Document Categorization

Perceptron, Topic Detection, Sentiment Classification

Information Retrieval

Importance of information categorization

P. Jackson and I. Moulinier. 2002. *Natural Language Processing for Online Applications*

- "There is no question concerning the commercial value of being able to classify documents automatically by content. There are myriad potential applications of such a capability for corporate intranets, government departments, and Internet publishers"

Spam filtering: Another text classification task

From: "" <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for similar courses

I am 22 years old and I have already purchased 6 properties using the methods outlined in this truly INCREDIBLE ebook.

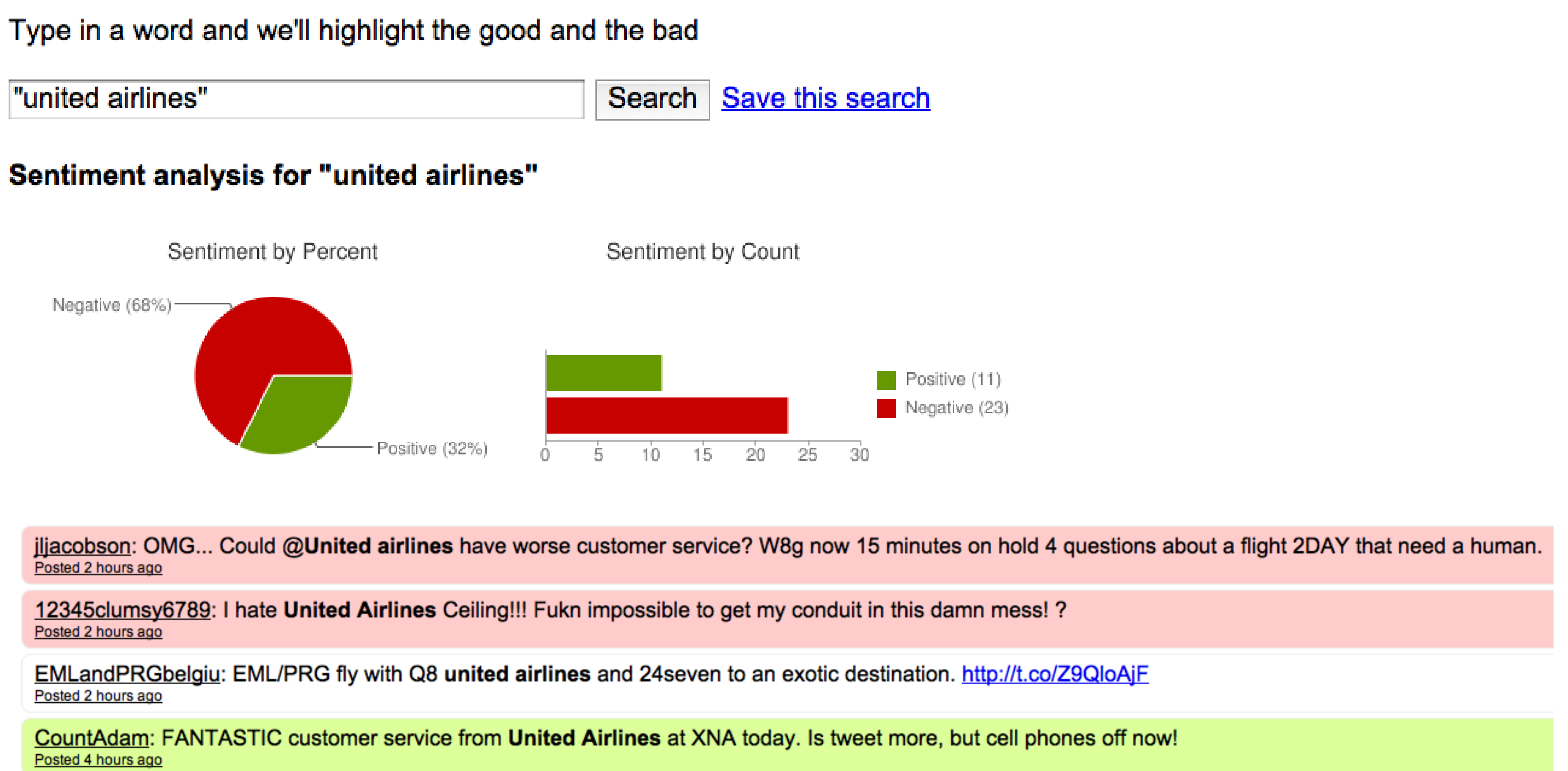
Change your life NOW !

===
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm

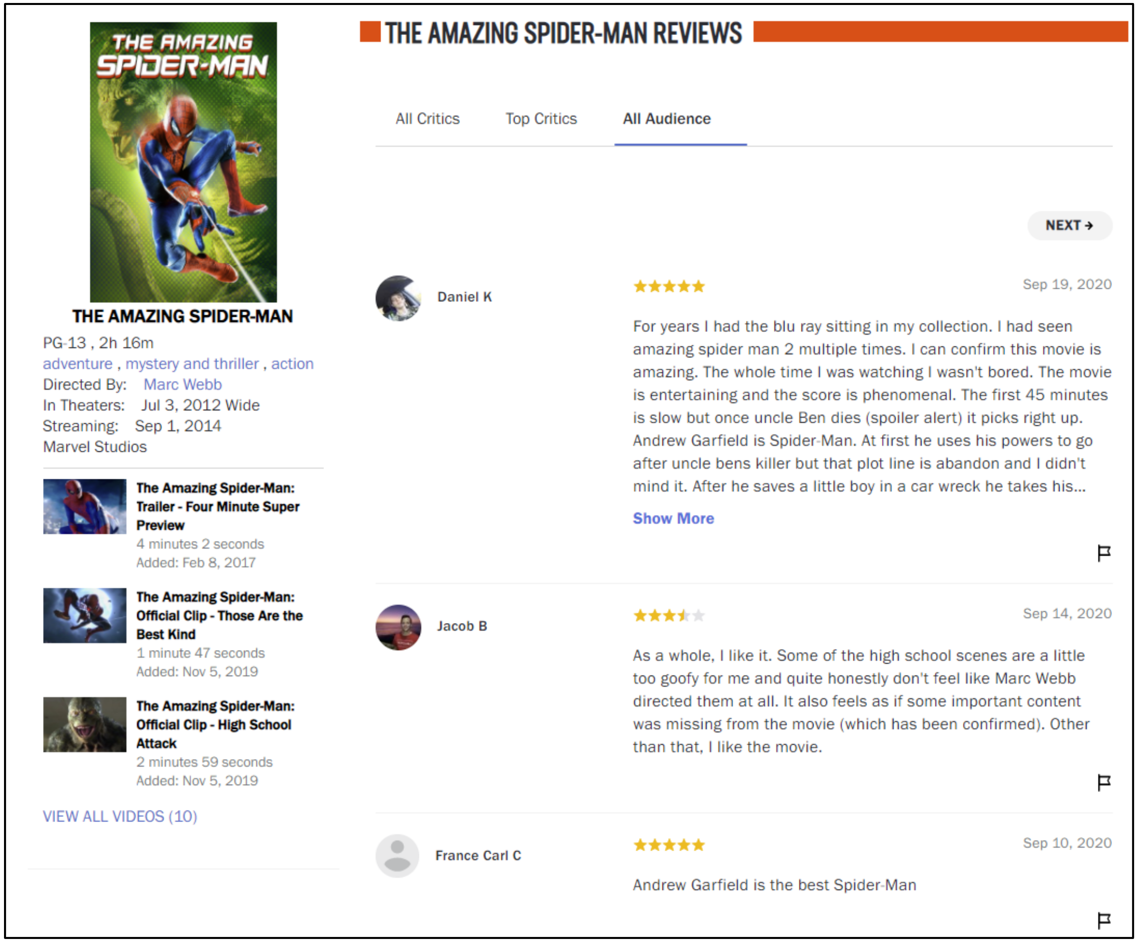
===

Target Sentiment on Twitter

- Twitter Sentiment App
- Alec Go, Richa Bhayani, Lei Huang. 2009. Twitter Sentiment Classification using Distant Supervision



Sentiment Classification in Movie Reviews



Sentiment Classification in Movie Reviews

- Polarity detection:
  - Is an IMDB movie review positive or negative?

