18-Regression-II-Logistic

November 7, 2016

1 Logistic Regression

```
/Users/markcrovella/anaconda/lib/python3.5/site-packages/sklearn/cross_validation.p
  "This module will be removed in 0.20.", DeprecationWarning)
```

So far we have seen linear regression: a continuous valued observation is estimated as linear (or affine) function of the independent variables.

Today we will look at the following situation.

Imagine that you are observing a binary variable – a 0/1 value.

That is, these could be pass/fail, admit/reject, Democrat/Republican, etc.

You believe that there is some **probability** of observing a 1, and that probability is a function of certain independent variables.

So the key properties of a problem that make it appropriate for logistic regression are:

- What you can observe is a **categorical** variable
- What you want to estimate is a **probability** of seeing a particular value of the categorical variable.

1.1 What is the probability I will be admitted to Grad School?

From http://www.ats.ucla.edu/stat/r/dae/logit.htm:

A researcher is interested in how variables, such as GRE (Graduate Record Exam scores), GPA (grade point average) and prestige of the undergraduate institution, effect admission into graduate school. The response variable, admit/don't admit, is a binary variable.

There are three predictor variables: **gre, gpa** and **rank.** We will treat the variables gre and gpa as continuous. The variable rank takes on the values 1 through 4. Institutions with a rank of 1 have the highest prestige, while those with a rank of 4 have the lowest.

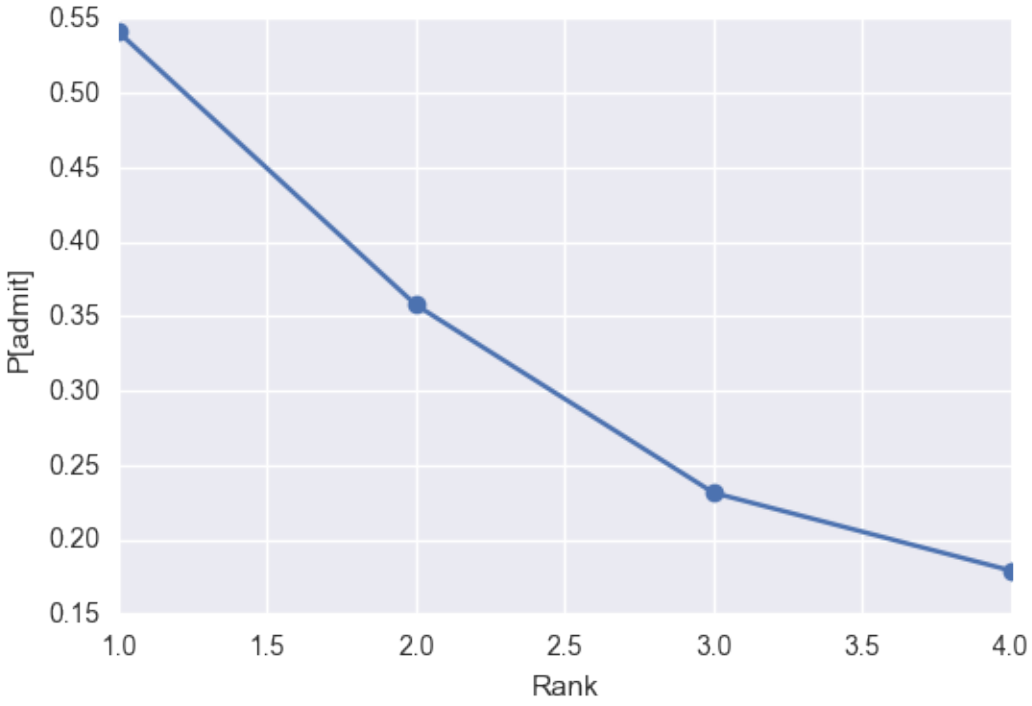
```
In [21]: # read the data in
         # data source: http://www.ats.ucla.edu/stat/data/binary.csv
         df = pd.read_csv('data/ats-admissions.csv')
         df.head(10)
```

```
Out[21]:    admit  gre   gpa  rank
         0      0  380  3.61     3
         1      1  660  3.67     3
```

```
2       1   800   4.00      1
3       1   640   3.19      4
4       0   520   2.93      4
5       1   760   3.00      2
6       1   560   2.98      1
7       0   400   3.08      2
8       1   540   3.39      3
9       0   700   3.92      2
```

