National Institute of Technology Rourkela, Odisha, India, 769008

Department of Computer Science Engineering

Laborotary-2

(Data Science Laborotary)

NIKHIL NAVIN KARN 121CS1136

```
import numpy as np #Numpy stands for Numerical Python. used for numerical Operations
import pandas as pd #Pandas is used for DataFrame/Database Operations
import warnings
warnings.filterwarnings("ignore")

pd.set_option("display.precision", 2)
```

Read the telecom_churn csv file.

```
#Your answer here
dataset = pd.read_csv('telecom_churn.csv')
```

Show the first 10 rows of the dataset

```
#Your answer here
dataset.head(10)
```

	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 night minutes	Total night calls	Total night charge	Total intl minutes
0	KS	128	415	No	Yes	25.0	265.1	110	45.07	197.4	99	16.78	244.7	91	11.01	10.0
1	OH	107	415	No	Yes	26.0	161.6	123	27.47	195.5	103	16.62	254.4	103	11.45	13.7
2	NJ	137	415	No	No	NaN	243.4	114	41.38	121.2	110	10.30	162.6	104	7.32	12.2
3	OH	84	408	Yes	No	NaN	299.4	71	50.90	61.9	88	5.26	196.9	89	8.86	6.6
4	OK	75	415	Yes	No	0.0	166.7	113	28.34	148.3	122	12.61	186.9	121	8.41	10.1
5	AL	118	510	Yes	No	0.0	NaN	98	37.98	220.6	101	18.75	203.9	118	9.18	6.3
6	MA	121	510	No	Yes	24.0	218.2	88	37.09	348.5	108	29.62	212.6	118	9.57	7.5
7	MO	147	415	Yes	No	0.0	157.0	79	26.69	103.1	94	8.76	211.8	96	9.53	7.1
8	LA	117	408	No	No	0.0	184.5	97	31.37	351.6	80	29.89	215.8	90	9.71	8.7
9	WV	141	415	Yes	Yes	37.0	258.6	84	43.96	222.0	111	18.87	326.4	97	14.69	11.2

Show the last 10 rows of the dataset

```
#Your answer here
dataset.tail(10)
```

	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 night minutes	Total night calls	Total night charge	Tota int minute
3323	IN	117	415	No	No	0.0	118.4	126	20.13	249.3	97	21.19	227.0	56	10.22	13
3324	WV	159	415	No	No	0.0	169.8	114	28.87	197.7	105	16.80	193.7	82	8.72	11
3325	OH	78	408	No	No	0.0	193.4	99	32.88	116.9	88	9.94	243.3	109	10.95	9
3326	OH	96	415	No	No	0.0	106.6	128	18.12	284.8	87	24.21	178.9	92	8.05	14
3327	SC	79	415	No	No	0.0	134.7	98	22.90	189.7	68	16.12	221.4	128	9.96	11
3328	AZ	192	415	No	Yes	36.0	156.2	77	26.55	215.5	126	18.32	279.1	83	12.56	9
3329	WV	68	415	No	No	0.0	231.1	57	39.29	153.4	55	13.04	191.3	123	8.61	9
3330	RI	28	510	No	No	0.0	180.8	109	30.74	288.8	58	24.55	191.9	91	8.64	14
3331	CT	184	510	Yes	No	0.0	213.8	105	36.35	159.6	84	13.57	139.2	137	6.26	5
3332	TN	74	415	No	Yes	25.0	234.4	113	39.85	265.9	82	22.60	241.4	77	10.86	13

Show the dimensions (No. of rows and coulmns) of the dataset

```
#Your answer here
print(dataset.shape)
```

```
(3333, 20)
```

Print all the column names of the dataset

```
#Your answer here
print(dataset.columns)
```

```
Index(['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 night minutes',
       'Total night calls', 'Total night charge', 'Total intl minutes',
       'Total intl calls', 'Total intl charge', 'Customer service calls',
       'Churn'],
      dtype='object')
```

Print general information of the dataset like column, and datatype.

```
#Your answer here
print(dataset.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3333 entries, 0 to 3332
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   State                   3333 non-null   object 
 1   Account length          3333 non-null   int64  
 2   Area code               3333 non-null   int64  
 3   International plan      3333 non-null   object 
 4   Voice mail plan         3333 non-null   object 
 5   Number vmail messages   3328 non-null   float64
 6   Total day minutes       3329 non-null   float64
 7   Total day calls         3333 non-null   int64  
 8   Total day charge        3333 non-null   float64
 9   Total eve minutes       3333 non-null   float64
 10  Total eve calls         3333 non-null   int64  
 11  Total eve charge        3333 non-null   float64
 12  Total night minutes     3333 non-null   float64
 13  Total night calls       3333 non-null   int64  
 14  Total night charge      3333 non-null   float64
 15  Total intl minutes      3333 non-null   float64
 16  Total intl calls        3333 non-null   int64  
 17  Total intl charge       3333 non-null   float64
 18  Customer service calls  3333 non-null   int64  
 19  Churn                   3333 non-null   bool   
dtypes: bool(1), float64(9), int64(7), object(3)
memory usage: 498.1+ KB
None
```

After getting the datatype of every feature, you can see that there is a feature "**churn**" having Dtype as **bool**. Change that Dtype to **int64**.

```
#Your answer here
dataset['Churn'] = dataset['Churn'].astype('int64')
```

Which method shows basic statistical characteristics of each numerical feature (`int64` and `float64` types): number of non-missing values, mean, standard deviation, range, median, 0.25 and 0.75 quartiles.

```
#Your answer here
dataset.describe()
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	Total intl minutes	Total intl calls	Total intl charge
count	3333.00	3333.00	3328.00	3329.00	3333.00	3333.00	3333.00	3333.00	3333.00	3333.00	3333.00	3333.00	3333.00	3333.00	3333.00
mean	101.06	437.18	8.11	179.78	100.44	30.56	200.98	100.11	17.08	200.87	100.11	9.04	10.24	4.48	2.76
std	39.82	42.37	13.70	54.48	20.07	9.26	50.71	19.92	4.31	50.57	19.57	2.28	2.79	2.46	0.75
min	1.00	408.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	23.20	33.00	1.04	0.00	0.00	0.00
25%	74.00	408.00	0.00	143.70	87.00	24.43	166.60	87.00	14.16	167.00	87.00	7.52	8.50	3.00	2.30
50%	101.00	415.00	0.00	179.40	101.00	30.50	201.40	100.00	17.12	201.20	100.00	9.05	10.30	4.00	2.78
75%	127.00	510.00	20.00	216.40	114.00	36.79	235.30	114.00	20.00	235.30	113.00	10.59	12.10	6.00	3.27
max	243.00	510.00	51.00	350.80	165.00	59.64	363.70	170.00	30.91	395.00	175.00	17.77	20.00	20.00	5.40

Now show basic statistics on non-numerical features.

