Statistical Data Analysis

problem sheet 2

sat sun mon tue wed thu fri Date: 04 / 11 / 2022

1 (i) Given that,

$$f(t) = \begin{cases} \frac{1}{2\theta\sqrt{t}} \exp\left(\frac{-\sqrt{t}}{\theta}\right), & \text{for } t > 0 \\ 0 & \text{for } t \not< 0 \end{cases}$$

Now we call $f(t_1, \ldots t_n | \theta)$ as the likelihood function. It always depends on unknown parameter θ and it is denoted as $L(\theta)$.

$$\text{So, } L(\theta) = \prod_{i=1}^{n} f(t_i | \theta) \quad \ldots\ldots (1)$$

$L(\theta)$ is defined as a product of n terms, which is not easy to be maximized. So, maximizing $\log L(\theta)$ because log is a monotonic increasing function. We define $\log L(\theta)$ as "Log likelihood function".

$$l(\theta) = \log L(\theta) = \log \prod_{i=1}^{n} f(t_i | \theta)$$

$$= \sum_{i=1}^{n} \log f(t_i | \theta) \quad \ldots\ldots (ii)$$

Maximizing $l(\theta)$ with respect to θ will give us the MLE estimation.

sat sun mon tue wed thu fri Date: / /

So, the log likelihood function,

$$l(\theta) = \log L(\theta) = \sum_{i=1}^{n} \log f(t_i|\theta) \text{ [using (ii)]}$$

$$= \sum_{i=1}^{n} \log \left\{ \frac{1}{2\theta\sqrt{t}} \exp\left(\frac{-\sqrt{t}}{\theta}\right) \right\}$$

$$= \sum_{i=1}^{n} \left[\log\left(\frac{1}{2\theta\sqrt{t}}\right) + \log\left(\exp\left(\frac{-\sqrt{t}}{\theta}\right)\right) \right]$$

$$= \sum_{i=1}^{n} \left[-\log 2\theta\sqrt{t} - \frac{\sqrt{t}}{\theta} \right]$$

$$= -\sum_{i=1}^{n} \log 2\theta\sqrt{t} - \sum_{i=1}^{n} \frac{\sqrt{t}}{\theta}$$

now, $$l'(\theta) = -\sum_{i=1}^{n} \left\{ \frac{d}{d\theta}(\log 2\theta\sqrt{t}) - \frac{d}{d\theta}\left(\frac{\sqrt{t}}{\theta}\right) \right\}$$

$$= -\sum_{i=1}^{n} \frac{1}{\theta} + \sum_{i=1}^{n} \frac{\sqrt{t}}{\theta^2}$$

$$\left[\frac{d}{dx}(\log x) = \frac{1}{x} \right]$$

$$\Rightarrow 0 = -\frac{n}{\theta} + \frac{1}{\theta^2} \sum_{i=1}^{n} \sqrt{t} \text{ [as derivative in } \theta]$$

$$\Rightarrow 0 = \frac{-n\theta + \sum_{i=1}^{n} \sqrt{t}}{\theta^2}$$

HEART'S

$$\Rightarrow n\theta = \sum_{i=1}^{n} \sqrt{t}$$

$$\therefore \hat{\theta}_{MLE} = \frac{1}{n} \sum_{i=1}^{n} \sqrt{t}$$

This is the maximum likelihood estimate.

Now, for the given sample,

$t_1 = 11300$, $t_2 = 5000$, $t_3 = 4300$, $t_4 = 8500$ and $t_5 = 7900$,

$$\hat{\theta}_{MLE} = \frac{1}{5} \sqrt{11300 + 5000 + 4300 + 8500 + 7900}$$

$$= \frac{1}{5} (\sqrt{37000}$$

$$= 38.471$$

(ii) Method of moments is an estimation method of population parameters.

Now, $\bar{T} = E(T) = 2\theta^2$

$\Rightarrow \theta^2 = \bar{T}/2$

$\therefore \theta = \sqrt{\bar{T}/2}$

(2) Given that,

$$f_\theta(x) = \frac{\theta^x}{x!} e^{-\theta}, \quad x = 0, 1, 2, \ldots$$

Maximum likelihood estimate,

$$\hat{\theta}_{MLE} = E\left(1/n \sum_{i=1}^{n} x\right)$$

$$= \frac{1}{n} E\left(\sum_{i=1}^{n} x\right)$$

$$= \frac{1}{n} \sum_{i=1}^{n} (E(x))$$

$$= \frac{1}{n} \sum_{i=1}^{n} \theta \quad \left[\because \text{it is poisson distribution function, } E(x) = \theta\right]$$

