

## گزارش تکلیف ۴ درس یادگیری ماشین

کسرا سینایی

شماره دانشجویی ۸۱۰۶۹۶۲۵۴

۱۰ دی ۱۴۰۰

## ۱ سوال یک

### الف

در روش KNN هرچه مقدار  $K$  را کوچکتر انتخاب کنیم پیچیدگی مدل و مرز تصمیم افزایش می‌یابد و به همین نسبت جنرالیزیشن مدل کمتر و حساسیت به نویز نیز بیشتر می‌شود. در واقع هر چه  $K$  کوچکتر باشد واریانس بیشتر خواهد بود. با افزایش  $K$  به حد بهینه‌ای برای مدل می‌رسیم و افزایش بیشتر آن منجر به خطای بایاس می‌شود. در تخمین پنجره پارزن هاپرپارامتر مدل طول پنجره در تابع کرنل می‌باشد. این پارامتر ( $h_n$ ) مانند یک smoothing factor عمل می‌کند. هر چه طول پنجره بزرگتر باشد شکل تابع به دست آمده برای توزیع متغیرها نرم‌تر می‌شود. برای  $h$  های بسیار کوچک تخمین نان پارامتری پارزن به تابع دلتا دیراک شبیه خواهد شد. به زبان دیگر انتخاب طول بزرگ برای پنجره منجر به خطای بایاس و انتخاب اندازه کوچک برای طول پنجره منجر به خطای واریانس می‌شود.

### ب

در روش‌های پارامتریک فرض می‌کنیم داده‌ها از یک توزیع شناخته شده و فرمول‌بندی شده مانند گوسی، پواسون و ... آمده‌اند و برای توصیف آن‌ها به تخمین پارامترهای مدل فرض شده می‌پردازیم. در روش‌های نان پارامتری هیچ فرض اضافی برای مدل حاکم بر داده‌های یادگیری قرار نمی‌دهیم و با قوانینی مانند نزدیک‌ترین همسایه، استفاده از تابع کرنل و ... توزیعی دلخواه به داده‌های یادگیری فیت می‌کنیم.

### ج

چون این روش‌ها نان پارامتریک هستند لازم است برای پیش‌بینی نقاط دیده نشده تمام مجموعه یادگیری را همراه با مدل نگه داریم. (در واقع مدل مستق و به کمک یک تابع و چند پارامتر تعریف نشده که بتوان با استفاده از آن پیش‌بینی کرد، بلکه مدل با استفاده از داده‌های مجموعه یادگیری در هر نقطه تعریف می‌شود) با افزایش ابعاد فیچرها برای دست‌یابی به تخمینی دقیق تعداد نقاط مورد نیاز در دیتاست به صورت نمایی افزایش می‌یابد و در مواردی که نتوانیم حجم داده‌ها را افزایش دهیم ممکن است دچار نفرین ابعادی شویم.

### د

در روش پنجره پارزن مفهوم حجم وابسته به کرنل استفاده شده است. برخی کرنل‌ها به تمامی نقاط موجود در دیتاست وزن می‌دهند (مانند کرنل گوسی) و برخی کرنل‌ها مانند hyper cube kernel تنها داده‌های موجود در ابر مکعب حول نقطه مورد نظر را برای طبقه‌بندی یا تخمین احتمال در نظر می‌گیرند. در روش KNN حجم با یک ابر کره که شعاع آن فاصله دورترین نقطه در بین  $K$  همسایه نقطه‌ای که می‌خواهیم احتمال آن را به دست بیاوریم، تعریف می‌شود.

## ۲ سوال دو

در این سوال می‌خواهیم نشان دهیم که سلول‌های voronoi تشکیل شده با الگوریتم نزدیک‌ترین همسایه محدب هستند. به ازای هر دو نقطه درون یک سلول، خط وصل کننده آن‌ها نیز داخل سلول قرار خواهد گرفت، فرض کنید  $x^*$  در یک سلول با لیبل مشخص واقع شده باشد و  $y$  نیز نقطه‌ای با یک لیبل متفاوت از  $x^*$  باشد. در این حالت ابر صفحه‌ای وجود دارد که فضا را طوری تقسیم می‌کند که یک طرف آن نقاط به  $x^*$  نزدیک تر هستند و طرف دیگر آن نقاط به  $y$  نزدیک ترند. اگر نقاط  $x(0)$  و  $x(1)$  در سلول متعلق به  $x^*$  قرار داشته باشند خطی که این دو نقطه را به هم وصل می‌کند نیز به صورت:  $x(\lambda) = (1 - \lambda)x(0) + \lambda x(1); 0 \leq \lambda \leq 1$  خواهد بود. این خط نیز به طور کامل درون سلول متعلق به نقطه  $x^*$  قرار خواهد گرفت. چون فضای نصف شده توسط ابر صفحه محدب است، تمامی نقاط موجود بر روی خط تعریف شده  $(x(\lambda))$  نیز به  $x^*$  نزدیک تر هستند تا  $y$ .

این موضوع به ازای تمام نقاط موجود در Voronoi Cell متعلق به نقطه  $x^*$  صادق است. همچنین برای سایر سمپل‌ها نیز صادق است (شرطی برای  $y$  نداریم) بنابراین Voronoi Cell تشکیل شده با قانون نزدیک‌ترین همسایه همواره محدب می‌باشند.

۳ سوال سه

الف

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \rightarrow P_n(x) = \frac{1}{nh_n} \sum_{i=1}^n \varphi\left(\frac{x-x_i}{h_n}\right)$$

$$\bar{P}_n(x) = E\{P_n(x)\} = \frac{1}{nh_n} \sum_{i=1}^n E\left\{\varphi\left(\frac{x-x_i}{h_n}\right)\right\} = \frac{1}{nh_n} \int_{-\infty}^{\infty} \varphi\left(\frac{x-v}{h_n}\right) P(v) dv$$

$$= \frac{1}{h_n} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-v}{h_n}\right)^2\right) \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left(\frac{v-\mu}{\sigma}\right)^2\right) dv$$

$$= \frac{1}{2\pi h_n \sigma} \exp\left(-\frac{1}{2}\left(\frac{x^2}{h_n^2} + \frac{\mu^2}{\sigma^2}\right)\right) \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}v^2\left(\frac{1}{h_n^2} + \frac{1}{\sigma^2}\right) - 2v\left(\frac{x}{h_n^2} + \frac{\mu}{\sigma^2}\right)\right) dv$$

