Predicting German Credit Default using logistic regression

Jerzy Grunwald

February 9, 2025

Contents

Introduction

We will focus on the task of credit scoring and related classification. Based on data regarding German bank clients, we will examine important features that affect creditability. We will also analyse methods of assessing classifier performance and optimal threshold selection.

Descriptive analysis

`German Credit Data` is a dataset containing 1000 rows describing bank clients. Their characteristics are described using 20 features, based on which we will try to predict binary variable `creditability`. Said variable informs us, whether the client is good, i.e. pays their rates on time, or bad. Below we provide description of every feature. DM stands for Deutsche Mark, which was the currency in Germany at the time of data collection.

Each feature will be described in a following manner: `feature name` (type [factor, numeric, character]) – description.

- `status.of.existing.checking.account` (factor, 4 levels) – money ammount in clients checking account,
- `duration.in.month` (numeric) – credit duration in months,
- `credit.history` (factor, 5 levels) – clients credit history,
- `purpose` (character) – what is the loan needed for,
- `credit.amount` (numeric) – loan ammount,
- `savings.accout.and.bonds` (factor, 5 levels) – money on clients savings account and/or in bonds,
- `present.employment.since` (factor, 5 levels) – how long has the client worked at their current job,
- `installment.rate.in.percentage.of.disposable.income` (numeric) – loan installment as a percentage of clients disposable income,
- `personal.status.and.sex` (factor, 5 levels) – gender and matrimonial status,
- `other.debtors.or.guarantors` (factor, 3 levels) – is the client the only debtor, are there others, or is the client a guarantor,
- `present.residence.since` (numeric) – years since the client has moved to their present residence,
- `property` (factor, 4 levels) – clients property (house, car, etc.),
- `age.in.years` (numeric) – clients age in years,
- `other.installment.plans` (factor, 3 levels) – other creditors,
- `housing` (factor, 3 levels) – does the client rent, own or live for free in their current residence,
- `number.of.existing.credits.at.this.bank` (numeric) – number of clients loans at this bank,
- `job` (factor, 4 levels) – is the client employed and how qualified is he,
- `number.of.people.being.liable.to.provide.maintenance.for` (numeric) – number of people client has to provide for,
- `telephone` (factor, 2 levels) – does the client have a telephone number,
- `foreign.worker` (factor, 2 levels) – is the client a foreign worker.

The response is `creditability`. it is a binary variable with levels `good` – in case the client pays their rates on time and `bad` – in case they don't.

There is no missing data. Variable `purpose` is of type character, so to enable proper analysis, we change its type to factor with 11 levels. In case of `personal.status.and.sex`, we notice unussed level `female: single`, thus we delete it.

Data split

Using `initial_split()` from `rsample` package, we split the dataset into training (80% of records) and test (20% of records) datasets. The training dataset will be used to build and train prediction models, which will be later evaluated using the test dataset.

Logistic regression model selection

We will fit logistic regression models. they are parametric models of the form

$$\log \frac{p(\mathbf{x})}{1-p(\mathbf{x})} = \beta^T \mathbf{x}, \tag{1}$$

where $p(\mathbf{x})$ is the probability that client of characteristic $\mathbf{x}$ is a good client, β is a vector of parameters.

Classification rule, or in short a classifier, we will call a function

$$d(\mathbf{x}) = \begin{cases} 1, & \text{when } p(\mathbf{x}) > c \\ 0, & \text{when } p(\mathbf{x}) \leq c \end{cases} \tag{2}$$

where c is a fixed number (often called a threshold) from the interval $[0,1]$. We begin by taking $c = \frac{1}{2}$.

From model given by the equation (1), we can extract the formula for the probability of success:

$$p(\mathbf{x}) = \frac{1}{1+\exp\left(-\beta^T \mathbf{x}\right)}.$$

Due to the fact, that the formula on the right takes values only from the interval $[0,1]$, $p(\mathbf{x})$ is a reasonable estimator for the probability of success. Logistic regression is not the only model with that property (there are also regression models such as probit, log-log, complementary log-log), however it is a popular and easily interpretable model, performing well with classification tasks. We will now focus on criteria allowing us to identify features significantly impacting $p(\mathbf{x})$.

