# Final Project Report

# of Internship Program

# on

# “PREDICT BLOOD DONATION”

## MEDTOUREASY,NEW DELHI

## SUBMITTED BY

**Nilesh Bairagi**

## on
## 28th June 2023

# ACKNOWLEDGEMENT

The internship opportunity that I had with MedTourEasy was a great change for learning and understanding the intricacies in the field of Data Analytics in Data Science; and also, for personal as well as professional development. I am very obliged for having a chance to interact with so many professionals who guided me throughout the internship project and made it a great learning curve for me.

Firstly, I express my deepest gratitude and special thanks to the Training Head of MedTourEasy, Mr.Ankit Hasija who gave me an opportunity to carry out my internship at their esteemed organization. Also, I express my thanks to him for making me understand the details of the Data Analytics profile and training me in the same so that I can carry out the project properly and with maximum clientsatisfaction and also for spearing his valuable time in spite of his busy schedule.

I would also like to thank the team of MedTourEasy who made the working environment productive and very conducive.

# ABSTRACT

Forecasting blood supply is a serious and recurrent problems that a blood collection manager deals with. The title of the project is "**Predict Blood Donation**". A blood transfusion is a way of adding blood to the body after an injury or illness. Blood transfusion saves lives - from replacing lost blood duringmajor surgery or a serious injury to treating various illnesses and blood disorders.Ensuring that there's enough blood in supply whenever needed is a serious challenge for the health professionals. The demand for blood fluctuates throughout the year. As one prominent example, blood donations slow down during busy holiday seasons. An accurate forecast for the future supply of bloodallows for an appropriate action to be taken ahead of time and therefore saving more lives.

It is generally seen that blood supply decreases during winters and peoples who are on travel or errands are very less likely to undergo blood donations. The project undergoes a development pipeline automated by TPOT auto-ML librarywhich is discussed in detail further in this report.

# TABLE OF CONTENT

# 1. INTRODUCTION

## 1.1 ABOUT THE COMPANY

MedTourEasy's mission is to provide access to quality healthcare for everyone, regardless of location, time frame, or budget. Patients can connect with internationally-accredited clinics and hospitals.

MedTourEasy, an online medical tourism marketplace, provides you the informational resources needed to evaluate your global options. It helps you findthe right healthcare solution based on your specific health needs, affordable care,while meeting the quality standards that you expect to have in healthcare.

MedTourEasy, a global healthcare company, provides you the informational resources needed to evaluate your global options. It helps you find the right healthcare solution based on specific health needs, affordable care while meeting the quality standards that you expect to have in healthcare.

MedTourEasy improves access to healthcare for people everywhere. It is an easy to use platform and service that helps patients to get medical second opinions andto schedule affordable, high-quality medical treatment abroad.

MedTourEasy commitment to quality and transparency in healthcare is core to our mission. At MedTourEasy, we integrate the same three factors that physicians themselves agree are most important when selecting or referring a healthcare provider (patient satisfaction, experience match, and the quality of the hospital where a physician provides care.

MedTourEasy aspires to be the leader in making information on physicians and hospitals more accessible and transparent. The purpose is to give people the confidence to make the right healthcare decisions.

## 1.2 ABOUT THE PROJECT

The title of the project is “Predict Blood Donations”. Forecasting blood supply isa serious and recurrent problems that a blood collection manager deals with. A blood transfusion is a way of adding blood to the body after an injury or illness. Blood transfusion saves lives - from replacing lost blood during major surgery ora serious injury to treating various illnesses and blood disorders. Ensuring that there's enough blood in supply whenever needed is a serious challenge for the health professionals. The demand for blood fluctuates throughout the year. As one prominent example, blood donations slow down during busy holiday seasons. An accurate forecast for the future supply of blood allows for an appropriate action to be taken ahead of time and therefore saving more lives.

The project involves working on blood transfusion dataset. It includes thefollowing

