1.Introduction

1.1 Overview

Banknotes are one of the most important assets of a country. Some miscreants introduce fake notes which bear a resemblance to original note to create discrepancies of the money in the financial market. It is difficult for humans to tell true and fake banknotes apart especially because they have a lot of similar features. Fake notes are created with precision, hence there is need for an efficient algorithm that accurately predicts whether a banknote is genuine or not.

1.2 Purpose

The purpose of this project is to build a supervised machine learning model , performed on the datasets, we visualize and detect fake notes automatically.

2.Literature Survey

2.1 Existing Problem

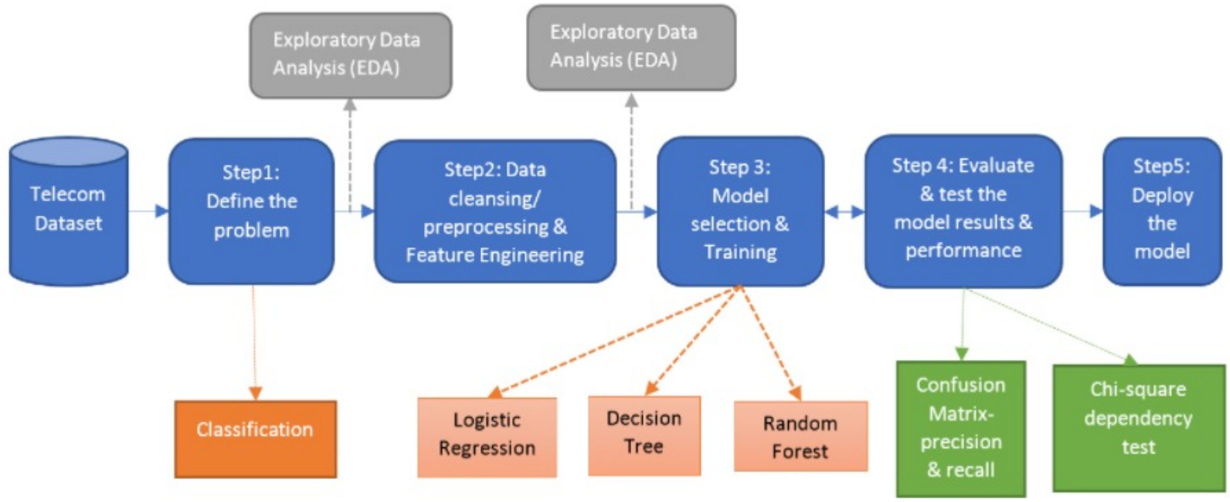
The existing system is inefficient because there is limited set of identification marks for the authentication of physical notes and it is very inefficient approach to authenticate.

2.2 Proposed Solution

Creating a supervised machine learning model, performed on the datasets, we visualize and detect fake notes automatically and the model is accurate to high extent and is frequently trained and fetches better result.