- Data: Polarity Data 2.0:
  - http://www.cs.cornell.edu/people/pabo/movie-review-data

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. EMNLP-2002, 79—86.

Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. ACL, 271-278

IMDB data in the Pang and Lee database

✓

when _star wars_ came out some twenty years ago , the image of traveling throughout the stars has become a commonplace image . […]

when han solo goes light speed , the stars change to bright lines , going towards the viewer in lines that converge at an invisible point .

cool .

october sky offers a much simpler image–that of a single white dot , traveling horizontally across the night sky .   [. . .]

✗

“ snake eyes ” is the most aggravating kind of movie : the kind that shows so much potential then becomes unbelievably disappointing .

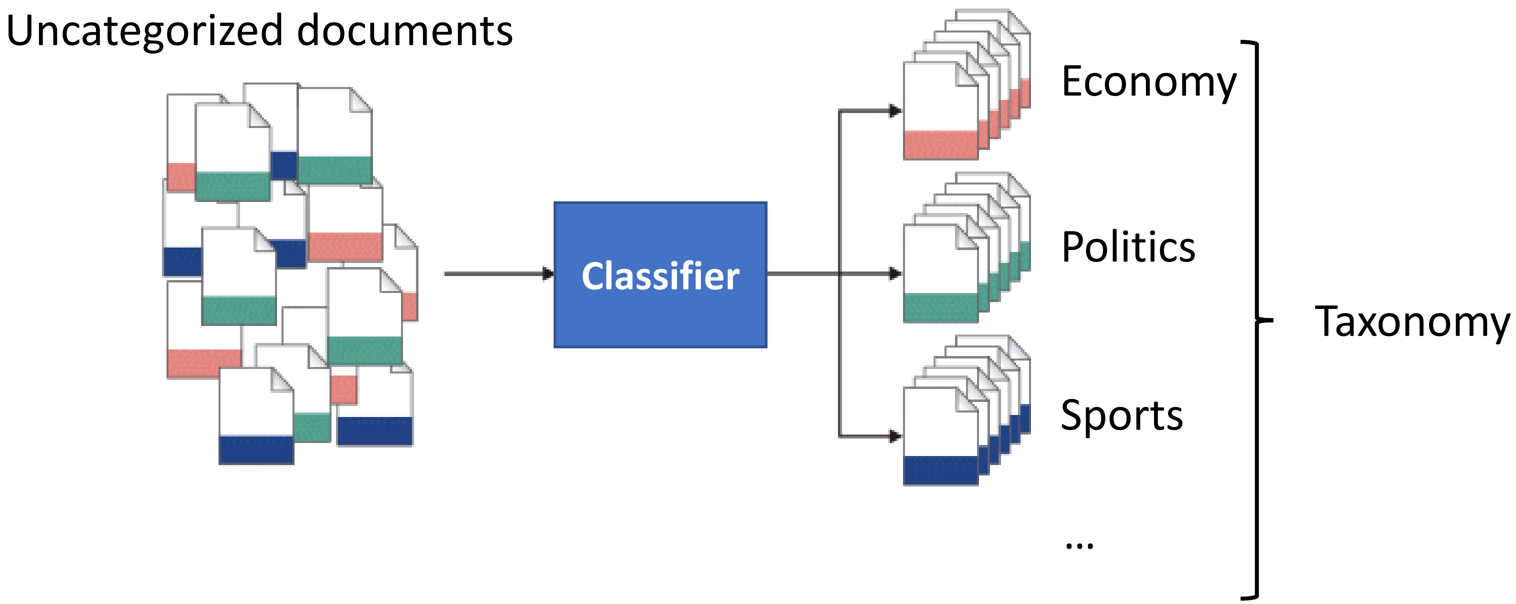
it’s not just because this is a brian depalma film , and since he’s a great director and one who’s films are always greeted with at least some fanfare .

and it’s not even because this was a film starring nicolas cage and since he gives a brauvara performance , this film is hardly worth his talents .

Baseline Algorithm

- Tokenization
- Feature Extraction
- Classification using different classifiers
  - Naïve Bayes
  - MaxEnt
  - SVM

Taxonomy



Uncategorized documents
Classifier
Economy
Politics
Sports
...
Taxonomy

How to define a documents taxonomy?

- Domain specific terminologies are curated by domain experts and are designed with specific tasks and workflows in mind.

- In the medical domain, the SNOMED-CT is intended to describe medical conditions, procedures, admin, etc.
  - http://browser.ihtsdotools.org/

- In the computer science domain the ACM Computing Classification Scheme is widely used to classify published articles.
  - https://dl.acm.org/ccs/ccs.cfm

