Examining How U.S. Citizen's Socio-Economic Status Responded to the 2007 Financial Crisis by County

Ryan Goodwin, rgoodwin@smu.edu; Dustin Bracy, dbracy@smu.edu; Bala Dakshinamoorthi, bdakshinamoorthi@smu.edu

***Abstract*—We explore how specific counties in the United States responded to the 2007 financial crisis. We aim to highlight trends suggesting a general increase or decline in socio-economic status in the timeframe immediately following the crisis to present day. The results of this analysis can be used to pinpoint regions where overall status has since not improved, allowing for a deeper look into why these regions are not improving compared to regions that are showing improvement in socio-economic status since the crisis.**

***Index Terms*—Artificial Intelligence, AWS, Data Science, Machine Learning, S3, SageMaker**

I. Introduction

THE financial crisis of 2007-2008 caused waves of negative social and economic impact across the United States. Now, over ten years after the occurrence, this paper aims to look at the long-term recovery timeline of various counties across the country and any affected socio-economic factors. We explore social and economic recovery indicators by examining several explanatory variables including:

- Average Income Change
- Financial Assistance
- Unemployment Rates
- Business Ownership
- Housing information

Analysis of these data in a time series provides some insight into some of the driving factors responsible for these regions' socio-economic status recovery or lack thereof in in the years following the crisis.

II. Problem Solving Approach

To answer these questions, we first sought out to find applicable metrics surrounding the US relating that would provide these insights. We then needed to transform this data into a format that could be analyzed by statistical software. In order to help us collaborate on a shared dataset, we decided to host all data and analysis in the cloud.

The cloud provides for networking and compute capacity required in order to analyze millions of records of household data across various counties in the US. It also facilitates sharing of information and provides readily accessible tools for data analysis.

We perform exploratory data analysis (EDA) techniques to visualize the data and validate any assumptions required by the statistical models employed. The EDA helps to find correlation in the explanatory and response variables which might explain factors attributing to the counties' recovery speed.

After EDA we can use various statistical models to test hypothesis and determine the accuracy and effectiveness at the various factors which appear to explain recovery response.

III. Cloud Implementation

We decided to utilize Amazon Web Services (AWS) to house and perform our research. The raw data was downloaded from the US Census website and stored in an Amazon Simple Storage Service (S3) bucket, which is housed in the cloud on Amazon web servers. This storage provides a mean to inexpensively house the large data files required for analysis. The total size of the Current Population Surveys (CPS) from 2007 through 2018 is just under one gigabyte in size.

Data processing was done in R and Python, in order to combine the annual reports into a single source on which we could perform analysis. EDA was performed in both R and Python, as was some preliminary modeling. Post-processing analysis and testing on the data again utilized AWS, with heavy lifting of feature selection and model building taking place on Amazon's SageMaker service.

SageMaker is a new web service which aims to automate machine learning and model selection tasks in a transparent way. Several competing Artificial Intelligence (AI) and Machine Learning (ML) automation platforms utilize proprietary algorithms that are hidden from the user.

Submitted for review on 1 Apr 2020 as part of MSDS Course 7346: Cloud Computing. This work is self-supported with assistance from resources made available by Southern Methodist University and educational credits available for use in Amazon Web Services (AWS).

Dustin Bracy is a graduate student studying under the Master of Science in Data Science program with Southern Methodist University, Dallas, TX 75205 USA (e-mail: dbracy@smu.edu).

Bala Dakshinamoorthi is a graduate student studying under the Master of Science in Data Science program with Southern Methodist University, Dallas, TX 75205 USA (e-mail: bdakshinamoorthi@smu.edu).

Ryan Goodwin is a graduate student studying under the Master of Science in Data Science program with Southern Methodist University, Dallas, TX 75205 USA (e-mail: rgoodwin@smu.edu).

