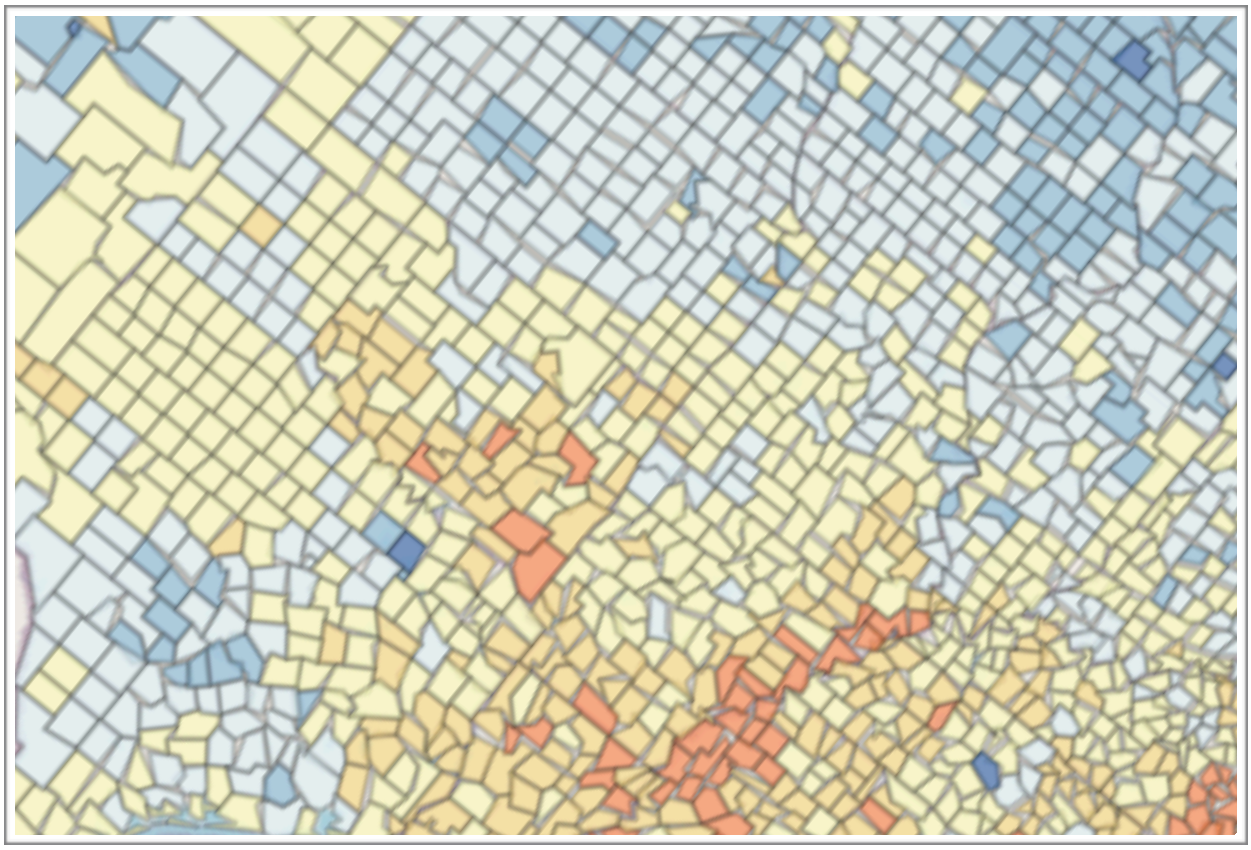
Your Neighborhood Is Killing You

Can Foursquare data predict life expectancies?