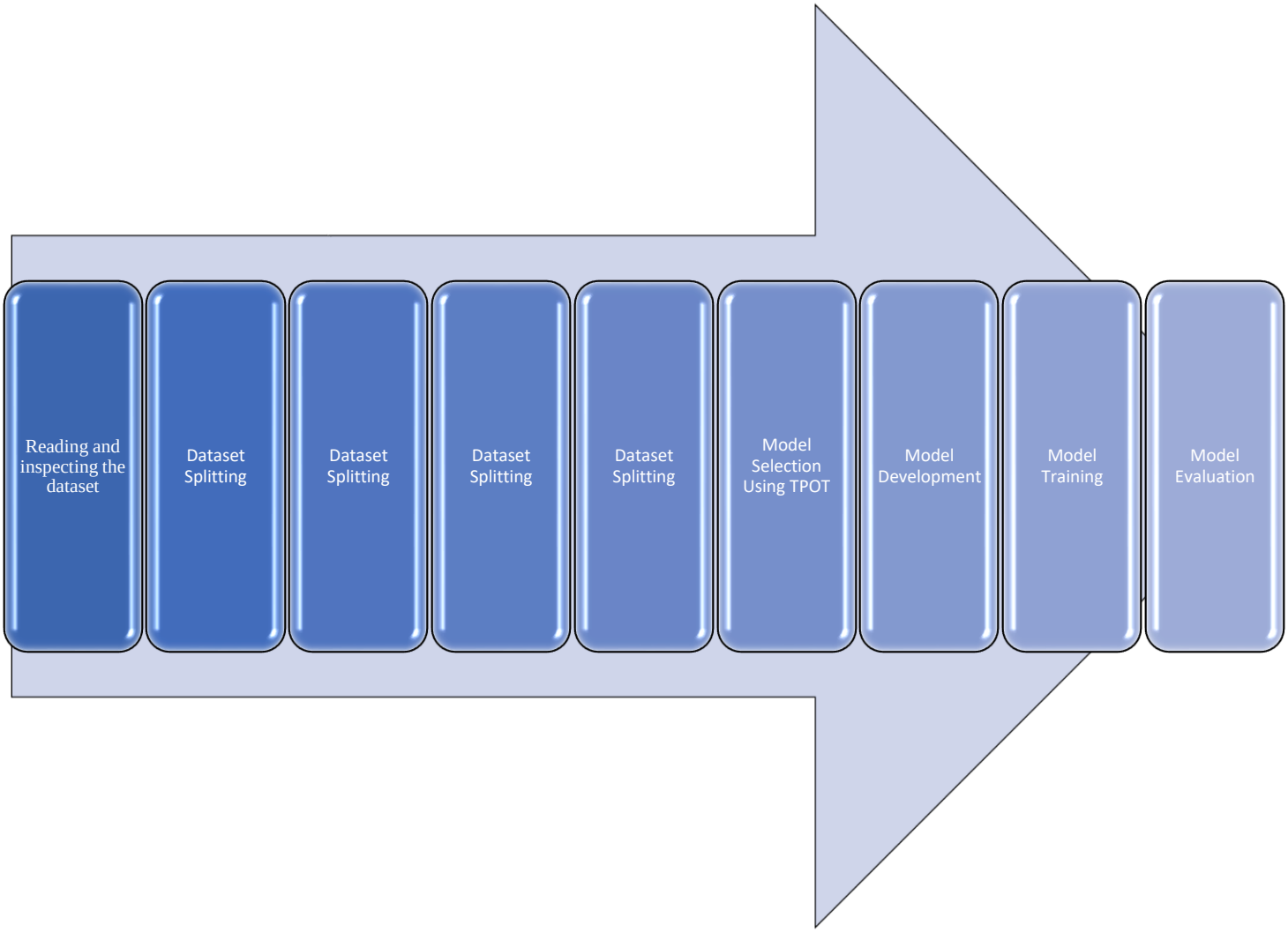
**steps:-**

- Loading the blood transfusion dataset
- Inspecting the data frame
- Specifying features and target columns
- Splitting dataset into training and testing dataset
- Selecting model using TPOT
- Model Building
- Model Training
- Model Evaluation

# 2. METHODOLOGY

## 2.1 FLOW OF THE PROJECT

The Project flow can be understood by below flowchart



## 2.2 LANGUAGES AND PLATFORM USED

### 2.2.1 LANGUAGE: PYTHON

Python is a high-level programming language known for its simplicity and readability. It was created by Guido van Rossum and released in 1991. Python is widely used in various domains, including web development, data analysis, scientific computing, artificial intelligence, and automation.

Here are some key features and characteristics of Python:

- Readability: Python emphasizes code readability and uses a clean syntax that makes it easier to understand and write code. It uses indentation instead of brackets to define code blocks.
- Interpreted Language: Python is an interpreted language, meaning that code is executed line by line by the Python interpreter without the need for compilation. This allows for rapid development and experimentation.
- Object-Oriented: Python supports object-oriented programming (OOP) concepts, such as classes, objects, and inheritance. It provides a straightforward way to structure and organize code.
- Large Standard Library: Python comes with a large standard library that provides a wide range of modules and functions for common tasks. These modules include functionalities for file I/O, network communication, regular expressions, and much more.
- Third-Party Libraries: Python has a rich ecosystem of third-party libraries and frameworks that extend its capabilities. For example, NumPy and Pandas are popular libraries for scientific computing and data analysis, while Django and Flask are widely used for web development.

- Cross-Platform Compatibility: Python is a cross-platform language, meaning that code written in Python can run on various operating systems, including Windows, macOS, and Linux.

- Dynamically Typed: Python is dynamically typed, which means that variable types are determined at runtime. You don't need to explicitly declare variable types, making it flexible and allowing for faster development.

- Beginner-Friendly: Python is often recommended as a first programming language due to its simplicity and readability. It has an extensive community and numerous learning resources available, making it easy for beginners to get started.

- Scalability: Python can be used for both small and large-scale projects. While it may not be as performant as lower-level languages, it offers scalability and ease of development, which is valuable for rapid prototyping and building complex applications.

- Open-Source: Python is an open-source language, meaning that its source code is freely available and can be modified and distributed by anyone. This fosters collaboration and continuous improvement within the Python community.

