Assignment 5 - Exploratory Analysis

July 3, 2019

0.1 Assignment 5

0.1.1 Problem

Consider the churn.csv dataset

Step 1 Estimate the probabilities, the Gini index and the entropy index for the target feature "churn?"

Step 2 Compute the Pearson index of "Int Min" and "Int Calls"

Step 3 Compute the odds ratio of "churn?" and "Area code"

0.1.2 Resolution

```
In [1]: import pandas as pd
        import numpy as np

        data = pd.read_csv('churn.csv')

        data.head()
```

Out[1]:

	State	Account Length	Area Code	Phone	Int'l Plan	VMail Plan	VMail Message	Day Mins	Day Calls	Day Charge	...	Eve Calls	Eve Charge	Night Mins	Night Calls	Night Charge	Intl Mins	Intl Calls	Intl Charge
0	KS	128	415	382-4657	no	yes	25	265.1	110	45.07	...	99	16.78	244.7	91	11.01	10.0	3	2.70
1	OH	107	415	371-7191	no	yes	26	161.6	123	27.47	...	103	16.62	254.4	103	11.45	13.7	3	3.70
2	NJ	137	415	358-1921	no	no	0	243.4	114	41.38	...	110	10.30	162.6	104	7.32	12.2	5	3.29
3	OH	84	408	375-9999	yes	no	0	299.4	71	50.90	...	88	5.26	196.9	89	8.86	6.6	7	1.78
4	OK	75	415	330-6626	yes	no	0	166.7	113	28.34	...	122	12.61						

```
4       186.9          121          8.41       10.1           3          2.73

   CustServ Calls  Churn?
0               1  False.
1               1  False.
2               0  False.
3               2  False.
4               3  False.

[5 rows x 21 columns]
```

After we can compute the absolute and relative frequency of the different occurences in 'Churn?' feature.

```
In [2]: false = data['Churn?'].value_counts()[False]
        true = data['Churn?'].value_counts()[True]
        total = false + true
        pfalse = false/total
        ptrue = true/total
        print(pfalse, ptrue)
```

```
0.8550855085508551 0.14491449144914492
```

We can compute the Gini-Simpson index, probability that 2 entities taken at random from the feature vector represent different classes (this index can be simplified, ie the probability of having equal values is subtracted from the total probability):

$$G = 1 - \sum_{j=1}^{n} p_j^2$$

```
In [3]: G = 1 - (np.power(pfalse,2) + np.power(ptrue,2))
        print("Gini-Simpson Index: ", G)
```

```
Gini-Simpson Index:  0.2478285632343613
```

We can compute the Shannon index that quantifies the uncertainty (entropy or degree of surprise) associated with prediction:

$$E = -\sum_{j=1}^{n} p_j log_2(p_j)$$

```
In [4]: S = -(pfalse*np.log2(pfalse) + ptrue*np.log2(ptrue))
        print("Shannon Index: ",S)
```

```
Shannon Index:  0.5969661117996699
```

In order to compute the Pearson index we must find the mean and variance of the features arrays and compute the covariance with:

$$v_{ij} = cov(a_j, a_k) = \frac{1}{m-1}\sum_{i=1}^{m}(x_{ij} - \mu_j)(x_{ik} - \mu_k)$$

The Pearson index, that expresses the degree of correlation between 2 features (-1.1), is obtained from:

$$r_{ij} = \frac{v_{ij}}{\sigma_j \sigma_k}$$

```
In [5]: m = data.shape[0]
        xj = data['Intl Mins']
        xk = data['Intl Calls']
        meanj = np.mean(xj)
        meank = np.mean(xk)
        varj = np.var(xj)
        vark = np.var(xk)
        cov = np.sum((xj- meanj)*(xk- meank))/m-1
        rjk = cov/(np.sqrt(varj)*np.sqrt(vark))
        print("Pearson index: ", rjk)
```

```
Pearson index:  -0.11327233689577174
```

We can compute the odds ratio, force of association or non-independence between 2 values of binary data:

$$or = \frac{p_{00}p_{11}}{p_{01}p_{10}}$$

```
In [6]: a = data['Churn?']
        b = data['Area Code']
        p00=data.groupby([a,b]).count()['State'][0]/data.shape[0]
        p01=data.groupby([a,b]).count()['State'][1]/data.shape[0]
        p02=data.groupby([a,b]).count()['State'][2]/data.shape[0]
        p10=data.groupby([a,b]).count()['State'][3]/data.shape[0]
        p11=data.groupby([a,b]).count()['State'][4]/data.shape[0]
        p12=data.groupby([a,b]).count()['State'][5]/data.shape[0]
        odds_408 = p00*p11/p01*p10
        odds_415 = p01*p12/p02*p11
        odds_520 = p02*p10/p00*p12

        print("Odds ratio for 408: ", odds_408)
        print("Odds ratio for 415: ", odds_415)
        print("Odds ratio for 520: ", odds_520)
```

```
Odds ratio for 408:  0.0013077713932093844
Odds ratio for 415:  0.005270207835018772
Odds ratio for 520:  0.001370857258301747
```