Project Report:

Bank Marketing Dataset: An overview of classification algorithms

CS229: Machine Learning

Henrique Ap. Laureano
ID 158811

/KAUST/CEMSE/STAT

Spring Semester
2018

Contents

Data and goals

In this project we study different approachs to predict the sucess of bank telemarketing. As instrument we have a dataset related with direct marketing campaigns based on phone calls of a Portuguese banking institution. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be (yes) or not (no) subscribed.

The data under study here is called Bank Marketing Dataset (BMD) and he was found in the Machine Learning Repository (UCI). The data is public available in the url https://archive.ics.uci.edu/ml/datasets/Bank+Marketing#. The size of the dataset is considerably large, especially if we consider its origin. Data from clients of financial institutions are usually difficult to find, and when found, are rarely available in this quantity. In the BMD data we have 41188 observations, with eighteen features.

The eighteen features are briefly described in Table 1, were in the left column we have the original feature name in the dataset, and in the right column its description, mentioning also if the feature is numeric, categorial, and with how many levels (if categorical, of course). The first one called of y is the response, the desired target. The other features are presented in the same order that they appear in the dataset.

To know better the data some descriptive analysis is performed, see Figure 1 and Figure 2.

Table 1: Features description of the Bank Marketing Dataset (BMD).

Feature	Description
y	desired target. has the client subscribed a term deposit? (no, yes)
age	numeric
job	type of job, twelve categories
marital	marital status, four categories
eduacation	eight categories
housing	has housing loan? (no, yes, unknown)
loan	has personal loan? (no, yes, unknown)
contact	contact communication type (cellular, telephone)
month	last contact month of year (twelve levels, months)
day.of.week	last contact day of the week (five levels, days)
campaign	number of contacts performed during this campaign and for this client
previous	number of contacts performed before this campaign and for this client
poutcome	previous marketing campaign (failure, nonexistent, success)
emp.var.rate	numeric. employment variation rate - quarterly indicator
cons.price.idx	numeric. consumer price index - monthly indicator
cons.conf.idx	numeric. consumer confidence index - monthly indicator
euribor3m	numeric. euribor 3 month rate - daily indicator
nr.employed	numeric. number of employees - quarterly indicator

