Diabetes Prediction using *Artificial Neural Network*

Subject Code: 7088CEM
Student ID: 13366670
Student Name: Aditya Ashok Gujar

Contents

Introduction

An estimated 463 million people worldwide suffer from diabetes, making it a significant global health issue. The prevalence of diabetes has been rising quickly in past few years, and very likely to continue for the foreseeable coming years.

The rise in obesity is one of the main causes of the global diabetes epidemic. A significant risk factor for type 2 diabetes, which makes up the majority of cases of diabetes worldwide, is obesity. Obesity and diabetes rates have increased as a result of dietary and lifestyle changes, as well as increased access to processed foods and sedentary employment.

Another important contributing cause to the rising prevalence of diabetes is the ageing of the world's population. Diabetes and other chronic diseases are more likely to occur as people age. Additionally, there is proof that genetics contribute to the onset of diabetes and that some racial and ethnic groups are more predisposed to the condition.

Both in terms of its impacts on people's health and on the economy, diabetes has a substantial impact. Many consequences, such as cardiovascular disease, kidney damage, and blindness, can result from diabetes. In addition to being expensive, diabetes care and treatment can also be difficult for many people to afford.

The prevention and management of the diabetes epidemic have received the majority of attention globally. Promoting healthy habits, such as frequent exercise and a balanced diet, is thought to be essential for lowering the chance of acquiring diabetes. For an early diagnosis and efficient management of the condition, better healthcare access and diabetes education are also crucial.

In order to combat the diabetes epidemic, several nations have put laws and programmes into place, including public health campaigns and attempts to increase access to diabetes treatment and medication. The need for more financing and resources to solve the expanding problem is one of the many difficult obstacles that must yet be addressed.

Diabetes prediction using Deep Learning (DL) is another approach that has been shown to be effective in identifying persons prone to developing diabetes. DL algorithms can analyse large and complex data sets to identify subtle patterns that may not be apparent to human observers. By using such models, healthcare providers can develop more accurate predictive models for diabetes risk assessment.

The significance of diabetes prediction using DL lies in the fact that diabetes is a complex and multifactorial disease. DL algorithms can consider all of these factors and identify patterns that may be missed by traditional methods.

In addition, DL can also be used to develop more personalized treatment plans for individuals who have already been diagnosed with diabetes. By analysing data such as blood glucose levels, medication use, and lifestyle factors, DL algorithms can identify the most effective treatment options for each individual, leading to better health outcomes and improved quality of life.

Overall, diabetes prediction using DL has the potential to transform healthcare by enabling early identification and personalized treatment of diabetes, leading to improved health outcomes and reduced healthcare costs. DL algorithms can analyse large and complex data sets, enabling healthcare providers to develop more accurate predictive models and personalized treatment plans.

In the context of diabetes prediction, DL algorithms can analyse a wide range of data points, including glucose levels, body mass index, age, sex, genetics, and lifestyle factors such as diet and exercise habits. By analysing all of these factors together, DL algorithms can identify patterns and make predictions about an individual's risk of developing diabetes.

DL has several advantages over traditional methods of diabetes prediction. Firstly, it can analyse large and complex data sets, enabling healthcare providers to develop more accurate predictive models. Secondly, it can identify subtle patterns that may not be apparent to human observers, leading to more accurate predictions of diabetes risk. Thirdly, DL can adapt and learn from new data, allowing healthcare providers to continually refine and improve their predictive models.

The significance of DL in diabetes prediction lies in its ability to enable early identification and personalized treatment of diabetes (Sharma et al., 2020). By accurately identifying persons prone to the risk of developing diabetes, healthcare providers can implement preventive measures to reduce the likelihood of developing the disease. Additionally, by analysing data such as blood glucose levels and lifestyle factors, DL algorithms can develop personalized treatment plans for individuals who have already been diagnosed with diabetes, leading to better health outcomes and improved quality of life.

Related Work

Many different medical predictor factors (also called independent variables) and a single outcome variable (dependent variable) make up each dataset. Independent variables include things like a patient's age, BMI, insulin levels, and the number of pregnancies they've already had.

For this challenge, the PIMA Indians Diabetes dataset is being used that can be found in Kaggle's data pool. Researchers interested in improving their methods for predicting diabetes prevalence are eager to get their hands on this dataset. The NIDDK provided the raw data that was used in this study (National Institute of Diabetes and Digestive and Kidney Disorders). The dataset's primary function is to detect instances of diabetes by using the collection's previously established diagnostic criteria. These patients are all PIMA Indian women above the age of 21.

Diabetes risk prediction algorithms developed using DL have been employed in several research. To forecast the likelihood of developing type 2 diabetes, Wang et al. (2020) used data from electronic health records in conjunction with a deep neural network. Predicting diabetes risk in a Chinese population, the study attained an AUC of 0.876. Similarly, Karim et al. (2020) employed retinal scans and a convolutional neural network to assess the likelihood of developing type 2 diabetes. Predictions of diabetes risk in an Indian population were accurate to the tune of 89.3 percent.

Other studies have used traditional machine learning techniques to predict diabetes risk. For instance, Al-Aidaroos and Bakar (2020) used a hybrid algorithm combining genetic algorithms and support vector machines to forecast the possibiity of diabetes in a Malaysian population. The study achieved an accuracy of 91.13% in predicting diabetes risk.

Despite the promising results of DL and machine learning techniques in predicting diabetes risk, there are still few dare that need to be labeled. One is the lack of standardized data formats and features, which can affect the performance of predictive models (Sharma et al., 2020). Another challenge is the potential bias in the data used to train predictive models, which can lead to inaccurate predictions (Raju et al., 2019).

Using DL to foresee diabetic problems has been the topic of other research. Abbas et al. (2020) employed DL to predict diabetic retinopathy. In comparison to more standard machine learning models, theirs obtained an accuracy of 97.2%. Kavakiotis et al. (2018) also employed DL for the diagnosis of diabetic nephropathy. They were able to get a better AUC-ROC (area under the receiver operating characteristic curve) for their model than any competing models.

In their paper titled "Deep learning for diabetes mellitus: A systematic review and meta-analysis," the authors perform meta-analysis of research that have employed DL for diabetes prediction and management. They examine research that has used electronic health record

(EHR) data, medical picture data, and genetic data, and evaluate the accuracy of DL models using measures including sensitivity, specificity, and area under curve (AUC). The authors conclude that DL may help enhance the precision of diabetes prediction and management, but they also note the need for additional study into the development of models that may be applied in actual clinical settings (Liu et al., 2021).

In "Diabetes mellitus prediction using deep learning models: A review", the authors review recent studies that have used DL models for diabetes prediction. They discuss the advantages of DL models, such as their ability to handle large and complex datasets, and they review several DL models that have been used for diabetes prediction, including CNNs, RNNs, and deep belief networks (DBNs). The authors conclude that DL models have the potential to improve diabetes prediction and management, but that more research is needed to address the challenges associated with using DL in clinical settings (Zhang et al., 2020).

In "Diabetes prediction using DL techniques: A review of recent developments and future prospects", the authors review recent studies that have used DL techniques for diabetes prediction. They discuss the challenges associated with diabetes prediction, such as the heterogeneity of the patient population and the need to integrate data from multiple sources, and they review several DL techniques that have been used for diabetes prediction, including autoencoders, generative adversarial networks (GANs), and transfer learning. The authors conclude that DL techniques have the potential to improve diabetes prediction and management, but that more research is needed to develop models that can be used in real-world clinical settings (Naeem et al., 2021).

One study by Gao et al. (2021) used a DL approach called the deep autoencoder to predict diabetes based on EHRs data. The deep autoencoder accomplished an area under the receiver operating characteristic curve (AUC-ROC) of 0.91, outperforming other ML models namely logistic regression and random forest.

