Improving Patient Experiences by Analyzing Net Promoter Score

Natalie Bondurant
921072832
STA 160: Practice in Statistical Data Science

Abstract

Hospitals want to provide their patients with the best service they can both so that they can improve patient wellbeing during stressful situations and so that their business may thrive as a result of their patients sharing their positive experiences with others to attract more business. In order to do so, they must analyze and understand the manner in which patients view the quality of their experiences, then implement changes in their service accordingly. Since patient satisfaction is a subjective, self-reported experience, it is important to utilize survey methods to gather data about how they felt about their experience. Previous analyses have shown that some ways to improve patient experience include minimizing wait times, ensuring cleanliness, engaged staff, and improved communication. This study aims to delve further into which variables have the most impact on patient satisfaction using Net Promoter Score as a metric for gauging how satisfied they were with their experience. I found that in addition to the aforementioned methods of improving experiences, the cost of the visit, quality of the explanation of treatment plans and packages, number of days spent in the hospital, and quality of both patient and attendee food also contribute to patient satisfaction.

Introduction

Hospitals are businesses that want to attract as many customers as possible by providing the best service they can, and one of the ways in which they can assess how people feel about their experiences at hospitals is via Net Promoter Score, or NPS. NPS is a survey that asks people to rate the likelihood that they would recommend a particular service or experience to other people on a scale of 1 to 10, with 10 being the most likely to recommend. It is then divided into categorical variables where 0-6 represents detractors, 7-8 represents passives, and 9-10 represents promoters. Hospitals can then use NPS to improve patient experience by examining how patients' perceptions of various services within hospitals contribute to their overall likelihood of recommending them. In this way, they can focus more attention and resources on areas that both have a lower rating and larger impact on overall NPS to most efficiently and effectively improve patient satisfaction.

My approach to analyzing this dataset was to first explore the data for potentially important relationships, then examine which variables are the most influential to NPS. My approach to modeling and predicting, because NPS is split into three categories, was focused on classification methods.

Data overview and exploratory data analysis

The dataset I used is from the "Predicting NPS to Improve Patient Experience" competition on Kaggle. It pertains to patient experiences at Manipal Hospitals, the oldest group of medical centers in India, known for being ethical and amenable with patients. Believing word of mouth to be the most effective type of promotion, the CEO of Manipal Hospitals implemented an efficient and detailed patient feedback system that included NPS as well as 50 other variables pertaining to patient satisfaction; the dataset consists of the results of this feedback system from 4,989 patients each represented by a row in the dataframe with no missing values.

First, to get an idea of how variables were related to each other and which variables were clustered together more tightly than others, I graphed a network model of the data. The graph showed that variables pertaining to the same subcategory were grouped closer together than others; for example, variables starting with EM_ represent concepts pertaining to patient experiences in emergency care. Such variables also tend to have thicker edges between each other which represent stronger partial correlations. It is also evident from the graph that the vast majority of variables are positively correlated with each other, as represented by the blue edges that make up most of the edges.

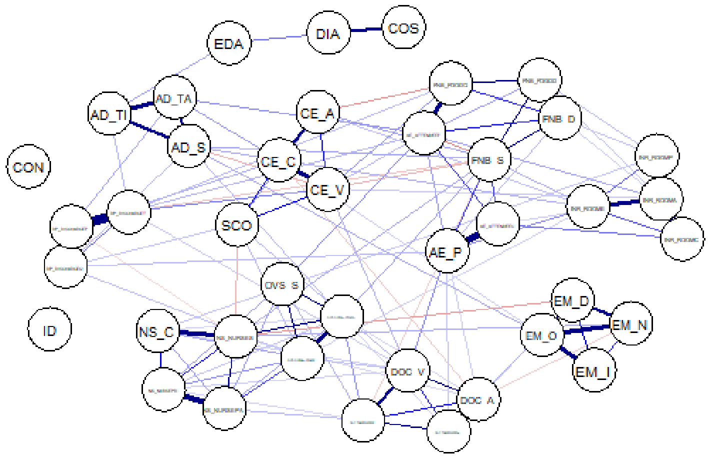


Fig 1. Network graph of relationships between variables

The two response variables both measure NPS, with one being the raw score from 1-10 and the other being the categories of promoter (63% of patients), passive (27% of patients), and detractors (10% of patients).

There are 41 numerical variables such as estimated cost of visit, age, and rating of food quality, doctors' attitude, overall staff promptness, and more, where variables denoting patients' rating of various aspects of their experience were on a scale of 1 to 4, with 4 representing the best experience. Histograms of all numerical variables except the number of days in hospital, cost of visit, and age showed left skewed distributions. Cost of visit was right skewed and had over 150 outliers on the high side. Many of the variables on a 1 to 4 scale had a lot of outliers on the low side since patients tended to rate highly on these scales.

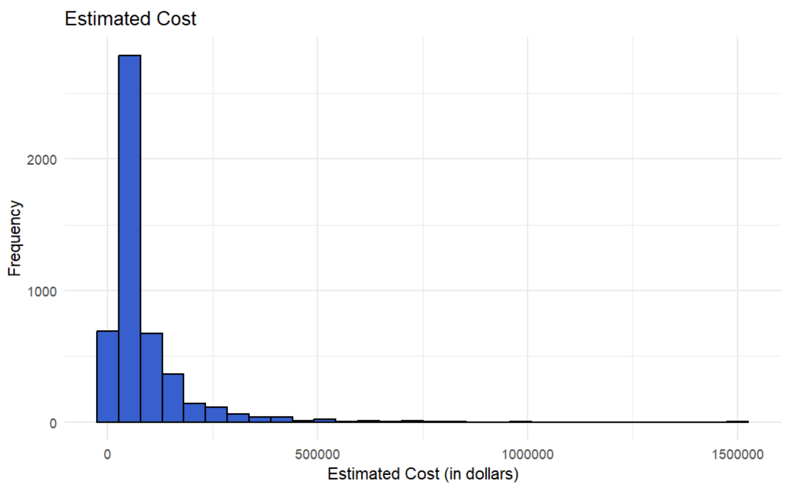


Fig 2. Histogram of estimated cost of visit

A plot of the correlation matrix of the numerical data shows that all of the numeric variables are either positively correlated or uncorrelated with each other, as shown in Fig . below where blue represents positive correlation, white represents no correlation, and darker colors represent stronger correlations.

