ML_3_Assignment

March 15, 2019

In [1]:
```
"""I decided to treat this as a classification problem by creating a new binary variab
(did the woman have at least one affair?) and trying to predict the classification for
woman.
The dataset I chose is the affairs dataset that comes with Statsmodels. It was derived
from a survey of women in 1974 by Redbook magazine, in which married women were
asked about their participation in extramarital affairs. More information about the st
is available in a 1978 paper from the Journal of Political Economy.
Description of Variables
The dataset contains 6366 observations of 9 variables:
rate_marriage: woman's rating of her marriage (1 = very poor, 5 = very good)
age: woman's age
yrs_married: number of years married
children: number of children
religious: woman's rating of how religious she is (1 = not religious, 4 = strongly rel
educ: level of education (9 = grade school, 12 = high school, 14 = some college, 16 =
college graduate, 17 = some graduate school, 20 = advanced degree)
occupation: woman's occupation (1 = student, 2 = farming/semi-skilled/unskilled, 3 =
"white collar", 4 = teacher/nurse/writer/technician/skilled, 5 = managerial/business,
professional with advanced degree)

occupation_husb: husband's occupation (same coding as above)
affairs: time spent in extra-marital affairs"""

#Importing basic libraries and loading the data

import numpy as np
import pandas as pd
import statsmodels.api as sm
affairdata=sm.datasets.fair.load_pandas().data
affairdata.head()
```

Out[1]:

	rate_marriage	age	yrs_married	children	religious	educ	occupation	\
0	3.0	32.0	9.0	3.0	3.0	17.0	2.0	
1	3.0	27.0	13.0	3.0	1.0	14.0	3.0	
2	4.0	22.0	2.5	0.0	1.0	16.0	3.0	
3	4.0	37.0	16.5	4.0	3.0	16.0	5.0	
4	5.0	27.0	9.0	1.0	1.0	14.0	3.0	

```
   occupation_husb   affairs
0              5.0  0.111111
1              4.0  3.230769
2              5.0  1.400000
3              5.0  0.727273
4              4.0  4.666666
```

In [2]:
```
# Here we have to predict wheather or not women are having affairs.So lets convet that
#Let's have a look at the data with respect to affair.

affairdata['affair']=(affairdata.affairs>0).astype(int)
affairdata.groupby('affair').mean()
```

Out[2]:
```
        rate_marriage        age  yrs_married  children  religious       educ  \
affair
0            4.329701  28.390679     7.989335  1.238813   2.504521  14.322977
1            3.647345  30.537019    11.152460  1.728933   2.261568  13.972236

        occupation  occupation_husb   affairs
affair
0         3.405286         3.833758  0.000000
1         3.463712         3.884559  2.187243
```

In [3]:
```
# Let's have a look at how the women rated their marriage

affairdata.groupby('rate_marriage').mean()
```

