

Performance analysis of machine learning algorithms in credit cards fraud detection

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

B YASHWANTH REDDY(Reg.No.37110079)
B CHAITANYA GUPTA(Reg.No.37110100)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU**

AUGUST 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **B. YASHWANTH REDDY (37110079)** and **B. CHAITANYA GUPTA (37110100)** who carried out the project entitled “**Performance analysis of machine learning algorithms in credit cards fraud detection**”

” under my supervision from December 2020 to March 2021.

Internal Guide

M Maheswari, M.E, Ph.D

Head of the Department

Dr. S.VIGNESHWARI , M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **B. YASHWANTH REDDY (37110079) and B. CHAITANYA GUPTA (37110100)**

hereby declare that the Project Report entitled **Performance analysis of machine learning algorithms in credit cards fraud detection** done by me under the guidance of Dr. /Prof. / Mr. /Ms. **M Maheswari, M.E, Ph.D** (Internal) and (External) at_____(Company name and address) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing , **Dr.S.Vigneshwari M.E., Ph.D., and Dr.L.Lakshmanan M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **M Maheswari M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of **the Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The use of credit cards is prevalent in modern day society. But it is obvious that the number of credit card fraud cases is constantly increasing in spite of the chip cards worldwide integration and existing protection systems. This is why the problem of fraud detection is very important now. The credit card fraud features uses user behavior scanning to check for unusual patterns. These patterns include user characteristics such as user spending patterns to verify his identity. In fraudsters are increasing day by day. And fallacious transactions are done by the credit card and there are various types of fraud. So to solve this problem combination of technique is used like Random Forest Classifier, AdaBoost classifier, CatBoost classifier, XGBoost, LightGBM, Machine learning. By this transaction is tested individually and whatever suits the best is further proceeded. And the foremost goal is to detect fraud by filtering the above techniques to get better result. In this project the general description of the developed fraud detection system and comparisons between models based. In the last section of this paper the results of evaluative testing and corresponding conclusions are considered.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	i
	LIST OF FIGURES	v
	LIST OF ABBREVIATIONS	v
1	INTRODUCTION	1
	1.1 PROBLEM DEFINITION	1
2	AIM AND SCOPE	2
	2.1 PURPOSE	2
	2.2 SCOPE	2
	2.3 EXISTING SYSTEMS	2
	2.4 DISADVANTAGES OF EXISTING SYSTEM	3
	2.5 PROPOSED SYSTEM	3
	2.6 ADVANTAGES OF PROPOSED SYSTEM	3
	2.7 SYSTEM REQUIREMENTS	4
	2.7.1 HARDWARE REQUIRMENTS	4
	2.7.2 SOFTWARE REQUIRMENTS	4
3	METHODS	5

	3.1 PYTHON	5
	3.2 FEASIBILITY STUDY	7
	3.2.1 ECONOMICAL FEASIBILITY	8
	3.2.2 TECHNICAL FEASIBILITY	8
	3.2.3 SOCIAL FEASIBILITY	8
4	SYSTEM DESIGN	9
	4.1 INPUT DESIGN	9
	4.2 OUTPUT DESIGN	9
	4.3 DATA FLOW DIAGRAM	10
	4.3.1 ROLE OF DFD	10
	4.4 SYSTEM ARCHITECTURE	11
5	SYSTEM TESTING	13
	5.1 TEST PLAN	13
	5.2 VERIFICATION	13
	5.3 VALIDATION	13
	5.4 BASIC OF SOFTWARE TESTING	13
	5.5 BLACK BOX TESTING	13
	5.6 WHITE BOX TESTING	14
	5.7 TYPES OF TESTING	14
	5.7.1 UNIT TESTING	14
	5.7.2 INTEGRATION TESTING	14
	5.7.3 FUNCTIONAL TESTING	15

	5.7.4 SYSTEM TESTING	15
	5.7.5 STRESS TESTING	15
	5.7.6 PERFORMANCE TESTING	15
	5.7.7 USABILITY TESTING	15
	5.7.8 ACCEPTANCE TESTING	16
	5.7.9 REGRESSION TESTING	16
	5.8 MODULES	16
	5.8.1 COLLECTION OF DATA	16
	5.8.2 PREDICTIVE MODELS	16
6	RESULTS AND DISCUSSIONS	17
7	CONCLUSION AND FUTURE ENHANCEMENT	18
	7.1 CONCLUSION	18
	7.2 FUTURE ENHANCEMENT	18
	7.3 REFERENCES	18
	APPENDIX	21
	SCREEN SHOT	21
	SOURCE CODE	24

LIST OF FIGURES

FIGURE No	FIGURE NAME	PAGE NO
4.3.1	DATA FLIOW DIAGRAM	11
4.4	SYSTEM ARCHITECTURE	12

LIST OF ABBREVATIONS

XG BOOST - EXTREME GRADIENT BOOSTING

ADABOOST -ADAPTIVE BOOST

CHAPTER 1

INTRODUCTION

In recent years, the prevailing data mining concerns people with credit card fraud detection model based on data mining. Since our problem is approached as a classification problem, classical data mining algorithms are not directly applicable. So an alternative approach is made by using general purpose approaches like random forest classifier. This project is to propose a credit card fraud system using different types of algorithms.

