Text Representation and Classification

Ayoub Bagheri & Qixiang Fang

Lecture's Plan

- How to represent a document?
- What are vector space and bag-of-words models?

- How to classify text data?
- How to evaluate a classifier?

Text Classification

Text classification

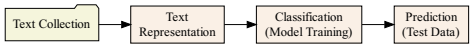
- Supervised learning: Learning a function that maps an input to an output based on example input-output pairs.
  - infer a function from labeled training data
  - use the inferred function to label new instances
- Human experts annotate a set of text data
  - Training set

Document	Class
Email1	Not spam
Email2	Not spam
Email3	Spam
...	...

Text classification?

- Which problem is not (or less likely to be) a text classification task?
  - Author's gender detection from text
  - Finding about the smoking conditions (yes/no) of patients from clinical letters
  - Grouping similar news articles
  - Classifying reviews into positive and negative sentiments

Pipeline



Text Collection
Text
Representation
Classification
(Model Training)
Prediction
(Test Data)

Text Representation

How to represent a document

- Represent by a string?
  - No semantic meaning
- Represent by a list of sentences?
  - Sentence is just like a short document (recursive definition)

Bag of Words (BOW)

- Words (terms) and weights as the basis for vector representations of text
  - Doc1: Text mining is to identify useful information.
  - Doc2: Useful information is mined from text.
  - Doc3: Apple is delicious.

	text	information	identify	mining	mined	is	useful	to	from	apple	delicious
Doc1	1	1	1	1	0	1	1	1	0	0	0
Doc2	1	1	0	0	1	1	1	0	1	0	0
Doc3	0	0	0	0	0	1	0	0	0	1	1

BOW weights: Binary

- Binary
  - with 1 indicating that a term occurred in the document, and 0 indicating that it did not

BOW weights: Raw Term frequency

- Idea: a term is more important if it occurs more frequently in a document
- use the raw frequency count of term t in doc d

BOW weights: TF-IDF

- Idea: a term is more discriminative if it occurs a lot but only in fewer documents.
- TF-IDF (term frequency–inverse document frequency) weight:

$$w_{d,t} = TF_{d,t} \cdot IDF_t$$

Let $n_{d,t}$ denote the number of times term t appears in document d. The relative frequency of t in d is:

$$TF_{d,t} = \frac{n_{d,t}}{\sum_i n_{d,i}}$$

Let N denote the number of documents annd N_t denote the number of documents containing term t.

$$IDF_t = log(\frac{N}{N_t})$$

Vector space model

- A vector space is a collection of vectors
- A vector is an ordered finite list of numbers.
- Represent documents by concept vectors
  - Each concept defines one dimension
  - A large number of concepts define a high-dimensional space
  - Element of vector corresponds to concept weight
  - The process of converting text into numbers is called Vectorization
- Distance between the vectors in this concept space
  - Relationship among documents

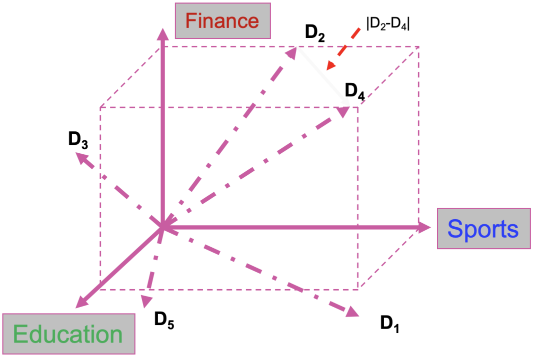
Vector space model

- Terms are generic features that can be extracted from text
- Typically, terms are single words, keywords, n-grams, or phrases
- Documents are represented as vectors of terms
- Each dimension (concept) corresponds to a separate term

