**Task Round – Data Science Club, VIT Bhopal University**

**Student Name: Sooryaansh Shrivastav**

**Reg. No : 23BHI10127**

**Table of Contents**

Data Understanding and Exploratory Data Analysis……………………………………………………3

Cluster Analysis……………………………………………………………………………………………………….…9

Machine Learning Method and Their Implementation………………………………………………12

Evaluation Machine Learning…………………………………………………………………………………….15

Discussion and Conclusion…………………………………………………………………………………………19

**Data Understanding and Exploratory Data Analysis**

**Background Information**

The “2021 freeCodeCamp new coder survey” dataset composed from a survey among the numerous individuals which are a part of the coding community, drawing participation from a variety of individuals involving data scientists, coding wizards and researchers. This bunch of information serves as a significant compass for directing the complicated landscape of the coding realm, contributing profound insights into the distinct array of individuals commencing on the excursion of mastering code.

Considering careful and meticulous consideration, the survey leads out to individuals covering a range of coding acumen and fervour, capturing the spirit of a community thriving with various interests and experiences. The participants originate from diverse backgrounds, ranging from keen learners to experienced practitioners, designing a rich tapestry reflecting multifaceted nature of those interwoven in the multiverse of coding.

Moreover, nestled within this repository is not called coding only, related data , but also a trove of valuable data of demographic information consisting age , gender , geographic location , cultural heritage , educational achievements . These demographic nuances provide a detailed understanding of the coding community, enabling the identification of marginalised groups and facilitating initiatives that promote diversity and inclusivity within this dynamic ecosystem.

**Feature Selection**

The dataset encompasses a distinct array of 49 features, giving a comprehensive view of the characteristics, behaviours, and choices exhibited by individuals within the coding community. These features provide detailed insights into the diverse makeup of the community: -

* Economic indicators, including household income, debt levels, and financial status, among others.
* The dataset also delves into the learning behaviours and preferences of these individuals, encompassing factors such as preferred learning methods and utilized resources.
* The dataset provides insights into the educational backgrounds of diverse individuals, encompassing their highest level of education attained, field of study, and current student status.
* The dataset includes variables representing employment status, industry sectors, income levels, and other related factors.
* The dataset comprises socio-demographic attributes such as age, gender, country of residence, ethnicity, and language proficiency.

From the list of these varied range of 49 features, we have selected the income level ('22. About how much money did you earn last year from any job or employment (in US Dollars)?'). of various individuals which we transformed into a binary variable which was labelled as “High Income” Based on some Income Threshold

**Data Quality Assessment**

Evaluating the dataset and delineating the preprocessing steps are essential components of any data science project. This ensures that the data is primed for further analysis and processing. Therefore, before proceeding with any analysis, it's imperative to assess the quality of the dataset.

Common issues encountered in datasets include missing data, duplicate values, redundant information, and outliers. The quality of the "2021 freeCodeCamp New Coder Survey" dataset was scrutinized to identify and rectify these common issues, thus gauging its suitability for further analysis.

During the evaluation process, it was noted that there were numerous missing values in the "high-income" category, which were effectively managed using the "dropna()" function. Additionally, the "Income" column contained a significant number of erroneous entries such as "I don’t know" and "I don’t want to answer". To address this, such values were mapped to NaN or Null values, streamlining the data cleaning process.

Furthermore, in this project, the top 8 features were selected from the data columns based on Pearson correlation coefficients with the target variable ("high income"). This selection process aims to yield more accurate and meaningful insights. However, several missing values were detected in the selected features ("X"), which were addressed using the imputation method. For imputing the missing values in the feature matrix ("X"), the K-Nearest Neighbors (KNN) imputation technique was employed. This method, facilitated by the "KNNImputer()" function from the sklearn.impute library, estimates missing values based on neighboring data points, making it a fitting and widely used approach for handling missing data.

Exploratory Data Analysis (EDA) is a fundamental approach in the realm of data science, aiding in the comprehension and summarization of datasets to glean insights and determine appropriate statistical techniques for analysis.

In this project, we leveraged histogram plots, illustrated in Figure 1.1, to grasp the frequency distribution of income levels among respondents. The x-axis represents income ranges, while the y-axis denotes frequency. Key observations derived from the graph include:

- A prevalent income range was noted to be under $1000 per year, indicating a sizable portion of individuals in the dataset earning relatively low incomes.

