**A hybrid of Isolation Forest algorithm and C4.5 Decision Tree algorithm for Supervised Anomaly Detection**

Dharavath Ramesh\*, Aarush Juneja and Pratyush Kesarwani

Department of Computer Science and Engineering

Indian Institute of Technology (Indian School of Mines), Dhanbad - 826004, India

**Email**: drramesh@iitism.ac.in\*, aarush.15je001424@cse.iitism.ac.in, pratyush.15je1100@cse.ism.ac.in

**Abstract:** Anomalies can be defined as patterns or data points that do not conform to a well-defined notion of normal behaviour.Anomaly detection is a popular research problem which caters the interest of a large amount of research scientists. It is a very important step in every good Data Mining framework. Several techniques involving one or more of the following fields, namely, Statistical Analysis, Machine Learning, Soft Computing, Deep Learning, Information Theory etc. have been used for making better anomaly detection systems. Anomaly detection finds its applications in various fields such as detecting malicious behaviour in online social media networks, detecting fraud in credit card transactions, fault detection systems etc.

Isolation Forest is the first model-based learning algorithm that is fundamentally different from other model-based algorithms in a way that it explicitly isolates anomalies while other previous models tend to construct models which first create a profile of non-anomalous instances, termed normal instances, and then classify the instances into being non-anomalous(normal) or non-anomalous.[1] The base estimator of Isolation Forest is an Isolation Tree which is a binary decision tree, which uses randomization to select a splitting attribute at each node and then chooses a random value between the minimum and maximum value of the splitting attribute as the split value. We have replaced the base estimator with the C4.5 Decision tree[2] which uses Shannon’s entropy[3] based information gain to decide on the splitting attribute, and as per our observations, it outperforms the Isolation Tree based Isolation Forest ensemble. The anomaly score evaluation of Isolation Forest algorithm has been used, thus making the algorithm a hybrid of Isolation Forest and C4.5 decision trees(also called J48 decision trees). Also, isolation trees follow unsupervised learning approach while C4.5 use supervised learning approach, thus bringing the isolation trees mathematics to supervised learning.

**Keywords:** Anomaly detection, ensemble learning, supervised machine learning, isolation forest, decision tree.

**1. Introduction:** Outliers of anomalies are extreme values that deviate from other observations on data , they may indicate a variability in a measurement, experimental errors or a novelty. In other words, an outlier is an observation that diverges from an overall pattern on a sample. There are two kinds of outliers, Univariate and Multivariate. Univariate outliers refers to looking for outliers in a single feature distribution and Multivariate outliers refers to looking for outliers in a n-feature distribution. Modern day anomaly detection techniques are designed keeping in mind the Multivariate outliers. Common cause of outliers include data entry errors, measurement errors, data processing errors, intentional errors etc. Many anomaly detection techniques that build a profile of normal instances first tend to provide only a binary categorization, i.e. whether an instance is an anomaly or not. A general utility expected from a good anomaly detecting mechanism is to be able to measure the degree to which an instance is an anomaly.

The standard Isolation Forest algorithm as well as the hybrid Isolation Forest Algorithm that we have proposed both assign anomaly scores to all the instances in the dataset such that the more the anomaly score, higher is the chance of being an anomaly.

Anomaly detection problem can be of the following variants[6]:

1. Given a data set **D**, find all the data points **x** ∈ **D** with anomaly score greater than some

pre-decided or calculated threshold **t**.

2. Given a data set **D**, find all the data points **x** ∈ **D** having **top-n** anomaly scores.

3. Given a data set **D**, containing mostly normal(but unlabeled) data points, and a test point **x**,

compute the anomaly score of **x** with respect to **D**.

The cases 1 and 2 above can be dealt using a supervised learning algorithm if the dataset is labeled. Case 3 has unlabeled data and thus requires an unsupervised learning algorithm.

The algorithm we have proposed is a supervised learning algorithm. So, it can be used for cases 1 and 2 when the dataset **D** has labels of classes to which each of the instance **x** ∈ D, belongs to.

For the cases where the dataset **D** is unlabeled, unsupervised learning algorithms based on clustering[7], or unsupervised neural networks such as Hebbian Learning based networks[8] may be used.

