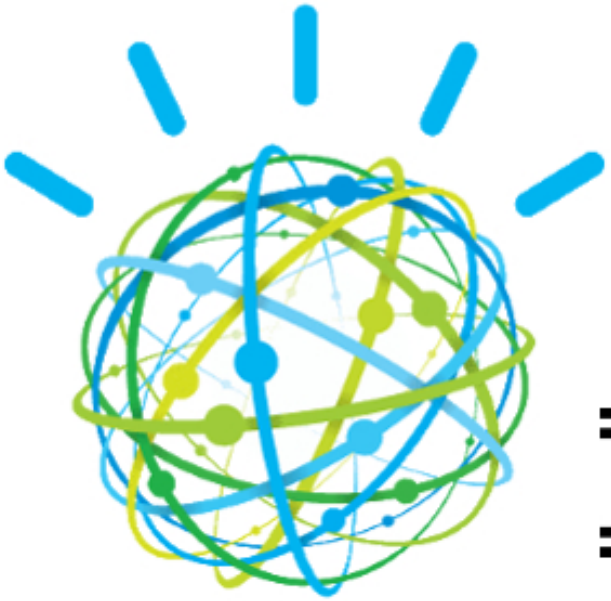


IBM HR Analytics
Employee Attrition & Performance

Bingying Shao

1. *Introduction*

The employee attrition analysis for Human Resources Analytics has caught more and more attention in the business world, especially for big company, how to use analytic methods to predict whether employees will leave or not can help the company improve the HR management and save the cost on it. Therefore, for the human resources managers, it is crucial to have a better idea of what kind of employees will tend to leave and what kind of features will influence them to leave. Most commonly, companies already engage in using analytic methods to analyzing the employee attrition such as employee surveys. But the result generated by the survey may not tell the truth, employees could lie to the company and hide their intention of leaving. Other methods will rely on the statistics results such as attrition rate to predict how many employees will leave. But this is not enough, here coming the play of the most prosperous trend of the big data era, which is the machine learning technique.

This study is aiming to use machine learning technique to:

1. Predict whether an employee will leave or not and how accurately can a machine learning algorithm predict
2. Find the underlying features and how these features will influence employees' attrition

2. *Data Description*

This study will use the IBM HR Analytics Employee Attrition & Performance data set to apply the machine learning technique to dive into and find the insights of the data. This data set is from Kaggle and it is a fictional data set created by some IBM data scientists with the purpose of mining the insights that lead to employee attrition. This data set consists of 1470 instances, which including 1 target variable "Attrition" ("Yes" or "No") and 34 predicting attributes. There is no missing value in the data set. The target attribute contains 1233 "No" cases and 237 "Yes" cases, which might lead to a data imbalance problem during the data mining process. Here is a table of the data description:

3. *Data Cleaning*

Before starting the data mining process, I conducted some data cleaning for the data set. Among all the 35 features in the data set, there are 4 features that are not useful, they are they are "EmployeeCount" (count the number of employees), "EmployeeNumber" (employee ID number), "Over18" (all the employee are over 18), and "StandardHours" (all the values are 80). Therefore, I removed these 4 features and kept 31 features including 30 predictors and 1 target variable.

During the data mining process, I first apply Min-max normalization for all the continuous features. And I also created dummy variables for the categorical features in order to apply the machine learning algorithms more smoothly on the data set.

4. ***Data Analysis***

There are 3 types of data in this data set, such as continuous, categorical and ordinal features. For the purpose of fully exploring all kinds of features and find the underlying relationship within features, I performed different data visualization techniques for each kind of feature and correlation analysis to reveal them.

