Churn_Week_01

January 16, 2019

1 Practical 1 - Churn Dataset

In [581]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")
```

In [582]:
```
from IPython.display import Markdown, display

OUT = "../../../topics/01-Module_Introduction/10-Churn_Dataset_-_Review_of_Pandas/files"
OUT = "output"
import os
os.makedirs(OUT, exist_ok=True)
```

1.1 Load and Prepare the Data

In [583]:
```
df = pd.read_csv("src/churn.csv")
```

In [584]:
```
print ("dataset has %s rows" %(df.shape[0]))
```

```
dataset has 3333 rows
```

In [585]:
```
message = (" * Data set consists of %d cases (rows) with %s attributes (cols) and a single target."
% (df.shape[0], df.shape[1]-1))

Markdown(message)
```

Out[585]:

- Data set consists of 3333 cases (rows) with 20 attributes (cols) and a single target.

In [586]:
```
df.head()
```

Out[586]:

	State	Account Length	Area Code	Phone	Int'l Plan	VMail Plan	\
0	KS	128	415	382-4657	no	yes	
1	OH	107	415	371-7191	no	yes	
2	NJ	137	415	358-1921	no	no	
3	OH	84	408	375-9999	yes	no	
4	OK	75	415	330-6626	yes	no	

	VMail Message	Day Mins	Day Calls	Day Charge	...	Eve Calls	\
0	25	265.1	110	45.07	...	99	
1	26	161.6	123	27.47	...	103	
2	0	243.4	114	41.38	...	110	
3	0	299.4	71	50.90	...	88	
4	0	166.7	113	28.34	...	122	

	Eve Charge	Night Mins	Night Calls	Night Charge	Intl Mins	Intl Calls	\
0	16.78	244.7	91	11.01	10.0	3	
1	16.62	254.4	103	11.45	13.7	3	
2	10.30	162.6	104	7.32	12.2	5	
3	5.26	196.9	89	8.86	6.6	7	
4	12.61	186.9	121	8.41	10.1	3	

	Intl Charge	CustServ Calls	Churn?
0	2.70	1	False.
1	3.70	1	False.

```
2         3.29              0  False.
3         1.78              2  False.
4         2.73              3  False.

[5 rows x 21 columns]
```

Get list of columns, some name contain spaces, or other unsuitable characters, which need to be removed.

```
In [587]: names = df.columns.tolist()
          print("Original columns names:\n", names)

Original columns names:
 ['State', 'Account Length', 'Area Code', 'Phone', "Int'l Plan", 'VMail Plan', 'VMail Message', 'Day Mins', 'Day Calls', 'Day Ch


In [588]: CORRECTIONS = {" ":"_", "'":"", "?":"", "CustServ":"Cust_Serv"}

          def fixName(s):
              for a,b in CORRECTIONS.items():
                  s = s.replace(a,b)
              return s

          mapping = {c:fixName(c) for c in names}
          mapping

Out[588]: {'State': 'State',
           'Account Length': 'Account_Length',
           'Area Code': 'Area_Code',
           'Phone': 'Phone',
           "Int'l Plan": 'Intl_Plan',
           'VMail Plan': 'VMail_Plan',
           'VMail Message': 'VMail_Message',
           'Day Mins': 'Day_Mins',
           'Day Calls': 'Day_Calls',
           'Day Charge': 'Day_Charge',
```

```
 'Eve Mins': 'Eve_Mins',
 'Eve Calls': 'Eve_Calls',
 'Eve Charge': 'Eve_Charge',
 'Night Mins': 'Night_Mins',
 'Night Calls': 'Night_Calls',
 'Night Charge': 'Night_Charge',
 'Intl Mins': 'Intl_Mins',
 'Intl Calls': 'Intl_Calls',
 'Intl Charge': 'Intl_Charge',
 'CustServ Calls': 'Cust_Serv_Calls',
 'Churn?': 'Churn'}
```