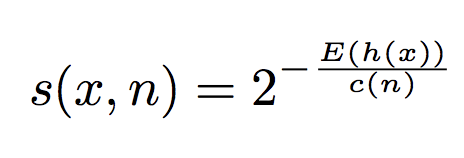
Fraud detection is a topic applicable to many industries including banking and financial sectors, insurance, government agencies . Fraud are on a rise in recent years, making fraud detection more important than ever. Despite efforts on the part of the affected institutions, hundreds of millions of dollars are lost to fraud every year. Since relatively few cases show fraud in a large population, detecting fraud is tricky. Isolation forest is an effective method for fraud detection. Isolation forest’s basic principle is that outliers are few and are far from the rest of the observations. Frauds, are outliers too. Isolation Forest explicitly prunes the underlying isolation tree once the anomalies are identified. We have observed that C4.5 Decision Trees perform better than Isolation Trees while using the underlying expected value mathematics of the Isolation Forests in the ensemble built using C4.5 Decision Trees.

In this paper, we describe a hybrid algorithm that uses Isolation Forests and C4.5 decision trees to efficiently assign anomaly scores, and then we do binary classification by setting a certain threshold, after which we are able to measure the recall metric with respect to anomalies as a measure of our proposed model’s performance with respect to other models. The content of the paper is divided into the following sections: **Section 2** describes the related work we went through as literature review while working on the problem of anomaly detection. **Section 3** describes the Isolation Forest algorithm. **Section 4** describes the C4.5 Decision Tree algorithm.

**Section 5** describes the Improved Isolation Forest + C4.5 Decision Tree algorithm. **Section 6** describes the results on testing proposed algorithm, which clearly indicate the improvement in the Recall metric with respect to anomalies, and also shows the comparisons with other classifiers. At last, in **Section 7**, we conclude the paper and mention the future work that can be done on this paper.

2. **Related Work:** Anomaly detection has been a topic of research from several decades. Many researchers working on problems like intrusion detection, fault detection, transaction fraud detection etc. tend to employ various techniques, as we have observed on reviewing a lot of literature on anomaly detection. SVMs (Support Vector Machines) have been there from a long time as a very widely used tool for classification. Anomaly detection tends to be a by-product of all model based classifiers that profile normal instances first. C-SVMs were first used for classification of instances. But, C-SVMs tend to perform badly when data set is imbalanced[9], hence making them not suitable for anomaly detection. In this case, W-SVMs were used as they handle imbalanced datasets effectively. But, kernel models such as SVMs can can be quite sensitive to overfitting the model selection criterion[10]. One Class SVMs were also used for anomaly detection[10], but they tend to be suited better for unlabeled data only. Anomaly detection system using entropy based technique has been done[11], which uses Shannon’s Entropy for classification of instances into anomalies or not. The C4.5 Decision trees have been used as a part of several outlier detection systems[12] and are also currently the most widely used decision trees out of all the decision tree variants for classification problems.C4.5 decision trees have been observed to handle missing data better than the other popular variant, ID3 decision trees[13]. Decision Trees often tend to overfit. Many individual models fail because of the same issue. Ensemble Learning is one such technique which is used to minimize the overfitting issue. Isolation Forests[1] was the first ensemble based supervised learning model that was dedicated to anomaly detection and gave anomaly scores to each test instance, and not just a binary classification. Hence, it is different from other models that build a profile of normal instances first. The application of Isolation forests for anomaly detection was demonstrated by Fei Tony Liu and Kai Ming Ting[5] and it was shown that Isolation based anomaly detection outperforms ORCA [Bay and Schwabacher 2003], one-class SVM [Scholkopf et al. 2001], LOF [Breunig et al. 2000] and Random Forests (RF) [Shi and Horvath 2006]. In this paper, we take C4.5 Trees as the base model and build an ensemble upon it. We use the concept of Isolation as described in the Isolation Forest algorithm, and use the mathematical formulations for computing the anomaly scores, as described in the Isolation Forest algorithm. It has been observed that this hybrid algorithm produces better Recall score than Isolation Forests having Isolation Trees as the base model.

