TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TIJUANA

SUBDIRECCIÓN ACADÉMICA

DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

SEMESTRE SEPTIEMBRE – ENERO 2021

INGENIERÍA EN SISTEMAS COMPUTACIONALES

DATOS MASIVOS

JOSE CHRISTIAN ROMERO HERNANDEZ

PROYECTO FINAL

MARTINEZ FLORES PAMELA STEPHANY 16212034

HERNANDEZ GAMBINO KEVIN JOSAFAT 17211049

Tijuana, Baja California, a 11 de Enero del 2021.

Index

Big Data Final Project

Pamela Stephany Martínez Flores, Kevin Josafat Hernández Gambino
Departamento de sistemas y computación, Instituto Tecnológico de Tijuana
Tijuana, México
pamela.martinez16@tectijuana.edu.mx, kevin.hernandezg17@tectijuana.edu.mx

Abstract. *Machine learning algorithms are those that aim to move learning processes. Thanks to them we can obtain the information we need to make decisions or predict the behavior of the data.*

The objective of this investigation is to make a performance comparison of Machine Learning algorithms: Decision Tree, Logistic Regression, Support Vector Machine and Multilayer Perceptron. A dataset containing 10% of an entire dataset of approximately 45000 records was used.

Keywords: *Machine learning, regression, prediction, feature selection, decision tree, multilayer perceptron, logistic regression, support vector machine, classification, algorithms, artificial intelligence.*

1. INTRODUCTION

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide.

There are different types of algorithms and these are applicable depending on the type of machine learning in which the algorithm will work. Among them there are the algorithms of: regression, Bayesian, grouping, neural networks, among others. Below we will see some machine learning algorithms and their implementation in a data set.

2. DEVELOPMENT

2.1 Support Vector Machine

Support Vector Machines constitute a learning-based method for solving classification and regression problems. In both cases, this resolution is based on a first training phase (where they are informed with multiple examples already solved, in the form of pairs {problem, solution}) and a second phase of use for problem solving. In it, SVMs become a "black box" that provides an answer (output) to a given problem (input).

The objective of the SVM (support vector machine) algorithm is to find a hyperplane in an N-dimensional space(N - the number of features) that distinctly classifies the data points.

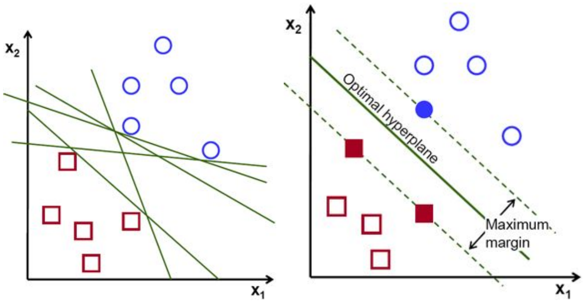

Figure 1. Hyperplane and support vector.

Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.[1]

2.2 Decision Tree

A decision tree is a tree structure similar to a flowchart where an internal node represents a feature (or attribute), the branch represents a decision rule, and each leaf node represents the result.

The top node in a decision tree known as the *root node.* Learn how to partition based on the value of the attribute. Divides the tree in a recursive way called *recursive partition.*

This flowchart-like structure helps you make decisions. It is a visualization as a flow diagram that easily mimics thought at the human level. That's why decision trees are easy to understand and interpret.

The decision trees classify the examples by sorting them by the tree from the root to some leaf node, with the leaf node providing the classification to the example, this approach is called Top-down Approach.[2]

2.2.1 Terminology related to Decision Trees

1. **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.
4. **Leaf / Terminal Node:** Nodes do not split is called Leaf or Terminal node.
5. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
6. **Branch / Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

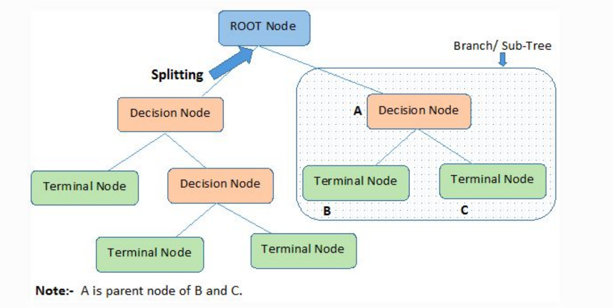

Figure 2. Decision Tree Representation.

Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive and repeated for each subtree rooted in the new nodes.

Each node in the tree acts as a test case for some attribute, and each edge descending from the node corresponds to the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new node.[3]

2.2.2 Operation of the decision tree algorithm

The basic idea behind any decision tree algorithm is as follows:

1) Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2) Make that attribute a decision node and divide the data set into smaller subsets. recursively for each child until one of the conditions coincides:
   - All tuples belong to the same attribute value.
   - There are no more attributes.
   - There are no other instances.