We are using evolutionary algorithms which aim at obtaining better solutions as time progresses. When a card is copied or stolen or lost and captured by fraudsters it is usually used until its available limit is depleted. Thus, rather than the number of correctly classified transactions, a solution which minimizes the total available limit on cards subject to fraud is more prominent. It aims in minimizing the false alerts using Random forest algorithm where a set of interval valued parameters are optimized.

1.1 PROBLEM DEFINITION

To develop a credit card fraud detection system using different types of algorithms. During the credit card transaction, the fraud is detected and the number of false alert is being minimized by using algorithms. Instead of maximizing the numbers of correctly classified transactions we defined an objective function where the misclassification costs are variable and thus, correct classification of some transactions are more important than correctly classifying the others.

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be fraud. This model is then used to identify whether a new transaction is fraudulent or not. Our aim here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications.

CHAPTER 2

AIM and SCOPE

2.1 PURPOSE

The purpose of this document is to define the requirements of credit card fraud detection. In detail, this document will provide a general description of our project, including user requirements, product perspective, and overview of requirements, general constraints. In addition, it will also provide the specific requirements and functionality needed for this project - such as interface, functional requirements and performance requirements.

2.2 SCOPE

The scope of this SRSdocument persists for the entire life cycle of the project. This document defines the final state of the software requirements agreed upon by the customers and designers. Finally at the end of the project execution all the functionalities may be traceable from the SRSto the product. The document describes the functionality, performance, constraints, interface and reliability for the entire life cycle of the project.

2.3 EXISTING SYSTEMS

- In case of credit card fraud detection the existing system is detect the fraud after fraud has been happen. Existing system maintain the large amount of data when customer comes to know about inconsistency in transaction he/she made complaint and then fraud detection system start it working. It first tries to detect that fraud has actually occur after that it transactions that was used to fraud detection mechanism developed by master and visa cards
- A machine learning paradigm classification, with Credit Card Fraud Detection being the base.
- Intrusion detections to track fraud location and so on. In case of existing system there is no confirmation of recovery of fraud and Customer satisfaction.
- Secure electronic system used to analyze the behaviour of legitimate users.
- Data Mining mechanisms to classify and preprocess the users data.
- Genetic algorithms

2.4 DISADVANTAGES OF EXISTING SYSTEM

- Each payment system has its limits regarding the maximum amount in the account, the number of transactions per day and the amount of output.
- If Internet connection fails, you cannot get to your online account.
- If you follow the security rules the threat is minimal. The worse situation when the system of processing company has been broken, because it leads to the leak of personal data on cards and its owners.
- The information about all the transactions, including the amount, time and recipient are stored in the database of the payment system. And it means the intelligence agency has an access to this information. Sometimes this is the path for fraudulent activities

2.5 PROPOSED SYSTEM

- The aim of the proposed system is to develop a application which has capability to restrict and block the transaction performing by attacker from genuine user's credit card details.
- As we seen the existing system detects the fraud after fraud has been occurred. The proposed system tries to detect fraudulent transaction in a better way.
- In proposed system we are using different types of algorithms, which works on transaction behavior of user. After certain transactions we find one value by using this value we can compare current transaction with value if is quite different from user behavior then check whether it is genuine or fraud.

2.6 ADVANTAGES OF PROPOSED SYSTEM

- Online alerts are available if suspicious activity is detect on your card. The issuer will notify if any unusual charges come through on the card.
- Credit scores are used in numerous ways and when a person uses a credit card responsibly and makes timely payments without over limits or late payments, their credit score rises.
- By making payments on time and keeping the balance low on the card, a person will save money on the amount of interest charged by the card issuer.
- Eco friendly, no personal presence is required, it treats with growing technology, time saving.

2.7 SYSTEM REQUIREMENTS

2.7.1 HARDWARE REQUIRMENTS

System : Pentium IV 2.4GHZ
 Hard Disk : 40 gb

Ram : 512 Mb

2.7.2 SOFTWARE REQUIREMENTS

Operating system : Window 7(32 bit)
Coding Language : python
Tool : jupyter

CHAPTER 3

METHODS

3.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include –

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python <https://www.python.org>.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to <https://www.python.org/downloads/>.

- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

3.2 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

3.2.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must

justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.2.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.2.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

CHAPTER 4

SYSTEM DESIGN

4.1 INPUT DESIGN

The input design is the link between the information system and the user. It

comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

4.2 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action

4.3 DATA FLOW DIAGRAM

Data Flow Diagram (DFD) is a two-dimensional diagram that describes how data is processed and transmitted in a system. The graphical depiction recognizes

each source of data and how it interacts with other data sources to reach a mutual output. In order to draft a data flow diagram one must

- Identify external inputs and outputs
- Determine how the inputs and outputs relate to each other
- Explain with graphics how these connections relate to what they result in.

4.3.1 Role of DFD:

- It is a documentation support which is understood by both programmers and nonprogrammers. As DFD postulates only what processes are accomplished not how they are performed.
- A physical DFD postulates where the data flows and who processes the data.
- It permits analyst to isolate areas of interest in the organization and study them by examining the data that enter the process and viewing how they are altered when they leave.