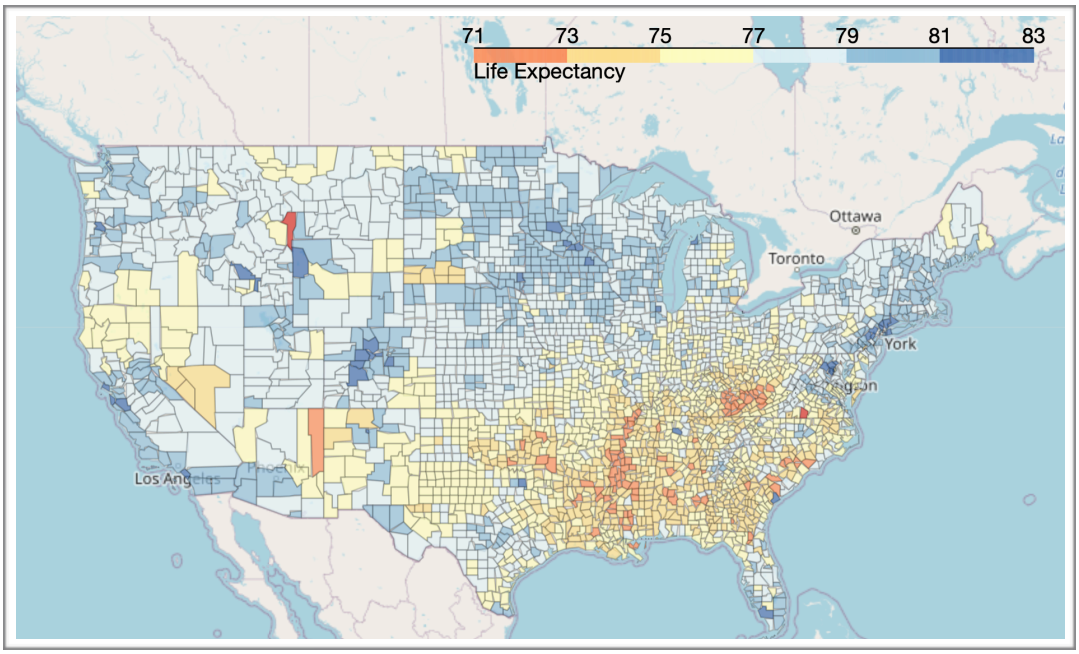


Aaron Chaiclin

May 2020

Introduction

No one lives forever. Throughout human history we have worked to increase our life expectancy and have succeeded in increasing the average around the globe.



Despite our progress, the life expectancy varies by gender, ethnicity and country. In fact, there are different life expectancies regionally within the United States.

Research by the University of Washington's Institute for Health Metrics and Evaluation (IHME) has pinpointed life expectancies at the county level. These life expectancies vary from 68.4 to 83.3 years old. This ~20% difference is a significant amount when talking about human life. In a first world country, it seems counterintuitive that there would be such disparity depending upon where you live.

For example, the life expectancy in Shelby county, Alabama is 79.15 while neighboring Talladega county is 73.3. What could cause a 7% drop in life expectancy in an area on the other side of the river?

There are many factors that play a role in how long an individual lives. Extrapolated across an entire county the factors influencing life expectancy is countless. In order to improve life expectancies, we need to understand why there

are regional differences. We can look at traditional metrics (e.g., income, ethnicity, health insurance access), but can we determine that there are other regional differences that account for higher or lower life expectancies? Do regions that have more fast food restaurants have lower life expectancies? Do more vegetarian restaurants or salad places attribute to a higher life expectancy? Do more gyms lead to better fitness and longer lives? What about more hospitals or doctor offices per capita?

Using Machine Learning we hope to create a model that can predict the life expectancy for a county primarily using Foursquare data.

If we can determine that restaurant types and the availability of specific services are correlated with life expectancies it could shed a new light on the way we think of planning for cities and counties. Certainly many government officials as well as many NGOs would have in interest in this type of study.

The Data

This project focuses on two main data sets. Life expectancy data from IHME and location data from Foursquare.

IHME has provided life expectancies at the county level, available on their website. This data set also includes fitness and obesity statistics. Although their data is a longitudinal study we are only interested in the most recent data. IHME used small area estimation methods to produce annual life tables and calculate age-specific mortality risk at the county level for the United States. De-identified death records from the National Center for Health Statistics (NCHS) and population counts from the census bureau, NCHS, and the Human Mortality Database were used in the analysis. Results of the study were published in JAMA in May 2017. This data was available for download in Excel format. In addition to the life expectancy information each county has a fitness and obesity measure. This data would be used as a comparison metric to the foursquare data.

