CS-534 Machine Learning

Implementation Assignment 2

Mohammand Velayati (33.$\bar{3}$%) Lucas Wells (33.$\bar{3}$%) Chirag Shah (33.$\bar{3}$%)

Introduction

In this assignment we are given a set of tweets from Hillary Clinton's and Donald Trump's Twitter accounts. We will train a Naïve Bayes classifier to predict the account that a new tweet comes from. Throughout this document we use the variable C to represent the account that a given tweet comes from, Hillary's or Trump's, and $\mathbf{x}$ represents a tweet, such that each x_i is a single word in the tweet $\mathbf{x}$ of class C. Also, let V be the vocabulary of unique words found in all tweets of the training set.

The Naïve Bayes classifier

The Naïve Bayes classifier is based on Bayes' Theorem,

$$p(C|\mathbf{x}) = \frac{p(C)p(\mathbf{x}|C)}{p(\mathbf{x})} \propto p(C)p(\mathbf{x}|C), \tag{1}$$

that states that the probability of of observing a class C, given a set of features $\mathbf{x}$, is equal to the prior probability of observing class C times the likelihood of the feature begin in that class divided by the evidence $p(\mathbf{x})$. As the denominator does not depend on the class we can state that the probability of observing a class is proportional to the numerator. This is equivalent to the joint probability $p(C, x_1, \ldots, x_n)$. In practice this computation on high dimensional features is prohibitive since we must estimate $k \times (2^d - 1)$ parameters. Naïve Bayes relaxes the conditional probability model and assumes that each feature is conditionally independent from all other features. Thus, we can write the conditional distribution over C as

$$p(C|x_1, \ldots, x_n) \propto p(C) \prod_{i=1}^{n} p(x_i|C). \tag{2}$$

Combining the Naïve Bayes probability model with the Maximum A Posteriori (MAP) decision rule we get

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i|C_k). \tag{3}$$

We will compare the performance of two Naïve Bayes event models, Bernoulli and Multinomial, on our Twitter classification problem.

Bernoulli event model

In the Bernoulli event model we represent tweets as binary vectors, so each word $x_i \in \{0, 1\}$ and $|\mathbf{x}| = |V|$. Thus, $x_i = 1$ means that the word V_i is contained in the tweet at least once, and $x_i = 0$ means that word V_i is not in the tweet $\mathbf{x}$. In the Bernoulli model we use

$$p(\mathbf{x}|C) = \prod_{i=1}^{|V|} p_{x_i|C}^{x_i}(1 - p_{x_i|C})^{(1-x_i)} \tag{4}$$

to compute the likelihood that a given tweet $\mathbf{x}$ is from account C. And the likelihood that word x_i is found in class C_k is estimated by

$$p_{x_i|C_k} = \frac{n_k(x_i)}{N_k}, \tag{5}$$

where $n_k(x_i)$ denotes the number of occurrences of word x_i in class k and N_k is the number of tweets from class k. Since the product of many probabilities could result in underflow, we compute the log of the likelihoods by

$$\log p(\mathbf{x}|C) = \log\Big(\prod_{i=1}^{|V|} p_{x_i|C}^{x_i}(1 - p_{x_i|C})^{(1-x_i)}\Big) \tag{6}$$

$$= \sum_{i=1}^{|V|}[x_i \cdot \log p_{x_i|C} + (1 - x_i)\log(1 - p_{x_i|C})]. \tag{7}$$

Combining equations (1) and (7) we perform MAP estimation in log space with

$$\hat{y} = \underset{k\in\{1,\ldots,K\}}{\operatorname{argmax}} \log p(C_k) \sum_{i=1}^{|V|}[x_i \cdot \log p_{x_i|C_k} + (1 - x_i)\log(1 - p_{x_i|C_k})]. \tag{8}$$

The priors for each class are calculated using Maximum Likelihood Estimation

$$p(C_k) = \frac{N_k}{N}, \tag{9}$$

where N is the total number of tweets in both classes.

