Efraín Pérez
September 21th, 2021

Capstone Project
Machine learning Engineer Nanodegree Program

Using Machine Learning to classify if a patient has diabetes

Project overview

Three years ago, I was diagnosed with pre-diabetes at 22 years old. It was a big shock for me, because being young you never have thoughts about any kind of disease or illness. But shortly after being diagnosed, I had so much interest and enthusiasm in the field of machine learning and was always wondering, how could I use this knowledge to help people like me and not have the same scenario like me?. So, with this project I have my opportunity!.

Diabetes is in the top 10 causes of death in my country, Panama. So for that, we have to take actions against this disease and prevent any future patients.

Problem Statement

The goal is to create a binary classifier using machine learning to predict whether or not a person is diabetic. The tasks are:

1. Download the dataset from kaggle
2. Split the dataset into 80% train, 20% test
3. Choose the machine learning model

4. Test the data
5. Show the metric (accuracy)

Metrics

A common metric for binary classification is accuracy. Using accuracy is enough to classify a possible patient of diabetes

$$accuracy = \frac{true\ positive + false\ positive}{total\ size\ of\ dataset}$$

Project design

- Download the data set
- Visualize variables to get a better understanding of the data
- Split the dataset into train and test sets
- Create the neural network and machine learning model
- Print any report and compare the results

Efraín Pérez
September 21th, 2021

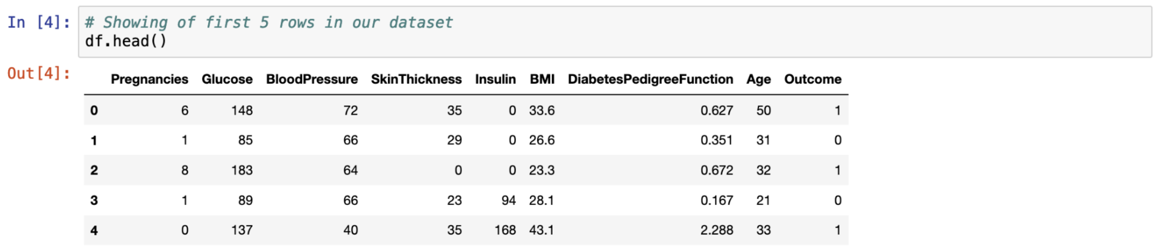
Data Exploration

Data set and inputs

We are going to use the Diabetes Kaggle dataset to achieve this project https://www.kaggle.com/uciml/pima-indians-diabetes-database. In this dataset all patients here are females at least 21 years old of Pima Indian heritage.
This data has many medical predictors as independent variables and one dependent variable.

Variables:

- **Pregnancies:** Number of times pregnant
- **Glucose:** Plasma glucose concentration 2 hours in an oral glucose tolerance test.
- **BloodPressure:** Diastolic blood pressure (mm Hg)
- **SkinThickness:** Triceps skin fold thickness (mm)
- **Insulin:** 2-Hour serum insulin (mu U/ml)
- **BMI:** Body mass index (weight in kg/(height in m)^2)
- **DiabetesPedigreeFunction**: Diabetes pedigree function
- **Outcome:** Is diabetic person or not



```
In [4]: # Showing of first 5 rows in our dataset
        df.head()
Out[4]:
```