```
#Your answer here
dataset.describe(include=['object', 'bool'])
```

	State	International plan	Voice mail plan
count	3333	3333	3333
unique	51	2	2
top	WV	No	No
freq	106	3010	2411

How to show the basic statistics for categorical (type `object`) and boolean (type `bool`) features?

```
#Your answer here
dataset.describe(include=['object', 'bool'])
```

	State	International plan	Voice mail plan
count	3333	3333	3333
unique	51	2	2
top	WV	No	No
freq	106	3010	2411

Check for Missing Values

How to check for missing values in each column?

```
#Your answer here
dataset.isnull().sum()
```

```
State                   0
Account length          0
Area code               0
```

```
International plan        0
Voice mail plan           0
Number vmail messages     5
Total day minutes         4
Total day calls           0
Total day charge          0
Total eve minutes         0
Total eve calls           0
Total eve charge          0
Total night minutes       0
Total night calls         0
Total night charge        0
Total intl minutes        0
Total intl calls          0
Total intl charge         0
Customer service calls    0
Churn                     0
dtype: int64
```

Sorting

How to sort a dataFrame using one of the variables (i.e columns) in ascending order and also in descending order?

```
#Your answer here
dataset.sort_values(by = 'Total eve minutes', ascending = True) # For ascending order
# dataset.sort_values(by = 'Total eve minutes', ascending = False) # For descending order
```

	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 night minutes	Total night calls	Total night charge	Tota int minute
2932	UT	97	415	No	No	0.0	209.2	134	35.56	0.0	0	0.00	175.4	94	7.89	11
32	LA	172	408	No	No	0.0	212.0	121	36.04	31.2	115	2.65	293.3	78	13.20	12
533	OK	125	415	No	Yes	36.0	201.3	117	34.22	42.2	78	3.59	125.7	104	5.66	5
889	MN	103	415	No	No	0.0	198.5	112	33.75	42.5	90	3.61	179.2	124	8.06	12
821	MN	80	415	No	No	0.0	105.8	110	17.99	43.9	88	3.73	189.6	87	8.53	13
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1601	AR	99	510	Yes	No	0.0	242.3	102	41.19	350.9	102	29.83	163.1	93	7.34	11
8	LA	117	408	No	No	0.0	184.5	97	31.37	351.6	80	29.89	215.8	90	9.71	8
2551	MD	102	415	No	No	0.0	129.5	56	22.02	354.2	118	30.11	145.5	93	6.55	10
2331	IN	46	415	No	Yes	34.0	191.4	102	32.54	361.8	96	30.75	147.5	132	6.64	7
2732	NC	130	408	Yes	No	0.0	216.2	106	36.75	363.7	86	30.91	126.7	123	5.70	16

3333 rows × 20 columns

How to sort a dataFrame using multiple variables (i.e columns) in ascending order and also in descending order?

```
#Your answer here
dataset.sort_values(by=['Churn', 'Total day calls'], ascending=[True, False])
```

	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 night minutes	Total night calls	Total night charge	Tota int minute
468	AZ	86	415	No	Yes	32.0	70.9	163	12.05	166.7	121	14.17	244.9	105	11.02	11
1460	MT	80	415	No	No	0.0	198.1	160	33.68	156.7	87	13.32	182.1	76	8.19	9
315	MA	39	408	No	No	0.0	60.4	158	10.27	306.2	120	26.03	123.9	46	5.58	12
1057	WV	86	415	No	Yes	38.0	123.0	158	20.91	133.9	119	11.38	138.2	103	6.22	13
2392	WY	90	510	No	No	0.0	125.4	158	21.32	269.1	83	22.87	238.6	103	10.74	11
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1346	PA	106	408	Yes	No	0.0	133.7	45	22.73	187.8	107	15.96	181.9	89	8.19	10
2884	UT	170	415	No	No	0.0	285.7	44	48.57	167.5	144	14.24	260.0	97	11.70	8
2964	OR	99	408	No	No	0.0	256.4	44	43.59	214.5	105	18.23	233.7	75	10.52	7
1322	DE	2	415	Yes	No	0.0	132.1	42	22.46	138.9	88	11.81	192.6	119	8.67	9
1345	SD	98	415	No	No	0.0	0.0	0	0.00	159.6	130	13.57	167.1	88	7.52	6

3333 rows × 20 columns

Indexing and retrieving data

What is the proportion of churned users in our dataframe?

```
#Your answer here
no_of_churn = dataset[dataset['Churn']==1].shape[0]
print("No of churn users:", no_of_churn)
percentage_of_churn = (no_of_churn/dataset.shape[0])*100
print("Percentage of churn users:", round(percentage_of_churn, 2), "%")
dataset[dataset['Churn']==1]
```

```
No of churn users: 483
Percentage of churn users: 14.49 %
```

	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 night minutes	Total night calls	Total night charge	Tota int minute
10	IN	65	415	No	No	0.0	129.1	137	21.95	228.5	83	19.42	208.8	111	9.40	12
15	NY	161	415	No	No	0.0	332.9	67	56.59	317.8	97	27.01	160.6	128	7.23	5
21	CO	77	408	No	No	0.0	62.4	89	10.61	169.9	121	14.44	209.6	64	9.43	5
33	AZ	12	408	No	No	0.0	249.6	118	42.43	252.4	119	21.45	280.2	90	12.61	11
41	MD	135	408	Yes	Yes	41.0	173.1	85	29.43	203.9	107	17.33	122.2	78	5.50	14
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
3301	CA	84	415	No	No	0.0	280.0	113	47.60	202.2	90	17.19	156.8	103	7.06	10
3304	IL	71	510	Yes	No	0.0	186.1	114	31.64	198.6	140	16.88	206.5	80	9.29	13
3320	GA	122	510	Yes	No	0.0	140.0	101	23.80	196.4	77	16.69	120.1	133	5.40	9
3322	MD	62	408	No	No	0.0	321.1	105	54.59	265.5	122	22.57	180.5	72	8.12	11
3323	IN	117	415	No	No	0.0	118.4	126	20.13	249.3	97	21.19	227.0	56	10.22	13

483 rows × 20 columns

What are average values of numerical features for churned users?

