Analytics & Statistics using Python and Numerical Methods

PG-DHPCAP

Vijay Barai

Session 12 : Statistics

- Basics of Statistics
- Statistical Analytics
- Descriptive Statistical Measures
- Statistics - Central Tendency & Dispersion (Mean, Median, Mode, Quartiles, Percentiles, Range, Interquartile Range, Standard Deviation, Variance, and Coefficient of Variation)

Lab Assignments:

- Load any dataset and find out the mean, median, mode and other central tendencies of the dataset

What Are Statistics ?

- In generally one can say that statistics is the methodology for collecting, analyzing, interpreting and drawing conclusions from information
- Putting it in other words, statistics is the methodology which scientists and mathematicians have developed for interpreting and drawing conclusions from collected data

Statistics consists of a body of methods for collecting and analyzing data. (Agresti & Finlay, 1997)

Basic Concepts

- Descriptive Statistics – describe a data
- Inferential Statistics – AI/ML
- Independent and dependent variables - Tablets (Independent) -> disease cure measure (dependent)
- Percentiles
- Levels of Measurement – Time, cm, very much; somewhat; low; bad, 10-20;30-40, ¼; ¾,
- Distributions – frequency, probability, skew distribution, continuous/discrete variable

Summation Notation

Grapes	X
1	4.6
2	5.1
3	4.9
4	4.4

$$\sum_{i=1}^{4} X_i$$

$\sum_{i=1}^{4} X_i = X_1 + X_2 + X_3 + X_4$= 4.6 + 5.1 + 4.9 + 4.4 = 19

Statistical Analysis

- Descriptive Analysis - Involves collecting, interpreting, analyzing, and summarizing data to present them in the form of charts, graphs, and tables. Rather than drawing conclusions, it simply makes the complex data easy to read and understand.
- Inferential Analysis- Focuses on drawing meaningful conclusions on the basis of the sample data analyzed. It studies the relationship between different variables or makes predictions for the whole population.
- Predictive Analysis - Type of statistical analysis that analyzes data to derive past trends and predict future events on the basis of them. It uses machine learning algorithms, data mining, data modelling, and artificial intelligence to conduct the statistical analysis of data.
- Prescriptive Analysis - Analysis conducts the analysis of data and prescribes the best course of action based on the results. It is a type of statistical analysis that helps you make an informed decision.
- Exploratory Data Analysis- Similar to inferential analysis, but the difference is that it involves exploring the unknown data associations. It analyzes the potential relationships within the data.
- Causal Analysis- Focuses on determining the cause and effect relationship between different variables within the raw data. In simple words, it determines why something happens and its effect on other variables. This methodology can be used by businesses to determine the reason for failure.

What Is Descriptive Statistics?

- Descriptive statistics is a means of describing features of a data set by generating summaries about data samples. It's often depicted as a summary of data shown that explains the contents of data. For example, a population census may include descriptive statistics regarding the ratio of men and women in a specific city.
- Example : high, low, mean max, average, win loss.

Statistics - *What is Central Tendency?*

- One definition of central tendency is the point at which the distribution is balance.

  - Mean
  - Median
  - Mode
  - Quartiles
  - Percentiles

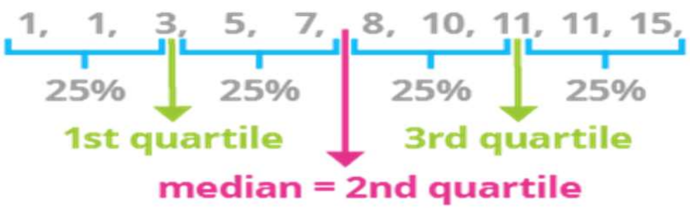
  - Range
  - Interquartile Range
  - Standard Deviation
  - Variance
  - Coefficient of Variation

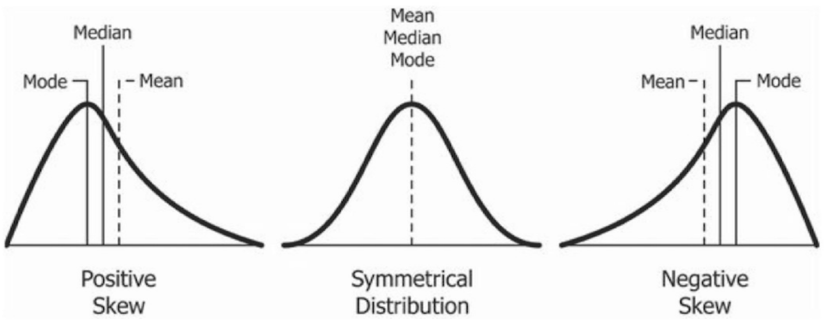
Statistics

- Mean - Sum of the numbers divided by the number of numbers

$$\mu = \frac{\sum x}{N}$$

- Median - midpoint of a distribution after sorting number
  - Odd Count : 2, 4, 7 => 4
  - Even Count: 2, <u>4, 7</u>, 12 => (4+7)/2 = 5.5
- Mode - most frequently occurring value
  - 37, 33, 33, 32, 29, 28, 28, <u>18,18,18,18</u>,16,15,6,3,4,5
- Quartiles