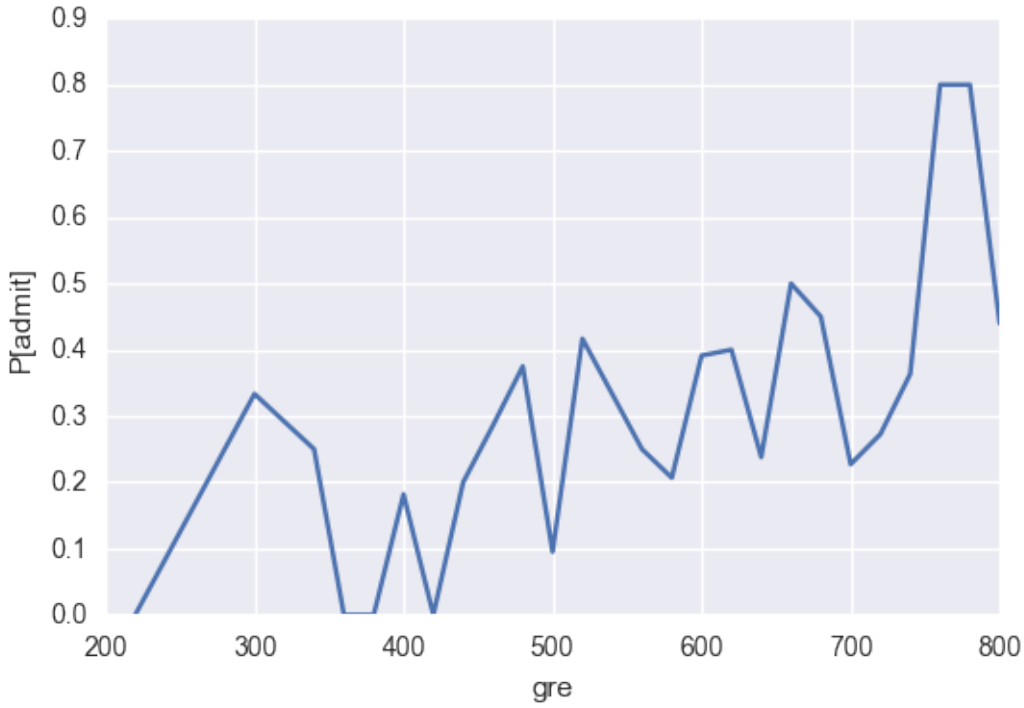
In [33]:
```
plt.plot(df.groupby('rank').mean()['admit'],'o-')
plt.xlabel('Rank')
plt.ylabel('P[admit]')
```

