CUSTOMER BEHAVIOR PREDICTION IN BANK MARKETING

Teamwork

رتاج حسين الحازمي - 445006594

غلا شاهر البشري - 445006326

هادن حسان الشعار - 445013952

خلود حامد المسعودي - 445002504

Group: 6

Report Highlights

Introduction

In this project, we wanted to see if machine learning can help a bank know which customers are more likely to say “yes” to a marketing offer. We used the Bank Marketing Dataset, which contains over 45,000 customer records with details like age, job, contact method, and whether they accepted the offer or not.
We used the SMO algorithm, which is a good way to train Support Vector Machines (SVM) in Weka.

The Goal

Our goal was to build a predictive model that helps the bank understand customer behavior. If the model works well, the bank can focus on customers who are more likely to accept the offer, saving both time and resources.

The Algorithm Used

We used SMO, which stands for Sequential Minimal Optimization. It works by finding the best boundary (hyperplane) that separates two classes — in our case, customers who said "yes" and those who said "no". We chose this method because it is powerful in handling binary classification problems like this one. However, we tested more than one algorithm and it was the best in results.

Result list (right-click for options)

05:23:21 - bayes.NaiveBayes
05:32:05 - bayes.NaiveBayes
05:33:56 - trees.J48
05:57:22 - trees.J48
06:09:23 - bayes.NaiveBayes
06:10:34 - trees.J48
06:15:38 - functions.SMO
06:22:46 - bayes.NaiveBayes
06:29:42 - bayes.NaiveBayes
06:31:44 - bayes.NaiveBayes
06:34:59 - functions.SMO
07:07:39 - functions.SMO
07:08:34 - functions.SMO

Dataset Description

This dataset contains client information collected from a marketing campaign of a Portuguese banking institution. Each record represents a client and their response to a direct marketing call.

UCI Machine Learning Repository: Bank Marketing Dataset

- Number of Instances: 45,211
- Number of Attributes: 13 (after cleaning).

Steps we followed:

- Downloaded the CSV file.
- Removed unnecessary columns (like duration, contact, day, and default).
- Saved it in Excel, then converted it to .arff format using Weka.
- Used Normalization for the data.

Dataset Description

This dataset contains client information collected from a marketing campaign of a Portuguese banking institution. Each record represents a client and their response to a direct marketing call.

UCI Machine Learning Repository: Bank Marketing Dataset

Variable Name	Typs	Description
age	Integer	
job	Categorical	type of job (‘employed’,’services’,’student’,’technician’,’unemployed’,’unknown’)
marital	Categorical	marital status (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
education	Categorical	Education Level
default	Binary	has credit in default?
balance	Integer	average yearly balance
housing	Binary	has housing loan?
loan	Binary	has personal loan?
contact	Categorical	contact communication type (categorical: 'cellular','telephone')

Dataset Description

This dataset contains client information collected from a marketing campaign of a Portuguese banking institution. Each record represents a client and their response to a direct marketing call.

UCI Machine Learning Repository: Bank Marketing Dataset

Variable Name	Typs	Description
day_of_week	Date	last contact day of the week
month	Date	last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
duration	Integer	last contact duration, in seconds (numeric).
campaign	Integer	number of contacts performed during this campaign and for this client (numeric, includes last contact).
pdays	Integer	number of days that passed by after the client was last contacted from a previous campaign (numeric; -1 means client was not previously contacted).
Previous	Integer	number of contacts performed before this campaign and for this client
poutcome	Categorical	outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')
y	Binary	has the client subscribed a term deposit?

