HR Attrition Analysis

Anja Wu

Executive Summary

Using an employee dataset from Kaggle, factors affecting attrition were studied, along with creating a model to predict attrition based on the employee features contained within the dataset. Several models and methods were tried to get maximum accuracy and deal with the imbalance of attrition feature. The best model was the decision tree, primarily when the oversampling method was used to randomly distribute the training data. A logistic regression was used to further understand any relationships between features and attrition. The top features which had the largest effects on attrition were: total working years, years at company, years with current manager, and percent salary hike.

1 Introduction

I am fairly new to analysis and have not worked with any HR analysis prior to this project but I will be looking at attrition in a company with specific employee features. Through the analysis of employee data using a Kaggle dataset [1], I will be trying to predict attrition of employees within a year at a company based on the data given. This dataset drew my interest given articles I have read online about the Great Resignation phenomenon that has arisen during the pandemic. The main goal is to understand what factors influence attrition the most and if there are any suggestions for the company to try to keep more of their employees. A secondary goal will be to find the best predictive model.

Two part hypothesis:

1. There are several factors which will impact attrition the most: "EnvironmentSatisfaction", "JobSatisfaction", "PercentSalaryHike", and "YearsWithCurrManager". I think the best model to give me the relationship between the features and attrition is logistic regression.
2. I think the decision tree model will lead to the best predictive model because of research that indicated it performs particularly well with imbalance outcomes.

2 Problem Definition

Problem: Attrition in general is a difficult topic to predict because so many factors need to be considered. Even if you have the perfect model with all factors considered, people's intent to leave changes on a variety of things that is based on personal preferences. This can make it difficult to extract a general rule for all people. Due to the vast variety of data in social science (such as this type of data), you would not expect the same level of correlation that you would want to see in other situations. It is enough to have a moderate correlation between any of your features and attrition to draw conclusions. It is all relative to the other features you have and how they perform. Analyzing attrition by looking at the correlations can help businesses adjust their company structure to retain as many personnel as possible because it gives more insight into the why.

Dataset: The dataset has personal information about the employee (age, gender, marital status, distance from office, etc.), work background of employee (education, number of companies worked, etc.), employee job information (job level, role, job performance, etc.), and job pay (monthly income, stock options, etc.). The dataset also had whether the employee has left in the past year or not (attrition). When looking at attrition, not all attrition is the same; it is important to look at regrettable vs non-regrettable attrition. Because sometimes attrition can be good, if it is for people who aren't

in the right role to make the impact on what they are working on. This is something that should be looked at. This particular dataset only had job performance of 3 or 4, which is fairly high. So I went ahead with the assumption that they were only looking at their higher performers and what was causing them to leave.

Data Pre-Processing: During the pre-processing of data, there were several things found:

- Several columns had only one possible option, this means they do not add any signal towards the analysis of attrition. I deleted these columns, so as to reduce dimensions.
- The columns that had features that were categorical were encoded using a simple integer encoding, ranging from 0 to n-1 categories. There are many encodings to pick from and the simple integer encoding I chose was not the best one, but I chose it for simplicity sake to allow focus on the other parts of my project.
- When doing regressions, collinearity must be checked. The correlation matrix can be seen to have several variables that have a high correlation. The best way to deal with this is to check for multicollinearity using VIF. [5] [3]
  - VIF runs a linear correlation for each feature against all other features
  - For the VIF calculations it can be seen that all values are below 5 implying that features are not highly correlated. This means that we can proceed with our regression without dropping any columns.

Dealing with imbalance: Upon further exploration of the data, it was discovered that the attrition feature is imbalance (which makes sense). There are different methods to deal with imbalance and I found a good article [2] that summarizes 10 ways. I chose the following methods:

a) Random over-sampling with imblearn: The imblearn library takes a random sample from the minority class, with replacement, until the number of samples is the same as the majority class.

b) Random under-sampling with imblearn: The imblearn library takes a randomly generated sample of the majority class, in my case, without replacement, to match the number of samples in the minority class.

c) Synthetic Minority Oversampling Technique (SMOTE): In general, SMOTE is an oversampling technique which uses k-nearest neighbours to increase the sample of minority class. It does this by taking k-nearest neighbours from a randomly selected minority point and then randomly selecting one of the neighbours. Then it creates a new synthetic point in between the minority point and its nearest neighbour. For this, I chose k =5. I tried a bunch of different k-values multiple times (5, 10, 15, 25, 30, 50, 100) and they all gave relatively the same predictive power. So for computational power and not risking overfitting, I chose the lowest k-value.

d) Under-sampling: Tomek links: In general, Tomek links uses k-Nearest Neighbours to select for undersampling. The concept is that the algorithm will pair minority and majority points on the boundary that are a small distance from each other and will convert the majority label to the minority label. As can be seen in our data, the number of points that it converted was very minor, leading to still a great divide between the majority and the minority label. So this method was not a great one to be used for this dataset.

