Digital Object Identifier

ICS-5110 Applied Machine Learning

JEROME AGIUS[1], ISAAC MUSCAT[2], and KYLE DEMICOLI[3]
[1]jerome.agius.21@um.edu.mt
[2]isaac.muscat.21@um.edu.mt
[3]kyle.demicoli.21@um.edu.mt

I. INTRODUCTION

Throughout this project, two datasets were used for model training and evaluation. The first dataset used was the UCI-Adult [1] dataset. This dataset released in 1996 is a widely utilised classification dataset primarily used to predict whether the annual income of an individual exceeds $50K/yr based on census data. This dataset consists of 32561 rows composed of the fifteen features outlined in Table 1. The other dataset used is the Healthcare Insurance [2] dataset retrieved from Kaggle. This dataset explores the relationship between personal attributes, geographic factors, and their impact on medical insurance charges. This dataset consists of 1338 rows comprised of seven features as outlined in Table 2.

Analysis was carried out using exploratory data analysis (EDA) to understand the distribution and relationships among the features, identify missing values, and evaluate the impact of each feature on the target variable. Additionally, the data was pre-processed to address any inconsistencies, which include handling missing values, encoding categorical variables, and normalising numerical features to ensure they are suitable for modelling. Finally, the dataset was split into training and testing subsets to facilitate effective model evaluation.

To facilitate the classification task, we implemented three machine learning models: Support Vector Machines (SVM), Logistic Regression, and Random Forests. These were chosen due to their widespread popularity and applicability with regards to classification problems.

II. BACKGROUND

A. MACHINE LEARNING TECHNIQUES

1) Support Vector Machines (SVM)

A Support Vector machine as presented in [3], is a supervised machine learning algorithm usable for both classification and regression. It operates by finding the optimal line or hyperplane that maximises the margin between the closest data points of distinct classes, thereby defining a clear separation.

The concept of a "line" or "hyperplane" depends on the number of features present. In a two-dimensional space, the decision boundary is a line, while in a higher-dimensional space, it becomes a hyperplane. Although multiple lines or hyperplanes may potentially separate the classes, the algorithm selects the one with the largest margin. This choice ensures the best decision boundary by maximising the separation between classes. This can be seen in Figure 1.

The closest data points from each class that define the margin are known as support vectors. The margin itself is the perpendicular distance between these support vectors. The algorithm performs best with data that is linearly separable, but real-world datasets often contain some overlap or noise. In such cases, a soft-margin SVM is employed. This approach allows for a small number of misclassifications, providing the algorithm with greater flexibility in handling imperfect data.

The 'kernel-trick' as visualised in Figure 2, is a technique used for non-linear classification problems, which involves mapping non-linearly separable data into higher-dimensional space, where linear separation is possible. A variety of kernels can be used to achieve this task the choice depending on the data characteristics and the specific use case. Some of the most commonly used kernels include Polynomial, Sigmoid and RBF kernels [4].

SVMs perform classification through the optimisation of 1, where $\|w\|^2$ represents the squared norm of the weight vector w which is inversely proportional to the margin between the decision boundary and the closest point in each class. x_i denotes the input feature vector whilst y_i denotes the classification between {-1, 1} and b is the bias term that shifts the decision boundary.

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_i(w \cdot x_i + b) \geq 1, \quad \forall i \qquad (1)$$

Feature	Data Type	Description
age	Continuous (17 - 90)	Age in years
workclass	Categorical (9)	Type of employment
fnlwgt	Continuous (12285 - 1484705)	Final weight denoting statistical representation
education	Categorical (16)	Highest level of education
education-num	Continuous (1-16)	Numerical representation of education feature
marital-status	Categorical (7)	Marital status
occupation	Categorical (15)	Job type
relationship	Categorical (6)	Relationship in the household
race	Categorical (5)	Racial background
sex	Categorical (2)	Gender
capital-gain	Continuous (0-99999)	Capital gains earned
capital-loss	Continuous (0-4356)	Capital losses incurred
hours-per-week	Continuous (1-99)	Number of hours worked weekly
native-country	Categorical (42)	Country of origin
salary	Categorical (2)	Yearly Salary

TABLE 1. UCI-Adult Dataset Structure

Feature	Data Type	Description
age	Continuous (17 - 90)	The insured person's age in years
Sex	Categorical (9)	Gender (Male or Female) of the insured
BMI	Continuous (12285 - 1484705)	A measure of body fat based on height and weight
Children	Categorical (16)	The number of dependents covered
Smoker	Continuous (1-16)	Whether the insured is a smoker
Region	Categorical (7)	The geographic area of coverage
Charges	Categorical (15)	The medical insurance costs incurred by the insured person.

TABLE 2. Healthcare Insurance Dataset Structure

SVMs are inherently binary classifiers as outlined prior, however they can be adapted for non-binary classification. Two commonly used approaches are the One-vs-All (OvA) and the One-vs-One (OvO). Assuming we have k classes that need to be classified, the OvA approach trains k SVMs, one for each class in K, such that each SVM would distinguish k_i from all the others. Alternatively, the OvO technique trains an SVM for each unique pair of values, resulting in $\frac{k(k-1)}{2}$ classifiers. Both approaches allow for non-binary classification, with the final classification being determined by which SVM returns the largest classification score [5]. The notable features of these two approaches are detailed below:

- One-vs-All
  - -- Less resource intensive and reduced training time due to having less classifiers.
  - -- Struggles to classify between classes with similar features.
  - -- Robust for imbalanced datasets.
  - -- Lower risk of over-fitting
- One-vs-One
  - -- More complex model and increased training time due to having more classifiers.
  - -- Performs better when classifying between classes with similar features.
  - -- Can struggle with imbalanced datasets.
  - -- Higher risk of over-fitting due to smaller class subsets.

Throughout the training process, SVM implementations have three primary parameters, these being: C, γ and the kernel function used [8].

- Kernel - The kernel as outlined prior determines how the data points are to be mapped to the higher dimensional plane to facilitate the separation of data.
- γ - This denotes the influence of each training example with smaller values resulting in a less-complex decision boundary susceptible to under-fitting whilst larger values result in a complex decision boundary susceptible to over-fitting [9]. This is outlined in Figure 3.
- C - This is a regularisation term denoting the balance between margin maximisation and the level of acceptable misclassifications, higher values result in a smaller margin but reduced number of misclassifications, with the opposite holding true for smaller values of C [10].

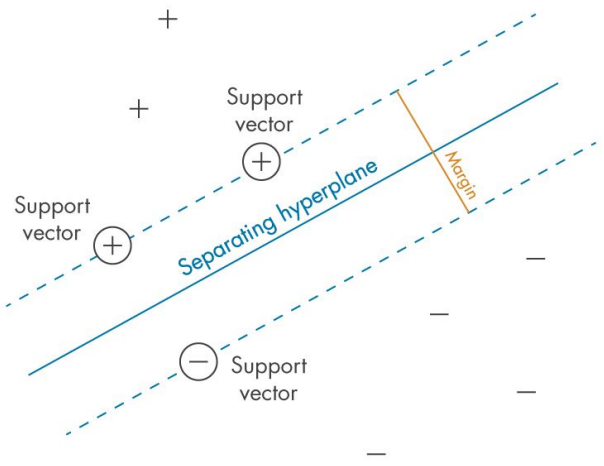

FIGURE 1. SVM Diagram [6]

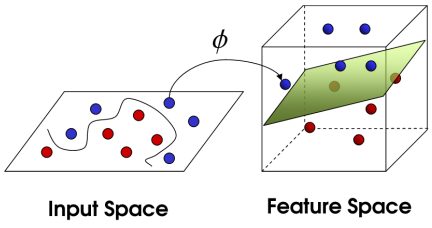

FIGURE 2. Kernel Trick Visualisation [7]

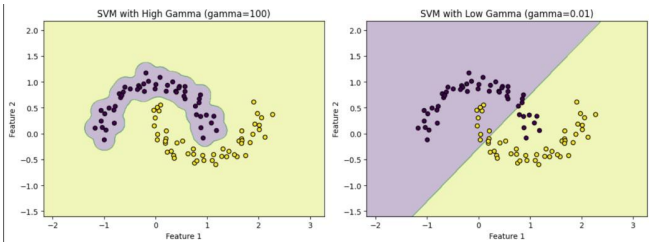

FIGURE 3. Affect of γ on decision boundary [9].

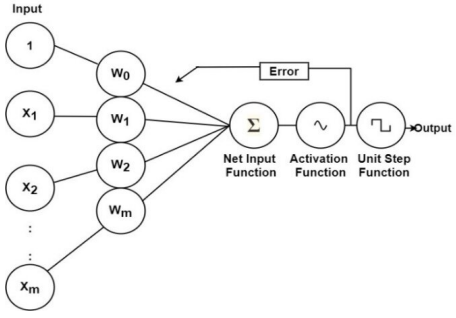

FIGURE 4. Logistic Regression Diagram [11]

2) Logistic Regression

Logistic Regression is a supervised machine learning algorithm used to perform classification tasks, where a model's role is to predict the probability that some instance belongs to a particular class or classes. This technique is mainly used for binary classification, typically employing the sigmoid function (denoted as an S-shaped curve), a method that takes independent variables as input and produces a probability between 0 and 1 as an output. If the output is above a chosen threshold (commonly 0.5), the instance is classified as one class; otherwise, it is classified as the other class. The sigmoid function is defined in 2.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

Where z refers to the linear combination of input features, which is defined in 3.

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n \tag{3}$$

Where X refers to an input feature and β represents the weight assigned to the input feature.

To train a logistic regression model, input data consisting of feature vectors and their corresponding weights are first summed together. The resulting value is passed through an activation function, usually the sigmoid function, which produces a value between 0 and 1. The predicted output is then evaluated using a loss function, which calculates the difference between the predicted value and the ground truth. Using this calculated error, the model then iteratively adjusts its weights using an optimisation function such as gradient descent to minimise the error. In classification scenarios, to decide whether the predicted output falls under a certain class or not, a unit step function is employed, where probabilities which meet a certain threshold are assigned a specific class. This training process can also be visualised in Figure 4. [12–14]

Apart from binary classification, logistic regression can also be applied to other tasks. Here are the primary types:

- **Binary Logistic Regression**: This is the most frequently used type of logistic regression, used for binary classification. The model predicts the probability that an input belongs to one of two classes, often represented as 0 and 1. This is typically achieved with the sigmoid activation function.
- **Multinomial Logistic Regression**: This is used when the target label has more than two classes in no particular order. In this case, the model predicts the probability of each class directly. Instead of the sigmoid function, multinomial logistic regression commonly employs the *softmax function* to handle multiple classes by calculating the probability for each class. The softmax function for a class i is defined in 4.

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{4}$$

where z_i is the linear combination of features for class i and k is the number of classes.

- **Ordinal Logistic Regression**: This type is used when there are multiple classes that follow a natural order (e.g., low, medium, high). Unlike multinomial logistic regression, ordinal logistic regression takes the order of classes into account, producing probabilities that capture the ordinal relationship among classes. [15]

In logistic regression, activation functions are used to calculate the predicted probability. The sigmoid function is the most commonly used activation function for this purpose, however, another commonly used activation function in logistic regression is the softmax function. This function is commonly used in *multinomial logistic regression*, when the model needs to predict the probability of one class out of multiple. Softmax converts the output of each class into a probability, allowing the model to select the class with the highest probability.

