Diabetes Prediction Using Ensemble Techniques

Sarath Chandra (MT19037), Mani Kumar (MT19065), Nikhil (MT19123), Murali Krishna (MT19132)

Abstract

Diabetes is one of the most commonly known chronic diseases, leading to complications in health if it is unidentified and not diagnosed. The busy days make it tedious for a person to consult a doctor and undergo a test physically. In recent days, due to the increased impact of data and analysis of data on humans lead to the application of machine learning techniques in the field of medicine for predicting the various types of diseases. For the past two decades, there has been an exponential growth of medical data from digital devices, which has helped researchers study the impact of data analysis in medicine. Many previous studies discuss the implementation of machine learning techniques for predicting a person's diabetes considering significant features. In this paper, we present the details of our work, which includes the implementation of previous documents and some improvement techniques, reducing the research gap between Machine learning and medicine. We implemented various machine learning algorithms on the data collected from PIMA Indian Diabetes Database, which is sourced from the UCI Machine learning repository. We applied machine learning techniques such as K Nearest Neighbors, Logistic regression, Naive Bayes, Decision trees, Gaussian process, Linear SVM, RBF SVM, Xgboost, Gradient boost, AdaBoost and Random forest. All these mentioned algorithms are applied to the normalized data. The performance comparison of the model is discussed based on the accuracy as an evaluation metric, along with a brief description of how every model is implemented in this paper. The voting classifier is applied on top of the best models from the above machine learning techniques listed. This report also describes the section about individual contributions of every author and the details about the model we deployed, which finds it' easier to predict diabetes based on user input.

2. Introduction

As technology is rapidly increasing, there is a massive outburst in the availability of medical data, which leads to easy and secure access to resources for data scientists. We can classify the medical field data into different classes, which becomes a classification problem in machine learning. For instance, we have two types of diabetes, and Cancer has four stages, etc. Diabetes is a chronic disease where the glucose level rises to such an extent that pancreas fails to produce sufficient amounts of insulin to neutralize it, due to which metabolism becomes abnormal. Diabetes can be categorized into two types: Type-1 just accounts for 5-10% of diabetic cases, as cells cannot respond to insulin secretion. Type 2 represents 90-95% of cases, resulting in the pancreas failing to produce enough insulin due to the loss of beta cells. India is referred to as “Diabetes Capital of the world” as 74 million people were affected in 2017. If the disease is not identified in the early stages, then it might lead to various severe problems such as cardiovascular problems, kidney-related problems, blindness.

Varshith et al. in “Prediction of Diabetes using Ensemble Techniques” [1] have used the Pima Indian Diabetes dataset, which consists of 768 patients records. The machine learning models used in this paper are KNN, Logistic Regression, Decision Tree, Naive Bayes, Linear SVM, RBF SVM, Gaussian Process, Ada Boosting, Random Forest. Normalization is done on the initial extracted data to avoid scaling problems. In order to increase the performance of the model, cross-validation techniques and ensemble methods are used. Voting based ensemble method is where it combines best machine learning techniques into one optimal model.

The major drawback of the existing model is the preprocessing techniques used in the paper. The ratio between diabetic samples to non-diabetic samples is 268:500, which is biased on non-diabetic patients. On analyzing the data carefully, we have found out that some features had incorrect values. Zero was assigned to features such as BMI, blood pressure, glucose levels, insulin levels, skin-thickness which is not possible.

To overcome these limitations, we have used the imputation technique and up-sampling methods. The imputation technique is used to get rid of all the inappropriate values. The up-sampling approach helps us to increase the minority class samples, which ensures the model does not overfit. Along with the models in the existing paper, we have also implemented Xgboost and

Gradient Boosting classifiers, which performed better than the models implemented in the paper. Instead of applying voting classifier on all models, we have chosen the models which yielded high accuracy. Rest of the paper is organized as follows; Section 3 describes Material and Methods used, Section 4 draws the results obtained, Section 5 discusses the performance of the model, Section 6 and Section 7 tells about deployment and individual contributions respectively.

3. Material and Methods

3.1 Dataset Description

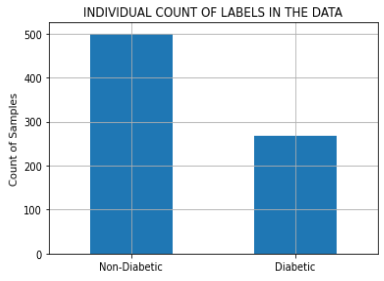
The dataset which we used for the project is taken from "PIMA Indian Diabetes Database" which is available publicly in the Kaggle site here. The data consists of 8 features such as age, diastolic blood pressure, insulin level, glucose level, pregnancies, diabetes pedigree, skin thickness, BMI (body mass index) and one class label which is either 0 or 1. Class label "0" indicates the human is non-diabetic, whereas "1" indicates human is diabetic.

3.2 Evaluation Parameters

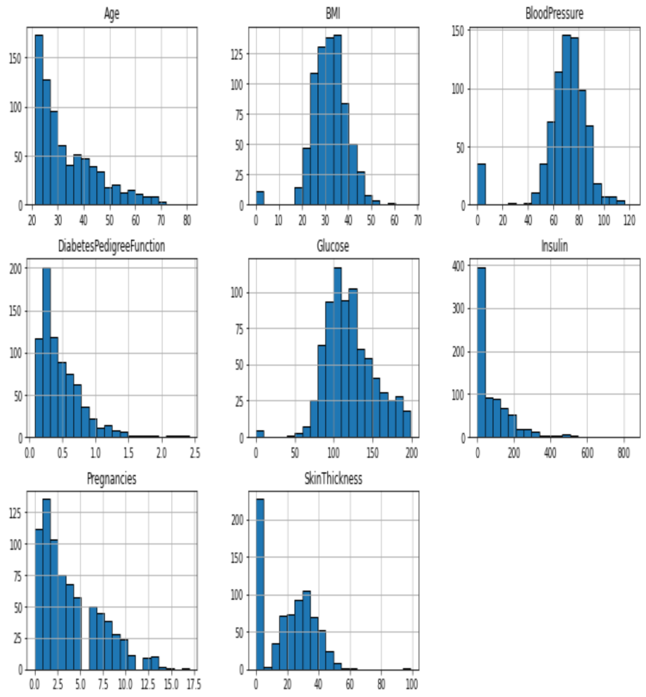
We have considered precision, recall, f1-score, specificity and accuracy as evaluation metrics/evaluation parameters. Precision is, count of true positives upon sum of true positives (TP) and false positives (FP). In this case it can be defined as, proportion of patients who are truly having diabetes are correctly classified as having diabetes. Recall is count of true positives upon sum of true positives and false negatives. It basically gives the true-positive rate of the model. F1-score is generally considered as harmonic mean of both precision and recall. Specificity in this case can be defined as proportion of patients who are not truly having diabetes and correctly classified as not having diabetes, which equals TN/(TN+FP).

3.3 EDA (Exploratory Data Analysis)

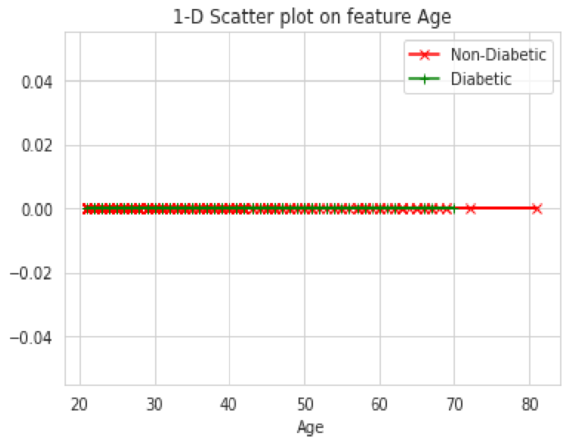
The count of samples present per class in the dataset can be visualized using bar plot. The plot has labels on the X-axis and count of samples on the Y-axis. In the original data non-diabetic had "0" class label whereas the diabetic had class label "1". Plot for the same can be seen here,

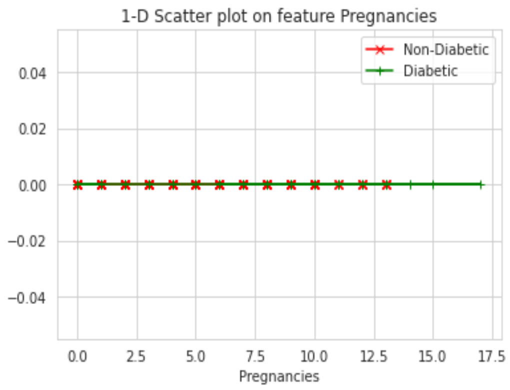


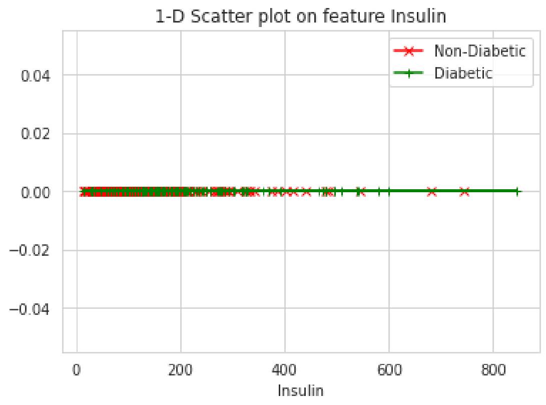
Count of non-diabetic samples is 500 whereas count of diabetic samples is 268. A histogram is also plotted for every feature to understand the range of values present in them and to detect anomaly values. The plot for the same can be seen below,

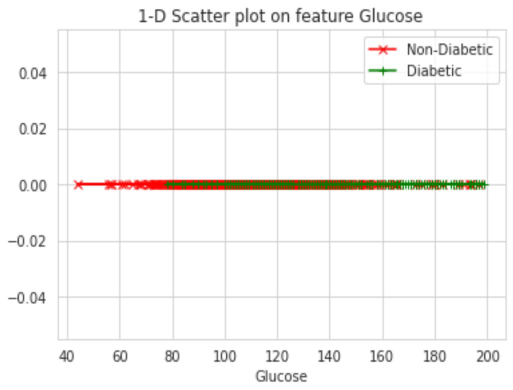


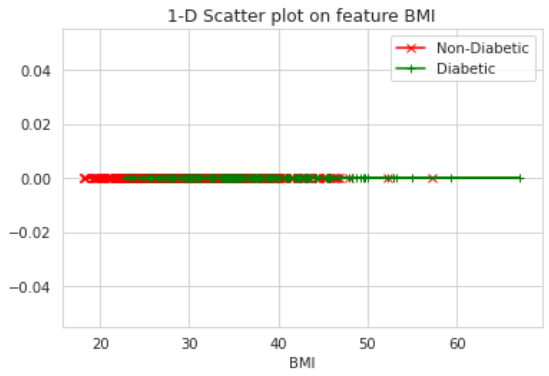
From the above histogram plot it is clear that some features in the dataset have some meaningless/incorrect values (spurious tuples) such as blood pressure, BMI, glucose, insulin, skin-thickness have zero values, which is not possible for any living human-beings. These incorrect tuples are handled by imputation technique (discussed in section 3.4). We performed univariate analysis (after performing imputation) with respect to every feature to understand their respective classification abilities. The plots with respect to every feature can be seen below,

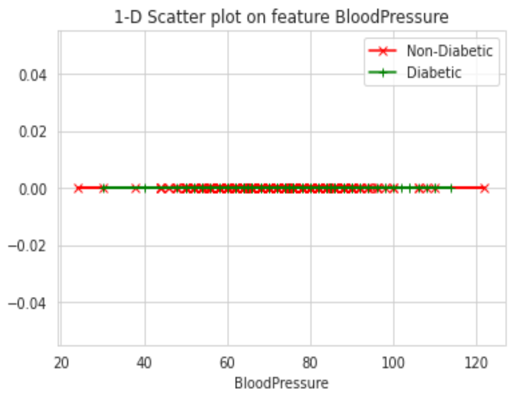


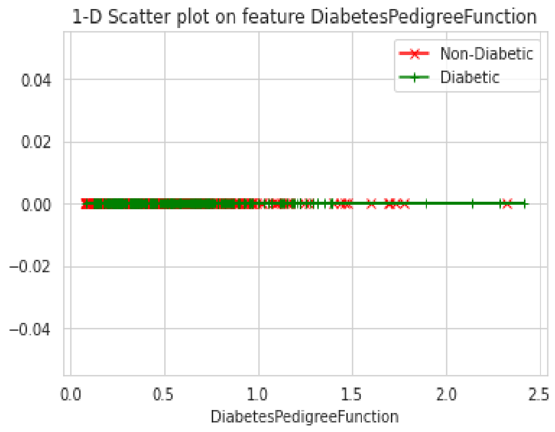


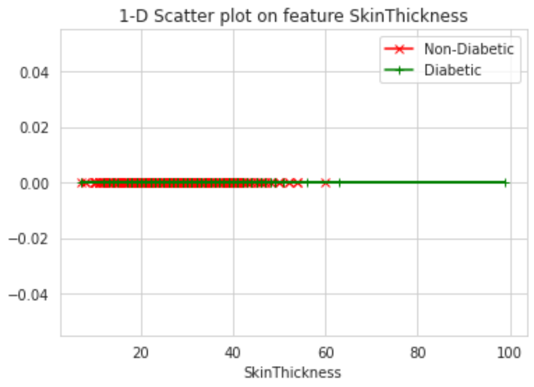




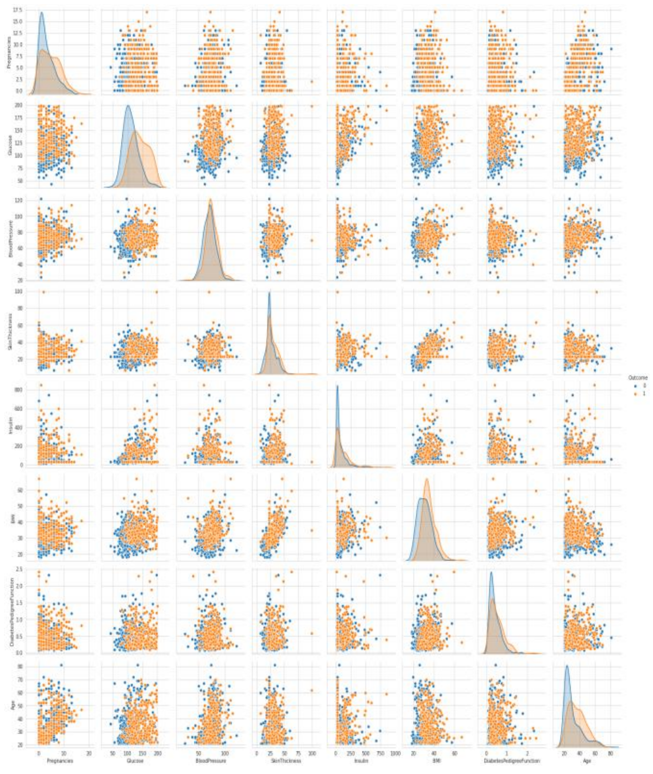








From the above plots of univariate analysis for every feature, we can clearly infer that none of single feature individually can classify the data at least in a decent manner. As a part of Exploratory Data Analysis (EDA), the final task we had done was visualizing a pair-plot. Pair plot gives us the scatter plots for every two features in the data and also clear information of distributions for every pair of features. The plot can be seen below,



We can clearly infer from the above plot that the combination of any two features cannot even decently classify the data as there is high amount of overlap between the PDFs of every two features.

3.4 Imputation Technique for handling incorrect values

As seen in the feature wise histogram plot, five out of eight features had zero values which is meaning-less according to the domain knowledge. In these scenarios, the technique of imputation comes into picture. In this technique, the meaningless zero values in the features are replaced by their respective feature medians. For example, the zero values in the feature blood pressure are replaced by the median of blood pressure feature.

3.5 Hyper-parameter tuning using cross-validation

For every machine learning algorithm, we implemented GridSearchCV technique with k-fold cross validation. The best hyper-parameters or the approximate range in which they lie can be obtained using this technique. Value of "k" we considered as 5 and 10. When we pass training data to grid-search it implicitly implements k-fold cross validation. In k-fold cross-validation we divide the entire training set into k parts. Out of which (k-1) parts of the data are used for training and remaining one part is used for testing. At every iteration different (k-1) parts of the data are considered for training and the remaining one part is considered for testing. The final performance will be mean of performances on test set at every iteration. The best hyper parameter is that the value for which maximum performance is obtained. Generally, the performance metric will be accuracy.

3.6 Up-Sampling of minority class samples

An up-sampling technique called ADSYN (adaptive synthetic sampling approach) is used for up-sampling the samples belonging to minority class. This technique is employed only for ensemble models because the ensemble models are generally prone to overfit. As the count of total samples in the data is already low, there is a high probability that model will overfit. Hence, we implemented the up-sampling approach for ensemble techniques. A new sample is generated as follows,

$$s_i = x_i + (x_{zi} - x_i)\lambda$$

Here, x_i is the sample belonging to minority class and x_{zi} is the nearest sample of same class. "λ" is a random floating-point number which has a range from 0 to 1.

3.7 Data standardization

To avoid scaling problems, for distance-based learning techniques such as KNN, Logistic Regression, SVMs we have used the "standardScaler()" library of "sklearn" in order to standardize the data.

3.8 Data split into train-test

We had split the data into train and test. The size of training data and test data are 70% and 30% of the total samples in the data (768 samples) respectively. The random state "0" is used for the split.

3.9 Machine Learning techniques implemented

3.9.1 KNN (K Nearest Neighbors)

The KNN algorithm considers the k nearest neighbors for a test vector, takes the majority vote of class labels of its neighbors and assigns it as label to the given test vector. We standardized the data using standardscaler. As, "k" is a hyper-parameter we found out the best "k" using GridSearchCV technique with 5-fold cross-validation considering accuracy as performance metric for best estimator. Then after we predicted the labels for test data using the same "k" value that gave optimal performance on cv splits.

3.9.2 Logistic Regression

The name has regression, but this technique is only used for two-class as well as multi-class classification purposes. The logistic regression draws a decision boundary that separates two classes (might be >2 based on class labels). The logistic regression assumes that points to one side of the decision boundary or hyperplane belong to one particular class and points on the other side of the hyperplane belong to another class. We standardized the data using standardscaler. The regularization coefficient "lambda" is the hyper-parameter in this technique. We used GridSearchCV with a 5-fold cross validation technique to determine the best value for lambda. The same lambda is used to predict labels on the test data.

3.9.3 Naïve Bayes

In Naïve Bayes, when a test vector is given it calculates the probability of that test vector belonging to each class. P(C|Test-vector) is maximized. The class which gives maximum P(C|Test- vector) is the class label of that test vector. The smoothing factor "alpha" is the hyper-parameter in Naïve Bayes. We standardized the data using standardscaler. We found out the best "alpha" using GridSearchCV technique with 5-fold cross validation on the basis of accuracy metric. Then after we predicted the labels for test data using the same "alpha" value**.**

3.9.4 Gaussian Process

Gaussian process classification is a non-parametric classification method which is mainly based on Bayesian methodology. The algorithm assumes some prior distribution on the underlying probability densities that guarantees some smoothness properties. We did not perform any hyper-parameter tuning in this technique. Gaussian process classification assumes that underlying data is normally distributed. Even though it is a strong assumption, GPC works amazingly well in most of the cases. In our case also, we achieved some good results on our data

3.9.5 Decision Trees

Decision tree is a tree-based classifier which is highly interpretable. Here each internal node represents a feature on which decision is made. Uses "Gini-impurity" (which works on the concept of entropy) technique in deciding what features to be considered as internal nodes. Entropy is defined as a measure of randomness in the data. The depth of the tree is the hyper-parameter here. We standardized the data using standardscaler. We found out the best "depth" using GridSearchCV technique with 10-fold cross-validation basis of accuracy metric. Then after we predicted the labels for test data using the same "depth" value

3.9.6 SVM (Linear, RBF)

SVM gives the hyperplane that best separates the data. It draws the margin maximizing hyperplane between two classes. The popular hyper parameters we have in SVM are kernel and "C" (regularization parameter). We have used both the kernels for our project namely "linear" and "RBF". Different kernels map the input data to the different feature spaces. We found out the best "C" using GridSearchCV technique with 10-fold cross validation on the basis of accuracy metric. Then after we predicted the labels for test data using the same "C" value.

3.9.7 Gradient Boosting Decision Trees (GBDT)

The base learners in boosting algorithms are models which have high-bias (high training error) and low variance combined with additivity. Decision stumps with maximum depth of 1 or 2 are the models which have high-bias and low variance. The base learner at a particular stage is trained on the x_i's and the errors that have been produced in the previous stage. So, as the number of base learners increases, we keep on trying to fit on the errors of the previous base learners thus reducing the training error. In gradient boosting, the negative value of gradient (derivative of loss function w.r.t x_i) at a point x_i is considered as error_i. The number of base learners to use is a hyper parameter in this technique. The training data considered has been up sampled before it was trained. The best value for hyper-parameter is found out using GridSearchCV with 5-fold cross validation technique. Thereafter we considered the range of estimators for which there is performance raise on the cv data and tried out all the values in that range to obtain best performance on the test data.

