Predicting Poverty Rates for Counties in the United States

Using Python Machine Learning

Benjamin D. Hamilton

EdX DAT102x

Microsoft Professional Program in Data Science

Capstone Project

January 2018

Executive Summary

The capstone project for the Microsoft Professional Program in Data Science is to predict poverty rates in the United States by understanding the relationships between factors related to these rates and develop models to quantify them. The data are a variety of socioeconomic indicators available at the United States Department of Agriculture Economic Research Service.

This project was approached with the objective of using evidence-, expert- and hypothesis based decision making. The initial analysis and mung was performed so the first model could be deployed quickly. This was as multi-linear regression using the percent-based features. The performance was RMSE 3.22, establishing a clear baseline for more detailed analyses, munging and modelling.

Initial improvements in performance were due to methodical increases in model complexity. When no more improvements could be made, the data was reanalyzed and munged with additional features to get better performance out of the algorithms.

Learning Python was a personal goal for the author, so all work was performed in Python 3.6 using Jupyter notebooks. A great deal was learned during this work and the quality of the error measurement, CV, model selection and scripting knowledge increased as the project progressed. These processes were consistent for the best models and the limiter in model performance was the calculated error in the k-fold CV.

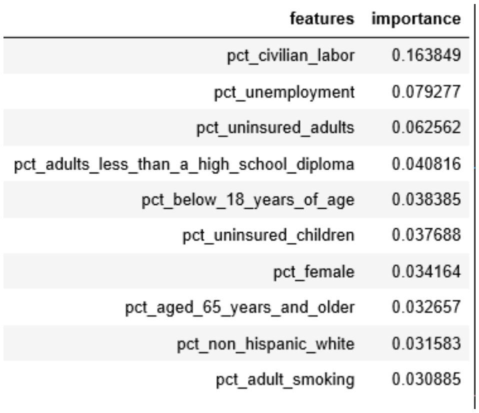
The best results were obtained using the XGBoost gradient boosted machine algorithm with features tailored towards its strengths, which included missing value handling, sparse matrices, and weak learners. **The best test RMSE was 2.69, which was worth 96/100 on the grading rubric and 91st percentile in student performance.**

The ten most important features, as measured by the XGBoost algorithm, are shown in the table. The importance listed is relative to the model, not correlation with poverty rate. For example, all features have positive importance, increasing '% unemployment' increases poverty rate, and increasing '% non_hispanic_white' decreases poverty rate.

The features added for the XGBoost were less important than any original feature. The highest, 'population estimate', scored an importance of appx. 0.006.

The model performance can improve if additional time is dedicated to exploring alternate features, including additional data, derived features and meta-features. Stacked model ensembles would have been the next step in this project.

Top Ten Features by Importance



features	importance
pct_civilian_labor	0.163849
pct_unemployment	0.079277
pct_uninsured_adults	0.062562
pct_adults_less_than_a_high_school_diploma	0.040816
pct_below_18_years_of_age	0.038385
pct_uninsured_children	0.037688
pct_female	0.034164
pct_aged_65_years_and_older	0.032657
pct_non_hispanic_white	0.031583
pct_adult_smoking	0.030885

Additional considerations may be put towards modelling the poverty level as it is experienced instead of the poverty line. This could include distinguishing between costs of living, medical costs, tax burdens and other factors the "Supplementary Poverty Measure" explores that the "poverty line" ignores. Having access to that data may reveal a richer feature space that allows for more accurate and meaningful models.

Introduction

Poverty rate is the percentage of people living beneath the poverty level, a group of income thresholds set by the U.S. Office of Management and Budget for determining whether individuals and families can meet their basic needs.[1] People below the level may be eligible for government aid, however, the level does not account for economic factors such as geographical factors, cost of living, taxes, or medical costs. Discrepancies in its usefulness arise in situations where poverty rates are higher in rural areas but the cost of living lower, compared with higher incomes in urban areas with much higher costs of living.

A Supplemental Poverty Measure[2] has been used since 2011 to assess the effect of additional areas on poverty level. Included in these measurements are factors such as medical costs, social security income, and taxes, and the studies include how many people are pushed into or out of poverty based on the supplementary factors.

The purpose of this project is to predict the poverty rates across the United States at the county-level given other socioeconomic indicators. The data is available at the United States Department of Agriculture Economic Research Service (USDA ERS).[3]

As the capstone project for the Microsoft Professional Program in Data Science (DAT102x), students are expected to apply what they have learned to analyze, process, and model the data. Students may use whatever tools they choose, however, the projects are individual and materials such as code cannot be shared nor published.

The report is structured as a story that combines the data science skills being tested with their evolution as the author learns what is required to use them. Techniques such as cross validation become more refined as the report progresses. The experiments are presented in the order they were performed, mistakes included. The author hopes this allows others to learn from the process.

The general process applied by Machine Learning competition winners include:[4]

- Understanding the problem and dataset
- Data cleansing, Preprocessing, Dummy Variables
- Feature selection and creation
- Selecting the modeling algorithm
- Parameter tuning through cross validation
- Building the model
- Checking the results by making a submission

The project was approached with a few other self-imposed principles:

- Each step will be based on experimental, technical or expert information. No guessing or “fiddling” with models.
- Work flow will always move forward to the next hypotheses unless a clear reason for backtracking presents itself.
- Data manipulation will be as replicable as skills allow.
- Each model will be studied in greater detail as it is employed.
- All work will be done in Python 3.6 using Jupyter Notebooks[5]

[1] https://www.census.gov/topics/income-poverty/poverty/guidance/poverty-measures.html accessed Jan. 2018

[2] https://www.census.gov/library/publications/2017/demo/p60-261.html accessed Jan. 2018

[3] https://www.ers.usda.gov/topics/rural-economy-population/rural-poverty-well-being/poverty-overview.aspx#howis acc. Jan. 2018

[4] https://www.analyticsvidhya.com/blog/2016/10/winning-strategies-for-ml-competitions-from-past-winners/ accessed Jan. 2018

