CREDITCARD FRAUD PREDICTION

1. INTRODUCTION

1.1 Overview:

Technology is increasing day by day and in our daily life, credit card playing a major role in transactions. This credit card has many benefits and at the same time in these days many frauds are increased this fraud in credit card section becomes a major problem in current days. Now many companies and other websites changed and updated their payment ways with help of credit cards and parallel many frauds are increased in the online transactions. Therefore many people are trying to detect these frauds using many techniques and many researchers started working in machine learning algorithms to propose one that detect these fraud transactions. The main aim here is to propose a method that detects transactions which are fraud using machine learning techniques. We are going to consider the transactions which occurred in past or recent and we have to find some insights about this data and then we have to build a classifier that is used to find these kind of transactions which are fraud.

1.2 Purpose:

There are many uses benefitted from this project and we are going to achieve the below things:

1. The main objective here is to help companies and people by predicting whether the transaction performed using credit card is good one or fraud one.
2. We have to build a classifier using logistic regression algorithm to predict whether a given instance has properties which are fraudulent or not.
3. The main aim of the dataset used in this project is to early stage detection of fraudulent transactions and it will be very helpful to people and many E-commerce companies.
4. We can even drag this project and turn this into an application where people or companies can find the transactions whether they are good or bad.
5. Automated prediction in E-commerce field is the key that should be from this fraud prediction using Logistic Regression.

2.LITERATURE SURVEY

2.1 Existing problem:

There are many methods existing in literature that has been proposed to detect whether a particular instance of transactions is fraud or not.

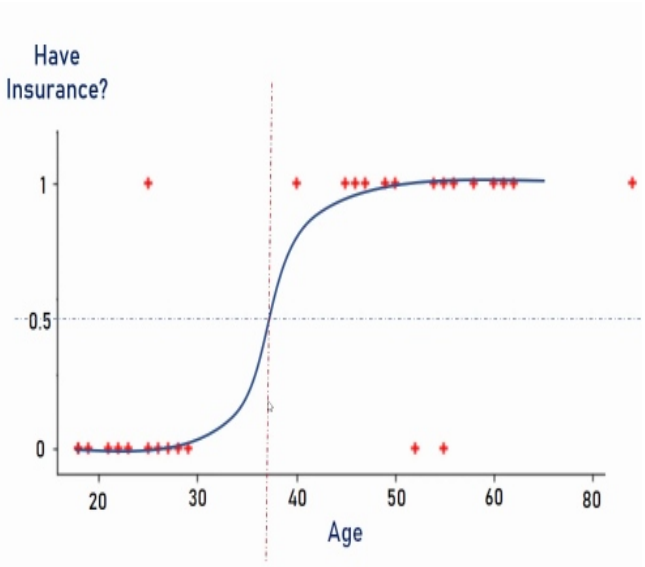
1. Zhejiang has been proposed a solution for this credit card fraud detection using outlier detection that is based on distance sum and here the outliers are nothing but frauds who behave abnormally than others.
2. Maniraj, Saini and Swarna proposed that fraud means which is unauthorized and that is used unwantedly other than owner of the card. They have proposed a technique that is related to isolation forest algorithm and local outlier factor.
3. Zahra Zojaji, Amir and Reza explained about classification techniques in two ways for fraud detection that is misuses which is based on supervised learning and detection of anomalies based on unsupervised learning.
4. Venkatrathnam proposed a method based on steam outlier detection which uses reverse K-nearest neighbors.

2.2 Proposed solution:

Here we are proposed a method that detects whether a transaction is fraud or not using Logistic Regression. Let us understand the algorithm based on small example. Consider an example will customer buy insurance or not and the answer here is yes/no and if we observe the below dataset here there are attributes age and have_insurance and if we carefully observe the data if the age increases he is more likely to buy an insurance and if the age is less then he is not interested in buying insurance and we can say 1-means interested and 0-means not interested. Now here we have to build a machine learning model that can do prediction whether they buy insurance or not.

