۱) الف)

می‌دانیم یکی از ساده‌ترین روش‌ها استفاده از هیستوگرام است اما این روش محدودیت‌هایی نیز دارد از جمله اینکه به سایز bin در نظر گرفته بسیار وابسته است. به این منظور از آن جا که روش پارزن شباهت زیادی به این روش دارد و در آن حجم ثابت در نظر گرفته می‌شود می‌توان گفت در ابعاد کوچک بهتر است از این روش استفاده کنیم.

ب) از آن جایی که هر دو این روش‌ها از جمله روش‌های نان پارامتریک هستند و برای حل مسئله احتیاج به کل داده‌ها دارند ممکن است سرعت بالایی نداشته باشند. در روش پارزن سرعت می‌تواند در فضاهایی که تعداد داده‌ها در آن حجم ثابت، بالاست پایین باشد. در حالی که در روش KNN بالا بودن تعداد k و یا انتخاب متریک پیچیده برای فاصله سرعت را پایین ببرد. علاوه بر این در روش پارزن چون حجم ثابت است ممکن است تعداد داده زیادی در حجم مشخص وجود نداشته باشد. بنابراین احتمالا در ابعاد بالا روش KNN به علت ثابت در نظر گرفتن k بهتر عمل کند. از خوبی‌های دیگر هر دو این روش‌ها نیز این است که برای داده‌ها یک توزیع خاص در نظر نمی‌گیرند و می‌توانند برای هر توزیع دلخواهی از آن‌ها استفاده کرد.

پ)

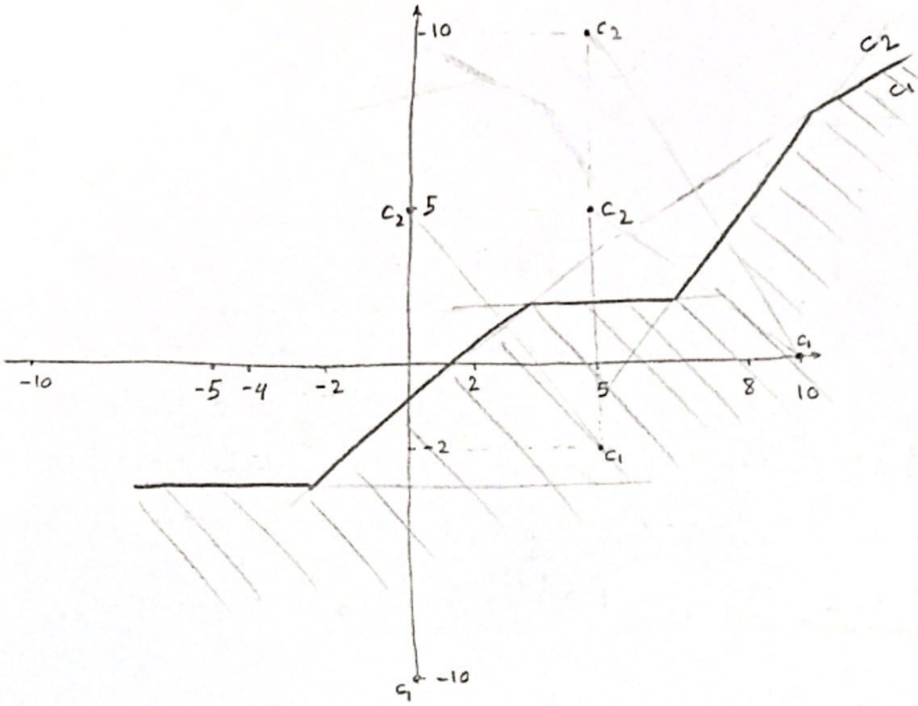
$$\hat{\theta}_{MAP} = \arg\max_{\theta} p(\theta|D) = \arg\max_{\theta} p(D|\theta)p(\theta) \rightarrow \text{posterior}$$ احتمال max کردن

$$\hat{\theta}_{MLE} = \arg\max_{\theta} L(\theta|D) = \arg\max_{\theta} p(D|\theta) \rightarrow \text{likelihood}$$ max کردن

← در صورتی که تعداد داده‌ها زیاد باشد و $p(\theta)$ برای همه‌ی کلاس‌ها یکی باشد (تا این عبارت $p(\theta)$ در MAP و MLE برای همه‌ی کلاس‌ها یکی باشد) این دو تخمین یکسان اند.

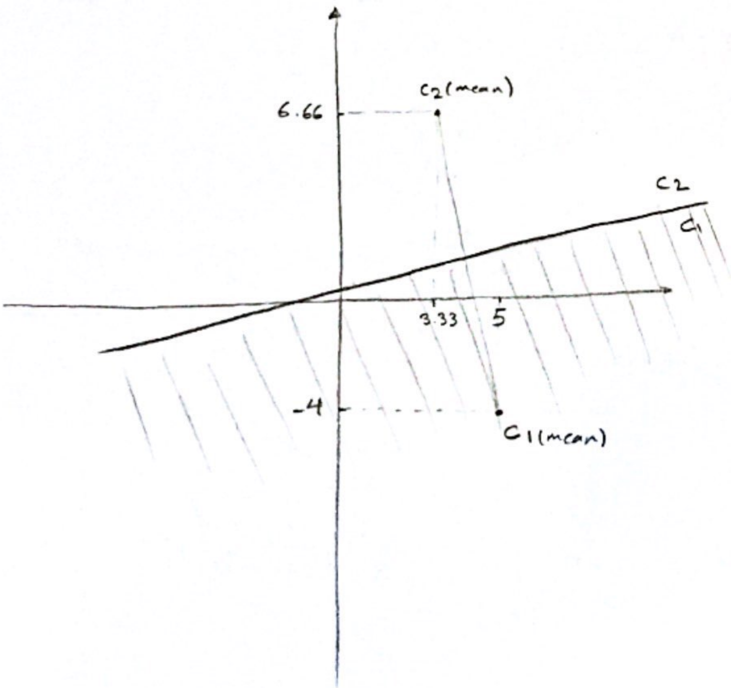
* دراین سوال خطوط به صورت تقریبی کشیده شده اند.

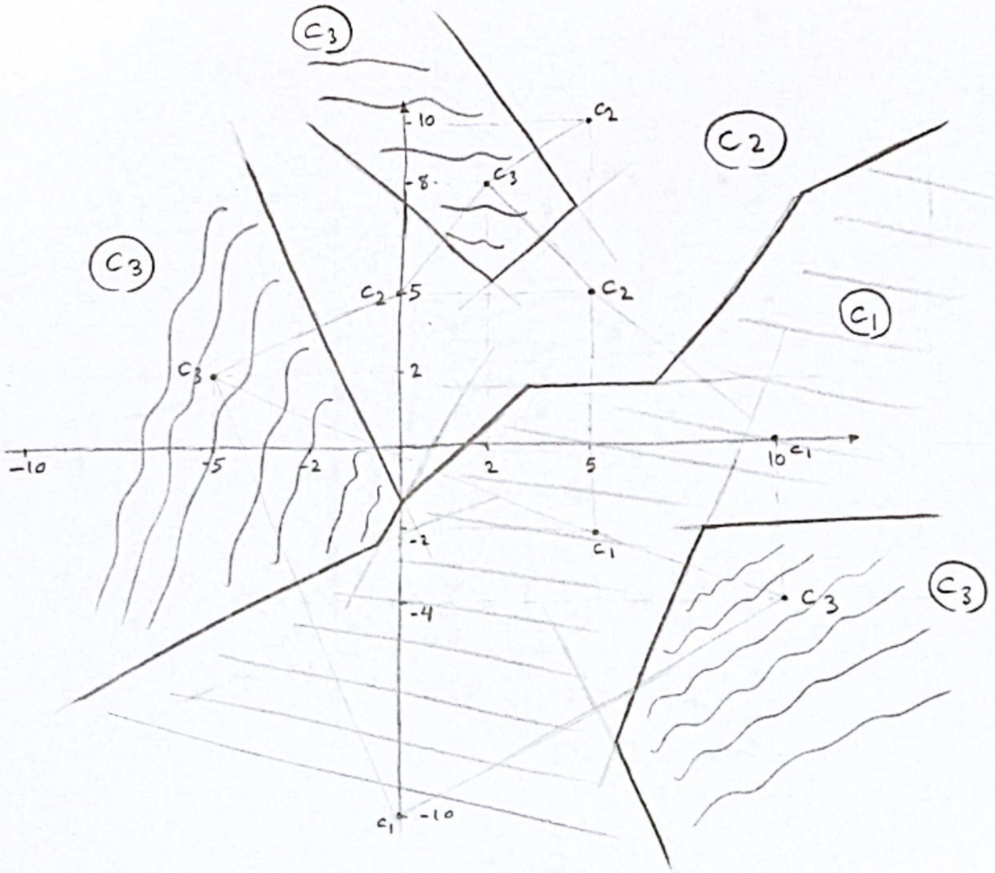
②



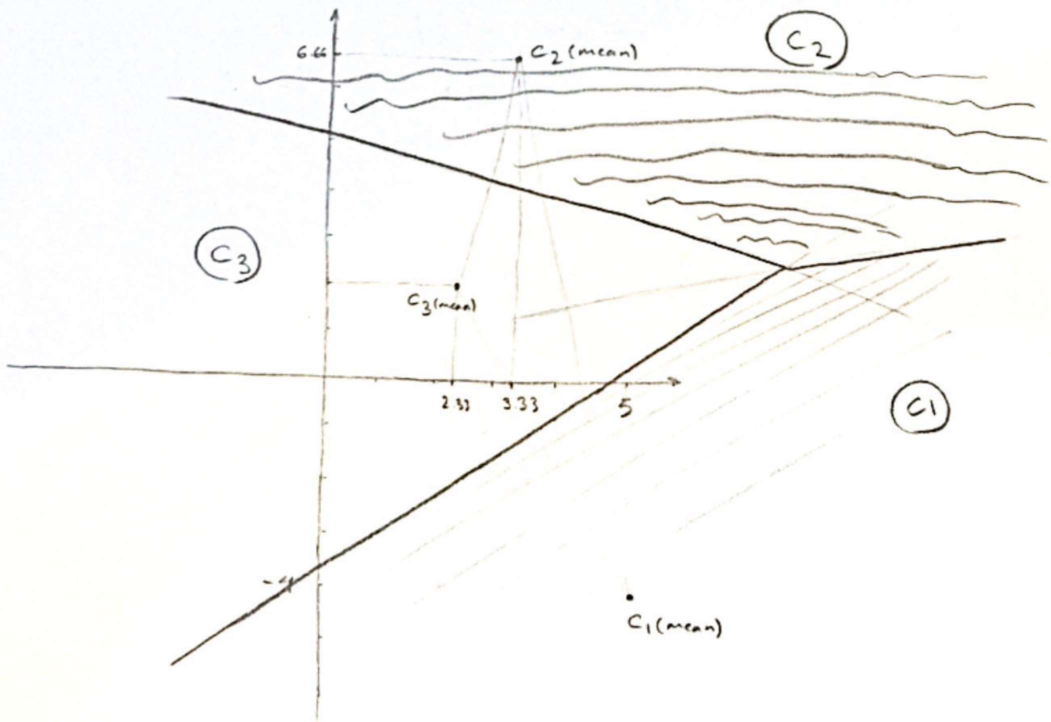
الف) در به دو نقاط به هم وصل شده (از اضلاع های مختلف) و عمود منصف آنها کشیده شده است و سپس مرزهای تقسیم به کمک آن به دست آمده است

ب) $c_1 \rightarrow \frac{10+5}{3}, \frac{-10-2}{3} = 5, -4$ $\qquad c_2 \rightarrow \frac{5+5}{3}, \frac{10+5+5}{3} = 3.33, 6.66$





$c_1 \rightarrow 5, -4$, $c_2 \rightarrow 3.33, 6.66$, $c_3 = \frac{2-5+10}{3} = 2.33$, $\frac{8+2-4}{3} = 2$