Foursquare provides location data for various businesses, and other landmarks, through an API (API doc link). The API provides real-time access to over 105MM places available across 190 countries and 50 territories. Foursquare refers to this as "venue" data. Foursquare categorizes the venues and provides geo coordinates for each location. The API will provide up to 50 results (per call) within a specified radius of a geo location. Since counties are not circular shaped, we need to calculate a big enough radius to capture all the venues and then provide a secondary process to verify that the geolocation is within the bounds of the county.

The Foursquare categories will be used to classify the venues, which will be aggregated at the county level. Since this project is limited in the amount of data we can gather from Foursquare, due to API and licensing limits, we are limit the data pulls to specific "seed" categories. The categories are listed below.

Seed Categories

fast food	bike trail	juice bar	ski area
fried chicken	park	gluten free	trail
pizza	pedestrian plaza	salad place	factory
hot dog	playground	vegetarian	industrial estate
fish & chips	rec center	gym	medical center
bbq	rock climbing	smoke shop	bar
er	spiritial center	hospital	gun shop

The seed categories were selected based on venue types that provide access to fitness (e.g., gyms & trails) or medical care, imply a certain diet (e.g., fast food or vegetarian), or be a potential health hazard (e.g., factory). Although these are the seed categories, we do not limit the actual categories returned from the API, as the data only shows the "primary" category for the venue.. For example, we may request the category "pizza" and receive a restaurant who's primary category is "italian food". In this example, we would then use the category of "italian restaurant" for that venue, not pizza. No presumptions have been inserted into the model, for example we might expect a correlation between medical access and longer life expectancies, but if the model finds a negative correlation, then we accept the truth of the data as is.

In addition to our two main datasets, we will import GeoJSON shape files (available from from a fellow data wrangler, Eric Celeste) and population counts for the counties, from the US Census. The GeoJSON data is for location validation and the population data is to provide per capita statistics of the venues, in addition to the raw counts.

Data Cleaning

There are a number of challenges in working with these data sources. Whenever working with multiple sets of data there is typically a challenges of getting the data to fit together. In this section, we'll cover how we joined these sets together and then cover each dataset's specific challenge.

When working with US county data, there is a standard unique ID, called a FIPS code, that can be used to reliably join the data when it is available from the dataset. Of course, our life expectancy source did not provide a FIPS code. FIPS codes were scraped from the wikipedia. A unique key was created by concatenating the state abbreviation and the county name. After capitalizing the code and removing spaces and periods, the data joined fairly well, connecting all 3,142 counties. Six counties required that we manually match the codes due to name variations (e.g., "DC-DISTRICTOFCOLUMBIA" vs "DC-WASHINGTON). Nine counties needed to have FIPS codes manually added because they were not in our source data. We were able to get FIPS codes for all 3,142 counties.

GeoJSON data was easily join via a FIPS code, once a prefix was added to match the GeoJSON file format.

The list of counties was used in our API calls to get the venue data and we imputed the county FIPS code into the JSON result. This allowed us to query the county shape file when validating the assigned county before aggregating the data.
It was difficult to find that county shape files that would work for generating choropleth maps as well as validating the venue locations. Once a reliable source had been found, the maps worked easily and the shape files were added to a GeoPandas DataFrame.

We were able to capture 241,701 venues from Foursquare which was subsequently reduced to 199,511 when validating that the venue was within the county shape.

Population data, which was retrieved from the US Census website, included 11 years of population broken down into different age groups. We simply selected the

total population for most recent year for this study. Only 3,138 counties merged cleanly with the population data. We manually adjusted some FIPS codes to better align the data allowing us to match all 3,142 counties.

As the data moved through the various processes we increased our number of counties by one but Foursquare only had data for 3,126 counties. In the end, we had enough data for 3,126 counties out of 3,143**.** Which accounts for 99.98% percent of the total US population.

