TABLE OF CONTENTS

ABSTRACT

We are doing this major project on EMPLOYRR TURNOVER PREDICTION USING MACHINE LEARNING.

Employee turnover is a critical challenge for organizations, impacting productivity, morale, and operational efficiency. This project addresses the prediction of employee turnover using machine learning techniques to enhance workforce management and retention strategies. We employed various machine learning algorithms, including logistic regression, decision trees, random forests, and gradient boosting, to analyze a dataset encompassing employee demographics, job roles, performance metrics, and historical turnover data. Our objective was to develop a predictive model capable of identifying employees at risk of leaving the organization and providing actionable insights for preemptive interventions.

The dataset was pre-processed and feature-engineered to improve model performance. We utilized techniques such as normalization, encoding categorical variables, and handling missing data. Model performance was evaluated using metrics such as accuracy, precision, recall, and the F1 score. The best-performing model demonstrated significant predictive power, with high accuracy and a low false positive rate, indicating its effectiveness in identifying potential turnover.

Our findings underscore the importance of integrating machine learning into human resource strategies. The predictive model not only aids in forecasting turnover but also highlights key factors influencing employee decisions to leave. This approach allows organizations to proactively address issues related to job satisfaction and retention, ultimately leading to a more stable and engaged workforce. The insights gained from this project provide a foundation for developing targeted retention programs and improving overall organizational performance.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 DOMAIN DESCRIPTION

What is Machine Learning ?

Machine learning is a branch of artificial intelligence (AI) that focuses on creating systems capable of learning from data and improving their performance over time without being explicitly programmed for every specific task. Instead of being given specific instructions, a machine learning model is trained on large datasets, learns patterns and relationships within that data, and then makes predictions or decisions based on what it has learned.

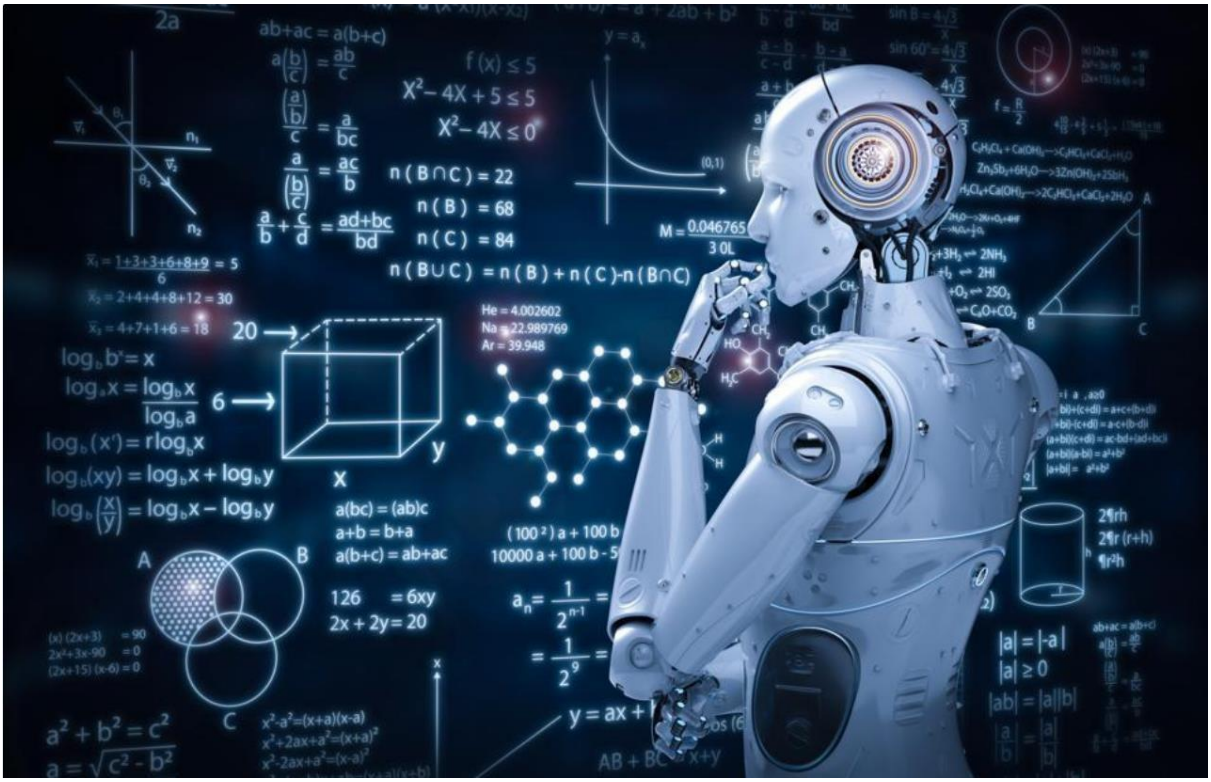


Fig 1.1.1 Image for Machine Learning

Here's a brief overview of how it works:

1. Data Collection: Gather data relevant to the problem you want to solve. This could be anything from images and text to numerical values and sensor readings.

2. Data Preparation: Clean and preprocess the data to ensure it's in a suitable format for analysis. This might involve handling missing values, normalizing data, or converting categorical variables.

3. Model Selection: Choose a machine learning algorithm or model that fits the problem you're working on. There are various types of models, including decision trees, neural networks, and support vector machines.

4. Training: Feed the prepared data into the model. During training, the model learns to recognize patterns and make predictions by adjusting its parameters to minimize errors.

5. Evaluation: Test the model on new, unseen data to evaluate its performance. This helps determine how well the model generalizes to real-world scenarios.

6. Hyperparameter Tuning: Fine-tune the model's settings to improve its performance based on the evaluation results.

7. Deployment: Once the model performs satisfactorily, it can be deployed in real-world applications where it can make predictions or decisions based on new data.

8. Monitoring and Maintenance: Continuously monitor the model's performance and update it as necessary to adapt to new data or changes in the environment.

- **Some machine learning methods:**

Machine learning is used in a variety of applications, such as recommendation systems (like those used by Netflix or Amazon), image and speech recognition, fraud detection, and even autonomous vehicles. It's a powerful tool for making sense of large amounts of data and finding patterns that might be too complex for humans to discern.

Machine learning methods can be broadly categorized based on the type of problem they are designed to solve and how they learn from data. Here's a rundown of the main methods and their subcategories:

1. **Supervised Learning**

Supervised learning involves training a model on a labeled dataset, where the correct output is provided for each input. The goal is for the model to learn a mapping from inputs to outputs that can be used to make predictions on new, unseen data.

Common Methods:

Regression: Predicts continuous values. For example, predicting house prices based on features like size and location.

Linear Regression: Models the relationship between variables using a straight line.

Polynomial Regression: Fits a polynomial equation to the data to capture non-linear relationships.

Classification: Predicts categorical labels. For example, classifying emails as spam or not spam.

Logistic Regression: Models the probability of a binary outcome.

Support Vector Machines (SVM): Finds the optimal boundary (hyperplane) that separates different classes in the feature space.

Decision Trees: Uses a tree-like model of decisions to classify data.

Random Forests: An ensemble method that uses multiple decision trees to improve classification performance.

Neural Networks: Consists of interconnected nodes (neurons) organized in layers, capable of learning complex patterns.

2. Unsupervised Learning

Unsupervised learning involves training a model on data that does not have labeled responses. The goal is to find hidden patterns or intrinsic structures in the data.

Common Methods:

Clustering: Groups data into clusters based on similarity. For example, segmenting customers into different groups for targeted marketing.

K-Means Clustering: Partitions data into k clusters by minimizing the variance within each cluster.

Hierarchical Clustering: Builds a tree of clusters by either merging or splitting them based on distance measures.

Dimensionality Reduction: Reduces the number of features while preserving as much information as possible. Useful for visualizing high-dimensional data.

Principal Component Analysis (PCA): Transforms data into a set of orthogonal components that capture the most variance.

t-Distributed Stochastic Neighbor Embedding (t-SNE): Reduces dimensions while preserving the local structure of data.

Association Rule Learning: Identifies rules that describe how features in the data are related. Commonly used in market basket analysis.

Apriori Algorithm: Finds frequent itemsets and generates association rules.

3. Semi-Supervised Learning

Semi-supervised learning uses a combination of labeled and unlabeled data to improve learning accuracy. It is particularly useful when acquiring labeled data is expensive or time-consuming.

Common Methods:

Self-Training: The model initially trained on labeled data is used to label unlabeled data, which is then added to the training set for further refinement.

Co-Training: Uses multiple models trained on different views of the data to label unlabeled data and improve each other's performance.

4. Reinforcement Learning

Reinforcement learning involves training an agent to make decisions by rewarding or penalizing it based on its actions in an environment. The goal is to learn a policy that maximizes cumulative reward over time.

Common Methods:

Q-Learning: An off-policy algorithm that learns the value of actions in particular states to determine the best action to take.

Deep Q-Networks (DQN): Combines Q-learning with deep neural networks to handle complex state spaces.

Policy Gradients: Directly optimizes the policy that the agent uses to make decisions, often using neural networks.

5. Self-Supervised Learning

Self-supervised learning is a subset of unsupervised learning where the system learns to predict part of the data from other parts of the data. This method generates labels from the data itself, often used in tasks like pre-training language models.

Common Methods:

Contrastive Learning: Learns to distinguish between similar and dissimilar pairs of data, often used in image and text representations.

6. Transfer Learning

Transfer learning involves using a pre-trained model on one task and adapting it to a different but related task. This is useful when you have limited data for the new task but a lot of data for the original task.

Common Methods:

Fine-Tuning: Adjusting the weights of a pre-trained model on a new, smaller dataset to adapt it to the new task.

Each method has its strengths and is chosen based on the nature of the problem, the data available, and the specific requirements of the task at hand.

What is Artificial Intelligence?

Artificial Intelligence (AI) refers to the field of computer science focused on creating systems or machines that can perform tasks typically requiring human intelligence. These tasks can include learning from experience, understanding natural language, recognizing patterns, solving problems, and making decisions.

Fig 1.1.2 Image for Artificial Intelligence

AI encompasses a variety of techniques and technologies, including:

1. Machine Learning (ML): A subset of AI where algorithms improve through experience. For example, a machine learning model might learn to recognize cats in images after being trained on many examples.

2. Natural Language Processing (NLP): This involves the ability of a machine to understand, interpret, and generate human language. Applications include chatbots and language translation services.

3. Computer Vision: This field enables machines to interpret and make decisions based on visual data, like identifying objects in an image or video.

4. Robotics: The integration of AI with physical machines to perform tasks in the real world, such as autonomous vehicles or industrial robots.

5. Expert Systems: These are AI programs designed to mimic the decision-making abilities of a human expert in a specific domain, like medical diagnosis or financial forecasting.

6. General AI: This is a more ambitious goal of creating machines with the ability to understand, learn, and apply intelligence across a broad range of tasks, similar to human cognitive abilities. It's also referred to as Artificial General Intelligence (AGI).

AI technologies are increasingly integrated into everyday applications, from virtual assistants like Siri and Alexa to recommendation systems on streaming services.

Some Machine Learning Examples in real life:

1. Image recognition

Image recognition is another machine learning technique that appears in our day-to-day life. With the use of ML, programs can identify an object or person in an image based on the intensity of the pixels. This type of facial recognition is used for password protection methods like Face ID and in law enforcement. By filtering through a database of people to identify commonalities and matching them to faces, police officers and investigators can narrow down a list of crime suspects.

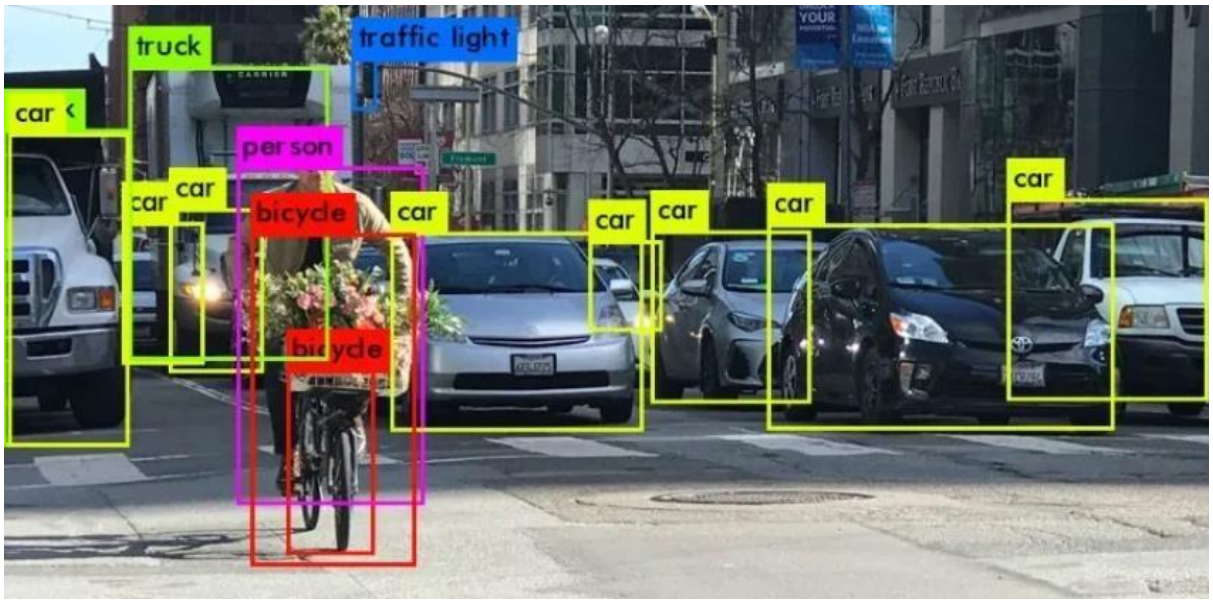


Fig 1.1.3 Image for Image recognition

2. Virtual personal assistants

Virtual personal assistants are devices you might have in your own homes, such as Amazon's Alexa, Google Home, or the Apple iPhone's Siri. These devices use a combination of speech recognition technology and machine learning to capture data on what you're requesting and how often the device is accurate in its delivery. They detect when you start speaking, what you're saying, and deliver on the command. For example, when you say, "Siri, what is the weather like today?", Siri searches the web for weather forecasts in your location and provides detailed information.

Fig 1.1.4 Image for Virtual personal assistants

3. Traffic predictions

When you use Google Maps to map your commute to work or a new restaurant in town, it provides an estimated time of arrival. Google uses machine learning to build models of how long trips will take based on historical traffic data (gleaned from satellites). It then takes that data based on your current trip and traffic levels to predict the best route according to these factor**.**

Fig. 2. Traffic Prediction of Less/No Traffic

Fig. 3. Traffic Prediction of Medium Traffic

Fig. 4. Traffic Prediction of Heavy Traffic

Fig 1.1.5 Image for Traffic predictions

4. Self-driving car technology

A frequently used type of machine learning is reinforcement learning, which is used to power self-driving car technology. Self-driving vehicle company Waymo uses machine learning sensors to collect data of the car's surrounding environment in real time. This data helps guide the car's response in different situations, whether it is a human crossing the street, a red light, or another car on the highway.

