Journal of Management & Entrepreneurship
ISSN 2229-5348

UGC Care Group I Journal
Vol-7 Issue-01 2018

PYTHON KERAS, BIG DATA, AND DEEP LEARNING TO PREDICT PATIENT DIABETES AND EMPLOYEE'S WAGES PER HOUR

[1]Anishetty Shiva Rama Krishna,Associate Professor,
[2]Sravani M ,Assistant Professor,
[3]Divya Ch,Assistant Professor,
Department of CSE Engineering,
Nagole Institute of Technology and Science,,
Kuntloor(V),Hayathnagar(M),Hyderabad,R.R.Dist.-501505.

Abstract: Big Data Analytics and Deep Learning are not supposed to be two entirely different concepts. Big Data means extremely huge large data sets that can be analyzed to find patterns, trends. One technique that can be used for data analysis so that able to help us find abstract patterns in Big Data is Deep Learning. If we apply Deep Learning to Big Data, we can find unknown and useful patterns that were impossible so far. These techniques are currently a much active area of research in medical science. With increasing size and complexity of medical data like X-rays, deep learning gained huge success in prediction of many diseases like pneumonia, diabetes. In this paper, we proposed two deep learning models using Keras and also we will build a regression model to predict an employee's wage per hour, and we will build a classification model to predict whether or not a patient has diabetes.

Index terms: Deep Learning, Bigdata.

I. INTRODUCTION

In simple word, the term of Big Data means collecting, processing and presenting the results of huge amounts of data that comes at high speed in a variety of formats. Traditional Machine Learning tools have shortcoming when they face with Big Data and want to solve Big Data area problems.

Big Data and deep learning are two important words in data science now days. Big Data Analytics and Deep Learning are two high-focus of data science. The large volumes of data collected by organizations are utilized for various purposes such as for solving problems in marketing, technology, medical science, national intelligence, fraud detection etc. Traditional data processing systems are not adequate to handle, analyze and process as the collected data are unlabelled, uncategorized and very complex.

Deep learning is an increasingly popular subset of machine learning. Deep learning models are built using neural networks. A neural network takes in inputs, which are then processed in hidden layers using weights that are adjusted during training. Then the model spits out a prediction. The weights are adjusted to find patterns in order to make better predictions. The user does not need to specify what patterns to look for — the neural network learns on its own. Deep learning is appropriate for exploiting large volumes of data and for analyzing raw data from multiple sources and in different styles.

II. REVIEW OF LITERATURE

A. BIGDATA:

While the term "big data" is relatively new, the act of gathering and storing large amounts of information for eventual analysis is ages old. The concept gained momentum in the early 2000s when industry analyst Doug Laney articulated the now-mainstream definition of big data as the three Vs.

B. DEEP LEARNING:

Deep learning is an aspect of artificial intelligence (AI) that is concerned with emulating the learning approach that human beings use to gain certain types of knowledge. At its simplest, deep learning can be thought of as a way to automate predictive analytics Share .Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. Yoshua Bengio is another leader in deep learning although began with a strong interest in the automatic feature learning that large neural networks are capable of achieving. He describes deep learning in terms of the algorithms ability to discover and learn good representations using feature learning."Deep learning algorithms seek to exploit the unknown structure in the input distribution in order to discover good representations, often at multiple levels, with higher-level learned features defined in terms of lower-level features".

C. KERAS:

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and Tensor Flow and allows you to define and train neural network models in just a few lines of code. Use Keras if you need a deep learning library that: 1.Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility) 2.Supports convolution networks and recurrent networks, as well as combinations of the two. 3. Runs seamlessly on CPU and GPU. It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks.

Build Deep Learning Models with Keras:

The focus of Keras is the idea of a model. The main type of model is called a Sequence which is a linear stack of layers. we create a sequence and add layers to it in the order that you wish for the computation to be performed. Once defined, you compile the model which makes use of the underlying framework to optimize the computation to be performed by your model. In this you can specify the loss function and the optimizer to be used. Once compiled, the model must be fit to data. This can be done one batch of data at a time or by firing off the entire model training regime. This is where all the compute happens. Once trained, you can use your model to make predictions on new data.

