Project 1 - PIMA Indian Diabetes Prediction

Thai Boonchai (6322790138) Preravitch Siripanich (6322773761)

Arnuparp Cheammarerng (6322770346) Phanuwich Thepnok (6322672730)

Department of Information, Computer and Communication Technology (ICT), Sirindhorn International Institute of Technology (SIIT), Thammasat University, Khlong Luang, Pathum Thani, Thailand

1. Team and Contribution

Our group comprises four dedicated individuals. Which are Thai Boonchai, Preravitch Siripanich, Arnuparp Cheammarerng, and Phanuwich Thepnok. each bringing their unique skills and contributions to the table. Together, we form a dynamic team that excels in various aspects of our project. By the first person is Thai Boonchai, Thai is responsible for researching an information and working on project report. His ability to summarize information and write a report in term of an academic tone helps us to make this project smoothly. For Preravitch and Arnuparp, is responsible for build a model and optimize our neural network. Their ability to analyze and build a model helps to make this project successful. And lastly, Phanuwich is responsible for optimizing our neural network and summarizing the report. His skill makes our project to be done in the most perfect way.

2. Introduction and data analysis

[1], [3] Diabetes, a chronic metabolic disorder, arises from the insufficient production of insulin by the pancreas or the body's diminished ability to utilize the insulin it generates. Insulin, a vital hormone, plays a pivotal role in regulating blood sugar levels. The consequence of uncontrolled diabetes is hyperglycemia, characterized by elevated blood sugar levels, which, over time, exacts a profound toll on various physiological systems, particularly the nervous and vascular systems.

[4], [5] This pressing issue is In India, where 77 million people aged 18 and older suffer from type 2. diabetes, and nearly 25 million are prediabetics with a high risk of developing the disease soon, this urgent problem is especially acute. Alarmingly, more than half of these people are unaware that they have diabetes, which can lead to crippling health problems if early detection and intervention are delayed. Indeed, the risk of having a heart attack or stroke is frighteningly increased in adults with diabetes by a factor of two to three. Additionally, neuropathy-induced nerve damage in the extremities increases the likelihood of foot ulcers, infections, and, ultimately, the urgent need for limb amputation in conjunction with impaired blood circulation. Diabetic retinopathy is yet another terrible side effect. a major contributor to blindness, precipitated by the cumulative damage inflicted upon the delicate retinal blood vessels over prolonged periods. It is imperative to recognize that diabetes stands as one of the primary causative factors behind the scourge of kidney failure.

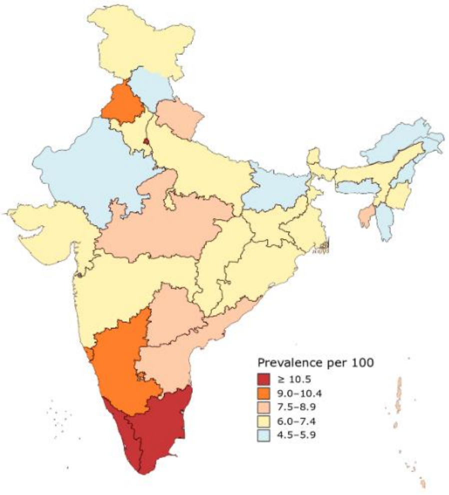
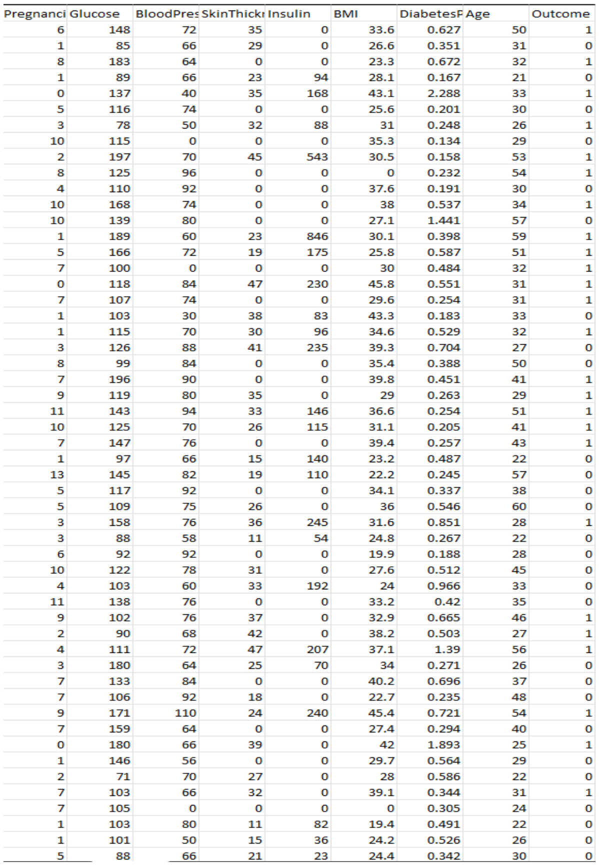