**3. Isolation Forest Algorithm:** Isolation Forest is an ensemble based supervised learning algorithm for anomaly detection.The underlying estimator of an Isolation Forest ensemble is Isolation Tree. It is a binary tree such that each node has either 2 children or no child. In these trees, partitions at each node are created by first randomly selecting a feature out of the entire feature set and then selecting a random split value between the minimum and maximum value of the selected feature. The instances with value of the selected attribute less than the split value form the left child of the node and the instances with value of the selected attribute higher than the split value form the right child of the node. This goes on until all instances of a node belong to the same class, or the max depth of the tree is reached. In principle, outliers are less frequent than regular observations and are different from them in terms of values (they lie further away from the regular observations in the n-dimensional vector space). This is why by using such random partitioning they should be identified closer to the root of the tree (shorter average path length, i.e., the number of edges an observation must pass in the tree going from the root to the terminal node), with fewer splits necessary. As with other outlier detection methods, an anomaly score is required for decision making. In case of Isolation Forest it is defined as:

 (1)

where ***h(x)***is the path length of observation *x*, ***c(n)*** is the average path length of unsuccessful search in a Binary Search Tree having n nodes. **E(h(x))** is the average of h(x) in the entire ensemble of Isolation Trees.

For a data set having n instances, according to the section 10.3.3 of [4], the average path length of unsuccessful search in a BST(Binary Search Tree) is:

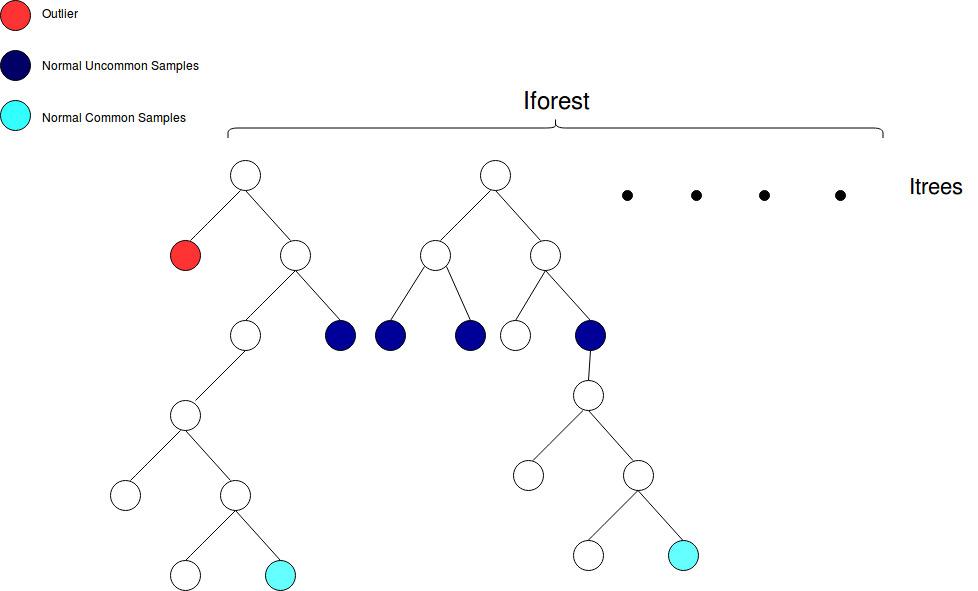
**c(n) = 2H(n − 1) − (2(n − 1)/n) (2)**

where **H(i)** is the harmonic number and it can be estimated

by **ln(i) + 0.5772156649 (Euler’s constant).**

Each observation is given an anomaly score and the following decision can be made on its basis:

* A score close to 1 indicates that the instance is an anomaly
* A score much smaller than 0.5 indicates normal observations
* If the scores of all the instances are close to 0.5 than the entire sample does not seem to have clearly distinct anomalies



**Fig 1**: Isolation Forest. Two trees of the ensemble are shown for understanding.

The following represents the algorithm used for building an Isolation Forest ensemble.

**Algorithm 1 : iForest(X, t,** 𝜳**)**  
**Inputs:** **X** - input data, **t** - number of trees, 𝜳 - sub-sampling size  
**Output:** a set of **t** iTrees  
1: Initialize Forest  
2: set height limit l = ceiling(log2𝜳)  
3: for i = 1 to **t** do  
4: X0 ← sample(**X**, **ψ**)  
5: Forest ← Forest ∪ iTree(X0 , 0, l)  
6: end for  
7: return F orest

The following represents how an Isolation Tree is formed.

