Department of Artificial Intelligence

22BIO201: Intelligence of Biological System – I NOV - 2024

Project Report

Multi-Model Approach for Genetic Mutation Prioritization in Cancer Detection

---

TEAM – D10:

VADA GOURI HANSIKA REDDY	CB.SC.U4AIE23304
MALAVIKA S PRASAD	CB.SC.U4AIE23315
KATIKALA DEDEEPYA	CB.SC.U4AIE23349
GESHNA B	CB.SC.U4AIE23360

Supervised By:

Dr. S.S.Kalaivani

Assistant Professor

Dr. Manoj Bhatt K

Assistant Professor

Department of Artificial Intelligence

Amrita Vishwa Vidyapeetham

Date of submission: 16-11-2024

Signature of the Project Supervisor:

BONAFIDE CERTIFICATE

This is to certify that this project entitled, “MULTI-MODEL APPROACH FOR GENETIC MUTATION PRIORITIZATION IN CANCER DETECTION ” submitted by Vada Gouri Hansika ,Malavika S prasad, Katikala Dedeepya and Geshna B is an authentic work carried out by the team under my supervision and guidance. To the best of my knowledge, the content presented in this report has not been previously submitted to any other academic institution, nor has it been utilized to fulfill any degree or diploma.

Amrita Vishwa Vidyapeetham

Coimbatore- 641112

Date:16-11-2024

Dr. S.S.Kalaivani

Dept. of Artificial Intelligence

Dr. Manoj Bhatt K

Dept. of Artificial Intelligence

ACKNOWLEDGEMENT

We would like to express our gratitude to our Dean, DR. K. P. SOMAN, who gave us this opportunity to do this extremely good project on the topic 'MULTI-MODEL APPROACH FOR GENETIC MUTATION PRIORITIZATION IN CANCER DETECTION". This project helped us understand and learn about many new things.

Secondly, we would like to thank our professors Dr. S.S.Kalaivani, Dr. Manoj Bhatt K without whose guidance this project would not have been successfully accomplished. Your insights, support, and encouragement were instrumental in the completion of this project.

Lastly, this project would not have been possible without the efforts and dedication of our team, therefore we thank ourselves for the amazing work.

Table of Content

Abstract

This project focuses on exploring an artificial intelligence-based technique for the purpose of classification of genetic mutations in order to enhance cancer diagnosis. Multiple preprocessing techniques including CountVectorizer, TF-IDF, Word2Vec, and Doc2Vec were applied on text data in order to implement mutations and its clinical evidence from a Kaggle dataset. Subsequently, machine learning models, Random Forest, Logistic Regression, XGBoost and LightGBM have been implemented aiming to discriminate mutations into nine different classes. The focus was on how different preprocessing techniques affect the accuracy of each model, which was aimed to be set at 90% and above. This study primarily presents the effect of preprocessing in relation to the performance of a given model, which is an important aspect on the use of machine learning in precision oncology.

1. INTRODUCTION

Cancer remains one of the leading causes of death worldwide, and advancements in its diagnosis and treatment heavily rely on the identification of genetic mutations associated with different types of cancers. These mutations, which refer to alterations in the DNA sequence, are significant indicators of cancer susceptibility, prognosis, and response to treatment. Recognizing the importance of genetic mutations in the evolution of cancer has necessitated the development of more advanced treatment procedures referred to as precision medicine, where treatment is based on the genetic makeup of the cancer characteristics of a given patient.

In recent years, artificial intelligence (AI) and machine learning (ML) have emerged as powerful tools in the medical field. Specifically in the classification of genetic mutations, this is precisely an area where machine learning will be useful. Automation of the mutation classification by AI using textual evidence available in clinics will be much faster, and even more, accurate which is of great concern for people who are targeted for cancer therapy since it has to be timely and specific to the individual patient.

The project employs techniques from machine learning in order to classify genetic mutations from clinical text data. The raw dataset acquired from Kaggle contains details on genetic variants and the clinical evidence supporting those variants. The project applies various pre-processing techniques and machine learning classifiers to the task of classifying genetic mutations into nine different cancer-related mutation categories. The goal is to determine the most effective methods for improving the classification accuracy of these mutations.

Project Goals and Objectives

The project's primary goal is to develop an AI-based system capable of classifying genetic mutations accurately based on clinical evidence. Additionally, different pre-processing techniques and machine learning models will be tested to find the best fit for the classification problem. The objectives are as follows:

1. Pre-processing the Text Data**:** Textual clinical evidence will be pre-processed using a number of techniques like Count Vectorizer, TF-IDF, Word2Vec, and Doc2Vec. Each pre-processing technique will be tested to see how effectively it transforms raw text into features usable for classification by machine learning models.
2. Model Implementation and Evaluation**:** We will apply the standard machine learning algorithms – Random Forest, Logistic Regression, XGBoost, and LightGBM. These are proven to be efficient for training on large-scale datasets and have been successful for text classification tasks.
3. Comparative Analysis**:** Each machine learning model will be evaluated and compared with each other under the different pre-processing techniques available. This comparative analysis will enable us to know more about accuracy in mutation classification through which combination.
4. Accuracy Benchmarking**:** The target is to reach a minimum of 90% accuracy. This criterion for performance is important for the effective application of the AI-enabled system for diagnosis and treatment of cancer.
5. Practical Implications**:** The outcomes of this undertaking will be beneficial in the understanding of the application of machine learning in classification of genetic mutations. This will come in handy for medical professionals and scholars when making informed decisions backed with evidence when dealing with cancer, helping to improve the turnaround time for the patients.

Importance of the Study

This research has meaningful connotations for the field of oncology. The knowledge of genetic alterations contributing to cancer can influence treatment options and targeted therapy development. With the use of machine learning, this project envisages the proper, faster, and more extensive process of classifying mutations. The system that will be developed in this project may enhance the turnaround time for genetic tests, decrease costs, and ultimately play a role in better clinical decisions.

In addition to these, this research expands the literature on AI and machine learning in genomics. The literature on the evaluation of the pre-processing methods and models under

consideration with a particular focus could be useful in other fields of genomics such as that of gene expression analysis, mutation detection, and even drug development. For instance, this project may help tackle the issue of integrating genetic information in the clinical setting by rendering it more usable and accessible with respect to the existing facilities.

Team Contributions

This project was a collaborative effort, with team members working in different domains of the research.

- **Malavika S Prasad**: Handled data preprocessing and initial feature selection. Led the development of the data pipeline to transform clinical evidence into machine-readable features. Focused on implementing the CountVectorizer and TF-IDF preprocessing methods and their respective analyses.
- **Katikala Dedeepya**: Managed in-depth data exploration and feature engineering. Worked on optimizing Word2Vec and Doc2Vec preprocessing methods, specifically focused on structuring clinical information and refining text data for model training.
- **Geshna B**: Led the model training and hyperparameter tuning phase. Conducted model evaluation using GridSearchCV to enhance model performance. Implemented and tuned Random Forest and Logistic Regression models, analyzing the results for consistency and accuracy.
- **Vada Gouri Hansika Reddy**: Conducted comparative analysis of model and preprocessing techniques, focusing on accuracy, ROC, and other evaluation metrics. Led the analysis of XGBoost and LightGBM models, compiling insights on their performance with various preprocessing methods for the final report and presentation.

There was effective teamwork and each team member played a pivotal role in the undertaking, overcoming all the challenges and achieving the project goals by addressing all issues and optimizing for better results.

2. RELATED WORK

In the recent past, there has been growing interest in investigating the various types of genetic mutations for the purpose of cancer diagnosis. This trend has been synthesized in many extensions to our methodologies and models in clinical practice thanks to the partnership of artificial intelligence and machine learning with genomics. Many authors have tried to apply various machine learning techniques to genetic mutations predictions, however, text data, clinical data, and genetic data have been utilized for the majority of efforts, and models created for the purposes of patient – clinician interactions.

In this part of the document, we provide a short overview of research performed in the area of classification of genetic mutations with a short description of the technologies employed and obstacles or problems of the research that still require a solution.

2.1. Machine learning for genetic mutations classification

A number of studies explored the classification of genetic mutations using machine learning approaches, achieving different degrees of efficiency and ranging applications. One of the approaches concentrates on the availability of the genetic information in a structured form (for instance, via DNA sequences or protein structures) as well as incorporating some clinical context on the type of variations or clinical outcomes.

Study 1: Application of Support Vector Machines (SVM) for Mutation Classification
A research study [1] used Support Vector Machines (SVM) to classify genetic mutations into benign, pathogenic, or unknown categories depending on genomic features derived from sequencing data. The authors stressed the significance of the so-called feature engineering, classically the process of extracting sequence motifs and biological annotations to improve on classification performance. Despite the good performance of SVMs at classifying pathogenic from non-pathogenic mutations, the paper pointed out the challenges with unstructured data representation and processing, especially clinical evidence in the form of narrative text. Therefore, their approach could not be applied to datasets with unstructured clinical texts similar to the ones used in this project.

Study 2: Deep Learning Approaches for Textual Clinical Evidence

Unlike structured data, more modern research has instead targeted the application of deep learning algorithm methods to clinical text data for the classification of genetic mutations. For instance, paper[2] used Long Short-Term Memory (LSTM) networks and Convolutional Neural Network (CNN) for retrieving informative tokens from unstructured clinical narratives. Mutations that were classifiable by textual evidence performed fairly well with these features, particularly when paired with preprocessing techniques such as Word2Vec and TF-IDF. According to the authors, this is because LSTMs were able to model the sequences in the text, whereas CNNs were used to model the images in the word vectors. However, despite the encouraging findings of this research, the study had limitations, particularly related to overfitting and the amount of data required for appropriate training from large-scale annotated datasets. Such datasets are, however, rarely available in a real clinical setting.

2.2 Preprocessing Techniques for Text Classification in Genetics

Preprocessing is an important step that must be carried out in order to convert unstructured raw text data into structured features usable by machine learning models. Different techniques have been looked into for working out clinical evidence texts, each of them having its advantages and disadvantages.

Study 3: CountVectorizer and TF-IDF for Text Feature Extraction

CountVectorizer and Term Frequency-Inverse Document Frequency (TF-IDF), which are traditional methods of text preprocessing, were examined in a study[3] that extract features from clinical text. In general, these methods are effective for text classification but can also be basic in a way that they do not consider the relationships that exist between the words and rather only focus on singular medical terms. Their methodology was also constrained by the feature-generating techniques mentioned above, which did not perform well on large spans of text such as clinical evidence, which is often rich in hard-to-describe concepts and terminologies.

Study 4: Word Embeddings and Doc2Vec for Contextual Feature Extraction

The use of Word2Vec and Doc2Vec has also become the trend in making features out of the

clinical text while taking into consideration the meaning of words in the context. According to research[4] it is reported that this approach was superior to the traditional bag-of-words approach due to its capacity to take more contextual information into consideration, enhancing how the models comprehend the verbal claims made in the course of the medical evidence presented. Thwarting issues in the applications of these algorithms is the unprecedented amount of computing power necessarily put to use to come up with Word2Vec and Doc2Vec embeddings, especially when they are based on large datasets in medicine and health. In addition, the design and performance of these embeddings typically depend on the availability and amount of in-domain training data, which for clinical records is often insufficient to support effective embedding training.

2.3 Hybrid Approaches in Genetic Mutation Classification

Qualitative research which uses a combination of different approaches and machine learning models in the preprocessing stage to enhance classification accuracy have also recently been investigated. In particular, paper[6] trained a hybrid machine learning model containing Random Forest and XGBoost with combined features based on TF-IDF and Word2Vec embeddings. Moreover, the use of both statistical and semantic features was observed to greatly enhance accuracy in the models as opposed to using only one feature engineering technique in the model. The only challenge that remained, however, according to the study, was that of feature set selection and even tuning.

