UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

LATIN AMERICAN JOURNAL OF APPLIED ENGINEERING

Data Analytics Applied to Predictive

Análisis de Datos Aplicados a lo Predictivo

Andrés Leonardo Alfonso Díaz [a], Marien Rocio Barrera Gómez [a], Iván David Alfonso Díaz [b], Jerley Andres Mejia Gallo[a]

[a] Universidad Pedagógica y Tecnológica de Colombia, Boyacá, Colombia
[b] Universidad de los Andes, Bogotá DC, Colombia

**Corresponding author: author-andres.alfonso@uptc.edu.co*

***Abstract*—**This paper describes in a simple way the development of a web application for predictive maintenance of equipment implemented in the electronics laboratory of the Universidad Pedagógica y Tecnológica de Colombia Sogamoso. This application was developed in Python language to achieve the manipulation and processing of large amounts of data. We developed a machine learning algorithm to predict damages in the laboratory equipment and enable the stakeholder to schedule maintenance to these equipments to prevent them from getting damaged. We implemented and compared the results obtained for two models (Random Forest and MLPRegressor Neural Network), being Random Forest the most accurate model.

***Keywords*—** Information management, predictive maintenance, data mining, decision making, random forest, neural network.

***Resumen*—** En este artículo se describe de manera sencilla el desarrollo de una aplicación web para mantenimiento predictivo de equipos implementado en el laboratorio de electrónica de la Universidad Pedagógica y Tecnológica de Colombia Sogamoso. Esta aplicación fue desarrollada en lenguaje Python para lograr la manipulación y tratamiento de gran cantidad de datos. Desarrollamos un algoritmo de machine learning para predecir daños en los equipos del laboratorio y habilitar al usuario para programar mantenimientos preventivos a estos equipos evitando que lleguen a dañarse. Implementamos y comparamos los resultados obtenidos para dos modelos (Random Forest y Red Neuronal MLPRegressor), siendo Random Forest el modelo más acertado.

***Palabras Claves*—** Gestión de la información, mantenimiento predictivo, minería de datos, toma de decisiones.

I. INTRODUCTION

Information and communication technologies (ICTs) are widely used to improve business and administrative processes. For cost reasons they are implemented only in particular segments of an organization as the development of a technological solution. However, in recent years there has been a movement in software development which has become a real solution for small businesses in the adoption of ICTs: open-source software, which can be defined as any program whose source code was written with the declared intention of making it available to be used, and eventually modified, by third parties [1].

This research is focused on applying Data Mining techniques and models on an existing database process from which a dataset with thousands of results will be obtained. Given the growing use of this equipment in the laboratory and the amount that is out of service due to lack of time, strategies and tools for maintenance, a web application was designed and developed in order to present to the operator a series of predictions of future damage to the equipment with a percentage of reliability and mean square error which will help the operator to make the best decision according to the case [2].

II. METHODOLOGY

During the development of this project, the most relevant aspects of the agile software development methodology, SCRUM, were implemented.

Scrum is a method for working in a team based on iterations or Sprint. Thus, it is an agile methodology, so its objective is to control and plan projects with a large volume of last-minute changes, where uncertainty is high [3].

The methodology is planned by weeks, at the end of each Sprint the work is reviewed and validated, based on this the activities to be focused on in the next Sprint are prioritized and planned.

TABLE I
APPLICATION TECHNOLOGIES

Technology	Description
Python 3.6	Programming Language (Backend), suitable for data mining.
SQL Server 2014, PostgreSQL	Database servers used to perform queries, sort the dataset and host the application in the cloud.
Django	High level web development framework for Python, allows to combine html and css.
HTML, CSS	HTML was used to indicate how the content of the application is arranged and CSS to design the visual aspect of the application. They are frontend languages (the part of the application that interacts with the user).
Visual Studio 2019	Integrated component-based development environment for the creation of high-performance applications.
Windows 10	Operating System.
Heroku	Free cloud service platform up to a certain limit compatible with Python language, allows to deploy applications quickly and securely.

Table I shows the technical specifications of the tools used for the design and development of the Web application. These include tools such as Python 3.6 (a backend programming language) to develop the business and data layers.

