CS37300 Homework	— Homework 3—	Last update: September 29, 2017
https://www.cs.purdue.edu/homes/ribeirob/courses/Fall2017		Due: Oct 8, 2017 (11:59pm)

Instructions and Policy: Each student should write up their own solutions independently. You need to indicate the names of the people you discussed a problem with; ideally you should discuss with no more than two other people.

- YOU MUST INCLUDE YOUR NAME IN THE HOMEWORK
- The answers (without the python scripts) MUST be in submitted in a single PDF file via Blackboard.
- The python scripts will be submitted separately via turnin at data.cs.purdue.edu.
- Please write clearly and concisely - clarity and brevity will be rewarded. Refer to known facts as necessary.
- Theoretical questions MUST include the intermediate steps to the final answer.
- Zero points in any question where the python code answer doesn't match the answer in the PDF.
- Python code answers MUST adhere to the format described below.

There are TWO (2) questions in this homework.

Python code guidelines

Turn in each question of your homework in a separate python file named `hw3-X.py`, where X is the question number.

Your code is REQUIRED to run on either Python 2 or Python 3 at scholar.rcac.purdue.edu. Preferably use Python 3 (Python 2 will also be accepted). The TA's will help you with the use of the scholar cluster. If the name of the executable is incorrect, it won't be graded. Please make sure you didn't use any library/source explicitly forbidden to use. If such library/source code is used, you will get 0 pt for the coding part of the assignment. If your code doesn't run on scholar.rcac.purdue.edu, then even if it compiles in another computer, your code will still be considered not-running and the respective part of the assignment will receive 0 pt.

Q1 (50 pts): [Only Pseudocode (no Python)] In this assignment we will consider a decision tree (this is not a programming assignment). Consider the following set of training data D:

Electricity use	*Season*	*Heating system*
High	Winter	Oil
Low	Winter	Oil
High	Winter	Electric
High	Winter	Electric
Low	Summer	Oil
Low	Summer	Electric
High	Summer	Electric
Low	Summer	Electric

(a) **(20 pts)** Give a greedy algorithm (in pseudocode) that we can use to build a binary decision tree to predict electricity-use (E) given season (S) and heating-system (H). Each each recursive iteration, in order to decide to split the remaining data into D_1 and D_2, we will be using a hypothetical score function $f(D_1, D_2)$. Your pseudocode should build the tree until there can be no further splits in any of the branches.

(b) **(20 pts)** Build the tree using entropy gain as the score function f (draw the tree). At each decision tree node, describe D_1 and D_2 (the data used at each of its children).
Hint: You can use the following approximations $\log_2(1) = 0$, $\log_2(2) = 1$, $\log_2(3) = 1.5$, $\log_2(4) = 2$, $\log_2(5) = 2.5$, $\log_2(6) = 2.5$, $\log_2(7) = 3$, $\log_2(8) = 3$. Tie breaks should take into account the fact that $\log_2(i) < \log_2(i+1)$, for all $i > 0$.

(c) **(10 pts)** Given a new test example $\{S = Winter,\ H = Electric\}$, what is the probability that it will have high electricity usage ($E = High$)? If the example's true class label is '*Low*', what is the 0-1 loss of your prediction?

Q2 (50 pts): [Only one item (Q2.2) asks for a Python code submission, but you will also need python to answer some of the other questions (but no code submission is required except for Q2.2)] In this programming assignment you will implement a naive Bayes classification (NBC) algorithm and evaluate it on the *Yelp* dataset. Instructions below detail how to turn in your code and assignment on data.cs.purdue.edu.

Note: *You are welcome to use any methods from the standard python libraries available on the CS computers (e.g., numpy) but you shouldn't use methods from machine learning libraries (e.g., scikit-learn).*

Dataset Details

Download the datafile `yelp2_train.csv` using the following routine:

```
import urllib.request

my_url = "https://www.cs.purdue.edu/homes/ribeirob/courses/Fall2017/data/yelp2_train.csv"

local_filename, headers = urllib.request.urlretrieve(my_url)

with open(local_filename) as in_file:
    for line in in_file.readlines():
        # REMEMBER TO SKIP THE FIRST HEADER LINE
        # WHEN ASKED, REMEMBER TO DIVIDE THE DATA INTO TEST AND TRAINING
        # process the string line into a matrix
```

This data set is part of the Yelp database. The datafile has $20,000$ rows and 19 attributes: 15 discrete and 4 continuous ({`latitude`, `longitude`, `reviewCount`, `checkins`}).

Algorithm Details

You will implement algorithms to learn (i.e., estimate) and apply (i.e., predict) the NBC that we discussed in class.

Features: Consider the 15 discrete attributes in `yelp2_train.csv` for $\mathbf{X}$. Since the NBC can handle multi-valued discrete attributes, there is no need to create binary features.

Class label: Consider the attribute `goodForGroups` as the class label, i.e., $Y = \{1, 0\}$.

Smoothing: Implement Laplace smoothing in the parameter estimation. For an attribute X_i with k values, Laplace correction adds 1 to the numerator and k to the denominator of the maximum likelihood estimate for $P(X_i = x_i \mid Y)$.

