



REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
CHEMICAL AND METALLURGICAL FACULTY
DEPARTMENT OF MATHEMATICAL ENGINEERING

DESIGN APPLICATIONS

SUPERVISED LEARNING METHOD IN MACHINE LEARNING USING HR ANALYTICS DATASET

Advisor
Prof.Dr. Ayla SAYLI

15058009
Sümeyye SEREN

Istanbul, 2019



REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
CHEMICAL AND METALLURGICAL FACULTY
DEPARTMENT OF MATHEMATICAL ENGINEERING

DESIGN APPLICATIONS

SUPERVISED LEARNING METHOD IN MACHINE LEARNING USING HR ANALYTICS DATASET

Advisor
Prof.Dr. Ayla SAYLI

15058009
Sümeyye SEREN

Istanbul, 2019

TABLE OF CONTENTS

LIST OF ABBREVIATIONS	iv
LIST OF FIGURES	v
ACKNOWLEDGEMENTS	vii
ABSTRACT	viii
ÖZET	ix
1 Introduction	1
1.1 Purpose of This Work	1
1.2 Machine Learning	1
1.2.1 History Of Machine Learning	2
1.2.2 Machine Learning Systems Working Principles	5
1.2.3 Machine Learning Categories	5
1.3 Python	10
1.3.1 Intoduction the Python	10
1.3.2 Using Python for Machine Learning Projects	10
2 Supervised Learning Method in ML Using HR Analytic Dataset	22
2.1 Data Scrubbing and Data Preparation	22
2.1.1 Noisy Data	23
2.1.2 Missing Data	23
2.1.3 Inconsistent Data	24
2.2 Analysing Dataset	24
2.3 Machine Learning Algorithms Implementation	36
2.3.1 Logistic Regression	36
2.3.2 Decision Tree(Gini)	38
2.3.3 Decision Tree(Entropy)	41
2.3.4 Random Forest	42
3 Conclusion	45

References	46
Curriculum Vitae	47

LIST OF ABBREVIATIONS

ML	Machine Learning
AI	Artificial Intelligence
CSV	Comma-Separated Values
INT	Integer Value

LIST OF FIGURES

Figure 1.1	The lineage of machine learning represented by a row of Russian matryoshka dolls	2
Figure 1.2	Brief History of Machine Learning	3
Figure 1.3	Historical mentions of “machine learning” in published books . Source: Google Ngram Viewer, 2017	4
Figure 1.4	Comparison of Input Command vs Input Data	5
Figure 1.5	Machine Learning Categories and Algorithms	6
Figure 1.6	Types of Classification	7
Figure 1.7	Supervised learning: regression models	8
Figure 1.8	Common algorithms used to perform supervised and unsupervised machine learning	9
Figure 1.9	Comparison of Most Popular Languages are used for Machine Learning	11
Figure 1.10	Linear Regression	16
Figure 1.11	Decision Tree	17
Figure 1.12	Naive Bayes	18
Figure 1.13	kNN	19
Figure 1.14	K-Means	20
Figure 1.15	K-Means	21
Figure 2.1	Data Scrubbing	22
Figure 2.2	Output of Listing 2.2	26
Figure 2.3	Output of Listing 2.3 for each Categorical Value	28
Figure 2.4	Output of Listing 2.4 for each Numerical Value	31
Figure 2.5	Output of Listing 2.5	32
Figure 2.6	Output of Listing 2.6	33
Figure 2.7	Outputs of Listing 2.7	34
Figure 2.8	Outputs of Listing 2.8	35
Figure 2.9	Outputs of Listing 2.10	36
Figure 2.10	Outputs of Listing 2.11	37
Figure 2.11	Outputs of Listing 2.12	38
Figure 2.12	Outputs of Listing 2.13	38

Figure 2.13 Outputs of Listing 2.14	39
Figure 2.14 Outputs of Listing 2.15	40
Figure 2.15 Outputs of Listing 2.16	41
Figure 2.16 Outputs of Listing 2.17	42
Figure 2.17 Outputs of Listing 2.18	44
Figure 3.1 The Excel Table to Compare 4 Models	45

ACKNOWLEDGEMENTS

I would like to express my special thanks to my teachers(Ayla Şaylı and Kemal Koşuta) who gave me the golden opportunity to do this wonderful project on the Machine Learning, which also helped me in doing a lot of Research and I came to know about so many new things I am really thankful to them. Secondly, I would also like to thank myself and my friends who helped me a lot in finalizing this project within the limited time frame.

Sümeyye SEREN

ABSTRACT

Supervised Learning Method in Machine Learning using HR Analytics Dataset

Sümeyye SEREN

Department of Mathematical Engineering

Mathematical Project

Advisor: Prof.Dr. Ayla SAYLI

In this study,it was aimed to create the analytical models that predict whether a employee will leave the job or not.The dataset which was used in this project was created for testing purposes and it is not organic.Because of this reason,there is no null values so only outlier analysis was implemented on this dataset.

In this method,Classification method which is a subbranch of supervised learning methods were used.Used algorithms are Logistic Regression,Decision Tree(Gini),Decision Tree(Entropy) and Random Forest.For each model,the dataset was divided into train and test sets , 75% and 25% ,respectively.

Those models,which were trained with the same training set using the corresponding functions in Jupyter Notebook with Python programming language,were tested with the same data set and the accuracy rates of them were compared.As a result of the comparison, with given data,it is observed that the model of the Logistic Regression has the highest accuracy rate of 88% .

Keywords: Machine Learning , Supervised Learning ,Classification ,Logistic Regression ,Decision Tree

ÖZET

Bu çalışmada makine öğrenmesi algoritmaları kullanılarak çalışanların işten ayrılma ihtimallerinin tahminleri yapılmıştır.Çalışmada kullanılan veri kümesi test kümesi olarak oluşturulmuş organik olmayan bir verisetidir.Bu sebeple veri üzerinde herhangi bir anonimleştirme işlemi uygulanmamıştır.

Çalışmada, gözetimli makine öğrenmesine dayalı sınıflandırma tekniklerinden yaygın olarak kullanılan dört tanesi ele alınmıştır.Bunlar,Lojistik Regresyon,Karar Ağaçları(Gini),Karar Ağaçları(Entropy),Rastgele Orman Metodlarıdır.Her bir model için veri seti %25 %75 şekilde eğitim ve modelin testi için bölünmüştür.

Jupyter Notebook ile Python programla dilindeki ilgili fonksiyonlar kullanılarak aynı eğitim seti ile eğitilen bu modeller yine aynı test seti ile test edilip tahmin oranları kıyaslanmıştır.Kıyaslama sonucunda bahsedilen veri kümesi üzerinden hesaplandığında,doğruluk oranı(%88) ile Lojistik Regresyona ait modelin tahmin ettiği gözlemlenmiştir.

Keywords: Makine Öğrenmesi , Gözetimli Öğrenme ,Sınıflandırma , Lojistik Regresyon , Karar Ağaçları

1

Introduction

1.1 Purpose of This Work

The purpose of this work is to detect employees that will quit the job beforehand. This project is useful for employers to take precautions before employees leave their jobs.

1.2 Machine Learning

Machine learning is a new trend topic. Nowadays, this name is used for every technological development. One of the main motivations why we develop (computer) programs to automate various kinds of processes. Originally developed as a sub field of Artificial Intelligence (AI), one of the goals behind machine learning was to replace the need for developing computer programs “manually.” Considering that programs are being developed to automate processes, we can think of machine learning as the process of “automating automation.” In other words, machine learning lets computers “create” programs (often, the intent for developing these programs is making predictions) themselves. In other words, machine learning is the process of turning data into programs.

Briefly, machine Learning is a branch of artificial intelligence that gives systems the ability to learn automatically and improve themselves from the experience without being explicitly programmed or without the intervention of human. Its main aim is to make computers learn automatically from the experience.

Today, machine learning algorithms enable computers to communicate with

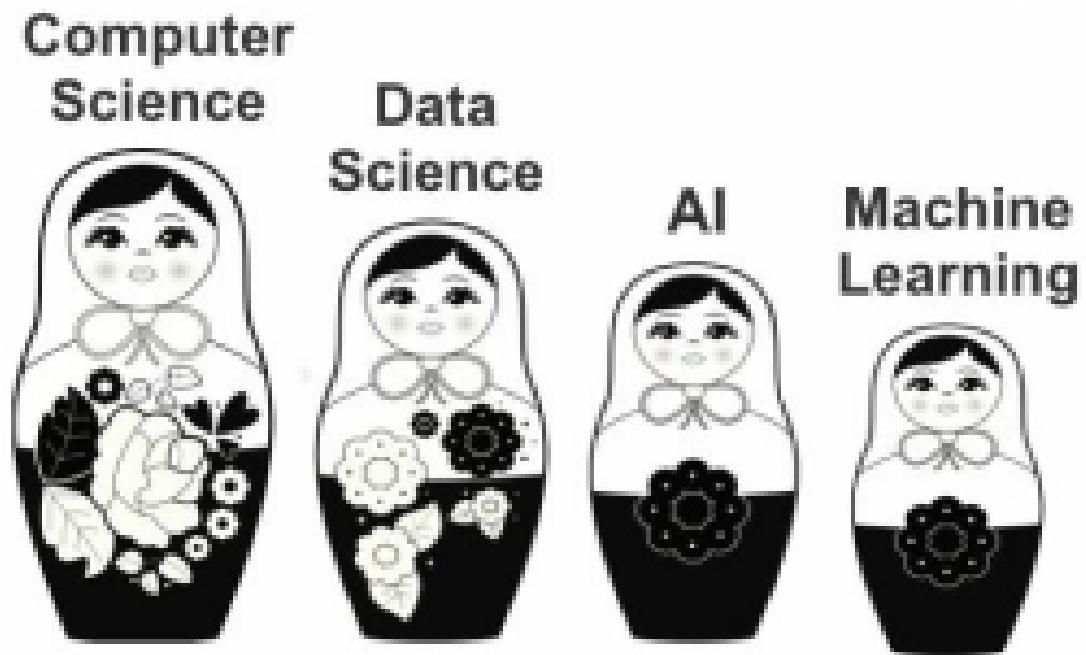


Figure 1.1 The lineage of machine learning represented by a row of Russian matryoshka dolls
[1]

humans, autonomously drive cars, write and publish sport match reports, and find terrorist suspects. This topic will be discussed in next chapters detailedly.

1.2.1 History Of Machine Learning

In 1946 the first computer system ENIAC was developed. At that time the word 'computer' meant a human being that performed numerical computations on paper and ENIAC was called a numerical computing machine. This machine was manually operated, i.e. a human would make connections between parts of the machine to perform computations. The idea at that time was that human thinking and learning could be rendered logically in such a machine.

In 1950 Alan Turing proposed a test to measure its performance. The Turing test is based on the idea that we can only determine if a machine can actually learn if we communicate with it and cannot distinguish it from another human. Although, there have not been any systems that passed the Turing test many interesting systems have been developed.

Around 1952 Arthur Samuel (IBM) wrote the first game-playing program, for checkers, to achieve sufficient skill to challenge a world champion. Samuel's machines learning programs worked remarkably well and were helpful in improving the performance of checkers players. Another milestone was the ELIZA system developed

in the early 60's by Joseph Weizenbaum. ELIZA simulated a psychotherapist by using tricks like string substitution and canned responses based on keywords. When the original ELIZA first appeared, some people actually mistook her for human.

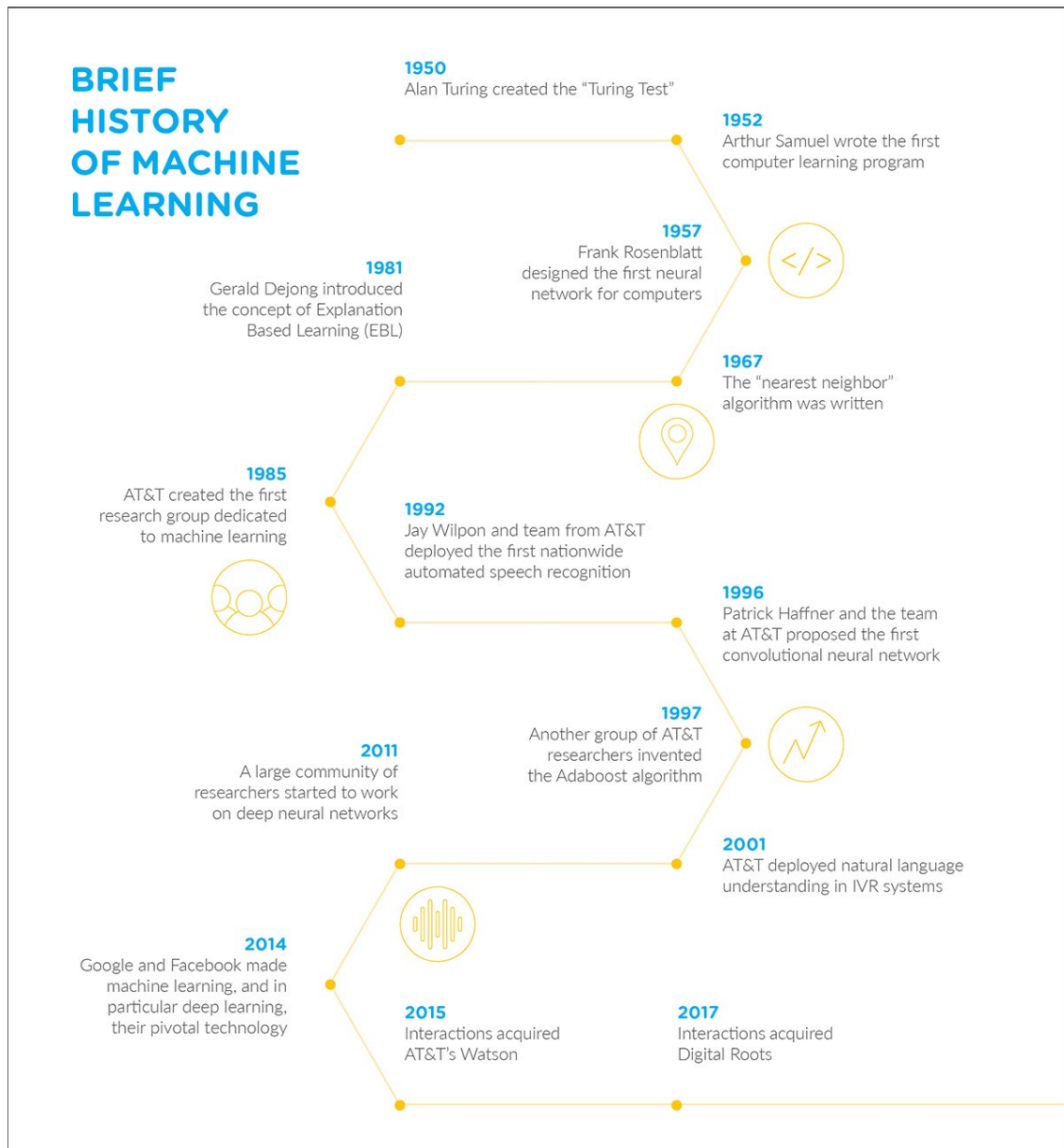


Figure 1.2 Brief History of Machine Learning
[2]

The illusion of intelligence works best, however, if you limit your conversation to talking about yourself and your life. Although the overall performance of ELIZA was disappointing, it was a nice proof of concept. Later on many other systems have been developed. Important was the work of the group of Ted Shortliffe on MYCIN (Stanford). They demonstrated the power of rule-based systems for knowledge representation and inference in the domain of medical diagnosis and therapy. This system is often called the first expert system.

At the same time when the expert systems were developed, other approaches to Machine Learning emerged. In 1957 Frank Rosenblatt invented the Perceptron at the Cornell Aeronautical Laboratory. The Perceptron is a very simple linear classifier but it was shown that by combining a large number of them in a network a powerful model could be created.

Neural network research went through many years of stagnation after Marvin Minsky and his colleagues showed that neural networks could not solve problems such as the XOR problem. However, several modifications have been produced later on that solve XOR and many more difficult problems.

In the early 90's Machine Learning became very popular again due to the intersection of Computer Science and Statistics. This synergy resulted in a new way of thinking in AI: the probabilistic approach. In this approach uncertainty in the parameters is incorporated in the models. The field shifted to a more data-driven approach as compared to the more knowledge-driven expert systems developed earlier. Many of the current success stories of Machine Learning are the result of the ideas developed at that time.[3]

Statistical AI is a center piece of Big Data analysis: as a result of the exponential growth in the amount of data that is available for scientific research, the sciences are on the brink of huge changes. That applies to all disciplines. In biology, for example, there will shortly be around 1 Exabyte of genomics data (10 to the power of 18 bytes) in the world. In 2024 the next generation of radio telescopes will produce in excess of 1 Exabyte per day. To deal with this data deluge, a new scientific discipline is taking shape. Big Data Science aims to develop new methods to store those substantial amounts, and to quickly find, analyze and validate complex patterns in Big Data.

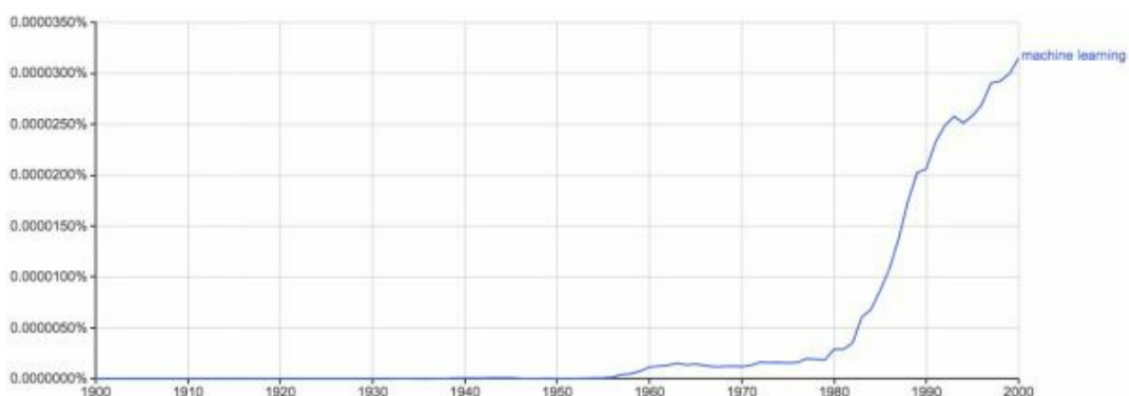


Figure 1.3 Historical mentions of “machine learning” in published books . Source: Google Ngram Viewer, 2017

In current status ,the study of Machine Learning has grown from the efforts of a

handful of computer engineers exploring whether computers could learn to play games and mimic the human brain, and a field of statistics that largely ignored computational considerations, to a broad discipline that has produced fundamental statistical-computational theories of learning processes.

1.2.2 Machine Learning Systems Working Principles

To create Machine Learning systems , some things are required. These are data (Input data is required for predicting the output), algorithms (Machine Learning is dependent on certain statistical algorithms to determine data patterns), iteration (The complete process is iterative i.e. repetition of process), scalability (The capacity of the machine can be increased or decreased in size and scale), automation (It is the ability to make systems operate automatically), modeling (The models are created according to the demand by the process of modeling). Classical programming works with input, command, action and outputs but in Machine Learning systems work with input data, model, action then outputs.

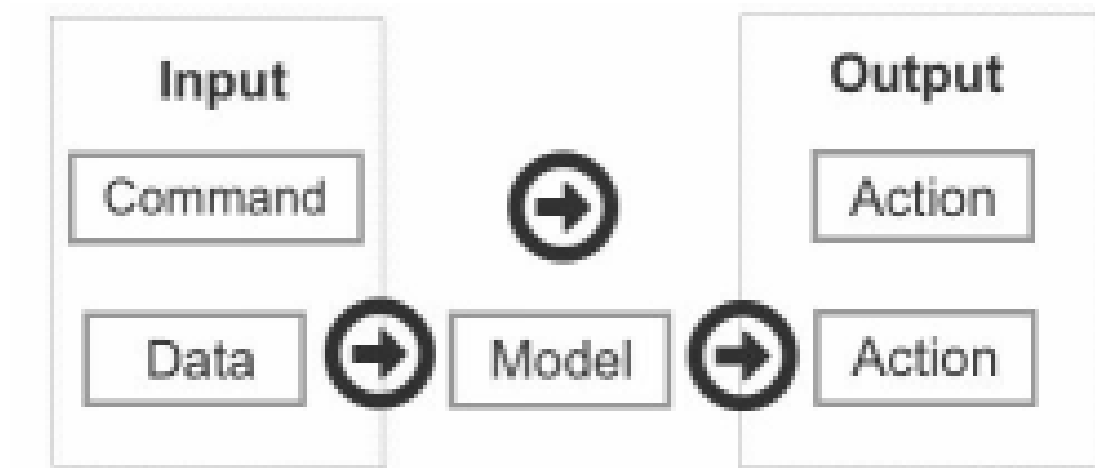


Figure 1.4 Comparison of Input Command vs Input Data
[4]

1.2.3 Machine Learning Categories

Selected method is changed with input data and purpose of the work. Before the method selecting , the input data is reviewed comprehensively. Then compare methods with each other to find most suitable way. Before the selecting algorithms and methods, we should understand these algorithms and methods. Machine Learning has three main categories and a lot of algorithms. These three categories are supervised, unsupervised and reinforcement.

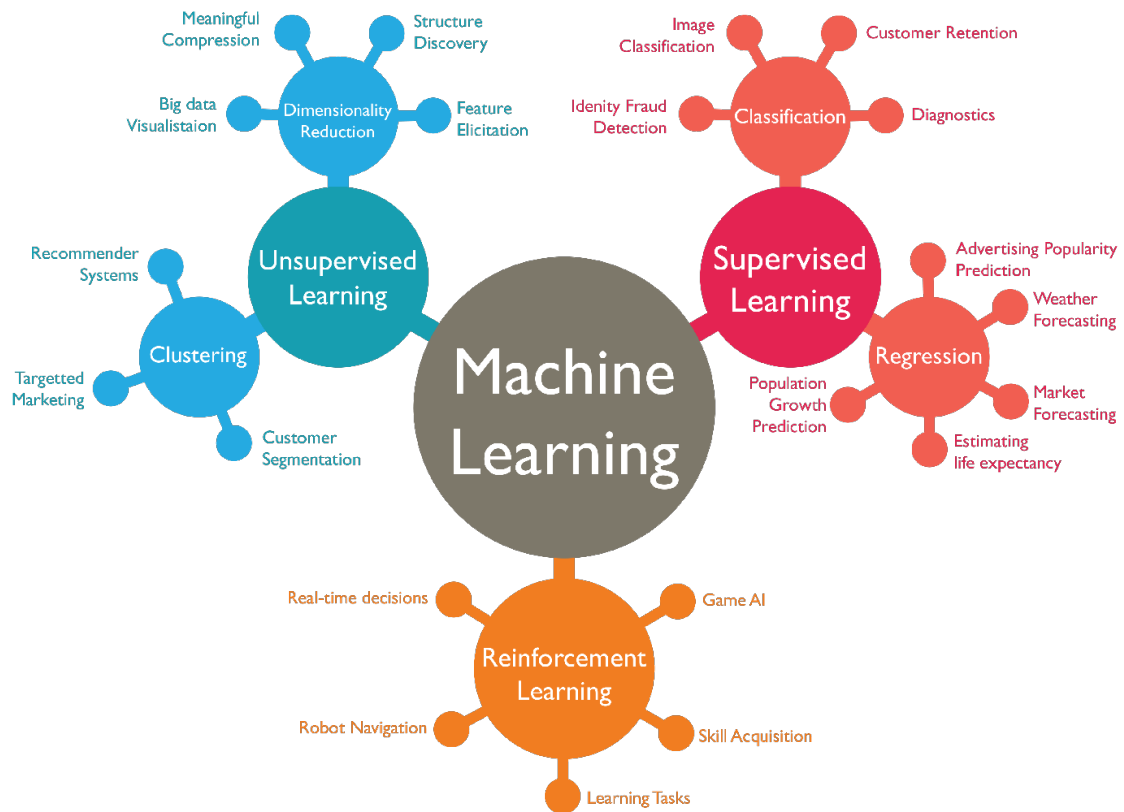


Figure 1.5 Machine Learning Categories and Algorithms
[5]

1.2.3.1 Supervised Learning

Supervised learning methods or algorithms include learning algorithms that take in data samples (known as training data) and associated outputs (known as labels or responses) with each data sample during the model training process. The main objective is to learn a mapping or association between input data samples x and their corresponding outputs y based on multiple training data instances. This learned knowledge can then be used in the future to predict an output y for any new input data sample x which was previously unknown or unseen during the model training process. These methods are termed as supervised because the model learns on data samples where the desired output responses/labels are already known beforehand in the training phase. Supervised learning basically tries to model the relationship between the inputs and their corresponding outputs from the training data so that we would be able to predict output responses for new data inputs based on the knowledge it gained earlier with regard to relationships and mappings between the inputs and their target outputs. This is precisely why supervised learning methods are extensively used in predictive analytics where the main objective is to predict some response for some input data that's typically fed into a trained supervised ML model. Supervised learning methods are of two major classes based on the type of ML tasks they aim to solve.

- Classification
- Regression

Look these two Machine Learning tasks and observe the subset of supervised learning methods that are best suited for tackling these tasks

Classification

The classification based tasks are a sub-field under supervised Machine Learning, where the key objective is to predict output labels or responses that are categorical in nature for input data based on what the model has learned in the training phase. Output labels here are also known as classes or class labels as these are categorical in nature meaning they are unordered and discrete values. Thus, each output response belongs to a specific discrete class or category.

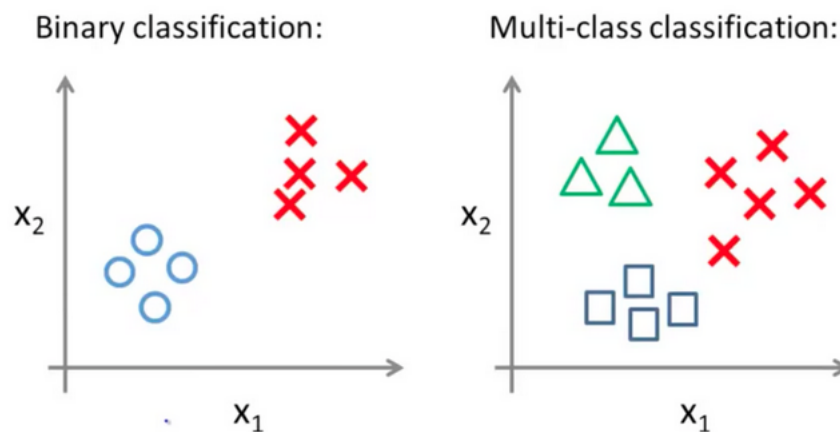


Figure 1.6 Types of Classification
[6]

Popular classification algorithms include logistic regression, support vector machines, neural networks, ensembles like random forests and gradient boosting, K-nearest neighbors, decision trees, and many more.

Regression

Machine Learning tasks where the main objective is value estimation can be termed as regression tasks. Regression based methods are trained on input data samples having output responses that are continuous numeric values unlike classification, where we have discrete categories or classes. Regression models make use of input data attributes or features (also called explanatory or independent variables) and their corresponding continuous numeric output values (also called as response, dependent, or outcome variable) to learn specific relationships and associations between the inputs and their corresponding outputs. With this knowledge, it can predict output

responses for new, unseen data instances similar to classification but with continuous numeric outputs.

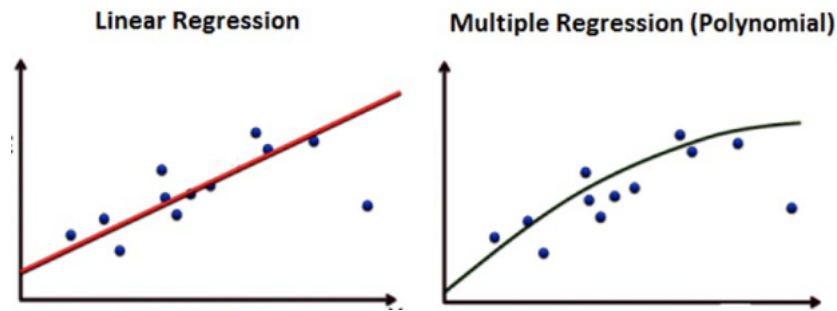


Figure 1.7 Supervised learning: regression models
[7]

1.2.3.2 Unsupervised Learning

Supervised learning methods usually require some training data where the outcomes which we are trying to predict are already available in the form of discrete labels or continuous values. However, often we do not have the liberty or advantage of having pre-labeled training data and we still want to extract useful insights or patterns from our data. In this scenario, unsupervised learning methods are extremely powerful. These methods are called unsupervised because the model or algorithm tries to learn inherent latent structures, patterns and relationships from given data without any help or supervision like providing annotations in the form of labeled outputs or outcomes.

