horizontal line

**Shawn McClain**

In-Depth Analysis

**3/4/2020**

Building from my statistical analysis, I had a pretty good idea as to which variables were most predictive for suicide rates thus far. Latitude and divorce rates seemed to be strong predictors from my bootstrapped regression models as well as year and mobile subscriptions with a high r coefficient for year. But, for my prior analysis, my data had been aggregated by year and country separately. In order to use my data to perform machine learning, I needed to disaggregate my data to provide more data points for train-test-splits and predictive modeling.

My first step was manipulating my dataframes by taking a previous dataframe with all of the columns to use more variables for prediction, and melting all of my dataframes into one. Once my machine learning dataframe was cleaned, melted and merged properly, I wanted to get an overview of good predictors with StatsModels Ordinary Least Squares package. Using just divorce rates, I fit a linear regression to my target (suicides per 100k). With only divorce rates, my model accounted for an R2 value of 0.185 and an F-Statistic of 71.92. This model confirmed to me that divorce rates were a significant predictor by alone being able to account for 18.5% of the variance.

Building off of my first OLS model, I tried adding a number of other variables and getting the summary of my OLS model. Some predictors had p-values that were greater than 0.05, so I filtered those features out of my model because they diminished the model’s F-Statistic and weighed down the model’s significance. Using mobile subscriptions per 100k, year, divorce rate, latitude, gdp per capita, and population, I constructed a model with an F-Stat of 43.38 and an R2 score of 0.455. Using these 6 predictors increased my R2 score and also decreased the Akaike Information Criterion from 2104 to 1985. All of my predictors had a p-value less than or equal to 0.001 except for year, which had a p-value of 0.04.

To check if my OLS model had any bias towards a non-linear regression, I plotted the residuals vs. the fitted values. For this graph, success is finding no patterns or relationships in the model. This is also useful to check and see if there are any outliers that are skewing the data. In my residual plot, the graph shows a slight propensity to overestimate vs. underestimate, but the data looks to be scattered well around the graph, indicating a linear model.

Another tool for testing linearity is with a Quantile Quantile plot. If our model is linear, we’d expect the residuals to be normally distributed, and follow the quantiles of a normal distribution corresponding to their value. A successful result is when our residuals are hugging the QQ line. In my graph, the residuals do hug the QQ line for the most part with only one or two outliers.

To mitigate against high leverage outliers that can skew our regression, I used an Influence plot. This plot also uses the residuals and plots the studentized residual value vs. the leverage of the point. High leverage points are points that are unusual to the bulk of the distribution, but not necessarily highly influential. High influence points have a relatively high leverage and a relatively large residual value. In the plot, these high influence points are indicated based on their relative bubble size. Looking at my Influence plot, there were some points with high influence, but looking deeper into the points did not indicate that these points were invalid or errors in the data.

After using the Stats Models library, I wanted to try some different methods with the Sci-Kit Learn library. For this model, I included the variables for my model because I planned on using regularization models to tune my model. I divided my predictors into the X data and my target variable (suicides per 100k) into Y. I used a train-test-split on X and Y with a test size of 0.3. This allowed me to test unknown variables on my own training data, from the same sample. My initial R2 score when tested off of the training model was 0.5542, and improvement from the Stats Models OLS model from before. My mean squared error for this model was 22.2319.

Using seaborn’s lmplot, I plotted my predicted Y values vs. my actual Y values. In a perfect model, we would expect a 1:1 ratio between predicted values and my actual values. However, in my model, below 18 suicides per 100k, my model tended to underestimate suicides and over 18 suicides per 100k, model tended to overestimate suicides.

To improve my model, I used some regularization algorithms. I fitted my training data with a Lasso Cross-Validation model. This model penalizes predictors that don’t help the fit of the model with small coefficients and minimizes them towards 0 in the OLS regression. In turn, this shrinks my model and leaves the best predictors. Using the lasso regression and tuning the alpha parameter for regularization, I increased the R2 to 0.61417 and decreased the MSE to 19.245.

I also tried a Ridge Regression, which is a similar relative to the Lasso Regression. In Ridge Regression, the coefficients are squared and multiplied by penalty, then added to the OLS. This regularization technique rewards models with a lot of good predictors, because generally, coefficients will not be set to 0. Using the ridge regression and tuning the alpha parameter for regularization, my R2 improved from the bare bones model, but did not improve upon the Lasso model with an R2 of 0.6032 and a MSE of 19.791.

Finally I tried an Elastic Net model, which is a combination of the Lasso and Ridge regression. I used a grid search to sort through different combinations of tuning parameters, and coming up with my best parameters for the model. My best result for the Elastic Net model was an R2 score of 0.6136 and a MSE of 19.274. Elastic Net outperformed the Ridge model, but the Lasso model still had the highest R2 and the lowest MSE.