Data Glacier

# HealthCare: Persistency of a Drug

Data Science Internship Project

Group Name: The Data Doctors

By:
Ashish Sasanapuri
Mohammad Shehzar Khan
Tomisin Abimbola Adeniyi

30/12/2023

# Table of contents

# List of Figures

# 1. Introduction

## 1.1 Problem Statement

One challenge for all pharmaceutical companies is to understand the persistence of a drug as per the physician's prescription. To solve this problem ABC Pharma company approached an analytics company to automate this process of identification. The aim is to build a persistency prediction application that classifies patients persistent to New Medication Therapies (NTM) of the drugs prescribed by their respective physicians.

This application will help in studying and understanding the factors impacting a patient to be compliant or non-compliant to the prescribes treatment. Along with this, the application will also help in improving the persistency based on the understanding of the historical data presented by the Pharma company.

## 1.2 Project Lifecycle

The project flow is divided into different sections from transforming the data into understandable format and analysing the data assisting in building the machine learning model.

### 1.2.1 Research/Study

This section involves going through the problem statement for better understanding on the objective, researching different methods for feature and model selection.

### 1.2.2 Data Processing

The initial phase of this step involves cleaning the dataset for converting the data from unstructured to structured data, handling missing values and outliers. Post the pre-processing of the data, analysing the data for business impacting patterns is an important step that is carried out by Exploratory Data Analysis (EDA). The prime focus of this step is to understand the impact of different features with the target variable through visualisations and obtaining their importance for model training.

### 1.2.3 Model Building

Post exploration of the dataset, feature selection methods will be implemented to extract features that have potential impact on the machine learning model to be trained and also help in reducing complexity and dimensionality while improving performance of the model.

The model selection step is carried out by comparing the performance of different models by implementing different performance metrics such as Accuracy, Precision, Recall and F1-score.

### 1.2.4 Deployment

The aim is to develop an application using Flask to create APIs for calling the model, perform the prediction and displaying the result on the application page. The application is then wrapped in a container using Docker for easy virtualisation and deployment of the application irrespective of the environment and running via a single command run.

This containerised application is hosted on a cloud environment that helps in providing abstract environment consisting of various services and network security for running the application.

# 2. Methodology

This section describes the dataset used in detail and explains the methodologies adopted to handle issues with the dataset. The sub-sections provide details regarding the pre-processing and analysis of the data.

## 2.1 Data Understanding

The dataset used is a Healthcare dataset containing medical attributes corresponding to persistency of each patient to New Therapy Medication (NTM) by the ABC's Pharmaceutical company. The dataset consists of 3242 records of patients and 69 different features described as below:

- 1 Target variable: *Persistency_Flag*
- 1 Unique identifier for each patient: *Ptid*
- 6 Demographic variables of each patient: *Age_Bucket*, *Gender*, *Race*, *Ethnicity*, *Region*, *Idn_Indicator*.
- 3 Physician Specialist attributes: *Ntm_Speciality*, *Ntm_Specialist_Flag*, *Ntm_Specialist_Bucket*.
- 13 Clinical factors: *T-Score* details, *Risk_Segment* details, *Multiple risk factors count*, *DEXA* details, *Fragility fracture* details, *Glucocorticoid* details.
- 45 Disease/Treatment factors: *Injectable drugs*, *Risk factors*, *Comorbidities*, *Concomitancies*, *Adherence to therapy*.

There 2 types of data points in the dataset – numerical and categorical. Out of these 69 features, two features *Dexa_Freq_During_Rx* and *Count_Of_Risks* are numerical features and the rest are categorical features.

## 2.2 Data Pre-processing

### 2.2.1 Handling Missing Values/Outliers

There were no missing values in the dataset. Given that there were only two numerical features, we specifically examined outliers for these features. The methods used to detect outliers were Boxplot visualization, Histogram, Interquartile Range (IQR) and Z-score.

Both of the features exhibited outliers, and the data distribution demonstrated a positive skew. To overcome inaccurate insights and decisions from the dataset, the outliers were handled via following techniques:

- *Box-Cox Transformation* – It tries to stabilize the variance in the data and improves the distribution of data to normal.
- *Log Transformation* – Applying log to data points changes the values but also help reduce skewness in the data.
- *Square Root Transformation* – Takes the square root of each data point helping with skewness of the data and converting into normal distribution.
- *Winsorization* – This helps cap the outliers with the upper and lower bound limits obtained from *IQR*.

Among the mentioned methods, *Box-Cox transformation*, Figure 1, obtained better results at handling the outliers most efficiently.

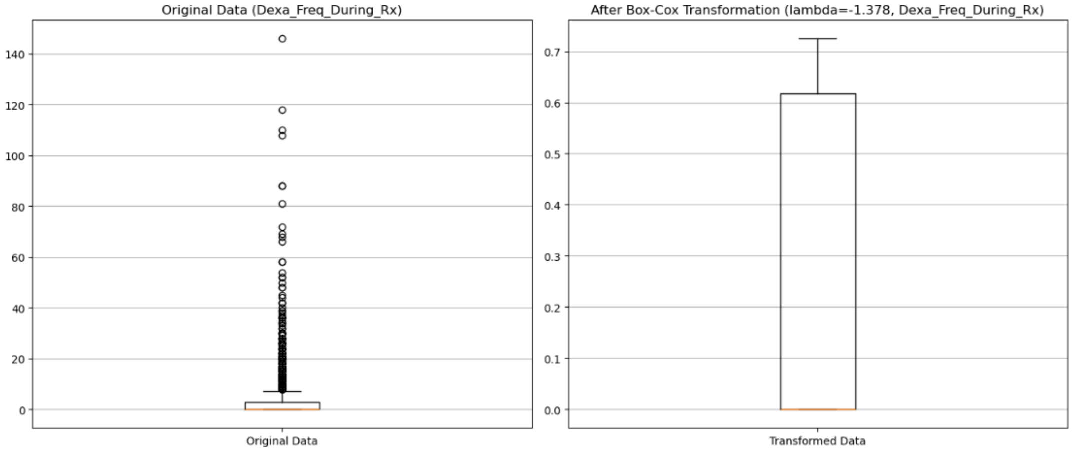


*Figure 1: Box plot visualisation for before and after transformation of Dexa_Freq_During_Rx feature using Box-Cox method*

For the *Count_Of_Risks* feature, as the categories are unique number of risks per patient the outliers were handled by reducing the categories to 0, 1, 2, >=3, Figure 2.