3.9.8 Xgboost

Xgboost is almost like an extension to GBDT. It implements GBDT with column sampling strategy. Row sampling is inbuilt in GBDT in "sklearn" implementation. The number of base learners to use is a hyper parameter in Xgboost. The training data considered has been up sampled before it was trained. The best value for hyper-parameter is found out using GridSearchCV with 5-fold cross validation technique. Thereafter we considered the range of estimators for which there is performance raise on the cv data and tried out all the values in that range to obtain best performance on the test data.

3.9.9 Adaboost (Adaptive Boosting)

At every stage, adaboost technique gives more weight to (can be done by up sampling) to the misclassified points. The weightage given to points exponentially increases from stage to stage. This creates a new dataset with a new plane of separation. Now, all these planes at every stage are applied on the dataset which eventually handles all the errors. The number of base learners to use is a hyper parameter in this technique. The training data considered has been up sampled before it was trained. The best value for hyper-parameter is found out using GridSearchCV with 5-fold cross validation technique. Thereafter we considered the range of estimators for which there is performance raise on the cv data and tried out all the values in that range to obtain best performance on the test data.

3.9.10 Random Forest

Random Forest is the most popular bagging technique used in current scenarios. This technique considers decision trees as base-learners with applying bagging on top and also uses column sampling strategy with aggregation. The trees considered here are of reasonable depth. The number of base learners to use is a hyper parameter in RFs. The training data considered has been up sampled before it was trained. The best value for hyper-parameter is found out using GridSearchCV with 5-fold cross validation technique. Thereafter we considered the range of estimators for which there is performance raise on the cv data and tried out all the values in that range to obtain best performance on the test data.

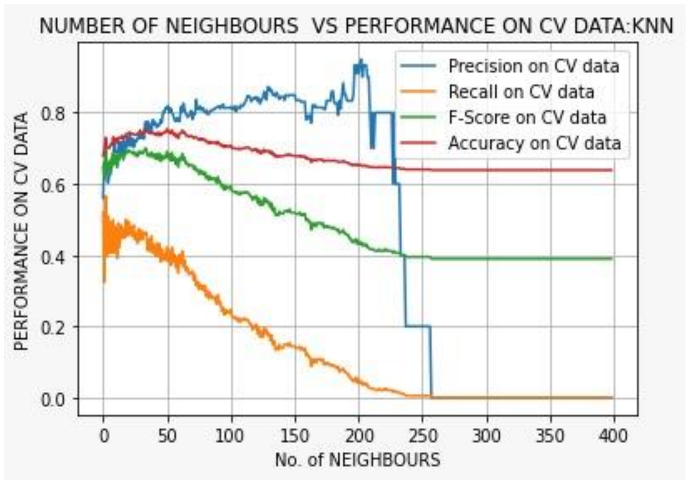
3.9.11 Voting Classifier

As a part of implementing a voting classifier, the label for a test sample is computed by considering the majority vote of all predictions which were predicted by top-3 performing models.

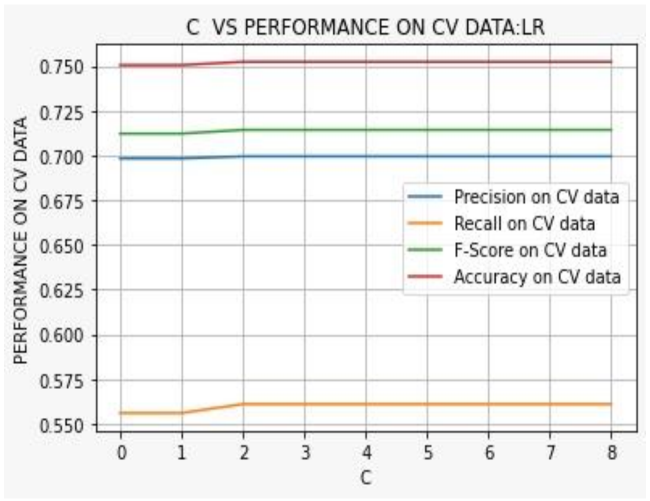
4. Results

We have used GridSearchCV technique with 5-fold cross validation and 10-fold cross-validation (for three techniques) in deciding the optimal hyper-parameters for a model. The plots are on CV data and table of results are on test data. Every plot in this section has 4 curves each of different color. As per the legend the blue color curve indicates how precision varies, yellow color curve indicates how recall varies, green color curve indicates how F-score varies, red color curve indicates how accuracy varies w.r.t respective hyper-parameters of the models. All plots are plotted based on performance of the model on CV data which can be seen below, here (CV) implies on “CV” data.

