Courses: Data Analytics and Data Science Python (Joined Assignment)

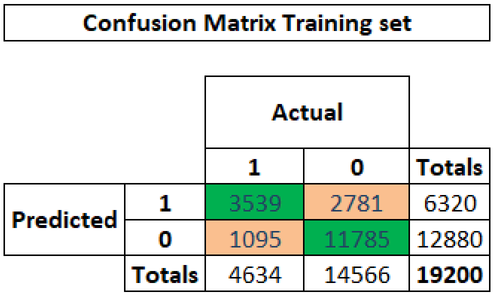
Title: Naïve Classifier System in Excel through the aid of python script to prep data and create training and testing subsets

Team: Team 10 Date: 11th November 2021

An analysis of the given US census data set was conducted with the use of python to prepare data and subset our test and training set. The data was then analyzed with the Naïve Bayes classifier system to predict people with specific characteristics that will earn $50k or more in the US. Our result shows a model with a precision of 51%, which can be further improved by grouping the variables differently, and a good recall and false negative rates of 87% and 6% respectively.

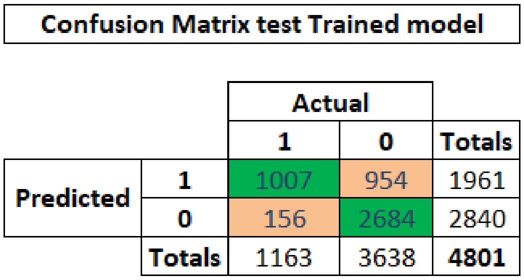
Confusion matrices and results

Our model was well trained in relation to recall and false negative rates after generating our confusion matrixes, which can be seen in the pictures below, where we got 87% and 6%, respectively. However, the precision can be improved since we got a 51%. This means that our model will be good at identifying people that makes more than $50k per year and it will not predict people with less than $50k income when they make more than this.



Confusion Matrix Training set

		Actual		
		1	0	Totals
Predicted	1	3539	2781	6320
	0	1095	11785	12880
	Totals	4634	14566	**19200**



Confusion Matrix test Trained model

		Actual		
		1	0	Totals
Predicted	1	1007	954	1961
	0	156	2684	2840
	Totals	1163	3638	**4801**

The precision shows that our model will classify people with less than $50k income as if they were making less. This occurs because our model adopted the values for the categorical values defined by our knowledge and research. Categorizing them in a different way could have a different result with increased precision, that is why it is important to keep training our model with different categorical grouping.

There are no signs of overfitting because our values and percentages from the test model did not significantly decrease when compared to the values from the “training set confusion matrix”.

Python and Naïve Bayes

Retraining or retesting the data will not yield fundamentally different result, random sampling was used to obtain the initial dataset. If the same dataset is used to produce new random testing and training set, the output will not be significantly different. The Naïve Bayes model does not improve with more interactions if we are basing our set generation on the exact dataset. The model is simple and does not require too many steps to easily accomplish the result in Excel. Python, on the other hand, took a lot of time to get the data ready but it would have been way longer in Excel.

A useful use of this model could be for example, in a bank to check if customers applying for loans will default or no based on past credit records, annual net income, debt to income ratio and past bad credit record. Through this model, the bank can generate profiles of customers (new or old) who may be prone to defaulting, thus reducing decision time, and contributing to business productivity – less money will be lost to bad credits.

Another use of the model can be adopted in FMCG industries to profile the consumer's appetite for new products and predict how well a product will perform in the market.

Project learnings and teamwork

The most difficult part of the project was first figuring out what we needed to do and how to do it because we had to do our own research on how to complete the project especially with the grouping of the categorical variables, it was also a challenge to find the missing value because it was presented in a format which is not part of the default in python to identify missing values. However, the enjoyable part was, after the long lines of coding, exporting the data to Excel and seeing how it translated beautifully. The training and testing of data set is quite new to us but it stretched our capacities, and we learned the basis of machine learning.

As a team, we learned the importance of starting assignments early to avoid becoming overwhelmed last minute and the need for synergy among the team and leveraging individual strengths toward effective completion of tasks. We also realized it is important to get acquainted with pre-reading before team meetings.

We learnt the power of synergy between Python and Excel and that completing most of the task was seamless in Python with the use of formulas. Python is faster in processing large pool of data. We also discovered new functions in Python by calling the help function constantly throughout the assignment. One opportunity area could be to further improve our data analysis and exploring Python to discover new functions that were not touched during the Python class sessions. We can conclude we were very effective on this project, but one thing we would explore for future projects will be to share portion of the project among team members to work on, then collate and refine information and set clear roles for each team member throughout the team meetings.