Diabetes Prediction

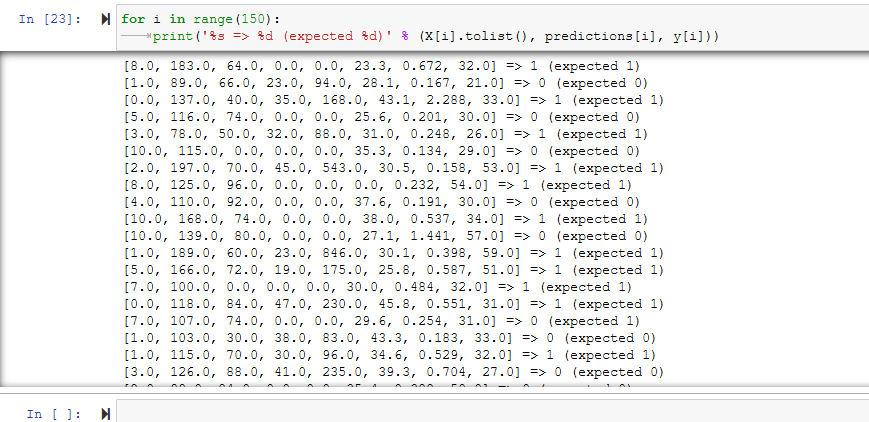
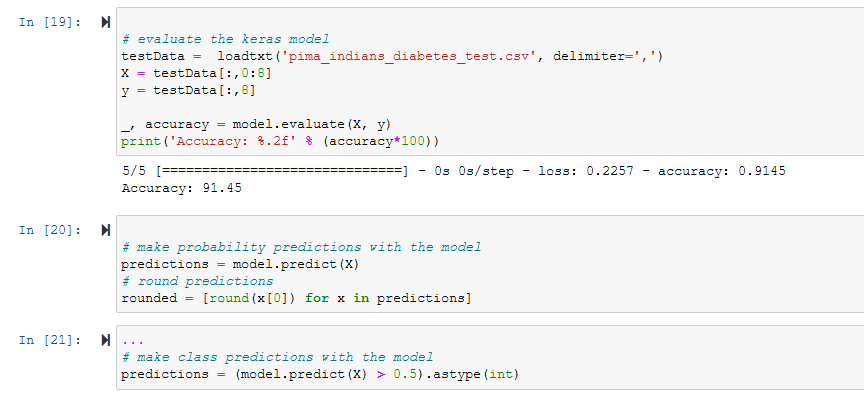
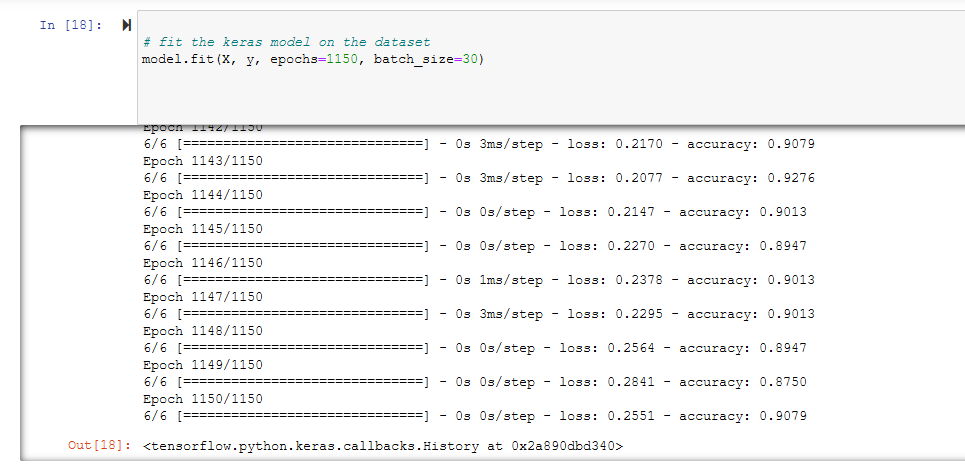
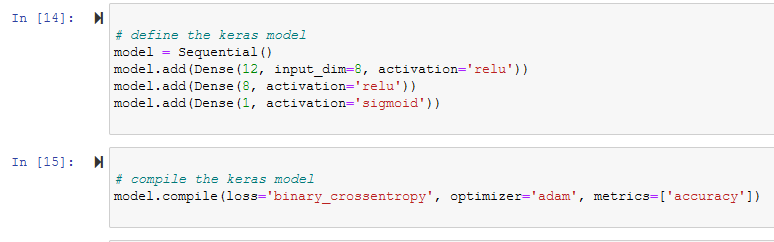
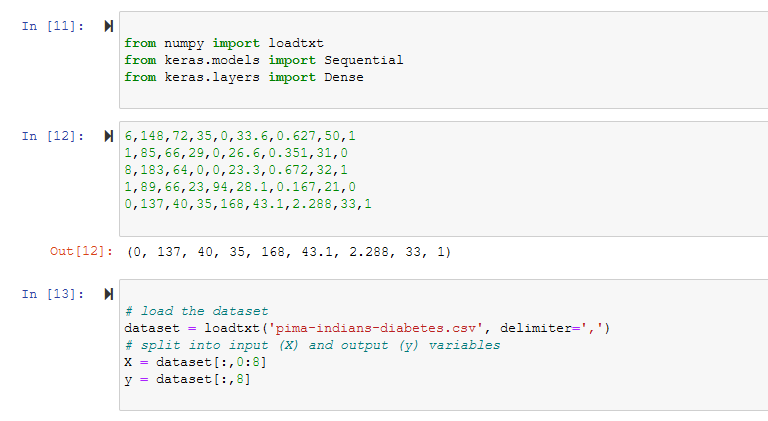
Summary

We are going to train a model that will help to predict diabetes in person based on some information about that person. We will use “pima-indians-diabetes” dataset to train our model. And based on the learning from the dataset we will use our model to predict diabetes.

Dataset description

The columns of the dataset contain the following values and on these values our model will be trained. We are working with deep learning so the feature extractions all the underlying work will be done by our model. We will not explicitly extract any feature.

1. Number of times pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/(height in m)^2)
7. Diabetes pedigree function
8. Age (years)



from numpy import loadtxt

from keras.models import Sequential

from keras.layers import Dense

6,148,72,35,0,33.6,0.627,50,1

1,85,66,29,0,26.6,0.351,31,0

8,183,64,0,0,23.3,0.672,32,1

1,89,66,23,94,28.1,0.167,21,0

0,137,40,35,168,43.1,2.288,33,1

# load the dataset

dataset = loadtxt('pima-indians-diabetes\_train.csv', delimiter=',')

# split into input (X) and output (y) variables

X = dataset[:,0:8]

y = dataset[:,8]

print(X[1])

print(y[5])

# define the keras model

model = Sequential()

model.add(Dense(12, input\_dim=8, activation='relu'))

model.add(Dense(4, activation='relu'))

model.add(Dense(8, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

# compile the keras model

model.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy'])

# fit the keras model on the dataset

model.fit(X, y, epochs=1150, batch\_size=30)

# evaluate the keras model

testData =  loadtxt('pima\_indians\_diabetes\_test.csv', delimiter=',')

X = testData[:,0:8]

y = testData[:,8]

\_, accuracy = model.evaluate(X, y)

print('Accuracy: %.2f' % (accuracy\*100))

# make probability predictions with the model

predictions = model.predict(X)

# round predictions

rounded = [round(x[0]) for x in predictions]

...

# make class predictions with the model

predictions = (model.predict(X) > 0.5).astype(int)

for i in range(150):

    print('%s => %d (expected %d)' % (X[i].tolist(), predictions[i], y[i]))

**Libraries Used**

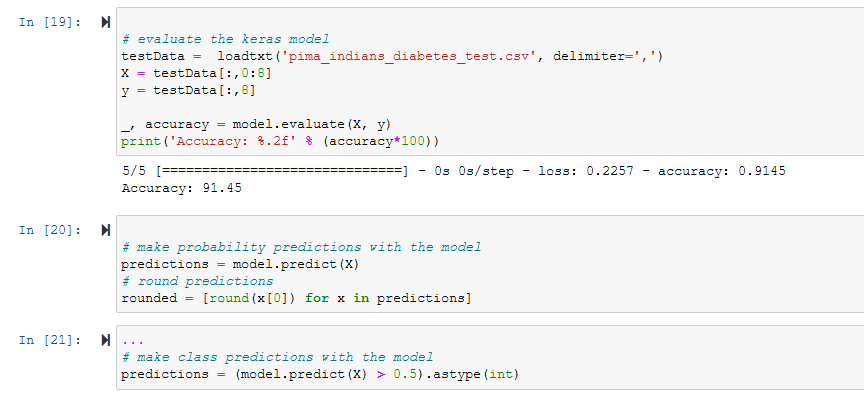
Numpy for numerical data processing. We are using numerical data so numpy is excellent and fast library for this purpose. It stands for numerical python and is written in c++ that’s why it is very fast

Keras for deep learning modeling. Keras is very popular deep learning library used for building models it runs on two backends theano or tensorflow.

keras.layers for adding layers in model

**Results**

After training our model for several times we get an accuracy of 91 percent.



**Conclusion / Future work**

Our model can be used to predict the diabetes in a patient with an accuracy of 91 percent.

In future we can improve our accuracy by running it on a powerful system.