Tutti Frutti:Income Prediction by Zipcode

Domenick Powers; Rommel Trejo; Min Jei Yeo; Kai Lau

Abstract:

This paper is about the approach used to create a program that can predict the income a zip code will earn in a given year. Through the data given by the IRS we can create a set of training data to feed our program. Linear regression is the best way to figure this problem out. Creating the dataset was the most difficult part of this project and finding out the proper piece of data to use. We decided on AGI (Adjusted Gross Income) to use as our statistic. This statistic is how much money is brought home by a person after deductions. We decided to use python as our programming language and pandas and scikit to do the data science. These tools are robust and allow us to see all of our data in a way that would let us make the best prediction. The prediction that we can create is fairly accurate and can predict how much money that a zip code will make using a decision tree. Linear regression is still the correct way to go with this problem as it will create the most accurate estimate and will be able to handle more information. This project could be expanded to also include more types of integer data as that is not important to the implementation. This paper could be used by different government agencies to see which areas of Maryland need more development.

Introduction:

An interesting topic in artificial intelligence is prediction models. It is highly interesting because it can help us predict the future and help us understand how and why trends happen. There are many different fields that use predictions in order to function, some of the most popular tools that can make predictions are based on money. Stock market fluctuations are predicted using supercomputers that try to predict trends in markets. Faulty predictions are capable of bringing down entire economies, the idea that a machine can be made to predict money exchanges is an interesting one and one that we tackled in this paper.

Our topic of research was income based on zip codes in Maryland. This was a topic that interested us because the information is readily available and there are many different pieces of data that could correlate to a range of income. One could look at population demographics, local industry, or availability of jobs to try to predict how much money a group of people will make. However, we decided to focus on AGI per zip code to try to predict income made in a certain year. In order to complete this, we created a regression model in order to make our predictions. This type of work would fit in the category of machine learning in AI.

The data that we are using will not necessarily be the only type of data that our machine will be able to ingest. As long as the type of data in the CSV is an integer the machine should be able to make predictions.

Another consideration that we wanted to take into consideration is the ability for user interaction. The user should be able to enter a zip code that they are interested in and get a prediction back. This will allow the data to be self filtering and also make the runtime faster as instead of calculating 402 different zip codes, the machine will only have to digest the information for one zip code.

Background/related work:

In order to make predictions based on zip code, the first thing that we must do it collect the data. We collected a number of years of IRS income data from the IRS as most of the data is available to the public. This data is the backbone of the project. The data that comes from the IRS contains a lot of information that was not useful in the scope of this project. Thus, one of the largest portions of our work was figuring out what data we could use to create predictions. We settled on AGI (adjusted gross income) which is how much money a person brings home after deductions. This seemed like the best indicator of wealth for a zip code as it is the money that is available for the people to use and that is how most people measure wealth.

After doing research on which models are best for this problem and trying different algorithms in Weka , we settled on linear regression. This fit our problem the best because of the types of data we were using. Creating a scatter plot of our data is possible which means that linear regression is one of the best options for creating a prediction model. We also poked around at decision trees to create a prediction but the data did not support this method as well. Another model that we looked at was nearest neighbor but once again we decided that we were not working with different classifiers but in fact we were working with numbers.

Our next step of research was to decide on what tools to use in order to get a model running. We decided that python would be our language of choice as these models would not take a long time to create so run time is not an issue. We also looked into different libraries that could be used in order to create the linear regression model. We settled on pandas in order to digest the data and Scikit to create the different models. These two tools are very powerful data science tools that are also pretty simple to use. These tools also allow for data withholding in order to make sure that the models they create are accurate and can give a lot of information about the models they create. We also decided to use Weka to help visualize the data that we were digesting.

In addition, We looked into the importance of flexibility in a program like this and found out that programs like this are generally pretty flexible with some exceptions. There are machines that will only accept one type of data from one data source. An example of a machine like this is the machines that predict stock market prices. These machines are required to be fairly locked down as a breach in security could cause a national terror. Our machine, on the other hand, is not as important thus we can try to have it digest many different sources and types of data.

Finally, we looked into different ways to test the accuracy of the data and settled upon using Weka as a tool to find accuracy. The machine itself would be able to find its own accuracy if we had more data points to work with. Because we have so few data points available, since 1998 to be specific, we must use them all in order to create the linear regression model. This is a problem that we ran into on multiple occasions for different problems. We decided that this was not a big issue because all of the data came from the IRS, the best option when it comes to sources for revenue data.

Approach:

In order to create the regression, we used Pandas and Scikit to look into the data to find the best line of best fit. In order to use these tools we used a tool called Jupyter Notebook in order to run the Python code; nonetheless, a traditional python file is included. Jupyter is a Python IDE that is capable of running one line of Python code at a time. This was especially useful in this project as it allowed for us to examine the data at every step of the compilation. Being able to see the data allowed us to estimate how accurate the program was in creating a prediction.

We had to alter the format of the data multiple times in order to have the programs work the CSV's. We used one CSV of cleansed data from the IRS that contained 402 zip codes and 11 years of data. The years of data were not continuous as some years were held back from the public on the IRS website.

The way that we set up the data is income is found along the y-axis and the year is situated along the x-axis. From this, we will be able to make a prediction based on year. We also experimented with income along the x-axis to see if we could predict the year but this ended up being non-important so it was cut from the final implementation.

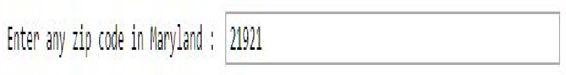
The program will export the formulas that are created by zipcode in order to be stored and examined in the future. This export feature will also allow us to more easily test accuracy as Weka will use a formula in order to create the mean squared difference. This mean squared difference can be used in order to calculate error which will tell us how far off our prediction was.

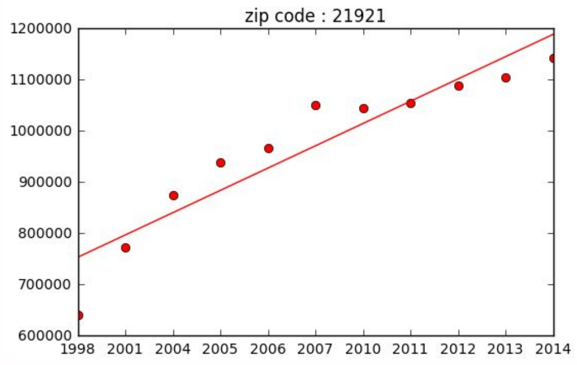
Results:

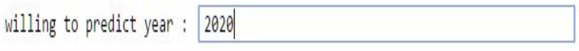
Our the only data set was from IRS. IRS opens well organized tax data from 1998 to 2014 with some years are missing. We could use eleven years of tax data. Each year of tax data, except a couple of years, contains detailed information about tax in Maryland specified by zip code. For example, taxable interest, business net income, unemployment compensation, or real estate taxes. Since a couple of old data sets did not have the same format of data, we had to use AGI which was included in every year data. So we collect only AGI data from each data and made one spreadsheet that contains year in a column and zip code in a row. Finally, convert the spreadsheet to CSV version and we used the data set throughout all of our implementation.

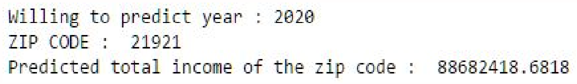
The following plot is from simple linear regression model of our total data which contains a sum of every zip code data. Our data has about 0.9 of R2(mean square) value which is 90% of accuracy with the regression. Also, we found about p-value which is used in probability to test the hypothesis and typically hypothesis with less than p-value of 0.05 is acceptable. From our data, we had a 1/100000 p-value. Some zip code data was much higher than that but still less than 0.05. So we could assume that our regression model works.

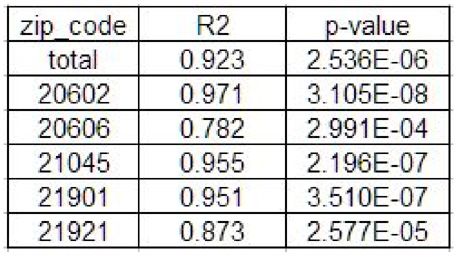
This is the way our program set up. The user can enter any zip code in Maryland.
And the program makes a linear regression model for the zip code. Once the model for the specified zip code is built, then the user can enter the year he/she wants to explore.


Enter any zip code in Maryland : 21921




willing to predict year : 2020


```
Willing to predict year :  2020
ZIP CODE :   21921
Predicted total income of the zip code :   88682418.6818
```



zip_code	R2	p-value
total	0.923	2.536E-06
20602	0.971	3.105E-08
20606	0.782	2.991E-04
21045	0.955	2.196E-07
21901	0.951	3.510E-07
21921	0.873	2.577E-05

Conclusion/ Improvement:

First of all, during our experiment, we realized that our data set was not enough to train a machine. While we collect the data, we had eleven years of data from IRS and each data file contains a variety of tax data. So we thought that was way enough data for our project. However, since our goal of the project was predicting income by each zip code, eleven years of every zip code's AGI is only data we had to use. As a result, we only had eleven data points to train a machine which does not very inefficient. Still, we could use simple linear regression model instead of machine learning method to test our data and what we learned from the class. And even with eleven data points, simple linear regression model seems worked.

Another pitfall of the data is, income normally increases consistently because our economy is usually growing. Here is a simple example, what can a person buy with ten dollars today in comparison to twenty years ago? During our data research, we found GDP(Gross Domestic Product) data from BEA(Bureau of Economic Analysis) and very interestingly their GDP data has raw GDP and "real" GDP. Real GDP means convert monetary values to 2009 currency, so people can actually compare the old data to today's data. Thus, predicting income in ten years or even further maybe useless since we do not know how will the economy look like in the far future and can not compare the future's currency to today's dollar, yet.

One way to make our project more useful would be to use multi-variable regression instead of linear regression. For our experiment, year and AGI were used as x and y-axis. However, we can introduce more variables to our model. For example, GDP, poverty rate, some index of the job market, or population changing. Once we find any relationship between our target which is income mapping to these variables, then we could do multidimensional regression. That will give us a more precise prediction for the near future.

References:

Anon. Python Data Analysis Library¶. Retrieved December 22, 2016 from http://pandas.pydata.org/

Anon. scikit-learn. Retrieved December 22, 2016 from http://scikit-learn.org/stable/

Anon. 2016. How to Run Your First Classifier in Weka. (2016). Retrieved December 22, 2016 from http://machinelearningmastery.com/how-to-run-your-first-classifier-in-weka/

Anon. Forms and Pubs. Retrieved December 22, 2016 from https://www.irs.gov/

@turbotax. What is Adjusted Gross Income (AGI)? Retrieved December 22, 2016 from https://turbotax.intuit.com/tax-tools/tax-tips/Taxes-101/What-is-Adjusted-Gross-Income--AGI--/INF19180.html

Anon. Definition of Adjusted Gross Income. Retrieved December 22, 2016 from https://www.irs.gov/uac/definition-of-adjusted-gross-income

Anon. sklearn.linear_model.LinearRegression¶. Retrieved December 22, 2016 from http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html