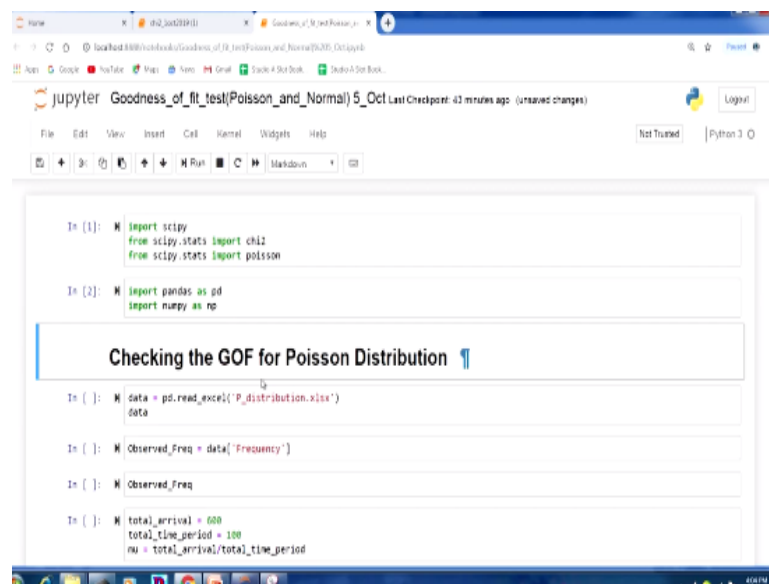


**Data Analytics with Python**  
**Prof. Ramesh Anbandam**  
**Department of Management Studies**  
**Indian Institute of Technology – Roorkee**

**Lecture – 48**  
**Chi Square Goodness of Fit Test**

Welcome students. In the previous lecture I have explained how to do goodness of fit for your Poisson distribution. I have explained the theory portions. In this class I am going to give you a python demo for that after the python demo I am going to explain how to do the goodness of fit for uniform distribution or normal distribution.

**(Refer Slide Time: 00:53)**



```
In [1]: import scipy
        from scipy.stats import chi2
        from scipy.stats import poisson

In [2]: import pandas as pd
        import numpy as np

Checking the GOF for Poisson Distribution

In [ ]: data = pd.read_excel('P_distribution.xlsx')
        data

In [ ]: Observed_Freq = data['frequency']

In [ ]: Observed_Freq

In [ ]: total_arrival = 600
        total_time_period = 100
        rv = total_arrival/total_time_period
```

The agenda for this lecture is python demo for testing goodness of it for the Poisson distribution, theory of how to do the goodness of fit for uniform or normal distribution. After that I will give you the python demo for testing goodness of fit for uniform and normal distribution. Now we will go to the python prompt there we will see how to do goodness of fit for your Poisson distribution.

Now we will see the python demo say some dataset is given we are going to test whether that dataset follow Poisson distribution or not. So I am importing the necessary library like scipy from scipy.stats chi square from scipy.stats Poisson then for I am importing pandas and numpy.

**(Refer Slide Time: 01:43)**

```

In [3]: data = pd.read_excel('P_distribution.xlsx')
data

Out[3]:

```

Arrivals	Frequency
0	0
1	1
2	4
3	10
4	14
5	20
6	12
7	12
8	9
9	8
10	6
11	3
12	1

So checking goodness of fit for your Poisson distribution we will see what is that dataset. This dataset is given arrival is given, the frequency is given. The arrival is in one minute arrival. For example, in one minute interval 0 interval there is 0 frequency. In one minute interval there is 1 arrival there is 1 frequency. In one minute interval there are 2 interval there are 4 frequency and so on.

**(Refer Slide Time: 02:11)**

```

In [4]: Observed_Freq = data['Frequency']

In [5]: Observed_Freq

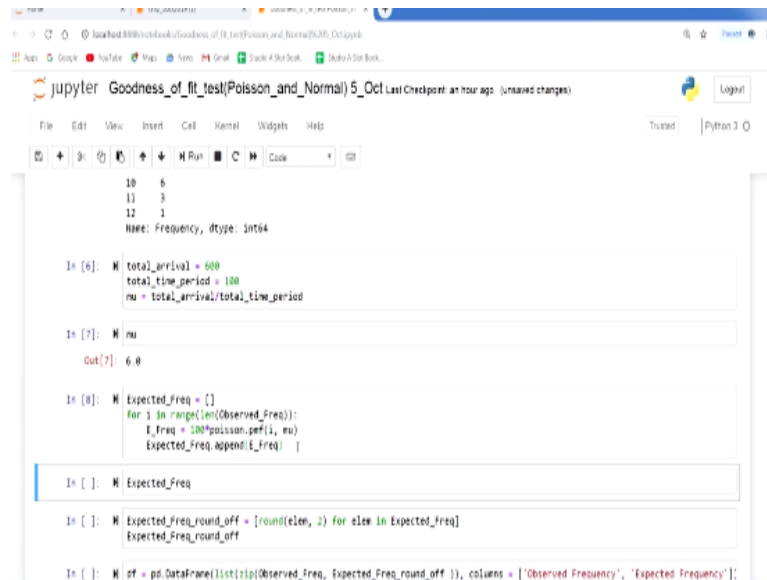
Out[5]:
0      0
1      1
2      4
3     10
4     14
5     20
6     12
7     12
8      9
9      8
10     6
11     3
12     1
Name: Frequency, dtype: int64

In [ ]: total_arrival = 600
total_time_period = 100
mu = total_arrival/total_time_period