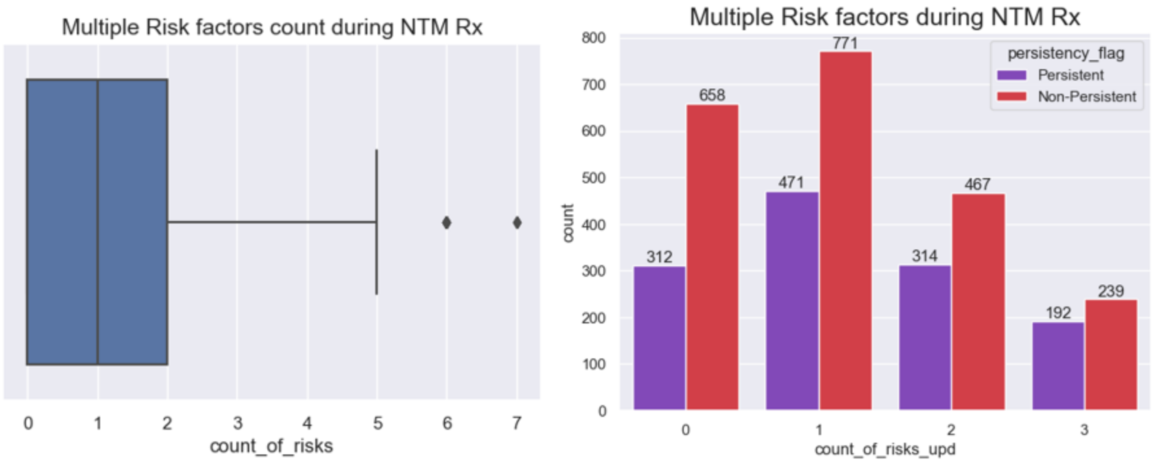


*Figure 2: Before and after transformation of Count_Of_Risks feature after handling outliers*

### 2.2.2 Data Transformations

Data transformation is one of the important sections when feeding the data to the machine learning model for training. As the machine learning models only understand numerical data, majority of features consisting of categorical data is converted into numerical data type using LabelEncoder library of SciKit Learn.

## 2.3 Exploratory Data Analysis (EDA)

Understanding the dataset is one of the most important and challenging tasks in machine learning. Exploring the data will help in gaining insights and making sense of the data which will guide us through selecting relevant features that impact the performance of the model and generate better results for testing new data points. This section will explore the steps performed to visualize the data to generate patterns in the data.

### 2.3.1 Demographics

This section describes the demographics of each patient such as *Age*, *Gender*, *Region*, *Race*, and *Ethnicity*. The features such as *Gender*, *Race* and *Ethnicity* don't provide much information when compared with the target variable, *Persistency_flag*, Figure 3.

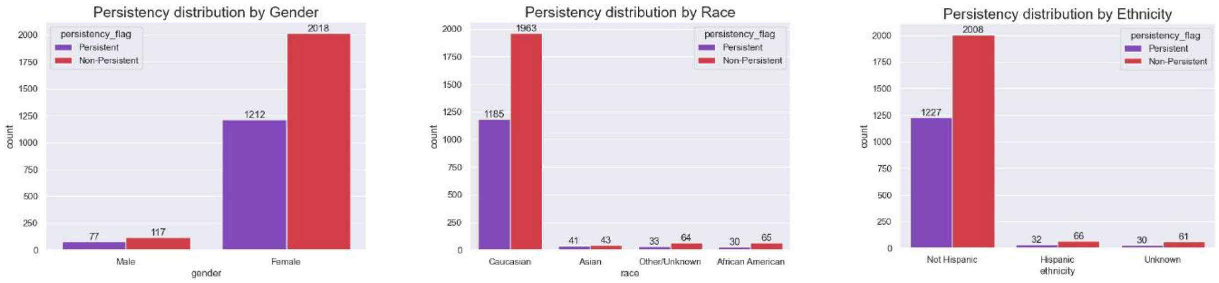


*Figure 3: Plots for Gender, Race and Ethnicity features*

The *Age_Bucket* feature shows the age groups of different patients recorded in the dataset. Based on the observations from the plot, Figure 4, the majority of patients recorded are above 55 years of age and most **Non-persistent** patients belong to the category of above 75 years of age.

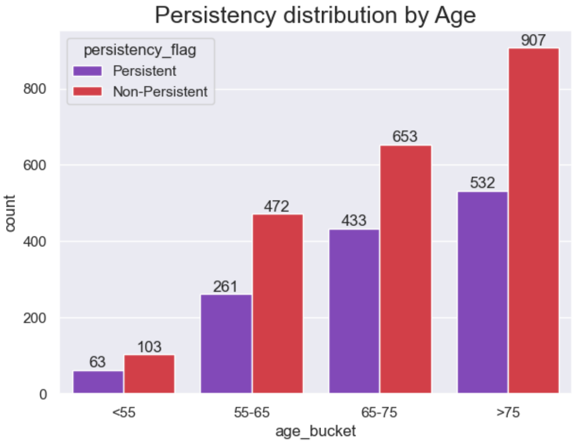


*Figure 4: Age group distribution*

A distribution of age brackets can also be observed with respect to the regions in the United States, Figure 5, where most of the patients belong to **Midwest** or **South** regions.

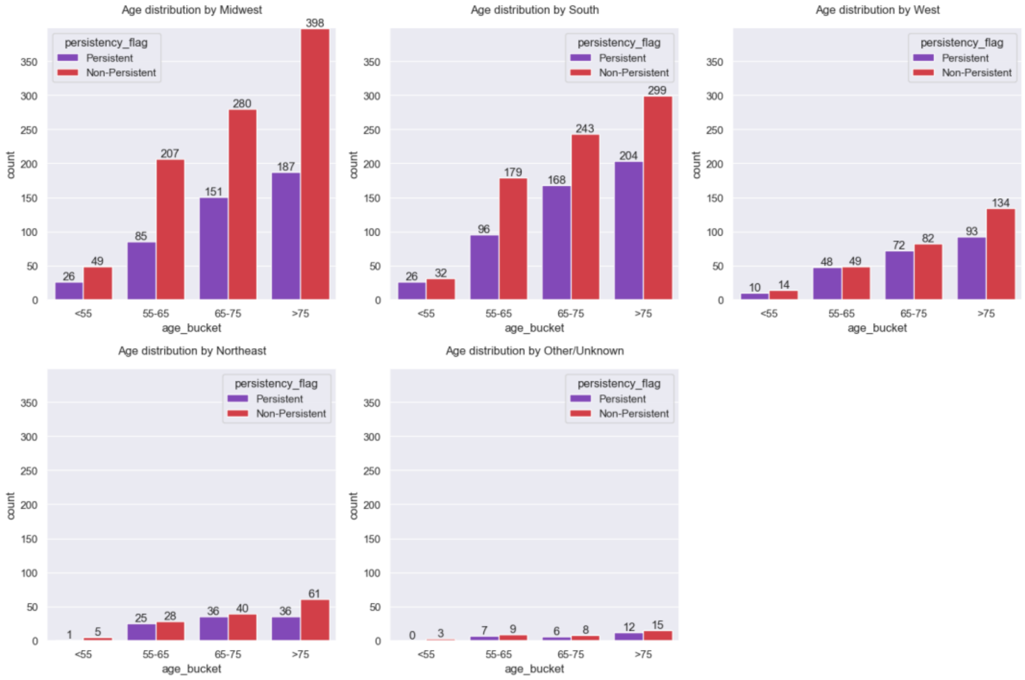


*Figure 5: Age distribution with respect to Region*

### 2.3.2 Physician Attributes

The physician attributes provide information regarding the speciality of physicians who prescribed the NTM to the patients. A total of 36 different physicians can be observed from Figure 6. Around *45%* of the physicians belong to **General Practitioner** category.

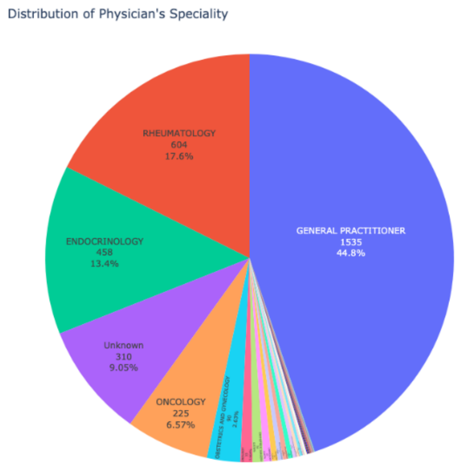


*Figure 6: Pie chart displaying Physician specialities*

One of the other attributes which combines these specialities into 3 different categories provides a significant distribution when compared with the target feature, Figure 7. Majority of patients are **Non-persistent** and have been prescribed by *non-specialists*.

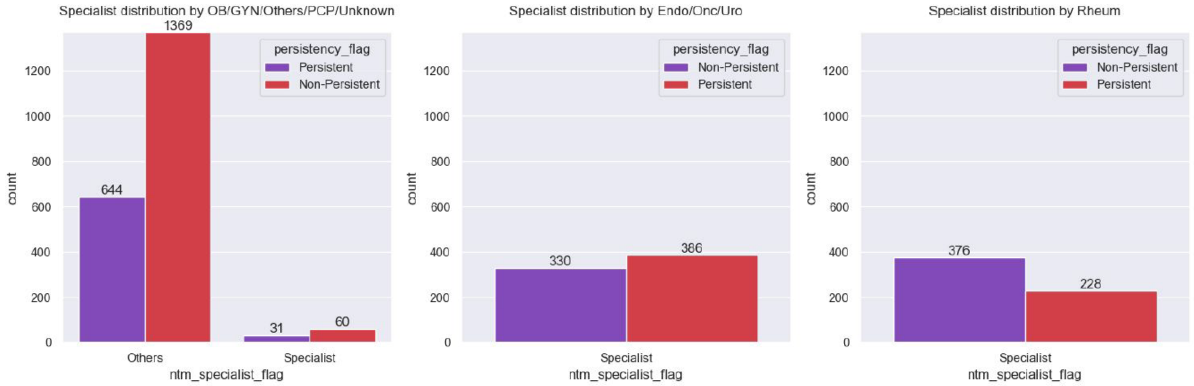
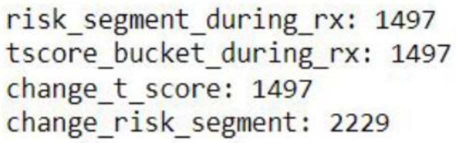


*Figure 7: Physician specialities divided into 3 different categories*

### 2.3.3 Clinical Factors

The clinical factors describe the factors such as DEXA scans carried out by patients, Fragility fractures of the patients, Bone density obtained from the DEXA scans, and the usage of Glucocorticoids by patients prior and during therapy. Although some of the features which have been recorded during the therapy have 'Unknown' values as evident in Figure 8.



```
risk_segment_during_rx: 1497
tscore_bucket_during_rx: 1497
change_t_score: 1497
change_risk_segment: 2229
```

*Figure 8: List of features with Unknown values*

Considering the usage of Glucocorticoids, the number of **Persistent** patients is less as compared to **Non-persistent** category patients prior to the therapy. However, the case is vice-versa during the therapy as observed from the plot in Figure 9.

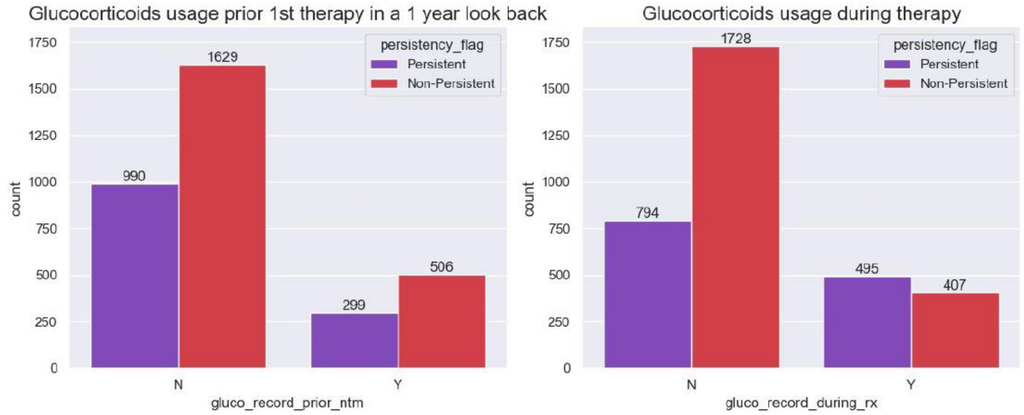


*Figure 9: Glucocorticoids usage prior and during therapy*

The *DEXA scans* prescribed to patients are the tests carried out to check the bone density of the patients after a Fragility Fracture or other factors. Most of the patients who haven't taken these tests majorly fall under **Non-persistent** category, Figure 10.