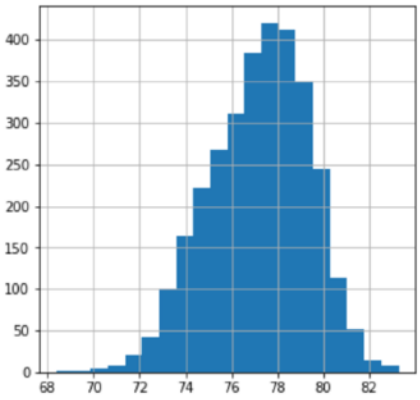
Feature Selection & Development

The aggregated Foursquare data provided for 463 different features. In analyzing the raw features, we discovered that no one particular feature has a high correlation with the life expectancy. The max correlation was only 0.35 as shown below in the top 10 features by correlation.

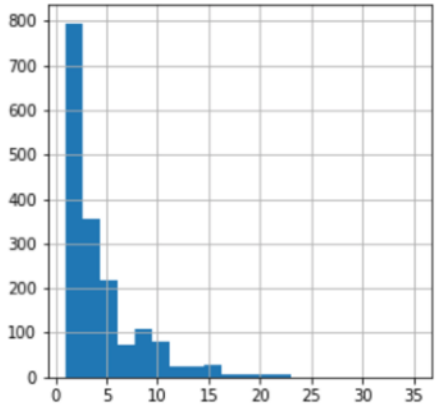
Feature	Pearson Correlation	Absolute Value
Bar	0.346542	0.346542
Park	0.313285	0.313285
Trail	0.312461	0.312461
fastfoodPerCap	-0.262772	0.262772
Pub	0.232082	0.232082
Playground	0.228892	0.228892
totalPop	0.205338	0.205338
Sports Bar	0.203747	0.203747
American	0.199809	0.199809
barPerCap	0.198616	0.198616

We eliminated 186 features due to sparsity of data, where 5 or less counties contained this feature and the max count was 1. In analyzing the remaining 277 features, we determined that the features did not align will with the target life expectancy data. The life expectancy data looks like a skewed normal curve, while our features did not.

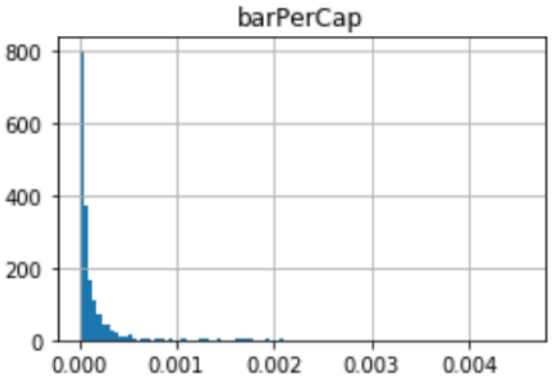
Life Expectancy Histogram

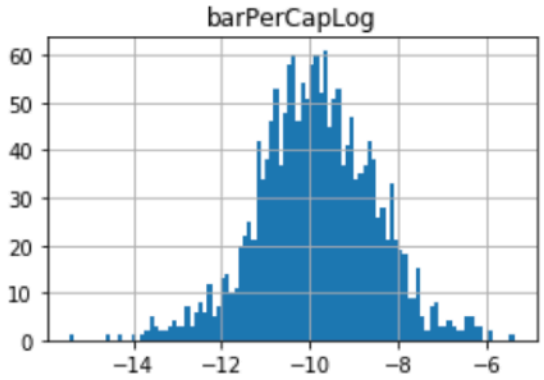


Bar Feature Histogram



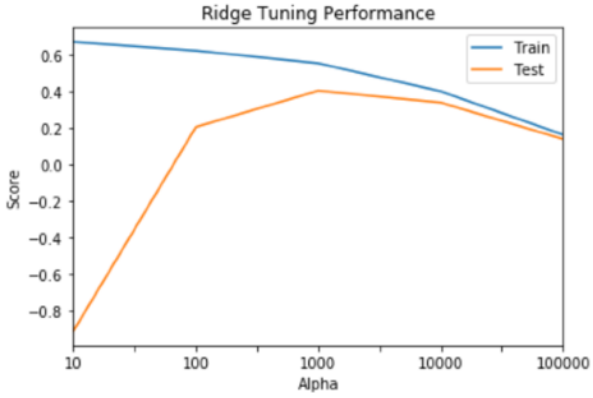
We took the remaining 277 features and applied county data creating PerCapita and PerSqMi versions of the features. We also created a set by taking the log of the "per" data. By combining the county data with the Foursquare data, we created 1,385 different features. Below is the histogram of barPerCap and barPerCapLog. You can see that the Log better shadows the target data.





