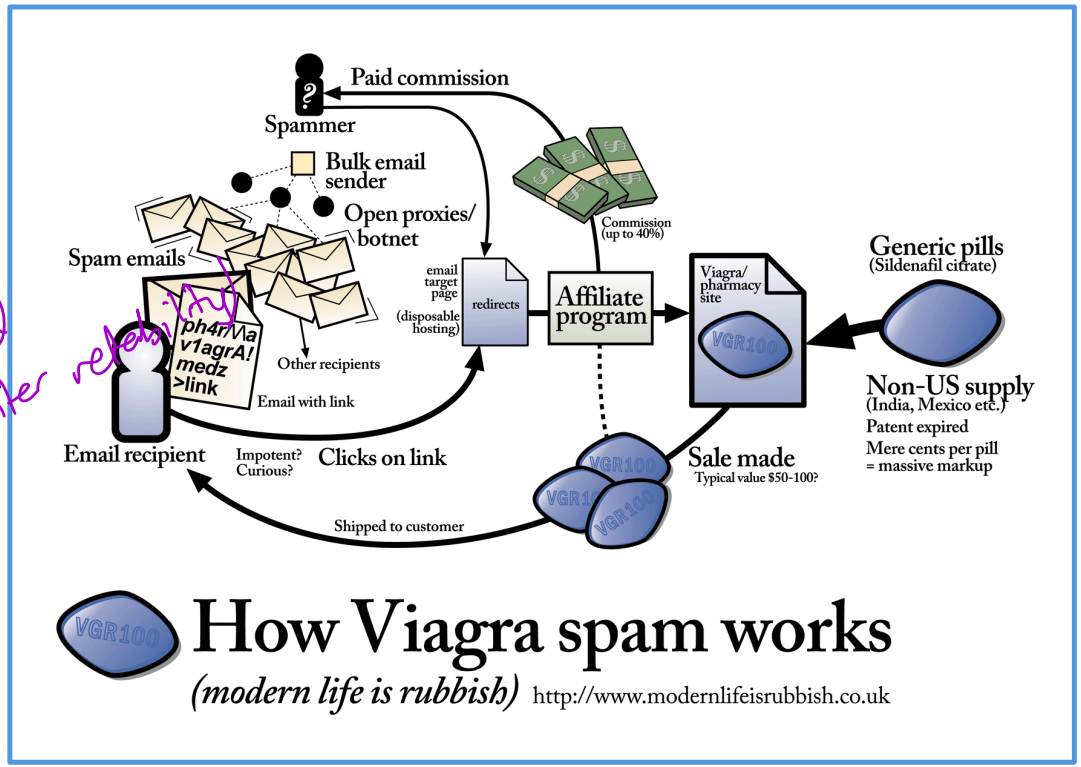


http://blog.dato.com/how-to-evaluate-machine-learning-models-part-2a-classification-metrics

Modelling Spam Detection as Classification Task

- Input:
  - Emails
  - SMS messages
  - Facebook pages
  - …
- Predict:
  - 1 (spam)
  - 0 (not-spam)

- Step 1: **Feature Extraction**



Core Idea: Text to Feature Vectors

Recall NLP folder from codebasics

- Represent document as a multiset of words
- Keep frequency information
- Disregard grammar and word order

- **Feature Vector**
  Which words occur how often in a given text?

http://www.python-course.eu/text_classification_python.php

Tokenisation

can be more than just singular words. represented by the tokens

"Friends, Romans, Romans, countrymen"

["Friends",
"Romans",
"Romans",
"countrymen"]

- Split a string (document) into pieces called **tokens**
- Possibly remove some characters, e.g., punctuation
- Remove "stop words" such as "a", "the", "and" which are considered irrelevant

- What about "O'Neill"? "Aren't"?

* Again, we ignore all punctuations

Normalisation

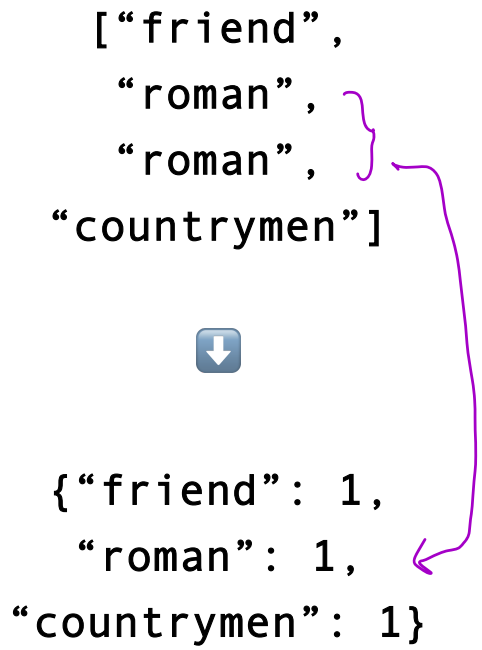
```
["Friends",
 "Romans",
 "Romans",
"countrymen"]
```

⬇️

```
["friend",
 "roman",
 "roman",
"countrymen"]
```

- Map similar words to the same token
- Stemming/lemmatisation
  - Avoid grammatical and derivational sparseness
  - E.g., “was” => “be”
- Lower casing, encoding
  - E.g., “Naïve” => “naive”

Indicator Features



- Binary indicator feature for each word in a document
- Ignore frequencies

1: occurs in the document
0: does not occur in the document.

Term Frequency Weighting

How often does a term occur in a specific document?

```
["friend",
 "roman",
 "roman",
"countrymen"]
```

```
{"friend": 1,
 "roman": 2,
"countryman": 1}
```

- Term frequency
  - Give more weight to terms that are common in document
  - ***TF = \|occurrences of term in doc\|***
- Damping
  - Sometimes want to reduce impact of high counts
  - ***TF = log(\|occurrences of term in doc\|)***

if there's too many counts

Example

Convert the following document into a bag of terms:

For a **successful** technology, reality must take precedence over public relations, for Nature cannot be fooled.

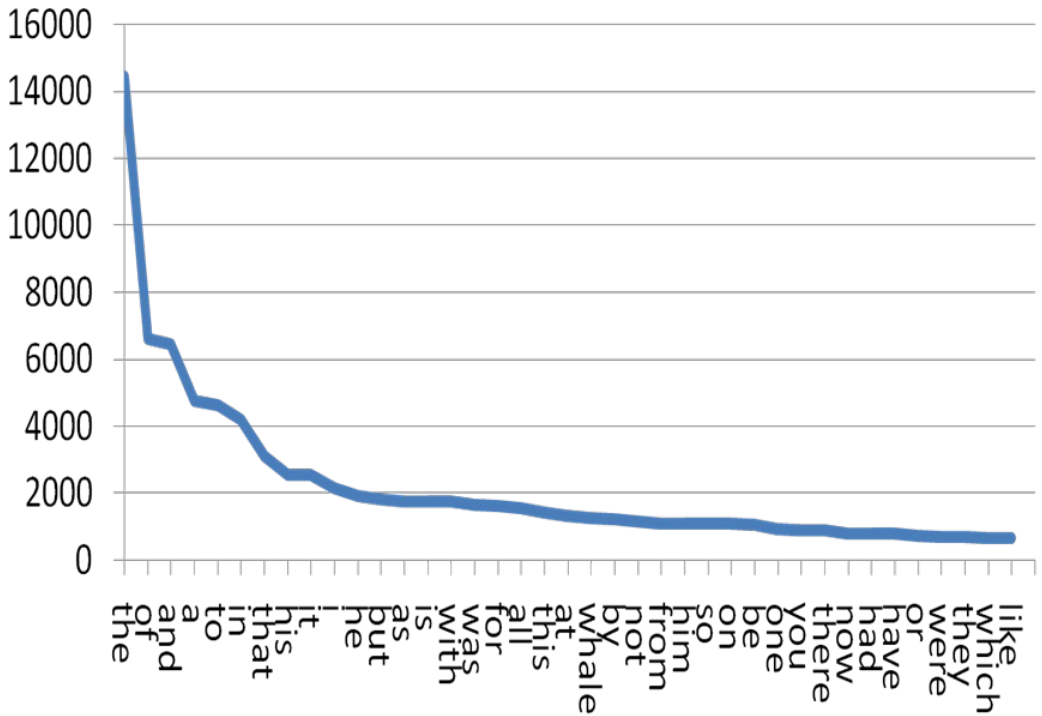
```
<blockquote
cite="http://www.brainyquote.com/quotes/quotes/r/richardpf104988.html">
For a <b>successful</b> technology, reality must take precedence over public
relations, for <u>Nature</u> cannot be fooled.
</blockquote>
```