Figure 1 Histograms for Continuous Features

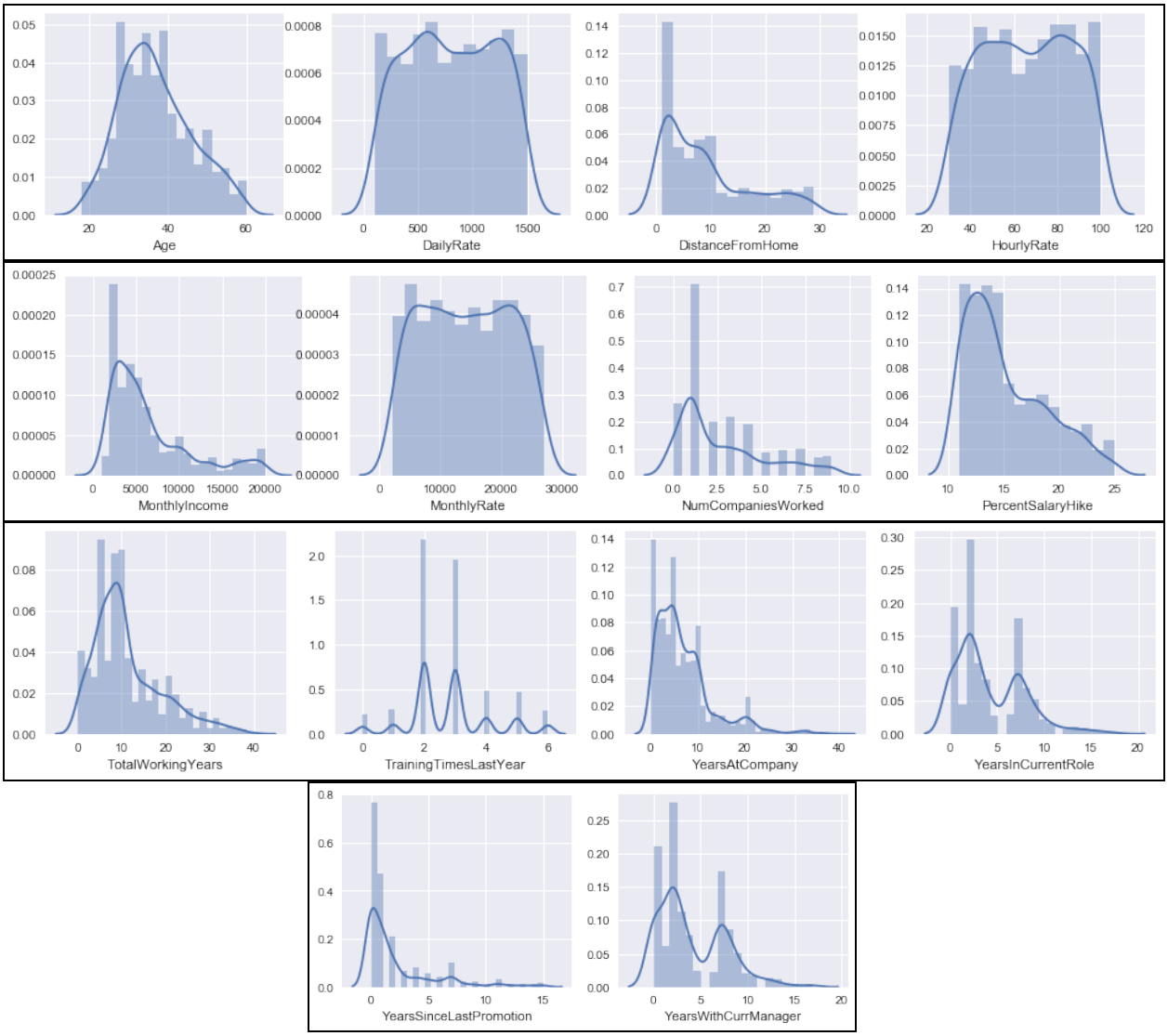


Figure 1 gives a very intuitively visualization for the distributions of all the continuous features. From the above plot, we would be able to get a rough idea of the employees from this data set. For example, what's the average age of the employees and average income of the employees.