We can summarize the construction of deep learning models in Keras as follows:

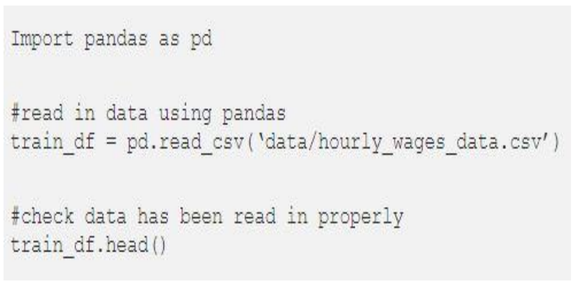
1. Define your model. Create a sequence and add layers.
2. Compile your model. Specify loss functions and optimizers.
3. Fit your model. Execute the model using data.
4. Make predictions. Use the model to generate predictions on new data.

III. PROPOSED WORK:

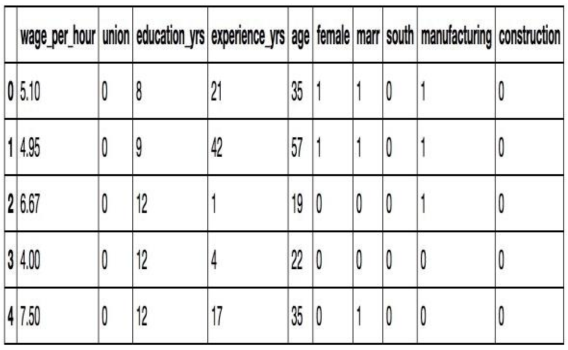
Keras is a user-friendly neural network library written in Python. We will propose two deep learning models using Keras: one for regression and one for classification. We will build a regression model to predict an employee's wage per hour, and we will build a classification model to predict whether or not a patient has diabetes. The datasets we will be using are relatively clean, so we will not perform any data preprocessing in order to get our data ready for modeling. Datasets that you will use in future projects may not be so clean — for example, they may have missing values — so you may need to use data preprocessing techniques to alter your datasets to get more accurate results.

A. Reading in the training data

For our regression deep learning model, the first step is to read in the data we will use as input. For this example, we are using the 'hourly wages' dataset



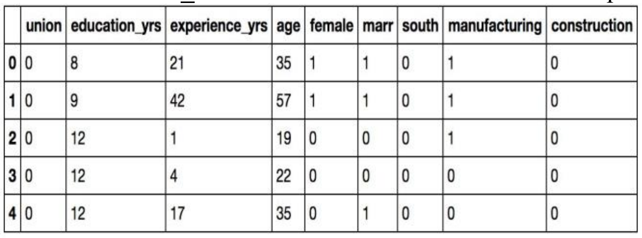
```
Import pandas as pd

#read in data using pandas
train_df = pd.read_csv('data/hourly_wages_data.csv')

#check data has been read in properly
train_df.head()
```



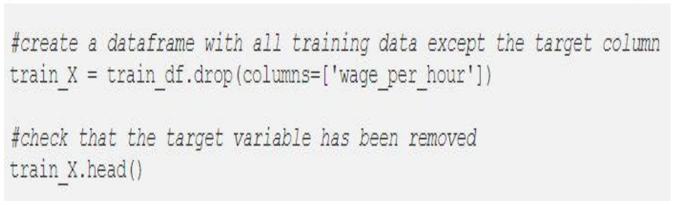
	wage_per_hour	union	education_yrs	experience_yrs	age	female	marr	south	manufacturing	construction
0	5.10	0	8	21	35	1	1	0	1	0
1	4.95	0	9	42	57	1	1	0	1	0
2	6.67	0	12	1	19	0	0	0	1	0
3	4.00	0	12	4	22	0	0	0	0	0
4	7.50	0	12	17	35	0	1	0	0	0

