Cardiovascular Death Rate Capstone: Project Report

Zach Wimpee

November 8, 2020

1 Proposal

1.1 Motivation and Problem

Heart disease has been one of the leading causes of death in the United States for decades.[1] An abundant variety of resources are available to help people understand their individual level of risk of cardiovascular health issues, as well as guidelines for reducing that risk. However, given the frequency, severity, and overall scale of cardiovascular health issues in the United States, a more macroscopic analysis of this public health crisis might provide some insight into less obvious contributing factors, and also how these factors could be addressed. To put this more succinctly;

If population demographic data can be used to model trends in cardiovascular death rate, what recommended course of action would the model suggest to address cardiovascular health issues, if any?

1.2 Potential Clients

The target primary audience for such a project would be individuals or organizations capable of introducing policies and implementing public health initiatives aimed at addressing any large-scale contributing factors identified by this work.

1.3 Data

Fluctuations in the annual rate of cardiovascular related deaths can be seen when examining data at the state and county level. Social and economic data are publicly available for these subsets of the national population from United States Census Bureau data bases.[2]

1.4 Approach

If a data set is constructed from a combination of CDC and Census data, **can a machine-learning model be built to predict changes in the rate of cardiovascular related deaths at the county level?**

[1] https://www.cdc.gov/nchs/fastats/deaths.htm
[2] https://www.census.gov/data/developers/guidance/api-user-guide.html

1.5 Deliverables

The primary result of this project would be a recommended course of action to address national cardiovascular health issues on a large scale. This recommendation would be based on possible demographic and population factors for poor cardiovascular health, and the potential significance of these factors would be supported quantitatively by resultant data obtained from a machine learning modeling approach.

2 Data Wrangling

2.1 Overview

The primary goal of this project is to build a model that predicts county level changes in annual cardiovascular death rate. To accomplish this, a data set is needed that contains county level information on population demographics and cardiovascular mortality counts over a range of years. This document describes the construction of such a data set using the U.S. Census Bureau API and the CDC WONDER databases. The associated code is available on GitHub

2.2 CDC Data

The first thing needed to construct this data set is county level cardiovascular death rate data over some range of years. The CDC WONDER online databases provide this information through a convenient interface. Using the interface, a request was made for cardiovascular death count for every county in the United States from 2010 through 2018. The response was a tab-delimited text file in which each row corresponds to a single county for a given year, and records both the county population and total cardiovascular related mortality count. This file is saved as `data/cdc_data.txt` in the GitHub repository linked above.

2.3 Combining with Census Data

The CDC data provided the information used to calculate the crude cardiovascular death rate (CDR), and equivalently the change in county CDR between years. Data is needed that can serve as the feature space over which CDR change will be predicted. The U.S Census Bureau has a number of APIs that can provide social and economic data at the county level. Two sets of variables were requested for all counties from 2011 through 2018 using ACS 1-year Comparison profiles data. The resulting data sets were then combined into a dataframe, and saved as `data/cdr_data.csv`. These procedures were performed in the notebook `data_wrangling.ipynb`

3 Statistical Analysis

Recall, **the primary goal of this project is to build a machine learning model that predicts the change in cardiovascular death rates (CDR) at the county level.** Some insight was gained from the data story about possible relationships between change in CDR (the dependent variable) and a set of census variables (independent variables). Statistical analysis techniques can be used to further explore these relationships, and can help determine if the data are appropriate for building a predictive model.

