Heart Risk Analysis using Big Data and Machine Learning Techniques

Group -11

Krishnasai Bharadwaj Atmakuri - 12597191

Kavya Reddy Manchireddy - 16342232

Hema Nagini Matta - 16347557

Phani Rajesh Kulkarni - 16342930

School of Computing and Engineering: University of Missouri - Kansas City CS, Principles of Big Data Management

Professor - Muhammad Asim

1. Abstract

The prediction of heart attack is a critical issue in the field of healthcare. The use of machine learning techniques has been proposed as a means to accurately predict the occurrence of heart attacks. In this study, the process of developing machine learning models for heart attack prediction using a dataset containing various risk factors. The dataset was preprocessed to clean the data, further the data is processed according to Random Forest Classifier and Naïve Bayes Classifier. Then the processed data undergoes feature selection, by selecting the most suitable features and therefore converting them into feature vector using one of the feature transformation techniques and then given as an input to different machine learning algorithms including random forest classifier and naïve bayes classifier to predict heart attack. The performance of each algorithm was evaluated using metrics such as accuracy, f1-score, precision and recall. This study demonstrates the potential of machine learning techniques on classification problems in predicting heart attacks and could help healthcare providers to identify individuals at risk of heart attack and take preventive measures.

The implementation was done in Google Colab using PySpark and Python and the in-built libraries such as matplotlib, etc., allowing us to perform heart stroke prediction on the entire heart disease dataset in a reasonable amount of time.

This project provides a useful tool for heart stroke prediction on heart disease data using PySpark and machine learning algorithms. It demonstrates the effectiveness of PySpark as well as Python for processing large datasets and extracting insights from the obtained heart disease data.

2. Introduction

Every year, 13 million individuals suffer a stroke according to the World Stroke Organization, resulting in about 5.5 million deaths, making it the primary cause of fatality and disability across the world. This condition has severe implications on all aspects of life since it not only influence the patients but also their social circle, family, and work environment. Furthermore, it is a common misconception that stroke only affects specific demographics, but in reality, it can occur at any age or physical condition, regardless of gender.

In this project, the heart attack prediction on the heart disease dataset which contains multiple attributes aims to develop a model that can accurately predict the heart stroke field (Stroke) based on the other 17 fields (‘HeartDisease’, ’BMI’, ’Smoking’, ’AlcoholDrinking’, ‘PhysicalHealth’, ‘MentalHealth’, ‘DiffWalking’, ‘Sex’, ‘AgeCategory’, ‘Race’, ‘Diabetic’, ‘PhysicalActivity’, ‘GenHealth’, ‘SleepTime’, ‘Asthma’, ‘KidneyDisease’, ‘SkinCancer’).

To accomplish this, PySpark, a powerful distributed computing framework that enables us to process large datasets efficiently is being used along with some popular machine learning algorithms namely Random Forest Classifier and Naïve Bayes Classifier.

The project is implemented in Google Colab, which provides a free and easy-to-use environment for working with PySpark, Python and machine learning libraries. The results show that the models perform well on the heart disease dataset with better accuracy.

This project uses PySpark, Python and machine learning algorithms to provide useful tools for heart disease data. This demonstrates the effectiveness of PySpark and Python for processing the large datasets and extracting insights from healthcare data. The results of this project can be applied by healthcare industries to improve their services by better understanding the serious causes of heart strokes in their patients.

2.a. Project Goals & Objectives

The main goal of this project is to perform heart attack prediction on the heart disease dataset using PySpark, Python, and machine learning algorithms. The specific objectives of the project include:

- Preprocessing the dataset for cleaning and processing the cleaned data according to machine learning algorithms.
- Analyzing the data by using python libraries to calculate the correlation between the features and visualize the data to get some valuable insights for feature selection.
- Implementing feature transformer namely Vector Assembler for feature transformation to get a feature vector as an input to the machine learning models based on the results of the feature selection mentioned before.
- Dividing the data into training and testing partitions to train the models and then to evaluate the models using various evaluation metrics such as accuracy, etc. to determine their performance on the testing dataset.
- Testing is done by passing a dummy or a sample feature vector to the trained models to verify the quality of the model to predict the heart stroke/attack.
- Demonstrating the effectiveness of PySpark for performing heart attack prediction on heart disease dataset, making it a valuable tool for healthcare industries to improve their services by better understanding the serious causes of heart diseases in their patients.
- By achieving these objectives, the project aims to provide a useful tool for heart attack prediction using PySpark, Python and machine learning algorithms.

2.b. Project Scope

The main scope of the project is to predict heart attack/stroke on the heart disease dataset using PySpark, Python and machine learning algorithms. The dataset contains different attributes such as BMI, Smoking, Alcohol Drinking, Stroke, Physical, Mental Health, Diffwalking, Sex, AgeCategory, Diabetic, Race, Physical Activity, Gen Health, Sleep time, Asthma, KidneyDisease, SkinCancer making it an ideal dataset for training and evaluating machine learning models. The project focuses on using machine learning algorithm such as Random Forest classifier and Naïve Bayes classifier for prediction of heart stroke.

The project involves cleaning and preprocessing the data, followed by data analysis to identify patterns and relationships between variables. Feature transformation is done to create a feature vector, and important features are selected based on the results of the analysis. The data is then splitted into training and testing sets, and the model is trained and evaluated using different metrics. Finally, the model is tested by passing a sample feature vector to verify its quality in predicting heart stroke/attack.

The implementation of the project is done using PySpark, a powerful distributed computing framework that allows for the efficient processing of large datasets. The use of PySpark enables us to perform heart stroke prediction on the entire heart disease dataset in a reasonable amount of time, which would not have been feasible using traditional machine-learning tools.

2.c. Project Limitations & Constraints

Here are the possible limitations and constraints for the project:

1. Data Constraints: The data related to heart disease has certain limitations in terms of its size and the types of features included, unlike data collected from hospitals or research institutes which may have more diverse and extensive data.

2. Accuracy: Although the machine learning algorithms utilized in the project are established and commonly used, there is no assurance that they will yield precise predictions for all feature types. As a result, the reliability of the prediction analysis findings could be limited.

3. Time Constraints: Even though data models consisting of various features that capture a comprehensive health profile of individuals could provide more detailed information, obtaining access to such data is often a challenging and time-consuming task due to privacy concerns.

4. Resources: The project might necessitate substantial computational resources, such as high-performance computing clusters, to effectively analyze and process the publicly accessible data set. If such resources are not readily available, this could pose a limitation.

5. Code dependencies: The project depends on various external libraries and frameworks, like PySpark and pandas, etc., which may require particular version or configuration. This could pose a limitation if there are compatibility concerns or version discrepancies with other dependencies.

2.d. Feasibility Study

1. Technical feasibility:

The project requires a good understanding of machine learning algorithms, PySpark, and data preprocessing techniques. It is feasible if the project team has the required technical knowledge and expertise.

The project requires a significant number of computational resources. It is feasible if the team has access to a computing environment that can handle large datasets and parallel processing.

The project requires a heart disease dataset which is freely available for research purposes.

2. Financial feasibility:

The project requires the use of PySpark, which is open-source software and can be used for free.

The project team may need to spend money on training or hiring additional resources if they lack the required technical expertise.

