Lab 5 - Multiple Linear Regression Diabetes Dataset

Author: Krishna Swaroop

181CO125, NITK Surathkal

Introduction

Linear Regression is a machine learning algorithm based on supervised learning. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting.

Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

Dataset

We are using the Pima Indians Diabetes Dataset. 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.

The datasets consists of several medical predictor variables and one target variable, Outcome. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

Multiple Linear Regression

1) Import libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

2) Import data

```
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ML-Lab/Multi Linear Regression/diabetes.csv')
```

```
print(f"Columns of dataset: {df.columns}")
```

```
Columns of dataset: Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
print(df.head())
```

```
   Pregnancies  Glucose  BloodPressure  ...  DiabetesPedigreeFunction  Age  Outcome
0            6      148             72  ...                     0.627   50        1
1            1       85             66  ...                     0.351   31        0
2            8      183             64  ...                     0.672   32        1
3            1       89             66  ...                     0.167   21        0
4            0      137             40  ...                     2.288   33        1

[5 rows x 9 columns]
```

```
X = np.asanyarray(df[['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin','BMI', 'DiabetesPedigreeFuncti
Y = np.asanyarray(df['Outcome'])
```

3) Split the data

Use `train_test_split()` to split the data to training and testing dataset. Here, 25% of the dataset is reserved to test our algorithm

```
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=0.25,random_state=42)
```

4) Fit the model

```
lr = LinearRegression()
```

```
lr.fit(X_train, y_train)
```

```
    LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

5) Accuracy and other measures

```
temp = lr.predict(X_test)
y_pred = []
for elm in temp:
  if elm < 0.5:
    y_pred.append(0)
  else:
    y_pred.append(1)

y_pred = np.asarray(y_pred)
```

```
from sklearn.metrics import confusion_matrix
mx = confusion_matrix(y_test, y_pred)
```

```
print(mx)
```

```
[[96 27]
 [26 43]]
```

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
print("Accuracy:",accuracy_score(y_test, y_pred))
print("Precision:",precision_score(y_test, y_pred))
print("Recall:",recall_score(y_test, y_pred))
```

```
Accuracy: 0.7239583333333334
Precision: 0.6142857142857143
Recall: 0.6231884057971014
```