Unsupervised learning is more concerned with trying to extract meaningful insights or information from data rather than trying to predict some outcome based on previously available supervised training data. There is more uncertainty in the results of unsupervised learning but you can also gain a lot of information from these models that was previously unavailable to view just by looking at the raw data. Often unsupervised learning could be one of the tasks involved in building a huge intelligence system. For example, we could use unsupervised learning to get possible outcome labels for tweet sentiments by using the knowledge of the English vocabulary and then train a supervised model on similar data points and their outcomes which we obtained previously through unsupervised learning. There is no hard and fast rule with regard to using just one specific technique. You can always combine multiple methods as long as they are relevant in solving the problem. Unsupervised learning methods can be categorized under the following broad areas of ML tasks relevant to unsupervised learning.

- Clustering

- Dimensionality reduction
- Anomaly detection
- Association rule-mining

Learning Category	Learning Method
Supervised Learning	
K-Nearest Neighbor	Classification
Decision Trees	Classification
Classification Rule Learners	Numeric prediction
Linear Regression	Numeric prediction
Regression Trees	Numeric prediction
Unsupervised Learning	
Association Rules	Pattern Detection
K-means clustering	Clustering

Figure 1.8 Common algorithms used to perform supervised and unsupervised machine learning

1.2.3.3 Reinforcement Learning

The reinforcement learning methods are a bit different from conventional supervised or unsupervised methods. In this context, we have an agent that we want to train over a period of time to interact with a specific environment and improve its performance over a period of time with regard to the type of actions it performs on the environment. Typically the agent starts with a set of strategies or policies for interacting with the environment. On observing the environment, it takes a particular action based on a rule or policy and by observing the current state of the environment. Based on the action, the agent gets a reward, which could be beneficial or detrimental in the form of a penalty. It updates its current policies and strategies if needed and this iterative process continues till it learns enough about its environment to get the desired rewards. The main steps of a reinforcement learning method are mentioned as follows.

1. Prepare agent with set of initial policies and strategy
2. Observe environment and current state
3. Select optimal policy and perform action
4. Get corresponding reward (or penalty)
5. Update policies if needed
6. Repeat Steps 2 - 5 iteratively until agent learns the most optimal policies

Consider a real-world problem of trying to make a robot or a machine learn to play chess. In this case the agent would be the robot and the environment and states would be the chessboard and the positions of the chess pieces.[7]

1.3 Python

For this project ,python programming language is used.This part will touch some main topics of using Python for Machine Learning projects.

1.3.1 Introduction the Python

Python is a popular object-oriented programming language having the capabilities of high-level programming language.It was developed by Guido van Rossum at Stichting Mathematisch Centrum in the Netherlands.It was written as the successor of programming language named 'ABC' and first version was released in 1991. The name Python was picked by Guido van Rossum from a TV show named Monty Python's Flying Circus. It is an open source programming language.It is an interpreted language, which means the source code of Python program would be first converted into bytecode and then executed by Python virtual machine.

1.3.2 Using Python for Machine Learning Projects

Python is currently the most popular programming language for research and development in Machine Learning.It has some reasons to be most popular in Machine Learning Projects.According to a lot of web sites, this reasons are

- Python is Easy To Use
- Python has multiple Libraries and Frameworks
 - Keras
 - TensorFlow
 - Scikit-learn
- Python has Community and Corporate Support
- Python is Portable and Extensible

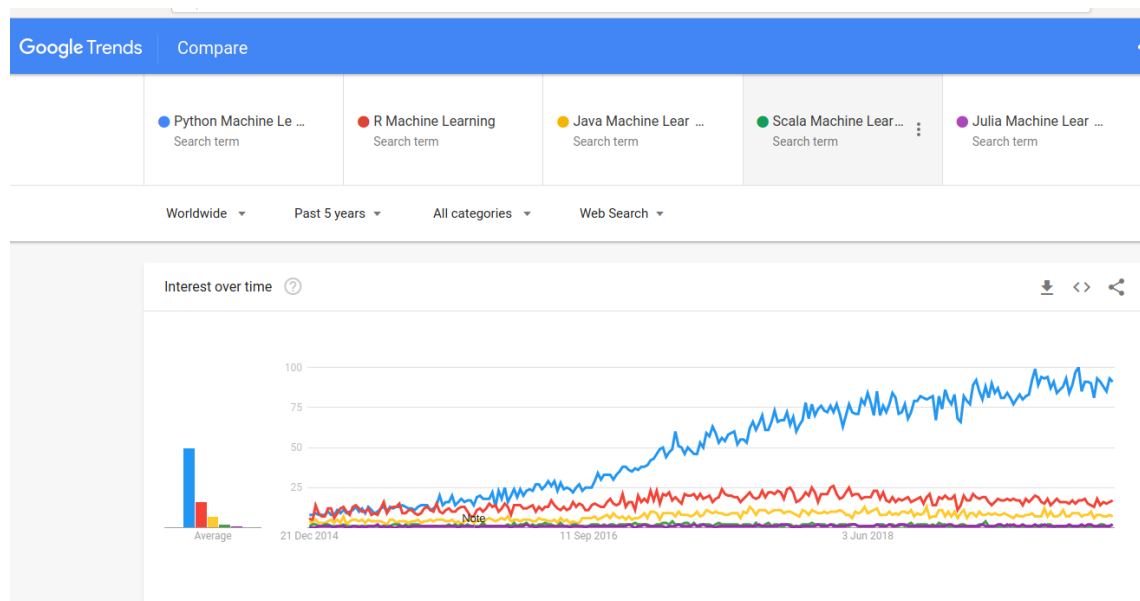


Figure 1.9 Comparison of Most Popular Languages are used for Machine Learning [8]

1.3.2.1 Useful Python Libraries for ML

Implementing ML and AI algorithms require a well-structured and well-tested environment to empower developers to come up with the best quality coding solutions. To reduce development time, there are countless Python libraries for machine learning. Python library or framework is a pre-written program that is ready to use on common coding tasks. Let us become familiar with the best Python machine learning libraries:

- **TensorFlow:** TensorFlow is an end-to-end python machine learning library for performing high-end numerical computations. TensorFlow can handle deep neural networks for image recognition, handwritten digit classification, recurrent neural networks, NLP (Natural Language Processing), word embedding and PDE (Partial Differential Equation). TensorFlow Python ensures excellent architecture support to allow easy computation deployments across a wide range of platforms, including desktops, servers, and mobile devices.

Abstraction is the major benefit of TensorFlow Python towards machine learning and AI projects. This feature allows the developers to focus on comprehensive logic of the app instead of dealing with the mundane details of implementing algorithms. With this library, python developers can now effortlessly leverage AI and ML to create unique responsive applications, which responds to user inputs like facial or voice expression.

- **Keras Python:** Keras is a leading open-source Python library written for constructing neural networks and machine learning projects. It can run

on DeepLearning4j, MXNet, Microsoft Cognitive Toolkit (CNTK), Theano or TensorFlow. It offers almost all standalone modules including optimizers, neural layers, activation functions, initialization schemes, cost functions, and regularization schemes. It makes it easy to add new modules just like adding new functions and classes. As the model is already defined in the code, you don't need to have a separate model config files.

Keras makes it simple for machine learning beginners to design and develop a neural network. Keras Python also deals with convolution neural networks. It includes algorithms for normalization, optimizer, and activation layers. Instead of being an end-to-end Python machine learning library, Keras functions as a user-friendly, extensible interface that enhances modularity total expressiveness.

- **Theano Python:** Since its arrival in 2007, Theano has captured the Python developers and researchers of ML and AI. At the core, it is a well-known scientific computing library that allows you to define, optimize as well as evaluate mathematical expressions, which deals with multidimensional arrays. The fundamental of several ML and AI applications is the repetitive computation of a tricky mathematical expression. Theano allows you to make data-intensive calculation up to a hundred times faster than when executing on your CPU alone. Additionally, it is well optimized for GPUs, which offers effective symbolic differentiation and includes extensive code-testing capabilities.

When it comes to performance, Theano is a great Python machine learning library as it includes the ability to deal with computations in large neural networks. It aims to boost development time and execution time of ML apps, particularly in deep learning algorithms. Only one drawback of Theano in front of TensorFlow is that its syntax is quite hard for the beginners.

- **Scikit-learn Python:** Scikit-learn is another prominent open-source Python machine learning library with a broad range of clustering, regression and classification algorithms. DBSCAN, gradient boosting, random forests, vector machines, and k-means are a few examples. It can interoperate with numeric and scientific libraries of Python like NumPy and SciPy.

It is a commercially usable artificial intelligence library. This Python library supports both supervised as well as unsupervised ML. Here is a list of the premier benefits of Scikit-learn Python that makes it one among the most preferable Python libraries for machine learning: Reduction of dimensionality , Decision tree pruning and induction, Decision boundary learning, Feature analysis and selection, Outlier detection and rejection, Advanced probability

modeling, Unsupervised classification and clustering.

- **NumPy:** Almost all Python machine-learning packages like Matplotlib, SciPy, Scikit-learn, etc rely on this library to a reasonable extent. It comes with functions for dealing with complex mathematical operations like linear algebra, Fourier transformation, random number and features that work with matrices and n-arrays in Python. NumPy Python package also performs scientific computations. It is widely used in handling sound waves, images, and other binary functions.
- **Python Pandas:** In machine learning projects, a substantial amount of time is spent on preparing the data as well as analyzing basic trends and patterns. This is where the Python Pandas receives machine learning experts' attention. Python Pandas is an open-source library that offers a wide range of tools for data manipulation analysis. With this library, you can read data from a broad range of sources like CSV, SQL databases, JSON files, and Excel.

It enables you to manage complex data operation with just one or two commands. Python Pandas comes with several inbuilt methods for combining data, and grouping and filtering time-series functionality. Overall, Pandas is not just limited to handle data-related tasks; it serves as the best starting point to create more focused and powerful data tools.

- **Seaborn Python:** the last library in the list of Python libraries for machine learning and AI is Seaborn – an unparalleled visualization library, based on Matplotlib's foundations. Both storytelling and data visualization are important for machine learning projects, as they often require exploratory analysis of datasets to decide on the type of machine learning algorithm to apply. Seaborn offers high-level dataset based interface to make amazing statistical graphics.

With this Python machine learning library, it is simple to create certain types of plots like time series, heat maps, and violin plots. The functionalities of Seaborn go beyond Python Pandas and matplotlib with the features to perform statistical estimation at the time of combining data across observations, plotting and visualizing the suitability of statistical models to strengthen dataset patterns.

1.3.2.2 ML Algorithms on Python

Brief History of ML Algorithms:

The roots of machine learning algorithms come from Thomas Bayes, who was English statistician who lived in the 18th century. His paper An Essay Towards Solving a

Problem in the Doctrine of Chances underpins Bayes' Theorem, which is widely applied in the field of statistics.

In the 19th century, Pierre-Simon Laplace published *Théorie analytique des probabilités*, expanding on the work of Bayes and defining what we know of today as Bayes' Theorem. Shortly before that, Adrien-Marie Legendre had described the “least squares” method, also widely used today in supervised learning.

The 20th century is the period when the majority of publicly known discoveries have been made in this field. Andrey Markov invented Markov chains, which he used to analyze poems. Alan Turing proposed a learning machine that could become artificially intelligent, basically foreshadowing genetic algorithms. Frank Rosenblatt invented the Perceptron, sparking huge excitement and great coverage in the media.

But then the 1970s saw a lot of pessimism around the idea of AI—and thus, reduced funding—so this period is called an AI winter. The rediscovery of backpropagation in the 1980s caused a resurgence in machine learning research. And today, it's a hot topic once again.

The late Leo Breiman distinguished between two statistical modeling paradigms: Data modeling and algorithmic modeling. “Algorithmic modeling” means more or less the machine learning algorithms like the random forest.

Machine learning and statistics are closely related fields. According to Michael I. Jordan, the ideas of machine learning, from methodological principles to theoretical tools, have had a long prehistory in statistics. He also suggested data science as a placeholder term for the overall problem that machine learning specialists and statisticians are both implicitly working on.

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

- Linear Regression
- Logistic Regression
- Decision Tree
- SVM
- Naive Bayes
- kNN

- K-Means
- Random Forest
- Dimensionality Reduction Algorithms
- Gradient Boosting algorithms(GBM,XGBoost,LightGBM,CatBoost)

Some algorithms in list will be explained.

1. Linear Regression:

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation $Y = a * X + b$.

The best way to understand linear regression is to relive this experience of childhood. Let us say, you ask a child in fifth grade to arrange people in his class by increasing order of weight, without asking them their weights! What do you think the child will do? He / she would likely look (visually analyze) at the height and build of people and arrange them using a combination of these visible parameters. This is linear regression in real life! The child has actually figured out that height and build would be correlated to the weight by a relationship, which looks like the equation above.

In this equation:

- Y – Dependent Variable
- a – Slope
- X – Independent variable
- b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Look at the below example. Here we have identified the best fit line having linear equation $y = 0.2811x + 13.9$. Now using this equation, we can find the weight, knowing the height of a person.

Linear Regression is mainly of two types: Simple Linear Regression and Multiple Linear Regression. Simple Linear Regression is characterized by one independent

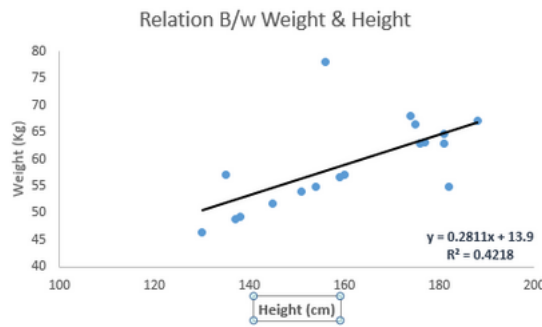


Figure 1.10 Linear Regression
[9]

variable. And, Multiple Linear Regression(as the name suggests) is characterized by multiple (more than 1) independent variables. While finding the best fit line, you can fit a polynomial or curvilinear regression. And these are known as polynomial or curvilinear regression.