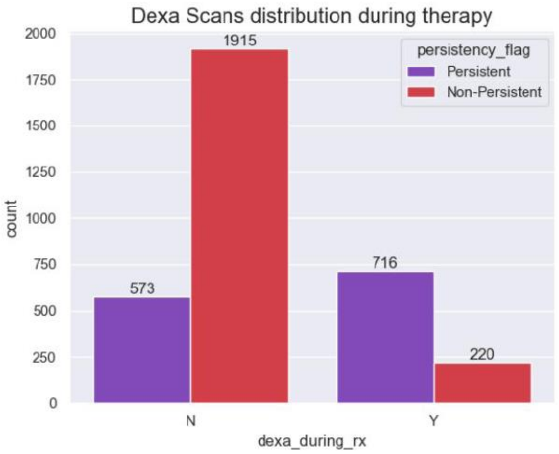


*Figure 10: Plot showing patients taking DEXA scans during therapy*

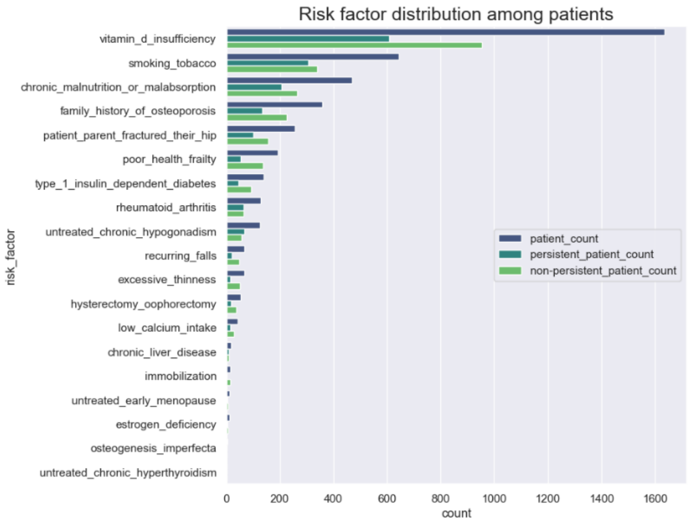


*Figure 11: Risk factor distribution by persistency*

### 2.3.4 Disease/Treatment Factors

This section details the different types of risks and comorbidities occurred to patients along with the concomitant drugs provided to the patients in a 1 year look back from the 1st NTM therapy.

Majority of patients have been susceptible to *Risk Factors* such as *Vitamin D deficiency*, *Smoking tobacco*, *Chronic malnutrition or malabsorption* and a *Family history of osteoporosis* as observed from Figure 11.

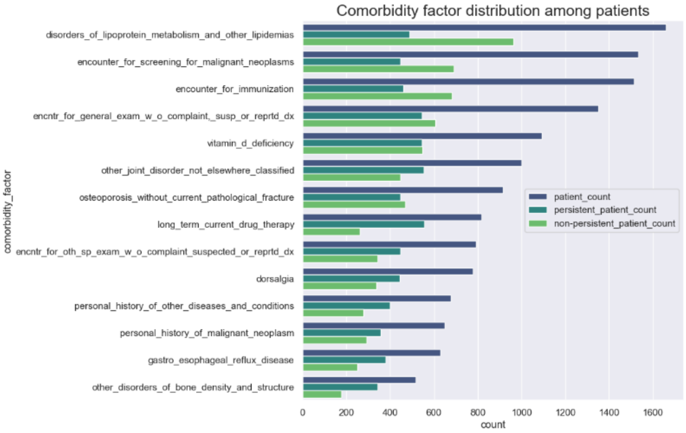


*Figure 12: Comorbidity factor distribution by persistency*

The *Comorbidities* of the patients recorded in the dataset are the occurrence of more than one disease or condition at the same time prior to the 1st therapy. There are a total of 14 *Comorbidities* as show in Figure 12.

*Concomitant* drugs are 2 or more drugs given to patients at the same time. The number of **Non-persistent** patients given *Concomitant* drugs such as *narcotics*, *Cholesterol and Triglyceride*, *systematic corticosteroids* and *Anti-depressants or mood stabilisers* prior to 1st therapy are greater in comparison to **Persistent** patients as evident from Figure 13.

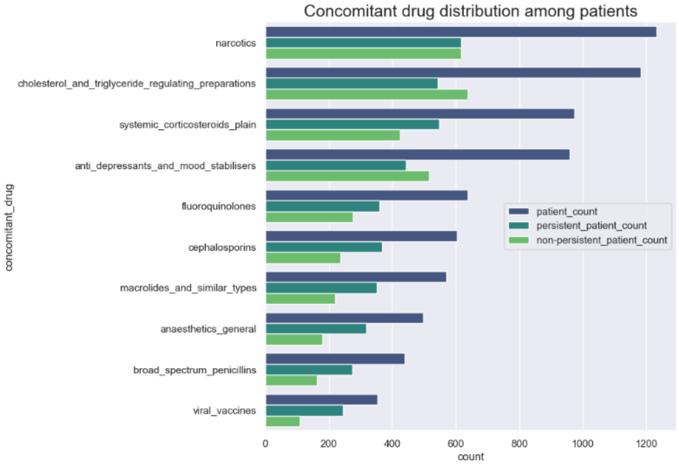


*Figure 13: Concomitant drugs provided to patients*

Figure 14 shows the number of risk factors patients had at the same time. As the number of risks at the same time increases, the count of patients decreases.

