1. Supervised Learning Logistic Regression

February 26, 2017

In [2]:
```
import pandas as pd
import numpy as np
```

In [95]:
```
import math

import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns
sns.set_style('whitegrid')
from pandas import DataFrame, Series

from sklearn.linear_model import LogisticRegression

from sklearn.model_selection import train_test_split

from sklearn import metrics

import statsmodels.api as sm
```

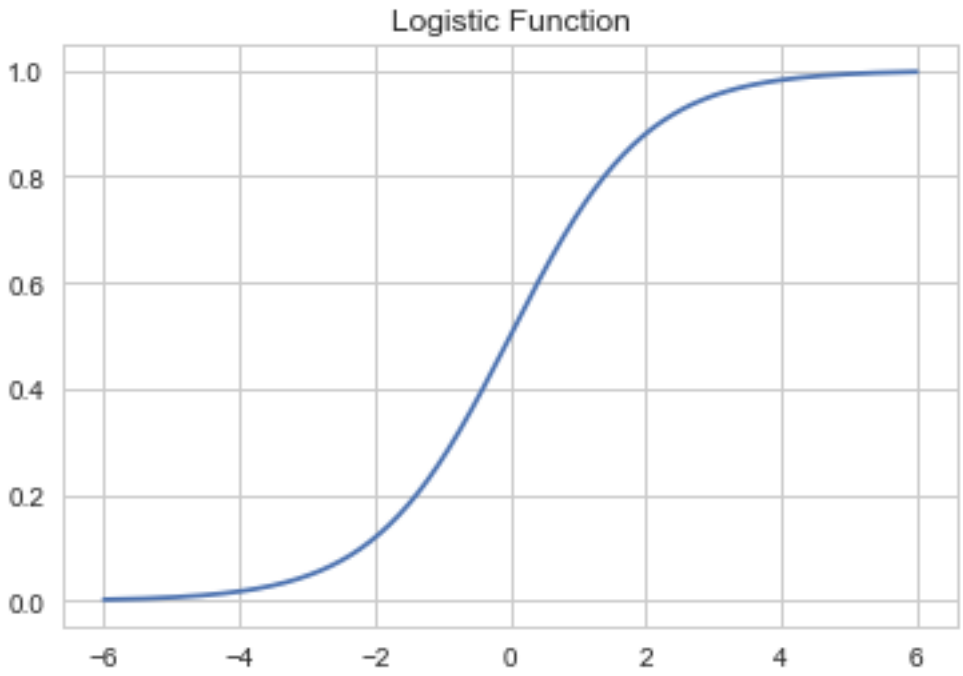
In [6]:
```
def logistic(t):
    return 1.0 / (1 + math.exp((-1.0)*t))

t = np.linspace(-6,6,500)

y = np.array([logistic(ele) for ele in t])

plt.plot(t,y)
plt.title('Logistic Function')
```

Out[6]:
```
<matplotlib.text.Text at 0x112d92150>
```




```
In [7]: df = sm.datasets.fair.load_pandas().data

In [8]: df.head()
```

Out[8]:

	rate_marriage	age	yrs_married	children	religious	educ	occupation	occupation_husb	affairs
0	3.0	32.0	9.0	3.0	3.0	17.0	2.0	5.0	0.111111
1	3.0	27.0	13.0	3.0	1.0	14.0	3.0	4.0	3.230769
2	4.0	22.0	2.5	0.0	1.0	16.0	3.0	5.0	1.400000
3	4.0	37.0	16.5	4.0	3.0	16.0	5.0	5.0	0.727273
4	5.0	27.0	9.0	1.0	1.0	14.0	3.0	4.0	4.666666

```
In [9]: def affair_check(x):
            if x!= 0:
                return 1
            else:
                return 0

In [13]: df['Had_Affair'] = df['affairs'].apply(affair_check)
```

```
In [15]: df.head()
```

Out[15]:

	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	Had_Affair
0	5.0	0.111111	1
1	4.0	3.230769	1
2	5.0	1.400000	1
3	5.0	0.727273	1
4	4.0	4.666666	1

```
In [16]: df.tail()
```

Out[16]:

	rate_marriage	age	yrs_married	children	religious	educ	occupat
6361	5.0	32.0	13.0	2.0	3.0	17.0	
6362	4.0	32.0	13.0	1.0	1.0	16.0	
6363	5.0	22.0	2.5	0.0	2.0	14.0	
6364	5.0	32.0	6.0	1.0	3.0	14.0	
6365	4.0	22.0	2.5	0.0	2.0	16.0	

	occupation_husb	affairs	Had_Affair
6361	3.0	0.0	0
6362	5.0	0.0	0
6363	1.0	0.0	0
6364	4.0	0.0	0
6365	4.0	0.0	0

```
In [20]: df.groupby('Had_Affair').mean()
```

