Computational Statistics and Data Analysis (MVComp2)

Solutions to exercise 11

Lecturer Tristan Bereau

Semester Wi23/24
Due Jan. 25, 2024, 23:59

1 Defective parts in a shipment (5 points)

A shipment of parts is received, out of which five are tested for defects. The number of defects, X, follows a binomial distribution, $X \sim \text{Binomial}(n = 5, p = \theta)$. The history of past shipments indicates that θ follows a prior distribution, $\text{Beta}(1, 9)$. The test reveals $X = 0$. We wish to establish whether there is significant evidence that the proportion of defective parts in the whole shipment exceeds 10%.

(a) Derive an expression for the posterior probability distribution, $p(\theta|X)$.

(b) Compare the posterior probabilities of the two models:

$$M_1 : \theta \leq 0.1$$
$$M_2 : \theta > 0.1$$

Feel free to evaluate your calculations using statistical libraries. From your results, can you conclude whether the proportion of defective parts in the whole shipment likely exceeds 10%?

1.1 Solution

(a) The problem follows what we covered in Homework 3 Problem 3: the beta distribution is a conjugate prior to the binomial. The resulting posterior yields

$$\begin{aligned} p(\theta|X = k) &= p(X = k|\theta)p(\theta) \\ &\propto \theta^k (1-\theta)^{N-k} \theta^{\alpha-1} (1-\theta)^{\beta-1} \\ &= \theta^{\alpha+k-1} (1-\theta)^{\beta+N-k-1} \\ &= \text{Beta}(\alpha + k, \beta + N - k). \end{aligned}$$

Plugging in the values of $N = 5$, $k = 0$, $\alpha = 1$, and $\beta = 9$, we get

$$p(\theta|X = 0) = \text{Beta}(1, 14).$$

(b) The two models can be written using the posterior

$$p(\theta|X=0, M_1) = \int_0^{0.1} \mathrm{d}\theta \, \mathrm{Beta}(1, 14)$$
$$p(\theta|X=0, M_2) = \int_{0.1}^{1} \mathrm{d}\theta \, \mathrm{Beta}(1, 14)$$

We use `scipy` to compute the integral via the cumulative distribution functions

```
from scipy.stats import beta

p_m1 = beta(1,14).cdf(0.1)
p_m2 = 1. - p_m1
print(f"Prob(M1): {p_m1:.2f},\nProb(M2): {p_m2:.2f}")
```

```
Prob(M1): 0.77,
Prob(M2): 0.23
```

which is very much in favor of model M_1. From this we conclude that there is likely no more than 10% defective parts in the whole shipment.

2 BIC for customer data (5 points)

Download the following dataset about the number of customers entering a store given the hour of the day: customers.csv. The are two features:

Variable	Description
`hour`	hour of the day
`customers`	number of customers in the store

We will consider three models for the number of customers as a function of the number of hours:

1. constant model (i.e., intercept, $\beta_0^{(0)}$)
2. linear model (i.e., intercept $\beta_0^{(1)}$ and slope $\beta_1^{(1)}$)
3. quadratic model (i.e., intercept $\beta_0^{(2)}$, slope $\beta_1^{(2)}$, and quadratic term $\beta_2^{(2)}$)

We want to determine which one of the three regression models performs best.

(a) Via a routine such as `numpy.polyfit`, fit the three models. Report the coefficients and plot the fits against the data.

(b) Under the assumption that the model errors follow a normal distribution, $\mathcal{N}(0, \sigma^2)$, derive an expression for the likelihood term of the BIC, using a maximum-likelihood estimate for the parameter, $\hat{\sigma}^2$. Use it to construct a simple expression for the BIC.

(c) Calculate the Bayesian Information Criterion (BIC) for the three models. Which one performs best according to that metric?

Hint: When dealing with a regression model with k degrees of freedom, the residual variance $\hat{\sigma}^2 = \frac{1}{n-k}\sum_{i=1}^n (y_i - \hat{y}_i)^2$ is an unbiased estimator.