$$= \frac{1}{2\pi h_n \sigma} \exp\left(-\frac{1}{2}\left(\frac{x^2}{h_n^2} + \frac{\mu^2}{\sigma^2}\right) + \frac{1}{2}\frac{\alpha^2}{\theta^2}\right) \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}\left(\frac{v-\alpha}{\theta}\right)^2\right) dv$$

$$\theta^2 = \frac{1}{\frac{1}{h_n^2} + \frac{1}{\sigma^2}} = \frac{(h_n \sigma)^2}{h_n^2 + \sigma^2}, \quad \alpha = \theta^2 \left(\frac{x}{h_n^2} + \frac{\mu}{\sigma^2}\right)$$

$$\Rightarrow \bar{P}_n(x) = \frac{\sqrt{2\pi}\theta}{2\pi h_n \sigma} \exp\left(-\frac{1}{2}\left(\frac{x^2}{h_n^2} + \frac{\mu^2}{\sigma^2}\right) + \frac{1}{2}\frac{\alpha^2}{\theta^2}\right)$$

$$= \frac{1}{\sqrt{2\pi} h_n \sigma} \frac{h_n \sigma}{\sqrt{h_n^2 + \sigma^2}} \exp\left(-\frac{1}{2}\underbrace{\left(\frac{x^2}{h_n^2} + \frac{\mu^2}{\sigma^2} - \frac{\alpha^2}{\theta^2}\right)}_A\right)$$

$$A = \frac{x^2}{h_n^2} + \frac{\mu^2}{\sigma^2} - \frac{\theta^4}{\theta^2} \left(\frac{x}{h_n^2} + \frac{\mu}{\sigma^2}\right)^2 = \frac{(x-\mu)^2}{h_n^2 + \sigma^2} \Rightarrow \bar{P}_n(x) = \frac{1}{\sqrt{2\pi} \sqrt{h_n^2 + \sigma^2}} \exp\left(-\frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right) \checkmark$$

$$\begin{aligned}
P(x) - \bar{P}(x) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}\right) - \frac{1}{\sqrt{2\pi} \sqrt{h_n^2 + \sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right) \\
&= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right) \left[1 - \frac{\sigma}{\sqrt{h_n^2 + \sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2} + \frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}\right)\right] \\
&= P(x) \left[1 - \frac{1}{\sqrt{1 + \left(\frac{h_n}{\sigma}\right)^2}} \exp\left(-\frac{(x-\mu)^2}{2} \left(\frac{1}{h_n^2 + \sigma^2} - \frac{1}{\sigma^2}\right)\right)\right] \\
&= P(x) \left[1 - \frac{1}{\sqrt{1 + \left(\frac{h_n}{\sigma}\right)^2}} \exp\left(\frac{1}{2} \frac{h_n^2 (x-\mu)^2}{h_n^2 + \sigma^2}\right)\right]
\end{aligned}$$

$$\text{if } h_n \rightarrow 0 : \frac{1}{\sqrt{1 + \left(\frac{h_n}{\sigma}\right)^2}} \approx 1 - \frac{1}{2} \left(\frac{h_n}{\sigma}\right)^2, \quad \exp\left(\frac{h_n^2}{2\sigma^2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right) = 1 + \frac{h_n^2}{2\sigma^2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}$$

$$\Rightarrow P(x) - \bar{P}(x) = P(x) \left[1 - \left(1 - \frac{1}{2} \left(\frac{h_n}{\sigma}\right)^2\right)^2 \left(1 + \frac{h_n^2}{2\sigma^2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right)\right]$$

$$= P(x) \left[1 - 1 + \frac{1}{2} \frac{h_n^2}{\sigma^2} - \frac{h_n^2}{2\sigma^2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right] = \frac{1}{2} \left(\frac{h_n}{\sigma}\right)^2 \left[1 - \frac{(x-\mu)^2}{\sigma^2}\right] P(x) \checkmark$$

2

$$\begin{aligned} \text{Var}\{P_n(x)\} &= \text{Var}\left\{\frac{1}{nh_n} \sum_{i=1}^n \varphi\left(\frac{x-x_i}{h_n}\right)\right\} = \frac{1}{n^2 h_n^2} \sum_{i=1}^n \text{Var}\left\{\varphi\left(\frac{x-x_i}{h_n}\right)\right\} \\ &= \frac{1}{nh_n^2} \text{Var}\left\{\varphi\left(\frac{x-v}{h_n}\right)\right\} = \frac{1}{nh_n^2} \left( \underbrace{E\left\{\varphi^2\left(\frac{x-v}{h_n}\right)\right\}}_A - \underbrace{E\left\{\varphi\left(\frac{x-v}{h_n}\right)\right\}^2}_B \right) \end{aligned}$$

$$\begin{aligned} A &= \int \varphi^2\left(\frac{x-v}{h_n}\right) P(v) dv = \int_{-\infty}^{\infty} \frac{1}{2\pi} \exp\left[-\frac{(x-v)^2}{h_n^2}\right] \exp\left[-\frac{1}{2}\left(\frac{v-\mu}{\sigma}\right)^2\right] \frac{1}{\sqrt{2\pi}\sigma} dv \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-v}{h_n/\sqrt{2}}\right)^2\right] \exp\left[-\frac{1}{2}\left(\frac{v-\mu}{\sigma}\right)^2\right] \frac{1}{\sqrt{2\pi}\sigma} dv \\ &= \frac{h_n/\sqrt{2}}{2\pi \sqrt{h_n^2/2 + \sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{h_n^2/2 + \sigma^2}\right) \quad \text{if } h_n \rightarrow 0: \sqrt{\frac{h_n^2}{2} + \sigma^2} \simeq \sigma \\ &\rightarrow \frac{1}{nh_n^2} E\left\{\varphi^2\left(\frac{x-v}{h_n}\right)\right\} = \frac{1}{nh_n} \frac{1}{2\pi} \exp\left(-\frac{1}{2} \left(\frac{x-\mu}{\sigma}\right)^2\right) = \frac{1}{2nh_n\sqrt{\pi}} P(x) \end{aligned}$$

$$B = h_n^2 \frac{1}{\sqrt{2\pi} \sqrt{h_n^2 + \sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{h_n^2 + \sigma^2}\right) \simeq \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2} \left(\frac{x-\mu}{\sigma}\right)^2\right) \simeq 0$$

$$\Rightarrow \text{Var}\{P_n(x)\} = \frac{P(x)}{2nh_n\sqrt{\pi}} \quad \checkmark$$