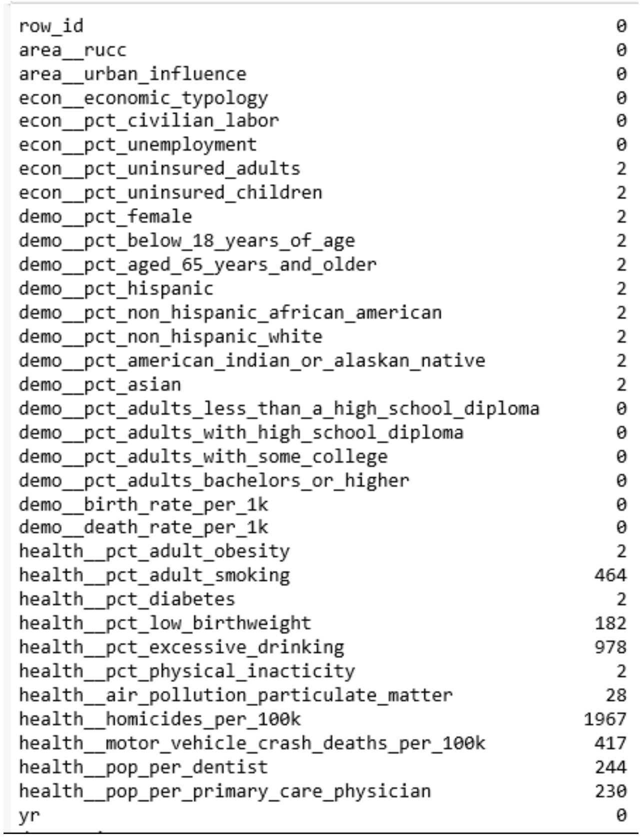
[5] See Appendix B for Python resources

Data Description

The training data had 3198 examples and 33 features, plus a row of column headers and a column of row IDs. The row IDs ranged from 0 to 6277. The column titles were all text and formatted with the first four letters representing a category (see Appendix A) and the rest describing the features themselves.

There were 4 text based features and 29 numerical features, which included 22 percentage and 7 non-percentage values.

Missing values were concentrated in the "health__" category. Other categories had features with missing values in two rows (Figure 1).



```
row_id                                        0
area__rucc                                    0
area__urban_influence                         0
econ__economic_typology                       0
econ__pct_civilian_labor                      0
econ__pct_unemployment                        0
econ__pct_uninsured_adults                    2
econ__pct_uninsured_children                  2
demo__pct_female                              2
demo__pct_below_18_years_of_age               2
demo__pct_aged_65_years_and_older             2
demo__pct_hispanic                            2
demo__pct_non_hispanic_african_american       2
demo__pct_non_hispanic_white                  2
demo__pct_american_indian_or_alaskan_native   2
demo__pct_asian                               2
demo__pct_adults_less_than_a_high_school_diploma  0
demo__pct_adults_with_high_school_diploma     0
demo__pct_adults_with_some_college            0
demo__pct_adults_bachelors_or_higher          0
demo__birth_rate_per_1k                       0
demo__death_rate_per_1k                       0
health__pct_adult_obesity                     2
health__pct_adult_smoking                   464
health__pct_diabetes                          2
health__pct_low_birthweight                 182
health__pct_excessive_drinking              978
health__pct_physical_inacticity               2
health__air_pollution_particulate_matter     28
health__homicides_per_100k                 1967
health__motor_vehicle_crash_deaths_per_100k 417
health__pop_per_dentist                     244
health__pop_per_primary_care_physician      230
yr                                            0
```

Figure 1. Feature Name and Missing Value Count

Mean, Median, StdDev, Min, and Max were calculated for each numeric feature. The results appeared reasonable in sign and magnitude, with no evidence of a systematic or clerical error affecting the data.

The covariance matrix was calculated for all numerical features and the poverty rate. The correlations were within +/- 0.6. Scatter plots of the features offered qualitative support of the correlations, but nothing was observed that warrants additional mention.

Fast Mung. One instructor recommended that a "quick and dirty" model be deployed as soon as possible so there was a clear baseline to work from.[6] The data requiring the least effort to employ consisted of percentage-value features from the econ__ and demo__ groups, as well as obesity, diabetes and inactivity from the health__ group (also percentage valued). The two null values appearing amongst these features all occurred in the same two rows, and were imputed using the mean of the feature. All other columns were omitted.

As the values were percentages, all of them were already between 0 and 1. The racial features appeared to be dependent, as did the education features. The ratios within each demographic set needed to stay constant, so feature scaling was not performed.

The heat map of the correlation of these 19 features and poverty rate (**Figure 2**) reveals that half have correlations with absolute values beneath 0.2 (grey areas), while the other half are as high as 0.6 (strong blue or red). The strongest positive correlations – features that increase with increasing poverty rate - include % unemployment, % uninsured adults, % Hispanic/African American, and % less than high school education. The strongest negative correlations – features that increase with decreasing poverty rate – are % civilian labor and % non-hispanic/white.

[6] Andrew Ng's Stanford ML Course. https://www.coursera.org/learn/machine-learning Fall 2017