Statistics

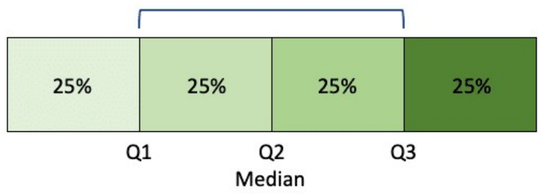
- Percentiles
  - Want to find 25th Percentiles
  - $R = \frac{P}{100} * (N + 1)$
  - $R = \frac{25}{100} * (8 + 1) = \frac{9}{4} = 2.25$
  - Integer=IR=2
  - Fraction=FR=0.25
  - Find number with IR & IR+1 rank = 5,7
  - Percentiles = 0.25*(7-5) = 5.5

Number	Rank
3	1
5	2
7	3
8	4
9	5
11	6
13	7
15	8

Statistics

- **Range** – Number falling between to boundary conditions
- **Interquartile Range** - The interquartile range (IQR) is the range of the middle 50% of the scores in a distribution.

IQR = 75th percentile - 25th percentile



Statistics

- Variance - average squared difference of the scores from the mean.

$$\sigma^2 = \frac{\sum (X-\mu)^2}{N} = 1.5$$

where σ^2 is variance, μ is mean, and N is number of numbers

- Standard Deviation - square root of the variance (σ)
- Coefficient of Variation – Standard Deviation divided by mean

$$CV = \frac{\sigma}{\mu}$$

Scores	Deviation from Mean	Squared Deviation
9	2	4
9	2	4
9	2	4
8	1	1
8	1	1
8	1	1
8	1	1
7	0	0
7	0	0
7	0	0
7	0	0
7	0	0
6	-1	1
6	-1	1
6	-1	1
6	-1	1
6	-1	1
6	-1	1
5	-2	4
5	-2	4
Mean		
7	0	**1.5**

Lab (Python)

```
data = [9,9,9,8,8,8,8,7,7,7,7,7,6,6,6,6,6,6,5,5]

mean = sum(data) / len(data)
deviation_from_Mean = sum((v - mean) for v in data)
squared_deviation= sum((v - mean) ** 2 for v in data)
variance = sum((v - mean) ** 2 for v in data) / len(data)
standard_deviation=variance**0.5

print("Mean : ", mean)
print("Deviation_from_Mean : ", deviation_from_Mean)
print("Squared_Deviation : ", squared_deviation)
print("variance : ", variance)
print("standard_deviation : ", standard_deviation)

# ------------------------------------------------------------

import numpy as np
print("Mean : ", np.mean(data))
print("variance : ", np.var(data))
print("standard_deviation : ", np.std(data))
```

Session 13 : Probability and Distribution

Basics of Probability

- $\text{Probabolity} = \frac{\text{Number of favourable outcomes}}{\text{Number of possible equally} - \text{likely outcomes}}$
- Probability of A and B *(independent event)*

  `P(A and B) = P(A) x P(B)`

  Example : flip a coin twice, what is probability heads come up both times.

- Probability of A or B *(independent event)*

  `P(A or B) = P(A) + P(B) - P(A and B)`

  Example 1: If you throw a six-sided die and then flip a coin, what is the probability that you will get either a 6 on the die or a head on the coin flip (or both)?

  - *A occurs and B does not occur*
  - *B occurs and A does not occur*
  - *Both A and B occur*

  `P(6 or head)=P(6)+P(head)-P(6 and head)=(1/6)+(1/2)-(1/6)(1/2)=7/12`

Basics of Probability

- Conditional Probabilities (not independent event, posterior):

  Example : what is the probability that two cards drawn at random from a deck of playing cards will both be aces?

  ✓ These are not independent events

  ✓ 1^{st} aces drawn from 52 cards (4 was there)

  ✓ 2^{nd} aces drawn from 51 cars (3 was there)

  ```
  P(ace on second draw | an ace on the first draw)= P(B|A)
  P(A and B) = P(A) x P(B|A)
             = 4/52 x 3/51 = 1/221
  ```

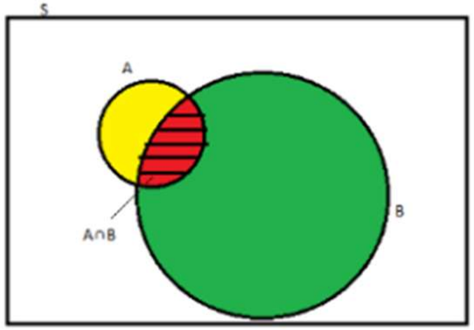
- Join Probabilities (independent event, posterior):
  - Conditional Probability of A given B - $P(A|B) = P(A \cap B) / P(B)$
  - Conditional Probability of B given A - $P(B|A) = P(B \cap A) / P(A)$

  Example : What is the probability that a student is absent given that today is Friday?

Basics of Probability

- **Marginal probability :** Marginal probability is the probability of an event happening, such as (p(A)), and it can be mentioned as an unconditional probability. It does not depend on the occurrence of another event.

Example : The likelihood that a card is drawn from a deck of cards is black (P(black) = 0.5), and the probability that a card is drawn is 7 (P(7)=1/13), both are independent events since the outcome of another event does not condition the result of one event.



Basics of Probability

- Permutations – sequence is important

$$_{\mathrm{n}}\mathrm{P}_{\mathrm{r}} = \frac{n!}{(n-r)!}$$

with 4 available option selecting 2 option then

$$_{4}\mathrm{P}_{2} = \frac{4!}{(4-2)!} = \frac{4*3*2*1}{2*1} = 12\ unique\ set$$

- Combinations – total unique set important

$$_{\mathrm{n}}\mathrm{C}_{\mathrm{r}} = \frac{n!}{(n-r)!r!}$$

$$_{4}\mathrm{C}_{2} = \frac{4!}{(4-2)!2!} = \frac{4*3*2*1}{(2*1)(2*1)} = 6\ unique\ set$$

Bayes' Theorem

- Bayes' theorem considers both the prior probability of an event and the diagnostic value of a test to determine the posterior probability of the event.

$$\mathrm{P(D|T)} = \frac{P(T|D)P(D)}{P(T|D)P(D) + P(T|D')P(D')}$$

Where

- P(D|T) is the posterior probability of condition D given test result T,
- P(T|D) is the conditional probability of T given D,
- P(D) is the prior probability of D,
- P(T|D') is the conditional probability of T given not D,
- P(D') is the probability of not D.

Bayes' Theorem

- Medical Example : Let's say there is a Disease X affecting 2% of the people. What is probability that you have the disease given that you test positive, provided diagnostic test is 99% accurate if has a disease and 91% if you do not have disease.
  - Event D => you have Disease X = $P(D)$ = 0.02
    you do not have Disease X = $P(D')$ = 1- $P(D)$ = 0.98
  - Event T => test is positive
    Test positive and you have disease = P(T|D) = 0.99
    Test negative and you do not have disease = P(T'|D') = 0.91
    Test positive and you do not have disease = P(T|D') = 1- 0.91 = 0.09

$$\mathrm{P(D|T)} = \frac{P(T|D)P(D)}{P(T|D)P(D)+P(T|D')P(D')} = \frac{0.99*0.02}{0.99*0.02+0.09*0.98} = 0.1833 = 18\%$$

Naive Bayes Algorithm(Supervised)

- It is based on “Bayes theorem” but assume that feature is independent of other features.
- Distinction between “Bayes theorem” and “Naive Bayes is that Naive Bayes assumes conditional independence where Bayes theorem does not.
- Formula : $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

Where P(A) and P(B) are two independent events.
P(A|B): is the conditional probability of an event A given B is true.
P(B|A): is the conditional probability of an event B given A is true.
P(A) and P(B): are the probabilities of A and B independently of one another.