In order make sure were were generating accurate results, we cut two different subsets of data and then ran them through our models to pick the best performing. We used two different methods for creating the subsets. SciKit Learn's RFE (recursive feature elimination) process was used to create a dataset 200 features. We used a combination of Pearson correlation and value frequency across counties was used to determine our top correlating features. We ranked the features by absolute value and created a subset with 200 features.
The information provided for a sparse matrix of data, as 30,350 elements have a value of one and 27,986 elements have a value greater than one while 803,965 elements have a value of zero. The sparsity is calculated as 0.9323.

It will be difficult to create an accurate prediction with a sparse matrix, but we'll put our regression models to the test and see how close we can get.



Prediction Methodology

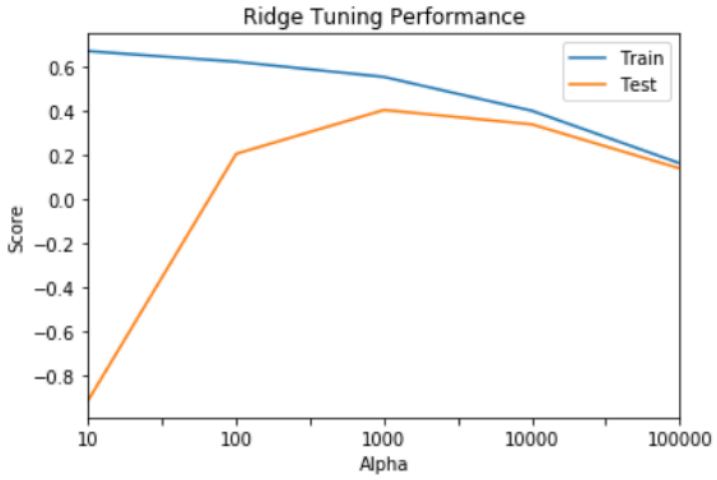
Overall, we wanted to follow an iterative process for developing our final prediction model. The steps we took were:

1) Initial feature development
2) Feature analysis
3) Additional feature development
4) Simple linear regression model
5) Test & training data set development
6) Iterative tuning for Ridge regression
7) Additional data set development
8) Iterative tuning for RidgeCV regression (with cross validation)
9) Iterative tuning for LassoLarsCV
10) Iterative tuning for MLPRegressor (Neural Network)
11) Final model selection & analysis

Once our data was ready and our initial set of features were created we did a basic linear regression to see how the model was handling the data. Although the model showed a validation score of 0.75, the model was, unsurprisingly, overfit to the data.

Next we cut training and test data to use with Ridge regression. This model allowed us to better fit the data by applying a regularization parameter. In order to tune the hyperparameters, we loop through various configurations and store the results in a dataframe. This allowed us to chart the results and find the optimal configuration. This was designed to allow for multiple iterations with manual tuning in between each iteration.

With Ridge we were able to get our test validation to 0.406756. When we added regularization the test scores increased while the training score decreased. This shows that regularization is working and finding a better fit. As we applied too much regularization we saw both scores drop. This also shows the limit of how much regularization alone can help.