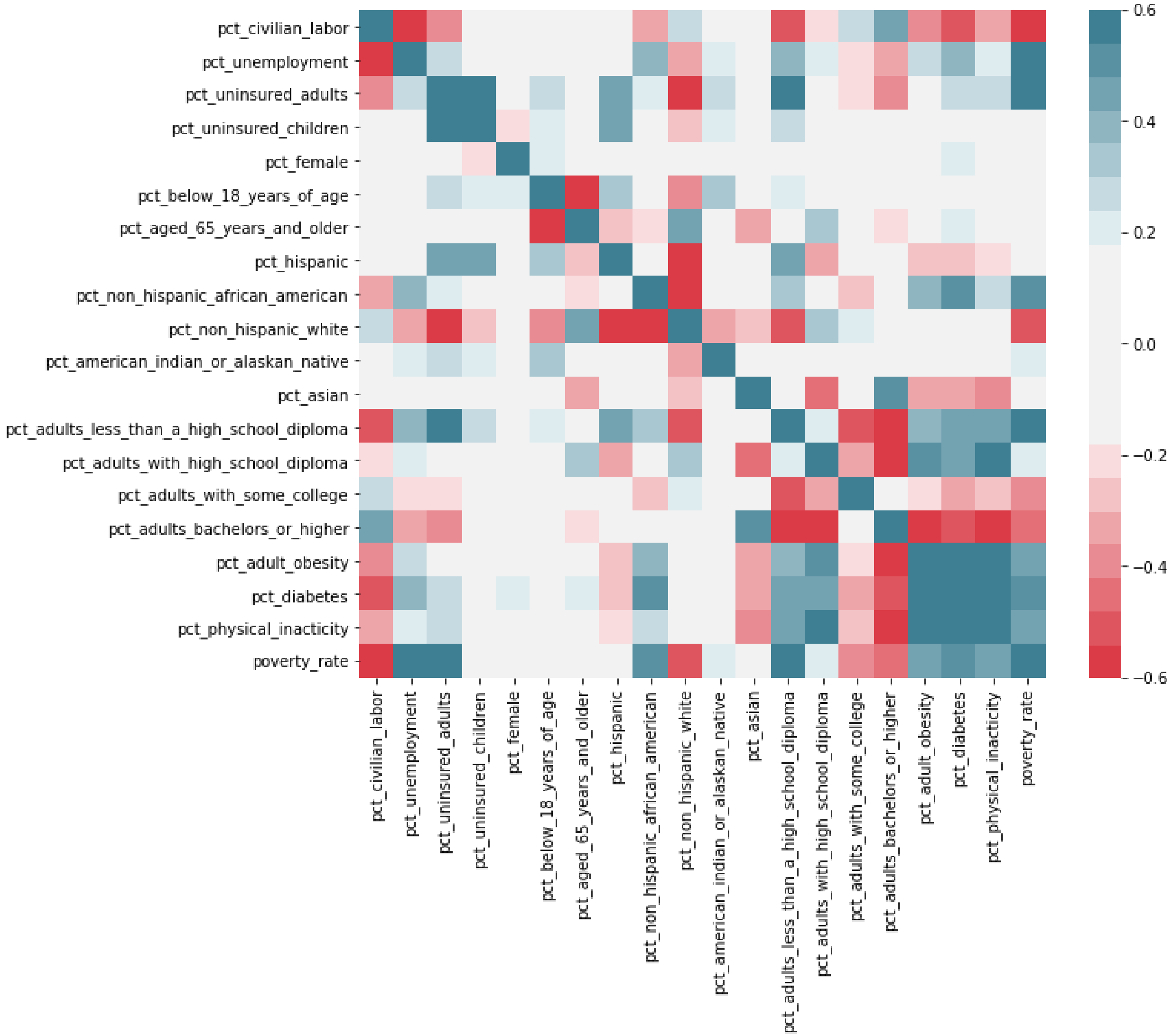


Figure 2. Heatmap of the "fast mung" features used in the baseline regression model

Multi Linear Regression (MLR). Linear regression was selected as the first model for its simplicity and speed of deployment. The model was trained using 70% of the fast mung training data. The remaining 30% was saved for cross validation. The split was chosen randomly and saved so that subsequent models could be compared.

Table 1 has the results for regression with 'None', 'L2' (ridge) and 'L1' (lasso) regularization. All three had similar outcomes, with a common test RMSE ~ 3.23. The local RMSE was denoted as R^2 in the table because the author did not realize at the time the default error in the program was R^2, not RMSE. Changes were made in subsequent models, but this model was not revisited to update the result.

Examination of the parameters calculated by the three regressions revealed slight differences and small regularization parameters. This was consistent with a high bias model warranting more complexity.

Table 1. Model Results

Model	Data Set	Local RMSE	Test RMSE
Multi-Linear Regression	Fast	0.735 (R^2)	3.23
MLR + L2 Reg	“	0.732 (R^2)	3.23
MLR + L1 Reg	“	0.734 (R^2)	n/s
SVR	“	2.26	2.85
SVR + Bagging	“	2.06	2.81
Adaboost	“	1.89	2.87
XGBoost	“	2.58	2.91
XGBoost (first)	Intermediate w/ nulls		2.88
XGBoost (best)	“	2.38	2.69 (Best)
XGBoost (post-best 1)	“	2.34	2.78
XGBoost (post-best 2)	“	2.35	2.73
XGBoost (prior best)	Intermed. + no nulls	2.39	2.78
XGBoost + L2 Reg	“	2.47	2.74
Extra Random Trees	“	2.37	2.82
Extra Random Trees (best)	“	2.33	2.77

Support Vector Regression (SVR). The complexity was increased by changing to SVR, which is expected to capture nonlinearities through the use of kernels.[6] A parameter search routine called GridSearchCV was used to fit the SVR parameters, and a 5-fold cross validation was used to score their fits. At the time, 100% of the data was used in the fitting and 5-fold CV. Gaussian kernel (a.k.a. radial bias function/rbf kernel) performed better than linear and polynomial kernels. RMSE improved to 2.85 (Table 1).

Next, multiple SVR predictors were bagged into one model. Each used the parameters of the single SVR but was trained on a different subset of the data. Again, random states were fixed. The parameters of the bagging included the number of rows (samples) and number of columns (features) to be used in creating each SVR, as well as how many SVRs (estimators) to create for the 'bag'. Plotting these parameters vs RMSE (Figure 3) shows that the model's test results level off at 10 estimators, 17 features, and 100% samples. These graphs were examined for several other random sets and the average result was 10 estimators, 15 features, and 100% of the samples.