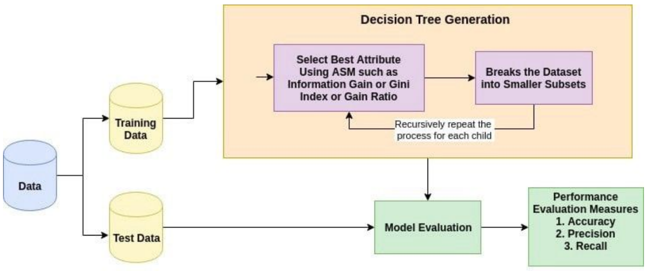
Decision Tree Generation
Select Best Attribute Using ASM such as Information Gain or Gini Index or Gain Ratio
Breaks the Dataset into Smaller Subsets
Recursively repeat the process for each child
Training Data
Data
Test Data
Model Evaluation
Performance Evaluation Measures
1. Accuracy
2. Precision
3. Recall

Figure 3. Operation of the decision tree algorithm.

2.3 Logistic Regression

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

Sometimes logistic regressions are difficult to interpret; the Intellectus Statistics tool easily allows you to conduct the analysis, then in plain English interprets the output.

Logistic Regression was used in the biological sciences in the early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical.

Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

$$\frac{1}{1+e^{-value}}$$

Where e is the base of the natural logarithms (Euler's number or the EXP() function in your spreadsheet) and value is the actual numerical value that you want to transform. Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.[4]

2.4 Multilayer perceptron classifier

Multi-layer Perceptron is a composite forward power supply (feedforward) network by a layer of input units (sensors), another layer of output units and a number of intermediate layers of process units, also called hidden layers because the exits of these neurons are not seen and they have no connections with the outside.

Each input sensor is connected with the units of the second layer, and each process unit of the second layer is connected with the units of the first layer and with the units of the third layer, and so on. The output units are connected only to the units of the last hidden layer, as shown in figure.[5]

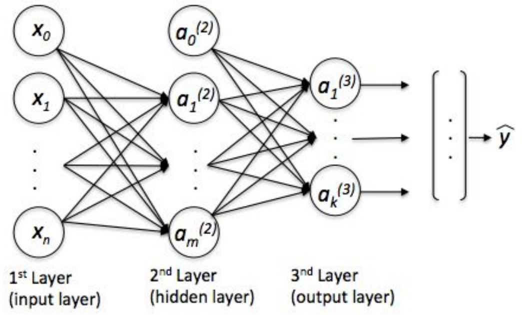

Figure 4. Representation of Multilayer Perceptron.

With this red if you want to establish a correspondence between a set of inputs and a set of detached outputs, so that

$$(x_1, x_2, \ldots, x_N) \in R^N \rightarrow (y_1, y_2, \ldots, y_M) \in R^M$$

For this a set of p training patterns is available, so we know perfectly than the input pattern $(x_1^k, x_2^k, \ldots, x_n^k)$ corresponds to the exit $(y_1^k, y_2^k, \ldots, y_m^k)$,k=1,2,…,p. That is, we know such correspondence for p patterns. So, our training package will be:

$$\{(x_1^k, x_2^k, \ldots, x_N^k) \rightarrow (y_1^k, y_2^k, \ldots, y_M^k) : k = 1,2,\ldots, p\}$$

2.4.1 Layer architecture in MLPC

Once the network is trained, the operation consists of calculating for each neuron the combination linear vector of the previous layer result (feature vector if first layer)[6]:

$$a_j^{(l)} = \sum_{i=1}^{n_l} w_{ij}^{(l)} x_i^{(l-1)} + b_j^{(l)}$$

Where:

- $x_i^{(0)}$: feature vector.
- $x_i^{(l)}$: vector result of layer *l*.
- $w_{ij}^{(l)}$ *y* $b_j^{(l)}$: are the weights of layer *l* and independent terms respectively.

These results are the *input* of the so-called activation function which is a non-linear function derivable:

$$x_j^{(l)} = f_{act}(a_j^{(l)})$$

For neurons in the *hidden layer* the activation function is the *hyperbolic tangent:*

$$f_{act}(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

For neurons in the *output layer* the *softmax function* is used:

$$f_{sal}(x) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

3. IMPLEMENTATION OF SOFTWARE

3.1 Spark

Apache Spark is an open-source, distributed processing system used for big data workloads. It utilizes in-memory caching and optimized query execution for fast queries against data of any size. Simply put, Spark is a fast and general engine for large-scale data processing. Spark runs on memory (RAM), and that makes the processing much faster than on disk drives.

The general part means that it can be used for multiple things like running distributed SQL, creating data pipelines, ingesting data into a database, running Machine Learning algorithms, working with graphs or data streams, and much more.[7]

We use Spark as it contains certain features that helped us run Machine Learning focused codes:

- **MLlib (Machine Learning Library).** This library contains a wide range of machine learning algorithms: *classification, regression, grouping and collaborative filtering.* It also includes other tools for building, evaluating, and refining ML pipes. All these features help Spark scale through a cluster.

