***TEST SPECIFICATION***

**Team Members:  
1. Suraj Raj - 2836416**

**2. Pradeep Nandagiri - 2835198**

**3. John Wesley - 2830415**

**4. Pavan Sai Pamujula - 2836714**

**1.0 Introduction**

Now a day the usage of credit cards has dramatically increased. As credit card becomes the most popular mode of payment for both online as well as regular purchase, cases of fraud associated with it are also rising. In this paper, we model the sequence of operations in credit card transaction processing using a Decision tree and Deep Neural Network show how it can be used for the detection of frauds. An both algorithms is initially trained with the normal behaviour of a cardholder. If an incoming credit card transaction is not accepted by the trained with sufficiently high probability, it is considered to be fraudulent. At the same time, we try to ensure that genuine transactions. We present detailed experimental results to show the effectiveness of our approach and compare it with other techniques available in the literature.

Although machine learning systems are not traditional software systems, not testing them properly for their intended purposes can lead to a huge impact in the real world. This is because machine learning systems reflect the biases of the real world. Not accounting or testing for them will inevitably have lasting and sometimes irreversible impacts.

**1.1 Goals and objectives**

The objectives of credit card fraud detection are to reduce losses due to payment fraud for both merchants and issuing banks and increase revenue opportunities for merchants.

**1.2 Statement of scope**

‘Fraud’ in credit card transactions is unauthorized and unwanted usage of an account by someone other than the owner of that account. Necessary prevention measures can be taken to stop this abuse and the behaviour of such fraudulent practices can be studied to minimize it and protect against similar occurrences in the future.In other words, Credit Card Fraud can be defined as a case where a person uses someone else credit card for personal reasons while the owner and the card issuing authorities are unaware of the fact that the card is being used. Fraud detection involves monitoring the activities of populations of users in order to estimate, perceive or avoid objectionable behaviour, which consist of fraud, intrusion, and defaulting.Machine learning algorithms are employed to analyse all the authorized transactions and report the suspicious ones. These reports are investigated by professionals who contact the cardholders to confirm if the transaction was genuine or fraudulent.

Functional vs Non Functional Requirements

Requirements analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and Non-functional requirements.

Functional Requirements: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

**Example:**

1) Authentication of user whenever he/she logs into the system.

2) System shutdown in case of a cyber-attack.

3) A Verification email is sent to user whenever he/she registers for the first time on some software system.

Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

1. Portability

2. Security

3. Maintainability

4. Reliability

5. Scalability

6. Performance

7. Reusability

8. Flexibility

**Example:**

1) Emails should be sent with a latency of no greater than 12 hours from such an activity.

2) The processing of each request should be done within 10 seconds

3) The site should load in 3 seconds when the number of simultaneous users are > 10000

**2.0 Test Plan**

This section describes the overall testing strategy, and the project management issues that are required to properly execute effective tests.

**2.1 Software to be tested**

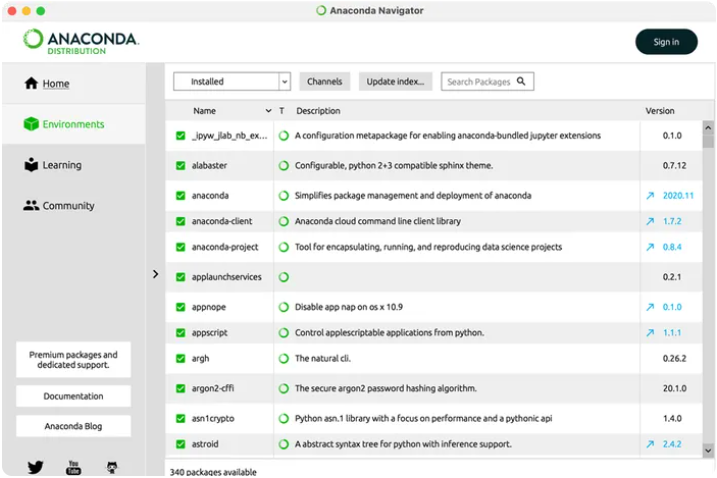
The software to be tested is identified by name. Exclusions are noted explicitly.

