به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده مهندسی برق و کامپیوتر

سیستم های هوشمند

تمرین شماره 5

نیما زمان پور

810198407

دی ماه 1401

فهرست سوالات

تمرین 1

بخش 1

.1

با پیاده سازی الگوریتم گفته شده در سوال و اجرای هر k برای 10 هزار بار، بهترین منشی با مقدار $K = 35$ و احتمال $\Pr = 36.96\%$ بدست می‌آید.

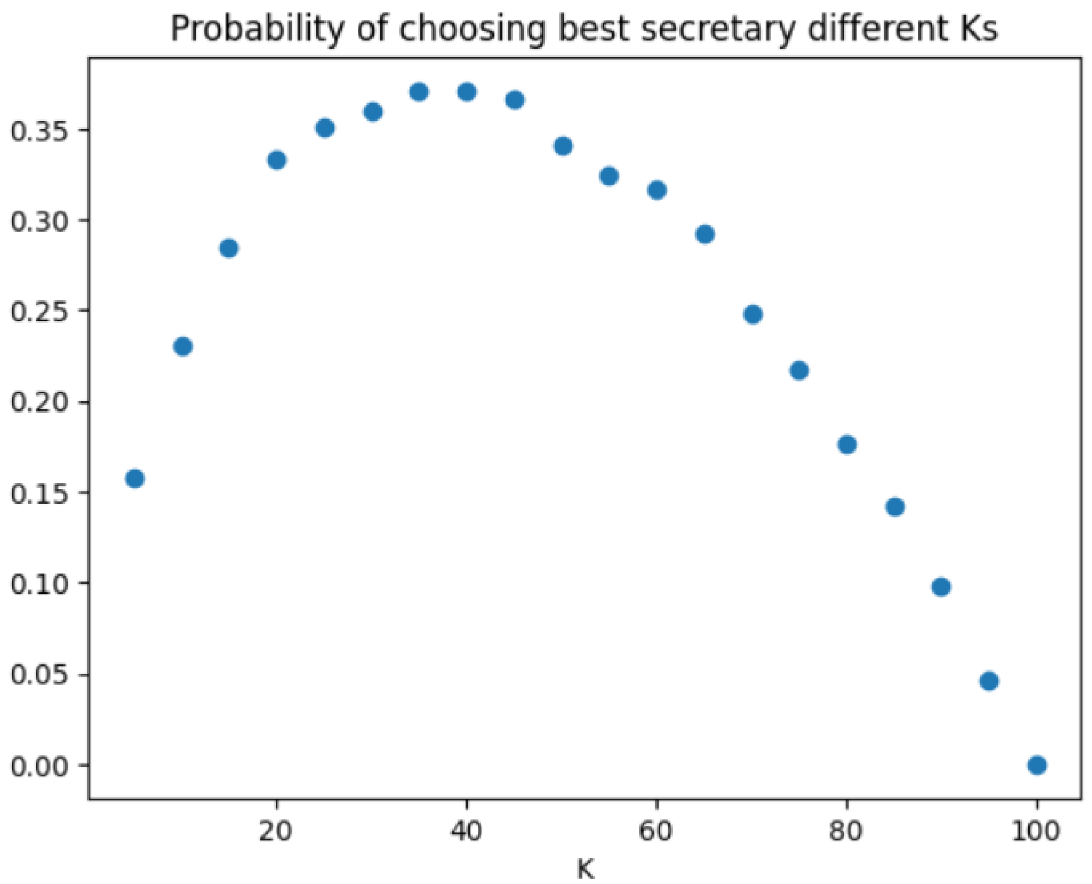


Figure 1 احتمال انتخاب بهترین کاندید بعد از رد کردن **k** کاندید اول برای **K** مختلف

.2

حال با بدست آوردن k بهینه، اینبار متغیر را N می‌گذاریم. این احتمالات با بیشتر شدن N و بالاتر رفتن دقت شبیه سازی به عدد $\frac{1}{e}$ میل می‌کند.

