COMPSCI 270 **Instructor:** Kris Hauser

Homework 3 **Due:** 03/29/2017

1 Instructions

Read this assignment carefully, complete the programming assignment and answer all of the written questions. Place all of your code in the `hw4_p1.py`, `hw4_p2.py`, and `hw4_p3.py` files and your written answers in the `hw4_answers.txt` file in the appropriate locations. Submit all of these files on Sakai before 4:20pm on 3/29/2017.

1.1 Homework Policies

Homework and lab assignments **must be completed individually**. Students are permitted and even encouraged to discuss assignments. However, any attempt to duplicate work that is not your own – for example, in the form of detailed written notes, copied code, or seeking answers from online sources – is strictly prohibited and will be considered cheating.

Homework assignments are due by the end of class on the due date unless otherwise specified. **No extensions will be granted** except for extenuating circumstances. Extensions are granted at the instructor's discretion, and valid extenuating circumstances include, for example, a debilitating illness (with STINF), death in the family, or travel for varsity athletics. Extensions will not be granted for personal or conference travel, job interviews, or a heavy course load.

2 Overview

This assignment covers Naïve Bayes classifiers and Maximum A Posteriori (MAP) learning.

3 Problems

3.1 Question 1 (35 points)

In this question, you will implement a Naïve Bayes model on a toy spam prediction problem. Each message is processed into four binary features, $f1, ..., f4$. The model gives a class probability $P(\text{Spam}) = 0.4$, and the following feature probabilities:

f	$P(f\|\text{Spam})$	$P(f\|\text{Not-Spam})$
$f1$	0.5	0.1
$f2$	0.3	0.6
$f3$	0.6	0.01
$f4$	0.99	0.98

Programming In `hw4_p1.py`, implement the Naïve Bayes function `naive_bayes` that accepts as input the class probabilities, feature probabilities, and an instance. As before, probability tables are

given as a dictionary that maps possible values to probabilities. The feature conditional probability tables are given as a thrice-nested dictionary, whose first key is the feature name, the second key is the class probability, and the third key is the feature value. An instance is a map from feature names to (0,1) values indicating presence or absence of each feature. [Note: be careful to make sure your implementation is not hard-coded for the spam detection problem – we will use it again in question 3]

Written

3.1.1 Which feature out of $f1, ..., f4$ is the most predictive of spam? That is, if you could choose only one feature to observe, which would give you the greatest ability to discriminate between Spam and Not-Spam? Why?

3.1.2 If you wanted to augment this problem so that it could also predict a third class of email, called Phishing, how many more (independent) probability values would you have to specify? What are they?

3.2 Question 2 (40 points)

In this question you will write an algorithm for learning spam filter probabilities from data.

Programming In `hw4_p2.py`, implement the Naïve Bayes learning function `learn_naive_bayes` that takes as input the class variable, a list of feature variables, and a dataset. It should output the Naïve Bayes class probabilities and feature probabilities so they can be directly used via the `naive_bayes` function of question 1.

In this problem you must use Maximum A Posteriori (MAP) learning to estimate the parameters using a Beta prior. In practice, this means that you will simply use virtual counts before you start tallying the elements of the dataset.

Written

3.2.1 The use of MAP learning is intended to make the system more robust to small amounts of data. But, as the designer of the system, the prior parameter adds another "knob" to tune. Try your learner with virtual counts 0, 1, and 100, and inspect the probability distributions. What is a potential problem with virtual counts of 0? What is the problem with a virtual count of 100?

3.2.2 What method might be used to quantitatively evaluate whether a prior is a good choice for a given problem? What measure of performance is used in that method?

3.3 Question 3 (25 points)

In this problem you are given a dataset of NCAA College Basketball statistics for the 2016-2017 season. Each game has a first-listed team, which will be considered the "team", and the second-listed team will be considered the "opponent". Your job will be to predict whether the "team" defeats the opponent. The source of this data is `https://www.spreadsheetsports.com/`

`2015-ncaa-basketball-game-data`.

You are provided framework code that parses and creates of Boolean features for this problem. It will then use your code from Problem 1 and Problem 2 to learn and make predictions of the `team_won` variable.

Programming

1. Implement functions to compute of the fraction of true positives, false positives, true negatives, and false negatives. Then implement functions to compute your classifier's precision and recall. Each of these elements should be placed in the indicated variables in `learn`().
2. As described below, use the `p3_explorations`() function as a sandbox to explore how the choice of features in learning the Naïve Bayes classifier affects the prediction accuracy on the testing set.

Written

3.3.1 What prediction accuracy on the training set do you get when including all available features? What prediction accuracy do you get when including no features? List the precision and recall for both of these cases.

3.3.2 Why does testing accuracy always increase as you add more features? What potential downsides are there to adding more features?

3.3.3 With all features, tweak the prediction threshold p_threshold to 0.01, 0.25, 0.75, and 0.99. What accuracies, false positive rate, and false negative rate do you get on the training set? Explain these trends.

3.4 Bonus Question (10 points)

Develop a Naïve Bayes classifier that achieves 75% prediction accuracy or better on the 2017 NCAA tournament results. You may tweak the features, the 0.5 classification threshold, and or the prediction method. You may wish to even generate entirely new features rather than simply relying on averages. Explain your method and include your code in your submission. What is the likelihood of the NCAA tournament bracket according to your model?