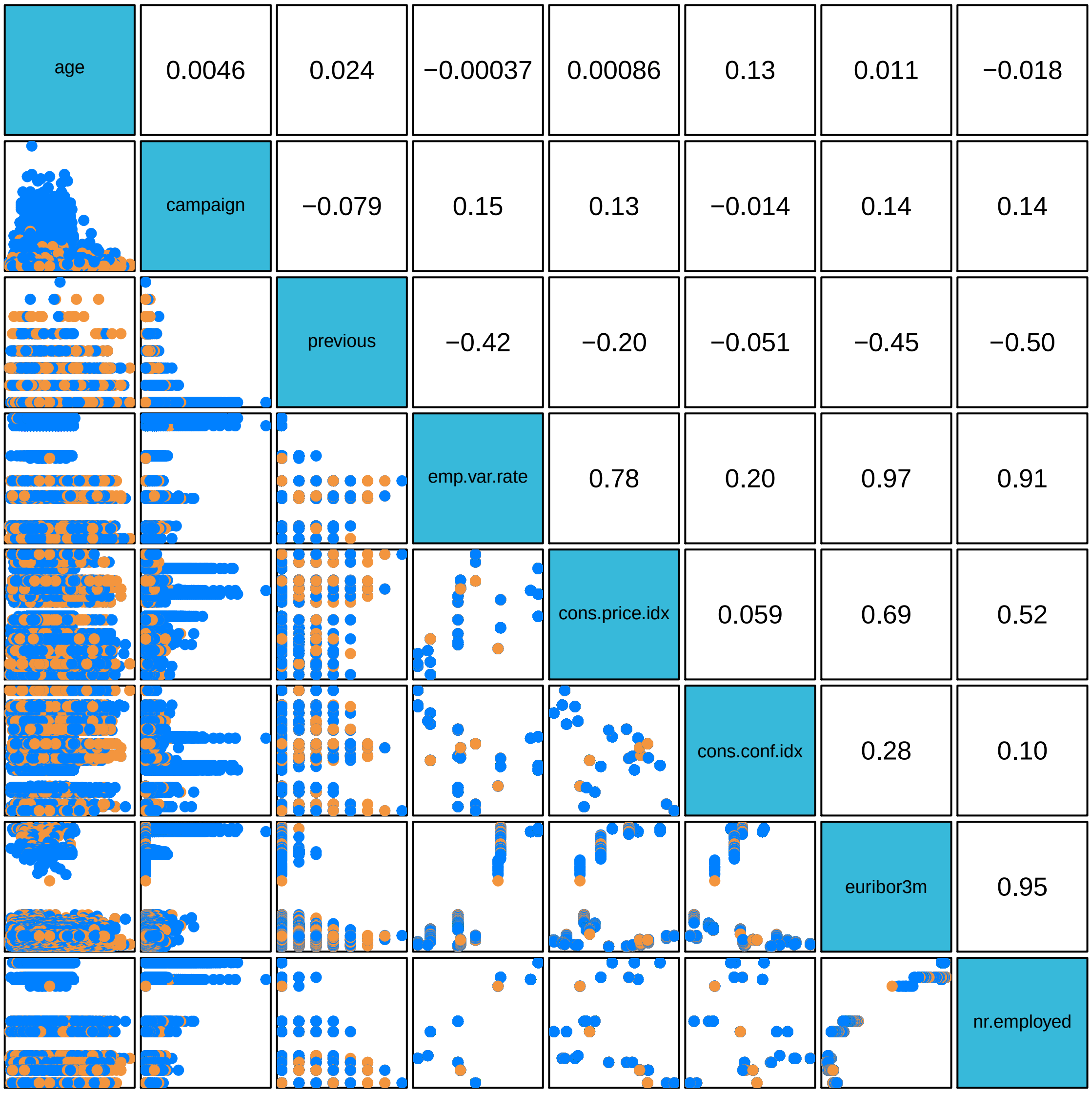


Figure 1: Scatterplot lower triangular matrix and correlation upper triangular matrix for all the quantitative features presented in the Bank Marketing Dataset (BMD).

In Figure 1 we see the scatterplots and correlations, two-by-two, for all the eight numerical features in the BMD. In more than half of them we see a random behaviour, that is also described by a correlation close to zero or between the interval -0.3 and 0.3. A (very) strong (and positive) correlation is seen in three cases. `emp.var.rate` vs. `euribor3m` (cor. 0.97), `euribor3m` vs. `nr.employed` (cor. 0.95), and `emp.var.rate` vs. `nr.employed` (cor. 0.91), i.e., involving only three features - employment variation rate, Euro Interbank Offered Rate (Euribor) and number of employees. During the analysis this point can be better studied.

Already in Figure 2 we have the frequencies for each level of the categorical features in the BMD. First, we see that the desired target is unbalanced, with more than 85% of the observations corresponding to clients that didn't subscribed to a term deposit. An equilibrium between levels is only present in the `day.of.week` last contact feature. By this Figure we can also see that the last contact of most of the clients was in may (`month` feature), that most of the clients have a nonexistent previous marketing campaign (`poutcome` feature), that they are married (`marital` feature) and that most have a `job` in the administrative sector.