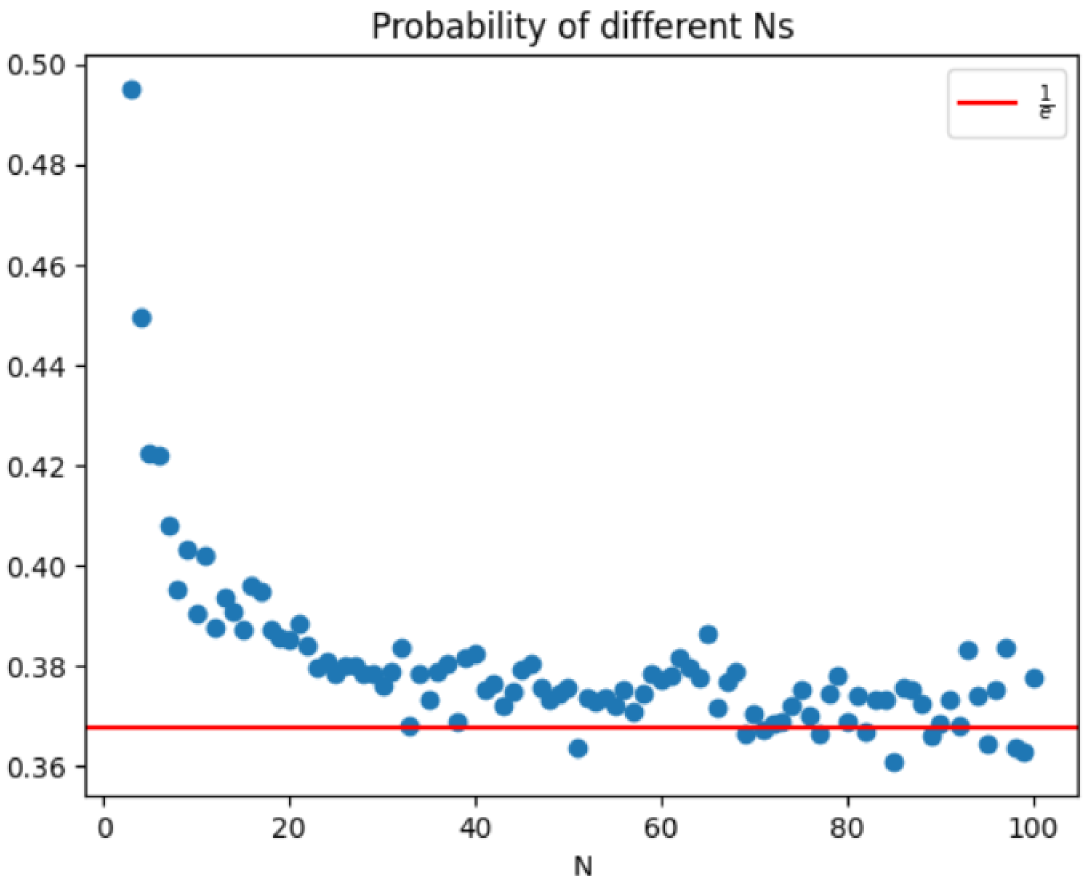


Figure 2 احتمال انتخاب بهترین کاندید بعد از رد کردن **k** کاندید اول برای **N** مختلف

.3

برای این کار احتمال انتخاب بهترین کاندید را با پارامتر k بصورت زیر تعریف می‌کنیم:

$$P(k) = \sum_{i=1}^{n} P\left(\text{کاندید } i\text{ام بهترین است} \cap \text{کاندید } i\text{ام انتخاب شده است}\right) =$$

$$\sum_{i=1}^{n} P\left(\text{کاندید } i\text{ام انتخاب شده است} | \text{کاندید } i\text{ام بهترین است}\right) . P\left(\text{کاندید } i\text{ام بهترین است}\right)$$

حالا احتمال بهترین بودن کاندید iام برابر $\frac{1}{n}$ است. چون توزیع یکنواخت است. حال به سراغ مشخص کردن عبارت دیگر می‌رویم. برای این کار باید ترتیب به گونه‌ای باشد که در افراد قبل از بهترین شخص، همه افراد برتر در k شخص اول باشند تا انتخاب نشوند و مستقیم بهترین فرد کل کاندید ها انتخاب شود.