3.Theoritical Analysis

3.1 Block diagram



3.2 Hardware/Software designing

<u>Software Requirements:</u>

Eclipse IDE
WEKA
Excel

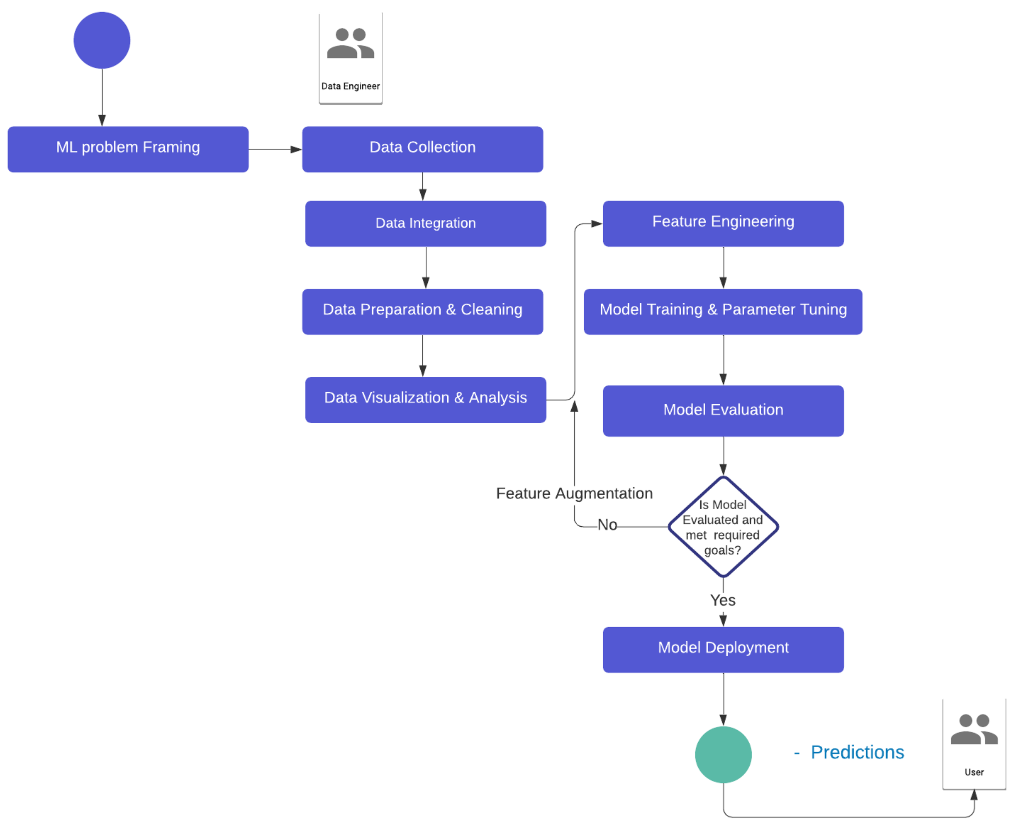
<u>Hardware Requirements:</u>

Computer (Windows/MAC/Linux)
Processor -intel i3 or above
Camera
RAM - 4GB or above
HDD/SSD

4.Experimental Investigations

Investigation and research is done during the building of model and help from college faculty has been considered and also oracle academy team has supported in lot of ways and helped us through conceptual knowledge. The following book has been referred.
Journal of Machine Learning Research,2010 .Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas Rückstieß, and Jurgen schmindhuber, Pybrain
https://towardsdatascience.com/machine-learning-in-java-e335b9d80c14

5.FlowChart



6.Result

```
Train dataset size823
** Logistic Regression Evaluation with Datasets **

Correctly Classified Instances         270               98.1818 %
Incorrectly Classified Instances         5                1.8182 %
Kappa statistic                          0.9636
Mean absolute error                      0.0164
Root mean squared error                  0.1057
Relative absolute error                  3.2918 %
Root relative squared error             21.0539 %
Total Number of Instances              275

Confusion matrix:
[139.0, 4.0]
[1.0, 131.0]
-------------------
Area under the curve
0.9995232040686586
-------------------
[Correct, Incorrect, Kappa, Total cost, Average cost, KB relative, KB information, Correlation, Complexity 0, Complexity scheme, Complexity improvement, MAE, RMSE
Recall :0.99
Precision:0.97
F1 score:0.98
Accuracy:0.98
-------------------
Predicted label:
1.0
```

7.Advantages & Disadvantages

Advantages:

Efficient over convectional methods of authentication of bank notes.

Disadvantages:

Camera quality plays a crucial role and poor camera quality lead to failure of authentication even when the model is efficient.

The model requires frequent change or updation of datset, else it looses efficiency over time.

8.Applications

Model can be applied where note authentication is required like banks, ATMs, retail shops etc.

9.Conclusion

After analyzing various techniques used to detect forged banknotes, this paper presents banknote authentication for recognizing the banknote as genuine or fake by using supervised learning techniques. Extensive experiments have been performed on banknotes dataset using model to find the best model suitable for classification of the notes.
By Using Logestic Regression we able to detect that our model detects 97% of data correcly.

10.Future Scope

Our future work will be concentrated on extraction of features from various currency notes belonging to different countries as well as recognition and classification. Our future scope will be conversion of currency denomination.

