Personalized Medicine: Redefining Cancer Treatment
Project Overview

Overview
Personalized-Medicine-Redefining-Cancer-Treatment is a problem of classifying the given genetic mutations based on the literature available in the medical domain into one of the given 9 classes.

A lot has been said during the past several years about how precision medicine and, more concretely, how genetic testing is going to disrupt the way diseases like cancer are treated.

But this is only partially happening due to the huge amount of manual work still required. In this project, we will try to take personalized medicine to its full potential.

Once sequenced, a cancer tumor can have thousands of genetic mutations. But the challenge is distinguishing the mutations that contribute to tumor growth (drivers) from the neutral mutations (passengers).

Currently, this interpretation of genetic mutations is being done manually. This is a very time-consuming task where a clinical pathologist has to manually review and classify every single genetic mutation based on evidence from text-based clinical literature. The machine learning solution to this problem will help to speed up this time-consuming procedure and produce results with significant accuracy.

In this project, we create features out of medical literature data and develop a machine learning algorithm that, using this knowledge base as a baseline, automatically classifies genetic variations.
We will be experimenting with different models such as Logistic Regression, Random Forest, KNN, and Naive Bayes to find the best-fitting model for the problem statement.

Aim
To classify genetic mutations on the basis of medical literature into the given 9 classes.

Data Description
The dataset is divided into variants and text for training and test datasets which includes the following features:

- Fields are ID (the id of the row used to link the mutation to the clinical evidence)
- Gene (the gene where this genetic mutation is located)
- Variation (the aminoacid change for these mutations)
- Class (1-9 the class this genetic mutation has been classified on)
- Training_text (ID, Text), Text (the clinical evidence used to classify the genetic mutation)

Tech Stack

➔ Language: Python

➔ Libraries: pandas, numpy, pretty_confusion_matrix, matplotlib, sklearn, pymongo[srv]

Approach

- Data Reading
- Data Analysis
    - Class
    - Gene
    - Variation
- Text Preprocessing
- Splitting Data, Evaluation, and Features Extraction
- Model Building
    - Logistic Regression
    - Random Forest
    - KNN
    - Naive Bayes
- Hyperparameter Tuning
    - Logistic Regression
- Model Evaluation
    - Confusion matrix
    - Log Loss

Modular code overview:

```
figures

lib
|_Personalized_medicine_project.ipynb


ml_pipeline
|_data_analysis.py
|_train.py
|_test.py
|_utils.py

engine.py

requirements.txt

config.yaml

readme.md
```

Once you unzip the modular_code.zip file, you can find the following folders within it.

1. lib
2. ml_pipeline
3. engine.py
4. requirements.txt
5. config.yaml
6. readme.md

1. The lib folder is a reference folder, and it contains the original ipython notebook as in the lectures.

2. The ml_pipeline is a folder that contains all the functions put into different python files, which are appropriately named. The Engine.py script then calls these python functions to run the steps in one go to train the model and print the results.

3. The requirements.txt file has all the required libraries with respective versions. Kindly install the file by using the command **pip install -r requirements.txt**
4. The config.yaml contains the project specifications needed.

5. **All the instructions for running the code are present in readme.md file**

Project Takeaways

1. Understanding the problem statement
2. Performing basic EDA
3. The various steps in Text Preprocessing
4. Lemmatization
5. Tokenization and converting text to sequences
6. Using "Tfidf Vectorizer" for the deriving relationships between different words
7. Performing training, testing, and validation split on the dataset
8. Multi-Class Classification
9. Understanding Evaluation Metrics
10. Understanding Log Loss and Confusion Matrix
11. Logistic Regression Implementation
12. KNN Implementation
13. Random Forest Classifier Implementation
14. Naive Bayes Classifier Implementation