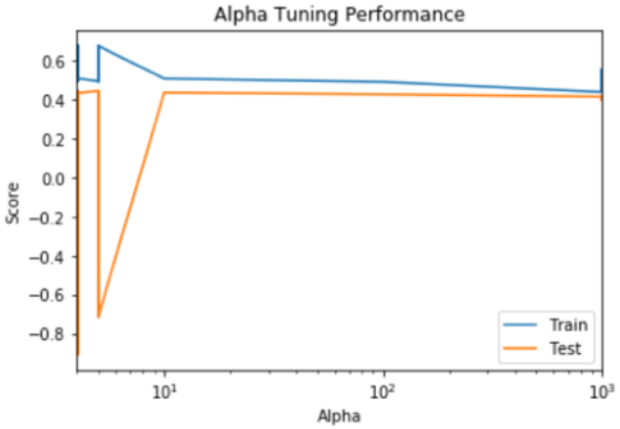


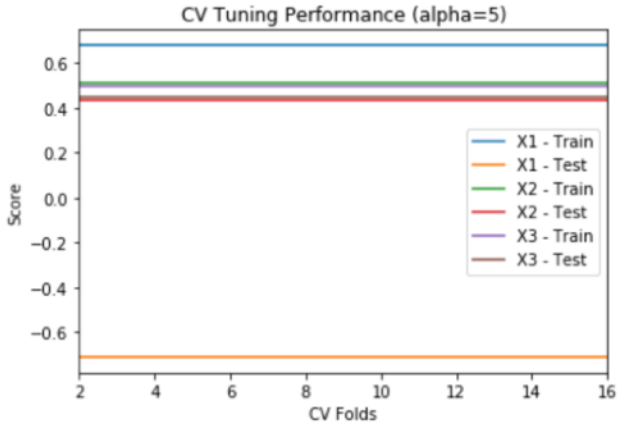
We decided to create some different cuts of data to see if we could separate the noise. We also wanted to add cross validation to help improve how well the model would generalize.

We created 3 different data sets (X1, X2, X3). X1 contained a complete set of all the features. X2 used SciKit RFE to find 200 features. RFE stands for recursive feature elimination. For this process used the Ridge regression model to estimate how important each feature is. We manually pick the number of features we want. In this case, we're using the Ridge model with our optimized hyperparameters. For simplicity sake we will use this cut of features with the other models we develop. X3 was created using the product of the correlation to the life expectancy data and the frequency with which the feature occurs across all the counties. This was fixed to the 200 top features.

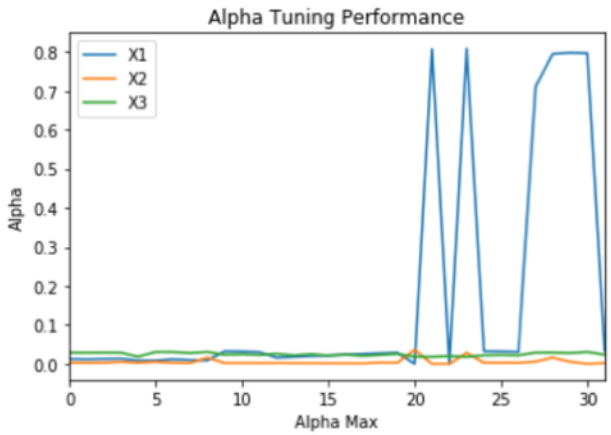
Since we were dealing with a lot of sparse data with low correlation, the models we used needed regularization to generalize the fit. We used the cross validation version of Ridge, RidgeCV, for this.

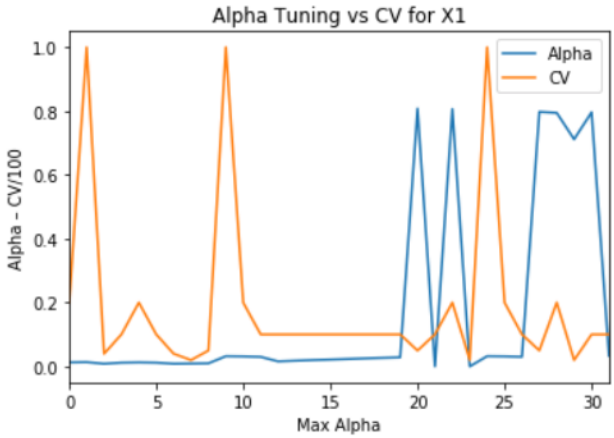
After a few rounds of multiple iterations number of iterations RidgeCV allowed us to increase our test validation to 0.446162. RidgeCV performed best of the X3 data that was cut with the Correlation X Frequency process.