Feature selection based on information criteria

Careful feature selection allows us to create better, more accurate and easier to interpret classifier. By focusing only on important features, model will adjust parameters better and lower the risk of overfitting.

The selection will be conducted using the `step()` function from `stats` package. We will use three step methods, that will create a model minimising given information criteria. By taking $k = 2$ in `step()`, model will be chosen by minimizing AIC, while with $k = \ln(n)$, where n is the number of rows in our dataset, we obtain a model minimizing BIC.

- Forward method
    1. Using the function `glm()` we create a logistic regression model with no predictors.
    2. We add more predictors to the model as long as adding any lowers the value of chosen criterium.
- Backward method
    1. Using the function `glm()` we create a logistic regression model with all possible predictors.
    2. We remove predictors from the model as long as removing any lowers the value of chosen criterium.
- Both method
    1. Using the function `glm()` we create a logistic regression model with no predictors.
    2. In each step we either add or remove a predictor, whichever causes a greater decrease in criterium value.

3. We stop, when adding or removing any feature would increase the criterium value.

Lets compare information criteria of our models.

	AIC	BIC
model AIC forward	800.39	954.98
model AIC backward	800.39	954.98
model AIC both	800.39	954.98
model BIC forward	847.31	889.47
model BIC backward	847.31	889.47
model BIC both	847.31	889.47

Table 1: Information criteria of logistic regression models

Table 1 shows AIC and BIC values of six considered models. By “model `criterium` `method`” we understand logistic regression model, in which features were selected using step method `method` in such a way, that minimizes `criterium`.

Let us note, that the first three models have the same AIC and BIC values, same as the last three. We will thus check, if they are not by chance the same model.

```
Are model AIC forward and model AIC backward the same? TRUE

Are model AIC forward and model AIC both the same? TRUE

Are model BIC forward and model BIC backward the same? TRUE

Are model BIC forward and model BIC both the same? TRUE

Are model AIC forward and model BIC forward the same? FALSE
```

For both criteria, the choice of step method does not impact feature selection. We get two distinct models, one chosen by minimizing AIC, the other by BIC. From now on they will be refered to as `model.AIC` and `model.BIC`. Let us examine the predictors chosen by each of them.

Features in `model.AIC`:

```
1.  status.of.existing.checking.account
2.  duration.in.month
3.  credit.history
4.  purpose
5.  savings.account.and.bonds
6.  present.employment.since
7.  foreign.worker
8.  number.of.existing.credits.at.this.bank
9.  credit.amount
10. installment.rate.in.percentage.of.disposable.income
11. other.installment.plans
12. telephone
```

Features in `model.BIC`:

```
1. status.of.existing.checking.account
2. duration.in.month
3. credit.history
```

`model.AIC` has more features than `model.BIC`. It is to be expected, since the only difference between the criteria is the penalty for the number of parameters. The penalty is bigger in BIC, thus model minimizing it will ignore some predictors, that `model.AIC` consideres important enough.

Feature selection based on Information Value

Fitting logistic regression models to a dataset containing numeric predictors has a degree of risk to it. The model assumes monotonic relation between the probability of success and a continuous feature. There is no guarantee that this relation must hold – it could be the case for instance, that the probability that a client is good is the highest for middle aged clients, but lower for very young and very old ones. Logistic regression will fail to ilustrate this relation, because `age.in.years` is a numeric variable.

This problem can be solved by discretization continuous predictors, which will allow each new class to have its own coefficient. This approach can lead to great increase in the number of parameters, so we should only discretize features we consider important.