We assign that if predicted value is >0.5 the person buy insurance and if it is <0.5 then he is not going to buy and Now if we imagine a line as shown in below figure it is much better and we can clearly say that this model works better and our problem here is related to classification and the predicted value is categorical. Logistic Regression is one of the techniques that is used for classification. Now let’s examine how to get this curve.

age	have_insurance
22	0
25	0
47	1
52	0
46	1
56	1
55	0
60	1
62	1
61	1
18	0
28	0
27	0
29	0
49	1



Now let examine how we get these line fitted to given data points. Here we are using a line equation as $h_\theta(x)=\theta_0+\theta_1 x$ which is used for mapping x values to y values. Here θ_0, θ_1 are parameters and we have to choose the values of this parameters such that the line fits best to given points and our value of hypothesis should nearly match the actual value y for given examples. Therefore our main aim is to decrease the difference between actual and predicted one and this is called as costfunction $J(\theta)$. What we have to do here is we start the model by giving different values to θ_0, θ_1 and we have to find the errors .Now the question arises in our mind how to make less the value of $J(\theta)$, here comes to rescue us that is nothing but gradient descent algorithm. The gradient descent algorithm is an algorithm that is very useful to cut down the value of cost function. What we have to done here is simply start training our model by taking some random values for θ_0, θ_1 and we have to change the values of these parameters in the sense of reducing the cost function $J(\theta)$ and we have to obtain the minimum value for it. If we observe the graph below where f(x) is $J(\theta)$ and the parameters on x-axis.

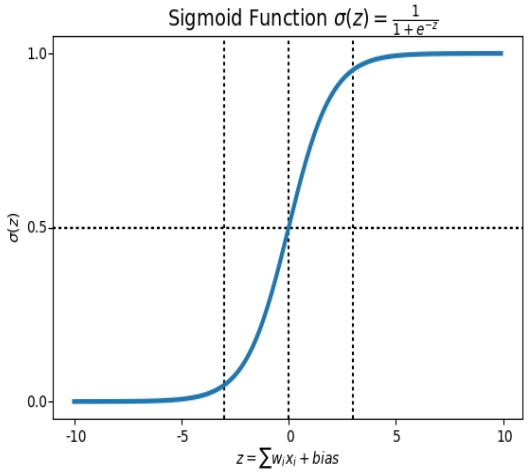
There are some terms in gradient descent algorithms like α, $\partial/\partial\theta\ J(\theta)$ where α is a learning rate and it is main parameter to model such that how fast it can learn and $\partial/\partial\theta\ J(\theta)$ is nothing but slope or gradient and how it is used is if the gradient is negative then the parameter moves towards right and if it is positive then the parameter is moving towards the left side as shown below in gradient descent module. If we carefully observe the graph shown below in gradient descent module for every iteration the value of cost is coming down and the value of cost comes to minimum after certain time and the convergence depends on parameters α such that if the value is very small then our algorithms is very slow to get the cost which is minimum

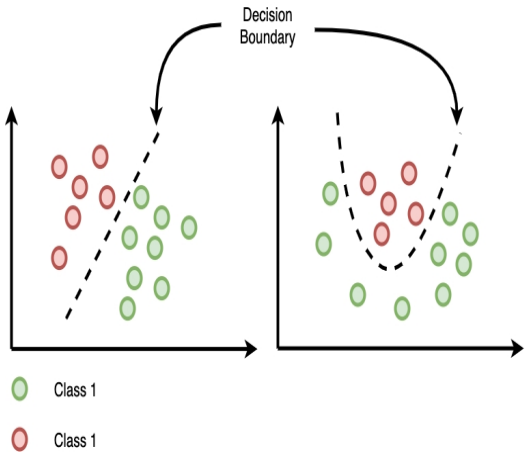
and it is very bad even if our α is very big then it may be impossible to get minimum cost because it may overshoot it.

