Optimizing Healthcare Business operations:

Abstract:

Every hospital on an average spends $150 on every appointment of the patient including the time and the effort spent on making arrangements for it. Patients not showing up for their appointments will create a loss for every hospital that cannot be compensated. In the worst case, if a hospital works only 8 hours a day and 5 days a week, let's consider that it takes 16 appointments per day. It accounts for 80 appointments per week. In an article published by 'Health Management Technology' website, there are 30% missed appointments per week, which to our numbers account for 24. Let's decrease that number to only 20. Considering the loss of 50$ per appointment, we have a loss of $1000 per week and $50,000 per year. But almost no hospital works only 8 hours a day and 5 days a week. Imagine this for a large hospital or a healthcare organization that handles around a hundred doctors and manages at least 1000 appointments per day. As the size of the organization increases, the No-show cases keep increasing compelling the organization to take action on this situation to control the losses.

This project provides a solution for a healthcare organization that handles numerous appointments and are plagued by the loss of No-shows. It uses the historical data that the doctors and the organizations have kept the record of over all these years and predict the possibility of No-show for an appointment as soon as it is made. As this solution requires the organization to adapt to certain technologies, it not only enables them to deploy this solution but also helps them use this architecture to understand the business to make smart decisions. Those architectural changes and several technologies that are involved, including the process of channeling the data into the system and into the application are discussed in this document.

Data Sourcing:

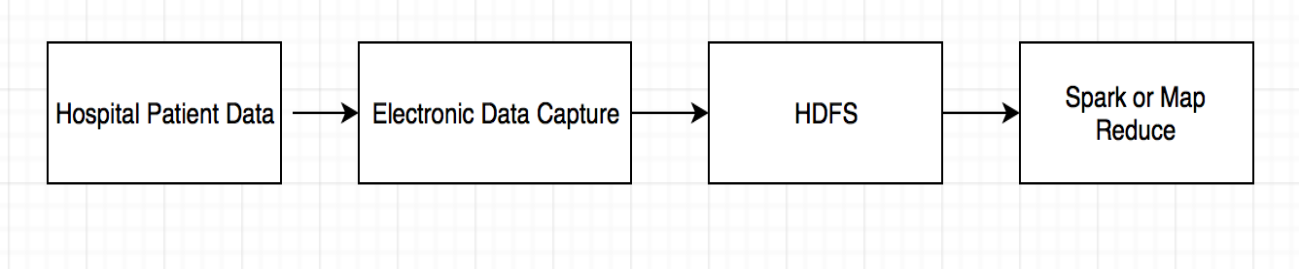
The huge advantage of machine learning is the fact that the whole business systems had moved towards computer-based or Electronic data almost two decades ago. This gives the base for using machine learning as all the historical data that has been collected till now will be used to predict

the events that will take place in the future. Big hospitals and healthcare organizations have been collecting the patient data which is the reason that opportunities for using data science solutions in healthcare are flourishing in the current market and research works. This is a valuable asset for the humankind as it not only enables us to improve the business which in term aids the doctors to better diagnose the patients.

As the data is too large, using Big data architecture for this project will not only enable this project but also provides opportunities for several other big data solutions. The data that has been collected from these hospitals will be directed into the big data architecture that we establish. The details on the infrastructure that are proposed to use for this data will be discussed further. In general, if the solution is being made for one particular organization, the historical patient data of that organization can be used. The data that hospitals have been collecting till now can be transferred to the big data platform, but the data that will be collected from now should be collected to cloud, as all the technologies are moving away from classic hard-disk storages. Several efficient Electronic Data Capture (EDC) tools are being offered to manage healthcare data, that store the data directly into the cloud. Tools like Medidata Rave, Oracle Clinical are already being used extensively by several healthcare and pharmaceutical businesses for data collection.

