Predicting Employee Turnover with Machine Learning

Prithvi Tanna ptanna@wisc.edu
Jerimiah Koll jjkoll@wisc.edu
Mike Hatfield mahatfield@wisc.edu

Abstract

With record high levels of employees quitting in 2021, human resources departments at companies have a difficult and important job of retaining employees. Using a data set from Kaggle of 15,000 employees with 10 metrics measuring employee satisfaction, performance, compensation, and work environment, we fit four models to predict employees leaving the company. We fit a logistic regression model, a K nearest neighbors model with grid search, a random forest model with grid search, and an XGBoost model with grid search to provide ourselves with a range of methods to approach this problem, each model with different strengths and weaknesses. Our best model, the random forest model with grid search, achieved an accuracy of 99%, finding that the most important factors to retaining an employee were their satisfaction, the number of hours they worked each month, and the amount of work they were given.

1. Introduction

The ongoing COVID-19 pandemic has caused many people to leave their jobs due to online working and the increased stresses due to the pandemic. This has caused increased stresses on the labor market and the economy at large.

HR departments across the nation want to prevent these losses as high employee turnover results in increased costs and lower efficiency. HR departments can prevent employee loss by giving benefits or raises to employees that are likely to quit before they leave the company.

Machine learning has been used in HR in the hiring process to choose candidates for an interview as well as predict employee satisfaction. Machine learning has also been used in predicting which employees are likely to quit and prevent employees from leaving[9]. HR can give benefits or raises to incentivize employees to stay with the company. This can be extremely effective as benefits are more likely to prevent an employee to quit before the employee makes the decision to quit.

The objective of this project is to accurately predict whether an employee of a company is going to leave based on a number of metrics available to companies. Using a data set from Kaggle with data on roughly 15,000 employees, we fit four models, compare their relative strengths and weaknesses, and assess what these models find as the most important factors in employee turnover. This information is incredibly valuable to companies as identifying the most important factors of employee turnover allows HR departments to address high turnover at its source.

2. Related Work

There has been a great deal of research done on the topic of predicting employee turnover utilizing statistical methods such as binomial classifiers or logistic regression. Recently machine learning has been implemented in predicting employee turnover with decision trees, random forests, and XGBoost models. A study done by Punnoose and Ajit [17] uses multiple approaches to determine the best way to predict turnover in a series of locations of a retail business in the US, this study follows the employees over an 18 month period with a data set very similar to ours. Punnoose and Ajit found that Extreme Gradient Boosting (XGBoost) was the most effective method providing them with 86% accuracy compared to 51% on a random forest model. While the Punnoose and Ajit study utilizes retail data, our data comes from a corporate environment with lower expected turnover. Punnoose and Ajit also provided a summary table of other research on this topic which we include in Figure 1 below ([11] [15] [10] [14] [7] [19]).

Another similar study by Khera and Divya [12] explores predicting employee attrition in Indian IT departments, using a Support Vector Machine (SVM). This study focuses more on a corporate environment with their model achieving an accuracy of 85%. The authors notes that models predicted when employees left better than when they stayed on. For this reason, we focus our model more on predicting employee leave, with an emphasis on addressing the most important factors that lead to employees leaving. This allows HR departments to focus on these factors to limit the number of employees that leave the company.

Research Authors	Problem studied	Data Mining Techniques studied	Recommend
Jantan, Hamdan and Othman [13]	Data Mining techniques for performance prediction of employees	C4.5 decision tree, Random Forest, Multilayer Perceptron(MLP) and Radial Basic Function Network	C4.5 decision tree
Nagadevara , Srinivasan and Valk [14]	Relationship of withdrawal behaviors like lateness and absenteeism, job content, tenure and demographics on employee turnover	Artificial neural networks, logistic regression, classification and regression trees (CART), classification trees (C5.0), and discriminant analysis)	Classification and regression trees (CART)
Hong, Wei and Chen [15]	Feasibility of applying the *Logit* and *Probit* models to employee voluntary Turnover predictions.	Logistic regression model (logit), probability regression model (probit)	Logistic regression model (logit)
Marjorie Laura Kane-Sellers [16]	To explore various personal, as well as work variables impacting employee voluntary turnover	Binomial logit regression	Binomial logit regression
Alao and Adeyemo [17]	Analyzing employee attrition using multiple decision tree algorithms	C4.5, C5, REPTree, CART	C5 decision tree
Saradhi and Palshikar [18]	To compare data mining techniques for predicting employee churn	Naïve Bayes, Support Vector Machines, Logistic Regression, Decision Trees and Random Forests	Support Vector Machines

