Universidad de Montemorelos

Facultad de Ingeniería y Tecnología

Ingeniería en Sistemas Computacionales

Prediction of the impact of Facebook posts with machine learning

Adlay Stephany Levy Méndez

1140318

Asesor: Dr. Germán Harvey Alférez Salinas

Montemorelos, Nuevo León, México

27 de abril de 2020

Abstract

Facebook is the most popular social network. It is seen as entertainment and also as a tool. This tool can help businesses to advertise themselves and see how much scope they have. In this research work we use Machine Learning to automatize the prediction of the success of posts to be published on Facebook. As a case study, we used information collected from the Facebook page of Montemorelos University. In order to predict the success of Facebook posts, we developed a web application. This web application allows users to upload a data set of the performance of a Facebook page. Our tool uses this information to train 4 classification models with 4 different Machine Learning algorithms. Then, it chooses the model with the highest accuracy and shows the result of the most accurate one. Then, the user can use this model to predict the impact of a new post.

Index terms: Facebook, Machine Learning, prediction, web application.

CONTENTS

Prediction of the impact of Facebook posts with machine learning

Adlay Levy and Germán H. Alférez

School of Engineering and Technology, Montemorelos University, Mexico

I. Introduction

A. Background

Facebook is the biggest social network worldwide. Many people use this network for entertainment. However, there are other users who use it for making themselves known as "influencers", people whose posts reach a high level of engagement. This is something desirable for companies as well. Facebook's success can be attributed to its ability to appeal to both people and businesses and its ability to interact with sites around the web [1].

Nowadays, many companies must make extensive use of social networks to properly engage their customers and to make themselves known through different ways such as advertisement. However, there is a lack of proper tools to predict the success of a new Facebook post. We argue that Machine Learning, a relatively new technology that has made a big impact in many different areas [2], can be applied to social network studies. This is why this research work describes a tool that can help users predict the impact, whether positive or negative, of a Facebook post using Machine Learning.

B. Problem statement

Since there are Facebook posts that have a better impact than others, it is important to know beforehand the impact of a new post. Despite the relevant need of predicting the impact of Facebook posts, there are no tools that can be used to predict their success.

C. Justification

Facebook helps to connect more people than any other company and has become an important asset for many businesses. Facebook has also generated data that is useful for artificial intelligence research [3]. A research carried out in 2009 confirms the usefulness of the data that Facebook has generated from its beginnings for the benefit of marketing [4]. It is because this platform has grown immensely since its inception and nowadays it has over 2.5 billion monthly active users [5].

This project intends to take advantage of these opportunities to create a tool that is useful for those who are willing to measure the impact of their posts on Facebook, before the posts are even sent. This tool is intended to be available to any user. This is why the tool is available via a web application. Machine Learning is the core of this project since it provides the mechanisms to create predictive models trained with historical data.

D. Objectives

- Create a web application to train four classification models with a data set of Facebook posts. This tool has a front end that allows users to upload a Comma Separated Value (CSV) file with activity information from Facebook posts (e.g. number of likes, type, etc.). Then, the tool automatically uses the data set to train the following classification algorithms: Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Logistic Regression (LR), and Decision Trees (DT). After training, the system returns the model with the best accuracy. The web application saves the model that offers the best accuracy to be used in future predictions.
- Create a web application that classifies the impact of a new post. Specifically, the user inputs the values of the following features: "Type" of post indicating whether it is a photo, video, link, etc.; "Day" when the post will be published; "Hour" in which the post will be published; and the expected values of "Likes", "Shares" and "Comments". With the aforementioned information, the algorithm shows the post prediction with the classification of High, Medium, or Low success.
- Evaluate the tool with a case study with data from the Facebook page of Montemorelos University.

E. Hypothesis

Machine Learning can be used to predict the negative or positive success of Facebook posts.

II. Theoretical Foundation

A. Underpinnings of our Approach

This research project is based on the following concepts as shown in Fig. 1:

1) Social Media: Social media is a group of Internet-based applications that are build on the ideological and technological foundations of the Web and that allows the creation and exchange of user-generated content [6].

With the rise of digital and mobile technologies, interaction on a large scale became easier for individuals than ever before; and as such, a new media age was born where interactivity was placed at the center of new media functions [7]. One individual can now speak to many and instant feedback is a possibility.

Most of the social media applications are used primarily for recreation or personal connections. However, they can also be used for work.

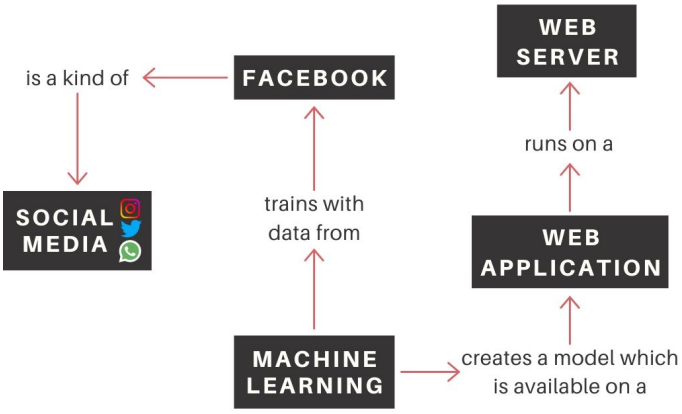

Fig. 1. Underpinnings of our approach.

2) Facebook: Is one of the biggest social networks. Facebook began in February of 2004 as a school-based social network at Harvard University. The service was created by 19-year-old Mark Zuckerberg, a Harvard University student, who ran it from a computer in his college room [3]. Then four fellow students joined him [8].

Facebook's rapid growth began as soon as it became available and has continued through the years. In the last 10 years, Facebook has accumulated more than one billion users worldwide. The aim of its creators has always been to connect the world [9]. This service is also a powerful media machine for the flow of information.

3) Machine Learning: Data analysis is a process of cleaning, transforming, and modeling data to discover useful information for business decision making [10]. It works based on analyzing the past to make future decisions. There are several types of data analysis techniques based on business and technology. For example [11]: Text Analysis, is a method to discover a pattern in a large data set using databases or data mining tools; Statistical Analysis uses past data to show what happened; Diagnostic Analysis identifies patterns behavior of data by finding the cause from the insight found in Statistical Analysis; Predictive Analysis makes predictions about future outcomes based on current or past data; and Prescriptive Analysis combines the insight from all previous analysis to determine which action to take in a current problem or decision. This project uses Predictive Analysis, specifically Machine Learning.

Machine Learning is a subfield of artificial intelligence that provides systems the ability to automatically learn and improve from experience without being explicitly programmed [12]. Machine Learning focuses on the development of computer programs that can access data and use it to learn by themselves. The two most popular methods for training Machine Learning are called Supervised and Unsupervised Learning [13]. Supervised Learning helps to build a concise model of the distributions of class labels in terms of predictor features. Unsupervised Learning uses information that is neither classified nor labeled. The algorithm acts on that information without guidance [14]. In this research work we use four Supervised Machine Learning algorithms.

There are several applications for Machine Learning. The most significant is data mining, which helps in the analysis of big data and is also useful to establish relationships between multiple features [15].

4) Web application: A web application is any computer program that performs a specific function by using a web browser at the client side. The application can be as simple as a message board or a contact form on a website or as complex as a word processor [16]. A client-server environment is one in which multiple computers share information such as entering information into a database. The "client" is the application used to enter the information, and the "server" is the application used to process and store the information. A web application relieves the developer of the responsibility of building a client for a specific type of computer or a specific operating system, so anyone can use the application along as they have Internet access [17].

5) Web server: A web server is server software, or hardware dedicated to run this software, which can satisfy client requests on the World Wide Web. A web server can, in general, contain one or more websites. A web server processes incoming network requests over HTTP and several other related protocols [18].

The web application created in this research work runs on the Flask[1] micro web framework, which works as a web server in our case. Micro frameworks are normally frameworks with little to no dependencies to external libraries [19].

Flask provides tools, libraries, and technologies that allow to build a web application with Python. A web application can be composed of web pages, blogs or even a commercial website. Flask allows to work with Python for the methods and actions and also with HTML and CSS for the creation of the page view.

B. Related Work

This project is inspired by the work presented by a computer science master's student at Middle East University. Magdy Salama works in social media for the Middle East and North Africa Union (MENA). In his research work in a course taught by Dr. Harvey Alférez, he obtained the data from the Facebook page of the Hope Channel Middle East Media Center. He included in the data set the features that he considered most important for a good classification of the post. The features that were chosen for the data set were: paid/unpaid, post type, month, period of day, content type, tone of voice, religious/nonreligious, focal gender, focal age, animation, category, and language. The classes in the data set were as follows: good, very good, bad, and very bad. Salama created a classification model with the data set via a feed forward back propagation neural network.

Also, several research works that propose Machine Learning solutions have been published at the School of Engineering and Technology of Montemorelos University.

[1]https://flask.palletsprojects.com/en/1.1.x/

In [20], the authors use deep learning and image recognition to create a mobile application to detect difficult airway. They used 240 images for model training to classify the following classes: Mallampati 1-2 for low risk of difficult airway and Mallampati 3-4 for high risk.

In [21], the authors use deep learning for image recognition of Legacy blueberries in the rooting stage that can be used in smart farms in Chile. Legacy blueberry is a variety of Southern Highbush blueberry. This species constitutes 80% of the blueberry crops in Chile. Specifically, they propose an image recognition approach based on a convolutional neural network (CNN) to detect the presence of trays with living blueberry plants, the presence of trays without living plants, and the absence of trays. The average results of the evaluation of the predictive model are as follows: accuracy: 86%, precision: 86%, recall: 88%, and F1-score: 86%.

In [22], the authors propose a solution for the dynamic evolution of simulated autonomous cars in the open world through tactics. Specifically, for an autonomous car to drive by itself, it needs to learn. A safe and economic way to teach a self-driving car to drive by itself is through simulation. However, current car simulators are based on closed world assumptions, where all possible events are already known as design time. Nevertheless, during the training of a self-driving car, it is impossible to account for all the possible events in the open world, where several unknown events may arise (i.e., events that were not considered at design time). Instead of carrying out particular adaptations for known context events in the closed world, the system architecture should evolve to safely reach a new state in the open world. In this research work, their contribution was to extend a car simulator trained by means of Machine Learning to evolve at run time with tactics when the simulation faces unknown context events.

In [23], the authors use convolutional neural networks to achieve the automatic detection of glaucoma. Glaucoma is a disease that damages the optic nerve and becomes chronic as time progresses. Its early detection is vital for treatment. Although there are tools to carry out the analysis of the optic nerve, they do not automatically detect this illness. The experiments performed obtained an average accuracy of 99%.

In [24], the authors propose a low cost software that automatically and objectively differentiates between a melanoma lesion and a benign nevus in a simple, non-invasive manner. Their approach is based on the "ABCDE" classification of lesions, image processing, and artificial neural networks. The software was developed using images of previously diagnosed malignant melanomas and non-malignant suspicious moles, obtaining a sensibility of 76.56% and a specificity of 87.58%.

