INTRODUCTION

To Data Science

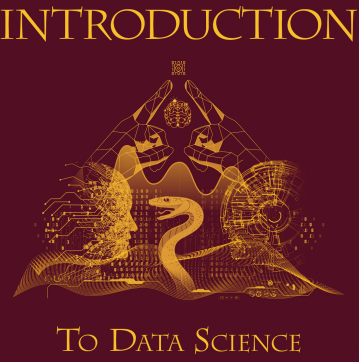
Introduction to Data Science

Inference - Part B - Class 8

Giora Simchoni

gsimchoni@gmail.com and add #intro2ds in subject

Stat. and OR Department, TAU



Last time on Hypothesis Testing

INTRODUCTION

TO DATA SCIENCE

Last time on Hypothesis Testing...

- Either calculate a **p-value** under H_0 (simulation, analytical calculation) and reject if too small (surprising)
- Or decide beforehand on a **rejection area** under H_0 for some statistic of the sample $T(X)$ and reject if $T(X)$ is in rejection area

Reality\Decision	**Not Reject H_0**	**Reject H_0**
H_0	Confidence: $1-\alpha$	Type I Error: α
H_1	Type I Error: β	Power: $1-\beta$

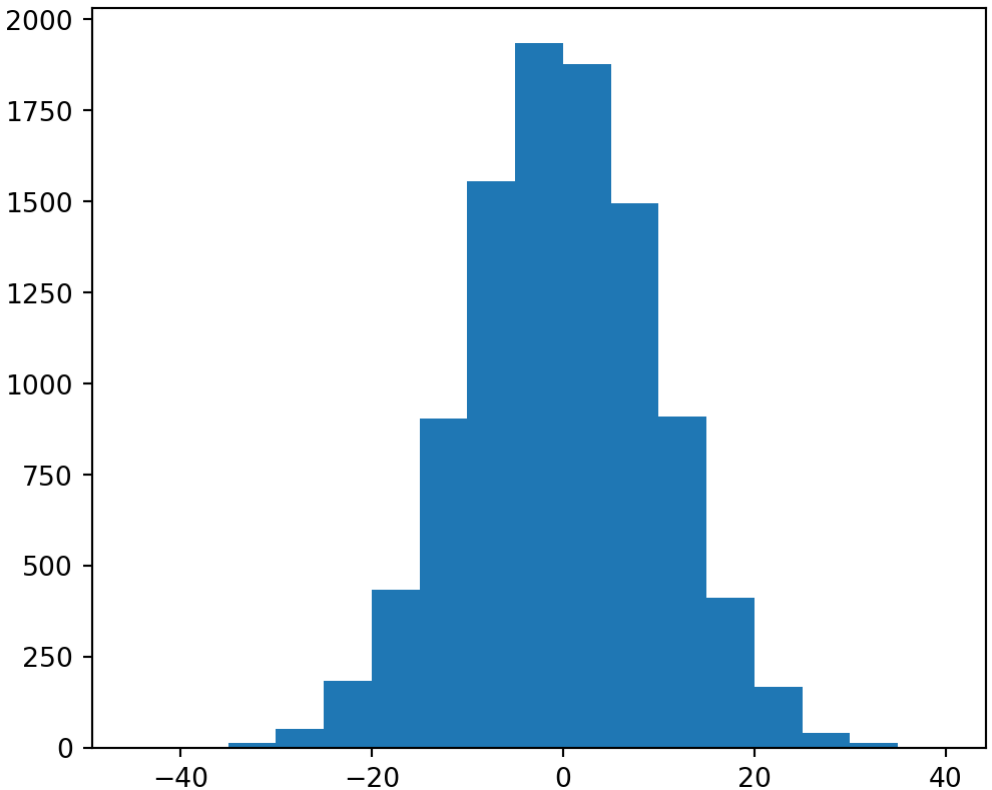
- We did both for only very simple scenarios.
- We need a way of computing these probabilities without having the population at the palm of our hand.

Back to Red Paintings

- Recall:
  - We hypothesize there is “more red” in impressionist paintings
  - The null hypothesis about the mean in differences: $H_0 : \mu = 0$
  - We only have capacity to get red pixel levels for 30 impressionist and 30 realist paintings
  - We computed the mean for our samples: impressionist paintings had ~15 points more red
  - We compared our specific difference of 15 points to a **simulated** null distribution of **means differences**

```
1 def sample_null_mean_diff(n = 30):
2     real_red_null = np.random.choice(population, n, replace=False)
3     impr_red_null = np.random.choice(population, n, replace=False)
4     return impr_red_null.mean() - real_red_null.mean()
5
6 null_mean_diffs = np.array([sample_null_mean_diff() for i in range(10000)])
```

```
plt.hist(null_mean_diffs, bins=np.arange(-45, 45, 5))
plt.show()
```



- We saw 15 points wasn't that surprising (simulated p-value = 7%) – we didn't reject H_0
- This distribution amazingly is bell shaped, "normal"
- If we could somehow trust this would always be the case we could easily calculate p-value, critical C, whatever.

The Normal Distribution

INTRODUCTION

TO DATA SCIENCE

The Normal Distribution: Refresher

- With Discrete RVs we usually talk about “Probability Mass Function” (PMF).
- With Continuous RVs we talk about “Probability Density Function” (PDF).
- If $X \sim N(\mu, \sigma^2)$ its density function is defined as:

