# Diabetes Prediction

Samvrit Srinath, Divyam Sengar, Hrithik Pai, Harsh Gurnani

## Introduction

Diabetes is a metabolic disease in which the body doesn't produce sufficient amounts of insulin and the bloodstream retains excess sugar. Typically, the cells in the pancreas produce insulin in response to a stimulus signifying rising blood sugar. Insulin then triggers liver and body cells to take up glucose from the bloodstream. However, in diabetic patients, the pancreas produces little to no insulin, and any insulin that is made may not trigger a reaction in body cells, culminating in an inability to absorb blood glucose. In the long run, the disorder can lead to various threatening conditions, such as kidney disease, heart disease, and more. [2]

Diabetes is extremely prevalent worldwide, and is predicted to affect more and more people over the next few decades (many members of this team have family members affected by the disease). Even so, people often lack knowledge about the disease and its symptoms, leading to a significant population that goes undiagnosed.

**In this assignment, we strive to apply various machine-learning based classification techniques on patient data to predict whether a patient may be diabetic or not**. To build the models, we first analyzed the dataset and in some cases modified how much of the data we trained our models on. We developed four models for classification - Logistic Regression, XG Boost, Soft-Margin Support Vector Machine (SVM), and Random Forest Classifier - and compared the performance on each one.

The dataset we use for these calculations is a "Diabetes Prediction Dataset"[6], with medical and demographic data regarding patients. The features for each patient in the data-set are: age, gender (a binary flag which this dataset treats as biological sex), BMI, hypertension (a binary flag), heart disease (a binary flag), smoking history (qualitatively defined), HbA1c level, blood-glucose level, and diabetes status (a binary flag).

There are 100,000 entries in the dataset: 91,500 without diabetes and 8,500 diabetic. Clearly, there is quite the skew regarding the portion of diabetic patients in the dataset, which we account for as we develop our predictive approach.

## Exploratory Data Analysis

Attached are some tables that reflect the distribution of each feature relative to the binary flag of *whether the individual had diabetes.* Most of the distributions **do not reflect a conventional Gaussian Distribution**. Rather, there are definitive separations (Class 0 is often located to the left of the distribution, Class 1 is often located to the right), with a clear overlapping region in between. An example is provided below:

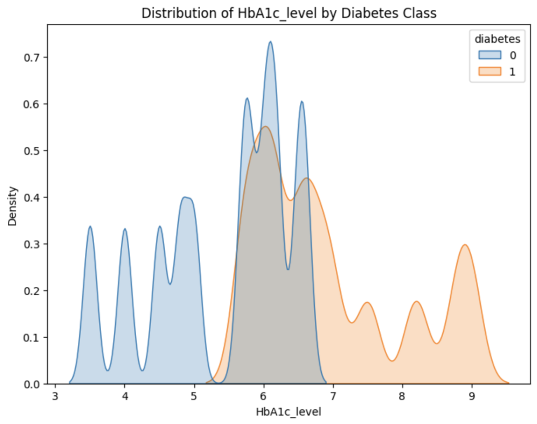


Figure 1: HbA1c levels vs Diabetes

### Tables

Table 1: Statistics for age

| | Diabetes = 0 | Diabetes = 1 |
|---|---|---|
| **Mean** | 40.115187 | 60.946588 |
| **std** | 22.306005 | 14.549880 |

Table 2: Statistics for bmi

| | **Diabetes = 0** | **Diabetes = 1** |
|---|---|---|
| **Mean** | 26.887163 | 31.988382 |
| **std** | 6.373406 | 7.558371 |

Table 3: Statistics for HbA1c_level

| | **Diabetes = 0** | **Diabetes = 1** |
|---|---|---|
| **Mean** | 5.396761 | 6.934953 |
| **std** | 0.971628 | 1.076562 |

Table 4: Statistics for blood_glucose_level

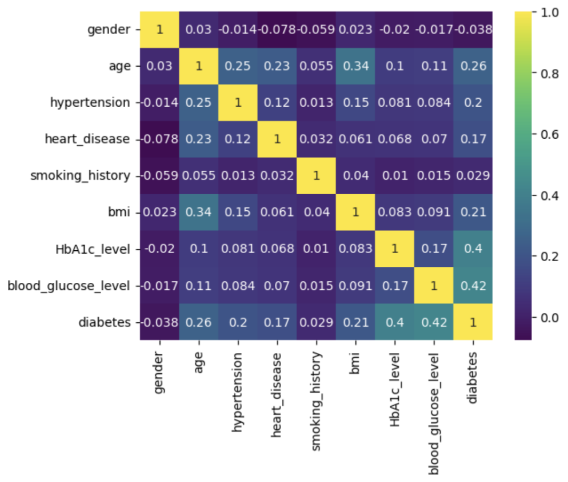
| | **Diabetes = 0** | **Diabetes = 1** |
|---|---|---|
| **Mean** | 132.852470 | 194.094706 |
| **std** | 34.247281 | 58.641111 |

Note that each variable contains 100000 entries, aforementioned with a split of 91500 negative examples and 8500 positive examples. Due to the presence of a clear, roughly linear decision boundary (at least when visualized in $\mathbb{R}^2$ space), we use Logistic Regression and Soft-Margin SVM, as well as decision tree algorithms like XGBoost and Random Forest. It's possible that we can use binary decisions as a result to converge for a given output layer - this property motivated us to explore Decision Tree formats.

One other interesting aspect is the lack of correlation between different variables in our dataset. Refer to Figure 2 below, which displays the correlation between different variables present in our dataset. Note that trivially, the correlation will be 1 along the diagonal.

As our Heatmap corroborates there exists little correlation between a variety feature-feature pairs. However, the most notable ones are (Indicated by Brighter, Green Colors):

- Diabetes and HbA1c Levels
- Diabetes and Blood Glucose Levels
- Age and Hypertension
- Age and BMI



| | gender | age | hypertension | heart_disease | smoking_history | bmi | HbA1c_level | blood_glucose_level | diabetes |
|---|---|---|---|---|---|---|---|---|---|
| gender | 1 | 0.03 | -0.014 | -0.078 | -0.059 | 0.023 | -0.02 | -0.017 | -0.038 |
| age | 0.03 | 1 | 0.25 | 0.23 | 0.055 | 0.34 | 0.1 | 0.11 | 0.26 |
| hypertension | -0.014 | 0.25 | 1 | 0.12 | 0.013 | 0.15 | 0.081 | 0.084 | 0.2 |
| heart_disease | -0.078 | 0.23 | 0.12 | 1 | 0.032 | 0.061 | 0.068 | 0.07 | 0.17 |
| smoking_history | -0.059 | 0.055 | 0.013 | 0.032 | 1 | 0.04 | 0.01 | 0.015 | 0.029 |
| bmi | 0.023 | 0.34 | 0.15 | 0.061 | 0.04 | 1 | 0.083 | 0.091 | 0.21 |
| HbA1c_level | -0.02 | 0.1 | 0.081 | 0.068 | 0.01 | 0.083 | 1 | 0.17 | 0.4 |
| blood_glucose_level | -0.017 | 0.11 | 0.084 | 0.07 | 0.015 | 0.091 | 0.17 | 1 | 0.42 |
| diabetes | -0.038 | 0.26 | 0.2 | 0.17 | 0.029 | 0.21 | 0.4 | 0.42 | 1 |

Figure 2: Heatmap of Variable Correlation

However, other pairs can be considered as roughly independent, which improves our ability to properly form a decision boundary between our two output classes. Similar to our initial assumptions, Diabetes, Blood Sugar, and HbA1c[8] are correlated with one another.

## Predictive Task

As mentioned, the task we explore is how machine-learning classification methods can predict a patient's diabetes status based on various physiological data. We used the logistic regression (LR) model as our baseline, and evaluated the XG Boost, Random Forest (RF), and SVM models in comparison. We aim for our chosen model to perform better than the LR model.

The predictive features in the dataset are described above, and pertain to physiological characteristics of the patient. We encode all binary flags (gender, hypertension, heart disease), with either 0 or 1; for quantitative features we simply used the value. Furthermore, smoking history was qualitatively defined, so we assigned each category a specific value. The categories and value are shown in Table 5.

In this scenario, it is important to consider the actual use-case of the model. In context, a false negative is when the model predicts that the patient is not diabetic even though they are and a false positive is a prediction that the patient is diabetic when they are not. For our task, predicting a false negative is significantly worse than predicting a false positive. We prefer to inform a patient that they may have diabetes and conduct further medical tests to determine the status of the disorder then fail to inform the patient of diabetes and leave their disease uncaught.

We first used basic accuracy (percentage of correct predictions) and Balanced Error Rate (BER) to evaluate the models. However, to account for the magnified impact of false negatives, we developed a Custom Error Rate (CER), in which we replicated BER except that we increased the weight for false negatives by a factor of 5, in order to penalize the model more for a false negative. Given that our CER assesses error across all predictions and gives special (increased) weight to the false negatives, we use the CER as our dominant error statistic for our model. Thus, we will evaluate performance of our model by how well it minimizes the CER. Furthermore, we assessed final model validity by comparing the predictions with observed data, and determining if the predictive weights (meaning the extent to which features influences the prediction) of each feature corresponded to the biological causes of diabetes.

In order to make these predictions, we used features from the data in our model. The features that we used are binarized gender (1 for male and 0 for female), age in years (as an integer), binarized hypertension (1 for presence and 0 for absence), binarized heart disease (1 for presence and 0 for absence), smoking history (see Table 5), bmi (as a float), HbA1c level (as a float), and blood glucose level (as a float). For the majority of these features, we simply pull their values from the dataset and incorporate them into the model. However, we had to process the data for a few of these features. For the gender feature, values were displayed as either 'Male' or 'Female', so we converted values of 'Male' to 1 and values of 'Female' to 0. Next, the smoking history feature had 5 different values, which we converted to features as shown in Table 5. We chose this order based on expected correlations between smoking and diabetes (i.e. that smoking is positively correlated with presence of diabetes). The rest of the features were numbers that we could simply pull from the dataset, so the remaining data processing was fairly simple.

Table 5: Smoking Values

| | |
|---|---|
| **Never** | 0 |
| **No Info** | 0.5 |
| **Former** | 1 |
| **Not Current** | 1.25 |
| **Current** | 2.5 |

## Models/Analysis

### Logistic Regression

As a baseline model, we wanted to see how directly using Logistic Regression to predict the presence of diabetes would fare on our dataset. Given that this is something we had explored in class and was relatively simple compared to our other approaches, we wanted to see how a simplistic model would perform. To do this, we first decided to split the data into 50% training, 20% validation, and 30% test portions. We used all the features above to fit the model on the training set and used the validation set to tune the hyper parameter C. After evaluating the mode on test set, we got accuracy = 0.886, BER = 0.114, and CER = 0.253. Next, we tried to normalize all the features and see how the model performs under these normalized features. We normalized all the features except gender as it is binary. Based on this, we made our new training set, tuned the hyper parameter C on the validation set, and finally made predictions on the test set, resulting in an accuracy = 0.888, BER = 0.111, and CER = 0.247. We tried different combinations of normalizations (dividing by the maximums of the features instead of their averages, only normalizing some features instead of others, etc.), but the best performance resulted from this average based combination. Thus, we get our baseline performance for the Diabetes Prediction, with an accuracy of 88.8%, BER of 0.111, and CER of 0.247 using LR, which gives a simple approach.

### XG Boost

For context, an XG Boost model is a Boosted Supervised Learning Approach that uses *Gradient Boosted* Training to increase the performance of the model. It uses a both

Regression and Classification ensembled with a Decision Tree architecture to ultimately make predictions. Similar to other model architectures, our current implementation used Logistic Loss, a general form following: $L(\theta) = \sum_i \left[y_i(\ln 1 + e^{-\hat{y}_i} + (1 - y_i)(\ln(1 + e^{-\hat{y}_i})\right]$. Note that all features mentioned above in previous sections are being used, although some features were weighted much less than others(see Figure 4).

Not only does an XGBoost try to minimize the logistic loss, the decision making process can be broken down into the following form:

1. Ensembling/Composing Regression and Classification Trees
2. We first classify a set of input vectors based on a family(i.e., the probability that a person with Blood Glucose Level $> 150$ is Diabetic or not) and assign a **score** to this family of vectors(corresponding to a singular leaf)
3. We utilize the score of the decision leaves in conjunction with other decision tree leaves to come to a joint decision to map to a given output.