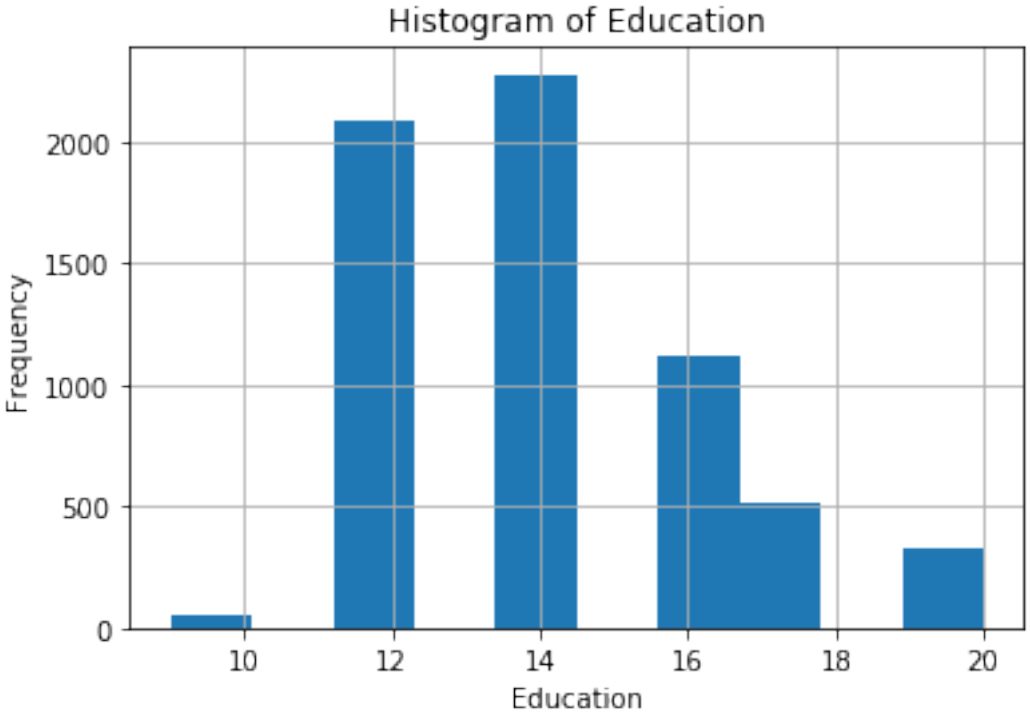
Out[3]:
```
                     age  yrs_married  children  religious       educ  \
rate_marriage
1.0            33.823232    13.914141  2.308081   2.343434  13.848485
2.0            30.471264    10.727011  1.735632   2.330460  13.864943
3.0            30.008056    10.239174  1.638469   2.308157  14.001007
4.0            28.856601     8.816905  1.369536   2.400981  14.144514
5.0            28.574702     8.311662  1.252794   2.506334  14.399776

               occupation  occupation_husb   affairs    affair
rate_marriage
1.0              3.232323         3.838384  1.201671  0.747475
2.0              3.327586         3.764368  1.615745  0.635057
3.0              3.402820         3.798590  1.371281  0.550856
4.0              3.420161         3.835861  0.674837  0.322926
5.0              3.454918         3.892697  0.348174  0.181446
```

We can see that as age,yrs_married and affairs increases,they rated the marriage low and vice versa. Religeous women rated the marriage higher.

In [4]:
```
# Visualizing Education level.
```

```
import matplotlib.pyplot as plt
%matplotlib inline
affairdata.educ.hist()
plt.title('Histogram of Education')
plt.xlabel('Education')
plt.ylabel('Frequency')
```

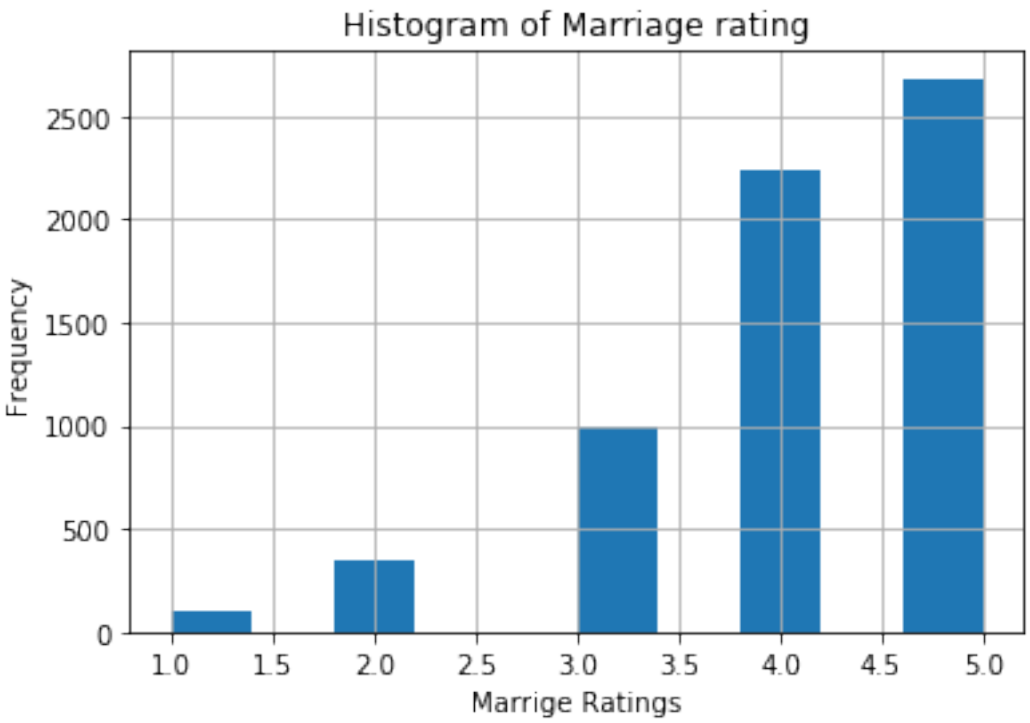
Out[4]: Text(0, 0.5, 'Frequency')



