# 

# 

# **FINAL PROJECT**

**NORTHEASTERN UNIVERSITY**

PROFESSOR : **Dr. Ani Aghababyan**

**ALY6140**

**Fall 2019,**

**Date : December 13th, 2020**

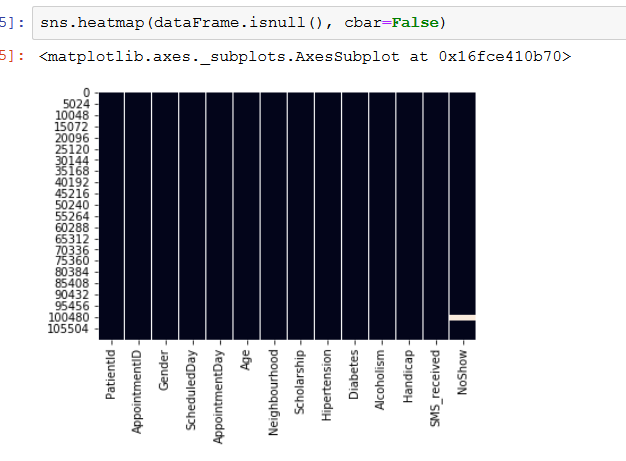
|  |
| --- |
| **By :Priyadarshi Kumar, Sheeba Patel,**  **Ibrahim Salam, Punith Krishne** |

**Introduction**

Healthcare is the most impactful domain for Machine Learning and there is a lot of research is going on to make the healthcare system better with the power of machine learning. There has been done a lot in health care from predicting cancer from a simple image to doing the operation using computer vision and robotic arms. But we are trying to solve a problem which seems very basic problem yet it is costing the US healthcare system 150 billion dollars a year and $200 for each unused slots for a physician and that problem is after scheduling an appointment the patient does not show up for the appointment. Yes, the physician charges a cancellation charge but that is not enough, and it also has an adverse impact on the patient’s well-being. What if we can predict the No-show and optimize the unused slots to make health care better and affordable, this is the question we are trying to answer in this project.

# **Data Cleaning**

The data set which we have chosen consists of 107,419 rows which is further expanded in 14 columns. We have done various data cleaning process to make the data clean.

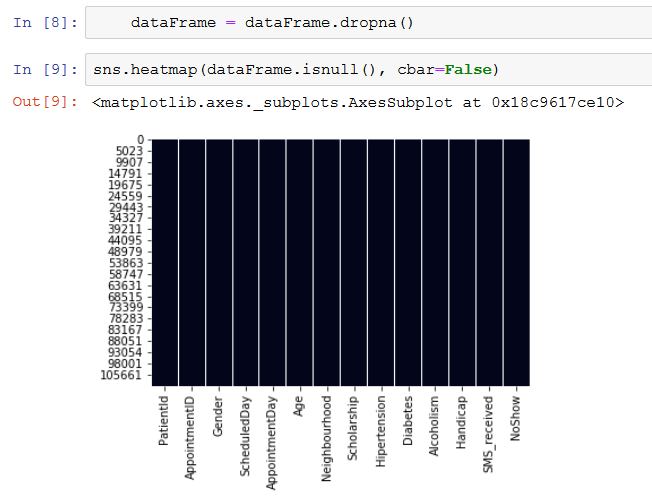


*Figure 1 : Showing the columns having null values*

The very first step is we have checked the NULL values using the above pieces of code and heat map and which shows we have around 3097 rows which consist of some NULL values in them, as it is less than 3 % of the total dataset we choose to drop it instead of imputing the values by using mean or mode. As we can see in *Figure 1* the No-show Column has a white patch which denotes that it has NULL in that region.



*Figure 2 : Total null values in the data frame*



*Figure 3 : No null values after they are dropped*

As we can see in *Figure 3,* we have used dropna() to drop the Null values and again plotting the heat map to confirm whether the null values have been dropped or not.

