|  |
| --- |
| **DATA SCIENCE PROJECT** |
| **ICR - Identifying Age-Related Conditions** |
| Use Machine Learning to detect conditions with measurements of anonymous characteristics |

**SUBMITTED BY – GROUP 4**

VANSHIKA (SID: 041044413)

SARTAJ SINGH BABRA (SID: 041094831)

NAITIK TALATI (SID: 041084256)

SAIYAM SHAH: (SID: 041080364)

JAY PATEL (SID: 041085500)

**Contents**

[**INTRODUCTION:** 3](#_Toc142162672)

[**DATASET DESCRIPTION:** 3](#_Toc142162673)

[**PROBLEM STATEMENT:** 4](#_Toc142162674)

[**APPROACH AND METHODOLOGY:** 4](#_Toc142162675)

[**DATA PREPROCESSING:** 4](#_Toc142162676)

[**EXPLORATORY DATA ANALYSIS:** 5](#_Toc142162677)

[**FEATURE ENGINEERING:** 9](#_Toc142162678)

[**MODELLING AND PIPELINES:** 10](#_Toc142162679)

**ABSTRACT:**

The purpose of this report is to describe the creation and testing of a predictive model for age-related illness disorders. The Invitro Cell Research, LLC (ICR) competition aims to predict whether a person has one of three age-related medical disorders (Class 1) or none of these conditions (Class 0). The model is based on anonymized measures of health parameters to protect patient privacy.

The goal of this competition is to simplify the process of determining whether someone has these medical disorders, which now includes a time-consuming and intrusive data collection process from patients. By utilizing crucial health variables while keeping patient details confidential, predictive models can assist shorten this procedure.

This competition is significant because it has the potential to help researchers find the association between health characteristic assessments and age-related medical disorders. Furthermore, it promotes developments in the fields of bioinformatics and data science by investigating novel approaches for solving essential problems using different data.

# **INTRODUCTION:**

Age is simply a number, but aging is accompanied by a multitude of health problems. Aging is a risk factor for a wide range of illnesses and consequences, including heart disease, dementia, hearing loss, and arthritis. Research exploring therapies that can help delay and reverse biological aging and prevent serious age-related illnesses is part of the expanding discipline of bioinformatics. Even if the number of samples is modest, data science may be able to contribute to the creation of novel approaches to challenges using diverse data.

This project aims to predict if a person has any of three medical conditions. We will be predicting if the person has one or more of any of the three medical conditions (Class 1) or none of the three medical conditions (Class 0). In this report, we will create a model trained on measurements of health characteristics.

Information gathering from patients is a time-consuming and intrusive process needed to evaluate whether someone has various medical disorders. By gathering crucial variables relevant to the conditions, then encoding these features, predictive models enable us to speed up this procedure while maintaining patient confidentiality.

# **DATASET DESCRIPTION:**

The dataset consists of three age-related diseases and more than fifty anonymized health characteristics. The variable "Class" has two potential values, with the data being a binary classification problem: 1 denotes the presence of one or more age-related conditions, while 0 denotes the absence of any of the three conditions in the subject's medical history.

The following files and fields are included in the dataset:

**train.csv:** The training set, includes fifty-six anonymized health characteristics and unique identifiers for each observation. The target variable is indicated by the 'Class' field.

**test.csv:** The test set for which the likelihood of participants belonging to each class must be predicted.

**greeks.csv:** Supplemental metadata accessible solely for the training set, including the type of age-related condition, three experimental characteristics, and the data collection date.

# **PROBLEM STATEMENT:**

The competition requires competitors to create a prediction model based on training data to assess whether a subject in the test set belongs to Class 1 or Class 0. Participants are required to employ data science techniques to develop an accurate and reliable model for predicting age-related medical problems. Current models, such as XGBoost and random forest, have not performed well in predicting medical disorders. As a result, the primary goal of this challenge is to improve on existing methods and construct models that can consistently make the right predictions between diverse scenarios, especially when dealing with significant problems that affect people's lives.