Multinomial event model

In the Multinomial case we represent tweets as integer vectors, so each word $x_i \in \mathbb{Z}$, where each element x_i denotes the frequency of that word in the tweet. The likelihood is given by

$$p(\mathbf{x}|C) = \prod_{i=1}^{|V|} p_{x_i|C}^{x_i}, \tag{10}$$

and the probability $p_{x_i|C_k}$ is the relative frequency of word x_i in tweets of class k divided by the total number of words in the tweets of that class. Mathematically, this can be expressed as

$$p_{x_i|C_k} = \frac{\sum_{j=1}^{N} x_{ji} b_{jk}}{\sum_{m=1}^{|V|} \sum_{j=1}^{N} x_{jm} b_{jk}}, \tag{11}$$

where b_{jk} is an indicator variable that equal 1 when x_i is of class $C = k$, and 0 otherwise. The estimate for all word likelihood in log space is then given by

$$\log p(\mathbf{x}|C) = \log\Big(\prod_{i=1}^{|V|} p_{x_i|C}^{x_i}\Big) \tag{12}$$

$$= \sum_{i=1}^{|V|} [x_i \cdot \log p_{x_i|C}]. \tag{13}$$

Finally, combining equations (1) and (13) we perform MAP estimation with

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \ \log p(C_k) \sum_{i=1}^{|V|} [x_i \cdot \log p_{x_i|C_k}], \tag{14}$$

where the priors are calculated using equation (9).

Implementation

1. (5 pts) Please explain how you use the log of probability to perform classification

2. (5 pts) Report the overall testing accuracy (number of correctly classified documents over the total number of documents) for both (Bernoulli and Multinomial) models.

Table 1: Overall testing accuracy for each model

Model	testing accuracy
Bernoulli	92.39%
Multinomial	92.70%

Table 2: Confusion matrices. H denotes Hillary Clinton's account and T for Trump's account.

(a) Bernoulli model

class	$y = H$	$y = T$
$\hat{y} = H$	47.20	5.12
$\hat{y} = T$	2.48	45.19

(b) Multinomial model

class	$y = H$	$y = T$
$\hat{y} = H$	47.20	4.81
$\hat{y} = T$	2.48	45.50

3. (5 pts) Whose tweets were confused more often than the other? Why do you think this is?

4. (5 pts) Identify, for each class, the top ten words that have the highest probability.

Table 3: Top ten most likely words from each account

Rank	H	T
1	make	great
2	people	The
3	America	#MakeAmericaGreatAgain
4	president	Trump
5	can	@realDonaldTrump
6	I	#Trump2016
7	We	Hillary
8	Donald	Thank
9	Hillary	will
10	Trump	I

Priors and overfitting (20 pts)

Identifying important features (20 pts)

We explored methods for reducing the vocabulary size to reduce the dimensionality of features and improve performance of classification. Intuitively we can think of the discriminative power of a word as the difference in the likelihood of that word appearing in one class versus the other. That is, for a given word x_i, the absolute difference of the likelihood $p_{x_i|C_k}$ where $k = 0$ and $k = 1$. Thus, as an example, if a word is very likely to appear in both classes, then the difference is small and it is said to have little discriminative power. This can be written as

$$\Delta\ell(\mathbf{x}_{\ell|k}) = |\mathbf{x}_{\ell|k=0} - \mathbf{x}_{\ell|k=1}|, \tag{15}$$

where $\mathbf{x}_{\ell|k} = \{p_{x_i|C_k}; \forall i, k\}$ given by equation (5) for the Bernoulli case and (11) in the Multinomial case. A new vocabulary set is generated by

$$V' = \{x_{m,\sigma} | x_{m,\sigma} = \underset{\mathbf{x}_{\ell|k}}{\text{argmax}^{\sigma}} \{\Delta\ell(\mathbf{x}_{\ell|k})\}, \sigma \in \{1, \ldots, \Sigma\}\}, \tag{16}$$

where the notation argmax^{σ} denotes a set of Σ maximum word likelihood differences. The number of words in the new vocabulary set is controlled by parameter λ in

$$\Sigma = \lfloor |V| \cdot (1 - \lambda) \rfloor. \tag{17}$$

We tested the effect of λ values in the range $(0.1, \ldots, 0.9)$ on the classification accuracy of each model and the size of the vocabulary (Table 4).

Table 4: Change in model accuracy resulting from vocabulary reduction. Bold face values indicate the best classification accuracy for each model.

λ	Bernoulli accuracy (%)	Multinomial accuracy (%)	$\lvert V' \rvert$
0.1	92.24	92.70	12,061
0.2	92.08	93.01	10,721
0.3	90.99	**93.17**	9,381
0.4	90.22	92.86	8,041
0.5	91.61	92.85	6,701
0.6	92.55	92.70	5,361
0.7	**93.17**	92.55	4,021
0.8	91.77	92.39	2,681
0.9	90.53	90.06	1,341

In both models, the accuracy can slightly improved by retaining only words with relatively large discriminative power in the dictionary. For the Bernoulli case, this improvement is highest at a 70% reduction in vocabulary size. And for the Multinomial case, at 30% reduction. In both cases, a comparable classification accuracy can be achieved while reducing the vocabulary size to 90% of it's original dimension.

Bonus