## ۴ سوال چهار

$$\alpha_k > 0 \rightarrow D(a, b) \geq 0 \quad \text{non-negative } \checkmark$$

$$\text{if } a = b \rightarrow \text{all components of } \Sigma \text{ become } 0 \rightarrow D(a, b) = 0 \quad \text{reflexivity } \checkmark$$

$$(a_k - b_k)^2 = (b_k - a_k)^2 \rightarrow D(a, b) = D(b, a) \quad \text{Symmetry } \checkmark$$

$$\text{triangle inequality: } D(a, b) + D(b, c) \geq D(a, c)$$

$$\text{if } A = \text{diag}(\alpha_1, \dots, \alpha_d) \rightarrow D(a, b) = \|A(a-b)\|$$

$$\text{if } x = a - c, y = c - b$$

$$\text{Proove: } \|Ax\| + \|Ay\| \geq \|A(x+y)\|$$

$$\text{if } Ax = x', Ay = y' \rightarrow \|x'\| + \|y'\| \geq \|x' + y'\|$$

$$\xrightarrow{\text{2 way}} (\|x'\| + \|y'\|)^2 = \|x'\|^2 + \|y'\|^2 + 2\|x'\|\|y'\| \geq \|x'\|^2 + \|y'\|^2 + 2\sum_{i=1}^d x'_i y'_i = \|x' + y'\|^2$$

$$\text{if } \text{is correct} \rightarrow \|x'\|\|y'\| \geq \sum_{i=1}^d x'_i y'_i \quad \checkmark \quad \text{based on Cauchy-Schwartz is always true}$$

## سوال پنج ۵

$$P(w_1) = P(w_2) = \frac{1}{2} \quad P(w_1|x) = \begin{cases} 1 & \|x\| \leq 1 \\ 0 & \text{otherwise} \end{cases} \quad P(w_2|x) = \begin{cases} 1 & \|x-a\| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

الف  $P_n(e) = P\{w_1 \text{ is true \& } w_2 \text{ is most frequent}\}$   
 $+ P\{w_2 \text{ is true \& } w_1 \text{ is most frequent}\}$

$$P_n(e) = 2P\{w_1 \text{ is true \& } w_2 \text{ is most frequent}\}$$

$$\begin{aligned} \rightarrow P_n(e) &= 2P\left\{\#w_1 \leq \frac{k-1}{2}\right\} = 2\left(\frac{1}{2} \sum_{j=0}^{\frac{k-1}{2}} P\{\#w_1 = j \text{ \& } \#w_2 = n-j\}\right) \\ &= \sum_{j=0}^{\frac{k-1}{2}} \binom{n}{j} \frac{1}{2^j} \frac{1}{2^{n-j}} = \left(\frac{1}{2^n}\right) \sum_{j=0}^{\frac{k-1}{2}} \binom{n}{j} \end{aligned}$$

ب  $P_n(e;1) = \frac{1}{2^n} < \frac{1}{2^n} \sum_{j=0}^{\frac{k-1}{2}} \binom{n}{j}$   $\xrightarrow{\text{if } k \geq 2 \text{ is greater than 1}}$   $\xrightarrow{\text{for } k \geq 2 ; P_n(e;1) < P_n(e;k)}$

ج  $B(n, p) \rightarrow \text{Binomial Distribution}$

$$P_n(e) = \frac{1}{2^n} \sum_{j=0}^{\frac{k-1}{2}} \binom{n}{j} = P\{B(n, \frac{1}{2}) \leq \frac{k-1}{2}\} = P\{Y_1 + \dots + Y_n \leq \frac{k-1}{2}\}$$

$$P\{Y_i = 1\} = P\{Y_i = 0\} = \frac{1}{2} ; \text{ based on convergence criteria of KNN: } k \leq \frac{a}{\sqrt{n}}$$

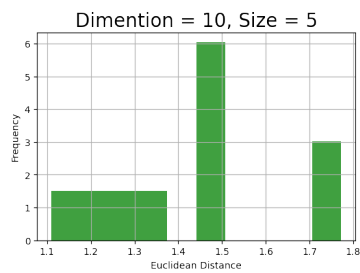
$$\Rightarrow P_n(e) \leq P\{Y_1 + \dots + Y_n \leq \frac{\frac{a}{\sqrt{n}} - 1}{2}\} = P\{Y_1 + \dots + Y_n \leq 0\} = 0$$

