Homework2

April 11, 2025

0.0.1 Homework #2; DATA 605

John Ferrara

0.0.2 Problem 1

(Bayesian):

A new credit scoring system has been developed to predict the likelihood of loan defaults. The system has a 90% sensitivity, meaning that it correctly identifies 90% of those who will default on their loans. It also has a 95% specificity, meaning that it correctly identifies 95% of those who will not default. The default rate among borrowers is 2%. Given these prevalence, sensitivity, and specificity estimates, what is the probability that a borrower flagged by the system as likely to default will actually default?

If the average loss per defaulted loan is 200,000 dollars and the cost to run the credit scoring test on each borrower is 500 dollars, what is the total first-year cost for evaluating 10,000 borrowers?

Answer

[ ]:
```
# Model is 90% correct. Sensitivity = TP / (TP+FN)

#  TP = Flagged & Default
#  FN = Not Flagged & Default


# 95% specificity for those who do not default.
# specificity = TN / (TN + FP)

# TN = Not Flagged & No Default
# FP = Flagged & No default

# Default rate 2% p = 0.02, q = 0.98

# If 1000 borrowers
# 20 default; 980 dont.

# 90% of 20  = 18 Flagged for Default & default. 2 Not flagged for Default &
 ↪default
# 95% of 980 = 931 no flag, no default. 49 flag w/ no default
```

```
#TP = 18
#FP = 49
#FN = 2
#TN = 931

## Total Flagged 18+49 = 67

# PART 1) want to know chances borrower flagged to default WILL default.
# 18/67 = 26.8% of those flagged will actually default.

## ANSWER: 26.8% of those flagged will actually default.


# PART 2) what is the total first-year cost for evaluating 10,000 borrowers

#200,000 avg loss per default
# If 2% default w 10,000 borrowers = 200 defaults * avg. Cost of 200,000 =␣
 ↪40,000,000
# Cost per test per borrower = 500 * 10,000 = 5,000,000

## Losses from default 40,000,000 dollars from 200 (2%) defaults.
## Plus cost of running tests 5,000,000
## Total Cost is $45,000,000

## ANSWER: Assuming the losses from defaulting take place in the first year,␣
 ↪the total cost will be $45,000,000.
## If the losses from defaults happen in the second, or later year, the first␣
 ↪year would be the cost of running tests, so Five million.
```

(Binomial): The probability that a stock will pay a dividend in any given quarter is 0.7. What is the probability that the stock pays dividends exactly 6 times in 8 quarters? What is the probability that it pays dividends 6 or more times? What is the probability that it pays dividends fewer than 6 times? What is the expected number of dividend payments over 8 quarters? What is the standard deviation?

ANSWER

[15]:
```
from scipy.stats import binom
import math

## dividend payment is p=0.7
p = 0.7

## PART 1) Exectly 6 times in 8 quarters
n = 8 #nuimber of quarters (2 years)
print(f"ANSWER: {round(binom.pmf(6, n, p),4)} percent that exactly 6 quarters␣
 ↪out of 8 a stock will pay dividend.")
```

```
## PART 2) What is the probability that it pays dividends 6 or more times?
print(f"ANSWER: {round(binom.sf(5, n, p),4)} percent that 6 quarters or more␣
 ↪out of 8 a stock will pay dividend.")

## PART 3) What is the probability that it pays dividends fewer than 6 times?
print(f"ANSWER: {round(binom.cdf(5, n, p),4)} percent that dividends are paid␣
 ↪less than 6 times.")

## PART 4) What is the expected number of dividend payments over 8 quarters?
print(f"ANSWER: {round((n*p),3)} if the expected number of dividend payments.")

## PART 5) What is the standard deviation?
print(f"ANSWER: {round(math.sqrt(n * p * (1 - p)),3)} is the standard deviation.
 ↪")
```

```
ANSWER: 0.2965 percent that exactly 6 quarters out of 8 a stock will pay
dividend.
ANSWER: 0.5518 percent that 6 quarters or more out of 8 a stock will pay
dividend.
ANSWER: 0.4482 percent that dividends are paid less than 6 times.
ANSWER: 5.6 if the expected number of dividend payments.
ANSWER: 1.296 is the standard deviation.
```

(Poisson): A financial analyst notices that there are an average of 12 trading days each month when a certain stock's price increases by more than 2%. What is the probability that exactly 4 such days occur in a given month? What is the probability that more than 12 such days occur in a given month? How many such days would you expect in a 6-month period? What is the standard deviation of the number of such days? If an investment strategy requires at least 70 days of such price increases in a year for profitability, what is the percent utilization and what are your recommendations?

ANSWER

[23]:
```
from scipy.stats import poisson

## Avg 12 trading days each month stock inc >2%
lambda_ = 12

# Part 1) What is the probability that exactly 4 such days occur in a given␣
 ↪month?
print(f"ANSWER: {round(poisson.pmf(4, lambda_),4)*100} percent chance that␣
 ↪exactly 4 such days.")

# Part 2) What is the probability that more than 12 such days occur in a given␣
 ↪month?
print(f"ANSWER: {round(poisson.sf(lambda_,lambda_),4)*100} percent change that␣
 ↪more than 12 days occur in a month.")
```

```
# Part 3) How many such days would you expect in a 6-month period?

# Monthly Lambda moved toward 6 month scale
lambda_6 = lambda_*6
print(f"ANSWER: {round(lambda_6,4)} is the expected value for a 6 month period.␣
 ↪")

# Part 4) What is the standard deviation of the number of such days?
std_dev_month = math.sqrt(lambda_)
std_dev_6 = math.sqrt(lambda_6)
print(f"ANSWER: {round(std_dev_6,3)} is the standard devation for 6 month scale␣
 ↪and a {round(std_dev_month,3)} on a monthly scale.")

# Past 5) If an investment strategy requires at least 70 days of such price␣
 ↪increases in a year for profitability,
# what is the percent utilization and what are your recommendations?

# If the monthly avg is 12 then yearly avg would be 144 days.
# THe ave would 144 / 70 days for profit.
print(f"ANSWER: {round(144/70,4)*100} percent utiltization.")
print("This is 200 plus percent, so the recommendation is to continue with the␣
 ↪current strategy. There is ample room to test new strategies without worring␣
 ↪about profitability.")
```

```
ANSWER: 0.53 percent chance that exactly 4 such days.
ANSWER: 42.4 percent change that more than 12 days occur in a month.
ANSWER: 72 is the expected value for a 6 month period.
ANSWER: 8.485 is the standard devation for 6 month scale and a 3.464 on a
monthly scale.
ANSWER: 205.71 percent utiltization.
This is 200 plus percent, so the recommendation is to continue with the current
strategy. There is ample room to test new strategies without worring about
profitability.
```

