Credit Card Defaulters: Predictive Models

Kasela Pranav[1], Pagani Miriam Beatrice[2], Savino Marco[3], Zaccaria Antonella[4]

Abstract
Credit card default happens when you've become severely delinquent on your credit card payment.
It's a serious credit card status that not only affects your standing with that credit card issuer, but also your credit standing in general and your ability to get approved for credit cards, loans, and other credit-based services.
When you accept a credit card, you agree to certain terms, e.g. you agree to make your minimum payment by the due date listed on your credit card statement.
If you miss the minimum credit card payment six months in a row, your credit card will be in default; your credit card issuer will likely close your account and report the default to the credit bureaus.
By the time your credit card defaults, you've likely accumulated hundreds of dollars in fees and interest charges.
Unfortunately, your options for clearing up the credit card default may be limited because of the number of payments you've missed on your account.
For this reason, assuming truthful the given data, we show the procedure used to create a prevision algorithm aiming to foresee the default payment's client.

Keywords
Credit Card — Defaulting — Prevision

[1] *Matricola: 846965, Department of Informatics, University of Bicocca, CdL: Data Science*
[2] *Matricola: 794274, Department of Informatics, University of Bicocca, CdL: Data Science*
[3] *Matricola: 793516, Department of Informatics, University of Bicocca, CdL: Data Science*
[4] *Matricola: 848647, Department of Informatics, University of Bicocca, CdL: Data Science*

Contents

Introduction 1
1 Preprocessing 2
2 Classification 2
3 Feature Selection 3
3.1 Evaluation 4
Wrapper NaiveBayes • Wrapper BayesNet
4 Alternative Approaches 4
4.1 Cost Analysis 4
4.2 SMOTE Technique 5
5 Conclusions 5
References 5

Introduction[1]

This dataset is available on Kaggle under the name *Default of Credit Card Clients Dataset* and it contains information on default payments, demographic factors, credit data, history of payment and bill statements of 30,000 credit card clients in Taiwan from April 2005 to September 2005.

The 25 attributes and their characteristics are:

- ID: ID of each client
- LIMIT_BAL: Amount of given credit in NT dollars
- SEX: Gender (1=male, 2=female)
- EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 0,5,6=unknown)
- MARRIAGE: Marital status (1=married, 2=single, 3=others, 0=others)
- AGE: Age in years
- PAY_0: Repayment status in September 2005 (-2=no consumption, 0=use of revolving credit, -1=pay duly, 1=payment delay for one month, 2=payment delay for two months, ... 8=payment delay for eight months, 9=payment delay for nine months and above)
- PAY_2: Repayment status in August, 2005 (scale same as above)
- PAY_3: Repayment status in July, 2005 (scale same as above)
- PAY_4: Repayment status in June, 2005 (scale same as above)
- PAY_5: Repayment status in May, 2005 (scale same as above)
- PAY_6: Repayment status in April, 2005 (scale same as above)
- BILL_AMT1: Amount of bill statement in September 2005 respectively (NT dollar)
- BILL_AMT2: Amount of bill statement in August, 2005 (NT dollar)
- BILL_AMT3: Amount of bill statement in July, 2005 (NT dollar)
- BILL_AMT4: Amount of bill statement in June, 2005

(NT dollar)

- BILL_AMT5: Amount of bill statement in May, 2005 (NT dollar)
- BILL_AMT6: Amount of bill statement in April, 2005 (NT dollar)
- PAY_AMT1: Amount of previous payment in September 2005 respectively (NT dollar)
- PAY_AMT2: Amount of previous payment in August, 2005 (NT dollar)
- PAY_AMT3: Amount of previous payment in July, 2005 (NT dollar)
- PAY_AMT4: Amount of previous payment in June, 2005 (NT dollar)
- PAY_AMT5: Amount of previous payment in May, 2005 (NT dollar)
- PAY_AMT6: Amount of previous payment in April, 2005 (NT dollar)
- default.payment.next.month: Default payment (1=yes, 0=no)

Goal:
Looking at the problem we see a potential use in predicting month by month, the default of the clients.