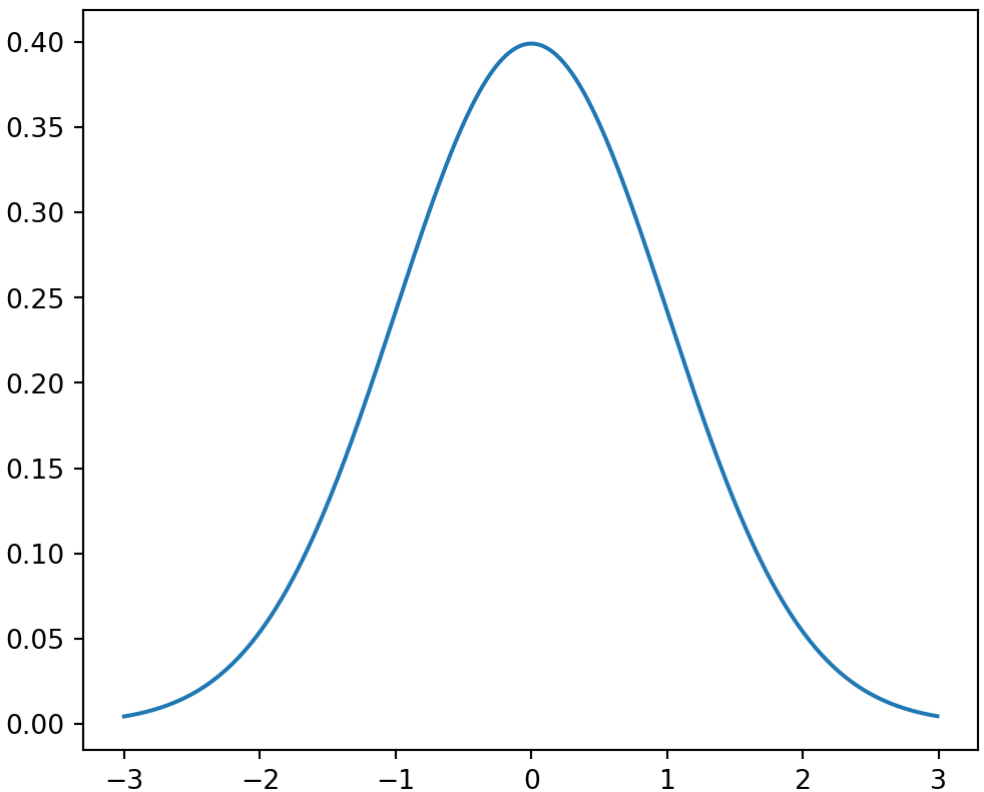
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

- If $\mu = 0$ and $\sigma = 1$ (a.k.a the Standard Normal Distribution $N(0, 1)$) it has the familiar form:

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2})$$

and the familiar bell shape around 0:

```
import scipy.stats as stats

x = np.arange(-3, 3, 0.01)
plt.plot(x, stats.norm.pdf(x, 0, 1))
plt.show()
```



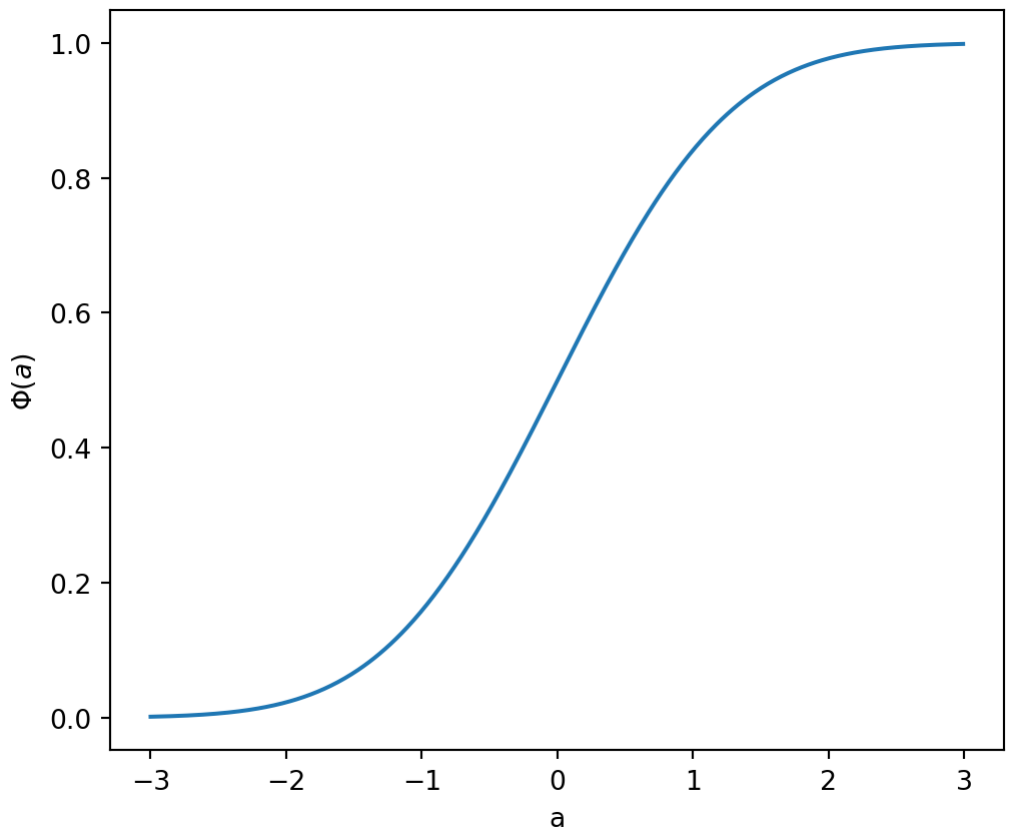
The density function $f(x)$ is a positive real function, the area under which is 1.

- But notice that $f(x)$ values are NOT probabilities but densities.
- Probabilities are *areas*. And to get those areas (probabilities) we *integrate*:

$$P(X \leq a) = \int_{-\infty}^{a} f(x)\, dx.$$

- That last function is known as the Cumulative Distribution Function (CDF), $F_X(a)$.
- It is used so much in the Standard Normal Distribution that we denote it $\Phi(a) = P(X \leq a)$ when $X \sim N(0, 1)$:

```
plt.plot(x, stats.norm.cdf(x, 0, 1))
plt.xlabel('a')
plt.ylabel('$\Phi(a)$')
plt.show()
```