Figure 2 Boxplots for Continuous Features

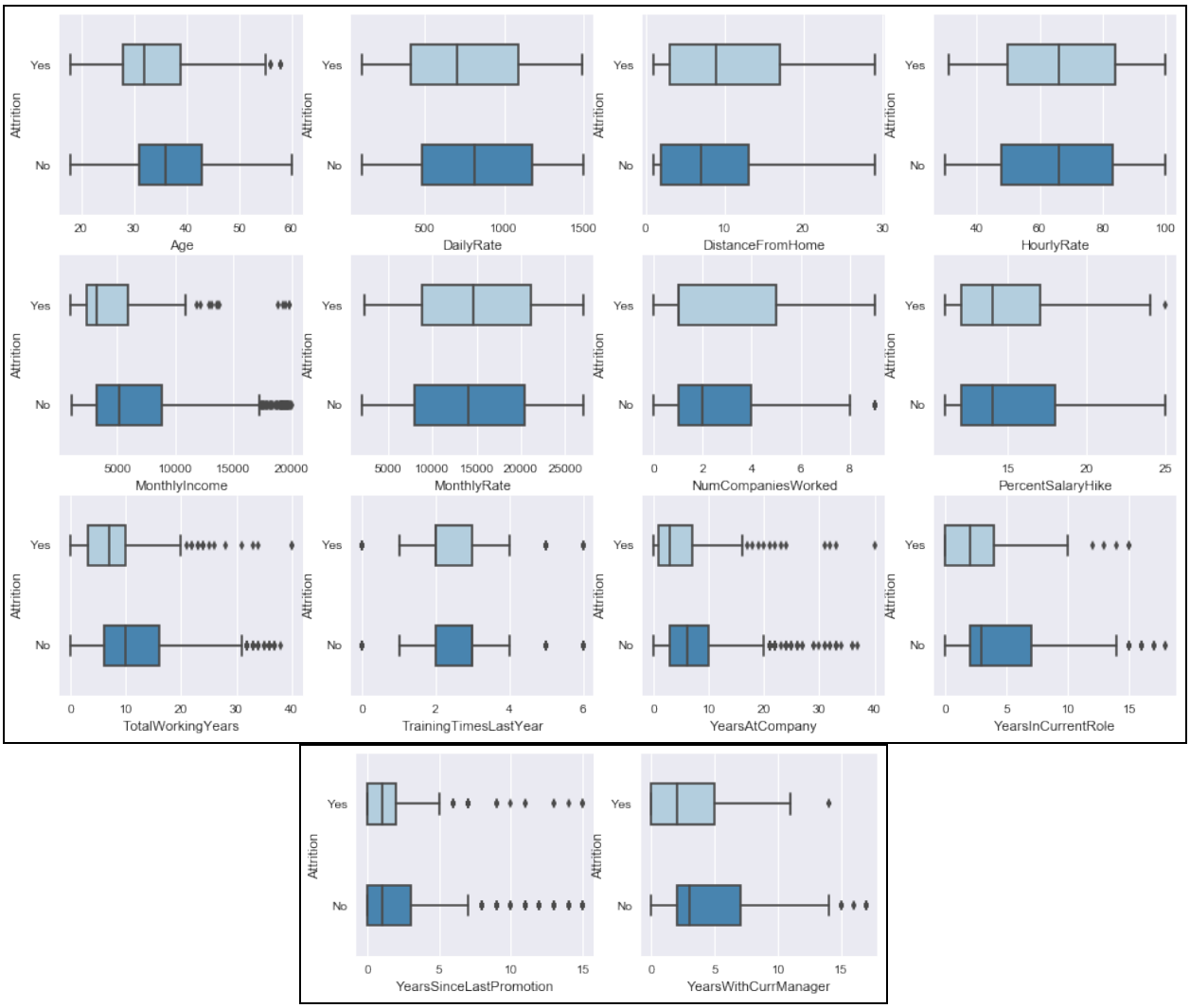
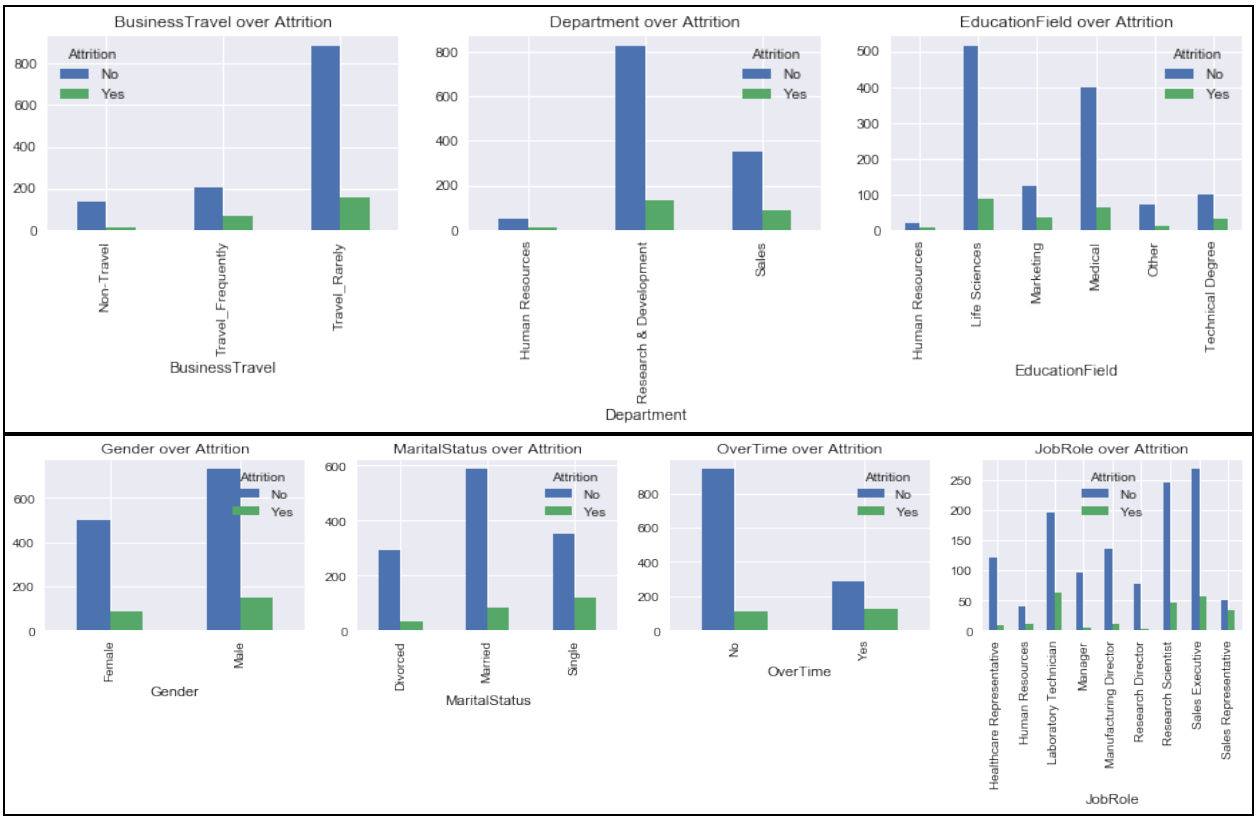