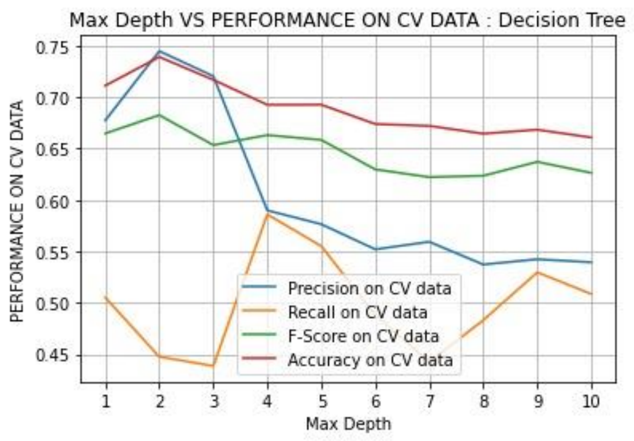
4.1 K vs Performance Metrics (CV) – KNN



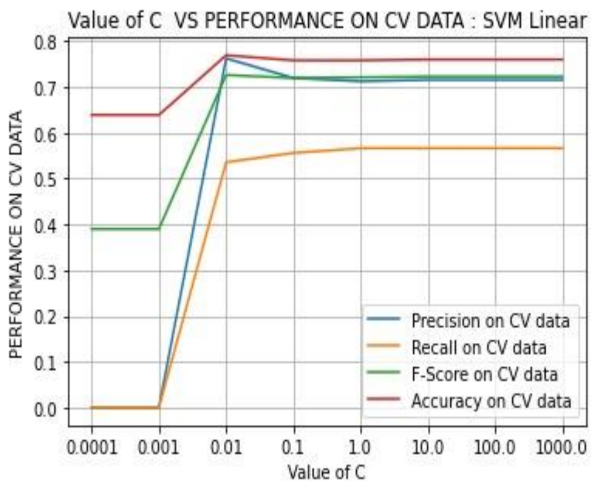
4.2 C vs Performance metrics (CV)-Logistic Regression



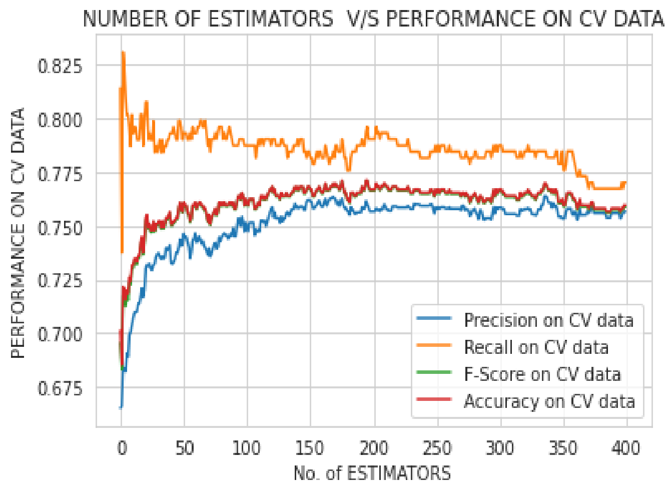
4.3 Depth vs Performance metrics (CV)-Decision Trees



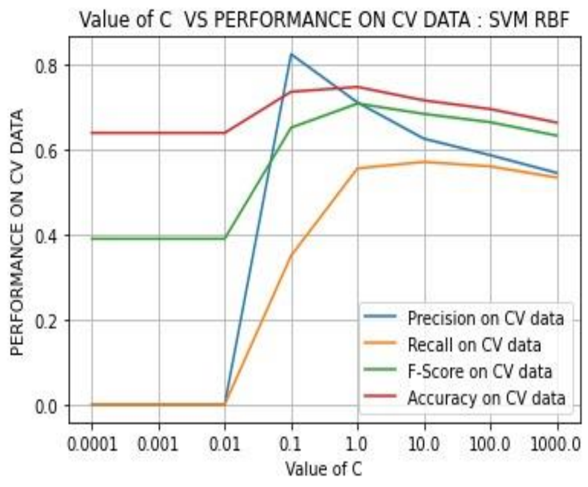
4.4 C v/s Performance metrics (CV) – Linear SVM



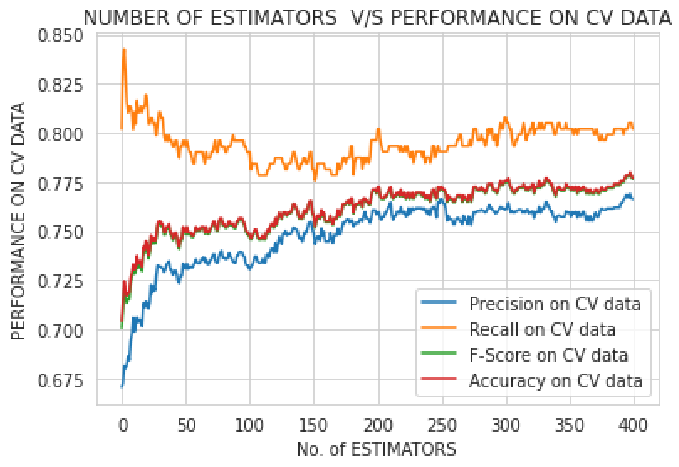
4.7 Estimator count vs Performance metrics - GBDT



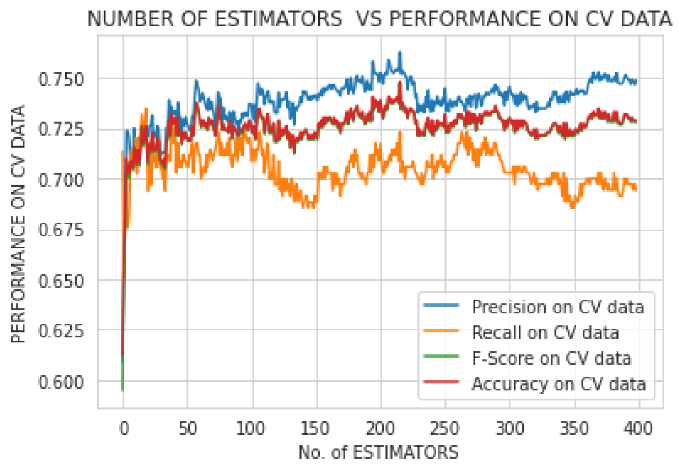
4.5 C v/s Performance metrics (CV) – RBF SVM