In [5]: *# Visualizing marriage rating*

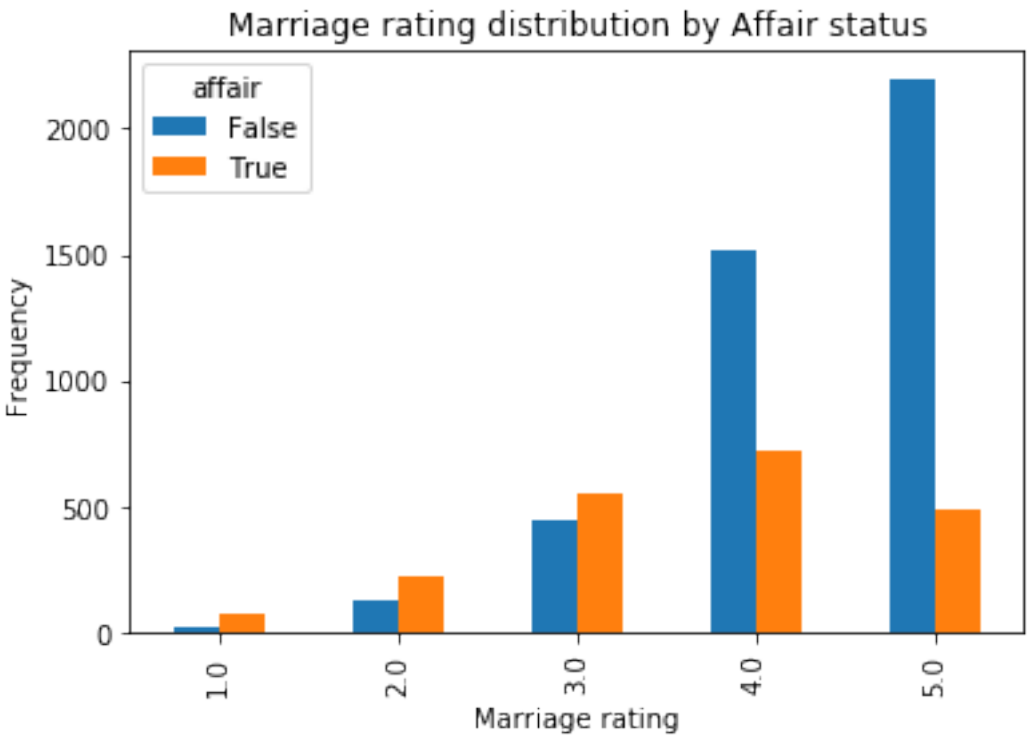
```
affairdata.rate_marriage.hist()
plt.title('Histogram of Marriage rating')
plt.xlabel('Marrige Ratings')
plt.ylabel('Frequency')
```

Out[5]: Text(0, 0.5, 'Frequency')



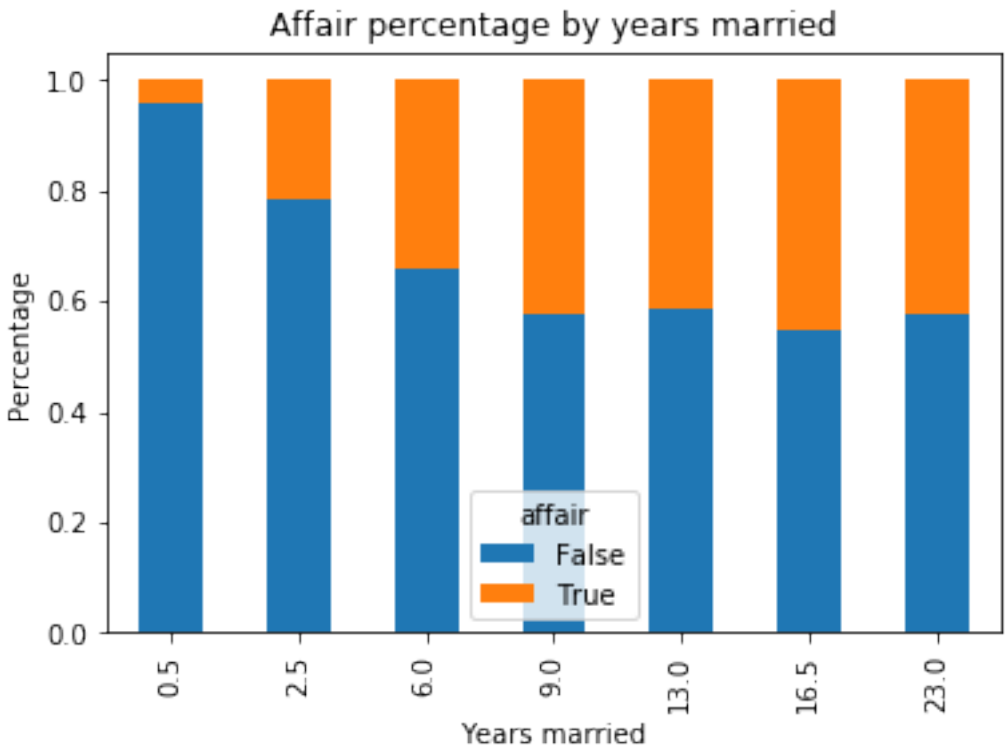
In [6]:
```
# Distribution of marriage rating by affairs.

pd.crosstab(affairdata.rate_marriage,affairdata.affair.astype(bool)).plot(kind='bar')
plt.title('Marriage rating distribution by Affair status')
plt.xlabel('Marriage rating')
plt.ylabel('Frequency')
```

Out[6]:
```
Text(0, 0.5, 'Frequency')
```



In [7]:
```
# Visualizing the affair with yrs narried in terms of percentage.

affair_yrs_married=pd.crosstab(affairdata.yrs_married,affairdata.affair.astype(bool))
affair_yrs_married.div(affair_yrs_married.sum(1).astype(float),axis=0).plot(kind='bar'
plt.title('Affair percentage by years married')
plt.xlabel('Years married')
plt.ylabel('Percentage')
```

Out[7]:
```
Text(0, 0.5, 'Percentage')
```



In [8]:
```
# Importing required libraries and changing the categorical variables into
# dummy variables i.e. occupation and occupation_husb

from sklearn import metrics
from patsy import dmatrices
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
y,X=dmatrices('affair~rate_marriage+age+yrs_married+children+religious+educ+C(occupati
print(X.columns)
```

```
Index(['Intercept', 'C(occupation)[T.2.0]', 'C(occupation)[T.3.0]',
       'C(occupation)[T.4.0]', 'C(occupation)[T.5.0]', 'C(occupation)[T.6.0]',
       'C(occupation_husb)[T.2.0]', 'C(occupation_husb)[T.3.0]',
       'C(occupation_husb)[T.4.0]', 'C(occupation_husb)[T.5.0]',
       'C(occupation_husb)[T.6.0]', 'rate_marriage', 'age', 'yrs_married',
       'children', 'religious', 'educ'],
      dtype='object')
```

In [9]:
```
# The dummy variables have lenthy names.We shoud rename them.

X=X.rename(columns={'C(occupation)[T.2.0]':'occ_2','C(occupation)[T.3.0]':'occ_3','C(oc
print(X.columns)
```

```
Index(['Intercept', 'occ_2', 'occ_3', 'occ_4', 'occ_5', 'occ_6', 'occ_husb_2',
       'occ_husb_3', 'occ_husb_4', 'occ_husb_5', 'occ_husb_6', 'rate_marriage',
       'age', 'yrs_married', 'children', 'religious', 'educ'],
      dtype='object')
```

In [12]:
```
# Conveting y into nd array.
y=np.ravel(y)
y
```

Out[12]:
```
array([1., 1., 1., ..., 0., 0., 0.])
```

In [13]:
```
model=LogisticRegression()
model=model.fit(X, y)
model.score(X, y)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning
  FutureWarning)
```

Out[13]:
```
0.7258875274897895
```

In [14]:
```
# Split into train and test set and apply the model.

X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.3,random_state=0)
model2=LogisticRegression()
model2.fit(X_train,y_train)
y_pred=model2.predict(X_test)
print(metrics.accuracy_score(y_test,y_pred))
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning
  FutureWarning)
```

```
0.7298429319371728
```