$$= \frac{1}{n} \cdot n\theta$$

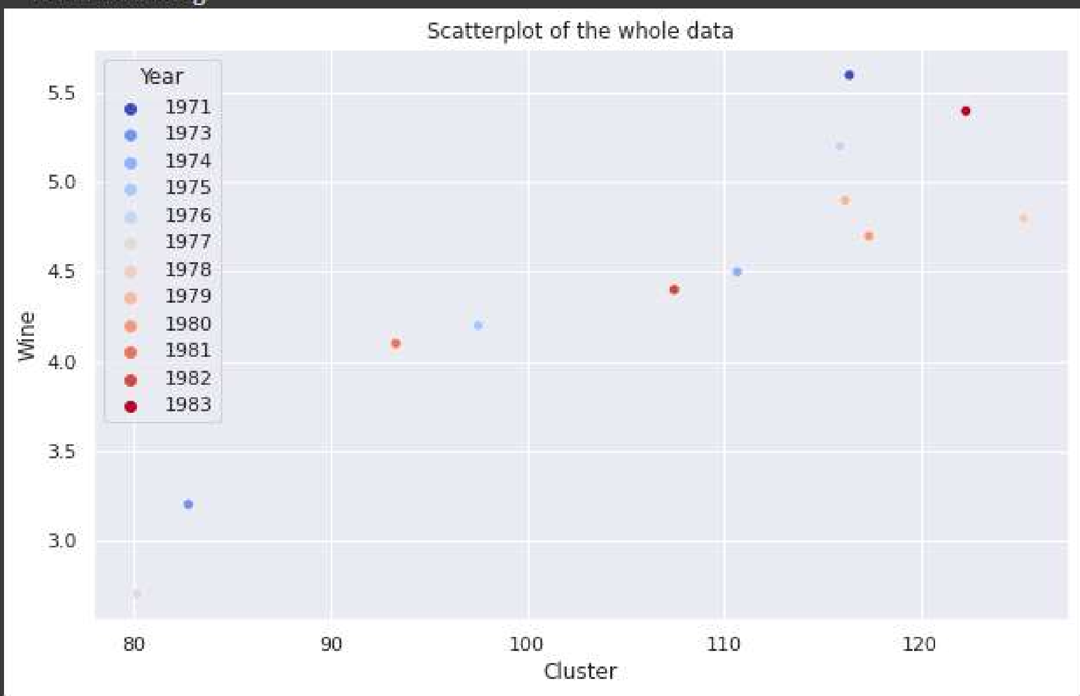
$$= \theta.$$

So, we see, that the estimate is unbiased.

```
[ ]
#setting style
sns.color_palette("Set2")
sns.set_style("whitegrid")
#define figure size
sns.set(rc={"figure.figsize":(10, 6)}) #width=8, height=4
#ploting scatterplot
sns.scatterplot(df["Cluster"],df["Wine"], df["Year"], legend="full", palette="coolwarm")
plt.title("Scatterplot of the whole data")

plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables a
  FutureWarning
```



2. Assume the a simple linear regression model and estimate the parameters β0 and β1 in the regression model (use your own derivation).

```
[ ]
#defining x and y for the regression model
x = df["Cluster"]
y = df["Wine"]

N = len(x)
#deriving a regression function
def lin_reg(x,y):
  x_mean=x.mean()
  y_mean=y.mean()
  x_var = x - x_mean
  y_var = y - y_mean
  #sum of products of x and y variance
  sumof_xy_var = (x_var * y_var).sum()

  sumof_sig_x = ((x_var)**2).sum()
  #now we can estimate b1
  b1 = sumof_xy_var/sumof_sig_x
  #estimating b0
  b0 = y_mean - (b1*x_mean)
  #printing the line as y = mx + b = b1(x) + b0
  simple_reg_lin = 'y = {}β + {}'.format(round(b1, 4),round(b0, 4))

  return(b0, b1, simple_reg_lin)

```

```
[ ]
#plugging in the data into the model
lin_reg(x,y)
```

```
(-1.027902709556801, 0.051380577893637314, 'y = 0.0514β + -1.0279')
```

3. Plot the regression line

To find the regression line we need

- Correlation Coefficient (R)
- Coefficient of Determination (R^2)

We will be using pearson's correlation coefficient here to get the R and then simply square it to get the coefficient of determination which will tell us the goodness of fit.

```
[ ]
#Pearson's correlation coefficient
def Pear_cor_coef(x,y):
  #sum of the products of x and y
  sumof_xy = (x * y).sum()
  x_sum = x.sum()
  y_sum = y.sum()
  #product of summation of x and y
  prod_sum = x_sum*y_sum
  #getting the summation of x and y square
  sumof_x_sq = (x**2).sum()
  sumof_y_sq = (y**2).sum()
  #now estimating R
  r = ((N*sumof_xy)-((x_sum * y_sum))) / (np.sqrt(((N * sumof_x_sq)-(x_sum**2))*((N * sumof_y_sq)-(y_sum**2))))
  return r

R = Pear_cor_coef(x,y)
#now estimating coefficient of determination
coef_det = R**2
```

