Machine Learning Approaches to NLP

Part II

Sameer Maskey
Topics for Today

NLP -- ML

- Text Categorization
- Linear Methods of Classification
  - Perceptron
  - Naïve Bayes
  - Weka Tutorial
Regression to Classification

- Can we build a regression model to model binary classes?
- Train Regression and threshold the output
  - If $f(x) \geq 0.7$ CLASS1
  - If $f(x) < 0.7$ CLASS2
  - $f(x) \geq 0.5$ ?
- Is this ok?

Happy/Good/ClassA

Sad/Not Good/ClassB
We looked at cosine similarity for text classification. Besides cosine similarity, there are many other ways for text classification. Dimensionality reduction is one way of classification: Fisher’s Linear Discriminant. We can also try to find the discriminating hyperplane by reducing the total error in training. Perceptrons is one such algorithm.
Half Plane and Half Spaces

- Half plane is a region on one side of an infinite long line, and does not contain any points from other side
- Half space n-dimensional space obtained by removing points on one side of hyperplane (n-1 dimension)
  - What would it look like for a 3 dimensional space
Discriminative Classification

\[ f(x) = \mathbf{w}^T x + b \]
We want to find a function that would produce least training error

\( R_n(w) = \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, f(x_i; w)) \)
Minimizing Training Error

Given training data $< (x_i, y_i) >$
We want to find $w$ such that
$y_i(w \cdot x_i) > 0$ if $y_i > 0$
$y_i(w \cdot x_i) < 0$ if $y_i < 0$

- We can iterate over all points and adjust the parameters
  \[
  w \leftarrow w + y_i x_i \quad \text{if } y \neq f(x_i; w)
  \]
- Parameters are updated only if the classifier makes a mistake
We are given \((x_i, y_i)\)
Initialize \(w\)
Do until converged
   if error\([(y_i, f(x_i, w)) == TRUE)\]
      \(w \leftarrow w + y_i x_i\)
   end if
End do
Text Classification with Perceptron

Which side of the hyperplane is this document?

Feature vector in D dimensions
Separating hyperplane in D-1 dimensions
Perceptron may not always converge

Ok for two classes, not trivial to extend it to multiple classes

Not the optimal hyperplane

- Many hyperplanes that separates the class
- Depends on random initialization
Generative vs. Discriminative

- **Generative Classifier**
  - Model joint probability $p(x,y)$ where $x$ are inputs and $y$ are labels
  - Make prediction using Bayes rule to compute $p(y|x)$
- **Discriminative Classifier**
  - Try to predict output directly
  - Model $p(y|x)$ directly
Generative Classifier

- We can model class conditional densities using Gaussian distributions
- If we know class conditional densities
  - $p(x|y=C1)$
  - $p(x|y=C2)$
- We can find a decision to classify the unseen example
Bayes Rule

\[ p(y|x) = \frac{p(x|y)p(y)}{p(x)} \]

→ So how would this rule help in classifying text in two different categories; Diabetes vs Hepatitis

→ Think about distribution of count of the word diabetes for example
Generative Classifier

- If we have two classes C1 and C2
- We can estimate Gaussian distribution of the features for both classes
  - Let’s say we have a feature x
    - x = length of a document
  - And class label (y)
    - y = 1 diabetes or 2 hepatitis

Find out $\mu_i$ and $\sum_i$ from data for both classes

Gaussian Distribution

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$
Generative Classifier

- Given a new data point find out posterior probability from each class and take a log ratio.
- If higher posterior probability for C1, it means new x better explained by the Gaussian distribution of C1.

\[ p(y|x) = \frac{p(x|y)p(y)}{p(x)} \]

\[ p(y = 1|x) \propto p(x|\mu_1, \sum_1)p(y = 1) \]
Naïve Bayes Classifier

- Naïve Bayes Classifier a type of Generative classifier
  - Compute class-conditional distribution but with conditional independence assumption
- Shown to be very useful for text categorization
Conditional Independence

- Given random variables X, Y, Z, X is conditionally independent of Y given Z if and only if

\[ P(X|Y, Z) = p(X|Z) \]

\[ P(X|Y) = P(X_1, X_2|Y) \]
\[ = P(X_1|X_2, Y)P(X_2|Y) \]
\[ = P(X_1|Y)P(X_2|Y) \]
Conditional Independence

- For a feature vector with ‘n’ features we get

\[ P(X_1, X_2, \ldots, X_N | Y) = \prod_{i=1}^{N} P(X_i | Y) \]

N features are conditionally independent of one another given Y

Why would this assumption help?
Naïve Bayes Classifier for Text

\[ P(Y_k, X_1, X_2, \ldots, X_N) = P(Y_k) \prod_i P(X_i | Y_k) \]

Here \( N \) is the number of words, not to confuse with the total vocabulary size.
Naïve Bayes Classifier for Text

\[
P(Y = y_k | X_1, X_2, ..., X_N) = \frac{P(Y = y_k) P(X_1, X_2, ..., X_N | Y = y_k)}{\sum_j P(Y = y_j) P(X_1, X_2, ..., X_N | Y = y_j)}
\]

\[
= \frac{P(Y = y_k) \prod_i P(X_i | Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i | Y = y_j)}
\]

\[
Y \leftarrow \arg \max_{y_k} P(Y = y_k) \prod_i P(X_i | Y = y_k)
\]
Naïve Bayes Classifier for Text

Given the training data what are the parameters to be estimated?

\[ P(y) \quad P(X|y_1) \quad P(X|y_2) \]

Diabetes : 0.8  
Hepatitis : 0.2

the: 0.001  
diabetic : 0.02  
blood : 0.0015  
sugar : 0.02  
weight : 0.018  
...

the: 0.001  
diabetic : 0.0001  
water : 0.0118  
fever : 0.01  
weight : 0.008  
...

\[ y \leftarrow \arg\max_y P(y = y_k) \prod_i P(X_i|y = y_k) \]
Estimating Parameters

- Maximum Likelihood Estimates
  - Relative Frequency Counts

- For a new document
  - Find which one gives higher posterior probability
    - Log ratio
    - Thresholding

- Classify accordingly
Smoothing

- MLE for Naïve Bayes (relative frequency counts) may not generalize well
  - Zero counts

- Smoothing
  - With less evidence, believe in prior more
  - With more evidence, believe in data more
Laplace Smoothing

- Assume we have one more count for each element
- Zero counts become 1

\[ P_{\text{smooth}}(w) = \frac{c_w + 1}{\sum_w \{c(w)+1\}} \]

\[ P_{\text{smooth}}(w) = \frac{c_w + 1}{N + V} \]

Vocab Size
Weka

- Publicly available free software that includes many common ML algorithms that are used in Natural Language Processing
- GUI and Commandline Interface
- Feature Selection, ML algorithms, Data filtering, Visualization
Weka Download and Setup

- [http://sourceforge.net/projects/weka/files/weka-3-4/3.4.17/weka-3-4-17.zip/download](http://sourceforge.net/projects/weka/files/weka-3-4/3.4.17/weka-3-4-17.zip/download)
- `>> unzip weka-3-4-17.zip`
- `>> java -jar weka-3-4-17/weka.jar`
- `>> Click on Explorer`
Filter Features
Visualize data

Data needs to be in ARFF format
Prediction Class at the end of feature list
Classifier Choice

Model Testing

Results

Weka Explorer

Select Classifier: BayesNet -D -Q weka.classifiers.bayes.net.search.local K2 -- -P 1 -S 5

--- Stratified cross-validation ---

Correctly Classified Instances 2508    73.7763 %
Incorrectly Classified Instances 927    26.2235 %
Kappa statistic                0.3668
Mean absolute error            0.2825
Root mean squared error        0.4569
Relative absolute error        74.0931 %
Root relative squared error    104.644 %
Total Number of Instances     3535

--- Detailed Accuracy by Class ---

<table>
<thead>
<tr>
<th>Class</th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>Class</th>
<th>TP</th>
<th>FP</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.773</td>
<td>0.38</td>
<td>0.896</td>
<td>0.773</td>
<td>0.815</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0.822</td>
<td>0.451</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>0.62</td>
<td>0.222</td>
<td>0.491</td>
<td>0.62</td>
<td>0.548</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0.548</td>
<td>0.451</td>
<td>0.62</td>
</tr>
</tbody>
</table>

--- Confusion Matrix ---

a  b <-- classified as
2046 333 | a = 0
344 552  | b = 1

Status
OK
Tasks

Modal Load/Save

Visualize Model
10-fold Cross Validation

- **10 fold cross validation**
  - Assuming we have 100K data points
    - Train on 90K (1 to 90,000)
    - Test on 10K (90,001 to 100,000)
  - But we can do this 10 times if we select different 10K of test data point each time
  - **10 experiments, build model and test times with 10 different sets of training and test data**
  - **Average the accuracy across 10 experiments**
  - **We can do any N-fold cross validation to test our model**

<table>
<thead>
<tr>
<th>Exp1</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
<th>10k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp2</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exp10</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
<td>10k</td>
</tr>
</tbody>
</table>
Interpreting Weka Results

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP  True Positive</td>
</tr>
<tr>
<td></td>
<td>FN  False Negative</td>
</tr>
</tbody>
</table>
## Precision, Recall, F-Measure

<table>
<thead>
<tr>
<th>Metric</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>$\frac{TP}{TP+FP}$</td>
</tr>
<tr>
<td>Recall</td>
<td>$\frac{TP}{TP+FN}$</td>
</tr>
<tr>
<td>F-Measure</td>
<td>$\frac{(1+\beta^2) \times \text{Precision} \times \text{Recall}}{(\beta^2 \times \text{Precision} + \text{Recall})}$</td>
</tr>
<tr>
<td>Accuracy</td>
<td>$\frac{TP+TN}{TP+TN+FP+FN}$</td>
</tr>
</tbody>
</table>
Confusion Matrix

- Assume we are classifying text into two categories Hepatitis (H) and Others (B)
- Let’s assume we had 1000 documents such that 500 are H and 500 are B
- Assume we got given predictions

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Actual</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>400</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>100</td>
<td>300</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.6667</td>
</tr>
<tr>
<td>Recall</td>
<td>0.8000</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.7273</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.7000</td>
</tr>
</tbody>
</table>
Commandline for Weka

- Make sure CLASSPATH variable is setup; can also give the path explicitly using –cp parameter
  - `>> export CLASSPATH=$CLASSPATH:/home/smaskey/soft/weka-3-4-17/weka.jar`

- Try to see if java can access the classes for classifiers
  - `>> java weka.classifiers.bayes.NaiveBayes`

- Try to build a model from commandline
  - `>>java weka.classifiers.trees.J48 -i -t data/weather.arff`

- Try other examples from Weka wiki
  - `>>java weka.classifiers.bayes.NaiveBayes -K -t soybean-train.arff -T soybean-test.arff -p 0`
Weka Demo

- Text Classification with Weka
  - Classify documents into Hockey or Baseball
- 20 Newsgroup corpus
- Code and data will be available from the course website