a be cannot fooled for for
must nature over precedence
public reality relations successful take technology

cannot fool must nature over precede
public real relatn success take technology

after lemmatisation

Zipf Term Distribution



"Moby Dick"
Herman Melville

http://searchengineland.com/the-long-tail-of-search-12198

TF-IDF Weighting

Inverse Document Frequency → Dampening factor.

refer to COMP2521 Assignment 1

Term Frequency

```
["friend",
 "roman",
"countrymen"]
```

⬇️

```
{"friend": 0.1,
 "roman": 0.8,
"countrymen": 0.2}
```

- Inverse document frequency
  - Give less weight to terms that are common across documents
  - deals with the problems of the Zipf distribution
  - ***IDF = log(|docs|/|docs containing term|)***
- TF-IDF
  - ***TFIDF = TF * IDF***

$$IDF = \log\left(\frac{|docs|}{|docs\ containing\ term|}\right)$$

$$TFIDF = TF \times IDF$$

Vector Space Model

- Documents are represented as *vectors* in term space
  - Terms are usually stems
  - Document vector values can be weighted by, e.g., frequency
- Queries represented the same as documents

	nova	galaxy	heat	h'wood	film		role	diet	fur
A	10	5	3						

"Nova" occurs 10 times in text A
"Galaxy" occurs 5 times in text A
"Heat" occurs 3 times in text A
(Blank means 0 occurrences.)

These numbers all represent Term Frequency (TF)

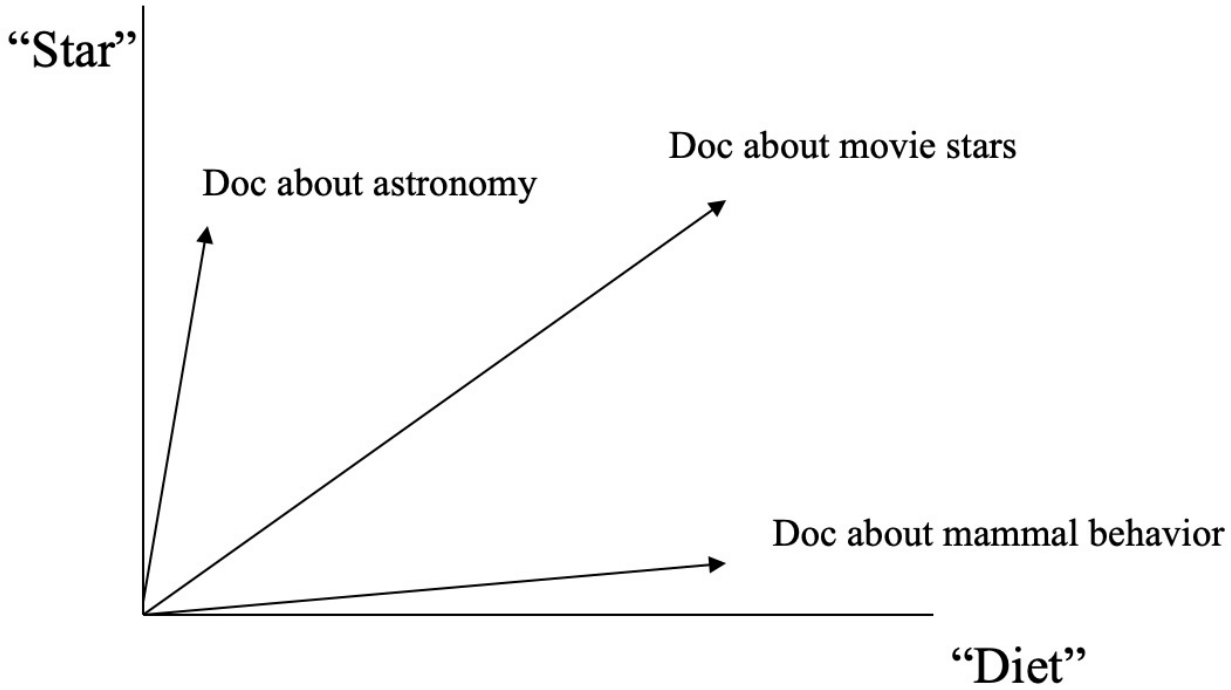
Document Vectors

- All document vectors together: *Document-Term-Matrix (Feature-Matrix)*

Document ids

	nova	galaxy	heat	h' wood	film	role	diet	fur
A	10	5	3					
B	5	10						
C				10	8	7		
D				9	10	5		
E							10	10
F							9	10
G	5		7			9		
H		6	10	2	8			
I				7	5		1	3

We Can Plot the Vectors



Assumption: Documents that are close in direction and length are similar to one another.

Feature Extraction in Python

- Scikit-learn library provides corresponding functionality via its **CountVectorizer**
- Example:

```
from sklearn.feature_extraction.text import CountVectorizer
from pprint import pprint
corpus = [ 'This is the first document.',
           'This is the second second document.',
           'And the third one.',
           'Is this the first document?', ... ]
vectorizer = CountVectorizer()
matrix     = vectorizer.fit_transform(corpus)
pprint(matrix)
```

* This can also be done on images.

images → vectors.

- https://scikit-learn.org/stable/modules/feature_extraction.html#text-feature-extraction
- See also: https://adataanalyst.com/scikit-learn/countvectorizer-sklearn-example/

Feature Extraction in Python (cont'd)

- **CountVectorizer** can be configured in quite some detail
  - By default, CounterVectorizer does tokenization for single words of minimum length 2
    - Change to also consider bigrams (terms consisting of 2 words):

      ```
      vectorizer = CountVectorizer(ngram_range=(1,2))
      ```
  - Convert input text to lower case; also ignore certain accents in text:

    ```
    vectorizer = CountVectorizer(lowercase=True, strip_accents='ascii')
    ```
  - Use indicator features (0 or 1) rather than term frequencies

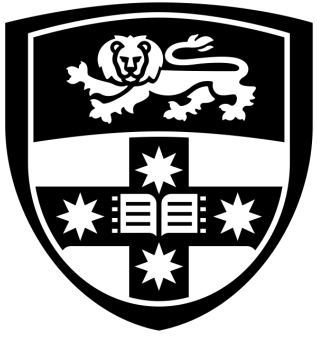
    ```
    vectorizer = CountVectorizer(binary=True)
    ```
  - Specify a list of *stop words* that get ignored