III. Results

A. Methodology

This section presents the methodology of this research project.

1) Data retrieval: The first step in the development of this project was to look for the data set to be used in the experiments. For convenience, the data related to the performance of Facebook posts on the Facebook page of Montemorelos University was used as a case study[2].

This page is in constant activity with several administrators who post messages, for instance, with images of the events carried out in the campus or with general announcements. The data from the Facebook page of Montemorelos University was collected manually taking into account mainly the type of post (image, video or text), and its scope. The data set was created with the information of 220 posts compiled from January to October, 2019.

Specifically, the data set contains the following features:

- *Type:* if the post was a video, a photo, a link, or multiple photos.
- *Date:* the date in which the post was posted.
- *Hour:* the time of the day when the post was posted.
- *Share:* the number of times the post was shared.
- *Like:* the number of likes or reactions that the post obtained.
- *Comments:* the number of the interactions in terms of comments with respect to the post.

The impact of each post was manually classified with the following formula:

$$(x_1 * 1) + (x_2 * 2) + (x_3 * 3) = success$$

Where x_1 represents the number of likes or reaction that the post had, x_2 represents the number of comments, and x_3 represents the number of times that people shared the post. Based on these results, we calculated the maximum and minimum values to classify a posts as High (25 or higher), Medium (16-24), or Low (15 or lower). The impact of the posts is considered as the class to be used to train the model.

2) Tool development (Backend): Before creating the web application, the four classification algorithms used in this research work were implemented with Scikit-learn[3] version 1.13. Appendix A shows the source code of the Python script described in this section. Also, the source code for this project is available online[4]. The four algorithms used in this research work are as follows.

K-Nearest Neighbor (KNN): It is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its *k* neighbors as shown in Listing 1.

In Line 1 the function is declared with four parameters related to the training (xTrain) and test (xTest) feature values and their classes (yTrain and yTest). Line 2 calls the function KNeighborsClassifier(). The fit function in line 3 creates the classification model and the score function in line 4 returns the accuracy of the model. In Line 5 the prediction based on the given training set (xTest) is saved in a variable (KNNpredict). Line 6 returns the accuracy of the model by means of comparing the classification results of the model with the real results in yTest.

```
1 def KNNeighbors(xTrain, xTest, yTrain,
    yTest):
```

[2]https://www.facebook.com/UniMontemorelos/
[3]https://scikit-learn.org/stable/
[4]https://github.com/FacebookPostsPredictor

```
2    knn = KNeighborsClassifier(
    n_neighbors = 3)
3    fited = knn.fit(xTrain, yTrain)
4    score = knn.score(xTrain, yTrain)
5    KNNpredict = knn.predict(xTest)
6    return("KNN_Accuracy:", metrics.
    accuracy_score(yTest, KNNpredict))
```

Listing 1. KNN implementation.

Logistic Regression (LR): It is a classification algorithm used to estimate discrete values based on a given set of independent variables. It predicts the probability of occurrence of an event by fitting data to a *logit function.*

In line 1, the function is declared with parameters related to the training and testing data sets. Line 2 calls the function LogisticReg(). The fit function in line 3 creates the classification model and the score function in line 4 returns the accuracy of the model. In Line 5 the prediction based on the given training set (xTest) is saved in a variable (LRPredict). Line 6 returns the accuracy of the model.

```
1 def LogisticReg(xTrain, xTest, yTrain,
    yTest):
2    logReg = LogisticRegression()
3    fited = logReg.fit(xTrain, yTrain)
4    score = logReg.score(xTrain, yTrain)
5    LRpredict = logReg.predict(xTest)
6    return("LR_Accuracy:", metrics.
    accuracy_score(yTest, LRpredict))
```

Listing 2. LR implementation.

Support Vector Machine (SVM): In this classification algorithm, each data point is plotted in an n-dimensional (*n* being the number of features) space where the value of each feature is the value of a particular coordinate. Then a line called *separating hyper plane* or (*decision boundary*) splits the data points between two or more groups of data. The further the data points from the *decision boundary*, the more confident the algorithm is about the prediction. The closest data points to the separating hyper plane are known as *support vectors*. Listing 3 shows how to call the SVM method.

In line 1, the function is declared with parameters related to the training and testing data sets. Line 2 calls the function SuperVectorMachine(). The fit function in line 3 creates the classification model and the score function in line 4 returns the accuracy of the model. In Line 5 the prediction based on the given training set (xTest) is saved in a variable (SVMpredict). Line 6 returns the accuracy of the model.

```
1 def SuperVectorMachine(xTrain, xTest,
    yTrain, yTest):
2    svmachine = SVC()
3    fited = svmachine.fit(xTrain, yTrain)
4    score = svmachine.score(xTrain,
    yTrain)
5    SVMpredict = svmachine.predict(xTest)
6    return("SVM_Accuracy:", metrics.
    accuracy_score(yTest, SVMpredict))
```

Listing 3. SVM implementation.

Decision Trees (DT): In this classification algorithm, the data is split into two or more homogeneous sets based on most significant attributes that makes the sets distinct.

In line 1, the function is declared with parameters related to the training and testing data sets. Line 2 calls the function DecisionTree(). The fit function in line 3 creates the classification model and the score function in line 4 returns the accuracy of the model. In Line 5 the prediction based on the given training set (xTest) is saved in a variable (DTPredict). Line 6 returns the accuracy of the model.

```
1 def DecisionTree(xTrain, xTest, yTrain,
    yTest):
2    dTree = tree.DecisionTreeClassifier()
3    fited = dTree.fit(xTrain, yTrain)
4    score = dTree.score(xTrain, yTrain)
5    DTpredict = dTree.predict(xTest)
6    return("DT_Accuracy:", metrics.
    accuracy_score(yTest, DTpredict))
```

Listing 4. DT implementation.