Another study by Yu et al. (2021) used a DL approach called the Graph Convolutional Network (GCN) to predict diabetes based on genetic data. The GCN model achieved an AUC-ROC of 0.79, outperforming other established machine learning models such as Support Vector Machines (SVM) and Random Forest.

Furthermore, some studies have investigated the use of DL in predicting diabetes complications. For example, Zhu et al. (2021) used a DL approach called the Attention Residual Network (ARN) to predict diabetic nephropathy (DN) based on clinical measurements. The ARN model achieved an AUC-ROC of 0.947, outperforming other machine learning models like the Random Forest and SVM.

Additionally, some studies have investigated role of DL in predicting diabetes in specific populations. For example, Ye et al. (2021) used a DL approach called the Deep Collaborative Filtering (DCF) to predict diabetes in rural Chinese populations based on clinical measurements. The DCF model achieved an AUC-ROC of 0.908, demonstrating the potential of DL in predicting diabetes in underrepresented populations.

Dataset

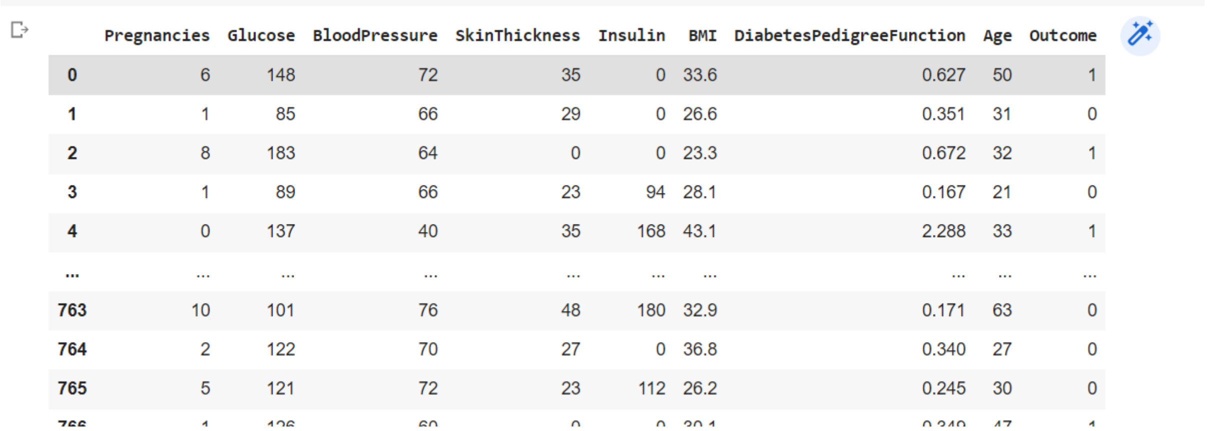
Kaggle is a popular online community of data scientists and machine learning enthusiasts that provides a platform for hosting data science competitions, building machine learning models, and collaborating with other members of the community. From the Kaggle, the publicly available dataset of diabetes was taken.

Link of the dataset - https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database?datasetId=228&searchQuery=neura

The datasets consist of a variety of medical predictor variables, also known as independent variables, and a single outcome variable, also known as a dependent variable. A patient's age, body mass index, insulin levels, and the number of pregnancies they've already had are all examples of independent variables.

The PIMA Indians Diabetes dataset is utilized in here for this task from Kaggle. Diabetes prediction is a topic that sees a lot of interest in this particular dataset. The National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK) readily provided this dataset with the raw data that was utilised in it. Using the collection's previously established diagnostic standards as a guide, the objective of the dataset is to locate instances of diabetes. In particular, every single one of these patients is a PIMA Indian woman who is at least 21 years old.

The following measurements and ranges of physical and clinical traits are included in the dataset. Pregnancies (number, [0-17]), glucose (value, [0-199]), blood pressure (mm Hg, [0-122]), skin thickness (mm, [0-99]), insulin (mu U/mL, [0-846]), BMI (kg/m2, [0-67.1]), and diabetes pedigree function (PDF), (value, [0-199]), [0.078–2.42]), age (years, [21–81]), and outcome (Boolean- 0, 1). The data are entirely numerical and comprise a total of 8 features and 768 samples. Below shows a few samples from the dataset.



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1

Figure 1 Sample of dataset

There were two columns of data with the float data type in the data, and there were seven columns with the integer data type. We did some in-depth analysis and research on the descriptive statistics of the data. A null value check was performed on the dataset, and it was discovered that the data did not contain any nulls. This was done so that the model could be trained more effectively.

The dataset was subjected to an exploratory data analysis, which resulted in the discovery of many insights.

Those whose glucose levels are higher have a greater risk of having their diabetes anticipated, however the BMI index does not appear to show any leading part in the diagnosis of diabetes in the vast majority of instances. This is demonstrated in the image below.

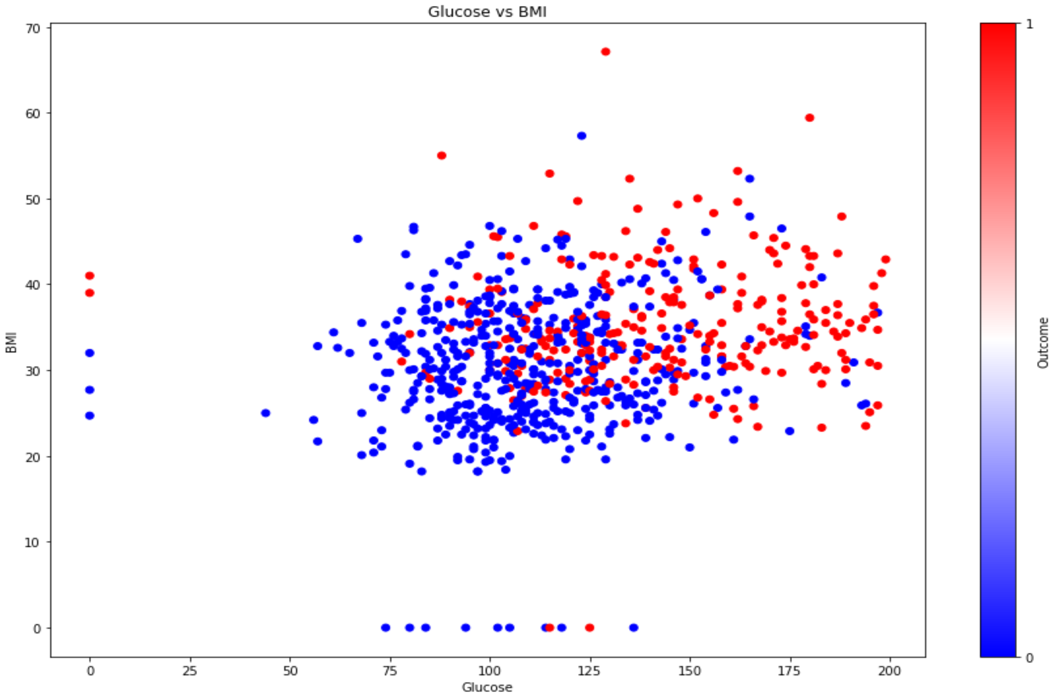


Figure 2 EDA - BMI vs Glucose impact over Diabetes

The second exploratory data analysis was done in order to visualize the distribution of the variables; some of the features were found to be skewed whereas some are normally distributed.

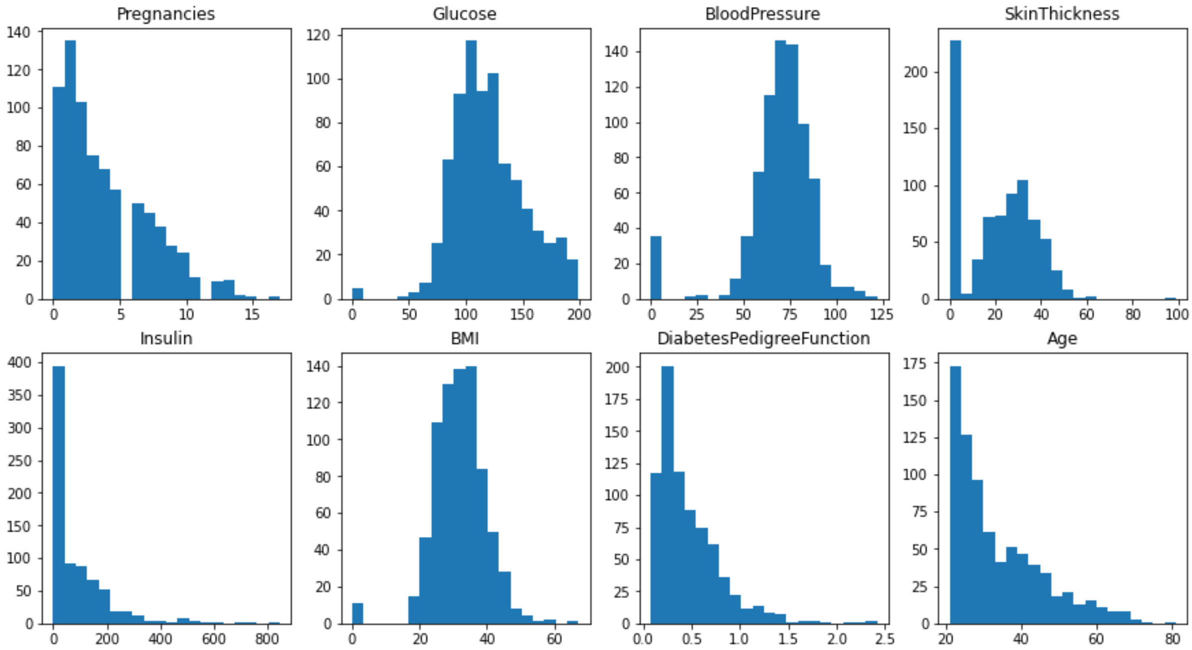


Figure 3 EDA 2 - Distribution of features

After that, for the modelling data preparation correlation between the features were analysed to avoid multi collinearity and over fitting of the model. No features were found to be multi collinear. Below figure shows the correlation matrix.

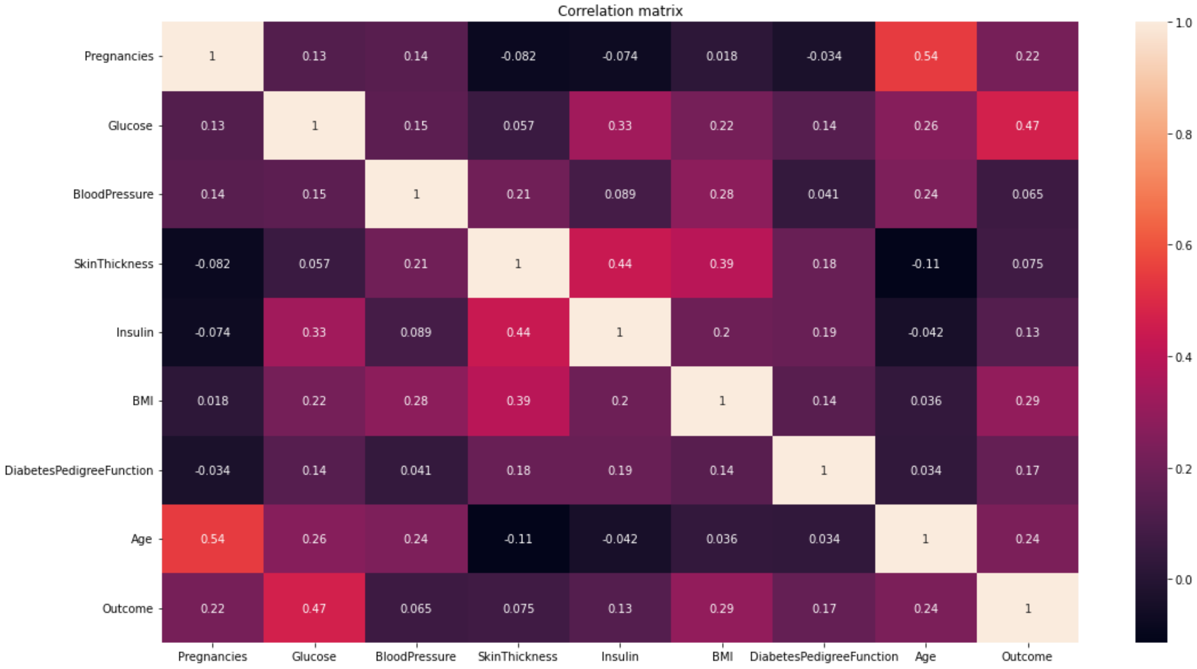


Figure 4 Correlation Matrix

Distribution of outcome columns was examined, and it was found that the dataset was imbalanced, and one category is having more samples and other class is having less samples which can result in poor performance over minority class.

For the correction of the same SMOTE was applied over the dataset to over sample the minority class with synthetic samples.

SMOTE (Synthetic Minority Over-sampling Technique) is a data augmentation technique commonly used in machine learning to tackle the difficulty of imbalanced datasets. Imbalanced datasets are datasets where one class of data is much more prevalent than another, which can result in biased or inaccurate models.

SMOTE works by creating synthetic examples of the minority class by randomly selecting a minority class data point and creating new synthetic examples along the line segments connecting the k-nearest neighbours of the selected data point. This process goes on till the desired level of oversampling is concluded. The resulting dataset will have a more balanced class distribution, which can improve the accuracy and robustness of machine learning models. SMOTE can be used in combination with other techniques such as under sampling or boosting to further improve the performance of models on imbalanced datasets. One potential limitation of SMOTE is that it may introduce synthetic examples that are too similar to existing examples, which can lead to overfitting.

After over sampling, both the classes had same number of samples (500 samples each class). Now, this data had been further taken for scaling as the features of the data are at different scale and they might lead to biased model.

Data scaling is an important pre-processing step in machine learning that involves transforming the numerical features of a dataset to a common scale. One popular method for data scaling is the standard scaler.

The standard scaler is a technique used to transform data to bear zero mean and a unit variance. This is achieved by subtracting the mean of each feature from its values and then dividing by the standard deviation. The final values will show mean to be 0 and the standard deviation as 1. This transformation can improve the performance of ML algorithms that are responsive to the scale of the input features. For this purpose, standard scaler was used, and dataset was scaled. The modelling data was then taken for training neural network.

Method

The form and operation of the biological neural network found in the human brain served as the inspiration for the development of the artificial neural network, or ANN, a type of machine learning algorithm. Because ANNs are able to discover patterns and correlations within data, they have become a popular tool for a variety of tasks involving prediction, categorization, and pattern identification (Naz & Ahuja, 2020).

A neuron or a node is the fundamental component of an artificial neural network (ANN). This component is responsible for receiving input signals from other neurons or from other sources, and then applying a mathematical function to these inputs in order to generate an output signal. The signal generated at the output of the network is then either sent to more neurons or used as the network's final output. In most cases, artificial neural networks are made up of numerous layers of neurons, which may include input, output, or hidden layers. Each layer is responsible for a distinct task, such as the collection of data or the extraction of features, and the result of one layer serves as the input for the layer that comes after it.

An artificial neural network (ANN) is made up of layers of interconnected nodes or artificial neurons. Each layer in the network processes the input from the preceding layer and produces an output that is passed on to the next layer. The three main types of layers in an ANN are the input layer, hidden layers, and output layer.

- Input Layer: The input layer is the first layer in the network, with the purpose to receive the input data and pass it on to the next layer. The number of nodes in the input layer is calculated by the number of input features in the data. For example, if the input data has 10 features, then the input layer will have 10 nodes, one for each input feature.

- Hidden Layers: Hidden layers are the layers between the input and output layers in the network. The purpose of the hidden layers is to meet changes on the input data to get higher-level features that are relevant for the task. Each node in a hidden layer gets input from all the nodes in the preceding layer and produces an output that is passed on to the next layer.

  A neural network can have one or more than one hidden layers, and the number of nodes in each layer can vary. The more layers and nodes a neural network have, the more complex the transformations it can perform on the input data.

- Output Layer: The output layer is the final layer in the network, with the purpose to produce the output for the task. The number of nodes in the output layer builds upon the type of task being performed. For example, for a binary classification task, the output layer will contain one node that produces the probability of belonging to one of the classes. For a multiclass classification task, the output layer will have multiple nodes, one for each class, and each node will produce the probability of belonging to that class. For a regression task, the output layer will contain one node that produces the predicted cost.

The process of training an ANN entail adjusting the weights and biases of the neurons in order to reduce the amount of variance that exists between the output that was anticipated and the output that actually occurred. During the training stage of supervised learning, the network is educated using a labelled dataset, which contains examples in which the appropriate output corresponds to each input. An optimization algorithm, such as backpropagation, is used by the network to update the weights and biases of the neurons. (Naz & Ahuja, 2020) This is accomplished by propagating the error that was generated in the output layer back through the network. Backpropagation is an example of an optimization algorithm.

Within the scope of this project, ANN were utilised to perform diabetes diagnosis on patients. ANN is employed because it has been demonstrated to be an algorithm that gives more accurate results than ML algorithms. In this study, two models were constructed, each with a distinct architecture of the neural networks and a unique set of parameters. This was done to better comprehend the process of model building and model training using the same training data, despite the fact that the models' structures and parameters were unique.

The dataset was separated into 80% training samples and 20% test samples and then feeded to the models.

In model 1 , model was built using 7 layers including 1 input layer, 1 output layer,2 drop out layer and 3 hidden layers with neurons 256, 64 and 32 respectively. After building the model, it was combined using adam optimizer and binary cross entropy loss.

The machine learning optimization technique ADAM (Adaptive Moment Estimation) is frequently used for stochastic gradient descent. Updates to the model parameters are calculated using gradients of the loss function relative to the parameters, making this a variant of the gradient descent algorithm.

The bounds in the ADAM method are updated using a combination of two adaptive learning rates. The initial adaptive learning rate is the exponential moving average of the gradient, which is the first moment estimate. Second moment estimate, an exponential weighted average of the square of the gradient, is the adaptive learning rate two. The update to the parameters, including a momentum term to smooth the updates, is calculated using these estimates.

All of the hidden layers made use of the relu activation function, whereas the sigmoid activation function was assigned to the output layer.

A training accuracy of 82% was achieved after fitting the built model over the trained dataset for 30 iterations.

In model 2, as the dataset was less and to avoid overfitting of the model, less nu8mber of neurons and less number of layers were experimented upon instead of making heavy network along with dropout layers. In this model along with input and output layers with sigmoid activation function, only 3 hidden layers were used having neurons 24,48,96 respectively and having activation function relu.

Adagrad optimizer was used in the compilation of Model 2. One popular optimization approach used in machine learning for stochastic gradient descent is called ADAGRAD (Adaptive Gradient Algorithm). Using past gradients, this variant of the gradient descent algorithm adjusts the learning rate for each parameter.

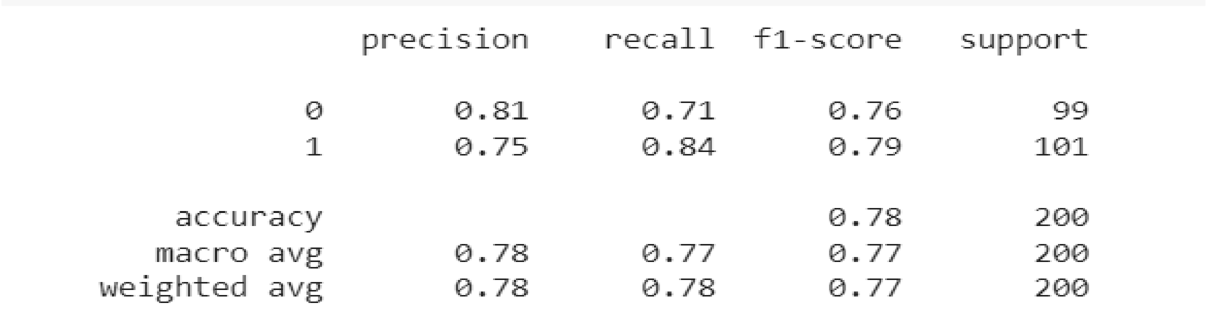
To achieve its goals, the ADAGRAD algorithm keeps track of individual learning rates for each model parameter, which are then modified according to the product of the squared gradients for that parameter. Parameters with big gradients or those that haven't been updated in a while will benefit from this method. We then used 10 epochs to fit the model to the train dataset. When compared to Model 1, this model's training accuracy was 85%.

Experimental Results

For the project, model 1was trained with more rigorous neural network along with dropout layers and model 2 without dropout layers and hidden layers with less number of neuron and less complex network to understand the change in performance of model over the test dataset if the parameters were changed.

The evaluation of both the models was done with the help of classification matrix in sklearn python.

Evaluation result of model 1:



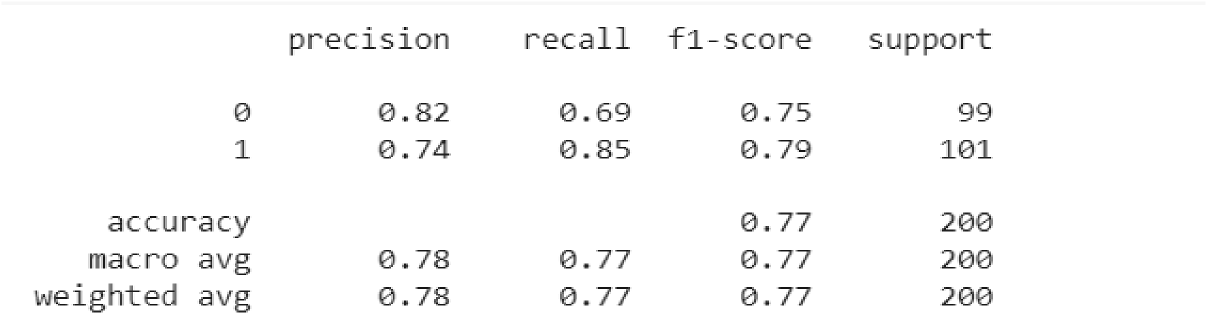
```
              precision    recall  f1-score   support

           0       0.81      0.71      0.76        99
           1       0.75      0.84      0.79       101

    accuracy                           0.78       200
   macro avg       0.78      0.77      0.77       200
weighted avg       0.78      0.78      0.77       200
```

Figure 5-Evaluation of model 1

Evaluation result of model2:



```
              precision    recall  f1-score   support

           0       0.82      0.69      0.75        99
           1       0.74      0.85      0.79       101

    accuracy                           0.77       200
   macro avg       0.78      0.77      0.77       200
weighted avg       0.78      0.77      0.77       200
```

Figure 6 - Evaluation of model 2

Model 1 with more complex network and drop out layers tend to produce more better results over test dataset which is 78% than model 2 which seems getting little overfitted even though with simplex network with accuracy of 77% over test dataset.

