Machine Learning Engineer Nanodegree

Capstone Project

Jorge Bartra

August 5th, 2018

I. Definition

1) Project Overview

The project is a real world scenario for a Non-Profit organization. The organization is facing a challenge today because there are more than 100K potential donors that are currently not being included by the Marketing department in their campaigns. The 100K potential donors has been overlooked or not used because there is a lack of knowledge in regards to Machine Learning skills.

Currently the Non-Profit organization requires additional source of income in order to generate extra revenue and expand charity missions around the globe and also support and increase disaster recovery campaigns.

Machine Learning is been used in all types of business. Non-profit organizations are not detached from Machine Learning at all. Rather they are using it to increase the amount of donors by executing marketing campaigns constructed with Machine Learning algorithms. These campaigns are based on specific demographics such as particular interest, propensity, financial status and some others (1).

This is the reason why this project will not be the first one on its class since Machine Learning has been used already to increase the donation to Non-Profit organizations. The success of machine learning applied to non-profit organizations rely on the amount and quality of the data used and that is the major concern today because Non-Profit organization do not collect the data efficiently (3).

The model to use falls under Classification type therefore a Precision Model will be used. The dataset has several different classes and this architecture works well with Classification type models. The desired score for Accuracy is at least 80% because the data is well balanced, see below the analysis:

```
Total number of records: 8996
Zip Code where median income is greater $60,000: 4783
Zip Code where median income is at most $60,000: 4215
Percentage of median by zip code making more than $60,000: 53.1680746999%
```

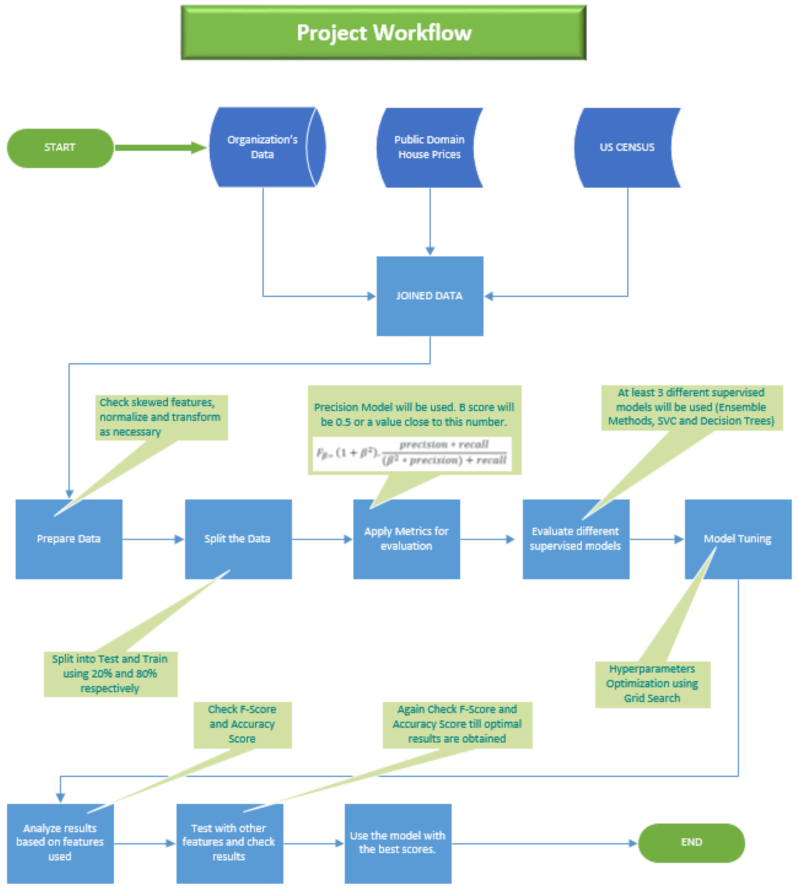
2) Problem Statement

There are different sources of data distributed in different servers and databases in the organization. The data can be combined into one single temporary table and stored for further analysis. Using Stored Procedures will save time next year since is highly probable that the model will be executed again.
The data presents a challenge because the data does not have quantitative features that could be used in the model such as "Income", therefore it is necessary to obtain data from a public domain that after is joined to the organization's data, it could produce a quantitative feature for every single record that is obtained within the organization's databases.
The model planned to be use is a Regression Model since the dependent variable is continuous. The output desired for this model is the Mean Income/ ZCTAs. After optimization the desired Accuracy and F-score would be close to 85% (+-).

The goal is to create additional revenue to a non-profit organization using data that is already stored in the organization's servers but it does not have all features needed to obtain a good prediction model. The initial tasks involved are as follows:

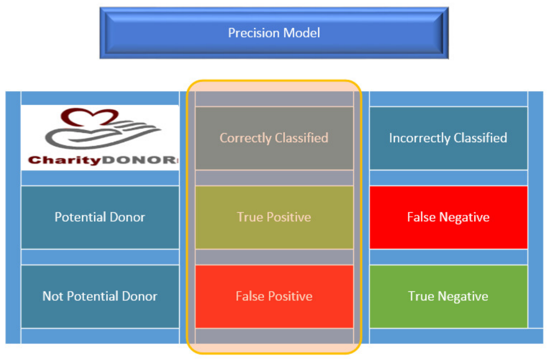
a. Download data from the below links:
   I. Census Data.
      https://www.census.gov/programs-surveys/decennial-census/data/datasets.2010.html
   II. Redfin Data Center.
      https://www.redfin.com/blog/data-center
      The problem specific to this dataset is that it is stored in a TDE extension file and it is embedded into a Tableau Workbook
   III. Merge the data from the 2 SQL servers in the organization.
b. Prepare and clean the data.
c. Inspect the data – Check for skewness, normalize the data if needed.
d. Split the data into training (80%) and testing (20%) sets.
e. Apply metrics for evaluation.
f. Evaluate 3 different supervised models.
g. Implement Model tuning.
h. Analyze the results based on the features used in the model and check which feature is more likely to contribute to the desired model outcome.
i. Test with other features or less features and observe the results and compare.
j. Use the model with the best score.

The model will use other features such as education, age, occupation, household income, etc. therefore the intention is not to drop everyone that makes higher than less than 60K/year but rather combined all the most relevant features in the data and obtain the best F score and Accuracy values.


