Mini Project

Employee Attrition and Employee Churn

Submitted By: -

Purvi Negi (20522006),
Kashish Jindal (20261273),
Shourya Kothiya (20261174),
Varun Singh Rana (20261140).

MBA AI & DS

Submitted To: -

Graphic Era University

DECLARATION BY THE CANDIDATE

I the undersigned solemnly declare that the project report **Employee Attrition and Employee Churn** is based on my own work carried out during the course of our study under the supervision of **Dr. Vikas Tomar.**

I assert the statements made and conclusions drawn are an outcome of my research work. I further certify that

I. The work contained in the report is original and has been done by me under the general supervision of my supervisor.

II. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University of India or abroad.

III. We have followed the guidelines provided by the university in writing the report.

IV. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.

Purvi Negi (20522006),
Kashish Jindal (20261273),
Shourya Kothiya (20261174),
Varun Singh Rana (20261140).

CERTIFICATE BY SUPERVISOR

This to certify that the report of the project submitted is the outcome of the project work entitled "**Employee Attrition and Employee Churn**" carried out by Purvi Negi (20522006), Kashish Jindal (20261273), Shourya Kothiya (20261174) and Varun Singh Rana (20261140) of Graphic Era Deemed to be University. Carried by under my guidance and supervision.

To the best of my knowledge the report

I. Embodies the work of the candidate him/herself,
II. Has duly been completed,
III. Fulfils the requirement of the ordinance relating to the MBA degree of the University and
IV. Is up to the desired standard for the purpose of which is submitted.

(Signature of the Supervisor)

Name: Dr. Vikas Tomar
Designation:
Department:

Certificate

This is to certify that the following students of Graphic Era Deemed to be University belonging to II semester have successfully completed the project work titled "**Employee Attrition and Employee Churn**" during the year 2021-2022.

Name of the students: -

Purvi Negi, Roll no (20522006)
Kashish Jindal, Roll no (20261273)
Shourya Kothiya, Roll no (20261174)
Varun Singh Rana, Roll no (20261140)

Internal examiner External examiner Head of the Department

Acknowledgement

We will use this opportunity to express our gratitude to everyone who supported us throughout the course of this MBA AI & DS project. We are thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. We are sincerely grateful to them for sharing their truthful and illuminating views on the number of issues related to the project.

We are highly thankful to Dr. Vikas Tomar sir for his supervision and guidance on the project "Employee Attrition and Employee Churn" with his immense knowledge and experience.

We are also very thankful to Capt. Rajshree Thapa ma'am for her continuous guidance and support throughout the completion of our project.

We are short of words to convey our gratitude to all the faculty members of Management department who were always there when we needed them.

Purvi Negi (20522006),
Kashish Jindal (20261273),
Shourya Kothiya (20261174),
Varun Singh Rana (20261140).

Index

Employee Attrition and Employee Churn

Purvi Negi, Kashish Jindal, Shourya Kothiya, Varun Singh Rana
MBA AI & DS
Graphic Era University

Abstract

Employee turnover is a serious concern in knowledge-based organizations. When employees leave an organization, they carry with them invaluable tacit knowledge which is often the source of competitive advantage for the business. In order for an organization to continually have a higher competitive advantage over its competition, it should make it a duty to minimize employee attrition. This study identifies employee related attributes that contribute to the prediction of employees' attrition in organizations. There are several areas in which organisations can adopt technologies that will support decision-making, artificial intelligence is one of the most innovative technologies that is widely used to assist organisations in business strategies, organisational aspects and people management. The goal of this work is to analyse how objective factors influence employee attrition, in order to identify the main causes that contribute to a worker's decision to leave a company, and to be able to predict whether a particular employee will leave the company. After the training, the obtained model for the prediction of employees' attrition is tested on a real dataset provided by IBM analytics, which includes 36 features and about 1470 samples. Results are expressed in terms of classical metrics and the algorithm that produced the best results for the available dataset are Cluster Analysis, Gradient Boosting classifier (85.48% Accuracy) , Random Forest classifier (85.03% Accuracy) and K-NN classifier (85.71% Accuracy)

What is employee attrition?

Employee attrition is when an employee leaves the company through any method, including voluntary resignations, layoffs, failure to return from a leave of absence, or even illness or death. Whenever anyone ceases working for the company for any reason *and is not replaced for a long time (if ever), that would be employee attrition.*

There are two main types of employee attrition:

Voluntary attrition: When an employee chooses to leave the company, that is voluntary attrition. This can include any reason an employee leaves on their own accord, whether it's truly voluntary or not. True voluntary terminations, such as resignations for a new job or to move across the country, are the ones you're probably most familiar with. But an employee who leaves due to health reasons or only quits because the work situation is toxic can also fall under voluntary attrition. The company retains the decision not to replace the employee–although there are some times the company would like to replace someone but cannot.

Involuntary attrition: When the company decides to part ways with an employee, this is involuntary attrition. This can be through a position elimination, for example, due to reorganization or layoffs, for cause (such as stealing or fighting), poor performance, or termination when someone abandons their job. (You can argue the last one is a voluntary termination, but the company makes the final call to terminate.) The company then doesn't backfill the position or eliminates it.

Involuntary attrition through position elimination is the most common form of attrition, as the company decides proactively to eliminate a position. For other types of termination, the company usually decides after the termination to leave the job vacant.

Is employee attrition always bad?

While having employees leaving in droves can be bad, if a company needs to eliminate positions to stay financially afloat, attrition doesn't have to be a negative thing.
When your business struggles, it can be easier not to replace people who leave voluntarily than eliminating positions. (Depending on where your organization based, you may be required to pay severance or pay for long notice periods if you decide to eliminate jobs proactively.)

However, if your attrition levels are too high, you can run into some serious problems. You can have a lack of continuity, training gaps, and a lack of institutional knowledge. It can take a long time to fill positions (especially specialized roles), and if you leave these positions empty, it can become difficult to fill these positions later. These holes can cause burnout for your remaining employees and lower overall productivity, leading to unhappy customers.

Sometimes companies implement hiring freezes in an attempt to increase attrition. If someone quits, the remaining employees must absorb their work. The work ceases to be done, or managers reorganize employees to fill holes.
Even when the attrition is involuntary and the result of carefully planned layoffs, you need to be concerned about the future. If you lay off all your salespeople today because product sales are down, you'll never have a chance to recover. No salespeople mean no sales, and no sales mean no company.

Recruiting, hiring, and training new employees is costly, so you might consider not terminating someone and not eliminating that position if you need it again later. It may make more sense to cross-train that employee for a time and retain the person.

What Caused employee attrition?

Several factors affect attrition, some of which are under your control and some of which are not. Here are a few:

Low unemployment: If your area or industry has low unemployment rates, you may not be able to replace employees who leave, even if you wanted to. You can either leave the positions vacant (attrition) or reduce your hiring standards or expand your search.

Workforce demographics: If your company is filled with Baby Boomers, their upcoming retirements may result in a loss of staff that you cannot easily replace.

Toxic workplace: If your business is not a good place to work, you may find it challenging to keep positions filled. You may wish to lower your attrition rate but find it challenging to do so because people leave quickly, and your company's reputation as a toxic employer spreads. If you're experiencing a high level of attrition, this is one of the first areas you should investigate. Listen to employees in exit interviews and pay attention to your online reviews. If your

Glassdoor reviews are negative, take them to heart and work to correct them.

Business relocation: For example, it might make sense for your company to move from California to Texas, as Elon Musk just announced he would do. Even if you offer to pay full relocation costs for all employees, many will choose not to come with you. It will take you time to hire replacements.

