TECHNICAL REPORT

Topic

The topic of my project is "What is the ultimate price of fame? A look at whether or not being famous shortens your lifespan, and what factors may contribute." My original topic had been limited to only looking at musicians' lifespans, but that dataset was very limited, so I had to expand my parameters and find a broader set of data that I could use.

Acquiring the Data

My original dataset was going to be used from scraping Wikipedia pages strictly devoted to musicians' deaths. But when that dataset turned out to be too small to be useful I instead turned to a dataset available on Kaggle for Celebrity Deaths. This dataset was scraped from Wikipedia and contained every dead celebrity, their reason for fame, their age at death and their cause of death from 2006 - 2016. Nationality and fame score (number of citations on Wikipedia) were also included. This original dataset contained 21,458 total observations.

Preprocessing, Cleaning and Feature Engineering

The original dataset contained 21,458 total observations. However, the feature column for fame_score only contained 19,852 values and the feature column for cause_of_death only contained 8,974 values. Since both were important features that I wanted to use in order to try and predict lifespan and whether or not celebrities die younger than the general population, I had to drop a lot these rows that did not contain any values. This left me with 8,408 fully populated observations.

The famous_for feature column contained a lot of parenthetical extraneous data that didn't seem relevant so I stripped those out, so I would only be left with a more standardized core reason for the person's fame. The same held true for the cause_of_death feature column so I followed the same process to come up with a more standardized core cause of death for each person. I then also used PorterStemmer to further standardize these text features along with nationality. As a final cleaning step, I dropped any potential duplicate rows leaving me with 8,375 cleaned, fully populated, non-duplicate observations.

I had to bring in additional data for average age of death for the general population from the World Health Organization's Mortality Database from 2006 – 2016 to determine a baseline average death age. I used this baseline to create a new binary feature for died_young which was set to 1 or 0 if a celebrity died younger than the baseline age. After observing the most frequent causes of death I created additional columns for these most frequent causes that would be flagged true if the celebrity died for one of those reasons. I created additional columns for each continent that took values from the nationality column and compiled each celebrity's nationality into its appropriate continent. As a final step, since I had a column for the month that each celebrity died, I thought it might be interesting to create an additional feature

for season to see if there might be any general correlation between seasonal death rates (i.e., do people die more often in Fall and Winter than Spring and Summer?)

Modeling Process

Since most of my data was textual, I decided to focus on classification models. Before beginning preliminary modeling, I established a baseline accuracy score for my died_young column since that was going to be the primary thing I was going to try and predict. This baseline accuracy level came in at 60% so my models would need to predict died_young better than 60% to be worthwhile. I used an evaluate_model function to get a general idea of which classification models might perform best before trying to tune any hyperparameters. I ran sample models using RandomForestClassifier, DecisionTreeClassifier, ExtraTreesClassifier, BaggingClassifier, LogisticRegression, KNeighborsClassifier, GaussianNB, BernoulliNB and MultinomialNB. My best results across different features were seen using MultinomialNB and RandomForestClassifier. Each of these models came in with accuracy scores around 72%. But ultimately, I chose to focus on the RandomForestClassifier model since it has a lot more hyperparameters to tune than the MultinomialNB model. I used both RandomizedSearchCV and GridSearchCV on my RandomForestClassifier model to see if I could get my accuracy scores up a bit. Using the GridSearchCV methods for best_params_, best_score_ and best_estimator_ I was able to get my RandomForestClassifier model's ROC curve score up to 0.740 and my MultinomialNB model's ROC curve score up to 0.756.

Evaluation / Future Steps

Overall, I was able to create models that performed 14% - 15% above baseline accuracy. More extensive tweaking of hyperparameters may have possibly been able to get those scores a little higher, but I saw diminishing returns once I hit around 75%.

Since I had to drop so many observations from my original dataset, in the future I might look for a more complete set of data or scrape my own. I thought it might be interesting to create a function that would allow a user to input different parameters related to celebrity and then predict age of death based on my models, but I simply ran out of time to create such a function. Maybe in Capstone version 2.0!