**Diabetes prediction dataset**

## About Dataset

The **Diabetes prediction dataset** is a collection of medical and demographic data from patients, along with their diabetes status (positive or negative). The data includes features such as age, gender, body mass index (BMI), hypertension, heart disease, smoking history, HbA1c level, and blood glucose level. This dataset can be used to build machine learning models to predict diabetes in patients based on their medical history and demographic information. This can be useful for healthcare professionals in identifying patients who may be at risk of developing diabetes and in developing personalized treatment plans. Additionally, the dataset can be used by researchers to explore the relationships between various medical and demographic factors and the likelihood of developing diabetes.

Code

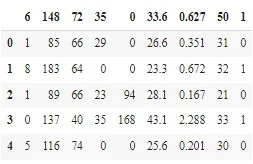
0 of 34 secondsVol which is critical in the diabetes prediction challenge.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **import** numpy as np  **import** pandas as pd  **import** rflow as tf  **from** keras.layers **import** Dense,Dropout  **from** sklearn.model\_selection **import** train\_test\_split  **import** matplotlib as mlp  **import** matplotlib.pyplot as plt  **%**matplotlib inline  **from** sklearn.preprocessing **import** StandardScaler |

### **Step 2 – Loading the Dataset**

We are now ready to begin importing the dataset. In the next piece of code, we import the dataset and use the head() method to get the top five data points.

|  |  |
| --- | --- |
| 1  2 | data**=**pd.read\_csv("pima-indians-diabetes.csv")  data.head() |

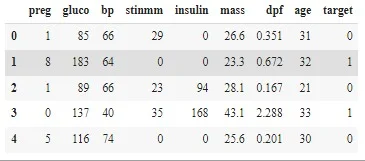
Diabetes Dataset Top5

### **Step 3 – Renaming the Columns**

You’ve probably realized that the columns are meaningless, right? Let us now rename the column names.

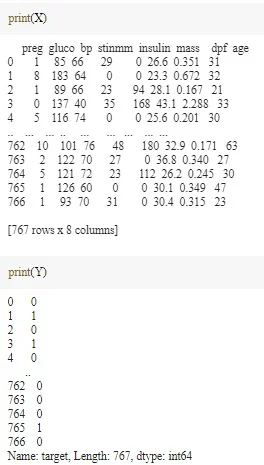
Also read: [head() in Pandas](https://www.askpython.com/python-modules/pandas/head-and-tail-of-dataframe-series)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | data **=** data.rename(index**=**str, columns**=**{"6":"preg"})  data **=** data.rename(index**=**str, columns**=**{"148":"gluco"})  data **=** data.rename(index**=**str, columns**=**{"72":"bp"})  data **=** data.rename(index**=**str, columns**=**{"35":"stinmm"})  data **=** data.rename(index**=**str, columns**=**{"0":"insulin"})  data **=** data.rename(index**=**str, columns**=**{"33.6":"mass"})  data **=**data.rename(index**=**str, columns**=**{"0.627":"dpf"})  data **=** data.rename(index**=**str, columns**=**{"50":"age"})  data **=** data.rename(index**=**str, columns**=**{"1":"target"})    data.head() |



### **Step 4 – Separating Inputs and Outputs**

|  |  |
| --- | --- |
| 1  2 | X **=** data.iloc[:, :**-**1]  Y **=** data.iloc[:,8] |

The X and Y values look somewhat like this:

Input N Output Diabetes Dataset

We separated our dataset into input and target datasets, which implies that the first eight columns will serve as input features for our model and the last column will serve as the target class.

### **Step 5 – Train-Test Split of the Data**

The next step involves the training and testing split into data and then standardizing the data to make computations simpler later on.

|  |  |  |
| --- | --- | --- |
| 1  2 | X\_train\_full, X\_test, y\_train\_full, y\_test **=** train\_test\_split(X, Y, random\_state**=**42)  X\_train, X\_valid, y\_train, y\_valid **=** train\_test\_split(X\_train\_full, y\_train\_full, random\_state**=**42) | |
| 1  2  3  4  5 | **from** sklearn.preprocessing **import** StandardScaler  scaler **=** StandardScaler()  X\_train **=** scaler.fit\_transform(X\_train)  X\_valid **=** scaler.transform(X\_valid)  X\_test **=** scaler.transform(X\_test) |

### **Step 6 – Building the Model**

We start off by using a**random seed** to generate a pseudo-random number and setting it to the **tf graph**. Then, we will be using a sequential model, and also some dropout layers in the model to avoid overfitting of the data.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | np.random.seed(42)  tf.random.set\_seed(42)    model**=**Sequential()  model.add(Dense(15,input\_dim**=**8, activation**=**'relu'))  model.add(Dense(10,activation**=**'relu'))  model.add(Dense(8,activation**=**'relu'))  model.add(Dropout(0.25))  model.add(Dense(1, act |

### **Step 7 – Training and Testing of the Model**

Now, let’s move forward to train our model and then fit the model on the testing dataset.

|  |  |
| --- | --- |
| 1  2 | model.compile(loss**=**"binary\_crossentropy", optimizer**=**"SGD", metrics**=**['accuracy'])  model\_history **=** model.fit(X\_train, y\_train, epochs**=**200, validation\_data**=**(X\_valid, y\_valid)) |

## **What Is Artificial Intelligence?**

## **Artificial intelligence (AI) is a concept that has no single, unequivocal definition. The Japanese Society provides one example of its definition for artificial intelligence, which states, “Artificial intelligence aims to accurately make advanced inferences on a large amount of data” [1]. Since there is no unequivocal definition of the term, some AI-labeled products are not used in the AI technology we mentioned in this paper. Therefore, one must be clear about what is being referenced when the term AI is used.**

## **Here, we briefly explain the concept of Strong AI and Weak AI [2]. Strong AI refers to a highly versatile AI that can establish a “consciousness” close to human thinking, make use of an appropriate program, and make comprehensive decisions. Examples to illustrate this are the Skynet from the movie The Terminator, the comic Doraemon, and C-3PO from the movie Star Wars. In contrast to Strong AI, Weak AI refers to the sort of AI specializing in a specific area or performs a specific task and does not have the sort of consciousness and ability to make comprehensive judgments like Strong AI. We may recognize Weak AI from news reports about the Chess computer program Deep Blue from IBM or the Go program AlphaGo from Google DeepMind beating professional human players. Self-driving technology and voice recognition technology, such as Siri on iPhone, are also examples of Weak AI. As of 2021, Strong AI is still in the research stage, and whenever we hear about the practical application of AI, this almost always refers to Weak AI. Weak AI technology is only weak in name, as some of their processing capacities no longer outperform human beings in many fields.onclusion**