③

a) $P(\omega_i) = \frac{1}{c}$

$$P^* = \int P^*(e|x)\, p(x)\, dx \quad , \quad P^*(e|x) = 1 - P(\omega_{max}|x)$$

$$\to p(x) = \sum_{i=1}^{c} p(x|\omega_i) P(\omega_i) = \begin{cases} \frac{1}{c}\sum_{i=1}^{c} 1 = \frac{c}{c} = 1 & 0 \leqslant x \leqslant \frac{cr}{c-1} \\ \frac{1}{c} & i \leqslant x \leqslant (i+1) - \frac{cr}{c-1} \\ 0 & \text{o.w} \end{cases}$$

$\to$ تنها یک عدد صحیح i وجود دارد، برای بقیه 0 می‌شود، Σ تنها برای یک غیر 0 است

$$P(\omega_i|x) = \frac{p(x|\omega_i) P(\omega_i)}{p(x)} = \begin{cases} \frac{1 \times \frac{1}{c}}{1} = \frac{1}{c} & 0 \leqslant x \leqslant \frac{cr}{c-1} \\ \text{if } i \neq j \to p(x|\omega_i) = 0 \to 0 \\ \text{if } i = j \to p(x|\omega_i) = 1 \to \frac{1 \times \frac{1}{c}}{\frac{1}{c}} = 1 & i \leqslant x \leqslant (i+1) - \frac{cr}{c-1} \\ 0 & \text{o.w} \end{cases}$$

$$\to P^*(e|x) = 1 - P(\omega_{max}|x) = \begin{cases} 1 - \frac{1}{c} & 0 \leqslant x \leqslant \frac{cr}{c-1} \\ 1 - 1 = 0 & i \leqslant x \leqslant i + 1 - \frac{cr}{c-1} \\ 0 & \text{o.w} \end{cases}$$

$$\to P^* = \int_0^{\frac{cr}{c-1}} \left(1 - \frac{1}{c}\right) p(x)\, dx = \int_0^{\frac{cr}{c-1}} \left(1 - \frac{1}{c}\right) \times 1\, dx = \left(1 - \frac{1}{c}\right) x \Big|_0^{\frac{cr}{c-1}} = \frac{c-1}{\cancel{c}} \frac{\cancel{c}r}{c-1} = \boxed{r}$$

b)

$$P = \int \left[1 - \sum_{i=1}^{c} P^2(\omega_i|x) \right] p(x)\, dx = \int_0^{\frac{cr}{c-1}} \left[1 - \sum_{i=1}^{c} \frac{1}{c^2} \right] \times 1\, dx +$$

$$\sum_{j=1}^{c} \int_{j}^{j+1-\frac{cr}{c-1}} \left[1 - \frac{1}{1} \right] \frac{1}{c}\, dx = \int_0^{\frac{cr}{c-1}} \left[1 - c \times \frac{1}{c^2} \right] dx = \int_0^{\frac{cr}{c-1}} \left(1 - \frac{1}{c}\right) dx = \frac{c-1}{c} x \Big|_0^{\frac{cr}{c-1}} =$$

Σ نماد

j=1 اشتراک دارد — تنها در یکی از آنها

اشتراک یک دارد

$$\frac{\cancel{c-1}}{\cancel{c}} \frac{\cancel{c}r}{\cancel{c-1}} = \boxed{r}$$

$$\Rightarrow a, b \rightsquigarrow \boxed{P^* = P}$$

4

a) $Q(\theta, \theta^0) = E\left[\ln p(x_g, x_b; \theta) | \theta^0, D_g \right] =$

$$\int_{-\infty}^{+\infty} \left[\ln p(x_1|\theta) + \log p(x_2|\theta) + \ln p(x_3|\theta) \right] p(x_{32}|\theta^0, x_{31} = 2)\, dx_{32} =$$

$$\ln p(x_1|\theta) + \ln p(x_2|\theta) + \int_{-\infty}^{+\infty} \ln p(x_3|\theta)\, p(x_{32}|\theta^0, x_{31} = 2)\, dx_{32} =$$

$$\ln p(x_1|\theta) + \ln p(x_2|\theta) + \int_{-\infty}^{+\infty} \ln p((2, x_{32})|\theta) \frac{p((2, x_{32})|\theta^0)}{\int_{-\infty}^{+\infty} p((2, x'_{32})|\theta^0)\, dx'_{32}}\, dx_{32}$$

$$p((2, x_{32})|\theta^0) = P_{x_1}(x_{31} = 2|\theta^0)\, P_{x_2}(x_{32}|\theta^0)$$

$$\int_{-\infty}^{+\infty} p((2, x'_{32})|\theta^0)\, dx'_{32} = \int_{-\infty}^{+\infty} P_{x_1}(x_{31} = 2|\theta^0) P_{x_2}(x'_{32}|\theta^0)\, dx'_{32} = P_{x_1}(x_{31} = 2|\theta^0) \underbrace{\int_{-\infty}^{+\infty} P_{x_2}(x'_{32}|\theta^0)\, dx'_{32}}_{1}$$

$$\rightarrow Q(\theta, \theta^0) = \ln p(x_1|\theta) + \ln p(x_2|\theta) + \int_{-\infty}^{+\infty} P_{x_2}(x_{32}|\theta^0) \ln p((2, x_{32})|\theta)\, dx_{32}$$

$$\rightarrow p(x|\theta) = p_{x_1}(x_1|\theta)\, p_{x_2}(x_2|\theta) = \begin{cases} \frac{1}{\theta_1\theta_2} e^{-\theta_1 x_1} & x_1 > 0 \;,\; 0 \leq x_2 \leq \theta_2 \\ 0 & \text{o.w} \end{cases}$$

$$\rightarrow \ln \underbrace{p((2, x_{32})|\theta)}_{0 \leq x_{32} \leq \theta_2 \;,} \; \underbrace{p_{x_2}(x_{32}|\theta^\circ)}_{0 \leq x_{32} \leq \theta_2^\circ = 4} \rightarrow \neq 0$$

else $\rightarrow = 0$

$$\rightarrow \ln p(x_1|\theta) + \ln p(x_2|\theta) = \ln\left(\frac{1}{\theta_1\theta_2} e^{-\theta_1}\right) + \ln\left(\frac{1}{\theta_1\theta_2} e^{-3\theta_1}\right) = -4\theta_1 - 2\ln(\theta_1\theta_2)$$