For this project, we have a patient appointment data from Brazil. The data contains 110,500 rows, that contains variable information about patients' appointments along with several patient details. It contains patient-ID (Integer), Appointment-ID (Integer), Gender (Character), Scheduled Date – Date+Time (String), Appointment Date – Date+Time (String), Scholarship – 0 or 1 (Integer), Hypertension – 0 or 1 (Integer), Diabetes – 0 or 1 (Integer), Alcohol – 0 or 1 (Integer), Handicap – 0 or 1 (Integer), SMS_received – 0 or 1 (Integer), No-show – 0 or 1 (Integer). The scholarship variable is specific to Brazil, which refers to a medical aid to the citizens who satisfy the requirements of the government, that is education and vaccinations. The concept of scholarship in Brazil is a detailed one and discussing it deviates the purpose of this paper. However, we will use this variable in our analysis as it provides possible insights and helps increase the efficiency of our machine learning algorithm.

As far as the future data collection go, Electronic Data Capture tools that are specifically developed for medical purposes are readily available in the market to equip. These tools directly push the data into the cloud or the big data infrastructure, enabling a seamless data collection into the system. Two of the most used EDC tools are Medidata Rave and Oracle Clinical.



Data Cleaning:

The data that we are using for this project needs to be cleaned. I will be going ahead in this project with R studio for the data cleaning and for algorithm implementation. First, I decided to delete the patient ID column, as there are only 37,920 unique patients available over the 110,000 unique appointments. This creates duplication of patient ID. Also, as we are not performing a historical analysis for each patient individually, we don't need to include patient ID.

Then I converted the two categorical variables, gender, and No_show, that have strings and characters like, M or F for gender and Yes or No for No_Show. Then the Appointment date and Scheduled date, both fields have string values that include both date and time with a delimiter 'T'. As I intend to use the appointment date and schedule date in the visualization and the algorithm, I used substring to separate the date and time and create a new column for the time field.

Now the age field has values from -1 to 115, in which the outliers should be removed. For this project, I considered anything below 0 and above 99 as outliers. It is possible to live more than 100 years but as there are very few values, let's eliminate them. Regarding the age field with value 0, there might be infants below age 1 year, and these data might be of those infants. It is

crucial data and cannot be eliminated. Also, as there are more than 3000 values, these cannot be considered as outliers too.

Finally, I renamed some columns as some of them are misspelled. With this, I finish the data cleaning and now I have the data ready for visualization and algorithm implementation.

Setting up the Big Data Environment:

Big Data Architecture – Why and how:

Health-care organizations and hospitals should use big data infrastructure to perform several business and domain operations. They need Big data infrastructure as they have the four 'V's of big data in their data, that are volume, velocity, variety, and veracity. As huge operational organizations, hospitals host a lot of information like the patient data, the Doctors' or employees' data, the insurance data, billing or transaction data and pharmaceutical data, which says it has volume. The data that these hospitals are storing is heterogeneous, meaning it will contain images, text, videos and several other varieties of information from patient's records like X-rays and scan reports. This makes up for variety. To perform the real-time computation for analytics and to apply solutions to the problems like Medical Appointment keep up prediction, they need velocity. As there is information being added from several sources, they will have abnormalities and biases in the data, which is veracity.

As a business adapts to big data, it's not just building the ability to store and retrieve the data. Big data architecture comes with several other functionalities that the business needs to use to leverage the advantages of using this technology. Some of them are purely optional but it is the option of better performance every business wants.