Figure 1 Prevalence of diabetes in Indian state

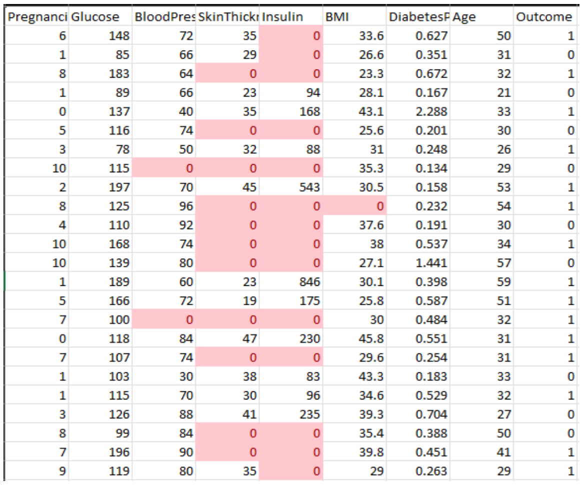
The PIMA Indian Diabetes database, which is meticulously maintained by the National Institute of Diabetes and Digestive and Kidney Diseases, provided the dataset Figure 2 that was used in this academic investigation. The 768 instances in this meticulously organized dataset have been thoughtfully divided into two groups: 268 instances that indicate a positive diabetes diagnosis and 500 instances that indicate a negative diabetes diagnosis. Within this dataset, one encounters an array of eight distinct features, namely, the count of pregnancies (times), plasma glucose concentration during a two-hour oral glucose tolerance test (measured in mg/dL), Diastolic blood pressure (in mmHg), Triceps skinfold thickness (in mm), 2-Hour serum insulin levels (in U/ml), Body Mass Index (in kg/m2), the Diabetes Pedigree Function, and Age (in years). And one target variable which is outcome.



Pregnanci	Glucose	BloodPres	SkinThickr	Insulin	BMI	DiabetesF	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1
7	100	0	0	0	30	0.484	32	1
0	118	84	47	230	45.8	0.551	31	1
7	107	74	0	0	29.6	0.254	31	1
1	103	30	38	83	43.3	0.183	33	0
1	115	70	30	96	34.6	0.529	32	1
3	126	88	41	235	39.3	0.704	27	0
8	99	84	0	0	35.4	0.388	50	0
7	196	90	0	0	39.8	0.451	41	1
9	119	80	35	0	29	0.263	29	1
11	143	94	33	146	36.6	0.254	51	1
10	125	70	26	115	31.1	0.205	41	1
7	147	76	0	0	39.4	0.257	43	1
1	97	66	15	140	23.2	0.487	22	0
13	145	82	19	110	22.2	0.245	57	0
5	117	92	0	0	34.1	0.337	38	0
5	109	75	26	0	36	0.546	60	0
3	158	76	36	245	31.6	0.851	28	1
3	88	58	11	54	24.8	0.267	22	0
6	92	92	0	0	19.9	0.188	28	0
10	122	78	31	0	27.6	0.512	45	0
4	103	60	33	192	24	0.966	33	0
11	138	76	0	0	33.2	0.42	35	0
9	102	76	37	0	32.9	0.665	46	1
2	90	68	42	0	38.2	0.503	27	1
4	111	72	47	207	37.1	1.39	56	1
3	180	64	25	70	34	0.271	26	0
7	133	84	0	0	40.2	0.696	37	0
7	106	92	18	0	22.7	0.235	48	0
9	171	110	24	240	45.4	0.721	54	1
7	159	64	0	0	27.4	0.294	40	0
0	180	66	39	0	42	1.893	25	1
1	146	56	0	0	29.7	0.564	29	0
2	71	70	27	0	28	0.586	22	0
7	103	66	32	0	39.1	0.344	31	1
7	105	0	0	0	0	0.305	24	0
1	103	80	11	82	19.4	0.491	22	0
1	101	50	15	36	24.2	0.526	26	0
5	88	66	21	23	24.4	0.342	30	0