Fig 1.1.6 Image for Self-driving car technology

Artificial Intelligence Examples in real life:

1. Healthcare

- Medical Imaging: AI algorithms assist radiologists by analyzing X-rays, MRIs, and CT scans to detect anomalies such as tumors or fractures with high accuracy. Examples include IBM’s Watson Health and Google's DeepMind Health.
- Predictive Analytics: AI models predict patient outcomes and disease progression, helping in early intervention. For example, AI tools can predict the likelihood of diseases like diabetes or heart conditions based on patient data.

2. Finance

- Fraud Detection: AI systems monitor transactions in real-time to detect and prevent fraudulent activities. For instance, credit card companies use machine learning algorithms to identify unusual spending patterns.

- Algorithmic Trading: AI algorithms analyze market trends and execute trades at high speeds to maximize returns. Companies like Renaissance Technologies and Two Sigma use AI for quantitative trading strategies.

3. Retail

- Recommendation Systems: Platforms like Amazon and Netflix use AI to suggest products or content based on user behavior and preferences. This is done through collaborative filtering and content-based filtering techniques.
- Inventory Management: AI helps optimize stock levels and forecast demand, reducing overstock and stockouts. Companies like Walmart use AI to manage inventory and supply chains more efficiently.

4. Transportation

- Autonomous Vehicles: Companies like Tesla and Waymo develop self-driving cars using AI to navigate roads, recognize obstacles, and make driving decisions. These systems use sensors, cameras, and machine learning algorithms.
- Traffic Management: AI systems analyze traffic patterns to optimize signal timings and reduce congestion. Cities use AI for smart traffic management systems to improve flow and safety.

1.2 ABOUT PROJECT

Employee turnover is one of the most significant problems an organization can encounter throughout its lifecycle, as it is difficult to predict and often introduces noticeable voids in an organization's skilled workforce. Service firms recognize that the timely delivery of their services can become compromised, overall firm produc tivity can decrease significantly and, consequently, customer loyalty can decline when employees leave unexpectedly. As a result, it is imperative that organizations formulate proper recruitment, acquisition and retention strategies and implement effective mechanisms to prevent and diminish employee turnover, while understanding its underlying, root causes. Most recently, the prevalence of intelligent machine learning algorithms in the field of computer science has led to the development of robust quantitative methods to derive insights from industry data. Supervised machine learning methods—wherein computers learn from analyses of large-scale, historical, labelled datasets—have been shown to garner insights in various fields, like biology and medical sciences , transportation, political science, as well as many other fields. Owing to the advancements in information technology, researchers have also studied numerous machine learning approaches to improve the outcomes of human resource (HR) man agement. A detailed listing of recent studies in using supervised machine learning on employee turnover is described in Table 1, and lists the data included and related machine learning algorithms that were used therein, including decision tree (DT) methods, random forest (RF) methods, gradient boosting trees (GBT) methods, extreme gradient boosting (XGB), logistic regression (LR), support vector machines (SVM), neural networks (NN), linear discriminant analysis (LDA), Naïve Bayes (NB)

methods, K-nearest neighbor (KNN), Bayesian networks (BN) and induction rule methods (IND). The performance evaluation of machine learning algorithms has also been studied previously by various researchers. Notably, Punnoose and Ajit compared the predictive capabilities of seven different machine learning algorithms, including recently developed algorithms, like Extreme Gradient Boosting, on employee turnover. Similarly, Sikaroudi and co-researchers conducted simulations to predict employee turnover using ten different data mining algorithms, including tests on various types of neural networks and induction rule methods. In addition to placing focus on classification and prediction ability, many researchers have also made substantial efforts to better understand which features (e.g. compensation, age, work experience, etc.) are most influential in predicting employee turnover. These features seldom carry equal value in data mining applications, so it is useful to gain a better understanding of their importance. For instance, many of the studies using tree-based quantified feature importance by calculating the impurity reduction by node split in decision trees. Moreover, modified genetic algorithms and sensitivity analysis have been used to under stand relative feature importance as well. Numerous studies have also generated classification rules or visualized the classification procedure to provide further insight and confidence in using machine learning methods. Despite the breadth of research outcomes mentioned above, the findings for pre dicting employee turnover that stem from using machine learning methods are often problem-specific and difficult to generalize. First and foremost, this is primarily because HR data is confidential, which inherently impedes conducting in-depth analyses on multiple datasets. In addition, HR data is often noisy, inconsistent and contains missing information, a problem that is exacerbated by the small pro portion of employee turnover that typically exists within a given set of HR data. Secondly, gaps tend to persist in model performance evaluation. Specifically, previous research on the assessment of machine learning algorithms has generally focused on a narrow evaluation of metrics across various models.

Accuracy has traditionally been selected as the primary evaluation standard for this problem, but this approach is questionable as accuracy measures are not reliable for imbalanced datasets. As the proportion of people who leave an organi zation is generally much smaller than that of those who stay, there is often a risk of computing misleadingly high accuracy correlations. The deficiency of the analysis is often made worse by the limited use of statistical instruments, often only opting for relatively simple comparisons instead. Thirdly, the attempt to improve the model interpretability by ranking feature importance and visualizing classifier rules should be executed cautiously. The analysis of feature importance in several studies could be biased as it takes classifier-dependent approaches, where model per formance matters. For instance, some works use decision trees to calculate the feature importance as part of the model building process. However, if decision trees do not perform well, the corresponding feature importance result may be inaccurate. With the assumption that decision trees perform well, visualizing their classification rules could improve the model interpretability. However, decision trees come with high variance and low stability, resulting in precarious model interpretation with a small change in data. The aim of this paper is to provide a comprehensive description, demonstration and assessment of supervised machine learning approaches for the prediction of employee turnover within organizations of varying size. In the present study, ten supervised machine learning methods are evaluated for organizations of small-, medium- and large-sized

populations. Details of each supervised machine learning method are given and the benefits, capabilities and performance of each are provided in the context of predicting employee turnover. The effect of data size and data type, and how to get reliable feature importance and data visualization are also discussed. Lastly, general guidelines are provided on the selection, use and interpretation of these ten supervised machine learning methods for reliable analysis of HR datasets of varying size and complexity.

1.3 OBJECTIVE

The objective of an employee turnover prediction project is to develop a system that can accurately forecast which employees are likely to leave an organization. By analyzing historical data, the project aims to identify key factors contributing to turnover and build predictive models to pinpoint employees at high risk of leaving. This allows the organization to take proactive measures, such as improving retention strategies, addressing common issues, and reallocating resources effectively. The ultimate goal is to reduce turnover rates, minimize associated costs, and enhance overall employee satisfaction and engagement.

CHAPTER 2

EMPLOYEE TURNOVER PREDICTION SURVEY

2.1 THEOROTICAL BACKGROUND

This project Employee Turnover Prediction addresses the problem of employee turnover in organizations, based on survey data it involves understanding how employee perceptions and experiences, as captured through surveys, influence their likelihood of leaving an organization. Surveys typically collect data on factors such as job satisfaction, organizational commitment, work environment, compensation, and management practices. Theories like the Herzberg's Two-Factor Theory highlight that factors such as job satisfaction and motivation influence employee retention, while the Job Demands-Resources Model emphasizes the balance between job demands and resources as a determinant of employee well-being and turnover intentions.

Predictive models use survey responses to identify patterns and correlations that indicate potential turnover risks. Techniques from statistics and machine learning analyze these survey responses to predict which employees are likely to leave. By integrating survey data into predictive analytics, organizations can better understand the drivers of turnover, assess the effectiveness of retention strategies, and make data-driven decisions to improve employee engagement and reduce turnover rates.

2.2 EXISTING SYSTEM WITH DRAWBACKS

Existing employee turnover prediction systems often face several drawbacks. Many rely on historical data and statistical models that may not account for recent changes in the workplace environment or emerging trends. These systems can be limited in their ability to adapt to new patterns of employee behavior or changes in job market dynamics. They may also suffer from data quality issues, such as incomplete or outdated information, which can skew predictions. Additionally, these systems might not integrate well with other human resource management tools, leading to fragmented insights. Furthermore, they often fail to consider individual employee circumstances or external factors affecting turnover, resulting in a one-size-fits-all approach that lacks personalization. Overall, the effectiveness of these systems is frequently hindered by their reliance on static data and inflexible algorithms, making it challenging to accurately predict and address turnover on a case-by-case basis.

2.3 PROPOSED SYSTEM WITH FEATURES

In the proposed system for employee turnover prediction, the key features are designed to enhance accuracy, adaptability, and usability. This system leverages advanced technologies and methodologies to address the limitations of existing models.

1. **Real-Time Data Integration**: The system incorporates real-time data from various sources, including employee surveys, performance reviews, and market trends, to ensure predictions reflect the current environment and emerging trends.

2. **Advanced Machine Learning Algorithms**: Utilizes sophisticated machine learning models, such as neural networks and ensemble methods, to analyze complex patterns and interactions in the data, improving predictive accuracy over traditional statistical methods.
3. **Personalized Analytics**: Provides individualized risk assessments by considering specific employee attributes and circumstances, rather than applying a uniform approach. This personalization helps identify high-risk individuals more accurately.
4. **Dynamic Adaptation**: Features adaptive algorithms that continuously learn and adjust based on new data and changing conditions, enhancing the system's ability to stay relevant and accurate over time.
5. **Integration with HR Systems**: Seamlessly integrates with existing Human Resource Management Systems (HRMS) and other organizational tools to provide a holistic view of employee data and turnover factors.
6. **Predictive and Prescriptive Analytics**: Not only predicts turnover likelihood but also offers actionable recommendations to mitigate risk, such as targeted interventions or changes in management practices.
7. **Visualization and Reporting Tools**: Includes advanced data visualization tools and customizable reporting features to help HR professionals easily interpret results and make informed decisions.
8. **User-Friendly Interface**: Designed with an intuitive interface that allows HR staff to interact with the system effectively, even without advanced technical expertise.
9. **Scenario Analysis and Forecasting**: Provides tools for scenario analysis to explore the impact of different variables and forecast potential turnover under various conditions.
10. **Feedback Mechanism**: Includes a feedback loop where users can provide input on prediction outcomes, helping to refine and improve the system's models and algorithms over time.

These features collectively aim to enhance the precision and relevance of turnover predictions, enabling organizations to proactively address potential issues and retain valuable employees.

2.4 ADVANTAGES OF PROPOSED SYSTEM

1. Enhanced accuracy through advanced machine learning algorithms and real-time data integration.

2. Personalized risk assessments for targeted retention strategies.

3. Adaptability to evolving organizational dynamics and trends.

4. Seamless integration with existing HR systems for comprehensive analysis.

5. Actionable insights and prescriptive analytics for proactive turnover management.

6. Intuitive user interface promoting usability among HR professionals.

7. Cost efficiency by identifying turnover risks early and implementing targeted interventions.

8. Empowered decision-making with clear data-driven insights.

9. Scenario analysis capabilities for forecasting turnover impacts.

10. Continuous improvement through feedback mechanisms for enhanced predictive models.

2.5 FEASIBILITY STUDY

A feasibility study for an employee turnover prediction system involves evaluating several critical factors to determine its viability. The study assesses technical feasibility by examining the availability and compatibility of data sources, ensuring that the necessary infrastructure and technology are in place to support advanced machine learning algorithms and real-time data processing. It also considers economic feasibility by analyzing the costs of implementation and maintenance versus the potential cost savings from reduced turnover and improved retention strategies. Operational feasibility is reviewed by evaluating the system's integration with existing HR processes and its impact on daily operations. Additionally, the study examines the organizational feasibility by ensuring that the system aligns with company goals and that staff are prepared for its use through adequate training and support. Overall, the feasibility study aims to ensure that the proposed system is practical, cost-effective, and beneficial for enhancing employee retention and optimizing HR functions.

A feasibility study for an employee turnover prediction system involves evaluating several key aspects to ensure the project's success:

2.5.1 Technical Feasibility: Assesses whether the current technology infrastructure can support the advanced analytics required for turnover prediction. This includes evaluating the availability of accurate and comprehensive data, the capability of existing software and hardware, and the potential need for additional tools or platforms.

2.5.2 Economic Feasibility: Analyzes the costs involved in developing, implementing, and maintaining the prediction system compared to the expected financial benefits. This includes evaluating costs related to software development, data management, and potential training for HR staff, as well as estimating the potential savings from improved employee retention.

2.5.3 Operational Feasibility: Examines how well the proposed system will integrate with existing HR processes and workflows. It considers the ease of adoption for HR professionals, the system’s impact on day-to-day operations, and the necessary changes to current procedures.

2.5.4 Organizational Feasibility: Evaluates whether the system aligns with the organization's strategic goals and whether there is sufficient support from stakeholders. It also looks at the readiness of the organization to embrace the system, including the need for training and the overall impact on organizational culture.

2.5.5 Legal and Ethical Feasibility: Reviews any legal and ethical considerations related to employee data usage and privacy. Ensures that the system complies with relevant data protection regulations and maintains high standards of confidentiality and security.

By assessing these factors, the feasibility study determines whether the employee turnover prediction system is practical, cost-effective, and capable of delivering meaningful improvements in employee retention and HR management.

CHAPTER 3

SYSTEM ANALYSIS

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

3.1 SPECIFICATION

Functional Requirements

1. **Data Integration**:
    - Ability to integrate with existing HR systems (e.g., HRMS, ERP).
    - Import data from various sources such as employee records, performance reviews, and surveys.
2. **Data Processing and Analysis**:
    - Support for data cleaning, transformation, and preprocessing.
    - Implementation of predictive modeling algorithms (e.g., logistic regression, decision trees, neural networks) to analyze employee turnover risk.
3. **Prediction and Reporting**:
    - Generate predictions of turnover risk for individual employees and overall trends.
    - Provide actionable insights and recommendations for retention strategies.
    - Generate and export reports, including dashboards and visualizations of turnover predictions and related metrics.
4. **User Management**:
    - Role-based access control to ensure appropriate data access levels (e.g., HR staff, managers).
    - Ability to customize user permissions and access based on roles.
5. **User Interface**:
    - Intuitive and user-friendly interface for data input, model management, and results visualization.
    - Dashboard for viewing turnover predictions, risk factors, and recommended actions.
6. **Alerts and Notifications**:
    - Automated alerts and notifications for high-risk employees or critical changes in turnover predictions.
    - Customizable alert settings based on user preferences and organizational needs.