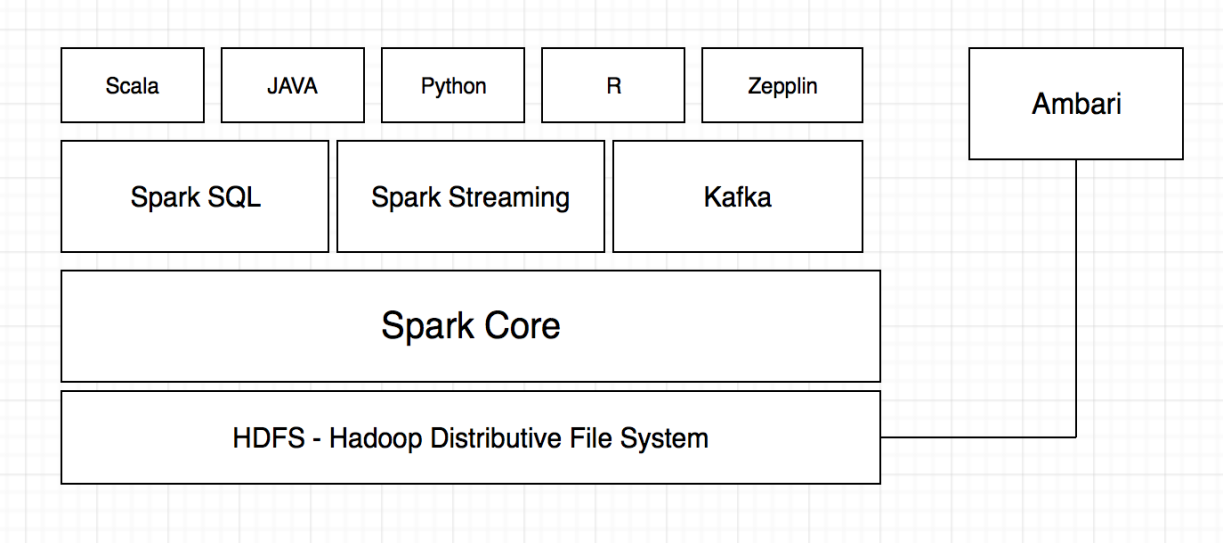
As the healthcare industry uses real-time computing for analytics they need computational speed. Computational speed is not just achieved by vertical scaling, i.e. incorporating powerful processors and using a lot of them to compute. In-memory computation delivers the fastest results for any heavy-duty computations. Having a cluster that can host a lot of data on Random

Access Memory (RAM) and perform computations on them is a lot efficient than vertical scaling. Spark comes into play here. Spark is one of the components of big data and used for in-memory computations. While adapting to big data infrastructure, the business should create a cluster of computers that can host and perform data.

During this setup, the business might decide to equip some of its clusters with a large RAM and optimum processor. These clusters are maintained by the spark, where the computational data will be divided into pieces on to these clusters, hold them in memory and perform necessary computations. The spark system takes care of how to distribute the memory and how many virtual CPUs to be allocated to the process. This entire in-memory system will enable real-time computation for analytical solutions like the one we are trying to deploy.

Once the environment is set-up, spark shell can be used for basic Extraction, Transformation, and Loading of the data. If actions are to be taken beyond the functionality of the spark shell commands, a Jar file can be created with Scala code and can be deployed over the entire HDFS to gain the desired results. Data from HDFS can be aggregated, transformed and distilled via spark and can be copied into a local file system from which we can download it as a CSV or ORC. This format of data is most compatible with the Hadoop Distributive File System (HDFS).

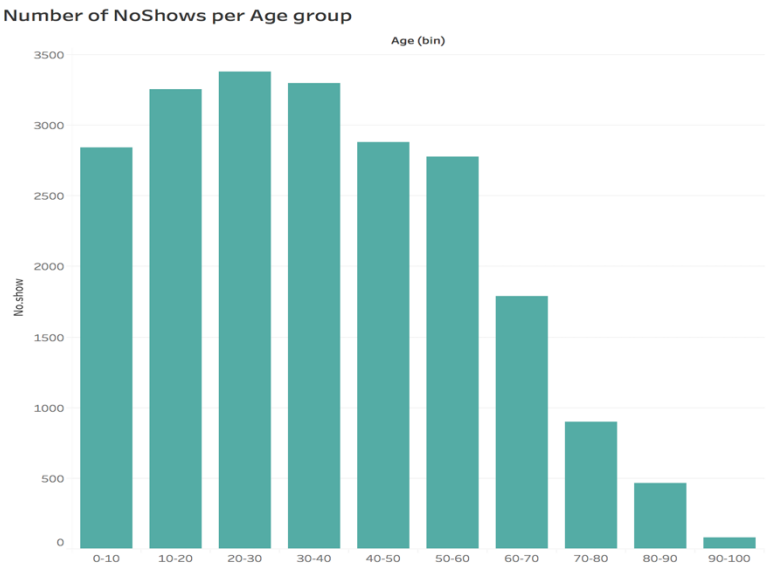
The advantage of using spark is that along with Scala, it is compatible with other data science languages like python and R as PySpark and SparkR. This enables data engineers to work with Scala and data scientists to work with R on the same platform. Visualization can be performed using tools like Shiny directly from R or a subset of data can be extracted to visualize in Tableau or D3 so that they can be better displayed to business heads during meetings. Ambari is a hardware manager for Hadoop, which displays Hadoop usage information like RAM, Processing power and other graphs displaying the performance of the entire system over time. Zepplin notebooks can be used for running small commands over the system in Scala, Python or R.