In multiclass classification tasks, one common strategy is the *One-vs-Rest (OvR)* approach. In OvR, separate binary logistic regression models are trained for each class. This means that for each class i, a model is trained to distinguish class i from all other classes. During prediction, each binary classifier outputs a probability that an instance belongs to its respective class, and the final classification is based on the highest probability among all models. OvR is advantageous as it simplifies the training process, especially when only basic binary classifiers are available, and can perform well in a range of applications. [16]

3) Random Forest

Random Forests is a model blending technique based on decision trees. A typical issue with individual decision trees is overfitting. During the training of the random forest model, several decision trees are built, and the aggregate of their predictions is taken in order to generate a more precise and reliable outcome. By averaging the results of these numerous decision trees, Random Forests aim to lower the risk of the aforementioned overfitting problem [17].

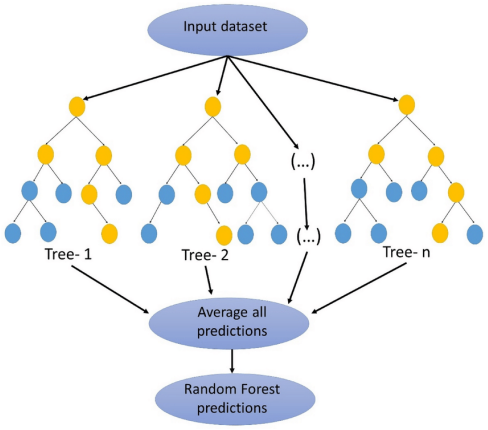

FIGURE 5. Random Forests Structure Diagram [18]

In Random Forests, a random subset of the original data is used to train each tree. This technique is called Bootstrapping (Sampling) [19]. For each tree T_k in the forest, a random subset of the training dataset D is produced with replacement. This new subset D_k has the same size as the original dataset D, but it may contain duplicates of some samples and omit others. As each tree sees a different portion of the data, this leads to diversity among the trees.

Furthermore, instead of taking into account the complete feature set F, each tree is constructed by randomly selecting a subset of features F_k. By including an extra layer of unpredictability, this helps to lessen the correlation between trees and keeps any one factor from dominating the model [19].

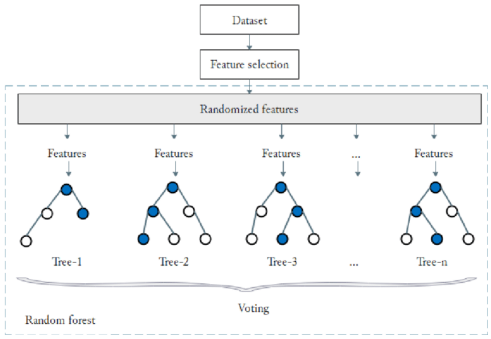

FIGURE 6. Random Feature Selection Diagram [20]

Each decision tree T_k in the forest grows using D_k and F_k and generates its own projections. Due to their independent construction and lack of pruning, the trees can identify intricate patterns within their own subset.

The last step is known as the process of Aggregation. Here, in classification tasks, each tree T_k in the forest "votes" on the projected class $\hat{y}_k$, and the most frequently selected class is chosen as the final prediction $\hat{y}$ as seen in 5.

$$\hat{y} = \text{mode}\left(\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N\}\right) \tag{5}$$

On the other hand, in regression tasks, all tree predictions are averaged. This aggregation process helps reduce the variance of the model, leading to better generalisation on new data as seen in 6.

$$\hat{y} = \frac{1}{N}\sum_{k=1}^{N}\hat{y}_k \tag{6}$$

To sum up, Random Forests integrate several decision trees that have been trained on different data and feature subsets to create a strong model that maximises each tree's advantages while minimising its drawbacks. Because of this, Random Forests may capture intricate patterns without overfitting, making it a strong and dependable methodology for structured data. [21]

B. RESCALING AND NORMALISATION

Rescaling and normalisation are crucial pre-processing techniques used in machine learning to prepare data for model

training. These methods are primarily applied to continuous numerical values to ensure that different features contribute equally to the model's performance, particularly when they operate on different scales or units.

Rescaling and normalisation are terms which encompass several techniques that aim to fit the data within a standard range such as 0 - 1 or transform the data to conform to a particular data distribution [22]. Some commonly used methods include:

- **Min-Max Normalisation** - Rescales the data to fit within the range [0, 1], particularly useful when the relationships between values needs to be maintained whilst ensuring that all features contribute equally to the model. The primary downside of this method is its susceptibility to outliers [23]. This technique uses the equation seen in 7.
- **Max-Abs Normalisation** - Rescales the data to fit within the range [-1, 1], primarily used for sparse data or data centered at zero [22]. This technique uses the equation seen in 8.
- **Z-Score Normalisation** - Normalises the data such that it has a mean of 0 and standard deviation of 1, thus bypassing the issue of outliers. However, its downside is that the data does not follow a uniform scale [23]. This technique uses the equation seen in 10.
- **Log Scaling** - Scales the data using their log value, thereby handling features with exponential growth or highly skewed distributions. It can stabilise variance and make patterns more interpretable, especially in cases including financial data or biological measurements where values can vary over several orders of magnitude [22]. This technique uses the equation seen in 11.
- **Categorical Scaling** - This involves converting categorical data into numeric counterparts through the use of methods such as One-Hot Encoding or Label Encoding. The former creates a boolean column for each unique categorical value denoting whether the row contains that value or not. Label encoding is much simpler, opting to map each unique categorical value to a numerical value [22]. These techniques are visualised in Tables 3, 4 and 5.

$$X_{normalised} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{7}$$

$$X = \frac{X}{|X_{max}|} \tag{8}$$

$$X = \frac{X - \mu}{X_{max} - X_{min}} \tag{9}$$

$$X = \frac{X - \mu}{\sigma} \tag{10}$$

$$X_{scaled} = \log(X + 1) \tag{11}$$

These techniques are used throughout the majority of Machine Learning applications as they provide several useful benefits such as improved model convergence, bias mitigation due to inconsistent feature scales, easier feature comparison, robustness against outliers, better interpretability and improved generalisation [22], [24].

Fruit	Price
apple	5
mango	3
apple	2

TABLE 3. Original Table

Fruit_apple	Fruit_mango	Price
1	0	5
0	1	3
1	0	2

TABLE 4. One-Hot Encoding

Fruit	Price
0	5
1	3
0	2

TABLE 5. Label Encoding

C. CROSS-VALIDATION

Cross-Validation is a widely-utilised technique in Machine Learning, responsible for the evaluation of models by testing their generalisability. The main purpose of Cross-Validation is to prevent models from overfitting, which is a phenomenon involving when a model learns the data it is trained on too closely, thus failing to generalise on new, unseen data. On the other hand, underfitting is the opposite of overfitting, where the model does not learn the training data enough to be able to make accurate predictions on unseen examples.

To perform cross-validation, the data on which the models learns on is split into two sets: the Training set and the Testing Set. The former set is used for the model to learn on. It provides the model with feature-label examples, allowing it to learn patterns and relationships within the data. Once training is complete, the latter set is used to evaluate the model, whose purpose is to test how reliable the model can perform on unseen examples. Thus, cross-validation aims to find a balance between underfitting and overfitting the models, allowing for accurate predictions on new data. [25]

Some of the most popular cross-validation techniques include:

- **Hold-out Cross-Validation** - The dataset is split into two parts: one to be used for training and the other for testing. The split ratio may vary, however some popular training to testing splits are 70-30 or 80-20. This technique has a disadvantage, with that being that if either of the sets are not representative of the entire dataset, problems may arise with regards to the generalisability of the model.
- **k-Fold Cross-Validation** - The dataset is split into k equal subsets. When training, one of the k sets are se-

lected as the testing set, whilst the remaining *k-1* sets are used as the training set. The process is repeated for *k* times, where each iteration selects a different set for the testing set. In the end, the model would have been tested on each subset available. The model's performance is then averaged over all *k* iterations, providing a more stable estimate of its performance. However, depending on the size of *k*, this technique can become very expensive and time-consuming for model training and testing. [26]

- **Leave-One-Out Cross-Validation** - Similar to k-Fold CV, the value of *k* is set to *n*, the total number of samples (data entries) in the dataset. In this method, each sample is used as the testing set once, with the remaining *n-1* samples serving as the training set. The process is repeated for *n* times, and the final performance score is the average over all *n* iterations. The main advantage of this method is that it nearly utilises the full dataset for training in each iteration, however due to it requiring *n* models, it is extremely computationally expensive. Hence, k-fold cross-validation is often preferred over LOOCV to balance accuracy with efficiency.
- **Stratified k-Fold Cross-Validation** - This approach is very similar to traditional k-Fold Cross Validation, however the splitting of the dataset is slightly different. Rather than randomly splitting the dataset into *k* folds, each fold is selected to retain approximately the same percentage of samples per label of the complete dataset. This technique is useful for imbalanced datasets, ensuring that each fold is representative of the whole dataset. [27]

D. DIMENSIONALITY REDUCTION AND FEATURE SELECTION

Dimensionality reduction and feature selection are crucial Machine Learning methods used to reduce the number of features or select the relevant features respectively.

1) Dimensionality Reduction

Dimensionality reduction (DR) comes in various different methods including but not limited to Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), t-SNE (t-Distributed Stochastic Neighbor Embedding) and Isometric Mapping (Isomap).

- **PCA** - Achieves dimensionality reduction through the creation of principal components (PC), which are linear combinations of the original features. Each PC captures as much variance (information) as possible from the data whilst being uncorrelated with the other PCs. Due to this, subsequent PCs capture less data overall and thus are generally ignored leading to dimensionality reduction with minimal data loss [28].
- **LDA** - A supervised machine learning technique used for both classification and dimensionality reduction. The latter achieved through the maximisation of between-class variance to within-class variance, thus ensuring that the classes are as distinct as possible. This is achieved by establishing a linear boundary which separates the different classes. The data is then projected onto the lower dimensional space defined by the linear discriminants, thereby achieving DR. [29].
- **t-SNE** - A non-linear dimensionality reduction technique which converts similarities between data points to joint probabilities and tries to minimie the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. Throughout this process it preserves data pairs, meaning that data points near one another in the original dataset have a higher probability of being near one another than those further away [30].
- **Isomap** - A non-linear dimensionality reduction technique that aims to map complex high-dimensional data into lower-dimensionality whilst preserving the essential relationships between the data points. This is achieved through the use of isometric mapping, which focuses on maintaining the intrinsic geometric structure of the data [31].

2) Feature Selection

Feature selection refers to methods used to choose the most relevant features that contribute effectively to the model's performance, whether for classification, regression, or other types of predictive and analytical tasks. This takes the forms of various methods including but no limited to: Filter methods, wrapper methods, embedded methods and hybrid methods [32].