$\uparrow$   
 $n \rightarrow \infty$

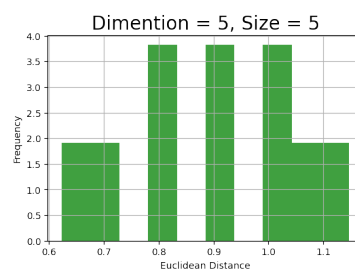


## ۶ سوال شش (شبیه سازی)

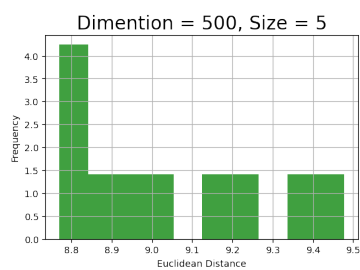
الف



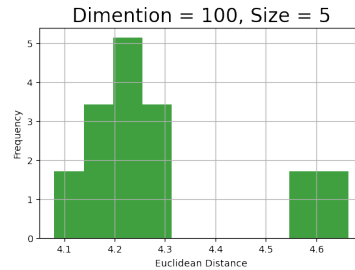
(ب) ۱۰ بعدی



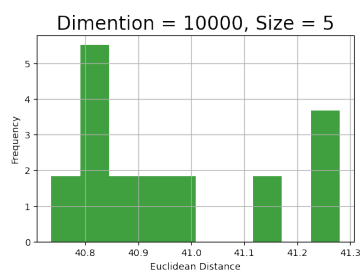
(آ) ۵ بعدی



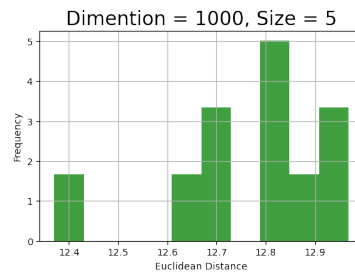
(د) ۵۰۰ بعدی



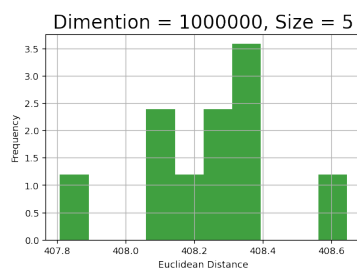
(ج) ۱۰۰ بعدی



(و) ۱۰۰۰۰ بعدی



(ه) ۱۰۰۰ بعدی

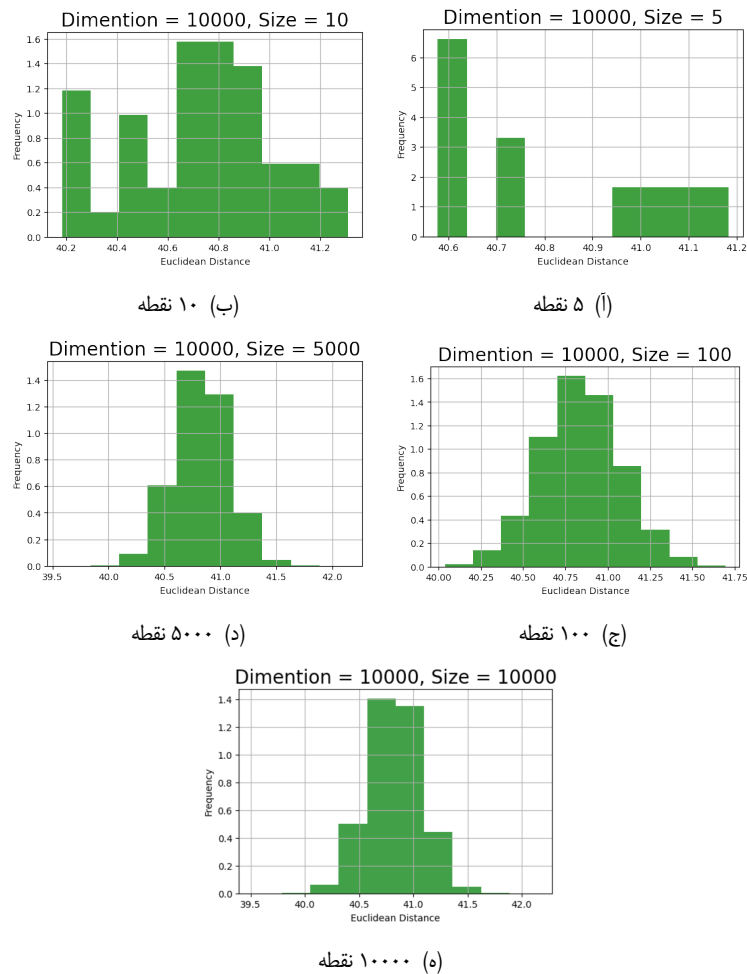


(ز) ۱۰۰۰۰۰ بعدی

شکل ۱: خواسته‌های قسمت الف

در این سوال با استفاده از تابع `rndom` کتابخانه `numpy` و تابع `hist` از کتابخانه `matplotlib` هیستوگرام فاصله نقاط را به دست می‌آوریم. برا این کار تابع `do task` پیاده سازی شده است که تعداد و بعد نقاط مورد نظر را در ورودی گرفته و نمودارها را رسم می‌کند.

ب



شکل ۲: خواسته‌های قسمت ب

ج

اولین مسئله‌ای که به چشم می‌آید افزایش مدت زمان محاسبه فاصله نقاط از یکدیگر می‌باشد. همانطور که می‌دانیم پیچیدگی محاسباتی الگوریتم KNN برابر با  $O(mn)$  می‌باشد. (m تعداد فیچر و n تعداد داده‌های یادگیری می‌باشد). یکی از مشکلات اساسی الگوریتم KNN این است که در ابعاد بالا برای دقیق شدن تخمین باید تعداد داده‌ها را نیز افزایش دهیم که این موجب افزایش زمان طبقه بندی می‌شود. این مشکل همان نفرین ابعادی (curse of dimensionality)

است.  
م

## ۷ سوال هفت (شبیه سازی)

در این سوال با استفاده از الگوریتم KNN تصاویر اعداد دست نویس فارسی را طبقه بندی می‌کنیم.

### الف

برای لود کردن دیتا ست از روی فایل‌های cdb از اسکریپت پایتون معرفی شده در لینک قرار داده شده در صورت سوال استفاده می‌کنیم. یکی از توابع موجود در این اسکریپت read\_hoda\_dataset می‌باشد. آرگومان‌های ورودی که آدرس فایل دیتاست و سائز تصاویر می‌باشد را مطابق توضیحات پیچ گیت‌هاب به تابع می‌دهیم. آرگومان reshape را نیز True قرار می‌دهیم و آرگومان one\_hot را False، در این صورت فیچرها که پیکسل عکس‌های  $32 \times 32$  هستند فلت شده و لیبل‌ها نیز یک عدد بین ۰ و ۹ خواهند بود. در شکل ۳ سه نمونه از تصاویر متعلق به هر کلاس نمایش داده شده‌اند. (رنگ‌ها غیر واقعی می‌باشند، تمام تصاویر یک کاناله و gray scale هستند).

### ب

در این قسمت الگوریتم KNN را به صورت دستی پیاده سازی می‌کنیم. به این منظور یک کلاس به نام KNN با سه متد زیر پیاده سازی شده‌است:

- constructor: این تابع تنها هاپیر پارامتر مدل که همان k یا تعداد همسایه‌ها می‌باشد را به همراه دیتاست و فیچرها می‌گیرد و در متغیرهای کلاس ذخیره می‌کند. تعداد سمپل‌ها و ابعاد فیچر نیز از دیگر متغیرهای این کلاس هستند که از روی ورودی‌های این تابع به دست می‌آیند. لازم به ذکر است که ورودی‌های مربوط به سمپل‌ها و لیبل‌ها باید آرایه numpy باشند.

- dist: این تابع به صورت برداری فاصله اقلیدسی دو بردار که آرگومان‌ها ورودی آن هستند را حساب می‌کند.

