MATH 118: Statistics and Probability **(Due: 07/06/21)**

Homework #2

Instructor: Dr. Zafeirakis Zafeirakopoulos *Name:* *Student Id:*
Assistant: Gizem Süngü

Course Policy: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- It is not a group homework. Do not share your answers to anyone in any circumstance. Any cheating means at least -100 for both sides.
- Do not take any information from Internet.
- No late homework will be accepted.
- For any questions about the homework, send an email to gizemsungu@gtu.edu.tr.
- Submit your homework (both your latex and pdf files in a zip file) into the course page of Moodle.
- Save your latex, pdf and zip files as "Name_Surname_StudentId".{tex, pdf, zip}.
- The answer which has only calculations without any formula and any explanation will get zero.
- The deadline of the homework is 07/06/20 23:55.
- I strongly suggest you to write your homework on LaTeX. However, hand-written paper is still accepted **IFF** your hand writing is **clear and understandable to read**, and the paper is well-organized. Otherwise, I cannot grade your homework.
- You do not need to write your Student Id on the page above. I am checking your ID from the file name.

Problem 1: (10+10+10+10+10+10+40 = 100 points)

WARNING: Please show your OWN work. Any cheating can be easily detected and will not be graded.

For the question, please follow the file called manufacturing_defects.txt while reading the text below.

In each year from 2000 to 2019, the number of manufacturing defects in auto manufacturers were counted. The data was collected from 14 different auto manufactory companies. The numbers of defects for the companies are indicated in 14 columns following the year column. Assume that the number of manufacturing defects per auto company per year is a random variable having a Poisson(λ) and that the number of defects in different companies or in different years are independent.
(Note: You should implement a code for your calculations for each following subproblem. You are free to use any programming languages (Python, R, C, C++, Java) and their related library.)

(a) Give a table how many cases occur for all companies between 2000 and 2019 for each number of defects (# of Defects).
Hint: When you check the file you will see: # of Defects = {0, 1, 2, 3, 4}.

(b) Estimate λ from the given data.

(c) Update Table 1 in Table 2 with Poisson predicted cases with the estimated λ.

(d) Draw a barplot for the actual cases (Table 2 in column 2) and the predicted cases (Table 2 column 3) with respect to # of defecrs. You should put the figure.

\# of Defects	\# of cases in all company between the years
0	
1	
2	
3	
4	

Table 1: Actual cases

\# of Defects	\# of cases in all companies between the years	Predicted \# of cases in all companies between the years
0		
1		
2		
3		
4		

Table 2: Actual vs. Predicted Cases

(e) According to the barplot in (c), does the poisson distribution fit the data well? Compare the values of the actual cases and the values of the poisson predicted cases, and write your opinions about performance of the distribution.
Yes the poisson distribution fit the data,because he majority of real cases and predicted cases are close to or equal to each other.
For First Company;
Actual Cases (0,1,2,3,4) are = 9 7 3 1 0
Poisson Predicted Cases (0,1,2,3,4) are also = 9 7 3 1 0
They are the same as each other. Although For Fourth Company;
Actual Cases (0,1,2,3,4) are = 11 6 3 0 0
Poisson Predicted Cases (0,1,2,3,4) are also = 11 7 2 0 0
Total Actual And Predicted Cases;
Actual Cases (0,1,2,3,4) are = 144 91 32 11 2
Poisson Predicted Cases (0,1,2,3,4) are also = 139 97 34 7 1
Since the total values are close to each other in the same way, it can be said to be fit.
Values are close to each other, but they are not exactly the same.
While the distribution finds the same values for some companies and also for most companies it gives close values.I can say that it has a good performance as it gives close prediction for most companies.

(f) According to your estimations above, write your opinions considering your barplot and Table 2. Which company do you prefer to buy a car? Why?
If I'm going to buy a car, I prefer the 9th company.
It's actual # of Defects {0, 1, 2, 3, 4} = {13, 7, 0, 0,0}
It's predicted # of Defects {0, 1, 2, 3, 4} = {14, 5, 1, 0, 0}
I prefer 9th company Because;
1)One of the companies with at most 0 actual defect cases.
2)This company has a lat of 0 actual defect cases and also has no any 2 3 4 actual defect cases it has 0 and 1 cases.This also makes it successful.
3)Actual cases are lower than predicted cases for all values. This shows that the cars produced perform better than the predictions made.