1. Preprocessing

Before proceeding with Feature Engineering and the implementation of Machine Learning, we analyze the dataset with Descriptive Statistics in order to understand more the composition and tendencies of our data.

We notice that in the attribute *EDUCATION* the values 0,5 and 6 are unknown while the value 4 is other, so we decide to categorize 0,5 and 6 as other too.

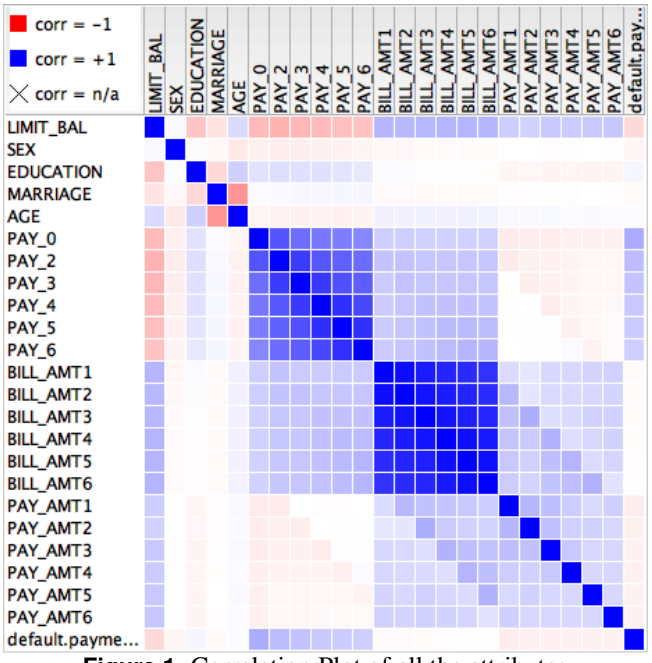

Figure 1. Correlation Plot of all the attributes

The *BILL_AMT1,...,BILL_AMT6* are heavily correlated (Figure 1); the best option is to create a new feature which is the mean of these columns, called *AVG_BILL_AMT*. We decide to compute the average because the bill indicates how much a person spends and it usually remains constant during the months. We remove the *BILL_AMT1,...,BILL_AMT6* attributes and keep only *AVG_BILL_AMT*.

For the *AGE* attribute we decide to discretize it, creating 4 groups: $\{[20,30],(30,40],(40,50],(50,+\infty)\}$. The choice to discretize is made to avoid the possible overfitting phenomena since the *AGE* attribute is imbalanced. The new attribute is called *AgeBin*.

In the *EDUCATION* attribute we combine *0=others* with *3=others* (it is considered to be divorced or separated). We combine *SEX* and *MARRIAGE* to reduce the dimension using the following python code, this way the we keep all the information but reduce the number of features:

```
#df is our dataframe given in the input node
df.loc[((df.SEX == 1) & (df.MARRIAGE == 1))
    , 'SEX_MARRIAGE'] = 1 #married man
df.loc[((df.SEX == 1) & (df.MARRIAGE == 2))
    , 'SEX_MARRIAGE'] = 2 #single man
df.loc[((df.SEX == 1) & (df.MARRIAGE == 3))
    , 'SEX_MARRIAGE'] = 3 #divorced man
df.loc[((df.SEX == 2) & (df.MARRIAGE == 1))
    , 'SEX_MARRIAGE'] = 4 #married woman
df.loc[((df.SEX == 2) & (df.MARRIAGE == 2))
    , 'SEX_MARRIAGE'] = 5 #single woman
df.loc[((df.SEX == 2) & (df.MARRIAGE == 3))
    , 'SEX_MARRIAGE'] = 6 #divorced woman
```

Due to the distribution of *LIMIT_BAL, PAY_AMT1,..., PAY_AMT6, AVG_BILL_AMT* we use the logarithmic transformation. Since there are negative values, we translate the minimum of the attributes to 1 and apply logarithm to the latter. We notice that the distribution of the *AVG_BILL_AMT* has one strong outlier so we remove it to avoid overfitting and standardize the latter so it is more sparse.