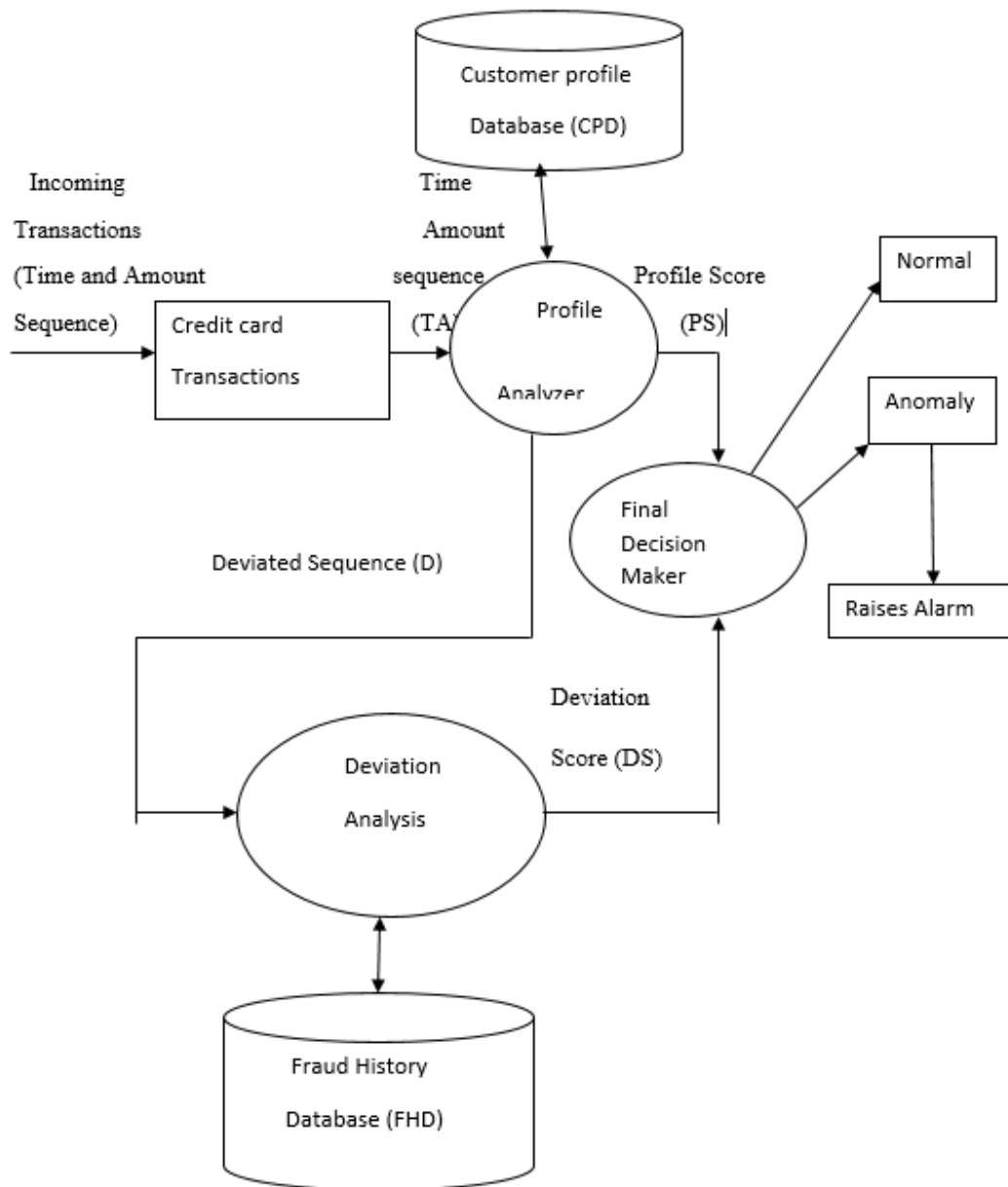


Fig.4.3.1 Data Flow Diagram

4.4 SYSTEM ARCHITECTURE

Describing the overall features of the software is concerned with defining the requirements and establishing the high level of the system. During architectural design, the various web pages and their interconnections are identified and designed. The major software components are identified and decomposed into processing modules and conceptual data structures and the interconnections among

the modules are identified. The following modules are identified in the proposed system.