**Anaconda Navigator** is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands.

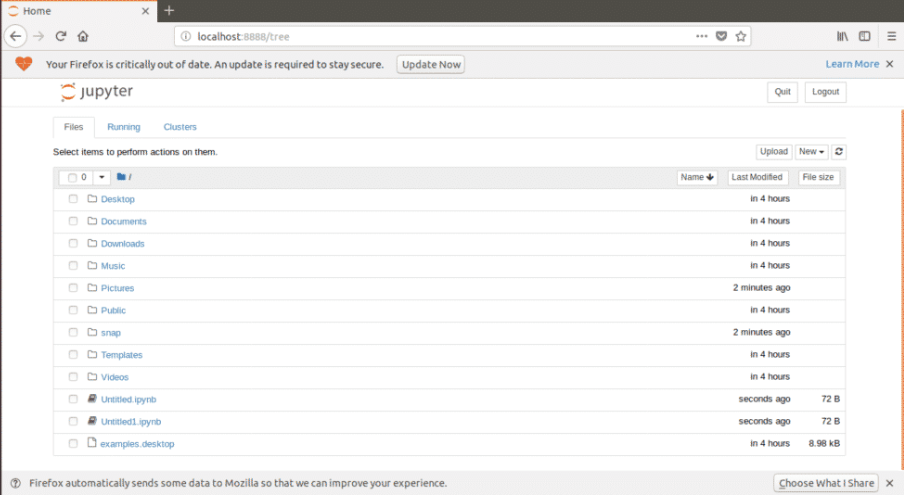
**2.3 Testing tools and environment**

A description of the test environment, test files, and other resources is presented here.

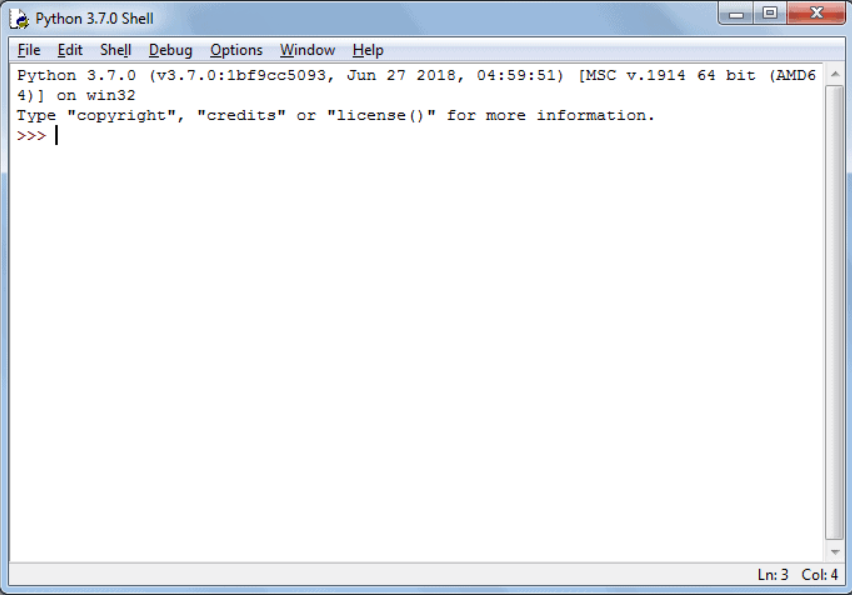
* Anaconda Navigator



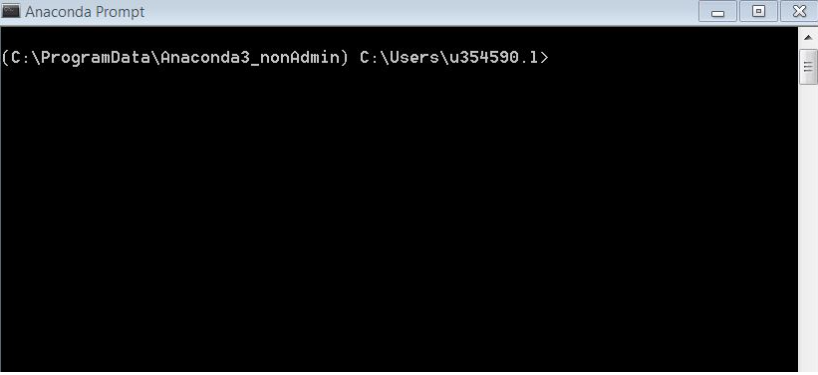
* Jupyter Notebook



* Python IDLE



* Anaconda prompt



**2.4 Test schedule**

**TESTING METHODS**

* **Functional Testing**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

* Functions: Identified functions must be exercised.
* Output: Identified classes of software outputs must be exercised.
* Systems/Procedures: system should work properly
* **Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