2.Logical Regression:

It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variable(s). In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as logit regression. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).

Again, let us try and understand this through a simple example.

Let's say your friend gives you a puzzle to solve. There are only 2 outcome scenarios – either you solve it or you don't. Now imagine, that you are being given wide range of puzzles / quizzes in an attempt to understand which subjects you are good at. The outcome to this study would be something like this – if you are given a trigonometry based tenth grade problem, you are 70% likely to solve it. On the other hand, if it is grade fifth history question, the probability of getting an answer is only 30%.This is what Logistic Regression provides you.

Coming to the math, the log odds of the outcome is modeled as a linear combination of the predictor variables.

$$\begin{aligned} \text{odds} &= p / (1-p) = \text{probability of event occurrence} \\ &\quad / \text{probability of not event occurrence} \\ \ln(\text{odds}) &= \ln(p/(1-p)) \\ \text{logit}(p) &= \ln(p/(1-p)) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 \dots + b_kX_k \end{aligned}$$

Above, p is the probability of presence of the characteristic of interest. It chooses parameters that maximize the likelihood of observing the sample values rather than that minimize the sum of squared errors (like in ordinary regression).

3. Decision Tree:

It is a type of supervised learning algorithm that is mostly used for classification problems. Surprisingly, it works for both categorical and continuous dependent variables. In this algorithm, we split the population into two or more homogeneous sets. This is done based on most significant attributes/ independent variables to make as distinct groups as possible.

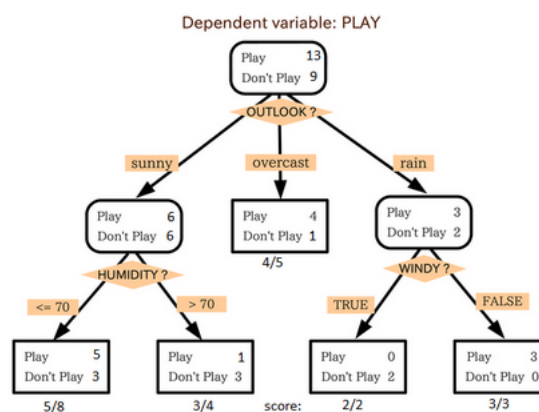


Figure 1.11 Decision Tree
[9]

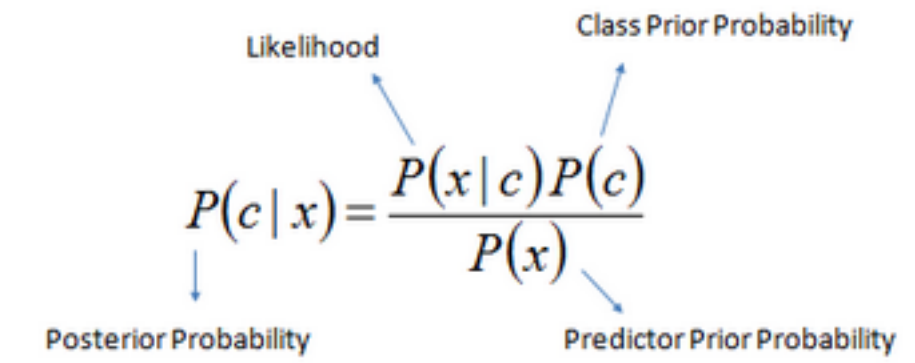
In the image above, you can see that population is classified into four different groups based on multiple attributes to identify 'if they will play or not'. To split the population into different heterogeneous groups, it uses various techniques like Gini, Information Gain, Chi-square, entropy.

4. Naive Bayes:

It is a classification technique based on Bayes' theorem with an assumption of independence between 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. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier would consider all of these properties to independently contribute to the probability that this fruit is an apple.

Naive Bayesian 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:



$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

Figure 1.12 Naive Bayes
[9]

Here,

- $P(c|x)$ is the posterior probability of class (target) given predictor (attribute).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

6.kNN (k- Nearest Neighbors): It can be used for both classification and regression problems. However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function.

These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If $K = 1$, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing kNN modeling.

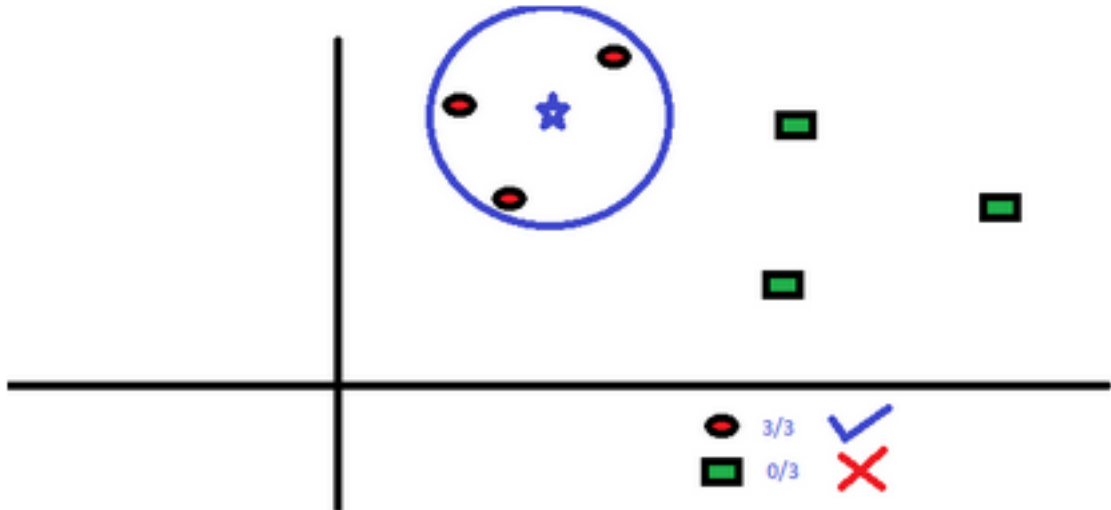


Figure 1.13 kNN
[9]

KNN can easily be mapped to our real lives. If you want to learn about a person, of whom you have no information, you might like to find out about his close friends and the circles he moves in and gain access to his/her information!

Things to consider before selecting kNN:

- KNN is computationally expensive
- Variables should be normalized else higher range variables can bias it
- Works on pre-processing stage more before going for kNN like an outlier, noise removal

7.Random Forest: Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we've collection of decision trees (so known as "Forest"). To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is planted and grown as follows:

- If the number of cases in the training set is N , then sample of N cases is taken at random but with replacement. This sample will be the training set for growing the tree.

- If there are M input variables, a number $m \ll M$ is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
- Each tree is grown to the largest extent possible. There is no pruning.

8.K-Means: It is a type of unsupervised algorithm which solves the clustering problem. Its procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). Data points inside a cluster are homogeneous and heterogeneous to peer groups.

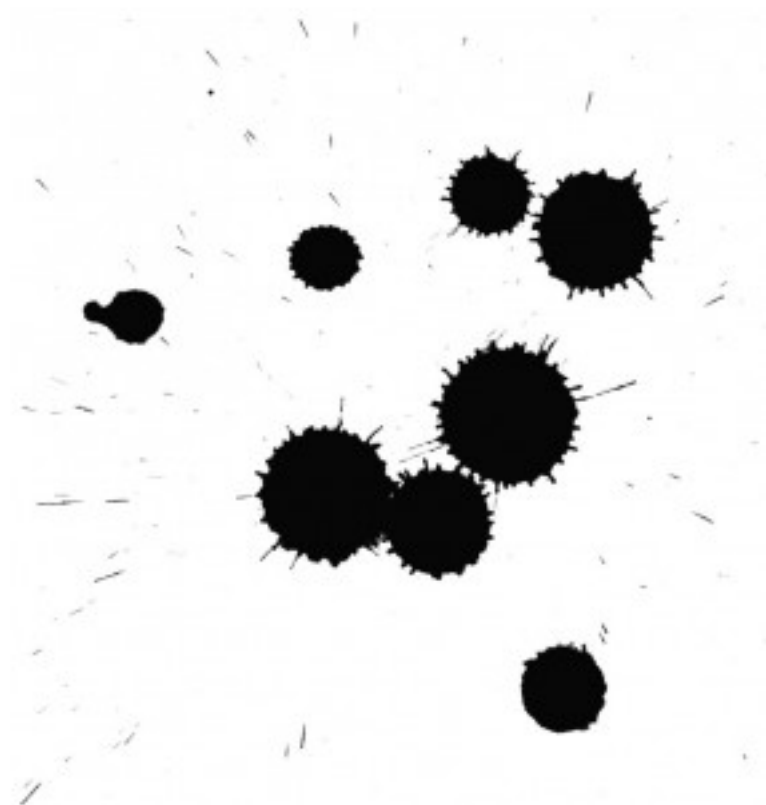


Figure 1.14 K-Means
[9]

How K-means forms cluster:

- 1.K-means picks k number of points for each cluster known as centroids.
- 2.Each data point forms a cluster with the closest centroids i.e. k clusters.
- 3.Finds the centroid of each cluster based on existing cluster members. Here we have new centroids.

- 4. As we have new centroids, repeat step 2 and 3. Find the closest distance for each data point from new centroids and get associated with new k-clusters. Repeat this process until convergence occurs i.e. centroids does not change.

How to determine value of K:

In K-means, we have clusters and each cluster has its own centroid. Sum of square of difference between centroid and the data points within a cluster constitutes within sum of square value for that cluster. Also, when the sum of square values for all the clusters are added, it becomes total within sum of square value for the cluster solution.

We know that as the number of cluster increases, this value keeps on decreasing but if you plot the result you may see that the sum of squared distance decreases sharply up to some value of k, and then much more slowly after that. Here, we can find the optimum number of cluster.

[9]

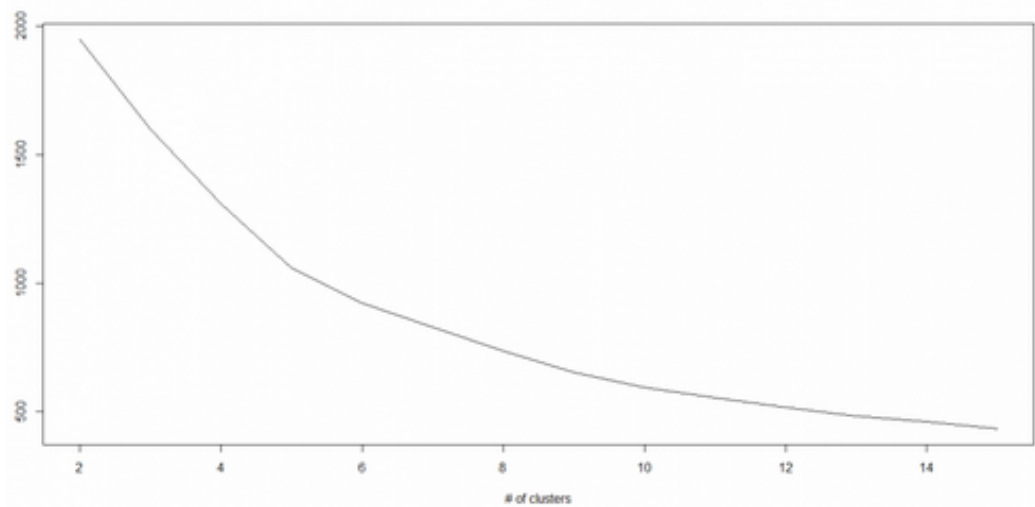


Figure 1.15 K-Means
[9]

2 Supervised Learning Method in ML Using HR Analytic Dataset

2.1 Data Scrubbing and Data Preparation

Data scrubbing, also called data cleansing, is a very important step before applying any machine learning algorithm. Dataset are normally drawn from real world sources which produce large amounts of messy datasets. Examples of sources are statistics, or massive amounts of records generated from organizations that work in data-intensive fields such as banking, communication systems, insurance, or transportation. These messy data might have noisy entries, missing data or have records that contradicts with each other. There are several reasons for why this may happen.

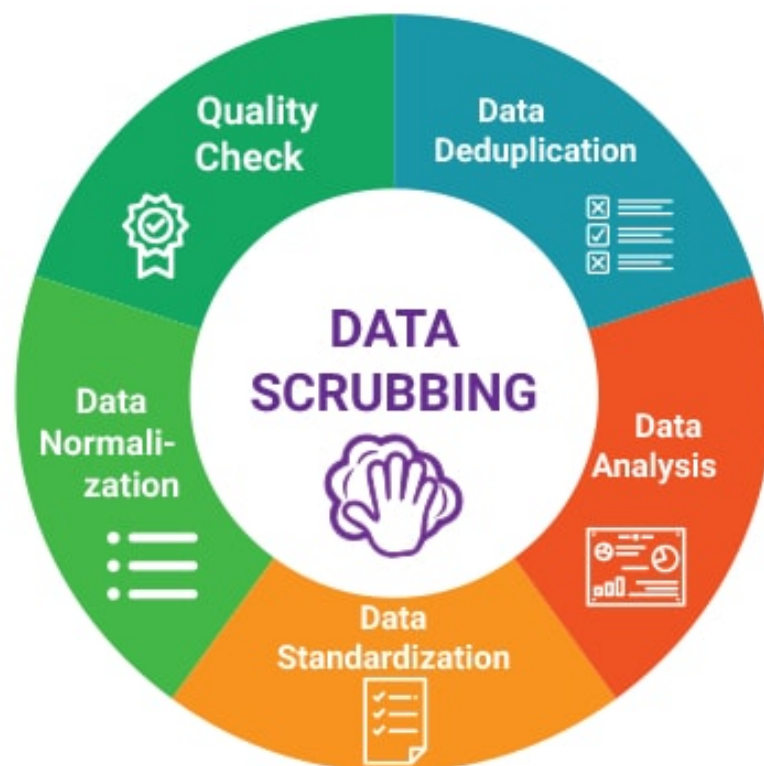


Figure 2.1 Data Scrubbing
[10]

Noise usually affects the data because of the hardware limitations and problems, as these can act as noise sources. For example, if a blood sample is tested by a medical device that encounters a problem, the device measurements may be affected and vary on each run for the same sample. If the device is connected to a database via a network, the unstable readings may be transferred automatically to the database regardless of the erroneous that happened. Noisy data can also be generated by human when he makes faults. Missing data problem, on the other hand, may occur due to technical issues, such as a server or a network hang during data transfer, or because of manual data entry. These errors combined; noisy data and missing data problems, may lead also to a third class of messy data, called: data contradiction. Based on that, the data scrubbing step is very important before starting the learning process.