- **Filter methods** - Assess the relevance of features based on statistical measures based on their correlation with the target variable. Examples include chi-square test, information gain, and correlation coefficient. Features are selected based on their individual characteristics, without considering the machine learning algorithm used.
- **Wrapper methods** - Utilise machine learning algorithms to select features based on their impact on model performance. This is achieved by wrapping the feature selection process around the model training and evaluating the model's performance to determine the optimal subset of features.
- **Embedded methods** - Incorporate the feature selection process into model training. Features are then selected based on their contribution to the model's predictive power.
- **Hybrid methods** - Combine aspects of filter, wrapper, and embedded methods. They leverage the strengths of multiple techniques.

E. QUANTITATIVE MEASUREMENTS

To evaluate machine learning models, different quantitative metrics are needed in this implementation. The metrics used include accuracy, precision, recall, and F1-score. Additionally, these values can be used to plot curves, these being the ROC and Precision-Recall curves, which provide deeper insights into model performance and help identify the optimal

threshold for specific models. The following equations are a list of metrics that will be utilised during the evaluation process:

Accuracy measures how the model performs across all classes. It gives the rate at which the model performs correctly across all samples, ranging from 0 to 1, where a value close to 1 indicates a high level of correct predictions, while a value closer to 0 indicates poor performance.

$$\textbf{Accuracy} = \frac{\text{True Positive (TP)} + \text{True Negative (TN)}}{\text{Total Number of Samples}} \tag{12}$$

Precision defines how many items retrieved by the model were relevant. Similar to accuracy, this metric ranges from 0 to 1 where a value near 1 indicates that most positive predictions are correct, while a value close to 0 indicates many false positives.

$$\textbf{Precision} = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Positive (FP)}} \tag{13}$$

Recall describes how many of the relevant items were successfully retrieved. Recall too ranges from 0 to 1, where a value near 1 indicates that the model identified almost all positive cases correctly, while a value close to 0 means many true positives were missed.

$$\textbf{Recall} = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Negative (FN)}} \tag{14}$$

The F1-Score serves as a harmonic mean between precision and recall, ranging from 0 to 1. A value close to 1 indicates an optimal balance between precision and recall, while a value close to 0 indicates that either the precision or recall values (or both) is low. [33, 34]

$$\textbf{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

A confusion matrix is a tool used to assess the quality of predictions against an expected ground truth. For a model to perform well, most of the predicted positives should match with true positives, same way the predicted negatives ought to match with the true negatives.

$$\textbf{Confusion Matrix} = \begin{array}{|c|c|c|} \hline & \textbf{Predicted Positive} & \textbf{Predicted Negative} \\ \hline \textbf{Actual Positive} & \text{True Positive} & \text{False Negative} \\ \hline \textbf{Actual Negative} & \text{False Positive} & \text{True Negative} \\ \hline \end{array} \tag{16}$$

A Receiver Operating Characteristic (ROC) curve plots the True Positive Rate against the False Positive Rate to evaluate a model's performance at different thresholds. The area under the curve (AUC) indicates performance: 1.0 indicates a perfect performance whilst 0.5 indicates that the model is guessing randomly. A curve closer to the top left corner suggests a model with better predictive ability, maximising true positives and minimising false positives.

Furthermore, Precision-Recall curves are plots which represent the trade-off between precision and recall. These curves are valuable for evaluating models with imbalanced datasets, where the number of positive instances is much smaller than negative ones. The interpretation of a PR curve involves examining how well a model maintains precision while recall increases. A curve that approaches the top right corner of the plot signifies a model with high precision and high recall, indicating strong performance. If the curve shows a sharp decline in precision as recall improves, it may suggest that the model's performance weakens at certain thresholds.

Mean Average Precision (mAP) is a metric that considers both the precision and recall values across multiple thresholds.

$$\textbf{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i \tag{17}$$

N is the number of classes, and AP_i is the Average Precision (AP) for class i. Average Precision (AP) for a single class is the area under its corresponding precision-recall curve. A high value indicates a better balance between precision and recall across different thresholds. [35]

III. DATA PREPARATION

A. DATA EXPLORATION

1) UCI-Adult Dataset

As outlined in the introduction, this dataset comprises of 32561 rows and 15 features, with the *salary* feature being used as the predicted label. The datatypes and descriptions of each feature are described in Table 1 in the Introduction. In this table, if the feature is categorical, the value in brackets refers to the number of unique values within that label, whilst if it is a continuous feature, the values show the range of possible values within that feature. Furthermore, as can be seen in Table 6, each categorical variable's number of unique values was retrieved. The results from this table would then help decide which features require a log1p transformation due to high-skewness.

Feature	Skewness
age	0.557663
workclass	-0.089160
fnlwgt	1.447703
education	1.231618
education-num	-0.309500
marital-status	2.155419
occupation	0.351007
relationship	0.775054
race	3.520308
sex	0.719449
capital-gain	11.949403
capital-loss	4.592702
hours-per-week	0.228759
native-country	5.117206
salary	1.211687

TABLE 6. UCI Adult Feature Skewness Values

Following this, some dataset features were evaluated to better understand the data that will be used during training and evaluation. As can be seen in Figure 7, approximately 70% of the entries in the dataset are male, with the remaining 30% female.

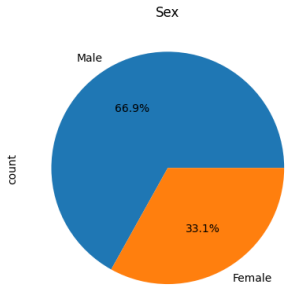

FIGURE 7. UCI Adult - Sex Distribution

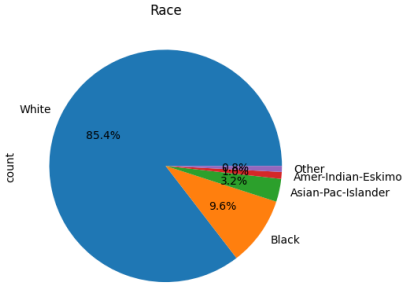

FIGURE 8. UCI Adult - Race Distribution

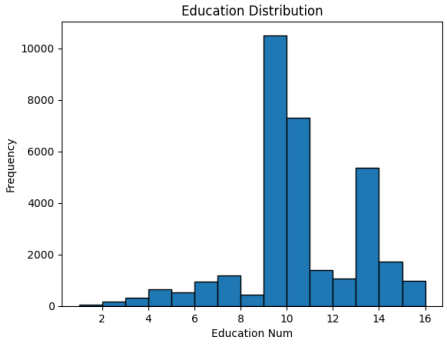

FIGURE 9. UCI Adult - Education Number Distribution

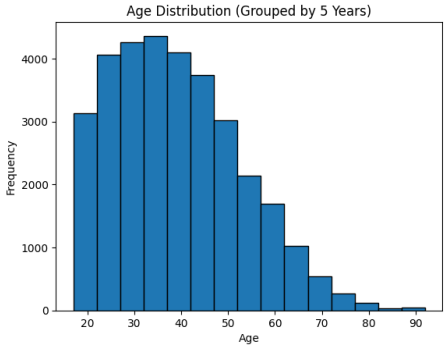

FIGURE 10. UCI Adult - Age Distribution

Moreover, a pie chart for the race distribution was created, as can be seen in Figure 8. The most frequently present race is *White* at 85%, followed by *Black* at 9.6% and the rest.

To understand the range of education numbers, a pie chart was depicted as seen in Figure 9. It can be concluded that the dataset offers a vast range of education levels, with most having studied up until High School level (Level 9).

Finally, with respect to the age distribution, as can be seen in Figure 10, a diverse range of ages was present within the dataset, with most people being aged around 30-40.

2) Healthcare Insurance Dataset

As outlined in the introduction, this dataset comprises of 1338 rows and 7 features, with the *charges* feature being used as the predicted label. The datatypes and descriptions of each feature are described in Table 2 in the Introduction. Furthermore, similar to what was done with the UCI-Adult dataset, this table depicts each categorical variable's number of unique values. Furthermore, as can be seen in Table 7, each categorical variable's number of unique values was retrieved and displayed.

Feature	Skewness
age	0.054781
sex	-0.019469
bmi	0.283914
children	0.937421
smoker	-1.463601
region	0.039068
charges	1.515391

TABLE 7. Healthcare Insurance Feature Skewness Values

Furthemore, to understand the dataset, a few charts and diagrams were created to better understand the distributions of feature data within the dataset. As shown in Figure 11, the sex label is evenly distributed across the dataset.

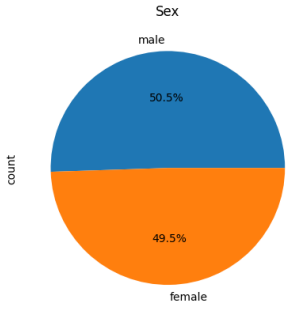

FIGURE 11. Healthcare Insurance - Sex Distribution

Additionally, the *region* label was found to contain four areas within a particular location, with all four unique values being approximately uniformly distributed as can be seen in Figure 12.

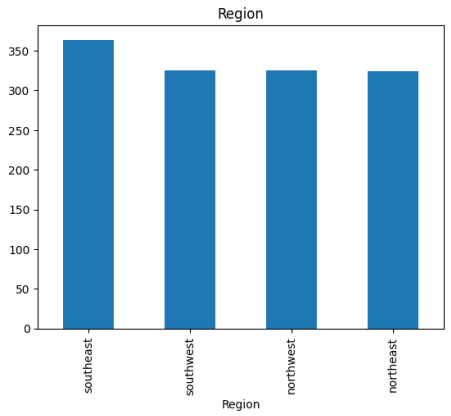

FIGURE 12. Healthcare Insurance - Region Distribution

With respect to the age distribution, as can be seen in Figure 13, whilst a high percentage of the entries are aged around 20, other ages were also represented.

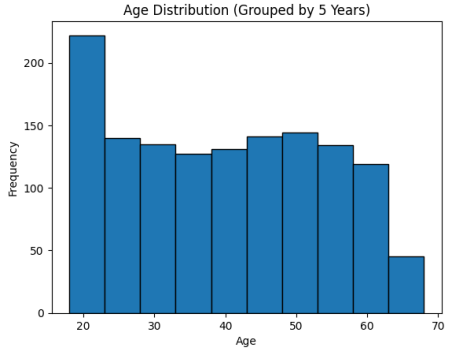

FIGURE 13. Healthcare Insurance - Age Distribution

Finally, the Body Mass Index (BMI) distribution, as seen in Figure 14, depicts a high concentration towards the 25-35 range of BMI values.