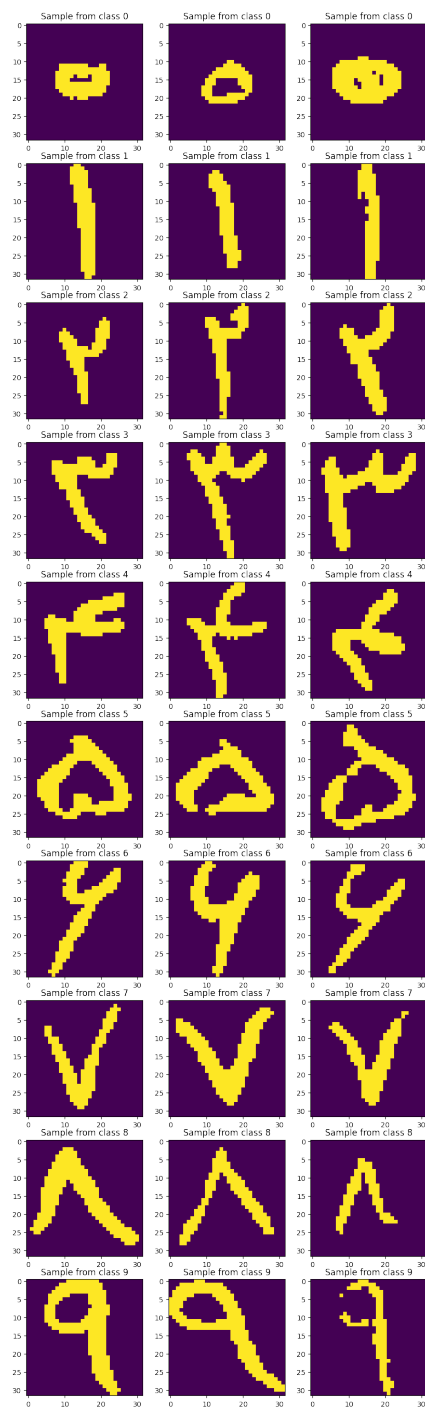
- predict: در این تابع برای هر سطر از متغیر ورودی (x) تخمین نان‌پارامتریک مدل را به دست می‌آوریم. ابتدا با استفاده از متد dist و تابع argsort کتابخانه numpy همسایه‌های نقاط را به دست می‌آوریم. در نهایت با استفاده از تابع unique و argmax پر تکرارترین لیبل در بین k همسایه نزدیک نقطه مورد بررسی را به دست آورده و آن نقطه را در همان طبقه قرار می‌دهیم.

سرعت محاسبه تابع predict در مقایسه با توابع آماده کتابخانه scikit learn به مقدار قابل توجهی کمتر است. لیبل کردن ۲۰۰۰ دیتا از مجموعه تست حدوداً ۳۰ دقیقه زمان برد. نتایج طبقه بندی در جدول ۱ و شکل ۴ آورده شده است.

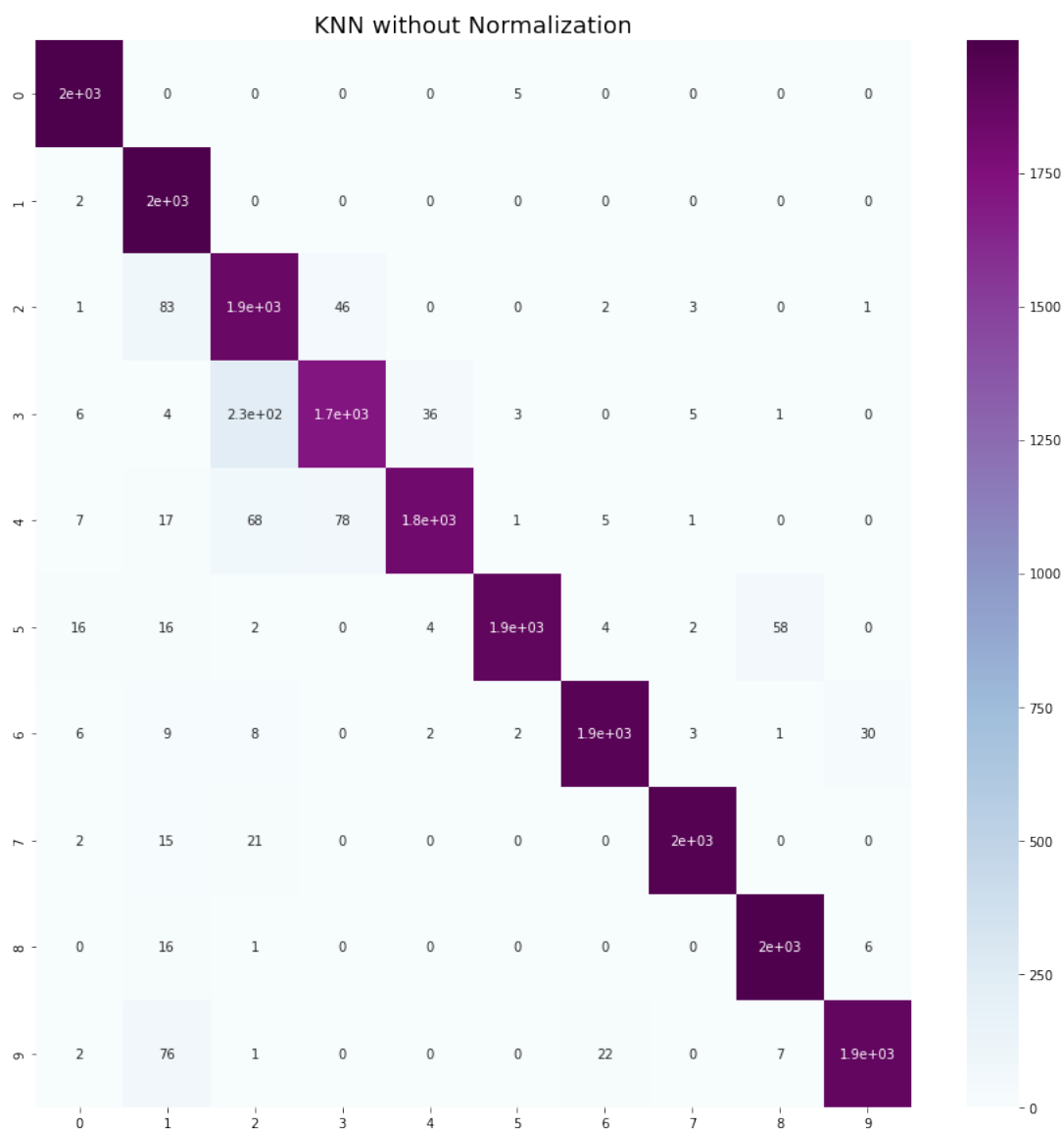
label	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹
f1 score	۸۳.۹۸	۳۷.۹۴	۸۲.۸۸	۲۸.۸۹	۳۳.۹۴	۱۰.۹۷	۶۳.۹۷	۶۹.۹۸	۷۷.۹۷	۳۰.۹۶

جدول ۱: نتایج طبقه بندی KNN با داده‌های نرمالایز نشده

همانطور که از ماتریس آشفستگی و f1 score نیز مشخص است، دقت تمایز بین برخی کلاس‌ها پایین تر است. برای نمونه اعداد ۲ و ۳ فارسی شباهت زیادی به یک دیگر دارند و در مجموعه تست تعدادی از نمونه‌های این دو کلاس اشتباه طبقه بندی شده‌اند. کلاس ۹ و ۱ نیز تعداد اشتباه به نسبت بالایی دارند.