Discussion and Future Works

Using an artificial neural network, the purpose of this experiment was to attempt a prediction of diabetes. It is important to help people understand their chance of getting diagnosed with the same disease because it is one of the major concerns in the healthcare industry these days and is increasing year on year on an exponential scale because of continued unhealthy lifestyle choices made by people. Because of this, it is important to help people understand their chance of getting diagnosed with the same disease so that they can begin taking precautionary steps at the appropriate time and can prevent the disease because prevention is better than cure. For this reason, it is essential to have a solution that is able to determine the likelihood that an individual will get diabetes before the individual is actually diagnosed with the condition. This solution was meant to be able to collect the data of a patient and be able to estimate the risk of disease. This was accomplished with the assistance of a neural network. In this investigation, two different models were developed and then contrasted; the model that performed superiorly on the test dataset was selected to serve as the final model. The final model that was taken into consideration offered an accuracy of 78% over the test dataset.

There are several areas for future research in diabetes prediction using DL. Here are some potential avenues of investigation:

Explainable DL: One limitation of DL models is that they can be difficult to interpret, which can be a barrier to their adoption in clinical settings. Future research could focus on developing DL models that provide explanations for their predictions, so that clinicians can understand how the model arrived at a particular diagnosis or risk score.

Multi-modal data integration: DL models can leverage data from a variety of sources, including medical images, EHRs, and genetic data. Future research could investigate the use of DL models that integrate multiple types of data to improve the accuracy of diabetes prediction.

Personalized risk prediction: DL models can potentially be used to predict an individual's risk of developing diabetes based on their unique characteristics, such as age, sex, and family history. Future research could focus on developing DL models that provide personalized risk scores, which could be used to inform preventive interventions.

Real-time monitoring: DL models could be used to monitor patients with diabetes in real-time, allowing for early detection of complications and timely interventions. Future research could investigate the development of DL models that can continuously monitor patients and provide alerts when a change in risk status is detected.

Transfer learning: DL models trained on large datasets from one population may not perform well on data from other populations due to differences in demographics, genetics, and environmental factors. Transfer learning techniques could be used to adapt DL models trained on one population to other populations, improving the generalizability of the models.

References

Al-Aidaroos, M. A., & Bakar, A. A. (2020). Hybrid GA-SVM algorithm for diabetes prediction in Malaysian population. Journal of Medical Systems, 44(3), 1-10. https://doi.org/10.1007/s10916-020-1521-6

Naz, H., & Ahuja, S. (2020). Deep learning approach for diabetes prediction using PIMA Indian dataset. Journal of Diabetes & Metabolic Disorders, 19(1), 391–403. https://doi.org/10.1007/s40200-020-00520-5

Karim, F., Chowdhury, M. E. H., Tariqul Islam, S. M., Khalil, A., & Rahman, M. A. (2020). Diabetes risk prediction using deep convolutional neural network. Computers in Biology and Medicine, 126, 104042. https://doi.org/10.1016/j.compbiomed.2020.104042

Raju, S., Sattigeri, P., Nair, A., & Gupta, R. (2019). Deep learning for risk prediction of type 2 diabetes using electronic health records. IEEE Journal of Biomedical and Health Informatics, 23(1), 435-442. https://doi.org/10.1109/JBHI.2018.2820193

Sharma, A., Rani, R., & Aggarwal, A. N. (2020). A review of machine learning techniques for diabetes prediction. Journal of Medical Systems, 44(8), 1-12. https://doi.org/10.1007/s10916-020-1552-z

Wang, Y., Huang, Q., Li, L., & Huang, Y. (2020). A comprehensive evaluation of deep learning models for predicting type 2 diabetes mellitus. Journal of Diabetes Research, 2020, 1-14. https://doi.org/10.1155/2020/1352385

Abbas, Q., Fondon, I., Sarmiento, A., Akram, M. U., Khalid, S., & Garcia-Sanchez, F. (2020). A deep learning approach for diabetic retinopathy detection using retinal fundus images. Sensors, 20(1), 101. https://doi.org/10.3390/s20010101

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., & Vlahavas, I. (2018). Machine learning and data mining methods in diabetes research. Computational and Structural Biotechnology Journal, 16, 97-118. https://doi.org/10.1016/j.csbj.2018.02.002

Liu, H., Chen, M., Wang, J., Wang, S., & Zhou, W. (2021). Deep learning for diabetes mellitus: A systematic review and meta-analysis. Frontiers in Endocrinology, 12, 687189. https://doi.org/10.3389/fendo.2021.687189

Zhang, X., Wang, Y., Ma, L., & Wu, Z. (2020). Diabetes mellitus prediction using deep learning models: A review. Frontiers in Public Health, 8, 219. https://doi.org/10.3389/fpubh.2020.00219

Naeem, A., Mushtaq, M. A., Mirza, H., Majeed, A., & Khan, M. A. (2021). Diabetes prediction using deep learning techniques: A review of recent developments and future prospects. Computers in Biology and Medicine, 131, 104248. https://doi.org/10.1016/j.compbiomed.2021.104248

Gao, H., Wang, J., Jia, Y., Sun, J., & Cao, J. (2021). Deep autoencoder-based prediction of type 2 diabetes mellitus. Frontiers in Public Health, 9, 660518.

Yu, J., Wang, L., Zhang, C., & Yang, Z. (2021). Diabetes prediction from genetic data using graph convolutional network. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 18(2), 637-649.

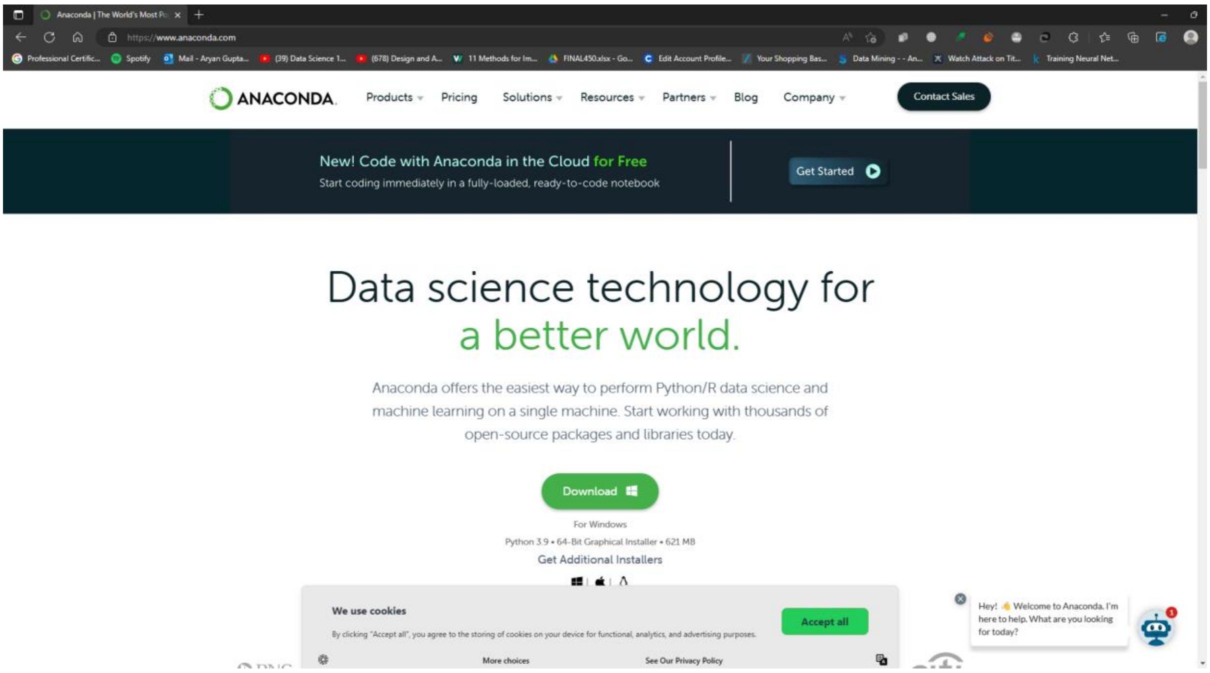
Zhu, L., Zhou, Y., Wang, X., Wang, Y., & Liao, W. (2021). Attention residual network for diabetic nephropathy prediction. Journal of Healthcare Engineering, 2021, 1-12.