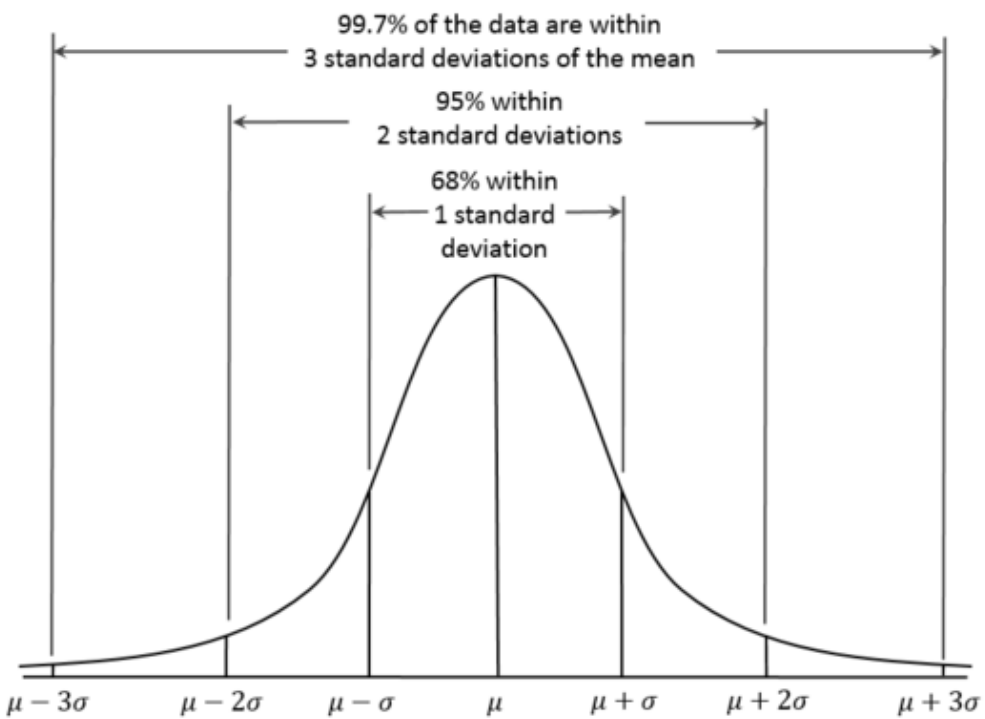


If $X \sim N(\mu, \sigma^2)$ then you can create $Z \sim N(0, 1)$ by *standardizing*:

$$Z = \frac{X - \mu}{\sigma} \sim N(0, 1)$$

μ and σ

- The pair of parameters μ and σ are enough to define a Normal distribution. The expectation is μ and standard deviation is σ.
- Morever, once a RV distributes Normal, we know roughly what percentage of the distribution is within one, two, three standard deviations off the mean, e.g. ~95% of the distribution is within 2 σs off the mean:



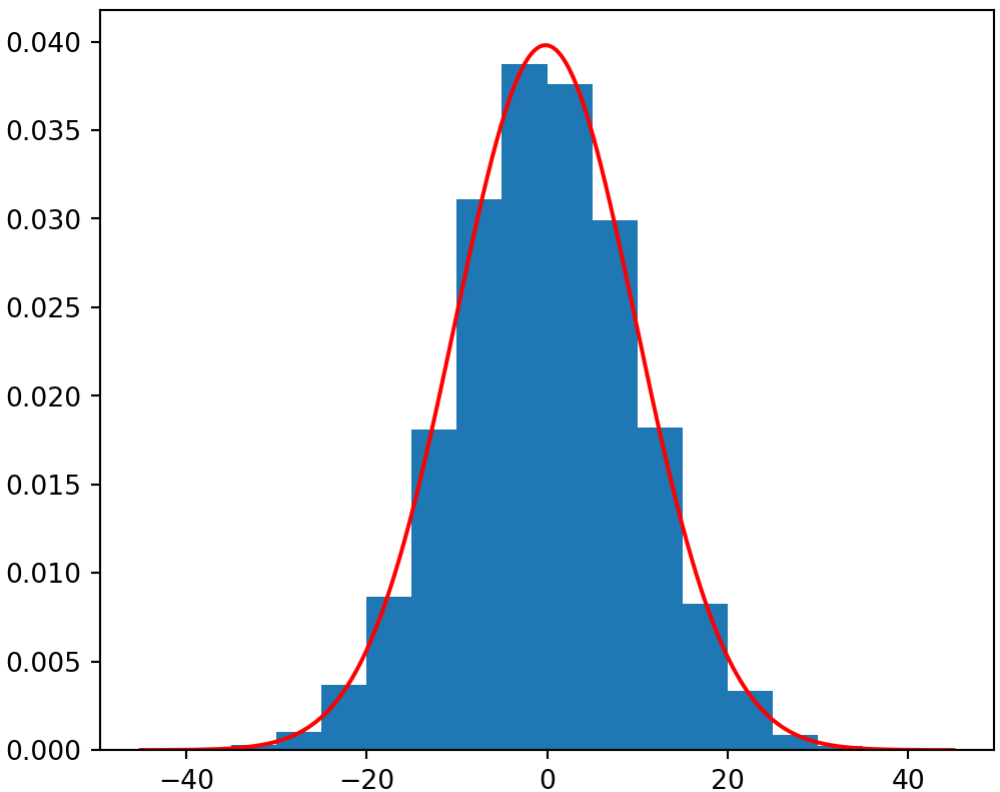
- For example, in our null distribution of mean differences, which had a bell shape to it:

```
mu = np.mean(null_mean_diffs)
sigma = np.std(null_mean_diffs)

print(f'Mean of mean differences: {mu: .2f}')
print(f'SD of mean differences: {sigma: .2f}')
```

```
Mean of mean differences: -0.18
SD of mean differences:  10.03
```

- We could fit the Normal distribution with these parameters over the (normalized) histogram, and see if the fit is “good”:

```
plt.hist(null_mean_diffs, bins = np.arange(-45, 45, 5), density = True)
x = np.arange(-45, 45, 0.01)

plt.plot(x, stats.norm.pdf(x, mu, sigma), color = 'red')
plt.show()
```



- The fit looks “good”, and so we could say that 95% of the distribution is within 2 standard deviations off the mean:

```
print(f'({mu - 2 * sigma: .2f}, {mu + 2 * sigma: .2f})')
```

```
(-20.23,  19.87)
```

- Our original samples means difference of 15 points is well within these boundaries.

Central Limit Theorem (CLT)

INTRODUCTION

TO DATA SCIENCE

Central Limit Theorem (CLT)

The CLT states that for a random sample $X_1, \ldots, X_n$ from a population with mean $E(X) = \mu$ and finite variance $V(X) = \sigma^2$, for large enough sample size n:

$$\frac{\sum_i X_i}{n} = \overline{X} \dot{\sim} N(\mu, \frac{\sigma^2}{n})$$

Or in other words:

$$\frac{\overline{X} - \mu}{\sigma/\sqrt{n}} \dot{\sim} N(0, 1)$$

- $\frac{\sigma}{\sqrt{n}}$ is called the Standard Error (SE) of the mean
- $Z = \frac{\overline{X} - \mu}{\sigma/\sqrt{n}}$ is the Z statistic

This has far reaching implications.