Note that this is simply a higher level approach to Decision Tree, and the nuances of Boosting (i.e., additive training and Structure Regularization) are outside the scope of the predictive task. That being said, with a more complex optimizer (as well as novel ways of penalizing the model) our decision tree approach not only takes little time for each iteration of Gradient Descent (although it takes 10000 iterations to converge), but yielded very high performance on our model. The hyperparemters tuned in order to maximize performance were:

1. Number of Estimators or Trees;
2. Maximum Depth
3. Minimum Child Weight

Note that in our current approach, we opted to use a Validation Set consisting of 20% of the utilized data, and this was consistently run on each epoch trained on the model. We defined an **early stopping round**, where after 10 iterations, if the objective had not changed

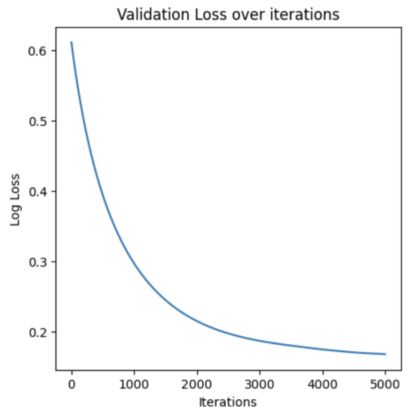


Figure 3: Validation Loss over Iteration

by a significant amount, we stopped training. Thus, mentioned in our results later, we did not encounter overfitting due to the aforementioned steps.

Here are the Feature Weights:

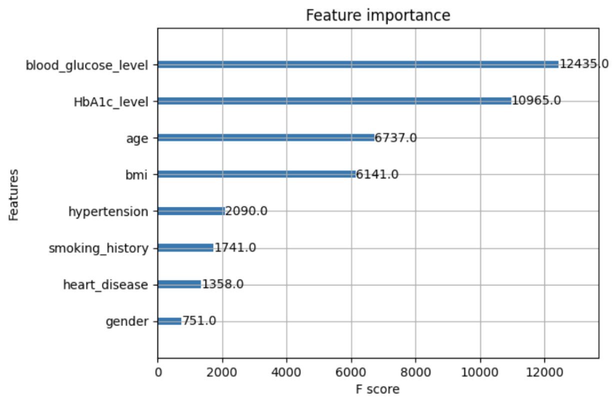


Figure 4: Feature Weights for XGBoost

As such, upon testing, blood_glucose_level and HbA1c Levels proved to have the most weight on predictions to our output. After tuning hyper-parameters using GridSearchCV, we found that accuracy was maximized when **n_estimators= 5000, max_depth = 7, min_child_weight = 5**.

## Support Vector Machine

Support Vector Machine is a supervised learning algorithm that is used for classification. In our problem, our support vector machine classified a certain datapoint to have diabetes (1) or not have diabetes (0). A support vector machine works by plotting all data points on an n-dimension hyperplane, where n is the number of features. Based on the type of kernel given to the model (linear, polynomial, sigmoid, rbf), the model is able to construct a division of the points and classify the value of unseen data. A Support Vector Machine is beneficial because it is easy to change to adapt different datasets. Also, Python offers a convenient svm.SVC package within SciKit. This makes it easy to train data and tune parameters as necessary.

For our model, we decided on using the features mentioned above to predict whether or not the person has diabetes. Since the dataset had significantly more negative data than positive data, we decided to use a 2.5:1 negative to positive ratio, which ensures that the model is not biased to predicting negative data. Doing this ensures that there is enough data to train the model, while limiting the runtime of the model because a soft-margin model is computationally expensive. Having more negative data can also skew the model - the more negative values the model sees, the more likely to predict negative, even if the other features do not correlate.

To train our model and tune hyper-parameters, we found that a cross-validation strategy is best, instead of splitting the data into train, validation, and test. This allows the model to 'view' all the data, while still keeping some data aside for tuning parameters. Some of the parameters we tuned were C, gamma, and the kernel. The kernel type, as mentioned above, refers to the type of boundary we want to find between groups. The gamma parameter specifies the influence a certain data point has on finding the boundary. A high gamma means that the boundary is less flexible, which leads to a smoother boundary, while a lower gamma value allows the model to capture more variations in the data. The C parameter is the regularization parameter which represents the error region of a model. A higher C value gives the model a smaller margin, but may lead to overfitting, wheras a smaller parameter may classify points incorrectly. As a result of our tuning, we found that the best parameters for our soft margin model model were: {'C': 1, 'gamma': 1, 'kernel': 'rbf'}.

Since our model has 6 features, we unfortunately cannot plot it on a graph as it uses 6 dimensions, but we can see how the model did on certain factors.

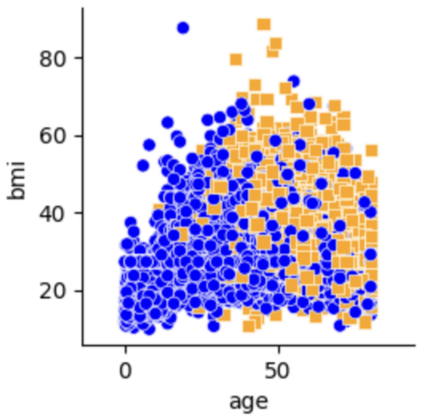


Figure 5: SVM Boundary Graph

In Figure 5, we can see how the model predicted the data points graphed on an 'age' and 'bmi' scale. Blue points represent the data that were classified as 0 while the yellow points were classified as 1. From this graph, we can see one version of the boundary the model used to determine if a user has diabetes or not.

## Random Forest Classifier

A Random Forest Classifier is a machine learning method that creates a large number of decision trees during training, and outputs the prediction class that is most common among all decision trees. As typical, each individual tree works by splitting data based on features at different levels of the tree, such that the data is split into homogeneous sets, and predicting a value by traversing down from the root and reaching a leaf node. It is important to note that a Random Forest is not deterministic - at each split in each tree, a random subset of the features are used; also, each tree is made with a random subset of the data. Therefore, for the same data, the model may make different predictions in different instances. Random Forests are great at preventing overfitting and dealing with noisy data, since the randomness ensures that each tree is slightly different and therefore offers significant diversity. This, combined with the robust and convenient *RandomForestClassifier* package offered by *scikit*, made it a good choice to use for classification.

The dataset is largely skewed towards negative data, which is an issue for a Random Forest Classifier since random subsets of the data are used for each tree. Therefore, for this model, we chose to remove a significant amount of negative data, such that there was an equal amount of negative and positive data. It is noteworthy that training with all the data actually produced a higher accuracy, but also a greater BER and much higher CER.

The model involves a plethora of hyperparameters, most of which were left as default. The three hyperparameters we tuned were:

- n_estimators - the number of decision trees
- max_depth - depth of each tree
- min_samples_split - samples to split each tree's nodes

Using an 'in-house' version of GridSearch, we optimized these three parameters to have the best accuracy and CER. We found the best values to be n_estimators = 100, max_depth = 10, and min_samples_split = 10. In Figure 6 below, it is visible that these parameters result in the highest accuracy (the purple dot in the top left corner with the highest z-value).

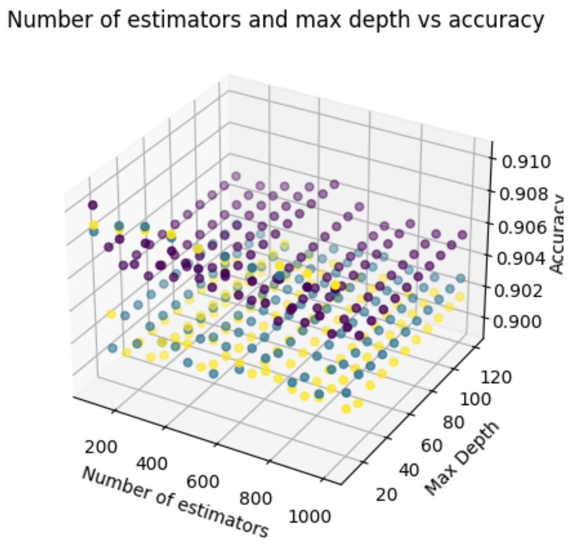


Figure 6: Random Forest Accuracy Plot

It is noteworthy that for each value of n_estimators that was tested, the optimum values of max_depth and min_samples_split were always 10 - we can see the highest CER for various n_estimator values in Figure 7. In addition, we used RandomizedSearchCV and found that n estimators = 1600 is the optimum value; however, upon testing, 100 had a better accuracy and CER.