$$= \sum_{i=k+1}^{n} P\left(\text{کاندید } i\text{ام بهترین است|بهترینِ } i-1 \text{ امین افراد؛ در } k \text{ شخص اول قرار دارند}\right) . P\left(\text{کاندید } i\text{ام بهترین است}\right)$$

$$= \frac{1}{n}\sum_{i=k+1}^{n}\frac{k}{i-1} = \frac{k}{n}\sum_{i=k+1}^{n}\frac{1}{i-1} \rightarrow P(K) = \frac{k}{n}\sum_{i=k+1}^{n}\frac{1}{i-1}$$

در حالت حدی $n \rightarrow \infty$ جمع تبدیل به انتگرل می‌شود:

$$\frac{1}{n} \rightarrow dt ==> P(k) = k\int_{k}^{1}\frac{dt}{t} = -klnk$$

حال برای بدست آوردن مقدار بهینه k از انتگرال بالا مشتق گرفته و مساوی صفر قرار می‌دهیم:

$$P'(k) = -(lnk+1) = 0 \rightarrow k = \frac{1}{e}$$

به این ترتیب اثبات شده که بهترین k برای بالاترین احتمال برابر $\frac{n}{e}$ است که احتمال $\frac{1}{e}$ را برمی‌گرداند.

بخش 2

با وارد کردن عدد 407 و قرار دادن $n = 1000, s = 10000$ میانگین نمونه ها را بدست آورده و در یک هیستوگرام فراوانی آن را به همراه KDE[1] رسم می‌کنیم.