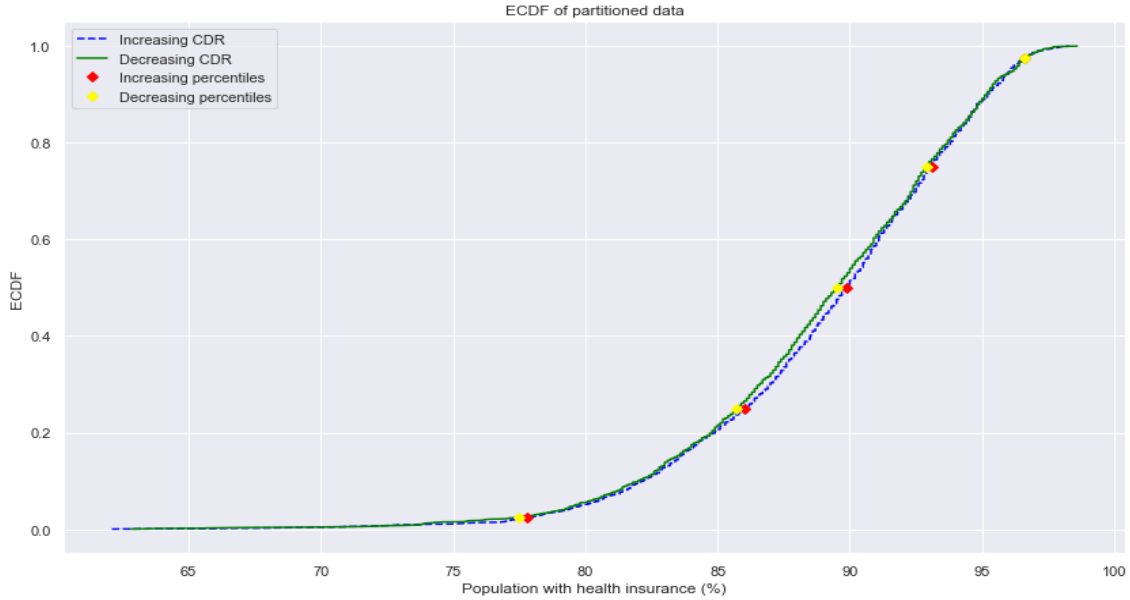
3.1 Null Hypothesis

Partition the data into two sets: One being the set of counties that experience an increase in CDR from the previous year they appear in the data, and the other being the set of counties

that do not experience an increase in CDR. Both sets have the same set of feature columns (census variables). **I hypothesize that the distribution of each feature is identical between these two data sets.**

3.1.1 Example Feature: Percent with Health Insurance

Consider the distribution of a single census variable, *percent of population with health insurance*. Below is the ECDF plot for the data partitioned by CDR change.



3.2 Sample Statistic

The null hypothesis is rejected if any feature has different distributions for the partition. To compare the feature distributions, the **sample statistic** T^F for feature F can be defined as:

$$T^F = \overline{F^{incr}} - \overline{F^{decr}}$$

where,

- $\overline{F^{incr}} \equiv$ Mean feature value for set of CDR increases
- $\overline{F^{decr}} \equiv$ Mean feature value for set of CDR decreases

For $F \equiv$ *percent of population with health insurance*, the observed value of this sample statistic is $T^F_{obs} = 0.211$. How reasonable is this value under the assumption of the null hypothesis?

3.2.1 Bootstrap Replicates

A new partition $\{\widetilde{F_i^{incr}}, \widetilde{F_i^{decr}}\}$, $i \in Z^+$, can be constructed by generating replicates of partition $\{F^{incr}, F^{decr}\}$ under the assumption of the null hypothesis. The sample statistic T_i^F can be calculated on this replicate:

$$T_i^F = \overline{\widetilde{F_i^{incr}}} - \overline{\widetilde{F_i^{decr}}}$$

Repeating this process N times results in a set of values for the sample statistic:

$$T^F_{bs} = \left\{ T_i^F \middle| i = 1, 2, \ldots, N \right\},$$

This set describes the distribution test statistic bootstrap replicates under the assumption of the null hypothesis. How does T^F_{obs} compare to this distribution for $F \equiv$ *percent of population with health insurance*?

3.3 p-value: Feature - Percent with Health Insurance

The p-value, denoted p, of T_{obs}^F for feature $F \equiv$ *percent of population with health insurance* is computed by:

$$p = \frac{|T_{bs}'^F|}{N}$$

Where,

$$T_{bs}'^F = \left\{ T_i^F \in T_{bs}^F \middle| T_i^F > T_{obs}^F \right\}$$

For $N = 10,000$, the p-value is $p = 0.0909$, and therefore feature $F \equiv$ *percent of population with health insurance* does not reject the null hypothesis.

3.4 p-value: Other Features

Repeating the process described above on the other features ...

- *mean household Social Security income*: $p = 0.0001$
- *SMOCAPI less than 10 percent*: $p = 0.0002$

These features result in p-values that suggest rejection of the null hypothesis. There are numerous other features that also reject the null hypothesis, but even just one such feature would suffice.