**Algorithm 2 : iT ree(X, e, l)**

**Inputs:** **X** - input data, **e** - current tree height, **l** - height limit

**Output:** an iTree

1: if e ≥ l or |X| ≤ 1 then

2: return exNode{Size ← |X|}

3: else

4: let Q be a list of attributes in X

5: randomly select an attribute q ∈ Q

6: randomly select a split point p from max and min values of attribute q in X

7: X\_left ← filter(X, q < p)

8: X\_right ← filter(X, q ≥ p)

9: return inNode{Left ← iT ree(X\_left , e + 1, l),

10: Right ← iT ree(X\_right, e + 1, l),

11: SplitAtt ← q,

12: SplitValue ← p}

13: end if

The following algorithm defines the procedure to be used for computing the anomaly score of a test instance, based on a single isolation Tree. The average of path lengths given by using the following procedure for each Isolation Tree in the Isolation Forest is **E(h(x))**. This value when put in equation (1) gives the anomaly score for this instance of data.

**Algorithm 3 : PathLength(x, T, e)**

Inputs : **x** - an instance, **T** - an iTree, **e** - current path length;

to be initialized to zero when first called

Output: path length of x

1: if T is an external node then

2: return e + c(T.size) {c(.) is defined in Equation 1}

3: end if

4: a ← T.splitAtt

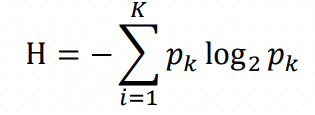
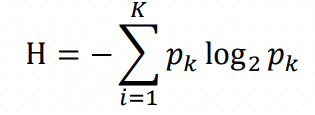
5: if x a < T.splitValue then

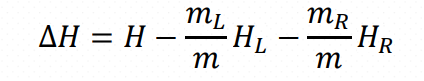
6: return PathLength(x, T.left, e + 1)

7: else {x a ≥ T.splitValue}

8: return PathLength(x, T.right, e + 1)

9: end if

**4.** **C4.5 Decision Tree Algorithm:** C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan. It is an improvement to ID3 algorithm. C4.5 is similar to ID3 as it uses the concept of information entropy. Information entropy is the average rate at which information is produced by a stochastic source of data. There are several mathematical variants of entropy. The one used in C4.5 Decision Tree is the Shannon’s Entropy is defined as :  The root node of the tree contains the entire dataset. At each node of the decision tree, C4.5 picks the feature of the data that most effectively divides its set of samples into subsets enriched in one class or the other. The splitting is based on normalized information gain (difference in entropy). Information gain (IG) measures how much “information” a feature gives us about the class. – Features that perfectly partition should give maximal information. Features which are unrelated shouldn’t give any information. With entropy defined as: 

Information Gain, is defined as: where 𝑚 is the total number of instances, with 𝑚𝑘 instances belonging to class 𝑘, where 𝐾 = 1, … , 𝑘. The attribute with the maximum normalized information gain is picked to make the decision. C4.5 algorithm then recurses on the partitioned sublists.

**Base cases :**

* When all samples belong to the same class it simply creates a leaf node for the decision tree saying to choose that class.
* If not even a single feature gives information gain then C4.5 creates a decision node higher up the tree using the expected value of the class.
* If an instance of previously unseen class is encountered then also C4.5 creates a decision node higher up the tree using the expected value.

Pseudocode :  
1. First check the above base cases.  
2. For each feature **a**, calculate the normalised information gain by splitting on **a**  
3. Suppose **a\_best** is the attribute which gives the highest normalized information gain.  
4. A decision node is to be created based on **a\_best**.  
5. Recur on the sublists obtained by splitting on **a\_best**, and add those nodes as children of node.

Advantages over ID3 variant of Decision Trees:

1. It can handle both continuous and discrete attributes to handle continuous attributes, C4.5 decies a threshold value and then splits the list into those whose attribute value is above the threshold value and those that are less than or equal to it.
2. It can handle training date with missing attribute values, missing values are simply not used in calculation of Information Gain and Entropy values.
3. It does Pruning of tree after its creation. C4.5 attempts to remove branches that do not help by replacing them with leaf nodes.

**5. Hybrid algorithm:**  In this approach, we have improved upon the standard Isolation Forest which is an ensemble of Isolation Trees, by replacing the base estimator (base model) of the ensemble with the mostly widely used variant of decision trees, namely C4.5 Decision Trees.   
  
