An Experimental Study using Unsupervised Machine Learning Techniques for Credit Card Fraud Detection

Abhishek Joshi[1], Shreedhar Soni[2], Prof. Vaibhav Jain[3]

Department of Information Technology[1], Department of Information Technology[2], Department of Computer Engineering[3], Institute of Engineering and Technology, DAVV, Indore, India

Abstract: *The unauthorised use of a credit card, or a card number, to acquire money or property is known as credit card fraud. Along with financial losses, it also makes people apprehensive and reluctant in using a card. Because of the large number of credit card users and the widespread use of digital payment technologies, a functional and reliable credit card fraud detection system is the need of the hour. In this paper we evaluate the performance of three unsupervised machine learning algorithms namely Local Outlier Factor, Isolation Forest Algorithm and K-means clustering on imbalanced credit card fraud data. The Hybrid sampling technique of oversampling, undersampling and k-fold cross validation are carried out on the skewed dataset. The impact of hyperparameter tuning or hyperparameter optimization on algorithm efficiency was also investigated. We have evaluated the algorithms with evaluation metrics like precision, sensitivity, specificity, PR-AUC, Matthews correlation coefficient, and balanced classification rate. Our results show that among these algorithms, Isolation Forest Algorithm outperforms Local Outlier Factor and K-means clustering algorithm.*

Keywords: *Credit Card Fraud, Local Outlier Factor, Isolation Forest, K-means Clustering, Hyperparameter Optimization, Oversampling, Undersampling, K-fold cross validation, Unsupervised machine learning algorithms.*

1. Introduction

A cashless society culture is gaining traction at a breakneck pace. People who pay their bills and handle their finances using their computers or smartphones tend to use bank cards (credit or debit cards). As per Research and Markets, due to the rising popularity of credit cards and the growing trend of buying goods first and paying later, the Indian credit card industry is expected to expand at a Compound Annual Growth Rate (CAGR) of more than 25% between 2020 and 2025 as reported in [1], so is the number of cases of fraud connected with it is also on the rise as mentioned in [2].

Fig 1: Trend of Global financial loss due to Credit Card Fraud[3].

As shown in Fig 1, the monetary losses accounting to credit card frauds has seen a rise through the last decade by a significant amount as well as the cents per 100 dollars lost to frauds has been on the increasing side.

Using an effective algorithm to determine whether or not a credit card transaction is fraudulent is a huge data science challenge. The challenges associated with Credit Card Fraud Detection are:

1. Manual detection of credit card fraud is impossible due to the large volume of transactions making it time consuming and inaccurate.
2. The characteristics of normal and fraudulent behaviour are continually shifting.
3. Credit card fraud data sets are highly skewed and rarely available due to confidentiality.
4. The sampling method on the dataset, the collection of variables, and the detection technique(s) used all have a significant impact on the efficiency of fraud detection in credit card transactions.

Generally speaking, fraud detection is a prediction problem, with the goal of maximising correct predictions while keeping incorrect predictions at a reasonable cost. Data mining is a method that collects valuable information with high quality and allows for efficient sharing. In the case of data mining and text analytics, predictive intelligence has been discovered by uncovering trends and associations present within structured and unstructured data. To support such analysis and classification problems, machine learning techniques such as Neural Networks have also been used. The premise behind supervised learning techniques is that fraudulent patterns can be learned from an examination of previous transactions. However, the challenge becomes more difficult as it must account for changes in consumer behaviour as well as fraudsters' ability to devise new fraud trends. Unsupervised learning techniques can aid fraud detection systems in detecting anomalies in this context. The aim of unsupervised outlier detection techniques is to characterise the data distribution of transactions without requiring knowledge of the transaction label. They are based on the premise that outliers in the transaction distribution are frauds;

as a result, they can be used to identify previously unseen forms of frauds because they do not depend on transactions that have previously been labelled as fraudulent.

The following is a list of the relevant contributions in our work:

1. We experimentally evaluated and compared the performance of three Unsupervised Machine learning algorithms: Local Outlier Factor, Isolation Forest Algorithm and K- means clustering.
2. We used performance metrics namely Accuracy, AUPRC (Area under Precision Recall Curve), Sensitivity, Specificity, Confusion Matrix, MCC (Matthew Correlation Coefficient) and BCR (Balanced Classification Rate) on imbalanced credit card fraud dataset.
3. We investigated hybrid technique of under-sampling and oversampling on skewed dataset and evaluated the performance of algorithms.
4. We applied k-Fold Cross-Validation to evaluate performance of machine learning models used in this experiment.
5. We also looked at the effect of hyperparameter tuning or hyperparameter optimization on the performance of algorithms.

The rest of this paper is organised as follows: In Section 2, we reviewed literature related to credit card fraud detection. Section 3 describes the classification algorithms and evaluation metrics used in experiment. In Section 4 we provided a detailed description of our experimental study along with system specifications. In Section 5 we report the overall results of our experiment along with discussion about comparative analysis. Section 6 concludes the comparative study and suggests future areas of research.

2. Literature Review

Credit card fraud detection is a data mining classification issue that aims to correctly classify credit card purchases as legitimate or fraudulent. We have compiled a list of publications that deals with detecting credit card fraud in this section.

Sara Makki et al. [4] in their study classified fraud analysis into two main categories namely Statistical Modeling and Machine Learning Techniques. Statistical Modeling is a branch of mathematics concerned with gathering and analyzing data using distributions that adhere to certain assumptions whereas Machine Learning is a form of data analysis that automates the development of analytical models. Machine Learning Fraud Detection technique consists of Supervised and Unsupervised Learning. Supervised Learning (Classification and Regression) uses labeled past transactions to learn a fraud prediction model that returns the likelihood of a new transaction being a fraud. Various Supervised Learning techniques include K-Nearest Neighbor, Naïve Bayes, Bayesian Belief Network, Logistic Regression, Decision Tree, Artificial Neural Networks, Support Vector Machines, Artificial Immune Systems and Hidden Markov Model. Unsupervised Learning (Clustering and Outlier Detection) is used to analyze and cluster unlabeled datasets. Its techniques include K-Means Clustering, Local Outlier Factor, Isolation Forest Algorithm, Self-Organizing Maps, Break point Analysis and Peer Group Analysis.

Fabrizio Carcillo et al. [5] to increase fraud detection accuracy,presented a hybrid technique that combines supervised and unsupervised techniques. On an actual, annotated credit card fraud detection

dataset, unsupervised outlier scores computed at various levels of granularity are compared and tested. The results of the experiments show that the combination is successful and improves detection accuracy. But because of the inadequate level of granularity required to take advantage of unsupervised knowledge, the findings are not persuasive in terms of global and local approaches.

Zhangyu Cheng et al. [6] proposed a two – layer progressive ensemble approach for outlier detection comprising of Isolation Forest Method and Local Outlier Factor as first and second method respectively which improve the outlier detection rate and greatly reduce the time complexity.

