# Predicting the client's purchasing intention using Machine Learning models

*S. AHSAIN* [1,*] and *M. AIT KBIR* [1]

[1] LIST laboratory, STI Doctoral Center, University Abdelmalek Essaadi, Morocco

**Abstract.** In this paper, we introduce a prediction algorithm that will determine the likelihood that a client will purchase from a website or not. This system is part of a global e-commerce solution that will help the clients to get the best possible experience. The paper presents an overview of the e-commerce system's various components and their various steps and also an activity diagram of the system, which shows the various steps that the platform can perform. It also provides a general idea of the system's workflow.

**Keywords**. Digital Marketing, Innovation, Machine Learning, e-commerce platform.

## 1 Introduction

Today, most domains are focused on anticipating and acting on future events. E-commerce platforms are very popular among consumers due to the existence of advanced technologies such as artificial intelligence. Among the key causes Data Mining and Machine learning have become an integral part of the Customer Relationship Management, is to replicate the close relationship that is emerging between small businesses and the purchaser. The goal is to find out what sets each customer apart and build trust between the two parties [1].

This paper is the continuation of [1], where we provided a general idea about the current state of art in terms of Machine Learning and Data Mining in the Digital Marketing domain. This paper presents the prediction of the likelihood that a client will pay for a product before leaving the Platform, the dataset used in this problematic was introduced by [2].

The paper is structured as follows, the first part explains the e-commerce platform's logical component, how it contributes to the platform's overall operations and how it will help the clients to get the best possible experience. The second part tackles the predictions of the likelihood that a user would purchase from the platform. This module can be used in a variety of e-commerce platforms. It can predict the best products in the platform, or it can introduce new features and concepts related to the consumer decision journey.

This paper represents a global view that can be used in different domains and not only in a company context.

## 2 Proposed system

### 2.1 Logical components of an e-commerce platform

In order to develop an adequate system that can meet the demand of the 2.0 clients, the system we propose is based on a package diagram developed in [3] that talks about the various components of a real-world e-commerce system.

The authors focused on the various aspects of designing an e-commerce plat-form that would enable them to handle various types of data. Some of these included data cleansing, data migration, and data customization [1].

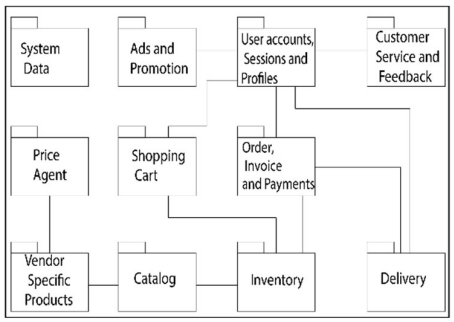


**Fig. 1.** Package diagram with logical components of e-commerce systems [3].

The UASP package is responsible for the management of the user accounts and the customization of the platform's user experience. It also acts as the

* Corresponding author: sara.ahsain@etu.uae.ac.ma AND m.aitkbir@fstt.ac.ma

central repository for the system's reports and functionalities.

The authors treat the various modules as interrelated. The diagram aims to lay out the foundations for an e-commerce system. Nevertheless, these packages can be developed further, added to or split into other modules.

This paper presents an approach that enables to have only one simple function per package. It will then interact with the others according to the user's needs.

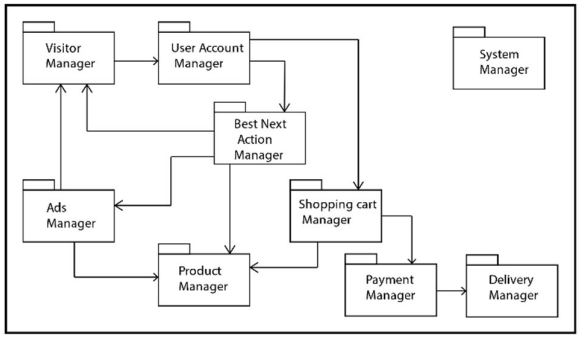


**Fig. 2.** Package diagram of the proposed system.

*Visitor manager* is designed to handle the temporary visit of a client who doesn't have an account. It sends him to the appropriate page in case of purchase. The visitor sees the data adapted from the Ads manager for the best rated products.

*User account manager* is used for creating a new user account, logging in, and storing history. It also manages the purchase history. This module communicates directly with **Best Next Action component** and **Shopping cart manager**.

