Analysis of implausible projects

Dieter Reinwald

2022

Contents

1 Executive summary

In highly competitive markets, customer loyalty and customer satisfaction are two essential aspects of successful businesses. One aspect is correct invoices issued to the customer. If they are correct, the customer will pay for the products and services without complaint. However, if they are incorrect or seem implausible to the customer, it can lead from complaint and refusal to pay to litigation. This matter is aggravated when, due to the high number of invoices, they are no longer generated manually but automatically and can no longer be checked by people. The impact on the company's reputation is critical and can lead to significant damage in the short but also in the long term (social media, word of mouth effects etc.)

The project presented here aims to distinguish plausible from implausible service project invoices on the basis of predictors. By this, a manual check can already take place before invoicing and thus avoid dissatisfaction. In order to do this, we will analyze data of all service projects and service project invoices from the last couple years. The invoices that have been classified as implausible are from year 2015 to 2020, the invoices to be predicted come from year 2021.

The company that we will use for this service project in invoice prediction is a European supplier of shading solutions with about 200,000 project invoices per year. It not only produces and sells these products, but also provides after-sales services, i.e. repair and maintenance. We will focus on this in the following.

We will train, test, and compare different classification models and apply the best one to a validation set in order to predict implausible projects and invoices.

2 Overview

The following steps will be performed to reach the goal:

1. Import data: The required data for projects, invoices and already flagged projects (based on business plausibility checks) is imported.
2. Explore, clean and visualize data: We start with a general exploratory data analysis and will continue by visualizing the data by use of histograms, bar charts, smooth plots and a correlation analysis. We will make some preprocess steps for the data as well.
3. Modeling approach: We take the steps needed for training and testing, i.e. reduction of the data set by unneeded variables and split of the data into train, test and validation set.
4. Train and compare different models: In total, we use four different models to identify the best performing model. These are naive bayes, logistic regression, decision tree, and random forest. The different performances are directly discussed.
5. Model prediction: The best two models are selected and applied on the validation set to predict implausible projects for 2021. Although, we present and discuss the different model results after each analysis briefly.
6. Conclusion: Finally, we make a brief summary of the report, its limitations and show potential approaches and ideas for future work.

3 Import data

For importing the data, we first load it from the publicly available GitHub website Implausible_projects. The data provided by the company has been anonymized and shared on a web site in different files:

1. **projects.csv** - this file represents the master data of the service projects including information like project id, sales channel, create date etc.
2. **invoices_xxx.csv** - these files show the invoice details for the service projects like invoice date, due date, net amount, tax, gross amount and some categorical data (e.g. type, employee or item details) for the years 2015 to 2021. These invoices are linked to the projects data via the variable proj_invoice_projid. The invoice files are separate for each year, will be merged and then left joined to the projects file to combine all data in one table. As a rule, one invoice exists per project. However, there are cases when there is more than one invoice per project.
3. **flagged_projects.csv** - in this file we see the project ids that have been flagged implausible by business experts.

We only take the unique values for projects, invoices and flagged projects.

4 Explore, clean and visualize data

In this section we explore, clean and visualize the data it to gain deeper insights about potential relationships between the different variables. Additionally, we preprocess the data.

4.1 Exploratory data analysis

We start with an overall analysis of the different data sets, including:

- the explanation of the data structure, dimensions, and first glimpse to the data
- the initial data types
- the required mutations (including data type conversion)
- the representation of the final summary

4.1.1 Projects data

4.1.1.1 Data structure We can see that the projects data consists of the following columns:

- proj_id - the unique id of the service project
- channel - the sales channel through which the project was sold
- correction - flag for a correction project when something went wrong in an earlier project and needs to be fixed
- fix - flag for a fixed price project
- project_create_date - the create date of the project in the enterprise resource planning (ERP) system
- project_start_date - the start date of the project in the ERP system
- project_end_date - the end date of the project in the ERP system
- project_status - the status of the project
- cust_account - the customer account for the project
- invoice_account - the invoice account for the project
- blocked_for_invoice - flag if the project is blocked for invoicing (e.g. if legal issues occurred)
- warranty_claim - the status of the project warranty claim
- proj_invoice_projid - the project contract id (also join element for the invoice data)

The dimensions and the first rows of the data file are as follows:

```
## [1] 289065     13
```

Table 1: Head data of table projects (continued below)

proj_id	channel	correction	fix	project_create_date	project_start_date
5488372	channel_25	0	0	2014-12-25	1900-01-01
5488655	channel_01	1	0	2014-12-25	1900-01-01
5507548	channel_23	0	0	2014-12-25	1900-01-01
5542726	channel_10	0	1	2014-12-25	1900-01-01
5562929	channel_16	0	0	2014-12-25	1900-01-01
5563364	channel_25	0	0	2014-12-25	1900-01-01

Table 2: Table continues below

project_end_date	project_status	cust_account	invoice_account	blocked_for_invoice
2016-09-26	work finished	8443742	8443742	0
2015-09-09	work finished	8254661	8254661	0
2016-07-29	work finished	8437389	8454203	0
2016-07-29	work finished	8025085	8025085	0
1900-01-01	work finished	8285009	8285009	0
1900-01-01	work finished	8434538	8434538	0

warranty_claim	proj_invoice_projid
0	PV-0001580
1	PV-0002428
0	PV-0007220
0	PV-0001306
0	PV-0007302
0	PV-0006976

4.1.1.2 Initial data types The initial data types are shown in the following table.We see that the date data types have two different types (POSIXct, POSIXt).

Table 4: Initial data types projects (continued below)

proj_id	channel	correction	fix	project_create_date	project_start_date
numeric	character	numeric	numeric	POSIXct	POSIXct
numeric	character	numeric	numeric	POSIXt	POSIXt

Table 5: Table continues below

project_end_date	project_status	cust_account	invoice_account	blocked_for_invoice
POSIXct	character	numeric	numeric	numeric
POSIXt	character	numeric	numeric	numeric

warranty_claim	proj_invoice_projid
numeric	character
numeric	character

4.1.1.3 Mutations We recognize that project_start_date is always set to 1900-01-01 (due to unique analysis) in all cases. We remove this variable since it has no additional value for the further analysis.

project_start_date	count
1900-01-01	289065

We also decompose the date values for project_create_date and project_end_date into year, month, and day values so that they are already available for further analyses. Additionally, we expand the project data to include the hint for flagged projects and call the new variable **flag**. We set it to 0 per default initially.

pcd_year	pcd_month	pcd_day	ped_year	ped_month	ped_day
2014	12	5	2016	9	2
2014	12	5	2015	9	4
2014	12	5	2016	7	6
2014	12	5	2016	7	6
2014	12	5	1900	1	2
2014	12	5	1900	1	2

After adjusting the data types, we look at other anomalies in detail. For this purpose we use basic pattern analysis.

- proj_id: we know that there is a particular notation pattern for so called cancellation projects. Those are projects that were created based on manual errors in advance. Since we are interested to identify only "implausible" projects we leave out these projects. The patterns of the cancellation projects are 9999999.9 and 9999999.99, i.e. they contain a "." . In total, there are 10,886 cancellation projects which is only 3.7% of all projects.

9999999	9999999.9	9999999.99
278179	5583	5303

- channel: the pattern analysis also revealed that there are 129 channel entries with value "NULL" (character, not NA). Since this is a negligible proportion we filter out these rows.

AAAA	aaaaaaa_99
129	278050

.	Freq
NULL	129

4.1.1.4 Summary

Let us see the summary of the projects data after the previous steps.

Table 12: Head data of table projects (after mutation) (continued below)

proj_id	channel	correction	fix	project_create_date
Min. :5488372	channel_19: 31829	Min. :0.00000	Min. :0.0000	Min. :2014-12-25
1st Qu.:6219924	channel_06: 26413	1st Qu.:0.00000	1st Qu.:0.0000	1st Qu.:2016-10-24
Median :6419878	channel_11: 24180	Median :0.00000	Median :0.0000	Median :2018-07-31
Mean :6419641	channel_10: 22643	Mean :0.07499	Mean :0.2089	Mean :2018-07-16
3rd Qu.:6623656	channel_12: 22226	3rd Qu.:0.00000	3rd Qu.:0.0000	3rd Qu.:2020-04-20
Max. :6837942	channel_08: 22117	Max. :1.00000	Max. :1.0000	Max. :2021-12-17
NA	(Other) :128642	NA	NA	NA

Table 13: Table continues below

project_end_date	project_status	cust_account	invoice_account
Min. :1900-01-01	closed : 3013	Min. :1000034	Min. :1000034
1st Qu.:2016-11-30	work finished :268618	1st Qu.:8283099	1st Qu.:8305677
Median :2018-07-31	work in progess: 6419	Median :8424506	Median :8451640
Mean :2001-03-04	NA	Mean :8366332	Mean :8393838
3rd Qu.:2020-03-31	NA	3rd Qu.:8534157	3rd Qu.:8542959
Max. :2021-12-15	NA	Max. :8644557	Max. :8644557
NA	NA	NA	NA

Table 14: Table continues below

blocked_for_invoice	warranty_claim	proj_invoice_projid	pcd_year	pcd_month
Min. :0.000000	Min. :0.0000	Length:278050	Min. :2014	Min. : 1.000
1st Qu.:0.000000	1st Qu.:0.0000	Class :character	1st Qu.:2016	1st Qu.: 4.000
Median :0.000000	Median :0.0000	Mode :character	Median :2018	Median : 6.000
Mean :0.005082	Mean :0.2008	NA	Mean :2018	Mean : 6.402
3rd Qu.:0.000000	3rd Qu.:0.0000	NA	3rd Qu.:2020	3rd Qu.: 9.000
Max. :1.000000	Max. :3.0000	NA	Max. :2021	Max. :12.000
NA	NA	NA	NA	NA

pcd_day	ped_year	ped_month	ped_day	flag
Min. :1.000	Min. :1900	Min. : 1.00	Min. :1.000	Min. :0
1st Qu.:2.000	1st Qu.:2016	1st Qu.: 2.00	1st Qu.:2.000	1st Qu.:0
Median :4.000	Median :2018	Median : 6.00	Median :4.000	Median :0
Mean :3.768	Mean :2001	Mean : 5.91	Mean :4.067	Mean :0
3rd Qu.:5.000	3rd Qu.:2020	3rd Qu.: 9.00	3rd Qu.:6.000	3rd Qu.:0
Max. :7.000	Max. :2021	Max. :12.00	Max. :7.000	Max. :0
NA	NA	NA	NA	NA

4.1.2 Invoices data

4.1.2.1 Data structure The invoices data consists of the following columns:

- proj_invoice_projid - the project contract id (also join element for the project table)
- proj_invoice_id - the id for the project invoice
- order_account - the account for which the order was made
- invoice_date - the date on the invoice
- due_date - the due date on the invoice
- net_amount - the net amount
- line_discount - the line discount
- total_discount - the total discount
- markup - additional amount for particular services (e.g. approvals, road closures)
- tax - the tax amount
- gross_amount - the amount to be paid from the customer
- weight - the weight of the line item
- volume - the volume of the line item
- invoice_create_date - the create date of the invoice

- type - the type of the line item (employee or item)
- category - the category of the line item
- empl_item_detail - detailed information to the line item
- qty - the quantity of the line item (hours for category employee, volume of items for category items)
- line_amount - the amount of the line item

The dimensions and the first rows of the data file are as follows:

```
## [1] 1244079      19
```

Table 16: Head data of table invoices (continued below)

proj_invoice_projid	proj_invoice_id	order_account	invoice_date	due_date
PV-0001310	1004146	8025085	2015-02-13	2015-03-15
PV-0001310	1004146	8025085	2015-02-13	2015-03-15
PV-0001310	1004146	8025085	2015-02-13	2015-03-15
PV-0001310	1004146	8025085	2015-02-13	2015-03-15
PV-0001310	1004146	8025085	2015-02-13	2015-03-15
PV-0001312	1016179	8200892	2015-04-10	2015-05-10

Table 17: Table continues below

net_amount	line_discount	total_discount	markup	tax	gross_amount
6,087.70	0.00	913.16	0.00	413.96	5,588.50
6,087.70	0.00	913.16	0.00	413.96	5,588.50
6,087.70	0.00	913.16	0.00	413.96	5,588.50
6,087.70	0.00	913.16	0.00	413.96	5,588.50
6,087.70	0.00	913.16	0.00	413.96	5,588.50
3,194.02	0.00	74.40	0.00	249.57	3,369.20

Table 18: Table continues below

weight	volume	invoice_create_date	type	category
0.0000000000000000	0.0000000000000000	2015-02-13 10:18:46	EMPL	NULL
0.0000000000000000	0.0000000000000000	2015-02-13 10:18:46	ITEM	item_category_1
0.0000000000000000	0.0000000000000000	2015-02-13 10:18:46	ITEM	item_category_1
0.0000000000000000	0.0000000000000000	2015-02-13 10:18:46	ITEM	item_category_8
0.0000000000000000	0.0000000000000000	2015-02-13 10:18:46	ITEM	item_category_8
0.0000000000000000	0.0000000000000000	2015-04-10 07:54:39	EMPL	empl_category_3

empl_item_detail	qty	line_amount
NULL	NULL	NULL
P204C.01	1.00	2,361.90
P204C.01	1.00	3,645.80
P903A.14	1.00	69.00
P903D.04	1.00	11.00

empl_item_detail	qty	line_amount
5637145741	3.00	307.80

4.1.2.2 Initial data types The initial data types are shown in the following. We can see that some date data types are identified correctly as date; invoice_create_date was typed as POSIXct and POSIXt.

Table 20: Initial data types invoices (continued below)

proj_invoice_projid	proj_invoice_id	order_account	invoice_date	due_date	net_amount
character	character	numeric	Date	Date	character
character	character	numeric	Date	Date	character

Table 21: Table continues below

line_discount	total_discount	markup	tax	gross_amount	weight
character	character	character	character	character	character
character	character	character	character	character	character

Table 22: Table continues below

volume	invoice_create_date	type	category	empl_item_detail	qty
character	POSIXct	character	character	character	character
character	POSIXt	character	character	character	character

line_amount
character
character

As we can see most of all variables have character as data type and and thus are not typed correctly. We correct this in the following mutation.

4.1.2.3 Mutation Similar to the projects data we also decompose the date values for date variables, i.e. invoice_create_date, invoice_date, and due_date into year, month, and day. In addition to adjusting the data types, we remove the variable volume since it only contains the value 0.

icd_year	icd_month	icd_day	invd_year	invd_month	invd_day	dd_year	dd_month	dd_day
2015	2	6	2015	2	6	2015	3	1
2015	2	6	2015	2	6	2015	3	1
2015	2	6	2015	2	6	2015	3	1
2015	2	6	2015	2	6	2015	3	1
2015	2	6	2015	2	6	2015	3	1
2015	4	6	2015	4	6	2015	5	1

We also perform a pattern analysis for particular variables of the invoices data:

- proj_invoice_id: this variable is the central element for the invoice. If the entry is NULL or empty we will remove it since it does not provide any further information - it is isolated. Business experts confirmed this as data quality issue of the ERP system. In addition, we neglect the invoices with pattern "GS-" (55068 rows); those are credit notes, i.e. corrected invoices based on incorrect ones.

9999999	AA-9999999
1188669	55068

- category: We remove invoices where the category is NULL (83151 rows) since we cannot detect any hints for these invoices on items or employee hours. This will also be further investigated by the business experts of the company.
- In addition, we note that the variable empl_item_detail has 3188 unique entries. Various algorithms cannot handle such a large number, if they are defined as factor data types. Therefore, we create a unique id variable for this column based on the row_number function. The result is joined with the invoices table when needed so that only the id column can be used. A reference back to the concrete details is then possible without any problems.
- net_amount, line_discount, markup, tax, gross_amount, qty, line_amount, and total_discount: Since it may well be the case that invoices (presumably implausible ones) have no values and have therefore been flagged as incorrect, these project invoices are retained. In the numerical values we find some NA's. It will be useful for further processing to have complete values, thus these values are replaced by the value 0.

4.1.2.4 Summary

Here we find the summary of the invoices data.

Table 26: Head data of table invoices (after mutation) (continued below)

proj_invoice_projid	proj_invoice_id	order_account	invoice_date
Length:1131275	Length:1131275	Min. :1000034	Min. :2015-01-07
Class :character	Class :character	1st Qu.:8293426	1st Qu.:2016-11-03
Mode :character	Mode :character	Median :8435119	Median :2018-08-16
NA	NA	Mean :8382110	Mean :2018-07-24
NA	NA	3rd Qu.:8534799	3rd Qu.:2020-04-08
NA	NA	Max. :8644503	Max. :2021-12-17
NA	NA	NA	NA

Table 27: Table continues below

due_date	net_amount	line_discount	total_discount	markup
Min. :2015-01-08	Min. :-750.0	Min. :-308.72	Min. :-750.00	Min. :-993.100
1st Qu.:2016-12-03	1st Qu.: 0.0	1st Qu.: 0.00	1st Qu.: 0.00	1st Qu.: 0.000
Median :2018-09-16	Median : 196.8	Median : 0.00	Median : 0.00	Median : 0.000
Mean :2018-08-23	Mean : 257.5	Mean : 13.97	Mean : 14.63	Mean : -1.821
3rd Qu.:2020-05-08	3rd Qu.: 439.2	3rd Qu.: 0.00	3rd Qu.: 0.00	3rd Qu.: 0.000
Max. :2022-02-07	Max. :1000.0	Max. : 999.37	Max. : 998.10	Max. : 888.050
NA	NA	NA	NA	NA

Table 28: Table continues below

tax	gross_amount	weight	invoice_create_date	type
Min. : 0.00	Min. : 0.0	Min. : 0.000	Min. :2015-01-09	EMPL:293759
1st Qu.: 18.12	1st Qu.: 0.0	1st Qu.: 0.000	1st Qu.:2016-11-03	ITEM:837516
Median : 38.50	Median : 194.9	Median : 0.000	Median :2018-08-16	NA
Mean : 80.43	Mean : 259.1	Mean : 1.355	Mean :2018-07-24	NA
3rd Qu.: 91.94	3rd Qu.: 453.6	3rd Qu.: 0.000	3rd Qu.:2020-04-08	NA
Max. :998.32	Max. :1000.0	Max. :9870.000	Max. :2021-12-17	NA
NA	NA	NA	NA	NA

Table 29: Table continues below

category	empl_item_detail	qty	line_amount	icd_year
item_category_8:484999	P903A.14:207003	Min. :-320.000	Min. :-981.75	Min. :2015
empl_category_3:289127	P903D.04:203972	1st Qu.: 1.000	1st Qu.: 11.00	1st Qu.:2016
item_category_6:248235	153467 : 39073	Median : 1.000	Median : 62.64	Median :2018
item_category_1: 70323	P903E.01: 26360	Mean : 2.455	Mean : 103.70	Mean :2018
item_category_7: 10340	105435 : 25487	3rd Qu.: 1.620	3rd Qu.: 104.00	3rd Qu.:2020
item_category_2: 6507	P903A.01: 24627	Max. : 980.000	Max. : 999.97	Max. :2021
(Other) : 21744	(Other) :604753	NA	NA	NA

Table 30: Table continues below

icd_month	icd_day	invd_year	invd_month	invd_day
Min. : 1.000	Min. :1.000	Min. :2015	Min. : 1.000	Min. :1.000
1st Qu.: 4.000	1st Qu.:3.000	1st Qu.:2016	1st Qu.: 4.000	1st Qu.:3.000
Median : 7.000	Median :4.000	Median :2018	Median : 7.000	Median :4.000
Mean : 6.754	Mean :4.047	Mean :2018	Mean : 6.748	Mean :4.026
3rd Qu.:10.000	3rd Qu.:5.000	3rd Qu.:2020	3rd Qu.:10.000	3rd Qu.:5.000
Max. :12.000	Max. :7.000	Max. :2021	Max. :12.000	Max. :7.000
NA	NA	NA	NA	NA

dd_year	dd_month	dd_day	empl_item_detail_id
Min. :2015	Min. : 1.000	Min. :1.000	Min. : 1.0
1st Qu.:2016	1st Qu.: 4.000	1st Qu.:4.000	1st Qu.: 3.0
Median :2018	Median : 7.000	Median :5.000	Median : 33.0
Mean :2018	Mean : 6.949	Mean :4.551	Mean : 312.8
3rd Qu.:2020	3rd Qu.:10.000	3rd Qu.:6.000	3rd Qu.: 205.0
Max. :2022	Max. :12.000	Max. :7.000	Max. :3188.0
NA	NA	NA	NA

4.1.3 Flagged projects data

The flagged project data contains the project ids that have been flagged as implausible based on different business rules and by different business roles (e.g. sales manager, accounts receivable manager). They are subsequently used to set the flags in the project table to 1.

4.1.3.1 Data structure

The flagged_projects table consists only of the following column:

- proj_id - the id of the project which has been flagged as implausible project by business experts.

```
## [1] 72578     1
```

proj_id
5612765
6612402
6557712
6298438
6285985
6181352

4.1.3.2 Initial data types

The data type of proj_id is numeric.

4.1.3.3 Mutation

For the flagged projects data no mutation takes place, since this contains only the project ids which is typed correctly.

4.1.3.4 Summary

In total, we have 72578 flagged projects. However, for some projects we have multiple reasons for being flagged. The amount of unique projects flagged is 47868.

proj_id
Min. :2474553
1st Qu.:6172258
Median :6333364
Mean :6329213
3rd Qu.:6504610
Max. :8464344

4.2 Preprocessing

As mentioned above, we now expand the project data to include the flagged projects (table flagged_projects). These were created in a previous step based on various business rules and are used to classify plausible (flag = 0) and implausible (flag = 1) projects.

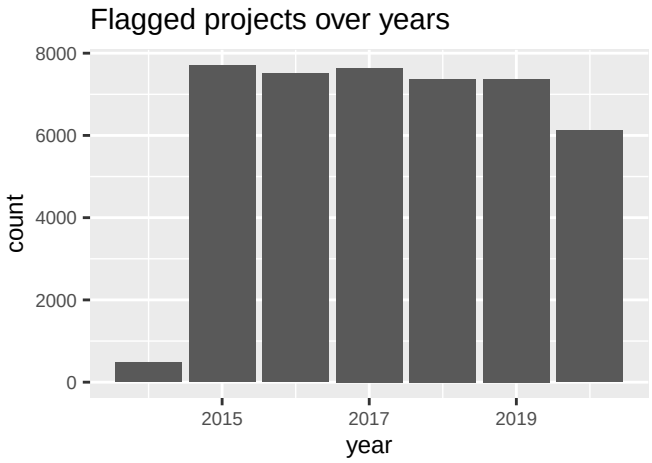
```
# flag projects in list of all projects based on flagged proj_ids
for (i in projects$proj_id) {
  if (i %in% unique_flagged_project_ids$proj_id) {
    projects$flag[projects$proj_id == i] <- 1
  }
}
```

After flagging the projects we have the following summary:

0	1
233862	44188

We can see that 44188 projects have been flagged as implausible which is about 0.1589211 of all projects (278050).

The following plot shows how these flagged projects are distributed over the years: Based on the rules to find implausible projects to flag between 6,000 and 8,000 projects have been flagged for the years 2015 to 2020. However, we recognize that in 2014 we have only a few flagged projects. This is due to the fact that we have only a few data entries for this year. Later on we will remove data from this year (since it is not representative for the total development) based on further business reasons.

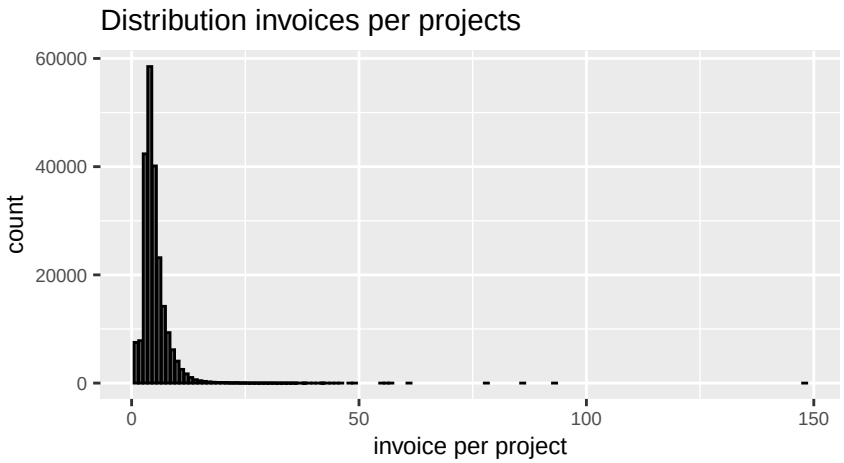


4.3 Data visualization

Before we visualize the data, we joined the service projects with their related invoices to include all potential predictors in terms of invoice details, lastly we removed duplicates.

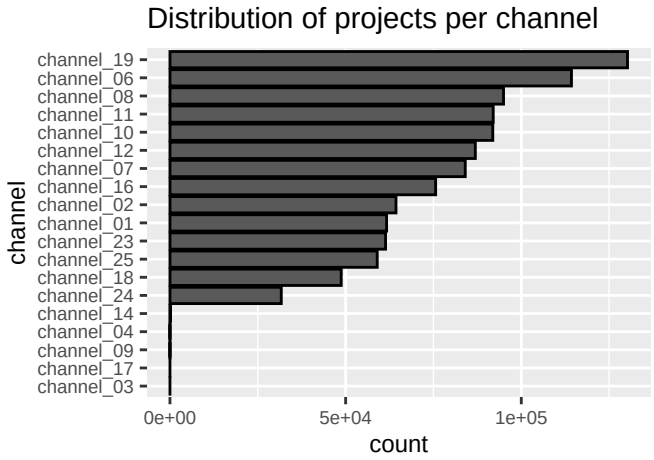
4.3.1 project_id

After the inner join we get 220883 distinct proj_ids and 222722 distinct proj_invoice_ids. As already mentioned, this means that usually we have one invoice per project. However, we can find projects with multiple invoices as we see in the plot. Interestingly, we see that there are some projects which have significant more invoices then around 25. The maximum is one project with almost 150 invoices which seems quite unrealistic and needs to be analyzed further in the ERP system.



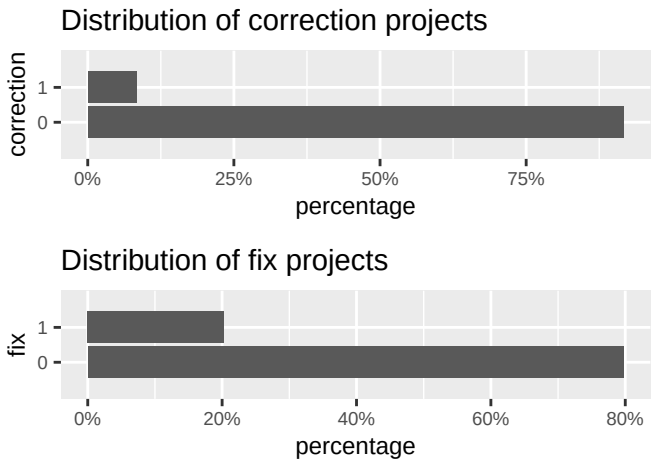
4.3.2 channel

The plot **Distribution of projects per channels** shows that we have 19 channels. 5 channels have less than 110 projects, 2 of them below 5 projects. These channels could be further investigated within the ERP system to validate if those are incorrect data entries.



4.3.3 correction and fix projects

Investigating the correction and fix projects we can see that around 90% of the projects are regular (not correction) projects and around 80% are time and material (not fixed) projects.

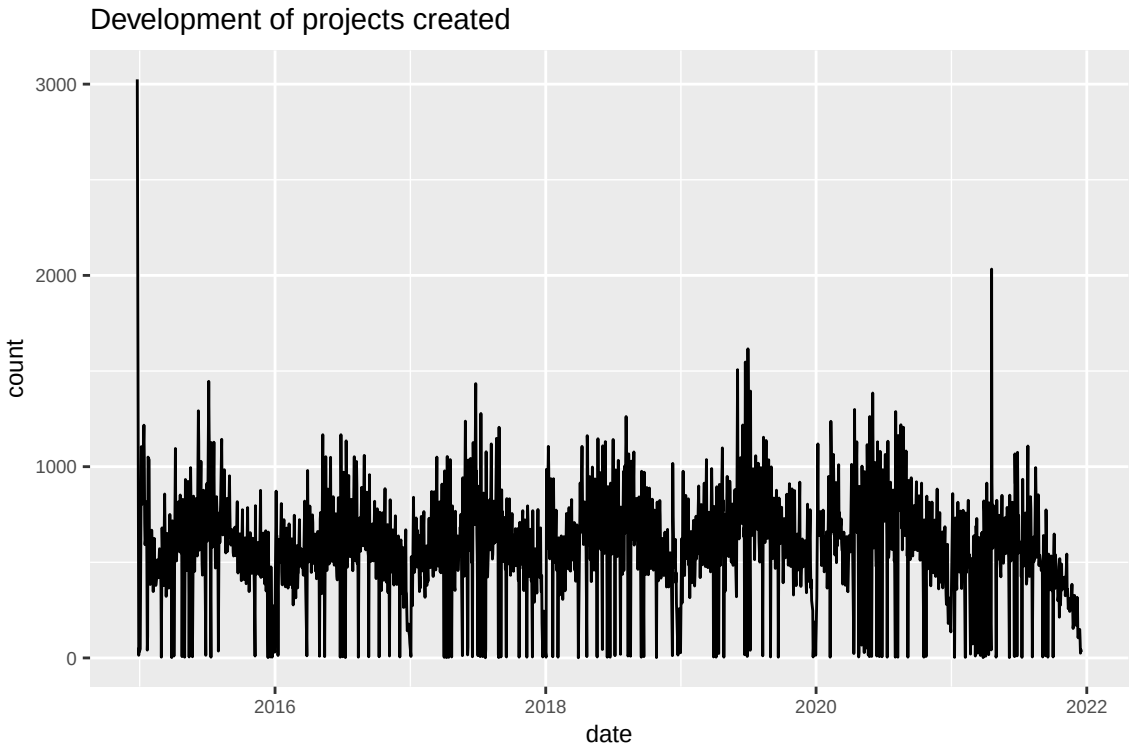


4.3.4 project_create_date and project_end_date

In terms of time following question could arise:

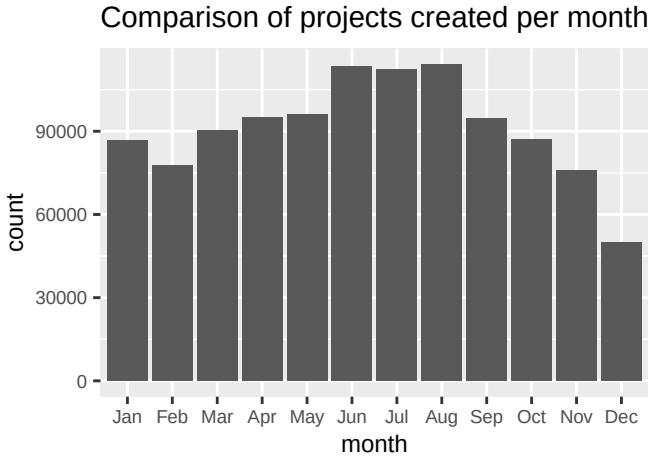
- When are the most projects created during the year?
- What is the "high season"?
- When do they end typically?

These and other questions are answered here. Let us look on the total temporal development first. We can see that we have a enormous peak in the first time period.



A closer look shows that this is the 25th of December in 2014 which seems quite uncommon to create projects (since it is a vacation day). The question to the business department revealed that there was the migration date from the former system - and all the projects so far have been booked on this date. Based on this and the fact that we have only a few aggregated data entries for 2014, we exclude the year 2014 from now on and start from 2015.

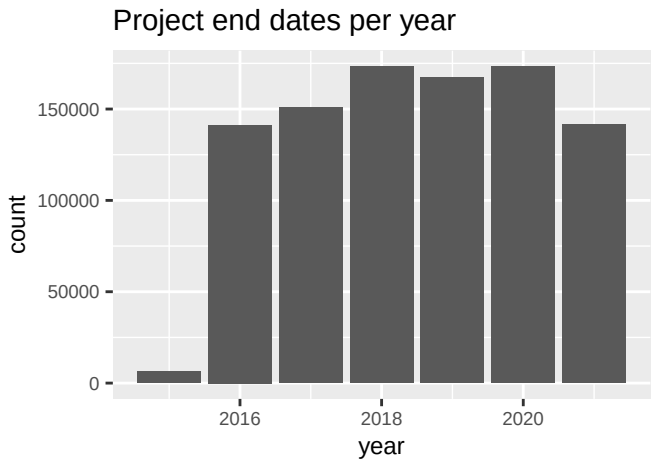
We can also see that there is a seasonality which we could investigate further. In what way are the months differently for each year? We can see that the high season of creating projects seems to be June till August, then if shrinks to the years end. Interestingly the January seems uncommon since it is higher than December and February. Reason could be that projects that are not "worth" it to start in the old year will be shifted to the next year.



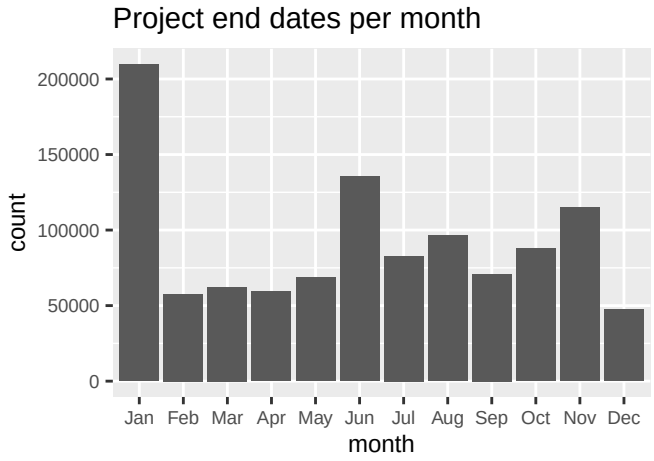
It looks a bit different when we look at the project end date. First of all, it is noticeable that many projects do not seem to have an end date recorded and it is set to the default value of 1900.

ped_year	count
1900	139897
2015	6414
2016	141313
2017	150948
2018	173250
2019	167268
2020	173192
2021	141423

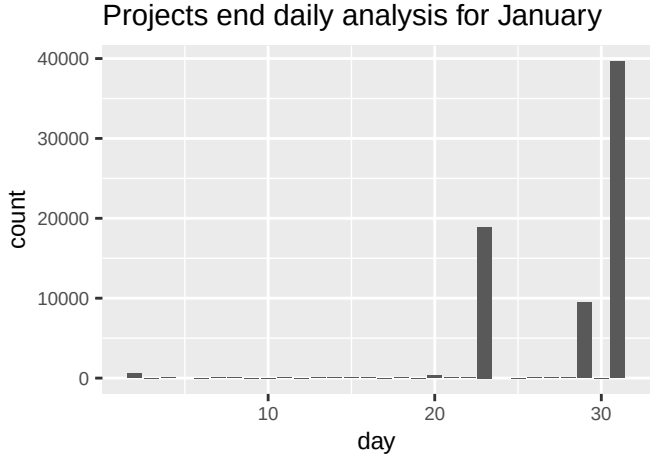
When we filter out these default values and show the plot **Project end dates per year**, we see: With the exception of 2015 - which in turn is due to the aforementioned introduction of the new system - there are no anomalies here.



Let us check if there are any anomalies in the months for the project end dates. We can see that by far the most projects are completed in January, followed by June and November.



If we look at January on a daily basis, we see that the most projects have the 31st as the end date. This may be related to the fact that this is possible with the month-end and years-end closing term and possibly adding value to the previous year.

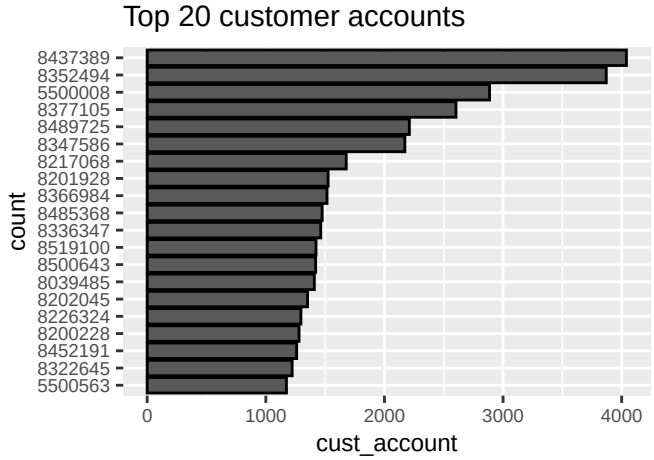


4.3.5 project_status

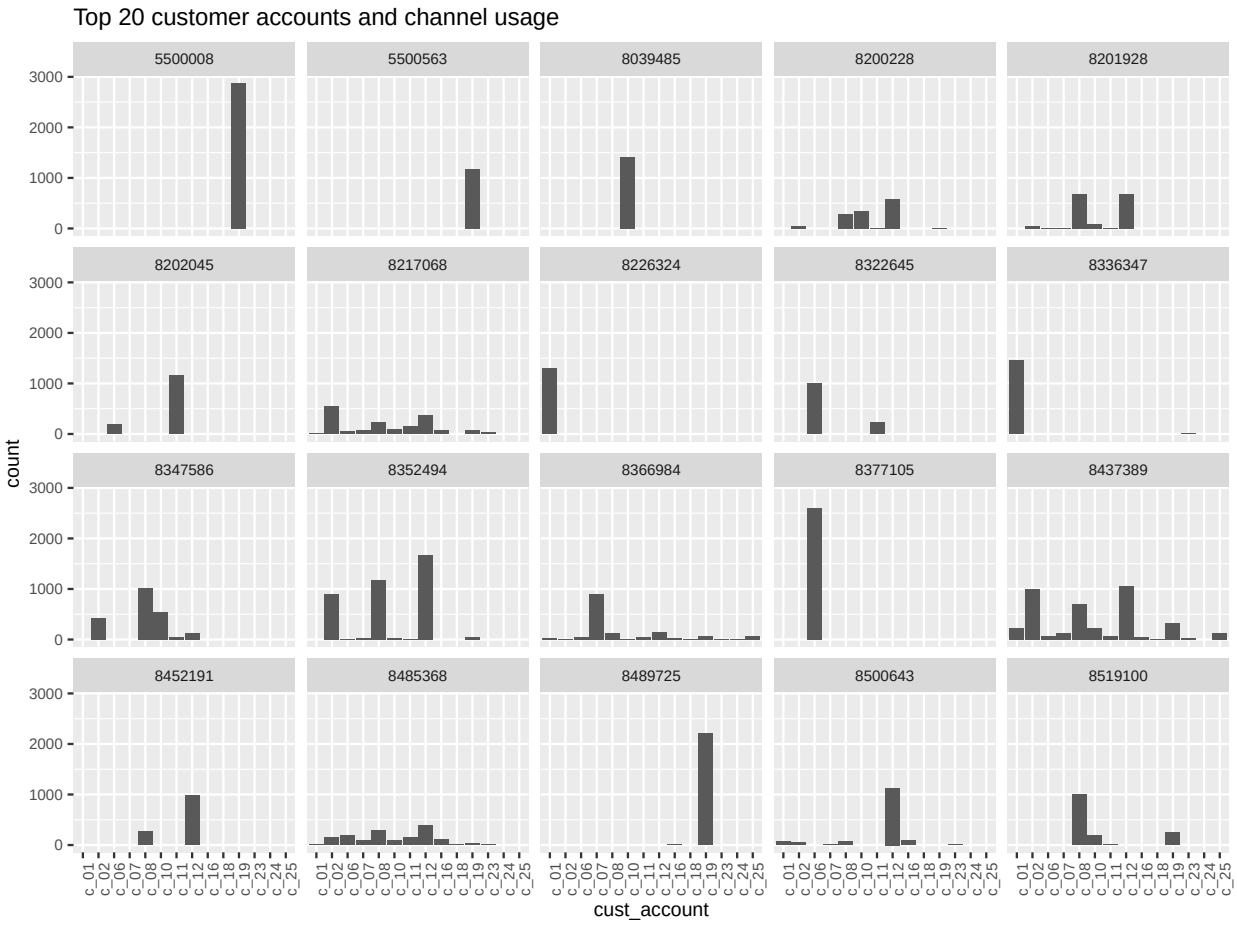
The view of the project status is very plausible: most projects are "work finished". Only a few are in "work in progress" status. However, it is surprising that the projects where the work has been finished are not set to the status "closed". Maybe this is a state which is not required anymore.

4.3.6 cust_account

The customer accounts can be analyzed as follows: There are a total of 84764 unique customer accounts to which invoices have been sent. Thereby, some customers are particularly active, which can be seen from the plot **Top 20 customer accounts**.

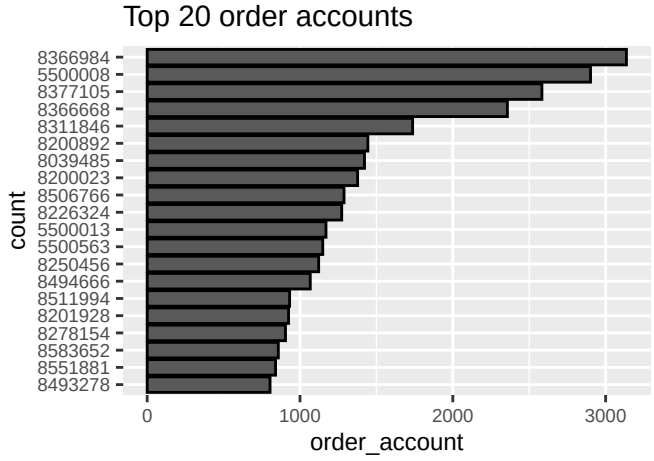


Additionally, it seems that the top 20 customers usually handle their services through only a few channels. However, there are some who also use significantly more channels, see **Top 20 customer accounts and channel usage**. Further analyses should be carried out to find out the reasons for this in discussion with the business experts.

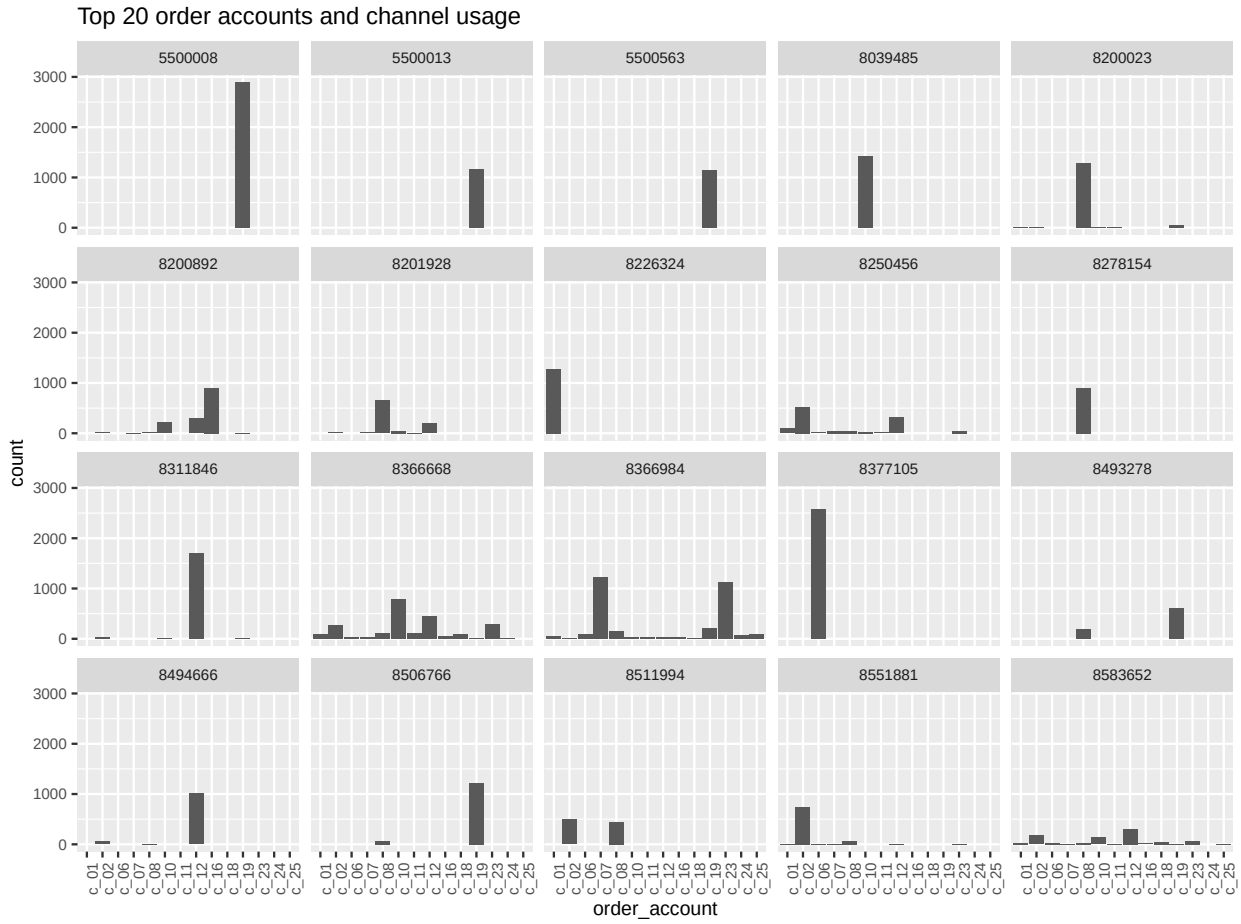


4.3.7 order_account

In this industry of construction, even for services, it can always happen that there are differences between the order account and the invoice account. For example, a property management company places the order, but the invoice itself goes to the apartment owner. The top 20 order accounts are as follows:

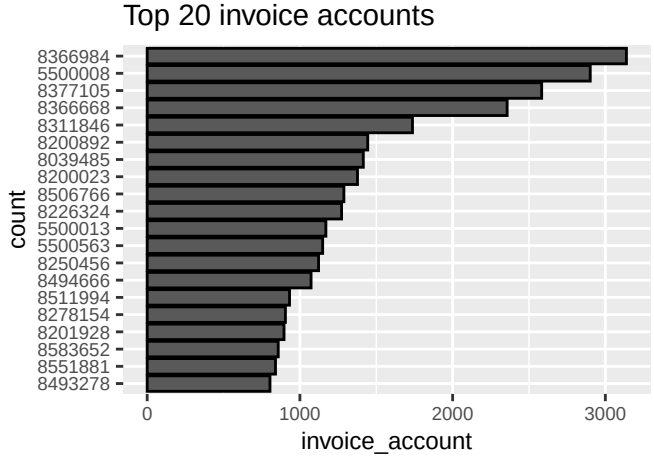


Again, we look at how these order accounts are distributed across the channels and see a similar result as above with cust_accounts.

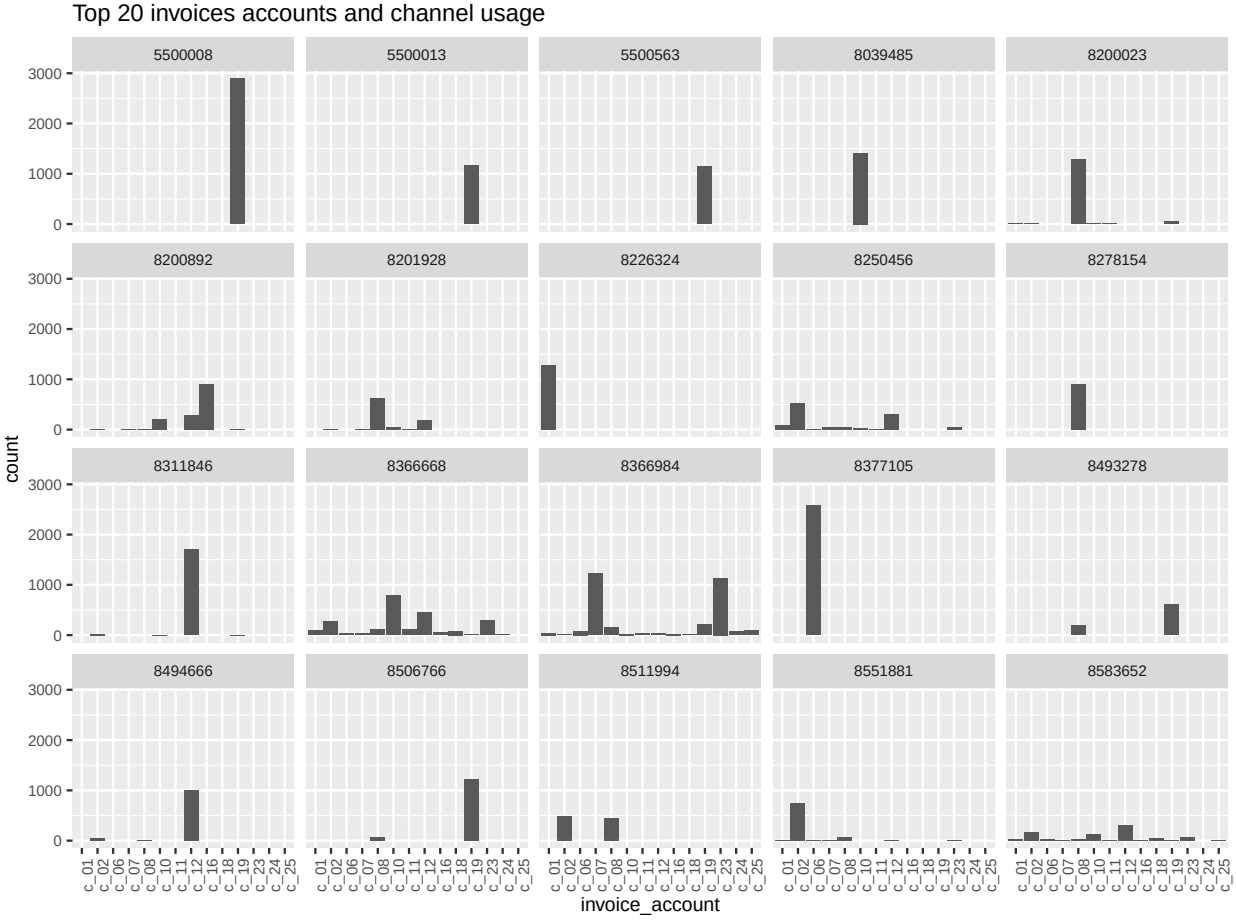


4.3.8 invoice_account

The picture for the invoice accounts is similar: Of the top 20 customer accounts, 7 are also in the top 20 invoice accounts.



Here is the overview of the distribution of the invoice accounts to the channels. You can see a similar picture as with the customer and order accounts.



4.3.9 blocked_for_invoice

The blocked for invoice status is assigned to customers whose payment status is being checked and for whom a new project must not be opened. However, this concerns only a very small part, as we can see from the table. There are only 6809 projects blocked for invoicing.

blocked_for_invoice	count
0	1086896
1	6809

4.3.10 warranty_claim

The warranty claims show four different statuses. The higher the number, the higher the critically of the case. Further details on the statuses are not available from the business.

warranty_claim	count
0	1006238
1	40230
2	893
3	46344

4.3.11 proj_invoice_projid and proj_invoice_id

The variable proj_invoice_projid represents only a legal basis for a project and is not analyzed further. Only the pattern is checked. It can be seen that this has a clear and unambiguous structure - which is also important for the connection between project and project invoice.

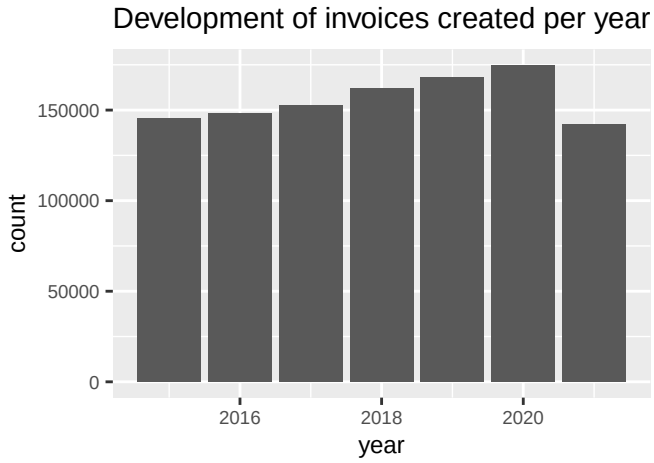
AA-9999999
1093705

The same applies to the analysis of the variable proj_invoice_id. Here we will also only check the pattern to see if there are any anomalies. This is not the case after the cleanup above.

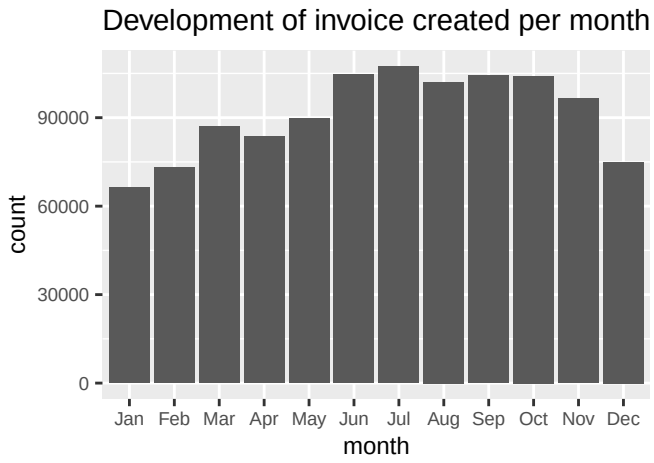
9999999
1093705

4.3.12 invoice_create_date

When were the invoices generated over time between the years? Are there any "highlights" at the monthly level? We see an increase of invoices created on a yearly perspective from 2015 to 2020. However, in 2021 the level was below the invoices created level from 2015.

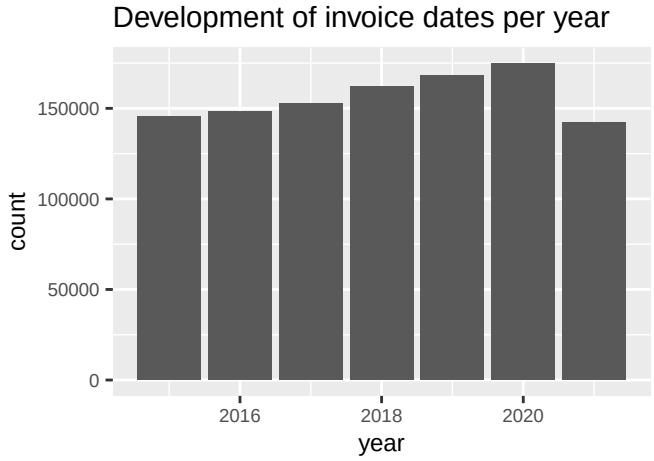


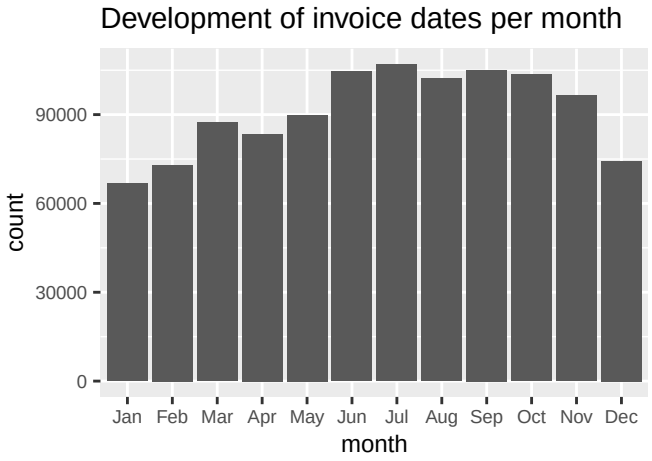
On a monthly basis the plot shows that we have a high season during summer what makes sense in this construction industry, also regarding after sales services.



4.3.13 invoice_date

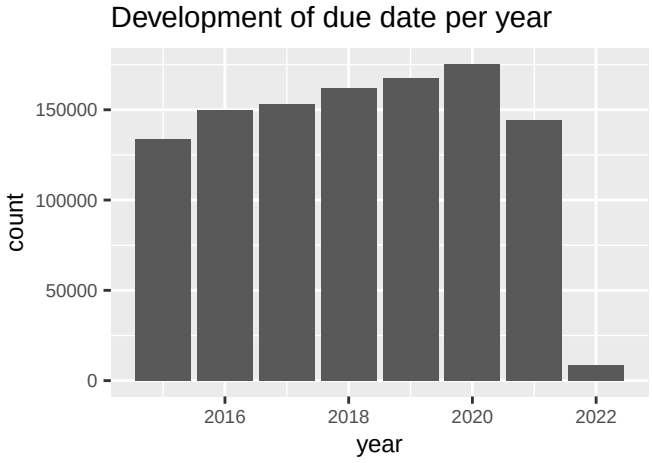
What is the typical invoice date for the customers perspective? Can we detect something unusual? Since it would be irritating if the time period between creation date and invoice date - and thus the pattern - would diverge much, it is understandable that the consideration of the invoice dates provides a very similar picture to the creation date of invoices; for years as well as for months.



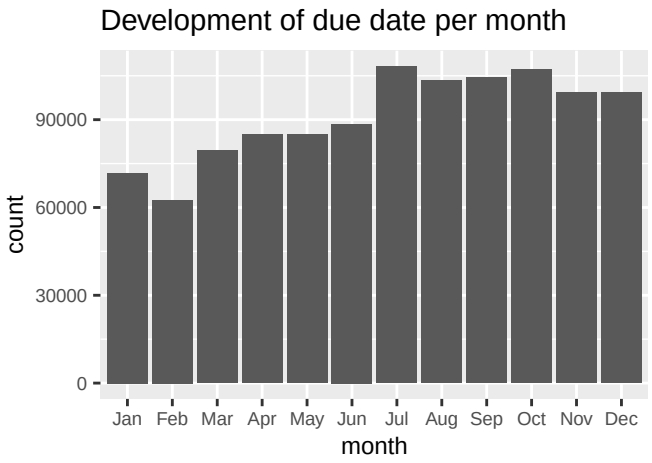


4.3.14 due_date

When looking at the due date, it is noticeable that some invoices are already due for the year 2022. In addition, it can be seen that fewer due dates were assigned in 2021 compared to the previous years (except 2015).



In the months of the due dates, one can clearly see the tendency that the invoices are usually set from the middle of the year so that they are due before the end of the year.

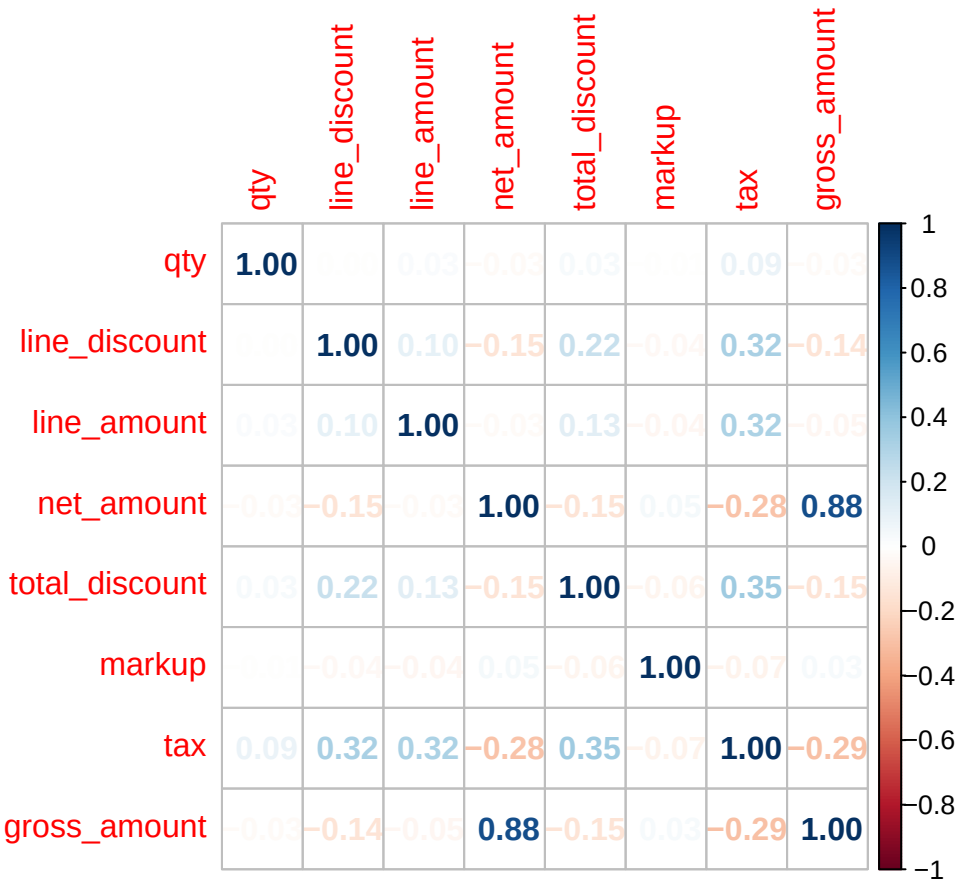


4.3.15 invoice amounts

The relationships regarding the invoice calculation are as follows:

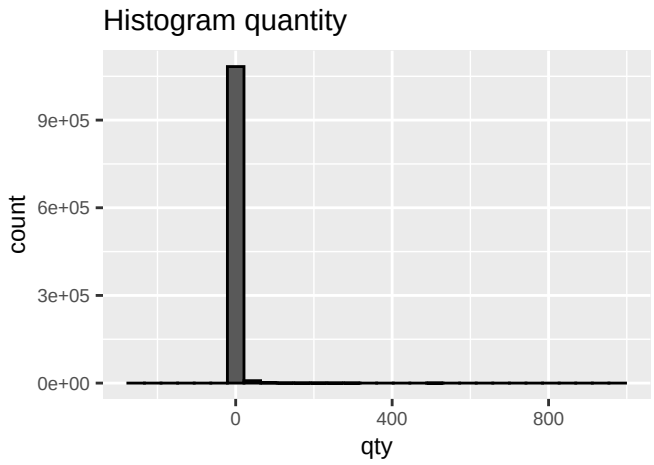
- The (a) quantity * base amount (not existing in the data) - (b) line discount = (c) line amount
- sum over all line amounts = (d) net amount
- net amount - (e) total discount + (f) markup + (g) tax = (h) gross amount

4.3.15.1 General analysis First, we want to provide a general overview of the key figures of the invoices. The correlation plot helps us to do this. We get a high positive correlation when looking at the net amount with the gross amount. This is understandable, since the gross amount is calculated on the basis of the net amount plus the markups and the taxes. All other variables do not have high correlations.

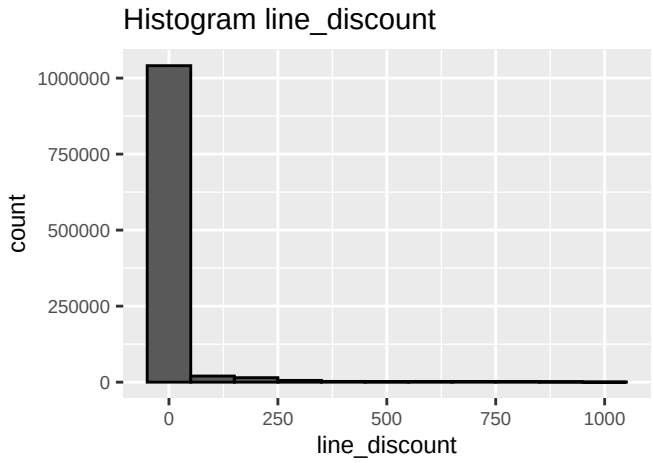


4.3.15.2 (a) qty It is quickly apparent that the quantity is very clearly distributed: The summary shows that the 1st quartile, the median, the average and the 3rd quartile are between 1 and 3. The outliers with a minimum of -250 and a maximum of 980 are apparently very unusual values, as the diagram shows.

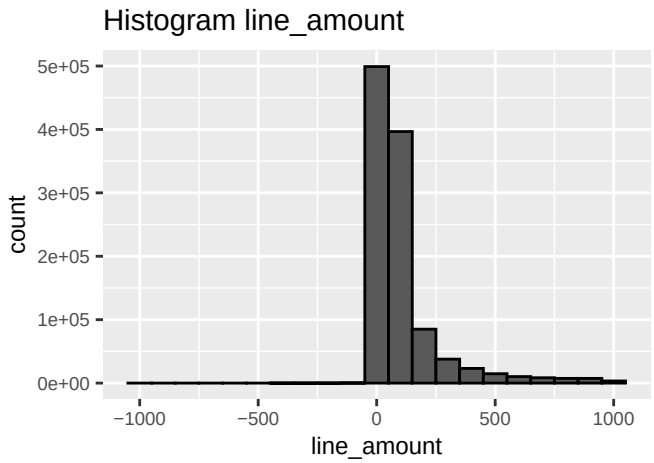
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-250	1	1	2.442	1.62	980



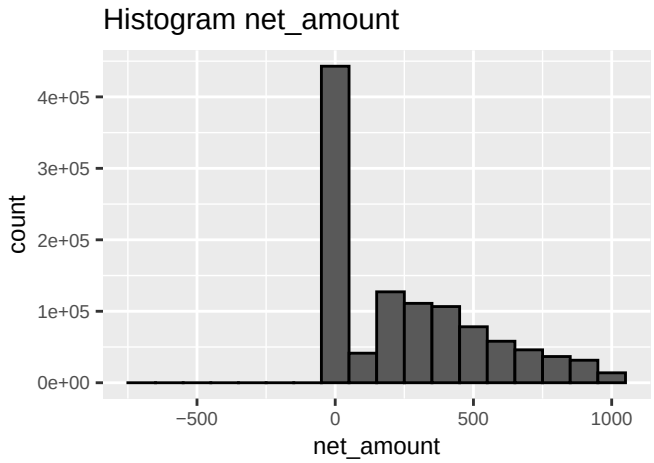
4.3.15.3 (b) line_discount The line discount, which leads to the line amount after deduction on line item level, is distributed similarly to low values as the quantity. The histogram provides a strongly skewed distribution. No line discount was given more than one million times.



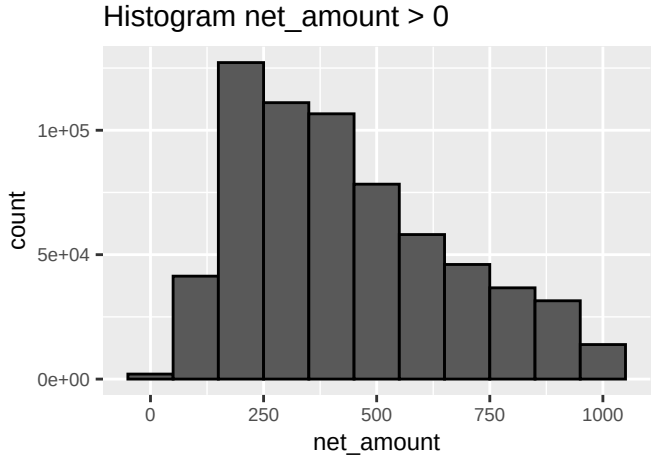
4.3.15.4 (c) line_amount The line amount represents the amount that is on the line item level on the invoice. We see that we have negative as well as positive values, showing that there seem to be correction items on the invoices. The most frequent values are between 0 and 100.



4.3.15.5 (d) net_amount The net amount could show a normal distribution, but this is not obvious at first glance.

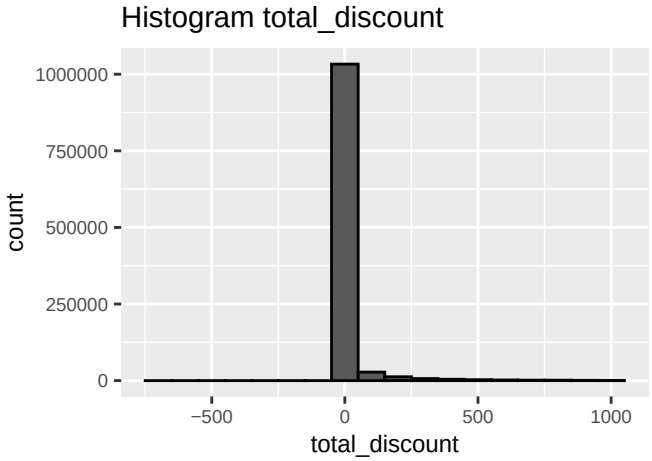


The 0 and negative amounts hide the view. If we remove them, we get a slightly different perspective.



Now we calculate the mean and the standard deviation and see that even this adjustment of the data still shows a very wide distribution of the net amount (mean = 259.1227096, sd = 277.6899667).

4.3.15.6 (e) total_discount The total discount is applied to the entire invoice and deducted at the end of the net amount. The correlation analysis showed that there is no statistically meaningful relationship between it and the line amount. This means that the sales staff allocate them independently of each other.

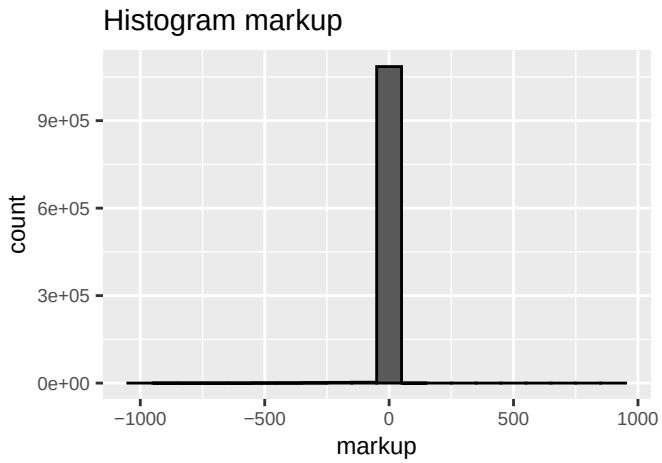


The total discount is also awarded similarly infrequently as the line discount. Again, there are over 1 million entries that do not award a discount. Let us have a look on the top 10 of the total discounts.

Selecting by count

total_discount	count
0	1022518
65	120
79.4	120
88.4	109
450	103
39	87
9.6	78
30	74
60	70
71.7	66

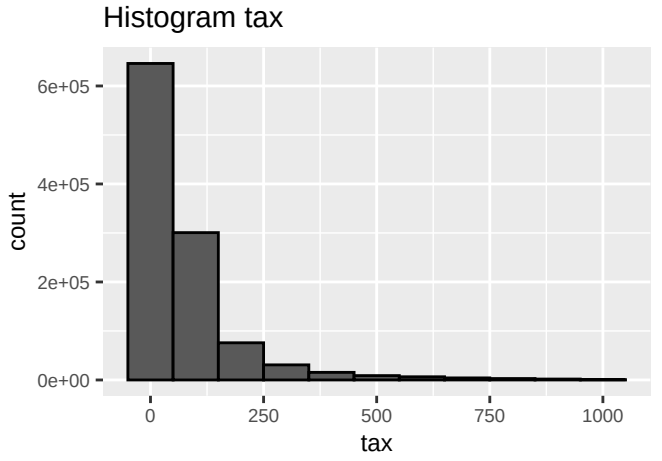
4.3.15.7 (f) markup Markup is the amount on an invoice that is either added or deducted. Most often, these are fees for road closures or permits that must be applied for in order for the service to be provided.



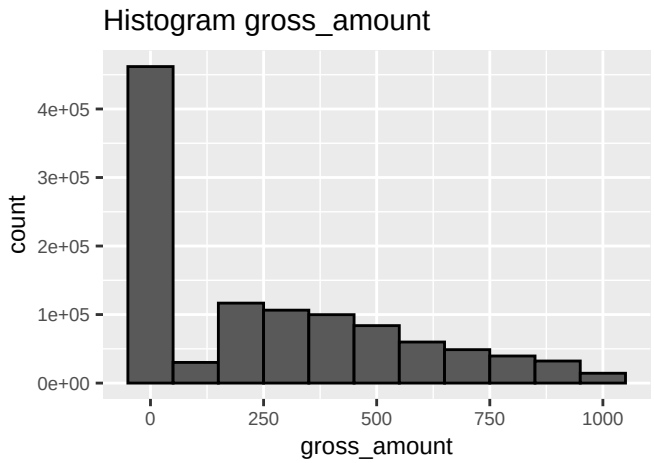
The summary shows that there are large outliers, but that the 1st quantile, the median, the mean and the 3rd quantile values are close to zero.

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-993.1	0	0	-1.748	0	888

4.3.15.8 (g) tax In the case of the tax, we see that we, of course, have no negative tax amount. Since we have many small values for the net amount, the proportional tax amount relates similar.

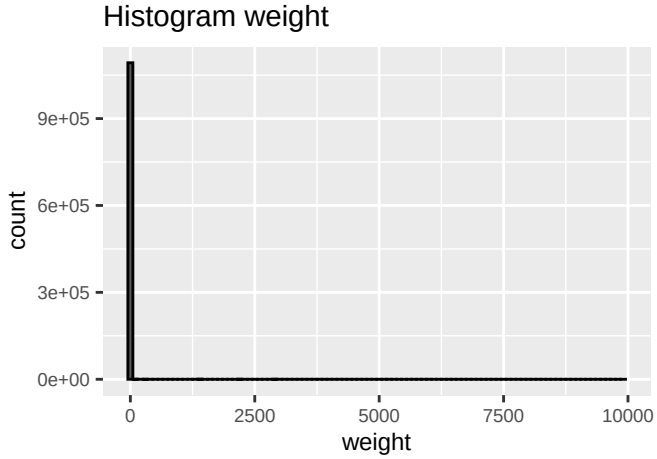


4.3.15.9 (h) gross_amount The gross amount is the final value of the invoice to the customer. Since this is a calculated result based on the net amount, which has already been analyzed in more detail, we do not expect any surprises in terms of values. The plot clearly underlines this and the high correlation as shown above betwenn gross amount and net amount.



4.3.16 weight

The variable weight reflects a variable that is not used very often: minimum, 1st quartile, median, and the 3rd quartile are 0. The mean value is only slightly above 0 due to the large outlier, which is almost 10,000.

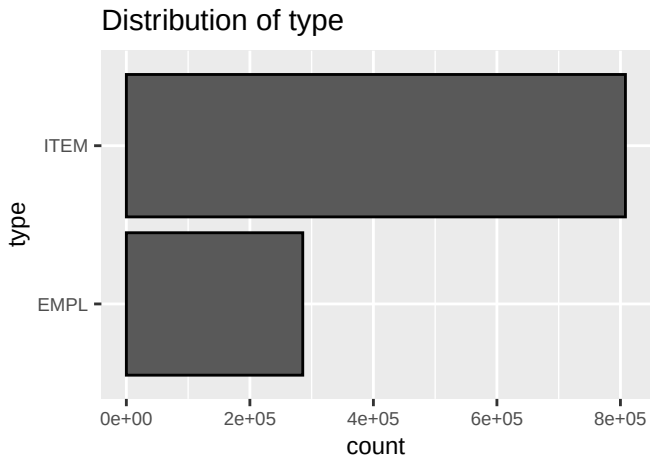


Due to the low variability, we will remove this variable from here on.

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0	0	0	1.373	0	9870

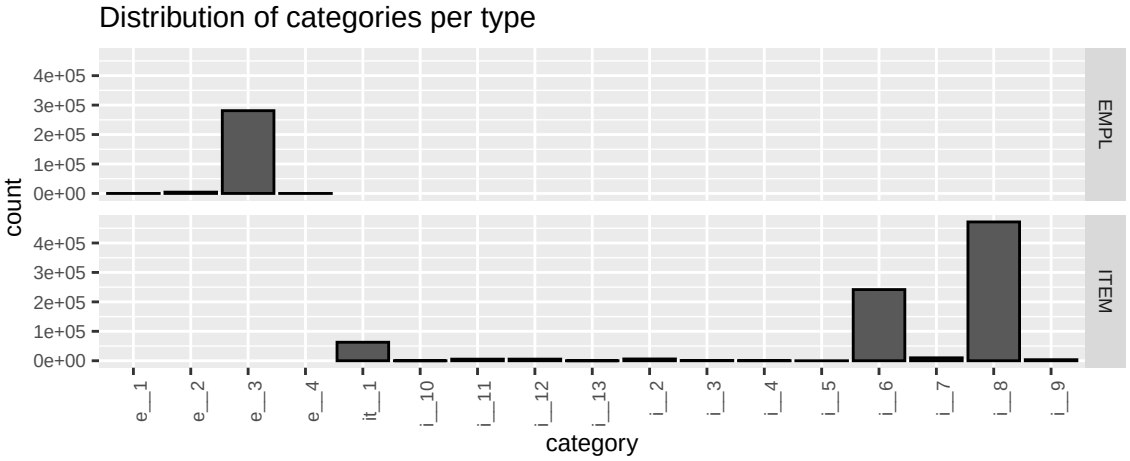
4.3.17 type

The variable type indicates whether the line item is a product or an hour worked by an employee. During the service, according to the plot, about 3x more item entries are listed on the invoices than labor hours. This is easily understandable, since the items also include numerous spare parts and other small and cleaning materials.



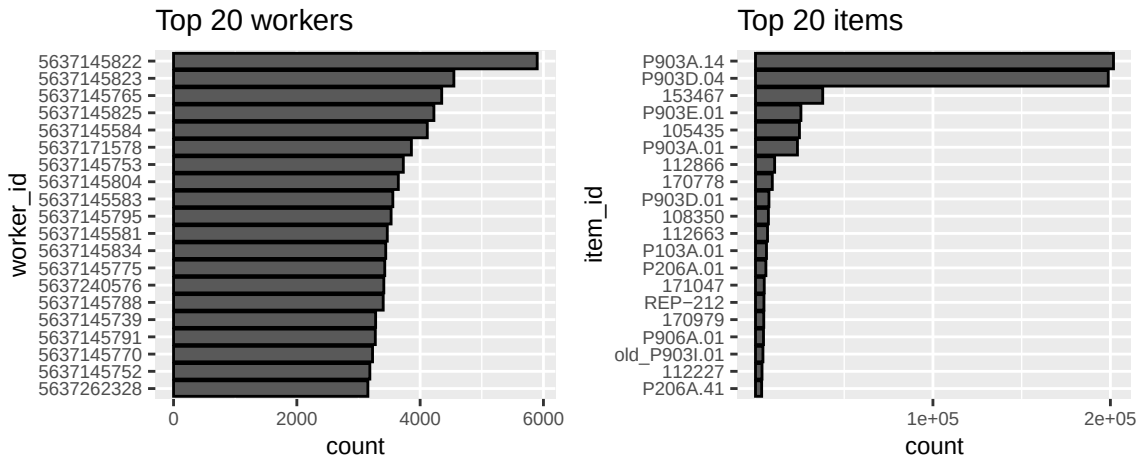
4.3.18 category

When analyzing the category, we find that on the one hand there are no data errors: The categories are clearly assigned to the types described above. In addition, the plot shows that for the employee categories, category 3 is the most frequent on the invoice; in contrast, for the items, category 8 is at the top of the list, followed by categories 6 and 1.



4.3.19 empl_item_detail

In total, we find 3178 different detail information about the employee or items in the data. The top 20 per type are as follows.



Among the workers, the distribution of the top 20 is quite similar, with only one employee standing out a bit. In the case of the items, on the other hand, there are two in particular that appear on the invoices with particular frequency. The investigation of the invoices shows that these are items that are usually found on the invoice: the general fee for travel and the general fee for small and cleaning material. These items are rarely removed from the invoice.

5 Modeling approach

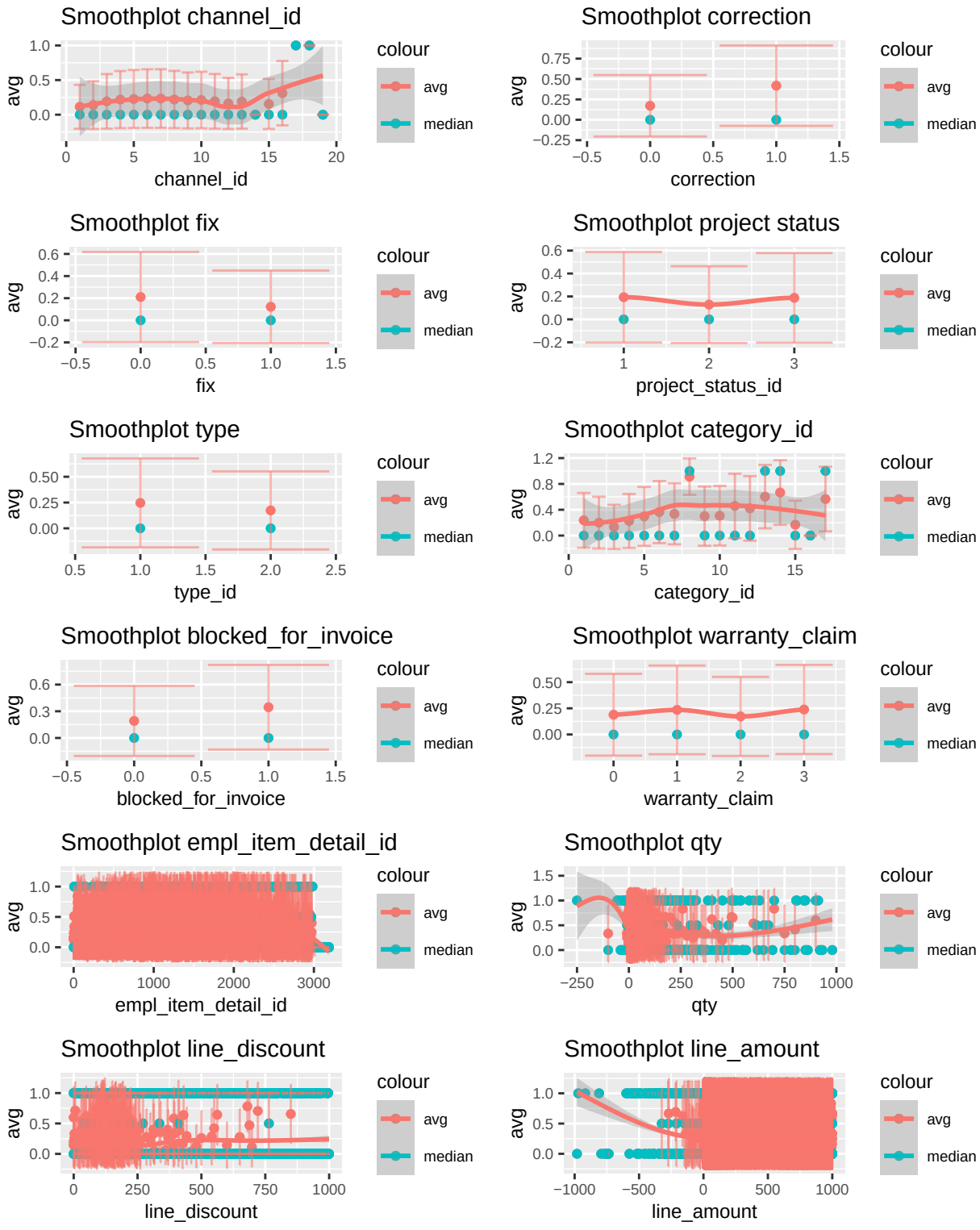
In order to find the best performing model we will use the predictors as described in the prior section individually or in combination and apply different methods to train the models.

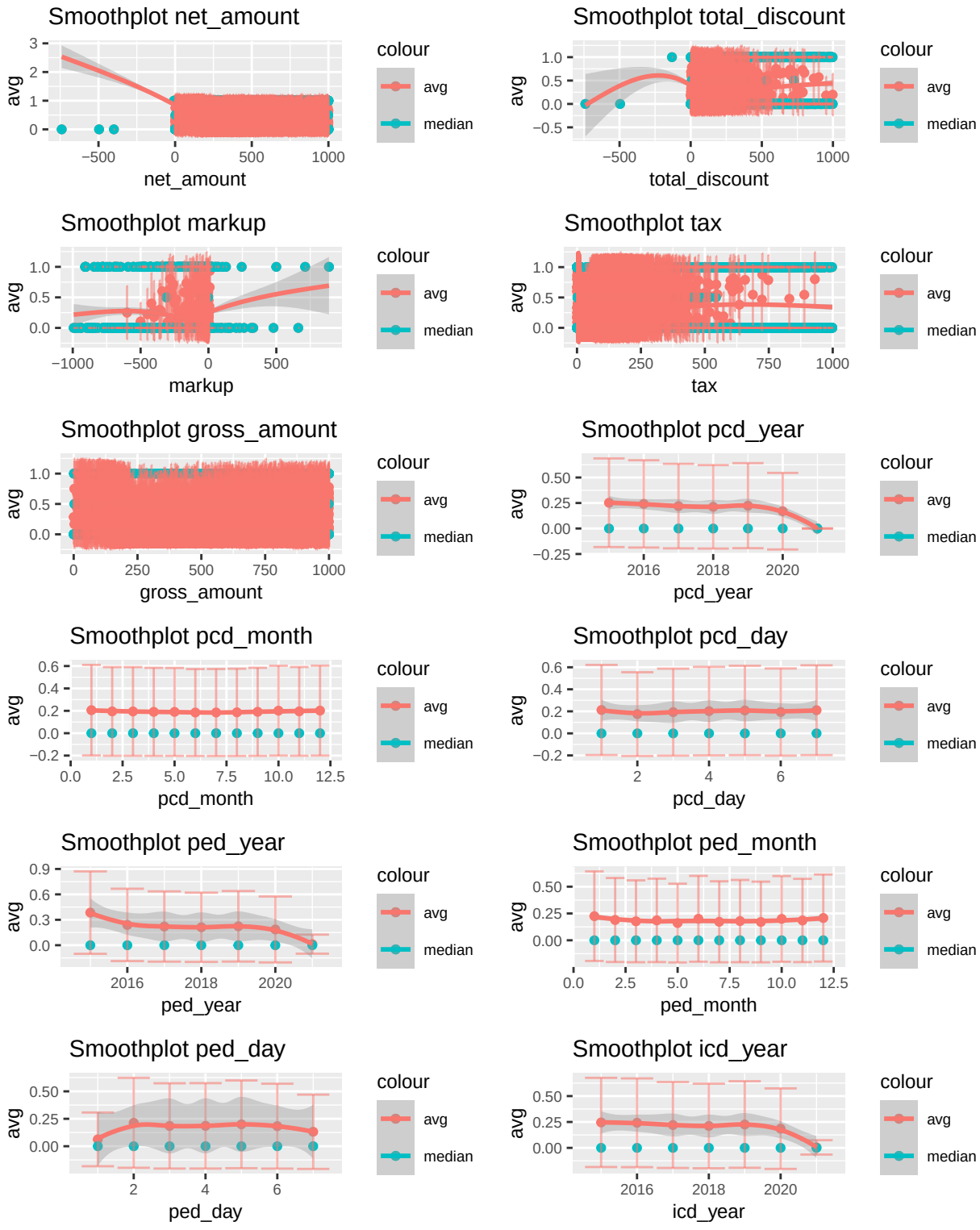
5.1 Initialization of modeling

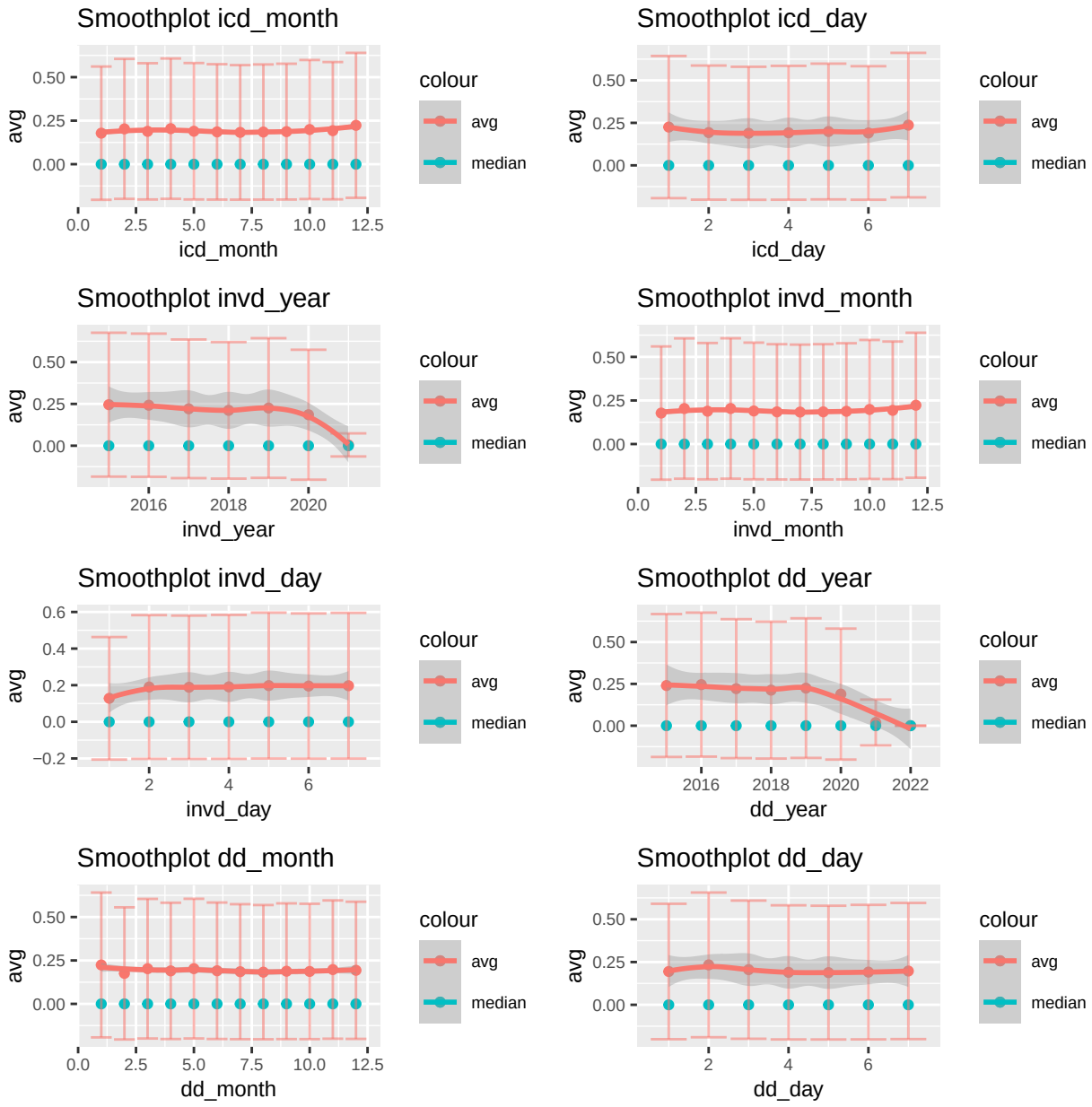
Since the date values for the variables project_end_date, invoice_date, due_date, invoice_create_date are not suitable for prediction, they will be removed. We also remove the variables proj_id, proj_invoice_projid and proj_invoice_id; they are unique or almost unique values respectively that do not represent additional information for our purpose. As mentioned above also the variable empl_item_detail is neglected, since we work with the unique id instead of the character variable.

5.2 Smoothplots for single predictors

To dive deeper into the data plots are created that show the average, the median and the standard error over each potential predictor. In addition to this a smoothplot is added to get a first idea about fitting based on single predictors using LOESS as a fitting algorithm. Since we have multiple smoothplots, we focus on those being relevant. The plots come first, followed by the findings for the plots.


Smoothplot channel_id
avg
channel_id
colour
avg
median
Smoothplot correction
correction
Smoothplot fix
fix
Smoothplot project status
project_status_id
Smoothplot type
type_id
Smoothplot category_id
category_id
Smoothplot blocked_for_invoice
blocked_for_invoice
Smoothplot warranty_claim
warranty_claim
Smoothplot empl_item_detail_id
empl_item_detail_id
Smoothplot qty
qty
Smoothplot line_discount
line_discount
Smoothplot line_amount
line_amount


Smoothplot net_amount
avg
net_amount
Smoothplot total_discount
total_discount
Smoothplot markup
markup
Smoothplot tax
tax
Smoothplot gross_amount
gross_amount
Smoothplot pcd_year
pcd_year
Smoothplot pcd_month
pcd_month
Smoothplot pcd_day
pcd_day
Smoothplot ped_year
ped_year
Smoothplot ped_month
ped_month
Smoothplot ped_day
ped_day
Smoothplot icd_year
icd_year
colour
avg
median



In total, we analyzed 32 smoothplots whereas 10 show a higher variability. All these plots are potential predictors for detecting implausible projects. For the other variables their smoothplots showed no or little variability. Thus, we do not consider them in term of the following steps. In addition, we remove the variable gross_amount since it is highly correlated with net_amount. The final selected variables are as follows:

- channel_id
- category_id
- net_amount
- line_discount
- total_discount
- markup
- tax
- qty

- line_amount
- empl_item_detail_id

5.3 Feature scaling - normalization

Since we have some numerical variables which have quite different ranges, we will normalize them in the following step at this point. Of course, there are options to apply the normalization within the training of the model depending on the function. However, we do this as a central step to have the comparable basis. An optimization for the gradient descent based algorithms (e.g. logistic regression) should already be done at this point. For the tree-based algorithms (e.g. random forest) this optimization has no influence; they are invariant to the scale of the features. The result after normalization looks like this, for example.

Table 44: normalized data (continued below)

channel_id	category_id	net_amount	line_discount	total_discount	markup	tax
0	0	0.5262	0	0.4258	0.5279	0.01407
0	0.0625	0.5262	0	0.4258	0.5279	0.01407
0	0.125	0.5262	0	0.4258	0.5279	0.01407
0	0.125	0.5262	0	0.4258	0.5279	0.01407
0	0.125	0.5262	0	0.4258	0.5279	0.01407
0	0	0.6171	0	0.4258	0.5279	0.02674
0	0.0625	0.6171	0	0.4258	0.5279	0.02674
0	0.0625	0.6171	0	0.4258	0.5279	0.02674
0	0.1875	0.6171	0	0.4258	0.5279	0.02674
0	0.125	0.6171	0	0.4258	0.5279	0.02674

qty	line_amount	empl_item_detail_id
0.2039	0.5363	0.08001
0.2041	0.4982	0.07436
0.2041	0.5403	0.0003138
0.2041	0.501	0.0006275
0.2024	0.4899	0.0006275
0.2047	0.5908	0.0273
0.2065	0.4969	0.005962
0.2072	0.5088	0.006903
0.2049	0.5031	0.1067
0.2041	0.5403	0.0003138

5.4 Feature selection

We examine the near zero variation of the variables to check the potential reduction for the variable list. The analysis shows that there is a total of six variables that have a near zero variation. These we will be removed for the next steps.

	freqRatio	percentUnique	zeroVar	nzv
net_amount	72.38	4.065	FALSE	TRUE
line_discount	694.1	0.731	FALSE	TRUE
total_discount	8342	0.6227	FALSE	TRUE
markup	1752	0.1391	FALSE	TRUE

	freqRatio	percentUnique	zeroVar	nzv
tax	34.83	2.675	FALSE	TRUE

5.5 Data splitting

The normalized data set is split into a train and a test set. Finally, we want to predict implausible projects for the year 2021 using the validation set. At first, we use the time split for 2020 and earlier projects by project_create_date, then we split the data from 2015 to 2020 based on the createDateParition function and take 90% for train data and 10% for test data. In total, we have 5 predictors in the model and two levels for the label flag to be predicted.

6 Train and compare different models

In terms of cross validation we use a 10-fold cross validation procedure.

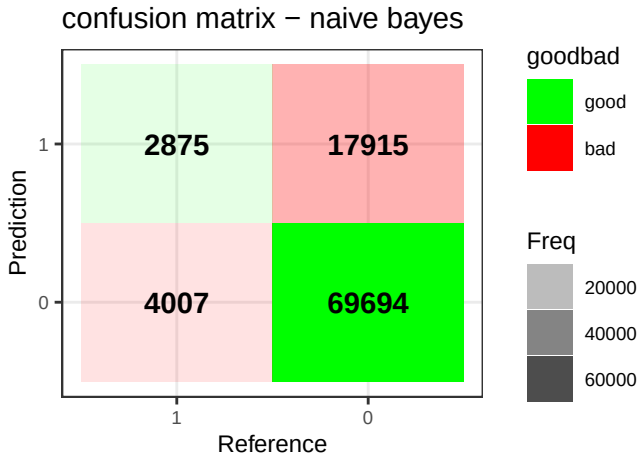
6.1 Model 1 - Naive bayes

First, we start with the Naive Bayes approach, which belongs to the family of simple "probabilistic classifiers" based on the application of Bayes theorem with strong assumptions on independence between features.

Model	Approach	Accuracy	Sensitivity	Specificity	Precision	Recall	F1
1	Naive Bayes	0.768	0.7955	0.4178	0.9456	0.7955	0.8641

The results show that we have an accuracy of 0.7679991 which is the first value to compare the following models with. We can see that in particular plausible projects are predicted quite good. However, the algorithm has a low specificity 0.4177565), a high precision of 0.9456317 and a solid F1 value (0.8641002).

In many cases the model predicts an implausible project where it is plausible in real. From a business perspective more problematic are those cases that are implausible or potentially wrong, but not detected by the model. This will have consequences for the business since dissatisfied customers will call the company and complain about it. The following plot shows the total numbers in the confusion matrix.



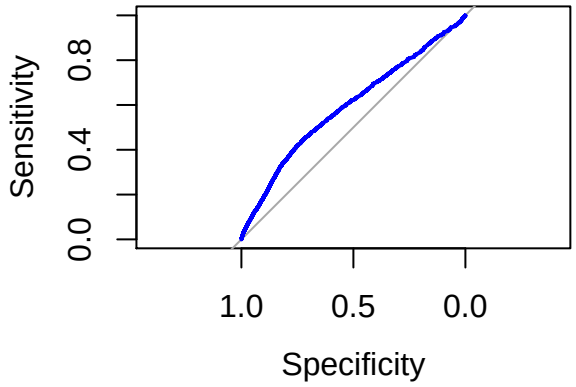
6.2 Model 2- Logistic regression

We continue with a logistic regression model (used for classification) that is created based on the train data. Since this is a binomial question (i.e. classification with two outcomes), we use family = "binomial": the model should predict whether project data is plausible or not.

Model	Approach	Accuracy	Sensitivity	Specificity	Precision	Recall	F1
1	Naive Bayes	0.768	0.7955	0.4178	0.9456	0.7955	0.8641
2	Logistic Regression	0.7801	0.7812	0.5115	0.9974	0.7812	0.8762

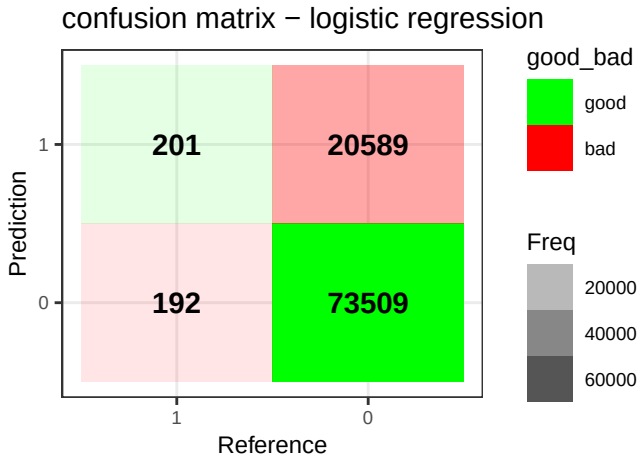
The results show that we have an accuracy of 0.7800743 which is slightly better compared to the Naive Bayes model. It seems to be a better fit. The specificity is better (0.5114504), the sensitivity lower (0.7811962). Interestingly, the precision is almost 1 and the F1 value also increased (0.8761554).

In addition, if we have a look on the ROC curve and calculate the area under the curve (auc) we see that we are a bit better compared to making predictions at random with a value of 0.5927403.

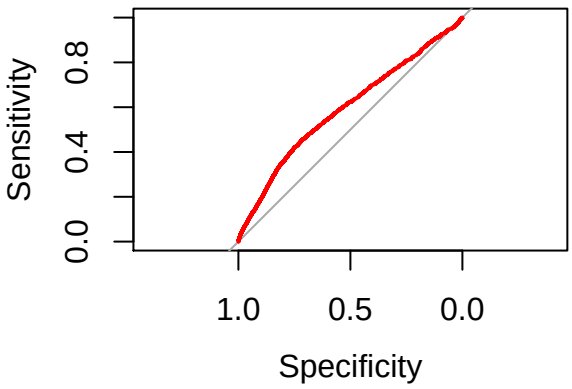



```
## NULL
```

Let us have a look on the confusion matrix. It can be seen that many cases are recognized correctly, as the accuracy already shows. Particularly outstanding are the projects that are plausible and are also recognized as plausible. Critical are the projects that are incorrect, but are marked as plausible. However, compared to the Naive Bayes model we reduced those cases to under 200 which really is a benefit of this model. On the other hand, we can see that the number of false positives increased. This means that we have more cases that are correct in real but flagged as implausible and results in more manual re-working of the projects and invoices. A significantly higher volume of manual testings would be required in the company. Although, we would like to have a balanced model, we estimate the damage based on reputation higher compared to manual re-work.



Before we continue we want to apply another approach to the logistic model: a step-wise extension of the model. For this we first create the null model and on the other hand define the maximum model with all predictors. Subsequently, the approach will extend the model step by step in the following order: line_amount, qty, empl_item_detail_id, category_id, and channel_id. Finally, we consider the ROC and the auc for the resulting best model.




```
## NULL
```

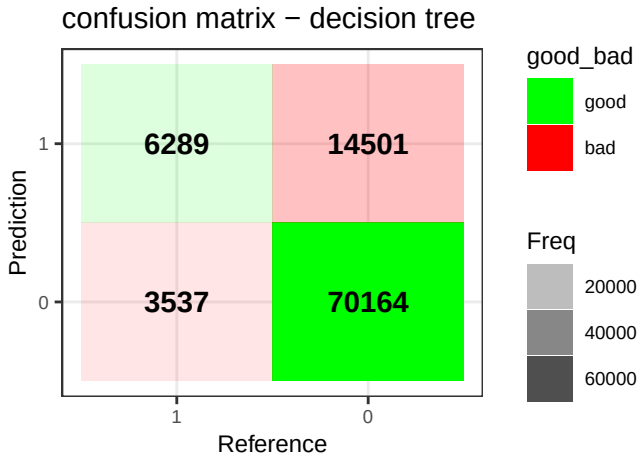
We find that the step-wise approach takes in all predictors, same as our first logistic model. Therefore, we see that we have the same auc value of 0.5927403. This means we have already obtained a relatively robust model in the first logistic regression model. Thus, we neglect to plot the same confusion matrix here again.

6.3 Model 3 - Decision tree

As a first step in terms of tree-based algorithms we take the decision tree as classifier using the rpart package. The results are as follows:

Model	Approach	Accuracy	Sensitivity	Specificity	Precision	Recall	F1
1	Naive Bayes	0.768	0.7955	0.4178	0.9456	0.7955	0.8641
2	Logistic Regression	0.7801	0.7812	0.5115	0.9974	0.7812	0.8762
3	Decision Tree	0.8091	0.8287	0.64	0.952	0.8287	0.8861

The accuracy is higher than for the models before with a value of 0.8091035, but if we look at the false positive and false negative values, we see that especially the false negatives have increased significantly. The model now more often gives the impression that the projects are actually correct, although this is not the case. In this respect, the logistic model still has an advantage. But since we put our focus on the reputational damage, we are happy to see that the true negative have increased well. As well, the number of false negative decreased.

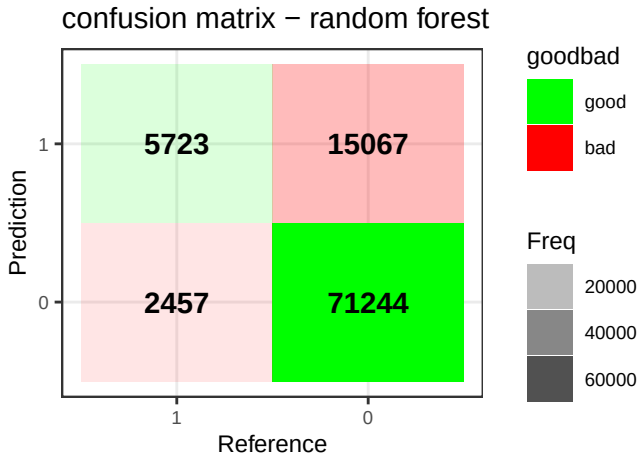


6.4 Model 4 - Random forest

In this approach, we apply the random forest approach for this classification problem. To get a first indication quickly despite the size of the data set, we use the ranger package. We find the results here:

Model	Approach	Accuracy	Sensitivity	Specificity	Precision	Recall	F1
1	Naive Bayes	0.768	0.7955	0.4178	0.9456	0.7955	0.8641
2	Logistic Regression	0.7801	0.7812	0.5115	0.9974	0.7812	0.8762
3	Decision Tree	0.8091	0.8287	0.64	0.952	0.8287	0.8861
4	Random Forest	0.8145	0.8254	0.6996	0.9667	0.8254	0.8905

The random forest model provides the best result in terms of an overall accuracy value of 0.8145432. Although sensitivity has decreased a little compared to the decision tree model, specificity has increased again (0.6996333). Thus, also more implausible projects are predicted as implausible compared to the former models. However, if we look at the false negative values, we find that we are significantly higher compared to the logistic regression model. This model here is significantly better at identifying implausible projects.



We can also see that the predictors have different importance: line_amount, qty and empl_item_detail_id seem to have the highest importance values (i.e. Gini index). The importance of the variables shows the following table:

	.
channel_id	13124
category_id	9850
qty	21609
line_amount	22307
empl_item_detail_id	19521

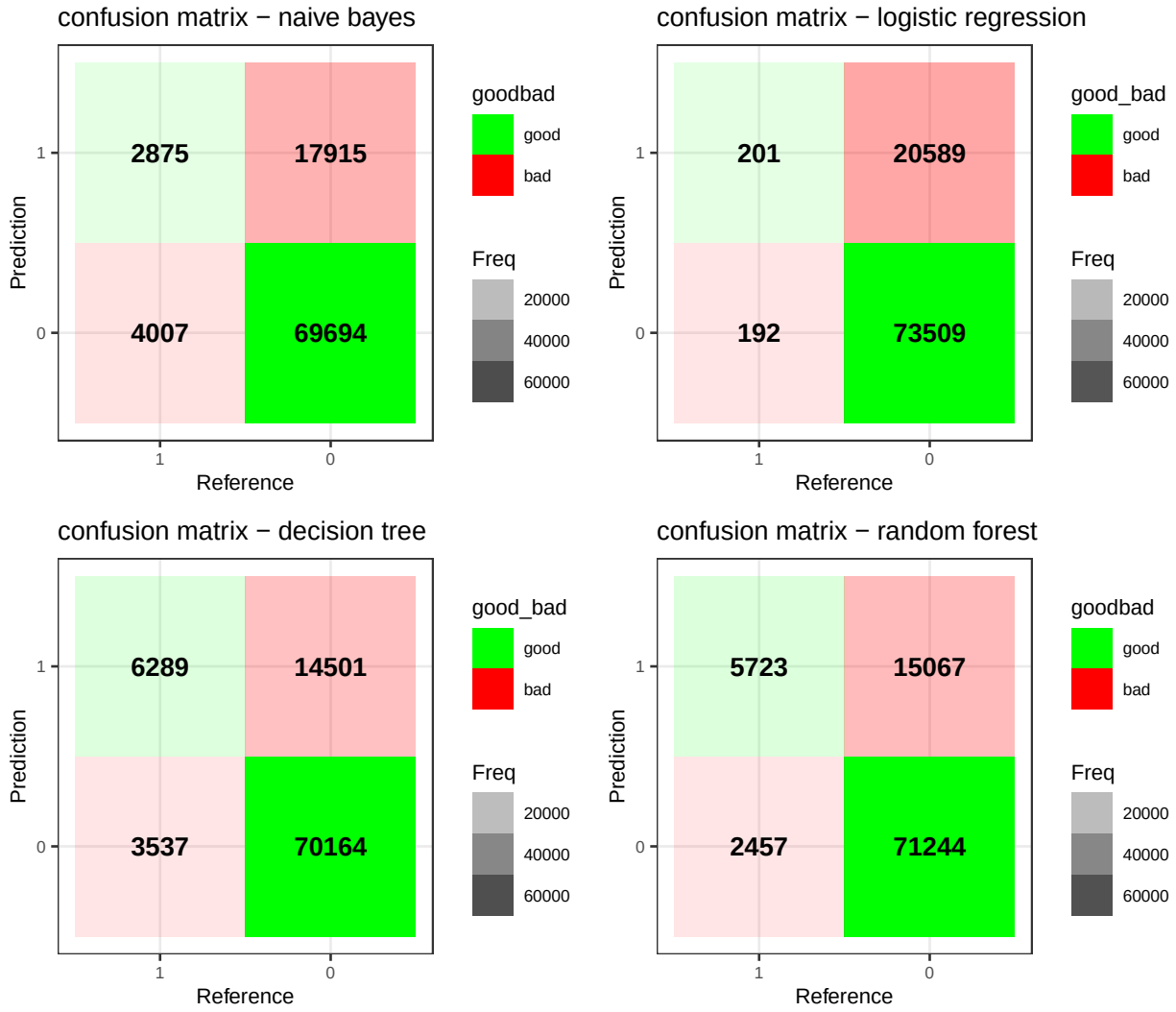
6.5 Model 5 - K-nearest neighbours

Although we tried to apply the k nearest neighbors model, this was not possible due to the conditions of the existing operating system. A run of 16 hours did not lead to any result. Therefore, we will refrain from taking a closer look at this model in the following.

6.6 Summary

Now that we have run the individual models, we will compare them with each other in terms of their results.

- Model 1 - Naive Bayes
- Model 2 - Logistic Regression
- Model 3 - Decision Tree
- Model 4 - Random Forest



Model	Approach	Accuracy	Sensitivity	Specificity	Precision	Recall	F1
1	Naive Bayes	0.768	0.7955	0.4178	0.9456	0.7955	0.8641
2	Logistic Regression	0.7801	0.7812	0.5115	0.9974	0.7812	0.8762
3	Decision Tree	0.8091	0.8287	0.64	0.952	0.8287	0.8861
4	Random Forest	0.8145	0.8254	0.6996	0.9667	0.8254	0.8905

We see that the model 4 “Random Forest” has the best results in terms of overall accuracy in the confusion matrices, i.e. the missclass error is minimal for the models. In terms of specificity, the decision tree approach is somewhat better, which is especially evident in precision. Here, the logistic regression model is also clearly better. Finally however, it can be seen that the F1 value again speaks for the random forest approach. In view of the fact that the company primarily wants to identify projects that are implausible in order to ensure its reputation with customers, we focus on this aspect. In addition, care should also be taken to ensure that the number of false positive cases is as low as possible so that the internal manual effort is not too great. For this reason, the decision is made in favor of the two models decision tree and random forest.

7 Model prediction

As defined in the last section, we now apply the two models decision tree and random forest to classify the projects from the year 2021 into plausible and implausible ones.

- Decision tree: In the case of the decision tree model, we find that we get 11215 flagged projects (i.e. 0.0881634 percent). We are thus below the percentage of about 16% for the projects from 2015 to 2020. This seems to indicate that we are not subject to overfitting.
- Random forest: In the result of the validation, we can see that from the 127207 projects in 2021 exactly 7688 will be flagged as implausible ones, i.e. about 0.0604369 percent of the projects. This is lower than the percentage for the decision tree model.

8 Conclusion

Overall, we can see that we can make a good prediction for the projects in 2021 with our two models. The danger of overfitting probably does not exist, as the first results show. Now the use of the models in practice must show whether they pay off and can help identify implausible projects and invoices. In addition, it must become clear whether the number of false positives or false negatives are indeed incorrect in order to better assess the consequences: the internal effort required to manually check correct project information or the unforeseen reputational damage that the company could suffer.

The limitations of this work are as follows: First, the predictions were made only on the basis of a few predictors. In reality, additional variables, such as information from orders, customer complaints, and evidence of manual rework, should be considered to extend the models. This would allow them to provide even better predictive content. In addition, only a subset of possible models was applied. The same applies to the composition of different combinations of predictors.

For future work, the following considerations could be applied: On the one hand, the training data could be extended. In addition, several classification models should be applied. It could also help to proceed in the sense of unsupervised models, i.e. to try to give the algorithm the possibility to find out by itself whether and if so which project information is implausible. Finally, a more comprehensive database would also help to give the model more predictive power.

9 Appendix

Please find the summary for the time elapsed for the different steps in the table below.

event	start	end	timeElapsed_min
Import data	2022-02-12 14:44:21	2022-02-12 14:44:46	0.42
Explore, clean and visualize data	2022-02-12 14:44:46	2022-02-12 14:48:23	3.62
Modeling approach	2022-02-12 14:48:23	2022-02-12 14:50:15	1.87
Train and compare different models	2022-02-12 14:50:15	2022-02-12 14:58:16	8.02
Naive bayes	2022-02-12 14:50:15	2022-02-12 14:50:26	0.18
Logistic regression	2022-02-12 14:50:26	2022-02-12 14:52:07	1.68
Random forest	2022-02-12 14:53:13	2022-02-12 14:58:15	5.03
Model validation and results	2022-02-12 14:58:16	2022-02-12 14:58:24	0.13
Conclusion	2022-02-12 14:58:24	2022-02-12 14:58:24	0