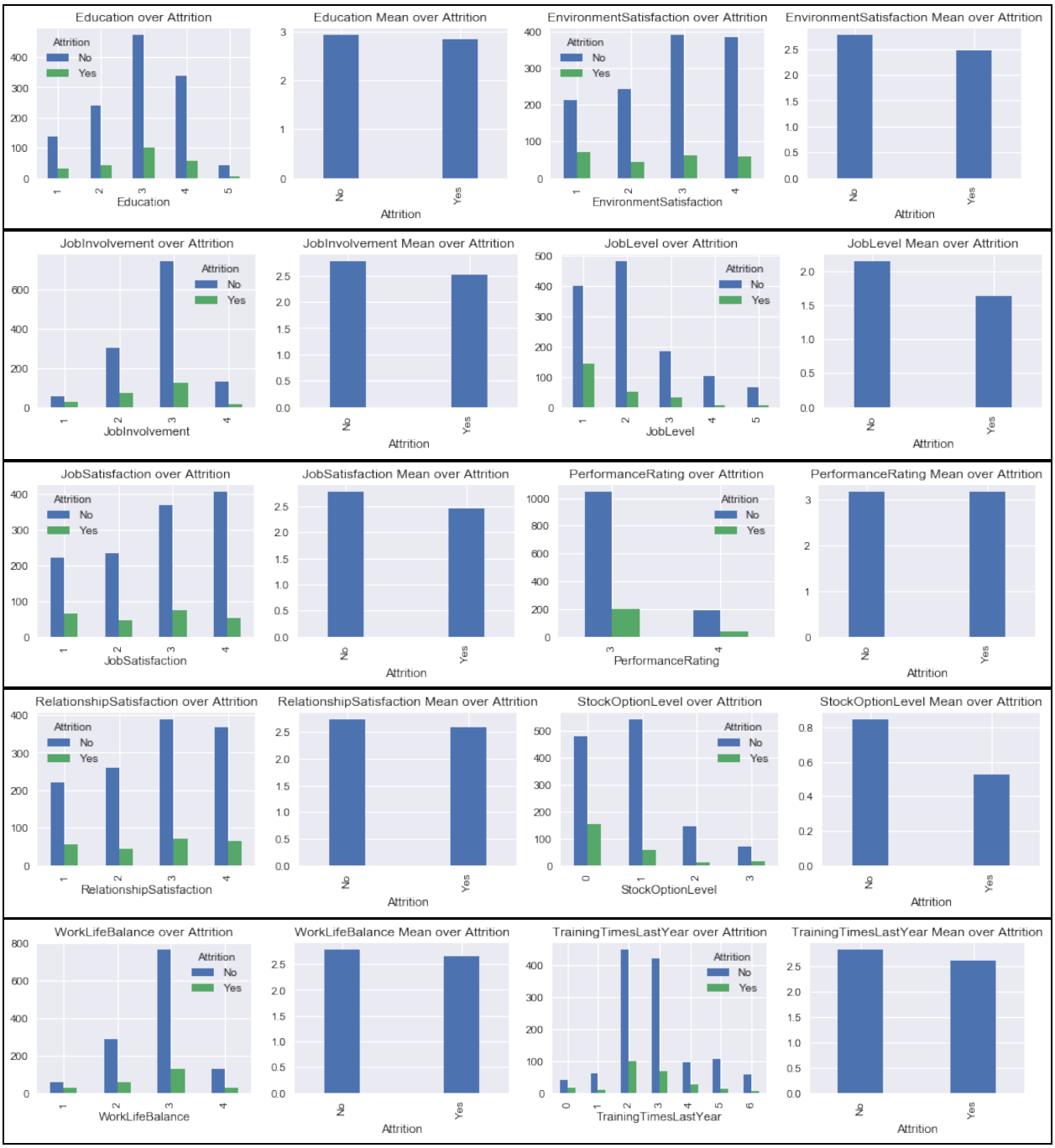
Figure 2 shows the underlying differences between those who leave the company and who stay at the company. We can see from the plots that the average age of the employees who leave is younger than the employees who choose to stay, which means that younger people are more likely to change jobs. And employees who leave mostly has a lower monthly income as well daily rate than employees who stay. For some other features such as “TotalWorkingYears”, “YearsAtCompany”, “YearsCurrentRole” and “YearsWithCurrManager”, they all shows that employees who stay and work for the company for a shorter time are more likely to leave the company, while employees who work longer for the company are less likely to leave.

