CS 529: Assignment #2

Z. Berkay Celik
zcelik@purdue.edu

(Due Monday 4/10/2021, 11:59 PM)

Instructions

Info: This HW includes both theory and coding problems. Please read the course policy before starting your HW.

- Logistic Regression, Naive Bayes, Decision Trees, k-NN and K-means are widely applied to security datasets (as evidenced by recent papers from top-tier conferences). These questions will help you understand them, and apply your own datasets in the future.
- Your code must work with Python 3.5+ (you may install the Anaconda distribution of Python).
- You will submit the solutions to the assignment on **Brightspace**, please check **Brightspace** to understand the format for the submission of the answers for each question.
- You can always use course **Campuswire** page to ask your questions. I encourage you to answer questions to help each other.
- Failure to follow the instructions will lead to a deduction in points.

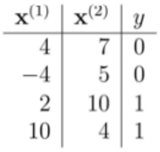
1 Problem 1: Concepts [15pt]

Info: Please use at most four sentences per question.

1. What is the principal assumption in the Naive Bayes' model, and when is this assumption useful?
2. When do we expect k-NN to be better than logistic regression?
3. Suppose we have two classes. 150 examples in the + class and 50 examples in the - class. What is the entropy of the class variable (you can leave this in terms of logs)?
4. Assume that you observed the following number of requests to a web server from two domains, Domain A and Domain B (Domain A: Class$_1$, Domain B: Class$_2$).

   Number of Requests from Domain A = $\{2, 3, 4, 3, 4, 3, 3, 4, 5, 3, 2, 3, 4, 3, 2, 2, 3, 4, 5, 6\}$
   Number of Requests from Domain B= $\{22, 23, 24, 23, 24, 23, 23, 24, 25, 23\}$

   Using the data, estimate the mean, std. dev. and prior (that is, $P(A)/P(A+B)$ and $P(B)/P(A+B)$) for each class.
5. We'd like to model each conditional probability of the form $\mathtt{P(x^{(i)}|y)}$ as a normal distribution for the training data below. We then use the naive Bayes assumption to write an expression for:
   $\mathtt{P(y = 0|x)}$ $\mathtt{P(y = 1|x)}$. Please show you answer clearly.



$\mathbf{x}^{(1)}$	$\mathbf{x}^{(2)}$	y
4	7	0
-4	5	0
2	10	1
10	4	1

2 Problem 2: Decision Trees [25pt]

> **Info:** You will investigate building a decision tree for a binary classification problem. The training data is given in Figure 1 with 16 instances that will be used to learn a decision tree for predicting whether a mushroom is edible or not based on its attributes (`Color, Size and Shape`). Please note the label set is a binary set `{Yes, No}`. For this problem, you need to submit a report in PDF for your solutions. It is okay for you to upload a hand-drawn image of the final decision tree, you don't need to generate the same on a computer. For all the remaining questions, please type up your solutions and do not submit hand-written solutions.

Instance	Color	Size	Shape	Edible?
D1	Yellow	Small	Round	Yes
D2	Yellow	Small	Round	No
D3	Green	Small	Irregular	Yes
D4	Green	Large	Irregular	No
D5	Yellow	Large	Round	Yes
D6	Yellow	Small	Round	Yes
D7	Yellow	Small	Round	Yes
D8	Yellow	Small	Round	Yes
D9	Green	Small	Round	No
D10	Yellow	Large	Round	No
D11	Yellow	Large	Round	Yes
D12	Yellow	Large	Round	No
D13	Yellow	Large	Round	No
D14	Yellow	Large	Round	No
D15	Yellow	Small	Irregular	Yes
D16	Yellow	Large	Irregular	Yes

Table 1: Mushroom data with 16 instances, three categorical features, and binary labels.

1. Which attribute would the algorithm choose to use for the root of the tree? Show the details of your calculations. Recall from lectures that if we let S denote the data set at current node, A denote the feature with values $v \in V$, H denote the entropy function, and S_v denote the subset of S for which the feature A has the value v, the gain of a split along the feature A, denoted `InfoGain(S; A)` is computed as:

   $$\text{InfoGain}(\mathcal{S}, A) = H(\mathcal{S}) - \sum_{v \in V} \left(\frac{|\mathcal{S}_v|}{|\mathcal{S}|} \right) H\left(\mathcal{S}_v\right)$$

   That is, we are taking the difference of the entropy before the split, and subtracting the entropy of each new node after splitting, with an appropriate weight depending on the size of each node.

2. Draw the full decision tree that would be learned for this data (assume no pruning and you stop splitting a leaf node when all samples in the node belong to the same class, i.e., there is no information gain in splitting the node).

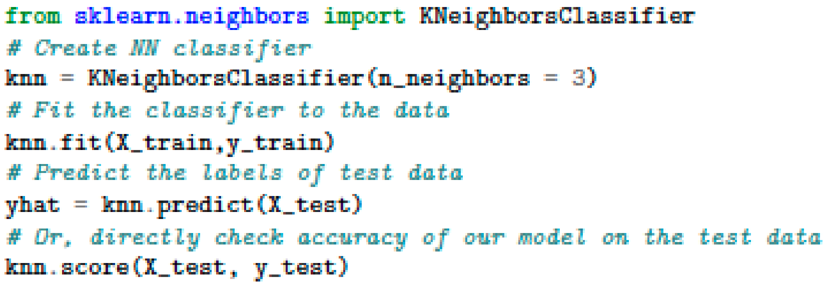
3. Table 1 includes only categorical features. However, some datasets such as the ones in security often include real valued (numerical) features. Handling real valued (numerical) features is totally different from categorical features in splitting nodes. In this problem, we look for a simple solution to decide good thresholds for splitting based on numerical features. Specifically, when there is a numerical feature in data, an option would be treating all numeric values of feature as discrete, i.e., proceeding exactly as we do with categorical data. Do any problems arise when we use a tree derived this way to classify an unseen example? Please support your claims.