To assess predictive power of a feature we will use WoE (Weight of Evidence) and IV (Information Value). In order to estimate importance of feature X, taking k values $(x_1, \ldots, x_k)$, we calculate

$$WoE(x_i) = \log\left(\frac{\mathbb{P}(X = x_i|\texttt{creditability} = \texttt{good})}{\mathbb{P}(X = x_i|\texttt{creditability} = \texttt{bad})}\right)$$

and

$$IV = \sum_{i=1}^{k} \left[\mathbb{P}(X = x_i|\texttt{creditability} = \texttt{good}) - \mathbb{P}(X = x_i|\texttt{creditability} = \texttt{bad})\right] \cdot WoE(x_i).$$

Exact probabilities are of course not known, so we estimate them by taking appropriate frequencies from the training dataset. For instance, we take

$$\mathbb{P}(X = x_i|\texttt{creditability} = \texttt{good}) = \frac{\text{number of good clients that have value } x_i \text{ in } X \text{ feature}}{\text{number of good clients}}.$$

We consider predictor X to be of middle predictive power, when $0.1 \leq IV < 0.3$ and of strong predictive power, when $IV \geq 0.3$.

To find optimal bins for discretization, we use `woebin()` function from `scorecard` package. This function will also merge some classes of factor type features, if it increases the predictive power.

Post discretization, some classes of factor features have been merged (for instance `purpose` now has only 5 classes instead of 10) and also all numeric predictors have been changed to factors. For example, continuous predictor `credit.ammount` has been changed to factor with levels [-Inf,1400), [1400,3400), [3400,4000), [4000,8000), [8000, Inf). Let us examine IV value of every feature.