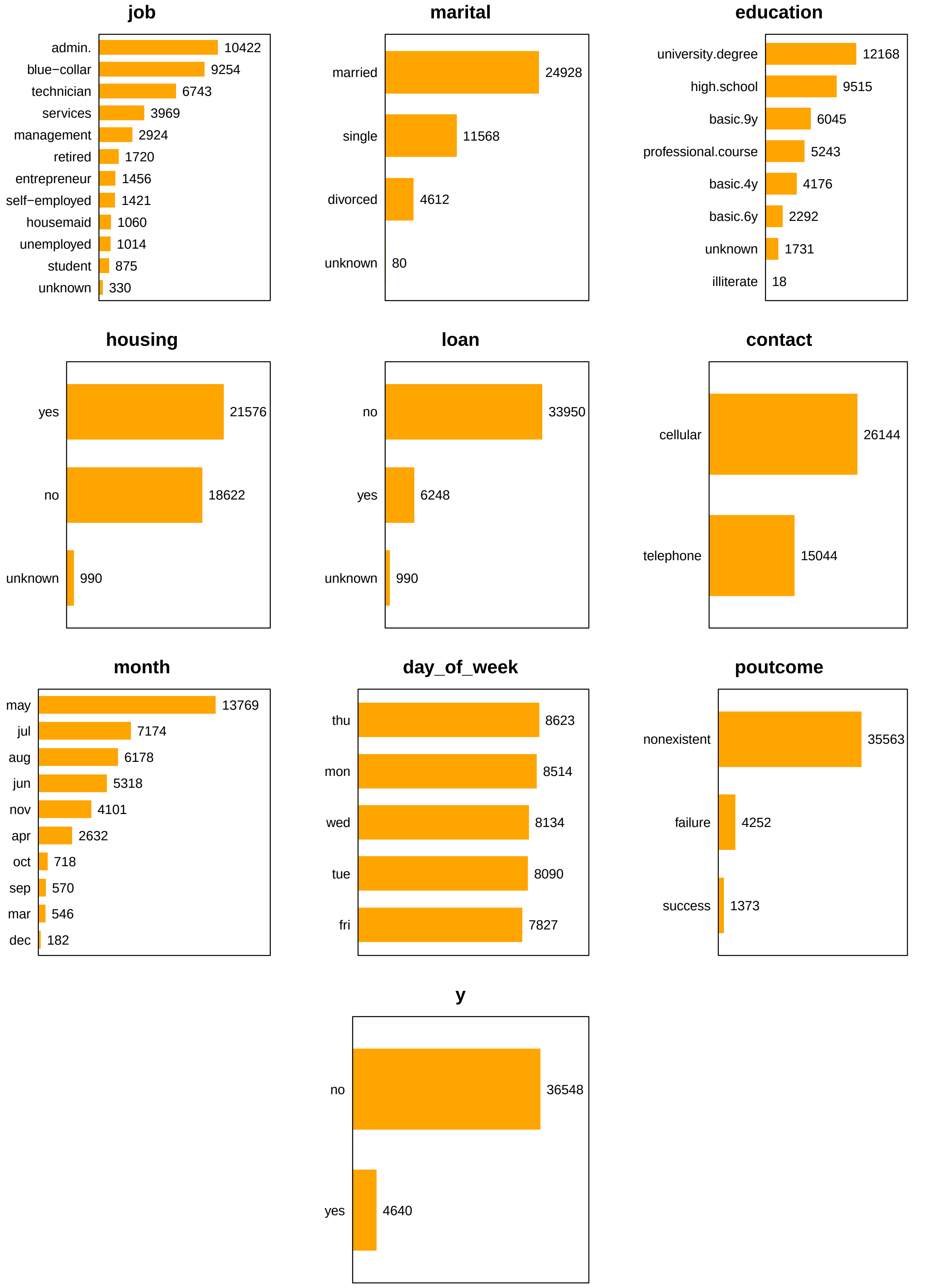


Figure 2: Bar plots for all the qualitative features presented in the Bank Marketing Dataset (BMD).

Using this features, described in Table 1, the goal here is test several algorithms to see how good they are to predict the desired target, i.e., predict, given the seventeen features, if the bank term deposit would be or not subscribed. The algorithms used for this task are described in the next section, together with some extra informations about the analysis procedure.

Methods

To predict, given the seventeen features, if the bank term deposit would be or not subscribed, fifteen algorithms are used. They are: four generalized linear models with a Bernoulli response and with different link functions (logit, probit, cauchit and complementary log-log); a standard linear regression model; naive Bayes classifier; three discriminant analysis algorithms (linear, quadratic and regularized); four support vector machines with different kernels (linear, polynomial, radial and sigmoid); a random forest; and a decision tree.

As mentioned before, the BMD consists of 41188 observations. A random sample of 10% of this size, 4119 observations, was withdrawn to be used as a test dataset. The rest, 37069 observations, was used as a train dataset.

All the analysis are performed using the `R` [1] language and environment for statistical computing. To take advantage of the most efficient available algorithm versions, we use some `R` libraries where the algorithms are implemented. A brief description of the algorithms is given now, always mentioning the corresponding `R` library where the algorithm is implemented.

Generalized Linear regression Model (GLM) with Bernoulli response

The GLM is a flexible generalization of ordinary linear regression that allows responses with error distribution models other than a normal distribution. The GLM generalizes linear regression by allowing the linear model to be related to the response via a link function. In a GLM each outcome Y of the response is assumed to be generated from a particular distribution in the exponential family, a large range of probability distributions. The mean, μ, of the distribution depends on the features, X, through:

$$\mathrm{E}(Y) = \boldsymbol{\mu} = g^{-1}(X\boldsymbol{\beta}),$$

where $\mathrm{E}(Y)$ is the expected value of Y; $X\boldsymbol{\beta}$ is the linear predictor, a linear combination of unknown parameters $\boldsymbol{\beta}$; g is the link function. The unknown parameters, $\boldsymbol{\beta}$, are typically estimated with maximum likelihood.