```
#Your answer here
for i in dataset.columns:
    if dataset[i].dtype in ['int64', 'float64']:
        print("Mean for the column '",i,"' is:",dataset[dataset['Churn']==1][i].mean())
```

```
Mean for the column ' Account length ' is: 102.66459627329192
Mean for the column ' Area code ' is: 437.8178053830228
Mean for the column ' Number vmail messages ' is: 5.115942028985507
Mean for the column ' Total day minutes ' is: 206.91407867494823
```

```
Mean for the column ' Total day calls ' is: 101.33540372670808
Mean for the column ' Total day charge ' is: 35.17592132505176
Mean for the column ' Total eve minutes ' is: 212.41014492753624
Mean for the column ' Total eve calls ' is: 100.56107660455487
Mean for the column ' Total eve charge ' is: 18.05496894409938
Mean for the column ' Total night minutes ' is: 205.23167701863352
Mean for the column ' Total night calls ' is: 100.39958592132506
Mean for the column ' Total night charge ' is: 9.23552795031056
Mean for the column ' Total intl minutes ' is: 10.700000000000001
Mean for the column ' Total intl calls ' is: 4.163561076604555
Mean for the column ' Total intl charge ' is: 2.8895445134575573
Mean for the column ' Customer service calls ' is: 2.229813664596273
Mean for the column ' Churn ' is: 1.0
```

How much time (on average) do churned users spend on the phone during daytime?

```
#Your answer here
print("Total avg Min Spend by churned users on the phone during the daytime:",dataset[dataset['Churn']==1]['Total day minutes'].mean())
```

```
Total avg Min Spend by churned users on the phone during the daytime: 206.91407867494823
```

What is the maximum length of international calls among loyal users (`Churn == 0`) who do not have an international plan?

```
#Your answer here
dataset[(dataset['Churn']==0) & (dataset['International plan']=='No')]['Total intl minutes'].max()
```

```
18.9
```

How to get the first or the last line of the data frame?

```
#Your answer here
print(dataset.iloc[0]) # First Line of the data frame
print(dataset.iloc[-1]) # Last Line of the data frame
```

```
State                     KS
Account length           128
Area code                415
International plan        No
Voice mail plan          Yes
Number vmail messages     25.0
Total day minutes        265.1
Total day calls            110
Total day charge         45.07
Total eve minutes        197.4
Total eve calls             99
Total eve charge         16.78
Total night minutes      244.7
Total night calls           91
Total night charge       11.01
Total intl minutes        10.0
Total intl calls             3
Total intl charge          2.7
Customer service calls       1
Churn                        0
Name: 0, dtype: object
State                     TN
Account length            74
Area code                415
International plan        No
Voice mail plan          Yes
Number vmail messages     25.0
Total day minutes        234.4
Total day calls            113
Total day charge         39.85
Total eve minutes        265.9
Total eve calls             82
Total eve charge          22.6
Total night minutes      241.4
Total night calls           77
Total night charge       10.86
Total intl minutes        13.7
Total intl calls             4
Total intl charge          3.7
Customer service calls       0
Churn                        0
Name: 3332, dtype: object
```

Applying Functions to Cells, Columns and Rows

How can you apply a function to each column of a DataFrame to find the maximum value in each column?

```
#Your answer here
dataset.max()
```

```
State                      WY
Account length            243
Area code                 510
International plan        Yes
Voice mail plan           Yes
Number vmail messages    51.0
Total day minutes       350.8
Total day calls           165
Total day charge        59.64
Total eve minutes       363.7
Total eve calls           170
Total eve charge        30.91
Total night minutes     395.0
Total night calls         175
Total night charge      17.77
Total intl minutes       20.0
Total intl calls           20
Total intl charge         5.4
Customer service calls      9
Churn                       1
dtype: object
```

How can you filter rows in a DataFrame to select all states that start with the letter "W"?

```
#Your answer here
dataset[dataset['State'].apply(lambda state: state[0]=='W')]
```

	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 night minutes	Total night calls	Total night charge	Tota int minute
9	WV	141	415	Yes	Yes	37.0	258.6	84	43.96	222.0	111	18.87	326.4	97	14.69	11
26	WY	57	408	No	Yes	39.0	213.0	115	36.21	191.1	112	16.24	182.7	115	8.22	9
44	WI	64	510	No	No	0.0	154.0	67	26.18	225.8	118	19.19	265.3	86	11.94	3
49	WY	97	415	No	Yes	24.0	133.2	135	22.64	217.2	58	18.46	70.6	79	3.18	11
54	WY	87	415	No	No	0.0	151.0	83	25.67	219.7	116	18.67	203.9	127	9.18	9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
3278	WI	87	415	No	No	0.0	238.0	97	40.46	164.5	97	13.98	282.5	132	12.71	10
3303	WI	114	415	No	Yes	26.0	137.1	88	23.31	155.7	125	13.23	247.6	94	11.14	11
3319	WY	89	415	No	No	0.0	115.4	99	19.62	209.9	115	17.84	280.9	112	12.64	15
3324	WV	159	415	No	No	0.0	169.8	114	28.87	197.7	105	16.80	193.7	82	8.72	11
3329	WV	68	415	No	No	0.0	231.1	57	39.29	153.4	55	13.04	191.3	123	8.61	9

327 rows × 20 columns

How can you replace values in a DataFrame column, such as changing the values of "International plan" and "Voice mail plan" from "Yes" or "No" to True or False?