    ```
    vectorizer = CountVectorizer(stop_words=['the','a'])
    ```
  - Only keep features within a certain document frequency range

    ```
    vectorizer = CountVectorizer(min_df=0.1, max_df=0.5)
    ```
- Example: `CountVectorizer(lowercase=True,strip_accents='ascii',binary=True)`

Feature Extraction with word2vec

- Open Source Tool by Google
  - https://code.google.com/archive/p/word2vec/



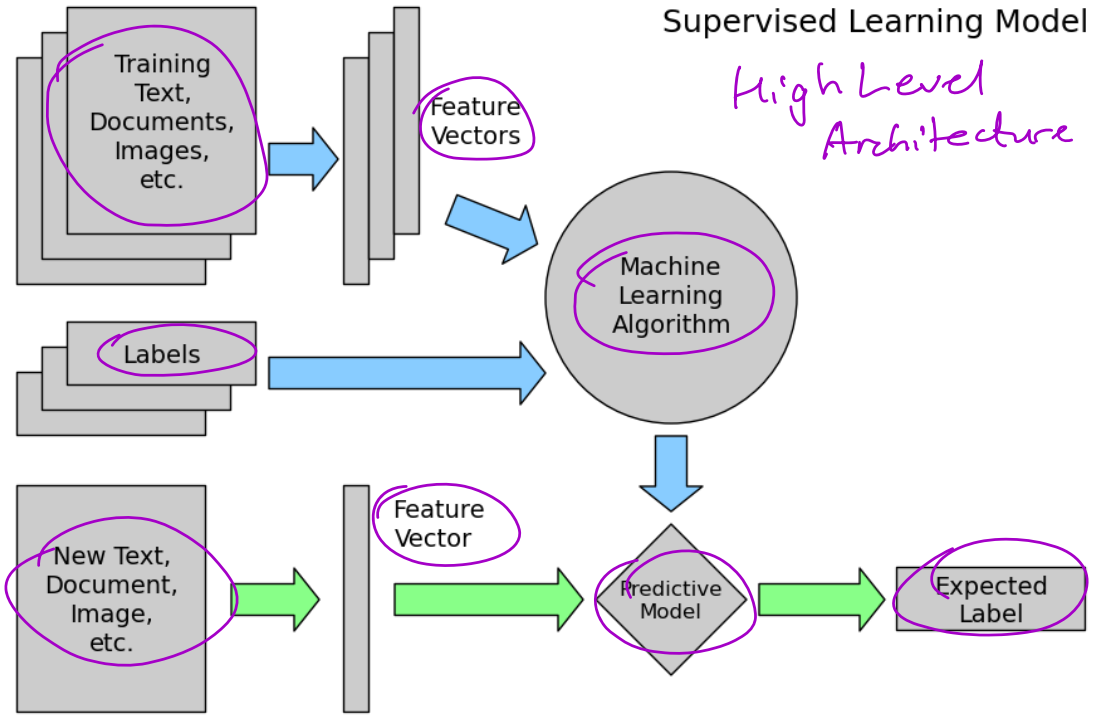
THE UNIVERSITY OF
SYDNEY

Text Classification

Part 2: Text Classification using Supervised Machine Learning

Spam Detection as Supervised Classification

- Input:
  - labelled text data
- Output:
  - A model that divides instances of the input data and can be used to classify future (unknown) data
- In our example: SPAM-/not-SPAM labelled documents



http://www.astroml.org/sklearn_tutorial/general_concepts.html#supervised-learning-model-fit-x-y

Roadmap

- Importing Data
  - Optional: data transformation and cleaning
- Feature Extraction
  - Text to vectors
- Machine Learning:
  - Split data into test and training sets
  - Choose and train model(s)
  - Evaluate
- Use best model for predictions / classifications

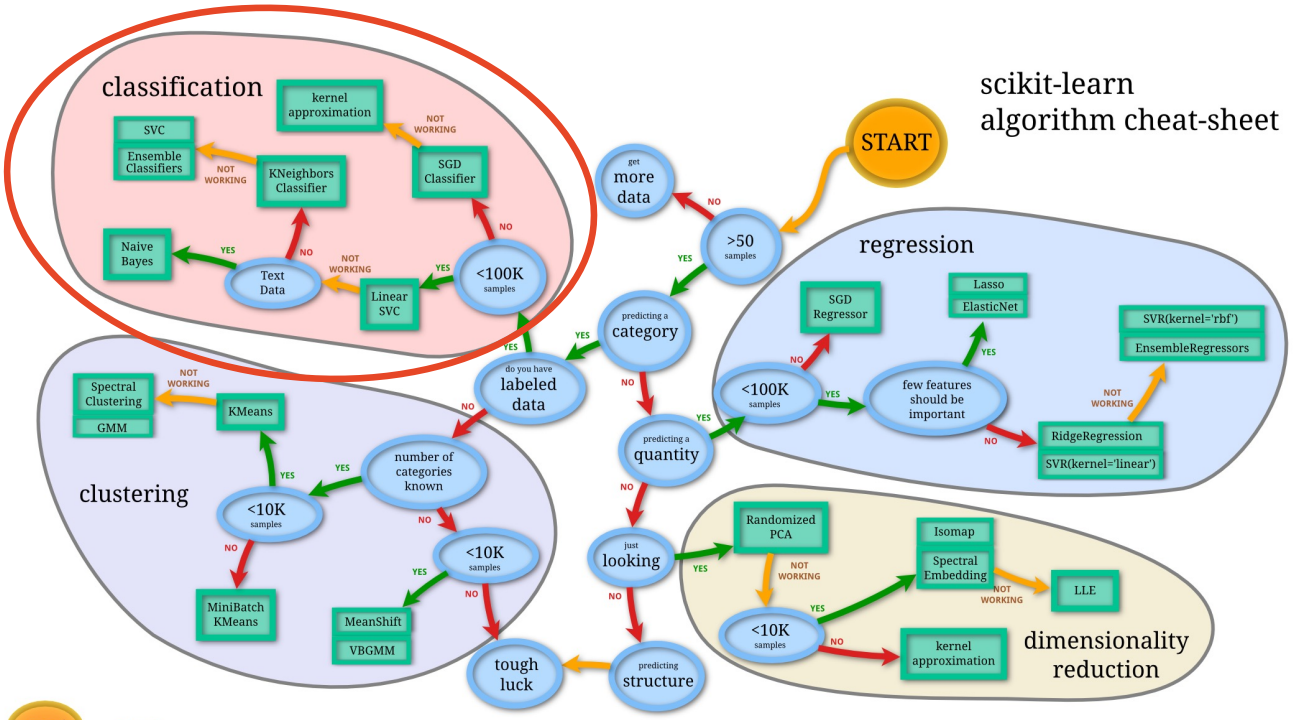
SMS Spam Detection

Label	Message Snippet
Ham	Go until jurong point, crazy.. Available only …
Spam	Ok lar… Joking wif u oni...
Ham	U dun say so early hor… U c already then say...
Spam	FreeMsg Hey there darling it's been 3 week's n…

- 425 SMS spam messages from UK Grumbletext web forum
- 3,375 ham randomly chosen from NUS SMS corpus (students)
- 450 ham from somebody's PhD thesis
- 322 spam and 1,002 ham from SMS Spam Corpus

https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection

Machine Learning Map from scikit-learn



http://scikit-learn.org/stable/tutorial/machine_learning_map/

Use scikit-learn Pipeline to manage Cross Validation

```
from sklearn.model_selection import GridSearchCV
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.metrics import classification_report
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC
```

Scikit-learn Pipelines provide a mechanism for fitting and predicting a sequence of components. This is good practice to avoid data leakage.

```
# Pipeline for multinomial naive Bayes
mnb = Pipeline([('vect', CountVectorizer(lowercase=False)),
                ('tfidf', TfidfTransformer()),
                ('clf', MultinomialNB())
               ])
```

Multinomial Naive Bayes Classifier

1. **Convert string to a bag-of-words token vector.**
2. **Transform vector counts using TFIDF weighting.**
3. **Train/predict using multinomial naïve Bayes.**

Data Leakage

What it is:

- Allowing your algorithm to use information that will not be available in production
- E.g., using a market index to predict individual stock performance

What to do:

- Understand your problem and data
- Don't trust nonsensical model components (e.g., index)
- If a result seems too good to be true, it probably is!