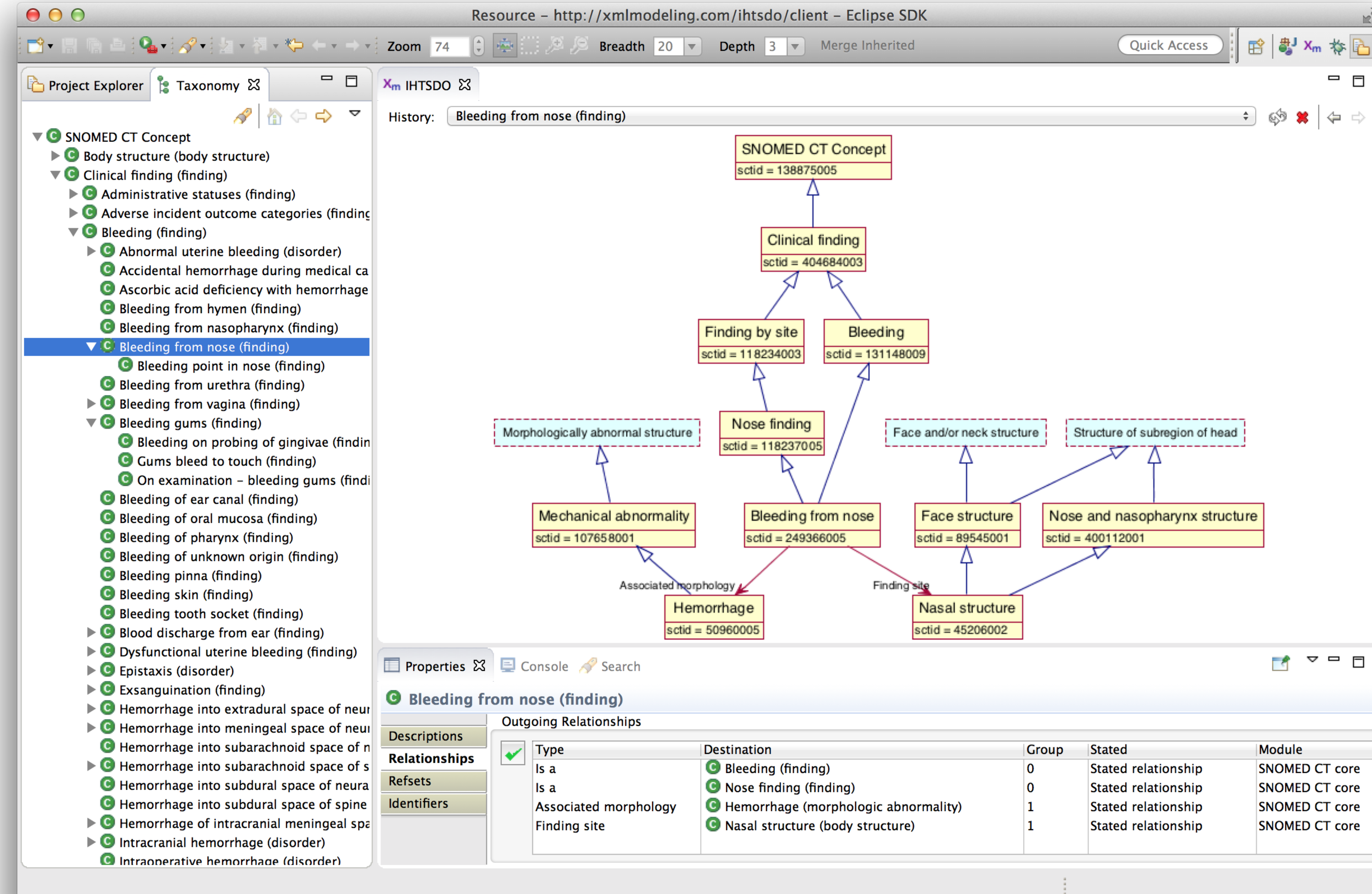
Wikipedia as a database

- Wikipedia contains large amounts of information largely unstructured but structured as a taxonomy.

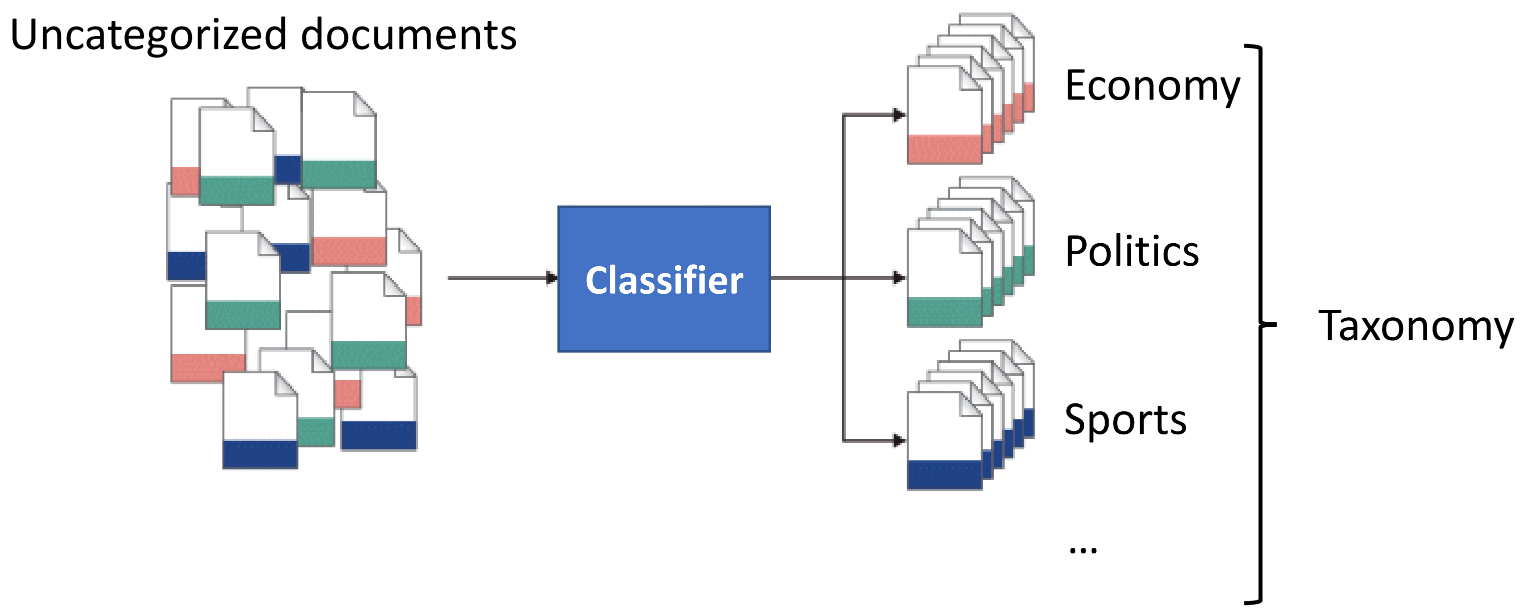
- **DBPedia** aims to create a rigorous database out of Wikipedia.

- A key application is to link data to Wikipedia entries.

https://en.wikipedia.org/wiki/Portal:Contents


Resource – http://xmlmodeling.com/ihtsdo/client – Eclipse SDK
Zoom 74
Breadth 20
Depth 3
Merge Inherited
Quick Access
Project Explorer
Taxonomy
SNOMED CT Concept
Body structure (body structure)
Clinical finding (finding)
Administrative statuses (finding)
Adverse incident outcome categories (finding
Bleeding (finding)
Abnormal uterine bleeding (disorder)
Accidental hemorrhage during medical ca
Ascorbic acid deficiency with hemorrhage
Bleeding from hymen (finding)
Bleeding from nasopharynx (finding)
Bleeding from nose (finding)
Bleeding point in nose (finding)
Bleeding from urethra (finding)
Bleeding from vagina (finding)
Bleeding gums (finding)
Bleeding on probing of gingivae (findin
Gums bleed to touch (finding)
On examination – bleeding gums (findi
Bleeding of ear canal (finding)
Bleeding of oral mucosa (finding)
Bleeding of pharynx (finding)
Bleeding of unknown origin (finding)
Bleeding pinna (finding)
Bleeding skin (finding)
Bleeding tooth socket (finding)
Blood discharge from ear (finding)
Dysfunctional uterine bleeding (finding)
Epistaxis (disorder)
Exsanguination (finding)
Hemorrhage into extradural space of neu
Hemorrhage into meningeal space of neu
Hemorrhage into subarachnoid space of n
Hemorrhage into subarachnoid space of s
Hemorrhage into subdural space of neura
Hemorrhage into subdural space of spine
Hemorrhage of intracranial meningeal spa
Intracranial hemorrhage (disorder)
Intraoperative hemorrhage (disorder)
IHTSDO
History: Bleeding from nose (finding)
SNOMED CT Concept
sctid = 138875005
Clinical finding
sctid = 404684003
Finding by site
sctid = 118234003
Bleeding
sctid = 131148009
Morphologically abnormal structure
Nose finding
sctid = 118237005
Face and/or neck structure
Structure of subregion of head
Mechanical abnormality
sctid = 107658001
Bleeding from nose
sctid = 249366005
Face structure
sctid = 89545001
Nose and nasopharynx structure
sctid = 400112001
Associated morphology
Finding site
Hemorrhage
sctid = 50960005
Nasal structure
sctid = 45206002
Properties
Console
Search
Bleeding from nose (finding)
Outgoing Relationships
Descriptions
Relationships
Refsets
Identifiers
Type | Destination | Group | Stated | Module
Is a | Bleeding (finding) | 0 | Stated relationship | SNOMED CT core
Is a | Nose finding (finding) | 0 | Stated relationship | SNOMED CT core
Associated morphology | Hemorrhage (morphologic abnormality) | 1 | Stated relationship | SNOMED CT core
Finding site | Nasal structure (body structure) | 1 | Stated relationship | SNOMED CT core

Document representations



Documents representation

- Once tokenization is done, a document is represented as a vector of tokens

$$d = (t_1, \dots, t_V),$$

- where each dimension of the document indicates the weight of that particular token in the document:
  - boolean; frequency (TF-IDF); probability (LM); dictionary based.