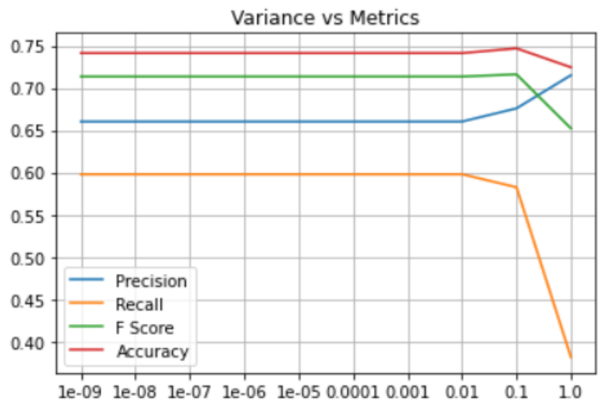
4.8 Estimator count vs Performance metrics – Xgboost



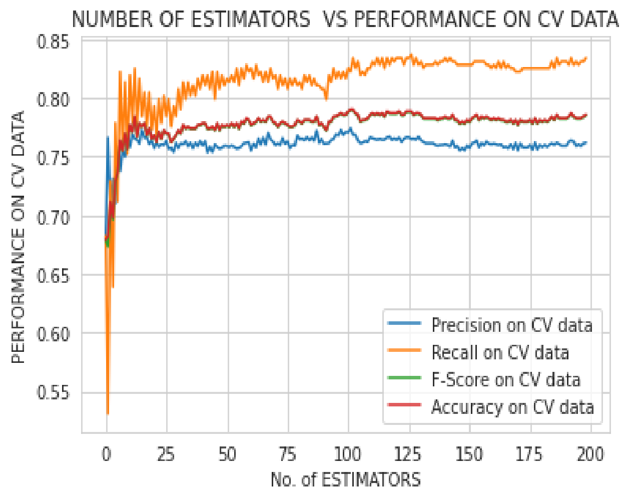
4.9 Estimator count vs Performance metrics-Adaboost



4.6 Alpha vs Performance metrics – Naïve Bayes



4.10 Estimator count vs Performance metrics – RFs



4.11 Results achieved on Test Data

S. No	Model Used	Accuracies in Published Paper	Accuracies Achieved by us
1.	KNN	74.89	75.75
2.	Logistic Regression	75.75	77.92
3.	Naive Bayes	73.59	76.623
4.	Decision Trees	71.86	72.72
5.	Gaussian Process	74.02	77.22
6.	SVM Linear	70.99	78.7878
7.	SVM RBF	75.32	75.7575
8.	Xgboost	Not Applied in Paper	78.787
9.	Gradient Boosting	Not Applied in Paper	79.220
10.	Adaboost	73.16	79.653
11.	Random Forest	75.75	80.086
12.	Voting Classifier	80.95	83.18

5. Discussion

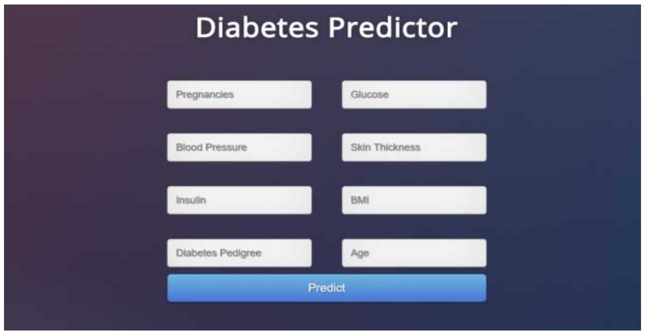
For hyper-parameter tuning we have used GridSearchCV with 5-fold cross validation and 10-fold cross-validation (for 3 techniques). The best estimator is decided on the basis of accuracy metric.

For K Nearest Neighbors and logistic regression, the optimal values of hyper-parameters “k” and “C” are found to be 34 and 3 respectively. In case of gaussian process we have taken default kernel i.e. RBF and optimized it during the training in each iteration. For decision trees, Linear SVM, RBF SVM GridSearchCV technique with 10-fold cross-validation is implemented and the optimal hyper-parameters obtained are 2 (max-depth), 0.01 (C), 0.1 (C) respectively. For gaussian naive bayes, we are able to achieve best results when variance is 0.1. In the case of random forest, gradient boosting, adaboost, xgboost classifiers we obtained optimal number of base estimators to be 181, 140, 74, 62 upon applying GridSearchCV technique with 5-fold cross validation and slightly even more fine-tuning the model.

The reason for random forest giving the optimal performance on test data compared to other models might be because, random forests combines (here combines means considering outputs) many base learners which are having low bias and high variance and performs aggregation on them (takes majority vote of outputs). Thus, reducing the final variance of the model. Now, this random forest model has least amount of bias and least amount of variance which therefore enabling the model to produce the best results. If the size of the data becomes surplus as a part of future work, we can implement deep-learning based models which introduce non-linearity thus improving the overall performance.

6. Deployment

We have deployed all our models using flask-web framework in local host environment. The final prediction will be the majority vote on predicted labels of top-3 performed classifiers.