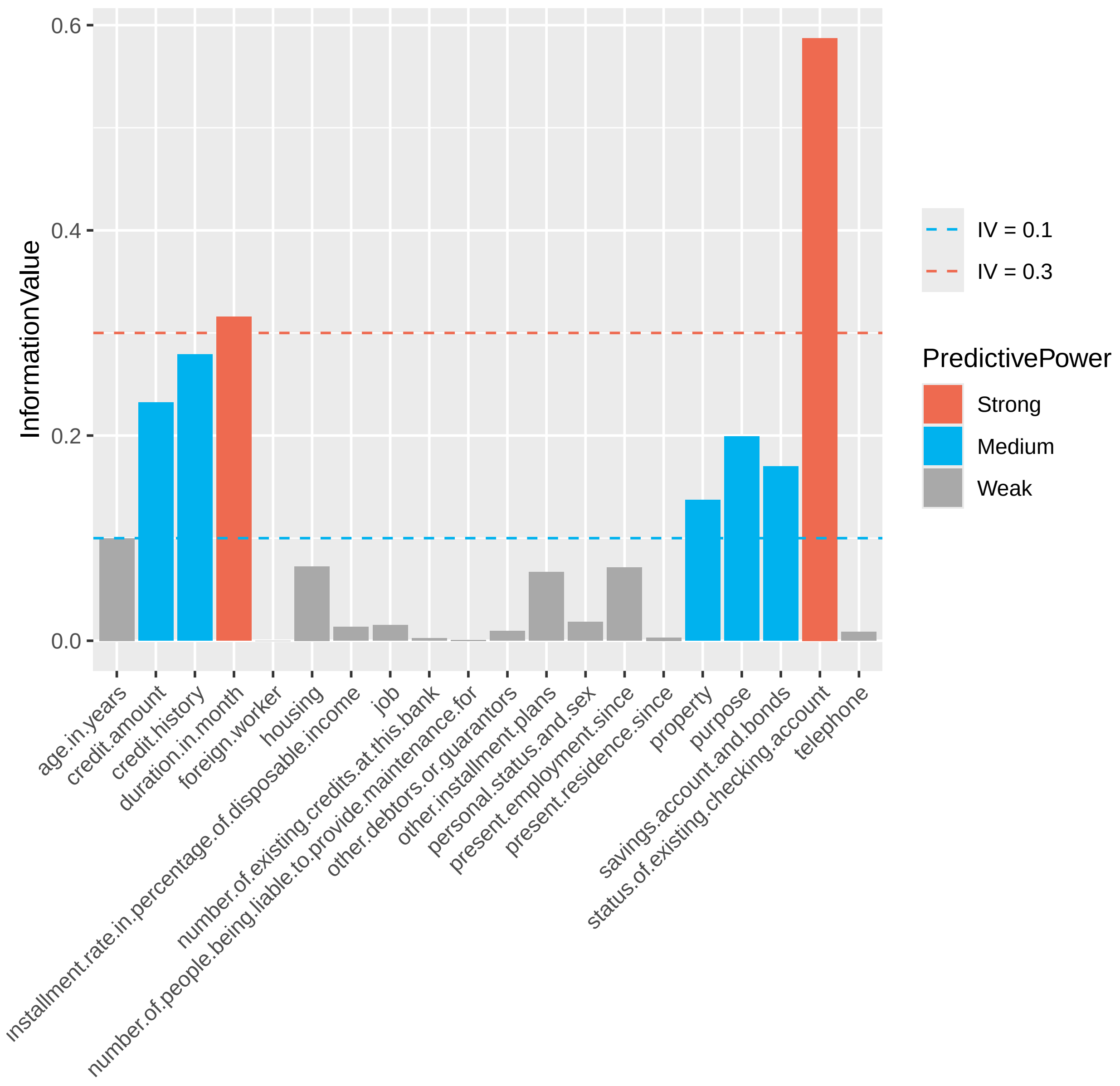

Figure 1: Information Value of discretized features.

In the picture 1, features with strong predictive power ($IV \geq 0.3$) have been coloured orange, while the ones with medium predictive power ($0.1 \leq IV \leq 0.3$) have been marked blue. We will construct two more logistic regression models (called `model.0.3` and `model.0.1`), where the former will only consider features with strong predictive power, while the latter will consider those with medium or above predictive power.

In order to do so, we will modify the training set in such a way, so that the values in it are compliant with our new discretization. This can be easily done using `woebin_ply()` from `scorecard` package.

By discretizing continuous predictors, number of parameters in our models can increase significantly. Let us observe, that `model.0.1` consists of 7 features and 23 parameters. While that number is not huge, if we had more numeric predictors, the increase in number of parameters could cause problems with the existance of their estimates. To prevent that, We can assign to each class of each feature its WoE value, turning them into numeric features, which should greatly reduce the number of parameters. Before we do that, we should check the assumption about monotonic relation between probability of being a good client and WoE values of subsequent classes of considered predictors.

