ואדים ליבינוב
314552365

(1)

תרגיל בית 1

נגדיר משתנה y שקובע האם המכונה פגומה או לא:

$$y = \begin{cases} 1 & P \\ -1 & 1-P \end{cases}$$

$$y \sim Bernoulli(P)$$

$$f(x \mid y=1) = \lambda_1 \exp(-\lambda_1 x)$$

$$f(x \mid y=-1) = \lambda_2 \exp(-\lambda_2 x)$$

זוהי פונקציית הנראות עבור n תצפיות בלתי תלויות:

$$\prod_{t=1}^{n} f(x_t ; \theta) = \prod_{t=1}^{n} \sum_{i \in \{-1,1\}} f(x_t \mid y=i ; \theta) \, P(y=i)$$

וכאשר נעשה $\log$ הנראות, המכפלה תהפוך לסכום.

E STeP

פונקציית ה- Auxiliary אשר נרצה למקסם:

$$Q(\theta, \theta_0) = \sum_{t=1}^{n} \sum_{i \in \{-1,1\}} P(y_t = i \mid x_t ; \theta_0) \log P(x_t, y_t = i ; \theta)$$

נפתח את הביטוי השמאלי:

$$P(y_t = i \mid x_t ; \theta_0) = \frac{f(x_t \mid y_t = i ; \theta_0) \cdot P(y_t = i)}{f(x_t)} =$$

$$= \begin{cases} \dfrac{\lambda_1 \exp(-\lambda_1 x_t) \cdot P}{P \lambda_1 \exp(-\lambda_1 \cdot x_t) + (1-P) \lambda_2 \exp(-\lambda_2 x_t)} & i=1 \\[2ex] \dfrac{\lambda_2 \exp(-\lambda_2 x_t) \cdot (1-P)}{P \lambda_1 \exp(-\lambda_1 \cdot x_t) + (1-P) \lambda_2 \exp(-\lambda_2 x_t)} & i=-1 \end{cases}$$

נגדיר את ההסתברויות הראשונות כ- w_{t1}, והשניה כ- w_{t2}.

כעת נפתח את הביטוי הימני:

$$\log p(x_t, y_t=i;\theta) = \log p(y_t=i;\theta)\,p(x_t|y_t=i;\theta) =$$

$$= \begin{cases} \log p + \log \lambda_1 + (-\lambda_1 x_t) & i=1 \\ \log(1-p) + \log \lambda_2 - \lambda_2 x_t & i=-1 \end{cases}$$

כעת נציב ב-Q, ונפתח את הסכום הפנימי:

$$\sum_{t=1}^{n} w_{t1}\left(\log p + \log \lambda_1 - \lambda_1 x_t\right) + w_{t2}\left(\log(1-p) + \log\lambda_2 - \lambda_2 x_t\right)$$

M Step

נתחיל מלמצוא p שממקסם את Q. נגזור ונשווה ל-0:

$$\frac{\partial}{\partial p} \sum_{t=1}^{n} w_{t1}\cdot \log p + w_{t2}\cdot \log(1-p) = \sum_{t=1}^{n} w_{t1}\cdot\frac{1}{p} + w_{t2}\cdot\frac{1}{1-p} = 0$$

לאחר העברות אגפים וצמצומים מתקבל p אופטימלי:

$$p^* = \frac{\sum_{t=1}^{n} w_{t1}}{n}$$

כעת נמצא את λ_1 שממקסם Q:

$$\frac{\partial}{\partial \lambda_1} \sum_{t=1}^{n} w_{t1}\left(\log \lambda_1 - \lambda_1 x_t\right) = \sum_{t=1}^{n} w_{t1}\cdot\frac{1}{\lambda_1} - w_{t1}\cdot x_t = 0$$

לבסוף מתקבל:

$$\lambda_1^* = \frac{\sum_{t=1}^{n} w_{t1}}{\sum_{t=1}^{n} w_{t1} x_t}$$

עבור λ_2 אותן הפעולות ומתקבל:

$$\lambda_2^* = \frac{\sum_{t=1}^{n} w_{t2}}{\sum_{t=1}^{n} w_{t2} x_t}$$

② שלב E:

פונקציית Auxiliary:

$$Q(\theta,\theta^t)=\sum_{i=1}^{n}\sum_{j=1}^{3}P(y_i=j|x_i;\theta)\cdot\log p(x_i,y_i=j;\theta^t)=$$

$$=\sum_{i=1}^{n}\sum_{j=1}^{3}P(y_t=j|x_i;\theta)\log\left(\frac{1}{3}\cdot\mathcal{N}(y_i\mu,1)\right)$$

נגדיר את הגורם השמאלי כ- w_{ji} כהסתברות שדוגמא i תבוא מגאוסיין j.

$$w_{1i}=\frac{P(x_i|y_t=1;\theta)\cdot\cancel{P(y_t=1;\theta)}}{\sum_{j=1}^{3}P(x_i|y_i=j;\theta)\cdot\cancel{P(y_i=j;\theta)}}=\frac{\mathcal{N}(\mu,1)}{\mathcal{N}(\mu,1)+\mathcal{N}(2\mu,1)+\mathcal{N}(3\mu,1)}$$

ובאופן דומה עבור w_{2i} ועבור w_{3i}.

כעת נגזור לפי μ ונשווה ל-0:

$$\frac{\partial}{\partial\mu}\sum_{i=1}^{n}\sum_{j=1}^{3}w_{ji}\log\left(\frac{1}{3}\cdot\left(\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}\cdot(x_i-j\mu)^2\right)\right)\right)=$$

$$\sum_{i=1}^{n}\sum_{j=1}^{3}w_{ji}j(x_i-j\mu)=0$$

לבסוף:

$$\mu^*=\frac{\sum_{i=1}^{n}x_iw_{1i}+2x_iw_{2i}+3x_iw_{3i}}{\sum_{i=1}^{n}w_{1i}+4w_{2i}+9w_{3i}}$$

3.

$$P(y=1|x,z) = \frac{P(z|y=1,x)\cdot P(y=1|x)}{P(z|x)} =$$

$$= \frac{N(\mu,1)\cdot\sigma(wx+b)}{(1-\sigma(wx+b))\cdot N(0,1)+\sigma(wx+b)\cdot N(\mu,1)}$$

המעבר נכון כיוון ש- $p(z|y=1,x)=p(z|y=1)$, הרי אם ידוע ש- $y=1$, z ידגם מ- $N(\mu,1)$, ואין זה משנה מה הוא x.

- המשך בעמוד הבא.

• אנו נמצאים במצב של למידה מונחית. נשתמש בעקרון של הרכבת איברים שתלויים בפרמטרים שונים כדי להקל עלינו.

$$L(\theta)=\sum_{t=1}^{n}\log P(x_t,y_t,z_t;\theta)=\sum_{t=1}^{n}\log\left[P(z_t|y_t,x_t;\theta_z)P(y_t|x_t;\theta_y)\cdot P(x_t)\right]=$$

$$=\sum_{t=1}^{n}\log p(z_t|y_t;\mu)+\sum_{t=1}^{n}\log p(y_t|x_t;w,b)+\sum_{t=1}^{n}P(x_t)$$

כעת נוכל למקסם את הנראות בנפרד כתלות בפרמטרים.

יהי $|S_i|=|\{t \mid y_t=1\}|=m$

$$\sum_{t\in S_i}\frac{\partial}{\partial\mu}\log\left[\frac{1}{\sqrt{2\pi}}\cdot\exp\left(-\frac{1}{2}(z_t-\mu)^2\right)\right]=\sum_{t\in S_i}z_t-\mu=\sum_{t\in S_i}z_t-\sum_{t\in S_i}\mu=0$$

$$\Rightarrow \mu^*=\frac{\sum_{t\in S_i}z_t}{m}$$

כעת נעבור לגבי w ו- b. ניתן לרשום:

$$P(y_t|x_t;w,b)=\sigma(wx+b)^{y_t}\left(1-\sigma(wx+b)\right)^{1-y_t}$$

כיוון שאין פתרון סגור ואנו בעצם מנסים לשערך משקולות של רגרסיה לוגיסטית, נשתמש ב- Gradient Descent ופונקציית המחיר שלנו תהיה NLL, לוג-נראות שלילי. וכך נעדכן הפרמטרים בעזרת קבוע למידה, ונמקסם את הנראות.

שלב E:

$$\omega_{ti} = P(y_t = i \mid x_t, z_t; \theta_0)$$

מצאנו את ההסתברות הזו בסעיף א', ואת ההסתברות המשלימה ניתן גם כן למצוא בקלות.

$$Q(\theta,\theta_0) = \sum_{t=1}^{n} \sum_{i\in\{0,1\}} \omega_{ti} \log P(x_t, y_t, z_t; \theta) =$$

$$\sum_{t=1}^{n} \sum_{i\in\{0,1\}} \omega_{ti} \cdot \log\left[P(x_t \mid y_t, z_t)\, P(z_t \mid y_t; \mu_i)\, P(y_t; \omega, b)\right]$$

שלב M:

כיוון שביצענו הפרדה פרמטרים, כעת נוכל לגזור ולהשוות ל-0.

$$\hat{\omega} = \frac{\partial Q}{\partial \omega} \qquad \hat{b} = \frac{\partial Q}{\partial b} \qquad \hat{\mu} = \frac{\partial Q}{\partial \mu}$$

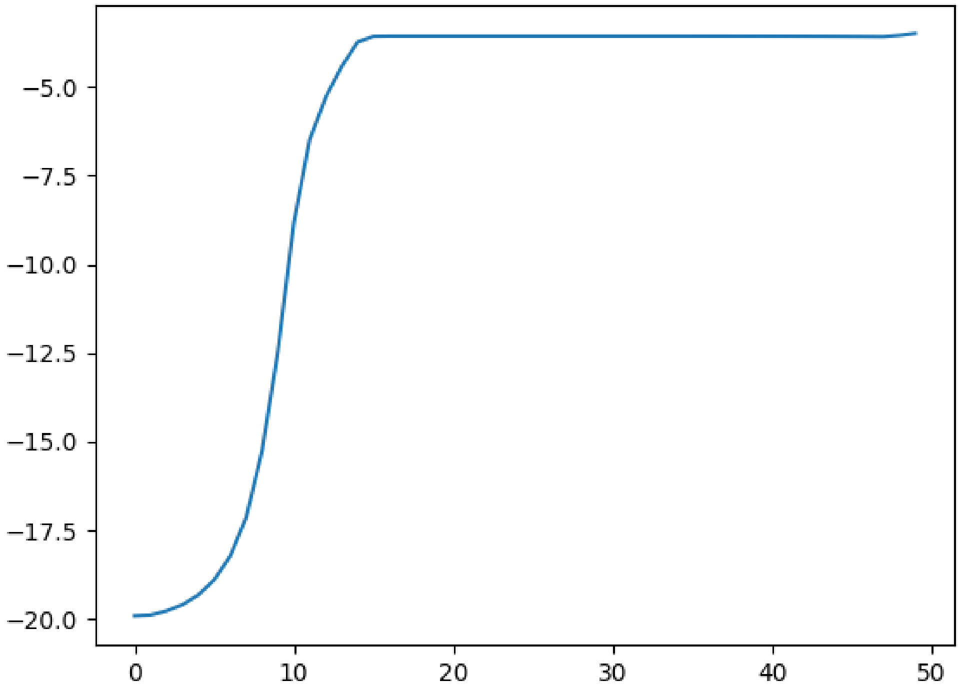
4. (b) הפרמטרים שהתקבלו כאשר היה שימוש ב-20 דוגמאות היו לא מדויקים ברוב המקרים. לדוגמא:

$$\mu = (2.3,\ 4.6,\ 9.1)$$
$$\sigma^2 = (0.5,\ 0.78,\ 0.76)$$
$$\alpha = (0.15,\ 0.3,\ 0.55)$$

כאשר מגדילים את מספר הדוגמאות ל-200, הפרמטרים שמתקבלים קרובים הרבה יותר ל"אמיתיים". זה קורה בגלל שככל שגודל המדגם עולה, ההסתברות שהממוצע שלו יסטה מהתוחלת, קטנה (חוק המספרים הגדולים).

