Business Context

A leading pharmaceutical company has tested five batches of a vaccine. With 300,000 doses already administered, the sixth batch of 60,000 doses needs quality assurance testing for effectiveness and curing time. Previous data shows each dose is twice as likely to be satisfactory as unsatisfactory. This test is to ensure the sixth batch's quality, not a clinical trial. Objective Analyze random samples from the sixth batch to infer its quality and curing time. Your tasks include:

Task 1: Analyze 10 randomly selected doses to determine:

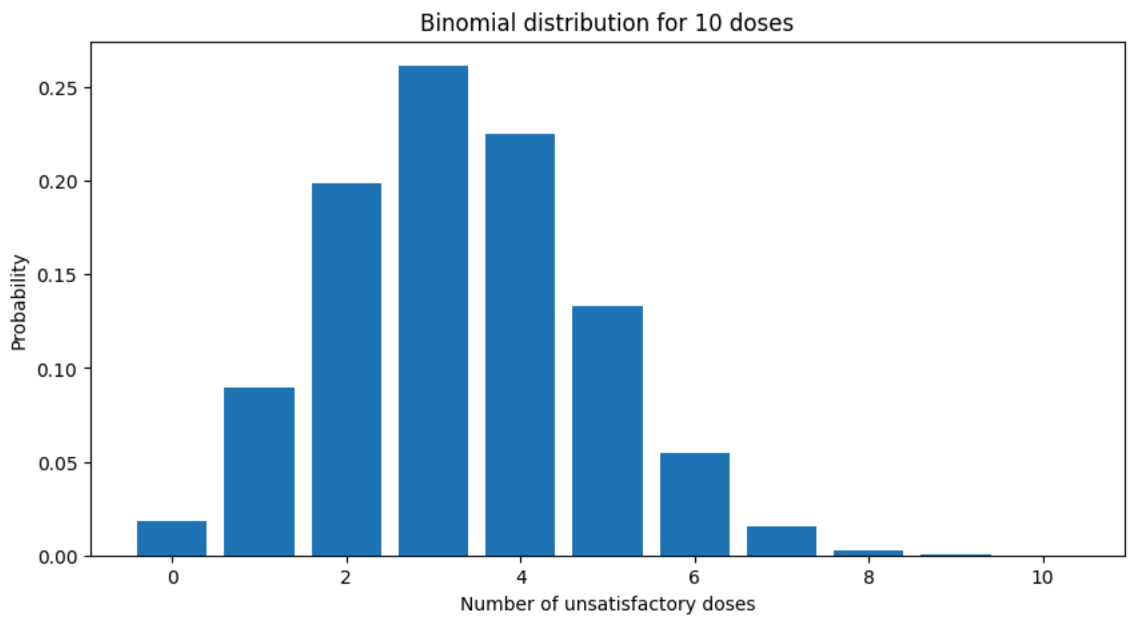
- Probability distribution of unsatisfactory doses
- Probability that exactly 3 out of 10 doses are unsatisfactory
- Probability that at most 3 out of 10 doses are unsatisfactory
- Probability that more than 8 out of 10 doses are satisfactory

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import binom
import scipy.stats as stats
```