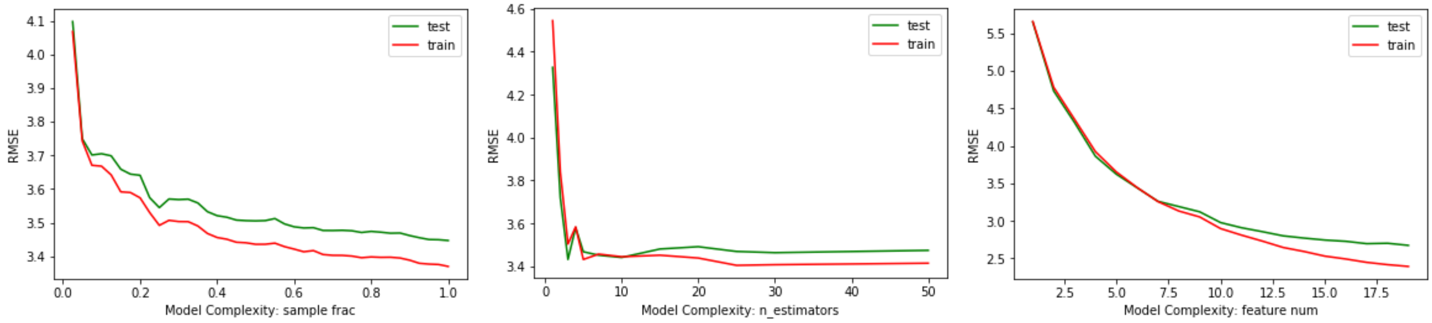


Figure 3. Graphs showing the impact of bagging parameters on SVR RMSE.

Combining the bagging parameters and the SVR parameters offered a slight improvement over the SVR alone. Further tuning of the combined set of parameters was not performed to leave time for Gradient Boosting Machines, which hold promise for data like these where the learners are weak and the system high in bias.

Gradient Boosted Machines

AdaBoost (for adaptive boosting) is the algorithm discussed in the courses. Initial local results improved using the same 5-fold cross validation as with SVR and using SVR as a kernel, however, the test results worsened slightly. This is likely due to overfitting, which is a risk using GBMs. Different kernels did not improve the results. One solution would be to add remaining features, but that was not done so another GBM could be compared.

XGBoost (Extreme gradient boosting)[7] is popular in online competitions,[8,9] and seems to be the algorithm many competitors start with their modeling. It implements an ensemble of gradient boosted trees with exceptional speed and has been effective on a wide variety of modelling problems. It is expected to perform better than Adaboost, SVR and MLR.

The first implementation of XGBoost used the fast mung data and 5-fold cross validation and achieved a test RMSE slightly worse than the Adaboost. This performance was below expectations, given the response to the algorithm in the competition community, so more research was done on its operation. The main takeaway was that XGBoost is at its best in data comprised by weak learners, sparse matrices, unknown feature importance, and missing values.[10]

Intermediate Mung

The best way to take advantage of XGB's power is to return to the original data set. This time, all of the features were allowed to remain. One book on the algorithm showed that it is possible for XGB to perform better with missing values in place, so that was made the case here. The main issue was what to do with the text columns. After doing some research on encoding categorical variables in Python, the following changes were made:

Area__rucc: This feature had 9 possible entries. It was encoded as 8 features. The first was binary for whether the entry was for a "Metro" county. There were six population scales, 3 for Nonmetro and 3 for Metro. Each of these was encoded as binary dummy values in a sparse 6-feature matrix.

The eighth feature was of population estimates to capture trends that might be observed with numeric population values. As the population data was not provided, each category was assigned a value from its listed range. For example, counties that were 'Metro areas with 250,000 people or less' were assigned 250,000.

Area__urban_influence: This feature had 12 unique entries and was encoded as 13 features. The original entries overlapped with Area__rucc. The key difference was extracted as one binary feature for whether a county was "adjacent" to a metro area. The starting feature was then encoded as dummy binary values in a sparse 12-feature matrix.

Economic Topology: the six unique entries of this feature were dissimilar and encoded as a 6-feature sparse dummy matrix.

Year: the two possible values of year were encoded as 0 and 1. It is uncertain whether there is a better way to do this one.

Greater detail on these manipulations can be found in Appendix A. All of the text was removed from the data after these columns were added. The final data set, which will be referred to as the **intermediate** data set, had 57 features that included 3 sets of sparse dummy features, a numerical population estimate feature, and intact missing values.

No Preprocessing. No further preprocessing was performed on the features at this point. All of the added features except population estimate were between 0 and 1, and XGB is not sensitive to the scaling of features like population estimate. Both the null values and the scaling needs to be reconsidered when changing to different models.

[7] https://github.com/dmlc/xgboost, Accessed Jan. 2018
[8] https://www.kaggle.com/dansbecker/learning-to-use-xgboost, Accessed Jan. 2018
[9] http://blog.kaggle.com/tag/xgboost/, Accessed Jan. 2018
[10] Brownlee, Jason. XGBoost With Python. Self Published. 2017.