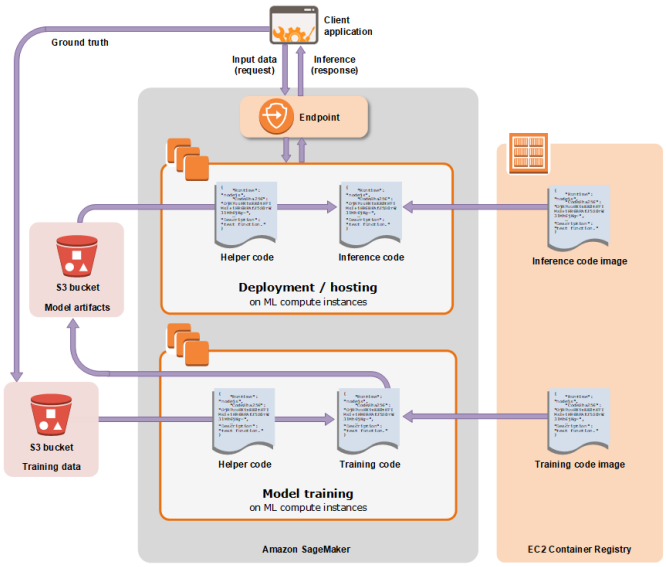

Fig. 1. Amazon Web Services (AWS) SageMaker model deployment. Source: https://docs.aws.amazon.com/sagemaker/latest/dg/how-it-works-training.html

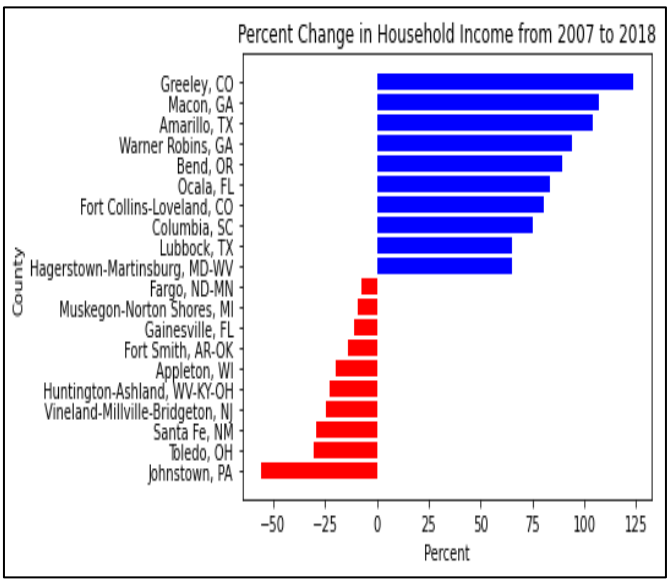

Fig. 2. EDA results for counties that recovered vs. those that did not recover.

SageMaker is different in that the results of the model building are made visible to users via notebooks, which allows further tuning and performance tweaks to be done by the user.

IV. Exploratory Data Analysis

Our initial EDA seeks to answer the questions of interest in the following subparagraphs. Before we can answer these questions, we first must define metrics which will define a successful recovery, and what it means to fail to recover. These measurements will become our response variables, those critical features which we will use to measure recovery of a county. Next, we must identify the explanatory variables, or those factors which are driving the response (recovery measure). These will provide the foundation for all further analysis, modeling, and prediction or hypothesis outcomes.

A. How do we measure recovery from the crisis?

The first objective of our EDA was to familiarize ourselves with the data source and select features that could best measure recovery from the crisis. We found total household income to be the most encapsulating variable to determine recovery. A rise in a counties average household income reasonably measures increased wellbeing among the county.

B. Which counties successfully recovered from the crisis?

Based on our selection of total household income as the response variable, we did a quick analysis on counties with the biggest percentage change in household income in 2007 and 2018. Our dataset provided us with CBSA codes for each county, which we then joined with a dataset mapping CBSA codes to County/County Equivalent names for clarity. Next, we dropped any counties that did not contain data for both 2007 and 2018. This left us with 200 counties to analyze. Although this is small subset of the total number of counties in the United States, it provided us with a large enough sample to continue analyzing. Once we had all valid counties, the percent difference in total household income was calculated.

Figure 2 above shows the top 10 counties in terms of recovery. During our analysis we also uncovered a few other notes in the data that we believed could be helpful when building our final model. 178 of the 200 valid counties saw an increase in their percent income, 15 counties saw an increase in income greater than 60 percent, and four counties saw greater than a 100 percent increase in income. Our initial hypothesis was that we could dig deeper into the top 15 counties in terms of income increase to discover insights, but ultimately the timeline did not allow for this activity.