The following algorithm represents the outline of the overall algorithm. It is used to build the ensemble from C4.5 trees:   
  
**Algorithm 4: Hybrid-iForest(X, t,** 𝜳, **limit)**  
**Inputs:** **X** - input data, **t** - number of trees, 𝜳 - sub-sampling size, **limit -** the maximum height of each C4.5 decision tree in the ensemble(can be predefined by the user, though there is a default value depending upon the sample size, as explained in the algorithm)  
**Output:** a set of **t** iTrees

1: Initialize Forest  
2: If **limit** is not specified, set height limit l = ceiling(log2𝜳)  
3: for i = 1 to **t** do  
4: X0 ← sample(**X**, **ψ**)  
5: Hybrid-Forest ← Hybrid-Forest ∪ C4.5-Decision-Tree(X0 , None, 0, l)  
6: end for  
7: return Hybrid-Forest

The following algorithm builds the C4.5 Decision Tree, which is the base model for the ensemble:

**Algorithm 5: C4.5-Decision-Tree(X, parent, height, limit)**

**Input: X** - input data, **parent** - the parent node of the node from the tree is further being developed, **height** - the height till the current node, **limit -** the maximum height of each C4.5 decision tree in the ensemble(can be predefined by the user, though there is a default value depending upon the sample size, as explained in the algorithm)

**Output:** a C4.5 Decision Tree

1. Node = C4.5-Node(True, None, None, None, **parent**, None, None, 0)

2. If **parent** is not None, then

3. Node.height = Node.parent.height + 1

4. end if

5. if all records in **X** belong to same class, then

6. Node.type = ‘Leaf Node’

7. Node.class = Class of any record in X /\* Because class of all records is same \*/

8. return Node

9. else

10. Node.type = ‘Internal Node’

11. end iF

12. overall\_maximum\_info\_gain = 0

13. overall\_split\_attribute\_index = None

14. overall\_split\_value = None

15. min\_info\_gain = 0.1 /\* Threshold \*/

16. entropy = Calculate-entropy(X)

17. For attribute\_index = 0 to Number\_of\_attributes, do

18. attribute\_max\_info\_gain = 0

19. attribute\_split\_value = 0

20. attribute\_value\_list = List of unique values of attribute values, for ith attribute of all

records in X

21. If length(attribute\_value\_list) > 100, then do

22. Sort(attribute\_value\_list)

23. new\_list = list of 10 attribute values such that the 10 values are equally spaced

in the attribute\_value\_list which is now sorted (see line 23)

24. attribute\_value\_list = new\_list

25. End if

26. For each value in attribute\_value\_list, do

27. current\_gain = Calculate\_information\_gain(**X**, attribute\_index,

28. value, entropy)

29. If current\_gain > attribute\_max\_info\_gain, then do

30. attribute\_max\_info\_gain = current\_gain

31. attribute\_split\_value = value

32. End if

33. End For

34. If attribute\_max\_info\_gain > overall\_max\_info\_gain, then do

35. overall\_max\_info\_gain = attribute\_max\_info\_gain

36. overall\_split\_attribute\_index = attribute\_index

37. overall\_split\_value = attribute\_split\_value

38. End if

39. End For

40. /\* Check if thresholds have been reached \*/

41. If overall\_max\_info\_gain <= min\_info\_gain OR node.height >= limit, then do

42. node.is\_leaf\_node = True

43. node.classification = Classify\_leaf(**X**)

44. return node

45. End if

46. node.attribute\_split\_index = overall\_split\_attribute\_index

47. node.attribute\_split\_name = name of the attribute with the index as

**overall\_split\_attribute\_index**

48. node.attribute\_split\_value = overall\_split\_value

49. Make datasets for left and right node using the overall\_split\_attribute\_index and

overall\_split\_value. The records in **X** with value at the splitting attribute higher than the

overall\_split\_value will be present in **X\_left** and the records with

the value at the splitting attribute lesser than or equal to the overall\_split\_value will be

present in **X\_right**.

50. node.left\_child = C4.5-Decision-Tree(X\_left, parent = node, node.height, limit)

51. node.right\_child = C4.5-Decision-Tree(X\_right, parent = node, node.height, limit)

52. return node

To evaluate any test instance **x** on a particular C4.5 Decision Tree, **Algorithm 3**  will be used, which is same as the algorithm used to evaluate any test instance on a particular Isolation Tree.