[1] Kernel Density Estimator

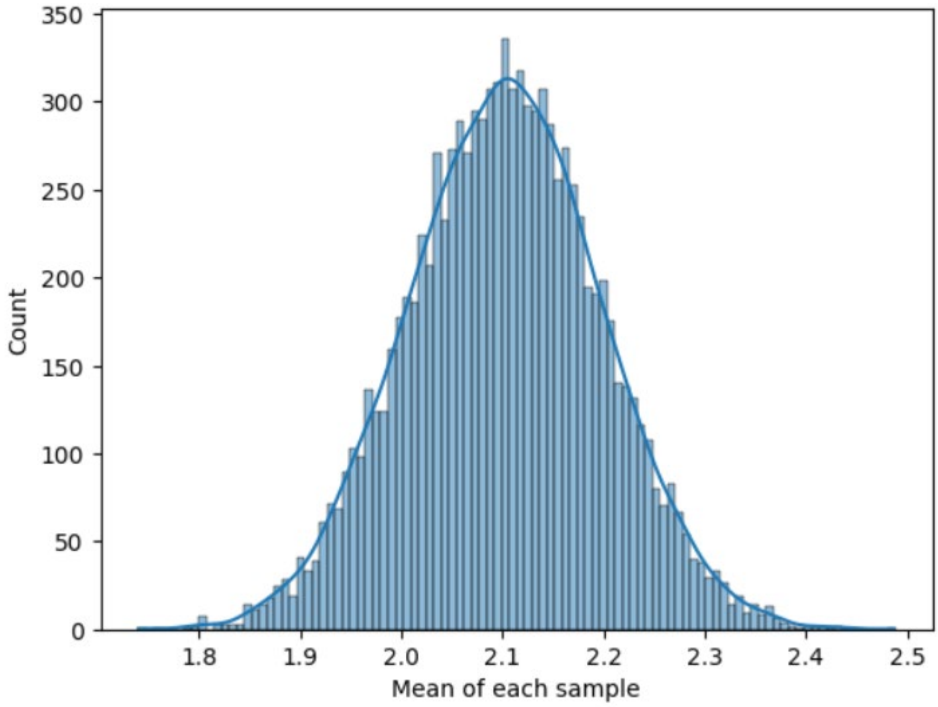


Figure 3 نمودار فراوانی میانگین نمونه های تابع **distribution**

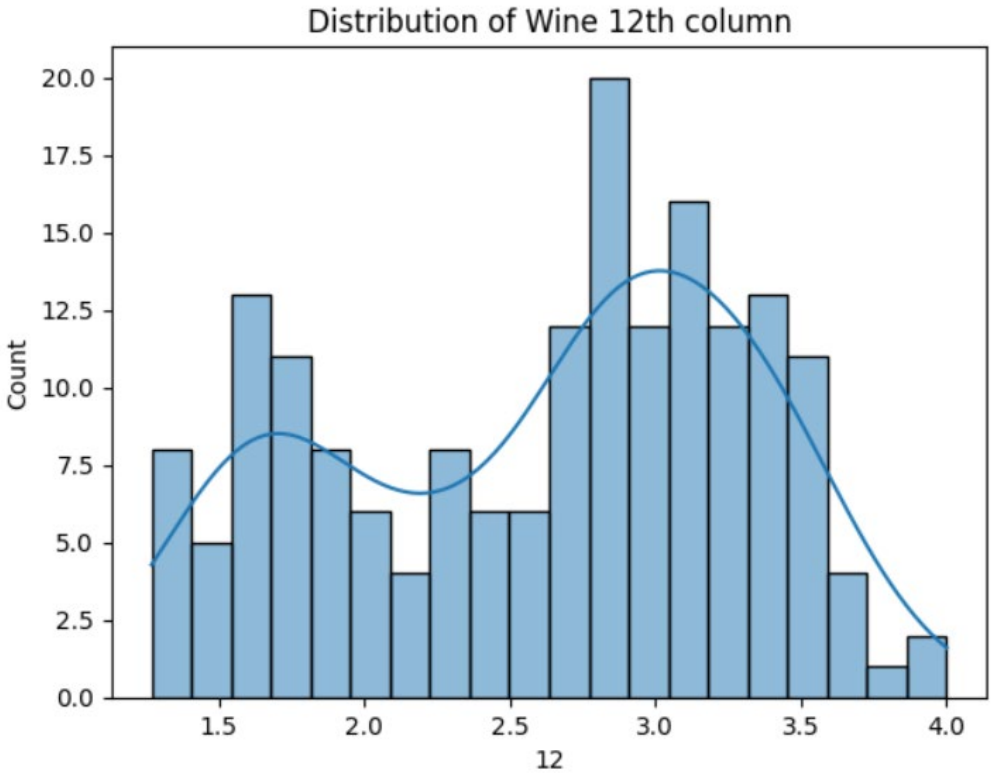


Figure 4 نمودار فراوانی ستون 12ام دیتاست **wine**

برای این دیتاست نیز مقادیر $n = 100, s = 100000$ انتخاب می‌کنیم. میانگین نمونه ها را بدست آورده و در یک هیستوگرام فراوانی آن را به همراه KDE[1] رسم می‌کنیم.

همانطور که از روی KDE مشاهده می‌کنید. نمودار فروانی میانگین نمونه ها بسیار نزدیک به نمودار نرمال است.

قضیه حد مرکزی (CLT) بیان می‌کند. هر گاه تعدادی متغیر تصادفی مستقل با توزیع دلخواهِ یکسان باشند. متغیر تصادفی حاصل از مجموع آن ها خود یک متغیر تصادفی با توزیع نرمال با همان میانگین ولی با انحراف معیار $\bar{\mu} = \frac{\mu}{\sqrt{n}}$ است. به این ترتیب میانگین هر دو دیتاست اصلی و نمونه برداری شده برابر 2.6117 و انحراف معیار دیتاست اصلی برابر 0.7100 و انحراف معیار دیتاست نمونه برداری شده 0.0708 است. که با قضیه همخوانی دارد.

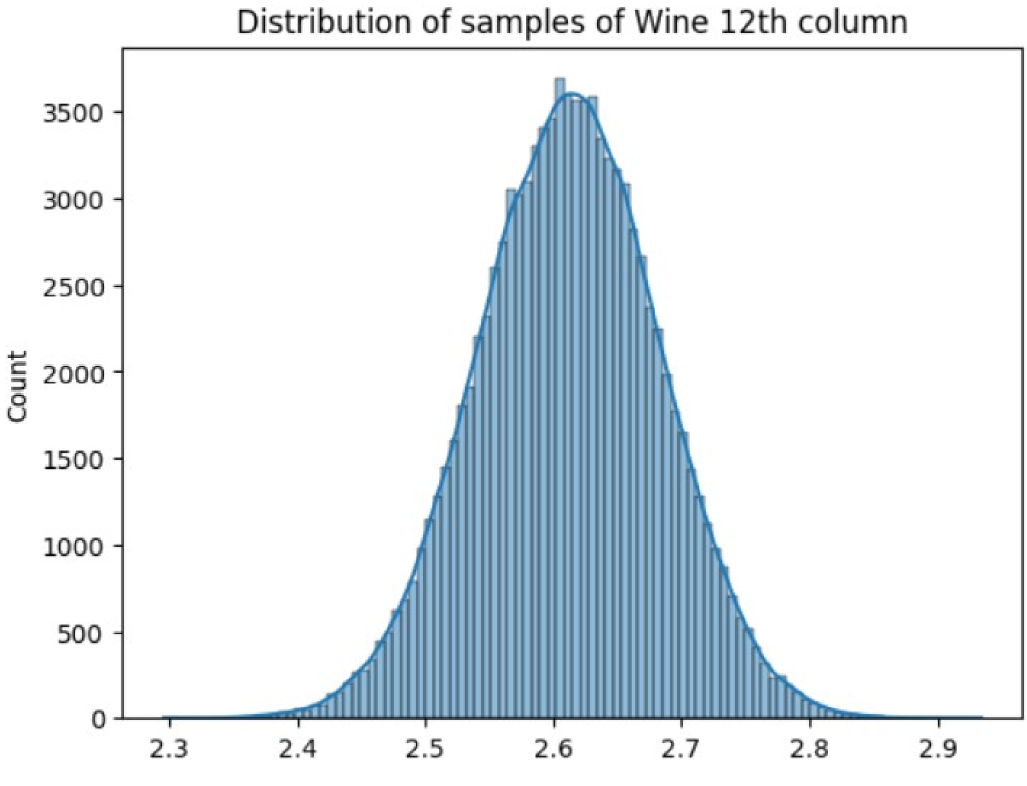


Figure 5 نمودار توزیع فراوانی میانگین نمونه‌های گرفته شده از ستون 12ام دیتاست **wine**

[1] Kernel Density Estimator

سوال 2

بخش 1

خیر به این دلیل که در این صورت داده ها استقلال دارند. ولی چیزی که مهم است استقلال شرطی است. یعنی ممکن است$P(A \cap B) = P(A)P(B)$ داشته باشیم و داده ها بر هم عمود باشند ولی $P(A, B|C) \neq P(A|C)P(B|C)$

بخش 2

تابع توزیع نرمال دارای پارامتر های میانگین μ و واریانس σ است.

$$X \sim N(\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

حالا طبق Maximum Likelihood Estimation داریم:

$$\theta_{ML} = argmax \sum_{i=1}^{n} \log P_X(x_i; \theta) = argmax\, \mathcal{L}(x_1, \dots, x_n; \theta)$$

$$\mathcal{L}(x_1, \dots, x_n; \mu, \sigma) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma) - \frac{1}{2}\sum_{i=1}^{n}\left(\frac{x_i - \mu}{\sigma}\right)^2$$

حالا برای پیدا کردن بهترین پارامتر یکبار نسبت به μ و یکبار نسبت به σ مشتق میگیریم:

$$\frac{\partial \mathcal{L}}{\partial \mu} = \frac{1}{\sigma}\left(\sum_{i=1}^{n} x_i - n\mu\right) = 0 \to \mu = \sum_{i=1}^{n} x_i$$

$$\frac{\partial \mathcal{L}}{\partial \sigma} = -\frac{n}{2\sigma} + \frac{1}{2\sigma^2}\left(\sum_{i=1}^{n}(x_i - \mu)^2\right) = 0 \to \sigma = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$