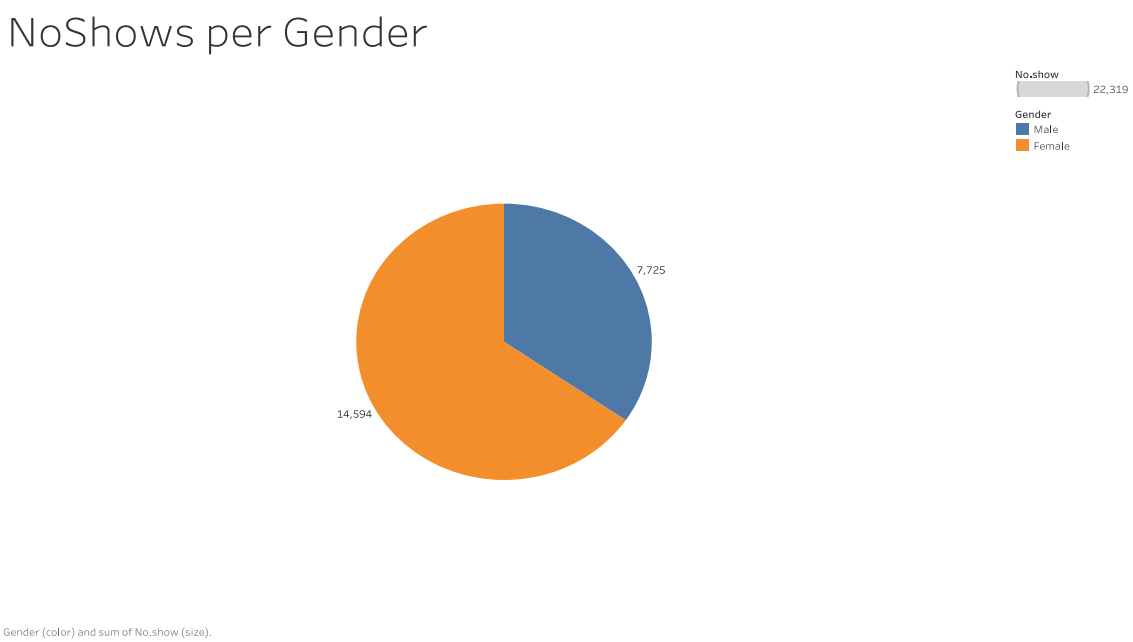


In the discussed process, the data will be stored and managed in Big data architecture. This will provide feasibility for our project to be deployed for the Healthcare systems.

Data visualization – Feature Selection:

Data visualization is one of the most important element in Data Science, which gives us an overview of the data allowing us to proceed further and evaluate the data cleaning that we carried out until now.