(f.2) According to your estimations above, write your opinions considering your barplot and Table 2. Do you think that road transportation is dangerous for us? Whether yes or no, explain your reason.

Yes, I think road transportation is dangerous. Since actual cases are higher than predicted cases in barplots and tables, dangerous results may occur when doing road transportation.

(g) Paste your code that you implemented for the subproblems above. Do not forget to write comments on your code.
Example:

- The common code block for all subproblems
  Paste here. Your code should read the file and compute other things which the following subproblems need.

```
def readFromFile():
    openFile = open("manufacturing_defects.txt", "r")  # Open file
    array = []
    for line in openFile:  # Fill 2d list from file
        line = line.rstrip('\n')
        row = line.split()  # split each line
        if len(row) != 0: # Pass free lines
            array.append(row)
    openFile.close()
    return array  # Return 2d list

def printTables(totalDefects, allpredicted):  # Print Tables to terminal
        print("\n\nTable 1: Actual Cases")
        print("\# of Defects \t\t # of cases in all company between the years(For
Total Cases")
        for i in range(0, 5):
            print("%2d" % (totalDefects[i][0]), "\t\t\t\t\t", end=" ")
            for x in range(len(totalDefects[i]) - 1):
                print("%2d" % (totalDefects[i][x + 1]), " ", end=" ")
            print()

        print("\n\nTable 2: Actual Cases vs Predicted Cases")
        print("\# of Defects \t\t # of cases in all company between the years(For
Total Cases\t\tPredicted \# of cases in all "
            "companies between the years\t\t     Total Cases")
        for i in range(0, 5):
            print("%2d" % (totalDefects[i][0]), "\t\t\t\t\t", end=" ")
            for x in range(len(totalDefects[i]) - 1):
                print("%2d" % (totalDefects[i][x + 1]), " ", end=" ")

            print("\t\t\t", end="")
            for x in range(len(allpredicted[i])):
                print("%2d" % (allpredicted[i][x]), " ", end=" ")
            print()


def printLamdas(allLambda):
        print("Lambdas for each country: ",end= " ")
        for i in range(0,14):
            print(allLambda[i], end=" ")
        print()
        print("Lambda:", allLambda[14])
```

- The code block for (a)
  Paste here. Your code should compute the values in Table 1 column 2.

```
def total_of_defect(allData): # Find Actual Cases
    w, h = 15, 5
```

```
            totalDefects = [[0 for x in range(w)] for y in range(h)]
            totalDefects[0][0] = 0
            totalDefects[1][0] = 1
            totalDefects[2][0] = 2
            totalDefects[3][0] = 3
            totalDefects[4][0] = 4
            k = 0
            for i in allData: # Find 0, 2, 3, 4 actual cases with nested loops
                if i != []:
                    for j in range(2, 16):
                        if int(i[j]) == 0:
                            totalDefects[0][j - 1] = totalDefects[0][j - 1] + 1
                        if int(i[j]) == 1:
                            totalDefects[1][j - 1] = totalDefects[1][j - 1] + 1
                        if int(i[j]) == 2:
                            totalDefects[2][j - 1] = totalDefects[2][j - 1] + 1 # In 0
find it while traverse the # list.
                        if int(i[j]) == 3:
                            totalDefects[3][j - 1] = totalDefects[3][j - 1] + 1
                        if int(i[j]) == 4:
                            totalDefects[4][j - 1] = totalDefects[4][j - 1] + 1
            total = 0
            k = 0
            for i in totalDefects:
                for j in range (1,16):
                    total += i[j]
                    totalDefects[k][15] = total
                    total = 0
                k+=1
            return totalDefects # Return all actual cases
```

- The code block for (b)
  Paste here. Your code should compute λ.

```
    def findLambda(totalDefects):  # Find Lambda
            k = 1
            allLambda = []
            for i in range(0, 14): # Find lambda for each country with calculated 0,1,
                temp = ((totalDefects[0][k] * 0) + (totalDefects[1][k] * 1) + (totalDe
                        totalDefects[3][k] * 3) + (totalDefects[4][k] * 4)) / 20
                allLambda.append(temp)
                k += 1
                avarageL = 0
            for i in allLambda:
                avarageL += i
            avarageL = avarageL / 14
            allLambda.append(avarageL)
            return allLambda # Return lambdas
```

