Raw Data to Feature Space*

Christina Fleurisma[1]

***Abstract*— The main focus of this paper is on challenges faced during the extraction from given raw data into feature space along with appropriate subspace in order to develop and train machine learning models.**

I. INTRODUCTION

Data science has become one of the most prominent field in Computer Science. It works by collecting, preparing, analyzing, visualizing, and preserving data. Data Science, especially Big Data and machine learning, there are four essential objectives which goes from the understanding of data, the understanding of machine learning, the understanding of systems, and the understanding of scalability and complexity. In addition, the learning and understanding of programming languages like Python, R, and, Scala and the most recent big data systems such as Hadoop and Spark is crucial for data analysts as well as data scientists.

II. SETTING PROGRAMMING ENVIRONMENT

Machine learning requires specific coding environment in order to collect, visualize, and transform data. For this assignment, anaconda was highly recommended and python was a suggested programming language. In order to download anaconda3, the used of some specific command lines were necessary.

cd Downloads/
bash script.sh
source activate ComputerVision
conda env remove -n ComputerVision

A.

III. PROCEDURE FOR IMAGE SELECTION

A.

In this assignment, one of the challenges was the choice of the data. Choosing which pictures, in this case would determine the learning efficiency of the machine. In this case, the pictures of cherry, mango, and pineapple were used taking into account multiple features such as size, texture, shape, and color. While cherry is small with a glossy texture, round shaped, and dark plain red; mango is medium size with a soft but non glossy texture, oval shape and a fading red to yellow and greenish color; pineapple on the other hand is bigger, has a rough, lumpy texture, a more cylinder shaped bottom with a green hat/leaves.

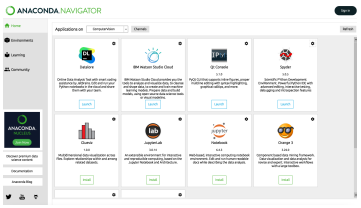


Fig. 1. a snapshot Anaconda3 enviroment

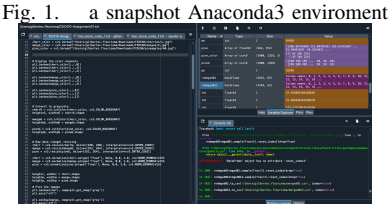


Fig. 2. a snapshot of Spyder IDE

Fig. 3. Cherry photo.

Fig. 4. Mango photo

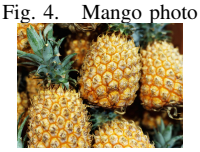


Fig. 5. Pineapple photo

The dataset is accurate, balanced, and complete. It could be considered as big data with no that need to be scaled and normalized.

IV. PART II

A. *Task 1*

1. In the first task of assignment 2, I used the traintestsplit from sklearn.modelselection and give

*This work was not supported by any organization

[1]H. Kwakernaak is with Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, 7500 AE Enschede, The Netherlands h.kwakernaak at papercept.net

[2]P. Misra is with the Department of Electrical Engineering, Wright State University, Dayton, OH 45435, USA p.misra at ieee.org

the test size of 0.2 to create the 80:20 splits of training and testing data respectively. I created four variable and give them name of Xtrain, Xtest, ytrain, and ytest, where y represent the response variable and x represent the independent variable and train and test refer to the training and testing data set. 2. I selected feature 15 and feature 30 to compare their distribution using histograms. I created histogram using matplotlib package. Histogram of 15 feature on train set

B. Task 2

I used LogisticRegression() from sklearn.linearmodel to implement the lasso regression. For sake of lasso regression, we used the liblinear solver and l1 as penalty. 2.After implementing the lasso regression, I create a new data frame containing the training, testing, and predicted values on column 65 using pd.concat() where pd represents the pandas package. 3. I used pd.crosstab() to create the confusion matrix from column 65 and 66, and give it a name of CCtest. 4. For two classes dataset,

5. I selected precision and recall, Pr0 represent the precision of 0 category, pr1 represent the precision of 1 category, and pr2 represent the precision of 2 category and represent the recall of 0,1, and 2 category respectively.

C. Task 3

I selected randomforest model and used it through RandomForestClassifier() from sklearn.ensemle, and implemented it with following hyper parameter, randomstate=0, nestimators=1000, oobscore=True, njobs=-1 I create a new data frame containing the training, testing, and predicted values on column 65 using pd.concat() where pd represents the pandas package. 3. I used pd.crosstab() to create the confusion matrix from column 65 and 66, and give it a name of $CC_testOur$AccuracyScore OurPrecisionScore OurSensitivityScore OurSpecificityScore BuiltInAccuracy BuiltInPrecision BuiltInSensitivity (recall) were also calculate after implementing the RandomForestClassifier

D. Task 4

I used metrics.classificationreport() to get both precision and recall for our model. 2. Following table represents the output of metrics.classificationreport.

By taking a look at metrics, one can easily identify that the two class data and two class classification models are superior among other variables.