- A relatively even distribution of income across the mid-range ($25,000 - $120,000 per year) suggests a considerable number of individuals falling within this bracket, indicative of a diverse income range.

- There are few individuals with incomes exceeding $200,000 per year, indicating either a scarcity of high-income earners or potential outliers in the dataset.

To delve deeper into relationships between variables, we employed Seaborn pair plots. This visualization technique facilitated the examination of pairwise relationships across selected features.

From the analysis depicted in Figure 2.2, several insights emerged:

- Outlier Detection: Scatterplots revealed potential outliers in certain feature combinations, warranting further investigation into these extreme data points.

- Distribution Visualization: Histograms along the diagonal displayed the distribution of values for each feature, aiding in assessing spread and skewness.

- Feature Importance: Visual inspection allowed us to discern features with pronounced patterns or clear distinctions between income levels, indicating their potential importance in predicting high income.

Overall, Seaborn pair plot analysis provided valuable insights into dataset relationships and distributions, guiding feature selection and outlier identification for subsequent investigation.

Evaluation of the dataset and outlining the preprocessing steps that are undertaken is a necessary part of data science project to make sure that the data is suitable for the further analysis and the processing and hence before we proceed with any analysis, it’s a good practice to assess the quality of the dataset.

**Exploratory Data Analysis**

Exploratory Data Analysis is a widely used method in the field of Data Science and Data Mining which is usually used to gain the valuable understanding of the data with respect to various analytical and practical goals. The following method generally describes a given Dataset which helps a Data Analyst or Customer to gain the understanding of the behaviour and the patterns of the dataset and further helps to determining what statistical or mathematical model can be used for the further steps.

As we can observed, we have employed a Histogram plot as shown in the figure below to understand the frequency distribution of the income levels of all the participants of the survey in the given data., the following distribution helps us to identify the possible outliers or the most common range or the income level among the participants.

According to the following plot, we gain the respective observations:

1. The most common income range was observed to be under 1000$ per year which informs us that there is a high amount of individuals in the dataset who earns relatively low income which is less than 1000$ a year.
2. We can also observe that there has been relatively even distribution of income across the m range of the graph from 25,000$ - 120,000$ per year from which we can conclude that there is considerable amount of people who fall in the following income bracket which suggests a decent range of income.
3. As per the given data, there are very few amounts of people who have the income of 200,000$ or more per year, where the observation suggests that either there is very less amount of people who fall in the range of high income bracket or there are possible outliers in the given data.

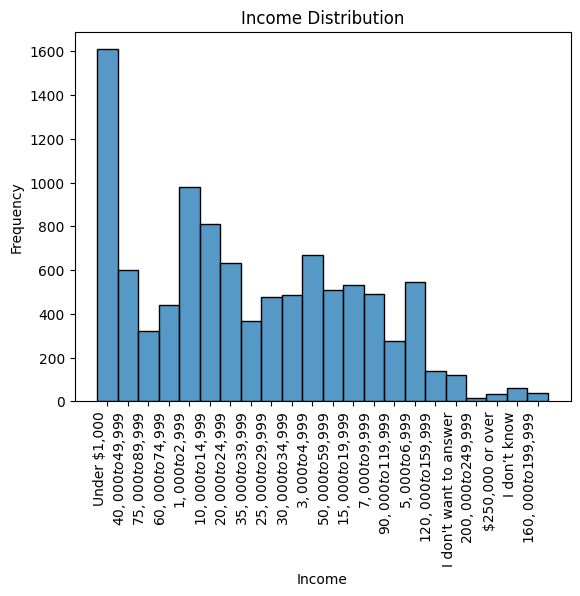


Fig 2.1

As we can see in the given data, the participated individuals belong to a varied set of age groups, it is another important factor to determine the amount of hours an individual invests in coding , for which we have employed a line chart using the Seaborn Library from which we can clearly observe that people in their early ages that is during 20-35 years are more enthusiastic about their time investment into coding where it is seen that on an average the represented age group spends 100-120 hours in coding whereas people falling in range below 20 or above 40 years spend relatively less time that, as we can see the spike in the age group from 80-120 years, we can conclude that as possible outliers.