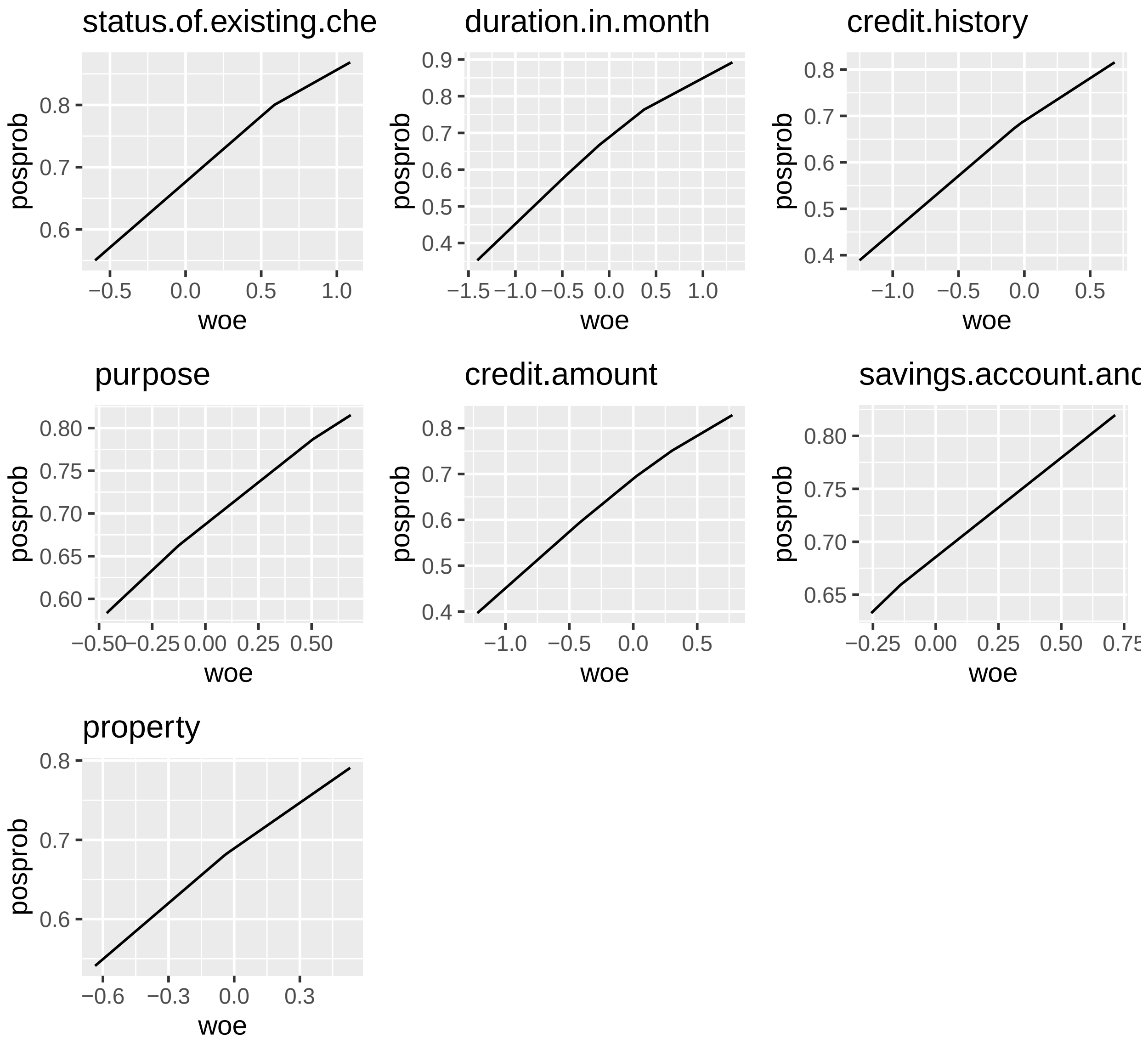

Figure 2: Estimated probability of being a good client based on WoE for features with at least medium predictive power.

Based on figure 2 we conclude, that the monotonicity assumption holds for each predictor. We can thus construct `model.0.3.new` and `model.0.1.new`, which consist of only numeric predictors.

Models created in that way have Exactly one more parameter than the number of features considered by the model (8 in case of `model.0.1.new` and 3 for `model.0.3.new`).

We have created six logistic regression models. Let us now assess their quality.

Binary classifier evaluation

To answer the question which model predicts clients creditability the best, we will make predictions based on the test dataset.

Evaluation based on a confusion matrix

Taking classifier described in (2) with threshold $c = \frac{1}{2}$, we classify clients as either good or bad. The results will be compared to true labels and presented in a confusion matrix, which is a matrix of a form

	bad	good
bad	TN	FP
good	FN	TP

Table 2: Binary classifier confusion matrix

Rows represent true labels of variable `creditability`, while columns represent the models prediction. The values should be understood as:

- TN (True Negative) — number of correctly classified bad clients,
- FP (False Positive) — number of bad clients incorrectly classified as good clients,
- FN (False Negative) — number of good clients incorrectly classified as bad clients,
- TP (True Positive) — number of correctly classified good clients.

	bad	good
bad	23	29
good	23	125

Table 3: Confusion matrix of `model.AIC`.

	bad	good
bad	12	40
good	10	138

Table 4: Confusion matrix of `model.0.3`.

	bad	good
bad	12	40
good	10	138

Table 5: Confusion matrix of `model.0.3.new`.

	bad	good
bad	24	28
good	17	131

Table 6: Confusion matrix of `model.BIC`.

	bad	good
bad	22	30
good	21	127

Table 7: Confusion matrix of `model.0.1`.

	bad	good
bad	19	33
good	19	129

Table 8: Confusion matrix of `model.0.1.new`.

