INTRODUCTION

1.1 Overview

Loans are the core business of banks. The main profit comes directly from the loan's interest. The loan companies grant a loan after an intensive process of verification and validation. However, they still don't have assurance if the applicant is able to repay the loan with no difficulties

1.2 Purpose

The main aim of this use-case is to build a predictive model to predict if an applicant is able to repay the lending company or not.

LITERATURE SURVEY

2.1 Existing problem

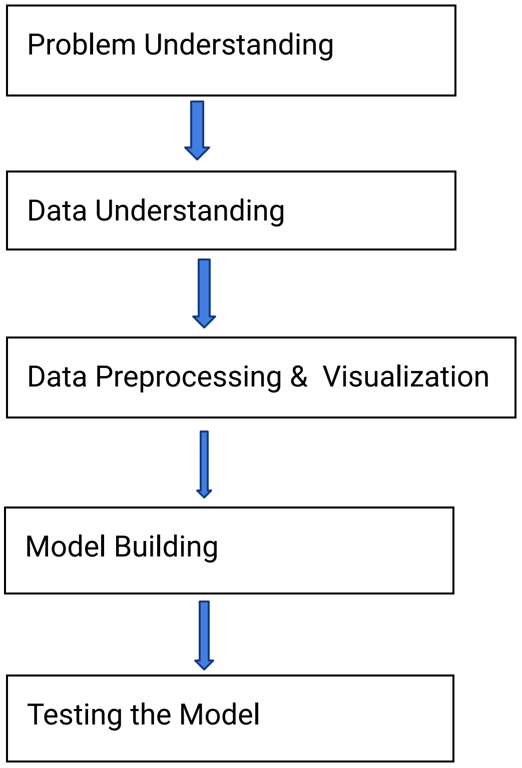
The Existing approach to solve this problem is to employ a person and do the background checks needed to make sure that the customer is eigible for the loanbased on customer detail provided while filling online application form.

2.2 Proposed Solution

It's a classification problem , given information about the application we have to predict whether the they'll be to pay the loan or not. Details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers.

THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hadware / Software designing

Software Required:

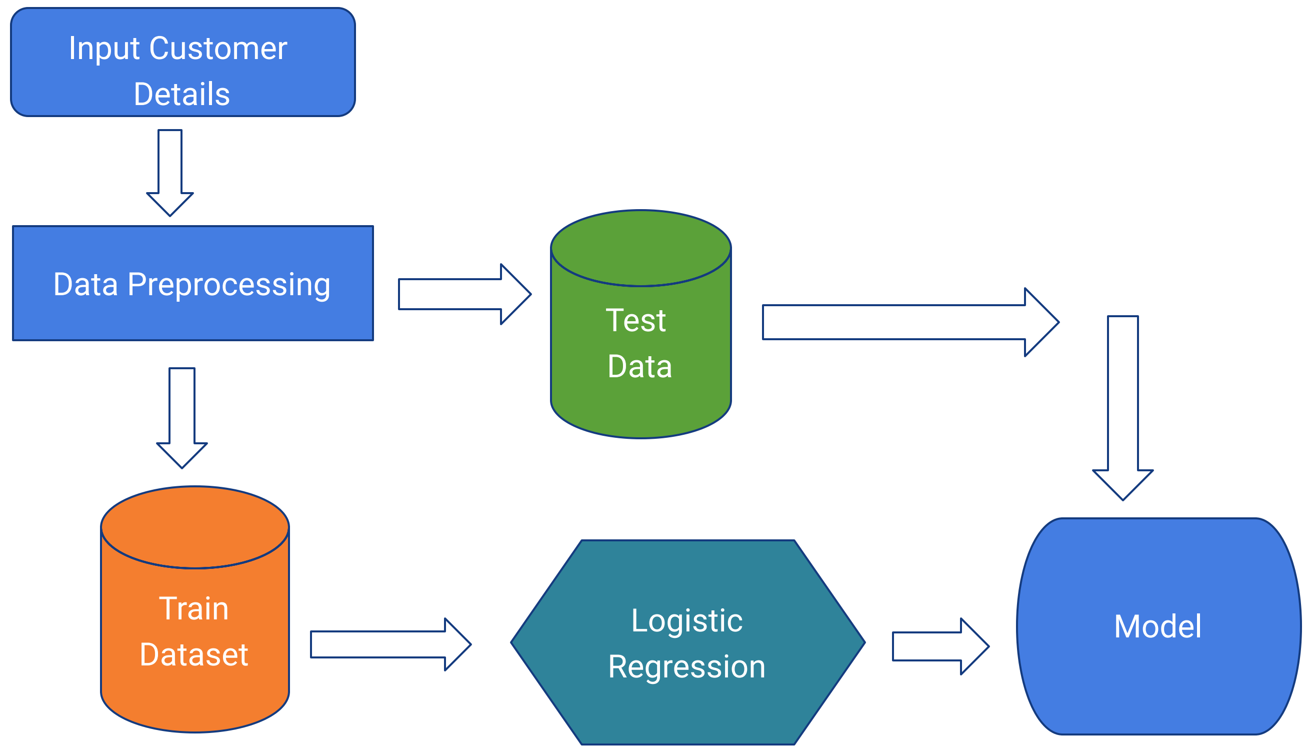
1) Weka Software
2) Eclipse IDE
3) Install tablesaw and weka packages