(c) לאחר ניסוי זה ניתן לראות שגם הפרמטרים המשוערכים וגם הנראות לעתים שונים מניסוי לניסוי. זה מצביע על כך ש-EM מוצא מקסימום לוקלי של הנראות, ולא מקסימום גלובלי (שהוא אחד ויחיד בנראות).

Scanned with CamScanner




```
# Vadim Litvinov
import collections
from random import choices
from random import gauss
from math import sqrt
from matplotlib import pyplot as plt
import numpy as np
import seaborn as sns

K = 3
N = 200
N_ITER = 1000
PARAMETERS = [[-1.0, sqrt(2)],
              [4.0, sqrt(3)],
              [9.0, sqrt(1)]]
WEIGHTS = [0.2, 0.3, 0.5]
EPS = 1e-8


def PDF(data, means, variances):
    return 1/(np.sqrt(2 * np.pi * variances) + EPS) * np.exp(-1/2 * (np.square(data - means)
/ (variances + EPS)))


def getRandomParams(n):
    x = np.random.normal(0, 1, size=(n, n))
    return np.dot(x, x.transpose())


def emGmm(data, k, n_iter):
    weights = np.ones((k, 1)) / k  # shape=(k, 1)
    means = np.random.choice(data, k)[:, np.newaxis]  # shape=(k, 1)
    print('starting point means: ', means)
    #variances = np.random.random_sample(size=k)[:, np.newaxis]  # shape=(k, 1)
    data = np.repeat(data[np.newaxis, :], k, 0)  # shape=(k, n)
    vars = np.expand_dims(np.mean(np.square(data - means), axis=1), -1)
    p_list = []
    for step in range(n_iter):
        p = PDF(data, means, vars)
        b = p * weights
        denom = np.expand_dims(np.sum(b, axis=0), 0) + EPS
        b = b / denom
        p_list.append(np.sum(np.log(np.amax(b, axis=0))))
        #print('average likelihood: ', np.sum(np.amax(b, axis=0)) / (N))
```

```
        means_n = np.sum(b * data, axis=1)
        means_d = np.sum(b, axis=1) + EPS
        means = np.expand_dims(means_n / means_d, -1)
        vars = np.sum(b * np.square(data - means), axis=1) / means_d
        vars = np.expand_dims(vars, -1)
        weights = np.expand_dims(np.mean(b, axis=1), -1)
    print('log likelihood: ', np.sum(np.log(np.amax(b, axis=0))))
    #plt.plot(range(n_iter), p_list)
    #plt.show()
    return means, vars, weights


if __name__ == '__main__':
    sampled_gaussians = choices([0, 1, 2], WEIGHTS, k=N)
    print(collections.Counter(sampled_gaussians))
    sampled_gaussians = collections.Counter(sampled_gaussians)
    #sampled_gaussians = list(sampled_gaussians.items())
    samples = np.empty([0, N])
    for i in range(len(sampled_gaussians)):
        #samples.append(gauss(PARAMETERS[sampled_gaussians[i]][0],
        #                PARAMETERS[sampled_gaussians[i]][1]))
        mu_i = PARAMETERS[i][0]
        sigma_i = PARAMETERS[i][1]
        sample_size = sampled_gaussians[i]
        samples = np.concatenate((samples, np.random.normal(mu_i, sigma_i,
size=sampled_gaussians[i])), axis=None)
    samples = np.asarray(samples, dtype=np.float32)
    #check weights is properly working
    #print(sampled_gaussians.count(0), sampled_gaussians.count(1),
sampled_gaussians.count(2))
    #print(sampled_gaussians)
    print('sampled_gaussians: ', sampled_gaussians)
    print('samples: ', samples)
    #plt.hist(samples, bins=200)
    #plt.show()
    #random_params = initializeParams()

    for i in range(10):
        means, variances, weights = emGmm(samples, K, N_ITER)
        print('means: ', means, 'variences: ', variances, 'weights: ', weights, '\n')
```