- Structured documents may be divided into multiple segmented

$$d = [d_{sectionA}, d_{sectionB}, d_{sectionC}]$$

$$d_{sectionA} = (t_1, \dots, t_V)$$
$$d_{sectionB} = (t_1, \dots, t_V)$$
$$d_{sectionC} = (t_1, \dots, t_V)$$

Word representations

- Binary seems to work better than full word counts

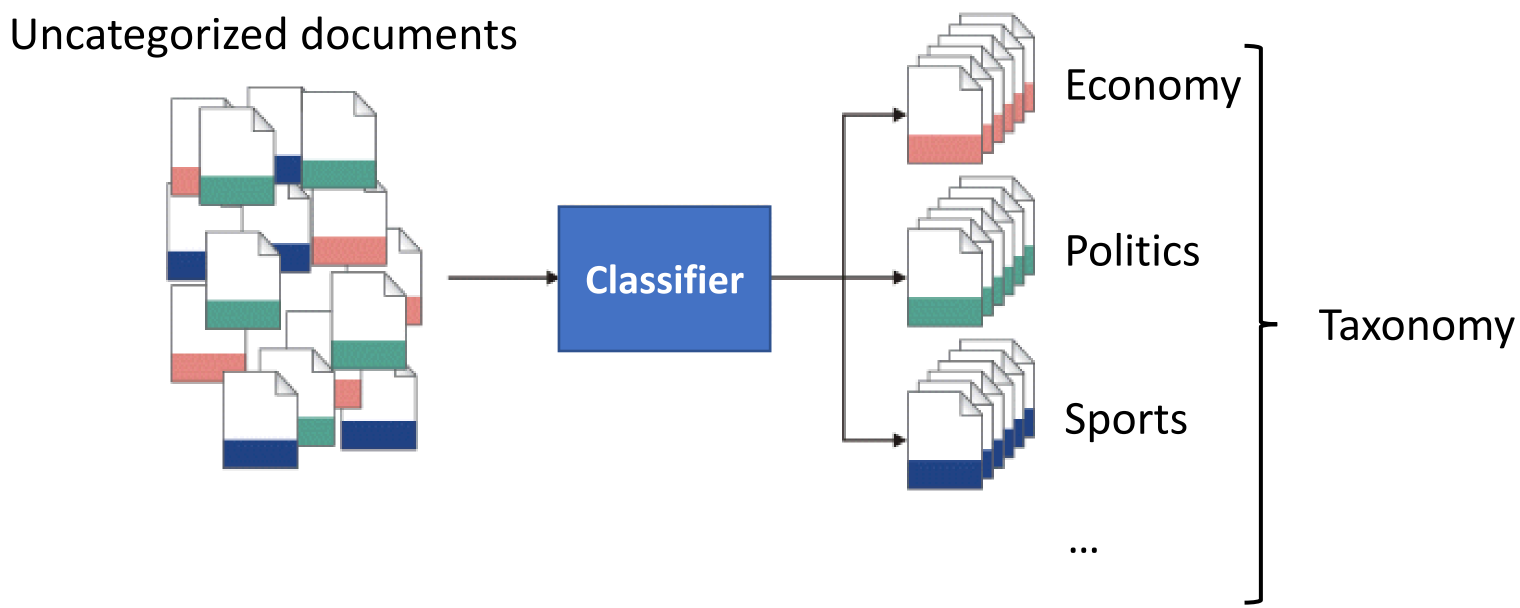
- **Sentiment lexicons**
  - e.g. SentiWordNet https://github.com/aesuli/SentiWordNet
  - All WordNet synsets automatically annotated for degrees of positivity, negativity, and neutrality/objectiveness

```python
breakdown = swn.senti_synset('amazing.s.02')
print(breakdown)
```

```
<amazing.s.02: PosScore=0.875 NegScore=0.125>
```

- **Affective lexicons**
  - joy–sadness
  - anger–fear
  - trust–disgust
  - anticipation–surprise

The classifier



Problem formulation

- Given:
  - A description of an instance, $d \in X$
    - X is the *instance language* or *instance space*.
      - Issue: how to represent text documents.
      - Usually some type of high-dimensional space
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
- Determine:
  - The category of d: $\gamma(d) \in C$, where $\gamma(d)$ is a *classification function* whose domain is X and whose range is C.
    - We want to know how to build classification functions ("classifiers").

Supervised Document Categorization

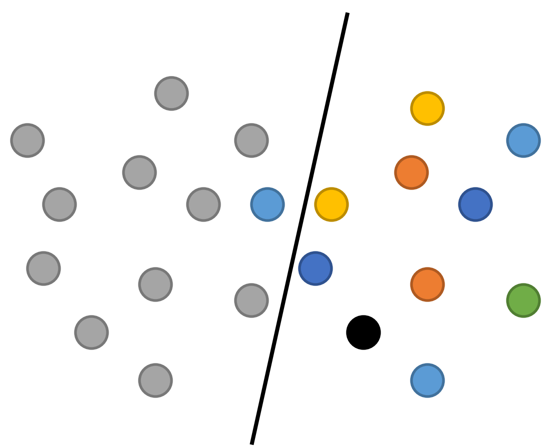
- Given:
  - A description of an instance, $d \in X$
    - X is the *instance language* or *instance space*.
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set D of labeled documents with each labeled document $\langle d,c \rangle \in X \times C$
- Determine:
  - A learning method or algorithm which will enable us to learn a classifier $\gamma: X \rightarrow C$
  - For a test document d, we assign it the class $\gamma(d) \in C$

Classification task

- For new unseen documents, we wish to classify documents with one of the known classes.
- New documents are represented in some feature space and then a machine learning algorithm classifies the new documents.