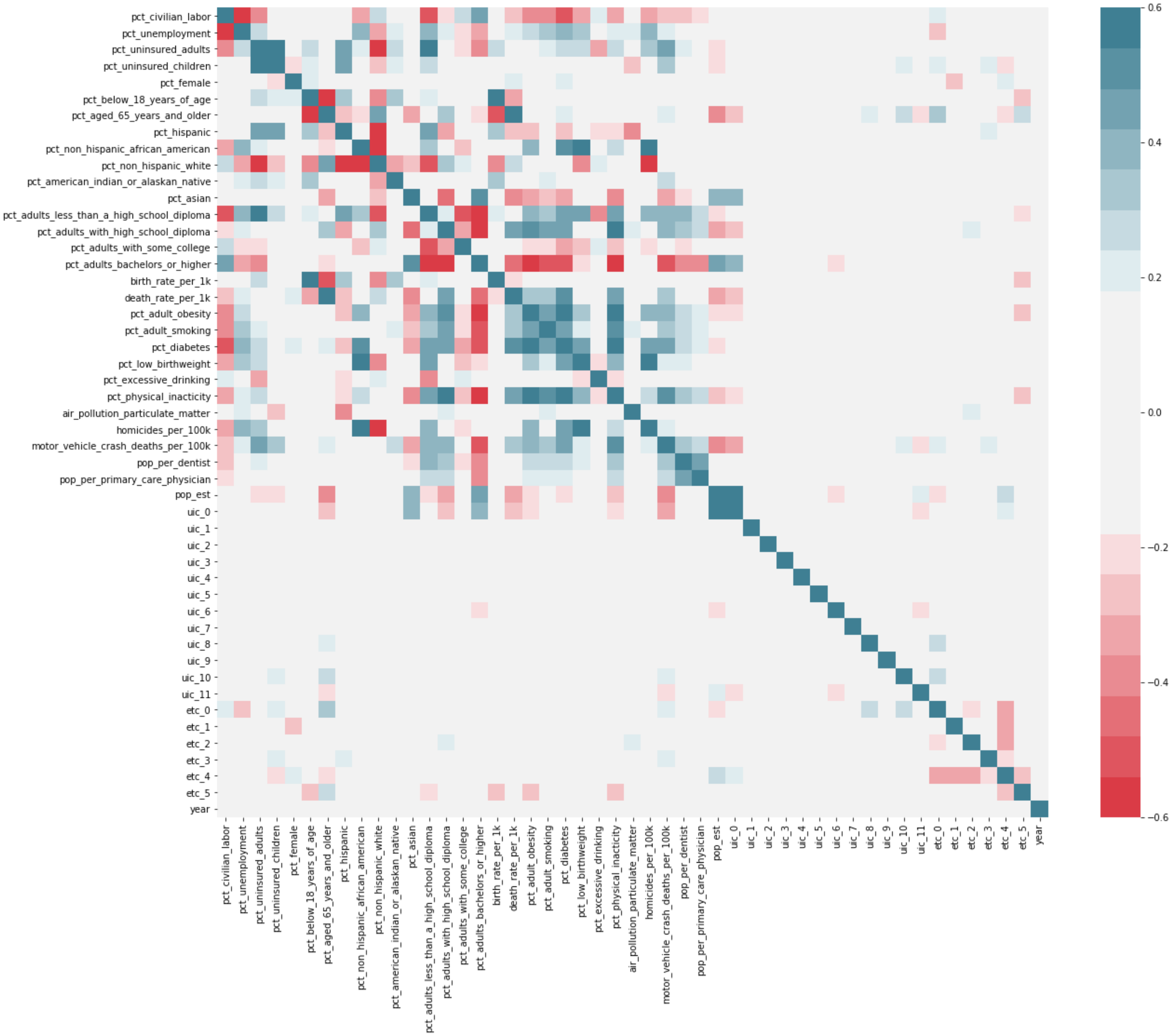
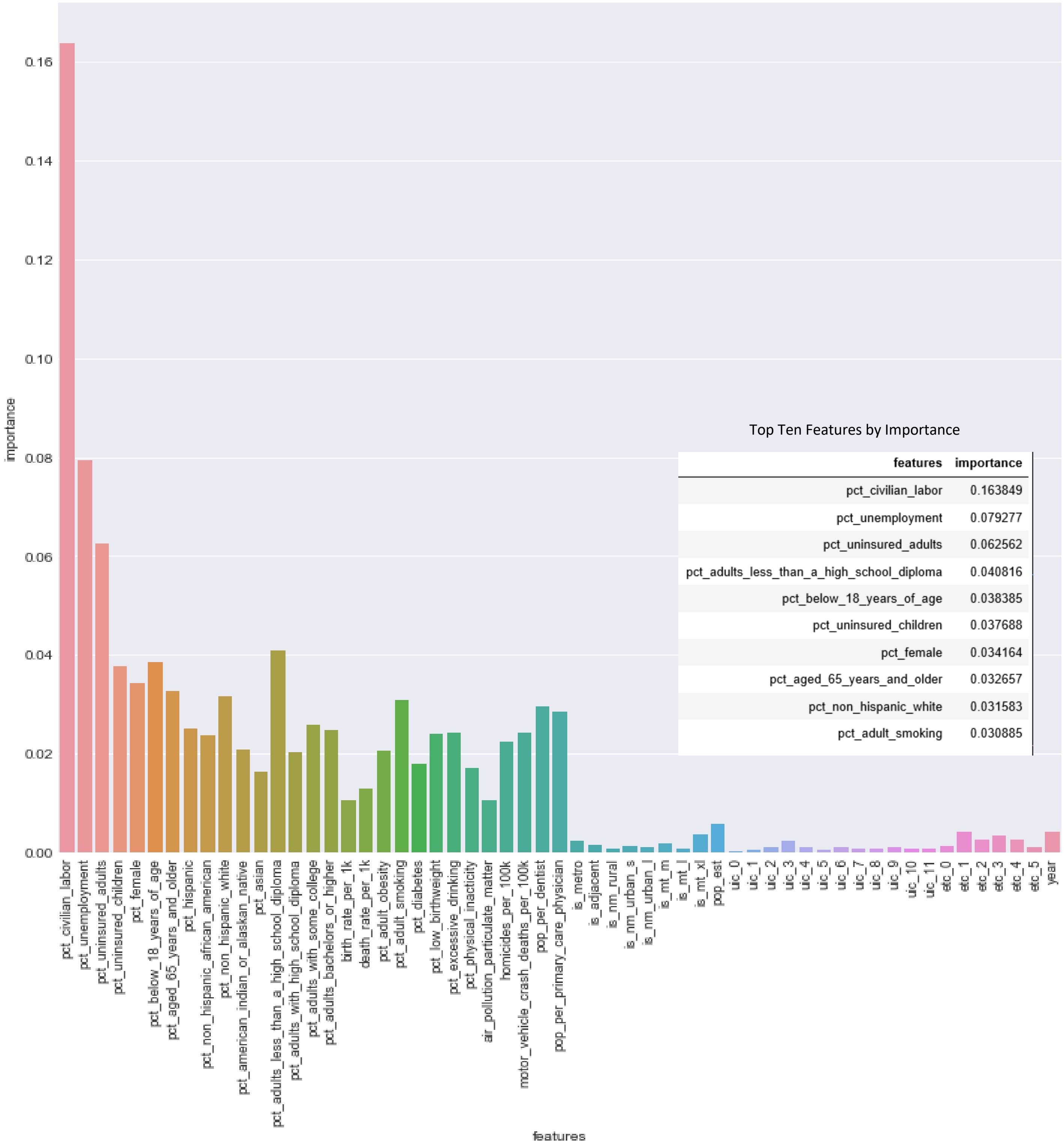


Figure 4. Heatmap of all features used in the intermediate data set. The original features are in the upper-left quadrant. The new features exhibit much weaker correlations than the original ones, which is also reflected in the feature importance. See Appendix A for feature details.