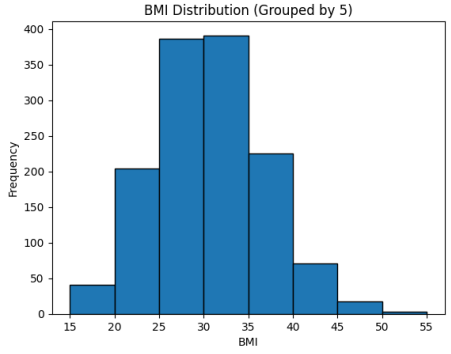

FIGURE 14. Healthcare Insurance - BMI Distribution

B. DATA CLEANING

1) UCI-Adult Dataset

In line with our findings in the data exploration section the UCI-Adult dataset underwent data cleaning to be used for model training. Although three distinct models were used throughout this research, the data cleaning approach remained consistent across all three. Firstly, the number of rows in the dataset was determined, this being 32561. This allowed us to easily remove duplicate rows and those containing null values as these numbered to 24 and 2398 rows respectively, leaving us with a total of 30139 rows. Imputation was considered for dealing with the null entries however given that they encapsulated only 7.44% of total rows it was determined to be more time efficient to remove them.

Furthermore, it was assessed that 85% of all remaining entries within the dataset had the native-country label as United-States. Given this severe skewness, the decision was taken to remove the native-country column and only consider entries associated with the United-States, resulting in a loss of 2652 rows and retaining 27487 rows viable for training. Additionally, the fnlwgt and education columns were removed as the latter denotes a mapping between the education-num and its corresponding categorical value whereas the fnlwgt column refers to sampling weights which are irrelevant to the final classification.

Following the initial data cleaning, the values need to be converted into their continuous equivalents. For categorical features such as workclass, marital-status, occupation, relationship and race, One-Hot Encoding (OHE) was used. This was done as these features do not have any order and ensures that each feature is given equal weight. The drop first approach was also used, which involves removing a column from the resultant OHE set. This is done to avoid the Dummy Variable Trap, which occurs when you have Multicollinearity. This phenomenon transpires when one of the new columns can be perfectly predicted by the others (i.e., if you know the values of *n-1* columns, you can deduce the value of the *nth* column). This can distort the model's interpretation and lead to redundant information [36]. Furthermore, binary categorical features such as sex and salary were converted using a simple mapping, achieving the same result as OHE with drop first. In relation to continuous features the log1p value was assigned to skewed features consisting of capital-gain and capital-loss, this was done to reduce the skewness severity. Finally, the continuous features underwent z-score normalisation to ensure equal importance relative to the label.

2) Healthcare Insurance Dataset

Based on the findings made during the data exploration section, a few changes were performed on the Healthcare Insurance dataset prior to being used for model training. Similar to the other dataset, whilst three models were trained on this dataset, the data cleaning remained consistent. The base number of rows within this dataset was 1338, however after deduplication of rows, this was reduced to 1337. Furthermore, no null value rows were found within the dataset. Given that

only one row was removed, imputation was determined to not be crucial to implement for this dataset.

Since the dataset is originally designed for a regression problem, the *charges* column was to be converted from a continuous to a categorical column. To do so, the label was grouped into 5 groups, these being:

- **Very Low**: ≤ 5000
- **Low**: $5001 - 10000$
- **Moderate**: $10001 - 15000$
- **High**: $15001 - 20000$
- **Very High**: > 20000

Once this cleaning was performed, the *region* column was converted from categorical to continuous using One-Hot Encoding (OHE). Moreover, the *sex* and *smoker* class were converted to boolean variables. No log1p transformations were required since the continuous columns were not highly skewed. Finally, the continuous features underwent z-score normalisation to ensure equal importance relative to the label.

IV. EXPERIMENTS

A. MODEL IMPLEMENTATIONS

For each of the models implemented, two separate models were developed. The first model would be trained on the UCI-Adult dataset to perform binary classification on the *salary* label. The second model would be trained on the Healthcare Insurance dataset to perform multiclass classification on the *Charges* label. This allowed comparisons to be made between how the models perform on binary and multi-class classification tasks.

1) SVM

To implement Support Vector Machines, the *sklearn* library's SVC model was used. The features and target variable were extracted from the normalised dataframe, and the dataset was split into training and testing sets, with 20% allocated for testing. The model was trained on the training set, predictions were made on the test set, and the performance was evaluated using appropriate metrics.

Furthermore, to address multiclass classification, two variants of SVM were implemented: One-vs-One SVMs and One-vs-Rest SVMs. For the former, the *OneVsRestClassifier* was used with *SVC* as the base estimator, whereas the *OneVsOneClassifier* was used for the latter. A few arbitrary hyperparameter sets were tested to check that the model performance after training was satisfactory prior to evaluation. To avoid manually running multiple sets of hyperparameters on the model separately and comparing each result, a grid search was conducted using *GridSearchCV* to efficiently optimise the model's performance. A parameter grid was defined, including key hyper-parameters such as *C*, *kernel*, *gamma*, and *degree*. The list of hyperparameters and their values are:

- **C**: {0.1, 1, 10, 100}
- **kernel**: {rbf, linear, poly}
- **gamma**: {scale, auto, 0.1, 0.01}
- **degree**: {3, 4, 5}

To ensure robust evaluation, a stratified k-fold cross-validation (SKFCV) strategy was employed with five splits, enabling a balanced distribution of classes across folds. The model was then fitted to the training data, allowing the selection of the best hyper-parameters and providing an optimised classifier for multi-class prediction.

2) Logistic Regression

A similar methodology was employed for Logistic Regression, where the multiclass classification variant was implemented. The *LogisticRegression* model from *sklearn* was initialised, and a grid search was performed using *GridSearchCV* to optimise hyperparameter selection.

A parameter grid was defined, including:

- **C**: {0.1, 1, 10, 100}
- **solver**: {saga, lbfgs}
- **multi_class**: {multinomial, ovr, auto}
- **max_iter**: {100, 200, 300}

Similar to the SVM implementation, SKFCV was used with five splits. This methodology enabled the optimal selection of hyperparameters and ensured a well-tuned multinomial logistic regression model for multi-class prediction.

3) Random Forests

The Random Forests model was implemented using the *RandomForestClassifier* from the *sklearn.ensemble* module. First, to prepare the data for implementation, the normalised dataframe, *zNormalisedDF*, was used to extract features and the target variable. The dataset was subsequently divided into training and validation sets, with 20% of the dataset designated for validation, in order to assess the model's performance.

To investigate different model configurations, a parameter grid was created. These included:

- **n_estimators**: {100, 200, 300}
- **max_depth**: {None, 10, 20}
- **min_samples_split**: {2, 5, 10}
- **min_samples_leaf**: {1, 2, 4}
- **bootstrap**: {True, False}

These parameters were selected to optimise the model in light of the characteristics of the classification problem and the data itself.

A stratified 5-fold cross-validation method, implemented by *StratifiedKFold*, was used to guarantee a balanced distribution of classes in each fold and to reduce any biases. This approach excels in maintaining class proportionality across training and validation subsets.

To maximise accuracy, the hyperparameter tuning procedure was automated using the *GridSearchCV* program. This method systematically evaluates various parameter combinations across the cross-validation folds, greatly increasing the efficiency of model selection. The optimal set of hyperparameters was identified when *GridSearchCV* was fitted to the training data, allowing the Random Forest model to be fine-tuned to its most efficient setup.

Lastly, Python's *pickle* module was used to save the top-performing model configuration for operational use and additional analysis. This facilitates both practical use and continuous development because it is simple to load for future forecasts or to replicate the study's findings.

B. EVALUATION & COMPARISON

For each model evaluation, the following performance metrics were used:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

1) SVM

As can be seen in Table 8, all SVM models performed well on both datasets and with both SVM variants. It could be noted that whilst One vs One is more computationally expensive in terms of time taken to train the models, it gained slightly better results than its One vs Many counterparts.

Model	Accuracy	Precision	Recall	F1-Score
One-vs-Many (Salary)	0.85	0.85	0.85	0.85
One-vs-One (Salary)	0.86	0.85	0.86	0.85
One-vs-Many (Medical)	0.87	0.87	0.87	0.86
One-vs-One (Medical)	0.89	0.89	0.89	0.89

TABLE 8. Performance metrics for different SVM models

Additionally, Confusion Matrices were generated to better understand the number of correct and incorrect classifications being done by the model. This can be seen in Figures 15, 16, 17 and 18, which depicts that many test cases were appropriately classified as True Positives or True Negatives.

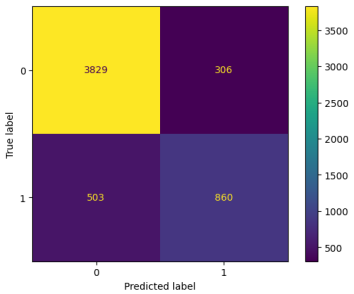

FIGURE 15. SVM Confusion Matrix (OvM) for Salary Dataset

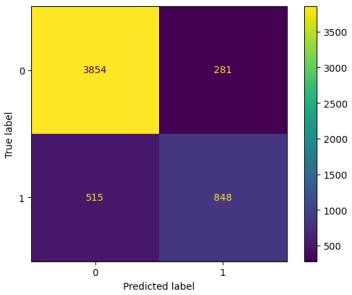

FIGURE 16. SVM Confusion Matrix (OvO) for Salary Dataset

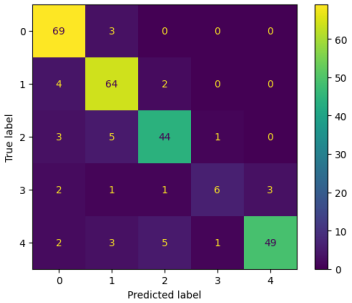

FIGURE 17. SVM Confusion Matrix (OvM) for Healthcare Dataset

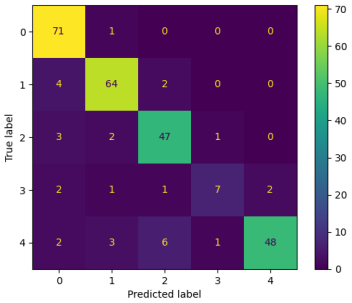

FIGURE 18. SVM Confusion Matrix (OvO) for Healthcare Dataset

Furthermore, as can be seen in the ROC-AUC and Precision-Recall curves in Figures 19, 20, 21 and 22, the models performed well. In the ROC-AUC curve, high areas under the curve were recorded, meaning that the model is able to distinguish between classes well. Moreover, the precision-recall curve denoted that whilst most classes are being accurately predicted, this was not necessarily the case for the fourth class of the Medical Model (*High*) which only retrieved an Average Precision of 0.63.

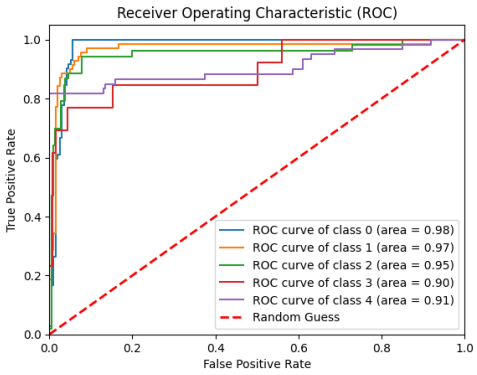

FIGURE 19. SVM ROC-AUC Curve for Salary Dataset

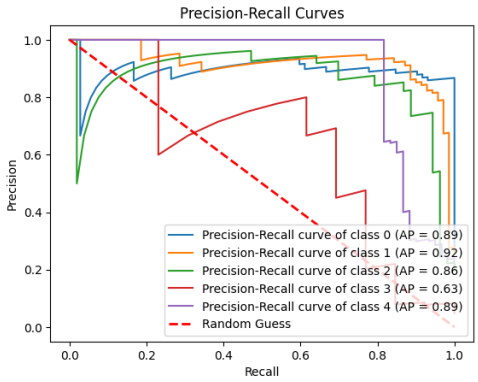

FIGURE 20. SVM PR Curve for Salary Dataset

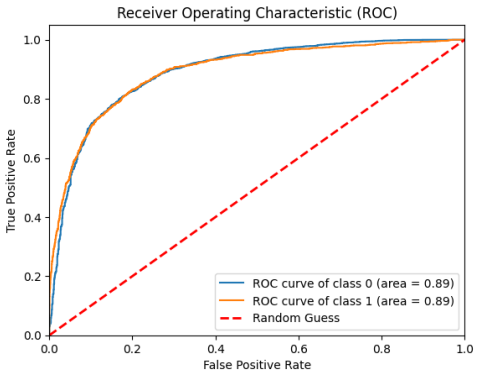

FIGURE 21. SVM ROC-AUC Curve for Healthcare Dataset

When evaluating the models, the best hyperparameters of the four models can be seen in Table 9. As can be noted, most of the hyperparameters remained consistent for both datasets.

To better understand the effect of these hyperparameters on the model's performance, we trained multiple separate models, each with a different hyperparameter value. For this evaluation, we started with the best hyperparameter set and altered one hyperparameter value per model. The hyperparameter sets used in this evaluation are shown in Table 10. The conclusions made can be seen in Section IV-B4.

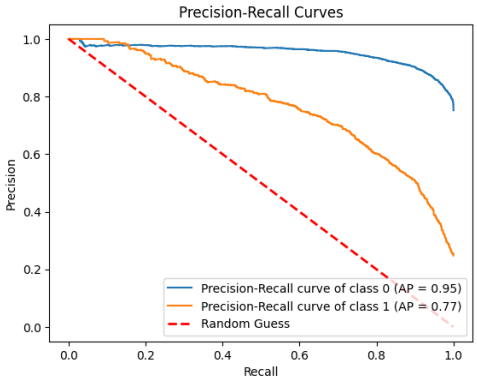

FIGURE 22. SVM PR Curve for Healthcare Dataset

Model	C	gamma	kernel	type
SVM (Salary)	100	0.01	rbf	ovo
SVM (Salary)	10	0.1	rbf	ovm
SVM (Medical)	10	0.1	rbf	ovo
SVM (Medical)	10	0.1	rbf	ovm

TABLE 9. Best Hyperparameters for SVM Models

Model	C	gamma	kernel	type
SVM (Medical)	1	0.01	rbf	ovm
SVM (Medical)	1	1	rbf	ovm
SVM (Medical)	100	0.01	rbf	ovm
SVM (Medical)	100	1	rbf	ovm
SVM (Salary)	1	0.01	rbf	ovm
SVM (Salary)	1	1	rbf	ovm
SVM (Salary)	100	0.01	rbf	ovm
SVM (Salary)	100	1	rbf	ovm

TABLE 10. Hyperparameter Testing for SVM Models

2) Logistic Regression

For Logistic Regression's models, as can be seen in Table 11, both models trained on the datasets performed well. As expected, Logistic Regression performed slightly better with a binary classification task rather than a multi-class classification one, since the UCI Adult dataset model obtained a higher accuracy than its Healthcare Insurance counterpart.

Model	Accuracy	Precision	Recall	F1-Score
LR (Medical)	0.81	0.79	0.73	0.74
LR (Salary)	0.84	0.80	0.77	0.78

TABLE 11. Performance Metrics for Logistic Regression Models

As depicted in Figures 23 and 24, both models obtained positive results in the confusion matrices, with few misclassifications being made.

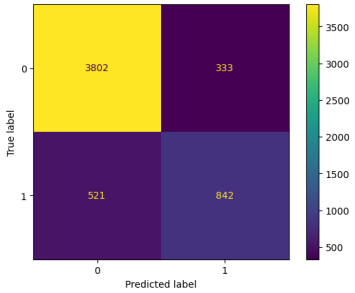

FIGURE 23. Logistic Regression Confusion Matrix for Salary Dataset

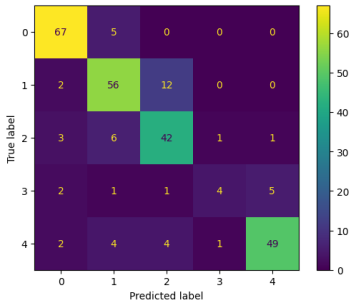

FIGURE 24. Logistic Regression Confusion Matrix for Healthcare Dataset

Moreover, as can be seen in Figures 25 and 26 a high AUC and Average Precision were obtained for the UCI Adult dataset model. Note that these graphs were not generated for the Healthcare Insurance dataset, this is because these types of graphs are used for binary classification tasks, thus it would not make sense to use it in a multi-class classification scenario.

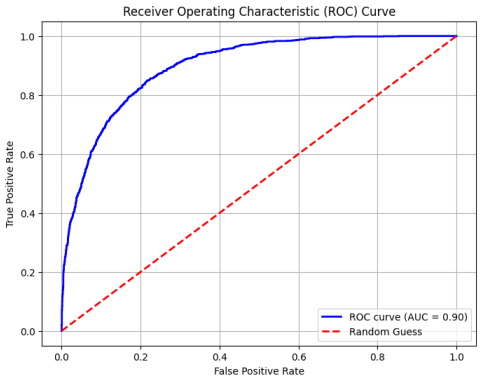

FIGURE 25. Logistic Regression ROC-AUC Curve for Salary Dataset

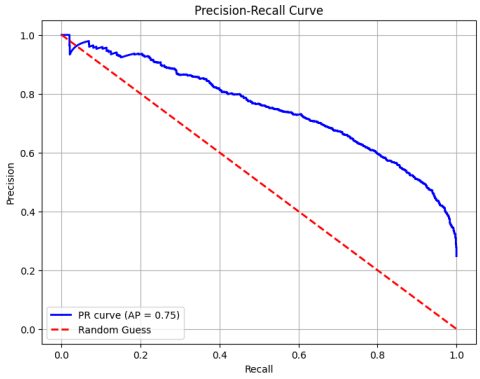

FIGURE 26. Logistic Regression PR Curve for Salary Dataset

When evaluating the models, the best hyperparameters of the two models can be seen in Table 12. As can be noted, most of the hyperparameters remained consistent for both datasets.

Model	C	max_iter	multi_class	solver
LR (Medical)	100	100	multinomial	lbfgs
LR (Salary)	1	100	multinomial	lbfgs

TABLE 12. Best Hyperparameters for Logistic Regression Models

To better understand the effect of these hyperparameters on the model's performance, we trained multiple separate models, each with a different hyperparameter value. For this evaluation, we started with the best hyperparameter set and altered one hyperparameter value per model. The hyperparameter sets used in this evaluation are shown in Table 13. The conclusions made from this testing can be seen in Section IV-B4.

Model	C	max_iter	multi_class	solver
LR (Medical)	0.1	50	multinomial	lbfgs
LR (Medical)	0.1	200	multinomial	lbfgs
LR (Medical)	200	50	multinomial	lbfgs
LR (Medical)	200	200	multinomial	lbfgs
LR (Salary)	0.1	50	multinomial	lbfgs
LR (Salary)	0.1	200	multinomial	lbfgs
LR (Salary)	10	50	multinomial	lbfgs
LR (Salary)	10	200	multinomial	lbfgs

TABLE 13. Hyperparameter Testing for Logistic Regression Models

3) Random Forests

The results of testing the Random Forest model's performance on two different datasets, proved its strong points while also highlighting variations specific to each dataset's features.

In both situations, the model demonstrated a high degree of performance, achieving an accuracy of 86.19% on the salary dataset and 85.82% on the medical dataset. Both datasets showed great precision, recall, and F1-scores with the salary dataset showing especially high precision and recall for the

majority class. The medical dataset, however, showed more variability in recall among different classes, reflecting the model's varied sensitivity to class details in a multi-class setting.

As can be seen in Figures 27 and 28, although the model is generally effective across both datasets, these ROC curves show subtle differences between classes in the medical dataset, which could guide future model refinement or training data adjustments to better balance predictive accuracy and class representation.

The model's ability to balance the trade-off between accuracy and recall across several classes is shown by the characteristics of the precision-recall curves for both datasets. These can be seen in Figures 29 and 30 below.

Overall, these precision-recall curves features are consistent with earlier findings, highlighting the model's advantages and identifying particular areas where performance could be improved, especially when dealing with classes that have less representation or more complicated decision limits.

Regarding hyperparameters, the model employed an elaborate setup for the salary dataset, with 300 trees and no restrictions on tree depth, appropriate for its greater size and complexity. On the other hand, for the medical dataset, a maximum depth of 10 and fewer trees (200) were added to the settings, in order to optimise for a smaller, potentially less varied dataset. Table 14 shows the best hyperparameters for both models trained on the different datasets. These unique hyperparameter values are probably essential to attaining the excellent model performance levels observed in both datasets.

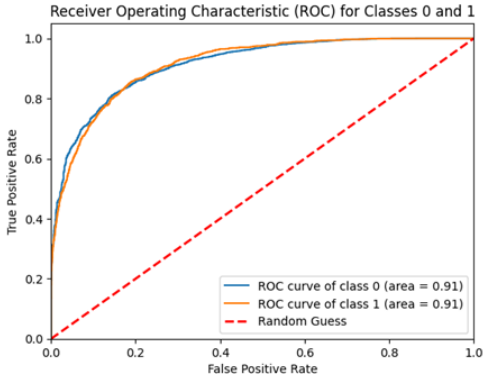

FIGURE 27. Random Forests ROC Curve for Salary Dataset

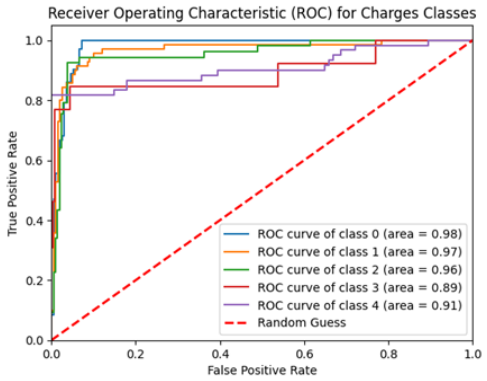

FIGURE 28. Random Forests ROC Curve for Healthcare Dataset

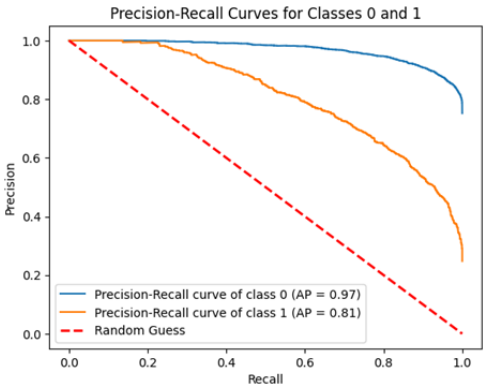

FIGURE 29. Random Forests PR Curve for Salary Dataset

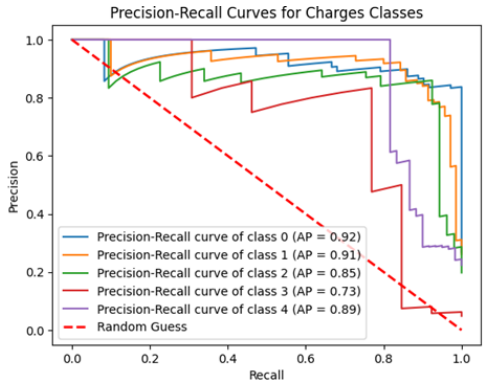

FIGURE 30. Random Forests PR Curve for Healthcare Dataset

Model	n_estimators	max_depth	min_samples_split	min_samples_leaf
RF (Medical)	200	10	5	2
RF (Salary)	300	None	2	4

TABLE 14. Hyperparameter Testing for Random Forest Models

To better understand the effect of these hyperparameters on the model's performance, we trained multiple separate models, each with a different hyperparameter value. For this evaluation, we started with the best hyperparameter set and altered one hyperparameter value per model. The hyperparameter sets used in this evaluation are shown in Table 15. The conclusions made from this testing can be seen in Section IV-B4.