11.Bibilography

Eugen Gillich and Volker Lohweg, “Banknote Authentication”, 2014.
Kaggle
https://www.vshsolutions.com/blogs/banknote-authentication-using-machine-learning-algorithms/

Appendix

```java
package org.ml;

import java.util.Arrays;

import tech.tablesaw.api.Table;
import tech.tablesaw.plotly.Plot;
import tech.tablesaw.plotly.components.Figure;
import tech.tablesaw.plotly.components.Layout;
import tech.tablesaw.plotly.traces.BoxTrace;
import tech.tablesaw.plotly.traces.HistogramTrace;
```

```
import weka.classifiers.Classifier;
import weka.classifiers.evaluation.Evaluation;
import weka.core.Instance;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class DataAnalysis {

    public static Instances getInstances (String filename)
    {
        DataSource source;
        Instances dataset = null;
        try {
            source = new DataSource(filename);
            dataset = source.getDataSet();
            dataset.setClassIndex(dataset.numAttributes()-1);


        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();

        }

        return dataset;
    }

    @SuppressWarnings("static-access")
    public static void main(String[] args) throws Exception{
        System.out.println("Data Analysis");

        Table data =
Table.read().csv("K:\\Ritin\\Eclipse\\workspace\\org.ml\\src\\main\\java\\org\\ml\\data_banknote_authentication.csv");
        System.out.println(data.shape());
        System.out.println(data.first(5));
        System.out.println(data.last(5));
        System.out.println(data.structure());
        System.out.println(data.summary());

```

```java
        //Histogram
        Layout layout1 = Layout.builder().title("Distribution of Skewness").build();
        HistogramTrace t1 = HistogramTrace.builder(data.nCol("Skewness")).build();
        Plot.show(new Figure(layout1,t1));

        Layout l2 = Layout.builder().title("Class Distribution").build();
        BoxTrace t2 = BoxTrace.builder(data.categoricalColumn("Class"),data.nCol("Skewness")).build();
        Plot.show(new Figure(l2,t2));
System.out.println("-------------------------------------------------------------------------------------\n\n");

        Instances test_data = getInstances("K:\\Ritin\\Eclipse\\workspace\\org.ml\\src\\main\\java\\org\\ml\\testdata.arff");
        Instances train_data = getInstances("K:\\Ritin\\Eclipse\\workspace\\org.ml\\src\\main\\java\\org\\ml\\traindata.arff");
        System.out.println(train_data.size());

        //Building a classifier
        Classifier classifier = new weka.classifiers.functions.Logistic();

        classifier.buildClassifier(train_data);


        /**
         * train the algorithm with the training data and evaluate the
         * algorithm with testing data
         */
        Evaluation eval = new Evaluation(train_data);
        eval.evaluateModel(classifier, test_data);
        /** Print the algorithm summary */
```

```java
        System.out.println("* Logistic Regression Evaluation with Datasets *");
        System.out.println(eval.toSummaryString());
     //System.out.print(" the expression for the input data as per algorithm is ");
    //System.out.println(classifier);

        double confusion[][] = eval.confusionMatrix();
        System.out.println("Confusion matrix:");
        for (double[] row : confusion)
             System.out.println(    Arrays.toString(row));
        System.out.println("-------------------");

        System.out.println("Area under the curve");
        System.out.println( eval.areaUnderROC(0));
        System.out.println("-------------------");

        System.out.println(eval.getAllEvaluationMetricNames());

        System.out.print("Recall :");
System.out.println(Math.round(eval.recall(1)*100.0)/100.0);

        System.out.print("Precision:");
System.out.println(Math.round(eval.precision(1)*100.0)/100.0);
        System.out.print("F1 score:");
System.out.println(Math.round(eval.fMeasure(1)*100.0)/100.0);

        System.out.print("Accuracy:");
        double acc = eval.correct()/(eval.correct()+ eval.incorrect());
        System.out.println(Math.round(acc*100.0)/100.0);


        System.out.println("-------------------");
        Instance predicationDataSet = test_data.get(4);
        double value = classifier.classifyInstance(predicationDataSet);
```

```
        /** Prediction Output */
        System.out.println("Predicted label:");
        System.out.print(value);


    }

}
```