Figure 3 Cross-tabulation Bar Plots for Categorical Features



For the categorical features, I applied the cross-tabulation frequency bar plots to show the differences between the employees who leave and stay within each category. From the plots, we can see that for the employees who travel a lot have a higher attrition rate, and for the employees who have overtime also have a higher attrition rate. Besides this, we can find that for different job roles, sales representative has a very high attrition rate. The “JobRole over Attrition” bar plot shows that for the sales role, there are about 40% of the sales representatives leave the company.

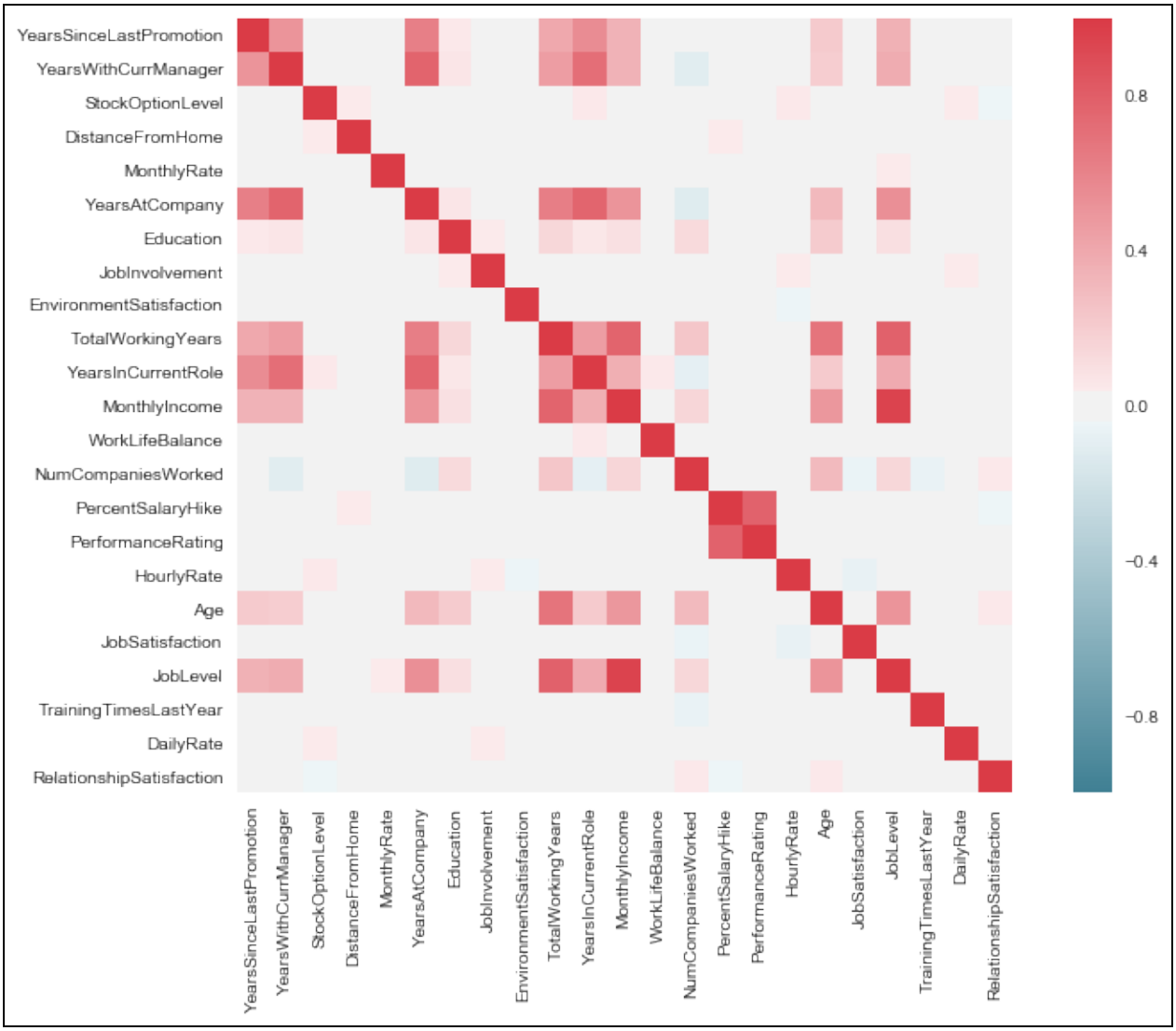
Figure 4 Cross-tabulation Bar Plots for Ordinal Features



For the ordinal features related to rating or satisfaction measure, I first applied the cross-tabulation in order to find differences and interesting patterns among them. But it doesn't show many interesting insights since the target feature is too imbalanced. Then I used the bar plot to compare every ordinal feature's mean between employees who leave and not leave, it gave me some useful information. From the plot, it is very obvious to see that employees who are more likely to leave company have lower environment satisfaction and job satisfaction,

lower job level and stock option level. But there is no significant difference in the education level.