$$d = (w_1, ..., w_n)$$

An illustration of VS model

- All documents are projected into this concept space

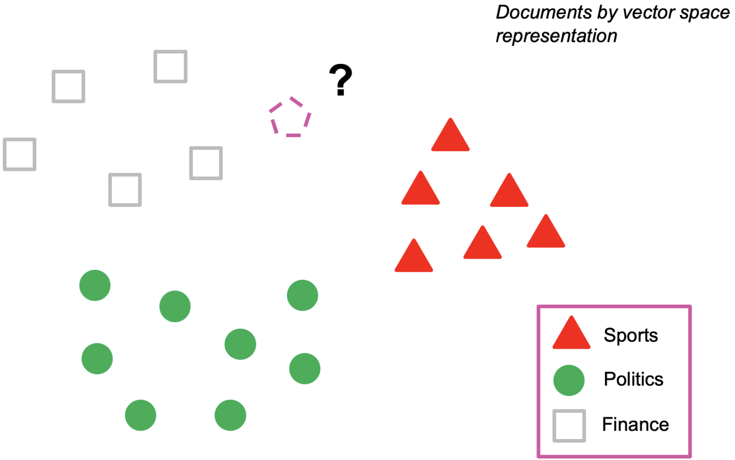


Vector space model

- Bag of Words, a Vector Space Model where:
  - Terms: words (more generally we may use n-grams, etc.)
  - Weights: number of occurrences of the terms in the document
- Topics (later)
- Word Embeddings (later)

Classification Algorithms

How to classify this document?



Text Classification: definition

- Input:
  - A training set of m manually-labeled documents $(d_1, c_1), \cdots, (d_m, c_m)$
  - A fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
- Output:
  - A learned classifier $y : d \to c$

Hand-coded rules

- Rules based on combinations of words or other features
- Rules carefully refined by expert
- But building and maintaining these rules is expensive
- Data/Domain specifics
- Not recommended!

Supervised Machine Learning

- Nearest centroid
- K-nearest neighbors
- Naïve Bayes
- Decision tree
- Random forest
- Support vector machines

More:

- Logistic regression
- Neural networks

Rocchio Classifier (Nearest Centroid)

Each class is represented by its centroid, with test samples classified to the class with the nearest centroid. Using a training set of documents, the Rocchio algorithm builds a prototype vector, centroid, for each class. This prototype is an average vector over the training documents' vectors that belong to a certain class.

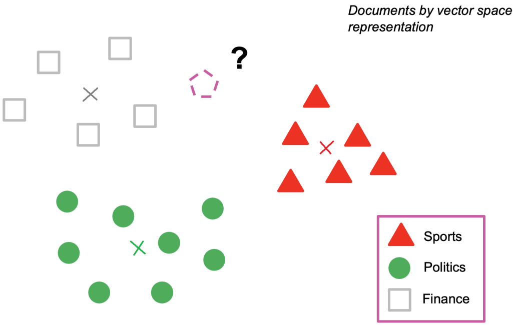
$$\boldsymbol{\mu_c} = \frac{1}{|D_c|} \sum_{\mathbf{d} \in D_c} \mathbf{d}$$

Where D_c is the set of documents in the corpus that belongs to class c and d is the vector representation of document d.

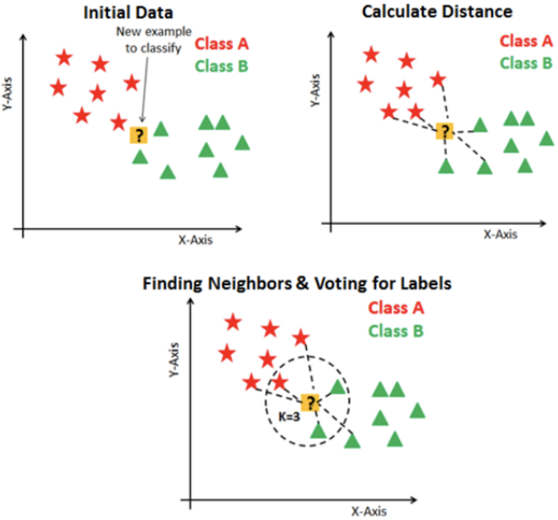
Rocchio Classifier (Nearest Centroid)

The predicted label of document d is the one with the smallest (Euclidean) distance between the document and the centroid.

$$\hat{c} = \arg\min_{c} ||\boldsymbol{\mu_c} - \mathbf{d}||$$



K-Nearest Neighbor



K-Nearest Neighbor

- Given a test document d, the KNN algorithm finds the k nearest neighbors of d among all the documents in the training set, and scores the category candidates based on the class of the k neighbors.
- After sorting the score values, the algorithm assigns the candidate to the class with the highest score.
- The basic nearest neighbors classification uses uniform weights: that is, the value assigned to a query point is computed from a simple majority vote of the nearest neighbors.
- Can weight the neighbors such that nearer neighbors contribute more to the fit.

Naïve Bayes

$\gamma($

great	2
love	2
recommend	1
laugh	1
happy	1
...	...

$)= c$

Bayes' Rule

- Applied to documents and classes
- For a document d and a class c

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}$$

Multinomial Naïve Bayes Assumptions

$$C_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \cdot P(w_1, w_2, \ldots, w_n | c)$$

- Bag of Words assumption: Assume position doesn't matter
- Conditional Independence: Assume the feature probabilities $P(w_i|c)$ are independent given the class c.

$$P(w_1, \ldots, w_n | c) = P(w_1|c) \cdot P(w_2|c) \cdot P(w_3|c) \cdot \ldots \cdot P(w_n|c)$$

- Hence:

$$C_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \cdot P(w_1|c) \cdot P(w_2|c) \cdot P(w_3|c) \cdot \ldots \cdot P(w_n|c)$$

$$C_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i \in positions} P(w_i|c)$$

Parameter estimation

- First attempt: maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c) = \frac{count(C = c)}{N_{doc}}$$

$$\hat{P}(w_i|c) = \frac{count(w_i, c)}{\sum_{w \in V} count(w, c)}$$

Problem with Maximum Likelihood

What if we have seen no training documents with the word coffee and classified in the topic positive (thumbs-up)?

$$\hat{P}(''\mathrm{coffee}''|\mathrm{positive}) = \frac{count(''\mathrm{coffee}'', \mathrm{positive})}{\sum_{w \in V} count(\mathrm{w}, \mathrm{positive})}$$

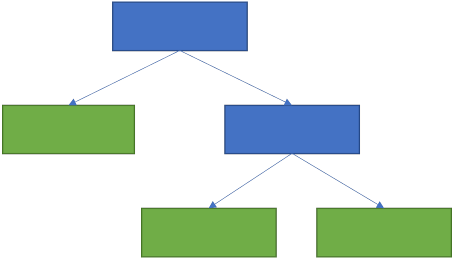
Zero probabilities cannot be conditioned away, no matter the other evidence!

$$C_{NB} = \underset{c \in C}{\mathrm{argmax}} P(c) \prod_{i \in positions} P(w_i|c)$$

Laplace (add-1) smoothing for Naïve Bayes

$$\hat{P}(w_i|c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} (count(w, c) + 1)}$$

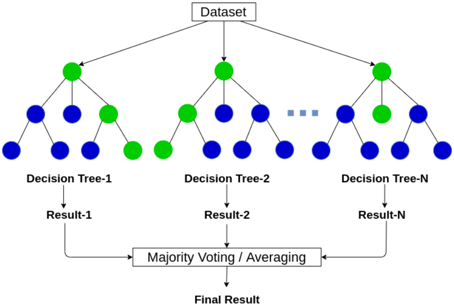
Decision Tree



- A decision tree is a hierarchical decomposition of the (training) data space, where a condition on the feature value is used to divide the data space hierarchically.
- Top-down, by choosing a variable at each step that best splits the set of items.
- Different algorithms to measure the homogeneity of the target variable within the subsets (e.g. Gini impurity, information gain)

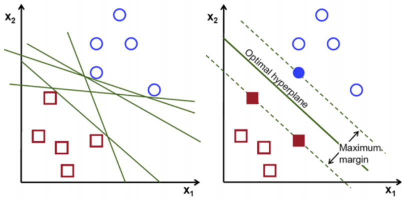
Random Forest

- Random forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.
- Fit multiple trees to bootstrapped samples of the data AND at each node select best predictor from only a random subset of predictors. Combine all trees to yield a consensus prediction



Support Vector Machine

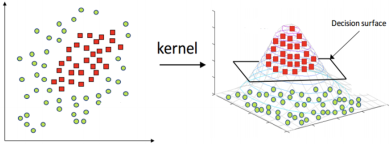
- The main principle of SVM is to determine separators in the search space which can best separate the different classes.
- SVM tries to make a decision boundary in such a way that the separation between the two classes is as wide as possible.



Support Vector Machine

- It is not necessary to use a linear function for the SVM classifier.
- With the kernel trick, SVM can construct a nonlinear decision surface in the original feature space by mapping the data instances non-linearly to a new space where the classes can be separated linearly with a hyperplane.

- SVM is quite robust to high dimensionality.