Naive Bayes Algorithm

- Naive Bayes classification algorithm is a probabilistic classifier
- Assumption :
  - strong independence of feature
  - all the predictors have an equal effect on the outcome
- When to use?
  - For very well-separated categories,
  - very high-dimensional data
  - when model complexity is less important
- Real-life applications
  - text classification – News topic
  - Spam filtering – Yes/No
  - Weather prediction
  - Sentiment Analysis

Naive Bayes - Mathematical calculations

Weather	Play
Sunny	No
Sunny	No
Overcast	Yes
Rainy	yes
Rainy	yes
Rainy	no
Overcast	yes
sunny	no
sunny	yes
rainy	yes
sunny	yes
Overcast	yes
Overcast	yes
Rainy	no

probabilities :

Weather	No	Yes	
Overcast	0	4	4/14 = 0.29
Sunny	2	3	5/14 = 0.36
Rainy	3	2	5/14 = 0.36
Total	5	9	
	5/14 = 0.36	9/14 = 0.64	

posterior probability

Weather	No	Yes	Posterior probability of No	Posterior probability of Yes
Overcast	0	4	0/5 = 0	4/9 = 0.44
Sunny	2	3	2/5 = 0.4	3/9 = 033
Rainy	3	2	3/5 = 0.5	2/9 = 0.22
Total	5	9		

Frequency Table

Weather	No	Yes
Overcast	0	4
Sunny	2	3
Rainy	3	2
Total	5	9

P(Playing) = P(Yes|Overcast)=P(Overcast|Yes)P(Yes)/ P(Overcast)

P(Overcast) = 4/14 = 0.29

P(Yes)= 9/14 = 0.64

P(Overcast | Yes) = 4/9 = 0.44

P(Yes|Overcast) = 0.44 * 0.64 / 0.29 = 0.98

Naive Bayes - Lab

```
from sklearn import datasets
dataset = datasets.load_wine() # load dataset
print ("Inputs: ", dataset.feature_names) # print the names of the 13 features
print ("Outputs: ", dataset.target_names) # print the label type of wine
print(dataset.data[0:3]) # print the wine data features
print(dataset.target) # print the wine labels
from sklearn.model_selection import train_test_split # import train_test_split function
inputs = dataset.data # input and outputs
outputs = dataset.target
X_train, X_test, y_train, y_test = train_test_split(inputs, outputs, test_size=0.3, random_state=1) # split dataset into training set and test set
from sklearn.naive_bayes import GaussianNB # import Gaussian Naive Bayes model
classifer = GaussianNB() # create a Gaussian Classifier
classifer.fit(X_train, y_train) # train the model using the training sets
y_pred = classifer.predict(X_test) # predict the response for test dataset
from sklearn import metrics # import scikit-learn metrics module for accuracy calculation
print("Accuracy:", metrics.accuracy_score(y_test, y_pred)) # printing accuracy

import seaborn as sns
from sklearn.metrics import confusion_matrix # importing the required modules
cm = confusion_matrix(y_test, y_pred) # passing actual and predicted values
sns.heatmap(cm, annot=True)
```

Random Variable

- Random sample selected from the population for analysis is called Random Variable.
- It is demoted by capital letter like X, Y

Probability Distributions

- **For discrete random variable :** A discrete random variable X has a countable number of possible values. The probability distribution of X lists the values and their probabilities

Value of X	x_1 x_2 x_3 . . . x_k
Probability	$P(x_1)$ $P(x_2)$ $P(x_3)$ . . . $P(X_k)$

The probabilities $P(x_i)$ must satisfy two requirements:

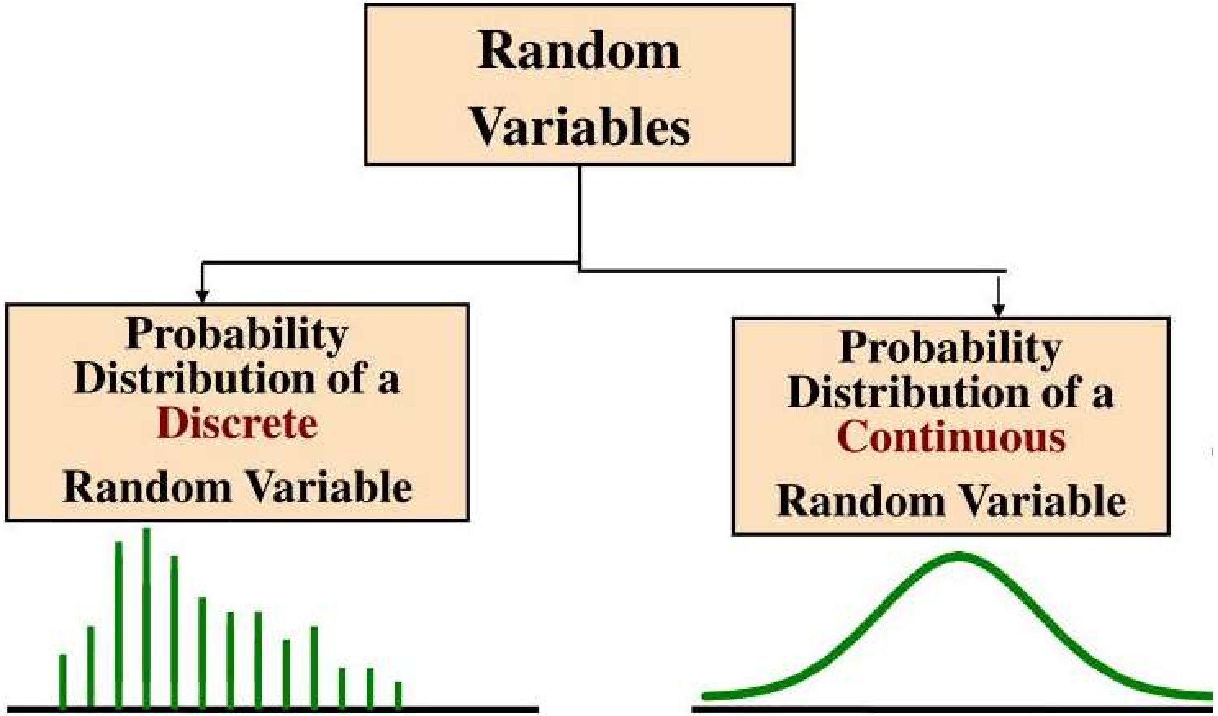
- Every probability $P(x_i)$ is a number between 0 and 1.
- $P(x_1) + P(x_2)+P(x_3)+ . . .+P(X_k) = 1$