We also have renamed the column names which has typos in the name using the below piece of code as shown in *Figure 4.*



*Figure 4 : Renaming the columns*

# **Feature Engineering**

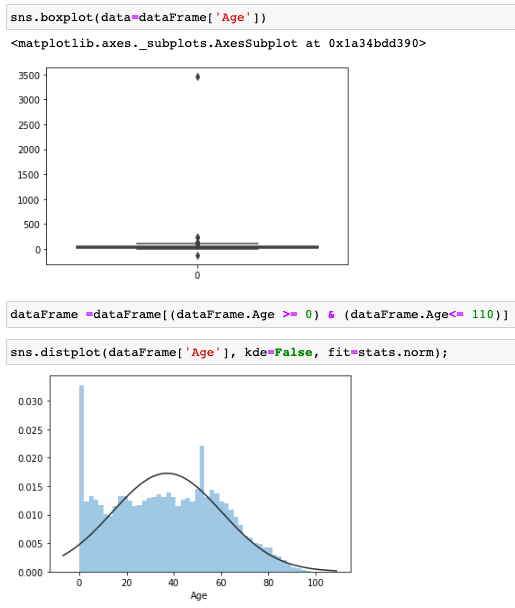
Once the data cleaning part is done we have done some feature engineering. As we have a Date column we have used the date column to extract the Month, Year, Day, Week, Day of The week information from the date column itself. Before extracting the information, we need to change the ScheduledDay and AppointmentDay column to pandas date and time using pd.to\_datetime() function .



*Figure 5 : Date and time split*

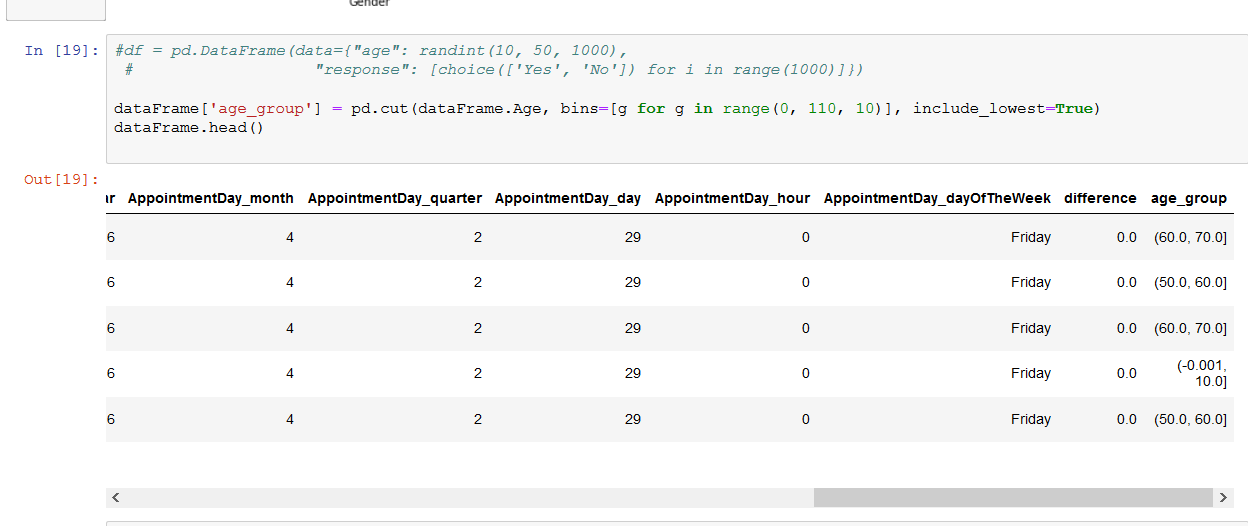
We also have calculated the difference in the actual appointment day and scheduled date by and converted it to days and found out that people has scheduled their appointment 175 days ahead and even after that they did not show up for the appointment. We can see all the things mentioned in the feature engineering in the figure above.

We also have done an outlier analysis for the Age by using boxplot and found out that age has value which is 3500 year which is definitely a typo error and it was affecting the distribution of age. After capping the age value between 0 and 110 we plotted the distribution the age does not fit the normal distribution as we can see it from Fig 6.



