Continuous Distribution

> A continuous probability distribution is a probability distribution whose support is an uncountable set, such as an interval in the real line. There are many examples of continuous probability distributions: normal, uniform, chi-squared and others. - Wikipedia

Import required packages

In [5]:
```
# for latex equations
from IPython.display import Math, Latex
# for displaying images
from IPython.core.display import Image

# import matplotlib, numpy and seaborn
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

Change some settings

In [6]:
```
#for inline plots in Jupyter
%matplotlib inline
#settings for seaborn plotting style
sns.set(color_codes = True)
#settings for seaborn plot size
sns.set(rc = {'figure.figsize':(5, 5)})
```

Uniform Distribution

> In probability theory and statistics, the continuous uniform distribution or rectangular distribution is a family of symmetric probability distributions. The distribution describes an experiment where there is an arbitrary outcome that lies between parameters *a* and *b*, indicating the lower and upper bound values. The interval can either be open i.e. (a, b) or closed i.e. [a, b]. - Wikipedia

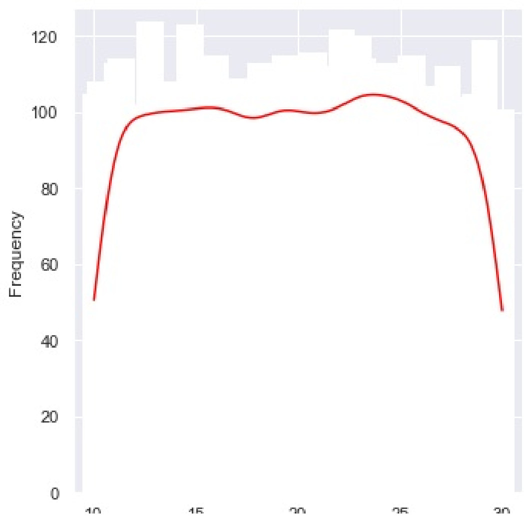
The probability distribution function (pdf) for U(a, b) is:

$$f(X = x) = \begin{cases} \frac{1}{b-a}, & \text{if x } \in [a, b] \\ 0, & \text{otherwise.} \end{cases}$$

In [7]:
```
from scipy.stats import uniform
n = 10000
start = 10
width = 20
data_uniform = uniform.rvs(size = n, loc = start, scale = width)

ax = sns.displot(data_uniform, bins = 100, kde = True, color = "red", linewidth = 15, alpha = 1)
ax.set(xlabel = "Uniform Continuous Dist.", ylabel = "Frequency")
```

Out[7]: <seaborn.axisgrid.FacetGrid at 0x1db862f66a0>



Normal Distribution

> In probability theory, a normal (or Gaussian or Gauss or Laplace-Gauss) distribution is a type of continuous probability distribution for a real valued random variable. - Wikipedia

The general form of its probability density function is

$$f(X = x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

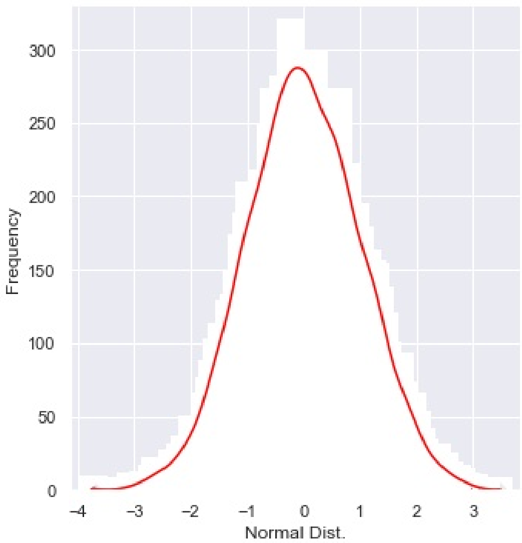
In [9]:

```python
from scipy.stats import norm
data_norm = norm.rvs(size = 10000, loc = 0, scale = 1)

ax = sns.displot(data_norm, bins = 100, kde = True, color = "red", linewidth = 15, alpha = 1)
ax.set(xlabel = "Normal Dist.", ylabel = "Frequency")
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x1db87c3e910>
```



Exponential Distribution

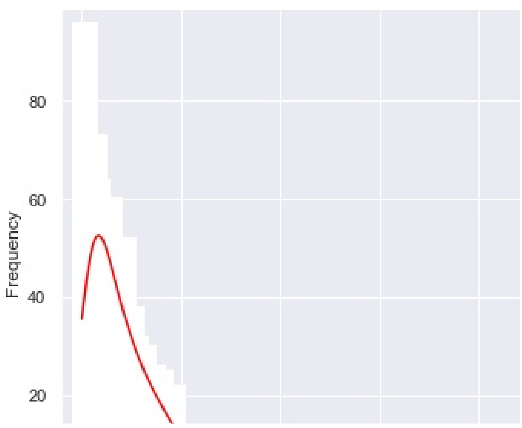
In probability theory and statistics, the exponential distribution is the probability distribution of the time between events in a Poisson point process, i.e., a process in which events occur continuously and independently at a constant average rate. It is a particular case of the gamma distribution

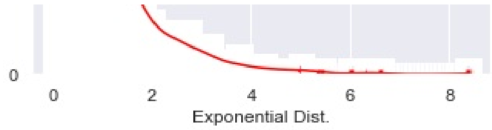
In [11]:

```python
from scipy.stats import expon
data_expon = expon.rvs(scale = 1, loc = 0, size = 1000)
ax = sns.displot(data_expon, bins = 100, kde = True, color = "red", linewidth = 15, alpha = 1)
ax.set(xlabel = "Exponential Dist.", ylabel = "Frequency")
```

Out[11]:

```
<seaborn.axisgrid.FacetGrid at 0x1db87f75eb0>
```





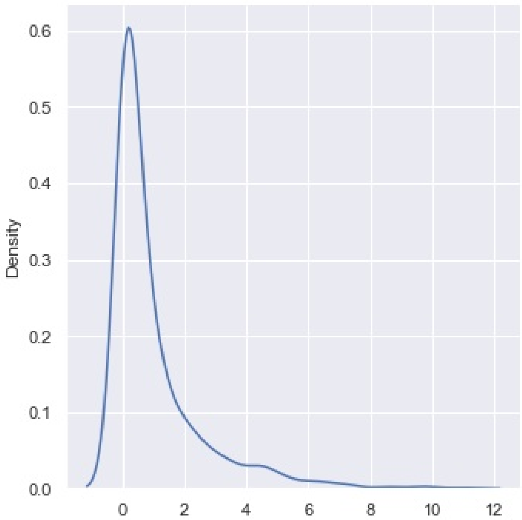
Chi-Squared distribution

In probability theory and statistics, the chi-squared distribution (also chi-square or χ2-distribution) with k degrees of freedom is the distribution of a sum of the squares of k independent standard normal random variables. The chi-squared distribution is a special case of the gamma distribution and is one of the most widely used probability distributions in inferential statistics, notably in hypothesis testing and in construction of confidence intervals

In [12]:

```
from numpy import random
x = random.chisquare(2, size = (2, 3))
print(x)
sns.displot(random.chisquare(df = 1, size = 1000), kind = "kde")
plt.show()
```

```
[[0.18064569 4.28207869 3.37997552]
 [1.92631047 1.80352806 0.47767716]]
```



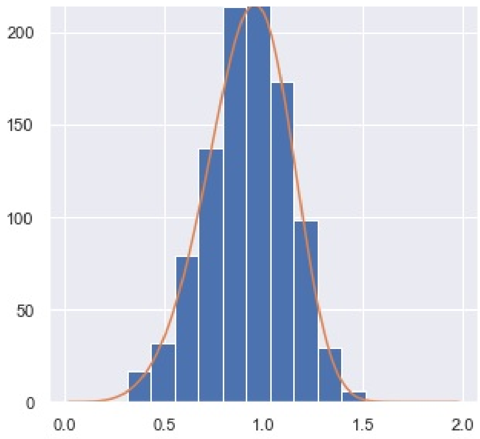
Weibull Distribution

In probability theory and statistics, the Weibull distribution is a continuous probability distribution. It is named after Swedish mathematician Waloddi Weibull, who described it in detail in 1951

In [13]:

```
a = 5
s = random.weibull(5, 10)
x = np.arange(1, 100) / 50
print(x)
def weib(x, n, a):
 return (a / n) * (x / n) ** (a - 1) * np.exp(- (x / n) ** a)
count, bins, ignored = plt.hist(np.random.weibull(5, 1000))
x = np.arange(1, 100) / 50
scale = count.max() / weib(x, 1, 5).max()
plt.plot(x, weib(x, 1, 5) * scale)
plt.show()
```

```
[0.02 0.04 0.06 0.08 0.1  0.12 0.14 0.16 0.18 0.2  0.22 0.24 0.26 0.28
 0.3  0.32 0.34 0.36 0.38 0.4  0.42 0.44 0.46 0.48 0.5  0.52 0.54 0.56
 0.58 0.6  0.62 0.64 0.66 0.68 0.7  0.72 0.74 0.76 0.78 0.8  0.82 0.84
 0.86 0.88 0.9  0.92 0.94 0.96 0.98 1.   1.02 1.04 1.06 1.08 1.1  1.12
 1.14 1.16 1.18 1.2  1.22 1.24 1.26 1.28 1.3  1.32 1.34 1.36 1.38 1.4
 1.42 1.44 1.46 1.48 1.5  1.52 1.54 1.56 1.58 1.6  1.62 1.64 1.66 1.68
 1.7  1.72 1.74 1.76 1.78 1.8  1.82 1.84 1.86 1.88 1.9  1.92 1.94 1.96
 1.98]
```



In [ ]:

Processing math: 100%