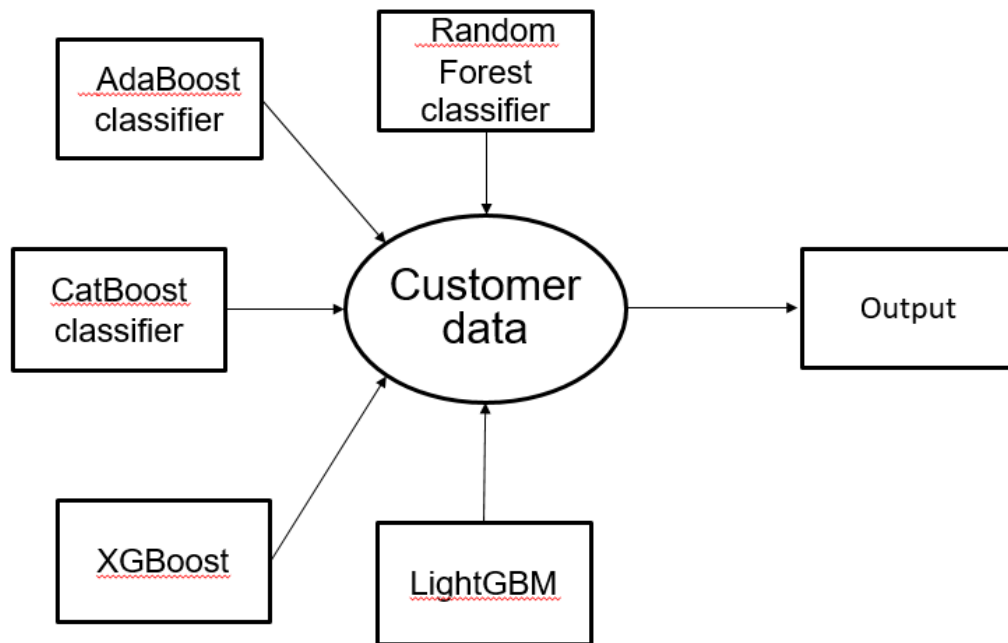


Fig.4.4 SYSTEM ARCHITECTURE

CHAPTER 5

SYSTEM TESTING

5.1 Test plan

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

5.2 Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

5.3 Validation

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

5.4 Basics of software testing

There are two basics of software testing: black box testing and white box testing.

5.5 Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

5.6 White box Testing

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

5.7 Types of testing

There are many types of testing like

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing
- Regression Testing
- Beta Testing

5.7.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under

the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

5.7.2 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

5.7.3 Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

5.7.4 System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

5.7.5 Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

5.7.6 Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

5.7.7 Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

5.7.8 Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

5.7.9 Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing

5.8 MODULES

- Collection of data
- Predictive Models

5.8.1 Collection of data

- Collection of data and verifying the number of data to be executed.
- Checking whether any of the data is missing from the collected data.

5.8.2 Predictive Models

- Random Forest classifier
- AdaBoost classifier
- CatBoost classifier
- XGBoost
- LightGBM

CHAPTER 6

RESULTS AND DISCUSSIONS

- The code prints out the number of false positives it detected and compares it with the actual values. This is used to calculate the accuracy score and precision of the algorithms.
- The fraction of data we used for faster testing is 25% of the entire dataset.
- The fraction of data we used for faster testing is 75% of the entire dataset.
- The complete dataset is also used at the end and both the results are printed.
- These results along with the classification report for each algorithm is given in the output as follows, where class 0 means the transaction was determined to be valid and 1 means it was determined as a fraud transaction.
- This result matched against the class values to check for false positives

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

In this paper, we have discussed that how Random Forest algorithm will be useful to detect fraudulent transaction through credit card. Proposed Fraud detection system gives genuine and fast result than existing system. The proposed Fraud Detection System is also scalable for handling vast volumes of transactions data processing.

7.2 FUTURE ENHANCEMENT

We have discussed an application of fraud in credit card fraud detection. The different steps in credit card transaction processing are represented as the underlying stochastic process of fraud. We have study the ranges of transaction amount as the observation symbols, whereas the types of item have been considered to be states of the fraud. We have study the suggested a method for finding the spending profile of cardholders, as well as application of this knowledge

in deciding the value of observation symbols and initial estimate of the model parameters. It has also been explained how the different algorithms can detect whether a transaction is fraudulent or not. The system is also scalable for handling large volumes of transactions.

7.3 REFERENCES

- [1] M. Kavitha and M. Suriakala, "Hybrid Multi-Level Credit Card Fraud Detection System by Bagging Multiple Boosted Trees (BMBT)," 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Coimbatore, 2017, pp. 1-5.
- [2] John O. Awoyemi; Adebayo O. Adetunmbi; Samuel A. Oluwadare," Credit card fraud detection using machine learning techniques: A comparative analysis,"2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Lagos, Nigeria,pp.- 29-31 Oct. 2017.
- [3] C. Wang, Y. Wang, Z. Ye, L. Yan, W. Cai and S. Pan, "Credit Card Fraud Detection Based on Whale Algorithm Optimized BP Neural Network," 2018 13th International Conference on Computer Science & Education (ICCSE), Colombo, 2018, pp. 1-4.
- [4] J. Han, J. Pei, and M. Kamber, Data mining: concepts and techniques, Elsevier, 2011
- [5] F. Ghobadi and M. Rohani, "Cost sensitive modeling of credit card fraud using neural network strategy," 2016 2nd International Conference of Signal Processing and Intelligent Systems (ICSPIS), Tehran, 2016, pp. 1-5.
- [6] H. Wang, P. Zhu, X. Zou and S. Qin, "An Ensemble Learning Framework for Credit Card Fraud Detection Set Partitioning and Clustering," 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart

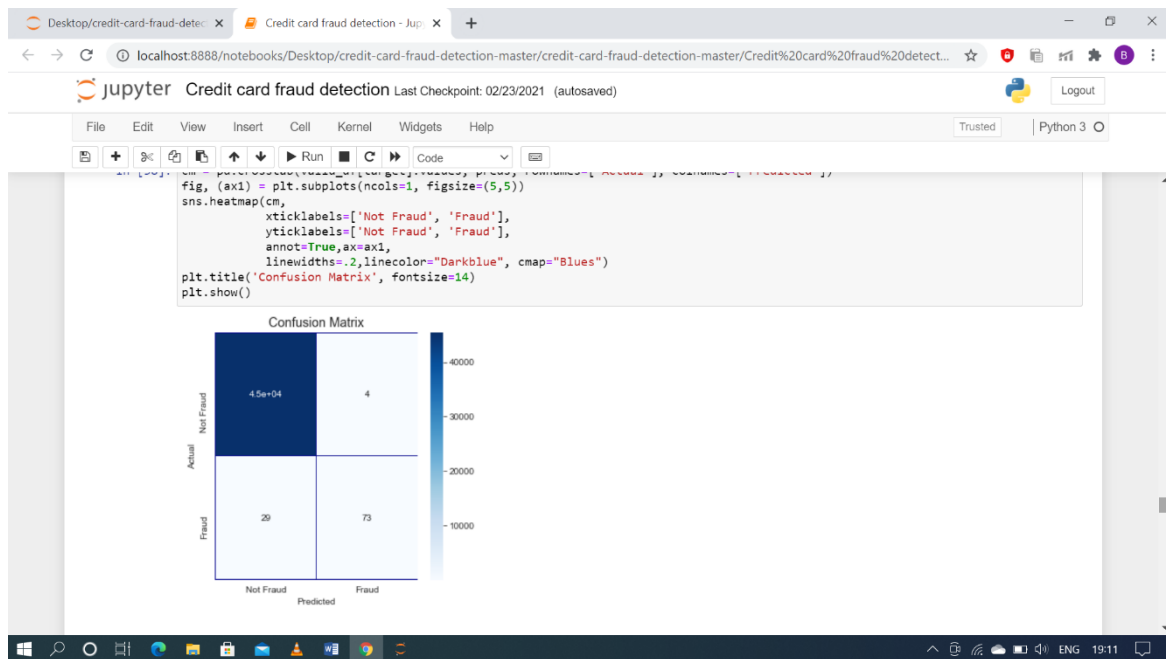
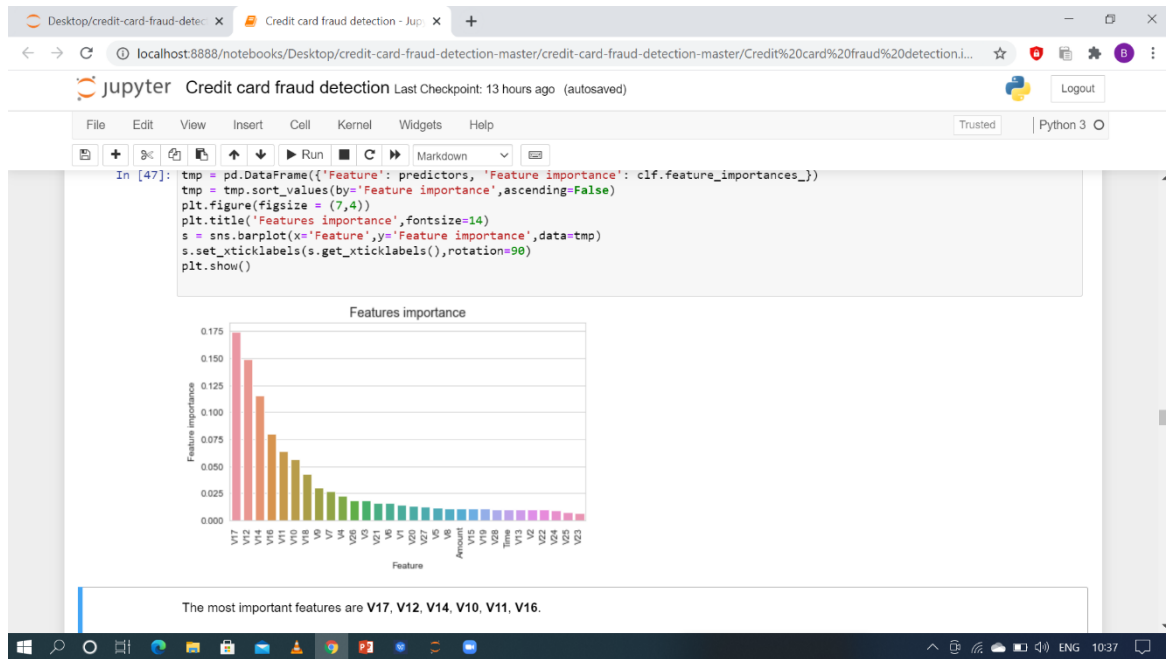
CityInnovation(SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI),
Guangzhou, 2018, pp. 94-98.