Sigmoid function:

The curve is obtained by passing all values through sigmoid function before plotting. The sigmoid function maps any value between 0 and 1 and here we are dealing with classification and expects output as 0 or 1. Therefore our predicted value turns into probability after passing through sigmoid function.

$$f(z)=1/(1+e^{(-z)}$$





Hypothesis:

In linear regression we have used hypothesis as and here also we are using same hypothesis but there is a small change here. Instead of plotting the result from hypothesis we are plotting the values after passing through sigmoid function. Therefore our hypothesis becomes.

$$h_\theta (x)=1/(1+e^{(-(\theta_0+ \theta_1 x))})$$

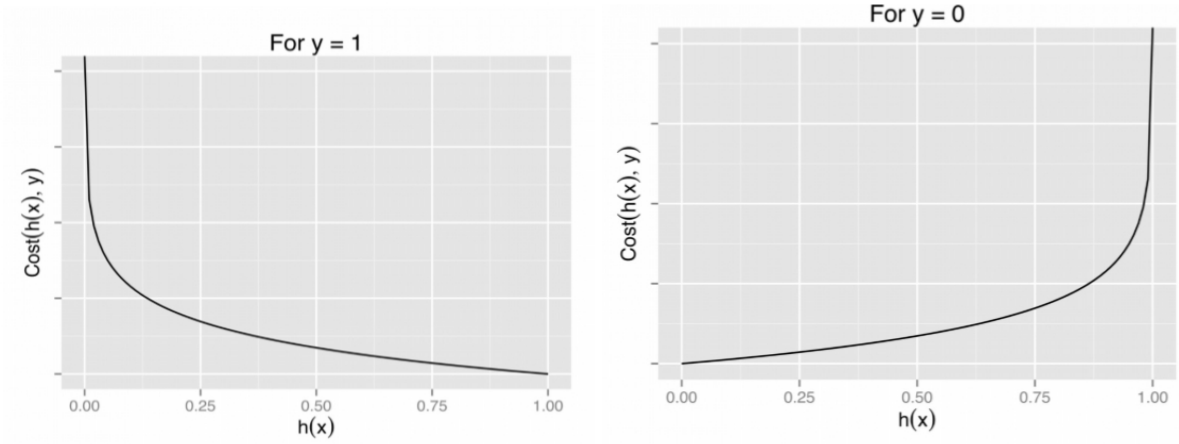
Decision boundary:

Here our model predicts a value between 0 and 1 because our value is passing through sigmoid function but our aim is to predict 0 or 1. So we have to set a threshold such that if we get a value above threshold we mark it as one class (1) and if we get a value below threshold then we put them in to second class (0). This threshold is nothing but a boundary that helps to take decisions.

Cost function:

If we use the cost function of linear regression then it will not get good results because due to intervention of sigmoid function here our graph of cost function may turns into non-convex and it has many local minima as shown in the figure below. So it is very hard to get global minimum and we can end up in any local minima and it is not we are expecting. Therefore for logistic regression the cost function can be written as

$$J(\theta)=-1/m \sum[y^{((i))} \log(h_\theta (x^{((i))})) + (1-y^{((i))}) \log((1-h_\theta (x^{((i))})))]$$



Gradient Descent:

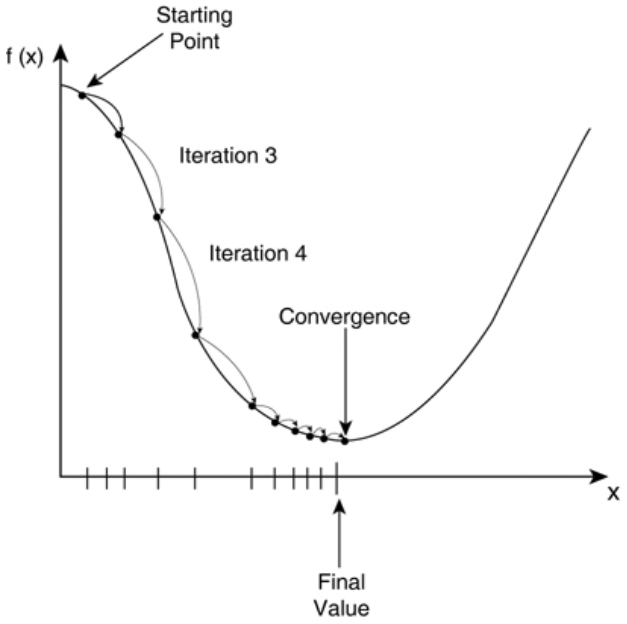
We have already discussed about gradient descent algorithm that is used to obtain minimum cost and we have to implement this on every parameter.