Correlation and Feature Importance. Error! Reference source not found. is a heat map of the correlations between all of the features. All new features exhibit weaker correlations than the original features.

The feature importance can be explored with the XGBoost models, as illustrated in Figure 5, below. The most important features are % civilian labor, % unemployment and % uninsured adults. The new features have low importance in comparison with the originals, but the algorithm recognizes population estimate, year, economic typologies (etc_) and the largest metro size (is_m_xl) as the strongest contributors among them. This intrinsic ability to feature-select means additional feature selection methods such as PCA were not needed.



Top Ten Features by Importance

features	importance
pct_civilian_labor	0.163849
pct_unemployment	0.079277
pct_uninsured_adults	0.062562
pct_adults_less_than_a_high_school_diploma	0.040816
pct_below_18_years_of_age	0.038385
pct_uninsured_children	0.037688
pct_female	0.034164
pct_aged_65_years_and_older	0.032657
pct_non_hispanic_white	0.031583
pct_adult_smoking	0.030885

Figure 5. Feature importance with features ordered as they are ordered in the data. A list of the top ten features is inset. Feature names are described in Appendix A.

XGBoost and the Intermediate Data Set

The training data was split into two fractions, 80% for training/cv and 20% holdout for model testing. A 10-fold CV scored using the mean RMSE was selected after plotting the mean RMSE for k-fold CV with k = 2 through 20 and noting that the RMSE levels off at k = 10. The random seeds were fixed for the training/holdout and the fold selection so all models could be compared.

Parameter tuning was performed by first researching starting points.[10,11] The behaviors of the key parameters were explored as illustrated in **Figure 6** for tree count, tree depth, and algorithm learning rate. Once intuition was developed regarding the behavior of the parameters, parameter grid searches were used to tune the parameters a few at a time. The fit of the model was judged by the 10-fold CV's mean RMSE and the RMSE of the holdout set.

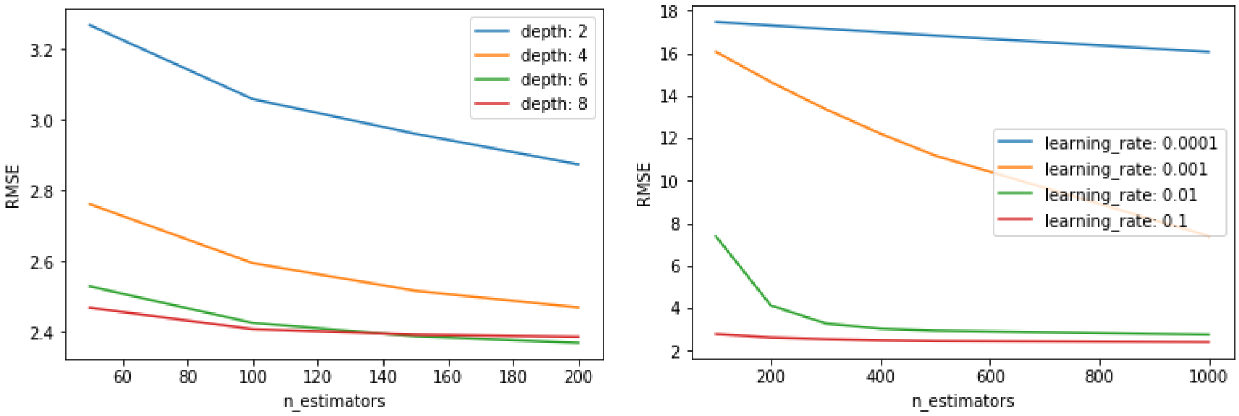


Figure 6. RMSE vs tree count (n_estimator) for XGBoost for various tree depths (Left) and learning rates (Right).

This process started with a test RMSE of 2.88 and led to parameters that achieved test RMSE of 2.69. Further search with small parameter grid steps revealed that the model and holdout RMSEs could be improved slightly, but corresponding test RMSEs increased (Table 1). This is attributed to the local gains being artefacts of the error in mean CV RMSE and the systematic error from the fixed random state.

Random Forests

According to online competition winners, the biggest gains in model performance come from model ensembling and feature creation/data preparation.[4]

Ensembling was the first to be considered, but due to the length of the report minimal details can be provided.

Random Forest Regressors and Extra Random Tree Regressors (Random Forests w/ replacement and more randomness in their branches) were compared to XGB. The null values of the data set had to be removed: the entries that were imputed as feature means for the fast mung were treated in the same fashion. The remaining entries, each with large numbers of missing values, were imputed as multi-linear fits of the feature with its two most correlated features. This imputation scheme attempted to avoid the impact of imputing hundreds of absent values with a single mean or 0 value.

There was no opportunity to test this assumption, but the best XGB model was refit with this modified intermediate data. This refit had a local RMSE was 2.35, which was better than the original (Table 1). The test RMSE of 2.78 was worse. To see if this was a result of overfitting, L2 regularization was applied, resulting in a higher local RMSE of 2.47 and improved test RMSE of 2.74.

Extra Random Trees (ERT) performed better than Random Forests. ERT had a test RMSE of 2.77 (Table 1).

[11] https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

Recommendations

Had there been time, model stacking would have been the next method explored. Model stacking involves using the results of the first set of models as features in the data for the second. Stacking is popular in competitions and a good way to smooth out overfitting and improve performance.

Additional data manipulations would have come in the form of feature exploration, specifically, finding new derived features to help explain the behaviors observed.

In retrospect, outliers and residuals were overlooked, so those could be examined for model improvement.

It is beyond the scope of the competition rules (no outside data), but it would be interesting to explore how factors of the Supplementary Poverty Measure improve the quality and meaningfulness of these results.

Conclusions

Beginning with a fast-deployed, simple multi-linear regression and proceeding methodically through increasing model and feature complexity resulted in a best model that uses XGBoost regression to achieve a test RMSE of 2.69. This can almost be improved through additional data treatment and model ensembling techniques, but there was not enough time to perform these operations.