2.1.1 Noisy Data

When there is a difference between the model and the measurements, there exists a “noise” that causes an error or variance. Examples of methods that deal with noisy datasets are as follow:

- Binning methods: In this method, the data is sorted and then smoothed by considering its neighborhood or surrounding values. This is called a local smoothing.
- Clustering: This method is useful in detecting and removing the outliers. Clustering each set of similar values means that the values located in one cluster are different from those in the other clusters. The remaining outliers can then be easily detected.
- Machine learning Algorithms: Regression, one of the basic supervised machine learning algorithms, can be used to smooth data by fitting it into regression functions
- Human inspection: Human can interfere to manually detect and eliminate outliers or smooth noise.

2.1.2 Missing Data

The followings, are some common ways to deal with the missing data:

- Ignore the tuple: This method of healing a dataset with missing values is considered efficient, when a record of data contains many attributes with

missing values. However, when the amount of missing values per attribute varies significantly, this scheme is no more effective. Efficiency of this method decreases when a regulation of data is produced by an undefined source to the machine. The machine will deal with this kind of data as missing data. For example, when a patient is admitted according to unusual condition.

- Determine and fill in the missing value manually: This scheme might have high accuracy. However, it is infeasible mainly in terms of time consumption. Also, it is tiring and tedious.
- Making Expectations: Usually, there are ways that one can use to predict a missing attribute. For instance, the average of the values nearby of the missing values. Although this way can cause a bias in the data because of mis-predictions, but it can be utilized to check and compare its results, to the results obtained by the first method, ignoring the tuples.
- Inferring-based algorithms: This kind of algorithms are employed when a missing value is filled by the most probable value. Examples algorithms are Bayesian formula and decision tree.

2.1.3 Inconsistent Data

The real life massive amounts of data are susceptible to contain discrepancies or duplications in codes, names or any other values. The inconsistency caused by data entry can be fixed manually. Also, some expert system tools can be adopted to detect and fix contradicts in data. For example, data entered by thousands of individuals, like in health care organizations, are automatically recorded by cloud stored databases. This type of datasets is an obvious case of datasets that are full of inconsistencies. For instance, the same information may be entered in different formats by different sources.

2.2 Analysing Dataset

This dataset is acquired with CSV data file. CSV (comma-separated values) is a simple file format which is used to store tabular data (number and text) such as a spreadsheet in plain text. In Python, we can load CSV data into with different ways. After the download dataset, in Python:

Listing 2.1 Dataset Importing

```
import pandas as pd
dataset = pd.read_csv("C:\\Users\\Sumeyye\\Documents\\dataset.csv")
```

This cell, DataFrame in panda module read_csv is used for reading dataset which is in C:\\Users\\Sumeyye\\Documents\\dataset.csv path.

In this command , **dataset** is new variable that includes data.

Afterthat,used dataset is analyzed with some methods.

Listing 2.2 Dataset Analysing

```
dataset.head()
dataset.info()
dataset.shape
dataset.isnull().sum().sum()
```

This cell , **dataset.head()** displays first five rows in dataset.Five is default value , if 10 rows are wanted , just **dataset.head(10)** is typed.In this cell other command is **dataset.info()**.This command is used to display number of entries , number of colums and their names and datatypes and memory usage.

In this cell ,other command is **dataset.shape** .This command displays total number of rows and columns.Output of this command is (1470, 35).This dataset has 1470 rows and 35 columns.Other command is **dataset.isnull().sum().sum()**.This command returns a number of null values in each column.For this dataset , output is zero.All of these commands help to understand used dataset.

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...

5 rows × 35 columns



```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
Age                1470 non-null int64
Attrition          1470 non-null object
BusinessTravel     1470 non-null object
DailyRate         1470 non-null int64
Department        1470 non-null object
DistanceFromHome  1470 non-null int64
Education         1470 non-null int64
EducationField     1470 non-null object
EmployeeCount     1470 non-null int64
EmployeeNumber    1470 non-null int64
EnvironmentSatisfaction 1470 non-null int64
Gender            1470 non-null object
HourlyRate        1470 non-null int64
JobInvolvement    1470 non-null int64
JobLevel          1470 non-null int64
JobRole           1470 non-null object
JobSatisfaction   1470 non-null int64
MaritalStatus     1470 non-null object
MonthlyIncome     1470 non-null int64
MonthlyRate       1470 non-null int64
NumCompaniesWorked 1470 non-null int64
Over18           1470 non-null object
OverTime          1470 non-null object
PercentSalaryHike 1470 non-null int64
PerformanceRating 1470 non-null int64
RelationshipSatisfaction 1470 non-null int64
StandardHours     1470 non-null int64
StockOptionLevel  1470 non-null int64
TotalWorkingYears 1470 non-null int64
TrainingTimesLastYear 1470 non-null int64
WorkLifeBalance   1470 non-null int64
YearsAtCompany    1470 non-null int64
YearsInCurrentRole 1470 non-null int64
YearsSinceLastPromotion 1470 non-null int64
YearsWithCurrManager 1470 non-null int64
dtypes: int64(26), object(9)
memory usage: 402.1+ KB

: (1470, 35)
: 0

```

Figure 2.2 Output of Listing 2.2

According to solutions, this dataset contains a 'Yes/No' field indicating whether the employee has left the company but we are not given any information about the departure date. There are 34 features that describe each employee, his/her role in the company, his/her level of satisfaction, the salary and share options available and his/her influence on the company.

Sometimes, this methods do not enough to understand completely. Because some variables have numerical value but normally they have categorical value. For example, in this dataset, Education column can has 1,2,3,4,5 int values. Normally it has Below College, College, Bachelor, Master, Doctor values. This data set uses some alias about some variables. For example,

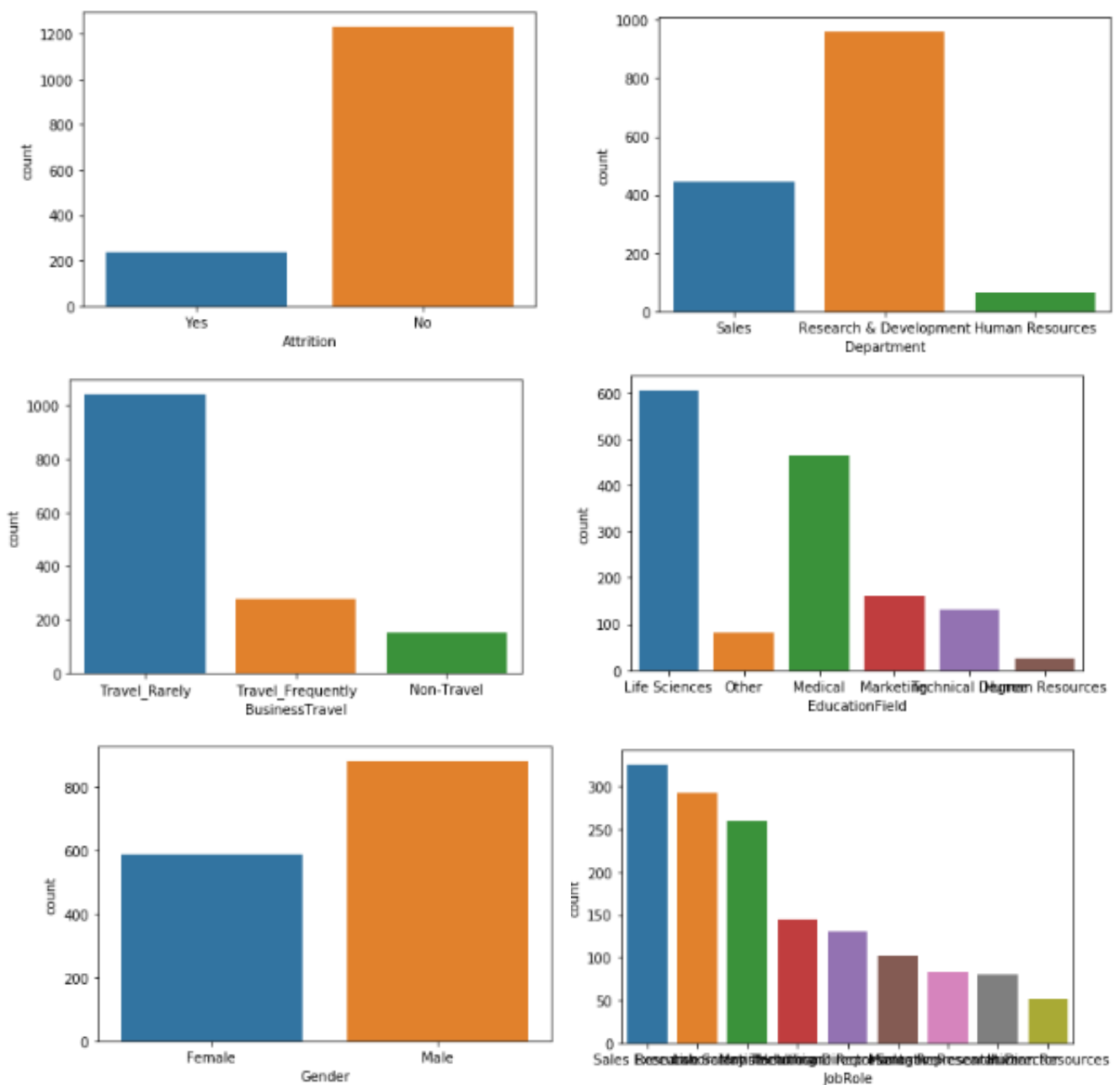
Education degrees (1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor')
EnvironmentSatisfaction (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
JobInvolvement (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
JobSatisfaction (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
PerformanceRating (1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding')
RelationshipSatisfaction (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
WorkLifeBalance

(1 'Bad' 2 'Good' 3 'Better' 4 'Best').These are explained by dataset providers.For this reason ,sources should be examined deeply.Some methods are used for draw graphs to data visualization.

Listing 2.3 Graph for Categorical Data

```
import seaborn as sns
sns.countplot(dataset.Attrition)
```

sns is alias of seaborn library , and countplot is used for categorical datas.This command means that draw a graph using countplot method from seaborn library and values from Attrition column in dataset. This command is repeated for each categorical data.