```
In [589]: df.rename(columns=mapping, inplace=True)
```

```
In [590]: df.head()
```

Out[590]:

	State	Account_Length	Area_Code	Phone	Intl_Plan	VMail_Plan	\
0	KS	128	415	382-4657	no	yes	
1	OH	107	415	371-7191	no	yes	
2	NJ	137	415	358-1921	no	no	
3	OH	84	408	375-9999	yes	no	
4	OK	75	415	330-6626	yes	no	

	VMail_Message	Day_Mins	Day_Calls	Day_Charge	...	Eve_Calls	\
0	25	265.1	110	45.07	...	99	
1	26	161.6	123	27.47	...	103	
2	0	243.4	114	41.38	...	110	
3	0	299.4	71	50.90	...	88	
4	0	166.7	113	28.34	...	122	

	Eve_Charge	Night_Mins	Night_Calls	Night_Charge	Intl_Mins	Intl_Calls	\
0	16.78	244.7	91	11.01	10.0	3	
1	16.62	254.4	103	11.45	13.7	3	
2	10.30	162.6	104	7.32	12.2	5	
3	5.26	196.9	89	8.86	6.6	7	

```
4       12.61       186.9          121          8.41       10.1           3

   Intl_Charge  Cust_Serv_Calls   Churn
0         2.70                1  False.
1         3.70                1  False.
2         3.29                0  False.
3         1.78                2  False.
4         2.73                3  False.

[5 rows x 21 columns]
```

Replace the binary target and the two binary features with numerical values.

```
In [591]: df.Intl_Plan = df.Intl_Plan.map( {"yes":1, "no":0} )
          df.VMail_Plan = df.VMail_Plan.apply( lambda x: int(x=='yes') )
          df.Churn = df.Churn.apply( lambda x: int(x=="True.") )

In [592]: df.head()

Out[592]:   State  Account_Length  Area_Code     Phone  Intl_Plan  VMail_Plan  \
          0    KS             128        415  382-4657          0           1
          1    OH             107        415  371-7191          0           1
          2    NJ             137        415  358-1921          0           0
          3    OH              84        408  375-9999          1           0
          4    OK              75        415  330-6626          1           0

             VMail_Message  Day_Mins  Day_Calls  Day_Charge  ...    Eve_Calls  \
          0             25     265.1        110       45.07  ...           99
          1             26     161.6        123       27.47  ...          103
          2              0     243.4        114       41.38  ...          110
          3              0     299.4         71       50.90  ...           88
          4              0     166.7        113       28.34  ...          122

             Eve_Charge  Night_Mins  Night_Calls  Night_Charge  Intl_Mins  Intl_Calls  \
          0       16.78       244.7           91         11.01       10.0           3
```

```
1       16.62       254.4          103         11.45       13.7           3
2       10.30       162.6          104          7.32       12.2           5
3        5.26       196.9           89          8.86        6.6           7
4       12.61       186.9          121          8.41       10.1           3

   Intl_Charge  Cust_Serv_Calls  Churn
0         2.70                1      0
1         3.70                1      0
2         3.29                0      0
3         1.78                2      0
4         2.73                3      0

[5 rows x 21 columns]
```

Split the columns in to target and type of attribute type

```
In [593]: target = "Churn"
          attributes = df.columns.tolist()
          attributes.remove(target)
          attributesDiscrete = ["Intl_Plan", "VMail_Plan"]
          df[attributesDiscrete].nunique()

