Project #1
CAC 350

Please choose one of the tasks below. Do not Google for answers! You may Google for syntax all you like, and you may Google to better understand the packages/functions, but do not Google for solutions. Instead, post questions to the Stack Overflow forum.

For whichever option you choose, please include text where appropriate to explain what the results imply. Interpret what is happening.

1. Using the cars dataset, create a binary classifier that determines whether or not a car is efficient. Just like we had the feature vector of the pixels and a label of what digit each number is, you will want the same thing for the cars.

- First, you'll need to retrieve the dataset and load it into your editor.
- You'll need to add a feature to the data that states whether or not the car is efficient by your standards. For example, if mpg is greater than or equal to 22, it is efficient; otherwise, it is inefficient. Another option would be if mpg is greater than or equal to 25 and cylinders equals 4, it is efficient.
- Determine the features you want to include in your analysis (we just grabbed the pixel data from the MNIST dataset).
- Then, you'll need to split the dataset into training and test.
- Train a classifier on what is efficient.
- Evaluate your model and make any appropriate changes. Would a different classifier be better?
- Run the model on the test data.
- How did it perform? What was the precision and recall?

2. From the text…Build a spam classifier (a more challenging exercise):

- Download examples of spam and ham from Apache SpamAssassin's public datasets (https://spamassassin.apache.org/old/publiccorpus/).
- Unzip the datasets and familiarize yourself with the data format.
- Split the datasets into a training set and a test set.
- Write a data preparation pipeline to convert each email into a feature vector. Your preparation pipeline should transform an email into a (sparse) vector indicating the presence or absence of each possible word. For example, if all emails only ever contain four words, "Hello," "how," "are," "you," then the email "Hello you Hello Hello you" would be converted into a vector [1, 0, 0, 1] (meaning ["Hello" is present, "how" is absent, "are" is absent, "you" is present]), or [3, 0, 0, 2] if you prefer to count the number of occurrences of each word.
- You may want to add hyperparameters to your preparation pipeline to control whether or not to strip off email headers, convert each email to lowercase, remove punctuation, replace all URLs with "URL," replace all numbers with "NUMBER," or even perform stemming (i.e., trim off word endings; there are Python libraries available to do this).
- Then try out several classifiers and see if you can build a great spam classifier, with both high recall and high precision.