Covid-19 shutdowns: This has been a unique problem to 2020 (and hopefully will resolve shortly in 2021), but many companies needed to terminate employees when governments required all non-essential businesses to shut down. Even businesses that remained open may have lost business and, therefore, needed to let employees go.

Reorganizations/restructuring: This type of attrition is the goal of these company reorganizations. Positions are intentionally and carefully eliminated with the plan of never refilling them.

How to prevent voluntary attrition

If you're finding it difficult to hire and retain employees, resulting in high attrition rates, you can work to fix the problems.

1. **Get your managers the training they need to manage employees effectively.** Remember, management is a different skill than doing. Invest in management training if you want to lower attrition. This will help you maintain managers as well as staff.

2. **Do a salary survey and benchmark all your salaries.** If you're paying below-market rates, it can be difficult to retain staff.

3. **Conduct stay interviews.** Only do this if you are interested in listening to the employees and willing to make changes based on what you learn. If you hold these and then don't act on all the information collected, you will build resentment among employees.

4. **Revise your benefits and perks to offer ones that your employees like.** You may need to do this every few years as your employees' needs change. For instance, if your company has mainly younger employees, they won't be as concerned about time off for school attendance. But as your employees age–if you retain them–they are likely to have children and favor more child-friendly perks.

5. **Consider allowing more flexibility**, either in start times or telecommuting. 2020 taught a lot of businesses that many positions could be done from home just as effectively as from the office.

6. **Take care to hire the right people in the first place.** You shouldn't be wasting time looking for a unicorn candidate, you should focus on finding

7. **Someone who can do the job and wants to do the job**. Finding the right fit in your selection process in the first place can reduce your resignations.

8. **Have accurate job postings.** This helps attract the type of candidates who are happy to work for your company. For instance, many companies are listing jobs as "remote" but plan to bring those employees into the office when governments lift restrictions. That type of bait and switch will likely result in increased attrition.

9. **Remember to promote from within.** Yes, it takes a lot of effort. However, if you employ strategic workforce planning – and succession management to make sure that you're filling in the skills/leadership gaps, you'll be able to promote people and plan for the business's future even when your employees retire or leave due to other reasons.

There are also ways to manage involuntary attrition. For example, thoroughly prepared hiring plans will help you prevent overstaffing and then having to let people go when your business doesn't grow as intended.

What is employee churn?

Employee churn is the overall turnover in an organization's staff as existing employees leave and new ones are hired. Employee churn can be defined as the percentage of employees leaving an organization over a specific period. An employee's exit can be voluntary or involuntary. Resignation or retirement falls in the voluntary category, whereas, an employee being let go will be in the involuntary category. Some voluntary churn is avoidable with hiring best practices, building a positive workplace culture, etc.

According to the Society of Human Resources Management's (SHRM) 2018 Employee Recognition Report, employee churn is the number one challenge for most organizations globally.

A staggering 29% of these organizations admitted to being stressed about finding replacements.

Although a certain level of employee turnover is normal in any organization, however, high rates of employee churn can be a costly affair. Employee onboarding, hiring, training, and development require a financial outlay, and a new hire may not be immediately effective in terms of bringing in profits. You have to give them a certain amount of time to learn, adjust, and start contributing.

Factors contributing to employee churn

While many factors add to employee churn, some easily stand out and need management attention. Let's look at the top 5 factors that contribute to employee churn:

1. **Lack of growth plan and opportunities:** Every employee needs to have a defined growth path that needs to be conveyed to them clearly. Lack of this can lead to dissent and doubts that their efforts are unrecognized. This could lead to resentment and employees leaving the organization.
2. **Skewed work-life balance:** Striking the right work-life balance is crucial for employees, be it for family, hobbies, or pursuing higher studies. Not having the right balance or organizations not helping find it can cause employees to exit the organization.
3. **Bad workplace culture:** Having a positive culture is essential for organizations. A culture where employees are appreciated, valued, rewarded is highly appreciated. If employees feel the work culture is toxic and non-conducive to their growth, they will leave the organization.
4. **Dissatisfactory appraisals:** Many organizations follow the yearly appraisal process. Employees eagerly wait for that for the entire year. If they feel the assessment is not in-line with their efforts and contributions, they may get disheartened. Some employees may stick it out for another cycle, but most will leave as early as possible.
5. **Low staff morale:** The above factors can affect employee morale and bring it down significantly. It could either be one of the elements or a combination, but employees with low morale are a significant attrition risk.

How to contain employee churn?

Employees join and exit, but when a significant number of employees leave the organization, it is undoubtedly a matter of concern. The amount of money, efforts, and resources required to hire new people or replacements are enormous. Arresting employee churn is essential for smooth operations. Here are 4 ways to help reduce employee churn:

1. **Hire the right talent:** It all starts with hiring an employee. As a rule of thumb, organizations hire an employee based on their skill set, but how well do you know if the employee is the best fit for your organization's culture? You must hire employees who not only have a strong skill set but are also the right fit for your culture. At the time of hiring, ask them organizational behavioural questions to understand their mindset and their intent to stay in the organization.
2. **Recognize achievements:** Your employees need to be recognized and encouraged. Show them you notice their hard work, reward them when they go above and beyond. This creates a positive impact on the minds of the employees, and they appreciate it.
3. **Provide the best compensation:** Employees want to be compensated well. There will always be a gap as to what they expect and what organizations can and will pay. What organizations can do, however, is offer compensation that is current, meets industry standards, and rewards excellent performance via incentives or other benefits.
4. **Provide benefits:** The workforce has changed over the years, and so have its benefits and expectations. Some employees travel long distances while some juggle multiple responsibilities. Taking note of that and allowing a few options like work from home or flexible hours will certainly make employees happy. Benefits such as college debt management, health insurance, and dental care coverage will go a long way in containing employee exits from organizations.

You may retain some employees, if not all, and that's fine. Employees have various ambitions, desires, and ideas about work that they need to pursue. All you can do in this case is to ensure they had a great experience. This way, they can always feel like and come back to the organization.

Machine Learning

Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.

Machine learning is the science of getting computers to act without being explicitly programmed. In the past decade, machine learning has given us self-driving cars, practical speech recognition, effective web search, and a vastly improved understanding of the human genome. Machine learning is so pervasive today that you probably use it dozens of times a day without knowing it. Many researchers also think it is the best way to make progress towards human-level AI.

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics. As big data continues to expand and grow, the market demand for data scientists will increase, requiring them to assist in the identification of the most relevant business questions and subsequently the data to answer them.

How machine learning works

1. **A Decision Process**: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labelled or unlabelled, your algorithm will produce an estimate about a pattern in the data.

2. **An Error Function**: An error function serves to evaluate the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

3. **A Model Optimization Process**: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this evaluate and optimize process, updating weights autonomously until a threshold of accuracy has been met.

Machine learning classifiers

Machine learning classifiers fall into three primary categories.

Supervised machine learning

Supervised learning, also known as supervised machine learning, is defined by its use of labeled datasets to train algorithms that to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately. This occurs as part of the cross validation process to ensure that the model avoids overfitting or underfitting. Supervised learning helps organizations solve for a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox. Some methods used in supervised learning include neural networks, naïve bayes, linear regression, logistic regression, random forest, support vector machine (SVM), and more.

Unsupervised machine learning

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyse and cluster unlabeled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, image and pattern recognition. It’s also used to reduce the number of features in a model through the process of dimensionality reduction; principal component analysis (PCA) and singular value decomposition (SVD) are two common approaches for this. Other algorithms used in unsupervised learning include neural networks, k-means clustering, probabilistic clustering methods, and more.