To evaluate any test instance **x** on the entire ensemble, **Algorithm 3,** is run on all the C4.5 Decision Trees in the ensemble and then the **equation (1)** will be used to generate the overall anomaly score of the instance **x**.

The utility functions used in Algorithm 5 are given below:

The following algorithm is used to calculate the entropy of a given dataset, based on the class labels.

**Algorithm 6: Calculate-entropy(X)**

Inputs : **X** - the dataset whose entropy is to be calculated

Output: Shannon’s entropy of the dataset

1. Ones = Number of records belonging to anomaly class

2. Zeroes = Total rows - Ones

3. Entropy = 0

4. P(Ones) = Ones/(Ones+Zeroes)

5. P(Zeroes) = Zeroes/(Ones+Zeroes)

6. If P(Ones) != 0, then do,

7. Entropy = Entropy + P(Ones)\*log2(P(Ones))

8. End If

9. If P(Zeroes) != 0, then do,

10. Entropy = Entropy + P(Zeroes)\*log2(P(Zeroes))

11. End If

12. return Entropy

The following algorithm is used to calculate the information gain obtained on splitting the dataset present on a particular node of the C4.5 decision tree. The dataset is split on a particular attribute’s particular split value as explained in Algorithm 6.

**Algorithm 7: Calculate\_information\_gain(X, attribute\_index, split\_value, entropy)**

Inputs : **X** - the dataset which is to be splitted at a particular Node,

**attribute\_index** - the index of the splitting attribute in the attribute list

**split\_value**  - the split value of the splitting attribute

**entropy** - the entropy of the dataset **X**

Output: Information gain on the splitting **X**  at the splitting value of the splitting attribute

1. L = Total records in X

2. New\_entropy = 0

3. Make datasets for left and right node using the **attribute\_index** and

**split\_value**. The records in **X** with value at the splitting attribute higher than the

value will be present in **X\_left** and the records with

the value at the splitting attribute lesser than or equal to the **split\_value** will be

present in **X\_right**.

4. New\_entropy = New\_entropy + **calculate\_entropy(X\_left)**

5.New\_entropy = New\_entropy + **calculate\_entropy(X\_right)**

6. return entropy - New\_entropy

The following algorithm is used to assign a class label to a leaf node of the C4.5 Decision Tree

**Algorithm 8: Classify\_leaf(X)**

Inputs : **X** - the dataset present at the leaf node, for which class label is to be decided

Output: Label of the Node

1. Ones = Number of records belonging to anomaly class

2. Zeroes = Total rows - Ones

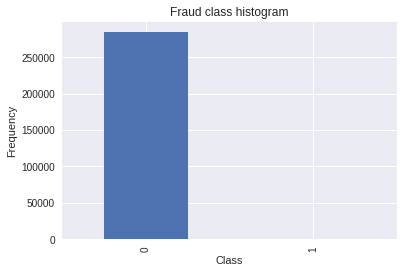
3. If Ones >= Zeroes, then return 1

4. Else return 0

5. End If

**6.** **Results of the proposed algorithm**

The algorithm was tested on credit card transactions by credit cards in September 2013 by european cardholders. The dataset is so imbalanced that the positive class (frauds) account for 0.172% of all transactions. It has the class label 1 in the dataset. The dataset has been collected and analysed during a research collaboration of Worldline and the Machine Learning Group (<http://mlg.ulb.ac.be>) of ULB (Université Libre de Bruxelles) on big data mining and fraud detection. The fraudulent transactions represent anomalies/outliers, and thus form the anomalous class in the labeled dataset.