To start studying the classifiers we split our data in training set(67%) and test set(33%), using a stratified sampling on *default.payment.next.month*.

2. Classification

The models selected for the classifications are:

- MLP
- SPegasos
- NaiveBayes
- NBTree
- BayesNet
- Heuristic:
  - J48
  - RandomForest
  - DecisionTree
- Logistic

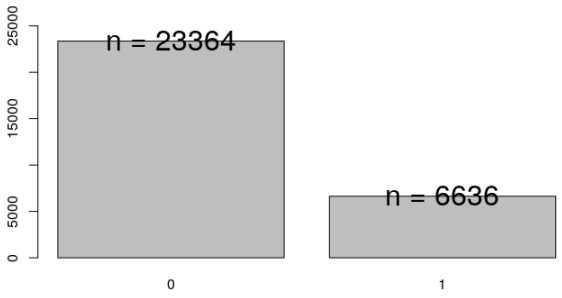

Figure 2. Target Class Distribution

The target class has 77.7% of not defaulters, 22.3% of defaulters (Figure 2), so we have a class imbalance problem and we cannot use accuracy as an evaluation measure for the classifiers. The optimal measure could be either Recall or Precision, depending on bank's preference. We decide to use the following measures: Recall, Precision, F_1-measure and AUC of the ROC Curve.

Classifier	Recall	Precision	F_1-measure	Accuracy	AUC
MLP	0.474	0.565	0.515	0.803	0.722
SPegasos	0.202	0.721	0.316	0.806	0.59
NaiveBayes	0.486	0.488	0.487	0.773	0.742
NBTree	0.481	0.53	0.504	0.791	0.754
BayesNet(TANB)	0.421	0.586	0.49	0.806	0.754
J48	0.325	0.631	0.429	0.809	0.629
RandomForest	0.325	0.612	0.425	0.805	0.723
DecisionTree	0.401	0.402	0.401	0.736	0.635
Logistic	0.221	0.685	0.335	0.805	0.745

Table 1. Evaluation measures using Holdout

In the Table 1 we see that using holdout, for the Recall measure NaiveBayes and NBTree are the optimal classifiers, but NaiveBayes has a low precision of 0.488. Using the F_1-measure or AUC the best classifiers are MLP, NBTree, BayesNet and NaiveBayes.

The Holdout method depends on the particular test set. We use the 3-Folds Cross Validation to have a less biased measure:

Classifier	Recall	Precision	F_1-measure	Accuracy	AUC
MLP	0.372	0.662	0.476	0.819	0.773
SPegasos	0.22	0.703	0.334	0.807	0.597
NaiveBayes	0.503	0.485	0.494	0.772	0.749
NBTree	0.456	0.561	0.501	0.8	0.748
BayesNet(TANB)	0.427	0.59	0.495	0.808	0.759
J48	0.351	0.63	0.451	0.811	0.651
RandomForest	0.326	0.608	0.424	0.804	0.729
DecisionTree	0.408	0.406	0.407	0.737	0.634
Logistic	0.241	0.673	0.355	0.806	0.748

Table 2. Evaluation measures using 3-Folds Cross Validation

In the Table 2 we see that using 3 Folds Cross Validation, for the Recall measure NaiveBayes is the optimal classifiers. Using the F_1-measure the best classifiers are NBTree, BayesNet and NaiveBayes. While for the AUC the best one is the MLP. In Figure 3 we plot the ROC curve of different classifiers using 3-Folds. We note that in the first of the two graphs in Figure 3 Logistic is (except for a small portion of the domain) on the top of the other 3 curves and at a certain point J48 falls even below the random line, while in the second graph MLP is at the top but the ROC curves of the Bayesian classifiers are not far from the MLP curve.