Results and Analysis

We tested the SMO model using two different validation methods:
10-fold cross-validation and 70/30 percentage split.

a. 10-Fold Cross-Validation

- Accuracy: 89.29%
- Correct predictions: 40,367
- Incorrect predictions: 4,844
- Most of the correct predictions were for customers who said "No".
- The model struggled with predicting "Yes" customers.

confusion matrix:

- 39,389 “No” predictions were correct.
- 978 “Yes” predictions were correct.
- Many "Yes" cases were misclassified as "No".

```
Time taken to build model: 215.58 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       40367               89.2858 %
Incorrectly Classified Instances      4844               10.7142 %
Kappa statistic                          0.2486
Mean absolute error                      0.1071
Root mean squared error                  0.3273
Relative absolute error                 51.8564 %
Root relative squared error            101.843  %
Total Number of Instances            45211

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.987    0.815    0.901      0.987    0.942      0.307    0.586     0.901     no
               0.185    0.013    0.647      0.185    0.288      0.307    0.586     0.215     yes
Weighted Avg.  0.893    0.721    0.872      0.893    0.866      0.307    0.586     0.821

=== Confusion Matrix ===

     a     b   <-- classified as
 39389   533 |     a = no
  4311   978 |     b = yes
```

Results and Analysis

b. 70/30 Split Validation

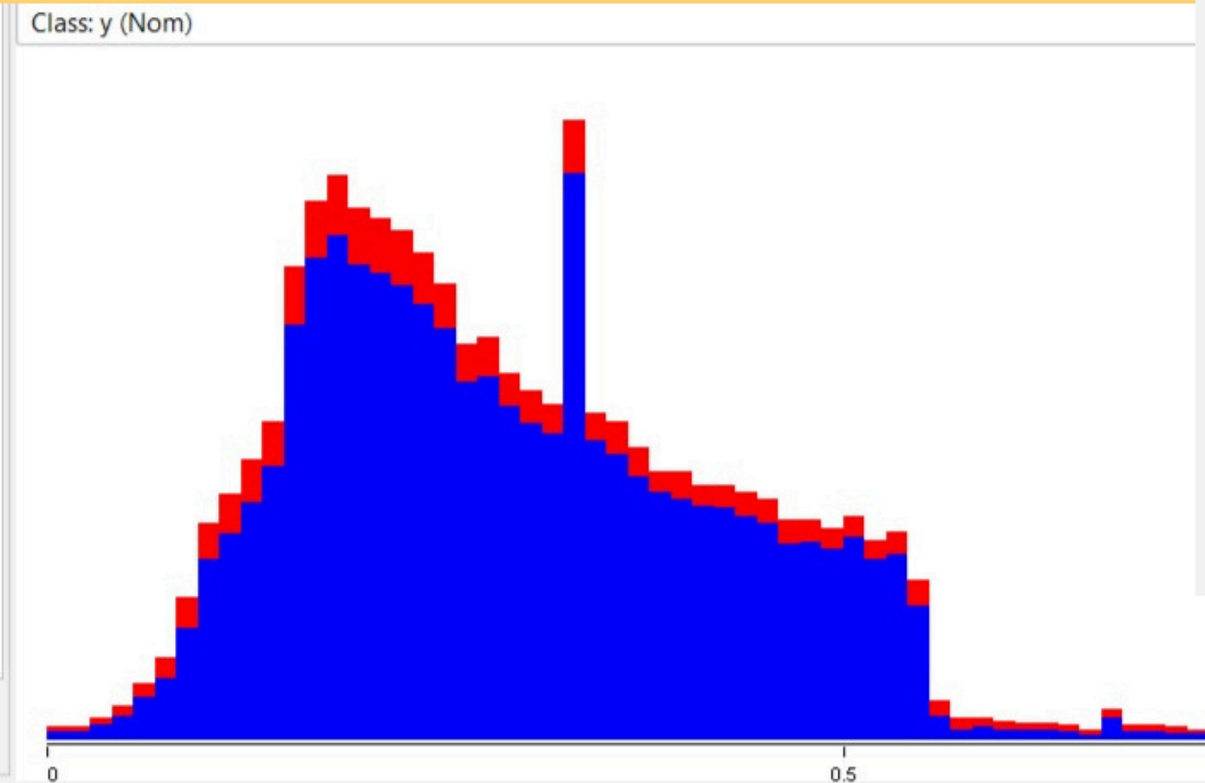
- Accuracy: 88.93%
- Correct predictions: 12,062
- Incorrect predictions: 1,501
- The result was similar to the first method — better at predicting "No".