4 Model Building

The aim of this project is not only to build a predictive model for changes in cardiovascular death rate, but also to recommend a course of action from its conclusions. The Scikit-learn **DecisionTreeClassifier** seems like a natural choice, since such a model would provide a greater level of interpretability than, say, a SVM classifier. By looking at the splitting criterion of the internal nodes, it should be easy to see what areas could be addressed to prevent increases in cardiovascular death rates. However, overfitting is a known limitation of Decision Tree classifiers. If issues with overfitting are encountered then an ensemble learner will be built from the Scikit-learn **RandomForestClassifier**.

5 Decision Tree

5.1 Parameter Tuning

The Scikit-learn **DecisionTreeClassifier** has several parameters that can have a significant impact on model performance:

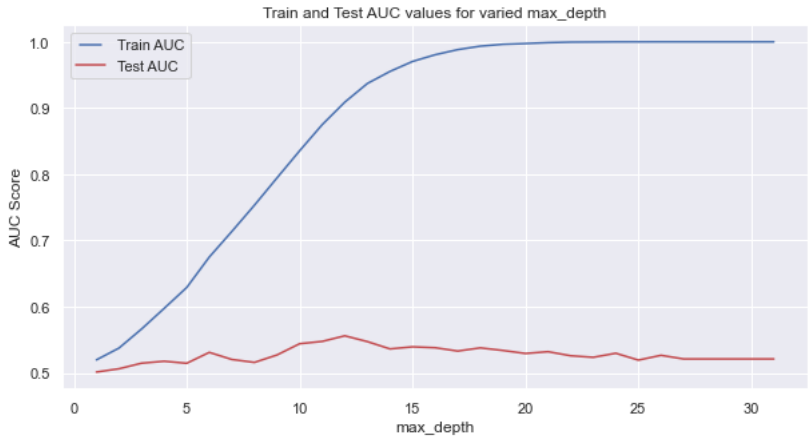
- max_depth
- min_samples_split/min_samples_leaf
- max_features
- max_leaf_nodes

To begin tuning these parameters, they will first be considered separately. A decision tree will be trained and tested over a range of values for each parameter. The tree's performance for each parameter value will be measured using the Area Under the Receiver Operating Characteristic curve (AUROC) for both the training and testing sets. This metric is chosen because it is a good indicator of how well the model can distinguish between the two classes. The train

and test set AUROC scores will then be plotted together versus the corresponding parameter values. These figures will then be used to determine the range of values for each parameter to use in a grid search cross-validation of parameter configurations.

5.1.1 max_depth

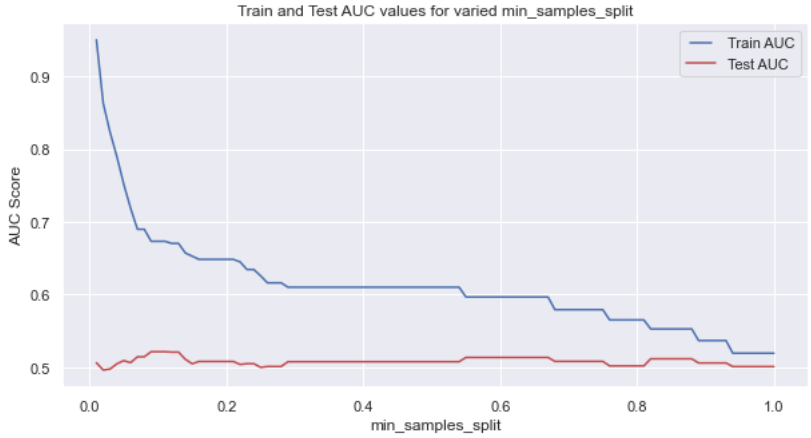
As its name suggests, this parameter controls the maximum depth for which to grow the tree. For example, setting $max\_depth = 1$ would result in a tree with a single split into two leaf nodes. Too high a value for this parameter can lead to high variance and will overfit the training set. Too low a value can lead to high bias and will underfit the training set.