Model	n_estimators	max_depth	min_samples_leaf	min_samples_split
RF (Medical)	100	10	2	5
RF (Medical)	500	10	2	5
RF (Medical)	200	20	2	5
RF (Medical)	200	None	2	5
RF (Medical)	200	10	10	5
RF (Medical)	200	10	2	10
RF (Salary)	100	10	2	5
RF (Salary)	500	10	2	5
RF (Salary)	200	20	2	5
RF (Salary)	200	10	10	5

TABLE 15. Hyperparameter Testing for Random Forest Models

4) Hyperparameter Testing

During evaluation, to assess individually how a modification in the parameters may affect the output with respect to a particular demographic, the *sex* feature was changed for the Healthcare dataset and the *race* feature was changed for the UCI-Adult dataset. With respect to the Healthcare dataset models, the base input given consisted of the following feature values as seen in Table 16.

Feature	Value
Age	40
Sex	Female
BMI	30.0
Children	1
Smoker	No
Region	southwest

TABLE 16. Healthcare Base Input

For the base input, the predictions across all models consistently classify the charges as *Low (5001 - 10000)*, demonstrating that the input features strongly align with this category. This consistency highlights the robustness of the models in handling this specific scenario. However, the prediction probabilities reveal nuances based on the hyperparameters assigned to each model.

For SVM models, varying the `C` hyperparameter (from 1 to 100) shows that lower values result in slightly less confident predictions, due to a broader distribution of probabilities across other categories being calculated. In contrast, higher `C` values concentrate probabilities in the dominant *Low* category, indicating stronger confidence. Similarly, the `gamma` hyperparameter has a significant effect: lower values (e.g., `gamma=0.01`) produce more distributed probabilities, while higher values (e.g., `gamma=1`) lead to greater confidence in the dominant class, although potentially coming at the cost of overfitting.

For Logistic Regression models, the `C` hyperparameter also impacts probability distributions. Lower `C` values, such as `C=0.1`, result in broader distributions, with noticeable effects on the *Very Low* and *Moderate* categories. Higher `C` values (e.g., `C=200`) concentrate probabilities in the *Low* category, reflecting greater confidence. Interestingly, changes in the `iter` hyperparameter (e.g., 50 vs. 200) in Logistic Regression models show negligible impact, suggesting that convergence occurs early in the optimisation process.

Random Forest models further support this consistent prediction of *Low (5001 - 10000)* charges. However, their probability distributions vary based on hyperparameters. Increasing the `n_estimators` (e.g., 100 vs. 500) shows minimal impact on predictions, with consistently high probabilities for the *Low* category and small variations in other categories. Adjusting the `maxDepth` parameter (`maxDepth=20` vs. `maxDepth=None`) yields near-identical results, with extremely high confidence in the *Low* category, suggesting a well-optimised tree depth. Modifying the `minSamplesLeaf` and `minSamplesSplit` parameters demonstrates sensitivity to these settings: higher values (e.g., `minSamplesLeaf=10`) result in slightly more distributed probabilities, indicating a more generalised model, whereas lower values focus confidence heavily on the dominant class.

Overall, SVM models exhibit higher confidence in predictions compared to Logistic Regression, especially with higher `C` and `gamma` values. Random Forest models also show strong confidence, particularly with an optimised depth and split parameters. The probabilities for *Very High* charges remain consistently low across all models, reflecting the input data's alignment with the *Low* category. These findings illustrate how hyperparameters influence model behaviour and confidence while maintaining consistency in predictions.

When changing the input feature from *Female* to *Male*, the predictions remain consistent across models, classifying the charges as *Low (5001 - 10000)*. However, noticeable differences arise in the probability distributions.

For SVM models, the *Low* category generally has slightly higher confidence for males, particularly with `C=100` (89.95% for males vs. 89.41% for females). Probabilities for *Very Low* charges also increase slightly for males in most configurations. For instance, with `C=1`, the probability for *Very Low* charges increases from 4.0% for females to 6.99% for males. Similarly, with `gamma=0.01`, probabilities for *Very Low* charges increase from 6.52% to 8.96%, indicating a subtle shift in the probability distribution based on gender.

For Logistic Regression models, probabilities for the *Low* category are slightly reduced for males at both lower and higher `C` values. For instance, at `C=0.1`, the probability decreases from 49.04% for females to 48.56% for males, while at `C=200`, it decreases from 68.63% to 67.66%. Notably, probabilities for *Very Low* charges are consistently higher for males across all configurations. For example, at `C=200`, the probability for *Very Low* charges increases from 8.8% for females to 14.65% for males. These shifts suggest that

the gender feature subtly influences model confidence, with males yielding a broader distribution of probabilities across categories compared to females.

For Random Forest models, the predictions remain highly consistent for both genders, but the probability distributions reveal some differences. For example, with `n_estimators=100`, the probability for *Very Low* charges increases slightly for males (0.94% vs. 0.78% for females), while the probability for *Low* charges remains stable at approximately 90.73%. Similarly, with `maxDepth=20` or `maxDepth=None`, the model shows extremely high confidence in the *Low* category (98.08% for males vs. 97.09% for females). Adjusting `minSamplesLeaf=10` produces more distributed probabilities for males, with the probability for *Very Low* charges increasing from 4.63% for females to 5.45% for males and the probability for *Moderate* charges increasing from 4.5% to 7.76%. These differences indicate that gender subtly shifts the distribution of probabilities in Random Forest models, particularly for the *Very Low* and *Moderate* categories.

Overall, the results across all models suggest that gender exerts an influence on the probability distributions, with males generally yielding higher probabilities for *Very Low* charges and slightly less concentrated predictions for the *Low* category. These differences, while small, highlight the sensitivity of the models to input feature variations, even when predictions remain consistent.

For the UCI-Adult dataset, the following base input features were used as seen in Table 17.

Feature	Value
Age	35
Workclass	Self-emp-inc
Education	Bachelors
Marital Status	Widowed
Occupation	Tech-support
Relationship	Not-in-family
Race	White
Sex	Male
Capital Gain	10000
Capital Loss	0
Hours per Week	40

TABLE 17. UCI-Adult Base Input

Most models predict the salary class as >50K, demonstrating strong consistency with the input features such as high capital gain, a Bachelor's degree, and self-employment. However, the SVM model with *gamma=1* is an outlier, predicting the salary class as $\leq 50K$ with a high confidence of 75.6%. This suggests that the choice of *gamma=1* in SVM likely results in overfitting or underfitting to specific patterns in the data.

For SVM models, increasing `C` from 1 to 100 improves the confidence in predicting >50K, with class probabilities increasing from 71.79% to 90.88%. Lowering *gamma* to 0.01 leads to predictions favouring >50K but with reduced confidence (64.79%). On the other hand, the outlier prediction with *gamma=1* highlights the sensitivity of SVM models to the *gamma* parameter when capturing feature relationships.

Logistic Regression models, in contrast, are less sensitive to hyperparameter variations. Adjusting `C` (e.g., from 0.1 to 10) or iterations (50 vs. 200) results in only minor changes in class probabilities, with the >50K class consistently predicted at a confidence level of approximately 52–54%. For instance, at `C=0.1`, the model predicts >50K with 52.99% confidence, while at `C=10`, the confidence increases slightly to 53.11%.

For Random Forest models, predictions show high confidence for the >50K class across all configurations, with slight variations based on hyperparameter settings. For example, with `n_estimators=100`, the probability of >50K is 80.12%, which increases marginally to 80.38% when `n_estimators=500`. Adjusting the maximum tree depth significantly impacts confidence levels, with `maxDepth=10` yielding a confidence of 76.28% for >50K, while `maxDepth=20` increases this to 83.22%. Similarly, tuning `minSamplesLeaf=10` results in a confidence of 75.62% for >50K, while reducing `minSamplesSplit=10` leads to a confidence of 81.38%. These variations demonstrate the importance of tree depth and data splitting parameters in shaping the predictions of Random Forest models.

When comparing the models, SVM generally exhibits higher confidence in predicting >50K (e.g., 90.88% with `C=100`) than Logistic Regression models. However, Random Forest models achieve comparable confidence levels (e.g., 83.22% with `maxDepth=20`) while maintaining greater stability across hyperparameter configurations. The SVM model's sensitivity to *gamma* remains a significant observation, with *gamma=1* leading to a significant deviation from the majority of predictions.

In summary, the given input features strongly favour the >50K class, with most models agreeing. Random Forest models exhibit robust predictions with high confidence, while SVM models demonstrate greater variability based on hyperparameter settings. Logistic Regression models show less sensitivity to hyperparameter changes but exhibit lower confidence levels compared to SVM and Random Forest models.

Changing the input feature for race from White to Black impacts the class probabilities across models, although the overall predictions remain largely consistent for most models. Notably, the SVM models with `C=1`, `C=100`, and $\gamma = 0.01$ still predict a salary class of >50K, but with reduced confidence. For instance, with `C=100`, the probability of the >50K class drops from 90.88% (for White) to 82.44% (for Black). Similarly, the `C=1` and $\gamma = 0.01$ models show a moderate decline in confidence compared to the White feature input.

The SVM model with $\gamma = 1$ continues to predict a salary class of $\leq 50K$, but the probability for this prediction increases slightly from 75.6% (White) to 76.02% (Black), indicating stronger confidence in this classification. This suggests that this model is particularly sensitive to the change in race, potentially reflecting its learned patterns from the training data.

Logistic Regression models generally maintain consistent

predictions of >50K across hyperparameters, though confidence in predictions shows minor shifts. For instance, `C=0.1` predicts >50K with a confidence of 50.27% (compared to 52.99% for White), showing a small reduction. Interestingly, with `iter=50`, the model predicts $\leq 50K$ with 51.42% confidence, diverging from the majority. These changes suggest that Logistic Regression models are moderately sensitive to this feature change, though the overall impact on predictions is less pronounced compared to SVM models.