Intro to Data Science

Bernoulli Example

- Assume $X_1, \ldots, X_n$ are a random sample, with $X_i \sim Ber(p = 0.5)$.
- Then $E(X_i) = p = 0.5$ and $Var(X_i) = p(1-p) = 0.25$.
- We know: $\sum_i X_i \sim Bin(n, p = 0.5)$.
- The CLT tells us:

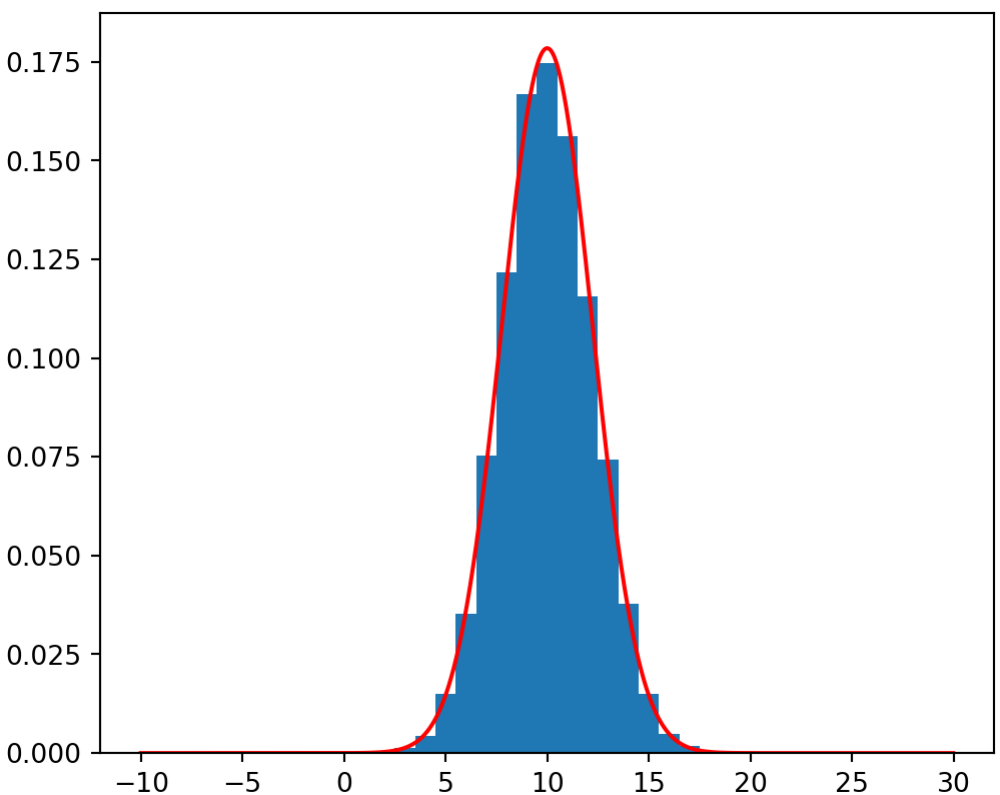
$$\frac{\sum_i X_i}{n} = \overline{X} \dot{\sim} N(0.5, \frac{0.25}{n}),$$

- or:

$$\sum_i X_i = n\overline{X} \dot{\sim} N(0.5n, 0.25n),$$

  if n is big enough.
- So: $N(0.5n, 0.25n) \approx Bin(n, 0.5)$

```
null_res = np.random.binomial(20, 0.5, size=10000)

bin_mean = 20 * 0.5
bin_sd = np.sqrt(20 * 0.5 * 0.5)

N, bins, patches = plt.hist(null_res, bins=np.arange(-0.5, 20.5, 1),density = True)
x = np.arange(-10, 30, 0.01)
plt.plot(x, stats.norm.pdf(x, bin_mean, bin_sd), color = 'red')
plt.show()
```



Exponential Distribution Example

- The Exponential Distribution is another well researched continuous distribution. $X \sim Exp(\lambda)$:

$Supp(X) = [0, \infty)$ (also $\lambda > 0$)

$f(X) = \lambda e^{-\lambda x}$

$F_x(k) = P(X \leq k) = 1 - e^{-\lambda x}$

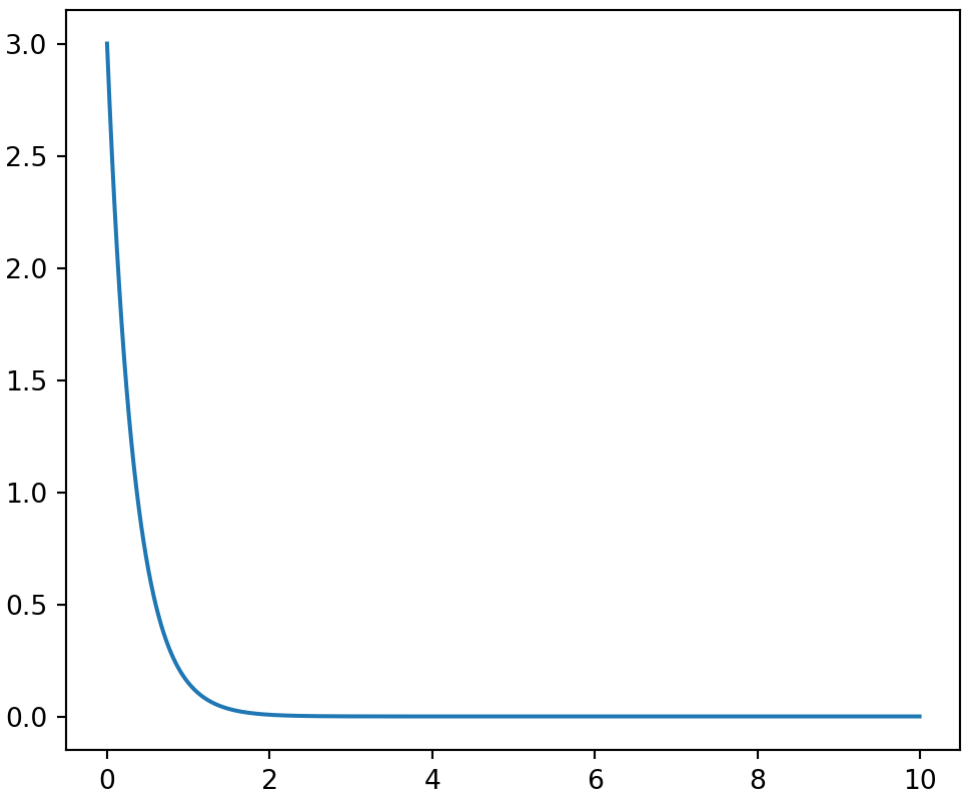
$E(X) = \frac{1}{\lambda}$

$Var(X) = \frac{1}{\lambda^2}$

- Example: X is the time between two trains from Tel-Aviv to Haifa which come on average every 20 minutes (1/3 hour). So: $X \sim Exp(3)$