Out[33]: <matplotlib.text.Text at 0x11a57b7f0>

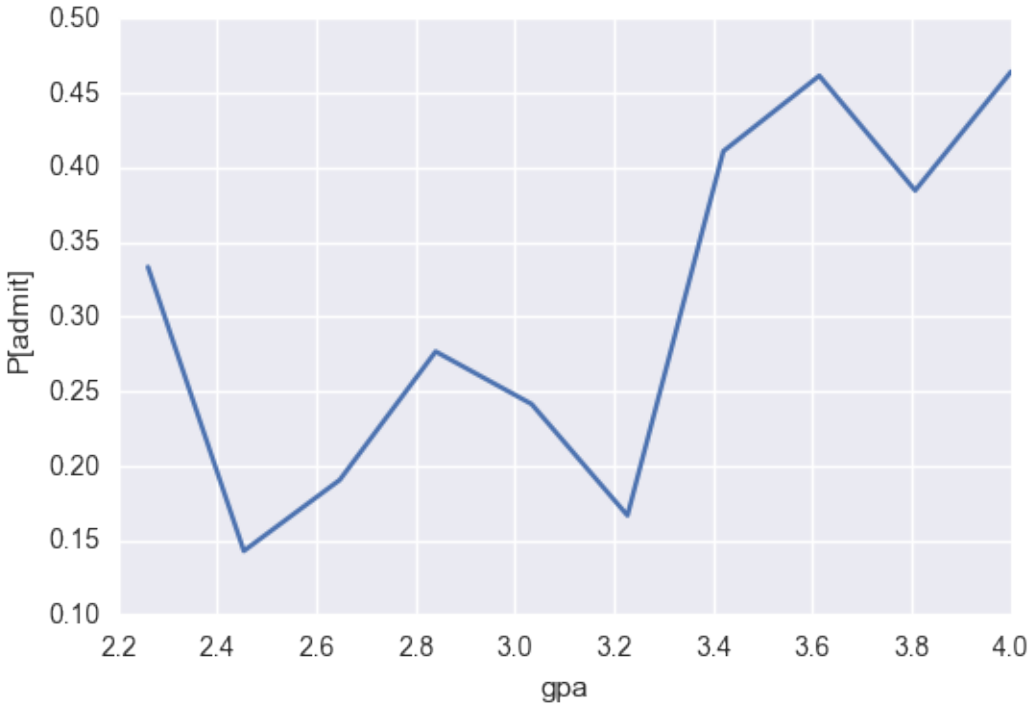


In [68]:
```
plt.plot(df.groupby('gre').mean()['admit'])
plt.xlabel('gre')
plt.ylabel('P[admit]')
```

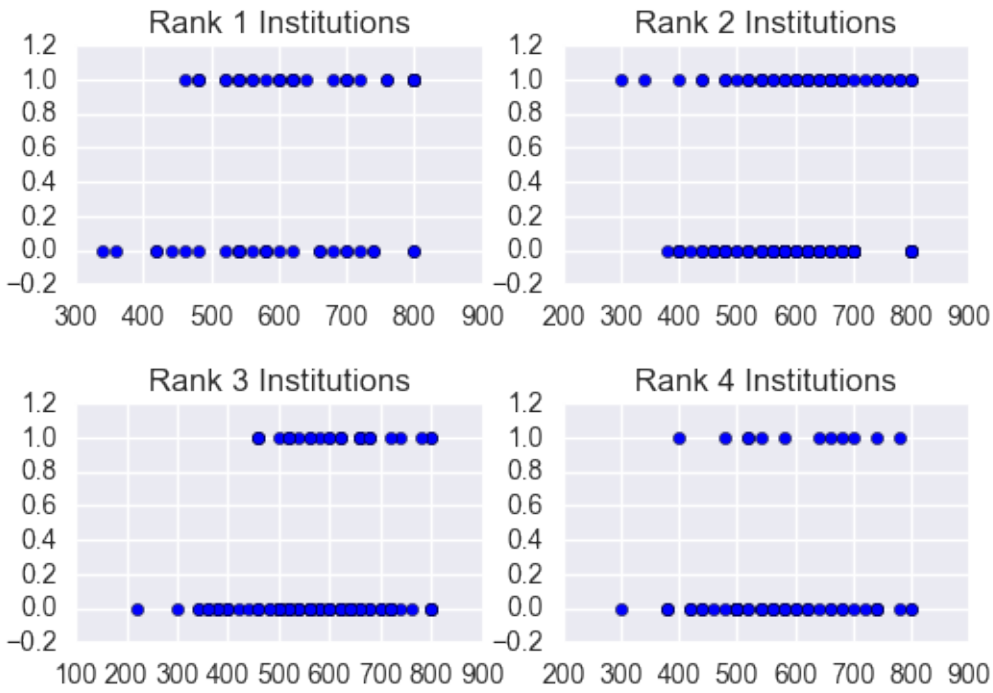
Out[68]: <matplotlib.text.Text at 0x11ada3160>




```
In [76]: bins = np.linspace(df.gpa.min(), df.gpa.max(), 10)
         groups = df.groupby(np.digitize(df.gpa, bins))
         plt.plot(bins,groups.admit.mean())
         plt.xlabel('gpa')
         plt.ylabel('P[admit]')

Out[76]: <matplotlib.text.Text at 0x11b911e80>
```




```
In [20]: train_cols = df.columns[1:]
         train_cols

Out[20]: Index(['gre', 'gpa', 'rank'], dtype='object')

In [40]: df1 = df[df['rank']==1]
         df2 = df[df['rank']==2]
         df3 = df[df['rank']==3]
         df4 = df[df['rank']==4]
         plt.subplot(221)
         plt.scatter(df1['gre'],df1['admit'])
         plt.title('Rank 1 Institutions')
         plt.subplot(222)
         plt.title('Rank 2 Institutions')
         plt.scatter(df2['gre'],df2['admit'])
         plt.subplot(223)
         plt.scatter(df3['gre'],df3['admit'])
         plt.title('Rank 3 Institutions')
         plt.subplot(224)
         plt.title('Rank 4 Institutions')
         plt.scatter(df4['gre'],df4['admit'])
         plt.subplots_adjust(hspace=0.5)
```



