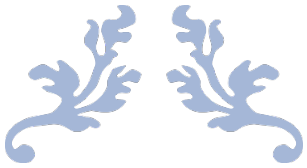

EMPLOYEE CHURN MODELLING AND PREDICTION

Data Mining Project (Fall 2022)

ABSTRACT

Retaining current employees is more difficult for the HR team than recruiting new ones. Any business that loses one of its valuable employees suffers a loss in terms of productivity, time, money, and other factors. This loss could be reduced if HR could able to foresee future employees who were considering leaving their positions; as a result, we looked into ways to address the employee turnover issue from a machine learning perspective. When the time comes to lay off workers as part of organizational changes, the corporation can use churn modeling to make a rational decision rather than randomly selecting layoff candidates. Using supervised and classification-based methods like Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM) and Kernel SVM, Naïve Bayes, Decision Tree, Random Forest, Bagging classifier, AdaBoost classifier, XGBoost Classifier, Artificial Neural Network we have created machine learning models. The models are trained with the Employee Attrition dataset retrieved from https://www.kaggle.com/datasets/whenamancodes/hr-employee-attrition and later fine-tuned to boost the performance of the models.

Keywords: Employee Churn, Machine learning, Artificial intelligence, Data mining Data analytics, Data visualization, Feature selection, Model stability, Employee turnover, Logistic regression, KNN, SVM, Kernel SVM, Naïve Bayes, Decision Tree, Random Forest, Bagging classifier, AdaBoost classifier, XGBoost Classifier, Artificial Neural Network, Human resources management, PCA, K-fold.

INTRODUCTION

Employee churn is defined as a decision made voluntarily by an employee to leave their position or retire, necessitating the hiring of a new applicant. People frequently leave their jobs for a variety of reasons, including a feeling of lacking coaching and feedback, a lack of growth, commute time, an unsatisfactory pay scale, a sense of devaluation, work stress, a lack of balance between work and life, a lack of trust in their line manager, etc. Employees are always considered valuable assets of the company [1]. In 2018, LinkedIn Talent Solutions issued a survey that indicated the IT industry is on the top with the greatest employee turnover of 13.2% [2].

A freshly recruited staff is required to be sent for varied training so that he/she can do his/her duties in a decent fashion and sending the personnel to the training cost more money to the organization. The newly hired personnel must acclimate to the new teammates and workplace before they can perform. Every time there is a search for a replacement resource, the HR team must go through all those stages. Therefore, the HR team concentrates on keeping their current employees to minimize that circumstance.

The HR team and managers have tried a number of different approaches to keep their best employees. The HR team works on creating a correct compensation structure across organizational levels, ensuring fair pay and equality, providing superior benefits packages like vesting, stock options, and cash bonuses, etc., as part of its compensation and benefits plans [3]. Company incentives such as longer PTO and flex time have been effective techniques for retaining personnel [4]. Establishing a staff/employee appreciation program can make employees feel their work is being recognized and consequently it may motivate them to stay with the organization for the longer-term [5].

We are optimistic that HR will be assisted in some manner by the employee churn forecast provided by our created models in order to take the necessary steps to retain their workforce.

- Can we predict if an employee will leave the company?
- Can existing machine learning models be used to predict employee attrition?

To solve the employee churn problem, we designed models using Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM) and Kernel SVM, Naïve Bayes, Decision Tree, Random Forest, Bagging classifier, AdaBoost classifier, XGBoost Classifier, Artificial Neural Network to predict whether an employee will churn or not. An exploratory data analysis was performed to explore and clean the Employee Attrition dataset that we retrieved from www.kaagle.com.

Data Set

Data source: https://www.kaggle.com/datasets/whenamancodes/hr-employee-attrition

There are 1470 records in the data, each with 35 attributes, including 34 feature attributes and one target attribute. Attrition is the target attribute and has the values "Yes" and "No," based on other dependent attributes. The following list of qualitative and quantitative traits includes both dependent and independent attributes.

Quantitative	**Qualitative**
Age	Attrition
Daily Rate	BusinessTravel
DistanceFromHome	Department
Education	EducationField
EmployeeCount	Gender
EmployeeNumber	JobRole
EnvironmentSatisfaction	MaritalStatus
HourlyRate	Over18
JobInvolvement	OverTime
JobLevel	
JobSatisfaction	
MonthlyIncome	
MonthlyRate	
NumCompaniesWorked	
PercentSalaryHike	
PerformanceRating	
RelationshipSatisfaction	
StandardHours	
StockOptionLevel	
TotalWorkingYears	
TrainingTimesLastYear	
WorkLifeBalance	
YearsAtCompany	
YearsInCurrentRole	
YearsSinceLastPromotion	
YearsWithCurrManager	

Methodology

Data mining techniques are beneficial for predicting employee turnover because they enable businesses to utilize data and statistical models to automatically learn the patterns and relationships in their data and forecast which employees are most likely to leave the company.

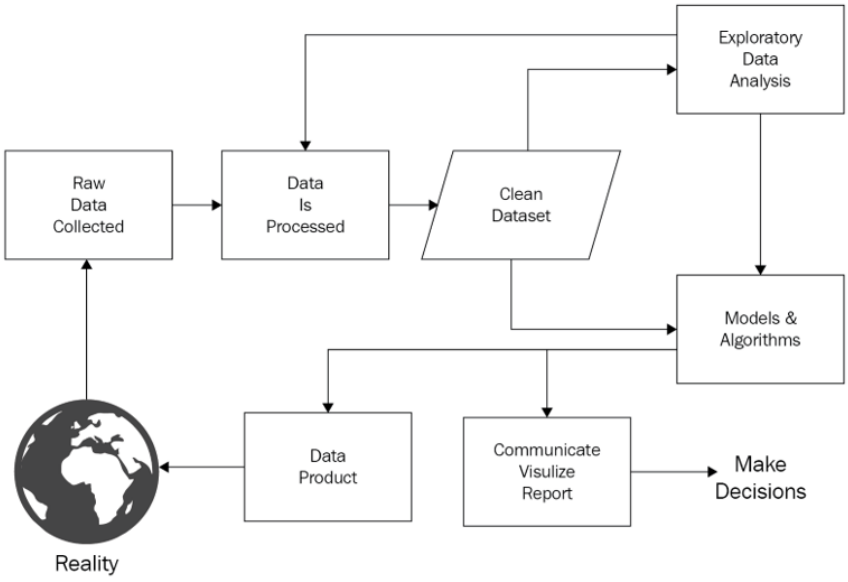

Fig1. Data Flow

A company provides the raw data, which is subsequently preprocessed and cleansed for use in exploratory data analysis utilizing data processing techniques. The cleansed data is fed into the models and algorithms as input. The analysis is then represented graphically using pie charts, box plots, and bar graphs.

Exploratory Data Analysis:

EDA is a crucial tool for gaining insights into the data, spotting potential difficulties or problems, and acquiring a deeper understanding of the data. It is often the first step in the data analysis process.

1. **Schema analysis:**

To comprehend the structure and data properties, schema analysis is used. We checked the shape and memory usage of the data set as our first step in EDA.

```
In [8]: #To get the column wise memory consumption and usage for the dataframe
        eda_df.memory_usage()

Out[8]: Index                       128
        Age                       11760
        Attrition                 11760
        BusinessTravel            11760
        DailyRate                 11760
        Department                11760
        DistanceFromHome          11760
        Education                 11760
        EducationField            11760
        EmployeeCount             11760
        EmployeeNumber            11760
        EnvironmentSatisfaction   11760
        Gender                    11760
        HourlyRate                11760
        JobInvolvement            11760
        JobLevel                  11760
        JobRole                   11760
        JobSatisfaction           11760
        MaritalStatus             11760
        MonthlyIncome             11760
        MonthlyRate               11760
```

Fig2. Memory usage of the data set for some attributes

2. Uni-variate Analysis:

The most fundamental method of statistical data analysis is known as univariate analysis. We utilized box plots for numerical attributes to comprehend the statistical distribution when there was only one variable in the data, and there were no causes or effects to be considered. The box plots can be used to spot outliers in data. Categorical data are analyzed using bar graphs.

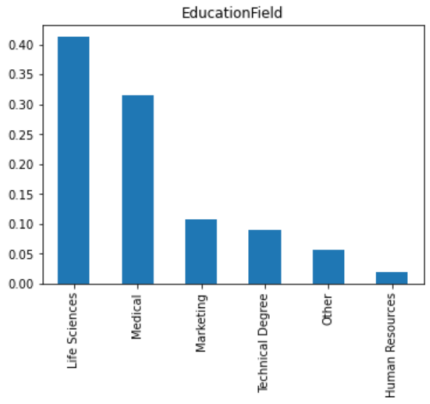

Fig3.Bar graph representation of Uni-Variate

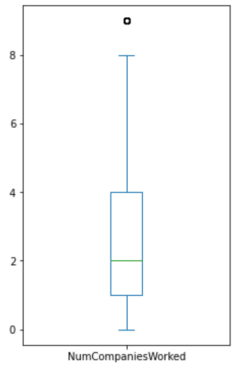

Fig4. Box plot with outlier

3. Bi-Variate Analysis:

Compared to univariate analysis, bivariate analysis is a little more analytical. Bivariate analysis is the appropriate form of analytic approach when the data set contains two variables and researchers want to compare the two data sets. The target attribute is examined utilizing other dependent qualities to understand their relationship.

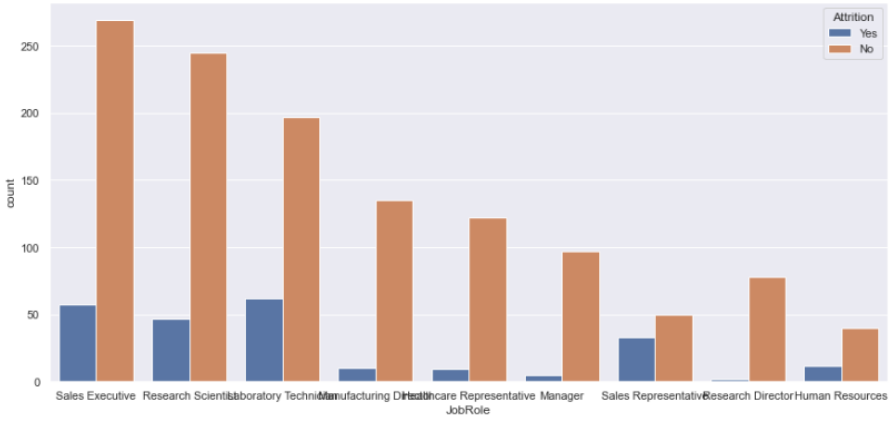

Fig5. Bi-variate representation of features and target attribute

4. Multi-variate analysis:

When there are more than two variables in the data set, a more complicated statistical analysis technique, multivariate analysis is utilized.

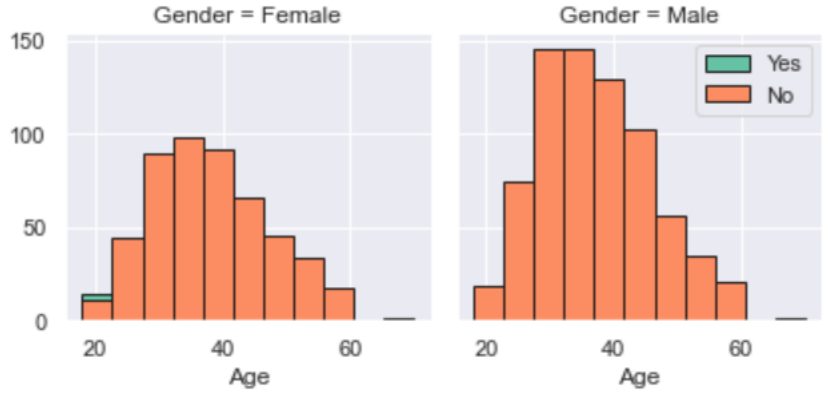

Fig6. Multi-variate representation of Gender and Age Attribute

5. Correlation:

A non-parametric test called Spearman rank correlation is used to gauge how closely two variables are related. Due to the non-linear nature of our dataset, we are using the Spearman approach.

```
In [26]: eda_df.corr(method="spearman")
```

5]:

	Age	DailyRate	DistanceFromHome	Education
Age	1.000000	0.006100	-0.015531	0.203437
DailyRate	0.006100	1.000000	-0.004223	-0.014752
DistanceFromHome	-0.015531	-0.004223	1.000000	0.015708
Education	0.203437	-0.014752	0.015708	1.000000

Fig7. Spearman Matrix for the

Data Preprocessing

1. **Feature Selection:** With just relevant data and eliminating irrelevant data, feature selection is a technique for lowering the input variable for your model. The attributes Age (because everyone is above 18) and EmployeeID (as this feature has unique values) were found to have no bearing on the analysis in the EDA; thus, we excluded them.

2. **Missing data:** We must replace the missing values because they trigger outliers. Therefore, missing data in categorical attributes are replaced with a constant value or the most popular category, and missing data in numerical attributes are replaced with the mean.

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values = np.nan, strategy = 'mean')
pre_processing_df.Age = imputer.fit_transform(pre_processing_df['Age'].values.reshape(-1,1))[:,0]
pre_processing_df.DailyRate = imputer.fit_transform(pre_processing_df['DailyRate'].values.reshape(-1,1))[:,0]
pre_processing_df.MonthlyIncome = imputer.fit_transform(pre_processing_df['MonthlyIncome'].values.reshape(-1,1))[:,0]
pre_processing_df.PercentSalaryHike = imputer.fit_transform(pre_processing_df['PercentSalaryHike'].values.reshape(-1,1))[:,0]
```

```
In [33]: pre_processing_df.isnull().sum()

Out[33]: Age                  0
         BusinessTravel       0
         DailyRate            0
         Department           0
```

Fig8. Missing value replacement

3. **Features & Target:** Features are distinct independent variables that serve as the analysis's input. The attribute with values depending on the feature attributes is considered the Target. The features and target columns were divided into X and y, respectively. Y is the target column, and X = "x1, x2, x3,.... " are feature vectors/columns.

```
In [37]: print(X)
         print(len(X))

[[36.98159509202454 'Travel_Rarely' 1102.0 ... 4 0 5]
 [49.0 'Travel_Frequently' 803.0368349249659 ... 7 1 7]
 [37.0 'Travel_Rarely' 1373.0 ... 0 0 0]
 ...
 [27.0 'Travel_Rarely' 155.0 ... 2 0 3]
 [49.0 'Travel_Frequently' 1023.0 ... 6 0 8]
 [34.0 'Travel_Rarely' 628.0 ... 3 1 2]]
1470

In [38]: print(y, len(y))

['Yes' 'No' 'Yes' ... 'No' 'No' 'No'] 1470
```

Fig9. Dividing Features and Target Attributes

4. **Encoding categorical data:** In order to provide the models with the data with transformed categorical values and to help them make better predictions, categorical data must first be converted into integer format, a process known as encoding. One-hot encoding divides the column into multiple columns to transform the categorical data into numerical data. Depending on which column carries the value, the numbers are changed to 1s or 0s.

5. **Splitting the dataset:** The training data set is used to train and create models in a simple two-part data split. Training sets are frequently used to estimate various parameters or to evaluate the effectiveness of various models. After the training is complete, the testing data set is used. To ensure that the final model operates properly, the training and test sets of data are compared. We used 80% of the data in our data set for training and 20% for testing.

6. **Feature scaling:** The process of feature scaling involves converting the values of numerical variables in a dataset to a standard scale, like 0 to 1 or -1 to 1. As many algorithms utilize a distance-based metric to assess the similarity between data points, and the magnitude of the variables might affect the distance calculations, this is frequently required for algorithms to operate properly. For the column MonthlyIncome, the values range from 1000 to 90,000. If we examine the TotalWorkingYears property, the numbers range from 0 to 10. Therefore, we have performed the feature scaling to put them within the same range and to remove any bias.

Dimensionality Reduction

A model's complexity can be decreased through dimensionality reduction, which also prevents overfitting. Feature selection and feature extraction are the two fundamental divisions of dimensionality reduction. In contrast to feature extraction, which uses data from the feature set to create a new feature subspace, feature selection allows us to choose a subset of the original features. An unsupervised linear transformation method known as Principal Component Analysis (PCA) is frequently utilized in a variety of domains, most notably for feature extraction and dimensionality reduction. Based on the connection between features, PCA aids in the discovery of patterns in data. Basically, PCA projects high-dimensional data onto a new subspace with the same dimensions as the original subspace to identify the directions of maximum variance.

We have reduced the dimensions to two components by using PCA as a dimension reduction approach. The most variance is captured by these two elements. The data is not completely linearly separable, as shown in the plot below.

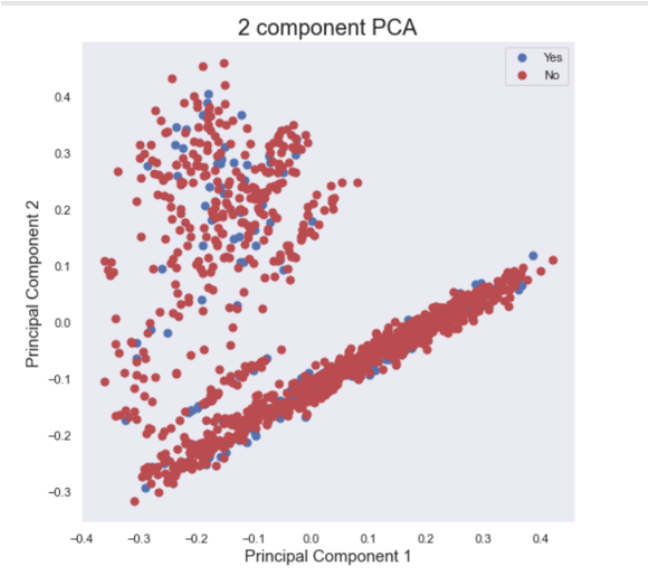

Fig10. Dividing Features and Target Attributes using PCA

Comparative Analysis of various Classification Models

1. **Logistic Regression (LR)**: As first proposed by Cox in 1958 [6], logistic regression is a conventional classification approach employing linear discriminants. The main output is the probability that the specified input point belongs to a particular class. The model establishes a linear border dividing the input space into two sections based on the probability value. One of the most popular classifiers is logistic regression since it is simple to use and effective with linearly separable classes [7].

```
#Performing K-Fold cross-validation to evaluate the model better. Here, k = 10
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = logistic_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.89830508 0.86440678 0.88983051 0.88135593 0.83050847 0.89830508
 0.90598291 0.90598291 0.88888889 0.87179487]
Accuracy is 88.35361437056352
Standard Deviation is 2.1983415274977895
```

Fig11. Accuracy for Logistic Regression

2. **Decision Tree (DT):** A decision tree is a supervised method which builds classification or regression models in a tree-like structure. It is an established method that was first published in 1963 by Morgan and Sonquist [8]. The criterion parameter chosen when training our model is "Entropy"

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = decisiontree_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.80508475 0.72881356 0.75423729 0.76271186 0.81355932 0.8559322
 0.85470085 0.82905983 0.74358974 0.76923077]
Accuracy is 79.16920179632044
Standard Deviation is 4.377848275612897
```

Fig12. Accuracy for Decision Tree

3. **Random Forests (RF):** Random forests use an ensemble technique that improves the fundamental decision tree structure by combining a set of weak learners with building a stronger learner [9]**.** Given their great accuracy and stability, random forests are excellent candidates for predicting staff attrition. Random forests can capture complex non-linear relationships between the predictors and the outcome by combining the predictions of many individual decision trees, and they can produce predictions that are more accurate than those made by a single decision tree. The number of estimators chosen here is 100 and the criterion parameter has the value as "Entropy."

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = randomforest_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.86440678 0.8559322  0.88135593 0.86440678 0.86440678 0.87288136
 0.86324786 0.88034188 0.86324786 0.84615385]
Accuracy is 86.56381283499928
Standard Deviation is 1.0027620115213363
```

Fig13. Accuracy for Random Forest

4. **Artificial Neural Networks (ANN):** The human brain's structure and operation inspired the development of artificial neural networks (ANNs), a form of a machine learning algorithm. Neurons, the numerous interconnected processing units that make up ANNs, are arranged into layers. We used 3 hidden layers with 16, 12 and 7 neurons respectively. The output layer has 1 neuron as we are predicting the Binary outcome. The activation functions for the hidden and output layers are ‘relu’ and ‘sigmoid’ respectively. We fit our classifier with a batch size of 32, and 16 epochs.