Based on confusion matrices 3–8 we will calculate basic quality measures.

We will focus on measures such as:

- Accuracy
$$ACC = \frac{TP + TN}{TP + TN + FP + FN},$$
- Classification error
$$1 - ACC = \frac{FP + FN}{TP + TN + FP + FN},$$
- Positive prediction value
$$PPV = \frac{TP}{TP + FP},$$
- Negative prediction value
$$NPV = \frac{TN}{TN + FN},$$
- Sensitivity (true positive rate)
$$TPR = \frac{TP}{TP + FN},$$
- Specificity (true negative rate)
$$TNR = \frac{TN}{TN + FP},$$
- F_1-score
$$F_1 = 2\frac{PPV \cdot TPR}{PPV + TPR}.$$

Each of the above takes values from the interval $[0, 1]$, where 0 indicates the worst fit, while 1 indicates perfect classification (except classification error, where the interpretation is reversed).

	accuracy	error	PPV	NPV	sensitivity	specificity	F_1
model.AIC	0.7400	0.2600	0.8117	0.5000	0.8446	0.4423	0.8278
model.BIC	0.7750	0.2250	0.8239	0.5854	0.8851	0.4615	0.8534
model.0.3	0.7500	0.2500	0.7753	0.5455	0.9324	0.2308	0.8466
model.0.1	0.7450	0.2550	0.8089	0.5116	0.8581	0.4231	0.8328
model.0.3.new	0.7500	0.2500	0.7753	0.5455	0.9324	0.2308	0.8466
model.0.1.new	0.7400	0.2600	0.7963	0.5000	0.8716	0.3654	0.8323

Table 9: Scoring models quality measures.

Based just on table 9 it is not yet possible to conclude, which model is the best, since no one model is the best looking at all measures at once. It is worth noting though, `model.BIC` is the best by six out of seven measures, despite its simplicity (only 9 parameters).

Evaluation based on ROC curve

We will also make use of a different approach of quality assessment, based on a ROC curve. In order to define said curve, we will need to define sensitivity and specificity of a diagnostic test of a classifier defined in (2). Let $p(\mathbf{X})$ be a random variable representing probability, that a client with characteristic $\mathbf{X}$ is a good client and E be the label ("good"/"bad").

For a fixed c, we define

- $F(c) = \mathbb{P}(p(\mathbf{X}) \leq c \,|\, E = \text{"bad"})$ – cumulative distribution function of $p(\mathbf{X})$ in the group of bad clients,
- $G(c) = \mathbb{P}(p(\mathbf{X}) \leq c \,|\, E = \text{"good"})$ – cumulative distribution function of $p(\mathbf{X})$ in the group of good clients,
- $SE(c) = 1 - G(c)$ – sensitivity of a diagnostic test of classifier d,
- $SP(c) = F(c)$ – specificity of a diagnostic test of classifier d.

ROC curve is a set

$$ROC = \{(1 - SP(c), SE(c)) : -\infty \leq c \leq \infty\}.$$

We only need to consider $c \in [0, 1]$, because random variable $p(\mathbf{X})$ takes only values from this interval.

In practice, for $F(\cdot)$ and $G(\cdot)$ we take empirical distribution functions $F_n(\cdot)$ and $G_m(\cdot)$, calculated using data from groups of good and bad clients.