```
#Your answer here
dataset['International plan'] = dataset['International plan'].map({'Yes': 'True', 'No': 'False'})
```

Grouping

In general, grouping data in Pandas works as follows:

```
df.groupby(by=grouping_columns)[columns_to_show].function()
```

1. First, the `groupby` method divides the `grouping_columns` by their values. They become a new index in the resulting dataframe.
2. Then, columns of interest are selected (`columns_to_show`). If `columns_to_show` is not included, all non groupby clauses will be included.
3. Finally, one or several functions are applied to the obtained groups per selected columns.

How can you group data by a specific column (e.g., "Churn") and then display statistical summaries (e.g., mean, standard deviation, etc.) for selected columns such as "Total day minutes," "Total eve minutes," and "Total night minutes"?

```
#Your answer here
cols = ['Total day minutes', 'Total eve minutes', 'Total night minutes']
for i in cols:
    print("For column '",i,"'")
    print("Mean: ")
    print(dataset.groupby(by='Churn')[i].mean())

    print("Standard Deviation: ")
    print(dataset.groupby(by='Churn')[i].std())

    print("Total: ")
    print(dataset.groupby(by='Churn')[i].sum())
    print()
```

```
For column ' Total day minutes '
Mean:
Churn
0    175.18
1    206.91
Name: Total day minutes, dtype: float64
Standard Deviation:
Churn
0    50.2
1    69.0
Name: Total day minutes, dtype: float64
Total:
Churn
0    498556.3
1     99939.5
Name: Total day minutes, dtype: float64

For column ' Total eve minutes '
Mean:
Churn
0    199.04
1    212.41
Name: Total eve minutes, dtype: float64
Standard Deviation:
Churn
0    50.29
1    51.73
Name: Total eve minutes, dtype: float64
Total:
Churn
0    567273.4
1    102594.1
Name: Total eve minutes, dtype: float64

For column ' Total night minutes '
Mean:
Churn
0    200.13
1    205.23
Name: Total night minutes, dtype: float64
Standard Deviation:
Churn
0    51.11
1    47.13
Name: Total night minutes, dtype: float64
Total:
Churn
0    570379.6
1     99126.9
Name: Total night minutes, dtype: float64
```

How can you group data by a specific column (e.g., "Churn") and then apply multiple aggregation functions (e.g., mean, standard deviation, min, max) to selected columns such as "Total day minutes," "Total eve minutes," and "Total night minutes"?

```
#Your answer here
cols = ['Total day minutes', 'Total eve minutes', 'Total night minutes']
for i in cols:
    print("For column '",i,"'")
    print("Mean: ")
    print(dataset.groupby(by='Churn')[i].mean())

    print("Standard Deviation: ")
    print(dataset.groupby(by='Churn')[i].std())

    print("Min: ")
    print(dataset.groupby(by='Churn')[i].min())

    print("Max: ")
    print(dataset.groupby(by='Churn')[i].max())
    print()
```

```
0    50.2
1    69.0
Name: Total day minutes, dtype: float64
Min:
Churn
0    0.0
1    0.0
Name: Total day minutes, dtype: float64
Max:
Churn
0    315.6
1    350.8
Name: Total day minutes, dtype: float64

For column ' Total eve minutes '
Mean:
Churn
0    199.04
1    212.41
Name: Total eve minutes, dtype: float64
Standard Deviation:
Churn
0    50.29
1    51.73
Name: Total eve minutes, dtype: float64
Min:
Churn
0     0.0
1    70.9
Name: Total eve minutes, dtype: float64
Max:
Churn
0    361.8
1    363.7
Name: Total eve minutes, dtype: float64

For column ' Total night minutes '
Mean:
Churn
0    200.13
1    205.23
Name: Total night minutes, dtype: float64
Standard Deviation:
Churn
0    51.11
1    47.13
Name: Total night minutes, dtype: float64
Min:
Churn
0    23.2
1    47.4
Name: Total night minutes, dtype: float64
Max:
Churn
0    395.0
1    354.9
Name: Total night minutes, dtype: float64
```

Summary tables

How can you create a contingency table to observe the distribution of data across two variables, such as "Churn" and "International plan"?

```
#Your answer here
pd.crosstab(dataset['Churn'], dataset['International plan'])
```

International plan	False	True
Churn		
0	2664	186
1	346	137

How can you create a contingency table to observe the distribution of data across two variables, such as "Churn" and "Voice mail plan"?

```
#Your answer here
pd.crosstab(dataset['Churn'], dataset['Voice mail plan'])
```

Voice mail plan	No	Yes
Churn		
0	2008	842
1	403	80

How can you create a pivot table to calculate the average number of day, evening, and night calls grouped by "Area code"?

```
#Your answer here
pd.pivot_table(dataset, ['Total day calls', 'Total eve calls', 'Total night calls'], 'Area code')
```

	Total day calls	Total eve calls	Total night calls
Area code			
408	100.50	99.79	99.04
415	100.58	100.50	100.40
510	100.10	99.67	100.60

DataFrame transformations

How can you calculate the total number of calls ("Total day calls", "Total eve calls", "Total night calls", and "Total intl calls") for all users and add this information as a new column named "Total calls" to the DataFrame?

```
#Your answer here
dataset['Total calls'] = dataset['Total day calls'] + dataset['Total eve calls'] + dataset['Total night calls'] + dataset['Total intl calls'
```

How can you delete specific columns or rows from a DataFrame? **Note:** It should alters the current DataFrame and should not return a new one.

```
#Your answer here
dataset.drop(['Total calls'], axis=1, inplace=True)
```

Visualizing the Data

We'll do this using visual analysis with `matplotlib` and `Seaborn`.

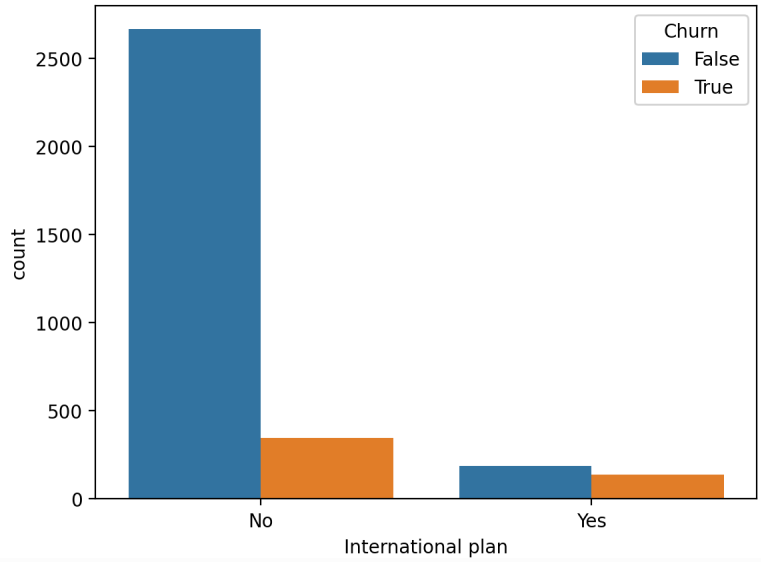
```
# some imports to set up plotting
import matplotlib.pyplot as plt
# !pip install seaborn
import seaborn as sns

# Graphics in retina format are more sharp and legible
%config InlineBackend.figure_format = 'retina'
```

How can you create a count plot to visualize the distribution of the "International plan" variable, with separate bars for each "Churn" category,

```
#Your answer here
sns.countplot(x='International plan', hue='Churn', data=dataset)
```

```
<Axes: xlabel='International plan', ylabel='count'>
```



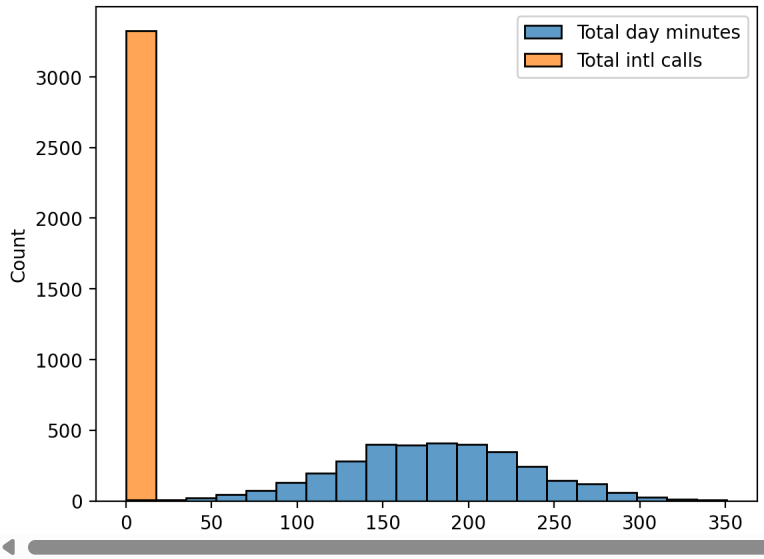
Univariate visualization

Univariate analysis looks at one feature at a time. When we analyze a feature independently, we are usually mostly interested in the *distribution of its values* and ignore other features in the dataset.

Create histograms to visualize the distribution of values for the features "Total day minutes" and "Total intl calls"?

```
#Your answer here
sns.histplot(dataset[['Total day minutes', 'Total intl calls']], bins=20, color='red', alpha=0.7, label='Total')
```

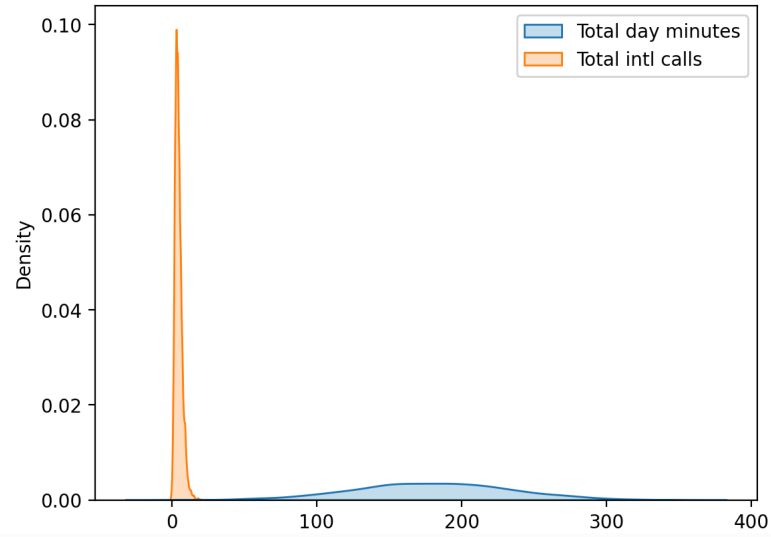
```
<Axes: ylabel='Count'>
```



Create density plots to visualize the distribution of values for the features "Total day minutes" and "Total intl calls"?

```
#Your answer here
sns.kdeplot(dataset[['Total day minutes', 'Total intl calls']], shade=True, label='Total')
```

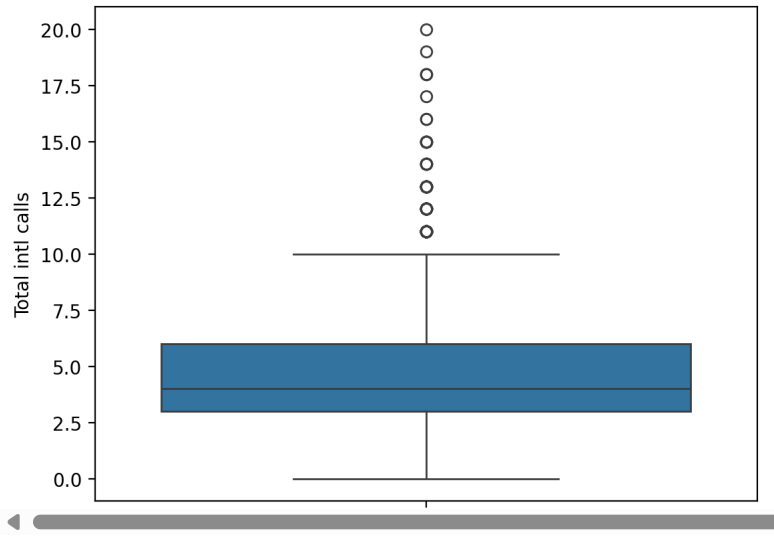
```
<Axes: ylabel='Density'>
```



Create a box plot to visualize the distribution of the "Total intl calls" feature