E. Part 3

The third part of the assignment was basically using Databricks to implement part II. In order to

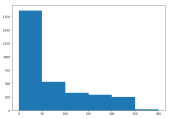


Fig. 6. Histogram of 15 feature on train set

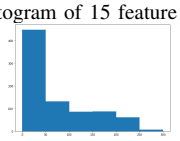


Fig. 7. Histogram of 15 feature on test set

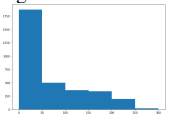


Fig. 8. Histogram of 30 feature on train set

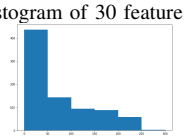


Fig. 9. Histogram of 30 feature on test set

For two class datasets,

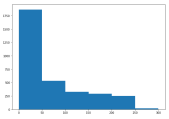


Fig. 10. Histogram of 15 feature on train set

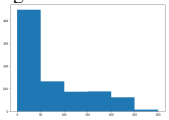


Fig. 11. Histogram of 15 feature on test set

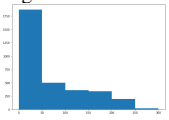


Fig. 12. Histogram of 30 feature on train set

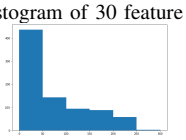
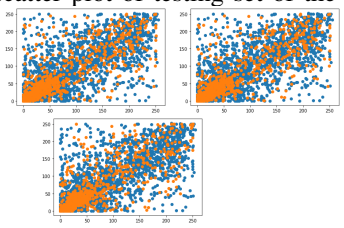


Fig. 13. Histogram of 30 feature on test set

Both features follows the same distribution on testing and training set

Following plot represent the scatter plot of training set in blue color for feature 15 and 30, orange color represent the scatter plot of testing set of the same features.



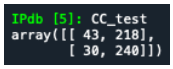


Fig. 14. Histogram of 15 feature on train set

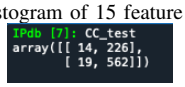


Fig. 15. Histogram of 15 feature on test set

For three classes data set,

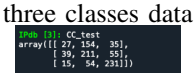
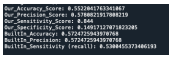
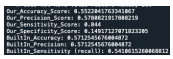
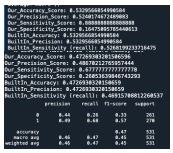
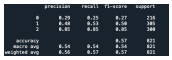


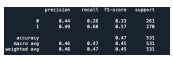
Fig. 16. Histogram of 30 feature on train set

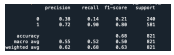












implement it to Databricks, 2 tutorials were followed. One to help signing up to Databricks, the other one provided by the professor to transfer data.

Once the dataset is imported to Databricks, SQL was used to access it.

```
df = sqlContext.sql("SELECT * FROM merged012")
```

```
# Read a feature space
input_data = pd.read_csv("/Users/gilbertec.fleurisma/merged012.csv",header=0)
```

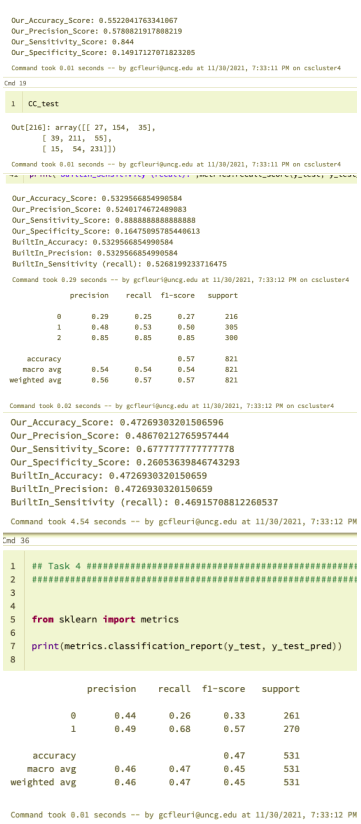
Instead of using pd.readcsv to read the data, SQL is used.

In order to read the imported data, .toPandas() is used. From then, the code will be similar for both spyder and Databricks. From using the two, it can be concluded that Databricks processes the data faster that Spyder does.

From evidence provided above, both Spyder and and Databricks give the same results.

V. CONCLUSION

The goal of this assignment were to extract feature from a set of given data, to construct a feature space or spaces, and finally to develop and train machine learning models. The challenges faced during this assignment such as creating the programming environment, selecting the data, scaling and normalizing the data were met successfully. Throughout this work, it was shown how computers can process and analyze data. In the second part of the assignment, the goals were to implement a regression-based model, the random forest, and evaluating the learning models. The data was split 20:80 for the purpose of training and testing. Once this task was complete, Databricks



was used in order to compare the difference between the performance within itself and Spyder. We have concluded that different methods were used to read the data between Databricks and Spyder, however, they provide the same responses and calculation but Databricks at a faster pace.