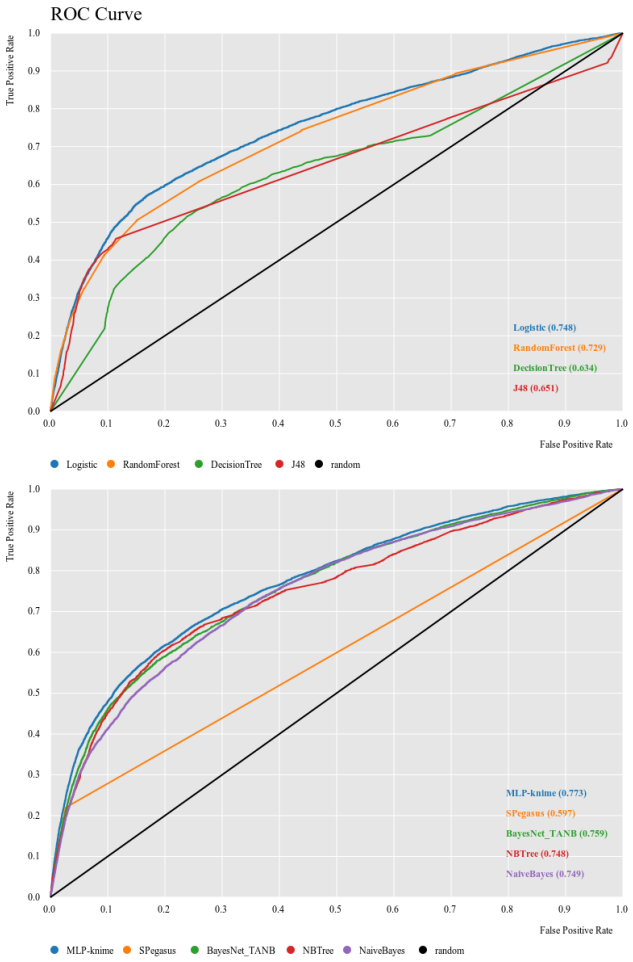

Figure 3. ROC Curves using 3-Folds CV

3. Feature Selection

For the feature selection we use 5 different filters:

- Gain Ratio feature evaluation
- Information Gain Ranking Filter
- Symmetrical Uncertainly Ranking Filter
- Relief Ranking Filter(k=10)
- Correlation Ranking Filter

But all these filters select all attributes. By using these filters we gain no additional information. The next approach is to use Wrapper, we decide to use as the classifiers: BayesNet

and NaiveBayes with 5 folds using as evaluation measure AUC. The wrapper with BayesNet selects 10 attributes:

- LIMIT_BAL
- EDUCATION
- PAY_0
- PAY_2
- PAY_3
- PAY_4
- PAY_6
- PAY_AMT1
- PAY_AMT4
- SEX_MARRIAGE

While the wrapper with NaiveBayes select the following 11:

- LIMIT_BAL
- EDUCATION
- PAY_0
- PAY_2
- PAY_3
- PAY_6
- PAY_AMT3
- PAY_AMT4
- AVG_BILL_AMT
- AgeBin
- SEX_MARRIAGE

3.1 Evaluation

In this phase we evaluate the same models that we used before feature selection in order to compare them.

3.1.1 Wrapper NaiveBayes

The Tables 3, 4 are obtained after applying the Feature Selection with NaiveBayes. The latter shows that AUC measure increases in Bayesian models and remains more or less unchanged for others. The F_1-measure gets worse slightly, but we cannot state anything about the model getting either worse or better without the confidence interval.