2.e. Work Break-Down Structure

Work Break-Down Structure	Time (in hours)	Person Assigned	Man Power
1. Research on the project and data set selection.	8	Bharadwaj Kavya Hema Phani	Bharadwaj - 2 Kavya – 2 Hema - 2 Phani - 2
2. Requirement analysis (software, hardware and functional)	4	Bharadwaj Kavya Hema Phani	Bharadwaj - 1 Kavya - 1 Hema - 1 Phani - 1
3. System Design (Architectural, Use-Case, and Sequence diagrams)	6	Bharadwaj Kavya Hema Phani	Bharadwaj – 1.5 Kavya – 1.5 Hema – 1.5 Phani – 1.5
4. Data Design (ETL Design, data management, data engineering, analysis, modelling and visualization)	16	Bharadwaj Kavya Hema Phani	Bharadwaj - 4 Kavya - 4 Hema - 4 Phani - 4
5. Code	70		
5.1. Installation and setup of Pyspark, Java, Hadoop and initialization.	1	Hema	Hema - 1

5.2. ETL Process (converting .xlsx to .csv file, and preprocessing the data by cleaning and processing the cleaned data according to the the ML model	4	Hema	Hema - 4
5.3.Data Analysis (Visualization of different features and perform pair-wise correlation of all features and of the Stroke field)	16	Bharadwaj Kavya Hema Phani	Bharadwaj - 4 Kavya - 4 Hema - 4 Phani - 4
5.4. Prediction of Heart Stroke using Machine Learning Models	50		
5.4.1 Balancing the imbalanced dataset using resampling technique	6	Bharadwaj Kavya	Bharadwaj -1 Kavya - 5
5.4.2 Random Forest Classifier (Feature transformation, splitting into training and testing data and implementation of	24	Bharadwaj	Bharadwaj - 24

the model)			
5.4.2 Naïve Bayes Classifier (Feature transformation, splitting into training and testing data and implementation of the model)	20	Bharadwaj	Bharadwaj - 20
6. Testing the models with a dummy feature vector.	4	Phani	Phani - 4
7. Report Preparation	16	Bharadwaj Kavya Hema Phani	Bharadwaj -1 Kavya - 6 Hema -3 Phani - 6
8. Group/Scrum Meetings	80	Bharadwaj Kavya Hema Phani	Bharadwaj - 20 Kavya - 20 Hema - 20 Phani - 20
Total Hours	205	Bharadwaj Kavya Hema Phani	Bharadwaj – 78.5 Kavya – 43.5 Hema – 40.5 Phani – 42.5

Table1: Work Breakdown Structure

3. System Requirement Specifications (SRS) - MDRE

3.a. Software Requirements:

The following software packages and libraries were used for the implementation of the project:

- PySpark: A powerful distributed computing framework for processing large datasets.
- Pandas: A popular data manipulation library in Python.
- Matplotlib: A data visualization library in Python.
- Seaborn: popular Python data visualization library built on top of Matplotlib.

3.b. Hardware Requirements:

- Google Colab, a cloud-based platform for data analysis and machine learning.

As such, there are no specific hardware requirements for running the project on a local machine. However, a stable internet connection is required to access and use the Google Colab platform.

3.c. Functional Requirements:

The project's functional requirements include:

- Preprocessing the heart disease dataset by cleaning the data.
- Processing the cleaned data according to machine learning algorithms.
- Performing data analysis to calculate correlation between all the features and to visualize the data for valuable insights, which helps in the feature selection.
- Extracting the features using feature transformer, Vector Assembler.
- Splitting the feature vector into training and testing datasets.
- Implementing and training machine learning models on the training dataset.
- Evaluating the performance of the model using different evaluation metrics such as accuracy.
- Testing the trained models using a dummy feature vector to evaluate the quality of the trained models and their performance on any unseen dataset.

4. System Design

4.a. Architectural Diagram:

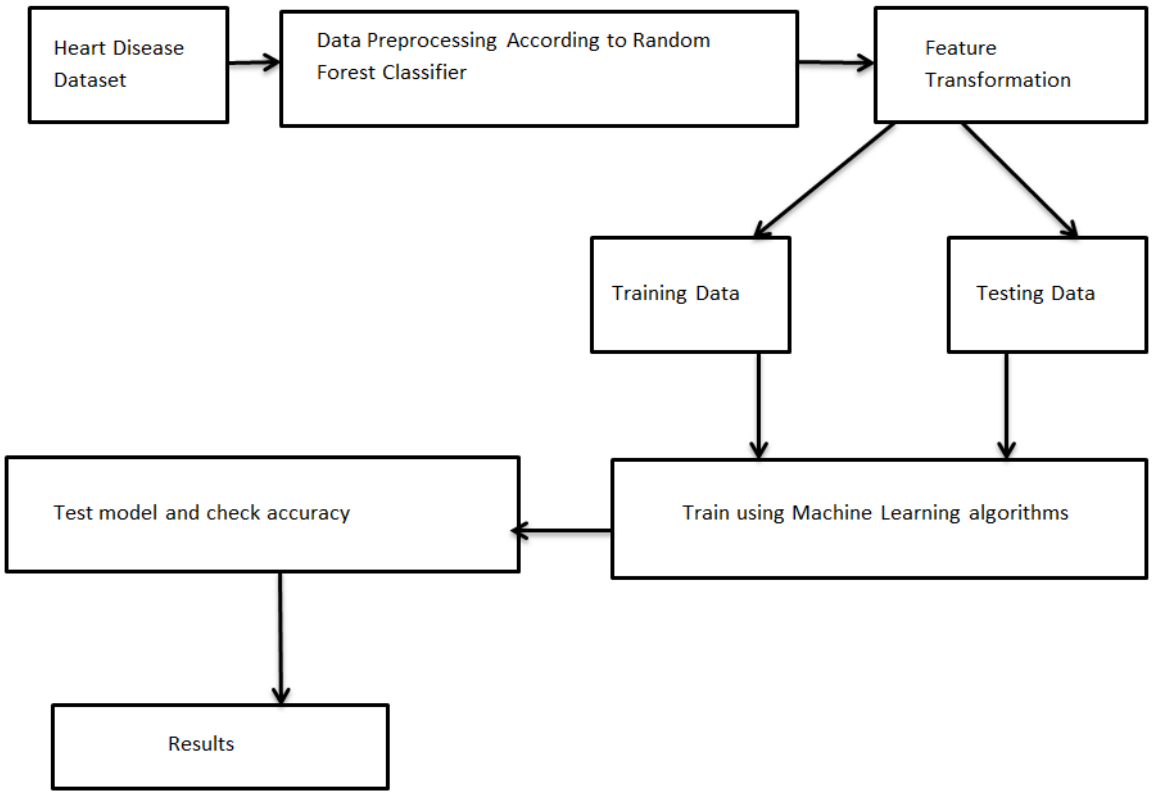

Fig 1: Architectural Diagram

4.b. Use-Case Diagram :

A use-case diagram provides a visual representation of the potential interactions between a user and a system. The diagram shows the system's actors, the goals they aim to achieve (i.e., use cases), and relationships or dependencies between these use cases. The primary aim of this diagram is to visually convey the various functions that the system performs for each actor. Additionally, it can portray the different roles that actors play within the system.

