به نام خدا

تمرین کامپیوتری اول

درس: آمار و احتمال مهندسی

مجید صادقی نژاد

۸۱۰۱۰۱۴۵۹

توضیحات:

کد اصلی پروژه google colab نوشته شد و آدرس آن در زیر آمده است پیشنهاد می شود برای تجربه بهتر کد در همین محیط اجرا شود هر چند که فایل ها ipynb. , py. که ذخیره شده از همین پروژه در colab هستند هم در ضمیمه فایل های تحویل شده آمده اند همچنین کد هر سوال توسط جدا کننده ها جدا شده است.

لینک پروژه :
https://colab.research.google.com/drive/۱۷۲ugOGDN۱ZOJbFgfWxBylcArj۷۹O۴۰rb?usp=sharing

سوال اول:

کد سوال :

```python
import numpy as np

m = 5000
n = 500
probabilities = np.arange(0.0, 1.01, 0.01)

samplesPerProb = np.zeros((len(probabilities), m))
expectationPerProb = np.zeros((2,len(probabilities)))
variancePerProb = np.zeros((2,len(probabilities)))

def bernoliSampleGenrator(prob):
  bernoliGenerated = np.random.choice([0, 1], size=(m * n), p=[1 - prob, prob])
  bernoliOrganized = np.reshape(np.array(bernoliGenerated), (m, n))
  BinominalSamples = np.sum(bernoliOrganized, axis=1)
  return BinominalSamples

for i, prob in enumerate(probabilities):
    samplesPerProb[i, :] = bernoliSampleGenrator(prob)
    expectationPerProb[0,i] = n*prob
    variancePerProb[0,i] = n*prob*(1-prob)

expectationPerProb[1,:] = np.sum(samplesPerProb, axis=1)/m
sampExDiffPerProb = samplesPerProb-(np.reshape(expectationPerProb[1,:], (len(probabilities), 1)))
sampExDiff2PerProb = (sampExDiffPerProb * sampExDiffPerProb)
variancePerProb[1,:] = np.sum(sampExDiff2PerProb, axis=1)/m
```

در ابتدا تعداد نمونه ها تعداد آزمایش و آرایه ی احتمال های مختلف از ۰ تا ۱ تعریف شده ست سپس از آنجا که آرگومان احتمال (p) تابع random.choice امکان پذیرش یک آرایه را نداشت مجبور به تعریف حلقه for شدیم و در باقی کد از حلقه استفاده نشده است حال با این حلقه نمونه گیری به ازای هر احتمال (p) را انجام داده و به ازای هر احتمال (p) میانگین و واریانس را با فرمول محاسبه می کنیم و در آرایه های expectationPerProb و variancePerProb که هر کدوم دو ردیف داشته و ردیف اول برای حالت فرمولی ذخیره می کنیم سپس اعمالی که برای محاسبه میانگین و واریانس از روی تعریف آن ها لازم است را بر روی نمونه های گرفته شده پیاده کرده تا میانگین و واریانس به ازای هر احتمال به روش نمونه گیری نیز به دست بیاید و این میانگین و واریانس های به دست آمده را در ردیف دوم آرایه هایی که قبل تر تعریف کرده بودیم ذخیره می کنیم حال دو آرایه ی expectationPerProb و variancePerProb در ردیف اول خود دارای حالت فرمولی و در ردیف دوم دارای حالت نمونه گیری هستند.

۱- تابع طراحی شده تابع زیر است که ابتدا از توزیع برنولی نمونه گیری کرده و سپس توزیع های برنولی را جمع می کند و به تعداد خواسته شده نمونه تحویل می دهد در این جا تعداد نمونه های خواسته شده m تاست .

```python
def bernoliSampleGenrator(prob):
  bernoliGenerated = np.random.choice([0, 1], size=(m * n), p=[1 - prob, prob])
  bernoliOrganized = np.reshape(np.array(bernoliGenerated), (m, n))
  BinominalSamples = np.sum(bernoliOrganized, axis=1)
  return BinominalSamples
```

۲- واریانس و میانگین های محاسبه شده به ازای احتمال های مختلف در آرایه اول به صورت فرمولی و در آرایه ی دوم به صورت نمونه گیری و با استفاده از تعریف محاسبه شده اند البته که به علت تعداد بالای احتمال ها (۱۰۰ تا) خوانایی بالایی ندارند . (قطعه کد چاپ کننده ی این دو آرایه در یک بخش جدا از نوت بوک تعریف شده اند.)

دو ردیف آرایه ی میانگین ها:

```
[  0.   5.  10.  15.  20.  25.  30.  35.  40.  45.  50.  55.  60.  65.
  70.  75.  80.  85.  90.  95. 100. 105. 110. 115. 120. 125. 130. 135.
 140. 145. 150. 155. 160. 165. 170. 175. 180. 185. 190. 195. 200. 205.
 210. 215. 220. 225. 230. 235. 240. 245. 250. 255. 260. 265. 270. 275.
 280. 285. 290. 295. 300. 305. 310. 315. 320. 325. 330. 335. 340. 345.
 350. 355. 360. 365. 370. 375. 380. 385. 390. 395. 400. 405. 410. 415.
 420. 425. 430. 435. 440. 445. 450. 455. 460. 465. 470. 475. 480. 485.
 490. 495. 500.]
[  0.       4.9908   9.9974  15.0306  20.042   24.9556  30.0858  35.012
  39.9702  44.942   50.0974  54.864   60.0668  64.6832  70.1044  75.12
  79.8444  85.1586  90.1128  94.888   99.9784 105.0576 109.9486 114.9106
 119.8532 125.0348 130.2874 135.1244 140.2626 145.0304 149.8446 154.902
 160.104  165.0096 170.0392 175.02   180.2086 184.8508 189.7886 194.8226
 199.8434 204.7806 210.0622 215.0694 220.2586 225.2208 229.922  234.966
 239.8922 245.103  250.249  254.9028 260.1558 265.1598 270.1318 274.9054
 279.8812 285.0416 289.9618 295.2648 300.158  304.9086 310.0274 315.1552
 320.1474 324.9248 330.0796 334.7476 339.9714 344.88   349.9714 354.999
 359.9102 364.9892 370.3494 374.933  379.7934 384.9126 389.916  395.237
 400.0656 404.8796 410.2268 415.1342 419.978  424.9728 430.0086 434.9996
 440.0906 444.999  450.0212 454.9894 460.1828 465.1    470.1176 474.9212
 480.0186 485.0352 489.9524 495.0026 500.    ]
```

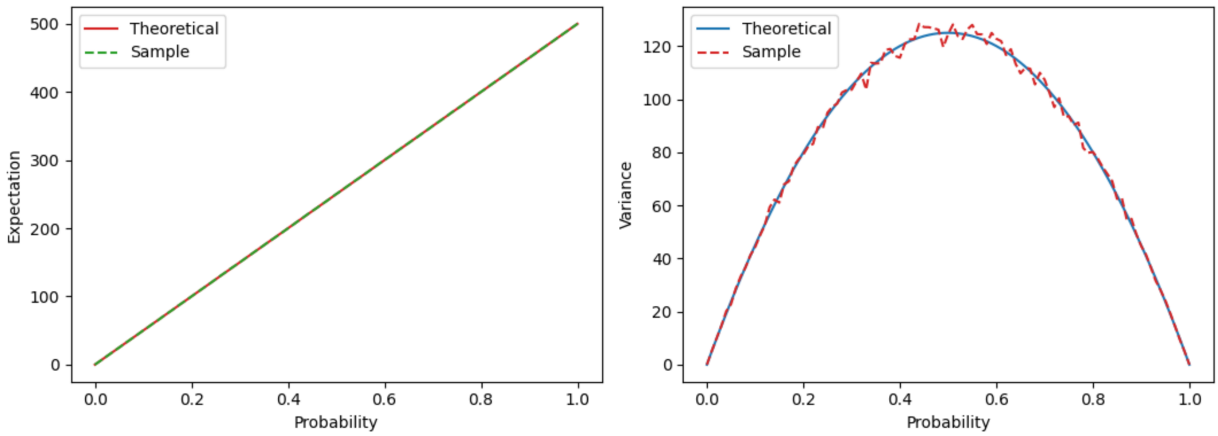
دو ردیف آرایه ی واریانس ها:

```
[  0.     4.95   9.8   14.55  19.2   23.75  28.2   32.55  36.8   40.95
  45.    48.95  52.8   56.55  60.2   63.75  67.2   70.55  73.8   76.95
  80.    82.95  85.8   88.55  91.2   93.75  96.2   98.55 100.8  102.95
 105.   106.95 108.8  110.55 112.2  113.75 115.2  116.55 117.8  118.95
 120.   120.95 121.8  122.55 123.2  123.75 124.2  124.55 124.8  124.95
 125.   124.95 124.8  124.55 124.2  123.75 123.2  122.55 121.8  120.95
 120.   118.95 117.8  116.55 115.2  113.75 112.2  110.55 108.8  106.95
 105.   102.95 100.8   98.55  96.2   93.75  91.2   88.55  85.8   82.95
  80.    76.95  73.8   70.55  67.2   63.75  60.2   56.55  52.8   48.95
  45.    40.95  36.8   32.55  28.2   23.75  19.2   14.55   9.8    4.95
   0.  ]
[  0.           5.06071536   9.85819324  14.60646364  20.099436
  22.74802864  28.71803836  33.222656    36.59491196  41.178636
  44.21311324  48.619504    52.43593776  59.27043776  62.16790064
  61.0344      67.93338864  69.04104604  74.81247616  77.309456
  79.20273344  82.21308224  83.11995804  89.49900764  88.89484976
  94.91958896  97.15080124  98.30972464 102.52924124 103.46907584
 103.54645084 106.747596   109.604784   103.60710784 113.70566336
 113.4308     113.58788604 118.53013936 118.93391004 116.30032924
 115.60647644 120.70206364 122.48793116 122.65578364 128.38532604
 127.15604736 127.100316   126.657244   126.16057916 119.635991
 124.520999   128.11935216 123.77192636 122.67786396 126.29842876
 127.98325084 124.50148656 124.37346944 120.77314076 124.95428096
 122.758236   121.83264604 116.63584924 118.81991296 113.23607324
 109.74914496 111.80726384 111.46509424 105.57258204 109.9364
 107.34938204 102.376599    96.99773596 100.35228336  93.30851964
  93.590111    90.28511644  91.16576124  81.597344    79.786431
  80.11209664  77.68230384  74.50376176  71.84499036  69.123516
  62.44606016  62.78452604  54.66079984  54.90519164  49.150599
  44.71275056  41.48608764  36.66538416  31.61         28.13017024
  24.23139056  19.52785404  14.59316096   9.48693424   4.94779324
   0.        ]
```

۳- کد این بخش :

```python
import matplotlib.pyplot as plt
fig, (p1, p2) = plt.subplots(1, 2, figsize=(10.8, 4))

p1.set_xlabel('Probability')
p1.set_ylabel('Expectation')
p1.plot(probabilities, expectationPerProb[0], color='tab:red', label='Theoretical')
p1.plot(probabilities, expectationPerProb[1], linestyle='dashed', color='tab:green', label='Sample')
p1.legend(loc='upper left')

p2.set_xlabel('Probability')
p2.set_ylabel('Variance')
p2.plot(probabilities, variancePerProb[0], color='blue', label='Theoretical')
p2.plot(probabilities, variancePerProb[1], linestyle='dashed', color='red', label='Sample')
p2.legend(loc='upper left')

plt.tight_layout()
plt.show()
```

در این بخش نمودار مقایسه ی دو حالت فرمولی و نمونه گیری برای واریانس و میانگین بر حساب احتمال های مختلف (p) رسم شده اند لازم به ذکر است از آن جا که حالت فرمولی و نمونه گیری میانگین ها به شدت به یکدیگر شبیه هستند تشخیص این دو در شکل سخت است.



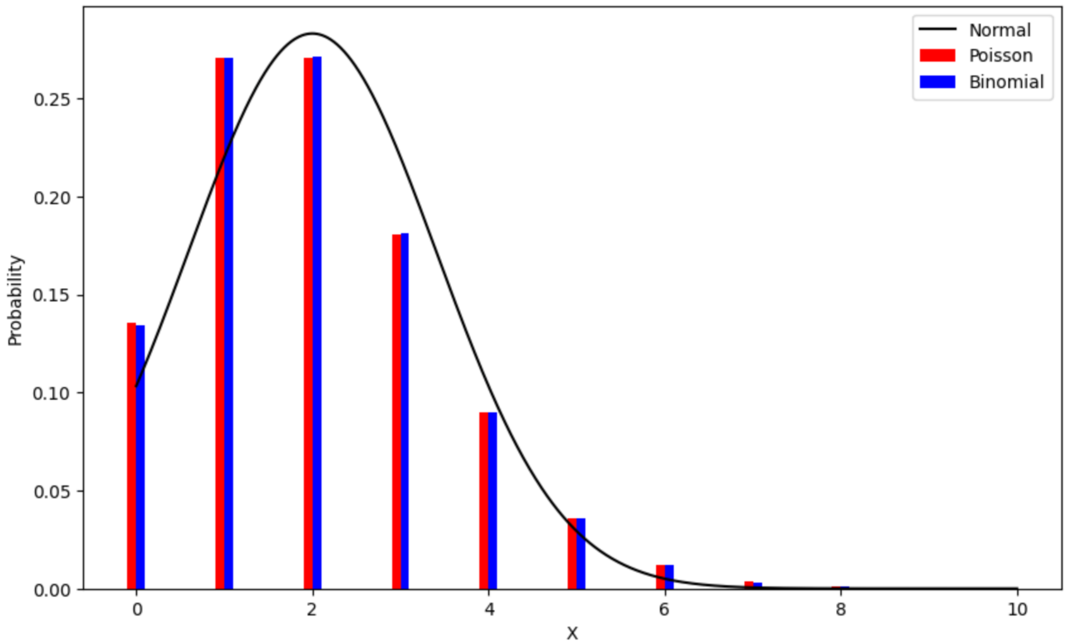
۴- همانطور که در دو نمودار بالا مشاهده می شود مقادیر تئوری و عملی میانگین ها به شدت به یکدیگر نزدیک هستند و برای واریانس نیز شباهت بسیار خوبی بین روش عملی و تئوری وجود دارد با اینکه واریانس عملی در برخی از احتمال ها مقداری تفاوت با واریانس تئوری دارد اما شکل کلی واریانس عملی بر حسب احتمال ها بر شکل کلی واریانس تئوری بر حسب احتمال ها منطبق است.

سوال دوم :
کد سوال:

```python
import numpy as np
from scipy.stats import norm
from scipy.stats import poisson
from scipy.stats import binom
import matplotlib.pyplot as plt

n = 250
p = 0.008

jointX = np.linspace(0, 10, 1000)
discreteX = np.arange(0, 10)

pdfNormal = norm.pdf(jointX, loc=n * p, scale=np.sqrt(n * p * (1 - p)))
pmfPoisson = poisson.pmf(discreteX, mu=n * p)
pmfBinomial = binom.pmf(discreteX, n=n, p=p)

plt.figure(figsize=(10, 6))

plt.plot(jointX, pdfNormal, color='black', label='Normal')
plt.bar(discreteX - 0.05, pmfPoisson, width=0.1, color='red', label='Poisson')
plt.bar(discreteX + 0.05, pmfBinomial, width=0.1, color='blue', label='Binomial')

plt.xlabel('X')
plt.ylabel('Probability')
plt.legend()

plt.show()
```

در ابتدا با استفاده از توابع آماده کتابخانه ای پی دی اف توزیع نرمال و پی ام اف توزیع های پواسون و دوجمله را تعریف می کنیم که مقادیر x این توابع در بالا هم برای حالت گسسته و هم حالت پیوسته تعریف شده است سپس با استفاده از کتابخانه ی matplotlib نمودار این سه توزیع بر حسب x را در یک پنجره نمایش می دهیم.

۱- نمودار سه توزیع به شکل زیر است :



۲- همانطور که در شکل بالا مشاهده می شود توزیع پواسون به شدت نزدیک به توزیع دو جمله ای است که این به این سبب است که n*p توزیع دو جمله یک عدد صحیح بین صفر و ده است پس تقریب با توزیع پواسون تقریب مناسبی است اما در رابطه با توزیع نرمال از آنجا که p توزیع دو جمله ای ۰.۰۰۸ است و تا ۰.۵ فاصله ی زیادی دارد بنابراین تقریب مناسبی نیست اما از آنجا که میانگین و واریانس این دو توزیع یکسان است بنابراین شکل کلی توزیع نرمال از شکل کلی توزیع دوجمله ای پیروی می کند.

سوال سوم – بخش اول:

کد سوال:

```python
from scipy.stats import norm
import numpy as np
from scipy.integrate import quad

def normalPdf(x):
  return norm.pdf(x, loc=80, scale=12)

def normalCdf(x):
  result, _ = quad(normalPdf, 0, x)
  return result

def cdfInverse(prob):
  lowerGuess = 0
  upperGuess = 120
  Guess = (lowerGuess + upperGuess) / 2
  while True:
    Guess = (lowerGuess + upperGuess) / 2
    result = normalCdf(Guess)

    if np.abs(result - prob) < 1e-5:
      break
    if result >= prob:
      upperGuess = Guess
    else:
      lowerGuess = Guess

  return Guess


print(f'\n 1. {cdfInverse(0.9)} \n')
print(f'\n 2. {cdfInverse(0.5)} to {cdfInverse(0.75)} \n')
print(f'\n 3. {normalCdf(90)-normalCdf(80)} \n')
```

در ابتدا با استفاده از توابع آماده کتابخانه ای پی دی اف توزیع نرمال تعریف می شود سپس با استفاده از تابع کتابخانه ای دیگری با استفاده از انتگرال گیری عددی تابع سی دی اف توزیع نرمال نیز تعریف می شود در نهایت تایع معکوس تابع سی دی اف که برای پیدا کردن صدک استفاده می شود تعریف می شود در این تابع با استفاده از الگوریتم دو بخشی (تنصیف) مقدار x (متغیر تصادفی) پیدا می شود. همچنین خروجی سه سوال بدین صورت است:

```
1. 95.37872314453125

2. 80.00015258789062 to 88.0938720703125

3. 0.29767161903635697
```

۱- اگر فردی می خواهد جزو ۱۰ درصد برتر کلاس باشد باید جزو صدک ۹۰ ام باشد یعنی باید معکوس سی دی اف به ازای ۰.۹ رو بیابیم با استفاده از توابعی که نوشته ایم مقدار را پیدا می کنیم:
مقدار یافت شده برابر : **۹۵.۳۷۸۷۲۳۱۴۴۵۳۱۲۵**
یعنی باید برای اینکه جزو ۱۰ درصد برتر باشیم باید نمره ی بالا ۹۵.۳۸ باشد.

۲- چارک دوم و سوم برابر صدک ۵۰ ام و صدک ۷۵ ام هستند یعنی باید معکوس سی دی اف به ازای ۰.۵ و ۰.۷۵ یافت شود.
که بازه یافت شده برابر: **۸۰.۰۰۰۱۵۲۵۸۷۸۹۰۶۲ تا ۸۸.۰۹۳۸۷۲۰۷۰۳۱۲۵**
یعنی نمرات ۸۰ تا ۸۸.۱ در این چارک ها قرار دارند.

۳- این سوال به معنی یافتن Cdf(۹۰)-Cdf(۸۰) است که تابع آن را از پیش پیدا کرده ایم مقدار یافت شده برابر است با : **۰.۲۹۷۶۷۱۶۱۹۰۳۶۳۵۶۹۷**

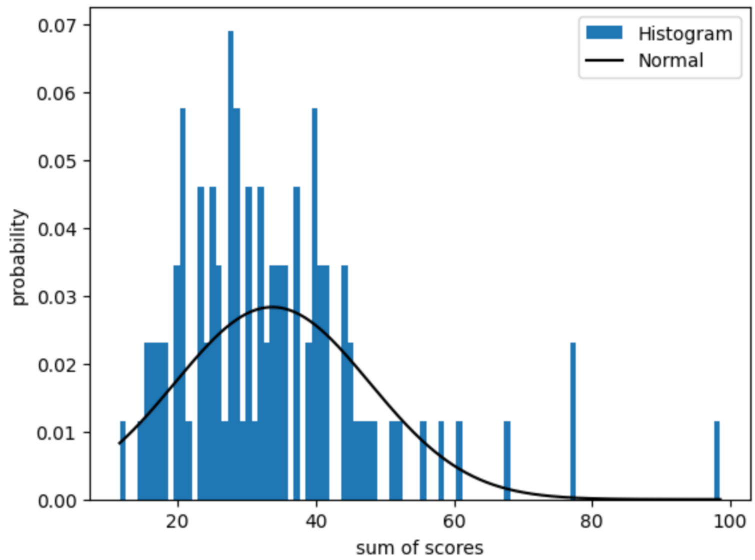
سوال سوم – بخش امتیازی :

کد سوال:

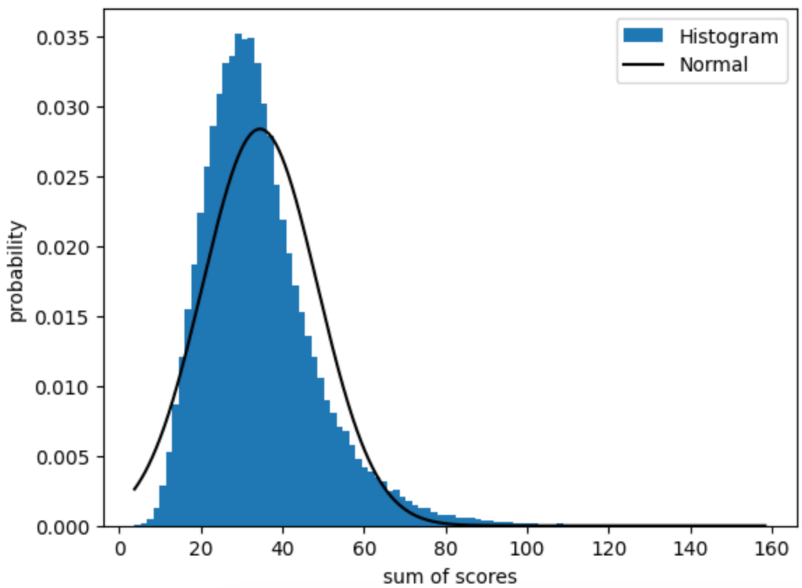
```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt

sampleCount = 1000

uniform = 20 * np.random.uniform(size=sampleCount)
expon = np.random.exponential(scale=1/0.08 , size=sampleCount)
poisson = np.random.poisson(lam= 12 , size=sampleCount)

total = expon + poisson +  uniform

x = np.linspace(min(total), max(total), 10000)
normalPdf = norm.pdf(x, loc=np.mean(total), scale=np.std(total))

plt.hist(total, bins=100, density=True, label='Histogram')
plt.plot(x, normalPdf, color='black', label='Normal')

plt.xlabel('sum of scores')
plt.ylabel('probability')
plt.legend()
plt.show()
```

در ابتدا از هر کدام از سه توزیع که برای نمرات درس ها هستند به تعداد مساوی نمونه گیری می کنیم لازم به ذکر است که برای نمونه گیری از توزیع پواسون ضریب پواسون به این علت ۱۲ انتخاب شده است که در یک بازه زمانی مشخص به طور متوسط دانش آموزان در این درس ۱۲ گرفته اند همچنین ضریب توزیع نمایی برابر ۰.۰۸ انتخاب شده است بدین منظور که در این صورت طبق pdf این توزیع همچنان احتمال برای گرفتن نمره ۲۰ به اندازه کافی وجود دارد وجود دارد و اگر این ضریب بیشتر از این مقدار (۰.۰۸) شود سبب می شود احتمال آوردن نمره های ۲۰ و حوالی آن به شدت کاهش یابد و از آنجا که می خواهیم برای گرفتن نمره های ۲۰ و حوالی آن احتمال وجود داشته باشد ضریب توزیع نمایی را ۰.۰۸ انتخاب می کنیم. سپس مقادیر نمونه های تولید شده را جمع می کنیم و هیستوگرام جمع آن ها را رسم می کنیم. سپس با گرفتن میانگین و واریانس از جمع آن ها یک نمودار پی دی اف توزیع نرمال با این واریانس و میانگین رسم می کنیم دیده می شود که با افزایش تعداد نمونه ها شباهت شکل کلی هیستوگرام و پی دی اف توزیع نرمال افزایش می یابد

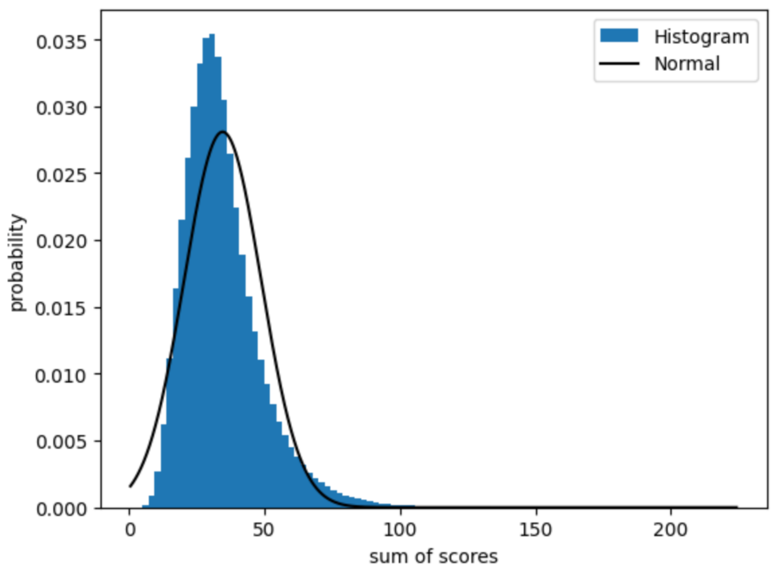
به ازای ۱۰۰ نمونه:



به ازای ۱۰۰.۰۰۰ نمونه:



به ازای ۱۰.۰۰۰.۰۰۰ نمونه:



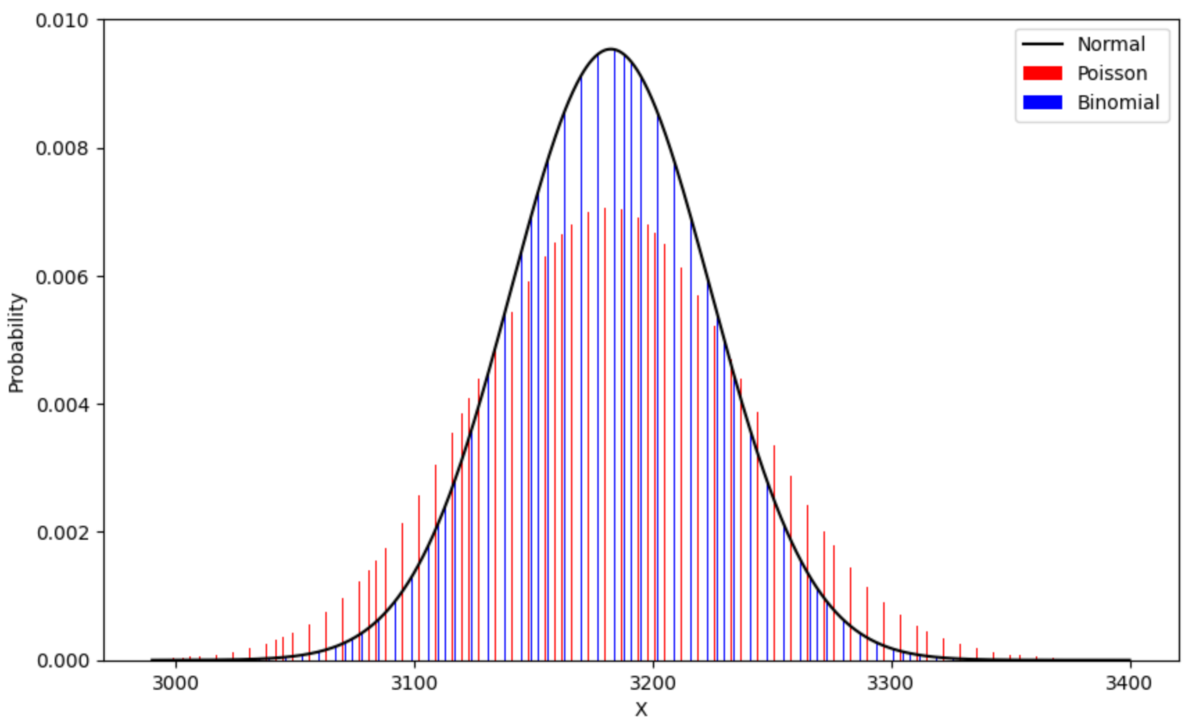
سوال سوم – بخش دوم:

کد سوال:

```
import numpy as np
from scipy.stats import norm
from scipy.stats import poisson
from scipy.stats import binom
import matplotlib.pyplot as plt

n = 7072
p = 0.45

jointX = np.linspace(2990, 3400, 1000)
discreteX = np.arange(2990, 3400)

pdfNormal = norm.pdf(jointX, loc=n * p, scale=np.sqrt(n * p * (1 - p)))
pmfPoisson = poisson.pmf(discreteX, mu=n * p)
pmfBinomial = binom.pmf(discreteX, n=n, p=p)

plt.figure(figsize=(10, 6))

plt.plot(jointX, pdfNormal, color='black', label='Normal')
plt.bar(discreteX - 0.05, pmfPoisson, width=0.1, color='red', label='Poisson')
plt.bar(discreteX + 0.05, pmfBinomial, width=0.1, color='blue', label='Binomial')

plt.xlabel('X')
plt.ylabel('Probability')
plt.legend()

plt.show()
```

این سوال نیز دقیقا همانند سوال دو مقایسه سه توزیع دو جمله ای، پواسون و نرمال است همراه با کشیدن نمودار های آن ها بنابراین کد این اسوال دقیقا مشابه کد سوال دوم است با تفاوت n و p توزیع دو جمله ای

۱- نمودار این سه توزیع بر حسب متغیر تصادفی x به شکل زیر است :



۲- همانطور که در شکل معلوم است شکل کلی توزیع دو جمله ای با شکلی توزیع نرمال یکسان است و توزیع پواسون فاصله ی زیادی با توزیع نرمال و دو جمله ای دارد که این مطلب قابل پیش بینی بود زیر

از آن جا که p برابر ۰.۴۵ است و بسیار به ۰.۵ نزدیک است پس توزیع نرمال می تواند تقریب بسیار خوبی برای این توزیع دو جمله ای باشد اما از آن جا که n*p این توزیع دو جمله ای که پارامتر توزیع پواسون است عدد ۳۱۸۳ است و خارج از محدوده ی اعداد ۰ تا ۱۰ است پس توزیع پواسون نمی تواند تقریب مناسبی باشد.