Castro [4], in his course "Data Science with Python", recommends the use of Python for the analysis and processing of large amounts of data. He describes the language as a tool whose goal is to develop algorithms that allow computers to learn automatically with a large amount of data as input. The author provides a series of keys that make Python one of the best when working with data mining, these keys are presented below.

- Documentation and libraries accelerate development times.
- Robust.
- Efficient, fast and scalable.
- Having libraries designed for Big Data.

Data mining techniques and models are mainly derived from artificial intelligence and statistics. These are algorithms that are applied to a certain amount of data in order to obtain a series of results [5].

Gutiérrez and Molina [6] propose an initial classification of data mining techniques into three groups: (1) predictive techniques in which variables can be classified into dependent and independent; (2) descriptive techniques in which all variables have the same status; and (3) auxiliary techniques in which a multidimensional data analysis is performed.

According to Pérez and Santín [7], predictive and descriptive techniques are used for discovery, while auxiliary techniques are used for verification.

The techniques and models implemented in the development of this application are explained in the following sections.

III. APPLICATION DEVELOPMENT

The development of the application was carried out following the flowchart in Figure 1, which primarily considers the requirements that the application must meet according to the client.

The application must communicate with and load documents from the database generated by the Agility LAB equipment loan and return control software. The Agility LAB software is a tool for the registration of the main activities that take place in the laboratories of the UPTC Sogamoso sectional, integrating the barcode plates and a computer to efficiently perform the processes of inventory of equipment, loans to users and reports for equipment damage report, active work groups and amount of equipment in use.

Figure 2 shows the diagram corresponding to the design and organization of the application to meet the customer's requirements. The three main tools of the application stand out, equipment usage prediction and fault detection by the two data mining models, neural network mlpregressor and random forest [8].

A. Data preprocessing phase

Before data can be used to make predictions, it is necessary to perform data cleaning. The steps to perform data preprocessing according to Quppler documentation are outlined below [9].

B. Import required libraries

Each time a new model is created, the *Numpy* and *Pandas* libraries need to be imported. *Numpy* is a library containing mathematical functions and is used for scientific computing, while *Pandas* is used to import and manage datasets.

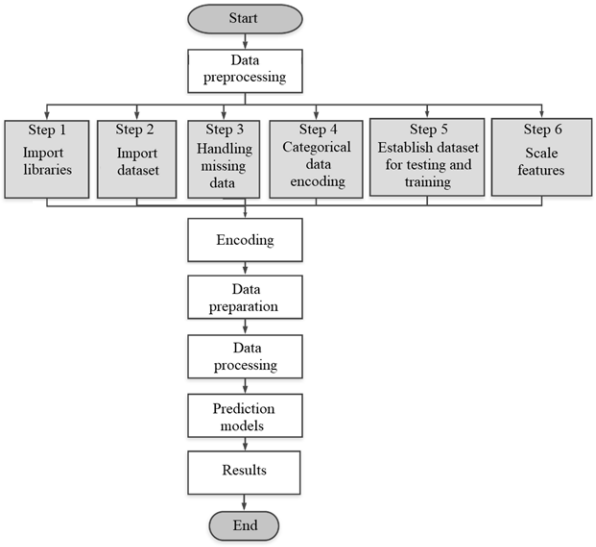


Fig. 1. Application development flowchart.

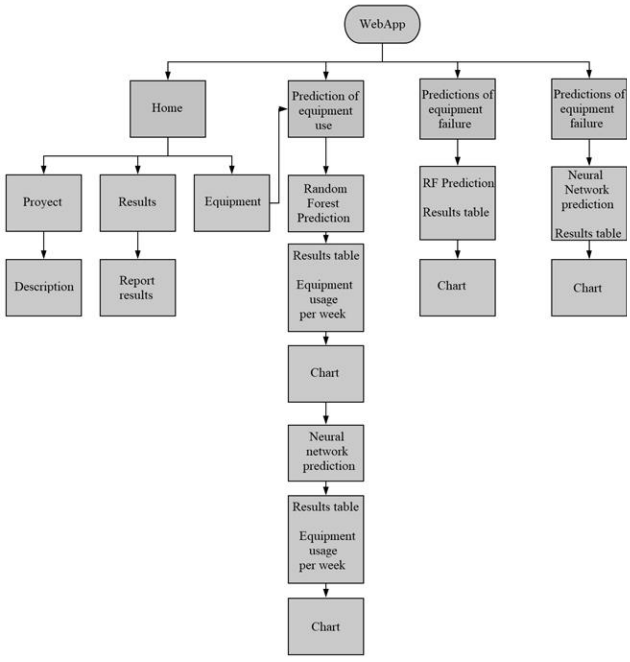


Fig. 2. Web application diagram.