Use scikit learn for GridSearch-CrossValidation

```
from sklearn.model_selection import GridSearchCV
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.metrics import classification_report
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC
```

```
# Define grid search parameters
param_grid = [{'vect__binary': [True],
               'vect__ngram_range': [(1, 1), (1, 2)],
               'tfidf__use_idf': [True, False]
              },
              {'vect__binary': [False],
               'vect__ngram_range': [(1, 1), (1, 2)],
               'tfidf__use_idf': [True, False]
              }
             ]

# Find best parameters for MNB and SVM
gs_mnb = GridSearchCV(mnb, param_grid, cv=3)
gs_mnb.fit(X_train, y_train)
print('\nMNB best params:\n', gs_mnb.best_params_)

gs_svm = GridSearchCV(svm, param_grid, cv=3)
gs_svm.fit(X_train, y_train)
print('\nSVM best params:\n', gs_svm.best_params_)
```

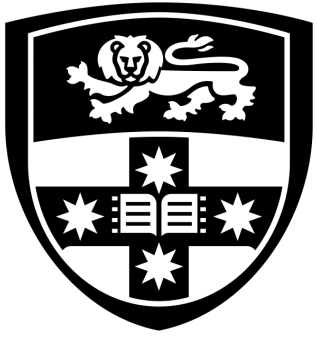
These 2 classifiers work well on text classification problems.

MNB :

SVM :

Evaluation metrics:

- Precision-recall
- F1 score etc... same as normal classification.



THE UNIVERSITY OF
SYDNEY

Text-driven Forecasting

Unstructured Data in Supervised Regression

- Unstructured data can also be input for regression
  (predict a numeric value, not a categorical label)
- Example: Predict box-office return from the movie reviews
- Example: Predict share price changes from stock-market announcements

Text-driven Forecasting

predictor(s)

Given a body of text *T* pertinent to a social phenomenon, make a concrete prediction about a measurement *M* of that phenomenon, obtainable only in the future.

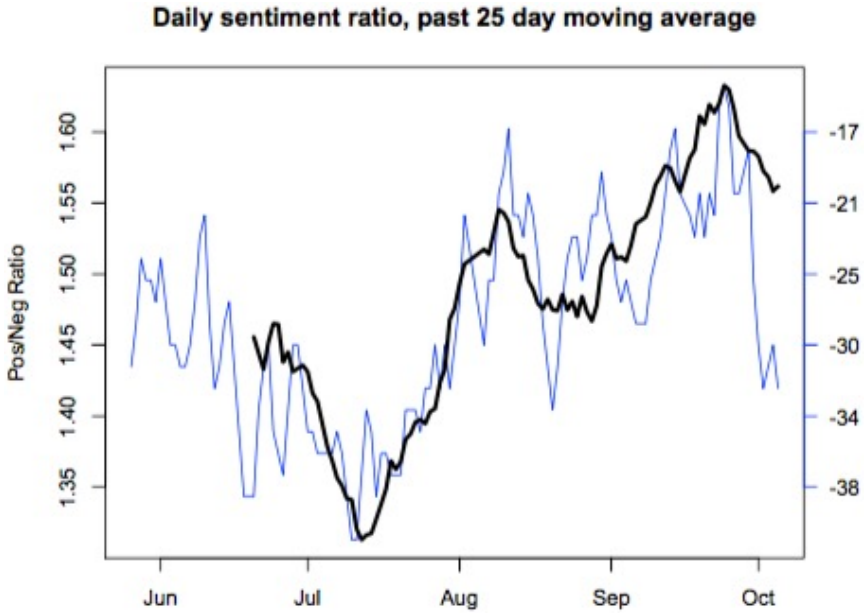
outcome

http://www.cs.cmu.edu/~nasmith/papers/smith.whitepaper10.pdf

Some Text-driven Forecasting Tasks

- Predict box office gross for films
  - T: description, script, reviews, etc
  - M: how much the film earns at the box office
- Predict volatility of a stock
  - T: annual report, etc
  - M: volatility over the following year
- Predict blog reader behaviour
  - T: political blog posts, etc
  - M: number of reader comments

Predicting Public Opinion from Tweets

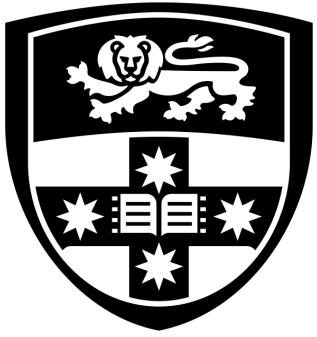


- T: tweets mentioning the word "economy"
- M: Gallup's economic confidence index (blue)
- Predictions (black) closely track Gallup's polling data

https://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/viewFile/1536/1842

Text-driven Forecasting

- CMU seminar on text-driven forecasting. http://www.cs.cmu.edu/~nasmith/TDF/
- Smith. Text-driven forecasting (whitepaper). https://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/viewFile/1536/1842
- Henry. Predicting with words (blog post). http://harmony-institute.org/latest/2012/10/19/forecasting-the-influence-of-entertainment/



THE UNIVERSITY OF
SYDNEY

Information Extraction

Many Language Tasks require Structured Prediction

Part-of-speech tagging.

Word	POS tag
This	DT (determiner)
is	VBZ (verb)
a	DT (determiner)
tagged	JJ (adjective)
sentence	NN (noun)
.	.

- **Structured prediction:** problems where output is a structured object, rather than discrete or real values
- E.g., sequence tagging for part-of-speech (POS) tagging or named entity recognition

We want to gain a clearer understanding of the semantics of these words.