- [7] N. K. Gyamfi and J. Abdulai, "Bank Fraud Detection Using Support Vector Machine," 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, 2018, pp. 37-41.
- [8] "Credit Card Fraud Detection Based on Transaction Behaviour -by John Richard D. Kho, Larry A. Veal" published by Proc. of the 2017 IEEE Region 10 Conference (TENCON), Malaysia, November 5-8, 2017.
- [9] "Credit Card Fraud Detection-by Ishu Trivedi, Monika, Mrigya, Mridushi" published by International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 1, January 2016.
- [10] "Survey Paper on Credit Card Fraud Detection by Suman" , Research Scholar, GJUS&T Hisar HCE, Sonapat published by International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 3 Issue 3, March 2014.
- [11] K. Modi and R. Dayma, "Review on fraud detection methods in credit card transactions, "International Conference on Intelligent Computing and Control (I2C2), Coimbatore, pp. 1-5, 2017.
- [12] Chang C. & Chang S. (2010). "The Design of E-Traveler's Check with efficiency and Mutual Authentication". Journal of Networks, Vol. 5, No. 3, pp. 275-282.
- [13] E. W. T. Ngai, Y. Hu, Y. H. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: a classification framework and an academic review of literature," Decision Support Systems, vol. 50, no. 3, pp. 559–569, 2011.
- [14] J. West and M. Bhattacharya, "Intelligent financial fraud detection: a comprehensive review," Computers & Security, vol. 57, pp. 47–66, 2016.
- [15] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2016.
- [16] M. Zanin, D. Papo, P. A. Sousa et al., "Combining complex networks and data mining: why and how," Physics Reports, vol. 635, pp. 1–44, 2016.

- [17] R. Patidar and L. Sharma, “Credit card fraud detection using neural network,” *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 1, 2011.
- [18] E. M. Carneiro, L. A. V. Dias, A. M. D. Cunha, and L. F. S. Mialaret, “Cluster analysis and artificial neural networks: a case study in credit card fraud detection,” in *Proceedings of the 12th International Conference on Information Technology: New Generations, ITNG 2015*, pp. 122–126, IEEE, April 2015.
- [19] M. Zanin, P. A. Sousa, and E. Menasalvas, “Information content: Assessing meso-scale structures in complex networks,” *Europhysics Letters*, vol. 106, article 30001, 2014.
- [20] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, “Data mining for credit card fraud: a comparative study,” *Decision Support Systems*, vol. 50, no. 3, pp. 602–613, 2011.

APPENDIX

SCREENSHOTS



SOURCE CODE

```
import pandas as pd

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

import plotly.graph_objs as go

import plotly.figure_factory as ff

from plotly import tools

from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot

init_notebook_mode(connected=True)

import gc

from datetime import datetime

from sklearn.model_selection import train_test_split

from sklearn.model_selection import KFold

from sklearn.metrics import roc_auc_score

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import AdaBoostClassifier

from catboost import CatBoostClassifier

from sklearn import svm

import lightgbm as lgb

from lightgbm import LGBMClassifier
```

```

import xgboost as xgb

pd.set_option('display.max_columns', 100)

RFC_METRIC = 'gini' #metric used for RandomForestClassifier

NUM_ESTIMATORS = 100 #number of estimators used for RandomForestClassifier

NO_JOBS = 4 #number of parallel jobs used for RandomForestClassifier

#TRAIN/VALIDATION/TEST SPLIT

#VALIDATION

VALID_SIZE = 0.20 # simple validation using train_test_split

TEST_SIZE = 0.20 # test size using_train_test_split

#CROSS-VALIDATION

NUMBER_KFOLDS = 5 #number of KFold for cross-validation

RANDOM_STATE = 2018

MAX_ROUNDS = 1000 #lgb iterations

EARLY_STOP = 50 #lgb early stop

OPT_ROUNDS = 1000 #To be adjusted based on best validation rounds

VERBOSE_EVAL = 50 #Print out metric result

IS_LOCAL = False

import os

data_df = pd.read_csv("creditcard.csv")

print("Credit Card Fraud Detection data - rows:",data_df.shape[0]," columns:",
data_df.shape[1])

data_df.head()

data_df.describe()

total = data_df.isnull().sum().sort_values(ascending = False)

```

```

percent = (data_df.isnull().sum()/data_df.isnull().count()*100).sort_values(ascending
= False)

pd.concat([total, percent], axis=1, keys=['Total', 'Percent']).transpose()

temp = data_df["Class"].value_counts()

df = pd.DataFrame({'Class': temp.index,'values': temp.values})

trace = go.Bar(

    x = df['Class'],y = df['values'],

    name="Credit Card Fraud Class - data unbalance (Not fraud = 0, Fraud = 1)",

    marker=dict(color="Red"),

    text=df['values']

)

data = [trace]

layout = dict(title = 'Credit Card Fraud Class - data unbalance (Not fraud = 0, Fraud =
1)',

    xaxis = dict(title = 'Class', showticklabels=True),

    yaxis = dict(title = 'Number of transactions'),

    hovermode = 'closest',width=600

)

fig = dict(data=data, layout=layout)

iplot(fig, filename='class')

class_0 = data_df.loc[data_df['Class'] == 0]["Time"]

class_1 = data_df.loc[data_df['Class'] == 1]["Time"]

#plt.figure(figsize = (14,4))

#plt.title('Credit Card Transactions Time Density Plot')

