MLM Semester Project – Bret Gitar

1. Define Hypothesis and value proposition

Since I switched teams within Optum one month into the MLM course, I had limited access to data during the time of project development. I brought this up to the DSU instructors and they recommended I find a dataset on Kaggle to use for my project. The dataset I am analyzing is an employee attrition dataset for a health care company. While this isn’t a typical healthcare dataset used in the DSU program, employee attrition is still extremely relevant and valuable for a company with around 325,000 employees. Also, as someone who works closely with human resource data this will be relevant to my future work.

The question I will be investigating is can we accurately predict employee attrition? If you know which features impact employee attrition, you can use this knowledge to potentially keep more good talent in your company, or at least know why people decide to leave and improve your company’s culture and morale.

H0: If Salary is high or increasing then attrition will be less likely to occur

HA: Salary and attrition are not related

1. Feature selection

The employee attrition dataset has 1470 rows, and 35 feature variables. The first thing I did was subset the dataset to contain only features which I thought would be useful in predicting attrition. The variables left to analyze are below with a brief description.

**Attrition2**: 1=Employee left company, 0=Employee stayed at company

**YearsInCurrentRole**: Number of years spent in current job

**StockOptionLevel**: Ordinal categorical feature for stock option level

**Gender2**: Flag feature for gender. 1=M, 2=F.

**Education**: Ordinal categorical feature for education. 5 is highest, 1 is lowest

**DistanceFromHome**: Distance in miles from office person lives

**Age**: Age of person

**MonthlyIncome**: Monthly income of person

**MaritalStatus2**: Categorical feature for marital status. 1=single, 2=married, 3=divorced

**JobSatisfaction**: Ordinal categorical feature for job satisfaction. 5 is highest, 1 is lowest.

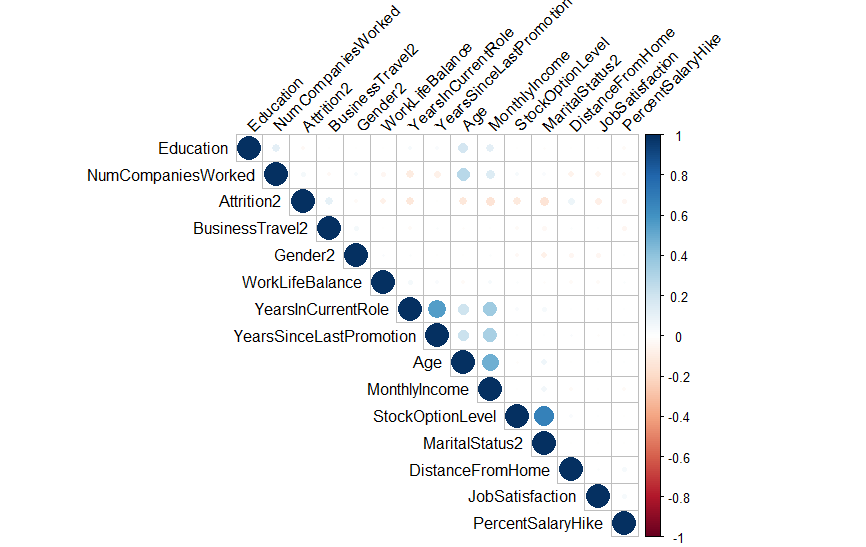
**BusinessTravel2**: Categorical feature for travel. 3 is most travel, 1 is no travel