EXPERIMENTAL INVESTIGATIONS

We have 12 independent variables and 1 target variable, i.e. Loan_Status in the training dataset.
We can see there are two formats of data types
i.e, Nominal and Numerical. The loan of 422(around 69%) people out of 614 were approved.

- It seems people with a credit history as 1 are more likely to get their loans approved.
- The proportion of loans getting approved in the semi-urban area is higher as compared to that in rural or urban areas.
- Applicant's income does not affect the chances of loan approval
- It shows that if co-applicants income is less the chances of loan approval are high. But this does not look right. The possible reason behind this may be that most of the applicants don't have any co-applicant so the co-applicant income for such applicants is 0 and hence the loan approval is not dependent on it.
- We can see that Proportion of loans getting approved for applicants having low Total_Income is very less compared to that of applicants with Average, High & Very High Income.
- It can be seen that the proportion of approved loans is higher for Low and Average Loan Amount as compared to that of High Loan Amount
- We see that the most correlate variables are (ApplicantIncome — LoanAmount) and (Credit_History — Loan_Status). LoanAmount is also correlated with CoapplicantIncome.

FLOWCHAT



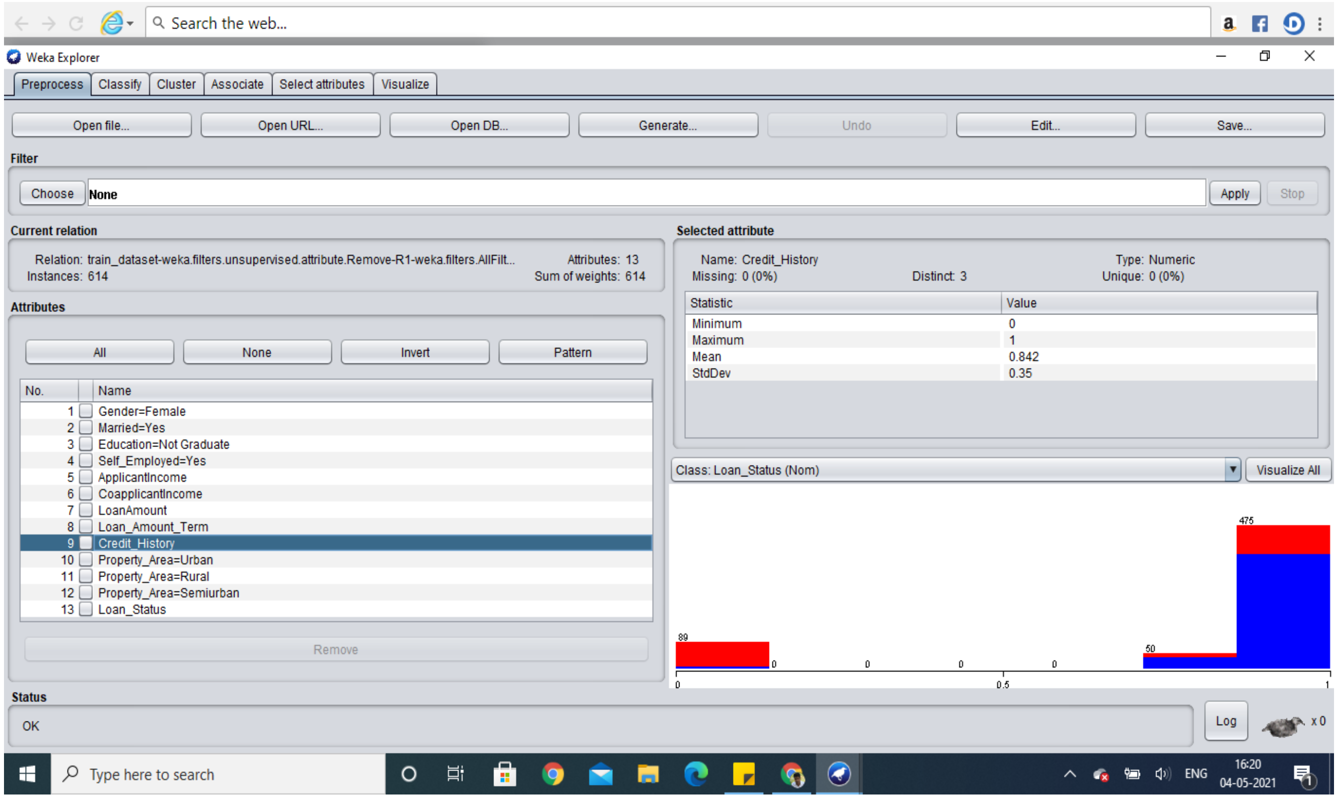
RESULT

The Accuracy of the Logistic Regression model which we have build is 84%
** Logistic Regression Evaluation with Datasets **

```
