Capstone_Zeit_EDA

November 25, 2020

Description: In this project We want to use different machine learning methods to do churn prediction of subscriptions of the german newspaper "Die Zeit". This notebook contains the first part op the project consisting of

- Business Understanding (link to our wiki)
- Data Overview and Editing
- Data Cleaning
- Exploratory Data Analysis
- Features Engineering & Data Dropping

Project Name: Churn Prediction - Die Zeit

Team: Carlotta Ulm, Silas Mederer, Jonas Bechthold

Date: 2020-10-26 to 2020-11-27

1 Setting up environment and imports

[280]:
```
# data analysis and wrangling
import pandas as pd
import numpy as np
import math
import itertools
from time import time

# own modules
import eda_methods as eda

# visualization
import seaborn as sns
sns.set(style="white")
#sns.set_theme()

import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('ggplot')
from pandas.plotting import scatter_matrix
```

```
# pandas profiling
from pandas_profiling import ProfileReport

# warnings handler
import warnings
warnings.filterwarnings("ignore")

random_state = 100           # Ensures modeling results can be replicated
np.random.seed(42)

# Display Options for pandas
pd.set_option('display.max_columns', None) # Sets maximum columns displayed in␣
 ↪tables
pd.set_option('display.max_rows', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', -1)

# Variables for plot sizes
matplotlib.rc('font', size=20)          # controls default text sizes
matplotlib.rc('axes', titlesize=16)     # fontsize of the axes title
matplotlib.rc('axes', labelsize=16)    # fontsize of the x and y labels
matplotlib.rc('xtick', labelsize=16)    # fontsize of the tick labels
matplotlib.rc('ytick', labelsize=16)    # fontsize of the tick labels
matplotlib.rc('legend', fontsize=16)    # legend fontsize
matplotlib.rc('figure', titlesize=20)

######################################################
# Machine Learning Libraries
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import fbeta_score, accuracy_score, f1_score,␣
 ↪recall_score, precision_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer
from sklearn.model_selection import KFold
from sklearn.metrics import roc_curve
from sklearn.metrics import roc_auc_score

#Pipeline
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
```

```
#geodaten
import geopandas as gpd
```

2 Business Understanding

Please check the wiki page: Business understanding (publishing and news).

2.1 Dataset Description

Let's get an idea about the columns and find out what they mean.

[2]:
```
# new dataframe 2
df = pd.read_csv('data/f_chtr_churn_traintable_nf_2.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209043 entries, 0 to 209042
Columns: 171 entries, Unnamed: 0 to date_x
dtypes: float64(32), int64(121), object(18)
memory usage: 272.7+ MB
```

[3]:
```
#df.head()
```

[4]:
```
#df.tail(1)
```

[5]:
```
df.drop(["Unnamed: 0","auftrag_new_id"], axis=1, inplace=True)
```

[6]:
```
eda.describe_plus(df).round(2)
```

[6]:

	count	mean	std	min	25%	\
lesedauer	209043.0	32.05	21.56	0.0	14.00	
rechnungsmonat	209043.0	0.10	0.30	0.0	0.00	
studentenabo	209043.0	0.11	0.31	0.0	0.00	
metropole	209043.0	0.29	0.46	0.0	0.00	
shop_kauf	209043.0	0.54	2.47	0.0	0.00	
unterbrechung	209043.0	0.10	0.30	0.0	0.00	
avg_churn	209043.0	0.31	0.15	0.1	0.18	
email_am_kunden	209031.0	0.94	0.25	0.0	1.00	
zon_che_opt_in	209043.0	0.00	0.05	0.0	0.00	
zon_sit_opt_in	209043.0	0.00	0.03	0.0	0.00	
zon_zp_grey	209043.0	0.17	0.46	0.0	0.00	
zon_premium	209043.0	0.17	0.47	0.0	0.00	
zon_boa	209043.0	0.01	0.11	0.0	0.00	
zon_kommentar	209043.0	0.00	0.07	0.0	0.00	
zon_sonstige	209043.0	0.05	0.27	0.0	0.00	
zon_zp_red	209043.0	0.04	0.25	0.0	0.00	

zon_rawr	209043.0	0.00	0.02	0.0	0.00
zon_community	209043.0	0.00	0.01	0.0	0.00
zon_app_sonstige	209043.0	0.00	0.06	0.0	0.00
zon_schach	209043.0	0.00	0.02	0.0	0.00
zon_blog_kommentare	209043.0	0.00	0.03	0.0	0.00
zon_quiz	209043.0	0.00	0.02	0.0	0.00
cnt_abo	209043.0	7.48	168.92	0.0	0.00
cnt_abo_diezeit	209043.0	4.91	97.80	0.0	0.00
cnt_abo_diezeit_digital	209043.0	0.53	11.28	0.0	0.00
cnt_abo_magazin	209043.0	0.60	10.20	0.0	0.00
cnt_umwandlungsstatus2_dkey	209043.0	2.90	72.65	0.0	0.00
nl_zeitbrief	209043.0	1.41	0.85	0.0	1.00
nl_zeitshop	209043.0	0.34	0.72	0.0	0.00
nl_zeitverlag_hamburg	209043.0	0.12	0.47	0.0	0.00
nl_fdz_organisch	209043.0	0.00	0.02	0.0	0.00
nl_blacklist_sum	209043.0	0.06	0.47	0.0	0.00
nl_bounced_sum	209043.0	0.19	0.72	0.0	0.00
nl_aktivitaet	209043.0	6.89	4.54	0.0	4.00
nl_sperrliste_sum	209043.0	0.11	1.71	0.0	0.00
nl_opt_in_sum	209043.0	0.29	0.66	0.0	0.00
boa_reg	209043.0	0.08	0.27	0.0	0.00
che_reg	209043.0	0.09	0.29	0.0	0.00
sit_reg	209043.0	0.08	0.26	0.0	0.00
sso_reg	209043.0	0.71	0.45	0.0	0.00
received_anzahl_1w	209043.0	3.34	4.45	0.0	0.00
received_anzahl_1m	209043.0	12.80	16.50	0.0	1.00
received_anzahl_3m	209043.0	36.71	45.26	0.0	2.00
received_anzahl_6m	209043.0	72.56	87.43	0.0	5.00
opened_anzahl_1w	209043.0	1.11	2.59	0.0	0.00
opened_anzahl_1m	209043.0	4.18	9.24	0.0	0.00
opened_anzahl_3m	209043.0	11.66	25.02	0.0	0.00
openedanzahl_6m	209043.0	22.61	47.68	0.0	0.00
clicked_anzahl_1w	209043.0	0.12	0.50	0.0	0.00
clicked_anzahl_1m	209043.0	0.45	1.64	0.0	0.00
clicked_anzahl_3m	209043.0	1.20	3.91	0.0	0.00
clicked_anzahl_6m	209043.0	2.24	7.07	0.0	0.00
unsubscribed_anzahl_1w	209043.0	0.01	0.13	0.0	0.00
unsubscribed_anzahl_1m	209043.0	0.01	0.13	0.0	0.00
unsubscribed_anzahl_3m	209043.0	0.03	0.21	0.0	0.00
unsubscribed_anzahl_6m	209043.0	0.06	0.30	0.0	0.00
openrate_1w	209043.0	0.23	0.42	0.0	0.00
clickrate_1w	209043.0	0.04	0.18	0.0	0.00
openrate_1m	209043.0	0.24	0.35	0.0	0.00
clickrate_1m	209043.0	0.07	0.21	0.0	0.00
openrate_3m	209043.0	0.25	0.34	0.0	0.00
clickrate_3m	209043.0	0.10	0.23	0.0	0.00
received_anzahl_bestandskunden_1w	209043.0	0.05	0.24	0.0	0.00

received_anzahl_bestandskunden_1m	209043.0	0.16	0.42	0.0	0.00
received_anzahl_bestandskunden_3m	209043.0	0.35	0.96	0.0	0.00
received_anzahl_bestandskunden_6m	209043.0	0.39	1.13	0.0	0.00
opened_anzahl_bestandskunden_1w	209043.0	0.02	0.14	0.0	0.00
opened_anzahl_bestandskunden_1m	209043.0	0.07	0.30	0.0	0.00
opened_anzahl_bestandskunden_3m	209043.0	0.14	0.60	0.0	0.00
openedanzahl_bestandskunden_6m	209043.0	0.16	0.70	0.0	0.00
clicked_anzahl_bestandskunden_1w	209043.0	0.00	0.04	0.0	0.00
clicked_anzahl_bestandskunden_1m	209043.0	0.01	0.08	0.0	0.00
clicked_anzahl_bestandskunden_3m	209043.0	0.01	0.12	0.0	0.00
clicked_anzahl_bestandskunden_6m	209043.0	0.01	0.13	0.0	0.00
unsubscribed_anzahl_bestandskunden_1w	209043.0	0.00	0.03	0.0	0.00
unsubscribed_anzahl_bestandskunden_1m	209043.0	0.00	0.03	0.0	0.00
unsubscribed_anzahl_bestandskunden_3m	209043.0	0.00	0.04	0.0	0.00
unsubscribed_anzahl_bestandskunden_6m	209043.0	0.00	0.04	0.0	0.00
openrate_bestandskunden_1w	209043.0	0.02	0.13	0.0	0.00
clickrate_bestandskunden_1w	209043.0	0.00	0.04	0.0	0.00
openrate_bestandskunden_1m	209043.0	0.06	0.24	0.0	0.00
clickrate_bestandskunden_1m	209043.0	0.01	0.08	0.0	0.00
openrate_bestandskunden_3m	209043.0	0.06	0.22	0.0	0.00
clickrate_bestandskunden_3m	209043.0	0.01	0.09	0.0	0.00
received_anzahl_produktnews_1w	209043.0	0.04	0.23	0.0	0.00
received_anzahl_produktnews_1m	209043.0	0.15	0.58	0.0	0.00
received_anzahl_produktnews_3m	209043.0	0.37	1.21	0.0	0.00
received_anzahl_produktnews_6m	209043.0	0.42	1.29	0.0	0.00
opened_anzahl_produktnews_1w	209043.0	0.02	0.15	0.0	0.00
opened_anzahl_produktnews_1m	209043.0	0.08	0.41	0.0	0.00
opened_anzahl_produktnews_3m	209043.0	0.18	1.00	0.0	0.00
openedanzahl_produktnews_6m	209043.0	0.20	1.02	0.0	0.00
clicked_anzahl_produktnews_1w	209043.0	0.00	0.07	0.0	0.00
clicked_anzahl_produktnews_1m	209043.0	0.01	0.14	0.0	0.00
clicked_anzahl_produktnews_3m	209043.0	0.03	0.25	0.0	0.00
clicked_anzahl_produktnews_6m	209043.0	0.04	0.25	0.0	0.00
unsubscribed_anzahl_produktnews_1w	209043.0	0.00	0.02	0.0	0.00
unsubscribed_anzahl_produktnews_1m	209043.0	0.00	0.02	0.0	0.00
unsubscribed_anzahl_produktnews_3m	209043.0	0.00	0.03	0.0	0.00
unsubscribed_anzahl_produktnews_6m	209043.0	0.00	0.03	0.0	0.00
openrate_produktnews_1w	209043.0	0.02	0.12	0.0	0.00
clickrate_produktnews_1w	209043.0	0.00	0.05	0.0	0.00
openrate_produktnews_1m	209043.0	0.05	0.21	0.0	0.00
clickrate_produktnews_1m	209043.0	0.01	0.09	0.0	0.00
openrate_produktnews_3m	209043.0	0.09	0.28	0.0	0.00
clickrate_produktnews_3m	209043.0	0.02	0.12	0.0	0.00
received_anzahl_hamburg_1w	209043.0	0.32	1.68	0.0	0.00
received_anzahl_hamburg_1m	209043.0	1.22	6.12	0.0	0.00
received_anzahl_hamburg_3m	209043.0	3.48	16.81	0.0	0.00
received_anzahl_hamburg_6m	209043.0	6.87	33.18	0.0	0.00

opened_anzahl_hamburg_1w	209043.0	0.14	0.88	0.0	0.00
opened_anzahl_hamburg_1m	209043.0	0.52	3.31	0.0	0.00
opened_anzahl_hamburg_3m	209043.0	1.45	8.99	0.0	0.00
openedanzahl_hamburg_6m	209043.0	2.81	17.09	0.0	0.00
clicked_anzahl_hamburg_1w	209043.0	0.02	0.23	0.0	0.00
clicked_anzahl_hamburg_1m	209043.0	0.07	0.77	0.0	0.00
clicked_anzahl_hamburg_3m	209043.0	0.20	2.00	0.0	0.00
clicked_anzahl_hamburg_6m	209043.0	0.38	3.68	0.0	0.00
unsubscribed_anzahl_hamburg_1w	209043.0	0.00	0.02	0.0	0.00
unsubscribed_anzahl_hamburg_1m	209043.0	0.00	0.02	0.0	0.00
unsubscribed_anzahl_hamburg_3m	209043.0	0.00	0.03	0.0	0.00
unsubscribed_anzahl_hamburg_6m	209043.0	0.00	0.05	0.0	0.00
openrate_hamburg_1w	209043.0	0.02	0.14	0.0	0.00
clickrate_hamburg_1w	209043.0	0.00	0.06	0.0	0.00
openrate_hamburg_1m	209043.0	0.02	0.13	0.0	0.00
clickrate_hamburg_1m	209043.0	0.01	0.06	0.0	0.00
openrate_hamburg_3m	209043.0	0.02	0.13	0.0	0.00
clickrate_hamburg_3m	209043.0	0.01	0.07	0.0	0.00
received_anzahl_zeitbrief_1w	209043.0	0.51	0.69	0.0	0.00
received_anzahl_zeitbrief_1m	209043.0	2.00	2.55	0.0	0.00
received_anzahl_zeitbrief_3m	209043.0	5.86	7.29	0.0	0.00
received_anzahl_zeitbrief_6m	209043.0	11.84	14.30	0.0	0.00
opened_anzahl_zeitbrief_1w	209043.0	0.21	0.53	0.0	0.00
opened_anzahl_zeitbrief_1m	209043.0	0.77	1.68	0.0	0.00
opened_anzahl_zeitbrief_3m	209043.0	2.19	4.39	0.0	0.00
openedanzahl_zeitbrief_6m	209043.0	4.34	8.53	0.0	0.00
clicked_anzahl_zeitbrief_1w	209043.0	0.02	0.14	0.0	0.00
clicked_anzahl_zeitbrief_1m	209043.0	0.07	0.36	0.0	0.00
clicked_anzahl_zeitbrief_3m	209043.0	0.20	0.85	0.0	0.00
clicked_anzahl_zeitbrief_6m	209043.0	0.41	1.58	0.0	0.00
unsubscribed_anzahl_zeitbrief_1w	209043.0	0.00	0.06	0.0	0.00
unsubscribed_anzahl_zeitbrief_1m	209043.0	0.00	0.06	0.0	0.00
unsubscribed_anzahl_zeitbrief_3m	209043.0	0.01	0.09	0.0	0.00
unsubscribed_anzahl_zeitbrief_6m	209043.0	0.02	0.13	0.0	0.00
openrate_zeitbrief_1w	209043.0	0.17	0.41	0.0	0.00
clickrate_zeitbrief_1w	209043.0	0.02	0.13	0.0	0.00
openrate_zeitbrief_1m	209043.0	0.16	0.32	0.0	0.00
clickrate_zeitbrief_1m	209043.0	0.03	0.15	0.0	0.00
openrate_zeitbrief_3m	209043.0	0.16	0.30	0.0	0.00
clickrate_zeitbrief_3m	209043.0	0.04	0.16	0.0	0.00
training_set	209043.0	1.00	0.00	1.0	1.00
churn	209043.0	0.32	0.47	0.0	0.00

	50%	75%	max	skew	\
lesedauer	27.00	47.00	88.00	0.69	
rechnungsmonat	0.00	0.00	1.00	2.65	
studentenabo	0.00	0.00	1.00	2.49	

metropole	0.00	1.00	1.00	0.91
shop_kauf	0.00	0.00	152.00	14.98
unterbrechung	0.00	0.00	1.00	2.64
avg_churn	0.29	0.41	0.70	0.69
email_am_kunden	1.00	1.00	1.00	-3.55
zon_che_opt_in	0.00	0.00	2.00	27.52
zon_sit_opt_in	0.00	0.00	2.00	41.15
zon_zp_grey	0.00	0.00	2.00	2.78
zon_premium	0.00	0.00	2.00	2.78
zon_boa	0.00	0.00	2.00	14.00
zon_kommentar	0.00	0.00	2.00	24.75
zon_sonstige	0.00	0.00	2.00	5.93
zon_zp_red	0.00	0.00	2.00	6.32
zon_rawr	0.00	0.00	2.00	67.63
zon_community	0.00	0.00	2.00	311.09
zon_app_sonstige	0.00	0.00	2.00	24.74
zon_schach	0.00	0.00	2.00	75.10
zon_blog_kommentare	0.00	0.00	2.00	55.39
zon_quiz	0.00	0.00	2.00	87.88
cnt_abo	1.00	2.00	7104.00	34.96
cnt_abo_diezeit	1.00	2.00	3708.00	31.20
cnt_abo_diezeit_digital	0.00	0.00	516.00	35.23
cnt_abo_magazin	0.00	0.00	414.00	29.90
cnt_umwandlungsstatus2_dkey	0.00	1.00	3129.00	38.32
nl_zeitbrief	2.00	2.00	2.00	-0.89
nl_zeitshop	0.00	0.00	2.00	1.75
nl_zeitverlag_hamburg	0.00	0.00	2.00	3.62
nl_fdz_organisch	0.00	0.00	1.00	44.59
nl_blacklist_sum	0.00	0.00	22.00	15.03
nl_bounced_sum	0.00	0.00	16.00	5.19
nl_aktivitaet	6.00	9.00	39.00	1.07
nl_sperrliste_sum	0.00	0.00	133.00	21.62
nl_opt_in_sum	0.00	0.00	19.00	4.00
boa_reg	0.00	0.00	1.00	3.17
che_reg	0.00	0.00	1.00	2.76
sit_reg	0.00	0.00	1.00	3.20
sso_reg	1.00	1.00	1.00	-0.92
received_anzahl_1w	2.00	5.00	54.00	2.72
received_anzahl_1m	7.00	19.00	203.00	2.60
received_anzahl_3m	20.00	57.00	573.00	2.35
received_anzahl_6m	43.00	116.00	1147.00	2.30
opened_anzahl_1w	0.00	1.00	60.00	4.51
opened_anzahl_1m	1.00	4.00	314.00	4.46
opened_anzahl_3m	2.00	11.00	572.00	4.32
openedanzahl_6m	4.00	22.00	1145.00	4.24
clicked_anzahl_1w	0.00	0.00	19.00	7.68
clicked_anzahl_1m	0.00	0.00	66.00	9.05

clicked_anzahl_3m	0.00	1.00	169.00	9.64
clicked_anzahl_6m	0.00	2.00	307.00	10.16
unsubscribed_anzahl_1w	0.00	0.00	5.00	12.12
unsubscribed_anzahl_1m	0.00	0.00	5.00	12.12
unsubscribed_anzahl_3m	0.00	0.00	8.00	8.59
unsubscribed_anzahl_6m	0.00	0.00	13.00	7.48
openrate_1w	0.00	0.36	22.00	4.10
clickrate_1w	0.00	0.00	3.00	4.67
openrate_1m	0.02	0.40	10.75	1.79
clickrate_1m	0.00	0.00	2.00	3.47
openrate_3m	0.07	0.43	9.00	1.66
clickrate_3m	0.00	0.07	2.00	2.94
received_anzahl_bestandskunden_1w	0.00	0.00	4.00	6.05
received_anzahl_bestandskunden_1m	0.00	0.00	4.00	3.39
received_anzahl_bestandskunden_3m	0.00	0.00	12.00	4.75
received_anzahl_bestandskunden_6m	0.00	0.00	16.00	5.44
opened_anzahl_bestandskunden_1w	0.00	0.00	1.00	7.01
opened_anzahl_bestandskunden_1m	0.00	0.00	4.00	6.40
opened_anzahl_bestandskunden_3m	0.00	0.00	12.00	8.29
openedanzahl_bestandskunden_6m	0.00	0.00	16.00	10.06
clicked_anzahl_bestandskunden_1w	0.00	0.00	1.00	22.40
clicked_anzahl_bestandskunden_1m	0.00	0.00	2.00	12.47
clicked_anzahl_bestandskunden_3m	0.00	0.00	3.00	9.79
clicked_anzahl_bestandskunden_6m	0.00	0.00	4.00	9.77
unsubscribed_anzahl_bestandskunden_1w	0.00	0.00	1.00	36.45
unsubscribed_anzahl_bestandskunden_1m	0.00	0.00	1.00	36.45
unsubscribed_anzahl_bestandskunden_3m	0.00	0.00	1.00	24.85
unsubscribed_anzahl_bestandskunden_6m	0.00	0.00	1.00	23.94
openrate_bestandskunden_1w	0.00	0.00	1.00	7.32
clickrate_bestandskunden_1w	0.00	0.00	1.00	22.85
openrate_bestandskunden_1m	0.00	0.00	2.00	3.79
clickrate_bestandskunden_1m	0.00	0.00	2.00	12.55
openrate_bestandskunden_3m	0.00	0.00	2.00	3.56
clickrate_bestandskunden_3m	0.00	0.00	1.00	10.24
received_anzahl_produktnews_1w	0.00	0.00	4.00	8.43
received_anzahl_produktnews_1m	0.00	0.00	8.00	6.44
received_anzahl_produktnews_3m	0.00	0.00	32.00	12.79
received_anzahl_produktnews_6m	0.00	0.00	32.00	11.18
opened_anzahl_produktnews_1w	0.00	0.00	5.00	10.38
opened_anzahl_produktnews_1m	0.00	0.00	8.00	9.92
opened_anzahl_produktnews_3m	0.00	0.00	32.00	21.17
openedanzahl_produktnews_6m	0.00	0.00	32.00	19.94
clicked_anzahl_produktnews_1w	0.00	0.00	4.00	19.43
clicked_anzahl_produktnews_1m	0.00	0.00	5.00	13.23
clicked_anzahl_produktnews_3m	0.00	0.00	9.00	10.38
clicked_anzahl_produktnews_6m	0.00	0.00	9.00	9.91
unsubscribed_anzahl_produktnews_1w	0.00	0.00	1.00	46.63

unsubscribed_anzahl_produktnews_1m	0.00	0.00	1.00	46.63
unsubscribed_anzahl_produktnews_3m	0.00	0.00	1.00	31.35
unsubscribed_anzahl_produktnews_6m	0.00	0.00	1.00	28.86
openrate_produktnews_1w	0.00	0.00	3.00	8.10
clickrate_produktnews_1w	0.00	0.00	2.00	18.28
openrate_produktnews_1m	0.00	0.00	5.00	4.65
clickrate_produktnews_1m	0.00	0.00	2.00	10.16
openrate_produktnews_3m	0.00	0.00	3.00	2.85
clickrate_produktnews_3m	0.00	0.00	1.00	7.26
received_anzahl_hamburg_1w	0.00	0.00	32.00	9.27
received_anzahl_hamburg_1m	0.00	0.00	100.00	8.38
received_anzahl_hamburg_3m	0.00	0.00	248.00	7.46
received_anzahl_hamburg_6m	0.00	0.00	500.00	7.62
opened_anzahl_hamburg_1w	0.00	0.00	28.00	8.06
opened_anzahl_hamburg_1m	0.00	0.00	52.00	8.53
opened_anzahl_hamburg_3m	0.00	0.00	132.00	7.93
openedanzahl_hamburg_6m	0.00	0.00	240.00	7.62
clicked_anzahl_hamburg_1w	0.00	0.00	10.00	16.62
clicked_anzahl_hamburg_1m	0.00	0.00	30.00	17.67
clicked_anzahl_hamburg_3m	0.00	0.00	71.00	17.26
clicked_anzahl_hamburg_6m	0.00	0.00	124.00	17.32
unsubscribed_anzahl_hamburg_1w	0.00	0.00	1.00	50.46
unsubscribed_anzahl_hamburg_1m	0.00	0.00	1.00	50.46
unsubscribed_anzahl_hamburg_3m	0.00	0.00	1.00	31.43
unsubscribed_anzahl_hamburg_6m	0.00	0.00	1.00	21.60
openrate_hamburg_1w	0.00	0.00	4.67	7.50
clickrate_hamburg_1w	0.00	0.00	1.17	14.38
openrate_hamburg_1m	0.00	0.00	2.55	6.47
clickrate_hamburg_1m	0.00	0.00	2.00	12.95
openrate_hamburg_3m	0.00	0.00	2.00	6.40
clickrate_hamburg_3m	0.00	0.00	1.05	11.79
received_anzahl_zeitbrief_1w	0.00	1.00	4.00	1.36
received_anzahl_zeitbrief_1m	0.00	4.00	20.00	1.60
received_anzahl_zeitbrief_3m	0.00	13.00	56.00	1.56
received_anzahl_zeitbrief_6m	0.00	26.00	108.00	1.52
opened_anzahl_zeitbrief_1w	0.00	0.00	25.00	4.23
opened_anzahl_zeitbrief_1m	0.00	1.00	26.00	3.92
opened_anzahl_zeitbrief_3m	0.00	2.00	52.00	3.43
openedanzahl_zeitbrief_6m	0.00	5.00	100.00	3.44
clicked_anzahl_zeitbrief_1w	0.00	0.00	4.00	8.02
clicked_anzahl_zeitbrief_1m	0.00	0.00	6.00	6.77
clicked_anzahl_zeitbrief_3m	0.00	0.00	14.00	7.58
clicked_anzahl_zeitbrief_6m	0.00	0.00	29.00	8.04
unsubscribed_anzahl_zeitbrief_1w	0.00	0.00	1.00	16.47
unsubscribed_anzahl_zeitbrief_1m	0.00	0.00	1.00	16.47
unsubscribed_anzahl_zeitbrief_3m	0.00	0.00	1.00	10.68
unsubscribed_anzahl_zeitbrief_6m	0.00	0.00	2.00	7.68

openrate_zeitbrief_1w	0.00	0.00	25.00	4.31
clickrate_zeitbrief_1w	0.00	0.00	3.00	7.75
openrate_zeitbrief_1m	0.00	0.20	5.25	2.00
clickrate_zeitbrief_1m	0.00	0.00	2.00	5.59
openrate_zeitbrief_3m	0.00	0.15	4.00	1.86
clickrate_zeitbrief_3m	0.00	0.00	2.00	4.79
training_set	1.00	1.00	1.00	0.00
churn	0.00	1.00	1.00	0.79

	kurtosis	variance
lesedauer	-0.57	464.71
rechnungsmonat	5.03	0.09
studentenabo	4.22	0.10
metropole	-1.18	0.21
shop_kauf	503.79	6.09
unterbrechung	4.95	0.09
avg_churn	-0.33	0.02
email_am_kunden	10.62	0.06
zon_che_opt_in	841.32	0.00
zon_sit_opt_in	1899.34	0.00
zon_zp_grey	6.98	0.22
zon_premium	6.94	0.22
zon_boa	216.14	0.01
zon_kommentar	652.74	0.00
zon_sonstige	36.36	0.07
zon_zp_red	41.83	0.06
zon_rawr	5139.57	0.00
zon_community	104519.22	0.00
zon_app_sonstige	666.89	0.00
zon_schach	6275.46	0.00
zon_blog_kommentare	3393.85	0.00
zon_quiz	8172.20	0.00
cnt_abo	1295.57	28532.40
cnt_abo_diezeit	1025.37	9565.73
cnt_abo_diezeit_digital	1326.81	127.22
cnt_abo_magazin	949.93	104.00
cnt_umwandlungsstatus2_dkey	1528.94	5277.83
nl_zeitbrief	-1.04	0.73
nl_zeitshop	1.24	0.51
nl_zeitverlag_hamburg	11.40	0.22
nl_fdz_organisch	1985.93	0.00
nl_blacklist_sum	332.83	0.22
nl_bounced_sum	38.26	0.53
nl_aktivitaet	2.78	20.62
nl_sperrliste_sum	652.59	2.93
nl_opt_in_sum	36.07	0.44
boa_reg	8.06	0.07

che_reg	5.64	0.09
sit_reg	8.24	0.07
sso_reg	-1.15	0.21
received_anzahl_1w	12.37	19.78
received_anzahl_1m	11.08	272.39
received_anzahl_3m	8.86	2048.62
received_anzahl_6m	8.70	7644.25
opened_anzahl_1w	32.04	6.69
opened_anzahl_1m	33.27	85.45
opened_anzahl_3m	28.88	625.92
openedanzahl_6m	27.83	2273.64
clicked_anzahl_1w	99.23	0.25
clicked_anzahl_1m	139.07	2.71
clicked_anzahl_3m	171.76	15.32
clicked_anzahl_6m	193.47	50.05
unsubscribed_anzahl_1w	200.52	0.02
unsubscribed_anzahl_1m	200.52	0.02
unsubscribed_anzahl_3m	112.16	0.04
unsubscribed_anzahl_6m	121.01	0.09
openrate_1w	78.87	0.18
clickrate_1w	22.72	0.03
openrate_1m	12.02	0.12
clickrate_1m	11.91	0.04
openrate_3m	10.96	0.11
clickrate_3m	8.31	0.05
received_anzahl_bestandskunden_1w	58.32	0.06
received_anzahl_bestandskunden_1m	18.34	0.18
received_anzahl_bestandskunden_3m	40.59	0.92
received_anzahl_bestandskunden_6m	52.81	1.28
opened_anzahl_bestandskunden_1w	47.07	0.02
opened_anzahl_bestandskunden_1m	61.11	0.09
opened_anzahl_bestandskunden_3m	116.57	0.36
openedanzahl_bestandskunden_6m	175.71	0.49
clicked_anzahl_bestandskunden_1w	499.95	0.00
clicked_anzahl_bestandskunden_1m	155.39	0.01
clicked_anzahl_bestandskunden_3m	106.95	0.01
clicked_anzahl_bestandskunden_6m	111.91	0.02
unsubscribed_anzahl_bestandskunden_1w	1326.52	0.00
unsubscribed_anzahl_bestandskunden_1m	1326.52	0.00
unsubscribed_anzahl_bestandskunden_3m	615.32	0.00
unsubscribed_anzahl_bestandskunden_6m	570.89	0.00
openrate_bestandskunden_1w	51.66	0.02
clickrate_bestandskunden_1w	520.25	0.00
openrate_bestandskunden_1m	12.95	0.06
clickrate_bestandskunden_1m	156.37	0.01
openrate_bestandskunden_3m	11.41	0.05
clickrate_bestandskunden_3m	107.82	0.01

received_anzahl_produktnews_1w	100.90	0.05
received_anzahl_produktnews_1m	57.04	0.34
received_anzahl_produktnews_3m	288.76	1.47
received_anzahl_produktnews_6m	228.16	1.67
opened_anzahl_produktnews_1w	147.41	0.02
opened_anzahl_produktnews_1m	138.59	0.17
opened_anzahl_produktnews_3m	634.32	1.00
openedanzahl_produktnews_6m	578.61	1.05
clicked_anzahl_produktnews_1w	464.22	0.00
clicked_anzahl_produktnews_1m	236.00	0.02
clicked_anzahl_produktnews_3m	143.42	0.06
clicked_anzahl_produktnews_6m	131.71	0.06
unsubscribed_anzahl_produktnews_1w	2172.58	0.00
unsubscribed_anzahl_produktnews_1m	2172.58	0.00
unsubscribed_anzahl_produktnews_3m	981.08	0.00
unsubscribed_anzahl_produktnews_6m	831.19	0.00
openrate_produktnews_1w	67.48	0.02
clickrate_produktnews_1w	339.78	0.00
openrate_produktnews_1m	23.59	0.05
clickrate_produktnews_1m	104.87	0.01
openrate_produktnews_3m	6.41	0.08
clickrate_produktnews_3m	53.11	0.01
received_anzahl_hamburg_1w	129.66	2.81
received_anzahl_hamburg_1m	104.37	37.40
received_anzahl_hamburg_3m	82.69	282.62
received_anzahl_hamburg_6m	86.96	1100.74
opened_anzahl_hamburg_1w	79.67	0.78
opened_anzahl_hamburg_1m	92.50	10.98
opened_anzahl_hamburg_3m	77.14	80.84
openedanzahl_hamburg_6m	69.23	292.13
clicked_anzahl_hamburg_1w	345.20	0.05
clicked_anzahl_hamburg_1m	392.87	0.60
clicked_anzahl_hamburg_3m	382.78	4.01
clicked_anzahl_hamburg_6m	388.78	13.57
unsubscribed_anzahl_hamburg_1w	2544.37	0.00
unsubscribed_anzahl_hamburg_1m	2544.37	0.00
unsubscribed_anzahl_hamburg_3m	985.75	0.00
unsubscribed_anzahl_hamburg_6m	464.77	0.00
openrate_hamburg_1w	75.58	0.02
clickrate_hamburg_1w	228.42	0.00
openrate_hamburg_1m	42.37	0.02
clickrate_hamburg_1m	188.57	0.00
openrate_hamburg_3m	41.09	0.02
clickrate_hamburg_3m	155.59	0.00
received_anzahl_zeitbrief_1w	2.44	0.47
received_anzahl_zeitbrief_1m	7.06	6.50
received_anzahl_zeitbrief_3m	7.10	53.10

```
received_anzahl_zeitbrief_6m          6.95       204.38
opened_anzahl_zeitbrief_1w           50.84         0.28
opened_anzahl_zeitbrief_1m           28.68         2.82
opened_anzahl_zeitbrief_3m           23.13        19.29
openedanzahl_zeitbrief_6m            22.96        72.69
clicked_anzahl_zeitbrief_1w          72.24         0.02
clicked_anzahl_zeitbrief_1m          58.64         0.13
clicked_anzahl_zeitbrief_3m          78.76         0.73
clicked_anzahl_zeitbrief_6m          89.13         2.50
unsubscribed_anzahl_zeitbrief_1w    269.34         0.00
unsubscribed_anzahl_zeitbrief_1m    269.34         0.00
unsubscribed_anzahl_zeitbrief_3m    112.06         0.01
unsubscribed_anzahl_zeitbrief_6m     57.09         0.02
openrate_zeitbrief_1w                90.38         0.17
clickrate_zeitbrief_1w               62.18         0.02
openrate_zeitbrief_1m                 4.11         0.10
clickrate_zeitbrief_1m               32.12         0.02
openrate_zeitbrief_3m                 2.49         0.09
clickrate_zeitbrief_3m               23.80         0.03
training_set                          0.00         0.00
churn                                -1.38         0.22
```

2.2 Get an idea of the column names by sampling

[7]:
```
#df.sample(2)
```

2.3 Dataset Overview with pandas profiling

Pandas Profiling for getting an overview of the new dataframe.

[8]:
```
# use pandas_profiling to create report
# save html to plots

#profile = ProfileReport(df)
#profile.to_file("plots/pandas_profiling_report_df2.html")
```

[9]:
```
# include pandas profiling report via html
from IPython.display import IFrame
IFrame(src='plots/pandas_profiling_report_df2.html', width=1000, height=700)
```

[9]:
```
<IPython.lib.display.IFrame at 0x7fc760db8ac8>
```

3 Data Editing

We reduce the number of cnt_abo maximum to 4 (maximum number of abos per customer, digital, print and magazines), see Business Understanding.

[10]:
```
df = df[df.cnt_abo < 5]
```

4 Data Cleaning

Purpose: Fix the inconsistencies within the data and handle the missing values

[11]:
```
null_rel = round(df.isin([0]).sum() / df.shape[0]*100,2)
null_rel = null_rel.to_frame()
null_rel.rename(columns={0: "zeros %"}, inplace=True)
eda.meta(df).T.join(null_rel).head(5)
```

[11]:

varname	nulls	percent	dtype	dup	nuniques	zeros %
kuendigungs_eingangs_datum	128706	69.67	object	True	349	0.00
ort	85	0.05	object	True	11260	0.00
email_am_kunden	12	0.01	float64	True	2	6.63
liefer_beginn_evt	0	0	object	True	406	0.00
clicked_anzahl_produktnews_3m	0	0	int64	True	8	97.53

[12]:
```
# numeric and categorical features
print(f"shape {df.shape}")
continues = df.select_dtypes(include=['float64','int64'])
print(f"numeric features {len(continues.columns)}")
categorial = df.select_dtypes(include="object")
print(f"object features  {len(categorial.columns)}")
```

```
shape (184745, 169)
numeric features 152
object features  17
```

Now we delete the null/missing values.

[13]:
```
df.dropna(subset=['ort','email_am_kunden'], inplace=True)
```

[14]:
```
null_rel = round(df.isin([0]).sum() / df.shape[0]*100,2)
null_rel = null_rel.to_frame()
null_rel.rename(columns={0: "zeros  %"}, inplace=True)
eda.meta(df).T.join(null_rel).head(5)
```

[14]:

varname	nulls	percent	dtype	dup	nuniques	\
kuendigungs_eingangs_datum	128680	69.68	object	True	349	
liefer_beginn_evt	0	0	object	True	406	
clickrate_produktnews_1w	0	0	float64	True	7	
clicked_anzahl_produktnews_1m	0	0	int64	True	6	
clicked_anzahl_produktnews_3m	0	0	int64	True	8	

```
                                zeros  %
varname
kuendigungs_eingangs_datum      0.00
liefer_beginn_evt               0.00
clickrate_produktnews_1w        99.68
clicked_anzahl_produktnews_1m   98.81
clicked_anzahl_produktnews_3m   97.53
```

4.1 Export the cleaned dataframe

[15]:
```
df.to_csv('data/df_clean_raw.csv',index=False)
```

4.2 Conclusion

Dataframe in general - The data set is complete and contains 184745 entries and 169 features. There are some features that we will not be able to use. For example the label "churn" or the feature "date_x", "kuendigungs_eingangs_datum" or "avg_churn" which probably contains the values from the current churn-prediction-model.

Missings - There are only two features with missing values. The feature "kuendigungs_eingangs_datum" is null if there is no notice of termination, so these nulls are correct. The missing values of ort and email_am_kunden have been dropped.

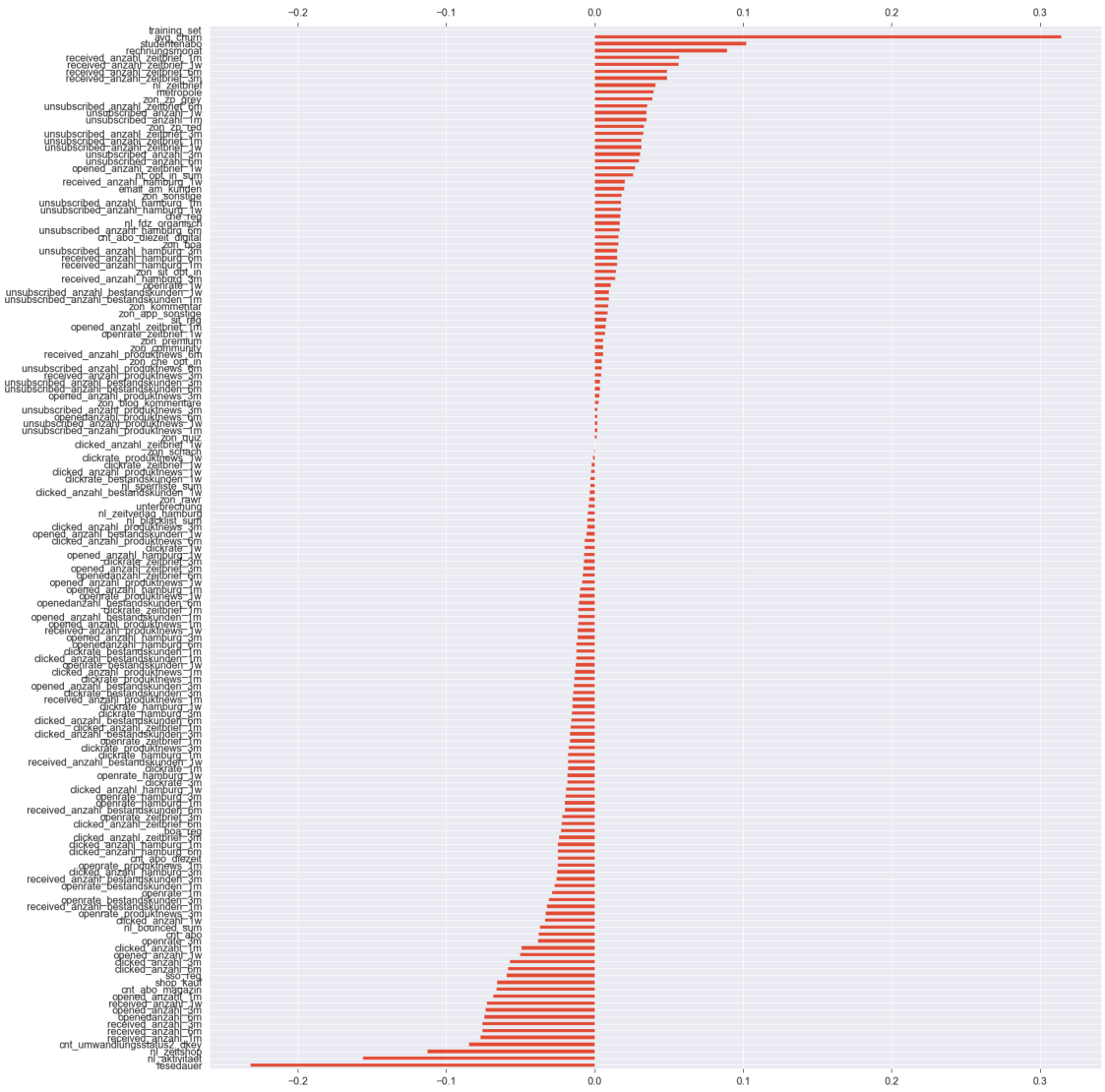
Dublicates - The feature "auftrag_new_id" is the only one with no dublicates. We will drop this, cause all these entries are unique given by the stakeholder to manage the contracts.

Data types - There are 151 numeric features, a lot of them with sparese entries (a lot of zeros). Also we have 18 object types.

Zero Values - As you can see a lot of the features contain high percentage of "0" values. "zon_community" for example is 100.00% zero although it has 3 kind of unique values. This is an example of how little elements have the different from zero characteristic.

5 Data Exploration - EDA

[79]:
```
fig = plt.figure()
ax = plt.gca()
df.drop('churn', axis=1).corrwith(df.churn).sort_values().
 ↪plot(kind='barh',figsize=(25, 30));
ax.tick_params(axis="x", bottom=True, top=True, labelbottom=True, labeltop=True)
fig.savefig('plots/churn_correlation_raw.pdf',bbox_inches='tight')
```



[89]:
```
def crosstab_evaluation(feature_column,target_column,relative=True):
    '''function to cross evaluate two features by a pandas cross table
    Inputs:
    feature_column: pandas Series of feature
    target_column: pandas Series of target (y-axis value)
    relative: False -> returns absolute values, True -> gives percentages

    Return:
    crosstable
    '''
    crosstable = pd.crosstab(feature_column,target_column)
    if relative:
        crosstable = crosstable.div(crosstable.sum(1),axis=0)
```

```
    return crosstable

def␣
 ↪crosstab_barplot(crosstable,labellist,figsize_x=10,figsize_y=7,xlabelname='Default'):
 ↪
    '''
    Function to plot a pandas crosstable.

    Inputs:
    crosstable: a pandas crosstable
    labellist: a list with the labels of the data
    xlabelname: Name of the x axis feature
    '''
    crosstable.plot(kind='bar', stacked=True,figsize=(figsize_x,figsize_y))
    plt.xlabel(xlabelname)
    plt.ylabel('Probability')

    #plt.title('Title');
    #plt.xticks(np.arange(2), ('60 months', '36 months',), fontsize=20)

    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])
```

5.1 Describe

[18]:
```
eda.describe_plus(df).head()
```

[18]:

	count	mean	std	min	25%	50%	75%	max	skew	kurtosis	variance
lesedauer	184660.0	33.06	21.76	0.0	14.0	28.0	48.0	88.0	0.62	-0.67	473.47
rechnungsmonat	184660.0	0.10	0.30	0.0	0.0	0.0	0.0	1.0	2.69	5.24	0.09
studentenabo	184660.0	0.11	0.31	0.0	0.0	0.0	0.0	1.0	2.48	4.15	0.10
metropole	184660.0	0.29	0.46	0.0	0.0	0.0	1.0	1.0	0.91	-1.18	0.21
shop_kauf	184660.0	0.47	2.15	0.0	0.0	0.0	0.0	152.0	13.24	424.71	4.60

5.2 Customer/Personal Features

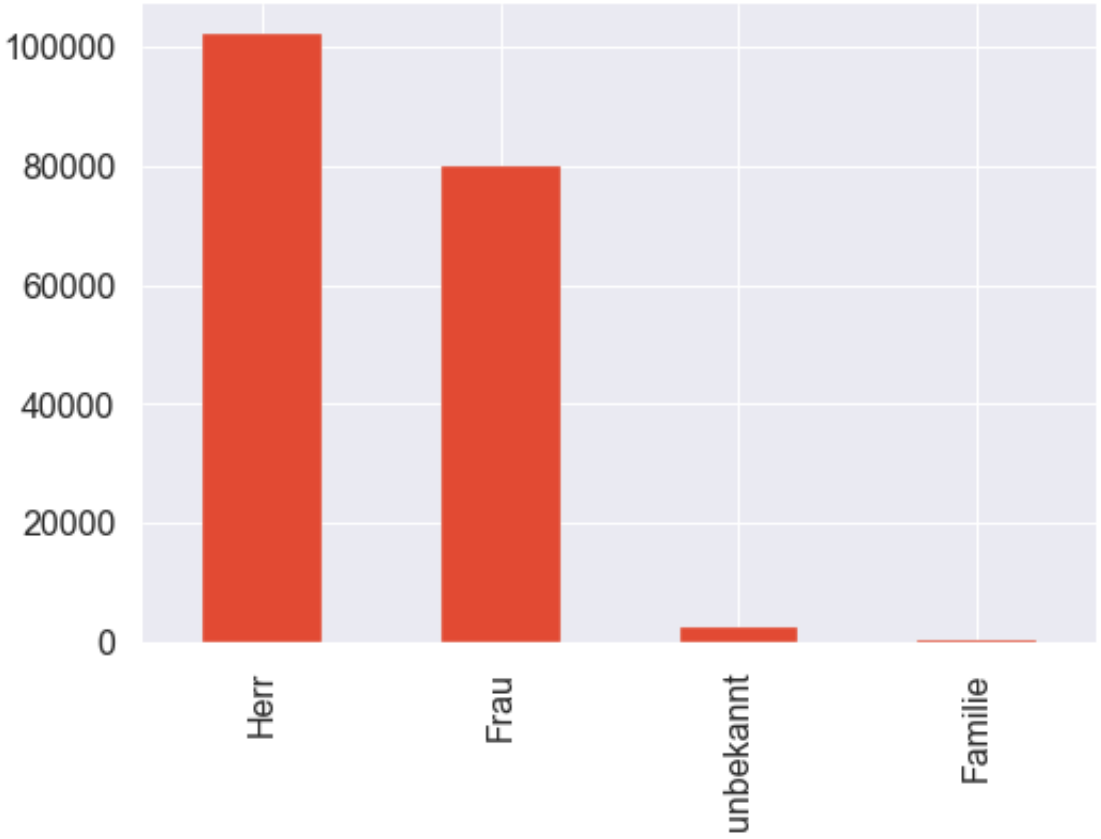
The following feature are considered as personal information and shortly described: - anrede: Mr, Mrs - titel: academic title - plz_1: first letter of zip code - plz_2: first two letters of zip code

- plz_3: first three letters of zip code - ort: city - metropole: is the city a metropolitan city - land_iso_code: DE, A, CH (Germany, Austria, Switzerland)

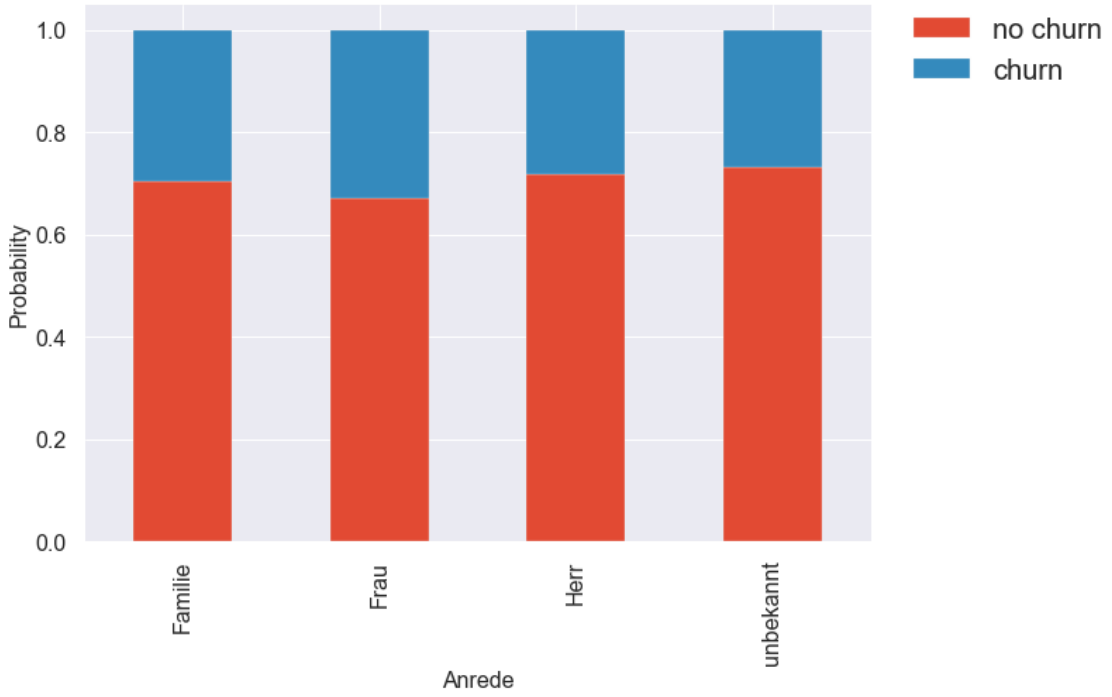
5.2.1 Anrede

[81]:
```
legendlist = ['churn','no churn']
```

[82]:
```
df.anrede.value_counts().plot(kind='bar');
```



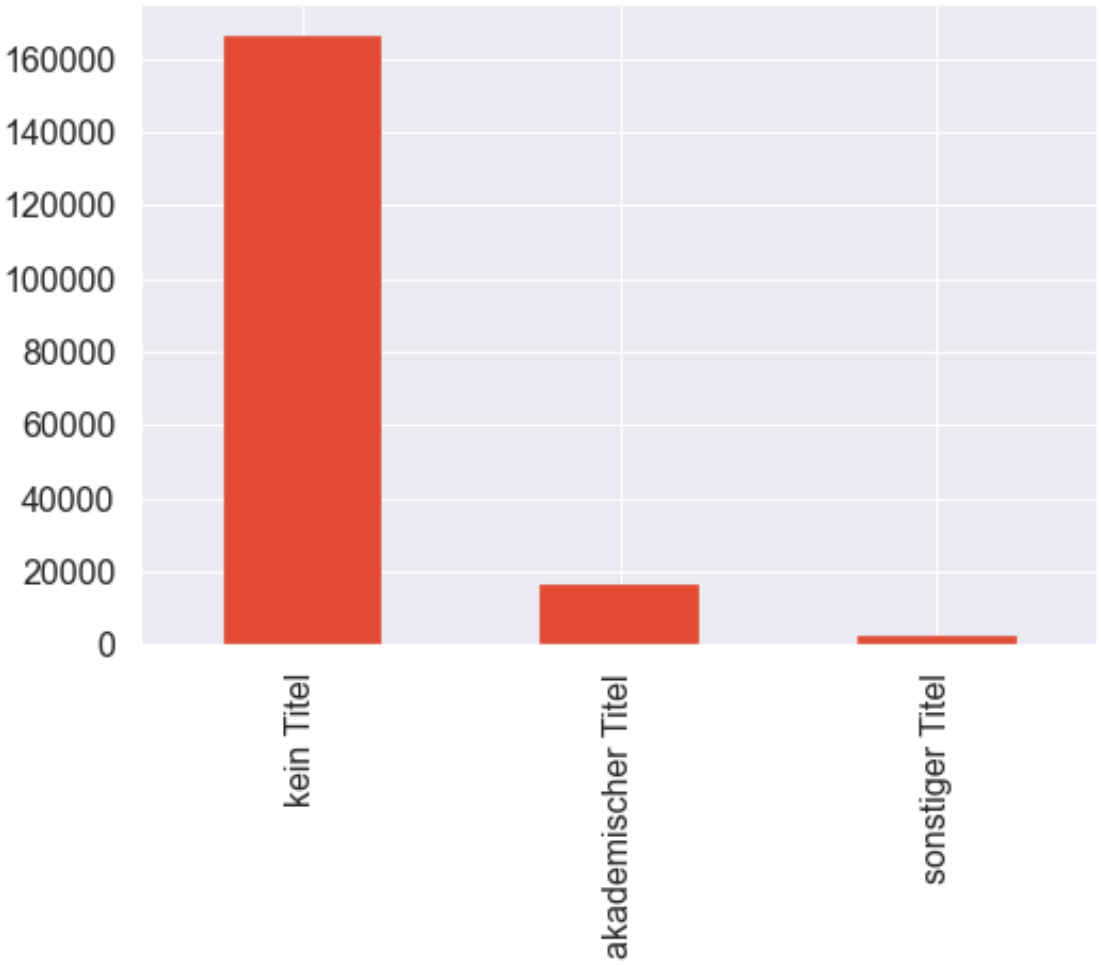
[90]:
```
anrede_churn = crosstab_evaluation(df.anrede,df.churn)
crosstab_barplot(anrede_churn,['no churn','churn'],xlabelname='Anrede')
plt.savefig('plots/anrede_churn.png',dpi=300,bbox_inches='tight')
```



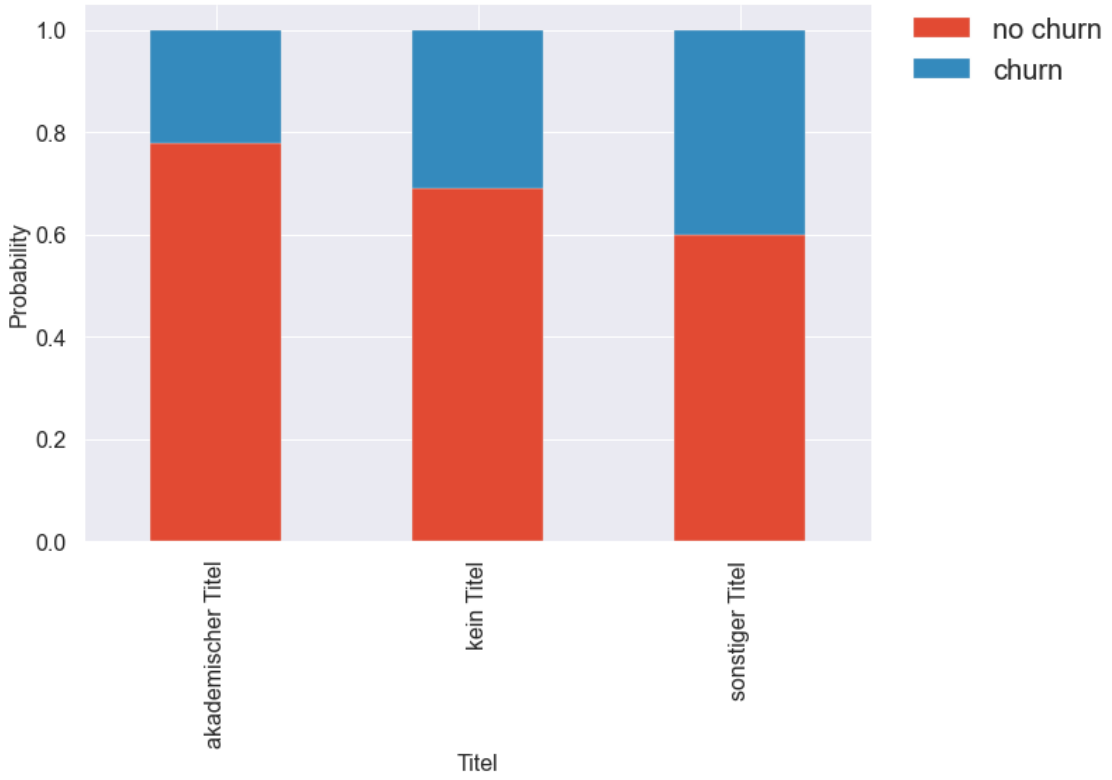
We can observe just small differences in the anrede, unknown and male customers seem to have a slightly higher churn probability.