*Best Next Action manager* uses the data collected from the user account to provide the best possible user experience. It sends the most relevant products and services to the user based on his usual activities and behavior.

*Product manager* depends on the **Ads manager** and on the **Best Next Action manager** in order to show the products that will interest the user.

*Ads manager* sends data to the **Best Next Action module** in order to show the ads that interest the user most.

*Shopping cart manager* is responsible for the products a user adds to their cart.

*Payment manager* is responsible for the entire payment process after the user validates his cart.

*Delivery manager* is a tool that tracks the progress of the delivery process. It sends notifications when the user has finished their payment and on every step of the shipping.

### 2.2 Activity diagram of the proposed system

This section shows the interaction between the various modules of the platform. It shows how the various components interact with each other to provide a better and more complete user experience.

In general, the activity diagrams' generic idea of the e-commerce platforms are similar (Ex: [4]).

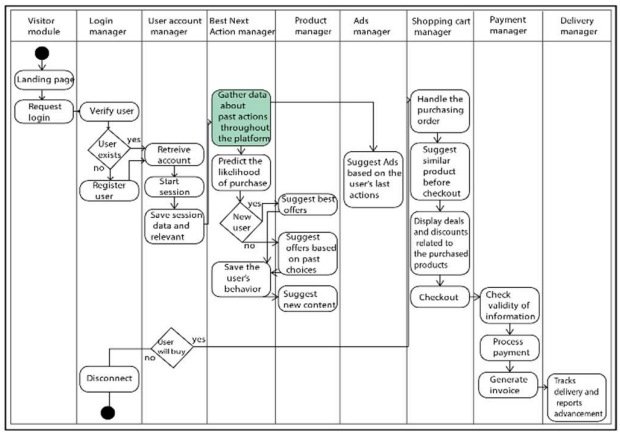


**Fig. 3.** The activity diagram of the predicting system based on the client's purchasing.

The activity diagram is Fig 3, The flow is initiated once a potential client browses the website and completes a purchase or registration. It is usually triggered by a visit to the website's homepage.

In Best Next Action module (BNMA), the data collected by the account manager is used to create the best possible user experience for each individual user using past behaviors with Machine Learning Algorithm. The suggestions and ads are also handled throughout the module. The operating of Best Next Action Module was detailed in the previous section. The user can customize the items he wants to buy from the cart, and then proceed with the payment. After the purchase, the user can add the billing address, visa card data and track his order.

## 3 Predicting the likelihood of the client's buying from the platform

### 3.1 Dataset description

We worked on the dataset proposed by the paper published by [2]. In this paper, the authors formulated a model that predicts the likelihood of abandonment and the purchasing intention of a user/session.

The dataset is composed of data gathered from 12,330 sessions; each session belongs to a new user to avoid any influence on the model. The target variable is a binary value describing whether the platform had any revenue from the session.

**Table 1.** Features names and types.

| Feature name | Type |
|---|---|
| Administrative | Numerical |
| Administrative duration | Numerical |
| Informational | Numerical |
| Informational duration | Numerical |
| Product related | Numerical |
| Product related duration | Numerical |
| Bounce rates | Numerical |
| Exit rates | Numerical |
| Page values | Numerical |
| Special day | Numerical |
| Month | Categorical |
| Operating systems | Categorical |
| Browser | Categorical |
| Region | Categorical |
| Traffic type | Categorical |
| Visitor type | Categorical |
| Weekend | Categorical |
| Revenue | Categorical |

The dataset has ten numerical features and eight categorical attributes. Some features' data is originated from the URL routes of the pages visited and is updated simultaneously when an action takes place. The dataset also includes operating system, browser, region, traffic type, visitor type as returning or new visitor, a Boolean value indicating whether the date of the visit is weekend, and month of the year [2].