Repeat{

$$\theta_j:=\theta_j- \alpha\sum(h_\theta (x^{((i))}))-y^{((i))})x_j^{((i))}$$

}

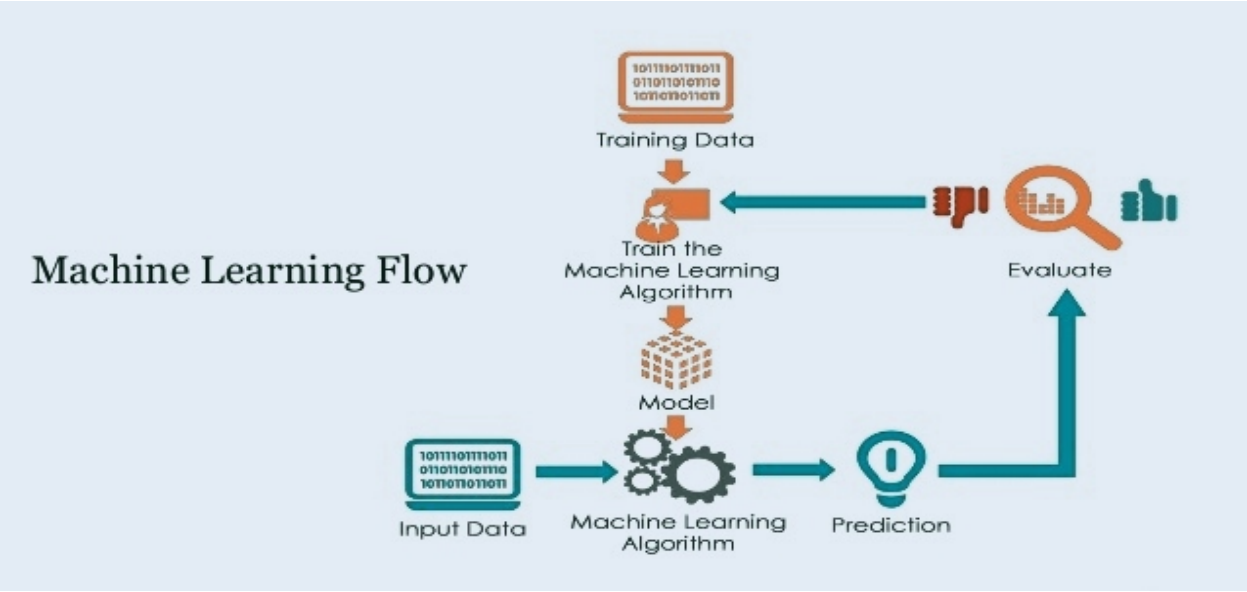
We have to update for j=0, 1 …n at a time and here ‘m’ is the number of training examples and ‘i’ is the training sample.



3.THEORITICAL ANALYSIS

3.1 Block diagram:

The below diagram shows the overview of how we are going to build the model.



3.2 Hardware / Software designing:

1. Hardware Requirements:

   Processor : minimum core i3 processor
   RAM : minimum 2GB or 8GB maximum
   Hard Disk Space : more than 50GB

2. Software Requirements:

   Operating System :Windows 8 or later versions of Windows
   Eclipse IDE , WEKA tool.

4. EXPERIMENTAL INVESTIGATIONS

Steps to build Machine Learning model:

Dataset:

Initially we have to collect the data from various sources and we have to collect the required data that is related to credit card fraud detection to build a classifier that predicts whether a particular instance given is fraud or not.

