NATIONAL TECHNOLOGICAL INSTITUTE OF MEXICO
TECHNOLOGICAL INSTITUTE OF TIJUANA

ACADEMIC SUBDIRECTION
Systems and Computing Department

SEMESTER
agosto - diciembre 2021

ACADEMIC CAREER
Eng. Information and communication technologies

MATERIA Y CLAVE:
Datos MasivosBDD-1704TI9A

STUDENT'S NAME AND REGISTRATION:
Castillo Ramirez Guadalupe 17213043
Velázquez Farrera César Alejandro 17212937

NAME OF THE JOB:
Practice #4 – Multilayer Perceptron Classifier

UNIT:
Unit II

NAME OF THE TEACHER:
M.C. José Christian Romero Hernández

Tijuana, Baja California **Due Date:** November 4, 2021

Introduction

In this current document, the topic Multilayer Perceptron Classifier (CPM), its operation will be explained. The example that comes in this document is made with the information provided by Apache Spark on its official website. It is available at the following URL: [https://spark.apache.org/docs/2.4.7/ml-classification-regression.html#multilayer-perceptron-classifier]

A multilayer perceptron classifier is a classifier supervised learning algorithm based on the Feedfoward artificial neural network. The MPC consists of several layers of neurons that are activated in various ways according to the input data. Each neuron has a function that, once activated, transmits an output signal. All neurons map inputs to outputs by a linear combination of inputs with node W weights and bias, applying the activation function. That can be written in matrix form for CPM.

$$\mathrm{y}(\mathbf{x}) = \mathrm{f_K}(\ldots \mathrm{f_2}(\mathbf{w}_2^T \mathrm{f_1}(\mathbf{w}_1^T \mathbf{x} + b_1) + b_2) \ldots + b_K)$$

The CPM also has a hidden layer because it is not visible when entering data. The last hidden layer is called the output layer, this is responsible for generating a value or vector of values that correspond to the format required by the problem.

All the layers are connected to the layer that follows it in the network. The nodes in the incoming layer represent the input of the data.

Depending on the computational resources available, the neural networks that can be built can be very thick. Building a very immersed classifier model falls into the category of deep learning, but requires a lot of resources.

Once the classifier has been trained, it can be used to make predictions on data.

Development

The example below uses a spark-shell environment powered by Java Runtime V8 and Scala is used as the input language. Spark is available for MacOS, Windows, and GNU / Linux distributions.

Step 1. Load the following libraries that will be used through the example:

- ml.classification.MultilayerPerceptronClassifier
- ml.evaluation.MulticlassClassificationEvaluator

```
import org.apache.spark.ml.classification.MultilayerPerceptronClassifier
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
```

Step 2. Load, analyze and convert the data into a Data Frame:

```
val data = spark.read.format("libsvm").load("sample_data/sample_multiclass_classification_data.txt")
```

Step 3. Divide the Data Frame into two parts (training_set, test_set). 70% of the data is saved for training purposes and the rest to evaluate the model.

```
val splits = data.randomSplit(Array(0.6, 0.4), seed = 1234L)
```

```
val train = splits(0)
val test = splits(1)
```

Step 4. Specify the layers for the neural network, the data input layer must have 4 inputs, two intermediate layers of size # 5 and # 4 and an output of 3 classes.

```
val layers = Array[Int](4, 5, 4, 3)
```

Step 5. Create a trainer and send its parameters:

```
val trainer = new MultilayerPerceptronClassifier()
  .setLayers(layers)
  .setBlockSize(128)
  .setSeed(1234L)
  .setMaxIter(100)
```

Step 6. Train the qualifying model

```
val model = trainer.fit(train)
```

Step 7. Calculate the accuracy of the model

```
val result = model.transform(test)
val predictionAndLabels = result.select("prediction", "label")
val evaluator = new MulticlassClassificationEvaluator()
  .setMetricName("accuracy")
println(s"Test set accuracy = ${evaluator.evaluate(predictionAndLabels)}")
```

Conclusion

The math of the multilayer perceptron classifier is a bit more complicated to understand comparing decision trees and GBT classifier, but it is easy to write the correct instruction lines in the spark environment.

As we can see in the image below, the precision of the model is around 90a.19%, which is an average in terms of precision compared to other classifiers.