ANOVA

Анастасия Миллер

1 мая 2015 г.

1 Дисперсионный анализ ANOVA

Загрузим данные в сессию:

In[]:
```
import numpy as np
import scipy as sp
import pandas as pd
from scipy.stats import f, f_oneway

data = pd.read_table("cities.txt", index_col=0, decimal="," )
data["CITY"]  = data["CITY"].astype("category")
data["STATE"] = data["STATE"].astype("category")
groupeddata = data.groupby("STATE")
n = len(data)
r = len(groupeddata)
```

Зададимся задачей проанализировать разницу в доходах в разных штатах. Тогда нам понадобятся колонки "STATE" и "INCOME", причём "STATE" будет определять подпопуляцию. Предположим, что наблюдения подчиняются модели

$$x_{ij} = \mu + a_i + \varepsilon_{ij},$$

где i - индекс штата, j - индекс города в соответствующем штате.

1.1 Метод наименьших квадратов

Оценим параметры модели по методу наименьших квадратов:

In[]:
```
mu = data["INCOME"].mean()
a = groupeddata["INCOME"].mean() - mu
```

Для того, чтобы понять, является ли штат значимым фактором для уровня доходов, проверим гипотезу $H_0 : a_1 = \cdots = a_n$. В случае, если эта гипотеза будет отвергнута, различие медианного дохода для разных штатов можно будет признать существенным. Посчитаем ошибку нашей оценки:

$$Q_2 = \sum_{i=1}^{r}\sum_{j=1}^{n_i}(x_{ij} - \bar{x}_i)^2$$

In[]:
```
Q2 = []
for state in groupeddata["INCOME"]:
    Q2.extend([(city - state[1].mean())**2 for city in state[1]])
Q2 = np.sum(Q2)
```

Если модель с фиксированными эффектами, которой мы сейчас придерживаемся, хороша, то величина аналогичной ошибки для модели без учёта влияния фактора ($x_{ij} = \mu + \varepsilon_{ij}$) будет существенно больше, чем Q_2. Вычислим ошибку для усечённой модели

$$Q = \sum_{i=1}^{r}\sum_{j=1}^{n_i}(x_{ij} - \bar{x})^2$$

In[]:
```
Q = data["INCOME"].var() * (len(data) - 1)
```

Используя тот факт, что $t = \frac{(Q-Q_2)/(r-1)}{Q_2/(n-r)} \sim F(r-1, n-r)$, мы можем проверить значимость различия:

In[]:
```
t_lsm = (Q - Q2) * (n - r) / (Q2 * (r - 1))
critical_value = f(r-1, n-r).ppf(0.95)
print "критическое значение: ", critical_value
print "значение статистики:", t_lsm
```

```
критическое значение:  1.7058281678
значение статистики: 0.841695848255
```

Следовательно, гипотеза о том, что местоположение не влияет на медианный размер доходов, не может быть отклонена.

1.2 Матрица плана

В обозначениях

$$Y = \begin{bmatrix} x_{11} \\ \vdots \\ x_{1n} \\ \vdots \\ x_{r1} \\ x_{rn} \end{bmatrix}, \beta = \begin{bmatrix} \mu \\ \alpha_1 \\ \vdots \\ \alpha_{r-1} \end{bmatrix}$$

матрицей плана X будет являться $X : Y = X\beta + \epsilon$, где ϵ – вектор случайных отклонений. Построим вектор Y и матрицу X:
(наверное, это можно сделать быстрее и изящнее, но я не придумала ничего проще итерации по рядам данных)

In[]:
```
states = data["STATE"].unique() # list of all available states
states.sort()
last_state = states.tolist()[-1] # last element in array
Y = data.sort("STATE")[["STATE","INCOME"]] # sorting data
# results in naturally grouped rows, so we let ourselves
# forget about groups' order
X = pd.DataFrame(data=np.zeros((n, r)),
                 columns=(["mean"] + states.tolist())[:-1], index=range(1, n+1))
X.loc[Y["STATE"] == last_state, "mean"] = 1
for ((yindex, city), (xindex, xrow)) in zip(Y.iterrows(), X.iterrows()):
    if city["STATE"] != last_state:
        xrow[city["STATE"]] = 1
        xrow["mean"] = 1
    else:
        xrow["mean"] = 1
        for state in states.tolist()[:-1]:
            xrow[state] = -1
X = X.as_matrix() # we do not need information about cities anymore
```

Построим оценку параметров $\hat{\beta} = \left(X^T X\right)^{-} X^T Y$

In[]:
```
beta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y["INCOME"])
```

Значимость гипотезы о том, что модель с факторами лучше усечённой (без влияния факторов) в случае с матрицей планов можно проверить с помощью статистики

$$t = \frac{n-r}{r-1} \frac{R_1^2 - R_0^2}{R_0^2} \sim F(r-1, n-r),$$

где $R_0^2 = \left(Y - X\hat{\beta}\right)^T \left(Y - X\hat{\beta}\right)$ – ошибка полной модели, а $R_1^2 = \left(Y - \tilde{X}\beta^*\right)^T \left(Y - \tilde{X}\beta^*\right)$ – ошибка усечённой модели, в которой под $\beta^* = (\tilde{X}^T\tilde{X})^-\tilde{X}^TY$ имеется в виду оценка усечённой модели с матрицей плана $\tilde{X} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 1 & 0 & \cdots & 0 \end{bmatrix}$.

Посчитаем R_1^2:

In[]:
```
X_cut = np.zeros((n,r))
X_cut[:,0] = np.ones((n,))
beta_cut = np.linalg.pinv(X_cut.T.dot(X_cut)).dot(X_cut.T).dot(Y["INCOME"])
R1 = (Y["INCOME"] - X_cut.dot(beta_cut)).T.dot(Y["INCOME"] - X_cut.dot(beta_cut))
```

Посчитаем R_0^2:

In[]:
```
R0 = (Y["INCOME"] - X.dot(beta)).T.dot(Y["INCOME"] - X.dot(beta))
```

И посчитаем статистику их различия (критическое значение у нас уже посчитано ранее):

In[]:
```
t_mp = (n - r) * (R1 - R0) / ((r - 1) * R0)
print "критическое значение: ", critical_value
print "значение статистики:", t_mp
```

```
критическое значение:  1.7058281678
значение статистики: 0.841695848255
```

Значение статистики с точностью до 10^{-12} совпадает со значением статистики, вычисленной по методу наименьших квадратов.

1.3 Стандартная реализация ANOVA

In[]:
```
t_std, pvalue = f_oneway(*[cities for (state, cities) in groupeddata["INCOME"]])
print "значение статистики:", t_std
```

```
значение статистики: 0.841695848255
```

Сравним значения тестовой статистики, полученные разными методами:

In[]:
```
print t_lsm
print t_mp
print t_std
```

```
0.841695848255
0.841695848255
0.841695848255
```

In[]:
```
print pvalue
```

```
0.69371024531
```

Следовательно, для всех стандартных уровней доверия гипотеза о невлиянии местоположения города на уровень дохода в нём не может быть отвергнута.
Несколько противоречащий интуиции результат, возможно, объясняется неравномерностью исходных данных. Внутри некоторых штатов для различных городов медианный доход различается на несколько порядков:

In[]:
```
print data.loc[data["STATE"] == "CA"]["INCOME"]
```

```
2     30925
6        10
11       26
13        3
31    31938
39    28183
40    24923
42    27095
55       74
58       53
67       50
75    26876
Name: INCOME, dtype: int64
```