**Lab Assignment #4 –**Spark machine learning

Due Date: Friday, Week 11

Purpose:

The purpose of this Lab assignment is to:

1. To carry out pre-processing steps using spark transformers.
2. To build a machine learning pipeline.
3. To build a supervised machine learning model.
4. To build an un-supervised machine learning model.

General Instructions:

Be sure to read the following general instructions carefully:

1. This assignment must be completed individually by all the students.
2. Only provide the requested screenshots and make sure to have a complete screenshot, partial screenshots will not earn any marks.
3. You will have to add all the analysis and screenshots in the Analysis report.
4. You will have to provide a **demonstration video for your solution**and upload the video together with the solution on **eCentennial** through the assignment link. See the **video recording instructions** at the end of this document.
5. In your 8-minute demonstration video you should explain your solution clearly, going over the main code blocks and the purpose of each method also demoing the execution of the code. Youtube links and links to google drive or any other media are not acceptable, the actual recording file must be submitted.
6. Any submission without an accompanying video will lose 25% of the grade.
7. Any submission without an accompanying Analysis report will lose 20% of the grade.

Assignment Pre-requisites:

1. Spark
2. Datasets indicated in the exercises
3. VMware platform image for COMP 251

Assignment Exercises

Exercise #1 (Supervised learning decision trees)

1. Research and investigate the LIBSVM format, in your analysis report define the format and show an example with explanation.
2. Load the data stored in the file “sample\_libsvm\_data.txt” from the data available on the VMware image under the directory /home/centos/data/ into a dataframe and name it df\_x *where x is your firstname*. Use infer schema, notice that you need to use the format is LIBSVM when you create the dataframe.
3. Carry out some basic investigation: count the number of records, count the number of columns print the inferred schema and explain what each column contains and record the results in your analysis report.
4. Use the StringIndexer to index labels, in other words you will add metadata to the label column. Name the output column "indexedLabel\_x” *where x is your first name*. Store the result in a variable named labelIndexer\_x *where x is your first name.*  To Learn more about the StringIndexer checkout the following link <https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.feature.StringIndexer.html>
5. Use the VectorIndexer to automatically identify categorical features, and index them. Set the maxCategories to 4. Name the output column " indexedFeatures \_x” *where x is your first name*. Store the result in a variable named featureIndexer \_x *where x is your first name.*  To Learn more about the VectorIndexer checkout the following link <https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.feature.VectorIndexer.html>
6. Printout the following:
   1. Name of input column
   2. Name of output column
   3. # of features
   4. Map of categories

Also note the results in your written response.

1. Split your original data into 65% for training and 35% for testing and store the training data into a datafrmae named training\_x and testing\_x respectively where x is your firstname.
2. Create an estimator object that contains a decision tree classifier make sure to set the correct input and output columns you created during the transformation steps 4 & 5 above. Name the estimator DT\_x *where x is your firstname.*
3. Create a pipeline object with three stages the first two are the transformers you defined in steps 4 & 5 and the third is the decision tree estimator you defined in step 8. Name the pipeline object pipeline\_x *where x is your firstname.*
4. Fit the training data to the pipeline. Store the results into an object named model\_x, *where x is your first name.*
5. Using the model\_x predict the testing data. Store the results into a dataframe named predictions\_x where x is your firstname.
6. Print the schema of the predictions and note the results into your analysis report.
7. Print the accuracy of your model and the test error and note the results in your analysis report.
8. Show the first 10 predictions with the actual labels and features take a screenshot and add it to your analysis report.