Listing 5 presents the code used to upload a data set of Facebook posts to the web application and perform the training with the aforementioned classifiers.

In line 1, the route is mapped to the upload_file function. Line 2 defines the function "upload_file()". Line 3 verifies if the POST request method is being used. The POST method is used to send data to a server to create/update a resource. The data sent to the server with POST is stored in the request body of the HTTP request. Lines 4-7 show how the file is uploaded. The requested file has to be in CSV format. When the data has been loaded, it is extracted in lines 8-10 to be used by the classification algorithms in lines 11-14. Specifically, in line 10, the training and test data are split from the original data set. The "test_size" parameter specifies to the function that only 20% of the data will be used for testing, the other 80% will be used for training. In lines 15-18, the sorted() function is used to sort the accuracy values of each classification model. Then, the max() function chooses the maximum accuracy.

Finally, the best predictor, i.e., the one with the highest accuracy value, is saved with pickle.dump() function for later use.

```
1 @app.route('/uploader', methods = ['GET
    ', 'POST'])
2    def upload_file():
3        if request.method == 'POST':

4        fileLoaded = request.files['file'
    ]
5        fileLoaded.save(secure_filename(
    fileLoaded.filename))
6        fileSaved = fileLoaded.filename
7        data = pd.read_csv(fileSaved)
8        arrX = data[data.columns[:-1]].
    values
9        arrY = data[data.columns[-1]].
    values
```

```
10      X_train, X_test, Y_train, Y_test
    = train_test_split(arrX, arrY,
    test_size=0.2, random_state=None)
11      LRpred = LogisticReg(X_train,
    X_test, Y_train, Y_test)
12      KNNpred = KNNeighbors(X_train,
    X_test, Y_train, Y_test)
13      SVMpred = SuperVectorMachine(
    X_train, X_test, Y_train, Y_test)
14      DTpred = DecisionTree(X_train,
    X_test, Y_train, Y_test)
15      largest = [LRpred, KNNpred,
    SVMpred, DTpred]
16      sor = sorted(largest)
17      bestPredictor = max(sor)
18      print(bestPredictor)
19      pickle.dump(bestPredictor, open('
    predictor.sav', 'wb'))
20      return flask.render_template('
    uploaded.html', bestPredictor=
    bestPredictor)
```

Listing 5. Upload function.

A form was created for inserting the feature values of a new Facebook post in order to predict its impact. Listing 6 shows how the feature values collected in a form are submitted to the server. Then, the server uses the feature values to return the classification of the post based on the model saved previously. Specifically, Line 4 shows how the form collects the data that was input in the form. Then in line 5 the data is saved in a list. On line 6, the result from line 5 is converted to a list of integers (in Python 3, the result had to be converted from a map to a list). Line 7 is used to avoid having an array that is out of boundaries. Line 8 shows how the saved model is loaded with pickle.load() function. Then the data collected from the form is put into a list to use the predict() function. Then line 10 saves the result in a variable. In lines 11-15, the prediction of the new Facebook post is returned based on the classification.

```
1  @app.route('/formulario', methods = ['
    GET','POST'])
2     def prediction():
3       if request.method == 'POST':
4       to_predict_list = request.form.
    to_dict()
5       to_predict_list = list(
    to_predict_list.values())
6       to_predict_list = list(map(int,
    to_predict_list))
7       to_predict = np.array(
    to_predict_list).reshape(1,6)
8       loaded_model = pickle.load(open("
    predictor.sav", "rb"))
9       #PREDICTION WITH PICKLE
10      result = loaded_model.predict(
    to_predict)
11      if (int(result)<= 15):
12          post_predict = 'LOW'
13      else if(result >= 25)
14          post_predict = 'HIGH'
15      else:
16          post_predict = 'MEDIUM'
17      return render_template('predict.
    html', post_predict=post_predict)
```

Listing 6. Predict function.

3) Tool development (Frontend): The steps that were carried out to develop the frontend of the application are described as follows:

1) In the first step the CSV file with the information of the Facebook posts is uploaded. Listing 7 shows the source code behind the form shown in Fig. 4. Appendix B contains the complete source code for the upload interface.

Line 1 in Listing 7 specifies that the form is submitted to the uploader function, which is running on Flask. Line 3 allows the user to choose a file to be uploaded. Line 4 shows the button code that is used to submit the form to the server.

```
1 <form action = "http://localhost:5000/
    uploader" method = "POST" enctype = "
    multipart/form-data" class="form">
2     <div class="upload-btn-wrapper" >
3         <input type = "file" name = "
    file" /></br>
4         <button class="btn">Cargar
    archivo .csv</button>
5     </div>
6 </form>
```

Listing 7. Form to submit a CSV file of historical Facebook posts to the server

The file to be uploaded must contain the following feature values: post type, post date, post hour, number of shares, reactions, and comments. The last column indicates the classification of the post. This classification has been already given by the user. For instance, Table 1 shows an example of the values given for three instances taken from the Facebook page of Montemorelos University.

TABLE I
SAMPLE INPUT INSTANCE TAKEN FROM THE DATA SET

PostMessage	Type	Date	Hour	Share	Reactions	Comments	Success	Class
"Queremos..."	Video	02/09/19	19:00	10	39	4	25.66	Medium
"¿Sabes de..."	Pictures	23/10/19	17:00	2	17	1	8.33	Low
"El próximo..."	Photo	18/09/19	15:29	8	124	8	54.66	High
"En 2009, la..."	Link	22/04/19	14:26	0	23	0	7.66	Low

The data set used in the experiments, with data taken from the Facebook page of Montemorelos University, is available online[5].