- The code block for (c)
  Paste here. Your code should compute the values in Table 2 column 3.

```
    def predictedDefects(allLambda):  # Find Predicted Cases
            tempLambda = 0
            allPredicted = []
            newPredicted = []
            x = 0
            px = (pow(math.e, -tempLambda) * pow(tempLambda, x)) / math.factorial(x)
# Use p(x) formula for find predicted values
```

```
    k = 0
    for i in range(0, 14): # Calculate for each country and round it
        tempLambda = allLambda[k]
        zero = ((pow(math.e, -tempLambda) * pow(tempLambda, 0)) / math.factori
        one = ((pow(math.e, -tempLambda) * pow(tempLambda, 1)) / math.factoria
        two = ((pow(math.e, -tempLambda) * pow(tempLambda, 2)) / math.factoria
        three = ((pow(math.e, -tempLambda) * pow(tempLambda, 3)) / math.factor
        four = ((pow(math.e, -tempLambda) * pow(tempLambda, 4)) / math.factori
        zero = round(zero)
        one = round(one)
        two = round(two)
        three = round(three)
        four = round(four)
        allPredicted.append([zero, one, two, three, four])
        k += 1
    for i in range(0, 5): # Fill predicted cases
        tempList = []
        for j in range(0, 14):
            tempList.append(allPredicted[j][i])

        newPredicted.append(tempList)
    tempLambda = allLambda[14]

    zero = ((pow(math.e, -tempLambda) * pow(tempLambda, 0)) / math.factorial(0
    one = ((pow(math.e, -tempLambda) * pow(tempLambda, 1)) / math.factorial(1)
    two = ((pow(math.e, -tempLambda) * pow(tempLambda, 2)) / math.factorial(2)
    three = ((pow(math.e, -tempLambda) * pow(tempLambda, 3)) / math.factorial(
    four = ((pow(math.e, -tempLambda) * pow(tempLambda, 4)) / math.factorial(4
    newPredicted[0].append(zero)
    newPredicted[1].append(one)
    newPredicted[2].append(two)
    newPredicted[3].append(three)
    newPredicted[4].append(four)
    return newPredicted # Return Predicted Cases
```

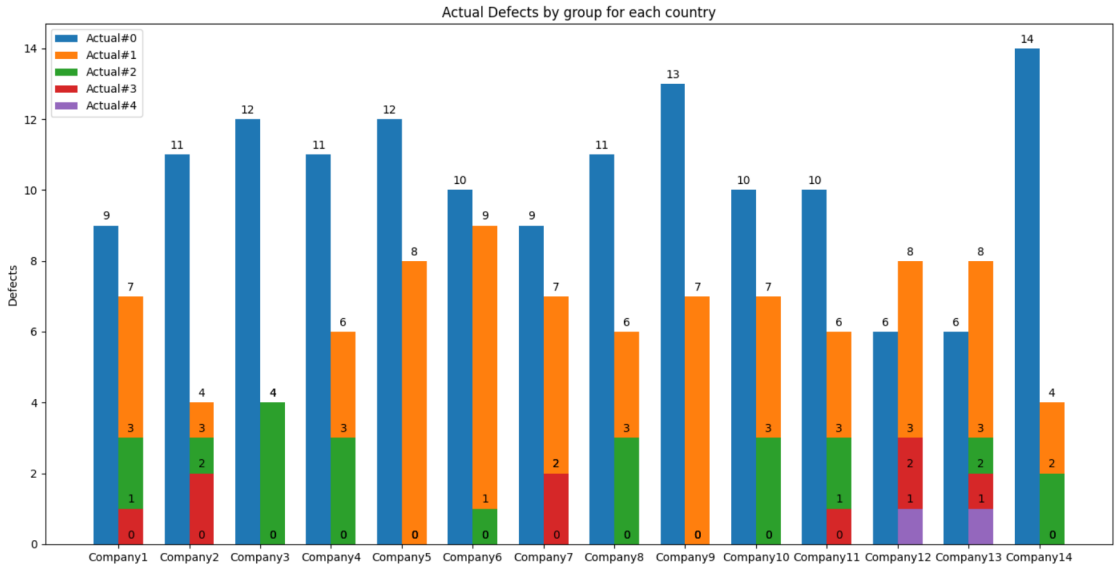
- The code block for (d)
  Paste here. Your code should draw the barplot.

