Northeastern University, Khoury College of Computer Science

CS 6220 Data Mining — Assignment 5

Due: February 1, 2023(100 points)

Chun Wei Tseng
https://github.com/Chun-Wei-Tseng/CS6220-Assignment-5
tseng.chun@northeastern.edu

Naïve Bayes, Bayes Rules

The original performance of acoustic classification for Parkinsons Disease leverages speech recordings from controlled subject responses from variety of questions. The task in the competition was to detect whether or not a person *X* had Parkinsons disease from a sampling of data. As of 2018, the state of the art classifiers have achieved 90% correct classification on a held out dataset, both for subjects who had Parkinsons and those who did not (at equal rates). So, when classifier *Y* sees person *X*, it works correctly 90% of the time.

1. Let's say that we run a clinic. This clinic leverages this classifier, which has 90% accuracy. Also, let us say that we know that our current patient load is that 10% of the population have Parkinsons and 90% of the population do not. Let's also say that we're seeing patient *X*, and the classification algorithm has detected that they have Parkinson's disease. **What's the probability that indeed *X* has Parkinson's disease?**

Come up with the numerical solution and show your written work.
→Ans:

- X = 0 → No Parkinsons, X = 1 → Parkinsons, Y = 0 → Test negative, Y = 1 → Test positive
- P(X = 1) = 0.1, P(X = 0) = 0.9, P(Y = 1 | X = 1) = 0.9, P(Y = 0 | X = 1) = 0.1, P(Y = 1 | X = 0) = 0.1, P(Y = 0 | X = 0) = 0.9
- Find P(X = 1 | Y = 1) = ?
- $P(Y = 1) = P(Y = 1 | X = 1) * P(X = 1) + P(Y = 1 | X = 0) * P(X = 0) = 0.9 * 0.1 + 0.1 * 0.9 = 0.18$
- $P(X = 1 | Y = 1) = \frac{P(Y = 1 | X = 1) * P(X=1)}{P(Y=1)} = \frac{0.9*0.1}{0.18} = \frac{1}{2}$

The Sum of Conditional Probabilities

In class, we reviewed three main rules in Bayesian probability inference:

- Conditional Probability:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

- Bayes Theorem:

$$P(A|B)P(B) = P(B|A)P(A)$$

- Total Probability:

$$P(A) = \sum_i P(A|B_i)P(B_i)$$

A well-known outcome of the three sets of rules is the fact that the sum of all the conditional probabilities equals one.

2. Prove that:

$$\sum_i P(A_i|B) = 1$$

→ Ans:

- Based on the conditional probability: $P(A_i|B) = \frac{P(A_i|B)}{P(B)} \rightarrow \sum_i \frac{P(A_i|B)}{P(B)} = \frac{1}{P(B)} \sum_i P(A_i|B)$
- A collection of disjoint events from $A_1, A_2, \ldots, A_i$ given B ($\sum_i P(A_i|B)$) together make up the sample space P(B):
    - $\sum_i P(A_i|B) = P(B)$
- Therefore, $\sum_i P(A_i|B) = \frac{P(B)}{P(B)} = 1$

Mining Reviews in Small-ish Datasets

The question in this section is best started with the starter kit Colab. You can copy and past the cells to a `*.py` Python3 file or or simply use the Colab in your own Google Drive.

We'll process datasets that are small enough that they can fit into CPU memory. We'll review the Amazon purchasing reviews dataset. Amazon has collaborated with Universities to aggregate an extensive set of buyer reviews data. Most of the datasets (including the subcategory datasets) cannot fit into CPU RAM, and an even smaller subset of machine learning algorithms can actually process them.

With this dataset, you will need to do the following for question number 2. Typically we would split the dataset into training / validation / testing dataset, but we will only split into training / testing dataset in this assignment.

library).

- Plot a histogram of the data with the number of positive and negative reviews.
- Balance the data so that you are training with equal probabilities, $P(\text{Liked}) = P(\text{Not Liked}) = 0.5$. We will use the original distribution for the evaluation dataset.
- Use machine learning models to predict whether or not a review as written will result in a "Liked" rating. Try changing different parameters, including the maximum vocabulary size. Also try normalization.
- Try out Naïve Bayes, Decision Trees, Random Forests, and Logistic Regression machine learning algorithms

- Print out your best accuracy, precision, and recall numbers for each algorithm.

How do they compare? Which algorithms are overfitting?

```
-----Naive Bayes-----
Accuracy:  0.8085620994074253
Confusion Matrix:  [[ 841  726]
 [ 857 5845]]
Precision / Recall AUC:  0.9326411791042538
-----Random Forest-----
Accuracy:  0.8075946305478292
Confusion Matrix:  [[1350  217]
 [1374 5328]]
Precision / Recall AUC:  0.9610074962666943
Accuracy of training data: 0.9765731408335603
The Model is overfitted
-----Decision Tree Classifier-----
Accuracy:  0.7476115612528722
Confusion Matrix:  [[1202  365]
 [1722 4980]]
Precision / Recall AUC:  0.9415106624475881
Accuracy of training data: 0.9765731408335603
The Model is overfitted
-----Logistic Regression-----
Accuracy:  0.815576248639497
Confusion Matrix:  [[1306  261]
 [1264 5438]]
Precision / Recall AUC:  0.9592310768478249
```

→ Ans: Logistic Regression has the highest accuracy, followed by Naïve Bayes, Random Forest, and Decision Tree. Random Forest has the highest Precision / Recall AUC, meaning that it has good overall performance in terms of classification.

Random Forest and the Decision Tree Classifier are overfitted because their accuracy on predicting the training data is a lot higher than their accuracy on predicting the testing dataset (larger than 0.1).

Tips and Tricks

You can preprocess the data using files in `cs6220hw5.py`, which you can download here. This will remove *stop words*, punctuation, and ensure that the texts are all lowercase. Caution that this implementation is exceedingly slow and un-optimized. (Extra credit to anyone who implements a faster version.)

Featurize your inputs to the algorithm with `CountVectorizer()`. Make sure to set the maximum vocabulary size / number of features, `max_features`.