Classifier	Recall	Precision	F_1-measure	Accuracy	AUC
MLP	0.377	0.544	0.446	0.792	0.696
SPegasos	0.243	0.705	0.362	0.81	0.607
NaiveBayes	0.401	0.585	0.476	0.805	0.754
NBTree	0.441	0.58	0.501	0.806	0.76
BayesNet(TANB)	0.384	0.649	0.483	0.818	0.762
J48	0.349	0.662	0.457	0.817	0.667
RandomForest	0.34	0.578	0.428	0.799	0.715
DecisionTree	0.407	0.4	0.403	0.734	0.639
Logistic	0.261	0.676	0.377	0.809	0.738

Table 3. Evaluation using Holdout

Classifier	Recall	Precision	F_1-measure	Accuracy	AUC
MLP	0.362	0.672	0.471	0.82	0.768
SPegasos	0.214	0.711	0.328	0.807	0.595
NaiveBayes	0.417	0.584	0.487	0.805	0.757
NBTree	0.425	0.609	0.498	0.811	0.756
BayesNet(TANB)	0.395	0.643	0.49	0.818	0.764
J48	0.344	0.662	0.453	0.816	0.687
RandomForest	0.334	0.567	0.421	0.796	0.708
DecisionTree	0.382	0.406	0.394	0.74	0.636
Logistic	0.245	0.685	0.36	0.808	0.737

Table 4. Evaluation using 3-Folds Cross Validation

The benefit from the Feature Selection is that the number of attributes is reduced from 17 to 10, while it doesn't seem to have any considerable statistical difference between the initial model and the model obtained after the Feature Selection.

3.1.2 Wrapper BayesNet

In the Tables 5, 6 contain the value of the evaluation measures using Holdout and 3-folds Cross Validation respectively.

Classifier	Recall	Precision	F_1-measure	Accuracy	AUC
MLP	0.357	0.644	0.459	0.814	0.717
SPegasos	0.232	0.696	0.348	0.808	0.602
NaiveBayes	0.417	0.565	0.48	0.8	0.751
NBTree	0.446	0.574	0.502	0.804	0.762
BayesNet(TANB)	0.387	0.651	0.485	0.818	0.763
J48	0.346	0.661	0.454	0.816	0.687
RandomForest	0.331	0.574	0.419	0.798	0.712
DecisionTree	0.39	0.419	0.404	0.746	0.665
Logistic	0.209	0.663	0.317	0.802	0.737

Table 5. Evaluation using Holdout

Classifier	Recall	Precision	F_1-measure	Accuracy	AUC
MLP	0.364	0.674	0.472	0.82	0.768
SPegasos	0.234	0.696	0.35	0.808	0.603
NaiveBayes	0.431	0.562	0.488	0.8	0.753
NBTree	0.422	0.586	0.488	0.805	0.752
BayesNet(TANB)	0.395	0.641	0.489	0.817	0.764
J48	0.352	0.648	0.456	0.814	0.678
RandomForest	0.339	0.565	0.423	0.796	0.715
DecisionTree	0.4	0.423	0.411	0.747	0.651
Logistic	0.229	0.664	0.34	0.804	0.738

Table 6. Evaluation using 3 Fold Cross Validation

The same condition as NaiveBayes Wrapper persists here, as a matter of fact, the AUC measure increases and in the Bayesian classifiers and the F_1-measure doesn't seem to have any significant statistical difference.
The behavior of the two wrappers could have been expected to be similar since the attributes they choose are similar: except for *PAY_4* and *PAY_AMT1* the attributes of the BayesNet Wrapper are a subset of the attributes of NaiveBayes Wrapper.

4. Alternative Approaches

4.1 Cost Analysis

One of the alternative approach consists in a cost analysis, giving a more weight to either Precision or Recall.

We proceed with a Brute Force Approach, trying some different cost matrix to select the most suitable one.
We chose two matrices A(TN=0, FP=2, FN=5, TP=-1) and B(TN=-1, FP=2, FN=5, TP=0):

$$A = \begin{bmatrix} 0 & +2 \\ +5 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} -1 & +2 \\ +5 & 0 \end{bmatrix}$$

Using all the features, with matrix A we have Recall= 0.586, Precision = 0.454 and F_1-measure = 0.512, while with the matrix B we have Recall = 0.527, Precision = 0.49 and F_1-measure = 0.508. We have improved the Recall of 0.1 in the first case despite of the reduction of Precision.