Category C_1

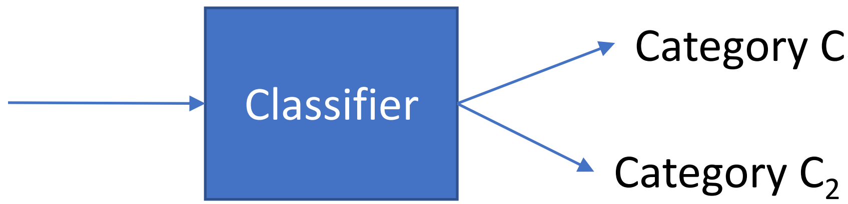


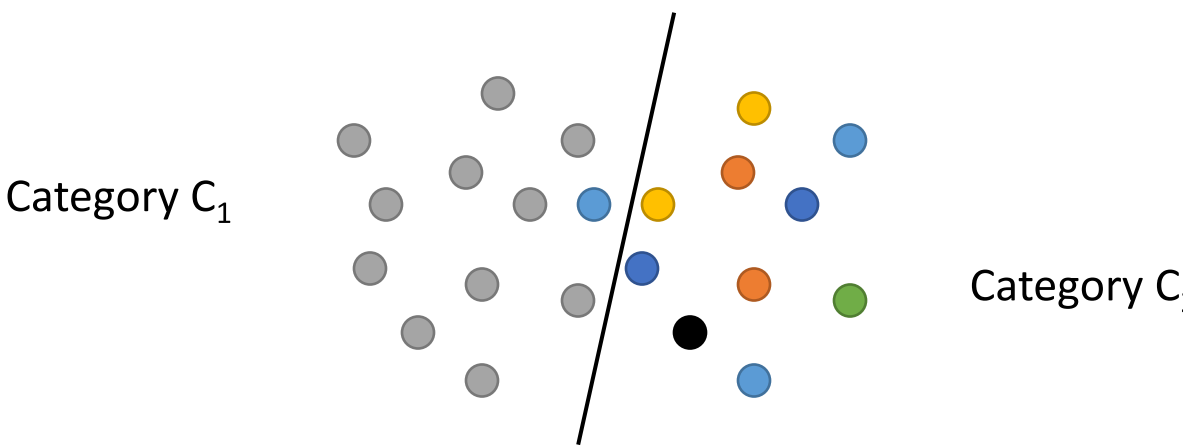
Category C_2

Classifier

Input sample *can be* the document word counts

$d_i = (t_1^i, t_2^i, \dots, t_V^i)$





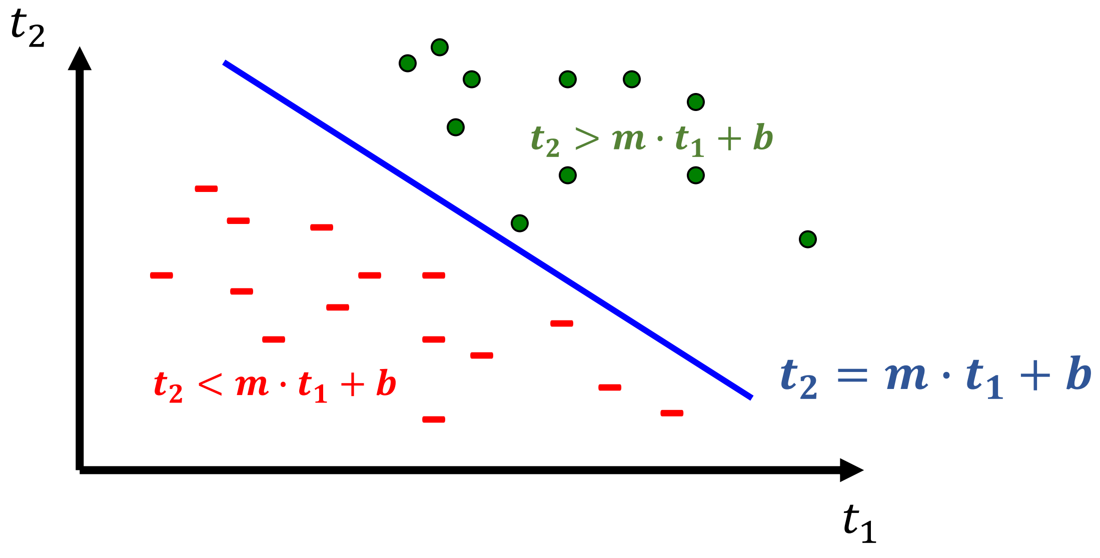
Classifier

Perceptron

- Every document d_i has its corresponding label $y_i = \{+1, -1\} = \{C_1, C_2\}$

- **The perceptron performs a binary prediction $\hat{y}$ of the true label y based on the observed data $d = \left(t_1, t_2, \dots, t_n\right)$:**

$$\hat{y} = f(d) = \begin{cases} +1 & , if\ t_2 \geq m \cdot t_1 + b \\ -1 & , if\ t_2 < m \cdot t_1 + b \end{cases}$$

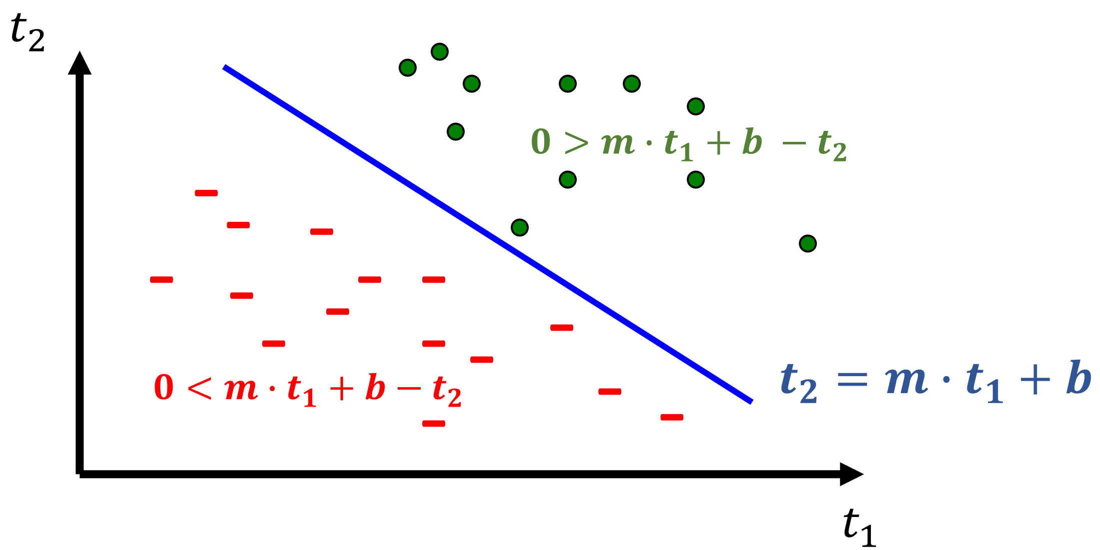


Perceptron

- Every document d_i has its corresponding label $y_i = \{+1, -1\} = \{C_1, C_2\}$

- **The perceptron performs a binary prediction $\hat{y}$ of the true label y based on the observed data $d = (t_1, t_2, \dots, t_V)$:**

$$\hat{y} = f(d) = \begin{cases} +1 & , if\ t_2 \geq m \cdot t_1 + b \\ -1 & , if\ t_2 < m \cdot t_1 + b \end{cases} = \begin{cases} +1 & , if\ 0 \geq m \cdot t_1 + b - t_2 \\ -1 & , if\ 0 < m \cdot t_1 + b - t_2 \end{cases}$$

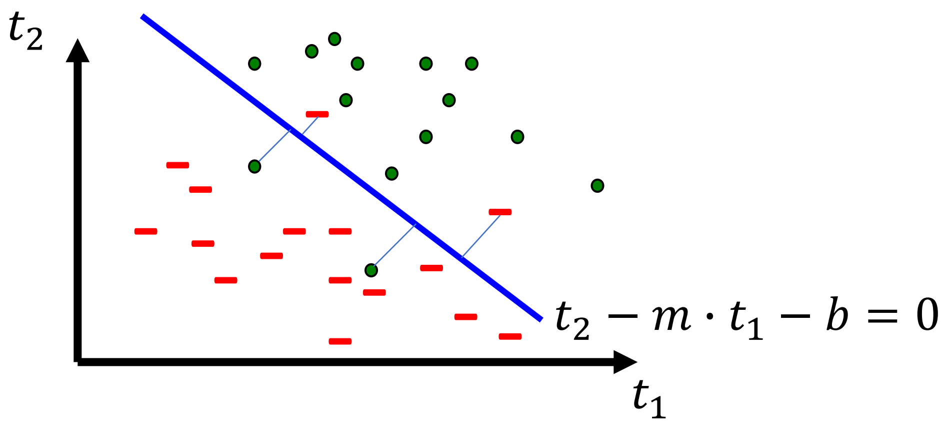


Model error

- The Mean Square Error (MSE) measures the error between the true labels and the predicted labels

$$MSE = \frac{1}{N}\sum_i^N (error_i)^2$$

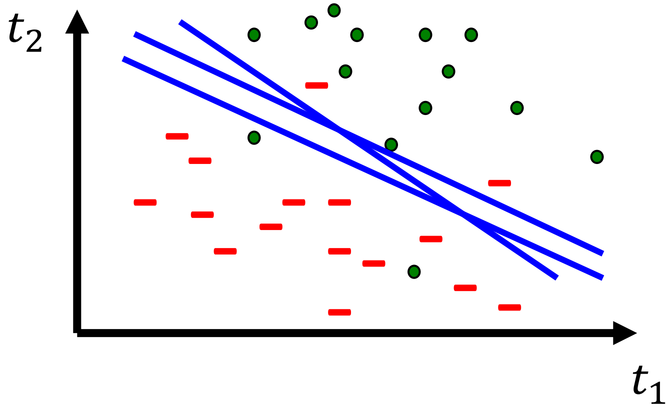
$$error_i = y_i - \widehat{y_i} = true\_label_i - predictedLabel_i$$