```
n=10
p=0.33
k=np.arange(0,11)
binomial_prob=binom.pmf(k=k,n=n,p=p)
binomial_prob
```

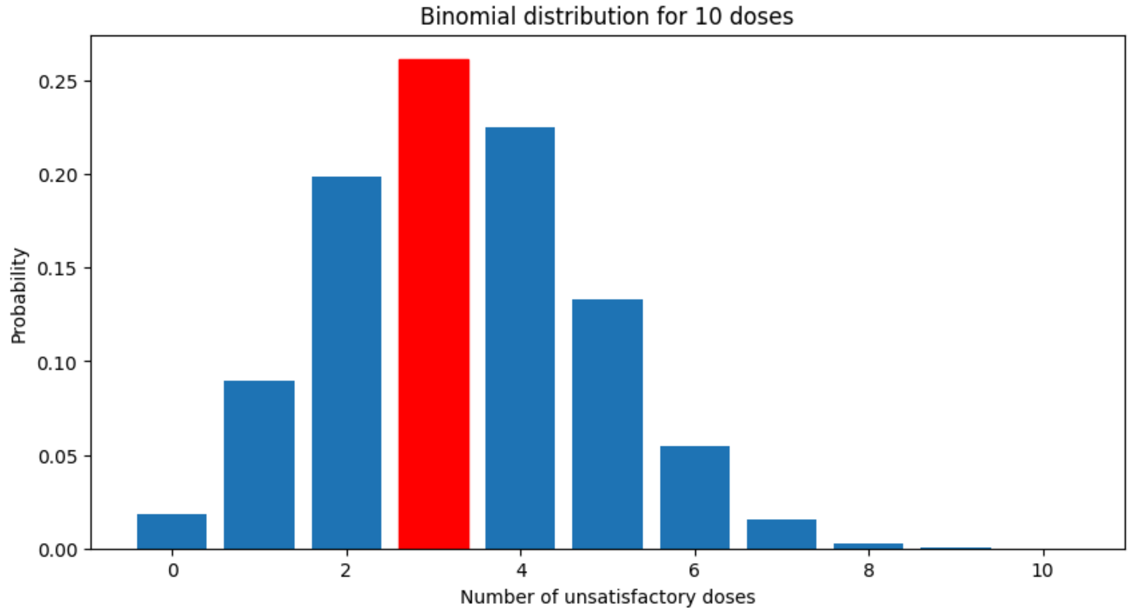
```
array([1.82283780e-02, 8.97815635e-02, 1.98993465e-01, 2.61364552e-01,
       2.25280640e-01, 1.33150945e-01, 5.46515074e-02, 1.53816609e-02,
       2.84101573e-03, 3.10956945e-04, 1.53157899e-05])
```

```
plt.figure(figsize=(10,5))
plt.bar(k,binomial_prob)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of unsatisfactory doses')
plt.ylabel('Probability')
plt.show()
```




```
# Probability that exactly 3 out of 10 doses are unsatisfactory
plt.figure(figsize=(10,5))
barl=plt.bar(k,binomial_prob)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of unsatisfactory doses')
```

```
plt.ylabel('Probability');
barl[3].set_color('r')
```



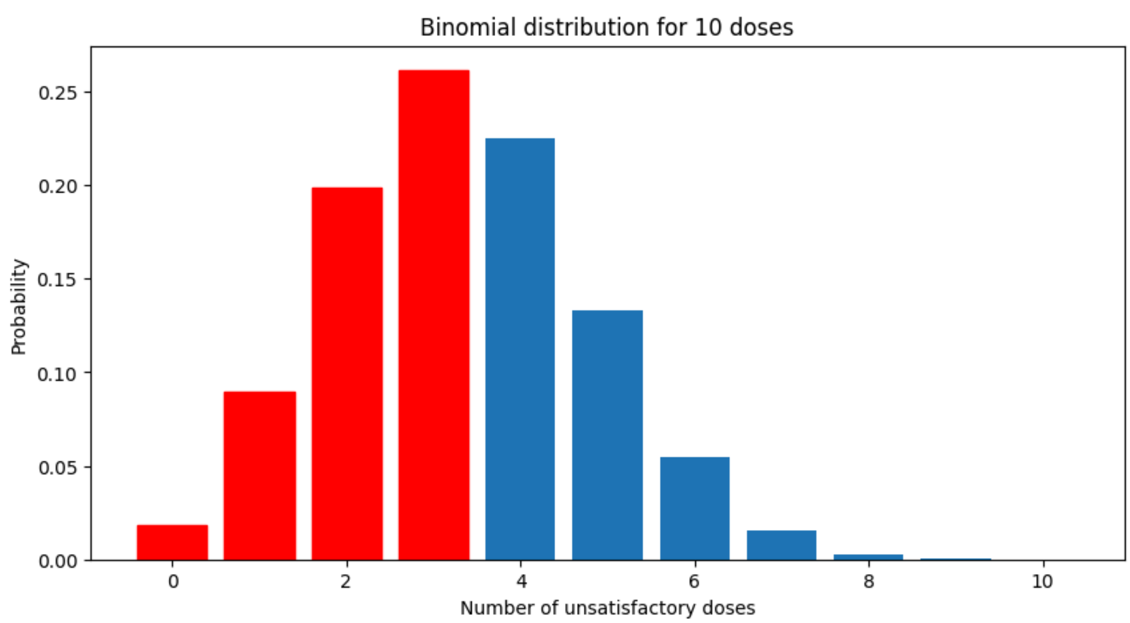