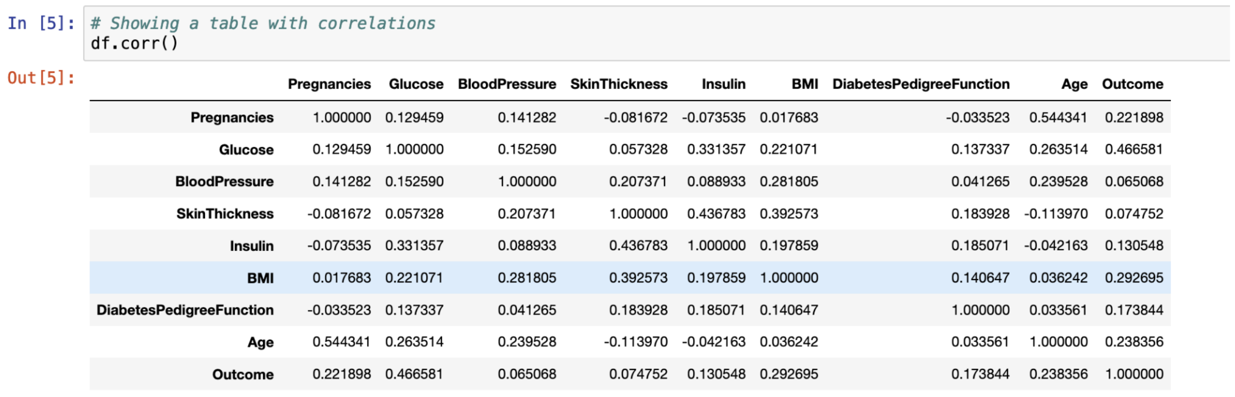
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

As we can see all features are numeric values; there is no need to make a Hot categorical transformation of the dataset.

Exploratory visualization

The plot below will show if our features have a strong correlation between them and understand if there is something to do with them.

Efraín Pérez
September 21th, 2021

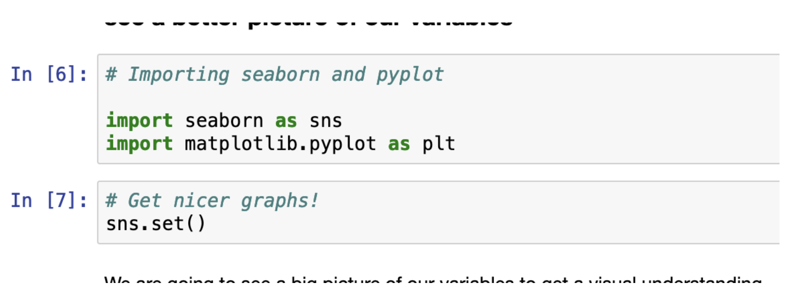


In [5]:
```
# Showing a table with correlations
df.corr()
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

As we can see the Insulin and Glucose have a strong correlation in this dataset, but i would like to see all this data in a plot. We are going to use Seaborn to plot it. With the help of the heat map we can see a better view of our correlation plot



see a better picture of our variables

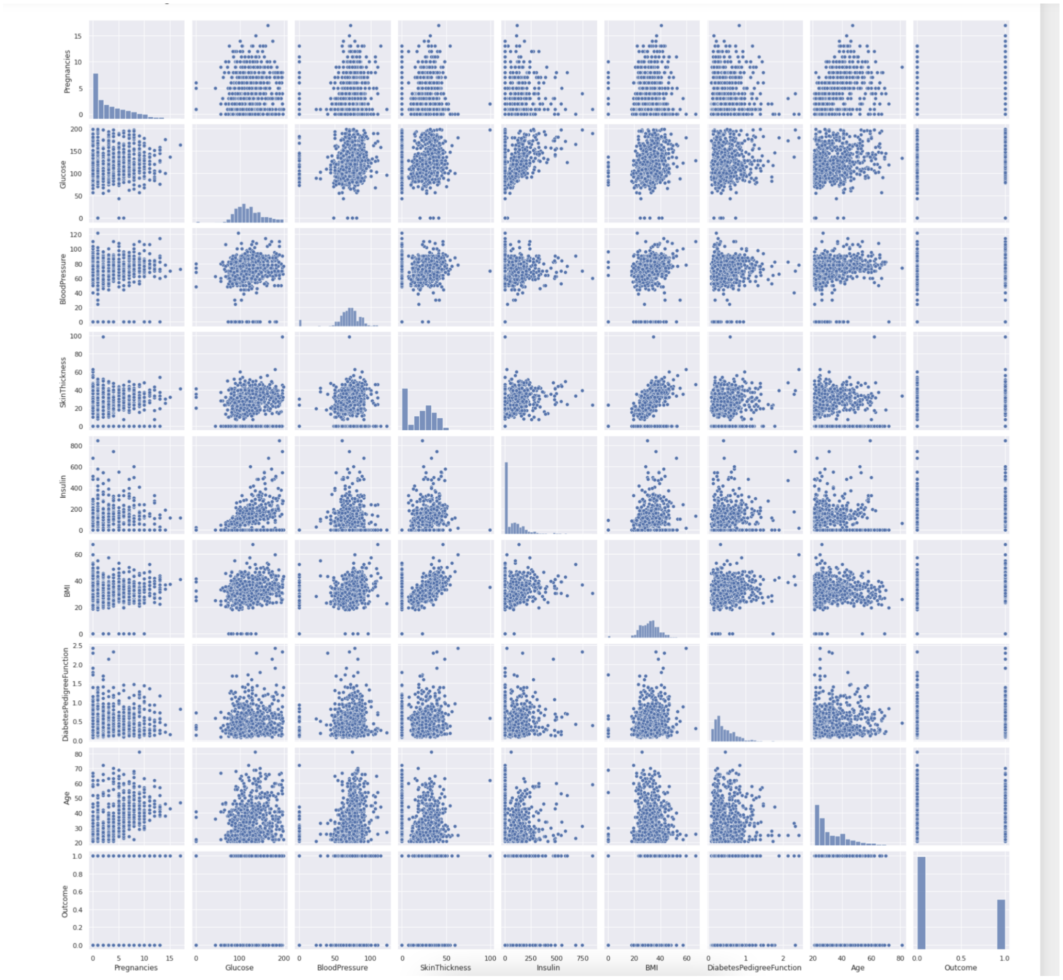
In [6]:
```
# Importing seaborn and pyplot

import seaborn as sns
import matplotlib.pyplot as plt
```

In [7]:
```
# Get nicer graphs!
sns.set()
```

We are going to see a big picture of our variables to get a visual understanding

Efraín Pérez
September 21th, 2021

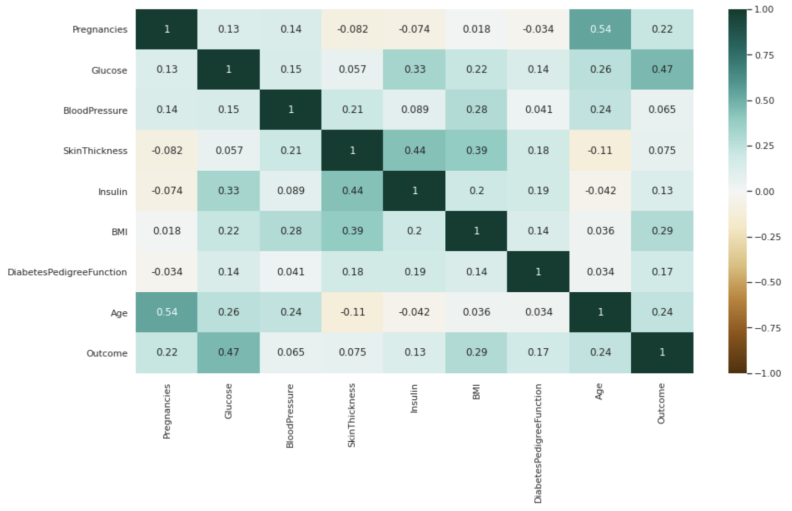


Let's see a heatmap to get a better understand of the correlation

```
In [12]: plt.subplots(figsize=(15, 8))

         sns.heatmap(df.corr(), annot=True, cmap='BrBG', vmin=-1, vmax=1)

Out[12]: <AxesSubplot:>
```



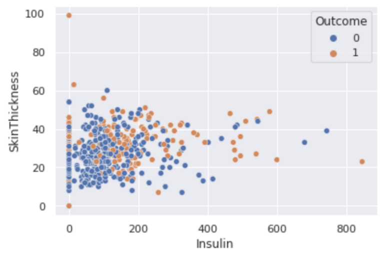
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1	0.13	0.14	-0.082	-0.074	0.018	-0.034	0.54	0.22
Glucose	0.13	1	0.15	0.057	0.33	0.22	0.14	0.26	0.47
BloodPressure	0.14	0.15	1	0.21	0.089	0.28	0.041	0.24	0.065
SkinThickness	-0.082	0.057	0.21	1	0.44	0.39	0.18	-0.11	0.075
Insulin	-0.074	0.33	0.089	0.44	1	0.2	0.19	-0.042	0.13
BMI	0.018	0.22	0.28	0.39	0.2	1	0.14	0.036	0.29
DiabetesPedigreeFunction	-0.034	0.14	0.041	0.18	0.19	0.14	1	0.034	0.17
Age	0.54	0.26	0.24	-0.11	-0.042	0.036	0.034	1	0.24
Outcome	0.22	0.47	0.065	0.075	0.13	0.29	0.17	0.24	1

Efraín Pérez
September 21th, 2021

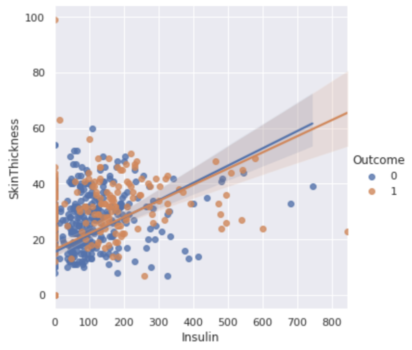
With these groups of graphs we can say:

- There is a strong relation between Skin Thickness and Insulin, a patient with a high skin thickness value will have more insulin in his body.

```
In [14]: sns.scatterplot(x='Insulin', y='SkinThickness', data=df, hue='Outcome')
Out[14]: <AxesSubplot:xlabel='Insulin', ylabel='SkinThickness'>
```

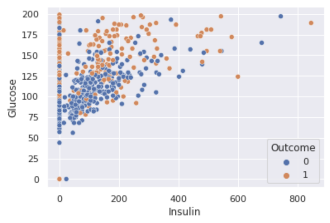


```
In [15]: sns.lmplot(x='Insulin', y='SkinThickness', data=df, hue='Outcome')
Out[15]: <seaborn.axisgrid.FacetGrid at 0x7fa332139128>
```

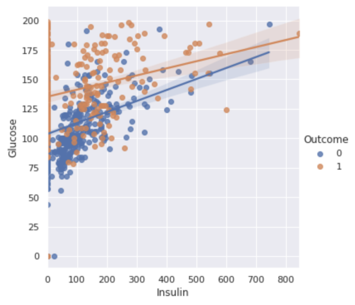


- If your levels of Glucose are high, your insulin as well.

```
In [10]: sns.scatterplot(x='Insulin', y='Glucose', data=df, hue='Outcome')
Out[10]: <AxesSubplot:xlabel='Insulin', ylabel='Glucose'>
```



```
In [11]: sns.lmplot(x='Insulin', y='Glucose', data=df, hue='Outcome')
Out[11]: <seaborn.axisgrid.FacetGrid at 0x7fa334924ba8>
```



- If the patient has a **BMI** value high, it is very probable will have diabetes

Algorithms and Techniques

The main classifier is a Neuronal Network with a loss of **binary cross entropy**, with 4 hidden layers and 35 nodes per layer and a dropout of 0.3. This Neuronal network has a strong architecture because we want to handle more data than 700 rows and more features.

With the binary cross entropy as loss we can have a result between 0 and 1 and we can predict if a patient is diabetic.

Also we use the following machine learning models to compare our results with the Neuronal Network:

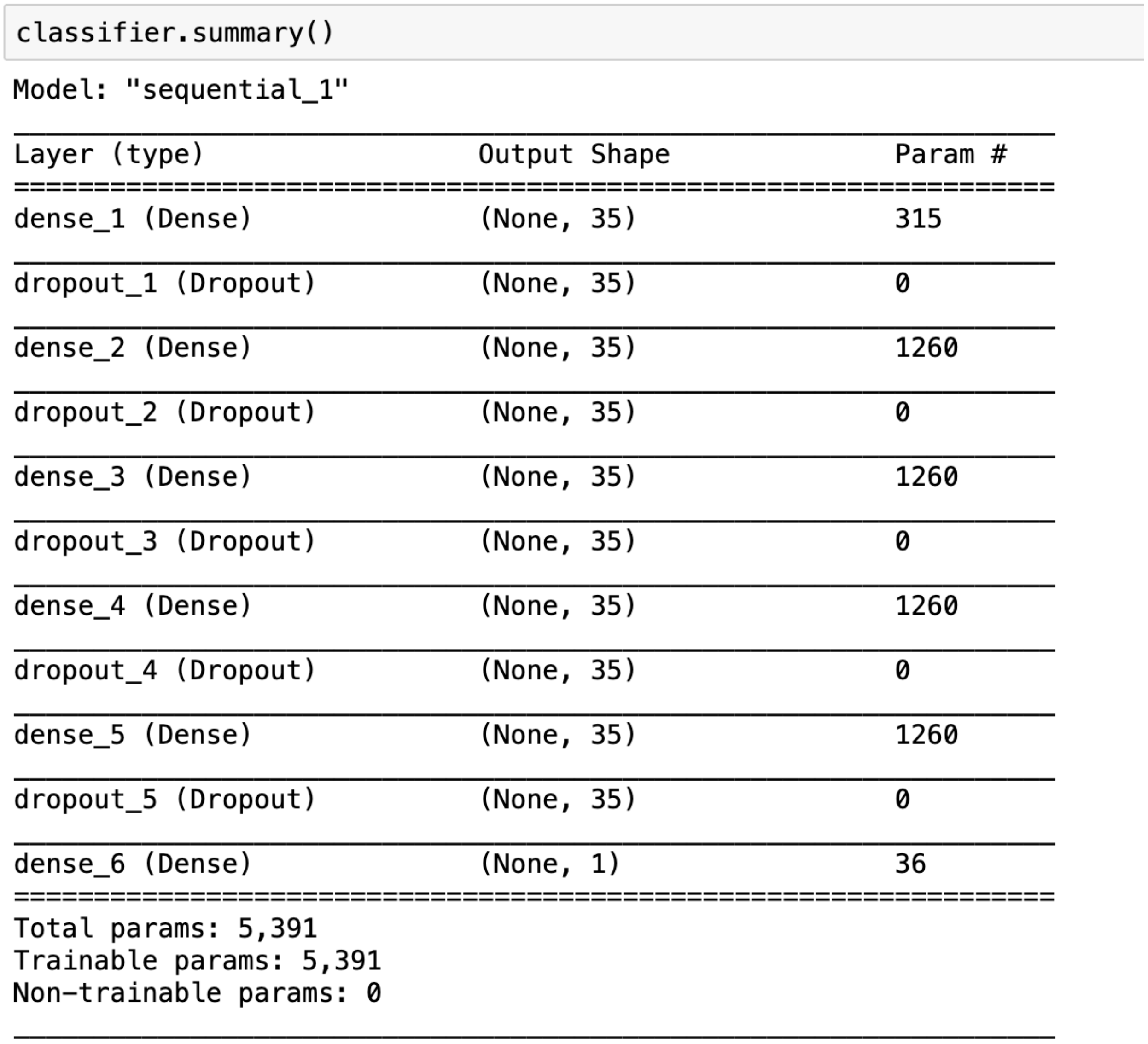
- Naive Bayes
- Decision tree
- Support Vector Machine (SVC)

Benchmark model

This is a kaggle competition, so there is no benchmark model. I am going to create a neuronal network using tensorflow and compare it with a naive Bayes model, to get a better approach treating this kind of problem.

The created Neuronal Network has these results:

Loss	Accuracy	Precision	Recall
0.3237	0.8876	0.8114	0.6958