Figure 2 diabetes.csv (Pima Indians Diabetes dataset)

However, as a result of the dataset problem, certain issues have come to our attention. Specifically, there is a notable concern pertaining to the presence of missing values within several columns, denoted by zeros. For instance, the "Glucose," "BloodPressure," "SkinThickness," "Insulin," and "BMI" columns exhibit such zero values, a situation that casts doubt on the accuracy of the information regarding these attributes. This phenomenon might be explained by a bias brought on by missing data. As **Figure 3** will demonstrate,



Pregnanci	Glucose	BloodPres	SkinThickr	Insulin	BMI	DiabetesF	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1
7	100	0	0	0	30	0.484	32	1
0	118	84	47	230	45.8	0.551	31	1
7	107	74	0	0	29.6	0.254	31	1
1	103	30	38	83	43.3	0.183	33	0
1	115	70	30	96	34.6	0.529	32	1
3	126	88	41	235	39.3	0.704	27	0
8	99	84	0	0	35.4	0.388	50	0
7	196	90	0	0	39.8	0.451	41	1
9	119	80	35	0	29	0.263	29	1

Figure 3 Missing data which is unlikely to be valid data for these attributes.

With the x-axis designating the binned glucose level range and the y-axis representing the frequency, or the count of individuals, within each respective bin, the histogram shown in Figure 4 is used to visually represent the distribution of glucose levels within the dataset. It is noteworthy to point out that depending on the outcome of interest, specifically the presence or absence of diabetes, the patterns of the histogram can change. This observation suggests that there may be a significant influence of blood glucose levels on the outcome if distinct peaks or patterns can be seen for each outcome group (diabetes and non-diabetes). On the other hand, if the histograms for these two groups substantially overlap, this may indicate that glucose levels alone are not a reliable indicator of the outcome and that additional influencing factors need to be considered.

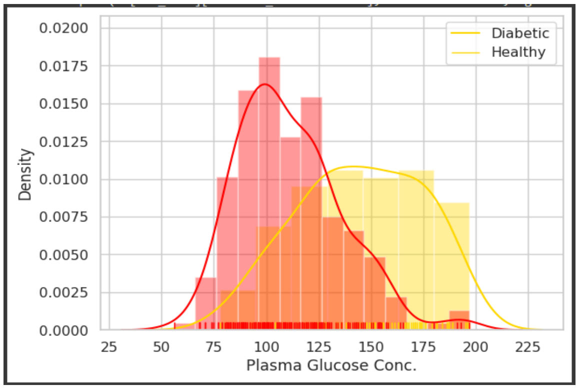


Figure 4 The histogram of the distribution of glucose levels.

3. Algorithm design

a. Overview

i) Part 1

What we've discovered on the initial model in part 1 was essential to optimizing the model. The initial model uses two hidden layers with its units 12 and 8. The model uses keras TensorFlow, 'Dense'. After training for 150 epochs of the initial model, the visualization for the model can be seen as overfitting. Since the initial model doesn't use dropout, batch normalization and no regularization, we could take this to our advantage into subduing the model being overfitted.

The initial model also uses SGD as the optimizer which would lack strength compared to Adam due to slower convergence. The model also fits X and y instead of X_train and y_train which causes data leakage.