Ye, S., Luo, S., Shen, L., Zhang, W., Liu, Z., & Wang, J. (2021). Diabetes prediction in rural China using deep collaborative filtering. International Journal of Environmental Research and Public Health, 18(3), 1081.

Appendix

Screenshots and Steps

Step1: Installing Anaconda which will provide environment for machine learning by visiting the first link which comes by searching anaconda on browser.



New! Code with Anaconda in the Cloud for Free

Start coding immediately in a fully-loaded, ready-to-code notebook

Data science technology for a better world.

Anaconda offers the easiest way to perform Python/R data science and machine learning on a single machine. Start working with thousands of open-source packages and libraries today.

For Windows

Python 3.9 • 64-Bit Graphical Installer • 621 MB

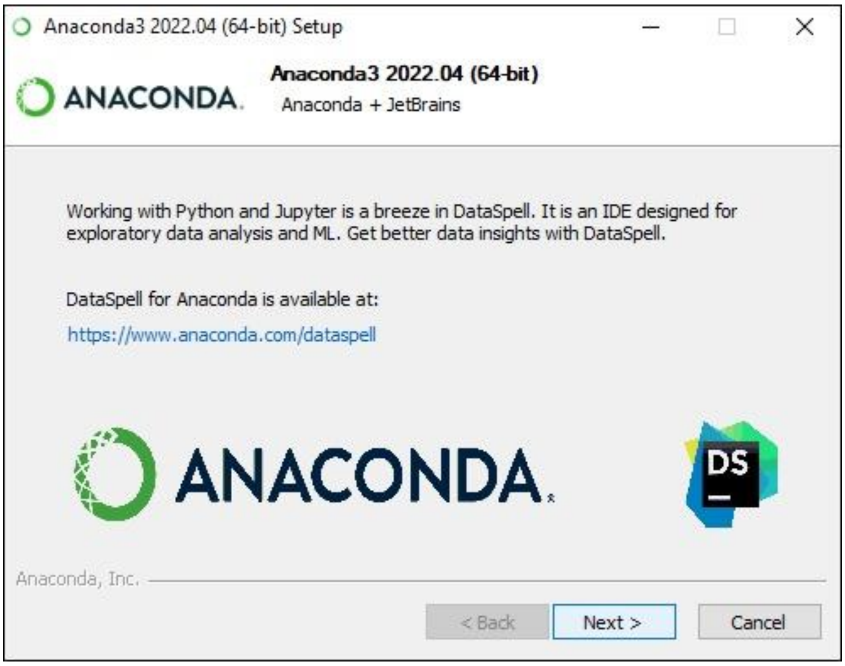
Get Additional Installers

We use cookies

By clicking "Accept all", you agree to the storing of cookies on your device for functional, analytics, and advertising purposes.

Hey! 👋 Welcome to Anaconda. I'm here to help. What are you looking for today?

Step2: After that install it and click on next.



Anaconda3 2022.04 (64-bit)

Anaconda + JetBrains

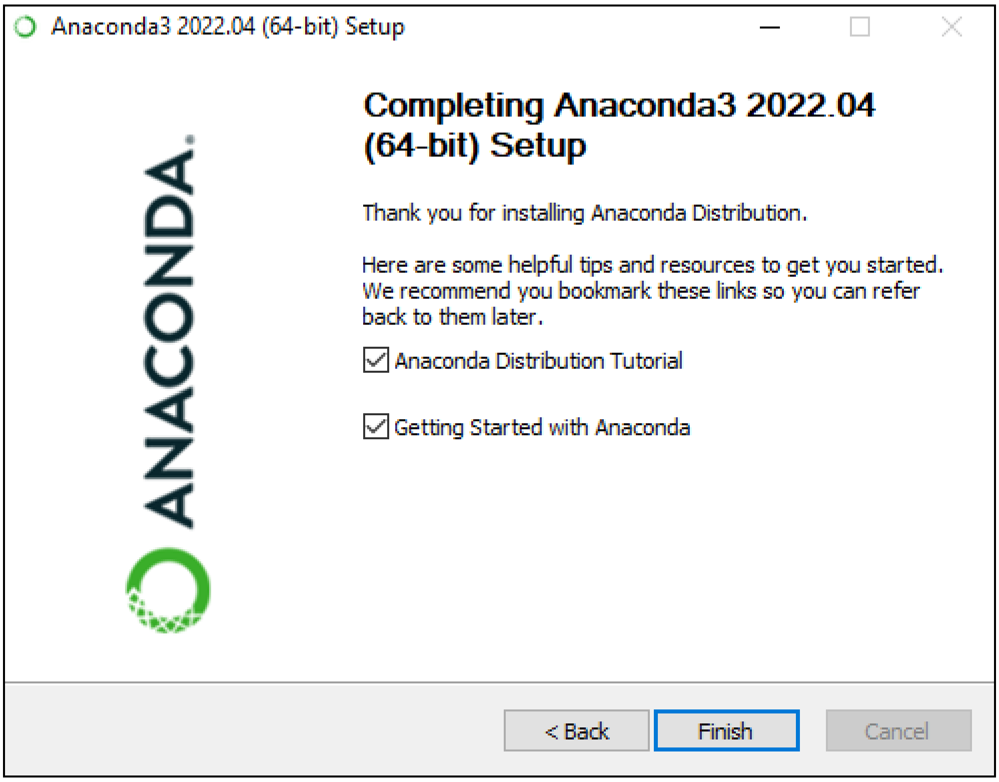
Working with Python and Jupyter is a breeze in DataSpell. It is an IDE designed for exploratory data analysis and ML. Get better data insights with DataSpell.

DataSpell for Anaconda is available at:

https://www.anaconda.com/dataspell

Anaconda, Inc.