```
binom.pmf(k=3,n=n,p=p)
```

```
0.2613645515525908
```

```
binomial_prob[3]
```

```
0.2613645515525908
```

```
# Probability that at most 3 out of 10 doses are unsatisfactory
plt.figure(figsize=(10,5))
barl=plt.bar(k,binomial_prob)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of unsatisfactory doses')
plt.ylabel('Probability');
for i in range(0,4):
    barl[i].set_color('r')
plt.show()
```




```
sum(binomial_prob[:4])
```

```
0.5683679584925132
```

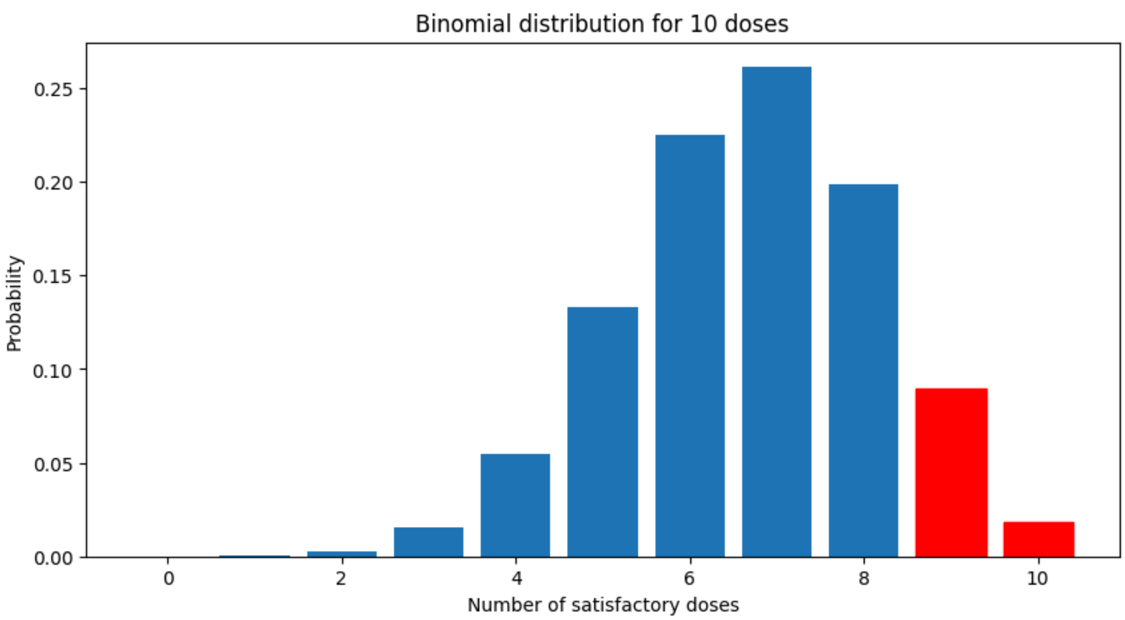
```
prob_atmost3=binom.cdf(k=3,n=n,p=p)
prob_atmost3
```