$$\rightarrow \int_{-\infty}^{+\infty} \ln p((2, x_{32})|\theta)\, p_{x_2}(x_{32}|\theta^\circ) = \int_0^{\min(4,\theta_2)} \ln\left(\frac{1}{\theta_1\theta_2} e^{-2\theta_1}\right) \times \frac{1}{4}\, dx_{32} =$$

$$\begin{cases} \frac{1}{4}\int_0^{\theta_2} \ln\left(\frac{1}{\theta_1\theta_2} e^{-2\theta_1}\right) dx_{32} = \frac{1}{4}\theta_2 \underbrace{\ln\left(\frac{1}{\theta_1\theta_2} e^{-2\theta_1}\right)}_{-2\theta_1 - \ln(\theta_1\theta_2)} & 3 \leq \theta_2 \leq 4 \\ \frac{1}{4}\int_0^{4} \ln\left(\frac{1}{\theta_1\theta_2} e^{-2\theta_1}\right) dx_{32} = \underbrace{\ln\left(\frac{1}{\theta_1\theta_2} e^{-2\theta_1}\right)}_{-2\theta_1 - \ln(\theta_1\theta_2)} & \theta_2 \geq 4 \\ 0 & \text{o.w} \end{cases}$$

$$\Downarrow$$

$$\boxed{Q(\theta, \theta^\circ) = \begin{cases} -4\theta_1 - 2\ln(\theta_1\theta_2) - \frac{\theta_2}{4}\left(\ln(\theta_1\theta_2 + 2\theta_1)\right) & 3 \leq \theta_2 \leq 4 \\ -4\theta_1 - 2\ln(\theta_1\theta_2) - 2\theta_1 - \ln(\theta_1\theta_2) & \theta_2 \geq 4 \\ -4\theta_1 - 2\ln(\theta_1\theta_2) & \text{o.w} \end{cases}}$$

انتگرال روی کل P برابر 1 $\rightarrow \int_{-\infty}^{+\infty} P_{x_1}(x_1)\, dx_1 = 1 \rightarrow \int_{-\infty}^{+\infty} \frac{1}{\theta_1} e^{-\theta_1 x_1} dx_1 = 1$

$$\rightarrow \int_0^{+\infty} \frac{1}{\theta_1} e^{-\theta_1 x_1} dx_1 = 1 \rightarrow -\frac{1}{\theta_1^2} e^{-\theta_1 x_1} \Big|_0^{+\infty} = 1 \rightarrow \frac{1}{\theta_1^2} = 1 \rightarrow \boxed{\theta_1 = 1}$$

b)

1. $3 \leq \theta_2 \leq 4$, $\theta_1 = 1$

$$Q(\theta, \theta^0) = -4 - 2\ln(\theta_2) + \frac{\theta_2}{4}\{-\ln\theta_2 - 2\} = -4 - \underbrace{\left(2\ln(\theta_2) + \frac{\theta_2}{4}\{\ln\theta_2 + 2\}\right)}_{\text{تابع نزولی هرچه } \theta \text{ کمتر مقدار بیشتر}}$$

$$\rightarrow \arg\max_{\theta_2} Q(\theta, \theta^0) = 3$$

$$Q(\theta, \theta^0) = -4 - \left(2\ln(3) + \frac{3}{4}\{\ln 3 + 2\}\right) = \boxed{-8.52}$$

2. $\theta_2 \geq 4$

$$\arg\max_{\theta_2} Q(\theta, \theta^0) = 4$$

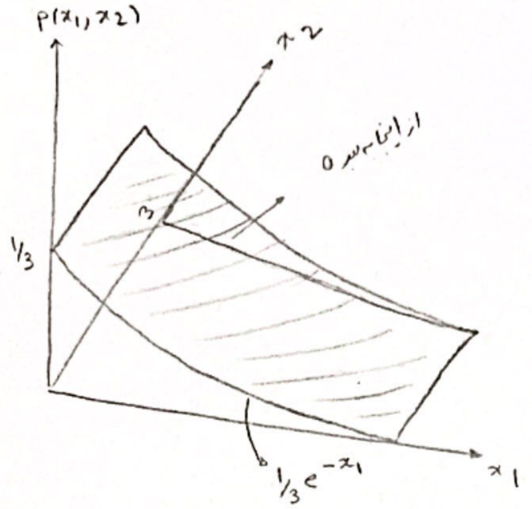
$$\rightarrow Q(\theta, \theta^0) = -4 - 2\ln 4 - 2 - \ln 4 = -6 - 3\ln 4 = \boxed{-10.16}$$

Q در حالت $\theta_2 = 3$ بیشینه شد $\rightarrow \boxed{\theta = \begin{pmatrix} 1 \\ 3 \end{pmatrix}}$

c) $\rightarrow \theta = \begin{pmatrix} 1 \\ 3 \end{pmatrix} \rightarrow p(x|\theta) = \begin{cases} \frac{1}{3} e^{-x_1} & x_1 > 0,\ 0 \leq x_2 \leq 3 \\ 0 & \text{o.w} \end{cases}$

$\rightarrow \theta = \begin{pmatrix} 2 \\ 4 \end{pmatrix} \rightarrow p(x|\theta) = \begin{cases} \frac{1}{8} e^{-2x_1} & x_1 > 0,\ 0 \leq x_2 \leq 4 \\ 0 & \text{o.w} \end{cases}$

* نمودار داده با پارامترهای جدید:



1) $L = \prod_{k=1}^{n} \theta e^{-\theta x_k} \xrightarrow{\ln} l = \sum_{k=1}^{n} \ln(\theta e^{-\theta x_k}) = n\ln(\theta) - \sum_{k=1}^{n} \theta x_k \underbrace{\ln(e)}_{1} =$

$$\to \frac{\partial l}{\partial \theta} = \frac{n}{\theta} - \sum_{k=1}^{n} x_k = 0 \to \frac{n}{\theta} = \sum_{k=1}^{n} x_k \to \boxed{\theta = \frac{n}{\sum_{k=1}^{n} x_k}}$$