(Hypergeometric): A hedge fund has a portfolio of 25 stocks, with 15 categorized as high-risk and 10 as low-risk. The fund manager randomly selects 7 stocks to closely monitor. If the manager selected 5 high-risk stocks and 2 low-risk stocks, what is the probability of selecting exactly 5 high-risk stocks if the selection was random? How many high-risk and low-risk stocks would you expect to be selected?

ANSWER

[32]:
```
from scipy.stats import hypergeom
stocks = 25
high_risk = 15 # 15/25 are high risk (60%)
low_risk = 10 # 10/25 are low risk (40%)

# Random selection: 5 high risk, 2 low risk (7 total)
```

```
# PART 1: what is the probability of selecting exactly 5 high-risk stocks if␣
 ↪the selection was random?
print(f"{round(hypergeom.pmf(5,stocks,high_risk,7)*100,2)} percent is the␣
 ↪probability of selecting exactly 5 high risk stocks.")

# PART 2: How many high-risk and low-risk stocks would you expect to be␣
 ↪selected?
print(f"{round(7*(0.6),0)} high risk stocks are expected to be selected.")
print(f"{round(7*(0.4),0)} low risk stocks are expected to be selected.")
```

```
28.11 percent is the probability of selecting exactly 5 high risk stocks.
4.0 high risk stocks are expected to be selected.
3.0 low risk stocks are expected to be selected.
```

(Geometric): The probability that a bond defaults in any given year is 0.5%. A portfolio manager holds this bond for 10 years. What is the probability that the bond will default during this period? What is the probability that it will default in the next 15 years? What is the expected number of years before the bond defaults? If the bond has already survived 10 years, what is the probability that it will default in the next 2 years?

ANSWER:

[48]:
```
#bond default p = 0.005 per year
p = 0.005
# holds for 10 years

# PART 1) What is the probability that the bond will default during this period?
 ↪
#Geometric form
not_default = (1-p)**10
default = (1-not_default)
print(f"{round(default*100,4)} percent probability taht the bond will default␣
 ↪in ten years.")

# PART 2) What is the probability that it will default in the next 15 years?
not_default = (1-p)**15
default = (1-not_default)
print(f"{round(default*100,4)} percent probability taht the bond will default␣
 ↪in 15 years.")

# PART 3) What is the expected number of years before the bond defaults?
print(f"The expected value for the years it takes for the bond to default is {1/
 ↪p} years.")

# PART 4) If the bond has already survived 10 years, what is the probability␣
 ↪that it will default in the next 2 years?
## No memory, so doesnt matter need to do 2 years
```

```
not_default = (1-p)**2
default = (1-not_default)
print(f"{round(default*100,4)} percent probability that the bond will default␣
 ↪in 2 years.")
```

```
4.889 percent probability taht the bond will default in ten years.
7.2431 percent probability taht the bond will default in 15 years.
The expected value for the years it takes for the bond to default is 200.0
years.
0.9975 percent probability taht the bond will default in 15 years.
```

(Poisson): A high-frequency trading algorithm experiences a system failure about once every 1500 trading hours. What is the probability that the algorithm will experience more than two failures in 1500 hours? What is the expected number of failures?

ANSWER:

[55]:
```
# 1 failure every 1500 trading hours

# PART 1) What is the probability that the algorithm will experience more than␣
 ↪two failures in 1500 hours?
print(f"{round(poisson.sf(2, 1)*100,2)} percent chance that 2 failures happen␣
 ↪in the 1500 hour window.")

# What is the expected number of failures?
print("The expected number of failures, as outlined in the probem is 1.")
```

```
8.03 percent chance that 2 failures happen in the 1500 hour window.
```

(Uniform Distribution): An investor is trying to time the market and is monitoring a stock that they believe has an equal chance of reaching a target price between 20 and 60 days. What is the probability that the stock will reach the target price in more than 40 days? If it hasn't reached the target price by day 40, what is the probability that it will reach it in the next 10 days? What is the expected time for the stock to reach the target price?

[69]:
```
# equal change of reaching target price between 20 and 60 days.
from scipy.stats import uniform

#Part 1) What is the probability that the stock will reach the target price in␣
 ↪more than 40 days?
interval_dist_uniform = uniform(20,(60-20))
print(f"{round(interval_dist_uniform.sf(40)*100,2)} percent chance that it␣
 ↪takes more than 40 days to reach price.")

#Part 2) If it hasn't reached the target price by day 40, what is the␣
 ↪probability that it will reach it in the next 10 days?
over_40 = interval_dist_uniform.sf(40)
over_40_to_50 = interval_dist_uniform.cdf(50) - interval_dist_uniform.cdf(40)
print(f"{over_40_to_50/over_40} probabilty that if the price isnt reached by␣
 ↪day 40, that it is reached in the next 10 days.")
```

```
#Part 3) What is the expected time for the stock to reach the target price?
print(f"{interval_dist_uniform.mean()} is the expected time for reaching the␣
 ↪price")
```

```
50.0 percent chance that it takes more than 40 days to reach price.
0.5 probabilty that if the price isnt reached by day 40, that it is reached in
the next 10 days.
40.0 is the expected time for reaching the price
```

(Exponential Distribution): A financial model estimates that the lifetime of a successful start-up before it either goes public or fails follows an exponential distribution with an expected value of 8 years. What is the expected time until the start-up either goes public or fails? What is the standard deviation? What is the probability that the start-up will go public or fail after 6 years? Given that the start-up has survived for 6 years, what is the probability that it will go public or fail in the next 2 years?

[81]:
```
#8 years until public

from scipy.stats import expon

exp_dist = expon(scale=8)

# PART 1) What is the expected time until the start-up either goes public or␣
 ↪fails?
print(f"The expected time is {exp_dist.mean()} years.")

# PART 2) What is the standard deviation?
print(f"The standard deviation is {exp_dist.std()}")

# PART 3) What is the probability that the start-up will go public or fail␣
 ↪after 6 years?
#More than 6 years so sf
print(f" {round(exp_dist.sf(6)*100,2)} perent chance that it will go public /␣
 ↪fail after 6 years.")

# PART 4) Given that the start-up has survived for 6 years, what is the␣
 ↪probability that it will go public or fail in the next 2 years?
# survived 6, max 8. no Memory
print(f"{round(exp_dist.cdf(2)*100,2)} percent chance that it goes public in␣
 ↪the 7th or 8th year.")
```

```
The expected time is 8.0 years.
The standard deviation is 8.0
 47.24 perent chance that it will go public / fail after 6 years.
22.12 percent chance that it goes public in the 7th or 8th year.
```

0.0.3 Problem 2

(Product Selection): A company produces 5 different types of green pens and 7 different types of red pens. The marketing team needs to create a new promotional package that includes 5 pens. How many different ways can the package be created if it contains fewer than 2 green pens?

[87]:
```
#5 types of green
#7 types of red
# New package of 5 pens, how many ways for <2 green. Order doesnt matter, so␣
 ↪combinations
# possibilities:
    # 1) 4 red, 1 green
    # 2) 5 red, 0 green

import math

case1 = math.comb(5,0) * math.comb(7,5) # Green Selection + Red Selection
#print(case1)
case2 = math.comb(5,1) * math.comb(7,4) # Green Selection + Red Selection
#print(case2)
possible_comb = case1+case2
print(possible_comb)

#Answer: There are 196 different types of ways to make these promotional␣
 ↪packages.
```

```
196
```

(Team Formation for a Project): A project committee is being formed within a company that includes 14 senior managers and 13 junior managers. How many ways can a project team of 5 members be formed if at least 4 of the members must be junior managers?

[90]:
```
#14 senior managers
#13 Junior Managers

# Total of 5 team members, if 4 must be jnr mgmt.

#combinations b/c order doesnt matter

snr_mgt_inc_case = math.comb(14,1)*math.comb(13,4)
# The problem does not say a team needs at least 1 snr manager, so technially␣
 ↪the team can be 5 jr mngr
only_jnr = math.comb(14,0)*math.comb(13,5)
possibilities = only_jnr+snr_mgt_inc_case
print(possibilities)
#Answer: 11,297 different team possibilities.
```

```
11297
```

(Marketing Campaign Outcomes): A marketing campaign involves three stages: first, a customer

is sent 5 email offers; second, the customer is targeted with 2 different online ads; and third, the customer is presented with 3 personalized product recommendations. If the email offers, online ads, and product recommendations are selected randomly, how many different possible outcomes are there for the entire campaign?

[93]:
```
# 3 stages

#1) 5 email offer
#2) 2 onlin ad
#3) 3 prod. recs.

answer = 5 * 2 * 3
print(answer)
#Answer: THere are 30 different outcomes.
```

```
30
```

(Product Defect Probability): A quality control team draws 3 products from a batch of size N without replacement. What is the probability that at least one of the products drawn is defective if the defect rate is known to be consistent?

[97]:
```
# 3 products from N (no replacement)
# Consistent defect rate
# Say if 20% defect rate, 80% success rate
# .8^3 for none being defective
#
print("Changes non defective with example constant rate:")
print(0.8**3)
print("Subtracting from whole number for AT LEAST 1 is defective")
print(1 - (0.8**3))




#Answer: Using the constant error rate of 20% there is a 48.7% chance that one␣
 ↪is defective. This formuala can be applied to any
# other constant defect rate.
```

```
Changes non defective with example constant rate:
0.5120000000000001
Subtracting from whole number for AT LEAST 1 is defective
0.4879999999999999
```

(Business Strategy Choices): A business strategist is choosing potential projects to invest in, focusing on 17 high-risk, high-reward projects and 14 low-risk, steady-return projects.

[100]:
```
# 17 high risk / high reward
# 14 low risk / steady return
```

```
#Step 1: How many different combinations of 5 projects can the strategist␣
 ↪select?

## Assuming no mandatory compositions, the total number of project is 31.
no_limit_combintations = math.comb(31, 5)
print(no_limit_combintations)
#Answer: 169,911 different combinations of 5 projects for a portfolio with no␣
 ↪mandates.

#Step 2: How many different combinations of 5 projects can the strategist␣
 ↪select if they want at least one low-risk project?

#with at least one low risk
case1 = math.comb(17,4)*math.comb(14,1)
case2 = math.comb(17,3)*math.comb(14,2)
case3 = math.comb(17,2)*math.comb(14,3)
case4 = math.comb(17,1)*math.comb(14,4)
case5 = math.comb(17,0)*math.comb(14,5)
#possiblities for combinations
possiblities = case1+case2+case3+case4+case5
print(possiblities)
#Answer: When having the mandata of at least one low risk project there are␣
 ↪163,723 possiblities
```

```
169911
163723
```

(Event Scheduling): A business conference needs to schedule 9 different keynote sessions from three different industries: technology, finance, and healthcare. There are 4 potential technology sessions, 104 finance sessions, and 17 healthcare sessions to choose from. How many different schedules can be made? Express your answer in scientific notation rounding to the hundredths place.

[106]:
```
# 9 different keynotes
# 3 diff industries (tech, fin, health)
# 4 tech
# 104 fin
# 17 health

### First there are 125 total different sessions
#### Order does matter for selection, not replaced
## No limit combinations
total_number_comb = math.comb(125,9)

## Now take a look at the scenarios that we dont want. So all those that are␣
 ↪not inclusive of each sector.
no_health = math.comb(108,9)
no_fin =  math.comb(21,9)
no_tech =  math.comb(121,9)
```

```
#Taking the total number of combinations and taking out those that dont include␣
 ↪each sector
abiding_combs = total_number_comb - (no_health + no_fin + no_tech)


## Now takign the abiding combs and considering that the order matters
answer = abiding_combs * math.factorial(9)
print(answer)

#Answer: There are a total of 2.98 e^16 different conference lineups that can␣
 ↪take place
```

```
29761863254323200
```

(Book Selection for Corporate Training): An HR manager needs to create a reading list for a corporate leadership training program, which includes 13 books in total. The books are categorized into 6 novels, 6 business case studies, 7 leadership theory books, and 5 strategy books.

[113]:
```
#13 books total
# 6 novels
# 6 bus. case studies
# 7 leadership theory
# 5 strategy


#Step 1: If the manager wants to include no more than 4 strategy books, how␣
 ↪many different reading schedules are possible?
#Express your answer in scientific notation rounding to the hundredths place.

#<= 4 strategy books; no other mandates
# order doesnt matter

## Total comb without Strategy
total_wo_strat = 6 + 6 + 7
#print(total_wo_strat)

case0 = math.comb(total_wo_strat,13)*math.comb(5,0)
case1 = math.comb(total_wo_strat,12)*math.comb(5,1)
case2 = math.comb(total_wo_strat,11)*math.comb(5,2)
case3 = math.comb(total_wo_strat,10)*math.comb(5,3)
case4 = math.comb(total_wo_strat,9)*math.comb(5,4)

total_possibilities = case0+case1+case2+case3+case4
print(total_possibilities)

#Answer: 2.42 e^6 total possibilities
```

```
#Step 2: If the manager wants to include all 6 business case studies, how many different reading schedules are possible?
#Express your answer in scientific notation rounding to the hundredths place.

## All 6 case studies included no other mandates
# order doesnt matter

remaining_comb = 13 - 6
non_casestudy = 6 + 7+ 5

part2_possibilities =  math.comb(non_casestudy, remaining_comb)
print(part2_possibilities)

#Answer: 3.18e^4 Total possibilities when all case studies must be included.
```

2420562
31824

(Product Arrangement): A retailer is arranging 10 products on a display shelf. There are 5 different electronic gadgets and 5 different accessories. What is the probability that all the gadgets are placed together and all the accessories are placed together on the shelf? Express your answer as a fraction or a decimal number rounded to four decimal places.

[ ]:
```
# 10 products displayed
# 5 electric
# 5 accessories

## Total different permutations for all items = 10!
total = math.factorial(10)

## Possible for accessories
accessory = math.factorial(5)

## possible device
device = math.factorial(5)

# Since the order of the two groups of products matters we can do:
total_combinations_for_order = math.factorial(2)*device*accessory

## for probabilty we need percentage
prob_ordered = round((total_combinations_for_order/total)*100,4)
print(prob_ordered)

#Answer: There are only ten total products and accessories, so all of them need to be placed on the shelf. However,
# if the problem is asking about if accessories are placed next to each other and all products are next to each other,
# the chances are 0.79% chance.
```

(Expected Value of a Business Deal): A company is evaluating a deal where they either gain 4 dollars for every successful contract or lose 16 dollars for every unsuccessful contract. A "successful" contract is defined as drawing a queen or lower from a standard deck of cards. (Aces are considered the highest card in the deck.)

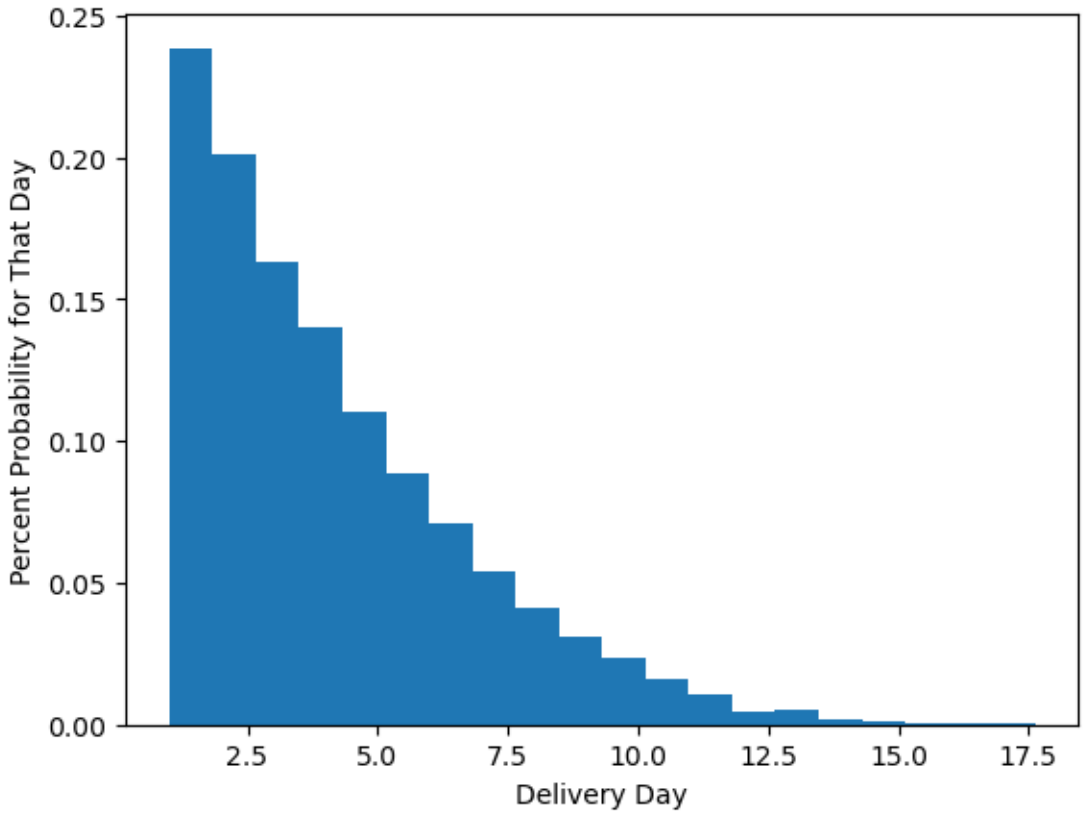
[122]:
```
# + 4 successful contract
# - 16 unseccessful contract

## Successful odds = (52-8)/52 => 44/52 chances of success
#4 aces
#4 kings


#Step 1: Find the expected value of the deal. Round your answer to two decimal␣
 ↪places.
#Losses must be expressed as negative values.

#Expected value for this would be (44/52=0.846 Success rate, 8/52 =0.154␣
 ↪Unsuccesful Rate)
#This means expected cost would be:
success = (44/52)*4
unsuccess = (8/52)*-16
expected_value = unsuccess + success
print(round(expected_value,2))
#Answer: $0.92 expected value per deal

#Step 2: If the company enters into this deal 833 times, how much would they␣
 ↪expect to win or lose?
#Round your answer to two decimal places. Losses must be expressed as negative␣
 ↪values.

answer = expected_value*833
print(round(answer,2))

#Answer: $ 768.92 is the expected win from this scenario.
```

```
0.92
768.9230769230768
```

0.0.4 Problem 3

(Supply Chain Risk Assessment): Let X1,X2 ... Xn represent the lead times (in days) for the delivery of key components from n = 5 different suppliers. Each lead time is uniformly distributed across a range of 1 to K = 20 days, reflecting the uncertainty in delivery times. Let Y denote the minimum delivery time among all suppliers. Understanding the distribution of Y is crucial for assessing the earliest possible time you can begin production. Determine the distribution of Y to better manage your supply chain and minimize downtime.

[16]:
```
# n = 5 (suppliers)
# 1 - 20 days; uniform distributions
# minimum delivery time for first delivery
# find distribution of Y

import numpy as np
import matplotlib.pyplot as plt
min_deliveries = []
for trial in range(10000):
    sample = np.random.uniform(1, 20, size=5)
    earliest_days = sample.min()
    min_deliveries.append(earliest_days)

plt.hist(min_deliveries, bins=20, density=True)
plt.xlabel("Delivery Day")
plt.ylabel("Percent Probability for That Day")
plt.show()

## The distribtuion of the earliest delivery times for this problem, after␣
 ↪running 10,000 uniform distribution simulations
## the earliest delivery distribution is left skewed with the most common␣
 ↪earliest delivery days being within the first 2 days.
```



(Maintenance Planning for Critical Equipment): Your organization owns a critical piece of equipment, such as a high-capacity photocopier (for a law firm) or an MRI machine (for a healthcare provider). The manufacturer estimates the expected lifetime of this equipment to be 8 years, meaning that, on average, you expect one failure every 8 years. It's essential to understand the likelihood of failure over time to plan for maintenance and replacements.

[20]:
```
# expected lifetime is 8 years
# one failure every 8 years so each year 1/8 = 0.125
p=0.125
```

a. Geometric Model: Calculate the probability that the machine will not fail for the first 6 years. Also, provide the expected value and standard deviation. This model assumes each year the machine either fails or does not, independently of previous years.

[22]:
```
# no failure for first 6 years: probability
from scipy.stats import geom
prob = geom.sf(6, p)
print(round(prob*100,2), " is the probability that the machine will not fail in␣
 ↪the first 6 years.")
exp_val = geom.mean(p)
std_dev= geom.std(p)
print(exp_val,"is the expected value.")
print(round(std_dev,2),"is the standard distribution.")
```

```
44.88  is the probability that the machine will not fail in the first 6 years.
8.0 is the expected value.
7.48 is the standard distribution.
```

b. Exponential Model: Calculate the probability that the machine will not fail for the first 6 years. Provide the expected value and standard deviation, modeling the time to failure as a continuous process.

[24]:
```
#no fail for the first 6 years
from scipy.stats import expon
prob = expon.sf(6, p)
print(round(prob*100,2), " is the probability that the machine will not fail in␣
 ↪the first 6 years.")
exp_val = expon.mean(p)
std_dev= expon.std(p)
print(exp_val,"is the expected value.")
print(round(std_dev,2),"is the standard distribution.")
```

```
0.28  is the probability that the machine will not fail in the first 6 years.
1.125 is the expected value.
1.0 is the standard distribution.
```

c. Binomial Model: Calculate the probability that the machine will not fail during the first 6 years, given that it is expected to fail once every 8 years. Provide the expected value

and standard deviation, assuming a fixed number of trials (years) with a constant failure probability each year.

[28]:
```
#no fail for the first 6 years. zero failure in 6 years
from scipy.stats import binom
prob = binom.pmf(0, 6, p)
print(round(prob*100,2), " is the probability that the machine will not fail in␣
 ↪the first 6 years.")
exp_val = binom.mean(6,p)
std_dev= binom.std(6,p)
print(exp_val,"is the expected value.")
print(round(std_dev,2),"is the standard distribution.")
```

```
44.88  is the probability that the machine will not fail in the first 6 years.
0.75 is the expected value.
0.81 is the standard distribution.
```

d. Poisson Model: Calculate the probability that the machine will not fail during the first 6 years, modeling the failure events as a Poisson process. Provide the expected value and standard deviation.

[34]:
```
# no fail for the first 6 years.
from scipy.stats import poisson
# convertign 1 year prob to 6 year window
l = p*6
prob =poisson.pmf(0,l)
print(round(prob*100,2), " is the probability that the machine will not fail in␣
 ↪the first 6 years.")
exp_val = poisson.mean(l)
std_dev= poisson.std(l)
print(exp_val,"is the expected value.")
print(round(std_dev,2),"is the standard distribution.")
```

```
47.24  is the probability that the machine will not fail in the first 6 years.
0.75 is the expected value.
0.87 is the standard distribution.
```

0.0.5 Problem 4

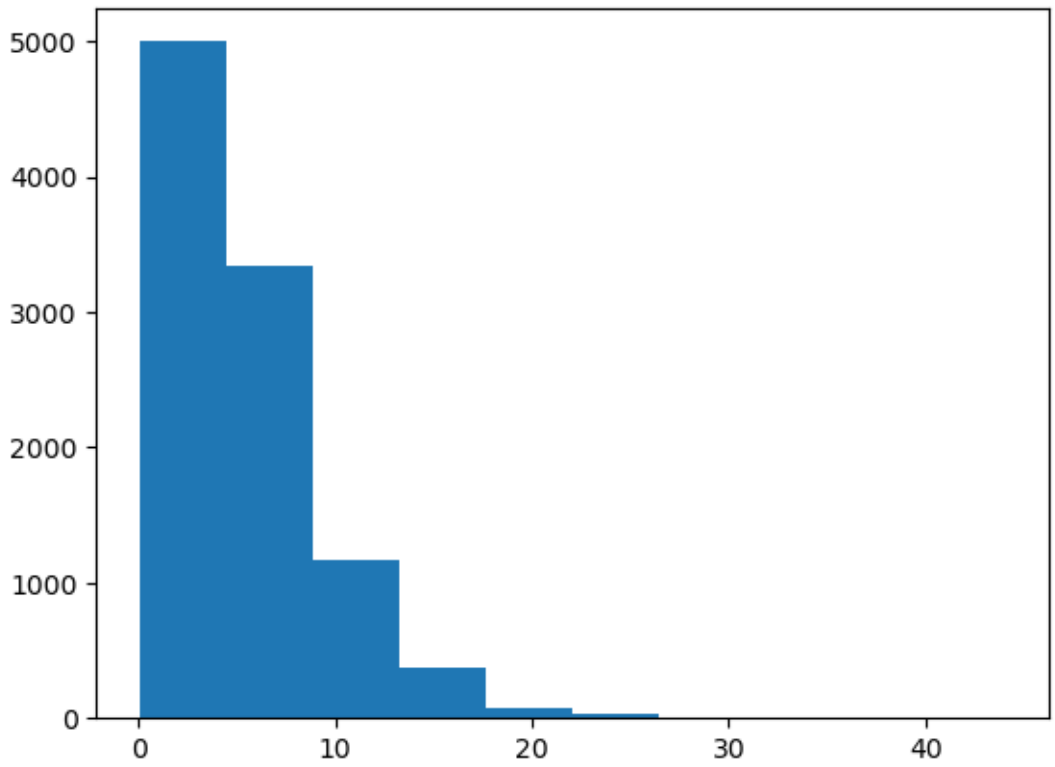
1. Scenario: You are managing two independent servers in a data center. The time until the next failure for each server follows an exponential distribution with different rates:

- Server A has a failure rate of Lambda_A = 0.5 failures per hour.
- Server B has a failure rate of Lambda_B = 0.3 failures per hour.

Question: What is the distribution of the total time until both servers have failed at least once? Use the moment generating function (MGF) to find the distribution of the sum of the times to failure.

[41]:
```python
#Exponential dist.

import numpy as np
import matplotlib.pyplot as plt
lambda_A = 0.5
lambda_B = 0.3

# THis means 1 failure every 2 hours for Server A
# This means 1 failure every ~3.33 hours for Server B


samples_A = np.random.exponential(scale=2, size=10000)
samples_B = np.random.exponential(scale=3.33, size=10000)

# plt.hist(samples_A)
# print("Lambda A Scenarios")
# plt.show()
# plt.hist(samples_B)
# print("Lambda B Scenarios")
# plt.show()
print("Combined Total Scnearios(Answer)")
total = samples_A+samples_B
plt.hist(total)
plt.show()
```

```
Combined Total Scnearios(Answer)
```

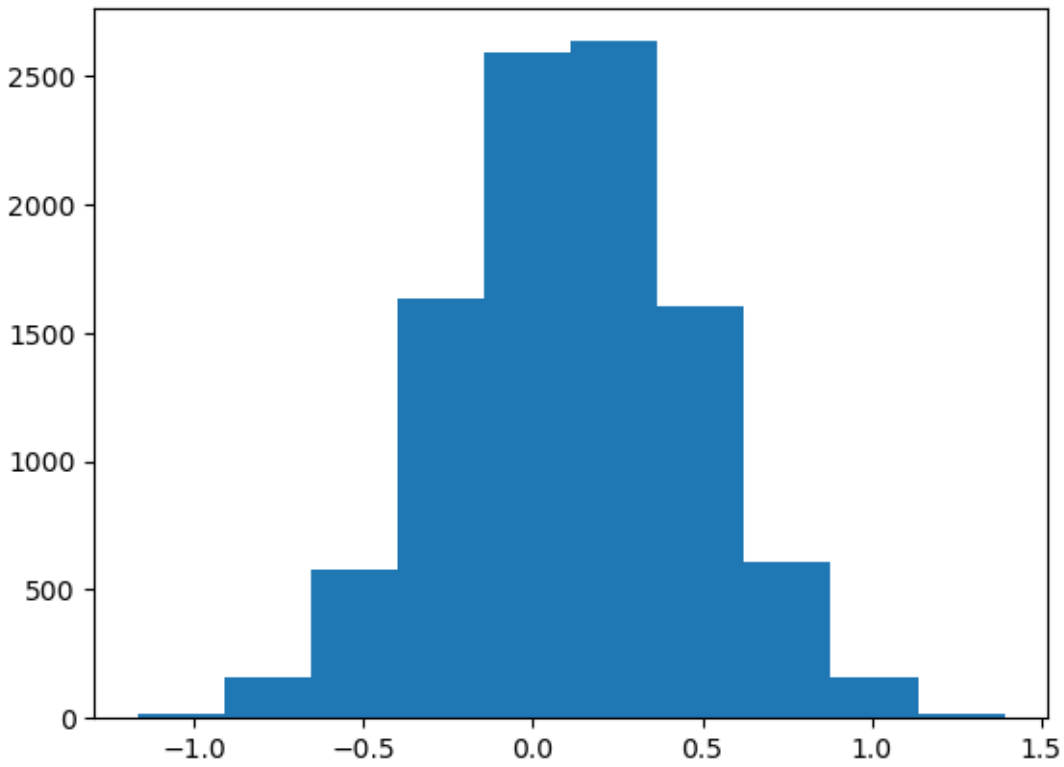


2. Sum of Independent Normally Distributed Random Variables Scenario: An investment firm is analyzing the returns of two independent assets, Asset X and Asset Y. The returns on these assets are normally distributed:

- Asset X: X ~ N(Mean x = 5%, simga^2 x = 4%)
- Asset Y: U ~ N(Mean y = 7%, simga^2 y = 9%)

Question: Find the distribution of the combined return of the portfolio consisting of these two assets using the moment generating function (MGF).

[46]:
```
import math

# Norm dist.
# X mean of 5 percent return with std dev of sqrt 4%
x_std_dev = math.sqrt(4/100)
# Y mean of 7 percent return with std dev of sqrt 9%
y_std_dev = math.sqrt(9/100)

x = np.random.normal(loc=5/100, scale=x_std_dev, size=10000)
y = np.random.normal(loc=7/100, scale=y_std_dev, size=10000)
total = x+y
plt.hist(total)
plt.show()
```

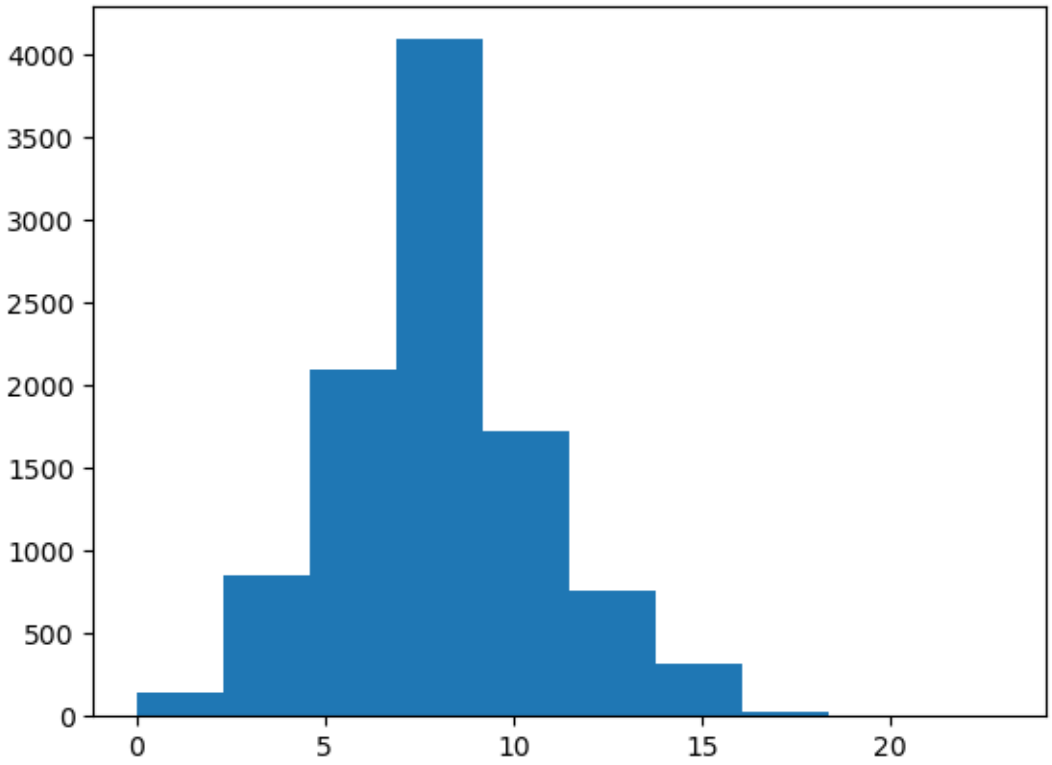


3. Scenario: A call center receives calls independently from two different regions. The number of calls received from Region A and Region B in an hour follows a Poisson distribution:

- Region A: Xa ~ Poisson(Lambda_a = 3)
- Region B: Xb ~ Poisson(Lambda_b = 5)

Question: Determine the distribution of the total number of calls received in an hour from both regions using the moment generating function (MGF).

[48]:
```
#poisson
reg_a = np.random.poisson(3,size=10000)
reg_b = np.random.poisson(5,size=10000)

total=  reg_a+reg_b
plt.hist(total)
plt.show()
```



0.0.6 Problem 5

1. Customer Retention and Churn Analysis

Scenario: A telecommunications company wants to model the behavior of its customers regarding their likelihood to stay with the company (retention) or leave for a competitor (churn). The company segments its customers into three states:

```
State 1: Active customers who are satisfied and likely to stay (Retention state).
State 2: Customers who are considering leaving (At-risk state).
State 3: Customers who have left (Churn state).
```

The company has historical data showing the following monthly transition probabilities:

```
- From State 1 (Retention): 80% stay in State 1, 15% move to State 2, and 5% move to State 3.
- From State 2 (At-risk): 30% return to State 1, 50% stay in State 2, and 20% move to State 3.
- From State 3 (Churn): 100% stay in State 3.
```

The company wants to analyze the long-term behavior of its customer base.

Question: - (a) Construct the transition matrix for this Markov Chain.

- (b) If a customer starts as satisfied (State 1), what is the probability that they will eventually churn (move to State 3)? -
- (c) Determine the steady-state distribution of this Markov Chain. What percentage of customers can the company expect to be in each state in the long run?

[67]:
```
#(a) Construct the transition matrix for this Markov Chain.
transition_matrix = np.array([
    #State 1
    [0.8,0.15,0.05],
    #State 2
    [0.3,0.5,0.2],
    #State3
    [0,0,1]])
print(transition_matrix)
print("-----")

#(b) If a customer starts as satisfied (State 1), what is the probability that␣
 ↪they will eventually churn (move to State 3)? -

initial_state =[1,0,0]
second_cut = initial_state @ transition_matrix
print(second_cut)
third_cut = second_cut@transition_matrix
print(third_cut)
print(f"{round(third_cut[-1]*100,2)} percent is the chances of moving to state␣
 ↪3 after 2 months.")
print('_____')

# Doing for longer term simulation
current_state =[1,0,0]
for i in range(100):
    current_state = current_state@transition_matrix
print(f"{round(current_state[-1]*100,2)} percent is the chances of moving to␣
 ↪state 3 after 100 months.")

#ANSWER: Overall there is a 99.99 percent chance that a customer ends up in␣
 ↪state 3.

#(c) Determine the steady-state distribution of this Markov Chain.
# What percentage of customers can the company expect to be in each state in␣
 ↪the long run?

#Finding steady state
current_state =[1,0,0]
for i in range(10000):
    current_state = current_state@transition_matrix
print(f"{current_state} is the steady-state of the distruction, it was reached␣
 ↪in my simulation at month number {i+1}.")
# ANSWER: The steady state in theory should be [0,0,100]. But the numbers in␣
 ↪the above print statemen are just super small numbers.
```

```
[[0.8  0.15 0.05]
```

```
 [0.3  0.5  0.2 ]
 [0.   0.   1.  ]]
-----
[0.8  0.15 0.05]
[0.685 0.195 0.12 ]
12.0 percent is the chances of moving to state 3 after 2 months.

-----
99.99 percent is the chances of moving to state 3 after 100 months.
[3.5e-323 9.9e-324 1.0e+000] is the steady-state of the distruction, it was
reached in my simulation at month number 10000.
```

2: Inventory Management in a Warehouse Scenario: A warehouse tracks the inventory levels of a particular product using a Markov Chain model. The inventory levels are categorized into three states: - State 1: High inventory (More than 100 units in stock). - State 2: Medium inventory (Between 50 and 100 units in stock). - State 3: Low inventory (Less than 50 units in stock).

The warehouse has the following transition probabilities for inventory levels from one month to the next: - From State 1 (High): 70% stay in State 1, 25% move to State 2, and 5% move to State 3. - From State 2 (Medium): 20% move to State 1, 50% stay in State 2, and 30% move to State 3. - From State 3 (Low): 10% move to State 1, 40% move to State 2, and 50% stay in State 3.

The warehouse management wants to optimize its restocking strategy by understanding the long-term distribution of inventory levels.

Question: - (a) Construct the transition matrix for this Markov Chain. - (b) If the warehouse starts with a high inventory level (State 1), what is the probability that it will eventually end up in a low inventory level (State 3)? - (c) Determine the steady-state distribution of this Markov Chain. What is the long-term expected proportion of time that the warehouse will spend in each inventory state?

[68]:
```
#(a) Construct the transition matrix for this Markov Chain.
transition_matrix = np.array([
    # State1
    [0.7,0.25,0.05],
    # State2
    [0.2,0.5,0.3],
    # State3
    [0.1,0.4,0.5]
])

#(b) If the warehouse starts with a high inventory level (State 1).
#what is the probability that it will eventually end up in a low inventory␣
 ↪level (State 3)?

initial_state =[1,0,0]
second_cut = initial_state @ transition_matrix
print(second_cut)
third_cut = second_cut@transition_matrix
print(third_cut)
```

```
print(f"{round(third_cut[-1]*100,2)} percent is the chances of moving to state␣
 ↪3 after 2 months.")
print('_____')

# Doing for longer term simulation
current_state =[1,0,0]
for i in range(100):
    current_state = current_state@transition_matrix
print(f"{round(current_state[-1]*100,2)} percent is the chances of moving to␣
 ↪state 3 after 100 months.")

#ANSWER: Overall there is a ~27% chance that a customer ends up in state 3.

#(c) Determine the steady-state distribution of this Markov Chain.
#What is the long-term expected proportion of time that the warehouse will␣
 ↪spend in each inventory state?

#Finding steady state
current_state =[1,0,0]
for i in range(10000):
    current_state = current_state@transition_matrix

print(f"{current_state} is the steady-state of the distruction, it was reached␣
 ↪in my simulation at month number {i+1}.")
#ANSWER: Long term expectation for the states are: about 35%, 37%, and 27%.
```

```
[0.7  0.25 0.05]
[0.545 0.32  0.135]
13.5 percent is the chances of moving to state 3 after 2 months.
-----
26.67 percent is the chances of moving to state 3 after 100 months.
[0.34666667 0.38666667 0.26666667] is the steady-state of the distruction, it
was reached in my simulation at month number 10000.
```