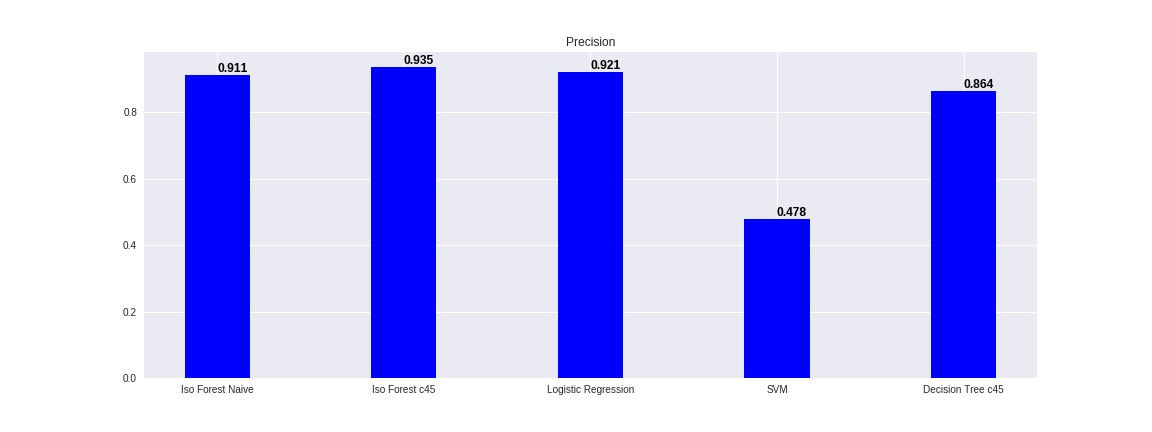
**Fig:** The above figure represents the imbalance

of the dataset. This dataset presents transactions

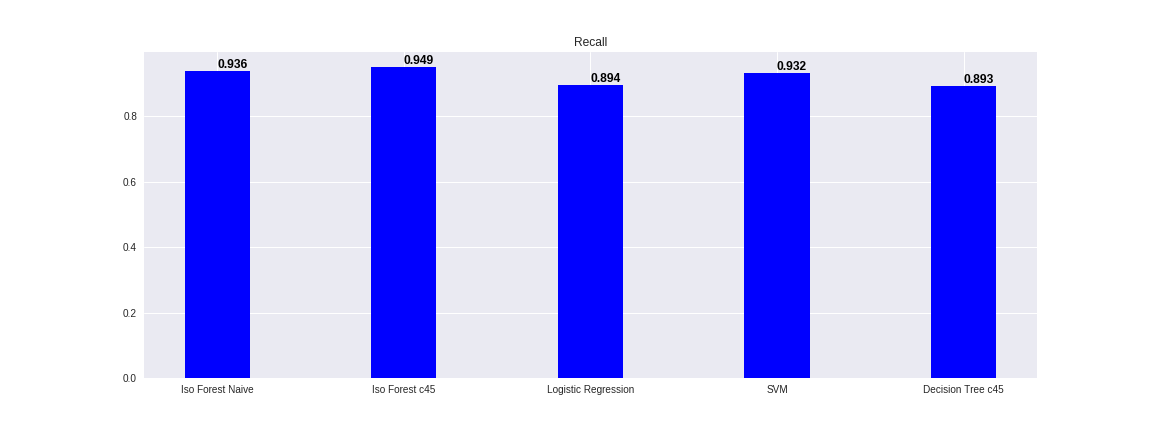
that occurred in two days, where we have 492

frauds out of 284,807 transactions.

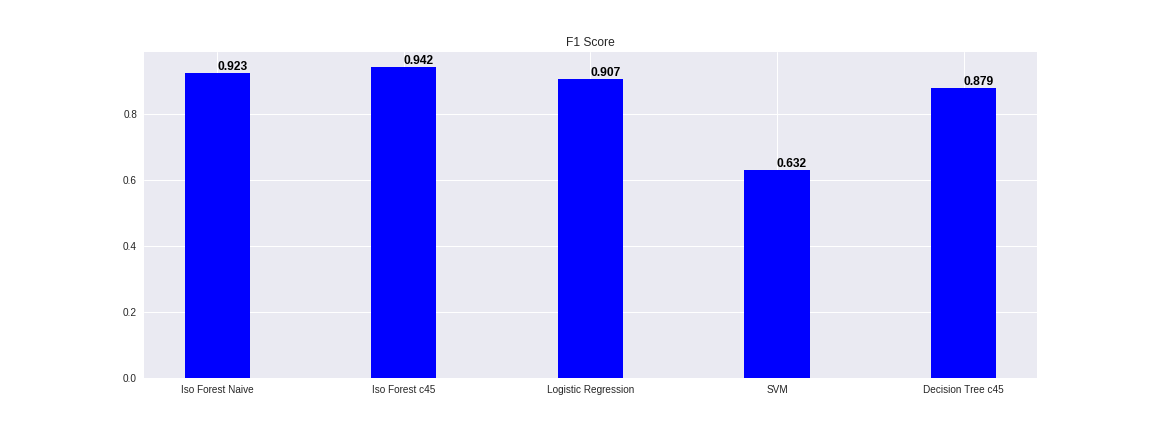
The data, after all the data preprocessing was tested on 5 models - Logistic Regression, C4.5 Decision Trees, Support Vector Machines(SVMs), Isolation Forests with Isolation Trees as the Base Estimator, and our own enseble, having C4.5 Decision Trees as the base estimator for Isolation Forests. Our method performed better than all the methods with respect to the precision score as well as recall score, both w.r.t. anomalous class. The following 3 figures show the comparison of Precision w.r.t anomalous class, Recall w.r.t. anomalous class and the corresponding F1 score w.r.t anomalous class. As is evident from the histograms, our proposed algorithm performs slightly better than the Isolation Forests with Isolation trees.