2) $L = \prod_{k=1}^{n} \frac{x_k}{\theta^2} e^{-\frac{x_k^2}{2\theta^2}} \xrightarrow{\ln} l = \sum_{k=1}^{n} \underbrace{\ln\left(\frac{x_k}{\theta^2} e^{-\frac{x_k^2}{2\theta^2}}\right)}_{\underbrace{\ln\left(\frac{x_k}{\theta^2}\right)}_{\ln(x_k) - \ln(\theta^2)} - \frac{x_k^2}{2\theta^2}} = \sum_{k=1}^{n} \ln(x_k) - 2n\ln(\theta) - \sum_{k=1}^{n} \frac{x_k^2}{2\theta^2}$

$$\to \frac{\partial l}{\partial \theta} = -\frac{2n}{\cancel{\theta}} + \frac{1}{\theta^{\cancel{3}2}} \sum_{k=1}^{n} x_k^2 = 0 \to 2n = \frac{1}{\theta^2} \sum_{k=1}^{n} x_k^2 \to$$

$$\to \theta = \left\{ \frac{\sum_{k=1}^{n} x_k^2}{2n} \right\}^{1/2} = \boxed{\left\{ \frac{1}{n} \sum_{k=1}^{n} \frac{x_k^2}{2} \right\}^{1/2}}$$

3) $L = \prod_{k=1}^{n} \sqrt{\theta}\, x_k^{\sqrt{\theta-1}} \xrightarrow{\ln} \sum_{k=1}^{n} \ln\left(\sqrt{\theta}\, x_k^{\sqrt{\theta-1}}\right) = \frac{n}{2} \ln(\theta) + \sqrt{\theta-1} \sum_{k=1}^{n} \ln(x_k)$

$$\frac{\partial l}{\partial \theta} = 0 \to \frac{n}{2\theta} + \frac{1}{2\sqrt{\theta-1}} \sum_{k=1}^{n} \ln(x_k) = 0$$

$$\xrightarrow{\text{صورت}=0} n\sqrt{\theta-1} + \theta \overbrace{\sum_{k=1}^{n} \ln(x_k)}^{A} = 0 \to \sqrt{\theta-1} = u \to \theta = u^2+1$$

$$nu + (u^2+1)A = 0 \to Au^2 + nu + A = 0 \to u = \frac{-n \pm \sqrt{n^2 - 4A^2}}{2A}$$

$$\to \theta = u^2+1 = \boxed{\left\{ \frac{-n \pm \sqrt{n^2 - 4\left[\sum \ln(x_k)\right]^2}}{2\sum_{k=1}^{n} \ln(x_k)} \right\}^2 + 1}$$

۶-

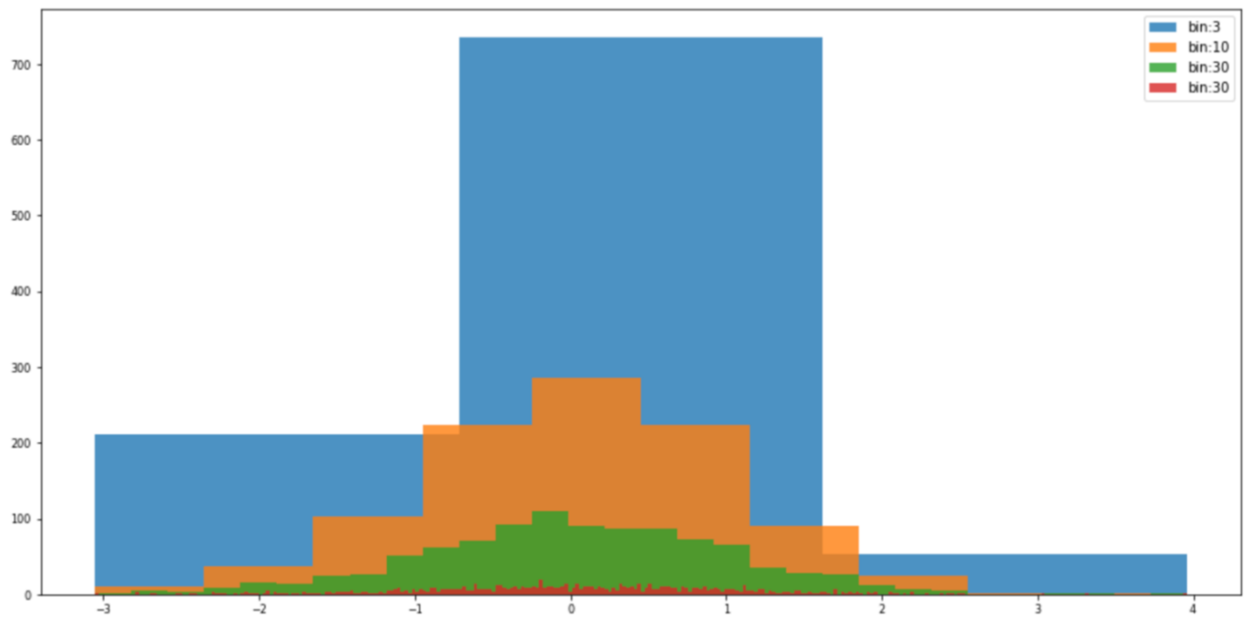
(a

```
import numpy as np
def make_data(N, f=0.3, rseed=1):
    rand = np.random.RandomState(rseed)
    x = rand.randn(N)
    return x

x = make_data(1000)
```

Part A: Plot histogram of x for bins = 3, 10, 30 and 300 and compare the result.

```
[ ] import matplotlib.pyplot as plt
    plt.hist(x, bins = 3 , label="bin:3",alpha = 0.8)
    plt.hist(x, bins = 10 , label="bin:10",alpha=0.8)
    plt.hist(x, bins = 30 , label="bin:30",alpha=0.8)
    plt.hist(x, bins = 300 , label="bin:30",alpha=0.8)
    plt.legend()
    #total area under the curve is equal to 1
```