Step3: Once installation is complete click finish



Anaconda3 2022.04 (64-bit) Setup

Completing Anaconda3 2022.04 (64-bit) Setup

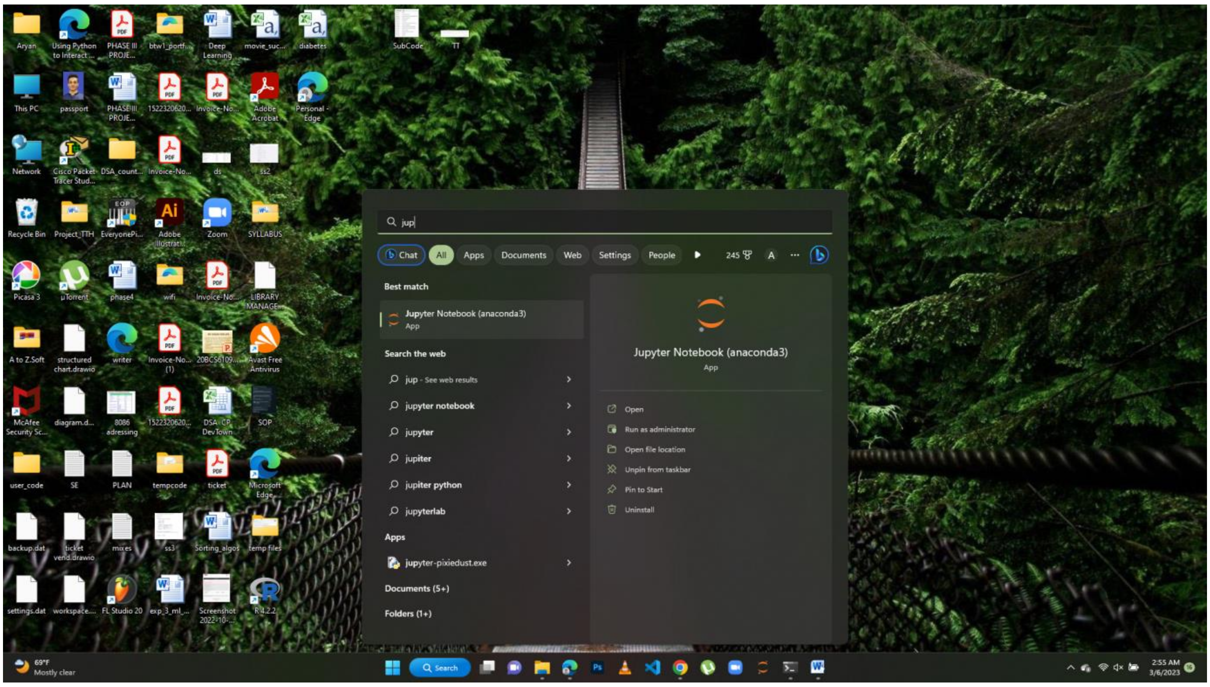
Thank you for installing Anaconda Distribution.

Here are some helpful tips and resources to get you started. We recommend you bookmark these links so you can refer back to them later.

☑ Anaconda Distribution Tutorial

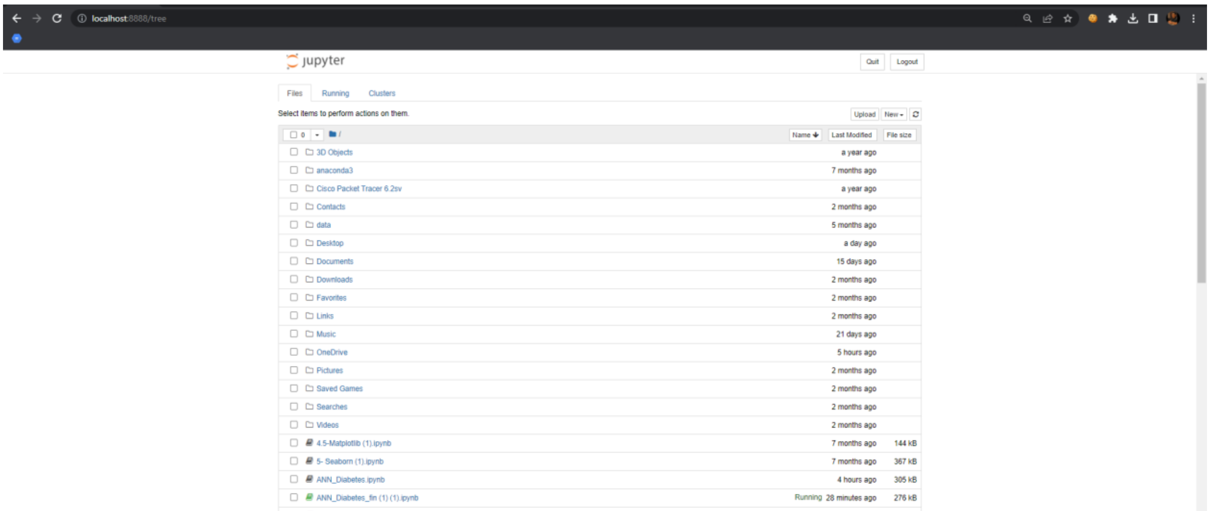
☑ Getting Started with Anaconda

Step4: Now open the start menu search for Jupyter Notebook.



jup

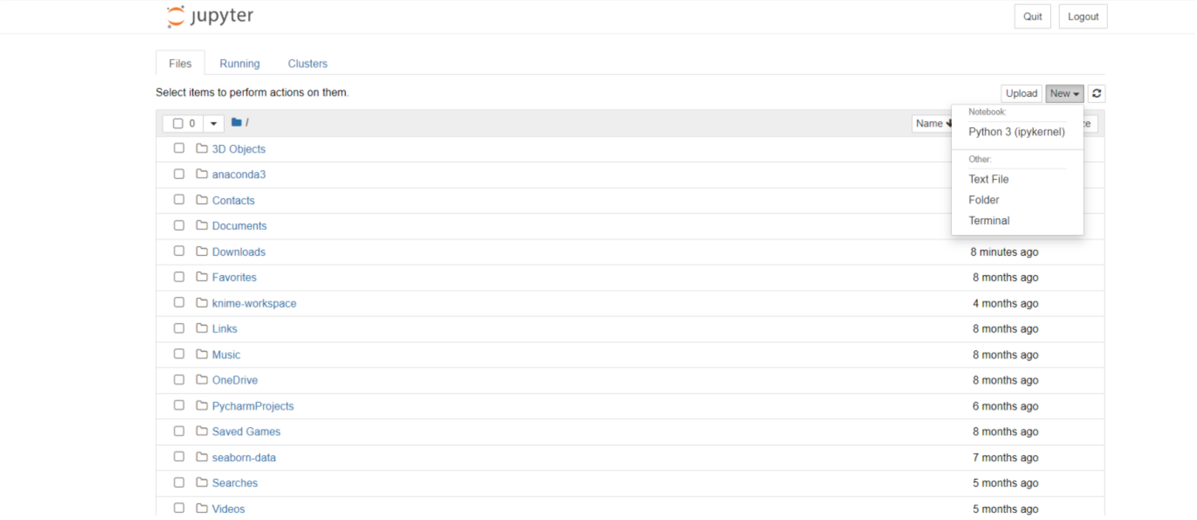
Step 5: Click on it and the main screen of Jupyter will get loaded



Select items to perform actions on them.

Name ↓		Last Modified	File size
3D Objects		a year ago	
anaconda3		7 months ago	
Cisco Packet Tracer 6.2sv		a year ago	
Contacts		2 months ago	
data		5 months ago	
Desktop		a day ago	
Documents		15 days ago	
Downloads		2 months ago	
Favorites		2 months ago	
Links		2 months ago	
Music		21 days ago	
OneDrive		5 hours ago	
Pictures		2 months ago	
Saved Games		2 months ago	
Searches		2 months ago	
Videos		2 months ago	
4.5-Matplotlib (1).ipynb		7 months ago	144 kB
5- Seaborn (1).ipynb		7 months ago	367 kB
ANN_Diabetes.ipynb		4 hours ago	305 kB
ANN_Diabetes_fin (1) (1).ipynb	Running	28 minutes ago	276 kB