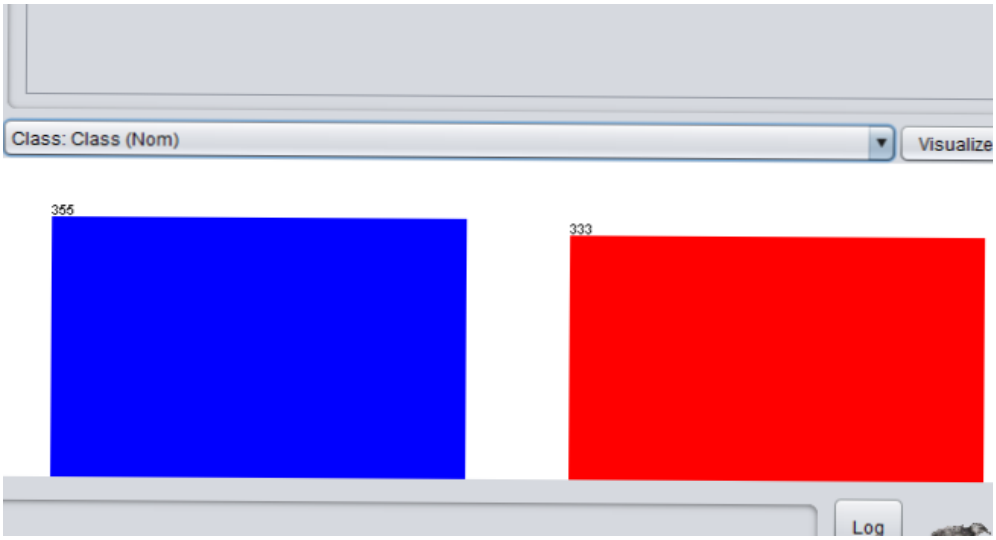
Data description:

In the dataset given there are nearly six hundred and eighty eight tuples or instances in training data set and there are two hundred and ninety six instances in testing set and we have thirty attributes. Using these attributes we have to find whether a given instance is fraud or not. The attributes and its related information are shown below.

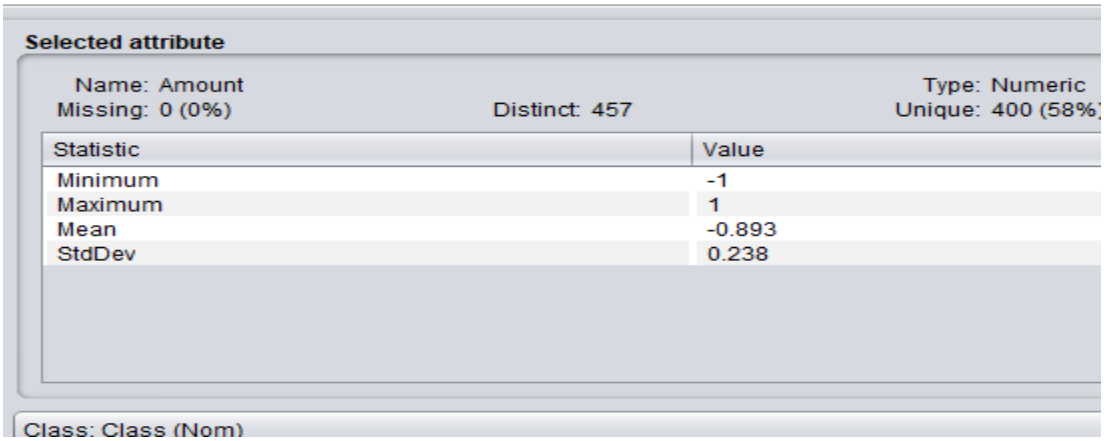
1. *Amount:* The amount of money paid in particular transaction.
2. *V1 to V28:* In the dataset there is no information given about these attributes because of privacy issues.
3. *Outcome*: it gives us the classification result that means if we get result as '0' means there is no diabetes and if we get '1' as result then there is a chance of getting diabetes.

Data Exploration and Data Visualization:

Now we can analyze the dataset by exploring it and knowing about features and we can get some details from it by using WEKA tool. We can visualize the dataset and there are nearly 355 instances who has output as ‘0’ means they are not fraudulent transactions and there are 333 tuples which outputs our class variable as ‘1’.



We can even get some measures like minimum value, maximum value and some statistic measures like mean, median and standard deviation etc. These are very helpful to understand the data and they are for each attribute or feature in given dataset. For example for attribute amount we can get these details.



We can individually examine each feature to know about it and we can even draw a histogram that tells about attribute.

Data Preprocessing:

There are many preprocessing techniques are there to build a good model.