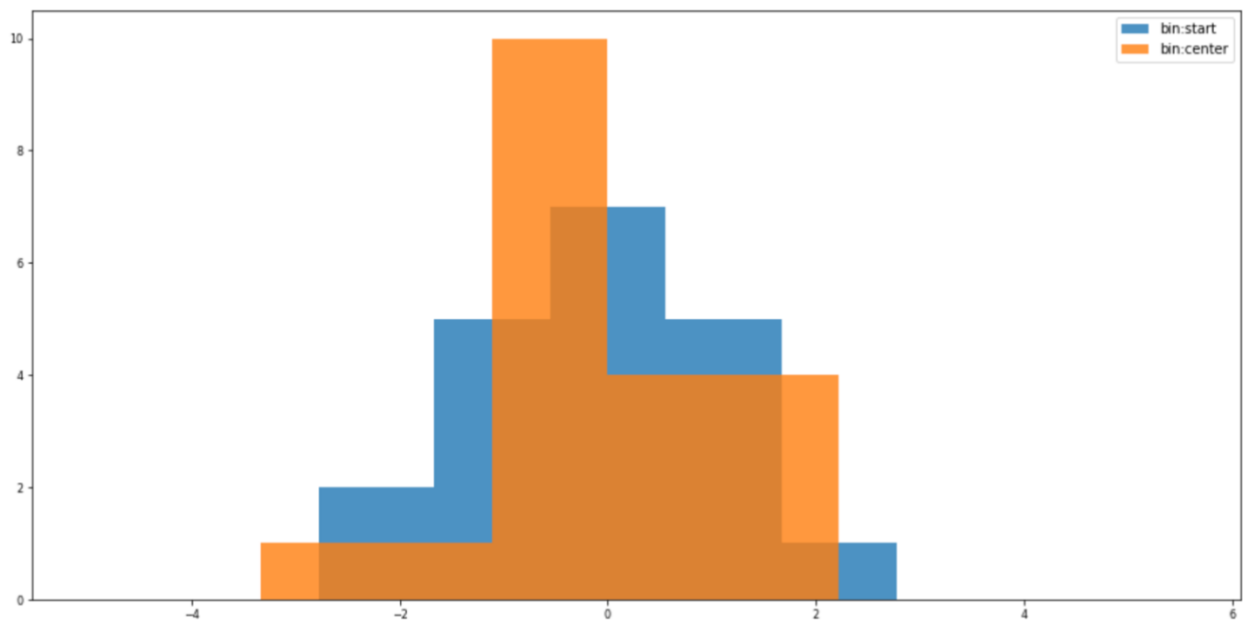


همان طور که مشاهده می شود هنگامی که تعداد bin ها زیاد باشد و در واقع طول هر بین کم باشد تعداد داده های کمتری در هر bin قرار می گیرد و مشابه شکل می بینیم که ارتفاع نمودار که همان تعداد داده ها است کمتر می شود. از طرفی افزایش تعداد بین ها نیز می تواند نمودار نویزی کند. در طرف مقابل و در صورتی که تعداد bin ها کم باشد نمودار حالت smooth تری دارد و تعداد داده ها در هر بین بیشتر است.

(b

```
[ ] x = make_data(20)
    bins1 = np.linspace(-5, 5, 10)
    bin_length = bins1[1]-bins1[0]
    bins2 = np.linspace(-5+bin_length/2,5+bin_length/2,10)
```

```
plt.hist(x, bins = bins1,label="bin:start", alpha = 0.8)
plt.hist(x, bins = bins2,label="bin:center", alpha = 0.8)
plt.legend()
```



همان طور که در قسمت قبل نیز توضیح داده شد طول bin و یا نحوه انتخاب بازه ها و نقاط شروع و پایان آن می تواند بر روی شکل هیستوگرام تاثیر بگذارد که در واقع یکی از مشکلات هیستوگرام ها همین موضوع است. در شکل بالا نیز مشاهده می شود که تنها انتخاب بازه های متفاوت برای bin ها باعث شده یک توزیع تقریبا متقارن و توزیع دیگر با مقداری چولگی باشد. در واقع درحالی که می دانیم هر دو این نمودار ها از یک توزیع تولید شده اند اما به دلیل انتخاب بین های متفاوت با دیدن شکل هیستوگرام این گونه به نظر می رسد که داده ها از دو توزیع متفاوت هستند.

۷-

a)

برای نوشتن توابع مورد نظر از روابط موجود در parzen استفاده شده است:

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right).$$

```
def hypercube_kernel(h, x, x_i):
    if np.linalg.norm(x-x_i) <= h/2:
        return 1
    else:
        return 0

def gaussian_kernel(h, x, x_i, sigma=0.1):
    return  multivariate_normal(mean=0, cov=sigma).pdf((x-x_i)/h)

def parzen_window(X, x, h, kernel_func):
    N = np.size(X,0)
    d = np.size(X,1)
    px = 1/N * 1/(h**d) * np.sum([kernel_func(h,x,xi) for xi in X ])
    return px


def parzen(X, X1, h, kernel_func):
    probs = []
    for x in X1:
        px = parzen_window(X,x,h,kernel_func)
        probs.append(px)
    return np.array(probs)
```

b)

در این قسمت در ابتدا یک dictionary تعریف شده تا به کمک آن بدانیم به ازای هر کلاس داده های آموزشی چه هستند و پس از آن تابع گفته شده پیاده سازی شده است که در ابتدا با توجه به این که قصد داریم توزیع داده را حول چه کلاسی رسم کنیم خروجی پارزن به دست آمده و سپس نمودار مربوطه کشیده شده است.

- در این جا برای راحتی کار در ادامه ورودی پارزن به درون تابع انتقال داده شده است.

```
[ ] train_data = {}
    for i in range(0,4):
        idx = np.where(y_true == i)[0]
        train_data[i] = X[idx]
```