```
#Your answer here
sns.boxplot(y = 'Total intl calls', data = dataset)
```

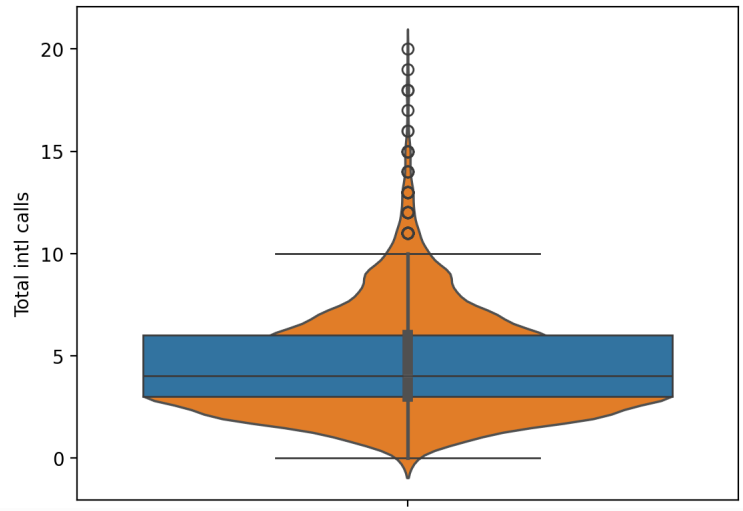
```
<Axes: ylabel='Total intl calls'>
```



Make a comparison between a box plot and a violin plot for visualizing the distribution of the "Total intl calls" feature.

```
#Your answer here
sns.boxplot(y='Total intl calls', data=dataset)
sns.violinplot(y='Total intl calls', data=dataset)
```

```
<Axes: ylabel='Total intl calls'>
```

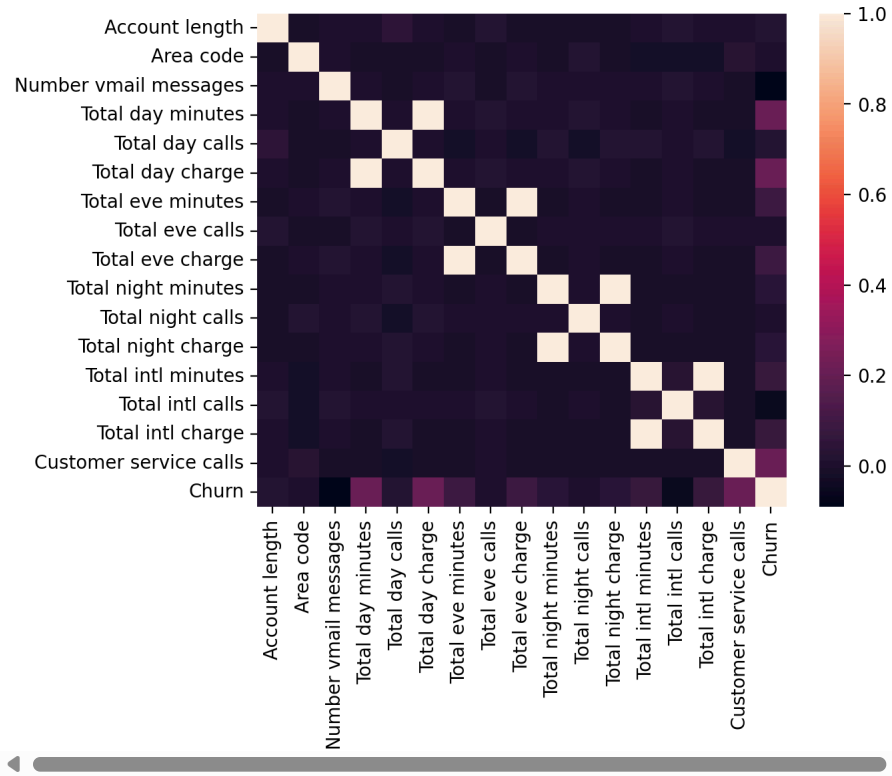


Multivariate visualization

Create a correlation matrix to visualize the relationships among numerical variables in a DataFrame, and display it using a heatmap?

```
#Your answer here
sns.heatmap(dataset.select_dtypes(include=['int64', 'float64','bool']).corr())
```

```
<Axes: >
```

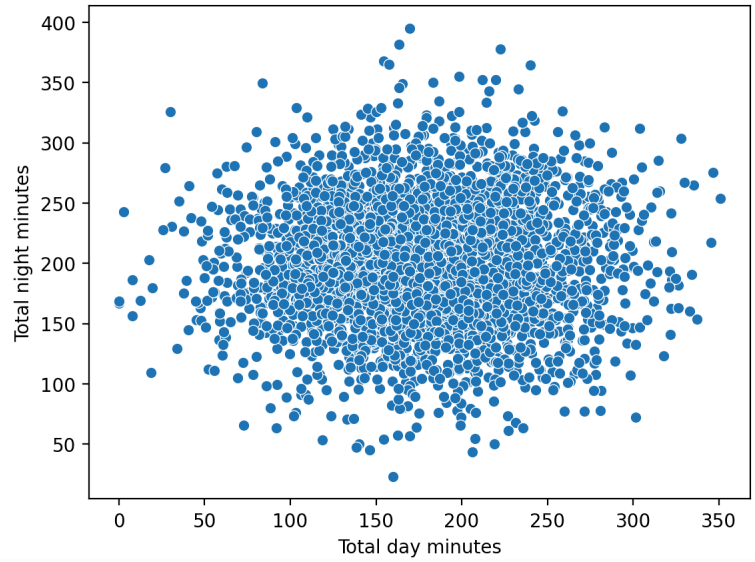


Scatter plot

Create a scatter plot to visualize the relationship between two numerical variables, such as "Total day minutes" and "Total night minutes,"?