## 2.2.2 MACHINE LEARNING:

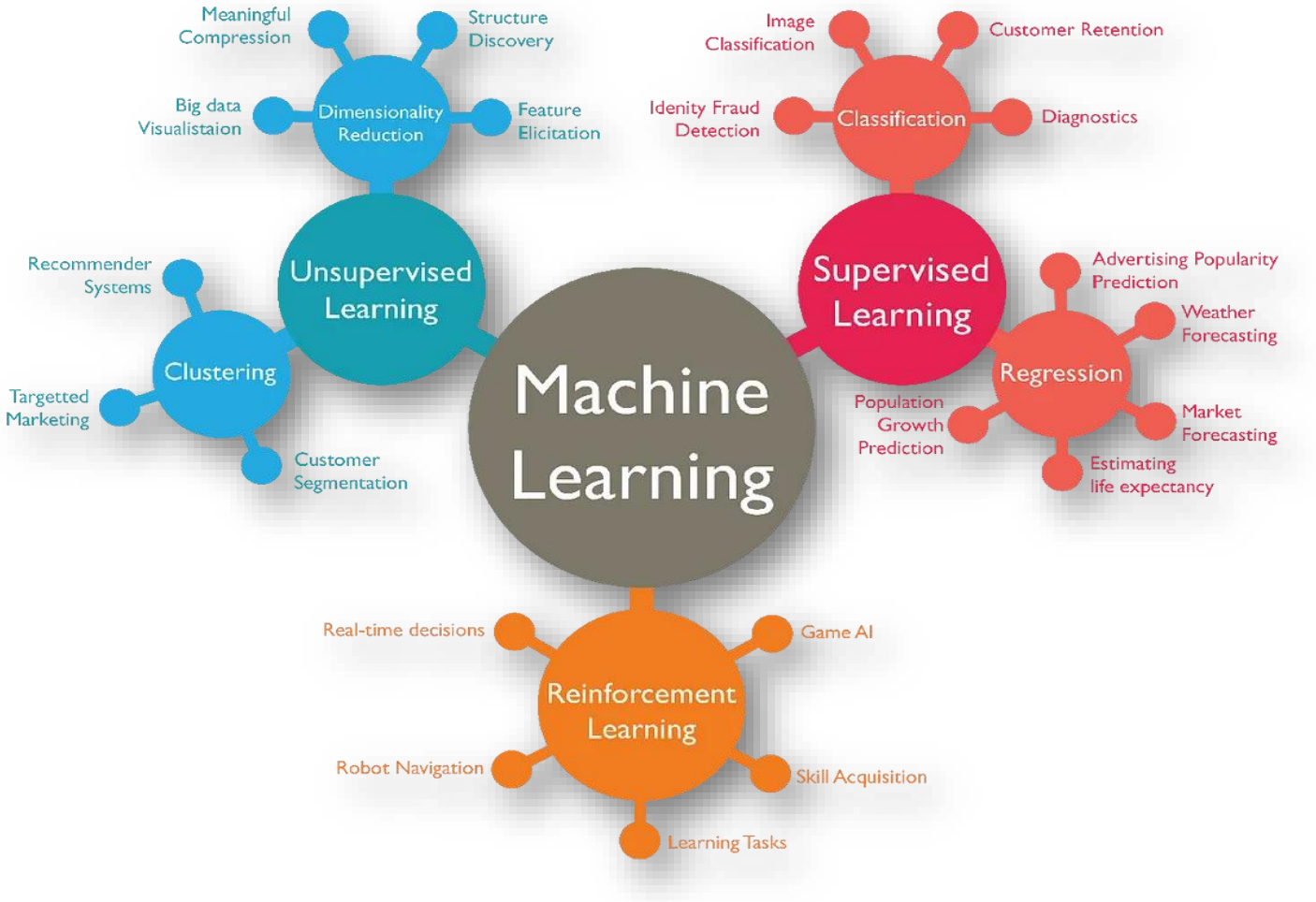



Fig. Machine Learning

Machine learning is a subfield of artificial intelligence (AI) that focuses on the development of algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programmed. It involves the study of algorithms and statistical models that allow computers to analyze and interpret complex data, identify patterns, and make predictions or take actions based on that analysis.

Here are some key concepts and components related to machine learning:

- Data: Machine learning algorithms require a significant amount of data to learn from. This data can be structured or unstructured and can come from various sources such as sensors, databases, or the internet.

- Training Data: To build a machine learning model, a labeled dataset is used for training. This dataset consists of input data points (features) and their corresponding known outputs or labels. The model learns from this data to make predictions or classifications.

- Algorithms: Machine learning algorithms are mathematical models that learn patterns and relationships from the training data. There are several types of machine learning algorithms, including supervised learning, unsupervised learning, and reinforcement learning.

- Supervised Learning: In supervised learning, the training data includes input data and corresponding labels or outputs. The algorithm learns to map the inputs to the correct outputs by generalizing from the provided examples. Examples of supervised learning algorithms include decision trees, support vector machines (SVM), and neural networks.

- Unsupervised Learning: Unsupervised learning deals with unlabeled data, where the algorithm learns patterns or structures in the data without explicit labels or outputs. Clustering and dimensionality reduction are common unsupervised learning techniques. Clustering algorithms group similar data points together, while dimensionality reduction techniques reduce the number of features while retaining important information.

- Reinforcement Learning: Reinforcement learning involves an agent that learns to interact with an environment and takes actions to maximize a reward signal. The agent receives feedback in the form of rewards or penalties based on its actions and learns to optimize its decision-making process. Reinforcement learning is often used in tasks such as robotics, gaming, and autonomous systems.

- Model Evaluation: To assess the performance of a machine learning model, it is necessary to evaluate its predictions or classifications on unseen data. Common

evaluation metrics include accuracy, precision, recall, F1 score, and area under the curve (AUC) for classification problems, as well as mean squared error (MSE) or root mean squared error (RMSE) for regression problems.

- Feature Engineering: Feature engineering involves selecting, transforming, and creating features from the raw data to improve the performance of machine learning models. This process requires domain knowledge and expertise to extract meaningful information and reduce noise from the data.

- Model Deployment: Once a machine learning model is trained and evaluated, it can be deployed to make predictions or decisions on new, unseen data. Deployment can involve integrating the model into production systems, creating APIs for accessing the model, or deploying it on cloud platforms for scalability.

- Deep Learning: Deep learning is a subset of machine learning that focuses on artificial neural networks with multiple layers. Deep learning models, such as deep neural networks and convolutional neural networks (CNNs), have achieved remarkable success in various domains, including image recognition, natural language processing, and speech recognition.

- Machine learning has a wide range of applications across different industries, including finance, healthcare, marketing, computer vision, natural language processing, recommendation systems, and autonomous vehicles. It continues to advance rapidly, with ongoing research and development to improve algorithms, models, and techniques for handling complex and large-scale data.

## 2.2.3 PLATFORM USED: JUPYTER NOTEBOOK


Fig : Jupyter Notebook Logo

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and explanatory text. It provides an interactive computational environment that supports various programming languages, including Python, R, Julia, and more. Jupyter Notebook is widely used for data analysis, scientific computing, machine learning, and education.