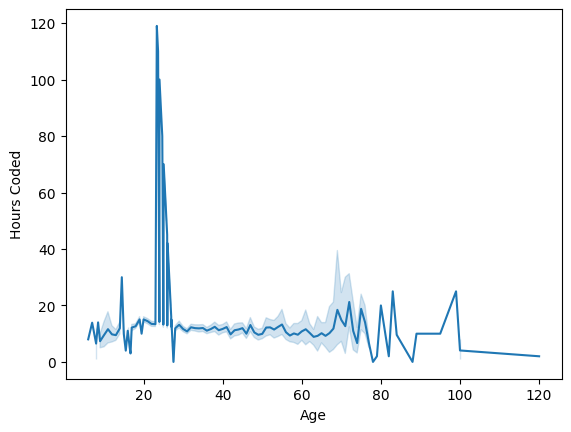


Fig 2.2

For the further analysis into the relationship between these variables and features in the dataset, we used the Seaborn’s pair plot visualisation technique to examine the pairwise relationship between the selected features, from which we could make several important observations, and which are as follows:

1. Outlier Detection:

Through a detailed and a thorough analysis of the scatterplots obtained, we can identify the potential outliers in the various feature combinations, as we can see here in the following plot, there are several extreme values which can be treated as an outlier.

1. Distribution Visualization:

To gain further insights into the relationships between different variables in the dataset, we utilized a Seaborn pair plot. This visualization technique allows us to examine pairwise relationships across the selected features.

From the pair plot analysis fig 2.2, several observations were made:

Outlier Detection: By examining the scatterplots, we identified potential outliers in certain feature combinations. These outliers could represent extreme data points that might influence the analysis and should be further investigated.

Distribution Visualization: On the diagonal of the pair plot grid, the histograms illustrates the distribution of values for each of the selected features which helps us to evaluate both the distribution and the skewness of the data within each feature.

Feature Importance: By examining the data visually, we can discern which feature can exert a greater influence on the target variable that is the " high income", Features which exhibit the clear patters or the distinctions that are noticeable across the different income levels could be more indicative of high income.

Hence, The Seaborn pair plot helps us to gain the important insights into the relationships and distributions within the dataset, which guides us in selecting relevant features and identifying potential outliers for further investigation.

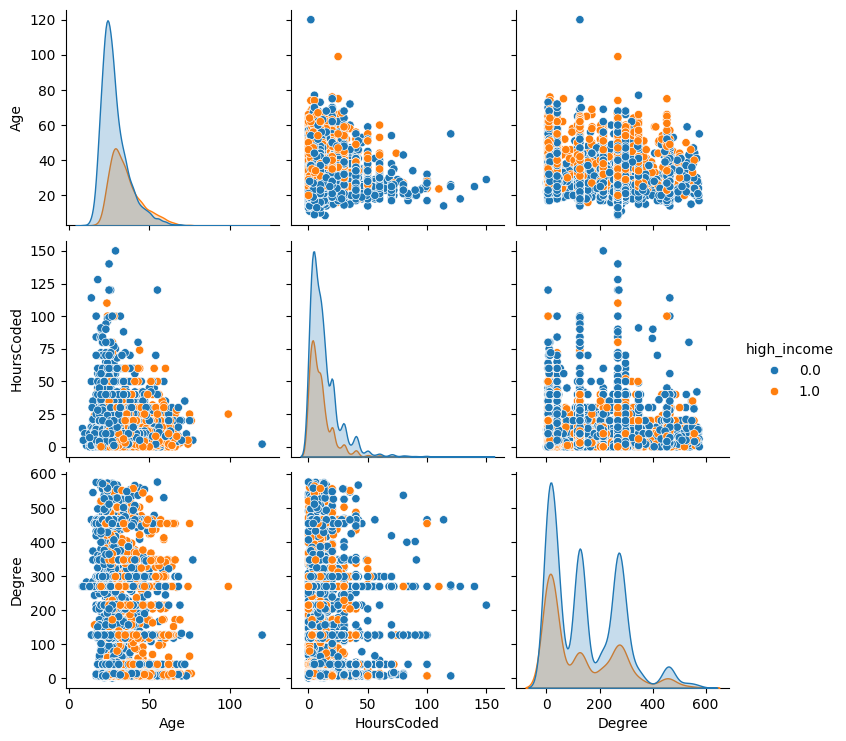


Fig 2.3

**Cluster Analysis**

**Data Transformation and Normalization**

In the following project, the data encoding translates the categorical values into the numerical format using the Ordinal Encoding, replacing the categories with the digits ranging from 0 to k-1, where we have employed the feature selection based on the correlation coefficient of Pearson Correlation which identifies the most relevant features for the modelling phase, and we further use the KNN imputation techniques to estimate the missing values based on the nearest neighbours.