Correctly Classified Instances        182             84.2593 %
Incorrectly Classified Instances       34             15.7407 %
Kappa statistic                    0.6034
Mean absolute error                    0.3106
Root mean squared error                   0.3824
Relative absolute error            71.2395 %
Root relative squared error           81.3534 %
Total Number of Instances             216
```

Confusion matrix:

```
[142.0, 3.0]
[31.0, 40.0]
-------------------
Area under the curve
0.7565808644973288
-------------------
[Correct, Incorrect, Kappa, Total cost, Average cost, KB relative, KB information, Correlation, Complexity 0, Complexity scheme, Complexity improvement, MAE, RMSE, RAE, RRSE, Coverage, Region size, TP rate, FP rate, Precision, Recall, F-measure, MCC, ROC area, PRC area]
Recall :0.56
Precision:0.93
F1 score:0.7
Accuracy:0.84
```



Filter: None

Current relation

Relation: train_dataset-weka.filters.unsupervised.attribute.Remove-R1-weka.filters.AllFilt... Attributes: 13
Instances: 614 Sum of weights: 614

Attributes

No.	Name
1	Gender=Female
2	Married=Yes
3	Education=Not Graduate
4	Self_Employed=Yes
5	ApplicantIncome
6	CoapplicantIncome
7	LoanAmount
8	Loan_Amount_Term
9	Credit_History
10	Property_Area=Urban
11	Property_Area=Rural
12	Property_Area=Semiurban
13	Loan_Status

Selected attribute

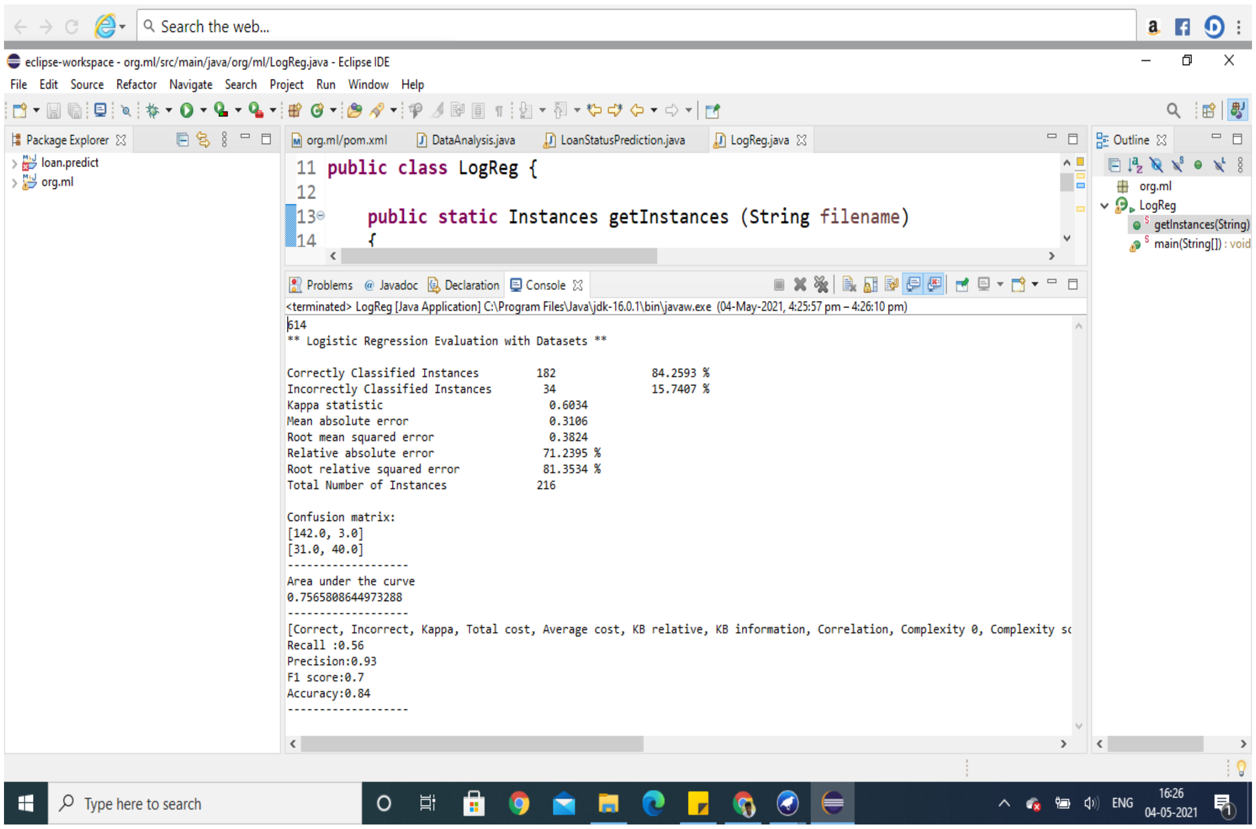
Name: Credit_History Type: Numeric
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.842
StdDev	0.35

