ML-MINOR-MAY

We'll be using Machine Learning to predict whether the patient has diabetes or not, based on information about the patient such as blood pressure, body mass index (BMI), age, etc.

In particular, the tutorial has the following sections:

- Overview
- Data Description
- Data Exploration
- Data Preparation
- Training and Evaluating the Machine Learning Model
- Making Predictions with the Model
- Next Steps

Overview:

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

We'll be using Python and some of its popular data science related packages. First of all, we will import pandas to read our data from a CSV file and manipulate it for further use. We will also use numpy to convert out data into a format suitable to feed our classification model. We will then import Logistic Regression algorithm from sklearn This algorithm will help us build our classification model.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

Data Description:

We have our data saved in a CSV file called diabetes.csv. We first read our dataset into a pandas dataframe called df. And then print the dataframe.

	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
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

The following features have been provided to help us predict whether a person is diabetic or not:

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration over 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 (a function which scores likelihood of diabetes based on family history)
- Age: Age (years)
- Outcome: Class variable (0 if non-diabetic, 1 if diabetic)

Let’s also make sure that our data is clean (has no null values, etc).

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

There are some 0 values in the dataframe in column “Insulin”.

```
df
```

	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
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

So we will first replace it with NaN, the from NaN to mean of that column.

Since the value of mean will come in float and insulin can not be in flaot value so we will convert float into int.

```
df.loc[df['Insulin']==0,'Insulin']=np.nan
df
```

```
df=df.fillna(df['Insulin'].mean())
df['Insulin']=df['Insulin'].astype(int)
df
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	155	33.6	0.627	50	1
1	1	85	66	29	155	26.6	0.351	31	0
2	8	183	64	0	155	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	155	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	155	30.1	0.349	47	1

✓ 0s completed at 3:45 AM

Data Exploration:

Let us now explore our data set

Let’s start by finding correlation of every pair of features (and the outcome variable), and visualize the correlations using a heatmap.

Dataset Preparation (splitting and normalization):
When using machine learning algorithms we should always split our data into a training set and test set. (If the number of experiments we are running is large, then we should be dividing our data into 2 parts, namely — training set and test set)

The data set consists of record of 767 patients in total. To train our model we will be using 70% of records and for testing we will be using 30% of records.

```
train_x, test_x, train_y, test_y=train_test_split(x,y,test_size=0.3)
```

Now we will the standardize the input, i.e.train_x and test_x, so that about 68% of values will lie in range -1 to +1.

```
sc=StandardScaler()

train_x=sc.fit_transform(train_x)
test_x=sc.fit_transform(test_x)
```

Training and Evaluating Machine Learning Model:

We can now train our classification model. We'll be using a machine simple learning model called logistic regression. Since the model is readily available in sklearn, the training process is quite easy and we can do it in few lines of code. First, we create an instance called clr and then use the fit function to train the model.

```
[722] clr=LogisticRegression()

clr.fit(train_x,train_y)
```

Making Predictions with Model:

Now when the model is trained we will predict that the patient is diabetic or not. First, we create an instance called pv_y and then use the predict function to test the model.

```
pv_y=clr.predict(test_x)
pv_y
```

Now, to get accuracy score, we will do following code:

```
[777] from sklearn.metrics import confusion_matrix, accuracy_score

[780] accuracy_score(test_y,pv_y)

confusion_matrix(test_y,pv_y)
```

BY,

DIVOLIKA BAJPAI