**WorkLifeBalance**: Ordinal categorical feature for work life balance, 5 is best, 1 is worst.

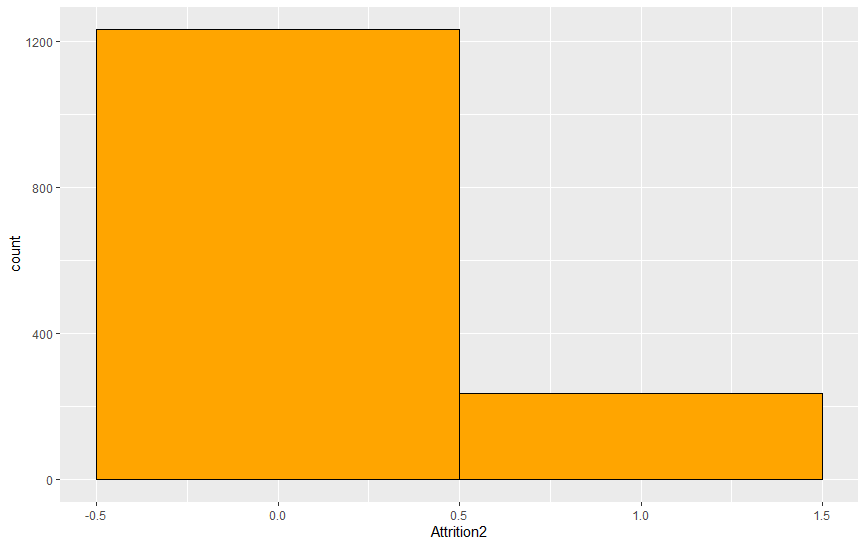
**PercentSalaryHike**: Percent salary increase last year

**NumCompaniesWorked**: Number of company’s person has worked for

**YearsSinceLastPromotion**: Number of years since last promotion

Next, I created a correlation matrix to see which variables are correlated with each other. This will be especially important in logistic regression, and Bayesian classification which both assume no endogeneity between the covariates. 

Looking at the results of the correlation plot, there are a few features that are correlated. It looks like StockOptionLevel and MaritalStatus have a strong correlation. Also, there is correlation between a group of features including YearsInCurrentRole , YearsSinceLastPromotion, Age, and MonthlyIncome. It makes sense to drop a few of these to remove reduce any multicolinearity. The good news is most of these features are getting at the same thing, so whatever I choose to drop probably won’t have a drastic impact on any models I build. I am going to drop Marital Status, Age, and YearsSinceLastPromotion from our model features.

One more important feature to examine is the dependent feature, Attrition. There is some class imbalance, with 1,233 “No”, and 237 “Yes”. With only about 16% of our population leaving, this could create some issues when predicting attrition with various models. I tested out two methods to deal with this, which were oversampling and SMOTE, but I still got better results leaving the distribution of attrition as it is. 

Attrition2 Freq

1 No 1233

2 Yes 237

1. Model selection

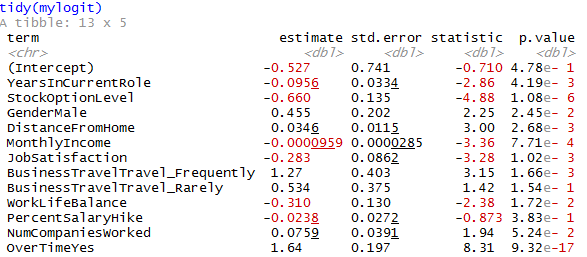
During the semester I ran several models, and went into detail with 7 different types of models with my employee attrition dataset. These models were:

1. Logistic regression
2. Random Forest
3. XGBoost
4. Bayesian classification
5. Knn
6. SVM
7. Artificial neural network

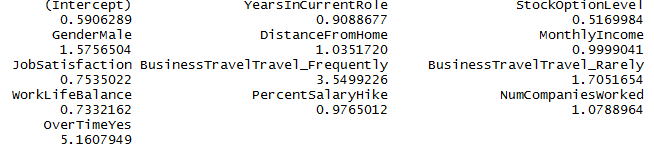
After running all of these models, I got the best results using Logistic, random forest, XGBoost, and naïve Bayesian classification. I think Knn, SVM, and Ann could have worked well, but they’re a little more complicated to tune and set up than the other four models. Since my hypothesis is a classification problem, I believe the four models I chose to use will work well.