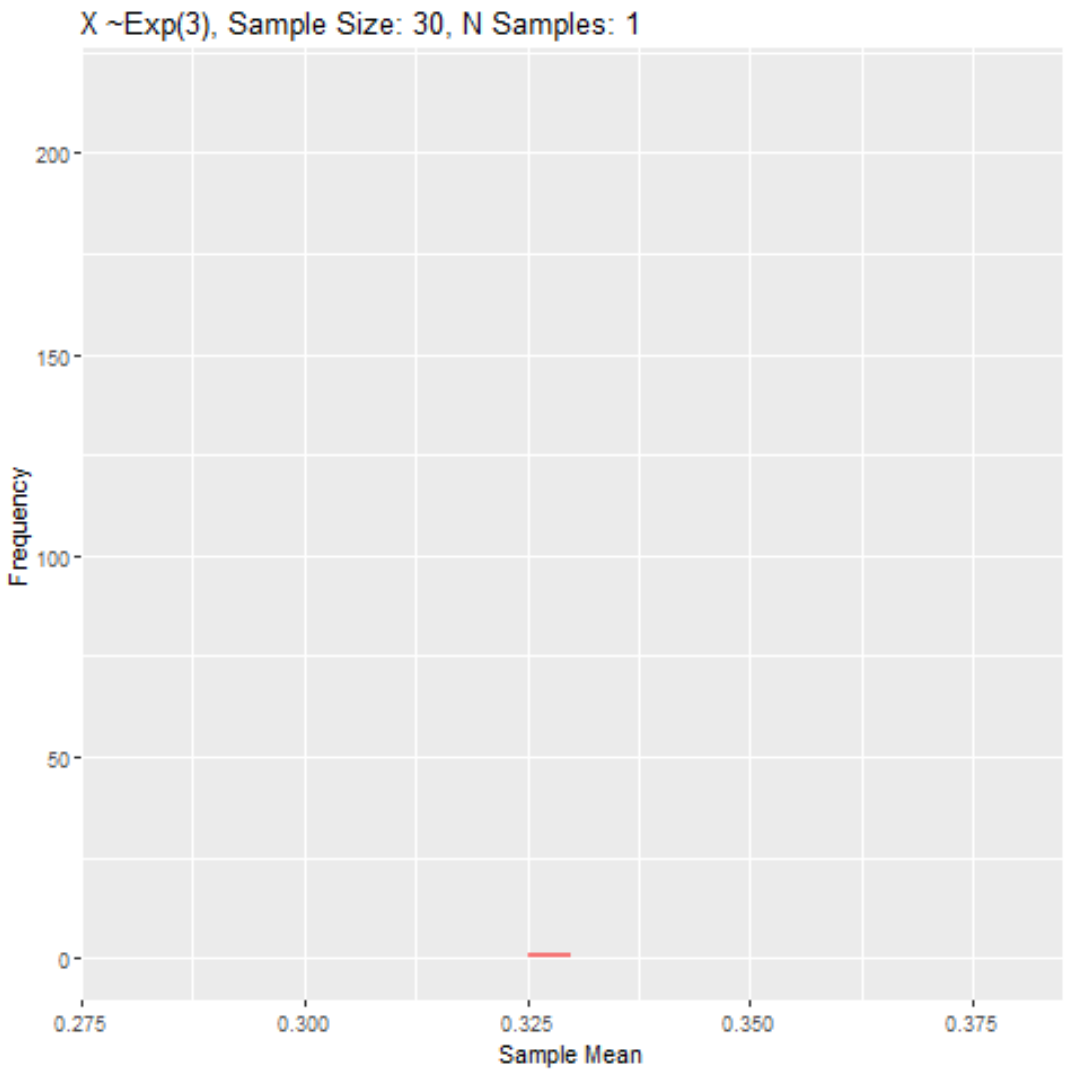
How it looks (definitely not normal):

```
x = np.arange(0, 10, 0.01)
plt.plot(x, stats.expon.pdf(x, scale = 1/3))
plt.show()
```



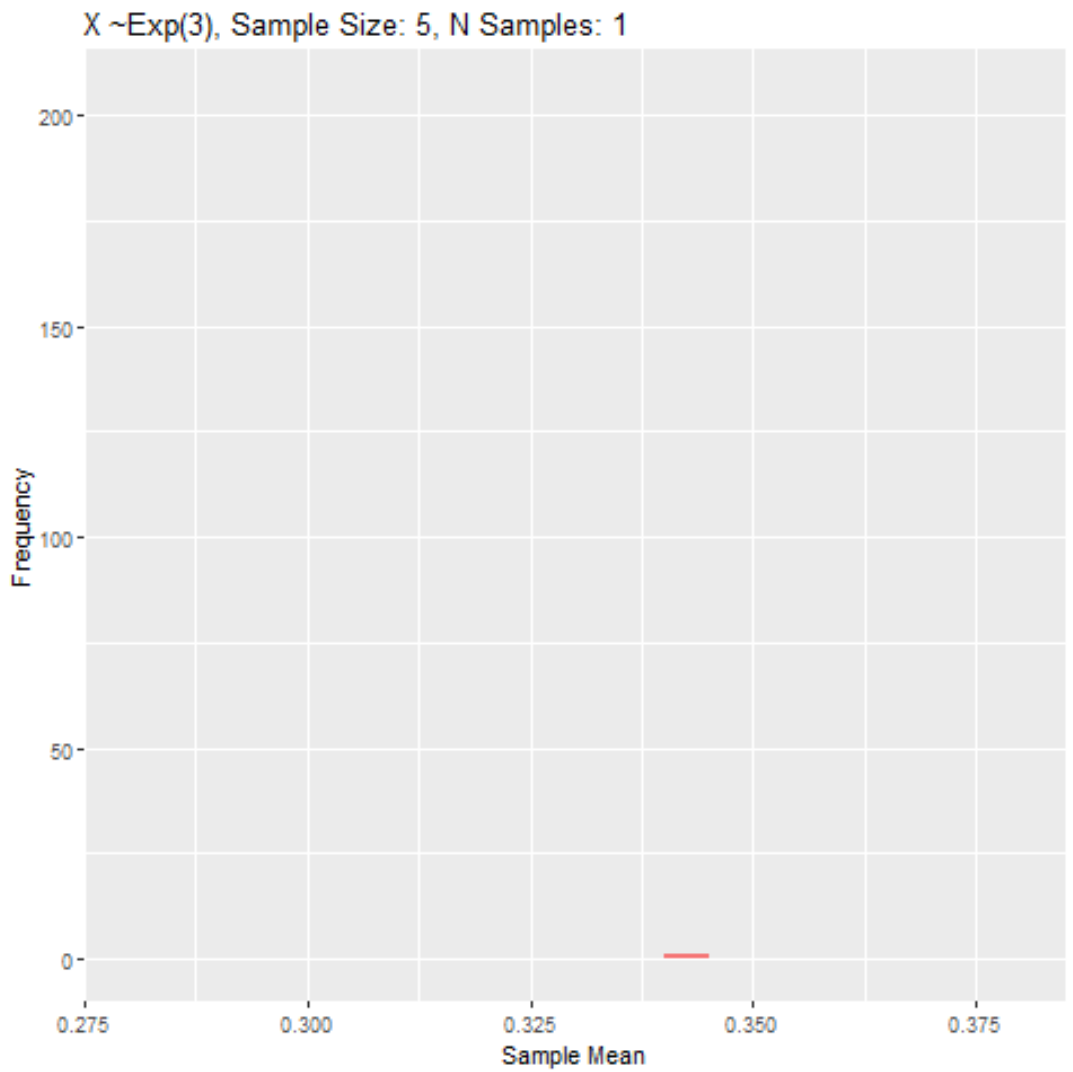
Exponential Distribution Demo

A demo where $X \sim Exp(3)$, sampling distribution of the mean with $n = 30$:



Exponential Distribution Demo

A demo where $X \sim Exp(3)$, sampling distribution of the mean with $n = 5$:



Z-Test

INTRODUCTION

To Data Science

Z-Test

- The trains from Tel-Aviv to Haifa come every 20 minutes? (null hypothesis)
- Lately it seems like a lot more (one-sided alternative hypothesis)

- I randomly sampled 30 waiting times between two trains, and got an average $\overline{X} = 4/9$ or 26 minutes and 40 seconds.
- Under the null hypothesis, according to CLT: $\overline{X} \dot{\sim} N(\frac{1}{3}, \frac{1/9}{30})$
- Or: $\frac{\overline{X} - 1/3}{1/\sqrt{270}} \dot{\sim} N(0, 1)$
- We got: $\overline{X} = 4/9$, or $Z = \frac{4/9 - 1/3}{1/\sqrt{270}} = 1.8257$
- We can compute a p-value!

```
one_sided_p_value = 1 - stats.norm.cdf(4/9, 1/3, np.sqrt(1/270))
#or
one_sided_p_value = 1 - stats.norm.cdf(1.8257, 0, 1)

print(f'P(X_bar >= 4/9 | H0) = {one_sided_p_value: .2f}')
```

```
P(X_bar >= 4/9 | H0) =  0.03
```

- And this looks pretty extreme (lower than 5%), and we “reject the null hypothesis” and conclude that indeed it seems like the waiting time has increased.

Important question: what if $\overline{X} = 2/9$?

Back to Red Paintings

- Let X be the red pixel level of impressionist paintings images.
- Let Y be the red pixel level of realist paintings images.
- (They don't have a normal distribution, which is fine).

- Under H_0 they both come from the same distribution with $E(X) = E(Y) = \mu$ and $Var(X) = Var(Y) = \sigma^2$.
- Sample size was $n = 30$ for both independent samples.

Null distribution of the difference in means

- So under H_0, according to CLT, the sampling distribution for both samples means is Normal: $\overline{X} \dot{\sim} N(\mu, \frac{\sigma^2}{n}), \overline{Y} \dot{\sim} N(\mu, \frac{\sigma^2}{n})$
- We were interested in the distribution of the means differences, which we now know should be approximately Normal:

$$\overline{X} - \overline{Y} \dot{\sim} N(0, \frac{\sigma^2}{n} + \frac{\sigma^2}{n})$$

(make sure you understand why!)

- In other words, under H_0:

$$\frac{(\overline{X} - \overline{Y}) - 0}{\sqrt{\frac{2\sigma^2}{n}}} \dot{\sim} N(0, 1)$$

- Only one thing is missing: we do not know what σ is (Do we know what μ is? Why is that not a problem?)

Intro to Data Science

Impressionist and Realist Paintings - Z-Test

- Let's assume for now we estimate it from the population: σ = `np.std(population) = 39.3`.
- So under H_0: $\overline{X} - \overline{Y} \dot{\sim} N(0, \frac{2 \cdot 39.3^2}{30} = 10.15^2)$
- Or: $\frac{(\overline{X} - \overline{Y}) - 0}{10.15} \dot{\sim} N(0, 1)$
- And we got: $\overline{X} - \overline{Y} = 15$, or $Z = \frac{15}{10.15} = 1.48$
- And we can perform a Z-Test and compute the p-value:

```
one_sided_p_value = 1 - stats.norm.cdf(15, 0, 10.15)
#or
one_sided_p_value = 1 - stats.norm.cdf(1.48, 0, 1)

print(f'P(mean_diff >= 15 | H0) = {one_sided_p_value: .2f}')
```

```
P(mean_diff >= 15 | H0) =  0.07
```

- Which is similar to the result we got using our "known population".

T-Test(s)

INTRODUCTION

To Data Science

From Z-Test to T-Test

But σ^2 isn't known!