https://en.wikipedia.org/wiki/Structured_prediction

Natural Language Processing

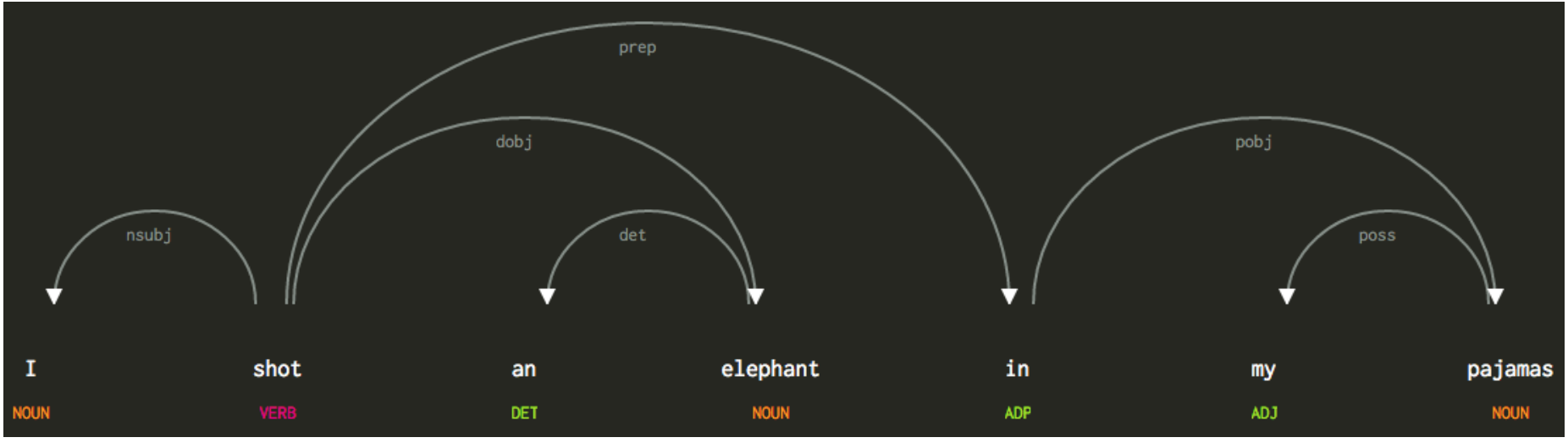
"Interdisciplinary field concerned with modelling natural language from a computational perspective."

https://en.wikipedia.org/wiki/Computational_linguistics

- Understanding
  - Tokenisation ✓✓
  - POS tagging ✓✓
  - Parsing ? Did we cover this?
- Generation
- Summarisation

– Lemmatisation.

Parsing Natural Language



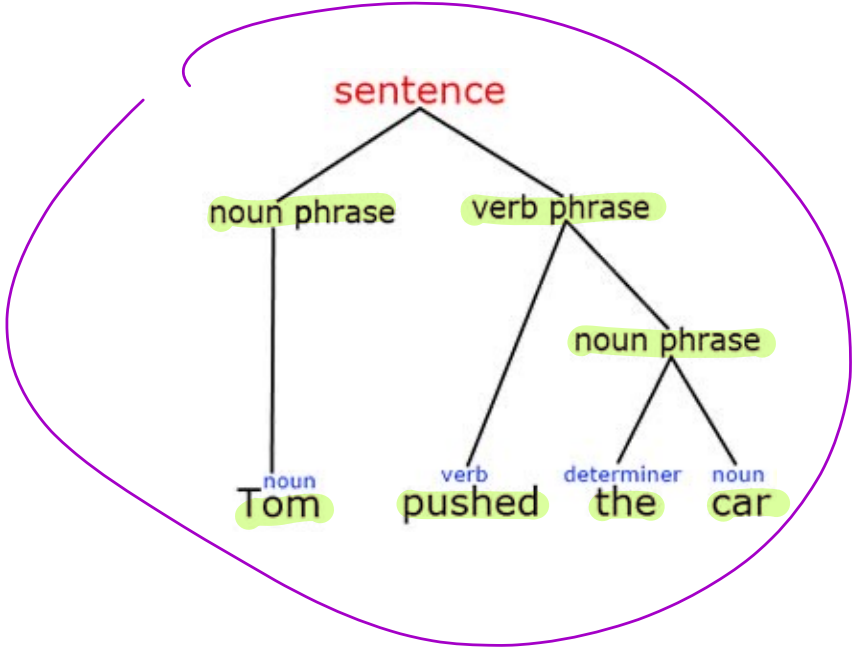
https://spacy.io/demos/displacy?share=2473569563126265042

Parsing etc in Python

Don't have to dwell on the semantics of how spaCy works.

- **spaCy** is an open-source software library for advanced Natural Language Processing, written in the programming languages Python and Cython.

- Visualisation of parses possible via https://spacy.io/demos/displacy



Information Extraction

"Task of automatically extracting structured information from unstructured and/or semi-structured documents."

https://en.wikipedia.org/wiki/Information_extraction

- Named entity recognition
- Entity disambiguation
- Relation extraction

Knowledge Base Population (KBP)

Not sure if this is useful to know...

- Aim is to build structured knowledge bases from massive unstructured text corpora
- Two subtasks:
  - **Entity linking**: identify mentions of entities, link to KB or NIL
  - **Slot filling**: extract and populate facts for given entity

Entity Linking

Which John Williams are we referring to?

Entity Linking helps us answer this question

John Williams

Richard Kaufman goes a long way back with **John Williams**. Trained as a classical violinist, Californian Kaufman started doing session work in the Hollywood studios in the 1970s. One of his movies was Jaws, with **Williams** conducting his score in recording sessions in 1975...

Michael Phelps

Debbie Phelps, the mother of swimming star **Michael Phelps**, who won a record eight gold medals in Beijing, is the author of a new memoir, ...

Michael Phelps is the scientist most often identified as the inventor of PET, a technique that permits the imaging of biological processes in the organ systems of living individuals. **Phelps** has ...

John Williams	author	1922-1994
J. Lloyd Williams	botanist	1854-1945
John Williams	politician	1955-
John J. Williams	US Senator	1904-1988
John Williams	Archbishop	1582-1650
John Williams	composer	1932-
Jonathan Williams	poet	1929-

Michael Phelps	swimmer	1985-
Michael Phelps	biophysicist	1939-

Identify matching entry, or determine that entity is missing from KB

Slot Filling

Target: EPA
(plus 1 document)

Generic Entity Classes
Person, Organization, GPE

Missing information to mine from text:

- **Date formed: 12/2/1970**
- **Website: http://www.epa.gov/**
- **Headquarters: Washington, DC**
- **Nicknames: EPA, USEPA**
- **Type: federal agency**
- **Address: 1200 Pennsylvania Avenue NW**

Slots we would like to fill.

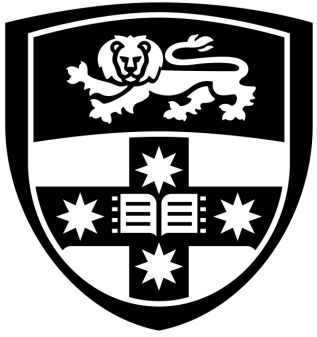
Need to mine some text to pla here.

Optional: Also want to link some learned values within the KB:

- **Headquarters: Washington, DC (kbid: 735)**

Overview of Wikification

- Roth and Ji. Wikification and beyond (slides 4-17)
  http://nlp.cs.rpi.edu/paper/wikificationtutorial.pdf



THE UNIVERSITY OF
SYDNEY

Model Evaluation

Setting up a Reliable Evaluation

- Aim is to create an experiment setup that
  - Is fair for approaches/participants
  - Prevents overfitting
  - Allows reliable comparison

Data Leakage

What it is:

- Allowing your algorithm to use information that will not be available in production → introducing new features
- E.g., using a market index to predict individual stock performance