Project Workflow
START
Organization's Data
Public Domain House Prices
US CENSUS
JOINED DATA
Check skewed features, normalize and transform as necessary
Precision Model will be used. B score will be 0.5 or a value close to this number.
$F_\beta = (1+\beta^2).\frac{precision * recall}{(\beta^2 * precision) + recall}$
At least 3 different supervised models will be used (Ensemble Methods, SVC and Decision Trees)
Prepare Data
Split the Data
Apply Metrics for evaluation
Evaluate different supervised models
Model Tuning
Split into Test and Train using 20% and 80% respectively
Hyperparameters Optimization using Grid Search
Check F-Score and Accuracy Score
Again Check F-Score and Accuracy Score till optimal results are obtained
Analyze results based on features used
Test with other features and check results
Use the model with the best scores.
END

3. Metrics Problem Statement

The intention is to use Precision Model rather than Recall. Below is the confusion matrix elaborated for the project:


Precision Model
CharityDONOR

	Correctly Classified	Incorrectly Classified
Potential Donor	True Positive	False Negative
Not Potential Donor	False Positive	True Negative

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

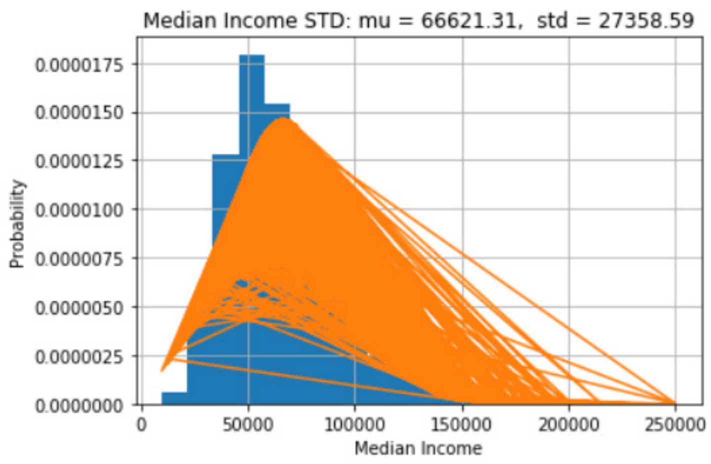
Since it is a Precision model then it is necessary to use the B score with a value of 0.5 using the below formula:

$$F_{\beta=} (1 + \beta^2).\frac{precision * recall}{(\beta^2 * precision) + recall}$$

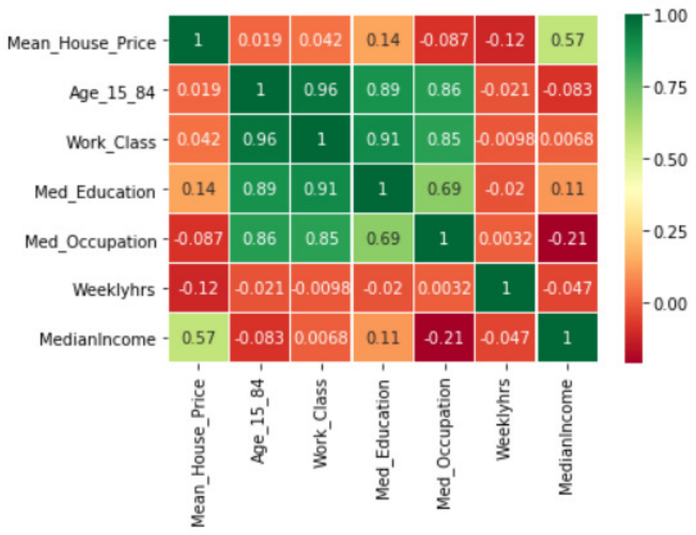
II. Analysis

1. Data Exploration

   The target variable distribution shown below has mean equal to 66K and the target variable chosen for this model select income greater than 60K which is a good indicator.



The heat correlation map shows interesting facts related to the dataset. There is high correlation between education and age, education and work_class, good correlation between education and occupation and positive correlation between House Prices and Median Income.



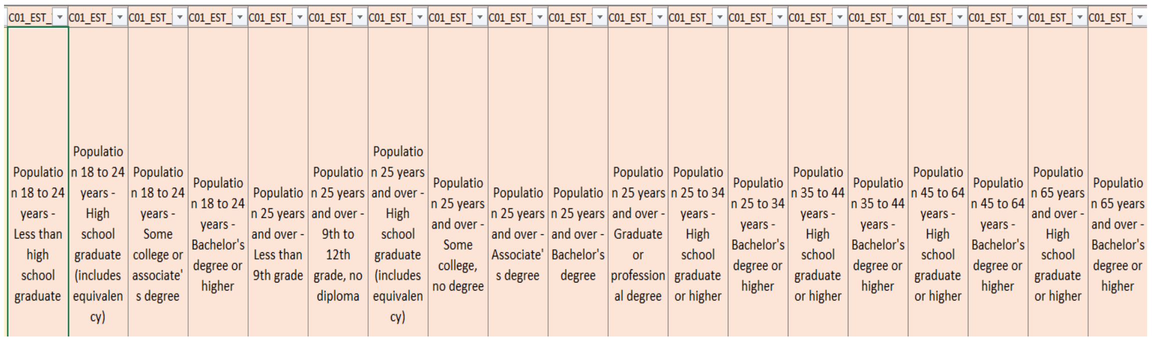
	Mean_House_Price	Age_15_84	Work_Class	Med_Education	Med_Occupation	Weeklyhrs	MedianIncome
Mean_House_Price	1	0.019	0.042	0.14	-0.087	-0.12	0.57
Age_15_84	0.019	1	0.96	0.89	0.86	-0.021	-0.083
Work_Class	0.042	0.96	1	0.91	0.85	-0.0098	0.0068
Med_Education	0.14	0.89	0.91	1	0.69	-0.02	0.11
Med_Occupation	-0.087	0.86	0.85	0.69	1	0.0032	-0.21
Weeklyhrs	-0.12	-0.021	-0.0098	-0.02	0.0032	1	-0.047
MedianIncome	0.57	-0.083	0.0068	0.11	-0.21	-0.047	1

For this project 3 different datasets were used:

1. SQL Server Data – this data is the organization's data. It contains Names, Last Names, Phone Numbers, Addresses, and Zip Codes. Unfortunately no other demographic information is available.