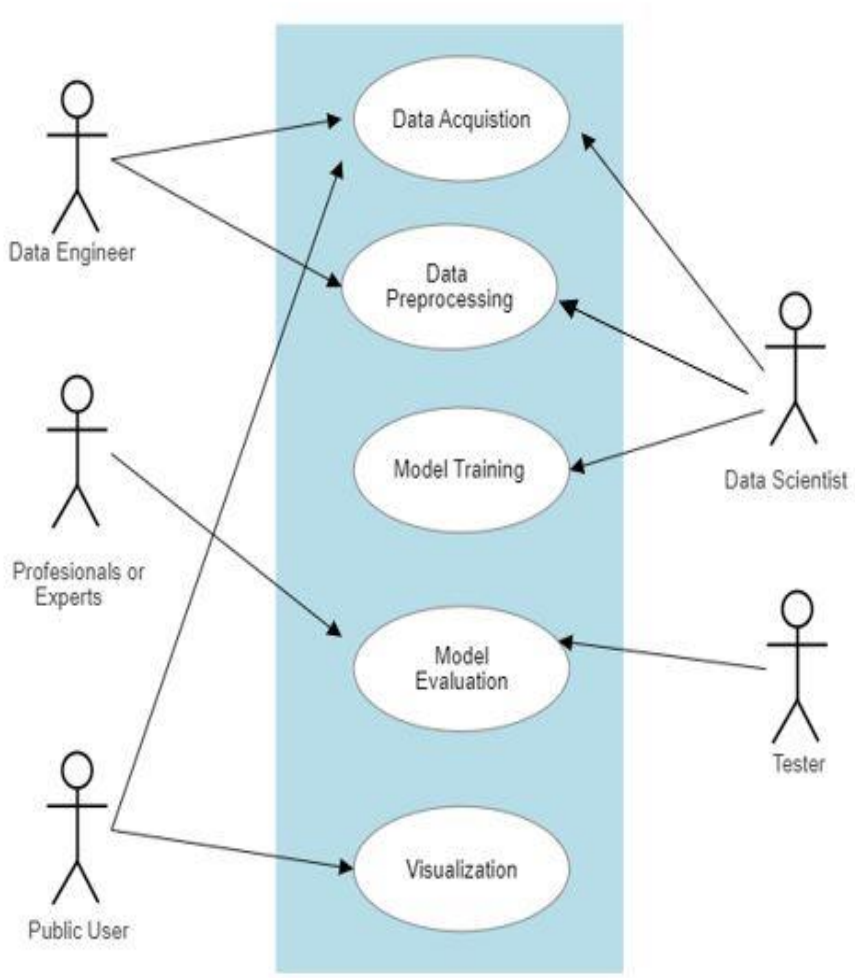

Fig 2: Use-Case Diagram.

4.c. Sequence Diagram :

A sequence diagram is a graphical representation of a scenario that manifest the sequence of messages exchanged between objects and classes involved in the scenario over time. The diagram depicts the specific objects and classes entrangled in the scenario and illustrates the sequence of interactions needed to accomplish the desired functionality. In other words, it provides a visual representation of the communication between objects to achieve a specific outcome**.**

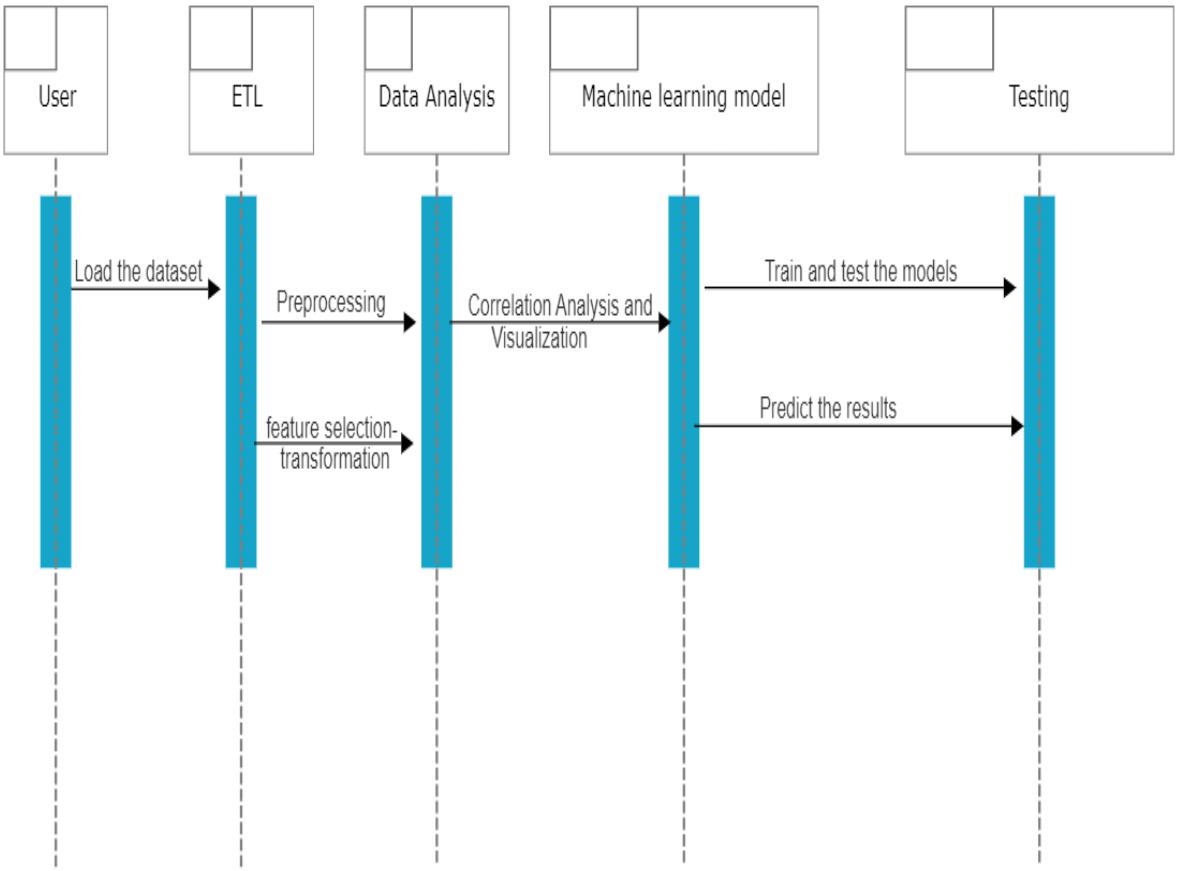

Fig 3: Sequence Diagram.