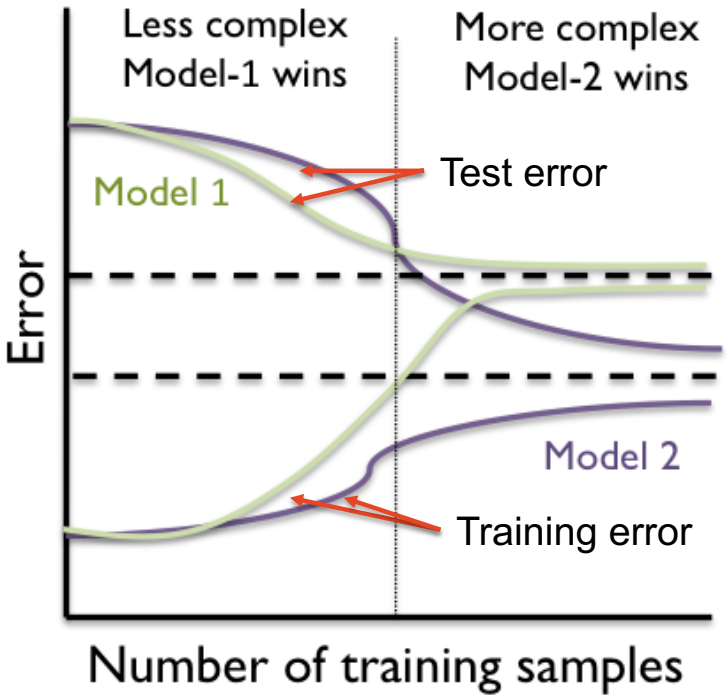
based on the past.

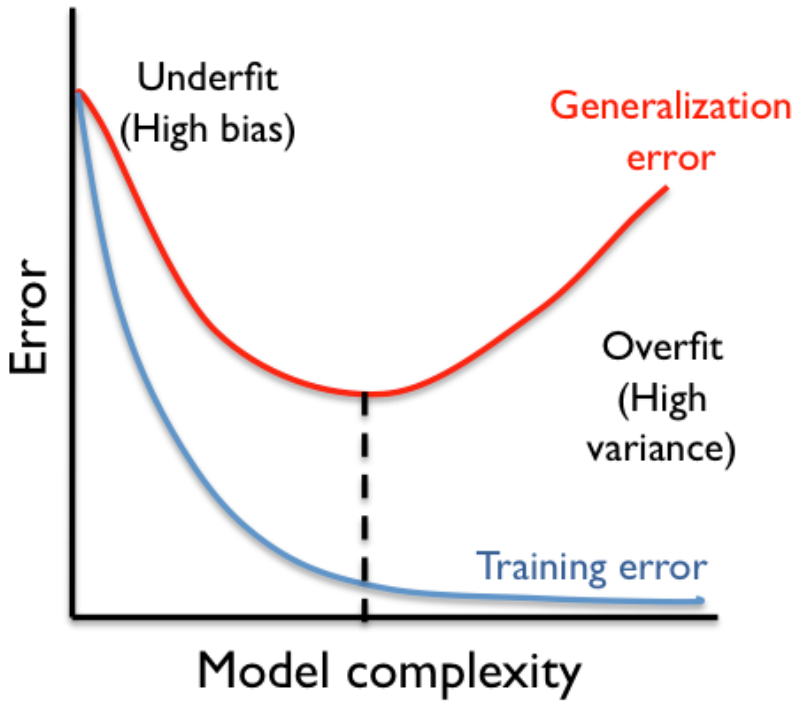
What to do:

- Understand your problem and data
- Don't trust nonsensical model components (e.g., index)
- If a result seems too good to be true, it probably is!

repeat of prev slides

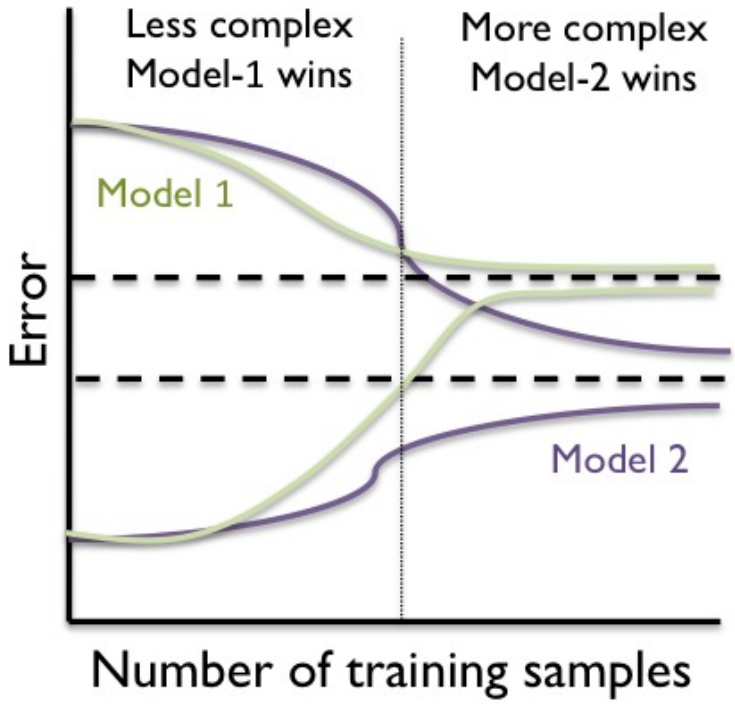
Generalisation: Accuracy on Unseen Data





https://thebayesianobserver.wordpress.com/2012/02/07/debugging-machine-learning-algorithms/

Model choice depends on amount of data available



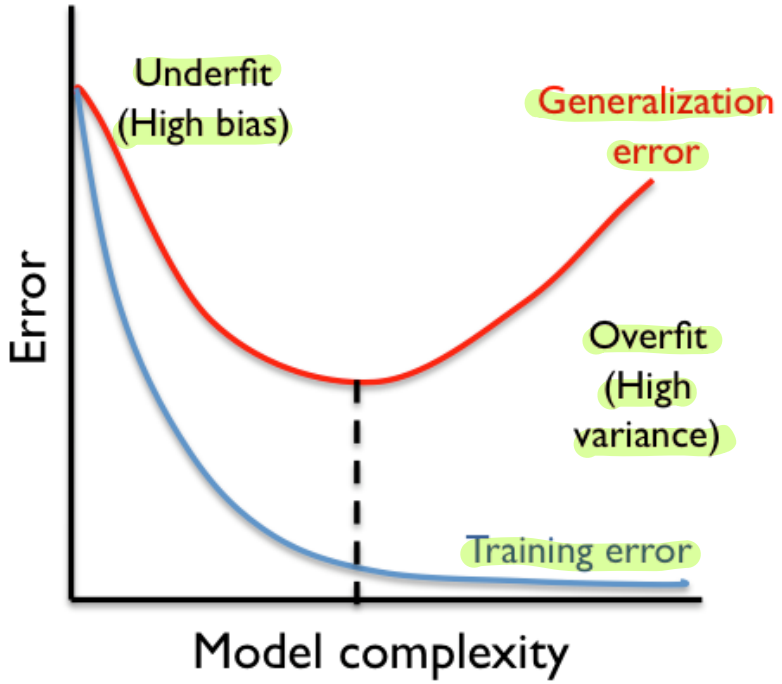
- Test error decreases
- Training error increases
- Two converge to asymptote
- If we can get more data, model 2 eventually wins
- Neither model will improve much with more data than we already have

since model 2 has the lower error

https://thebayesianobserver.wordpress.com/2012/02/07/debugging-machine-learning-algorithms/

Finding a Model that Generalises

- The dashed line on right shows point where we switch form underfitting to overfitting
- Goal: Find this dotted line
- Generalisation error should model application as closely and reliably as possible
  - Sample must be representative
  - Larger sample better