C. *Import the data set*

Data sets are available in csv format or directly from an Excel file. A csv file stores tabular data in plain text. Each line of the file is a data record. The "read_csv" method of the *Pandas* library is implemented to read a local CSV file as a data frame.

D. *Handling of missing data*

The data we obtain is rarely homogeneous. Sometimes, data can be missing and must be handled so that it does not reduce the performance of the machine learning model.

Therefore, we replace the missing data by the mean or median of the whole column. We used the library *sklearn.preprocessing* which contains a class called *Imputer* that helps to address missing data.

E. *Coding of categorical data*

Any variable that is not quantitative is categorical. Examples include hair color, gender, field of study, college attended, political affiliation, disease infection status.

It is not possible to use values such as "Male" and "Female" in the mathematical equations of the model, so we coded these types of variables in numbers.

F. *Split the data set into training set and test set.*

Any variable that is not quantitative is categorical. Examples include hair color, gender, field of study, college attended, political affiliation, disease infection status.

The data are split into two sets: one to train the model (known as the training set) and the other to test the model performance (known as the test set). The split is generally 70% for training and 30% for validation. We import the *train_test_split* method from the *sklearn.model_selection* library. To build the training and test sets, we create four sets as follows.

- *X_train*: training set of the feature matrix.
- *X_test*: test set of the feature matrix.
- *Y_train*: training set of the dependent variables associated with the X-Y train sets, therefore also the same indices.
- *Y_test*: test set of the dependent variables associated with the X-Y test sets, therefore also the same indices).

G. *Feature scaling*

Df Most machine learning algorithms use the Euclidean distance between two data points in their calculations. Because of this, high magnitude features will weigh more in distance calculations than low magnitude features. To avoid this feature, we use feature normalization or Z-score normalization. We achieve this by using the StandardScaler class of sklearn.preprocessing.

H. Codification

The web application was developed in Visual Studio 2019 using the Django framework, based on a Model View Controller Architecture that separates its components to provide greater control over each part of the application, facilitating its development and maintenance.

- Model: to develop this layer, Python language was used, which allows a set of machine learning oriented technologies, including some important libraries that were imported in the data preprocessing phase.
- View: the view is basically in charge of showing the user all the answers to the requests made. Its environment was developed in HTML and CSS languages.
- Controller: the controller receives the user's requests and returns the corresponding view in response.

I. Data preparation

The information is extracted from a database generated by Agility LAB software designed and generated in SQL server 2014. The access to this database was authorized by the client, which in this case is the university. With this information we made new queries to prepare the necessary data for the development of the application.

The dataset is composed of 1,999,720 data records with information on the number of loans, reports and users using the equipment available in the laboratory inventory from January 1, 2016 to 2019.

Using SQL statements, we created new queries in order to perform cleaning and grouping of the data contained in the dataset. For the cleaning process we reviewed the data in detail, eliminated null, repeated and empty data.

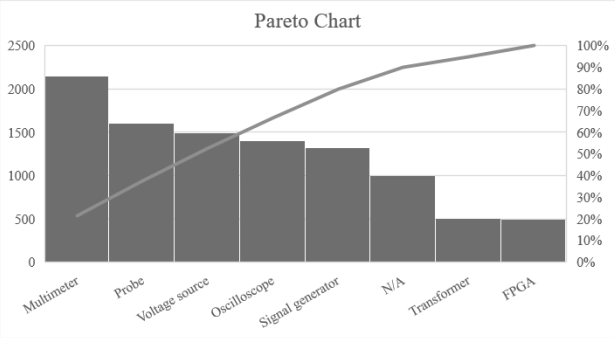
We applied bivariate classification and considered some SQL server statements. Initially we analyzed the content of the column *Loan_Report_Detail* and used the *like* operation to sort by some keyword or specific characters the report detail and then make the respective grouping.