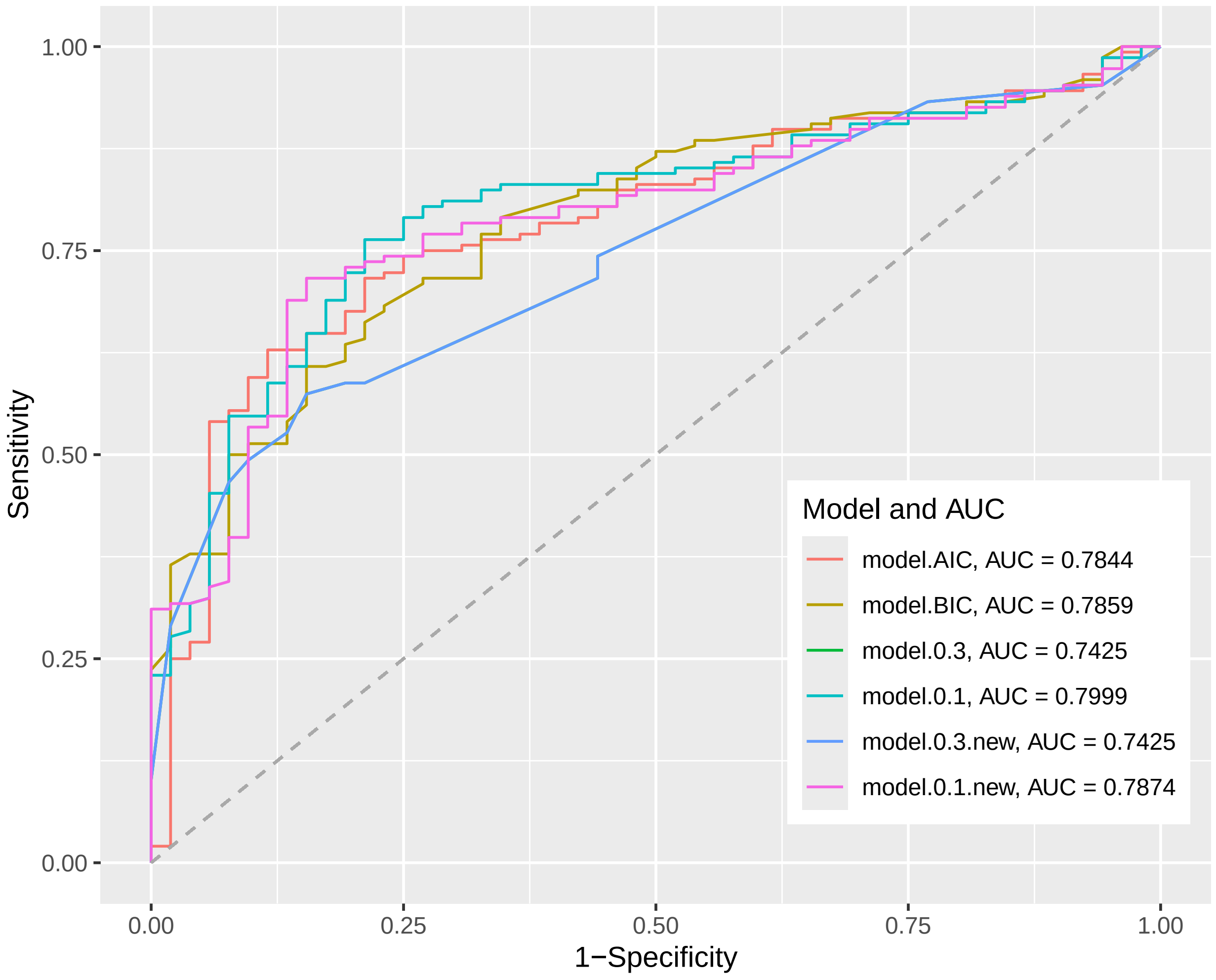

Figure 3: ROC curves of considered models.

From figure 3 we see, that the ROC curves intersect. In order to decide which model is the best, we can compare the areas between their curves, i.e. AUC values. The biggest AUC is obtained for ROC curve of `model.0.1`.

Evaluation based on Kolmogorov-Smirnov statistic

The last measure we will consider is the Kolmogorov-Smirnov statistic. Its value is defined as

$$D = \sup_{t\in[0,1]} |F_n(t) - G_n(t)|,$$

where $F_n(\cdot)$ and $G_m(\cdot)$ are empirical distribution functions defined as before.