Semi-supervised learning

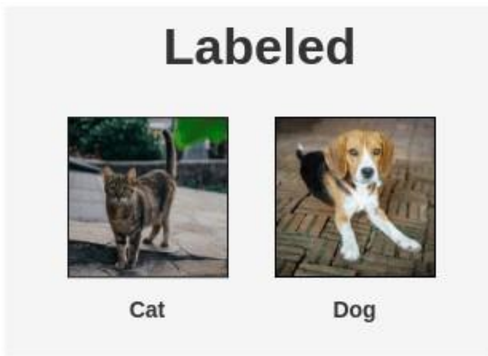
Semi-supervised learning offers a happy medium between supervised and unsupervised learning. During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set. Semi-supervised learning can solve the problem of having not enough labeled data (or not being able to afford to label enough data) to train a supervised learning algorithm.

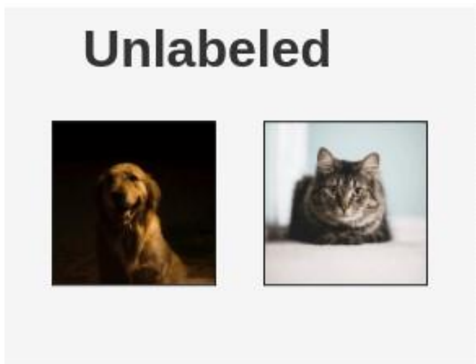
Labelled and Unlabelled Data

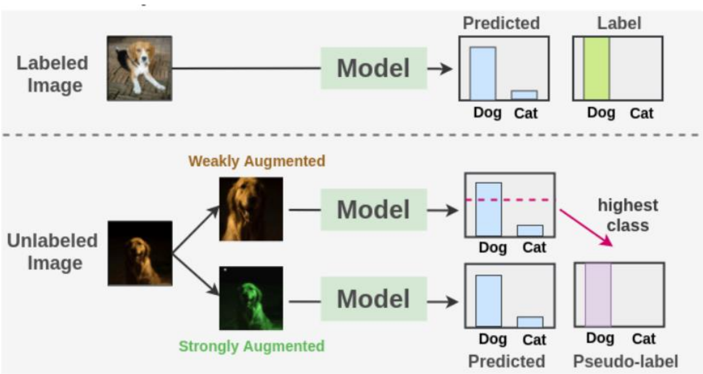
Labelled Data: Data that comes with a label

Unlabelled Data: Data that comes without a label

UNLABELLED DATA	LABELLED DATA
Used in unsupervised machine learning	Used in supervised machine learning
Obtained by observing and collecting	Needs human/expert to annotate
Comparatively easy to get and store	Expensive, hard and time-consuming to get and store
Often used to pre-process datasets	Used for complex predicting tasks







Preprocessing the data for ML

Pre-processing the data for ML involves both data engineering and feature engineering. Data engineering is the process of converting *raw data* into *prepared data*. Feature engineering then tunes the prepared data to create the features expected by the ML model.

Pre-processing operations

Data pre-processing includes various operations. Each operation aims to help machine learning build better predictive models. The details of these preprocessing operations are outside the scope of this article, but for your reference, some of the operations are briefly discussed in this section.

For structured data, data preprocessing operations include the following:

- **Data cleansing**. Removing or correcting records with corrupted or invalid values from raw data, as well as removing records that are missing a large number of columns.

- **Instances selection and partitioning**. Selecting data points from the input dataset to create training, evaluation (validation), and test sets. This process includes techniques for repeatable random sampling, minority classes oversampling, and stratified partitioning.

- **Feature tuning**. Improving the quality of a feature for ML, which includes scaling and normalizing numeric values, imputing missing values, clipping outliers, and adjusting values with skewed distributions.

- **Representation transformation**. Converting a numeric feature to a categorical feature (through bucketization), and converting categorical features to a numeric representation (through one-hot encoding, learning with counts, sparse feature embeddings, and so on). Some models work only with numeric or categorical features, while others can handle mixed type features. Even when models handle both types, they can benefit from different representation (numeric and categorical) of the same feature.

- **Feature extraction**. Reducing the number of features by creating lower-dimension, more powerful data representations using techniques such as PCA, embedding extraction, and hashing.

- **Feature selection**. Selecting a subset of the input features for training the model, and ignoring the irrelevant or redundant ones, using filter or wrapper methods. This can also involve simply dropping features if the features are missing a large number of values.

- **Feature construction**. Creating new features either by using typical techniques, such as polynomial expansion (by using univariate mathematical functions) or feature crossing (to capture feature interactions). Features can also be constructed by using business logic from the domain of the ML use case.

In this case we used Label Encoder for preprocessing as Label Encoder can be used to normalize labels and can also be used to transform non-numerical labels (as long as they are hashable and comparable) to numerical labels.

```
 # Import LabelEncoder
from sklearn import preprocessing
#creating labelEncoder
le = preprocessing.LabelEncoder()
# Converting string labels into numbers.
data['Department']=le.fit_transform(data['Department'])

#Spliting data into Feature and
X=data[['JobSatisfaction', 'YearsInCurrentRole', 'JobInvolvement',
       'MonthlyIncome', 'YearsAtCompany', 'JobLevel',
       'YearsSinceLastPromotion', 'Department', 'AppraisalRating' , 'PercentSalaryHike']]
y=data['Attrition']

# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
print(data)
```

```
      Index  Age Attrition     BusinessTravel  DailyRate  Department  \
0         0   41       Yes      Travel_Rarely       1102           2
1         1   49        No  Travel_Frequently        279           1
2         2   37       Yes      Travel_Rarely       1373           1
3         3   33        No  Travel_Frequently       1392           1
4         4   27        No      Travel_Rarely        591           1
...     ...  ...       ...                ...        ...         ...
1465   1465   36        No  Travel_Frequently        884           1
1466   1466   39        No      Travel_Rarely        613           1
1467   1467   27        No      Travel_Rarely        155           1
1468   1468   49        No  Travel_Frequently       1023           2
1469   1469   34        No      Travel_Rarely        628           1
```

Cluster Analysis

Cluster analysis is a technique to group similar observations into a number of clusters based on the observed values of several variables for each individual. Cluster analysis is similar in concept to discriminant analysis.

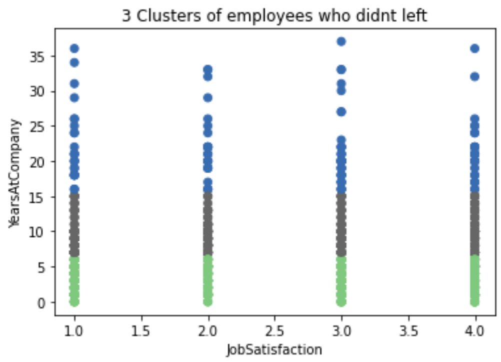
Cluster analysis is the name given to a set of techniques which ask whether data can be grouped into *categories* on the basis of their similarities or differences. It began when biologists started to classify plants on the basis of their various phyla and species and wanted to derive a less subjective technique. It has been applied to diagnostic classification in a similar way.

Cluster analysis aims at the detection of natural partitioning of objects. In other words, it groups observations that are similar into homogeneous subsets. These subclasses may reveal patterns related to the phenomenon under study. A distance function is used to assess if the similarity between objects and a wide variety of clustering algorithms based on different concepts is available. Similarity measures are first computed between observations, and between clusters once observations begin to be grouped into clusters. Several metrics, such as Euclidean and Manhattan distance, correlation, or mutual information, can be used to compute similarity. Additionally, several merging strategies that lead to different clustering patterns are possible. Clustering results are therefore somewhat subjective, as they greatly depend on the users' choices. Traditional cluster analysis is usually performed to group either observations or variables separately but simultaneous co-clustering (or biclustering) of the rows and the columns of the data matrix constitutes also a suitable alternative to search for biomarkers.