****

**Fig: Comparison of Precision score wrt Anomalous class**

****

**Fig: Comparison of Recall score w.r.t. Anomalous class**

****

**Fig: Comparison of F1 score w.r.t Anomalous class**

**7. Conclusions and future work**

The approach we provided performs better than the Isolation Forest with isolation trees, which is generally the most used approach for outlier detection. The Isolation Forest works on unlabeled datasets but our approach brings isolation forest’s mathematics to labelled datasets

as replacing isolation trees with C4.5 decision trees brings supervised learning into the picture.

Several methods to calculate entropy exist, which can be tried in the future instead of Shannon’s entropy, for C4.5 Decision trees. Using Tsallis Entropy to combine the effects of Gini Impurity and Shannon’s entropy in constructing decision trees may further improve the results.

**References:**

[1] Liu, F. T.,Ting, K. M.,and Zhou, Z.-H.2008a. Isolation Forest. InICDM ’08: Proceedings of

the 2008 Eighth IEEE International Conference on Data Mining. IEEE Computer

Society,413–422

[2] Quinlan, J. R.1993. C4.5: Programs for Machine Learning (Morgan Kaufmann Series in

Machine Learning), 1 ed. Morgan Kaufmann

[3] A mathematical theory of communication, C. E. Shannon.

(https://ieeexplore.ieee.org/document/6773024)

[4] B. R. Preiss. Data Structures and Algorithms with Object-Oriented Design Patterns in Java.

Wiley, 1999.

[5] Isolation-based Anomaly Detection, Fei Tony Liu and Kai Ming Ting (Gippsland School of

Information Technology, Monash University) and Zhi-Hua Zhou (National Key Laboratory

for Novel Software Technology, Nanjing University)

[6] Introduction to Data Mining (Second Edition), Pang-Ning Tan, (Michigan State University),

MichaelSteinbach (University of Minnesota) AnujKarpatne (University of Minnesota),

Vipin Kumar (University of Minnesota).

[7] Clustering-Based Anomaly Detection Approaches, Kishan G. Mehrotra, Chilukuri K.

Mohan, HuaMing Huang

[8] Multiscale Hebbian Neural Network for Cyber Threat Detection, Sana Siddiqui,

Muhammad Salman Khan, Ken Ferens Electrical and Computer Engineering, University of

Manitoba, Winnipeg, MB, Canada

[9] Support Vector Machines for Anomaly Detection, Xueqin Zhang, Chunhua Gu and

Jiajun Lin

[10] G. C. Cawley and N. L. C. Talbot, Over-fitting in model selection and subsequent

selection bias in performance evaluation, Journal of Machine Learning Research, 2010.

Research, vol. 11, pp. 2079-2107, July 2010.

[11] An Entropy-Based Network Anomaly Detection Method Przemysław Berezinski,

Bartosz Jasiul, and Marcin Szpyrka

[12] A Combination Classification Algorithm Based on Outlier Detection and C4.5, ShengYi

Jiang and Wen Yu School of Informatics, Guangdong University of Foreign Studies,

510006 Guangzhou, Guangdong

[13] A comparative study of decision tree ID3 and C4.5, Badr HssinA, Abdelkarim

Merbouha, Hanane Ezzikouri, Mohammed Erritalitiad laboratory, Computer Sciences

Department, Faculty of sciences and techniques, Sultan Moulay Slimane University