```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[207  29]
 [ 38  20]]

0.7721088435374149
```

Fig14. Accuracy for ANN

5. **Support Vector Machine (SVM):** While the Support Vector Machine is mostly used for classification, it can also be utilized for regression in supervised learning algorithms. The key concept is that the algorithm searches for the best hyperplane that may be used to categorize new data points based on the labeled data (training data). The hyperplane is a straightforward line in two dimensions.

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = svm_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.89830508 0.8559322  0.88983051 0.88983051 0.84745763 0.88135593
 0.90598291 0.8974359  0.8974359  0.88888889]
Accuracy is 88.5245545415037
Standard Deviation is 1.8042850829673611
```

Fig15. Accuracy for SVM

6. **Kernel-SVM**: A set of mathematical operations employed in Support Vector Machines, which provide the window to alter the data, are referred to in this as the "Kernel." In order for a non-linear decision surface to transform into a linear equation, the kernel function typically changes the training set of data.

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = kernel_svm_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.8559322  0.8559322  0.88135593 0.84745763 0.86440678 0.88983051
 0.88034188 0.87179487 0.87179487 0.86324786]
Accuracy is 86.82094741416776
Standard Deviation is 1.2587231209178158
```

Fig16. Accuracy for Kernel-SVM

Here, we also implemented the technique of Grid Search to understand the best parameters suitable for our dataset classification. Based on this, we found that the 'Linear' kernel with a 'C' value of 0.25 is the best.

```
#Grid Search to find the best model and the best parameters
from sklearn.model_selection import GridSearchCV
parameters = [{'C': [0.25, 0.5, 0.75, 1], 'kernel': ['linear']}, {'C': [0.25, 0.5, 0.75, 1], 'kernel': ['rbf'], 'gamma': [0.1
gridsearch =  GridSearchCV(estimator = svm_classifier, param_grid = parameters, scoring = 'accuracy', cv = 10, n_jobs = -1)

gridsearch.fit(X_train, y_train)
best_accuracy = gridsearch.best_score_
best_parameters = gridsearch.best_params_

print("Best Accuracy is {}".format(best_accuracy*100))
print("Best parameters are ", best_parameters)
```

```
Best Accuracy is 88.86426191510937
Best parameters are  {'C': 0.25, 'kernel': 'linear'}
```

Fig17. Accuracy with Grid Search for Kernel-SVM

7. **K-Nearest Neighbors (KNN):** A non-parametric approach for classification and regression issues is called K-nearest neighbors. The Minkowski distance, the Manhattan distance, and the Euclidean distance are the three most often used distance measurements in everyday life. The principle behind regression problems is to determine the new instance value by averaging its K neighbors, which is 5 in our case. KNN may perform well when there are few features, but it struggles as feature dimensions grow significantly. For more details, consider the books by Friedman, Hastie, and Tibshirani [10] and Murphy [11].

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = knn_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.83050847 0.8559322  0.83898305 0.83050847 0.84745763 0.88983051
 0.88034188 0.87179487 0.86324786 0.84615385]
Accuracy is 85.54758800521513
Standard Deviation is 1.948186000722126
```

Fig18. Accuracy for KNN

8. **Naïve Bayes (NB)**: A probabilistic method that makes use of the Bayes Theorem is naive Bayes. The Bayes Theorem defines the likelihood that an event will occur based on knowledge of related aspects that have already occurred. The conditional independence assumption of its features is another crucial aspect of Naive Bayes. According to this presumption, the existence of a feature would not affect any other features.

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = naivebayes_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.66101695 0.70338983 0.79661017 0.66949153 0.66949153 0.76271186
 0.73504274 0.64102564 0.70940171 0.62393162]
Accuracy is 69.7211357380849
Standard Deviation is 5.211717474042669
```

Fig19. Accuracy for Naïve bayes

9. **AdaBoost**: AdaBoost is a potent and effective technique for lessening a machine learning model's bias, which can raise prediction accuracy. It is an ensemble learning technique that was initially developed to boost the performance of binary classifiers (sometimes referred to as "meta-learning"). AdaBoost uses an iterative process to improve poor classifiers by learning from their errors. The base estimator here is the DecisionTreeClassifier with number of estimators 70 and the learning rate of 2.

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = adaboost_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.77966102 0.74576271 0.77966102 0.75423729 0.81355932 0.8559322
 0.82051282 0.77777778 0.76068376 0.72649573]
Accuracy is 78.1428364479212
Standard Deviation is 3.6864231551344773
```

Fig20. Accuracy for AdaBoost

10. **Bagging:** A data mining algorithm's performance can be enhanced by using a sort of ensemble learning method called bagging. It is an easy and efficient way to lower the variance of a machine learning model, which can increase predictability's stability and resilience. When working with employee data, where the relationships between the predictors and the outcome may be complex and non-linear, this is very crucial. The base estimator here

is the DecisionTreeClassifier with number of estimators = 250 and the max_samples = 16.