Class: Loan_Status (Nom)

Status

OK



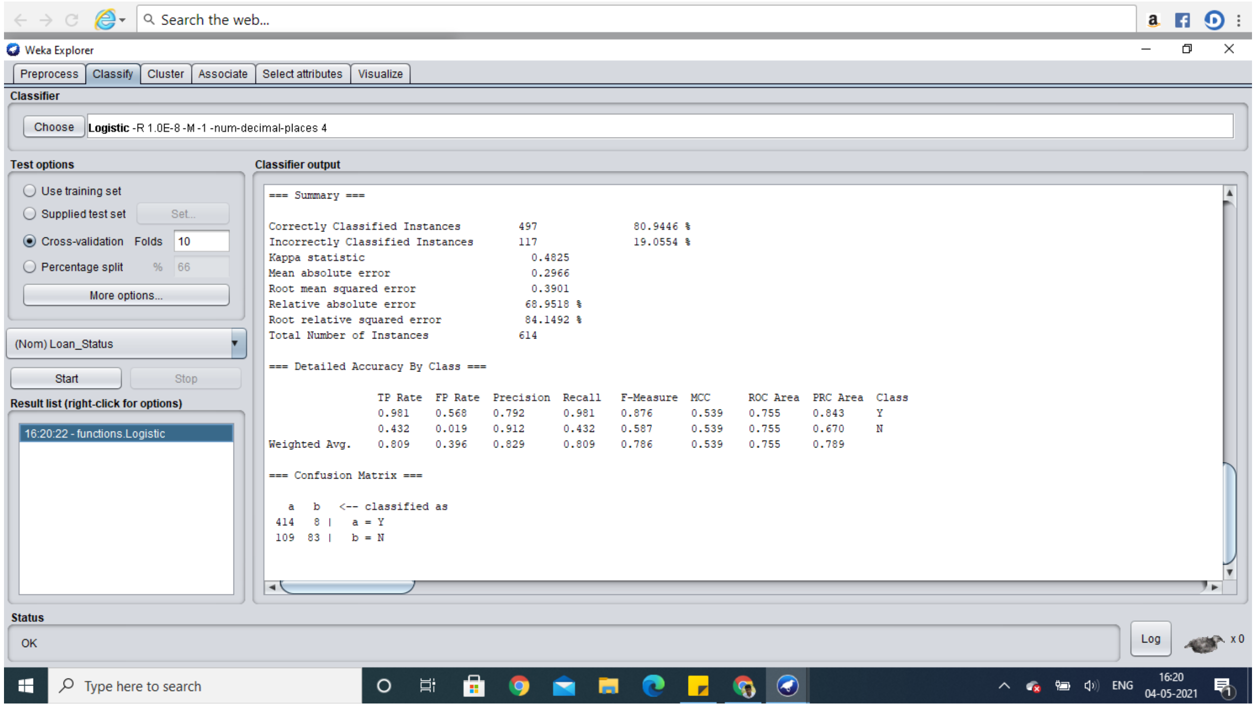
```java
public class LogReg {

    public static Instances getInstances (String filename)
    {
```

```
<terminated> LogReg [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (04-May-2021, 4:25:57 pm – 4:26:10 pm)
614
** Logistic Regression Evaluation with Datasets **

Correctly Classified Instances        182               84.2593 %
Incorrectly Classified Instances       34               15.7407 %
Kappa statistic                         0.6034
Mean absolute error                     0.3106
Root mean squared error                 0.3824
Relative absolute error                71.2395 %
Root relative squared error            81.3534 %
Total Number of Instances             216

Confusion matrix:
[142.0, 3.0]
[31.0, 40.0]
------------------
Area under the curve
0.7565808644973288
------------------
[Correct, Incorrect, Kappa, Total cost, Average cost, KB relative, KB information, Correlation, Complexity 0, Complexity sc
Recall :0.56
Precision:0.93
F1 score:0.7
Accuracy:0.84
------------------
```