Listing 8 shows how the accuracy value of the model with the highest accuracy is called with the POST request method shown on Line 1. Line 3 shows the "bestPredictor" variable that brings the result of the highest accuracy value.

[5]https://github.com/ADLAYstephanyLEVY/FacebookPostSuccessPredictor

Fig. 2. Upload file interface.

```
1 <form action="http://localhost:5000/
    uploader" method="POST">
2    <div>
3        <h1> {{ bestPredictor }}</h1>
4    </div>
5 </form>
```

Listing 8. Best predictor printed in the user interface.

3) In this step, the user inputs the feature values of the new Facebook post as shown in Figure 5. Appendix C shows the code to insert the feature values from the form.

4) The web application uses the previously saved model with the best accuracy result to classify the new Facebook post.

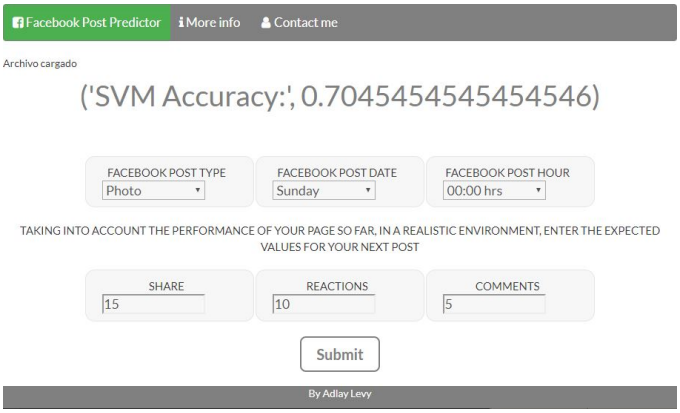

Fig. 3. User interface to insert the feature values.

5) In Fig. 4 the web application returns the classification result for the following feature values: post type, date and hour when the post will be published, and number of expected shares, reactions, and comments.

B. Outcomes

In this research work, the Montemorelos University Facebook page was used as a case study. By training the four classification algorithms with the collected data set we obtained good results. Here we present the resulting accuracy of each algorithm:

- LR Accuracy : 65.09%

Fig. 4. Interface for the prediction result.

- KNN Accuracy: 79.54%
- SVM Accuracy : 68.18%
- DT Accuracy: 86.36%

In the evaluation, we also used the precision, recall, and F1-score values. Precision is the fraction of relevant instances among the retrieved instances, while recall is the fraction of the total number of relevant instances that were actually recovered. Therefore, both precision and recall are based on an understanding and a measure of relevance [25]. The F1-score is used to combine the precision and recovery measurements into a single value. This value assumes that precision and recall matter equally. Tables 1-4 show the results for precision, recall, and F1-score in the case of the posts in the Facebook page of Montemorelos University.

TABLE II
PRECISION AND RECALL WITH LR IN THE CASE OF THE FACEBOOK POSTS OF MONTEMORELOS UNIVERSITY.

	Precision	**Recall**	**F1-score**	**Support**
HIGH	0.88	0.75	0.81	20
LOW	0.52	0.92	0.67	12
MEDIUM	0.50	0.25	0.33	12
macro avg	**0.64**	**0.64**	**0.60**	**44**
weighted avg	**0.68**	**0.66**	**0.64**	**44**

TABLE III
PRECISION AND RECALL WITH KNN IN THE CASE OF THE FACEBOOK POSTS OF MONTEMORELOS UNIVERSITY.

	Precision	**Recall**	**F1-score**	**Support**
HIGH	0.88	0.75	0.81	20
LOW	0.79	0.92	0.85	12
MEDIUM	0.69	0.75	0.72	12
macro avg	**0.79**	**0.81**	**0.79**	**44**
weighted avg	**0.80**	**0.80**	**0.80**	**44**

TABLE IV
PRECISION AND RECALL WITH SVM IN THE CASE OF THE FACEBOOK POSTS OF MONTEMORELOS UNIVERSITY.

	Precision	Recall	F1-score	Support
HIGH	0.61	0.95	0.75	20
LOW	0.78	0.58	0.67	12
MEDIUM	1.00	0.33	0.50	12
macro avg	**0.80**	**0.62**	**0.64**	**44**
weighted avg	**0.76**	**0.68**	**0.66**	**44**

TABLE V
PRECISION AND RECALL WITH DT IN THE CASE OF THE FACEBOOK POSTS OF MONTEMORELOS UNIVERSITY.

	Precision	Recall	F1-score	Support
HIGH	0.94	0.85	0.89	20
LOW	0.92	0.92	0.92	12
MEDIUM	0.71	0.83	0.77	12
macro avg	**0.86**	**0.87**	**0.86**	**44**
weighted avg	**0.87**	**0.86**	**0.87**	**44**

C. Discussion

We used four classification algorithms to predict the post success of the Facebook page of Montemorelos University. According to the results, the DT algorithm has the best accuracy (86%), precision (87%), recall (86%), and F1-score (87%) values.

IV. Conclusions and Future Work

This research work shows how Machine Learning can be used to predict the success of new posts on Facebook. As a case study, we used information collected from the Facebook page of Montemorelos University. In order to predict the success of Facebook posts, we developed a web application. This web application allows the user to upload a data set of the performance of a Facebook page. Our tool uses this information to train 4 classification models with 4 different algorithms. In the case of the data set from Montemorelos University, we got the following accuracy results: KNN=80%, LR=66%, SVM=68%, and DT=86%. We also evaluated the models in terms of precision, recall and F1-score. The best model was the one trained with the DT algorithm. Our tool chooses the best model with the highest value and shows the classification result for a new Facebook post.