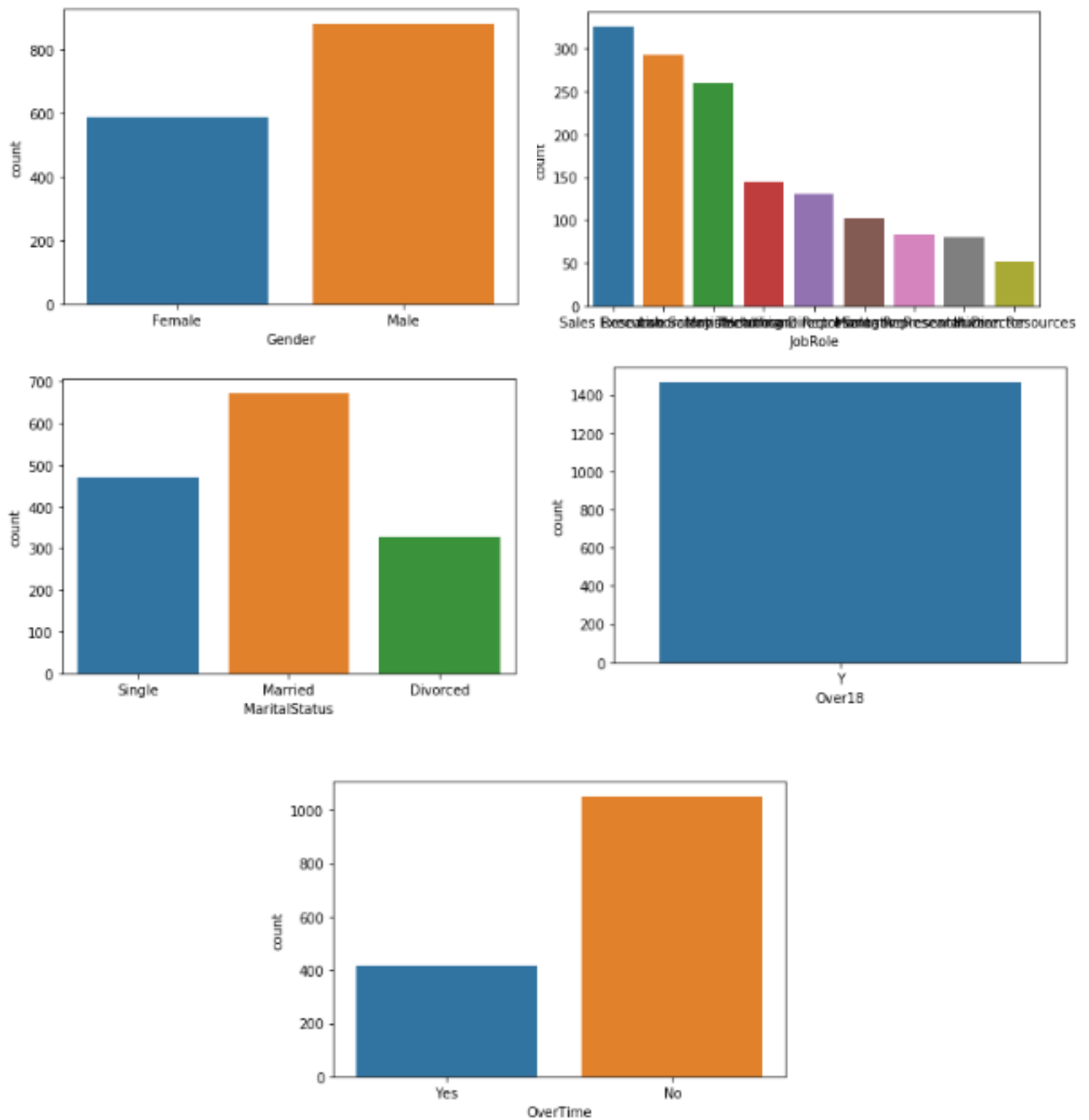


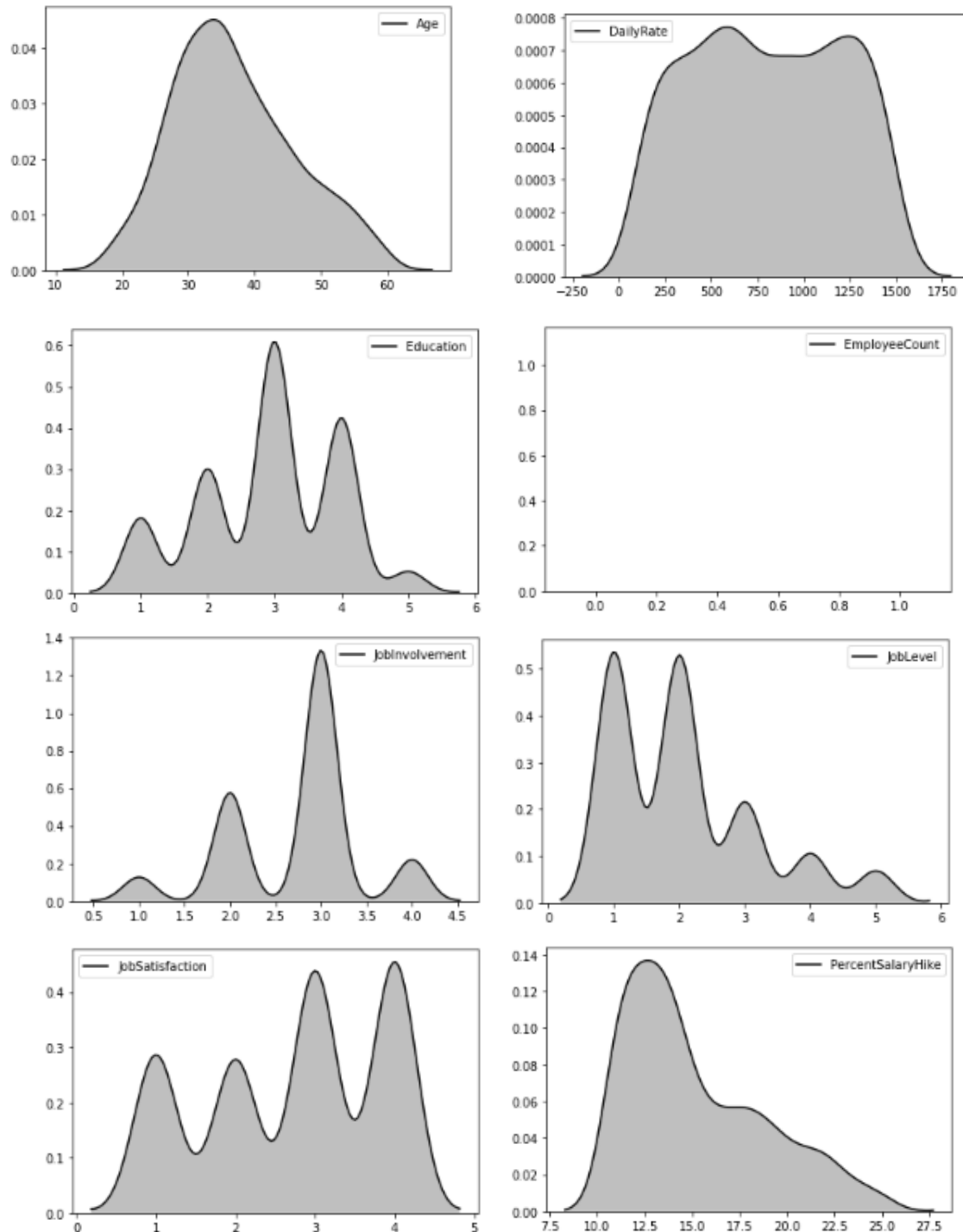
Figure 2.3 Output of Listing 2.3 for each Categorical Value

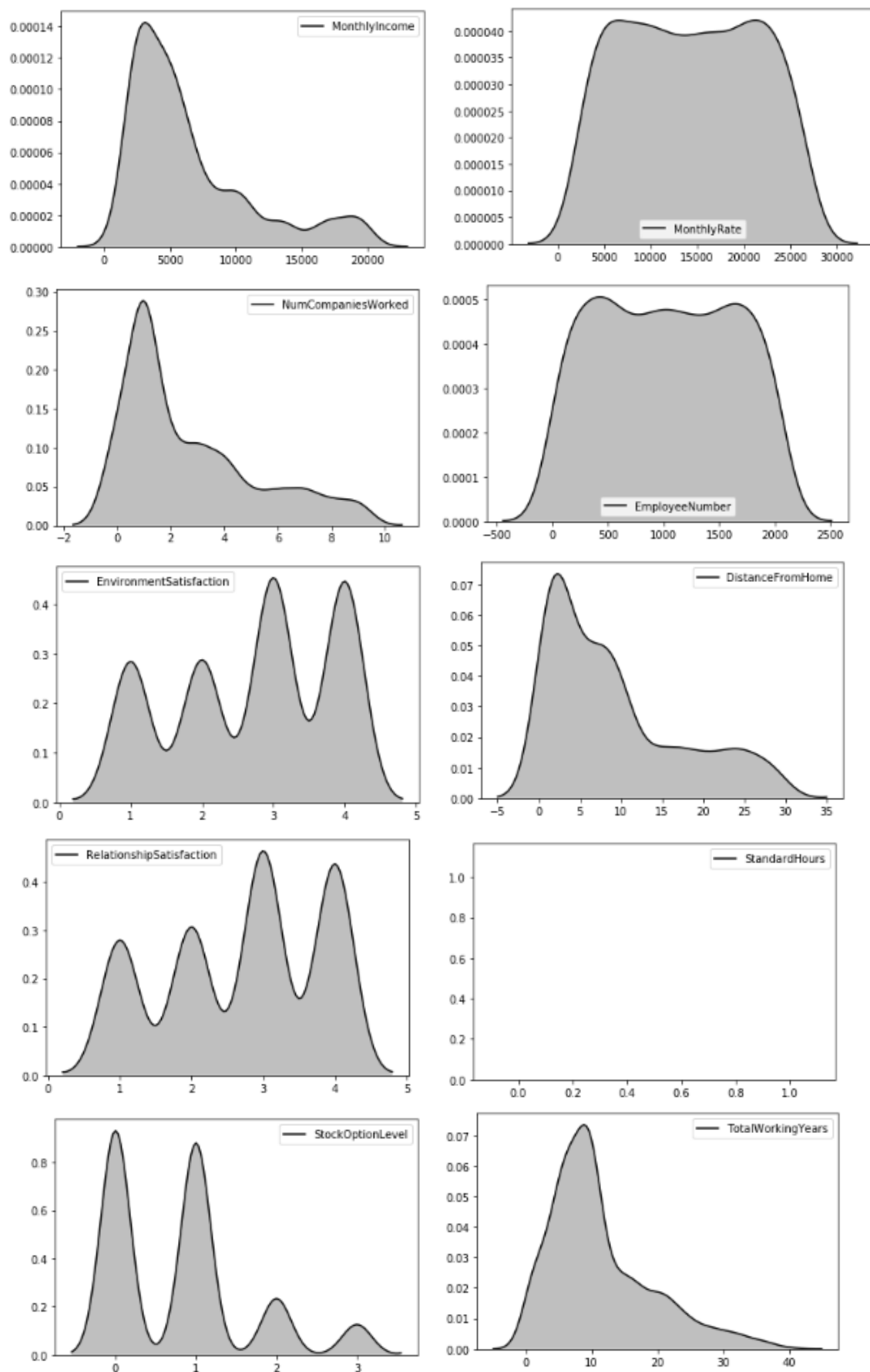
Listing 2.4 Graph for Numerical Data

```
sns.kdeplot(dataset[ 'Age ' ] ,shade="True" , color=" blue ")
```

Seaborn library is used for Data visualization. Seaborn is a library for making attractive and informative statistical graphics in Python. It is built on top of matplotlib and tightly integrated with the PyData stack, including support for numpy and pandas data structures and statistical routines from scipy and statsmodels. In this command , sns is alias of seaborn library , and kdeplot is used for numerical datas

virtualization. This command means that draw a graph using kdeplot method from seaborn library and values from Age column in dataset. Also this command has some arguments and parameters. `shade` is used for line (`shade="True"` means that the graph is covered by a line) and `color` is used for color of line which covered the graph.





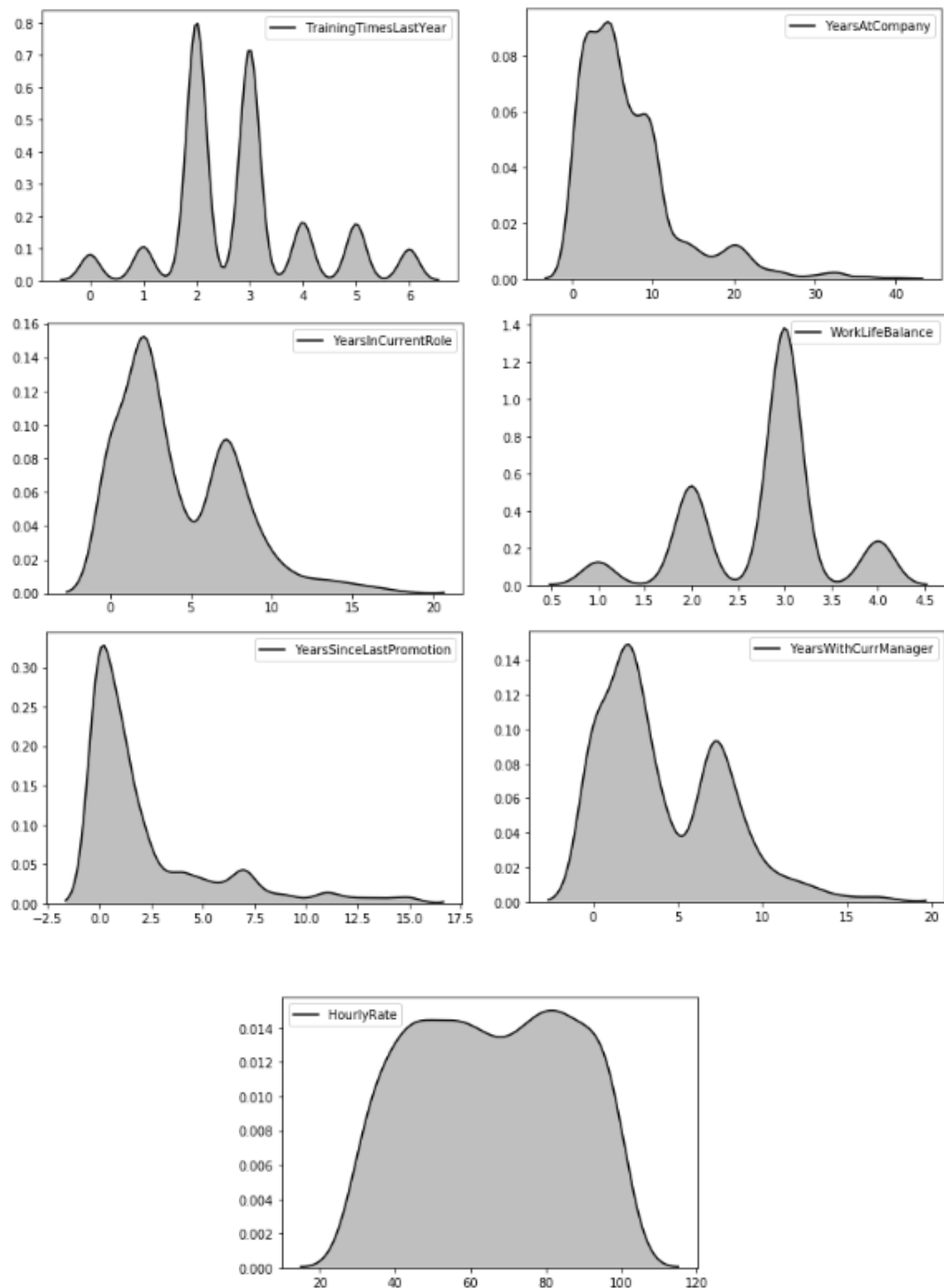


Figure 2.4 Output of Listing 2.4 for each Numerical Value

Listing 2.5 Selecting Pandas Columns by dtypes

```

num_dataset = dataset.select_dtypes(include=["number"])
num_dataset.dtypes
cat_dataset = dataset.select_dtypes(exclude=["number"])
cat_dataset.dtypes

```

This part, datas are separated by their data types. The command is `num_dataset = dataset.select_dtypes(include=["number"])` means that create a new variable (num_dataset) and it includes numerical datas. Then `cat_dataset = dataset.select_dtypes(exclude=["number"])` means that create a new variable (cat_dataset) and it includes categorical datas.

```

Age                                int64
DailyRate                          int64
DistanceFromHome                    int64
Education                           int64
EmployeeCount                       int64
EmployeeNumber                      int64
EnvironmentSatisfaction             int64
HourlyRate                          int64
JobInvolvement                      int64
JobLevel                            int64
JobSatisfaction                     int64
MonthlyIncome                       int64
MonthlyRate                         int64
NumCompaniesWorked                  int64
PercentSalaryHike                   int64
PerformanceRating                   int64
RelationshipSatisfaction             int64
StandardHours                       int64
StockOptionLevel                    int64
TotalWorkingYears                   int64
TrainingTimesLastYear               int64
WorkLifeBalance                     int64
YearsAtCompany                      int64
YearsInCurrentRole                  int64
YearsSinceLastPromotion              int64
YearsWithCurrManager                int64
dtype: object

Attrition                           object
BusinessTravel                       object
Department                           object
EducationField                       object
Gender                               object
JobRole                              object
MaritalStatus                        object
Over18                               object
OverTime                             object
dtype: object

```

Figure 2.5 Output of Listing 2.5

Listing 2.6 Outlier Analysis

```

Q1 = num_dataset.quantile(0.15)
Q3 = num_dataset.quantile(0.85)
IQR = Q3 - Q1
print(IQR)