Test Case for Excel Sheet Verification:

Here in machine learning we are dealing with dataset which is in excel sheet format so if any test case we need means we need to check excel file. Later on, classification will work on the respective columns of dataset.

**3.0 MODULES**

1. **DATA COLLECTION**
2. **DATA PRE-PROCESSING**
3. **FEATURE EXTRATION**
4. **EVALUATION MODEL**
5. **FLASK**

**DATA COLLECTION**

Data used in this paper is a set of product reviews collected from credit card transactions records. This step is concerned with selecting the subset of all available data that you will be working with. ML problems start with data preferably, lots of data (examples or observations) for which you already know the target answer. Data for which you already know the target answer is called *labelled data*.

**DATA PRE-PROCESSING**

Organize your selected data by formatting, cleaning and sampling from it.

Three common data pre-processing steps are:

* **Formatting:** The data you have selected may not be in a format that is suitable for you to work with. The data may be in a relational database and you would like it in a flat file, or the data may be in a proprietary file format and you would like it in a relational database or a text file.
* **Cleaning:** Cleaning data is the removal or fixing of missing data. There may be data instances that are incomplete and do not carry the data you believe you need to address the problem. These instances may need to be removed. Additionally, there may be sensitive information in some of the attributes and these attributes may need to be anonymized or removed from the data entirely.
* **Sampling:** There may be far more selected data available than you need to work with. More data can result in much longer running times for algorithms and larger computational and memory requirements. You can take a smaller representative sample of the selected data that may be much faster for exploring and prototyping solutions before considering the whole dataset.

**FEATURE EXTRATION**

Next thing is to do Feature extraction is an attribute reduction process. Unlike feature selection, which ranks the existing attributes according to their predictive significance, feature extraction actually transforms the attributes. The transformed attributes, or features, are linear combinations of the original attributes. Finally, our models are trained using Classifier algorithm. We use classify module on Natural Language Toolkit library on Python. We use the labelled dataset gathered. The rest of our labelled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre-processed data. The chosen classifiers were Random Forest. These algorithms are very popular in text classification tasks.

**EVALUATION MODEL**

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. Evaluating model performance with the data used for training is not acceptable in data science because it can easily generate overoptimistic and over fitted models. There are two methods of evaluating models in data science, Hold-Out and Cross-Validation. To avoid over fitting, both methods use a test set (not seen by the model) to evaluate model performance. Performance of each classification model is estimated based on its averaged. The result will be in the visualized form. Representation of classified data in the form of graphs. **Accuracy** is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions.

**FLASK**

**Flask** is a micro [web framework](https://en.wikipedia.org/wiki/Web_framework) written in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)). It is classified as a [microframework](https://en.wikipedia.org/wiki/Microframework) because it does not require particular tools or libraries. It has no [database](https://en.wikipedia.org/wiki/Database) abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

**Features:**

Development server and [debugger](https://en.wikipedia.org/wiki/Debugger)

Integrated support for [unit testing](https://en.wikipedia.org/wiki/Unit_testing)

[RESTful](https://en.wikipedia.org/wiki/Representational_state_transfer) request dispatching

Uses Jinja templating

Support for secure cookies (client-side sessions)

100% [WSGI](https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface) 1.0 compliant

[Unicode](https://en.wikipedia.org/wiki/Unicode)-based

Complete documentation

[Google App Engine](https://en.wikipedia.org/wiki/Google_App_Engine) compatibility

Extensions available to extend functionality

**4.0 Test Cases**

This section enumerates a complete list of test cases for the software. A template for test cases is as follows.