- **For continuous random variable :** A continuous random variable X takes all values in an interval of numbers [a, b]. The probability distribution of X describes the probabilities $P(x1 \leq X \leq x2)$ of all possible intervals of numbers [x1, x2].

The probabilities $P(x1 \leq X \leq x2)$ must satisfy two requirements:

- For every interval [x1, x2], the probability $P(x1 \leq X \leq x2)$ is a number between 0 and 1.
- $P(a \leq X \leq b) = 1$.

Probability Distributions



Probability and Distribution functions

- Probability distribution is a function that gives the relative likelihood of occurrence of all possible outcomes of an experiment.
  - Probability density function or probability mass function
  - Cumulative distribution function.

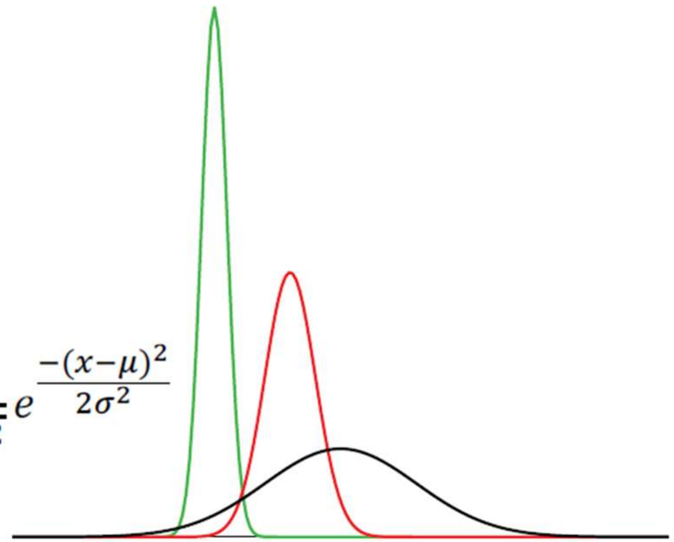
Central Limit Theorem

- The central limit theorem states that if we take repeated random samples from a population and calculate the mean value of each sample, then the distribution of the sample means will be approximately normally distributed, even if the population the samples came from is not normal.
- It help to draw conclusion about larger population
  - <u>Economics :</u> sample to find average annual income of the individuals
  - <u>Biology :</u> measure sample mean height to estimate the population mean height
  - <u>Manufacturing:</u> get sample products produced by the plant to find how many of the products are defective

