**Background**

* Currently, employment levels in the UK are around 75%, however, according to UK Money, attrition rates are around 17%.
* Attrition is defined as the departure of employees from employment for whatever reason, voluntary or involuntary.
* A normal attrition rate for a company is around 10%. After 20%, the company would need to start investigating their employees and work procedures so as to find a way to lower attrition back to a healthy level.
* Our project looks at employee attrition, the factors that affect it, and a logistic regression model that can accurately predict employee departure, given background data on the subject.

**Data Source**

* For our project, we used IBM’s HR analytics and employee attrition csv, sourced via Kaggle.
* We liked this dataset as it had a range of metrics for us to analyze.

**Data Import**

PostgreSQL

* Since we were required to import our data source from either SQL or Spark, we opted to use SQL.
* Firstly, we created a table in postgres to house our csv data (see table schema).
* We then imported the csv file into SQL and verified the import.

Psycopg2

* In order for us to execute SQL commands in python (including visualizing the table), we needed to install the psycopg2 module.
* First, we establish a connection to the database.
  + We needed to provide details such as the database name, postgres username, and postgres password.
* A SQL query was executed to view the whole table, while pandas was used to convert this into a DataFrame.
* And at the end, the connection was closed to disconnect it from the database.

Gitignore

* Since we had used a confidential password for the postgres database, we included it in out gitignore.

**Data Pre-processing**

Input data

* Our input data had 35 columns, each measuring different metrics relating to job employment and work. Some were personal details such as age, marital status, gender, while some were different facets of work life, such as salary, satisfaction, training, etc.
* Initially, we identified all these columns as features for our model, except for attrition, which we designated as our target. Most of the data were integers, with a few categorical columns

Dropping columns

* For our first step in pre-processing, we identified 4 columns that we could remove straight away: employee count, over 18, and standard hours were the same for all employees, while employee number was just an identifier number that was not necessary for analysis.

Encoding

* We used get dummies to encode any categorical data, and we created a function to encode columns that contained Boolean data (e.g. we changed all the “Yes” and “Male” to 1, and “No” and “Female” to 0).

**LR Model**

* After obtaining our new features from the target, we split the data into testing and training set then fit the training data into our Logistic regression model.
* We then made predictions on the training data and measured it up against the actual outcomes.

**Evaluation**

* To evaluate our model, we generated a classification report and a confusion matrix, as shown here.
* Our model has an accuracy of 83%, which is acceptable.
* Our model has an almost perfect recall when it comes to attrition; of the 309 cases of attrition in the testing data set, the model was accurately able to predict 304 of them as attrition.

**Optimization**

Optimization methods

* In order to optimize our model, we looked at a range of methods:
* We looked at pre-processing our input data further, to remove and reduce columns e.g. removing columns outright.
* We also looked at reducing the number of features so that only the most important ones are used in the model i.e. the features that have the greatest impact on attrition. We investigated 4 methods to discern the most important features, which we will explore later:
  + Comparing model coefficients
  + Permutation importance.
  + Recursive feature elimination (RFE)
  + Correlation Analysis.

Further pre-processing

* We looked at pre-processing the data further; from the source data, we noticed that there were 3 “rate” columns – hourly rate, daily rate, and monthly rate, as well as monthly income. We dropped these 3 columns as they were a source of redundancy. We kept the monthly income column as a signifier for salary.

Model coefficients

* Comparing model coefficients by looking at the weights of model coefficients for each feature.
* We obtained the coefficients for our model in an array, before converting it into a dataframe, and exporting it as a bar plot.
* The bar plot shows that the most important features, by model weight, are: number of companies worked, years since last promotion, environment satisfaction, job satisfaction, and years with current manager.
* So we recreated our model, and ran it with only these 5 columns as features. We will discuss the evaluation of this later.

Permutations

* We repeated this process but this time, we used permutation importance to obtain our most important features.
* Permutation Importance shuffles the values in a certain feature, and tests the model. If the performance of the model drops significantly, then that feature is important. However, if shuffling random values in a feature has minimal impact on performance, we can ignore that feature.
* We can see from the bar plot that the most important features are distance from home, years in current role, years with current manager, number of companies worked, and monthly income.
* Again, we re-fitted our model with only these 5 features.

Correlation analysis

* Correlation analysis identifies the variables that have a strong correlation with our target. (e.g. correlation of >0.8 signifies that 2 variables are strongly linked)
* For this, we created a correlation matrix, which is a table detailing how strongly 2 features are linked to each other.
* We converted the matrix to a heatmap, as well as isolating the “attrition” column to identify the features that are strongly linked to attrition.
* Again, we ran our Logistic regression model with only the 5 most important features.

Recursive feature elimination

* RFE ranks all the features in terms of importance and removes the least important feature. This process repeats until we are left with a desired number of features (we chose to look at the 5 most important features).
* RFE identified these features as being important, so we re-ran our model with these features.

**Evaluation**

* We evaluated the accuracy scores of our different optimization methods and discovered that RFE performed best, with an accuracy of 85%.