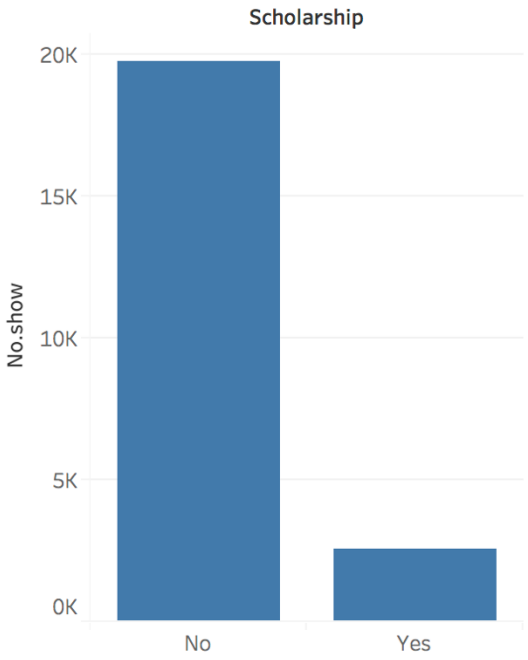


I used Tableau to create visualizations for this project. After cleaning the data, I imported it as a CSV and uploaded the CSV into the tableau. Tableau provides the measures and dimensions of the data automatically once it is uploaded. However, once I uploaded the data, tableau did not provide the desired dimensions and measures that I need, so I included both columns and rows from the measures and converted on as a dimension. I also changed it to discrete so that the value can be better portrayed. I chose the NoShow field that had 'Yes' or 'No' string field previously was changed to 1's and 0's while data cleaning. I used these 1's to calculate the count of NoShows per Age and Gender. The below charts represent the data that I'm working with, in the Tableau. I binned the Age field to 10 values per bin so that the visualization can be understood easily. I used a pie chart for the Gender to NoShow chart as the Gender has only two values 'Male' and 'Female'.

These visualizations already tell us some facts. The Gender to NoShow charts tells us that female patients have missed their appointments more than male patients. The Age to NoShow chart shows the patterns in which people miss appointments based on the age. Although there might be some discrepancies as the data might be generalized, for example, there might be more females in that particular data than more males. But the data contains a hundred thousand records, which gives us a certain amount of confidence on what the data is trying to tell us.

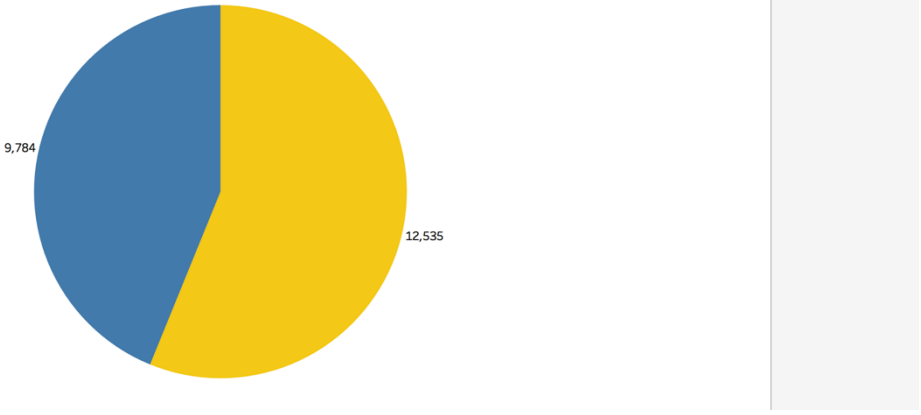
Another features that I have included in this solution is the 'scholarship' feature. As discussed, It is a location specific variable that is included in this dataset. The data collected in this dataset is from Brazil, where the government provides some allowance to people when they meet certain requirements. Although this is a very specific feature for the location, it is an important factor to consider as it can be taken as a form of medical insurance. I created charts on scholarship with respect to the count of missed appointments, it appears that people with scholarship tend not to miss their appointments. The visualization of the reference is on the right, which shows that people who don't have scholarship tend to miss their appointment more.

Scholarship



There is also an attribute that tells if the patient received an SMS reminder regarding the appointment. This too is an important attribute to consider as the reminder will help the patient not to miss the appointment.