Exercise #2 (Un-supervised learning clustering)

1. Download the wine dataset wine.csv accompanied with this assignment and move it to a folder on your virtual machine.
2. Load the wine dataset into a data frame named wine\_x1
3. Using spark high level api functions (i.e. not pandas), carry out some initial investigation and record the results in your analysis, at minimum provide the following:
   1. Printout the names of columns
   2. Printout the types of each column
   3. Printout the basic statistics mean, median, the four quartiles
   4. Printout the minimum, maximum value for each column
   5. Generate and printout a table showing the number of missing values for each column. (*Hint: use isnan, when, count, col)*
4. Show all the distinct values in the “quality” column.
5. Show the mean of the various chemical compositions across samples for the different groups of the wine quality.
6. Re-load the wine dataset into a data frame named wine\_x as you load add a new column named feature\_x of vector type that contains four columns as follows:

“citric acid","volatile acidity","chlorides","sulphates"

Spread the data frame across 3 RDD partitions. (Hint: use coalesce)

1. Cache the dataframe.
2. Define a estimator that uses K-means clustering to cluster all the wine instances into 6 clusters using the new feature\_x vector column you added in step #6.
3. Print the cluster sizes and the cluster centroids, record the results in your analysis report and write some conclusions.
4. Repeat steps 8&9 but set the number of k to 4.

**Naming and Submission Rules:**

1. You must name your submission according to the following rule:

**YourFullname\_COMP251assignmentnumber**.Example: **AdamPerjouski\_COMP251assignment1**

1. Please add all the commands/instructions into a python script.
2. There is no need to zip the files.
3. Upload the submission file on e-Centennial using the Assignment link(s).
4. In total you should submit the following:
   1. One demonstration video
   2. One analysis report
   3. 2 python scripts for exercises 1&2

**Rubric (applies to each exercise)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Evaluation criteria | **Not acceptable** | **Below**  **Average** | **Average** | **Competent** | **Excellent** |
|  | **0% - 24%** | **25%-49%** | **50-69%** | **70%-83%** | **84%-100%** |
| Requirements in exercises  50% | Missing all requirements required | Some requirements are implemented. | Majority of requirements are implemented but some are malfunctioning. | Majority of requirements implemented. | All requirements are implemented  Correctly. |
| Instruction/ Code Documentation on python script  5% | No comments explaining code.  Missing screenshots | Minor comments are implemented. | Some code is correctly commented. | Majority of code is correctly commented. | All code is correctly commented. |
| Written analysis  Content  15% | Missed all the key ideas; very shallow. | Shows some thinking and reasoning but most ideas are underdeveloped. | Indicates thinking and reasoning applied with original thought on a few ideas. | Indicates original thinking and develops ideas with sufficient and firm evidence. | Indicates synthesis of ideas, in-depth analysis and evidences original thought and support for the topic. |
| Written analysis report format and organization  5% | Writing lacks logical organization. It shows no coherence and ideas lack unity. Serious errors. No transitions.  Format is very messy. | Writing lacks logical organization. It shows some coherence but ideas lack unity. Serious errors.  Format needs attention, some major errors. | Writing is coherent and logically organized. Some points remain misplaced.  Format is neat but has some assembly errors. | Writing is coherent and logically organized with transitions used between ideas and paragraphs to create coherence. Overall unity of ideas is present. Format is neat and correctly assembled. | Writing shows high degree of attention to logic and reasoning of all points. Unity clearly leads the reader to the conclusion.  Format is neat and correctly assembled with professional look. |
| Demonstration Video  25% | Very weak no mention of the code changes. Execution of code not demonstrated. | Some parts of the code changes presented.  Execution of code partially demonstrated. | All code changes presented but without explanation why. Code demonstrated. | All code changes presented with explanation, exceeding time limit. Code demonstrated. | A comprehensive view of all code changes presented with explanation, within time limit. Code demonstrated. |

**Demonstration Video Recording**

Please record a short video (max 4-5 minutes) to explain/demonstrate your assignment solution. You may use the Windows 10 Game bar to do the recording:

1. Press the Windows key + G at the same time to open the Game Bar dialog.

2. Check the "Yes, this is a game" checkbox to load the Game Bar.

3. Click on the Start Recording button (or Win + Alt + R) to begin capturing the video.

4. Stop the recording by clicking on the red recording bar that will be on the top right of the program window.

(If it disappears on you, press Win + G again to bring the Game Bar back.)

You'll find your recorded video (MP4 file), under the Videos folder in a subfolder called Captures.

Or

You can use any other video recording package freely available.