1. *Balancing:* In real world many datasets are Imbalanced and due to this when we train our model using these kinds of datasets our model may show bias. Consider in real life the persons who got cancer are very less compared to normal people. There are some techniques to balance the dataset like oversampling which increases the instances that are present in smallest class. Undersampling decreases the instances that are present in largest class. Another manual technique is to adjusting the threshold.

2. *Missing Values*: There are many samples in our data that has missing values in columns. We have to check in the dataset that where the missing values are there. There are many techniques to handle this situation. We can delete the rows when we encountered missing values and we can even replace those missing values using some statistics. For example if we consider numeric data we can replace missing values with mean and if we take categorical data we can replace the missing values with most frequently occurring value that is mode.

3. *Normalization*: normalization is very useful to convert every attribute which is measured on different scales to single common scale. Z-score is one of the normalization techniques that are performed like this every value in the column is subtracted with average of that column and divided by standard deviation. Another technique in normalization is dividing each value in column by maximum value.

4. *Outlier detection*: In any dataset it is common to have outliers who are acting different from the other instances and having these outliers in our dataset may decrease our model accuracy and to get more accuracy what we have to do is to remove those abnormal behavioral instances. There are many techniques like extreme value analysis, statistical and probabilistic models etc.

5. *Standardization*: There is difference between normalization and standardization where standardization is used to change the data such that the mean of data is zero and root of variance is one and we already seen about normalization.

Splitting the dataset:

After the data preprocessing is done now we have to divide the dataset into training and testing datasets whereas training dataset is used to make the model learn and testing dataset is to verify our model such that how good it learns. Normally 70-80% of data is used for training purpose and 20-30% of data is used for testing the model.

Training the model:

After splitting of data the training data is used for training the model and it is used to make the model learn. We have already discussed that how to make the model learn data using gradient descent algorithm such that by minimizing the error. For each epoch the whole training dataset is passed through the model and updating the weights thereby decreasing the error. After training we get a model that is best fitted to the data that is used for training and now our model is ready.

Testing the model:

After training is completed what we have to do is with the help of testing data we have to test the trained model whether it is classifying our instances correctly or not. Here in our testing set we have nearly 137 instances who has output as '0' means they are not fraudulent transactions and there are 159 tuples which outputs our class variable as '1'.

Model Evaluation:

After testing phase we can evaluate our machine learning model based on its performance. Here there is a matrix called confusion matrix which is very helpful for this kind of job. Before moving to the performance measures we have to know about some terms. Consider covid-19 disease and machine learning model to know about the terms true negative, false positive, true positive and true negative.

1. *True Positive*: let a person have covid-19 disease it is considered as positive and our model predicts that it is true and this is called true positive.
2. *True Negative*: let a person have no covid-19 disease it is considered as negative and our model predicts that it is true and this is called true negative.
3. *False Positive*: let a person have no covid-19 disease it is considered as negative and our model predicts that it is false means our model wrongly tells that a person has covid-19 although it is negative sample and this is called false positive.
4. *False Negative*: let a person have covid-19 disease it is considered as positive and our model predicts that it is false means our model wrongly tells that a person has no covid-19 although it is positive sample and this is called false negative.

Now there are some evaluation measures like Accuracy, Specificity etc. let's see about them.

a. *Accuracy*: it can be also called as classifier recognition rate means how much percentage of tuples that are correctly classified by our model.

$$Accuracy=(TP+TN)/(TP+TN+FP+FN)$$

b. *Error Rate*: It can also be called as misclassification rate and it is opposite to accuracy.

$$Error\ rate=1-Acuuracy$$

c. *Sensitivity or Recall:* It is also called as rate of true positives or recognition rate means how many tuples that are positive are truly classified.

$$Sensitivity=TP/(TP+FN)$$

d. *Specificity*: It is also called as rate of true negatives means that how many tuples that belongs to negative class are truly classified as negative.

$$Specificity=TN/(FP+TN)$$

e. *Precision:* It can be also called as measure of exactness and it tells about that how many instances that belong to class positive are classified exactly as positive.

$$Precision=TP/(FP+TP)$$

f. *F1-score*: The evaluation measures recall and precision can be termed as single evaluation measure and that is called F-score or F1-score.