```
0.5683679584925142
```

```
# Probability that more than 8 out of 10 doses are satisfactory
p_satisfactory=1-.33
n=10
k=np.arange(0,11)
bi_prob_satis=binom.pmf(k=k,n=n,p=p_satisfactory)
bi_prob_satis
```

```
array([1.53157899e-05, 3.10956945e-04, 2.84101573e-03, 1.53816609e-02,
       5.46515074e-02, 1.33150945e-01, 2.25280640e-01, 2.61364552e-01,
       1.98993465e-01, 8.97815635e-02, 1.82283780e-02])
```

```
plt.figure(figsize=(10,5))
barl=plt.bar(k,bi_prob_satis)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of satisfactory doses')
plt.ylabel('Probability');
barl[9].set_color('r')
barl[10].set_color('r')
```




```
prob_atmost1=binom.cdf(k=8,n=n,p=1-p)
1-prob_atmost1
```

```
0.10800994155329091
```

Task 2: For 20 doses requested by the New York City administration:

- Probability that at least 11 out of 20 doses are unsatisfactory
- Probability that at most 5 out of 20 doses are unsatisfactory
- Probability that at least 13 out of 20 doses are satisfactory

Double-click (or enter) to edit

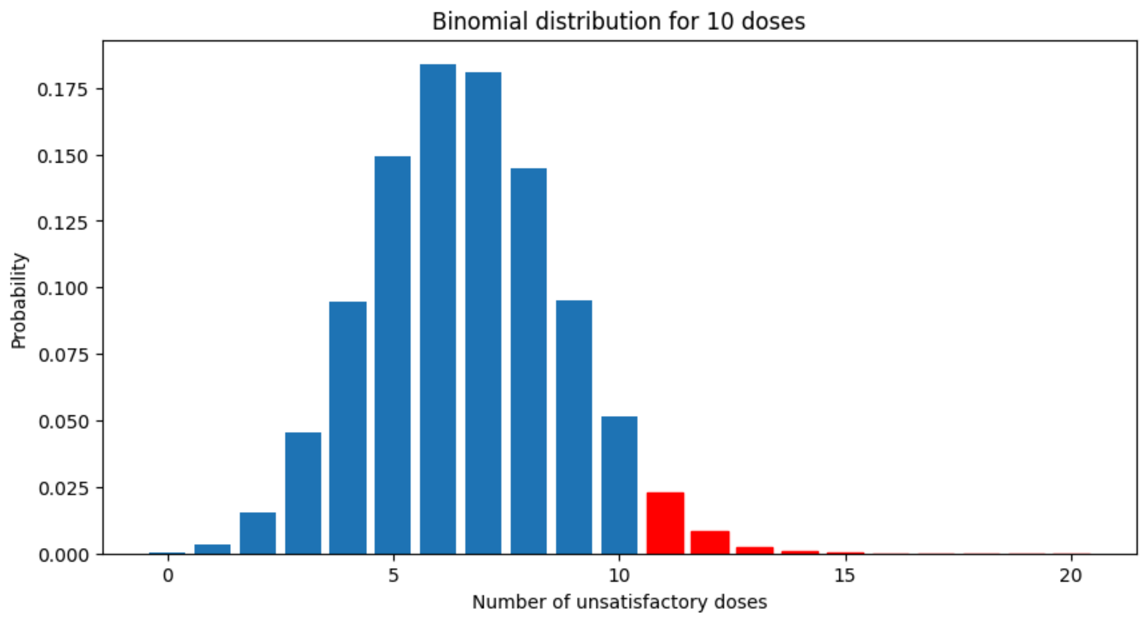
```
n=20
p=.33
k=np.arange(0,21)
k
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20])
```

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

```
array([3.32273766e-04, 3.27314456e-03, 1.53153854e-02, 4.52603926e-02,
       9.47428368e-02, 1.49326023e-01, 1.83871596e-01, 1.81127244e-01,
       1.44969380e-01, 9.52037718e-02, 5.15805510e-02, 2.30957691e-02,
       8.53164605e-03, 2.58594094e-03, 6.36836201e-04, 1.25466237e-04,
       1.93115010e-05, 2.23803172e-06, 1.83719022e-07, 9.52510246e-09,
       2.34573419e-10])
```

```
# Probability that at least 11 out of 20 doses are unsatisfactory
plt.figure(figsize=(10,5))
barl=plt.bar(k,binomial_prob)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of unsatisfactory doses')
plt.ylabel('Probability');
for i in range(11,21):
    barl[i].set_color('r')
plt.show()
```



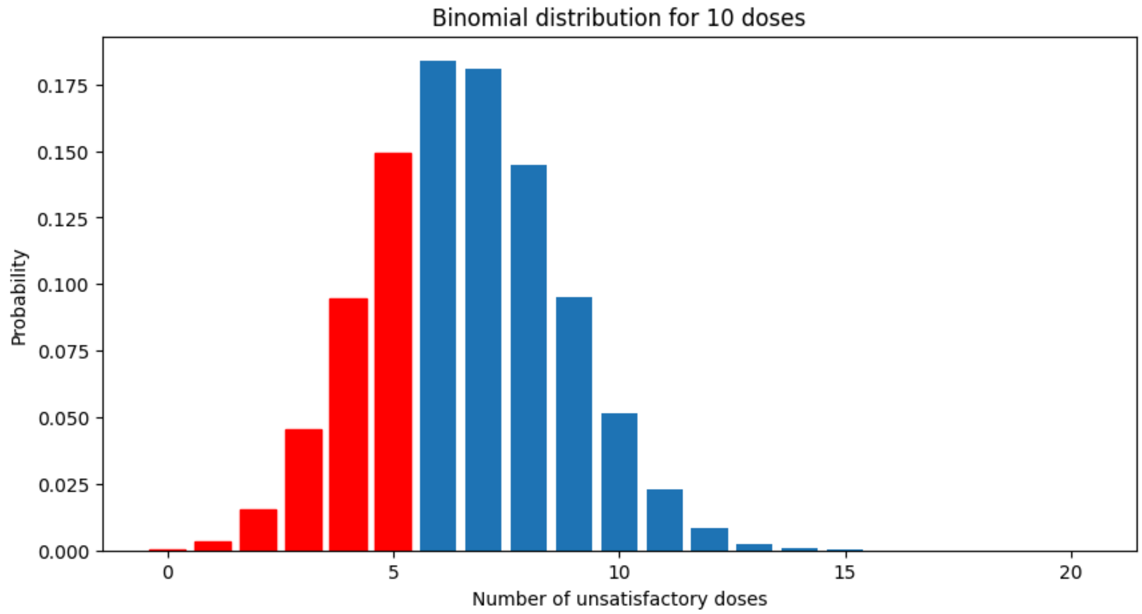