As future work, the sample data from the Facebook posts will be extended in order to improve the classification results.

References

[1] S. Saige Driver, "Facebook for business: everything you need to know," Jan. 2019. [Online]. Available: https://www.businessnewsdaily.com/7761-facebook-business-guide.html

[2] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[3] N. Brügger, "A brief history of facebook as a media text: The development of an empty structure," *First Monday*, vol. 20, no. 5, 2015.

[4] J. Casteleyn, A. Mottart, and K. Rutten, "How to use data from facebook in your market research," *International Journal of Market Research*, vol. 51, no. 4, pp. 439–447, 2009.

[5] C. Jarret, "How facebook is changing our social lives," Oct. 2015. [Online]. Available: https://www.weforum.org/agenda/2015/10/how-facebook-is-changing-our-social-lives/

[6] A. M. Kaplan and M. Haenlein, "Users of the world, unite! the challenges and opportunities of social media," *Business horizons*, vol. 53, no. 1, pp. 59–68, 2010.

[7] J. Manning, "Definition and classes of social media," *K. Harvey, Encyclopedia of social media and politics*, pp. 1158–1162, 2014.

[8] D. Nations, "What is facebook?" Mar. 2019. [Online]. Available: https://www.lifewire.com/what-is-facebook-3486391

[9] M. della Cava, "How facebook changed our lives," USA TODAY, Feb. 2014. [Online]. Available: https://www.usatoday.com/story/tech/2014/02/02/facebook-turns-10-cultural-impact/5063979/

[10] J. W. Tukey, "The future of data analysis," *The annals of mathematical statistics*, vol. 33, no. 1, pp. 1–67, 1962.

[11] H. R. Bernard and G. Ryan, "Text analysis," *Handbook of methods in cultural anthropology*, vol. 613, 1998.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[13] B. C. Love, "Comparing supervised and unsupervised category learning," *Psychonomic bulletin & review*, vol. 9, no. 4, pp. 829–835, 2002.

[14] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.

[15] I. G. Maglogiannis, *Emerging artificial intelligence applications in computer engineering: real word ai systems with applications in ehealth, hci, information retrieval and pervasive technologies*. Ios Press, 2007, vol. 160.

[16] M. J. Hadley, "Web application description language," *(WADL)*, 2006.

[17] D. Nations, "What exactly is a web application?" Lifewire, 2019.

[18] T. Macaulay, "What are the best open source web servers?" ComputerworldUK, Feb. 2019.

[19] I. Maia, *Building Web Applications with Flask*. Packt Publishing Ltd, 2015.

[20] K. Aguilar, G. H. Alférez, and C. Aguilar, "Detection of difficult airway using deep learning," *Machine Vision and Applications*, vol. 31, no. 1, p. 4, 2020.

[21] I. A. Quiroz and G. H. Alférez, "Image recognition of legacy blueberries in a chilean smart farm through deep learning," *Computers and Electronics in Agriculture*, vol. 168, p. 105044, 2020.

[22] J. R. Sylnice and G. H. Alférez, "Dynamic evolution of simulated autonomous cars in the open world through tactics," p. 28, 2018.

[23] M. Espinoza, G. H. Alférez, and J. Castillo, "Prediction of glaucoma through convolutional neural networks," in *Proceedings of the 2018 International Conference on Health Informatics and Medical Systems*, 2018, pp. 90–95.

[24] C. Marín, G. H. Alférez, J. Córdova, and V. González, "Detection of melanoma through image recognition and artificial neural networks," in *Proceedings of the World Congress on Medical Physics and Biomedical Engineering, June 7-12, 2015, Toronto, Canada*. Springer, 2015, pp. 832–835.

[25] W. Koehrsen, "Beyond accuracy: Precision and recall," *Towards Data Science*, vol. 13, 2018.