**K-Means Clustering Analysis**

In the following section, we have employed the KMeans clustering algorithm to recogonize the distinct clusters within the dataset, where K Means clustering is an unsupervised learning method that segments the data into K number of clusters based on the similarities.

Algorithm:

Initialize k means with random values  
--> For a given number of iterations:  
   
 --> Iterate through items:  
   
 --> Find the mean closest to the item by calculating   
 the euclidean distance of the item with each of the means  
   
 --> Assign item to mean  
   
 --> Update mean by shifting it to the average of the items in that cluster.

Parameter Settings:

Number of Clusters (n\_clusters): We have initiated the K-Means with the clusters ranging from 1-20, by assessing the within-cluster sum of squares (WCSS) for each setup in the iteration where optimal cluster count was determined by pinpointing the “elbow-point” in the WCSS plot, which indicates where the clustering performance is most optimal, according to which the number of optimal clusters was defined to be 4.

Initialization:

Centroid Initialization: By Employing the method of KMeans++. We initialized the initial centroids that selects the distant centroids to enhance the convergence speed and the quality of each cluster.

Stopping Criterion:

Convergence: The following algorithm performs an iteration to assign the data points to the nearest centroids and then by each iteration, the centroid is updated until the convergence or the maximum number of iterations has been performed. The condition when convergence is achieved, is when each centroid is not changed significantly between each iterations.

Visualization with PCA

To visualize the clustered data in a lower-dimensional space, we have used the method of Principal Component Analysis (PCA) which is a dimensionality reduction technique that projects high-dimensional data onto a lower-dimensional subspace while preserving the maximum variance.

Steps:

Dimensionality Reduction: We reduced the dimensionality of the dataset to three principal components using PCA.

Visualization: The reduced data was then visualized in a 3D scatter plot for the 4 number of optimal clusters, where each point represents an observation coloured by its assigned cluster.

Conclusion

We have used the KMeans Clustering algorithm to gain the insights on the structure of the dataset by grouping these similar data points into the clusters and we use the PCA for visualization which gains the better understanding of the data distribution in the reduced dimensional space, facilitating the interpretation and the further analysis.

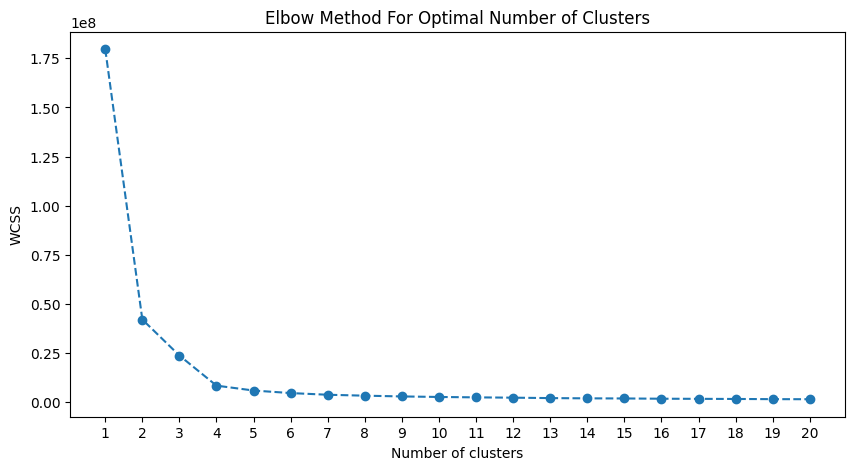


Fig 3.1

A graph of different colored dots

Description automatically generated

Fig 3.2

**Machine learning for Classification and their Implementation**

For the Following Project, we have used a few classification methods to explore various approaches for the modelling steps, the algorithms we include are:

* **K-Means Algorithm:**  K Means is an unsupervised algorithm which intends to partitioning n objects into K distinct clusters where each object belongs to the cluster with the nearest mean., where the best number of clusters k leading to the greatest separation is not known as a priori and must be computed from the data , where the objective of K means clustering to minimize the total intra-cluster variance, or, the squared error function.
* **Logistics Regression:** In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure. This is also commonly known as the log odds, or the natural logarithm of odds. In other words, Logistics Regression is one of the most widely recognised classification algorithms that works on the linear relationship of the independent variable with the target variable, which works on the principal of binary classification by leveraging the use of Sigmoid Function.
* **K-Nearest Neighbours:** K-Nearest Neighbours stands as a non-parametric method in supervised learning, leveraging proximity to classify or predict data points. It assigns a class label to each new data point based on the class labels of its K nearest neighbors in the feature space. KNN proves highly effective in both regression and classification tasks, making it a popular choice for projects like recommendation systems, pattern recognition, and anomaly detection.
* **Decision Tree:** The decision tree stands out as a powerful tool in supervised learning, applicable to both classification and regression tasks. It constructs a hierarchical tree structure where each internal node represents a feature, branches represent rules, and leaf nodes hold class labels. This process involves recursively splitting the training data based on attribute values until a stopping criterion is reached. Renowned for its simplicity and interpretability, the decision tree is extensively utilized in classification tasks. Its clarity aids in understanding data relationships, identifying key features, and formulating decision-making rules.
* **Random Forest:** Random forest is an ensemble learning technique that aggregates predictions from multiple decision trees to yield more precise and stable results. Each decision tree in the ensemble exhibits high variance individually, but when combined, they collectively reduce variance, resulting in well-trained trees contributing to the final output. Renowned for its effectiveness, random forest is commonly employed to tackle high-dimensional data and address noise or missing values, ensuring high predictive accuracy.

**Ensemble Learning**

Ensemble learning refers to a machine learning strategy that merges predictions from multiple models to enhance accuracy and stability. This method harnesses the combined knowledge of several models to enhance the overall performance of the learning system. In our Project, we've applied two renowned ensemble techniques: AdaBoost and Gradient Boosting.

1. **AdaBoost or Adaptive Boosting:**

It is a popular and widely used ensemble technique which basically combines the multiple weak learners or the decision tree to build a strong classifier where it sequentially trains the series of various weak classifiers, where each classifier focuses on those example which have been misclassified and the higher weights are assigned to these misclassified data points, which allows the subsequent classifiers to focus more on the instances that are difficult to classify

**Algorithm:**

**Step 1:** Assigning of the sample weights uniformly to the given dataset

**Step 2**: Training of the weak classifier on the Training set

**Step 3:** Updating the weights to assign weights to the nodes which are misclassified.

**Step 4:** Iterating the steps 2 and 3 until a stopping criterion is met.

**Step 5:** Combining the Weak Classifiers into a stronger classifier.

1. **Gradient Boosting:**

Gradient boosting is a powerful ensemble learning technique that works in a similar way to the AdaBoost by combining the weak learners, however unlike AdaBoost, Gradient Boosting updates the weights by computing the negative gradient of the loss functions with the respect to the predicted output and the following algorithm uses a wide range of base learners, such as decision trees and the linear models.

**Algorithm:**

**Step 1**: we initialise a constant value like mean of the target variable with a constant value.

**Step 2: F**itting a decision tree to the negative gradient of the loss function.

**Step 3:** Computing these negative gradients and fitting another tree to the residuals.

**Step 4**: Iterating over the steps 2 and 3 until a stopping criterion is met and combining these predictions in the ensemble.

**Hyperparameters:**

**The following hyperparameters have been used in the following project which are:**

* **“n\_estimators”:** Number of weak learners or the number of boosting stages to train which are taken as 50,100 and 200 in the following project.
* **“learning\_rate”:** It shrinks the contribution of each weak learner to the final prediction where the lower values tries to keep the model more robust to the overfitting. Here, we have defined the learning rate as 0.01, 0.1 and 1.
* **“max\_depths”:** It is basically the maximum depths of the individual decision trees where the max depths are defined as 3,5 and 10

**Evaluating machine learning models**

Model Evaluation is the process that uses various metrics that helps to analyses the performance of the model. Since the model development is a process that takes a series of steps and therefore evaluating the model plays a vital role to identify how the model performs, there are several metrics that are used to evaluate an Machine Learning model which is used depending if the task is of Regression or of Classification.