Utkarsh Porwal et al. [7] suggested a novel method for distinguishing outliers and pure inliers by assigning a consistency score to each and every data point. They demonstrated that commonly used area under the Receiver Operating Characteristic curve is not the right metric because inliers (true negatives) are significantly higher than outliers (true positives). Precision Recall curves are better suited because precision compares false positives to true positives (outliers) rather than true negativities (inliers) and therefore is not affected by the problem of class imbalance as compared to Area under ROC(Receiver Operating Characteristic) in imbalanced datasets.

We also looked into the literature on comparative studies of different algorithms. Aihua Shen et al. [8] tested decision tree, neural networks and logistic regression are tested for fraud detection. The proposed neural network classifier and logistic regression methods outperform decision trees in solving the problem under investigation.

Y. Sahin et al. [9] evaluated the performance of SVM and Decision Tree on Credit Card Fraud dataset where result showed that Decision Tree outperforms SVM approach on accuracy. Jha et al. [10] in a comparison of logistic regression, support vector machine, and random forest found that random forest outperformed the two other approaches.

John O. Awoyemi et al. [11] carried out hybrid technique of oversampling and undersampling on imbalanced credit card fraud dataset and applied three supervised learning algorithms namely K- nearest neighbor, Logistic Regression and Naïve Bayes. The results show that K-nearest neighbour outperforms Naive Bayes and Logistic Regression techniques.

Rajeshwari U et al. [12] proposed a method for detecting fraudulent transactions on realtime. Later on, the transaction will be cancelled, and the card owner will receive warnings about the fraudulent transactions.The value provided for the Hidden Markov Model outcome is used to determine if the transaction is fraudulent or not. This system reduces the number of false alarms generated by comparisons between actual transactions that have been flagged as fraudulent. Using streaming analytics, fraud is avoided and false alarm rates are reduced.

3. Algorithms and Evaluation Metrics

In this section, we present unsupervised machine learning algorithms that we used and discuss about the various performance evaluation metrics to compare these algorithms.

3.1 Unsupervised Machine Learning Algorithms

The brief description of the Algorithms we used in our paper are as follows:

1. **Local Outlier Factor**: The Local Outlier Factor (LOF) algorithm is an unsupervised anomaly detection method that calculates a data point's local density deviation from its neighbors. It finds samples with a significantly lower density than their neighbors to be outliers (fraudulent transaction).Local Outlier Factor's Algorithm is described below [6].

Algorithm: LOF (*n_neighbour*, *n_outliers*,*data*)

Input: *n_neighbour* - number of near neighbors,
n_outliers - number of outliers,
data - outlier candidate dataset.

Output: *n_outliers*in the candidate dataset.

1: **for** iteration = 1 to len(data) **do**
2: compute n_neighbor – $dist_{p}$
3: compute $N_{n\_neighbourP}$
4: **end for**
5: calculate ($reach$ - $dist_{n\_neighbourP,\ r)}$ and $lrdp$
6: calculate lof_{p}
7: sort the lof values of all points in descending order
8: **return** the n_outliersdata objects with the large lof values, which are the outliers.

Where,
$dist_{p}$: distances from point *p* to other data points
$N_{n\_neighbourP}$: Data point set to point *p* distance less than n_neighbour - $dist_{p}$, recorded as $N_{n\_neighbourP}$
reach – $dist_{n\_neighbourP,\ r}$: Reachability distance calculated as max{n_neighbour - $dist_{r}$, $d_{p,\ r}$}
where $d_{p,\ r}$ represents the distance from point *p* to point *r*
lrdp: local reachability density calculated as mean of the reachable distance of the data point *p* and its *k* nearest neighbours
lof_{p}: The average of the ratio of the local reachable density of the point p neighbourhood point to the local reachable density point p is the local outlier factor.

2. **Isolation Forest Algorithm**: The Isolation Forest isolates observations by selecting a feature at random and then selecting a split value between the feature's maximum and minimum values at random.
The number of splitting needed to isolate a sample is equal to the path length from the root node to the terminating node, since recursive partitioning can be expressed by a tree structure. This path length is a measure of normality and our decision function, averaged over a forest of such random trees.Anomalies' paths are slightly shorter when random partitioning is used. As a result, when a forest of random trees produces shorter path lengths for specific samples, they are almost certainly anomalies. This classifier's job is to identify fraudulent transactions by isolating samples by selecting a function at random, and then generating a split value between the maximum and minimum values of the selected feature. As a result, the data is classified as fraudulent depending on how isolated it is[6].
Its Algorithm is described below[15].

The algorithms comprises of two phases first is the training phase in which isolation trees are constructed and an isolation forest is made by recursively splitting the given training dataset until instances are isolated.

Algorithm 1: isolationForest (*data*, *num_trees*, *sb_size*)

Input: *data* - input dataset,
num_trees - number of trees,
sb_size - subsamplingsize.

Output: a set of *num_trees*isolationTrees

1: Initialize Forest
2: set height limit *l*im = $ceil(\log_2(sb\_size))$
3: **for** iteration = 1 to *num_trees* **do**
4: data ′←sam(data, sb_size)
5: $Forest \leftarrow Forest \cup i$solation$T$(data′, 0, *l*im)
6: **end for**
7: r**eturn** $Forest$

Where,
Forest: Consists of a group of binary trees constructed from the random property of the dataset
data ′: Consists of sampled value of data sets into subsampling size
isolationTree(data ′, 0, lim): The function returns an isolation tree by randomly selecting an attribute in the dataset and randomly selecting a split value between the maximum and minimum values of the selected attribute.

After the execution of isolationForest function we have a collection of trees returned as the output which is ready for the evaluation stage. At the evaluation phase, an anomaly score is derived from the expected path length for each test instance derived by passing instances through each isolation tree in an isolation forest. Using *pathlength* function a single path length is derived by counting the no. of edges e from root to a terminating node as instance traverses in an isolation tree.

Algorithm 2: pathLength (*ins*, *Tree*, *cur*)

Input: *ins* – an instance,
Tree – an isolation tree,
cur – current path length; to be initialized to zero when first called

Output: path length of instance *ins*

1: **if** Tree is an external node, **then**
2: return cur + c(Tree.size)
3: **end if**
4: a ← Tree.splitAttribute
5: **if** ins_a < Tree.splitVal **then**
6: return pathLength(ins, Tree.left, cur + 1)
7: **else**
8: return pathLength(ins, Tree.right, cur + 1)

3. K means Clustering: It's an unsupervised learning algorithm that places k centroids in the data and assigns each data point to the cluster closest to it. It tries to keep the centroids as small as possible when doing so, where ‘means' in K means data averaging, i.e. finding the centroid. Its Algorithm is described below[16].

Algorithm: K-Means Clustering

1: **while** True **do**
2: **for** j = 1 to m **do**
3: $c^{(j)}$:= index (1 to k) of cluster centroid closest to $X^{(j)}$
4: **end for**
5: **for** *k* = 1 to *K* **do**
6: $\mu_{(k)}$:= average or mean of points assigned to cluster k
7: **end for**
8: **end while**

Where,
K: Total number of centroids
k: Current centroid under consideration
The first inner for loopassigns the most appropriate centroid to the instance under consideration which is $X^{(j)}$by giving it the index of the centroid from which the instance is at the smallest distance calculated as:
$c^{(i)} = \min||x^{(i)} - \mu_{(k)}||$
The 'k' for which we get the smallest distance will be the centroid to which this instance will be assigned.
In the second inner for loop, the new position of a centroid is calculated by taking average or mean of all the instances belonging to this centroid and assigned to $\mu_{(k)}$.
The above steps are repeated till no more position changes occur in centroids or we run out of maximum iteration limit. [16].