Unfortunately, generalization to unseen data is poor across the board regardless of the value chosen for this parameter. Even more unfortunately, **this issue will continue to be seen throughout the parameter tuning procedures, and can likely be attributed to poor predictive power of the constructed feature set.** However, for the purposes of this project we will work with what we have and simply keep in mind what may have gone wrong or what should have been done differently.

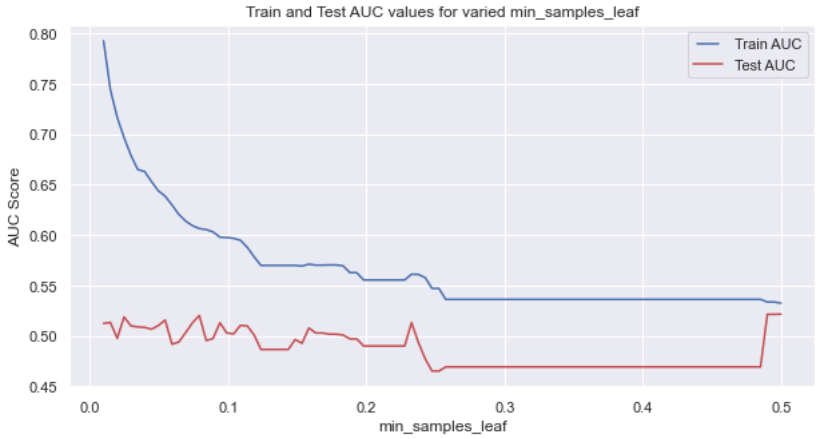
5.1.2 min_samples_split

This parameter designates the minimum number of samples required to split an internal node. It can be either an integer value greater than 1, or a floating point value for a fraction of the total number of samples.



5.1.3 min_samples_leaf

This parameter sets the minimum number of samples that must be at each leaf node. This means that an internal node can only be split if each resulting leaf node contains at least this number of samples. This parameter can be either an integer greater than 0, or a float value.



5.1.4 max_features

This parameter controls the number of features to consider at each node when looking for the best split. It can be an integer value, a float value, or one of three possible string values corresponding to an operation on the total number of features. If $max\_features = None$, then the value is set to the total number of features.

5.1.5 max_leaf_nodes

This parameter sets the maximum allowed number of leaf nodes.

5.2 Grid Search Cross Validation

Based on the above plots, the following set of parameter values will be searched over to find the optimal parameter configuration.

- max_depth: 10, 11, 12, 13, 14
- min_samples_split: 10 values between 0.10-0.15
- min_samples_leaf: 10 values between 0.05-0.10
- max_features: None, 'auto', 'sqrt', 'log2'
- max_leaf_nodes: 55, 56, 57, 58, 59

This grid of parameter values was searched for the best configuration using *roc_auc* as the scoring criteria. As mentioned previously, the plots suggested poor generalization to unseen data, so unsurprisingly even the optimal parameter configuration resulted in an *roc_auc* score of 0.495 on the test set. Now it needs to be determined if this poor performance is due to the tendency of decision trees to overfit, or if the feature set we have constructed is lacking any predictive power.

5.3 Feature Selection and Decision Tree Results

As a first step, the feature importance values of the tuned model can be examined. Using the ranked feature importance values to train a new decision tree on a reduced feature set might provide a clue to where the issue truly lies. This procedure resulted in an *roc_auc* score of 0.526 on the test set, a rather insignificant improvement.

The data will now be used to attempt to train a random forest classifier. If this also results in poor performance, it seems reasonable to suspect that the issue lies within the data set itself, and new features should be acquired in order to build a model with any predictive power.

6 Model Comparison and Results

6.1 Results

Repeating the above procedures again for a random forest classifier resulted in a test set *roc_auc* score of 0.527 on the full feature set, and 0.523 on the reduced feature set. This is essentially a worst case scenario.

6.2 Conclusion

The goal of this project was to build a predictive model for county-level changes in annual cardiovascular death rates, and then use the model to provide recommendations for preventing increases in this rate.

The first attempt to train a model with a decision tree failed due to overfitting, with minimal improvements seen even by reducing model complexity. A second attempt to train a model using an ensemble learner also failed. This suggests an inherent issue with the data itself, particularly that the features have little to no predictive power regarding changes in cardiovascular death rate.