Hacettepe University

BBM 409: Introduction to Machine Learning Laboratory

Fall 2016

Assingment II photo collection

Author:
Batuhan Yaman

Instructor:
Aykut Erdem
TAs:
Burçak Asal
Aysun Koçak

November 10, 2016

1 Introduction

This report consists two parts.Part one contains answers for theoretical question about MLE and Naive Bayes.Part two is the analysis and results of Sentiment Analysis.

2 Informations

2.1 MLE

In statistics, maximum likelihood estimation (MLE) is a method of estimating the parameters of a statistical model given observations, by finding the parameter values that maximize the likelihood of making the observations given the parameters. MLE can be seen as a special case of the maximum a posteriori estimation (MAP) that assumes a uniform prior distribution of the parameters, or as a variant of the MAP that ignores the prior and which therefore is unregularized. [1]

2.2 Naive Bayes

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression,[1]:718 which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. [2]

2.3 Bag of Words

The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model has also been used for computer vision. [3]

3 Part I: Theory Questions

3.1 Maximum Likelihood Estimation

3.1.1 Question 2

Problem: Whizzco decide to make a text classifier. To begin with they attempt to classify documents as either sport or politics. They decide to represent each document as a (row) vector of attributes describing the presence or absence of words

Solution: Lets assume politics probabillity list as p and sports as s.
$p = [2, 1, 1, 5, 5, 1, 4, 5] s = [4, 4, 1, 4, 1, 1, 0, 1]$ and our target is $x = [1, 0, 0, 1, 1, 1, 1, 0]$
$P(w|c) = \frac{count(w,c)+1}{count(c)+|V|}$
$P(x|p) = \frac{1}{2} * \frac{3}{32} * \frac{6}{32} * \frac{6}{32} * \frac{2}{32} * \frac{5}{32} * \frac{6}{32} = 1.60 * 10^{-5}$
$P(x|s) = \frac{1}{2} * \frac{5}{32} * \frac{5}{32} * \frac{2}{32} * \frac{2}{32} * \frac{1}{32} = 1.49 * 10^{-6}$
$P(x|p) > P(x|s)$ Therefore, new document is about politics.

4 PART II: Image Classification

4.1 Introduction

In this section I implemented an algorithm using python to predict if the given review is positive or negative. I used "Bag of words" method to get the frequency of the words and Naive bayes to determine the class.

4.2 Algorithm

Firstly, all data is read and set properly.We need full path to read both train and test and also to use regex to retrieve filenames to extract ratings. Then bag of words is applied with given parameters in Analysis section.After that possibility table is calculated with or without using IDF.Ratings also added on this part. Finally, Naive bayes is done. Test data is added to an array in order. So if the size is k, our array size is 2k and first k is negative and the last k is positive. Correction of the estimation is calculated according to this. If the possibility is higher and it's in the correct "k" area, it named as correct.

4.3 Analysis

Parameters: $size = 12.500 \quad min\_df = 0.05$

Feature	Accuracy
Unigram without IDF	75.592
Biagram without IDF	66.168
Unigram + Biagram without IDF	76.308
Unigram with IDF	69,192
Biagram with IDF	65.832
Unigram + Biagram with IDF	75.008
Unigram with IDF and Ratings	77.076
Biagram with IDF and Ratings	63.168
Unigram + Biagram with IDF and Ratings	77.652

Table 1: Accuracy results.

Firsty,because we used min_df as 0.05, our accuracy has dropped.But we had so much features that makes the calculation difficult on insufficient RAM.With all features($min\_df = 1$) even with 1000 train and 1000 test data I had %84 accuracy results. In other way, result would be higher with more RAM or using sparse arrays.
Lets look at the first three item on our table. Unigram showed a better performance than Biagram and combined result is in between. Normally we can expect Biagram to do better accuracy but it may differ according to the data, just like happened here. Again, normally using idf would benefit us about accuracy but since we are cutting out many features it showed lower performance than calculations without IDF.
Other idea that came to my mind was adding ratings to our features. As we can see, if we strengthen the impact of the word according to the rating, it gives us a huge advantage on Unigram model. Also, I couldn't apply the ratings for results without IDF. It's because rating addition was experimental and i needed to calculate the correct affection amount otherwise it would affect too much or too little. This aproach can be improved.

4.4 Libraries

1. import sys - *for providing nice progress bar for calculation on console.*
2. from sklearn.feature_extraction.text import CountVectorizer - *for Bag of Words.*
3. from sklearn.feature_extraction.text import TfidfTransformer - *for IDF.*
4. import numpy as np - *for arrays and mathematical calculations.*
5. import glob - *to retrieve all files from a folde.r*
6. import re - *for regex to get ratings from file names.*

References

[1] `https://en.wikipedia.org/wiki/Maximum_likelihood_estimation`

[2] Russell, Stuart; Norvig, Peter (2003) [1995]. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955.

[3] Sivic, Josef (April 2009) , Efficient visual search of videos cast as text retrieval