1.2 Logistic Regression

Logistic regression is concerned with estimating a **probability.**

However, all that is available are 0/1 observations.

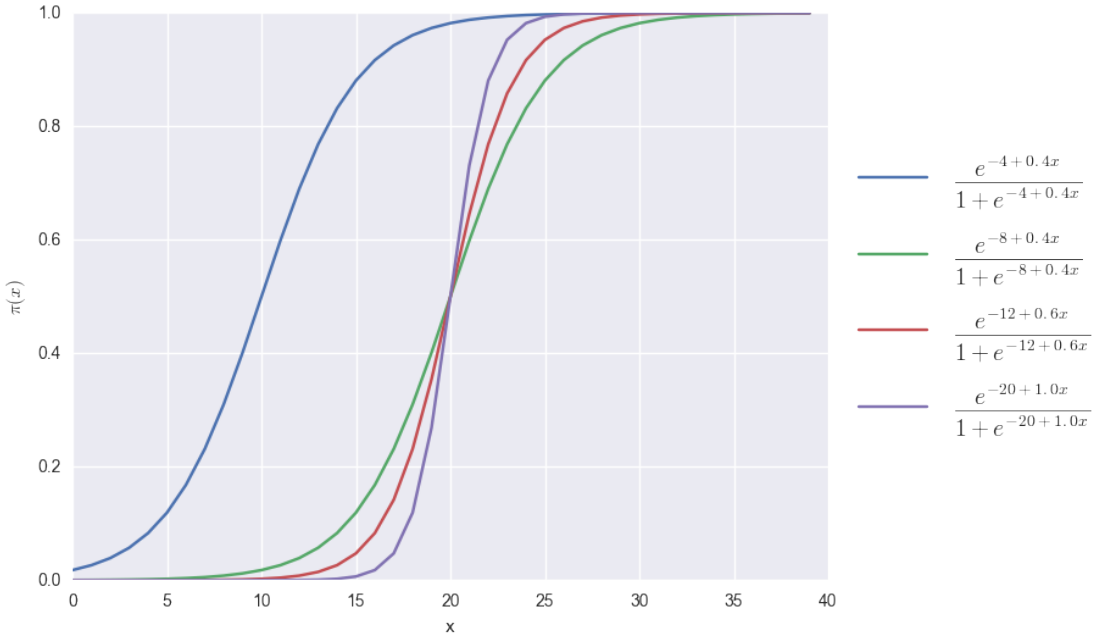
That is, these could be pass/fail, admit/reject, Democrat/Republican, etc.