Figure 5 Correlation Plot for Continuous Features



After taking a thorough overview on all the features, I performed the correlation analysis for all the numerical features including continuous features and ordinal features. From the above correlation plot, I found there were several features have strong positive correlation relationships. The strongest positive correlation relationship is between the “JobLevel” and “MonthlyIncome”, which means that a higher job level will have more monthly income. Besides this, “MonthlyIncome” is also strongly correlated with “Age”, “TotalWorkingYears” and features related to years stay in company.

Based on all the above exploratory analysis, we could find that there are some important features might have big influence on the attrition. Briefly speaking, employee's age, monthly income, overtime, years staying at company, working environment and job satisfaction and job role regarding to sales are very likely to influence whether an employee will leva or stay.

5. ***Experimental Results***

For proving the results from the data exploratory analysis and better understanding the underlying reasons lead to employee's attrition, the data mining techniques can contribute a lot in this section. Regarding to the purpose of this analysis is to classify if employee's attrition to be "Yes" or "No", supervised learning techniques should be applied on this data set.

Here I used 3 individual supervised learning classifiers and 3 ensemble algorithms to compare which classifier performs the best. I split the data set into train and test set, 70% for the training set and 30% for the test set. Later I performed stratified 10-fold cross validation on the training set to build the model. Then I computed the confidence interval and conducted the t-test for verifying the best model for each classifier. Finally, I fit the model on the test set to compare which algorithms performs better.

For the key performance metrics, normally the accuracy will be the choice. But since this data set has a big class imbalanced problem, the accuracy will not be a good measure. The better ways are to use "Recall" or "Sensitivity" and "Accuracy" together as the key performance metrics. The reason why to use "Recall" or "Sensitivity" as the main criteria is it is a measure of the true positive class over the actual positive class. In our case, if the classifier could get a higher recall score, it means that the classifier does a good job at predicting more true positive classes and less false negative classes, which also means the classifier does a good job at predicting the employee's attrition to be "Yes". This is the purpose of this case study.

Decision Tree Classifier:
First, I tried the grid search Decision Tree Classifier with 10-fold cross validation for the training set and I got a recall of 38.6% with a 95% confidence interval = (27.7%, 49.7%) and an accuracy of 81%, while the recall of the test set was only 24% and the accuracy also decrease to 83%. The recall of the test set was even smaller than the lower bound of the CI = 27.7% from the cross-validation process for the training set. This meant that the grid search Decision Tree Classifier had an over fitting problem. To avoid this problem, I thought a better way would be try to tune the parameters and find the best parameters before over fitting problem appears during the cross-validation process. After tuning, the best parameters of the Decision Tree Classifier I found were the parameters consists of maximum depth of 3, minimum samples leaf of 1, minimum samples split of 2, and using "Entropy" as the criterion.

To prove the parameters that I got was the best one, I run a t-test for comparing this set of parameters to another set of parameters by setting the null hypothesis for the first parameter's

cross-validation mean score equals to the second set of parameters. The p-value of the t-test was 0.009 < 0.05, which means that the null hypothesis should be rejected. Finally, I got the prediction results as below:

Table 1 Recall & Accuracy Scores for Decision Tree Classifier

DT Cross-validation		DT Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
26%	85%	27%	83%	(14.7%, 38%)