We group the data of the new dataset by the number of weeks since the Agility lab software is running in the lab. From the data in the *start date* column, *End_date* we applied the function *DATEPART* with option *ISO_WEEK* to obtain the number corresponding to the week of the year in which the loan was made.

To select the items to be considered, we use the Pareto diagram in Figure 3. This diagram shows that the equipment with the highest number of loans and reports are multimeters, probes, oscilloscopes, and generators. This equipment is in the top 80% of circulation and is therefore more prone to damage. These data are taken from the equipment loan software and correspond to the year 2018.

In the *other* category, we classified equipment that unusually presented a report and that were within the 20% with the lowest circulation, all this to avoid errors in the prediction models. With the "sum" statement, we grouped a total of reports by item on a weekly basis, this statement adds the reports that were presented by item previously defined in the *case* and delivers a weekly total.

As an additional tool we have included in the dataset columns corresponding to the type of equipment that is loaned in the laboratory most requested according to the Agility LAB equipment loan software. With them we will also make a prediction of possible future loans. This tool aims to help the user to have an idea of equipment that should be in good condition for loan in the 4 weeks following the consultation.



	Multimeter	Probe	Voltage source	Oscilloscope	Signal generator	N/A	Transformer	FPGA
Count	2151	1610	1495	1401	1325	1002	509	498
Percent	21.5%	16.1%	15.0%	14.0%	13.3%	10.0%	5.1%	5.0%
Cum %	21.5%	37.6%	52.6%	66.6%	79.9%	89.9%	95.0%	100.0%

Fig. 3. Pareto chart.

J. MLPREGRESSOR

We implemented a neural network MLPREGRESSOR because it has a good performance when the dataset is composed of thousands of data as in our case. This is an excellent pattern classifier and processes a large amount of data simultaneously [10].

We used RELU (Rectified Linear Unid) as activation function, this function is currently the most used in neural networks. As for the learning rate, we use the following types of learning rate [11].

- Adaptive: consists of reducing the learning rate as the error stagnates after a certain number of iterations, i.e., the learning rate is initially large and

will decrease in value, with the objective of converging faster.
- Constant: is a constant learning rate that gradually decreases at each time step 't' using an inverse scaling exponent.

We use three different solution methods (Adam, lbfgs, and Sgd).

- Adam solver: a gradient-based stochastic solver performs well when the dataset is composed of thousands of data.
- Solver lbfgs: is an optimizer in the family of quasi-Newton methods, it uses the Taylor quadratic approximation of the objective function at a d-vicinity of x.
- Sgd solver: this type of solver is based on stochastic gradient descent.

We evaluate the neural network by varying the number of neurons in each hidden layer. The network will be composed of three hidden layers *('hidden_layer_sizes')*, and neural networks with 50, 75, 85 and 100 neurons in each layer are evaluated. According to the results obtained, the ideal Score should be close to 1, being the Score the reliability value of each prediction.

K. Neural Networks

One of the advantages for the implementation of the random forest algorithm is the functionality efficiently on large amount of data, improving the prediction accuracy. We implemented the *Random Forest Regressor* method from *Scikit Learn* to model the neural network.

We also used the function *GridSearchCV* to iterate over the model parameters and find the model that presents the best ones. This function implements the cross-validation algorithm for the selection of the model with the lowest error. The iterated parameters for the Random Forest model are presented below.

- n-estimators: it is equivalent to the number of trees contained in the forest. The values we iterate are 10,100,300,500,800 and 100.
- Max Depth: equals the maximum depth of the tree. The values we iterate are 2, 5, 10, 100, 300 and 500.

IV. RESULTS

The web application is intended for priority use by the user in charge of the UPTC electronics laboratory, but it can be adapted for use in other laboratories. This application has a home bar with access to the two prediction models to establish the best predictive maintenance strategy (Random Forest Regressor and Neural Network MLPregressor), a prediction option for equipment uses, and a results report corresponding to the model score according to the case. The startup screen can be seen in Figure 4.