Here are some key features and information about Jupyter Notebook:

- Notebooks: Jupyter Notebook documents are called "notebooks" and have a .ipynb file extension. Notebooks consist of cells that can contain code, Markdown text, equations, or raw text. The notebook interface allows you to edit and run code cells interactively.

- Code Execution: Jupyter Notebook allows you to write and execute code in cells. You can run a cell by clicking the "Run" button in the toolbar or by using keyboard shortcuts. Each code cell can be executed independently, allowing you to modify and rerun

specific parts of your code.

- Interactive Outputs: Jupyter Notebook provides interactive outputs that can be displayed within the notebook itself. This includes text output, tables, plots, interactive visualizations, and more. The ability to visualize data and view results within the notebook makes it a powerful tool for data analysis and exploration.

- Rich Text Formatting: Jupyter Notebook supports Markdown, a lightweight markup language, for formatting text cells. You can use Markdown to add headings, lists, links, images, and even mathematical equations to your notebook. This allows you to create comprehensive and well-documented notebooks.

- Collaboration and Sharing: Jupyter Notebook facilitates collaboration and sharing among users. Notebooks can be easily shared with others, allowing them to view and interact with the code and outputs. Additionally, Jupyter supports version control systems like Git, enabling collaboration on notebook development.

- Integration with Libraries and Tools: Jupyter Notebook integrates seamlessly with a wide range of libraries and tools used in data science and scientific computing. It supports popular Python libraries like NumPy, Pandas, Matplotlib, and SciPy, making it a versatile environment for data analysis and modeling.

- Jupyter Extensions: Jupyter Notebook offers a collection of extensions that enhance its functionality. These extensions provide additional features like code formatting, code linting, table of contents, spell-checking, and more. Extensions can be installed and managed easily using the Jupyter Notebook interface.

- JupyterLab: JupyterLab is the next-generation user interface for Jupyter Notebook. It provides a more flexible and powerful environment for interactive computing. Jupyter Lab allows you to arrange multiple notebooks, code editors, terminals, and interactive

outputs in a customizable and tab-based layout.

- Deployment and Cloud Services: Jupyter Notebook can be run locally on your computer or deployed on cloud services like JupyterHub, Google Colaboratory (Colab), or Microsoft Azure Notebooks. Cloud-based Jupyter services enable easy access and collaboration on notebooks without requiring local installations.

- Education and Documentation: Jupyter Notebook is widely used in educational settings to teach programming, data science, and scientific concepts. It provides an intuitive and interactive environment for students to learn and experiment. The Jupyter community maintains comprehensive documentation and tutorials to help users get started and explore advanced features.

- Jupyter Notebook has gained popularity due to its versatility, interactive nature, and ability to combine code, text, and visualizations in a single document. It has become an essential tool for data scientists, researchers, educators, and anyone working with data analysis and computation.

### 2.2.4 Libraries Used:

### A. Pandas:

The pandas library is a popular open-source data manipulation and analysis library for the Python programming language. It provides high-performance, easy-to-use data structures and data analysis tools that make working with structured data efficient and convenient. pandas is widely used in data science, finance, economics, social sciences, and other fields. Here are some key features and information about the pandas library:

- DataFrame: The pandas DataFrame is a two-dimensional labeled data structure that resembles a table or spreadsheet. It consists of columns, each of which can have a different data type (e.g., numeric, string, datetime). DataFrames allow for efficient indexing, slicing, and manipulation of data. They can be created from various data sources such as CSV files, Excel spreadsheets, databases, or by converting other data

structures like NumPy arrays or Python dictionaries.

- Series: A pandas Series is a one-dimensional labeled array that can hold any data type. It is similar to a column in a DataFrame or a single column of a spreadsheet. Series provide powerful indexing capabilities and are often used as building blocks for DataFrames. They can be created from lists, arrays, or dictionaries.

- Data Cleaning and Preparation: pandas provides a rich set of functions for cleaning, transforming, and preparing data. These functions include handling missing data, removing duplicates, reshaping data, applying functions across rows or columns, and more. pandas also offers powerful string manipulation and regular expression capabilities for working with text data.

- Data Manipulation and Analysis: pandas provides a wide range of functionalities for data manipulation and analysis. You can perform operations such as filtering, sorting, grouping, merging, pivoting, and aggregating data easily using pandas. It also supports advanced operations like time series analysis, handling categorical data, and handling hierarchical data structures.

- Indexing and Selection: pandas offers flexible indexing and selection methods to access and manipulate data within DataFrames and Series. It allows for label-based indexing, integer-based indexing, boolean indexing, and more. This makes it easy to select, filter, and modify specific rows or columns based on various conditions.

- Data Visualization: pandas integrates well with other Python libraries like Matplotlib and Seaborn, enabling you to create insightful data visualizations directly from DataFrames and Series. You can generate plots, histograms, scatter plots, bar charts, and more to visually analyze and present your data.

- Integration with Other Tools: pandas seamlessly integrates with other data analysis and

machine learning libraries in the Python ecosystem, such as NumPy, SciPy, scikit-learn, and TensorFlow. This allows for smooth data interchange and interoperability with other tools and frameworks.

- Performance and Efficiency: pandas is designed to handle large datasets efficiently. It leverages the power of underlying libraries, such as NumPy, to perform fast and vectorized operations on data. Additionally, pandas provides options for optimizing memory usage and handling large datasets through features like chunked reading and writing of data.

- Documentation and Community Support: pandas has extensive documentation that includes detailed explanations, tutorials, and examples to help users learn and utilize its functionalities effectively. It also has a large and active community that provides support, shares knowledge, and contributes to the development and improvement of the library.

- Open-Source and Active Development: pandas is an open-source project, which means its source code is freely available and can be modified and distributed by anyone. It has a dedicated development team that regularly releases updates and improvements to the library, ensuring its continued growth and relevance in the data analysis ecosystem.