Non-Functional Requirements

1. **Performance**:
    - System should handle large volumes of data and process predictions in a timely manner.
    - Ensure minimal latency in generating predictions and reports.
2. **Scalability**:
    - Ability to scale up to accommodate growing data volumes and user numbers.
    - Support for expanding model complexity and integrating additional data sources.
3. **Reliability and Availability**:
    - High system reliability with minimal downtime.
    - Backup and recovery mechanisms to protect against data loss.
4. **Security**:
    - Ensure data security and privacy, adhering to relevant data protection regulations (e.g., GDPR, CCPA).
    - Implement encryption for data at rest and in transit.
    - Regular security audits and vulnerability assessments.
5. **Usability**:
    - Intuitive design for ease of use by HR professionals with varying levels of technical expertise.
    - Comprehensive user documentation and training materials.
6. **Maintainability**:
    - Easy to update and maintain with clear documentation for system changes and model updates.
    - Support for version control and rollback capabilities.

3.2 SOFTWARE REQUIREMENTS

One of the most difficult tasks is that, the selection of the software, once system requirement is known that is determining whether a particular software package fits the requirements.

PROGRAMMING LANGUAGE	PYTHON
TECHNOLOGY	PYCHARM
OPERATING SYSTEM	WINDOWS 10
BROWSER	GOOGLECHROME

Table 3.2.1SoftwareRequirements

3.3 HARDWARE REQUIREMENTS

The selection of hardware is very important in the existence and proper working of any software. In the selection of hardware, the size and the capacity requirements are also important.

PROCESSOR	INTEL CORE
RAM CAPACITY	4GB
HARDDISK	1TB
I/O	KEYBOARD,MONITER,MOUSE

Table 3.3.1 Hardware Requirements

3.4 MODULE DESCRIPTION

For predicting the literacy rate of India, our project has been divided into following modules:

1. Data Analysis & Pre-processing
2. Model Training &Testing
3. Accuracy Measures
4. Prediction & Visualization

1. Data Analysis & Pre-processing

Data Analysis is done by collecting raw data from different websites.Data pre-processing technique involves transforming raw data into an understandable format. Real world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues. We use pandas module for Data Analysis and pre- processing.

Pandas:

In order to be able to work with the data in Python, we'll need to read the csv file into a Pandas Data Frame. A Data Frame is a way to represent and work with tabular data. Tabular data has rows and columns, just like our csv file.

2. Model Training &Testing

For Literacy rate prediction, we perform “converting into 2D array” and “scaling using normalization” operations on data for further processing. We use fit_transform to center the data in a way that it has 0 mean and 1 standard error. Then, we divide the data into

x_train and y_train. Our model will get the 0-th element from x_train and try to predict the 0-th element from y_train. Finally, we reshape the x_train data to match the requirements for training using keras. Now we need to train our model using the above data.

The algorithm that we have used is Linear Regression.

Linear Regression:

Linear Regression is a machine learning algorithm based on supervised learning. It is a statistical approach for modeling relationship between a dependent variable with a given set of independent variables. Here we refer dependent variables as response and independent variables as features for simplicity.

Simple Linear Regression is an approach for predicting a response using a single feature. It is assumed that the two variables are linearly related. Hence, we try to find a linear function that predicts the response value(y) as accurately as possible as a function of the feature or independent variable(x).

For predicting the employee turnover rate of any given organization, first we need predict the population for that year. Then the predicted population is given as input to the model which predict turnover rate .

Testing:

In testing, now we predict the data. Here we have 2 steps: predict the turnover rate and plot it to compare with the real results.Using fit transform to scale the data and then reshape it for the prediction. Predict the data and rescale the predicted data to match its real values. Then plot real and predicted literacy rate on a graph. Then calculate the accuracy.

We use Sklearn and Numpy python module for Training and testing

Sklearn:

support It features various classification, regression and clustering algorithms including vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy.

Numpy:

Numpy is the core library for scientific computing in Python. It provides a high performance multidimensional array object, and tools for working with these arrays.It is used for Numerical Calculations.

3. Accuracy Measures

The Accuracy of the model is to be evaluated to figure out the correctness of the prediction. The proposed model got 87% Accuracy.

4. Prediction & Visualization

Using the Proposed model prediction is made for coming years. Graphs are used to visualize state wise turnover rate predictions. We use Matplotlib python module for Visualization.

Matplotlib:

It is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib**.**

CHAPTER 4

DESIGN

4.1 BLOCK DIAGRAM

The block diagram is typically used for a higher level, less detailed description aimed more at understanding the overall concepts and less at understanding the details of implementation.

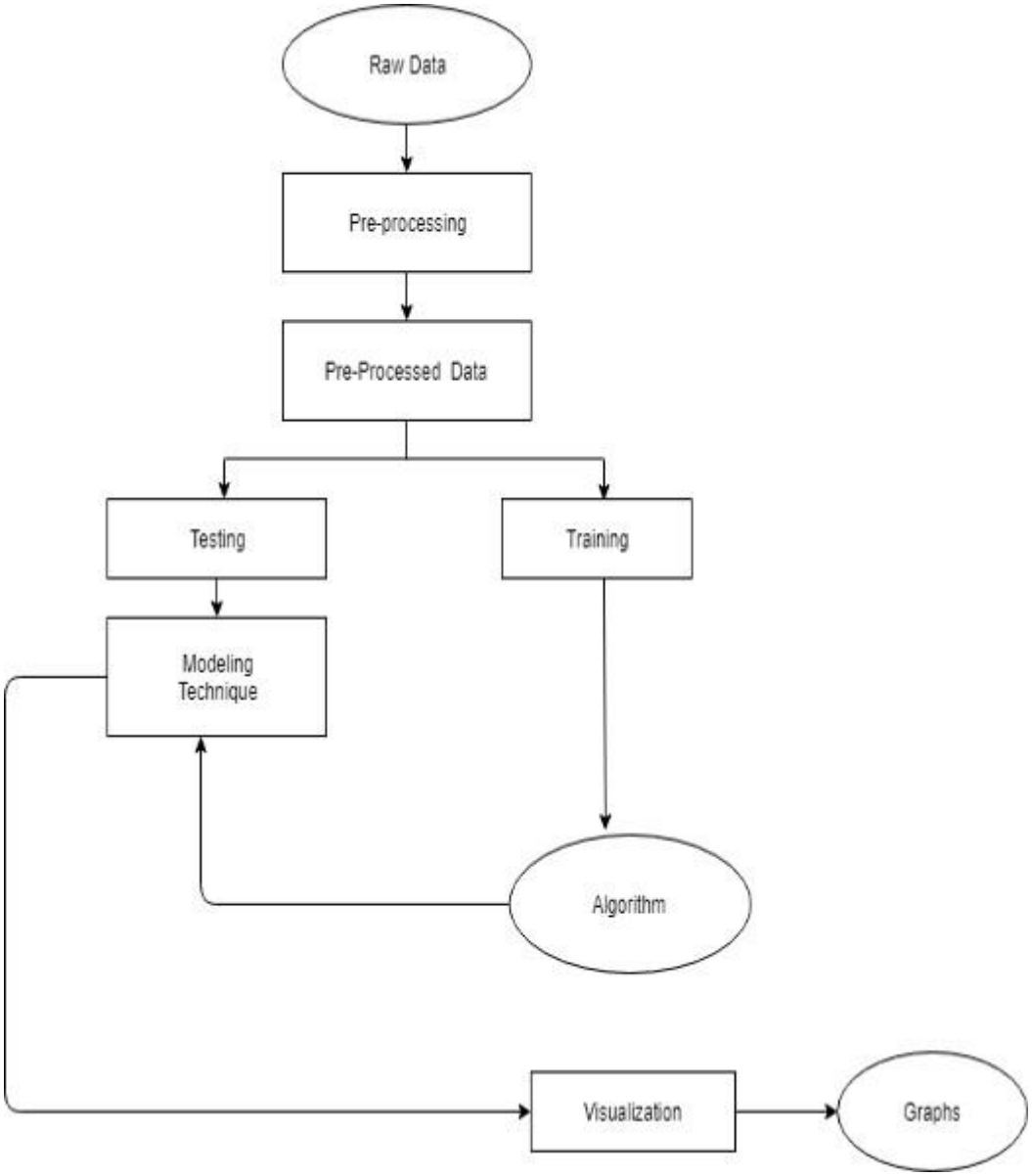


Figure 4.1.1 Block Diagram for Sentimental Analysis

4.2 DATA FLOW DIAGRAMS:

Data flow diagram (DFD) is a graphical representation of "flow" of data through an information system, modelling its process concepts. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFD's can also be used for the visualization of data processing (structured design).

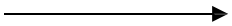
A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It doesn't show information about timing of processes, or information about whether processes will operate in sequence or parallel. A DFD is also called as "bubble chart".

DFD Symbols:

In the DFD, there are four symbols:

- A square define a source or destination of system data.
- An arrow indicates dataflow. It is the pipeline through which the information flows.
- A circle or a bubble represents transforms dataflow into outgoing dataflow.
- An open rectangle is a store, data at reset or at temporary repository of data.

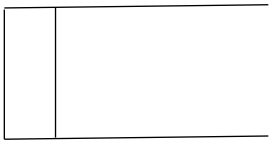
Dataflow: Data move in a specific direction from an origin to a destination.



Process: People, procedures or devices that use or produce (Transform) data. The physical component is not identified.

Sources: External sources or destination of data, which may be programs, organizations or other entity.

Data store: Here data is stored or referenced by a process in the system's #



In our project, we had built the data flow diagrams at the very beginning of business process modelling in order to model the functions that our project has to carry out and the interaction between those functions together with focusing on data exchanges between processes.

4.2.1 Context level DFD:

A Context level Data flow diagram created using select structured systems analysis and design method (SSADM). This level shows the overall context of the system and its operating environment and shows the whole system as just one process. It does not usually show data stores, unless they are "owned" by external systems, e.g. are accessed by but not maintained by this system, however, these are often shown as external entities. The Context level DFD is shown in fig.3.2.1

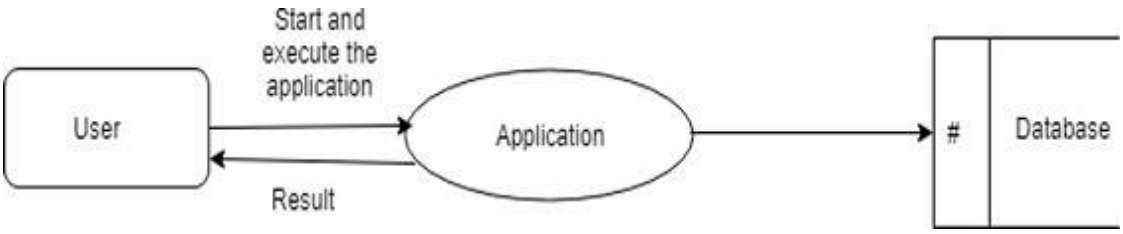


Figure 4.2.1 Context Level DFD for Sentimental Analysis

The Context Level Data Flow Diagram shows the data flow from the application to the database and to the system.

4.2.2 Top level DFD:

A data flow diagram is that which can be used to indicate the clear progress of a business venture. In the process of coming up with a data flow diagram, the level one provides an overview of the major functional areas of the undertaking. After presenting the values for most important fields of discussion, it gives room for level two to be drawn.

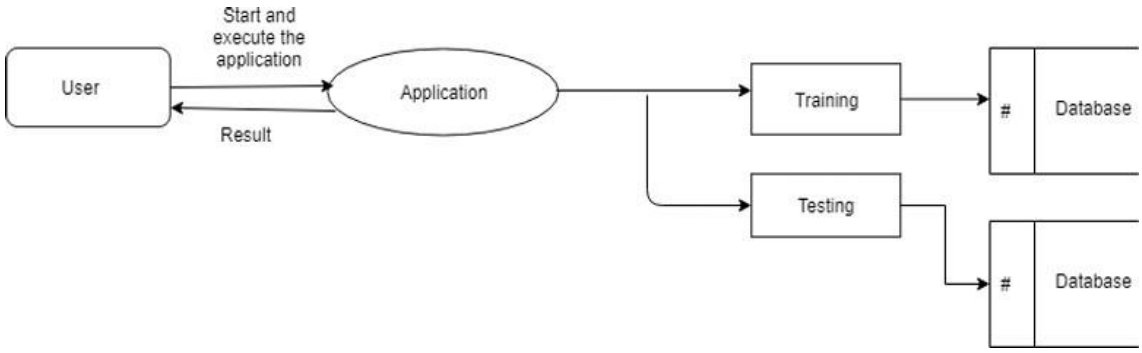


Figure 4.2.2 Top Level DFD for Sentimental Analysis

After starting and executing the application, training and testing the dataset can be done as shown in the above figure

4.2.3 Detailed Level Diagram

This level explains each process of the system in a detailed manner. In first detailed level DFD (Generation of individual fields): how data flows through individual process/fields in it are shown.

In second detailed level DFD (generation of detailed process of the individual fields):

how data flows through the system to form a detailed description of the individual processes.

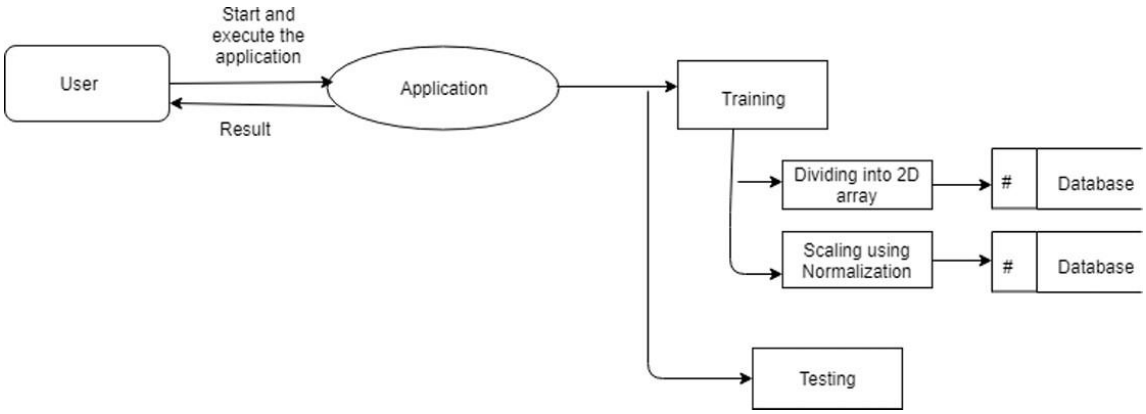


Figure 4.2.3.1 Detailed level DFD for Sentimental Analysis

After starting and executing the application, training the dataset is done by using linear regression and then testing is done.