Classifier

Logistic -R 1.0E-8 -M -1 -num-decimal-places 4

Test options

( ) Use training set
( ) Supplied test set
(•) Cross-validation Folds 10
( ) Percentage split % 66

(Nom) Loan_Status

Result list (right-click for options)

16:20:22 - functions.Logistic

Classifier output

```
=== Summary ===

Correctly Classified Instances         497               80.9446 %
Incorrectly Classified Instances       117               19.0554 %
Kappa statistic                          0.4825
Mean absolute error                      0.2966
Root mean squared error                  0.3901
Relative absolute error                 68.9518 %
Root relative squared error             84.1492 %
Total Number of Instances              614

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.981    0.568    0.792      0.981    0.876      0.539    0.755     0.843     Y
                 0.432    0.019    0.912      0.432    0.587      0.539    0.755     0.670     N
Weighted Avg.    0.809    0.396    0.829      0.809    0.786      0.539    0.755     0.789

=== Confusion Matrix ===

   a   b   <-- classified as
 414   8 |   a = Y
 109  83 |   b = N
```

Status

OK

ADVANTAGES AND DISADVANTAGES

Advantages:

- The Proposed Model has automated the Loan Eligibility Process of the customer.
- The Model showed 84% percent accuracy.
- Logistics Regression proved to be the best algorithm to build the model, since all other classification algorithms showed only 78-80% accuracy.
- A loan company can use this prediction model without any concerns.

Disadvantages:

- The loan companies grant a loan after an intensive process of verification and validation. However, they still don't have assurance if the applicant is able to repay the loan with no difficulties.

APPLICATIONS

The main application of the model are in Banking sector and other loan lending companies. Loans are the core business of banks. The main profit comes directly from the loan's interest.

CONCLUSION

The conclusions that are can be derived from this project are:
1) The Project is more usefull in predicting the loan eligibility status of a customer.
2)The Accuracy is 84%.
3) Banking sector and other loan lending companies can be benified by this prediction model.
4) With just limited information we can do great predictions.

FUTURE SCOPE

In Future we can improve the accuracy of the model by adding some more attibutes like previous loans status, years of more service etc..

BIBILOGRAPHY

References:

https://www.analyticsvidhya.com/blog/2015/11/easy-methods-deal-categorical-variables-predi
ctive-modeling/

https://towardsdatascience.com/predict-loan-eligibility-using-machine-learning-models-7a14ef9
04057

Kaggle for datasets.

https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148

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 LogReg {

	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;
	}

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

		Instances train_data =
getInstances("C:\\\\Users\\\\reddy\\\\eclipse-workspace\\\\org.ml\\\\src\\\\main\\\\java\\\\o
rg\\\\ml\\\\train_Weka_dataset.arff");
		Instances test_data =
getInstances("C:\\\\Users\\\\reddy\\\\eclipse-workspace\\\\org.ml\\\\src\\\\main\\\\java\\\\o
rg\\\\ml\\\\test_sample.arff");
		System.out.println(train_data.size());

		/** Classifier here is Linear Regression */
		Classifier classifier = new weka.classifiers.functions.Logistic();
		/** */
		classifier.buildClassifier(train_data);


		/**
		 * train the alogorithm 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());
//		System.out.print(" the expression for the input data as per alogorithm 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(2);
//                  double value = classifier.classifyInstance(predicationDataSet);
//                  /** Prediction Output */
//                  System.out.println("Predicted label:");
//                  System.out.print(value);



          }

}
```