B. Split up the dataset into inputs and targets

Next, we need to split up our dataset into inputs (train_X) and our target (train_y). Our input will be every column except 'wage_per_hour' because 'wage_per_hour' is what we will be attempting to predict. Therefore, 'wage_per_hour' will be our target. We will use pandas 'drop' function to drop the column 'wage_per_hour' from our data frame and store it in the variable 'train_X'. This will be our input.



	union	education_yrs	experience_yrs	age	female	marr	south	manufacturing	construction
0	0	8	21	35	1	1	0	1	0
1	0	9	42	57	1	1	0	1	0
2	0	12	1	19	0	0	0	1	0
3	0	12	4	22	0	0	0	0	0
4	0	12	17	35	0	1	0	0	0



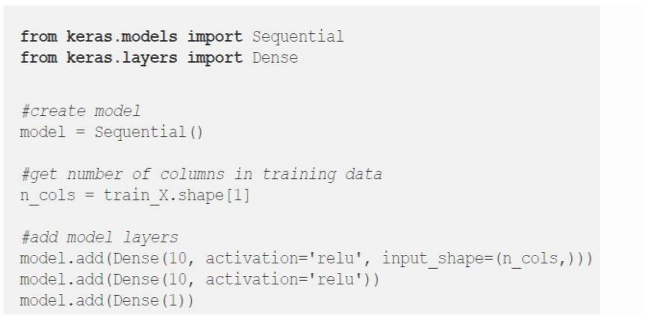
```
#create a dataframe with all training data except the target column
train_X = train_df.drop(columns=['wage_per_hour'])

#check that the target variable has been removed
train_X.head()
```

We will insert the column 'wage_per_hour' into our target variable (train_y).

#create a dataframe with only the target column
train_y = train_df[['wage_per_hour']]
#view dataframe
train_y.head()

C. Building the model

Next, we have to build the model. Here is the code:



```
from keras.models import Sequential
from keras.layers import Dense

#create model
model = Sequential()

#get number of columns in training data
n_cols = train_X.shape[1]

#add model layers
model.add(Dense(10, activation='relu', input_shape=(n_cols,)))
model.add(Dense(10, activation='relu'))
model.add(Dense(1))
```

D. Compiling the model

Next, we need to compile our model. Compiling the model takes two parameters: optimizer and loss.The optimizer controls the learning rate.

#compile model using mse as a measure of model performance
model.compile(optimizer='adam', loss='mean_squared_error')

E. Training the model

Now we will train our model. To train, we will use the 'fit()' function on our model with the following five parameters: training data (train_X), target data (train_y), validation split, the number of epochs and callbacks.

```
from keras.callbacks import EarlyStopping

#set early stopping monitor so the model stops training when it won't
improve anymore
early_stopping_monitor = EarlyStopping(patience=3)

#train model
model.fit(train_X, train_y, validation_split=0.2, epochs=30,
callbacks=[early_stopping_monitor])
```

F. Making predictions on new data

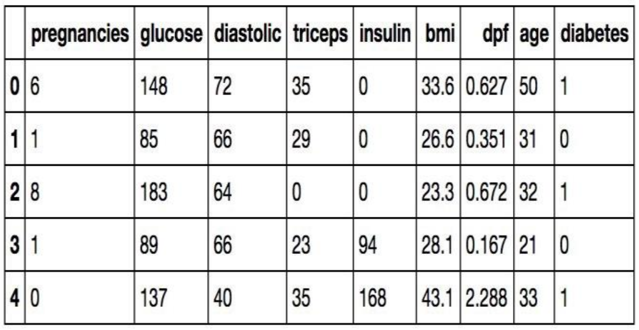
If you want to use this model to make predictions on new data, we would use the 'predict()' function, passing in our new data. The output would be 'wage_per_hour' predictions.

```
#example on how to use our newly trained model on how to make
predictions on unseen data (we will pretend our new data is saved in
a dataframe called 'test_X').

test_y_predictions = model.predict(test_X)
```

IV. Classification model

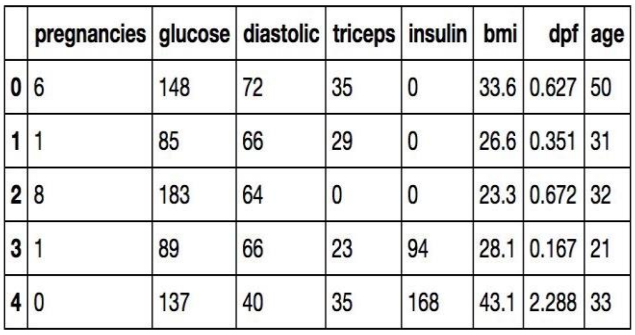
Now let's move on to building our model for classification. Since many steps will be a repeat from the previous model, I will only go over new concepts.For this next model, we are going to predict if patients have diabetes or not.

```
#read in training data
train_df_2 = pd.read_csv('documents/data/diabetes_data.csv')

#view data structure
train_df_2.head()
```



	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age	diabetes
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

```
#create a dataframe with all training data except the target column
train_X_2 = train_df_2.drop(columns=['diabetes'])

#check that the target variable has been removed
train_X_2.head()
```

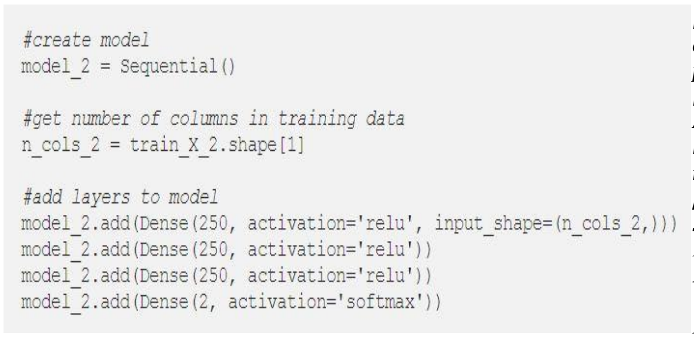


	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age
0	6	148	72	35	0	33.6	0.627	50
1	1	85	66	29	0	26.6	0.351	31
2	8	183	64	0	0	23.3	0.672	32
3	1	89	66	23	94	28.1	0.167	21
4	0	137	40	35	168	43.1	2.288	33

When separating the target column, we need to call the 'to_categorical()' function so that column will be 'one-hot encoded'. Currently, a patient with no diabetes is represented with a 0 in the diabetes column and a patient with diabetes is represented with a 1. With one-hot encoding, the integer will be removed and a binary variable is inputted for each category. In our case, we have two categories: no diabetes and diabetes. A patient with no diabetes will be represented by [1 0] and a patient with diabetes will be represented by [0 1].

```
from keras.utils import to_categorical

#one-hot encode target column
train_y_2 = to_categorical(train_df_2.diabetes)

#vcheck that target column has been converted
train_y_2[0:5]
```

```
array([[ 0.,  1.],
       [ 1.,  0.],
       [ 0.,  1.],
       [ 1.,  0.],
       [ 0.,  1.]], dtype=float32)
```



```
#create model
model_2 = Sequential()

#get number of columns in training data
n_cols_2 = train_X_2.shape[1]

#add layers to model
model_2.add(Dense(250, activation='relu', input_shape=(n_cols_2,)))
model_2.add(Dense(250, activation='relu'))
model_2.add(Dense(250, activation='relu'))
model_2.add(Dense(2, activation='softmax'))
```

The last layer of our model has 2 nodes — one for each option: the patient has diabetes or they don't.The activation is 'softmax'. Softmax makes the output sum up to 1 so the output can be interpreted as probabilities. The model will then make its prediction based on which option has a higher probability.

#compile model using accuracy to measure model performance
model_2.compile(optimizer='adam',
loss='categorical_crossentropy', metrics=['accuracy']).

We will use 'categorical_crossentropy' for our loss function. This is the most common choice for classification. A lower score indicates that the model is performing better.To make things even easier to interpret, we will use the 'accuracy' metric to see the accuracy score on the validation set at the end of each epoch.

#train model
model_2.fit(X_2, target, epochs=30, validation_split=0.2,
callbacks=[early_stopping_monitor])

CONCLUSION AND FUTURE ENHANCEMENT

Big Data presents significant challenges to deep learning Deep learning technique can be used for data analysis so that able to help us find abstract patterns in Big Data. If we apply Deep Learning to Big Data, we will find unknown and useful patterns that were impossible. Deep Learning has an advantage of potentially providing a solution to address the data analysis and learning problems found in massive volumes of input data. In this paper , two models are utilized to train an efficient deep machine learning based prediction model for predicting diabetes in patients and predicting employee's wage per hour . Deep learning makes this task more effective as deep learning is efficient in case of image data processing. For future work, optimization of results will be done for improving the performance of prediction. Further, more volume of image data will be collected and data processing is done on the top of Hadoop framwork.

REFERENCES

[1] Karan Jakhar, Nishtha Hooda, "Big Data Deep Learning Framework using Keras: A Case Study of Pneumonia Prediction" IEEE 978-1-5386-6947-1/18,2018

[2] Yann LeCun, Yoshua Bengio and Geoffrey Hinton, "Deep learning", Nature, Vol 521, May 2015.

[3] Xue-Wen Chen and Xiaotong Lin, "Big Data Deep Learning: Challenges and Perspectives", IEEE, Volume 2, May 2014.

[4] Shui Yu, "Big Privacy: Challenges and Opportunities of Privacy Study in the Age of Big Data", IEEE, Volume 4, June 2016.

[5] P. Rajpurkar et al. "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning."arXiv preprint arXiv:1711.0522,2017.

[6] WHO. Pneumonia, 2016 [Online] Available: http://www.who.int/news-room/fact-sheets/detail/pneumonia [Accessed: May 24, 2018]

[7] X. Chen. "Big data deep learning: challenges and perspectives."IEEE access", pp. 514-525, 2014.

[8] A. Gandomi et al. "Beyond the hype: Big data concepts, methods, and analytics."International Journal of Information Management vol 35(2), pp.137-144, 2014.

[9] Building A Deep Learning Model using Keras, 2018, [ONLINE] Available: https://towardsdatascience.com/building-a-deep-learning-model-using-keras-1548ca149d37

[10] Daniele Ravı, Charence Wong, Fani Deligianni, Melissa Berthelot, Javier Andreu-Perez, Benny Lo, and Guang-Zhong Yang. "Deep Learning for Health Informatics", IEEE Journal of Biomedical and Health Informatics, Vol. 21, No. 1, January 2017.

[11] B M Wilamowiski, Bo Wu, and Janusz Korniak, "Big data and Deep Learning", , 20th Jubilee IEEE International Conference on Intelligent Engineering Systems, June 30-July 2, 2016.

[12] X. Chen, and X. Lin, "Big Data Deep Learning: Challenges and Perspectives". In Access, IEEE, vol.2,pp.514,525,doi:10.1109/ACCESS.2014.2325029, 2014.

[13] L. Deng, "A tutorial survey of architectures, algorithms, and applications for Deep Learning", In APSIPA Transactions on Signal and Information Processing, 2013.

[14] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, ''Exploring strategies for training deep neural networks,'' J. Mach. Learn. Res., vol. 10, pp. 1–40, Jan. 2009.

[15] K. Kavukcuoglu, M. A. Ranzato, R. Fergus, and Y. LeCun, ''Learning invariant features through topographic filter maps,'' in Proc. Int. Conf. CVPR, 2009, pp. 1605–1612.