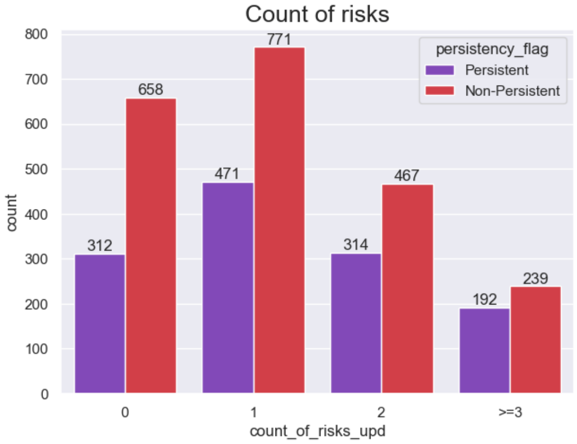


*Figure 14: Number of risks patients had at the same time*

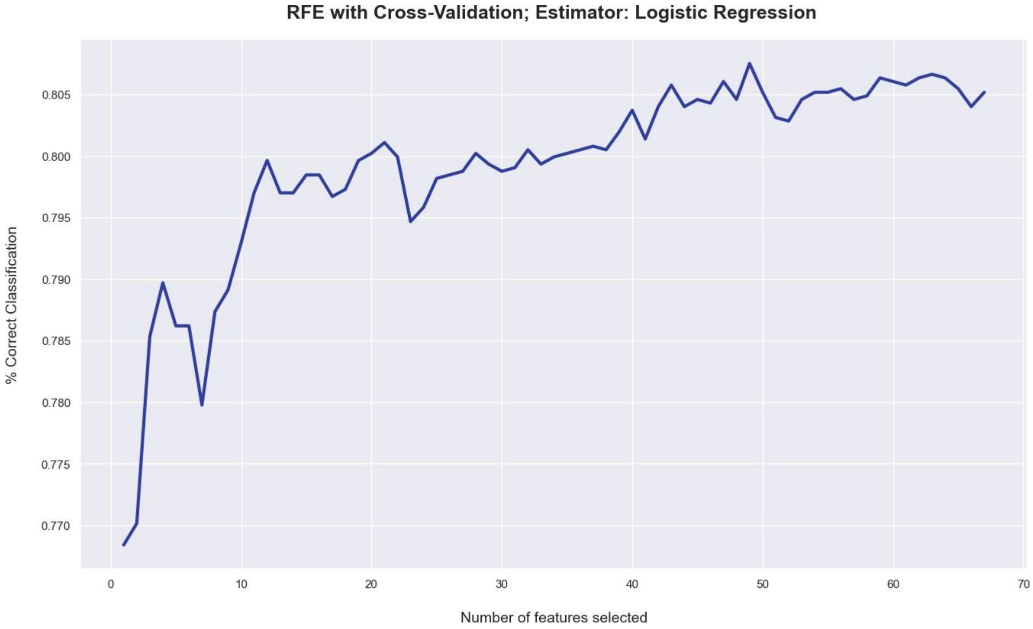


*Figure 15: Correct classification rate using RFECV*

# 3. Model Building

This section introduces the feature selection methods and machine learning models used in this project to train the dataset.

## 3.1 Feature Selection

Feature Selection is a crucial preprocessing step in machine learning that involves choosing a subset of relevant features from the original set. The objective is to enhance model performance, reduce complexity, and mitigate the risk of overfitting. Techniques like Recursive Feature Elimination with Cross-validation (RFECV) dynamically assess feature importance, ensuring the selected subset aligns with the training data. Effective feature selection optimizes model efficiency, interpretation, and generalization, contributing to streamlined, impactful machine learning workflows.

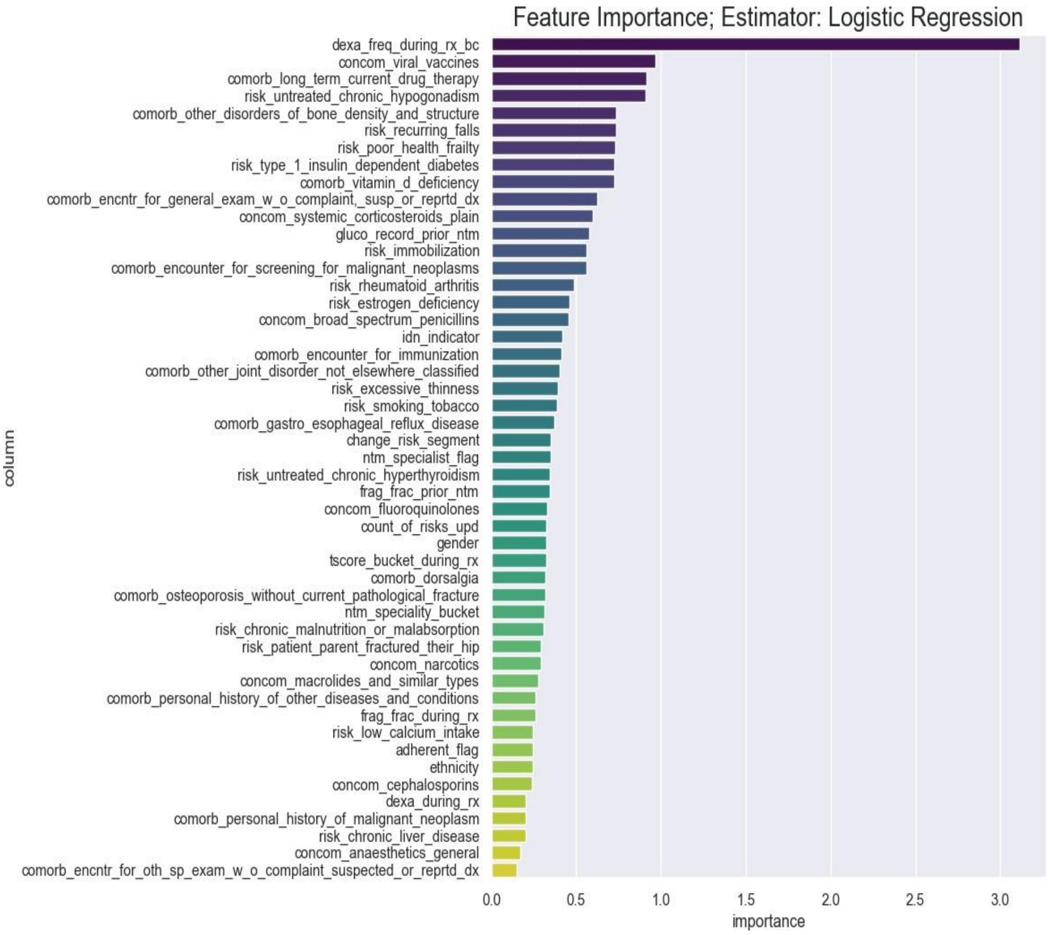


*Figure 16: Feature Importance of optimal features*

We implemented RFECV using Logistic Regression as the estimator and obtained 49 optimal features with accuracy of above 80% for model training, Figure 15. Based on the feature importance of each feature, Figure 16, a threshold of 0.5 is considered for selecting features for model training. A total 14 features were selected for model training.