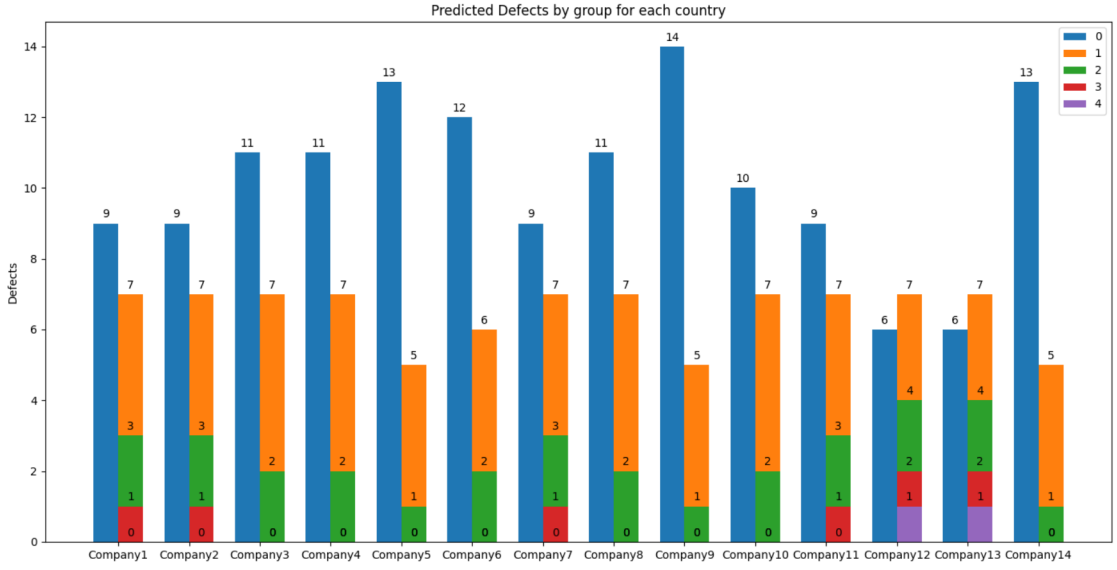
```
def actualDefectsPlot(totalDefects, allPredicted): # Draw Bar Plot for actual Cases
    labels = ['Company1', 'Company2', 'Company3', 'Company4', 'Company5', 'Company6',
              'Company9',
              'Company10', 'Company11', 'Company12', 'Company13', 'Company14']
    x = np.arange(len(labels))
    width = 0.35

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width / 2, totalDefects[0][1:15], width, label='Actual#0')
# matplotlib.pyplot library function to fill plot
    rects2 = ax.bar(x + width / 2, totalDefects[1][1:15], width, label='Actual#1')
    rects3 = ax.bar(x + width / 2, totalDefects[2][1:15], width, label='Actual#2')
    rects4 = ax.bar(x + width / 2, totalDefects[3][1:15], width, label='Actual#3')
    rects5 = ax.bar(x + width / 2, totalDefects[4][1:15], width, label='Actual#4')

    ax.set_ylabel('Defects') # Title of Plot
    ax.set_title('Actual Defects by group for each country') # Numbers name of plot
    ax.set_xticks(x)
    ax.set_xticklabels(labels)
    ax.legend()
```

