Capstone 2 Milestone #2 Report

Machine Learning

The first test I ran is a logistic regression to predict the wine quality category. Using the scikit-learn package, I first used the CountVectorizer to create a vector representation of the sommelier description. Then, I created a training set and a test set with train_test_split, with the features of the vectors as the X, and the wine quality category as the y. After fitting the logistic regression model to the training set, I obtained the accuracy score of the test set, which was decent at about 71%. To test the model, I obtained probability estimates of the first wine in the dataset, which was in the "ok" quality category. The model predicted with 73% probability that this wine would be in the "ok" category.

I also ran this same model, but to instead predict the grape variety. Using the same steps, this logistic regression model yielded an accuracy score of about 68%, slightly lower than the accuracy score when predicting quality. The probability estimates on the first wine in the dataset, which is a Portuguese Red variety, was about 48% for Portuguese Red and 35% for Bordeaux-style Red Blend, so while this is a less effective prediction than wine quality, it is still accurate on this particular wine.

Recommendation Tools

The first recommendation tool I built utilized the TfidfVectorizer and linear_kernel tool from the scikit-learn package. I created a TfidfVectorizer object and fitted it to the description column of the training set. I only trained on 5% of the data in this case due to the amount of memory required to train on more. I then used the linear_kernel tool to create a matrix containing the cosine similarities of each description. I populated an empty dictionary with the wine ID number as a key, and a list of similar wines based on the cosine similarities of the sommelier descriptions. Then, I wrote simple functions to extract the wine ID number from the dataset, as well as give a list of recommendations of different wines based on the given ID number.

The second recommendation tool I built utilized the K Nearest Neighbors algorithm. I dropped all of the columns from the wine dataset besides the province, variety, points, and price. I then made a pivot table from the dataframe with the variety as the index, the province as the columns, and the points and price as the values. I then made a compressed sparse row matrix with the scipy.sparse package from the wine pivot table, and fit a nearest neighbors model to the matrix. Using 10 as my number of neighbors, I chose a random wine from the pivot table and output a list of recommended varieties based on the variety of the randomly chosen wine.