5. Data Design

5.a. ETL Process

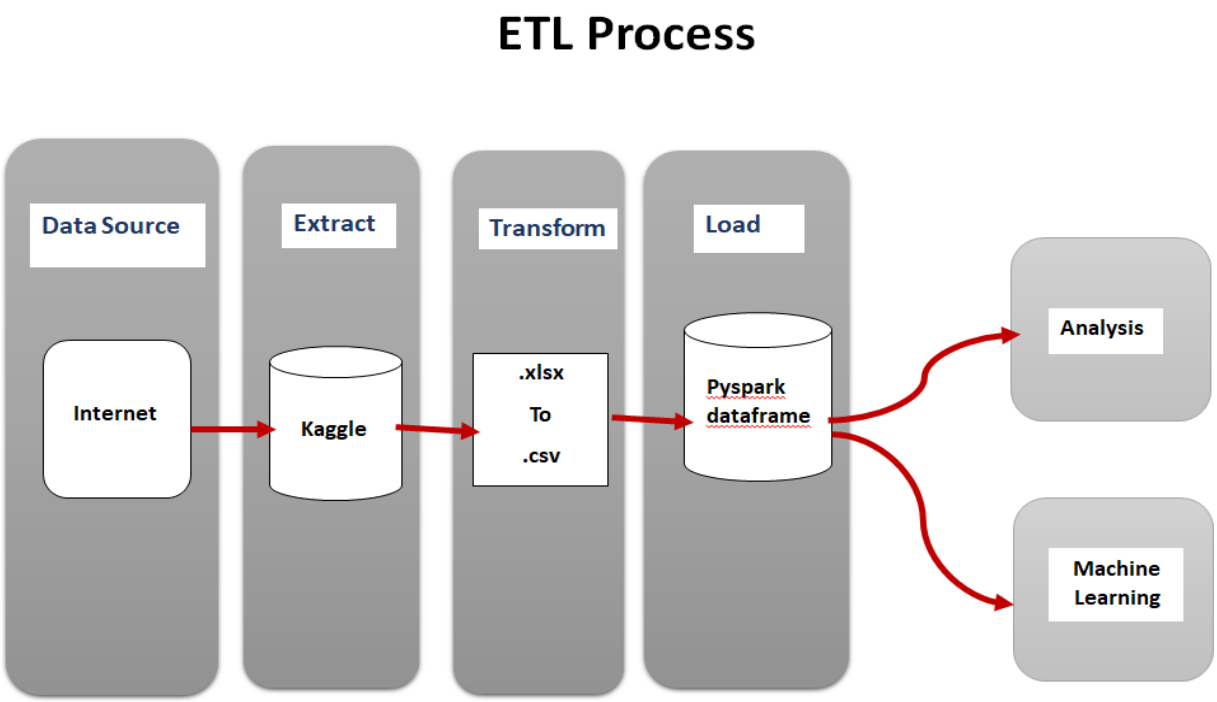

Fig 4: ETL Process.

Data source: The data source that is being used for heart risk analysis using big data and machine learning techniques is taken from the Internet.

Extract: Data is extracted from the Kaggle which is a popular online platform that hosts machine learning and data science competitions, as well as provides access to a wide range of public datasets.

Data Transformation: The integrated data is transformed into a format that can be used for machine learning from .xlsv to .csv.

Load: Once data has been transformed, it can be loaded into a PySpark DataFrame, which is a distributed collection of data similar to a relational database or a Pandas DataFrame.

5.b. Data Management:

Effective data management is crucial for heart risk analysis using big data and machine learning techniques. Here are some key considerations for data management in this context.

1. Data collection: Collecting relevant and high-quality data is critical for accurate heart risk analysis. Information can be obtained from diverse origins, such as digital medical records, portable gadgets, and medical scans, among others. It is important to ensure that the data is representative of the population being studied and that it is collected ethically and in compliance with applicable regulations.

2. Data preprocessing: Raw data often needs to be preprocessed before it can be used for heart risk analysis. This can involve tasks such as cleaning, transformation, and feature extraction. It is important to document all preprocessing steps to ensure reproducibility and transparency.

3. Data storage: Big data requires a robust and scalable storage solution. PySpark supports various data storage options, including Hadoop Distributed File System (HDFS). It is important to choose a storage solution that meets the specific needs of the heart risk analysis project, such as data volume, data access patterns, and data retention requirements.

4. Data processing: PySpark provides a powerful distributed computing framework for processing big data. It is important to optimize data processing pipelines for efficiency and scalability, such as by parallelizing computations, minimizing data shuffling, and caching frequently used data.

5.c. Data Engineering:

Data engineering plays a crucial role in heart risk analysis using big data and machine learning techniques. Here are some key considerations for data engineering in this context:

1. Data ingestion: Data ingestion involves collecting and processing data from various sources, such as electronic health records, wearable devices, and medical imaging. This data may be structured or unstructured, and may require preprocessing before it can be used for analysis. Data ingestion pipelines should be designed to handle large volumes of data and to minimize data loss and corruption.

2. **Data storage**: Big data requires a scalable and efficient storage solution. PySpark supports various data storage options, including Hadoop Distributed File System (HDFS). The choice of storage solution will depend on factors such as data volume, data access patterns, and data retention requirements.

3. Data processing: PySpark provides a powerful distributed computing framework for processing big data. Data processing pipelines should be designed to optimize performance, scalability, and fault tolerance. This may involve techniques such as data partitioning, caching, and parallelization.

4. Data transformation: Data transformation involves cleaning, filtering, and transforming raw data into a format suitable for analysis. This may involve tasks such as feature extraction, normalization, and data imputation. It is important to document all transformation steps to ensure reproducibility and transparency.

5. Data visualization: Data visualization is an important tool for communicating heart risk analysis results to stakeholders. Visualization tools such as matplotlib, seaborn, and plotly can be used to create informative and engaging visualizations.

6. **Data modeling**: Data modeling involves developing machine learning models that can predict heart risk based on input data. This may involve techniques such as regression analysis, tree based models, and neural networks. It is important to evaluate model performance and to document all modeling decisions.

5.d. Data Analysis and Modelling:

Data analysis and modeling are key steps in heart risk analysis using big data and machine learning techniques. Here are some key considerations for data analysis and modeling in this context:

1. Exploratory data analysis: involves examining the data to identify patterns, relationships, and anomalies. This can involve techniques such as data visualization, statistical analysis, and data profiling. The goal of exploratory data analysis is to gain insights into the data and to inform subsequent modeling decisions.

2. Feature engineering: Feature engineering involves selecting, transforming, and extracting features from the raw data as an input to the prediction models. This may involve techniques such as principal component analysis, feature scaling, and feature selection. Feature engineering can have a outstanding impact on model performance, so it is very crucial to choose appropriate features and to evaluate their effectiveness.

3. Model selection: Model selection involves choosing an appropriate machine learning algorithm for the heart risk analysis task. This may involve techniques such as regression analysis, tree based models, and neural networks. The choice of algorithm will depend on factors such as data characteristics, performance metrics, and interpretability.

4. Model evaluation: Model evaluation involves assessing the performance of the predictive models on a held-out test set. This may involve metrics such as accuracy, precision, recall, and F1-score. It is important to choose appropriate evaluation metrics that reflect the goals of the heart risk analysis task.

5. Hyperparameter tuning: Machine learning models have hyperparameters that control the quality of the algorithm, such as the number of bins, or number of hidden layers, the learning rate. The technique hyper-parameter tuning involves selecting optimal values for these hyperparameters to improve model performance. This may involve techniques such as grid search, random search, or Bayesian optimization.

6. Model deployment: Model deployment involves integrating the machine learning model into a production system, such as a web application or a medical decision support system. This may involve considerations such as model performance, scalability, security, and privacy.

5.e. Data Visualization:

Matplotlib: widely-used Python library offers comprehensive capabilities for generating static, interactive, and animated visual representations of data. Its broad range of plot types encompasses line plots, scatter plots, bar plots, histograms, heatmaps, and additional chart types.