Out[593]: Intl_Plan     2
          VMail_Plan    2
          dtype: int64
```

1.2 Exploratory Data Analysis

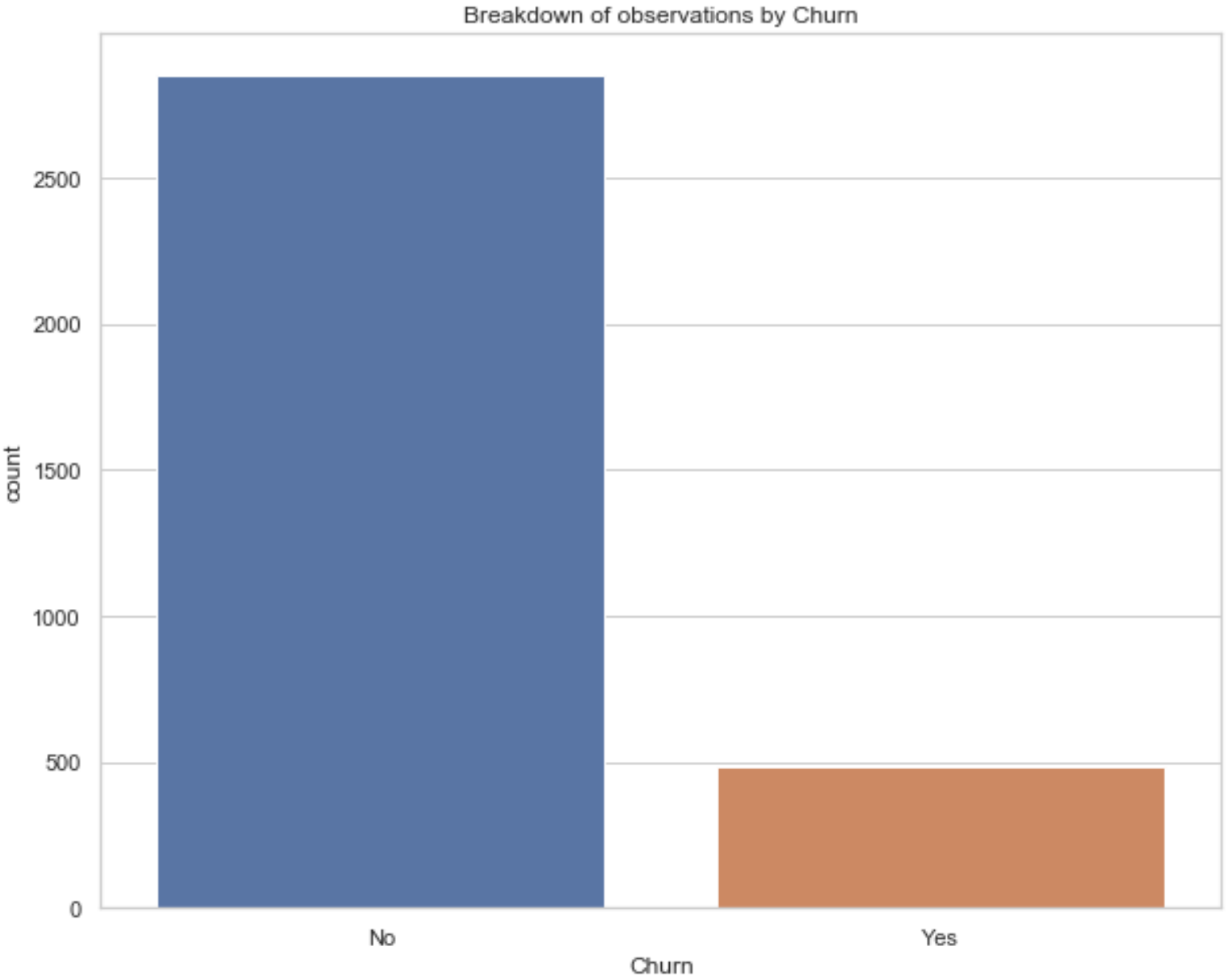
Note, you don't have to answer all questions for all variables. This list is intended to prompt you to 1) think about each of these and 2) can you do it?

1.2.1 Target Variable

```
In [594]: df1 = df.Churn.value_counts()
          df1
```

```
Out[594]: 0    2850
          1     483
          Name: Churn, dtype: int64
```

```
In [595]: ax = sns.countplot(x="Churn", data=df)
          ax.set_title("Breakdown of observations by Churn")
          ax.set_xticklabels(["No","Yes"])
          plt.savefig(OUT + "/eda_Churn.png",  bbox_inches="tight")
          plt.show()
```


Breakdown of observations by Churn
count
2500
2000
1500
1000
500
0
No
Yes
Churn