شکل ۳: چند نمونه از تصاویر موجود در دیتاست



شکل ۴: Confusion Matrix طبقه بند KNN با داده‌های نرمالایز نشده

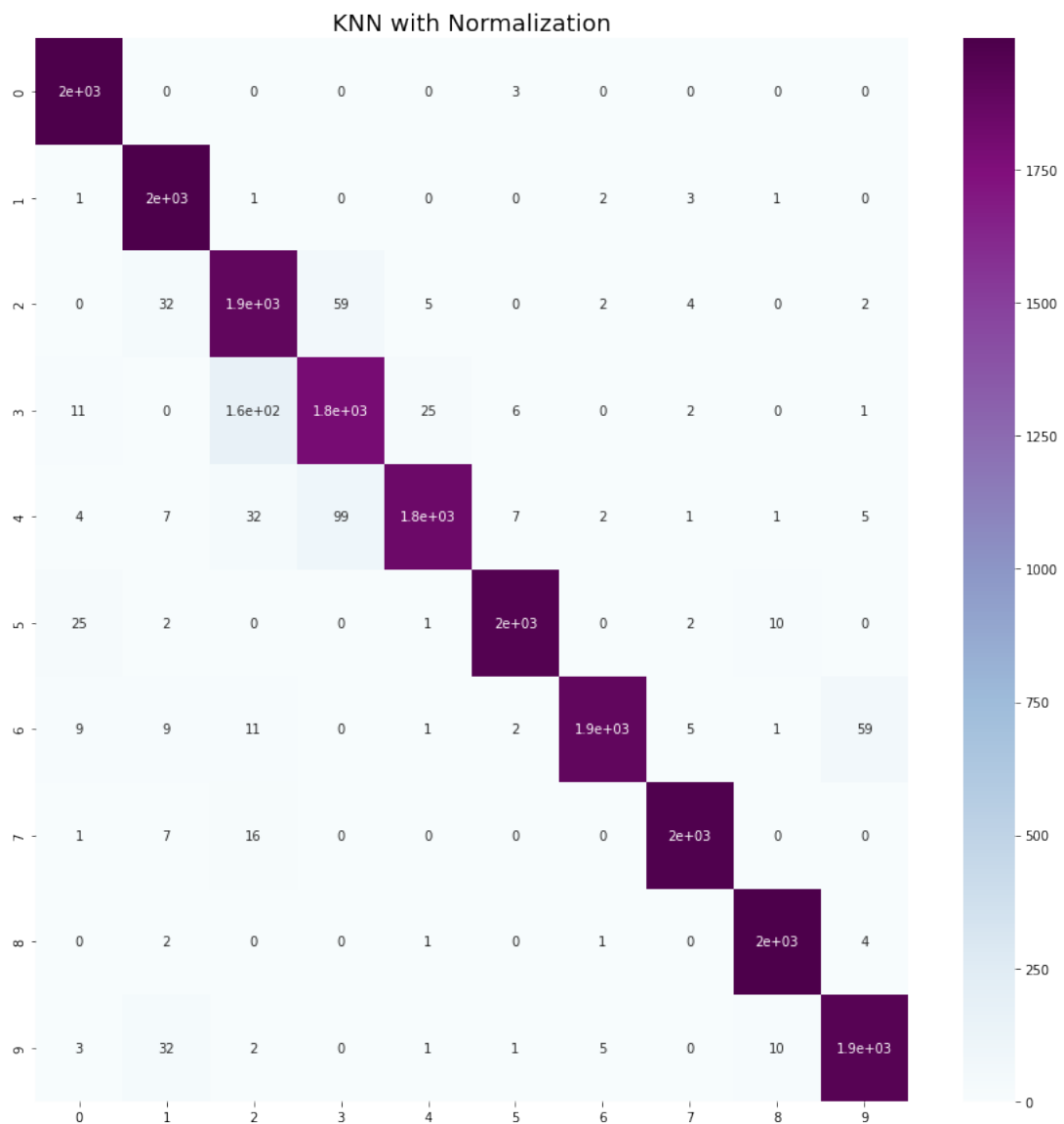
ج

در این قسمت بردار فیچر را با استفاده از رابطه زیر نرمال می‌کنیم:

$$\mathbf{D} = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$$

$$\vec{x}_i := \frac{\vec{x}_i}{\sum_{j=1}^d x_i^j}$$

با استفاده از رابطه فوق دو بردار  $X_{train}$  و  $X_{test}$  نرمال شده و در آرایه‌ی جدیدی ذخیره می‌کنیم و مجدداً عکس‌ها را طبقه‌بندی می‌کنیم. نتایج طبقه‌بندی با داده‌های نرمال شده بر روی مجموعه تست در شکل ۵ نشان داده شده است.



شکل ۵: Confusion Matrix طبقه‌بند KNN با داده‌های نرمال‌ایز شده

از ماتریس آشفتگی و جدول ۲ می‌توان نتیجه گرفت که در مقایسه با نتایج قسمت قبل بهبود اندکی در عملکرد طبقه‌بند به چشم می‌خورد. در تمامی قسمت‌های این تمرین از تعداد همسایه  $K = 5$  در طبقه‌بند استفاده شده است.