Seaborn: Seaborn is a Python library used for data visualization that is constructed on Matplotlib. It offers a more advanced and user-friendly interface for generating statistical graphics that are both visually attractive and informative, unlike the standard plots produced by Matplotlib.

Visualizations:

1. Plot 1 - Bar plot/chart of number of persons having never diagnoised with major diseases like Asthma, Heart Disease, Skin Cancer, and Kidney Disease and got a Heart Stroke and it`s count is being visualized by grouping according to the General Health field.

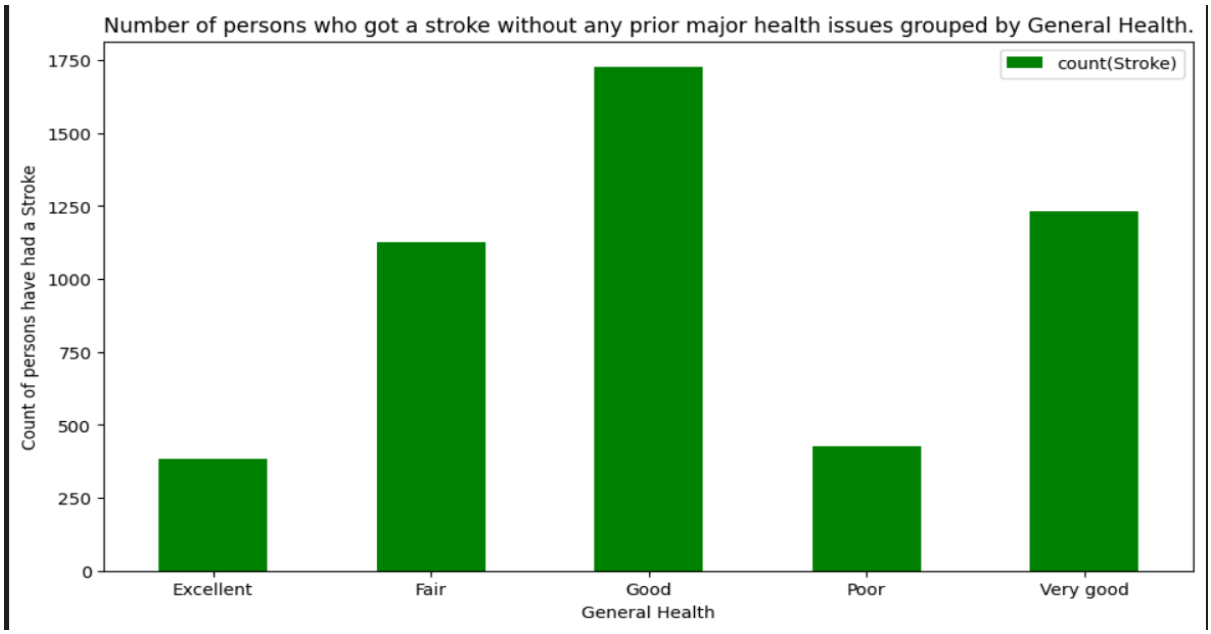

Fig 5: Bar plot for Number of persons who got a stroke without any prior major health issues grouped by General Health.

2. Plot 2 - Stacked Bar Plot/Chart of the count of the persons having Asthma and habits like Smoking, and got a Heart Stroke visualized by grouping according to the Age Category and the percentage of the Race for each "AgeCategory" Category.

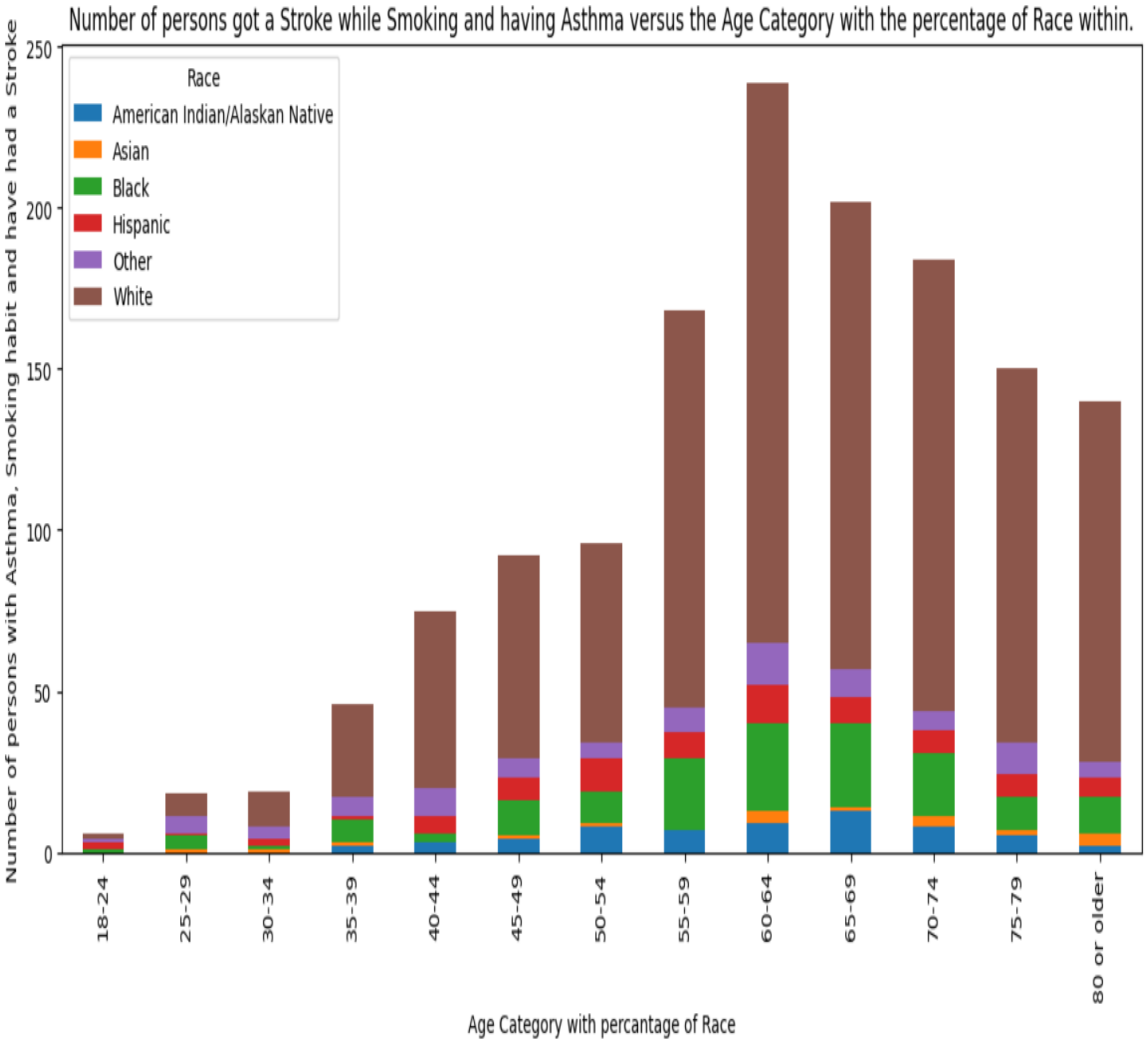

Fig 6: Stacked Bar Plot fir Nuber of persons got a stroke while smoking and having Asthama versus the Age Category with percentage of Race within.