```

Whatever the data which is given that is observed frequency. So the observed frequency I am going to call it separately these are our observed frequency. Then next one is we have to find out the expected frequency for given x values. To know the expected frequency, we have to know the mean of the Poisson distribution. We know that mean of the Poisson distribution like  $\sigma f$  and  $n$  divided by  $\sigma f$ . See the total arrival is 600 total time period is 100 so 600 divided by 100 so the  $\mu$  value is your 6.

(Refer Slide Time: 02:58)



```
10 6
11 3
12 1
Name: Frequency, dtype: int64

In [6]: total_arrival = 600
total_time_period = 100
mu = total_arrival/total_time_period

In [7]: mu
Out[7]: 6.0

In [8]: Expected_Freq = []
for i in range(len(Observed_Freq)):
    E_Freq = 100*poisson.pmf(i, mu)
    Expected_Freq.append(E_Freq)

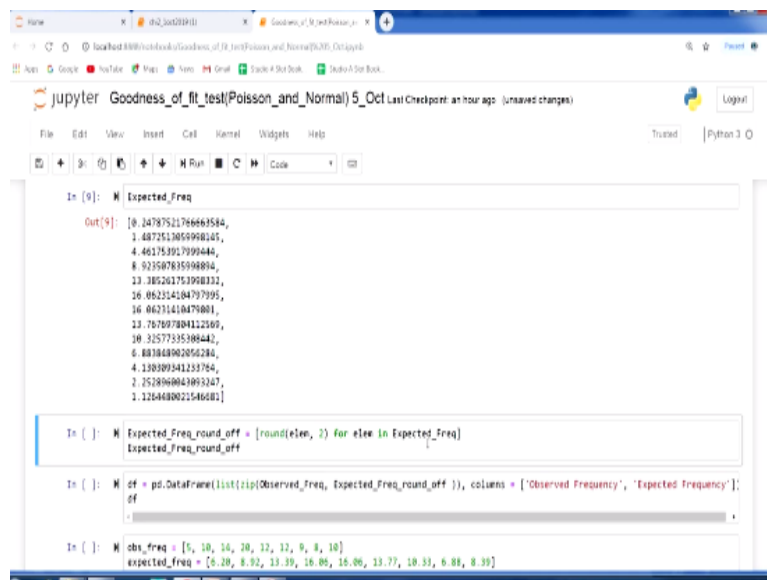
In [ ]: Expected_Freq

In [ ]: Expected_Freq_round_off = [round(elem, 2) for elem in Expected_Freq]
Expected_Freq_round_off

In [ ]: df = pd.DataFrame(list(zip(Observed_Freq, Expected_Freq_round_off)), columns = ['Observed Frequency', 'Expected Frequency'])
```

So we know the mu value, we know the x value now we have to find out the expected frequency. So I am making expected underscore frequency so for i in range of length observed frequency then finding the expected frequency. So E underscore frequency = 100 multiplied by because n is 100 multiplied by Poisson. Pmf probability mass function i, mu then I am going to get the expected frequency. So I am using a for loop so that it will save our time.

(Refer Slide Time: 03:41)



```
In [9]: Expected_Freq
Out[9]: [0.14787511746663544,
1.4872513054998105,
4.461753917999444,
8.923507835998888,
13.385261753998332,
16.062254104797995,
16.062314104798001,
13.767897804112509,
10.3257733530441,
6.00384840205230,
4.13939341233794,
2.3528460043053247,
1.1264080021546601]

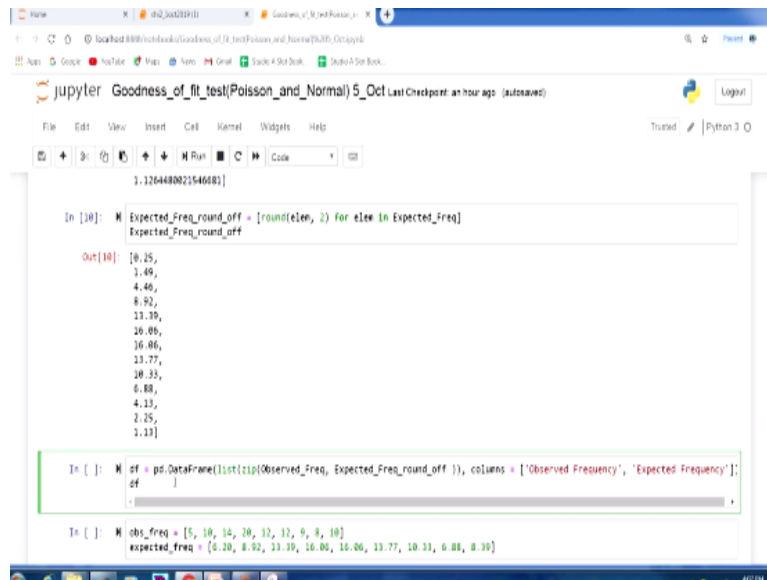
In [ ]: Expected_Freq_round_off = [round(elem, 2) for elem in Expected_Freq]
Expected_Freq_round_off

In [ ]: df = pd.DataFrame(list(zip(Observed_Freq, Expected_Freq_round_off)),
df

In [ ]: Obsv_freq = [5, 10, 10, 20, 32, 32, 9, 8, 10]
expected_freq = [0.20, 0.92, 3.39, 10.00, 16.00, 13.77, 10.33, 6.00, 8.39]
```