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = bagging_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.84745763 0.84745763 0.84745763 0.84745763 0.84745763 0.84745763
 0.85470085 0.84615385 0.84615385 0.84615385]
Accuracy is 84.77908155874256
Standard Deviation is 0.2375999466812101
```

Fig21. Accuracy for Bagging Classifier

11. **XGBoost**: In 2014, Chen [12] presented the tree-based approach known as Extreme Gradient Boosting (XGBoost). It is a precise and scalable implementation of gradient boosted trees that was specifically created to maximize computation efficiency and model performance. In contrast to gradient boosting, XGBoost uses a regularization term to minimize the overfitting effect, producing better predictions [13] and significantly faster computational run times.

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = randomforest_classifier, X = X_train, y = y_train, cv = 10)
print("Accuracies for the 10-folds is {}".format(accuracies))
print("Accuracy is {}".format(accuracies.mean()*100))
print("Standard Deviation is {}".format(accuracies.std()*100))
```

```
Accuracies for the 10-folds is [0.86440678 0.8559322  0.88135593 0.86440678 0.86440678 0.87288136
 0.86324786 0.88034188 0.86324786 0.84615385]
Accuracy is 86.56381283499928
Standard Deviation is 1.0027620115213363
```

Fig22. Accuracy for XGBoost

Results

We compare the above models applied for this dataset with the help of below metrics:

1. Accuracy
   Ex:

```
[[207  29]
 [ 38  20]]

0.7721088435374149
```

Fig23. Accuracy Metrics

2. Confusion Matrix
   Ex:

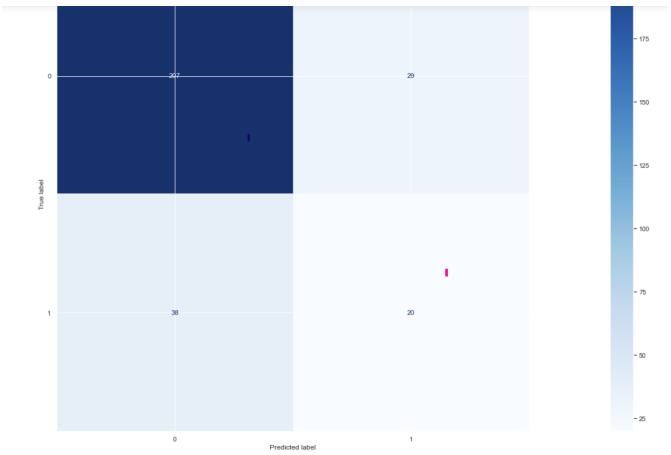

Fig24. Confusion Matrices

3. Precision, Recall, F1 score, and support.
   Ex:

	precision	recall	f1-score	support
0	0.84	0.88	0.86	236
1	0.41	0.34	0.37	58
accuracy			0.77	294
macro avg	0.63	0.61	0.62	294
weighted avg	0.76	0.77	0.76	294

Fig25. Precision, recall, f1-score, and support representation

4. K-fold Cross-validation (k=10)
   Ex:

```
Accuracies for the 10-folds is [0.86440678 0.8559322  0.88135593 0.86440678 0.86440678 0.87288136
 0.86324786 0.88034188 0.86324786 0.84615385]
Accuracy is 86.56381283499928
Standard Deviation is 1.0027620115213363
```