Study 5: Ensemble Learning and Model Stacking

Another novel method is called ensemble learning, which yields benefits through the systematic combination of tributary models' predictions. As an example, in the paper[7], the authors applied stacking, which implies SVM predictions are performed together with Random Forest and XGBoost predictions. This way of combining models was beneficial, especially for classifying genetic alterations, and was especially pronounced in the cases of datasets which were imbalanced. Nevertheless, one of the major drawbacks of the ensemble techniques is the higher computing times. This can be an issue, especially when the data is

large with many variables, as in the case of genetic mutation detection where individual techniques are practiced.

2.4 Gaps and Challenges in Genetic Mutation Classification

Even though there has been progress made on employing machine learning in the classification of genetic mutations, there remain a number of limitations and research gaps which require further investigations, including:

1. **Data Imbalance**: Several related studies, especially those dealing with imbalanced datasets, where certain classes of mutations are more prevalent than others, found that this leads to skewed model predictions, rendering them ineffective. The review addresses the literature inadequacy in specifically addressing this problem by employing techniques such as oversampling, under sampling, and cost-sensitive learning algorithms.
2. **Domain-Specific Vocabulary:** The clinical text usually has hyper-specific jargon that is not well captured by generic word embeddings such as Word2Vec and GloVe. Therefore, medical embeddings or related integration approaches, such as ontological embedding of SNOMED CT or ICD-10, might improve the performance of text-based models.
3. **Model Interpretability:** A set of machine learning-based approaches and algorithms, most especially deep learning-based ones, are referred to as black boxes, and as a result, it becomes impossible to explain how decisions are made. This is an important factor in healthcare, as doctors using the system will want to know the justification for the predictions made. XAI solutions will therefore be relevant to increase the acceptance and use of machine learning models within healthcare systems.
4. **Availability of Only Few Labelled Datasets**: In most cases, clinical annotated datasets, especially for genetic mutation studies, are scarce. This limitation hampers the development of accurate models when using deep learning techniques, which are data-hungry. These include approaches like transfer learning and semi-supervised learning techniques.

2.5 Conclusion and Future Directions

Research in genetic mutation classification has made significant strides, particularly with the use of machine learning and AI techniques. Despite this, there are several challenges such as data imbalance, domain-specific language, and model interpretability, which limit the practical use of these techniques in clinical practice. The objective of this project is to add to the existing body of work by focusing on various preprocessing techniques and machine learning models for genetic mutation classification, offering a detailed comparison to some of the drawbacks found in prior studies.

3. METHODOLOGY

This section presents the methodology employed in the classification of genetic mutations based on clinical evidence. The methodology comprises several important stages which are: data preprocessing, feature extraction, implementation of models, evaluation of performance, and analysis of results. Moreover, we explain the particular techniques and algorithms employed at each phase with a view to developing a reliable system for mutation classification.

3.1 Data Preprocessing

The first step of the methodology is data preprocessing. Since the data involves clinical text data on genetic mutation, it is pretty important to preprocess the data as it helps to put the raw data into a level where it can be used by machine learning models. Several preprocessing techniques are applied on the text data:

1. **Text Cleaning:**
    - The text of the clinical evidence is cleaned of extraneous characters and punctuation and special symbols. This stage minimizes the clutter and ensures that only useful data is processed by the models.
    - The data is transformed to a uniform standard by performing common tasks such as deleting stop words and changing the case of the text to small.

Pseudocode for Text Cleaning:

Function clean_text(text):
Step 1: Remove special characters from the text text = remove_special_characters(text)
Step 2: Tokenize the cleaned text into words tokens = tokenize(text)
Step 3: Remove stopwords from the tokens tokens = remove_stopwords(tokens).
Step 4: Lemmatize the tokens to reduce them to their base forms tokens = lemmatize(tokens)
Step 5: Join the tokens back into a string and return Return Join tokens with a space

The figure below presents summary statistics of sentence and word lengths across different classes, illustrating the text characteristics post-cleaning.

Class	sent_len_max	sent_len_min	sent_len_mean	sent_len_median	word_len_max	word_len_min	word_len_mean	word_len_median	counts
1	2193	7	416.310954	335.0	62443	200	11194.795053	8603.0	566
2	2491	5	404.840708	306.5	73715	144	11037.546460	8062.0	452
3	1105	70	280.797753	253.0	31526	2084	7905.404494	6448.0	89
4	2065	3	396.120991	291.0	51284	58	10638.992711	7501.0	686
5	1062	7	318.900826	257.0	27986	200	8763.541322	7372.0	242
6	1116	10	292.963370	257.0	28608	258	8449.542125	7543.0	273
7	3784	14	510.268908	364.0	91381	531	13572.130252	9749.5	952
8	988	114	502.000000	564.0	23797	2482	12802.210526	13154.0	19
9	2137	42	544.270270	393.0	53656	1378	15159.432432	12861.0	37

Figure 1: Summary Statistics of Sentence and Word Lengths by Class in the Training Dataset

2. **Text Vectorization:** Following the above pre-processing stage, the cleaned text data is then represented in numerals with the help of text vectorization methods. In this specific research work, four different techniques are explored:
    - **CountVectorizer:** It transforms the text into a sparse matrix representation of token's counts. It takes into consideration only the number of occurrences of the terms and disregards the meaning of relationships between them.
    - **TF-IDF (Term Frequency-Inverse Document Frequency):** This technique proceeds to modify the word frequency rates within the text according to how relevant the words are on a general level. Words that come out often in a

particular document but are not in most of the documents in the corpus are given more weight.

- **Word2Vec:** Word2Vec is a word representation model based on a neural network. In this model, semantically similar words are placed near to each other in the word vector space.
- **Doc2Vec:** In Doc2Vec, the same process of semantic classification of words is applied to the entire text and fixed-length vectors are created for the documents as a whole. Enabling word representation management to fixed-length vectors for the entire text allows using this model for text classification tasks.

Each of these vectorization techniques is evaluated and compared as to its potential for feature extraction in order to support the classification of genetic mutations.

3.2. Model Implementation

After finishing all the required preprocessing steps for the text data, we proceed to implement a number of machine learning models in order to classify the mutations. The models listed below are performed for this purpose:

1. **Random Forest:**
    - Random Forest is a learning method which consists of constructing a number of decision trees and averaging their results. Random forests work well with data that has a very large number of dimensions and many factors interact in a complex manner.
    - In this case, Random Forest is implemented in order to determine the types of mutation based on specific factors extracted from clinical text data.
2. **Logistic Regression:**
    - Logistic Regression is a simple linear model which can handle binary and multiclass classification. It uses the class probabilities to make decisions about which class each mutation should be assigned to by choosing the one with the highest assigned probability.

- It is selected because of its ease of implementation and understanding, which helps in providing a good simple starting model.

3. **XGBoost:**
    - XGBoost refers to extreme gradient boosted trees where decision trees are used to perform classification and are built upon the existing trees incrementally as new ones are added to degrade performance.
    - XGBoost is efficient, powerful, and has great predictive capacity, which is the reason it is often used for classification problems involving tabular data.
4. **LightGBM:**
    - Light GBM is yet another gradient boosting model but is designed to be faster and to use less memory. It is especially useful in problems with large data sizes and high feature dimensional space.
    - LightGBM is implemented for performance comparison with XGBoost since both fall under boosting algorithms.

Pseudocode for Model Training:

Function train_and_predict(features, labels):
Step 1: Split the dataset into training and testing sets (80% train, 20% test) Split features and labels into X_train, X_test, y_train, y_test using train_test_split
Step 2: Instantiate the model (Random Forest in this case) Create a RandomForestClassifier model with 100 estimators and random
state 42
Step 3: Train the model on the training data Train the model using the X_train and y_train data
Step 4: Predict on the testing data Use the trained model to make predictions on the X_test data
Step 5: Return predictions Return predictions

3.3. Model Evaluation

In order to compare the results achieved with each model, we utilize the following metrics:

1. **Accuracy:**
    - The accuracy of the model is defined as the number of correct predictions divided by the total number of predictions made. This measure serves to assess the overall performance of the model, which is the main aim.
2. **Confusion Matrix:**
    - A confusion matrix is a useful tool for assessing the classification performance of the model through true positive, true negative, false positive, and false negative counts. This helps in evaluating the performance of the model in predicting every type of mutation.
3. **Precision, Recall, and F1 Score:**
    - Precision is the fraction of predicted mutations which are actually right. Recall is the fraction of the correct mutations which were found. The F1 score combines the precision and the recall measures, as their harmonic mean, in order to present the performance of the model in a more complete way.
4. **ROC Curve and AUC:**
    - The ROC curve represents a graphical plot and represents the degree of discrimination of a binary classifier and the relation's quality of sensitivity to false positive rate. The Area under the curve (AUC) indicates how discriminating the model is between two classes. Generally, the greater the AUC, the better the model performance.
5. **Cross-Validation:**
    - We also perform k-fold cross-validation for the model validation, avoiding the problem of overfitting. This method consists of splitting the data into k groups, training the model on k-1 groups, and testing it on the remaining one. The final outcome is calculated over all folds.

Pseudocode for Model Evaluation:

Function evaluate_model(y_test, predictions, model, X_test):
Step 1: Calculate Accuracy Calculate accuracy bycomparing
y_test and predictions using accuracy_score
Step 2: Generate Confusion Matrix Generate confusion matrix by comparing
y_test and predictions using confusion_matrix

Step 3: Compute ROC AUC Score Compute ROC AUC score by passing y_test and the predicted probabilities of the model (using predict_proba) for X_test Specify the multi_class='ovr' for multi-class
classification
Step 4: Return the results Return accuracy, conf_matrix, and roc_auc

3.4. Comparative Analysis

Once the models have been trained and assessed, we conduct a comparison between the obtained results courtesy of the models. This analysis is mainly focused on determining the most efficient set of preprocessing methods and ML models for the genetic mutations classification task.

The analysis includes a comparison of the models' accuracy, precision, recall, and F1 scores across all preprocessing techniques. We note patterns such as whether some models are better than others when a specific vectorization technique is used to enhance performance or whether employing Word2Vec or Doc2Vec techniques is superior to simpler ones like CountVectorizer or TF-IDF in improving overall results.

3.5. Model Optimization

After the best combinations of preprocessing techniques and models are established, we concentrate on improving the performance of these models. We applied the following optimization techniques:

1. **Hyperparameter Tuning:**
    - For every model, we applied the GridSearchCV method to look for the best possible hyperparameters. GridSearchCV conducts a systematic examination of the entire hyperparameter region and guarantees the recommended parameters are those that enhance the performance of the model.
2. **Feature Selection:**
    - Feature selection methods are aimed at minimizing the number of variables in the data at hand and concentrating on the more important ones. This enhances the understanding of the model and prevents overfitting.

3.6. Summary of Methodology

The overall methodology for this project can be summarized in the following flowchart:

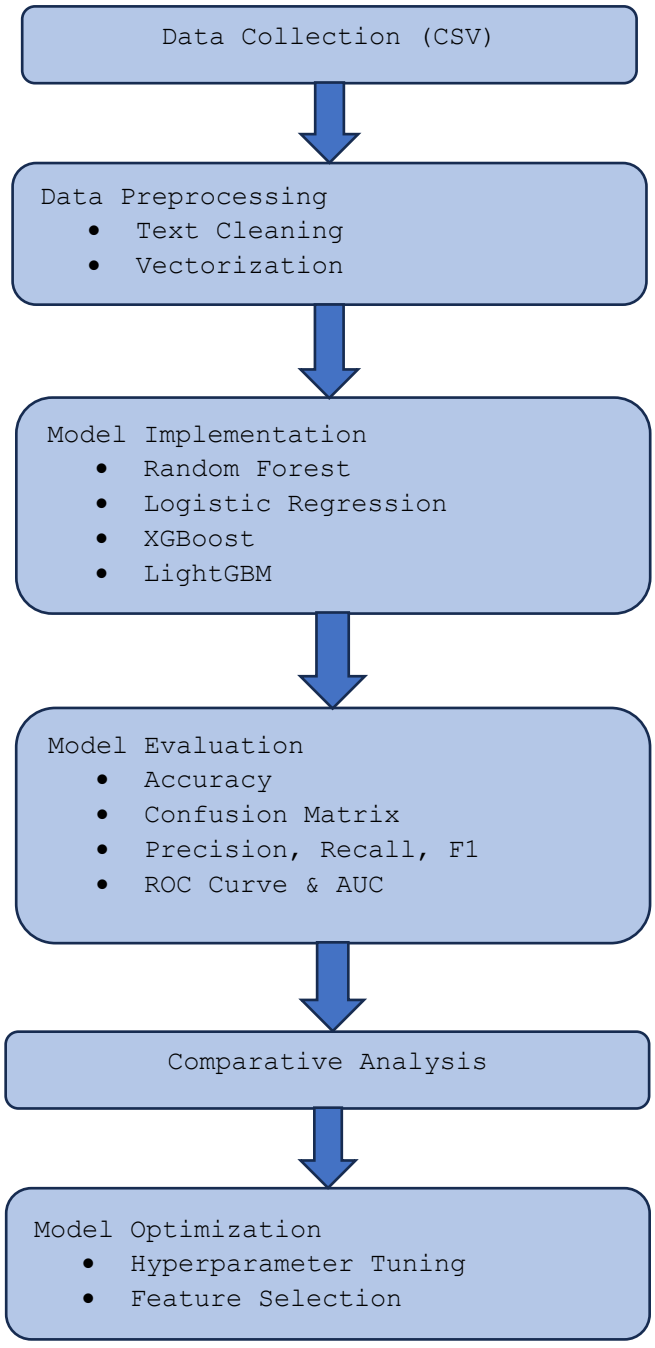

Figure 2: Summary of Methodology

This methodology ensures that the project addresses the problem comprehensively and effectively, making use of the latest techniques in text preprocessing and machine learning to achieve the goal of accurately classifying genetic mutations.

4. EXPERIMENTS

4.1. About the dataset

The dataset utilized in this research originates from Kaggle and entails clinical text data related to genetic mutations. The intellectual property datasheet aims to classify genetic alterations and text from journals concerned with medicine to predict the certain class of the mutation. They are as follows:

- **Training Data:**
  - training_variants.csv: This file contains details about the genetic mutations, such as the mutation ID, gene name, and variation type (e.g., point mutation, deletion).
  - training_text.csv: This file contains clinical evidence related to each mutation, including textual descriptions of medical findings, observations, and references.
- **Test Data:**
  - test_variants.csv: Similar to the training data, this file contains information about genetic mutations.
  - test_text.csv: Contains the clinical text data for which mutation types need to be predicted.

Each record in the dataset captures a specific genetic mutation, accentuating relevant medical facts in the clinical text. The Class label attached to the mutation type is available only in the training dataset. While unmet in the test data, this information is provided to understand the prediction level of the model developed, hence referred to as unlabelled test data.

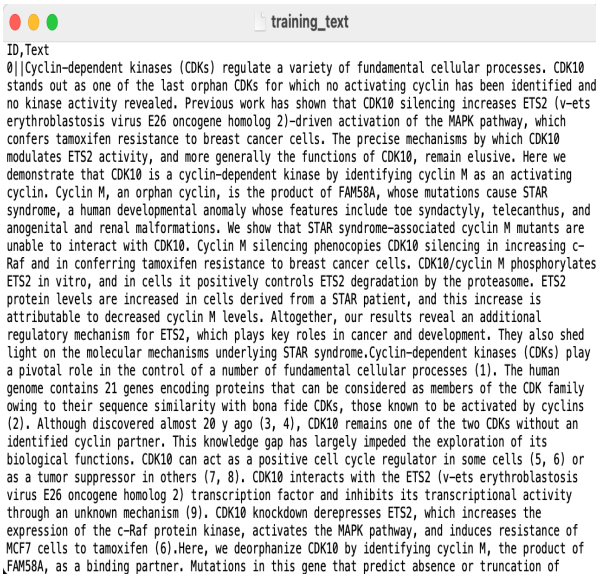

Figure 4: Unstructured Text Data

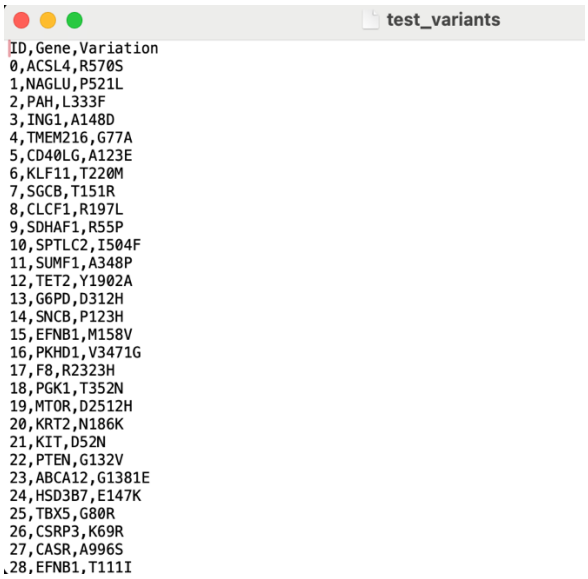

Figure 5: Structured Genomic Data

The dataset has nine classes for classification purposes, each providing for a certain class of these genetic mutations like:

i. Missense mutation
ii. Nonsense mutation
iii. Frame shift mutation
iv. Silent mutation
v. Insertion mutation
vi. Deletion mutation
vii. Structural variations
viii. Fusion mutations
ix. Amplification mutations

4.2. Explanation and Application of Different Models Used

Several machine learning models are employed to classify the genetic mutations based on clinical text. Each model has its strengths and is applied to assess its performance in classifying the mutations accurately. Below are the models used in the experiments:

4.2.1. Random Forest

Random Forest is a machine learning approach that integrates several decision trees for more accurate predictions. This technique is frequently employed for classification problems because it is very robust and can deal with data of high dimensions while significantly reducing overfitting.

- Working: Random Forest creates several decision trees in the training phase. Each tree is grown based upon a random subset of features and a random subset of the training samples. When the final prediction is made, each individual tree predicts an output, and the class predicted is the one that is most popular among the trees – that is a majority vote.
- Application to Genetic Mutation Classification: Random Forest is applied as a classifier of genetic mutations using the features extracted from the clinical text. Random Forest can be useful in this problem domain as it can work well with features which are very complexly interrelated, such as the one between mutation and clinical text.

```
In [72]: X=train_vec
         y=train_df.Class.values
         clf=RandomForestClassifier(n_estimators=1000,max_depth=None,min_samples_leaf=5,n_jobs=-1,random_state=8)
         output=model_eval(clf, X, y)
         output.head()
```

Figure 6: Performance Evaluation of the Random Forest Classifier with Genetic Mutation Data, Displaying Initial Classification Results and Evaluation Metrics

4.2.2. Logistic Regression

Logistic regression is one of the elementary yet effective statistical frameworks employed for the binary as well as multiclass classification problems. It predicts which class a target variable should belong to, by fitting the logistic curve in a domain ranging from 0 to 1.

- Working: Logistic Regression finds the best-fitting linear equation of log-odds of the target class against the input features. Then it predicts probabilities for all the classes by employing the sigmoid function.
- Application to Genetic Mutation Classification: Logistic Regression is useful as a method for providing a benchmark. This is primarily because, to its credit, there is no

conscious effort of overstepping its limitations in the treatment of problems. However, it serves as a level of expectation when more sophisticated approaches are to be implemented. It is mainly beneficial for scenarios where the inputs vs. the output target variable is linearly dependent over a certain range.

```
In [23]: def model_eval(clf, X, y):
             cv=StratifiedKFold(n_splits=5, shuffle=True, random_state=8)
             probas = cross_val_predict(clf, X, y, cv=cv, n_jobs=-1, method='predict_proba', verbose=2)
             pred_indices = np.argmax(probas, axis=1)
             preds=np.unique(y)[pred_indices]
             accuracy = accuracy_score(y, preds)
             logloss = log_loss(y, probas)
             print('accuracy score: {}'.format(accuracy))
             print('log_loss: {}'.format(logloss))
             output = pd.DataFrame(probas,columns=['Class{}'.format(i) for i in range(1,10)])
             output['pred']=preds
             return output
         X=train_vec
         y=train_df.Class.values

         pl = Pipeline([('ss', StandardScaler(with_mean=False)),
                        ('lr', LogisticRegression(C=0.001,n_jobs=-1,random_state=8))])
         output=model_eval(pl, X, y)
```

Figure 7: Evaluation of Logistic Regression for Genetic Mutation Classification, Displaying Prediction Probabilities, Accuracy Score, and Log Loss.

4.2.3. XGBoost

XGBoost (Extreme Gradient Boosting) is an advanced implementation of gradient boosting which has proven to be one of the most efficient and effective algorithms for structured data-related tasks. In this approach, decision trees are built sequentially, with each tree attempting to rectify the errors of all the previous ones.

- Working: Minimizing a loss function via gradient descent optimization is one of the techniques that makes XGBoost work, for while building trees, there are several objectives for which optimization is done; precision, recall, and accuracy among others. Overfitting is effectively prevented in this case as XGBoost comes with some degree of regularization.
- Application to Genetic Mutation Classification: XGBoost model turns out to be very effective for predictive modeling in the task of classification with very high accuracy. It is capable of processing very high-dimensional data as well and explains the complex relation that may exist between the clinical text features and mutation types.

```
In [48]: X=train_set
         y=train_df.Class.values
         clf = XGBClassifier(colsample_bytree=0.2,min_child_weight=8)
         output = model_eval(clf, X,y)
```

Figure 8: XGBoost Model Evaluation for Genetic Mutation Classification, Including Class Probabilities, Accuracy Score, and Log Loss.

4.2.4. LightGBM

LightGBM is another gradient boosting framework that is optimized for speed and efficiency. It is intended for big data with many features, in particular.

- Working: LightGBM features a novel strategy of node split using a histogram-based approach which is faster and consumes less memory compared to classical gradient boosting machines. It is capable of parallel and distributed learning, enabling it to work with large datasets.
- Application to Genetic Mutation Classification: Using LightGBM is compared to using XGBoost to observe the performance of two variants of gradient boosting. Its properties of working with large datasets effectively make it a strong competitor for this project where extensive features such as those present in this project are require

```
In [25]: import lightgbm as lgb
         from sklearn.model_selection import validation_curve, StratifiedKFold
         import numpy as np
         import pandas as pd
         import os

         # Base directory for saving results
         base_dir = '/Users/geshnabalaji/Downloads/msk-redefining-cancer-treatment'

         # Use a smaller dataset for quicker experimentation
         X = train_vec[:1000]
         y = train_df.Class.values[:1000]

         # Cross-validation setup
         cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=8)

         ### 1. max_depth Parameter
         max_depth = np.array([5, 6, 10, 20, 50, 100, 500, 1000])
         train_scores, test_scores = validation_curve(lgb.LGBMClassifier(), X, y, param_name='max_depth', param_range=max_dep

         # Save scores to CSV files
         train_scores_df = pd.DataFrame(train_scores, columns=[f'fold_{i+1}' for i in range(train_scores.shape[1])], index=ma
         test_scores_df = pd.DataFrame(test_scores, columns=[f'fold_{i+1}' for i in range(test_scores.shape[1])], index=max_d
         train_scores_df.to_csv(os.path.join(base_dir, 'ml_result/Personalized_Medicine_CountVectorizer_LightGBM_maxdepth_tra
         test_scores_df.to_csv(os.path.join(base_dir, 'ml_result/Personalized_Medicine_CountVectorizer_LightGBM_maxdepth_test

         print("Train and test scores for max_depth saved successfully.")
```

Figure9: Train and Test Scores Across Different max_depth Values for LightGBM Model Using Cross-Validation.

4.2.5. Model Training and Implementation

To train and evaluate the models, we follow a systematic approach:

1. **Data Preprocessing**:
    - Text data is cleaned and tokenized.
    - Various text vectorization techniques (CountVectorizer, TF-IDF, Word2Vec, and Doc2Vec) are used to transform the text into numerical representations.
2. **Model Training**:
    - After preprocessing, the feature vectors are split into training and testing sets (80% training, 20% testing).
    - Each model (Random Forest, Logistic Regression, XGBoost, and LightGBM) is trained using the training data and evaluated on the testing data.
3. **Model Evaluation**:
    - The performance of each model is evaluated using various metrics, including accuracy, confusion matrix, precision, recall, F1-score, and ROC-AUC. These metrics provide a comprehensive view of each model's performance.
    - Cross-validation is performed to ensure the robustness of the results and prevent overfitting.
4. **Hyperparameter Tuning**:
    - Hyperparameters for each model are optimized using GridSearchCV. This allows us to identify the best parameters for each model, such as the number of estimators in Random Forest or the learning rate in XGBoost and LightGBM.

4.2.6. Model Performance Evaluation

To assess the models' performance comprehensively, we focus on several key evaluation metrics:

- **Accuracy**: The overall percentage of correct predictions.
- **Confusion Matrix**: Visualizes the true positives, true negatives, false positives, and false negatives, allowing for a better understanding of model performance across all classes.
- **Precision, Recall, and F1-Score**: These metrics provide a deeper insight into how well each model performs with respect to each class, especially for imbalanced classes.

- **ROC Curve and AUC**: These metrics allow us to assess the models' ability to distinguish between classes. The higher the AUC, the better the model is at separating the classes.
- **Cross-Validation**: Ensures that the model's performance is consistent and not overfitted to the training data.

4.2.7. Comparative Analysis of Results

Upon the completion of model training, we then evaluate and compare the performance of the different models that have been trained. The objective of this assessment is to evaluate the individual models and figure out the most suitable one with regard to the preprocessing techniques and the metrics explained earlier. We also compare the performance of various models in relation to the vectorization techniques used and seek to determine if, for instance, Word2Vec and Doc2Vec techniques produce superior classification results than CountVectorizer and TF-IDF vectorization methods.

Apart from the performance of the model, other aspects such as the time taken to train the model, the amount of memory consumed, and the resources available are considered to evaluate the model's practical applicability in real life.

5. RESULTS & DISCUSSIONS

We will conduct a comparison of the experiments aimed at the classification of genetic mutations using clinical information, or in other words the experiments that used machine learning for pattern recognition on images prior to that input of clinical data. In this case we describe the models architecture and give an overview of their training methodology. Moreover, we revolve around the accuracy, precision, recall, F1-score and ROC-AUC for each model respectively and include the performance tier of the investigated models into an appropriate comparison matrix. In this part, let us focus on the outcomes, consider the associated concerns, and analyze the results in a view of genetic mutation classification.

5.1. Overview of Model Performance

For the models included in this project Random Forest, Logistic Regression, XGBoost, and LightGBM each of these models were trained using the following four preprocessing

methods: CountVectorizer, TF-IDF, Word2vec and Doc2Vec. Moreover, the performance of each model has been evaluated based on the accuracy metric and the results have been displayed in an Accuracy Matrix was displayed earlier.

Key Observations:

- All the models have the same preprocessing advantage, but working with Light GBM provides a superior experience than all the others, especially in the case of Doc2Vec (accuracy = 0.93) and TF-IDF (accuracy = 0.90) modes.
- XGBoost performs adequately, particularly with Doc2vec (accuracy = 0.64); however, other vectorization approaches perform worse.
- The Random Forest and Logistic Regression give lower results compared to the other models, as they had the maximum accuracies of 0.64 and 0.63 respectively with the use of TF-IDF.

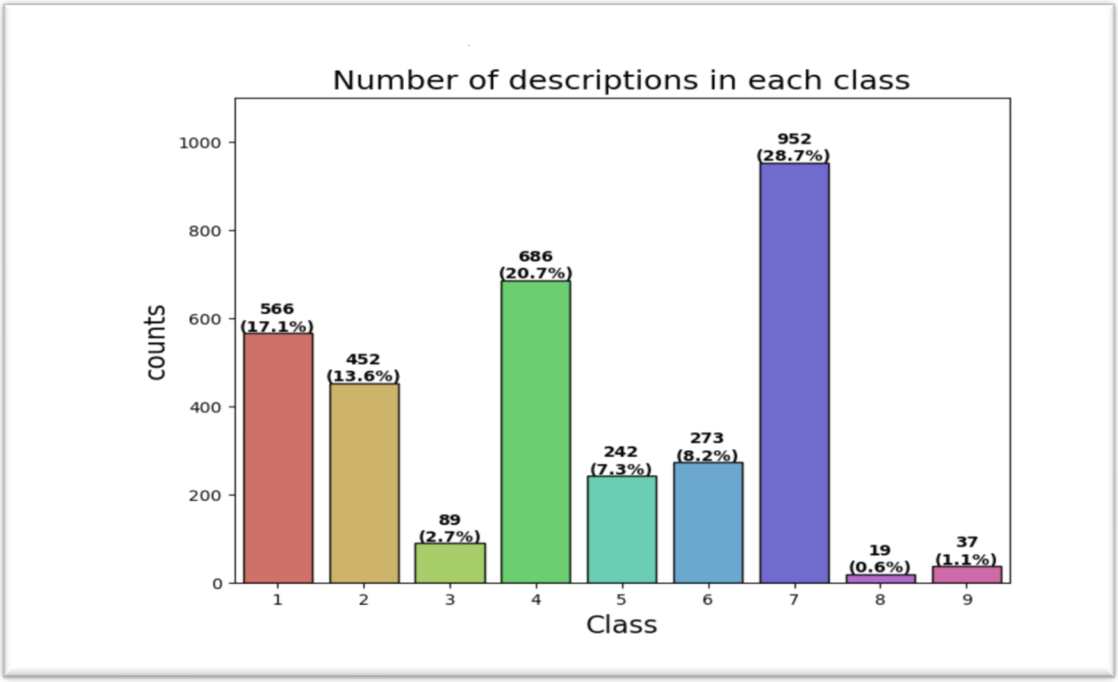

Figure 10:Distribution of Descriptions Across Classes: This bar plot visualizes the count of descriptions in each class, with percentage annotations for each class. The counts highlight the class distribution in the dataset

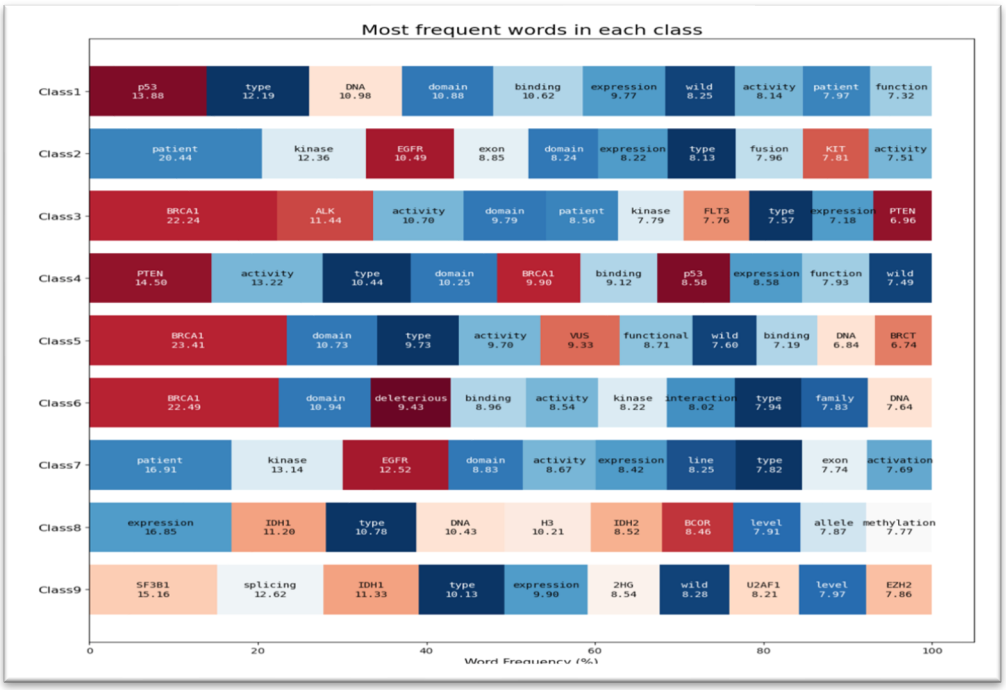

Figure 11: Most Frequent Words in Each Class: This stacked bar plot shows the distribution of word frequencies across different classes, with the most frequent words annotated to highlight their importance in each class.

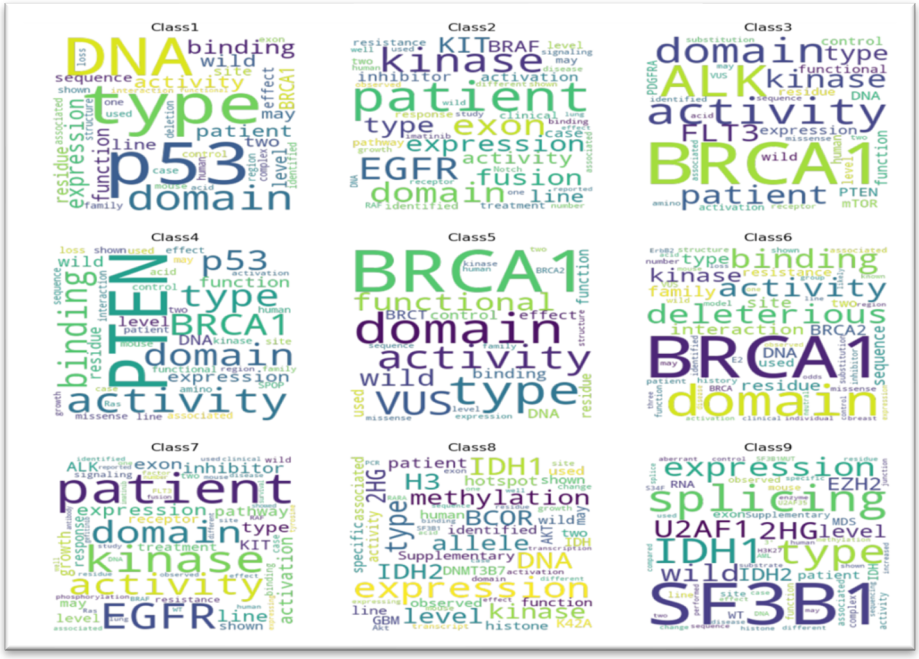

Figure 12: Word Clouds for Each Class: The word clouds visualize the most common words in each class. Larger words appear more frequently, providing insights into the key terms associated with each mutation class

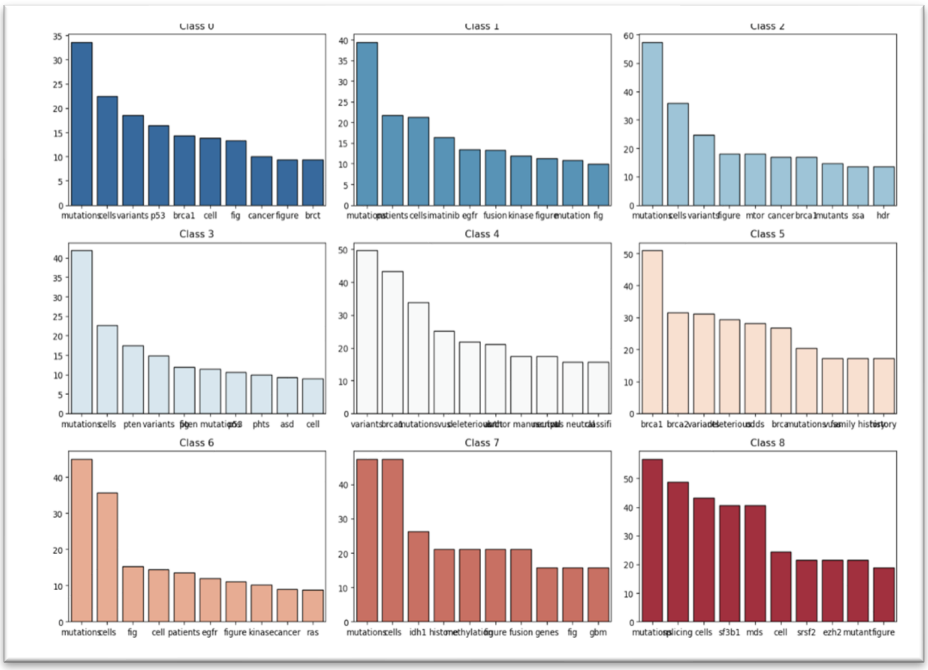

Figure 13: Top 10 Keywords for Each Class: This set of bar plots represents the top 10 most frequent keywords in each class, showing their average occurrences per description and revealing the critical terms associated with each mutation class

5.2. Detailed Model Analysis

5.2.1. Random Forest

- Random Forest performs the worst compared to other models, achieving an accuracy of 0.47 with CountVectorizer and 0.50 with Doc2Vec.
- An improvement is seen in Random Forest when using TF-IDF (accuracy = 0.64), which further means that the model scale enhances the performance of features such as term frequencies and overall term frequency across the data.
- One of the reasons Random Forest bears this low performance is because it is a high dimensional data based model where often simple decision trees fail to relate clinical evidence with mutation classes.

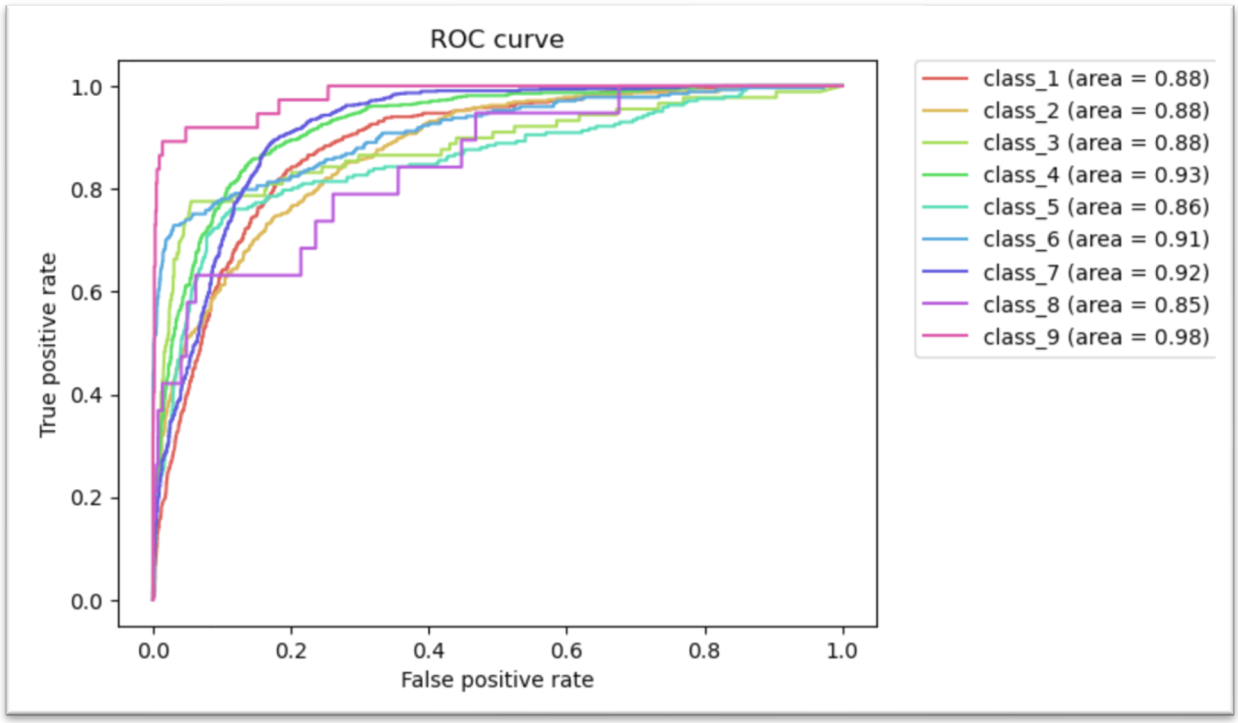

Figure 14: ROC curve under TF-ID vectorizer for Random Forest

5.2.2. Logistic Regression

- In comparison Logistic Regression is more understandable and to their operation relatively easy. This enables them to be useful as the first level models. With CountVectorizer it attained an accuracy of 0.36, 0.63 with TF-IDF and 0.59 in the case of Doc2Vec.
- Likewise, when it is still possible to approach the problem linearly, which is not the case with say classification of gene mutations, ... Such considerations make more sophisticated approaches such as XGBoost and LightGBM to outperform Logistic Regression.

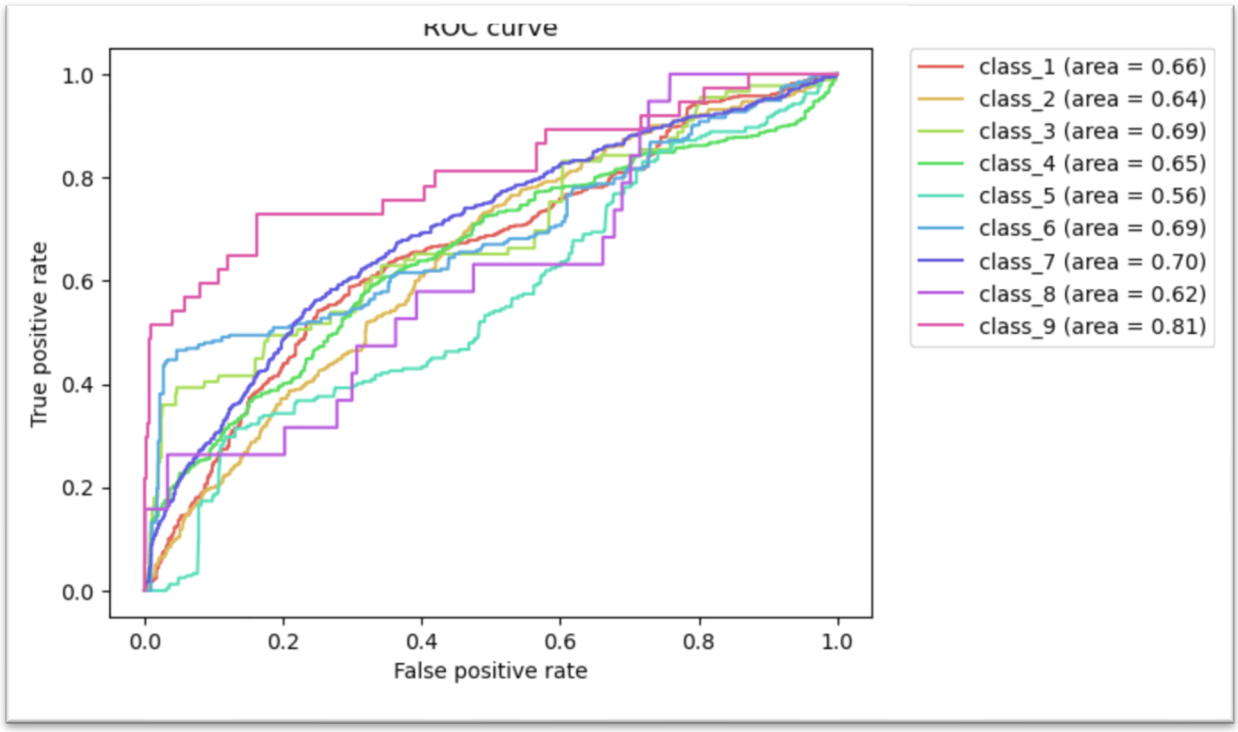

Figure 15: ROC curve under CountVectorizer for Logistic Regression

5.2.3. XGBoost

- XGBoost is an effective implementation of the gradient boosting framework with focus on enhanced prediction capabilities, particularly in structured datasets. It did acceptably well with the use of Doc2Vec (accuracy = 0.64) but faced challenges with other preprocessing techniques like CountVectorizer (accuracy = 0.49) and Word2Vec (accuracy = 0.54).
- CountVectorizer and Word2Vec performance issues can also be explained by the fact that these techniques tend to produce less effective feature representations of the text for XGBoost as they do not encode the meanings of words relative to each other as effectively as TF-IDF or Doc2Vec.

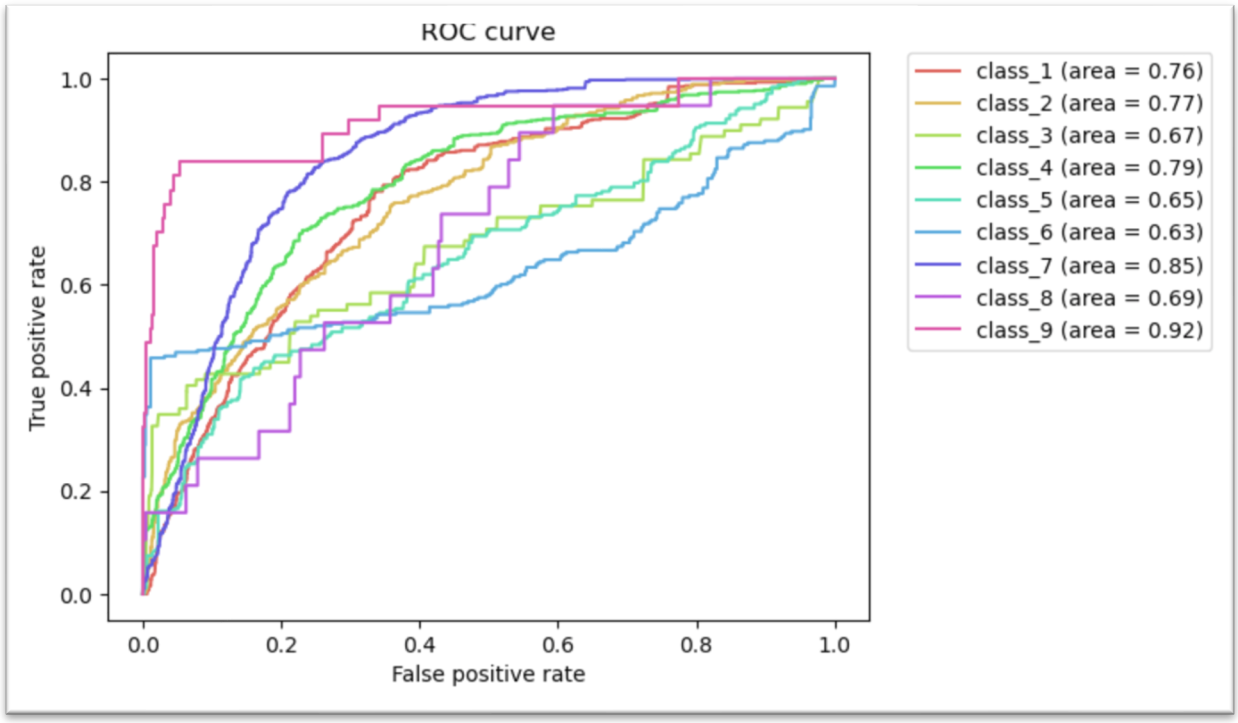

Figure 16: ROC curve under CountVectorizer for XGBoost

5.2.4. LightGBM

- All other models have been surpassed by LightGBM as the leading model. The highest being reached with Doc2Vec (accuracy = 0.93) After that the results are still high for TF-IDF (accuracy = 0.90).
- The use of Doc2vec also improves the prediction of LightGBM because it leverages the ability to represent documents as a vector fixed length encoding all its contents and its meaning, which is relevant when dealing with a lot of registers of clinical data with complex illustrations.
- The fact that LightGBM works efficiently even when high dimensional feature space and large volume of data exists explains why the model is the best suited for the task.

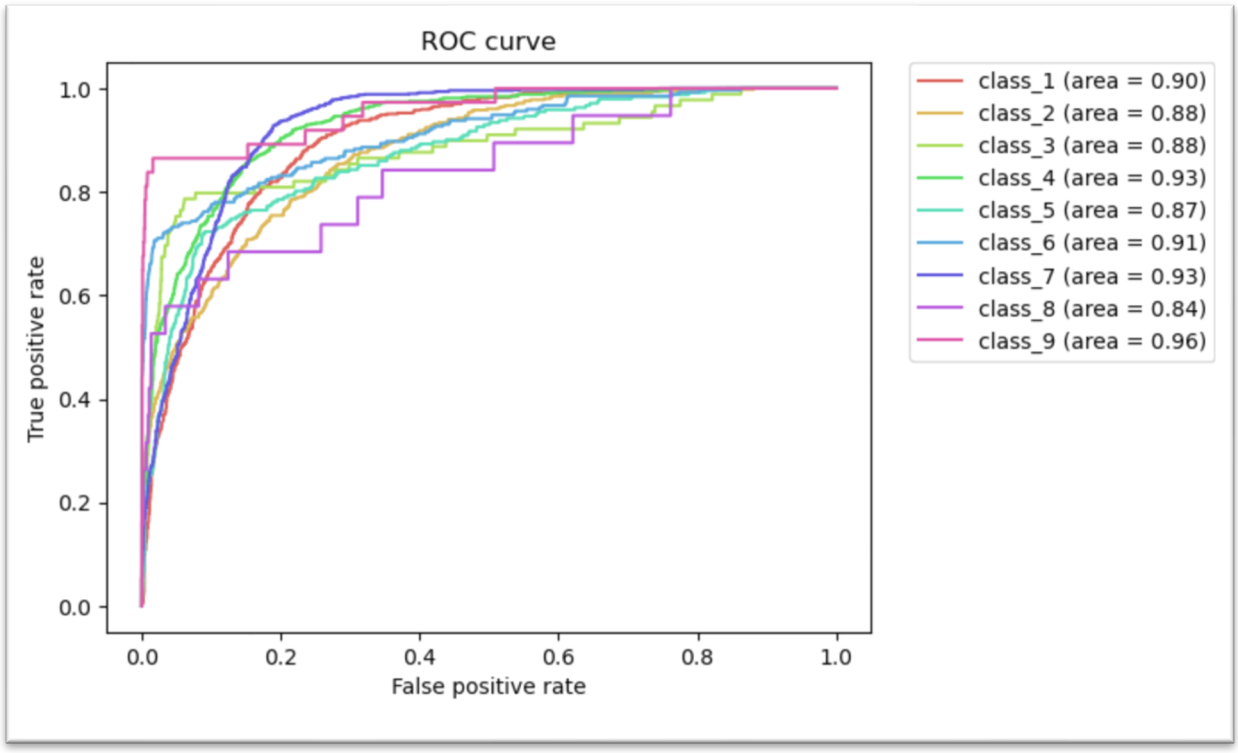

Figure 17: ROC curve under TF-ID vectorizer for LightGBM

5.3. Preprocessing Technique Analysis

5.3.1. CountVectorizer

- In CountVectorizer, which is widely used machine learning the technique ignores the context and importance of the documents. It transforms the input text into a sparse matrix containing word counts. However, the accuracies achieved by the different models are unyieldingly low with the best accuracy being recorded at 0.47 for the Random Forest model. This is an extreme case that can barely manage to recognize the relationships of the words within the document.
- With an accuracy of 0.47 for CountVectorizer(Random Forest), it is clear that the model is too simplistic and is in need of more elaborate and sophisticated features to disentangle the data.

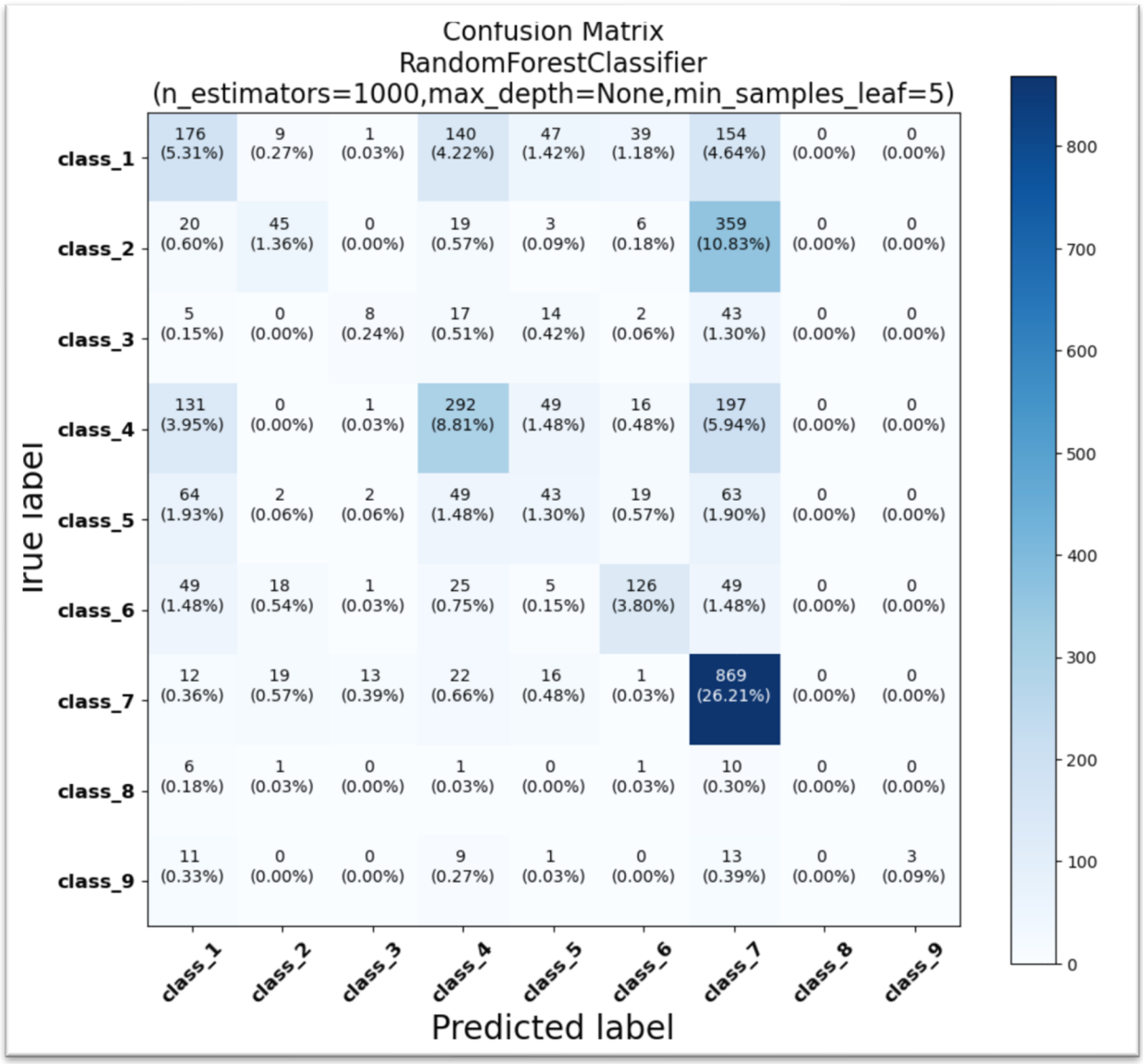

Figure 18: Confusion matrix under CountVectorizer

5.3.2. TF-IDF

- TF-IDF helps improve the feature representation values in which terms are assigned weights depending on how relevant they are throughout the corpus. This yields better results from the model as compared to CountVectorizer.

- Both XGBoost and LightGBM are effective in using TF-IDF with accuracy scores of 0.65 and 0.90 respectively.
- TF-IDF also allows to zonate the models in the sense of providing specific words which may be rare to certain types of mutations thus aiding in effective classification.

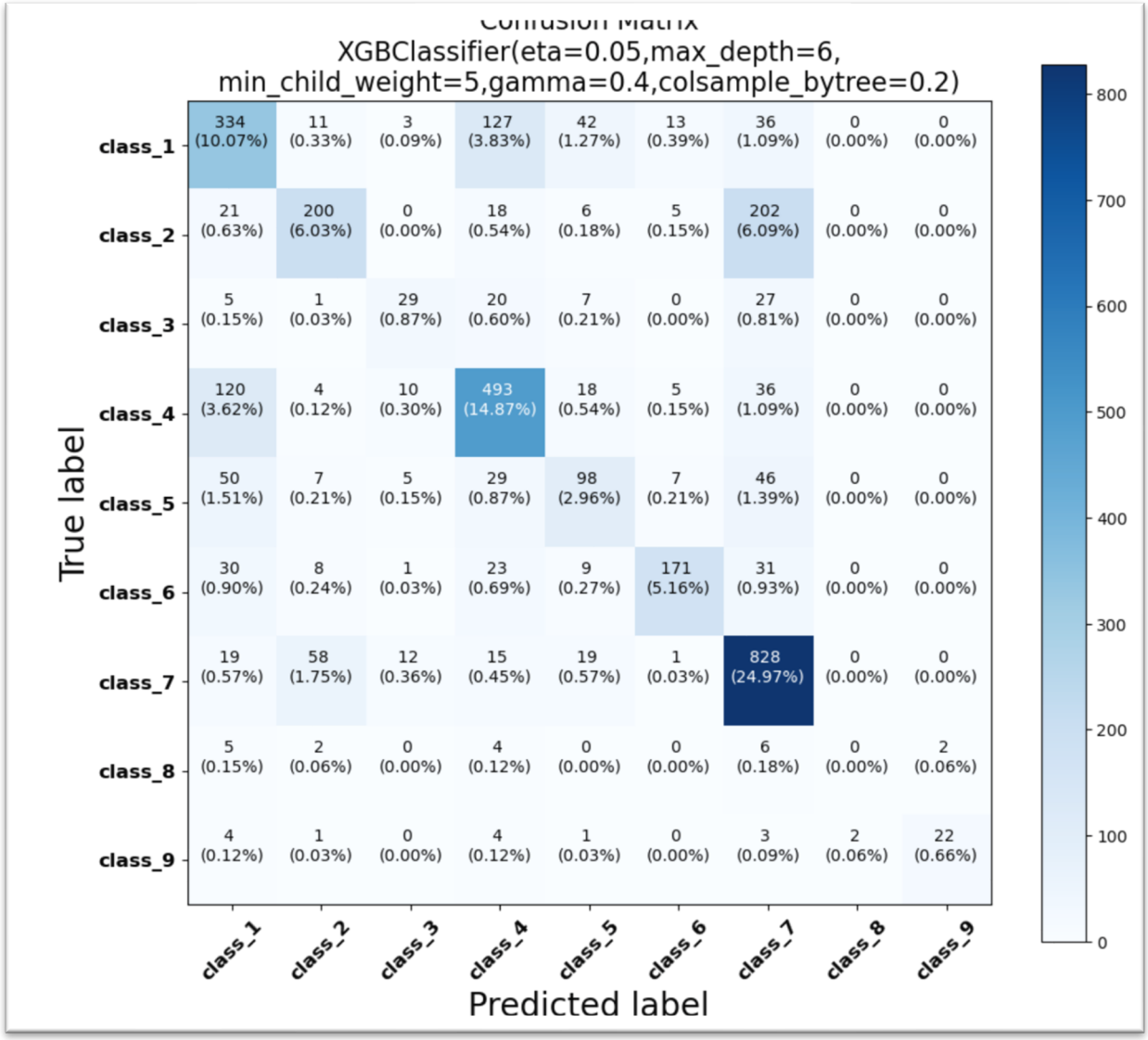

Figure 19: Confusion matrix under TF-ID vectorizer

5.3.3. Word2Vec

- In Word2Vec, individual words are represented in the form of vectors such that they capture the context of surrounding words, thus enabling modelling of meanings of relations among words. Even so, especially for the most models, it is worse than both TF-IDF and Doc2Vec.
- Speaking of Model LightGBM, accuracy is low with the use of Word2Vec only 0.43 which implies that in general that model helps understand the context and meaning contained within a word. However, it may not be appropriate for this specific task of classification that requires more than just words but also the whole document where such words exist.