```
In [596]: message = (" * Dataset is not balanced with %.1f%% of the cases negative and %.1f%% positive.\n" %
                     (df1[0]/df1.sum()*100, df1[1]/df1.sum()*100) )
          display(Markdown(message))
```

- Dataset is not balanced with 85.5% of the cases negative and 14.5% positive.

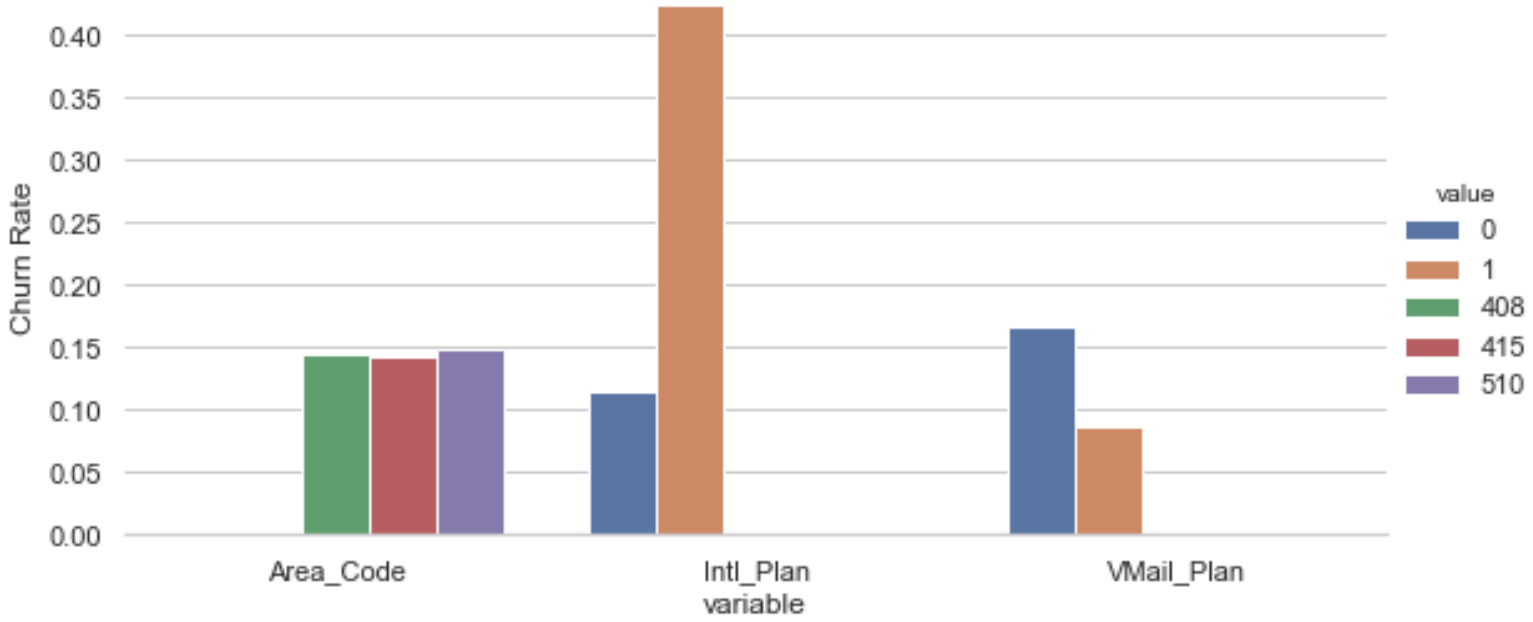
1.2.2 Attributes

```
In [597]: df.agg("nunique")
```

```
Out[597]: State                51
          Account_Length      212
          Area_Code             3
          Phone              3333
          Intl_Plan             2
          VMail_Plan            2
          VMail_Message        46
          Day_Mins           1667
          Day_Calls           119
          Day_Charge         1667
          Eve_Mins           1611
          Eve_Calls           123
          Eve_Charge         1440
          Night_Mins         1591
          Night_Calls         120
          Night_Charge        933
          Intl_Mins           162
          Intl_Calls           21
          Intl_Charge         162
          Cust_Serv_Calls      10
          Churn                 2
          dtype: int64
```

Looking at the categorical variables with a few levels (<=3) we have: **Area_Code**, **Intl_Plan**, and **VMail_Plan**.