4.3 UNIFIED MODELLING LANGUAGE DIAGRAMS:

The Unified Modelling Language (UML) is a Standard language for specifying, visualizing, constructing and documenting the software system and its components. The UML focuses on the conceptual and physical representation of the system. It captures the decisions and understandings about systems that must be constructed. A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

- **User Model View**
  i. This view represents the system from the user's perspective.
  ii. The analysis representation describes a usage scenario from the end- users perspective.

- **Structural Model View**
  i. In this model the data and functionality are arrived from inside the system.
  ii. This model view models the static structures.

- **Behavioral Model View**

  It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

- **Implementation model View**

In this the structural and behavioral as parts of the system are represented as they are to be built.

- **Environmental Model View**

  In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

4.3.1 Use Case Diagram:

Use case diagrams are one of the five diagrams in the UML for modeling the dynamic aspects of the systems (activity diagrams, sequence diagram, state chart diagram, collaboration diagram are the four other kinds of diagrams in the UML for modeling the dynamic aspects of systems).Use case diagrams are central to modeling the behavior of the system, a sub-system, or a class. Each one shows a set of use cases and actors and relations.

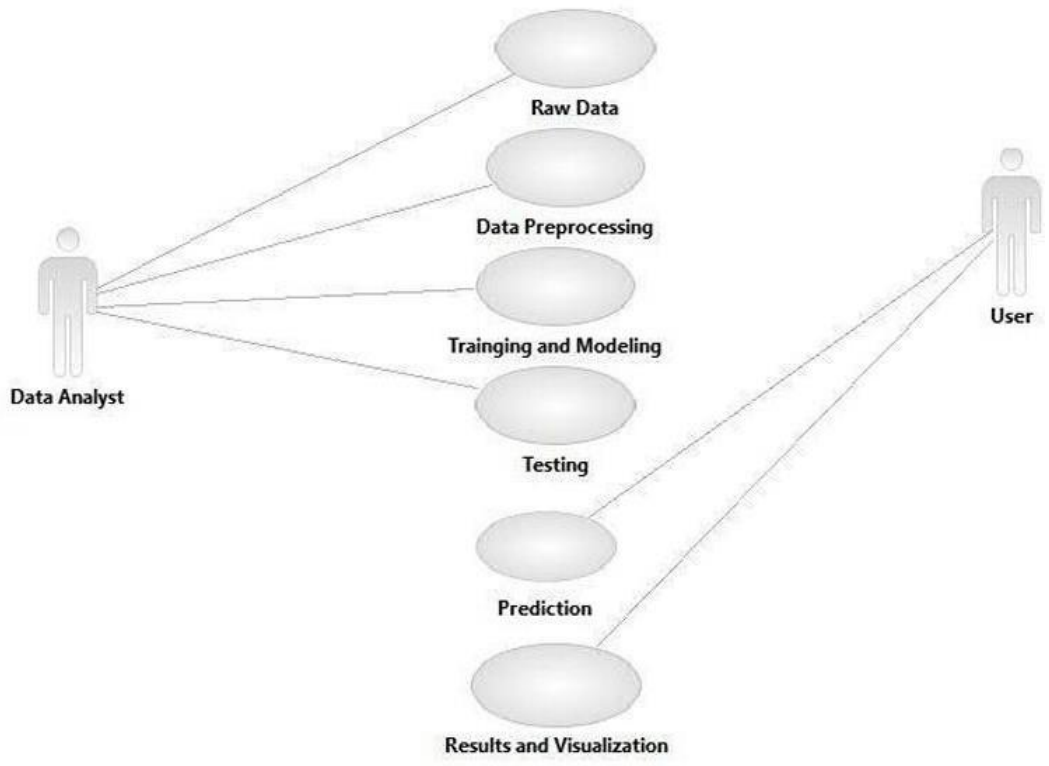


Figure 4.3.1 Usecase Diagram

4.3.2 Sequence Diagram:

Sequence diagram is an interaction diagram which is focuses on the time ordering of messages. It shows a set of objects and messages exchanged between these objects. This diagram illustrates the dynamic view of a system.

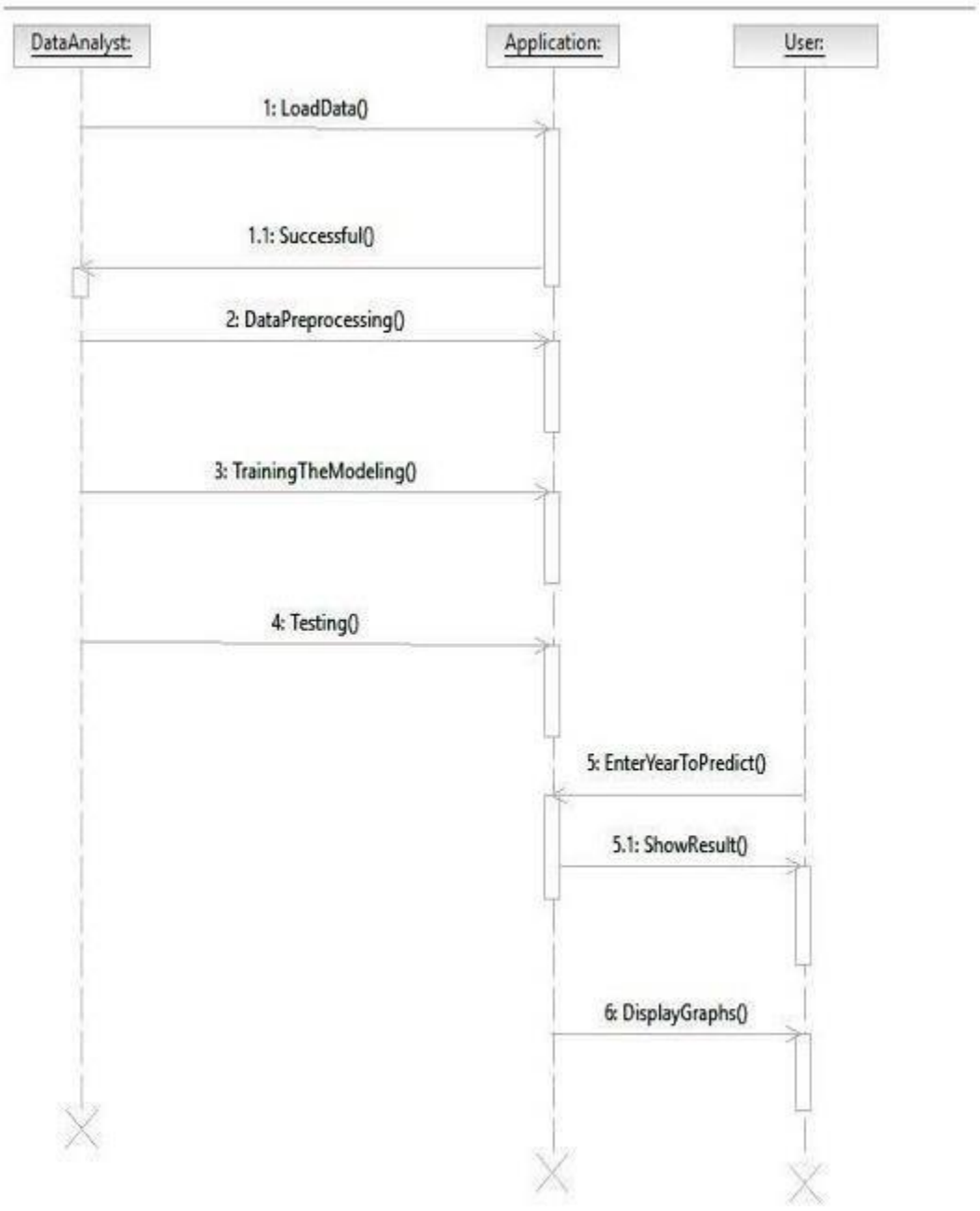


Figure 4.3.2 Sequence Diagram

4.3.3 Collaboration Diagram:

Collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Collaboration diagram and sequence diagram are isomorphic.

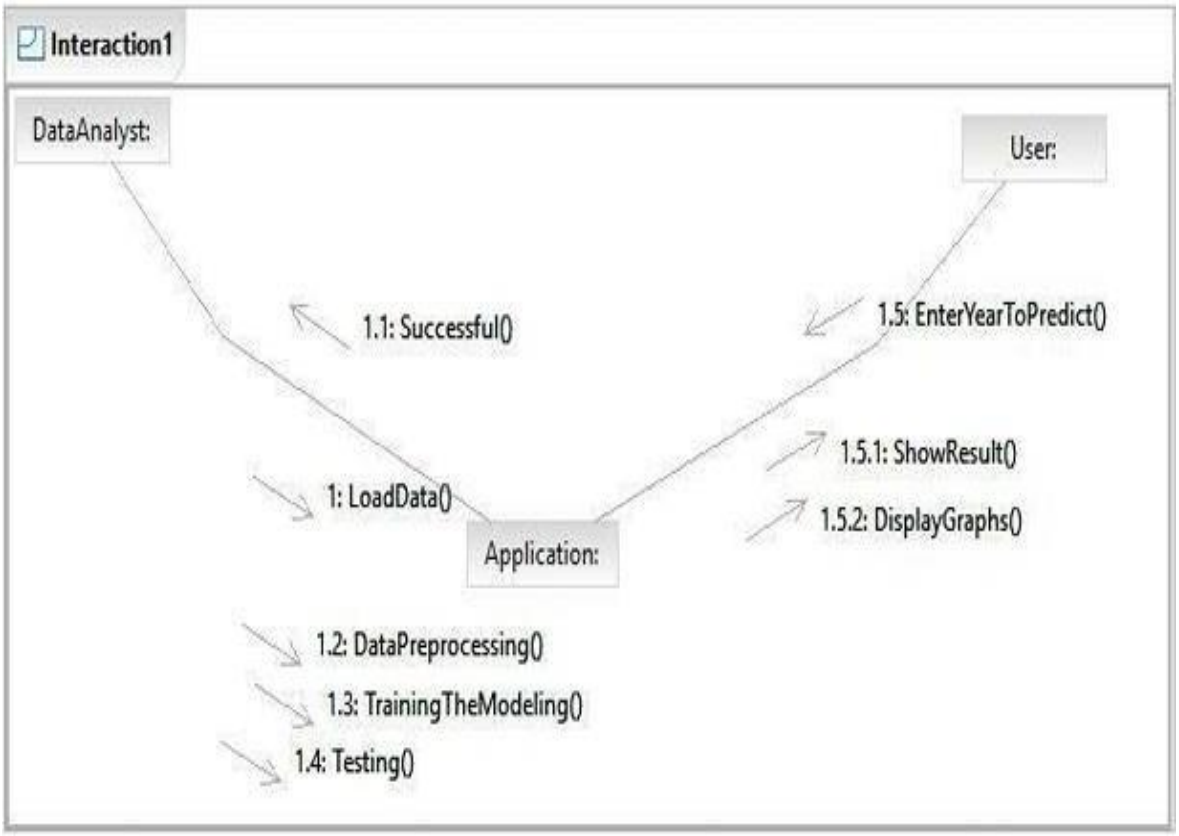


Figure 4.3.3 Collaboration Diagram

4.3.4 Activity Diagram:

An Activity diagram shows the flow from activity to activity within a system it emphasizes the flow of control among objects.

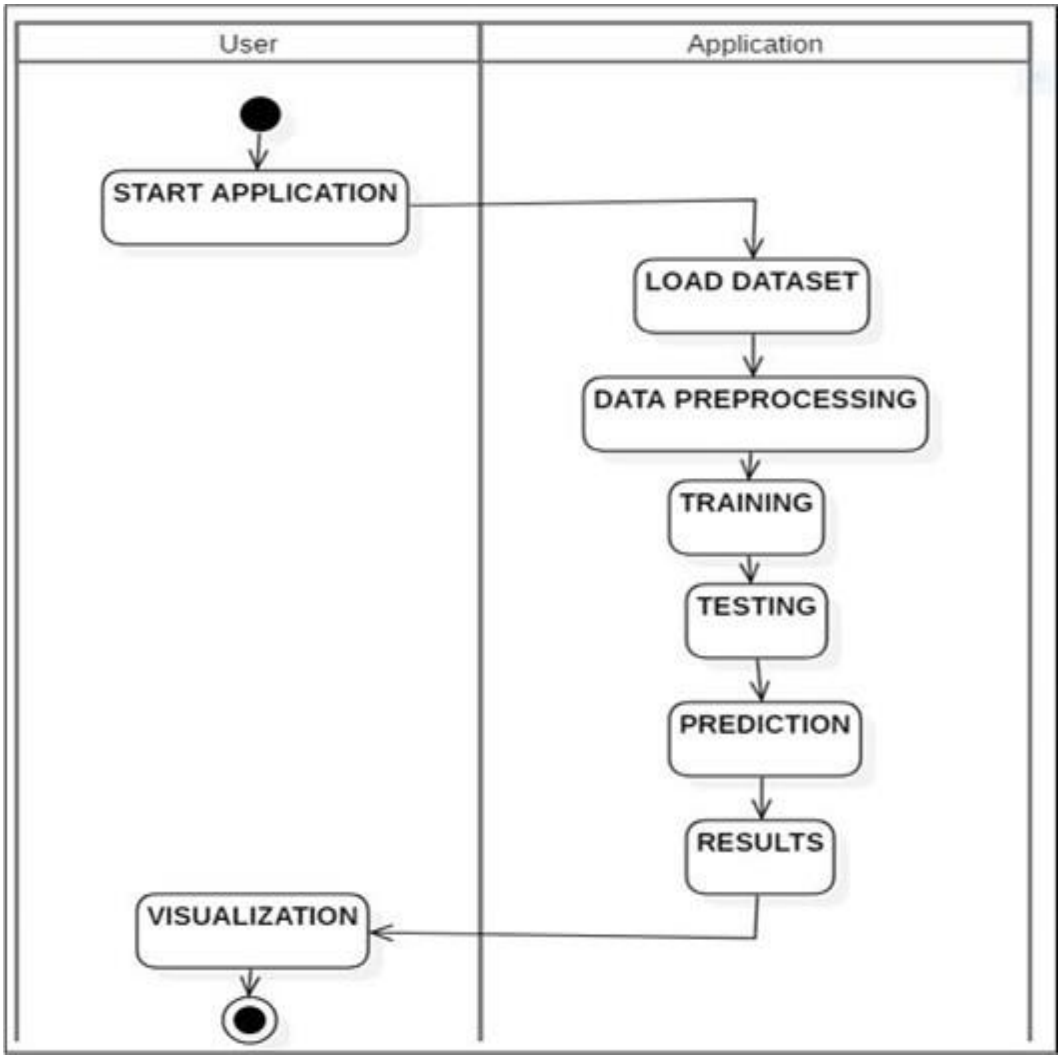


Figure 4.3.4 Activity Diagram

4.3.5 DATA DICTIONARY

	A	B	C	D	E	F	G	H	I	J
1	satisfactio	last_evalu	number_p	average_n	time_spen	Work_acci	left	promotion	sales	salary
2	0.38	0.53	2	157	3	0	1	0	sales	low
3	0.8	0.86	5	262	6	0	1	0	sales	medium
4	0.11	0.88	7	272	4	0	1	0	sales	medium
5	0.72	0.87	5	223	5	0	1	0	sales	low
6	0.37	0.52	2	159	3	0	1	0	sales	low
7	0.41	0.5	2	153	3	0	1	0	sales	low
8	0.1	0.77	6	247	4	0	1	0	sales	low
9	0.92	0.85	5	259	5	0	1	0	sales	low
10	0.89	1	5	224	5	0	1	0	sales	low
11	0.42	0.53	2	142	3	0	1	0	sales	low
12	0.45	0.54	2	135	3	0	1	0	sales	low
13	0.11	0.81	6	305	4	0	1	0	sales	low
14	0.84	0.92	4	234	5	0	1	0	sales	low
15	0.41	0.55	2	148	3	0	1	0	sales	low
16	0.36	0.56	2	137	3	0	1	0	sales	low
17	0.38	0.54	2	143	3	0	1	0	sales	low
18	0.45	0.47	2	160	3	0	1	0	sales	low
19	0.78	0.99	4	255	6	0	1	0	sales	low
20	0.45	0.51	2	160	3	1	1	1	sales	low
21	0.76	0.89	5	262	5	0	1	0	sales	low
22	0.11	0.83	6	282	4	0	1	0	sales	low
23	0.38	0.55	2	147	3	0	1	0	sales	low
24	0.09	0.95	6	304	4	0	1	0	sales	low
25	0.46	0.57	2	139	3	0	1	0	sales	low
26	0.4	0.53	2	158	3	0	1	0	sales	low
27	0.89	0.92	5	242	5	0	1	0	sales	low
28	0.82	0.87	4	239	5	0	1	0	sales	low
29	0.4	0.49	2	135	3	0	1	0	sales	low

Fig 4.3.5 Data Dictionary

CHAPTER 5

IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

The project is implemented by accessing simultaneously from more than one system and more than one window in one system. The application is implemented in the Internet Information Services 5.0 web server under the Windows XP and accessed from various clients.