Logistic regression fits a probability of the following form:

$$\pi(x) = P(y = 1 \mid x) = \frac{e^{\alpha+\beta x}}{1 + e^{\alpha+\beta x}}$$

This is a sigmoid function; when $x \to \infty$, then $\pi(x) \to 1$ and when $x \to -\infty$, then $\pi(x) \to 0$.

```
In [65]: alphas = [-4, -8,-12,-20]
         betas = [0.4,0.4,0.6,1]
         x = np.arange(40)
         fig = plt.figure(figsize=(8, 6))
         ax = plt.subplot(111)

         for i in range(len(alphas)):
             a = alphas[i]
             b = betas[i]
             y = np.exp(a+b*x)/(1+np.exp(a+b*x))
             plt.plot(x,y,label=r"$\frac{e^{%d + %3.1fx}}{1+e^{%d + %3.1fx}}$" % (a
         plt.xlabel('x')
         plt.ylabel('$\pi(x)$')
         _=ax.legend(loc='center left', bbox_to_anchor=(1, 0.5), prop={'size': 20})
```



Parameter β controls how fast $\pi(x)$ raises from 0 to 1
The value of -α/β shows the value of x for which $\pi(x) = 0.5$

Logodds

$$\pi(x) = \frac{e^{\alpha+\beta x}}{1+e^{\alpha+\beta x}}$$

$$1-\pi(x) = \frac{1}{1+e^{\alpha+\beta x}}$$

$$\frac{\pi(x)}{1-\pi(x)} = e^{\alpha+\beta x}$$

$$\text{logit}\left(\pi(x)\right) = \log\left(\frac{\pi(x)}{1-\pi(x)}\right) = \alpha+\beta x$$

For variable $y_i \in \{0,1\}$, the expected valye of y_i given x_i is $\text{logit}\left(\pi(x_i)\right)$

Logistic vs Linear regression **Linear regression:** $y_i = \alpha + \beta x_i$
y_i comes from a normal distribution with standard deviation σ
$y_i = \alpha + \beta x_i$ is the linear predictor
Logistic regression: The expected value of $E[y_i]$ *given* x_i is $\pi_i = \pi(x_i)$ and

$$\text{logit}\left(E[y_i]\right) = \text{logit}\left(\pi(x_i)\right) = \alpha+\beta x_i$$

$y_i \in \{0,1\}$ with $\Pr(y_i = 1) = \pi(x_i)$

Logistic Regression Computation **Input** pairs (x_i, y_i)

Output parameters $\widehat{\alpha}$ and $\widehat{\beta}$ that maximize the likelihood of the data given these parameters for the logistic regression model.

Method Maximum likelihood estimation

```
In [66]: logit = sm.Logit(df['admit'], df[train_cols])

         # fit the model
         result = logit.fit()

Optimization terminated successfully.
         Current function value: 0.586372
         Iterations 5


In [67]: result.summary()

Out[67]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                  Logit Regression Results
         ==============================================================================
         Dep. Variable:                   admit   No. Observations:
         Model:                           Logit   Df Residuals:
         Method:                            MLE   Df Model:
         Date:                Mon, 07 Nov 2016   Pseudo R-squ.:                  0.0
         Time:                        21:49:36   Log-Likelihood:                -23
         converged:                       True   LL-Null:                       -24
                                                 LLR p-value:                1.971
         ==============================================================================
                          coef    std err          z      P>|z|      [95.0% Conf. I
         ------------------------------------------------------------------------------
         gre            0.0015      0.001      1.420      0.155        -0.001      0
         gpa           -0.0042      0.201     -0.021      0.983        -0.398      0
         rank          -0.6695      0.121     -5.527      0.000        -0.907     -0
         ==============================================================================
         """
```

Some more information on performing logistic regression using the statmodels package can be found here:

http://blog.yhathq.com/posts/logistic-regression-and-python.html