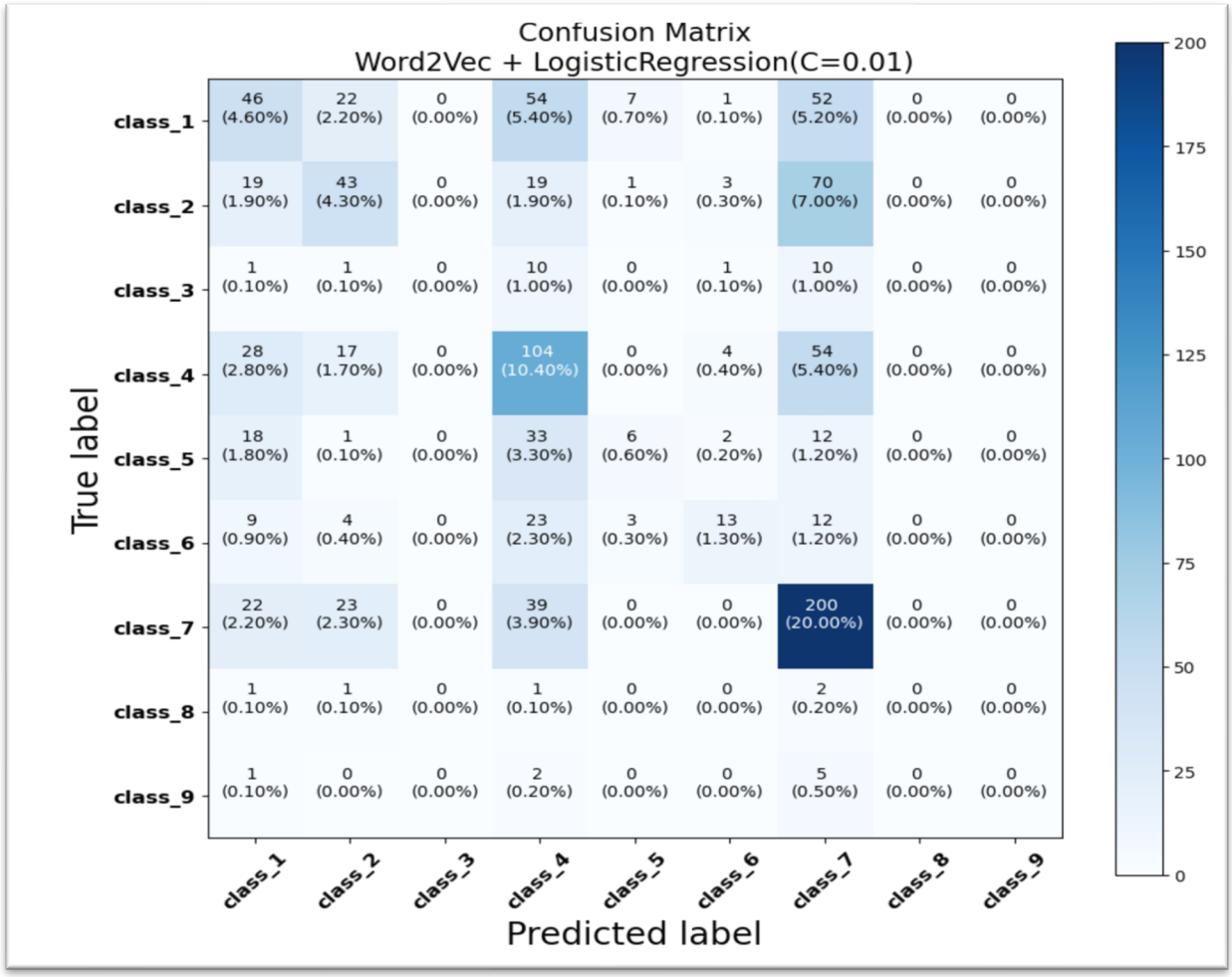

Figure 20: Confusion matrix under Word2Vec

5.3.4. Doc2Vec

- Doc2Vec stands out as the best preprocessing technique, especially for LightGBM and XGBoost. This method extends Word2Vec by generating fixed-length vectors for entire documents, capturing both semantic and syntactic information.
- Doc2Vec performs exceptionally well with LightGBM (accuracy = 0.93), indicating that this method is highly effective for classification tasks where the context of the entire document is important.