```

```

#sns.set_color_codes("pastel")

#sns.distplot(class_0,kde=True,bins=480)

#sns.distplot(class_1,kde=True,bins=480)

#plt.show()

hist_data = [class_0, class_1]

group_labels = ['Not Fraud', 'Fraud']

fig = ff.create_distplot(hist_data, group_labels, show_hist=False, show_rug=False)

fig['layout'].update(title='Credit Card Transactions Time Density Plot',
axis=dict(title='Time [s]'))

iplot(fig, filename='dist_only')

fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(12,6))

s = sns.boxplot(ax = ax1, x="Class", y="Amount", hue="Class",data=data_df,
palette="PRGn",showfliers=True)

s = sns.boxplot(ax = ax2, x="Class", y="Amount", hue="Class",data=data_df,
palette="PRGn",showfliers=False)

plt.show();

tmp = data_df[['Amount','Class']].copy()

class_0 = tmp.loc[tmp['Class'] == 0]['Amount']

class_1 = tmp.loc[tmp['Class'] == 1]['Amount']

class_0.describe()

class_1.describe()

fraud = data_df.loc[data_df['Class'] == 1]

trace = go.Scatter(
    x = fraud['Time'],y = fraud['Amount'],

```

```

name="Amount",
marker=dict(
    color='rgb(238,23,11)',
    line=dict(
        color='red',
        width=1),
    opacity=0.5,
),
text= fraud['Amount'],
mode = "markers"
)
data = [trace]
layout = dict(title = 'Amount of fraudulent transactions',
    xaxis = dict(title = 'Time [s]', showticklabels=True),
    yaxis = dict(title = 'Amount'),
    hovermode='closest'
)
fig = dict(data=data, layout=layout)
iplot(fig, filename='fraud-amount')
plt.figure(figsize = (14,14))
plt.title('Credit Card Transactions features correlation plot (Pearson)')
corr = data_df.corr()
sns.heatmap(corr,xticklabels=corr.columns,yticklabels=corr.columns,linewidths=.1,c
map="Reds")

```

```

plt.show()

s = sns.lmplot(x='V20', y='Amount', data=data_df, hue='Class',
               fit_reg=True, scatter_kws={'s': 2})

s = sns.lmplot(x='V7', y='Amount', data=data_df, hue='Class',
               fit_reg=True, scatter_kws={'s': 2})

plt.show()

s = sns.lmplot(x='V2', y='Amount', data=data_df, hue='Class',
               fit_reg=True, scatter_kws={'s': 2})

s = sns.lmplot(x='V5', y='Amount', data=data_df, hue='Class',
               fit_reg=True, scatter_kws={'s': 2})

plt.show()

var = data_df.columns.values

i = 0

t0 = data_df.loc[data_df['Class'] == 0]
t1 = data_df.loc[data_df['Class'] == 1]

sns.set_style('whitegrid')

plt.figure()

fig, ax = plt.subplots(8, 4, figsize=(16, 28))

for feature in var:

    i += 1

    plt.subplot(8, 4, i)

    sns.kdeplot(t0[feature], bw=0.5, label="Class = 0")

    sns.kdeplot(t1[feature], bw=0.5, label="Class = 1")

    plt.xlabel(feature, fontsize=12)

```

```

locs, labels = plt.xticks()

plt.tick_params(axis='both', which='major', labelsize=12)

plt.show();

target = 'Class'

predictors = ['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',\
              'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19',\
              'V20', 'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28',\
              'Amount']

train_df, test_df = train_test_split(data_df, test_size=TEST_SIZE,
                                     random_state=RANDOM_STATE, shuffle=True )

train_df, valid_df = train_test_split(train_df, test_size=VALID_SIZE,
                                     random_state=RANDOM_STATE, shuffle=True )

clf = RandomForestClassifier(n_jobs=NO_JOBS,
                             random_state=RANDOM_STATE,
                             criterion=RFC_METRIC,
                             n_estimators=NUM_ESTIMATORS,
                             verbose=False)

clf.fit(train_df[predictors], train_df[target].values)

preds = clf.predict(valid_df[predictors])

tmp = pd.DataFrame({'Feature': predictors, 'Feature importance':
                    clf.feature_importances_})

tmp = tmp.sort_values(by='Feature importance',ascending=False)

plt.figure(figsize = (7,4))

plt.title('Features importance',fontsize=14)

```

```

s = sns.barplot(x='Feature',y='Feature importance',data=tmp)

s.set_xticklabels(s.get_xticklabels(),rotation=90)

plt.show()

roc_auc_score(valid_df[target].values, preds)

clf = AdaBoostClassifier(random_state=RANDOM_STATE,

                        algorithm='SAMME.R',

                        learning_rate=0.8,

                        n_estimators=NUM_ESTIMATORS)

clf.fit(train_df[predictors], train_df[target].values)

preds = clf.predict(valid_df[predictors])

tmp = pd.DataFrame({'Feature': predictors, 'Feature importance':
clf.feature_importances_})

tmp = tmp.sort_values(by='Feature importance',ascending=False)

plt.figure(figsize = (7,4))

plt.title('Features importance',fontsize=14)

s = sns.barplot(x='Feature',y='Feature importance',data=tmp)

s.set_xticklabels(s.get_xticklabels(),rotation=90)

plt.show()

cm = pd.crosstab(valid_df[target].values, preds, rownames=['Actual'],
colnames=['Predicted'])

fig, (ax1) = plt.subplots(ncols=1, figsize=(5,5))

sns.heatmap(cm,

            xticklabels=['Not Fraud', 'Fraud'],

            yticklabels=['Not Fraud', 'Fraud'],