**Logistic Regression Model**

Attrition2 = YearsInCurrentRole + StockOptionLevel + GenderMale + DistanceFromHome + MonthlyIncome + JobSatisfaction + BusinessTravel\_Frequently + BusinessTravel\_Rarely + WorkLifeBalance + PercentSalaryHike + NumCompaniesWorked + OverTimeYes



Using the features I determined to reduce the risk on multicolinearity, I ran my losgistic regression model. Several features were determined to be significant at the 5% level or higher, these features were YearsInCurrentRole, GenderMale, StockOptionLevel, DistanceFromHome, MonthlyIncome, JobSatisfaction, BusinessTravel\_Frequently, WorkLifeBalance, and OverTimeYes. The only two features that didn’t have any significant impact on the model were BusinessTravel\_Rarely, and PercentSalaryHike. Next, I exponentiated the beta coefficients of the independent features in order to interpret the estimate correctly.



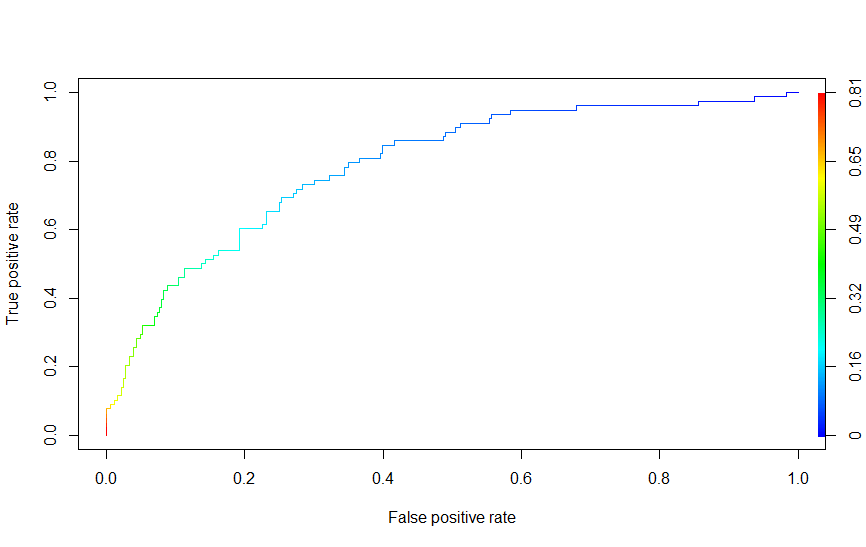
The result that stands out the most here is the coefficient for OverTimeYes at 5.16. This means that employees who work over time are about 516% more likely to leave the company than employees who don’t work overtime, and it’s significant at the 1% level. This is likely the most important feature. I also expected MonthlyIncome to have a big impact, while the coefficient is only 0.999, which means employees are about 0.1% more likely to stay for every $1 increase in monthly income, the scale here is very different from the binary flags in the model. Perhaps this would be more comparable with normalized data of a feature transformation on MonthlyIncome.