Figure 1. Summary of Related Work Table

3. Proposed Method

3.1. Problem Definition

Our project involves a classification problem where we predict whether or not someone will leave their job. Therefore, all models we used are classifiers.

For hyper parameter tuning on the various models, we used grid search, which searches all possible combinations of hyper parameters in the specified parameter space, this is computationally expensive but its guaranteed to give us the best performance out of the hypothesis space. Ten fold stratified cross validation is used to obtain the best combination of hyper parameters within the grid search.

3.2. Logistic Regression

We start with logistic regression as a baseline to compare all other models with. Logistic regression predicts the probability of a label being 1 or 0 given the observed training data. Logistic regression assumes a linear relationship between the features and the sigmoid activation function, and selects coefficients for each feature that minimizes the loss function below.

$$l = -(y * log(p) + (1 - y)(log(1 - p))$$

. Outputs from the linear combination of coefficients and the feature values are mapped to probabilities using the sigmoid activation function, provided below.

$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$

3.3. K-NN Classifier with Grid Search

The second model we used is K Nearest Neighbors for classification. This model's training step consists of storing the training data. The prediction step involves finding the K closest training points using a distance function to the data point being predicted and using a majority vote to classify the data point. We used the Euclidean norm which calculates the distance between two data points by taking the square root of the sum of the squared differences in values of the features. We calculated a 3, 5, and 7 nearest neighbors model and picked between them with ten fold stratified cross validation.

3.4. Random Forest Classifier with Grid Search

The third model we used is a Random Forest Classifier. In the training stage, a random forest classifier fits a series of decision trees. Each decision tree is fit on bootstrapped training data. At each split within the decision trees, only a random subset of the features are considered for splitting. The prediction step involves using all the trained decision trees to make a prediction on the sample and using a majority vote to obtain the final prediction. We also used grid search to tune the number of estimators and the maximum depth hyper parameters.

3.5. XGBoost Classifier with Grid Search

The fourth model we used is a XGBoost Classifier. This classifier is a collection of weak decision trees. The weak learners are "boosted" by predicting the errors of the previous weak learners. Eventually these models are added together for prediction. When a new weak learner is added, gradient descent is used to minimize the loss function[8]. We used grid search to tune the number of estimators, the max depth, and the learning rate of the XGBoost algorithm.

3.6. Feature Importance Analysis

After comparing our models, we want to analyze the importance of our features to make our model more interpretable and identify the key features leading to employees leaving their jobs. We used a permutation feature importance method for this task. For each feature in the model, this method permutes the values of the feature and then fits a model with the feature in its permuted state. It then calculates the difference in accuracy between the model fit on the data with the feature not permuted and the data with the permuted feature on a test set. It intuitively follows that if the model is performing significantly worse when a feature is permuted, that specific feature is important to the model. For categorical features, the feature is permuted before one hot encoding is applied.

4. Experiments

4.1. Data set

Our data set was obtained from Kaggle [1] and contains information about 14,999 employees in a large US company

with 10 metrics. The 10 metrics are: **satisfaction_level** is a numeric variable describing the self reported level of satisfaction of the employee as a percentage from 0-100, **last_evaluation** is a numeric variable describing the employer's evaluation of the employee's performance as a percentage, **n_projects** is an integer variable of the number of projects the employee worked on, **average_monthly_hours** is a numeric variable measuring the average number of hours the employee works each month (around 160 is an average 40 hour work week), **n_years** is an integer variable measuring the number of years the employee has worked at the company, **Work_accident** is a binary classifier describing if the employee was involved in a work accident, **promotion_last_5years** is a binary classifier for if the employee received a promotion in the last 5 years, **department** is a categorical variable describing which department the employee worked in, **salary** is a categorical variable describing the relative salary level of the employee, and **left** is our binary outcome variable describing if the employee left the company or not. Figure 2 is a table of descriptive statistics of the distribution of the numerical, integer, and binary variables.