Any data analyst would come across a dataset which obstructed them from moving forward with their analysis with anomalies, imbalanced, or as previously stated, missing data. Thus, we must find a way to prevent that obstruction, and this could be done in multiple ways with data preprocessing.

ii) Part 2

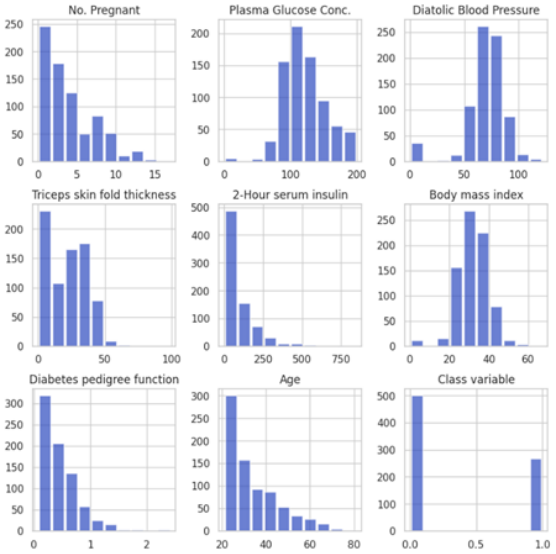


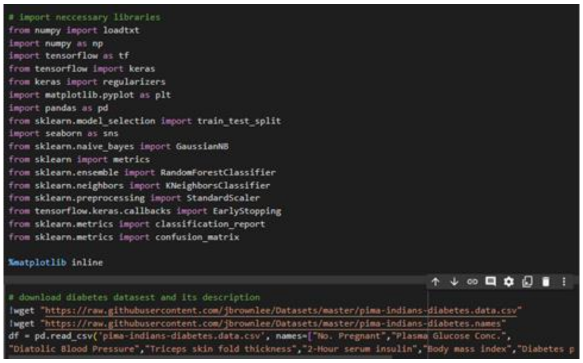
Figure 5 before preprocessing.

From **Figure 5**, this dataset contains a hefty amount of 0, especially for 'Triceps skin fold thickness' and '2-Hour serum insulin' which are filtered for 0 values from the data. These two columns contain at the very least 200 data which is input with 0. There were at least two approaches for preprocessing this dataset. One was to drop two columns which have been filled with many rows of zeroes as mentioned above. The other was to completely drop rows of data which contain at least a single 0 (excluding the Class Variable and No. of Pregnant). Another touch on preprocessing is to create an imputation procedure where we could replace the data containing zero with the average of the column or the median of the column. The outcome of dropping rows was better than the approach of dropping two columns with many zeroes. The outcome of the approach of the imputation procedure was found to be similar to the result of dropping rows containing at least a single zero.

b. Data Preparation

The initial phase of the code involves the vital process of importing essential libraries, as depicted in Figure 6. This includes the inclusion of indispensable libraries such as Pandas (pd), NumPy (np), Matplotlib (plt), TensorFlow Keras

(keras), regularizes from TensorFlow Keras, and optimizers from TensorFlow Keras. Subsequently, Figure 6 illustrates the download of two pivotal files: the primary dataset ('pima-indians-diabetes.data.csv') and a companion document containing descriptive information about the dataset ('pima-indians-diabetes.names'). The dataset is then ingested into a Pandas DataFrame denominated as 'df,' as shown in Figure 6, with appropriate column names meticulously assigned to ensure data integrity and comprehensibility.



```
# import neccessary libraries
from numpy import loadtxt
import numpy as np
import tensorflow as tf
from tensorflow import keras
from keras import regularizers
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
import seaborn as sns
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

%matplotlib inline
```

```
# download diabetes datasest and its description
!wget "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
!wget "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.names"
df = pd.read_csv('pima-indians-diabetes.data.csv', names=["No. Pregnant","Plasma Glucose Conc.",
"Diatolic Blood Pressure","Triceps skin fold thickness","2-Hour serum insulin","Body mass index","Diabetes p
```

Figure 6 Data Preparation

c. Collinearity

The PIMA DIABETES dataset is often used to predict the onset of diabetes in individuals based on various health-related features. This can cause model instability and cause overfitting. Handling collinearity can be done by feature selection and adding regularization in which we did to our optimized model.

d. Data Preprocessing

The approach that was chosen for preprocessing was different from feature selection, where rows containing zeros would be dropped out of the data frame. We decided to not perform feature selection and instead filtered out all the values to be greater than zero, excluding No. Pregnant and Class variable, which causes the 'Null' or 'NaN' to replace zero in the data frame. Then, any rows that contain 'Null' are filtered out from the data frame, which removes any zero values, resulting in Figure 7.

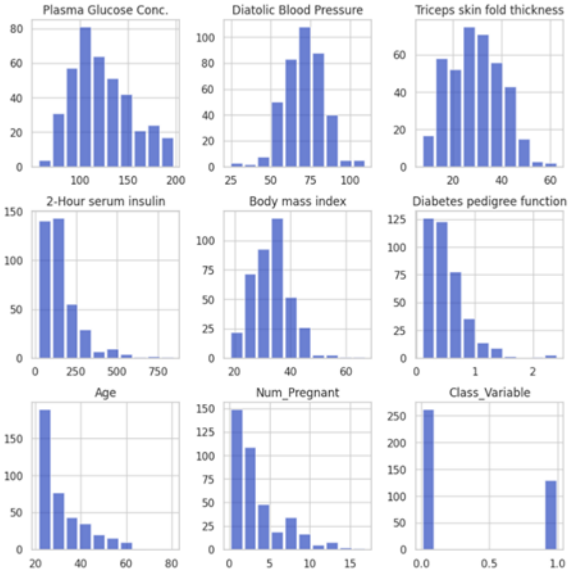
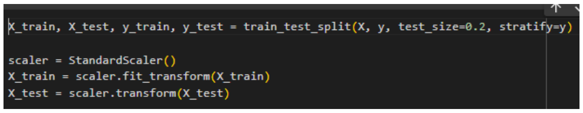


Figure 7 after preprocessing.

e. Train-Test Split with Stratification

Figure 8 shows the train-test split phase. This visual representation showcases the division of the dataset into training and testing sets using 'train_test_split' function with the stratification parameter. The use of 'stratify=y' ensures that the random partitioning of the dataset into training and testing subsets. This stratified option is important when dealing with imbalanced datasets, as it prevents skewed class distributions in both training and testing data sets.

As Figure 8 illustrates, 80% of the data is allocated to the training dataset (X_train and y_train), while the remaining 20% is earmarked for the testing dataset (X_test and y_test). This partitioning strategy is fundamental to subsequent model evaluation processes.



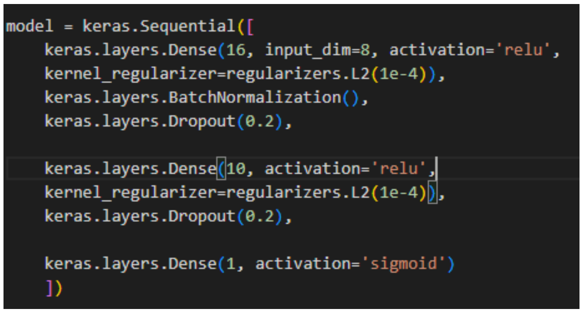
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 8 Division of the dataset into training and testing data set with Stratification

f. Neural Network Model

The neural network model depicted in Figure 10 shows the depiction of the input layer, characterized by eight input dimensions, corresponding to the number of features within the dataset. Subsequently, Figure 10 illustrates the composition of two hidden layers, 16 and 10 respectively. each comprising eight neurons. The input layer is employing the Rectified Linear Unit (ReLU) as the activation function. The culminating layer, the output layer, is depicted with a solitary neuron utilizing the sigmoid activation function, a fitting choice for binary classification tasks.



```
model = keras.Sequential([
    keras.layers.Dense(16, input_dim=8, activation='relu',
    kernel_regularizer=regularizers.L2(1e-4)),
    keras.layers.BatchNormalization(),
    keras.layers.Dropout(0.2),

    keras.layers.Dense(10, activation='relu',
    kernel_regularizer=regularizers.L2(1e-4)),
    keras.layers.Dropout(0.2),

    keras.layers.Dense(1, activation='sigmoid')
    ])
```

Figure 9 ANN Hidden Layers

I. Regularization

Figure 9 contains the application of L1L2 kernel regularization and L2 bias regularization to the input layer within the neural network model. These regularization techniques help prevent overfitting by introducing a penalty term to the loss function.

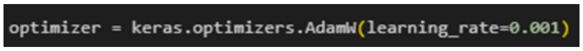
Kernel Regularization (L1L2) combines L1 (encouraging sparsity in weights) and L2 (penalizing large weights) regularization on the input layer's weights which discourages complex weight patterns while promoting sparse. Bias Regularization (L2) is applied to the biases in the input layer which helps maintain stability and constrains the magnitudes of bias terms.

II. .Dropout

Figure 19 also shows the use of dropout layers, shown after the initial hidden later and before the output layer. These dropout layers introduce randomness during training, preventing overreliance on specific neurons and improving model generalization.

g. Optimizer

Figure 10 [6] illustrates the utilization of the AdamW optimizer to orchestrate the optimization of the neural network model's weights during the training process with the default learning rate value. The inclusion of weight decay capabilities in AdamW enhances the efficiency of weight adjustments, contributing to the convergence of the model.

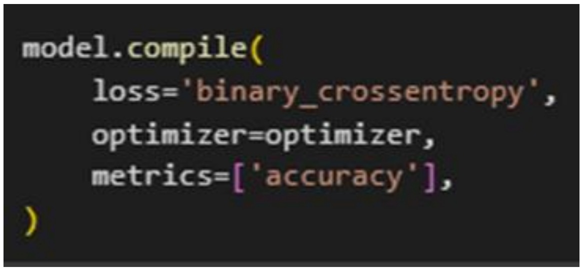


```
optimizer = keras.optimizers.AdamW(learning_rate=0.001)
```

Figure 10 AdamW Optimize

h. Model Compilation

In Figure 11, the model compilation phase is visually represented. Key attributes are specified, including the selection of binary cross-entropy as the loss function, which is a standard choice for binary classification tasks. The AdamW optimizer is denoted for weight optimization, and accuracy is designated as the evaluation metric to assess model performance.



```
model.compile(
    loss='binary_crossentropy',
    optimizer=optimizer,
    metrics=['accuracy'],
)
```

Figure 11 Model Compilation

i. Model Training

Figure 8 offers a graphical depiction of the model training phase. It commences with the invocation of the 'fit' method, training the model for 150 epochs with a batch size of 10 samples per iteration. Notably, Figure 8 portrays the utilization of both the training and validation datasets during the training process, a conventional approach, especially valuable for smaller datasets. Real-time updates, including loss and accuracy metrics, are systematically printed during training, facilitating comprehensive tracking of training progress.

j. Model Training

In total, we have used 5 classification models which are KNN, Random Forest, Naive Bayes, Decision Tree and MLP (our model). The choice of using specific classification models in a machine learning project depends on various factors, including the nature of the problem, the characteristics of the dataset, and the goals of the analysis.

K-Nearest Neighbors (KNN):

KNN is used because there is no clear assumption about the underlying data distribution, making it a good choice for exploratory analysis.

Random Forest:

Random Forest is an ensemble learning method that combines multiple decision trees to improve predictive accuracy and reduce overfitting.

Naive Bayes:

Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem. Naive Bayes is chosen when there is a need for a fast and simple algorithm, especially in cases involving text or categorical data.

Decision Tree:

Decision trees are interpretable models that make decisions by recursively splitting data based on feature attributes.

Multilayer Perceptron (MLP):

For deep learning and complex, non-linear problems. We use this because it is capable of learning intricate patterns and representations in data.

In summary, these five classification models were likely chosen because they offer a diverse set of tools.

k. Model Training

Overall, the provided algorithm design represents a meticulous framework for the creation of a neural network model, rigorous data preprocessing, and systematic model training. The incorporation of regularization techniques and the utilization of the AdamW optimizer are integral elements, as explained in Figures 10 and 11. The 'hist' variable retains historical training metrics, enabling post-training analysis and potential model refinement, as exemplified in Figure 8.

4. Evaluation

For evaluation, we choose to represent the comparison in terms of accuracy using **confusion matrix** and **precision**, **recall** and **f1 score**.

$$\boldsymbol{Accuracy = \frac{TP + TN}{TP + TN + FP + FN}}$$

A confusion matrix, also known as an error matrix, is a table used in machine learning and statistics to evaluate the performance of a classification model. It is a square matrix that summarizes the predicted and actual classifications of a machine learning algorithm on a dataset.

It consists of **four** components:

- TP: True positive ⟶ Correctly predicted as positive by the model
- FP: False positive ⟶ Incorrectly predicted as positive by the model
- TN: True negative ⟶ Correctly predicted as negative by the model
- FN: False negative ⟶ Incorrectly predicted as negative by the model

The accuracy is calculated by TP, FP, TN, FN as shown above.

Next is **Precision**, **Recal**l and **F1 score**.

Precision calculates the ratio of true positives (TP) to the sum of true positives and false positives (FP):

$$\boldsymbol{Precision = \frac{TP}{TP + FP}}$$

High precision indicates that when the model predicts a positive class, it is usually correct.

Recall calculates the ratio of true positives (TP) to the sum of true positives and false negatives (FN):

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FN}}$$

High recall means the model is good at identifying most of the actual positive instances.

F1 Score is the mean of precision and recall. It provides a balance between these two metrics and is useful when you want to consider both false positives and false negatives.

$$\boldsymbol{F1\ Score} = \frac{\boldsymbol{2 \cdot Precision \cdot Recall}}{\boldsymbol{Precision + Recall}}$$

The F1 score is particularly useful when the class distribution is imbalanced, as it considers both false positives and false negatives.

For comparison between different classifier models, we use the weighted average of F1 score and accuracy rate for global accuracy. All the models are fed with preprocessed data.

Classification Model	Accuracy Rate (%)	F1 score
KNN	78.5%	78%
Random Forest	74.7%	74%
Naive Bayes	78.5%	79%
Decision Tree	74.7%	75%
Our Model(MLP)	82.1%	83%

The high F1 score suggests that there is a good balance between precision and recall which suggests that the model is good at identifying and predicting positive cases.

a. Graph Visualization

The visualization of the performance of our MLP model at 150 epochs can be seen to converge more steadily and is more stabilized on model loss and model accuracy compared to other models. The graph visualization can be seen below.

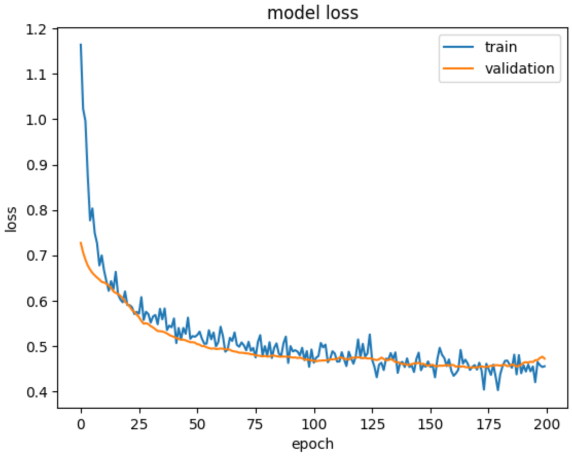


Figure 12 model loss for MLP

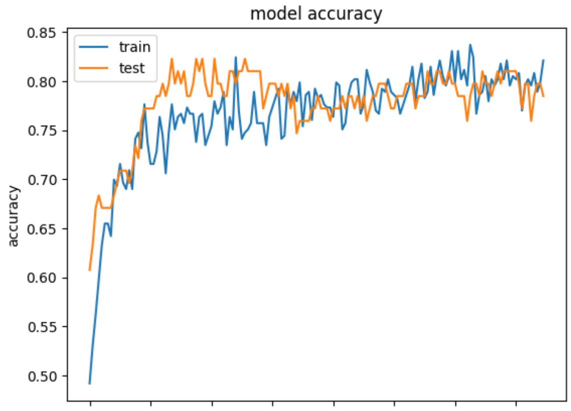


Figure 13 model accuracy for MLP

5. Conclusion

We focus on the diabetes issue in India, where millions are affected, but many remain undiagnosed. We use the PIMA Indian Diabetes dataset with 768 instances and eight features.

Our algorithm, our initial model, we have a lot of problems. Model stability and overfitting management are hampered by a lack of batch normalization, dropout, and regularization. Data leaking owing to incorrect model usage, and the presence of many zero values in the dataset. Fit using x and y rather than X_train and y_train, and the initial model uses SGD rather than the more efficient Adam optimizer.

So, to control overfitting, we include batch normalization and dropout layers for model stability, as well as regularization (L1, L2). Then, in the Dataset problem, treat multiple zero values based on their significance or impute missing data. To prevent data leaks with model. Fit, use X_train and y_train for training. Improve optimizers by utilizing Adam Optimizer instead of SGD.

In comparison, we use 4 classification model by using KNN with 78.5% accuracy rate and 78% F1 score, Random Forest with 74.7% accuracy rate and 74% F1 score, Naive Bayes with 78.5% accuracy rate and 79% F1 score, and Decision Tree with 74.7% accuracy rate and 75% F1 score to compare with our model(MLP) with 82.1% accuracy rate and 83% F1 score. As you can see, our model is the most effective.

Overall, our project shows promise in diabetes prediction, with potential for significant impact in healthcare. However, we can better prove the model if we have more and diverse information.

6. References

[1] World Health Organization, "Diabetes in India," https://www.who.int/india/health-topics/mobile-technology-for-preventing-ncds

[2] Kaggle. "Pima Indians Diabetes Database." Kaggle, https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database?resource=download.

[3] Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the Symposium on Computer Applications and Medical Care (). IEEE Computer Society Press.

[4] Times of India. "Why India is diabetes capital of the world." Nov 14, 2022.: https://timesofindia.indiatimes.com/india/why-india-is-diabetes-capital-of-the-world/articleshow/95509990.cms

[5] NCBI. Pima Indians diabetes mellitus classification based on machine learning (ML) algorithms.

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8943493/.

[6} Hasty AI (n.d.) "AdamW." Hasty AI Documentation. Available at: https://hasty.ai/docs/mp-wiki/solvers-optimizers/adamw#:~:text=AdamW%20is%20very%20similar%20to,AdamW%20fixes%20this%20implementation%20mistake.

[7] ArogyaWorld. (2013) India Diabetes Fact Sheets. http://www.arogyaworld.org/wp-content/uploads/2010/10/ArogyaWorld_IndiaDiabetes_FactSheets_CGI2013_web.pdf.

[8] Lakhwani, Kamlesh; Bhargava, Sandeep; Hiran, Kamal Kant; Bundele, Mahesh M.; Somwanshi, Devendra. "Prediction of the Onset of Diabetes Using Artificial Neural Network and Pima Indians Diabetes Dataset." In 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE), 2020. IEEE. Available online: https://ieeexplore.ieee.org/document/9358308

[9] Kala, Rahul; Shukla, Anupam; Tiwari, Ritu. "Comparative analysis of intelligent hybrid systems for detection of PIMA indian diabetes," in 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Year: 2009, Publisher: IEEE, DOI: https://ieeexplore.ieee.org/document/5393877.

[10] Paul, Banibrata; Karn, Bhaskar. "Diabetes Mellitus Prediction using Hybrid Artificial Neural Network." In 2021 IEEE Bombay Section Signature Conference (IBSSC), 2021. IEEE. Available online: https://ieeexplore.ieee.org/document/9673397.