Using the map function

A map function is usually used to convert columns into another equivalent columns. It can be used to convert the Categorical data column to a 0 and 1 value column. Map() is a Series Method.

In [28]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
train = pd.read_csv('http://bit.ly/kaggletrain')
```

In [5]:
```
train.head()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [6]:
```
train['sex_num'] = train['Sex'].map({'female':0,'male':1})
```

In [17]:
```
#Just an example to get some specific indexed numbers
train.iloc[10:15][['sex_num','Sex']]
```

Out[17]:

	sex_num	Sex
10	0	female
11	0	female
12	1	male
13	1	male
14	0	female

In [18]:
```
train[['sex_num','Sex']].head(6)
```

Out[18]:

	sex_num	Sex
0	1	male
1	0	female
2	0	female
3	0	female
4	1	male
5	1	male

This is such a nice way and the most usually used method to convert the categorical variable

Using the Apply() function. Using this, a function can be made to apply to all the rows of a column of a dataframe.

Creating a new column in the Train dataframe specially for the last name.

In [24]:
```
train['LastName'] = train['Name'].apply(lambda x:x.split(',')[0]).head()
```

In [25]:
```
train.head()
```

Out[25]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	sex_num	LastName
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	1	Braund
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	0	Cumings
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	0	Heikkinen
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	0	Futrelle
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	1	Allen

In [34]:
```
# Using the apply() function, which affects the function to each of the values in the column.
train['count_name_length'] = train['Name'].apply(len)
```

In [36]:
```
train.head(2)
```

Out[36]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	sex_num	LastName	count_n
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	1	Braund	23
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	0	Cumings	51

In [37]:
```
# apply function of the fare column
train['Fare_approx'] = train['Fare'].apply(np.ceil)
```

In [38]:
```
train.head(2)
```

Out[38]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	sex_num	LastName	count_n
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	1	Braund	23
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	0	Cumings	51

Exploring the data of Alcohol Consumption in various countries

In [41]:
```
drinks = pd.read_csv('http://bit.ly/drinksbycountry')
```

In [42]:
```
drinks.head(3)
```

Out[42]:

	country	beer_servings	spirit_servings	wine_servings	total_litres_of_pure_alcohol	continent
0	Afghanistan	0	0	0	0.0	Asia
1	Albania	89	132	54	4.9	Europe
2	Algeria	25	0	14	0.7	Africa

In [46]:
```
drinks1 = drinks[['beer_servings','spirit_servings','wine_servings','country']]
```

In [47]:
```
drinks1.head(3)
```

Out[47]:

	beer_servings	spirit_servings	wine_servings	country
0	0	0	0	Afghanistan
1	89	132	54	Albania
2	25	0	14	Algeria

In [61]:
```
drinks1[['beer_servings','spirit_servings','wine_servings']].apply(max)
```

Out[61]:
```
beer_servings      376
spirit_servings    438
wine_servings      370
dtype: int64
```

Exploring the countries with the maximum values found in the previous result

In [80]:
```
# Top 10 beer consuming countries. Strange to see Namibia. May be an error.
drinks[['country','beer_servings']].sort_values(by='beer_servings',ascending=False).head(10)
```

Out[80]:

	country	beer_servings
117	Namibia	376
45	Czech Republic	361
62	Gabon	347
65	Germany	346
98	Lithuania	343
135	Poland	343
188	Venezuela	333
81	Ireland	313
129	Palau	306
140	Romania	297

In [63]:
```
drinks[drinks['beer_servings']==376]
```

Out[63]:

	country	beer_servings	spirit_servings	wine_servings	total_litres_of_pure_alcohol	continent
117	Namibia	376	3	1	6.8	Africa

In [65]:
```
drinks[drinks['spirit_servings']==438]
```

Out[65]:

	country	beer_servings	spirit_servings	wine_servings	total_litres_of_pure_alcohol	continent
68	Grenada	199	438	28	11.9	North America

In [66]:
```
drinks[drinks['wine_servings']==370]
```

Out[66]:

	country	beer_servings	spirit_servings	wine_servings	total_litres_of_pure_alcohol	continent
61	France	127	151	370	11.8	Europe

In [73]:
```
# Known countries for the Highest beer consumption
drinks[(drinks['country']=='Czech Republic') | (drinks['country']=='Germany') | (drinks['beer_servings']==376)]
```

Out[73]:

	country	beer_servings	spirit_servings	wine_servings	total_litres_of_pure_alcohol	continent
45	Czech Republic	361	170	134	11.8	Europe
65	Germany	346	117	175	11.3	Europe
117	Namibia	376	3	1	6.8	Africa