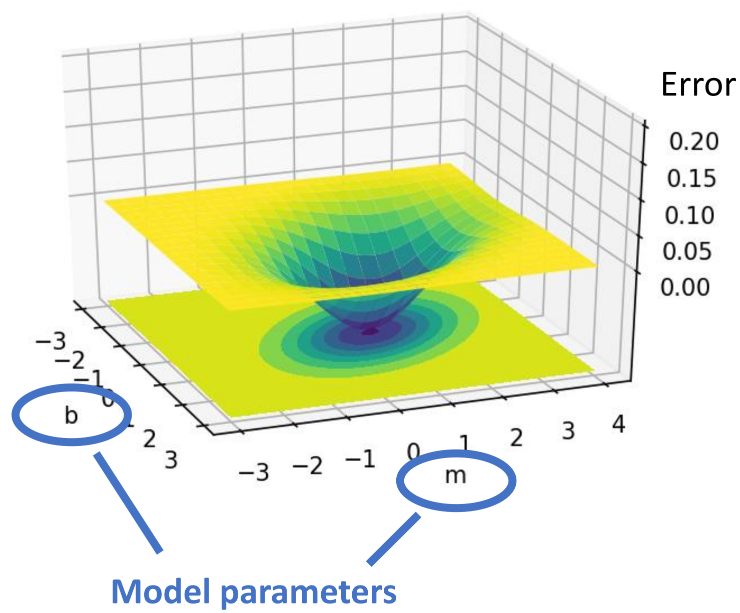


Minimizing the error

$$MeanSquareError = \frac{1}{TotalSamples} \sum_{i}^{TotalSamples} (label_i - predictedLabel_i)^2$$

$$t_2 = t_1 \cdot \boldsymbol{m} + \boldsymbol{b}$$





Learning to minimize the model error

- Initialize the model with random weights
- Compute the model predictions
- Compute the error of each prediction
- Update the model with the samples incorrectly classified.

True label	Predicted label	Error	Update
-1	-1	0	0
-1	+1	-1	-1*x
+1	-1	+1	+1*x
+1	+1	0	0

Learning algorithm

```
model = LogisticRegression.fit(data, labels)
```

```
[ ]: b=0
     m=0
     model = [m,b]

     max_iters = 30
     mean_square_error = []
     for iter in range(0,max_iters):

         # Compute the model predictions
         predicted_labels = ((observations_x2 - m*observations_x1 - b ) >= 0)*2-1

         # Compute the model error
         error_of_all_samples = (true_labels-predicted_labels)/2

         # Update the model parameters
         update_m = np.mean(error_of_all_samples*observations_x1)
         update_b = np.mean(error_of_all_samples)

         m = m - update_m*0.1
         b = b - update_b*0.1
```

Model predictions

$$\hat{y} = f(d) = \begin{cases} +1 & , if\ t - m \cdot t_1 - b \geq 0 \\ -1 & , if\ t_2 - m \cdot t_1 - b < 0 \end{cases}$$

Model error

$$error = (y - \hat{y})/2 = \begin{cases} +1 \\ 0 \\ -1 \end{cases}$$

Model parameter update

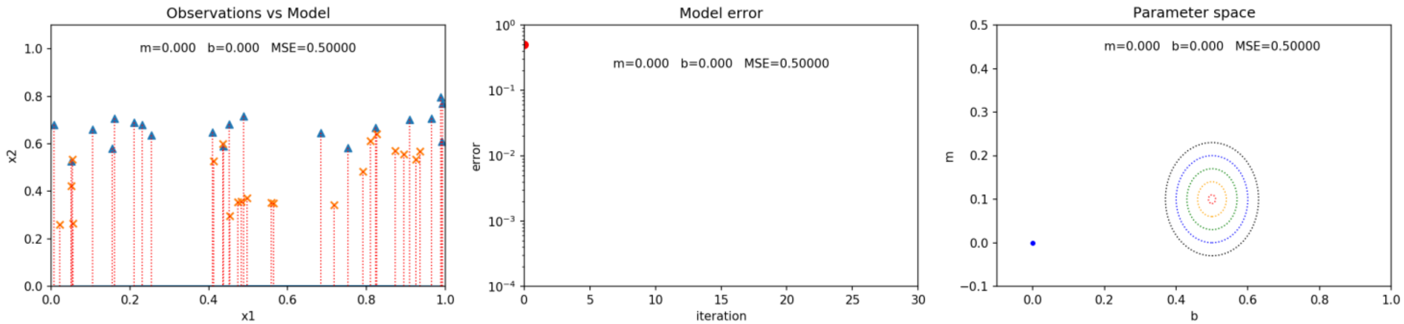
$$update_m = error \cdot t_1$$

$$m = m - update_m \cdot learning_{rate}$$

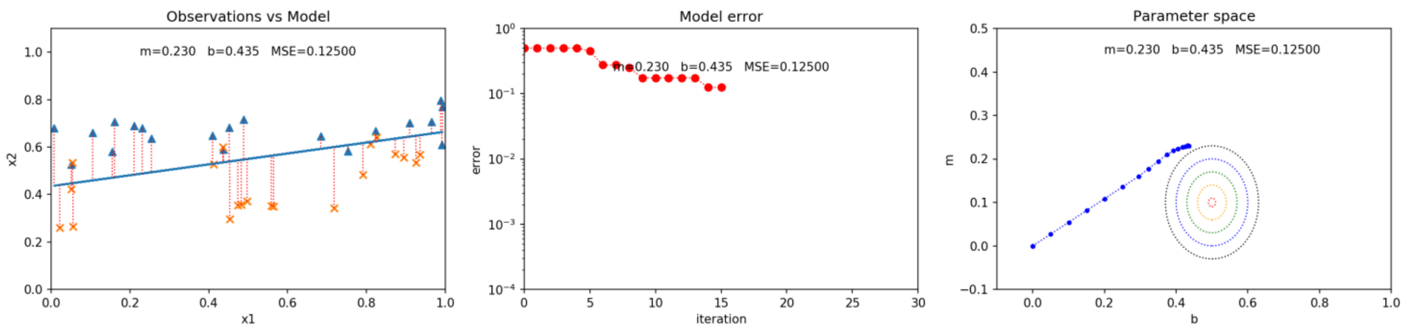
Perceptron learning example

Model predictions | Model error | Model parameter update

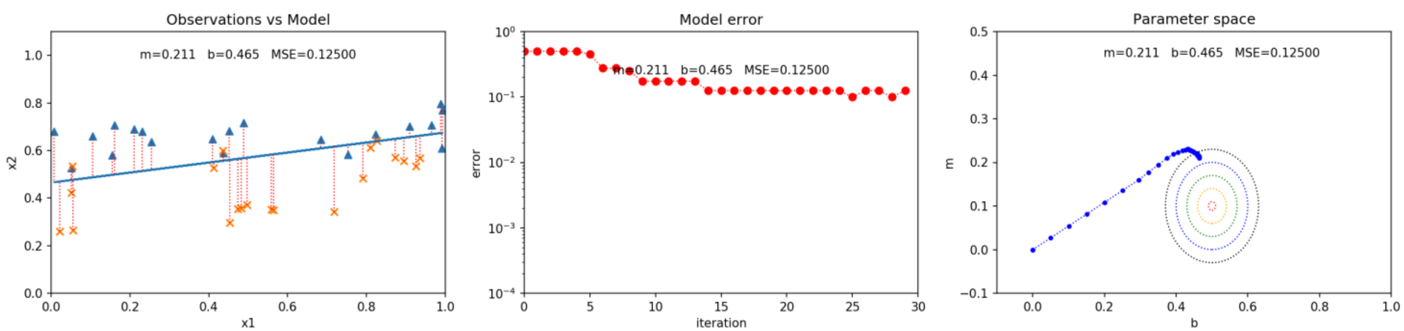
iter = 1



iter = 15



iter = 36

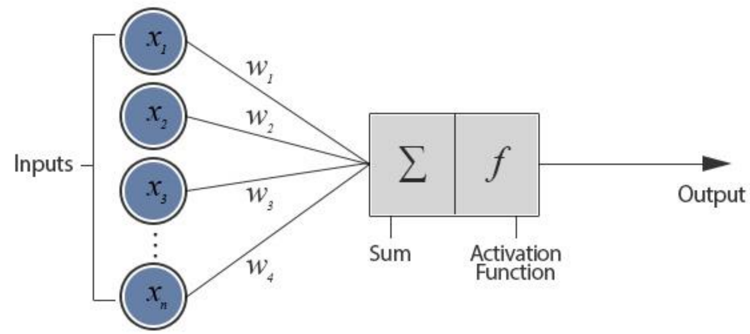


Perceptron: general formulation

- **Binary classification:**

$$z = w_0 + w_1 t_1 + \ \dots + w_V t_V$$

$$\hat{y} = \sigma(z) = \begin{cases} +1 & , if\ z \geq 0 \\ -1 & , if\ z < 0 \end{cases}$$