[ ]

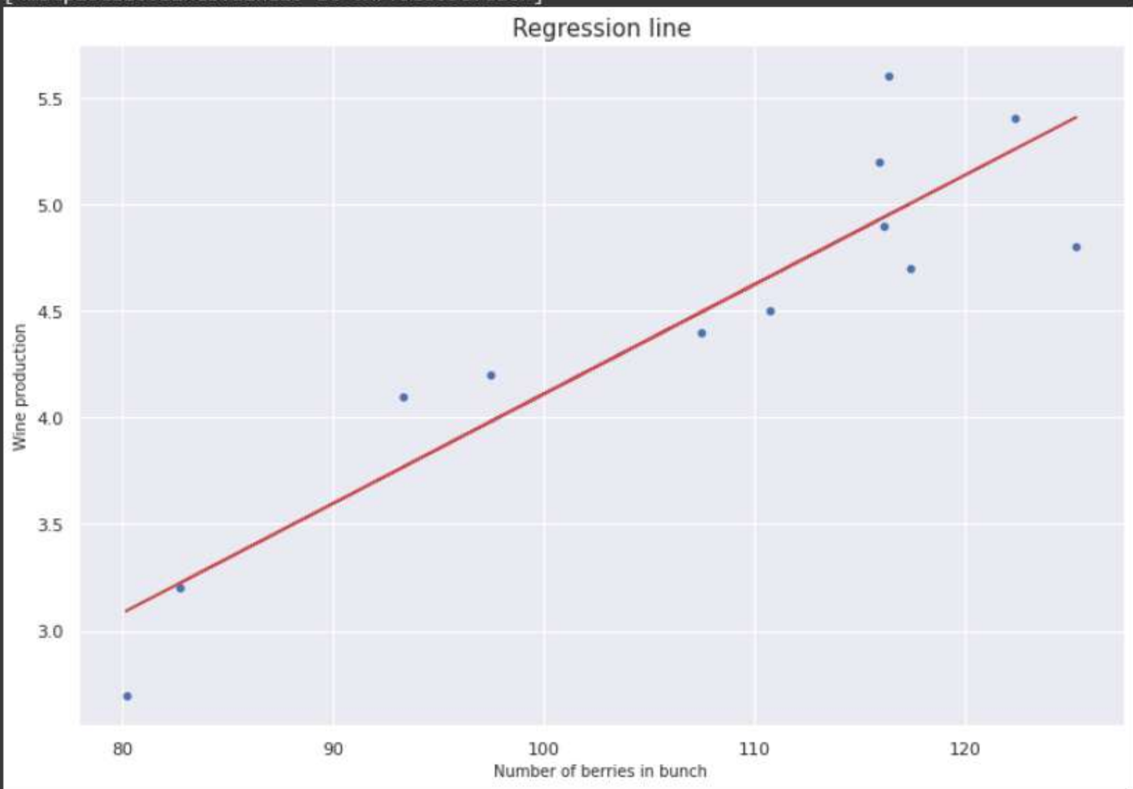
```
#printing the correaltion coefficient and coefficient of determination
print('Correlation Coefficient: ', R)
print('Coefficient of Determination: ', coef_det)
```

```
Correlation Coefficient:  0.913214127416157
Coefficient of Determination:  0.8339600425124531
```

[ ]

```
#now plotting the regression line
plt.figure(figsize=(12,8))
sns.scatterplot(x, y, palette="inferno")
plt.title('Regression line', fontsize=15)
plt.xlabel('Number of berries in bunch', fontsize=10)
plt.ylabel('Wine production', fontsize=10)
b0, b1, regline = lin_reg(x,y)
plt.plot(x, b0 + b1*x, c='r')
```

```
[<matplotlib.lines.Line2D at 0x7f9e539efed0>]
```



4. Predict the yearly production of wine if the number of berries in a bunch of grapes is 100.

```
[ ]
#predicting the value of wine production when the number of berries in a bunch of grapes is 100
newy = b0 + (b1*100)
print("Predicted Wine Production : ", round(newy, 2))
```

```
Predicted Wine Production :  4.11
```

So The Predicted yearly wine production is 4.11 tons per 100 meter square if the number of berries in a bunch of grapes is 100.