Diabetic Hospital Readmission

Rohit Pradhan and Sai Ananya Kodiboyena

Abstract—**Approximately 15% of patients discharged after an acute hospitalization are readmitted within 30 days, leading to potentially worse clinical outcomes and billions of dollars in healthcare costs. Given these concerns, multiple quality efforts have been instituted in recent years to reduce readmission in the United States. For example, the Medicare Hospital Readmission Reduction Program (HRRP) was created as part of the Patient Protection and Affordable Care Act and financially penalizes U.S. hospitals with excess 30-day readmission rates among Medicare beneficiaries. Similar programs are being launched for patients with commercial insurance with the goal of further hospitals to reduce readmission.**

Not surprisingly, the development of these programs has led to an increased demand for statistical models that accurately predict readmission using available healthcare claims data. As the likelihood of readmission is related to key input features of patients (e.g., age, number of procedures, number of lab tests and time in hospital), differences in the distribution of patients across hospitals based on such features may lead to unfair penalization of hospitals that care for more at-risk individuals. Therefore, using statistical prediction models to adjust for patient risk across hospitals is a major priority for accountability programs. However, the performance of prediction models for readmission have been generally poor.

Reducing unplanned readmission is a major focus of current hospital quality efforts. To avoid unfair penalization, administrators and policymakers use prediction models to adjust for the performance of hospitals from healthcare claims data. Regression-based models are a commonly utilized method for such risk-standardization across hospitals; however, these models often suffer in accuracy. So, we use AUC/ROC as evaluation metrics for classification problems. In this study we compare various prediction models such as logistic regression, KNN, Naive Bayes, Random Forest, Boosting Techniques and Artificial Neural Network.

I. Introduction

It is increasingly recognized that the management of hyper- in the hospitalized patient has a significant bearing on outcome, in terms of both morbidity and mortality. This recognition has led to the development of formalized protocols in the intensive care unit (ICU) setting with rig- glucose targets in many institutions However, the same cannot be said for most non-ICU inpatient admissions. Rather, anecdotal evidence suggests that inpatient manage- is arbitrary and often leads to either no treatment at all or wide fluctuations in glucose when traditional man- strategies are employed. Although data are few, recent controlled trials have demonstrated that protocol-driven inpatient strategies can be both effective and safe. As such, implementation of protocols in the hospital setting is now recommended. However, there are few national assessments of diabetes care in the hospitalized patient which could serve as a baseline for change. The present analysis of a large clinical database was undertaken to examine historical patterns of diabetes care in patients with diabetes admitted to a US hospital and to inform future directions which might lead to improvements in patient safety. We examined the use of HbA1c as a marker of attention to diabetes care in many individuals identified as having a diagnosis of diabetes. We hypothesize that measurement of HbA1c is associated with a reduction in readmission rates in individuals admitted to the hospital.

Rohit Pradhan is with CS, Stevens Institute of Technology
Sai Ananya Kodiboyena is with ECE, Stevens Institute of Technology

Databases of clinical data contain valuable but heterogeneous and difficult data in terms of missing values, incomplete or inconsistent records, and high understood not only by number of features but also their complexity. Additionally, analyzing external data is more challenging than analysis of results of a carefully designed experiment or trial because one has no impact on how and what type of information was collected. Nonetheless, it is important to utilize these huge amounts of data to find new information/knowledge that is possibly not available anywhere.

II. Related Work

Previous studies revealed that admission times, age, sex are relevant to multiple hospitalizations. A length of stay longer than 5 days was associated with a greater than 87% risk of readmission compared to a length of stay shorter than or equal to 2 days. More admission times was the main component of patients who had more readmissions [1] , primarily elderly patients with more serious conditions, and the length of hospitalizations were much longer than that of general patients. Besides, the frequent diagnoses indicated that these patients had a higher probability of developing diabetes-related complications. Moreover, diag_2 was more important than diag_1 among the three diagnostic codes, indicating that the subsequent diagnosis in a patient's EHR could more accurately reflect the patient's condition.

III. Data Set Description

Health Facts is a voluntary program offered to organizations which use the Cerner Electronic Health Record System. The database contains data systematically collected from participating institutions electronic medical records and includes encounter data (emergency, outpatient, and inpatient), provider specialty, demographics (age, sex, and race), diagnoses and in-hospital procedures documented by ICD-9-CM codes, laboratory data, pharmacy data, in-hospital mortality, and hospital characteristics. All data were identified in compliance with the Health Insurance Portability and Accountability Act of 1996 before being provided to the investigators. Continuity of patient encounters within the same health system (EHR system) is preserved. The Health Facts data we used was an extract representing 10 years (1999–2008) of clinical care at 130 hospitals and integrated delivery networks throughout the United States: Midwest (18 hospitals), Northeast (58), South (28), and West (16). Most of the hospitals (78) have bed size between 100 and 499, 38 hospitals have bed size less than 100, and bed size of 14 hospitals is greater than 500 [2] .

The database consists of 41 tables in a fact-dimension schema and a total of 117 features. The database includes 74,036,643 unique encounters (visits) that correspond to 17,880,231 unique patients and 2,889,571 providers. Because this data represents integrated delivery network health - in addition to stand-alone hospitals, the data contains both inpatient and outpatient data, including emergency department, for the same group of patients. However, data from out-of-network providers is not captured.

The data set was created in two steps. First, encounters of interest were extracted from the database with 55 attributes.

Second, preliminary analysis and of the data were performed resulting in retaining only these features (attributes) and encounters that could be used in further analysis, that is, contain sufficient information. Both steps are described in the following subsections.

Extraction of the Initial from the Database.

Information was extracted from the database for encounters that satisfied the following criteria.

(1) It is an inpatient encounter (a hospital admission).

(2) It is a "diabetic" encounter, that is, one during which any kind of diabetes was entered into the system as a diagnosis.

(3) The length of stay was at least 1 day and at most 14 days.

(4) Laboratory tests were performed during the encounter.

(5) Medications were administered during the encounter.

Criteria 3-4 were applied to remove admissions for procedures and so forth, which were of less than 23 hours of duration, and in which changes in diabetes management were less likely to have occurred. It should be noted that the diabetic encounters are not all encounters of diabetic patients but rather only these encounters where diabetes was coded as an existing health condition. 101,766 encounters were identified to fulfill all the above five inclusion criteria and were used in further analysis. Attribute/feature selection was performed by our clinical experts and only attributes that were potentially associated with the diabetic condition or management were retained. From the information available in the database, we extracted 55 features describing the diabetic encounters, including demographics, diagnoses, diabetic medications, number of visits in the year preceding the encounter, and payer information [3], [4] .

Since we are primarily interested in factors that lead to early readmission, we defined the readmission attribute (outcome) as having two values: "readmitted," if the patient was readmitted within 30 days of discharge or "otherwise," which covers both readmission after 30 days and no readmission at all. The values of the readmission attribute were determined by examination of all patient records in the database to determine the first inpatient visit after discharge. Note that 30 days was chosen based on criteria often used by funding agencies. Hemoglobin A1c (HbA1c) is an important measure of glucose control, which is widely applied to measure performance of diabetes care. The measurement of HbA1c at the time of hospital admission offers a unique opportunity to assess the efficacy of current therapy and to make changes in that therapy if indicated (e.g., HbA1c &gt; 8.0% on current regimen). We considered the possibility that if an HbA1c test result was available from a measurement (outpatient or inpatient) done within three months prior to the sentinel admission, the test might not be repeated. In these cases (0.1% of the total), we used the measurement available from the previous visit. In all other cases, measurement of HbA1c was performed at the time of hospital admission. We examined both the frequency of HbA1c test ordering and the response to its result, which we defined as a change in diabetic medications. By a "change of medication" we understand any dosage change (increase or reduction) as well as change to [5] a drug with a different generic name, for example, a change of the type of insulin or an introduction of a new drug. The database contains detailed information about dosage but is restricted only to medications administered during the encounter. It was not possible to track any admission and discharge medications. We considered four groups of encounters: (1) no HbA1c test performed, (2) HbA1c performed and in normal range, (3) HbA1c performed, and the result is greater than 8% with no change in diabetic medications, and (4) HbA1c performed, result is greater than 8%, and diabetic medication was changed.

The original database contains incomplete, redundant, and noisy information as expected in any real-world data. There were several features that could not be treated directly since they had a high percentage of missing values. These features were weight (97% values missing), payer code (40%), and medical specialty (47%). Weight attribute was too sparse, and it was not included in further analysis. Payer code was removed since it had a high percentage of missing values, and it was not considered relevant to the outcome. Medical specialty attribute was maintained, adding the value "missing" to account for missing values [6] . Large percentage of missing values of the weight attribute can be explained by the fact that prior to the HITECH legislation of the American Reinvestment and Recovery Act in 2009 hospitals and clinics were not required to capture it in a structured format.