Normal, Binomial and Poisson distribution

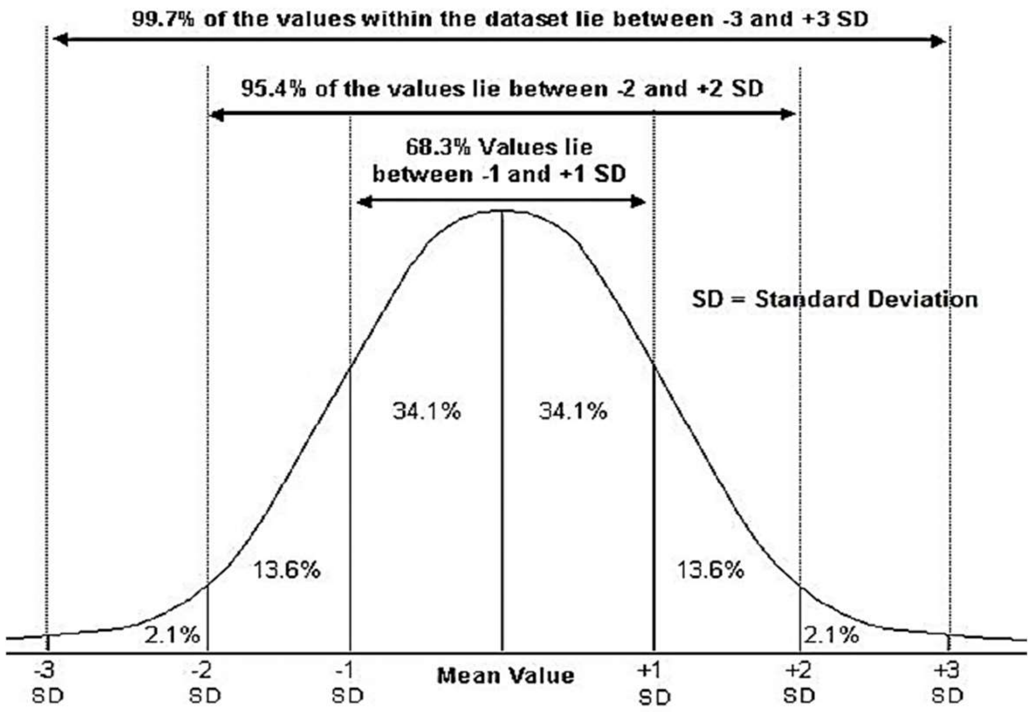
- Normal distribution
  - Gaussian Distribution, Bell Curve
  - Unimodal- Single peak at centre, symmetric
  - area under the normal curve is equal to 1.0
  - The mean, median, and mode of a normal distribution are equal.
  - Parameter μ and σ are mean and SD

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$



Normal distribution

- Area under 1SD of mean is 68%
- Area under 2SD of mean is approximately 95%
- Empirical Rule : 68-95-99
- https://onlinestatbook.com/2/calculators/normal.html



Normal distribution and Z-table

- Z – value of standard normal distribution
- Z = -2.5 represents a value 2.5 standard deviations below the mean
- area below Z is 0.0062.
- $Z = (X - \mu)/\sigma$
- Example : what portion of a normal distribution with a mean of 50 and a standard deviation of 10 is below 26?

  Z = (26 - 50)/10 = -2.4

  using z-table area is 0.0082

Z	Area below
-2.5	0.0062
-2.49	0.0064
-2.48	0.0066
-2.47	0.0068
-2.46	0.0069
-2.45	0.0071
-2.44	0.0073

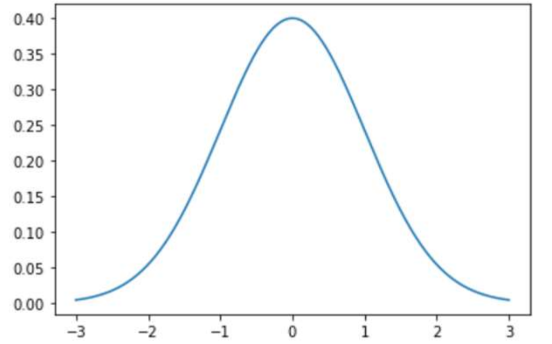
Normal distribution and Probability

- Example : Suppose the weight of the people follows normal distribution with mean 150 and SD 20 kg. Find the probability that a randomly selected person weighs a) at most 160 kg b) over 160 kg

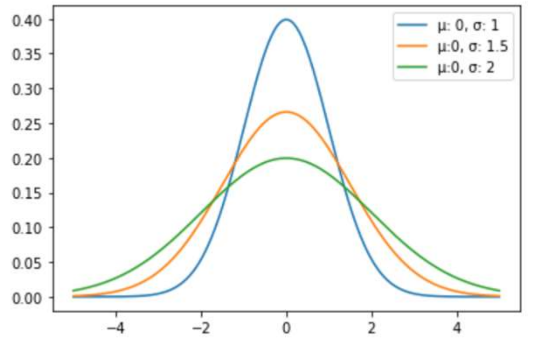
- P(X ≤ 160) = P($\frac{X-150}{20} \leq \frac{160-150}{20}) = P(Z \leq 0.5) = 0.6915$
- P(X>160) = 1- P(X ≤ 160) = 1- 0.6915 = 0.3085