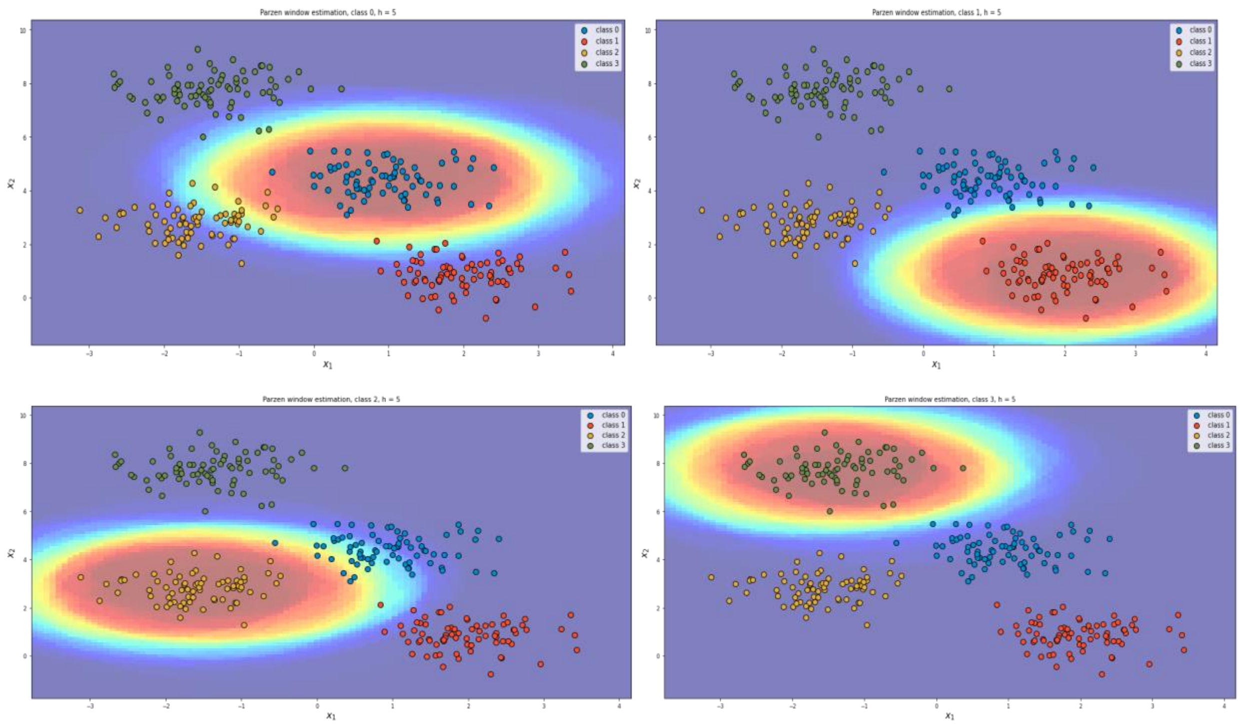
Part B:

```
[ ] def draw_point_distribution(distclass,hsize,kernel):
        pz = parzen(train_data[distclass],xy,hsize,kernel).reshape(100, 100)
        for i in classes:
            data = train_data[i]
            plt.scatter(data[:,0],data[:,1], c=colors[i], marker='o',edgecolors='black', label="class "+str(i))
        plt.ylabel('$x_2$', fontsize=14)
        plt.xlabel('$x_1$', fontsize=14)
        plt.title('Parzen window estimation, class '+str(distclass)+', h = '+str(hsize))
        plt.legend()
        plt.imshow(pz,origin='lower', extent=(x1min,x1max,x2min,x2max), alpha=.5, aspect='auto')
        plt.show()
```

c) تصاویر مورد نظر به ازای کرنل های مختلف به صورت زیر هستند.

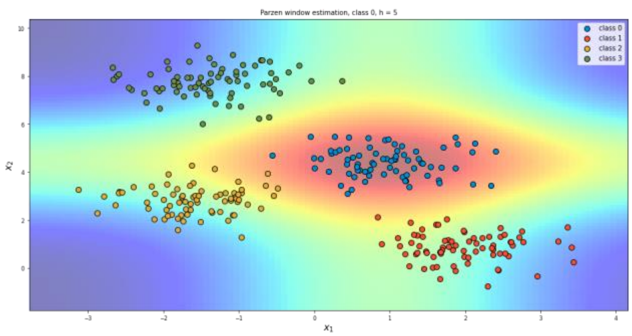
hypercube kernel plots

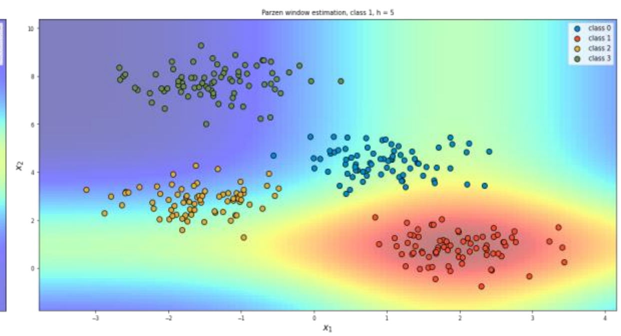
```
f = hypercube_kernel
h=5
distclasslist = [0,1,2,3]
for distclass in distclasslist:
    draw_point_distribution(distclass,h,f)
```

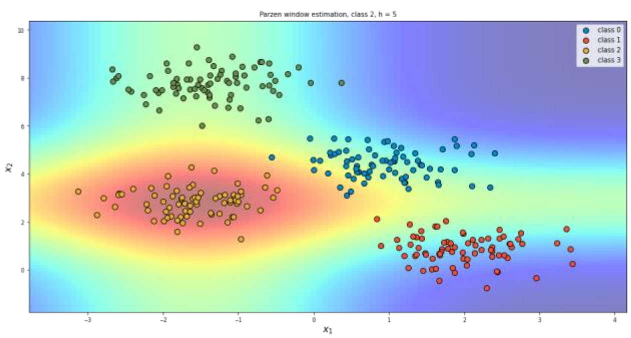


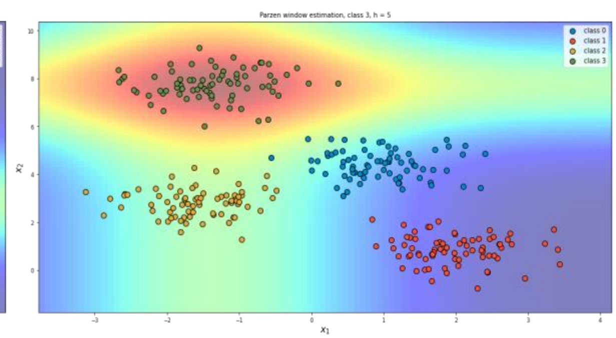
guassian kernel plots

```
f = gaussian_kernel
h=5
distclasslist = [0,1,2,3]
for distclass in distclasslist:
    draw_point_distribution(distclass,h,f)
```