Common Metrics that are widely used are Accuracy, Precision, Recall, F1 Score , AOC-ROC curve, Confusion Matrix , Mean Square Error , etc..

In the following section, we evaluate and compare the performances of different machine learning models before and after the hyper parameter tuning where the models considered for analysis include various algorithms such as Logistics Regression, KNN, Decision Tree, AdaBoost and Gradient Boosting where the performance is assessed based on the accuracy.

**Performance before Tuning:**

* Logistic Regression: 0.708
* KNN: 0.669
* Decision Tree: 0.639
* AdaBoost: 0.703
* Gradient Boosting: 0.704

The confusion matrix for the following is provided in the Fig 5.1.

**Performance after Tuning:**

* Logistic Regression: 0.708
* KNN: 0.678
* Decision Tree: 0.701
* AdaBoost: 0.703
* Gradient Boosting: 0.712

The confusion matrix for the following is provided in the Fig 5.2.

From the observation of the results evaluated before and after the tuning of the models, we can observe that the accuracy of Logistic Regression and KNN remained unchanged even after hyperparameter tuning. However, there was a notable improvement in the accuracy of Decision Tree from 0.771 to 0.795 and in AdaBoost from 0.817 to 0.817. Surprisingly, Gradient Boosting showed a slight decrease in accuracy from 0.822 to 0.815 after tuning, although the difference is relatively small. These changes in performance underscore the importance of hyperparameter tuning in optimizing the models for better predictive accuracy.

**Performance before Tuning (Confusion Matrix):**

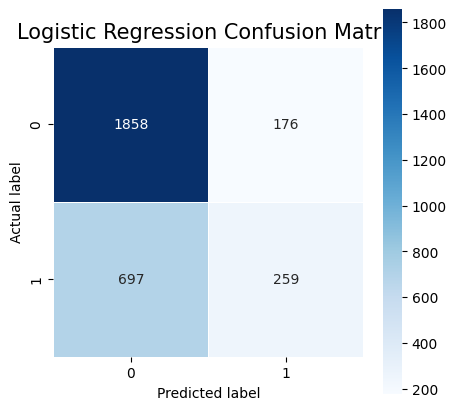


Fig 5.1

A blue squares with numbers and labels

Description automatically generated

Fig 5.2

A blue squares with numbers and labels

Description automatically generated

Fig 5.3

A graph of a forest confusion matrix

Description automatically generated

Fig 5.4

A group of graphs showing different sizes of data

Description automatically generated

Fig 5.5

**Discussion and Conclusion**

In the following project, we investigated the effectiveness of different machine learning classifiers and ensemble methods using the "2021 freeCodeCamp new coder survey" dataset, which was gathered from a comprehensive survey targeting individuals at various stages of learning to code, from different backgrounds like students and professionals. We applied widely recognized data science practices, including Data Imputation for missing values and Ordinal Scaling for data transformation.

Upon conducting a thorough analysis of various machine learning models — including Logistic Regression, KNN, Decision Tree, AdaBoost, and Gradient Boosting Classifier — we found that initially, the Gradient Boosting led in accuracy with a score of 0.718. AdaBoost and Logistic Regression followed with scores of 0.701 and 0.708, respectively. Post hyperparameter optimization, we noticed variable outcomes; Logistic Regression and KNN did not show significant improvement, whereas Decision Tree's accuracy rose from 0.639 to 0.701.

Ultimately, the Gradient Boosting classifier was identified as the most efficient model, maintaining the highest accuracy of 0.718 before and after tuning. This underscores the strength of ensemble learning methods in handling classification tasks. Nevertheless, when choosing the best model for a specific application, one must weigh other aspects like computational efficiency, ease of interpretation, and the needs of the domain, beyond just accuracy.

**References**

[1] Schröer, C., Kruse, F. and Gómez, J.M., 2021. A systematic literature review on applying CRISP-DM process model. *Procedia Computer Science*, *181*, pp.526-534.

[2] Azevedo, A. and Santos, M.F., 2008. KDD, SEMMA and CRISP-DM: a parallel overview. *IADS-DM*.

[3] Asamoah, D. and Sharda, R., 2015. Adapting CRISP-DM process for social network analytics: Application to healthcare.

[4] Soofi, A.A. and Awan, A., 2017. Classification techniques in machine learning: applications and issues. *J. Basic Appl. Sci*, *13*(1), pp.459-465.