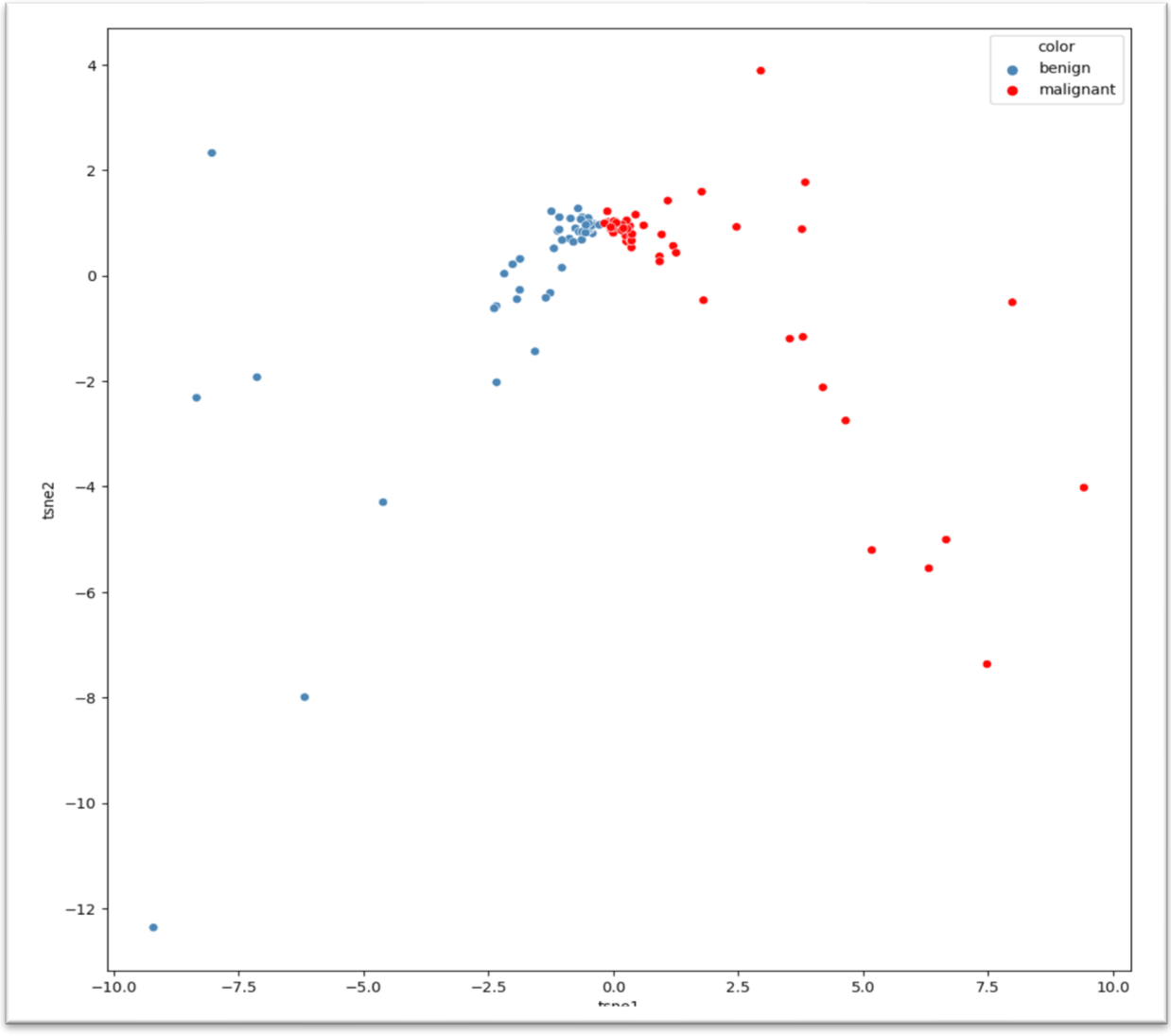

Figure 21: Benign vs Malignant plat under Doc2Vec vectorisation

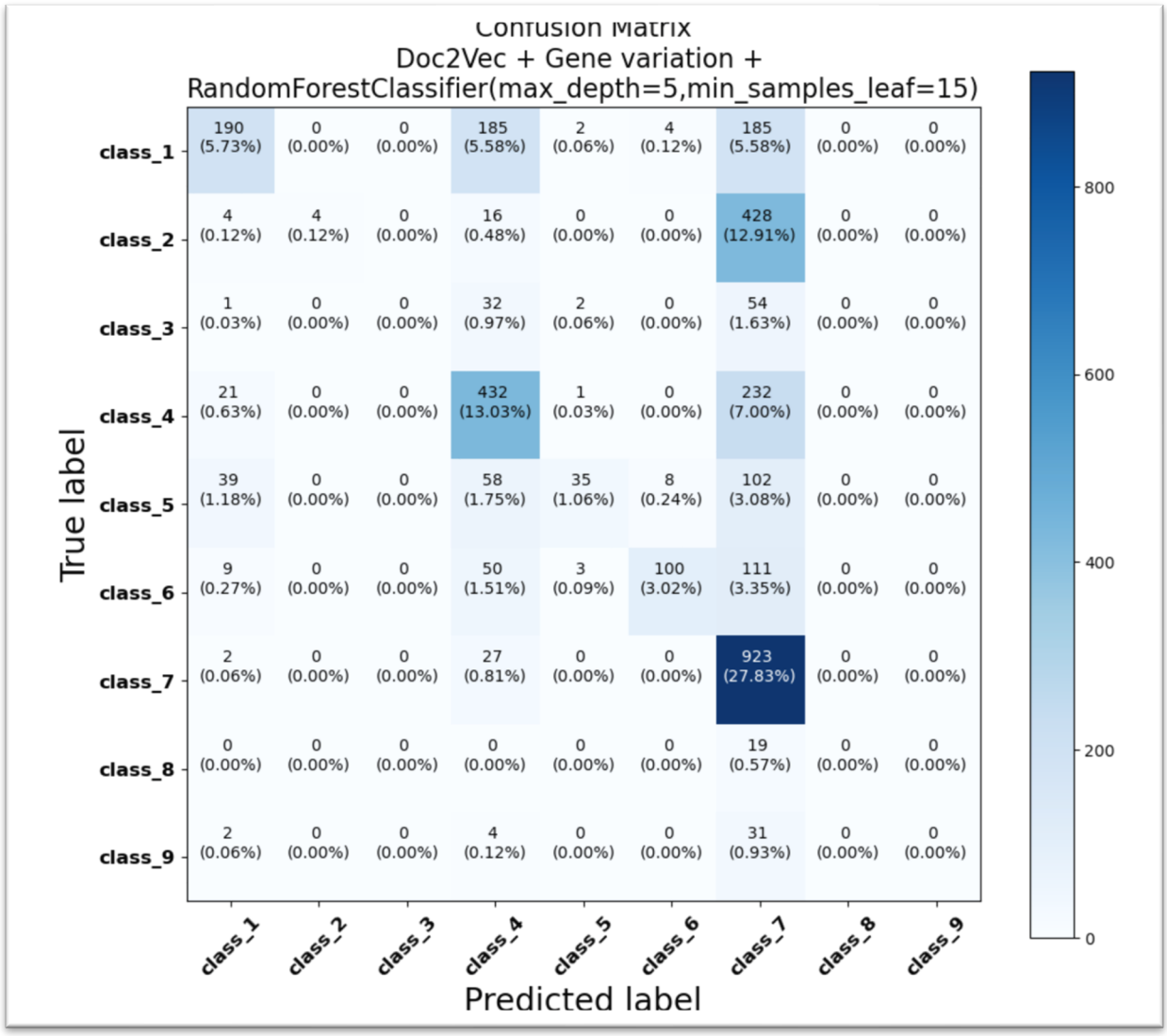

Figure 22: Confusion matrix under Doc2Vec

5.4. Comparison of Models and Preprocessing Techniques

Preprocessing Techniques The performance of each model is greatly determined by the preprocessing technique adopted. The experiments' results indicated that LightGBM is the best among all other models especially if it is incorporated with Doc2Vec. This implies that in this project the task of mutation classification that requires intense representation of features and large data sizes is well manageable with LightGBM.

In contrast, Random Forest and Logistic Regression have problems with applied most preprocessing approaches especially the less complex ones such as CountVectorizer and Word2Vec. This also proves that different advanced models and preprocessing techniques would be necessary in the area to improve the classification metrics.

Preprocessing Technique	Random Forest	Logistic Regression	XGBoost	LightGBM
CountVectorizer	0.47	0.36	0.49	0.88
TF-IDF	0.64	0.63	0.65	0.90
Word2Vec	0.49	0.49	0.54	0.43
Doc2Vec	0.50	0.59	0.64	0.93

Table 1: Summary of Model Accuracy Across Preprocessing Techniques

5.5. Limitations and Future Work

While the results from LightGBM and Doc2Vec are promising, there are some limitations to this approach:

- It has been noted that the Doc2Vec design works quite well, but it may be necessary to spend more time adjusting the hyperparameters.
- Performance of the models can be enhanced with the usage of additional data or specific knowledge, for example, text in combination with genetic data or clinical information.
- It would be beneficial to examine other embedding techniques such as BERT or GPT, most likely leading to improved classification results for clinical evidence.

5.6. Conclusion

It has been shown through the experiments conducted in this project that classification of genetic mutations in the clinical scenarios is best achieved through the application of LightGBM and Doc2Vec in unison. The incorporation of the Doc2Vec technique allows the model to incorporate valuable semantic context and hierarchy at the document level, which improves the overall accuracy of the classification model.

6. CONCLUSION & FUTURE WORK

The main aim of the project was to propose an innovative methodology to assess genetic mutations based on clinical evidence through machine learning techniques. In this instance the problem is the difficulty in understanding and translating textual clinical data into the corresponding genetic mutations, which causes the disease under study. The data set that has been supplied incorporates records based clinical evidence and mutations, and the job involves the process of identification of the mutations into several classes as described in the texts.

The experiments conducted were focused on testing different approaches to preprocessing and machine learning models in order to improve classification. Four text preprocessing techniques – CountVectorizer, TF-IDF, Word2Vec, and Doc2Vec – were used in combination with four machine learning classifiers: Random Forest, Logistic Regression, XGBoost and LightGBM. The purpose was to find the best model and preprocessing combination to obtain the most accurate classification results.

6.2. Major Experimental Findings

The experiments can be summarized in the following major results:

- LightGBM was the best performing model, as it achieved the best accuracy with all the available preprocessing approaches and especially with the use of Doc2Vec (accuracy = 0.93).
- Doc2Vec – which provides a vector representation of the whole document – was the best performing preprocessing approach. This method achieved better results than simpler means such as CountVectorizer and Word2Vec, which shows how important it is to use both the meaning and the structure of the clinical evidence.
- XGBoost had fair results using Doc2Vec but particularly it was not as good as LightGBM.
- Both Random Forest and Logistic Regression performed poorly for most of the approaches tried including the one with CountVectorizer that was very shallow in terms of its semantic meaning of the data.

It has emerged from these observations that LightGBM with Doc2Vec presents the most potential strategy for this kind of mutation classification. Integrating better feature representations with a fast gradient boosting tree model makes it possible to achieve high accuracy in mutation classification.

6.3. Future Work

While this project has yielded results that can be applauded, a number of elements are left to be improved and extended in the future:

1. Model Optimization and Hyperparameter Tuning:
    - Despite LightGBM being effective, there is a need for further hyper-parameters optimization aimed at improving its accuracy. This will include but not limited to the adjusting learning rates, tree depth and number of iterations among other parameters.
    - On the other hand, XGBoost and random forest could be used but this time with more sophisticated hyperparameter optimization- Bayesian optimization in order to perform better with the models.
2. Advanced Text Embeddings:
    - Implementing more advanced text embeddings e.g. BERT or any other transformer based embedding can be very useful in providing more comprehensive representations of clinical evidence hence classification accuracy can be improved.
    - Using transformers that have been pre-trained on medical text and refitted such as BioBERT or ClinicalBERT will help the system to grasp the domain's terminologies and intricacies better.
3. Incorporating Multimodal Data:
    - In addition to textual data, the introduction of genetic metrics like gene sequence and mutation metrics in conjunction with clinical evidence would also improve the model significantly. For example, fusion of data types would give a better outlook of the genetic mutation distribution.
    - The inclusion of clinical notes, patient background data, and similar structured information can also provide extra information to facilitate the classification.

4. Transfer Learning and Few-Shot Learning:
    - This is in contrast to other areas of biomedical research where there is usually data, especially labeled data, challenges many biomedical tasks. Future investigations could include investigating techniques for transfer learning that would enable mutation classification based on models that have already been trained for other domains.
    - Adopting few-shot learning techniques could help the model perform better in the few situations where there is no or low availability of labeled data, a common challenge in genetic mutation datasets.
5. Model Explainability:
    - Given the clinical nature of the data, model interpretability will be crucial for real-world adoption. In particular, lightgbm approaches should not only be introduced but also their understanding on the application of mrem the focus now be to advance the explainability of the machine learning models. Incorporating methods such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-Agnostic Explanations) would also make it easier to appreciate the main contributors to the output of the model.
6. Validation and Deployment:
    - For the strength of the developed models, it would be good to carry out additional validation using external datasets, from other health institutions or mutation banks. This is to investigate the extent to which the model can be used in real life.
    - The last stage will entail rolling out the model within a clinical decision support system. This may entail incorporating a frontend where the clinicians can key in clinical information and retrieve predicted classes of mutations instantaneously.

6.4 Conclusion

This project was able to show how machine learning can be used to classify different gene mutations based on the clinical evidence available. The experiments also pointed out that LightGBM with Doc2Vec is evidently the best performing combination with regard to

classification accuracy. Nevertheless, there is potential for enhancement by way of model refinement, enhanced text embeddings and employing multimodal data.

If these future directions are taken into account, the system could turn out to be one of the most effective instruments in the field of genomics and precision medicine that would support the improvement of clinical onco-genetic decision making about drug therapy for cancer and mutation diagnosis.

REFERENCES

[1] Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). "Gene Selection for Cancer Classification Using Support Vector Machines." *Machine Learning*, 46(1–3), 389-422. Available online: http://dx.doi.org/10.1023/A:1012487302797

[2] Aburass, S., Dorgham, O., & Al Shaqsi, J. (2024). "A Hybrid Machine Learning Model for Classifying Gene Mutations in Cancer Using LSTM, BiLSTM, CNN, GRU, and GloVe." *Scientific Advances in Smart Computing*, Open Access, Creative Commons License. Available online: https://doi.org/10.1016/j.sasc.2024.200110

[3] Gadia, V., & Rosen, G. (2008). "A Text-Mining Approach for Classification of Genomic Fragments." *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW 2008)*.
Available online: https://doi.org/10.1109/BIBMW.2008.4686216

[4] Yoon, S., Hwang, I., Cho, J., Yoon, H., & Lee, K. (2023). "miGAP: miRNA–Gene Association Prediction Method Based on Deep Learning Model." *Applied Sciences*, 13(22), 12349.
Available online: https://doi.org/10.3390/app132212349

[5] Kim, S., Lee, H., Kim, K., & Kang, J. (2018). "Mut2Vec: Distributed Representation of Cancerous Mutations." *BMC Medical Genomics*, 11(Suppl 2), 33.
Available online: https://doi.org/10.1186/s12920-018-0349-7

[6] Ankrah, B., Brew, L., & Acquah, J. (2024). "Multi-Class Classification of Genetic Mutation Using Machine Learning Models." *Computational Journal of Mathematical and Statistical Sciences*, 3(2), 280-315.
Available online: http://dx.doi.org/10.21608/cjmss.2024.267064.1040

[7] Yeturu, J., Elango, P., Raja, S. P., & Kumar, P. N. (2021). "A Novel Ensemble Stacking Classification of Genetic Variations Using Machine Learning Algorithms." *International*

Journal of Image and Graphics, 23(02).
Available online: https://doi.org/10.1142/S0219467823500158

[8] Shubair, R. (2016). "Comparative Study of Machine Learning Algorithms for Breast Cancer Detection and Diagnosis." *Proceedings of the 2016 IEEE 5th International Conference on Electronic Devices, Systems, and Applications (ICEDSA 2016)*.
Available online: https://doi.org/10.1109/ICEDSA.2016.7818560

[9] Sun, Y., & Wong, A. (2011). "Classification of Imbalanced Data: A Review." *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04).
Available online: https://doi.org/10.1142/S0218001409007326

[10] Minamoto, T., & Ronai, Z. (2001). "Gene Mutation as a Target for Early Detection in Cancer Diagnosis." *Critical Reviews in Oncology/Hematology*.
Available online: https://doi.org/10.1016/S1040-8428(01)00098-1

[11] Mohanty, A., Prusty, A. R., & Cherukuri, R. C. (2022). "Cancer Tumor Detection Using Genetic Mutated Data and Machine Learning Models." *Proceedings of the 2022 International Conference on Intelligent Controller and Computing for Smart Power (ICICCSP)*.
Available online: https://doi.org/10.1109/ICICCSP53532.2022.9862476

[12] Koboldt, D. C., Zhang, Q., Larson, D. E., Shen, D., McLellan, M. D., Lin, L., Miller, C. A., Mardis, E. R., Ding, L., & Wilson, R. K. (2012). "VarScan 2: Somatic Mutation and Copy Number Alteration Discovery in Cancer by Exome Sequencing." *Genome Research*, 22(3), 568-576.
Available online: https://doi.org/10.1101/gr.129684.111

[13] Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., & Fotiadis, D. I. (2014). "Machine Learning Applications in Cancer Prognosis and Prediction." *Computational and Structural Biotechnology Journal*, 13, 8-17.
Available online: https://doi.org/10.1016/j.csbj.2014.11.005