```
classifier.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 35)                315
_________________________________________________________________
dropout_1 (Dropout)          (None, 35)                0
_________________________________________________________________
dense_2 (Dense)              (None, 35)                1260
_________________________________________________________________
dropout_2 (Dropout)          (None, 35)                0
_________________________________________________________________
dense_3 (Dense)              (None, 35)                1260
_________________________________________________________________
dropout_3 (Dropout)          (None, 35)                0
_________________________________________________________________
dense_4 (Dense)              (None, 35)                1260
_________________________________________________________________
dropout_4 (Dropout)          (None, 35)                0
_________________________________________________________________
dense_5 (Dense)              (None, 35)                1260
_________________________________________________________________
dropout_5 (Dropout)          (None, 35)                0
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 36
=================================================================
Total params: 5,391
Trainable params: 5,391
Non-trainable params: 0
_________________________________________________________________
```

In the result section we will see the compare results for all the models i trained.

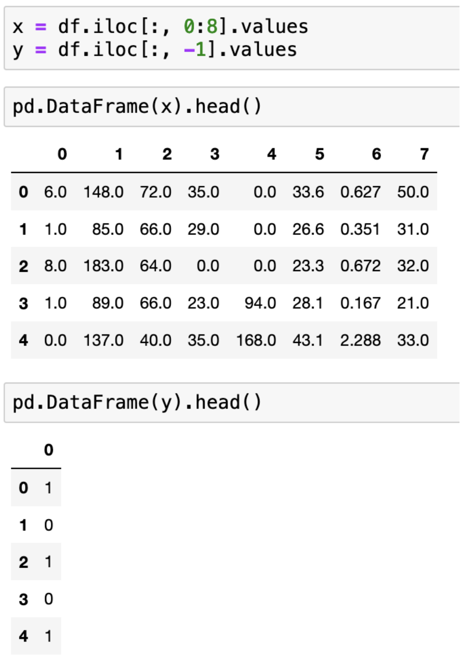
Efraín Pérez
September 21th, 2021

Methodology

Data preprocessing

The preprocessing data is done in the notebook file as follows:

- Take the features columns as our **x** value *(from column 0 to 8) and **y** value as the last column



```
x = df.iloc[:, 0:8].values
y = df.iloc[:, -1].values
```

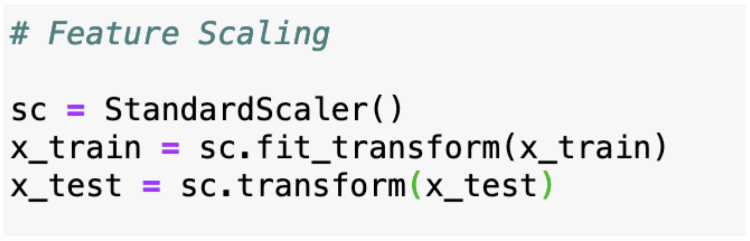
```
pd.DataFrame(x).head()
```

	0	1	2	3	4	5	6	7
0	6.0	148.0	72.0	35.0	0.0	33.6	0.627	50.0
1	1.0	85.0	66.0	29.0	0.0	26.6	0.351	31.0
2	8.0	183.0	64.0	0.0	0.0	23.3	0.672	32.0
3	1.0	89.0	66.0	23.0	94.0	28.1	0.167	21.0
4	0.0	137.0	40.0	35.0	168.0	43.1	2.288	33.0

```
pd.DataFrame(y).head()
```

	0
0	1
1	0
2	1
3	0
4	1

- Split our dataset into two sets. Train set as 80% and test set as 20%
- Scale our dataset before training



```
# Feature Scaling

sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

Efraín Pérez
September 21th, 2021

Implementation

Using the tensorflow library, we created a Neuronal network to classify our dataset, using binary cross entropy.

Using the following configuration:

Input layer Nodes	Hidden layers	Nodes per hidden layer	Activation function of hidden layers	Activation function of output layer	Dropout	Batch size	epochs
8	4	35	relu	sigmoid	0.35	10	1000

Refinement

At the beginning preparing our dataset, I did not scale our dataset thinking the data has no significant range of values or NN had an accuracy of 0.86 then i scaled an

Efraín Pérez
September 21th, 2021

References

- MINSA. (2018). *LA DIABETES, SEXTA CAUSA DE MUERTE EN PANAMÁ*. MINSA. http://www.css.gob.pa/web/6-julio-2018au.html
- *Relationship of Skin Thickness to Duration of Diabetes, Glycemic Control, and Diabetic Complications in Male IDDM Patients*. (1989). American Diabetes Association. https://care.diabetesjournals.org/content/12/5/309