The XGBoost algorithm highlighted the relative importance of each of the features considered. The most important were % civilian labor, % unemployment, and % uninsured adults. Low education, racial demographics (Hispanic, African American) and certain health factors were also strong contributors. These areas are should be examined for causes that can be affected to improve poverty rates.

Appendix A

Feature Information, as provided at:

https://www.datasciencecapstone.org/competitions/3/county-poverty/page/10/

There are 33 variables in this dataset. Each row in the dataset represents a United States county, and the dataset we are working with covers two particular years, denoted a, and b We don't provide a unique identifier for an individual county, just a row_id for each row.

The variables in the dataset have names that of the form category__variable, where category is the high level category of the variable (e.g. econ or health). variable is what the specific contains.

Categories

AREA — INFORMATION ABOUT THE COUNTY

- area__rucc — Rural-Urban Continuum Codes "form a classification scheme that distinguishes metropolitan counties by the population size of their metro area, and nonmetropolitan counties by degree of urbanization and adjacency to a metro area. The official Office of Management and Budget (OMB) metro and nonmetro categories have been subdivided into three metro and six nonmetro categories. Each county in the U.S. is assigned one of the 9 codes." (USDA Economic Research Service, https://www.ers.usda.gov/data-products/rural-urban-continuum-codes/)
- area__urban_influence — Urban Influence Codes "form a classification scheme that distinguishes metropolitan counties by population size of their metro area, and nonmetropolitan counties by size of the largest city or town and proximity to metro and micropolitan areas." (USDA Economic Research Service, https://www.ers.usda.gov/data-products/urban-influence-codes/)

ECON — ECONOMIC INDICATORS

- econ__economic_typology — County Typology Codes "classify all U.S. counties according to six mutually exclusive categories of economic dependence and six overlapping categories of policy-relevant themes. The economic dependence types include farming, mining, manufacturing, Federal/State government, recreation, and nonspecialized counties. The policy-relevant types include low education, low employment, persistent poverty, persistent child poverty, population loss, and retirement destination." (USDA Economic Research Service, https://www.ers.usda.gov/data-products/county-typology-codes.aspx)
- econ__pct_civilian_labor — Civilian labor force, annual average, as percent of population (Bureau of Labor Statistics, http://www.bls.gov/lau/)
- econ__pct_unemployment — Unemployment, annual average, as percent of population (Bureau of Labor Statistics, http://www.bls.gov/lau/)
- econ__pct_uninsured_adults — Percent of adults without health insurance (Bureau of Labor Statistics, http://www.bls.gov/lau/)
- econ__pct_uninsured_children — Percent of children without health insurance (Bureau of Labor Statistics, http://www.bls.gov/lau/)

HEALTH — HEALTH INDICATORS

- health__pct_adult_obesity — Percent of adults who meet clinical definition of obese (National Center for Chronic Disease Prevention and Health Promotion)
- health__pct_adult_smoking — Percent of adults who smoke (Behavioral Risk Factor Surveillance System)

- health__pct_diabetes — Percent of population with diabetes (National Center for Chronic Disease Prevention and Health Promotion, Division of Diabetes Translation)
- health__pct_low_birthweight — Percent of babies born with low birth weight (National Center for Health Statistics)
- health__pct_excessive_drinking — Percent of adult population that engages in excessive consumption of alcohol (Behavioral Risk Factor Surveillance System,)
- health__pct_physical_inacticity — Percent of adult population that is physically inactive (National Center for Chronic Disease Prevention and Health Promotion)
- health__air_pollution_particulate_matter — Fine particulate matter in µg/m³ (CDC WONDER, https://wonder.cdc.gov/wonder/help/pm.html)
- health__homicides_per_100k — Deaths by homicide per 100,000 population (National Center for Health Statistics)
- health__motor_vehicle_crash_deaths_per_100k — Deaths by motor vehicle crash per 100,000 population (National Center for Health Statistics)
- health__pop_per_dentist — Population per dentist (HRSA Area Resource File)
- health__pop_per_primary_care_physician — Population per Primary Care Physician (HRSA Area Resource File)

DEMO — DEMOGRAPHICS INFORMATION

- demo__pct_female — Percent of population that is female (US Census Population Estimates)
- demo__pct_below_18_years_of_age — Percent of population that is below 18 years of age (US Census Population Estimates)
- demo__pct_aged_65_years_and_older — Percent of population that is aged 65 years or older (US Census Population Estimates)
- demo__pct_hispanic — Percent of population that identifies as Hispanic (US Census Population Estimates)
- demo__pct_non_hispanic_african_american — Percent of population that identifies as African American (US Census Population Estimates)
- demo__pct_non_hispanic_white — Percent of population that identifies as Hispanic and White (US Census Population Estimates)
- demo__pct_american_indian_or_alaskan_native — Percent of population that identifies as Native American (US lyCensus Population Estimates)
- demo__pct_asian — Percent of population that identifies as Asian (US Census Population Estimates)
- demo__pct_adults_less_than_a_high_school_diploma — Percent of adult population that does not have a high school diploma (US Census, American Community Survey)
- demo__pct_adults_with_high_school_diploma — Percent of adult population which has a high school diploma as highest level of education achieved (US Census, American Community Survey)
- demo__pct_adults_with_some_college — Percent of adult population which has some college as highest level of education achieved (US Census, American Community Survey)
- demo__pct_adults_bachelors_or_higher — Percent of adult population which has a bachelor's degree or higher as highest level of education achieved (US Census, American Community Survey)
- demo__birth_rate_per_1k — Births per 1,000 of population (US Census Population Estimates)
- demo__death_rate_per_1k — Deaths per 1,000 of population (US Census Population Estimates)

Example Row

Here's an example of one of the rows in the dataset so that you can see the kinds of values you might expect in the dataset. Some are numeric, some are categorical, and there can be missing values.