3.2 Metrics for evaluation of Performance

Below we present the list of performance metrics to evaluate the machine learning algorithms we used:

1.**Accuracy:** It is known as the ratio of the classifier's correct fraud prediction to the total number of input data samples in order to determine the algorithm's effectiveness in detecting fraud transactions where tp denotes the number of true positives (fraud samples), tn denotes the number of true negatives (non-fraud samples), fp denotes the number of false positives (non-fraud samples misrepresented as fraud samples), and fn denotes the number of false negatives (fraud samples misrepresented as non-fraud samples), and fn denotes the number of false negatives (fraud samples misrepresented as non-fraud).Value of accuracy ranges from 0% to 100% where 100% is the best score when you're working on a classification problem.

$$\text{Accuracy} = \frac{tp+tn}{\text{tp+tn+fp+fn}}$$

2. **Precision:** It's a score that measures the classifiers' ability to accurately label the legitimate datapoint as 0 and the fraud datapoint as 1. It is defined as the ratio of the total number of true positives (fraud samples) to the sum of true and false positives (fraud and valid samples), where tp represents the number of true positives (fraud samples) and fp represents the number of false positives (fraud and valid samples) (valid samples).Its value ranges from 0(no precision) to 1(perfect or full precision).

$$\text{Precision} = \frac{tp}{tp+fp}$$

3. **AUC-PRC**(Area under Precision-Recall Curve): With a single value, it summarizes the details in the precision-recall curve.. Across various judgement thresholds, a PR curve depicts the trade-off between accuracy and recall. In comparison to the AUROC (Area under Receiver Operating Characteristic) curve, the Precision-Recall (PR) curve is a better measure since it is more sensitive to the issue of class imbalance.In general, the higher the AUC-PR score, the better the classifier performs the given task.Its value ranges from 0 to 1(optimum).

4. **MCC**(Matthews Correlation Coefficient): The Matthews Correlation Coefficient (MCC) is a binary classification problem evaluation metric. Since its evaluation consists of TP, FP, TN, and FN, MCC is mostly used for unbalanced data sets. The MCC value is usually between -1 and +1; where +1 value indicates excellent classification, and -1 value indicates complete separation of classification and observation [11].

$$\text{MCC}= \frac{(tp \times tn) - (fp \times fn)}{\sqrt{(tp + fp)\,(tp + fn)\,(tn+fp)\,(tn+fn)}}$$

5. **Sensitivity**(Recall): The accuracy of positive (fraud) case classification is determined by sensitivity (recall). The ratio of the total number of true positives (fraud samples) to the amount of true positives (fraud) and false negatives (fraud samples marked as valid samples) is used to calculate it. The number of true positives (fraud samples) is tp, and the number of false negatives is fn (fraud sample misrepresented as valid samples)[11]. Value of sensitivity ranges from 1(optimum) to zero.

$$\text{Sensitivity} = \frac{\text{tp}}{tp + tn}$$

6. **Specificity:**The accuracy of negative (legitimate) case classification is determined by specificity. It's a metric that assesses a model's ability to predict true negatives in each group[10]. Its value ranges from 0 to 1(optimum).

$$\text{Specificity} = \frac{\text{tn}}{tn + fp}$$

7. **Balanced Classification Rate**(BCR): The average of sensitivity and specificity, which is the amount of negatives categorized as negatives, is called the Balanced Classification Rate[11]. Best value of BCR is 1 and its worst value is 0.

$$\text{BCR} = \frac{1}{2}\left(\frac{\text{tp}}{tp+fn} + \frac{\text{tn}}{tn+fp}\right)$$

4. Experimental Study

In this section, we present the experimental setup that we used to implement and evaluate unsupervised machine learning algorithms on credit card data.

4.1 Dataset Used

The data was gathered and analysed as part of a collaborative research project from Worldline and Machine Learning Group of ULB on Big Data Mining and fraud detection detailed in [14].

Credit card purchases made by European cardholders in September 2013 are included in the dataset. This dataset contains 284,807 transactions that occurred over the course of two days. The positive class(fraud cases) make up 0.172% of the transactions data.The data is extremely unbalanced and skewed in favour of the positive.It only has numerical (continuous) input variables, which are the result of a feature selection transformation using Principal Component Analysis (PCA), which yielded 28 principal components.Thus a total of 30 input features are made in use in this study. Owing to security concerns, the features' specifications and context information cannot be shared. The number of seconds between each transaction and the dataset's first transaction. in the dataset are stored in the time function. The transaction amount is represented by the 'amount' attribute. The binary classification's target class is 'class', which has a value of 1 for a positive case (fraud) and 0 for a negative case (non-fraud).

4.2 Workflow of Experiment

The experiments which we have performed on the dataset is divided in two phases or parts. The first phase comprises of splitting the dataset in three ratios respectively:

1. 60% training set, 40% testing set
2. 70% training set, 30% testing set
3. 80% training set, 20% testing set

The three algorithms are trained against the training datasets and then tested against the testing datasets and the performance metrics at each split are then comparedto evaluate the performance. The performance at each split is displayed using the performance evaluation metrics which are namely accuracy, precision, recall, confusion matrix, area under the precision recall curve, Matthew's correlation coefficient, sensitivity and specificity. The results are then illustrated graphically for each of the algorithms at each split.

In the second phase, we have experimented whether the imbalance dataset classification approaches which includes oversampling and undersampling improve the performance of algorithms or not. All three algorithms are then compared on the basis of different performance metrics to observe the impact of undersampling and oversampling the dataset. In this phase, we have also shed light on an important aspect which is hyperparameter tuning for an algorithm: meaning to set the hyperparameters in the algorithms in such a way to maximize their performance for datasets with class imbalance problem.We also performed K-fold cross validation to assess evaluation of our machine learning models .The results are then presented diagrammatically.

4.3 Phase1: Comparative analysis of performance at different dataset splits

We split the dataset using train-test split in 60% training, 40% testing set, 70% training, 30% testing set and 80% training, 20% testing set. Then we fit the dataset on all the three algorithms, evaluated performance based on different metrics and compared the performance metrics as shown below:

Comparison of Accuracy at different dataset splits

Algorithms Implemented	60% training set 40% testing set	70% training set 30% testing set	80% training set 20% testing set
Local Outlier Factor	99.6752	99.6804	99.6804
Isolation Forest	99.7787	99.7799	99.7928
K-Means Clustering	53.9978	53.8756	53.9043

Table 1: Comparison of Accuracy of various algorithms at different dataset splits

As shown in Table 1, in reference to the performed experiment, the accuracy of local outlier factor and isolation forest algorithm are comparable but isolation forest outperforms local outlier factor by very close margin while k-means clustering lacks in accuracy and has the minimum accuracy

Comparison of precision, recall, f1-score and area under PRC at different dataset splits

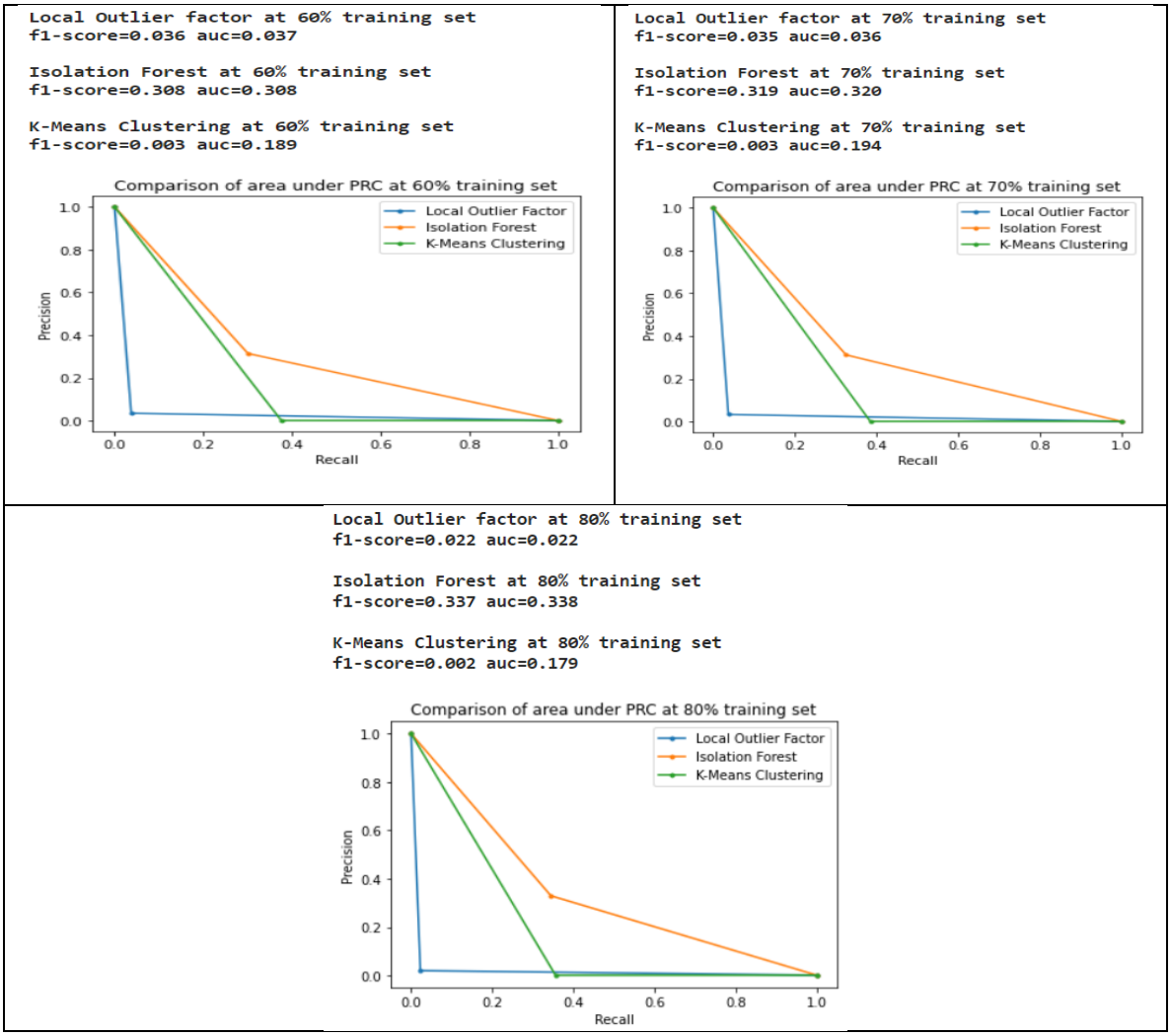

Fig 2: Comparison of precision, recall, f1-score and area under PRC at different dataset splits

As shown in Fig 2, is a diagrammatic view of the precision and recall value of the three algorithms is presented as well as the area under the precision recall curve is calculated which is one of the most important metrics to consider if data set suffers from class imbalance problem .The isolation forest algorithm outperforms both the other algorithms in this aspect as well while we notice that k-means clustering being so much behind in terms of accuracy outperforms local outlier factor in area under precision recall curve. Local outlier factor show results that vary much from both other algorithms and is the least performing algorithm in terms of precision, recall and area under precision recall curve.

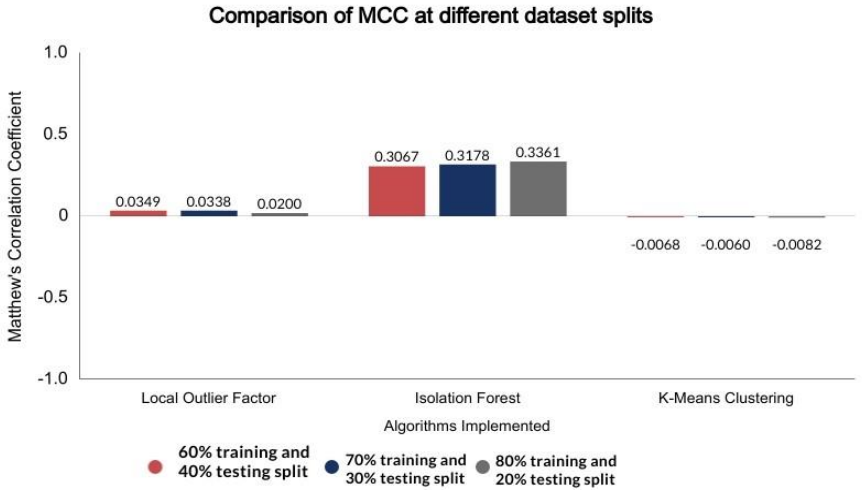

Fig 3: Comparison of MCC of various algorithms at different dataset splits

As shown in Fig 3, Performed experiment indicates that Matthews Correlation Coefficient(MCC) which is considered as one of the most reliable statistical rates for binary classification problems has the highest value for isolation forest algorithm while local outlier factor having values just higher than 0. K-means clustering algorithm has the least MCC value reaching beyond 0 on the negative scale.

Comparison of Sensitivity at different dataset splits

Algorithms Implemented	60% training set 40% testing set	70% training set 30% testing set	80% training set 20% testing set
Local Outlier Factor	0.998320	0.998323	0.998294
Isolation Forest	0.998927	0.998862	0.998927
K-Means Clustering	0.540246	0.538999	0.539323

Table 2: Comparison of Sensitivity of various algorithms at different dataset splits

From the shown Table 2, observation is that local outlier factor and isolation forest algorithm have comparable sensitivity value nearing to 1 while k-means clustering having relatively low sensitivity value in comparison to both the other algorithms.