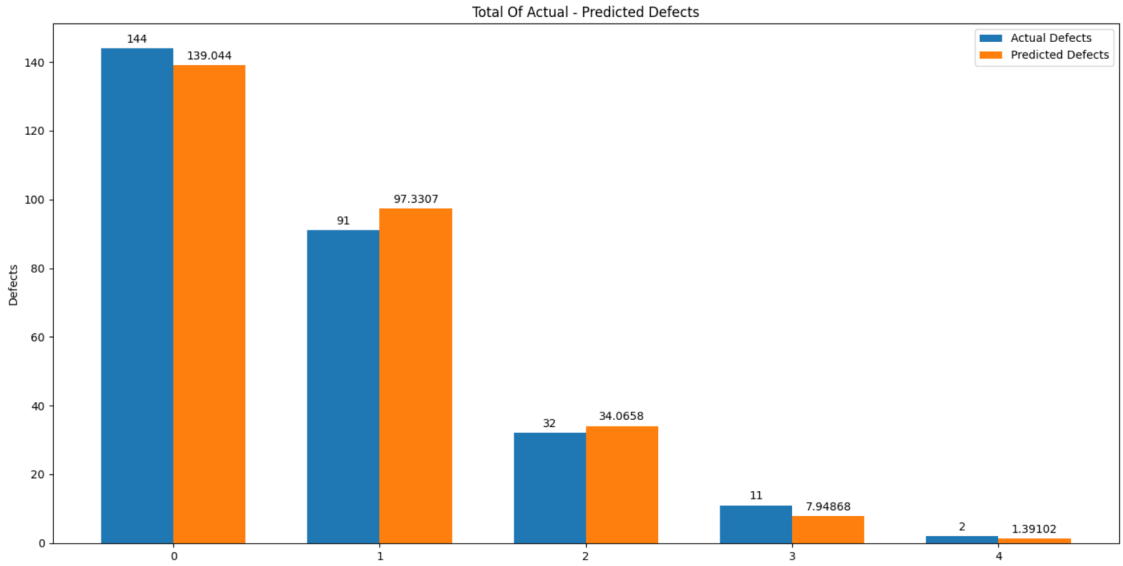
```
    ax.bar_label(rects1, padding=3)
    ax.bar_label(rects2, padding=3)
    ax.bar_label(rects3, padding=3)
    ax.bar_label(rects4, padding=3)
    ax.bar_label(rects5, padding=3)
    fig.tight_layout()
    plt.show()  # Show Plot


def predictedDefectsPlot(totalDefects, newPredicted): # Draw Bar Plot for Predicted Ca
    tempList = []
    labels = ['Company1', 'Company2', 'Company3', 'Company4', 'Company5', 'Company6',
              'Company9',
              'Company10', 'Company11', 'Company12', 'Company13', 'Company14']
    print("all predicted", allPredicted[0][0])

    x = np.arange(len(labels))
    width = 0.35

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width / 2, newPredicted[0], width, label='0')  # matplotlib.py
    rects2 = ax.bar(x + width / 2, newPredicted[1], width, label='1')
    rects3 = ax.bar(x + width / 2, newPredicted[2], width, label='2')
    rects4 = ax.bar(x + width / 2, newPredicted[3], width, label='3')
    rects5 = ax.bar(x + width / 2, newPredicted[4], width, label='4')

    ax.set_ylabel('Defects')  # Title of Plot
    ax.set_title('Predicted Defects by group for each country')  # Numbers name of plo
    ax.set_xticks(x)
    ax.set_xticklabels(labels)
    ax.legend()

    ax.bar_label(rects1, padding=3)
    ax.bar_label(rects2, padding=3)
    ax.bar_label(rects3, padding=3)
    ax.bar_label(rects4, padding=3)
    ax.bar_label(rects5, padding=3)

    fig.tight_layout()
    plt.show()  # Show Plot
    return newPredicted


def totalPlot(totalDefects, allPredicted): # Draw Bar Plot for actual Cases
    actualTotal = []
    predictedTotal = []
    labels = ['0', '1', '2', '3', '4']

    x = np.arange(len(labels))
    for i in totalDefects:
        actualTotal.append(i[15])

    for i in allPredicted:
        predictedTotal.append(i[14])
    width = 0.35
    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width / 2, actualTotal, width, label='Actual Defects')
# matplotlib.pyplot library function to fill plot
```

```
rects2 = ax.bar(x + width / 2, predictedTotal, width, label='Predicted Defects')

ax.set_ylabel('Defects') # Title of Plot
ax.set_title('Actual Defects by group for all country') # Numbers name of plot
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)
fig.tight_layout()
plt.show() # Show Plot
```

Bar Plot for part (d)






Total Of Actual - Predicted Defects
Actual Defects
Predicted Defects
Defects
144
139.044
91
97.3307
32
34.0658
11
7.94868
2
1.39102
0
20
40
60
80
100
120
140
0
1
2
3
4