After the Feature Selection with NaiveBayes, the Recall is reduced from 0.586 to 0.546, instead the Precision is increased from 0.454 to 0.502; overall the F_1-measure is increased from 0.512 to 0.523 with the matrix A. With the matrix B we have the same behavior (Precision = 0.552, Recall = 0.484 and F_1-measure = 0.515).

With the BayesNet we have: matrix A - Recall = 0.537, Precision = 0.508 and F_1-measure = 0.522. Matrix B - Recall = 0.477, Precision = 0.548 and F_1-measure = 0.51. The behavior is similar to the NaiveBayes Wrapper as expected.

4.2 SMOTE Technique

Another approach we used is to reduce the class imbalance oversampling the rare class in the training set using the *SMOTE* technique.
We use SMOTE on the training set filtered after the NaiveBayes Wrapper using the number of neighbors = 5. To evaluate the performance we use two models: Logistic and Gradient Boosted Trees. The trees and Logistic models are the one who usually benefit more using the SMOTE technique. We use Holdout on the same test used for evaluating the feature selection measures.

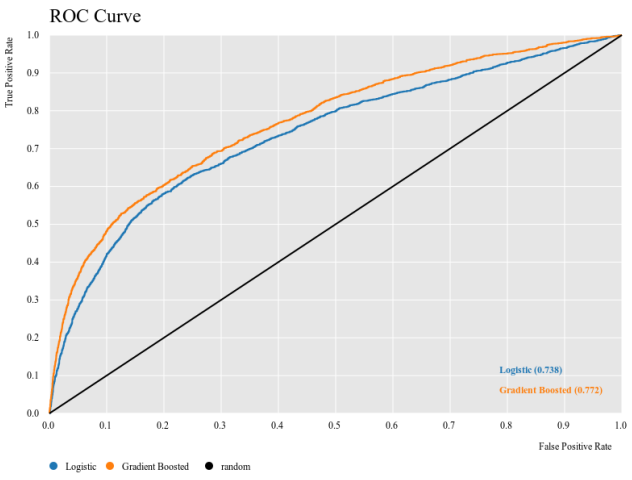

Figure 4. ROC Curves using SMOTE

In the case of Logistic we can see a drastic improvement in Recall from 0.261 to 0.605 and F_1-measure from 0.377 to 0.505. The Precision drops down from 0.676 to 0.433.
For the Gradient Boosted Trees we have more balanced values: Recall: 0.511, Precision: 0.555, F_1-measure: 0.532. In the Figure 4 we can see that using ROC Curve, the Gradient Boosted Trees is a better classifier than Logistic.

We decided to choose the SMOTE instead of a normal oversampling (taking the same record multiple times) because it is less prone to the overfitting phenomena.
To use SMOTE with cross validation we need to apply SMOTE on the training set and not on the combination of training and validation set otherwise we won't have the right values of the various measures. Using SMOTE with 3-Folds Cross Validation we confirm the values obtained with Holdout.

5. Conclusions

These results are certainly not god enough so that the models developed during this project can be directly used to predict a defaulter. We analyzed and created/modified the features without any domain knowledge. It might not be the best way of approaching the problem, since it could have brought us in the wrong direction and thus getting poor results.
We have shown different approached to improve the models, one of these approaches was using the cost matrix. The bank could use a domain expert to decide the costs of TP, TN, FP and FN, having the right cost matrix might improve significantly the models and we might even understand if the bank is interested in improving the Precision or Recall and proceed accordingly.
A practical use of the models just created could be to use the models with more than 50% Recall and a decent value of Precision (more than 40-45%). Let's take as an example the Gradient Boosted Trees with SMOTE: We know that the models classifies 50% of the defaulters with 50% of precision, the bank can now concentrate on the defaulters classified by the model and use their domain experts to find the defaulters from it. The bank does this usually with all clients now they can concentrate and thus invest less resources and still be able to obtain approximately 50% of the defaulters. And with the cost matrix we might even increase the percentage of the defaulters found with the models.
Having more data would have been useful to have more information on the defaulter's behavior or even better: to find a pattern in the data (maybe using a clustering technique). Even having more attributes could be useful to find the pattern we talked about, some relevant attributes might be, in our opinion (with low or no domain knowledge): number of children, total family income, number of other accounts with credit card (if possible, even though two different banks might not cooperate to give away such information) and many more.