$$F=(2*precision*recall)/(precision+recall)$$

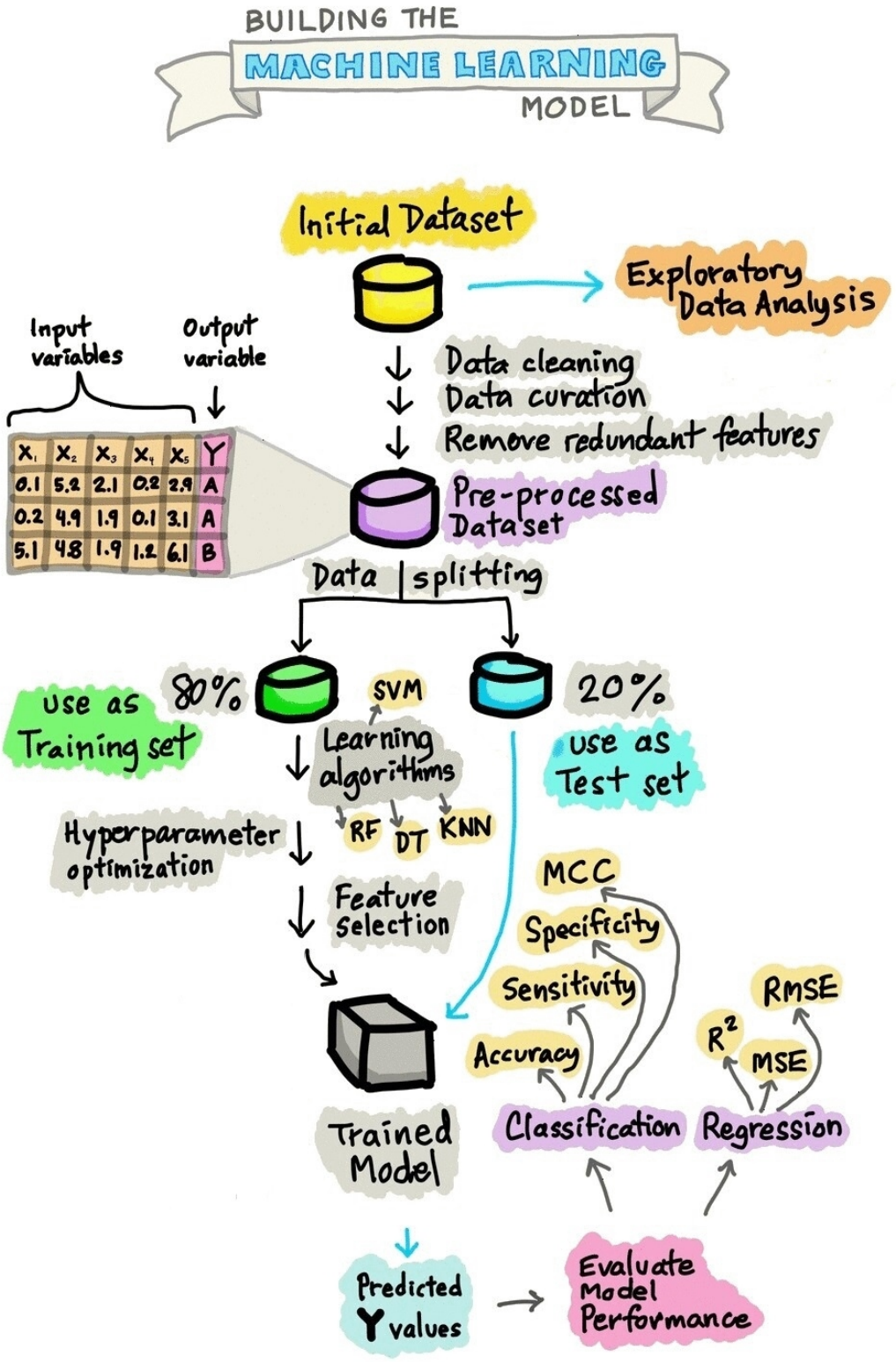
Saving the model:

We can save our machine learning model which is already trained for further use and when we want to classify new examples we can use this saved model by avoiding training again and again when we want to classify new tuples. This process of saving the model is called serialization and when we want to classify new instances we can perform deserialization.

