Lab 1

```
# importing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import math as m

 # Bernoulli
def bernoulli(success_event, prob):
  return (prob**success_event)*((1-prob)**(1-success_event))

def plot_bernoulli():
  # successful and unsuccessful event
  x = [0,1]
  prob = 0.2
  y = [None]*2
  for _ in range(len(x)):
    y[_] = bernoulli(x[_],prob)

  plt.bar(x,y)

def normal(x,mean, std):
  return (1/std*(np.pi)**(1/2)) * ((np.e)**(-((x-mean)**2)/2*(std**2)))

def plot_normal():
  x = [-1,-1,0,0,0,0,0.5,1,1]

  m, std = 0,1
  y = [None]*len(x)
  for i in range(len(x)):
    y[i] = normal(x[i],m,std)

  normal_dist = pd.DataFrame({'x':x,'y':y})

  plt.bar(x,y)

def factorial(n):
  prod = 1
  for i in range(2,n+1):
    prod *= i
  return prod

def binomial(x,p,q,n):



  c_term = factorial(n) / (factorial(x) * factorial((n-x)))

  return c_term*(p**x)*((1-p)**(1-x))



def plot_binomial():

  x = [i for i in range(100)]
  n = 500
  p = 0.2
  q = 1 - p

  y = [None]*len(x)

  for i in range(len(x)):
    y[i] = binomial(x[i],p,q,n)
  plt.bar(x,y)

def poisson(x,lamb):
  return (m.e)**(-(lamb))*(lamb**x)/factorial(x)

def plot_poisson():
  x = [0,1,2,3,4,5]
  r = 0.5
  y = [None]*len(x)

  for i in range(len(x)):
    y[i] = poisson(x[i],r)
  plt.bar(x,y)
```

```
    plt.bar(x,y)

  def expo(x,lamb):
    if x<=0:
      return 0

    return lamb*((m.e)**(-(lamb*x)))

  def plot_expo():

    x = np.linspace(-1,5,60)
    x = x.tolist()
    r = 0.5
    y = [None]*len(x)

    for i in range(len(x)):
      y[i] = expo(x[i],r)
    plt.bar(x,y)


```

Lab 2

1. alpha: probability that fail to reject H0 when true
2. p-value - probability that get extreme values than calculated z-stat assuming H0 True (randomness)
3. if p < alpha, reject otherwise failed to reject : more randomness p = 0.05 : 5 % chance that Null is True , doesn't mean 95% Ha is correct
4. One Tailed , Two tailed, one sample , two sample
5. 5 % alpha, 95% confidence - z is +-1.96 for two tailed
6. z = x - u / (std/root n) for 2 sample - s1-x2 / root whole std1^2/n1 + std2^2/n2

Lab 3

```
filename = 'DATA.txt'
with open(filename,'r') as f:
  lines = f.read().replace("\n", " ")
  objs = lines.split()

```

```
'A B C A B C B D A C D D D C C'
```

General Points

BDA Domain Process

1. Databases , DFS, Paralle Programming
2. ML
3. Cloud
4. Apache Hive - Parallel Programming , HiveQL Scalaable, natural queries into mapreduce jobs over clusters - distributed computing, metadata indexes - on top of HDFS - METDATA - ACCESS CONTROL, PARTION EFFICIENCY ,PROCESSING SPEED
5. Data science - exact and analytical, BDA is not simple , requires optimization , scalability, large datasets , no exact method right, combination of methods

```
1
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.