Picking a model: Through the course we learnt k-nearest neighbours, decision trees, random forests, linear regression, logistic regression, and forward-feed neural networks. In my initial goal of this project, seeing which features have the greatest effect on attrition, a regression would be the best bet. Regressions allow the analysis of the relationship between variables. A logistic regression is used when your outcome is binary and you would like to know to what degree your features affect the outcome. I wanted to approach this problem to maximize the understanding of what factors lead to attrition, so I chose the logistic regression algorithm. Regression models allow the understanding of which features contribute the most to the outcomes you are looking for, this is done by looking at the coefficients. The accuracy of predictions was going to indicate to me how well the model predicted the importance of the features. A secondary goal of this project was to get as good of a predictive model to be able to predict whether an individual would leave the company based on the features inputted.

Through research, I found that decision trees, in particular, usually perform the best with imbalance data. However, I still wanted to look at the relationship between the features and attrition, so I went

ahead and did a logistic regression first. Then I also wanted to look at the best approach to see the difference in the performance, so I also did a decision tree. Random forest ensemble method would be a good idea to also use, however, I had found several drawbacks during my research[4] [6] including being more of a black box, and having very little interpretability. So I discounted this method, due to the fact that a decision tree was said to perform quite well for imbalanced data and is slightly more interpretable than random forest. From there, I became curious and wanted to see how the k-nearest neighbours would perform compared to the other two models, since it did not stand out in the research for best inference nor best prediction of imbalance outcomes.

3 Model

Below I will be discussing how I designed the three models, keeping in mind that for all I used 5 different sampling methods to deal with the imbalanced data: no adaptation (this creates the baseline), random over-sampling with imblearn, random under-sampling with imblearn, Synthetic Minority Oversampling Technique (SMOTE), and under-sampling using Tomek links.

3.1 Model 1: Logistic Regression

Due to the correlation that I had found during the data exploration phase, I chose to try different variations of dropping correlated columns to analyze effects on the attrition outcome. Overall, I found not much of a difference when columns were dropped. In fact, the full dataframe performed better than having any potentially correlated features removed. So for all other methods of dealing with imbalance, I used the full dataframe.

The data was separated into just training and testing sets, and the distribution of the labels was checked. After I fit the model using LogisticRegression() on the training set, I plotted the correlation coefficients for all features to get more of a visual of what features seemed to have the greatest effect. A confusion matrix was created to display the precision, recall, F1-score, and accuracy of the predictions. After this was done on all 5 methods (listed above), I created a table which held all results (highest correlation coefficient features and the metrics associated with the methods) to be able to more easily compare the different approaches.

3.2 Model 2: Decision Tree

I modified the split data function we had in assignment two to work for this dataset. The function splits the data into 3 sets: training, validation, and test. The hyperparameters will be tuned using the training and the validation set, to only be tested once on the test set once the hyperparameters have been chosen. A select model function has been created (similar to assignment 2) which fits the tree onto the training data for various depths, and criterion and returns the accuracy (using the get_acc function) for both the training and validation set.

Like before, the distribution of the attrition labels was analyzed for each method. Then the select model function was used to run different depths, using both Entropy and Gini criterion. The validation set accuracy was plotted along with the training set accuracy for those sets of depths and criterion. This was run 10 times to find an approximate best depth and criterion for each sampling method to be considered the best model (in the code, after each method, there is a description of why each was chosen). I fit the model using DecisionTreeClassifier() on the training set. Like before, a confusion matrix was created to check the accuracy, precision, recall, and F1-score for the final model chosen. Each method had a decision tree created, *pictures are attached in the upload of the assignment.*

3.3 Model 3: k-Nearest Neighbours

This model uses a similar function to split the data into training, validation, and test sets as model 2. Also like before, the distribution of the attrition labels was analyzed for each method. A heatmap is displayed to show how closely related the training set is to the validation set. For this heatmap, it is better the closer the sets are to one another to better classify the data. I used a for loop to collect and plot all validation and training set accuracy to determine which number of nearest neighbours (k) should be used. Once the ideal k was selected for each method, I fit the model using KNeighborsClassifier(n_neighbors=k) on the training set. Also like before, a confusion matrix was created to check the accuracy, precision, recall, and F1-score for the final model chosen.