3. Plot 3 - Histograms of the persons who got a Heart Stroke visualized against the BMI index and splitting the data according to Sex field (Male and Female) into two groups which produces two histograms.

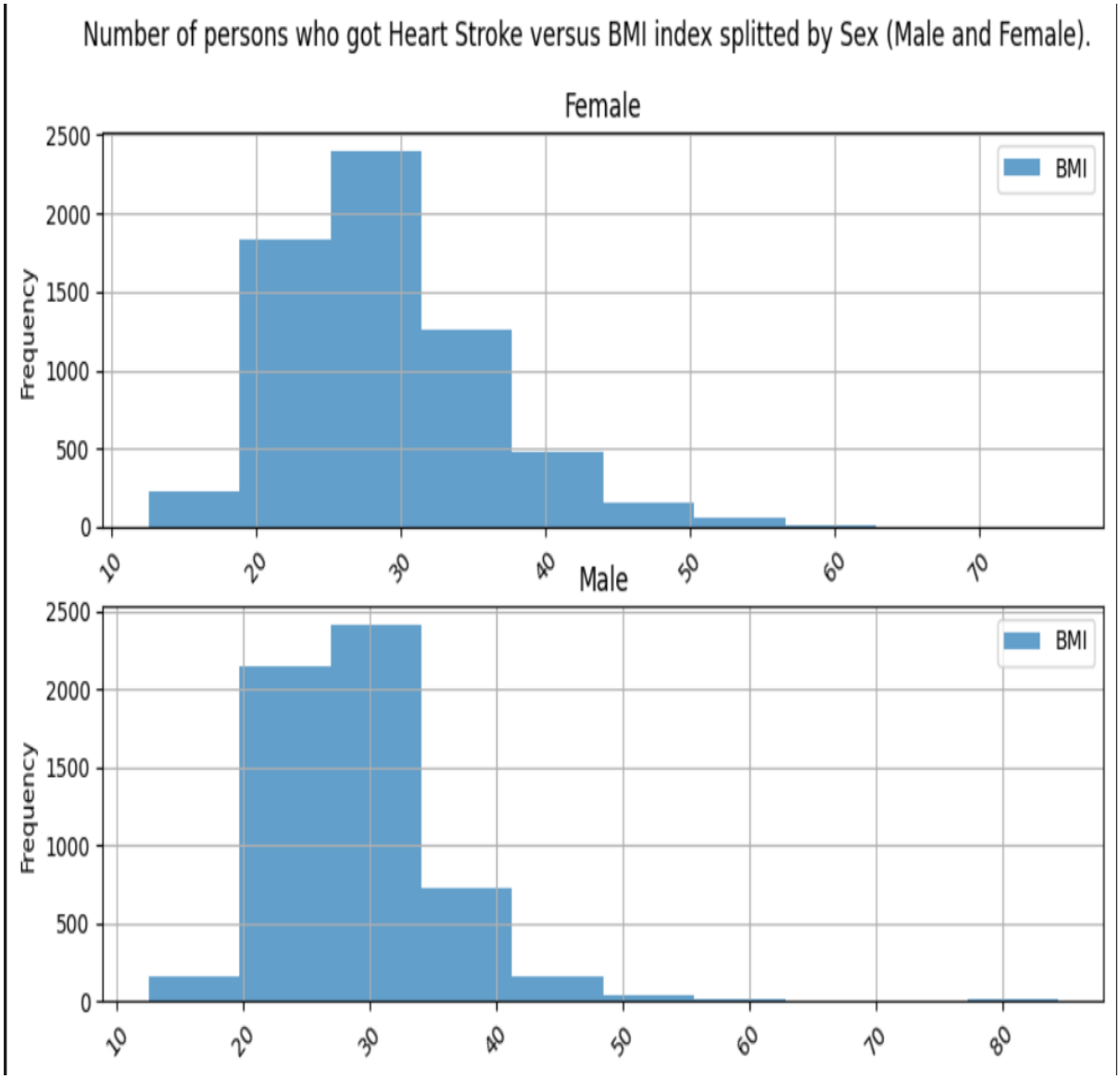

Fig 7: Number of persons who got heart stroke versus BMI Index splitted by Sex(Male and Female).

4. Plot 4 - Grouped Bar Plot/Chart of the count of the persons having previous Heart Stroke or a Heart Disease with Drinking or Smoking habit, and no major diseases visualized by grouping according to the Sex (Male and Female) and the percentage of the Stroke and Heart Disease within.