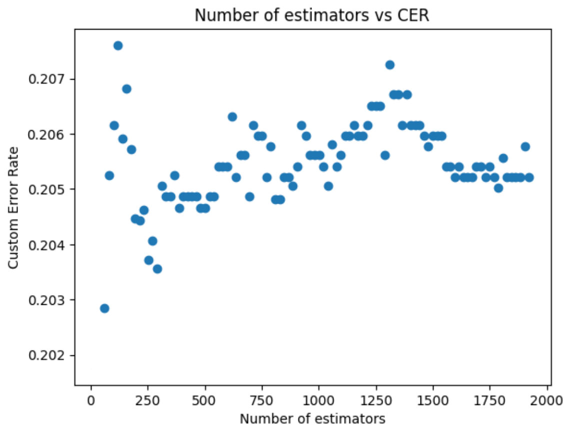


Figure 7: N Estimators vs Custom Error Rate

## Literature Review

We aren't the only ones to have done diabetes prediction via machine learning using datasets regarding physiological data. In fact, there is quite a bit of literature on this topic. However, we must first analyze the source of our data: We gathered our dataset from Kaggle after prioritizing certain factors of a desired dataset, relative size, number of features, as well as the reputability of the dataset. Mohammad Mustafa [6], the author of our dataset compiled Electronic Health Records(EHR) from a variety of Healthcare providers and anonymized the records such that they can be quantitatively analyzed. However, the identity of the Healthcare providers as well as the region that data sourced from was not available.

To introduce the literature review, we start off by discussing the recent advancements in this area. In *A review on current advances in the machine learning based diabetes prediction* [3], Varun Jaiswal, Anjli Negi, and Tarun Pal explore a plethora of machine learning models to predict presence of diabetes and report their accuracy to compare the models' relative efficacy. Although they used a variety of datasets, the two most common datasets used were the National Health and Nutriton Exmanation Survey (NHANES) and the Pima Indians Diabetes Database (PIDD), both of which contain physiological and demographic data about the patients (similar to our database). Using such similar databases, they use multiple models, such as Support Vector Machines (SVMs), various Neural Networks, Naive Bayes Algorithms, etc., but the most successful models (the highest accuracy) were combinations, such as SVM + ANN (Artificial Neural Networks), LS-SVM-MPSO (Least Squares SVM Multi-Objective Particle Swarm Optimization), and K-means +Amalgam KNN (K means clustering and K nearest neighbors). So, we note that the current state of art methods typically involve hybrid/combined model approaches. Another study by Xu et al.[9] ensembled both a Random Forest Classifier for specific feature engineering and an extreme boosting XGBoost classifier to identify risk groups for a multiclass Diabetes Risk Classifier. This method proved to be more accurate than other approaches (SVM, Naive Bayes) using the same Pima Indian Dataset (which derived from a clinical study, but was very similar to our Kaggle dataset). The paper also outlined the usage of a variety of contrast experiments to tune hyperparameters. Ultimately, the decision making ability relied on using the XGBoost for prospective studies, similar to the work presented by us, by defaulting to using a boosting algorithm for quicker convergence.

## Models in Practice

Wei et. al. used support vector machines to predict if a person had diabetes or pre-diabetes [10]. Their model was trained on data pulled from the 1994-2004 National Health and Nutrition Examination Survey (NHANES). NHANES is a sample survey that collects demographic and health history data about individuals. Out of the total variables from the survey, their team decided to use 14 variables that were commonly associated with diabetes. Out of the 14 variables they decided to use, 5 features overlapped with what we used: gender, age, bmi, hypertension, and smoking. To tune the hyperparameters, the authors used 10 folds, while we decided to use 5 for run-time purposes. The authors found that a radial function gave them the best performance, which differed from our results of an rbf kernel being the best. This difference could be due to the fact that they used more features than we did and had different categories of data.

While not relating to diabetes, Muchlinkski et. al. used a Random Forest to predict rare events in civil wars, and compared it to various versions of logistic regression. In particular, the data they used pertained to the onset of civil wars, and was class imbalanced in that the model must predict probabilities of true events but ends up returning a value close to 0 very often. The authors found that a Random Forest worked very well for class-imbalanced data, due to the fact that class weights are a part of the model and therefore it is sensitive to misidentifying even the minority class. Likely for this reason, a Random Forest worked very well for our imbalanced dataset as well [5].