5.1 TECHNOLOGIES USED

What is Python?

Python is an interpreter, high-level programming language for general-purpose programming by "Guido van Rossum" and first released in 1991, Python has a design philosophy that emphasizes code readability, and a syntax that allows programmers to express concepts in fewer lines of code, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional, procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. Python is managed by the non-profit Python Software Foundation.

Python is a general purpose, dynamic, high level and interpreted programming language. It supports object-oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high level data structures.

- Windows XP
- Python Programming
- Open source libraries: Pandas, NumPy, SciPy, matplotlib, OpenCV

Python Versions

Python 2.0 was released on 16 October 2000 and had many major new features, including a cycle-detecting, garbage collector, and support for Unicode. With this release, the development process became more transparent and community-backed.

Python 3.0 (initially called Python 3000 or py3k) was released on 3 December 2008 after a long testing period. It is a major revision of the language that is not completely backward compatible with previous versions. However, many of its major features have been back ported to the Python 2.6.xand 2.7.x version series, and releases of Python 3 include the 2to3 utility, which automates the translation of Python 2 code to Python 3.

Python 2.7's end-of-life date (a.k.a. EOL, sunset date) was initially set at 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward ported to Python 3.In January 2017, Google announced work on a Python 2.7 to go Trans compiler to improve performance under concurrent workloads.

Python 3.6 had changes regarding UTF-8 (in Windows, PEP 528 and PEP 529) and Python 3.7.0b1 (PEP 540) adds a new "UTF-8 Mode" (and overrides POSIX locale).

Why Python?

- Python is a scripting language like PHP, Perl, and Ruby.
- No licensing, distribution, or development fees
- It is a Desktop application.
- Linux, windows
- Excellent documentation
- Thriving developer community
- For us job opportunity

Libraries Of python:

Python's large standard library, commonly cited as one of its greatest strengths,provides tools suited too many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications (for example, the Web Server Gateway Interface (WSGI) implementation wsgiref follows PEP 33), but most modules are not.

They are specified by their code, internal documentation, and test suites (if supplied). However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of March 2018, the Python Package Index (PyPI), the official repository for thirdparty Python software, contains over 130,000 packages with a wide range of functionality, including:

• Graphical user interfaces

• Web frameworks

• Multimedia • Databases

• Networking

• Test frameworks •

Automation

• Web scraping

• Documentation

• System administration

5.2 MACHINE LEARNING

Machine Learning is an application of artificial intelligence (AI) that provides system the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

Basics of python machine learning:

• You'll know how to use Python and its libraries to explore your data with the help of matplotlib and Principal Component Analysis (PCA).

• And you'll preprocess your data with normalization and you'll split your data into training and test sets.

• Next, you'll work with the well-known K-Means algorithm to construct an unsupervised model, fit this model to your data, predict values, and validate the model that you have built.

• As an extra, you'll also see how you can also use Support Vector Machines (SVM) to construct another model to classify your data.

Why Machine Learning?

• It was born from pattern recognition and theory that computers can learn without being programmed to specific tasks.

• It is a method of Data analysis that automates analytical model building.

Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning "signal" or "feedback" available to a learning system. They are

Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs. As special cases, the input signal can be only partially available, or restricted to special feedback:

Semi-supervised learning: The computer is given only an incomplete training signal: a training set with some (often many) of the target outputs missing**.**

Active learning: The computer can only obtain training labels for a limited set of instances (based on a budget), and also has to optimize its choice of objects to acquire labels for. When used interactively, these can be presented to the user for labelling.

Reinforcement learning: Training data (in form of rewards and punishments) is given only as feedback to the program's actions in a dynamic environment, such asdriving a vehicleor playing a game against an opponent.

Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).

Inregression, also a supervised problem, the outputs are continuous rather than discrete.

Regression: The analysis or measure of the association between one variable (the dependent variable) and one or more other variables (the independent variables), usually formulated in an equation in which the independent variables have parametric coefficients, which may enable future values of the dependent variable to be predicted.

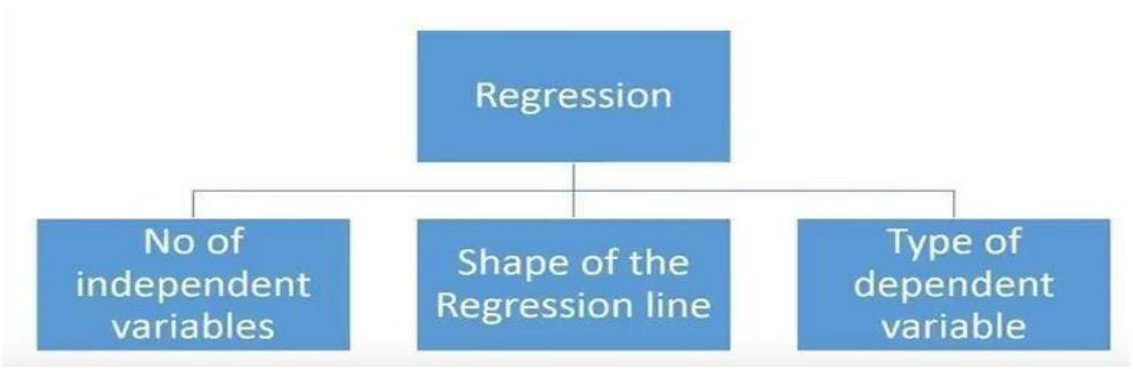


Figure 5.1.1 Regression Structure

What is Regression Analysis?

Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent(target) and independent variable (s) (predictor). This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables. For example, relationship between rash driving and number of road accidents by

Types of Regression:

1. **Linear Regression**
2. **Logistic Regression**
3. **Polynomial Regression**
4. **Stepwise Regression**
5. **Ridge Regression**
6. **Lasso Regression**
7. **Elastic Net Regression**

1.Linear Regression: It is one of the most widely known modelling techniques. Linear regression is usually among the first few topics which people pick while learning predictive modelling. In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear.

Linear Regression establishes a relationship between dependent variable (Y) and one or more independent variables (X) using a best fit straight line (also known as regression line).

2. Logistic Regression: Logistic regression is used to find the probability of event=Success and event=Failure. We should use logistic regression when the dependent variable is binary (0/ 1, True/ False, Yes/ No) in nature. Here the value of Y ranges from 0 to 1 and it can have represented by following equation.

```
odds= p/ (1-p) = probability of event occurrence / probability of not event occurrence
```

```
ln(odds) = ln(p/(1-p)) logit(p) = ln(p/(1-p)) = b0+b1X1+b2X2+b3X3....+bkXk
```

3. Polynomial Regression: A regression equation is a polynomial regression equation if the power of independent variable is more than 1. The equation below represents a polynomial equation:

```
y=a+b*x^2
```

4. Stepwise Regression: This form of regression is used when we deal with multiple independent variables. In this technique, the selection of independent variables is done with the help of an automatic process, which involves no human intervention.

This feat is achieved by observing statistical values like R-square, t-stats and AIC metric to discern significant variables. Stepwise regression basically fits the regression model by adding/dropping co-variants one at a time based on a specified criterion. Some of the most commonly used Stepwise regression methods are listed below:

- Standard stepwise regression does two things. It adds and removes predictors as needed for each step.
- Forward selection starts with most significant predictor in the model and adds variable for each step.
- Backward elimination starts with all predictors in the model and removes the least significant variable for each step. The aim of this modelling technique is to maximize the prediction power with minimum number of predictor variables. It is one of the methods to handle higher dimensionality of data set.

5. Ridge Regression: Ridge Regression is a technique used when the data suffers from multi collinearity (independent variables are highly correlated). In multi collinearity, even though the least squares estimate (OLS) are unbiased; their variances are large which deviates the observed value far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.

Above, we saw the equation for linear regression. Remember? It can be represented as:

y=a+ b*x

This equation also has an error term. The complete equation becomes:

y=a+b*x+e (error term), [error term is the value needed to correct for a prediction error between the observed and predicted value]

=> y=a+y= a+ b1x1+ b2x2+... +e, for multiple independent variables.

In a linear equation, prediction errors can be decomposed into two sub components. First is due to the biased and second is due to the variance. Prediction error can occur due to any one of these two or both components. Here, we'll discuss about the error caused due to variance.

Ridge regression solves the multi collinearity problem throughshrinkage parameterλ (lambda). Look at the equation below.

$$= \underset{\beta \in R^p}{\operatorname{argmin}} \underbrace{\|y - X\beta\|_2^2}_{\text{Loss}} + \lambda \underbrace{\|\beta\|_2^2}_{\text{Penalty}}$$

In this equation, we have two components. First one is least square term and other one is lambda of the summation of β2 (beta- square) where β is the coefficient. This is added to least square term in order to shrink the parameter to have a very low variance.

Important Points:

- The assumptions of this regression is same as least squared regression except normality is not to be assumed.
- It shrinks the value of coefficients but doesn't reaches zero, which suggests no feature selection feature .
- This is a regularization method and uses l2 regularization.

6. Lasso Regression: Similar to Ridge Regression, Lasso (Least Absolute Shrinkage and Selection Operator) also penalizes the absolute size of the regression coefficients. In addition, it is capable of reducing the variability and improving the accuracy of linear regression models. Look at the equation below:

$$= \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\|y - X\beta\|_2^2}_{\text{Loss}} + \lambda \underbrace{\|\beta\|_1}_{\text{Penalty}}$$

Lasso regression differs from ridge regression in a way that it uses absolute values in the penalty function, instead of squares. This leads to penalizing (or equivalently constraining the sum of the absolute values of the estimates) values which causes some of the parameter estimates to turn out exactly zero. Larger the penalty applied, further the estimates get shrunk towards absolute zero. This results to variable selection, out of given n variables.

Important Points:

- The assumptions of this regression is same as least squared regression except normality is not to be assumed .
- It shrinks coefficients to zero (exactly zero), which certainly helps in feature selection.
- This is a regularization method and usesl1 regularization.
- If group of predictors are highly correlated, lasso picks only one of them and shrinks the others to zero

7. Elastic Net Regression: Elastic Net is hybrid of Lasso and Ridge Regression techniques. It is trained with L1 and L2 prior as regularize. Elastic-net is useful when there are multiple features which are correlated. Lasso is likely to pick one of these at random, while elastic-net is likely to pick both.

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta\|_1).$$

A practical advantage of trading-off between Lasso and Ridge is that, it allows Elastic- Net to inherit some of Ridge's stability under rotation.

Important Points:

- It encourages group effect in case of highly correlated variables .
- There are no limitations on the number of selected variables.
- It can suffer with double shrinkage

Beyond these 7 most commonly used regression techniques, you can also look at other models likeBayesian,EcologicalandRobust regression.

Classification

A classification problem is when the output variable is a category, such as "red" or "blue" or "disease" and "no disease". A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes. For example, when filtering emails "spam" or "not spam", when looking at transaction data, "fraudulent", or "authorized". In short Classification either predicts categorical class labels or classifies data (construct a model) based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data.

There are a number of classification models. Classification models include

1. **Logistic regression**
2. **.Decision tree**
3. **Random forest**
4. **Naive Bayes.**

1. Logistic Regression

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes. In simple words, the dependent variable is binary in nature having data coded as either 1 (stands for success/yes) or 0 (stands for failure/no).

Mathematically, a logistic regression model predicts P(Y=1) as a function of X. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc.

2. Decision Tree

Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves.

3. Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:

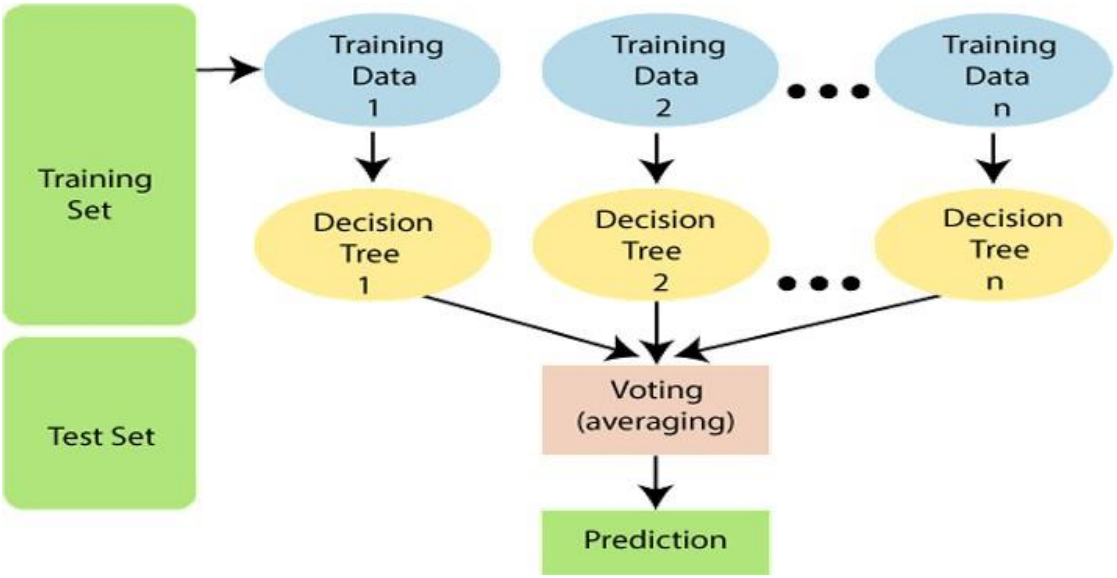
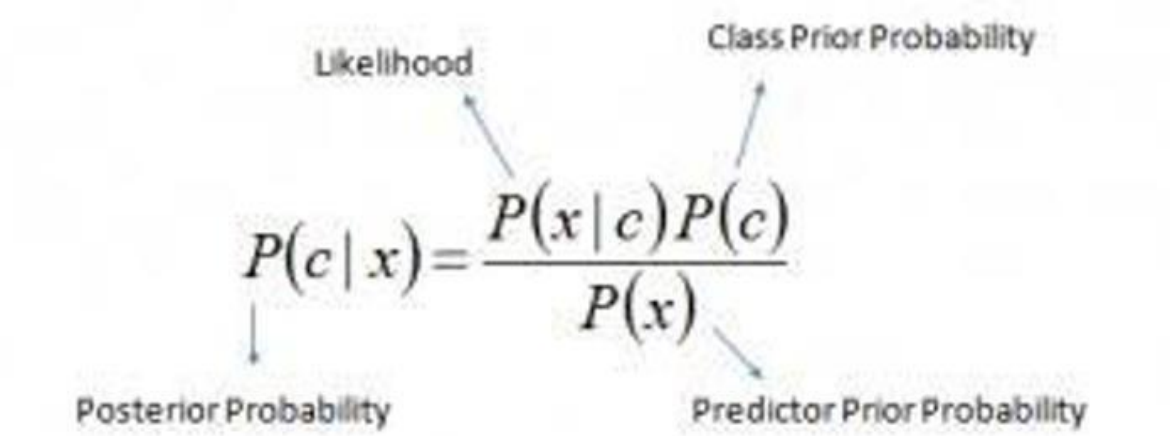


Fig 5.2.1 Image for Random Forest Algorithm

4. Naive Bayes

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

5.3 Sickit-learn:

Scikit-learn (formerly scikits. learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits. learn, a Google Summer of Code project by David Courmayeur. Its name stems from the notion that it is a "SciKit"(SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.

The original codebase was later rewritten by other developers. In 2010 Fabian Pedrosa, Gael Viroqua, Alexandre Gramfort and Vincent Michel, all from INRIA took leadership of the project and made the first public release on February the 1st 2010 .Of the

various scikits, scikit-learn as well as scikit-image were described as “well- maintained and popular” in November 2012. Scikit-learn is largely written in Python, with some core algorithms written in Cython to achieve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

Some popular groups of models provided by scikit-learn include:

- **Ensemble methods:** for combining the predictions of multiple supervised models.
- **Feature extraction:** for defining attributes in image and text data.
- **Feature selection:** for identifying meaningful attributes from which to create supervised models.
- **Parameter Tuning:** for getting the most out of supervised models.
- **Manifold Learning:** For summarizing and depicting complex multi- dimensional data.
- **Supervised Models:** a vast array not limited to generalize linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

Matplotlib:

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of matplotlib.

CHAPTER 6

TESTING

It is the process of testing the functionality and it is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as at undiscovered error. A successful test is one that uncovers an as at undiscovered error. Software testing is usually performed for one of two reasons:

- Defect Detection
- Reliability estimation

Testing in an employee turnover prediction project involves several key stages to ensure the system functions correctly and delivers accurate, reliable results. Here's a brief overview of the testing process:

1. Unit Testing

- **Objective:** Verify that individual components or modules of the system work correctly.

- **Approach:** Test functions and algorithms in isolation. For instance, validate data preprocessing functions, individual machine learning algorithms, and prediction logic.

2. Integration Testing

- **Objective:** Ensure that different system components work together as expected.

- **Approach:** Test the interaction between modules, such as data import, processing, and reporting functionalities. Verify that data flows correctly between the HR management system and the prediction models.

3. System Testing

- **Objective:** Assess the complete and integrated system to ensure it meets the specified requirements.

- **Approach:** Conduct end-to-end testing to evaluate the system's overall performance, including data integration, prediction accuracy, and user interface functionality. Ensure that the system meets all functional and non-functional requirements.

4. Acceptance Testing

- **Objective:** Confirm that the system meets user needs and is ready for deployment.

- **Approach:** Perform user acceptance testing (UAT) with actual HR professionals and stakeholders. Collect feedback on system usability, accuracy of predictions, and overall satisfaction.

5. Performance Testing

- **Objective:** Evaluate the system's performance under various conditions.

- **Approach:** Test system response times, processing speed, and scalability. Assess how the system handles large volumes of data and multiple simultaneous users.

6. Security Testing

- **Objective:** Ensure the system is secure and protects sensitive data.

- **Approach:** Conduct vulnerability assessments, penetration testing, and check for compliance with data protection regulations (e.g., GDPR, CCPA). Verify data encryption and access controls.

7. Regression Testing

- **Objective:** Ensure that new changes or updates do not introduce new issues.

- **Approach:** Re-test the system after updates or bug fixes to confirm that existing functionality remains intact and unaffected.

8. Usability Testing

- **Objective:** Evaluate the system's ease of use and user experience.

- **Approach:** Test the interface and workflow with end-users to identify any usability issues. Gather feedback on user satisfaction and make necessary adjustments to improve the user experience.

9. Data Quality Testing

- **Objective:** Verify the accuracy and completeness of the data used by the system.

- **Approach:** Check data integrity, consistency, and correctness. Validate that data sources are accurate and that preprocessing steps do not introduce errors.

10. Scenario Testing

- **Objective:** Assess how the system performs under different scenarios and conditions.

- **Approach:** Create and test various scenarios, such as high turnover rates, different employee profiles, and unexpected data inputs, to ensure the system handles diverse situations effectively.

By following these testing phases, the employee turnover prediction system can be validated for functionality, performance, security, and user satisfaction, ensuring it meets the project's objectives and provides valuable insights for managing employee retention.

CHAPTER 7

OUTPUT SCREENS

We use Machine Learning models to evaluate a model. The following algoritms are used

1. Random Forest

The Random Forest algorithm is an ensemble learning method that combines multiple decision trees to make predictions. Each tree is built using a random subset of the training data and features, and the final prediction is determined by aggregating the results of all trees—voting for classification tasks or averaging for regression tasks. This approach enhances accuracy and reduces overfitting by leveraging the diversity among the trees. In an employee turnover prediction project, Random Forest is helpful because it can handle large and complex datasets with numerous features, such as employee demographics, performance metrics, and job satisfaction scores. Its ability to assess the importance of different features aids in identifying key factors influencing turnover. Additionally, its robustness to noise and missing values makes it a reliable choice for predicting turnover risks and providing actionable insights for improving employee retention strategies.

2. Decision Tree

The Decision Tree algorithm is a machine learning technique that models decisions and their possible consequences using a tree-like structure. It splits data into subsets based on feature values, creating a tree where each node represents a decision rule, and branches represent possible outcomes. The process continues until the data is divided into homogeneous groups or meets a stopping criterion. For employee turnover prediction, Decision Trees are useful because they can easily handle both numerical and categorical data, providing clear, interpretable results about which factors influence turnover. They help in understanding complex relationships within the data and making straightforward predictions about employee attrition based on various attributes, such as job satisfaction, tenure, and performance. Their ability to visualize decision paths makes them particularly valuable for generating actionable insights and guiding HR strategies for improving employee retention.

1.Required libraries

```
[ ]  import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.preprocessing import StandardScaler
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score, classification_report
```

2.Load the dataset

```
[ ]  # Load the dataset
     data = pd.read_csv('/content/HR.csv')
```

3.Data preprocessing & Feature selection

>>Overview of data

```
data
```

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	sales	salary
0	0.38	0.53	2	157	3	0	1	0	sales	low
1	0.80	0.86	5	262	6	0	1	0	sales	medium
2	0.11	0.88	7	272	4	0	1	0	sales	medium
3	0.72	0.87	5	223	5	0	1	0	sales	low
4	0.37	0.52	2	159	3	0	1	0	sales	low
...	...	...	...	...	...	...	...	...	...	...
14994	0.40	0.57	2	151	3	0	1	0	support	low
14995	0.37	0.48	2	160	3	0	1	0	support	low
14996	0.37	0.53	2	143	3	0	1	0	support	low
14997	0.11	0.96	6	280	4	0	1	0	support	low
14998	0.37	0.52	2	158	3	0	1	0	support	low

14999 rows × 10 columns

```
data.info()
```

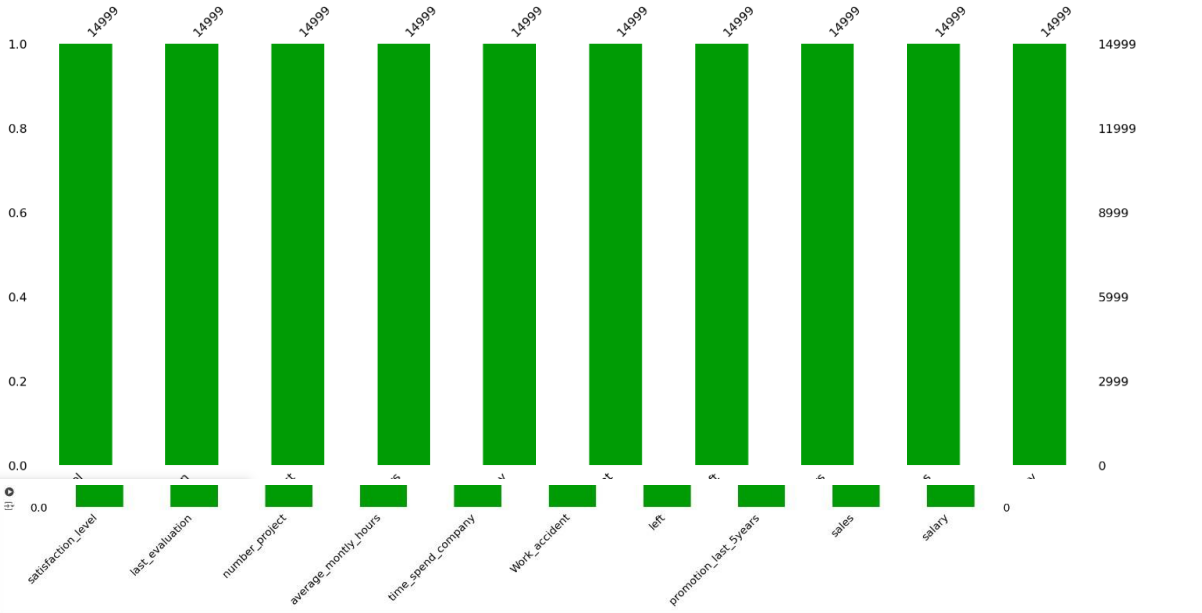
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   satisfaction_level     14999 non-null  float64
 1   last_evaluation        14999 non-null  float64
 2   number_project         14999 non-null  int64  
 3   average_montly_hours   14999 non-null  int64  
 4   time_spend_company     14999 non-null  int64  
 5   Work_accident          14999 non-null  int64  
 6   left                   14999 non-null  int64  
 7   promotion_last_5years  14999 non-null  int64  
 8   sales                  14999 non-null  object 
 9   salary                 14999 non-null  object 
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

```
[ ] !pip install missingno
```

```
Requirement already satisfied: missingno in /usr/local/lib/python3.10/dist-packages (0.5.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.25.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from missingno) (3.7.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.13.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from missingno) (0.13.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn->missingno) (2.0.3)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn->missingno) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn->missingno) (2024.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->missingno) (1.16.0)
```

```
import missingno as msno
import matplotlib.pyplot as plt # import the matplotlib library

# Data Cleaning
# Check for missing values
data.isnull().sum()

# Fill missing values with the mean for numerical columns
numerical_cols = data.select_dtypes(include=['number']).columns
data[numerical_cols] = data[numerical_cols].fillna(data[numerical_cols].mean())

# Generate the missing values matrix using missingno.bar()
msno.bar(data, color='#009c05')
```

```
# Display the plot
plt.show()

# Fill missing values with the mode for categorical columns
categorical_cols = data.select_dtypes(exclude=['number']).columns
data[categorical_cols] = data[categorical_cols].fillna(data[categorical_cols].mode().iloc[0])

# Remove duplicates
data.drop_duplicates(inplace=True)

# Handling Noisy Data
# Example: Binning for 'Age'
if 'Age' in data.columns:
  data['Age'] = pd.cut(data['Age'], bins=5, labels=False)

# Removal of Outliers using IQR method
numerical_data = data.select_dtypes(include=['number'])
Q1 = numerical_data.quantile(0.25)
Q3 = numerical_data.quantile(0.75)
IQR = Q3 - Q1
data_cleaned = data[~((numerical_data < (Q1 - 1.5 * IQR)) | (numerical_data > (Q3 + 1.5 * IQR))).any(axis=1)]
```



>>Transforming Categorical variables into numeric variables(or vice versa)

This code first uses one-hot encoding to transform the categorical features into numerical ones. Then, it uses binning to convert the numerical features into categorical ones.

```
# Select categorical columns
categorical_cols = data_cleaned.select_dtypes(include=['object']).columns

# Apply one-hot encoding
data_encoded = pd.get_dummies(data_cleaned, columns=categorical_cols)


# Select numerical columns (excluding those already one-hot encoded)
numerical_cols = data_encoded.select_dtypes(include=['number']).columns
numerical_cols = [col for col in numerical_cols if col not in categorical_cols]

# Apply binning to convert numerical to categorical
for col in numerical_cols:
  data_encoded[col] = pd.cut(data_encoded[col], bins=5, labels=False)
```