## 3.2 Learning Models

Training the learning models also requires tuning the hyper-parameters for better performance. For this purpose, Grid Search with Cross Validation method was employed during each model training that helped in providing the best hyper-parameters. This iterative process aids in identifying the optimal hyperparameter configuration, fostering improved accuracy and generalization.

### 3.2.1 Logistic Regression

Logistic Regression, a fundamental statistical method, is widely used for binary classification tasks. This model estimates the probability of an event using the logistic or sigmoid function. The Grid Search method applied helped in selecting the best values for hyper-parameters like C (regularization strength) and penalty (regularization type).

### 3.2.2 Support Vector Classifier

The Support Vector Classifier (SVC) is a robust classification algorithm rooted in support vector machines (SVM). Known for its versatility, it adeptly delineates both linear and non-linear decision boundaries. Key parameters, including C for regularization and kernel type, allow tailored adjustments, enhancing adaptability. SVC excels in diverse domains, particularly suited for intricate datasets. This algorithm proves instrumental in scenarios requiring nuanced classification solutions, making it a valuable tool in machine learning.

### 3.2.3 Random Forest Classifier

The Random Forest Classifier is a potent ensemble learning method designed for classification tasks. Rooted in decision tree principles, it excels in capturing intricate patterns within data. Hyperparameter tuning, involving parameters like the number of estimators and tree depth, optimizes its performance. Renowned for handling complex relationships, Random Forest delivers competitive accuracy. The algorithm's strength lies in its ability to mitigate overfitting while providing robust predictions.

### 3.2.4 CatBoost Classifier

CatBoost Classifier, a proficient gradient boosting algorithm, is tailored for classification tasks, especially adept at handling categorical features. Rooted in decision tree algorithms, CatBoost optimizes performance through parameters like the number of iterations, learning rate, depth of the tree, and regularization. Its noteworthy attribute lies in mitigating overfitting and delivering high predictive accuracy. Embraced for efficiency in various domains, CatBoost outperforms competitors in intricate datasets.

# 4. Results

This section delves into the different performance metrics used to understand the performance of the models trained on the transformed dataset. The dataset was split into training, validation and test data with a proportion of 80:10:10 respectively. The dependent and independent variables were separated into input features and target variables.

## 4.1 Performance Metrics

### 4.1.1 TP, TN, FP, FN

The performance metrics are calculated based on the following values:

a. True Positive – It is a measure of positive samples that are predicted correctly as positive.
b. True Negative – It is a measure of negative samples that are predicted correctly as negative.
c. False Positive – It is a measure of negative samples that are predicted falsely as positive.
d. False Negative – It is a measure of positive samples that are predicted falsely as negative.

### 4.1.2 Accuracy, Precision, Recall, and F1-Score

Metrics that were employed:

- Accuracy - Accuracy is a ratio that measures the percentage of correct predictions out of the total predictions made by a model. It provides an overall assessment of how well the model is performing on the entire dataset. It is suitable when the classes are balanced. However, it may not be the best metric for imbalanced datasets.
  Accuracy = No. of correct predictions / Total no. of predictions

- Precision - Precision, also known as Positive Predictive Value, measures the accuracy of the positive predictions made by a classifier. It quantifies the ratio of correctly predicted positive instances to the total instances predicted as positive. It is useful when the cost of false positives (incorrectly predicting positive) is high, and you want to minimize the number of false alarms.
  Precision = True Positives / (True Positives + False Positives)

- Recall - Recall, also known as Sensitivity or True Positive Rate, measures the ability of a classifier to capture all the relevant positive instances. It quantifies the ratio of correctly predicted positive instances to the total actual positive instances. It is important in situations where cost of false negatives (missing positive instances) is high, and you want to ensure that all relevant positive instances are identified.
  Recall = True Positives / (True Positives + False Negatives)

- F1-Score - The F1 score provides a holistic assessment of a classifier's performance by considering both false positives and false negatives. It is a valuable metric in scenarios where there is a need to strike a balance between precision and recall, especially in situations with imbalanced datasets or when the consequences of false positives and false negatives are not equal.
  F1-score = 2 x (Precision x Recall) / (Precision + Recall)

In this case, where we have an imbalanced target, we need to make sure that our model maintains the trade-off between correctly identifying positive instances (minimizing false negatives) and avoiding unnecessary false positives. This is crucial because, in imbalanced datasets, the model might be biased towards the majority class, leading to high accuracy but poor performance in capturing instances of the minority class. F1-score and Accuracy were used as the metrics for analysing the performance of the Machine Learning models and deciding on the best model. Figure 17 and Figure 18 show the **F1-score** and **Accuracy** comparison of the 4 models used.