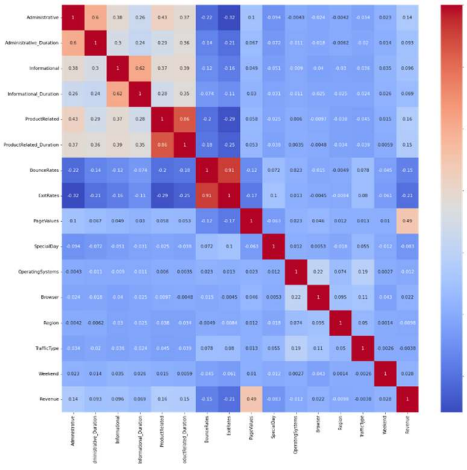

**Fig. 4.** Correlation Heatmaps with Seaborn & Matplotlib

As an exploratory data analysis tool, this correlation matrix contain same important visual information. In fact, It shows that features are not very correlated and no potential multicollinearity problem was detected.

### 3.2 Prediction

In this study, we used the features offered by the Python library PyCaret [5]. It is a Python library that provides a complete set of features for creating and manipulating machine learning models [6].

After getting the data and setting up the PyCart Environment, the library compiles the data and allows the user to compare different supervised or unsupervised models. It takes into account the most interesting results. The best algorithms are chosen based on various metrics such as Accuracy, Recall, etc. and presented in a table. The output prints a score grid that shows the average of the Accuracy, AUC, Recall, Precision, F1, Kappa and MCC across the folds (10 folds by default) along with the training times. Before using the data with PyCaret, it was cleaned and split in train and test data:

```
df.replace({True:1,False:0},inplace=True)

 dummies =
pd.get_dummies(df.drop(columns='Month'))

data = dummies.drop(columns='Revenue')

target = dummies['Revenue']
```

The following results were obtained using the dataset mentioned above using the *compare_models()* method.

```
model_setup = setup(data=data, target='Revenue',
session_id=123)

best_model = compare_models()
```

| | Model | Accuracy | AUC | Recall | Prec. | F1 | Kappa | MCC |
|---|---|---|---|---|---|---|---|---|
| LIGHT GBM | Light Gradient Boosting Machine | 0.8989 | 0.9300 | 0.5882 | 0.7062 | 0.6409 | 0.5827 | 0.5865 |
| GBC | Gradient Boosting Classifier | 0.8988 | 0.9307 | 0.5906 | 0.7054 | 0.6418 | 0.5834 | 0.5872 |
| RF | Random Forest Classifier | 0.8973 | 0.9214 | 0.5016 | 0.7497 | 0.5995 | 0.5433 | 0.5586 |
| ADA | Ada Boost Classifier | 0.8862 | 0.9152 | 0.5476 | 0.6590 | 0.5960 | 0.5307 | 0.5350 |
| LR | Logistic Regression | 0.8841 | 0.8970 | 0.3870 | 0.7308 | 0.5051 | 0.4467 | 0.4761 |
| LDA | Linear Discriminant Analysis | 0.8793 | 0.8959 | 0.3489 | 0.7195 | 0.4687 | 0.4100 | 0.4448 |
| ET | Extra Trees Classifier | 0.8733 | 0.8861 | 0.2997 | 0.7106 | 0.4195 | 0.3615 | 0.4015 |
| RIDGE | Ridge Classifier | 0.8721 | 0.0000 | 0.2416 | 0.7595 | 0.3652 | 0.3151 | 0.3802 |
| DT | Decision Tree Classifier | 0.8642 | 0.7415 | 0.5644 | 0.5602 | 0.5614 | 0.4812 | 0.4817 |
| KNN | K Neighbors Classifier | 0.8571 | 0.7579 | 0.2837 | 0.5677 | 0.3777 | 0.3072 | 0.3311 |
| SVM | SVM – Linear Kernel | 0.8486 | 0.0000 | 0.3527 | 0.7548 | 0.4101 | 0.3476 | 0.4166 |
| NB | Naive Bayes | 0.7631 | 0.7848 | 0.6326 | 0.3495 | 0.4500 | 0.3147 | 0.3375 |
| QDA | Quadratic Discriminant Analysis | 0.2761 | 0.5032 | 0.8307 | 0.1547 | 0.2607 | 0.0027 | 0.0047 |

**Fig. 5.** Results of PyCaret's comparing models' method.

The best models based on PyCaret with the best metrics are:

- **Light Gradient Boosting Machine – LIGHTGBM** is a gradient boosting framework based on decision trees, it increases the efficiency of the model and reduces memory usage [7],
- **Gradient Boosting Classifier – GBC** combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting [8],
- **Random Forest Classifier – RF** consists of a large number of individual decision trees that operate as an ensemble. Each tree spits out a class prediction and the class with the most votes become the model's

prediction [9] RF can handle high dimensional data and use a large number of trees in the ensemble [10].