So this was our expected frequency. This expected frequency there are different decimal there suppose I want to round it off to 2 decimal for that purpose you have to use this command equal to square bracket round element, 2 for element in expected frequency. So expected frequency rounded off let us see what is that value.

(Refer Slide Time: 04:05)



```
3.1264488021646688]]

In [10]: Expected_Freq_round_off = [round(elem, 2) for elem in Expected_Freq]
Expected_Freq_round_off

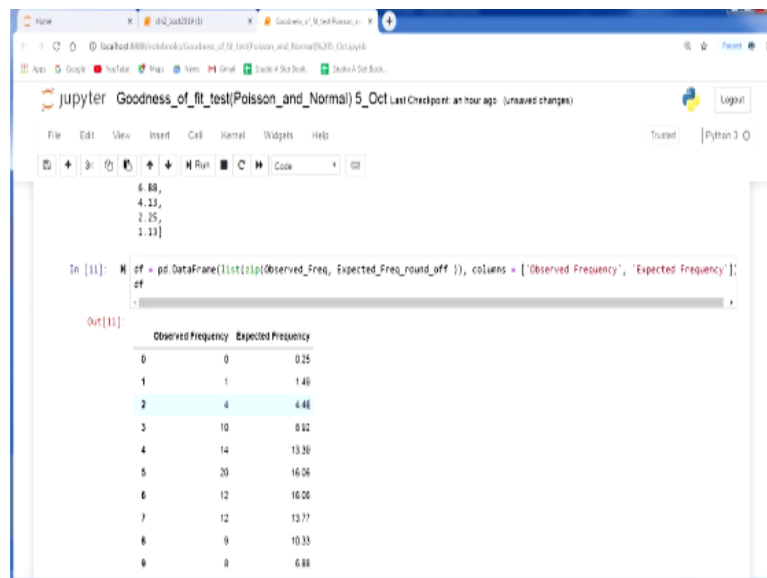
Out[10]: [0.25,
1.49,
4.46,
8.92,
13.39,
16.06,
16.06,
13.77,
10.33,
6.88,
4.13,
2.25,
1.13]
```

```
In [ ]: df = pd.DataFrame(list(zip(Observed_Freq, Expected_Freq_round_off)),
df

In [ ]: obs_freq = [5, 10, 14, 20, 12, 9, 8, 10]
expected_freq = [0.25, 1.49, 4.46, 8.92, 13.39, 16.06, 16.06, 13.77, 10.33, 6.88, 4.13]
```

So this is our expected frequency rounded value. Now we will add these both the variables by using zip command into the object called df.

(Refer Slide Time: 04:19)



```
6.88,
4.13,
2.25,
1.13]
```

```
In [11]: df = pd.DataFrame(list(zip(Observed_Freq, Expected_Freq_round_off)),
df

Out[11]:
```

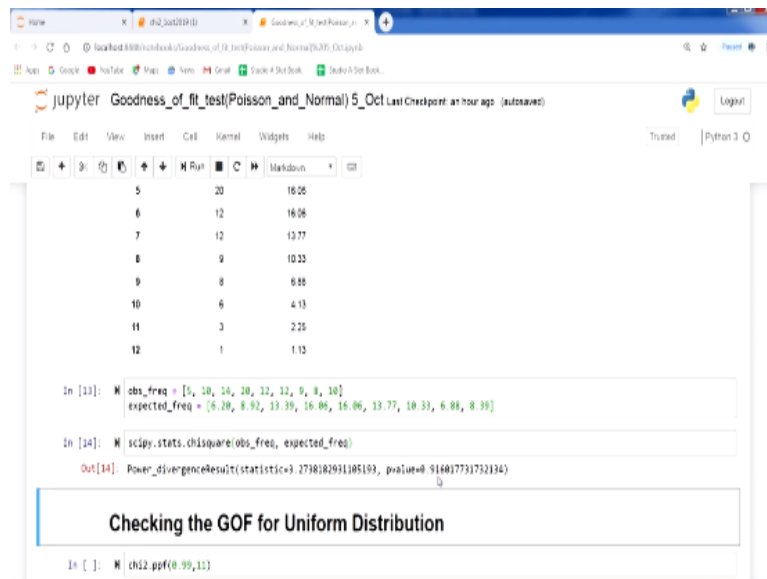
	Observed Frequency	Expected Frequency
0	0	0.25
1	1	1.49
2	4	4.46
3	10	8.92
4	14	13.39
5	20	16.06
6	12	16.06
7	12	13.77
8	9	10.33
9	8	6.88

So the df says that our observed frequency and expected frequency. Once we know the observed frequency and expected frequency then we can get the chi square value directly, but look at here, here the expected frequency is 0.25 that is less than 5. So, up to 3 when you add this 3 then only your expected frequency will be more than 5. So we have added this so after adding so the observed frequency is 415 the expected frequency is 6.20.