- Solution: replace the unknwon σ^2 by the unbiased estimator $S^2 = \frac{1}{n-1}\sum(X_i - \overline{X})^2$
- For a Standard Normal RV $Z = \frac{\overline{X}-\mu}{\sigma/\sqrt{n}}$, if we replace the unknown σ by S, we get a new distribution called **Student's t** distribution:

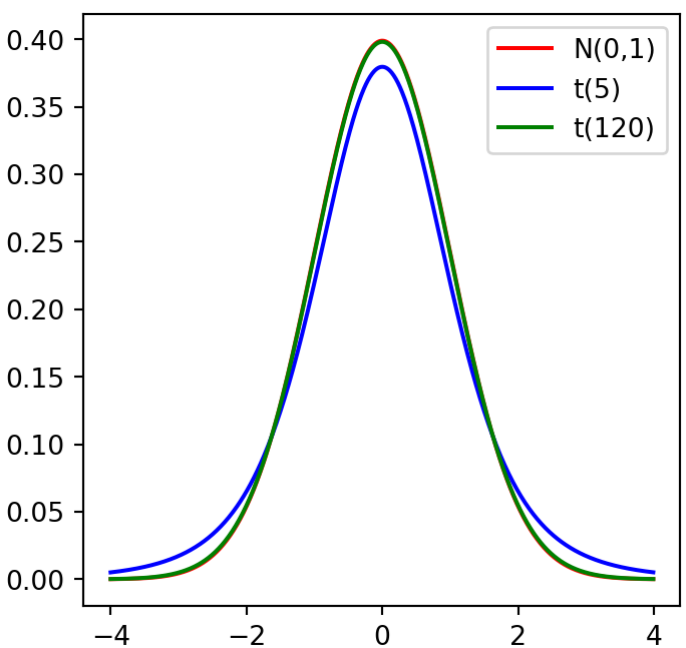
$$T = Z\frac{\sigma}{S} = \frac{\overline{X}-\mu}{S/\sqrt{n}} \sim t_{n-1}$$

- This statistic distributes "t with $n-1$ degrees of freedoms (df)", hence called the T statistic.

The t distribution has “longer tails” than the Standard Normal distribution for small n, reflecting the added uncertainty once σ isn’t known but estimated by S.

But for $n \geq 120$ it is similar to the Standard Normal:

```
x = np.arange(-4, 4, 0.01)
plt.figure(figsize=(4, 4))
plt.plot(x, stats.norm.pdf(x, 0, 1), color = 'red')
plt.plot(x, stats.t.pdf(x, 5), color = 'blue')
plt.plot(x, stats.t.pdf(x, 120), color = 'green')
plt.gca().legend(('N(0,1)','t(5)', 't(120)'))
plt.show()
```



One Sample T-Test

- Again: Under H_0, with a large sample n, according to CLT: $\frac{\overline{X}-\mu}{\sigma/\sqrt{n}} \sim N(0,1)$
- Now the σ is unknown. μ is known under the null hypothesis, often $H_0 : \mu = 0$
- We replace σ by its estimator: $S = \sqrt{\frac{1}{n_x - 1}\sum (X_i - \overline{X})^2}$
- We get:

$$T = \frac{\overline{X} - \mu_{H0}}{\sqrt{\frac{S^2}{n}}} \sim_{H_0} t_{n-1}$$

- And you can perform a t-test using e.g. `stats.ttest_1samp(x, 0)`.

Two Independent Samples T-Test

- In our kind of problems we have two samples: $X_1, \ldots, X_{n_x}$ and $Y_1, \ldots, Y_{n_y}$. In our case also $n_x = n_y = 30$.
- Our null hypothesis of interest is $H_0 : \mu_x = \mu_y$. We will also assume $\sigma_X^2 = \sigma_Y^2$.
- Now, **assuming the variances are equal** we can use the CLT and write that under H_0:

$$\frac{(\overline{X} - \overline{Y}) - (\mu_x - \mu_y)}{\sqrt{\sigma^2(\frac{1}{n_x} + \frac{1}{n_y})}} \sim N(0,1)$$

- Since we do no know σ^2, but we assume it is equal for both groups under the null, we estimate it: $S_p^2 = \frac{(n_x-1)S_x^2+(n_y-1)S_y^2}{n_x+n_y-2}$
- And the distribution is t with $n_x + n_y - 2$ degrees of freedom:

$$T = \frac{\overline{X}-\overline{Y}-(\mu_x-\mu_y)}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} \sim t_{n_x+n_y-2}$$

Intro to Data Science

Unequal Variances Assumed (extra credit)

Can still use the CLT and write that under H_0:

$$\frac{(\overline{X}-\overline{Y})-(\mu_x-\mu_y)}{\sqrt{\frac{\sigma_X^2}{n_x}+\frac{\sigma_Y^2}{n_y}}} \sim N(0,1)$$

We replace σ_x by its estimator: $S_x = \sqrt{\frac{1}{n_x-1}\sum(X_i-\overline{X})^2}$ and the same for σ_y.

We keep the SE estimator of $\sqrt{\frac{S_x^2}{n_x}+\frac{S_y^2}{n_y}}$:

$$\frac{\overline{X}-\overline{Y}-(\mu_x-\mu_y)}{\sqrt{\frac{S_x^2}{n_x}+\frac{S_y^2}{n_y}}} \sim t_{df'}$$

Where: $df' = \frac{\left(\frac{S_x^2}{n_x}+\frac{S_y^2}{n_y}\right)^2}{\frac{\left(S_x^2/n_x\right)^2}{n_x-1}+\frac{\left(S_y^2/n_2\right)^2}{n_y-1}}$

Intro to Data Science

Impressionist and Realist Paintings - T-Test

- Recall the null hypothesis: $H_0 : \mu_x = \mu_y$
- Recall our samples:

```
print(real_red[:10])
print(impr_red[:10])
```

```
[161.3162 147.0798 140.2261 122.5448 191.0334  52.9117  96.4099 110.3566
 171.9048  78.3864]
[ 51.7256  99.2127  95.8073 105.3173  81.8901 125.3137 128.208   78.7658
  81.1359 201.9115]
```

- First manually:

```
S2_x = np.var(impr_red, ddof = 1) # for getting the "n - 1" unbiased estimator of sigma
S2_y = np.var(real_red, ddof = 1)
n_x = n_y = 30
S2_p = ((n_x - 1)*S2_x + (n_y - 1)*S2_y) / (n_x + n_y - 2)

t_statistic = (np.mean(impr_red) - np.mean(real_red))/np.sqrt(S2_p/n_x + S2_p/n_y)
print(f'T statistic: {t_statistic: .4f}')

one_sided_p_value = 1 - stats.t.cdf(t_statistic, n_x + n_y - 2)
print(f'P(mean_diff >= 15 | H0) = {one_sided_p_value: .3f}')
```

```
T statistic:  1.3854
P(mean_diff >= 15 | H0) =  0.086
```

- Now with Python's built-in function:

```
stats.ttest_ind(impr_red, real_red, alternative='greater')
```

```
Ttest_indResult(statistic=1.385446057741497, pvalue=0.08561058232473277)
```

- Anyway, p-value of 8.5% isn't convincing and we do not reject the null hypothesis.