	satisfaction_level	last_evaluation	n_projects	average_monthly_hours	n_years	Work_accident	left	promotion_last_5years
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000
mean	0.612834	0.716102	3.803054	201.050337	3.498233	0.144610	0.238083	0.021268
std	0.248631	0.171169	1.232592	49.943099	1.460136	0.351719	0.425924	0.144281
min	0.090000	0.360000	2.000000	96.000000	2.000000	0.000000	0.000000	0.000000
25%	0.440000	0.560000	3.000000	156.000000	3.000000	0.000000	0.000000	0.000000
50%	0.640000	0.720000	4.000000	200.000000	3.000000	0.000000	0.000000	0.000000
75%	0.820000	0.870000	5.000000	245.000000	4.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	7.000000	310.000000	10.000000	1.000000	1.000000	1.000000

Figure 2. Numeric Variables Summary

We anticipated the most important variables impacting if an employee leaves would be their salary, satisfaction level, and how much they worked. To investigate the relationship between these variables and our outcome **left**, we produced the plots Figures 3, 4, and 5.

From these plots, it becomes clear that there are many clear breaks in the data that would likely make decision trees or KNN highly effective in predicting the accuracy of employees leaving the company. For this reason we decided to test four different models. First we estimated logistic regression as a baseline, second, K nearest neighbors, third, a random forest with grid search, and finally, XGBoost with grid search. These models should perform well with the nature of our data and the binary target variable. For all

left	0	1
salary		
high	0.933711	0.066289
low	0.703116	0.296884
medium	0.795687	0.204313

Figure 3. Table of Left Proportions by Salary level

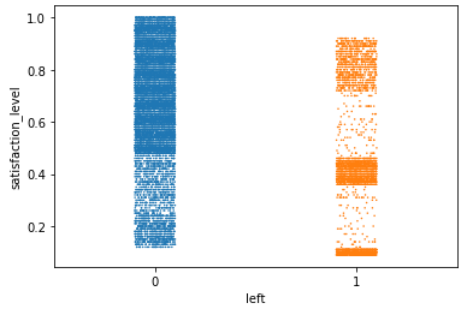

Figure 4. Plot of Satisfaction Level against Left

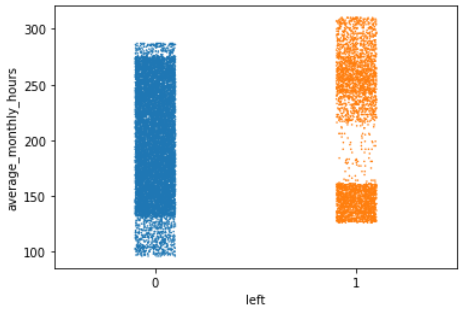

Figure 5. Plot of Average Monthly Hours against Left

models trained, we utilized every feature in the data set.

In order for our data to be used with machine learning models, some minor pre- processing was needed. Specifically, we had to one hot encode the categorical variables and normalize our categorical data.

In order to get an idea of the generalization performance of our models, we randomly designated twenty percent of our data for testing, stratified to make sure that our test set is balanced in the proportion of people that quit and stayed on.

4.2. Software

This project conducted its analysis in Jupyter Notebook [13] using Python 3 [6] and the packages Pandas [4], NumPy [3], Matplotlib [2], Seaborn [20], scikit-learn [16], XGBoost [5], and mlxtend [18].

4.3. Logistic Regression

Because logistic regression was to be used as a baseline model, we did not tune any hyper parameters for this model. However, we did make sure to drop one level from each one hot encoded categorical variable to avoid perfect colinearity which would make coefficient estimates unstable.

4.4. K Nearest Neighbors

For the K nearest neighbors model, we used grid search to tune the hyper parameter K using ten fold cross validation. K specifies the number of closest neighbors to exam-

ine from the training set when making a prediction. Specifically, we had a parameter space of K = 3, K = 5, and K = 7.