4 Results and Findings

4.1 Model 1: Logistic regression

As can be seen in table 1, the method that lent the best predictions was the undersampling method because it has the best metrics (fair accuracy and has the best precision and recall), which I believe makes the ordering of the features the most likely to tell the story about which ones are the most important. In analyzing the regression coefficients, the most influential factors were:

- The more years worked, the less likely the individual was to leave
- The more years spent at company, the less likely the individual was to leave
- The more years with current manager, the less likely the individual was to leave
- The higher the individual's salary increase, the more likely they are to leave the company

It is also interesting to note that individuals who work in sales have a much higher tendency to leave.
**There is more of a breakdown for each method and coefficient in the .ipynb file.*

	Method0_Control	Method1_Over	Method2_Under	Method3_SMOTE	Method4_Tomek
1	TotalWorkingYears	TotalWorkingYears	TotalWorkingYears	TotalWorkingYears	TotalWorkingYears
2	YearsAtCompany	PercentSalaryHike	YearsAtCompany	YearsAtCompany	YearsAtCompany
3	YearsWithCurrManager	YearsAtCompany	YearsWithCurrManager	PercentSalaryHike	YearsWithCurrManager
4	Age	YearsWithCurrManager	PercentSalaryHike	YearsWithCurrManager	Age
5	NumCompaniesWorked	NumCompaniesWorked	JobRole_encoded	NumCompaniesWorked	NumCompaniesWorked
6	PercentSalaryHike	DistanceFromHome	NumCompaniesWorked	EducationField_encoded	PercentSalaryHike
7	MaritalStatus_encoded	JobRole_encoded	DistanceFromHome	JobRole_encoded	DistanceFromHome
8	JobSatisfaction	MaritalStatus_encoded	MaritalStatus_encoded	TrainingTimesLastYear	MaritalStatus_encoded
9	EnvironmentSatisfaction	PerformanceRating	PerformanceRating	PerformanceRating	JobSatisfaction
no_precision	0.827664	0.625	0.742857	0.632653	0.868812
yes_precision	0.0	0.598131	0.69863	0.619647	0.0
no_recall	1.0	0.531335	0.702703	0.589674	1.0
yes_recall	0.0	0.686327	0.73913	0.66129	0.0
no_f1	0.905707	0.574374	0.722222	0.610408	0.929801
yes_f1	0.0	0.639201	0.71831	0.639792	0.0
accuracy	0.827664	0.609459	0.72028	0.625676	0.868812

Table 1: Summary of the top 9 logistic regression coefficients for each method, along with the precision, recall, F1-score, and accuracy.

This model did a good job in showing the relationship between our features and the attrition outcome. The next model, we will be trying to increase prediction - instead of focusing on the inference.

4.2 Model 2: Decision Tree

As can be seen in table 2, all methods for a decision tree are significantly better at predicting than the logistic regression. In particular, method 1 (random oversampling) performed the best, followed closely by method 3 (SMOTE). It seems that in general the oversampling is what made the predictions better. In analyzing the decision trees created, it was very clear that the data is not sorted according to the most correlated feature to attrition, but rather the algorithm focuses on the most information gain at each split. The algorithm cares most about the final output/classification and not how it gets there, and there could be cases where the optimal tree isn't the one that learns the most at the beginning of the tree (splits at hierarchy don't necessarily signify importance). So this model would not be best at determining relationships between our features and the attrition outcome.

	Method0	Method1	Method2	Method3	Method4
no_precision	0.983636	0.967495	0.915663	0.966102	0.913043
yes_precision	0.875000	0.981293	0.885496	0.967370	0.811475
no_recall	0.974775	0.978723	0.835165	0.971039	0.785047
yes_recall	0.915888	0.971380	0.943089	0.961832	0.925234
no_f1	0.979186	0.973077	0.873563	0.968564	0.844221
yes_f1	0.894977	0.976311	0.913386	0.964593	0.864629
accuracy	0.965257	0.974797	0.897196	0.966697	0.855140

Table 2: Summary of the metrics (precision, recall, F1-score, and accuracy) for the different methods for the decision tree model.

4.3 Model 3: k-Nearest Neighbours

As can be seen in table 3, most of the methods for a k-NN are worse at predicting than the decision tree and only slightly better than logistic regression, with the exception of method 1 (random oversampling). The random oversampling method with k-NN did a relatively good job at predicting both yes and no attrition, meaning it was successful in achieving the goal of predicting our attribute labels.

	Method0	Method1	Method2	Method3	Method4
no_precision	0.819572	0.972727	0.571429	0.744086	0.543689
yes_precision	0.500000	0.825633	0.471545	0.691950	0.540541
no_recall	0.992593	0.785321	0.444444	0.634862	0.523364
yes_recall	0.032787	0.978799	0.597938	0.789753	0.560748
no_f1	0.897822	0.869036	0.500000	0.685149	0.533333
yes_f1	0.061538	0.895715	0.527273	0.737624	0.550459
accuracy	0.815710	0.883888	0.514019	0.713771	0.542056

Table 3: Summary of the metrics (precision, recall, F1-score, and accuracy) for the different methods for the k-Nearest Neighbours.

Real life application: Finding the relationship between features and attrition can be useful to the company if we are looking at features that they can influence. For things such as total working years and years at the company, not much can be done. However, it would be good for the company to further explore the "why" behind more employees with high salary hikes leaving and if there is anything that can be done to get those individuals to stay (since they are most likely the higher performing employees).

In terms of the predictive capabilities of the decision tree, it might be worthwhile to analyze out of certain employees that are instrumental to your organization, what is the likelihood of them leaving. Having a model that can predict an individual leaving, would be very useful in knowing where to focus efforts in getting high impact employees who might be close to leaving to stay.

5 Conclusions and Future Work

Overall, logistic regression should be used to find relationships between features and the class (attrition). In this case, although the accuracy of prediction was not high, we can see some relationships between years worked, years spent at the specific company, percent salary hike, and years with the current manager all impact attrition. For the best predictive model, I would recommend using the decision tree with the random oversampling method to deal with imbalance, as it gave the greatest overall accuracy (precision, recall, and F1-score).

Limitations/Future Opportunities:

- The encodings for the categorical data were done using simple encoding. I would consider exploring different encodings based on the type of data we are encoding. Pulling a pattern from "marital status" would be a different encoding than something like "job role". These should not have been encoded using 0 to n-1 due to the fact that they are not ordinal. One fix would be to potentially use one hot encoding.
- The p-values should have been analyzed more closely for the coefficients of the logistic regression to gauge the statistical significance of the features.
- One final limitation is that I do not have much HR experience and I might be missing some key fairness considerations in this domain/context.

Strengths:

- The distributions of the features were checked, which is important for checking for fairness. In general, there seemed to be no major imbalance in the features
- More than one model and method was used to see which would perform best with the dataset
- More than one run of each method and model was run to find an average (similar to bagging) for hyperparameters

References

[1] V. Choudhary. HR Analytics Case Study, 2018. URL https://www.kaggle.com/datasets/vjchoudhary7/hr-analytics-case-study.

[2] B. Kumar. 10 Techniques to deal with Imbalanced Classes in Machine Learning, 2020. URL https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/.

[3] W.-M. Lee. Statistics in Python — Collinearity and Multicollinearity, 2021. URL https://towardsdatascience.com/statistics-in-python-collinearity-and-multicollinearity-4cc4dcd82b3f.

[4] J. Singh. Random Forest: Pros and Cons, 2020. URL https://medium.datadriveninvestor.com/random-forest-pros-and-cons-c1c42fb64f04.

[5] I. Team. Variance Inflation Factor (VIF), 2021. URL https://www.investopedia.com/terms/v/variance-inflation-factor.asp.

[6] D. Trehan. Why Choose Random Forest and Not Decision Trees, 2020. URL https://towardsai.net/p/machine-learning/why-choose-random-forest-and-not-decision-trees.