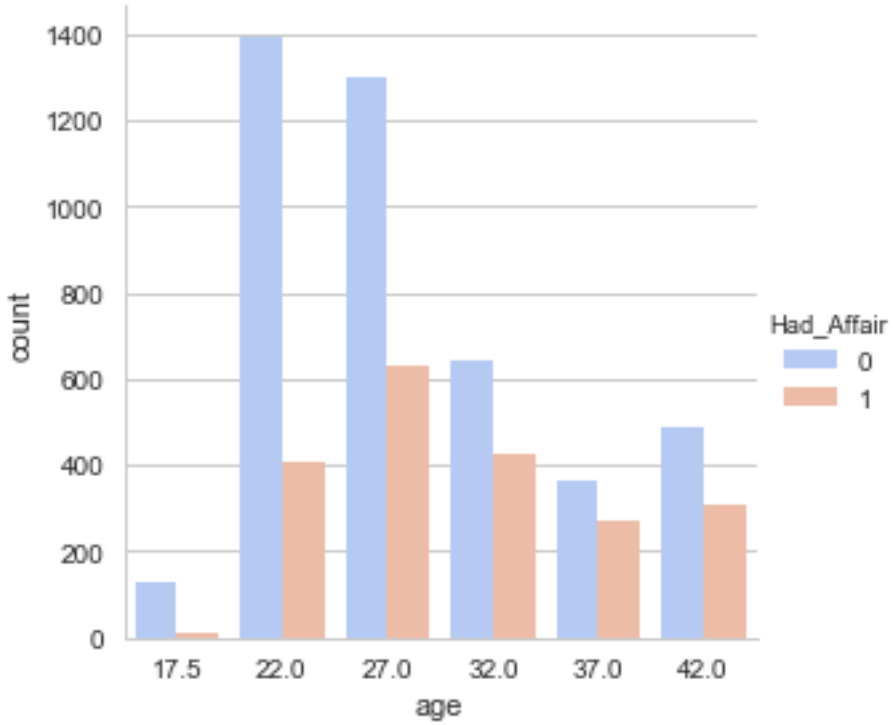
Out[20]:

Had_Affair	rate_marriage	age	yrs_married	children	religious	\
0	4.329701	28.390679	7.989335	1.238813	2.504521	
1	3.647345	30.537019	11.152460	1.728933	2.261568	

Had_Affair	educ	occupation	occupation_husb	affairs
0	14.322977	3.405286	3.833758	0.000000
1	13.972236	3.463712	3.884559	2.187243

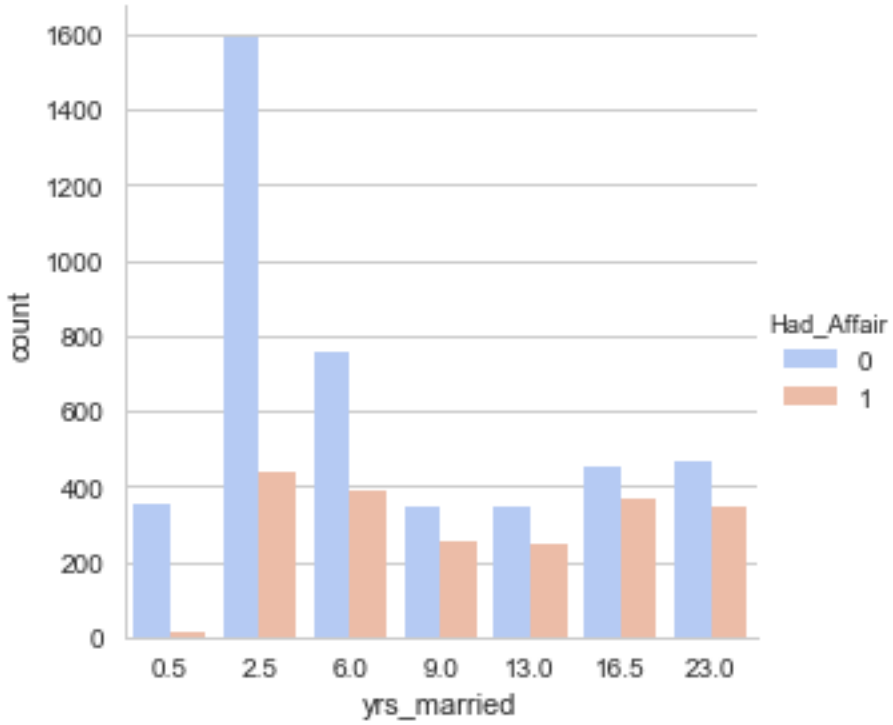
```
In [29]: sns.factorplot('age',kind='count',data= df,hue = 'Had_Affair',palette = 'c
```

```
Out[29]: <seaborn.axisgrid.FacetGrid at 0x116743950>
```