```
In [598]: # CODE DELETED
```



State

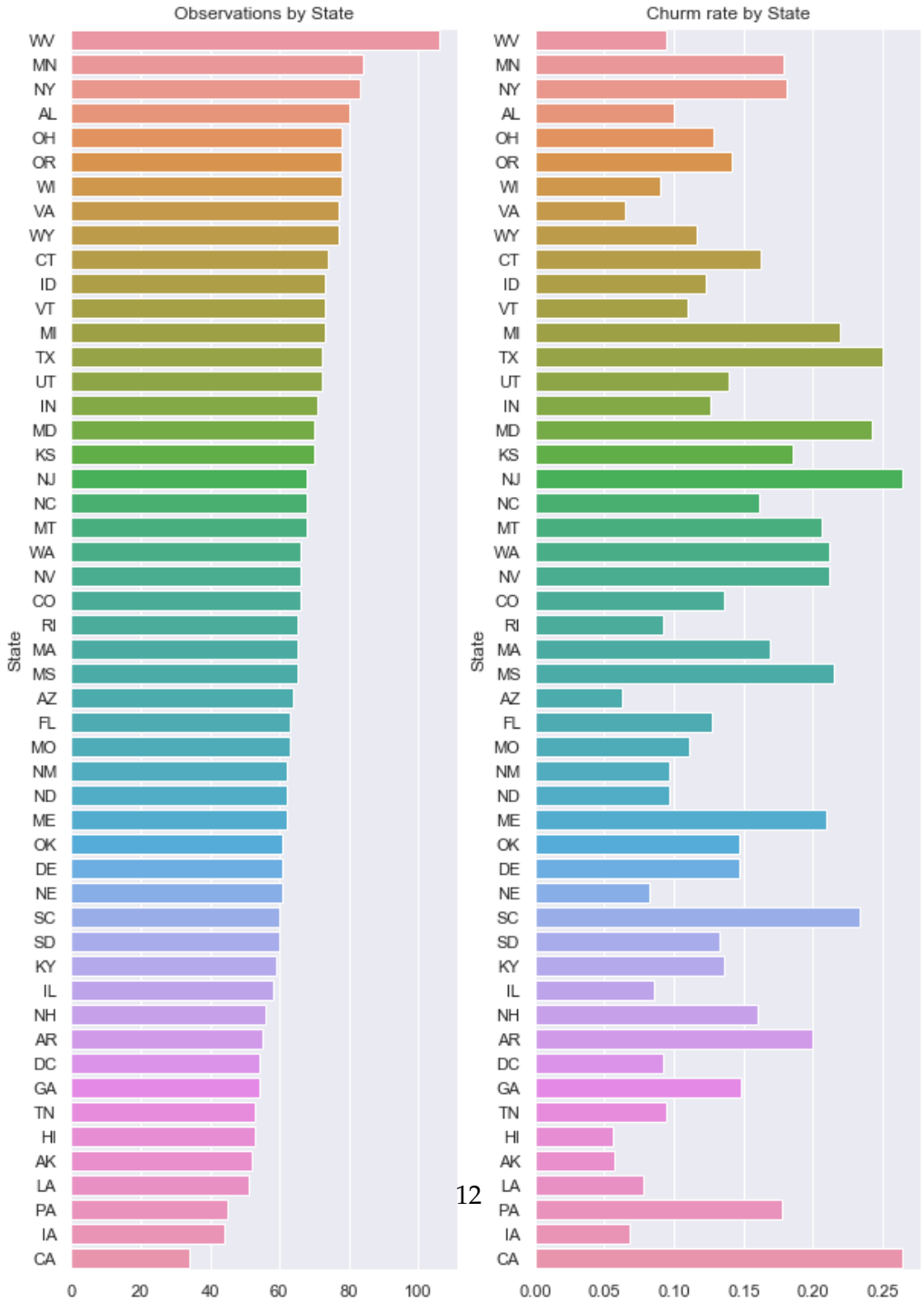
- Categorical (51 levels).
- Indicates the state where the customer lives.