4.Featuew selection & engineering

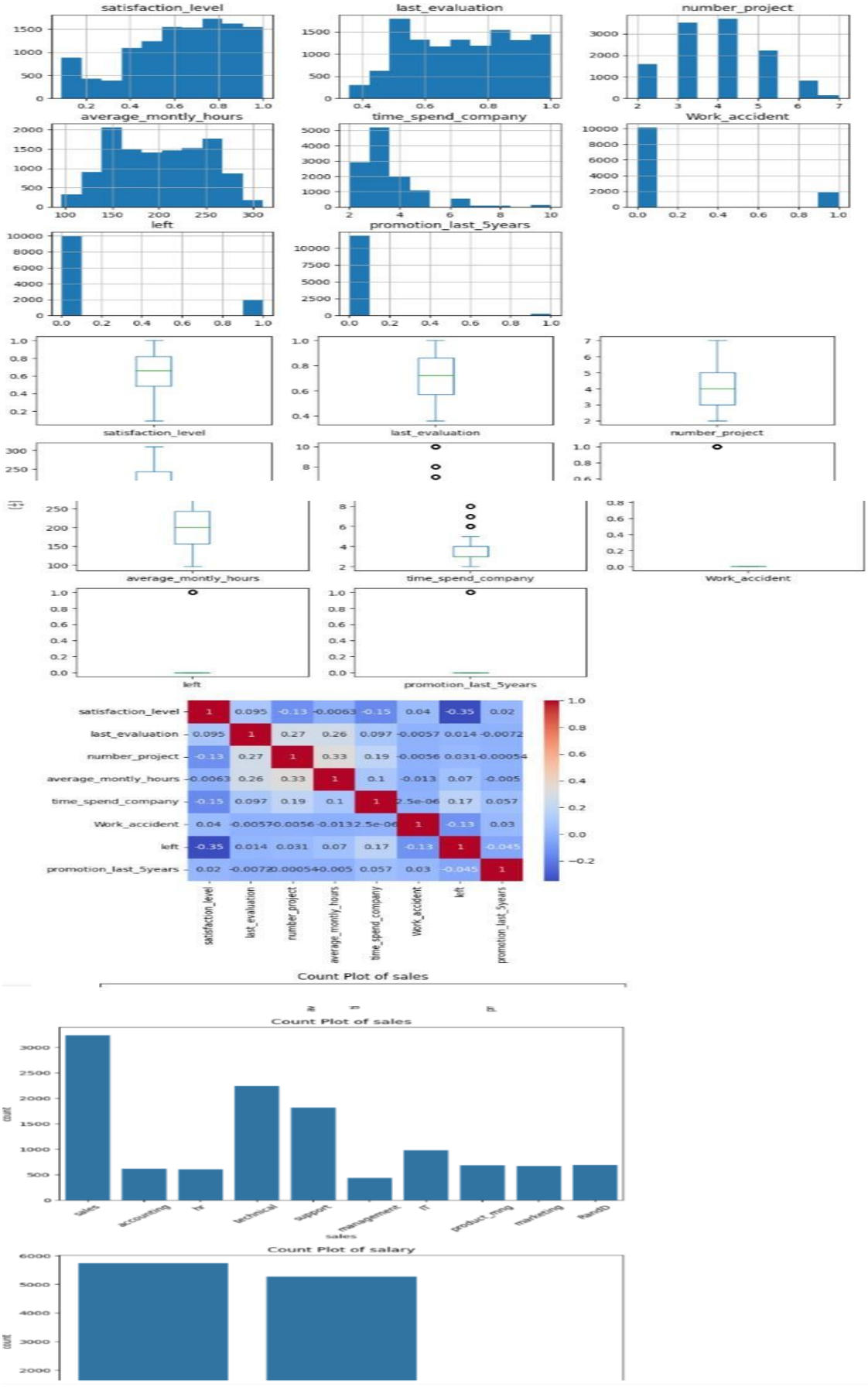
This code performs feature selection using SelectKBest with f_classif to select the top 5 features based on ANOVA F-value between label/feature for classification tasks.

Additionally, it demonstrates feature engineering by creating an interaction term by multiplying satisfaction_level and last_evaluation.

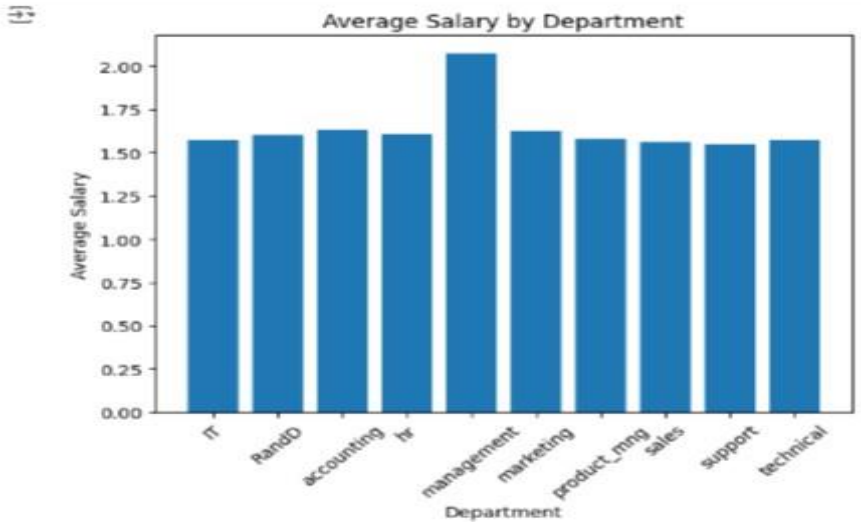
Remember to replace 'left' with the actual name of your target variable if it's different.

```
from sklearn.feature_selection import SelectKBest, f_classif

# Separate features (X) and target variable (y)
X = data_encoded.drop('left', axis=1)  # Assuming 'left' is the target variable
y = data_encoded['left']

# Apply SelectKBest for feature selection
selector = SelectKBest(f_classif, k=5)  # Select top 5 features
X_new = selector.fit_transform(X, y)

# Get the selected feature names
selected_features = X.columns[selector.get_support()]

# Check if required columns are present in selected features
required_features = ['satisfaction_level', 'last_evaluation']
for feature in required_features:
  if feature not in selected_features:
    print(f"Warning: {feature} not selected by SelectKBest. Interaction term will not be created.")
```

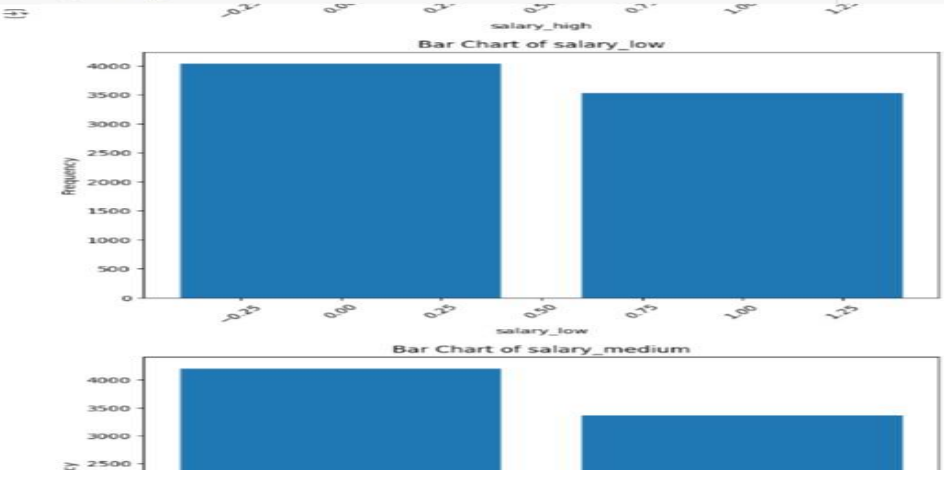
```
import matplotlib.pyplot as plt
import seaborn as sns

# Assuming 'data' is your DataFrame

# Histograms for numerical features
data.hist(figsize=(12, 8))
plt.show()

# Box plots for numerical features
data.plot(kind='box', subplots=True, layout=(3, 3), sharex=False, sharey=False, figsize=(12, 8))
plt.show()

# Correlation matrix
# Include only numerical features for correlation calculation
corr = data.select_dtypes(include=['number']).corr()
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.show()

# Count plots for categorical features
for col in data.select_dtypes(include=['object']):
  plt.figure(figsize=(8, 6))
  sns.countplot(x=col, data=data)
  plt.title(f'Count Plot of {col}')
  plt.xticks(rotation=45)
  plt.show()

# Scatter plots for relationships between numerical features
sns.pairplot(data)
plt.show()
```


satisfaction_level
last_evaluation
number_project
average_montly_hours
time_spend_company
Work_accident
left
promotion_last_5years
satisfaction_level
last_evaluation
number_project
average_montly_hours
time_spend_company
Work_accident
left
promotion_last_5years
satisfaction_level 1 0.095 -0.13 -0.0063 -0.15 0.04 -0.35 0.02
last_evaluation 0.095 1 0.27 0.26 0.097 -0.0057 0.014 -0.0072
number_project -0.13 0.27 1 0.33 0.19 -0.0056 0.031 -0.00054
average_montly_hours -0.0063 0.26 0.33 1 0.1 -0.013 0.07 -0.005
time_spend_company -0.15 0.097 0.19 0.1 1 2.5e-06 0.17 0.057
Work_accident 0.04 -0.0057 0.0056 -0.013 2.5e-06 1 -0.13 0.03
left -0.35 0.014 0.031 0.07 0.17 -0.13 1 -0.045
promotion_last_5years 0.02 -0.0072 0.00054 -0.005 0.057 0.03 -0.045 1
Count Plot of sales
Count Plot of sales
count
sales
accounting
hr
technical
support
management
IT
product_mng
marketing
randD
Sales
Count Plot of salary
count

```python
#Bar chart for data file
import pandas as pd
import matplotlib.pyplot as plt

# Load the HR data (replace 'hr_data.csv' with your file)
hr_data = pd.read_csv('/content/HR.csv')

# Convert 'salary' column to numeric values
salary_mapping = {'low': 1, 'medium': 2, 'high': 3} # create a dictionary for mapping
hr_data['salary'] = hr_data['salary'].map(salary_mapping) # map the values

# Example: Bar chart of average salary by department
average_salary = hr_data.groupby('sales')['salary'].mean()

# Create bar chart
plt.bar(average_salary.index, average_salary.values)

# Add labels and title
plt.xlabel("Department")
plt.ylabel("Average Salary")
plt.title("Average Salary by Department")

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Display the chart
plt.show()
```



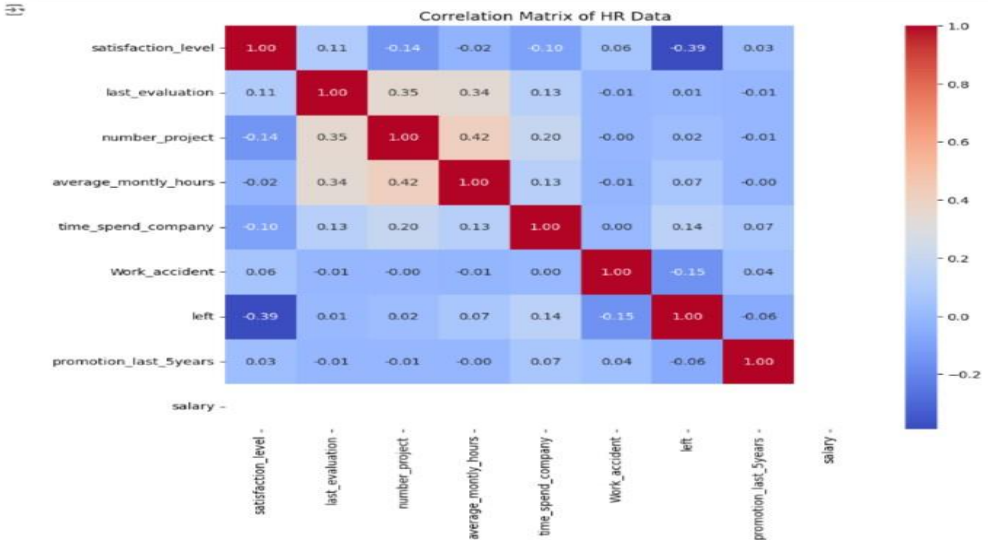

```python
#barchart for encoded file
import pandas as pd
import matplotlib.pyplot as plt

# Assuming 'data_encoded' is your DataFrame with encoded categorical features

for col in data_encoded.columns:
  plt.figure(figsize=(8, 6))
  value_counts = data_encoded[col].value_counts()
  plt.bar(value_counts.index, value_counts.values)
  plt.title(f'Bar Chart of {col}')
  plt.xlabel(col)
  plt.ylabel('Frequency')
  plt.xticks(rotation=45)
  plt.show()
```



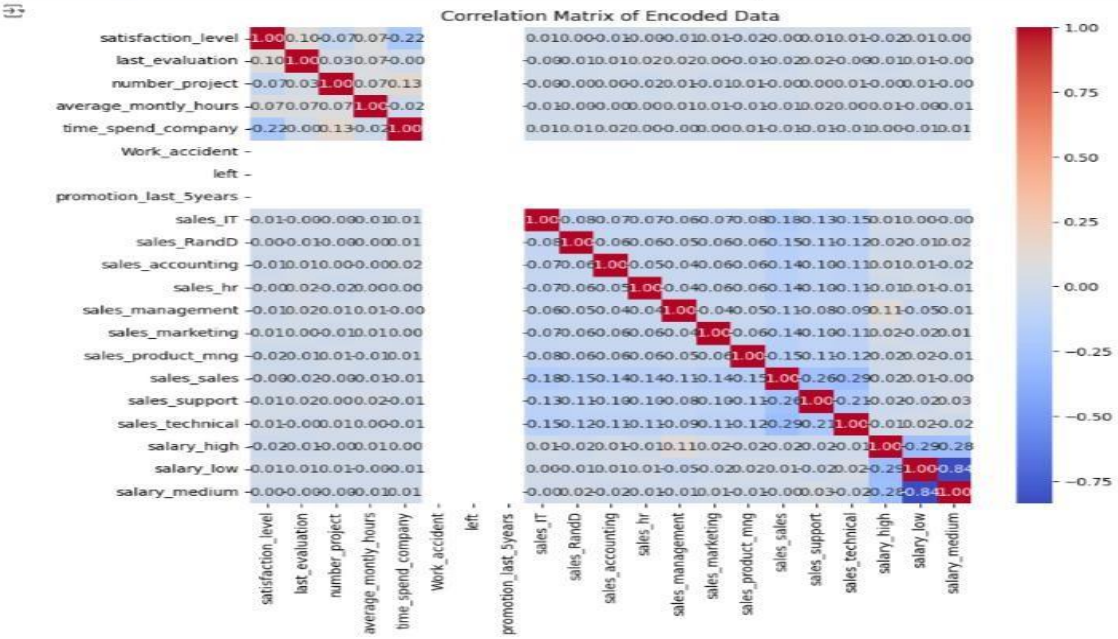
>>Heatmap

A heatmap is like a colorful grid that shows how different things are related to each other. Imagine a table where the colors in the cells represent the values - the more intense the color, the stronger the relationship.

