CS244 Exercise 3

Task 1: Categorical Features

Most machine learning algorithms are designed to use numerical data (e.g., integers, real numbers, or boolean 0/1 values). But what if we have categorical features that take on one of a few discrete values that do not correspond to numbers? For example, perhaps one of our features is whether someone has taken CS305, so that the values for each person for this feature are either "False" or "True". Or maybe one of our features is whether someone likes chocolate or not so that the values are either "No" or "Yes". Or perhaps we have data on movies and one of the features is the genre of the movie, which could be "Action", "RomCom", "Documentary", or "Horror" for each film. When a feature takes on one of two different values, like the feature indicating whether someone has taken CS305 or the feature indicating whether someone likes chocolate, normally we convert the feature values ("False" and "True", "No" and "Yes") to boolean 0 and 1 values in order to convert the non-numerical data into numerical data. However, when a categorical feature takes on more than two values, such as the genre of a movie, which in the example above contained four categories ("Action", "RomCom", "Documentary", and "Horror"), normally we remove the feature and replace it with *T* new features where *T* is the number of categories, i.e., for the movie genre example above *T* would be 4. The four new features in our movie example would correspond to isTheMovieGenre_Action? (0 or 1), isTheMovieGenre_RomCom (0 or 1), isTheMovieGenre_Documentary (0 or 1), and isTheMovieGenre_Horror (0 or 1). This is known as a *one-hot* encoding because exactly one of the four genre features should take on the value of 1 for a movie and the other three genre features should take on a value of 0.

Suppose we have data on movies. Our data contains five features: the year the movie came out (an integer), the length of the movie in minutes (an integer), whether the movie is suitable for children (text values of "no" or "yes"), the genre of the movie (text values of "Action", "RomCom", "Documentary", and "Horror"), and the average rating given to the movie out of five stars (a decimal number between 0 and 5).

If we converted non-numerical features to numerical features as described above, what would the movie vector (2018, 120, no, RomCom, 4.2) be converted to?

If we have a feature corresponding to four non-numerical categories, a one-hot encoding would replace the feature with four new features each taking on values of 0 or 1. As an alternative to using one-hot encoding, we could just keep the one feature and convert the non-numerical values to numerical values, e.g., "Action" could be replaced by 0, "RomCom" could be replaced by 1, "Documentary" could be replaced by 2, and "Horror" could be replaced by 3. This alternative approach is not commonly used.
What is one disadvantage of this alternative approach?

Download the Jupyter Notebook for Exercise 3 from the course website. Open the Notebook in your web browser and work through it. As you work through the Notebook, answer the following questions.

Task 2: Bag of Words

In the matrix X above, why are the first two rows identical?

In the matrix X above, what does the 2 in the third row correspond to?

In the 2x17 matrix above, the first row contains three non-zero values (one 2 and two 1's). What features do these three non-zero values correspond to?

What feature has the lowest weight? Why?

Task 3: Sentiment analysis of Twitter data

How many different features were extracted from the corpus?

Which classifier yielded the highest accuracy on this dataset and what is this classifier's accuracy?

When creating the Perceptron classifier, it was given an argument of `max_iter=20`. What does this correspond to?

When using 2-grams, how many different features were extracted from the corpus?

When using 2-grams, which classifier yielded the highest accuracy and what is this classifier's accuracy?

Task 4: To spam or not to spam: that is the question

How many different features were extracted from the corpus?

Which classifier yielded the highest accuracy on this dataset and what is this classifier's accuracy?

What are the five words with the lowest weight? What are the five words with the highest weight?

Do the three classifiers classify the message as ham or spam? Do the results for any of the classifiers change when executed multiple times on this message?