### 3.3 Fine-tuned prediction

PyCaret allows the users to train and make predictions about individual models. It can also be tuned to get better results; It can also automatically tune the hyper-parameters of a model using the Random Grid Search.

*rf = create_model('rf')*
*tuned_rf = tune_model(rf)*

| Model | Accuracy | AUC | Recall | Prec. | F1 | Kappa | MCC |
|---|---|---|---|---|---|---|---|
| **LIGHT GBM** (hyperparameter 8) | 0.9061 | 0.9355 | 0.6111 | 0.7333 | 0.6667 | 0.6125 | 0.6160 |
| **Tuned lightgbm** (hyperparameter 4) | 0.9085 | 0.9424 | 0.5952 | 0.7576 | 0.6667 | 0.6145 | 0.6205 |
| **GBC** (hyperparameter 9) | 0.9061 | 0.9401 | 0.6190 | 0.7290 | 0.6695 | 0.6152 | 0.6180 |
| **Tuned GBC** (hyperparameter 6) | 0.9073 | 0.9422 | 0.5952 | 0.7500 | 0.6637 | 0.6108 | 0.6163 |
| **RF** (hyperparameter 9) | 0.9110 | 0.9348 | 0.5635 | 0.7978 | 0.6605 | 0.6110 | 0.6232 |
| **Tuned RF** (hyperparameter 10) | 0.9060 | 0.9159 | 0.6800 | 0.6967 | 0.6883 | 0.6329 | 0.6330 |

**Fig. 6.** Representation of the results from the three top algorithms based on PyCaret.

We notice satisfying results of the trained models in this paper by PyCaret using the dataset we have discussed above. the three top models discussed in the first step have near values. The tuned Random Forest model scored the best results with a 91% Accuracy score before tuning the hyper-parameters when used with the sessions dataset.

## 4 Conclusion

Due to the convenience of shopping online, many people prefer to do business online. This is also beneficial for the retailers as it allows them to reach out to a wider customer base and offer various products at lower prices [11].

This paper aims to provide a global view of the various components that a platform can use to reinforce its consumer decision journey. It shows how the various components can be utilized to enhance the platform's effectiveness.

The prediction module can be used in various aspects of a website, such as prediction of the user's future purchase. The goal of this paper is to develop a platform that will allow the customer to act on their needs instantly.

## References

1. S. Ahsain, M. Ait Kbir, Data *Mining and Machine Learning Techniques Applied to Digital Marketing Domain Needs*, in Proceedings of the Third International Conference on Smart City Applications SCA, Innovations in Smart Cities Applications **4** (2021)
2. CO. Sakar, SO. Polat, M. Katircioglu, Y. Kastro, *Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural net-works,* in Neural Computing and Applications, **31**, 6893 (2019)
3. I-Y. Song, K-Y. Whang, T. Korea, *Database Design for Real-World E-Commerce Systems*, in TCDE of the IEEE Computer Society **23**, 1 (2000)
4. K. Baati, M. Mohsil, *Real-Time Prediction of Online Shoppers' Purchasing Intention Using Random Forest*, in IFIP **583**, 43 (2020)
5. PyCaret. In: PyCaret. https://pycaret.org/, accessed 15 July 2021
6. Creating the Whole Machine Learning Pipeline with PyCaret (https://www.datasource.ai/en/data-science-articles/creating-the-whole-machine-learning-pipeline-with-pycaret) accessed 15 July 2021
7. LightGBM (Light Gradient Boosting Machine). In: GeeksforGeeks( https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/, accessed 15 July 2021
8. Gradient Boosting Classifiers in Python with Scikit-Learn (https://stackabuse.com/gradient-boosting-classifiers-in-python-with-scikit-learn), accessed 15 July 2021
9. Understanding Random Forest (https://towardsdatascience.com/understanding-random-forest-58381e0602d2), accessed 15 Jul 2021
10. L. Breiman, *Random forests*, in Machine learning no 1, **45,** 5 (2001)
11. M. Mud, R. Iswari, M. Ahsan M, *Prediction of Online Shopper's Purchasing Intention Using Binary Logistic Regression, Decision Tree, and Random Forest*, in AIAI no. 583, 43 (2020)