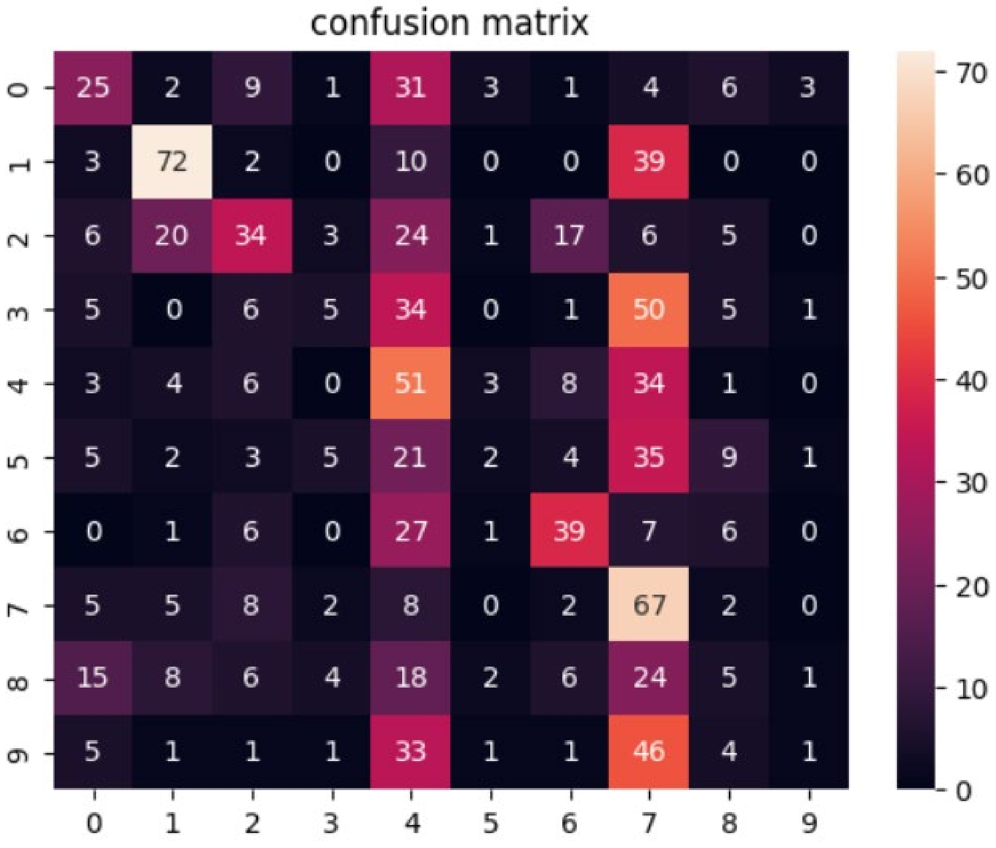
سوال 3

1)

ویژگی هایی که برای طبقه بند خود استفاده می‌کنیم. تعداد # ها و تعداد + ها در 4 قسمت گوشه بالا چپ، بالا راست، پایین چپ و پایین راست است. این داده ها کمی هستند و برای تبدیل آن ها به categorical آن ها را به 4 قسمت بر اساس چارک هایشان تقسیم می‌کنیم.

2)

برای قسمتی که ویژگی ها را بصورت دستی استخراج کردیم دقت مدل به 30.1٪ رسید که مقدار آن پایین است. علت آن سخت بودن استخراج ویژگی از میان عکس ها است. چیزی که در شبکه های عصبی فرق بین معماری MLP,CNN را دیدیم. اما اگر ویژگی ها را دستی استخراج نکنیم و کل اندازه تصویر را بصورت یک بردار flatten شده به مدل بدهیم مدل به دقت 72.5٪ می‌رسد.



	0	1	2	3	4	5	6	7	8	9
0	25	2	9	1	31	3	1	4	6	3
1	3	72	2	0	10	0	0	39	0	0
2	6	20	34	3	24	1	17	6	5	0
3	5	0	6	5	34	0	1	50	5	1
4	3	4	6	0	51	3	8	34	1	0
5	5	2	3	5	21	2	4	35	9	1
6	0	1	6	0	27	1	39	7	6	0
7	5	5	8	2	8	0	2	67	2	0
8	15	8	6	4	18	2	6	24	5	1
9	5	1	1	1	33	1	1	46	4	1