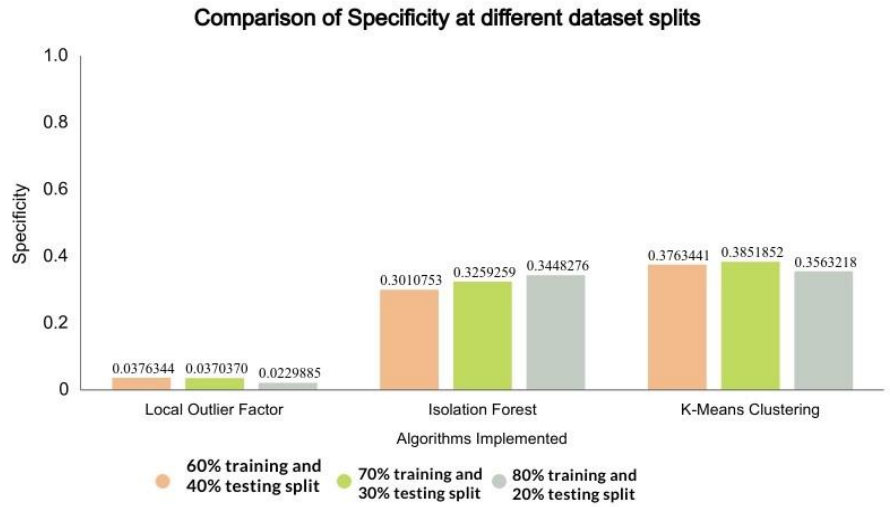

Fig 4: Comparison of Specificity of various algorithms at different dataset splits

As shown in Fig 4, Specificity indicates accuracy of negative case classification in which k-means clustering has the maximum value in each split while isolation forest fared quite well but local outlier factor has the minimum specificity value.

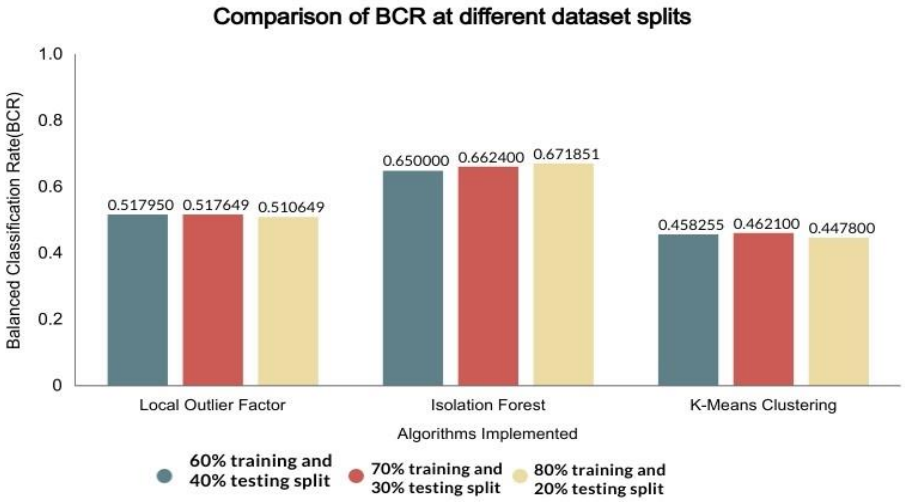

Fig 5: Comparison of BCR of various algorithms at different dataset splits

As shown in Fig 5, BCR indicates the amount of negatives categorised as negative in which isolation forest performed better than both other algorithms while k-means clustering having the minimum BCR.

4.3 Phase 2: Undersampling and oversampling

The second phase of our experiment starts with first selecting a random split of the dataset, which we have taken as 67% training set and 33% testing set. In the chosen training set, we obtain the count of fraudulent and non-fraudulent credit card transactions. The next steps are to undersample and oversample the dataset is as follows:

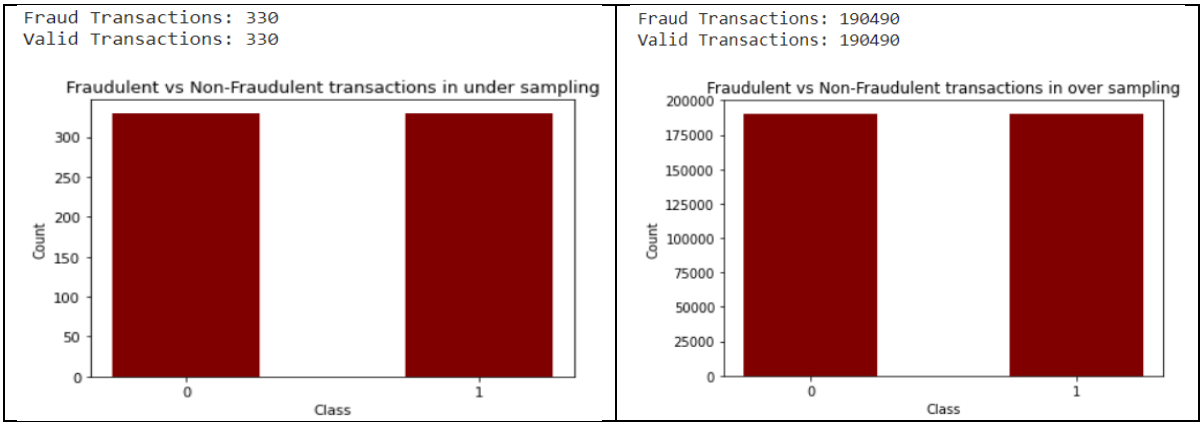

Fig 6: Fraudulent and Non-Fraudulent transactions in under sampling and over sampling

As indicated in Fig 8, in undersampling, we sample the dataset by making the number of fraudulent transactions equal to the count of non-fraudulent transactions in our chosen training dataset. The undersampled dataset we have obtained is as shown above. In oversampling, we sample the dataset by making the number of non-fraudulent transactions equal to the count of fraudulent transactions in our chosen training dataset. The oversampled dataset we have obtained is as shown above:

The next step is to evaluate and compare the different performance metrics by fitting all three models on the undersampled and oversampled datasets. The comparative analysis of performance is presented as shown below:

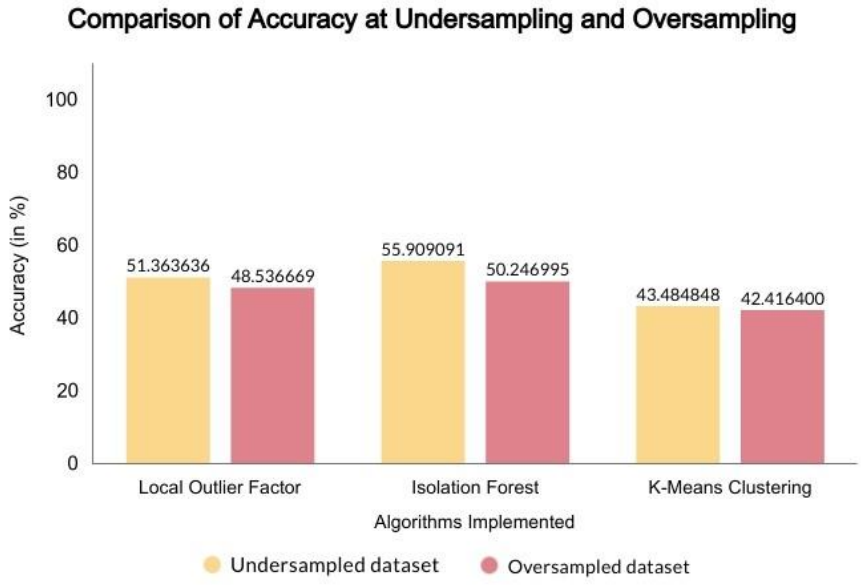

Fig 7: Comparison of Accuracy of various algorithms at undersampling and oversampling

As indicated in Fig 7, in both sampled datasets isolation forest has maximum accuracy and k-means clustering has the minimum accuracy.