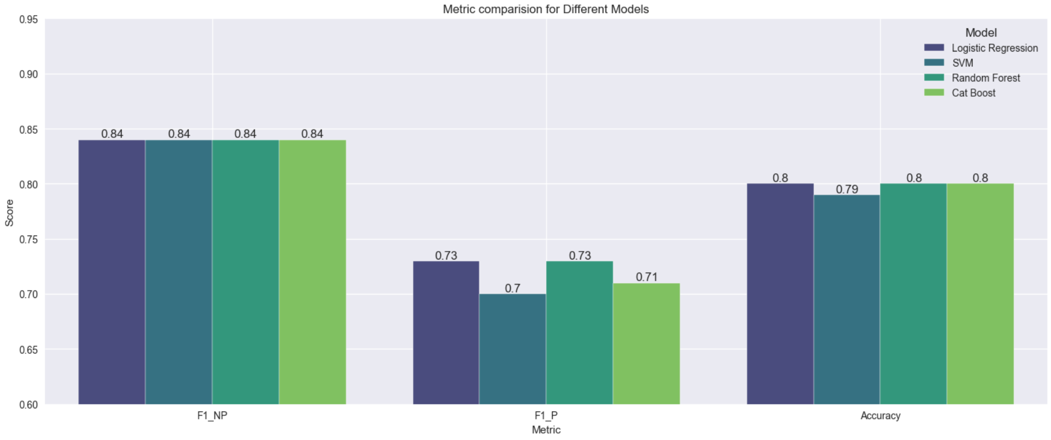


*Figure 17: Performance evaluation on validation data*

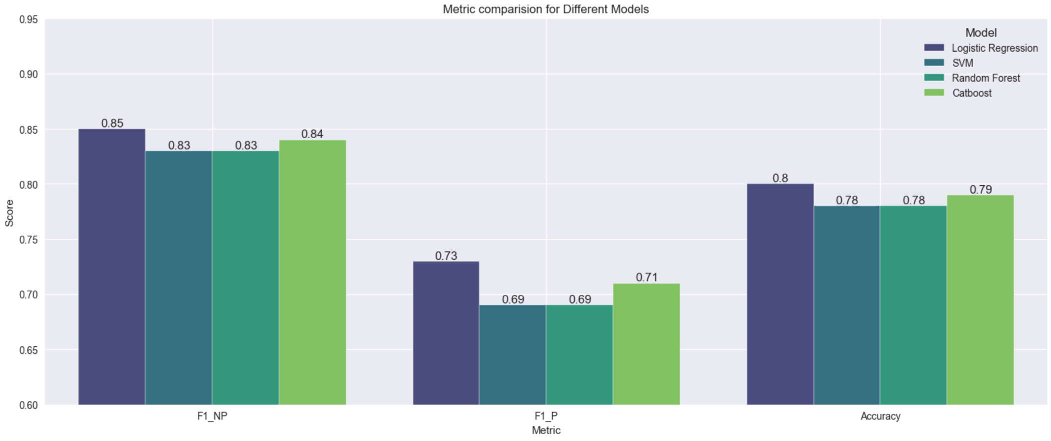


*Figure 18: Performance evaluation on test data*

After observing the models' performance on both validation and test data, we can clearly see that Logistic Regression performs the best as it gives highest accuracy and F1-score on validation data and same performance on test data, indicating model's generalization to unseen data.

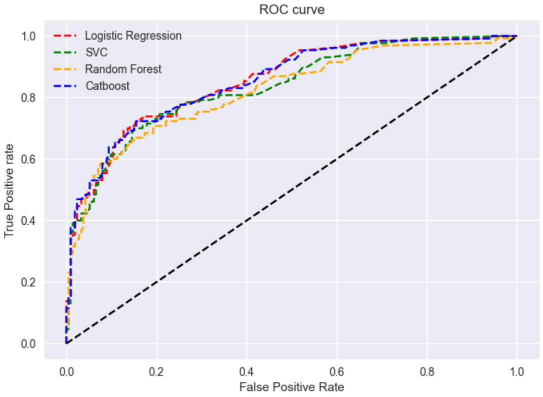


*Figure 19: ROC curve comparison between 4 models*

The ROC curve in Figure 19, clearly shows that Logistic Regression's performance over other models in discriminating between positive and negative instances on test data.

# 5. Deployment

The deployment section describes the building of the web page that takes inputs from the user, using Flask and Docker for creating APIs and containerising the application for hosting the application on cloud for easy and remote access irrespective of the underlying environment.

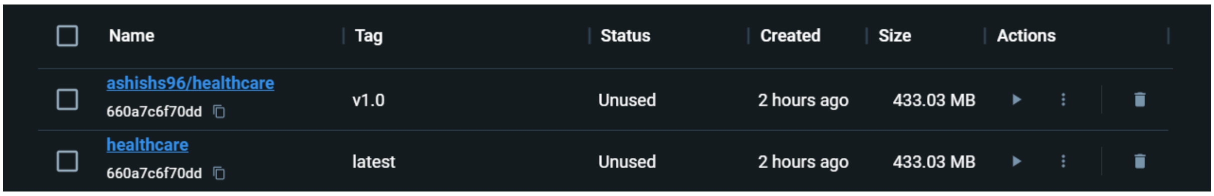
## 5.1 Application Building with Flask and Docker

Implementing the Flask framework requires a HTML page for the input by end user, an API for accessing the form values and passing the result through the form to the HTML page for displaying the prediction result. The prediction result is calculated using the saved trained model.

In this project's scenario, below are 4 different files helping in fulfilling the above requirements –

- *Healthcare_final.ipynb* – This python notebook generates the trained and evaluated Logistic Regression model.
- *app*.py – This python file acts as the interface for access form values and using these values to predict the result by loading the Logistic Regression model.
- *home.html* – The webpage required for access by the end users and also displays the prediction result.
- *style.css* – The style sheet for building an interactive and easy to use interface web page.

Once the application was built on local machine, a docker image was built to wrap the application in a container, Figure 20. This helps in deploying the application on any platform with out considering the underlying environment providing easy access to the application. The docker image is then pushed to the Docker repository for remote deployment or cloud deployment.



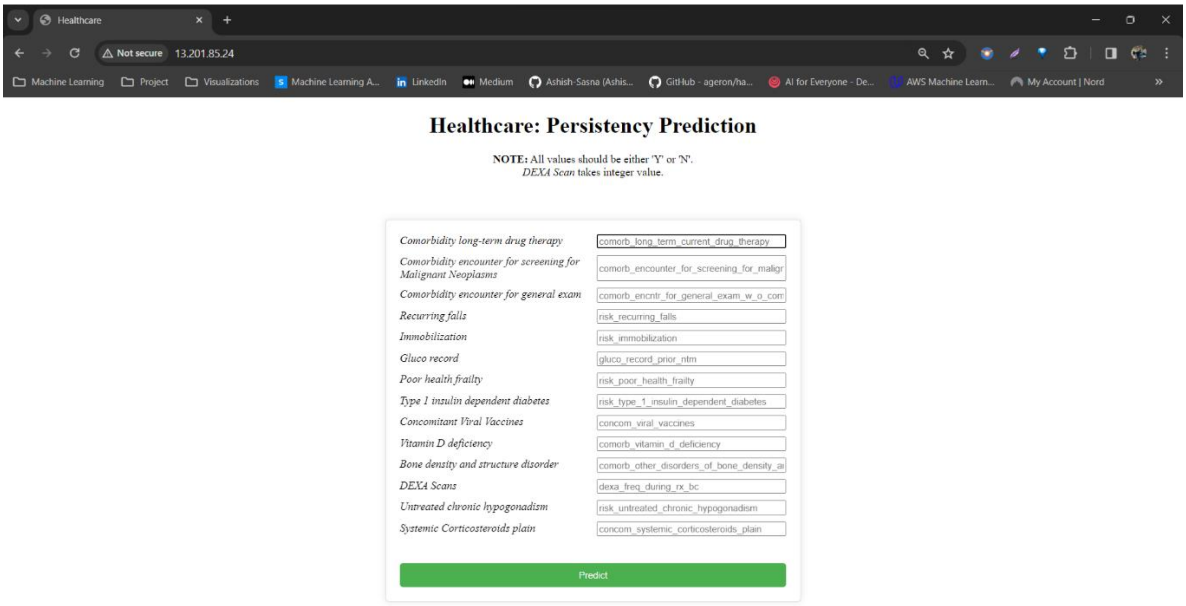
| □ | Name | Tag | Status | Created | Size | Actions |
|---|---|---|---|---|---|---|
| □ | ashishs96/healthcare<br>660a7c6f70dd | v1.0 | Unused | 2 hours ago | 433.03 MB | ▶ ⋮ 🗑 |
| □ | healthcare<br>660a7c6f70dd | latest | Unused | 2 hours ago | 433.03 MB | ▶ ⋮ 🗑 |

*Figure 20: Docker images of the Healthcare application*

## 5.3 Cloud Deployment

For the final part, we created an EC2 instance on AWS for deploying the web application. This involved pulling the Docker image saved in the Docker repository and establishing a connection to the web application with the help of a public IP address.