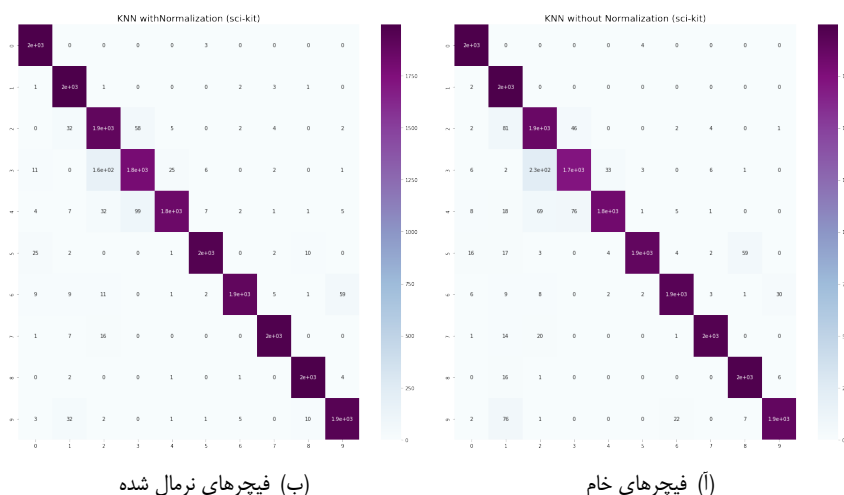
label	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹
F1 Score (Raw Features)	۸۳.۹۸	۳۷.۹۴	۸۲.۸۸	۲۸.۸۹	۳۳.۹۴	۱۰.۹۷	۶۳.۹۷	۶۹.۹۸	۷۷.۹۷	۳۰.۹۶
F1 Score (Normalized)	۵۹.۹۸	۵۷.۹۷	۹۷.۹۱	۶۷.۹۰	۰۴.۹۵	۵۱.۹۸	۲۱.۹۷	۹۷.۹۸	۲۲.۹۹	۸۸.۹۶

جدول ۲: مقایسه عملکرد طبقه‌بند KNN با داده‌های پردازش نشده و داده‌های نرمال شده

## ۵

در این قسمت با استفاده از کلاس آماده KNeighborsClassifier از کتابخانه Scikit Learn طبقه‌بندی را برای داده‌های پردازش نشده و پردازش شده انجام می‌دهیم. عملکرد طبقه‌بند با متریک F1 Score و همچنین ماتریس‌های آشفتگی در شکل ۶ و جدول ۳ نشان داده شده‌اند.

برای ساخته نمونه از این کلاس تنها کافی است مقدار n neighbours را در کانستراکتور تنظیم کنیم و سپس با استفاده از تابع fit داده‌های یادگیری و لیبیل آن‌ها را به آبجکت تشکیل شده بدهیم. در نهایت با استفاده از دستور predict داده‌های تست را طبقه‌بندی کنیم.



شکل ۶: طبقه‌بندی با کلاس کتابخانه scikit learn

label	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹
F1 Score (Raw Features)	۸۳.۹۸	۴۴.۹۴	۸۴.۸۸	۵۰.۸۹	۳۷.۹۴	۰۵.۹۷	۶۰.۹۷	۶۹.۹۸	۷۵.۹۷	۳۰.۹۶
F1 Score (Normalized)	۵۹.۹۸	۵۷.۹۷	۹۹.۹۱	۷۰.۹۰	۰۴.۹۵	۵۱.۹۸	۲۱.۹۷	۹۷.۹۸	۲۲.۹۹	۸۸.۹۶

جدول ۳: نتایج طبقه‌بندی با استفاده از کتابخانه scikit learn



## ۸ سوال هشت (شبیه سازی)

در این تمرین به پیاده سازی روش نان پارامتری پنجره پارزن می پردازیم. داده های موجود در فایل اکسل را با دستورات کتابخانه pandas باز کرده و ستون duration را در یک آرایه numpy ذخیره می کنیم.

برای پیاده سازی طبقه بند یک کلاس به نام parzen پیاده سازی شده است. این کلاس چهار متد برای اجرای الگوریتم طبقه بندی و پیش بینی احتمال نقاط جدید دارد.

- constructor: برای ایجاد یک نمونه از این کلاس لازم است داده های یادگیری، نوع کرنل و اندازه پنجره برای فضاخوانی این تابع ارسال کرد. تعداد داده های یادگیری باید حتماً آرایه numpy باشند.

- gaussian kernel: این تابع با استفاده از رابطه  $\phi(\mathbf{x}) = \frac{1}{(\sqrt{2\pi})^d h_n^d} \exp \left[ -\frac{1}{2} \left( \frac{\mathbf{x} - \mathbf{x}_i}{h_n} \right)^2 \right]$  کرنل گوسی را برای بردار ورودی حساب می کند.

- hypercube kernel: برای تست عملکرد طبقه بند قبل از یادگیری و تست بر روی داده های اصلی با استفاده از کرنل مکعبی تابع prob تست شده است. برای استفاده از این کرنل در کانستراکتور نمونه ساخته شده از این تابع باید آرگومان آخر را "cube" بگذاریم.

- prob: این تابع با استفاده از کرنل مشخص شده در کانستراکتور و رابطه  $p_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \frac{1}{h^d} \phi \left[ \frac{\mathbf{x} - \mathbf{x}_i}{h_n} \right]$  احتمال مشاهده هر سطر از بردار ورودی را با توجه به داده های دریافت شده از تابع کانستراکتور به دست می آورد.

- plot dist: این تابع با استفاده از دستورات displot و line plot از کتاب خانه seaborn هیستوگرام داده های یادگیری را به همراه تابع توزیع احتمال تخمین زده شده را رسم می کند.

## الف

با استفاده از کلاس پیاده سازی شده با کرنل گوسی و  $h = 10$  توزیع تابع را برای داده های تد به دست آمده و نتیجه در شکل ۷ نشان داده شده است.

## ب

مانند بخش الف این تمرین با تغییر هایپرپارامتر طبقه بند (همان طول پنجره) یادگیری و تخمین توزیع را انجام می دهیم. نتایج در شکل ۸ نشان داده شده اند. اینکه با افزایش  $h_n$  تابع توزیع پیش بینی شده نرم تر می شود کاملاً مشخص است.

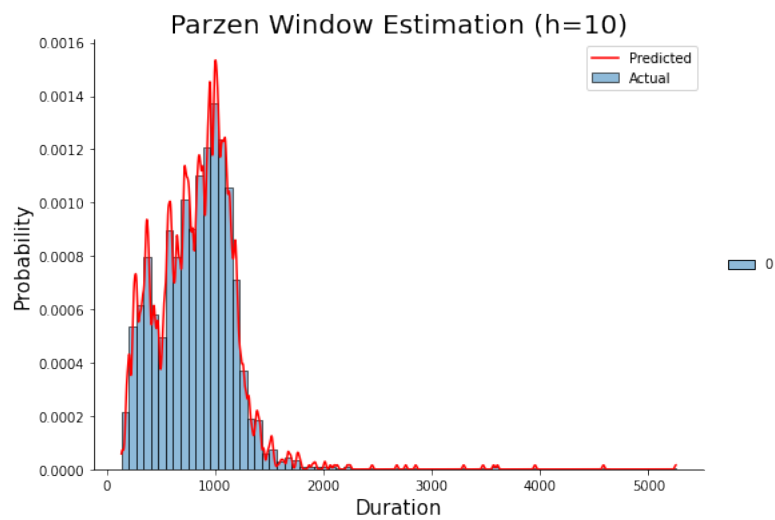
## ج

در این قسمت داده ها را به طور تدریجی به تخمین زن می دهیم تا روند تغییر و همگرا شدن به توزیع اصلی را ببینیم. این نمودار در شکل ۹ آورده شده است.