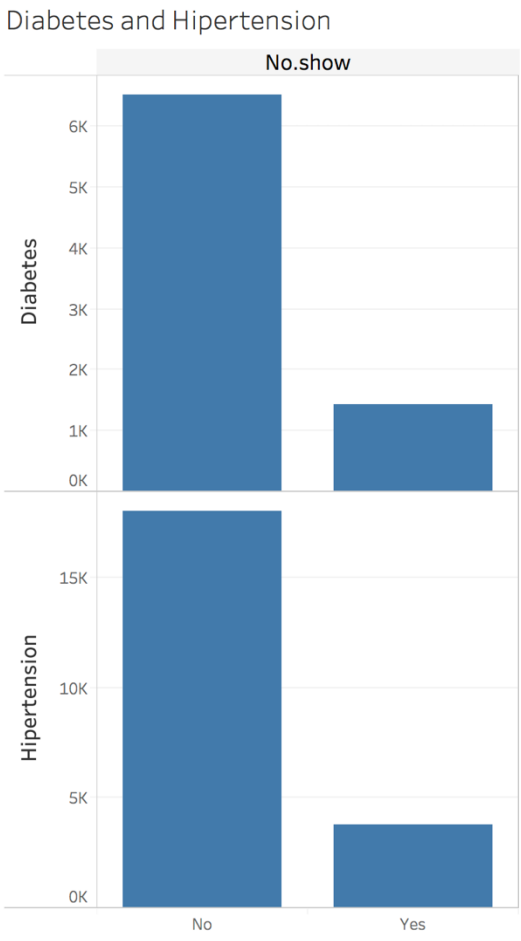
NoShow count against SMS notification



I created a visualization for the counts of the missed appointments with respect to the number of people who received an SMS reminder. In the total people who missed their appointments, people who did not receive their appointment are more considered to those who did. The visualization of this metric is pasted below.

While I started working on the machine learning in the later part of the project, I discovered a hidden feature that will be very significant if included in the model. The data includes schedule date and appointment date which are in string format. These variables even if converted into date format cannot be used as features in the model. So I decided to obtain the difference between these two dates which will give us the number of days the patient had to wait to attend his appointment. I created a field called DaysWaited, which is an Integer column and can be included into the model as a feature.

One last set of useful features that I wanted to include is the set of problems that he or she is undergoing. In this case, our data has two attributes 'Hypertension' and 'Diabetes'. Using the multi-axis in the tableau, I created a graph that represents the counts of both hypertension and diabetes in a single graph on patients who missed their appointments. On the right is the graph that represents this.



These visualizations gave me enough information on what features to be included in the model. As the data we have in this case is scalable to a human brain, we can use tableau or any other BI tool like power BI or even Excel to create these graphs to get an overview of the data that we are working with. But in the case

of larger datasets, we need to aggregate and group the data to make the dataset smaller. We try to limit the data as much as possible using spark shell, in our case of big data architecture. That will be considered as data cleaning, after which we use BI tools over the smaller datasets.

By the end, I have the features as,

1. Gender
2. DayWaited
3. Age
4. Scholarship
5. Hipertension
6. Diabetes
7. Alcoholism
8. Handcap
9. SMS_received

Machine Learning algorithm:

When we create a machine learning model, we train the model on the data, i.e. we try to fit the model on the data and we use this model to make predictions on data that has not been trained. Partitioning the data into test and train data is important because if the entire data is used for training, then there will be a possible overfit. To avoid this, we divide the given dataset into two splits and use one for training and one for testing. The test data will ensure that the model is not overfitted onto the train data.

We can split the data based on the specific need of the application, but the ideal split is having train data between 60% to 80% and use the remaining data for the test. In R, we can either find out the number of rows for the desired split percentage in the entire data and assign a value to it. We can use this row count to set seed to get a random row number and split the data to create two datasets. In another method, we need an R package called caTools. In this scenario, I decided to use 75% of data as train data and rest 25% as the test, in the beginning, depending on the efficiency results, I will re-shuffle and again split the data.

Clearly, this problem can be solved by using Logistic regression. Although I started working on the logistic regression with this data, I was interested to perform Neural networks on this data to

see how the efficiency of the model varies every time. The main reason I decided to try neural networks is that the dataset is considerably large, and the features are well labeled.

I started by implementing the logistic regression, by creating a model and training the model with the features that we have decided on. Logistic regression is generally used to answer Yes or No questions. In our solution, when we train the model on the train data and predict the values on which will be set to output either 1 or 0.