# **APPROACH AND METHODOLOGY:**

To create their predictive models, we will experiment with various data science methodologies. Common strategies include model selection, hyperparameter tuning, ensemble methods, and feature engineering. It is advised to take the following actions when creating the prediction model:

* **DATA PREPROCESSING**: Handle missing data, transform categorical variables if necessary, and normalize, or scale numerical features during data preprocessing.
* **EXPLORATORY DATA ANALYSIS:** Gain understanding of the dataset, examine feature distributions, and locate connections between the features and the desired variable using exploratory data analysis (EDA).
* **FEATURE ENGINEERING:** Feature engineering is the process of removing pertinent aspects from data that may help in prediction.
* **MODEL SELECTION:** Experiment with various classification algorithms such as logistic regression, support vector machines, decision trees, gradient boosting, and so on.
* **ENSEMBLE METHOD:** Consider employing ensemble approaches like bagging and boosting to improve model robustness and accuracy.
* **HYPERPARAMETER TUNING**: Optimise the model hyperparameters to attain the optimum performance.
* **MODEL EVALUATION:** To examine the model's performance, use relevant evaluation metrics like accuracy, precision, recall, and F1-score.

# **DATA PREPROCESSING:**

The dataset contains a total of 617 rows and 58 columns. Each row represents an observation, and each column corresponds to a specific health characteristic or attribute. The target variable, 'Class,' is a binary variable indicating whether a person has any of the three age-related medical conditions (Class 1) or none of these conditions (Class 0).To perform data manipulation and visualization, we imported the necessary libraries, including pandas, numpy, seaborn, and matplotlib.

# **EXPLORATORY DATA ANALYSIS:**

* **General information and datatypes**: We loaded the dataset using the pandas 'read\_csv' function and displayed the first five records using 'df. head()'. The output showed a snapshot of the dataset, including the 'Id' column and the first few rows with their respective health characteristic measurements. To gain a better understanding of the dataset, we examined the data types of each feature using 'df.dtypes. The output revealed that most features are represented as float64 (numeric), and one feature ('Class') is represented as int64 (integer). Additionally, the 'Id' column is represented as an object (string) data type.
* **Dataset summary**: To obtain summary statistics for the numerical features, we used
* 'df. describe()'. The summary provides information such as count, mean, standard deviation, minimum, 25th percentile, median (50th percentile), 75th percentile, and maximum values for each numerical attribute. This summary helps in understanding the distribution and range of health characteristic measurements. It's essential to note that the summary statistics are provided for numerical features, and not for the 'Id' column, which contains unique identifiers for each observation. Additionally, the 'Class' column is not included in the summary statistics since it is a categorical target variable.
* **Count of unique values:** To explore the dataset further, we counted the occurrences of each unique value in each column using the 'value counts()' function. This step allows us to understand the cardinality of categorical features and identify any data imbalances in the target variable ('Class'). By examining the count of unique values for each feature, we can gain insights into the data distribution, identify missing values, and check for any potential data anomalies.
* **Data Distribution visualization:** We drew histograms for each feature to better understand the distribution of health characteristic readings in the dataset. The histogram plots reveal information about how frequently and widely different numerical properties occur**.**

Additionally, using the Seaborn library, we built subplots to contrast the distribution of each numerical characteristic depending on the two classes (Class 0 and Class 1) to better study the link between numerical features and the goal variable ('Class').

The distribution of the numerical features (such as "AB," "AF," "AH," etc.) for the two classes is compared in the subplots. According to the value of the 'Class' variable, the bars are colored using the 'hue' parameter, and the 'kde=True' option adds kernel density estimation curves to the histograms for a smoother depiction.