3 Problem 3: k-NN [30pt]

Info: You will use the Pima Indians Diabetes data set from the UCI repository to experiment with the k-NN algorithm. Your goal is to find the optimal value for the number of neighbors k.

Running and Deliverable. You will create **Problem3.ipynb** file to put the implementation code for all questions below. Make sure your notebook is runnable on Anaconda with Python 3.5+. The results and discussions should be included in the PDF file.
You do not need to implement the k-NN algorithm and encouraged to use the implementation in scikit-learn. Below is a simple code showing the steps to use the NN implementation when the number of neighbors is 3:

NOTE: For this question, you are free to choose any objective function (distance measure) to minimize. There is no one correct answer.

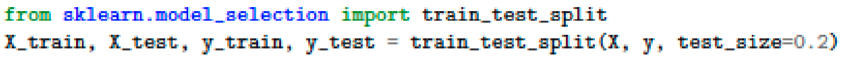


```
from sklearn.neighbors import KNeighborsClassifier
# Create NN classifier
knn = KNeighborsClassifier(n_neighbors = 3)
# Fit the classifier to the data
knn.fit(X_train,y_train)
# Predict the labels of test data
yhat = knn.predict(X_test)
# Or, directly check accuracy of our model on the test data
knn.score(X_test, y_test)
```

To accomplish this task, please do:

1. Download the provided **Pima.csv** data file and load it using pandas. As a sanity check, make sure there are 768 rows of data (potential diabetes patients) and 9 columns (8 input features including Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age, and 1 target output). Note that the data file has no header and you might want to explicitly create the header. The last value in each row contains the target label for that row, and the remaining values are the features. Report the statistics of each feature (min, max, average, standard deviation) and the histogram of the labels (target outputs).

2. Split the data into training and test sets with 80% training and 20% test data sizes. You can easily do this in scikit-learn, e.g.,



```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

   Use 5-fold cross-validation on training data to decide the best number of neighbours k. To this end, you can use the built in functionality in scikit-learn such as `cross_val_score` (note that this function returns the accuracy for each fold in an array and you have to average them to get the accuracy for all splits). For $k = 1, 2, 3, \ldots, 15$ compute the 5-fold cross validation error and plot the results (with values of k on the x-axis and accuracy on the y-axis). Include the plot in your report and justify your decision for picking a particular number of neighbors k.

3. Evaluate the k-NN algorithm on test data with the optimal number of neighbours you obtained in previous step and report the test error (you will use the same value of k you have found in 2.).

4. Process the input data by subtracting the mean (a.k.a. centralization) and dividing by the standard deviation (a.k.a. standardization) over each dimension (feature), repeat the previous part and report the accuracy. Do centralization and standardization impact the accuracy? Why?

4 Problem 4: Clustering (k-Means++) [30pt]

For this problem, you will implement the k-means++ algorithm in Python. You will then use it to cluster the *Iris dataset*. The corresponding data file *iris.data* file is provided, while the file iris.names contains a description of the data. The features x are given as the first four comma-separated values in each row in the data file. The labels y are the last entry in each row, but you do NOT need the class label as it is unknown for clustering. It is upto you to decide if you want to standardize the input or not, before running k-Means++. You can report whichever gives the best results but clearly mention if you have standardized the data or not. You will create a Jupyter notebook named **Problem4.ipynb** for this question, please answer the following questions in the Jupyter notebook.

1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
5. class: {Iris Setosa, Iris Versicolour, Iris Virginica}

You need to,

1. Create a new data set with two features by computing the ratio of raw features $x = (x_1, x_2)$ where x_1 = (sepal length/sepal width) and x_2 = (petal length/petal width)) and plot the data to observe the clusters in data by yourself (use class label to color the data points for better illustration of clusters).
2. Implement the k-means++ algorithm. Submit the source code (documented!) of your implementation.
3. Cluster the modified Iris dataset with the two features explained above. Run your algorithm 50 times over the data with different values of clusters $k = 1, 2, \ldots, 5$ and plot the accuracies (x and y axes should be the number of clusters and the clustering objective.
4. Based on the above plot, decide the number of final clusters and justify your answer. For the chosen number of clusters,
   - Create a plot showing how the accuracy changes with the number of iterations.
   - Create a plot with the data colored by assignment, and the cluster centers.

5 Problem 5: Logistic Regression–Extra Credit [50pt]

> **Info:** For this question you will use the *Optdigits data* (provided). You have the following the files:
>
> - optdigits.names // explanation of data
> - optdigits.tra // training data
> - optdigits.tes // test data
>
> You will submit **problem5.pynb** for this question that includes your code and answers.
>
> **Hint:** We showed the derivation of logistic regression objective function for binary classification. You can now make logistic regression classifier work on multi-class classification problems with one versus all method. See Andrew Ng's explanation.

1. Use the logistic regression (multi class; implemented from scratch) training algorithm and plot the training and test errors.

2. Use the logistic regression classifier with regularization so that you also penalize large weights ($\lambda \|\boldsymbol{w}\|_2^2$). Plot the average training and test errors for at least 5 different values of regularization parameter λ.