2.1 Solution

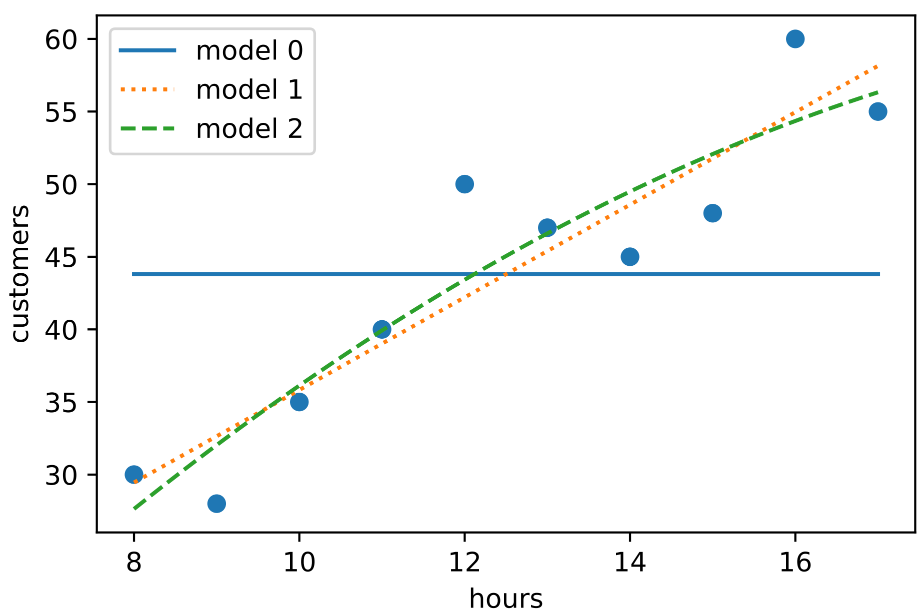
(a) We solve for the fitting parameters of the three models

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("data_11_customers.csv")
f0 = np.polyfit(df["hour"], df["customers"], 0)[0]
f1_0, f1_1 = np.polyfit(df["hour"], df["customers"], 1)
f2_0, f2_1, f2_2 = np.polyfit(df["hour"], df["customers"], 2)
print(f"model 0: f0={f0:.2f}")
print(f"model 1: f1_0={f1_0:.2f}, f1_1={f1_1:.2f}")
print(f"model 2: f2_0={f2_0:.2f}, f2_1={f2_1:.2f}, f2_2={f2_2:.2f}")

x = np.linspace(8, 17)
plt.scatter(df["hour"], df["customers"])
plt.plot(x, 0*x + f0, label="model 0")
plt.plot(x, f1_0*x + f1_1, ":", label="model 1")
plt.plot(x, f2_0*x**2 + f2_1*x + f2_2, "--", label="model 2")
plt.xlabel("hours")
plt.ylabel("customers")
plt.legend()
plt.show()
```

```
model 0: f0=43.80
model 1: f1_0=3.19, f1_1=3.95
model 2: f2_0=-0.15, f2_1=6.98, f2_2=-18.47
```



(b) The BIC is defined as

$$\begin{aligned}\mathcal{L}_{\text{BIC}}(m) &= -2\log p(\mathcal{D}|\boldsymbol{\theta}, m) + k\log n \\ &= -2\log\left[\frac{1}{(2\pi\sigma^2)^{n/2}}\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(f(x_i)-y_i)^2\right)\right] + k\log n\end{aligned}$$

where $f(x_i)$ is the model prediction for the i-th datapoint and y_i is its reference value. Using the residual variance as unbiased estimator of σ^2, we obtain

$$\mathcal{L}_{\text{BIC}}(m) = n\log(2\pi) + n\log(\hat{\sigma}^2) + \frac{1}{\hat{\sigma}^2}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + k\log n.$$

Clearly the first term will be identical across all three models, and so will not contribute in differentiating them.

(c) We compute the BIC for the three models

```python
# Model predictions
df["model_0"] = 0 * df["hour"] + f0
df["model_1"] = f1_0 * df["hour"] + f1_1
df["model_2"] = f2_0*df["hour"]**2 + f2_1*df["hour"] + f2_2

def compute_bic(predictions: pd.Series, ref_values: pd.Series, num_dof: int) -> float:
  assert len(predictions) == len(ref_values)
  num_data = len(ref_values)
  rss = ((predictions - ref_values)**2).sum()
  sig = np.sqrt(rss / (num_data - num_dof))
  loglik = (
    - num_data / 2 * np.log(2 * np.pi)
    - num_data / 2 * np.log(sig**2)
    - 1 / (2 * sig**2) * rss
  )
  return -2 * loglik + num_dof * np.log(num_data)

bic_0 = compute_bic(df["model_0"], df["customers"], 1)
bic_1 = compute_bic(df["model_1"], df["customers"], 2)
bic_2 = compute_bic(df["model_2"], df["customers"], 3)

print(f"BIC_0: {bic_0:.2f}, BIC_1: {bic_1:.2f}, BIC_2: {bic_2:.2f}")
```

```
BIC_0: 76.66, BIC_1: 60.24, BIC_2: 62.03
```

From which we conclude that the linear model has the lowest BIC, and intuitively seems to be the best compromise.