For Random Forest models, predictions for the >50K class remain robust, though confidence levels generally decrease when the race is changed from White to Black. For instance, with `n_estimators=100`, the probability of >50K decreases from 80.12% to 77.55%. Similarly, with `n_estimators=500`, the confidence reduces from 80.38% to 78.36%. Adjusting the maximum tree depth shows consistent behavior: `maxDepth=10` yields a probability of 72.84% for >50K (compared to 76.28% for White), while `maxDepth=20` predicts >50K with 78.61% confidence (compared to 83.22% for White). Further hyperparameter adjustments also reflect a decrease in confidence. For instance, with `minSamplesLeaf=10`, the confidence for the >50K class reduces from 75.62% to 72.04%, while for `minSamplesSplit=10`, the confidence drops from 81.38% to 78.62%. These reductions suggest that Random Forest models are affected by the change in race, though they exhibit greater resilience compared to SVM models.

In summary, the change in race from White to Black leads to subtle reductions in prediction confidence across most models, with Random Forest models showing relatively stable confidence in predicting >50K. SVM models demonstrate higher sensitivity, particularly with $\gamma = 1$, which strengthens its confidence in predicting $\leq 50K$. Logistic Regression models show minor shifts in confidence, with the primary predictions remaining consistent except for a divergence in the `iter=50` configuration. The results highlight the need for careful consideration of demographic features such as race, as they can influence model predictions and reveal potential biases in the underlying training data.

V. ETHICAL REVIEW

A. DATA SOURCE AND INTEGRITY

Throughout this project, two datasets were utilised: the UCI-Adult dataset and the Healthcare Insurance dataset. The UCI-Adult dataset originates from the UC Irvine Machine Learning Repository [1], a well-regarded source known for its well-documented and widely used datasets. In contrast, the Healthcare Insurance dataset was retrieved from Kaggle [2]. While Kaggle serves as a valuable platform for collaboration among data scientists and the machine learning community, it imposes no restrictions on what datasets can be uploaded or maintained. This openness raises potential concerns regarding the credibility and integrity of datasets hosted on the platform.

Beyond the reputability of the sources, it is crucial to evaluate the datasets' credibility and reliability. Regarding credibility, the UCI-Adult dataset is robust, as it comprises data sourced from the Census Bureau and has been widely referenced in academic research. However, the dataset's reliability is limited by its age—it contains census data from 1994, which makes it unsuitable for real-world applications requiring up-to-date models or insights.

In contrast, the Healthcare Insurance dataset demonstrates higher reliability, as it is reportedly updated annually according to its Kaggle page. However, its credibility is less robust. The dataset is compiled through a combination of word-of-mouth data and publicly available hospital data, but lacks sufficient documentation to verify the integrity of its sources or the consistency of its collection methods.

B. DATA BIAS

Data bias is a critical concern when utilising any dataset, as it can compromise the validity and fairness of analysis. The UCI-Adult dataset includes data from over 41 distinct countries, excluding entries from unknown countries. While this initially suggests a diverse dataset, closer inspection reveals significant bias. The majority of entries originate from the United States, meaning the dataset does not accurately represent its intended population. This imbalance introduces a skewed perspective that limits its applicability in contexts requiring global representation.

Additionally, census data is inherently susceptible to biases within the data collection process. Common issues include the under-representation of marginalised groups due to hard-to-access populations, language barriers, and distrust in government institutions. Sampling decisions can further exclude specific demographics, and self-reported income levels are often prone to inaccuracies. These factors result in a dataset that, while representative in broad strokes, reflects systemic biases that may affect its reliability for unbiased analysis.

Turning to the Healthcare Insurance dataset, the author claims that it incorporates publicly accessible hospital data from around the world. However, the dataset falls short of this promise. Entries are categorised under ambiguous region labels consisting of only four values, with no specification of the countries of origin. This lack of clarity raises doubts about whether the dataset accurately represents the intended population. Furthermore, the data collection methods, relying on word-of-mouth accounts and arbitrarily chosen publicly accessible hospital data, lack rigour. The absence of thorough verification of source credibility allows various biases to permeate the dataset, undermining its reliability and utility for meaningful analysis.

C. DATA PRIVACY

Data privacy is a crucial consideration when working with datasets, particularly in contexts where sensitive information could potentially be exposed or misused. In this case, both the UCI-Adult and Healthcare Insurance datasets do not contain any Personally Identifiable Information (PII), such as names, surnames, IDs, or other unique identifiers that could directly link data entries to individuals. This absence

of PII significantly mitigates the risk of privacy breaches and ensures compliance with data protection regulations such as GDPR and HIPAA. However, it is essential to acknowledge that even without explicit PII, datasets may still present re-identification risks if combined with other datasets containing overlapping or complementary information. For instance, attributes like age, income, or region could, in certain scenarios, be triangulated to infer individual identities. Thus, while the datasets themselves respect fundamental data privacy principles, care must still be taken when sharing, merging, or analysing the data to uphold privacy standards and prevent inadvertent disclosure.

D. TRANSPARENCY AND ACCOUNTABILITY

Transparency and Accountability are crucial for the deployment of any machine learning model to ensure proper deployment whilst ensuing the safety of those affected by it. In the context of this project transparency has been maintained with regards to the data collection process as all dataset sources and collection procedures have been specified and criticised, furthermore the training process behind the resultant models has been thoroughly documented and all three models have been assessed and compared against one another to deduce the best overall model. Furthermore, the code responsible for model training is publicly accessible in the designated GitHub repository [37].

Moving on to the aspect of accountability, it falls upon the institute, company or individual who deploys the model to remain accountable for its actions. In cases where large scale models are concerned it would be a disservice to those impacted by the model for its creators to absolve themselves of all accountability. Instead, accountability should be assigned to the company, institute or individual as a whole based on the context to ensure that the proper research, ethical consideration and care is taken in the curation of the model prior to its deployment. In the case of the models presented throughout this project, they are intended solely for research purposes and are not to be used in any other contest with the accountability falling upon the adopters of the models if they choose to do otherwise.

E. FAIRNESS AND EQUITY

The machine learning models presented in this project may exhibit biases due to the datasets used, which could lead to unfair or unequal treatment depending on the context in which they are applied. If the models are used solely for research purposes, their biased or skewed outputs may not directly result in unfair treatment. However, if these models were applied in sensitive real-world scenarios, such as predicting household income for loan eligibility or assessing healthcare insurance, the potential for unfair or unequal treatment could arise. For example, using biased models to determine loan approval or insurance eligibility could disproportionately impact certain individuals or communities, perpetuating existing inequalities. Therefore, the context and use case of these models are crucial in determining whether they might cause harm or unfair treatment.

F. SOCIAL IMPACT

The deployment of the UCI-Adult and Health-Insurance models could have both positive and negative societal consequences, depending on how they are used. On the positive side, these models can provide valuable insights that may help inform policy decisions and improve services. For example, the UCI-Adult model, which predicts income levels based on demographic data, could be used to highlight economic disparities and guide targeted interventions to reduce inequality. Similarly, the Health-Insurance model could help identify gaps in healthcare coverage, potentially enabling better resource allocation and improving access to care. However, there are significant risks associated with deploying these models. Both models rely on historical data which given their current implementations would need to be updated to better reflect the times and ensure proper generalisability. If used in decision-making processes such as, loan approvals, or healthcare eligibility assessments, these aspects could lead to unfair treatment of certain groups, exacerbating societal inequalities. Furthermore, the models might reinforce existing stereotypes or misrepresent minority groups, perpetuating harmful social patterns. As such, while the models hold potential for positive social impact, their deployment must be approached with caution to ensure that they do not inadvertently contribute to discrimination or inequality.

G. ENVIRONMENTAL IMPACT

The models presented in this project are relatively small-scale and can be trained and run on standard home computing hardware, resulting in minimal environmental impact. As long as the models remain at this scale, the environmental cost of data collection, storage, and model training is negligible, making their use justifiable. However, if the models were to be retrained on a larger scale with higher-quality data, this could require more powerful hardware and greater computational resources, potentially increasing their environmental footprint. The use of techniques such as Principal Component Analysis (PCA) and model optimisation methods could help mitigate this impact by reducing the model size and computational load. Additionally, running large-scale models in production, especially if they require constant re-training or real-time processing, could lead to significant energy consumption. Therefore, while the environmental cost of these models at their current scale is minimal, scaling up could have a noticeable impact, and strategies to minimise that impact would be essential. Furthermore, proper justification of their usage would required detailed assessment of their positive impact in relation to their negative environmental effects.

H. ACCESSIBILITY

The web tool is designed with simplicity and ease of use in mind, featuring straightforward input fields that minimise the potential for user error. The model outputs are clearly

labelled and easy to understand, reducing the chance of misinterpretation. Additionally, the user interface (UI) is fully navigable using only the keyboard, making it accessible to individuals with physical impairments. However, the Gradio UI selected for the web tool offers limited support for screen readers, which partially hinders its usability for individuals with visual impairments. While the tool is accessible in many ways, improvements in screen reader compatibility would enhance its overall inclusivity.

I. CONSENT AND OPT-IN

No consent is required for either model variant, as neither model collects any data from end-users. Since both models operate without gathering or tracking user data, there is no need for an opt-in process or consent from users.

J. LONG-TERM EFFECTS

In line with was outlined prior if deployed at scale, the long-term societal impact of the models could be both positive and negative. On the positive side, large-scale deployment could lead to improved decision-making in areas such as economic planning, healthcare access, and social services. For example, the models could help identify patterns in income inequality or healthcare coverage gaps, providing valuable insights for policy reforms and more equitable resource allocation. However, there are potential risks. If the models are not carefully monitored and maintained, it could perpetuate or even worsen existing biases in data, leading to unfair treatment of certain groups, especially in sensitive areas including loan approvals or insurance coverage. Over time, these biases could reinforce societal inequalities, disproportionately affecting marginalised communities. Additionally, widespread reliance on automated systems for decision-making could reduce human oversight and empathy, making decisions appear impersonal or detached. Therefore, while large-scale deployment could bring about positive changes, it is crucial to continuously assess the model's impact and ensure it operates fairly and ethically to avoid unintended consequences.

VI. WEB PORTAL USAGE

This user guide outlines how the Gradio webpage was setup whilst providing the steps required to produce a prediction from the trained models.

A. GRADIO SETUP

The Gradio application consists of two methods one for each dataset which produce the required prediction based on the features provided.

These features have been altered from how they appear in the dataset to facilitate a greater degree of readability, these encapsulate changes such as from **State-gov** to **State Government**. These changes are purely cosmetic and have no affect on the models or the training carried out.

Once the **app.py** file was fully implemented and tested it was uploaded to a Hugging Face repository as seen in Figure 31 which contains the app.py file, the relevant models and the required environment libraries listed in the requirements.txt file. This was done to facilitate hosting. Once the Gradio application was hosted a github page was setup using the html present in the index.html file found in the github repository. This resulted in the webpage seen in Figure 32.