Fig26. K-fold cross validation

5. ROC and AUC
   Ex:

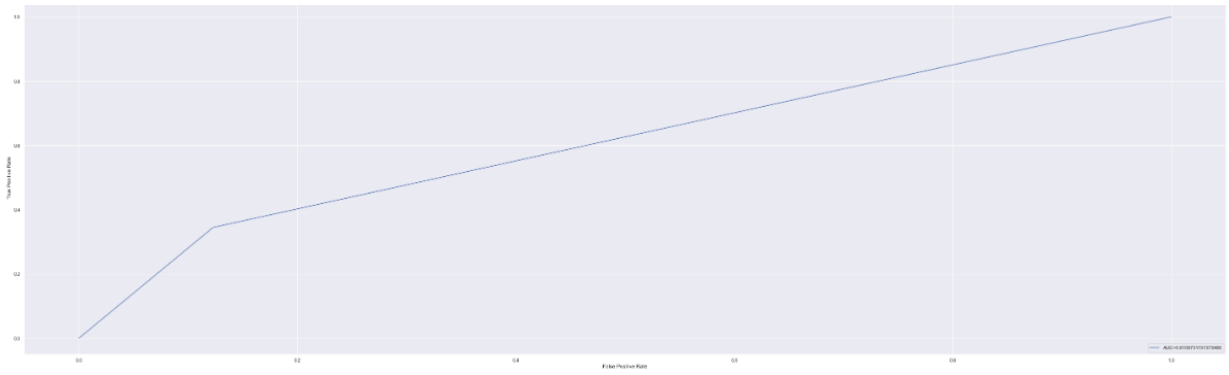

Fig27. ROC and AUC

From the results and comparison metrics obtained, we see that the algorithms – Logistic Regression, Random Forest, Kernel SVM and XGBoost give us the best accuracy scores of >85% even with 10-fold cross-validation.

Conclusion

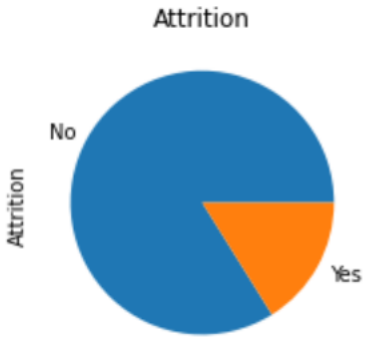

Fig28. Pie chart representation of target attribute

When compared to "Yes," the dataset has more values for "No" than for "Yes" for attrition. Even though the data is not entirely accurate, we were nevertheless able to achieve accuracy levels above 85%. Dimensionality reduction has also shown that the data are not entirely linearly separable. As we can see from the aforementioned classification models, Logistic Regression, SVM and Kernel SVM, Random Forest, and XGBoost Classifier offer greater comparative accuracies. Since the mean accuracy of 10-fold cross-validation is near to that of the test accuracy, we employed the K-Fold cross-validation procedures to make sure we did not get fortunate with the accuracy computation.

REFERENCES

1. L. Branham, The 7 hidden reasons employees leave: How to recognize the subtle signs and act before it's too late, 2005. [Online]. Available:https://ebookcentral.proquest.com/lib/sjsu/reader.action?docID=242984.
2. https://business.linkedin.com/talent-solutions/blog/trends-and-research/2018/the-3-Industries- with-the-highest-turnover-rates.
3. P. C. Bryant and D. G. Allen, "Compensation, Benefits and Employee Turnover: HR Strategies For Retaining Top Talent," Compensation and benefits review, vol. 45, issue 3, pp. 171-175, May 2013.
4. A. Robison, "Effective Employee Retention Strategies Use HR Self-Service," kioskMarketplace.com, Nov. 2018. [Online]. Available: https://link.gale.com/apps/doc/A562171993/ITOF?u=csusj&sid=ITOF&xid=cd0dc0cc.
5. Abrams and M. N," Employee retention strategies: lessons from the best," Healthcare executive, vol. 19, issue 4, pp. 18-22, 2004.
6. Cox, D.R.: The regression analysis of binary sequences. J. Roy. Stat. Soc. B. Met., 215–242 (1958)
7. Efron, B.S., Hastie, T.: Computer Age Statistical Inference. Cambridge University Press, Cambridge (2016)
8. Morgan, J.N., Sonquist, J.A.: Problems in the analysis of survey data, and a proposal. J. Am. Stat. Assoc. 58, 415–434 (1963)
9. Breiman, L.: Random forests. Mach. Learn. 45,5–32 (2001)
10. Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning. Springer, New York (2001)
11. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT press, Cambridge (2012)
12. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785–794, ACM (2016)
13. Punnoose, R., Ajit, P.: Prediction of employee turnover in organizations using machine learning algorithms. Int. J. Adv. Res. Artif. Intell. 5,22–26 (2016)