A group of graphs showing different sizes and colors

Description automatically generated with medium confidence

* **Boxplot of numerical feature:** We also made a boxplot to show the distribution of numerical features in addition to the histograms. Boxplots show the median, quartiles, and any probable outliers in a summary of the data distribution. With the x-axis representing the features and the y-axis representing the raw values, the boxplot illustrates the distribution of each numerical feature. The horizontal line inside each box denotes the median, while the boxes stand in for the interquartile range (IQR). Any points outside of the whiskers are regarded as outliers. Whiskers are defined as the minimum and maximum values that are 1.5 times the IQR apart.

A graph showing a number of objects

Description automatically generated with medium confidence

* **Class Distribution visualization**: Finally, to better understand the balance between Class 0 and Class 1, we visualized the distribution of the target variable ('Class'). The plot shows a histogram of the 'Class' variable, with the two classes (Class 0 and Class 1) shown on the x-axis and the frequency of occurrence on the y-axis. We can examine the target variable's distribution and evaluate any class imbalances thanks to this visualization.

A graph with orange lines

Description automatically generated

* **Correlation Matrix:** We created a correlation matrix to better comprehend the connections between the numerical features. The pairwise correlation coefficients between numerical features are represented by the correlation matrix. A strong positive correlation is indicated by a correlation value that is close to 1, and a strong negative correlation is indicated by a value that is close to -1. A result that approaches 0 denotes a weak correlation between the features. We can find potential multicollinearity between characteristics and determine which features might have the biggest influence on predicting the target variable by looking at the correlation matrix. Please be sure to properly title and label the plots in your report, and feel free to change the explanations to fit your unique study and situation.

The visualization part investigates the connections between numerical features and the target variable ('Class'):

* **Heatmap of Correlation Matrix:** Uses a color-coded heatmap of correlation coefficients to visualize linear correlations between numerical features. The darker the color, the stronger the association.

A close-up of a screen

Description automatically generated

* **Correlation with Target Variable:** This function identifies numerical parameters that are highly connected with the target variable. A heatmap highlights the attributes that are most connected with the objective, with a concentration on coefficients larger than or equal to 0.15.

A blue screen with white text

Description automatically generated

* **Features with the Highest Correlation**: Uses a correlation matrix heatmap to examine the highest relationships among a subset of numerical features.

A graph of a number of purple squares

Description automatically generated with medium confidence

* A screenshot of a graph

  Description automatically generated**Scatter Plots of Highly Correlated Features:** Uses scatter plots with regression lines to illustrate linear correlations between pairs of highly correlated features.

These visualizations aid in the comprehension of feature relationships, the identification of potential multicollinearity, and the selection of informative characteristics for the construction of an appropriate prediction model.

# **FEATURE ENGINEERING:**

Feature engineering involves the following steps:

* **Checking for Duplicate Values:** The duplicated () method is used to examine the original dataset for duplicate rows. If duplicates are discovered, they are removed with the drop\_duplicates() method.
* **Checking for missing values:** The dataset is examined for missing values using the isnull() method, which returns a Data Frame of Boolean values indicating missing values. The sum() method is then applied to each column to count the number of missing values.
* **Replacing Missing Values:** Missing Values are replaced with the mean of each column using the fillna() function using df.mean() as a parameter. This method is based on the assumption that missing values can be estimated by the mean value of the corresponding column.
* **Convert Categorical Data:** One-hot encoding is used to convert categorical data to numerical data. The pd.get\_dummies() method is used on the columns "Id" and "EJ" to generate binary columns for each unique value in these columns, thus converting them to numerical representations.
* **Rescaling Features:** Using Min-Max scaling, the data is rescaled to a range of 0 to 1. This ensures that all features have the same scale and avoids specific characteristics from dominating the model because of their bigger magnitudes.
* **Setting the Target Variable:** To build the 'X' (input) and 'Y' (target) data frames, the target variable "Class" is separated from the input features.
* **Handling Class Imbalance:** The target variable 'Y' is examined for class imbalance, as indicated by the number of 0s and 1s in its value counts. To solve the imbalance, the imbalanced-learn library's RandomOverSampler is used to generate synthetic samples of the minority class (Class 1) equal to the number of samples in the majority class (Class 0). This contributes to the reduction of class disparities.