The Figure 21 shows the home page of the Healthcare web application which allows end users to enter the desired patient details and display the result after submitting the form using **Predict** button, Figure 22.

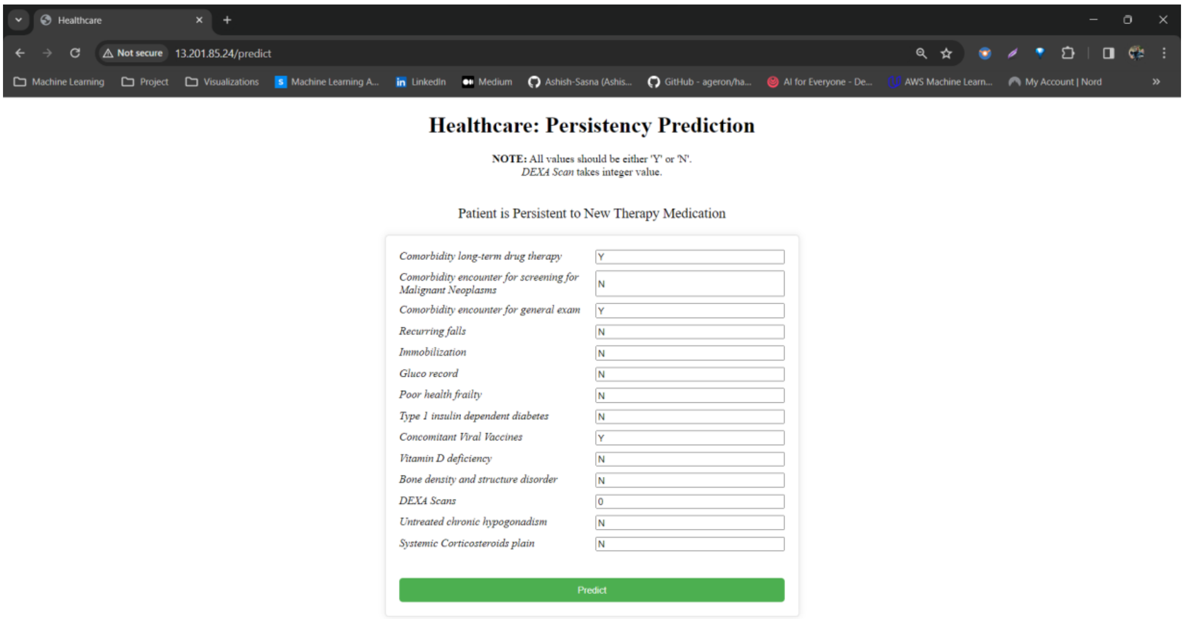


**Healthcare: Persistency Prediction**

**NOTE:** All values should be either 'Y' or 'N'.
*DEXA Scan* takes integer value.

| | |
|---|---|
| *Comorbidity long-term drug therapy* | comorb_long_term_current_drug_therapy |
| *Comorbidity encounter for screening for Malignant Neoplasms* | comorb_encounter_for_screening_for_maligr |
| *Comorbidity encounter for general exam* | comorb_encntr_for_general_exam_w_o_com |
| *Recurring falls* | risk_recurring_falls |
| *Immobilization* | risk_immobilization |
| *Gluco record* | gluco_record_prior_ntm |
| *Poor health frailty* | risk_poor_health_frailty |
| *Type 1 insulin dependent diabetes* | risk_type_1_insulin_dependent_diabetes |
| *Concomitant Viral Vaccines* | concom_viral_vaccines |
| *Vitamin D deficiency* | comorb_vitamin_d_deficiency |
| *Bone density and structure disorder* | comorb_other_disorders_of_bone_density_ar |
| *DEXA Scans* | dexa_freq_during_rx_bc |
| *Untreated chronic hypogonadism* | risk_untreated_chronic_hypogonadism |
| *Systemic Corticosteroids plain* | concom_systemic_corticosteroids_plain |

*Figure 21: Healthcare Home page*



**Healthcare: Persistency Prediction**

**NOTE:** All values should be either 'Y' or 'N'.
*DEXA Scan* takes integer value.

Patient is Persistent to New Therapy Medication

| | |
|---|---|
| *Comorbidity long-term drug therapy* | Y |
| *Comorbidity encounter for screening for Malignant Neoplasms* | N |
| *Comorbidity encounter for general exam* | Y |
| *Recurring falls* | N |
| *Immobilization* | N |
| *Gluco record* | N |
| *Poor health frailty* | N |
| *Type 1 insulin dependent diabetes* | N |
| *Concomitant Viral Vaccines* | Y |
| *Vitamin D deficiency* | N |
| *Bone density and structure disorder* | N |
| *DEXA Scans* | 0 |
| *Untreated chronic hypogonadism* | N |
| *Systemic Corticosteroids plain* | N |

*Figure 22: Healthcare Result page*

# 6. Conclusion/Future Scope

The Healthcare web application provides a seamless platform with and interactive user interface for end users such as Doctors, Hospitals or Pharmaceutical companies to help understand the effectiveness of the New Therapy Medication prescribed to patients with an accuracy of 80% along with good F1-scores providing optimal prediction results. The use of Machine Learning in this field provides an opportunity for different public or private organisations a way to access advanced technologies in the field of AI to solve complex business and health problems.

Deep Learning is another advancement in machine learning which has a vast range of applications. An approach for machine learning model building can be provided using neural networks. Neural networks help in extracting hidden features from the dataset. A further study can be applied for implementing neural networks that can help in avoiding use of pre-processing and data handling techniques. Many neural networks take raw data as input and extract low-level and high-level features.