This we have done it manually when you are running a large program with huge dataset you can make the program so that the expected frequency is more than 5, but here we have not

done that way. We have manually added and checked whether the expected frequency is more than 5 or not.

**(Refer Slide Time: 05:13)**



The screenshot shows a Jupyter Notebook interface with a table of observed and expected frequencies. Below the table, there are two code cells. The first cell defines the observed and expected frequency arrays. The second cell runs the `scipy.stats.chisquare` function. The output shows the test statistics and p-value.

5	20	16.06
6	12	16.06
7	12	13.77
8	9	10.33
9	8	6.88
10	6	4.13
11	3	2.25
12	1	1.13

```
In [13]: M obs_freq = [5, 10, 14, 20, 12, 12, 9, 8, 10]
         M expected_freq = [6.28, 8.92, 13.39, 16.86, 16.06, 13.77, 10.33, 6.88, 8.39]

In [14]: M scipy.stats.chisquare(obs_freq, expected_freq)

Out[14]: Power_divergenceResult(statistic=3.2736182931105193, pvalue=0.914017731732134)
```

**Checking the GOF for Uniform Distribution**

```
In [ ]: M ch22.ppf(0.99, 11)
```

So this is observed frequency then expected frequency. Now similarly you see that for 10 and 11 and 12 the expected frequency is not more than 5 so we have collapsed these 3 intervals and made into one interval or that is called 10 or more so that is your 8.39 and the interval is 10 or 11 or 12. Now we have the observed frequency and expected frequency. Simply you pass this command that `s scipy.stats.chisquare` observed frequency and expected frequency.

I have to run this. Now I am getting the test statistics chi square statistics 3.27 see the p value is 0.91 it is above our 0.045. So we have to accept our null hypothesis in our presentation also I have told you the p value is 0.91 the calculated chi square value is 3.27. So we have to accept our null hypothesis and we have to conclude that the given dataset follows Poisson distribution.

**(Refer Slide Time: 06:22)**

## Goodness of fit for Uniform Distribution

- Milk Sales Data

Month	Litres
January	1,610
February	1,585
March	1,649
April	1,590
May	1,540
June	1,397
July	1,410
August	1,350
September	1,495
October	1,564
November	1,602
December	<u>1,655</u>
	18,447

We have tested whether the given dataset follow Poisson distribution or not. So Poisson distribution is the discrete distribution. We will see another example where the given dataset follow uniform distribution or not. So the milk sales is given for 12 months January, February, March, April, May, June up to Decembers then liters also given like this So we are going to say that the sales follow uniform distribution or not. So the assumption is the sales follow uniform distribution.

(Refer Slide Time: 06:56)

## Hypotheses and Decision Rules

$H_0$ : The monthly milk figures for milk sales are uniformly distributed

$H_a$ : The monthly milk figures for milk sales are not uniformly distributed

$$\alpha = .01$$

$$df = k - 1 - (p)$$

$$= 12 - 1 - 0$$

$$= \underline{11}$$

$$\chi^2_{.01,11} = \underline{24.725}$$

If  $\chi^2_{\text{cal}} > 24.725$ , reject  $H_0$ .

If  $\chi^2_{\text{cal}} \leq 24.725$ , do not reject  $H_0$ .

So what is a null hypothesis the monthly milk figures for milk sales are uniformly distributed. Alternative hypothesis is the monthly milk figures of milk sales are not uniformly distributed. You see that the not appears in our alternative hypothesis which is not our traditional way of forming the null hypothesis. So we have alpha = 0.01 the k is the given dataset because there are 12 month dataset is there 12.

And here see the p value is 0 because the uniform distribution is not having any parameter because if you know the lower limit and upper limit of uniform distribution you can easily construct the uniform distribution. So for uniform distribution having the 0 parameter after simplifying it is 11 so when  $\alpha = 0.01$  the degrees of freedom is 11, the calculated chi square value is 24.725.

So if we are using the critical value method if the calculated value is greater than not calculated value I am correcting this is the table value, table value which we got from the chi square table 24.725. If the calculated chi square value is greater than 24.725 we have to reject it otherwise we have to accept our null hypothesis.

**(Refer Slide Time: 08:24)**

## Python code

```
In [1]: from scipy.stats import chi2
```

```
In [2]: import pandas as pd  
import numpy as np
```

```
In [3]: chi2.ppf(0.99,11)
```

```
Out[3]: 24.724970311318277
```

We can find out the chi square table value by using this one `chisquare.ppf.0.99` and 1 24.72 see that the same value 24.72.