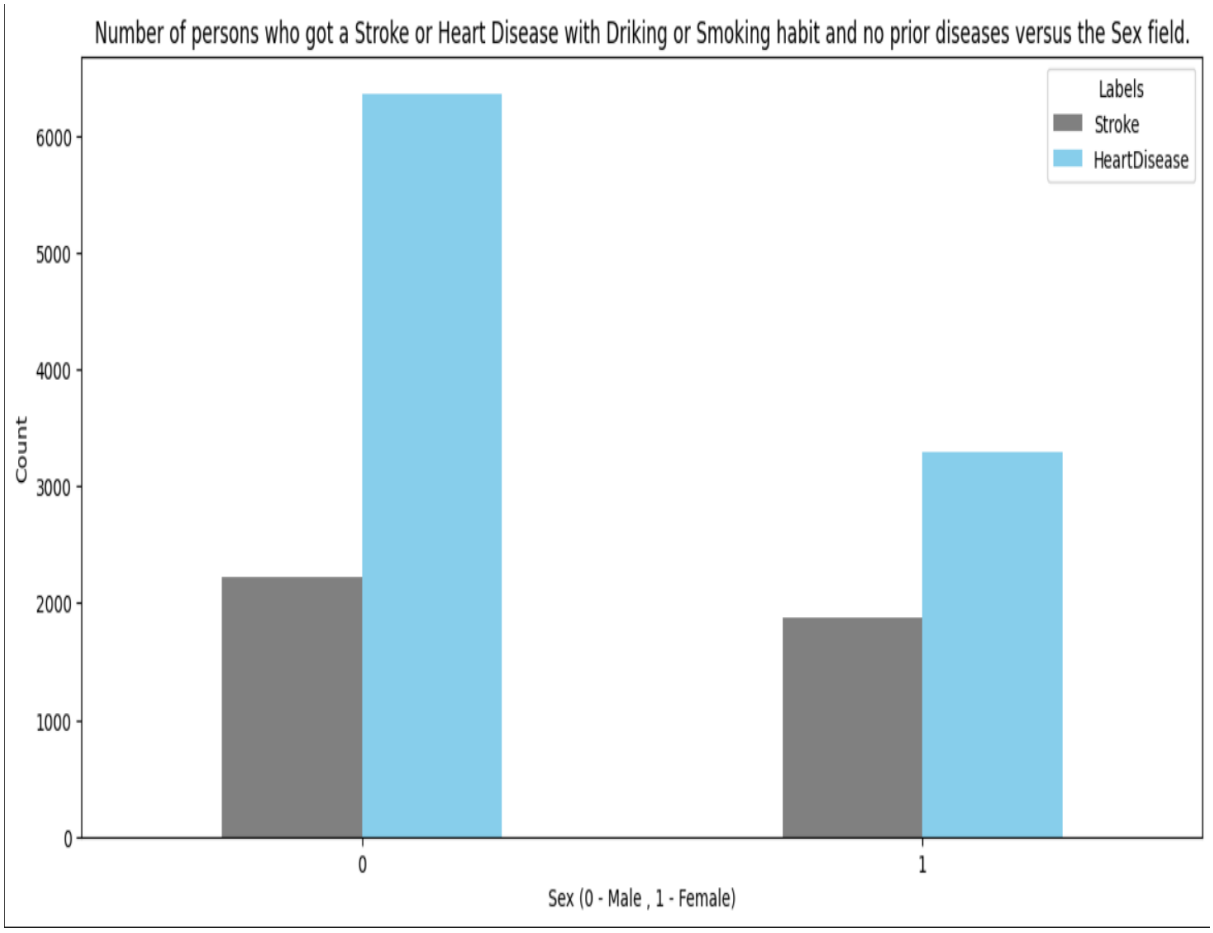

Fig 8: Grouped bar plot for Number of persons who got a stroke or Heart Disease with Drinking or smoking habit and no prior diseases versus the sex field.

5. Plot 5 - Pie Charts/ Plots of the persons have had a Heart Stroke visualized by grouping according to the General Health fields which invloves pregnancy and diabetes in gener.

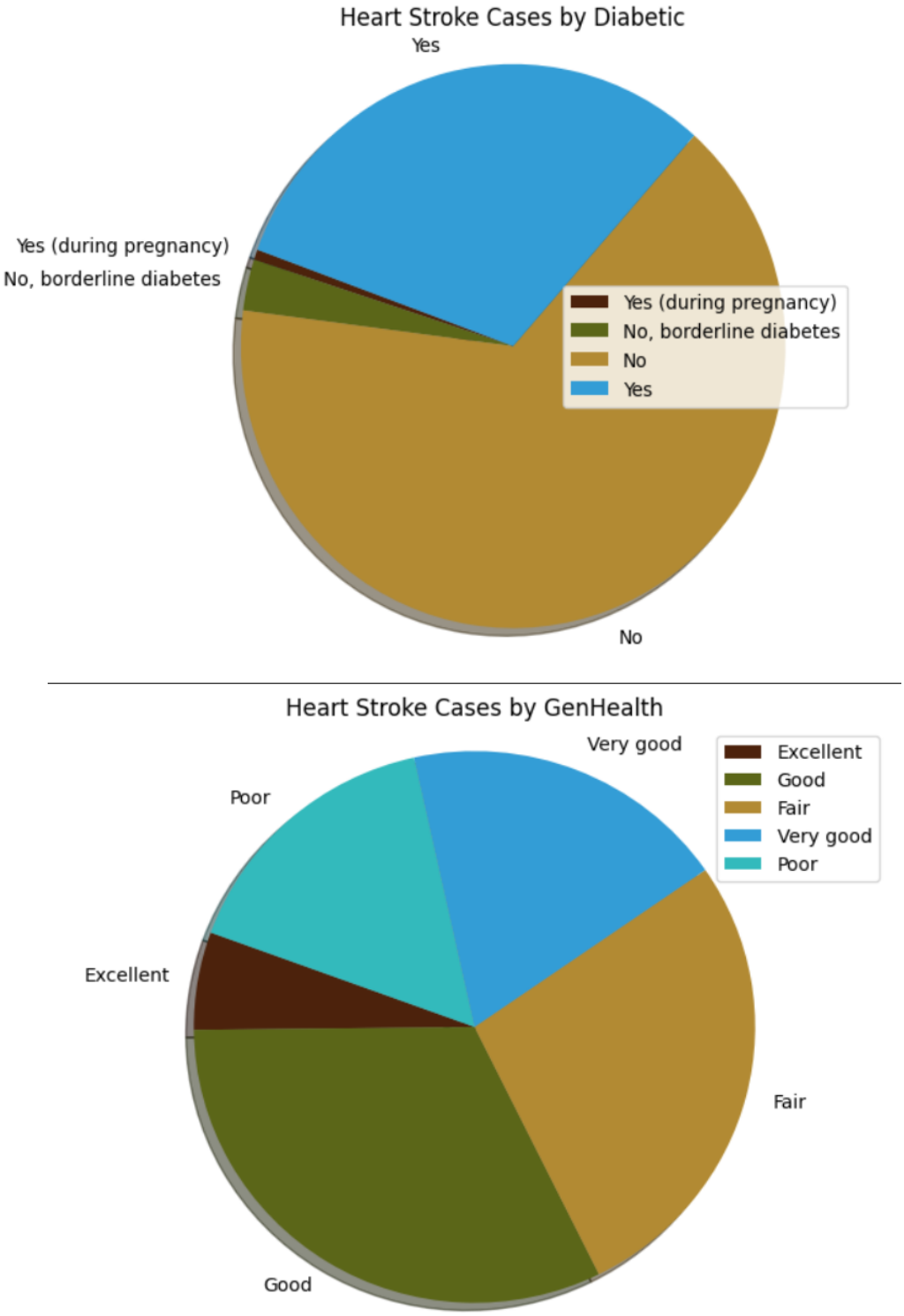

Fig 9: Pie Chart for Heart stroke cases by Diabetic and by GenHealth.

6. Code

1. Balancing the imbalanced dataset using resampling technique:

In General, it is said that a dataset is balanced when there is a ratio of 70:30 on class variation of the features. Since the heart disease dataset is imbalanced, resample technique is being used to generate a greater number of the minority class features.

```
# splitting data into majority and minority based on the Stroke field

heartrisk_df_majority = correlation_stroke[correlation_stroke.Stroke==0]

heartrisk_df_minority = correlation_stroke[correlation_stroke.Stroke==1]

# resampling the dataframe

heartrisk_df_minority_resample = resample(heartrisk_df_minority,

                                          replace=True,

                                          n_samples = 60345,

                                          random_state=7191)

# concatenating the resampled and majority data

heartrisk_df_balanced_pandas =
pd_pro.concat([heartrisk_df_majority,heartrisk_df_minority_resample])
```

2. Feature Transformation using Vector Assembler:

The independent variables must be vectorized to be accepted as an input for the machine learning algorithms, so feature tranformation is performed using Vector Assembler along with the dependant class label/ predictor feature.

```
# importing Vector Assembler for feature tranformation

from pyspark.ml.feature import VectorAssembler

# tranforming the features into a single vector feature

assembler_pro = VectorAssembler(inputCols=['HeartDisease','DiffWalking','AgeCategory'],

                                outputCol="features")

feature_vec_ml=assembler_pro.transform(heartrisk_df_balanced).select('features','Stroke')
```

