# METHODOLOGY

Anomaly detection is a decisive procedure used to spot lop-sided patterns that are deviating to the “ideal” pattern (Schneider & Xhafa, 2022). In this thesis, the methodology to be used will be an **unsupervised outlier detection methodology** that will follow the following steps:

1. Algorithm/Model Selection
2. Data identification and exploration
3. Data visualization and presentation
4. Dataset splitting and training
5. Dataset pre-processing
6. Resampling of data in the dataset
7. Dataset outlier detection using various algorithms (IForest, LOF, COPOD, and DAN)
8. Visualization of the outliers and inliers (concentrating more on the outliers)
9. Evaluation and metrics
10. Predicting fraudulent transactions with ‘unseen data’

In this thesis, the dataset to be used would be downloaded from Kaggle at <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud?resource=download>. The dataset contains 284, 807 credit card transactions from European credit card users and this dataset will be explored to gain better understanding and how to exploit it towards the success of this work. It comprises the following attributes: ***time***, ***V1-V28***, ***Amount***, and ***Class;*** however, only “Time”, “Amount”, and “Class” were not transformed using the “Principal Component Algorithm – PCA” to transform them into principal components. “Time” keeps entries about the time passed between each transaction and the initial transactions and it is an Integer variable, “V1-V2” are ‘double’ variables that represent the principal components 1-28, and “Amount” is also a ‘double’ that holds the amount transacted, while “Class” captures information about the category the transaction belongs to (either legitimate or fraudulent).

The data will be described, pre-processed, and explored. Data pre-processing will include: treatment of missing values, standardization of non-standard values, data normalization, formatting, data binning, and dropping of duplicate entries. A correlation matrix will be produced for “Time”, “Class”, “Amount”, and all the principal components (V1-V28), to find out if they are correlated to one another or not. Additionally, the ***StandardScalar*** library in ***scikit-learn*** package will be used to standardized any unstandardized variable identified during the dataset exploration.

The dataset is then split into train and test sets (a 70-30% or 65-35%) to be used for training and testing the models to be used in this work. The splitting will be done with a random seed of 42, after which the training set will be checked data consistency using the **great expectations library** (found at greatexpectations.io). All data in the dataset with high consistency are tagged as “Negatives” and those with low consistency are tagged “Positives”. Resampling of the imbalanced dataset will be achieved by under sampling, oversampling, or a mix of both with the goal of creating a balance between the “positives” and “negatives” – gotten from the consistency check and ensuring the classification algorithms are not loaded positively or negatively skewed data, which might affect the results. **SMOTE, Tomek-Link, and ADASYN** would be used for resampling as proposed by Lin and Jiang (2021) since the dataset is unbalanced. Model training is also done with the train dataset while the test dataset will then be used to check the performance of the models while predicting by means of classification, the legitimate and fraudulent transactions.

\*\*\*Recall that resampling is implemented only on the training dataset to ensure it is balanced before been used to train the classifier models selected for this research.

The **python** **pandas’ library** will be used for all pre-processing tasks. At the conclusion of these steps, data visualization and presentation will be done using the **python Matplotlib, Seaborn, and Plotly** because of their ease of use and ability to interact with each other freely without collision. Seaborn for instance is an upgraded version of Matplotlib, while Plotly makes possible interactive graphing of most Matplotlib and Seaborn’s outputs.

Additionally, classification will be achieved using the following classifiers (Local Outlier Factor Algorithm (LOF), Isolation Forest Algorithm (IForest), the Copula-Based Outlier Detection (COPOD), and the Deep Autoencoder Network (DAN)) found in the Python Outlier Detection (PyOD) library. The **PyOD** is the most scalable and all-inclusive Python-based toolkit for carrying out outlier detection in multivariate datasets and is comprised of over 40 outlier detection algorithms. Additionally, to the already mentioned algorithms, the PCA (found at pyod.models.pca.PCA), was be used to maintain the privacy in the credit card transaction dataset. The LOF is an unsupervised outlier detection that will be implemented in the “scikit-learn” LOF class. Outliers are identified by comparing the “Local Outlier Factor” of a particular data with that of other data in the dataset.

The IForest works through isolation using special random features and split values between specified maximum and minimum values. The AutoEncoder (AE) is similar to the PCA and detects outliers by computing the error from reconstruction of the dataset.