*Figure 6 : Showing outliers and distribution for the Age column*

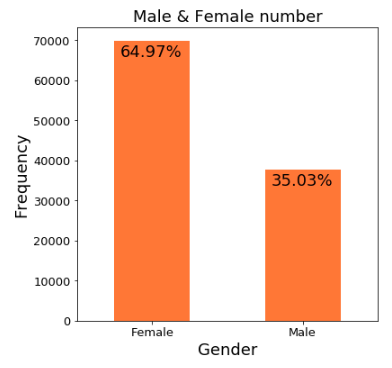
We also have created an age bin of bin size 10 and divided the age from 0 to 110 because we want to see which age range has significant number of No-shows, and the code can be seen below in *Figure 7*.



*Figure 7 : Indicating the different age group*

# **EDA**

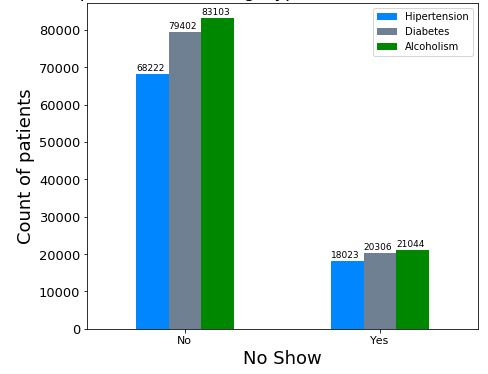
After the Feature engineering we have done some exploratory data analysis . As we have more categorical values we have done bar graphs. Because we believe that bar graphs are the best way to describe categorical values.



*Figure 8 : Showing the gender distribution*

The first analysis which we have done is about the Male and Female percentage in our gender column as we can see it in *Figure 8*, and we found that Female percentage is more than male percentage and it means the dataset is somehow not balanced.

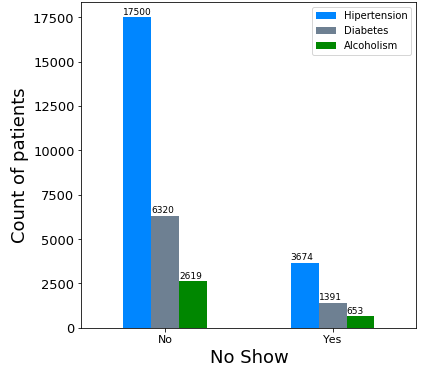
Moving further, considering the different conditions the patients are suffering, like for instance : Hypertension, Diabetes and Alcoholism. Let us see the number of patients who show or not show up if they are not suffering from the above mentioned ailments in the figure below,



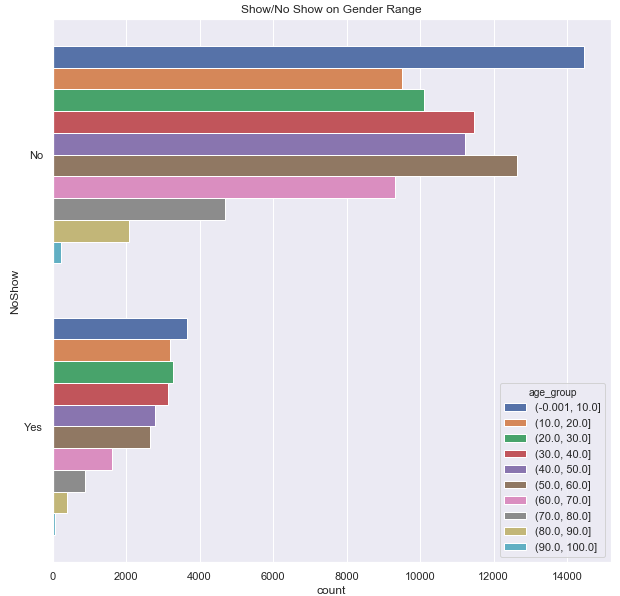
*Figure 9 : No - show of patients not having ailments*

From the above plot, we can observe that the patients not turning up is more in number even though they are not suffering from these ailments. We need to understand why they are not turning up, is there any other reason? We need to do further analysis.