The variables chosen to control for patient demographic and illness severity, age, race, admission source, discharge disposition, primary diagnosis, medical specialty of the admitting physician, and time spent in hospital. To summarize, our consists of hospital admissions of length between one and 14 days that did not result in a patient death or discharge to a hospice. Each encounter corresponds to a unique patient diagnosed with diabetes, although the primary diagnosis may be different. During each of the analyzed encounters, lab tests were ordered, and medication was administered.

There were three target variables: no readmission, readmission &>30 days and readmission &<30 days. There was imbalance in the target variable, we converted the multi class classification problem to binary classification in order to balance the dataset.

Target variable is balanced with 53% of no admission and 46% of re admissionFigure 1 .

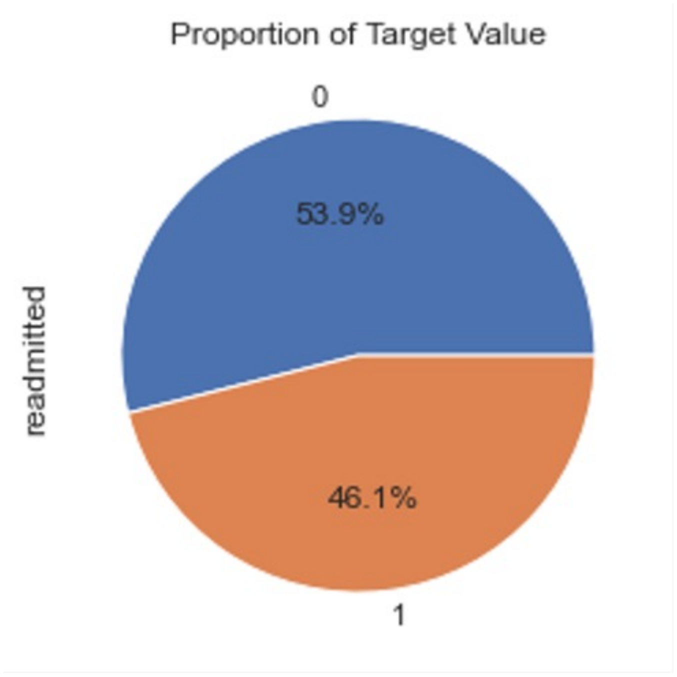


Fig. 1. Proportion of target variable

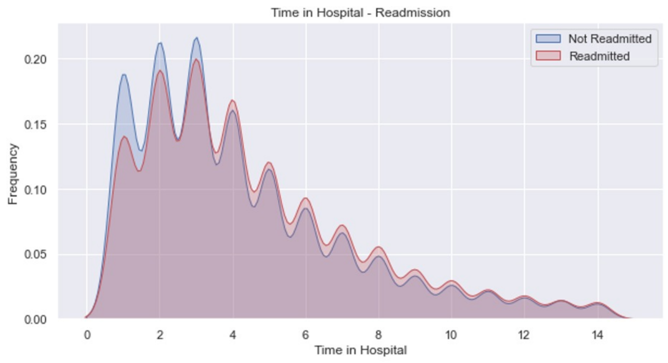


Fig. 2. Density curve for time in hospital feature

Most people stayed 2-4 days in hospital. From this curve shown in it is evident that most of the patients are admitted for 2-4 days. This is the density curve for time in hospital featureFigure 2 .

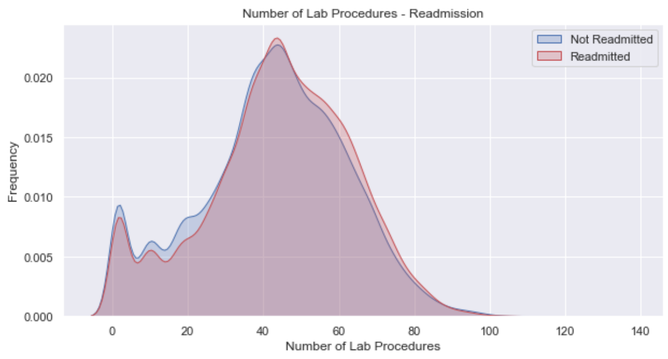


Fig. 3. Lab Procedures

Most of the patients are diagnosed with 40 to 60 procedures as shown in Figure 3 .

We see that irrespective of the number of lab procedures the readmission rate is the sameFigure 4 .

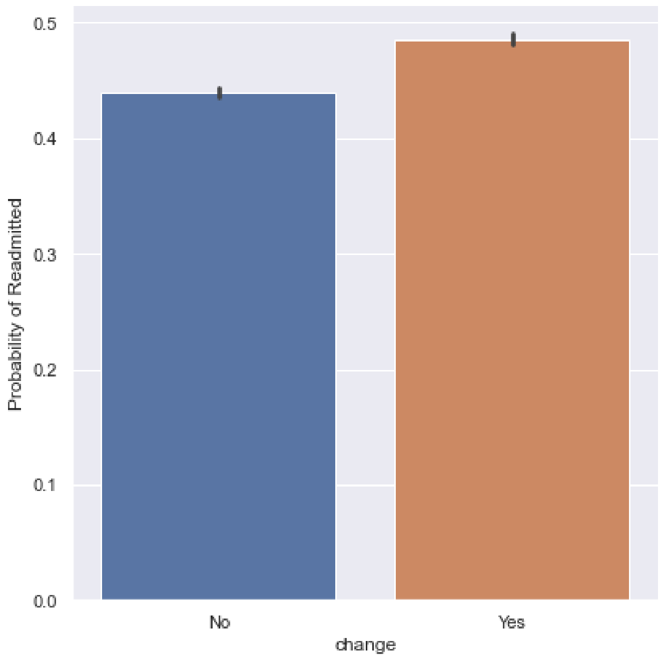


Fig. 4. Probability of readmission

We see that the Probability of readmission is more when there is alteration in the medicationFigure 5 .

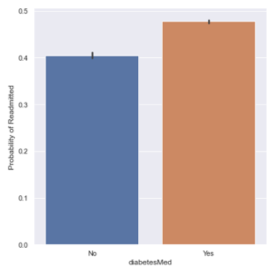


Fig. 5. Diabetes Medication

These graphsFigures 6 and 7 show the number of patients who were prescribed diabetes medications VS Readmission.

This graphFigure 7 shows non-linearity in data. Such non linearity was dealt with log transformation.

A. Feature Engineering and Feature Selection

First,we performed log transformation on the dataset columns which suffered from skewness. This helps us to normalize our data.

Multicollinearity is a condition when there is a significant dependency or association between the independent variables or the predictor variables. A significant correlation between the independent variables is often the first evidence of presence of multicollinearity.

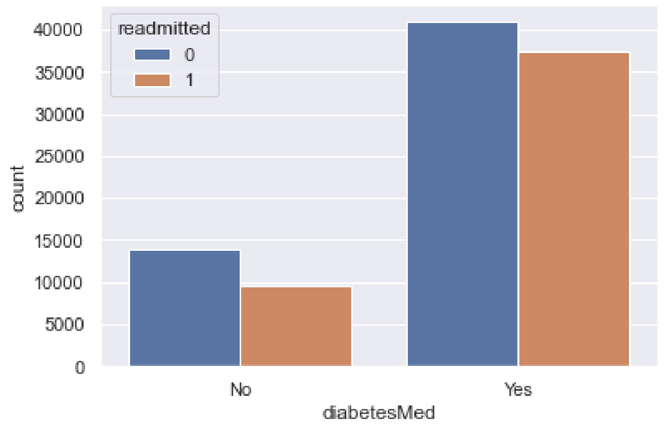


Fig. 6. Readmission and Medications

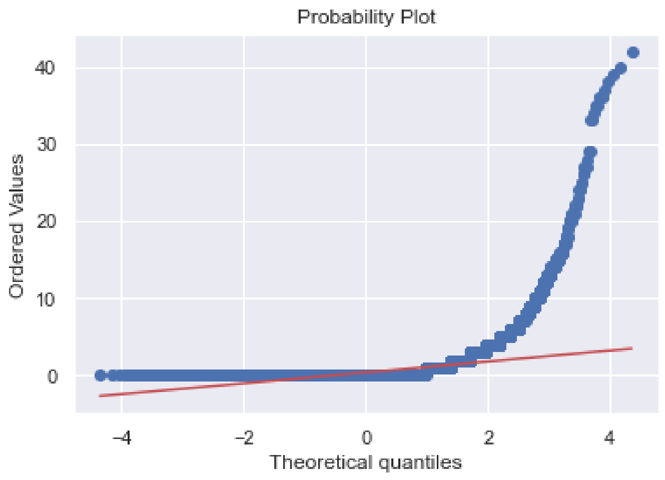


Fig. 7. Non-linearity in data

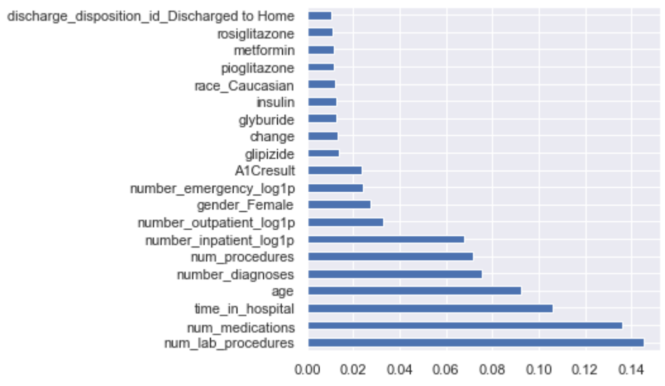
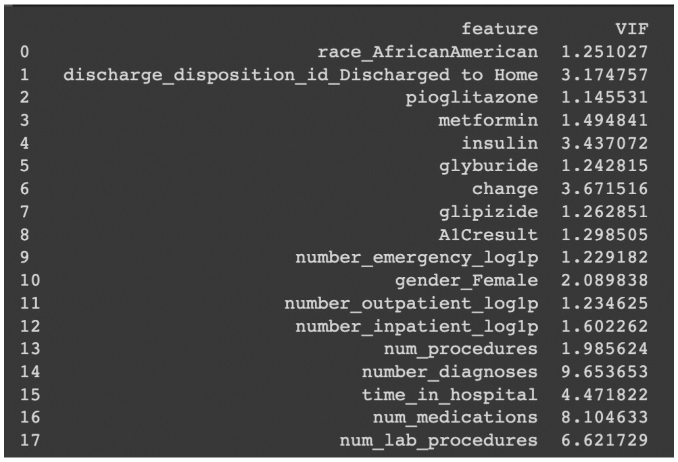


Fig. 8. Top features

To overcome the issue of multicollinearity we perform Variance inflation factor for all variables. Then we finally remove variables which have high VIF Figure 9.



```
                                      feature       VIF
0                          race_AfricanAmerican  1.251027
1   discharge_disposition_id_Discharged to Home  3.174757
2                                  pioglitazone  1.145531
3                                     metformin  1.494841
4                                       insulin  3.437072
5                                     glyburide  1.242815
6                                        change  3.671516
7                                     glipizide  1.262851
8                                     A1Cresult  1.298505
9                        number_emergency_log1p  1.229182
10                                gender_Female  2.089838
11                      number_outpatient_log1p  1.234625
12                       number_inpatient_log1p  1.602262
13                               num_procedures  1.985624
14                             number_diagnoses  9.653653
15                             time_in_hospital  4.471822
16                              num_medications  8.104633
17                          num_lab_procedures  6.621729
```

Fig. 9. VIF

To verify correlation between variables we plot correlation matrix and the results are shown inFigure 10 .

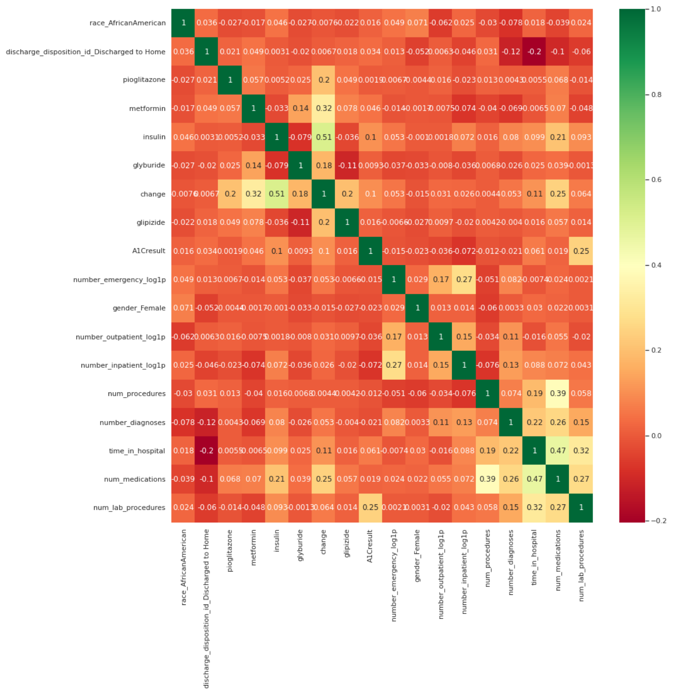


Fig. 10. correlation heat map

After Data cleaning and removing multicollinearity from data we have our important feature as shown in Figure 8.

IV. Hyperparameter Tuning

A. Overview of hyperparameters

Hyperparameters are variables that control a machine learning model's learning process [7] . They ultimately dictate how the model learns a specific relationship between input and predictions.

Optimizing a model's hyperparameters to solve a specific problem is often necessary. Importance of optimizing parameters:

- Machine learning hyperparameters are not one-size-fits-all as these models can require different constraints to generalize different out-of-sample data patterns and different problems.
- Hyperparameter optimization allows us to yield an optimal model with the best sets of hyperparameters. This model should be capable of giving optimal results that minimizes loss function.

B. Optimization methods

The problem with optimization problems is that, often the search space can be indefinite and the budget to perform this search is constrained (maximum x amount of time or iterations). Therefore, to search for the global minimum efficiently, an algorithm has to implement ways to utilize the budget efficiently.

C. Traditional methods

The traditional way of performing hyperparameter optimization has been exhaustive and uninformed about previous information. Some examples are:

Grid search, exhaustively searches through a predefined subset. Random search selects randomly from a predefined subset.

As data volume and hyperparameter space grow, some of the traditional search methods deteriorate rapidly. They either spend too much time or can't even locate the minima.

D. Bayesian method

Bayesian optimization methods search for global optimization by iteratively building a probabilistic model of the function mapping from hyperparameter values to the objective function [8] . The probabilistic model captures beliefs about the behavior of the function to form a posterior distribution over the objective function.

After that, the posterior distribution is further used to form an acquisition function that determines the next point with the best improvement probability.

However, as it uses available information to make decisions, an exploration vs exploitation problem arises.

Exploration is similar to a global search; we are interested in exploring the hyperparameter search space looking for even better solutions. Exploitation is similar to a local search; we want to refine our current solution and try to avoid wasting precious resources on undesirable search spaces.

E. Evolutionary method

Evolutionary optimization methods search for the space of hyperparameters using evolutionary algorithms [9] . This method was inspired by the concept of evolution by Charles Darwin. It generally follows a process following the biological concept of evolution:

An initial sample is drawn from the population hyperparameters search space.The hyperparameters are evaluated using a fitness function and ranked by their relative fitness.The worst-performing hyperparameters are discarded and new hyperparameter sets are generated through crossover and mutation.The evolutionary processes are repeated until a Budget constraint or until performance does not improve.

F. Optuna

Optuna is an advanced hyperparameter optimization framework with visualizations for interpretability. Optuna enables users to adopt state-of-the-art algorithms for sampling hyperparameters and pruning unpromising trials [10] . This helps to speed up optimization time and performance greatly compared to traditional methods such as GridSearch. It also allows users to plot optimization histories for a better understanding of the model.Optuna is a hyperparameter optimization software framework that is able to easily implement different state-of-the-art optimization methods to perform hyperparameter optimization rapidly with great performance.

By default, Optuna implements a Bayesian optimization algorithm (TPE) but it can be easily switched to other existing algorithms in the package.

Optuna names its optimization algorithms into two different categories, namely the sampling strategy and the pruning strategy [11] .

1) Sampling strategy: Algorithms that select the best parameter combination by concentrating on areas where hyperparameters are giving better results.Pruning strategy: Early-stopping-based optimization method.

We will briefly discuss the intuition behind an example for the three types of algorithms discussed in the previous section so that we can have a general understanding of how these algorithms work [12] .

a) TPESampler (Tree-Structured Parzen Estimator): A Bayesian optimization algorithm that:

First, randomly selects a subset of hyperparameters and sort them based on their scores.The hyperparameters are further divided into two groups based on some predefined quantile.The two groups are then modeled into estimated densities l(x1) and g(x2) using Parzen Estimators (kernel density estimators).Locate the hyperparameters with highest expected improvement [lowest l(x1)/g(x2)].The hyperparameter with highest expected improvement are evaluated, sorted and divided again. This process repeats until the Budget is finished and the best hyperparameters will be returned.

b) NSGAIISampler (Non-dominated sorting genetic algorithm II): A Evolutionary optimization algorithm with multiple objective functions.

An individual (A) is said to dominate another individual (B), if

There is no objective of A worse than that objective of BThere is at least one objective of A better than that objective of B.

Main process of the algorithm:

Initially, a random parent population is sampled and each solution evaluated is assigned a fitness rank equal to its nondomination level. It first selects all the non-dominated solutions from population P and assigns them to rank 1, then selects all from the remaining solutions and assigns them to rank 2, and so on so forth, until all individuals have been assigned to a rank.Two random trials are picked and the better one becomes Parent 1. The process is repeated once to select another Parent 2 [Binary tournament selection].These two parents recombine to produce offspring which go into the child population [Recombination]. The child undergoes mutation and changes some of its values [Mutation]. This step is repeated until you have twice the initial population size.The population is sorted according to nondomination again. A new generation will be chosen in the order of rank. Crowding-sort will be implemented to calculate the density of solutions if the particular rank is only partially included for the next generation. The less dense trials are chosen into the next generation until the population count reaches the initial population size again.The new generation are reproduced and discarded again iteratively until the maximum number of generations is met and the best hyperparameters will be returned.

Some other popular sampling strategies: CMA-ES Sampler, MOTPE Sampler

2) Pruning Strategy:

a) SuccessiveHalvingPruner (Asynchronous Successive Halving): Randomly selects an initial set of hyperparameter values.Train the trials for 1 epoch until the defined maximum number of trials is reached.At the same time, a trial is simultaneously promoted to another rung (similar to rank) to train more epochs whenever the trial's score is among the top d percent within the rung where d is a predefined divisor.

Note that this is different to Synchronous Successive Halving where the algorithm waits for all defined trials in a rung to finish their epochs and only then decides which trials are having the top scores to be promoted to train more epochs in another rung.

Some other popular pruning strategies: MedianPruner, HyperbandPruner

Before we start implementing optimization with Optuna, it requires us to define an objective function.

The objective function will contain the entire logic of a regular model definition, training and testing process. After the model evaluation, it should return the evaluation metric which is also picked by the user.

The Trial class will be used to store information of one specific combination of hyperparameters later used by the machine learning model.

A Study object can then be called to optimize the objective function to find the best hyperparameters combination. It will then run trials iteratively until a user-defined maximum trial or time. The trial with the best hyperparameters will be stored in study.best_trial.

V. Machine Learning Algorithms

A. KNN

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

1. Ease to interpret output
2. Calculation time
3. Predictive Power

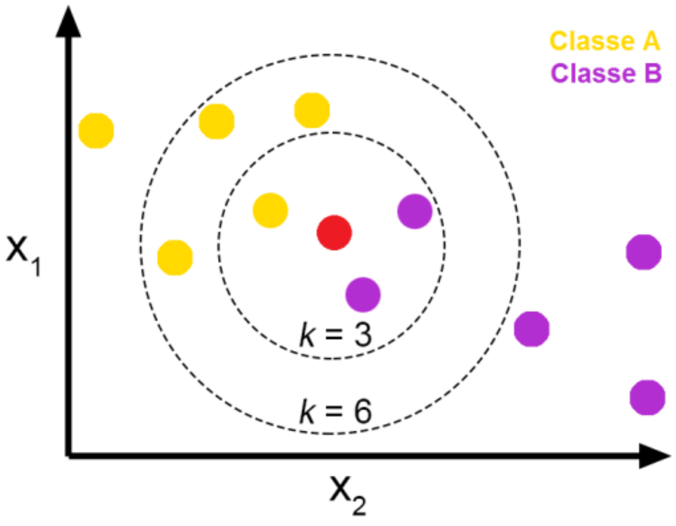


Fig. 11. K-Nearest Neighbor

K-NN is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for underlying data distribution i.e. the model structure is determined from the data set. It is called Lazy algorithm because it does not need any training data points for model generation. All training data is used in the testing phase which makes training faster and testing phase slower and costlier.

K-Nearest Neighbor (K-NN) is a simple algorithm that stores all the available cases and classifies the new data or case based on a similarity measure.

In K-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor. To determine which of the K instances in the training data set are most similar to a new input, a distance measure is used. For real-valued input variables, the most popular distance measure is the Euclidean distanceFigure 11 .

Other popular distance measures are:

Hamming Distance: Calculate the distance between binary vectors.

Manhattan Distance: Calculate the distance between real vectors using the sum of their absolute difference. Also called City Block Distance.

Minkowski Distance: Generalization of Euclidean and Manhattan distance.

Performance of the K-NN algorithm is influenced by three main factors :

The distance function or distance metric used to determine the nearest neighbors.

The decision rule used to derive a classification from the K-nearest neighbors.

The number of neighbors used to classify the new example.

1) Elbow Method in Supervised Machine Learning (Optimal K Value): The most important step in k-Nearest Neighborhood supervised machine learning is to determine the optimal value of K; that is, how many clusters your data should be divided into?

The optimal value of k reduces the effect of the noise on the classification, but makes boundaries between classes less distinct. Elbow method helps to select the optimal number of clusters for KNN clustering. It is one of the most popular methods to determine this optimal value of K. Because the user must specify in advance what k to choose, the algorithm is somewhat naive it assigns all members to k clusters even if that is not the right k for the data set.

To find optimal K we performed elbow method on our data set and found optimal K as follows Figure 12 :

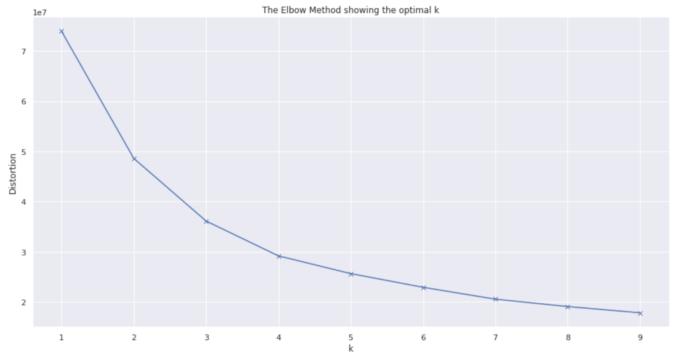


Fig. 12. Elbow method showing the optimal k

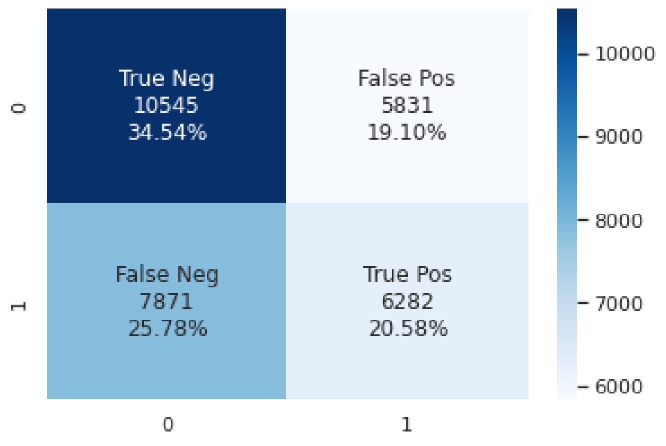


Fig. 13. Confusion matrix of KNN

Figure 13 shows the confusion matrix for KNN.

The best parameters after hyperparameter tuning with Gridsearch CV with cv=10 :

BEST PARAMS: {'learning_rate': 1, 'n_estimators': 100}

Hyperparameter importance with optuna is shown in Figure 14.

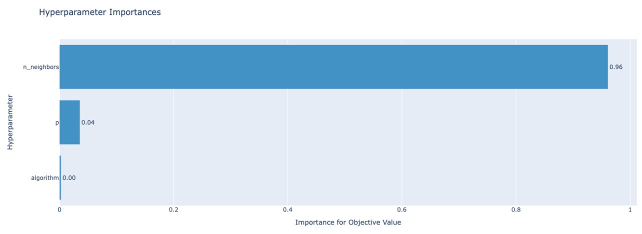


Fig. 14. Hyperparameter importance for KNN with optuna

B. Naïve Bayes

It is a supervised classification algorithm based on probability. It assumes that each feature is independent and accordingly calculates the posterior probability based on the target class. It assigns the unseen data to the class label which has maximum posterior probability.

Figure 15 shows confusion matrix for Naive Bayes

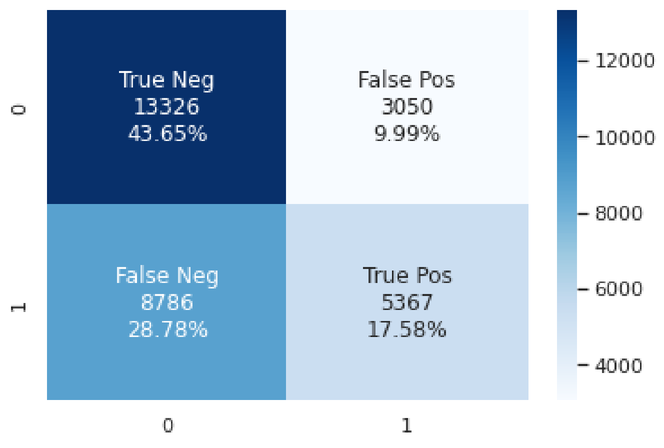


Fig. 15. Confusion matrix for Naive Bayes

C. Logistic Regression

Logistic Regression is a Supervised statistical technique to find the probability of dependent variable(Classes present in the variable).Logistic regression uses functions called the logit functions,that helps derive a relationship between the dependent variable and independent variables by predicting the probabilities or chances of occurrence [13] .The logistic functions (also known as the sigmoid functions) convert the probabilities into binary values which could be further used for predictions.

Types of Logistic Regression:

Binary Logistic Regression:The dependent variable has only two 2 possible outcomes/classes.Example-Male or Female.

Multinomial Logistic Regression:The dependent variable has only two 3 or more possible outcomes/classes without ordering. Example: Predicting food quality.(Good, Great and Bad).

Ordinal Logistic Regression:The dependent variable has only two 3 or more possible outcomes/classes with ordering. Example: Star rating from 1 to 5.

Even Though Logistic Regression belongs to the Linear models, it does not make any assumptions of the Linear Regression models, like:

→ It does not require linear relationship between dependent and independent variables.

→ The error terms do not need to be normally distributed.

→ Homoscedasticity is not required.

However,it has few of it's own assumption, It assumes that there is minimal,or no multi-collinearity among the independent variables.The best way to check the prescence of multi-collinearity is to perform VIF(Variance Inflation Factor).It assumes that independent variables that linearly related to log of odds.It can be checked with the Box-Tidwell test.It assumes a large sample for good prediction.It assumes that the observations are independent of each other.There is no influential values(outliers) in the continous predictors(independent variables).This can be checked with the help of IQR,z-score or can be visualized using box or violin plots.Logistic Regression with 2 classes that the dependent variable is binary and the ordered Logistic Regression.

Odds ratio is defined as the ratio of the odds in presence of B and odds of A in the absence of B and vice versa.

In other words, Odds are the ratio of the probability of success to the probability of failure and Logit is Just the Log of the Odds Ratio.

Formula of Odds is:

$odds = \frac{P}{1-P}$

If we want the odds ratio between binary classes then:

$$oddsratio = \frac{\frac{P1}{1-P1}}{\frac{P0}{1-P0}}$$

Logit Function is just the log of odds and the formula is :

$Logit\ function = \log\left(\frac{P}{1-p}\right)$

In Logistic regression we can calculate odds ratio between the classes:

$\ln\left(\frac{P}{1-P}\right) = a + bX$

$\frac{P}{1-P} = e^{a+bX}$

$P = \frac{e^{a+bX}}{1+e^{a+bX}}$

Now, that you've understood what odds ratio is let's see what decision boundary is:

A decision Boundary is a line or margin that separates the classes. Classification algorithm is all about finding the decision boundary that helps distinguish between the classes perfectly or close to perfect Logistic Regression decides a proper fit to the decision boundary so that we will be able to predict which class a new data will correspond toFigure 16 .

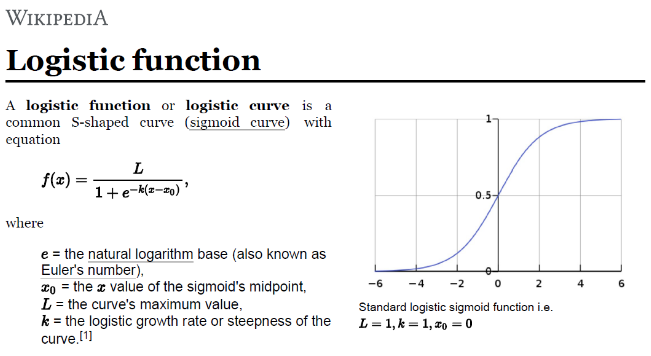
Cost Function is a function that measures the performance of a Machine Learning model for given data. Cost Function is basically the calculation of the error between predicted values and expected values and presents it in the form of a single real number. Many people gets confused between Cost Function and Loss Function, Well to put this in simple terms Cost Function is the average of error of n-sample in the data and Loss Function is the error for individual data points.In other words,Loss Function is for one training example, Cost Function is the for the entire training set.

$\cos t\,(h_\theta(x), y) = -\log(h_\theta(x)) \quad if\ y = 1$

$\cos t\,(h_\theta(x), y) = -\log(1 - h_\theta(x)) \quad if\ y = 0$

Let's have a look at the logistic(sigmoid) function.

$f(x) = \frac{1}{1+e^{-(x)}}$



WIKIPEDIA

Logistic function

A **logistic function** or **logistic curve** is a common S-shaped curve (sigmoid curve) with equation

$$f(x) = \frac{L}{1+e^{-k(x-x_0)}},$$

where

e = the natural logarithm base (also known as Euler's number),
x_0 = the x value of the sigmoid's midpoint,
L = the curve's maximum value,
k = the logistic growth rate or steepness of the curve.[1]

Standard logistic sigmoid function i.e. $L = 1, k = 1, x_0 = 0$

Fig. 16. Logistic function

Here, x = mx+b or x = b0 + b1x

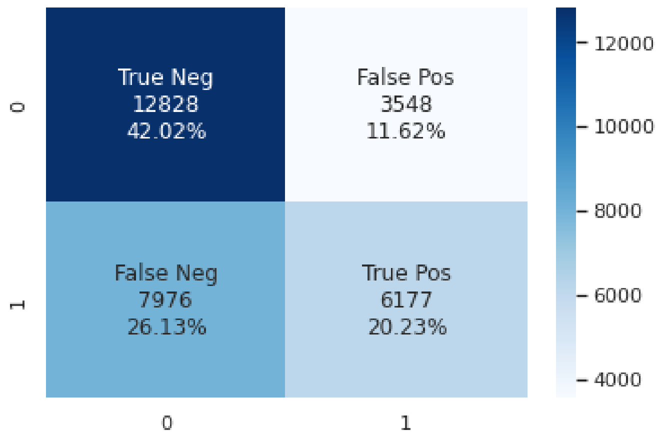


Fig. 17. Logistic Regression confusion matrix

Figure 17 shows the confusion matrix of logistic regression.

The best parameters after hyperparameter tuning with GridsearchCV with cv=10 :

BEST PARAMS: {'C': 0.03359818286283781, 'penalty': 'l1', 'solver': 'liblinear'}

Hyperparameter importance with optuna is shown in Figure 18.

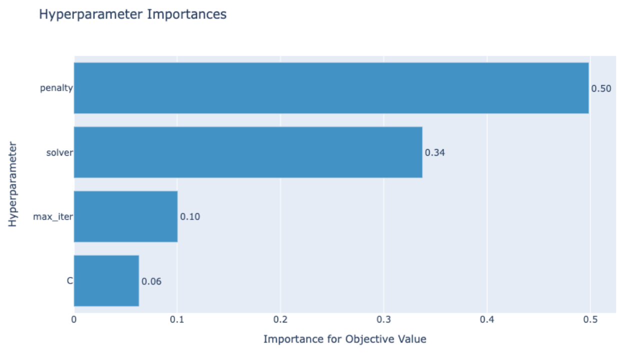


Fig. 18. Hyperparameter importance with optuna

D. Ensemble Techniques

Ensembles can be of two types —

ii) Boosting — Boosting is a ML algorithm in which the weak learners are converted into strong learners. Weak learners are classifiers which always perform slightly better than chance irrespective of the distribution over the training data. In Boosting, the predictions are sequential wherein each subsequent predictor learns from the errors of the previous predictors. Gradient Boosting Trees (GBT) is a commonly used method in this category.

1) Random Forest: It is an ensemble technique of classification, widely used in classification and regression problems (CART). It stacks multiple decision trees.

Gradient Boosting Machine (GBM): A Gradient Boosting Machine combines the predictions from multiple decision trees to generate the final predictions. Here, decision trees are the weak learners.

The nodes in every decision tree take a different subset of features for selecting the best split. Each new tree is sequentially built, meaning it considers the errors of the previous trees. Every successive decision tree is built on the errors of the previous trees.

Figure 20 shows the confusion matrix of random forest.

The best parameters after hyperparameter tuning with Grid-searchCV with cv=10 :

BEST PARAMS: {'criterion': 'entropy', 'max_depth': 4, 'max_features': 'log2', 'max_samples': 0.2, 'min_samples_split': 5, 'n_estimators': 300}

Hyperparameter importance with optuna is shown in Figure 19

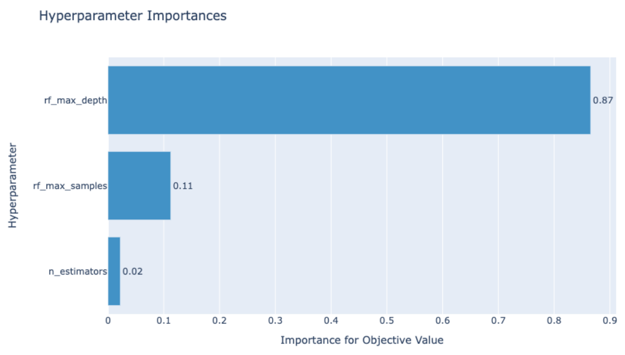


Fig. 19. Hyperparameter importance of Random forest with optuna

2) XGBoost (eXtreme Gradient Boosting) : Works on the same principal of GBM. It is an advanced version of GBM with advantages as follows:

It provides better, which drives fast learning through parallel and distributed computing and offers efficient memory usage. It has variety of regularization techniques that reduce over fitting and improve overall performance. The XGBM model takes care of missing values on its own. The model learns whether missing values should be in the right or left node. Large number of trees might lead to over fitting. So, it is necessary to carefully choose the stopping criteria for boosting.

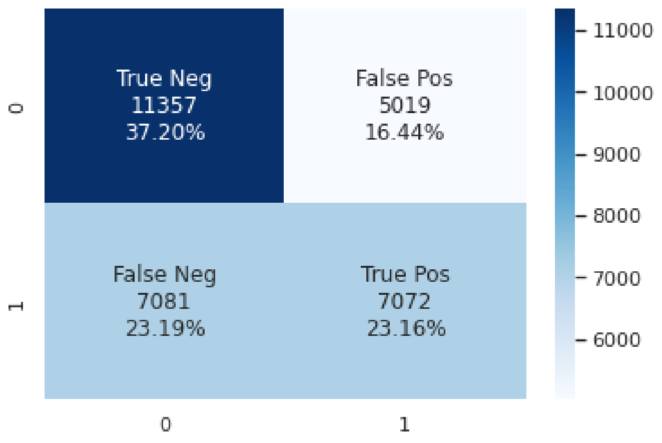


Fig. 20. confusion matrix of random forest

Bias and Variance — While building any model, our objective is to optimize both variance and bias but in the real world scenario, one comes at the cost of the other. It is important to understand the trade-off and figure out what suits our use case.

Ensembles are built on the idea that a collection of weak predictors, when combined together, give a final prediction which performs much better than the individual ones.

Performance comparison of these two methods in reducing Bias and Variance — Bagging has many uncorrelated trees in the final model which helps in reducing variance [14] . Boosting will reduce variance in the process of building sequential trees. At the same time, its focus remains on bridging the gap between the actual and predicted values by reducing residuals, hence it also reduces bias.

Gradient Boosting refers to a methodology in machine learning where an ensemble of weak learners is used to improve the model performance in terms of efficiency, accuracy, and interpretability. These learners are defined as having better performance than random chance. Such models are typically decision trees and their outputs are combined for better overall results.

The hypothesis is to filter out instances that are difficult to accurately predict and develop new weak learners to handle them.

The initial model is trained and predictions are run on the whole dataset. The error between the actual value and prediction is calculated and more weightage is given to the incorrect predictions. Subsequently, a new model that attempts to fix the error of the previous model and in a similar way several models are thus created. We arrive at the final model by weighting the mean of all models.

While constructing any model using GBT, the values of the following can be tweaked for improvement in model performance -

number of trees

learning rate — Scales the contribution of each tree as discussed before. There is a trade-off between learning rate and number of trees. Commonly used values of learning rate lie between 0.1 to 0.3

maximum depth — Maximum depth of each estimator. It limits the number of nodes of the decision trees

The advantages are as follows:

Faster training speed and accuracy resulting from LightGBM being a histogram-based algorithm that performs bucketing of values (also requires lesser memory)

Also compatible with large and complex datasets but is much faster during training support for both parallel learning and GPU learning

In contrast to the level-wise (horizontal) growth in XGBoost, LightGBM carries out leaf-wise (vertical) growth that results in more loss reduction and in turn higher accuracy while being faster. But this may also result in overfitting on the training data which could be handled using the max-depth parameter that specifies where the splitting would occur. Hence, XGBoost is capable of building more robust models than LightGBM.

XGBoost (eXtreme Gradient Boosting) is a machine learning algorithm that focuses on computation speed and model performance. The algorithm can be used for both regression and classification tasks and has been designed to work with large and complicated datasets.

Results for gradient boosting

The best parameters after hyperparameter tuning with GridsearchCV with cv=10 :

BEST PARAMS: {’criterion’: ’friedman_mse’, ’max_depth’: 2, ’n_estimators’: 300}

Hyperparameter importance with optuna is shown in Figure 21

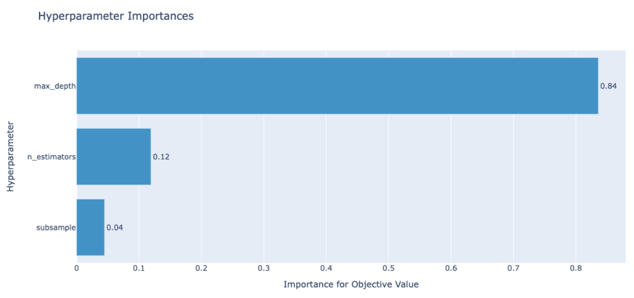


Fig. 21. Hyperparameter importance with optuna

Algorithm Enhancements:

Tree Pruning — Pruning is a machine learning technique to reduce the size of regression trees by replacing nodes that don’t contribute to improving classification on leaves. The idea of pruning a regression tree is to prevent overfitting of the training data. The most efficient method to do pruning is Cost Complexity or Weakest Link Pruning which internally uses mean square error, k-fold cross-validation and learning rate. XGBoost creates nodes (also called splits) up to max_depth specified and starts pruning from backward until the loss is below a threshold. Consider a split that has -3 loss and the subsequent node has +7 loss, XGBoost will not remove the split just by looking at one of the negative loss. It will compute the total loss (-3 + 7 = +4) and if it turns out to be positive it keeps both.Sparsity Aware Split Finding — It is quite common that the data we gather has sparsity (a lot of missing or empty values) or becomes sparse after performing data engineering (feature encoding). To be aware of the sparsity patterns in the data, a default direction is assigned to each tree. XGBoost handles missing data by assigning them to default direction and finding the best imputation value so that it minimizes the training loss. Optimization here is to visit only missing values which make the algorithm run 50x faster than the naïve method.

System Enhancements:

Parallelization — Tree learning needs data in a sorted manner. To cut down the sorting costs, data is divided into compressed blocks (each column with corresponding feature value). XGBoost sorts each block parallelly using all available cores/threads of CPU. This optimization is valuable since a large number of nodes gets created frequently in a tree. In summary, XGBoost parallelizes the sequential process of generating trees.Cache Aware — By cache-aware optimization, we store gradient statistics (direction and value) for each split node in an internal buffer of each thread and perform accumulation in a mini-batch manner. This helps to reduce the time overhead of immediate read/write operations and also prevent cache miss. Cache awareness is achieved by choosing the optimal size of the block (generally 2^{16}).

Flexibility in XGBoost:

Customized Objective Function — An objective function intends to maximize or minimize something. In ML, we try to minimize the objective function which is a combination of the loss function and regularization term.

Optimizing the loss function encourages predictive models whereas optimizing regularization leads to smaller variance and makes prediction stable. Different objective functions available in XGBoost are:

- reg: linear for regression
- reg: logistic, and binary: logistic for binary classification
- multi: softmax, and multi: softprob for multiclass classification

2. Customized Evaluation Metric — This is a metric used to monitor the model’s accuracy on validation data.

- error — Binary classification error (Classification)
- logloss — Negative log-likelihood (Classification)
- auc — Area under the curve (Classification)

Most important XGBoost parameters:

n_estimators – Number of trees in the ensemble. A higher value means more weak learners contribute towards the final output but increasing it significantly slows down the training time.

max_depth [default 3] – This parameter decides the complexity of the algorithm. The lesser the value assigned, the lower is the ability for the algorithm to pick up most patterns (underfitting). A large value can make the model too complex and pick patterns that do not generalize well (overfitting).

learning_rate/ eta [default 0.3] – The rate of learning of the model is inversely proportional to the accuracy of the model. Lowering the learning rate, although slower to train, improves the ability of the model to look for patterns and learn them. If the value is too low then it raises difficulty in the model to converge.

subsample – Similar to colsample_bytree, the subsample parameter instructs the algorithm on the fraction of the total number of instances to be used for a tree during training. This also reduces the chances of overfitting and improves training time.

Figure 22 shows confusion matrix of XGB.

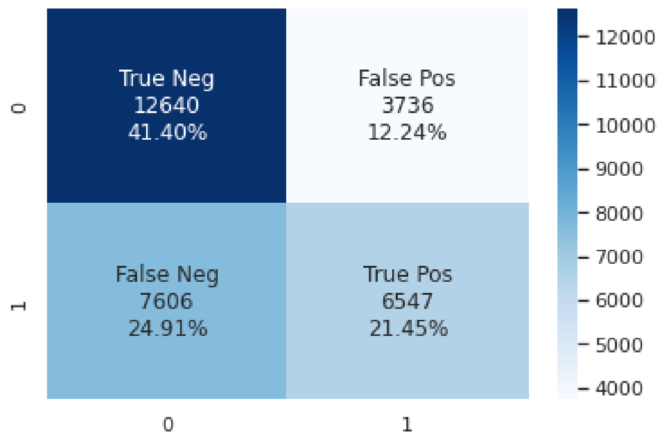


Fig. 22. confusion matrix of XGB

The best parameters after hyperparameter tuning with Grid-searchCV with cv=10 :

BEST PARAMS: {'learning_rate': 0.1, 'max_depth': 4, 'n_estimators': 100}

3) AdaBoost: AdaBoost or Adaptive Boosting is Boosting ensemble model. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights to incorrectly classified instances. Boosting is used to reduce bias as well as the variance for supervised learning.

It is a sequential model. Each of the subsequent learner is grown from previously grown learners, meaning weak learners are converted into strong ones. In practice, this boosting technique is used with simple classification

Figure 24 shows confusion matrix of Adaboost

The best parameters after hyperparameter tuning with Grid-searchCV with cv=10 :

BEST PARAMS: {'learning_rate': 1, 'n_estimators': 100}

Hyperparameter importance with optuna is shownFigure 23

4) Catboost: CatBoost is machine learning algorithm based on gradient boosting. This algorithm performs better than existing boosting algorithms like XGBoost, LightGBM etc in various aspects which includes training period, accuracy, computational power, tuning hyper parameters, implementation.

In CatBoost, the term 'Cat' refers to Category and 'Boost' refers to Boosting, this doesn't mean that CatBoost is restricted only for categorical features it also supports Numerical and Text features but it has an effective handling technique for categorical data.

The main difference between CatBoost and other boosting algorithms is that the CatBoost uses symmetric trees. And builds them level by level.

Symmetric tree is a tree where nodes of each level use the same split. This allows to encode path to leaf with an index which helps in decreasing prediction time, and it's extremely important for low latency environments.

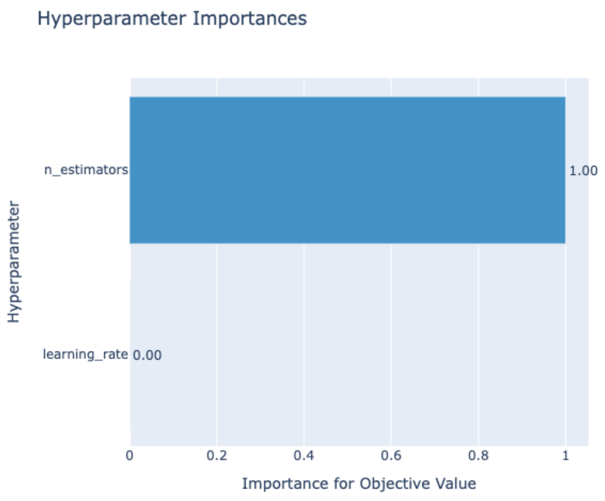


Fig. 23. Hyperparameter importance with optuna

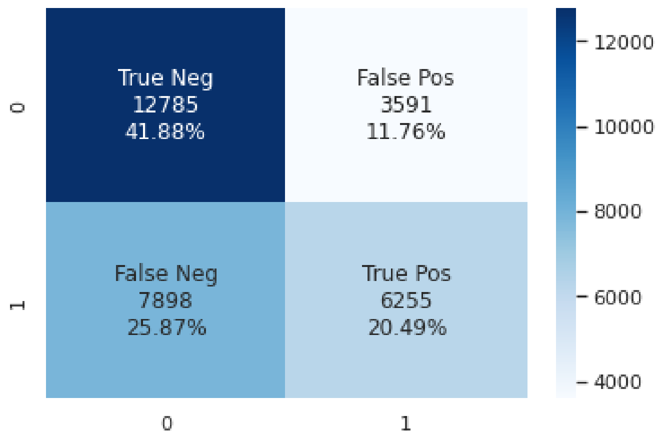


Fig. 24. confusion matrix of Adaboost

Importance of CatBoost:

Great quality without parameter tuning : Since the default parameters of CatBoost itself gives better result, there is no need of tuning hyper parameters, which in turn reduces the time spent for tuning.Categorical features support : CatBoost supports working with non numeric factors because of which time spent for pre-processing the data will be eliminated, and improves the training result too.Fast and scalable GPU version : Training the model on GPU gives better speedup compared to training the model on CPU. CatBoost efficiently supports multi-card configuration for large datasets [15] .Improved accuracy : CatBoost reduces over-fitting while constructing models with a novel gradient-boosting scheme.Fast prediction : CatBoost uses distributed GPUs, this feature enables Cat-Boost to learn faster and make predictions 13–16 times faster than other algorithms.

Figure 25 shows confusion matrix of Catboost.

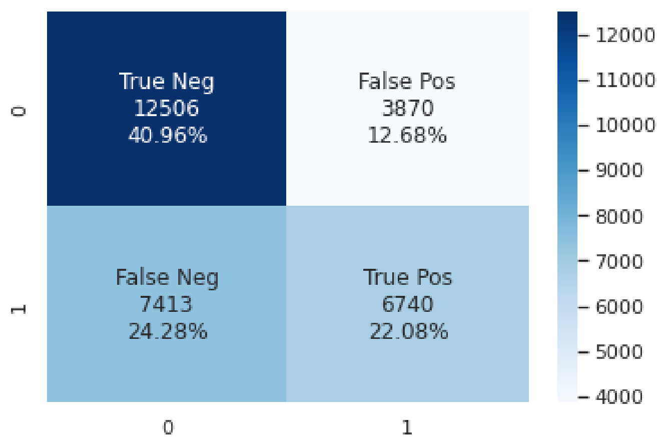


Fig. 25. Confusion matrix of Catboost

E. *Artificial Neural Network*

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions. Other advantages include: Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience. Self-Organisation: An ANN can create its own organization or representation of the information it receives during learning time.

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units. Input units:- The activity of the input units represents the raw information that is fed into the network. This is also called the input layer.Hidden units:- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units. This is also called a hidden layer. Output units:- The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units. This is also called the output layer.

The output Y from the neuron is computed as shown in the Figure above. The function f is non-linear and is called the Activation Function. The purpose of the activation function is to introduce non-linearity into the output of a neuron. This is important because most real world data is non linear and we want neurons to learn these non linear representations.

Every activation function (or non-linearity) takes a single number and performs a certain fixed mathematical operation on it. There are several activation functions you may encounter in practice:

Sigmoid: takes a real-valued input and squashes it to range between 0 and 1

σ(x) = 1 / (1 + exp(−x))

Softmax function: In classification tasks, we generally use a Softmax function as the Activation Function in the Output layer of the Multi Layer Perceptron to ensure that the outputs are probabilities and they add up to 1. The Softmax function takes a vector of arbitrary real-valued scores and squashes it to a vector of values between zero and one that sum to one. So, in this case,

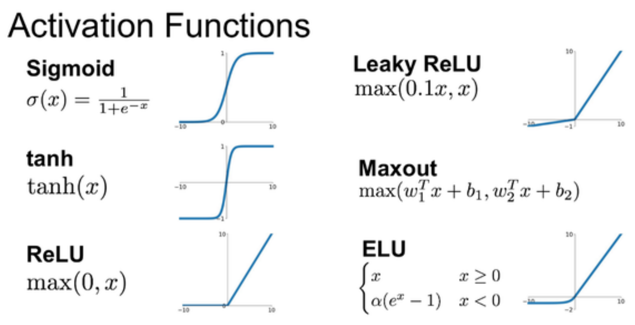
Probability (Pass) + Probability (Fail) = 1

tanh: takes a real-valued input and squashes it to the range [-1, 1]

ReLU: ReLU stands for Rectified Linear Unit. It takes a real-valued input and thresholds it at zero (replaces negative values with zero)

f(x) = max(0, x)

Figure 26 shows several other activation functions.



Activation Functions
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$
tanh
$\tanh(x)$
ReLU
$\max(0, x)$
Leaky ReLU
$\max(0.1x, x)$
Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$
ELU
$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$

Fig. 26. activation functions

Figure 27 shows the confusion matrix of ANN.

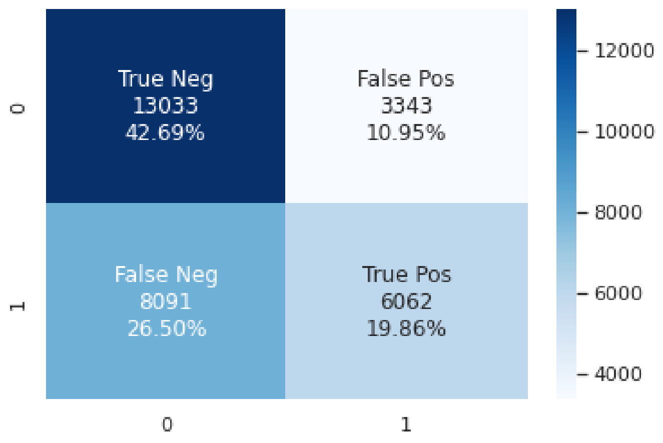


Fig. 27. Confusion matrix of ANN

Hyperparameter importance with optuna is shown Figure 28

VI. Evaluation Metrices

R-Squared calculated for Logistic Regression is not same as the R-Squared calculated for the Linear Regression Models. These Pseudo R-Squared helps measure the predictive power of the model [16] . There are many different ways to calculate the R-Squared value for Logistic Regression, However, No one approach is agreed to be the best. But, among all the type of R-Squared value McFadden's R-Squared is a better approach.

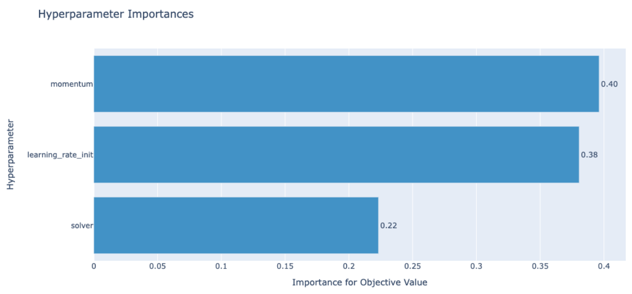


Fig. 28. Optuna for ANN

AIC(Akaike Information Criteria): AIC is the estimator of the goodness-of-fit of the model. Whenever we create a model we loose some information,no one can create the perfect model. AIC estimates the amount of information loss. Lesser the value of AIC means Lesser the information lost which means better the model. Adding variables to the model will not increase the value of AIC. One of the use of AIC is also that it helps in selecting the model.We can fit the whole data to train the model and compare the AIC values of different models and select the model with the best AIC Value.

Other Methods Include Confusion Matrix,Roc-Auc Curve.

ROC curve is one the important evaluating metrics that should be used to check the performance of an classification model. It is also called relative operating characteristic curve, because it is a comparison of two main characteristics (TPR and FPR). It is plotted between sensitivity (aka recall aka True Positive Rate) and False Positive Rate (FPR = 1-specificity).

AUC also called as AREA UNDER CURVE. It is used in classification analysis in order to determine which of the used models predicts the classes best. An example of its application are ROC curves. AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0 and if the predictions are 100% correct has an AUC of 1.

Characteristics of AUC:

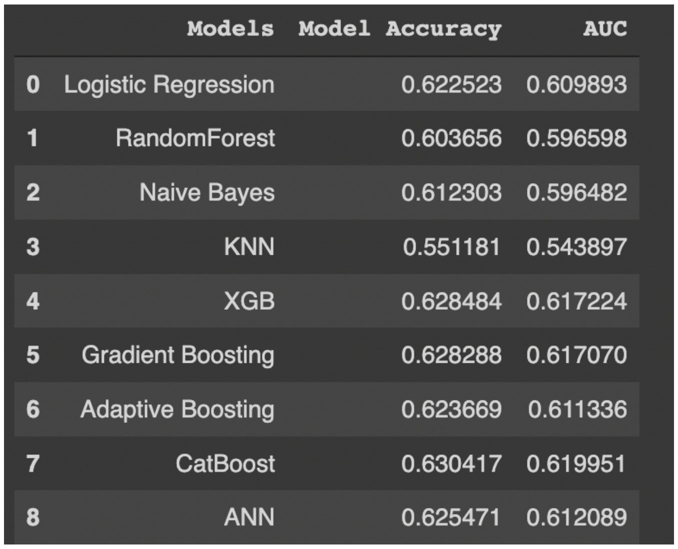
AUC is scale-invariant. It measures how well predictions are ranked, rather than their absolute values.AUC is classification-threshold-invariant. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

VII. Conclusion and Results

However, we must understand that these advanced algorithms are built to efficiently search for the best objective when the cost to iterate the model training process is too expensive.

If we are training with a small amount of data while using a model that is less complex, it is possible for GridSearch and RandomSearch to beat Optuna algorithm in terms of speed and performance.That being said, as data volume gets larger and models get more complex, the cost of randomly training a set of hyperparameters without any information increases tremendously, and these advanced algorithms greatly outperform traditional methods.

From Figure 29 we can conclude that Catboost gives the best performance with respect to model accuracy and AUC.



	Models	Model Accuracy	AUC
0	Logistic Regression	0.622523	0.609893
1	RandomForest	0.603656	0.596598
2	Naive Bayes	0.612303	0.596482
3	KNN	0.551181	0.543897
4	XGB	0.628484	0.617224
5	Gradient Boosting	0.628288	0.617070
6	Adaptive Boosting	0.623669	0.611336
7	CatBoost	0.630417	0.619951
8	ANN	0.625471	0.612089

Fig. 29. Evaluation comparision

Figure 30 shows the ROC:AUC curves for all the models.

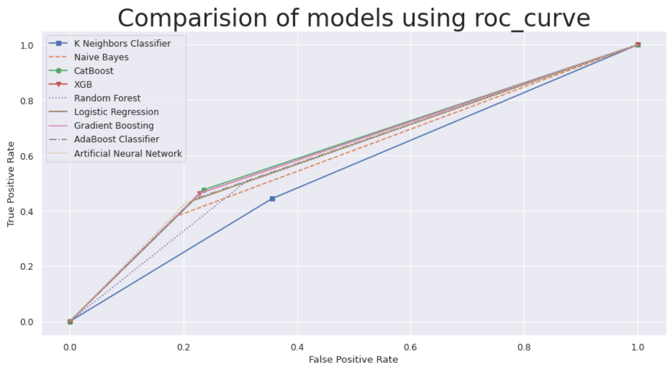


Fig. 30. ROC curve

VIII. Future Research

Deep learning can be an effective approach in predicting hospital re admissions among diabetic patients. A combination of Convolutional neural networks and advanced Deep Learning algorithms such as Bi-directional LSTM can outperform other machine learning algorithms when employed and evaluated against real life data.

References

[1] Y. Shang, K. Jiang, and L. Wang, "The 30-days hospital readmission risk in diabetic patients: predictive modeling with machine learning classifiers," *BMC Med Inform Decis Mak*, vol. 21, pp. 57–57, 2021.

[2] B. Strack, J. P. Deshazo, C. Gennings, J. L. Olmo, S. Ventura, K. J. Cios, and J. N. Clore, "Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records," *BioMed Research International*, vol. 2014, pp. 11–11, 2014.

[3] ——, "Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records," *BioMed Research International*, vol. 2014, pp. 11–11, 2014.

[4] K. M. Dungan, "The effect of diabetes on hospital readmissions," *J Diabetes Sci Technol*, vol. 6, no. 5, pp. 1045–52, 2012.

[5] E. Eby, C. Hardwick, M. Yu, S. Gelwicks, K. Deschamps, J. Xie, and T. George, "Predictors of 30 day hospital readmission in patients with type 2 diabetes: a retrospective, case-control, database study," *Curr Med Res Opin*, vol. 31, no. 1, pp. 107–121, 2014.

[6] M. T. Kassin, R. M. Owen, S. D. Perez, I. Leeds, J. C. Cox, K. Schnier, V. Sadiraj, and J. F. Sweeney, "Risk factors for 30-day hospital readmission among general surgery patients," *J Am Coll Surg*, vol. 215, no. 3, pp. 322–352, 2012.

[7] R. Dubey, J. Zhou, Y. Wang, P. M. Thompson, and J. Ye, "Analysis of sampling techniques for imbalanced data: an n=648 ADNI study," *NeuroImage*, vol. 87, pp. 220–261, 2014.

[8] A. Moreo, A. Esuli, and F. Sebastiani, "Distributional random oversampling for Imbalanced text classification," in *Proceedings of the 39th ACM conference on research and development in information retrieval (SIGIR 2016)*, 2016, pp. 805–808.

[9] X. S. Hu, R. J. Zhang, and Y. Zhong, "An unbalanced data classification algorithm based on clustering improvement," *Integr Technol*, vol. 2, pp. 35–41, 2014.

[10] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, 2019.

[11] K. Li, A. Jamieson, E. Rostamizadeh, M. Gonina, B. Hardt, A. Recht, and Talwalkar, 2018.

[12] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Proceedings of the 24th International Conference on Neural Information Processing Systems (NIPS'11*. Curran Associates Inc, 2011, pp. 2546–2554.

[13] S. Dreiseitl and L. Ohno-Machado, 2002.

[14] P. Jeatrakul, K. W. Wong, and C. C. Fung, 2010.

[15] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning*, 2015.

[16] A. Zheng, *Evaluating Machine Learning Models: A Beginner's Guide to Key Concepts and Pitfalls*, O'Reilly Media, 2015.