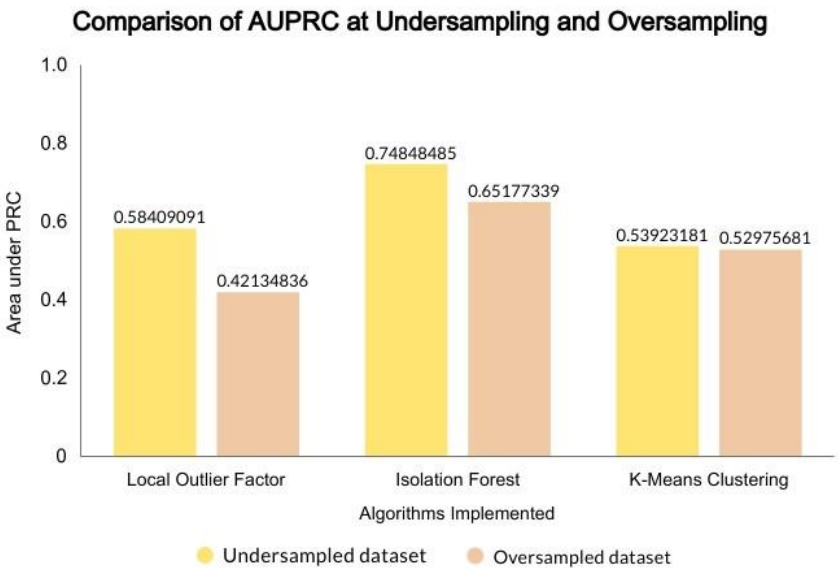

Fig 8: Comparison of AUPRC of various algorithms at undersampling and oversampling

As represented in Fig 8, Isolation forest outperforms both other algorithms in terms of area under precision-recall curve while k-means clustering has the lowest value in undersampling while local outlier factor having the lowest area under curve in oversampling.

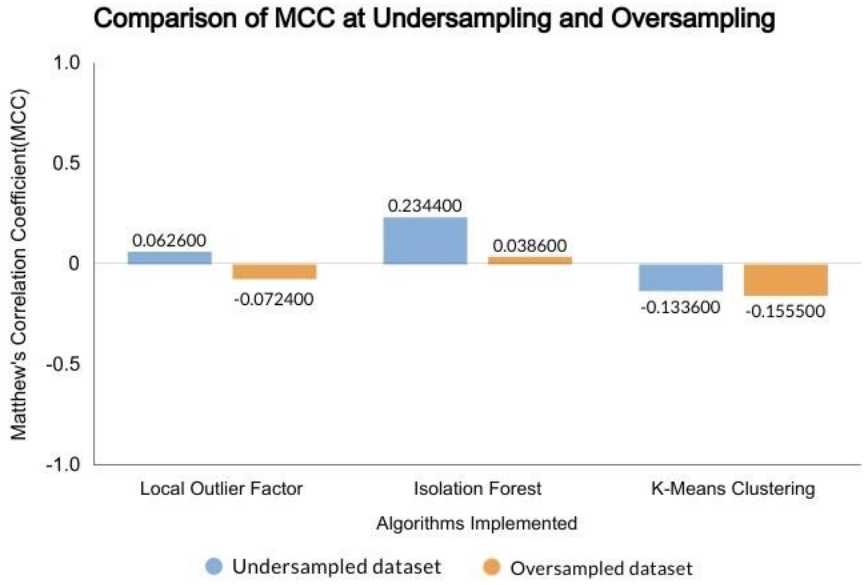

Fig 9: Comparison of MCC of various algorithms at undersampling and oversampling

As shown in Fig 9, Maximum MCC at both the sampled datasets is of isolation forest while k-means clustering having the minimum MCC.

4.4 K-Fold Cross Validation - Cross-validation is a statistical procedure used to estimate the skill of machine learning models. The technique has one argument called k referring to the number of sets that a

given dataset is to be split into. Cross-validation is a machine learning technique that is used to assess the performance of a machine learning model on previously unseen results. It's a common method because it's straightforward and produces a less skewed or positive estimation of model ability than other approaches, such as a simple train/test split. The value of k is set to 5 as it has been found through experimentation that k=5 results in model evaluation with a low bias. The models are then trained on the dataset splits obtained after the k-fold split and parameters are evaluated by cross_validation_score and mean values are taken into consideration. The comparative analysis of k-fold cross validation is presented below:

Comparison of Accuracy at k-fold cross validation

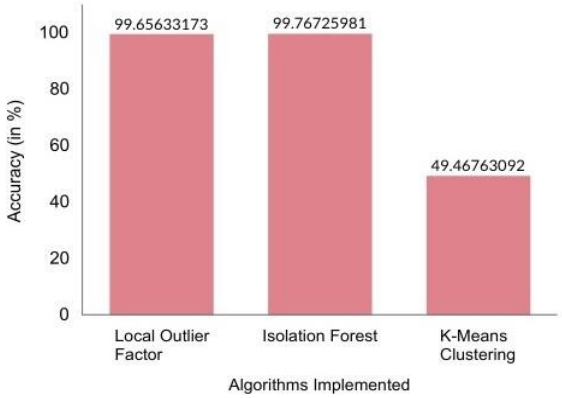

Fig 10: Comparison of Accuracy of various algorithms at k-fold cross validation

As represented in Fig 10, accuracy of local outlier factor and isolation forest have found to be comparable with isolation forest being ahead by a small amount while k-means clustering has minimum accuracy.

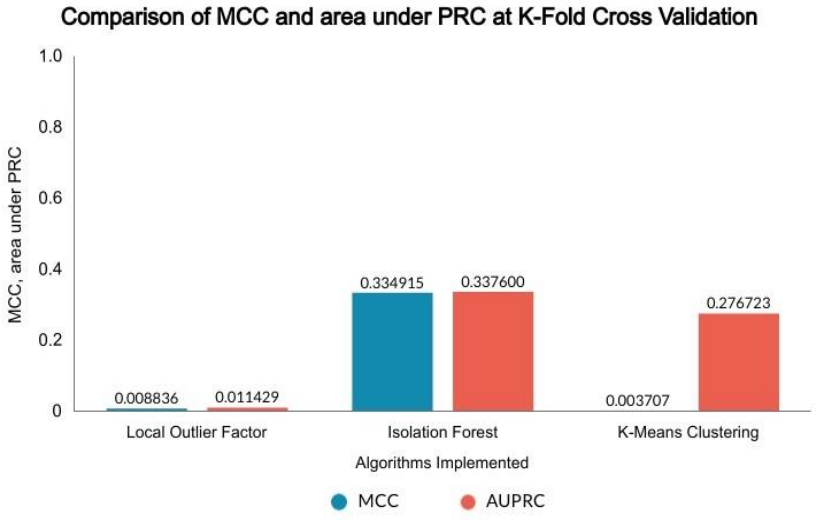

Fig 11: Comparison of MCC and AUPRC of various algorithms at k-fold cross validation

As shown in Fig 11, isolation forest outperforms the other models in both area under precision recall curve and Matthews Correlation Coefficient while k-means clustering has the least MCC and local outlier factor has minimum area under precision recall curve.