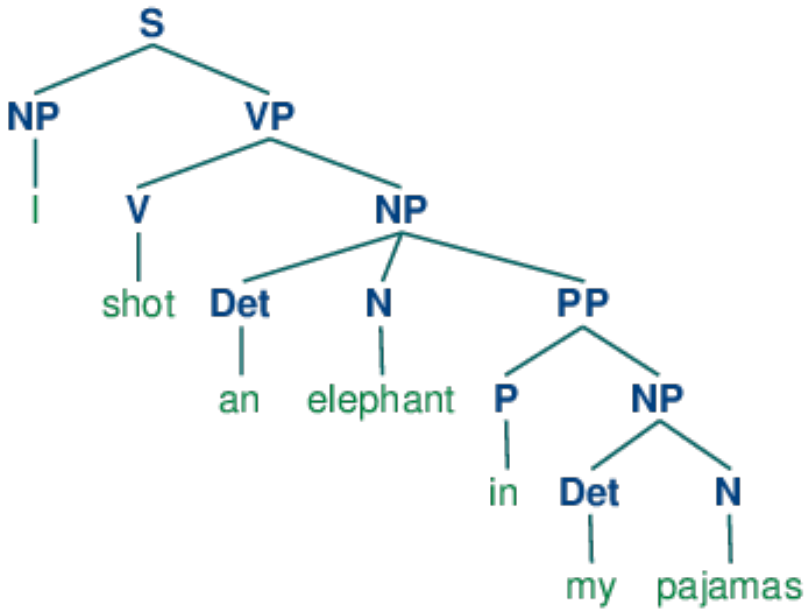


https://thebayesianobserver.wordpress.com/2012/02/07/debugging-machine-learning-algorithms/

IID: Independent and Identically Distributed

- Assumption that training and test examples are independently drawn from the same overall distribution of data
  - There is a distribution from which the data is generated
  - Training and test data are samples from the same distribution
  - Each example is independent of the other examples
- Train/test splits can be selected randomly

Non-IID Data: Human Language



- There is a relationship between labels
- E.g., predicting grammar
- Train/test splits include at least different sentences

Non-IID Data: Time Series

- Previous data is biggest single predictor
- E.g., predicting stock return
- Train split should always have data from before test

ECON2209 stuff...
/3206

Data Drift (non-identical/non-stationary data)

What it is:

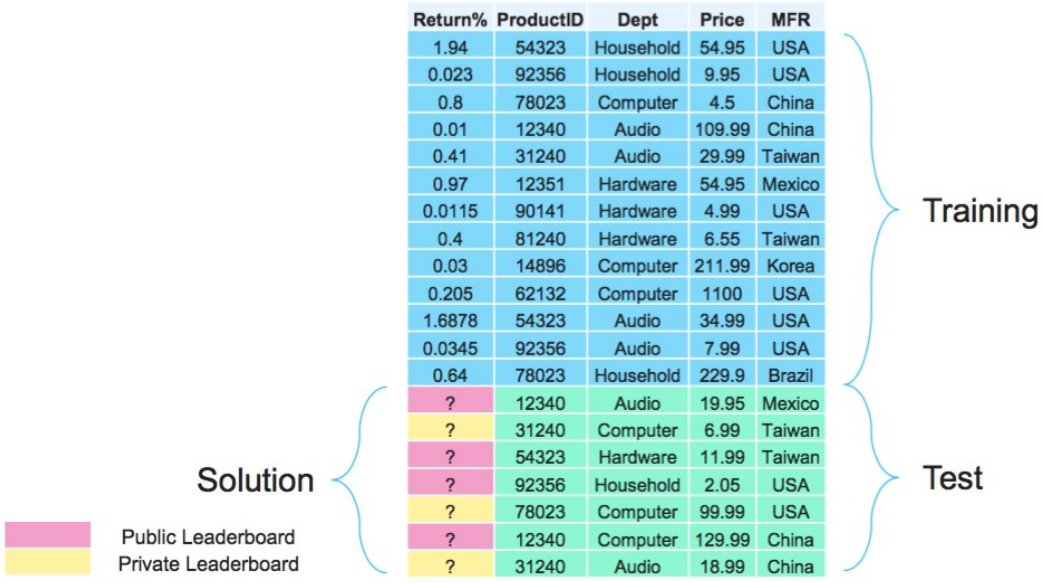
- Typical train/test setups assume stationarity
- Should be near-true for train and test samples
- Only near-true in production for a little while

What to do:

- Monitor offline metric on live data
- May require monitoring/annotation
- If there are large changes, then retrain on new data

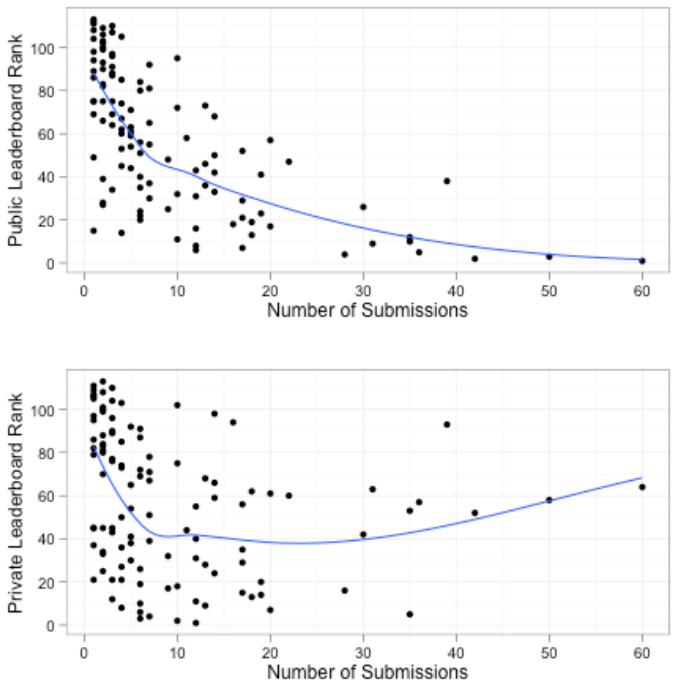
Seasonal patterns will affect the behaviour of the data. For example, 2008 → Global Financial Crisis.

The Kaggle Solution to Testing Generalisation



- Public leaderboard over random sample of test data
- Final competition score over remaining test data
- Participants don't know the split

Overusing Validation Data still leads to Overfitting



- One Kaggle participant reports dropping
  - 2nd on dev data
  - 52nd on test data
- Participants that did well made fewer submissions!!!
- http://gregorypark.org/blog/Kaggle-Psychopathy-Postmortem/

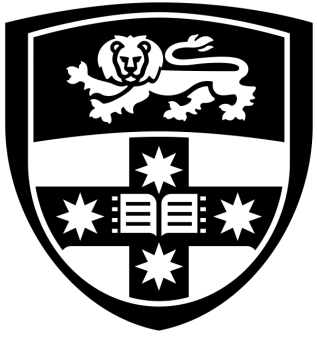
Recommendation: Use Validation Data Sparingly

- If a result seems too good to be true, it probably is!
- Use cross-validation on training data for repeated experiments
- Evaluate on validation data sparingly (semi-held-out eval)

K-Fold cross validation

Tips and Tricks

- Good Kaggle result ≠ good real-world performance
- What we really want is a deployable solution to a problem
- Tips:
  - Use a evaluation measure that reflects the real problem
  - A larger test data set reduces variance of estimated performance
  - Always keep held-out test set for final validation
  - Use cross validation on training set for development experiments
  - Use development test set sparingly
  - Include preprocessing and feature selection in training step for all folds



THE UNIVERSITY OF
SYDNEY

Building a Good Solution

* Think about the research question.