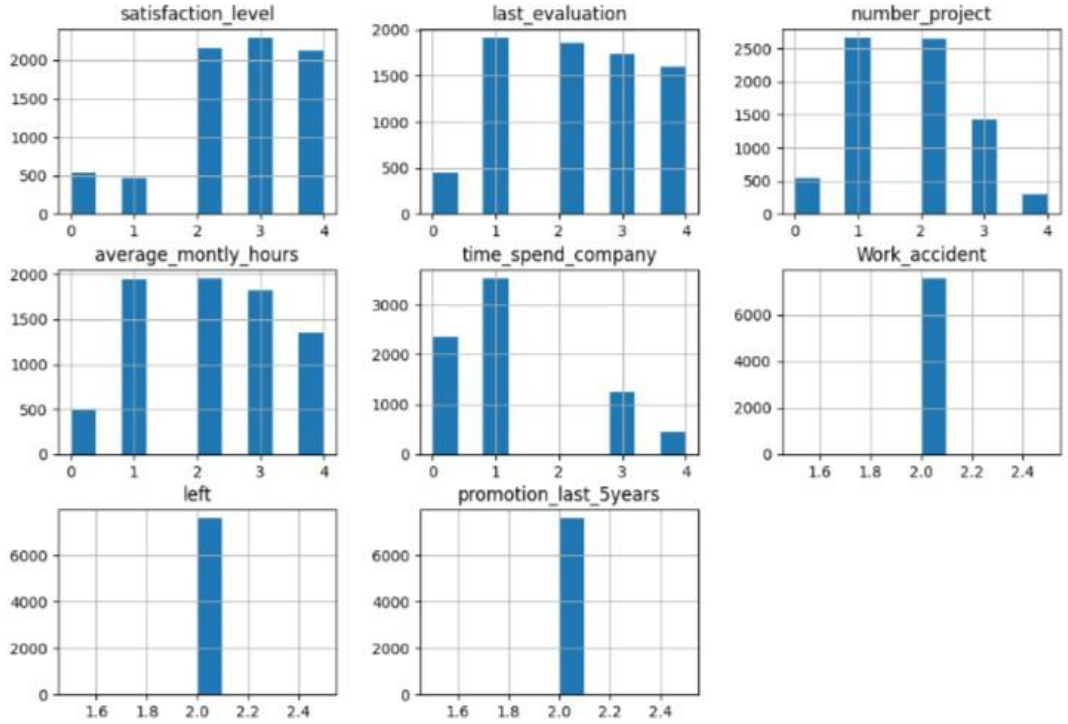
```
#Heat map
# Convert 'salary' column to numeric values for correlation calculation
salary_mapping = {'low': 1, 'medium': 2, 'high': 3}
hr_data['salary'] = hr_data['salary'].map(salary_mapping)

# Calculate the correlation matrix
# Include only numerical features for correlation calculation
corr = hr_data.select_dtypes(include=['number']).corr()

# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix of HR Data')
plt.show()
```



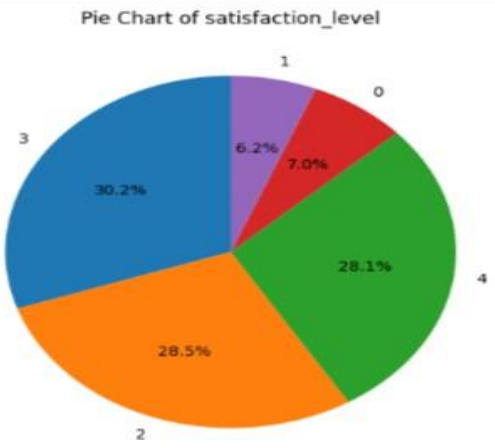

```
#heatmap for encoded data
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'data_encoded' is your DataFrame with encoded categorical features

# Calculate the correlation matrix
corr = data_encoded.corr()

# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix of Encoded Data')
plt.show()
```



```python
#histogram for encoded data
import pandas as pd
import matplotlib.pyplot as plt

# Assuming 'data_encoded' is your DataFrame with encoded categorical features

# Histograms for all features
data_encoded.hist(figsize=(12, 8))
plt.show()
```



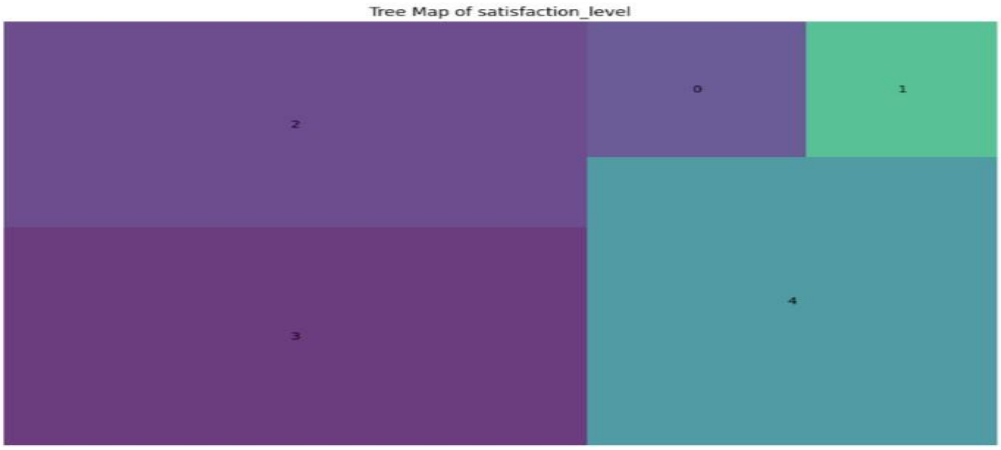

```python
import pandas as pd
import matplotlib.pyplot as plt

# Assuming 'data_encoded' is your DataFrame with encoded categorical features

# Choose a column for the pie chart
column_name = 'satisfaction_level' # Changed to the correct column name

# Calculate value counts
value_counts = data_encoded[column_name].value_counts() #Ensure that data_encoded contains the specified column

# Create pie chart
plt.figure(figsize=(8, 6))
plt.pie(value_counts.values, labels=value_counts.index, autopct='%1.1f%%', startangle=90)
plt.title(f'Pie Chart of {column_name}')
plt.show()
```



```
[ ] !pip install squarify
```

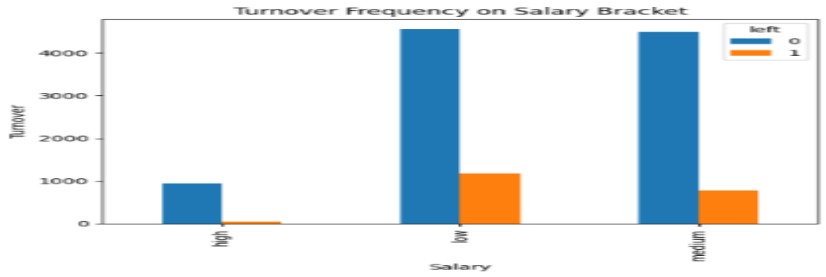
```
Collecting squarify
  Downloading squarify-0.4.4-py3-none-any.whl.metadata (600 bytes)
Downloading squarify-0.4.4-py3-none-any.whl (4.1 kB)
Installing collected packages: squarify
Successfully installed squarify-0.4.4
```

```
[ ] import pandas as pd
    import matplotlib.pyplot as plt
    import squarify

    # Assuming 'data_encoded' is your DataFrame with encoded categorical features

    # Choose a column for the tree map
    column_name = 'satisfaction_level'  # Replace with the actual column name

    # Calculate value counts
    value_counts = data_encoded[column_name].value_counts()

    # Create tree map
    plt.figure(figsize=(12, 8))
    squarify.plot(sizes=value_counts.values, label=value_counts.index, alpha=.8)
    plt.title(f'Tree Map of {column_name}')
    plt.axis('off')
    plt.show()
```



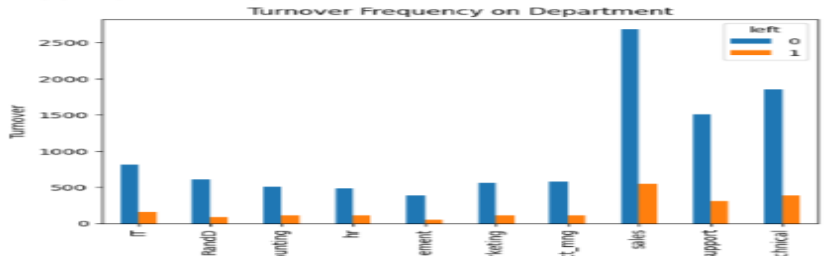

```
    pd.crosstab(data.salary,data.left).plot(kind='bar')
    plt.title('Turnover Frequency on Salary Bracket')
    plt.xlabel('Salary')
    plt.ylabel('Turnover')
```

```
Text(0, 0.5, 'Turnover')
```




```
[ ] pd.crosstab(data.sales,data.left).plot(kind='bar')
    plt.title('Turnover Frequency on Department')
    plt.xlabel('Department')
    plt.ylabel('Turnover')
```

```
Text(0, 0.5, 'Turnover')
```



```
crit      gini
split     best
depth     2
min_split 2
min_leaf  1
Decision Tree Training Accuracy : 100.00%
Decision Tree Test Accuracy : 100.00%

Confusion Matrix for Test Data :
[[1514]]

Confusion Matrix for Training Data :
[[6052]]
```



```
crit      gini
          bootstrap
depth     5
forest    50
min_split 2
min_leaf  2
Random Forest Training Accuracy : 100.00%
Random Forest Test Accuracy : 100.00%

Confusion Matrix for Test Data :
[[1514]]

Confusion Matrix for Training Data :
[[6052]]
```


```
[ ] from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
    print(classification_report(y_test,y_preds))
```

```
              precision    recall  f1-score   support

           2       1.00      1.00      1.00      1514

    accuracy                           1.00      1514
   macro avg       1.00      1.00      1.00      1514
weighted avg       1.00      1.00      1.00      1514
```

```
[ ] #If accuracy of model is less than 0.75 improve model
    #clumsy example,there are better inbuilt methods
    #try different amount of n_estimators
```

```
[ ] import numpy as np
    np.random.seed(42)
    for i in range(10,100,10):
      print(f"Trying model with {i} estimators...")
      clf=RandomForestClassifier(n_estimators=i).fit(X_train,y_train)
      print(f"Model accuracy on test set:{clf.score(X_test,y_test)*100:.2f}%")
      print("")
```

```
Trying model with 10 estimators...
Model accuracy on test set:100.00%

Trying model with 20 estimators...
Model accuracy on test set:100.00%

Trying model with 30 estimators...
Model accuracy on test set:100.00%

Trying model with 40 estimators...
Model accuracy on test set:100.00%

Trying model with 50 estimators...
Model accuracy on test set:100.00%

Trying model with 60 estimators...
Model accuracy on test set:100.00%

Trying model with 70 estimators...
Model accuracy on test set:100.00%

Trying model with 80 estimators...
Model accuracy on test set:100.00%

Trying model with 90 estimators...
Model accuracy on test set:100.00%
```

```python
from sklearn.model_selection import cross_val_score
np.random.seed(42)
for i in range(100,200,10):
  print(f"Trying model with {i} estimators...")
  model=RandomForestClassifier(n_estimators=i).fit(X_train,y_train)
  print(f"Model accuracy on test set:{model.score(X_test,y_test)*100:.2f}%")
  #measure the mean cross validation score across different train and test spilits
  cross_val_mean=np.mean(cross_val_score(model,X,y,cv=5))
  print(f"5-fold Cross-validation score: {cross_val_mean*100:.2f}%")
  print("")
```

```
Trying model with 100 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 110 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 120 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 130 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 140 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 150 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 160 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 170 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 180 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%

Trying model with 190 estimators...
Model accuracy on test set:100.00%
5-fold Cross-validation score: 100.00%
```

10.Prediction based on feedback data

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression

# Sample data - replace with your actual data
data = {
    'features': ['High workload', 'Low salary', 'Lack of growth', 'Toxic environment', 'Good work-life balance'],
    'prediction': [1, 1, 1, 0, 0],  # 1 for likely to leave, 0 for likely to stay
    'feedback': ['Agree', 'Agree', 'Disagree', 'Agree', 'Disagree']
}

# Create a DataFrame
feedback_df = pd.DataFrame(data)

# Combine features and feedback for richer input data
feedback_df['combined'] = feedback_df['features'] + ' - ' + feedback_df['feedback']

# Prepare the data for training
X = feedback_df['combined']
y = feedback_df['prediction']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Convert text data to numerical representation using TF-IDF
vectorizer = TfidfVectorizer()
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)

# Train a Logistic Regression model
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
# Function to predict employee attrition and provide suggestions
def predict_attrition_and_suggest(feedback, model, vectorizer):
  """
  Predicts employee attrition based on feedback and provides suggestions.

  Args:
    feedback: The employee feedback.
    model: The trained machine learning model.
    vectorizer: The TF-IDF vectorizer used for text processing.

  Returns:
    A tuple containing the prediction (0 or 1) and a suggestion string.
  """
  # Transform the feedback using the trained vectorizer
  feedback_vec = vectorizer.transform([feedback])
  # Predict the likelihood of leaving
  prediction = model.predict(feedback_vec)[0]

  if prediction == 1:
    suggestion = "HR should address this concern immediately. Consider offering growth opportunities, improving work conditions, or adjusting compensation."
  else:
    suggestion = "This is a positive sign! Continue fostering a positive environment and address any potential issues proactively."

  return prediction, suggestion

# Example usage
new_feedback = "High workload - Agree"
prediction, suggestion = predict_attrition_and_suggest(new_feedback, model, vectorizer)

print(f"Prediction: {prediction}")  # 0 for likely to stay, 1 for likely to leave
print(f"Suggestion: {suggestion}")
```

```
Prediction: 1
Suggestion: HR should address this concern immediately. Consider offering growth opportunities, improving work conditions, or adjusting compensation.
```

CHAPTER 8

CONCLUSION

In conclusion, the employee turnover prediction project has successfully demonstrated the value of leveraging advanced data analytics to address a critical organizational challenge. By implementing sophisticated algorithms such as Random Forest and Decision Trees, the project has provided actionable insights into the factors influencing employee attrition. The system developed offers a robust, scalable solution capable of handling complex datasets and delivering accurate, reliable predictions.

The integration of real-time data and the incorporation of user feedback mechanisms have enhanced the system's adaptability and usability. This approach ensures that predictions remain relevant and actionable, enabling HR professionals to proactively manage and mitigate turnover risks.

Overall, the project underscores the importance of data-driven decision-making in optimizing workforce management. By identifying high-risk employees and understanding the underlying causes of turnover, the organization can implement targeted retention strategies, improve employee satisfaction, and ultimately achieve greater stability and productivity. The successful deployment and integration of this system mark a significant step toward a more informed and strategic approach to human resource management.