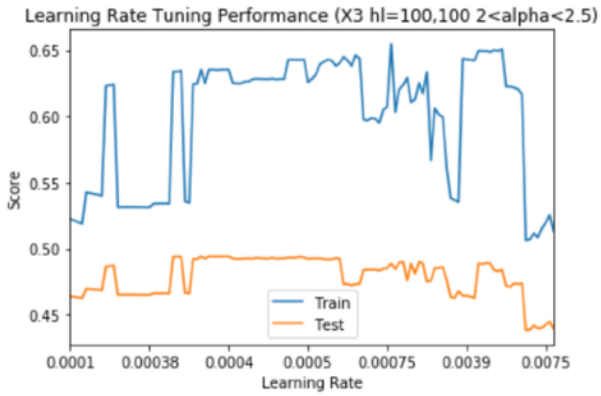


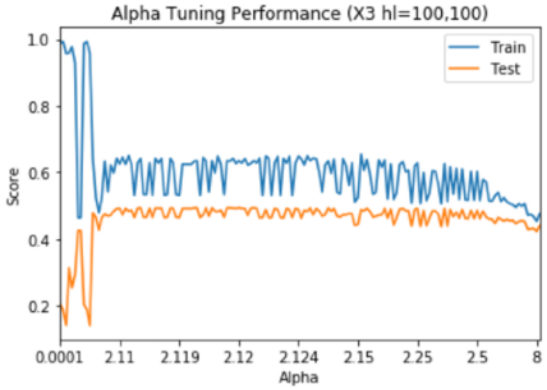
We seemed to reach the limit of RidgeCV, so we turned to LassoLarsCV. The LassoLars is known to be better at handling a sparse matrix of data. LassoLarsCV was not able to best RidgeCV, though. Its best result was a score of 0.440637 just shy of RidgeCV's score by 0.005525, not much. Interesting that LassoLarsCV chose the X2, RFE subset data, as a clear favorite. It proved to be a hard model to tune which might have been why we it did not handle the sparse matrix as well as expected. Looking at the performance graphs below, you can see that the actual Alpha did not appear to correlate to how we set the Max Alpha. It seemed somewhat correlational to a combination of cross validation folds in conjunction with the Max Alpha.



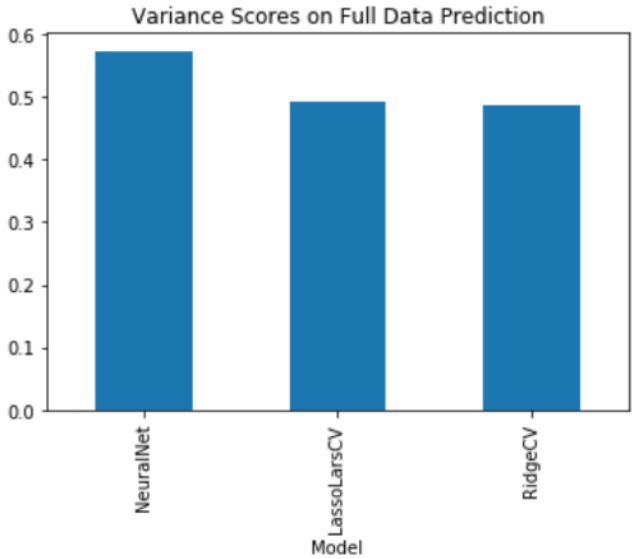


Finally, we turned to SciKit's neural network to see if this could move the needle any more. We used the Multi-layer Perceptron regressor, MLPRegressor, which seemed like SciKit's best option for our problem. Like RidgeCV, the MLPRegressor favored the X3 data. We were able to tune the neural net to get us a validation score of 0.494110. We quickly found that two hidden layers of 100 nodes was offering the best result, so we turned to the Alpha and Learning Rate parameters a the way to tune the model. Tuning the model was somewhat chaotic, as evidenced in the graphs below. We tried to get past the .5 threshold but seemed to hit a firm wall with the given data.





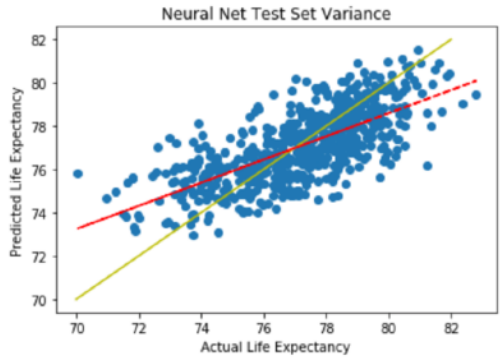
In the end we ran the three main models on the complete dataset to see how well they did. Below is a chart of the final numbers.