2. Sale House Prices – this data comes from a public domain. The links are provided in the project proposal. The data was downloaded in a TDE extension file. It was embedded in a Tableau workbook, therefore it was challenging to extract it. The size of the data extract was 1.28 GB and the computers I used ran out of memory during the extract. There was no other way to extract the data, even though different techniques were used. At the end it had to be manually copied by chunks and paste it in Excel and then converted to CSV. This dataset contains House Sale Price, Zip Code, City, State and Country and some other fields no relevant on the model. The data was splitter into data with ZIP code and without ZIP code so again it was a little tedious to clean up and prepare the data.

3. Census Dataset – this dataset was obtained from the CENSUS website. It was a little challenging to clean up and prepared the date because of the nature of the data. For example for age the below fields were available. Some of the buckets were removed for obvious reasons such as population less than 18 years old since it is unlikely that they could be potentials donors. For the other buckets the MEDIAN was calculated by ZIP code.



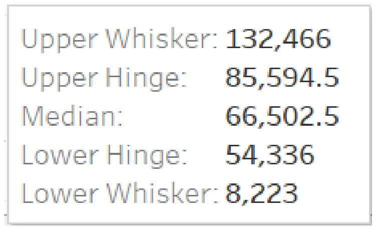
C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_	C01_EST_
Population 18 to 24 years - Less than high school graduate	Population 18 to 24 years - High school graduate (includes equivalency)	Population 18 to 24 years - Some college or associate's degree	Population 18 to 24 years - Bachelor's degree or higher	Population 25 years and over - Less than 9th grade	Population 25 years and over - 9th to 12th grade, no diploma	Population 25 years and over - High school graduate (includes equivalency)	Population 25 years and over - Some college, no degree	Population 25 years and over - Associate's degree	Population 25 years and over - Bachelor's degree	Population 25 years and over - Graduate or professional degree	Population 25 to 34 years - High school graduate or higher	Population 25 to 34 years - Bachelor's degree or higher	Population 35 to 44 years - High school graduate or higher	Population 35 to 44 years - Bachelor's degree or higher	Population 45 to 64 years - High school graduate or higher	Population 45 to 64 years - Bachelor's degree or higher	Population 65 years and over - High school graduate or higher	Population 65 years and over - Bachelor's degree or higher

4. The number of rows in each dataset is as follows:
   - Combined table using SQL servers: 102510
   - Census Data: 14519 rows (These rows are aggregated at the ZCTA level)
   - RedFin Data Center: Original 4.2 M but after cleanup 15700 ((These rows are aggregated at the City and Zip Code level)

   There are 14519 distinct target variables but when this dataset is joined to the SQL table it will be 102510 target variables since the SQL Server data is at the lowest level (SYSTEM ID or potential donor level)

   For categorical variables, one-hot encoding algorithm seems to work very well for the type of features the model will use.

   The dataset distribution seems to be balanced and it will be analyzed using a simple Box Plot graph or using python scripts. In the screen shot below they is a simple Box Plot.



Upper Whisker:	132,466
Upper Hinge:	85,594.5
Median:	66,502.5
Lower Hinge:	54,336
Lower Whisker:	8,223

2. Data Challenges
   A. Census Dataset – The Median Income was calculated using year 2017 and 2018's data. Previous income data was corrupted, missing a lot of values, many NULSS found or simply the data did not sense. Difference between "Median Income" values for some ZCTAs between year 2016 and 2014 were too high, in some cases the percentage increase from 2014 to 2016 was more than 50% which is wrong. To avoid misleading information only most recent data was used (years 2017 and 2018 the rest was removed from the dataset)

   B. Sale House Prices – The Median Income was the most relevant feature. It was calculated and it also represented a challenge because the data obtained from the public domain was not 100% accurate. After the initial analysis, more than 50K records had to be removed from the dataset because they do not have the corresponding house sale prices.

      a. In order to reflect the most accurate results, the date range used was from 2017-2018 period. The date range is the most accurate "Median House Sale Price" to date.

      b. The data obtained from the public domain was a mixture of Tableau Aggregated Measures and Dimensions. The measures were easily disaggregated but the dimensions could not be disaggregated because the nature of the data and Tableau behavior (Cube), therefore it was not an option. The next option was to filter the data using Tableau filters and calculated formulas and then export it into an excel spreadsheet. Unfortunately several errors were raised by the computer system resources. The messages received were all OUT OF MEMORY related.

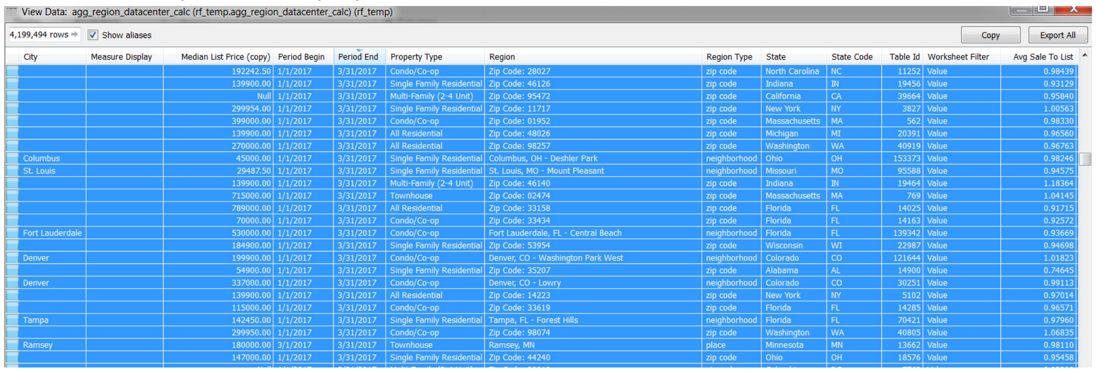
      c. Tableau 'Data Export' feature was then used as the last alternative. After the export task was performed I had a CSV file saved in the File Share with a 1.28 GB in size. While preparing the data for the model a very important scenario was overlooked. Since Excel has a 1 million row cap, the rest of the data did not load. The row count was more than 4.2 million of rows but it was not noted after it was too late. When I run the model I was getting an Over-Fitted model with 100% on Accuracy and F-Score because the data was not just incomplete but there were a lot of duplicated records and some other invalid data such as NULLs on BLANKs.

      d. Because of the data was bad and the Model Scores were really bad, it was necessary to start again from the scratch.

e. The next featured used to extract the data was the use of the 'Load Query to Excel Data Model' option in Excel, but it did not work because the two computers used run out of memory.

f. After trying to download the data for a couple of days, Tableau Data viewer had to be used. The data was sorted manually by date and then copied and pasted into an excel worksheet. It took around ½ hours just to copy one year worth of data and past the data into Excel.

