LING 520: Computational Analysis of English

Semester: FALL '16

Instructor: Sowmya Vajjala

Iowa State University, USA

18 October 2016

Class Outline

- Assignment 3 Discussion
- Text Classification: last week Recap
- Brief overview of some more classification algorithms
- Assignment 4 Description

Assignment 3 Discussion

Text Classification - last week Recap

What is text classification?

- Assuming we have some example texts which have some pre-defined class/category labels,
- text classification has this goal: developing a ”model” of categorization based these example texts (training data)
- ... and using this model to assign categories to new texts.

Note: J&M 3rd Edition draft chapters on Jurafksy's website has a good chapter on Text Classification. Please read. `https://web.stanford.edu/~jurafsky/slp3/7.pdf`

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model
- Step 2: You design some features that you think can distinguish between these categories (kitchen sink strategy or hand-crafted)

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model
- Step 2: You design some features that you think can distinguish between these categories (kitchen sink strategy or hand-crafted)
- Step 3: Convert those texts into feature vectors.

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model
- Step 2: You design some features that you think can distinguish between these categories (kitchen sink strategy or hand-crafted)
- Step 3: Convert those texts into feature vectors.
- Step 4: Develop or use an existing learning algorithm that can learn a classification function based on the values of all these features for the texts

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model
- Step 2: You design some features that you think can distinguish between these categories (kitchen sink strategy or hand-crafted)
- Step 3: Convert those texts into feature vectors.
- Step 4: Develop or use an existing learning algorithm that can learn a classification function based on the values of all these features for the texts
- Step 5: Evaluate the classification based on some measure. Tune your classifier with better features, better learning algorithm or with more data (or all of those)

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model
- Step 2: You design some features that you think can distinguish between these categories (kitchen sink strategy or hand-crafted)
- Step 3: Convert those texts into feature vectors.
- Step 4: Develop or use an existing learning algorithm that can learn a classification function based on the values of all these features for the texts
- Step 5: Evaluate the classification based on some measure. Tune your classifier with better features, better learning algorithm or with more data (or all of those)
- Step 6: Stop when you are satisfied, and deploy your classifier in some real world application

Text Classification - Process

- Step 1: You have some collection of texts labeled with some categories, which you will use as training data for your model
- Step 2: You design some features that you think can distinguish between these categories (kitchen sink strategy or hand-crafted)
- Step 3: Convert those texts into feature vectors.
- Step 4: Develop or use an existing learning algorithm that can learn a classification function based on the values of all these features for the texts
- Step 5: Evaluate the classification based on some measure. Tune your classifier with better features, better learning algorithm or with more data (or all of those)
- Step 6: Stop when you are satisfied, and deploy your classifier in some real world application
  (and discover all that research still does not result in material benefit!)

Measuring Success

Multiple ways. Depends on the nature of your dataset, and your application.

- Prediction accuracy on test set: typically used in most ML evaluation for text, images, videos, all sorts of things
- False positive rate (Type 1 Error), False negatives (Type 2 error) - typically in medical applications
- Precision (TP/(TP+FP)), Recall (TP/(TP+FN)), F-score (2PR/(P+R)) - typically in information retrieval, text classification
- Revenue increase - in e-commerce applications

Some commonly used features in text classification

- ngrams (word, character, POS, mixed representations)
- specific hand-crafted features: e.g., number of spelling errors, number of dependent clauses per clause, number of preposition phrases per sentence etc.
- feature representation: binary (presence or absence), count (number of occurrences), ratios etc.

Some commonly used learning algorithms

- **Naive bayes classifier**
- **K-nearest neighbors classifier**
- Logistic regression
- Decision Trees and Random forests
- Support vector machines, neural network classifiers

.. etc. Note: I will only give an overview of how these work, to give an intuitive idea. Details are found in machine learning classes or textbooks.

Logistic Regression

Logistic Regression

- Goal: same as any other classification algorithm. Classify a given text into one of the pre-defined categories, based on some feature representation.
- Difference compared to naive bayes or knn: learning function.
- Learning function in Logistic Regression:
    1. If x is my text, f_1, f_2... f_i is my feature vector for this text, C = c1, c2, c3 are my three possible categories,

Logistic Regression

- Goal: same as any other classification algorithm. Classify a given text into one of the pre-defined categories, based on some feature representation.
- Difference compared to naive bayes or knn: learning function.
- Learning function in Logistic Regression:
  1. If x is my text, f_1, f_2... f_i is my feature vector for this text, C = c1, c2, c3 are my three possible categories,
  2. for a class c,
     $p(c|x) = \frac{exp(\sum_{i=1}^{n}(w_i * f_i(c,x))}{\sum_{c' \in C} exp(\sum_{i=1}^{n}(w_i * f_i(c',x))}$

Logistic Regression

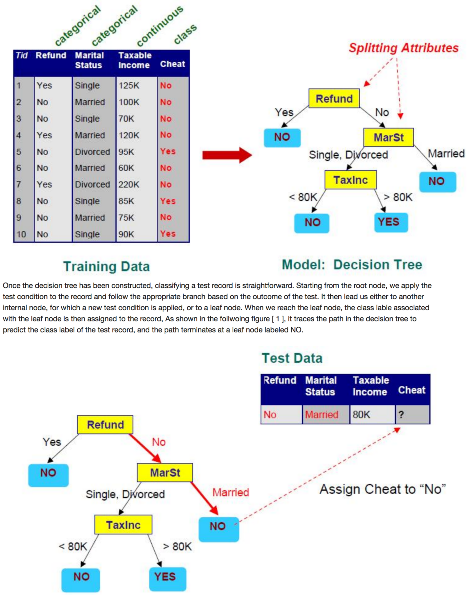
- Goal: same as any other classification algorithm. Classify a given text into one of the pre-defined categories, based on some feature representation.
- Difference compared to naive bayes or knn: learning function.
- Learning function in Logistic Regression:
  1. If x is my text, f_1, f_2... f_i is my feature vector for this text, C = c1, c2, c3 are my three possible categories,
  2. for a class c,
     $p(c|x) = \frac{exp(\sum_{i=1}^{n}(w_i * f_i(c,x))}{\sum_{c' \in C} exp(\sum_{i=1}^{n}(w_i * f_i(c',x))}$
- Note: You don't have to struggle with the math. There are ready to use implementations you can use if you want. This is just to give an intuitive understanding of the differences between different learning algorithms.

Decision Trees and Random forests

Decision Trees

- Idea: Perform classification by asking a series of questions. Next question asked depends on answer to the current question.
- Construct a hierarchy (e.g., tree) of such questions. Keep asking until you reach some leaf node (leaf notes here are the text categories)
- Advantage: Relatively interpretable model. Fast to classify because it is just rule-checking once there is the rule-model ready.

Decision Trees - Example



	categorical	categorical	continuous	class
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

Model: Decision Tree

Once the decision tree has been constructed, classifying a test record is straightforward. Starting from the root node, we apply the test condition to the record and follow the appropriate branch based on the outcome of the test. It then lead us either to another internal node, for which a new test condition is applied, or to a leaf node. When we reach the leaf node, the class lable associated with the leaf node is then assigned to the record, As shown in the follwoing figure [1], it traces the path in the decision tree to predict the class label of the test record, and the path terminates at a leaf node labeled NO.

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

How should we construct the tree?

General process overview:

- Among all features, pick the one with most discriminative value (How?).

How should we construct the tree?

General process overview:

- Among all features, pick the one with most discriminative value (How?). One approach: calculate "information gain" of all features and pick the one with highest gain.
- What is IG?: IG measures how much of a grouping can one feature do. How much reduction in "entropy" of the data (disorder) occurs due to this feature?

How should we construct the tree?

General process overview:

- Among all features, pick the one with most discriminative value (How?). One approach: calculate "information gain" of all features and pick the one with highest gain.
- What is IG?: IG measures how much of a grouping can one feature do. How much reduction in "entropy" of the data (disorder) occurs due to this feature?
- What is entropy?:
  1. Entropy of a training dataset T is given by: $\mathrm{H(T)} = -\sum_{i=1}^{c} P(cat_i) log_2 P(cat_i)$ where $P(cat_i)$ is the probability of getting cat_i if you pick a random instance from training data.
  2. If my training data has 9 instances of cat.A, 7 instances of cat.B, $\mathrm{H(T)} = -[\frac{9}{16} log_2 \frac{9}{16} + \frac{7}{16} log_2 \frac{7}{16}]$, which is 0.9836.
- What is the information gain of a feature? if T is the training data and f is a feature, IG(f) = H(T) - H(T|f)

How should we construct the tree? - intuition

- The feature with the most IG splits the training data into some groups. Now, pick another feature which splits the data further.
- Keep organizing features that lets us split in a hierarchy like this, until no more splitting is possible and we end up with category labels at leaf nodes.

How should we construct the tree? - intuition

- The feature with the most IG splits the training data into some groups. Now, pick another feature which splits the data further.
- Keep organizing features that lets us split in a hierarchy like this, until no more splitting is possible and we end up with category labels at leaf nodes.
- Note of caution: I am oversimplifying. The actual training process involves more than this. Fortunately, you don't need to do all that yourself.
- in NLTK: there is a class called DecisionTreeClassifier, which allows you to train and predict using decision trees.

Random Forests

- General idea: Combination of several classifiers will result in a better classifier ("bagging")
- Random forests use decision trees for each of those "several" classifiers.

Random Forests

- General idea: Combination of several classifiers will result in a better classifier ("bagging")
- Random forests use decision trees for each of those "several" classifiers.
- Process:
  1. Separate training data into some N datasets.
  2. Build a decision tree with each of these N datasets (with all or some subset of features)
  3. During prediction, use predictions from all the trees, and take a majority voting (or any such aggregation method) among all decision trees.
  4. Same procedure can be used for numeric scale as well (average instead of majority in last step).
- Generally considered robust. Can become slow with n-gram features as there are too many features.

Other Popular Algorithms

- Support Vector Machines
- Neural network algorithms (aka "Deep Learning")
- ... and many many more. If you really really want to know all details, take a machine learning course.
- scikit-learn is a free machine learning library in Python that has implementations of several classification algorithms.

Other Popular Algorithms

- Support Vector Machines
- Neural network algorithms (aka "Deep Learning")
- ... and many many more. If you really really want to know all details, take a machine learning course.
- scikit-learn is a free machine learning library in Python that has implementations of several classification algorithms.
- Caution: scikit-learn, NLTK, or some other toolkit - each expect their input to be in a certain format. You should check their documentation about that, and write some code to convert data to the format these tools can understand.

Assignment 4 Description

- 2 questions, 15 marks (5+10)
- for a change, no coding!!

Next Class

- Hui-Hsien's talk
- Discussion about final projects
- Text classification review and practice
- Mid-term feedback