- Overall, the pandas library is a versatile and powerful tool for data manipulation, analysis, and preparation in Python. Its intuitive syntax and comprehensive functionalities make it a go-to choice for working with structured data in various domains.

## B. Numpy:

The NumPy library is a fundamental open-source numerical computing library for the Python programming language. It provides a powerful array object and a collection of mathematical functions that allow efficient manipulation and computation on large multidimensional arrays and matrices. NumPy forms the foundation for many other

scientific and data analysis libraries in the Python ecosystem.

Here are some key features and information about the NumPy library:

- ndarray: The ndarray (n-dimensional array) is the central data structure in NumPy. It is a homogeneous collection of elements of the same data type, arranged in a grid-like structure of dimensions specified by a shape. NumPy arrays can have any number of dimensions and are more efficient for storing and manipulating large datasets compared to Python lists.

- Vectorized Operations: NumPy enables vectorized operations, which allow you to perform mathematical computations on entire arrays without writing explicit loops. This leads to faster and more concise code execution, especially when dealing with large datasets.

- Mathematical Functions: NumPy provides a wide range of mathematical functions that operate element-wise on arrays. These functions include basic arithmetic operations, trigonometric functions, exponential and logarithmic functions, statistical functions, and more. NumPy functions are optimized for performance and provide efficient implementations of common mathematical operations.

- Broadcasting: Broadcasting is a powerful feature in NumPy that allows arrays of different shapes to be used together in operations. It automatically adjusts the dimensions of arrays to perform element-wise operations efficiently, even when the shapes are not identical.

- Array Manipulation: NumPy offers a variety of functions for manipulating arrays. You can reshape, transpose, concatenate, split, and stack arrays, as well as extract specific elements or subsets of arrays. NumPy also provides functions for sorting, searching, and manipulating array elements.

- Random Number Generation: NumPy includes a submodule called numpy.random for generating random numbers. It provides functions for sampling from various probability distributions, generating random arrays, and shuffling arrays. Random number generation is crucial for simulations, statistical analysis, and machine learning applications.

- Linear Algebra Operations: NumPy provides a comprehensive suite of linear algebra functions. You can perform matrix multiplication, matrix factorization, eigenvalue and eigenvector calculations, and solve linear systems of equations. These functions are particularly useful in scientific computing and numerical simulations.

- Integration with Other Libraries: NumPy seamlessly integrates with other scientific and data analysis libraries in the Python ecosystem, such as SciPy, pandas, Matplotlib, and scikit-learn. This allows for smooth data interchange and interoperability, enabling you to build powerful analytical workflows.

- Performance and Efficiency: NumPy is implemented in C and provides highly efficient and optimized functions for numerical computations. It leverages low-level optimizations and hardware acceleration to achieve fast execution speeds. NumPy arrays are stored in a contiguous block of memory, which allows for efficient memory access and enables NumPy to handle large datasets efficiently.

- Documentation and Community Support: NumPy has extensive documentation with detailed explanations, tutorials, and examples to help users learn and utilize its functionalities effectively. It also has a large and active community that provides support, shares knowledge, and contributes to the development and improvement of the library.

- NumPy is widely used in scientific computing, data analysis, machine learning, and many other domains. Its array-based computing capabilities, efficient operations, and integration with other libraries make it an essential tool for numerical computations.

## C. Matplotlib:

Matplotlib is a widely used open-source data visualization library for the Python programming language. It provides a comprehensive set of tools for creating static, animated, and interactive visualizations in various formats. Matplotlib is flexible and highly customizable, allowing users to create a wide range of plots and charts to analyze and present their data.

Here are some key features and information about the Matplotlib library:

- Plotting Functions: Matplotlib provides a wide variety of plotting functions to create different types of visualizations. These include line plots, scatter plots, bar plots, histogram plots, pie charts, 3D plots, and more. These functions can be used to visualize data from arrays, lists, pandas DataFrames, or any other data structures.

- Customization: Matplotlib offers extensive options for customizing the appearance of plots. You can control aspects such as colors, line styles, markers, labels, titles, axes, grids, legends, and more. This allows you to create visually appealing and informative plots that suit your specific requirements.

- Multiple Plotting Styles: Matplotlib supports multiple plotting styles, including the MATLAB-style interface, the object-oriented interface, and the pyplot interface. The MATLAB-style interface provides a simple and convenient way to create plots similar to those in MATLAB. The object-oriented interface offers more control and flexibility over plot elements. The pyplot interface provides a state-machine-like interface for creating plots interactively.

- Figures and Subplots: Matplotlib organizes plots within figures. A figure can contain multiple subplots, allowing you to create multiple plots within a single figure. This is useful for creating complex layouts and comparing different visualizations side by side.

- Backend Support: Matplotlib supports various backends, which determine how plots are displayed and saved. The default backend renders plots to a graphical user interface (GUI) window, but Matplotlib also supports backends for rendering plots to various file formats, including PNG, PDF, SVG, and more. Additionally, Matplotlib can be integrated with Jupyter Notebook to display plots inline within the notebook.

- Integration with NumPy and pandas: Matplotlib integrates seamlessly with NumPy and pandas, allowing you to create visualizations directly from NumPy arrays or pandas DataFrames. This simplifies the process of visualizing data and enables the combination of data manipulation and plotting in a single workflow.

- Animation and Interactivity: Matplotlib provides features for creating animations and interactive visualizations. You can animate plots by updating data in real-time, creating dynamic visualizations. Interactive features can be added to plots, enabling zooming, panning, and data exploration by interacting with the plots.

- Extensibility: Matplotlib is highly extensible, allowing users to create custom plot types and styles. It provides a rich ecosystem of add-on packages and toolkits, such as Seaborn, for specialized plotting needs. Matplotlib's API is designed to be easy to use and encourages developers to build upon and extend its functionality.