3. Random Forest Classifier on Stroke Prediction:

Random Forest Classifier is used for classification problems which uses the ensembling learning method, that is a combination of many decision trees which can be pruned based on the entropies.

```
# importing RandomForestClassfier and confusion_matrix and other required libraries

from pyspark.ml.classification import RandomForestClassifier

from sklearn.metrics import confusion_matrix

from pyspark.ml.tuning import ParamGridBuilder, CrossValidator

from pyspark.ml.evaluation import MulticlassClassificationEvaluator

# setting paramenters for the grid search for the model

rf_pro = RandomForestClassifier( labelCol='Stroke',seed=7191)

# hypertuning the model by setting the parameters

parameters_Grid_rfc = (ParamGridBuilder()\

        .addGrid(rf_pro.maxDepth,[8,9,10])\

        .addGrid(rf_pro.numTrees,[10,20,30])\

        .build())

# setting evaluator for the model with metrics

evaluator_rfc = MulticlassClassificationEvaluator(predictionCol='prediction',
labelCol='Stroke', metricName='accuracy')

# Creating Cross Validator with 3 folds and other paramters

cv_rfc = CrossValidator(estimator=rf_pro, estimatorParamMaps=parameters_Grid_rfc,
evaluator=evaluator_rfc, numFolds=3)

# training the model with the training data

cvModel_rfc = cv_rfc.fit(balanced_train_data)

# testing the trained model against the unseen testing data
```

```
predictions_rfc = cvModel_rfc.transform(balanced_test_data)
# storing the predictions of the model
evaluator_rfc.evaluate(predictions_rfc)
# reading predicted and oridinal values to create a confusion matrix
predictions_stroke_rfc=predictions_rfc.select("prediction").collect()
original_stroke_rfc=predictions_rfc.select("Stroke").collect()
# creation of confusion matrix
cm_rfc = confusion_matrix(original_stroke_rfc, predictions_stroke_rfc)
# displaying the confusion matrix of the Random Forest Classifier
print("Confusion Matrix produced by the Random Forest Classifier:")
print(cm_rfc)
```

4. Naïve Bayes Classifier on Stroke Prediction:

Random Forest Classifier is used for classification problems which uses the bayes theoem, assuming there is no dependancy between the features.

```
# importing Naivebayes and confusion_matrix and other required libraries
from pyspark.ml.classification import NaiveBayes
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.ml.feature import VectorAssembler
from sklearn.metrics import confusion_matrix
# setting paramenters for the grid search for the model
nb_pro = NaiveBayes(smoothing=1.0, modelType="multinomial")
# training the model with the training data
nb_pro = nb_pro.fit(balanced_train_data)
```

```
# testing the trained model against the unseen testing data
predictions_nbc = nb_pro.transform(balanced_test_data)
# reading predicted and oridinal values to create a confusion matrix
predictions_stroke_nbc=predictions_nbc.select("prediction").collect()
original_stroke_nbc=predictions_nbc.select("label").collect()
# creation of confusion matrix
cm_nbc = confusion_matrix(original_stroke_nbc, predictions_stroke_nbc)
accuracy = (cm_nbc[0][0]+cm_nbc[1][1])/cm_nbc.sum()
# displaying the confusion matrix of the Naive Bayes Classifier
print("Confusion Matrix produced by the Naive Bayes Classifier:")
print(cm_nbc)
```

7. Interface Design / Analytical Outcomes

On performing both Naïve Bayes and Random Forest Classifiers on the Stroke feature prediction, the after performance metrics of both the trained models on testing data are displayed below:

Algorithms	Precision(%)	Recall(%)	F1-Score(%)	Accuracy(%)
Random Forest Classifier	**0.86**	**0.98**	**0.81**	**0.85**
Naive Bayes Classifier	**0.84**	**0.99**	**0.77**	**0.84**

Table 2: Performance comparison of the algorithms on heart disease dataset.

Here, it can be clearly observed that random forest classifier outperformed naïve bayes clssifier in 3 out of 4 performance metrics in classifying the heart storke prediction problem, because of the hyper-tuning pararmeter setting that comes with the random forest classifier, which can be changed to improve the performance metrics.

8. Test Cases

A dummy dataframe is created as an input to test both the Naïve Bayes and Random Forest Classifier models and to their performance on unseen or dummy feature vector. Below is the testing feature vector created and given as an input and the performance metrics of both the classifier models:

```
testing feature vector:
+--------------+-----+
|      features|label|
+--------------+-----+
| [1.0,1.0,3.0]|    1|
| [0.0,0.0,6.0]|    1|
| [1.0,1.0,2.0]|    0|
|[0.0,0.0,10.0]|    1|
+--------------+-----+

performance metrics of the random forest classifier:
+--------------+----------+----------------------------------------+
|features      |prediction|probability                             |
+--------------+----------+----------------------------------------+
|[1.0,1.0,3.0] |1.0       |[0.4342063033813863,0.5657936966186137] |
|[0.0,0.0,6.0] |0.0       |[0.8887261771062628,0.11127382289373726]|
|[1.0,1.0,2.0] |1.0       |[0.36758065846949456,0.6324193415305055]|
|[0.0,0.0,10.0]|0.0       |[0.8808571867359053,0.11914281326409473]|
+--------------+----------+----------------------------------------+

performance metrics of the naive bayes classifier:
+--------------+----------+----------------------------------------+
|features      |prediction|probability                             |
+--------------+----------+----------------------------------------+
|[1.0,1.0,3.0] |1.0       |[0.433823021009113,0.566176978990887]   |
|[0.0,0.0,6.0] |0.0       |[0.8765760665991316,0.12342393340086846]|
|[1.0,1.0,2.0] |1.0       |[0.42026136266628655,0.5797386373337134]|
|[0.0,0.0,10.0]|0.0       |[0.8986323982106772,0.10136760178932273]|
+--------------+----------+----------------------------------------+
```

Fig 10: Performance metrics of both the classifier trained models by testing with dummy feature vector.

Here, it can be seen that both of the classifiers predicted the heart storke identically with the given feature vector, so that both the models does through any abnormal predictions or errors while being performed on any unseen feature vectors or dataset.

9. Conclusion

As per the dataset, the prediction is done on the feature Stroke (Heart Stroke), whether or with how much probability does a person can expect a heart stroke in the near future depending on various other effecting features like previous Heart Disease, Kidney Disease, Skin Cancer, Difficulty in walking, etc. On pre-processing the data and with the help of visualizations and pair-wise correlation between all the features and on the Stroke feature, a feature vector is created using feature transformation: ['HeartDisease','DiffWalking','AgeCategory'] as it is passed into machine learning models. Random Forest Classifier and Naive Bayes Classifier are the two machine learning models which have been selected to classify the Stroke feature (0 or 1), as they are the model which perform their best when comes to the classification problems.It can be observed that the accuracy provided by the Random Forest Classifier, i.e. around 85%, whereas the accuracy provided by the Naive Bayes Classifier i.e. around 84%. Both the models performed close to one another, but Random Forest Classifier clearly well suited for the existing problem of Stroke Classification.

10. References

1. Heart Disease Dataset (https://www.kaggle.com/datasets/vaisakhnair/heart-disease-data)

2. Stroke Risk Prediction with Machine Learning Techniques (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9268898/)

3. Learn about Stroke. (https://www.world-stroke.org/world-stroke-day-campaign/why-stroke-matters/learn-about-stroke)

4. Elloker T., Rhoda A.J. The relationship between social support and participation in stroke: A systematic review. Afr. J. Disabil. 2018;7:1–9. (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6191741/)

5. Katan M., Luft A. Seminars in Neurology. Volume 38. Thieme Medical Publishers; New York, NY, USA: 2018. Global burden of stroke; pp. 208–211. (https://pubmed.ncbi.nlm.nih.gov/29791947/)