**(Refer Slide Time: 08:38)**

## Calculations

Month	$f_o$	$f_e$	$(f_o - f_e)^2 / f_e$
January	1,610	1,537.25	3.44
February	1,585	1,537.25	1.48
March	1,649	1,537.25	8.12
April	1,590	1,537.25	1.81
May	1,540	1,537.25	0.00
June	1,397	1,537.25	12.80
July	1,410	1,537.25	10.53
August	1,350	1,537.25	22.81
September	1,495	1,537.25	1.16
October	1,564	1,537.25	0.47
November	1,602	1,537.25	2.73
December	1,655	1,537.25	9.02
	<u>18,447</u>	<u>18,447.00</u>	<u>74.38</u>

$$f_e = \frac{18447}{12}$$

$$= 1537.25$$

$$\chi^2_{cal} = 74.37$$

Now this is the given dataset month is given observed frequency is given. To know the expected frequency what you have to do we have to sum this dataset see that this is the sum value 18,447 then because it is followed uniform distribution you divide by 12. So, equal value is 1,537.25. So everywhere you can write this should be our expected frequency. Then you find out observed frequency minus expected frequency whole square divided by expected frequency. Then you sum it you are getting 74.38. So the calculated chi square value is 74.38. **(Refer Slide Time: 09:32)**

## Python code

```

In [6]: x = [1610, 1585, 1649, 1590, 1540, 1397, 1410, 1350, 1495, 1564, 1602, 1655]

In [7]: np.mean(x)
Out[7]: 1537.25

In [8]: exp_f = [1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25]

In [9]: from scipy.stats import chisquare
chisquare(x, exp_f)

Out[9]: Power_divergenceResult(statistic=74.37583346885673, pvalue=1.78545252783034e-11)
```

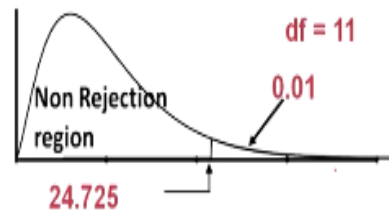
Let us see the python code for this the x is given we are finding the mean 1,537. So expected frequency is nothing but the same value expected frequency this also we have entered manually then from scipy.stats import chi square. So chi square x, expected frequency so, when you give chi square x, expected underscore frequency. You will get the calculated chi



square value is 74.37 the p value is 1.7 into 10 to the power – 11. So you see that this our python outputs and our calculated values are same.

**(Refer Slide Time: 10:19)**

## Conclusion



$$\chi^2_{Cal} = 74.37 > 24.725, \text{ reject } H_0.$$

So obviously this was the table value 24.725 our calculated value is this one. So we have to reject our null hypothesis. When we reject a null hypothesis what we are concluding that the dataset is not following uniform distribution.

**(Refer Slide Time: 10:41)**

## Goodness of Fit Test: Normal Distribution

1. Set up the null and alternative hypotheses.
2. Select a random sample and
  - a. Compute the mean and standard deviation.
  - b. Define intervals of values so that the expected frequency is at least 5 for each interval.
  - c. For each interval record the observed frequencies
3. Compute the expected frequency,  $e_i$ , for each interval.

Now we will go to another very interesting example testing some dataset and checking whether it is following normal distribution or not. So what are the different steps are there. The first step is setup null and alternative hypothesis, select the random sample and compute the mean and standard deviation. Define intervals of values so that the expected frequency is

at least 5 for each interval. For each interval record the observed frequency then compute the expected frequency for each interval.

**(Refer Slide Time: 11:19)**

### Goodness of Fit Test: Normal Distribution

4. Compute the value of the test statistic.

$$\chi^2 = \sum_{i=1}^k \frac{(f_i - e_i)^2}{e_i}$$

5. Reject  $H_0$  if

$$\chi^2 \geq \chi_{\alpha}^2$$

$$k - 1 - 2$$

(where  $\alpha$  is the significance level and there are  $k - 3$  degrees of freedom)

Then compute the value of test statistics then reject  $H_0$  if the chi square value is greater than your value which you got from the table. Here the alpha is significance level here the degrees of freedom is  $k - 3$  because we know that this is  $K - 1 - P$ . So here  $P$  is 2 because there are 2 parameter for a normal distribution so the value of  $P = 2$  that is why we have got  $k - 3$ .

**(Refer Slide Time: 11:53)**

### Normal Distribution Goodness of Fit Test

- Example: IQL Computers

IQL Computers manufactures and sells a general purpose microcomputer. As part of a study to evaluate sales personnel, management wants to determine, at  $\alpha = 0.05$  significance level, if the annual sales volume (number of units sold by a salesperson) follows a normal probability distribution.

We will take an example see computer manufacturers and sells a general purpose microcomputer. As part of a study to evaluate the sales personnel management wants to determine that  $\alpha = 5\%$  significance level if the annual sales volume that is the number of units sold by the sales person follows normal probability distribution. So there are some

dataset that dataset is nothing, but the number of units sold by the sales people. They want to test whether that sales follow normal distribution or not.