5.2.2 Titel

[91]:
```
df.titel.value_counts().plot(kind='bar');
```



[92]:
```
titel_churn = crosstab_evaluation(df.titel,df.churn)
crosstab_barplot(titel_churn,['no churn','churn'],xlabelname='Titel')
plt.savefig('plots/titel_churn.png',dpi=300,bbox_inches='tight')
```



We can observer a smaller churn rate for academic titles compared to no title. The column other title (sonstiger Titel) is quite useless because we don't know hat kind of titles are included in there.

5.2.3 PLZ 1

First, get the average churn rate of the whole dataset:

[44]:
```
churn_prob_avg = df.churn.value_counts().values/df.shape[0]
churn_prob_true = churn_prob_avg[1]
churn_prob_true
```

[44]:
```
0.3031517383299036
```

Find the missing number of zip codes (xx) which belong to the foreign countries:

[24]:
```
df_zip_xx = df[df.plz_1 == 'xx']
print('Number of missing zip codes entries:',df_zip_xx.shape[0])
print('Percentage of missing zip codes in total:',df_zip_xx.shape[0]/df.
 ↪shape[0])
```

```
Number of missing zip codes entries: 23427
Percentage of missing zip codes in total: 0.1268655908155529
```

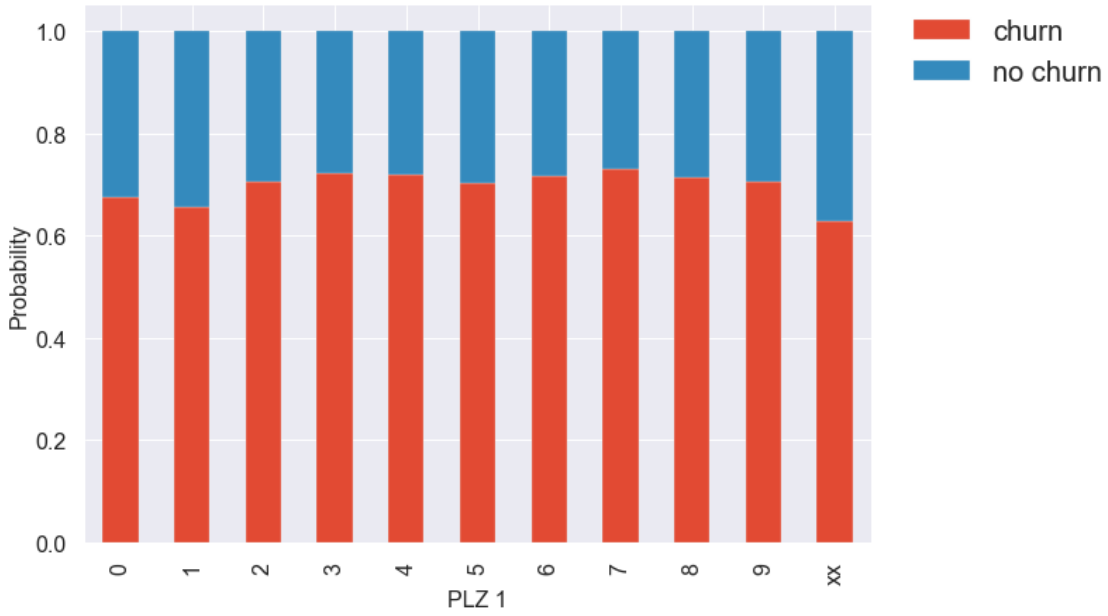
There are 19788 entries with no zip code, which is about 12 percent.

[25]:
```
df_zip_xx.ort.value_counts().nlargest(10)
```

[25]:
```
Wien          4725
Zürich        1687
Graz           708
Basel          694
Innsbruck      526
Salzburg       438
Bern           419
Linz           388
Winterthur     188
Luzern         153
Name: ort, dtype: int64
```

The xx zip codes are all non german cities (e.g. Switzerland), so we could divide the dataset into germany and out of germany areas.

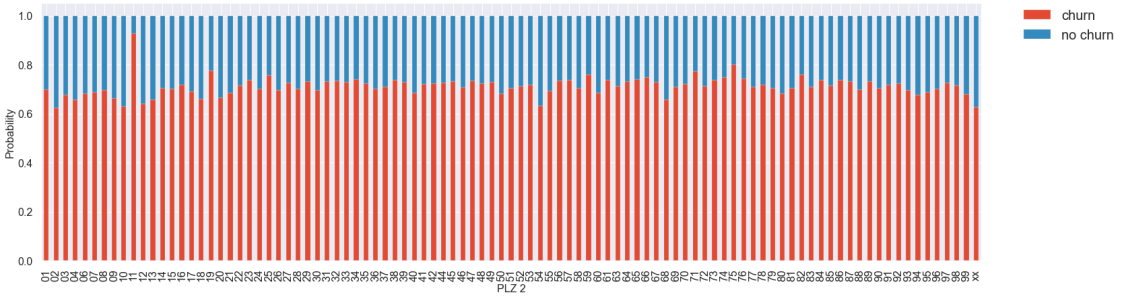
[93]:
```
plz1_churn = crosstab_evaluation(df.plz_1,df.churn)
crosstab_barplot(plz1_churn,['churn','no churn'],xlabelname='PLZ 1')
plt.savefig('plots/plz1_churn.png',dpi=300,bbox_inches='tight')
```



Simplified geographic features, here the first digit of the zip code are quite uselfull indicators for determine regions with higher churn rate.

5.2.4 PLZ 2

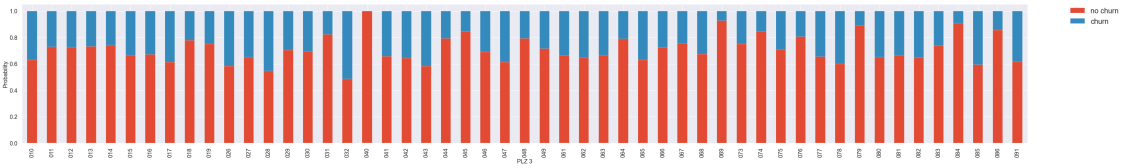
[94]:
```
plz2_churn = crosstab_evaluation(df.plz_2,df.churn)
crosstab_barplot(plz2_churn,['churn','no churn'],xlabelname='PLZ␣
↪2',figsize_x=25)
plt.savefig('plots/plz2_churn.png',dpi=300,bbox_inches='tight')
```



Compated to plz_1 we can see more details by this geographic information, the smearing of geographical data is less with two digits and we can have nice insights in the geographical distribution of our churns.

5.2.5 PLZ 3

[99]:
```
plz3_churn = crosstab_evaluation(df.plz_3,df.churn)
#plz3_churn.sort_values(by=1,ascending=False,inplace=True)
crosstab_barplot(plz3_churn,['no churn','churn'],xlabelname='PLZ␣
↪3',figsize_x=50)
plt.xlim(-0.5,50.5);
plt.savefig('plots/plz3_churn.png',dpi=300,bbox_inches='tight')
```

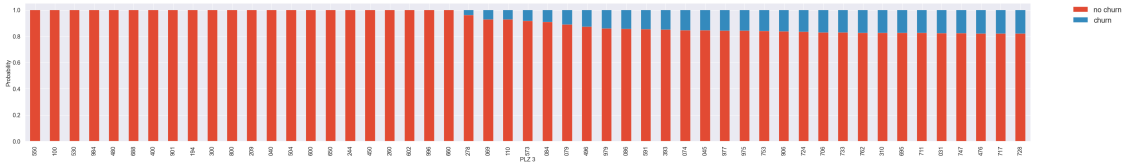


By using the 3 digit PLZ we can observe much higher variations in the churn rate.

Largest no churn:

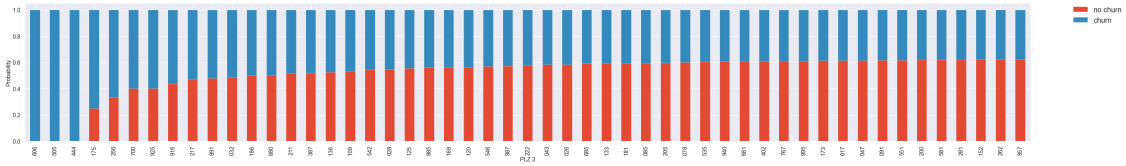
[100]:
```
plz3_churn = crosstab_evaluation(df.plz_3,df.churn)
plz3_churn.sort_values(by=1,ascending=True,inplace=True)
crosstab_barplot(plz3_churn,['no churn','churn'],xlabelname='PLZ␣
↪3',figsize_x=50)
plt.xlim(-0.5,50.5);
```

```
plt.savefig('plots/plz3_churn_largest.png',dpi=300,bbox_inches='tight')
```



Smallest no churn:

[101]:
```
plz3_churn = crosstab_evaluation(df.plz_3,df.churn)
plz3_churn.sort_values(by=1,ascending=False,inplace=True)
crosstab_barplot(plz3_churn,['no churn','churn'],xlabelname='PLZ␣
↪3',figsize_x=50)
plt.xlim(-0.5,50.5);
plt.savefig('plots/plz3_churn_smallest.png',dpi=300,bbox_inches='tight')
```



5.2.6 Geographical distribution with maps

Since this feature has a lot of entries, it is usefull to use a nice geographical visualization with geopandas. The link to geopandas for germany can be found here: https://juanitorduz.github.io/germany_plots/

[102]:
```
# geodata
plz_shape_df = gpd.read_file('data/plz-gebiete.shp', dtype={'plz': str})
plz_shape_df.info()
```

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 8706 entries, 0 to 8705
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   plz       8706 non-null   object
 1   note      8706 non-null   object
 2   geometry  8706 non-null   geometry
dtypes: geometry(1), object(2)
memory usage: 204.2+ KB
```

[46]:
```
top_cities = {
    'Berlin': (13.404954, 52.520008),
    'Cologne': (6.953101, 50.935173),
    'Düsseldorf': (6.782048, 51.227144),
    'Frankfurt am Main': (8.682127, 50.110924),
    'Hamburg': (9.993682, 53.551086),
    'Leipzig': (12.387772, 51.343479),
    'Munich': (11.576124, 48.137154),
    'Dortmund': (7.468554, 51.513400),
    'Stuttgart': (9.181332, 48.777128),
    'Nuremberg': (11.077438, 49.449820),
    'Hannover': (9.73322, 52.37052)
}
```

[47]:
```
plz_region_df = pd.read_csv(
    'data/zuordnung_plz_ort.csv',
    sep=',',
    dtype={'plz': str}
)

plz_region_df.drop('osm_id', axis=1, inplace=True)

plz_region_df.head()
```

[47]:

	ort	plz	bundesland
0	Aach	78267	Baden-Württemberg
1	Aach	54298	Rheinland-Pfalz
2	Aachen	52062	Nordrhein-Westfalen
3	Aachen	52064	Nordrhein-Westfalen
4	Aachen	52066	Nordrhein-Westfalen

[48]:
```
# Merge data.
germany_df = pd.merge(
    left=plz_shape_df,
    right=plz_region_df,
    on='plz',
    how='inner'
)

germany_df.drop(['note'], axis=1, inplace=True)
```

Mapping Functions to map churn probabilities for each subset of plz digits into a dataframe.

[49]:
```
def convert_plz_1_to_prob(plz):
    index = str(plz)[0]
    #print(index)
    value = plz1_churn.iloc[int(index),1]
```

```
    return value

def convert_plz_2_to_prob(plz):
    index = str(plz)[0:2]
    #print(index)
    value = plz2_churn[plz2_churn.index == index].iloc[0,1]
    #print(value)
    return value

def convert_plz_3_to_prob(plz):
    index = str(plz)[0:3]
    #print(index)
    value = plz3_churn[plz3_churn.index == index].iloc[0,1]
    #print(value)
    return value
```

[50]:
```
germany_df['churn_plz_1'] = germany_df.plz.apply(lambda x:␣
 ↪convert_plz_1_to_prob(x))
# subtract the average churn probability
germany_df['churn_plz_1'] = germany_df['churn_plz_1'] - churn_prob_true
```

[55]:
```
germany_df['churn_plz_2'] = germany_df.plz.apply(lambda x:␣
 ↪convert_plz_2_to_prob(x))
# subtract the average churn probability
germany_df['churn_plz_2'] = germany_df['churn_plz_2'] - churn_prob_true
```

[56]:
```
germany_df['churn_plz_3'] = germany_df.plz.apply(lambda x:␣
 ↪convert_plz_3_to_prob(x))
# subtract the average churn probability
germany_df['churn_plz_3'] = germany_df['churn_plz_3'] - churn_prob_true
```

Include the number of inhabitants in the dataframe:

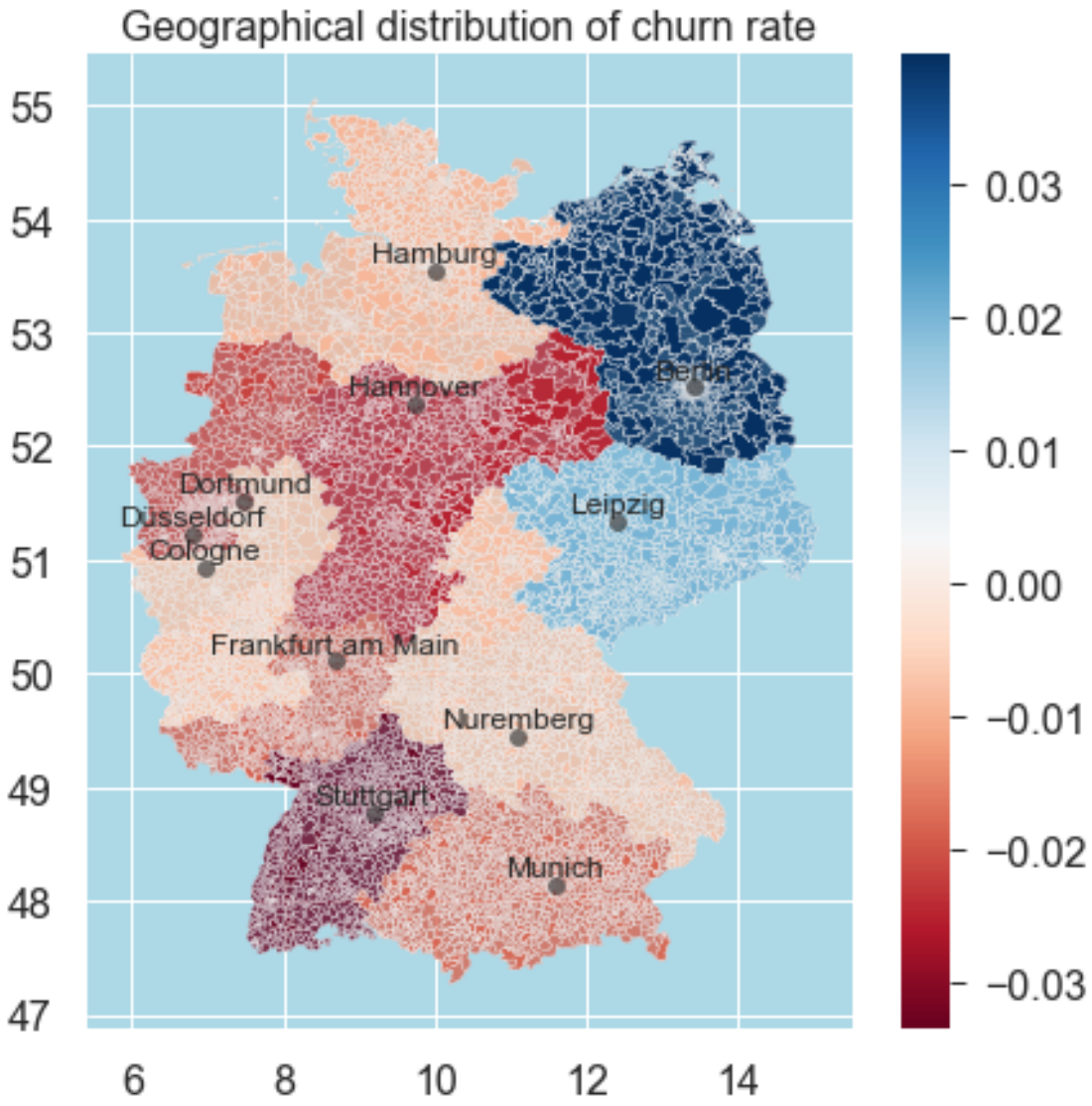
[57]:
```
plz_einwohner_df = pd.read_csv(
    'data/plz_einwohner.csv',
    sep=',',
    dtype={'plz': str, 'einwohner': int}
)

plz_einwohner_df.head()
```

[57]:

	plz	einwohner
0	01067	11957
1	01069	25491
2	01097	14811
3	01099	28021
4	01108	5876

[58]:
```
# Merge data.
germany_df = pd.merge(
    left=germany_df,
    right=plz_einwohner_df,
    on='plz',
    how='left'
)
```

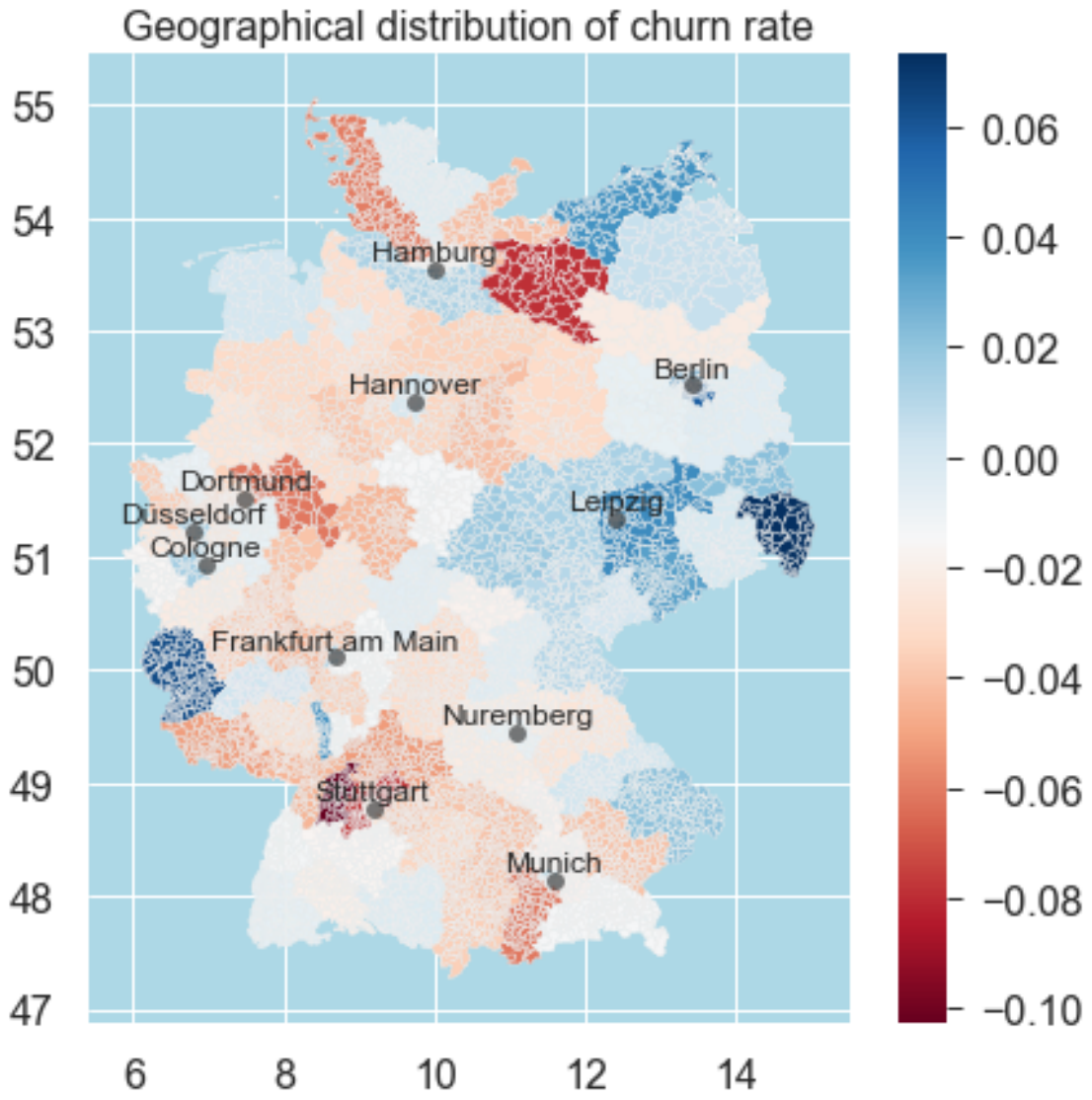
[103]:
```
fig, ax = plt.subplots(figsize=(7,7))

germany_df.plot(
    ax=ax,
    column='churn_plz_1',
    categorical=False,
    legend=True,
    #cmap='jet',
    cmap=plt.cm.RdBu,
    alpha=0.8,

)

for c in top_cities.keys():

    ax.text(
        x=top_cities[c][0],
        y=top_cities[c][1] + 0.08,
        s=c,
        fontsize=12,
        ha='center',
    )

    ax.plot(
        top_cities[c][0],
        top_cities[c][1],
        marker='o',
        c='black',
        alpha=0.5
    )

ax.set(
    title='Geographical distribution of churn rate',
    aspect=1.5,
    facecolor='lightblue'
);
fig.savefig('plots/churn_rate_landscape_plz_1_digit.png',dpi=300)
```



We can now visually see the higher churn rate in the eastern part of Germany.

[104]:
```
fig, ax = plt.subplots(figsize=(7,7))

germany_df.plot(
    ax=ax,
    column='churn_plz_2',
    categorical=False,
    legend=True,
    cmap=plt.cm.RdBu,
    alpha=1.0,

)
```

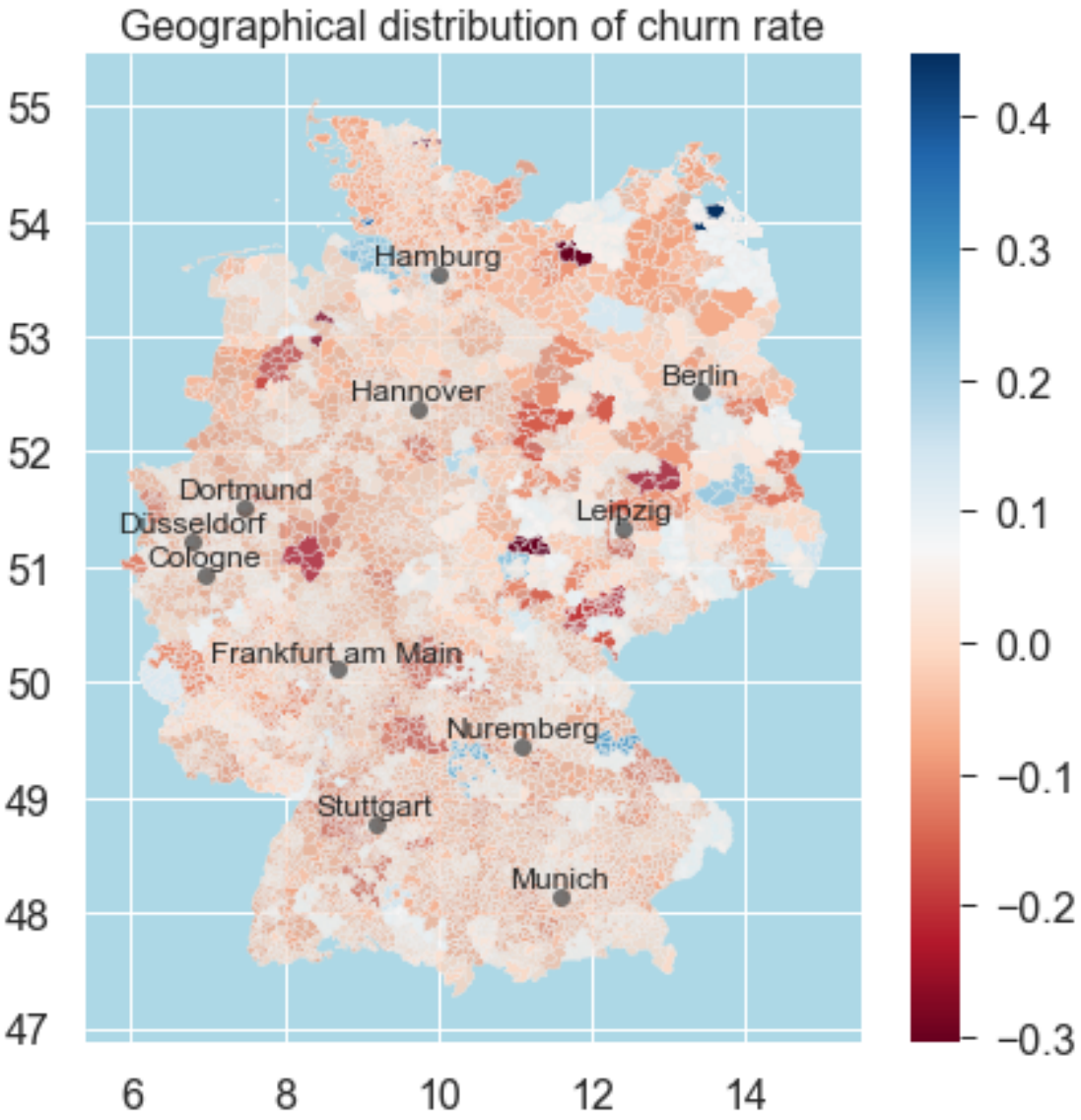
```
for c in top_cities.keys():

    ax.text(
        x=top_cities[c][0],
        y=top_cities[c][1] + 0.08,
        s=c,
        fontsize=12,
        ha='center',
    )

    ax.plot(
        top_cities[c][0],
        top_cities[c][1],
        marker='o',
        c='black',
        alpha=0.5
    )

ax.set(
    title='Geographical distribution of churn rate',
    aspect=1.5,
    facecolor='lightblue'
);
fig.savefig('plots/churn_rate_landscape_plz_2_digit.png',dpi=300)
```



Now we can more clearly see some smaller regions with much lower and much higher churn rates.

[105]:
```
fig, ax = plt.subplots(figsize=(7,7))

germany_df.plot(
    ax=ax,
    column='churn_plz_3',
    categorical=False,
    legend=True,
    cmap=plt.cm.RdBu,
    alpha=0.8,

)
```

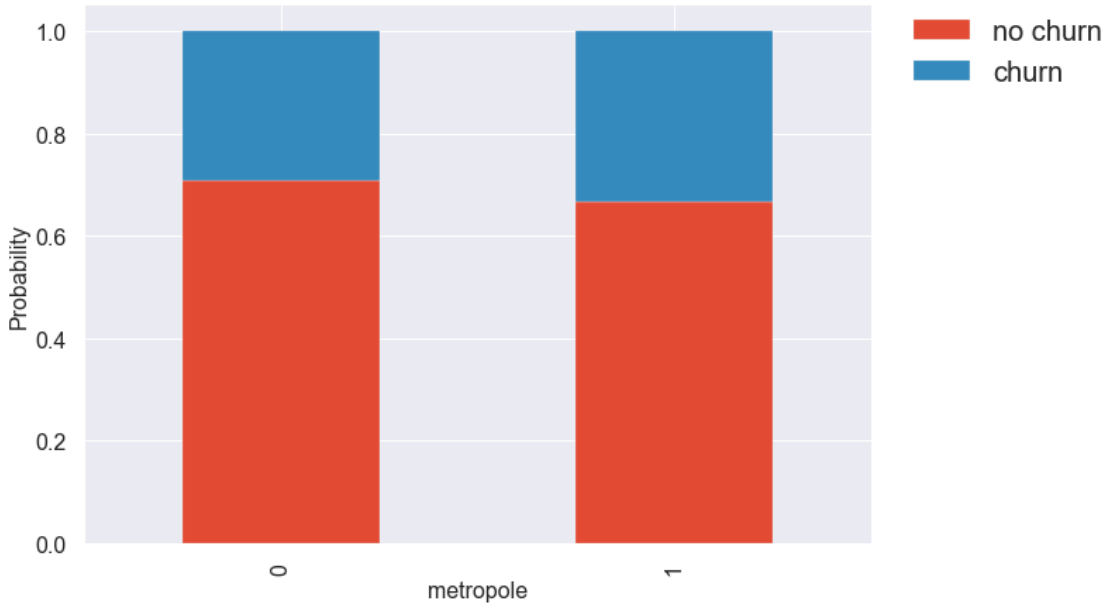
```
for c in top_cities.keys():

    ax.text(
        x=top_cities[c][0],
        y=top_cities[c][1] + 0.08,
        s=c,
        fontsize=12,
        ha='center',
    )

    ax.plot(
        top_cities[c][0],
        top_cities[c][1],
        marker='o',
        c='black',
        alpha=0.5
    )

ax.set(
    title='Geographical distribution of churn rate',
    aspect=1.5,
    facecolor='lightblue'
);
fig.savefig('plots/churn_rate_landscape_plz_3_digit.png',dpi=300)
```



This gives us a very detailled overview of the geographical distribution of the churn rate, with much less flattening of the data. We can observe areas with almost zero churn rate and regions with more than 60 percent churn rate.

5.2.7 Metropole

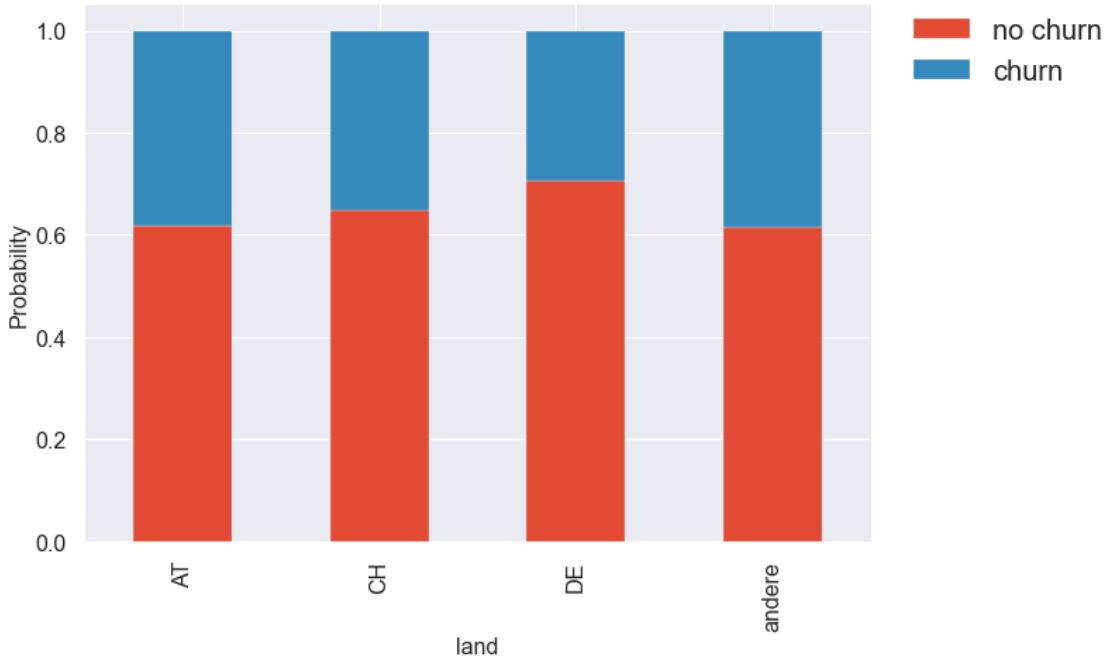
[106]:
```
metropole_churn = crosstab_evaluation(df.metropole,df.churn)
crosstab_barplot(metropole_churn,['no churn','churn'],xlabelname='metropole')
plt.savefig('plots/metropolitan_churn.png',dpi=300,bbox_inches='tight')
```



No significant influence if either a metropolitan city or not is found.

5.2.8 Land iso code

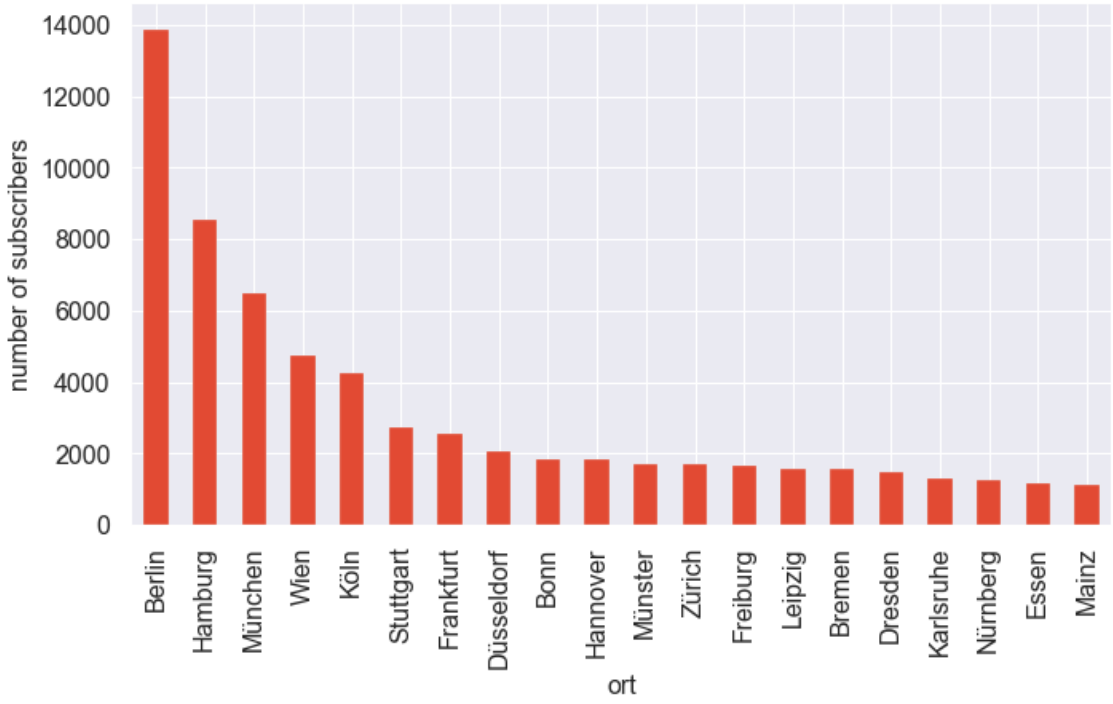
[107]:
```
land_iso_churn = crosstab_evaluation(df.land_iso_code,df.churn)
crosstab_barplot(land_iso_churn,['no churn','churn'],xlabelname='land')
plt.savefig('plots/land_iso_code_churn.png',dpi=300,bbox_inches='tight')
```



Germany has the highest churn rate, people from abroad (Austria (AT), Switzerland (CH) and other countries abroad) tend to churn with a lower probability. People from abroad who are subscribers tend to be more commited to their subscription.

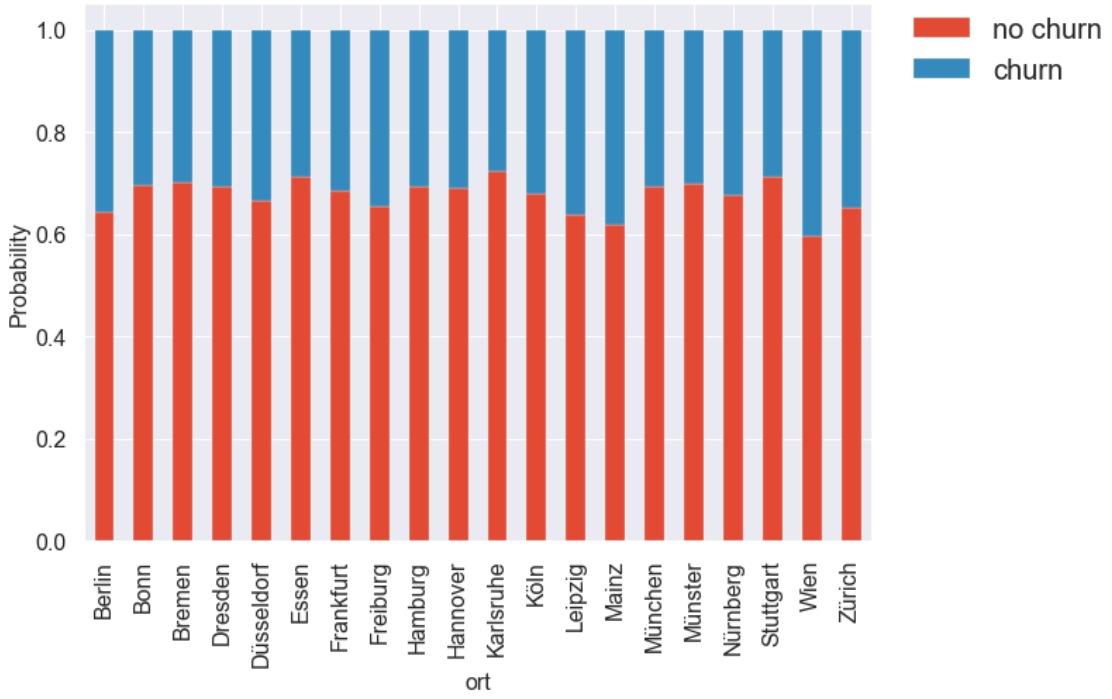
5.2.9 Ort/City

[74]:
```
a = df.groupby('ort').size().nlargest(20)
a.plot(kind='bar',ylabel='number of subscribers',figsize=(11,6));
```



The Zeit Top City is Berlin, followed by Hamburg and Munich. Vienna surpsisingly has a quite high number of subscribers of about 4000.

[108]:
```
b = list(a.index)
df_top_cities = df[df['ort'].apply((lambda x: x in b))]

ort_churn = crosstab_evaluation(df_top_cities.ort,df_top_cities.churn)
crosstab_barplot(ort_churn,['no churn','churn'],xlabelname='ort')
plt.savefig('plots/top20_cities_churn.png',dpi=300,bbox_inches='tight')
```



The churn rate varies over the twenty largest cities.

5.2.10 Summary Customer Related Features

To sum up, the following was found about the customer related features:

- **anrede**: Small differences: keep feature.
- **titel**: Smaller churn rate for academic titles compared to no title: keep feature
- **plz_1, plz_2, plz_3**: The plz code gives us a very detailled overview of the geographical distribution of the churn rate: keep feature

- **ort:** Geographic information, but similar to plz: should be dropped
- **metropole:** Just slight difference in churn rate: keep feature
- **land_iso_code:** There is a country influence on the churn rate: keep feature

5.3 Subscription Features

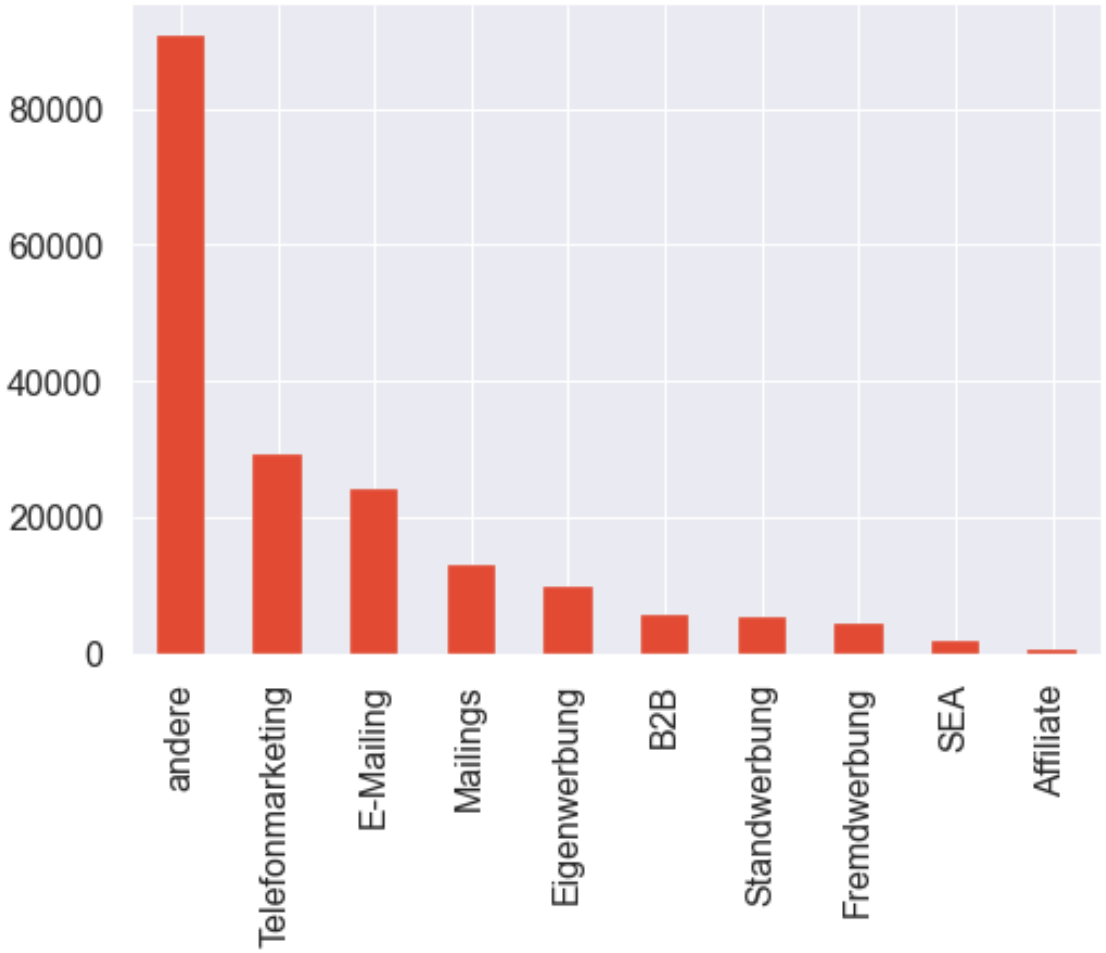
The following features are related to the subscription kind and shortly described:

- kanal
- objekt_name
- aboform_name
- zahlung_rhythmus_name
- rechnungsmonat
- zahlung_weg_name
- studentenabo

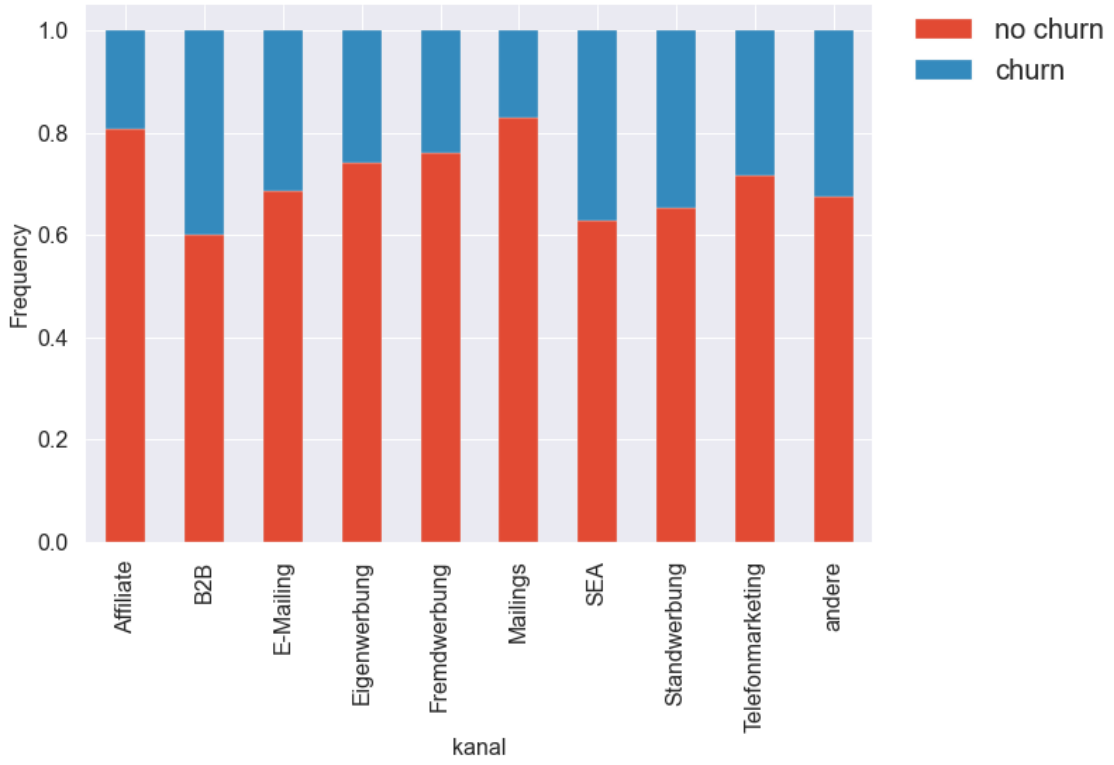
- unterbrechung

5.3.1 Kanal

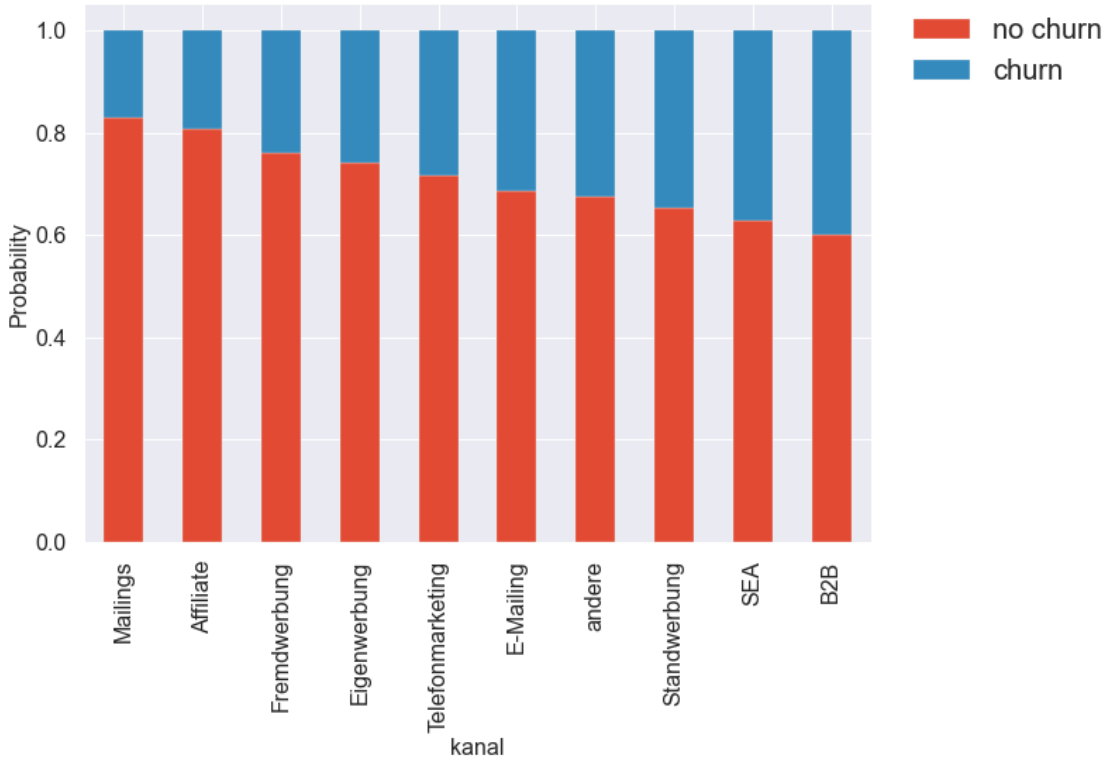
[76]:
```
df.kanal.value_counts().plot(kind='bar');
```



[77]:
```
kanal_churn = crosstab_evaluation(df.kanal,df.churn)
# sort crosstable by churn probability
#kanal_churn.sort_values(by=1,ascending=True,inplace=True)
crosstab_barplot(kanal_churn,['no churn','churn'],xlabelname='kanal')
```



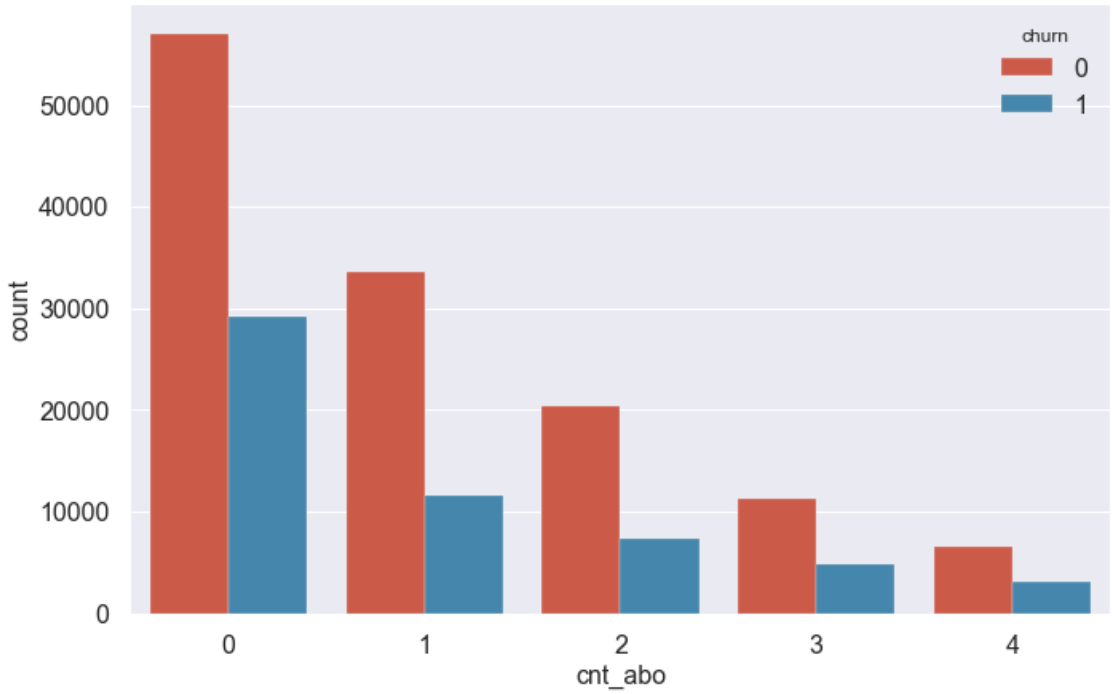
[109]:
```
kanal_churn = crosstab_evaluation(df.kanal,df.churn)
# sort crosstable by churn probability
kanal_churn.sort_values(by=1,ascending=True,inplace=True)
crosstab_barplot(kanal_churn,['no churn','churn'],xlabelname='kanal')
plt.savefig('plots/channel_sorted_churn.png',dpi=300,bbox_inches='tight')
```



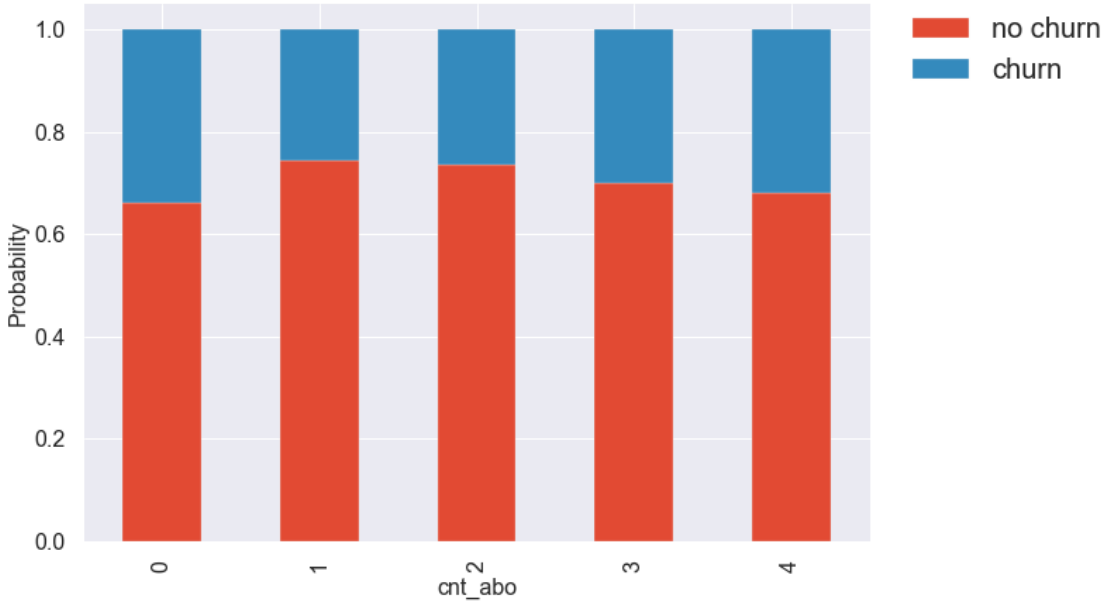
The channel of the subrscitpion is an important feature for churn. The churn rate increases from less than 20 percent for mailings to almost 40 percent for SEA (Search Engine Advertising).

5.3.2 cnt_abo

[111]:
```
fig, ax = plt.subplots(figsize=(11,7))
ax = sns.countplot(x="cnt_abo", hue='churn',data=df)
plt.savefig('plots/abo_cnt_churn.png',dpi=300,bbox_inches='tight')
```

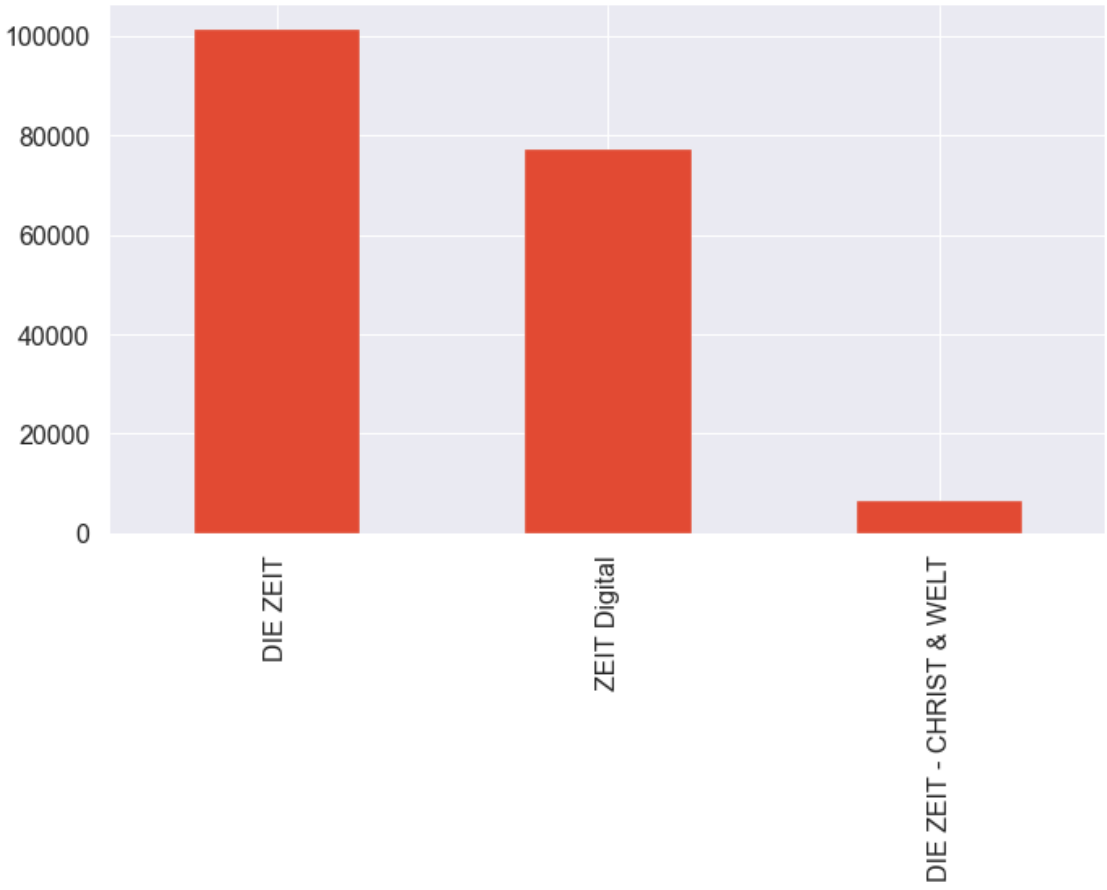


[115]:
```
abo_cnt_churn = crosstab_evaluation(df.cnt_abo,df.churn)
# sort crosstable by churn probability
#abo_cnt_churn.sort_values(by=1,ascending=True,inplace=True)
crosstab_barplot(abo_cnt_churn,['no churn','churn'],xlabelname='cnt_abo')
plt.savefig('plots/cnt_abo_relative_prob_churn.png',dpi=300,bbox_inches='tight')
```

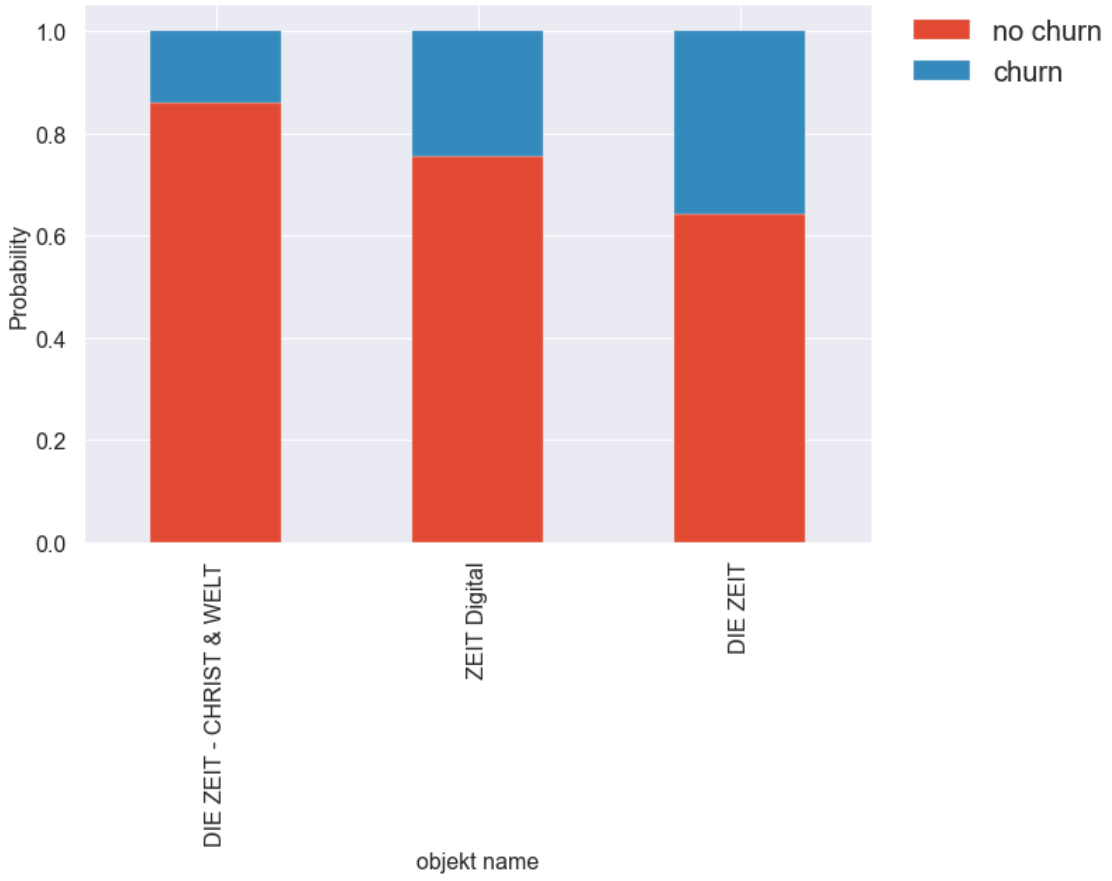


5.3.3 objekt_name

[50]:
```
df.objekt_name.value_counts().plot(kind='bar',figsize=(11,6));
```

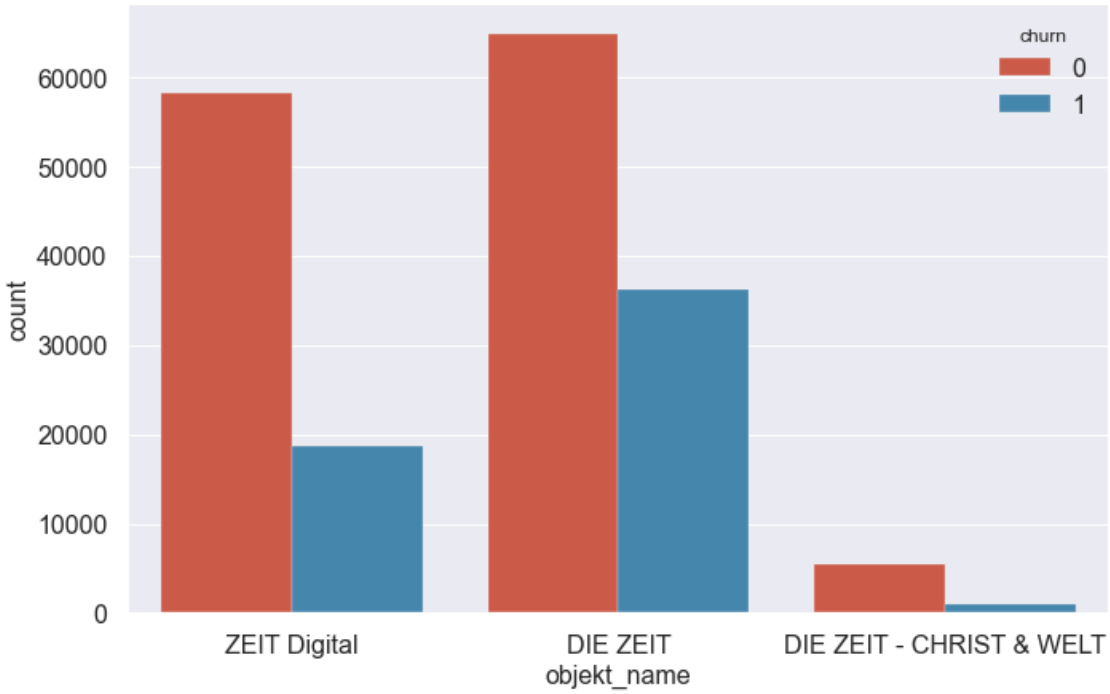


[116]:
```
objekt_churn = crosstab_evaluation(df.objekt_name,df.churn)
objekt_churn.sort_values(by=1,ascending=True,inplace=True)
crosstab_barplot(objekt_churn,['no churn','churn'],xlabelname='objekt name')
plt.savefig('plots/subscription_kind_churn.png',dpi=300,bbox_inches='tight')
```



Combined subscription of Die Zeit with Christ & Welt has a remarkably smaller churn rate than Zeit Digital and Die Zeit print.

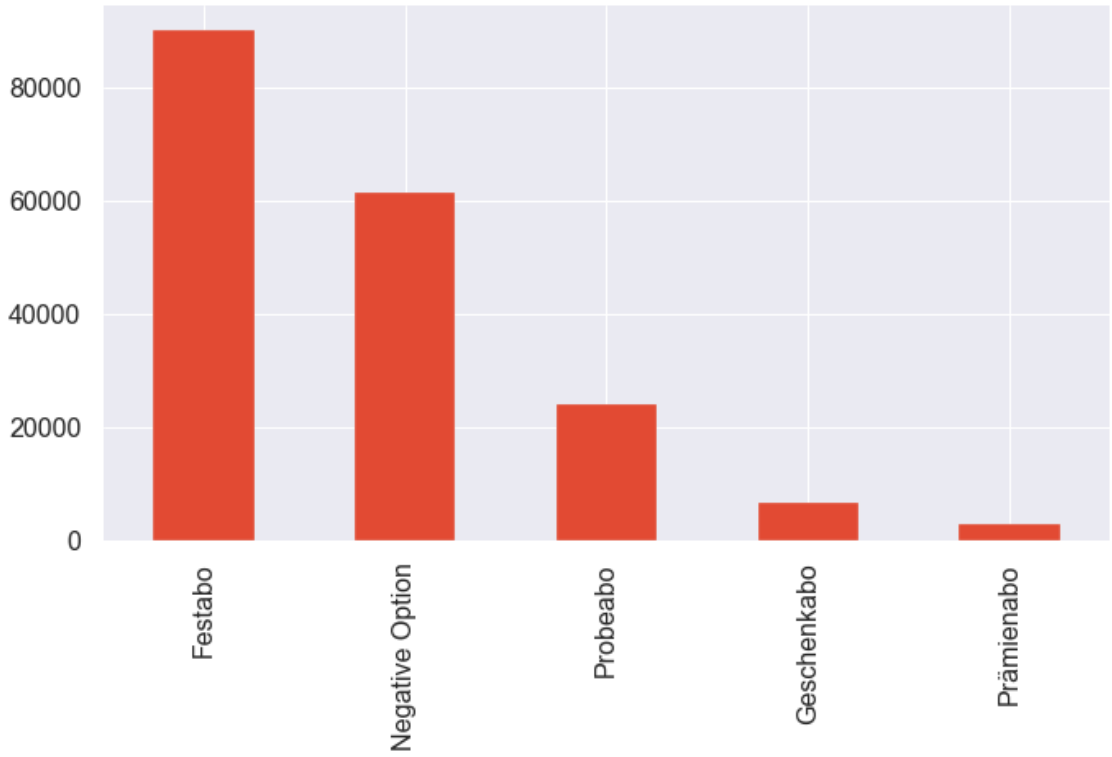
[117]:
```
fig, ax = plt.subplots(figsize=(11,7))
ax = sns.countplot(x="objekt_name", hue='churn',data=df)
```



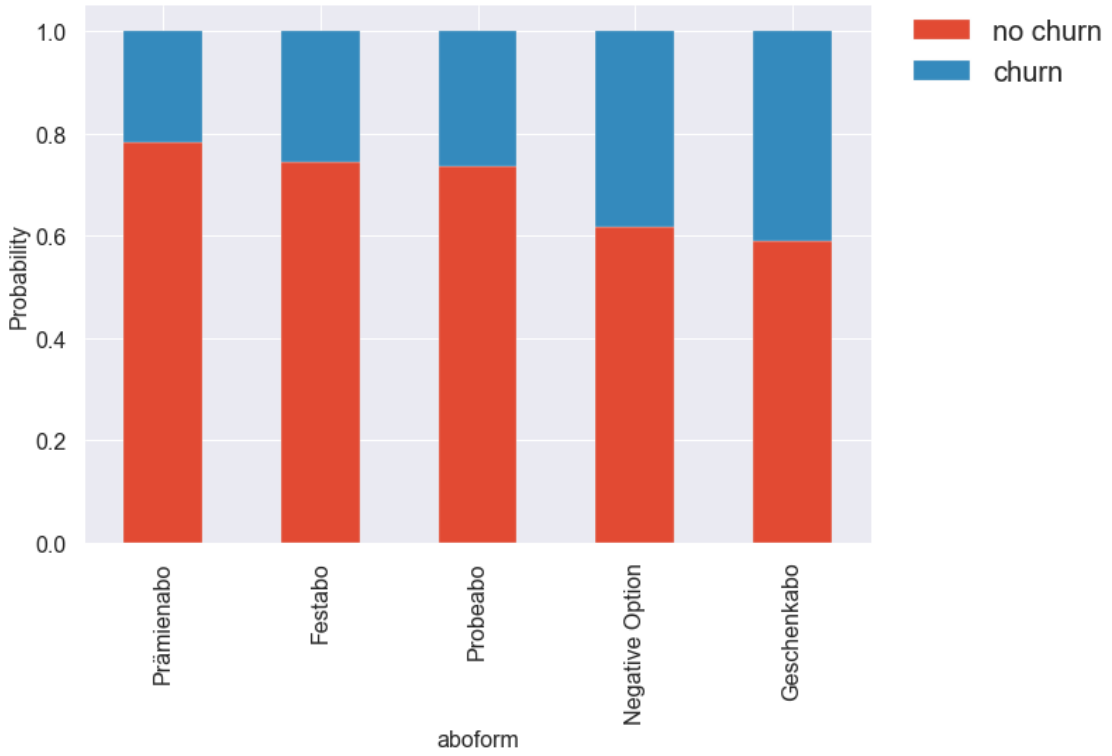
A small customer segmentation study with different machine learning methods was done for the two groups "Zeit Digital" and "Die Zeit" to see if machine learning methods do a better prediction, when the customers are separated by the subscription type. This can be found in the Customer Segmentation Notebook.

5.3.4 aboform_name

[53]:
```
df.aboform_name.value_counts().plot(kind='bar',figsize=(11,6));
```



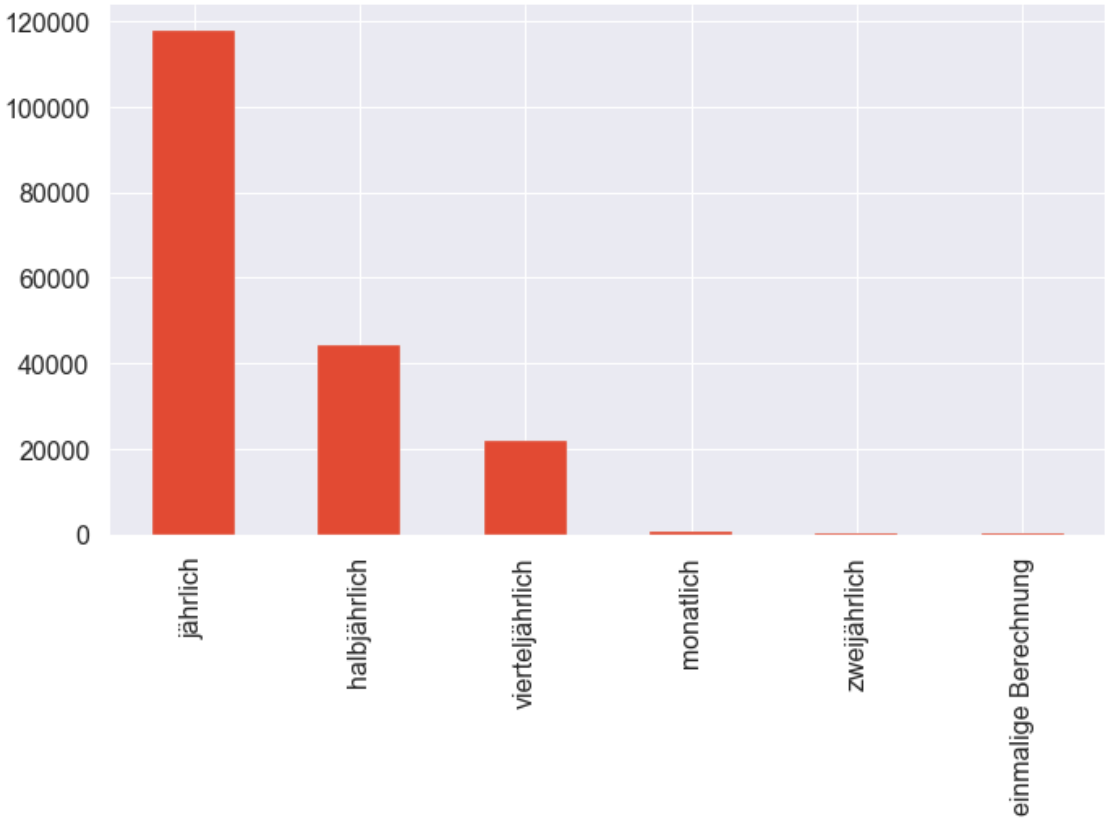
[118]:
```
aboform_churn = crosstab_evaluation(df.aboform_name,df.churn)
aboform_churn.sort_values(by=1,ascending=True,inplace=True)
crosstab_barplot(aboform_churn,['no churn','churn'],xlabelname='aboform')
plt.savefig('plots/abo_type_churn.png',dpi=300,bbox_inches='tight')
```



Also here, we can see that Prämienabo has a much smaller churn rate than others, in particular Geschenkabo and Negative Option.

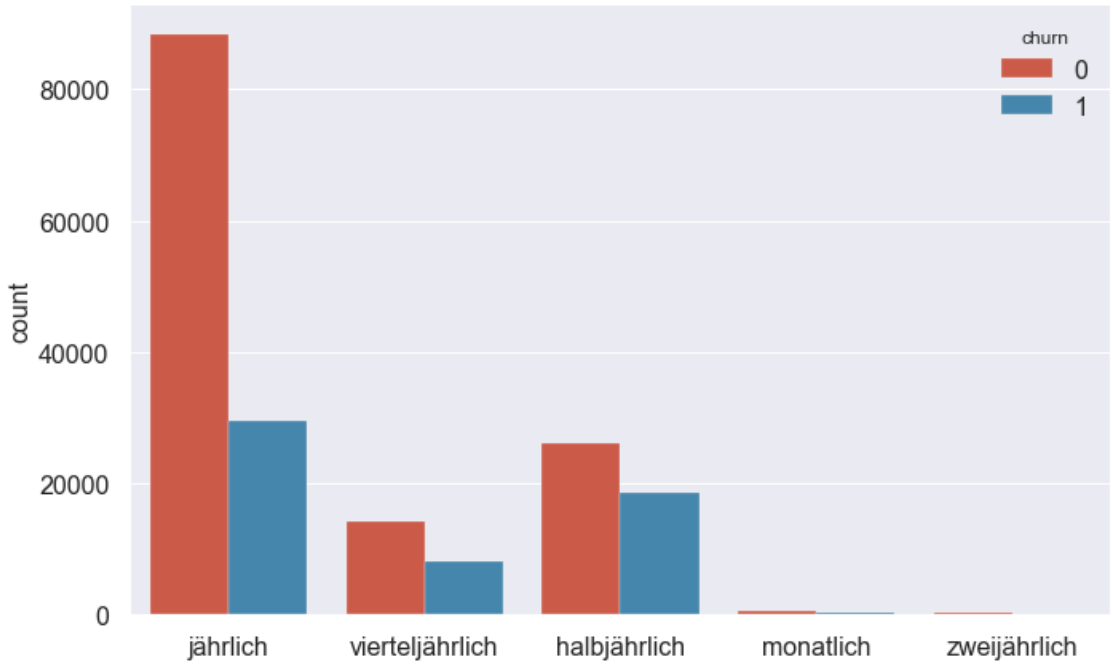
5.3.5 zahlung_rhythmus_name

[55]:
```
df.zahlung_rhythmus_name.value_counts().plot(kind='bar',figsize=(11,6));
```

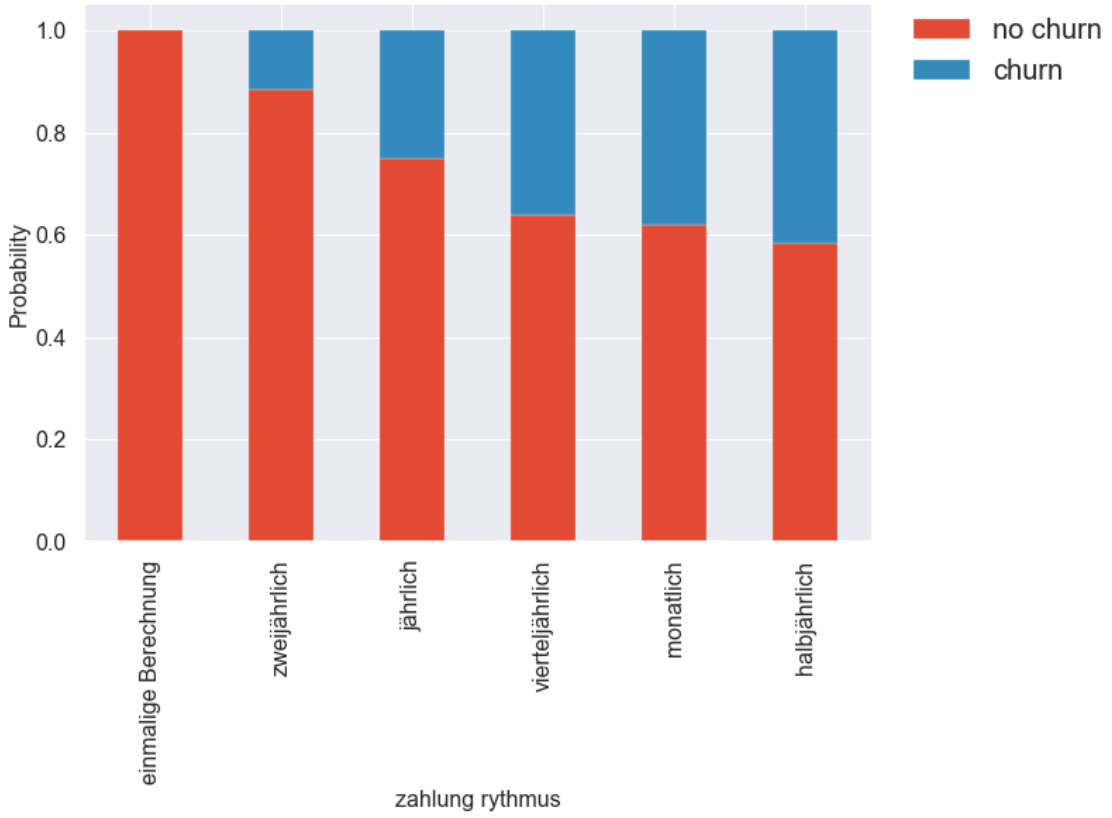


[121]:
```
fig, ax = plt.subplots(figsize=(11,7))
ax = sns.countplot(x='zahlung_rhythmus_name',data=df,hue='churn')
ax.set_xlabel('');
plt.xlim(-0.5,4.5)
```

[121]: (-0.5, 4.5)



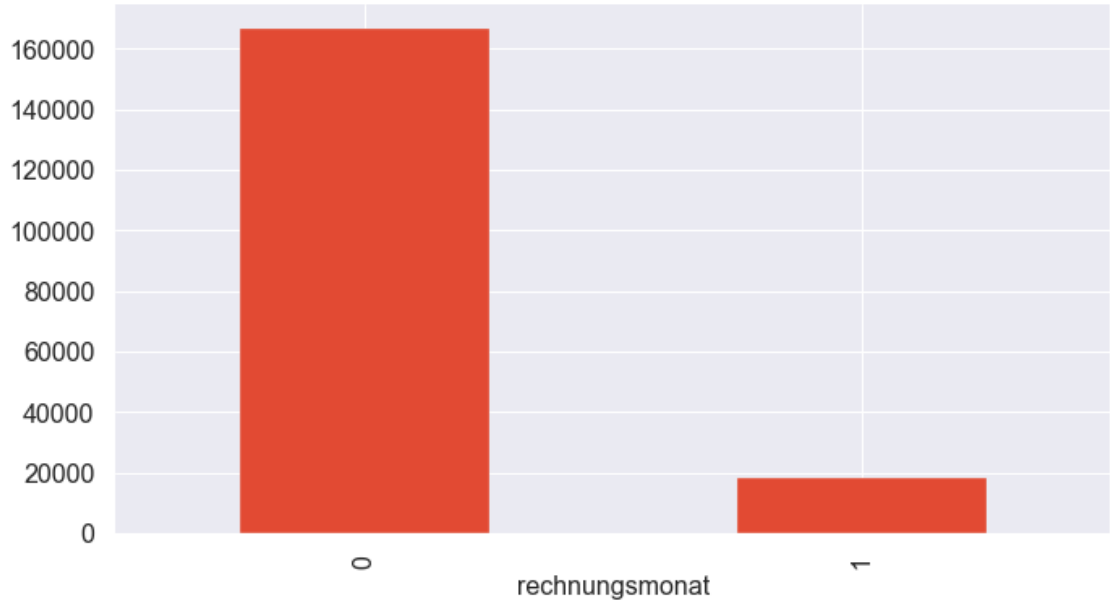
[122]:
```
zahlung_rythmus_churn = crosstab_evaluation(df.zahlung_rhythmus_name,df.churn)
zahlung_rythmus_churn.sort_values(by=1,inplace=True,ascending=True)
crosstab_barplot(zahlung_rythmus_churn,['no churn','churn'],xlabelname='zahlung␣
↪rythmus')
plt.savefig('plots/payment_rhythm_churn.png',dpi=300,bbox_inches='tight')
```



We can observe a strong dependence of the payment period on the churn rate. Payments with less than one year period tend to have higher churn rates.

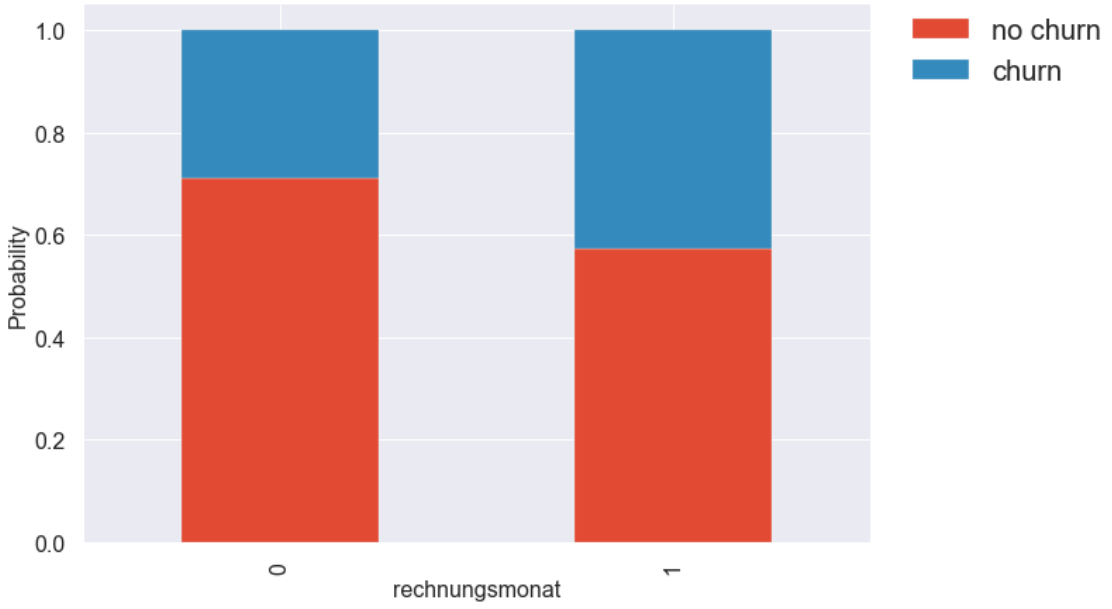
5.3.6 rechnungsmonat

[58]:
```
df.rechnungsmonat.value_counts().plot(kind='bar',figsize=(11,6));
plt.xlabel('rechnungsmonat');
```



Rechnungsmonat? What is the meaning?

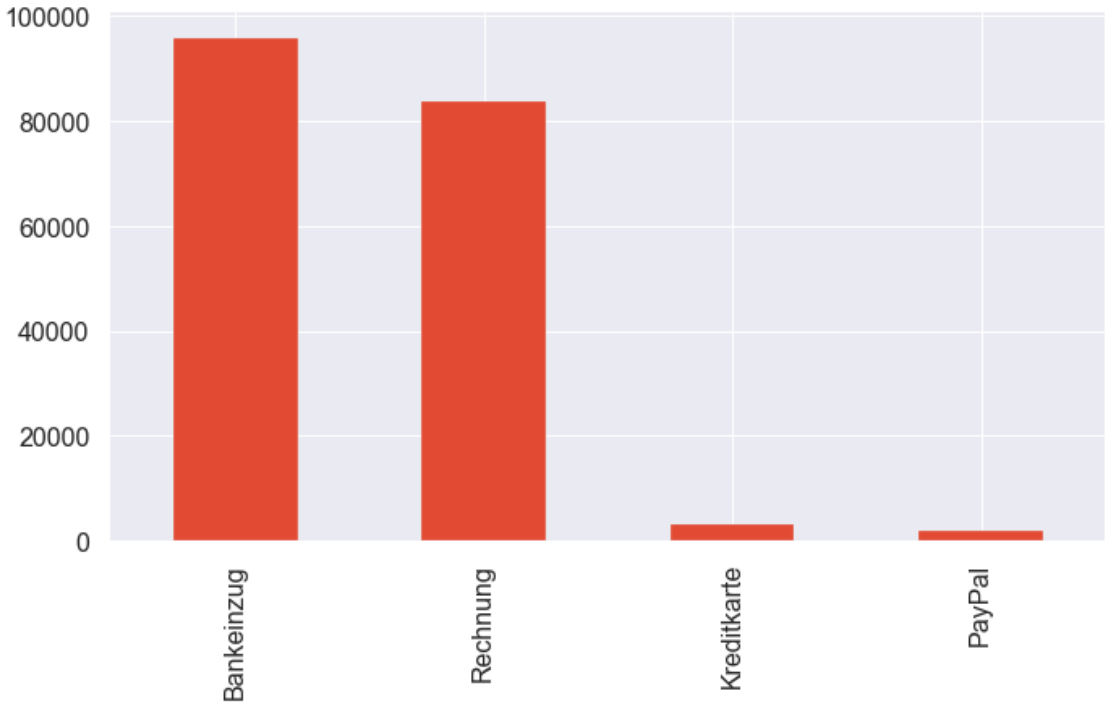
[123]:
```
rechnungsmonat_churn = crosstab_evaluation(df.rechnungsmonat,df.churn)
crosstab_barplot(rechnungsmonat_churn,['no␣
↪churn','churn'],xlabelname='rechnungsmonat')
plt.savefig('plots/bill_month_churn.png',dpi=300,bbox_inches='tight')
```



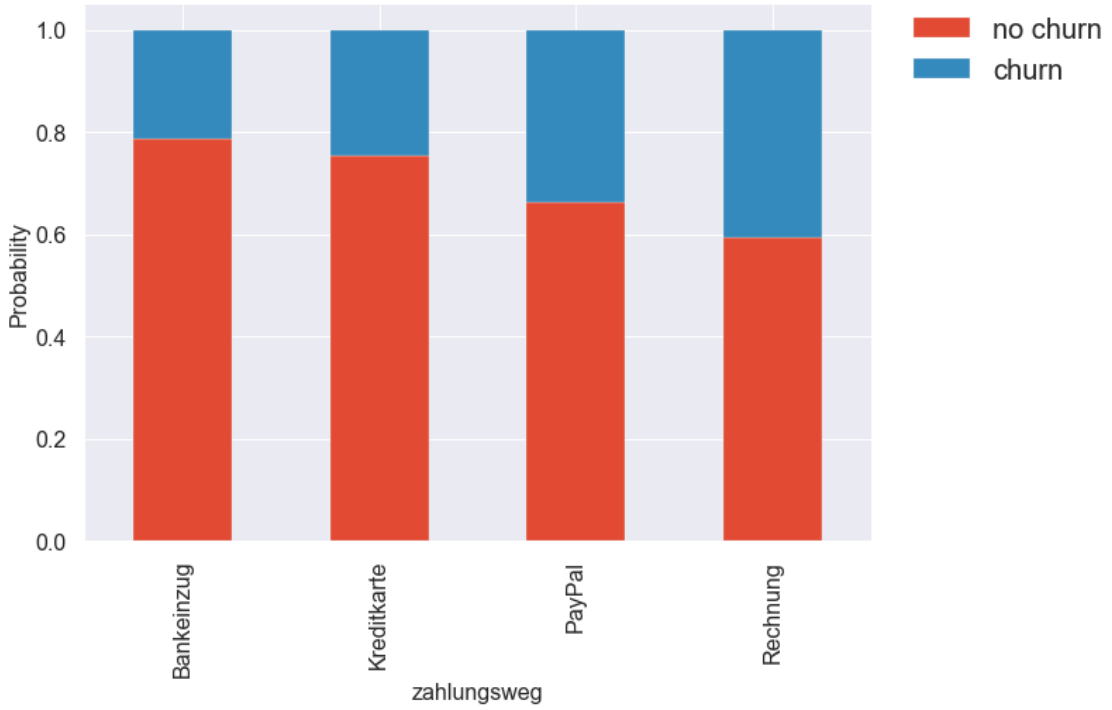
If there is a billing months, a subscriber has a higher tendency to churn the subscription than without a billing month.

5.3.7 zahlung_weg_name

[60]:
```
df.zahlung_weg_name.value_counts().plot(kind='bar',figsize=(11,6));
```



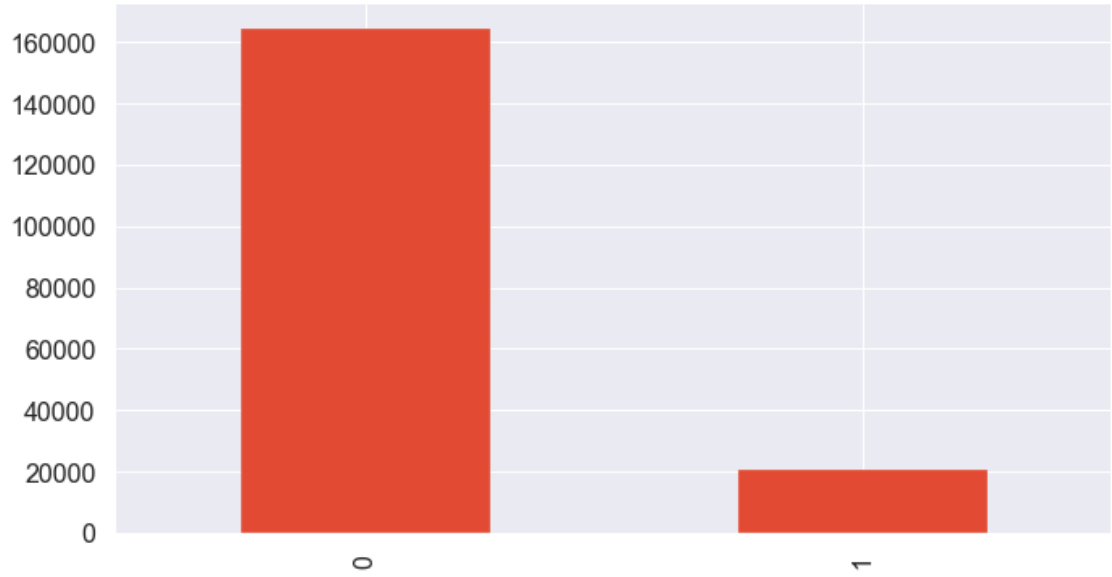
[124]:
```
zahlungsweg_churn = crosstab_evaluation(df.zahlung_weg_name,df.churn)
crosstab_barplot(zahlungsweg_churn,['no␣
↪churn','churn'],xlabelname='zahlungsweg')
plt.savefig('plots/payment_type_churn.png',dpi=300,bbox_inches='tight')
```



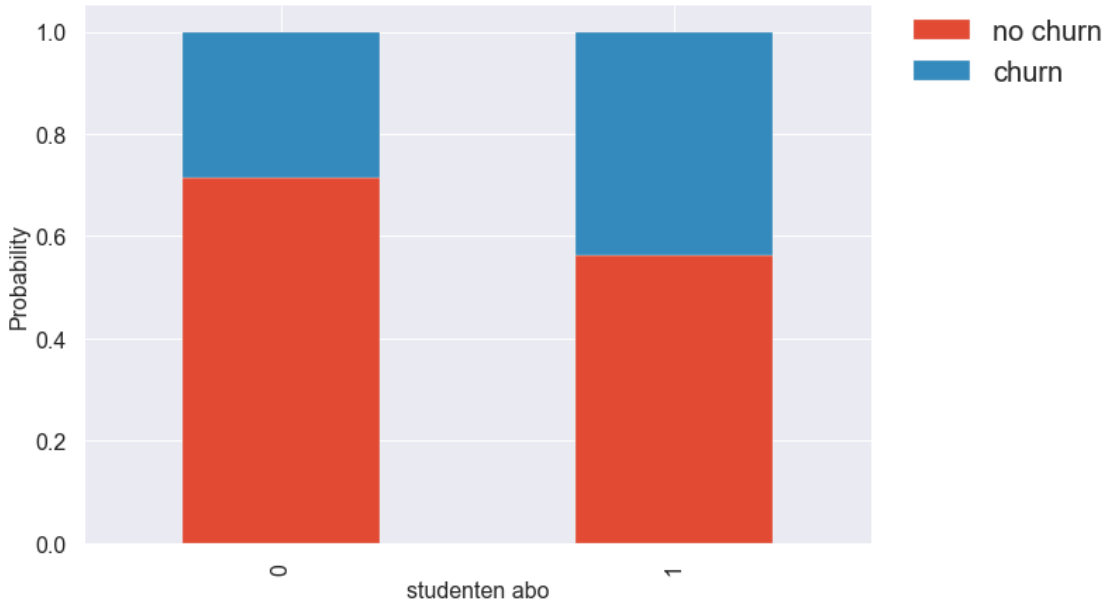
If the payment is made by direct debit (Bankeinzug), the churn rate is much lower than if the payment is made by invoice. Credit Cards and Paypal are in between.

5.3.8 studentenabo

[62]:
```
df.studentenabo.value_counts().plot(kind='bar',figsize=(11,6));
```



[125]:
```
studentenabo_churn = crosstab_evaluation(df.studentenabo,df.churn)
crosstab_barplot(studentenabo_churn,['no churn','churn'],xlabelname='studenten␣
↪abo')
plt.savefig('plots/student_churn.png',dpi=300,bbox_inches='tight')
```



The students tend to churn more frequently.

Student Map

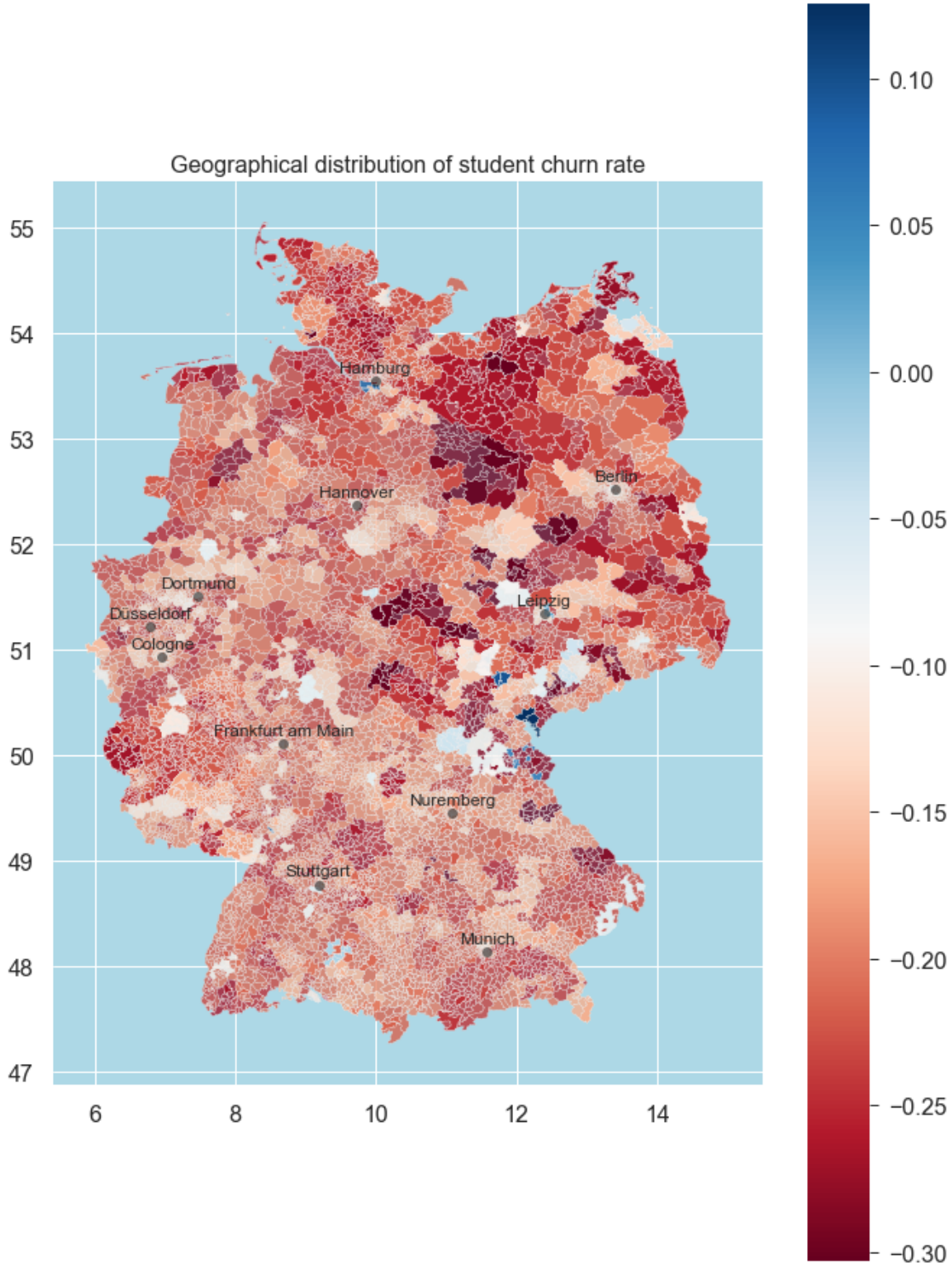
[126]:
```
plz3_students = crosstab_evaluation(df.plz_3,df.studentenabo)
```

[127]:
```
def convert_plz_3_to_student(plz):
    index = str(plz)[0:3]
    value = plz3_students[plz3_students.index == index].iloc[0,1]
    return value
```

[128]:
```
germany_df['studenten_plz_3'] = germany_df.plz.apply(lambda x:␣
↪convert_plz_3_to_student(x))
germany_df['studenten_plz_3'] = germany_df['studenten_plz_3'] - churn_prob_true
```

[129]:
```
fig, ax = plt.subplots(figsize=(11,16))

germany_df.plot(
    ax=ax,
    column='studenten_plz_3',
    categorical=False,
```

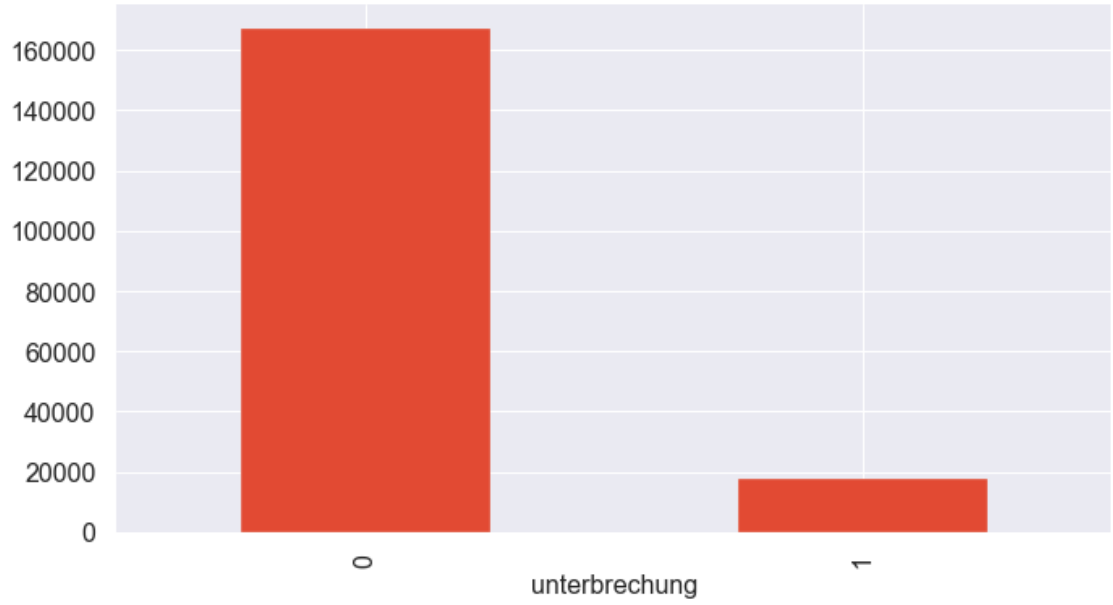
```
    legend=True,
    cmap=plt.cm.RdBu,
    alpha=0.8,