```
sum(binomial_prob[11:])
```

```
0.034997401526483854
```

```
prob_atleast1=1-binom.cdf(k=10,n=n,p=p)
prob_atleast1
```

```
0.03499740152648401
```

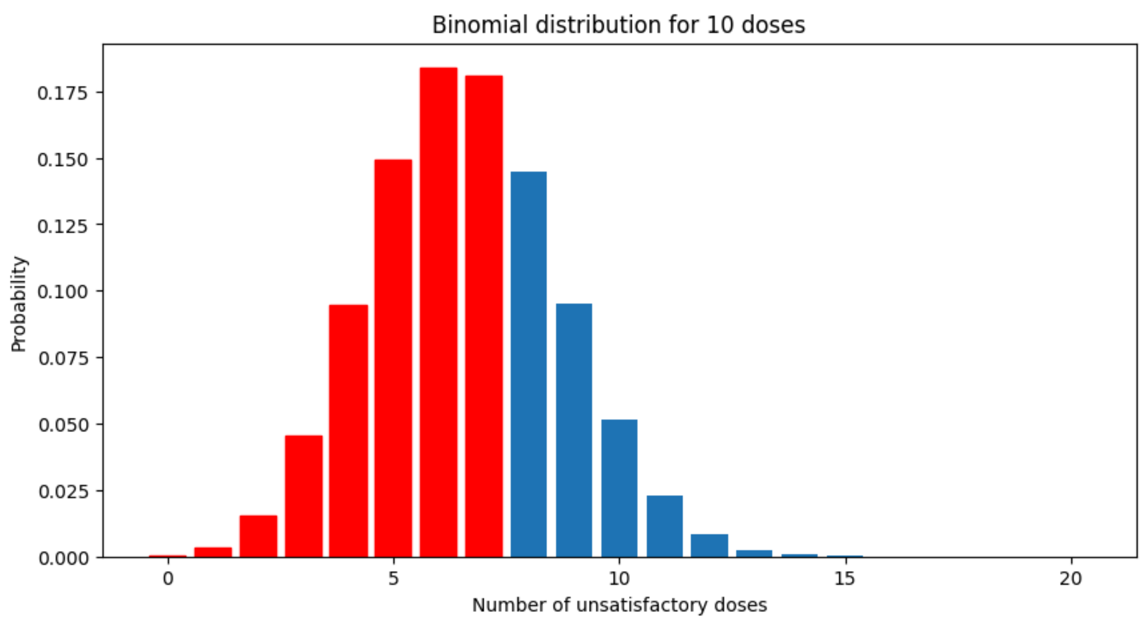
```
# Probability that at most 5 out of 20 doses are unsatisfactory
plt.figure(figsize=(10,5))
barl=plt.bar(k,binomial_prob)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of unsatisfactory doses')
plt.ylabel('Probability');
for i in range(0,6):
    barl[i].set_color('r')
plt.show()
```




```
prob_atmost5=binom.cdf(k=5,n=n,p=p)
prob_atmost5
```

```
0.30825005639386527
```

```
# Probability that at least 13 out of 20 doses are satisfactory
plt.figure(figsize=(10,5))
barl=plt.bar(k,binomial_prob)
plt.title('Binomial distribution for 10 doses')
plt.xlabel('Number of unsatisfactory doses')
plt.ylabel('Probability');
for i in range(0,8):
    barl[i].set_color('r')
plt.show()
```




```
prob_atmost6=binom.cdf(k=7,n=n,p=p)
prob_atmost6
```

```
0.6732488959678964
```

Task 3: Analyze the time of effect for 50 doses given to volunteers.

- Probability that the time of effect is less than 11.5 hours
- Probability that the time of effect is more than 10 hours
- Calculate the 90th percentile of the time of effect
- Use dataset doses.csv.

```
# Probability that the time of effect is less than 11.5 hours
df=pd.read_csv('doses.csv')
df.head()
```

	drug_serial_number	time_of_effect
0	672	5.8
1	895	17.3
2	518	16.7
3	448	13.1
4	402	13.6

```
len(df[df['time_of_effect']  <11.5])/len(df)*100
```

```
34.0
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 2 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   drug_serial_number  50 non-null     int64
 1   time_of_effect      50 non-null     float64
dtypes: float64(1), int64(1)
memory usage: 928.0 bytes
```

```
mean=df['time_of_effect'].mean()
mean
```

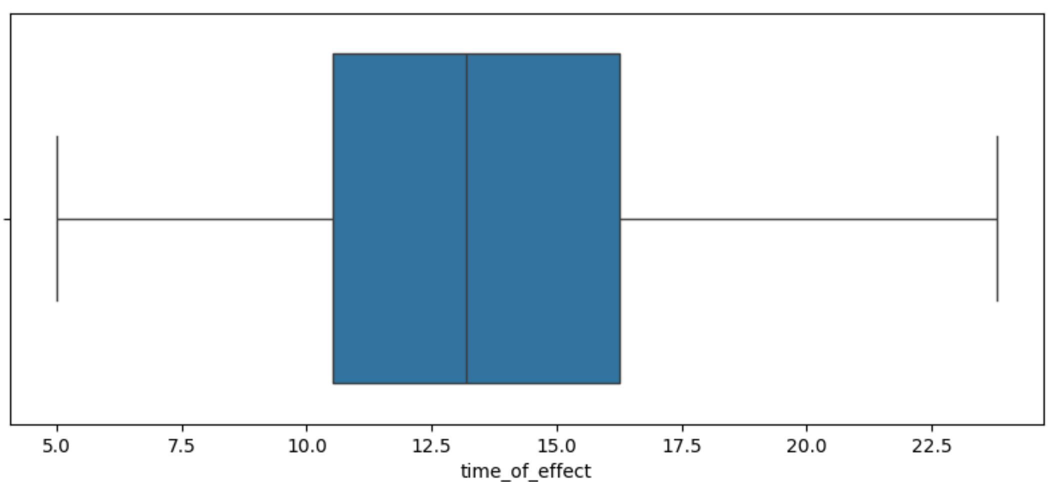
```
13.442
```

```
std=df['time_of_effect'].std()
std
```

```
4.7455280775371955
```

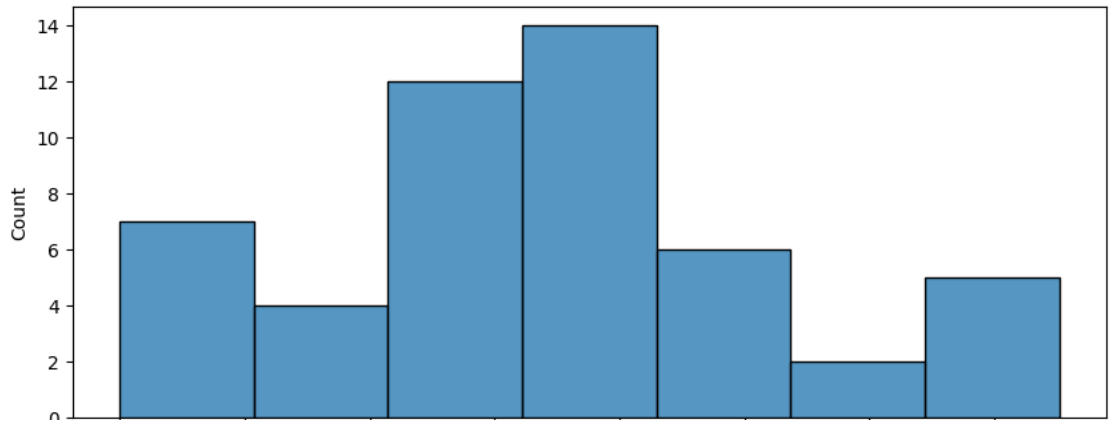
```
plt.figure(figsize=(10,4))
sns.boxplot(df['time_of_effect'],orient='h')
```

```
<Axes: xlabel='time_of_effect'>
```




```
plt.figure(figsize=(10,4))
sns.histplot(df['time_of_effect'])
```

```
<Axes: xlabel='time_of_effect', ylabel='Count'>
```




```
df['time_of_effect'].mean(), df['time_of_effect'].median()
```

```
(13.442, 13.2)
```

```
stats.norm.cdf(11.5,df['time_of_effect'].mean(),df['time_of_effect'].std())
```

```
0.3411864031732611
```

```
stats.norm.cdf(11.5,mean,std)
```

```
0.3411864031732611
```

```
1-stats.norm.cdf(10,mean,std)
```

```
0.7658704228920478
```

```
from scipy.stats import norm
norm.cdf(50,45,1.7)
```

```
0.9983651589936532
```

```
norm.ppf(0.998365,45,1.7)
```

```
49.999948792243856
```

```
norm.ppf(0.90,mean,std)
```

```
19.52363893710365
```