Practical Text Analytics: Text Classification

Fan Dai

Iowa State University

December 2, 2019

Recall: Text analytics process

Text Analytics Process

- Planning the text analytics projects
- Preparing and preprocessing text
- Analyzing text data
  1. Latent Semantic Analysis (LSA)
  2. Topic Models
     - Probabilistic Latent Semantic Analysis
  3. Sentiment Analysis

Recall: Text analytics process

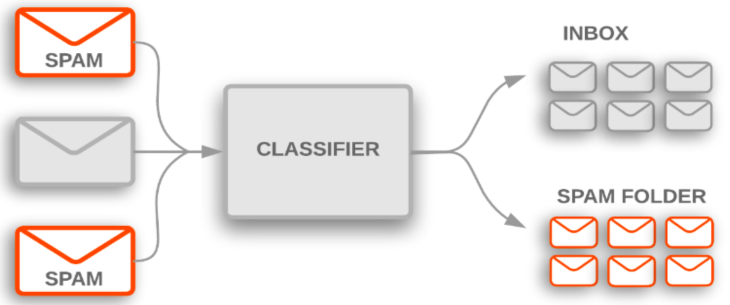
Text Analytics Process

- Planning the text analytics projects
- Preparing and preprocessing text
- Analyzing text data
  1. Latent Semantic Analysis (LSA)
  2. Topic Models
     - Probabilistic Latent Semantic Analysis
  3. Sentiment Analysis
  4. *Classification Analysis*

     (*Chapter 9, Anandarajan et al.(2019)*)

Text Classification Example

- Email software: spam filtering



Source: **https://developers.google.com/machine-learning/guides/text-classification**

- Classification model: identify the message as spam and redirecting the message to your spam folder

Text Classification: Intro

- Text classification/categorization
  - Usage: Email classification, news filtering, document organization and retrieval, opinion mining, . . .

Text Classification: Intro

- Text classification/categorization
  - Usage: Email classification, news filtering, document organization and retrieval, opinion mining, . . .
  - **Supervised learning model**
    - Automatically categorize or classify text documents when the true classification of the document is known
    - Infer the naturally occurring groupings in the data, without knowing the actual groupings or categories
    - Make predictions

Text Classification: Intro

- Text classification/categorization
  - Usage: Email classification, news filtering, document organization and retrieval, opinion mining, . . .
  - **Supervised learning model**
    - * Automatically categorize or classify text documents when the true classification of the document is known
    - * Infer the naturally occurring groupings in the data, without knowing the actual groupings or categories
    - * Make predictions
  - Differences from LSA and topic models
    - * There is certainty in the natural groupings or underlying factors of the text data
    - * The analysis process
    - * The assessment of the model fit

Text Classification: General Process

The General Text Classification Process

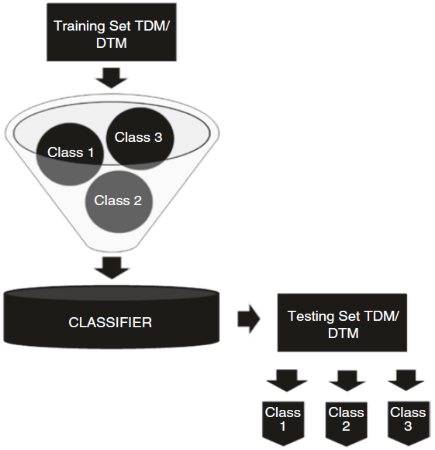


Figure: Text classification process

		Actual		
		Yes	*No*	**Total Predicted**
Predicted	*Yes*	3	6	9
	No	7	4	11
	Total Actual	10	10	20

- Goodness of fit
  1. Accuracy = $\frac{\text{\# of correct predictions}}{\text{\# of total predictions}}$
  2. $\text{Accuracy}_i = \frac{\text{\# of correct predicted}_i}{total\text{\# of predicted}_i}$
  3. $\text{Recall}_i = \frac{\text{\# of correct predicted}_i}{total\text{\# of actual}_i}$

Source: **Practical text analytics: maximizing the value of text data**

Text Classification: Models)

- Classification Models
    1. Naive Bayes
    2. k-nearest neighbors
    3. Support vector machines
    4. Decision trees
    5. Random forests

Text Classification: Naive Bayes

Naive Bayes (NB) models

- The most likely classification $\hat{C}$, of document D, is equal to

$$\hat{C} = \mathrm{argmax} P(D|C)P(C)$$

  - $P(D|C)$: Conditional probability of a document given its class
  - $P(C)$: Probability of the classification

Text Classification: Naive Bayes

Naive Bayes (NB) models

- The most likely classification $\hat{C}$, of document D, is equal to

$$\hat{C} = \text{argmax} P(D|C)P(C)$$

  - $P(D|C)$: Conditional probability of a document given its class
  - $P(C)$: Probability of the classification

What is naive in naive Bayes?