g. Below is a sample data displayed in Tableau Viewer:



View Data: agg_region_datacenter_calc (rf_temp.agg_region_datacenter_calc) (rf_temp)

4,199,494 rows ➩ ☑ Show aliases

City	Measure Display	Median List Price (copy)	Period Begin	Period End	Property Type	Region	Region Type	State	State Code	Table Id	Worksheet Filter	Avg Sale To List
		192242.50	1/1/2017	3/31/2017	Condo/Co-op	Zip Code: 28027	zip code	North Carolina	NC	11252	Value	0.98439
		139900.00	1/1/2017	3/31/2017	Single Family Residential	Zip Code: 46126	zip code	Indiana	IN	19456	Value	0.93129
		Null	1/1/2017	3/31/2017	Multi-Family (2-4 Unit)	Zip Code: 95472	zip code	California	CA	39664	Value	0.95840
		299954.00	1/1/2017	3/31/2017	Single Family Residential	Zip Code: 11717	zip code	New York	NY	3827	Value	1.00563
		399000.00	1/1/2017	3/31/2017	Condo/Co-op	Zip Code: 01952	zip code	Massachusetts	MA	562	Value	0.98330
		139900.00	1/1/2017	3/31/2017	All Residential	Zip Code: 48026	zip code	Michigan	MI	20391	Value	0.96560
		270000.00	1/1/2017	3/31/2017	All Residential	Zip Code: 98257	zip code	Washington	WA	40919	Value	0.96763
Columbus		45000.00	1/1/2017	3/31/2017	Single Family Residential	Columbus, OH - Deshler Park	neighborhood	Ohio	OH	153373	Value	0.98246
St. Louis		29487.50	1/1/2017	3/31/2017	Single Family Residential	St. Louis, MO - Mount Pleasant	neighborhood	Missouri	MO	95588	Value	0.94575
		139900.00	1/1/2017	3/31/2017	Multi-Family (2-4 Unit)	Zip Code: 46140	zip code	Indiana	IN	19464	Value	1.18364
		715000.00	1/1/2017	3/31/2017	Townhouse	Zip Code: 02474	zip code	Massachusetts	MA	769	Value	1.04145
		789000.00	1/1/2017	3/31/2017	All Residential	Zip Code: 33158	zip code	Florida	FL	14025	Value	0.91715
		70000.00	1/1/2017	3/31/2017	Condo/Co-op	Zip Code: 33434	zip code	Florida	FL	14163	Value	0.92572
Fort Lauderdale		530000.00	1/1/2017	3/31/2017	Condo/Co-op	Fort Lauderdale, FL - Central Beach	neighborhood	Florida	FL	139342	Value	0.93669
		184900.00	1/1/2017	3/31/2017	Single Family Residential	Zip Code: 53954	zip code	Wisconsin	WI	22987	Value	0.94698
Denver		199900.00	1/1/2017	3/31/2017	Condo/Co-op	Denver, CO - Washington Park West	neighborhood	Colorado	CO	121644	Value	1.01823
		54900.00	1/1/2017	3/31/2017	Single Family Residential	Zip Code: 35207	zip code	Alabama	AL	14900	Value	0.74645
Denver		337000.00	1/1/2017	3/31/2017	Condo/Co-op	Denver, CO - Lowry	neighborhood	Colorado	CO	30251	Value	0.99113
		139900.00	1/1/2017	3/31/2017	All Residential	Zip Code: 14223	zip code	New York	NY	5102	Value	0.97014
		115000.00	1/1/2017	3/31/2017	Condo/Co-op	Zip Code: 33619	zip code	Florida	FL	14285	Value	0.96571
Tampa		142450.00	1/1/2017	3/31/2017	Single Family Residential	Tampa, FL - Forest Hills	neighborhood	Florida	FL	70421	Value	0.97960
		299950.00	1/1/2017	3/31/2017	Condo/Co-op	Zip Code: 98074	zip code	Washington	WA	40805	Value	1.06835
Ramsey		180000.00	3/1/2017	3/31/2017	Townhouse	Ramsey, MN	place	Minnesota	MN	13662	Value	0.98110
		147000.00	1/1/2017	3/31/2017	Single Family Residential	Zip Code: 44240	zip code	Ohio	OH	18576	Value	0.95458

h. Finally the data was loaded in Excel (only 2017-2018 data). The data was then imported into a SQL Server table using SQL ETL SSIS and joined with the rest of the data.

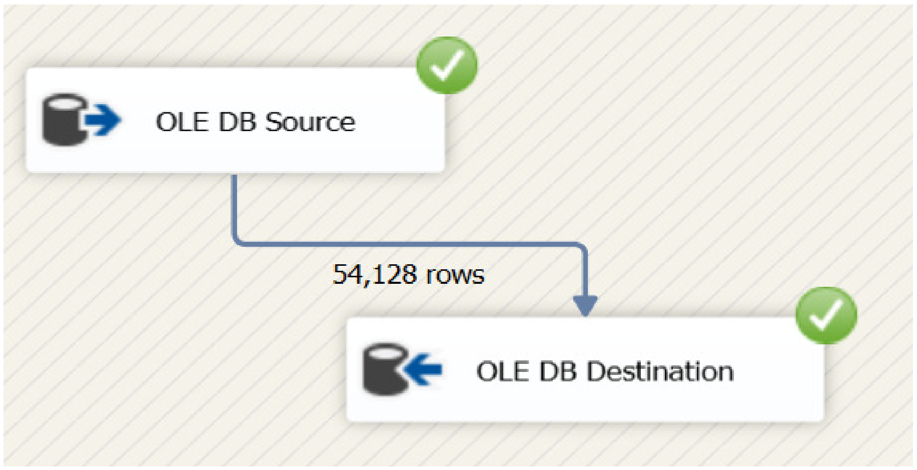
C. SQL Server Data/Organizations Data – The data from the organization was obtained from two different servers. It was imported into a Development Data Warehouse environment and then joined with the Data Warehouse data. It was not difficult to join the data from the two server since it was just using SQL scripts.