area__rucc	Nonmetro - Completely rural or less than 2,500 urban population, adjacent to a metro area
area__urban_influence	Noncore adjacent to a large metro area
econ__economic_typology	Federal/State government-dependent
econ__pct_civilian_labor	0.358
econ__pct_unemployment	0.089
econ__pct_uninsured_adults	0.253

econ__pct_uninsured_children	0.099
demo__pct_female	0.494
demo__pct_below_18_years_of_age	0.2
demo__pct_aged_65_years_and_older	0.195
demo__pct_hispanic	0.044
demo__pct_non_hispanic_african_american	0.517
demo__pct_non_hispanic_white	0.378
demo__pct_american_indian_or_alaskan_native	0.056
demo__pct_asian	0
demo__pct_adults_less_than_a_high_school_diploma	0.223896
demo__pct_adults_with_high_school_diploma	0.345382
demo__pct_adults_with_some_college	0.273092
demo__pct_adults_bachelors_or_higher	0.157631
demo__birth_rate_per_1k	10
demo__death_rate_per_1k	11
health__pct_adult_obesity	0.345
health__pct_adult_smoking	0.219
health__pct_diabetes	0.159
health__pct_low_birthweight	0.154
health__pct_excessive_drinking	NaN
health__pct_physical_inacticity	0.317
health__air_pollution_particulate_matter	12
health__homicides_per_100k	9.33
health__motor_vehicle_crash_deaths_per_100k	33.75
health__pop_per_dentist	5429
health__pop_per_primary_care_physician	6949
yr	b

References

- Economic Research Service (ERS), U.S. Department of Agriculture (USDA). Poverty Series. https://www.ers.usda.gov/topics/rural-economy-population/rural-poverty-well-being/poverty-overview.aspx.

- University of Wisconsin Population Health Institute. County Health Rankings & Roadmaps. www.countyhealthrankings.org.

New features are described on the next page.

Appendix A (continued)

New Feature Information

The following features are 0 (False) or 1 (True) depending on whether the rural urban continuum code (area__rucc) feature contained:

'is_metro' = 1 if feat. contains 'Metro'

'is_adjacent' = 1 if feat. contains ', adjacent' (this is for urban_influence, not rucc)

'is_nm_rural' = 1 if feat. contains 'Completely rural or less than 2,500 urban population'

'is_nm_urban_s' = 1 if feat. contain 'Urban population of 2,500 to 19,999'

'is_nm_urban_l' = 1 if feat. contains 'Urban population of 20,000 or more'

'is_mt_m' = 1 if feat. contains 'Counties in metro areas of fewer than 250,000 population'

'is_mt_l' = 1 if feat. contains 'Counties in metro areas of 250,000 to 1 million population'

'is_mt_xl' = 1 if feat. contains 'Counties in metro areas of 1 million population or more'

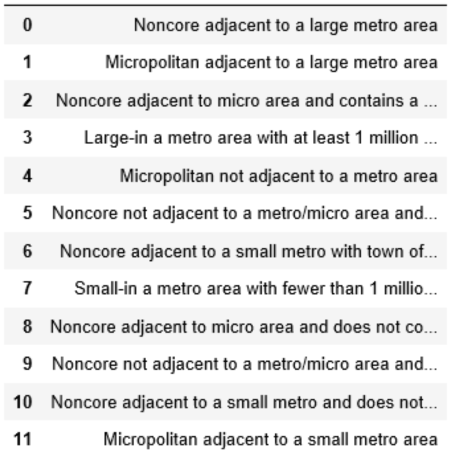
Numeric Feature:

'pop_est' = one of the following values, based on the above:

- ruralpop = 1500
- nm_urban_s = 10000
- nm_urban_l = 25000
- nm_urban_s = 10000
- nm_urban_l = 25000
- mt_m = 250000
- mt_l = 500000
- mt_xl = 1000000

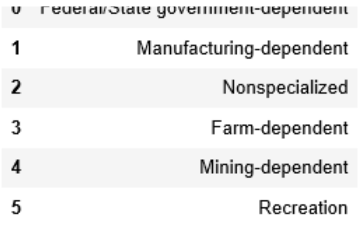
The urban influence feature (area__urban_influence) is encoded as 12 binary features, uic_0 through uic_11 with each feature corresponding to one of the values in Figure 7

The economy typology feature was encoded as 6 binary features etc_0 through etc_5, with each feature corresponding to the value in Figure 8



0	Noncore adjacent to a large metro area
1	Micropolitan adjacent to a large metro area
2	Noncore adjacent to micro area and contains a ...
3	Large-in a metro area with at least 1 million ...
4	Micropolitan not adjacent to a metro area
5	Noncore not adjacent to a metro/micro area and...
6	Noncore adjacent to a small metro with town of...
7	Small-in a metro area with fewer than 1 millio...
8	Noncore adjacent to micro area and does not co...
9	Noncore not adjacent to a metro/micro area and...
10	Noncore adjacent to a small metro and does not...
11	Micropolitan adjacent to a small metro area

Figure 7. Urban_Influence feature entries corresponding to features uic_0 through uic_11



0	Federal/State government-dependent
1	Manufacturing-dependent
2	Nonspecialized
3	Farm-dependent
4	Mining-dependent
5	Recreation

Figure 8. Economic Typology feature entries corresponding to features etc_0 through etc_5

Appendix B – Python Resources

Python 3.6

Jupyter Notebook 5.3.1

Anaconda 1.6.11

Orange 3.8

Courses

EdX: Introduction to Python for Data Science

EdX: Applied Machine Learning

EdX: Programming with Python for Data Science

Udemy: Python for Data Science and Machine Learning Bootcamp

Coursera: Machine Learning

Primary Python packages

NumPy	www.numpy.org	Array package
Pandas	pandas.pydata.org	Data structures & analysis
Matplotlib	matplotlib.org	2D Plotting
Seaborn	seaborn.pydata.org	Data Visualization
SciPy	scipy.org	Scientific and technical computing
Scikit-Learn	scikit-learn.org	Machine Learning
Statsmodel	www.statsmodels.org	Statistics
Xgboost	github.com/dmlc/xgboost	Xgboost model

Key books

Brownlee, Jason. Machine Learning Mastery with Python. V1.9 Self-published. 2017.

Raschka, Sebastian. Python Machine Learning. Pack Publishing, 2016.

Key websites/blogs

All of the python package documentation, especially Pandas and scikit-learn!

www.analyticsvidhya.com

www.stackoverflow.com

www.github.com

www.machinelearningmastery.com

www.kaggle.com

www.kdnuggets.com