Warsaw University of Technology

FACULTY OF
MATHEMATICS AND INFORMATION SCIENCE

Master's diploma thesis

in the field of study Mathematics
and specialisation Mathematical Statistics and Data Analysis

auditor: an R package and methodology for validation of statistical models

Alicja Gosiewska

student record book number 287714

thesis supervisor
dr hab. inż. Przemysław Biecek, prof. PW

WARSAW 2018

....

supervisor's signature author's signature

Abstract

auditor: an R package and methodology for validation of statistical models

Machine learning has spread to almost every area of life. It is successfully applied in biology, medicine, finance, physics, and other fields. The problem arises if models fail when confronted with the real-world data. Therefore, there is a need for validation methods.

This paper describes methodology and tools for model-agnostic audit. Introduced techniques facilitate assessing and comparing the goodness of fit and performance of models. In addition, they may be used for analysis of the similarity of residuals and for the identification of outliers and influential observations. The examination is carried out by diagnostic scores and visual verification.

Presented methods are implemented in the **auditor** package for R. Due to the flexible and consistent grammar, it is simple to validate models of any classes.

Keywords: machine learning, R, diagnostic, visualization, modeling

Streszczenie

auditor: narzędzie i metodologia diagnostyki i walidacji modeli statystycznych

Uczenie maszynowe z powodzeniem znajduje zastosowanie w różnych obszarach nauki oraz biznesu. Odgrywa ono istotną rolę w biologii, medycynie, fizyce, finansach oraz wielu innych dziedzinach. Jednakże, poważny problem stanowią modele, które zawodzą w konfrontacji z rzeczywistymi danymi. Dlatego też, tak ważna jest ich walidacja.

W niniejszej pracy opisano metodologię oraz narzędzia służące do audytu dowolnych modeli predykcyjnych. Zaprezentowane techniki umożliwiają, między innymi, identyfikację wartości odstających i obserwacji wpływowych. Ponadto, ułatwiają one ocenę dokładności oraz jakości dopasowania modelu. Mogą być wykorzystane również do analizy podobieństwa reszt. Walidacja przeprowadzana jest na podstawie statystyk diagnostycznych oraz oceny wizualnej.

Opisane metody zostały zaimplementowane w pakiecie **auditor**, napisanym w języku R. Elastyczność oraz spójność funkcji umożliwiają wygodną oraz prostą walidację dowolnych klas modeli.

Słowa kluczowe: uczenie maszynowe, R, diagnostyka, wizualizacja, modelowanie

Warsaw, 27.08.2018

Declaration

I hereby declare that the thesis entitled „auditor: an R package and methodology for validation of statistical models", submitted for the Master degree, supervised by dr hab. inż. Przemysław Biecek, prof. PW, is entirely my original work apart from the recognized reference.

.....

Acknowledgements

Firstly, I would like to express my gratitude to my supervisor Professor Przemysław Biecek for his patient guidance, insightful comments, and encouragement to complete this master thesis.

Furthermore, I wish to thank Dr. Rafael De Andrade Moral for his precious assistance. I would also be glad to acknowledge all members of the MI^2 Data Lab for providing a pleasant workplace and a friendly atmosphere, in particular, I wish to thank Aleksandra Grudziąż and Mateusz Staniak for valuable discussions.

Finally, I would like to express my very profound gratitude to my parents for providing me with unfailing support throughout my years of study and through the process of researching and writing this thesis.

The work was supported by NCN Opus grant 2016/21/B/ST6/02176.

Contents

Introduction

Predictive modeling is a process that uses mathematical and computational methods to forecast outcomes. Lots of algorithms in this area have been developed and are still developing. Therefore, there are countless possible models to choose from and a lot of ways to do it. A poorly- or over-fitted model usually will be of no use when confronted with future data. Its predictions will be misleading (Sheather, 2009)[48] or harmful (O'Neil, 2016)[42]. That is why methods that support model diagnostic are important.

Diagnostic is often carried out only by checking model assumptions. However, it is usually neglected for complex machine learning models, since many of them are used as if they were assumption free. Still, there is a need to verify their quality. As the diagnostic, or audit, we consider a broad approach to model exploration. This includes three objectives.

- **Objective 1:** Enrichment of information about model performance.
- **Objective 2:** Identification of outliers, influential and abnormal observations.
- **Objective 3:** Examination of other problems with a model by analyzing distributions of residuals. In particular problems with bias, heteroscedasticity of variance and autocorrelation of residuals.

In this paper, we introduce the **auditor** package for R, which is a tool for diagnostic and visual verification. As it focuses on residuals[1] and does not require any additional model assumptions, most of the presented methods are model-agnostic. A consistent grammar across various tools reduce the amount of effort needed to create informative plots and makes the validation more convenient and available.

Diagnostic methods have been a subject of much research (Atkinson, 1985)[3]. Atkinson and Riani (2012)[4] focus on graphical methods of diagnostic regression analysis. Liu et al. (2017)[39] present an overview of interactive visual model validation. One of the most popular tools for verification are measures of the differences between the values predicted by a model and the observed values (Willmott et al., 1985)[65]. This includes Root Mean Square Error (RMSE)

[1] Residual of an observation is the difference between the observed value and the value predicted by a model.

and Mean Absolute Error (MAE) (Hastie et al., 2001)[33]. Such measures are used for well-researched and easily interpretable linear model as well as for complex models like a random forest (Ho, 1995)[35], an XGBoost (Chen and Guestrin, 2016)[12] or a neural network (Venables and Ripley, 2002)[59].

However, no matter which measure of model performance we use, it does not reflect all aspects of the model. As Breiman (2001)[10] indicates, the linear regression model validated only on the basis of R^2 may lead to many false conclusions. The best known example for this issue is the Anscombe Quartet (Anscombe, 1973)[2]. It contains four different data sets constructed to have nearly identical simple statistical properties like mean, variance, correlation, etc. These measures directly correspond to the coefficients of the linear models. Therefore, by fitting a linear regression to the Anscombe Quartet we obtain four almost identical models (see Figure 0.1). However, residuals of these models are very different. The Anscombe Quartet is used to highlight that the numerical measures should be supplemented by graphical data visualizations. In Chapter 4 we show more advanced example related to this issue.

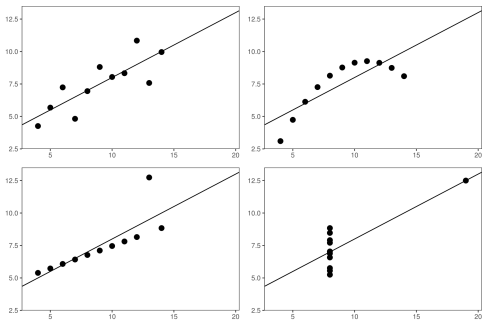


Figure 0.1: Anscombe Quartet data sets are identical when examined using simple summary statistics. The difference is noticeable after plotting the data.

The diagnostic analysis is well researched for linear and generalized linear models. This is typically done by extracting raw, studentized, deviance, or Pearson residuals and examining residual plots. Common problems with model fit and basic diagnostic methods are presented in Faraway (2002)[17] and Harrell (2006)[28] (see Figure 1.3). Model validation may involve both, checking the overall trend in residuals and looking at residual values of individual observations (Littell et al., 2007)[38]. Gałecki and Burzykowski (2013)[24] discussed methods based on residuals for individual observation and groups of observations.

Diagnostic methods are commonly used for linear regression (Faraway, 2004)[18]. Complex models are treated as if they were assumption free, that is why diagnostic of them is often ignored. Considering the above, there is a need for more extensive methods and dedicated software for model auditing. Many of diagnostic tools like plots and statistics developed for linear models are still useful for exploring any machine learning model. Applying the same tools to all models makes it easy to compare them. Yet another approach to auditability of models was proposed in Becker and Chambers (1988)[5].

The paper is organized as follows. Chapter 1 summarizes related work and state of the art. Chapter 2 contains an architecture of the **auditor** package. In Chapter 3 we provide the notation and discuss four main aspects of the model audit. In Section 3.1 we describe methods that help to assess the goodness-of-fit and may be useful to check whether the model does not miss relevant information. In Section 3.3 we present tools that help to assess which model has better performance. In Section 3.2 we describe graphical visualizations which allow checking the similarity of the models behavior. Section 3.4 contains methods for identification of influential observations. A use-case of model auditing is in Chapter 4 and conclusions are in Chapter 5.

1. Related work

In this chapter, we overview common methods and tools for auditing and examining the validity of the models. There are several attempts to validation. This includes diagnostic for predictor variables before and after model fit, methods dedicated to specific models, and model-agnostic approaches.

1.1. Data diagnostic before model fitting

The problem of data diagnostic is related to the Objective 2 presented in Introduction, that is, the identification of problems with observations. There are several tools that address this issue. We review the most popular of them.

- One of the tools that support the identification of errors in data is the **dataMaid** package (Petersen and Ekstrom, 2018)[43]. It creates a report that contains summaries and error checks for each variable in data. Package **lumberjack** (van der Loo, 2017)[56] provides row-wise analysis. It allows for monitoring changes in data as they get processed. The **validatetools** (de Jonge and van der Loo, 2018)[15] is a package for managing validation rules.

- The `datadist` function from **rms** package (Harrell Jr, 2018)[29] computes distributional summaries for predictor variables. The characteristics include the overall range and certain quantiles for continuous variables, and distinct values for discrete variables. It automates the process of fitting and validating several models, due to storing model summaries by `datadist` function.

- While above packages use pipeline approaches, there are also tools that focus on a specific step of data diagnostic. The package **corrgram** (Wright, 2018)[67] calculates a correlation of variables and displays corrgrams. Corrgrams (Friendly, 2002)[21] are visualizations of correlation matrices, which help to identify the relationship between variables.

Part 1

Data report overview

The dataset examined has the following dimensions:

Feature	Result
Number of observations	15
Number of variables	6

Checks performed

The following variable checks were performed, depending on the data type of each variable:

	character	factor	labelled	numeric	integer	logical	Date
Identify miscoded missing values	×	×	×	×	×		×
Identify prefixed and suffixed whitespace	×	×	×				
Identify levels with < 6 obs.	×	×	×				
Identify case issues	×	×	×				
Identify misclassified numeric or integer variables	×	×	×				
Identify outliers				×	×		×

Please note that all numerical values in the following have been rounded to 2 decimals.

Figure 1.1: Page from an exemplary report generated with **dataMaid** package. Here we have a table that summarizes checks for variables. This includes, among others, identification of missing values and outliers. In the further part of the report, there are summary tables and plots for each variable separately. The Figure comes from the report generated with the code from **dataMaid** documentation.

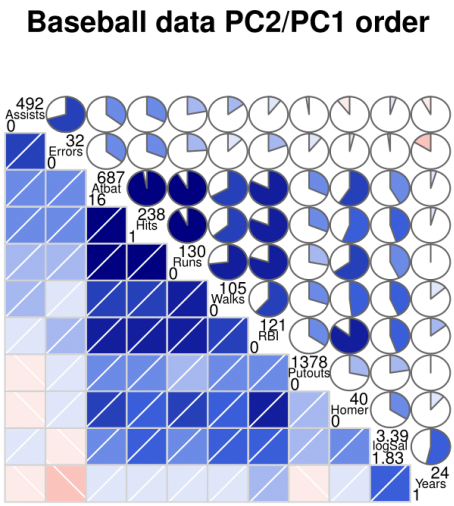


Figure 1.2: A graphical display of the correlation matrix generated by the `corrgram` function from the **corrgram** package. Lower triangle visualizes correlations by color and intensity of shading, upper triangle by pie charts. The plot is taken from Friendly (2002)[21] and generated with the code from vignette "Examples for the corrgram package".

1.2. Diagnostic methods for linear models

As linear models have a very simple structure and do not require high computational power, they have been and still are used very frequently. Therefore, there are many tools that validate different aspects of linear models. Below, we overview the most known of them that are implemented in R packages.

- The **stats** package (R Core Team, 2018)[45] provides basic diagnostic plots for linear models. Function `plot` generates six types of charts for `"lm"` and `"glm"` objects. A plot of residuals against fitted values, a scale-location plot of $\sqrt{|residuals|}$ against fitted values and a normal quantile-quantile plot. These visual validation tools may be referred to the Objective 3 of diagnostic, related to the examination of model by analyzing the distribution of residuals. The other three plots, that are: a plot of Cook's distances, a plot of residuals against leverages, and a plot of Cook's distances against $\frac{leverage}{1-leverage}$ may be referred to the identification of influential observations (Objective 1). Cook's distance and leverages are introduced in details in Section 3.18.

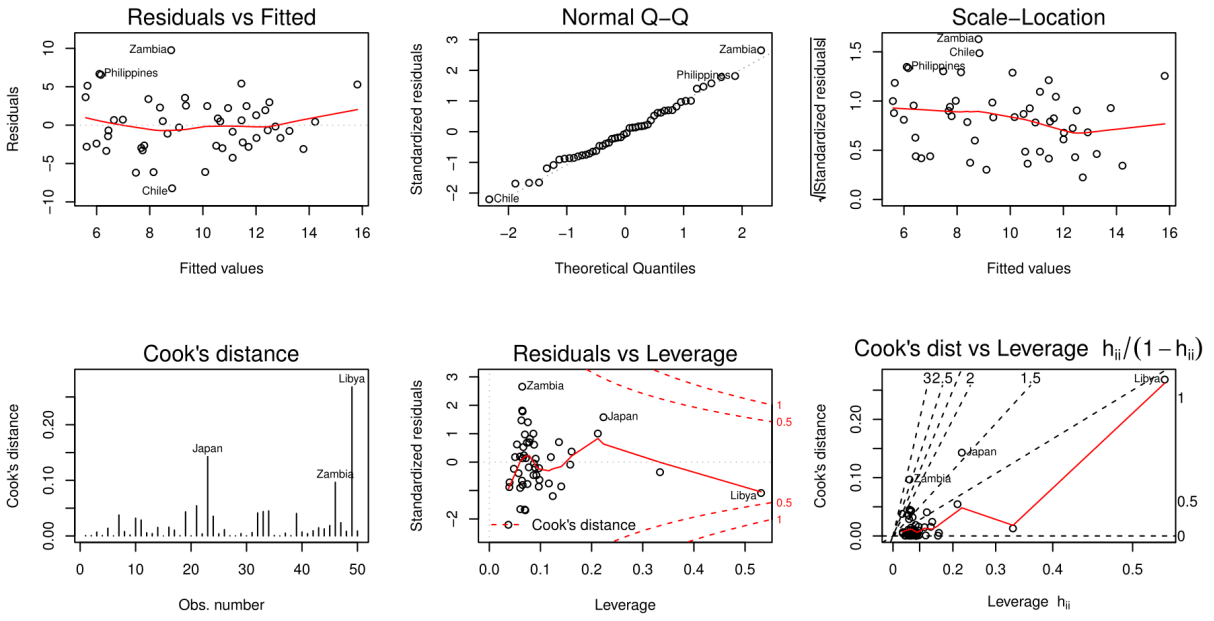


Figure 1.3: Six diagnostic plots for linear models created with generic `plot` function. Upper row contains plots related to the distribution of residuals, lower row is related to an influence of observations. Plots are generated with the code included in the documentation of function `plot.lm`.

- Package **car** (Fox and Weisberg, 2011)[20] extends the capabilities of **stats** by including more types of residuals, such as Pearson and deviance residuals. It is possible to plot against values of selected variables and grouping residuals by levels of factor variables. What is more, **car** provides more diagnostic plots. Among others, partial residual plot (`crPlot`), index plots of influence (`infIndexPlot`) and bubble plot of studentized residuals versus hat values (`influencePlot`). These plots allow to check both, the effect of observation and the distribution of residuals, which relates to the Objective 2 and the Objective 3 respectively.

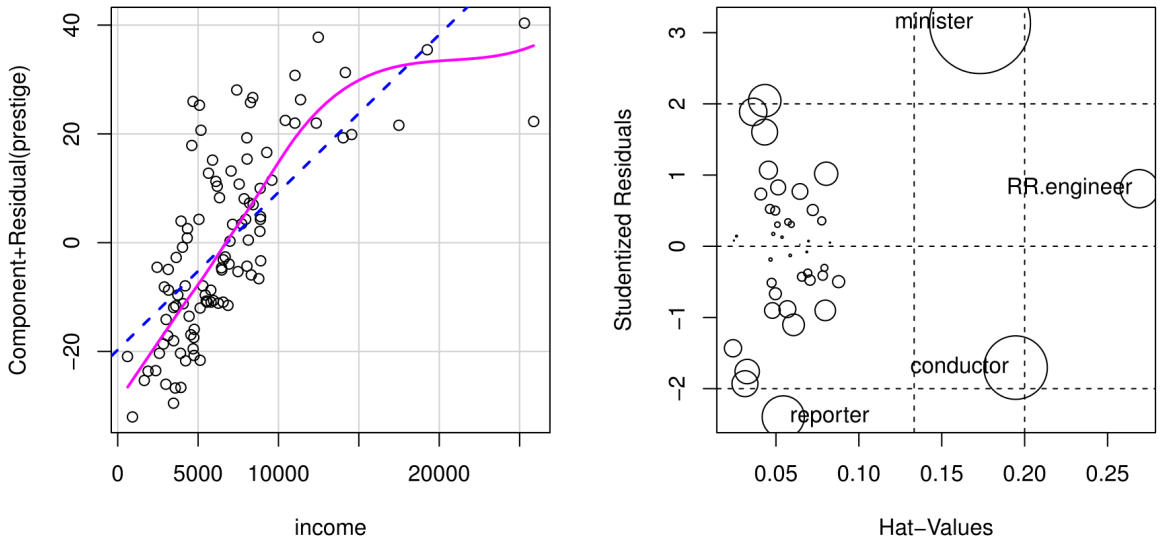


Figure 1.4: Two diagnostic plots generated with the **car** package. Left plot is a partial residual plot, right is a bubble plot of studentized residuals versus hat values. The areas of circles represent Cook's distances. Plots were generated with code from the documentation of **car** package.

- A linear regression model is still one of the most popular tools for data analysis due to its simple structure. Therefore, there is a rich variety of methods for checking its assumptions. For example, the normality of residual distribution and the homoscedasticity of the variance.

  The package **nortest** (Gross and Ligges, 2015)[27] provides five tests for normality: the Anderson-Darling (Anderson and Darling, 1952)[1], the Cramer-von Mises (Cramer, Von Mises, 1928, 1928)[14, 60], the Kolmogorov-Smirnov (Stephens, 1974)[50], the Pearson

chi-square (F.R.S., 1900)[22], and the Shapiro-Francia tests. The **lmtest** package (Zeileis and Hothorn, 2002)[69] also contains a collection of diagnostic tests: the Breusch-Pagan (Breusch and Pagan, 1979)[11], the Goldfield-Quandt (Goldfeld and Quandt, 1965)[25] and the Harrison-McCabe (Harrison and McCabe, 1979)[30] tests for heteroscedasticity and the Harvey-Collier (Harvey and Collier, 1977)[32], the Rainbow (Utts, 1982)[55], and the RESET (Ramsey, 1969)[46] tests for nonlinearity and misspecified functional form. A unified approach for examining, monitoring and dating structural changes in linear regression models is provided in **strucchange** package (Zeileis et al., 2002)[70]. This includes methods to fit, plot and test fluctuation processes and F-statistics. The **gvlma** implements the global procedure for testing the assumptions of the linear model, find more details in Peña and Slate (2006)[44].

The Box-Cox power transformation, introduced by Box and Cox (1964)[8], is a way to transform the data to follow a normal distribution. For simple linear regression, it is often used to satisfy the assumptions of the model. Package **MASS** (Venables and Ripley, 2002)[59] contains functions that compute and plot profile log-likelihoods for the parameter of the Box-Cox power transformation.

- The **broom** package (Robinson, 2018)[47] provides summaries for about 30 classes of models. It produces results, such as, coefficients and p-values for each variable, R^2, adjusted R^2, and residual standard error.

1.3. Other model-specific approaches

There are also several tools to generate validation plots for time series, principal component analysis, clustering, and others.

- Tang et al. (2016)[53] introduced the **ggfortify** interface for visualizing many popular statistical results. Plots are generated with **ggplot2** (Wickham, 2009)[63], which makes them easy to modify. With one function `autoplot` it is possible to generate validation plots for wide range of models. It works for, among others, `lm`, `glm`, `ts`, `glmnet`, and `survfit` objects.

  The **autoplotly** (Tang, 2018)[52] package is an extension of **ggfortify** and provides functionalities that produce plots generated by **plotly** (Sievert et al., 2017)[49]. This allows for both, modification and interaction with plots.

  However, **ggorftify** and **autoplotly** do not support some popular types of models. For

instance, random forests from **randomForest** (Liaw and Wiener, 2002)[37] and **ranger** (Wright and Ziegler, 2017)[68] packages.

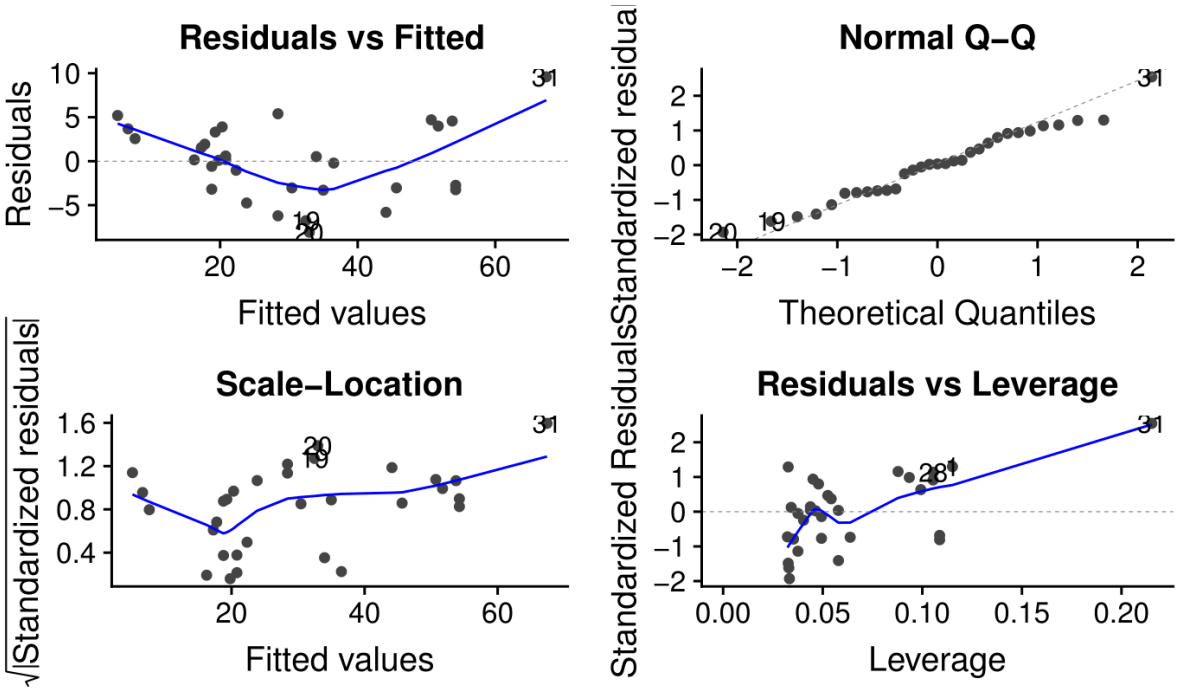


Figure 1.5: The diagnostic plots for linear model automaticaly generated with `autoplot` function from the **ggfortify** package. Plots correspond to the Objective 3 of diagnostic, that covers analysis of distribution of residuals. Plots were taken from the vignette "Introduction toggfortify package".

- The **hnp** package (Moral et al., 2017)[41] provides half-normal plots with simulated envelopes. These charts evaluate the goodness of fit of any generalized linear model and its extensions. It is a graphical method for comparing two probability distributions by plotting their quantiles against each other. The more detailed description is provided in the Section 3.1.7. There is a possibility to extend the `hnp` for new model classes. However, this package provides only one tool for model diagnostic. In addition, plots are not based on **ggplot2**, which makes it difficult to modify them.

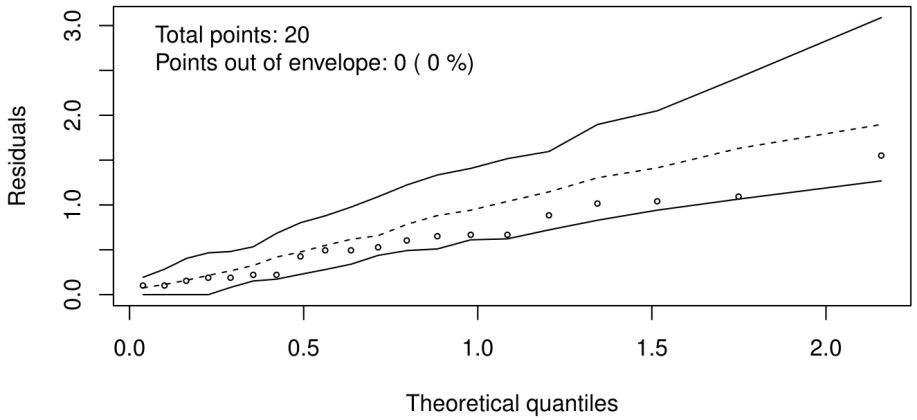


Figure 1.6: The half-normal plot generated with the `hnp` function from the **hnp** package. Here we have residuals and envelope that evaluate the goodness of fit.

1.4. Model-agnostic approach

Tools, presented above, target specific model classes. The model-agnostic approach allows to compare different models.

- The **DALEX** (Descriptive mAchine Learning EXplanations) (Biecek, 2018)[7] is a methodology for exploration of black-box models. Main functionalities focus on understanding or proving how the input variables impact on final predictions. There are also two simple diagnostic diagnostic: reversed empirical cumulative distribution function for absolute values of residuals and box plot of absolute values of residuals. As methods in the **DALEX** are model-agnostic, they allow to compare two or more models.

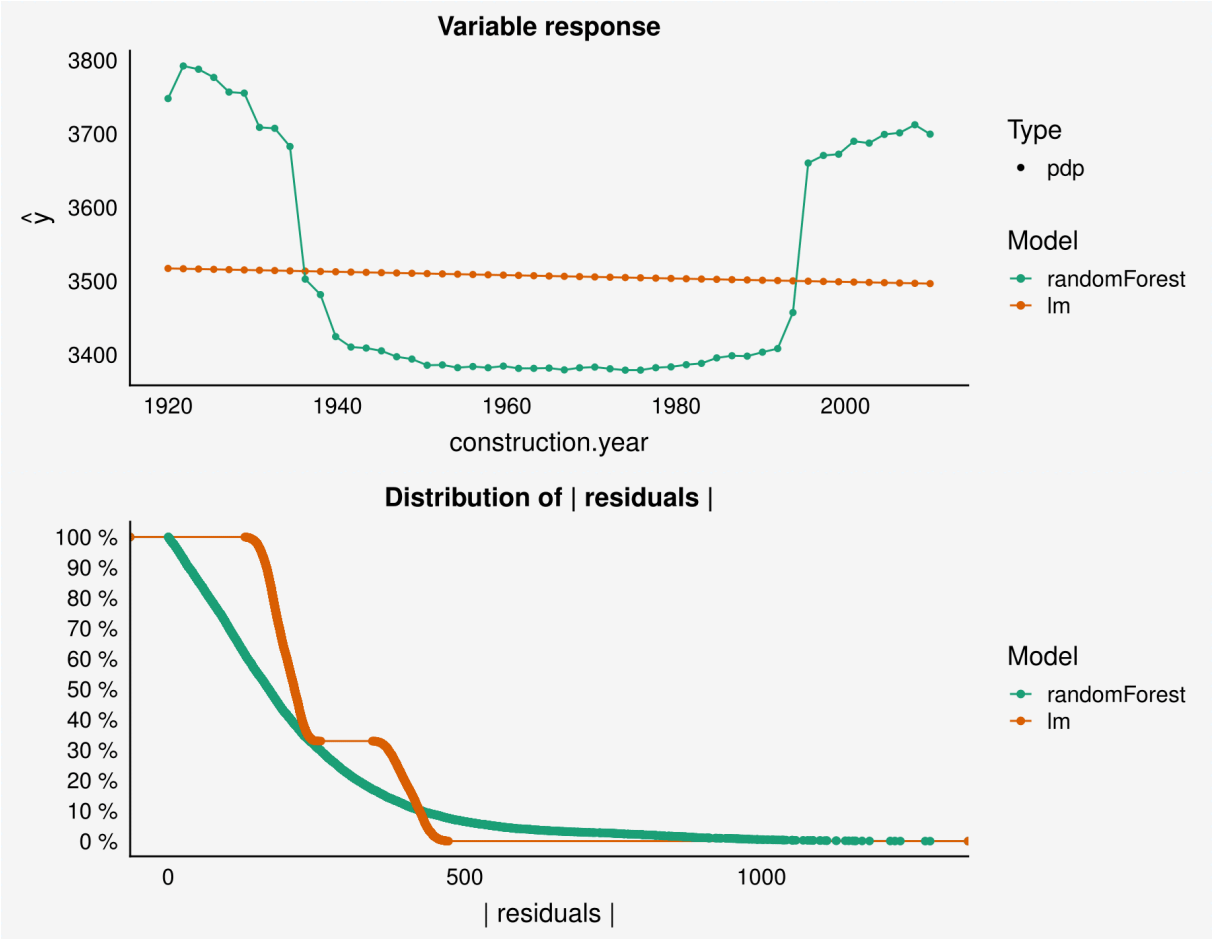


Figure 1.7: Model explanation plots generated with the **DALEX** package. The upper plot is a Partial Dependence Plot (Greenwell, 2017)[26]. The lower plot is an empirical cumulative distribution function of absolute values of residuals. Plots were taken from Biecek (2018)[7].

1.5. Model-agnostic audit

In this paper, we present the **auditor** package for R, which fills out the part of model-agnostic validation. As it expands methods used for linear regression, it may be used to verify any predictive model.

2. Package Architecture

The **auditor** package works for any predictive model which returns a numeric value. It offers a consistent grammar of model validation, which is an efficient and convenient way to generate plots and diagnostic scores. A diagnostic score is a number that evaluates one of the properties of a model. That might be, for example, an accuracy of model, an independence of residuals or an influence of observation.

Figure 2.1 presents the architecture of the package. The **auditor** provides 2 pipelines for model validation. First of them consists of two steps. Function `audit` wraps up the model with meta-data, then the result is passed to the `plot` or `score` function. Second pipeline includes an additional step, which consists of calling one of the functions that generate computations for plots and scores. These functions are: `modelResiduals`, `modelEvaluation`, `modelFit`, `modelPerformance`, and `observationInfluence`. Further, we call them computational functions. Results of these functions are tidy data frames (Wickham, 2014)[64].

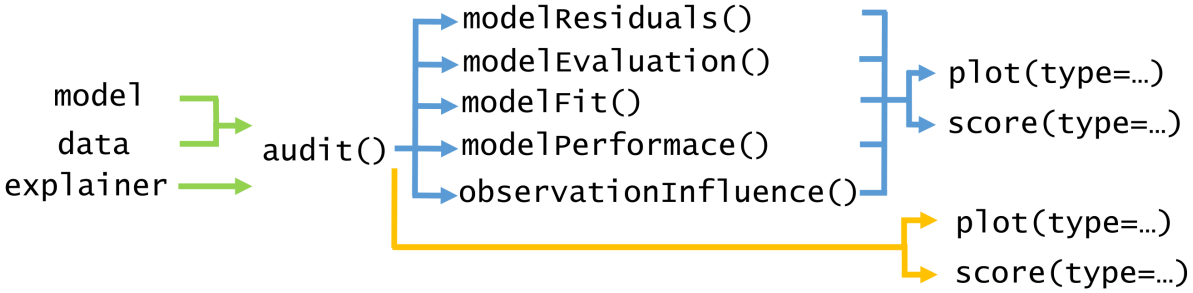


Figure 2.1: Architecture of the **auditor**. Blue color indicates the first pipeline, orange indicates the second. Function `audit` takes model and data or `"explainer"` object created with the **DALEX** package.

Both pipelines for model audit are compared below.

1. **model %>% audit() %>% computational function %>% plot(type=...)**
   We recommend this pipeline. Function `audit` wraps up a model with meta-data used for modeling and creates a `"modelAudit"` object. One of the computational functions takes `"modelAudit"` object and computes the results of validation. Then, output may be printed or passed to functions `score` and `plot` with defined type. We describe types of plots in the Chapter 3.

   This approach requires one additional function within the pipeline. However, once created output of the computational function contains all necessary calculations for related plots. Therefore, generating multiple plots is fast.

2. **model %>% audit() %>% plot(type=...)**
   This pipeline is shorter than the previous one. The only difference is that it does not include computational function. Calculations are carried out every time a generic `plot` function is called. Omitting one step might be convenient for ad-hoc model analyses.

Implemented types of plots are presented in Table 2.1. Scores are presented in Table 2.2. All plots are generated with **ggplot2**. This provides a convenient way to modify and combine plots.

Plot	Function	plot(type = ...)	Reg.	Class.
Autocorrelation Function	`modelResiduals`	`"ACF"`	+	+
Autocorrelation	`modelResiduals`	`"Autocorrelation"`	+	+
Cooks's Distances	`observationInfluence`	`"CooksDistance"`	+	+
Half-Normal	`modelFit`	`"HalfNormal"`	+	+
LIFT Chart	`modelEvaluation`	`"LIFT"`		+
Model Correlation	`modelResiduals`	`"ModelCorrelation"`	+	+
Model PCA	`modelResiduals`	`"ModelPCA"`	+	+
Model Ranking	`modelPerformance`	`"ModelRanking"`	+	+
Predicted Response	`modelResiduals`	`"ModelPerformance"`	+	+
REC Curve	`modelResiduals`	`"REC"`	+	+
Residuals	`modelResiduals`	`"Residual"`	+	+
Residual Boxplot	`modelResiduals`	`"ResidualBoxplot"`	+	+
Residual Density	`modelResiduals`	`"ResidualDensity"`	+	+
ROC Curve	`modelEvaluation`	`"ROC"`		+
RROC Curve	`modelResiduals`	`"RROC"`	+	+
Scale-Location	`modelResiduals`	`"ScaleLocation"`	+	+
Two-sided ECDF	`modelResiduals`	`"TwoSidedECDF"`	+	+

Table 2.1: Columns contain as follows: name of the plot, name of the computational function, value for `type` parameter of the function `plot`, indications whether the plot can be applied to regression and classification tasks.

Score	Function	score(type = ...)	Reg.	Class.
Cook's Distance	`observationInfluence`	`"CooksDistance"`	+	+
Durbin-Watson	`modelResiduals`	`"DW"`	+	+
Half-Normal	`modelFit`	`"HalfNormal"`	+	+
Mean Absolute Error	`modelResiduals`	`"MAE"`	+	+
Mean Squared Error	`modelResiduals`	`"MSE"`	+	+
Area Over the REC	`modelResiduals`	`"REC"`	+	+
Root Mean Squared Error	`modelResiduals`	`"RMSE"`	+	+
Area Under the ROC	`modelEvaluation`	`"ROC"`		+
Area Over the RROC	`modelResiduals`	`"RROC"`	+	+
Runs	`modelResiduals`	`"Runs"`	+	+
Peak	`modelResiduals`	`"Peak"`	+	+

Table 2.2: Columns contain as follows: name of a score, name of a computational function, value for `type` parameter of function the `score`, indications whether the score can be applied to regression and classification tasks.

3. Model audit

Diagnostic allows to evaluate different properties of a model. We divide them into four groups. Each of them is related to one of the following aspects and corresponding questions.

1. First aspect is assessing the goodness-of-fit and whether the model does not miss relevant information (Does the model fit data?). See section 3.1.

2. Second one is examining similarity of models (How similar models are?). See section 3.2.

3. Third aspect is an evaluation of model performance. (Which model has better performance?). See section 3.3.

4. Last one is an identification of influential observations. (Which observations have the most impact on a model?) See section 3.4.

These aspects are directly related to the diagnostic objectives described in the Introduction. First two of them are related to the examination of distribution of residuals, which was proposed as the Objective 3. The third aspect is an evaluation of a model performance (Objective 2). The last one refers to influential observations (Objective 3).

We introduce notation to follow throughout the paper.

Let us use the following notation: $x_i = (x_i^{(1)}, x_i^{(2)}, ..., x_i^{(p)}) \in \mathcal{X} \subset \mathcal{R}^p$ is a vector in space $\mathcal{X}$, $y_i \in \mathcal{R}$ is an observed response associated with x_i. A single observation we denote as a pair (y_i, x_i) and n is the number of observations.

Let denote a model as a function $f : \mathcal{X} \to \mathcal{R}$. Predictions of the model f for particular observation we denote as

$$f(x_i) = \hat{y}_i. \tag{3.1}$$

Row residual or, simply, residual is the difference between the observed value y_i and the predicted value $\hat{y}_i$. We denote residual of particular observation as

$$r_i = y_i - \hat{y}_i. \tag{3.2}$$

3.1. Aspect: Does the model fit data?

A good fitted model predicts response variable well. In this section, we present methods to assess the goodness-of-fit and check whether the model does not miss any relevant information.

To illustrate applications of the **auditor** we use an artificial data set `apartments` available in the **DALEX** package. First, we fit two models: simple linear regression and random forest.

```
library("auditor")
library("DALEX")
library("randomForest")


lm_model <- lm(m2.price ~ ., data = apartments)
set.seed(59)
rf_model <- randomForest(m2.price ~ ., data = apartments)
```

Next step is creating `"modelAudit"` objects related to this two models.

```
lm_audit <- audit(lm_model, label = "lm",
                  data = apartmentsTest, y = apartmentsTest$m2.price)
rf_audit <- audit(rf_model, label = "rf",
                  data = apartmentsTest, y = apartmentsTest$m2.price)
```

Below, we create objects of class `"modelResidual"`. They are required for generating plots. Parameter `variable` determines the order of residuals in the plot. For `variable = "Fitted values"` residuals are sorted by values of predicted responses. Entering a name of a variable (`"m2.price"`) implies that residuals are in order of this variable.

```
lm_res_fitted <- modelResiduals(lm_audit, variable = "Fitted values")
rf_res_fitted <- modelResiduals(rf_audit, variable = "Fitted values")


lm_res_observed <- modelResiduals(lm_audit, variable = "m2.price")
rf_res_observed <- modelResiduals(rf_audit, variable = "m2.price")
```

3.1.1. Residuals Plot

Residuals vs Fitted Values is the most frequently used plot in model validation. It is a scatter plot of residuals r_i on the y axis against fitted values $\hat{y}_i$ on the x axis. Example plot is presented in Figure 3.1. On alterations of this plot on the x-axis are values of the variable.

This plot is used to detect dependence of errors, unequal error variances, and outliers. For appropriate model, residuals should not show any functional dependency. Expected mean value should be equal 0, regardless of $\hat{y}$ values. Structured arrangement of points suggests a problem with the model. It is worth looking at the observations that clearly differ from the others. If points on the plot are not randomly dispersed around the horizontal axis, it may be presumed that model is not appropriate for the data.

This plot is generated by the `plot` function with parameter `type = "Residual"` or by `plotResidual` function. Other variants of the Residual Plot may be obtained by parameter `variable`.

```
plot(rf_res_fitted, lm_res_fitted, type = "Residual")
```

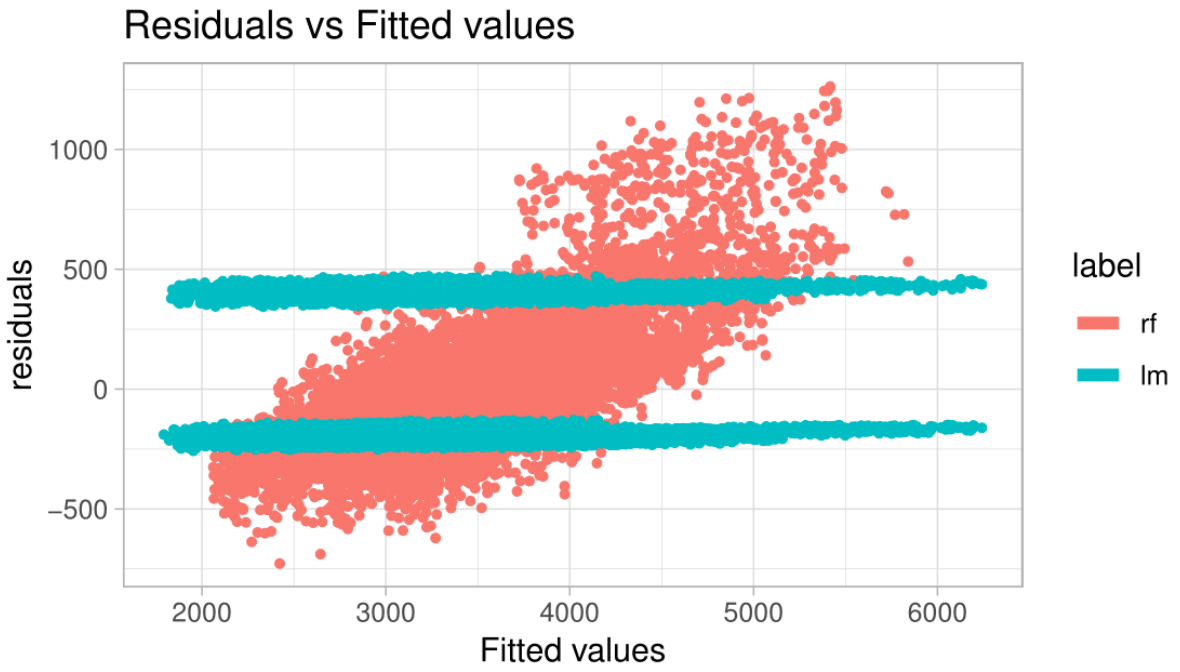


Figure 3.1: Residual Plot. The patterns for both models are non-random. The values of residuals of random forest (red) increase with the increase of the fitted values. Residuals of linear model (blue) are divided into two separate groups. That suggests problems with structures of the models.

3.1.2. Residual Boxplot Plot

Residual boxplot shows the distribution of the absolute values of residuals r_i. They may be used for analysis and comparison of residuals. Example plots are presented in Figure 3.2.

Boxplots (Tukey, 1977)[54] usually consists of five components. Box corresponds to the first quartile, median, and third quartile. The whiskers extend to the smallest and largest values, no further than 1.5 of Interquartile Range (IQR) from the first and third quartile, respectively. Residual boxplots consist of the sixth component. Red dot stands for Root Mean Square Error (RMSE).

For the appropriate model, most of the residuals should lay near zero. A large spread of values indicates problems with a model.

This plot is generated by `plotResidualBoxplot` or by `plot` function with parameter `type = 'ResidualBoxplot'` function.

```
plot(lm_res_fitted, rf_res_fitted, type = "ResidualBoxplot")
```

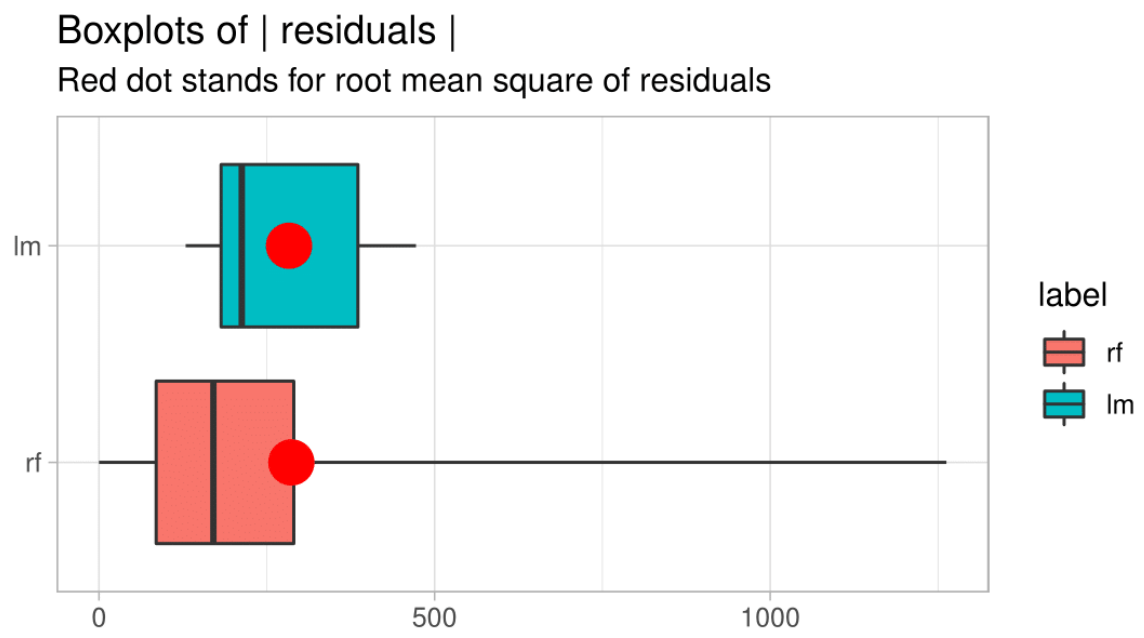


Figure 3.2: Boxplots of absolute values of residuals. Dots are in the similar places, so RMSE for both models is almost identical. However, the distribution of residuals of this two models is different. For the linear model (blue), most of the residuals are around the average. For the random forest (red), most residuals are small. But there is also a fraction of large residuals.

3.1.3. Autocorrelation Plot

Autocorrelation Plot is a tool for checking whether there is a relationship between residuals on the lag 1. Example chart is presented in Figure 3.3. It is a scatter plot of i-th vs $i+1$-th residuals, ordered by fitted values or by values of one of the variables.

We expect that model residuals are independent. Therefore, points should be randomly dispersed. The structured arrangement of residuals suggests a problem with the model.

This plot is generated by `plot` function with parameter `type = "Autocorrelation"` or by function `plotAutocorrelation`.

```
plot(lm_res_fitted, rf_res_fitted,  type = "Autocorrelation")
```

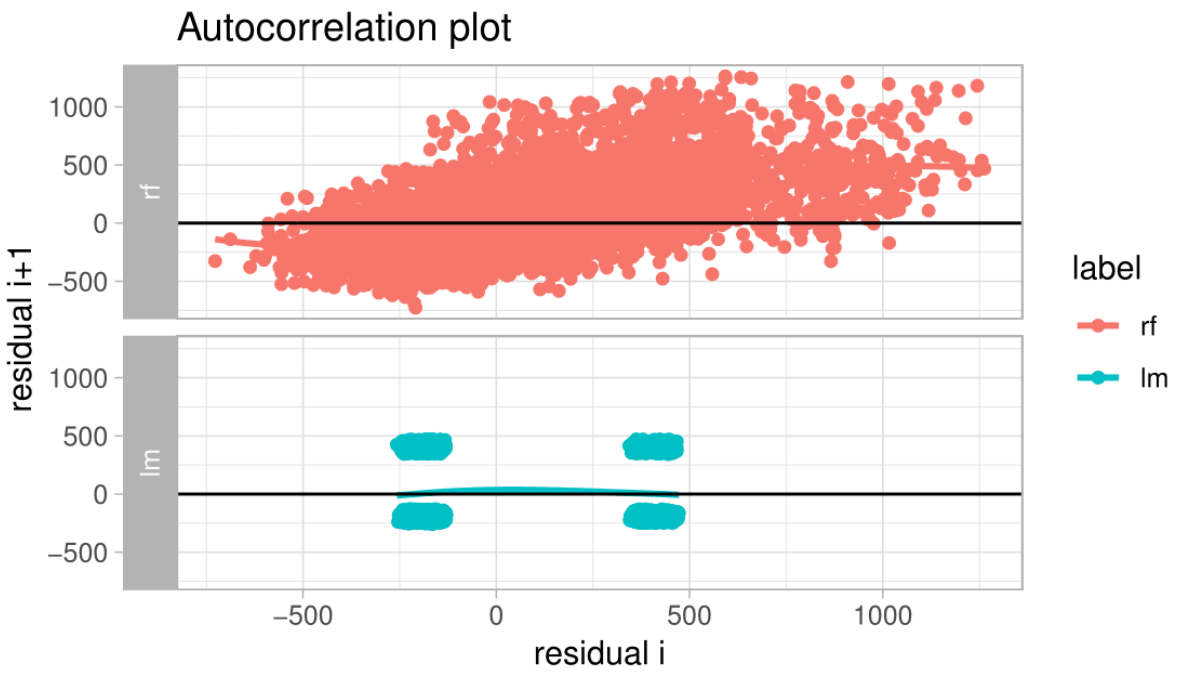


Figure 3.3: Autocorrelation plot of i-th residual vs $i+1$-th residual. The residuals of the linear model (blue) are divided into 4 groups, which indicates that they are strongly non-random. Residuals of random forest (red) show an increasing trend, which also suggests a problem with the model structure.

As results of plots may be ambiguous, there are two scores implemented in the **auditor**: Durbin-Watson (DW) and Runs. They help to asses the autocorrelation of residuals.

The Durbin–Watson (DW) statistic is used to detect the autocorrelation in the residuals at lag = 1. It is introduced by Durbin and Watson (1971)[16].

Definition 3.1. The Durbin-Watson test statistic is given by

$$DW = \frac{\sum_{i=2}^{n}(r_i - r_{i-1})^2}{\sum_{i=1}^{n} r_i^2}. \tag{3.3}$$

It detects only linear autocorrelation and only first order effects.

We treat Durbin-Watson statistic as a score. Its values lie between 0 and 4. DW around 2 means no autocorrelation. Small value may be an evidence for positive correlation and large value for negative correlation.

Runs score is the second tool for checking the independency of residuals.

A run is a series of increasing or decreasing values. The U is an observed number of runs in residuals ordered by fitted values or values of one of the variables. The $\bar{U}$ is an expected number of runs and s_U is the standard deviation of the number of runs. Runs test statistic was defined by Wald and Wolfowitz (1940)[62].

Definition 3.2. The Runs test statistic is given by

$$Z = \frac{U - \bar{U}}{s_U}. \tag{3.4}$$

For random sequence, the distribution of Z follows the $\mathcal{N}(\mu, \sigma^2)$.

We consider values of Runs statistic as scores.

Both presented scores may be obtained by `score` function with `type = "DW"` or `type = "Runs"`.

```
score(lm_res_fitted,  type = "DW")
score(rf_res_fitted,  type = "DW")
score(lm_res_fitted,  type = "Runs")
score(rf_res_fitted,  type = "Runs")
```

3.1.4. Autocorrelation Function Plot

Autocorrelation Function (ACF) is a tool for finding patterns in the data. In particular, the correlation between observations. Figure 3.4 shows an example of the Autocorrelation Function Plot of residuals.

There are values of ACF on the y axis against lags on the x axis. As the value of autocorrelation of lag 0 always equals 1, it is skipped. Blue horizontal lines are confidence intervals.

We assume that $(X_1, X_2, ...X_n) \in \mathbb{R}^n$ are observations of a time series X_t with mean μ. Covariance function $\gamma_t = Cov(X_{t+\tau}, X_\tau)$ and correlation function $\rho_t = Cov(X_{t+\tau}, X_\tau)$ do not depend on τ. The following estimators come from Venables and Ripley (2013)[58].

Definition 3.3. The sample autocovariance function is given by

$$\hat{\gamma}(t) = \frac{1}{n} \sum_{s=max(1,-ht)}^{min(n-t,n)} (X_{s+t} - \bar{S})(X_s - \bar{X}) \tag{3.5}$$

and the sample autocorrelation function is given by

$$\hat{\rho}(t) = \frac{\hat{\gamma}(t)}{\hat{\gamma}(0)}. \tag{3.6}$$

Now, let us consider residuals r_i as elements of the time series X_t. Autocorrelation Function Plot visualizes the sequence of $(\hat{\rho}(t))$.

This plot is generated by `plot` function with parameter `type = "ACF"` or by function `plotACF`.

```
plot(lm_res_fitted, rf_res_fitted,  type = "ACF")
```

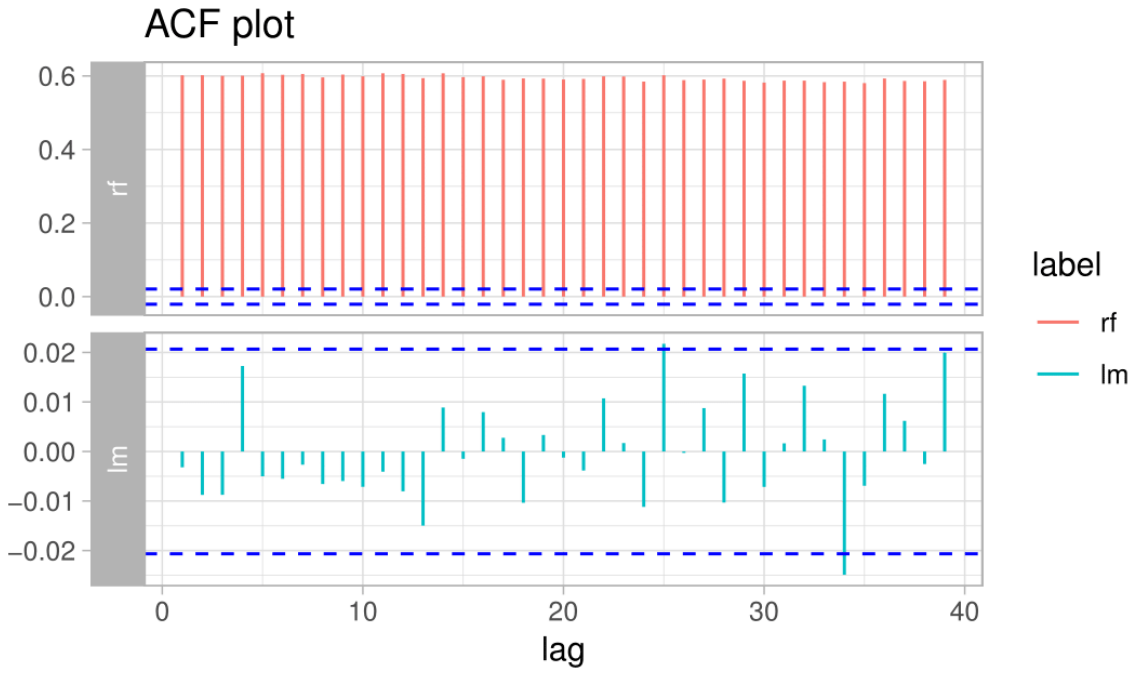


Figure 3.4: Autocorrelation Function plot. All values for random forest (red) go out of the confidence intervals. That suggests a problem with the correlation of residuals. Residuals of the linear model (blue) do not indicate serious correlation problems.

3.1.5. Scale-Location Plot

Scale Location Plot is used to visualize the variance of the residuals. Example chart is presented in Figure 3.5. This plot is similar to the Residuals Plot, but it uses the square root of the standardized residuals instead of raw residuals. Let $\bar{r}$ be mean of residuals,

$$\bar{r} = \frac{1}{n}\sum_{i=1}^{n} r_i. \tag{3.7}$$

Definition 3.4. The standardized residual r_i^{std} is the i-th residual divided by the residual standard deviation:

$$r_i^{std} = \frac{r_i}{\frac{1}{n-1}\sum_{i=1}^{n}(r_i - \bar{r})^2}. \tag{3.8}$$

On the x axis, there are responses fitted by a model. On the y axis, there are square roots of the absolute value of the standardized residuals. Blue line is an estimate of the conditional mean function. Black dots represent peaks explained below. An alternative version of this plot contains values of one of the variables on the x axis.

The presence of any trend suggests that the variance depends on fitted values, which is against the assumption of homoscedasticity. Not every model has an explicit assumption of homogeneous variance, however, the heteroscedasticity may indicates potential problems with the goodness-of-fit.

This plot is generated by `plot` function with parameter `type = "ScaleLocation"` or by function `plotScaleLocation`. Variable on the x axis may be specified by `variable` parameter.

```
plot(rf_res_fitted, lm_res_fitted, type = "ScaleLocation")
```

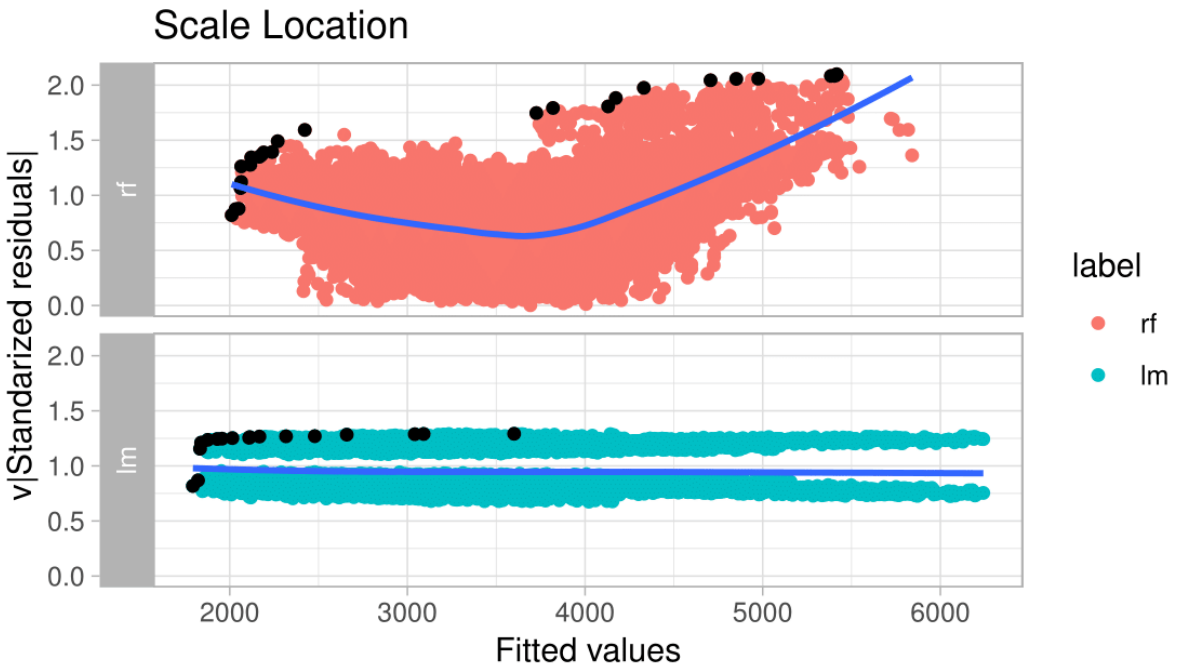


Figure 3.5: Scale Location plot. The trend in residuals of random forest (red) suggests that variance of residuals changes with the change in fitted values. The linear model (blue) seems to have no problems with variance. However, residues formed into separate groups suggest a problem with model structure.

A tool for assesing the homoscedasticity of variance is **Peak** test introduced by Goldfeld and Quandt (1965)[25]. A peak is an occurrence of observation i such that $|r_i| \geqslant |r_j|$ for all $i > j$.

Definition 3.5. We define Peak score as

$$P = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{\forall_{j<i} : |r_j| \leqslant |r_i|\}. \tag{3.9}$$

Let us note that $P \in (0, 1]$. If the residuals r_i are heteroscedastic, the Peak score is close to 1. Scores may be obtained by `score` function with `type = "Peak"` or `scorePeak` function.

```
score(lm_res_fitted,  type = "Peak")
score(rf_res_fitted,  type = "Peak")
```

3.1.6. Residual Density Plot

Residual Density plot detects the incorrect behavior of residuals. Example is presented in Figure 3.10. On the plot, there are estimated densities of residuals. Their values are displayed as marks along the x axis.

For some models, the expected shape of density derives from the model assumptions. For example, simple linear model residuals should be normally distributed. However, even if the model does not have an assumption about the distribution of residuals, such a plot may be informative. If most of the residuals are not concentrated around zero, it is likely that the model predictions are biased.

This plot is generated by `plotResidualDensity` function or by `plot` function with parameter `type = "ResidualDensity"`.

```
plot(rf_res_observed, lm_res_observed, type = "ResidualDensity")
```

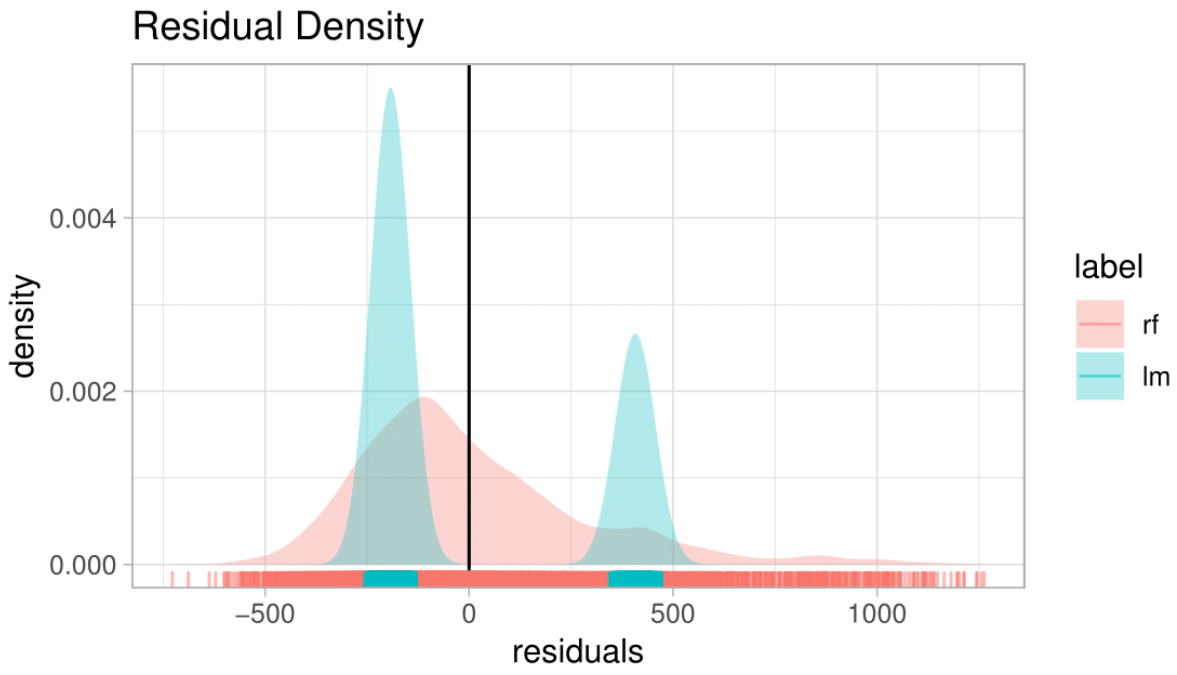


Figure 3.6: Residual Density Plot. The density of residuals for the linear model (blue) forms two peaks. There are no residuals with values around zero. Residuals do not follow the normal distribution, which is one of the assumptions of the simple linear regression. There is an asymmetry of residuals generated by random forest (red).

3.1.7. Half-Normal Plot

The Half-Normal Plot is one of the tools designed to evaluate the goodness-of-fit of a statistical model. It is a graphical method for comparing two probability distributions by plotting their quantiles against each other.

Points correspond to ordered absolute values of model diagnostic (i.e. standardized residuals) plotted against theoretical order statistics from a half-normal distribution. Lines mark the simulated envelope. The simulation process consists of simulating response variables using the model matrix, error distribution and parameters of an original model. Then the same model is fitted to data with new simulated response variable. The envelopes are formed on the basis of the 2.5 and 97.5 percentiles of the extracted simulated residuals at each value of the expected order statistic. Further details on the half-normal plots with simulated envelopes may be found in Moral et al. (2017)[41].

If residuals come from the normal distribution, they are close to a straight line. However, even if there is no certain assumption of a specific distribution, points still show a certain trend. Simulated envelopes help to verify the correctness of this trend. For a good-fitted model, diagnostic values should lay within the envelope.

This plot is generated by `plot` function with parameter `type = "HalfNormal"` or by function `plotHalfNormal`. The parameter `quant.scale = TRUE` transforms y axis into quantile scale.

```
lm_mf <- modelFit(lm_audit)
plot(lm_audit, type = "HalfNormal")
```

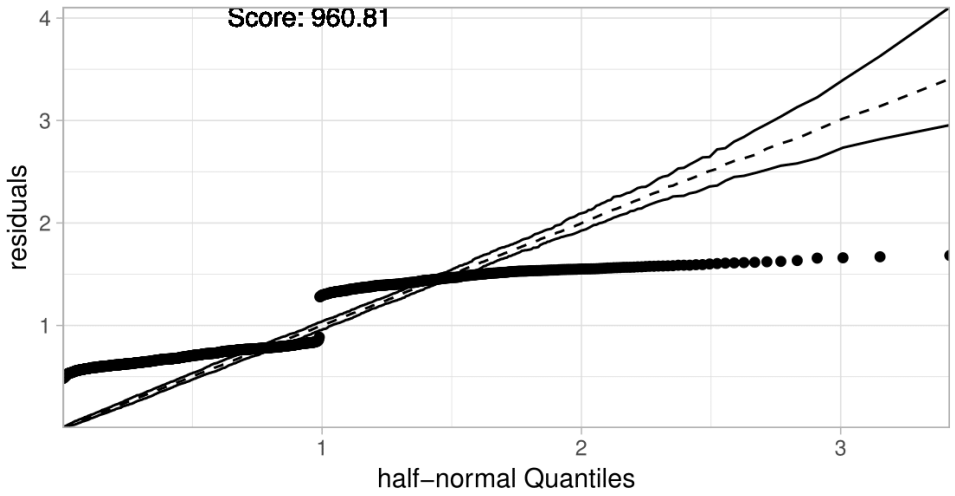


Figure 3.7: Half-Normal plot with simulated envelope. Most of the residuals lie outside the envelope. They do not behave as we would expect from the simulations. That suggests a poor model fit.

A useful tool to compare goodness-of-fit of two models is a Half-Normal Score.

Let us consider n observations and m simulations for each observation. We use the following notation: r_i is a residual of i-th observation, $r_i^{sim_j}$ is the residual of j-th simulation for i-th observation.

Definition 3.6. The score of i-th observation is given by

$$S_i = \sum_{j=1}^{m} \mathbb{1}\{r_i^{sim_j} \geqslant r_i\}. \tag{3.10}$$

S_i is a number of simulated residuals for observation i that are greater or equal than r_i.

Value of S_i close to 0 or m means that the original residual stands out from the simulated ones. The closer it is to $\frac{m}{2}$, the less it deviates from the simulated results.

Definition 3.7. We define Half-Normal score as

$$HN = \sum_{i=1}^{n} |R_i - \frac{m}{2}|. \tag{3.11}$$

HN is the sum of the deviations of S_i from $\frac{m}{2}$.

Let us note that $HN \in [0, \frac{nm}{2}]$ and lower value of score means better model fit.

HalfNormal score is calculated by `score` function with parameter `type = "HalfNormal"` or by function `soreHalfNormal`. Scores are calculated on the basis of simulated data, so they may differ between function calls.

```
scoreHalfNormal(lm_mf)
```

3.2. Aspect: How similar models are?

In this section, we present methods for assessing the similarity of models, understood as the closeness of the residuals structures. The following methods may be used to identify such a similarity or its absence.

3.2.1. Model PCA Plot

Model PCA plot can be used to assess the similarity of the models in terms of residuals. Example plot is presented in Figure 3.8.

The idea of PCA is reducing the dimension of the data set matrix by creating a set of linearly uncorrelated variables called principal components. At the same time, keeping as much variance as possible Jolliffe (1986)[36].

Model PCA plot is a biplot introduced by GABRIEL (1971)[23]. On axis of the plot, there are first two principal components. Grey dots represent observations. Arrows are pointing in the direction of the models projected into a two-dimensional space. The interposition of arrows provides information about the similarity of models in terms of residuals. If they are close to each other, it indicates similar residuals structures.

This plot is generated by `plot` function with parameter `type = "ModelPCA"` or by function `plotPCA`.

```
rf_mp <- modelPerformance(rf_audit)
lm_mp <- modelPerformance(lm_audit)
plot(rf_mp, lm_mp, type = "ModelPCA")
```

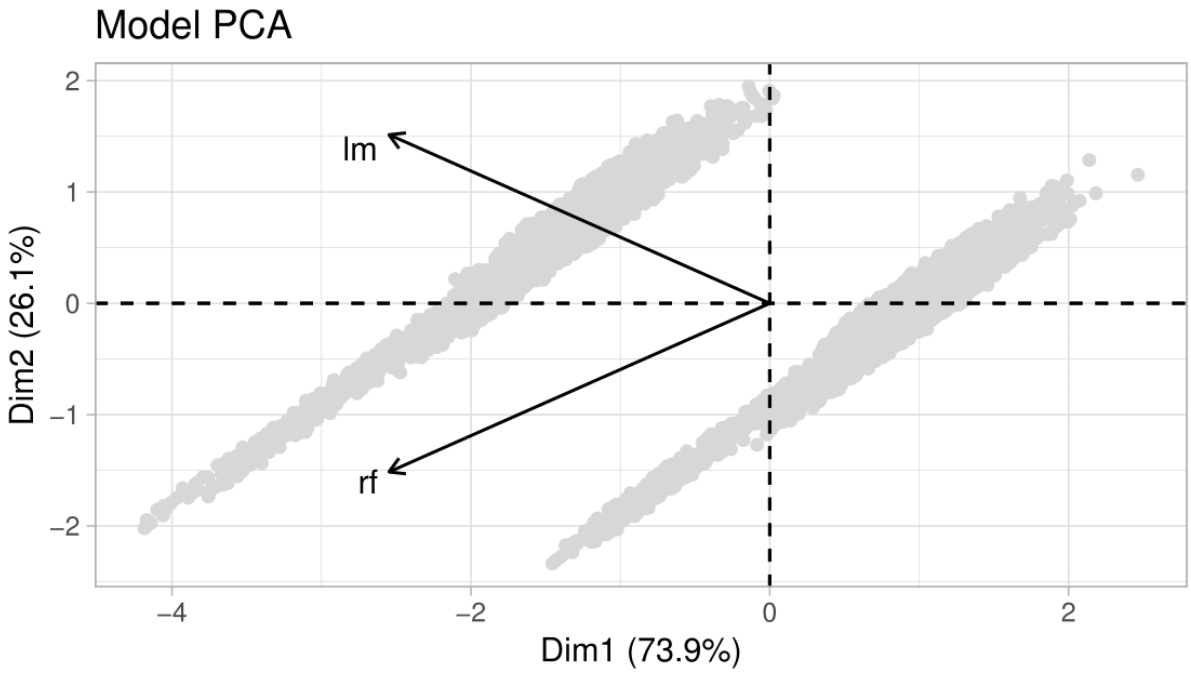


Figure 3.8: Model PCA plot. The diagram shows the arrangement of the residuals in two groups. The arrow corresponding to the linear model is almost perpendicular to the groups direction. Therefore, such a division can also be seen in the structure of the residuals of this model. The arrow corresponding to the random forest is almost parallel to the direction of the residuals. Probably the division into groups does not appear in the residuals of this model.

3.2.2. Model Correlation Plot

Model Correlation plot, presented in Figure 3.9, is a pairs plot of observed response and fitted values of different models. Pairs plot first appeared in Hartigan (1975)[31]. In the diagonal, there are estimated densities, in the upper triangle, the correlations between the different models and between models and observed response. In the lower triangle, there are scatter plots of these values.

This plot is generated by `plot` function with parameter `type = "ModelCorrelation"` or by function `plotCorrelation`. Taking the parameter `values = "Residuals"` residuals will be plotted instead of fitted values.

```
plot(rf_mp, lm_mp, type = "ModelCorrelation")
```

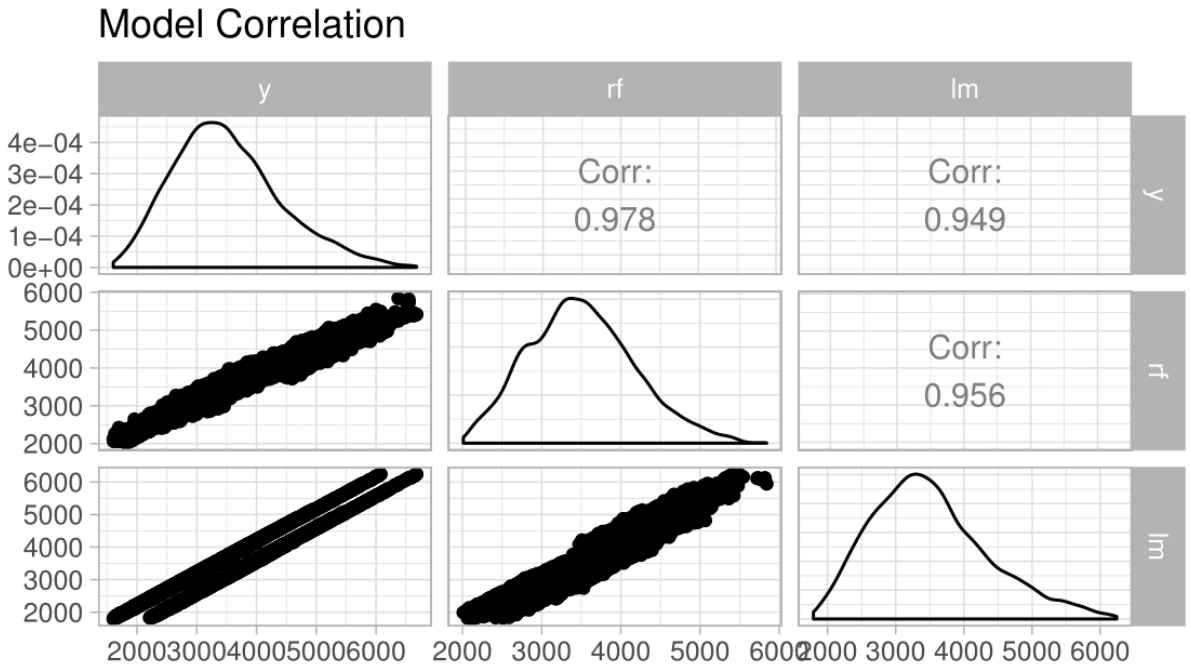


Figure 3.9: Model Correlation plot. The values predicted by random forest and linear model are correlated with each other and with the values observed. The plot does not show any serious problems.

3.3. Aspect: Which model has better performance?

We consider the model performance more widely than the value of one measure. The values and structures of the residuals also help to assess the quality of the model. In this section, we present visual methods of evaluating models performance.

3.3.1. Predicted Response Plot

Predicted Response Plot gives an additional information about model performance. It is a scatter plot of predicted values $\hat{y}_i$ on the y axis against observed response y_i on the x axis. Example plot is presented in Figure 3.10. On alterations of this plot on the x-axis are values of one the model variables. Ideally, all points are close to a diagonal line. For the appropriate model, points should not show any functional dependency.

This plot is generated by `plot` function with parameter `type = "Prediction"` or by `plotPrediction` function. Other variable on the x axis may be specified by `variable` parameter.

```
plot(rf_res_observed, lm_res_observed, type = "Prediction")
```

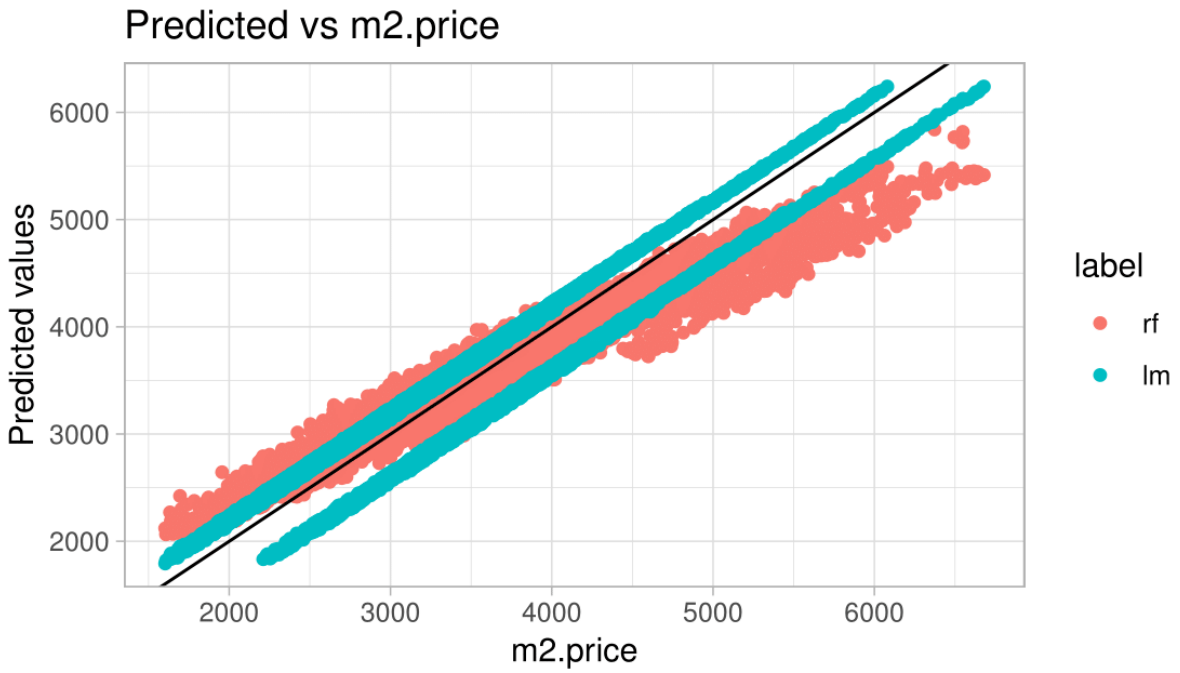


Figure 3.10: redicted Response Plot. The patterns for both models are non-random around the diagonal line. The points corresponding to a random forest (red) show the tendency to underprediction for large values of observed response and over-prediction for small values. Points for linear model (blue) are arranged into two separate groups. That suggests problems with the structures of the models.

3.3.2. Plots: ROC Curve and LIFT Chart

In this section, we present methods for assessing model performance for classification problem. Most common methods for regression models were inspired by tools for classification.

Let us consider a binary classification problem, in which the responses are labeled either as positive (P) or negative (N). True Positive (TP) is a case that the prediction is P and the actual value is also P. If the prediction is P and the actual value is N then it is a False Positive (FP).

First, we fit two binary classification models on the `PimaIndianDiabetes` data set from the `mlbench` package.

```
library("mlbench")
data("PimaIndiansDiabetes")
pima <- PimaIndiansDiabetes
pima$diabetes <- ifelse(pima$diabetes == "pos", 1, 0)

model_glm_class <- glm(diabetes~., data = pima, family = binomial)
library("e1071")
model_svm_class <- svm(diabetes~., data = pima, probability = TRUE,
                       type = "C-classification")
```

Next, we create corresponding `"modelAudit"` objects. We use created objects to generate example plots in this section.

```
au_glm_class <- audit(model_glm_class, data = pima, y = pima$diabetes)

custom_predict <- function(model, data){
  pred <- predict(model_svm_class, pima, probability=TRUE)
  attr(pred, "probabilities")[,1]
}

au_svm_class <- audit(model_svm_class, data = pima, y = pima$diabetes,
                      label = "svm", predict.function = custom_predict)
```

Receiver Operating Characteristic (ROC) curve is a way of visualising a classifier's performance (Swets, 1988)[51]. It answers the question of how well the model discriminates between the two classes. The boundary between classes is determined by a threshold value. ROC illustrates the performance of a classification model at various threshold settings.

Let $t \in [0, 1]$ be a threshold. We introduce parametric definition of ROC curve.

Definition 3.8. The ROC curve is a plot of the True Positive Rate (TPR) against the False Positive Rate (FPR) on a threshold t.

$$y = TPR(t) = \frac{TP(t)}{\mathrm{P(t)}} \tag{3.12}$$

and

$$x = FPR(t) = \frac{FP(t)}{N(t)}. \tag{3.13}$$

Each point on the ROC curve represents values of TPR and FPR of different thresholds.

The diagonal line y = x corresponds to a classifier that randomly guess the positive class half the time. Any model that appears in the lower right part of plot performs worse than random guessing. The closer the curve is to the the left border and top border of plot, the more accurate the classifier is.

This plot is generated by `plot` function with parameter `type = "ROC"` or by function `plotROC`.

```
plot(au_glm_class, au_svm_class, type = "ROC")
```

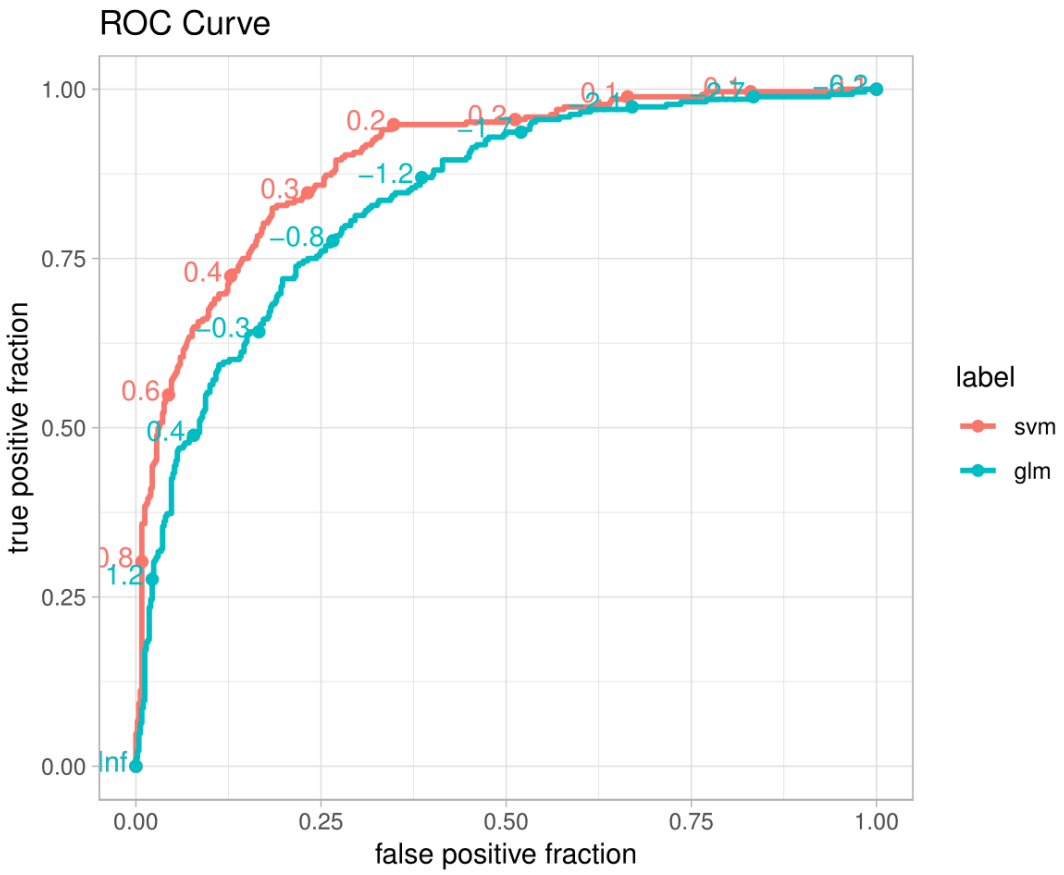


Figure 3.11: Receiver Operating Characteristic curve. The svm (red) curve is mostly above glm (blue). In the remaining part they overlap. This indicates that the support vector machines performs better than linear model.

ROC curves usually intersect. Therefore, it is not possible to choose a better model explicitly. A good tool to help assess the quality of a classifier is the Area Under the Curve (AUC) (Bradley, 1997)[9]. The AUC is a part of the area of the unit square. Therefore, its value is always be between 0 and 1. The AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (Fawcett, 2006)[19].

AUC may be obtained by `score` function with `type = "ROC"` or `scoreROC` function.

```
score(au_glm_class,  type = "ROC")
score(au_svm_class,  type = "ROC")
```

LIFT charts (Witten et al., 2011)[66] also evaluate performance of classification models.

Vuk and Curk (2006)[61] introduced parametric definition of LIFT chart. Let $t \in [0, 1]$ be a threshold.

Definition 3.9. The LIFT chart is Rate of Positive Prediction (RPP) plotted against True Positive (TP) on a threshold t.

$$y = TP(t) \tag{3.14}$$

and

$$x = RPP(t) = \frac{TP(t) + FP(t)}{P + N}. \tag{3.15}$$

Each point on the LIFT chart represents values of TP and RPP of different thresholds.

As for ROC curve, LIFT illustrates varying the model performance for different thresholds. LIFT chart shows how good the classifier distinguishes between two classes.

A random and ideal models are represented by black and orange curves, respectively. The closer the LIFT get to the orange curve, the better a model is.

This plot is generated by `plot` function with parameter `type = "LIFT"` or by function `plotLIFT`.

```
plot(au_glm_class, au_svm_class, type = "LIFT")
```

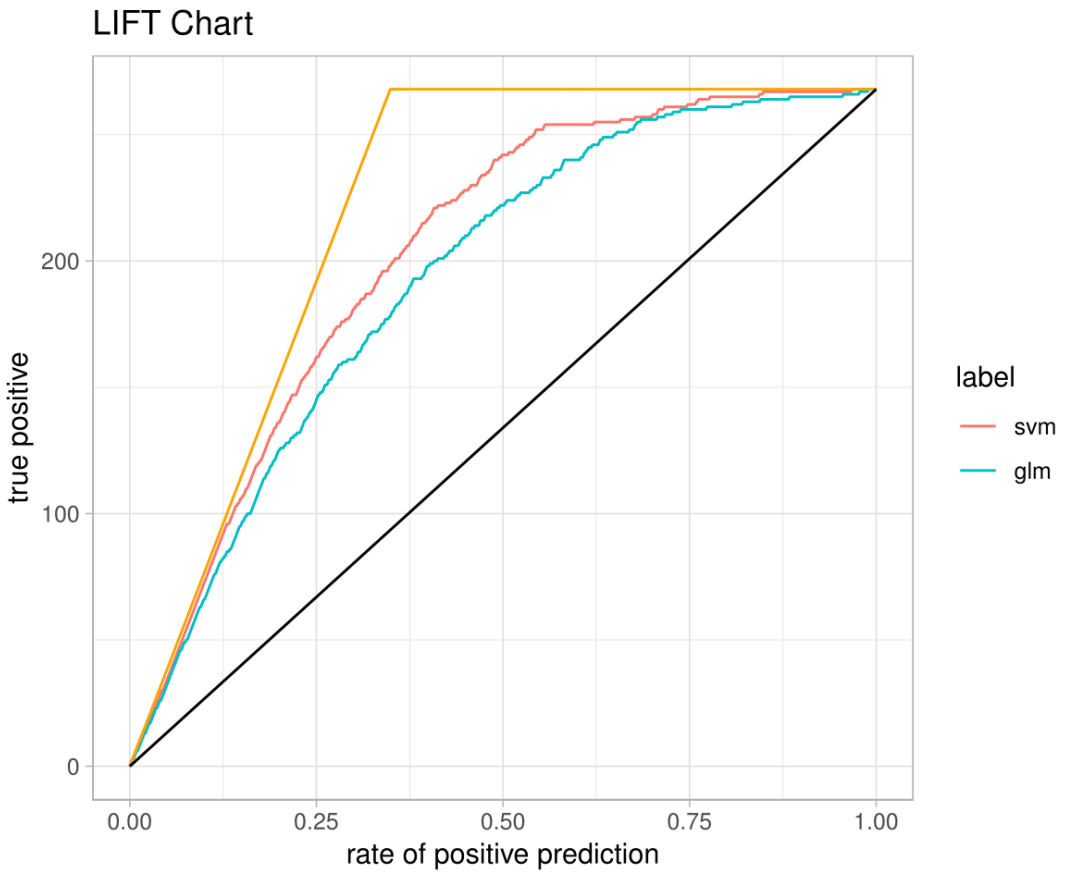


Figure 3.12: LIFT Chart.

3.3.3. REC Curve Plot

Regression Error Characteristic (REC) curve is a generalization of Receiver Operating Characteristic (ROC) curve for binary classification described in Section 3.3.2.

REC curve estimates the Cumulative Distribution Function of the error. On the x axis of the plot there is an error tolerance. On the y axis there is an accuracy at the given tolerance level. Bi and Bennett (2003)[6] define the accuracy at tolerance ϵ as a percentage of observations predicted within the tolerance ϵ. In other words, residuals larger than ϵ are considered as errors.

Let consider pairs (y_i, x_i) as at the beginning of the Chapter 3. Bi and Bennett (2003)[6] define an accuracy as follows.

Definition 3.10. An accuracy at tolerance level ϵ is given by

$$acc(\epsilon) = \frac{|\{(x, y) : loss(f(x_i), y_i) \leqslant \epsilon, i = 1, ..., n\}|}{n}. \tag{3.16}$$

REC Curves implemented in the **auditor** are plotted for a special case of Definition 3.10 where the loss is defined as

$$loss(f(x_i), y_i) = |f(x_i) - y_i| = |r_i|. \tag{3.17}$$

The shape of the curve illustrates the behavior of errors. The quality of the model can be evaluated and compared for different tolerance levels. The stable growth of the accuracy does not indicate any problems with the model. A small increase of accuracy near 0 and areas, where the growth is fast, signalize bias of the model predictions.

This plot is generated by `plot` function with parameter `type = "REC"` or by `plotREC` function.

```
plot(rf_res_fitted, lm_res_fitted, type = "REC")
```

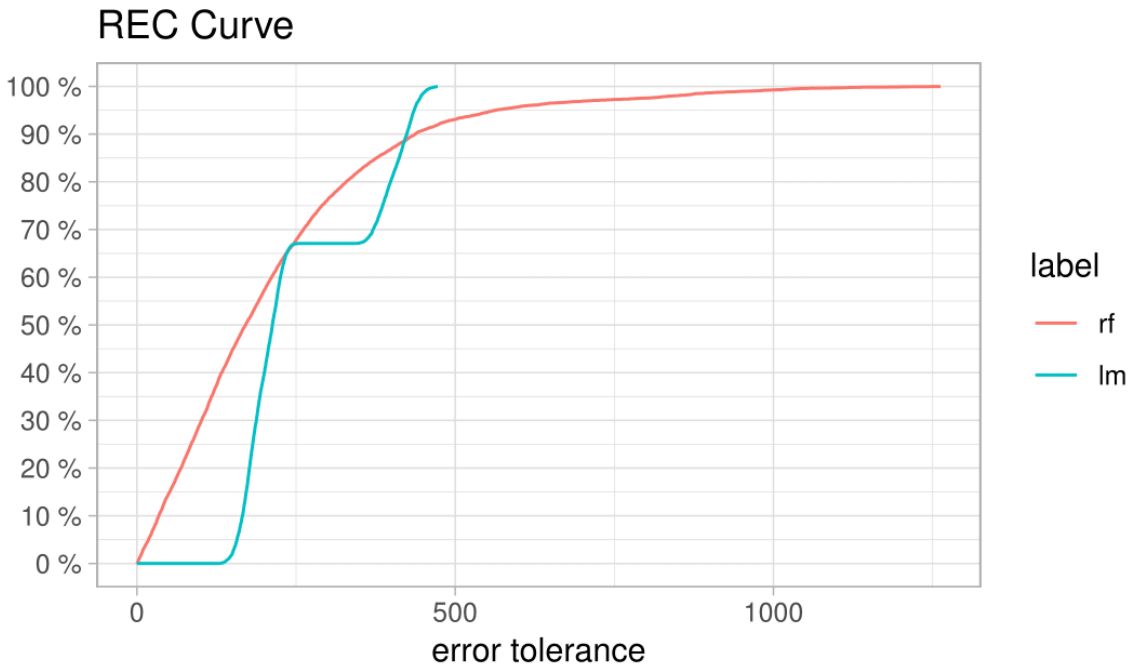


Figure 3.13: REC curve. Curve for linear model (blue) suggests that model is biased. It has poor accuracy when the tolerance ϵ is small. However, once ϵ exceeds the error tolerance 130 the accuracy rapidly increases. The random forest (red) has a stable increase of accuracy when compared to the linear model. However, there is s fraction of large residuals.

As often it is difficult to compare models on the plot, there is an REC score implemented in the **auditor**. This score is the Area Over the REC Curve (AOC), which is a biased estimate of the expected error for a regression model. As Bi and Bennett (2003)[6] proved, AOC provides a measure of the overall performance of regression model.

Scores may be obtained by `score` function with `type = "REC"` or `scoreREC` function.

```
scoreREC(lm_res_fitted)
scoreREC(rf_res_fitted)
```

3.3.4. Two-sided ECDF Plot

Two-sided ECDF plot shows an Empirical Cumulative Distribution Functions (ECDF) for positive and negative values of residuals separately.

Let $x_1, ..., x_n$ be a random sample from a cumulative distribution function $F(t)$. The following definition comes from van der Vaart (2000)[57].

Definition 3.11. The empirical cumulative distribution function is given by

$$F_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{x_i \leqslant t\}. \tag{3.18}$$

Empirical cumulative distribution function presents a fraction of observations that are less or equal than t. It is an estimator for the cumulative distribution function $F(t)$.

On the positive side of the x-axis, there is the ECDF of positive values of residuals. On the negative side, there is a transformation of ECDF:

$$F_{rev}(t) = 1 - F(t). \tag{3.19}$$

Let n_N and n_P be numbers of negative and positive values of residuals, respectively. Negative part of the plot is normalized by multiplying it by the ratio of the n_N over $n_N + n_P$. Similarly, positive part is normalized by multiplying it by the ratio of the n_P over $n_N + n_P$. Due to the scaling, the ends of the curves add up to 100% in total.

The plot shows the distribution of residuals divided into groups with positive and negative values. It helps to identify the asymmetry of the residuals. Points represent residuals.

This plot is generated by `plotTwoSidedECDF` function or by `plot` function with parameter `type = "TwoSidedECDF"`.

```
plot(rf_res_fitted, lm_res_fitted, type = "TwoSidedECDF")
```

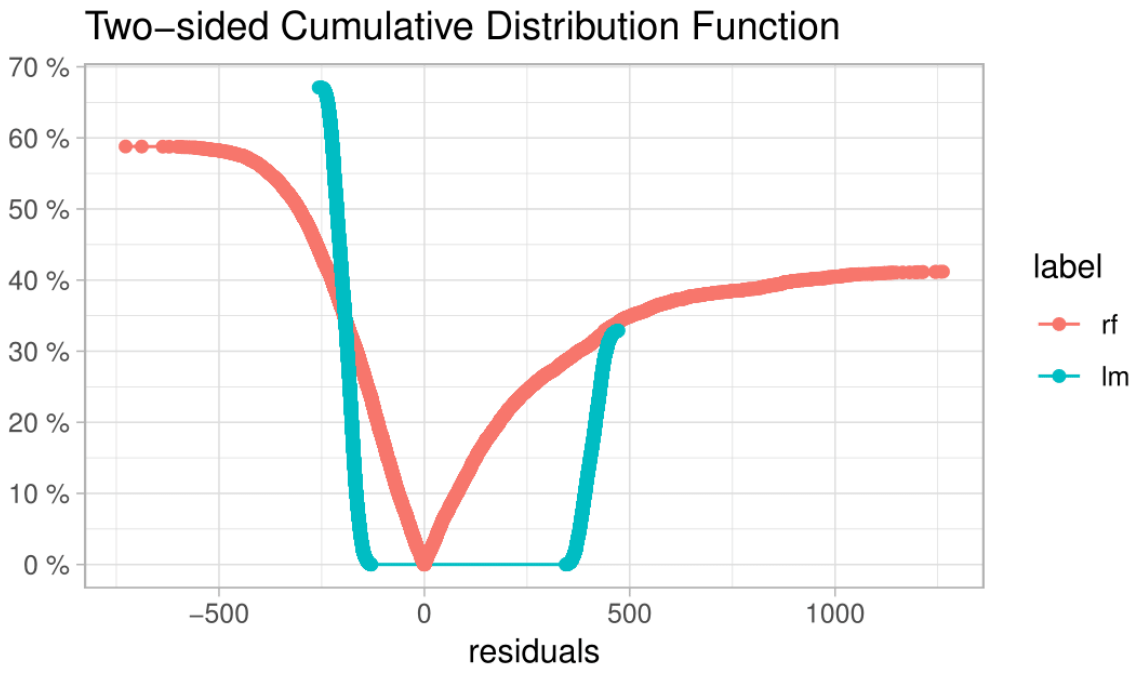


Figure 3.14: Two-sided ECDF plot. The plot shows that majority of residuals for the random forest (red) is smaller than residuals for the linear model (blue). However, random forest has also fractions of large residuals.

3.3.5. RROC Curve Plot

The Regression Receiver Operating Characteristic (RROC) curve for regression shows model asymmetry, which is an inequality in the number and values of positive and negative residuals. It may be useful for problems, where there is an asymmetric cost of errors. An example of RROC plot is presented in Figure 3.15.

The RROC curve was introduced by Hernández-Orallo (2013)[34]. The base of plot is a shift s. It is an equivalent to the threshold for ROC curves presented in Section 3.3.2. For each observation we calculate new prediction: $\hat{y}'^s = \hat{y} + s$. Hernández-Orallo (2013)[34] define over- and under-estimation as follow.

Definition 3.12. Over-estimation depending on shift s is given by

$$OVER(s) = \sum_{i=1}^{n} r_i \mathbb{1}\{r_i + s > 0\}. \tag{3.20}$$

Definition 3.13. Under-estimation depending on shift s is given by

$$UNDER(s) = \sum_{i=1}^{n} r_i \mathbb{1}\{r_i + s < 0\}. \tag{3.21}$$

The RROC plot consists of under-estimation on the y axis against over-estimation on the x axis. The shift equals 0 is represented by a dot.

The shape of the curve illustrates the behavior of errors. The quality of the model can be evaluated and compared for different shifts. The AOC measures the model insensitivity to asymmetric costs of errors.

This plot is generated by `plot` function with parameter `type = "RROC"` or by `plotRROC` function.

```
plot(rf_res_fitted, lm_res_fitted, type = "RROC")
```

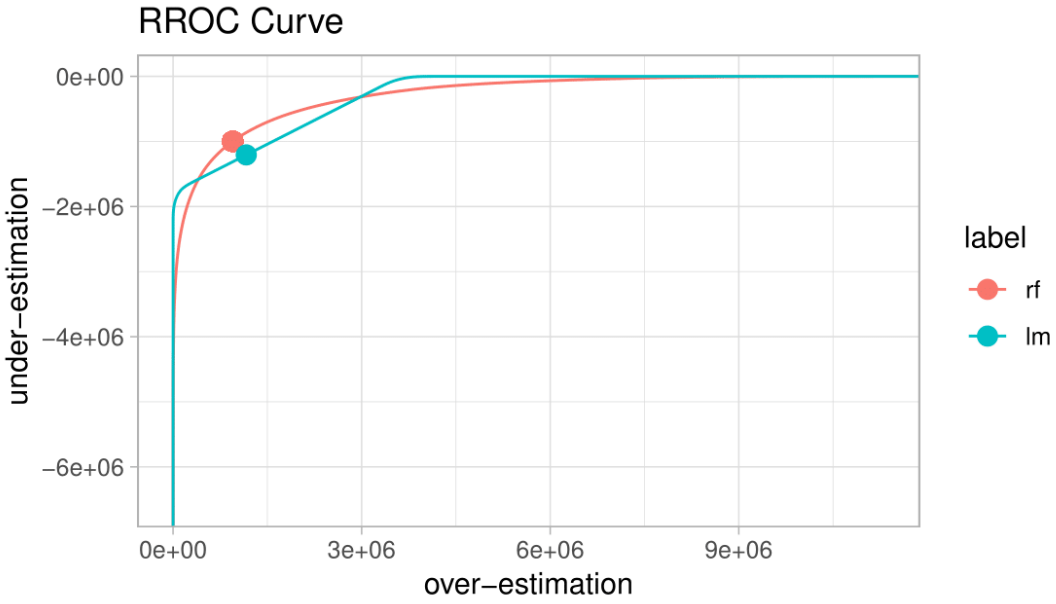


Figure 3.15: The Regression Receiver Operating Characteristic (RROC) curve for regression. For shifts around 0 random forest (red) performs better than linear regression (blue).

As often it is difficult to compare AOC on the plot, there is an RROC score implemented in the auditor package. This score is the Area Over the RROC Curve (AOC). As Hernández-Orallo (2013)[34] prove AOC equals to the variance of the errors multiplied by a $\frac{n^2}{2}$ which is independent of the model. Lower values for AOC are better.

Score may be obtained by `score` function with `type = 'RROC'` or `scoreRROC` function.

```
scoreRROC(lm_res_fitted)
scoreRROC(rf_res_fitted)
```

3.3.6. Model Ranking Plot

In this section, we propose a Model Ranking plot, which compares models performance across multiple measures (see Figure 3.16). The implemented measures coincide with the scores listed in the Chapter 2.

In previous sections, we introduced the following scores:

- DW (Section 3.1.3),
- Peak (Section 3.1.5),
- Half-Normal (Section 3.1.7),
- REC(Section 3.3.3),
- ROC (Section 3.3.2),
- RROC(Section 3.3.5),
- Runs (Section 3.1.3).

Additional scores implemented in the auditor are:

- MAE (Mean Absolute Error)
$$MAE = \frac{1}{n}\sum_{i=1}^{n} \mid r_i \mid, \tag{3.22}$$
- MSE (Mean Squared Error)
$$MSE = \frac{1}{n}\sum_{i=1}^{n} r_i^2, \tag{3.23}$$
- RMSE (Root Mean Squared Error)
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} r_i^2}. \tag{3.24}$$

Model Ranking Radar plot consists of two parts. On the left side, there is a radar plot. Colors correspond to models, edges to values of scores. Score values are inverted and rescaled to $[0, 1]$.

Let us use the following notation: $m_i \in \mathcal{M}$ is a model in a finite set of models $\mathcal{M}$, where $|\mathcal{M}| = k$, $score : \mathcal{M} \to \mathbb{R}$ is a scoring function for the model under consideration, that higher value means worse model performance. The $score(m_i)$ is a performance of model m_i.

Definition 3.14. We define the inverted score of model m_i as

$$invscore(m_i) = \frac{1}{score(m_i)} \min_{j=1...k} score(m_j). \tag{3.25}$$

Models with the larger $invscore$ are closer to the centre. Therefore, the best model is located the farthest from the center of the plot.

On the right side of the plot there is a table with results of scoring. In the third column, there are scores scaled to one of the models.

Let $m_l \in \mathcal{M}$ where $l \in \{1, 2, ..., k\}$ be a model to which we scale.

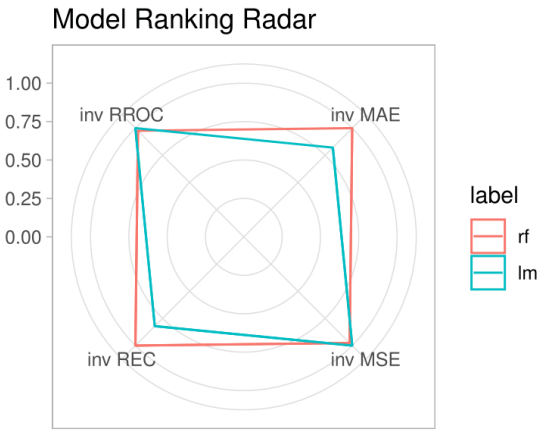
Definition 3.15. We define the scaled score of model m_i to model m_l as

$$scaled_l(m_i) = \frac{score(m_l)}{score(m_i)}. \tag{3.26}$$

As values of $scaled_l(m_l)$ are always between 0 and 1, comparison of models is easy, regardless of the ranges of scores.

This plot is generated by `plot` function with parameter `type = "ModelRanking"` or by function `plotModelRanking`. The scores included in the plot may be specified by `scores` parameter.

```
rf_mp <- modelPerformance(rf_audit)
lm_mp <- modelPerformance(lm_audit)
plot(rf_mp, lm_mp, type = "ModelRanking")
```



name	label	score	scaled
MAE	rf	2.16e+02	1.000
MAE	lm	2.63e+02	0.821
MSE	rf	8.21e+04	1.000
MSE	lm	8.01e+04	1.025
REC	rf	2.16e+02	1.000
REC	lm	2.63e+02	0.821
RROC	rf	3.32e+12	1.000
RROC	lm	3.24e+12	1.024

Figure 3.16: Model Ranking Plot. Random forest (red) has better performance in aspect of MAE and REC scores, while linear model (blue) is better in aspect of MSE and RROC scores.

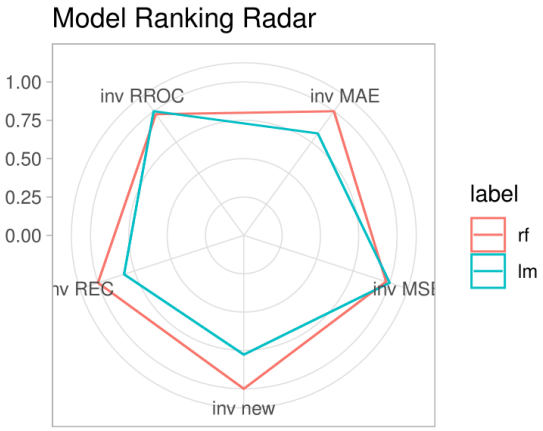
There is also a possibility to add custom scores (Figure 3.17). They may be provided by parameter `new.score`. It requires a named list of functions that take one argument: object of

class `"modelAudit"` and return a numeric value. The measure calculated by the function should have the property that lower score value indicates better model.

```
new_score <- function(object){sum((object$residuals - object$fitted.values)^4)}

rf_mp <- modelPerformance(rf_audit, new.score = list("new" = new_score))
lm_mp <- modelPerformance(lm_audit, new.score = list("new" = new_score))

plot(rf_mp, lm_mp, type = "ModelRanking")
```



name	label	score	scaled
MAE	rf	2.16e+02	1.000
MAE	lm	2.63e+02	0.821
MSE	rf	8.21e+04	1.000
MSE	lm	8.01e+04	1.025
REC	rf	2.16e+02	1.000
REC	lm	2.63e+02	0.821
RROC	rf	3.32e+12	1.000
RROC	lm	3.24e+12	1.024
new	rf	1.51e+18	1.000
new	lm	1.94e+18	0.777

Figure 3.17: Radar plot with model scores. In terms of new score, the random forest (red) has better performance than the linear model (blue).

3.4. Aspect: Influence of Observations

In this section, we focus on the impact of individual observation on a model.

3.4.1. Cook's Distances Plot

Cook's distances plot presented in Figure 3.18 is a tool for identifying observations that may negatively affect the model. They can be also used for indicating regions of the design space where it would be good to obtain more observations. Data points indicated by Cook's distances are worth checking for validity.

Let us extend the notation provided iat the beginning of the Chapter 3. We denote the number of predictors as p and the mean squared error as s^2. We consider recalculated model that is fitted on the original data set with removed j-th obsetvation. Let $\hat{y}_{i(j)}$ be the prediction of such model calculated for i-th observation. Cook (1977)[13] defined the influence of a single observation as follows.

Definition 3.16. Cook's distance of i-th observation is

$$D_i = \frac{\sum_{j=1}(\hat{y}_j - \hat{y}_{i(j)})^2}{ps^2}. \tag{3.27}$$

Cook's Distances are calculated by removing the i-th observation from the data and recalculating the model. It shows an influence of i-th observation on the model.

Cook (1977)[13] proved that for linear models Cook's distance may be computed in an alternative and computationally convenient way. Let $\mathbf{X}$ be the design matrix and $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ be a projection matrix. The leverage h_i is the i-th diagonal element of $\mathbf{H}$. D_i is obtained by

$$D_i = \frac{r_i^2}{ps^2}\frac{h_i}{(1-h_i)^2}. \tag{3.28}$$

This plot is generated by `plot` function with parameter `type = "CooksDistance"` or by function `plotCooksDistance`. For model of class `"lm"` and `"glm"` the distances are computed from the diagonal elements of the hat matrix. For other models they are computed directly from the definition.

```
lm_oi <- observationInfluence(lm_audit)
plotCooksDistance(lm_oi)
```

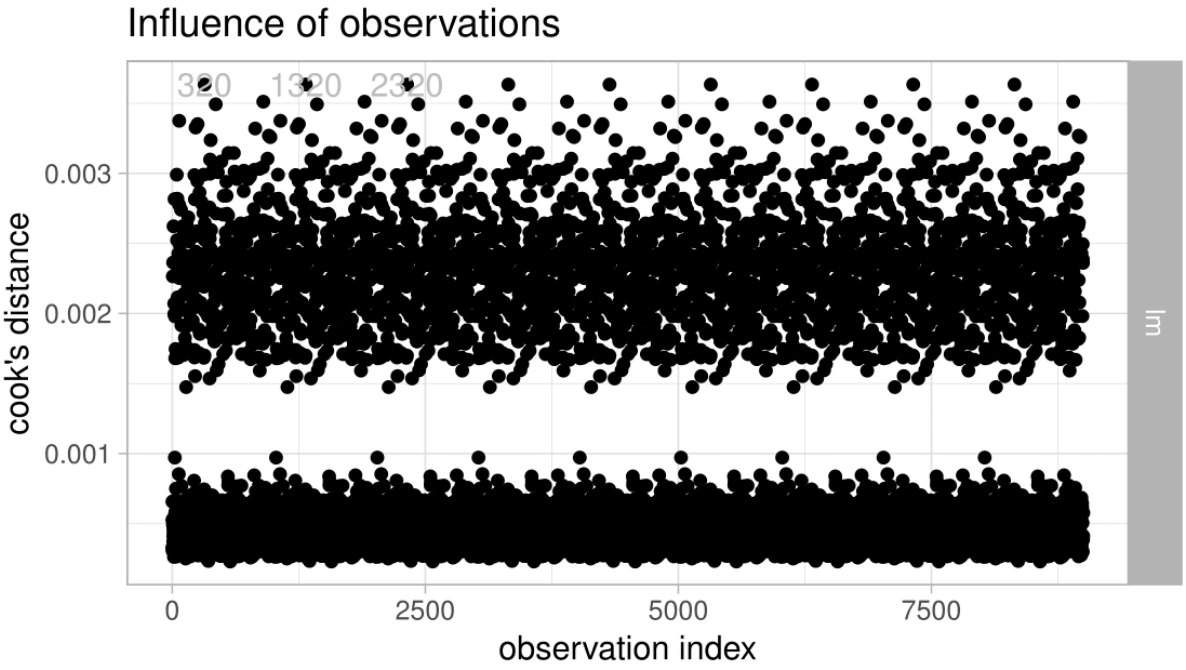


Figure 3.18: Cook's Distances plot. The 3 observations with the highest values of the Cook's distance are marked. However, they do not significantly differ from others.

4. Example of model audit

This section contains a use-case of model audit. We show an example, where the choice of a model on the basis of a single measurement is not obvious. We illustrate, how to use **auditor** to analyze and compare models. In addition, we present how to use residual analysis to identify outliers and better understand a model structure.

4.1. The artificial data set

In order to present the **auditor** we created an artificial data set `auditorData`. It consists of 2000 observations. We generated first 1998 as follows. Let

- $X_1 = (x_1^{(1)}, x_2^{(1)}, ..., x_{1998}^{(1)})$ where $x_i^{(1)} \sim \mathcal{U}[0,1]$,
- $X_2 = (x_1^{(2)}, x_2^{(2)}, ..., x_{1998}^{(2)})$ where $x_i^{(2)} \sim \mathcal{U}[0,1]$,
- $X_3 = (x_1^{(3)}, x_2^{(3)}, ..., x_{1998}^{(3)})$ where $x_i^{(3)} \sim \mathcal{U}[0,1]$,
- $X_4 = (x_1^{(4)}, x_2^{(4)}, ..., x_{1998}^{(4)})$ where $x_i^{(4)}$ is from discrete probability distribution, $P(x_i^{(4)} = 0) = 0.5$, $P(x_i^{(4)} = 1) = 0.35$, $P(x_i^{(4)} = 4) = 0.15$,
- $\epsilon = (\epsilon_1, \epsilon_2, ..., \epsilon_{1998})$ where $\epsilon_i \sim \mathcal{N}(0, 0.5)$

for $i \in 1, 2, ..., 1998$.

Let us simulate a response $Y = (y_1, y_2, ..., y_{1998})$ as a function of five arguments: X_1, X_2, X_3, X_4, ϵ, where

$$y_i = 20(x_i^{(1)} - 1)^2 + 2(x_i^{(2)} - 0.25)(x_i^{(2)} - 0.5)(x_i^{(2)} - 1) + 22x_i^{(3)} - 1 + 5x_i^{(4)}x_i^{(1)} + \epsilon_i. \quad (4.1)$$

The 1999-th and 2000-th observations are added manually. They are meant to be outliers. Their values are

$$(y_{1999}, x^{(1)}_{1999}, x^{(2)}_{1999}, x^{(3)}_{1999}, x^{(4)}_{1999}) = (92, 0.32, 0.21, 0.1, 0) \tag{4.2}$$

and

$$(y_{2000}, x^{(1)}_{2000}, x^{(2)}_{2000}, x^{(3)}_{2000}, x^{(4)}_{2000}) = (98, 0.86, 0.82, 0.85, 0), \tag{4.3}$$

respectively.

First four of simulated variables are treated as continuous while the fifth one is categorical. In the Table 4.1 we present first six observations from the `auditorData` included in the **auditor** package.

```
library("auditor")
data("auditorData")
head(auditorData)
```

y	X1	X2	X3	X4
25.08	0.10	0.11	0.45	0
11.43	0.78	0.38	0.52	0
31.36	0.61	0.33	0.78	4
0.84	0.96	0.01	0.09	0
22.23	0.96	0.32	0.84	1
29.35	0.22	0.16	0.83	0

Table 4.1: First 6 observations from the data set `auditorData`.

Our goal is to predict Y based on selected variables X1, X2, X3 and X4.

4.2. Fitting models

We fit 3 models and audit them. They are: simple linear regression, random forest, and support vector regression. We use `randomForest` function from **randomForest** package (Liaw and Wiener, 2002)[37] and `svm` function from **e1071** package (Meyer et al., 2017)[40].

```
model_lm <- lm(y ~ ., auditorData)


library("randomForest")
set.seed(1994)
model_rf <- randomForest(y~., auditorData)


library("e1071")
model_svm <- svm(y ~ ., auditorData)
```

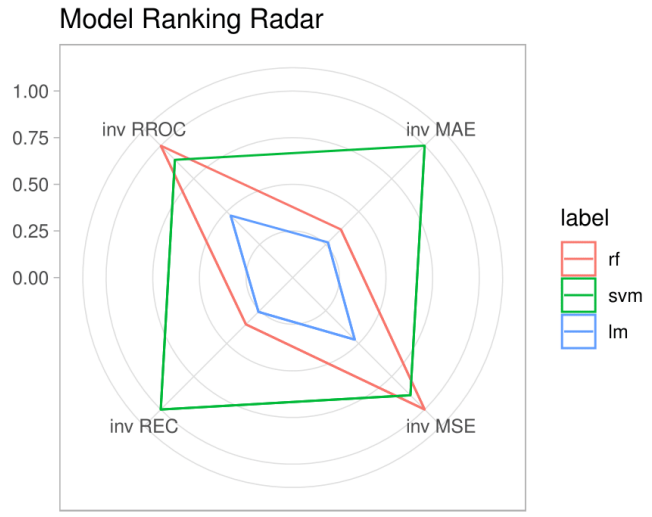
Next step is creating three `"modelAudit"` objects, corresponding to the models.

```
au_lm <- audit(model_lm, data = auditorData, y = auditorData$y)
au_rf <- audit(model_rf, data = auditorData, y = auditorData$y, label = "rf")
au_svm <- audit(model_svm, data = auditorData, y = auditorData$y, label = "svm")
```

4.3. Model audit

At first, we generate four diagnostic plots. They are: Model Ranking Plot, Predicted Response Plot, PCA of Models, and Residuals Plot. We present results in Figure 4.1 and Figure 4.2.

```
plot(au_rf, au_svm, au_lm,
     type = c("ModelRanking", "Prediction"),
     variable = "Observed response", smooth = TRUE)
```



name	label	score	scaled
MAE	rf	1.49e+00	1.000
MAE	svm	5.43e−01	2.741
MAE	lm	2.03e+00	0.732
MSE	rf	6.09e+00	1.000
MSE	svm	6.83e+00	0.892
MSE	lm	1.30e+01	0.470
REC	rf	1.47e+00	1.000
REC	svm	5.23e−01	2.820
REC	lm	2.01e+00	0.733
RROC	rf	1.22e+07	1.000
RROC	svm	1.36e+07	0.893
RROC	lm	2.59e+07	0.470

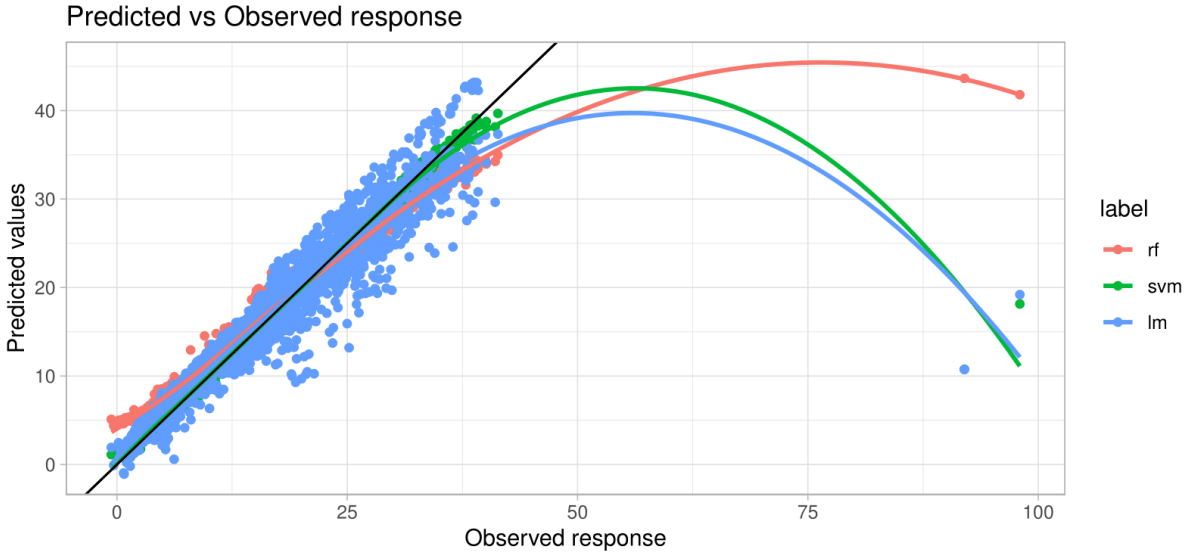


Figure 4.1: Diagnostic plots for linear regression (blue), random forest (red), support vector (green) regression. The upper part shows Model Ranking plot. In the lower part, there is a Predicted Response Plot.

```
plot(au_rf, au_svm, au_lm,
     type = c("ModelPCA", "Residual"),
     variable = "Observed response", nlabel = 6)
```

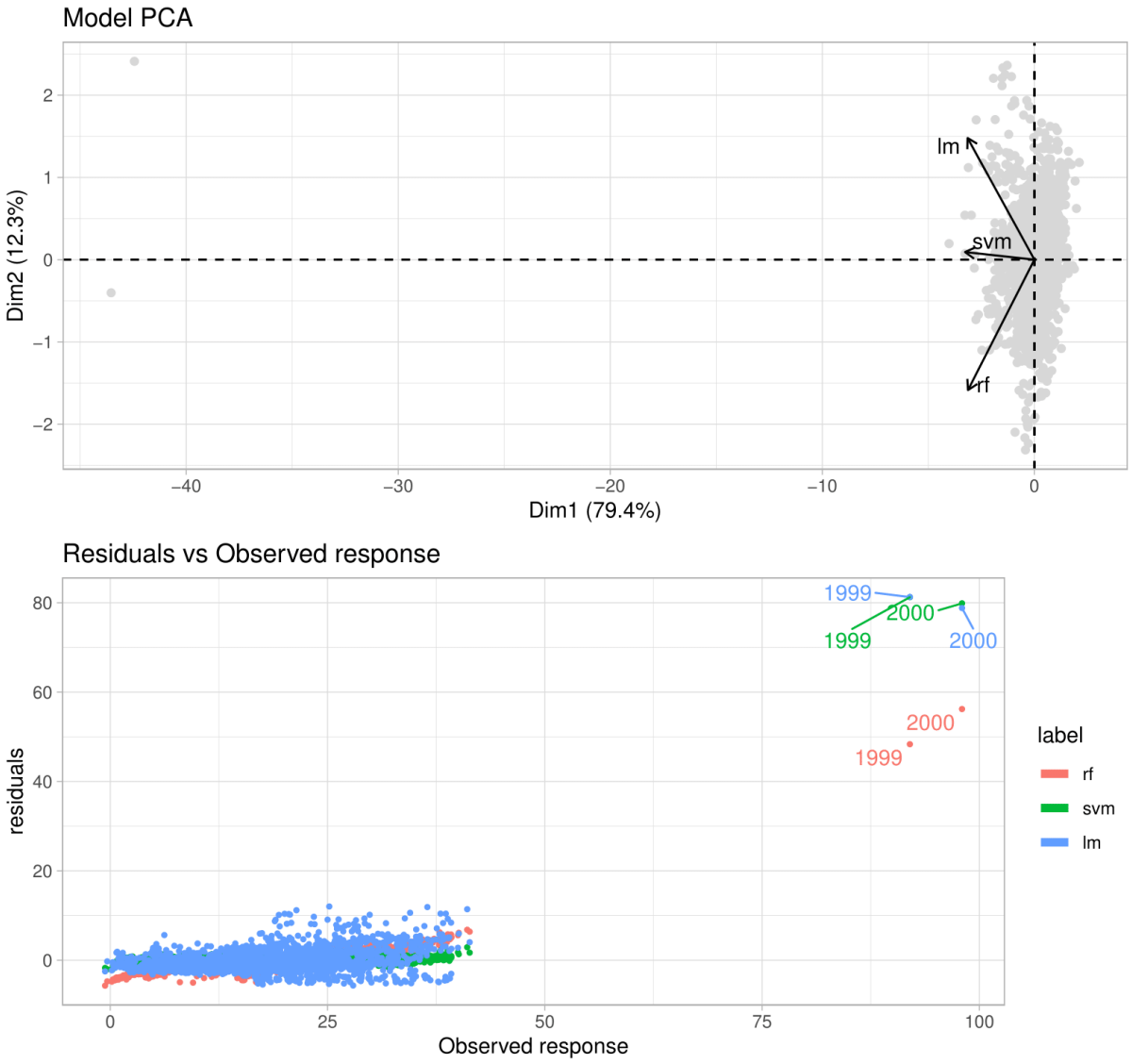


Figure 4.2: Diagnostic plots for linear regression (blue), random forest (red), support vector regression (green). The upper part shows PCA of Models. In the lower part, there is a Residuals Plot.

On the Model Ranking plot (see Figure 4.1) we can see that random forest has the best performance in terms of MSE and $RROC$ sores. Support vector regression is the best in terms of MAE and REC scores. Linear regression appears to have the worst performance in every aspect. As none of the models performs best in terms of all measures, it is clear that

Model Ranking plot is not enough to evaluate models. There is a need for further analysis of residuals.

Prediction Response plot (see Figure 4.1) indicates that there are observations that may be outliers and have significant influence on model's structures. Most of the residuals of all models are arranged along a black line that shows the ideal trend. While, there is a group of points that clearly stands out from the rest. Further plots allow to take a closer look at these observations.

PCA of Models (see Figure 4.2) confirms that there are 2 observations that have high influence on the structures of all models. Residuals plot with labeled points shows the numbers of outliers. These are 1999-th and 2000-th observations. The same that we added artificially.

4.4. Removing identified outliers and models improvement

In previous section, we identified two outliers in `auditorData`. In this section, we remove those outliers, fit models to a new data set, and create `"modelAudit"` objects.

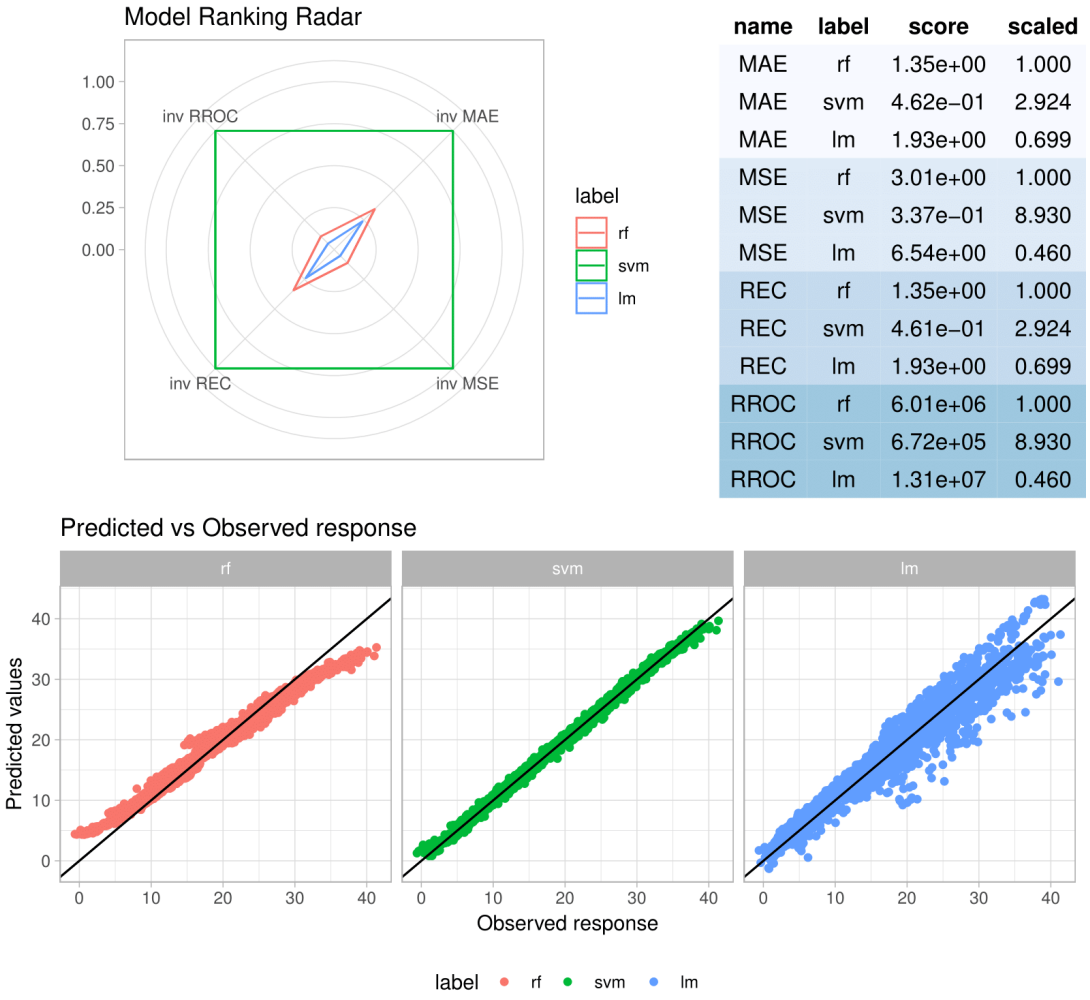
```
auditorData_clean <- auditorData[-c(1999, 2000), ]

model_lm <- lm(y ~ ., auditorData_clean)
set.seed(1994)
model_rf <- randomForest(y~., auditorData_clean)
model_svm <- svm(y ~ ., auditorData_clean)

au_lm <- audit(model_lm, data = auditorData_clean, y = auditorData_clean$y)
au_rf <- audit(model_rf, data = auditorData_clean, y = auditorData_clean$y,
  label = "rf")
au_svm <- audit(model_svm, data = auditorData_clean, y = auditorData_clean$y,
  label = "svm")
```

We again generate Model Ranking and Predicted Response Plots. Both of them changed noticeably.

```
plot(au_rf, au_svm, au_lm,
  type = c("ModelRanking", "Prediction"),
  variable = "Observed response",
  split = "model")
```



name	label	score	scaled
MAE	rf	1.35e+00	1.000
MAE	svm	4.62e−01	2.924
MAE	lm	1.93e+00	0.699
MSE	rf	3.01e+00	1.000
MSE	svm	3.37e−01	8.930
MSE	lm	6.54e+00	0.460
REC	rf	1.35e+00	1.000
REC	svm	4.61e−01	2.924
REC	lm	1.93e+00	0.699
RROC	rf	6.01e+06	1.000
RROC	svm	6.72e+05	8.930
RROC	lm	1.31e+07	0.460

Figure 4.3: Diagnostic plots for linear regression (blue), random forest (red), support vector regression (green) fitted on new data set. The upper part shows Model Ranking Plot. In the lower part there is a Predicted Response Plot.

This time, the Model Ranking plot clearly indicates that support vector regression performs best. However, to better exploration of models, it is still worth to carry out a further audit. The Predicted Response Plot shows that residuals of support vector regression are closest to the diagonal line than residuals of other models. Residuals of linear model have the largest dispersion, while random forest residuals indicate over-prediction for small values of observed response and under-prediction for large values.

4.5. Extended model audit

In this section we take a closer look on the residuals. We generate Residual Boxplot and Two-sided ECDF plot for better comparison of models in aspect of residuals.

```
plot(au_rf, au_svm, au_lm,
  type = c("ResidualBoxplot", "TwoSidedECDF"),
  variable = "Observed response")
```

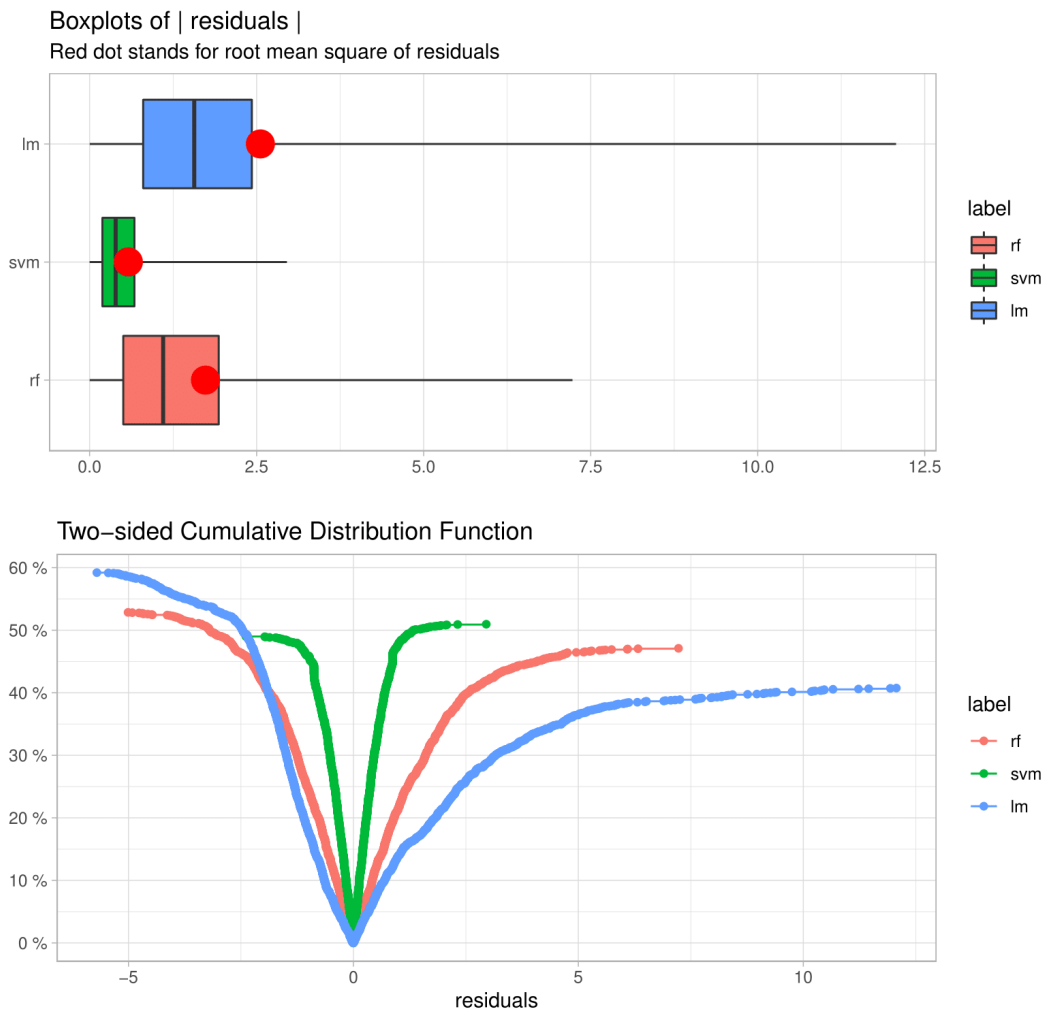


Figure 4.4: Diagnostic plots for linear regression (blue), random forest (red), support vector (green) regression. The upper part shows Residual Boxplot. In the lower part there is a Two-sided ECDF plot.

In Figure 4.3, we noticed that support vector regression has the smallest residuals. By analyzing boxplots (see Figure 4.4), we can additionally see that random forest has smaller residuals than linear model. Two-sided ECDF plots show that conclusions from boxplots are, in general,

correct. In addition, we can see that growth of small negatives residuals for random forest and linear model is similar.

Now, we audit residuals due to the model variables. We use Residual Density to understand residual behaviour for different values of the $X4$ variable.

```
plot(au_rf, au_svm, au_lm,
  type = "ResidualDensity",
  variable = "X4")
```

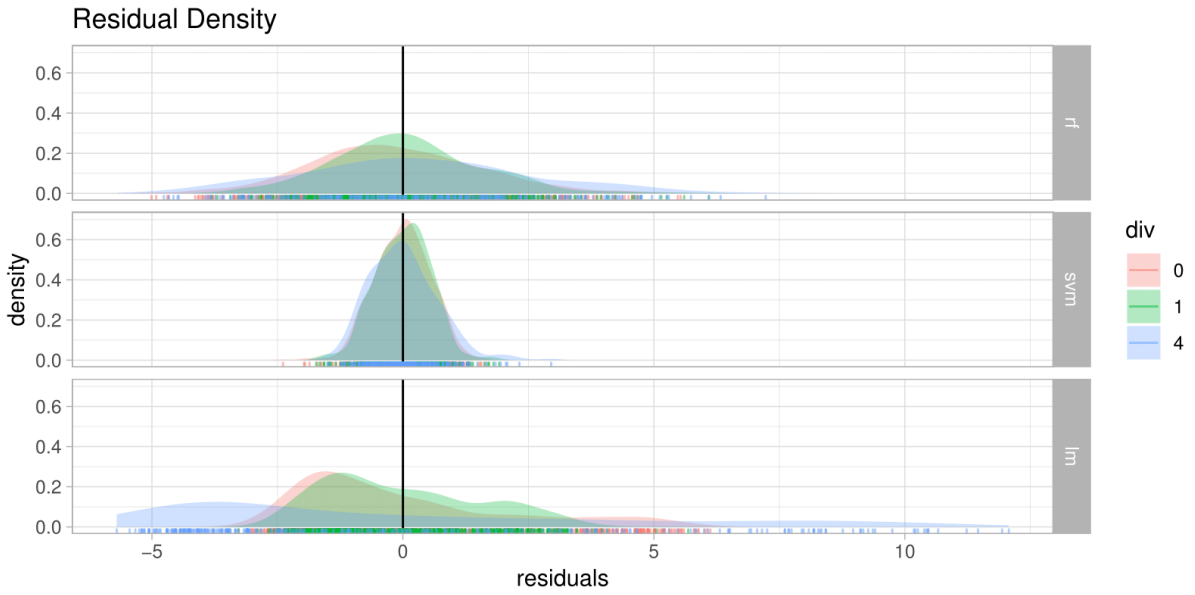


Figure 4.5: Each panel of the Residual Density Plot corresponds to one of the models. Colors indicates values of $X4$ variable.

In the Figure 4.5, we see that structure of residuals for random forest and support vector regression do not vary due to the value of $X4$ variable. In contrast, there are differences in the shape of the densities for linear model's residuals. In the data set, we included the $X4$ variable in the interaction with $X1$. Due to its structure, the linear model do not catch this interaction. We use Residuals Plot to examine behaviour of residuals due to second variable included in the interaction.

```
plot(au_rf, au_svm, au_lm,
 type = "Residual",
 variable = "X1")
```

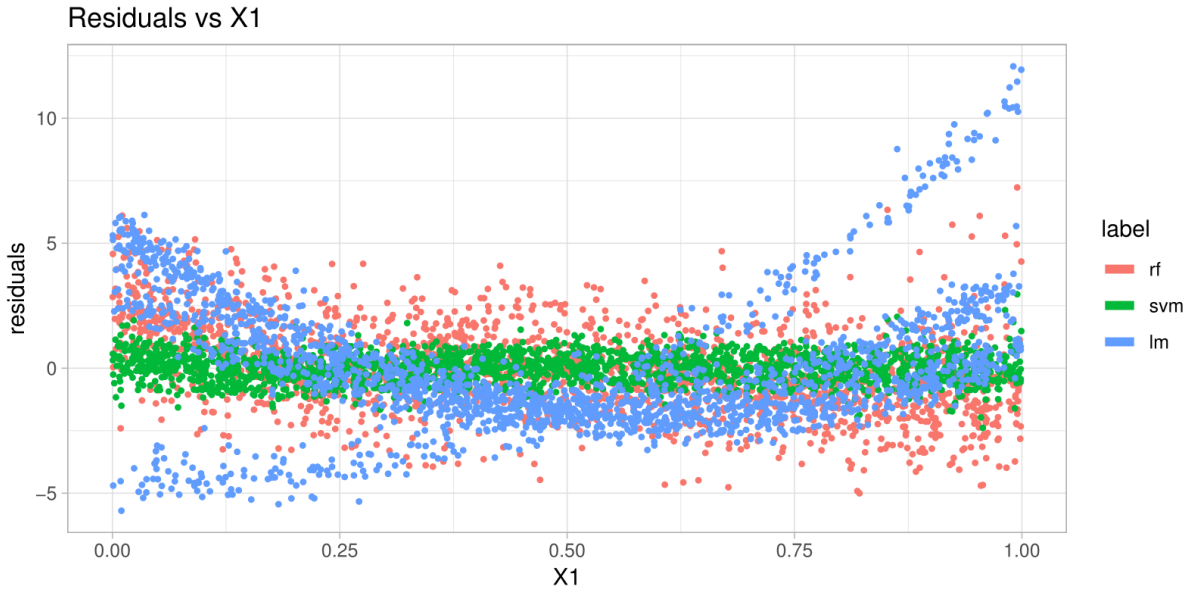


Figure 4.6: Residuals vs variable $X1$ Plot generated for random forest, support vector regression and linear model with ordinary least squares.

In the Figure 4.6, we see that residuals form three groups. Based on the previous analysis, we conclude that groups correspond to the levels of the $X4$ variable.

4.6. Summary

We fitted three different models to the data set `auditData`. We used Predicted Response, Model PCA and Residuals plots to identify two observations, that highly influenced model's structures. Then we refitted models on data without outliers. The audit showed that support vector regression has the best performance and we did not identify any serious problems with the structure of the residuals.

5. Conclusion and future work

In this article, we presented a number of diagnostic scores and plots. We discussed the existing methods of model validation and proposed their adaptations to any model. We also specified three objectives of model audit, proposed relevant verification tools, and demonstrated their usage by incorporating examples.

- First objective of model diagnostic was the identification of observations that are worth further examination. The detection of influential observations is possible with Cook's Distances Score and related Cook's Distances Plot presented in Section 3.4.
- Second objective was examination of a model performance. It involves using scores and supplementing them by visual validation. In Section 3.3 we showed tools that may be used for that purpose. These include scores such as Mean Squared Error, Area Over the RROC, and Area Over the REC. In addition, we presented plots that enrich information about model performance. They are, among others, Model Ranking Plot, Receiver Operating Characteristic Curve, and Regression Error Curve.
- Third considered objective was the recognition of problems that might be identified by analysis of distribution of residuals. With tools described in Sections 3.1 and 3.2, it is possible to answer questions like: "Does the model fit data?"," How similar models are?", and "Does the model miss any relevant information?". Examination of the behavior of residuals provides valuable information about a model. The scores related to this issue were, for example, Runs and Durbin-Watson scores. Presented plots were, among others, Residuals Plot, Residual Boxplot, and Residual Density.

Aside from describing existing methods and extending them to any model, we proposed new plots and scores. They are Model Ranking Plot, Two-Sided ECDF Plot and Half-Normal Score.

We implemented all introduced scores and plots in the **auditor** package for R. Included functions are based on a uniform grammar introduced in Figure 2. Documentation and examples are available at `https://mi2-warsaw.github.io/auditor`. The stable version of the package is on CRAN, the development version is on GitHub (`https://github.com/mi2-warsaw/auditor`).

In Section 4 we showed the use-case of the model audit with the **auditor**. We have presented a broad exploration and comparisons of three models by analyzing their residuals.

There are many potential areas for future work that we would like to explore. This includes more extensions of model-specific diagnostic to model-agnostic methods and residual-based methods for investigating interactions. Another aim is to develop methods for local audit based on the diagnostic of a model around a single observation or a group of observations.

Bibliography

[1] T. W. Anderson and D. A. Darling. Asymptotic theory of certain goodness of fit criteria based on stochastic processes. *Ann. Math. Statist.*, 23(2):193–212, 06 1952. doi: 10.1214/aoms/1177729437. URL https://doi.org/10.1214/aoms/1177729437.

[2] F. J. Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1):17–21, 1973. doi: 10.1080/00031305.1973.10478966. URL https://www.tandfonline.com/doi/abs/10.1080/00031305.1973.10478966.

[3] A. Atkinson. *Plots, Transformations, and Regression: An Introduction to Graphical Methods of Diagnostic Regression Analysis*. Oxford statistical science series. Clarendon Press, 1985. URL https://books.google.pl/books?id=oFjgnQEACAAJ.

[4] A. Atkinson and M. Riani. *Robust Diagnostic Regression Analysis*. Springer Series in Statistics. Springer New York, 2012. ISBN 9781461211600. URL https://books.google.pl/books?id=sZ3SBwAAQBAJ.

[5] R. Becker and J. Chambers. Auditing of data analyses. *SIAM Journal on Scientific and Statistical Computing*, 9(4):747–760, 1988. doi: 10.1137/0909049. URL https://doi.org/10.1137/0909049.

[6] J. Bi and K. P. Bennett. Regression error characteristic curves. In *ICML*, 2003.

[7] P. Biecek. DALEX: explainers for complex predictive models. *ArXiv e-prints*, June 2018.

[8] G. E. P. Box and D. R. Cox. An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological*, pages 211–252, 1964.

[9] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145 – 1159, 1997. ISSN 0031-3203. doi: https://doi.org/10.1016/S0031-3203(96)00142-2. URL http://www.sciencedirect.com/science/article/pii/S0031320396001422.

[10] L. Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statist. Sci.*, 16(3):199–231, 08 2001. doi: 10.1214/ss/1009213726. URL https://doi.org/10.1214/ss/1009213726.

[11] T. S. Breusch and A. R. Pagan. A simple test for heteroscedasticity and random coefficient variation. *Econometrica*, 47(5):1287–1294, 1979. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1911963.

[12] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016. URL http://arxiv.org/abs/1603.02754.

[13] R. D. Cook. Detection of influential observation in linear regression. *Technometrics*, 19(1): 15–18, 1977. ISSN 00401706. URL http://www.jstor.org/stable/1268249.

[14] H. Cramer. On the composition of elementary errors: second paper: statistical applications. *Scandinavian Actuarial Journal*, 1928(1):141–180, 1928.

[15] E. de Jonge and M. van der Loo. *validatetools: Checking and Simplifying Validation Rule Sets*, 2018. URL https://CRAN.R-project.org/package=validatetools. R package version 0.4.3.

[16] J. Durbin and G. S. Watson. Testing for serial correlation in least squares regression. iii. *Biometrika*, 58(1):1–19, 1971. ISSN 00063444. URL http://www.jstor.org/stable/2334313.

[17] J. Faraway. *Practical Regression and Anova Using R*. University of Bath, 2002. URL https://books.google.pl/books?id=UjhBnwEACAAJ.

[18] J. Faraway. *Linear Models with R*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 2004. ISBN 9780203507278. URL https://books.google.pl/books?id=fvenzpofkagC.

[19] T. Fawcett. An introduction to roc analysis. *Pattern Recogn. Lett.*, 27(8):861–874, June 2006. ISSN 0167-8655. doi: 10.1016/j.patrec.2005.10.010. URL http://dx.doi.org/10.1016/j.patrec.2005.10.010.

[20] J. Fox and S. Weisberg. *An R Companion to Applied Regression*. Sage, Thousand Oaks CA, second edition, 2011. URL http://socserv.socsci.mcmaster.ca/jfox/Books/Companion.

[21] M. Friendly. Corrgrams: Exploratory displays for correlation matrices. *The American Statistician*, 56(4):316–324, November 2002.

[22] K. P. F.R.S. *X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling*, volume 50. Taylor & Francis, 1900. doi: 10.1080/14786440009463897. URL https://doi.org/10.1080/14786440009463897.

[23] K. R. GABRIEL. The biplot graphic display of matrices with application to principal component analysis. *Biometrika*, 58(3):453–467, 1971. doi: 10.1093/biomet/58.3.453. URL http://dx.doi.org/10.1093/biomet/58.3.453.

[24] A. Gałecki and T. Burzykowski. *Linear Mixed-Effects Models Using R: A Step-by-Step Approach*. Springer Texts in Statistics. Springer New York, 2013. ISBN 9781461439004. URL https://books.google.pl/books?id=rbk_AAAAQBAJ.

[25] S. M. Goldfeld and R. E. Quandt. Some tests for homoscedasticity. *Journal of the American Statistical Association*, 60(310):539–547, 1965. doi: 10.1080/01621459.1965.10480811. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1965.10480811.

[26] B. M. Greenwell. pdp: An R Package for Constructing Partial Dependence Plots. *The R Journal*, 9(1):421–436, 2017. URL https://journal.r-project.org/archive/2017/RJ-2017-016/index.html.

[27] J. Gross and U. Ligges. *nortest: Tests for Normality*, 2015. URL https://CRAN.R-project.org/package=nortest. R package version 1.0-4.

[28] F. E. Harrell, Jr. *Regression Modeling Strategies*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387952322.

[29] F. E. Harrell Jr. *rms: Regression Modeling Strategies*, 2018. URL https://CRAN.R-project.org/package=rms. R package version 5.1-2.

[30] M. J. Harrison and B. P. M. McCabe. A test for heteroscedasticity based on ordinary least squares residuals. *Journal of the American Statistical Association*, 74(366):494–499, 1979. ISSN 01621459. URL http://www.jstor.org/stable/2286361.

[31] J. Hartigan. Printer graphics for clustering. *Journal of Statistical Computation and Simulation*, 4(3):187–213, 1975. doi: 10.1080/00949657508810123. URL https://doi.org/10.1080/00949657508810123.

[32] A. C. Harvey and P. Collier. Testing for functional misspecification in regression analysis. *Journal of Econometrics*, 6(1):103 – 119, 1977. ISSN 0304-4076. doi: https://doi.org/10.1016/0304-4076(77)90057-4. URL http://www.sciencedirect.com/science/article/pii/0304407677900574.

[33] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[34] J. Hernández-Orallo. Roc curves for regression. *Pattern Recognition*, 46(12):3395 – 3411, 2013. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2013.06.014. URL http://www.sciencedirect.com/science/article/pii/S0031320313002665.

[35] T. K. Ho. Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95, pages 278–, Washington, DC, USA, 1995. IEEE Computer Society. ISBN 0-8186-7128-9. URL http://dl.acm.org/citation.cfm?id=844379.844681.

[36] I. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.

[37] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3): 18–22, 2002. URL https://CRAN.R-project.org/doc/Rnews/.

[38] R. Littell, G. Milliken, W. Stroup, R. Wolfinger, and O. Schabenberger. *SAS for Mixed Models, Second Edition*. SAS Institute, 2007. ISBN 9781599940786. URL https://books.google.pl/books?id=z9qv32OyEu4C.

[39] S. Liu, X. Wang, M. Liu, and J. Zhu. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1):48 – 56, 2017. ISSN 2468-502X. doi: https://doi.org/10.1016/j.visinf.2017.01.006. URL http://www.sciencedirect.com/science/article/pii/S2468502X17300086.

[40] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2017. URL https://CRAN.R-project.org/package=e1071. R package version 1.6-8.

[41] R. Moral, J. Hinde, and C. Demétrio. Half-normal plots and overdispersed models in r: The hnp package. *Journal of Statistical Software, Articles*, 81(10):1–23, 2017. ISSN 1548-7660. doi: 10.18637/jss.v081.i10. URL https://www.jstatsoft.org/v081/i10.

[42] C. O'Neil. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown Publishing Group, New York, NY, USA, 2016. ISBN 0553418815, 9780553418811.

[43] A. H. Petersen and C. T. Ekstrom. *dataMaid: A Suite of Checks for Identification of Potential Errors in a Data Frame as Part of the Data Screening Process*, 2018. URL https://CRAN.R-project.org/package=dataMaid. R package version 1.1.2.

[44] E. A. Peña and E. H. Slate. Global validation of linear model assumptions. *Journal of the American Statistical Association*, 101(473):341–354, 2006. doi: 10.1198/016214505000000637. URL https://doi.org/10.1198/016214505000000637. PMID: 20157621.

[45] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. URL https://www.R-project.org/.

[46] J. B. Ramsey. Tests for specification errors in classical linear least-squares regression analysis. *Journal of the Royal Statistical Society. Series B (Methodological)*, 31(2):350–371, 1969. ISSN 00359246. URL http://www.jstor.org/stable/2984219.

[47] D. Robinson. *broom: Convert Statistical Analysis Objects into Tidy Data Frames*, 2018. URL https://CRAN.R-project.org/package=broom. R package version 0.4.4.

[48] S. Sheather. *A Modern Approach to Regression with R*. Springer Texts in Statistics. Springer New York, 2009. ISBN 9780387096070. URL https://books.google.pl/books?id=zS3Jiyxqr98C.

[49] C. Sievert, C. Parmer, T. Hocking, S. Chamberlain, K. Ram, M. Corvellec, and P. Despouy. *plotly: Create Interactive Web Graphics via 'plotly.js'*, 2017. URL https://CRAN.R-project.org/package=plotly. R package version 4.7.1.

[50] M. A. Stephens. Edf statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, 69(347):730–737, 1974. ISSN 01621459. URL http://www.jstor.org/stable/2286009.

[51] J. Swets. Measuring the accuracy of diagnostic systems. *Science*, 240(4857):1285–1293, 1988. ISSN 0036-8075. doi: 10.1126/science.3287615. URL http://science.sciencemag.org/content/240/4857/1285.

[52] Y. Tang. Autoplotly - Automatic Generation of Interactive Visualizations for Popular Statistical Results. *ArXiv e-prints*, Mar. 2018.

[53] Y. Tang, M. Horikoshi, and W. Li. ggfortify: Unified Interface to Visualize Statistical Results of Popular R Packages. *The R Journal*, 8(2):474–485, 2016. URL https://journal.r-project.org/archive/2016/RJ-2016-060/index.html.

[54] J. Tukey. *Exploratory Data Analysis*. Addison-Wesley series in behavioral science. Addison-Wesley Publishing Company, 1977. ISBN 9780201076165. URL https://books.google.pl/books?id=UT9dAAAAIAAJ.

[55] J. M. Utts. The rainbow test for lack of fit in regression. *Communications in Statistics - Theory and Methods*, 11(24):2801–2815, 1982. doi: 10.1080/03610928208828423. URL https://doi.org/10.1080/03610928208828423.

[56] M. van der Loo. *lumberjack: Track Changes in Data*, 2017. URL https://CRAN.R-project.org/package=lumberjack. R package version 0.2.0.

[57] A. van der Vaart. *Asymptotic Statistics*. Asymptotic Statistics. Cambridge University Press, 2000. ISBN 9780521784504. URL https://books.google.pl/books?id=UEuQEM5RjWgC.

[58] W. Venables and B. Ripley. *Modern Applied Statistics with S*. Statistics and Computing. Springer New York, 2013. ISBN 9780387217062. URL https://books.google.pl/books?id=CzwmBQAAQBAJ.

[59] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL http://www.stats.ox.ac.uk/pub/MASS4. ISBN 0-387-95457-0.

[60] R. Von Mises. *Wahrscheinlichkeit, Statistik und Wahrheit*. Number t. 3 in Schriften zur wissenschaftlichen Weltauffassung. J. Springer, 1928. URL https://books.google.pl/books?id=W1IaAAAAIAAJ.

[61] M. Vuk and T. Curk. ROC Curve, Lift Chart and Calibration Plot. *Metodološki zvezki*, 3(1):89–108, 2006.

[62] A. Wald and J. Wolfowitz. On a test whether two samples are from the same population. *Ann. Math. Statist.*, 11(2):147–162, 06 1940. doi: 10.1214/aoms/1177731909. URL https://doi.org/10.1214/aoms/1177731909.

[63] H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. ISBN 978-0-387-98140-6. URL http://ggplot2.org.

[64] H. Wickham. Tidy data. *The Journal of Statistical Software*, 59, 2014. URL http://www.jstatsoft.org/v59/i10/.

[65] C. J. Willmott, S. G. Ackleson, R. E. Davis, J. J. Feddema, K. M. Klink, D. R. Legates, J. O'Donnell, and C. M. Rowe. Statistics for the evaluation and comparison of models. *Journal of Geophysical Research*, 90(C5):8995, 1985. doi: 10.1029/jc090ic05p08995. URL https://doi.org/10.1029/jc090ic05p08995.

[66] I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011. ISBN 0123748569, 9780123748560.

[67] K. Wright. *corrgram: Plot a Correlogram*, 2018. URL https://CRAN.R-project.org/package=corrgram. R package version 1.13.

[68] M. N. Wright and A. Ziegler. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17, 2017. doi: 10.18637/jss.v077.i01.

[69] A. Zeileis and T. Hothorn. Diagnostic checking in regression relationships. *R News*, 2(3): 7–10, 2002. URL https://CRAN.R-project.org/doc/Rnews/.

[70] A. Zeileis, F. Leisch, K. Hornik, and C. Kleiber. strucchange: An r package for testing for structural change in linear regression models. *Journal of Statistical Software, Articles*, 7(2): 1–38, 2002. ISSN 1548-7660. doi: 10.18637/jss.v007.i02. URL https://www.jstatsoft.org/v007/i02.

List of Figures

List of Tables

List of Annexes

1. Content of the included CD

Content of the included CD

The attached CD contains the **auditor** package for R (version 0.3.0) and the code included in the paper.

If for some reason CD version is not available, codes and package are accessible from the following release on the **auditor's** GitHub repository: `https://github.com/mi2-warsaw/auditor/releases/tag/0.3.0`.