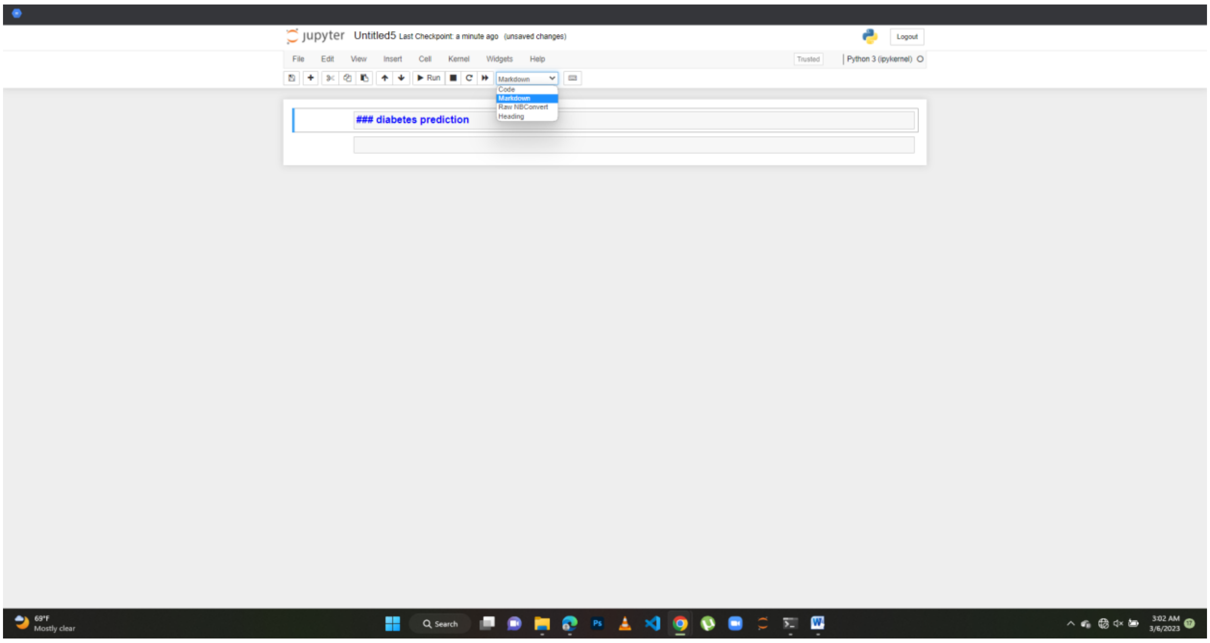
Step6: By clicking on new button create a new python notebook



Select items to perform actions on them.

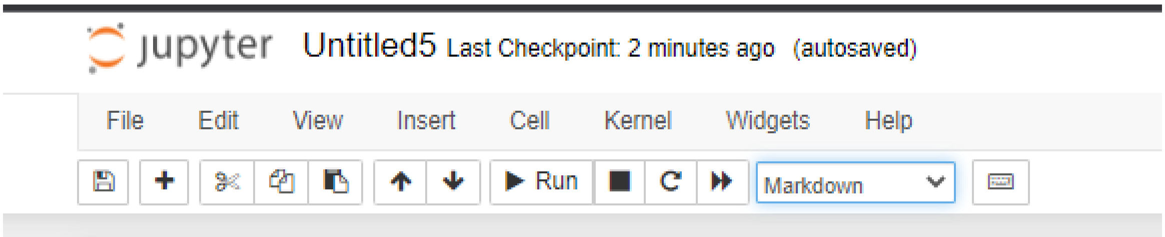
Name ↓	Last Modified
3D Objects	
anaconda3	
Contacts	
Documents	
Downloads	8 minutes ago
Favorites	8 months ago
knime-workspace	4 months ago
Links	8 months ago
Music	8 months ago
OneDrive	8 months ago
PycharmProjects	6 months ago
Saved Games	8 months ago
seaborn-data	7 months ago
Searches	5 months ago
Videos	5 months ago

Step7: Convert the cell into markdown for writing headings by adding hashtags in the beginning.


jupyter Untitled5 Last Checkpoint: a minute ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown
Code
Markdown
Raw NBConvert
Heading

diabetes prediction

Step8: To add more cells click on plus sign and to delete cells click on the scissors sign and in all the cells, write the code for the programs and run them by clicking on run button.


jupyter Untitled5 Last Checkpoint: 2 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Markdown

Step9: Files are autosaved in Jupyter notebook aur you can manually save by ctrl+s

Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

## loading the data
df= pd.read_csv('diabetes.csv')
df
```

```
df['Outcome'].value_counts()
## descriptive statistics
df.describe()

## checking null values
df.isnull().sum()

## check info of features
df.info()

## EDA
plt.figure(figsize=(15,10))
plt.scatter(df['Glucose'], df['BMI'], c=df['Outcome'], cmap='b
wr')
plt.xlabel('Glucose')
plt.ylabel('BMI')
plt.title('Glucose vs BMI')
plt.colorbar(ticks=[0, 1], label='Outcome')
plt.show()

fig, axs = plt.subplots(2, 4, figsize=(15, 8))
for i, ax in enumerate(axs.ravel()):
    ax.hist(df.iloc[:, i], bins=20)
    ax.set_title(df.columns[i])

plt.show()

# Correlation matrix
plt.figure(figsize=(18,10))
sns.heatmap(df.corr(), annot = True)
plt.title('Correlation matrix')
from imblearn.over_sampling import SMOTE
x= df.drop('Outcome', axis='columns')
y=df['Outcome']

## over sampling the dataset
smote=SMOTE(sampling_strategy = 'auto')
x_sm, y_sm = smote.fit_resample(x,y)
y_sm.value_counts()
x_sm

## scaling the dataset
from sklearn.preprocessing import StandardScaler
scaler= StandardScaler()
x_sm[x_sm.columns]= scaler.fit_transform(x_sm[x_sm.columns])

## Splitting the dataset into train and test
from sklearn.model_selection import train_test_split
```

```
x_train, x_test,y_train, y_test= train_test_split(x_sm,y_sm,te
st_size=0.2,random_state=42)
# importing more libraries
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.layers import Dropout

# Building model 1
model= Sequential()
model.add(tensorflow.keras.layers.Input(shape=8,))
model.add(tensorflow.keras.layers.Dense(256,activation='relu')
)
model.add(tensorflow.keras.layers.Dropout(0.3))
model.add(tensorflow.keras.layers.Dense(64, activation='relu')
)
model.add(tensorflow.keras.layers.Dropout(0.3))
model.add(tensorflow.keras.layers.Dense(32, activation='relu')
)
model.add(tensorflow.keras.layers.Dense(1, activation='sigmoid
'))

# Compile model 1
model.compile(optimizer='adam',loss='binary_crossentropy', met
rics= 'accuracy')

# fit model 1
model.fit(x_train,y_train, epochs=30, verbose=1)
y_pred1=model.predict(x_test)
import numpy as np
y_pred1 = np.where(y_pred1<0.5,0,1)
from sklearn import metrics
metrics.accuracy_score(y_test,y_pred1)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred1))

# Build model2
model2= Sequential()
model2.add(tensorflow.keras.layers.Input(shape=8,))
model2.add(tensorflow.keras.layers.Dense(24,activation='relu')
)
model2.add(tensorflow.keras.layers.Dense(48, activation='relu'
))
model2.add(tensorflow.keras.layers.Dense(96, activation='relu'
))
model2.add(tensorflow.keras.layers.Dense(1, activation='sigmoi
d'))
```

```
# Compile model 2
model2.compile(optimizer = 'Adagrad', loss = 'binary_crossentr
opy', metrics = ['accuracy'])
model2= model.fit(x_train, y_train, epochs=10, batch_size=32,
verbose = 1)
y_pred2=model.predict(x_test)
y_pred2 = np.where(y_pred2<0.5,0,1)
from sklearn import metrics
metrics.accuracy_score(y_test,y_pred2)
print(classification_report(y_test, y_pred2))
```