```

Normally, 25% and 75% are used in IQR method but in this dataset , 691 rows were dropped when 25% and 75% are used. For this reason, in this command 15% and 85% are used. These commands are used for finding outlier values.

```

Age                                19.00
DailyRate                          978.65
DistanceFromHome                    18.00
Education                           2.00
EmployeeCount                        0.00
EmployeeNumber                      1447.30
EnvironmentSatisfaction              3.00
HourlyRate                          48.00
JobInvolvement                      1.00
JobLevel                            2.00
JobSatisfaction                      3.00
MonthlyIncome                       8451.10
MonthlyRate                         17190.45
NumCompaniesWorked                   5.00
PercentSalaryHike                    8.00
PerformanceRating                    1.00
RelationshipSatisfaction              3.00
StandardHours                        0.00
StockOptionLevel                     2.00
TotalWorkingYears                    16.00
TrainingTimesLastYear                2.00
WorkLifeBalance                      1.00
YearsAtCompany                       9.00
YearsInCurrentRole                   8.00
YearsSinceLastPromotion              5.00
YearsWithCurrManager                 8.00
dtype: float64

```

Figure 2.6 Output of Listing 2.6

Listing 2.7 Outlier Analysis

```

idx = ((num_dataset < (Q1 - 1.5 * IQR)) | (num_dataset >
(Q3 + 1.5 * IQR))).any(axis=1)
idx.value_counts()
dataset_cleaned = dataset[~((num_dataset < (Q1 - 1.5 * IQR)) | (num_dataset >
(Q3 + 1.5 * IQR))).any(axis=1)]
dataset_cleaned = dataset[~((num_dataset < (Q1 - 1.5 * IQR)) | (num_dataset >
(Q3 + 1.5 * IQR))).any(axis=1)]
dataset_cleaned.shape

```

This command is used to drop the values that fall outside of the specified range.

```

      Age  DailyRate  DistanceFromHome  Education  EmployeeCount  \
0      False      False      False      False      False      False
1      False      False      False      False      False      False
2      False      False      False      False      False      False
3      False      False      False      False      False      False
4      False      False      False      False      False      False
...      ...      ...      ...      ...      ...      ...
1465    False      False      False      False      False      False
1466    False      False      False      False      False      False
1467    False      False      False      False      False      False
1468    False      False      False      False      False      False
1469    False      False      False      False      False      False

      EmployeeNumber  EnvironmentSatisfaction  HourlyRate  JobInvolvement  \
0      False      False      False      False      False
1      False      False      False      False      False
2      False      False      False      False      False
3      False      False      False      False      False
4      False      False      False      False      False
...      ...      ...      ...      ...      ...
1465    False      False      False      False      False
1466    False      False      False      False      False
1467    False      False      False      False      False
1468    False      False      False      False      False
1469    False      False      False      False      False

      JobLevel  ...  RelationshipSatisfaction  StandardHours  \
0      False  ...      False      False
1      False  ...      False      False
2      False  ...      False      False
3      False  ...      False      False
4      False  ...      False      False
...      ...  ...      ...      ...
1465    False  ...      False      False
1466    False  ...      False      False
1467    False  ...      False      False
1468    False  ...      False      False
1469    False  ...      False      False

      StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
0      False      False      False      False
1      False      False      False      False
2      False      False      False      False
3      False      False      False      False
4      False      False      False      False
...      ...      ...      ...
1465    False      False      False      False
1466    False      False      False      False
1467    False      False      False      False
1468    False      False      False      False
1469    False      False      False      False

      WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
0      False      False      False      False
1      False      False      False      False
2      False      False      False      False
3      False      False      False      False
4      False      False      False      False
...      ...      ...      ...
1465    False      False      False      False
1466    False      False      False      False
1467    False      False      False      False
1468    False      False      False      False
1469    False      False      False      False

      YearsSinceLastPromotion  YearsWithCurrManager
0      False      False
1      False      False
2      False      False
3      False      False
4      False      False
...      ...      ...
1465    False      False
1466    False      False
1467    False      False
1468    False      False
1469    False      False

[1470 rows x 26 columns]
: False      1419
: True       51
: dtype: int64
: (1419, 35)

```

Figure 2.7 Outputs of Listing 2.7

Listing 2.8 Outlier Analysis

```
d = {'No': 0, 'Yes': 1}
dataset_cleaned['Attrition'] =
dataset_cleaned['Attrition'].map(d).fillna(dataset_cleaned['Attrition'])
dataset_cleaned['Attrition'].astype(int)
dataset_final = pd.get_dummies(dataset_cleaned, drop_first=True)
dataset_final.columns
```

This command is used for to change the YES and NO values of the Attrition feature to 1 and 0. Then this column datatype change to integer. After these processes final columns are created.

```
0      1
1      0
2      1
3      0
4      0
..
1465   0
1466   0
1467   0
1468   0
1469   0
Name: Attrition, Length: 1419, dtype: int32

Index(['Age', 'Attrition', 'DailyRate', 'DistanceFromHome', 'Education',
       'EmployeeCount', 'EmployeeNumber', 'EnvironmentSatisfaction',
       'HourlyRate', 'JobInvolvement', 'JobLevel', 'JobSatisfaction',
       'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'PercentSalaryHike', 'PerformanceRating', 'RelationshipSatisfaction',
       'StandardHours', 'StockOptionLevel', 'TotalWorkingYears',
       'TrainingTimesLastYear', 'WorkLifeBalance', 'YearsAtCompany',
       'YearsInCurrentRole', 'YearsSinceLastPromotion', 'YearsWithCurrManager',
       'BusinessTravel_Travel_Frequently', 'BusinessTravel_Travel_Rarely',
       'Department_Research & Development', 'Department_Sales',
       'EducationField_Life Sciences', 'EducationField_Marketing',
       'EducationField_Medical', 'EducationField_Other',
       'EducationField_Technical Degree', 'Gender_Male',
       'JobRole_Human Resources', 'JobRole_Laboratory Technician',
       'JobRole_Manager', 'JobRole_Manufacturing Director',
       'JobRole_Research Director', 'JobRole_Research Scientist',
       'JobRole_Sales Executive', 'JobRole_Sales Representative',
       'MaritalStatus_Married', 'MaritalStatus_Single', 'OverTime_Yes'],
      dtype='object')
```

Figure 2.8 Outputs of Listing 2.8

Listing 2.9 Outlier Analysis

```
y = dataset_final['Attrition']
X = dataset_final.drop(columns = ['Attrition'])
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=42)
```

This code means that Attrition is the class label. It creates train and test sets of sizes 75%, 25%, respectively.

2.3 Machine Learning Algorithms Implementation

2.3.1 Logistic Regression

Listing 2.10 Logistic Regression

```
logreg = LogisticRegression()  
logreg.fit(X_train, y_train)  
y_pred=logreg.predict(X_test)  
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)  
cnf_matrix
```

Logistic Regression code is applied to the dataset.

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,  
                    intercept_scaling=1, l1_ratio=None, max_iter=100,  
                    multi_class='warn', n_jobs=None, penalty='l2',  
                    random_state=None, solver='warn', tol=0.0001, verbose=0,  
                    warm_start=False)  
  
array([[292,  1],  
       [ 51, 11]], dtype=int64)
```

Figure 2.9 Outputs of Listing 2.10

Listing 2.11 Logistic Regression

```
class_names=[0,1]  
fig, ax = plt.subplots()  
tick_marks = np.arange(len(class_names))  
plt.xticks(tick_marks, class_names)  
plt.yticks(tick_marks, class_names)  
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu", fmt='g')  
ax.xaxis.set_label_position("top")  
plt.tight_layout() plt.title('Confusion_matrix', y=1.1)  
plt.ylabel('Actual_label')  
plt.xlabel('Predicted_label')
```

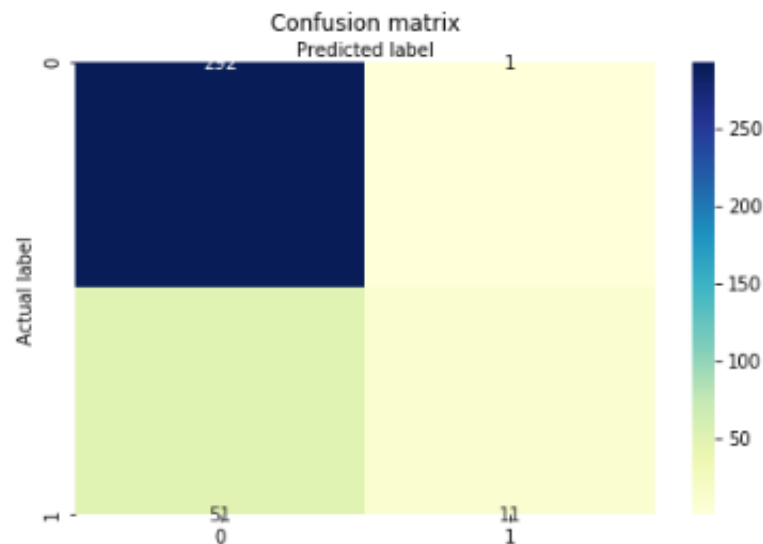


Figure 2.10 Outputs of Listing 2.11

Listing 2.12 Logistic Regression

```

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
print("Precision:", metrics.precision_score(y_test, y_pred))
print("Recall:", metrics.recall_score(y_test, y_pred))

y_pred_proba = logreg.predict_proba(X_test)[:,1]
logit_roc_auc = roc_auc_score(y_test, logreg.predict(X_test))
fpr, tpr, thresholds = roc_curve(y_test, logreg.predict_proba(X_test)[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic_Regression_(area_=%0.2f) '% logit_roc_auc)
plt.plot([0, 1], [0, 1], 'r—')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False_Positive_Rate')
plt.ylabel('True_Positive_Rate')
plt.title('Receiver_operating_characteristic')
plt.legend(loc="lower_right")
plt.savefig('Log_ROC')
plt.show()

```

This command is used for printing the Accuracy, Precision and Recall values also it creates the ROC graph.

```

Accuracy: 0.8535211267605634
Precision: 0.9166666666666666
Recall: 0.1774193548387097

```

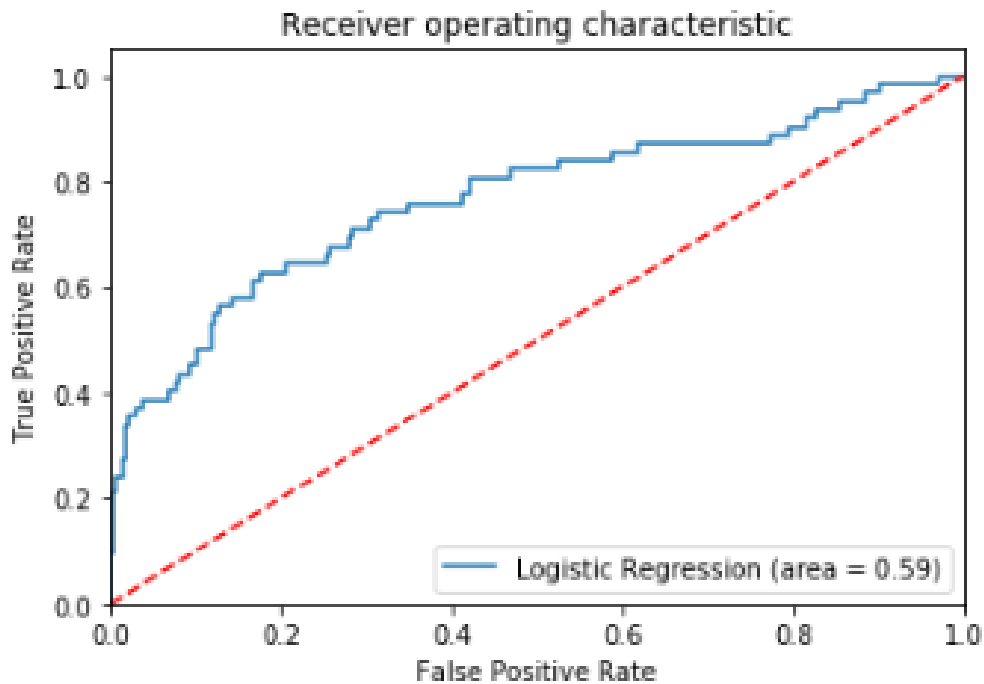



Figure 2.11 Outputs of Listing 2.12

2.3.2 Decision Tree(Gini)

Listing 2.13 Decision Tree (Gini)

```
max_depth_range = list(range(1, 10))

accuracy = []
for depth in max_depth_range:
    clf = DecisionTreeClassifier(max_depth = depth )
    clf = clf.fit(X_train , y_train)
    y_pred = clf.predict(X_test)
    print("Accuracy_of_max_depth" +str(depth) , metrics.accuracy_score(y_test , y_pred))
```

This command is used to compare depth of the decision trees with Gini criterion. It creates depth value from 1 to 10.

