BANKNOTE AUTHENTICATION

Github repo link : https://github.com/prasuna5/Authentication.git

1. INTRODUCTION

1.1 Overview

The objective is to analyze the given datasets from the bank_authentication_notes.csv which is taken from kaggle datasets, is to identify the forged and real notes using Logistic Regression for real and forged notes. Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

1.2 Purpose

The ultimate aim is to use this dataset to train a machine to detect fake notes automatically. However, before implementation, it is important to access if this dataset can sufficiently distinguish forged banknotes from genuine ones. Hence, in this report, with , unsupervised machine learning, performed on the datasets, we will visualize and outline the results and make according to recommendations.

2. LITERATURE SURVEY

2.1 Exisiting Problem

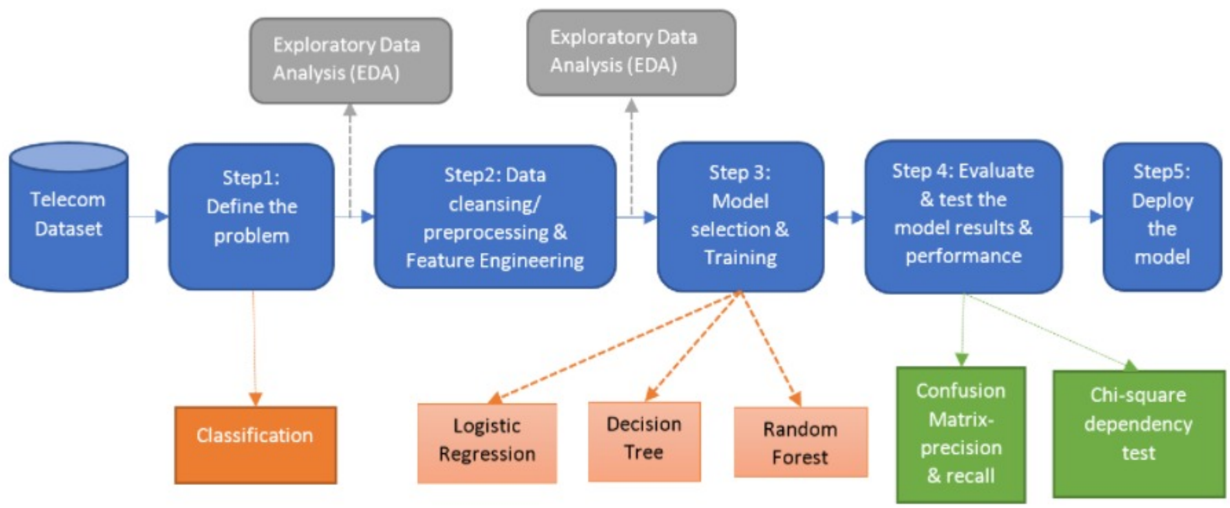
Preserving genuineness of higher denomination printed Banknotes is one of the critical issues. It has the major role in financial activities of every country . The study in evaluates different machine learning algorithms and concludes that Decision-Tree and MLP technique is best to classify a bank note. In , the features of the banknote are extracted using Fast Wavelet Transforms. Later, one-against-all classification approach was employed that classifies the note into four different categories: Genuine, High-Quality Forgery, Low-Quality Forgery, and Inappropriate ROI which resulted in 100% detection rate. Evaluation of SVM and BPN is done in and where BPN outperforms SVM. In , BPN gives 10% more accuracy than SVM and is determined as the best classifier for predicting proteins sequence based on their compositions, whereas in and , SVM outperforms BPN. The results depend on the dataset and the type of classification problem.

2.2 Proposed Solution

The research in implements a system for classifying Thai banknotes using Logistic regression. Firstly, a dataset is collected from Kaggle. Features are extracted from this data and inputted to classifier for learning and recognition. A new method is proposed for banknote recognition Logistic Regression and 98% success rate is obtained. The study uses classifier for banknote recognition. The experiment has been applied to Indian Rupees.

3. THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware/Software Designing

Software Requirements:

Eclipse

Weka

Excel

Hardware Requirements:

Computer

OS - Windows/MAC/Linux

Processor - Intel i3 or above

Ram - 4 GB

HDD - 1 TB

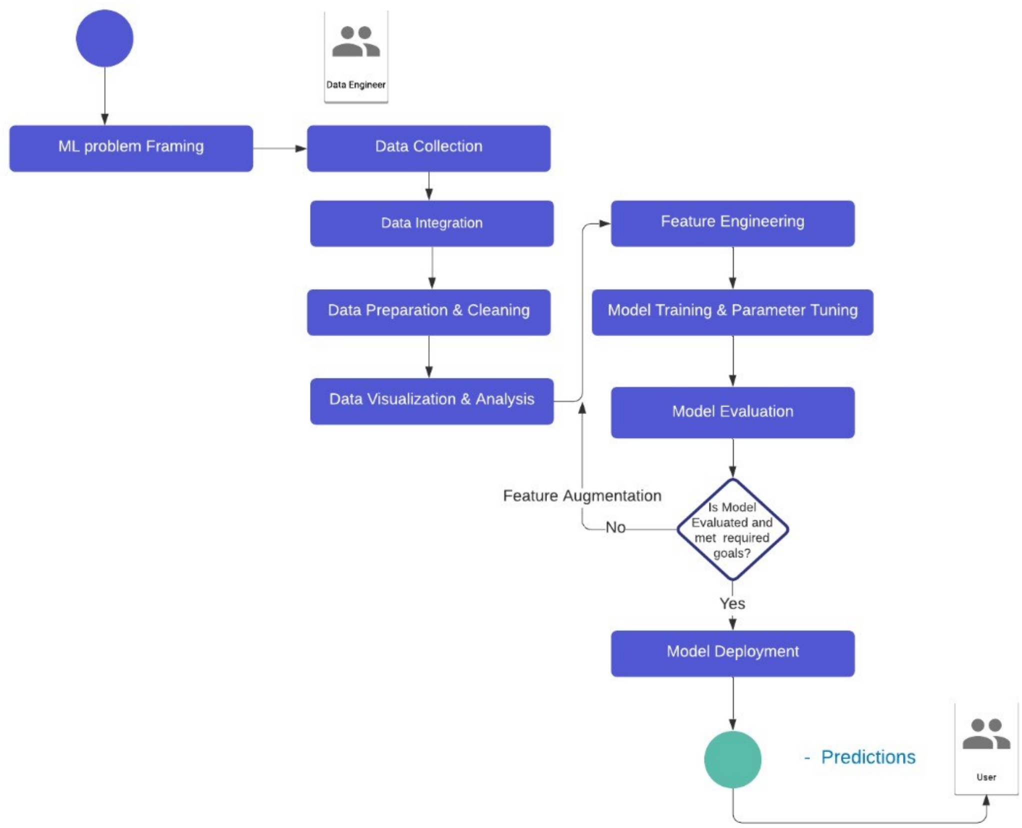
Programing Language:

Java

4. EXPERIMENTAL INVESTIGATIONS

- 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
```

The model is 97% accurate

7. ADVANTAGES AND DISADVANTAGES

Advantages

Efficient over conventional methods of authentication of bank notes.

Disadvantages:

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

The model requires frequent changes or updation of dataset or else the model will loose it's efficiency

11. BIBILOGRAPHY

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

12. APPENDIX

```
package authentication.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 Authentication {

 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;
```

```java
}

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

  Table data = Table.read().csv("java\\authentication\\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());

  //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");

```

```java
Instances test_data = getInstances("buildthon\\src\\main\\java\\authentication\\ml\\test_data.arff");
Instances train_data = getInstances("C:\\Users\\Welcome\\OneDrive\\Desktop\\oraclebuildthon\\src\\main\\java\\authentication\\ml\\train_data.arff");
System.out.println("Train dataset size"+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 */
System.out.println("** Logistic Regression Evaluation with Datasets **");
System.out.println(eval.toSummaryString());

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:");
```