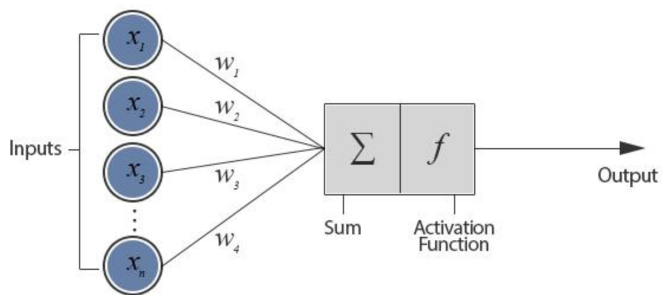
- **Input:** Vectors $d^{(j)}$ and labels $y^{(j)}$
  - $d^{(j)}$ are V dimensional real valued vectors, where $\|d\|_2 = 1$

- **Goal:** Find vector $w = (w_0, w_1, w_2, \dots, w_V)$
  - Each w_i is a real number

The sigmoid activation function

$$\begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ \\ t_n \end{bmatrix}$$

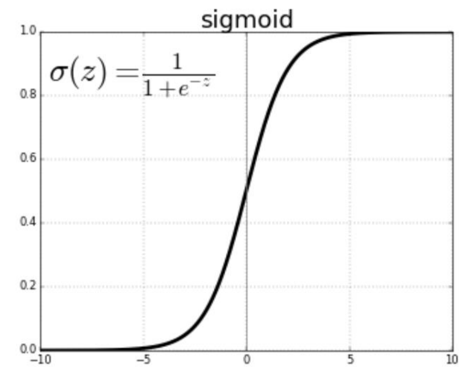
Document tokens



$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \\ w_n \end{bmatrix}$$

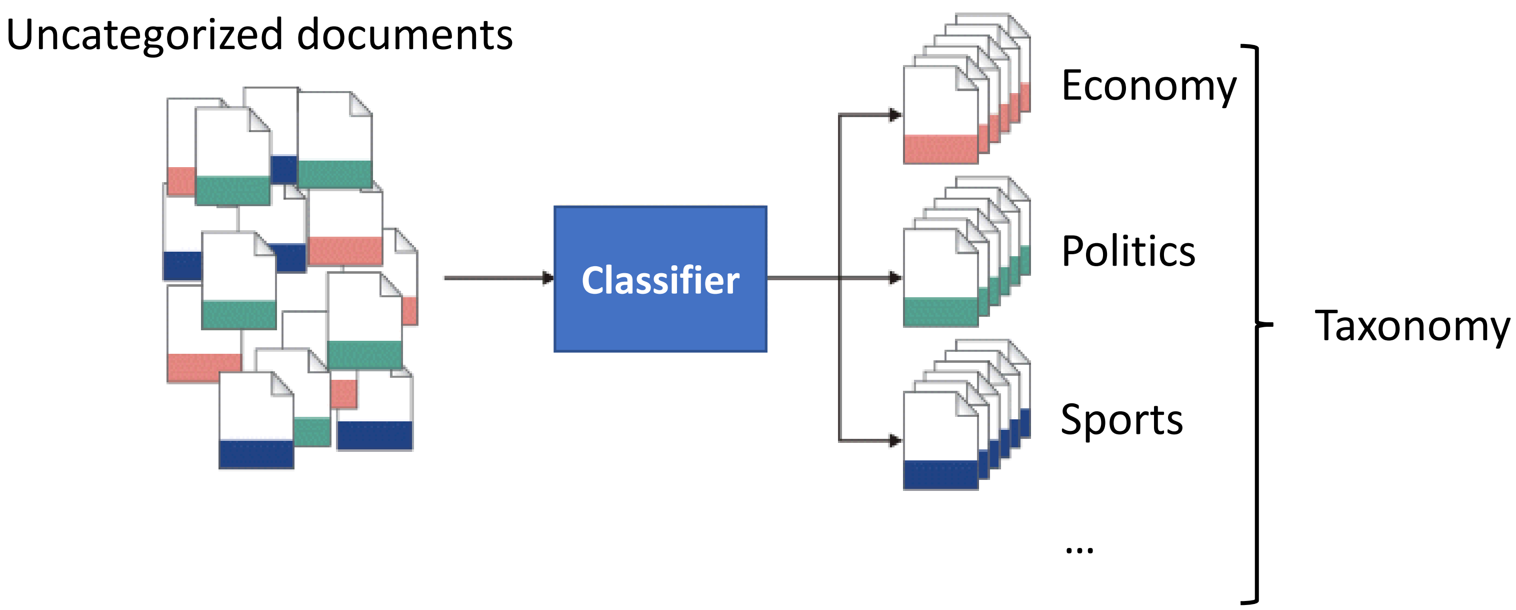
Token model weights

$\sigma(z)$



$$z = w_0 + w_1 t_1 + \ \ldots + w_n t_n$$

Real-world model training



Real-world model training

- Robustly training a model for Web data is a complex task.
- In most of the cases, we will use pre-trained models.
- These models were trained on large-scale data.
- These pre-trained models are robust and reliable.

Which and how many categories are detectable?