Text Classification: Naive Bayes

Naive Bayes (NB) models

- The most likely classification $\hat{C}$, of document D, is equal to

$$\hat{C} = \text{argmax} P(D|C)P(C)$$

  - $P(D|C)$: Conditional probability of a document given its class
  - $P(C)$: Probability of the classification

What is naive in naive Bayes?

Assumption: Terms are conditionally independent given the class

Text Classification: Naive Bayes

Naive Bayes (NB) models

- The most likely classification $\hat{C}$, of document D, is equal to

$$\hat{C} = \text{argmax} P(D|C)P(C)$$

  - $P(D|C)$: Conditional probability of a document given its class
  - $P(C)$: Probability of the classification

What is naive in naive Bayes?

Assumption: Terms are conditionally independent given the class

- $P(D|C) = P(W_{D,1}|C)P(W_{D,2}|C)\cdots P(W_{D,D_m}|C)$
  - $W_{D,1},\ldots,W_{D,D_m}$ are the terms in Document D.

Text Classification: Naive Bayes

Naive Bayes (NB) models

- The most likely classification $\hat{C}$, of document D, is equal to

$$\hat{C} = \text{argmax} P(D|C)P(C)$$

  - $P(D|C)$: Conditional probability of a document given its class
  - $P(C)$: Probability of the classification

What is naive in naive Bayes?

Assumption: Terms are conditionally independent given the class

- $P(D|C) = P(W_{D,1}|C)P(W_{D,2}|C)\cdots P(W_{D,D_m}|C)$
  - $W_{D,1},\ldots,W_{D,D_m}$ are the terms in Document D.
- Simple and efficient, but not necessarily true in real-world data

Text Classification: k-Nearest Neighbors

k-Nearest Neighbors (kNN)

- Find the k nearest matches in training data and then using the label of closest matches to predict.
- Distances such as euclidean or cosine similarity are used to find the closest match.

Text Classification: k-Nearest Neighbors

k-Nearest Neighbors (kNN)

- Find the k nearest matches in training data and then using the label of closest matches to predict.
- Distances such as euclidean or cosine similarity are used to find the closest match.
- Simple algorithm steps
  1. Start with the training data $\{(D_1, C_1), (D_2, C_2), \ldots, (D_m, C_m)\}$ and a new document D^* to be classified
  2. Calculate distance $dist(D^*, D_i)$ for each label document in the training set
  3. Select the k nearest documents to D^* based on $dist(D^*, D_i)$
  4. Assign to D^* the most frequent class in the k nearest documents

Text Classification: k-Nearest Neighbors

k-Nearest Neighbors (kNN)

- Find the k nearest matches in training data and then using the label of closest matches to predict.
- Distances such as euclidean or cosine similarity are used to find the closest match.
- Simple algorithm steps
  1. Start with the training data $\{(D_1, C_1), (D_2, C_2), \ldots, (D_m, C_m)\}$ and a new document D^* to be classified
  2. Calculate distance $dist(D^*, D_i)$ for each label document in the training set
  3. Select the k nearest documents to D^* based on $dist(D^*, D_i)$
  4. Assign to D^* the most frequent class in the k nearest documents
- Non-parametric, sensitive to the chosen value of k

Text Classification: Support Vector Machines

Support Vector Machines (SVM)

- Search for the optimal hyperplane, among all possible separating hyperplanes, which has the largest margin of separation between classes.

Text Classification: Support Vector Machines

Support Vector Machines (SVM)

- Search for the optimal hyperplane, among all possible separating hyperplanes, which has the largest margin of separation between classes.

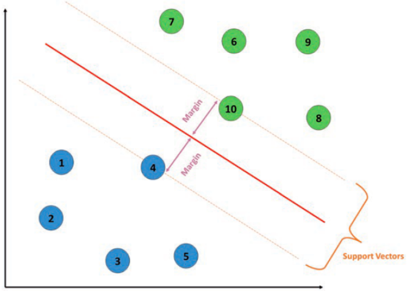


Figure: Text classification process

Source: **Practical text analytics: maximizing the value of text data**

- Class “blue”: documents 1–5
- Class "green”: documents 6–10
- Red line: the optimal hyperplane that creates the largest margin

Text Classification: Support Vector Machines

Support Vector Machines (SVM)

- Search for the optimal hyperplane, among all possible separating hyperplanes, which has the largest margin of separation between classes.