Evaluation: When you apply the learned NBC to predict the class labels of the examples in the test set, use (i) zero-one loss, and (ii) squared loss to evaluate the predictions.

Let $y(i)$ be the true class label for example i and let $\hat{y}(i)$ be the prediction for i. Let p_i refer to the probability that the NBC assigns to the true class of example i (i.e., $p_i := p(\hat{y}(i) = y(i))$. If $p_i \geq 0.5$, the prediction for example i will be correct (i.e., $\hat{y}(i) = y(i)$), but otherwise if $p_i < 0.5$, the prediction will be incorrect (i.e., $\hat{y}(i) \neq y(i)$).

Then the zero-one loss for a test dataset T of n instances is:

$$Loss_{0/1}(T) = \frac{1}{n} \sum_{i \in n} \left\{ \begin{array}{ll} 0 & \text{if } y(i) = \hat{y}(i) \\ 1 & \text{otherwise} \end{array} \right\} \tag{1}$$

The squared loss for a test dataset T of n instances is:

$$Loss_{sq}(T) = \frac{1}{n} \sum_{i \in n} (1 - p_i)^2 \tag{2}$$

Code specification

Your python script should take two arguments as input.

1. *trainingDataFilename*: corresponds to a subset of the Yelp data (in the same format as `yelp2_train.csv`) that should be used as the *training set* in your algorithm.
2. *testDataFilename*: corresponds to another subset of the Yelp data (in the same format as `yelp2_train.csv`) that should be used as the *test set* in your algorithm.

Your code should read in the training/test sets from the csv files, learn a NBC model from the training set, apply the learned model to the test set, and evaluate the predictions with zero-one loss and squared loss.

Name your file `hw3-2.py`. The input and output should look like this:

```
$ python hw3-2.py train-set.csv test-set.csv
ZERO-ONE LOSS=0.2305
SQUARED LOSS=0.0711
```

THERE SHOULD BE NO OTHER OUTPUT TO THE SUBMITTED CODE

Note: This is how we will run your code to grade correctness in Q2.2 below.

Assignment

Given the Yelp dataset D with discrete attributes $\mathbf{X}$ and class Y as described above,

1. NBC details (20 pts)
   (a) Write down the mathematical expression for $P(Y \mid X)$ given by the NBC **(in PDF)**
   (b) State the naive assumption that lets us simplify the expression $P(X \mid Y)P(Y)$. What rule(s) of probability are used to simplify the expression? **(in PDF)**
   (c) What part of the expression corresponds to the class prior? Considering the entire Yelp data as the training dataset, calculate the maximum likelihood estimate for the class prior with and without smoothing. What is the effect of smoothing on the final probabilities? **(in PDF)**
   (d) Specify the full set of parameters that need to be estimated for the NBC model of the Yelp data. How many parameters are there? **(in PDF)**

(e) Write an expression for an arbitrary conditional probability distribution (CPD) of a discrete attribute X_i with k distinct values (conditioned on a binary class Y). Include a mathematical expression for the maximum likelihood estimates of the parameters of this distribution (with smoothing), which correspond to counts of attribute value combinations in a data set D. **(in PDF)**

(f) For the Yelp data, explicitly state the mathematical expression for the maximum likelihood estimates (with smoothing) of the CPD parameters for the attribute `priceRange` conditioned on the the class label `goodForGroups`. **(in PDF)**

(g) Consider the entire Yelp data as the training dataset and `goodForGroups` as the class label. Estimate the conditional probability distributions of the following attributes with and without smoothing:

(i) `priceRange`
(ii) `alcohol`
(iii) `noiseLevel`
(iv) `attire`

What is the effect of smoothing (e.g., any difference compared to Q1c)? Which attribute shows the most association with the class? **(in PDF)**

2. Implement a naive Bayes classification algorithm in python. (20 pts)
We will run several tests on your code to assess the accuracy for different samples. **(Python Code to Submit Using Turnin)**

3. Evaluate the NBC using cross validation and learning curves. (10 pts)

(a) For each % in $[1, 10, 50]$:

For i in [0...9]:

- Randomly sample % of the data to use as the training dataset.
- Use the remaining (1-%) of the data as the test dataset.
- Learn a model from the training data and apply it to the test data.
- Measure the performance on the test data using zero-one loss.

Record the mean zero-one loss observed across the ten trials for each training set size (i.e., sample %). Record the mean squared loss across the ten trials for each training set size. **(all in PDF)**

(b) Plot a learning curve of training set size vs. zero-one-loss (report the mean performance measured above) **(add to PDF)**. Compare to the baseline *default* error that would be achieved if you just predicted the most frequent class label in the overall data. Discuss the results (e.g., how is zero-one loss impacted by training set size). **(all in PDF)**

(c) Plot a learning curve of training set size vs. square-loss. Discuss how zero-one loss performance compares to square-loss. **(all in PDF)**

Submission Instructions:

After logging into data.cs.purdue.edu, borg01.cs.purdue.edu, xinu01.cs.purdue.edu, or moore01.cs.purdue.edu, please follow the steps on Piazza to submit your code.