Confusion matrix:

- 11,770 "No" correctly predicted.
- 292 "Yes" correctly predicted.
- The model still had difficulty with the minority class ("Yes").

```
Time taken to build model: 215.63 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.02 seconds

=== Summary ===

Correctly Classified Instances       12062               88.9331 %
Incorrectly Classified Instances      1501               11.0669 %
Kappa statistic                          0.2396
Mean absolute error                      0.1107
Root mean squared error                  0.3327
Relative absolute error                 53.2834 %
Root relative squared error            102.572  %
Total Number of Instances            13563

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.986    0.820    0.899      0.986    0.940      0.295    0.583     0.898     no
                 0.180    0.014    0.628      0.180    0.280      0.295    0.583     0.211     yes
Weighted Avg.    0.889    0.724    0.866      0.889    0.861      0.295    0.583     0.816

=== Confusion Matrix ===

     a     b   <-- classified as
 11770   173 |     a = no
  1328   292 |     b = yes
```

Results and Analysis



c. Visualization

We also used Weka's histogram tool. The chart showed that the model was more confident when predicting "No" (blue bars), while "Yes" predictions (red bars) were fewer and less confident. This is a common issue with imbalanced datasets.

Challenges Faced

We noticed that SMO took a long time to run, especially with cross-validation, because the dataset was large (around 45,000 records). Also, the data was imbalanced, so the model often predicted "No" instead of "Yes", which affected the results.

And to solve that we tried to use Resample technique:
(Trying to Improve the Model Using Resample).
Because the dataset was imbalanced (most customers said “no” and only a few said “yes”), we used the Resample filter in Weka with a bias of 1 to make the class distribution more balanced. After that, we ran the SMO algorithm again using both cross-validation and percentage split methods.
The results were interesting. The model got much better at finding the “yes” cases. The recall for the “yes” class increased from 0.180 to 0.489, and the F-measure improved from 0.280 to 0.592. This means the model became better at identifying customers who actually accepted the offer.
However, the overall accuracy dropped from 89% to 66%, which is normal when you rebalance the data. Even though the accuracy became lower, this step showed that we understood the issue of class imbalance and tried to fix it using preprocessing. It also helped us see how resampling can make a model better at predicting the important but less frequent class.

```
Time taken to build model: 1158.89 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       29995               66.3459 %
Incorrectly Classified Instances     15215               33.6541 %
Kappa statistic                          0.3269
Mean absolute error                      0.3365
Root mean squared error                  0.5801
Relative absolute error                 67.3081 %
Root relative squared error            116.0242 %
Total Number of Instances            45210

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.838     0.511     0.621       0.838   0.713      0.349    0.663     0.602     no
               0.489     0.162     0.751       0.489   0.593      0.349    0.663     0.623     yes
Weighted Avg.  0.663     0.337     0.686       0.663   0.653      0.349    0.663     0.612

=== Confusion Matrix ===

     a     b   <-- classified as
 18932  3673 |     a = no
 11542 11063 |     b = yes
```

Future Work

- Explore more lightweight algorithms like Logistic Regression for large datasets.
- Try different resampling and balancing techniques such as SMOTE.
- Implement feature selection to reduce dimensionality and improve model efficiency.
- Consider using ensemble methods for higher accuracy.

Conclusion

This project showed how Support Vector Machines (SMO in Weka) can be used to classify customer responses to marketing offers. The model performed well overall, achieving about 89% accuracy, but it had challenges with predicting "Yes" responses due to the class imbalance.

REFERENCES

1. UCI Machine Learning Repository: Bank Marketing Dataset
2. Weka Documentation: https://www.cs.waikato.ac.nz/ml/weka/
3. Ega Febridh. Guide for Resampling an Imbalance Dataset. Medium. https://medium.com/@egafebridh/guide-for-resampling-an-imbalance-dataset-8ba2ec288fbe
4. 7 Techniques to Handle Imbalanced Data. KDnuggets. https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html
5. University Lectures in Principles and Techniques of Artificial Intelligence — Dr. Emtethal M. Alafghani