```
In [599]: df.State.describe()
```

```
Out[599]: count      3333
          unique       51
```

```
top          WV
freq        106
Name: State, dtype: object
```

In [600]:
```
# CODE DELETED
```


Observations by State
Churm rate by State
State
WV
MN
NY
AL
OH
OR
WI
VA
WY
CT
ID
VT
MI
TX
UT
IN
MD
KS
NJ
NC
MT
WA
NV
CO
RI
MA
MS
AZ
FL
MO
NM
ND
ME
OK
DE
NE
SC
SD
KY
IL
NH
AR
DC
GA
TN
HI
AK
LA
PA
IA
CA
0
20
40
60
80
100
0.00
0.05
0.10
0.15
0.20
0.25

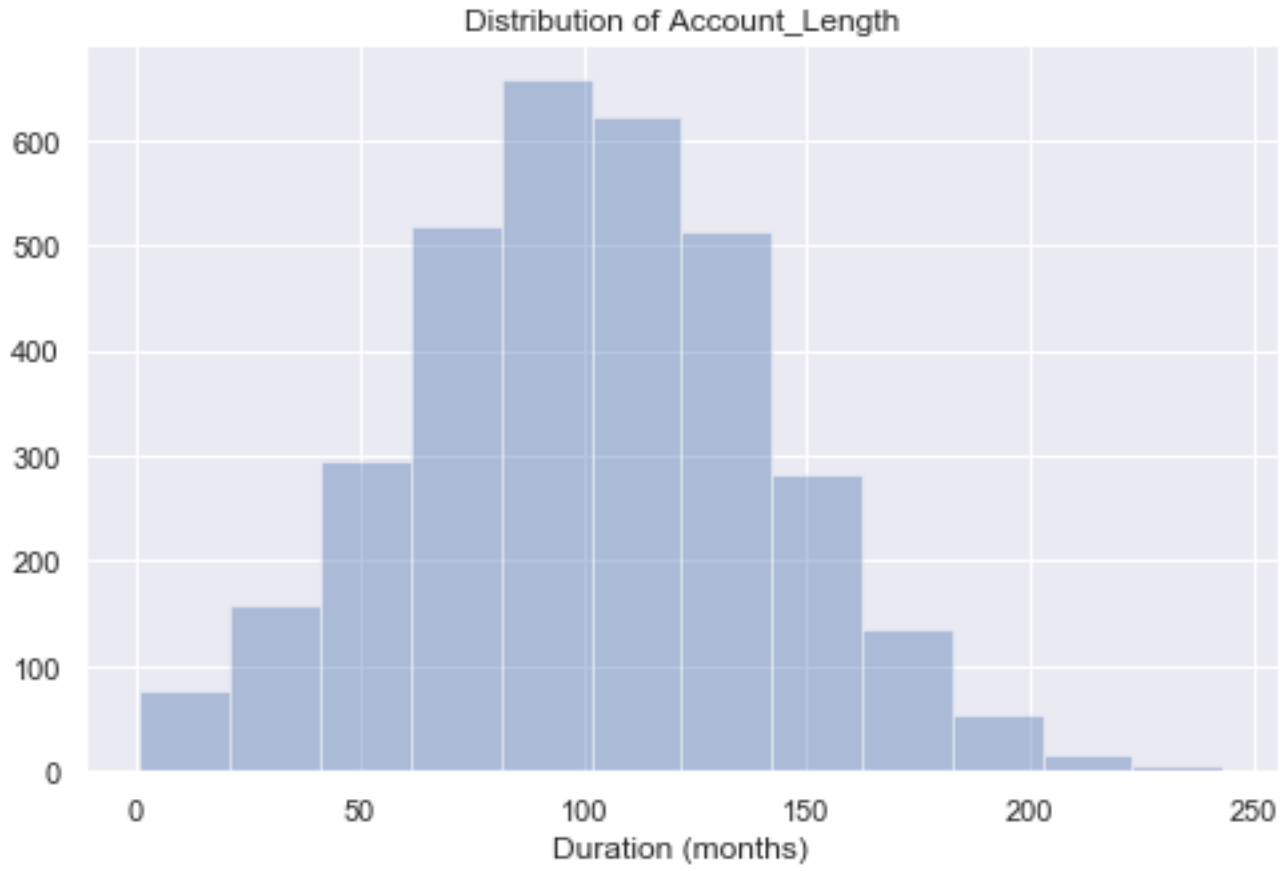
Comments

- 51 distinct values - appearing with frequencies from 34 to 106 inclusive.
- **Churn** is different across states (possibly due to different number of alternative providers).
- But given the granularity - don't want to do a model with **State** as it currently is. A better approach would be list of provides by state and group states based on that.

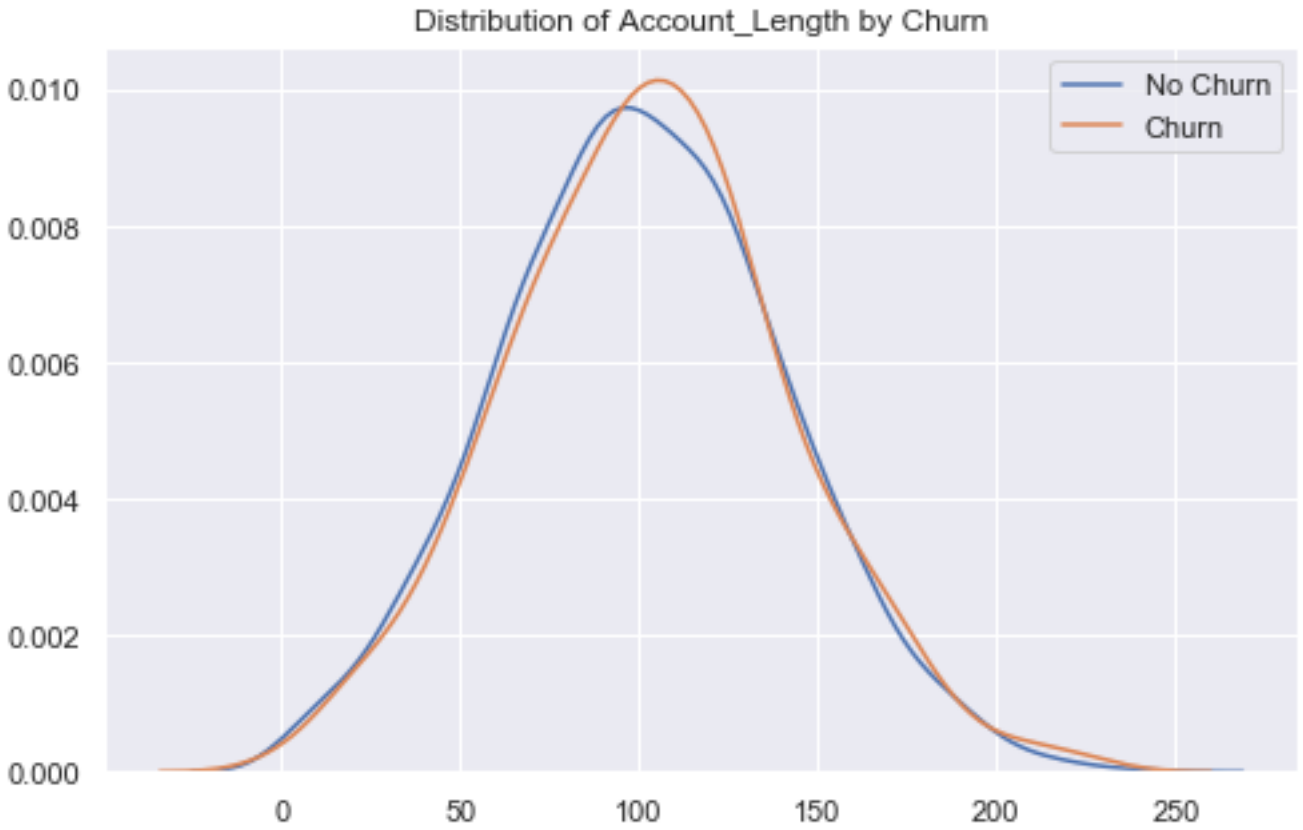
Account_Length

- integer
- length (in months)the account has been active.