A. Modules

The home option is a quick access back to the main page of the application, it can be accessed from any of the other functions in the menu. The modules of the web application are summarized below.

Equipment usage forecast

This function provides a suggestion of the possible number of equipment to be used in the four weeks following the query and the report values of the previous two weeks using the two prediction models.

Fig. 4. Predilab web application.

By selecting the "*Predicción uso equipos*" option, the user obtains a table (see Figure 5) corresponding to the prediction results by the *MLPRegressor* neural network model. By selecting this option, the user also obtains a graph (see Figure 6) with the prediction values of the equipment to be used by the Random Forest Regressor model classified in the eight items that correspond to the highest demand in the laboratory.

In the table and graph that the user will find in Figure 5 and Figure 6, week-1 and week-2 contain the values corresponding to the two weeks prior to the query, week+1, week+2, week+3, week+4 the prediction values for each model in the weeks following the query. All the above applies to the different functions of the application. The user will be able to consider the convenient model according to

the case. The item "N/A" groups the data that up to the date of consultation do not have data within the Agility LAB software.

Predicción Por Metodo Uno "Red Neuronal MLPRegressor"

Cantidad Uso de Equipos Predichos Por Semana

Semana	N/A	Sondas	Multimetros	Osciloscopio	Generadores	FPGA	Fuente	Transformador
Semana-2	0	0	4	0	0	0	0	0
Semana-1	56	84	114	90	112	63	82	31
Semana+1	8	1	19	2	11	-2	7	2
Semana+2	95	93	120	85	98	77	92	43
Semana+3	31	139	176	99	158	60	127	20
Semana+4	74	61	134	65	119	70	54	33

Fig. 5. Table of values corresponding to the prediction of equipment usage in the following weeks by the MLPRegressor Neural Network model.

Equipment failure prediction (model one)

We implemented the MLPRegressor Neural Network model, which makes a prediction of future damage to the most frequently borrowed equipment in the laboratory. This functionality returns a table (see Figure 7) and a bar chart (see Figure 8) with the data corresponding to the number of damages in the weeks following your query. With this information the laboratory technician in charge will be able to make an analysis of the future equipment to be damaged and take the best strategy to perform the maintenance.

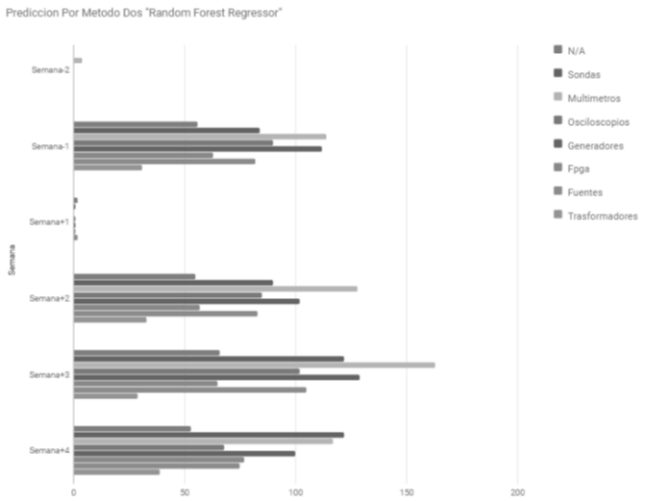


Fig. 6. Bar chart corresponding to the prediction of future equipment use by the Random Forest Regressor model.

Predicción Modelo Uno "Red Neuronal MLPRegressor"

Cantidad de Reportes Predichos por semana

Semana	Fusibles	Fuente	Gen y Osc	Multimetros	Sondas	Otros	Nrep
Semana-2	0	0	0	0	0	0	0
Semana-1	0	0	0	0	0	2	2
Semana+1	0	0	0	0	0	0	1
Semana+2	1	0	0	2	0	2	4
Semana+3	3	0	0	2	1	0	7
Semana+4	4	0	0	5	2	0	10

Fig. 7. Table of values corresponding to the prediction of future equipment damage in the following weeks by the MLPRegressor Neural Network model.