Figure 6 Decision Tree Plot

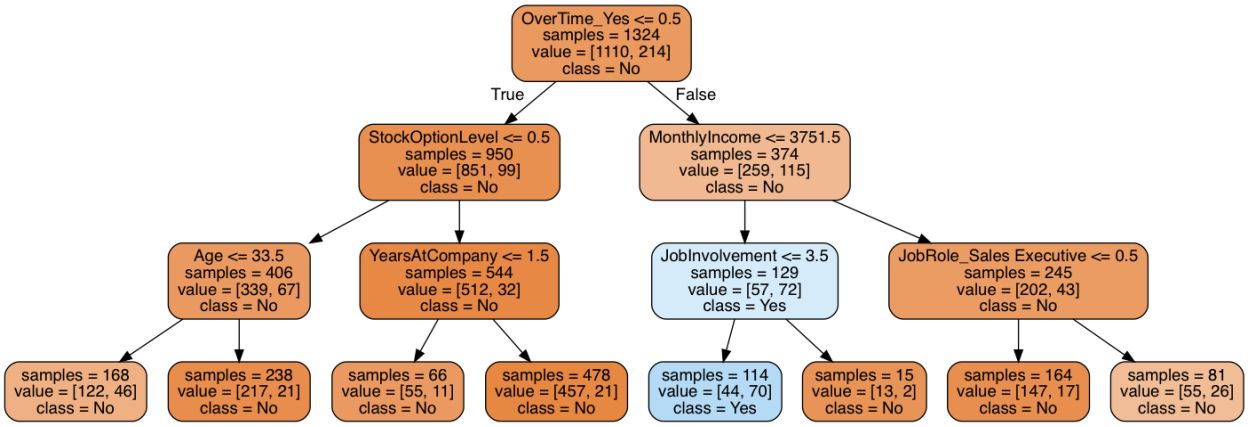
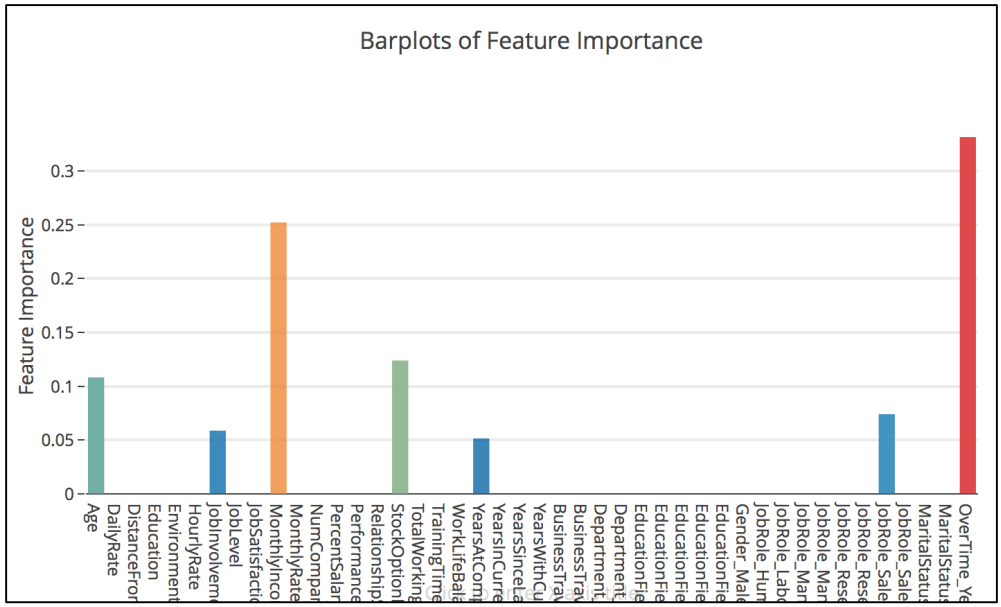


Figure 7 Barplots of Feature Importance



The above plot shows that the most important features of the Decision Tree Classifier are "OverTime", "MonthlyIncome", "StockOptionLevel", "Age" and "JobRole_Sales", which proves that the results got from the exploratory analysis.

Gaussian Naïve Bayes Classifier:
Then I tried the Gaussian Naïve Bayes Classifier and I got the results as below:

Table 2 Recall & Accuracy Scores for Naive Bayes Classifier

GNB Cross-validation		GNB Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
77%	63%	62%	63%	(67%, 86%)

The over fitting problem appeared again for the Gaussian Naïve Bayes Classifier, besides the over fitting, the accuracy scores of the Gaussian Naïve Bayes Classifier were very low. Therefore, I considered this classifier might not be a good choice for this data set, although the recall was higher than the Decision Tree Classifier.

K Nearest Neighbor Classifier:
For the last individual classifier, I tried KNN classifier, and the results I got were showed as below:

Table 3 Recall & Accuracy Scores for KNN Classifiers

KNN Cross-validation		KNN Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
0.28	0.80	0.18	0.78	(18.7%, 36.9%)

From the results, I found that the over fitting problem also appeared on KNN Classifier. The KNN Classifier gave a very low recall of 18%, which was also a little smaller than the lower bound of the confidence interval.

After trying all the three individual classifiers, the best classifier was Decision Tree Classifier. It yielded a recall score of 27% and an accuracy score of 83% after dealing with the over fitting problem. But I found this over fitting problem was a big issue for this data set. There are several reasons that can cause it. The first reason is that this data set is not big enough, it only has 1470 instances. Over fitting problem is usually vary often to happen on a small data set. The second reason might be the imbalanced issue of the data set.

After utilized the individual classifiers, I started to apply the Ensemble Classifiers to see if they could perform better on the data set. I tried three ensemble methods, Stacking, Bagging, and

Boosting. For the Bagging, I tried the Random Forest Classifiers. For Boosting, I used "AdaBoost" and "XGBoost".

Stacked KNN, DT, GNB Classifier:
For the stacking classifier, I used the three individual classifiers used as before and used "hard" voting as the criterion. The results I got are showed as below:

Table 4 Recall & Accuracy Scores for Stacked KNN, DT, GNB Classifiers

Stacking Cross-validation		Stacking Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
39%	83%	34%	80%	(26.6%, 50.3%)

The stacked classifier got a better result compared to any of the previous individual classifier. The recall and accuracy scores are a little bit lower than the cross-validation's scores for the training set. But it's not very serious.