C. Which counties did not appear to recover from the crisis?

Figure 2 also shows the bottom 10 counties in terms of percent change in income. Of the 200 valid counties we analyzed in our EDA, 22 saw a decrease in income percent from 2007 to 2018. Notably, 7 of the 22 counties with a decrease in income were in the Midwest. Although this is small sample size, given more time this was a hypothesis we would have liked to dig deeper into.

D. What are the primary factors leading to these outcomes?

After our skim of the data that led to findings on percent income change, our response variable, the next objective was to discover insights in the explanatory variables that could help us when building our model. A correlation heatmap was generated to help cut down the number of variables used in the model. The idea is to increase clarity into the model by reducing the number of features in the data set. Another benefit to feature reduction is the lightened load on training the model. Figure 3 below shows correlation the heat map generated from our data set.

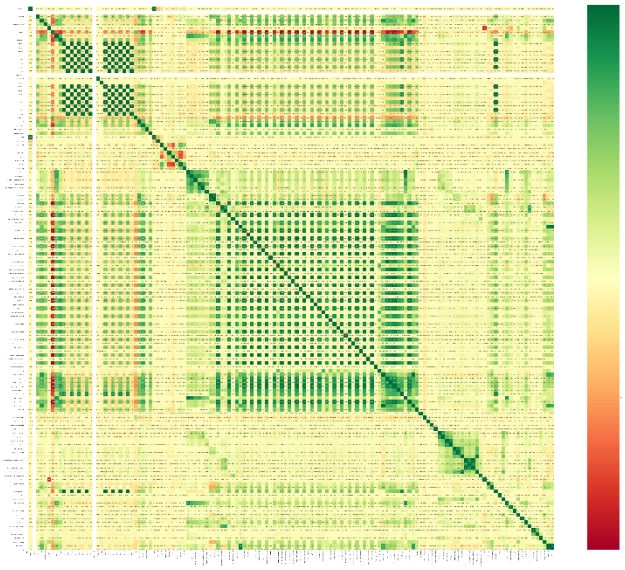

Fig. 3. Correlation Heat Map for Census Dataset

As can be seen in Figure 3, about one third of our variables are highly correlated with another variable. Picking these correlated variables out would have likely led to improved accuracy in our model. However, we decided to leave them in, partly due to our lack of expertise using SageMaker, as the belief was SageMaker would provide deeper analysis into correlation and drop these values for us.

Ultimately, although due diligence was performed in the EDA phase, our insights were left out of the SageMaker model. In hindsight, configuring our dataset after the exploratory phase would have likely led to increased performance in our model, working with SageMaker rather than relying on it to do the heavy lifting on its own.

V. Amazon SageMaker Model Process

Amazon SageMaker offers an incredible service to users, which makes good on its claim to make artificial intelligence and machine learning algorithms available to developers with no previous knowledge of these techniques. It is very well integrated with S3, which makes reading and writing data very efficient and relatively painless. Once a user is configured in AWS in Identity and Access Management (IAM), this user can be configured to cross the various features available in AWS. We leveraged this capability to create a 'cloud' user group, with a service account 'appuser' user to be used programmatically by our code to interact with SageMaker, S3, and the model endpoint after our model was put into production.

A. Accessing SageMaker

The first step in accessing SageMaker was to create an account in AWS at https://aws.amazon.com. After authenticating the account, one may log into the AWS Management Console in order to access the resources we would use to train and deploy our model. Amazon makes this easy to do, and offers the SageMaker service, which one can sign up for and enter via Amazon's SageMaker Studio. Upon creating a SageMaker user account, one can enter the Studio, where models can be built and trained. At the time of writing, the studio offers a one stop shop as an IDE, where one has access to a file explorer, a terminal/session explorer, options to incorporate source control via git repositories, a settings menu, an experiments section, a notepad settings menu and a tab to manage model endpoints.