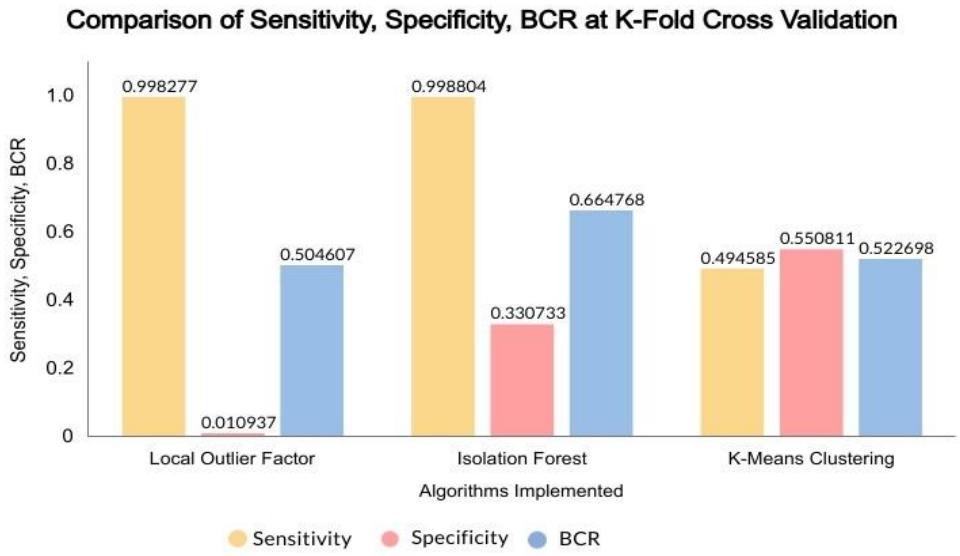

Fig 12: Comparison of Sensitivity, Specificity, BCR of various algorithms at k-fold cross validation

As represented in Fig 12, The performance of models based on sensitivity gives us a view of local outlier factor and isolation forest being comparable while k-means clustering is least performing. Coming to specificity k-means clustering outperforms the other two with local outlier factor being the least performing. The BCR value for isolation forest is also the maximum and the other two having just comparable values with local outlier factor being the least performing.

4.5 Hyperparameter Tuning

For Local Outlier Factor, the two important hyperparameters are n_neighbours and contamination. The value of n_neighbours is found out to be 20 by grid search approach for obtaining the optimum parameters. The value of contamination is set to the value of outlier_fraction for our dataset. The tuning of these two hyperparameters in Local Outlier Factor gives us the best performance which is verified experimentally.

For Isolation Forest, the first hyperparameter to be tuned is the max_samples which is the number of random samples it will pick from the original data set for creating isolation trees, the value of max_samples should be as large as possible in accordance with the splitted dataset so as to get the proper number of isolation trees for classifying the transactions. So in each iteration the value of max_samplesis set to the size of training set of that split. The contamination parameter here is set to the outlier_fraction for our dataset to obtain maximum performance.

For K-Means Clustering, the first hyperparameter tuned is init which is the method for initialization. The value of init is set to k-means++ which selects initial cluster centers for k-mean clustering in a smart way to speed up convergence. The performance is better than what was obtained by setting init to random and this is verified experimentally. The other hyperparameter is the n_clusters which is the number of clusters to form as well as the number of centroids to generate. The optimal value of n_clusters is obtained by calculating the Silhouette Coefficient and the optimal value is found out to be 2.

4.6 System Specification

In our work, we have used Google Colab platform to perform the experiments. Colab platform supports all python notebook files. It also enables us to use and share Jupyter notebooks without the need to download, install, or run something. It includes a completely optimised machine learning runtime as well as free access to a powerful GPU. The accelerated runtime hardware is comparable to a standard workstation and a powerful Linux server with 20 physical cores.

- **OS:** Ubuntu 17:10
- **System:** 64 bit
- **Processor:**Intel(R) Xeon(R)
- **Processing Speed:** 2.30 GHz
- **Memory:** 12 GB
- **Development Language:** Python 3.6

Certain different libraries are present in python to perform operation on the dataset and apply machine learning algorithms. The ones we have used in our experimentation are as follows:

- **Numpy:** It is the core library for scientific computing in python providing a high-performance multidimensional array object and tools for working with these arrays.
- **Pandas**It is an open-source library for Python that provides high-performance, user-friendly data structures and data analysis tools**.**. It allows for fast data cleaning and preparation as well as analysis.
- **Matplotlib:** It's a Python library that lets you make static, animated, and interactive visualisations. It allows you to build. With only a few lines of code, you can build publication-quality plots.
- **Seaborn**:It's a matplotlib-based Python data visualisation library. It has a high-level gui for creating visually appealing and insightful statistical graphics**.**
- **SciPy:** It is a collection of mathematical algorithms and utility functions based on the Python NumPy extension. It gives the user a lot of control by providing high-level commands and classes for manipulating and visualising data in an interactive Python session.
- **Sklearn:** Scikit-learn offers a consistent Python framework for a variety of supervised and unsupervised learning algorithms. The library is based on the SciPy programming language (Scientific Python).
- **Plotly**: It offers individuals and groups online graphing, analytics, and statistics resources, as well as analytical graphing libraries for Python.

5. Overall Result

In the results section, we conclude and summarize the results obtained after performing the experiment in the mentioned two phases. Table 1 summarises the results.According to the results obtained we see that Isolation Forest algorithm outperforms both Local Outlier Factor and K-means Clustering in the various performance metrics such as accuracy, area under PRC, Matthews correlation coefficient, balanced classification rate etc. Local Outlier Factor has accuracy and sensitivity values comparable to that of Isolation Forest but fails miserably in the important metrics such as area under PRC, MCC, sensitivity,

specificity and BCR. K-Means Clustering being the baseline method, performs poorly in the case of accuracy and MCC but outperforms Local Outlier Factor in area under PRC, sensitivity,