Lab : Normal distribution

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
#Generate a data x-axis
x = np.arange(-3, 3, 0.001)
# 1) plot normal distribution mu= 0 and sd=1
plt.plot(x, norm.pdf(x, 0, 1))
```

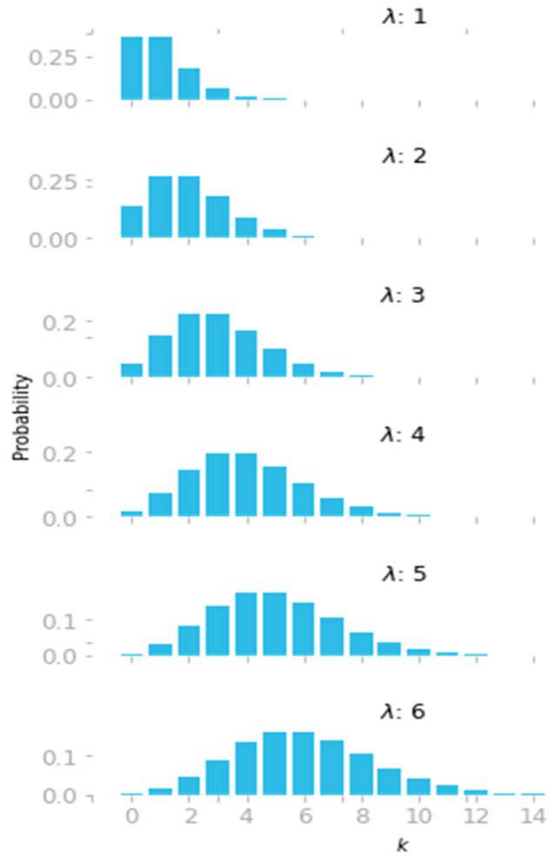


```
# 2) multiple normal distributions
plt.plot(x, norm.pdf(x, 0, 1), label='μ: 0, σ: 1')
plt.plot(x, norm.pdf(x, 0, 1.5), label='μ:0, σ: 1.5')
plt.plot(x, norm.pdf(x, 0, 2), label='μ:0, σ: 2')
plt.legend()
```



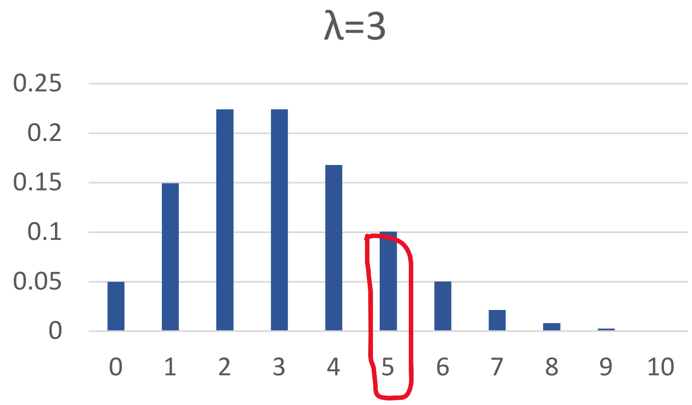
Poisson Distribution

- Discrete distribution
- Describe the number of events occurring in a fixed time interval or region.
- Require only one parameter λ (expected turnaround/mean)
- Bounder by 0 and ∞
- Rate of interval is constant
- Independent events
- Formula $P(X = x) = \frac{e^{-\lambda}\lambda^x}{x!}$ *Probability mass function*



Poisson Distribution

- *Customer turnout at every hour with a mean / $\lambda = 3$*
- $P(X = x) = \frac{e^{-\lambda}\lambda^{x}}{x!}$
- $P(X = 5) = \frac{e^{-3}3^{5}}{5!}$=0.101