For example: - in this case, we use cluster to classify the dataset into two classes Green (Happy), Blue (Frustrated) and Grey (Unhappy)

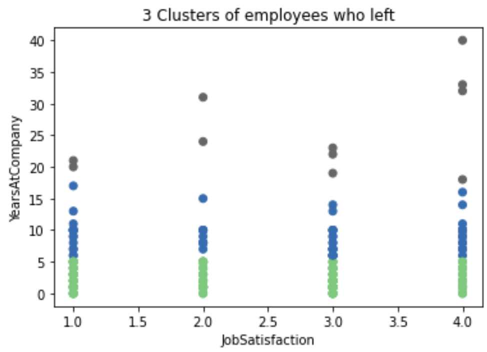
- Cluster for employees who didn't leave

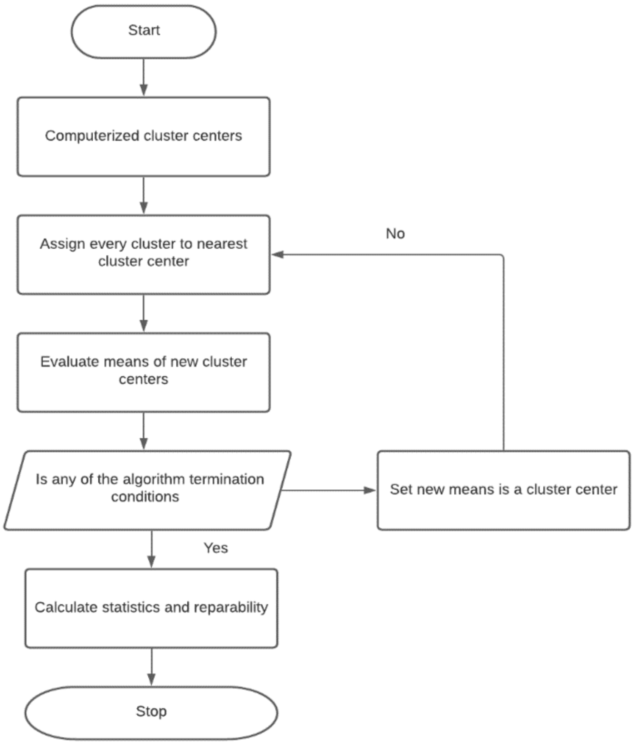
```
#Cluster Analysis who didnt left
from sklearn.cluster import KMeans
# Filter data
Attrition_emp =  data[['JobSatisfaction', 'YearsAtCompany']][data.Attrition == "No"]
# Create groups using K-means clustering.
kmeans = KMeans(n_clusters = 3, random_state = 0).fit(Attrition_emp)

# Add new column "label" annd assign cluster labels.
Attrition_emp['label'] = kmeans.labels_
# Draw scatter plot
plt.scatter(Attrition_emp['JobSatisfaction'], Attrition_emp['YearsAtCompany'], c=Attrition_emp['label'],cmap='Accent')
plt.xlabel('JobSatisfaction')
plt.ylabel('YearsAtCompany')
plt.title('3 Clusters of employees who didnt left')
plt.show()
```



- Cluster for employees who left

```
#Cluster Analysis who left
from sklearn.cluster import KMeans
# Filter data
Attrition_emp =  data[['JobSatisfaction', 'YearsAtCompany']][data.Attrition == "Yes"]
# Create groups using K-means clustering.
kmeans = KMeans(n_clusters = 3, random_state = 0).fit(Attrition_emp)

# Add new column "label" annd assign cluster labels.
Attrition_emp['label'] = kmeans.labels_
# Draw scatter plot
plt.scatter(Attrition_emp['JobSatisfaction'], Attrition_emp['YearsAtCompany'], c=Attrition_emp['label'],cmap='Accent')
plt.xlabel('JobSatisfaction')
plt.ylabel('YearsAtCompany')
plt.title('3 Clusters of employees who left')
plt.show()         # Green- Happy , Blue- Frustated , Grey- unhappy
```





Gradient Boosting Classifier

In Gradient Boosting, each predictor tries to improve on its predecessor by reducing the errors. But the fascinating idea behind Gradient Boosting is that instead of fitting a predictor on the data at each iteration, it actually fits a new predictor to the residual errors made by the previous predictor.
Gradient boosting is one of the most powerful techniques for building predictive models.

Gradient boosting is a supervised learning algorithm.

How Gradient Boosting Works

Gradient boosting involves three elements:

1. A loss function to be optimized.
2. A weak learner to make predictions.
3. An additive model to add weak learners to minimize the loss function.

1. Loss Function

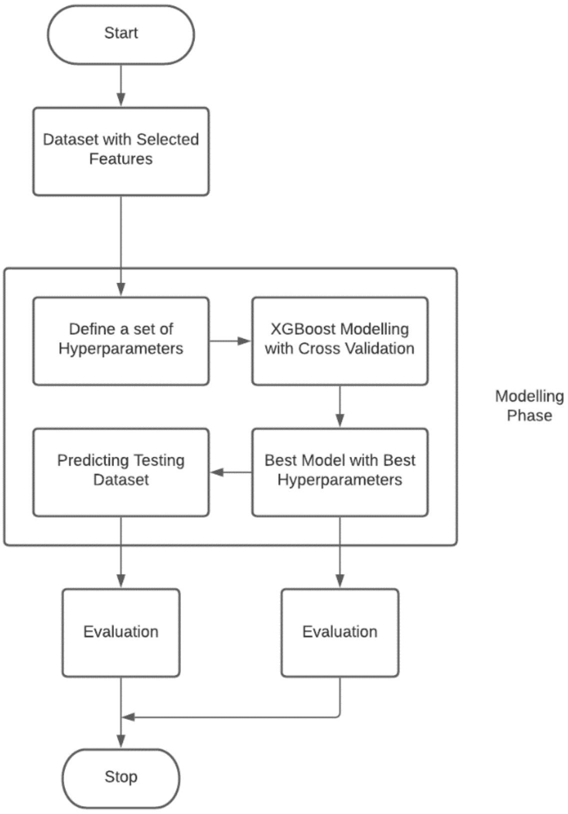
The loss function used depends on the type of problem being solved.
It must be differentiable, but many standard loss functions are supported and you can define your own.
For example, regression may use a squared error and classification may use logarithmic loss.
A benefit of the gradient boosting framework is that a new boosting algorithm does not have to be derived for each loss function that may want to be used, instead, it is a generic enough framework that any differentiable loss function can be used.

2. Weak Learner

Decision trees are used as the weak learner in gradient boosting.
Specifically, regression trees are used that output real values for splits and whose output can be added together, allowing subsequent models outputs to be added and “correct” the residuals in the predictions.
Trees are constructed in a greedy manner, choosing the best split points based on purity scores like Gini or to minimize the loss.
Initially, such as in the case of AdaBoost, very short decision trees were used that only had a single split, called a decision stump. Larger trees can be used generally with 4-to-8 levels.
It is common to constrain the weak learners in specific ways, such as a maximum number of layers, nodes, splits or leaf nodes.
This is to ensure that the learners remain weak, but can still be constructed in a greedy manner.

3. Additive Model

Trees are added one at a time, and existing trees in the model are not changed.
A gradient descent procedure is used to minimize the loss when adding trees.
Traditionally, gradient descent is used to minimize a set of parameters, such as the coefficients in a regression equation or weights in a neural network. After calculating error or loss, the weights are updated to minimize that error.