**(Refer Slide Time: 12:30)**

## Normal Distribution Goodness of Fit Test

A simple random sample of 30 of the salespeople was taken and their numbers of units sold are below.

33 43 44 45 52 52 56 58 63 64  
64 65 66 68 70 72 73 73 74 75  
83 84 85 86 91 92 94 98 102 105

(mean = 71, standard deviation = 18.23)

A simple random sample of 30 of the salespeople who has taken and their number of units sold are given below. 33, 43, 44 and so on. So for this dataset the mean is 71, standard deviation is 18.23.

**(Refer Slide Time: 12:49)**

## Python code

```
In [12]: A = [33, 43, 44, 45, 52, 52, 56, 58, 63, 64, 64, 65, 66, 68, 70, 72, 73, 73, 74, 75, 83, 84, 85, 86, 91, 92, 94, 98, 102, 105]

In [13]: mean = np.mean(A)
         mean
Out[13]: 71.0

In [14]: std = np.std(A)
         std
Out[14]: 18.226754544898825
```

So we have imported the data so we are finding the mean is 71, standard deviation is 18.22.

**(Refer Slide Time: 12:57)**

## Normal Distribution Goodness of Fit Test

- Hypotheses

$H_0$ : The population of number of units sold has a normal distribution with mean 71 and standard deviation 18.23

$H_a$ : The population of number of units sold does not have a normal distribution with mean 71 and standard deviation 18.23

For this dataset what is the null hypothesis. The population of number of units sold has a normal distribution with mean 71 and standard deviation is 18.23. The alternative hypothesis is the population of number of unit sold does not have a normal distribution with mean 71 and standard deviation 18.23 because many times whenever you collect the data we have to test what distribution it follows.

Because when you do the simulation rather purpose we have to knowing the exact distribution of given dataset is more important that is why testing the particular distribution will be very helpful for further analysis of our dataset.

**(Refer Slide Time: 13:47)**

## Normal Distribution Goodness of Fit Test

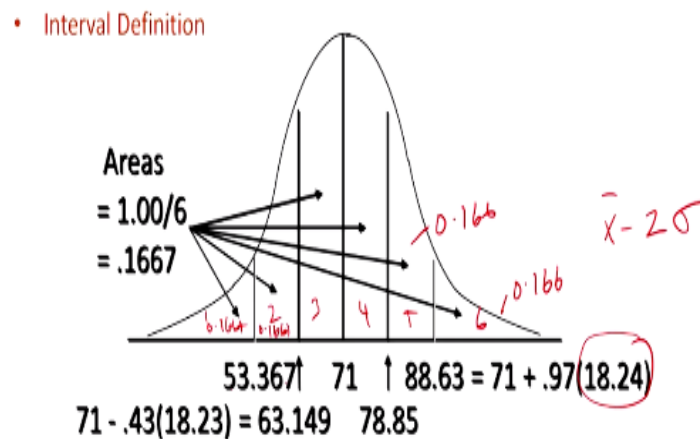
- Interval Definition

To satisfy the requirement of an expected frequency of at least 5 in each interval we will divide the normal distribution into  $30/5 = 6$  equal probability intervals.

First we make an interval to satisfy the requirement of an expected frequency of at least 5 this is very important at least 5 in each interval we will divide the normal distribution by 5 so 30 divided by 5 so that we will get 6 equal probability interval.

**(Refer Slide Time: 14:06)**

### Normal Distribution Goodness of Fit Test



You see that 1, 2, 3, 4, 5, 6. So when you divide by 5 we will get 6 interval because then when you divide this way then we can make sure that in every interval you will get minimum 5 or more observed value. So total area = 1 so when you divide 1 divided by 6 we will get 0.1667 is because this area is 0.1667 this area is 0.1667 everywhere. So when the area is 0.167 you can get the z value. We know that their lower limit  $\bar{X} - Z\sigma$ .

So  $\bar{X}$  is given Z value you can get it sigma value also given you can find out this interval so this is 53.367 the second point is we know  $\bar{X}$  is 71 how we got the 71 because the mean is 71 right. The mean is 71 the sigma is given, sigma is 18.23 when it is 0.66 + 0.66 the corresponding Z value is 0.43 so 0.43 and this sigma value that is your 16.17. Similarly, at the right hand side see that how we got 88.63.

The mean value is given so when this side area is 0.166 this area is 0.166 when you add that then corresponding Z value is 0.97. So 71 + 0.97 this was our sample standard deviation the sigma value then you will get 88.63. What we got it we got different intervals.

**(Refer Slide Time: 16:01)**

## Python code

```
In [15]: x = 1/6 #for 6 equal probability intervals.

In [16]: for j in range(1,6):
          Prob_intervals = [scipy.stats.norm.ppf(j*x, mean, std)]
          print ( Prob_intervals)
```

```
[53.36743154175236]
[63.14941153083116]
[71.0]
[78.85058846916884]
[88.63256845824763]
```

So what are that intervals you see that 1 divided by 6 so that we will get different 6 equals probability intervals. So for j in range 1, 6 the probability underscore interval is scipy.stat.norm.ppf we can substitute j value directly here then x, mean in standard deviation. So when you print this probability interval you are getting this is your different intervals when we got after solving manually you see 53.67, the second value is 63.149 this was our interval of the normal distribution.