The PyOD library was also considered because of its flexibility and inclusion of proximity-based, linear, probabilistic, outlier ensembles, and neural networks. The classification algorithms selected for this work fall under the linear (PCA), outlier ensembles (IForest), proximity-based (LOF), Neural Networks (AutoEncoder), and probabilistic (MVOD) – cutting across the PyOD library. The PCA and AE complement each other and allow for EDA execution to better understand the interior structure of the dataset, IForest does not only function effectively in anomaly detection, but also less memory space since it is built with decision trees as its bedrock (datapoint splitting is first done into individual entities before classification). MVOD was selected since the dataset contains multiple variables and the LOF helps to detect outliers across data in the dataset from a side-by-side perspective. A very common seen across these algorithms is their ability to help in gaining deeper understand of the internal structure of the dataset while performing precise classification of the data points.

1. **IForest:** this algorithm is existent in the PyOD library and will be executed in two stages – the training and testing stages respectively. In the training stage, IForest model is built tailored to the dataset for a deeper understanding; after which sub-samples of the training dataset will be used to generate isolation trees that would be used for the testing stage. The testing stage uses the isolation trees created using the training dataset to assign anomaly score to the data points in the test dataset. This algorithm will be implemented using the ***numpy***, ***pandas***, ***seasborn***, and ***IsolationForest*** Python packages. The model will be defined and fitted for training, after which anomaly scores will be obtained then used for prediction. At the end, the model will be evaluated and the result stored for the comparative analysis.
2. **LOF:** this algorithm will be used to compute the local density deviation of the data points in the test, with respect to their respective neighbors. For this algorithm, the model will be needing the following Python packages: ***numpy***, ***matplotlib***, ***sklearn.neighbors*** from the **LocalOutlierFactor** class. A random seed of 42 will be selected prior to training of the model with the train dataset after which the model will be fit then the test dataset will be inserted into the model that was trained with the train dataset.
3. **Deep Autoencoder Network (DAN):** in this algorithm, the training dataset will be fed into the encoder for automatic encoding while doing this, the model generates the compressed feature vector. The decoder decodes this vector to create the output data. A good thing about the resultant model is its ability to “almost” accurately work on similar dataset that was used to train it. In our case, as credit card dataset will be used to train the model, it will always perform well with credit card data and poorly with a dataset that has nothing to do with credit card transactions.

The following Python packages will be used for this model: ***pandas***, ***numpy***, ***matplotlib***, ***tensorflow***, ***sklearn***, and ***mpl.*** The ***tensorflow*** and ***mpl*** will be used for model building while others will be for dataset processing and visualization. In this model, after splitting into test and train dataset, the train dataset will be split further into normal and abnormal entries – then the model will be trained with only the normal data points from the train dataset. Modeling will be done using the “sequential modelling” through the Keras API (executed using Tensorflow as pointed out earlier). “Early Stopping Phenomenon” will be used for terminating model training, “Adam Optimizer” will be used for compiling the model and “Mean Absolute Error – MAE” as a loss function.

1. **COPOD:** this is probabilistic unsupervised anomaly detection algorithm known as “Copula-Based Outlier Detection” that will be used in this master’s thesis. This algorithm is inspired by statistics and multivariate data distribution; low computational complexity, and high interpretability. This algorithm will be modelled using the following Python packages: ***numpy***, ***numba***, ***scipy***, ***scikit\_learn.statsmodel***, and ***COPOD.***

A comprehensive comparative analysis of the classification algorithms will be done by measuring their performance and comparing them using different graphs/plots and dashboarding. The performance metrics to be used are the F1-score, Precision, Recall, Support, the Accuracy measure, the ROC curve, and the AUC-PR curve.

The following resources will be used for experimenting this methodology:

1. Jupyter Lab – to be used as the development environment because of its flexibility a lightweight nature.
2. Pandas’ python libraries – for all data pre-processing and exploration task. Then the Matplotlib | Seaborn | Plotly python libraries – for all data visualization and presentation task. Other python libraries to be used include PyOD, Scit-learn, Streamlit, and Great [Expectation](https://greatexpectations.io/)
3. GitHub and Datahub.io repository for data science
4. Kaggle dataset for credit card transactional data
5. Microsoft word for word processing and Microsoft Excel for Gantt charting
6. HP Laptop (a 1TB HDD; 8GB Ram; running Windows 10Pro build 21H2; and 2.30Ghz processor speed)