Instead of parameters, we have weak learner sub-models or more specifically decision trees. After calculating the loss, to perform the gradient descent procedure, we must add a tree to the model that reduces the loss (i.e., follow the gradient). We do this by parameterizing the tree, then modify the parameters of the tree and move in the right direction by (reducing the residual loss.

For example, in this case, Gradient Boosting had accuracy: 0.85487

Code:

```
#Import Gradient Boosting Classifier model
from sklearn.ensemble import GradientBoostingClassifier

#Create Gradient Boosting Classifier
gb = GradientBoostingClassifier()

#Train the model using the training sets
gb.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = gb.predict(X_test)

#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Gradient Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Gradient Accuracy: 0.854875283446712
```

Random Forest Classifier

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result.

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes become our model's prediction

The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds. In data science speak, the reason that the random forest model works so well is:

A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

The low correlation between models is the key. Just like how investments with low correlations (like stocks and bonds) come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions.

The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. So, the prerequisites for random forest to perform well are:

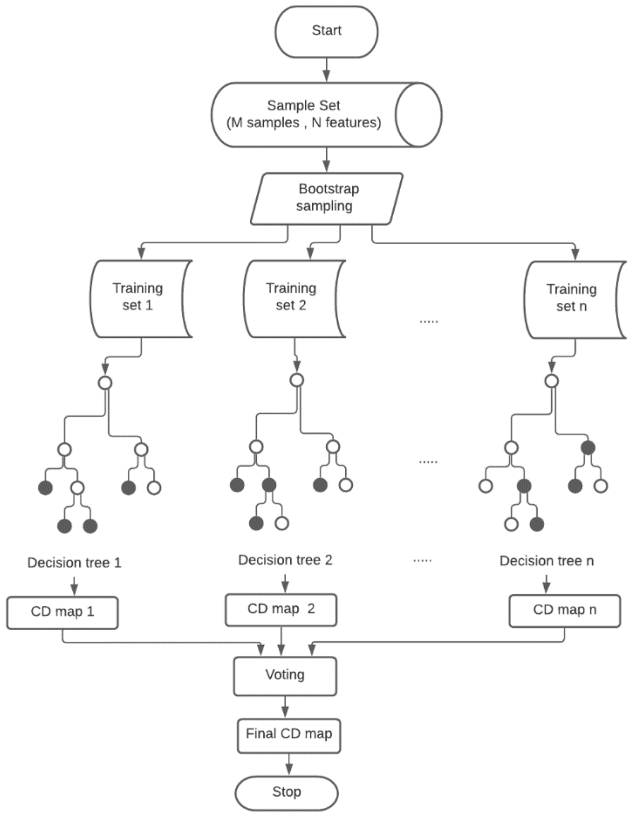
1. There needs to be some actual signal in our features so that models built using those features do better than random guessing.
2. The predictions (and therefore the errors) made by the individual trees need to have low correlations with each other.

Bagging (Bootstrap Aggregation) — Decisions trees are very sensitive to the data they are trained on, small changes to the training set can result in significantly different tree structures.

Random forest takes advantage of this by allowing each individual tree to randomly sample from the dataset with replacement, resulting in different trees. This process is known as bagging.

For example, in this case, random forest had,

Recall Score: 0.08196721311475409
Acuuracy_Train_Log: 1.0
Acuuracy_Test_Log: 0.8503



Code:

```
# Random Forest Classifier

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n_estimators=100 ,criterion='entropy', random_state=100 )
classifier.fit(X_train , y_train)

rf_pred = classifier.predict(X_test)

from sklearn.metrics import classification_report , accuracy_score , recall_score

print("Random Forest")
print(classification_report(y_test, rf_pred))
print("Recall Score : ", recall_score(y_test, rf_pred))
print("Acuuracy_Train_Log : ", classifier.score(X_train, y_train))
print("Acuuracy_Test_Log : ", accuracy_score(y_test, rf_pred))
```

```
Random Forest
              precision    recall  f1-score   support

           0       0.87      0.97      0.92       380
           1       0.33      0.08      0.13        61

    accuracy                           0.85       441
   macro avg       0.60      0.53      0.52       441
weighted avg       0.79      0.85      0.81       441

Recall Score :  0.08196721311475409
Acuuracy_Train_Log :  1.0
Acuuracy_Test_Log :  0.8503401360544217
```

K-NN

The k-nearest neighbour (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.
The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

"Birds of a feather flock together."

The **k-nearest neighbours algorithm** (**k-NN**) is a non-parametric classification method first developed by Evelyn Fix and Joseph Hodges in 1951,[1] and later expanded by Thomas Cover. It is used for classification and regression. In both cases, the input consists of the *k* closest training examples in data set. The output depends on whether k-NN is used for classification or regression:

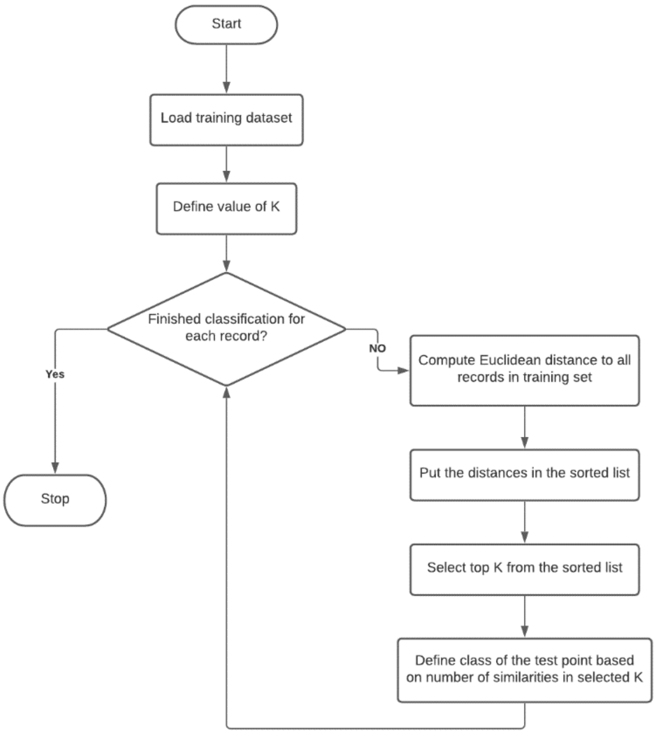
- In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbours, with the object being assigned to the class most common among its *k* nearest neighbours (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbour.

- In k-NN regression, the output is the property value for the object. This value is the average of the values of *k* nearest neighbours.

k-NN is a type of classification where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically.
Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbours, so that the nearer neighbours contribute more to the average than the more distant ones.