- Documentation and Community Support: Matplotlib has comprehensive documentation with detailed explanations, tutorials, and examples to help users learn and utilize its functionalities effectively. It also has an active community that provides support, shares knowledge, and contributes to the development and improvement of the library.

- Matplotlib is a powerful and versatile library for data visualization in Python. Its ability to create high-quality plots, customizable appearance, and integration with other scientific and data analysis libraries make it a popular choice for visualizing data in various domains, including scientific research, data analysis, and machine learning.

## D. ScikitLearn :

Scikit-learn, also known as sklearn, is a popular open-source machine learning library for Python. It provides a wide range of machine learning algorithms and tools for tasks such as classification, regression, clustering, dimensionality reduction, model selection, and preprocessing. Scikit-learn is built on top of other scientific computing libraries, such as NumPy, SciPy, and matplotlib, and is designed to be user-friendly and efficient.

Here are some key features and information about scikit-learn:

- Machine Learning Algorithms: Scikit-learn provides a comprehensive collection of machine learning algorithms, including both supervised and unsupervised learning methods. These algorithms cover a broad range of tasks, such as linear regression, logistic regression, support vector machines, decision trees, random forests, gradient boosting, k-means clustering, hierarchical clustering, and more.

- Consistent API: Scikit-learn offers a consistent and intuitive API across all its algorithms, making it easy to switch between different models and experiment with various techniques. This uniform interface allows for rapid prototyping, code reusability, and easy integration into existing workflows.

- Data Preprocessing: Scikit-learn provides a rich set of preprocessing modules to handle common data preprocessing tasks. It includes functions for handling missing data, scaling and normalization, encoding categorical variables, feature selection, and more.

These preprocessing techniques help to improve the quality of the data and enhance the performance of machine learning models.

- Model Evaluation and Selection: Scikit-learn offers tools for evaluating and selecting machine learning models. It provides functions for splitting datasets into training and test sets, cross-validation, and performance metrics such as accuracy, precision, recall, F1 score, and area under the curve (AUC). These tools help in assessing model performance and selecting the best model for a given task.

- Feature Extraction and Dimensionality Reduction: Scikit-learn includes techniques for feature extraction and dimensionality reduction. It provides methods like Principal Component Analysis (PCA), t-SNE, and various feature selection algorithms to reduce the dimensionality of the data and extract meaningful features. These techniques are useful for visualizing data, reducing computational complexity, and improving model performance.

- Model Persistence: Scikit-learn allows models to be saved and loaded for later use. This is particularly useful when training models on large datasets or when deploying models in production environments. The saved models can be easily reloaded, and predictions can be made without retraining the models.

- Integration with Other Libraries: Scikit-learn integrates seamlessly with other libraries in the Python ecosystem, such as NumPy, SciPy, pandas, and matplotlib. This allows for smooth data interchange and interoperability, enabling you to build end-to-end machine learning pipelines with ease.

- Extensibility: Scikit-learn is designed to be extensible, allowing users to create custom estimators by subclassing the base classes provided by the library. This flexibility enables the implementation of novel algorithms and techniques tailored to specific use cases.

- Documentation and Community Support: Scikit-learn has extensive documentation with clear explanations, tutorials, and examples to help users understand and apply its functionalities. The library also has an active community of users and developers who provide support, share knowledge, and contribute to its development and improvement.

- Scikit-learn is widely used by data scientists, researchers, and practitioners for machine learning tasks. Its easy-to-use interface, extensive algorithm selection, and robust functionality make it a valuable tool for a broad range of applications, from simple data analysis to complex predictive modeling.

## E. TPOT:


TPOT (Tree-based Pipeline Optimization Tool) is an automated machine learning (AutoML) library for Python. It automates the process of building and optimizing machine learning pipelines by intelligently searching and selecting the best combination of preprocessing techniques and models for a given dataset. TPOT combines concepts from genetic programming and tree-based algorithms to efficiently explore and optimize the search space of possible pipelines.

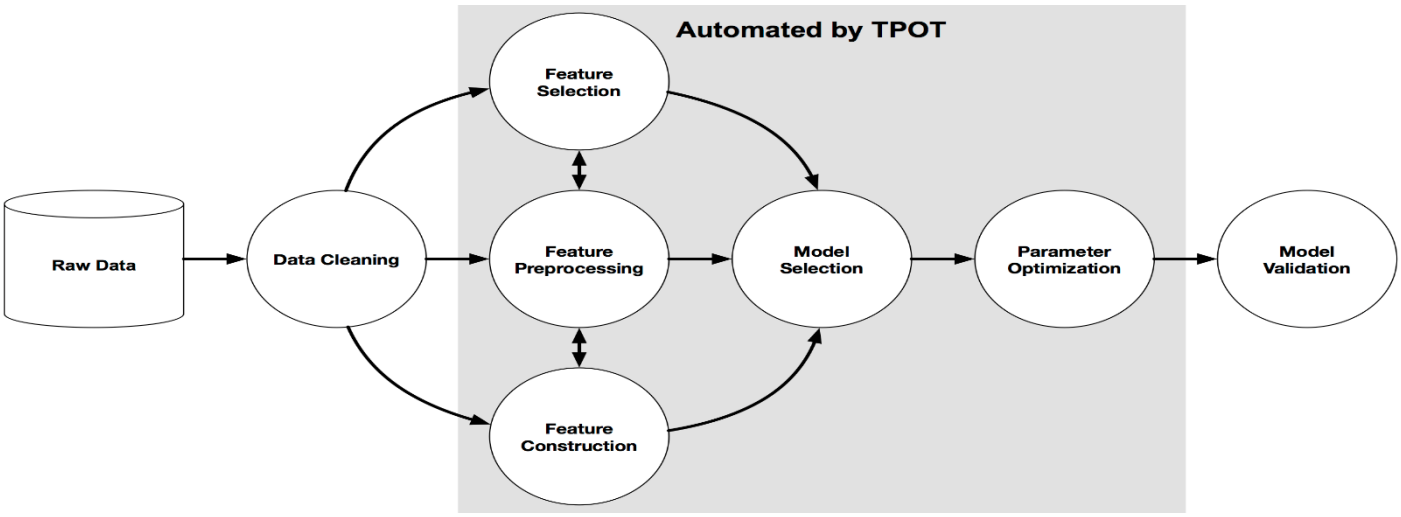



Fig.TPOT Pipeline example

Here are some key features and information about TPOT:

- Automated Machine Learning: TPOT automates the end-to-end process of machine learning, including data preprocessing, feature selection, model selection, hyperparameter tuning, and pipeline optimization. It eliminates the need for manual trial and error by automatically searching for the best pipeline configuration for a given dataset.

- Genetic Programming: TPOT employs a genetic programming-based optimization algorithm to evolve and breed a population of pipelines. It starts with a diverse population of randomly generated pipelines and iteratively improves them through generations by applying genetic operators such as crossover and mutation. The fitness of each pipeline is evaluated using a user-defined scoring metric, such as accuracy or mean squared error.

- Pipeline Representation: TPOT represents pipelines as directed acyclic graphs (DAGs), where nodes represent data preprocessing steps or models, and edges represent the flow of data between these steps. Each node in the pipeline can consist of various preprocessing techniques (e.g., feature scaling, dimensionality reduction) or machine learning models (e.g., random forests, support vector machines).

- Pipeline Search Space: TPOT searches a predefined search space of possible pipeline configurations, which includes different data preprocessing techniques and a wide range of machine learning models. The search space can be customized based on the problem domain and user preferences, allowing for flexibility in the types of pipelines that TPOT explores.

- Model Evaluation and Selection: TPOT uses cross-validation to evaluate the performance of different pipeline configurations. It estimates the performance of each pipeline on unseen data by splitting the training data into multiple folds and measuring the average performance across the folds. TPOT considers various performance metrics and selects the pipeline with the best performance as the final result.

- Parallelization: TPOT supports parallel execution, allowing it to leverage multiple CPU cores to accelerate the pipeline search and evaluation process. This speeds up the optimization process, especially when dealing with large datasets or complex pipelines.

- Interpretability: TPOT provides information about the best pipeline discovered during optimization, including the sequence of preprocessing steps and the selected machine learning model with its hyperparameters. This helps users understand the underlying pipeline and interpret the results.

- Extensibility: TPOT is designed to be extensible, allowing users to customize the optimization process and search space according to their specific needs. Users can define their own preprocessing techniques, models, and evaluation metrics to be included in the search process.

- Documentation and Community Support: TPOT has comprehensive documentation that includes explanations, tutorials, and examples to help users understand and effectively use the library. It also has an active community of users and developers who provide

support and share knowledge.

- TPOT offers a powerful and efficient approach to automate the machine learning pipeline construction and optimization process. It reduces the manual effort required to find the best pipeline configuration and allows users to focus on higher-level tasks such as problem understanding and data analysis. TPOT is a valuable tool for both beginners and experienced practitioners looking to streamline their machine learning workflows.

# 3.IMPLEMENTATION

## 3.1 DATASET DESCRIPTION :

The dataset, 'transfusion.csv', obtained from the Machine Learning Repository, consists of a random sample of 748 donors. Our dataset is from a mobile blood donation vehicle in Taiwan. The Blood Transfusion Service Center drives to different universities and collects blood as part of a blood drive. We want to predict whether or not a donor will give blood the next time the vehicle comes tocampus. It is structured according to RFMTC marketing model (a variation of RFM).

RFM stands for Recency, Frequency and Monetary Value and it is commonly used in marketing for identifying your best customers. In our case, our customersare blood donors.

RFMTC is a variation of the RFM model. Below is a description of what each column means in our dataset:

- R (Recency - months since the last donation)
- F (Frequency - total number of donation)
- M (Monetary - total blood donated in c.c.)
- T (Time - months since the first donation)
- a binary variable representing whether he/she donated blood in March 2007(1 stands for donating blood; 0 stands for not donating blood).

## 3.2 STATISTICAL INSIGHTS OF DATASET:

Every dataset contains multiple hidden information which can be seen by performing statistical analysis on it. The insights found by performing statisticalanalysis on our dataset are the following: -

- Every column in our dataset is of numeric type.
- After specifying the features and target column, it is found that thedistribution of target class in target columns are as:
    - **0: 0.762**
    - **1: 0.238**
- Due to the uneven distribution of both classes, splitting of dataset is performed using the following parameters:
    - Test size: **0.25**
- Due to the uneven distribution of both classes, splitting of dataset is performed using the following parameters:
    - Test size: **0.25**
    - Random State : **42**
    - Stratify : Target column

- The correlation matrix describing the correlation between different features through a heatmap is shown below :