Comparison of algorithms for performance metrics based on each split

MODEL	TRAIN: 60%, TEST: 40%	TRAIN: 70%, TEST: 30%	TRAIN: 80%, TEST: 20%
LOCAL OUTLIER FACTOR	ACCURACY: 99.6752	ACCURACY: 99.6805	ACCURACY: 99.6805
	AUCPRC: 0.0373	AUCPRC: 0.0362	AUCPRC: 0.0223
	MCC: 0.0349	MCC: 0.0338	MCC: 0.0200
	Sensitivity: 0.9983	Sensitivity: 0.9983	Sensitivity: 0.9983
	Specificity: 0.0376	Specificity: 0.0370	Specificity: 0.0230
	BCR: 0.5180	BCR: 0.5176	BCR: 0.5106
ISOLATION FOREST	ACCURACY: 99.7788	ACCURACY: 99.7800	ACCURACY: 99.7928
	AUCPRC: 0.3084	AUCPRC: 0.3195	AUCPRC: 0.3377
	MCC: 0.3067	MCC: 0.3178	MCC: 0.3361
	Sensitivity: 0.9989	Sensitivity: 0.9989	Sensitivity: 0.9989
	Specificity: 0.3011	Specificity: 0.3259	Specificity: 0.3448
	BCR: 0.6500	BCR: 0.6624	BCR: 0.6719
K-MEANS CLUSTERING	ACCURACY: 53.9979	ACCURACY: 53.8757	ACCURACY: 53.9044
	AUCPRC: 0.1893	AUCPRC: 0.1937	AUCPRC: 0.1792
	MCC: -0.0068	MCC: -0.0060	MCC: -0.0082
	Sensitivity: 0.5402	Sensitivity: 0.5390	Sensitivity: 0.5393
	Specificity: 0.3763	Specificity: 0.3852	Specificity: 0.3563
	BCR: 0.4583	BCR: 0.4621	BCR: 0.4478
RECOMMENDED ALGORITHM	ISOLATION FOREST	ISOLATION FOREST	ISOLATION FOREST

Table 3: Comparison of algorithms for performance metrics based on each split

As shown in Table 3, after the careful evaluation of the various performance metrics at different splittings of the dataset, an overall performance result of all the three algorithms is obtained. It is experimentally observed that at each dataset split, isolation forest algorithm outperforms the other two algorithms in overall performance evaluation.

Comparison of Algorithms based on Performance Metrics

Performance Metrics	Local Outlier Factor	Isolation Forest	K-Means Clustering
Accuracy	Comparable accuracy to IF	Highest Accuracy	Lowest Accuracy
Area under PRC	Lowest area under PRC	Highest area under PRC	Area under PRC between LOF and IF
Matthews Correlation Coefficient	MCC score between IF and KMC	Highest MCC score	Lowest MCC score
Sensitivity	Comparable sensitivity to IF	Highest sensitivity	Lowest sensitivity
Specificity	Lowest specificity	Specificity between LOF and KMC	Highest specificity
Balanced Classification Rate	BCR between IF and KMC	Highest BCR	Lowest BCR

Table 4: Comparison of algorithms based on performance metrics

Presented in Table 4, a comparison of all the three algorithms based on performance metrics. It is obseverved that different algorithms have different performance based on evaluation metrics and every algorithm has some advantage over the other ones in some criterias but if overall performance evaluation is done, isolation forest outperforms the other two algorithms.

6. Conclusion

In our work, we evaluated performance of Local Outlier Factor, Isolation Forest Algorithm, and K-means clustering Algorithm for credit card fraud detection. The impact of hybrid sampling, K-fold cross validation and hyperparameter optimization on the efficiency of binary classification of unbalanced data is also demonstrated in our work. In our work, we demonstrated that using only one evaluation metric to assess imbalanced learning is not sufficient . The results of the experiment show that the Isolation Forest outperforms both Local Outlier Factor and K- means clustering in all of the metrics tested. Future research may look at meta-classifiers and meta-learning approaches for dealing with highly unbalanced credit card fraud data. One can explore use of ensemble methods and combining multiple algorithms into modules. We will build a Big Data-driven ecosystem and put our model to the test with help of bigger and variety of datasets.

REFERENCES

[1] “Credit Card Users in India 2020: A Comparison of Leading Brands”. [Online]. Available: https://www.globenewswire.com/news-release/2020/11/03/2119495/0/en/Credit-Card-Users-in-India-2020-A-Comparison-of-Leading-Brands.html. [Accessed: 07-May-2021].

[2] “Credit card fraud is on the rise due to Covid pandemic”. [Online]. Available: https://www.cnbc.com/2021/01/27/credit-card-fraud-is-on-the-rise-due-to-covid-pandemic.html. [Accessed: 07-May-2021].

[3] “Card Fraud Losses Reach $21.84 Billion in 2015 - The Nilson Report”. [Online]. Available: https://www.prweb.com/releases/creditcardfraud/2015/prweb13791784.htm. [Accessed: 07-May-2021].

[4] S. Makki et al., "Fraud Analysis Approaches in the Age of Big Data - A Review of State of the Art," 2017 IEEE 2nd International Workshops on Foundations and Applications of Self* Systems (FAS*W), 2017, pp. 243-250, doi: 10.1109/FAS-W.2017.154.

[5] Carcillo, Fabrizio et al. “Combining unsupervised and supervised learning in credit card fraud detection.” *Inf. Sci.* 557 (2021): 317-331.

[6] Cheng, Zhangyu et al. “Outlier detection using isolation forest and local outlier factor.” *Proceedings of the Conference on Research in Adaptive and Convergent Systems* (2019): n. pag

[7] U. Porwal and S. Mukund, "Credit Card Fraud Detection in E-Commerce," 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2019, pp. 280-287, doi: 10.1109/TrustCom/BigDataSE.2019.00045.

[8] A. Shen, R. Tong and Y. Deng, "Application of Classification Models on Credit Card Fraud Detection," 2007 International Conference on Service Systems and Service Management, 2007, pp. 1-4, doi: 10.1109/ICSSSM.2007.4280163.

[9] Sahin, Yusuf &Duman, Ekrem. (2011). Detecting credit card fraud by ANN and logistic regression. INISTA 2011 - 2011 International Symposium on INnovations in Intelligent SysTems and Applications. 10.1109/INISTA.2011.5946108.

[10] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, J. Christopher Westland, Data mining for credit card fraud: A comparative study, Decision Support Systems, Volume 50, Issue 3, 2011, Pages 602-613, ISSN 0167-9236, https://doi.org/10.1016/j.dss.2010.08.008. (https://www.sciencedirect.com/science/article/pii/S0167923610001326)

[11] J. O. Awoyemi, A. O. Adetunmbi and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," 2017 International Conference on Computing Networking and Informatics (ICCNI), 2017, pp. 1-9, doi: 10.1109/ICCNI.2017.8123782.

[12] Rajeshwari U and B. S. Babu, "Real-time credit card fraud detection using Streaming Analytics," 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), 2016, pp. 439-444, doi: 10.1109/ICATCCT.2016.7912039.

[13] Pozzolo, A. D., Caelen, O., Johnson, R. A., and Bontempi, G.,(2015). Calibrating Probability with Undersampling for Unbalanced Classification. *In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE*.

[14] “Credit Card Fraud Detection | Kaggle”. [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud. [Accessed: 07-May-2021].

[15] Liu, Fei Tony & Ting, Kai & Zhou, Zhi-Hua. (2009). Isolation Forest. 413 - 422. 10.1109/ICDM.2008.17.

[16] “Critical Analysis of K-Means Clustering - AI Objectives”. [Online]. Available: https://aiobjectives.com/2019/11/04/critical-analysis-of-k-means-clustering/. [Accessed: 07-May-2021].