The neighbours are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

For example, in this case, K-NN had an accuracy of 0.8571



Code:

```
#K -NN
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

for K in range(25):
    K_value = K+1
    neigh = KNeighborsClassifier(n_neighbors = K_value, weights='uniform', algorithm='auto')
    neigh.fit(X_train, y_train)
    predict_y = neigh.predict(X_test)
    print ("K-NN Accuracy is ", accuracy_score(y_test,predict_y)*100,"% for K-Value:",K_value)
```

```
K-NN Accuracy is  74.37641723356009 % for K-Value: 1
K-NN Accuracy is  82.99319727891157 % for K-Value: 2
K-NN Accuracy is  78.45804988662131 % for K-Value: 3
K-NN Accuracy is  83.6734693877551 % for K-Value: 4
K-NN Accuracy is  82.76643990929705 % for K-Value: 5
K-NN Accuracy is  84.12698412698413 % for K-Value: 6
K-NN Accuracy is  83.6734693877551 % for K-Value: 7
K-NN Accuracy is  85.03401360544217 % for K-Value: 8
K-NN Accuracy is  83.6734693877551 % for K-Value: 9
K-NN Accuracy is  84.5804988662131 5 % for K-Value: 10
K-NN Accuracy is  84.12698412698413 % for K-Value: 11
K-NN Accuracy is  85.26077097505669 % for K-Value: 12
K-NN Accuracy is  85.26077097505669 % for K-Value: 13
K-NN Accuracy is  85.26077097505669 % for K-Value: 14
K-NN Accuracy is  85.4875283446712 % for K-Value: 15
K-NN Accuracy is  85.71428571428571 % for K-Value: 16
K-NN Accuracy is  85.4875283446712 % for K-Value: 17
K-NN Accuracy is  85.94104308390023 % for K-Value: 18
K-NN Accuracy is  85.94104308390023 % for K-Value: 19
K-NN Accuracy is  85.71428571428571 % for K-Value: 20
K-NN Accuracy is  85.71428571428571 % for K-Value: 21
K-NN Accuracy is  85.71428571428571 % for K-Value: 22
K-NN Accuracy is  85.71428571428571 % for K-Value: 23
K-NN Accuracy is  85.71428571428571 % for K-Value: 24
K-NN Accuracy is  85.71428571428571 % for K-Value: 25
```

Further analysis and coding

After getting prediction from each algo, we have to convert it into one format as some are in words and some in number, as shown

```
print(rf_pred)
```

```
[0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

```
print(y_pred)
```

```
['No' 'No' 'Yes' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'Yes' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'Yes' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'Yes' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'Yes' 'No'
 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No'
 'No' 'No' 'No' 'Yes' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'No' 'Yes'
```

and as we need some way to connect it, so we will create a data set which have training dataset and the prediction dataset combined and the prediction index will be test index as it gives us a connection point, and combining it into one dataset, and we get

attr - DataFrame

Index	GB	RF	KNN
14	1	1	1
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	1	1	1
22	0	0	0
23	0	0	1
24	1	1	1
25	0	0	0
26	1	1	1

Code:

```
#Creating DataFrame for Gradient Boosting Classifier (pred + train)

gbpr = pandas.DataFrame(y_pred)
gbpr.index = y_test.index           # Changing the prediction index with y_test index to link the emp

gbrs = pandas.concat([gbpr , y_train])  # Merging the whole Dataset back
gbrs = gbrs.sort_index()                # Sorting the whole Dataset

#Creating DataFrame for Random Forest Classifier (pred + train)

rfpr = pandas.Series(rf_pred)                            #Converting into Series
rfpr = rfpr.map({ 1 : 'Yes', 0 : 'No'}).astype(str)   #Converting 0 and 1 to Yes and No

rfrs = pandas.DataFrame(rfpr)
rfrs.index = y_test.index

rfrs = pandas.concat([rfrs , y_train])             # Merging the whole Dataset back
rfrs = rfrs.sort_index()                           # Sorting the whole Dataset


#Creating DataFrame for K-NN (pred + train)

knnpr = pandas.DataFrame(predict_y)
knnpr.index = y_test.index           # Changing the prediction index with y_test index to link the emp

knnrs = pandas.concat([knnpr , y_train])  # Merging the whole Dataset back
knnrs = knnrs.sort_index()                # Sorting the whole Dataset
```

```
gbrs = pandas.Series(gbrs[0].values)
rfrs = pandas.Series(rfrs[0].values)
knnrs = pandas.Series(knnrs[0].values)

gbrs = gbrs.map({'Yes': 1, 'No': 0}).astype(int)
rfrs = rfrs.map({'Yes': 1, 'No': 0}).astype(int)
knnrs = knnrs.map({'Yes': 1, 'No': 0}).astype(int)

#Creating a DataFrame with all Predictions
attr = (gbrs , rfrs , knnrs)
attr = pandas.concat(attr, axis=1)
attr=attr.set_axis(['GB', 'RF', 'KNN'], axis=1)  # Axis 0 (Row) , 1(Column)
```

Creating an average of prediction

Next task is to calculate the mean in the dataset column wise as the models are not 100% accuracy so by doing this we will be able to get a more accurate result in which we can rely on , here we will have 4 following values

- 0 (All 3 are zero)
- 0.33 (One of them was one)
- 0.66 (Two of them were one)
- 1 (All of them were one)

But we need either 0 or 1, no or yes, therefore we will add a condition filter which would be 0.5, as less then 0.5 will be zero, and above will be one, then adding average into dataset we get

attr - DataFrame

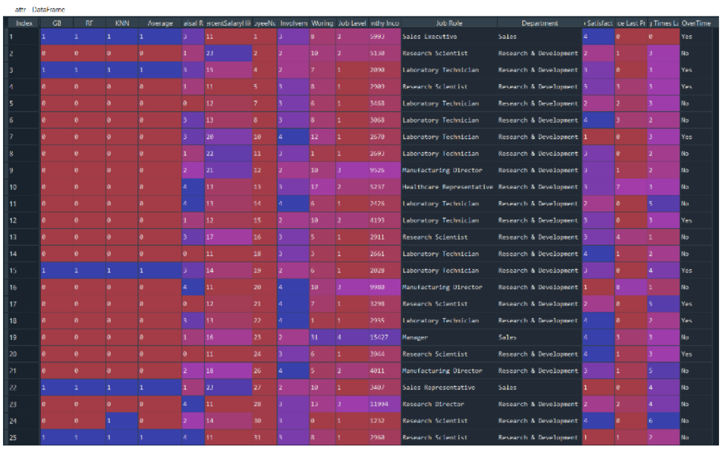
Index	GB	RF	KNN	Average
18	0	0	0	0
19	0	0	0	0
20	0	0	0	0
21	0	0	0	0
22	1	1	1	1
23	0	0	0	0
24	0	0	1	0
25	1	1	1	1
26	0	0	0	0
27	1	1	1	1
28	0	0	0	0
29	0	0	0	0

Code:

```
#Taking the average of all the models prediction
mn= attr.mean(axis = 1 )

#Converting decimals into 1 and 0
for i in range(1,len(mn)-1):
    if mn[i] < 0.5 : mn[i]=0
    else : mn[i]=1

#Adding Average to DataFrame
attr = (gbrs , rfrs , knnrs, mn)
attr = pandas.concat(attr, axis=1)
attr=attr.set_axis(['GB', 'RF', 'KNN' , 'Average'])
```

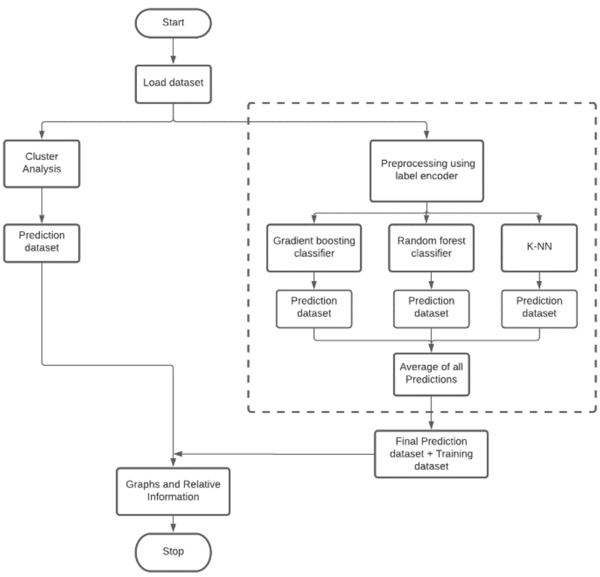
Creating a Dataset with all the relative Info

In the final dataset, we add all the relative info a person taking decision can look and easily see all the details he/she might require to make that decision.



Code:

```
#Creating a final DataFrame

data['Department']=le.inverse_transform(data['Department'])  # Converting Number back to labels

attr = (gbrs , rfrs , knnrs, mn ,data.AppraisalRating, data.PercentSalaryHike , data.EmployeeNumber
        , data.JobInvolvement , data.TotalWorkingYears , data.JobLevel ,data.MonthlyIncome ,data.JobRole
        ,data.Department , data.JobSatisfaction , data.YearsSinceLastPromotion , data.TrainingTimesLastYear , data.OverTime)

attr = pandas.concat(attr, axis=1)
attr=attr.set_axis(['GB', 'RF', 'KNN' , 'Average' , 'Appraisal Rating', 'PercentSalaryHike', 'EmployeeNumber'
                  , 'Job Involvement ', 'Total Woring Years' , 'Job Level' , 'Monthy Income' , 'Job Role'
                    , 'Department' ,'Job Satisfaction' ,'Years Since Last Promotion' , 'Training Times Last Year'
                     , 'OverTime'], axis=1)  # Axis 0 (Row) , 1(Column)

attr = attr.sort_index()                # Sorting the whole Dataset
attr.index=attr.index+1                 # Starting value was 0 before this
attr.index.name= 'Index'
```

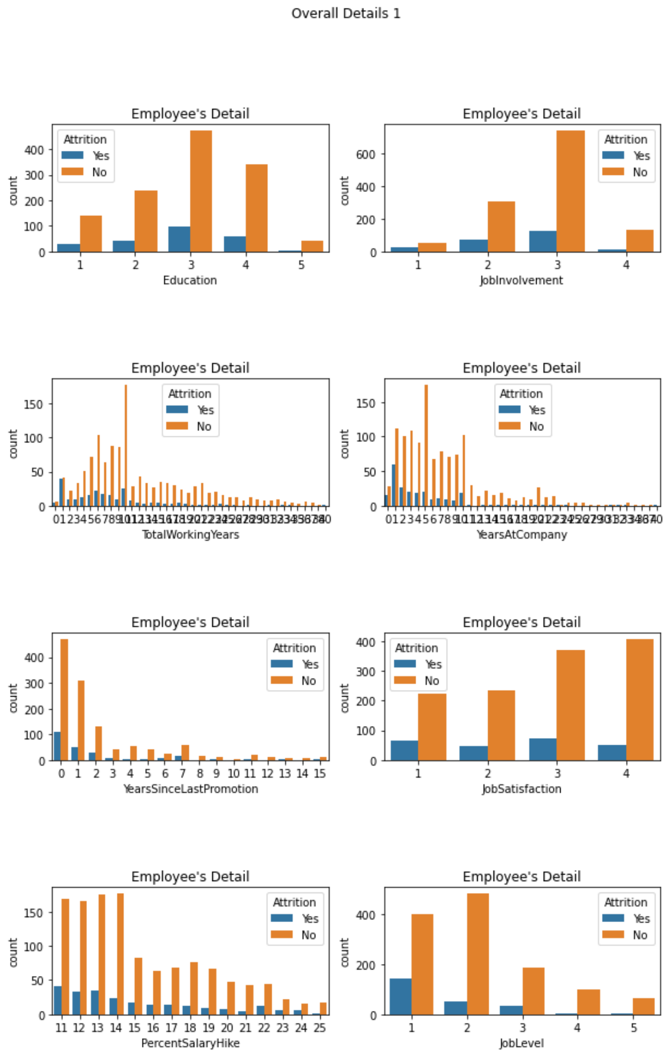


Graphs

For graphs we have three perspectives Overall, Manager and Hr.

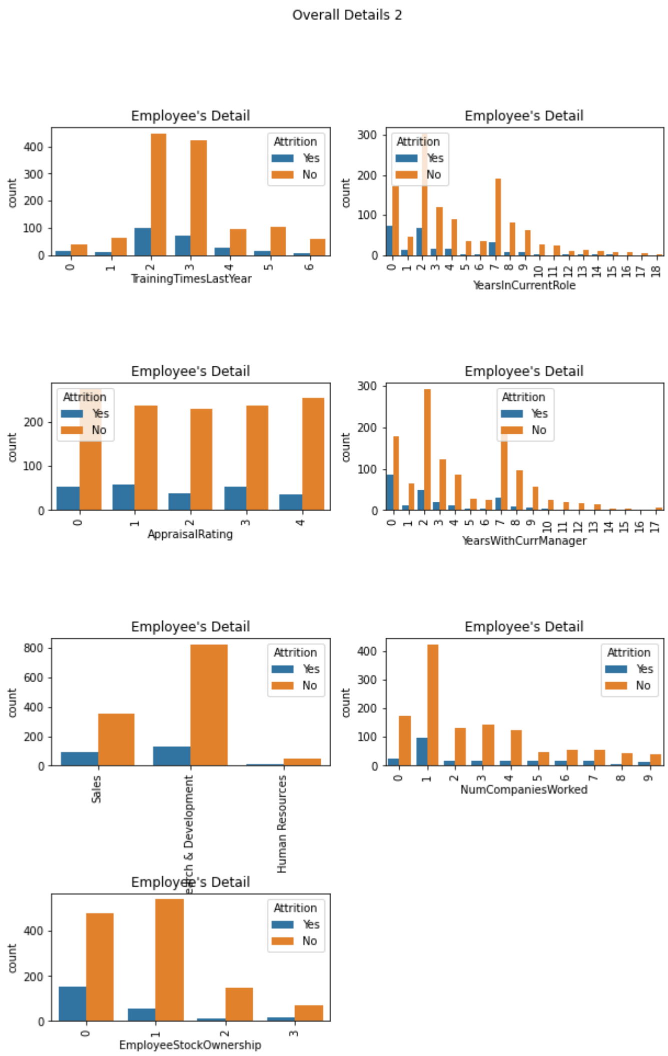
1. Overall

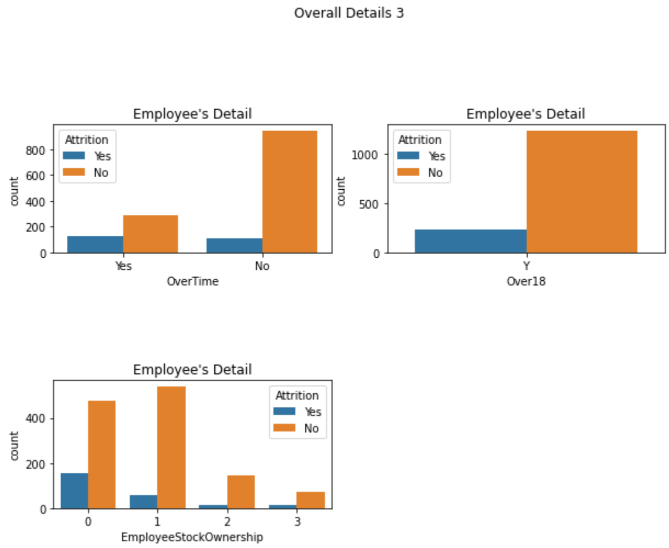
It this we have the count number with employee details



Code:

```
# Overall

features=['Education', 'JobInvolvement','TotalWorkingYears','YearsAtCompany'
          ,'YearsSinceLastPromotion','JobSatisfaction','PercentSalaryHike','JobLevel']

fig=plt.subplots(figsize=(10,15))
for i, j in enumerate(features):
    plt.subplot(4, 2, i+1)
    plt.subplots_adjust(hspace = 1.0)
    sns.countplot(x=j,data = data, hue='Attrition')
    plt.title("Employee's Detail")
    plt.suptitle('Overall Details 1')
```





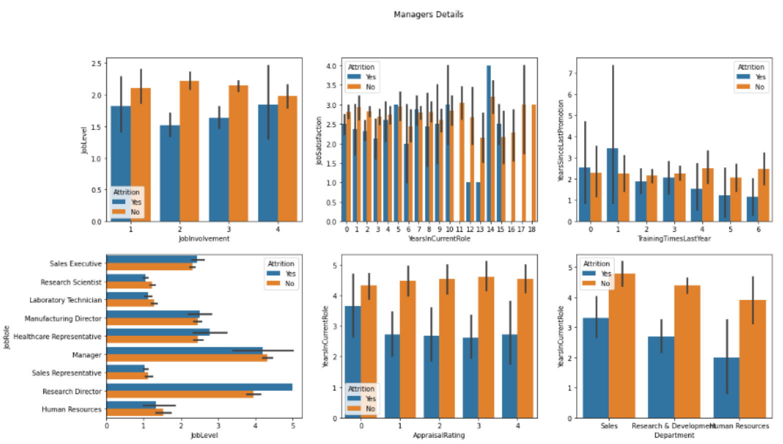
Code:

```
# Overall

features=['TrainingTimesLastYear','YearsInCurrentRole','AppraisalRating','YearsWithCurrManager'
         ,'Department','NumCompaniesWorked','EmployeeStockOwnership']

fig=plt.subplots(figsize=(10,15))
for i, j in enumerate(features):
    plt.subplot(4, 2, i+1)
    plt.subplots_adjust(hspace = 1.0)
    sns.countplot(x=j,data = data, hue='Attrition')
    plt.xticks(rotation=90)
    plt.title("Employee's Detail")
    plt.suptitle('Overall Details 2')
```

Code:

```
# Overall

features=['OverTime','Over18','EmployeeStockOwnership']

fig=plt.subplots(figsize=(10,15))
for i, j in enumerate(features):
    plt.subplot(4, 2, i+1)
    plt.subplots_adjust(hspace = 1.0)
    sns.countplot(x=j,data = data, hue='Attrition')
    plt.title("Employee's Detail")
    plt.suptitle('Overall Details 3')
```

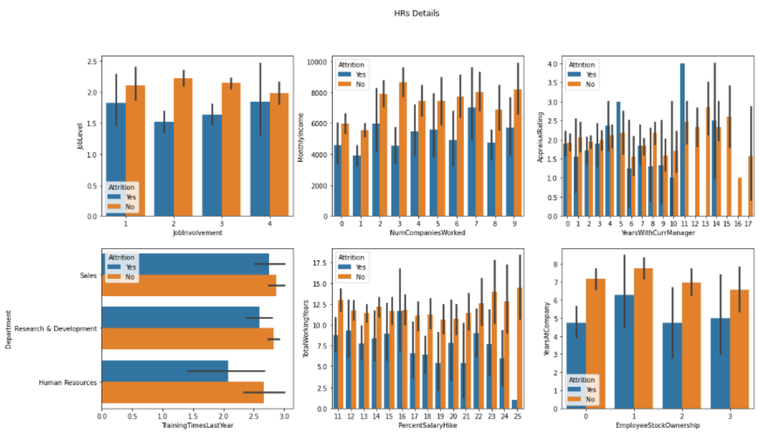
2. Manager's Perspective



Code:

```
# Manager

fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Managers Details')

sns.barplot(ax=axes[0, 0] , x="JobInvolvement", y="JobLevel", hue="Attrition", data=data)
sns.barplot(ax=axes[0, 1] , x="YearsInCurrentRole", y="JobSatisfaction", hue="Attrition", data=data)
sns.barplot(ax=axes[0, 2] , x="TrainingTimesLastYear", y="YearsSinceLastPromotion", hue="Attrition", data=data)
sns.barplot(ax=axes[1, 0] , x="JobLevel", y="JobRole", hue="Attrition", data=data)
sns.barplot(ax=axes[1, 1] , x="AppraisalRating", y="YearsInCurrentRole", hue="Attrition", data=data)
sns.barplot(ax=axes[1, 2] , x="Department", y="YearsInCurrentRole", hue="Attrition", data=data)
```

3. Hr's Perspective



Code:

```
# HR

fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('HRs Details')

sns.barplot(ax=axes[0, 0] , x="JobInvolvement", y="JobLevel", hue="Attrition", data=data)
sns.barplot(ax=axes[0, 1] , x="NumCompaniesWorked", y="MonthlyIncome", hue="Attrition", data=data)
sns.barplot(ax=axes[0, 2] , x="YearsWithCurrManager", y="AppraisalRating", hue="Attrition", data=data)
sns.barplot(ax=axes[1, 0] , x="TrainingTimesLastYear", y="Department", hue="Attrition", data=data)
sns.barplot(ax=axes[1, 1] , x="PercentSalaryHike", y="TotalWorkingYears", hue="Attrition", data=data)
sns.barplot(ax=axes[1, 2] , x="EmployeeStockOwnership", y="YearsAtCompany", hue="Attrition", data=data)
```

Conclusion

The foremost objective of any organisation is to earn profit. But in order to accomplish maximum revenue, the organisation should focus more on employees and the methods to retain them for their long run. It is acknowledged that due to deficiency of growth prospects and salary many employees are under pressure to switch their jobs. We can conclude that if attrition has to be reduced industries should generate some opportunities for the development of their employees within the organisation by implementing new pioneering technologies and effective training programs. Organisations should from time-to-time conduct exit, engagement and cultural assessments to identify the fluctuating anticipations of the critical workforce and take all these inputs to have a complete understanding of the factors influencing retention of employees. The findings denote that employee need adaptable workloads, support and appreciation from their co-workers and management, and opportunities for growth and innovation. Companies should scrutinise their attrition rate and the cost attached to it on consistent basis. It is recommendable to have a transparent functioning system so that every employee can know what he's expected to do in the organisation. It is necessary to point out that there's no universal attrition management solution for every company. Each organisation has to develop its own impetus based on compatibility between organisational and individual goals.

This exploration endeavours to think about a portion of the variables which might be the conceivable explanations behind a representative to leave the association. This examination will encourage the association or administration to ponder promotes on those regions and turn out with inventive/imaginative activity intends to make the representatives feel faithful, great and intriguing work environment. This will unquestionably cut down the whittling down level and in future help administration to lessen cost burned through all through this procedure of enlistment to exit.

REFERENCES

1. https://www.digitalhrtech.com/employee-attrition/
2. https://www.questionpro.com/workforce/employee-churn.html
3. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1012.2947&rep=rep1&type=pdf
4. https://www.mdpi.com/2073-431X/9/4/86
5. https://amitness.com/2020/03/fixmatch-semi-supervised/
6. https://www.sciencedirect.com/topics/medicine-and-dentistry/cluster-analysis
7. https://cloud.google.com/architecture/data-preprocessing-for-ml-with-tf-transform-pt1
8. https://towardsdatascience.com/understanding-random-forest-58381e0602d2
9. https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/
10. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
11. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
12. Lucid chart
13. https://www.researchgate.net/publication/307546344_EMPLOYEE_ATTRITION_AND_STRATEGIC_RETENTION_CHALLENGES_IN_INDIAN_MANUFACTURING_INDUSTRIES_A_CASE_STUDY
14. https://www.researchgate.net/publication/333104347_A_STUDY_ON_ATTRITION_-_TURNOVER_INTENTIONS_OF_EMPLOYEES