Similarly, let us look at the plot which shows the count of the patients who are suffering from Hypertension, Diabetes, etc. and not turning up to the appointment. Below is the plot indicating the number,



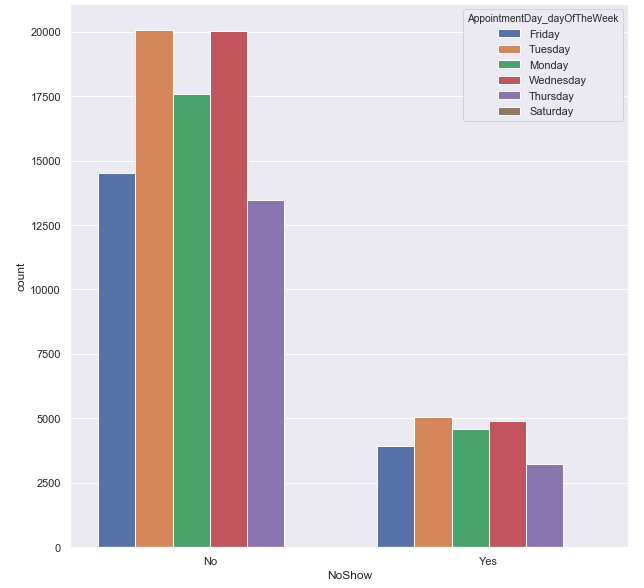
*Figure 10 : No - show of patients having ailments*



*Figure 11 : No - show/Show of patients of different age groups*

As we have discussed previously about the bins we have used the bins to plot the above graph and, it is quite evident that, the range of people between 0 - 10 are most likely not to show up for their appointment, may be because they are dependent upon someone to bring them to the appointment.

As we have generated the day of the week in the Feature Engineering part, we have plotted the days of the week with our target variable “No - Show” to see which days are affecting the no shows. And it can be clearly understandable from the Fig 8that people does not show up for their appointments on Tuesday and Wednesday, and as it is mid-week, that can be a huge reason.



*Figure 12 : No - show/Show of patients of different days of the week*

**Modeling**

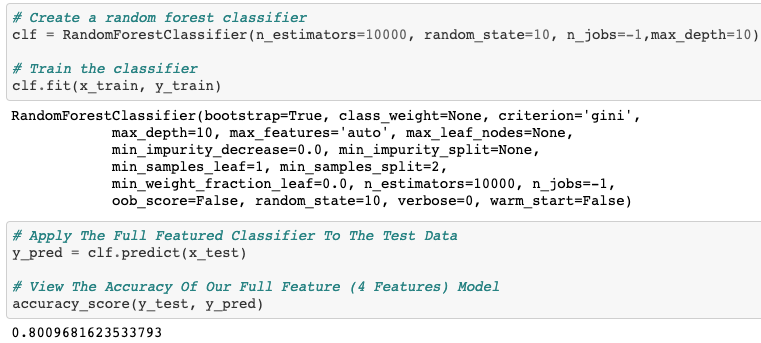
**Data Preprocessing :**

In this step let us prepare the data for the prediction and modeling. This is such an important step which aids in removing unwanted features which don’t have contribution to the prediction of the target variable “No - Show”.

Initially, we start with renaming some of the variables which are very lengthy. Altering names will help us in easy identification. Then let us convert some of the variable into ‘0’s and ‘1’s. The Gender and the No - Show columns are converted using the replace function or the lambda function learnt in the class lectures. As I have already mentioned earlier, let us drop some of the variable like the Patient Id and the Appointment Id which have no importance in the prediction process.

## **Random Forest :**

To start, it was essential to import RandomForestClassifier from sklearn.ensemble. In order to gauge the effectiveness of our predictions, a Random Forest was done on the data.