)

for c in top_cities.keys():

    ax.text(
        x=top_cities[c][0],
        y=top_cities[c][1] + 0.08,
        s=c,
        fontsize=12,
        ha='center',
    )

    ax.plot(
        top_cities[c][0],
        top_cities[c][1],
        marker='o',
        c='black',
        alpha=0.5
    )

ax.set(
    title='Geographical distribution of student churn rate',
    aspect=1.5,
    facecolor='lightblue'
);
fig.savefig('plots/student_churn_rate.png',dpi=300)
```



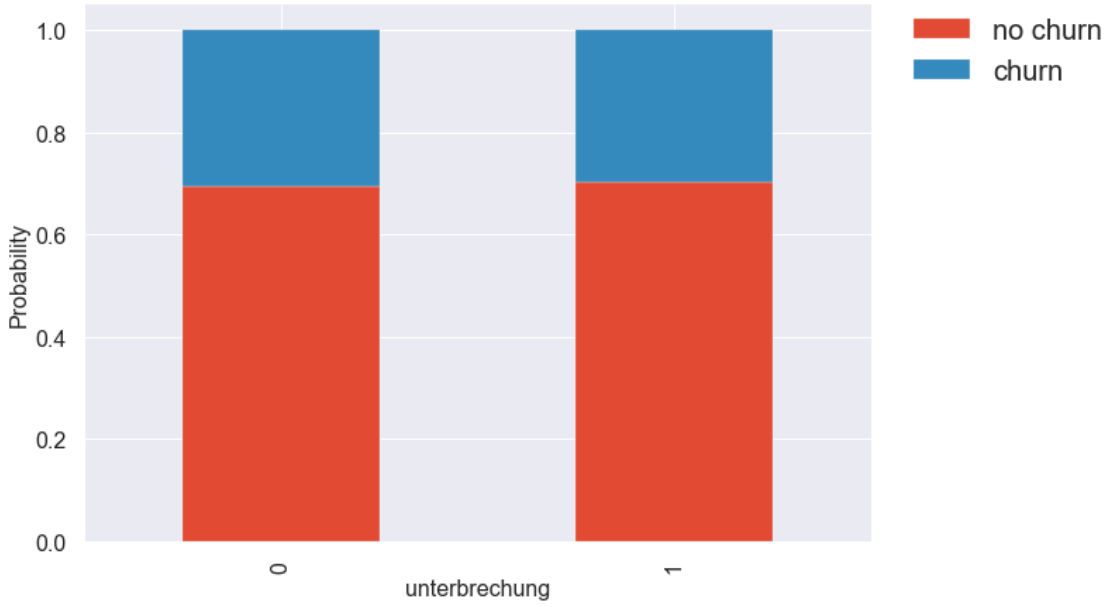
Here we can see where the student churn rate is the highest and where it is the lowest.

5.3.9 unterbrechung

[130]:
```
df.unterbrechung.value_counts().plot(kind='bar',figsize=(11,6));
plt.xlabel('unterbrechung');
```



[131]:
```
unterbrechung_churn = crosstab_evaluation(df.unterbrechung,df.churn)
crosstab_barplot(unterbrechung_churn,['no␣
↪churn','churn'],xlabelname='unterbrechung')
```



The churn rate for interruptions of the subscription seems to be the same.

5.3.10 Summary Subscription Data

To sum up, the following was found about the subscription related features:

- **kanal:** An important feature for churn (20 to almost 40 percent churn rate for different channels: keep feature

- **objekt_name:** Combined subscription have much less churn than digital and print subscriptions: keep feature

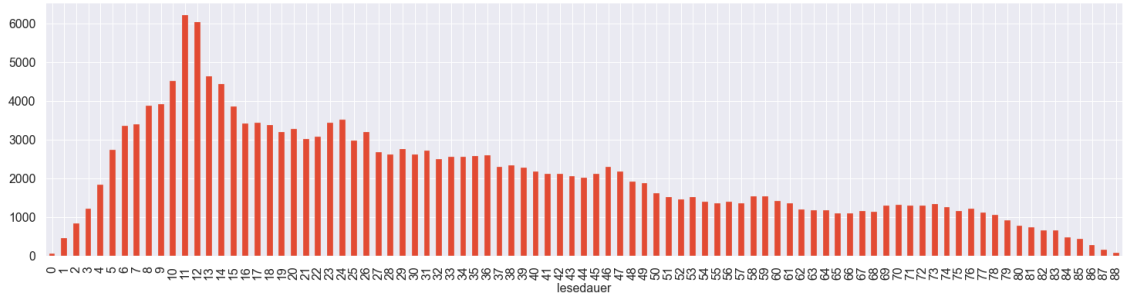
- **aboform_name:** Churn rate is dependent on the form of subscription: keep feature
- **zahlung_rhythmus_name:** Dependence of the payment period on the churn rate: keep feature
- **rechnungsmonat:** If there is a billing months, a subscriber has a higher tendency to churn: keep feature
- **zahlung_weg_name:** Direct debit (Bankeinzug) churn rate is much lower than if the payment is made by invoice: keep feature
- **studentenabo:** Students tend to churn more frequently: keep feature
- **unterbrechung:** The churn rate for interruptions of the subscription seems to be the same: could be kept!

5.4 Time/Temporal Features

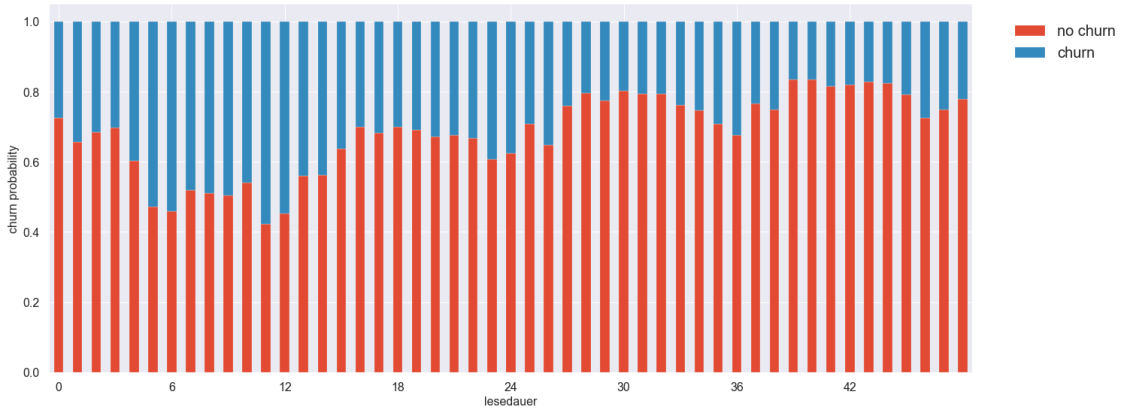
The following features are related to time subscription kind and shortly described: - lesedauer - liefer_beginn_evt - abo_registrierung_min - nl_registrierung_min - date_x - kuendigungs_eingangs_datum

5.4.1 lesedauer

[132]:
```
df.lesedauer.value_counts().sort_index().plot(kind='bar',figsize=(25,6));
#df.lesedauer.value_counts().sort_values('lesedauer').
 ↪plot(kind='bar',figsize=(25,6));
plt.xlabel('lesedauer');
```

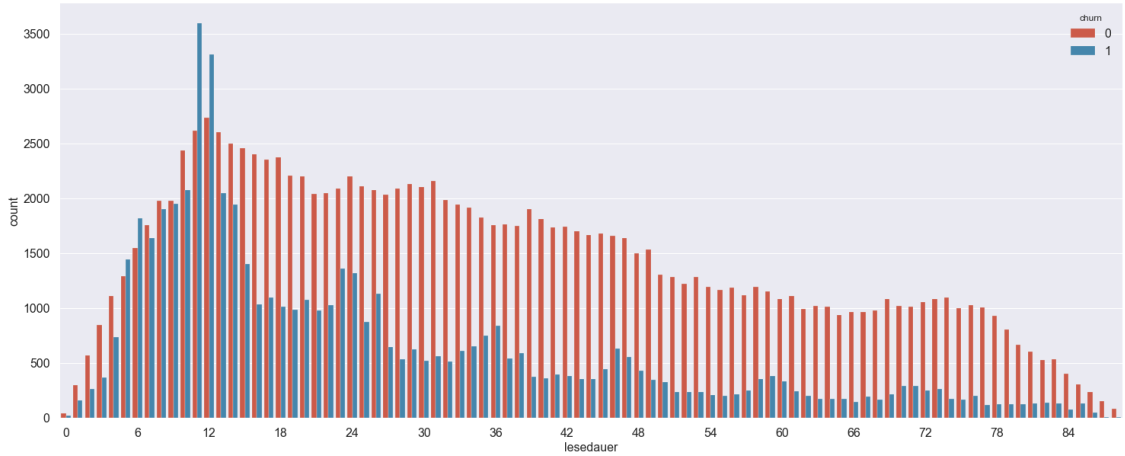


[174]:
```
lesedauer_churn = crosstab_evaluation(df.lesedauer,df.churn)
crosstab_barplot(lesedauer_churn,['no␣
↪churn','churn'],xlabelname='lesedauer',figsize_x=22,figsize_y=9)
plt.ylabel('churn probability');
plt.xticks(np.arange(0, 48, 6.0),rotation=0);
plt.xlim(-0.5,48.5)
plt.savefig('plots/lesedauer_churn.png',dpi=300,bbox_inches='tight')
```



The lesedauer is an important measure for the churn probability. With fewer months of lesedauer, the churn rate decreases, and then after 12 months there is a wave pattern with a period of 12 months (24, 36, 38, 60 months and so on) in which the churn rate increases.

[175]:
```
fig, ax = plt.subplots(figsize=(25,10))
ax = sns.countplot(x='lesedauer',data=df,hue='churn')
ax.set_xlabel('');
plt.xticks(np.arange(0, 88, 6.0));
plt.xlabel('lesedauer');
plt.savefig('plots/lesedauer_absolute_churn.png',dpi=300,bbox_inches='tight')
```



5.4.2 liefer_beginn_evt

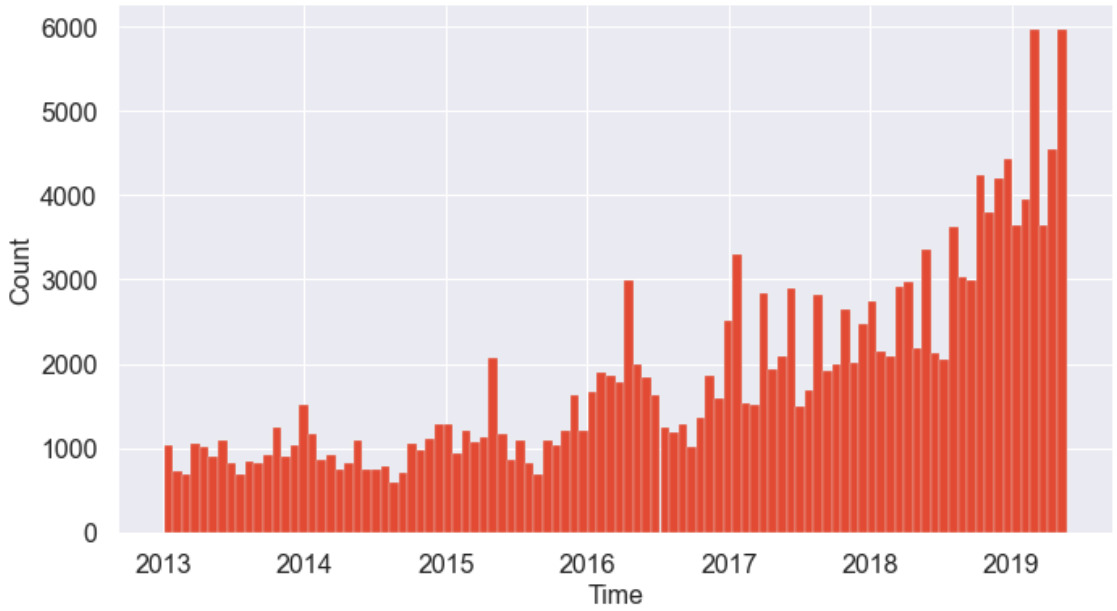
[73]:
```
df.liefer_beginn_evt.isna().sum()
```

[73]:
```
0
```

[136]:
```
df['liefer_beginn_evt'] = pd.to_datetime(df['liefer_beginn_evt'])
#df['liefer_beginn_evt'] = df['liefer_beginn_evt'].dt.date
df.liefer_beginn_evt.describe()
```

[136]:
```
count      184660
unique     406
top        2019-01-10 00:00:00
freq       1389
first      2013-01-03 00:00:00
last       2019-05-23 00:00:00
Name: liefer_beginn_evt, dtype: object
```

[176]:
```
df.liefer_beginn_evt.hist(bins=100,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
plt.savefig('plots/liefer_beginn_churn.png',dpi=300,bbox_inches='tight')
```



This feature can be kept, since all users have this feature included.

5.4.3 abo_registrierung_min

[138]:
```
df.abo_registrierung_min.isna().sum()
```

[138]:
```
0
```

[139]:
```
# convert abo registrierung to datetime
df['abo_registrierung_min'] = pd.to_datetime(df['abo_registrierung_min'])
df.abo_registrierung_min.describe()
```

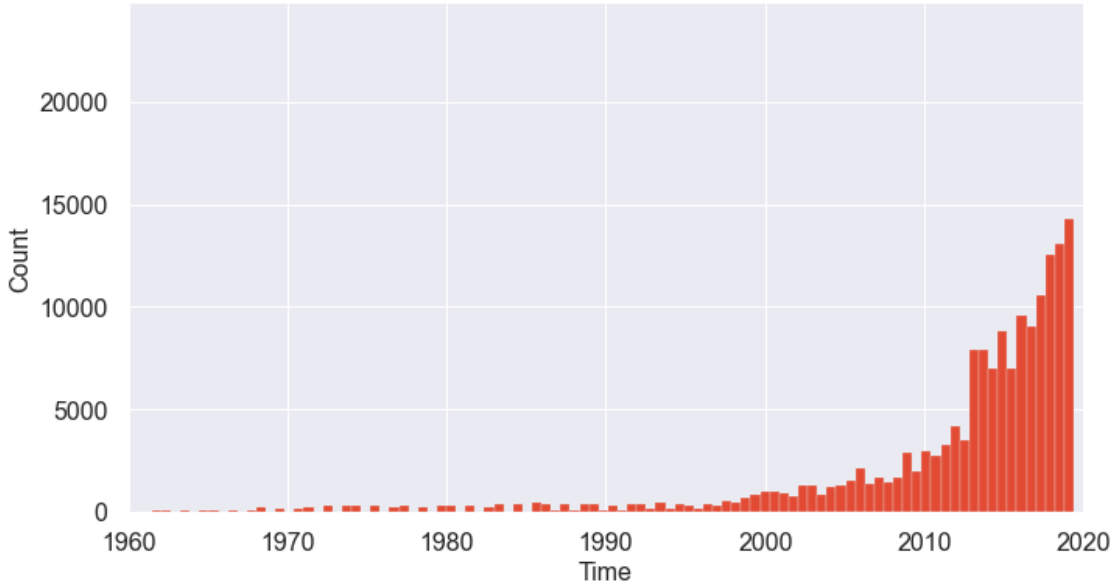
[139]:
```
count      184660
unique     109890
top        1900-01-01 00:00:00
freq       23630
first      1900-01-01 00:00:00
last       2019-05-21 03:44:07
Name: abo_registrierung_min, dtype: object
```

[140]:
```
df.abo_registrierung_min.hist(bins=200,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
```

Due to the huge amount of unknown time stamps, which are set to 1900-01-01, this feature can not be used for the model. Therefore it should be dropped.

[141]:
```
df.abo_registrierung_min.hist(bins=200,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
```

```
plt.xlim('1960-01-01','2020-01-01');
```



5.4.4 nl_registrierung_min

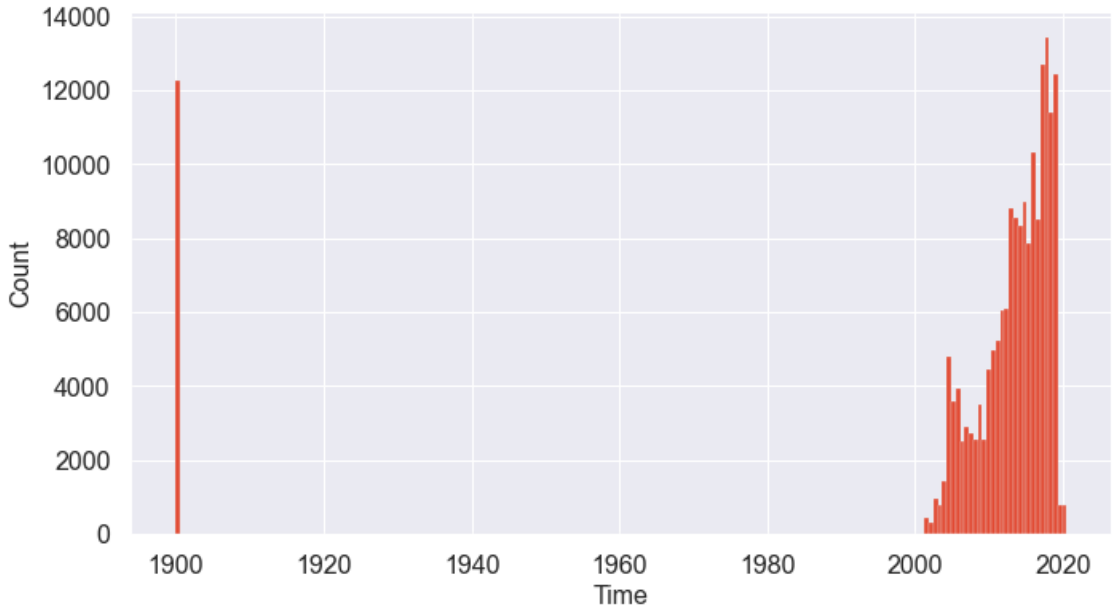
[142]:
```
df.nl_registrierung_min.isna().sum()
```

[142]:
```
0
```

[143]:
```
# convert nl_registrierung to datetime
df['nl_registrierung_min'] = pd.to_datetime(df['nl_registrierung_min'],␣
 ↪format='%Y-%m-%d')
#df['nl_registrierung_min'] = df['nl_registrierung_min'].dt.date
df.nl_registrierung_min.describe()
```

[143]:
```
count      184660
unique     111660
top        1900-01-01 00:00:00
freq       12243
first      1900-01-01 00:00:00
last       2020-07-21 08:30:49
Name: nl_registrierung_min, dtype: object
```

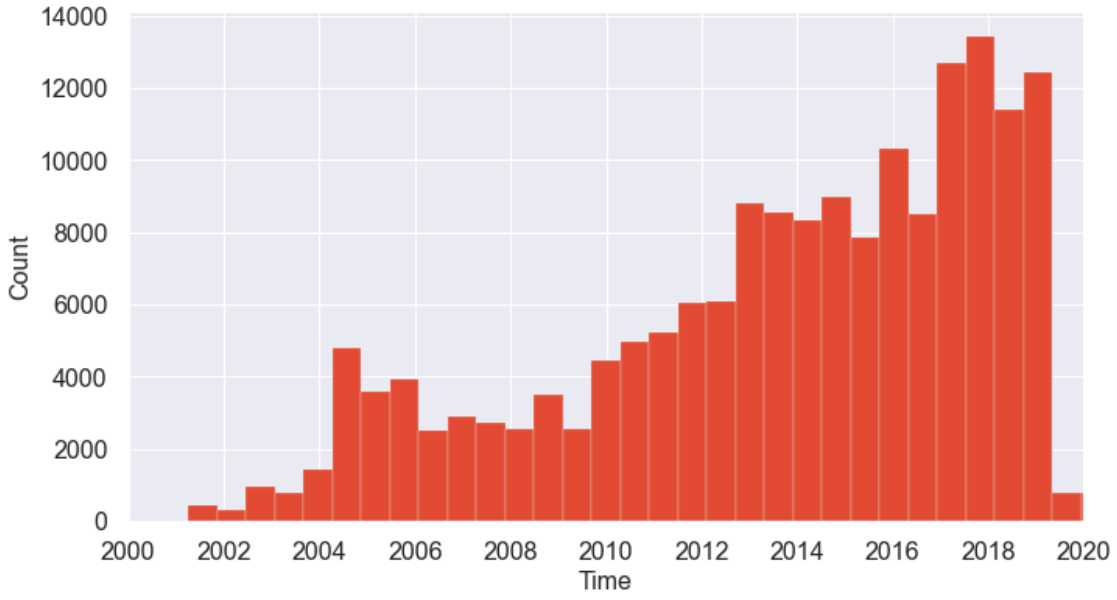
[145]:
```
df.nl_registrierung_min.hist(bins=200,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
```



We can observe a large number of unknown data which is set to 1900-01-01. This feature can therefore not be used in the model.

[146]:
```
df.nl_registrierung_min.hist(bins=200,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
plt.xlim('2000-01-01','2020-01-01')
```

[146]: (10957.0, 18262.0)



5.4.5 date_x

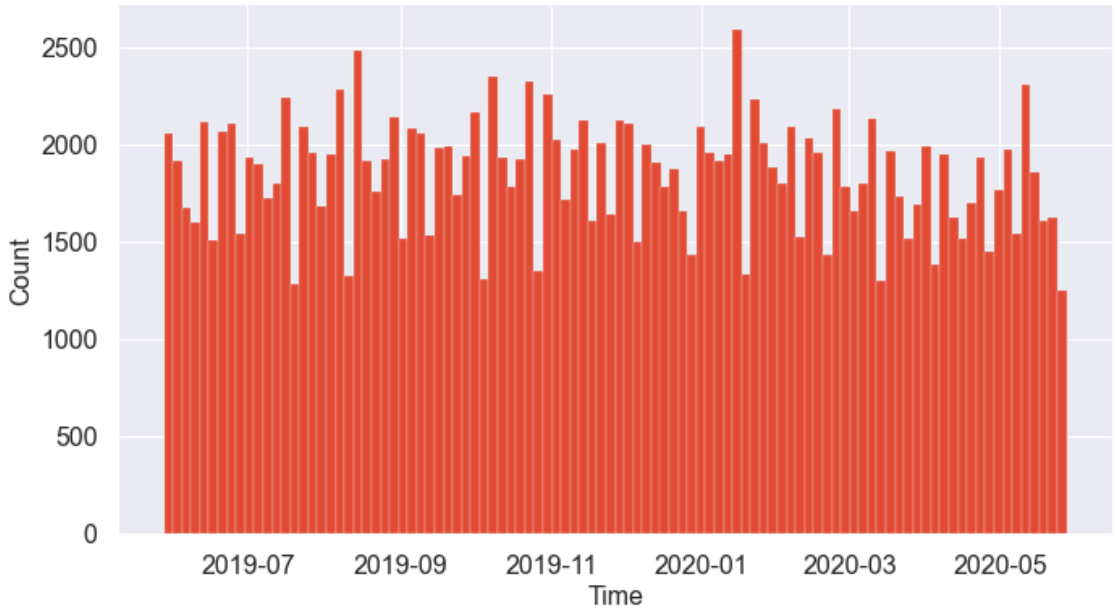
[89]:
```
df.date_x.isna().sum()
```

[89]:
```
0
```

[157]:
```
# convert date_x to datetime
df['date_x'] = pd.to_datetime(df['date_x'], format='%Y-%m-%d')
df.date_x.describe()
```

[157]:
```
count      184660
unique     367
top        2020-01-29 00:00:00
freq       747
first      2019-05-28 00:00:00
last       2020-05-28 00:00:00
Name: date_x, dtype: object
```

[158]:
```
df.date_x.hist(bins=100,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
```



5.4.6 kuendigungs_eingangs_datum

[150]:
```
df.kuendigungs_eingangs_datum.isna().sum()
```

[150]:
```
128680
```

[151]:
```
df.kuendigungs_eingangs_datum.fillna(value='1900-01-01 00:00:00',inplace=True);
```

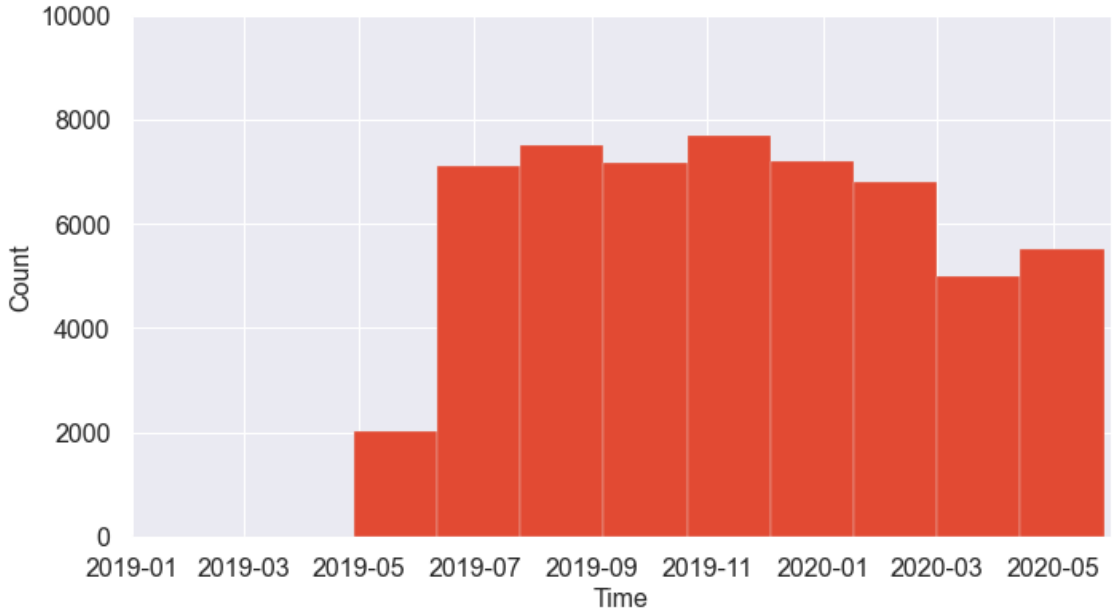
[152]:
```
# convert date_x to datetime
df['kuendigungs_eingangs_datum'] = pd.
 ↪to_datetime(df['kuendigungs_eingangs_datum'],errors='coerce',format='%Y-%m-%d')
df.kuendigungs_eingangs_datum.describe()
```

[152]:
```
count      184660
unique     350
top        1900-01-01 00:00:00
freq       128680
first      1900-01-01 00:00:00
last       2020-05-28 00:00:00
Name: kuendigungs_eingangs_datum, dtype: object
```

[153]:
```
df.kuendigungs_eingangs_datum.hist(bins=200,figsize=(11,6));
plt.xlabel('Time');
plt.ylabel('Count');
```

[177]:
```
df.kuendigungs_eingangs_datum.hist(bins=1000,figsize=(11,6));
plt.xlabel('Time');
```

```
plt.ylabel('Count');
plt.xlim('2019-01-01','2020-06-01');
plt.ylim(0,10000);
plt.savefig('plots/kuendigungseingang_absolute_churn.
 ↪png',dpi=300,bbox_inches='tight')
```



Kündigungen 2020

[159]:
```
kuendigungen_2020 = df[df['kuendigungs_eingangs_datum'] > '2020-01-01']
kuendigungen_2020.shape
```

[159]: (20644, 169)

[160]:
```
df['kuendigungs_eingangs_datum'][0]
```

[160]: Timestamp('1900-01-01 00:00:00')

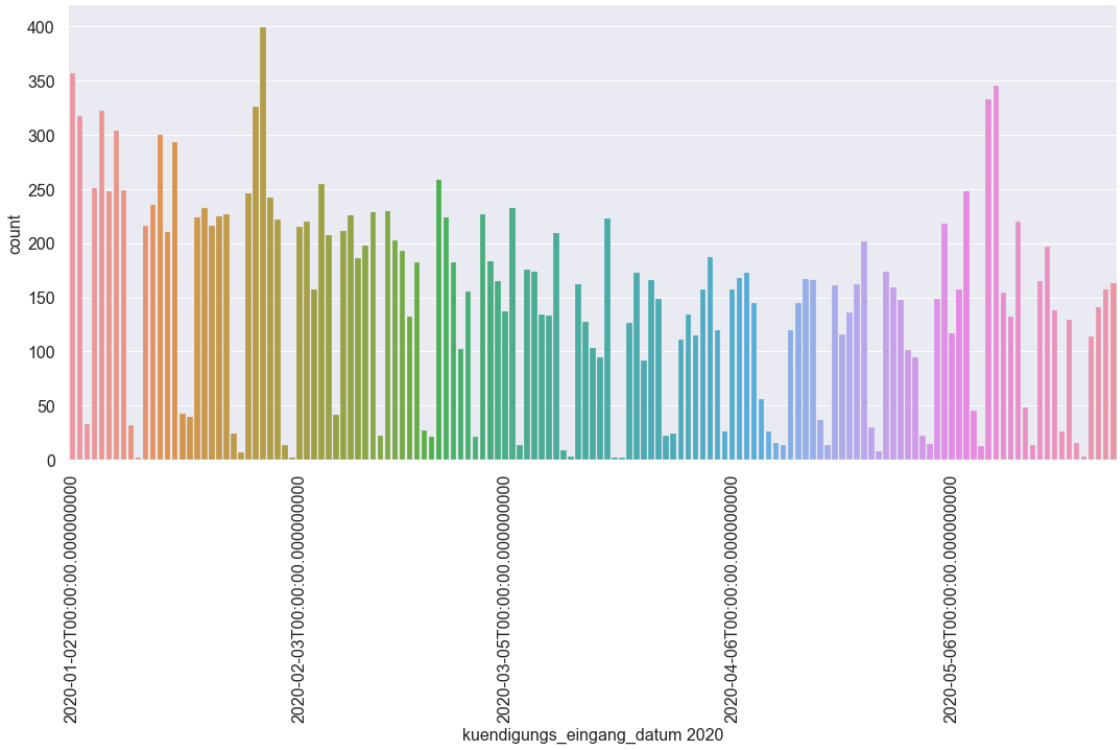
[161]:
```
kuendigungen_2020.kuendigungs_eingangs_datum.max()
```

[161]: Timestamp('2020-05-28 00:00:00')

[162]:
```
kuendigungen_2020 = kuendigungen_2020.
 ↪sort_values(by="kuendigungs_eingangs_datum",ascending=True)
```

[163]:
```
import matplotlib.dates as mdates
from datetime import datetime
from matplotlib.dates import DateFormatter, MonthLocator, YearLocator
```

```
years = mdates.YearLocator()   # every year
months = mdates.MonthLocator()  # every month
years_fmt = mdates.DateFormatter('%Y')

fig, ax = plt.subplots(figsize=(18,8))

sns.countplot(kuendigungen_2020['kuendigungs_eingangs_datum']);
plt.xticks(rotation=90);
plt.xlabel('kuendigungs_eingang_datum 2020')

months = MonthLocator()
monthsFmt = DateFormatter("%b")

years = YearLocator()
yearsFmt = DateFormatter("%y")

ax.xaxis.set_major_locator(mdates.MonthLocator(interval=1))

#ax.xaxis.set_major_locator(months)
#ax.xaxis.set_major_formatter(monthsFmt)
```



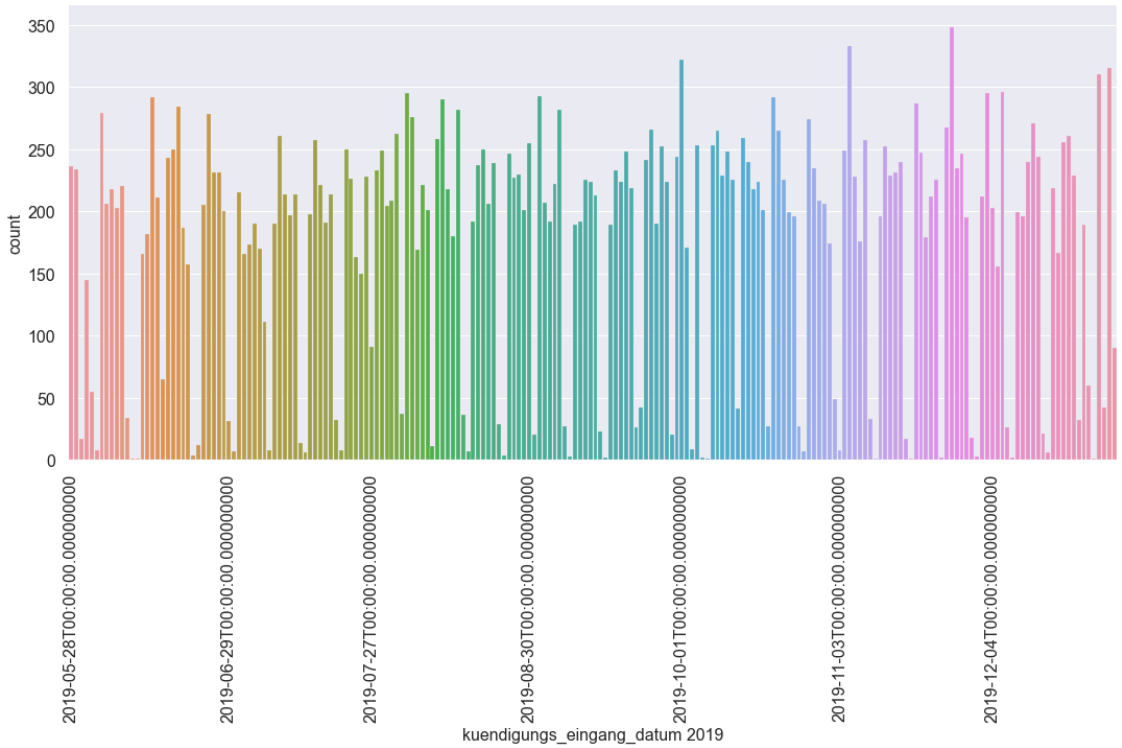
Kündigungen 2019

[164]:
```
kuendigungen_2019 = df[(df['kuendigungs_eingangs_datum'] < '2020-01-01') &␣
↪(df['kuendigungs_eingangs_datum'] > '2019-01-01')]
kuendigungen_2019.shape
```

[164]:
```
(35336, 169)
```

[165]:
```
kuendigungen_2019 = kuendigungen_2019.
↪sort_values(by="kuendigungs_eingangs_datum",ascending=True)
```

[166]:
```
import matplotlib.dates as mdates
from datetime import datetime
from matplotlib.dates import DateFormatter, MonthLocator, YearLocator

years = mdates.YearLocator()   # every year
months = mdates.MonthLocator()  # every month
years_fmt = mdates.DateFormatter('%Y')

fig, ax = plt.subplots(figsize=(18,8))

ax = sns.countplot(kuendigungen_2019['kuendigungs_eingangs_datum']);
plt.xticks(rotation=90);
plt.xlabel('kuendigungs_eingang_datum 2019')

months = MonthLocator()
monthsFmt = DateFormatter("%m-%d")

years = YearLocator()
yearsFmt = DateFormatter("%y")

#ax.xaxis.set_major_formatter(monthsFmt)
#ax.xaxis.set_major_locator(months)

date_form = DateFormatter("%m-%d")
#ax.xaxis.set_major_formatter(date_form)

# Ensure a major tick for each week using (interval=1)
ax.xaxis.set_major_locator(mdates.MonthLocator(interval=1))

#ax.set_xlim(pd.datetime.strptime(startDate,'%Y-%m-%d'),pd.datetime.
↪strptime(stopDate,'%Y-%m-%d'))
```

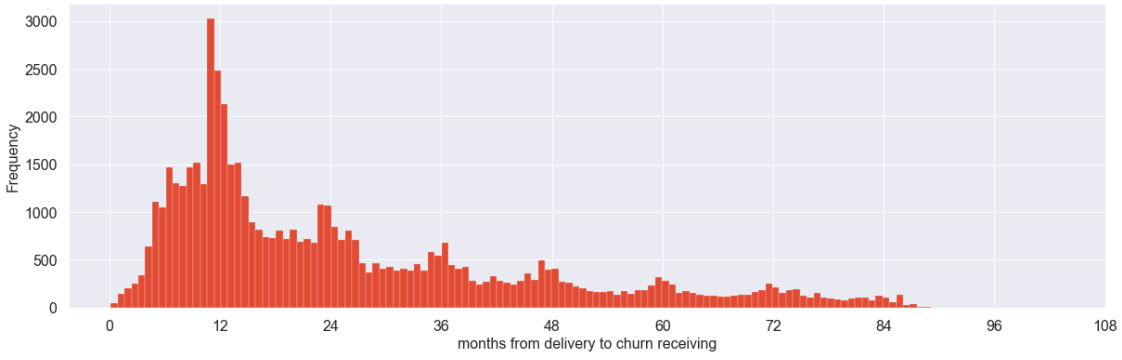


5.4.7 Time relation between kuendigungs_eingangs_datum und lieferstart

[179]:
```
df_churn = df.query('churn == 1')
```

[180]:
```
df_churn['delivery_to_churn'] = df_churn['kuendigungs_eingangs_datum'] -␣
↪df_churn['liefer_beginn_evt']
```

[181]:
```
df_churn['delivery_to_churn'].astype('timedelta64[s]').div(2592000).plot.
↪hist(bins=120,figsize=(20,6));
plt.xlabel('months from delivery to churn receiving');
plt.xticks(np.arange(0,120,12));
plt.savefig('plots/time_delivery_to_churn.png',dpi=300,bbox_inches='tight')
```

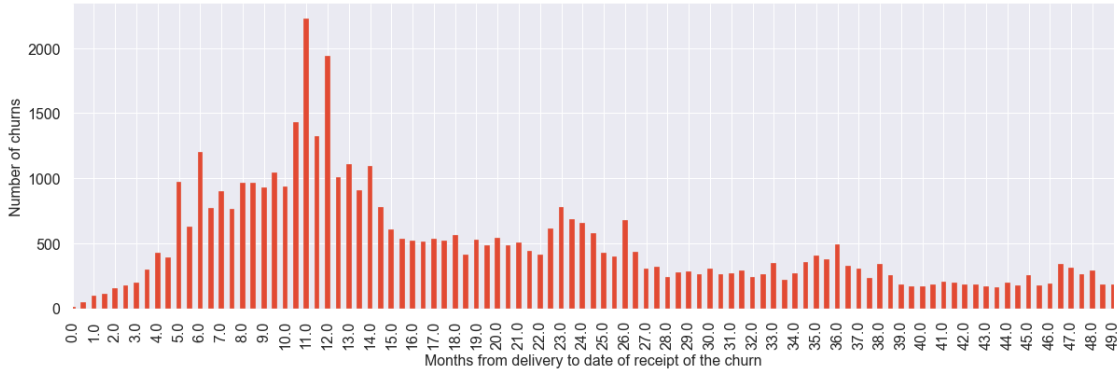


[182]:
```
df_churn['delivery_to_churn'] = df_churn['delivery_to_churn'].
 ↪astype('timedelta64[s]').div(2592000)
```

[183]:
```
months_bins = np.arange(0, 90.5, 0.5)
```

[184]:
```
months_bin_series = pd.cut(df_churn['delivery_to_churn'], bins=months_bins,␣
 ↪labels=months_bins[:-1])
months_bin_series.name = 'months_bins_2m'
df_churn['months_bins_2m'] = months_bin_series
```

[186]:
```
import matplotlib.ticker as ticker
fig, ax = plt.subplots(figsize=(11,16))
df_churn.groupby('months_bins_2m').count()['churn'].
 ↪plot(kind='bar',figsize=(20,6))
x_ticks = np.arange(0, 120, 2);
plt.xticks(x_ticks);
plt.xlim(0,98);
plt.ylabel('Number of churns');
plt.xlabel('Months from delivery to date of receipt of the churn');
plt.savefig('plots/time_delivery_to_churn_half_months.
 ↪png',dpi=300,bbox_inches='tight')
```



5.4.8 Time Relation between kuendigungs_eingangs_datum and nl_registrierung_min

[187]:
```
df_churn['nl_start_to_churn'] = df_churn['kuendigungs_eingangs_datum'] -␣
 ↪df_churn['nl_registrierung_min']
```

[188]:
```
df_churn['nl_start_to_churn'].astype('timedelta64[s]').div(2592000).plot.
 ↪hist(bins=1200,figsize=(20,6));
```

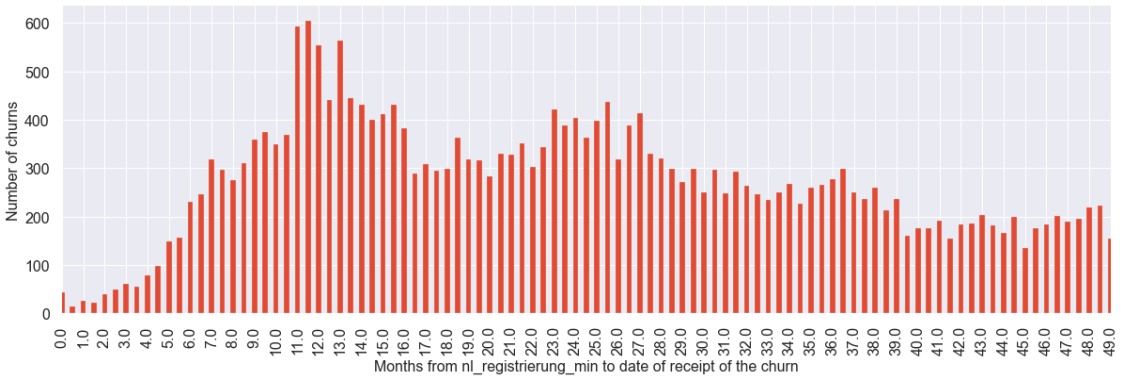
```
plt.xlabel('months from nl_registrierung min to churn receiving');
plt.xlim(-10,200);
```

[189]:
```
df_churn['nl_start_to_churn'] = df_churn['nl_start_to_churn'].
↪astype('timedelta64[s]').div(2592000)
```

[190]:
```
months_bins = np.arange(0, 90.5, 0.5)
```

[191]:
```
nl_min_bins_series = pd.cut(df_churn['nl_start_to_churn'], bins=months_bins,␣
↪labels=months_bins[:-1])
nl_min_bins_series.name = 'nl_min_bins_05m'
df_churn['nl_start_to_churn_bins_05m'] = nl_min_bins_series
```

[192]:
```
fig, ax = plt.subplots(figsize=(11,16))
df_churn.groupby('nl_start_to_churn_bins_05m').count()['churn'].
↪plot(kind='bar',figsize=(20,6))
x_ticks = np.arange(0, 120, 2);
plt.xticks(x_ticks);
plt.xlim(0,98);
plt.ylabel('Number of churns');
plt.xlabel('Months from nl_registrierung_min to date of receipt of the churn');
```



5.4.9 Summary Time/Temporal Features

To sum up the EDA on the temporal features, the following was found:

- **lesedauer:** The lesedauer is an important measure for the churn probability: keep feature
- **liefer_beginn_evt:** This feature can be kept, since all users have this feature included: keep feature
- **abo_registrierung_min:** Due to the huge amount of unknown time stamps, which are set to 1900-01-01, this feature can not be used: drop feature
- **nl_registrierung_min:** Large number of unknown data: drop feature
- **date_x:** Is a random date or the kuendigungs_eingangs_datum, since we have to drop the kuendigungs_eingangs_datum, since it is directly related to the churn.
- **kuendigungs_eingangs_datum:** must be dropped, since it is directly related to the churn date.

5.5 Activity features

These features were we can see interactions and communicate with the customers.

- newsletter
- clickrates
- openrates
- community

[193]:
```
index=0
for elem in df.columns:
    print(f"{index} {elem}")
    index+=1
```

```
0 liefer_beginn_evt
1 kanal
2 objekt_name
3 aboform_name
4 zahlung_rhythmus_name
5 lesedauer
6 rechnungsmonat
7 zahlung_weg_name
8 studentenabo
9 plz_1
10 plz_2
11 plz_3
12 ort
13 metropole
14 land_iso_code
15 shop_kauf
16 unterbrechung
17 anrede
18 titel
```

```
19 avg_churn
20 email_am_kunden
21 zon_che_opt_in
22 zon_sit_opt_in
23 zon_zp_grey
24 zon_premium
25 zon_boa
26 zon_kommentar
27 zon_sonstige
28 zon_zp_red
29 zon_rawr
30 zon_community
31 zon_app_sonstige
32 zon_schach
33 zon_blog_kommentare
34 zon_quiz
35 cnt_abo
36 cnt_abo_diezeit
37 cnt_abo_diezeit_digital
38 cnt_abo_magazin
39 cnt_umwandlungsstatus2_dkey
40 abo_registrierung_min
41 nl_zeitbrief
42 nl_zeitshop
43 nl_zeitverlag_hamburg
44 nl_fdz_organisch
45 nl_blacklist_sum
46 nl_bounced_sum
47 nl_aktivitaet
48 nl_registrierung_min
49 nl_sperrliste_sum
50 nl_opt_in_sum
51 boa_reg
52 che_reg
53 sit_reg
54 sso_reg
55 received_anzahl_1w
56 received_anzahl_1m
57 received_anzahl_3m
58 received_anzahl_6m
59 opened_anzahl_1w
60 opened_anzahl_1m
61 opened_anzahl_3m
62 openedanzahl_6m
63 clicked_anzahl_1w
64 clicked_anzahl_1m
65 clicked_anzahl_3m
66 clicked_anzahl_6m
```

```
67 unsubscribed_anzahl_1w
68 unsubscribed_anzahl_1m
69 unsubscribed_anzahl_3m
70 unsubscribed_anzahl_6m
71 openrate_1w
72 clickrate_1w
73 openrate_1m
74 clickrate_1m
75 openrate_3m
76 clickrate_3m
77 received_anzahl_bestandskunden_1w
78 received_anzahl_bestandskunden_1m
79 received_anzahl_bestandskunden_3m
80 received_anzahl_bestandskunden_6m
81 opened_anzahl_bestandskunden_1w
82 opened_anzahl_bestandskunden_1m
83 opened_anzahl_bestandskunden_3m
84 openedanzahl_bestandskunden_6m
85 clicked_anzahl_bestandskunden_1w
86 clicked_anzahl_bestandskunden_1m
87 clicked_anzahl_bestandskunden_3m
88 clicked_anzahl_bestandskunden_6m
89 unsubscribed_anzahl_bestandskunden_1w
90 unsubscribed_anzahl_bestandskunden_1m
91 unsubscribed_anzahl_bestandskunden_3m
92 unsubscribed_anzahl_bestandskunden_6m
93 openrate_bestandskunden_1w
94 clickrate_bestandskunden_1w
95 openrate_bestandskunden_1m
96 clickrate_bestandskunden_1m
97 openrate_bestandskunden_3m
98 clickrate_bestandskunden_3m
99 received_anzahl_produktnews_1w
100 received_anzahl_produktnews_1m
101 received_anzahl_produktnews_3m
102 received_anzahl_produktnews_6m
103 opened_anzahl_produktnews_1w
104 opened_anzahl_produktnews_1m
105 opened_anzahl_produktnews_3m
106 openedanzahl_produktnews_6m
107 clicked_anzahl_produktnews_1w
108 clicked_anzahl_produktnews_1m
109 clicked_anzahl_produktnews_3m
110 clicked_anzahl_produktnews_6m
111 unsubscribed_anzahl_produktnews_1w
112 unsubscribed_anzahl_produktnews_1m
113 unsubscribed_anzahl_produktnews_3m
114 unsubscribed_anzahl_produktnews_6m
```

```
115 openrate_produktnews_1w
116 clickrate_produktnews_1w
117 openrate_produktnews_1m
118 clickrate_produktnews_1m
119 openrate_produktnews_3m
120 clickrate_produktnews_3m
121 received_anzahl_hamburg_1w
122 received_anzahl_hamburg_1m
123 received_anzahl_hamburg_3m
124 received_anzahl_hamburg_6m
125 opened_anzahl_hamburg_1w
126 opened_anzahl_hamburg_1m
127 opened_anzahl_hamburg_3m
128 openedanzahl_hamburg_6m
129 clicked_anzahl_hamburg_1w
130 clicked_anzahl_hamburg_1m
131 clicked_anzahl_hamburg_3m
132 clicked_anzahl_hamburg_6m
133 unsubscribed_anzahl_hamburg_1w
134 unsubscribed_anzahl_hamburg_1m
135 unsubscribed_anzahl_hamburg_3m
136 unsubscribed_anzahl_hamburg_6m
137 openrate_hamburg_1w
138 clickrate_hamburg_1w
139 openrate_hamburg_1m
140 clickrate_hamburg_1m
141 openrate_hamburg_3m
142 clickrate_hamburg_3m
143 received_anzahl_zeitbrief_1w
144 received_anzahl_zeitbrief_1m
145 received_anzahl_zeitbrief_3m
146 received_anzahl_zeitbrief_6m
147 opened_anzahl_zeitbrief_1w
148 opened_anzahl_zeitbrief_1m
149 opened_anzahl_zeitbrief_3m
150 openedanzahl_zeitbrief_6m
151 clicked_anzahl_zeitbrief_1w
152 clicked_anzahl_zeitbrief_1m
153 clicked_anzahl_zeitbrief_3m
154 clicked_anzahl_zeitbrief_6m
155 unsubscribed_anzahl_zeitbrief_1w
156 unsubscribed_anzahl_zeitbrief_1m
157 unsubscribed_anzahl_zeitbrief_3m
158 unsubscribed_anzahl_zeitbrief_6m
159 openrate_zeitbrief_1w
160 clickrate_zeitbrief_1w
161 openrate_zeitbrief_1m
162 clickrate_zeitbrief_1m
```

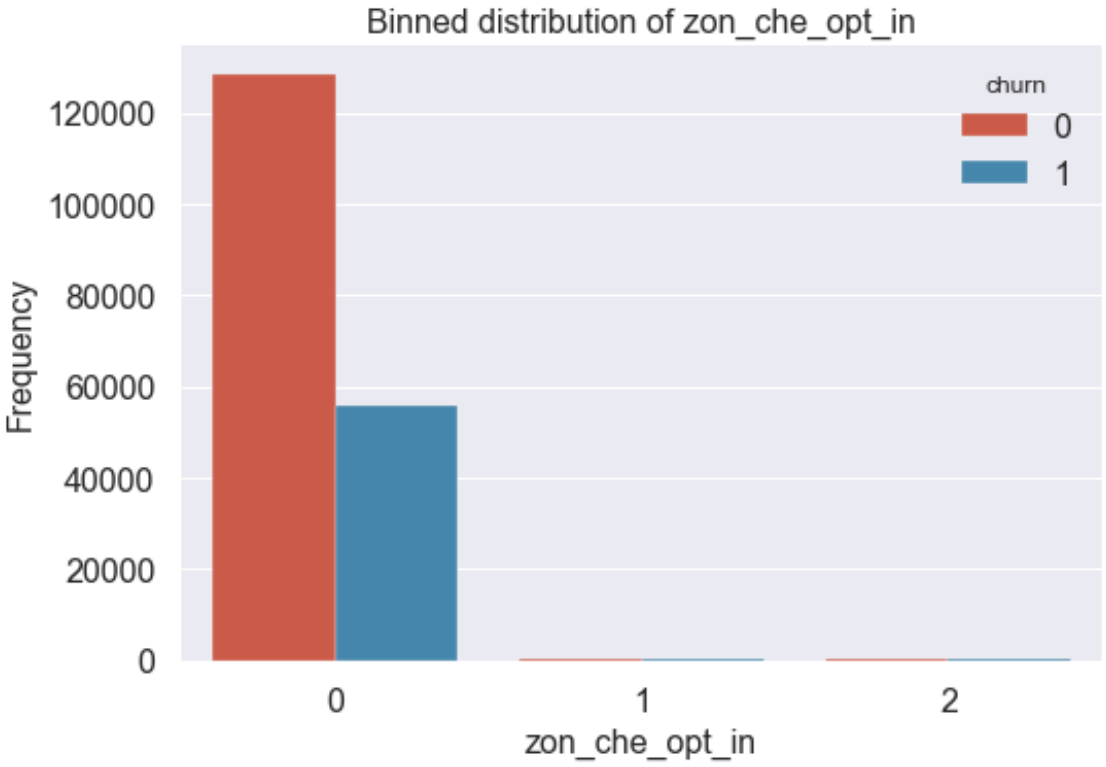
```
163 openrate_zeitbrief_3m
164 clickrate_zeitbrief_3m
165 training_set
166 kuendigungs_eingangs_datum
167 churn
168 date_x
```

Create dataframes for different activity features to plot and describe them one by one.

[194]:
```
df_zon = df.iloc[:, 21:35]                # zones are special areas that need␣
↪registration
df_cnt = df.iloc[:, 35:40]                # cnt is the number of subscribtions␣
↪the contract holds (families, libaries etc.)
df_nl = df.iloc[:, 41:51]                 # newsletter drop technical details
df_nl.drop(["nl_blacklist_sum", "nl_bounced_sum", "nl_sperrliste_sum",␣
↪"nl_opt_in_sum", "nl_fdz_organisch", "nl_registrierung_min"], axis=1,␣
↪inplace=True)
df_reg = df.iloc[:, 51:55]                # registration needed (for accses to␣
↪these services)
df_nl_interact = df.iloc[:, 55:77]        # newsletter interactions
df_nl_bestandskunden = df.iloc[:, 77:99]  # newsletter existing customers
df_nl_produktnews = df.iloc[:, 99:121]    # productnews (kind of newsletter␣
↪but more commercial)
df_nl_hamburg = df.iloc[:, 121:143]       # newsletter region hamburg
df_zb = df.iloc[:, 143:165]               # zb = zeitbrief kind of letter
```

5.5.1 zon features

[195]:
```
for elem in df_zon.columns[:2]:
    describe_frame = pd.DataFrame(round(df[elem].describe(),1))
    ax = sns.countplot(x=elem, data=df, hue="churn")
    ax.set(xlabel=elem, ylabel="Frequency", xlim=[-0.5, 2.5])
    plt.title(f"Binned distribution of {elem}")
    plt.show()
    print(describe_frame.T)
    print("\n")
```



	count	mean	std	min	25%	50%	75%	max
zon_che_opt_in	184660.0	0.0	0.1	0.0	0.0	0.0	0.0	2.0

	count	mean	std	min	25%	50%	75%	max
zon_sit_opt_in	184660.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0

[134]:
```
eda.correlogram(df_zon)
```

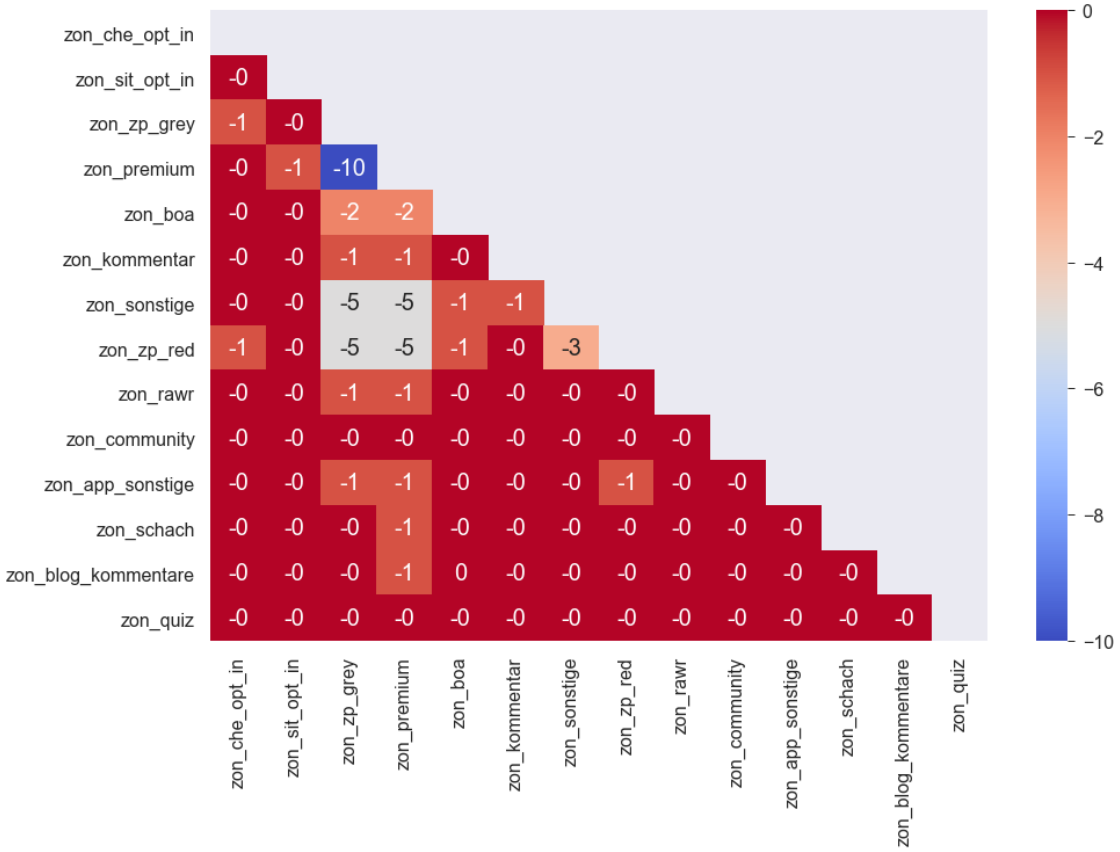
[134]:

	zon_che_opt_in	zon_sit_opt_in	zon_zp_grey	zon_premium
zon_che_opt_in	1.00	-0.00	-0.01	-0.00
zon_sit_opt_in	-0.00	1.00	-0.00	-0.01
zon_zp_grey	-0.01	-0.00	1.00	-0.10
zon_premium	-0.00	-0.01	-0.10	1.00
zon_boa	-0.00	-0.00	-0.02	-0.02
zon_kommentar	-0.00	-0.00	-0.01	-0.01
zon_sonstige	-0.00	-0.00	-0.05	-0.05
zon_zp_red	-0.01	-0.00	-0.05	-0.05
zon_rawr	-0.00	-0.00	-0.01	-0.01
zon_community	-0.00	-0.00	-0.00	-0.00
zon_app_sonstige	-0.00	-0.00	-0.01	-0.01
zon_schach	-0.00	-0.00	-0.00	-0.01
zon_blog_kommentare	-0.00	-0.00	-0.00	-0.01
zon_quiz	-0.00	-0.00	-0.00	-0.00

	zon_boa	zon_kommentar	zon_sonstige	zon_zp_red	\
zon_che_opt_in	-0.00	-0.00	-0.00	-0.01	
zon_sit_opt_in	-0.00	-0.00	-0.00	-0.00	
zon_zp_grey	-0.02	-0.01	-0.05	-0.05	
zon_premium	-0.02	-0.01	-0.05	-0.05	
zon_boa	1.00	-0.00	-0.01	-0.01	
zon_kommentar	-0.00	1.00	-0.01	-0.00	
zon_sonstige	-0.01	-0.01	1.00	-0.03	
zon_zp_red	-0.01	-0.00	-0.03	1.00	
zon_rawr	-0.00	-0.00	-0.00	-0.00	
zon_community	-0.00	-0.00	-0.00	-0.00	
zon_app_sonstige	-0.00	-0.00	-0.00	-0.01	
zon_schach	-0.00	-0.00	-0.00	-0.00	
zon_blog_kommentare	0.00	-0.00	-0.00	-0.00	
zon_quiz	-0.00	-0.00	-0.00	-0.00	

	zon_rawr	zon_community	zon_app_sonstige	zon_schach	\
zon_che_opt_in	-0.00	-0.0	-0.00	-0.00	
zon_sit_opt_in	-0.00	-0.0	-0.00	-0.00	
zon_zp_grey	-0.01	-0.0	-0.01	-0.00	
zon_premium	-0.01	-0.0	-0.01	-0.01	
zon_boa	-0.00	-0.0	-0.00	-0.00	
zon_kommentar	-0.00	-0.0	-0.00	-0.00	
zon_sonstige	-0.00	-0.0	-0.00	-0.00	
zon_zp_red	-0.00	-0.0	-0.01	-0.00	
zon_rawr	1.00	-0.0	-0.00	-0.00	
zon_community	-0.00	1.0	-0.00	-0.00	
zon_app_sonstige	-0.00	-0.0	1.00	-0.00	
zon_schach	-0.00	-0.0	-0.00	1.00	
zon_blog_kommentare	-0.00	-0.0	-0.00	-0.00	
zon_quiz	-0.00	-0.0	-0.00	-0.00	

	zon_blog_kommentare	zon_quiz
zon_che_opt_in	-0.00	-0.0
zon_sit_opt_in	-0.00	-0.0
zon_zp_grey	-0.00	-0.0
zon_premium	-0.01	-0.0
zon_boa	0.00	-0.0
zon_kommentar	-0.00	-0.0
zon_sonstige	-0.00	-0.0
zon_zp_red	-0.00	-0.0
zon_rawr	-0.00	-0.0
zon_community	-0.00	-0.0
zon_app_sonstige	-0.00	-0.0
zon_schach	-0.00	-0.0
zon_blog_kommentare	1.00	-0.0
zon_quiz	-0.00	1.0



Observation: All the features in the list have a range between 0 and 2. 0 is a code for no interactions, 1 describes "registration started", 2 is "registration completed". As you can see in the plots before these comments, the rates of customers to even register or completed register is very low. The correlogram(df_zon) showed no remarkable correlations (rounded values -1 or 0).

5.5.2 cnt features

[196]:
```
df_cnt.sample(5)
```

[196]:

	cnt_abo	cnt_abo_diezeit	cnt_abo_diezeit_digital	cnt_abo_magazin	cnt_umwandlungsstatus2_dkey
34987	0	0	0	0	0
115274	2	2	0	0	0
3584	0	0	0	0	0
175170	0	0	0	0	0
189104	1	1	0	0	

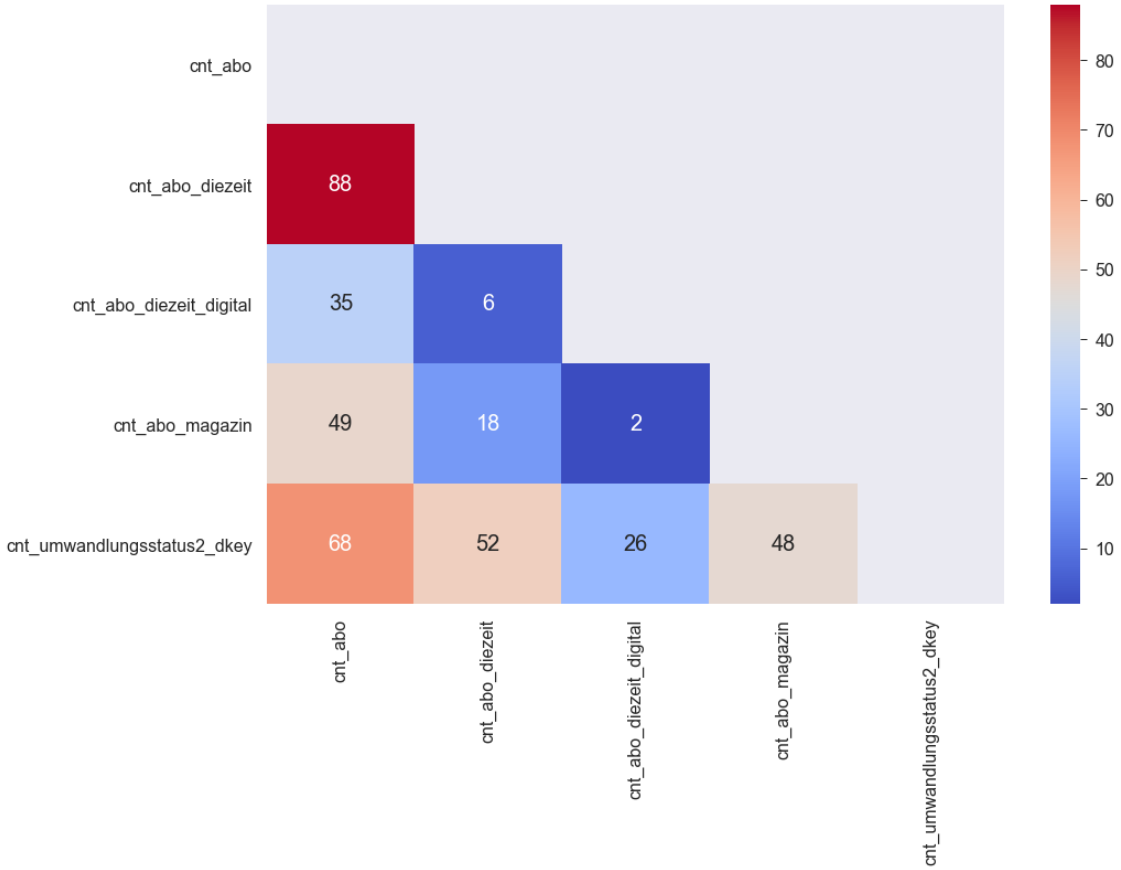
```
189104  1
```

[197]:
```
df_cnt.cnt_abo.value_counts()
```

[197]:
```
0    86216
1    45199
2    27536
3    16094
4    9615
Name: cnt_abo, dtype: int64
```

[198]:
```
eda.correlogram(df_cnt)
```

[198]:
```
                            cnt_abo  cnt_abo_diezeit  \
cnt_abo                     1.00     0.88
cnt_abo_diezeit             0.88     1.00
cnt_abo_diezeit_digital     0.35     0.06
cnt_abo_magazin             0.49     0.18
cnt_umwandlungsstatus2_dkey 0.68     0.52