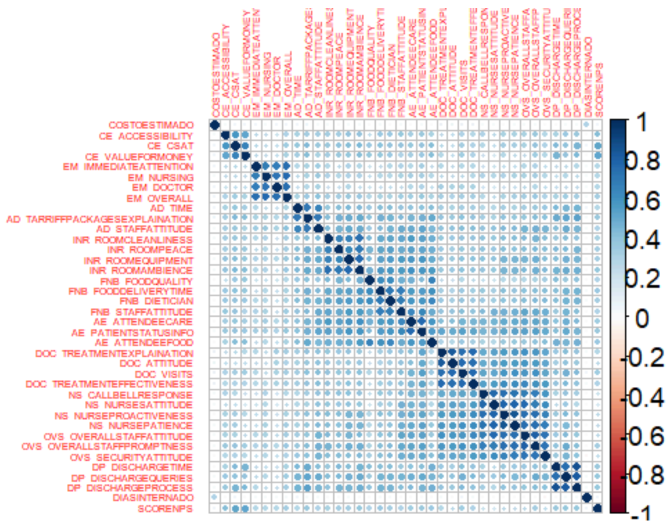


Fig 3. Correlation matrix plot of numeric variables

There are 9 categorical variables such as gender, marital status, type of hospitalization, and department. In many of these variables, one or more of the categories were more represented than the others; for example, the proportion of types of hospitalization that are either general or semispecial is 0.68 despite there being 9 other categories. Additionally, the vast majority of patients were from India, and the vast majority of hospitals were located in South India. As a consequence of this unbalanced data, any predictive models may have more trouble fitting predictions to levels that have considerably fewer observations within them.

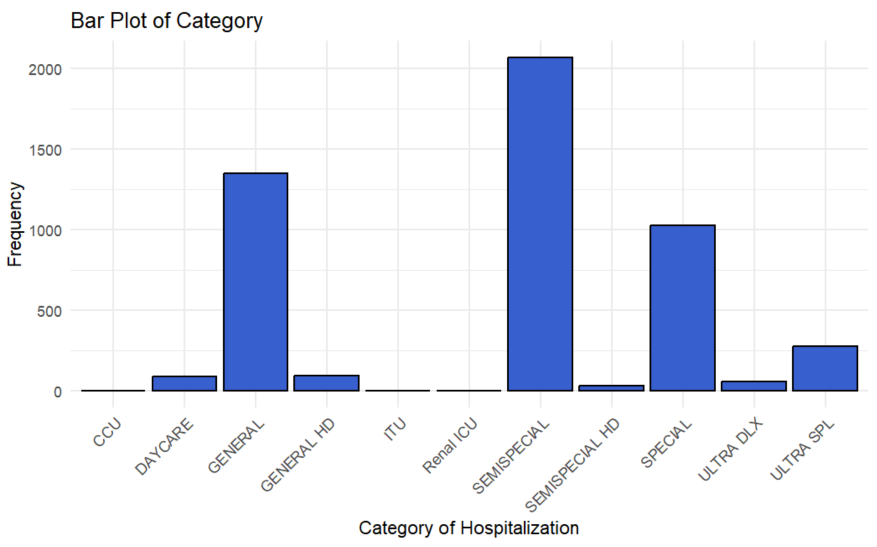


Fig. 4 Bar chart depicting the category of hospitalization

Methodology

Preparing Data for Analysis

One of the goals of this analysis is to see which variables have the most impact on NPS in order to most efficiently and effectively improve patient experiences. The other goal is to be able to most accurately predict NPS using the other variables in the dataset. As such, NPS is the response variable, and the other 50 variables are predictors.

Additionally, k-fold cross validation was used to test whether the accuracy of the models was robust to changes in other test data sets. In each case, 10 folds were used to validate the accuracy because it helps minimize the bias of the model by evaluating the model's performance 10 times rather than only 1 which improves the robustness of the model. I included the multipleStatistic parameter to get more information about the performance of each model.

Data was split from the original dataset containing all 4,989 observations into two datasets, with one containing 70% of the data (or 3,492 observations) acting as the training set, and the other contianing 30% of the data (or 1,497 observations) acting as the testing set. This way, models could be trained on the training set, then evaluated based on their performance at predicting NPS in the test set. To do this, I randomly sampled 70% of the data into the training set, and the remaining observations became the test set.

Another possible concern was that the proportion of the response variable was unbalanced between the three levels which could lead models trained on that data to learn that they can result in high accuracy by simply predicting the majority class, Promoters in this case. To balance the data, I oversampled from the Passive and Deterrent classes using the upsample function to weight minority classes more strongly. This way, each level of NPS had the same number of observations as the Promoter level originally had. I also ran each test with both the unbalanced and balanced data to see if they would be more accurate when the levels of NPS were balanced.

I used one-hot encoding in order to be able to include information from the categorical variables into models that do not accommodate categorical data. Many predictive models are not meant to take in categorical data, so my way around this was to one-hot encode all of the categorical variables so that each level was turned into a new column of binary data. I did this by turning each of the 9 categorical variables into matrices where each column was a level of the variable and each row was either a 1 or 0 depending on if that level was present or not, then returning them to the original dataframe and deleting the original variable's column.

Principal Component Analysis (PCA)

50 predictor variables can be unwieldy to work with and, since a lot of the variables in this dataset are fairly closely related to each other, not all of the variables will contribute much predictive value to any potential models. Therefore, it made sense to use PCA to reduce the dimension of the dataset. BecausePCA can only take numeric variables, I created a subset of the original dataset that contained only the numeric variables. So, I used PCA to reduce the dimension of the dataset after scaling the variables; this gave me an idea of which numeric variables contributed the most to NPS and which contributed the least. This was also useful for data visualization which helped me see how predictor variables were related to each other and the response variable.

Multiple Correspondence Analysis (MCA)

MCA, which is similar to PCA but used for categorical variables, was used to see which levels of the categorical variables contributed the most to NPS. It provides another way to visualize how categorical variables related to each other, where closer points represented stronger correlations. Creating a scree plot also helped show how much each categorical variable contributed to NPS.

Random Forest

Random forest is an appropriate model for classifying data into groups because it is a flexible, nonparametric method of classification that can take in large amounts of data, so I used it to classify each observation into either promoter, passive, or deterrent based on the predictor variables. Additionally, the response variable must be discrete while the predictors must be continuous, which matches the types of my data. To do this, I converted the class of NPS from character to factor, then fit the model using the train function from the caret package with method parameter set to random forest. Another parameter, trControl, performed k-fold cross validation with k=10 to evaluate the performance of the model in a more robust way, so it looks at how well the model may generalize to other datasets. As random forest tend to overfit and lead to reduced accuracy in unseen test sets, it was important to validate the results. To assess model performance, I predicted NPS for the test set using the random forest model and compared it to the observed results in the test set with a confusion matrix.

I also used the random forest model to assess the importance of each variable on NPS by extracting the variable importances then converting them into a dataframe so that I could manipulate the data into a histogram that showed the percent that they contributed from most to least contribution.