FALSE TRUE

No 349 14

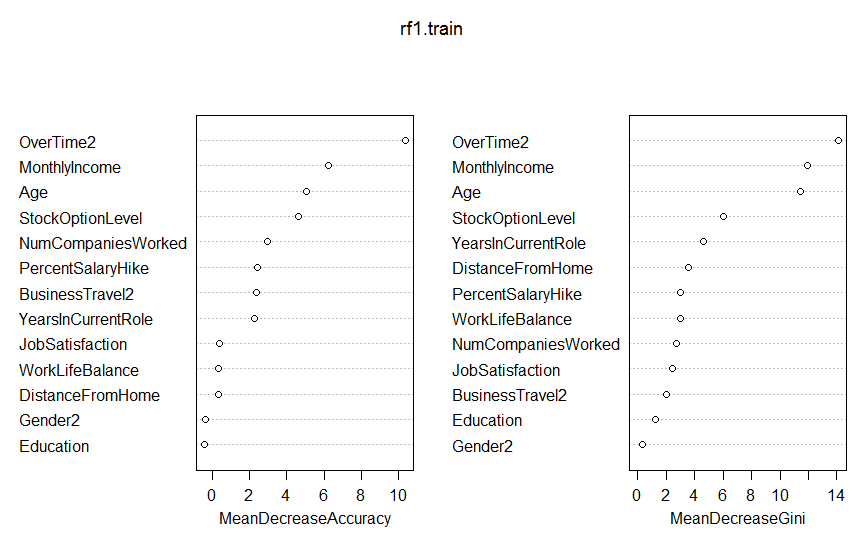
Yes 58 20

Next I predicted the results of the logistic model from the training data on my test data, and the results are encouraging. The logistic model correctly predicted 20 people to leave, and 349 people to stay out of 441. That’s an overall accuracy of 83.7%. Keep in mind however, you could get an accuracy of 84% by simply predicting everyone to stay, since there is some class imbalance with the dependent feature. The graph below is an ROC curve of the predicted output.



**Random Forest**

The second model I ran was a random forest model. Using the RandomForest package in R, I played around with some tuning parameters before settling on a final model. While I admit that I don’t fully understand how each tuning parameter interacts with each other, I feel like I was able to get a decent prediction out of my choices. The first graph show the variable importance of each independent feature. I found that Education kept showing little to no importance, so I ended up dropping that variable from the remaining models.



Once again, Overtime and MonthlyIncome show up as the most important variables. Gender and Education are the worst variables in terms of importance, I decided to drop Education but keep Gender. Below are the results of the model.

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 363 72

1 0 6

Accuracy : 0.8367

95% CI : (0.7989, 0.87)

No Information Rate : 0.8231

P-Value [Acc > NIR] : 0.2485

Kappa : 0.1206

Mcnemar's Test P-Value : <2e-16

Sensitivity : 1.00000

Specificity : 0.07692

Pos Pred Value : 0.83448

Neg Pred Value : 1.00000

Prevalence : 0.82313

Detection Rate : 0.82313

Detection Prevalence : 0.98639

Balanced Accuracy : 0.53846

'Positive' Class : 0

This model actually didn’t perform very well compared to the logistic model. I was a bit surprised by this, I thought the random forest method would provide better classification. While the accuracy is about the same at 83%, this model only predicted 6 people correctly to leave the company. The logistic model predicted 20 people correctly. Going back to the business justification, there is more value in correctly picking people to leave than correctly predicting them to stay.

**XGBoost**

The third model I ran was the XGBoost package. I tried my best to play around with tuning parameters, and settled on 30 rounds of boosting and eta=0.1. Still, I have some issues with overfitting here. I would guess this also happened in the random forest model. As you can see below, the training model data was very accurate, at about 96.5% while the test data model had an accuracy of about 84%. Still, this is the best performing model yet correctly predicting 25 people to leave the company.

Train: truth

0 1

FALSE 870 36

TRUE 0 123

Test: truth

0 1

FALSE 346 53

TRUE 17 25

One other neat package I ran with XGBoost was the boruta package, determining which variables are important and unimportant. The results are shown below.