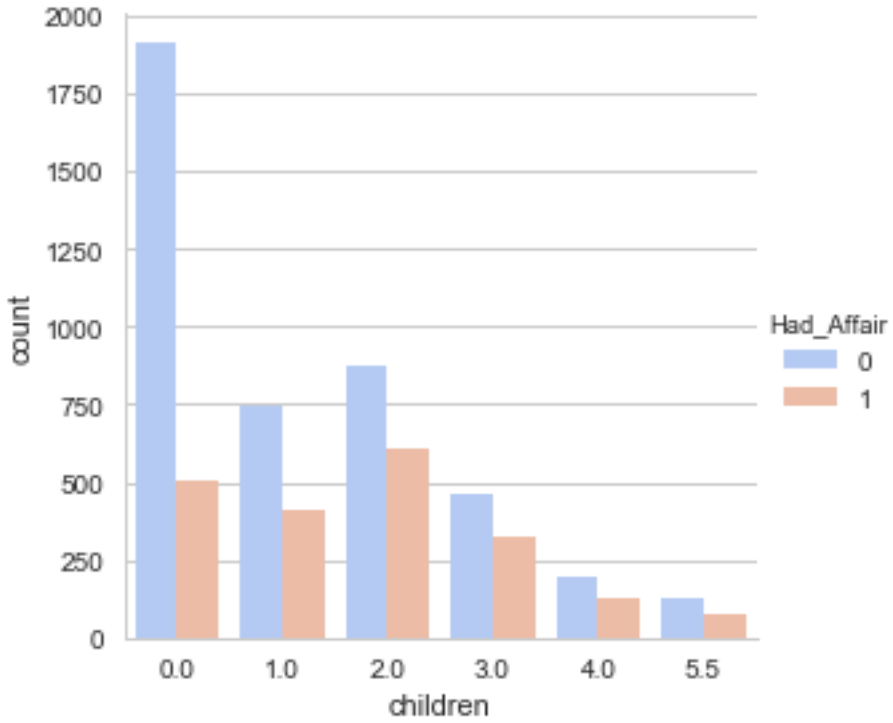

```
In [30]: sns.factorplot('yrs_married',kind='count',data= df,hue = 'Had_Affair',pale
```

```
Out[30]: <seaborn.axisgrid.FacetGrid at 0x112dbdb90>
```



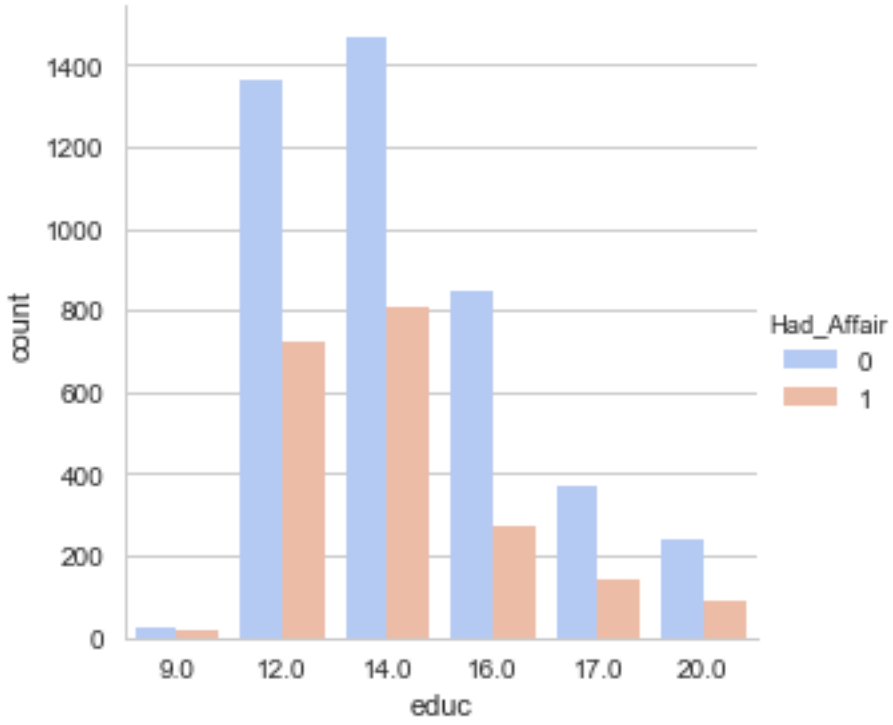
```
In [31]: sns.factorplot('children',kind='count',data= df,hue = 'Had_Affair',palette
```

```
Out[31]: <seaborn.axisgrid.FacetGrid at 0x116a66a50>
```




```
In [32]: sns.factorplot('educ',kind='count',data= df,hue = 'Had_Affair',palette = '
```

```
Out[32]: <seaborn.axisgrid.FacetGrid at 0x116657450>
```




```
In [40]: #dummies variables

         occ_dummies = pd.get_dummies(df['occupation'])

In [46]: hus_occ_dummies = pd.get_dummies(df['occupation_husb'])

In [47]: occ_dummies.head()

Out[47]:    1.0  2.0  3.0  4.0  5.0  6.0
         0    0    1    0    0    0    0
         1    0    0    1    0    0    0
         2    0    0    1    0    0    0
         3    0    0    0    0    1    0
         4    0    0    1    0    0    0

In [48]: hus_occ_dummies.head()

Out[48]:    1.0  2.0  3.0  4.0  5.0  6.0
         0    0    0    0    0    1    0
         1    0    0    0    1    0    0
         2    0    0    0    0    1    0
         3    0    0    0    0    1    0
         4    0    0    0    1    0    0

In [50]: occ_dummies.columns = ['occ1','occ2','occ3','occ4','occ5','occ6']
```

```
In [51]: hus_occ_dummies.columns = ['hocc1','hocc2','hocc3','hocc4','hocc5','hocc6'

In [53]: X = df.drop(['occupation','occupation_husb','Had_Affair'],axis =1)

In [54]: dummies = pd.concat ([occ_dummies,hus_occ_dummies],axis = 1)

In [55]: X.head()
```