Random Forest Bagging Classifier:
Since the Random Forest Bagging Classifier is based on the Decision Tree Classifier, I used the same method to deal with the over fitting problem that might appear to the Random Forest Classifier as well. After solving the over fitting problem, the Random Forest Classifier gave the results as below:

Table 5 Recall & Accuracy Scores for Random Forest Bagging

Random Forest Cross-validation		Random Forest Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
13%	86%	14%	85%	(6.6%, 19.6%)

The Random Forest Classifier returned a very bad recall score with only 14% for the test set.

AdaBoost:
For the boosting ensemble method, I tried two boosting algorithms. First, I used the AdaBoosting and the results are almost as good as stacking. But there are still a little over fitting for this boosted classifier.

Table 6 Recall & Accuracy Scores for AdaBoosting

AdaBoosting Cross-validation		AdaBoosting Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
45%	88%	37%	85%	(35.1%, 55.3%)

XGBoost:
Lastly, I utilized the XGBoosting and I got the best recall and accuracy scores among all the other ones. The XGBoosting yielded the highest recall score to 42%. It still has some over fitting problem.

Table 7 Recall & Accuracy Scores for XGBoost

XGBoost Cross-validation		XGBoost Test		Recall 95% CI
Recall	Accuracy	Recall	Accuracy	
51%	89%	42%	86%	(43.3%, 57.9%)

6. *Experimental Analysis*

Table 8 Test Scores for Different Classifiers

Classifier	**Test Scores**				**Over Fitting**
	Recall	**Precision**	**Accuracy**	**AUC**	
Decision Tree	27%	44%	83%	70%	N
Gaussian Naïve Bayes	72%	27%	63%	71%	Y
KNN	18%	26%	78%	54%	Y
Stacking	34%	38%	83%	62%	N
Random Forests	14%	71%	85%	74%	N
AdaBoost	37%	58%	85%	78%	Y
XGBoost	42%	61%	86%	81%	Y

After performing all the experiments above, it is easy to see that most ensemble classifiers could yield a better result than any of the individual classifier except for the Random Forest Classifier. The range of ensemble classifiers' predicted recall is from 30% to 40% and the accuracy score are all above 80%. But the problem of over fitting is an issue. As stated before, over fitting can be caused by too many features, not enough instances.

From the perspective of avoiding the over fitting problem. After handling the over fitting problem during the cross validation process, Decision Tree and Random Forest will be more reliable classifiers to predict the attrition. But compare the classifiers' performances for the test set, Decision Tree will be a better choice because of its higher recall.

For the rest of the individual classifiers, Gaussian Naïve Bayes and KNN both returned low test scores for recall or accuracy which can be seen from the Table 8, furthermore, both of these two classifiers have over fitting problems. Therefore, neither of them is a good choice.

For the rest of the ensemble algorithm, XGboosting yielded the best scores for recall (42%) and accuracy (86%). But the algorithm's computational cost is much higher than Stacking and AdaBoosting. Considering the cost for the model building, Stacking is a better choice. Moreover, Stacking Ensemble algorithm doesn't have a very big over fitting problem.

In conclusion, I think either a simple Decision Tree Classifier or the Stacking Ensemble Classifiers will be a good choice for this data set. Decision Tree Classifier will be very easy to understand and interpret but very case sensitive, if the data changes a little bit, the decision tree will change accordingly. The Stacking Ensemble Classifier is more predictable but with a higher computational cost.

This data set is an artificial data set. But in the real world, to predict the employee's attrition problem, we might not be able to get a data set with so many features. We might also end up with a data set with missing values. In this case, I will expect the classifier to be changed accordingly.

7. Conclusion

After conducting all the analysis above, and get back to the questions from the introduction. For the first question, we can say that machine learning is truly a very helpful way of predicting the employee's attrition problem. The capability of how accurately one machine learning algorithm can achieve really depends on the data set's quality such as the amount of instances and the quality of features and also the characteristics of algorithms. And in real world, when dealing with the data analysis related to business problem, we should consider more beyond the accuracy. In this domain, the accuracy isn't the best metrics, in stead, recall is better one.

For the second question, we can say that the underlying features that will influence whether employees will leave or stay could be: "OverTime", "MonthlyIncome", "StockOptionLevel", "Age" and "JobRole_Sales". Therefore, employees who are younger, travel a lot, or have lower monthly income, or with lower stock option level, or their job role are sales are more likely to leave the company according to this analysis.

Finally, I think the limitation of this case study will be the lack of instances. Is there are more instances, the over fitting problem might not be too worse. For the future work, I think it's necessary to do the repeated stratified 10-fold cross validation to reduce the high variance within the data set.