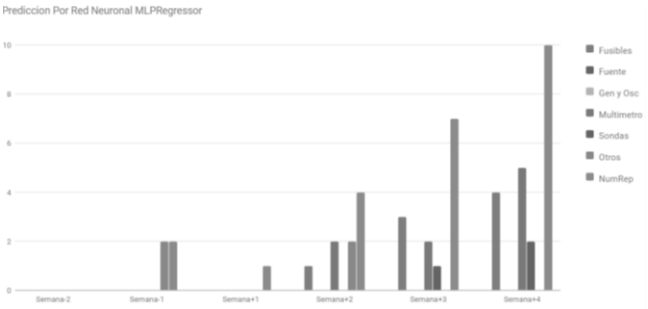


Fig. 8. Bar chart corresponding to the prediction of future equipment damage by the MLPRegressor Neural Network model.

Equipment failure prediction (model two)

With this functionality the user obtains the prediction of possible failures in the four weeks following the query date using the Random Forest Regressor model. This function is enabled for items *Fuses*, *source*, *Generator and Oscilloscope*, *Multimeters*, *Probes* and *others*. In the item *others* are grouped equipment that, due to their not so continuous use, present few reports or their report is very detailed and significantly alters the prediction algorithm.

This function provides the user with a table (see Figure 9) with the values corresponding to the data for the weeks prior to the query (week-1 and week-2), the prediction results for the 4 weeks following the query (week+1, week+2, week+3 and week+4) and a graph (see Figure 10) for a visual analysis of the data.

Predicción por Metodo Dos "Random Forest Regressor"

Cantidad de Reportes Predichos por semana

Semana	Fusibles	Fuente	Gen y Osc	Multimetros	Sondas	Otros	Nrep
Semana-2	0	0	0	0	0	0	0
Semana-1	0	0	0	0	0	2	2
Semana+1	0	0	0	0	0	0	0
Semana+2	0	0	0	0	0	1	1
Semana+3	2	0	0	2	1	0	7
Semana+4	4	0	0	4	1	0	8

Fig. 9. Tabla de valores correspondientes a predicción de futuros daños de equipos en las semanas siguientes por el modelo Random Forest Regressor.

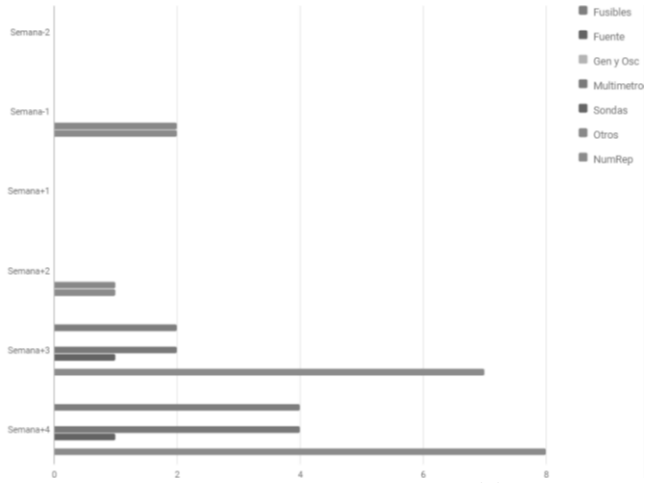


Fig. 10. Bar chart corresponding to the prediction of future equipment damage by the Random Forest Regressor model.

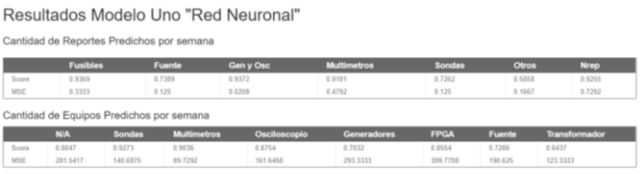
Results

In this section the user can access a report with detailed values for each prediction. This is of great help in making a better decision about which model to consider.

The main values delivered to the user are presented below.

- Score: according to scikit learn documentation the score is the R^2 regression score function (coefficient of determination). The best possible score is 1.0 and can be negative (because the model can be arbitrarily worse). A constant model that always predicts the expected value of y (output), regardless of the input characteristics, would obtain an R^2 score of 0.0. [12]
- MSE: (mean squared error function) calculates the mean squared error, it is the average of the squared loss of the individual examples, the best possible mse is the closest to zero.

The application provides two tables (Figure 11 and Figure 12) with the Score and MSE values corresponding to each predicted item and two graphical comparisons (Figure 13 and Figure 14) of the two functions in the two models. These graphs provide a visual aid to the operator to compare the two models and select the most appropriate one.


Resultados Modelo Uno "Red Neuronal"

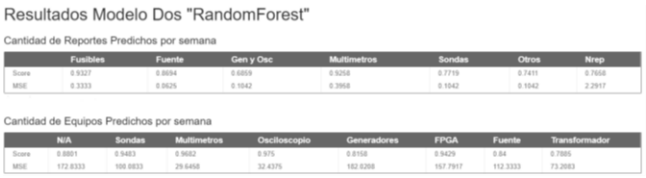
Cantidad de Reportes Predichos por semana

	Fusibles	Fuente	Gen y Osc	Multimetros	Sondas	Otros	Nrep
Score	0.9369	0.7389	0.9372	0.9101	0.7262	0.5858	0.9255
MSE	0.3333	0.125	0.0208	0.4792	0.125	0.1667	0.7292

Cantidad de Equipos Predichos por semana

	N/A	Sondas	Multimetros	Osciloscopio	Generadores	FPGA	Fuente	Transformador
Score	0.8047	0.9273	0.5030	0.8754	0.7032	0.8554	0.7266	0.6437
MSE	281.5417	140.6875	89.7292	161.6458	293.3333	399.7708	190.625	123.3333

Fig. 11. Table of values corresponding to Score and MSE for the Neural Network prediction model.


Resultados Modelo Dos "RandomForest"

Cantidad de Reportes Predichos por semana

	Fusibles	Fuente	Gen y Osc	Multimetros	Sondas	Otros	Nrep
Score	0.9327	0.8694	0.6859	0.9258	0.7719	0.7411	0.7658
MSE	0.3333	0.0625	0.1042	0.3958	0.1042	0.1042	2.2917

Cantidad de Equipos Predichos por semana

	N/A	Sondas	Multimetros	Osciloscopio	Generadores	FPGA	Fuente	Transformador
Score	0.8801	0.9483	0.9682	0.975	0.8158	0.9429	0.84	0.7885
MSE	172.8333	100.0833	29.6458	32.4375	182.0208	157.7917	112.3333	73.2083

Fig. 12. Table of values corresponding to Score and MSE for the Neural Network prediction model.

Figure 13 shows the comparison between the Score of each of the two prediction methods. With the information presented in this figure, the user can analyze the information and select the efficient prediction method for the item. Figure 23 enables the user to make a comparison between the two mean squared errors (MSE) of each method (Random Forest and Neural Network).

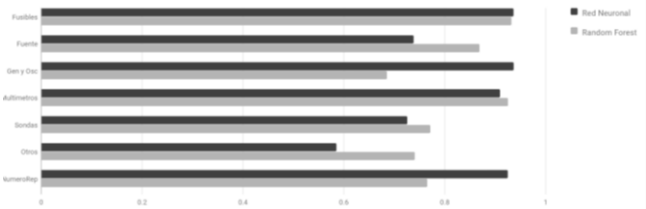


Fig. 13. Score comparison between Random forest and Neural Network models.

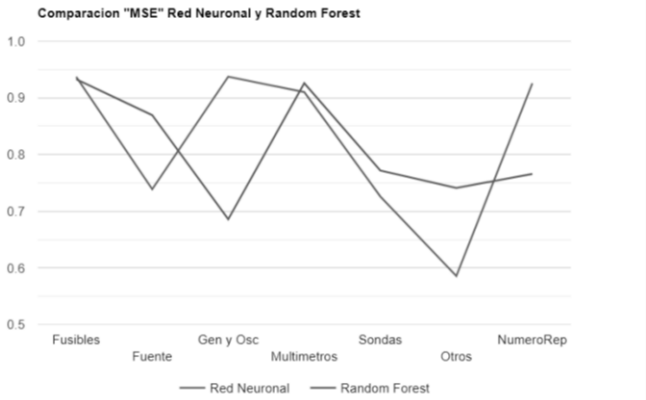


Fig. 14. Comparison of MSE between Random forest and Neural Network models.