Appendix A : SOURCE CODE FOR MAIN SCRIPT

```
import os
import numpy as np
import pandas as pd
import flask
import pickle
from flask import Flask, render_template, request, url_for
from werkzeug import secure_filename
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC from sklearn import tree
from sklearn.externals import joblib
from decimal import *

app = Flask(__name__)

@app.route('/')
@app.route('/index')

def index():
 return flask.render_template('index.html')

def LogisticReg(xTrain, xTest, yTrain, yTest):
 ### Logistic Regression ###
 logReg = LogisticRegression()
 fited = logReg.fit(xTrain, yTrain)
 score = logReg.score(xTrain, yTrain)
 LRpredict = logReg.predict(xTest)
 return("Logistic Regression Accuracy:",metrics.accuracy_score(yTest, LRpredict))

def KNNeighbors(xTrain, xTest, yTrain, yTest):
 ### KNN ###
 knn = KNeighborsClassifier(n_neighbors = 3)
 fited = knn.fit(xTrain, yTrain)
 score = knn.score(xTrain, yTrain)
 KNNpredict = knn.predict(xTest)
 return("KNN Accuracy:", metrics.accuracy_score(yTest, KNNpredict))

def SuperVectorMachine(xTrain, xTest, yTrain, yTest):
 ### Support Vector Machines ###
 svmachine = SVC()
 fited = svmachine.fit(xTrain, yTrain)
 score = svmachine.score(xTrain, yTrain)
 SVMpredict = svmachine.predict(xTest)
 return("SVM Accuracy:", metrics.accuracy_score(yTest, SVMpredict))

def DecisionTree(xTrain, xTest, yTrain, yTest):
 ### Decision Tree ###
 dTree = tree.DecisionTreeClassifier()
 fited = dTree.fit(xTrain, yTrain)
 score = dTree.score(xTrain, yTrain)
 DTpredict = dTree.predict(xTest)
 return("Decision Tree Accuracy:", metrics.accuracy_score(yTest, DTpredict))

@app.route('/uploader', methods = ['GET', 'POST'])
```

```
57 def upload_file():
58  if request.method == 'POST':
59      ###LOAD FILE
60      fileLoaded = request.files['file']
61      fileLoaded.save(secure_filename(fileLoaded.filename))
62              ###GET FILE INFO
63      fileSaved = fileLoaded.filename
64      data = pd.read_csv(fileSaved)
65      ###MATRIX
66      arrX = data[data.columns[:-1]].values
67      arrY = data[data.columns[-1]].values
68      X_train,X_test,Y_train,Y_test=train_test_split(arrX,arrY,test_size=0.2,
    random_state=None)
69              ###PREDICTION
70      LRpred = LogisticReg(X_train, X_test, Y_train, Y_test)
71      KNNpred = KNNeighbors(X_train, X_test, Y_train, Y_test)
72      SVMpred = SuperVectorMachine(X_train, X_test, Y_train, Y_test)
73      DTpred = DecisionTree(X_train, X_test, Y_train, Y_test)
74      ###CHOOSE THE BEST PREDICTOR
75      largest = [LRpred, KNNpred, SVMpred, DTpred]
76      sor = sorted(largest)
77      bestPredictor = max(sor)
78      ###SAVE MODEL WITH PICKLE
79      pickle.dump(bestPredictor, open('predictor.sav', 'wb'))
80  return flask.render_template('uploaded.html', bestPredictor=bestPredictor)
81
82 @app.route('/formulario', methods = ['GET','POST'])
83 def prediction():
84  if request.method == 'POST':
85      to_predict_list = request.form.to_dict()
86      to_predict_list = list(to_predict_list.values())
87          to_predict_list = list(map(int, to_predict_list))
88          to_predict = np.array(to_predict_list).reshape(1,6)
89          loaded_model = pickle.load(open("predictor.sav", "rb"))
90      #PREDICTION WITH PICKLE
91      classes = loaded_model.predict(to_predict)
92
93      if (classes == 0):
94                  post_predict = 'LOW'
95      else if(classes == 2):
96                  post_predict = 'HIGH'
97      else if (classes == 1):
98                  post_predict = 'MEDIUM'
99
100 return render_template('predict.html', model = model, post_predict=post_predict)
101
102  if __name__ == '__main__':
103       app.run(debug = True)
```

Appendix B : SOURCE CODE FOR INDEX.HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Index</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/
    css/bootstrap.min.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
    awesome/4.7.0/css/font-awesome.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js
    "></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.
    js"></script>
<style>
body {
 font-family: 'Lato', sans-serif;
}
.upload-btn-wrapper {
  position: center;
  overflow: hidden;
  display: inline-block;
  text-align: center;
}
.btn {
  border: 2px solid gray;
  color: gray;
  background-color: white;
  padding: 8px 20px;
  border-radius: 8px;
  font-size: 20px;
  font-weight: bold;
}
.form{
  text-align: center;
}
.footer{
  position: fixed;
  left: 0;
  bottom: 0;
  width: 100%;
  background-color: gray;
  color: white;
  text-align: center;
}
.navbar {
  width: 100%;
  background-color: gray;
  overflow: auto;
}
.active {
  background-color: #4CAF50;
}
/* Navbar links */
.navbar a {
```