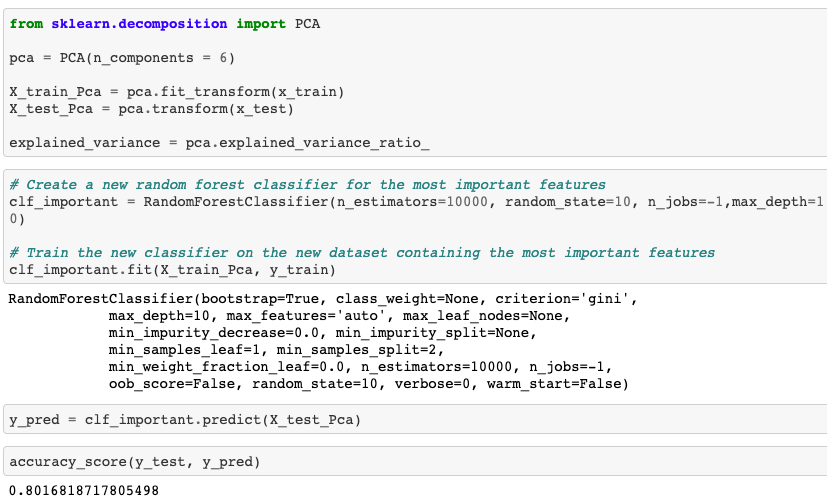


*Figure 13 : Implementation of Random Forest*

To start off, RandomForestClassifier was called with n\_estimator equating to 1000. N\_estimator entails the number of trees in the forest. Next, random\_state was set to 10, random\_state is involved with controlling the random choices. Next, n\_jobs was set to -1, it was set to -1 since n\_jobs controls the number of jobs that are running in parallel therefore -1 allows the use of all processors. Lastly, max\_depth was set to 10, max\_depth entails the maximum depth of the tree, this is important since if max\_depth is not utilized then the tree can be significantly large. Finally, we trained the model and applied the full feature to the test data, and consequently, received an accuracy of 80.00%. Above is a depiction of the code and the subsequent accuracy result.

## **PCA Analysis :**

To start, it was essential to import PCA from sklearn.decomposition. The purpose of running PCA analysis was to gauge the association of each variable with one another and find the direction that the data is being dispersed. N\_component was set to 6 since otherwise if it was not set all components would be accounted for. After training the model and establishing the test set. The predictions were consequently made on the test data and when computing the accuracy score the subsequent results were 80.16%. This leads to an increase of 0.16% which is not a considerable value. Below is a depiction of the code and the subsequent accuracy.



*Figure 14 : Implementation of PCA technique*

**Conclusion**

In conclusion, although there is a disconnect between the patients and their physician in terms of communication about their scheduled booking, predictive analytics is slowly filling in those patches which allow the health professionals to understand why there are patients not coming to appointments and how they can mitigate such no-shows. The models presented help us measure how each feature is associated with each other. RandomForest and PCA analysis were very similar when juxtaposed next to one another, however, a deeper analysis can be done to gauge the sensitivity and specificity which was beyond the scope of the project and this will allow the data scientist decide which model works better since it is specific to the situation. Overall, from cleaning the data to exploratory data analysis and lastly creating the models helped simulate a real-world example since in the workforce, raw data will never be cleaned for analyzing as half of the effort goes towards cleaning and preparing the data.

**References**

Will Koehrsen (Dec 27, 2017), *Random Forest in Python*, Retrieved from : <https://towardsdatascience.com/random-forest-in-python-24d0893d51c0>

Saimadhu Polamuri (June 26, 2017), *Building Random Forest Classifier With Python Scikit Learn*, Retrieved from : <https://dataaspirant.com/2017/06/26/random-forest-classifier-python-scikit-learn/>

Michael Galarnyk (Dec 4, 2017), PCA using Python (scikit-learn), Retrieved from : <https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60>

Felipe (22 Dec 2017), Pandas Data frame: Plot Examples with Matplotlib and Pyplot, Retrieved from : <http://queirozf.com/entries/pandas-dataframe-plot-examples-with-matplotlib-pyplot>