**(Refer Slide Time: 16:43)**

## Normal Distribution Goodness of Fit Test

- Observed and Expected Frequencies

$i$	$f_i$	$e_i$	$f_i - e_i$
Less than <u>53.02</u>	⑥	⑤	1
53.02 to <u>63.03</u>	③	5	-2
63.03 to <u>71.00</u>	⑥	5	1
71.00 to <u>78.97</u>	5	5	0
78.97 to <u>88.98</u>	4	5	-1
More than <u>88.98</u>	6	5	1
Total	30	30	

$$\frac{30}{6} = 5$$

See that the first value is 0 to 53.36 see that was this value the second one is 53.02, 63.03, 63.14 then 71, 78, 88 the maximum value is 88.63. So what we have to do now we have to go to this values in that interval we have to count it how many numbers are appearing. For example, in the interval you see that when the interval less than 53.02 this 6 was observed frequency.

So how we got the 6 one from this given dataset you have to find out how many numbers are below that when you count it, it will be 6. Similarly, in the interval 53.02, 63.03 in our given dataset we have to count it how many numbers are appearing this range there will be a 3 this is our observed frequency. This is 5 is expected frequency because there was a 30 dataset since we divided 30 divided by 6 there will be a 5 expected frequency will be there.

Because why we are dividing by 5 because minimum we need to have 5 expected frequency that is why we have divided by 6 so we have to divide the given dataset by a number so that the expected frequency is 5. So we got this observed then expected frequency then find the different square the difference.

**(Refer Slide Time: 18:26)**

## Python code

```
In [17]: Expected_Freq = [5,5,5,5,5,5] #will divide the normal distribution into 6 intervals at frequency 5 in each
```

```
In [18]: Obs_f = [6,3,6,5,4,6]
```

```
In [19]: scipy.stats.chisquare(Obs_f, Expected_Freq)
```

```
Out[19]: Power_divergenceResult(statistic=1.5999999999999999, pvalue=0.9012493445012737)
```

$\alpha = 5\%$

This was our expected frequency this is our observed frequency. So `scipy.stat.chisquare` observed frequency, expected frequency. We are getting 1.5 see the p value is 0.90 that is we say that 5% it is more than that. We say we have to accept our null hypothesis so the given dataset follow normal distribution.

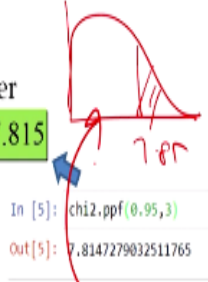
**(Refer Slide Time: 18:50)**

## Normal Distribution Goodness of Fit Test

- **Rejection Rule**

With  $\alpha = .05$  and  $k - p - 1 = 6 - 2 - 1 = 3$  d.f.  
 (where  $k$  = number of categories and  $p$  = number  
 of population parameters estimated),  $\chi^2_{.05} = 7.815$

Reject  $H_0$  if  $p\text{-value} \leq .05$  or  $\chi^2 \geq 7.815$ .



- **Test Statistic**

$$\chi^2 = \frac{(1)^2}{5} + \frac{(-2)^2}{5} + \frac{(1)^2}{5} + \frac{(0)^2}{5} + \frac{(-1)^2}{5} + \frac{(1)^2}{5} = 1.600$$

When you look at that one alpha = 0.05 there are k = 6. How we get got is 6 number of intervals I am going back 1, 2, 3, 4, 5, 6 that is why it is k = 6. P is a 2 parameter n so 6 – 2 – 1 is the 3 degrees of freedom. When 3 degrees of freedom when alpha = 5% the table value is 7.8815 you look at this one our test statistic observed frequency minus expected frequency whole square divided by expected frequency when we add it is 1.6.

So what is happening in chi square distribution so this value is 7.815 our calculated value we are getting 1.6 so we have to accept our null hypothesis and we have to conclude that the given dataset follow normal distribution. Now I will explain the python code of testing how to check whether the given dataset follow uniform distribution and normal distribution.

**(Refer Slide Time: 20:01)**

```

In [15]: chi2.ppf(0.95, 3)
Out[15]: 7.8147279032511765

In [16]: x = [1630, 1585, 1645, 1590, 1540, 1397, 1410, 1350, 1495, 1544, 1602, 1655]

In [17]: np.mean(x)
Out[17]: 1537.25

In [18]: exp_f = [1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25, 1537.25]

In [19]: from scipy.stats import chisquare
chisquare(x, exp_f)
Out[19]: Power_divergenceResult(statistic=74.37588346886675, pvalue=1.78545251783894e-11)
    
```

**Checking the GOF for Normal Distribution**

```

In [ ]: x = [32, 43, 44, 45, 51, 51, 56, 58, 63, 64, 64, 65, 66, 68, 70, 71, 73, 73, 74, 75, 83, 84, 85, 86, 91, 92, 94, 98, 100, 100]
    
```



Suppose we want to know the chi square table value when alpha equal to 1 percentage when it is alpha 1 percentage we have to write `chisquare.ppf(0.99, 11)` degrees of freedom. So we are getting the calculate the table value of chi square value is 24.72. Then next one this is our x value then we are finding the mean of that one. So the expected frequency is nothing but it is going to be our mean.