Understand the Problem

Optus: "multiple viable solutions"

- Read the documentation and follow relevant discussion ✓
- Talk to experts about problem and possible solutions ✓
- Understand the evaluation measure; evaluate in real terms ✓

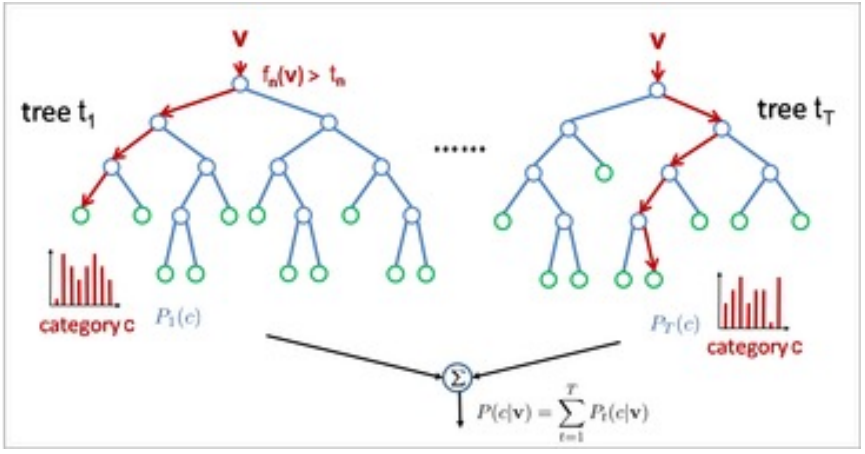
Build a Simple Model first – Evaluate – Iterate

- Start by building an end-to-end pipeline and evaluation
- Replicate published benchmarks to sanity check pipeline
- Wash, rinse, repeat:
  - Review the data and problem
  - Hypothesise next best approach in terms of elegance and impact
  - Implement and evaluate approach

Feature Engineering is Often Key

- Relates back to understanding the problem
- Design informative and discriminative features
- Understand and validate features to avoid overfitting
  - Beware if a model weights a feature more than makes sense

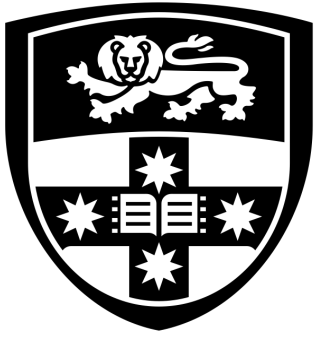
Ensembles of Predictors often do very well



http://www.iis.ee.ic.ac.uk/icvl/iccv09_tutorial.html

- Vote across many classifiers
- Random forest
  - Bootstrap many trees on samples of training data
  - Often do well on Kaggle
  - Become more biased
  - But lower variance
- Lose explainability of trees!

All of this covered in STAT5003



THE UNIVERSITY OF
SYDNEY

Communicating Results

Telling a Story

- Construct a **narrative** around the **problem**
- Briefly explain technical approach (the **solution**)
- Describe results focusing on **impact** and **caveats**

Construct a Narrative around the Problem

- It should be absolutely clear why the problem matters
- This is the “saving babies” bit and is key to the story
- How are you framing the problem in terms of (a) specific research question(s)?
- How will you validate the success of your proposed solution?

Explain Technical Approach (the Solution)

- Detail should suit audience (academic publications:↑; client reports:↓)
- Extensive technical detail is unnecessary
  - Tweaking may help find a good solution
  - But only need to convey that the approach is sound and appropriate

- …It's the story of the problem that matters!

Describe Results focusing on Impact and Caveats

- Raw numbers are good, but should be accompanied by analysis
- Are results reliable (e.g., significant, acceptable precision/variance)?
- When does it work and when doesn't it?
- What is the implication for the original problem?
- Can we deploy/operate the solution? How?
- How could we improve? What's the expected benefit and cost?

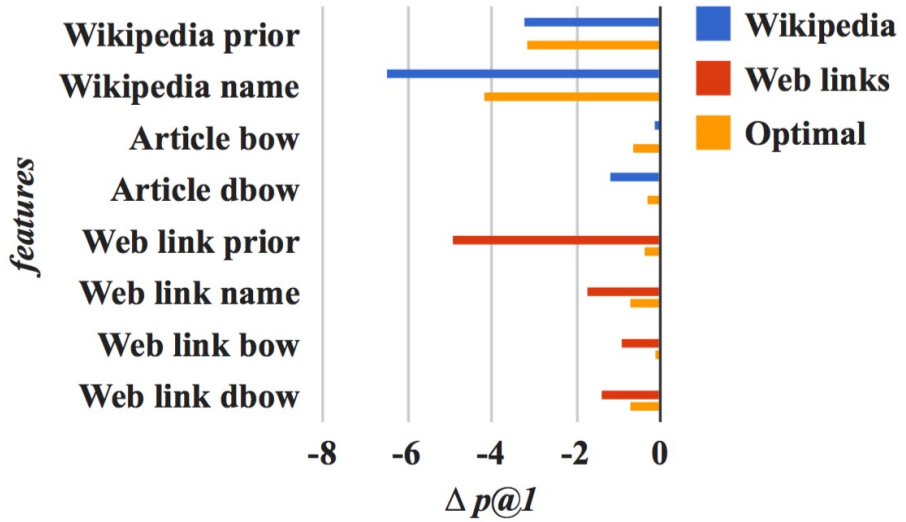
Reporting Accuracy and Reliability

- Understand the problem and the data
  - Report annotation process and agreement
  - Confusion matrices to assess less frequent categories
  - Report human upper bound as a benchmark where possible
  - http://www.mitpressjournals.org/doi/pdf/10.1162/089120102762671936
- Report simplest reasonable model as a benchmark (baseline)
- Report accuracy numbers with reliability, e.g.:
  - Pairwise significance tests to compare to benchmarks
  - Confidence intervals
  - Training versus generalisation performance

Error Analysis

- Error analysis seeks to identify systematic problems, e.g.:
  - Sample 20 false positives and 20 false negatives
  - Look at feature vectors and corresponding data
  - Group errors into categories and count
- Requires manual inspection but provides qualitative insight
- Should not be overlooked in favour of parameter tweaking
- Should be part of development cycle, not just final reporting
- Confusion matrices can also help to identify common errors

Subtractive Feature Analysis



- Assess impact of each feature by removing it
- The more performance goes down, the more critical
- If performance goes up, it's not a good feature

http://www.aclweb.org/anthology/Q15-1011

Deploying Machine Learning

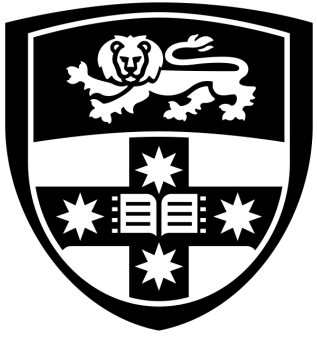
- Remember the goal is a practical and usable solution
- It does no good to solve a problem if it can’t be deployed
- Some things to keep in mind:
  - Efficiency
  - Reliability of code
  - Monitoring drift

Efficiency

- Will method work at production scale?
- Can we compute features and predictions online?

Testing and Debugging

- (Unit) testing to ensure features are consistent
- Will we be able to debug/fix?
- What if the team changes?



THE UNIVERSITY OF
SYDNEY