The statistic takes values from the interval $[0, 1]$, where bigger number indicates better ability to differentiate between good and bad clients. The values of $K - S$ statistic of our models we present in table 10.

	model.AIC	model.BIC	model.0.3	model.0.1	model.0.3.new	model.0.1.new
Statystyka K-S	0.5130	0.4543	0.4205	0.5520	0.4205	0.5624

Table 10: Kolmogorov-Smirnov statistic values.

Kolmogorov-Smirnov statistic values indicate, that `model.0.1.new` provides the best distinction between good and bad clients, with `model.0.1` being a close second.

Choosing the best model

Having six models and nine quality measures (7 based on confusion matrix, area under ROC curve and $K - S$ statistic), we will try to determine, which model is in some sense the best.

Based on measures derived from confusion matrices (table 9), the most promising model is `model.BIC`. We also consider models `model.AIC` and `model.0.1` worth considering, since they give satisfying results for each of the measures. Other models we deem worse, as they have much worse specificity value.

Biggest areas under ROC curves we get successively for `model.0.1`, `model.0.1.new`, `model.BIC` and `model.AIC`. Other models we consider worse, because their AUC values are significantly lower.

Based on Kolmogorov-Smirnov statistic, the best ability to distinguish between good and bad clients is shown by `model.0.1.new` and `model.0.1`.

We take `model.0.1` as the best one, because apart from having the biggens area under ROC curve, it provided good values of every other considered measure.

Optimal threshold selection

Until now, we have only considered threshold $c = \frac{1}{2}$, i.e. client was predicted to be good, if the probability of them being a good client was at least 50%. Let us now consider other c values, selected for each model individually.

By an optimal threshold we understand such a number c, such that classifier (2) optimizes some criterium. Of course, for different criteria we can obtain very different thresholds. In `optimalCutoff` package we have over 30 criteria, but we will only focus on three:

1. minimizing $\sqrt{FPR^2 + FNR^2}$,
2. maximizing $TP + TN$,
3. maximizing Youden index: $TPR + TNR - 1$.

FPR and FNR represent False Positive Rate and False Negative Rate, defined as

$$FPR = \frac{FP}{FP + TN}, \qquad FNR = \frac{FN}{FN + TP}.$$

Criterium 2 is equivalent with maximizing ACC.

	model.AIC	model.BIC	model.0.3	model.0.1	model.0.3.new	model.0.1.new
kryterium 1.	0.70	0.69	0.76	0.67	0.79	0.72
kryterium 2.	0.39	0.49	0.45	0.58	0.44	0.65
kryterium 3.	0.78	0.75	0.76	0.67	0.79	0.72

Table 11: Optimal thresholds.

Let us verify, if thresholds in table 11 indeed optimize corresponding criteria.

	model.AIC	model.BIC	model.0.3
criterium 1 for threshold 0.5	0.5789	0.5506	0.7722
criterium 1 for optimal threshold	0.3540	0.3926	0.4526
criterium 2 for threshold 0.5	0.7400	0.7750	0.7500
criterium 2 for optimal threshold	0.7650	0.7750	0.7500
criterium 3 for threshold 0.5	0.2869	0.3467	0.1632
criterium 3 for optimal threshold	0.5130	0.4543	0.4205

Table 12: Selected criteria values for the first three models.

	model.0.1	model.0.3.new	model.0.1.new
criterium.1.for.threshold.0.5	0.5941	0.7722	0.6475
criterium.1.for.optimal.threshold	0.3224	0.4526	0.3288
criterium.2.for.threshold.0.5	0.7450	0.7500	0.7400
criterium.1.for.optimal.threshold.1	0.7800	0.7500	0.7600
criterium.3.for.threshold.0.5	0.2812	0.1632	0.2370
criterium.1.for.optimal.threshold.2	0.5452	0.4205	0.5556

Table 13: Selected criteria values for the second three models.

Tables 12 and 13 show expected results. For each model and criterium, optimal threshold always provides better or identical results as thresholds $\frac{1}{2}$.