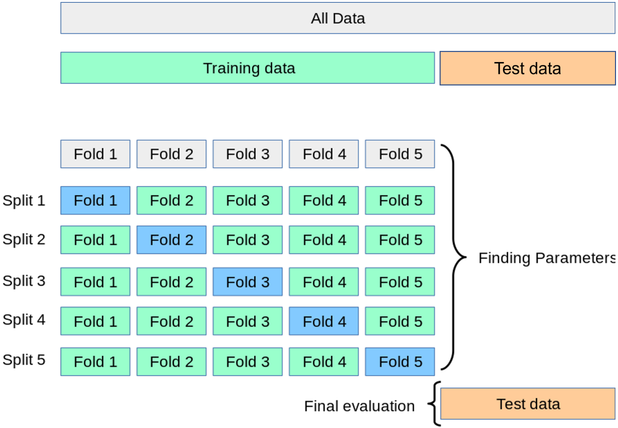


Evaluation

Data Splitting

- Training set
  - Validation set (dev set)
    - A dataset of examples used to tune the hyperparameters (i.e. the architecture) of a classifier. It is sometimes also called the development set or the “dev set”.
- Test set

Nested Cross Validation



adapted from
https://scikit-learn.org/stable/modules/cross_validation.html

Confusion matrix

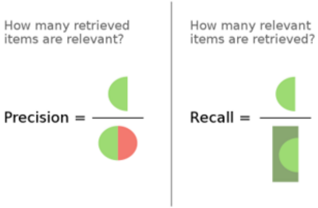
		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Accuracy

- What proportion of instances is correctly classified?
  TP + TN / TP + FP + FN + TN
- Accuracy is a valid choice of evaluation for classification problems which are well balanced and not skewed.
- Let us say that our target class is very sparse. Do we want accuracy as a metric of our model performance? What if we are predicting if an asteroid will hit the earth? Just say “No” all the time. And you will be 99% accurate. The model can be reasonably accurate, but not at all valuable.

Precision and recall

- Precision (also Positive Predictive Value): % of selected/retrieved items that are correct/relevant
- Recall (also sensitivity): % of correct/relevant items that are selected/retrieved.



- Precision is a valid choice of evaluation metric when we want to be very sure of our prediction.
- Recall is a valid choice of evaluation metric when we want to capture as many positives as possible.

A combined measure: F

A combined measure that assesses the precision/recall tradeoff is F measure (weighted harmonic mean):

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

where β is a positive real number and is chosen such that recall is considered β times as important as precision.

Balanced F1 measure: $\beta = 1$, $F = 2PR/(P + R)$

The Real World

No training data?

- Manually written rules
  - If (x or y) and not (w or z) then categorize as class1
  - Need careful crafting
  - Low accuracy
  - Domain-specific
  - Time-consuming
- Active learning
- Unsupervised methods

Very little data?

- Use Naïve Bayes, KNN, Rocchio
- Get more labeled data
- Find ways to label data
- Try semi-supervised methods
- Try transfer learning

A reasonable amount of data?

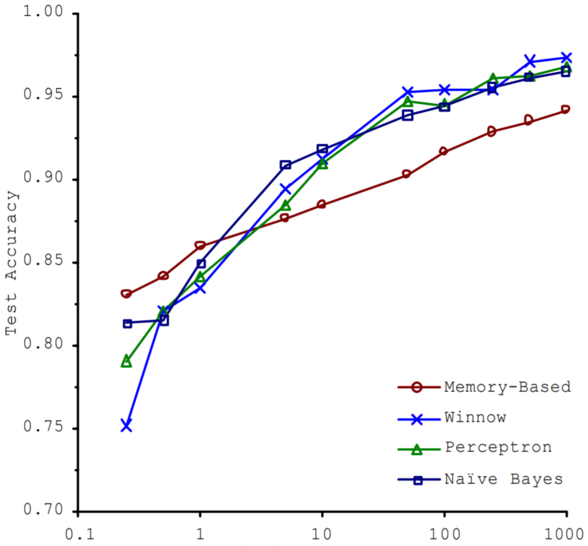
- Works with the more complex classifiers
  - SVM
  - Random forest

A huge amount of data?

- Can achieve high accuracy!
- At a cost:
  - SVMs (train time) or KNN (test time) can be too slow

Accuracy as a function of data size

- With enough data
  - Classifier may not matter



How to tweak performance

- Domain-specific features and weights: very important in real performance
- Sometimes need to collapse terms:
  - Part numbers, chemical formulas, ...
  - But stemming generally doesn't help
- Upweighting: Counting a word as if it occurred twice:
  - Title words
  - First sentence of each paragraph (Murata, 1999)
  - In sentences that contain title words
- Hyperparameter optimization

Summary

Summary

- Vector space model & BOW
- Text Classification
- Evaluation

Practical 3