Now we got the observed frequency that is our 1, 610 that is x value than expected frequency is this one exp underscore f. So when you write chi square x, expected underscore f this is our expected frequency. So now we are getting look at the p value. The p value is  $1.7 \times 10^{-11}$  it is very low value. So we have to reject our null hypothesis. You can look at this our calculated chi square value is 74.

The table chi square value is 24 so 74 is larger than 24 so we have to reject our null hypothesis then we are concluding that the given dataset is not following uniform distribution.

**(Refer Slide Time: 21:26)**

```

In [21]: mean = np.mean(A)
         mean
Out[21]: 71.0

In [22]: std = np.std(A)
         std
Out[22]: 18.226354544998845

In [23]: k = 1/6 #for 6 equal probability intervals.

In [24]: for j in range(1,6):
         Prob_intervals = [scipy.stats.norm.ppf(j*k, mean, std)]
         print ( Prob_intervals)

[5]: [39.743154275236]
[6]: [49.41153083116]
[7]: [71.0]
[8]: [85.65840916884]
[9]: [98.6325604524763]

In [ ]: Expected_Freq = [5,5,5,5,5] #will divide the normal distribution into 6 intervals at frequency 5 in each

```

After that you will see the second example there are some dataset is given we are going to test whether this dataset follow normal distribution or not. So what I am running this dataset. First I am finding the mean the mean of the given dataset is 71 and the standard deviation of the given dataset is 18.23. So what is a null hypothesis the given dataset whose mean is 71 and standard deviation 18.22 follow normal distribution.

Alternative hypothesis is it is not following normal distribution then the given dataset the  $x$  value we have to divide by 6 because we need to have minimum 5 expected frequency not observed frequency, expected frequency in each interval. So the given dataset is divided into 6 so that because 30 divided by 6 I will get 5 expected frequency in each interval. Now in the range of 1, 6 I am going to get the different intervals. So one interval is 0 to 53.36 another interval is 53.36 to 63.14 another interval is 63.14 to 71 next one 71 to 78.85 the next interval is 78.85 and above. So this was our interval.

**(Refer Slide Time: 22:54)**

```

Out[22]: 18.22635454698845

In [23]: N = 176 #for 6 equal probability intervals.

In [24]: N
for j in range(1,6):
    Prob_intervals = [scipy.stats.norm.ppf(j)*x, mean, std]
    print ( Prob_intervals)

[53.36743154175236]
[63.14941157081116]
[71.0]
[78.85058846916884]
[88.63258845824793]

In [25]: Expected_Freq = [5,5,5,5,5] #will divide the normal distribution into 6 intervals of frequency 5 in each

In [ ]: Obs_f = [0,3,6,5,4,6]

In [ ]: scipy.stats.chisquare(Obs_f, Expected_Freq)

```

So, how we got this interval now we got the different intervals. Our expected frequency is 555 I am running this because we have divided by 6 and our observed frequency. So from the given dataset in the range of 0 to 53.67 we have to count it how many dataset is there, there are 6 dataset. In the interval 53.36 to 63.14 there are 3 dataset. In the interval 63.14 to 71 there are 6 dataset and so on. We have to manually count it how many dataset is appearing in this interval that is our observed frequency. So now we got the expected frequency and observed frequency.

**(Refer Slide Time: 23:39)**

```

In [24]: for j in range(1,6):
        Prob_Intervals = [scipy.stats.norm.ppf(j**x, mean, std)]
        print ( Prob_Intervals)

[53.367433541752186]
[63.149413530881116]
[71.0]
[78.85058846918884]
[88.6325684524763]

In [25]: Expected_Freq = [5,5,5,5,5,5] #will divide the normal distribution into 6 intervals of frequency 5 in each

In [26]: Obs_f = [6,3,6,5,4,6]

In [27]: scipy.stats.chisquare(Obs_f, Expected_Freq)

Out[27]: Power_divergenceResult(statistic=1.5999999999999999, pvalue=0.9012493405012737)

In [ ]:

```

Then when we do the chi square test so we are getting p value 0.90 that is bigger than our alpha value. So we have to accept null hypothesis and we are concluding that the given dataset follow normal distribution. In this lecture, first I have explained a demo for testing goodness of fit for a Poisson distribution then I have explained the theory behind how to test goodness of fit for a uniform distribution normal distribution that is some dataset is given.

How to test the given dataset follow uniform distribution or normal distribution then I have done the python code with the help of python demo I have explained how to test the uniform and normal distribution for goodness of fit because the testing another important point you have to remember all the random numbers follow uniform distribution. Sometime you may ask to say some random numbers.

And you have to test whether the numbers are really random or not. For that purpose, we have to test whether the given dataset it is following uniform distribution or not. If certain numbers following uniform distribution we can conclude that, that numbers are random numbers.