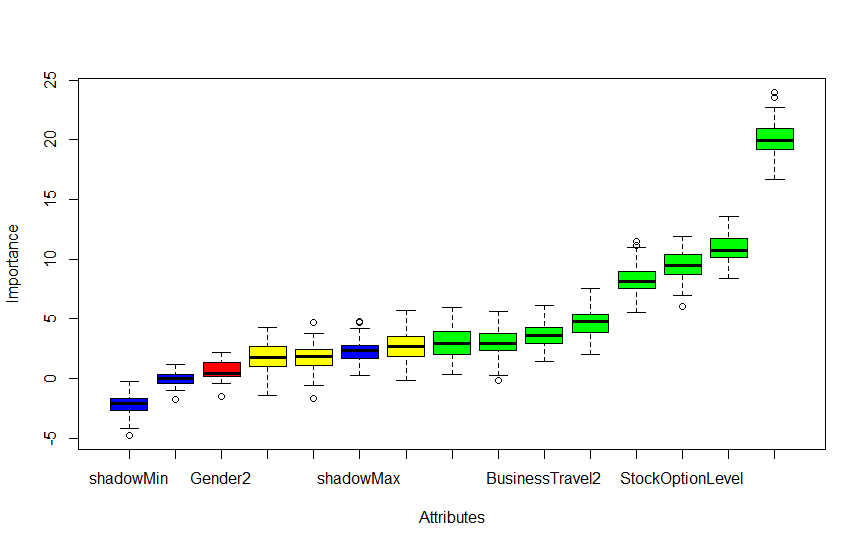
Boruta performed 99 iterations in 37.04574 secs.

8 attributes confirmed important: Age, BusinessTravel2, MonthlyIncome, NumCompaniesWorked, OverTime2 and 3

more;

1 attributes confirmed unimportant: Gender2;

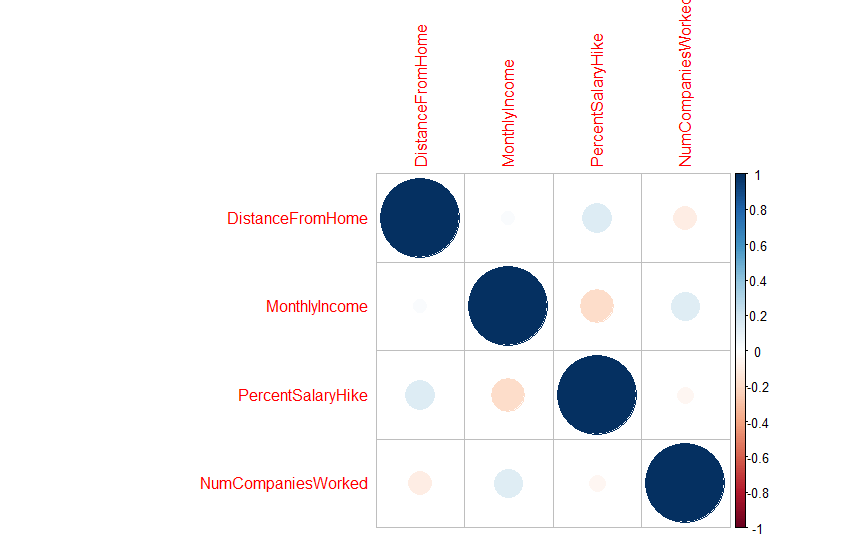
3 tentative attributes left: DistanceFromHome, JobSatisfaction, PercentSalaryHike;



Once again, Gender was determined unimportant. At this point the evidence is substantial that gender should be removed from the model. This nice graph also show how important OverTime is to the model.

**Naïve Bayesian Classification**

The last model I ran was a naïve Bayesian classification model. I was really surprised at how well this model ended up performing. It bested XGBoost in accuracy, and almost matched it in number of people correctly predicted to leave. This might be a telling sign that my xgboost model could be better tuned. I did a few things differently with the Bayes model. First, I dropped Gender, Age, and YearsInCurrentRole to try to best match the Bayes assumption that the features are not correlated with each other. Below is another graph showing the correlation matrix of the continuous features.



One area of potential trouble in this model is the fact that I used a few categorical variables in the model, despite using a Gaussian distribution for classification. I know you can use Jaccard for categorical features, but I wasn’t sure how to use both in one model. I also didn’t want to use a completely different set of variables for this model either, so I went with what I knew. Another thing I did with the Bayesian model is I set up a tuning grid to improve my prediction results. What I added into the tuning grid was the usekernal parameter, which allows for a kernel density estimate instead of a Gaussian density estimate, and I also added in a laplace smoother. I also added in pre-processing features to normalize and scale the features, which should result in additional accuracy. I learned these techniques from the UC business Analytics R Programming Guide2.