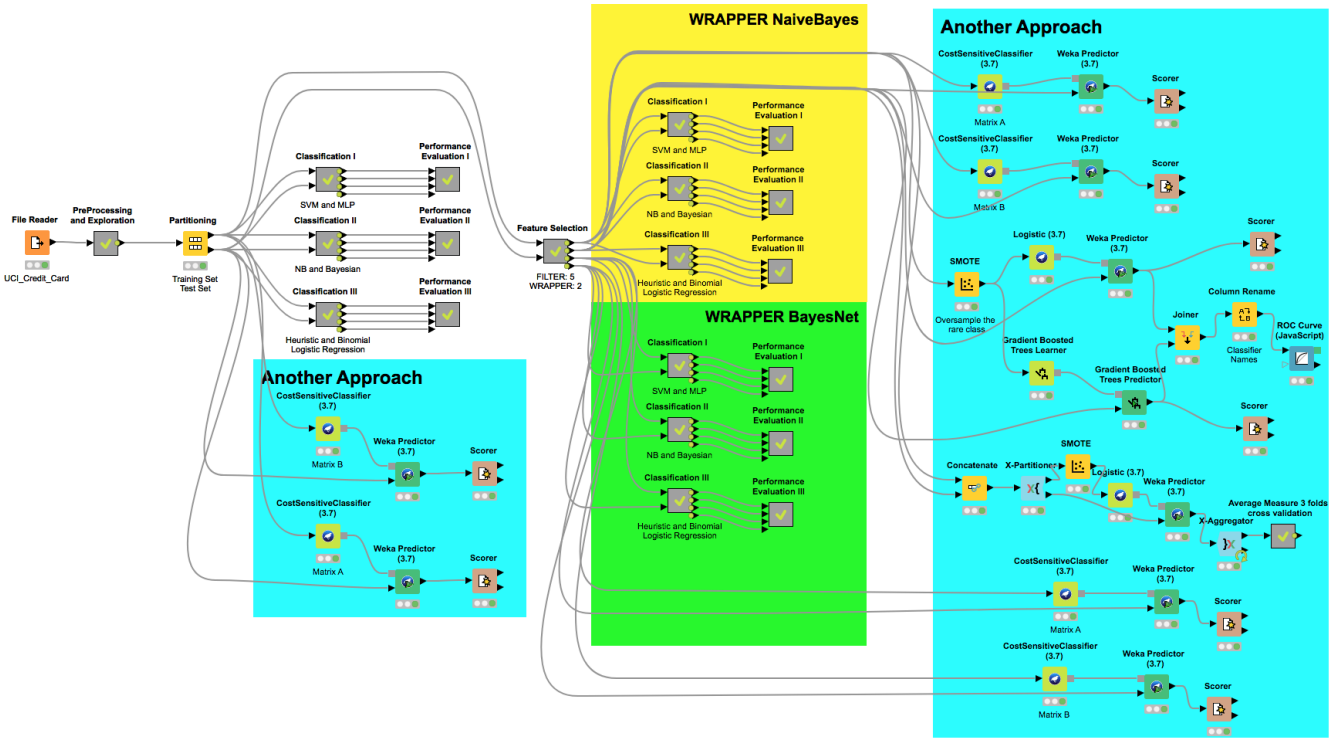

Figure 5. Knime Workflow Overview

References

[1] UCI I-Cheng Yeh. https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset.

[2] Luca Basanisi. https://www.kaggle.com/lucabasa/credit-card-default-a-very-pedagogical-notebook.

[3] Latoya Irby. https://www.thebalance.com/what-is-credit-card-default-960209.

[4] ItalPress. https://www.italpress.com/lifestyle/mancato-pagamento-carta-credito/.

[5] Marco Altini. https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation.