Stock_Basic_Statistics

September 29, 2021

1 Basic Statistics in Python

1.1 Statistical Analysis

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import norm
import math

import warnings
warnings.filterwarnings("ignore")

# fix_yahoo_finance is used to fetch data
import fix_yahoo_finance as yf
yf.pdr_override()
```

[2]:
```
# input
symbol = 'AAPL'
market = '^GSPC'
start = '2017-01-01'
end = '2019-01-01'

# Read data
df = yf.download(symbol,start,end)
dfm = yf.download(market,start,end)

# View Columns
df.head()
```

```
[**********************100%**********************]  1 of 1 downloaded
[**********************100%**********************]  1 of 1 downloaded
```

[2]:

Date	Open	High	Low	Close	Adj Close
2017-01-03	115.800003	116.330002	114.760002	116.150002	111.709831

```
2017-01-04  115.849998  116.510002  115.750000  116.019997  111.584778
2017-01-05  115.919998  116.860001  115.809998  116.610001  112.152229
2017-01-06  116.779999  118.160004  116.470001  117.910004  113.402542
2017-01-09  117.949997  119.430000  117.940002  118.989998  114.441246

              Volume
Date
2017-01-03  28781900
2017-01-04  21118100
2017-01-05  22193600
2017-01-06  31751900
2017-01-09  33561900
```

[3]:
```
df.shape
```

[3]:
```
(502, 6)
```

[4]:
```
new_df = pd.DataFrame({symbol : df['Adj Close'], market : dfm['Adj Close']},␣
 ↪index=df.index)

# compute returns
new_df[['stock_returns','market_returns']] = new_df[[symbol,market]] /␣
 ↪new_df[[symbol,market]].shift(1) -1
new_df = new_df.dropna()
covmat = np.cov(new_df["stock_returns"],new_df["market_returns"])

# calculate measures now
beta = covmat[0,1]/covmat[1,1]
alpha= np.mean(new_df["stock_returns"])-beta*np.mean(new_df["market_returns"])
```

[5]:
```
print('Beta:', beta)
print('Alpha:', alpha)
```

```
Beta: 1.27474949119
Alpha: 0.000476775370087
```

[6]:
```
close = df['Adj Close']
```

1.2 Mean is the average

[7]:
```
mean = np.mean(close)
mean
```

[7]:
```
166.18815917729088
```

1.3 Median is the value of middlemost value

[8]:
```
median = np.median(close)
median
```

[8]:
```
165.22975150000002
```

1.4 Mode is the most frequent value

[9]:
```
mode = stats.mode(close)
print("The modal value is {} with a count of {}".format(mode.mode[0], mode.
 ↪count[0]))
```

```
The modal value is 115.41264299999999 with a count of 2
```

1.5 Range is a measure of how spread apart the values are

[10]:
```
range_of_stock = np.ptp(close)
range_of_stock
```

[10]:
```
117.807312
```

1.6 Variance is a measure of how variable the data is

[11]:
```
variance = np.var(close)
variance
```

[11]:
```
726.388795436073
```

1.7 Standard deviation is the square root of the variance and is measure how the data is spread out

[12]:
```
standard_deviation = np.std(close)
standard_deviation
```

[12]:
```
26.951600980944953
```

1.8 Standard error is the mean (SE of the mean) estimates the variability between sample means that you would obtain if you took multiple samples from the same population

[13]:
```
standard_error = stats.sem(close)
standard_error
```

[13]:
```
1.2041087306392817
```

1.9 Z-Scores measure how many standard deviations an element is from the mean

[14]:
```
z = np.abs(stats.zscore(close))
print(z)
```

```
[ 2.02133922  2.02597913  2.00492469  1.95853364  1.91999404  1.91571192
  1.89287324  1.91071589  1.9182098   1.88395176  1.88430926  1.89180261
  1.88395176  1.8810972   1.88502235  1.8168635   1.81472229  1.81436562
  1.82578524  1.83577704  1.57170668  1.57955727  1.55993038  1.51675168
  1.4725024   1.45430337  1.42025564  1.4310072   1.38907493  1.3270713
  1.30950975  1.31524432  1.30198326  1.26686015  1.25216558  1.27295251
  1.26829364  1.25861693  1.25646666  1.15611622  1.18586143  1.15647346
  1.17224313  1.16579179  1.18442849  1.19589768  1.17941065  1.17726098
  1.18478688  1.13210218  1.12385896  1.14894585  1.09626227  1.1543232
  1.0976958   1.11561629  1.12565139  1.11704979  1.01239656  1.00092741
  1.00773769  1.01741497  1.01598147  0.97763351  1.00451172  1.01741497
  1.0288836   1.03497596  1.09016875  1.08407635  1.11095627  1.08300235
  1.10558008  1.1242179   1.06113856  1.06723212  1.01813118  0.9862334
  1.01669764  1.01275554  1.01777335  0.91276174  0.87943051  0.89555857
  0.91455421  0.82746254  0.68231168  0.64718854  0.67335229  0.62584787
  0.54847314  0.56286935  0.57114712  0.75900249  0.6765918   0.65787755
  0.62440815  0.63124562  0.64780112  0.62872737  0.63808422  0.6359249
  0.66867357  0.65355947  0.5718667   0.62656917  0.60785436  0.57474501
  0.58842108  0.80470764  0.93282459  0.89071963  0.94218088  0.97349097
  1.04618528  0.89971528  0.94757915  0.91662971  0.92526586  0.90187404
  0.91842842  0.99364384  0.91806948  0.99544362  0.98320731  1.00192045
  0.98068791  1.0296309   0.97744951  0.94577937  0.92886546  0.92130732
  0.84825347  0.80254832  0.78383463  0.76512094  0.73129148  0.7557652
  0.75828234  0.69278497  0.66939315  0.64348137  0.74784642  0.78599339
  0.81370499  0.76620002  0.5110469   0.56754807  0.53803777  0.45094784
  0.40524324  0.36997517  0.55459441  0.47655559  0.3909297   0.32770321
  0.3511879   0.46282632  0.47583263  0.48630989  0.39345815  0.38623291
  0.41188481  0.39056849  0.33240004  0.28037433  0.26447725  0.24099426
  0.2391871   0.31036131  0.3165041   0.33998764  0.43500715  0.33131639
  0.35443873  0.39815554  0.44765282  0.38984609  0.43356178  0.43139453
  0.52749654  0.62432381  0.67851732  0.72693037  0.63335622  0.59397563
  0.62829819  0.59794946  0.60914972  0.58494374  0.62107235  0.55206654
  0.55531685  0.53580769  0.53363988  0.51015464  0.53002611  0.49425814
  0.38984609  0.36852921  0.39418115  0.53074966  0.52099481  0.52388558
  0.49028428  0.51521382  0.47908461  0.27531686  0.14272218  0.05890289
  0.13658054  0.09250308  0.06610416  0.12932956  0.14956224  0.2012273
  0.18821987  0.16719006  0.14180875  0.04644725  0.03549805  0.03774603
  0.00329943  0.00286485  0.11171362  0.17770469  0.1780676   0.14615907
  0.10917611  0.02099427  0.06493959  0.03593263  0.00939151  0.01519287
  0.03803611  0.02679511  0.02498283  0.09467237  0.05950054  0.08016859
  0.07835578  0.14180875  0.23064358  0.16247624  0.15558726  0.17951809
  0.17951809  0.01852847  0.0196166   0.03702021  0.030059    0.07980568
```

```
0.0787181   0.10772506  0.17915518  0.15558726  0.15486259  0.15341151
0.1893086   0.25493676  0.22230412  0.32781733  0.33361873  0.30461232
0.25167399  0.25312392  0.15087344  0.03810723  0.05261212  0.0761078
0.1120043   0.09532536  0.08263499  0.34659975  0.49199861  0.25486505
0.38140926  0.54058548  0.47177669  0.24241306  0.18307091  0.07275843
0.13184838  0.11146057  0.09034468  0.06194745  0.1140094   0.22322977
0.34956168  0.32844575  0.31861561  0.20502562  0.24907874  0.27128673
0.26582617  0.20611888  0.27565519  0.38633174  0.44968     0.38596883
0.33026538  0.33791075  0.31497523  0.21594846  0.21376425  0.06922879
0.01887629  0.16122583  0.12383954  0.03744335  0.10516003  0.05783175
0.09787872  0.0356226   0.0816072   0.12493109  0.03598663  0.02481221
0.14131505  0.11182459  0.17371669  0.1951966   0.2348807   0.32298463
0.30842141  0.12493109  0.13282974  0.15030355  0.23404017  0.20819179
0.18743937  0.25661222  0.14957661  0.00977423  0.26218523  0.27383556
0.52649874  0.57491975  0.60732195  0.65501485  0.7525849   0.72627113
0.71019005  0.64769502  0.71128672  0.66779524  0.64294328  0.69118535
0.67400823  0.71786488  0.71019005  0.72590596  0.70105345  0.68643476
0.66340979  0.7865746   0.84468421  0.8987741   0.92325973  0.90425618
0.83993303  0.82275587  0.86113043  0.80338544  0.80704055  0.73540773
0.73175322  0.62028404  0.64988717  0.61187778  0.59214274  0.49163773
0.57423356  0.56436662  0.61333959  0.59908607  0.67473913  0.55559574
0.60968563  0.70361192  0.79899999  0.79059429  0.70032255  0.81544625
0.82641042  0.81106023  0.83079643  0.79242127  0.84651175  0.83043066
0.83664309  0.88744475  0.95396065  0.93166713  0.81361927  0.7745138
0.78840102  1.19809583  1.41335937  1.43528712  1.47475832  1.40312555
1.40824186  1.46781443  1.44507537  1.49422047  1.52649529  1.54446617
1.65742658  1.81366335  1.73591156  1.72050818  1.72087446  1.73701217
1.76158444  1.82686664  1.89141513  2.01171065  2.08689494  2.18225191
2.20902424  2.1543778   2.01611195  1.95009658  1.84117058  2.04361863
1.94166142  2.13750752  2.04325119  1.82466659  1.83786933  1.84263747
1.90351786  1.81659772  1.93139079  1.98273712  1.9178212   2.08396113
2.11293525  2.16868174  2.24276598  2.34509003  2.19545465  2.05975518
2.04068422  2.1543778   1.76891955  1.69886972  1.97980331  1.80559451
1.98126964  1.94606216  1.75644997  1.87711178  1.92625632  2.00254122
1.72234139  1.8950827   1.76671946  1.61781724  1.65669286  1.86060779
1.98383773  1.4432416   1.22722394  1.30717637  1.5338298   1.50696405
1.3590141   0.97993981  0.90854083  0.70869938  0.87836251  0.95638548
0.67410392  0.34728912  0.33992908  0.17468197  0.26043354  0.2464484
0.49303104  0.44187441  0.4061749   0.63582842  0.33661651  0.26411469
0.0348292   0.07568125  0.03998177  0.05727952  0.12536535  0.0759487
0.13262593  0.05423497  0.24487641  0.39429833  0.61879868  0.76233205
0.38178516  0.41932467  0.41638065  0.36080685]
```

1.10 Contingency Table shows correlations between two variables

[15]:
```
data_crosstab = pd.crosstab(df['High'],
                            df['Low'],
                               margins = False)
print(data_crosstab)
```

```
Low         114.760002  115.750000  115.809998  116.470001  117.940002  \
High
116.330002           1           0           0           0           0
116.510002           0           1           0           0           0
116.860001           0           0           1           0           0
118.160004           0           0           0           1           0
119.300003           0           0           0           0           0
119.379997           0           0           0           0           0
119.430000           0           0           0           0           1
119.620003           0           0           0           0           0
119.930000           0           0           0           0           0
120.089996           0           0           0           0           0
120.099998           0           0           0           0           0
120.239998           0           0           0           0           0
120.449997           0           0           0           0           0
120.500000           0           0           0           0           0
120.809998           0           0           0           0           0
121.389999           0           0           0           0           0
121.629997           0           0           0           0           0
122.099998           0           0           0           0           0
122.349998           0           0           0           0           0
122.440002           0           0           0           0           0
129.190002           0           0           0           0           0
129.389999           0           0           0           0           0
130.490005           0           0           0           0           0
130.500000           0           0           0           0           0
132.089996           0           0           0           0           0
132.220001           0           0           0           0           0
132.449997           0           0           0           0           0
132.940002           0           0           0           0           0
133.820007           0           0           0           0           0
135.089996           0           0           0           0           0
…                  …           …           …           …           …
222.639999           0           0           0           0           0
222.820007           0           0           0           0           0
222.880005           0           0           0           0           0
222.949997           0           0           0           0           0
222.990005           0           0           0           0           0
223.250000           0           0           0           0           0
223.360001           0           0           0           0           0
223.490005           0           0           0           0           0
```

223.750000	0	0	0	0	0	
224.229996	0	0	0	0	0	
224.300003	0	0	0	0	0	
224.800003	0	0	0	0	0	
225.000000	0	0	0	0	0	
225.369995	0	0	0	0	0	
225.839996	0	0	0	0	0	
226.350006	0	0	0	0	0	
226.440002	0	0	0	0	0	
226.839996	0	0	0	0	0	
227.270004	0	0	0	0	0	
227.350006	0	0	0	0	0	
228.259995	0	0	0	0	0	
228.350006	0	0	0	0	0	
228.410004	0	0	0	0	0	
228.869995	0	0	0	0	0	
229.179993	0	0	0	0	0	
229.419998	0	0	0	0	0	
229.669998	0	0	0	0	0	
230.000000	0	0	0	0	0	
232.350006	0	0	0	0	0	
233.470001	0	0	0	0	0	

Low	118.209999	118.220001	118.300003	118.599998	118.809998	…	\
High						…	
116.330002	0	0	0	0	0	…	
116.510002	0	0	0	0	0	…	
116.860001	0	0	0	0	0	…	
118.160004	0	0	0	0	0	…	
119.300003	1	0	0	0	0	…	
119.379997	0	0	1	0	0	…	
119.430000	0	0	0	0	0	…	
119.620003	0	0	0	0	1	…	
119.930000	0	0	0	1	0	…	
120.089996	0	0	0	0	0	…	
120.099998	0	0	0	0	0	…	
120.239998	0	1	0	0	0	…	
120.449997	0	0	0	0	0	…	
120.500000	0	0	0	0	0	…	
120.809998	0	0	0	0	0	…	
121.389999	0	0	0	0	0	…	
121.629997	0	0	0	0	0	…	
122.099998	0	0	0	0	0	…	
122.349998	0	0	0	0	0	…	
122.440002	0	0	0	0	0	…	
129.190002	0	0	0	0	0	…	
129.389999	0	0	0	0	0	…	
130.490005	0	0	0	0	0	…	

```
130.500000            0            0            0            0            0  …
132.089996            0            0            0            0            0  …
132.220001            0            0            0            0            0  …
132.449997            0            0            0            0            0  …
132.940002            0            0            0            0            0  …
133.820007            0            0            0            0            0  …
135.089996            0            0            0            0            0  …
…                   …            …            …            …            …  …
222.639999            0            0            0            0            0  …
222.820007            0            0            0            0            0  …
222.880005            0            0            0            0            0  …
222.949997            0            0            0            0            0  …
222.990005            0            0            0            0            0  …
223.250000            0            0            0            0            0  …
223.360001            0            0            0            0            0  …
223.490005            0            0            0            0            0  …
223.750000            0            0            0            0            0  …
224.229996            0            0            0            0            0  …
224.300003            0            0            0            0            0  …
224.800003            0            0            0            0            0  …
225.000000            0            0            0            0            0  …
225.369995            0            0            0            0            0  …
225.839996            0            0            0            0            0  …
226.350006            0            0            0            0            0  …
226.440002            0            0            0            0            0  …
226.839996            0            0            0            0            0  …
227.270004            0            0            0            0            0  …
227.350006            0            0            0            0            0  …
228.259995            0            0            0            0            0  …
228.350006            0            0            0            0            0  …
228.410004            0            0            0            0            0  …
228.869995            0            0            0            0            0  …
229.179993            0            0            0            0            0  …
229.419998            0            0            0            0            0  …
229.669998            0            0            0            0            0  …
230.000000            0            0            0            0            0  …
232.350006            0            0            0            0            0  …
233.470001            0            0            0            0            0  …

Low          222.520004   222.570007   223.539993   224.020004   225.100006  \
High
116.330002            0            0            0            0            0
116.510002            0            0            0            0            0
116.860001            0            0            0            0            0
118.160004            0            0            0            0            0
119.300003            0            0            0            0            0
119.379997            0            0            0            0            0
119.430000            0            0            0            0            0
```

119.620003	0	0	0	0	0
119.930000	0	0	0	0	0
120.089996	0	0	0	0	0
120.099998	0	0	0	0	0
120.239998	0	0	0	0	0
120.449997	0	0	0	0	0
120.500000	0	0	0	0	0
120.809998	0	0	0	0	0
121.389999	0	0	0	0	0
121.629997	0	0	0	0	0
122.099998	0	0	0	0	0
122.349998	0	0	0	0	0
122.440002	0	0	0	0	0
129.190002	0	0	0	0	0
129.389999	0	0	0	0	0
130.490005	0	0	0	0	0
130.500000	0	0	0	0	0
132.089996	0	0	0	0	0
132.220001	0	0	0	0	0
132.449997	0	0	0	0	0
132.940002	0	0	0	0	0
133.820007	0	0	0	0	0
135.089996	0	0	0	0	0
…	…	…	…	…	…
222.639999	0	0	0	0	0
222.820007	0	0	0	0	0
222.880005	0	0	0	0	0
222.949997	0	0	0	0	0
222.990005	0	0	0	0	0
223.250000	0	0	0	0	0
223.360001	0	0	0	0	0
223.490005	0	0	0	0	0
223.750000	0	0	0	0	0
224.229996	0	0	0	0	0
224.300003	0	0	0	0	0
224.800003	0	0	0	0	0
225.000000	0	0	0	0	0
225.369995	0	0	0	0	0
225.839996	0	0	0	1	0
226.350006	0	0	0	0	0
226.440002	0	0	1	0	0
226.839996	1	0	0	0	0
227.270004	0	0	0	0	0
227.350006	0	0	0	0	0
228.259995	0	0	0	0	0
228.350006	0	1	0	0	0
228.410004	0	0	0	0	0
228.869995	0	0	0	0	0

229.179993	0	0	0	0	0
229.419998	0	0	0	0	0
229.669998	0	0	0	0	1
230.000000	0	0	0	0	0
232.350006	0	0	0	0	0
233.470001	0	0	0	0	0

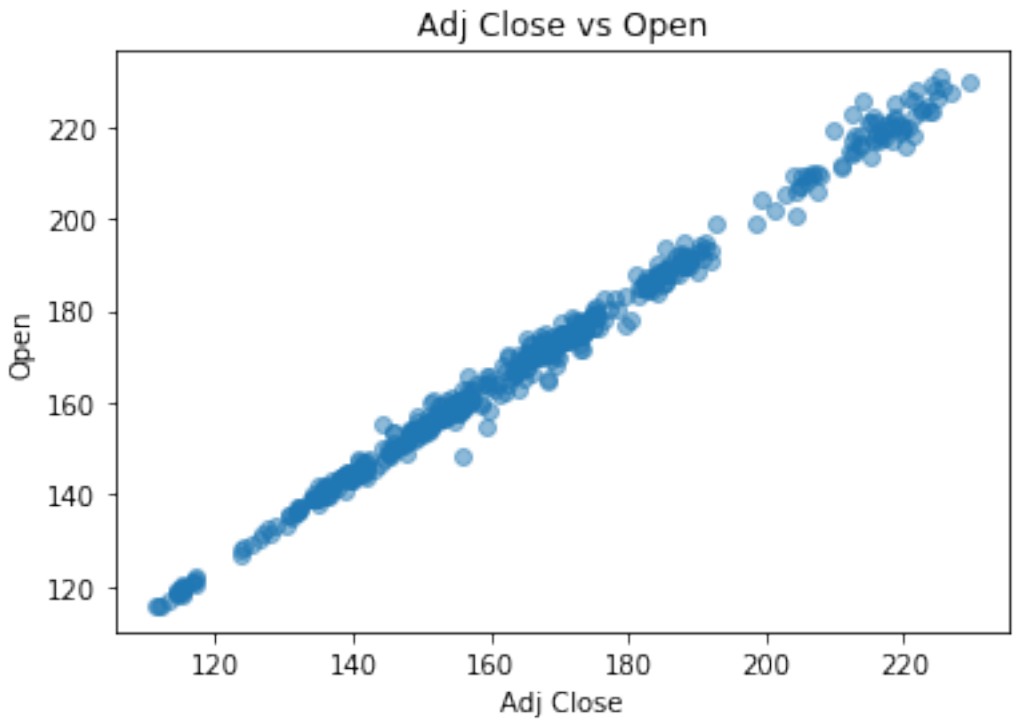
Low	226.000000	226.350006	226.630005	226.729996	229.779999
High					
116.330002	0	0	0	0	0
116.510002	0	0	0	0	0
116.860001	0	0	0	0	0
118.160004	0	0	0	0	0
119.300003	0	0	0	0	0
119.379997	0	0	0	0	0
119.430000	0	0	0	0	0
119.620003	0	0	0	0	0
119.930000	0	0	0	0	0
120.089996	0	0	0	0	0
120.099998	0	0	0	0	0
120.239998	0	0	0	0	0
120.449997	0	0	0	0	0
120.500000	0	0	0	0	0
120.809998	0	0	0	0	0
121.389999	0	0	0	0	0
121.629997	0	0	0	0	0
122.099998	0	0	0	0	0
122.349998	0	0	0	0	0
122.440002	0	0	0	0	0
129.190002	0	0	0	0	0
129.389999	0	0	0	0	0
130.490005	0	0	0	0	0
130.500000	0	0	0	0	0
132.089996	0	0	0	0	0
132.220001	0	0	0	0	0
132.449997	0	0	0	0	0
132.940002	0	0	0	0	0
133.820007	0	0	0	0	0
135.089996	0	0	0	0	0
…	…	…	…	…	…
222.639999	0	0	0	0	0
222.820007	0	0	0	0	0
222.880005	0	0	0	0	0
222.949997	0	0	0	0	0
222.990005	0	0	0	0	0
223.250000	0	0	0	0	0
223.360001	0	0	0	0	0
223.490005	0	0	0	0	0

223.750000	0	0	0	0	0
224.229996	0	0	0	0	0
224.300003	0	0	0	0	0
224.800003	0	0	0	0	0
225.000000	0	0	0	0	0
225.369995	0	0	0	0	0
225.839996	0	0	0	0	0
226.350006	0	0	0	0	0
226.440002	0	0	0	0	0
226.839996	0	0	0	0	0
227.270004	0	0	0	0	0
227.350006	0	0	0	0	0
228.259995	0	0	0	0	0
228.350006	0	0	0	0	0
228.410004	0	0	0	0	0
228.869995	1	0	0	0	0
229.179993	0	0	1	0	0
229.419998	0	1	0	0	0
229.669998	0	0	0	0	0
230.000000	0	0	1	0	0
232.350006	0	0	0	1	0
233.470001	0	0	0	0	1

[482 rows x 484 columns]

1.11 Scatter plot shows two variables that plot along two axes and it shows correlation or not

[16]:
```
plt.scatter(df['Adj Close'], df['Open'], alpha=0.5)
plt.title('Adj Close vs Open')
plt.xlabel('Adj Close')
plt.ylabel('Open')
plt.show()
```



1.12 Regression is a measure of the relation between the mean value of one variable (e.g. output) and corresponding values of other variables

[17]:
```
from sklearn.linear_model import LinearRegression

X = np.array(df['Open']).reshape(502,-1)
y = np.array(df['Adj Close'])
LR = LinearRegression().fit(X, y)
LR.score(X, y)
```

[17]: 0.99395097797644405

[18]:
```
LR.coef_
```

[18]: array([1.02181913])

[19]:
```
LR.intercept_
```

[19]: -7.2944207057100527

[20]:
```
LR.predict(X)
```

```
[20]: array([ 111.03223793,  111.08332378,  111.15485112,  112.03361659,
              113.22914293,  114.06703462,  114.03638107,  114.19987622,
              114.41445722,  113.62765137,  115.32387522,  114.71078579,
              115.78369077,  115.32387522,  114.86405968,  115.75303721,
              117.03031113,  117.51056714,  116.27416701,  116.49896927,
              122.5072627 ,  123.47799497,  123.81519017,  124.65308901,
              126.09384173,  126.92152851,  127.22806199,  128.05574877,
              128.68927152,  129.08777996,  131.18251225,  131.33577899,
              130.75335026,  131.90799566,  132.11235642,  133.08309686,
              131.58102171,  132.83785413,  132.77654805,  133.60421848,
              135.76025788,  134.51363751,  135.11650671,  134.79974585,
              134.68734472,  134.47277088,  134.99389353,  134.585172  ,
              135.04498755,  135.15738868,  136.49596867,  136.78207701,
              136.1689794 ,  137.91629727,  135.60699114,  137.04774487,
              137.29298658,  135.13694718,  136.69011737,  139.52054513,
              140.04168209,  139.56142607,  139.55121401,  139.08117006,
              140.07233564,  140.1438548 ,  139.57163915,  139.43881288,
              138.76440817,  137.39517462,  137.71193651,  137.27254611,
              137.20102694,  137.68128295,  137.00687824,  138.2534986 ,
              139.33662484,  139.75557477,  140.32779042,  139.76578683,
              139.93949404,  140.97154158,  143.46476698,  141.47222274,
              142.42252271,  142.6677501 ,  144.98728362,  149.93288414,
              149.68765776,  148.48190301,  150.78099606,  152.11957708,
              152.0480569 ,  149.65700421,  147.27616359,  149.43220298,
              150.06572573,  150.98535682,  149.90223059,  149.78983048,
              150.06572573,  149.47306859,  150.03507218,  149.21761381,
              149.63656374,  150.41314015,  149.96353769,  151.10798534,
              151.34299965,  151.28169255,  141.62550481,  143.07648695,
              143.42390137,  139.15270455,  139.62273318,  139.50011999,
              142.78015021,  141.40070358,  141.65615836,  141.00219514,
              143.08669901,  140.87956662,  140.34823089,  140.57303314,
              140.30734995,  140.74674035,  139.53077252,  138.84615574,
              138.72352723,  139.95993553,  140.59345829,  141.75833108,
              141.38026311,  143.90415739,  144.77270978,  145.16099083,
              146.4689183 ,  147.5111779 ,  145.96823612,  146.57110634,
              147.81772671,  149.40154943,  149.81027095,  145.86604808,
              145.87626116,  145.05881811,  155.46092973,  153.18227715,
              152.18089849,  153.19249024,  154.76609988,  155.44048926,
              156.09445249,  152.72246161,  155.50181067,  156.87104525,
              158.17897169,  156.72799057,  154.00994861,  153.6420927 ,
              154.38801658,  155.24635589,  156.6360156 ,  155.8389977 ,
              156.33969419,  156.29882857,  160.0795563 ,  159.91606115,
              161.10137543,  160.02846228,  158.96577753,  158.33223843,
              157.07540601,  156.7075501 ,  158.86358949,  156.06379893,
              155.16460832,  154.63325828,  156.30904166,  155.69594405,
              154.05081422,  151.90500324,  147.55204351,  145.96823612,
              147.79728624,  149.86136497,  149.95332461,  149.25849577,
```

```
150.3313936 ,  150.07593882,  149.68765776,  150.24964603,
151.05689131,  151.91521632,  152.1706711 ,  152.07871045,
152.46700683,  152.85528788,  154.05081422,  155.9718393 ,
156.62580252,  152.87572835,  152.73267469,  153.01878201,
152.4056844 ,  153.0392235 ,  153.36619745,  155.47114179,
160.17151594,  164.26900555,  166.28199026,  162.94065294,
170.50210839,  168.83653809,  170.41014875,  171.1765131 ,
171.63632865,  171.63632865,  169.99119882,  169.52115487,
166.38417831,  167.62057128,  167.4775166 ,  166.71115225,
167.21184976,  169.84814517,  171.62611557,  171.57502154,
170.80865719,  169.10222129,  166.85420693,  166.36373784,
168.94893922,  165.45431983,  163.86028403,  165.42366628,
166.91552834,  165.59737349,  168.61173686,  168.96937969,
168.86719165,  170.12404042,  171.40131433,  171.55458107,
171.39108593,  170.6758156 ,  171.19693825,  167.23229023,
166.5170199 ,  167.43665099,  166.94618189,  166.57832701,
169.00003324,  169.0102453 ,  169.92989172,  170.85975122,
171.06411198,  169.6437844 ,  171.10497759,  172.72966695,
174.48719688,  172.69901339,  175.98927202,  175.21269561,
173.87411459,  173.87411459,  173.82302057,  171.02323104,
168.45847012,  166.57832701,  161.84729931,  163.21653286,
163.52308167,  162.32755533,  155.27700944,  150.91383766,
159.35405756,  156.49296093,  153.20271762,  154.66391183,
158.18918477,  159.30296354,  166.20024269,  168.82632603,
168.50956415,  169.30658205,  168.25410936,  170.16490603,
172.90338948,  175.7133921 ,  175.87687192,  175.1411601 ,
169.27592849,  171.73851669,  174.49742528,  171.46262042,
172.01439662,  174.54851931,  176.92934358,  179.27953065,
176.96001246,  175.10029449,  175.25356123,  173.89455506,
171.76916922,  171.56479313,  166.41483186,  164.76970203,
164.44272808,  170.17511911,  163.60482924,  164.17704591,
162.98151855,  164.00333768,  161.18312301,  169.05112726,
167.40599744,  166.29221867,  169.48028926,  168.69348444,
169.89923919,  171.29912629,  171.55458107,  173.04644314,
174.39523724,  170.25686669,  167.02792947,  163.17566725,
161.99035297,  158.87380155,  160.40653025,  160.28391706,
158.37312039,  162.74650526,  171.75894183,  172.42313347,
174.8448397 ,  181.92603914,  181.73190577,  183.32594157,
184.54190838,  186.33009186,  185.83960846,  183.56095588,
182.83547247,  184.80757625,  183.97990479,  184.80757625,
185.19587263,  183.12158081,  185.59438107,  185.04259056,
184.39885472,  184.52146791,  184.01055835,  184.79736316,
188.52699687,  189.9882064 ,  190.56042307,  191.0815447 ,
188.04674085,  188.23067647,  188.27154209,  189.32401477,
188.43503723,  186.88186806,  184.68496306,  181.88517251,
183.12158081,  184.0412119 ,  182.88655117,  180.10720211,
179.6882675 ,  181.97713316,  180.82248776,  183.06025838,
```

```
180.53637942,  184.59298708,  182.00778671,  182.17128084,
186.34030495,  187.57671325,  185.31848581,  186.3709585 ,
187.95478122,  188.40438368,  186.59575973,  188.67005155,
186.53445262,  188.67005155,  187.54604437,  189.35466832,
189.97797901,  191.56180174,  191.95009709,  188.79266473,
187.15776332,  196.1804283 ,  197.66206298,  204.25279332,
205.2439589 ,  206.59276731,  203.25141466,  206.80734115,
204.58999568,  206.58253992,  207.45109232,  206.49057927,
209.07578065,  210.80265703,  215.56433827,  214.23597033,
211.47706174,  212.03905   ,  214.03160957,  214.59359783,
216.49418244,  217.65905523,  220.82670068,  224.15782594,
226.09929149,  226.69194761,  223.8717176 ,  219.39616002,
218.4765136 ,  215.47236331,  222.55357705,  221.10259593,
223.38124851,  219.7026935 ,  215.24756106,  215.9730598 ,
217.7510302 ,  218.3028064 ,  214.2564108 ,  217.25033371,
218.52760763,  221.40914473,  222.40029499,  225.62924753,
224.91397721,  227.77507384,  228.52099772,  225.63947594,
219.76401593,  221.22520912,  223.08492811,  211.90622373,
217.93495049,  218.69110278,  216.41243487,  219.85597556,
215.31909657,  215.52345733,  217.29119932,  213.24480475,
220.16252437,  215.16582983,  213.31632392,  216.67811704,
208.46268304,  214.31771791,  216.53506338,  206.82778162,
201.46323118,  199.03129653,  203.16966708,  207.2671567 ,
202.74050509,  196.04758671,  188.51678481,  190.836303  ,
185.20608469,  187.36212408,  186.85121451,  174.96745289,
176.35712793,  171.46262042,  170.74735009,  167.95777364,
173.29167053,  179.35106617,  176.92934358,  181.19034367,
177.6037483 ,  168.21322842,  169.98098574,  161.30573619,
168.11105571,  166.82355338,  166.91552834,  165.39301273,
161.76555174,  161.69403257,  162.32755533,  156.60536205,
152.98812948,  144.08807768,  144.24135974,  151.94586885,
153.6420927 ,  154.69456538])
```

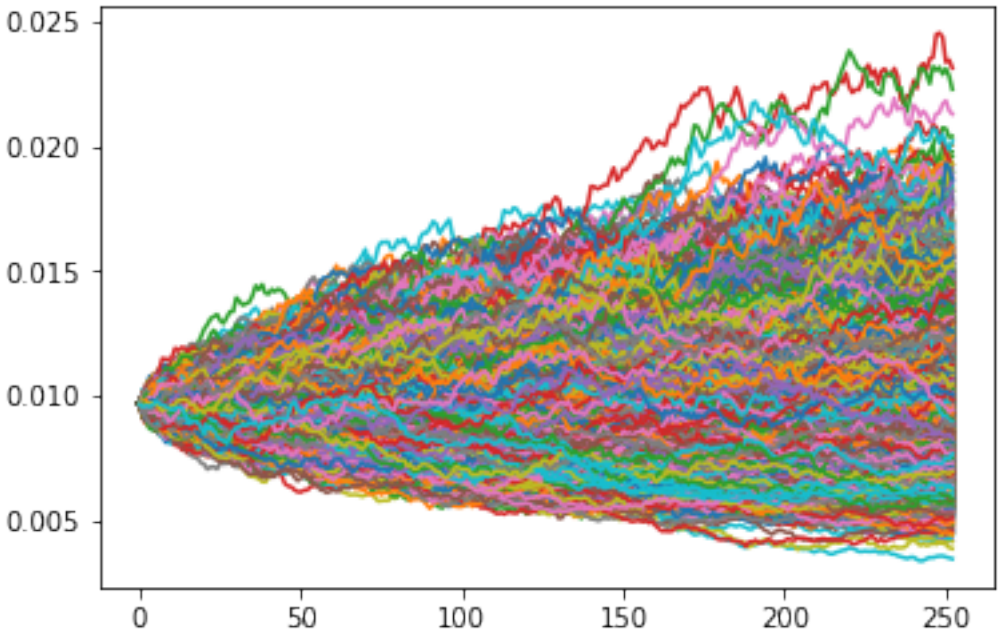
1.13 Elementary Probability Theory the outcome that could happen

1.13.1 Monte Carlo method is an experimentals of computational algorithms that rely on repeated random samples.

[21]:
```
df['Returns'] = df['Adj Close'].pct_change()
df['Returns'] = df['Returns'].dropna()
```

[22]:
```
values = []
S = df['Returns'][-1] #Starting stock price
T = 252 #Number of trading days
mu = df['Returns'].mean() #Mean
sigma = df['Returns'].std()*math.sqrt(252) #Volatility
```

[23]:
```
for i in range(10000):
    # Create list of daily returns using random normal distribution
    daily_returns=np.random.normal(mu/T,sigma/math.sqrt(T),T)+1

    # Set starting price and create price series generated by above random␣
 ↪daily returns
    price_list = [S]

    for x in daily_returns:
        price_list.append(price_list[-1]*x)

    # Plot the data
    plt.plot(price_list)
plt.show()
```



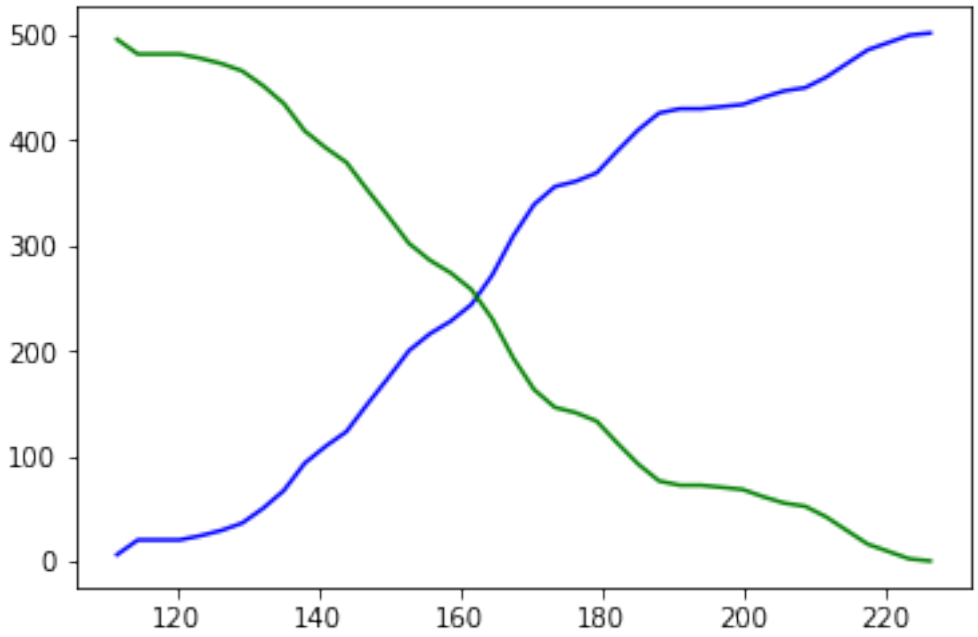
1.14 Random variables and probability distributions

https://www.investopedia.com/articles/06/probabilitydistribution.asp

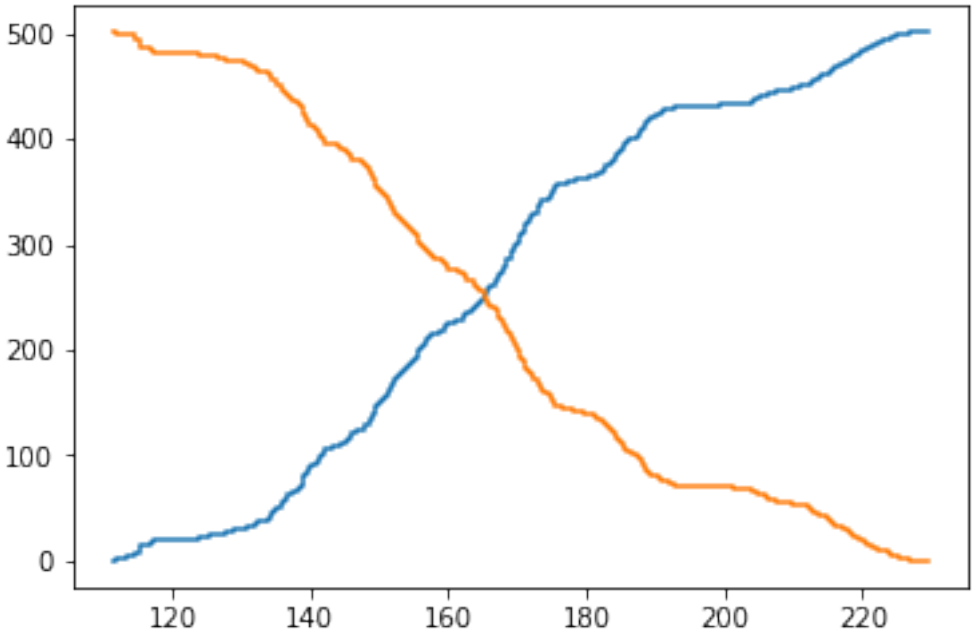
1.14.1 Cumulative Distribution

[24]:
```
data = df['Adj Close']
values, base = np.histogram(data, bins=40)
#evaluate the cumulative
cumulative = np.cumsum(values)
```

```
# plot the cumulative function
plt.plot(base[:-1], cumulative, c='blue')
#plot the survival function
plt.plot(base[:-1], len(data)-cumulative, c='green')

plt.show()
```



[25]:
```
sorted_data = np.sort(data)  # Or data.sort(), if data can be modified

# Cumulative counts:
plt.step(sorted_data, np.arange(sorted_data.size))  # From 0 to the number of␣
 ↪data points-1
plt.step(sorted_data[::-1], np.arange(sorted_data.size))  # From the number of␣
 ↪data points-1 to 0

plt.show()
```

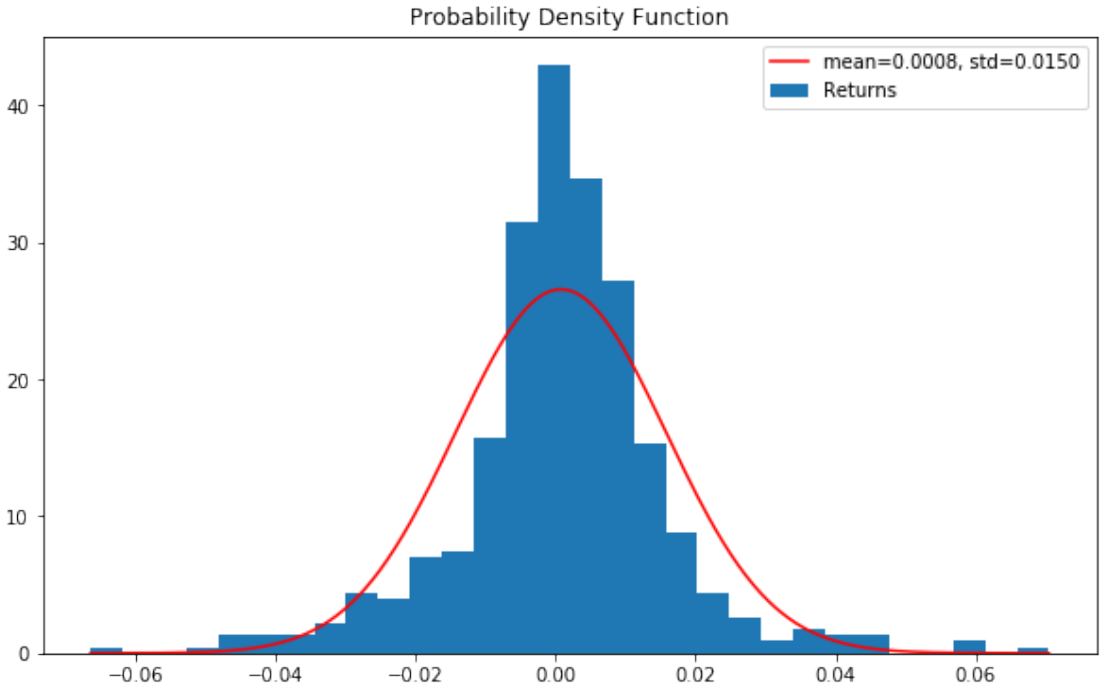


1.14.2 Probability Density Function

Probability Density Function (PDF) is continuous random variable and have value that is given sample in the sample space can be interpreted as providing a relative likelihood that the value of the random variable would equal that sample. (Wikipedia)

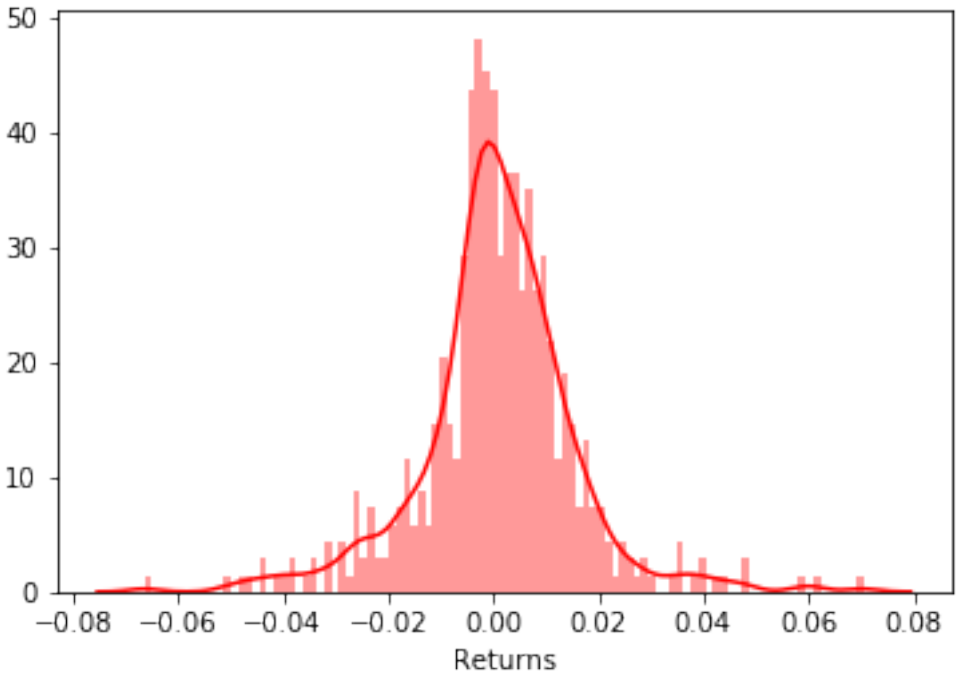
[26]:
```
values = df['Returns'][1:]
x = np.linspace(values.min(), values.max(), len(values))
loc, scale = stats.norm.fit(values)
param_density = stats.norm.pdf(x, loc=loc, scale=scale)
label = 'mean=%.4f, std=%.4f' % (loc, scale)

fig, ax = plt.subplots(figsize=(10, 6))
ax.hist(values, bins=30, normed=True)
ax.plot(x, param_density, 'r-', label=label)
ax.legend(loc='best')
ax.set_title("Probability Density Function")
```

[26]:
```
Text(0.5,1,'Probability Density Function')
```



[27]:
```
sns.distplot(df['Returns'].dropna(),bins=100,color='red')
```

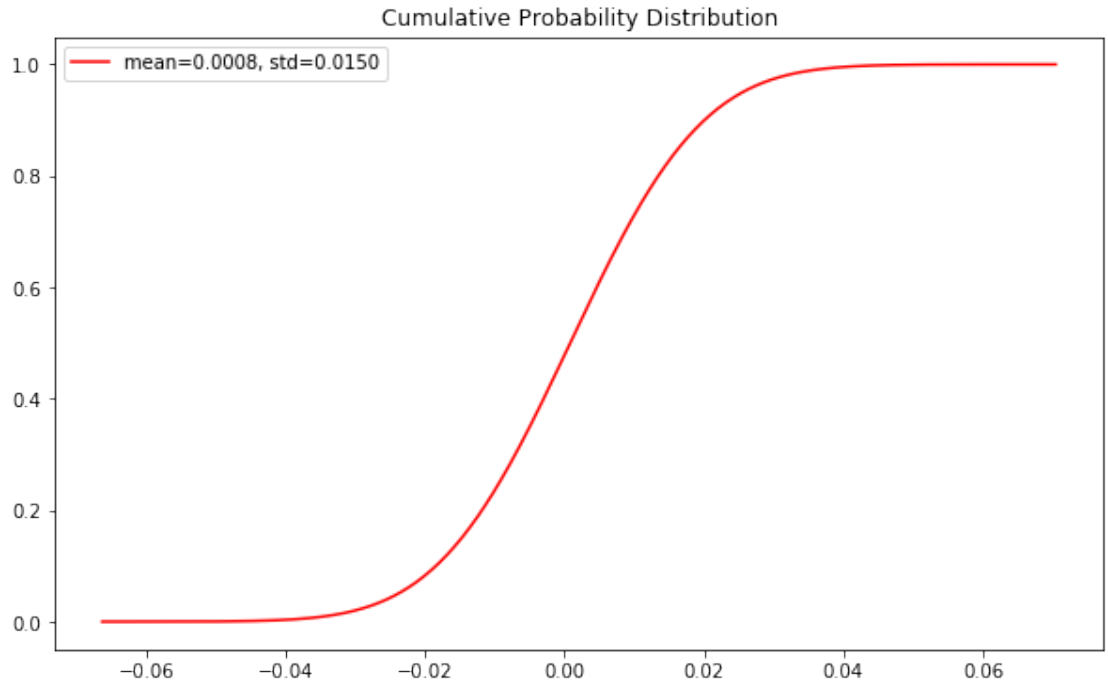
[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1bac62f0c88>
```



1.15 Cumulative Probability Distribution

[28]:
```
values = df['Returns'][1:]
x = np.linspace(values.min(), values.max(), len(values))
loc, scale = stats.norm.fit(values)
param_density = stats.norm.cdf(x, loc=loc, scale=scale)
label = 'mean=%.4f, std=%.4f' % (loc, scale)

fig, ax = plt.subplots(figsize=(10, 6))
#ax.hist(values, bins=30, normed=True)
ax.plot(x, param_density, 'r-', label=label)
ax.legend(loc='best')
ax.set_title("Cumulative Probability Distribution")
```

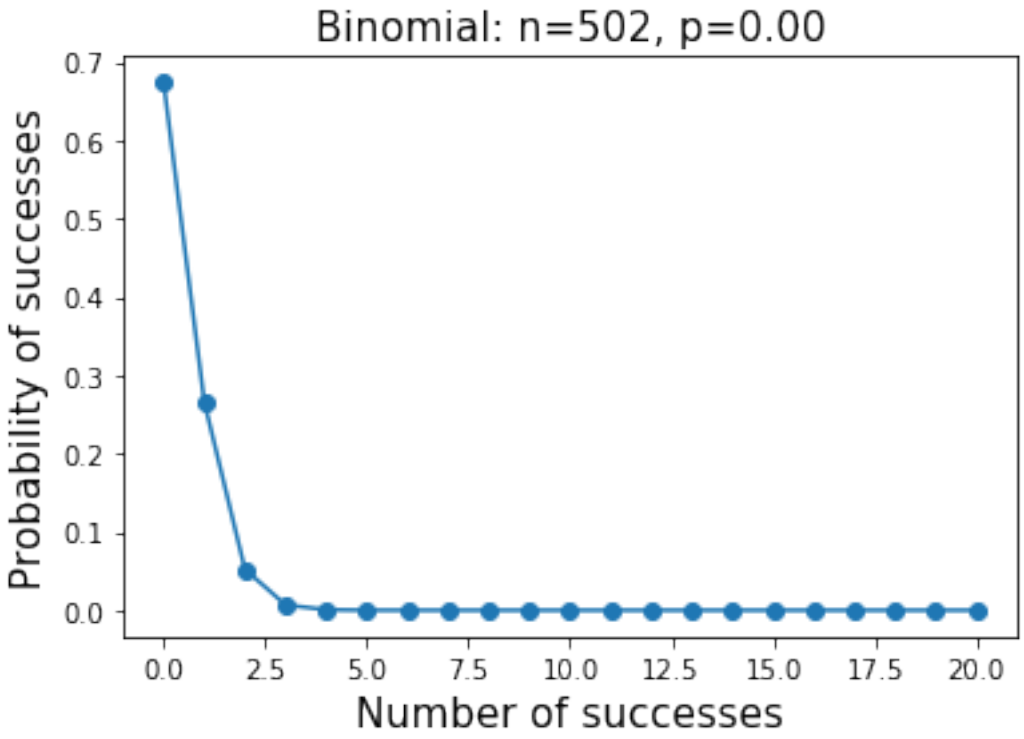
[28]:
```
Text(0.5,1,'Cumulative Probability Distribution')
```



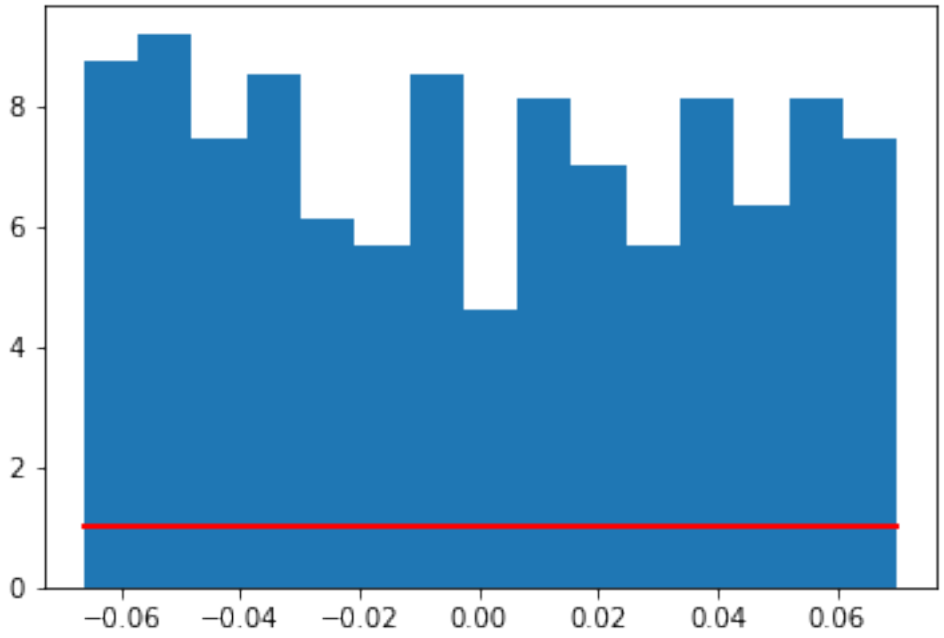
1.16 Binomial Distribution

[29]:
```
from scipy.stats import binom

n = len(df['Returns'])
p = df['Returns'].mean()
k = np.arange(0,21)
```

```
binomial = binom.pmf(k,n,p)

plt.plot(k, binomial, 'o-')
plt.title("Binomial: n=%i, p=%.2f" % (n,p), fontsize=15)
plt.xlabel("Number of successes", fontsize=15)
plt.ylabel("Probability of successes", fontsize=15)
plt.show()
```

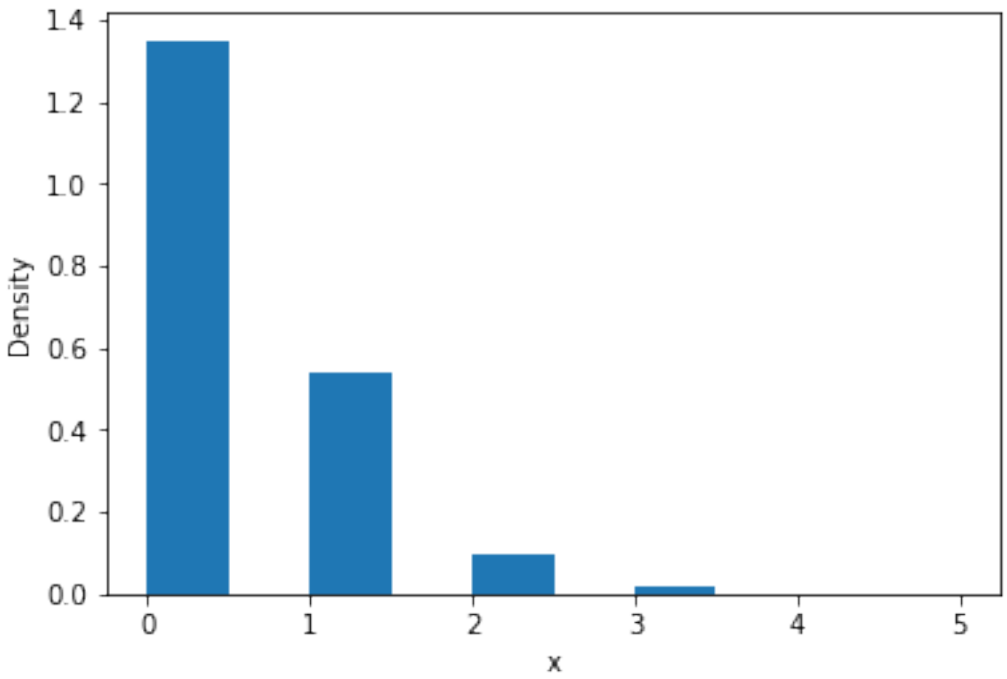


[30]:
```
s = np.random.uniform(values.min(), values.max(), len(values))

import matplotlib.pyplot as plt
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
plt.show()
```



[31]:
```
binom_sim = binom.rvs(n = n, p = p, size=10000)
print("Mean: %f" % np.mean(binom_sim))
print("SD: %f" % np.std(binom_sim, ddof=1))
plt.hist(binom_sim, bins = 10, normed = True)
plt.xlabel("x")
plt.ylabel("Density")
plt.show()
```

```
Mean: 0.391000
SD: 0.623665
```

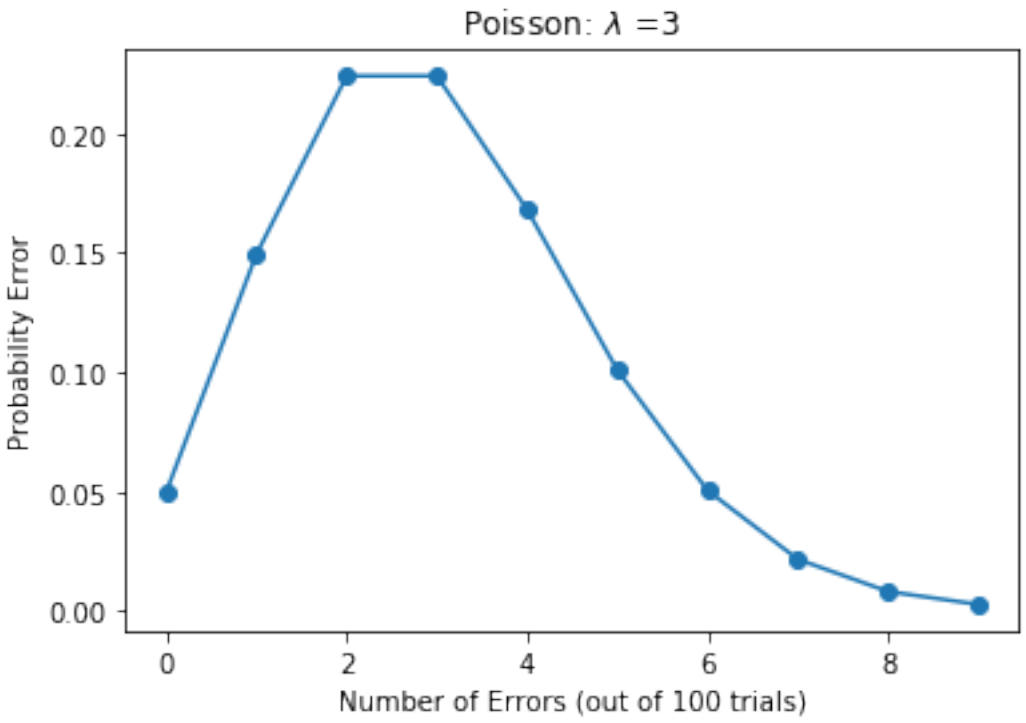


1.17 Poisson Distribution

[32]:
```
rate = 3 # Error Rate
n = np.arange(0,10) # Number of Trials
y = stats.poisson.pmf(n, rate)
y
```

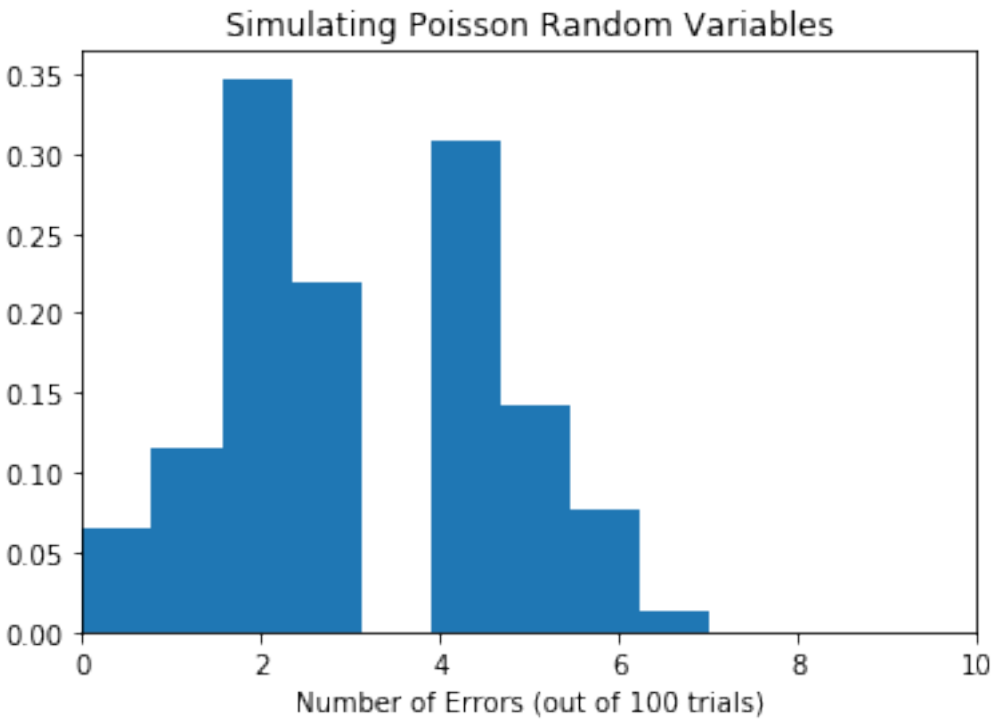
[32]:
```
array([ 0.04978707,  0.14936121,  0.22404181,  0.22404181,  0.16803136,
        0.10081881,  0.05040941,  0.02160403,  0.00810151,  0.0027005 ])
```

[33]:
```
plt.plot(n, y, 'o-')
plt.title('Poisson: $\lambda$ =%i' % rate)
plt.ylabel('Probability Error')
plt.xlabel('Number of Errors (out of 100 trials)')
plt.show()
```



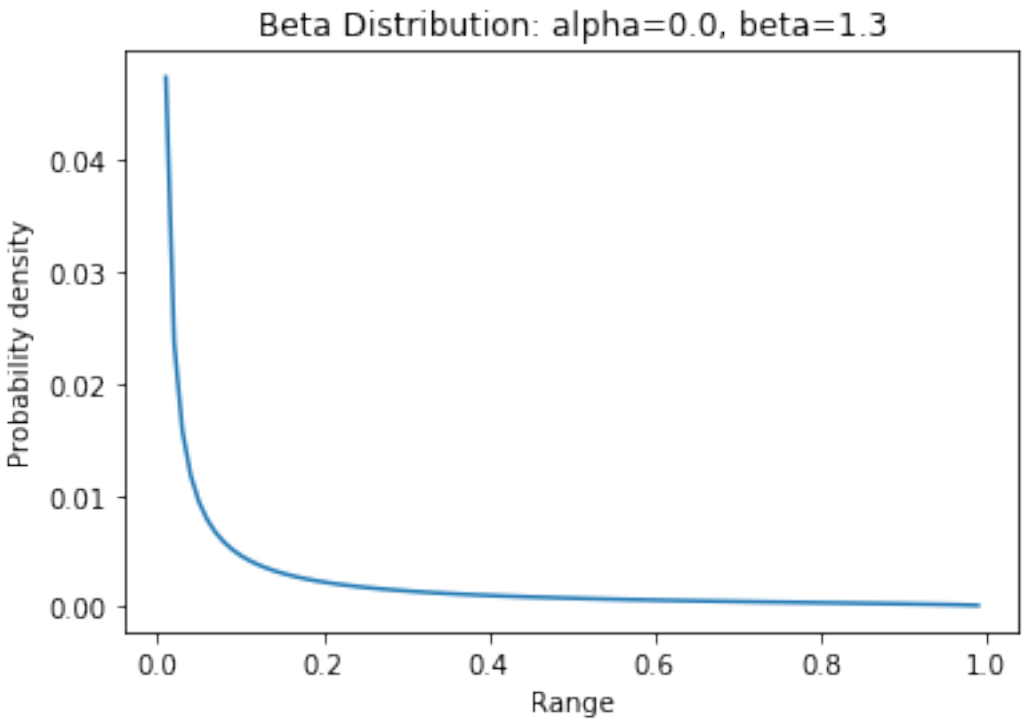
[34]:
```
data = stats.poisson.rvs(mu=3, loc=0, size=100)
print("Mean: %f" % np.mean(data))
print("Standard Deviation: %f" % np.std(data, ddof=1))

plt.hist(data, bins = 9, normed = True)
plt.xlim(0,10)
plt.xlabel('Number of Errors (out of 100 trials)')
plt.title('Simulating Poisson Random Variables')
plt.show()
```

```
Mean: 3.080000
Standard Deviation: 1.574288
```



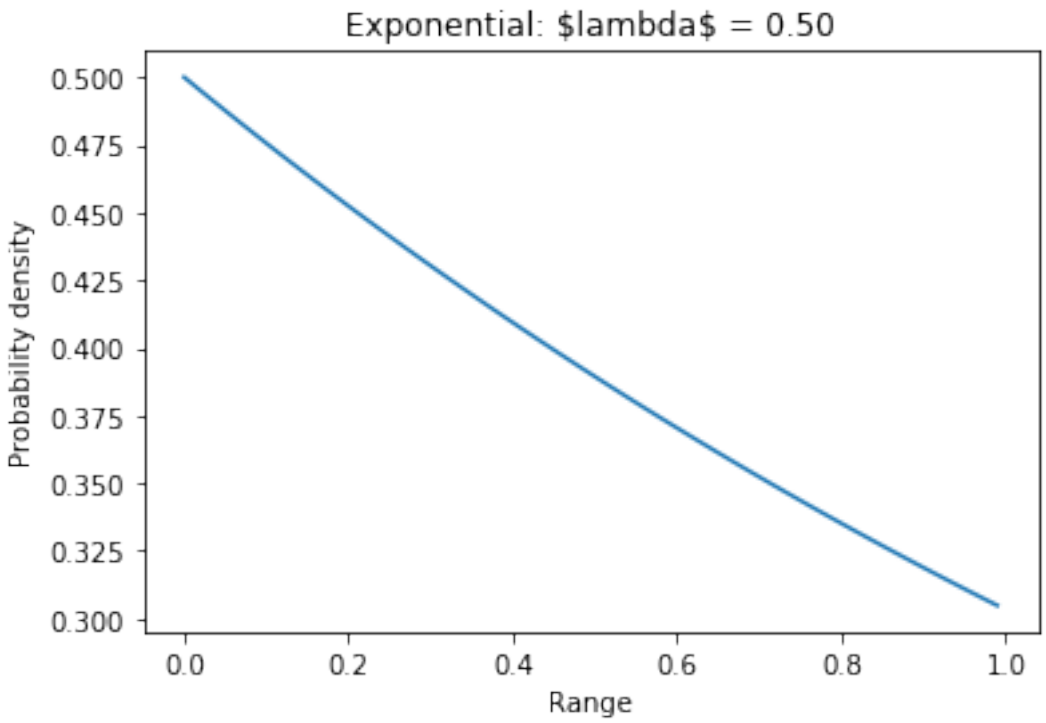
1.18 Beta Distribution

[35]:
```
alpha = alpha
beta = beta
x = np.arange(0, 1, 0.01)
y = stats.beta.pdf(x, alpha, beta)
plt.plot(x, y)
plt.title('Beta Distribution: alpha=%.1f, beta=%.1f' % (alpha,beta))
plt.xlabel('Range')
plt.ylabel('Probability density')
plt.show()
```



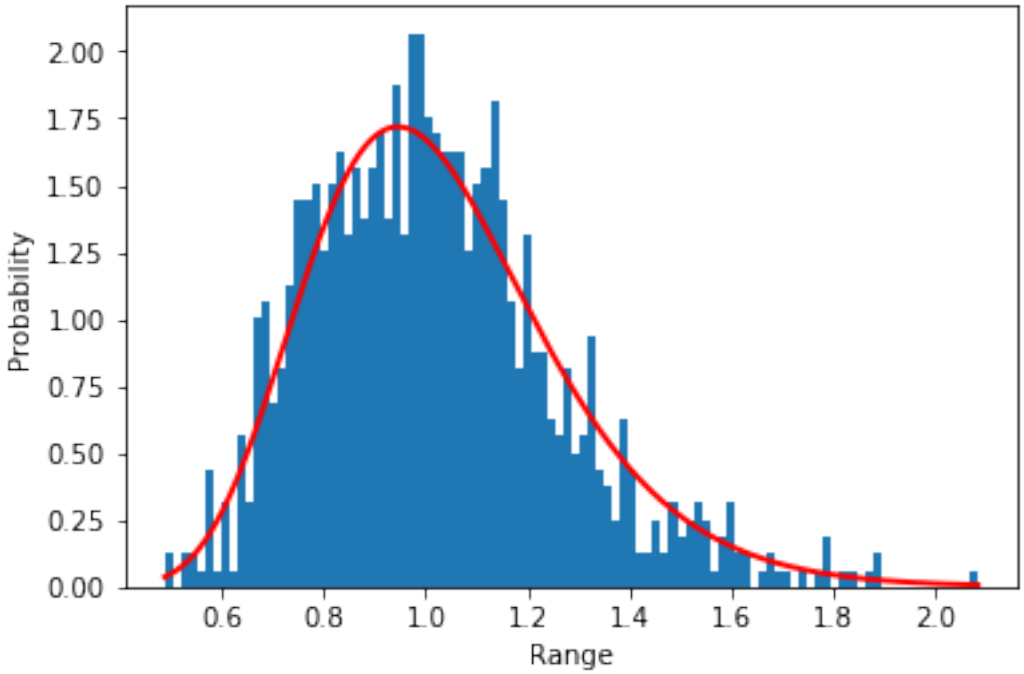
1.19 Exponential Distribution

[36]:
```
lambd = 0.5 # lambda
x = np.arange(0, 1, 0.01)
y = lambd * np.exp(-lambd * x)
plt.plot(x, y)
plt.title('Exponential: $lambda\$ = %.2f' % lambd)
plt.xlabel("Range")
plt.ylabel("Probability density")
plt.show()
```



1.20 Lognormal Distribution

[37]:
```
from scipy.stats import lognorm

s = np.random.lognormal(mu, sigma, 1000)

count, bins, ignored = plt.hist(s, 100, normed=True, align='mid')
x = np.linspace(min(bins), max(bins), 10000)
pdf = (np.exp(-(np.log(x) - mu)**2 / (2 * sigma**2)) / (x * sigma * np.sqrt(2 *␣
 ↪np.pi)))

plt.plot(x, pdf, linewidth=2, color='r')
plt.xlabel('Range')
plt.ylabel('Probability')
plt.axis('tight')
plt.show()
```


Probability
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
Range