# **MODELLING AND PIPELINES:**

To predict the target variable, below regression models were created, and hyperparameter and ensembling techniques were followed to improve the performance.

**Logistic Regression:** Logistic Regression is a statistical regression model used for binary classification problems. It predicts the probability that an instance belongs to a particular class using a logistic function. The model fits a linear decision boundary to the input features and then applies the logistic function to map the predictions to probabilities. It is widely used due to its simplicity, interpretability, and effectiveness in many scenarios.

**Hyperparameter tuning Logistic Regression:** Hyperparameter tuning for Logistic Regression involves systematically searching for the best combination of hyperparameters that optimize the model's performance. Hyperparameters are settings that are not learned from the data but are set before the training process, such as the regularization strength (C) or the type of penalty (L1 or L2). Techniques like grid search or random search are used to find the optimal hyperparameters that yield the best performance on the validation dataset.

**Gradient Boosting (Ensembling - Boosting):** Gradient Boosting is an ensemble learning method that combines multiple weak learners (usually decision trees) to create a strong predictive model. It builds trees sequentially, with each tree trying to correct the errors made by the previous ones. It assigns higher weights to the instances that were misclassified in the previous tree and combines their predictions to make the final prediction. Gradient Boosting is known for its high predictive power and ability to handle complex datasets.

**Hyperparameter tuning Gradient Boosting (Ensembling with HP tuning):** Similar to Hyperparameter tuning for Logistic Regression, here we perform hyperparameter tuning for the Gradient Boosting model. We search for the optimal combination of hyperparameters, such as the number of trees (n\_estimators), the learning rate (learning\_rate), and the maximum depth of the trees (max\_depth), to achieve the best model performance.

**Support Vector Machine:** Support Vector Machine (SVM) is a powerful supervised learning algorithm used for both classification and regression tasks. In classification, SVM finds the optimal hyperplane that best separates data points belonging to different classes. It tries to maximize the margin (distance between the hyperplane and the nearest data points) to achieve better generalization.

**Hyperparameter tuning Support Vector Machine:** Hyperparameter tuning for Support Vector Machine involves searching for the best combination of hyperparameters, such as the kernel type (linear, polynomial, radial basis function, etc.), the regularization parameter (C), and the kernel-specific parameters. Tuning these hyperparameters helps optimize the SVM model's performance on the validation dataset.

**KNN Classifier:** KNN (K-Nearest Neighbors) is a non-parametric, lazy learning algorithm used for classification tasks. Given a new data point, KNN searches for the K nearest neighbors based on a distance metric (e.g., Euclidean distance) and assigns the class label by a majority vote among the K neighbors.

**Hyperparameter tuning KNN Model:** Hyperparameter tuning for the KNN model involves finding the optimal value of K (the number of neighbors to consider) and selecting the appropriate distance metric. The performance of the KNN classifier can vary significantly based on the choice of K and the distance metric.

**Random Forest (Ensembling - Bagging):** Random Forest is an ensemble learning method that creates multiple decision trees and combines their predictions to make the final prediction. Each tree is built on a random subset of the data and features, which reduces the risk of overfitting and increases the model's robustness. The final prediction is obtained by averaging (for regression) or voting (for classification) the individual tree predictions.

Finally, we ran the models and the accuracies came out to be as below,

A screenshot of a calculator

Description automatically generated

Logistic Regression and the KNN classifier model is considered the best-performing model after the random forest regression. Hyperparameter adjustment appears to have greatly improved the accuracy of both the Logistic Regression and KNN models. They currently have an accuracy of nearly 0.99, which is nearly flawless. This means that the hyperparameter tuning process has discovered the best collection of hyperparameters for these models, allowing them to perform extraordinarily well on the supplied dataset.