When the response data, Y, are binary (taking on only values 0 and 1), the distribution function is generally chosen to be the Bernoulli distribution and the interpretation of μ_i is then the probability, p, of Y_i taking on the value one. The logit is the canonical link function and when used the resulting model is called of logistic regression. However, other link function can be used. The four most popular link functions, and used here, are:

- Logit function: $g(p) = \ln\left(\frac{p}{1-p}\right)$;
- Probit or inverse Normal function: $g(p) = \Phi^{-1}(p)$;
- Cauchit function: $\tan\left(\pi p - \frac{\pi}{2}\right)$;
- Complementary log-log function: $g(p) = \log(-\log(1-p))$.

To test the significance of the features we use the Akaike information criterion (AIC). Given a collection of models, the AIC estimates the quality of each model, relative to each of the other models. Thus, AIC provides a means for model selection. The AIC value of a given model is the following:

$$\mathrm{AIC} = 2par - 2\log \widehat{L}.$$

With $\widehat{L}$ being the maximum value of the likelihood function for the model and *par* being the number of estimated parameters in the model.

More details about GLM can be see, for example, in `https://en.wikipedia.org/wiki/Generalized_linear_model`.

Linear regression Model (LM)

LM is a linear approach to modelling the relationship between a response and features, where the response have a normal error distribution. Commonly, the conditional mean of the response given the values of the features is assumed to be an affine function of those values. This relationship is modeled through a disturbance term or error ϵ - an unobserved feature that adds "noise" to the linear relationship between the response and features. Thus the model takes, in matrix notation, the form

$$Y = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

The unknown parameters, $\boldsymbol{\beta}$, are typically estimated via least squares. Ordinary Least Squares (OLS) method minimizes the sum of squared residuals, and leads to a closed-form expression for the estimated value of the unknown parameter $\boldsymbol{\beta}$

$$\widehat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top Y.$$

More details about LM can be see, for example, in `https://en.wikipedia.org/wiki/Linear_regression`.

Naive Bayes classifier

Naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Naive Bayes is a reference to the use of Bayes' theorem in the classifier's decision rule, but is not (necessarily) a Bayesian method.

Considering each attribute and class label as random variables, given a record with attributes $A_1, A_2, \ldots, A_n$, and a goal - predict class C, we want to find the value of C that maximizes $\mathrm{P}(A_1, A_2, \ldots, A_n \mid C)\mathrm{P}(C)$. The naive Bayes classifier assume independence among attributes A_i when class is given, therefore

$$\mathrm{P}(A_1, A_2, \ldots, A_n \mid C) = \mathrm{P}(A_1 \mid C_j)\mathrm{P}(A_2 \mid C_j) \ldots \mathrm{P}(A_n \mid C_j).$$

This approach greatly reduces the computation cost - only counts the class distribution, and can estimate $\mathrm{P}(A_i \mid C_j)$ for all A_i and C_j. A new point is classified to C_j if $C_j \prod \mathrm{P}(A_i \mid C_j)$ is maximal.

In `R` the main implementation of the naive Bayes classifier is found in the `e1071` library [2].

Discriminant Analysis

Linear Discriminant Analysis (LDA) or discriminant function analysis is a generalization of Fisher's linear discriminant, a method used to find a linear combination of features that characterizes or separates two or more classes of objects or events.

LDA approaches the problem by assuming that the conditional probability density functions (considering two classes) are both normally distributed with mean and covariance parameters. Under this assumption, the Bayes optimal solution is to predict points as being from the second class if the log of the likelihood ratios is bigger than some threshold. Without any further assumptions, the resulting classifier is referred to as QDA (Quadratic Discriminant Analysis). LDA instead makes the additional simplifying homoscedasticity assumption (i.e. that the class covariances are identical). More details about can be see, for example, in `https://en.wikipedia.org/wiki/Linear_discriminant_analysis`.

Considering two more parameters that flexibilize the possible difference between the covariance matrices between classes and the dependence between the same covariances, we have a Regularized Discriminant Analysis (RDA).

In `R` the main implementation for LDA and QDA is found in the `MASS` library [3], and the main implementation for RDA is found in the `klaR` library [4].

Support Vector Machine (SVM)

With SVM a data point is viewed as a p-dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a (p-1)-dimensional hyperplane. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane. More formally, a SVM constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outliers detection.

Often happens that the sets to discriminate are not linearly separable in that space. We can construct nonlinear classifiers applying the kernel trick, $K(x_i, x_j)$, to maximum-margin hyperplanes. Here we use the four most common kernels in SVM

- Linear: $K(x_i, x_j) = \langle x_i, x_j \rangle$
- Polynomial $K(x_i, x_j) = (c_0 + \gamma \langle x_i, x_j \rangle)^d$
- Radial: $K(x_i, x_j) = \exp(-\gamma \left\| x_i, x_j \right\|^2)$
- Sigmoid: $K(x_i, x_j) = \tanh(c_0 + \gamma \langle x_i, x_j \rangle)$

More details about can be see, for example, in `https://en.wikipedia.org/wiki/Support_vector_machine`. In `R` the main implementation of SVM is found in the `e1071` library [2].

Random forest

Random forests are an ensemble learning method that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

Random forests differ in only one way from tree bagging: they use a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. Tree bagging repeatedly selects a random sample with replacement of the training set and fits trees to these samples. This bootstrapping procedure leads to better model performance because it decreases the variance of the model, without increasing the bias. More details about can be see, for example, in `https://en.wikipedia.org/wiki/Random_forest`.

In `R` the main implementation of random forest is found in the `randomForest` library [5].

Decision tree

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

A decision tree is a flowchart-like structure in which each internal node represents a ”test” on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label. The paths from root to leaf represent classification rules. Algorithms for constructing decision trees usually work top-down, by choosing a variable at each step that best splits the set of items. Different algorithms use different metrics for measuring ”best”. These generally measure the homogeneity of the target within the subsets and are applied to each candidate subset, the resulting values are combined (e.g., averaged) to provide a measure of the quality of the split.

In `R` the main implementation of decision tree is found in the `rpart` library [6].

Results

With the GLM's and LM we are able to do feature selection. Here we do this via AIC. Which features are keeped and which features are dropped can be seen in Table 2.

The main measure that can be used to compare all the fifthteen algorithms is the Receiver Operating Characteristic curve, i.e. ROC curve. A graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the specificity, true negative rate, against the sensitivity, true positive rate, at various threshold settings. More details about can be see, for example, in `https://en.wikipedia.org/wiki/Receiver_operating_characteristic`. In `R` the main implementation of the ROC curve is found in the `pROC` library [7].

When dealing with ROC curves the main measure returned is the Area Under the Curve (AUC), that is qual to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. The AUC for each model is presented in Figure 3. The highest is obtained with the probit link function in the GLM.

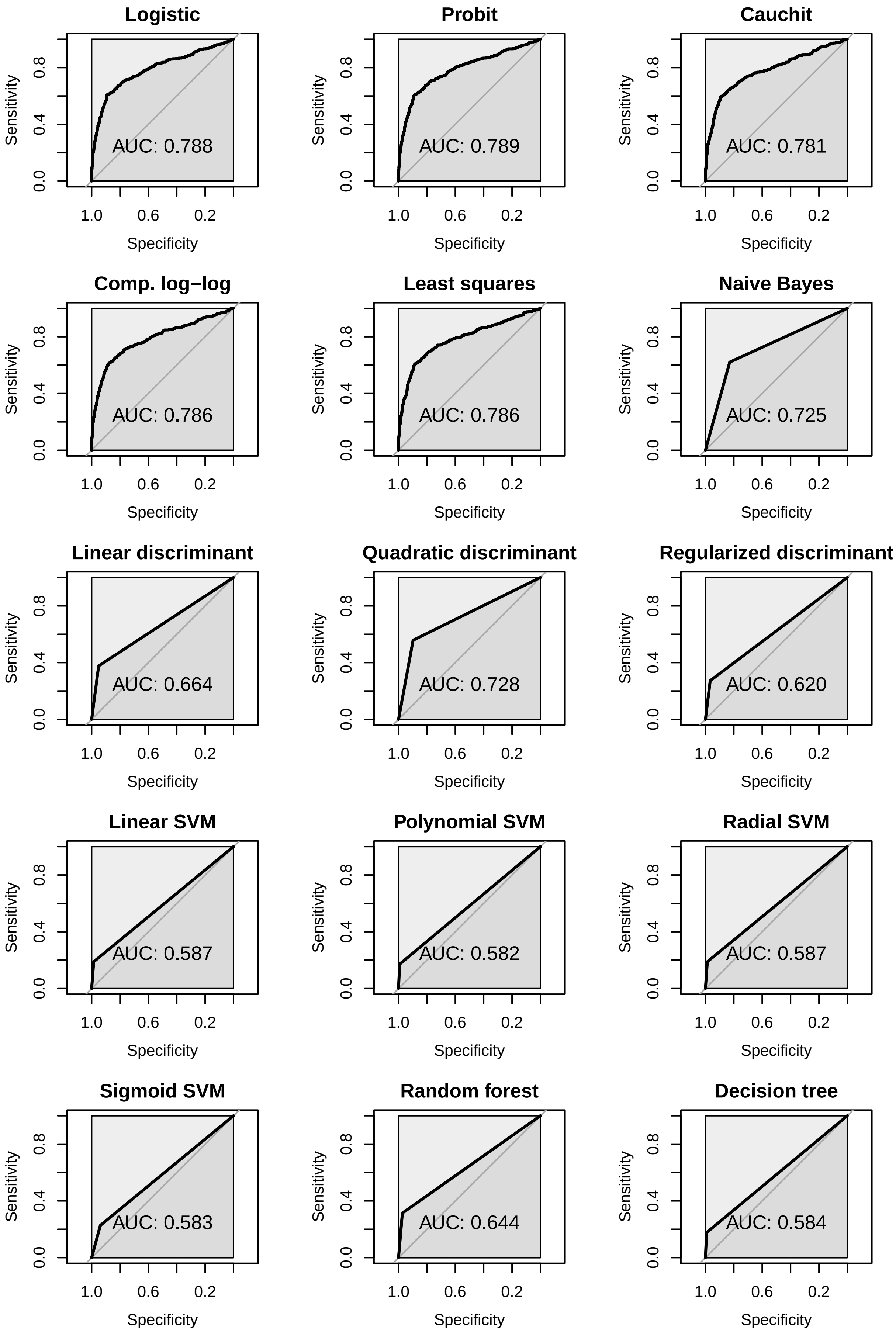
Others measures as the specificity, sensitivity and the risk, are presented in Table 3. The risk here is defined as the proportion of observations in the test dataset that are wrongly classified by the trained model.

The GLM's and the LM approachs return a probability for each observation, where more close to zero means that the observation is more likely to be provinient from the `no` class - a client that did not subscribed a term deposit. However, with the ROC curve we obtain a optimized threshold for this decision. Thus, to compute the risk in this models we use this obtained threshold, instead the default value half - that in a first moment is the logical choice, since the returned probability in between zero and one. This optimal threshold is defined as the cutting point that returns the best specificity and sensitivity - in general we don't want a good specificity value but with a bad sensitivity, or vice-versa. We want the best possible value - combination - for both, at the same time. So the threshold of this scenario is the used value to compute the risk in this models. The others algorithms return directly the class label, not a probability.

Again, this values - specificity, sensitivity and the classification risk/error - can be checked in Table 3.

Table 2: Remaining features in each model after features selection by AIC.

Feature	**Model**				
	Logistic	**Probit**	**Cauchit**	**Comp. log-log**	**Least squares**
`age`					
`job`	✓	✓	✓	✓	✓
`marital`					
`eduacation`					
`housing`					
`loan`					
`contact`	✓	✓	✓	✓	✓
`month`	✓	✓	✓	✓	✓
`day.of.week`	✓	✓	✓	✓	✓
`campaign`	✓	✓	✓	✓	✓
`previous`					
`poutcome`	✓	✓	✓	✓	✓
`emp.var.rate`	✓	✓	✓	✓	✓
`cons.price.idx`	✓	✓	✓	✓	✓
`cons.conf.idx`	✓	✓	✓	✓	✓
`euribor3m`	✓	✓	✓		✓
`nr.employed`	✓	✓	✓	✓	



Logistic

Probit

Cauchit

Comp. log−log

Least squares

Naive Bayes

Linear discriminant

Quadratic discriminant

Regularized discriminant

Linear SVM

Polynomial SVM

Radial SVM

Sigmoid SVM

Random forest

Decision tree

Figure 3: ROC curve for each model (in the test) with respective AUC and thresholds.

Table 3: Specificity, sensitivity and risk for each fitted model in the test Bank Marketing Dataset (BMD), in bold we have the best performances. The models in bold are the models with the best AUC.

Model	**Specificity**	**Sensitivity**	**Risk**
Logistic regression (GLM with logit link)	0.891	0.609	0.212
Probit regression (GLM with probit link)	0.888	0.609	0.197
Cauchit regression (GLM with cauchit link)	0.893	0.594	0.261
Comp. log-log regression (GLM with comp. log-log link)	0.878	**0.614**	0.265
Least squares regression (linear regression)	0.886	0.609	0.16
Naive Bayes	0.829	0.621	0.192
Linear discriminant analysis	0.950	0.377	0.107
Quadratic discriminant analysis	0.897	0.558	0.137
Radial discriminant analysis	0.966	0.278	0.103
Linear support vector machine	0.985	0.188	0.095
Polynomial support vector machine	**0.992**	0.171	**0.09**
Radial support vector machine	0.987	0.188	0.094
Sigmoid support vector machine	0.939	0.227	0.133
Random forest (bagging)	0.973	0.309	0.093
Decision tree	**0.992**	0.176	**0.09**

Conclusion

Keep a feature means that the feature was significant, statistically significant, in describing the difference between the classes of the desired target - if the bank term deposit would be or not subscribed. In Table 2 we can see a very high concordance between the models, in a general form. Each model finished with eleven, from seventeen, features. This are the dropped, nonsignificant in describing the difference between classes, features in all models: `age`, `marital` status, `education`, `housing` loan, personal `loan`, and `previous` number of contacts performed before this campaign and for this client.

Looking by the AUC, Figure 3, the best model is the GLM with probit link function. However, very similar values are obtained with the others link functions and with the LM. With the other algorithms the AUC's are considerable smaller, but always above 0.55 (a not bad value, but also not so good). Looking to the other computed measures in Table 3, we see that for all the algorithms we obtain a very good specificity, true negative rate, and a bad or not so good sensitivity, true positive rate. For all the algorithms we have a good risk value, less than 0.30, but a very good risk value is obtained only with the non-GLM/LM techniques.

The best specificities - pratically perfect - are obtained with the SVM with polynomial kernel and with the decision tree. However, the sensitivities are very low (nevertheless the risks are the lowest). The best sensitivity is obtained with the GLM with complementaty log-log link function. The corresponding specificity of 0.878 is still pretty good, but the risk of 0.265 is not.

To summarize, in a general form we obtain a good specificity with all algorithms - with some a vey good specificity. However, we only obtain a sensitivity above 0.5 with the GLM's, LM, naive Bayes and quadratic discriminant analysis. Using as a final criterium the risk in the models with specificity and sensitivity above 0.6, we have the naive Bayes, LM and GLM with probit link function. This three present very similar measures, making very hard to choose one between them tree.

References

[1] R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `https://www.R-project.org/`.

[2] Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A. and Leisch, F. (2017). e1071: Misc Functions of the Department of Statistics, Probability Theory Group, TU Wien. R package version 1.6-8. `https://CRAN.R-project.org/package=e1071`.

[3] Venables, W. N. & Ripley, B. D. (2002). Modern Applied Statistics with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0. `http://www.stats.ox.ac.uk/pub/MASS4`.

[4] Weihs, C., Ligges, U., Luebke, K. and Raabe, N. (2005). klaR Analyzing German Business Cycles. In Baier, D., Decker, R. and Schmidt-Thieme, L. (eds.). Data Analysis and Decision Support, 335-343, Springer-Verlag, Berlin.

[5] Liaw, A. & Wiener M. (2002). Classification and Regression by randomForest. R News 2(3), 18–22. `http://CRAN.R-project.org/doc/Rnews/`.

[6] Therneau, T., Atkinson, B. and Ripley, B. (2017). rpart: Recursive Partitioning and Regression Trees. R package version 4.1-11. `https://CRAN.R-project.org/package=rpart`.

[7] Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, JC. and Müller, M. (2011). pROC: an open-source package for R and S+ to analyze and compare ROC curves. BMC Bioinformatics, 12, p. 77. DOI: 10.1186/1471-2105-12-77. `http://www.biomedcentral.com/1471-2105/12/77/`.