K-Nearest Neighbor (KNN)

K-nearest neighbor is another appropriate method of classifying observations into NPS groups because it is a nonparametric method of classification, meaning it does not make any assumptions of the data other than assuming that points are more related to each other if they are closer to each other in space. However, since K-nearest neighbor does not take categorical data, I used indicator variables to incorporate the 9 categorical variables into the model. I fit models with between 1 and 20 clusters using the train function with method set to k-nearest neighbors, then selected the model that best fit the data. I used the same method of assessing model performance as with the random forest model and predicted NPS for the test set using the model with k-fold cross validation then compared the results to the actual NPS from the test set.

XGBOOST

XGBOOST is a method of classification that combines sets of simple decision trees to create a better fitting model, and it sequentially adds trees to reduce loss of information. It can take in nominal categorical predictors by hot-encoding them to binary features that represent each of the levels. Additionally, it does not assume normality and it is largely unaffected by multicollinearity, but it may overfit the data if it is not regularized. As with the previous models, I used the train function, this time with the method as xgbTree, to create the model. The tuneGrid parameter, which can be used to specify values for various parameters in the xgb model, was set to xgb_params, a table I created of parameters including eta, max_depth, and nrounds. Some parameters were set to default, but others contained multiple values so that the model could pick the most effective one. As with the other models, I also predicted NPS for the test set; I created the predicted class by predicting NPS from the test set and the actual class by predicting NPS in the test set, then compared the results.

I also used an XGBOOST model to assess the importance of each variable in predicting NPS by plotting the output of the xgb.importance function. This way, I could compare its results with the results of the random forest model as well as PCA and MCA to see which variables overall contributed the most to NPS.

Support Vector Machines (SVM)

SVM, which can be used for classification and nonlinear prediction, is another appropriate method of predicting NPS since the goal is to predict a categorical variable with three levels. The data cannot be separated into distinct categories using any straight lines because it is nonlinear and there are a lot of variables, so I used nonlinear SVM to classify the data into higher dimensions by using kernels to map the data into higher dimensional spaces until it can be separated. Because SVMs cannot take in categorical variables, I used the hot-encoded binary variables representing each level of each categorical variable.

SVMs are appropriate to use when datasets are high-dimensional and/or unstructured and they are used to classify data into groups, which aligns with the goal of these predictive models. Additionally, they can perform better than other deep learning methods for smaller training sets, and since the training set that I had to work with was not particularly large, SVM was likely to result in higher accuracy. To try to find the type of kernel that would result in the best performance of the SVM, I first recognized that it would need to be nonlinear since my data is nonlinear. Of the nonlinear kernels, the radial bias function is the most powerful and flexible because it uses exponents to create a classification line with infinite power.

Evaluation Metrics

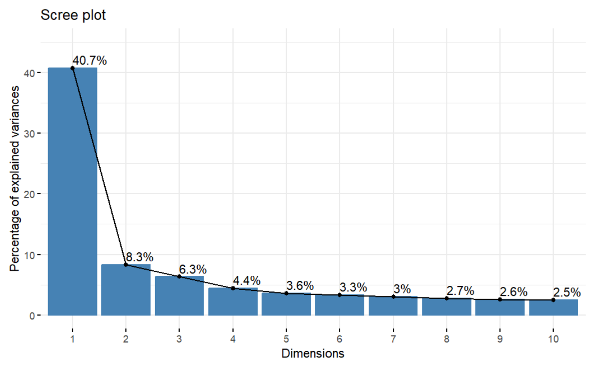
The metrics used to evaluate each model were accuracy and mean F1 score in addition to the specificity and sensitivity for each level of NPS. This way, I could assess how well a model performed based on how many true positives, false positives, true negatives, and false negatives it resulted in. Accuracy is the percentage of observations for which the model correctly predicts NPS for a dataset that the model was not trained on, and it works well with datasets that have balanced levels of the response variable, which I obtained by upsampling the levels with fewer observations. F1 scores, on the other hand, are measures of how well the model performs in both the recall and precision of the response variable, which is a useful metric when both recall and precision are important, which I believe is the case in these models since both false negatives and false positives have undesired but not disastrous consequences.

Since there are three levels of NPS, performance measures such as true positive, false positive, recall, and more do not have inherent meaning in a 3x3 confusion matrix , but they can be assessed separately in each of the three levels. For Promoters, a false positive could lead to the use of time and resources on areas of patient experience that do not have a significant impact on NPS, whereas a false negative could mean failure to dedicate sufficient time and resources to areas that do significantly impact NPS. For Detracter, the opposite is true. Neither of these types of errors are considerably worse than the other, but since hospitals want to improve patient experiences and increase their number of promoters, it may be more beneficial to prioritize Promoter recall in order to decrease the probability of missing a variable that has a significant effect on NPS.

Results

Analyzing Importance of Variables

PCA showed that the first principal component explained 40.7% of the data, which is a much larger percent than any of the other principal components which explain between 8.3 and 2.5% of the variance in the data. The first component was a linear combination of mostly the ratings of room ambiance, doctor visits, attendee care, staff attitude, and patient status info. The second component, which explained 8.5% of the variance, consisted of the variables mostly the ratings of doctor attitude, doctor treatment explanation, and food quality.



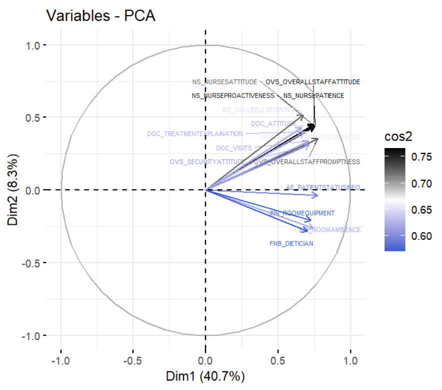


Fig 5. Scree Plot and Biplot for PCA

MCA for the categorical variables showed that none of them contributed especially heavily to NPS, with the first multiple correspondence component only explaining 6.5% of the data and the tenth explaining 3.8% of the data. The most influential variables to the first multiple correspondence component were: marital status (married, separated), semi special and special types of hospitalizations, as well as cardiology and general departments.