                            cnt_abo_diezeit_digital  cnt_abo_magazin  \
cnt_abo                     0.35                     0.49
cnt_abo_diezeit             0.06                     0.18
cnt_abo_diezeit_digital     1.00                     0.02
cnt_abo_magazin             0.02                     1.00
cnt_umwandlungsstatus2_dkey 0.26                     0.48

                            cnt_umwandlungsstatus2_dkey
cnt_abo                     0.68
cnt_abo_diezeit             0.52
cnt_abo_diezeit_digital     0.26
cnt_abo_magazin             0.48
cnt_umwandlungsstatus2_dkey 1.00
```



Observation: The cnt features contain information of how many and which subscription types are held by the "auftrag_new_id". As we can see above, the column "cnt_abo" holds the sum of the other columns. The last column is holds information about how many of these are "conversion" by trial subscription. When we had a look at the correlogram(df_cnt) there were strong correlation. This is surprising cause the cnt_abo features are these that count the sum of abos a customers keeps. For example if one person has more than one newspaper, or a newspaper and a magazine, or newspaper print and digital. For more information about the subscription types check the wiki: Business understanding.

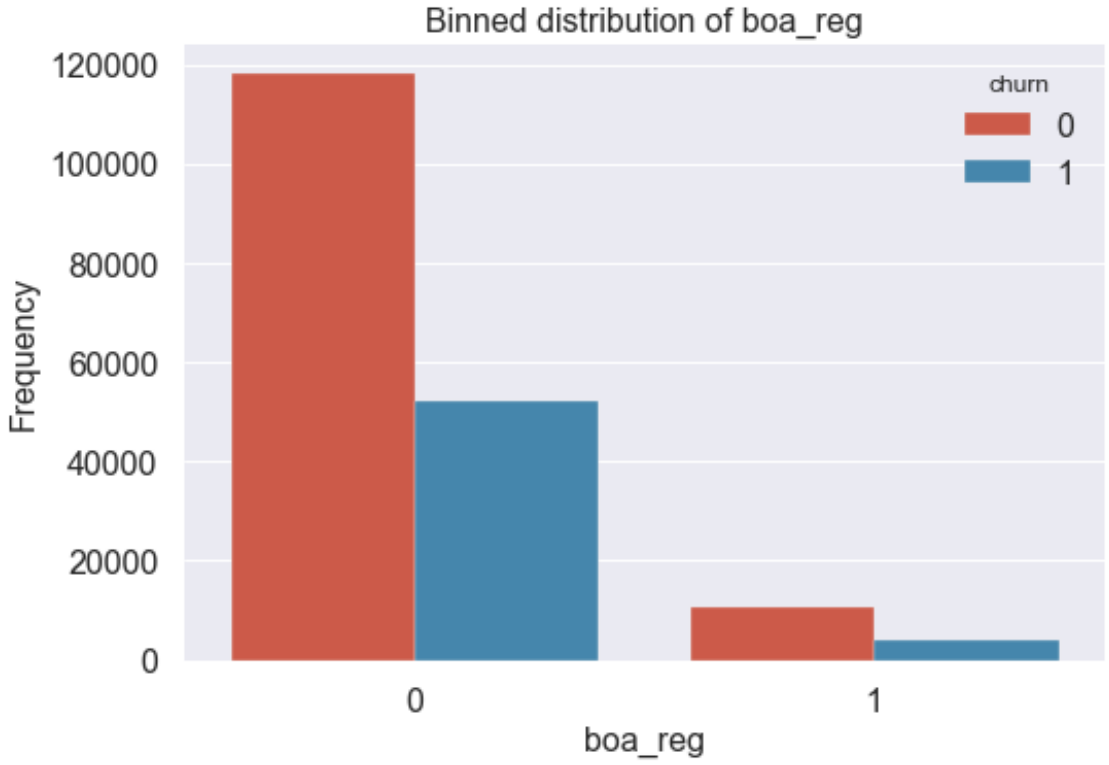
5.5.3 reg features

[199]:
```
df_reg.boa_reg.value_counts()
```

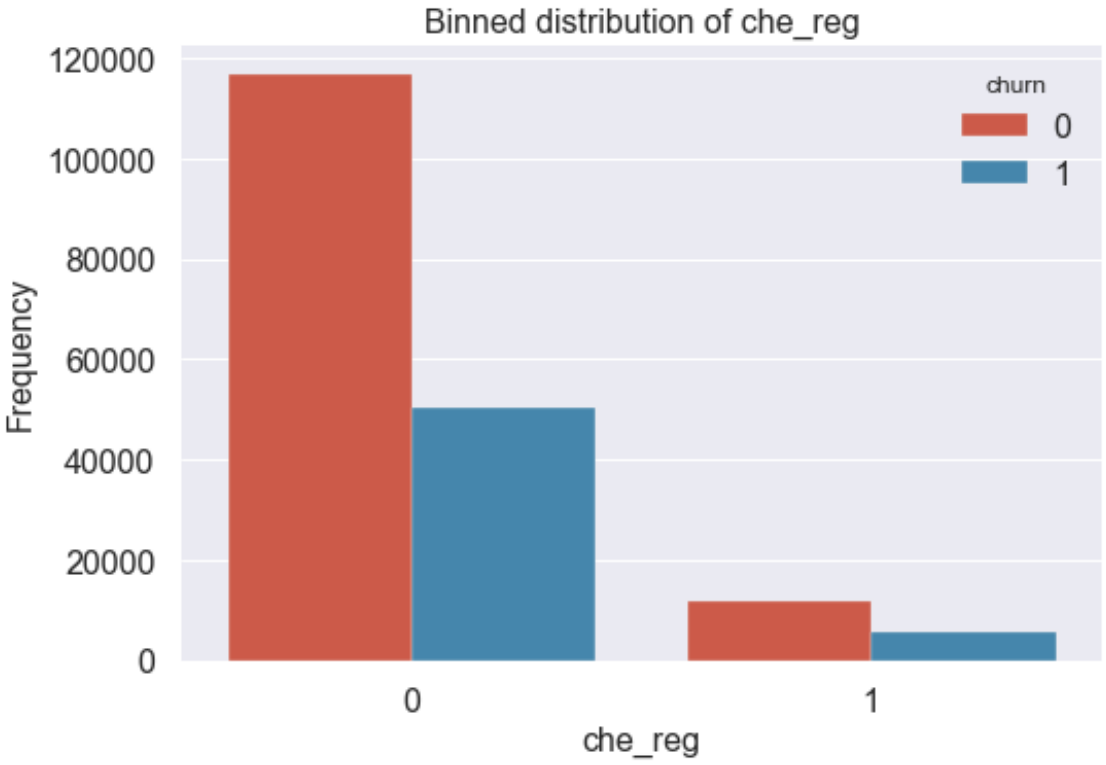
[199]:
```
0    170395
1     14265
Name: boa_reg, dtype: int64
```

[200]:
```
for elem in df_reg.columns:
    describe_frame = pd.DataFrame(round(df[elem].describe(),1))
    ax = sns.countplot(x=elem, data=df, hue="churn")
```

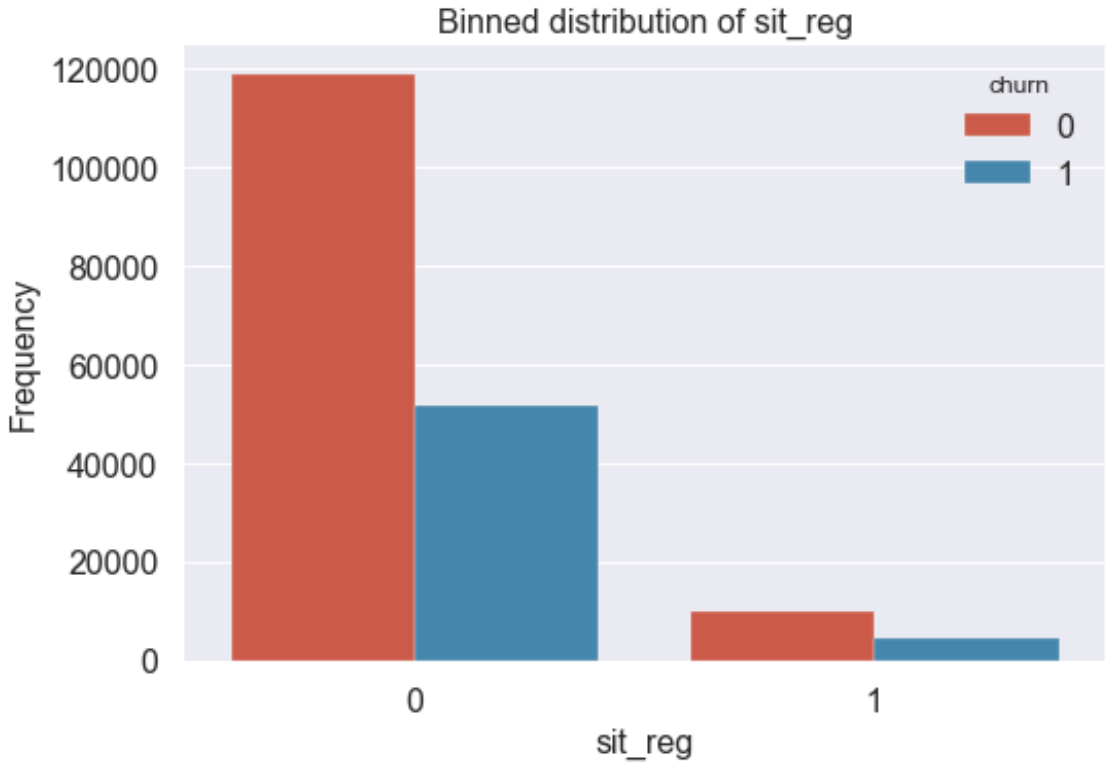
```
    ax.set(xlabel=elem, ylabel="Frequency", xlim=[-0.5, 1.5])
    plt.title(f"Binned distribution of {elem}")
    plt.show()
    print(describe_frame.T)
    print("\n")
```



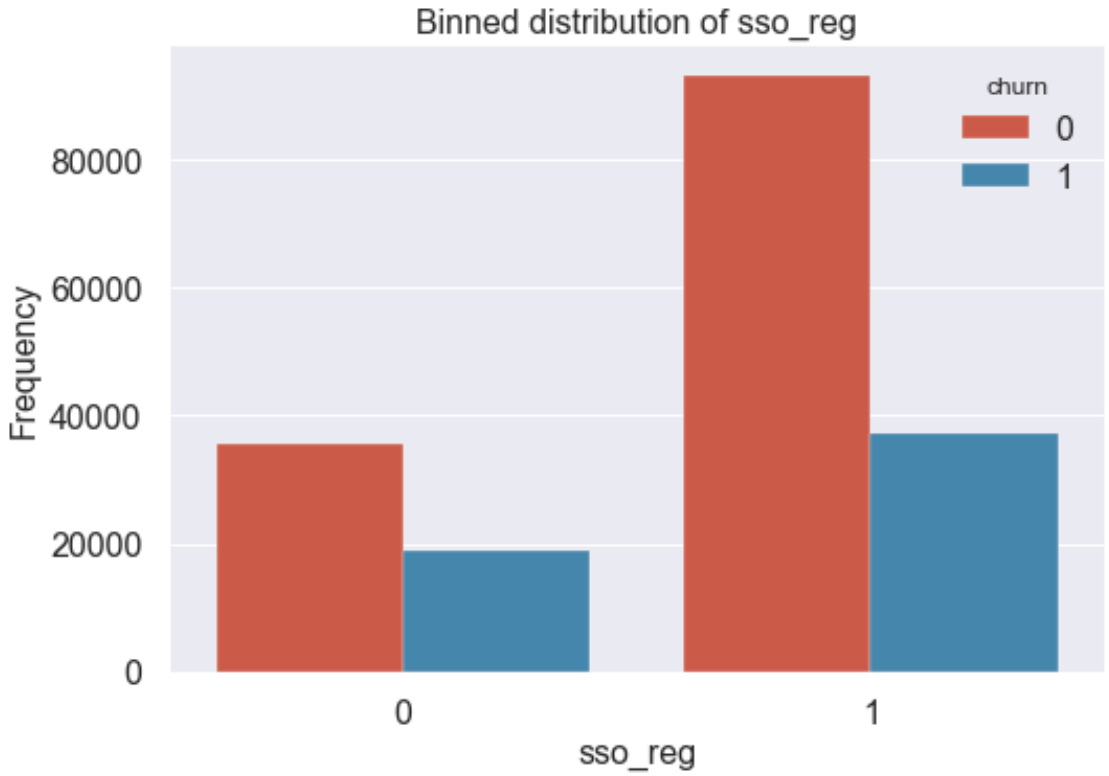

```
            count  mean  std  min  25%  50%  75%  max
boa_reg  184660.0   0.1   0.3  0.0  0.0  0.0  0.0  1.0
```



	count	mean	std	min	25%	50%	75%	max
che_reg	184660.0	0.1	0.3	0.0	0.0	0.0	0.0	1.0



	count	mean	std	min	25%	50%	75%	max
sit_reg	184660.0	0.1	0.3	0.0	0.0	0.0	0.0	1.0

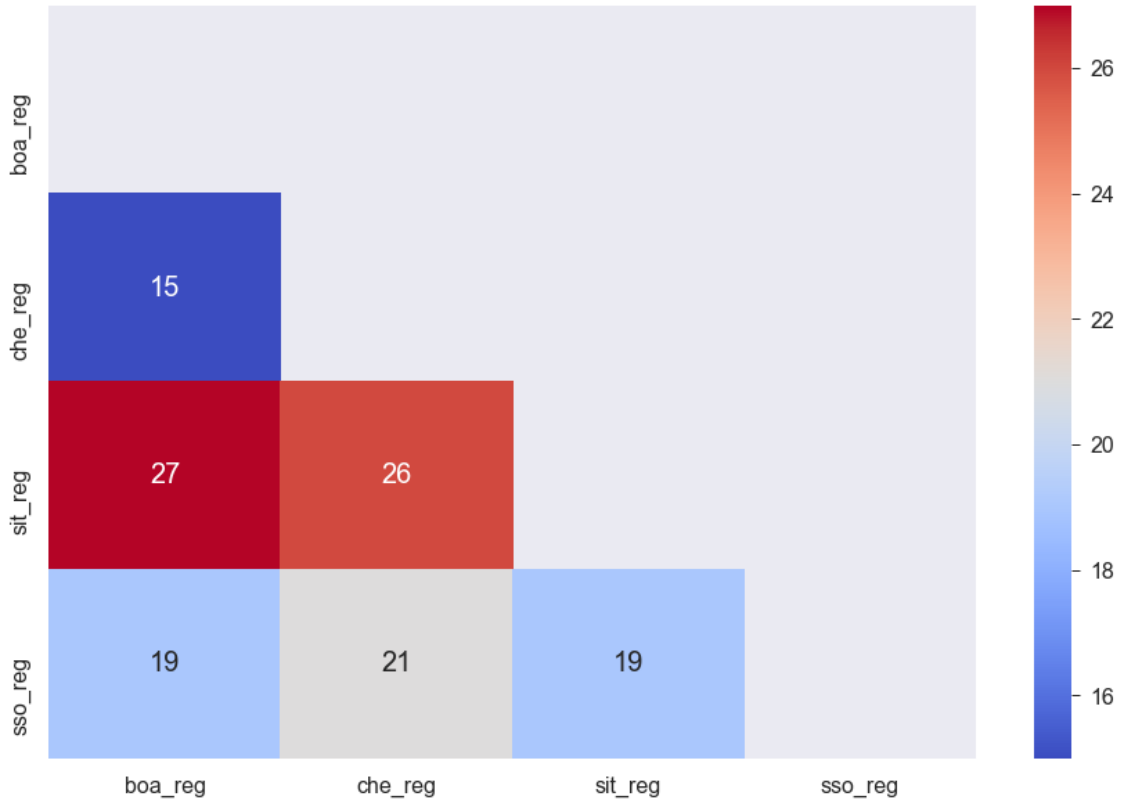


	count	mean	std	min	25%	50%	75%	max
sso_reg	184660.0	0.7	0.5	0.0	0.0	1.0	1.0	1.0

[201]:
```
eda.correlogram(df_reg)
```

[201]:

	boa_reg	che_reg	sit_reg	sso_reg
boa_reg	1.00	0.15	0.27	0.19
che_reg	0.15	1.00	0.26	0.21
sit_reg	0.27	0.26	1.00	0.19
sso_reg	0.19	0.21	0.19	1.00

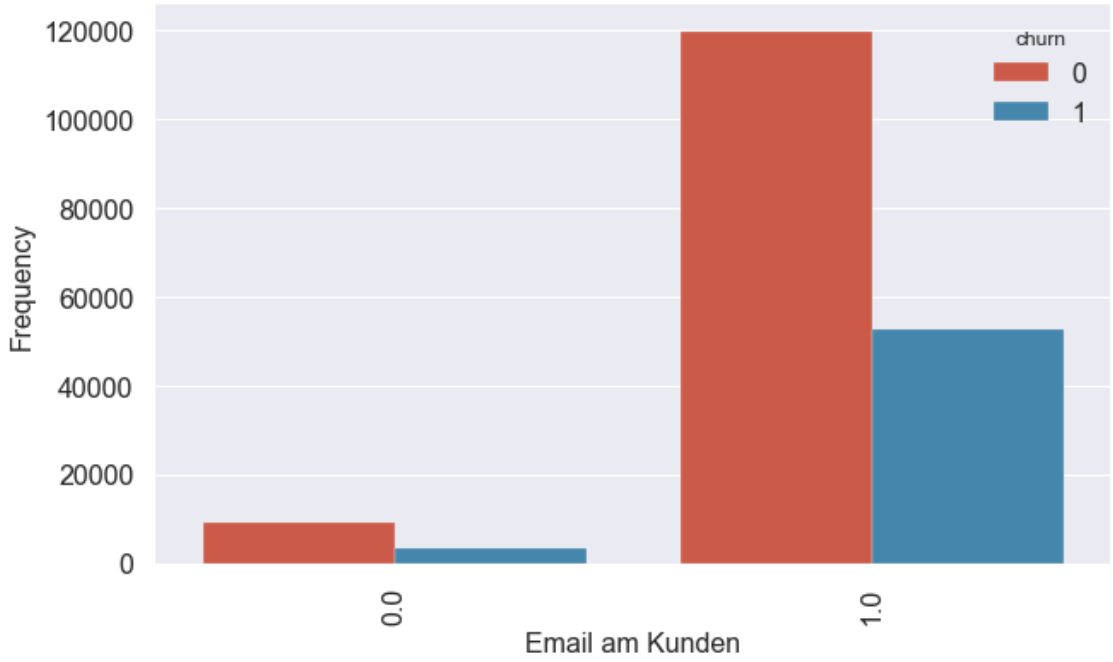


Observations: These features contain information, if clients have registered for special areas of the homepage. If it is 1 they have if not it is 0. As you can see most users, are not interested in these areas and services. The restricted hp areas (registration needed) show correlations between each other between 15 and 27.

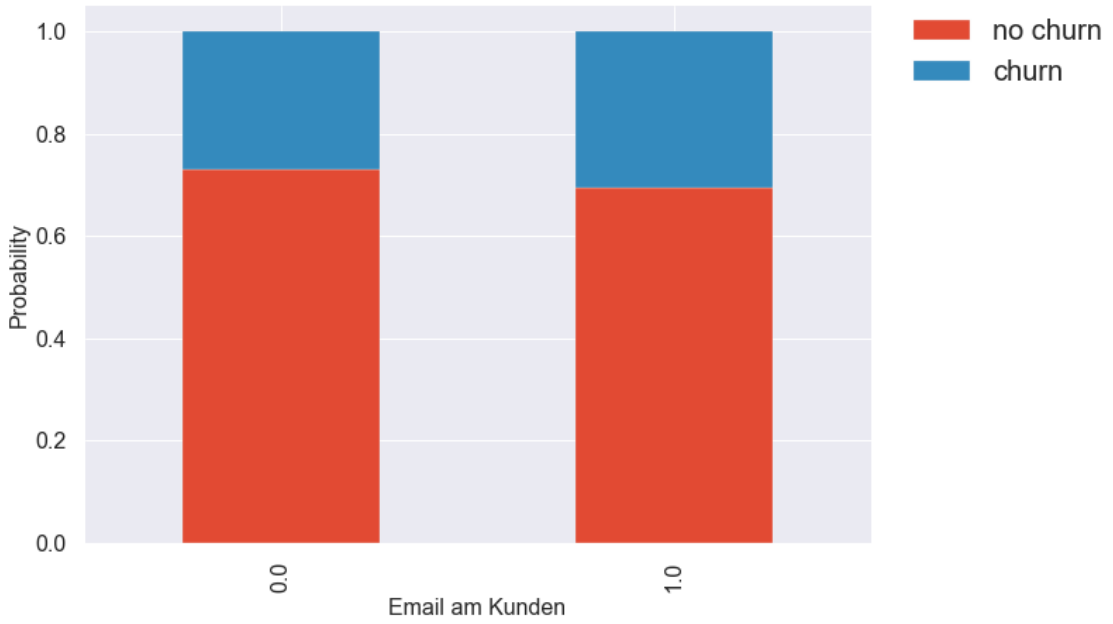
5.5.4 Customer Email

The feature email_am_kunden is a feature which shows if the email of an subscriber is known to the publisher. * 0:not known * 1:known

[202]:
```
plt.subplots(figsize=(10,6))
ax = sns.countplot(x='email_am_kunden', hue='churn', data=df)
ax.set(xlabel='Email am Kunden', ylabel='Frequency')
ax.set(ylim=(0, None))


ax.set_xticklabels(ax.get_xticklabels(), rotation=90);
```



[203]:
```
email_am_kunden_churn = crosstab_evaluation(df.email_am_kunden,df.churn)
crosstab_barplot(email_am_kunden_churn,['no churn','churn'],xlabelname='Email␣
↪am Kunden')
```



[204]:
```
df['email_am_kunden'].value_counts()/df.shape[0]
```

[204]:
```
1.0    0.933705
0.0    0.066295
Name: email_am_kunden, dtype: float64
```

Observations: * Only 6.4% of the mail adresses of all subscribers are unknown * The relative churn rate is a little bit higher for subscribers with known email adresses, but not a big difference is visible

5.5.5 Newsletter features

[205]:
```
eda.correlogram(df_nl_interact)
```

[205]:

	received_anzahl_1w	received_anzahl_1m	\
received_anzahl_1w	1.00	0.97	
received_anzahl_1m	0.97	1.00	
received_anzahl_3m	0.95	0.98	
received_anzahl_6m	0.94	0.97	
opened_anzahl_1w	0.58	0.57	
opened_anzahl_1m	0.60	0.62	
opened_anzahl_3m	0.59	0.61	
openedanzahl_6m	0.58	0.60	
clicked_anzahl_1w	0.26	0.25	
clicked_anzahl_1m	0.32	0.33	
clicked_anzahl_3m	0.33	0.33	
clicked_anzahl_6m	0.32	0.33	
unsubscribed_anzahl_1w	-0.00	0.02	
unsubscribed_anzahl_1m	-0.00	0.02	
unsubscribed_anzahl_3m	-0.02	-0.01	
unsubscribed_anzahl_6m	-0.04	-0.04	
openrate_1w	0.17	0.18	
clickrate_1w	0.11	0.10	
openrate_1m	0.18	0.18	
clickrate_1m	0.09	0.10	
openrate_3m	0.15	0.15	
clickrate_3m	0.07	0.07	

	received_anzahl_3m	received_anzahl_6m	\
received_anzahl_1w	0.95	0.94	
received_anzahl_1m	0.98	0.97	
received_anzahl_3m	1.00	0.99	
received_anzahl_6m	0.99	1.00	
opened_anzahl_1w	0.56	0.55	
opened_anzahl_1m	0.61	0.60	
opened_anzahl_3m	0.62	0.61	
openedanzahl_6m	0.62	0.62	
clicked_anzahl_1w	0.24	0.23	
clicked_anzahl_1m	0.32	0.31	

clicked_anzahl_3m	0.33	0.33
clicked_anzahl_6m	0.33	0.33
unsubscribed_anzahl_1w	0.03	0.03
unsubscribed_anzahl_1m	0.03	0.03
unsubscribed_anzahl_3m	0.01	0.03
unsubscribed_anzahl_6m	-0.02	0.01
openrate_1w	0.18	0.17
clickrate_1w	0.10	0.10
openrate_1m	0.18	0.18
clickrate_1m	0.09	0.09
openrate_3m	0.15	0.15
clickrate_3m	0.07	0.07

	opened_anzahl_1w	opened_anzahl_1m	opened_anzahl_3m	\
received_anzahl_1w	0.58	0.60	0.59	
received_anzahl_1m	0.57	0.62	0.61	
received_anzahl_3m	0.56	0.61	0.62	
received_anzahl_6m	0.55	0.60	0.61	
opened_anzahl_1w	1.00	0.91	0.88	
opened_anzahl_1m	0.91	1.00	0.97	
opened_anzahl_3m	0.88	0.97	1.00	
openedanzahl_6m	0.86	0.94	0.98	
clicked_anzahl_1w	0.42	0.39	0.37	
clicked_anzahl_1m	0.47	0.50	0.48	
clicked_anzahl_3m	0.46	0.50	0.51	
clicked_anzahl_6m	0.45	0.49	0.50	
unsubscribed_anzahl_1w	0.03	0.04	0.04	
unsubscribed_anzahl_1m	0.03	0.04	0.04	
unsubscribed_anzahl_3m	0.02	0.02	0.04	
unsubscribed_anzahl_6m	0.01	0.01	0.02	
openrate_1w	0.62	0.50	0.48	
clickrate_1w	0.15	0.12	0.12	
openrate_1m	0.56	0.60	0.57	
clickrate_1m	0.07	0.08	0.07	
openrate_3m	0.51	0.55	0.57	
clickrate_3m	0.02	0.01	0.01	

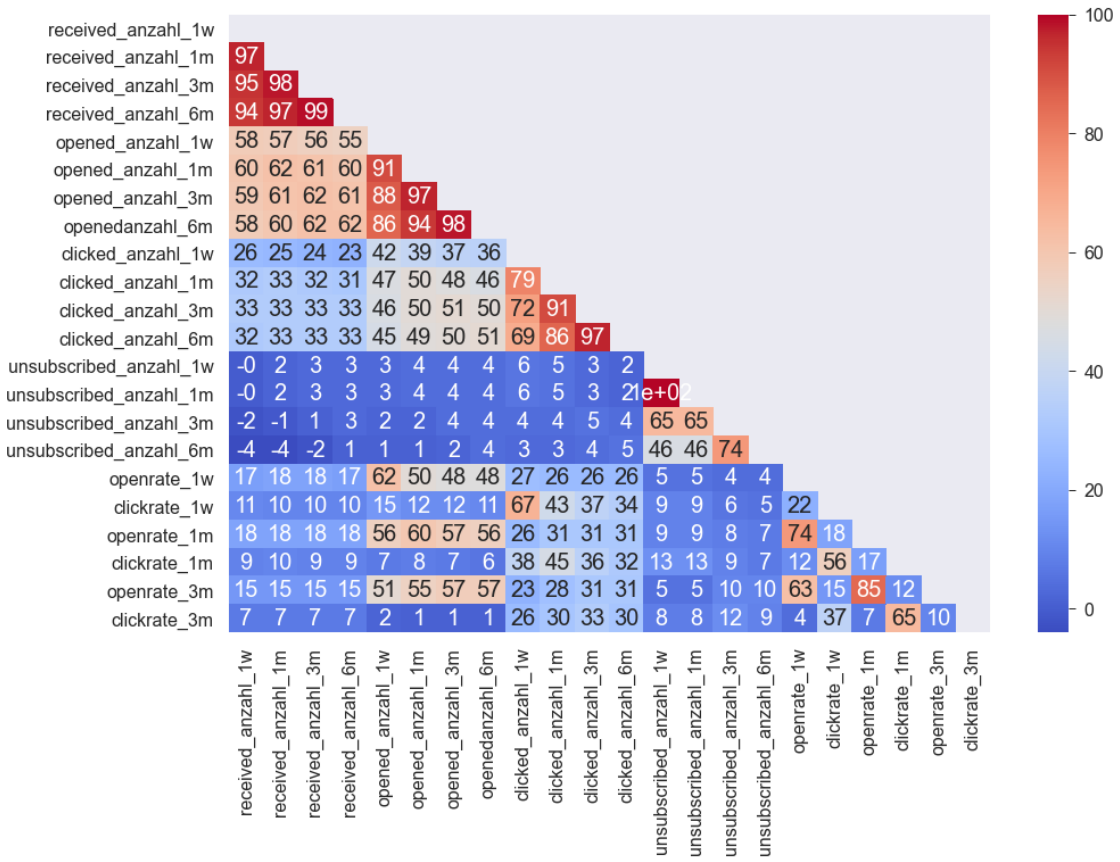
	openedanzahl_6m	clicked_anzahl_1w	clicked_anzahl_1m	\
received_anzahl_1w	0.58	0.26	0.32	
received_anzahl_1m	0.60	0.25	0.33	
received_anzahl_3m	0.62	0.24	0.32	
received_anzahl_6m	0.62	0.23	0.31	
opened_anzahl_1w	0.86	0.42	0.47	
opened_anzahl_1m	0.94	0.39	0.50	
opened_anzahl_3m	0.98	0.37	0.48	
openedanzahl_6m	1.00	0.36	0.46	
clicked_anzahl_1w	0.36	1.00	0.79	

```
clicked_anzahl_1m       0.46              0.79              1.00
clicked_anzahl_3m       0.50              0.72              0.91
clicked_anzahl_6m       0.51              0.69              0.86
unsubscribed_anzahl_1w  0.04              0.06              0.05
unsubscribed_anzahl_1m  0.04              0.06              0.05
unsubscribed_anzahl_3m  0.04              0.04              0.04
unsubscribed_anzahl_6m  0.04              0.03              0.03
openrate_1w             0.48              0.27              0.26
clickrate_1w            0.11              0.67              0.43
openrate_1m             0.56              0.26              0.31
clickrate_1m            0.06              0.38              0.45
openrate_3m             0.57              0.23              0.28
clickrate_3m            0.01              0.26              0.30

                        clicked_anzahl_3m  clicked_anzahl_6m  \
received_anzahl_1w      0.33               0.32
received_anzahl_1m      0.33               0.33
received_anzahl_3m      0.33               0.33
received_anzahl_6m      0.33               0.33
opened_anzahl_1w        0.46               0.45
opened_anzahl_1m        0.50               0.49
opened_anzahl_3m        0.51               0.50
openedanzahl_6m         0.50               0.51
clicked_anzahl_1w       0.72               0.69
clicked_anzahl_1m       0.91               0.86
clicked_anzahl_3m       1.00               0.97
clicked_anzahl_6m       0.97               1.00
unsubscribed_anzahl_1w  0.03               0.02
unsubscribed_anzahl_1m  0.03               0.02
unsubscribed_anzahl_3m  0.05               0.04
unsubscribed_anzahl_6m  0.04               0.05
openrate_1w             0.26               0.26
clickrate_1w            0.37               0.34
openrate_1m             0.31               0.31
clickrate_1m            0.36               0.32
openrate_3m             0.31               0.31
clickrate_3m            0.33               0.30

                        unsubscribed_anzahl_1w  unsubscribed_anzahl_1m  \
received_anzahl_1w     -0.00                   -0.00
received_anzahl_1m      0.02                    0.02
received_anzahl_3m      0.03                    0.03
received_anzahl_6m      0.03                    0.03
opened_anzahl_1w        0.03                    0.03
opened_anzahl_1m        0.04                    0.04
opened_anzahl_3m        0.04                    0.04
openedanzahl_6m         0.04                    0.04
```

```
clicked_anzahl_1w       0.06                    0.06
clicked_anzahl_1m       0.05                    0.05
clicked_anzahl_3m       0.03                    0.03
clicked_anzahl_6m       0.02                    0.02
unsubscribed_anzahl_1w  1.00                    1.00
unsubscribed_anzahl_1m  1.00                    1.00
unsubscribed_anzahl_3m  0.65                    0.65
unsubscribed_anzahl_6m  0.46                    0.46
openrate_1w             0.05                    0.05
clickrate_1w            0.09                    0.09
openrate_1m             0.09                    0.09
clickrate_1m            0.13                    0.13
openrate_3m             0.05                    0.05
clickrate_3m            0.08                    0.08

                        unsubscribed_anzahl_3m  unsubscribed_anzahl_6m  \
received_anzahl_1w     -0.02                   -0.04
received_anzahl_1m     -0.01                   -0.04
received_anzahl_3m      0.01                   -0.02
received_anzahl_6m      0.03                    0.01
opened_anzahl_1w        0.02                    0.01
opened_anzahl_1m        0.02                    0.01
opened_anzahl_3m        0.04                    0.02
openedanzahl_6m         0.04                    0.04
clicked_anzahl_1w       0.04                    0.03
clicked_anzahl_1m       0.04                    0.03
clicked_anzahl_3m       0.05                    0.04
clicked_anzahl_6m       0.04                    0.05
unsubscribed_anzahl_1w  0.65                    0.46
unsubscribed_anzahl_1m  0.65                    0.46
unsubscribed_anzahl_3m  1.00                    0.74
unsubscribed_anzahl_6m  0.74                    1.00
openrate_1w             0.04                    0.04
clickrate_1w            0.06                    0.05
openrate_1m             0.08                    0.07
clickrate_1m            0.09                    0.07
openrate_3m             0.10                    0.10
clickrate_3m            0.12                    0.09

                        openrate_1w  clickrate_1w  openrate_1m  clickrate_1m  \
received_anzahl_1w      0.17         0.11          0.18         0.09
received_anzahl_1m      0.18         0.10          0.18         0.10
received_anzahl_3m      0.18         0.10          0.18         0.09
received_anzahl_6m      0.17         0.10          0.18         0.09
opened_anzahl_1w        0.62         0.15          0.56         0.07
opened_anzahl_1m        0.50         0.12          0.60         0.08
opened_anzahl_3m        0.48         0.12          0.57         0.07
```

openedanzahl_6m	0.48	0.11	0.56	0.06
clicked_anzahl_1w	0.27	0.67	0.26	0.38
clicked_anzahl_1m	0.26	0.43	0.31	0.45
clicked_anzahl_3m	0.26	0.37	0.31	0.36
clicked_anzahl_6m	0.26	0.34	0.31	0.32
unsubscribed_anzahl_1w	0.05	0.09	0.09	0.13
unsubscribed_anzahl_1m	0.05	0.09	0.09	0.13
unsubscribed_anzahl_3m	0.04	0.06	0.08	0.09
unsubscribed_anzahl_6m	0.04	0.05	0.07	0.07
openrate_1w	1.00	0.22	0.74	0.12
clickrate_1w	0.22	1.00	0.18	0.56
openrate_1m	0.74	0.18	1.00	0.17
clickrate_1m	0.12	0.56	0.17	1.00
openrate_3m	0.63	0.15	0.85	0.12
clickrate_3m	0.04	0.37	0.07	0.65

	openrate_3m	clickrate_3m
received_anzahl_1w	0.15	0.07
received_anzahl_1m	0.15	0.07
received_anzahl_3m	0.15	0.07
received_anzahl_6m	0.15	0.07
opened_anzahl_1w	0.51	0.02
opened_anzahl_1m	0.55	0.01
opened_anzahl_3m	0.57	0.01
openedanzahl_6m	0.57	0.01
clicked_anzahl_1w	0.23	0.26
clicked_anzahl_1m	0.28	0.30
clicked_anzahl_3m	0.31	0.33
clicked_anzahl_6m	0.31	0.30
unsubscribed_anzahl_1w	0.05	0.08
unsubscribed_anzahl_1m	0.05	0.08
unsubscribed_anzahl_3m	0.10	0.12
unsubscribed_anzahl_6m	0.10	0.09
openrate_1w	0.63	0.04
clickrate_1w	0.15	0.37
openrate_1m	0.85	0.07
clickrate_1m	0.12	0.65
openrate_3m	1.00	0.10
clickrate_3m	0.10	1.00

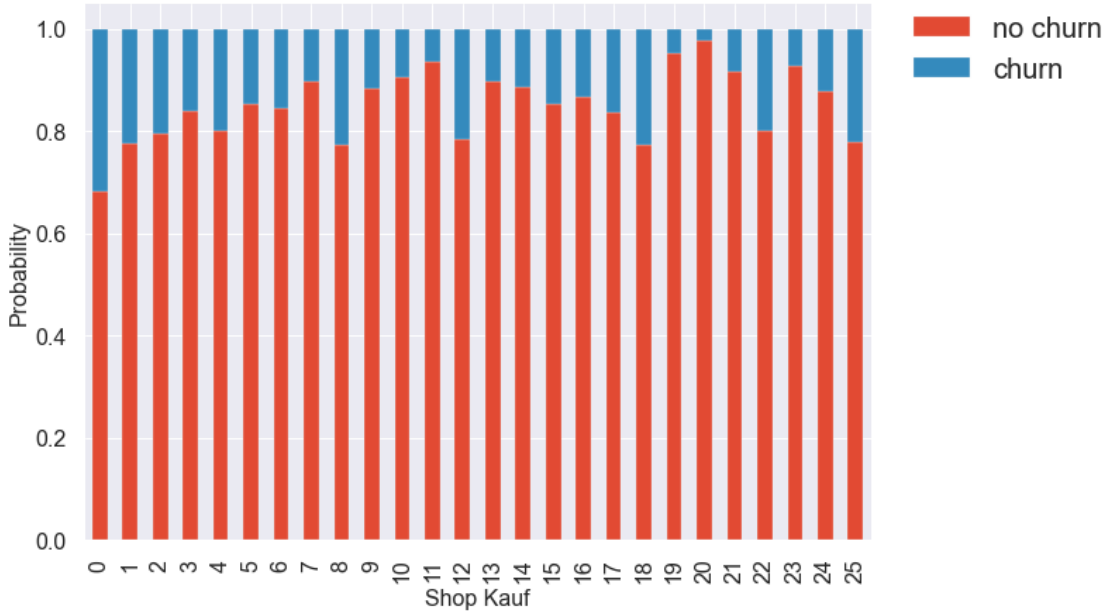


Observations: df_nl holds the four feature "nl_zeitbrief", "nl_zeitshop", "nl_zeitvrlag_hamburg" and "nl_aktivitaet". The only correlation is between "nl_aktivitaet" and "nl_zeitbrief". Although this is not surprising, because the feature "nl_aktivitaet" is an aggregation of user activity on the newsletters.

5.5.6 Shop Kauf

[206]:
```
labellist = ['no churn','churn']

plt.subplots(figsize=(20,6))
ax = sns.countplot(x='shop_kauf', hue='churn', data=df)
ax.set(ylabel='Frequency')
ax.set(ylim=(0, 400))
plt.title('Shop Kauf', fontsize=22)
L=plt.legend(fontsize=20,loc=(1.04,0.83))
L.get_texts()[0].set_text(labellist[0])
L.get_texts()[1].set_text(labellist[1])

ax.set_xticklabels(ax.get_xticklabels(), rotation=90);
```

[207]:
```
shop_churn = crosstab_evaluation(df.shop_kauf,df.churn)
crosstab_barplot(shop_churn,['no churn','churn'],xlabelname='Shop Kauf')
plt.xlim(-0.5,25.5);
```

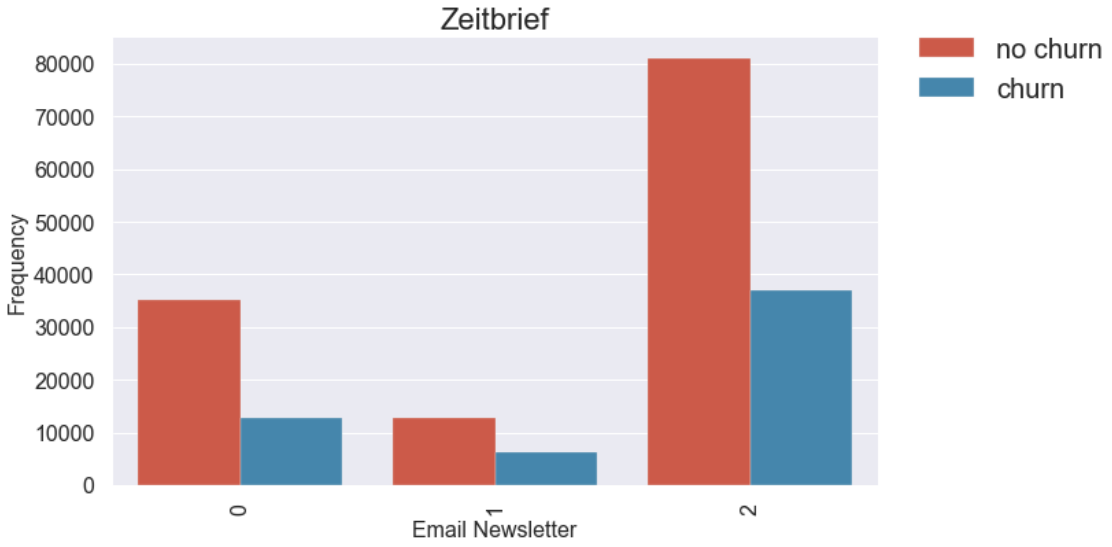


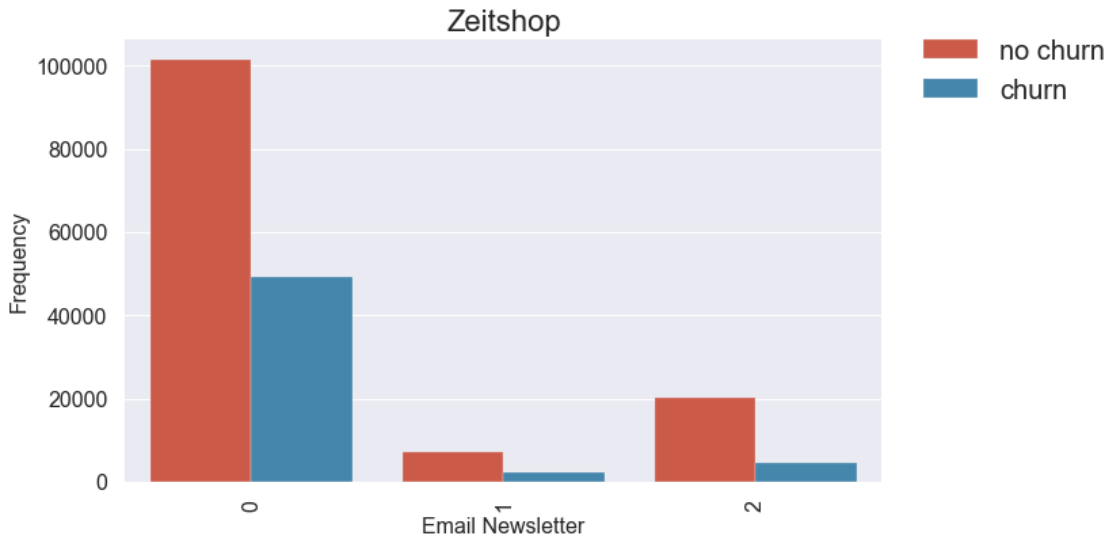
5.5.7 Newsletter

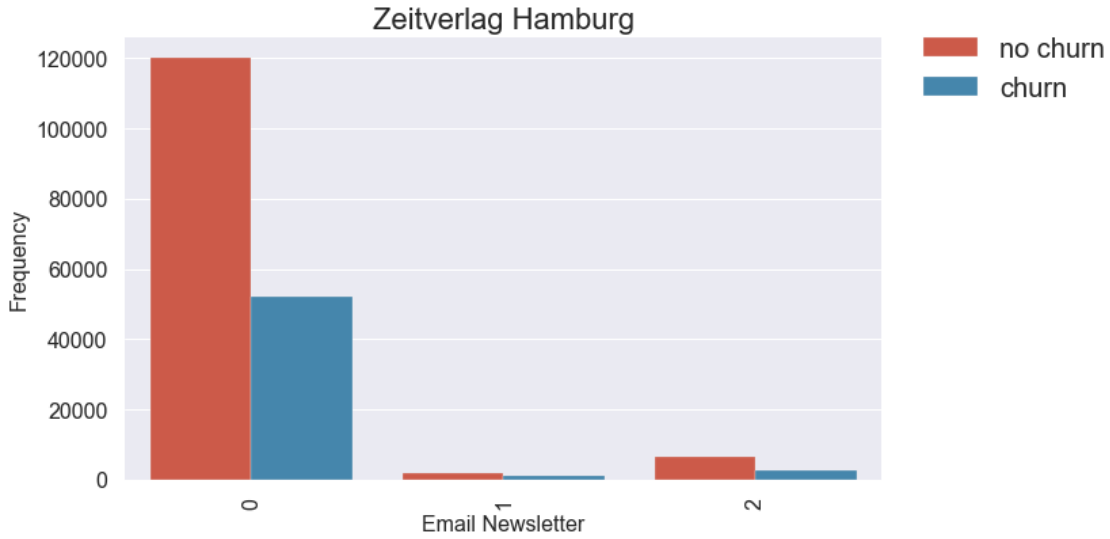
Email-Newsletter: with: 0:not available, 1: unsubscribed, 2: active * nl_zeitbrief * nl_zeitshop * nl_zeitverlag_hamburg * nl_fdz_organisch: on organic recipient list loyalty program fdZ

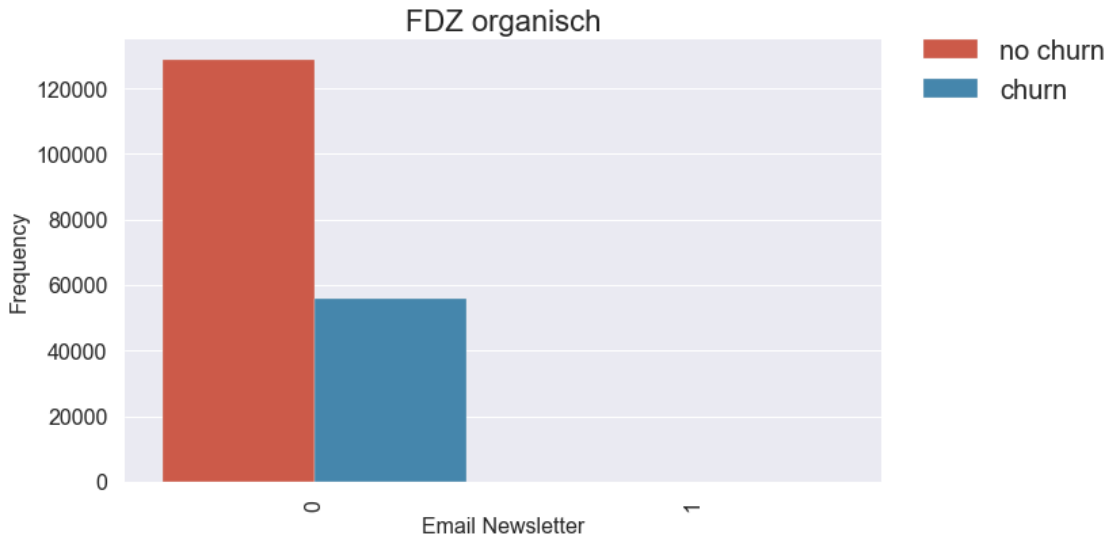
- Blacklist: Number of blacklist entries
- Bounced: Number of Bounces
- Aktivität: Number on how many newsletters in the house
- Sperrliste
- nl_opt_in_sum: Number of Optins

[208]:
```
title = ['Zeitbrief', 'Zeitshop', "Zeitverlag Hamburg", 'FDZ organisch']
labellist = ['no churn','churn']

for i, nl in enumerate(['nl_zeitbrief', 'nl_zeitshop', 'nl_zeitverlag_hamburg',␣
↪'nl_fdz_organisch']):


    plt.subplots(figsize=(10,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Email Newsletter', ylabel='Frequency')
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```

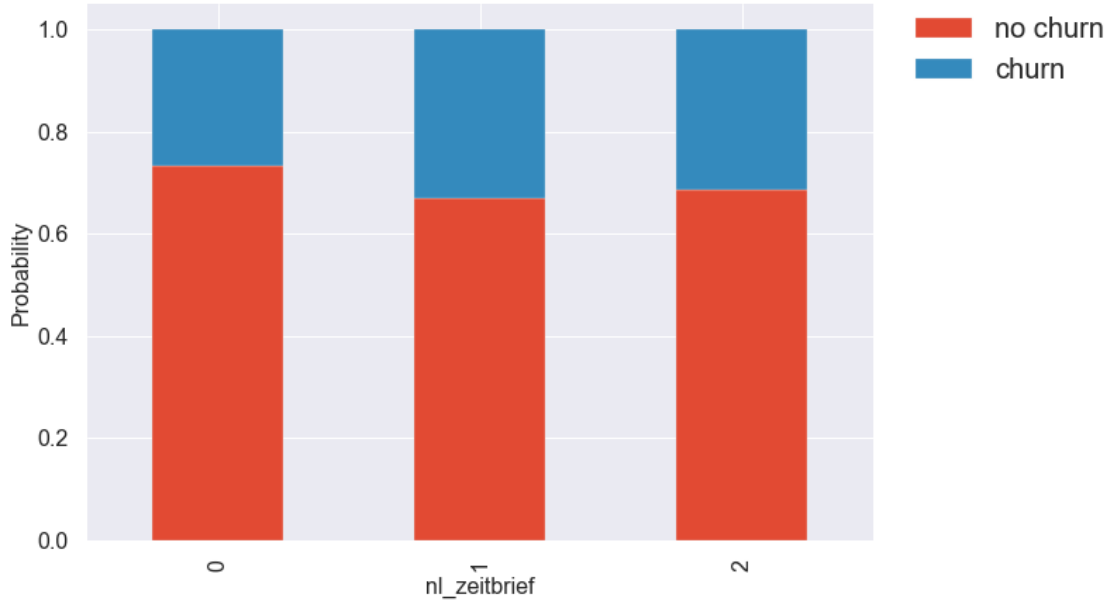


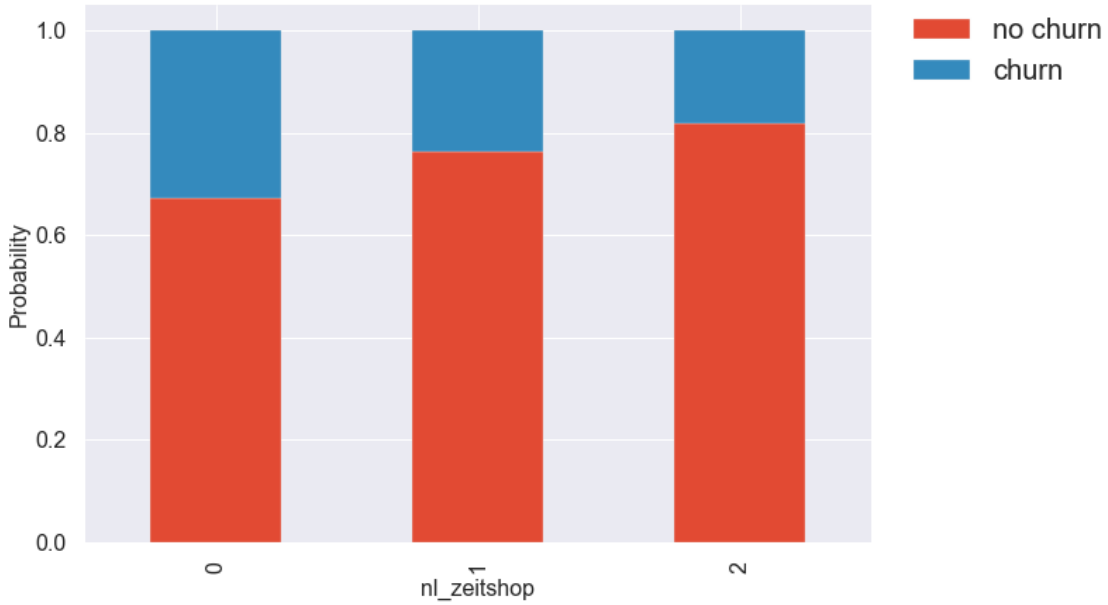

Zeitshop
no churn
churn
Frequency
100000
80000
60000
40000
20000
0
0
1
2
Email Newsletter

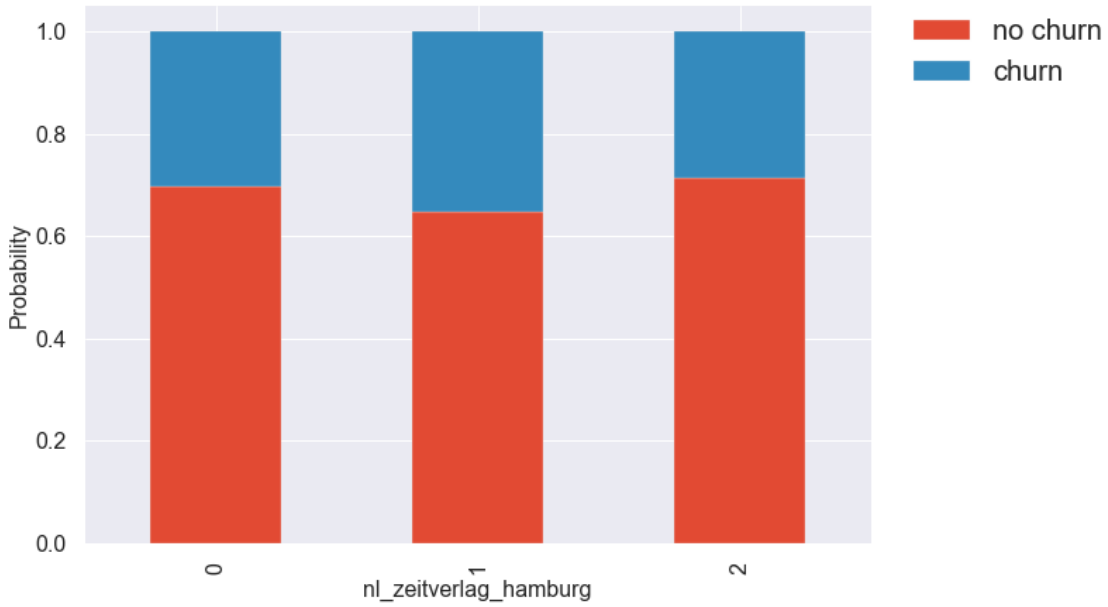

Zeitverlag Hamburg
no churn
churn
Frequency
120000
100000
80000
60000
40000
20000
0
0
1
2
Email Newsletter

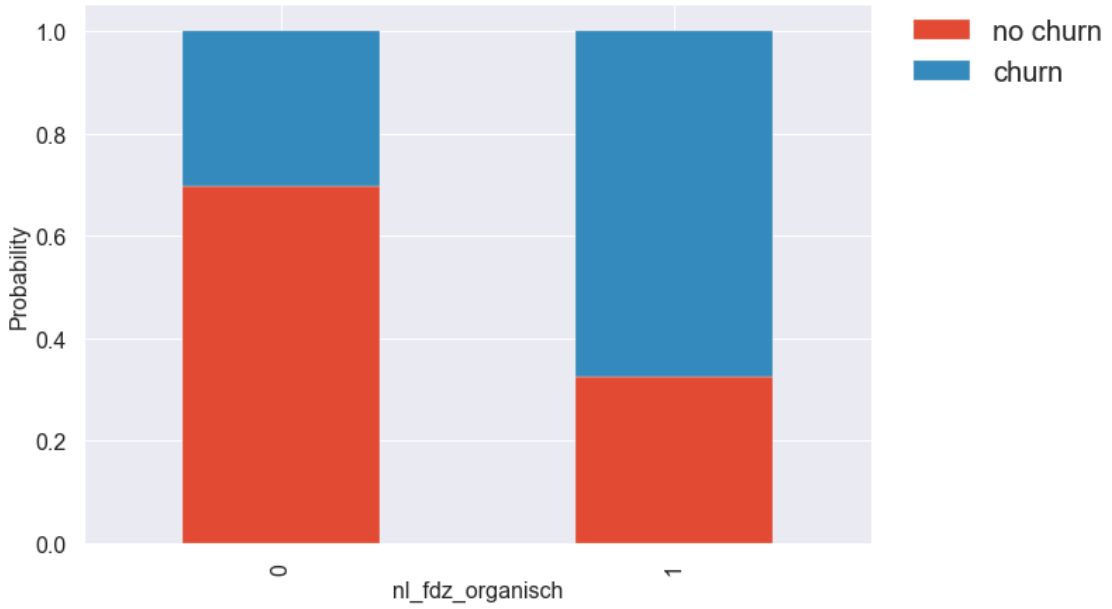


[209]:
```
for i in ['nl_zeitbrief', 'nl_zeitshop', 'nl_zeitverlag_hamburg',␣
↪'nl_fdz_organisch']:
    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i)
```



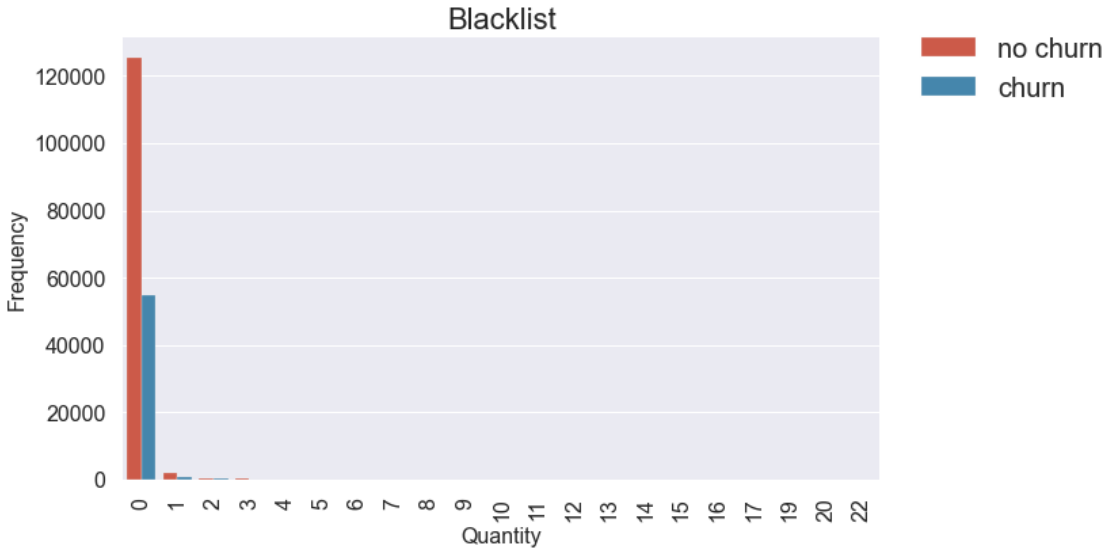

1.0
0.8
0.6
0.4
0.2
0.0
Probability
0
1
2
nl_zeitshop
no churn
churn

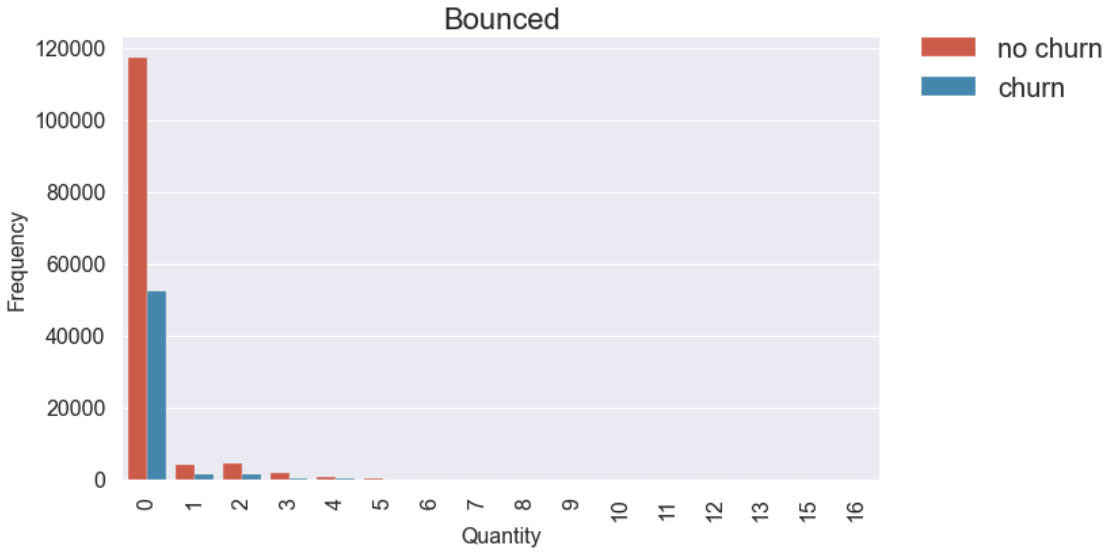

1.0
0.8
0.6
0.4
0.2
0.0
Probability
0
1
2
nl_zeitverlag_hamburg
no churn
churn



Absolute Blacklist and Bounced

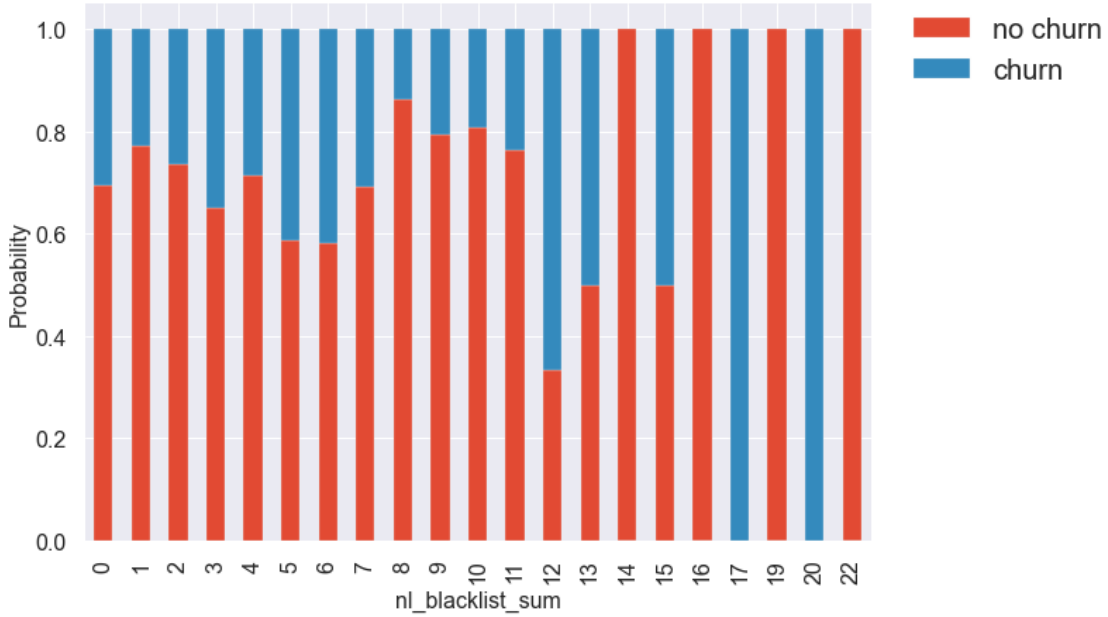
[210]:
```
title = ['Blacklist', 'Bounced']
labellist = ['no churn','churn']

for i, nl in enumerate(['nl_blacklist_sum', 'nl_bounced_sum']):


    plt.subplots(figsize=(10,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```

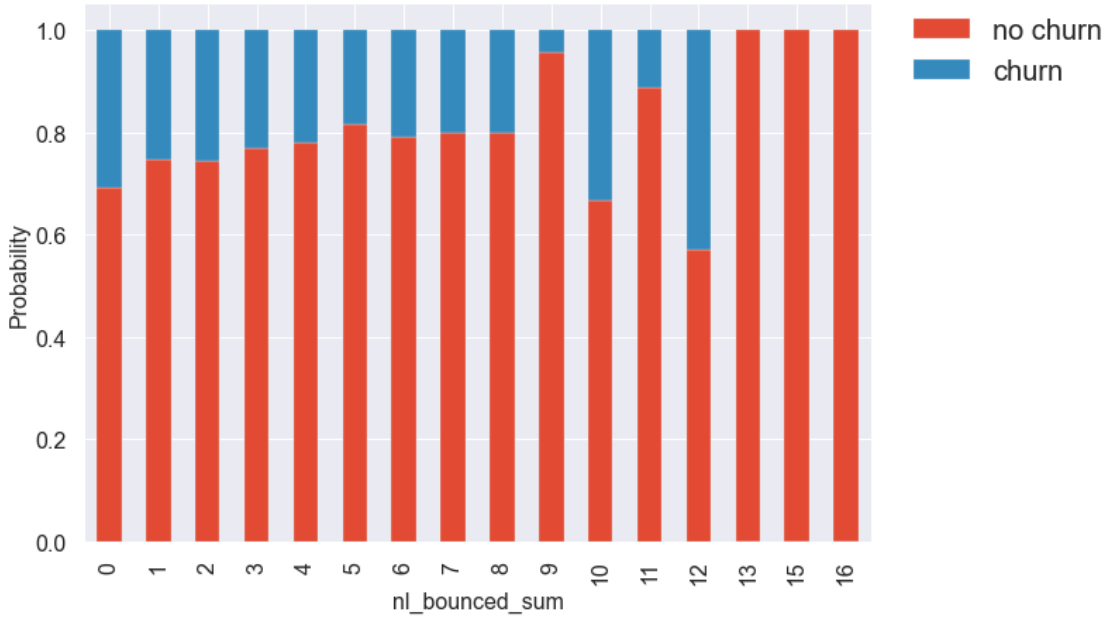




Relative Blacklist and Bounced

[211]:
```
for i in ['nl_blacklist_sum', 'nl_bounced_sum']:
    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i)
```

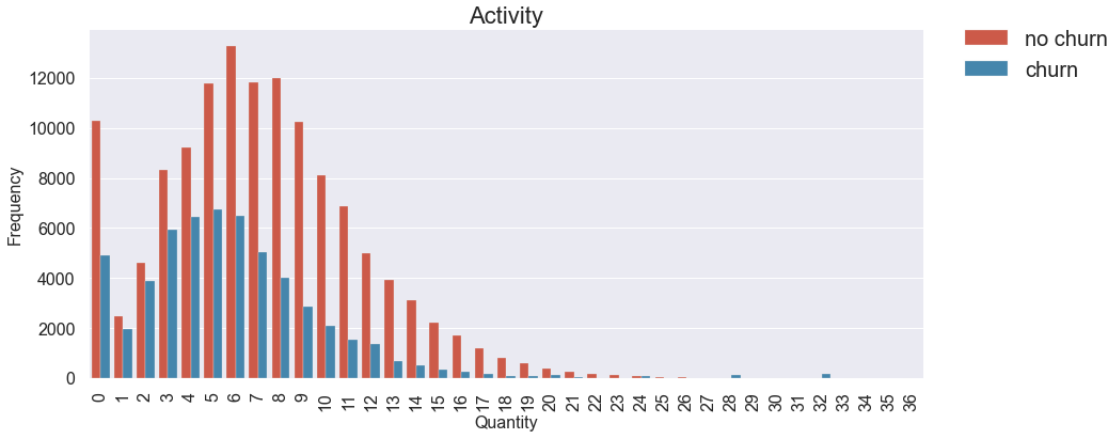




Absolute NL Aktivitaet and Sperrliste

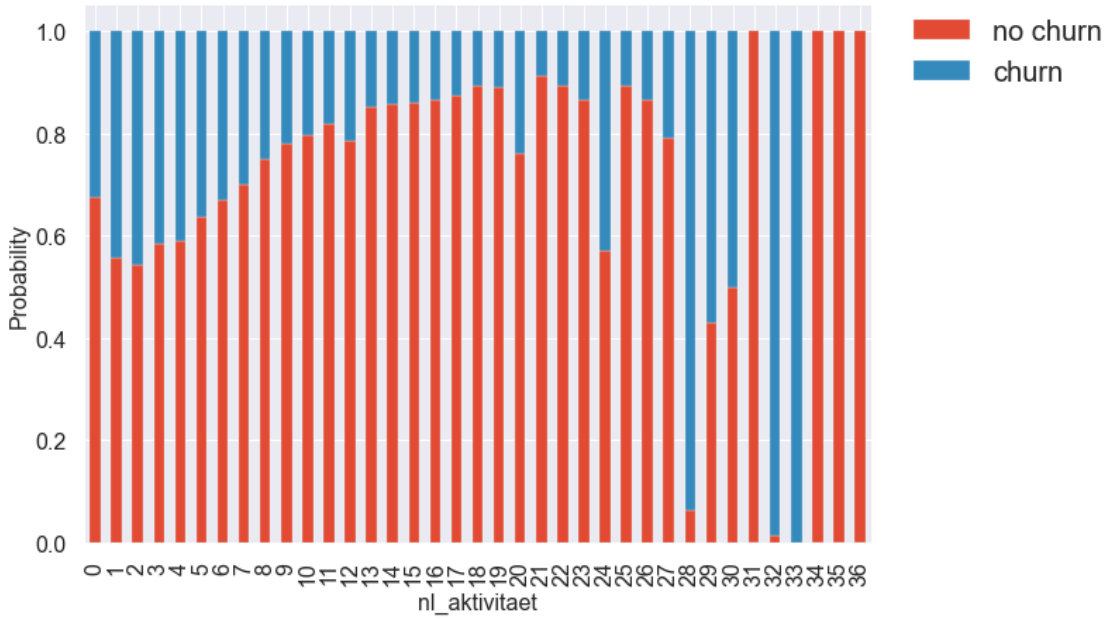
[212]:
```
title = ['Activity', 'Blocking list']
labellist = ['no churn','churn']

for i, nl in enumerate(['nl_aktivitaet', 'nl_sperrliste_sum']):
```

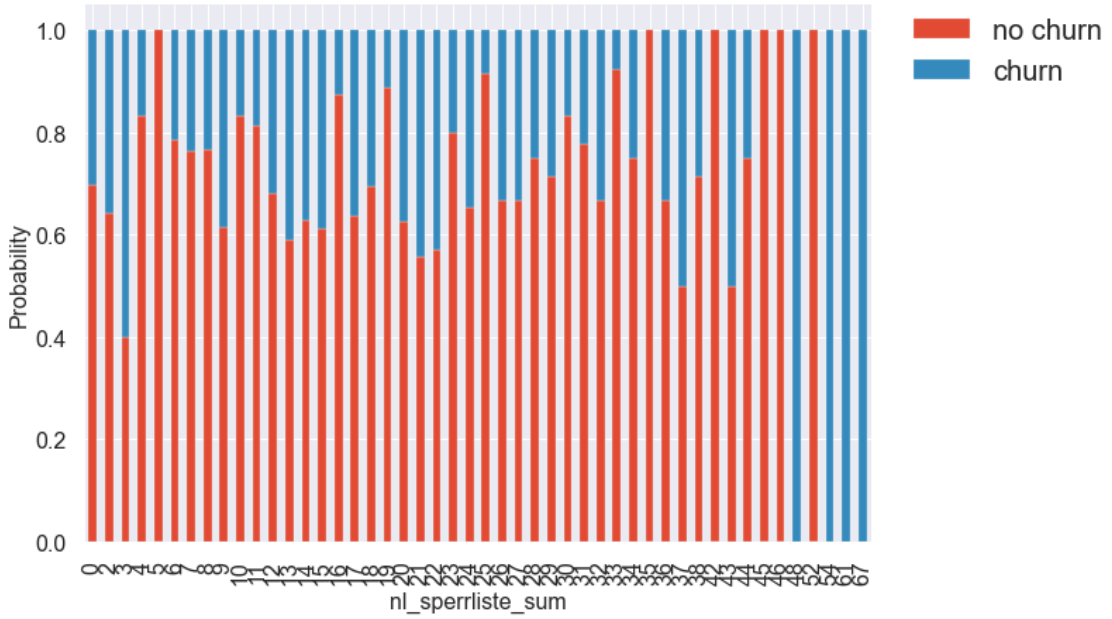
```
    plt.subplots(figsize=(14,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```



Relative NL Aktivität and Sperrliste

[213]:
```
for i in ['nl_aktivitaet', 'nl_sperrliste_sum']:
    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i)
```

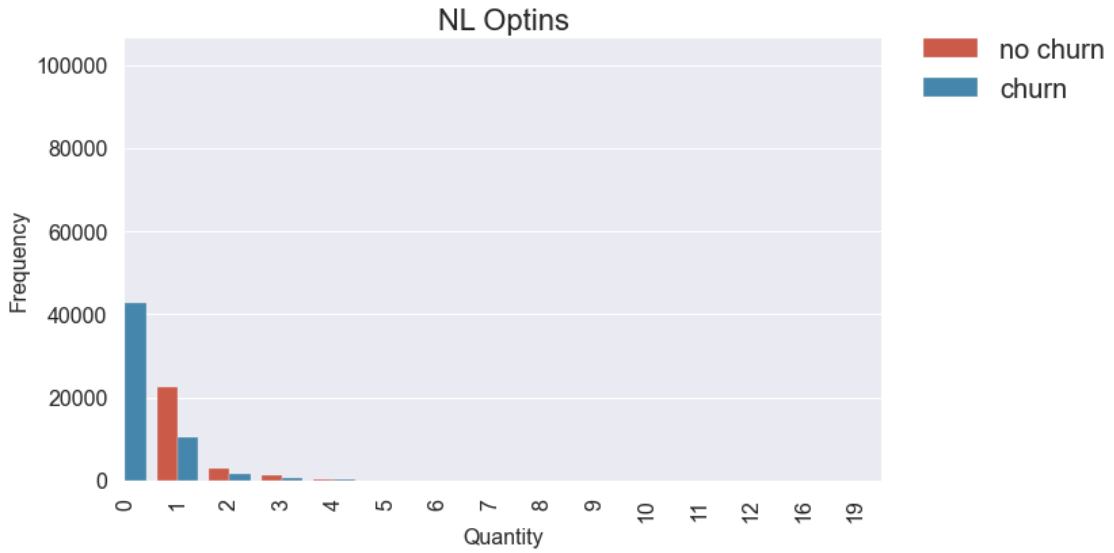




Absolute NL Optins

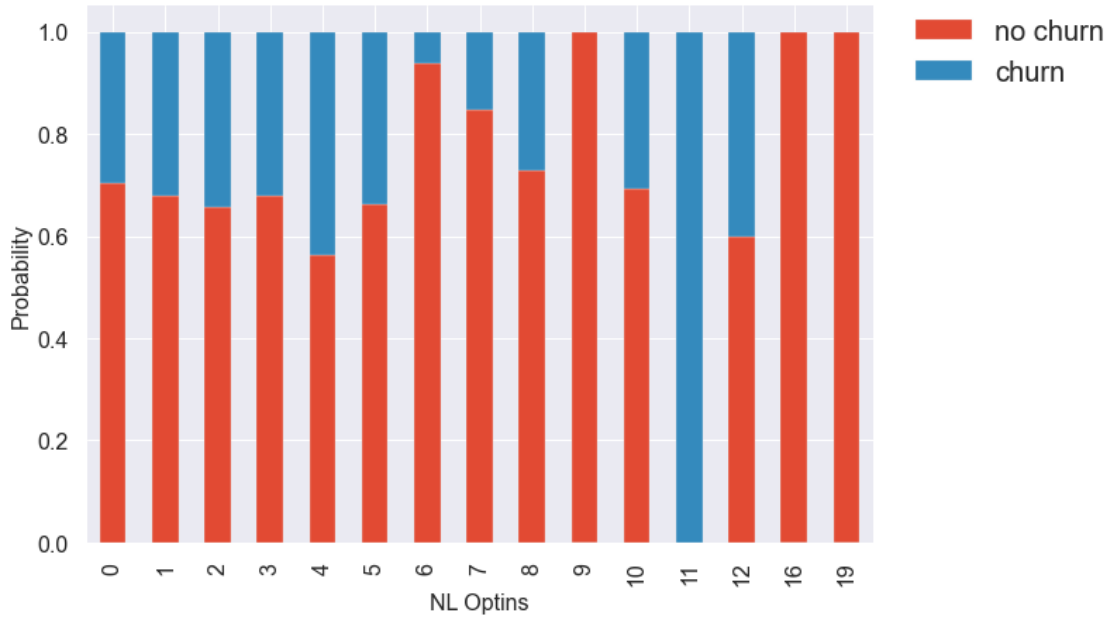
[214]:
```
plt.subplots(figsize=(10,6))
ax = sns.countplot(x=df['nl_opt_in_sum'], hue='churn', data=df)
ax.set(xlabel='Quantity', ylabel='Frequency', xlim=[0,None])
plt.title('NL Optins', fontsize=22)
L=plt.legend(fontsize=20,loc=(1.04,0.83))
```

```
L.get_texts()[0].set_text(labellist[0])
L.get_texts()[1].set_text(labellist[1])
ax.set_xticklabels(ax.get_xticklabels(), rotation=90);
```



Relative NL Optins

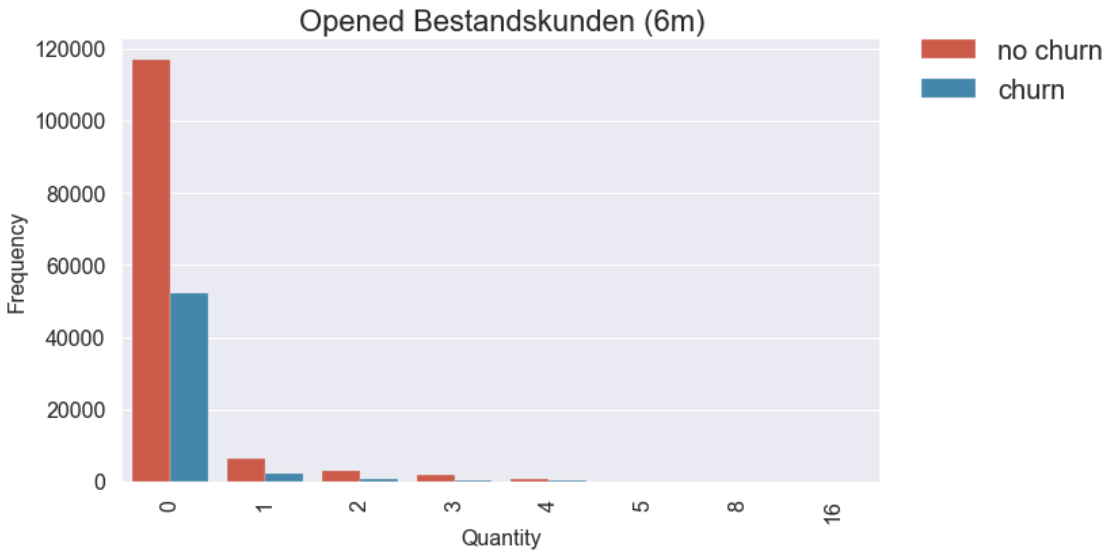
[215]:
```
x = crosstab_evaluation(df['nl_opt_in_sum'],df.churn)
crosstab_barplot(x,['no churn','churn'],xlabelname='NL Optins')
```

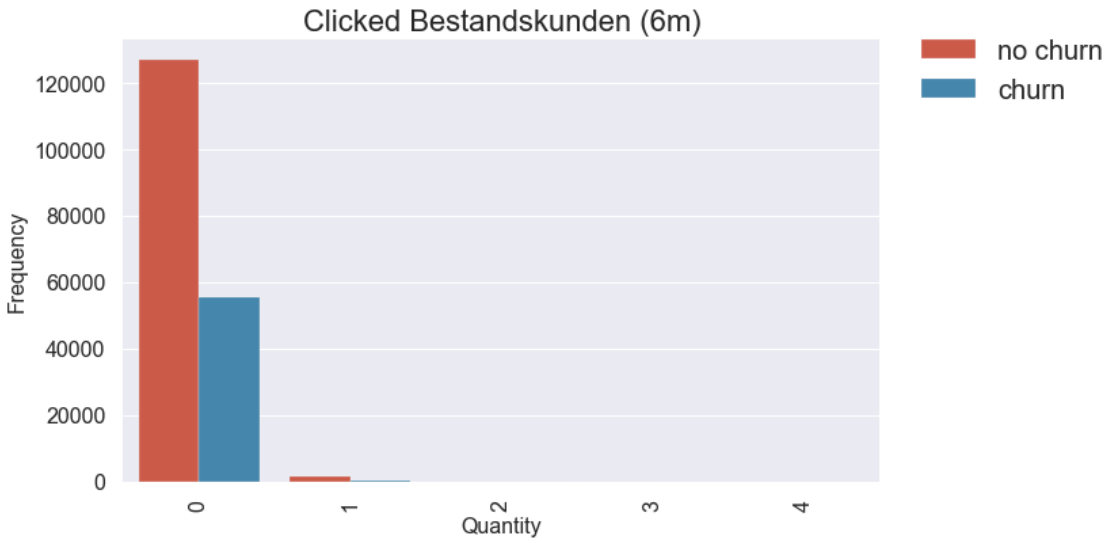


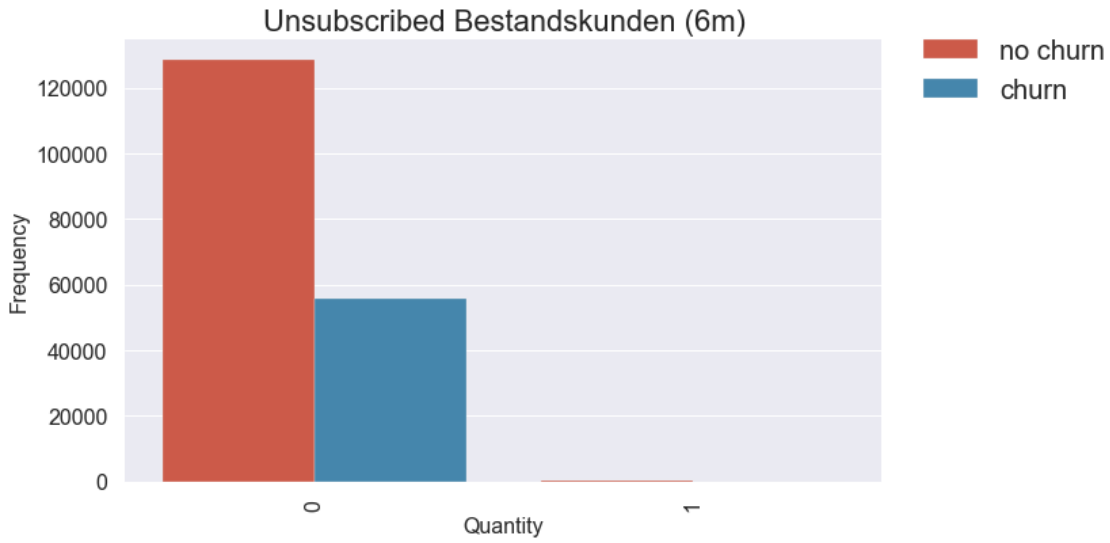
Bestandskunden

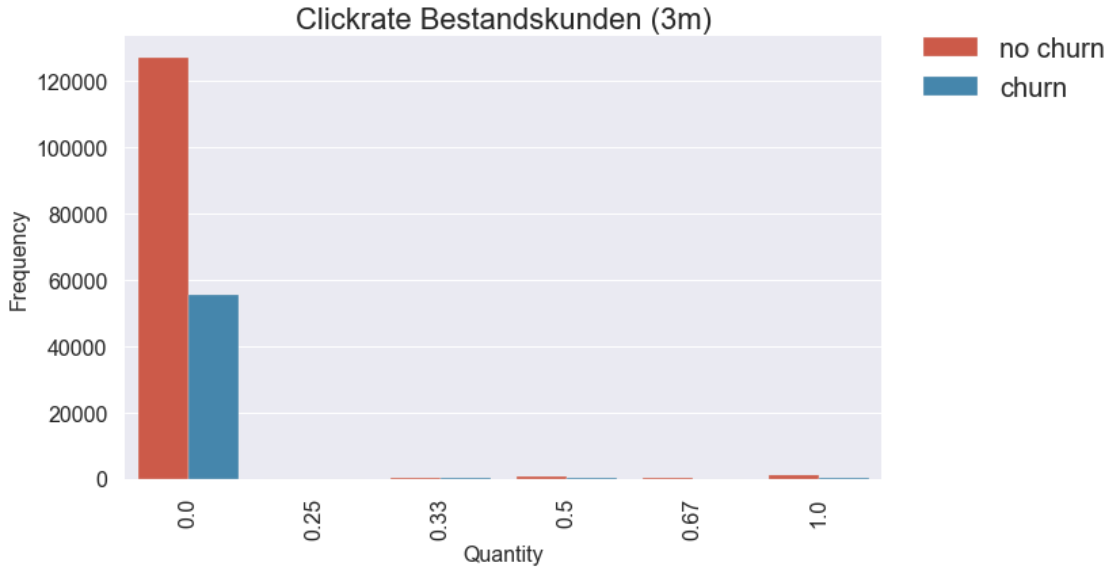
- with received, opened, clicked, unsubscribed mails
- time period = 6m

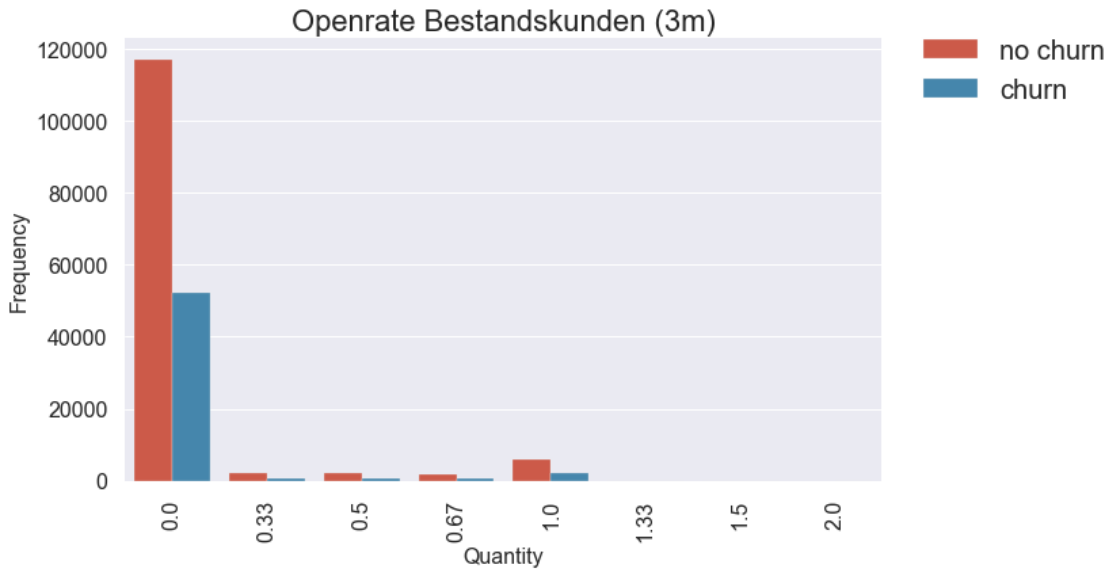
[216]:
```
title = ['Received Bestandskunden (6m)', 'Opened Bestandskunden (6m)', 'Clicked␣
 ↪Bestandskunden (6m)', 'Unsubscribed Bestandskunden (6m)', 'Clickrate␣
 ↪Bestandskunden (3m)', 'Openrate Bestandskunden (3m)']
labellist = ['no churn','churn']

for i, nl in enumerate(['received_anzahl_bestandskunden_6m',␣
 ↪'openedanzahl_bestandskunden_6m', 'clicked_anzahl_bestandskunden_6m',␣
 ↪'unsubscribed_anzahl_bestandskunden_6m', 'clickrate_bestandskunden_3m',␣
 ↪'openrate_bestandskunden_3m']):


    plt.subplots(figsize=(10,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```


Opened Bestandskunden (6m)
no churn
churn
Frequency
120000
100000
80000
60000
40000
20000
0
0
1
2
3
4
5
8
16
Quantity

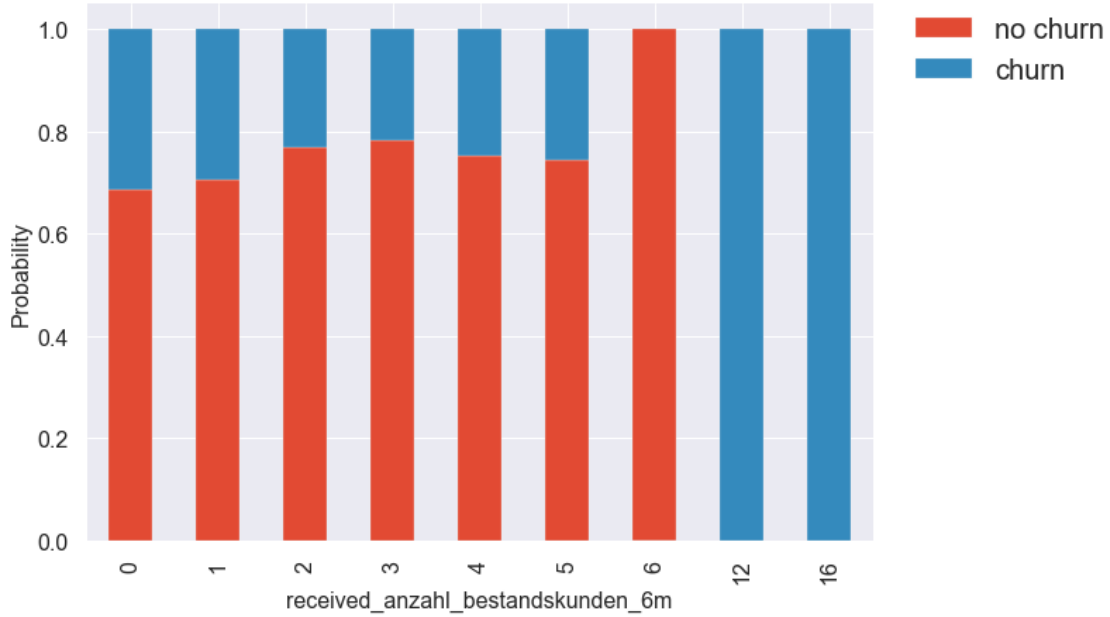

Clicked Bestandskunden (6m)
no churn
churn
Frequency
120000
100000
80000
60000
40000
20000
0
0
1
2
3
4
Quantity

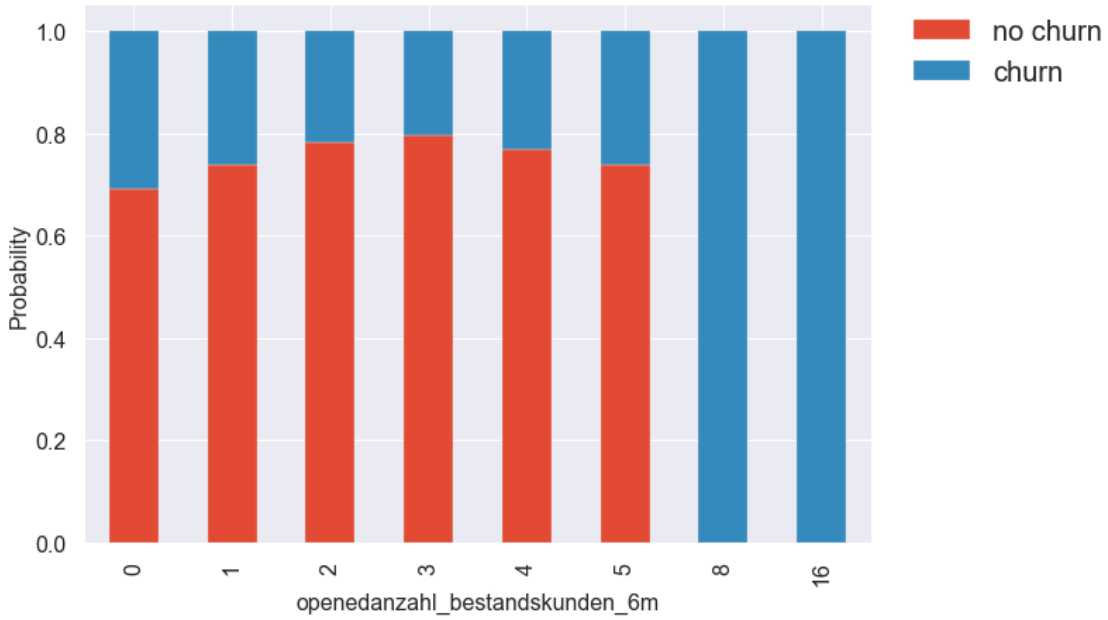

Unsubscribed Bestandskunden (6m)
no churn
churn
120000
100000
80000
60000
40000
20000
0
Frequency
0
1
Quantity

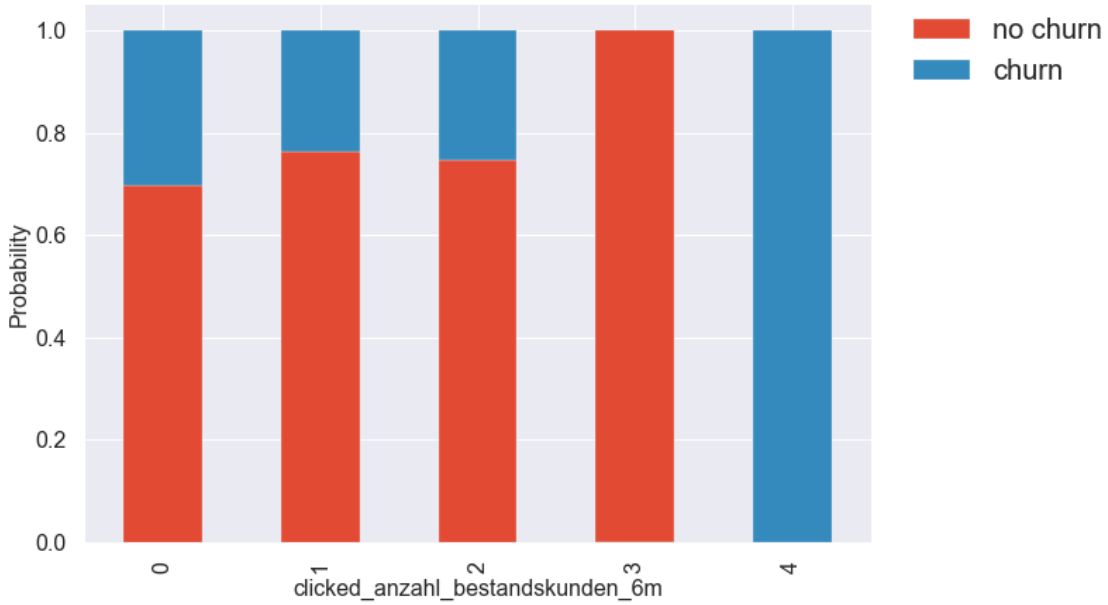

Clickrate Bestandskunden (3m)
no churn
churn
120000
100000
80000
60000
40000
20000
0
Frequency
0.0
0.25
0.33
0.5
0.67
1.0
Quantity

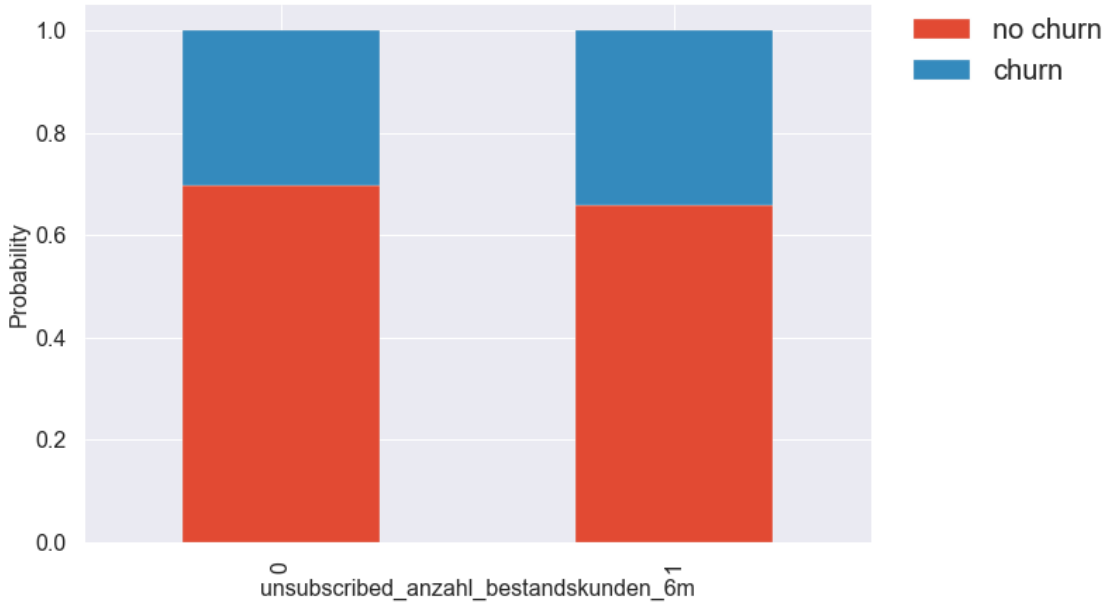


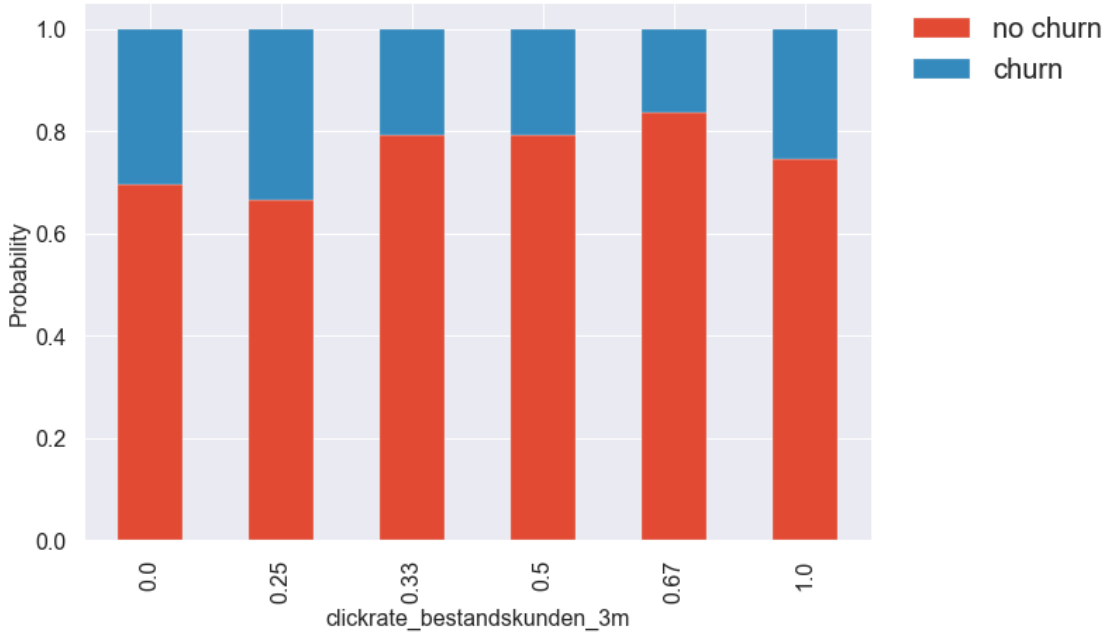
[217]:
```
for i in ['received_anzahl_bestandskunden_6m',␣
↪'openedanzahl_bestandskunden_6m', 'clicked_anzahl_bestandskunden_6m',␣
↪'unsubscribed_anzahl_bestandskunden_6m', 'clickrate_bestandskunden_3m',␣
↪'openrate_bestandskunden_3m']:

    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i)
```

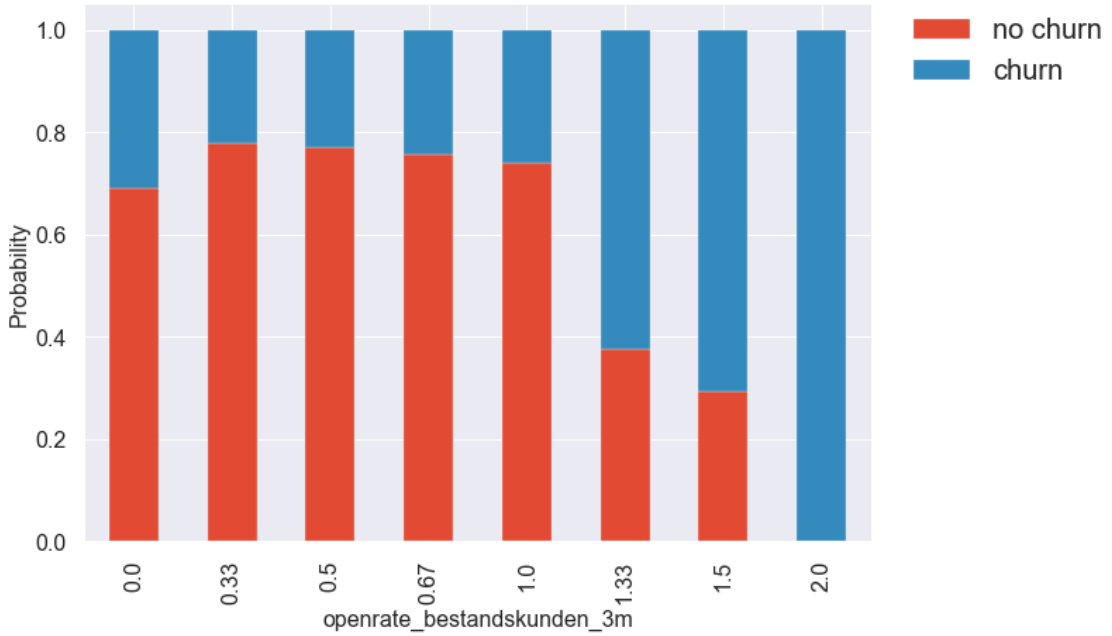



no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
8
16
openedanzahl_bestandskunden_6m


no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
clicked_anzahl_bestandskunden_6m


no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
unsubscribed_anzahl_bestandskunden_6m

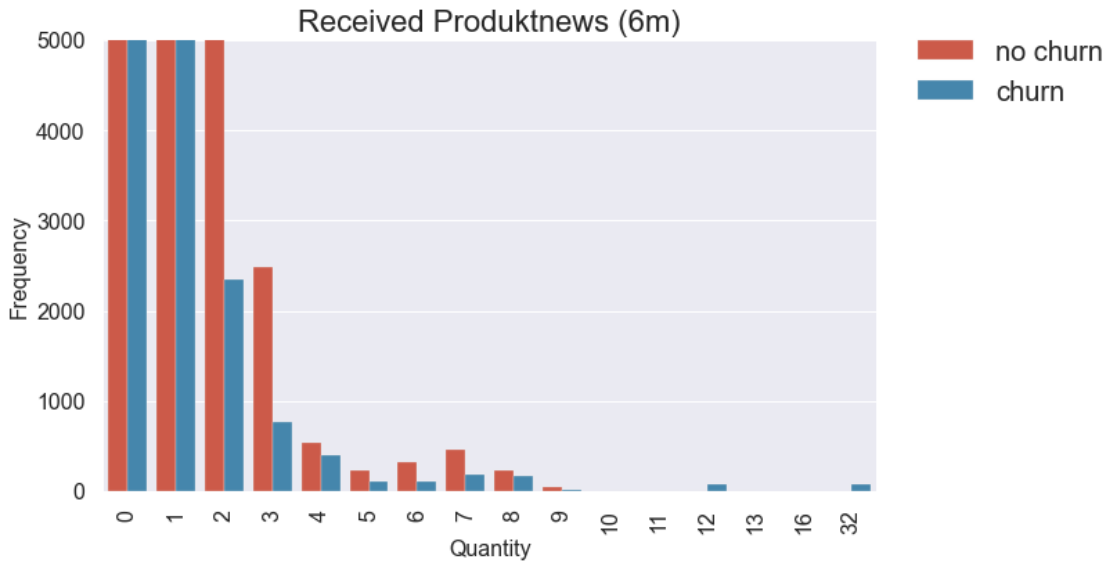

no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.25
0.33
0.5
0.67
1.0
clickrate_bestandskunden_3m

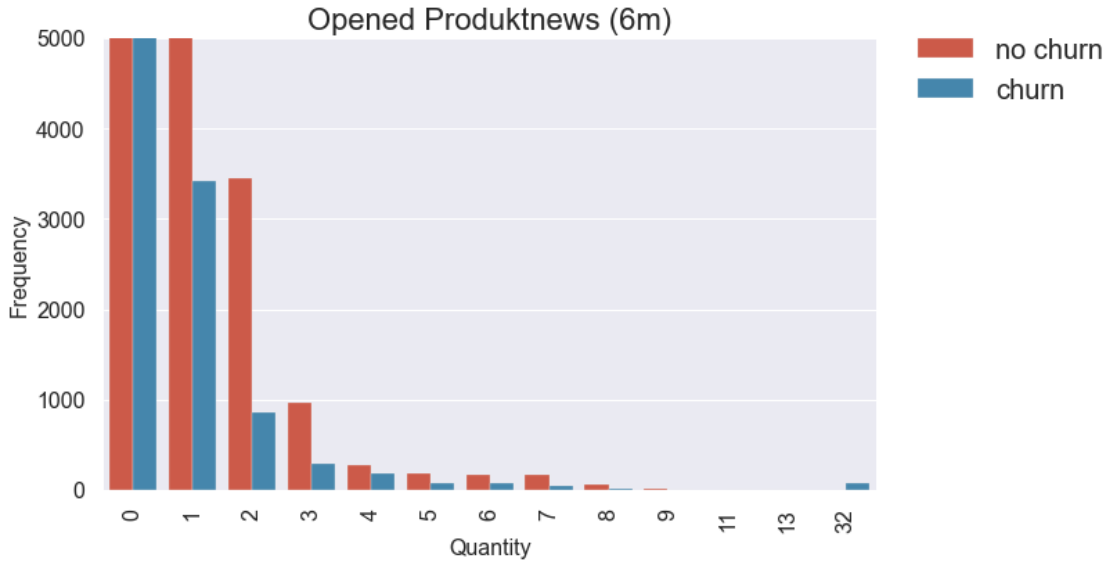


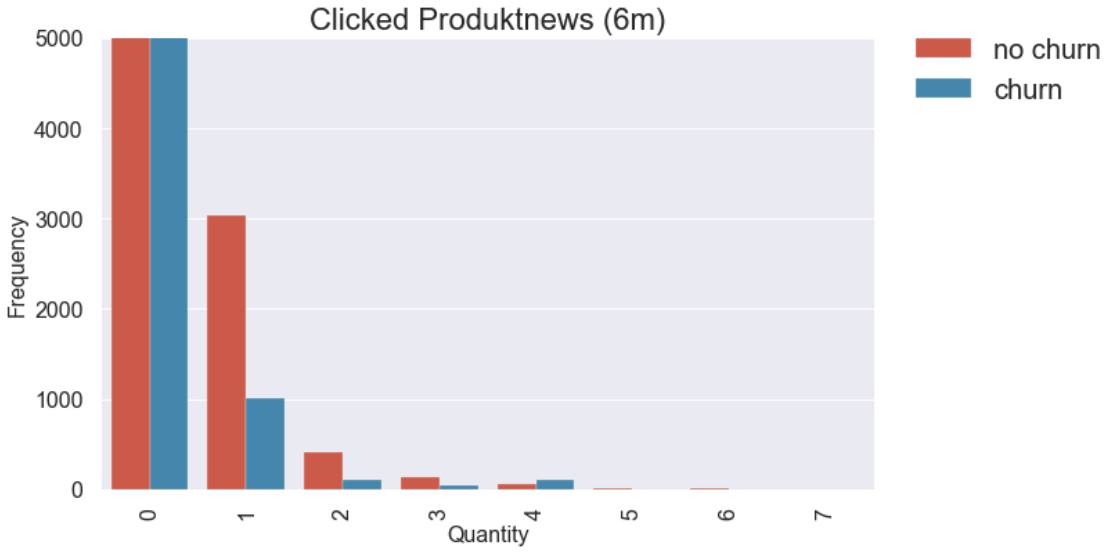
Produktnews

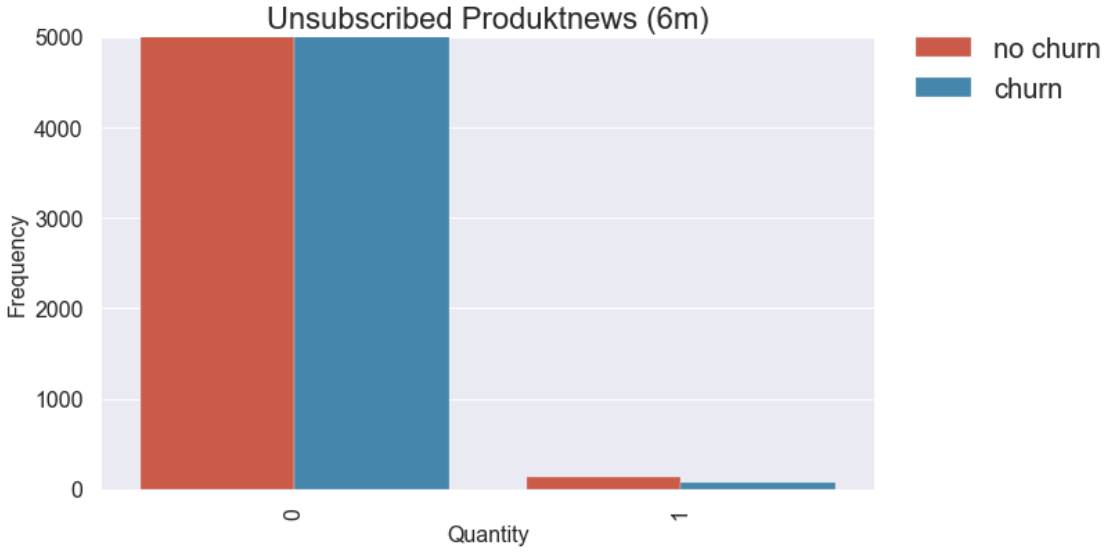
- with received, opened, clicked, unsubscribed mails
- time period = 6m

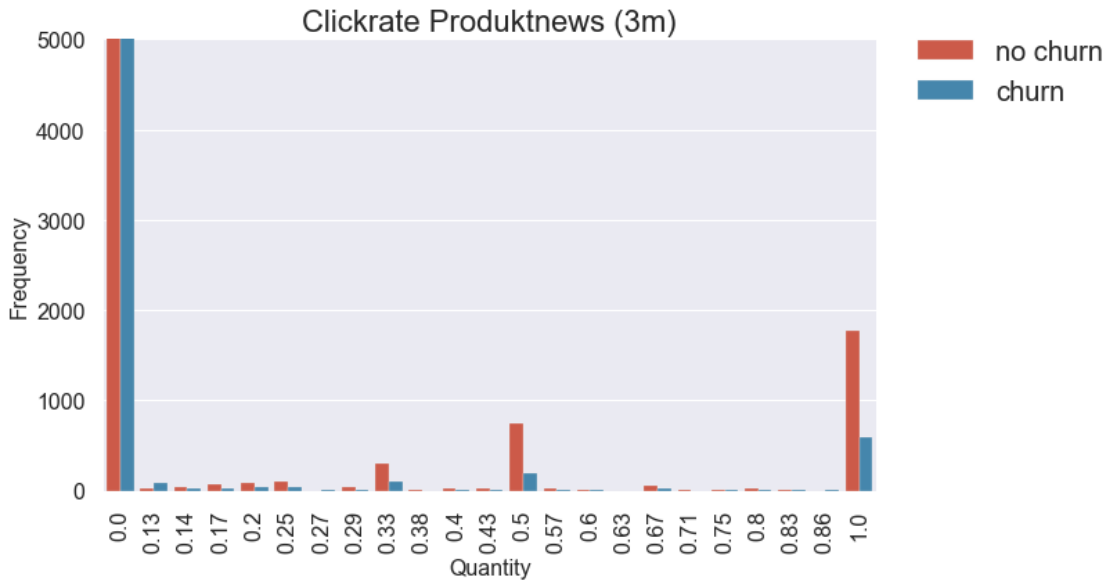
[218]:
```
title = ['Received Produktnews (6m)', 'Opened Produktnews (6m)', 'Clicked␣
 ↪Produktnews (6m)', 'Unsubscribed Produktnews (6m)', 'Clickrate Produktnews␣
 ↪(3m)', 'Openrate Produktnews (3m)']
labellist = ['no churn','churn']

for i, nl in enumerate(['received_anzahl_produktnews_6m',␣
 ↪'openedanzahl_produktnews_6m', 'clicked_anzahl_produktnews_6m',␣
 ↪'unsubscribed_anzahl_produktnews_6m', 'clickrate_produktnews_3m',␣
 ↪'openrate_produktnews_3m']):


    plt.subplots(figsize=(10,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    ax.set(ylim=(0, 5000))
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```

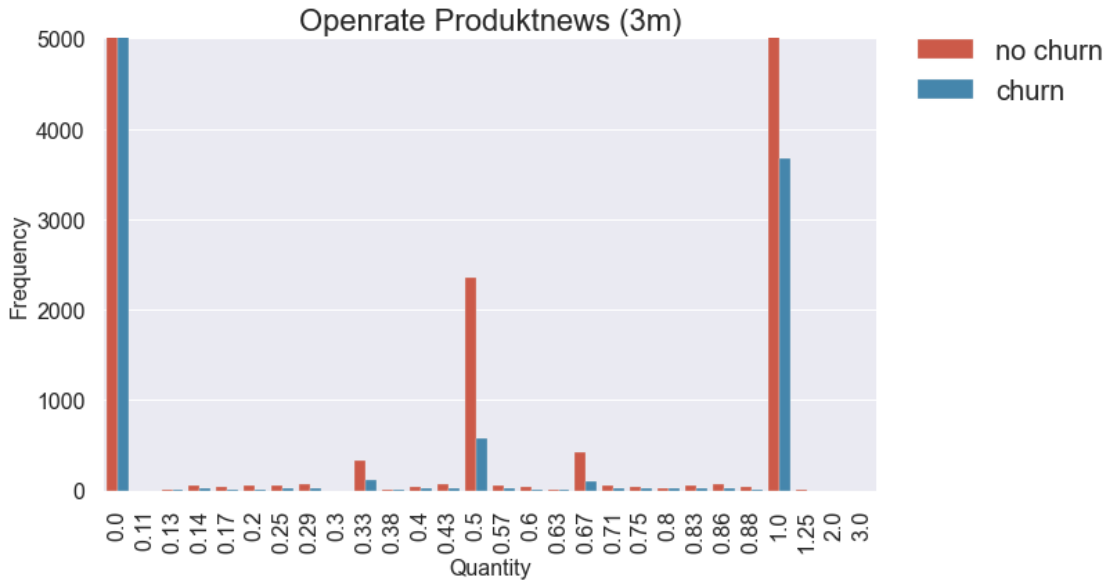

Received Produktnews (6m)
no churn
churn
Frequency
5000
4000
3000
2000
1000
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
16
32
Quantity


Opened Produktnews (6m)
no churn
churn
Frequency
5000
4000
3000
2000
1000
0
0
1
2
3
4
5
6
7
8
9
11
13
32
Quantity

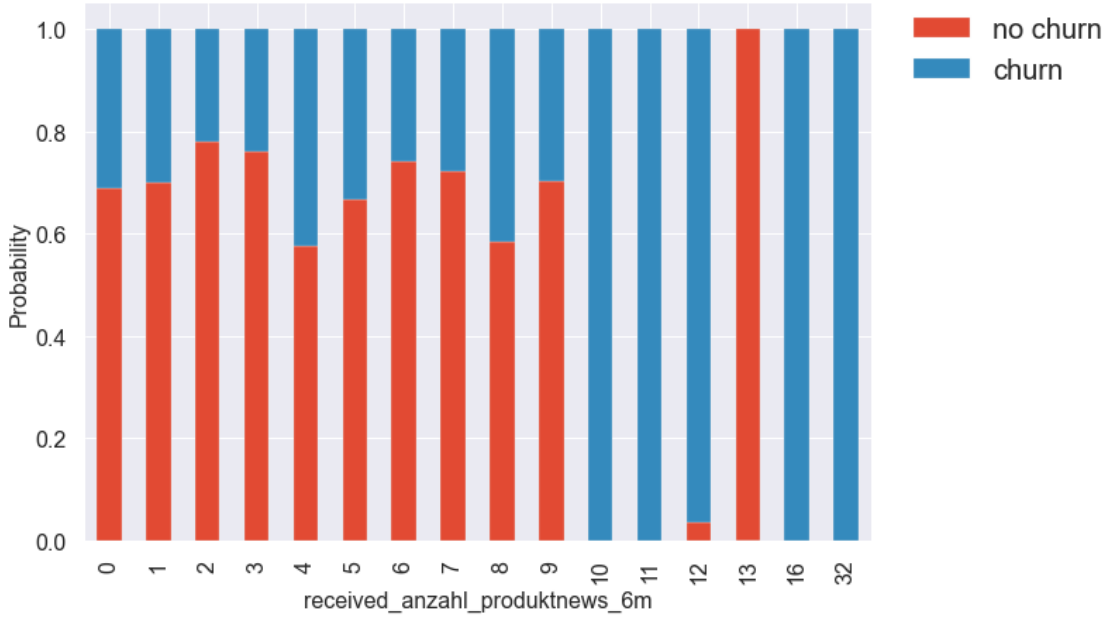

Clicked Produktnews (6m)
5000
4000
3000
2000
1000
0
Frequency
0
1
2
3
4
5
6
7
Quantity
no churn
churn

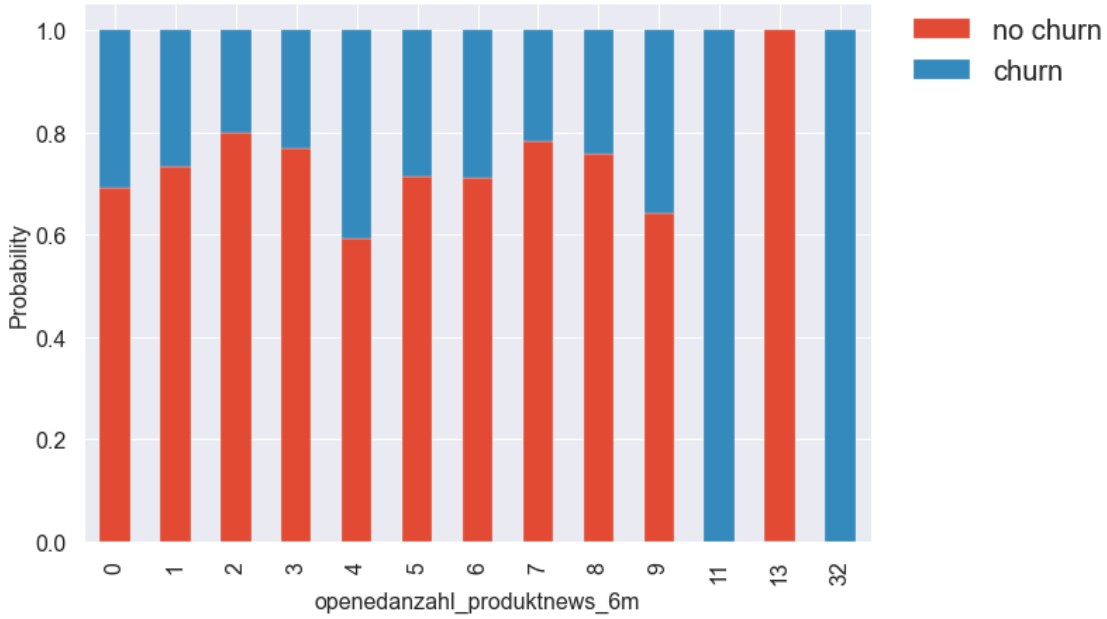

Unsubscribed Produktnews (6m)
5000
4000
3000
2000
1000
0
Frequency
0
1
Quantity
no churn
churn

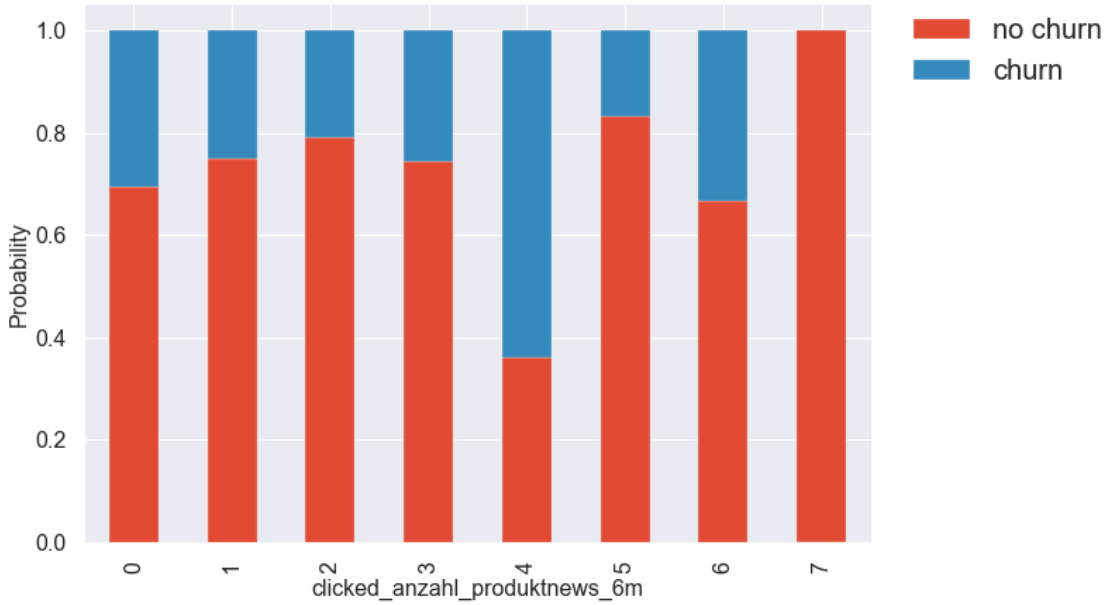


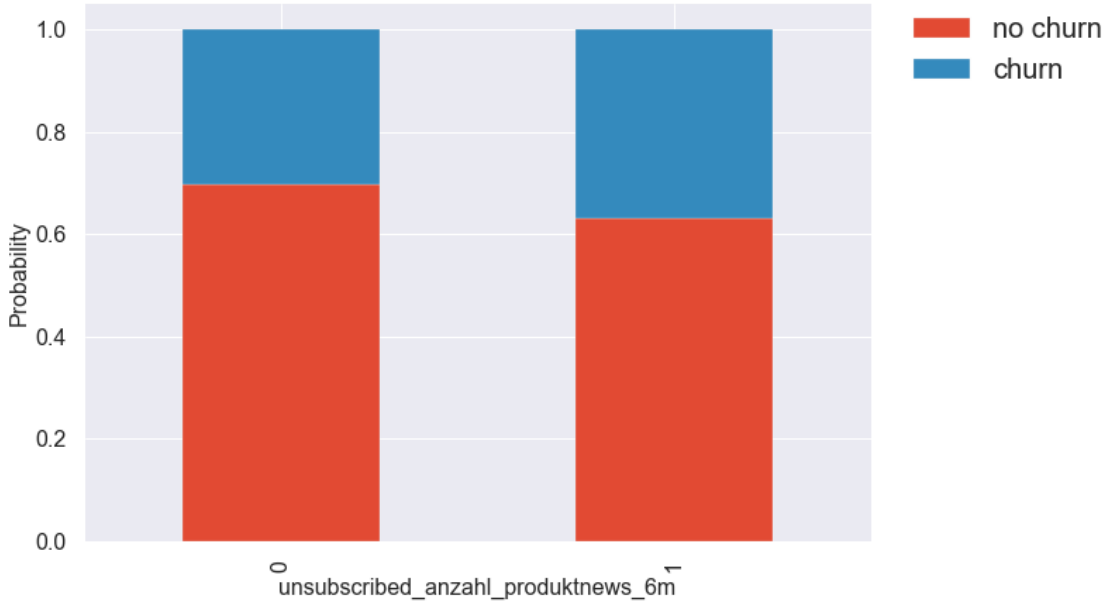


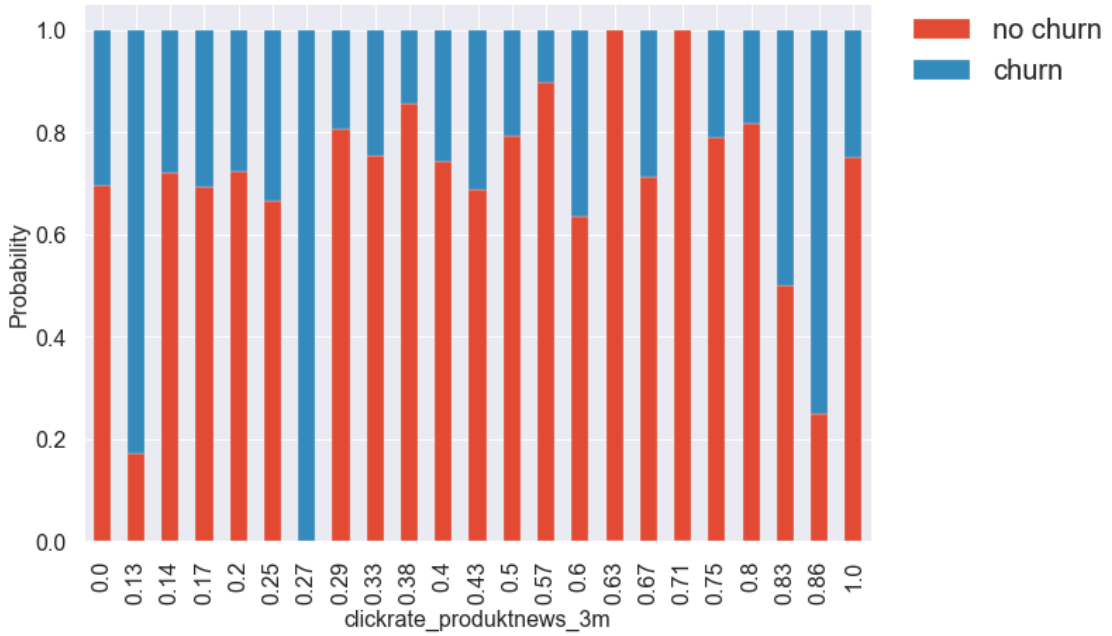
[219]:
```
for i in ['received_anzahl_produktnews_6m', 'openedanzahl_produktnews_6m',␣
↪'clicked_anzahl_produktnews_6m', 'unsubscribed_anzahl_produktnews_6m',␣
↪'clickrate_produktnews_3m', 'openrate_produktnews_3m']:
    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i)
```

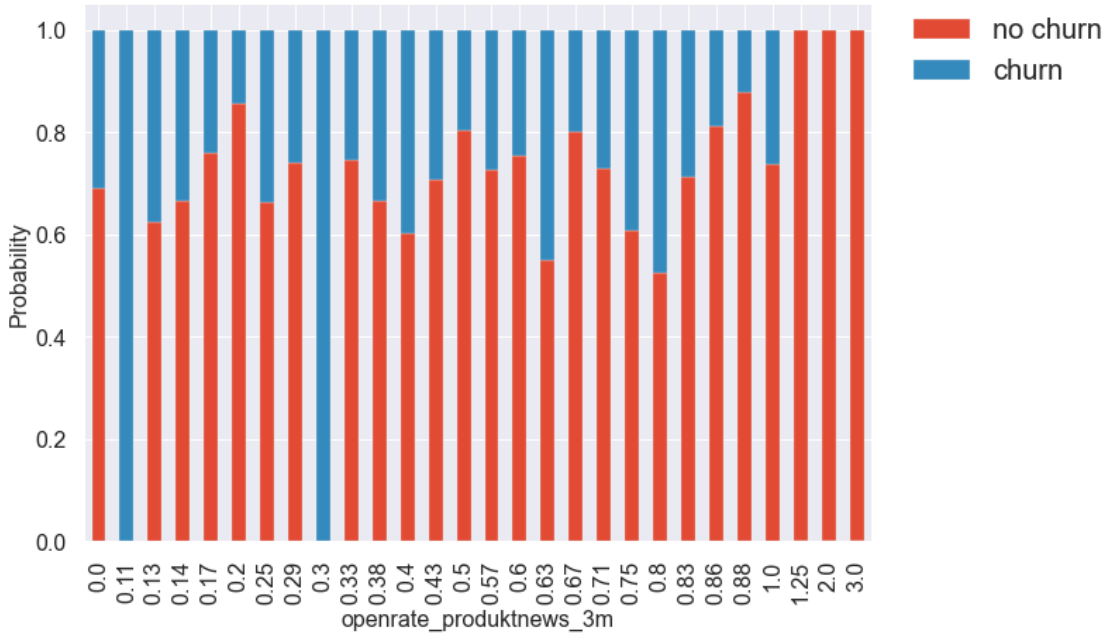

no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
16
32
received_anzahl_produktnews_6m


no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
7
8
9
11
13
32
openedanzahl_produktnews_6m


no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
7
clicked_anzahl_produktnews_6m


no churn
churn
Probability
1.0
0.8
0.6
0.4
0.2
0.0
0
1
unsubscribed_anzahl_produktnews_6m

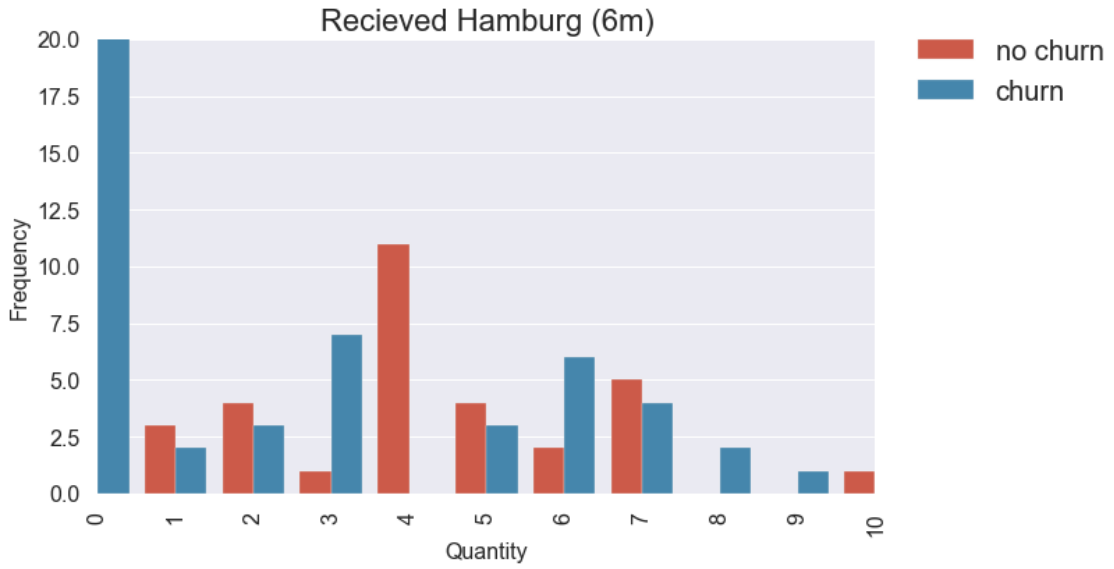




Hamburg Daily Newsletter (Mo-Fr) * with received, opened, clicked, unsubscribed mails * time period = 6m

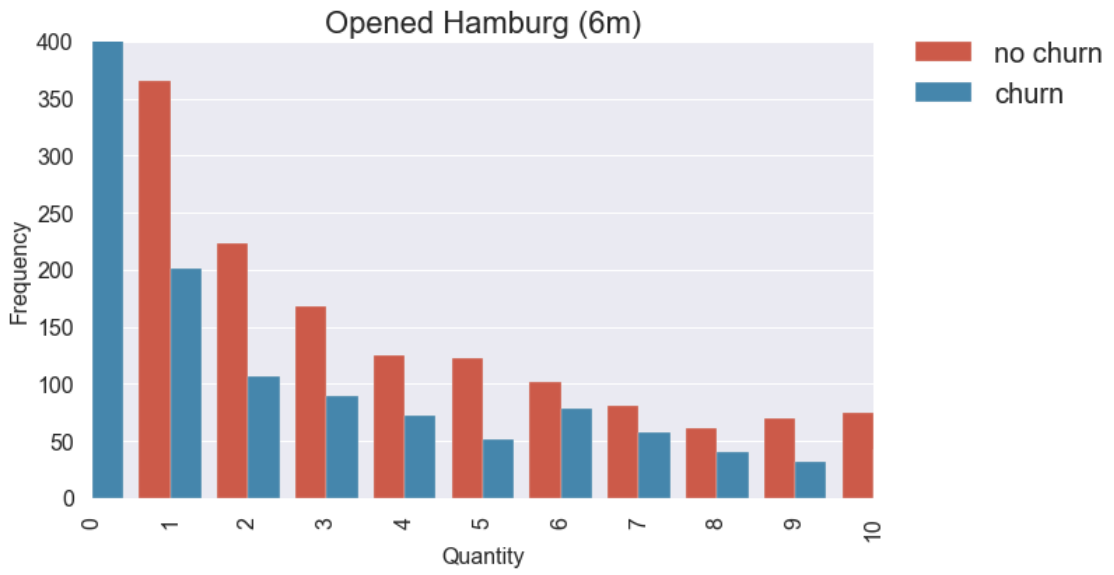
[220]:
```
plt.subplots(figsize=(10,6))
ax = sns.countplot(x='received_anzahl_hamburg_6m', hue='churn', data=df)
```

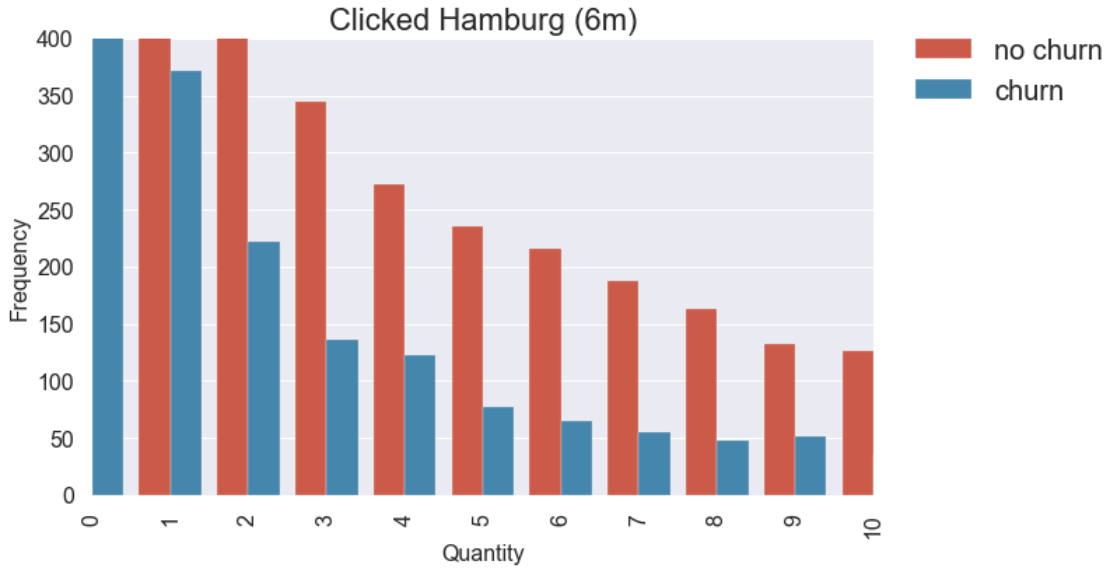
```
ax.set(xlabel='Quantity', ylabel='Frequency')
ax.set(ylim=(0, 20))
ax.set(xlim=(0, 10))
plt.title('Recieved Hamburg (6m)', fontsize=22)
L=plt.legend(fontsize=20,loc=(1.04,0.83))
L.get_texts()[0].set_text(labellist[0])
L.get_texts()[1].set_text(labellist[1])

ax.set_xticklabels(ax.get_xticklabels(), rotation=90);
```

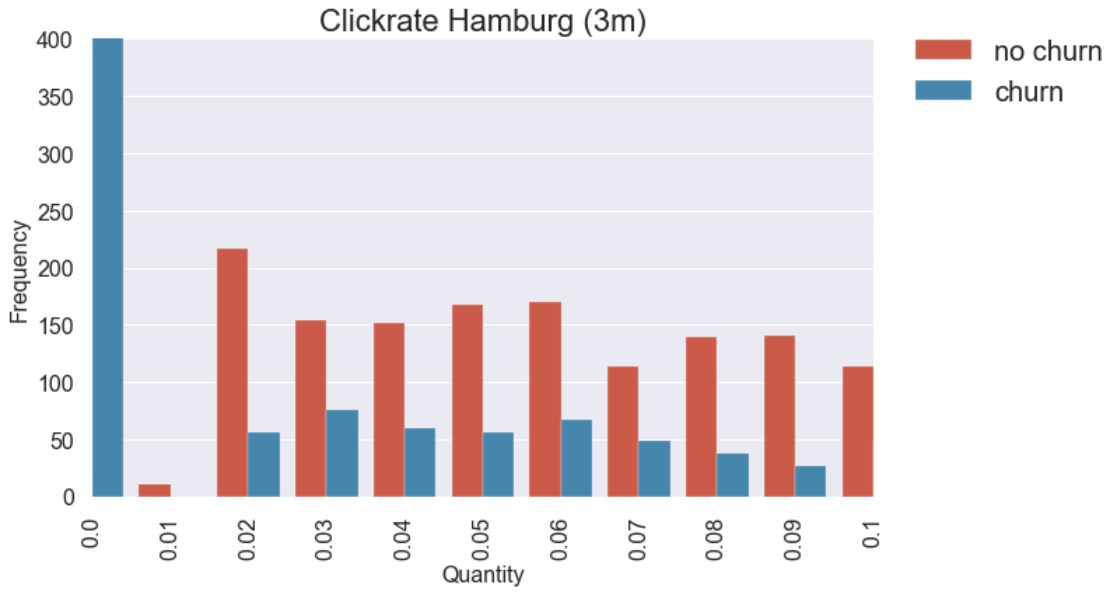


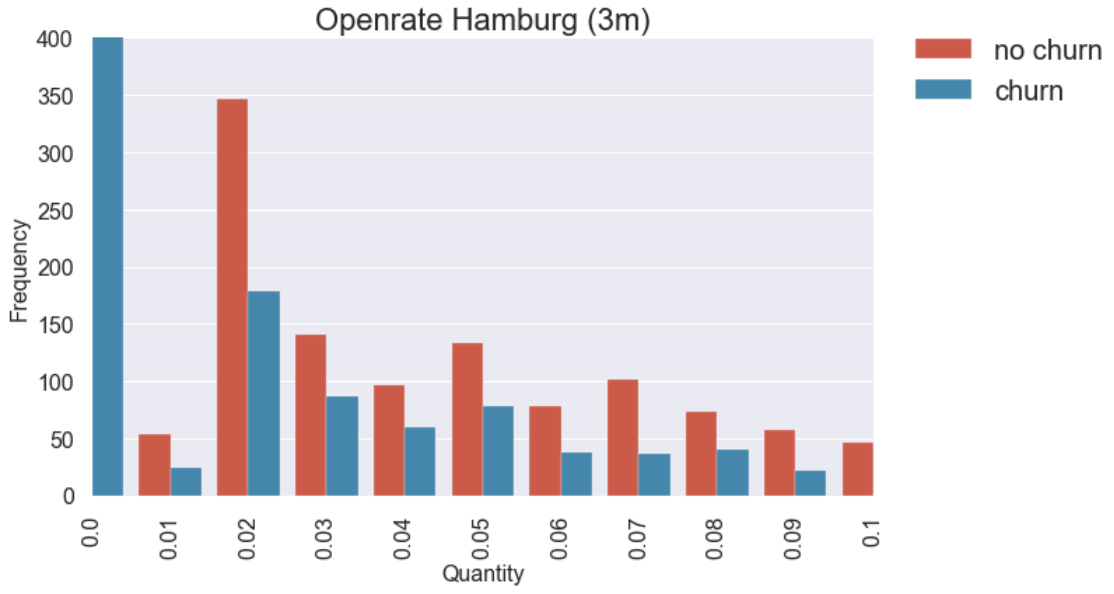
[221]:
```
title = ['Opened Hamburg (6m)', 'Clicked Hamburg (6m)', 'Clickrate Hamburg␣
↪(3m)', 'Openrate Hamburg (3m)']
labellist = ['no churn','churn']

for i, nl in enumerate(['openedanzahl_hamburg_6m', 'clicked_anzahl_hamburg_6m',␣
↪'clickrate_hamburg_3m', 'openrate_hamburg_3m']):


    plt.subplots(figsize=(10,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    ax.set(ylim=(0, 400))
    ax.set(xlim=(0, 10))
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])
```

```
    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```



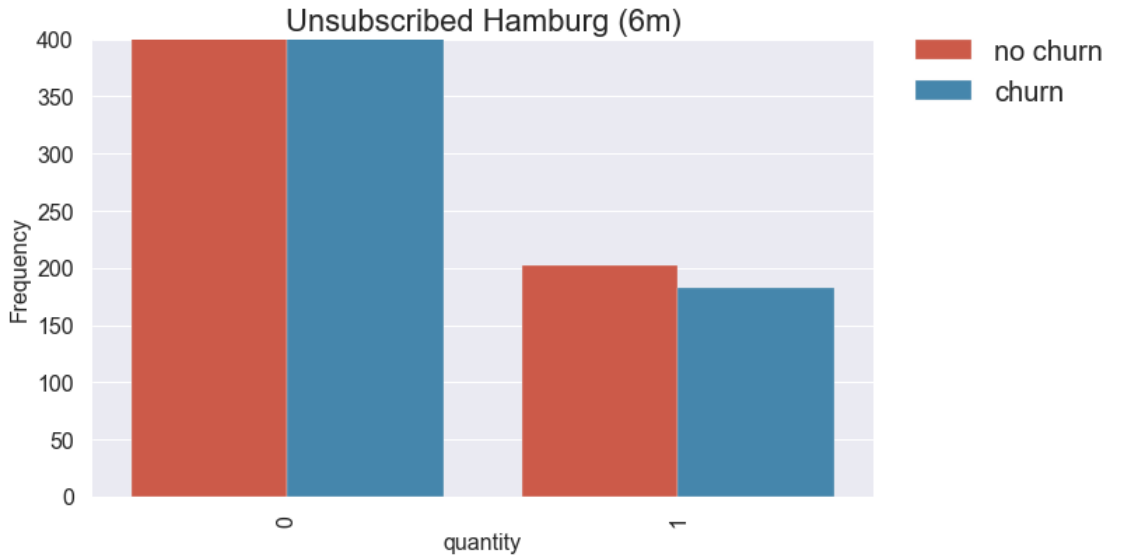




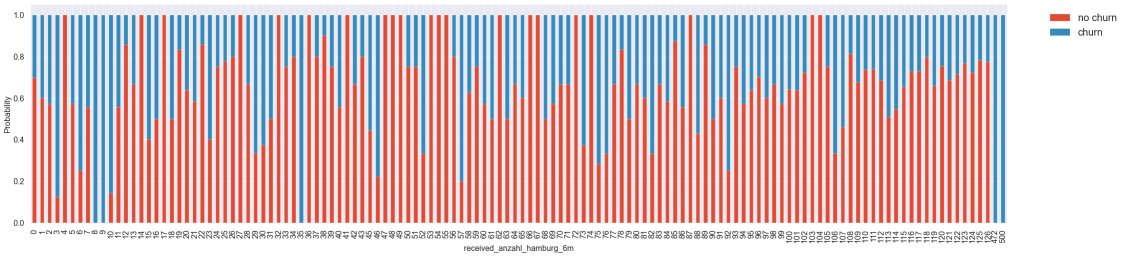


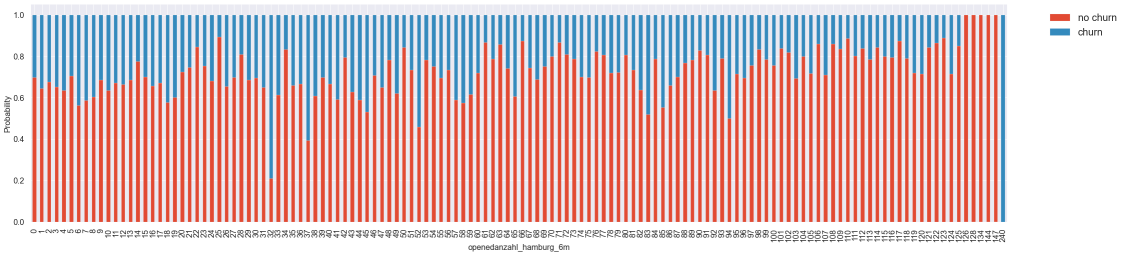
[222]:
```
plt.subplots(figsize=(10,6))
ax = sns.countplot(x='unsubscribed_anzahl_hamburg_6m', hue='churn', data=df)
ax.set(xlabel='quantity', ylabel='Frequency')
ax.set(ylim=(0, 400))
plt.title('Unsubscribed Hamburg (6m)', fontsize=22)
L=plt.legend(fontsize=20,loc=(1.04,0.83))
L.get_texts()[0].set_text(labellist[0])
L.get_texts()[1].set_text(labellist[1])
```

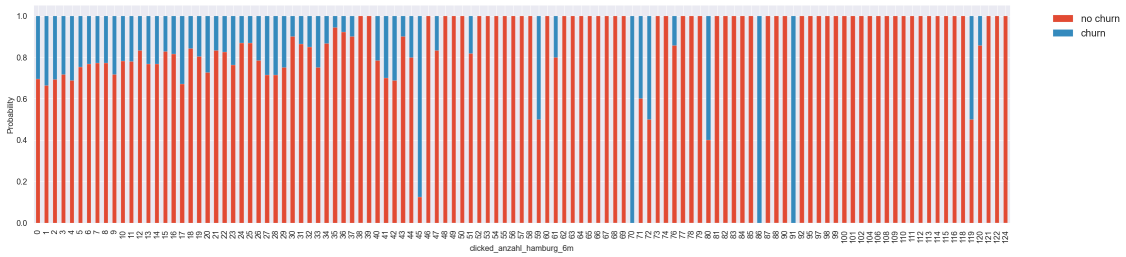
```
ax.set_xticklabels(ax.get_xticklabels(), rotation=90);
```

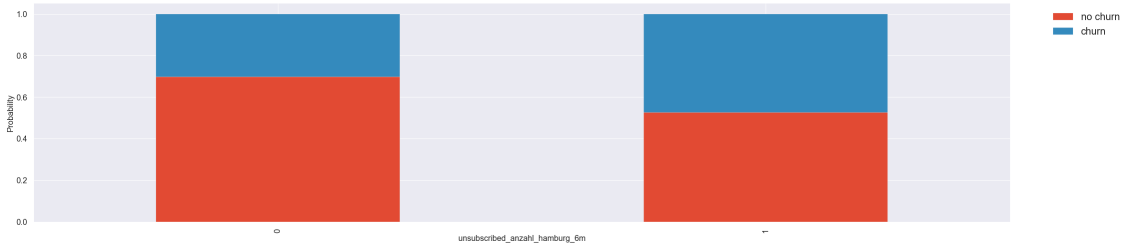


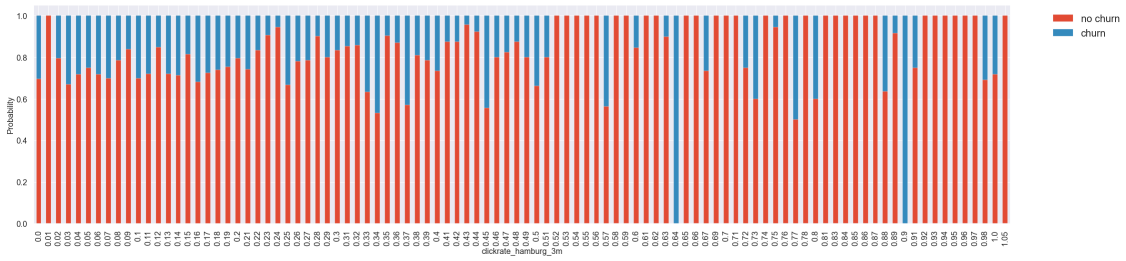
[226]:
```
for i in ['received_anzahl_hamburg_6m', 'openedanzahl_hamburg_6m',␣
↪'clicked_anzahl_hamburg_6m', 'unsubscribed_anzahl_hamburg_6m',␣
↪'clickrate_hamburg_3m', 'openrate_hamburg_3m']:
    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i, figsize_y=8,␣
↪figsize_x=35)
```

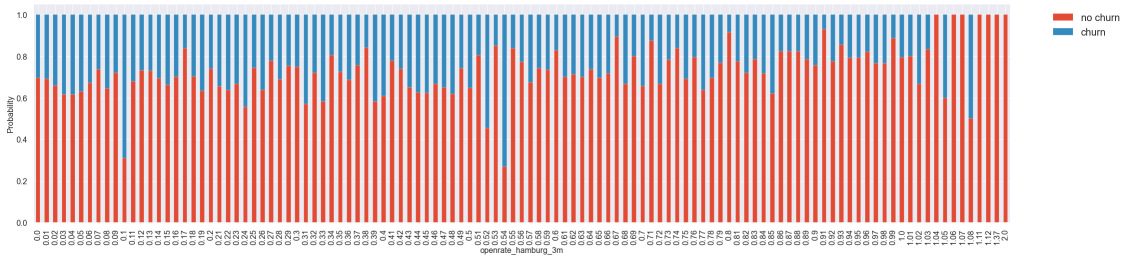








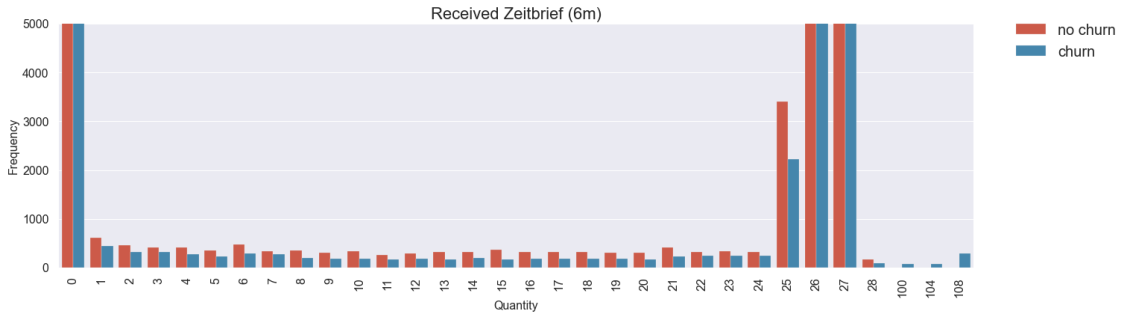


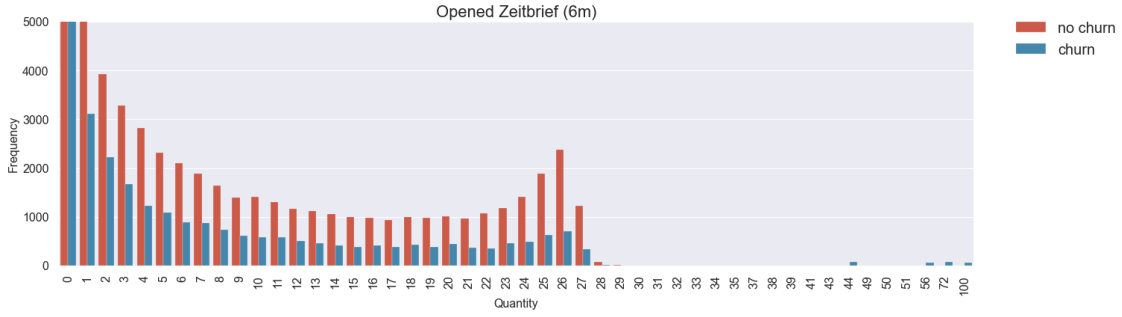


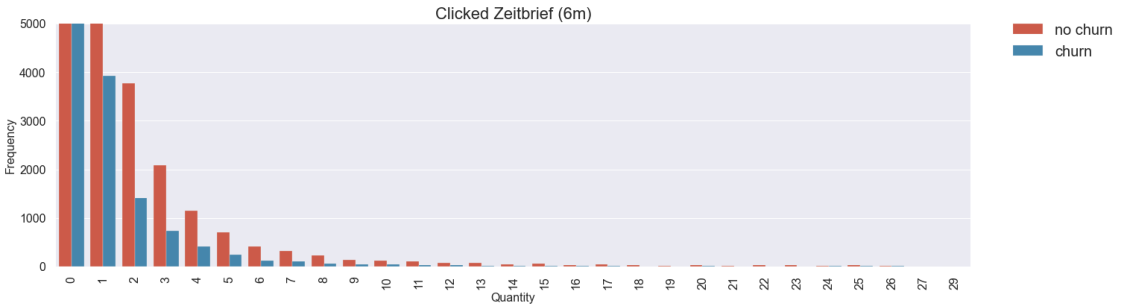
Zeitbrief

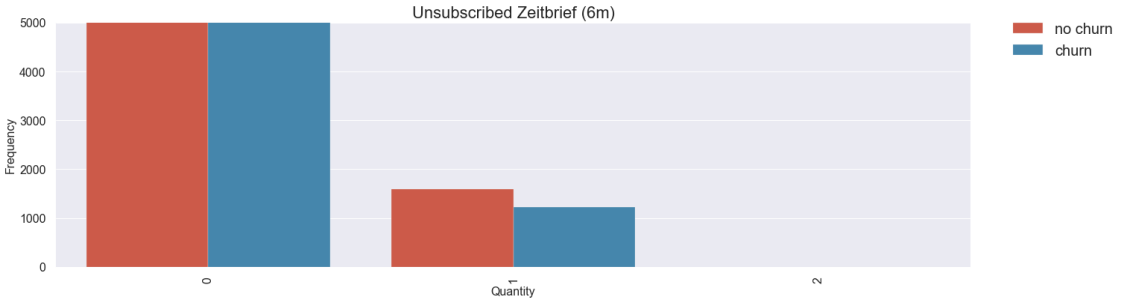
- with received, opened, clicked, unsubscribed mails
- time period = 6m

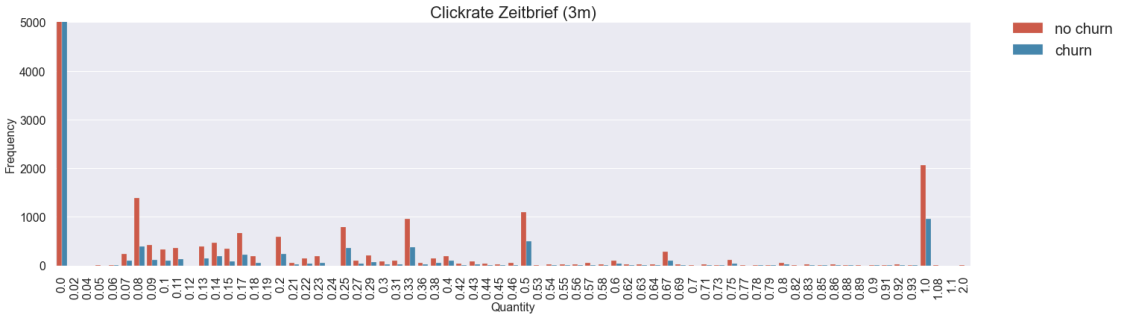
[228]:
```
title = ['Received Zeitbrief (6m)', 'Opened Zeitbrief (6m)', 'Clicked Zeitbrief␣
↪(6m)', 'Unsubscribed Zeitbrief (6m)', 'Clickrate Zeitbrief (3m)', 'Openrate␣
↪Zeitbrief (3m)']
labellist = ['no churn','churn']

for i, nl in enumerate(['received_anzahl_zeitbrief_6m',␣
↪'openedanzahl_zeitbrief_6m', 'clicked_anzahl_zeitbrief_6m',␣
↪'unsubscribed_anzahl_zeitbrief_6m', 'clickrate_zeitbrief_3m',␣
↪'openrate_zeitbrief_3m']):


    plt.subplots(figsize=(22,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    ax.set(ylim=(0, 5000))
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```

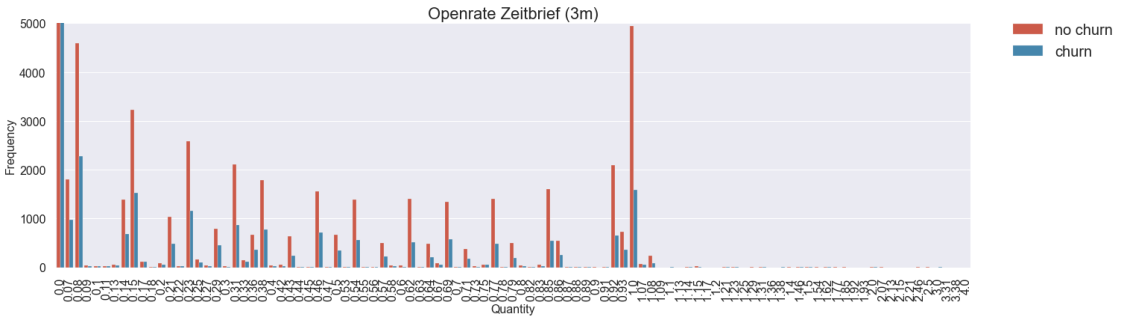




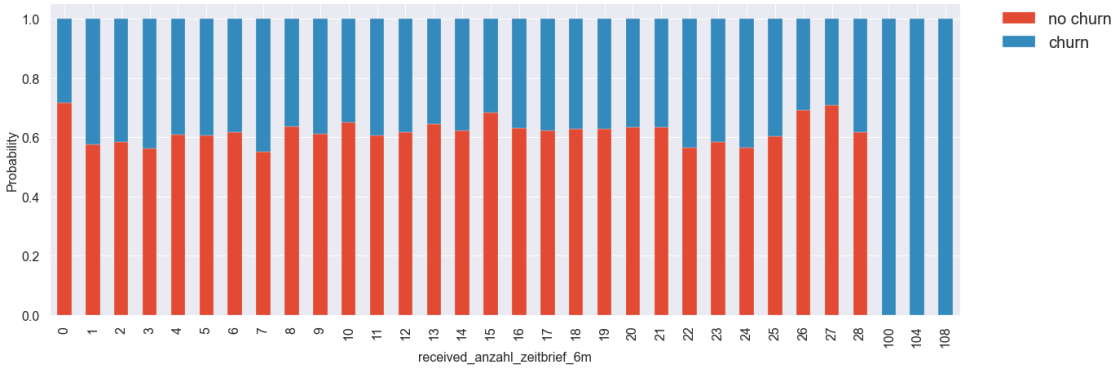

Clicked Zeitbrief (6m)
no churn
churn
Frequency
Quantity

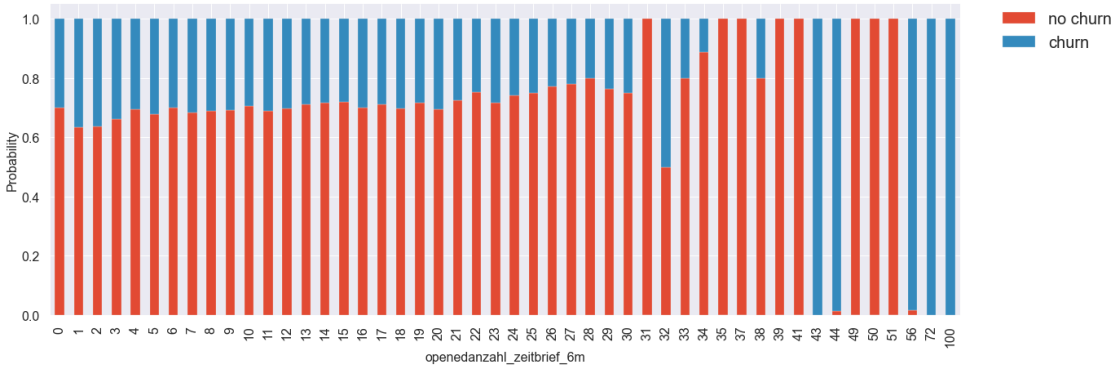

Unsubscribed Zeitbrief (6m)
no churn
churn
Frequency
Quantity

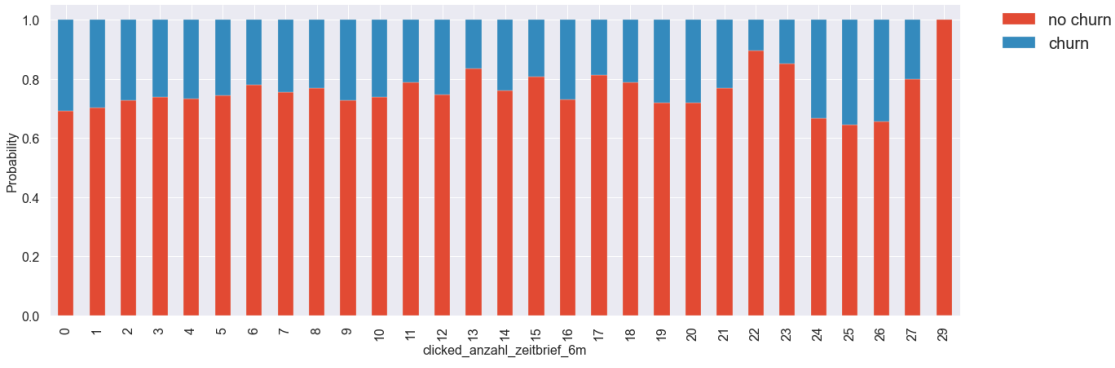

Clickrate Zeitbrief (3m)
no churn
churn
Frequency
Quantity

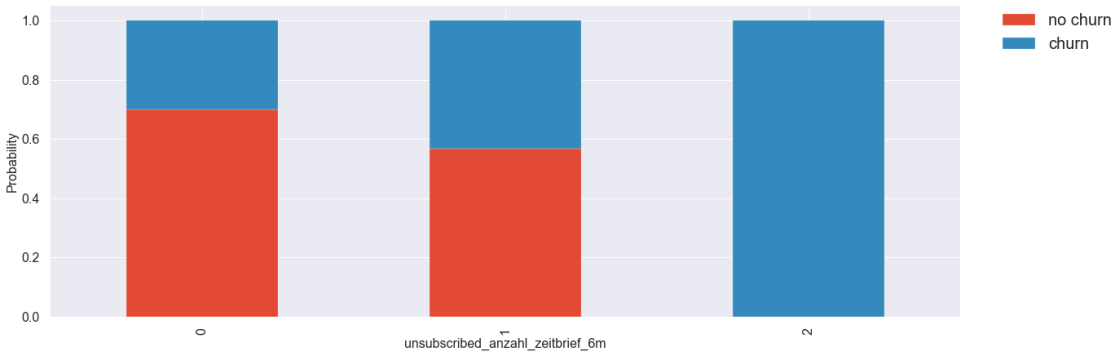

Openrate Zeitbrief (3m)
no churn
churn
Frequency
Quantity

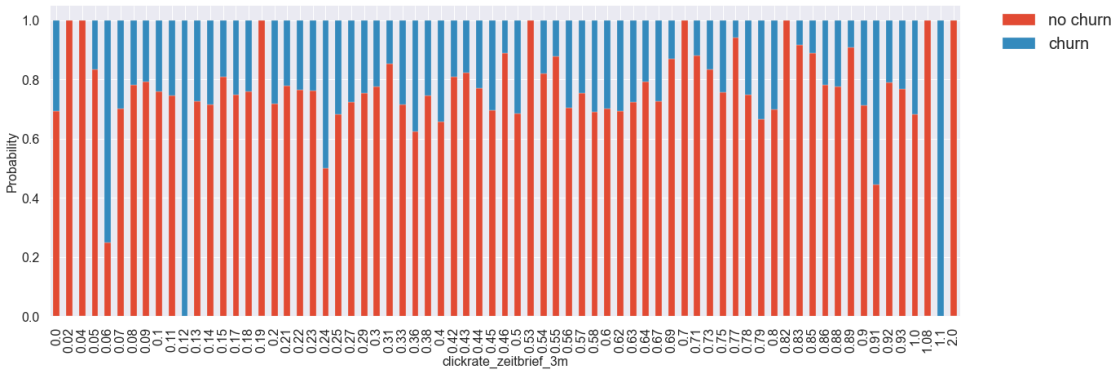
[231]:
```
for i in ['received_anzahl_zeitbrief_6m', 'openedanzahl_zeitbrief_6m',␣
↪'clicked_anzahl_zeitbrief_6m', 'unsubscribed_anzahl_zeitbrief_6m',␣
↪'clickrate_zeitbrief_3m', 'openrate_zeitbrief_3m']:
    x = crosstab_evaluation(df[i],df.churn)
    crosstab_barplot(x,['no churn','churn'],xlabelname=i,figsize_x=20)
```

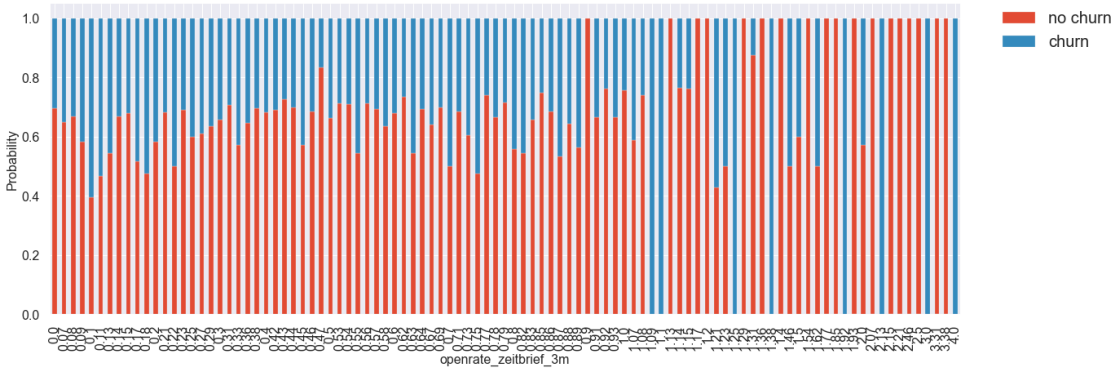










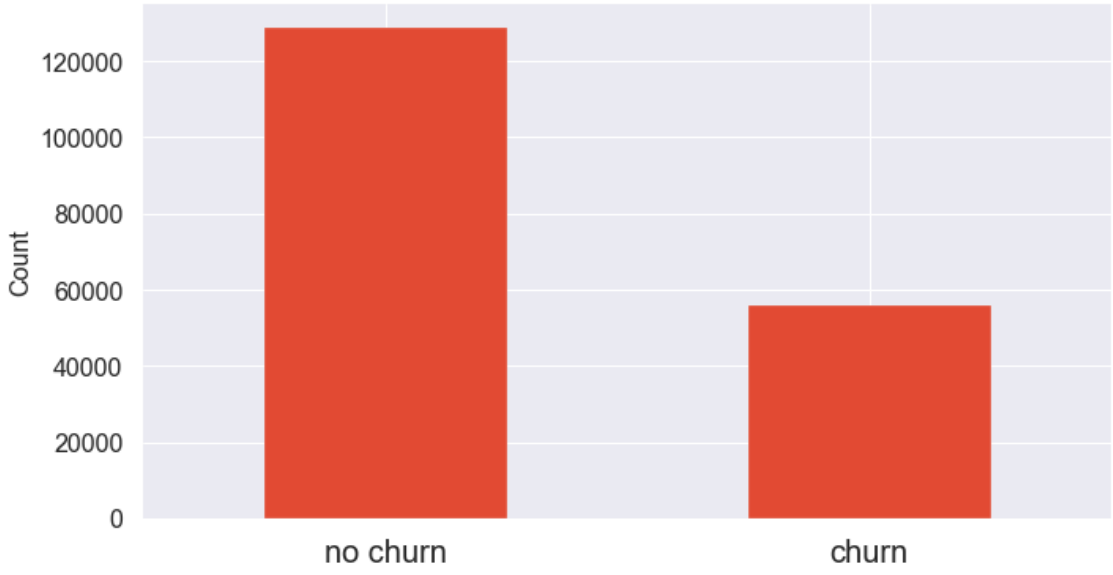


5.6 Target Variable Analysis

Our target variable is “churn”, which is 1 - churn or 0 - no churn.

[232]:
```
## Show the imbalance of churn target
df.churn.value_counts().plot(kind='bar',figsize=(11,6));
plt.ylabel('Count');
```

```
#plt.xlabel('Churn')
plt.xticks(np.arange(2), ('no churn', 'churn'),fontsize=20,rotation=0);
```



[233]:
```
df.churn.value_counts()/df.shape[0]
```

[233]:
```
0    0.696848
1    0.303152
Name: churn, dtype: float64
```

6 Feature Engineering

6.1 zon features

[234]:
```
len(df_zon.columns)
```

[234]:
```
14
```

The 14 features in this list will be transformed. We will use the information only if the registration is completed (value = 2) and than we will aggregate them to get a continues feature "sum_zon" that keeps the information how many users registered for the zones and how many.

[235]:
```
def flatten_greater_1(flat):
    if flat > 1:
        return 1
    else:
        return 0
```

[236]:
```
print("Sum zon features")
for i in df_zon:
    df[i] = df[i].apply(flatten_greater_1)
sum_zon = df_zon.sum(axis=1)
print(sum_zon.sample(5))
print("there are a few customers registred or active in more than one")
```

```
Sum zon features
46467     0
202958    1
180101    2
74716     0
12175     0
dtype: int64
there are a few customers registred or active in more than one
```

[237]:
```
sum_zon = sum_zon.to_frame(name="sum_zon")
sum_zon.head(1)
```

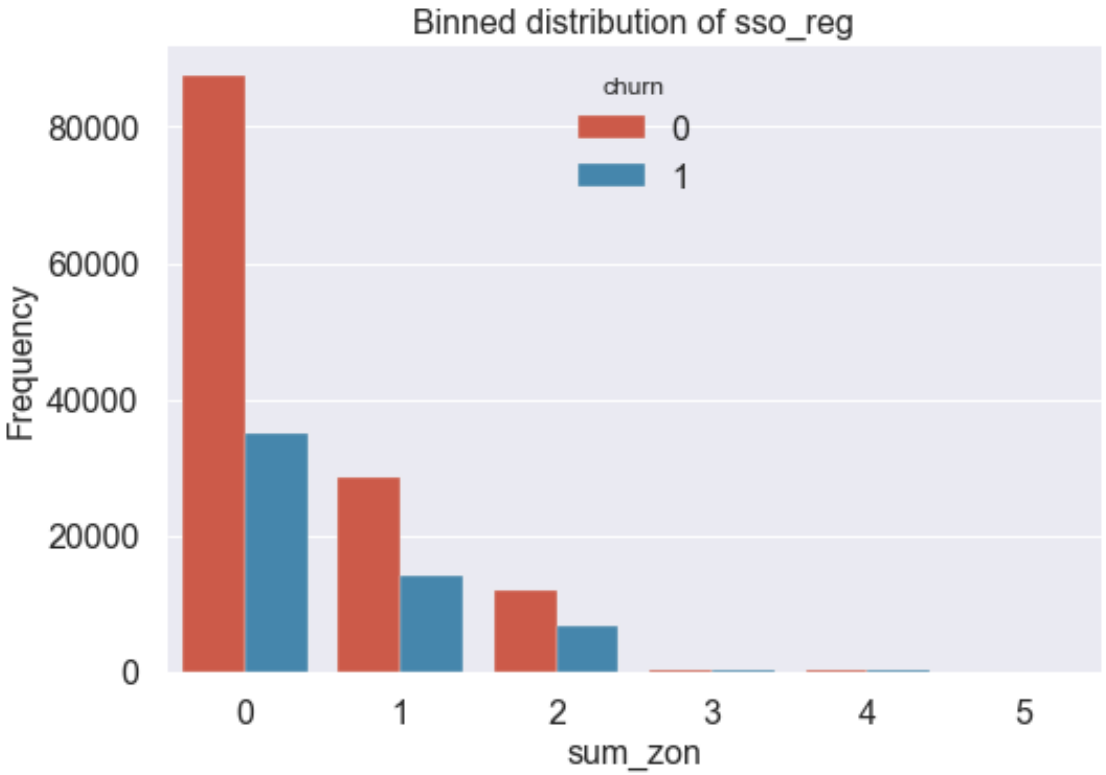
[237]:

	sum_zon
0	0

[238]:
```
df = df.join(sum_zon)
```

[239]:
```
print(df.sum_zon.value_counts())
ax = sns.countplot(x=df.sum_zon, data=df, hue="churn")
ax.set(ylabel="Frequency", xlim=[-0.5, 5.5])
plt.title(f"Binned distribution of {elem}")
plt.show()
```

```
0    122537
1    42569
2    18697
3    598
4    252
5    7
Name: sum_zon, dtype: int64
```



6.2 reg features

[240]:
```
len(df_reg.columns)
```

[240]:
```
4
```

[241]:
```
eda.meta(df_reg)
```

[241]:

varname	boa_reg	che_reg	sit_reg	sso_reg
nulls	0	0	0	0
percent	0	0	0	0
dtype	int64	int64	int64	int64
dup	True	True	True	True
nuniques	2	2	2	2

The 4 features in this list will be aggregated to a feature "sum_reg" where we will find the sum of how many registered areas the user is registered to.

[242]:
```
print("Sum registrated for special areas features")
sum_reg = df_reg.sum(axis=1)
print(sum_reg.sample(10))
print("there are a few customers registred or active in more than one")
```

```
Sum registrated for special areas features
43149     3
145281    1
77065     1
138563    1
109270    1
128832    1
12923     0
195232    2
63449     3
80899     2
dtype: int64
there are a few customers registred or active in more than one
```

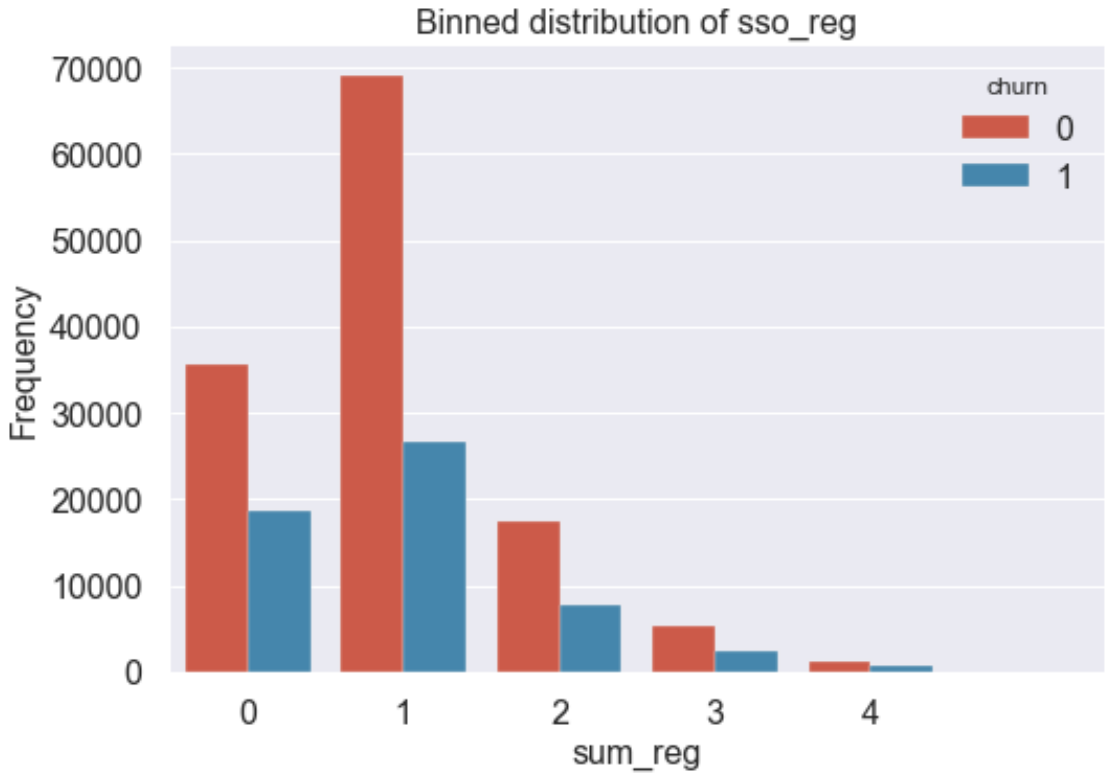
[243]:
```
sum_reg = sum_reg.to_frame(name="sum_reg")
sum_reg.head(1)
```

[243]:
```
   sum_reg
0  1
```

[244]:
```
df = df.join(sum_reg)
```

[245]:
```
print(df.sum_reg.value_counts())
ax = sns.countplot(x=df.sum_reg, data=df, hue="churn")
ax.set(ylabel="Frequency", xlim=[-0.5, 5.5])
plt.title(f"Binned distribution of {elem}")
plt.show()
```

```
1    95768
0    54285
2    25134
3    7655
4    1818
Name: sum_reg, dtype: int64
```



6.3 Newsletter Flattening

6.3.1 Newsletter Email

Subscribers which are subscribed to an Newsletter online will be send emails. The number depends on the frequency of delivery of the newsletter. The Newsletter *Hamburg* and the Newsletter *Zeitbiref* are for example daily (Mo-Fr) Newsletter. For the purpose of our analysis it is only important if the subscriber becomes Email-Newsletter or not. Therefore, the following section is going to flatten those columns into 1 and 0. * df_nl_bestandskunden * df_nl_produktnews * df_nl_hamburg * df_nl_zeitbrief

- received
- opened quantity
- clicked quantity
- unsubscribed

[246]:
```
def flatten_greater_0(flat):
    if flat > 0:
        return 1
    else:
        return 0
```

[247]:
```
df.rename(columns={'openedanzahl_bestandskunden_6m':␣
↪'opened_anzahl_bestandskunden_6m',
                   'openedanzahl_produktnews_6m':␣
↪'opened_anzahl_produktnews_6m',
                   'openedanzahl_hamburg_6m': 'opened_anzahl_hamburg_6m',
                   'openedanzahl_zeitbrief_6m': 'opened_anzahl_zeitbrief_6m'},␣
↪inplace=True)
```

[248]:
```
df_nl_bestandskunden_1 = df.iloc[:, 77:93]  # newsletter existing customers␣
↪without rates
df_nl_produktnews_1 = df.iloc[:, 99:115]    # productnews (kind of newsletter␣
↪but more commercial)without rates
df_nl_hamburg_1 = df.iloc[:, 121:137]       # newsletter region hamburg␣
↪without rates
df_zb_1 = df.iloc[:, 143:159]               # newsletter zeitbrief without␣
↪rates
```

[249]:
```
for i in df_nl_bestandskunden_1:
    df[i] = df[i].apply(flatten_greater_0)

for i in df_nl_produktnews_1:
    df[i] = df[i].apply(flatten_greater_0)

for i in df_nl_hamburg_1:
    df[i] = df[i].apply(flatten_greater_0)

for i in df_zb_1:
    df[i] = df[i].apply(flatten_greater_0)
```

Typos in the colum names corrected

Furthermore we can observe the same pattern over all the different Newsletter: Only the ones which are not subscribed to an newsletter churn. Therefore the aggregation of the Newsletter is being considered.

[250]:
```
name = ['received_anzahl', 'opened_anzahl', 'clicked_anzahl',␣
↪'unsubscribed_anzahl']
art = ['bestandskunden','produktnews','hamburg','zeitbrief']
zeitraum = ['1w', '1m', '3m', '6m']
titel = ['nl_received_1w', 'nl_received_1m', 'nl_received_3m',␣
↪'nl_received_6m', 'nl_opened_1w', 'nl_opened_1m', 'nl_opened_3m',
       'nl_opened_6m','nl_clicked_1w', 'nl_clicked_1m', 'nl_clicked_3m',␣
↪'nl_clicked_6m', 'nl_unsubscribed_1w', 'nl_unsubscribed_1m',
       'nl_unsubscribed_3m', 'nl_unsubscribed_6m']
```

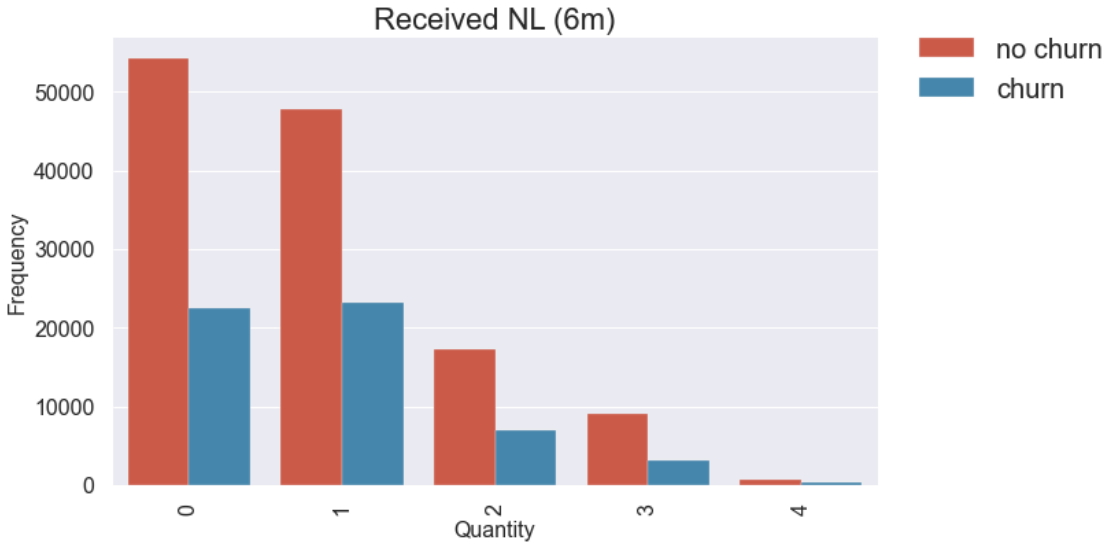
[251]:
```
links = []
for n in name:
```

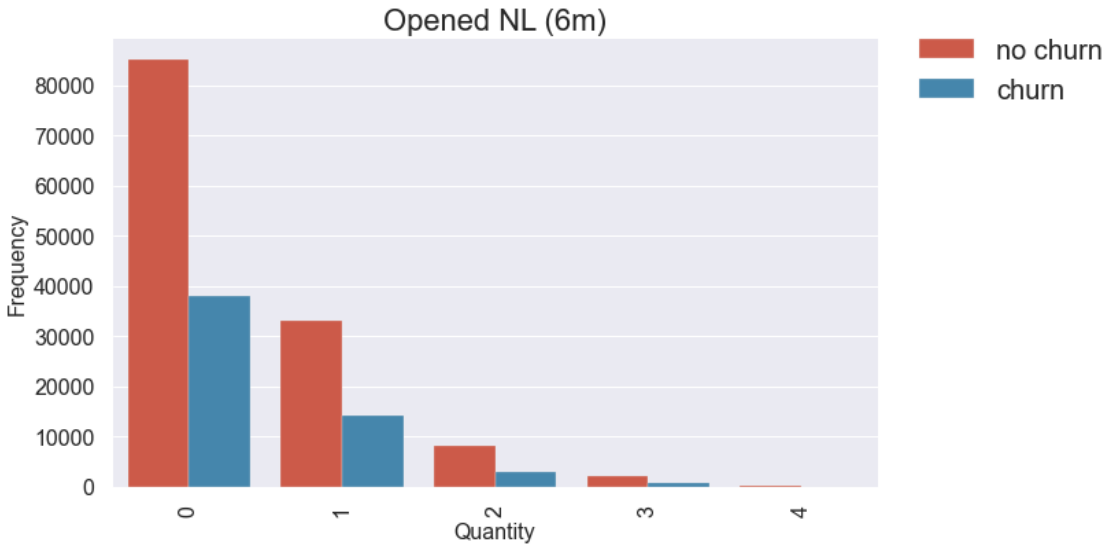
```
    for z in zeitraum:
        for a in art:
            links.append(n + '_' + a + '_' + z)
```

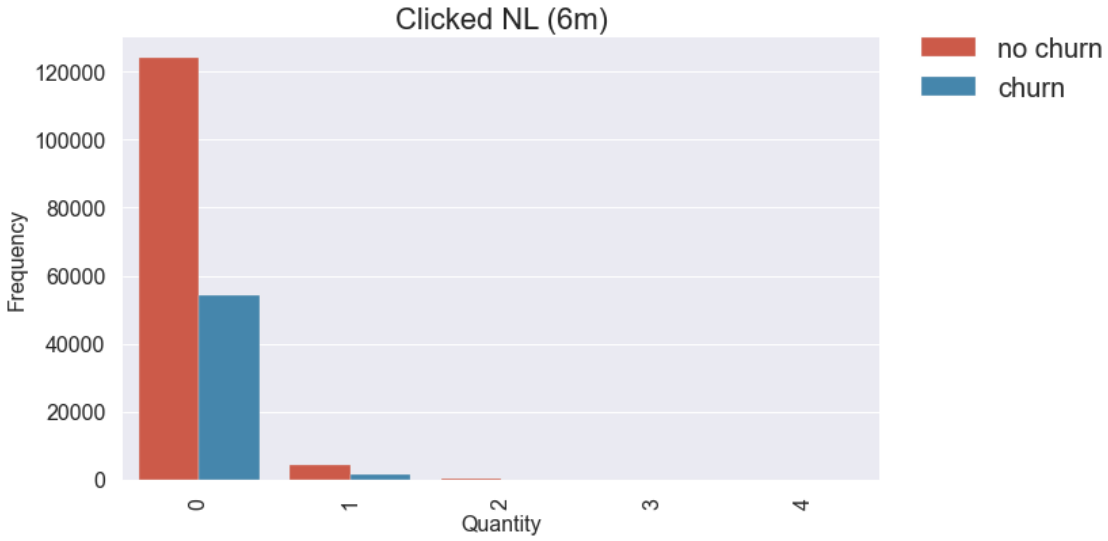
[252]:
```
for t in titel:
    df[t] = df[links[0]] + df[links[1]] + df[links[2]] + df[links[3]]
    links = links[3:]
```

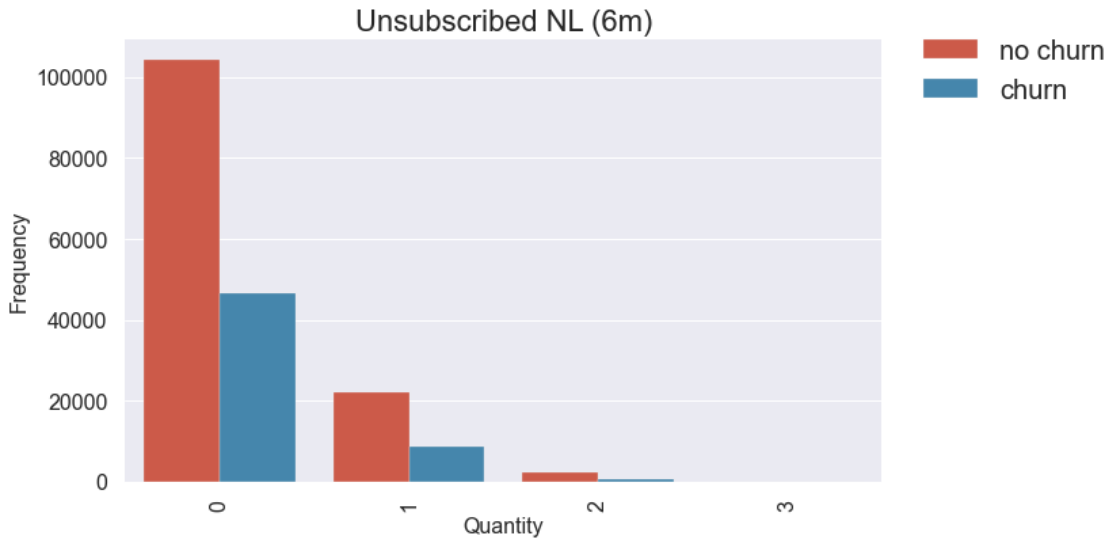
Aggregated Columns:

[253]:
```
title = ['Received NL (6m)', 'Opened NL (6m)', 'Clicked NL (6m)', 'Unsubscribed␣
↪NL (6m)']
labellist = ['no churn','churn']

for i, nl in enumerate(['nl_received_6m', 'nl_opened_6m', 'nl_clicked_6m',␣
↪'nl_unsubscribed_6m']):



    plt.subplots(figsize=(10,6))
    ax = sns.countplot(x=nl, hue='churn', data=df)
    ax.set(xlabel='Quantity', ylabel='Frequency')
    ax.set(ylim=(0, None))
    plt.title(title[i], fontsize=22)
    L=plt.legend(fontsize=20,loc=(1.04,0.83))
    L.get_texts()[0].set_text(labellist[0])
    L.get_texts()[1].set_text(labellist[1])

    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```




Opened NL (6m)
no churn
churn
Frequency
80000
70000
60000
50000
40000
30000
20000
10000
0
0
1
2
3
4
Quantity


Clicked NL (6m)
no churn
churn
Frequency
120000
100000
80000
60000
40000
20000
0
0
1
2
3
4
Quantity



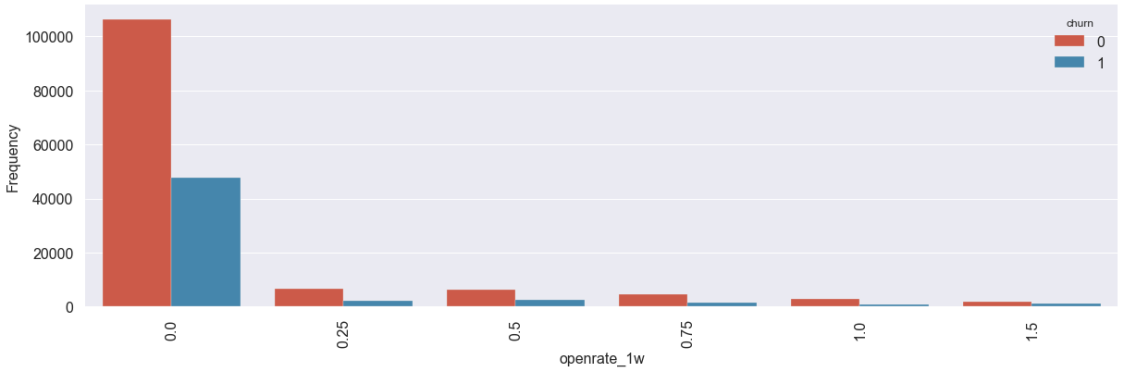
6.3.2 Clickrate and Openrate Features

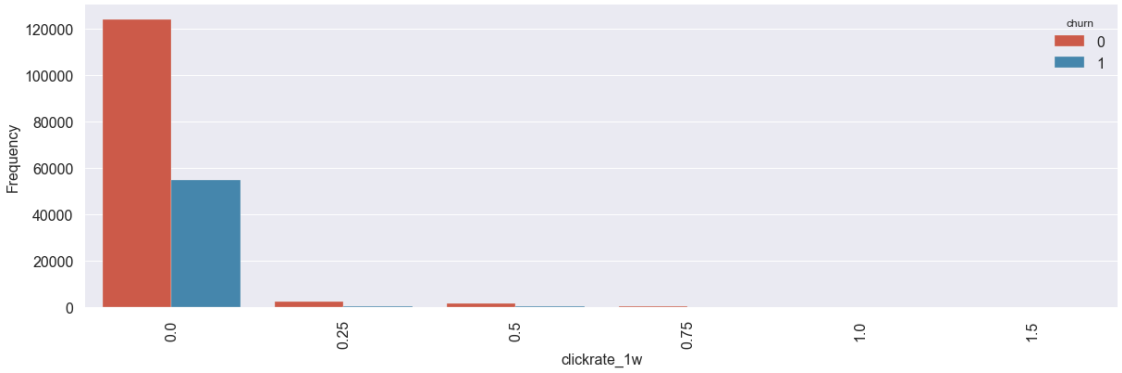
[254]:
```
def flatten_rate(flat):
    if 0.75 < flat < 1:
        return 1
    elif 0 < flat < 0.25:
        return 0.25
    elif 0.25 < flat < 0.5:
        return 0.5
    elif 0.5 < flat < 0.75:
        return 0.75
    elif flat > 1:
        return 1.5
    else:
        return 0
```

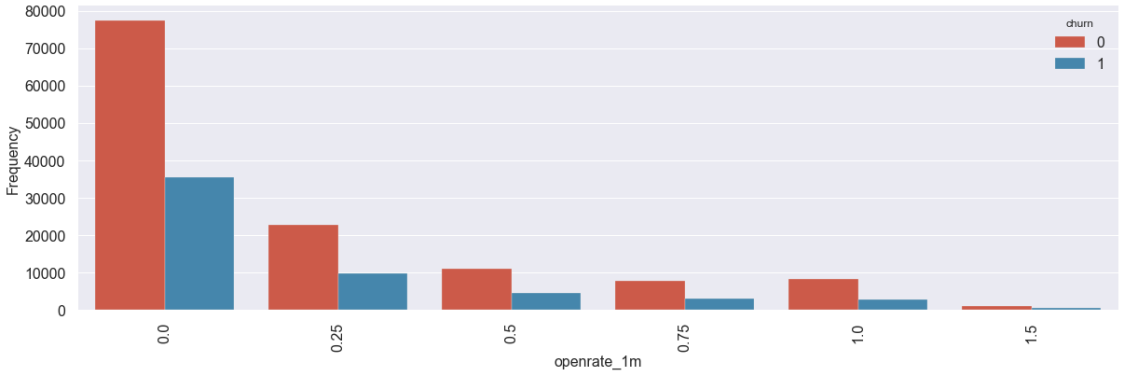
[255]:
```
for i in ['openrate_1w', 'clickrate_1w', 'openrate_1m', 'clickrate_1m',␣
↪'openrate_3m', 'clickrate_3m']:
    df[i] = df[i].apply(flatten_rate)
```

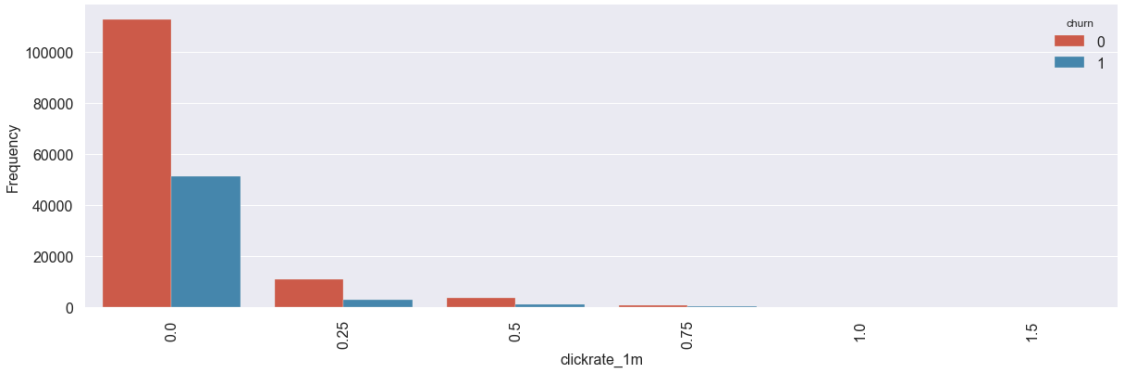
[256]:
```
for i in ['openrate_1w', 'clickrate_1w', 'openrate_1m', 'clickrate_1m',␣
↪'openrate_3m', 'clickrate_3m']:


    plt.subplots(figsize=(20,6))
    ax = sns.countplot(x=i, hue='churn', data=df)
    ax.set(xlabel=i, ylabel='Frequency')
```

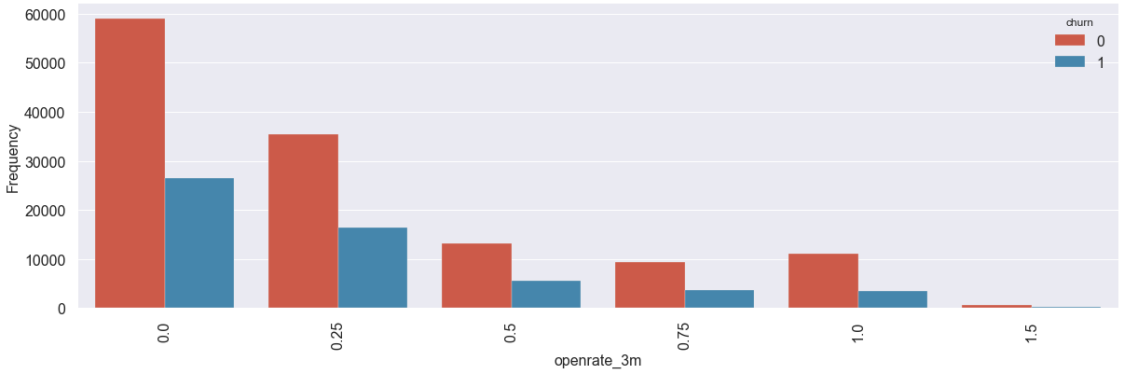
```
    ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```

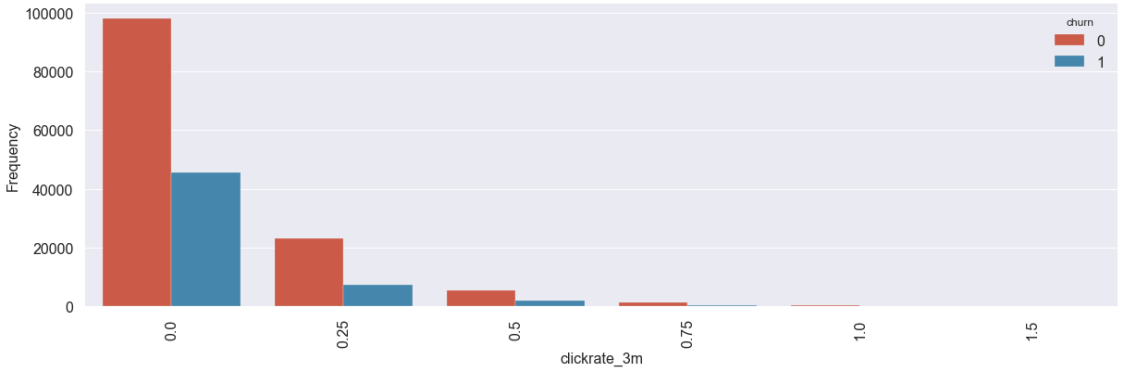












Observations: To have an better overview of the clickrates we binned them into manageable rates (0,0.25,0.5,0.75,1 and 1.5 as a representative of all rates bigger than 1)

6.3.3 Time Features to Integer Values

[257]:
```
df['liefer_beginn_evt'] = df['liefer_beginn_evt'].map(lambda x: x.year + x.
↪month/12.0)
```

[258]:
```
df['MONTH_DELTA_abo_min'] = (df.abo_registrierung_min - df.
↪abo_registrierung_min.min()).dt.days
```

[259]:
```
df['MONTH_DELTA_abo_min'] = df['MONTH_DELTA_abo_min'].map(lambda x: x/30)
```

[260]:
```
df['MONTH_DELTA_nl_min'] = (df.nl_registrierung_min - df.nl_registrierung_min.
↪min()).dt.days
```

[261]:
```
df['MONTH_DELTA_nl_min'] = df['MONTH_DELTA_nl_min'].map(lambda x: x/30)
```

6.4 Dropping

[266]:
```
df_drop = df.copy()
```

[267]:
```
droppinglist_obvious =␣
↪['training_set','avg_churn','date_x','kuendigungs_eingangs_datum']
```

[268]:
```
## Export Dataframe before dropping
```

[269]:
```
df_drop = df_drop.drop(droppinglist_obvious, axis=1)
```

[270]:
```
df_drop.to_csv('data/df_clean_engineered_all.csv',index=False)
```

6.4.1 Dropping NL mails

In the following section we drop the columns of the different newsletter mail activities, because we created some aggregated new columns consisting of those. Therefore these columns are redundant.

[271]:
```
droppinglist_tech = ["nl_blacklist_sum",
                     "nl_bounced_sum",
                     "nl_sperrliste_sum",
                     "nl_opt_in_sum",
                     "nl_fdz_organisch"]
```

[272]:
```
droppinglist_nl_mail = df_nl_bestandskunden_1.columns.values.tolist() +␣
↪df_nl_produktnews_1.columns.values.tolist() + df_nl_hamburg_1.columns.values.
↪tolist() + df_zb_1.columns.values.tolist()
```

6.4.2 Dropping cnt, zon, reg

[273]:
```
droppinglist_cnt = list(df_cnt.columns[1:])
droppinglist_zon = list(df_zon.columns)
droppinglist_reg = list(df_reg.columns)
```

6.4.3 Dropping Time and Customer/personal features

[274]:
```
# time feature dropping
droppinglist_time = ['abo_registrierung_min','nl_registrierung_min']
```

[275]:
```
# we want to include only the best geographical information, so we drop plz_1,␣
↪plz_2 and keep plz_3. ort is therefore also deleted.
droppinglist_geo = ['plz_1','plz_2','ort'] # maybe keep ort instead of plz due␣
↪to the higher information for abroad cities.
```

6.4.4 Do the dropping

[276]:
```
# function to drop feature lists
def drop_list(df_drop,list_to_drop):
    df_drop = df_drop.drop(list_to_drop, axis=1)
    print(f"Number of features after dropping {df_drop.shape[1]}")
    return df_drop
```

[277]:
```
# drop obvious features
#df_drop = drop_list(df_drop, droppinglist_obvious)
# drop nl_mail features
df_drop = drop_list(df_drop, droppinglist_nl_mail)
# drop tech features
df_drop = drop_list(df_drop, droppinglist_tech)
# drop cnt feature
df_drop = drop_list(df_drop, droppinglist_cnt)
# drop zon features
df_drop = drop_list(df_drop, droppinglist_zon)
# drop reg features
df_drop = drop_list(df_drop, droppinglist_reg)
# drop time features
df_drop = drop_list(df_drop, droppinglist_time)
# drop geographical features
df_drop = drop_list(df_drop, droppinglist_geo)
```

```
Number of features after dropping 121
Number of features after dropping 116
Number of features after dropping 112
Number of features after dropping 98
Number of features after dropping 94
```

```
Number of features after dropping 92
Number of features after dropping 89
```

[278]:
```
fig = plt.figure()
ax = plt.gca()
df_drop.drop('churn', axis=1).corrwith(df_drop.churn).sort_values().
 ↪plot(kind='barh',figsize=(10, 25));
ax.tick_params(axis="x", bottom=True, top=True, labelbottom=True, labeltop=True)
fig.savefig('presentation/churn_correlation_feature_engineering.
 ↪pdf',bbox_inches='tight')
```


−0.2
−0.1
0.0
0.1
0.2
liefer_beginn_evt
studentenabo
rechnungsmonat
sum_zon
MONTH_DELTA_nl_min
nl_zeitbrief
metropole
unsubscribed_anzahl_1m
unsubscribed_anzahl_1w
nl_opened_1m
unsubscribed_anzahl_3m
unsubscribed_anzahl_6m
email_am_kunden
nl_clicked_3m
nl_received_1w
openrate_zeitbrief_1w
clickrate_produktnews_1w
clickrate_zeitbrief_1w
clickrate_bestandskunden_1w
unterbrechung
nl_zeitverlag_hamburg
nl_opened_1w
clickrate_zeitbrief_3m
nl_received_6m
openrate_produktnews_1w
clickrate_zeitbrief_1m
clickrate_bestandskunden_1m
openrate_bestandskunden_1w
clickrate_produktnews_1m
clickrate_bestandskunden_3m
nl_received_3m
nl_clicked_6m
clickrate_hamburg_1w
clickrate_hamburg_3m
MONTH_DELTA_abo_min
openrate_zeitbrief_1m
clickrate_produktnews_3m
clickrate_hamburg_1m
openrate_hamburg_1w
nl_received_1m
openrate_hamburg_3m
openrate_hamburg_1m
nl_opened_3m
openrate_zeitbrief_3m
openrate_1w
openrate_produktnews_1m
nl_clicked_1m
openrate_bestandskunden_1m
nl_opened_6m
nl_clicked_1w
nl_unsubscribed_1w
nl_unsubscribed_6m
openrate_bestandskunden_3m
nl_unsubscribed_1m
sum_reg
clickrate_1w
openrate_produktnews_3m
clicked_anzahl_1w
nl_unsubscribed_3m
openrate_1m
openrate_3m
cnt_abo
clickrate_1m
clickrate_3m
clicked_anzahl_1m
opened_anzahl_1w
clicked_anzahl_3m
clicked_anzahl_6m
shop_kauf
opened_anzahl_1m
received_anzahl_1w
opened_anzahl_3m
openedanzahl_6m
received_anzahl_3m
received_anzahl_6m
received_anzahl_1m
nl_zeitshop
nl_aktivitaet
lesedauer
−0.2
−0.1
0.0
0.1
0.2

6.5 Export final Dataframe

[279]:
```
df_drop.to_csv('data/df_clean_engineered_drop.csv',index=False)
```

7 Predictive Modelling

Details of the used machine learning methods are given in our machine learning notebook: ML Notebook

Details of the Artifical Neural Network are given in our ANN notebook: ANN Notebook