```
Accuracy of max_depth4 0.8169014084507042
Accuracy of max_depth5 0.8
Accuracy of max_depth6 0.8
Accuracy of max_depth1 0.8253521126760563
Accuracy of max_depth2 0.8366197183098592
Accuracy of max_depth3 0.8366197183098592
Accuracy of max_depth7 0.7915492957746478
Accuracy of max_depth8 0.7690140845070422
Accuracy of max_depth9 0.7661971830985915
```

Figure 2.12 Outputs of Listing 2.13

Listing 2.14 Decision Tree(Gini)

```
clf = DecisionTreeClassifier(max_depth=3)
clf = clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix

lass_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(lass_names))
plt.xticks(tick_marks, lass_names)
plt.yticks(tick_marks, lass_names)
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu", fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion_matrix', y=1.1)
plt.ylabel('Actual_label')
plt.xlabel('Predicted_label')
```

After selecting the depth of the tree ,the algorithm is implemented then the confusion matrix is constructed.

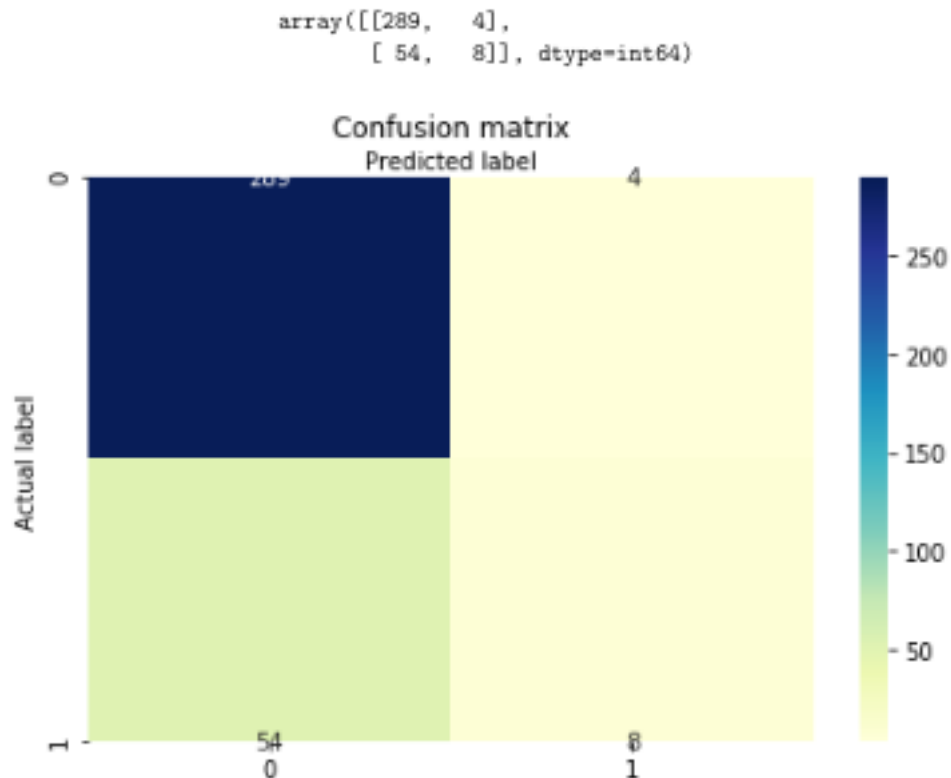


Figure 2.13 Outputs of Listing 2.14

Listing 2.15 Decision Tree (Gini)

```
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
print("Precision:", metrics.precision_score(y_test, y_pred))
print("Recall:", metrics.recall_score(y_test, y_pred))

dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
               filled=True, rounded=True,
               special_characters=True, feature_names = X.columns, class_names=['0', '1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('ibm_HR.png')
Image(graph.create_png())
```

This command prints Accuracy, Precision and Recall values of decision tree with Gini criterion. Then the tree is created.

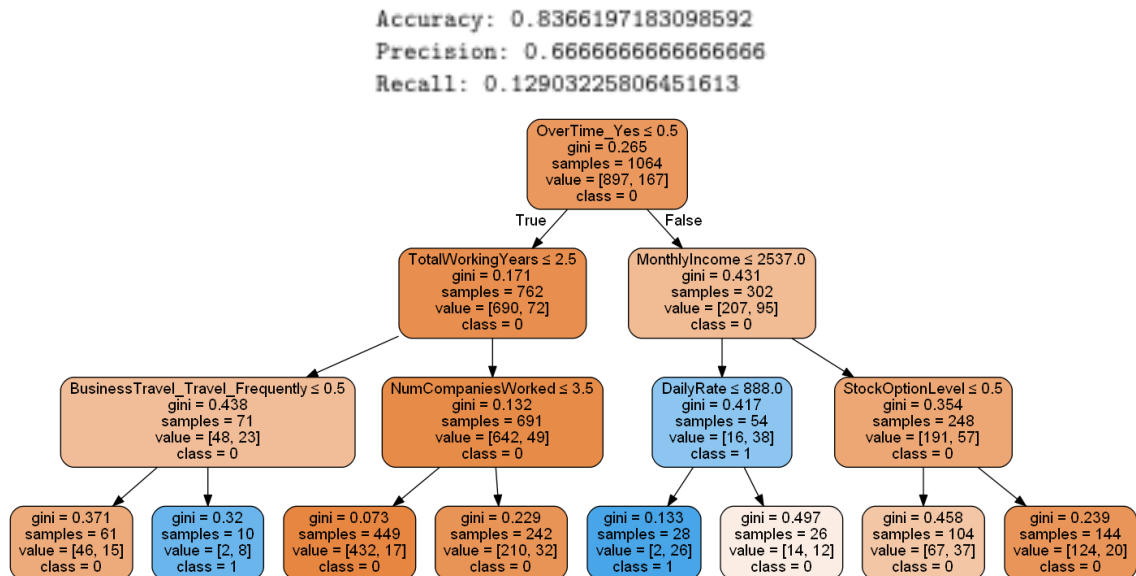


Figure 2.14 Outputs of Listing 2.15

2.3.3 Decision Tree(Entropy)

Listing 2.16 Decision Tree(Entropy)

```
max_depth_range = list(range(1, 10))
accuracy = []
for depth in max_depth_range:
    clf = DecisionTreeClassifier(max_depth = depth, criterion='entropy')
    clf = clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    print("Accuracy_of_max_depth" +str(depth) , metrics.accuracy_score(y_test, y_pred))
    clf = DecisionTreeClassifier(max_depth=3, criterion='entropy')
    clf = clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
    cnf_matrix

class_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu", fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion_matrix', y=1.1)
plt.ylabel('Actual_label')
plt.xlabel('Predicted_label')
```

This command is used to compare depth of the decision trees with Entropy criterion. It creates depth value from 1 to 10. After selecting the depth of the tree, the algorithm is implemented then the confusion matrix is constructed.

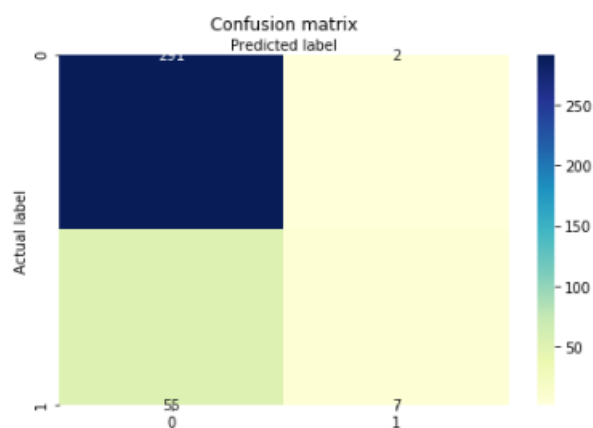


Figure 2.15 Outputs of Listing 2.16

Listing 2.17 Decision Tree(Entropy)

```
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
print("Precision:", metrics.precision_score(y_test, y_pred))
print("Recall:", metrics.recall_score(y_test, y_pred))

dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
filled=True, rounded=True,
special_characters=True, feature_names = X.columns, class_names=['0', '1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('ibm_HR.png')
Image(graph.create_png())
```

This command prints Accuracy, Precision and Recall values of decision tree with Entropy criterion. Then the tree is created.

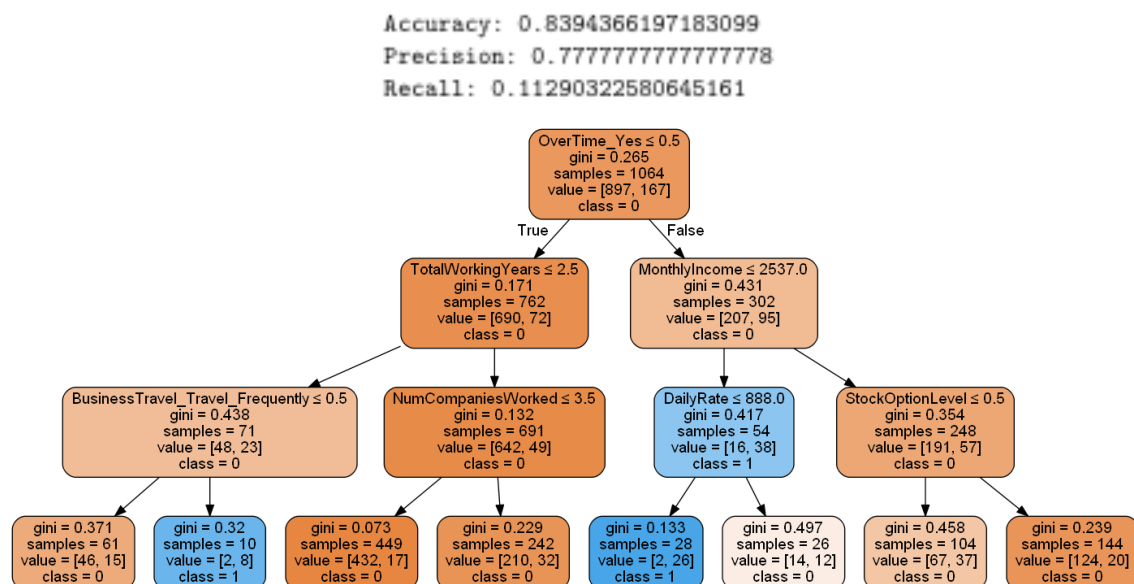


Figure 2.16 Outputs of Listing 2.17

2.3.4 Random Forest

Listing 2.18 Random Forest

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(random_state = 42)
rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)
y_train_pred = rfc.predict(X_train)
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matr
#Confusion Matrix
```

```

class_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu", fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')

#Accuracy, Precision and Recall

print("Accuracy:", metrics.accuracy_score(y_test, _y_pred))
print("Precision:", metrics.precision_score(y_test, _y_pred))
print("Recall:", metrics.recall_score(y_test, _y_pred))

```

This command is used for implementing the Random Forest algorithm then constructing the confusion matrix. Then it prints Accuracy, Precision and Recall values.

```

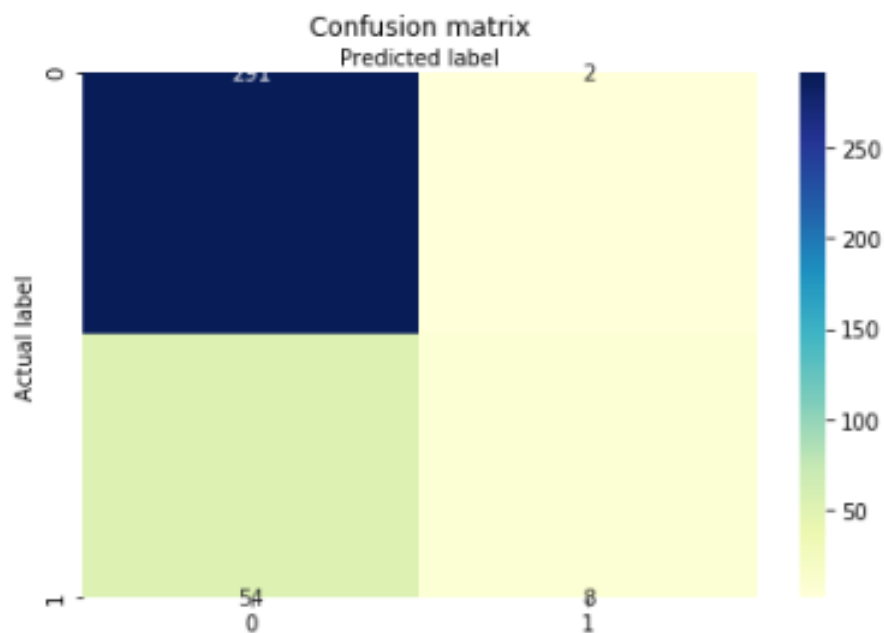
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                        max_depth=None, max_features='auto', max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, n_estimators=10,
                        n_jobs=None, oob_score=False, random_state=42, verbose=0,
                        warm_start=False)

```

```

array([[291,  2],
       [ 54,  8]], dtype=int64)

```



```
Accuracy: 0.8535211267605634  
Precision: 0.9166666666666666  
Recall: 0.1774193548387097
```

Figure 2.17 Outputs of Listing 2.18

3

Conclusion

After implementing Logistic Regression, Decision Tree and Random Forest algorithms it can be seen that Logistic Regression is the best algorithm based on the accuracy value for this dataset. It gives a considerably high accuracy.

Model	Accuracy	Precision
Logistic Regression	0.8816901408450705	0.9166666666666666
Decision Tree (Gini)	0.8366197183098592	0.6666666666666666
Decision Tree (Entropy)	0.8394366197183099	0.7777777777777778
Random Forest	0.8422535211267606	0.8

Figure 3.1 The Excel Table to Compare 4 Models

References

- [1] O. Theobald, "Machine learning for absolute beginners ,2nd edition," 2018.
- [2] Interactions. (2017). A History of Machine Learning infographic, [Online]. Available: https://www.interactions.com/wp-content/uploads/2018/10/brief_history_of_ML.jpg.
- [3] M. L. in Netherlands. (2018). History of machine learning, [Online]. Available: <https://mlplatform.nl/what-is-machine-learning/> (visited on 11/29/2019).
- [4] O. Theobald, "Machine learning for absolute beginners ,2nd edition," 2018.
- [5] S. Etlä. (2017). Data Science Central decoding machine learning methods, [Online]. Available: <http://storage.ning.com/topology/rest/1.0/file/get/2808358673?profile=original> (visited on 11/29/2019).
- [6] R. Ng. (2019). Logistic regression, [Online]. Available: <https://www.ritchieng.com/logistic-regression/> (visited on 12/02/2019).
- [7] D. Sarkar and R. e. a. Bali, "Practical machine learning with python," 2018.
- [8] Google. (2019). Comparegoogle trends, [Online]. Available: <https://trends.google.com/trends/explore?date=all&q=Python%20Machine%20Learning,R%20Machine%20Learning,Java%20Machine%20Learning,Scala%20Machine%20Learning,Julia%20Machine%20Learning> (visited on 12/14/2019).
- [9] S. RAY. (2017). Analyticsvidhya, [Online]. Available: <https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/> (visited on 12/16/2019).
- [10] B. Capricorn. (2017). Pickhost.net, [Online]. Available: <https://pichost.net/img/data-scrubbing-services-b2b-capricorn.6vZZ> (visited on 12/16/2019).

Curriculum Vitae

Name-Surname: Sümeyye Seren

Birthdate and Place of Birth: 24.01.1997

Birth Place: Bahçelievler/İstanbul

High School: Dr.Kemal Naci Ekşi Anatolian High School

Internship: Zingat ,İstanbul (6 months)