Deploying the Model:

We can even build a user interface which takes input as features in the dataset and outputs result after passing features through classifier. This interface can useful to all and it can be used for model validation in future. We can even get feedbacks from user that our model will correctly predicting or not.

5.FLOW CHART


BUILDING THE
MACHINE LEARNING
MODEL
Initial Dataset
Exploratory Data Analysis
Input variables
Output variable
Data cleaning
Data curation
Remove redundant features
Pre-processed Dataset
Data splitting
80%
20%
use as Training set
use as Test set
SVM
Learning algorithms
RF
DT
KNN
Hyperparameter optimization
Feature Selection
MCC
Specificity
Sensitivity
Accuracy
RMSE
MSE
Classification
Regression
Trained Model
Predicted Y values
Evaluate Model Performance

6.RESULT

By using logistic regression algorithm we get nearly 94% accuracy and other evaluation metrics are shown in below figure.

```
Number of instances in Training set--688
Number of instances in Testing set--296

* Evaluation of Model with Testing dataset *


Correctly Classified Instances         277               93.5811 %
Incorrectly Classified Instances        19                6.4189 %
Kappa statistic                          0.872
Mean absolute error                      0.0751
Root mean squared error                  0.2168
Relative absolute error                 14.9879 %
Root relative squared error             43.2347 %
Total Number of Instances              296

--------------------------------------------------------------------------------
Confusion matrix:
[136.0, 1.0]
[18.0, 141.0]
-------------------
Area under the curve
0.980122113574806
-------------------
Recall :0.89
Precision:0.99
F1 score:0.94
Accuracy:0.94
-------------------
```

7.ADVANTAGES AND DISADVANTAGES

Advantages:

1. Logistic regression is very easy to implement.
2. The algorithm used here is very efficient to train and it takes less time.
3. Over fitting is rare in logistic regression.
4. When there is linearly separable data it works well.
5. It gives higher accuracy for small datasets.

Disadvantages:

1. Over fitting occurs if the features are greater than that of instances.
2. The relation between dependent and independent variables should be linear.
3. It doesn’t work well if there are complex relationships in data.
4. It is unable to handle huge number of categorical variables.

8. APPLICATIONS

1. Prediction of diabetes.
2. Loan Insurance prediction.
3. Predicting probability that a person may get heart attack or not.
4. Breast cancer prediction.
5. Spam detection.
6. Car Insurance Prediction.

9.CONCLUSION

In this we have seen how to detect frauds in credit card transactions using Logistic Regression algorithm. This is very useful for people and companies to detect fraudulent transactions that occur in day to day life. Automated classification is key thing that will benefit from this project.

10.FUTURE SCOPE

In this project we have used logistic regression algorithm for training our model to detect the transactions which are fraud and in future we can even build classifiers using some advanced algorithms like Decision Trees, Naïve Bayes and we can improve our accuracy in classification. We can even use Artificial Neural Networks for even better results.

11.BIBILOGRAPHY

- https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc
- https://towardsdatascience.com/an-introduction-to-logistic-regression-8136ad65da2e
- https://medium.com/analytics-vidhya/calculating-accuracy-of-an-ml-model-8ae7894802e
- https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html

APPENDIX

```
package org.ml;
import java.util.Arrays;
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) {
                              e.printStackTrace();
                            }
                return dataset;

            }

   public static void main(String[] args) throws Exception{

      Instances train_data = getInstances("credit_train.arff");
```

```
        Instances test_data = getInstances("credit_test.arff");

       System.out.println("Number of instances in Training set--"+train_data.size());
       System.out.println("Number of instances in Testing set--"+test_data.size());

       Classifier classifier = new weka.classifiers.functions.Logistic();
       classifier.buildClassifier(train_data);

       Evaluation eval = new Evaluation(train_data);
        eval.evaluateModel(classifier, test_data);
        System.out.println();
       System.out.println("* Evaluation of Model with Testing dataset *");
       System.out.println();
       System.out.println(eval.toSummaryString());
       System.out.println("---------------------------------------------------------------------------------");
        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.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("------------------");

    }

}
```