B. Configuring SageMaker

This step is documented out of order, as our research took us first down the path of model development and deployment, before realizing we must configure access for our code to interact with the model after building and deploying in order to view our prediction results. It makes much more sense to configure first, which can streamline the deployment effort of the model. As mentioned above, best practice is to create a security group in IAM, and then add users to the group. We created a 'cloud' group and assigned access to SageMaker, and S3, in order to read input data, build, execute and document the model results, then write these as output back into an S3 bucket for analysis. Once users are created and assigned permissions (preferably via group), we can access the AWS Access Key ID and the AWS Secret Access Key for the user to which we will add to our app configuration file, along with the region and default output format we intend to use in the application.

C. Building the model

We chose to utilize an 'Autopilot Experiment' within SageMaker to see what results we could obtain with minimal finessing of the data or knowledge of machine learning. The Autopilot Experiment only requires: an experiment name, an S3 bucket location for the input file, the name of the response attribute (for which it should predict/classify), and an S3 bucket location for the output file(s). You may optionally select the machine learning type (auto, binary classification, linear regression, or multiclass classification). After entering these input parameters, SageMaker begins its process where it will iterate though methods for Analyzing Data, Feature Engineering, and Model Tuning.

During analysis, SageMaker creates a Data Exploration notebook, in which it describes the columns, what type of prediction problem it thinks we have, provides some summary statistics, descriptive statistics, as well as recommendations for preprocessing which can help improve prediction results. Another notebook is generated with Candidate Definitions which describes the generated models and provides code for the transformation logic (used in feature engineering) and hyperparameter tuning. This code may be executed locally to see how the model works or to fine tune the results of the algorithm. By default, SageMaker will run through 250 iterations for hyperparameter tuning to select the best parameters for the model for which it can make the best prediction for the provided response attribute.

D. Selecting and deploying the model

After SageMaker completes analysis and tuning, the user is greeted with a plethora of model analysis data, with each training, transformation, and processing job displayed with time and performance metrics. The training jobs offer details on prediction performance in an 'objective' column which

provides the best practice performance metric for the given prediction method (e.g. Mean Square Error, Validation Accuracy etc). The job with the best objective score has a star to flag it as the best iteration, which can then be selected for deployment.

Deploying the model to an endpoint takes only a few minutes for AWS to process before the endpoint is made available on an AWS instance of your chosen shape for use by your chosen AWS SDK. Users are given a choice to save prediction requests and responses for later performance analysis. Invoking the model is done by only a few lines of code, sending data in similar form as the training data via HTTP request and capturing the response.

VI. Interpretation

SageMaker provided a highly tuned, high performance model on the input data, which was the entirety of the CPS census data from 2007 through 2018. This input data contains several income metrics which can be summed to get to the total household income metric, which we used as our predicted response variable. The actual mathematics of the model weren't easily accessible through the SageMaker interface, but it is apparent by using the prediction results that the model is biased in using income metrics. Figure 4 documents the prediction results of a family of four, with employment, no business ownership, no farm income, no assistance of any kind, with a mortgage and a property value of $200,000. Because we provided no income or pseudo income data, the linear regression model was crippled with using only the categorical survey responses and location.

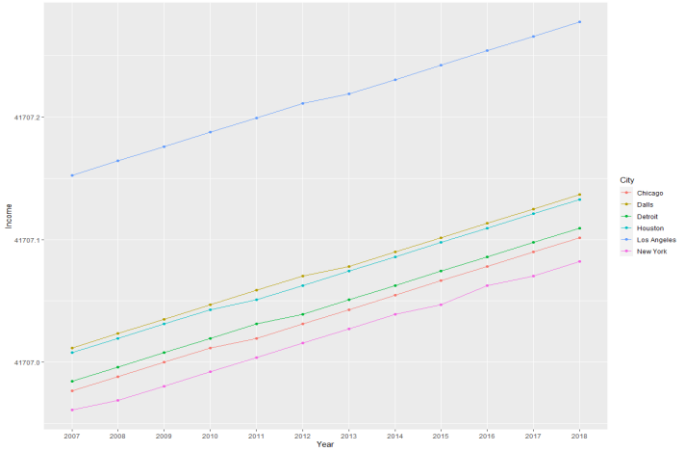

Fig. 4. Predicted Income for each city/year pair.