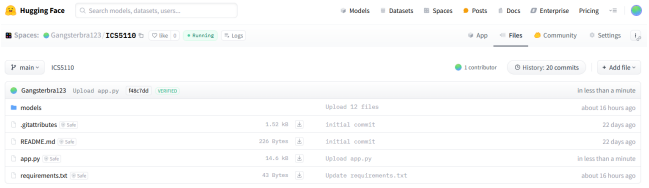

FIGURE 31. Hugging Face Repository

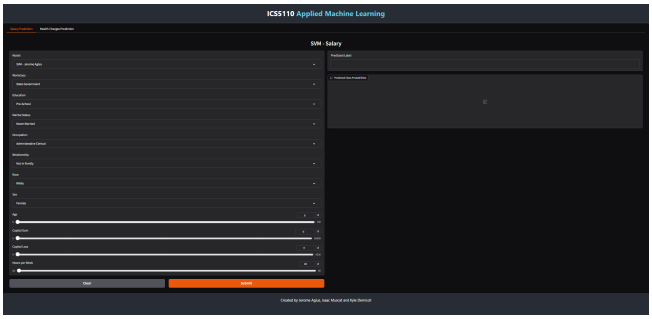

FIGURE 32. Gradio Page

B. USAGE GUIDE

The web portal showcased in Figure 32 can be accessed via the link located in the GitHub repository README as seen in Figure 33.

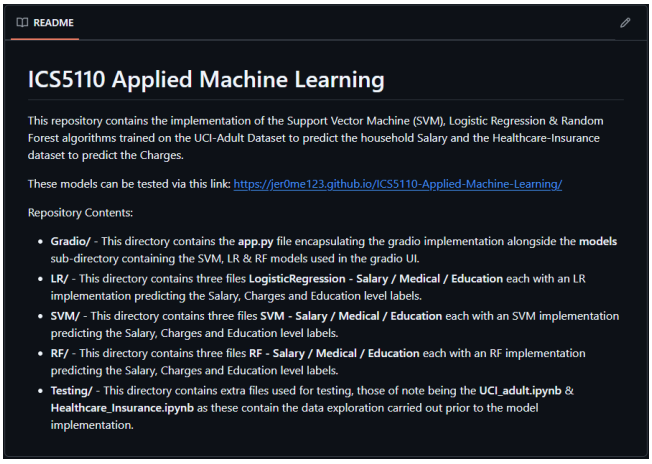

FIGURE 33. GitHub Repository README

This redirects the user to the Gradio page visible in Figure 32 wherein the user has the option to select between the two dataset implementation **Salary Prediction** and **Health Charges Prediction**.

The user is required to enter the input features based on the selected model these being Workclass, Education, Marital Status, Occupation, Relationship, Race, Sex, Age, Capital Gain, Capital Loss and Hours per Week for the **Salary Prediction** model or Age, Sex, BMI, No. of Children, Is Smoker and Region for the **Health Charges Prediction** model. Across both models the feature values are enforced to fall within a particular range or are selectable from a drop-down.

In addition to these input features the user needs to select the model to be used from the model drop-down which contains the following options **SVM - Jerome Agius**, **Logistic Regression - Isaac Muscat** and **Random Forest - Kyle Demicoli**.

Upon submitting the input the user will be provided with the classification label alongside the probability distribution in pie chart form as showcased in Figures 34 and 35.

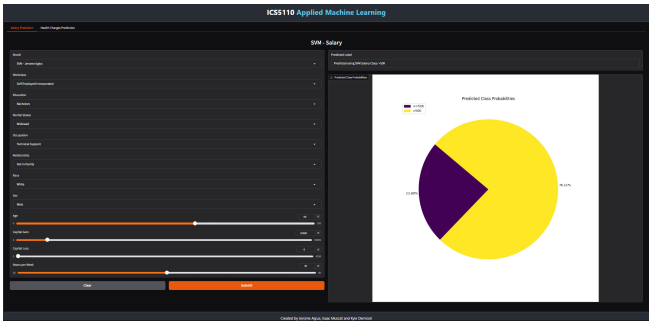

FIGURE 34. Salary Prediction Output

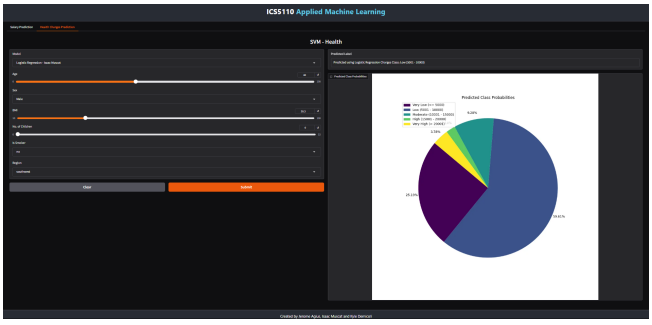

FIGURE 35. Health Charges Prediction

VII. GITHUB

The GitHub repository containing the code relevant to this project can be accessed via this link: GitHub link [37]. Furthermore the GitHub pages webpage can be directly accessed via the following link: https://jer0me123.github.io/ICS5110-Applied-Machine-Learning/

REFERENCES

[1] B. Becker and R. Kohavi, *Adult*, UCI Machine Learning Repository, DOI: https://doi.org/10.24432/C5XW20, 1996.

[2] A. Jangir and willian oliveira, *Healthcare insurance*, 2023. DOI: 10.34740/KAGGLE/DSV/6678394. [Online]. Available: https://www.kaggle.com/dsv/6678394.

[3] IBM, *Support vector machine (svm)*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.ibm.com/topics/support-vector-machine.

[4] Engati, *Kernel method*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.engati.com/glossary/kernel-method.

[5] NB-Data, *One-vs-all vs one-vs-one: Which multi-class classification strategy to choose?* Accessed: 2024-11-09, n.d. [Online]. Available: https://www.nb-data.com/p/one-vs-all-vs-one-vs-one-which-multi.

[6] MathWorks, *Support vector machine*, Accessed: 2024-11-09, n.d. [Online]. Available: https://www.mathworks.com/discovery/support-vector-machine.html.

[7] A. Durán, "The kernel trick," *Towards Data Science*, 2019, Accessed: 2024-11-09. [Online]. Available: https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f.

[8] S.-l. Developers, *Support vector machines*, Accessed: 2024-11-09, 2024. [Online]. Available: https://scikit-learn.org/1.5/modules/svm.html.

[9] GeeksforGeeks, *Gamma parameter in svm*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.geeksforgeeks.org/gamma-parameter-in-svm/.

[10] GeeksforGeeks, *Support vector machine algorithm*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.geeksforgeeks.org/support-vector-machine-algorithm/.

[11] S. Biswas, S. Ghosh, S. Roy, R. Bose, and S. Soni, *A study of stock market prediction through sentiment analysis*, Feb. 2023. DOI: 10.12723/mjs.64.6.

[12] J. Peng, K. Lee, and G. Ingersoll, *An introduction to logistic regression analysis and reporting*, Sep. 2002. DOI: 10.1080/00220670209598786.

[13] S. Sperandei, *Understanding logistic regression analysis*, Feb. 2014. DOI: 10.11613/BM.2014.003.

[14] J. Osborne, *Best practices in logistic regression*, 2015. DOI: 10.4135/9781483399041. [Online]. Available: https://doi.org/10.4135/9781483399041.

[15] Coursera, *Logistic regression: An overview*, Accessed: 08 November 2024. [Online]. Available: https://www.coursera.org/articles/logistic-regression.

[16] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006, ch. 4, pp. 225–230.

[17] S. Buschjäger and K. Morik, "There is no double-descent in random forests," *arXiv preprint arXiv:2111.04409*, 2021.

[18] H. Sahour, V. Gholami, J. Torkman, M. Vazifedan, and S. Saeedi, "Random forest and extreme gradient boosting algorithms for streamflow modeling using vessel features and tree-rings," *Environmental Earth*

Sciences, vol. 80, Nov. 2021. DOI: 10.1007/s12665-021-10054-5.

[19] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[20] M. Prasetiyowati, N. Maulidevi, and K. Surendro, "Feature selection to increase the random forest method performance on high dimensional data," *International Journal of Advances in Intelligent Informatics*, vol. 6, p. 303, Nov. 2020. DOI: 10.26555/ijain.v6i3.471.

[21] S. M. Andrade *et al.*, "Identifying biomarkers for tdcs treatment response in alzheimer's disease patients: A machine learning approach using resting-state eeg classification," *Frontiers in Human Neuroscience*, vol. 17, p. 1 234 168, 2023.

[22] GeeksforGeeks, *Normalization and scaling in machine learning*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.geeksforgeeks.org/normalization-and-scaling/.

[23] C. Team, *Normalization in machine learning*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.codecademy.com/article/normalization.

[24] GeeksforGeeks, *Ml feature scaling (part 2)*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.geeksforgeeks.org/ml-feature-scaling-part-2/.

[25] D. Berrar, *Cross-validation*, Jan. 2018. DOI: 10.1016/B978-0-12-809633-8.20349-X.

[26] S. Yadav and S. Shukla, *Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification*, 2016.

[27] I. Tougui, A. Jilbab, and J. E. Mhamdi, *Impact of the choice of cross-validation techniques on the results of machine learning-based diagnostic applications*, Jul. 2021. DOI: 10.4258/hir.2021.27.3.189.

[28] IBM, *Principal component analysis (pca)*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.ibm.com/topics/principal-component-analysis.

[29] IBM, *Linear discriminant analysis (lda)*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.ibm.com/topics/linear-discriminant-analysis.

[30] IBM, *Dimensionality reduction*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.ibm.com/topics/dimensionality-reduction.

[31] GeeksforGeeks, *Isomap for dimensionality reduction in python*, Accessed: 2024-11-03, 2024. [Online]. Available: https://www.geeksforgeeks.org/isomap-for-dimensionality-reduction-in-python/.

[32] N. D. Analyst, *Understanding feature selection techniques in machine learning*, Accessed: 2024-11-03, 2024. [Online]. Available: https://medium.com/@nirajan_DataAnalyst/understanding-feature-selection-techniques-in-machine-learning-02e2642ef63e.

[33] H. Dalianis, *Evaluation metrics and evaluation*, Cham, 2018. DOI: 10.1007/978-3-319-78503-5_6. [Online]. Available: https://doi.org/10.1007/978-3-319-78503-5_6.

[34] C. Goutte and E. Gaussier, *A probabilistic interpretation of precision, recall and f-score, with implication for evaluation*, Apr. 2005. DOI: 10.1007/978-3-540-31865-1_25.

[35] T. Saito and M. Rehmsmeier, *The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets*, 2015. DOI: 10.1371/journal.pone.0118432.

[36] *Onehotencoder — scikit-learn 1.3.0 documentation*, Accessed: 2024-12-01, 2024. [Online]. Available: https://scikit-learn.org/dev/modules/generated/sklearn.preprocessing.OneHotEncoder.html.

[37] D. K. Agius Jerome Muscat Isaac, *Ics5110 applied machine learning repository*, 2024. [Online]. Available: https://github.com/Jer0me123/ICS5110-Applied-Machine-Learning.