4.5. Random Forest Classifier

For our Random Forest Classifier, we tuned the number of estimators and the maximum depth of our decision trees using grid search with ten fold cross validation. We explored relatively few hyper parameters to tune because the accuracy on the test set was already very high with this combination and the high computational costs of a grid search for random forests. The hyper parameter space that we grid searched through is provided in the table below.

n estimators	max depth
100, 1000	10, 30, 50, None

Table 1. Random Forest Hyper Parameters Tuned

4.6. XGBoost Classifier

For our XGBoost Classifier, we tuned the number of estimators, the maximum depth, and learning rate of the algorithm using grid search with ten fold cross validation. A tuned model with these parameters produced very high accuracy on the test set so we did not explore more combinations of hyper parameters due to computational costs. We decided to try out lower max depth combinations compared to the random forest because we are fitting a series of weak learners. The different hyper parameter combinations considered are seen in the table below.

n estimators	max depth	lr
100, 1000	2, 5, 10, None	0.01, 0.1, 0.5

Table 2. XGBoost Hyper Parameters Tuned

5. Feature Importance

We used permutation feature importance to extract our feature importances from the best model. We repeated the permutation feature importance algorithm thirty times for each feature and reported the mean of these repeated trials as the final permutation feature importance for each variable.

6. Results and Discussion

6.1. Logistic Regression

Estimating a logistic regression as a baseline to work off of gave us a 79.43% accuracy, this actually relatively poor performance as very few people actually left overall. In fact, logistic regression performed just 3% better than predicting the majority group at all times. This is likely due to non-linear effects of our features, as can be seen in the below plot of hours worked.

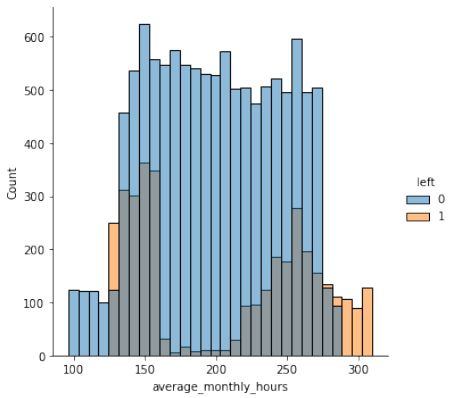

Figure 6. Hours worked, relative to if the employee left or not

Figure 6 indicates that if the employee is working relatively few or many hours then they are quite likely to leave. This is in stark contrast to those who work 160-200 hours per month who are extremely unlikely to leave the company.

There may also be interactions between features of the model. If people who have been with the company for a long time and work 400 hours a month leave at appreciably higher rates than new hires that work 400 hours a month then there will be substantial errors in the model as it doesn't take into account this interaction.

While a logistic regression model could be estimated with a full group of interactions and quadratic terms to correct for this non-linearity and interactions, it would take a substantial number of degrees of freedom and computational time for no guarantee it would work. Additionally logistic regression is only included as a baseline estimator so there's little need to solve these problems.

We also include a confusion matrix for the logistic regression to investigate the recall and precision of the model

left	N	Y
N	2131	155
Y	462	252

Table 3. Logistic Regression Confusion Matrix on Test Set (Actual Labels on Left and Predicted Labels on Top)

Accuracy	Precision	Recall	F1 Score
79.43%	61.92%	35.29%	44.96%

Table 4. Logistic Regression Metrics on Test Set

The logistic regression gave us a precision of 61.92% and a recall of 35.29%. These are quite worrying results, only 1 in 3 of people who leave are accurately predicted to

leave, indicating severe flaws in the model. This, as well as the relatively low precision combine to give an F1 score of .4496, which is less than half of the F1 scores of the next best model estimated.

In practice we would want to weight false negatives more as they are when the model states they aren't going to leave when they actually quit, this is likely to incur more costs than the case where the company tries to get them to stay, but it depends on the effectiveness of whatever intervention the company tries to do. As such we report F1 statistics.