The ideal cut-off is 0.5, as there are only two possibilities, 0 and 1. But as our data has less number of missed appointments when compared to attended ones, our prediction score is mostly around 0.4 and 0.5. If we use cutoff 0.5, the model performs poorly, so we used a cut-off of 0.4. This means that if the model predicts a probability of 40%, it will be considered as a missed appointment. Moving further, to get the statistics on getting the accuracy of the model, I imported 'caret' library. I also created a confusion matrix for the results.

```
          Reference
Prediction     0     1
         0 25737  6371
         1   725   326

               Accuracy : 0.786
```

Now to continue with the process, I created a neural networks model by importing 'nnet'. In the same process of creating the logistic regression model, I created a neural network model and used the train function to train the data which will autonomously decide the number of neurons and iterations. After this, I predicted the values for test data and created a confusion matrix just as I did for the logistic regression.

```
                Reference
Prediction      0     1
         0  25729  6168
         1    733   529

               Accuracy : 0.7919
```

Now I created a table that displays the result for both the models in a table. These results not only include the accuracy, but also the precision, which represents the percentage of true positives. This is a very important measure for using the model in practical real-life situations. Both the models are producing good true positive percentage, but they are only producing true negatives moderately. This might be the result of having data biased towards the number of attended appointments than the missed ones.

Still, this should not be a problem, because considering this problem is used in healthcare, having a false negative is better than having false positive which will result in incontinence to the patients. The table with the results is posted below. I obtained an accuracy of around 78% for the logistic regression and 79% for the neural networks model.

	ANN Model <dbl>	Logistic Model <dbl>
accuracy	0.7918815	0.7860008
precision	0.8066276	0.8015759
recall	0.9722999	0.9726022
F1	0.8817492	0.8788458

4 rows

Demo:

Sample data

Created Sample dataframes to test the model

Hide

```
x <- data.frame("Gender" = 0,"DayWaited"= 10,"Age" = 25,"Scholarship" = 0,"Hipertension" = 0,
                "Diabetes" = 0,"Alcoholism"= 1,"Handcap" = 0,"SMS_received" = 1)
nn.prediction<-predict(nn.p,x)
nn.prediction
```

```
      [,1]
1 0.3649398
```

Hide

```
nn.prediction[nn.prediction<0.4] <-"Show"
nn.prediction[nn.prediction>=0.4 && nn.prediction!="Show"] <-"NoShow"
nn.prediction
```

```
  [,1]
1 "Show"
```

Hide

```
y <- data.frame("Gender" = 1,"DayWaited"= 20,"Age" = 22,"Scholarship" = 0,"Hipertension" = 0,
                "Diabetes" = 0,"Alcoholism"= 1,"Handcap" = 0,"SMS_received" = 0)
nn.prediction<-predict(nn.p,y)
nn.prediction
```

```
      [,1]
1 0.4143105
```

Hide

```
nn.prediction[nn.prediction<0.4] <-"Show"
nn.prediction[nn.prediction>=0.4 && nn.prediction!="Show"] <-"NoShow"
nn.prediction
```

```
  [,1]
1 "NoShow"
```

To provide a sample working prototype of the solution, I created two simple data frames to which I added values of the features. Then I predicted the values of these data frames using the neural networks model. I adjusted the model to print 'Show' or 'NoShow' based on the prediction score.

Business justification:

Healthcare is a field that can impact the world at a huge level but being very resistant towards technological advancements. Particularly it's a field that has huge amounts of data that can be used not only to optimize business but also to better diagnose diseases and finding a cure. Radiology is already moving towards implementing these techniques to cure cancer. Healthcare

has the data and potential, if tapped using data science, can revolutionize the healthcare field. Recalling our numbers again, if the model has an accuracy of around 80%, out of the 20 missed appointments, let's say it predicts and forecasts 16. This will save us around 40,000 dollars on a scale of one year out of 50,000. For a larger organization, the impact will be even higher as the loss rate will be more. This can result in a lot of savings if employed and the architecture that we implemented will help us conduct several medical and business researches to help make the world a better place.

References:

https://www.healthmgttech.com/missed-appointments-cost-u.s.healthcare-system-150b-year