```
In [601]: # CODE DELETED
```



In [602]: *# CODE DELETED*



Comments

- **Account_Length** appears unimodal and symmetrical => possibly normally distributed.
- In advance would have thought, distribution would have be skewed with some very old accounts. Not sure as to why not.
- **Account_Length** distribution does not differ with respect to **Churn**.

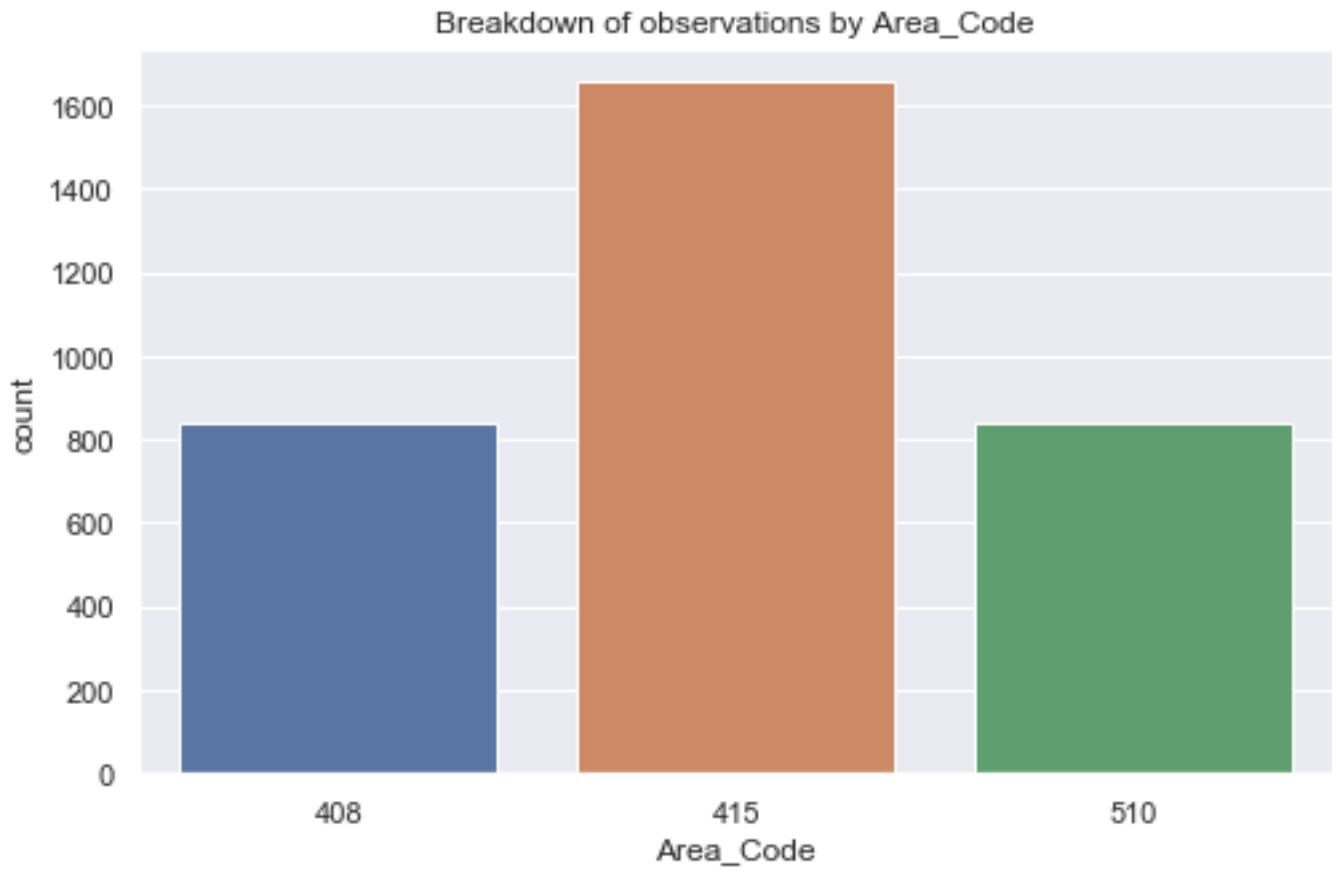
Area_Code

- Categorical (3 levels)