6.2. K-NN with Grid Search

Another model we estimated was a K Nearest Neighbor estimator, iterating over 3, 5, and 7 for values of K. This gives us a baseline for algorithms to work off of, if we don't do significantly better than the K-NN model with a more complex model then the additional computational costs are unlikely to be worthwhile. Ultimately the 3-Nearest Neighbor model performed the best of the K-NN estimators with an accuracy of about 95%

left	N	Y
N	2191	95
Y	40	674

Table 5. K-NN Confusion Matrix on Test Set (Actual Labels on Left and Predicted Labels on Top)

Accuracy	Precision	Recall	F1 Score
95.50%	87.65%	94.40%	90.90%

Table 6. KNN Metrics on Test Set

These results are much better than the logistic regression, which makes some intuitive sense, a nearest neighbor model can deal with non-linear effects while a logistic regression with only linear terms can't. Given the non-linear effects of many of our features and potential interactions between features, it should be no surprise that a nearest neighbor predictor performs much better than logistic regression.

6.3. Random Forest with Grid Search

left	N	Y
N	2282	4
Y	26	688

Table 7. Random Forest Confusion Matrix on Test Set (Actual Labels on Left and Predicted Labels on Top)

The random forest with grid search was the best performing model with an accuracy of ninety nine percent on the test set. This is likely due to the fact that random forests

Accuracy	Precision	Recall	F1 Score
99%	99.42%	96.36%	97.87%

Table 8. Random Forest Metrics on Test Set

utilize decision trees, a model that can capture non-linear effects extremely well. This classifier is also a relatively large improvement over the 3-NN model, correctly identifying 80% of the errors made in the 3-NN model. This is all the more impressive when the 3-NN model had a 95% accuracy. A possible explanation for the improvement is that the random forest classifier extracts its prediction from an ensemble of varying decision trees. The use of different bootstrapped examples and only considering random subsets of features at each split allow for each decision tree to be uncorrelated from the others. This independence and majority voting means that our predictions won't be as sensitive to variations in the training set. Overall, the model has a very high score for accuracy, precision, recall, and F1 score.

The best performing hyper parameters through grid search with ten fold cross validation were a max depth of 30 and 1000 estimators. Therefore, we used this combination of hyper-parameters and achieved an accuracy of 99 percent on the test set. Although this model performs the best in all metrics, it is only marginally better than the XGBoost model. This makes sense because XGBoost is also an ensemble method that can capture the non linear relationships in the data. Additionally, while it does perform better than the 3-NN model estimated earlier, the improvements in recall are quite small relative to the improvements in precision, this may be a case of the model not having much space to improve, the precision of the 3-NN model is quite low so its easier to improve relative to the recall of the model.

6.4. XGBoost with Grid Search

left	N	Y
N	2274	12
Y	22	692

Table 9. XGBoost Confusion Matrix on Test Set (Actual Labels on Left and Predicted Labels on Top)

Accuracy	Precision	Recall	F1 Score
98.67%	98.30%	96.92%	97.60%

Table 10. XGBoost Metrics on Test Set

The XGBoost model with grid search actually performs marginally worse than the random forest model. However, it is hard to concretely conclude that the random forest model fits the data better because the difference in all per-

formance metrics is quite small. Despite these small differences, both the XGBoost and random forest both achieve near perfect results, with accuracy within 2% of 100% and F1 scores near 1. Because of this small performance differences, and XGBoost actually performing worse than the random forest, a random forest is likely the best model to use given the high computational cost of a XGBoost approach. The high performance of both random forest and Xg]GBoost seem to indicate that ensemble methods that use decision trees seem to work really well with this data set.

The hyper parameter combinations selected by grid search with ten fold cross validation were a learning rate of 0.5, no max depth, and one hundred estimators. Therefore we used this combination and achieved an accuracy of 98.67 percent on the test set.

