Faculty of Engineering & Technology

Electrical & Computer Engineering Department

Machine Learning and Data Science – ENCS5341

Assignment #3

Prepared by:

Dana Bornata	1200284	**Section:** 1
Zahia Elabour	1190610	**Section:** 2

Instructor: Dr. Yazan Abu Farha

Date: Jan 20, 2024

Table of Contents

Table of Tables

Table of Figures

Introduction

The aim of the project

The goal of this project is to evaluate several machine learning models' abilities to predict an outcome on an actual problem using a dataset (determining if an individual has diabetes or not). This project builds, tests, and evaluates different models, including Random Forest, Support Vector Machine (SVM), as well as K-Nearest Neighbour (KNN) as a baseline model. The chosen models were first tuned with their hyperparameters, and then evaluated against the baseline model, comparing evaluation criteria.

Machine Learning Models

Three models are used: K-nearest neighbor (KNN), Random Forests (RF), and Support Vector Machine (SVM).

1. KNN

The KNN algorithm is used for classification of discrete data, numerical or categorical. It does not require parameters other than (k). The prediction of a data point (x) will depend on the classification of other K neighboring points. The KNN algorithm uses various distance metrics to find the K neighbors around the point, one such distance is the Euclidean distance [2] that we use in the project. In this project, a KNN classifier implementation of the python scikit-learn library was used.

The parameters used to tune the performance of the model:

- **metric**: this is the distance metric. It finds the distance between given points. Many distance functions can be used such as (Euclidean, Manhattan, etc.)
- **K**: the number of the neighboring points that have the smallest distance with the query point, calculated using the distance formula specified by the “metric” parameter.

2. Random Forest

Random Forest is an ensemble learning technique that is used for classification. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."[3] Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. For this project, Random Forest classifier implementation of scikit-learn python library was used.

The hyper-parameters used to tune the performance of the model:

- **max_features parameter:** which determines the maximum number of features considered for splitting at each node in an individual tree. Various values were tested to evaluate their impact on the Random Forest model's performance.

3. Support Vector Machine

Support Vector Machine (SVM) is another supervised machine learning algorithm that can be used for classification. The main objective of the SVM algorithm is to find the optimal hyperplane that can separate the data points in different classes in the feature space. The margin between the points of different classes should be as large as possible, to provide a good classification [4].

The hyper-parameters used to tune the performance of the model:

- **Kernel**: The SVM model is trained and evaluated with different kernels (e.g. linear, polynomial, rbf, sigmoid) to determine the impact of the kernel choice on performance.
- **C**: C is a regularization parameter that influences the trade-off between achieving a smooth decision boundary and classifying training points correctly.

The evaluation metric

In the evaluation of the models, several evaluation metrics were used to evaluate their performance. Why they are used for the chosen dataset is explained below.

1. Accuracy:

Accuracy measures the overall correctness of the model's predictions. Accuracy is not the best measure when the classes are highly skewed.

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

Figure 1: Accuracy account

2. Recall:

Recall, also known as sensitivity or true positive rate, measures a model's ability to correctly identify positive cases.

$$TPR(recall\ or\ sensitivity) = \frac{TP}{TP + FN}$$

Figure 2: Recall account

In the context of our dataset, which involves diagnosing diabetes patients, the emphasis here is to avoid diagnosing someone as diabetic-free, when in reality they have the disease.

Recall is preferred because it minimizes instances where the model fails to identify actual positive cases. For this reason, when comparing the models in the following sections, models with higher recall will be selected.

Dataset

Dataset Overview

The dataset used is a collection of medical data from more than 100,000 people, and whether they have diabetes or not. The data includes features such as age, gender, body mass index (BMI), hypertension, heart disease, smoking history, HbA1c level, and blood glucose level[1].

Exploratory Data Analysis (EDA)

EDA is a technique used to get insights and information about the data. And while there are several techniques to be done in the EDA, we chose some of the most important ones, such as description and information of features (mean, std, minimum, etc) using the functions describe() and info(), as well as detecting null and negative values (which we found none of), finding feature correlation, and plotting some pairs of features to further explore the correlation between them

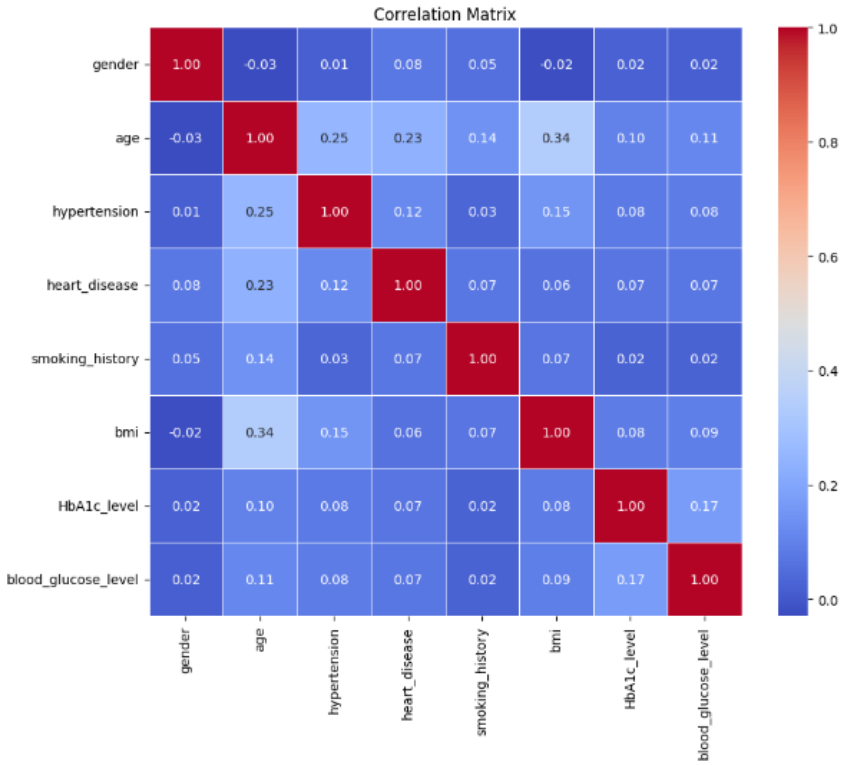

Figure 3: Correlation matrix between features

We plot the features with the most and least correlation which are (age and BMI) and (age and gender) with correlations of 0.33 and of -0.03 respectively.

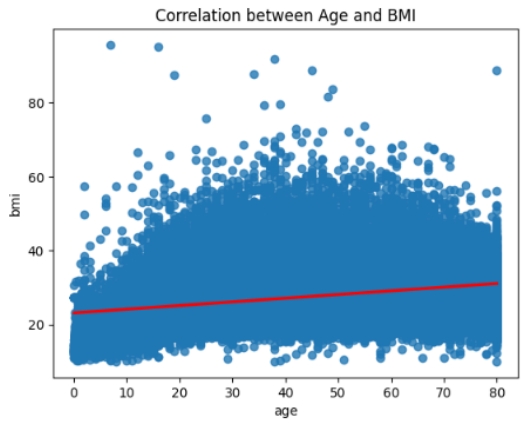

Figure 4: Features with highest correlation

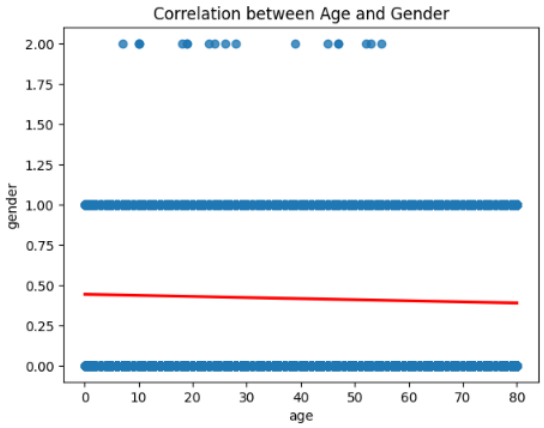

Figure 5: Features with lowest correlation

A useful insight from the plots above can be that there is a positive correlation between age and the body mass index (age and BMI). The plots also reassures the logical conclusion that there is no correlation between features such as age and gender, hence the negative correlation.

Experiments and Results

1. Baseline model

For the KNN (the baseline model), There are some hyperparameters that must be adjusted: the distance metric and the value of k. These parameters are crucial in determining how the algorithm behaves and, in turn, how predictive it is. The goal is to choose the best performing model with the highest recall rate, so the performance for 2 KNN models with k = 1 and k = 3 was evaluated on the testing set. The results were as follows:

Table 1: Knn Baseline model for K=1, K=3

Model	***K***	***Dist. Metric***	***Accuracy***	***Recall***
knn_baseline_1_1	k=1	Euclidean	0.937367	0.587359
knn_baseline_3_1	k=3	Euclidean	0.946933	0.530033

The highest value for recall on the testing data was achieved when using k=1. So the baseline model is the KNN with K=1. The table below summarizes the performance of the chosen baseline model.

Table 2: Performance of KNN with K=1

Selected Model on testing	*Accuracy*	*Recall*
Random Forest	0.94	0.58

2. Random Forest model

Firstly, the hyper-parameter "maximum number of features" was tuned. Multiple options for the parameter were tested, such as ('sqrt', 'log2', 0.6, 0.1). The models were trained on the split train set, and evaluated on the validation set. Accuracy and recall of each model are shown in the table below:

Table 3: Accuracy and recall of each model in Random Forest

Model	*max_features*	*Accuracy*	*Recall*
random_1	sqrt	0.972401	0.686275
random_2	log2	0.972251	0.690731
random_3	0.6	0.971951	0.696078
random_4	0.1	0.972401	0.685383

Based on the results, we will choose the model with the "max_features" hyper-parameter set to (0.6) for the Random Forest algorithm, as it achieved the highest recall rate among the tested models. The table below summarizes the performance of the chosen random forest model with (max_features= 0.6).

Table 4: Performance of Random Forest with max_features = 0.6

Selected Model on testing	*Accuracy*	*Recall*
Random Forest	0.97	0.69

3. Support vector machine model

In this part, the support vector machine model was used to perform classification on the testing data. To choose the best SVM model that performs the best, hyperparameter tuning is done on the validation set, and the model with tuned hyperparameter that performs the best on validation set is then chosen and tested on the testing data. In this part, we chose 2 hyperparameters instead of 1. The first hyperparameter used is the kernel, and we try with different kernel functions to get the kernel that performs the best on the validation set as we mentioned previously.

Table 5: SVM Model Performance with Different Kernels and Hyperparameters

Model	*Kernel*	*C*	*Accuracy*	*Recall*
svm_model_1	linear	1	0.959952	0.571301
svm_model_2	poly	1	0.949228	0.410873
svm_model_3	rbf	1	0.944353	0.338681
svm_model_4	sigmoid	1	0.862532	0

As seen from above, the model with the best performance is the linear kernel. So, then we try tuning the hyperparameter "C" to different values,

Table 6: SVM Model Performance with Different Values of C (Regularization Parameter) using Linear Kernel

Model	***Kernel***	***C***	***Accuracy***	***Recall***
svm_model_1	linear	0.1	0.959877	0.569519
svm_model_2	linear	1	0.959952	0.571301
svm_model_3	linear	5	0.959427	0.590909
svm_model_4	linear	10	0.958077	0.592692

As seen from above, the best model is the model that uses C=10 as it gives the best recall and accuracy. After the model is selected with kernel= linear and C=10, we can test the performance of the model on predicting and classifying the testing data.

Table 7: SVM Model Performance on Testing Set with C=10

Selected Model on testing	***Accuracy***	***Recall***
SVM	0.96	0.57

Analysis:

In this section, we will compare the performance of the chosen models from the previous steps, and then get the model with the best performance. The models the performed the best after experimentation and hyperparameter tuning: Baseline model: KNN with k=1, Random forest model with parameter max_feature= 0.6, and SVM model with parameters kernel= "linear", and C=10.

The table below is a summary of their performance:

Table 8: Comparison of Models Performance

model	***Accuracy***	***Recall***
Baseline model_KNN	0.94	0.58
Random Forest	0.97	0.69
SVM model on the testing set	0.96	0.57

The best performance is measured by highest recall, and from the table above, the model with the best performance is ***the random forest model***, with a recall of 69% and accuracy of 97%. Compared to the baseline model (the KNN), the recall has improved by 11%, this means that less FNs are predicted, and the accuracy has improved by 3%. This improvement comes from the fact that using random forest classifier is more suitable for large datasets, and it can balance the classes [2]

The SVM model on the other hand, did not provide significant improvements on the performance when compared to the baseline model. The accuracy and recall are very similar between the two models. This can be explained by the fact that the SVM is not the perfect model

to use for large datasets, or if classes have overlapping (due to noise for example) points and cannot be separated by a hyperplane.[3]

Even though the accuracy is high in the selected model (the random forest model), we can still aim for a higher recall than (0.69). One thing we noticed is that the number of the points from the classes is not close:

Table 9: Class Distribution in the Dataset

Class	***Count***
0	91500
1	8500

And since we want to have emphasis on the positive class (having diabetes), and to reduce the chances of not diagnosing a person who actually has diabetes, we gave weights to the class, such that the positive class (which has less data points) gets a higher weight that the negative class (not having diabetes). This can be done by tuning the parameter (class weight) of the random forest classifier. The summary of performance before and after the enhancement are seen through the confusion matrices and table 10:

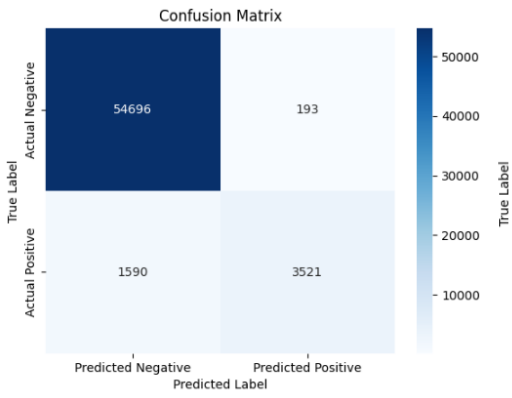

Figure 6: Confusion matrix before

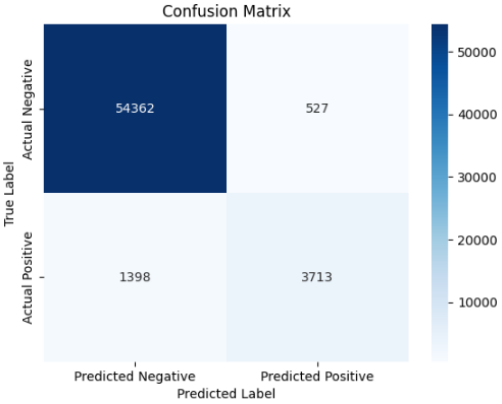

Figure 7: Confusion matrix after

Table 10: Models Performance Metrics Comparison

model	***accuracy***	***recall***	***precision***
Random forest	0.97	0.69	0.95
Enhanced random forest	0.97	0.73	0.88

As it can be seen from above, the recall has improved by 4%, and the accuracy remains the same. However, the precision will be reduced, this is because the number of FNs decreased, but the number of FPs increased. It can be said that there is a tradeoff between recall and precision, and improvement on one lead to reduction of the other.

Conclusions

In this project, the performance of three classification models was evaluated on a diabetes study dataset collected from 100,000 people. After specifying the baseline model to be the KNN classifier with K=3, a random forest classifier as well as an SVM classifier were used and compared to the performance of the baseline model. To ensure better performance is achieved, hyper-parameter tuning was performed for both models, where different models were tested on the validation set, and the ones that performed the best were selected and then tested on the testing set. Performance here is measured based on the highest recall as to reduce the false negatives as much as possible. The accuracy of the model is also taken into account. The model that performed the best was found to be the random forest model, and then its performance was enhanced as well by tuning another hyperparameter for the class weights.

Some limitations of the models were noticed, such as the drawback of the SVM model on large datasets and the long time it takes. Some general limitation in the KNN model is its sensitivity to K and the distance metric, and its expensive computation, however these were not noticed in our case. Other than that, no other significant limitations were noticed.

References

[1] [Online]. Available: https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset?fbclid=IwAR1gkKRETVuoUVixLCRU1dHNJc-EJ-p6yKJZXQutNX3k8rCRLpOg3x9GlBk. [Accessed 20 1 2024].

[2] [Online]. Available: https://corporatefinanceinstitute.com/resources/data-science/random-forest/. [Accessed 23 1 2024].

[3] [Online]. Available: https://towardsdatascience.com/everything-about-svm-classification-above-and-beyond-cc665bfd993e. [Accessed 23 1 2024].

[4] [Online]. Available: https://www.geeksforgeeks.org/k-nearest-neighbours/. [Accessed 15 1 2024].

[5] [Online]. Available: https://www.javatpoint.com/machine-learning-random-forest-algorithm. [Accessed 15 1 2024].

[6] [Online]. Available: https://www.javatpoint.com/machine-learning-random-forest-algorithm. [Accessed 16 1 2023].