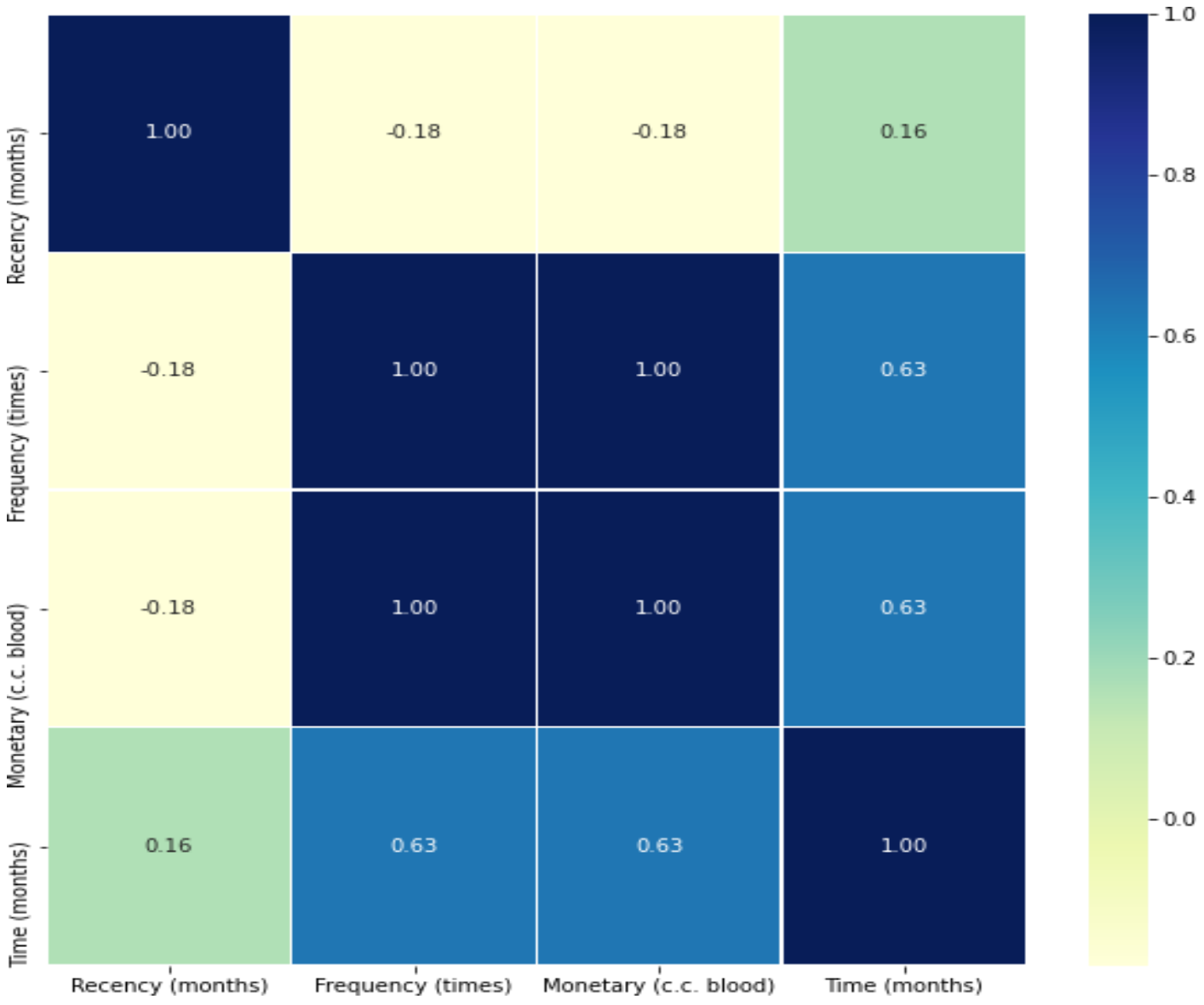



Fig. Correlation Heat Map

## 3.3 MODEL TRAINING AND EVALUATION :

- One of the assumptions for linear regression models is that the data and the features we are giving it are related in a linear fashion, or can be measured with a linear distance metric. If a feature in our dataset has a high variance that's an order of magnitude or greater than the other features, this could impact the model's ability to learn from other features in the dataset.
- Correcting for high variance is called normalization. It is one of the possible transformations we do before training a model.
- Monetary (c.c. blood)'s variance was very high in comparison to any other column in the dataset. This means that, unless accounted for, this feature might get more weight by the model (i.e., be seen as more important) than any other feature. One way to correct for high variance was to use log normalization.

Finally, the logistic regression model is trained with the following parameters:

  - Solver = liblinear
  - Random State = 42

- The model evaluation is performed by checking AUC Score of the model which is **0.7891**.

The Jupyter notebook can be viewed at :

**GitHub Link** **OR**

# 4.CONCLUSION AND FUTURE SCOPE

The demand for blood fluctuates throughout the year. As one prominent example, blood donations slow down during busy holiday seasons. An accurate forecast forthe future supply of blood allows for an appropriate action to be taken ahead of time and therefore saving more lives.

In this project, we explored automatic model selection using TPOT and AUC score which was **0.7850**. This is better than simply choosing '0' all the time (thetarget incidence suggests that such a model would have 76% success rate). We then log normalized our training data and improved the AUC score by 0.5%. In the field of machine learning, even small improvements in accuracy can be important, depending on the purpose.

Another benefit of using logistic regression model is that it is interpretable. We can analyze how much of the variance in the response variable (target) can be explained by other variables in our dataset.

Pre-donation information and counselling are linked to the process of donor selection in which each individual's suitability to donate is carefully assessed against a set of criteria related to their medical history.

Pre-donation information is an important first step in informing and educating donors about the blood donation process, including donor selection criteria and deferral or self-deferral, blood screening for TTI, blood grouping, counselling and referral. This will enable individuals who may be unsuitable to donate bloodto self-defer without going through the blood donation process.

# 5.REFERENCES

The following websites have been referred for input data and statistics:


- https://www.webmd.com/a-to-z-guides/blood-transfusion-what-to-know#1
- https://www.kjrh.com/news/local-news/red-cross-in-blood-donation-crisis
- https://www.ncbi.nlm.nih.gov/books/NBK310569/
- http://epistasislab.github.io/tpot/


The following websites have been referred for coding part:-


- https://www.python.org/
- https://github.com/perborgen/LogisticRegression
- http://epistasislab.github.io/tpot/