6.5. Feature Permutation

The feature permutation to evaluate feature importance gave easily interpretable results. Employee satisfaction was the most important factor in evaluating whether an employee was going to leave or not, with a roughly 20% reduction in accuracy after employee satisfaction was permuted, by far the biggest effect of all the features permuted. Hours worked, number of years with the company, number of projects, and performance at the last employer evaluation were all relatively important with a 12-10% reduction in accuracy after the permutation was done for each of these features. All other features were quite negligible, that is after the feature was permuted, there was a very small reduction in accuracy. Interestingly salary, something that's normally associated with employee satisfaction, doesn't seem to play all that large of a role in determining if people leave their job or not. Feature permutation gave extremely stable result over all models estimated, below is the feature permutation for the random forest, the model with the highest accuracy. Because the random forest model performs extremely well, we have a high degree of confidence in these permutation feature importances.

A model could be estimated with just these 5 predictors and perform only marginally worse than the full random forest and XGBoost model. Such an approach could be useful if computational resources are limited.

6.6. Feature Correlation

In addition we want to check if these features are correlated with each other. If they are correlated it likely explains the poor performance of the logistic regression. Not only that, but highly correlated features in this data set may not hold for the broader population, which would result in poor performance outside of this data set.

We found that leaving is negatively correlated with satisfaction levels, and positively correlated with number of projects, monthly hours worked, and number of years working at the company. These results all make sense, if you aren't satisfied with your job you're more likely to leave, and an increase in your workload would make you more likely to leave as well..

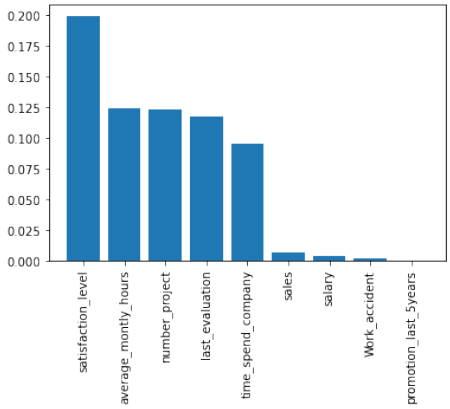

Figure 7. Relative Feature Performance, y axis is difference in performance between average permuted performance and base performance

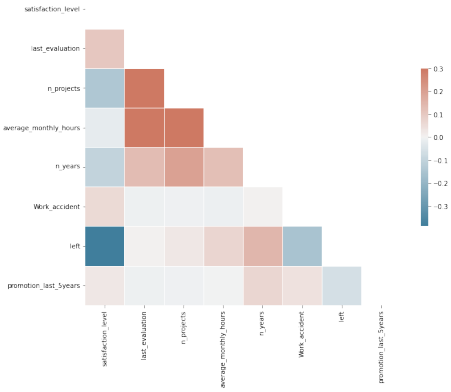

Figure 8. Correlation between Features and Outcome Variables

Of note is how employee satisfaction is also negatively correlated with number of years working with the company, number of projects, and monthly hours. This seems to indicate that workload is a major driver of employee satisfaction, and when workload gets too high employees leave. This likely explains the poor performance of the logistic regression model. Multicollinearity that exists in this correlation matrix results in inflated standard errors and unstable coefficient estimates, so its unsurprising the model performs poorly, in addition to any non-linear effects and interaction terms.

While this correlation may be part of why the logistic regression performed so poorly, there's little to worry about in our other models. While a few of the features are correlated with each other, the correlation is quite small and is unlikely to cause any significant problems with extrapolation.

7. Conclusions

Our findings were that a random forest was the best model to use to predict whether or not people leave the company or not. With a 99% accuracy, its unlikely that any additional improvements can be made to increase accuracy.

We found that the biggest factor in whether or not employees would leave or not is employee satisfaction, followed by number of hours worked, number of projects, time spent at the company, and performance on the last employer interview. Additionally, employee satisfaction, by far the most important feature in all models, was significantly correlated with hours worked and number of projects. This seems to indicate that employee satisfaction is driven by workload, the more people have to do at their job, the less satisfied they are, and the more likely they are to leave.

This company could reduce workload in one of two ways, reducing monthly hours worked, or reducing how many projects people are working on at a time. The company would likely have to hire new employees to make up for the reduced productivity of employees.

Another potential solution to high employee turnover would have to do with employees who have been with the company for many years. Employees who have been with the company for 3-5 years are very likely to quit. Obviously, the employer is unlikely to be able to sustain an environment where the entire staff has only been working there for two years so something would need to be done to fix the problem of experienced workers leaving. Workload reduction would seem to benefit employees who have been with the company longer as more experienced employees tend to work more hours and have more projects.