Out[55]:

	rate_marriage	age	yrs_married	children	religious	educ	affairs
0	3.0	32.0	9.0	3.0	3.0	17.0	0.111111
1	3.0	27.0	13.0	3.0	1.0	14.0	3.230769
2	4.0	22.0	2.5	0.0	1.0	16.0	1.400000
3	4.0	37.0	16.5	4.0	3.0	16.0	0.727273
4	5.0	27.0	9.0	1.0	1.0	14.0	4.666666

```
In [56]: dummies.head()
```

Out[56]:

	occ1	occ2	occ3	occ4	occ5	occ6	hocc1	hocc2	hocc3	hocc4	hocc5
0	0	1	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
2	0	0	1	0	0	0	0	0	0	0	1
3	0	0	0	0	1	0	0	0	0	0	1
4	0	0	1	0	0	0	0	0	0	1	0

	hocc6
0	0
1	0
2	0
3	0
4	0

```
In [72]: X = pd.concat([X,dummies],axis = 1)

In [73]: X.head()
```

Out[73]:

	rate_marriage	age	yrs_married	children	religious	educ	affairs
0	3.0	32.0	9.0	3.0	3.0	17.0	0.111111
1	3.0	27.0	13.0	3.0	1.0	14.0	3.230769
2	4.0	22.0	2.5	0.0	1.0	16.0	1.400000
3	4.0	37.0	16.5	4.0	3.0	16.0	0.727273
4	5.0	27.0	9.0	1.0	1.0	14.0	4.666666

	occ2	occ3	occ4	...	occ3	occ4	occ5	occ6	hocc1	hocc2	hocc3	\
0	1	0	0	...	0	0	0	0	0	0	0	
1	0	1	0	...	1	0	0	0	0	0	0	
2	0	1	0	...	1	0	0	0	0	0	0	
3	0	0	0	...	0	0	1	0	0	0	0	
4	0	1	0	...	1	0	0	0	0	0	0	

```
   hocc4  hocc5  hocc6
0      0      1      0
1      1      0      0
2      0      1      0
3      0      1      0
4      1      0      0

[5 rows x 29 columns]
```

In [60]:
```
Y = df.Had_Affair
```

In [62]:
```
Y.head()
```

Out[62]:
```
0    1
1    1
2    1
3    1
4    1
Name: Had_Affair, dtype: int64
```

In [63]:
```
Y.tail()
```

Out[63]:
```
6361    0
6362    0
6363    0
6364    0
6365    0
Name: Had_Affair, dtype: int64
```

