NLP Note: Text Classification

Lizi Chen

1 Text Classification (Naive Bayes, Logistic Regression / Maximum Entropy)

Represent the i^{th} instance in a corpus X as x^i, the true class label is $y^i, y^i \in Y$. We use vector $\mathbf{f}_i(y^i, x^i))$ to represent the features of the instance. To show $\mathbf{f}$ in blocked feature vector form:

$$\mathbf{f}_i(y, x) = [b_1, b_2, \cdots, b_k]$$

, where k is the number of true class labels; $|Y|$.

There are m features for each input x and class y, the dimension of a blocked feature vector is mk.

In Linear Models: we aim to find a weight vector $\mathbf{w} \in \mathbb{R}^{mk}$ that the class of x^i can be identified by $w^T\mathbf{f}_i(y, x)$. There are two broad ways to estimate $\mathbf{w}$:

1.1 Generative Model 生成式模型 vs. Discriminative 判别式模型 Model:

- **Generative:**
  - work with a probabilistic model of the data, weights are (log) local conditional probabilities, i.e., Naive Bayes.
  - Assume functional form for $P(X|Y),\ P(Y)$.
  - Estimate *probabilities* from data (need smoothing).
  - Use Bayes rule to calculate $P(X|Y)$.
- **Discriminative:**
  - set weights based on some error-related criterion, i.e., Logistic Regression aka. Maximum Entropy, KNN, Decision Tree, SVM.
  - Estimate *parameters* from data (need regularization).
  - Directly predict label by computing $P(X|Y)$.

Another example of Generative-Discriminative pair of learning methods: Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs).

2 Bayes Classifier, An Impractical Approach

This is a short summarized explanation of the example given in Tom M. Mitchell's Machine Learning book.

- Bayes Rule: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$
- Assume Y is boolean.
- X contains n boolean attributes, $X = [x_1, x_2, \cdots, x_n]$.

Thus, by applying Bayes Rule, we have:

$$P(Y = y_i|X = X_k) = \frac{P(X = X_k|Y = y_i)P(Y = y_i)}{\sum_{\mathbf{j}} P(X = X_k|Y = y_{\mathbf{j}})P(Y = y_{\mathbf{j}})} \tag{1}$$

It is impractical to accurately estimate $P(X|Y)$ because of the number of **parameters** to train: There are n features for a single X, each feature is a boolean, meaning 2 possible **values**, that is 2^n **values**, which requires at least $(2^n - 1)$ **parameters**. There are 2 possible y **values**, hence requires at least $2(2^n - 1)$ parameters for a $(X - y)$ pair. Then, there are k number of X instances, so the quantity of parameters are $2k(2^n - 1)$.

2.1 Conditional Independence

Therefore, we have to make assumption (just like all science do): *Given 3 sets of r.v. X, Y, Z, we say X is* **conditionally independent** *of Y given Z, i.f.f. the probability distribution governing X is independent of the value of Y given Z:*

$$\forall i, j, k, \quad P(X = x_i|Y = y_j, Z = z_k) = P(X = x_i|Z = z_k) \tag{2}$$

3 Naive Bayes for Classification

Given input x and true label y, where each input x can be a word vector or a feature vector: $x = [w_1, w_2, \cdots, w_n]$, NB Algorithm solves the problem by maximizing $P(y|x)$ using Bayes rule and assumption that each w_i for x is conditionally independent given y.

$$\begin{aligned} P(y|x) &= \frac{P(x|y)P(y)}{P(x)} = \frac{P(x|y)P(y)}{\sum_{y'} P(x|y')P(y')} \\ &= \frac{P(w_1, w_2, \cdots, w_n|y)P(y)}{\sum_{y'} P(w_1 m w_2, \cdots, w_n|y')P(y')} \\ &= \frac{P(y)\prod_i P(w_i|y)}{\sum_{y'}\left[P(y')\prod_i P(w_i|y')\right]} \end{aligned} \tag{3}$$

In the second line of Equation 3, we apply the assumption, so as to have the third result line.

We want to predict the $\hat{y}$, which is $\hat{y} = \arg\max P(y|x)$, since the denominator of Equation 3 does not depend on y, we can have:

$$\hat{y} = \arg\max P(y) \prod_i P(w_i|y) \tag{4}$$

Thanks to the assumption, we now only need $2n$ parameters.

clarify and visualize

3.1 Maximum Likelihood Estimate:

$$\hat{\theta}_{ijk} = \hat{P}(w_i = j|y = k) = \frac{\#(w_i = j \cap y = k)}{\#(y = k)} \tag{5}$$

$$\hat{\pi}_k = \hat{P}(y = k) = \frac{\#(y = k)}{|D|} \tag{6}$$

, where # refers the count function, $|D|$ means the number of elements in the training set D.

Smoothing: The numerator in Equation 5 may be 0 when $(w_i = j \cap y = k) = \emptyset$, thus we add smoothing factor l:

$$\hat{\theta}_{ijk} = \hat{P}(w_i = j|y = k) = \frac{\#(w_i = j \cap y = k) + l}{\#(y = k) + l\mathcal{J}}$$

, where $\mathcal{J}$ denotes the number of distinct values X_i can take on. when $l = 1$, this is Laplace smoothing.

Same thing for $\hat{\pi}$:

$$\hat{\pi}_k = \hat{P}(y = k) = \frac{\#(y = k) + l}{|D| + l\mathcal{K}}$$

, where $\mathcal{K}$ is the number of distinct values Y can take on.

3.2 Learning
3.3 Independence Assumption and Features, Non-Independence Issues
3.4 Code

4 Logistic Regression for Classification, Discriminative Model

5 Feature Representations

Write in a separate doc?

5.1 Challenge in feature engineering, p29
5.2 Learning Classifier Weights, and hoe to pick Weights
5.3 Derivative for Maximum Entropy

6 Logistic Regression aka. Maximum Entropy

7 TF-IDF

8 Neural Nets

8.1 From Maximum Entropy to Neural Nets

9 Further Readings:

Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression, Tom M. Mitchell http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf