Exploratory Data Analysis(Part 2)

AcadView

May 28, 2018

1 Overview

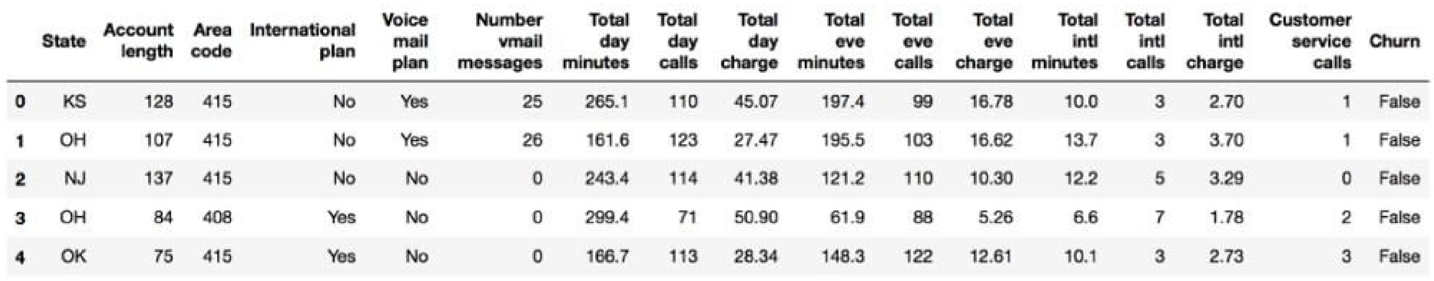
Working on single variables allows you to spot a large number of outlying observations. However, outliers do not necessarily display values too far from the norm. Sometimes outliers are made of unusual combinations of values in more variables. They are rare, but influential, combinations that can especially trick machine learning algorithms.

In such cases, the precise inspection of every single variable wont suffice to rule out anomalous cases from your dataset. Only a few selected techniques, taking in consideration more variables at a time, will manage to reveal problems in your data.

In this section we will look into how to view multivariate data and analyze it to get insights from it. Well demonstrate the main methods in action by analyzing a dataset on the churn rate of telecom operator clients. Lets read the data (using read csv), and take a look at the first 5 lines using the head method:

```
import pandas as pd
import numpy as np
df = pd.read_csv('../../data/telecom_churn.csv')
df.head()
```

The output is as follows:

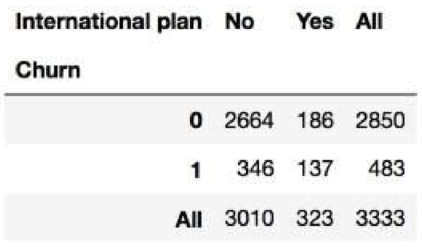


	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total intl minutes	Total intl calls	Total intl charge	Customer service calls	Churn
0	KS	128	415	No	Yes	25	265.1	110	45.07	197.4	99	16.78	10.0	3	2.70	1	False
1	OH	107	415	No	Yes	26	161.6	123	27.47	195.5	103	16.62	13.7	3	3.70	1	False
2	NJ	137	415	No	No	0	243.4	114	41.38	121.2	110	10.30	12.2	5	3.29	0	False
3	OH	84	408	Yes	No	0	299.4	71	50.90	61.9	88	5.26	6.6	7	1.78	2	False
4	OK	75	415	Yes	No	0	166.7	113	28.34	148.3	122	12.61	10.1	3	2.73	3	False

2 Simple Bivariate Data Analysis

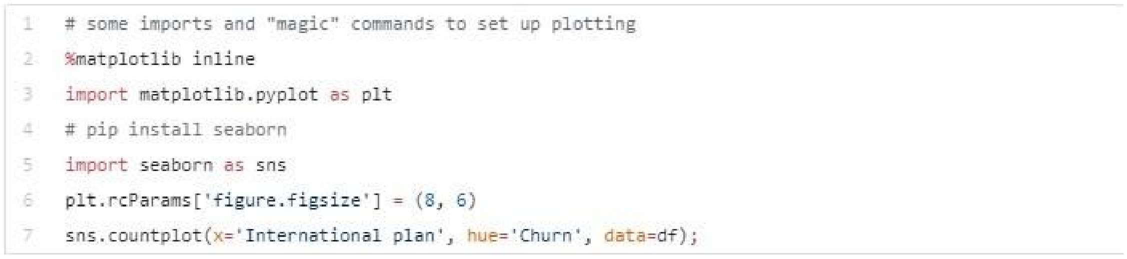
Lets see how churn rate is related to the International plan variable. Well do this using a crosstab contingency table and also through visual analysis with Seaborn .

```
pd.crosstab(df['Churn'], df['International plan'], margins=True)
```



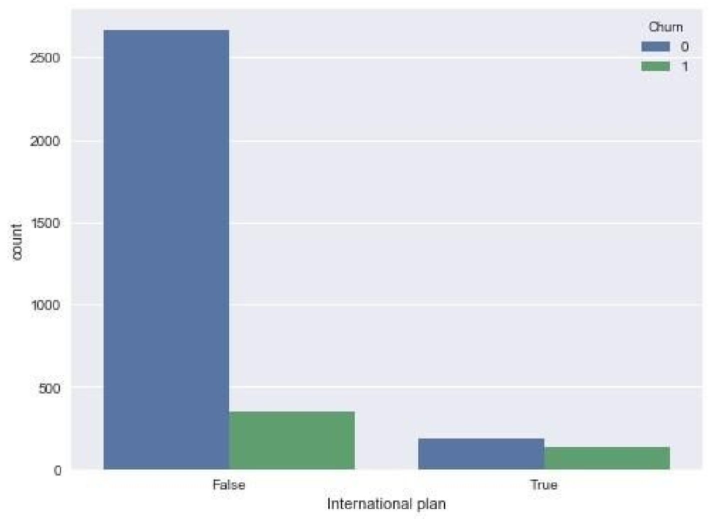
International plan	No	Yes	All
Churn			
0	2664	186	2850
1	346	137	483
All	3010	323	3333

Visualization using seaborn



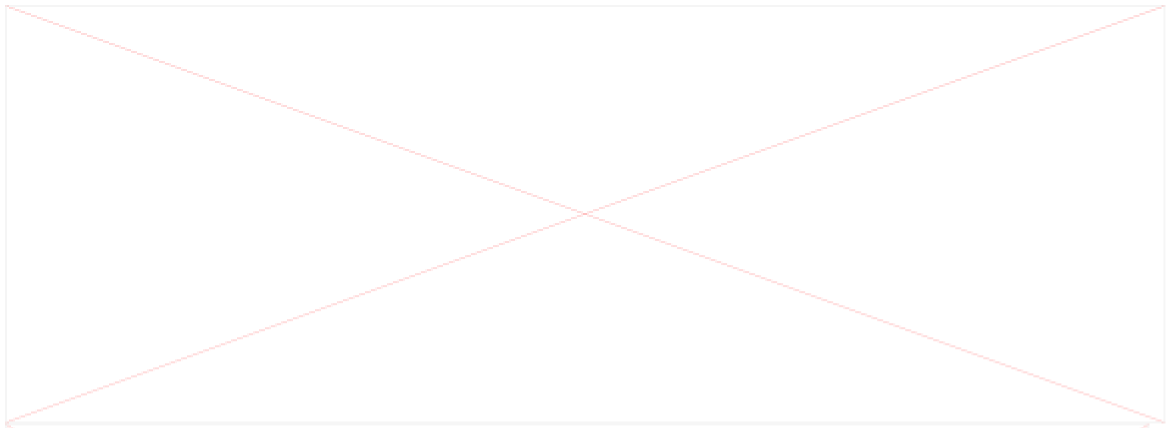
```
# some imports and "magic" commands to set up plotting
%matplotlib inline
import matplotlib.pyplot as plt
# pip install seaborn
import seaborn as sns
plt.rcParams['figure.figsize'] = (8, 6)
sns.countplot(x='International plan', hue='Churn', data=df);
```

The output is as follows.



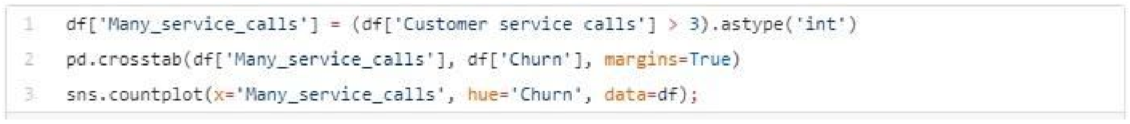
We see that, with International Plan, the churn rate is much higher, which is an interesting observation! Perhaps large and poorly controlled expenses with international calls are very conflict-prone and lead to dissatisfaction among the telecom operators customers.

Next, lets look at another important featureCustomer service calls. Lets also make a summary table and a picture.



Perhaps, it is not so obvious from the summary table, but the picture clearly states that the churn rate strongly increases starting from 4 calls to the service center.

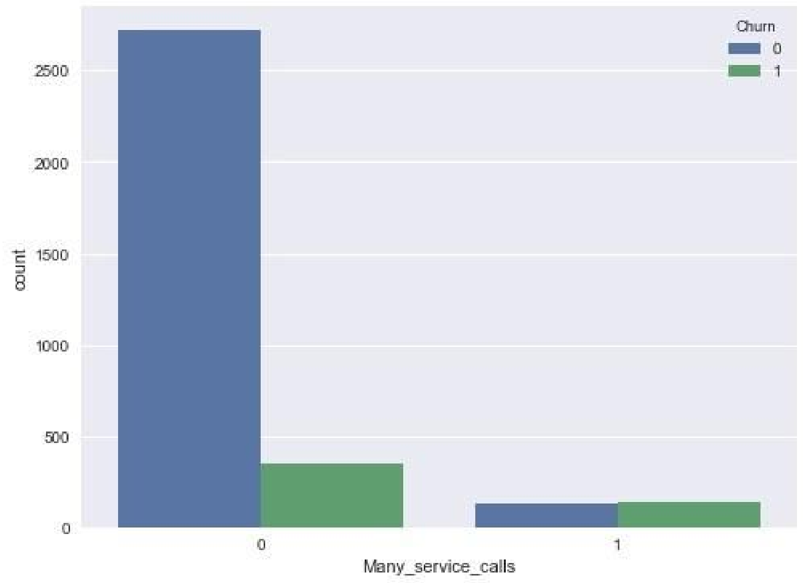
Lets now add a binary attribute to our DataFrameCustomer service calls ¿ 3. And once again, let’s see how it relates to the churn.



```
df['Many_service_calls'] = (df['Customer service calls'] > 3).astype('int')
pd.crosstab(df['Many_service_calls'], df['Churn'], margins=True)
sns.countplot(x='Many_service_calls', hue='Churn', data=df);
```

The output is as follows:

Churn	0	1	All
Many_service_calls			
0	2721	345	3066
1	129	138	267
All	2850	483	3333



Therefore, predicting that a customer will churn (Churn=1) in the case when the number of calls to the service center is greater than 3 and the International Plan is added (and predicting Churn=0 otherwise), we might expect an accuracy of 85.8% (we are mistaken only 464 + 9 times). This number, 85.8%, that we got with very simple reasoning serves as a good starting point (baseline) for the further machine learning models that we will build.

3 Multivariate visualization

Multivariate plots allow us to see relationships between two and more different variables, all in one figure. Just as in the case of univariate plots, the specific type of visualization will depend on the types of the variables being analyzed.

3.1 QuantitativeQuantitative

We are going to start with the interaction between quantitative variables.

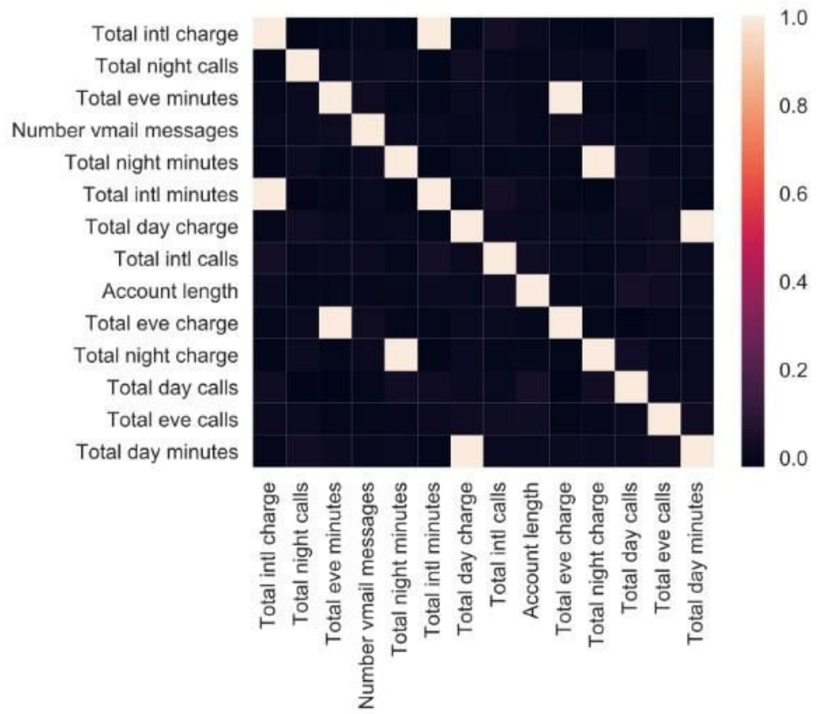
3.1.1 Correlation matrix

Lets look at the correlations among the numerical variables in our dataset. This information is important to know as there are Machine Learning algorithms (for example, linear and logistic regression) that do not handle highly correlated input variables well.

First, we will use the method corr() on a DataFrame that calculates the correlation between each pair of features. Then, we pass the resulting correlation matrix to heatmap() from seaborn, which renders a color-coded matrix for the provided values:

```
# Drop non-numerical variables
numerical = list(set(df.columns) -
                 set(['State', 'International plan',
                      'Voice mail plan', 'Area code', 'Churn',
                      'Customer service calls']))

# Calculate and plot
corr_matrix = df[numerical].corr()
sns.heatmap(corr_matrix);
```



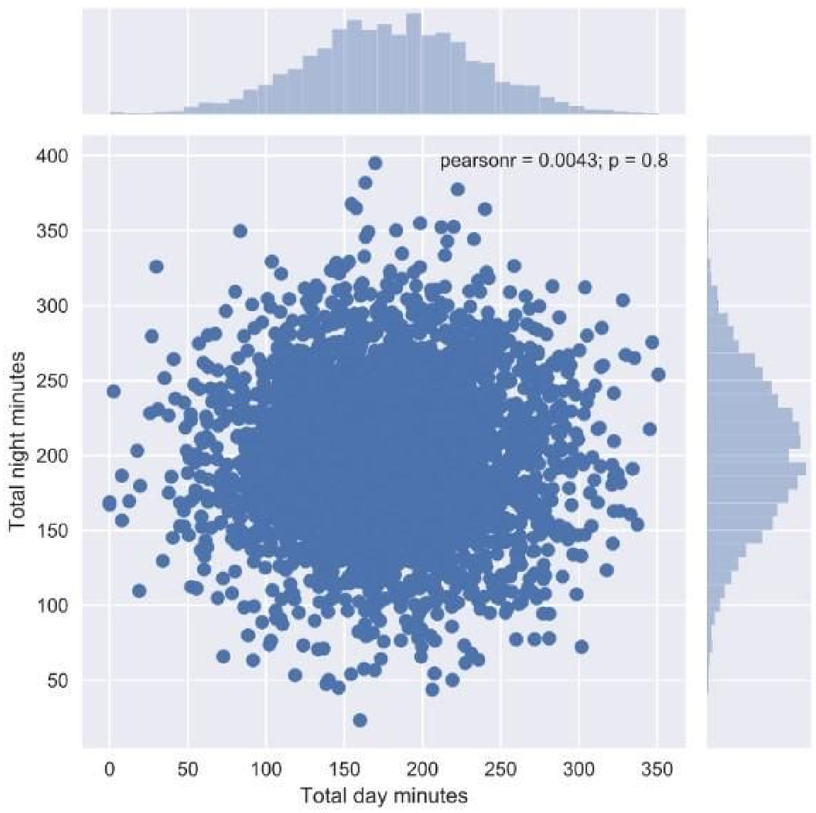
From the colored correlation matrix generated above, we can see that there are 4 variables such as Total day charge that have been calculated directly from the number of minutes spent on phone calls (Total day minutes). These are called dependent variables and can therefore be left out since they do not contribute any additional information. Lets get rid of them:

```
numerical = list(set(numerical) -
                 set(['Total day charge', 'Total eve charge',
                      'Total night charge', 'Total intl charge']))
```

3.1.2 Scatter plot

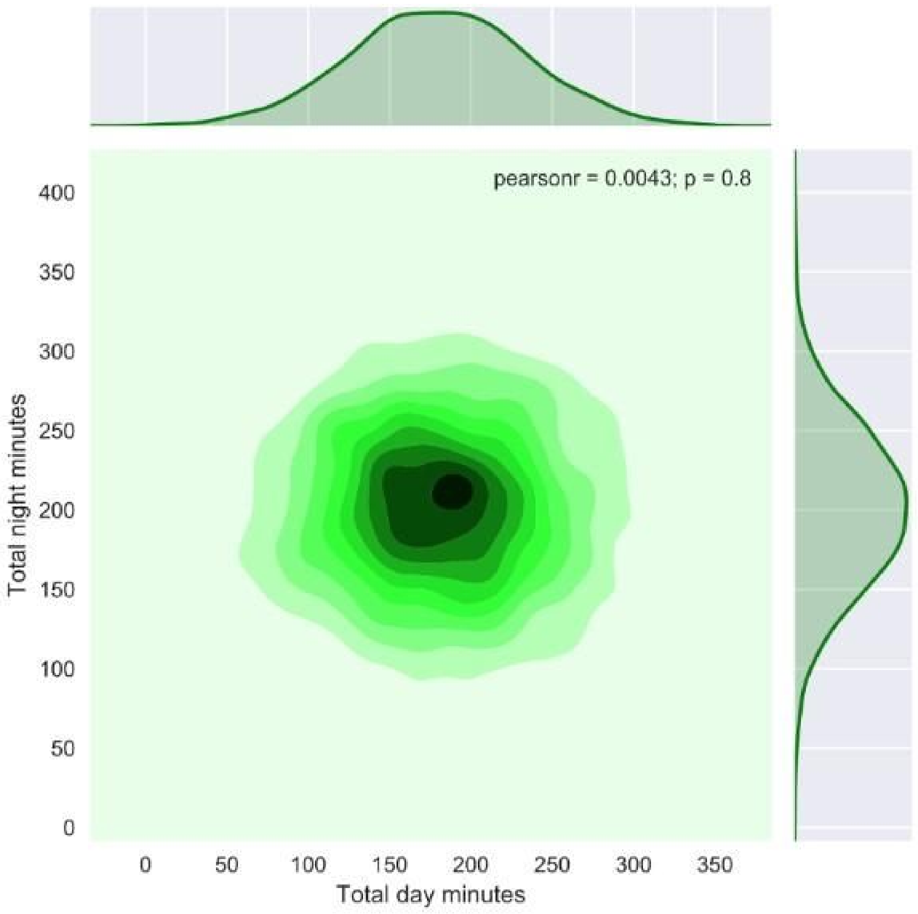
The scatter plot displays values of two numerical variables as Cartesian coordinates in 2D space. Scatter plots in 3D are also possible.
There is a slightly fancy option to create a scatter plot with the seaborn library:

```
sns.jointplot(x='Total day minutes', y='Total night minutes',
              data=df, kind='scatter');
```



The function jointplot() plots two histograms that may be useful in some cases.Using the same function, we can also get a smoothed version of our bivariate distribution:

```
sns.jointplot('Total day minutes', 'Total night minutes',
              data=df, kind="kde", color="g");
```



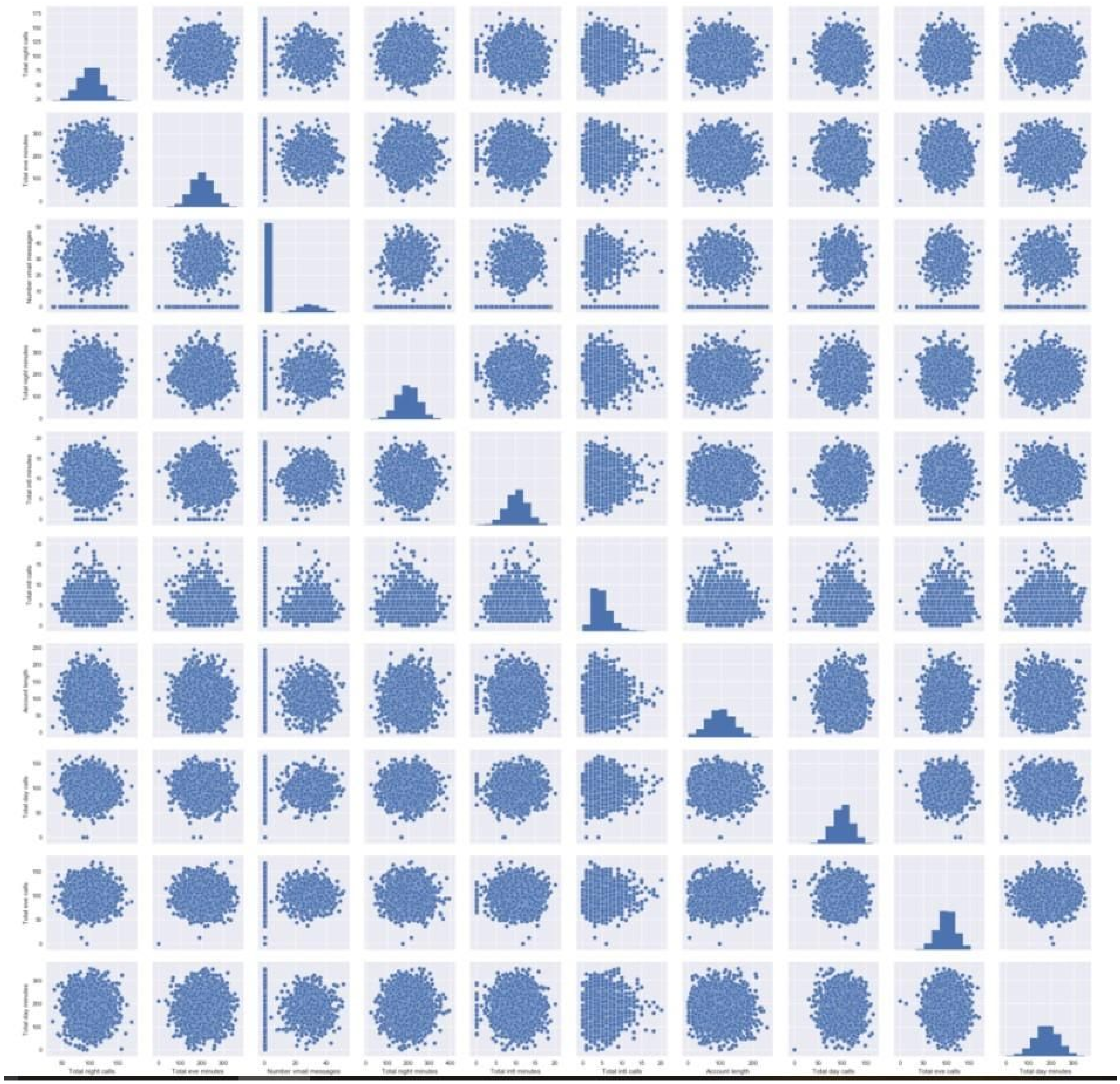
This is basically a bivariate version of the Kernel Density Plot discussed earlier in EDA part1.

3.1.3 Scatterplot matrix

In some cases, we may want to plot a scatterplot matrix such as the one shown below. Its diagonal contains the distributions of the corresponding variables, and the scatter plots for each pair of variables fill the rest of the matrix.

```
# pairplot may become very slow with the SVG format
%config InlineBackend.figure_format = 'png'
sns.pairplot(df[numerical]);
```

The output is as follows:

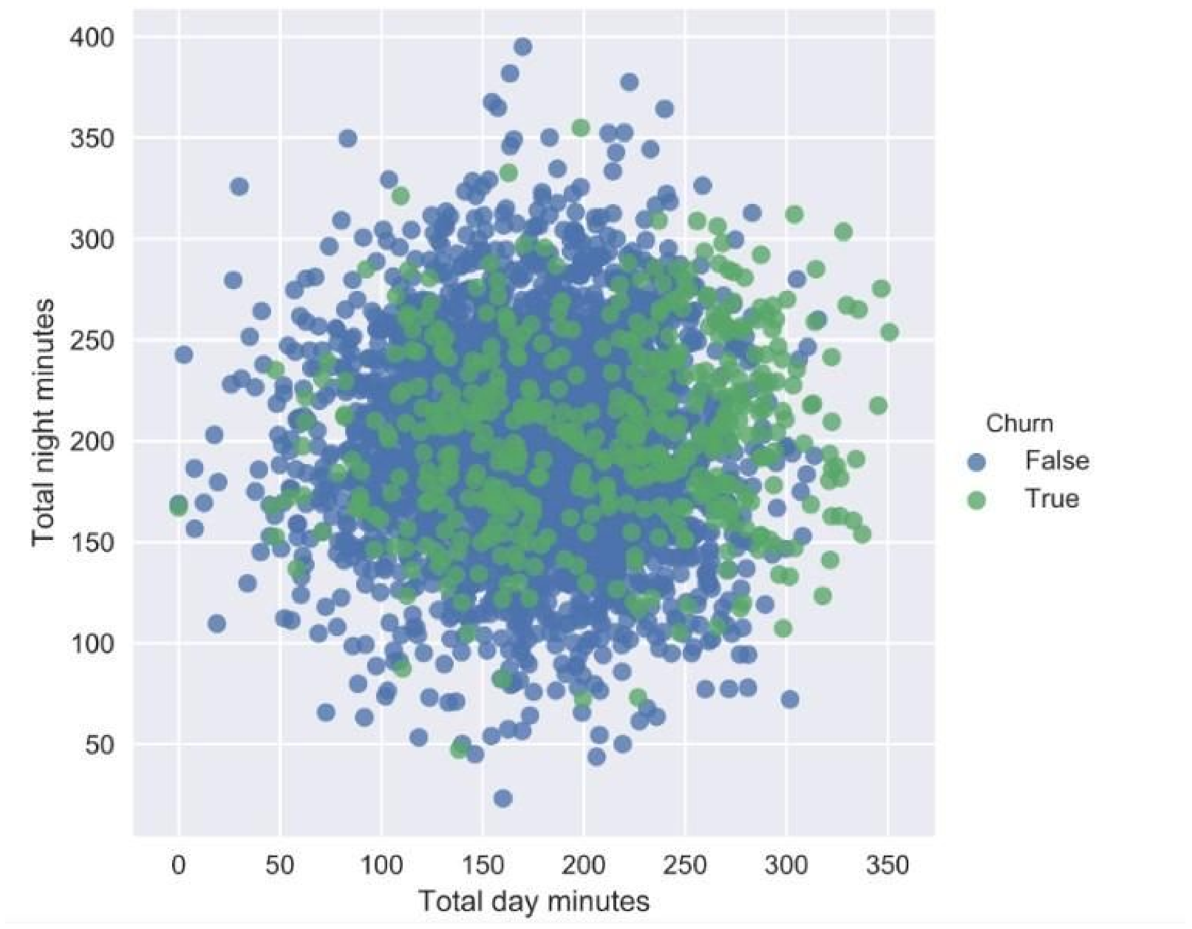


Sometimes, such visualization may help draw conclusions about data; but, in this case, everything is pretty clear with no surprises.

3.2 QuantitativeCategorical

In this section, we will make our simple quantitative plots a little more exciting. We will try to gain new insights for churn prediction from the interactions between the numerical and categorical features.
More specifically, lets see how the input variables are related to the target variable Churn. Previously, you learned about scatter plots. Additionally, their points can be color or size coded so that the values of a third categorical variable are also presented in the same figure. We can achieve this with the scatter() function seen above, but, let's try a new function called lmplot() and use the parameter hue to indicate our categorical feature of interest:

```
sns.lmplot('Total day minutes', 'Total night minutes', data=df,
           hue='Churn', fit_reg=False);
```



It seems that our small proportion of disloyal customers lean towards the top-right corner; that is, such customers tend to spend more time on the phone during both day and night. But this is not absolutely clear, and we wont make any definitive conclusions from this chart.

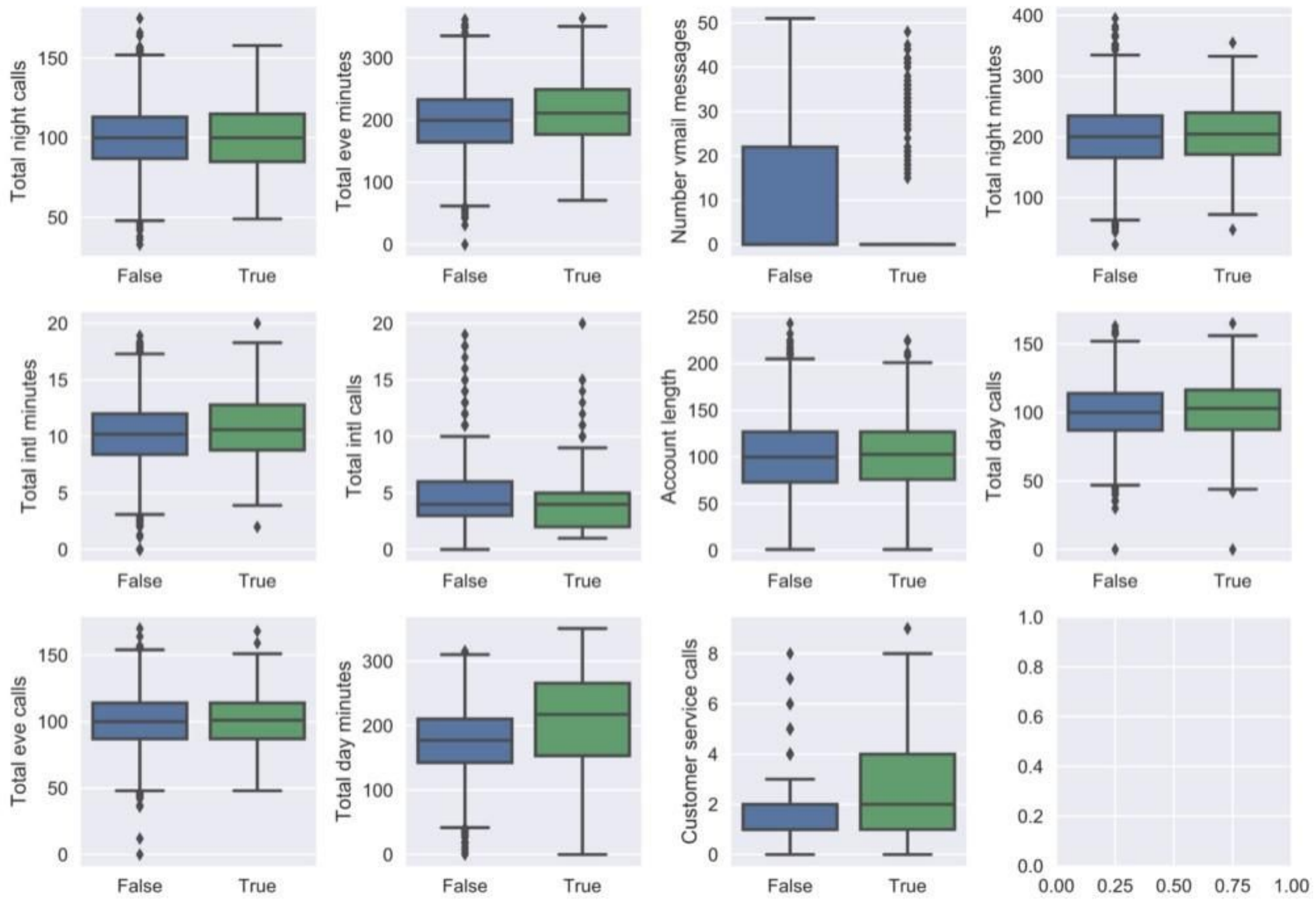
Now, lets create box plots to visualize the distribution statistics of the numerical variables in two disjoint groups: the loyal customers (Churn=False) and those who left (Churn=True).

```
# Sometimes you can analyze an ordinal variable as numerical one
numerical.append('Customer service calls')

fig, axes = plt.subplots(nrows=3, ncols=4, figsize=(10, 7))

for idx, feat in enumerate(numerical):
    ax = axes[int(idx / 4), idx % 4]
    sns.boxplot(x='Churn', y=feat, data=df, ax=ax)
    ax.set_xlabel('')
    ax.set_ylabel(feat)
fig.tight_layout();
```

The output is as follows:



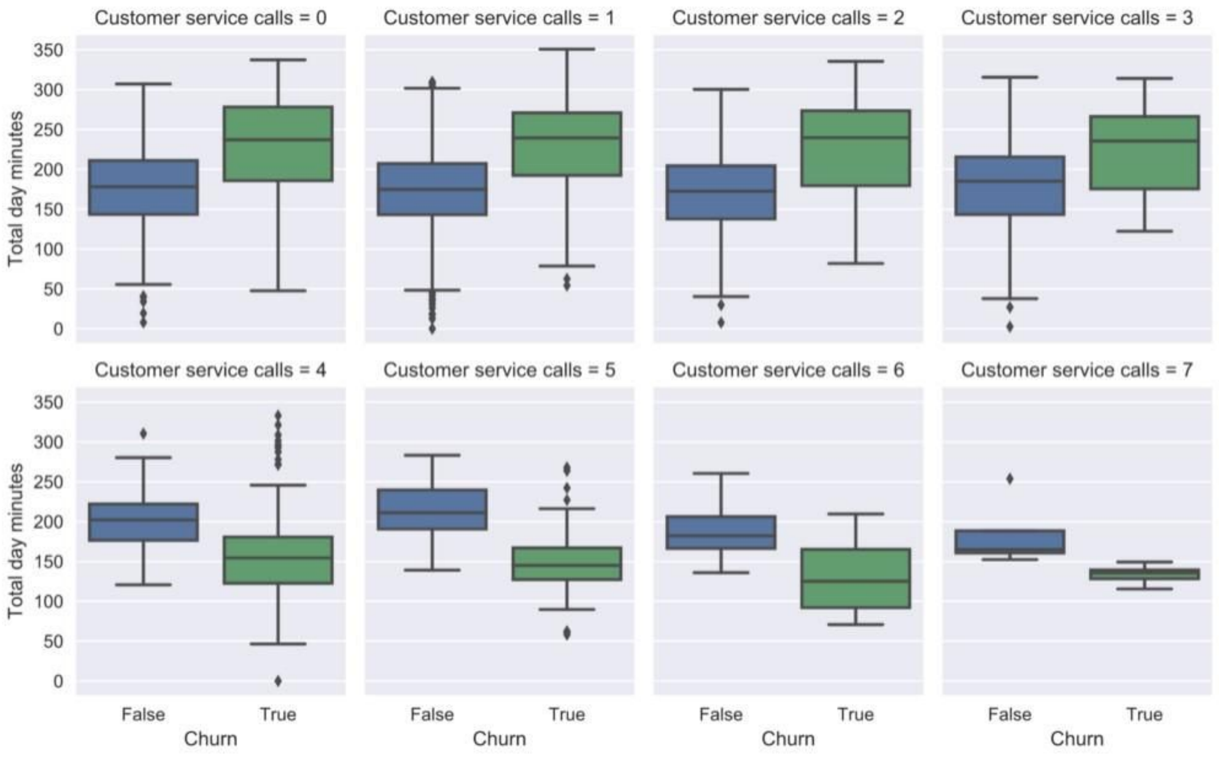
From this chart, we can see that the greatest discrepancy in distribution between the two groups is for three variables: Total day minutes, Customer service calls, and Number vmail messages.

An interesting observation: on average, customers that discontinue their contracts are more active users of communication services. Perhaps they are unhappy with the tariffs, so a possible measure to prevent churn could be a reduction in call rates. The company will need to undertake additional economic analysis to find out whether such measures would be beneficial.

When we want to analyze a quantitative variable in two categorical dimensions at once, there is a suitable function for this in the seaborn library called factorplot(). For example, let's visualize the interaction between Total day minutes and two categorical variables in the same plot:

```
sns.factorplot(x='Churn', y='Total day minutes',
               col='Customer service calls',
               data=df[df['Customer service calls'] < 8],
               kind="box", col_wrap=4, size=3, aspect=.8);
```

The output is as follows:

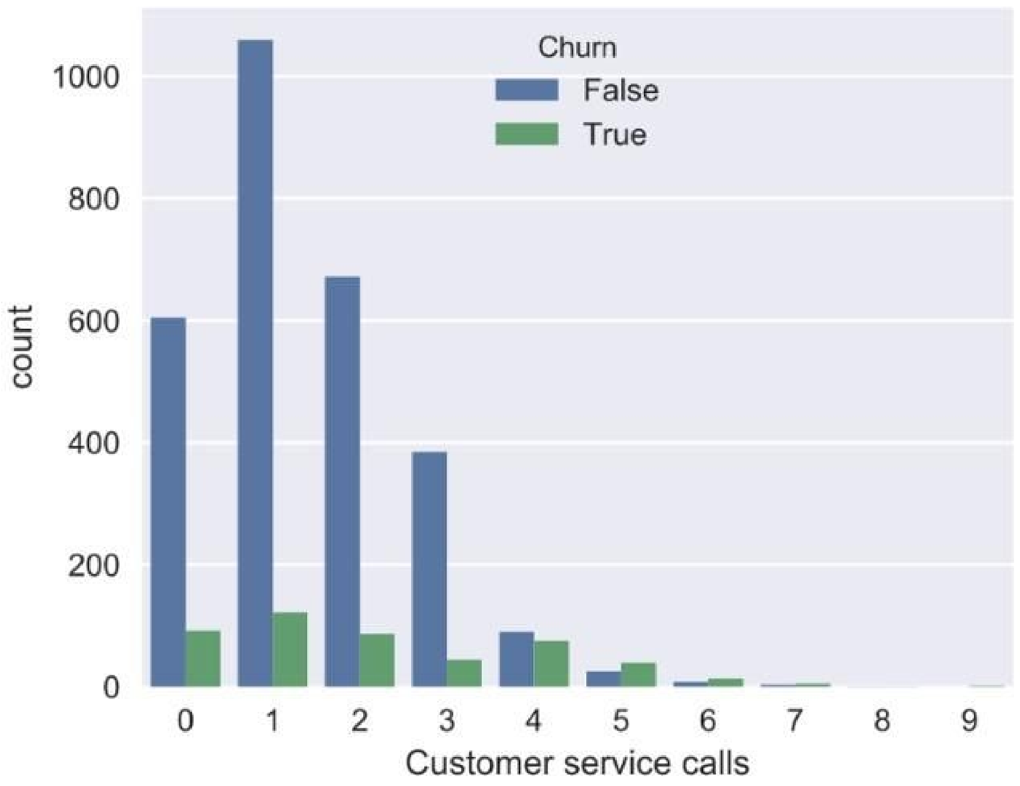


3.3 CategoricalCategorical

As we saw earlier in this article, the variable Customer service calls has few unique values and, thus, can be considered either numerical or ordinal. We have already seen its distribution with a count plot. Now, we are interested in the relationship between this ordinal feature and the target variable Churn.

Lets look at the distribution of the number of calls to the customer service, again using a count plot. This time, lets also pass the parameter hue=Churn that adds a categorical dimension to the plot:

```
sns.countplot(x='Customer service calls', hue='Churn', data=df);
```



observation: the churn rate increases significantly after 4 or more calls to the customer service.

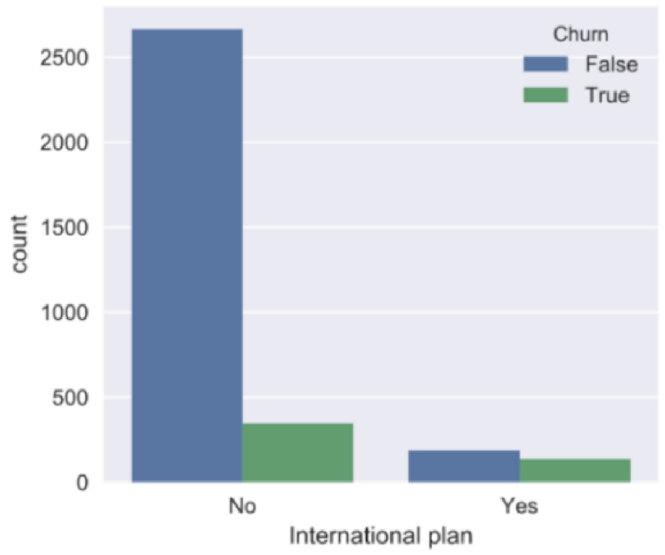
Now, lets look at the relationship between Churn and the binary features, International plan and Voice mail plan.

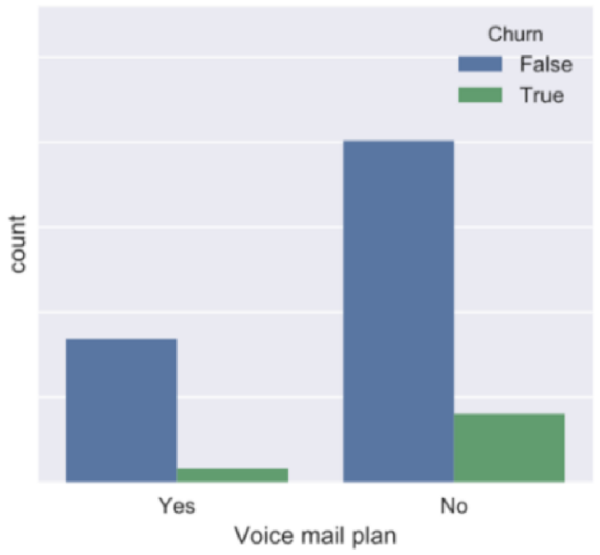
```
_, axes = plt.subplots(1, 2, sharey=True, figsize=(10, 4))

sns.countplot(x='International plan', hue='Churn',
              data=df, ax=axes[0]);
sns.countplot(x='Voice mail plan', hue='Churn',
              data=df, ax=axes[1]);
```

The output is as follows:

observation: when International Plan is enabled, the churn rate is much higher; the usage of the international plan by the customer is a strong feature. We do not observe the same effect with Voice mail plan.


Churn
False
True
count
No
Yes
International plan


Churn
False
True
count
Yes
No
Voice mail plan