## Disease Prediction

Kumar and Pavani used a dataset similar to ours in a similar fashion, specifically to diagnose the type of diabetes. In their study, they used a dataset that has 15 features and 650 records of diabetic patients from India [4]. To asses the type of diabetes an individual had, they implemented 4 different Machine Learning models, Random Forest, LDA, k-NN, SVM. They found that KNN had the lowest accuracy at 0.59, while the the accuracy of other models ranged between 0.93 and 1. One of the reasons their models produced more accurate results than ours could be because they had more features that the model could analyze. Having more features allows the model to draw better conclusions about the data, thus improving it's accuracy to classify future data.

Deberneh and Kim went a step further and used machine learning models to include pre-diabetic as a potential patient outcome. They utilized an ensemble of various models used in our research as well, including Random Forest and SVM, as well as others such as Artificial Neural Networks - however, they specifically predicted if Type 2 Diabetes would occur in the following year. In the study, a dataset with 28 features and records from 4685 instances of patient data from Korean National Health and Nutrition Examination Survey [1]. Two key takeaways from this paper pertaining to our model are that our models are appropriate for this task, and that the class imbalance problem needs to be addressed (the authors solved it using synthetic over-sampling).

In *Apply Machine Learning Methods in Diagnosing Heart Disease for Diabetic Patients* ([7]), G. Parthiban and S.K. Srivasta explore using machine learning models to predict heart disease for diabetic patients. They base their models on a dataset from a diabetes healthcare institute that incorporates features about heart disease symptoms into its data. They use SVMs and a Naive Bayes approach, with the SVMs performing better. They determine a high accuracy with correlation to the diabetic and heart disease symptom features, highlighting the salience of our heart disease feature used in our model.

## Results

We found that while most models discussed performed better than the baseline logistic regression, the XG Boost model performed the best with the given dataset. The accuracy, BER, and CER for each model are in Table 6. When generalizing our model, a false negative means that we predicted a person did not have diabetes when they actually did, which can have devastating effects. To combat this, we decided to penalize false negatives when evaluating the error of a model.

Table 6: Accuracy, BER & CER for each model

| | **Acc** | **BER** | **CER** |
|---|---|---|---|
| **Logistic** | 0.888 | 0.111 | 0.247 |
| **XG Boost** | 0.913 | 0.073 | 0.186 |
| **SVM** | 0.910 | 0.131 | 0.316 |
| **RF** | 0.916 | 0.090 | 0.202 |

Furthermore, across all models, we found that the weights of features (in other words, which features were the most important) were roughly consistent(with some slack). HbA1c level and blood glucose level were the strongest predictors, followed by age and BMI. Figure 8 is based on the weights from the Random Forest model, but all models followed a similar trend.

HbA1c level is the most correlated to diabetes prediction, which makes sense given that the HbA1c test measures how much blood sugar is attached to hemoglobin. Various tests have been developed to determine the levels of this protein, and in fact HbA1c concentration levels are being applied more and more in diabetes diagnosis in practice [8]. Furthermore, the correlation of blood glucose level and diabetes is trivial - the disorder itself causes high blood sugar.

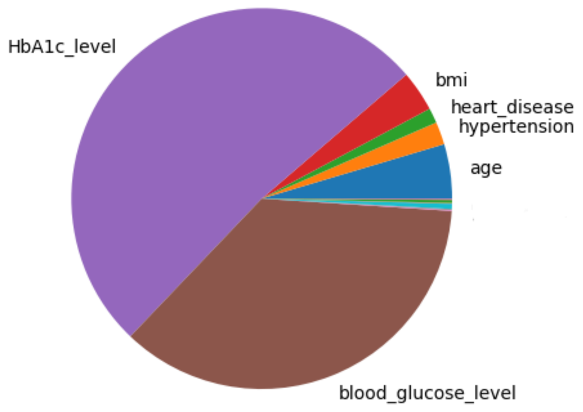


Figure 8: Feature Weights

Possible reasons as to why XGBoost performed the best is due to its aggressive gradient descent approach (extreme boosting). As it is a Deep Learning Approach, the idea of having both scores and clusters associated for a given input vector causes its accuracy to be slightly more accurate than other approaches. Subsequently, Decision Tree algorithms performed incrementally higher than conventional Classifiers. One other interesting factor is that an XGBoost, rather than ensembling random decision trees like, builds very weak learners/lower depth decision trees and builds up a classifier that is composed of all of them. This scalable approach not only is able to train quite fast, but also mitigates the effects of overfitting as **no single sub-tree has a specificity that is tuned directly towards the training set**, and generalizations can successfully be made about similar distributions.