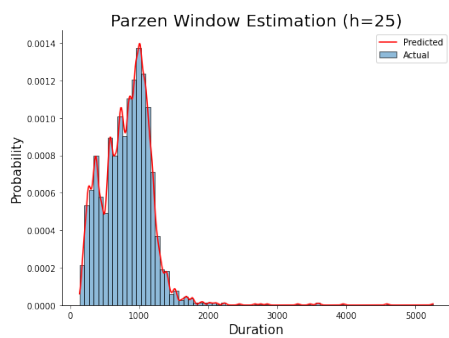
## د

در این قسمت با استفاده از KernelDensity از کتابخانه scikit learn نتایج قسمت الف را صحت سنجی می کنیم.

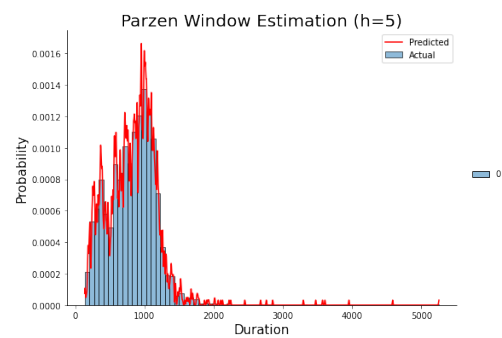
یک آبجکت KernelDensity ساخته و طول پنجره را در آرگومان ورودی (band width) تنظیم می کنیم. سپس داده های یادگیری را با فراخوانی تابع fit به مدل می دهیم. در نهایت می توان با تابع score samples احتمال داده های تست را حساب کرد. این تابع لگاریتم طبیعی احتمال را در خروجی باز می گرداند، پس لازم است آن ها را با تابع exp از کتابخانه numpy به احتمال تبدیل کرد.



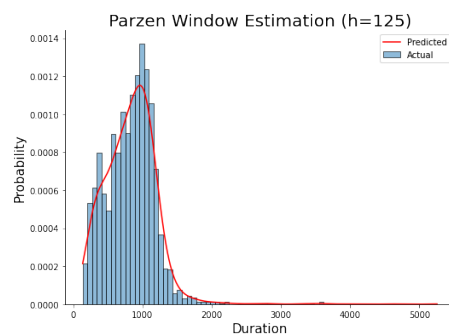
شکل ۷: توزیع تخمین زده شده با  $k = 10$



(ب)  $h_n = 25$

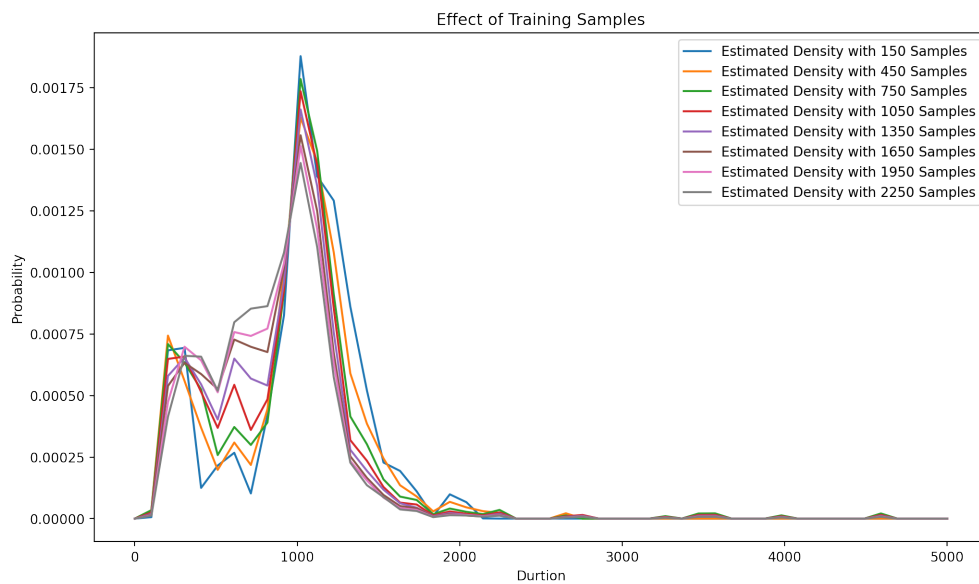


(ا)  $h_n = 5$

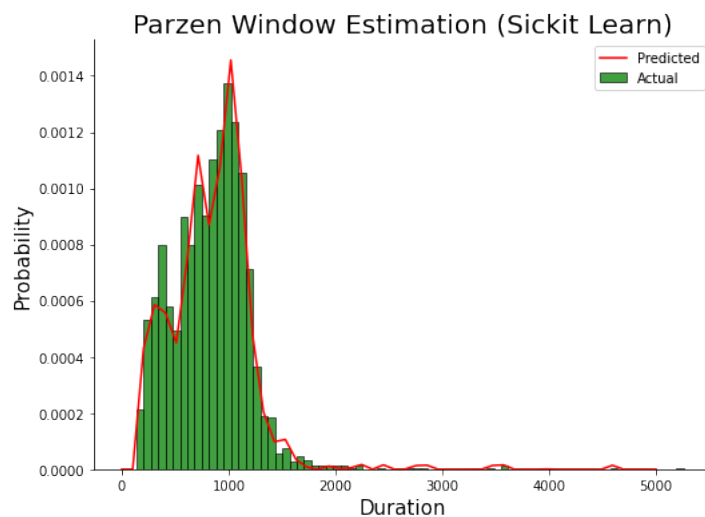


(ج)  $h_n = 125$

شکل ۸: توزیع‌های تخمین زده شده با طول پنجره متفاوت



شکل ۹: روند همگرایی تخمین نان پارامتری پنجره پازن



شکل ۱۰: تخمین نان پارامتری پازن با استفاده از کتابخانه scikit learn