```
53    float: left;
54    text-align: center;
55    padding: 12px;
56    color: white;
57    text-decoration: none;
58    font-size: 17px;
59  }
60
61  /* Navbar links on mouse-over */
66  .navbar a:hover {
67    background-color: #000;
68  }
69  </style>
70
71  </head>
72  <header>
73      <div class="navbar">
74          <a class="active" href="uploaded.html"><i class="fa fa-facebook-square"
    aria-hidden="true"></i> Facebook Post Predictor</a>
75          <a href="#"><i class="fa fa-info" aria-hidden="true"></i></i> More info</a
    >
76          <a href="#"><i class="fa fa-fw fa-user"></i>Contact me</a>
77      </div>
78  </header>
79  <body>
80
81     <form action = "http://localhost:5000/uploader" method = "POST" enctype = "
    multipart/form-data" class="form">
82          <div class="upload-btn-wrapper" >
83              <input type = "file" name = "file" /></br>
84              <button class="btn">Cargar archivo .csv</button>
85              <!-- <input type = "submit"/> -->
86          </div>
87      </form>
88
89      <div class="footer">
90          <p>By Adlay Levy<p>
91      </div>
92  </body>
93  </html>
```

Appendix C : SOURCE CODE FOR UPLOAD.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Prediction</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/
    css/bootstrap.min.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
    awesome/4.7.0/css/font-awesome.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js
    "></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.
    js"></script>
<style>
body {
  font-family: 'Lato', sans-serif;
}
.footer{
  position: fixed;
  left: 0;
  bottom: 0;
  width: 100%;
  background-color: gray;
  color: white;
  text-align: center;
}
.btn {
  border: 2px solid gray;
  color: gray;
  background-color: white;
  padding: 8px 20px;
  border-radius: 8px;
  font-size: 20px;
  font-weight: bold;
}
.predictor {
  position: center;
  overflow: hidden;
  display: inline-block;
  text-align: center;
}
.predict-form-title {
  display: block;
  width: 100%;
  font-size: 39px;
  color: gray;
  line-height: 1.2;
  text-align: center;
  padding-bottom: 59px;
}
.wrap-input100 {
  width: 25%;
  position: center;
  border: 1px solid #e6e6e6;
```

```
53    border-radius: 13px;
54    padding: 10px 30px 9px 22px;
55    margin-bottom: 20px;
56  }
57  .bg1 {background-color: #f7f7f7;}
58  .label-input100 {
59    font-size: 15px;
60    color: #393939;
61    line-height: 1.5;
62    text-transform: uppercase;
63  }
64  .input100 {
65    display: block;
66    width: 80%;
67    background: transparent;
68    font-size: 18px;
69    color: #555555;
70    line-height: 1.2;
71    padding-right: 15px;
72  }
73  .all-center{
74    position: center;
75    display: inline-block;
76  }
77  .form{
78    text-align: center;
79  }
80  .navbar a:hover {
81    background-color: #000;
82  }
83  .navbar a {
84    float: left;
85    text-align: center;
86    padding: 12px;
87    color: white;
88    text-decoration: none;
89    font-size: 17px;
90   }
100  .active {
101    background-color: #4CAF50;
102   }
103  .navbar {
104    width: 100%;
105    background-color: gray;
106    overflow: auto;
107  }
108  </style>

109  </head>
110   <header>
111      <div class="navbar">
112         <a class="active" href="index.html"><i class="fa fa-facebook-square" aria-
    hidden="true"></i> Facebook Post Predictor</a>
113         <a href="#"><i class="fa fa-info" aria-hidden="true"></i></i> More info</a
    >
114         <a href="https://adlyfany8.wixsite.com/curriculum"><i class="fa fa-fw fa-
    user"></i>Contact me</a>
115      </div>
```

```
116 </header>
117 <body>
118    <p> Archivo cargado </p>
119    <form action="http://localhost:5000/uploader" method="POST">
120       <span class="predict-form-title">
121       {{ bestPredictor }}
122       </span>
123     </form>
124     <form action="http://localhost:5000/formulario" method="POST" class="form" >
125   <div class="wrap-input100 bg1 all-center" id="xType" name="xType">
126           <span class="label-input100">Facebook post Type</span>
127           <select id="xType" name="xType" class="input100">
128                   <option value="1">Photo</option>
129                   <option value="2">Multiple photos</option>
130                   <option value="3">Video</option>
131                   <option value="4">Link</option>
132               </select>
133       </div>
134       <div class="wrap-input100 bg1 all-center" id="xDate" name="xDate">
135           <span class="label-input100">Facebook post Date</span>
136           <select id="xDate" name="xDate" class="input100">
137                   <option value="1">Sunday</option>
138                   <option value="2">Monday</option>
139                   <option value="3">Tuesday</option>
140                   <option value="4">Wednesday</option>
141                   <option value="5">Thursday</option>
142                   <option value="6">Friday</option>
143                   <option value="7">Saturday</option>
144               </select>
145       </div>
146       <div class="wrap-input100 bg1 all-center" id="xHour" name="xHour">
147           <span class="label-input100"> Facebook post Hour</span>
148           <select id="xHour" name="xHour" class="input100">
149                   <option value="0">00:00 hrs</option>
150                   <option value="1">01:00 hrs</option>
151                   <option value="2">02:00 hrs</option>
152                   <option value="3">03:00 hrs</option>
153                   <option value="4">04:00 hrs</option>
154                   <option value="5">05:00 hrs</option>
155                   <option value="6">06:00 hrs</option>
156                   <option value="7">07:00 hrs</option>
157                   <option value="8">08:00 hrs</option>
158                   <option value="9">09:00 hrs</option>
160                   <option value="10">10:00 hrs</option>
161                   <option value="11">11:00 hrs</option>
161                   <option value="12">12:00 hrs</option>
167                <option value="13">13:00 hrs</option>
168                   <option value="14">14:00 hrs</option>
169                   <option value="15">15:00 hrs</option>
170                   <option value="16">16:00 hrs</option>
171                   <option value="17">17:00 hrs</option>
172                   <option value="18">18:00 hrs</option>
173                   <option value="19">19:00 hrs</option>
174                   <option value="20">20:00 hrs</option>
175                   <option value="21">21:00 hrs</option>
176                   <option value="22">22:00 hrs</option>
177                   <option value="23">23:00 hrs</option>
178               </select>
```

```
179         </div>
180                 <br>
181                 <span class="label-input100">
182                 TAKING INTO ACCOUNT THE PERFORMANCE OF YOUR PAGE SO FAR,
183                 IN A REALISTIC ENVIRONMENT,
184                 ENTER THE EXPECTED VALUES FOR YOUR NEXT POST
185                 </span>
186                 </br></br>
187                     <div class="wrap-input100 bg1 all-center">
188                      <span class="label-input100">SHARE</span>
189                         <input class="input100" id="xShare" name="xShare"
    placeholder="Enter shares expected">
190                     </div>
191                     <div class="wrap-input100 bg1 all-center" >
192                         <span class="label-input100">REACTIONS</span>
193                         <input class="input100" id="xReactions" name="xReactions"
    placeholder="Enter reactions expected">
194                     </div>
195                     <div class="wrap-input100 bg1 all-center" >
196                         <span class="label-input100">COMMENTS</span>
197                         <input class="input100" id="xComment" name="xComment"
    placeholder="Enter comments expected">
198                     </div>
199                     </br>
200                     <div class="all-cecnter">
201                         <input type="submit" value="Submit" class="btn">
202                         <!--<button class="btn" type="submit">Predecir</button>-->
203                     </div>
204         </form>
205         <div class="footer">
206         <p>By Adlay Levy<p>
207         </div>
208     </body>
209 </html>
```