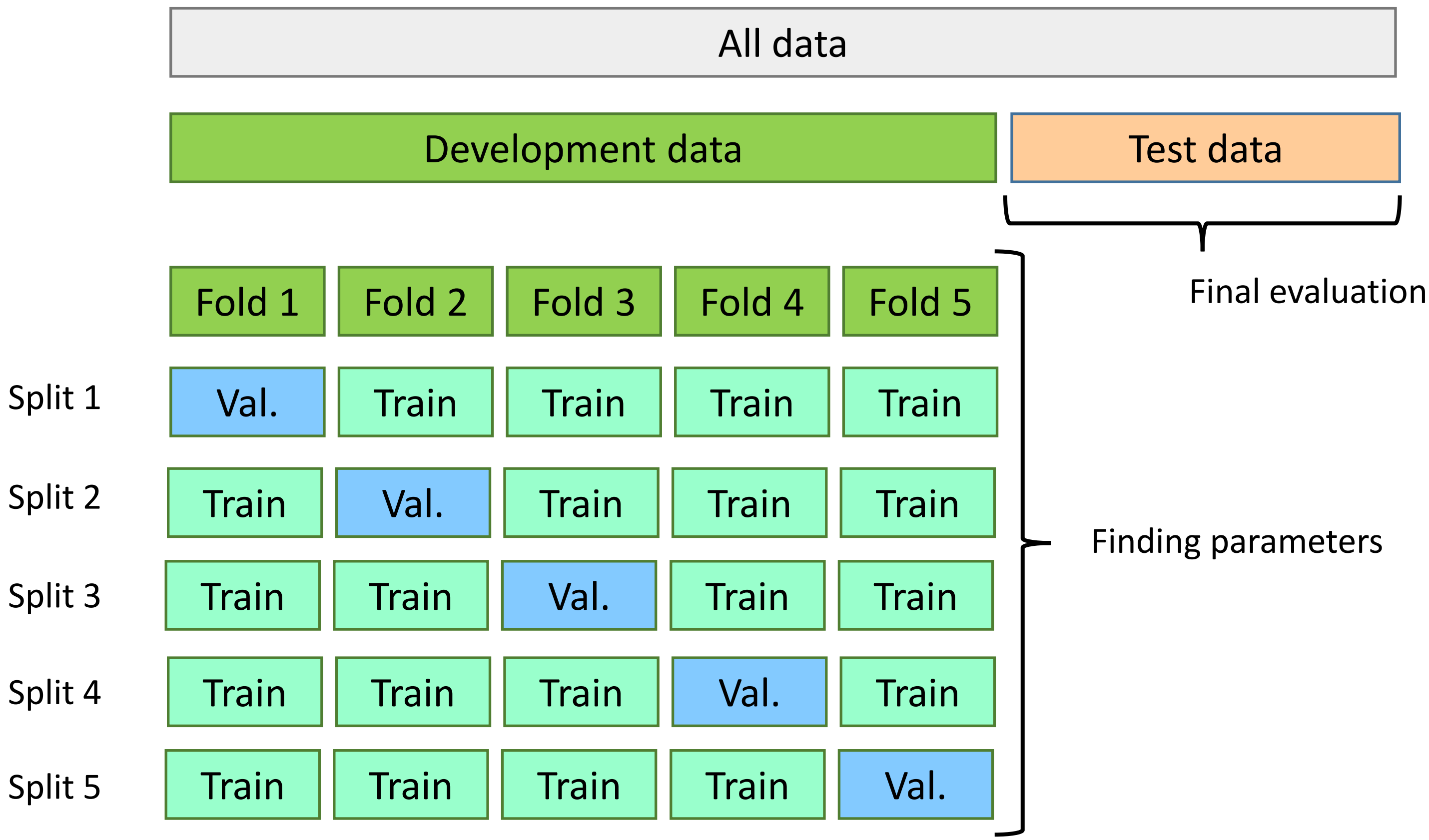
- An important question to ask is which and how many items of the taxonomy are detectable in data?
- A few (well separated ones)? -> Easy!
- A zillion closely related ones? -> Not so easy…
  - Think: Yahoo! Directory, Library of Congress classification, legal applications
  - Quickly gets difficult!
    - Classifier combination is always a useful technique
      - Voting, bagging, or boosting multiple classifiers
    - Much literature on hierarchical classification
      - Definitely helps for scalability, even if not in accuracy
    - May need a hybrid automatic/manual solution

Kappa statistics over a small set of annotations

Taxonomies and classification

- In practice, only a few elements of the taxonomy should be used as classes for classification
  - Only the ones offering a stable document class representation.
- The ultimate goal is to link information to an entry on a taxonomy capturing the target domain.
- Ultimately more complete domain representation should be used, e.g. an ontology.

Cross-Validation with held-out test data



Rich data volumes scenarios

Scarce data scenarios

Cross-Validation with limited data

- Break up data into 5 folds
- For each fold
  - Choose the fold as a temporary test set
  - Train on 4 folds, compute performance on the test fold
- Report the average performance of the 5 runs

	All data				
Split 1	Test	Val	Train	Train	Train
Split 2	Train	Test	Val	Train	Train
Split 3	Train	Train	Test	Val	Train
Split 4	Train	Train	Train	Test	Val
Split 5	Val	Train	Train	Train	Test

Per-class evaluation measures

		Ground-truth	
		True	False
Method	True	True positive	False positive
	False	False negative	True negative

- **Recall**: Fraction of docs in class i classified correctly:

$$Recall = \frac{truePos}{truePos + falseNeg}$$

- **Precision**: Fraction of docs assigned class i that are actually about class i:

$$Precision = \frac{truePos}{truePos + falsePos}$$

- **Accuracy**: Fraction of docs classified correctly:

$$Accuracy = \frac{truePos + trueNeg}{truePos + falsePos + trueNeg + falseNeg}$$

* abragência, precisão e exatidão.

Micro- vs. Macro-Averaging

- If we have more than one class, how do we combine multiple performance measures into one quantity?

- **Macroaveraging**: Compute performance for each class, then average.

- **Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.

Micro- vs. Macro-Averaging: Example

Class 1

	Truth yes	Truth no
Classifier yes	10	10
Classifier no	10	970

Class 2

	Truth yes	Truth no
Classifier yes	90	10
Classifier no	10	890

Micro Ave. Table

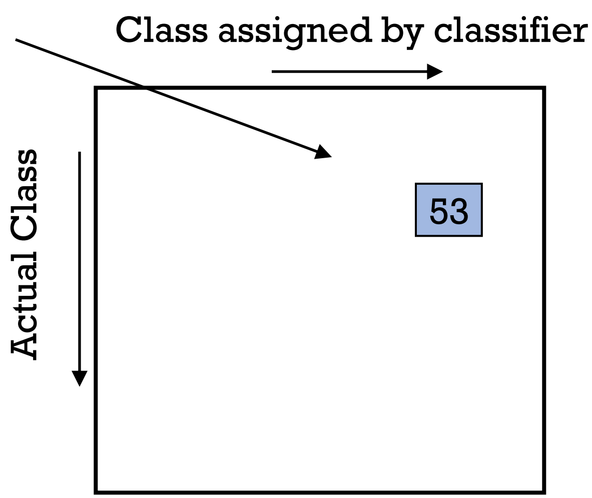
	Truth yes	Truth no
Classifier yes	100	20
Classifier no	20	1860

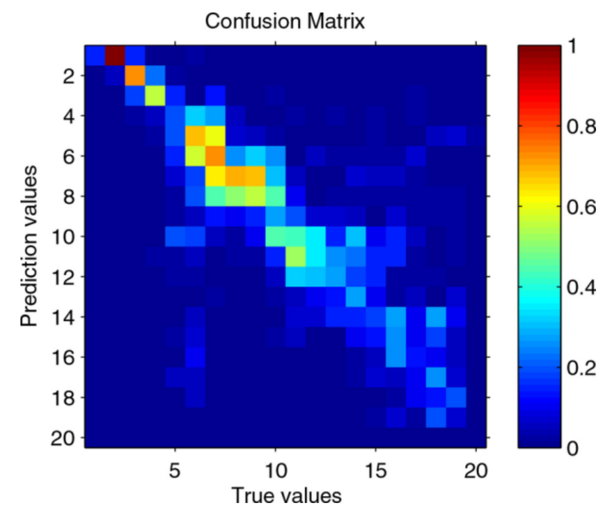
- Macroaveraged precision: (10/(10+10) + 90/100) (0.5 + 0.9)/2 = 0.7
- Microaveraged precision: 100/(100+20) = .83

- Microaveraged score is dominated by score on common classes

Good practice: Make a confusion matrix

- This (i, j) entry means 53 of the docs actually in class i were put in class j by the classifier.





- In a perfect classification, only the diagonal has non-zero entries
- Look at common confusions and how they might be addressed

Success measure vs Algorithm understanding

- The best way of measuring success is in the real-world scenario!
  - All metrics are in a fact a proxy for the real-world setting;
  - A/B testing is **the accurate** way of measuring a method's success.

- Metrics are supposed to help the data scientist in
  - *predicting* an algorithm success;
  - understanding the problem;
  - detecting flaws in the approach;
  - understanding the shortcomings of the algorithm;
  - decomposing the problem into orthogonal sub-problems;
  - the design of algorithmic improvements.

Summary

- Document topic categorization
- Perceptron and sigmoid function
- Model training
- Cross validation

Section 5.1, 5.2