Figure 6 ماتریس آشفتگی مدل در حالت اسخراج دستی ویژگی ها

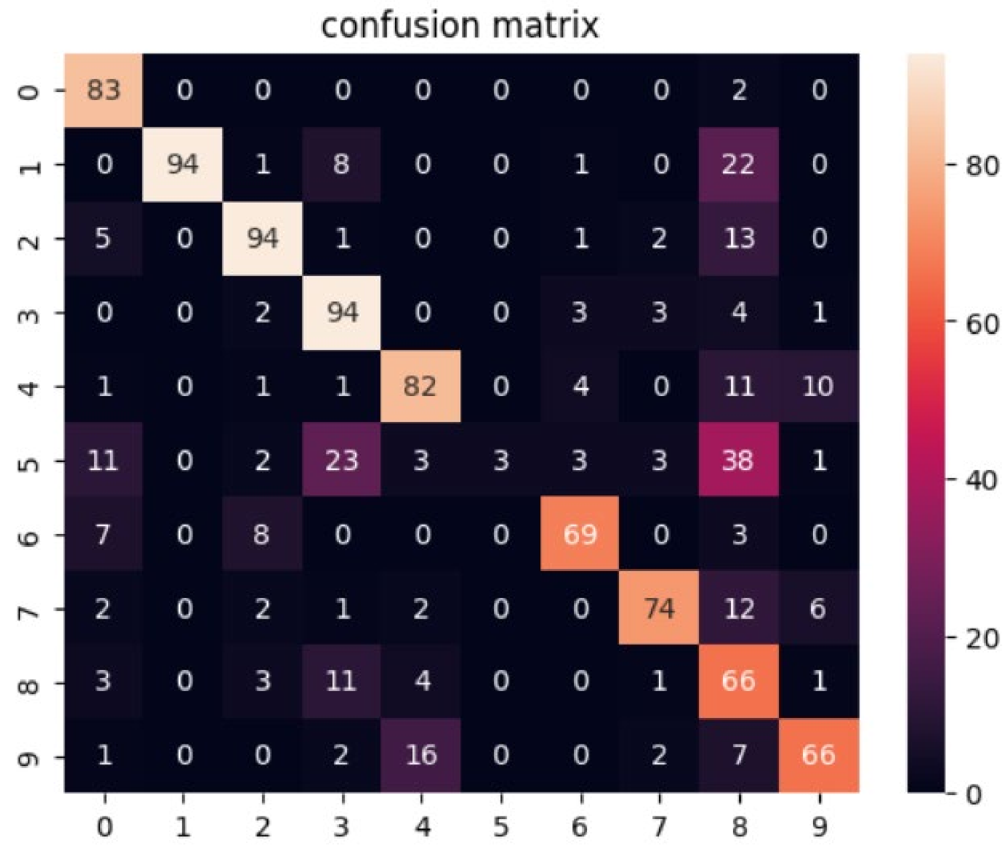


Figure 7 ماتریس آشفتگی مدل بدون استخراج دستی ویژگی

پیوست

سوال 1

بر طبق فرض سوال k,n را مشخص کرده. و هر با یک جمعیت رندوم بدون جایگذاری از 0 تا 99 ایجاد می‌کنیم. هر بار k max() داده اول را حساب کرده و در یک حلقه میبینیم که آیا بیشترین عدد که 99 است اول انتخاب می‌شود. هر 10000 بار اجرا تعداد موفقیت ها را برای هر k ثبت کرده و در نمودار رسم می‌کنیم. بار دیگر این کار را برای n متغیر انجام داده و نمودار آن را رسم می‌کنیم.

برای قسمت 2 صرفا نمونه ها را از تابع آماده گرفته و با حساب کردن میانگین آن ها نمودار هیتسوگرام آن را رسم می‌کنیم.

سوال 3

ابتدا کتابخانه ها را وارد می‌کنیم. سپس بر اساس ویژگی های گفته شده تعداد 1و2 های هر ناحیه از عکس را در لیست features میریزیم. با یک تابع آن را categorical کرده و سپس برای هر ویژگیِ هر کلاس بر اساس فرمول Laplace smoothing for Naïve Bayes احتمال $\hat{P}(w_i, c)$ کلاس ها و ویژگی های مختلف را حساب می‌کنیم. تابع give_count_c() عبارت count(c) را حساب می‌کند. که جمع همه ویژگی های در نظر گرفته شده برای یک کلاس است. سپس به همان منوال ویژگی های دیتاست آزمون نیز استخراج شده. و لگاریتم احتمال شرطی $\hat{P}(w_i, c)$ را برای هر کلاس در یک حلقه حساب می‌کنیم. و آرگومان ماکسیمم را بر می‌ردانیم. سپس از لیست predictions دقت مدل را حساب می‌کنیم.

برای قسمتی که ویژگی بصورت دستی حساب نمی‌شود. کد ها تقریبا مشابه است. تنها فرق این است که لیست features در این حالت حاوی همه پیکسل های تصویر بصورت flatten است و ویژگی دستی استخراج نشده است.