V. CONCLUSION

In this paper we present the development and testing of a web application to support predictive maintenance of equipment in the laboratories of the Universidad Pedagógica y Tecnológica de Colombia. We have implemented two machine learning models to predict equipment failures. The results are presented to the user through a web application developed in Python. We have implemented the models *Random Forest* and *Neural Network mlpregressor*. According to the results obtained from the tests, the *Random*

Forest Regression model yielded the best results (score close to 1 and MSE close to 0) for each item to be predicted [13], [14].

For the *Random Forest* and the *Neural Network mlpregressor* models, the prediction results will improve as the amount of data fed to the processing dataset increases. As the number of inputs to the prediction system increases, more data will be available for the train and test processes.

In the data preparation process, grouping the data by day produced a very low score, making the prediction models inefficient.

According to the results, the values corresponding to the items "Multimeters" and "Fuses" in some cases may be the same because the damage report of multimeters is highly related to the damage of a fuse.

The implementation of a predictive maintenance strategy in the use of laboratory equipment reduces equipment downtime due to lack of supplies for corrective maintenance.

The results of the project represent a strategic advantage for decision making in the general maintenance plan for laboratory equipment at UPTC.

REFERENCES

[1] A. Jimenez, J. Velazquez, and A. Fuentes, "Mejoramiento de la gestion y uso de Tics en las mipymes a traves de software libre," *Ing. Sist.*, vol. 22, pp. 31–55, 2008, [Online]. Available: http://www.dii.uchile.cl/~ris/RISXXII/RISXXII.pdf#page=33.

[2] K. Gandhi, B. Schmidt, and A. H. C. Ng, "Towards data mining based decision support in manufacturing maintenance," *Procedia CIRP*, vol. 72, pp. 261–265, 2018, doi: https://doi.org/10.1016/j.procir.2018.03.076.

[3] A. Peralta, "Metodologia scrum," 2003. [Online]. Available: https://fi.ort.edu.uy/innovaportal/file/2021/1/scrum.pdf.

[4] N. Castro, "Introducción Data Science con Python," 2015.

[5] D. J. Hand and N. M. Adams, "Data Mining," *Wiley StatsRef: Statistics Reference Online*. pp. 1–7, Jun. 22, 2015, doi: https://doi.org/10.1002/9781118445112.stat06466.pub2.

[6] J. G. O. and B. Molina, "Identificación de técnicas de minería de datos para apoyar la toma de decisiones en la solución de problemas empresariales," *Rev. Ontare*, vol. 3, no. 2 SE-Artículos científicos, May 2016, doi: 10.21158/23823399.v3.n2.2015.1440.

[7] D. Santín and C. López, *Minería de datos: técnicas y herramientas*. 2007.

[8] R. Ihaka and R. Gentleman, "A Language for Data Analysis and Graphics," *J. Comput. Graph. Stat.*, vol. 5, no. 3, pp. 299–314, Sep. 1996, doi: 10.1080/10618600.1996.10474713.

[9] Sumit Anand, "Data Preprocessing in Python," *Quppler*, 2019. https://quppler.com/data-preprocessing-in-python/ (accessed Feb. 15, 2021).

[10] J. M. M. Diaz Araque, "Introducción a las Redes Neuronales Aplicadas Universidad Carlos III de Madrid," 2012.

[11] D. Jorge Matich, "Cátedra: Informática Aplicada a la Ingeniería de Procesos-Orientación I Redes Neuronales: Conceptos Básicos y Aplicaciones," 2001.

[12] J. Hao and T. K. Ho, "Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language," *J. Educ. Behav. Stat.*, vol. 44, no. 3, pp. 348–361, Feb. 2019, doi: 10.3102/1076998619832248.

[13] A. Liaw and M. Wiener, "Classification and Regression by RandomForest," *Forest*, vol. 23, Nov. 2001.

[14] J. O. Alvear, *Arboles de decision y Random Forest*. 2018.

Este estudio fue financiado por los autores. Los autores declaran no tener ningún conflicto de interés.