```
#Your answer here
sns.scatterplot(x='Total day minutes', y='Total night minutes', data=dataset)
```

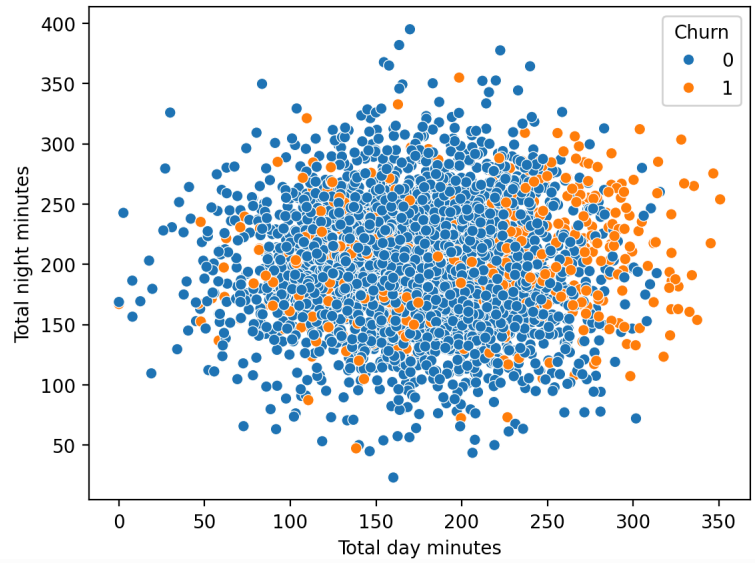
```
<Axes: xlabel='Total day minutes', ylabel='Total night minutes'>
```



Create a scatter plot with different colors for the "Churn" categories to visualize the relationship between "Total day minutes" and "Total night minutes"

```
#Your answer here
sns.scatterplot(x='Total day minutes', y='Total night minutes', data=dataset, hue='Churn')
```

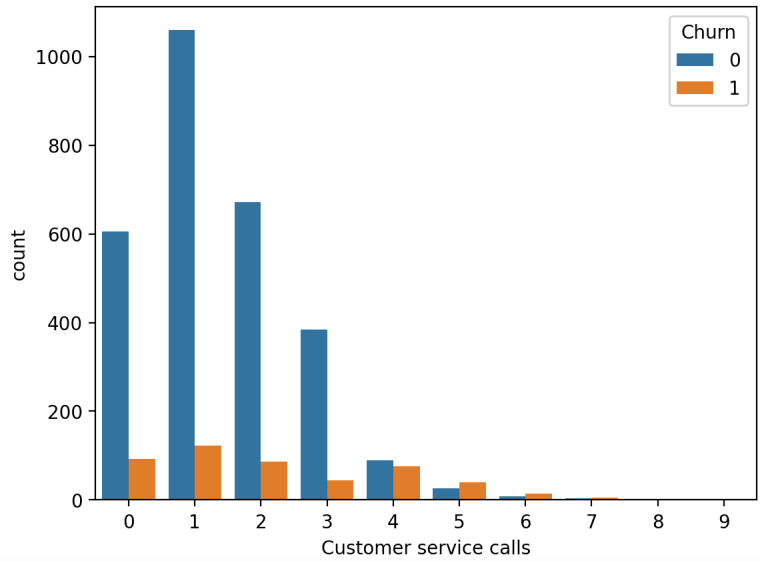
```
<Axes: xlabel='Total day minutes', ylabel='Total night minutes'>
```



Create a count plot to visualize the distribution of the "Customer service calls" variable, with different colors representing the "Churn" categories.

```
#Your answer here
sns.countplot(x='Customer service calls', data=dataset, hue='Churn')
```

```
<Axes: xlabel='Customer service calls', ylabel='count'>
```



Data Transformation

```
from sklearn.preprocessing import StandardScaler
```

Delete specific columns ("Churn", "State") from a DataFrame, convert binary categorical values ("Yes"/"No") into numerical values

```
#Your answer here
dataset.drop(['State', 'Churn'], axis=1, inplace=True)
dataset['International plan'] = dataset['International plan'].map({'True': '1', 'False': '0'})
dataset['Voice mail plan'] = dataset['Voice mail plan'].map({'Yes': '1', 'No': '0'})
scaler = StandardScaler()
dataSet_scaled = scaler.fit_transform(dataset)
dataSet_scaled
```

```
array([[ 0.67648946, -0.52360328, -0.32758048, ..., -0.60119509,
         -0.0856905 , -0.42793202],
       [ 0.14906505, -0.52360328, -0.32758048, ..., -0.60119509,
         1.2411686 , -0.42793202],
       [ 0.9025285 , -0.52360328, -0.32758048, ...,  0.21153386,
         0.69715637, -1.1882185 ],
       ...,
       [-1.83505538,  1.71881732, -0.32758048, ...,  0.61789834,
         1.3871231 ,  0.33235445],
       [ 2.08295458,  1.71881732,  3.05268496, ...,  2.24335625,
        -1.87695028,  0.33235445],
       [-0.67974475, -0.52360328, -0.32758048, ..., -0.19483061,
         1.2411686 , -1.1882185 ]])
```

Now normalize the feature data by subtracting the mean and dividing by the standard deviation, and then convert the scaled data into a DataFrame

```
#Your answer here
dataSet_scaled = (dataSet_scaled - dataSet_scaled.mean(axis = 0))/dataSet_scaled.std(axis = 0)
dataSet_scaled = pd.DataFrame(dataSet_scaled, columns=dataset.columns)
dataSet_scaled.head()
```

	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 night minutes	Total night calls	Total night charge	Total intl minutes	Total intl calls
0	0.68	-0.52	-0.33	1.62	NaN	NaN	0.48	1.57	-0.07	-0.06	-0.07	0.87	-0.47	0.87	-0.09	-0.60
1	0.15	-0.52	-0.33	1.62	NaN	NaN	1.12	-0.33	-0.11	0.14	-0.11	1.06	0.15	1.06	1.24	-0.60
2	0.90	-0.52	-0.33	-0.62	NaN	NaN	0.68	1.17	-1.57	0.50	-1.57	-0.76	0.20	-0.76	0.70	0.21
3	-0.43	-0.69	3.05	-0.62	NaN	NaN	-1.47	2.20	-2.74	-0.61	-2.74	-0.08	-0.57	-0.08	-1.30	1.02
4	-0.65	-0.52	3.05	-0.62	NaN	NaN	0.63	-0.24	-1.04	1.10	-1.04	-0.28	1.07	-0.28	-0.05	-0.60

Splitting a Dataset into Train & Test Set

How can you split a dataset into training and testing sets, with 25% of the data reserved for testing?