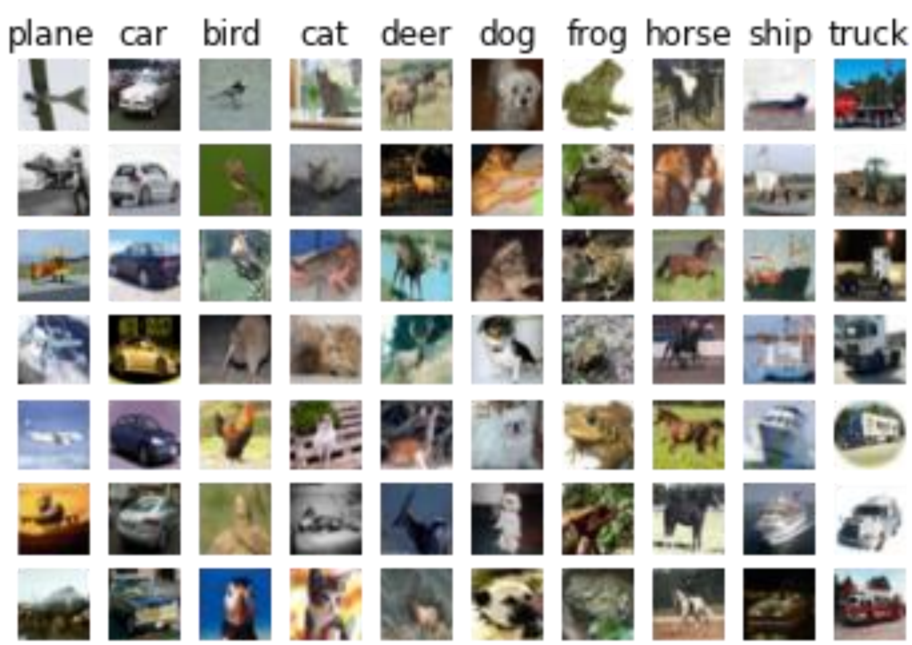
۸-

a)

داده های این سوال به این صورت هستند که داده های آموزشی به صورت batch هایی که هر کدام در فایل های یک تا پنج قرار گرفته اند موجود اند و یک فایل تست نیز وجود دارد. علاوه بر این هر سطر از این داده ها بیانگر یک پیکسل است که دارای ۱۰۲۴ مقدار برای هر کدام از رنگ های rgb است یعنی هر سطر دارای ۳۲*۳۲*۱۰۲۴ داده است.

```
def unpickle_batch(file):
    with open(file, 'rb') as batch_file:
        cifar = pickle.load(batch_file, encoding='bytes')
    X = cifar['data'.encode()]
    Y = cifar['labels'.encode()]
    X = X.reshape(10000, 3, 32, 32).transpose(0,2,3,1).astype("uint8")
    Y = np.array(Y)
    return X,Y


def load_CIFAR10(ROOT):
    x = []
    y = []
    for i in range(1,6):
        batch_file = ROOT+'/data_batch_'+str(i)
        Xb, Yb = unpickle_batch(batch_file)
        x.append(Xb)
        y.append(Yb)
    Xtrain = np.concatenate(x)
    Ytrain = np.concatenate(y)
    Xtest, Ytest = unpickle_batch(ROOT+'/test_batch')
    return Xtrain, Ytrain, Xtest, Ytest
```



b)

L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p \left(I_1^p - I_2^p\right)^2}$$

```python
def eucli_distance_function(x, X_train):
    return np.linalg.norm(X_train - x,axis=1)

def manhattan_distance_function(x, X_train):
    dist = []
    for i in range (0,len(X_train)):
        dist.append(np.abs(x - X_train[i]).sum())
    return dist
```

c)

به ازای هر نقطه تست، فاصله از تمام نقاط train را به کمک توابع تعریف شده حساب کرده و پس از مرتب کردن آن ها به ترتیب صعودی k تای اول را برداشته و لیبل های آن ها را مشاهده می کنیم. لیبلی که بیشترین تعداد را در میان k همسایه انتخاب شده داشت به عنوان جواب بازگردانده می شود.

```python
test_num = 1000
def KNN(X_train, y_train, X_test,k, distance_function):
    flatten_X_train = X_train.reshape(X_train.shape[0], 32 * 32 * 3)
    flatten_X_test =  X_test.reshape(X_test.shape[0], 32 * 32 * 3)[:test_num]
    y_pred = []
    for x in flatten_X_test:
        dist = distance_function(x,flatten_X_train)
        knn_idx = np.argsort(dist)[:k]
        knn_labels = y_train[knn_idx]
        y_pred.append(np.bincount(knn_labels).argmax())
    return y_pred
```

```python
def calculate_accuracy(X_train, y_train, X_test,y_test,k_list, distance_function):
    for k in k_list:
        y_pred = KNN(X_train, y_train, X_test, k,distance_function)
        print('k = %d, accuracy = %f' % (k,  np.mean(y_test[:len(y_pred)] == y_pred)))
```

d)

eucli_distance_function

```
[ ] k_list = [1,5,10,15,50,100]
```

```
[ ] calculate_accuracy(X_train, y_train, X_test,y_test,k_list,eucli_distance_function )

k = 1, accuracy = 0.209000
k = 5, accuracy = 0.211000
k = 10, accuracy = 0.204000
k = 15, accuracy = 0.212000
k = 50, accuracy = 0.190000
k = 100, accuracy = 0.180000
```

manhattan_distance_function

```
calculate_accuracy(X_train, y_train, X_test,y_test,k_list,manhattan_distance_function)

k = 1, accuracy = 0.281000
k = 5, accuracy = 0.273000
k = 10, accuracy = 0.284000
k = 15, accuracy = 0.279000
k = 50, accuracy = 0.305000
k = 100, accuracy = 0.279000
```

مشاهده می شود که استفاده از Manhattan distance نتایج بهتری داشته است. در فاصله ی اقلیدسی نتایج میان تعداد همسایه های ۱ تا ۱۵ تقریبا نزدیک به هم بوده است، در تعداد ۵۰ و ۱۰۰ کمی اختلاف بیشتر است و دقت کمتر شده است. در فاصله ی منهتن مشاهده می شود که بهترین جواب برای تعداد همسایه ۵۰ و در فاصله ی اقلیدسی به تعداد همسایه ۱۵ مربوط می شود.

- توجه شود که به علت کند بودن زمان اجرا نتایج تنها بر روی ۱۰۰۰ داده تست اعلام شده است.