```
In [603]: df.Area_Code.value_counts()
```

```
Out[603]: 415    1655
          510     840
          408     838
          Name: Area_Code, dtype: int64
```

```
In [604]: # CODE DELETED
```



Comments

- Small difference on **Churn** based on **Area_Code**.
- Unlikely to use it in model.

Phone

- Unique for each observation. Not useful as it, may use it for feature extraction later – say extracting the prefix.

Intl_Plan

- binary variable
- the customer has a international plan.

```
In [611]: # CODE DELETED - pivot table
```

```
Out[611]:              len   sum      mean
                     Churn Churn     Churn
          Intl_Plan
          0           3010   346  0.114950
          1            323   137  0.424149
```

```
In [614]: # CODE DELETED - crosstab
```

```
Out[614]: Churn         0    1   All
          Intl_Plan
          0          2664  346  3010
          1           186  137   323
          All        2850  483  3333
```

```
In [615]: # CODE DELETED -
```

- Probability of churning is 42% (137/323) given International plan and 11% (346/3010) given no International plan.

Comments

- From graph above and from crosstab, **Intl_Plan** is linked to **Churn**

VMail_Plan

- binary variable
- the customer has a voice mail plan.

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

1.2.3 VMail_Message

- integer variable
- the number of voice mail messages.

```
In [ ]: # CODE DELETED
```

1.3 Exploring Multivariate Relationships

```
In [ ]: # CODE DELETED
```

```
In [616]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

1.4 Classification Models

```
In [ ]: # CODE DELETED
```

1.4.1 Decision Trees

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

```
In [ ]: # CODE DELETED
```

1.5 Conclusions

```
In [618]: # CODE DELETED
```

```
In [619]: # CODE DELETED
```