Refer to the Section "Models/Analysis" for an in-depth explanation of the design choices that guided constructing an XGBoost Model(how we can interpret our parameters). The two main goals that we aimed to do was minimize CER and maximize the accuracy(an auxiliary goal was to have a reasonable training period). Thus, the parameters at our disposal(number of estimators dictates the training time as well as the complexity of our model, the maximum depth of our Decision Tree Model also impacts the complexity of our model(the more clusters/families, the harder it is to derive meaning from a given subtree), and the min_child_weight corresponds to how impactful a given predictive output is). Our model has a lot of estimators, corresponding to a very *broad* decision tree(with many different estimators corresponding to smaller subclasses of data). The max_depth is also quite small, showing that our model is not as complex as depth is quite small. We have created a very stratified yet robust decision tree to tackle our binary classification, that weights each cluster to some extent for a given output.

As mentioned earlier, a shortcoming of the dataset is that it was highly skewed towards negative data, or data from patients without diabetes. We were able to account for this by removing portions of the data, which was crucial due to the importance of ensuring the model is not trained to overpredict 0's and create false negatives. In the future, we can use synthetic data to combat this problem as well. Rather than remove data, we can use generative AI models to artificially generate data, instead of obtaining it from observations. Recent developments in this technology have allowed for the creation of data that follows similar statistical trends as real data, making it suitable for training and testing a machine learning model.

In regard to the feature representations that worked best, we tried normalizing our features with respect to their maximum values seen during training and their average values seen during, yet the changes were marginal and generally insignificant. Due to the marginal effect, we decided to use the features' quantitative values (specifically, our quantitative representations) directly for our models.

Overall, our models were able to predict diabetes existence to a high level of accuracy, while avoiding as many false negatives as possible. While we can change some factors in the future - such as what models were used, using an ensemble of models, using synthetic data, etc. - we consider our model to be successful in its predictive task. Technology like this one has great potential in the medical field, and we are sure that our model and other similar technologies will be widely adopted soon.

## References

[1] Henock Deberneh and Intaek Kim. Prediction of type 2 diabetes based on machine learning algorithm. *International journal of environmental research and public health*, 2021.

[2] Centers for Disease Control and Prevention. What is diabetes?, 2023.

[3] Varun Jaiswal, Anjli Negi, and Tarun Pal. A review on current advances in machine learning based diabetes prediction. *Primary Care Diabetes*, 15(3):435–443, 2021.

[4] P Suresh Kumar and S Pranavi. Performance analysis of machine learning algorithms on diabetes dataset using big data analytics. In *2017 international conference on infocom technologies and unmanned systems (trends and future directions)(ICTUS)*, pages 508–513. IEEE, 2017.

[5] David Muchlinksi, David Siroky, Jingrui He, and Matthew Kocher. Comparing random forest with logistic regression for predicting class-imbalanced civil war onset data. *Political Analysis*, 2016.

[6] Mohammad Mustafa. Diabetes prediction dataset, 2023.

[7] G Parthiban and SK Srivatsa. Applying machine learning methods in diagnosing heart disease for diabetic patients. *International Journal of Applied Information Systems*, 3(7):25–30, 2012.

[8] Cas WeyKamp. Hba1c: A review of analytical and clinical aspects. *Annals of laboratory medicine*, 2013.

[9] Zhongxian Xu and Zhiliang Wang. A risk prediction model for type 2 diabetes based on weighted feature selection of random forest and xgboost ensemble classifier. In *2019 Eleventh International Conference on Advanced Computational Intelligence (ICACI)*, pages 278–283, 2019.

[10] Wei Yu, Tiebin Liu, Rodolfo Valdez, Marta Gwinn, and Muin J Khoury. Application of support vector machine modeling for prediction of common diseases: the case of diabetes and pre-diabetes. *BMC medical informatics and decision making*, 10(1):1–7, 2010.