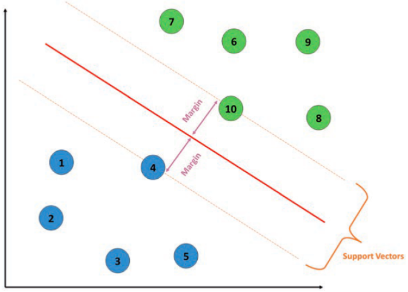


Figure: Text classification process

- Class “blue”: documents 1–5
- Class "green”: documents 6–10
- Red line: the optimal hyperplane that creates the largest margin

Source: **Practical text analytics: maximizing the value of text data**

- SVM can handle the high dimensionality and sparsity of the DTM used as the input quite efficiently

Text Classification: Decision Trees

Decision Trees

- Use recursive partitioning to separate classes within a dataset

Text Classification: Decision Trees

Decision Trees

- Use recursive partitioning to separate classes within a dataset

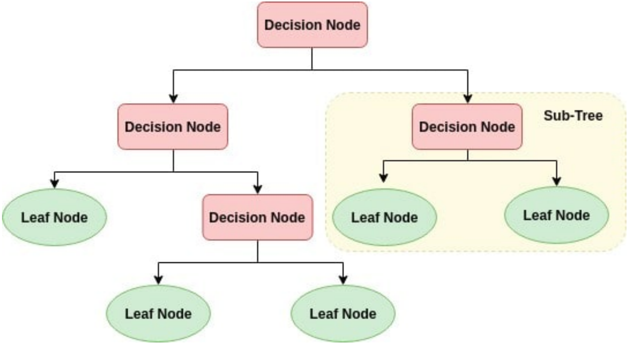


Figure: Decision tree

Source: **Decision Tree Classification in Python (article) - DataCamp**

Text Classification: Decision Trees

Decision Trees

- Use recursive partitioning to separate classes within a dataset

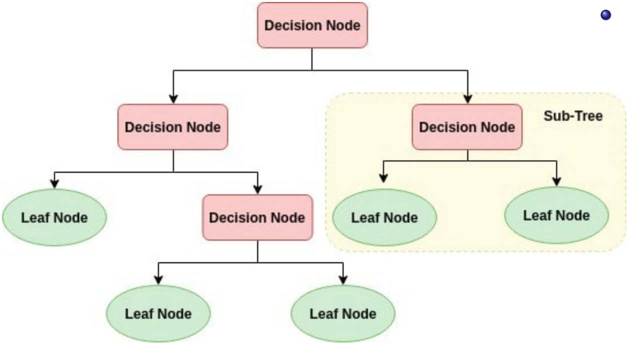


Figure: Decision tree

Source: **Decision Tree Classification in Python (article) - DataCamp**

- Splitting criteria:
  1. Gini Index: measures divergence between probability distributions
  2. Entropy/Deviance: measures homogeneity
  3. Information Gain: measures the reduction in Entropy
  4. Chi-Square Test: measures the likelihood of a split

Text Classification: Decision Trees

Decision Trees

- Use recursive partitioning to separate classes within a dataset

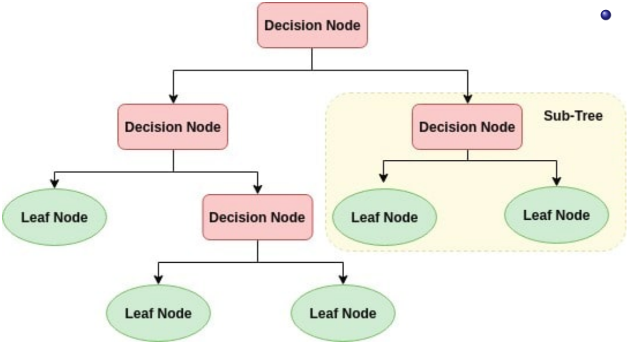


Figure: Decision tree

Source: **Decision Tree Classification in Python (article) - DataCamp**

- Splitting criteria:
  1. Gini Index: measures divergence between probability distributions
  2. Entropy/Deviance: measures homogeneity
  3. Information Gain: measures the reduction in Entropy
  4. Chi-Square Test: measures the likelihood of a split

- See more details in *reference link*

Text Classification: Random Forests

Random Forests

- A "combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest" (Breiman 2001, 1).

Text Classification: Random Forests

Random Forests

- A "combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest" (Breiman 2001, 1).
  - creates a forest of decision trees that on average are more accurate

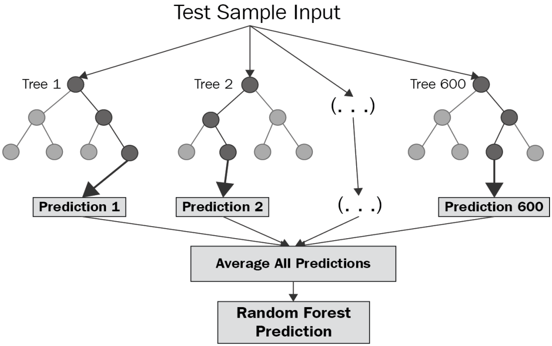


Figure: Random forest

Source: **https://towardsdatascience.com/**