Figure 5 offers the actual mean response for all city/year pairs garnered from the CPS data. The models appear to predict the large swings occurring in 2011 and 2015 by ever slightly changing the predicted income slope but doesn't have a great fit without income data to use in prediction. The end prediction result is low compared to mean income for the cities.

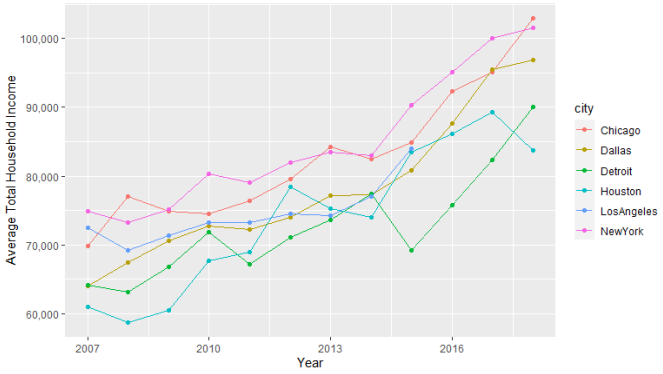

Fig. 5. Mean Income for each city/year pair.

VII. Conclusion

SageMaker does an effective job at hypertuning and feature engineering, even selecting the right type of machine learning algorithm for the given problem. It handles missing data admirably, by either dropping observations, or imputing results (this can be tweaked if needed). It does not, currently, offer a complete solution for someone with little to no machine learning experience. One can't simply offer a dataset and expect accurate predictions without at least some rigor and thought placed in the input. SageMaker's prediction results, given biased or inappropriate data, can be misleading, or just plain wrong in worst case scenarios.

Our intention was to use the simplest input possible to see how SageMaker responds, and while the results look good on paper (exemplary performance metrics), SageMaker does require data pre-processing and a correctly formed training set to be effective. The input data we provided contained several detrimental variables which can be interpreted as noise, such as: IDs for households, counties, locations and other variables which would be best to leave out of a prediction algorithm. This could be a frequent beginner mistake by ML/AI novices eager to find a magic bullet for prediction. Given the income categories, our model performs exceptionally, however, one doesn't need a machine learning algorithm to predict total income given the separate income statistics!

If we were to run this study again from the beginning, the process would drastically be changed. Exploratory data analysis and model training were initially run in parallel, with the though that we could update the model accordingly as we progressed. This process likely resulted in the poor performance of the model as the iterations between EDA and model training were not as seamless as we had hoped. A more efficient solution would have been to hold back on any model training until we had manually implemented our insights discovered through EDA. If this were the case, retraining of the model would not have been as gargantuan of a task.

Our original question sought to answer whether certain counties recovered from the 2008 financial crisis, and the visual plots suggest they did, in fact, recover but our untuned SageMaker model doesn't give us confidence to definitively answer this research question.

ACKNOWLEDGMENT

We would like to thank Dr. Sohail Rafiqi for his continued guidance and education that made this research possible.

REFERENCES

[1] US Census Bureau. *Current Population Survey Datasets: Annual Social and Economic Supplements*. (2007-2018). [Online] Available: https://www.census.gov/programs-surveys/cps/data/datasets.html

[2] H. Wikham. (2016, Mar 29). Feather: A Fast On-Disk Format for Data Frames for R and Python, powered by Apache Arrow. *RStudio Blog*. [Online] Available: https://blog.rstudio.com/2016/03/29/feather/

[3] Julien Simon. (2019) Youtube. *NEW! Amazon SageMaker Studio*. Available: https://www.youtube.com/playlist?list=PLJgojBtbsuc0MjdtpJPo4g4PL8mMsd2nK

[4] Amazon. *AWS SDK for Python (Boto 3) Documentation*. Available: https://docs.aws.amazon.com/pythonsdk/?id=docs_gateway

[5] Amazon. *Amazon SageMaker Documentation*. Available: https://docs.aws.amazon.com/sagemaker/?id=docs_gateway

[6] United States Patent and Trademark Office, *Patenting In U.S. Metropolitan and Micropolitan AreasRegional Components January 2000 -- December 2015 Reports*, Available: https://www.uspto.gov/web/offices/ac/ido/oeip/taf/cls_cbsa/cbsa_countyassoc.htm