```

```

        annot=True,ax=ax1,

        linewidths=.2,linecolor="Darkblue", cmap="Blues")

plt.title('Confusion Matrix', fontsize=14)

plt.show()

roc_auc_score(valid_df[target].values, preds)

clf = CatBoostClassifier(iterations=500,

                        learning_rate=0.02,

                        depth=12,

                        eval_metric='AUC',

                        random_seed = RANDOM_STATE,

                        bagging_temperature = 0.2,

                        od_type='lter',

                        metric_period = VERBOSE_EVAL,

                        od_wait=100)

clf.fit(train_df[predictors], train_df[target].values,verbose=True)

preds = clf.predict(valid_df[predictors])

tmp = pd.DataFrame({'Feature': predictors, 'Feature importance':
clf.feature_importances_})

tmp = tmp.sort_values(by='Feature importance',ascending=False)

plt.figure(figsize = (7,4))

plt.title('Features importance',fontsize=14)

s = sns.barplot(x='Feature',y='Feature importance',data=tmp)

s.set_xticklabels(s.get_xticklabels(),rotation=90)

plt.show()

```

```

cm = pd.crosstab(valid_df[target].values, preds, rownames=['Actual'],
colnames=['Predicted'])

fig, (ax1) = plt.subplots(ncols=1, figsize=(5,5))

sns.heatmap(cm,

            xticklabels=['Not Fraud', 'Fraud'],

            yticklabels=['Not Fraud', 'Fraud'],

            annot=True,ax=ax1,

            linewidths=.2,linicolor="Darkblue", cmap="Blues")

plt.title('Confusion Matrix', fontsize=14)

plt.show()

roc_auc_score(valid_df[target].values, preds)

# Prepare the train and valid datasets

dtrain = xgb.DMatrix(train_df[predictors], train_df[target].values)
dvalid = xgb.DMatrix(valid_df[predictors], valid_df[target].values)
dtest = xgb.DMatrix(test_df[predictors], test_df[target].values)

#What to monitor (in this case, **train** and **valid**)

watchlist = [(dtrain, 'train'), (dvalid, 'valid')]

# Set xgboost parameters

params = {}

params['objective'] = 'binary:logistic'

params['eta'] = 0.039

params['silent'] = True

params['max_depth'] = 2

params['subsample'] = 0.8

```

```

params['colsample_bytree'] = 0.9

params['eval_metric'] = 'auc'

params['random_state'] = RANDOM_STATE

model = xgb.train(params,
                  dtrain,
                  MAX_ROUNDS,
                  watchlist,
                  early_stopping_rounds=EARLY_STOP,
                  maximize=True,
                  verbose_eval=VERBOSE_EVAL)

fig, (ax) = plt.subplots(ncols=1, figsize=(8,5))

xgb.plot_importance(model, height=0.8, title="Features importance (XGBoost)",
ax=ax, color="green")

plt.show()

preds = model.predict(dtest)

roc_auc_score(test_df[target].values, preds)

params = {
    'boosting_type': 'gbdt',
    'objective': 'binary',
    'metric': 'auc',
    'learning_rate': 0.05,
    'num_leaves': 7, # we should let it be smaller than 2^(max_depth)
    'max_depth': 4, # -1 means no limit

```

'min_child_samples': 100, # Minimum number of data need in a child(min_data_in_leaf)

'max_bin': 100, # Number of bucketed bin for feature values

'subsample': 0.9, # Subsample ratio of the training instance.

'subsample_freq': 1, # frequency of subsample, <=0 means no enable

'colsample_bytree': 0.7, # Subsample ratio of columns when constructing each tree.

'min_child_weight': 0, # Minimum sum of instance weight(hessian) needed in a child(leaf)

'min_split_gain': 0, # lambda_l1, lambda_l2 and min_gain_to_split to regularization

'nthread': 8,

'verbose': 0,

'scale_pos_weight':150, # because training data is extremely unbalanced

}

dtrain = lgb.Dataset(train_df[predictors].values,

label=train_df[target].values,

feature_name=predictors)

dvalid = lgb.Dataset(valid_df[predictors].values,

label=valid_df[target].values,

feature_name=predictors)

evals_results = {}

model = lgb.train(params,

dtrain,

```

valid_sets=[dtrain, dvalid],

valid_names=['train','valid'],

evals_result=evals_results,

num_boost_round=MAX_ROUNDS,

early_stopping_rounds=2*EARLY_STOP,

verbose_eval=VERBOSE_EVAL,

feval=None)

fig, (ax) = plt.subplots(ncols=1, figsize=(8,5))

lgb.plot_importance(model, height=0.8, title="Features importance (LightGBM)",
ax=ax,color="red")

plt.show()

preds = model.predict(test_df[predictors])

roc_auc_score(test_df[target].values, preds)

```