In [64]:
```
X.head()
```

Out[64]:
```
   rate_marriage   age  yrs_married  children  religious  educ   affairs
0            3.0  32.0          9.0       3.0        3.0  17.0  0.111111
1            3.0  27.0         13.0       3.0        1.0  14.0  3.230769
2            4.0  22.0          2.5       0.0        1.0  16.0  1.400000
3            4.0  37.0         16.5       4.0        3.0  16.0  0.727273
4            5.0  27.0          9.0       1.0        1.0  14.0  4.666666

   occ1  occ2  occ3  occ4  occ5  occ6  hocc1  hocc2  hocc3  hocc4  hocc5
0     0     1     0     0     0     0      0      0      0      0      1
1     0     0     1     0     0     0      0      0      0      1      0
2     0     0     1     0     0     0      0      0      0      0      1
3     0     0     0     0     1     0      0      0      0      0      1
4     0     0     1     0     0     0      0      0      0      1      0

   hocc6
0      0
1      0
2      0
3      0
4      0
```

```
In [74]: X=X.drop('occ1',axis =1 )

In [75]: X=X.drop('hocc1',axis =1 )

In [76]: X=X.drop('affairs',axis =1 )

In [77]: X.head()

Out[77]:    rate_marriage   age  yrs_married  children  religious  educ  occ2  occ3
         0            3.0  32.0          9.0       3.0        3.0  17.0     1     0
         1            3.0  27.0         13.0       3.0        1.0  14.0     0     1
         2            4.0  22.0          2.5       0.0        1.0  16.0     0     1
         3            4.0  37.0         16.5       4.0        3.0  16.0     0     0
         4            5.0  27.0          9.0       1.0        1.0  14.0     0     1

            occ4  occ5  ...    occ2  occ3  occ4  occ5  occ6  hocc2  hocc3  hocc4  \
         0     0     0  ...       1     0     0     0     0      0      0      0
         1     0     0  ...       0     1     0     0     0      0      0      1
         2     0     0  ...       0     1     0     0     0      0      0      0
         3     0     1  ...       0     0     0     1     0      0      0      0
         4     0     0  ...       0     1     0     0     0      0      0      1

            hocc5  hocc6
         0      1      0
         1      0      0
         2      1      0
         3      1      0
         4      0      0

         [5 rows x 26 columns]

In [78]: Y.head()

Out[78]: 0    1
         1    1
         2    1
         3    1
         4    1
         Name: Had_Affair, dtype: int64

In [79]: Y.tail()

Out[79]: 6361    0
         6362    0
         6363    0
         6364    0
         6365    0
         Name: Had_Affair, dtype: int64
```

```
In [81]: Y = np.ravel(Y)  # flattens the array
         Y

Out[81]: array([1, 1, 1, ..., 0, 0, 0])

In [90]: log_model = LogisticRegression()

         log_model.fit(X,Y)
         log_model.score(X,Y)

Out[90]: 0.72588752748978946

In [84]: Y

Out[84]: array([1, 1, 1, ..., 0, 0, 0])

In [85]: Y.mean()

Out[85]: 0.32249450204209867

In [97]: coeff_df = DataFrame(zip(X.columns,np.transpose(log_model.coef_)))

In [98]: coeff_df

Out[98]:
                0                   1
0   rate_marriage   [-0.698728582253]
1             age    [-0.05686194081]
2     yrs_married    [0.104228095566]
3        children   [0.0180776006077]
4       religious   [-0.369346829473]
5            educ  [0.00804196881983]
6            occ2    [0.247674937881]
7            occ3    [0.402952919846]
8            occ4     [0.27314210874]
9            occ5     [0.57206561344]
10           occ6    [0.565606951259]
11          hocc2    [0.115828986658]
12          hocc3    [0.169753288608]
13          hocc4    [0.100663124167]
14          hocc5    [0.112579947473]
15          hocc6    [0.114122753857]
16           occ2    [0.247674937881]
17           occ3    [0.402952919846]
18           occ4     [0.27314210874]
19           occ5     [0.57206561344]
20           occ6    [0.565606951259]
21          hocc2    [0.115828986658]
22          hocc3    [0.169753288608]
23          hocc4    [0.100663124167]
24          hocc5    [0.112579947473]
25          hocc6    [0.114122753857]
```

```
In [99]: # traing and testing data

In [100]: X_train,X_test,Y_train,Y_test = train_test_split(X,Y)

In [104]: log_model2 = LogisticRegression()

          log_model2.fit(X_train,Y_train)

Out[104]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=Tr
                    intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
                    penalty='l2', random_state=None, solver='liblinear', tol=0.0001
                    verbose=0, warm_start=False)

In [105]: class_predict = log_model2.predict(X_test)

In [106]: print metrics.accuracy_score(Y_test,class_predict)

0.718592964824


In [ ]:
```