3. SQL Code used for the Data Preparation

It was necessary to create a temp table in the SQL Database since the regular tables could not be used. The temp table was created using SQL Integration Services (SSIS) using the SQL script shown below:

```
WITH PotentialDonors (constituentsystemid, constituentlookupid,PRIMARYADDRESSCITY, PRIMARYADDRESSSTATE,
primaryaddressstateabbreviation,
PRIMARYADDRESSCOUNTRY, zipcode, MeanPrice) AS
(
select A.constituentsystemid, A.constituentlookupid,
a.PRIMARYADDRESSCITY, a.PRIMARYADDRESSSTATE, a.primaryaddressstateabbreviation,
a.PRIMARYADDRESSCOUNTRY,

CONVERT(INT,
CASE WHEN IsNumeric(CONVERT(VARCHAR(12), substring(a.PRIMARYADDRESSPOSTCODE,0,6))) = 1
            then CONVERT(VARCHAR(12), substring(a.PRIMARYADDRESSPOSTCODE,0,6))
            else 0
            End) zipcode,
 MaxMeanPrice.MeanPrice

from bbdw.dim_constituent A

join dbo.OLEDBDestination B
on
A.constituentlookupid = B.lookupid

join
(
select city, state, max([mean price]) MeanPrice from dbo.HousePrices
group by city, state
) as MaxMeanPrice
on
(A.PRIMARYADDRESSCITY = MaxMeanPrice.city
and
A.PRIMARYADDRESSSTATE = MaxMeanPrice.state)

where a.PRIMARYADDRESSPOSTCODE is not null
and a.PRIMARYADDRESSCOUNTRY <> 'No Country' and a.PRIMARYADDRESSSTATE <> 'No State'
and a.PRIMARYADDRESSPOSTCODE <> '0' and a.PRIMARYADDRESSCOUNTRY = 'United States'
)

select * from PotentialDonors
order by 7
```

The total record count transferred to the new temp table was 54128. After cleaning up the data further such as eliminating nulls, converting float type values into integers and removing unnecessary features such as Country, City, State, State Abrevation and Lookup ID the total unique record count is 50392



As mentioned before the first attempt to obtain a good Accuracy and F-Score values was a failure because 100 % obtained both. I could not find the error with the Python Code therefore I started looking at the data and I found interesting facts, for example I noticed some duplication and missing data.

After that I connected to SQL server and using SSIS I was able to move the data from the CSV file into a table in SQL server.

The next challenge found was how to calculate the MEDIAN House Sale Price for a single Zip Code, otherwise the results of SQL joins would create multiple instances of the same row in the table. The SQL below was created to calculate the median House Sale Price by Zip Code and then in the Main SQL (Outer SQL) I had to extract the unique instances using the Group By clause together with the MAX function would select one single instance. I could have used FIRST or LAST function as well.

I had to join two different datasets from the Median Sale House Price data. The first set had the Zip Codes therefore I joined the SQL Server data with the Median House Price data using the ZIP Code as key field.

The SQL is shown below:

STEP#1

```
select max(xyz.MedianContCalc)Unique_MedianSalePrice, xyz.zipcode,
xyz.state,xyz.statecode
from
(
	SELECT zipcode, state, statecode,MedianSalePrice,
	PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY MedianSalePrice)
	OVER (PARTITION BY zipcode) AS MedianContCalc
	FROM dbo.housepricewithzipcode
) xyz
group by xyz.zipcode, xyz.state,xyz.statecode
```

The second dataset from the same Median Sale House Price data did not have Zip Codes therefore I had do basically the same thing I did above. In this case was a little more challenging to create the data because I had to exclude the previous data obtained from the above SQL first and then I had to join that data with the SQL server data using the City and State as key fields. The SQL has different levels due to the complexity of the data. Each level has been documented in the same SQL below:

STEP#2

```
select
max(FinalData.MedianContCalc)Unique_MedianSalePrice,FinalData.PRIMARYADDRESSCITY,FinalData.PRIMARYADDRE
SSPOSTCODE, FinalData.PRIMARYADDRESSSTATE,
FinalData.PRIMARYADDRESSCOUNTRY, finaldata.primaryaddressstateabbreviation
from --1st level to obtain a single or unique row for the City, State and Median Sale Price
	(
		select
xyz.PRIMARYADDRESSCITY,xyz.PRIMARYADDRESSSTATE,xyz.PRIMARYADDRESSPOSTCODE,xyz.primaryaddressstateab
breviation,xyz.primaryaddresscountry,
		PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY xyz.MedianSalePrice)
		OVER (PARTITION BY xyz.PRIMARYADDRESSPOSTCODE) AS MedianContCalc
		from -- 2nd level to obtain the Median Sale House Price for the City and State combination since prices were
different in some of the rows
				(
				select DataRemoved.*, b.mediansaleprice
				from( -- 3rd level to exclude the data selected in the first dataset
					select A.* from  dbo.advocatescleaned A
					where not exists ( -- 4rd Data obtained in the same SQL described on Step 1
							select remzip.PRIMARYADDRESSPOSTCODE from
								(
									select A.*, b.Unique_MedianSalePrice from
									dbo.advocatescleaned A
									join dbo.HPriceZipUnique B
									on
									A.PRIMARYADDRESSPOSTCODE = b.zipcode
								) remzip
					where remzip.PRIMARYADDRESSPOSTCODE = a.PRIMARYADDRESSPOSTCODE)
					) DataRemoved

				join dbo.HousePrice_NoZipCode B
				on
				DataRemoved.PRIMARYADDRESSCITY = b.city
				and
				DataRemoved.PRIMARYADDRESSSTATE = b.state
				) xyz
		)FinalData
group by FinalData.PRIMARYADDRESSCITY,FinalData.PRIMARYADDRESSPOSTCODE,
FinalData.PRIMARYADDRESSSTATE,
FinalData.PRIMARYADDRESSCOUNTRY, finaldata.primaryaddressstateabbreviation
```

The last step to join the Median Sale House Price was to join the two SQLs (Part#1 and Part#2) using the UNION SQL clause to obtain the desired dataset:

```
---------Median House Sale Price with Zip Codes and SQL Data joined
select A.*, b.Unique_MedianSalePrice from dbo.advocatescleaned A

join dbo.HPriceZipUnique B
on
A.[PRIMARYADDRESSPOSTCODE] = b.zipcode


---Median House Sale Price for the dataset based on City, State, State Initials and Country only
union

select
max(FinalData.MedianContCalc)Unique_MedianSalePrice,FinalData.PRIMARYADDRESSCITY,FinalData.PRIMARYADDRE
SSPOSTCODE, FinalData.PRIMARYADDRESSSTATE,
FinalData.PRIMARYADDRESSCOUNTRY, finaldata.primaryaddressstateabbreviation
from --1st level to obtain a single or unique row for the City, State and Median Sale Price
		(
        select
xyz.PRIMARYADDRESSCITY,xyz.PRIMARYADDRESSSTATE,xyz.PRIMARYADDRESSPOSTCODE,xyz.primaryaddressstateab
breviation,xyz.primaryaddresscountry,
        PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY xyz.MedianSalePrice)
        OVER (PARTITION BY xyz.PRIMARYADDRESSPOSTCODE) AS MedianContCalc
        from -- 2nd level to obtain the Median Sale House Price for the City and State combination since prices were
different in some of the rows
                (
                select DataRemoved.*, b.mediansaleprice
                from( -- 3rd level to exclude the data selected in the first dataset
                        select A.* from  dbo.advocatescleaned A
                        where not exists ( -- 4rd Data obtained in the same SQL described on Step 1
                                        select remzip.PRIMARYADDRESSPOSTCODE from
                                                (
                                                        select A.*, b.Unique_MedianSalePrice from
                                                        dbo.advocatescleaned A
                                                        join dbo.HPriceZipUnique B
                                                        on
                                                        A.PRIMARYADDRESSPOSTCODE = b.zipcode
                                                ) remzip
                        where remzip.PRIMARYADDRESSPOSTCODE = a.PRIMARYADDRESSPOSTCODE)
                        ) DataRemoved

                join dbo.HousePrice_NoZipCode B
                on
                DataRemoved.PRIMARYADDRESSCITY = b.city
                and
                DataRemoved.PRIMARYADDRESSSTATE = b.state
                ) xyz
        )FinalData
group by FinalData.PRIMARYADDRESSCITY,FinalData.PRIMARYADDRESSPOSTCODE,
FinalData.PRIMARYADDRESSSTATE,
FinalData.PRIMARYADDRESSCOUNTRY, finaldata.primaryaddressstateabbreviation
```

Finally I joined the two datasets using the Zip Code as key field (Mean Sale House Price with Census Income and other features together) and exported the data into a CSV file.

```
SELECT A.*, B.*
 FROM dbo.IncomeZip A

join dbo.SQLData_HousePrice_Combined B
on
A.Zip = b.PRIMARYADDRESSPOSTCODE
```

5. Algorithms and Techniques
   Model Selection – After some research these are models selected:

   A) Adaboost

   - A real world application is Image and Face Detection. (1) Google uses this algorithm when buyers compare prices with a single upload image into Google website. The application is call GoogleShopper (2)
   - AdaBoost works basically creating a series learners or models avoiding the errors obtained in the previous iteration. The way it works is sequentially and this is the reason

why is able to improve itself after each iteration, the features used in previous runs can be discarded and then use new features from the dataset with more weight.

- When there are noise and outliers ADABOOST does not perform well.
- First we have remove the outliers by normalizing the data. After that we applied one-hot encoded algorithm to get rid of the non-numerical features. This scenario works well with AdaBoost because the prediction is based on decision stump. Another reason for using this model is because it is well known for a very robust algorithm.(3)

  (1) https://www.analyticsvidhya.com/blog/2015/05/boosting-algorithms-simplified/

  (2) https://patents.google.com/patent/WO2015192239A1/en

  (3) https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/

B) SVC

- A real world application is to recognize hand written characters and it is used widely.(1)
- The strengths of SVC is that it reduces or avoid over fitting by regulating the parameters, the kernel can be adjusted to work together with SVC thus obtaining better results (i.e rbf, linear or poly).
- In large tasks, the extensive memory usage is high and due to the algorithmic complexity it becomes a serious issue. The speed and data set size is also a weakness. It also does not work well with discrete data. It can also be extremely slow in Test.(2)
- I think the dataset we have is very small therefore the size should not be an issue in terms of time, also it has several features which makes it perfect for the multivariable data.

  (1) https://data-flair.training/blogs/applications-of-svm/

  (2) https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

C) Decission Trees

- ATS Trust is a company dedicated to maintain, install and remove homes oil tanks used in their heating system. The company uses an answer tool to guide and help people learn all about oil tank removal.(1)
- Decision Tree model is easy to understand. This model is almost the fastest in identifying the most significant variable among other variables. Also data types are not a constrain and it is considered as non-parametric model.
- Decision Trees tends to over fit and it performs poorly with continuous data.
- I used this model since it is easy to understand but I not sure if it will be a good model, given the data we have. I did not want to use other model that we did not cover in the class so far therefore this was my third choice. I think the model will prove itself wrong at the end of this lab, which I think is other way to learn how not to use this model.
    1. http://www.funderstanding.com/blog/real-life-application-decision-trees/

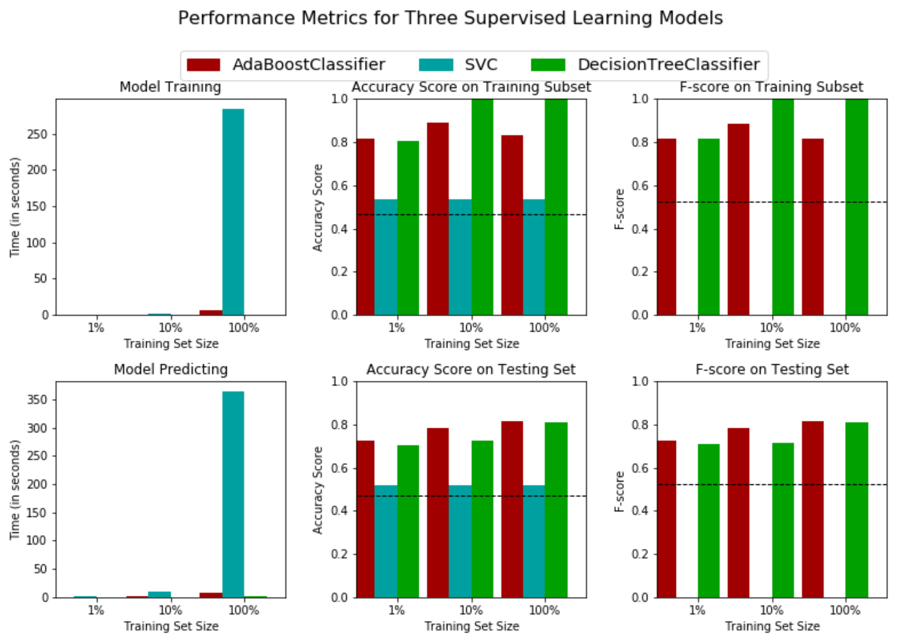
6. AdaBoost Hyper parameters tuning.

   N_estimators – Initially the range of the number of model to train was selected between 60 and 350. Time to execute it was around 12 minutes with Accuracy score = 0.8122 and and F-Score = 0.8131. Then the number of estimators was increased to 450 and the scores were improved to 0.8383 and 0.8393 respectively. Increasing the number of estimators to 450 also increased the training time to 14.5 minutes approx. The next test used 550 number of estimators but the scores decreased and the time training time was more than 17 minutes. Unfortunately patience it needed since the training time after several iterations could take several hours and a good score is worth the effort.The best scored were obtained with 450 number of estimators and training time Learning Rate – The default value was changed several times but no difference was noted in the performance score, therefore several learning rate values were included starting at 0.1 to 1. Performance or training time was impacted but better scores were obtained.

7. Methodology

   The type of model used for this project is a Supervised Model because of the nature of the data. The data has 10 labeled features used as the input and the desired outcome is an Accuracy and F-Score greater than 80%.Three models were trained but the best scores were obtained with Adaptive boosting algorithm (AdaBoost). The selection of hyper parameters described in the previous step was a sort of trial error where the best scores we obtained in the testing set after several trials. The data matrix used as input were represented by X and Y or train and test samples. Because of the reason explained above in Item#5 (Algorithms and Techniques) the models chosen for the dataset were Decision Trees, SVC and AdaBoost.

   After running the performance metrics for the 3 different algorithms the results were as shown below in the graphic:



Support Vector Machine (SVC) algorithm takes too long to run when the sample data is 100% of the data. When SVC performance is compared with AdaBoost and Decision Tree algorithms the graphic shows a really bad or poor performance. Therefore SVC was removed from the list of choices.

Decision Trees algorithm prone to overfitting also shown in the above chart. In conclusion, AdaBoost offers the best performance and also the best scores according to the above charts summary, the Accuracy score and F-Score values for all data samples (1%, 10% and 100%) are above 80%.

AdaBoost works basically creating a series learners or models avoiding the errors obtained in the previous model (2). The way it works is sequentially and this is the reason why is able to improve itself after each iteration, the features used in previous runs can be discarded and use new features from the dataset with more weight (1). The number of models is equal to the number of boosting iterations the user sets up in the n_estimator parameter. When AdaBoost finds the best prediction it stops, even if the number of estimators are less than the parameter value. AdaBoost can be used for regression or classification methods (2).

(1) https://blog.statsbot.co/ensemble-learning-d1dcd548e936 (

2) http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

The model then was tuned using GridSearchCV and the detail of the parameters used are already explain above in the section "Parameter Tuning" (item #6).

8. Issues

There were many issues mostly with the data. The first Accuracy and F-Score obtained after the model tuning was 100%. It was an over fitted model and it was difficult to find the error. The code was checked and re-run it several time but the results were wrong and there was nothing wrong with the code. I created a test dataset with unique values and the scores obtained were much better. The next step was to review the data again. Many errors were found with the data, many duplicated values, null, empty cells, wrong formatting such as numerical features with percentage sign.

The SQL scripts used in the first model has to be revised and modified to exclude duplication and proper joins between the different datasets.

Some other errors were obtained during the coding phase but it was not as bad as the data issues.

III. Benchmark Model

Since the data is not at the individual level but rather at the zip code and city level, the organization is interested in targeting individuals where the house mean sale price is higher than 30K.

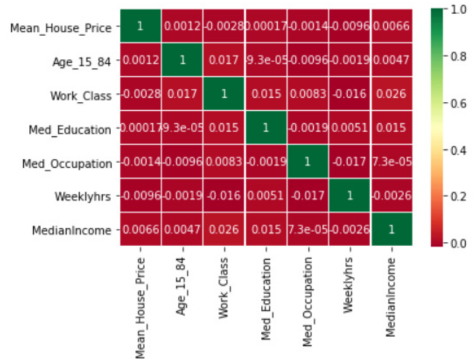
In addition to the house sale price the estimated mean income by zip code will be also used. The targeted individual within a certain zip code where median income is higher than

60K/year combined with the house median sale price will be the two numerical features in this model.

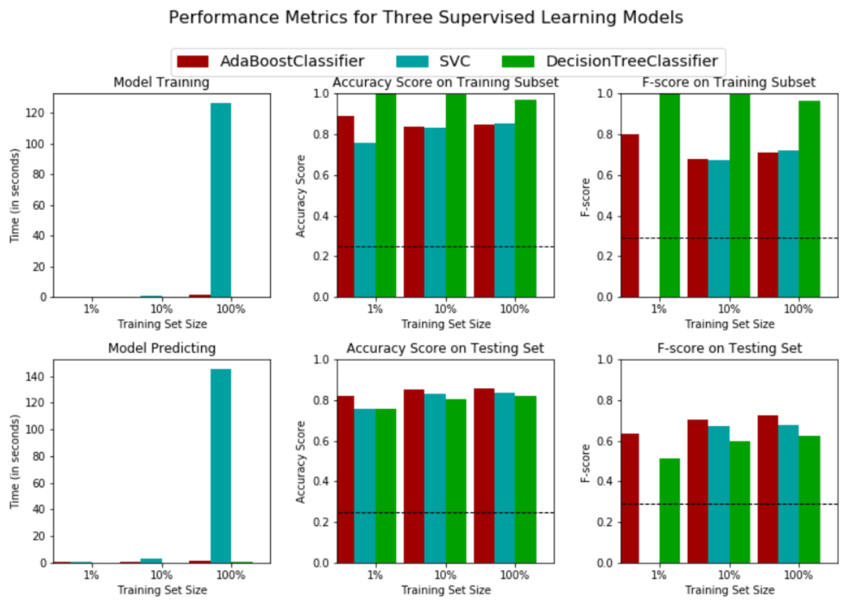
The Naïve Predictor was used to compare the results of the model. The scored obtained with the Naïve Predictor were Accuracy = 0.4685 and F-Score = 0.5243

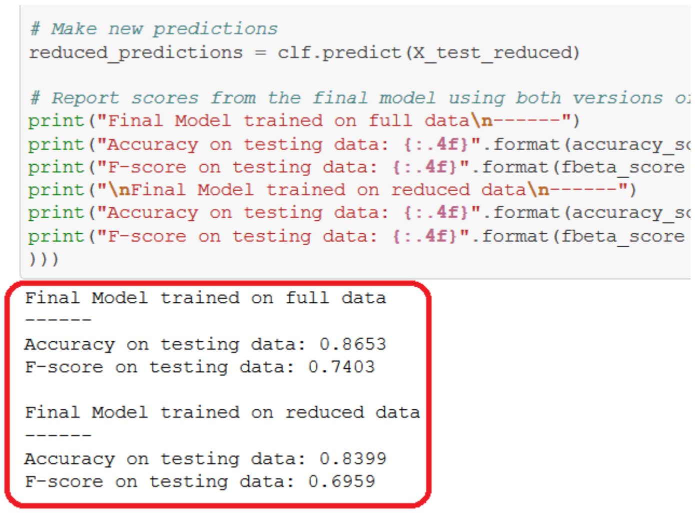
IV. Results

The model was tested with two different datasets. The first dataset was created randomly in excel using the highest and lowest values as range for every single feature in the data. The results are inconsistent. First the dataset is not balanced, the target variable is almost 80% above the Income Benchmark needed for the model (60k). This is causing the Accuracy to be 0.7733% and F-Score very low 0.1111%.There is a problem with misclassification rate and that is probably why the F-Score is too low. The correlation map shows no correlation:



The second dataset used was a sample data from CENSUS. Some of the features has to be modified. For example House Sale Price had to be removed and Capital Gain was added to the model. The results are correct and the model seems to be a robust model. Below is the performance metric obtained with the second data set:




```
# Make new predictions
reduced_predictions = clf.predict(X_test_reduced)

# Report scores from the final model using both versions o
print("Final Model trained on full data\n------")
print("Accuracy on testing data: {:.4f}".format(accuracy_sc
print("F-score on testing data: {:.4f}".format(fbeta_score
print("\nFinal Model trained on reduced data\n------")
print("Accuracy on testing data: {:.4f}".format(accuracy_sc
print("F-score on testing data: {:.4f}".format(fbeta_score
)))
```

```
Final Model trained on full data
------
Accuracy on testing data: 0.8653
F-score on testing data: 0.7403

Final Model trained on reduced data
------
Accuracy on testing data: 0.8399
F-score on testing data: 0.6959
```

Accuracy and F-Score look very good on the second test with a sample data from Census therefore the model looks robust.

1. Benchmark used for this model is as follows

Naïve Predictor were Accuracy = 0.4685% and F-Score = 0.5243%

Current results

The Accuracy and F-Score of the model after optimization is as follows:

Final Model trained on full data

Accuracy on testing data: 0.8383
F-score on testing data: 0.8393

V. Improvement

The data collected for this project was not easy to obtain. The organization needs to work better on collecting the data from the different groups that support the mission. The ideal project would be calculating the model at the individual level, unfortunately the CENSUS data does not offer this type of detail or the data is not available to the public domain. The data in the organization lacks of consistency, some of them have education or income but the majority of the records either the data is NULL or there are blank spaces making harder to work with.
The model can be improved collecting more data, although not always more data means better results.
There other factor to take into consideration for the future. For example checking donor's propensity (3). Therefore improvement will be discussed as they are necessary to improve accuracy of the model and increase revenue in the organization.

VI. Conclusion:

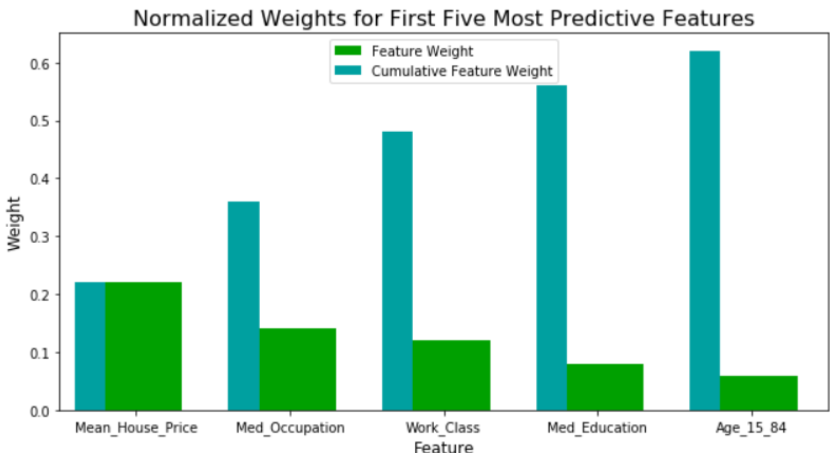
The results of this model does not suffer from high variance and it could be due to the low number of features used but on the other hand the number of points were low. The explanation on the low data points is due to the fact that model is summarizing the outcome based on zip codes rather that at the individual level. The dataset from the organization has 50K plus records which most of them will fall into the model predictions.
The model does not suffer from high bias neither which is a good news.

Overall Scores
The Accuracy and F-Score of the model after optimization is as follows:

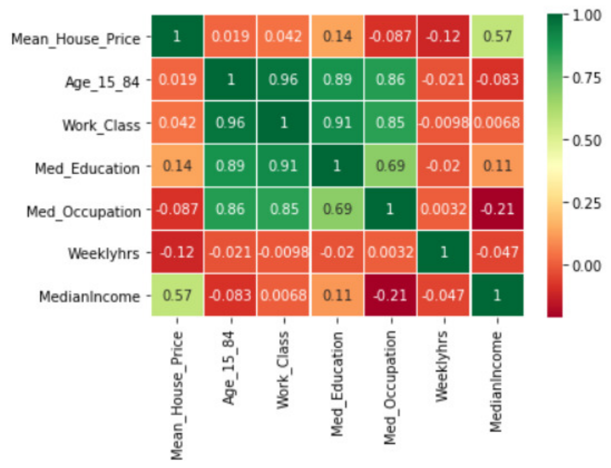
Final Model trained on full data

Accuracy on testing data: 0.8383%
F-score on testing data: 0.8393%

The features of importance used by the model are as follows:



From the above chart the Mean_House_Price is the most predictive feature. That was the original assumption when the model was created and it was a must to have in the model. Even though the data for House Prices was very difficult to obtain it plays a very important role to determine the good model scores. The first five predictive features are in sync with the correlation heat map below:



Reference

1. http://blog.abila.com/artificial-intelligence-machine-learning-nonprofits-associations/
2. https://www.census.gov/geo/reference/zctas.html
3. https://www.iwave.com/2018/07/26/prospect-ratings-the-foundation-of-fundraising-intelligence/
4. https://dssg.uchicago.edu/2014/06/18/why-doing-data-science-with-non-profits-is-different-from-industry/