```
from scipy.stats import poisson
#calculate probability, pmf = Probability mass function
poisson.pmf(k=5, mu=3)
Output => 0.100819
#calculate probability. Cdf = Cumulative distribution function
poisson.cdf(k=4, mu=7)
Output =>0.8152632445237722
```

Session 14 : Correlation, Outliers, Regression

Correlation

- To find relationship between two variables is to use the Pearson correlation coefficient, which measures the linear association between two variables.
  - -1 indicates a perfectly negative linear correlation
  - 0 indicates no linear correlation
  - 1 indicates a perfectly positive linear correlation

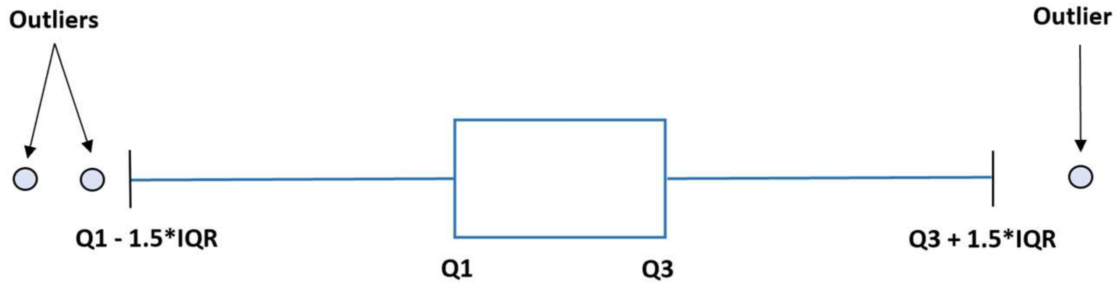
Correlation - Lab

```
import pandas as pd
import seaborn as sns

data = {'A': [4, 5, 5, 6, 7, 8, 8, 10],
'B': [12, 14, 13, 7, 8, 8, 9, 13],
'C': [22, 24, 26, 26, 29, 32, 20, 14] }
df = pd.DataFrame(data, columns=['A','B','C'])

#create correlation matrix
df.corr()

#create same correlation matrix with coefficients rounded to 3 decimals
df.corr().round(3)
corr = df.corr()

sns.heatmap(corr, cmap ="YlGnBu", linewidths = 0.1)
```

Outliers

- An outlier is an observation that lies abnormally far away from other values in a dataset.
  - way to find outliers in a dataset is interquartile range
  - The interquartile range, often abbreviated IQR, is the difference between the 25th percentile (Q1) and the 75th percentile (Q3) in a dataset. It measures the spread of the middle 50% of values.
  - Outliers = values > Q3 + 1.5*IQR or Q1 – 1.5*IQR
  - Outliers = values with z-scores > 3 or < -3
  - https://www.statology.org/interquartile-range-calculator/

- Q1= ¼(n+1)th term
- Q3= ¾(n+1)th term
- Q2= Q3–Q1 (Median)
- **z** = (X – μ) / σ => z-score



Outliers

Data
1
4
8
11
13
17
19
19
20
23
24
24
25
28
29
31
32

- https://www.statology.org/interquartile-range-calculator/
- Q1=(n+1)/4 =(17+1)/4= 4.5 => (11+13)/2=12
- Q3= (n+1)*(3/4) =(17+1) *(3/4)= 13.5 => (25+28)/2 =26.5
- IQR = Q3 – Q1 = 26.5-12=14.5

```
import numpy as np
import scipy.stats as stats
#define array of data
data = np.array([14, 19, 20, 22, 24, 26, 27, 30, 30, 31, 36, 38, 44, 47])
#calculate interquartile range
q3, q1 = np.quantile(data,[0.75,0.25])
iqr = q3 - q1 #display interquartile range
z = np.abs(stats.zscore(data))
data_clean = data[(abs(z)<=3)]
```

Linear Regression

- understand the relationship between a single explanatory variable and a single response variable.
- technique finds a line that best “fits” the data and takes on the following form:
- $\hat{y} = b0 + b1x$

  where:

  - $\hat{y}$: The estimated response value
  - b0: The intercept of the regression line
  - b1: The slope of the regression line

Linear Regression Lab

```
import pandas as pd
df = pd.DataFrame({'hours': [1, 2, 4, 5, 5, 6, 6, 7, 8, 10, 11, 11, 12, 12, 14],
'score': [64, 66, 76, 73, 74, 81, 83, 82, 80, 88, 84, 82, 91, 93, 89]})
import matplotlib.pyplot as plt
plt.scatter(df.hours, df.score)
plt.title('Hours studied vs. Exam Score')
plt.xlabel('Hours') ; plt.ylabel('Score') ; plt.show()
df.boxplot(column=['score'])
import statsmodels.api as sm
y = df['score'] #define response variable
x = df[['hours']] #define explanatory variable
x = sm.add_constant(x) #add constant to predictor variables
model = sm.OLS(y, x).fit() #fit linear regression model
print(model.summary())  #view model summary
print("fitted regression equation : Score = "+ str(model.conf_int()[0][0]) + " + " + str(model.conf_int()[0][1])
+ "*hours" )
fitted regression equation : Score = 65.334 + 1.9824*(hours)
```