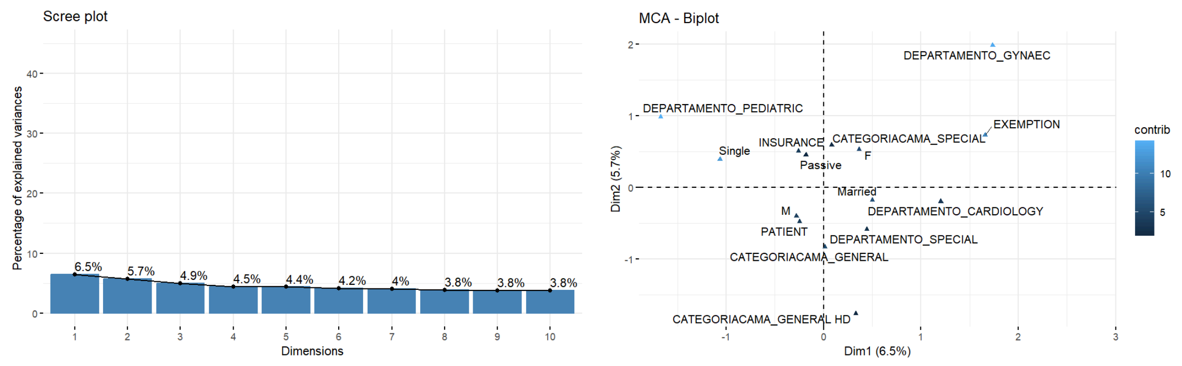


Fig 6. Scree plot and Biplot for MCA

Predictive Models

Since each of the four models perform better in some situations and worse in others, their ability to accurately predict NPS in the test set after being trained on the train set differed fairly significantly, with random forest, XGBOOST, and SVM all performing similarly well at about 90% accuracy while KNN only achieved around 68% accuracy as seen in Table 1 below.

KNN likely performed worse than the other models since it generally works better with a smaller number of variables, and at 50 variables, it is reasonable to assume that the large number of variables in this dataset is part of the reason why KNN underperformed. Since KNN is based on the idea that the closer points are to each other, the more related they are, it is also possible that the dataset fails to meet this assumption, thus also contributing to the low accuracy.

In any case, since KNN did not compete with the other models in terms of accuracy which was an important metric in determining model performance, it will not be further assessed as a contender for the best model for this dataset and research questions.

Model	Accuracy	Mean F1
Random Forest	0.8939	0.8672
KNN	0.6802	0.8943
XGBOOST	0.8942	0.8524
SVM	0.9131	0.9225

Table 1: Comparing evaluation metrics between models

Random Forest

To assess variable importance using the random forest model, I plotted the variables in descending importance as calculated using the varImp function from the caret package. As seen in Fig 7. below, the five most important variables together contribute to the majority of the variance in NPS. Value for money and age are the most important at around 11%, and estimated cost, customer engagement satisfaction, and days in hospital are the next most important at around 8.5%, 7.5%, and 6.5% respectively.

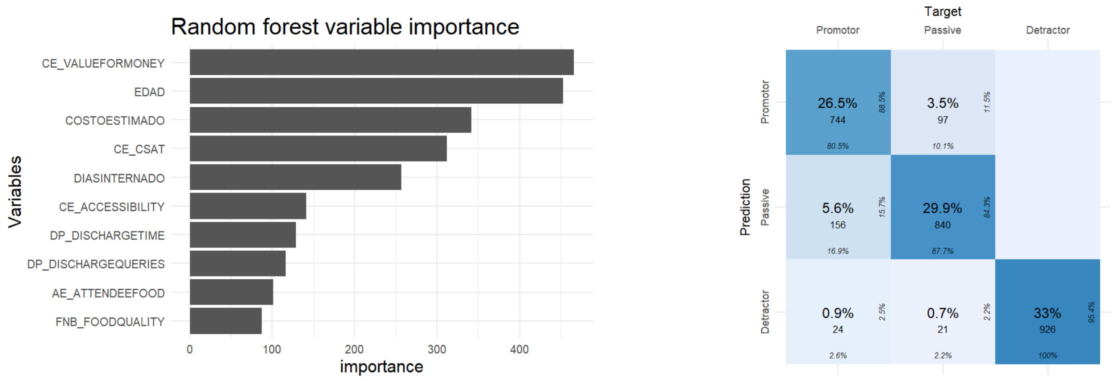


Fig 7. Random Forest Variable Importance and Confusion Matrix

From Table 2 below, we see that the random forest model was best at predicting sensitivity, specificity, true positives, and true negatives for Deterrent and fairly worse at predicting Promoters. In general, it had higher proportions of specificity and true negative than sensitivity and true positive, meaning random forest tended to prioritize correctly identifying true negatives (or the absence of a level of NPS) over correctly identifying true positives (or the presence of a level of NPS). It was worst at correctly identifying Promoters, and best and correctly identifying Deterrents while also avoiding incorrectly identifying any Deterrents.

Level	**Sensitivity**	**Specificity**	**True Pos**	**True Neg**
Promoter	0.8052	0.9485	0.8847	0.9085
Passive	0.8768	0.9157	0.8434	0.9349
Deterrent	1.0000	0.9761	0.9537	1.0000

Table 2: Evaluating performance random forest model for each level of NPS

XGBOOST

As with the random forest model, I plotted the variables in descending importance as calculated using the varImp function from the caret package to assess variable importance with the XGBOOST model. As Fig 8. shows, the five most important variables together contribute about 56% of the variance in NPS, with age contributing around 17% and value for money and estimated cost contributing around 12% each.

The XGBOOST model had the most trouble with correctly predicting Passive and Promoter and tended to conflate the two levels of NPS, resulting in false positives for Passive when the observed response was Promoter and false positives for Promoter when the observed response was Passive. It resulted in false positives for Promoters around twice as much as it did for Passives at 5.6% and 3.5% respectively. Of the three levels, it was best at correctly predicting Detractor; there were no false positives for Detractor, but when the predicted value was Detractor, the observed value was either Promoter or Passive a total of 1.5% of the time.

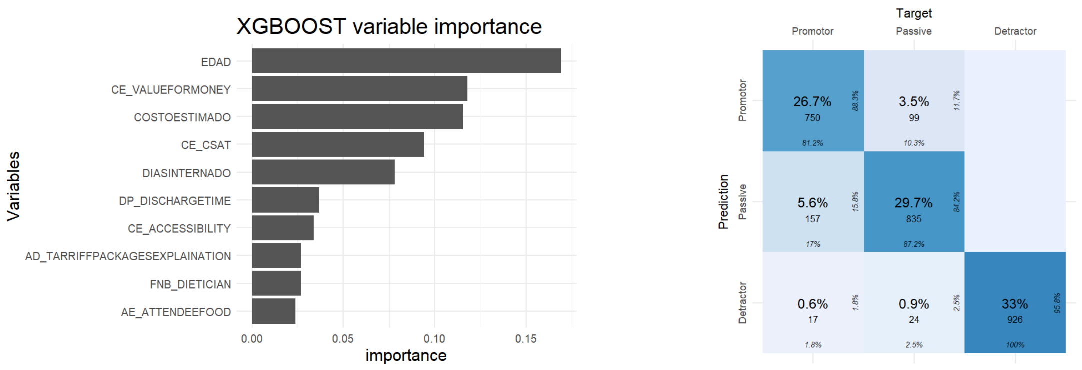


Fig 8. XGBOOST Variable importance and Confusion Matrix

From Table 3 below, we see that the XGBOOST model was best at predicting sensitivity, specificity, true positives, and true negatives for Deterrent, then for Passive, then for Promoter. In general, it had higher proportions of specificity and true negative than sensitivity and true positive, meaning XGBOOOST tended to prioritize correctly identifying true negatives (or the absence of a level of NPS) over correctly identifying true positives (or the presence of a level of NPS). As also seen in the confusion matrix, it was worst at correctly identifying true positives for Promoters.

Level	**Sensitivity**	**Specificity**	**True Pos**	**True Neg**
Promoter	0.8117	0.9475	0.8834	0.9112
Passive	0.8716	0.9151	0.8417	0.9323
Deterrent	1.0000	0.9782	0.9576	1.0000

Table 3: Evaluating performance XGBOOST model for each level of NPS

SVM

As seen in Fig 9., SVM has the highest accuracy for predicting Promoters at 32.9% and the lowest accuracy for predicting Passives at 25.6%. Only once did it incorrectly predict Passive when the observed value was Promoter, but it incorrectly predicted Promoter 8.5% of the time when the correct value was Passive. Additionally, when the observed value was Detractor, it only resulted in 0.2% incorrect predictions, and these only occurred when the predicted value was Promoter. Overall, this means that

SVM was very accurate in predicting Promoters and Detractors, but considerably less accurate in predicting Passives as it tended to conflate them with Promoters.

Since there is no built-in way to get variable importance from SVM models as there is with random forest and XGBOOST models, I used recursive feature elimination (RFE) to fit a reduced SVM model multiple times so that the least important variable was removed each time. I had to use a significantly reduced set of variables since each model took a while to fit, so I selected the top 10 most important variables as determined by the random forest and XGBOOST models to rank their relative importance to NPS using SVM with RFE. RFE determined the five most important variables to be value for money, rating of the quality of customer engagement satisfaction, rating of the quality of the accessibility of customer engagement, rating of the quality of the duration of the discharge process, and rating of the quality of the explanation of available rates and packages during the admission process.

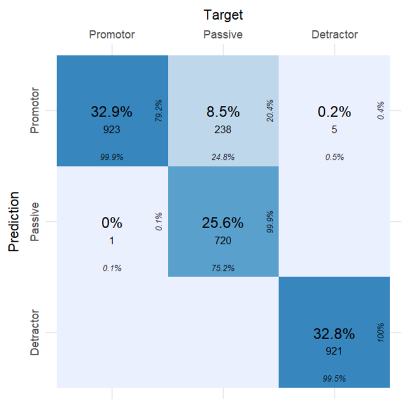


Fig 9. SVM Variable Importance and Confusion Matrix

From Table 4 below, we see that the SVM model was best at predicting sensitivity, specificity, true positives, and true negatives for Deterrent. In general, it had higher proportions of specificity and true negative than sensitivity and true positive, meaning SVM tended to prioritize correctly identifying true negatives (or the absence of a level of NPS) over correctly identifying true positives (or the presence of a level of NPS). As also seen in the confusion matrix, it was worst at correctly identifying the presence of Passives because it tended to conflate them with Promoters.

Level	**Sensitivity**	**Specificity**	**True Pos**	**True Neg**
Promoter	0.9989	0.8710	0.7916	0.9994
Passive	0.7516	0.9995	0.9986	0.8860
Deterrent	0.9946	1.0000	1.0000	0.9974

Table 4: Evaluating performance SVM model for each level of NPS

Final Model Selection

Based on the assessments of variable importance which included PCA and MCA as well as random forest, XGBOOST, and SVM assessments of variable importance, the variables were selected to be in the model if they appeared commonly throughout these methods or showed strong evidence of being important to contributing to NPS. For example, food quality was determined to be an important factor in PCA, random forest, and XGBOOST, so it was included in the final set of variables to be included in the predictive models. Age, value for money, estimated cost, customer engagement satisfaction, and number of days in hospital appeared in both the random forest and XGBOOST assessments as the top five most important variables contributing to NPS, so they were also included.

The variables included in the final models are, in no specific order, as follows:

- Estimated cost of visit
- Rating of the quality of attendee food
- Rating of the quality of regular nutritionist advice
- Rating of the quality of patient status information updates to attendees
- Rating of the quality of attendee care
- Rating of the quality of the overall security attitude
- Rating of the quality of the doctor's treatment explanations
- Rating of the overall emergency room services
- Age of patient
- Department of the hospital
- Country in which the patient was treated
- Category of hospitalization
- Rating of the quality of the patient's room's ambiance
- Rating of the quality of the patient's room's cleanliness
- Rating of the quality of the value for money
- Number of days as a patient
- Rating of the quality of the accessibility
- Rating of the quality of the food
- Rating of the quality of customer engagement satisfaction
- Rating of the quality of the explanation of available rates and packages during the admission process
- Rating of the quality of the duration of discharge process
- Net Promoter Score (NPS)

When comparing all three models, SVM had the highest accuracy for predicting Promoters at 32.9% compared to 26.5% for random forest and 26.7% for XGBOOST. With the research question being most interested in Promoters since one of the goals is to see which variables correctly predict a patient's NPS being Promoter, it is clear that SVM is the most valuable model in this regard since its accuracy for doing so is higher than the other two models. This way, it would likely be most beneficial to hospitals to increase the attention and resources provided to the variables that contribute most to SVM.

Discussion and Reflection

A strength of this dataset is that it contains a decent spread of variables covering different areas such as country of treatment, patient's age, estimated cost, number of days in hospital, category of hospitalization, and more, and as a result, it can predict NPS with good accuracy. Additionally, the train set contained 6,523 observations, which was a more than sufficient amount according to the rule of ten since the number of observations was more than ten times larger than the amount of parameters of which there were 22. The labels and descriptions for the variables were also clear and sufficiently explanatory as to what they were measuring, which is important for being able to interpret any results. Finally, one of the greatest strengths of this dataset is that it represents real life occurrences and is thus applicable and useful in the real world when used to analyze relationships between variables and to train and assess the performance of predictive models.

A limitation of this dataset is that all of the rating variables were very similarly distributed to each other, and many of them measured similar phenomena, so the overall amount of information gathered by these variables is likely not as much as expected since their individual influence on NPS is not as great. For this reason, to avoid overfitting, and to improve interpretability, I selected only the variables that appeared to be the most important based on the results from PCA, MCA, and the variable importance results from the random forest and XGBOOST models to include in the final models. In doing so, some information from the variables that were left out was lost and the accuracy of the model may be decreased. Additionally, I did not have a very precise method of choosing which variables to include since I just looked at the variables deemed most important using different methods and combined those results.

One challenge was finding models that fit the types of data in the dataset, which ranged from nominal to continuous, or finding ways to modify either the data or the models to be able to accommodate the various data types. I dealt with this by dummy coding the nominal variables and treating the ordinal variables as if they were numeric because there were a lot of them and I worried that dummy coding all of them might lead to problems since the models I used are originally intended to be used with numeric variables. However, treating ordinal data as numeric could have unintended consequences if the levels of the ordinal variables are not equally spaced. I think it is reasonable to assume that the ordinal variables in this dataset are equally spaced, so treating them as numeric should not cause any big problems.

Potential ways of improving and extending these findings could range from changing model parameters to manipulating the original data differently to collecting more data.

My method of selecting model parameters was to accept default parameters when they were appropriate and run through a sequence of values for other parameters so that the model could choose the values that were the most accurate since the models were all trained to prioritize accuracy. If the sequences I chose happened to not include a value that would have resulted in higher accuracy, choosing larger ranges of values or choosing values based on some logic could result in the models having better accuracy.

I trained the final models on a reduced dataset that contained only the 22 most important variables as determined by PCA and MCA and assessed variable importance by running random forest and XGBOOST models with all 50 variables. As a result, some information by the other less important variables that were left out of the final models could be lost which could result in worse models if that information was important to predicting NPS. However, because the accuracy of the models did not change much from using all 50 variables to only 22, I believe it is likely that the manipulations I did with the data prior to training did not result in significantly worse outcomes.

Lastly, another way to extend these finals would be to collect more data either from the same locations as in the original dataset to improve robustness or from other locations to improve the generalizability of the findings. This method of improving results is more costly in terms of time and resources, so it is not feasible for independent researchers.

Conclusion

Since my research questions were which predictor variables contributed the most to Net Promoter Score (NPS) and, relatedly, how accurately NPS could be predicted using those variables, using statistical methods and models to answer these questions resulted in information that could be useful to hospital administration staff who are aiming to improve their patients' experiences by increasing the probability with which patients recommend their services to others.

All methods tended to agree on which variables were among the top ten most important to predicting NPS. SVM, which had the highest overall accuracy as well as the best overall specificity and sensitivity for predicting Promoters and Deterrents which were the levels of NPS that I was most interested in accurately predicting since they reveal patients' opinions about the hospitals' services as opposed to Passives who were more neutral about their experiences and thus provided less information.

As previously stated, the practical implications about these results are that the most important predictors of NPS should be given more resources to improve those experiences so that overall patient experience will see the most improvement which should increase the number of Promoters. These variables include value for money, customer engagement satisfaction, accessibility, discharge time, and explanation of available rates and packages. So, it may be beneficial to dedicate more resources to improving patients' ratings of these variables in order to improve overall patient satisfaction.

Works Cited

"Predicting NPS to Improve Patient Experience." *Kaggle*, Concurso Maestría Analítica Pontificia Universidad Javeriana- Métodos y Aplicaciones de Analítica I, www.kaggle.com/competitions/puj-1910-predicting.

Appendix

Natalie Bondurant

2025-05-20

```r
#unzip("puj-1910-predicting.zip")
dat2 = read.csv("Training Data.csv")
```

```r
library(corrr)
library(ggcorrplot)
library(tidyverse)
library(ggplot2)
library(caret)
library(caretEnsemble)
library(psych)
library(GGally)
library(rpart)
library(cvms)
library(randomForest)
library(entropy)
library(keras)
library(mlbench)
library(dplyr)
library(magrittr)
library(neuralnet)
library(magrittr)
library(tensorflow)
library(rpart)
library(qgraph)
library(corrplot)
library(bootnet)
library(FNN)
library(gmodels)
library(FactoMineR)
library(factoextra)
```

```r
colnames(dat2)
```

VISUALIZING WITH NETWORK

```r
numeric = dat2[sapply(dat2, is.numeric)]
nw <- estimateNetwork(numeric, default = 'EBICglasso', tuning = 0.5, threshold = TRUE)

maxi <- max(nw$graph)
qgraph(nw$graph, maximum=maxi, theme = "colorblind", palette = "rainbow",layout = 'spring')
title('Network of All Variables', cex.main=1.2, line=3)
```

HISTOGRAMS OF NUMERICAL VARIABLES

```r
plots = for (n in colnames(numeric)){
  ggplot(numeric, aes(x = numeric[[n]]))+
    geom_histogram()+
    ggtitle(n)
}
#plots
ggplot(dat2, aes(x = COSTOESTIMADO))+
  geom_histogram(fill = "royalblue3", color = "black")+
  ggtitle("Estimated Cost")+
  labs(
    y = "Frequency",
    x = "Estimated Cost (in dollars)"
  )+
  theme_minimal()
```

LINE PLOTS OF NPS SCORE AGAINST EACH NUMERIC VARIABLE

```r
plotline = for (n in colnames(numeric)){
  ggplot(numeric, aes(x = numeric[[n]], y= numeric$SCORENPS))+
    geom_line()+
    ggtitle(n)
}
#plotline
```

BAR CHARTS OF CATEGORICAL VARIABLES

```r
cat = dat2[sapply(dat2, is.character)]

ack = for (c in colnames(cat)){
  print(ggplot(cat, aes(x = cat[[c]],color = "royalblue3"))+
    geom_bar(fill = "royalblue3", color = 'black')+
    ggtitle(c))+
    theme_minimal()
}
ggplot(dat2, aes(x = CATEGORIACAMA))+
  geom_bar(fill = "royalblue3", color = 'black')+
  ggtitle("Bar Plot of Category")+
  labs(
    y = "Frequency",
    x = "Category of Hospitalization"
  )+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

FREQUENCY TABLES OF CATEGORICAL VARIABLES

```r
colnames(cat)
cat = subset(cat, select = -c(FECHAENTRADA, FECHASALIDA))
#sapply(cat, function(x) table(x))
```

SUMMARY STATISTICS OF NUMERICAL VARIABLES

```r
num = subset(numeric, select = -c(CONSECUTIVO, ID))

numsumstats = sapply(num, function(x) summary(x))
#numsumstats
```

FINDING OUTLIERS

```r
catfreq = sapply(num, function(x) table(boxplot.stats(x)$out))
#catfreq
```

CORRELATION MATRIX PLOT

```r
library(corrplot)
library(corrr)
cormat = cor(num)
corrplot(cormat, type = 'full', tl.cex= .3)
```

MANIPULATING DATA FOR ANALYSIS

DATA FOR ANALYSIS

```r
a = dat2
a = subset(a, select = -c(FECHASALIDA, FECHAENTRADA, ID, CONSECUTIVO, SCORENPS))
a$NPS = as.factor(a$NPS)
```

ONE-HOT ENCODING CATEGORICAL VARIABLES INTO

```r
enc = a

dum = dummyVars('~.', data = enc)
datdum = data.frame(predict(dum, newdata = enc))
datdum$NPS = a$NPS
datdum = subset(datdum, select = -c(NPS.Promotor, NPS.Detractor, NPS.Passive))
#colnames(datdum)
```

CATEGORICAL VARIABLES AS FACTORS NUMERICAL VARIABLES

```r
fac = a
fac$ESTADOCIVIL = as.factor(fac$ESTADOCIVIL)
fac$CATEGORIACAMA = as.factor(fac$CATEGORIACAMA)
fac$DEPARTAMENTO = as.factor(fac$DEPARTAMENTO)
fac$CATEGORIA = as.factor(fac$CATEGORIA)
fac$ESTADO = as.factor(fac$ESTADO)
fac$PAIS = as.factor(fac$PAIS)
fac$ZONA = as.factor(fac$ZONA)
fac$GENERO = as.factor(fac$GENERO)

catmca = fac[sapply(fac, is.factor)]
catmca = subset(catmca, select = -c(PAIS, ESTADO, ZONA))
#colnames(catmca)

numa = fac[sapply(fac, is.numeric)]
```

```r
numa$NPS = fac$NPS
#colnames(numa)
numonly = fac[sapply(fac, is.numeric)]
```

BALANCING NPS

```r
set.seed(29374)
upa = upSample(x = datdum[,-ncol(datdum)],
                 y = datdum$NPS)
table(a$NPS)
table(upa$Class)
#nrow(upa)
```

REDUCED VARIABLES

```r
red = subset(a, select = c(COSTOESTIMADO, AE_ATTENDEEFOOD, AE_PATIENTSTATUSINFO, AE_ATTENDEECARE,
                           OVS_SECURITYATTITUDE, DOC_TREATMENTEXPLAINATION, EM_OVERALL, EDAD,
                           AD_TARRIFFPACKAGESEXPLAINATION, DEPARTAMENTO, ESTADO, CATEGORIACAMA,
                           INR_ROOMAMBIENCE, FNB_DIETICIAN, INR_ROOMCLEANLINESS, CE_VALUEFORMONEY,
                           CE_CSAT, DIASINTERNADO, CE_ACCESSIBILITY, DP_DISCHARGETIME,
                           FNB_FOODQUALITY, NPS))
redd = dummyVars('~.', data = red)
reddum = data.frame(predict(redd, newdata = red))
reddum$NPS = a$NPS
reddum = subset(reddum, select = -c(NPS.Promotor, NPS.Detractor, NPS.Passive))
upred = upSample(x = reddum[,-ncol(reddum)],
                 y = reddum$NPS)
```

VERY REDUCED VARIABLES FOR SVM RFE

```r
redred = subset(upred, select = c(EDAD, CE_VALUEFORMONEY, COSTOESTIMADO, CE_CSAT, DIASINTERNADO,
                                  DP_DISCHARGETIME, CE_ACCESSIBILITY, AD_TARRIFFPACKAGESEXPLAINATION,
                                  FNB_FOODQUALITY, Class))
```

SPLIT INTO TEST AND TRAIN

```r
traina = sample(c(TRUE, FALSE), nrow(a),
            replace = TRUE, prob = c(0.7, 0.3))

atrain = a[traina, ]
atest = a[!traina, ]

numtrain = numa[traina, ]
numtest = numa[!traina, ]

dumtrain = datdum[traina, ]
dumtest = datdum[!traina, ]

uptrain = upa[traina, ]
uptest = upa[!traina, ]

redtrain = upred[traina, ]
```

```r
redtest = upred[!traina, ]

redredtrain = redred[traina, ]
redredtest = redred[!traina, ]
```

K-FOLD CROSS VALIDATION

```r
library(ISLR)

set.seed(378)
kcv = trainControl(method = 'cv',
                   number = 10,
                   verboseIter = TRUE,
                   classProbs = TRUE,
                   summaryFunction = multiClassSummary,
                   savePredictions = TRUE)
```

MODELING

FACTORIAL ANALYSIS OF MIXED DATA (FAMD)

```r
library(factoextra)
ffamd = FAMD(fac, graph = FALSE)
#ffamd

ind_coord = as.data.frame(ffamd$ind$coord)
ind_cos2 = as.data.frame(ffamd$ind$cos2)
ind_coord$cos2 = rowSums(ind_cos2)
ind_coord$ID = paste0("ind_", seq_len(nrow(ind_coord)))
top_n = 250
selected_inds = ind_coord[order(-ind_coord$cos2), ][1:top_n, ]

ggplot(selected_inds, aes(x = Dim.1, y = Dim.2)) +
  geom_point(aes(color = cos2)) +
  scale_color_gradientn(colors = c("royalblue3", "chartreuse3", "gold2")) +
  labs(title = "FAMD - Individuals factor map",
       x = "Dim.1", y = "Dim.2") +
  theme_minimal()
```

MULTIPLE CORRESPONDENCE ANALYSIS (MCA)

```r
fmca = MCA(catmca, ncp = 5, graph = FALSE)
fviz_mca_biplot(fmca, repel = TRUE, invisible = 'ind',
                select.var = list(contrib = 15),
labelsize = .5,
col.var = 'contrib')
fviz_screeplot(fmca, addlabels = TRUE, ylim = c(0, 45))
#fmca

mcva = get_mca_var(fmca)
head(mcva$contrib)
```

PCA

```r
pc = prcomp(numonly,
            center = TRUE,
            scale. = TRUE)
#attributes(pc)
#pc$center
#pc$scale
#pc
#summary(pc)
fviz_eig(pc, addlabels = TRUE)
fviz_cos2(pc, choice = "var", axes = 1:2)
get_eigenvalue(pc)

pc1_pc2 <- pc$rotation[, c("PC1", "PC2")]
#pc1_pc2
pc1.6 <- pc$rotation[, c("PC1", "PC2", 'PC3','PC4','PC5','PC6')]
#pc1.6
kmeans<-eclust(pc1_pc2, k=3)

fviz_screeplot(pc, color = "royalblue3", addlabels = TRUE, ylim = c(0, 45))

library(ggfortify)
autoplot(pc1.6, loadings=TRUE, loadings.colour='black', loadings.label=TRUE, loadings.label.size=0.5)
abc = fviz_pca_var(pc,
             col.var = "cos2",
             gradient.cols = c("royalblue3", "white", "black"),
             repel = TRUE,
             select.var = list(contrib = 15),
             labelsize = 2
             )
abc

library(factoextra)
eig = get_eigenvalue(pc)
eig

pcva = get_pca_var(pc)
pcva$coord          # Coordinates
pcva$contrib        # Contributions to the PCs
pcva$cos2           # Quality of representation
```

Regression and VIF

```r
numnps = numonly
numnps$SCORENPS = dat2$SCORENPS
model = lm(SCORENPS~., data = numnps)
#summary(model)

library(car)
#vif(model)

NS_avg = (a$NS_CALLBELLRESPONSE+ a$NS_NURSESATTITUDE+ a$NS_NURSEPROACTIVENESS+ a$NS_NURSEPATIENCE)/4
DOC_avg = (a$DOC_ATTITUDE +a$DOC_VISITS +a$DOC_TREATMENTEFFECTIVENESS)/3
```

FUNCTION FOR MEAN F1

```r
meanf1 = function(x) {
  # precision = true pos / true pos + false pos
  # recall = true pos / true pos + false neg
  # [row, col]
  p1 = x[1,1] / (x[1,1] + x[1,2] + x[1,3])
  r1 = x[1,1] / (x[1,1] + x[2,1] + x[3,1])
  p2 = x[2,2] / (x[2,2] + x[2,1] + x[2,3])
  r2 = x[2,2] / (x[2,2] + x[1,2] + x[3,2])
  p3 = x[3,3] / (x[3,3] + x[3,1] + x[3,2])
  r3 = x[3,3] / (x[3,3] + x[1,3] + x[2,3])

  p = (p1+p2+p3)/3
  r = (r1+r2+r3)/3

  f1 = 2* (p*r)/(p+r)
}
```

RANDOM FOREST

```r
library(caret)
library(randomForest)
library(varImp)

rfd <- caret::train(Class ~. ,
              data = redtrain,
              method = 'rf',
              trControl = kcv,
              metric = 'Accuracy')
#rfd
plot(rfd)

p1d = predict(rfd, redtrain)
p1dt = confusionMatrix(p1d, redtrain$Class)
#p1dt

p2d = predict(rfd, redtest)
p2dt = confusionMatrix(p2d, redtest$Class)
#p2dt
p2df = as_tibble(p2dt$table)
plot_confusion_matrix(p2df, target_col = "Reference", prediction_col = "Prediction", counts_col = "n")

var_imp <- caret::varImp(rfd, scale=FALSE)$importance
var_imp <- data.frame(variables=row.names(var_imp), importance=var_imp$Overall)
var_imp %>%
  slice_max(order_by = importance, n=10) %>%
  ggplot(aes(x=reorder(variables, importance), y=importance)) +
    geom_bar(stat='identity') +
    coord_flip() +
    xlab('Variables') +
    labs(title='Random forest variable importance') +
    theme_minimal() +
    theme(axis.text = element_text(size = 10),
          axis.title = element_text(size = 15),
          plot.title = element_text(size = 20),  )

f1rf = table(redtest$Class, p2d)
wa = meanf1(f1rf)
```

K NEAREST NEIGHBOR

```r
library(caret)

set.seed(874)
grid = expand.grid(.k=seq(1,20,by=1))
knn <- caret::train(Class~., data=redtrain, method="knn", metric='Accuracy' ,trControl=kcv,
                    preProcess = c('center','scale'))
#knn
knn.1 <- knn$bestTune
plot(knn)

pred = predict(knn, newdata = redtest)
conmat = confusionMatrix(pred, redtest$Class)
#conmat
formcon = as_tibble(conmat$table)

plot_confusion_matrix(formcon, target_col = "Reference", prediction_col = "Prediction",
                      counts_col = "n")

kt = matrix(
  c(750,99,0, 157,835,0, 17,24,926),
  nrow = 3,
  ncol = 3,
  byrow = TRUE
)
kt
ktf1 = meanf1(kt)
ktf1
```

XGBOOST

```r
library(dplyr)
library(cvms)
library(xgboost)
xgb_params <- expand.grid(
  eta = seq(from = 0.1, to = 1, by = 0.2),
  max_depth = c(1,3,5)^2,
  nrounds = c(5000,1000,500),
  gamma = 0,
  colsample_bytree = 1,
  min_child_weight = 1,
  subsample = 1
)

xg <- caret::train(Class~., data=redtrain,
  method = 'xgbTree',
  metric = 'Accuracy',
  tuneGrid = xgb_params,
  trControl = kcv
)
#xg

xpred = predict(xg, newdata = redtest)
cm = confusionMatrix(xpred, redtest$Class)
#cm

cfm = as_tibble(cm$table)
plot_confusion_matrix(cfm, target_col = "Reference", prediction_col = "Prediction",
                      counts_col = "n")

xgimp <- caret::varImp(xg, scale=FALSE)$importance
view(xgimp)
xgimp <- data.frame(variables=row.names(xgimp), importance=xgimp$Overall)
xgimp %>%
  slice_max(order_by = importance, n=10) %>%
  ggplot(aes(x=reorder(variables, importance), y=importance)) +
    geom_bar(stat='identity') +
    coord_flip() +
    xlab('Variables') +
    labs(title='XGBOOST variable importance') +
    theme_minimal() +
    theme(axis.text = element_text(size = 10),
          axis.title = element_text(size = 15),
          plot.title = element_text(size = 20),  )

xgt = matrix(
  c(750,99,0, 157,835,0,174,24,926),
  nrow = 3,
  ncol = 3,
  byrow = TRUE
)
xgt
xgf1 = meanf1(xgt)
xgf1
```

SVM

```r
library(e1071)
library(kernlab)
library(MLmetrics)

set.seed(32784)
svgrid <- expand.grid(C = 2^(1:5), sigma = 2^(-15:3))
smod = caret::train(Class~., data = redtrain, method = 'svmRadial', trControl = kcv,
                    tuneLength = 10, tuneGrid = svgrid)
#smod
#plot(smod)

svpred = predict(smod, newdata = redtest)
coma = confusionMatrix(data = svpred, reference = redtest$Class)
```

```r
#coma

fcoma = as_tibble(coma$table)
plot_confusion_matrix(fcoma, target_col = "Reference", prediction_col = "Prediction",
                      counts_col = "n")

srFuncs = caretFuncs
srFuncs$fit = function(x, y, first, last, ...) {caret::train(
  x, y,
  method = "svmRadial",
  preProcess = c("center", "scale"),
  trControl = trainControl(method = "cv", number = 10),
  tuneLength = 3)}
ctrl = rfeControl(functions = srFuncs, method = "cv", number = 10)

ncol(redredtrain)
head(redredtrain[, -10])
svmrfe = rfe(redredtrain[, -10],
             redredtrain$Class,
             sizes = c(1:9),
             rfeControl = ctrl)
svmrfe

svmt = matrix(
  c(923,238,5, 1,720,0, 0,0,921),
  nrow = 3,
  ncol = 3,
  byrow = TRUE
)
svmt
svmf1 = meanf1(svmt)
svmf1
```

NETWORK MODEL OF FINAL NUMERIC VARIABLES

```r
numeric2 = redredtrain[sapply(redredtrain, is.numeric)]
nw2 <- estimateNetwork(numeric2, default = 'EBICglasso', tuning = 0.5, threshold = TRUE)

maxi2 <- max(nw2$graph)
qgraph(nw2$graph, maximum=maxi2, theme = "colorblind", palette = "rainbow",layout = 'spring')
```