Care should be taken when extrapolating this model however, monthly hours worked are much higher here than an average company, while it may still work for those working a standard 40 hour work week, this relation may fall apart with part time employees. Additionally company culture, type of job, and non-wage benefits can play a significant role in determining if an employee leaves or not, and the model used doesn't take these factors into account. The Ajit and Punnoose study done gives significantly different results because it focuses on retail establishments which tend to have higher turnover and employees come from a completely different cohort than the corporate data investigated here.

Additional research and investigation could be done in creating counterfactuals, taking a baseline example and then shifting values of features to switch the predicted outcome. Such an approach would essentially find what would need to change for an employee to not leave, given that they currently plan to leave. We tried implementing counterfactuals, but there were problems implementing counterfactuals with the categorical data in the model. Some of these categorical data could be ignored, for example it doesn't make much sense to transfer someone to a different job type to affect how likely they are to leave. Even some of the continuous variables wouldn't be included in a counterfactual. How long the employee has been with the company is important to whether or not they leave, but it's not something that the company can change to the employee to get them to stay.

8. Acknowledgements

We would like to thank Professor Raschka for teaching us how to use and interpret the machine learning models we used in the project.

9. Contributions

Mike was responsible for the data cleaning, prepossessing, and data investigation. In writing the paper, Mike also was responsible for the Introduction, Related Work, Data set, and Software sections.

Jerimiah produced additional graphs and data investigation. In writing the paper Jerimiah focused on the Results and Discussion, as well as the Conclusion.

Prithvi trained the models, performed the experimentation with the hyper parameter tuning, and extracted permutation feature importances. In the paper, Prithvi contributed to the proposed method, experiments, and Results and Discussion section.

Everyone was responsible for model choice

References

[1] HR dataset.

[2] Matplotlib — Visualization with Python.

[3] NumPy.

[4] pandas - Python Data Analysis Library.

[5] Python API Reference — xgboost 1.5.1 documentation.

[6] Python Release Python 3.10.0.

[7] D. Alao and A. B. Adeyemo. Analyzing employee attrition using decision tree algorithms. *Computing, Information Systems, Development Informatics and Allied Research Journal*, 4, 2013.

[8] J. Brownlee. A Gentle Introduction to XGBoost for Applied Machine Learning. *Machine Learning Mastery*, Aug. 2016.

[9] E. Choice. The Impact Of Machine Learning In HR. *Hppy*, Feb. 2018.

[10] W. C. Hong, S. Y. Wei, and Y. F. Chen. A comparative test of two employee turnover prediction models. *International Journal of Management*, 24(4):808, 2007.

[11] H. Jantan, A. R. Hamdan, and Z. A. Othman. Towards Applying Data Mining Techniques for Talent Managements. *International Conference on Computer Engineering and Applications*, 2, 2011.

[12] S. N. Khera and Divya. Predictive Modelling of Employee Turnover in Indian IT Industry Using Machine Learning Techniques. *Vision*, 23(1):12–21, 2019.

[13] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing. Jupyter notebooks – a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87 – 90. IOS Press, 2016.

[14] L. K. Marjorie. Predictive Models of Employee Voluntary Turnover in a North American Professional Sales Force using Data-Mining Analysis. *Texas, A&M University College of Education*, 2007.

[15] V. Nagadevara, V. Srinivasan, and R. Valk. Establishing a link between employee turnover and withdrawl behaviours: Application of data mining techniques. *Research and Practice in Human Resource Management*, 16(2):81–97, 2008.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] R. Punnoose and P. Ajit. Prediction of Employee Turnover in Organizations using Machine Learning Algorithms. *International Journal of Advanced Research in Artificial Intelligence*, 5(9):22–26, 2016.

[18] S. Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *The Journal of Open Source Software*, 3(24), Apr. 2018.

[19] V. V. Saradhi and G. K. Palshikar. Employee churn prediction. *Expert Systems with Applications*, 38(3):1999–2006, 2011.

[20] M. L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.