Results

In the end, we picked the MLPRegressor (Neural Network) to use for our final prediction. Here's how the test set prediction compared to the actual numbers:

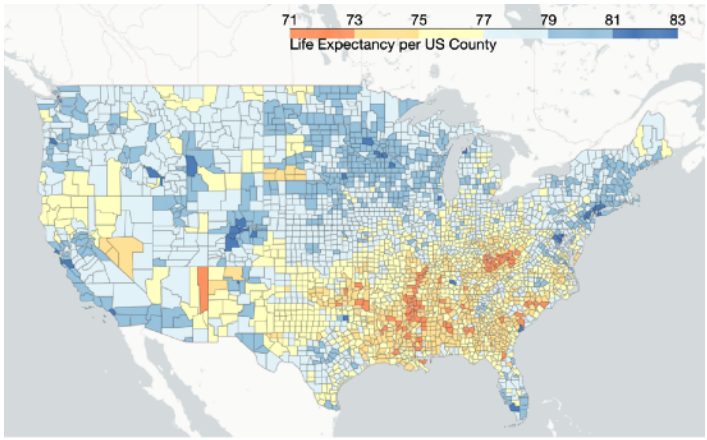
```
Variance score: 0.49
```



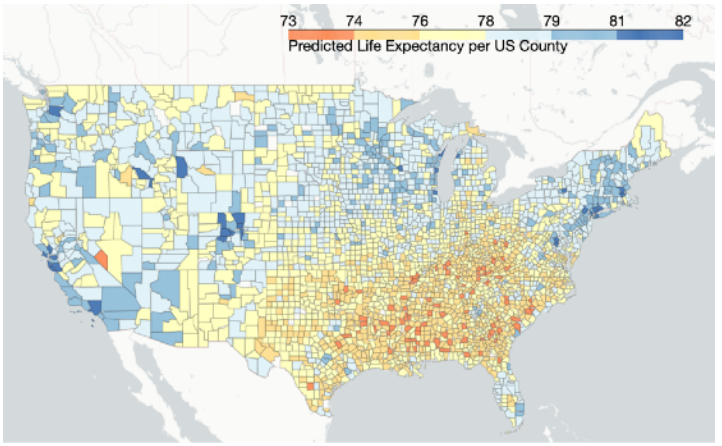

```
Yellow line shows perfect prediction.
Red line shows trend of our prediction.
```

For a more meaningful visual we created choropleth maps of the actual life expectancy vs the predicted version.

Actual



Actual



Although there are stark visual differences it is encouraging that we see consistencies across the larger regions of the us. The South does not look great in either map. New England and the West Coast have generally better outlooks. But the major cities in Texas seemed to have taken a beating in the predictions.

Discussion

The most interesting aspect of this project has been in trying to harness a sparse matrix of venue counts into something that can be useful. Although we were not able to achieve a reliable prediction, the fact that we could train the model to create results within the correct range feels like a win.

It would be worth revisiting this with a more complete pull of data from Foursquare to see if we can improve the accuracy. It would also be interesting to integrate in more census data, like median incomes and populations by age group which would likely increase the accuracy when used to develop more features.

I was pleasantly surprised as to how well the smaller feature sets did. It seems that reducing the number of features reduced the noise coming from non-correlated data. I think this does not hold true for data that is less sparse, but it opened by eyes as to how this could be used in future model development.

Ultimately, we would have liked to have built a model that could predict the changes in life expectancy as the neighborhood changes. The challenge of doing this is in finding alignment of the temporal states of the data. Our life expectancy data, census, and foursquare data was not close enough to build a truly predictive model.

Conclusion

In this study, we analyzed the ability to create a predictive model based primarily on Foursquare venue data. We compiled various venues from each US county to determine which types of venues, and their relative numbers, correlated to the life expectancy within that county. We were able to create realistic looking predictions, within the overall human age range. Due to project time limit, API access limits, and laptop CPU limits we did hit a will with our prediction accuracy. Given additional resources we think it's possible to create a more accurate model using Foursquare's venue data.