Pre-tuning results:

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 358 68

1 5 10

Accuracy : 0.8345

95% CI : (0.7964, 0.8679)

No Information Rate : 0.8231

P-Value [Acc > NIR] : 0.2904

Kappa : 0.1676

Mcnemar's Test P-Value : 3.971e-13

Sensitivity : 0.9862

Specificity : 0.1282

Pos Pred Value : 0.8404

Neg Pred Value : 0.6667

Prevalence : 0.8231

Detection Rate : 0.8118

Detection Prevalence : 0.9660

Balanced Accuracy : 0.5572

'Positive' Class : 0

Post tuning results:

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 352 56

1 11 22

Accuracy : 0.8481

95% CI : (0.8111, 0.8803)

No Information Rate : 0.8231

P-Value [Acc > NIR] : 0.09324

Kappa : 0.3255

Mcnemar's Test P-Value : 7.639e-08

Sensitivity : 0.9697

Specificity : 0.2821

Pos Pred Value : 0.8627

Neg Pred Value : 0.6667

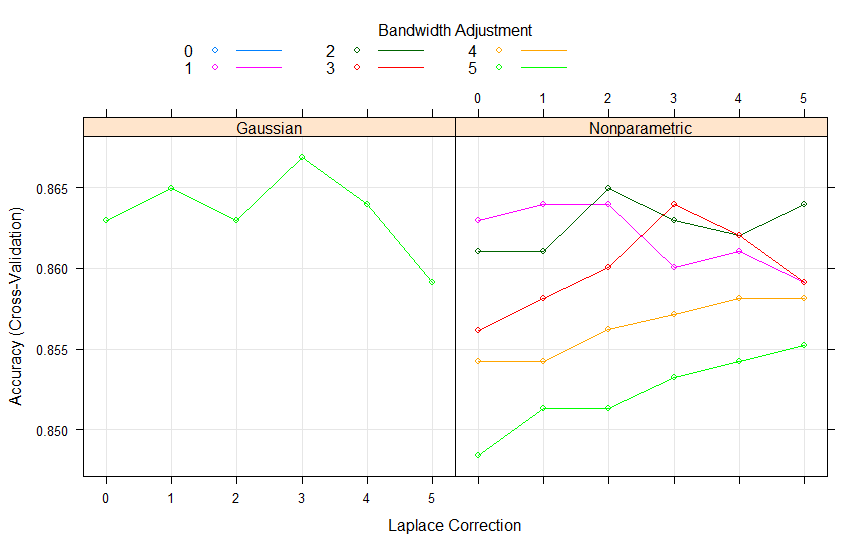
Prevalence : 0.8231

Detection Rate : 0.7982

Detection Prevalence : 0.9252

Balanced Accuracy : 0.6259

'Positive' Class : 0



So after tuning the parameters accuracy went up from 83.5% to 84.8%, and the number of correctly predicted attritions went up from 10 to 22. The graph of the bandwidth adjustment shows that the laplace smoothing didn’t have quite as nice of an effect, I think this is because I’m using categorical variables in my model and not all continuous variables.

**Conclusions**

After running 4 different machine learning models, I feel comfortable accepting the null hypothesis that salary (MonthlyIncome) is related to employee attrition. Employees with lower salaries were more likely to leave the company in every model I ran. I also found that over time hours is highly predictive of employee attrition, possibly even more important than salary when predicting employee attrition. These models can provide a lot of value to a company interested in keeping talented employees which are valuable to the long term success of any business.

**References**

1 Dataset – IBM HR Analytics Employee Attrition & Performance - 1470 rows, 35 variables

<https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset/downloads/ibm-hr-analytics-employee-attrition-performance.zip/1>

2 UC business Analytics R Programming Guide, https://uc-r.github.io/naive\_bayes