7. Contributions

Every author has put up lots of efforts in designing the models to work with great performance. The contributions made by every author are listed below,

Author	Contribution
Sarath Chandra (MT19037)	• Naïve Bayes • Gaussian process
Mani Kumar (MT19065)	• Random Forests • Gradient boosting • Xgboost • Adaboost • Deployment
Kolla Nikhil (MT19123)	• KNN(K-nearest-neighbors) • Logistic Regression
Murali Krishna (MT19132)	• Decision Trees • Linear SVM • RBF SVM

8. References

[1] Sandhiya, K. DESIGN AND DEVELOPMENT OF SUPERVISED LEARNING ALGORITHM FOR DIABETES DIAGNOSIS.

[2] Sisodia, D., & Sisodia, D. S. (2018). Prediction of diabetes using classification algorithms. *Procedia computer science*, *132*, 1578-1585.

[3] Lai, H., Huang, H., Keshavjee, K., Guergachi, A., & Gao, X. (2019). Predictive models for diabetes mellitus using machine learning techniques. *BMC Endocrine Disorders*, *19*(1), 1-9.

[4] Shailaja, K., Seetharamulu, B., & Jabbar, M. A. (2018, March). Machine Learning in Healthcare: A Review. In *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)* (pp. 910-914). IEEE.

[5] Sarwar, M. A., Kamal, N., Hamid, W., & Shah, M. A. (2018, September). Prediction of Diabetes Using Machine Learning Algorithms in Healthcare. In *2018 24th International Conference on Automation and Computing (ICAC)* (pp. 1-6). IEEE.

[6] "IDF DIABETES ATLAS - 8TH EDITION," International Diabetes Federation, 2017. [Online]. Available: https://diabetesatlas.org/. [Accessed: 15-Dec2018].

[7] GLOBAL REPORT ON DIABETES WHO Library Cataloguing-in-Publication Data Global report on diabetes. 2016.

[8] "PIMA Indian Diabetes Dataset, An open dataset," UCI Machine Learning Repository. [Online]. Available: http://ftp.ics.uci.edu/pub/machine-learningdatabases/pima-indians-diabetes/. [Accessed: 11-Jan2019]

[9] R. Bansal, S. Kumar, and A. Mahajan, "Diagnosis of diabetes mellitus using PSO and KNN classifier," in 2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN), 2017, pp. 32–38

[10] L. Li, "Diagnosis of Diabetes Using a Weight-Adjusted Voting Approach," in 2014 IEEE International Conference on Bioinformatics and Bioengineering, 2014, pp. 320–324

[11] Prema, N S and Pushpalatha, M P. "Prediction of gestational diabetes mellitus (GDM) using classification", 2017 IEEE International Conference on science, technology, engineering and management (icstem'17), Coimbatore, 2017

[12] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, 2825-2830.

[13] T. Mitchell, "Machine Learning", McGraw Hill. P. 2, 1997.

[14] Kumari, V.A. and R. Chitra, "Classification of Diabetes Disease Using Support Vector Machine", International Journal of Engineering Research and Applications, vol.3, pp. 1797-1801, 2013.

[15] Aishwarya, R., Gayathri, P., Jaisankar, N., 2013. A Method for Classification Using Machine Learning Technique for Diabetes. International Journal of Engineering and Technology (IJET) 5, 2903–2908

[16] Aljumah, A.A., Ahamad, M.G., Siddiqui, M.K., 2013. Application of data mining: Diabetes health care in young and old patients. Journal of King Saud University - Computer and Information Sciences 25, 127–136. doi:10.1016/j.jksuci.2012.10.003.

[17] Choubey, D.K., Paul, S., Kumar, S., Kumar, S., 2017. Classification of Pima indian diabetes dataset using naive bayes with genetic algorithm as an attribute selection, in: Communication and Computing Systems: Proceedings of the International Conference on Communication and Computing System (ICCCS 2016), pp. 451–455.

[18] Han, J., Rodriguez, J.C., Beheshti, M., 2008. Discovering decision tree based diabetes prediction model, in: International Conference on Advanced Software Engineering and Its Applications, Springer. pp. 99–109.

[19] Iyer, A., S, J., Sumbaly, R., 2015. Diagnosis of Diabetes Using Classification Mining Techniques. International Journal of Data Mining & Knowledge Management Process 5, 1–14. doi:10.5121/ijdkp.2015.5101, arXiv:1502.03774.

[20] Balkau B, Lange C, Fezeu L, et al. Predicting diabetes: clinical, biological, and genetic approaches: data from the epidemiological study on the insulin resistance syndrome (DESIR). Diabetes Care. 2008;31:2056–61.

[21] Griffin SJ, Little PS, Hales CN, Kinmonth AL, Wareham NJ. Diabetes risk score: towards earlier detection of type 2 diabetes in general practice. Diabetes Metab Res Rev. 2000;16:164–71.

[22] Kandhasamy JP, Balamurali S. Performance analysis of classifier models to predict diabetes mellitus. Procedia Comput Sci. 2015;47:45–51.

[23] Lindström J, Tuomilehto J. The diabetes risk score: a practical tool to predict type 2 diabetes risk. Diabetes Care. 2003;26:725–31.

[24] Habibi S, Ahmadi M, Alizadeh S. Type 2 diabetes mellitus screening and risk factors using decision tree: results of data mining. Global J Health Sci. 2015; 7(5):304–10.

[25] Ioannis K, Olga T, Athanasios S, Nicos M, et al. Machine learning and data mining methods in diabetes research. Comput Struct Biotechnol J. 2017;15: 104–16.

[26] Kahn HS, Cheng YJ, Thompson TJ, Imperatore G, Gregg EW. Two riskscoring systems for predicting incident diabetes mellitus in U.S. adults age 45 to 64 years. Ann Intern Med. 2009;150:741–51.