```
#Your answer here
dataSet = pd.read_csv(r"telecom_churn.csv")
dataSet['International plan'] = dataSet['International plan'].map({'Yes': '1', 'No': '0'})
dataSet['Voice mail plan'] = dataSet['Voice mail plan'].map({'Yes': '1', 'No': '0'})

print(dataSet.shape)
```

```
(3333, 20)
```

```
#Your answer here
train_index = int(dataSet.shape[0] * 0.75)
print(train_index)
# predict churn
X_train = dataSet.drop(['Churn','State'], axis=1).iloc[:train_index, :]
X_train = X_train
print(X_train.shape)
y_train = dataSet['Churn'].iloc[:train_index]

X_test = dataSet.drop(['Churn','State'], axis=1).iloc[train_index:, :]
y_test = dataSet['Churn'].iloc[train_index:]


print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
2499
(2499, 18)
(2499, 18) (2499,)
(834, 18) (834,)
```

Classification using Machine Learning

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

Train a Decision Tree classifier on the training data and make predictions on the test data

```
#Your answer here
tree = DecisionTreeClassifier(random_state=17)
tree.fit(X_train, y_train)
tree_predictions = tree.predict(X_test)

score = accuracy_score(y_test, tree_predictions)
print("Accuracy: ", score)
```

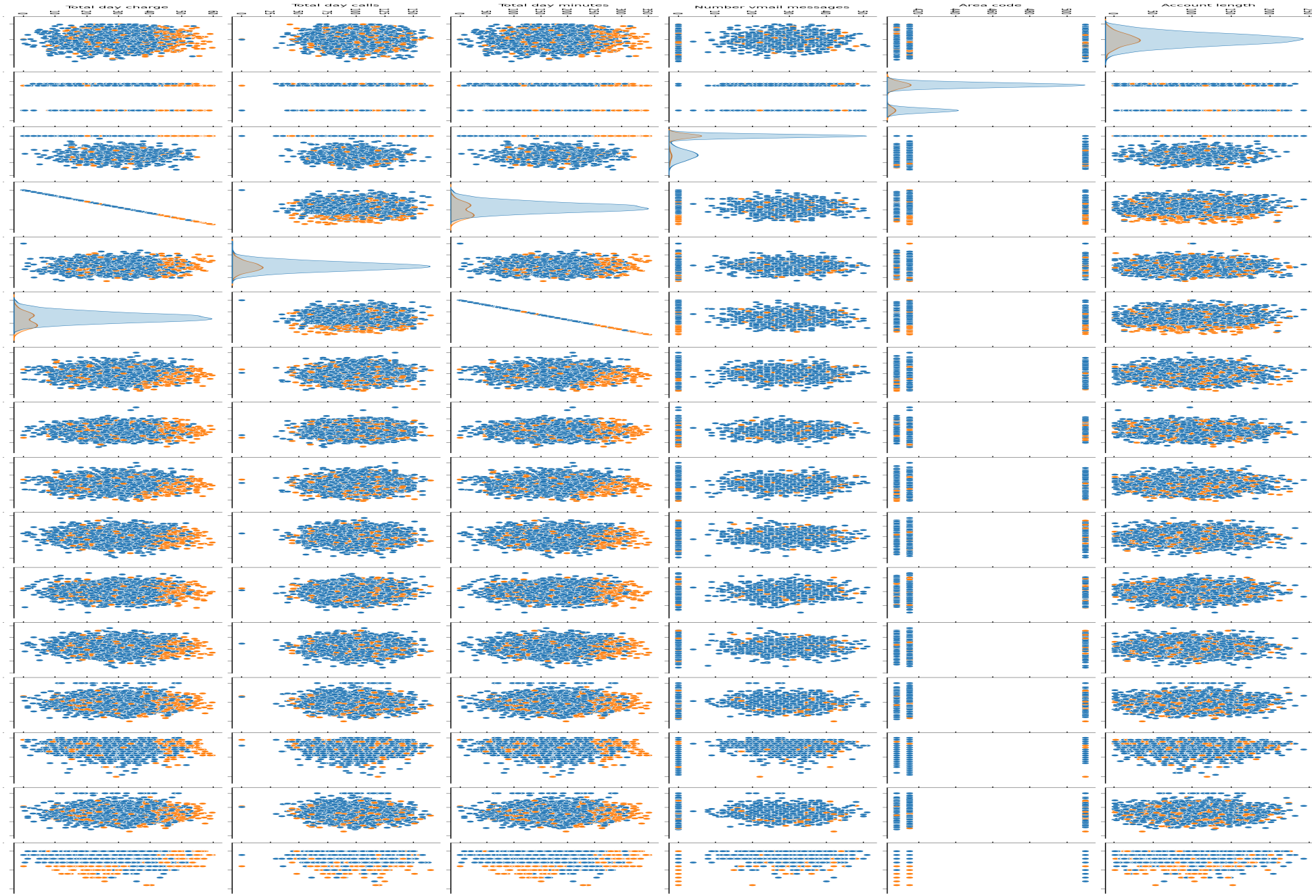
```
Accuracy:  0.9124700239808153
```

Some additional questions

1) If you will be asked to choose a single feature for classification which one you will choose. Justify with analysis.

```
#Your answer here
data = dataSet.drop(['International plan', 'Voice mail plan', 'State'], axis=1)


sns.pairplot(data, hue='Churn')
```


Total day charge
Total day calls
Total day minutes
Number vmail messages
Area code
Account length