- **Flexibility.** Apache Spark supports multiple languages and allows the developers to write applications in *Java, Scala, R, or Python.*

- **In-memory computing.** Spark stores the data in the RAM of servers which allows quick access and in turn accelerates the speed of analytics.

- **Better analytics.** Apache Spark consists of a rich set of SQL queries, machine learning algorithms, complex analytics, etc. With all these functionalities, analytics can be performed in a better fashion with the help of Spark.

3.2 Scala

Scala is a compiler based and a multi-paradigm programming language which is compact, fast and efficient. The major advantage of Scala is the JVM (Java Virtual Machine). Scala code is first compiled by a Scala compiler and the byte code for the same is generated, which will be then transferred to the Java Virtual Machine to generate the output.
Scala became the key to success for managing the huge amount of big-data.[8]

We use this language as it meets certain features that make it compatible with the platform we use to work with Machine Learning algorithms:

- Scala is capable to work with the data which is stored in a Distributed fashion. It accesses all the available resources and supports parallel data processing.
- Scala is an upgraded version of Java which was designed to eliminate unnecessary code. It supports multiple Libraries and APIs which will allow the programmer to achieve Less Down Time.
- Scala supports multiple type Constructs which enables the programmer to work with wrappers/container types with ease.
- Spark Framework is designed to handle, and process big-data and it solely supports Scala.

4. RESULTS

The following tables show the averages of the results obtained when performing 10 runs in each algorithm with the dataset used.

The first table shows the performance. In order to obtain the performance of each algorithm, variables were used to measure the time and memory used in Megabytes.

The second table shows the accuracy. In order to obtain the accuracy of each algorithm, variables were used to calculate the accuracy and error test.

Algorithm	Memory used in Megabytes	Time (Seconds)
SVM	441.9413	1265.5
Decision Tree	291.7009	3700
Logistic Regression	338.1727	969.6
Multilayer Perceptron	317.9537	1780

Table 1. Algorithms perfomance.

Algorithm	Accuracy	Test Error
SVM	0.8842	0.1158
Decision Tree	0.8840	0.1160
Logistic Regression	0.8879	0.1121
Multilayer Perceptron	0.8719	0.1281

Table 2. Algorithms accuracy.

5. CONCLUSIONS

Based on the results obtained with the runs of each of the algorithms, it can be seen that the one that consumed the least memory was the Decision Tree algorithm, however it was the longest to execute, since it took almost twice as long as the others algorithms.

Regarding the time that each algorithm consumed, we can conclude that the one that consumed the least was the Logistic Regression algorithm, also being the algorithm with the highest precision of all, but the third most time consuming to execute.

Regarding the precision of the algorithms, the least precise was that of Multilayer Perceptron Algorithm, however the four algorithms were very close in precision, having a difference of less than 2%.

SVM was the most memory consuming algorithm of the four, but it was the second most accurate on the list.

As a final conclusion we can say that the algorithm that best stood out in the characteristics that were evaluated was that of Logistic Regression, being the best balanced of the four in terms of time, memory and precision.

6. REFERENCES

[1] Rohith Gandhi. Support Vector Machine - Introduction to Machine Learning Algorithms. Retrieved from https://towardsdatascience.com/support-vector-machine

[2] Decision Tree in Machine Learning (2019, December 14). Retrieved from https://sitiobigdata.com/2019/12/14/arbol-de-decision-en-machine-learning-parte-1/

[3] Nagesh Singh Chauhan, Data Science Enthusiast. Decision Tree Algorithm. Retrieved from https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

[4] Jason Brownlee. Logistic Regression for Machine Learning. Retrieved from https://machinelearningmastery.com/logistic-regression-for-machine-learning/

[5] Multilayer Neural Networks. Multilayer Perceptron. Retrieved from https://www.inf.utfsm.cl/~hallende/bajadas/Redes/redes%20de%20Multicapa.pdf

[6] Classifier theory. Recognition of road signs for a driving aid system. Retrieved from http://bibing.us.es/proyectos/abreproy/70448/fichero/05_Capitulo4.pdf

[7] Rohan Joseph, Chartio Data Tutorials. What is Spark?. Retrieved from https://chartio.com/learn/data-analytics/what-is-spark/#components

[8] Ravi Kiran, What is Scala? A Complete Guide to Scala Programming (2020, April 24). Retrieved from

https://www.edureka.co/blog/what-is-scala/#scala-features

Alvin Alexander. How to show memory usage in a running Scala application. Retrieved from https://alvinalexander.com/scala/how-show-memory-ram-use-scala-application-used-free-total-max/