- Notice the T-test p-value of 8.5% is greater than the Z-test p-value of 7%, which makes sense:
- **We gave up the known variance assumption, added uncertainty (the SD estimate S is a RV), got the t distribution with "heavier" tails, need more extreme values to impress.**

Confidence Intervals (CI)

INTRODUCTION

TO DATA SCIENCE

Confidence Intervals (CI)

- One (of a few) problem with p-value: it is not informative.
- With large enough sample size any result can become “significant” (why?).
- Statistical significance is not scientific significance.

- Often we would prefer reporting what the actual mean μ (or means difference $\mu_x - \mu_y$) was, to show it was “interesting”.
- But since our sample mean $\overline{X}$ (or sample means difference $\overline{X} - \overline{Y}$) is most probably WRONG, it comes with uncertainty, we report a Confidence Interval:
- “μ is within $\left[\overline{X} - \epsilon, \overline{X} + \epsilon\right]$ with 95% level of confidence”

Building the CI: Z-Test

- In general: for a random sample X from distribution with unknown parameter θ, $[LB(X), UB(X)]$ is a $100(1-\alpha)\%$ CI for θ if $P(LB(X) < \theta < UB(X)) = 1 - \alpha$
- Usually: $\alpha = 0.05$ and we would build a *symmetric* CI around some $\hat{X}$ estimator of θ, like $\overline{X}$
- For the Z-test:

$$P(Z_{0.025} < \frac{\overline{X} - \mu}{\sigma/\sqrt{n}} < Z_{0.975}) = P(\overline{X} + Z_{0.025}\frac{\sigma}{\sqrt{n}} < \mu < \overline{X} + Z_{0.975}\frac{\sigma}{\sqrt{n}}) = 0.95$$

where Z_q is the qth quantile (100q precentile) of the $N(0,1)$ distribution

$$\implies \left[\overline{X} + Z_{0.025}\frac{\sigma}{\sqrt{n}}\ ,\ \overline{X} + Z_{0.975}\frac{\sigma}{\sqrt{n}}\right] \text{ or } \overline{X} \pm 1.96\frac{\sigma}{\sqrt{n}} \text{ is a 95\% CI for } \mu$$

CI for Waiting Time for Train

- Recall: Under H_0 $X \sim Exp(3)$ so: $\sigma = \frac{1}{3}$
- We got: $\overline{X} = \frac{4}{9}$; $n = 30$;
- 95% CI for μ:

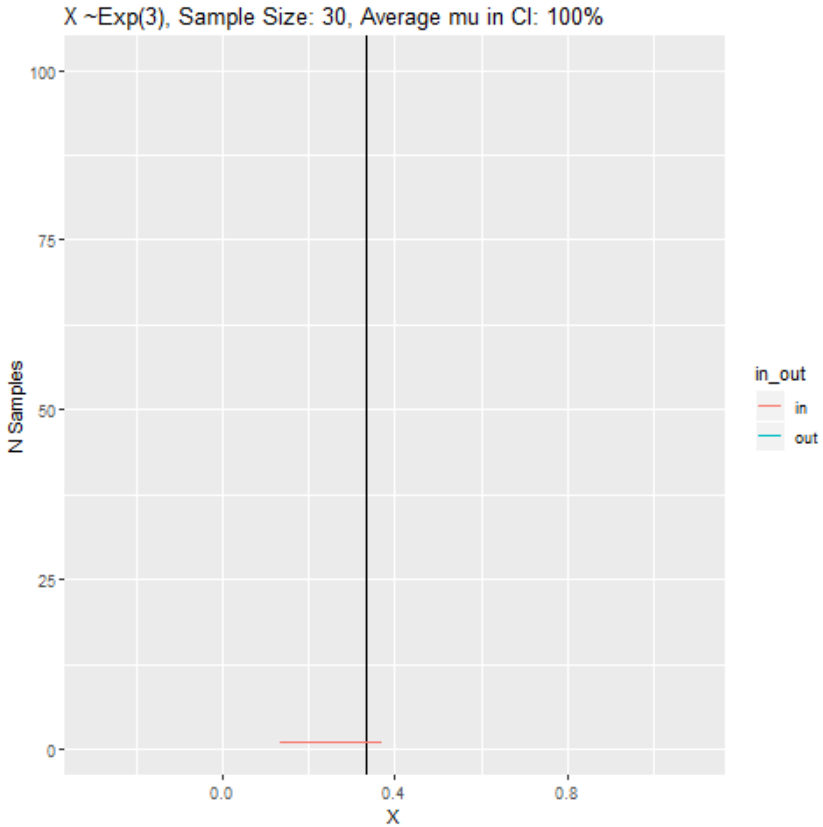
```
LB = 4/9 - stats.norm.ppf(0.975) * (1/3)/np.sqrt(30)
UB = 4/9 + stats.norm.ppf(0.975) * (1/3)/np.sqrt(30)
print(f'[{LB: .3f}, {UB: .3f}]')
```

```
[ 0.325,  0.564]
```

Meaning of CI

- Most common misconception of CI: “The probability that μ is within [0.33, 0.56] is 95%”
- **The number** μ is either within the CI or not, μ is a parameter, not a RV!
- But what *does* it mean?

A demo where $X \sim Exp(3)$, confidence interval for the mean with $n = 30$:


X ~Exp(3), Sample Size: 30, Average mu in CI: 100%
100
75
50
25
0
N Samples
0.0
0.4
0.8
X
in_out
in
out

Building the CI: T-Test

- For One-Sample T-Test:

$$\left[\overline{X} + t_{n-1;0.025}\ \frac{S}{\sqrt{n}}, \overline{X} + t_{n-1;0.975}\ \frac{S}{\sqrt{n}}\right] \text{ or}$$

$$\overline{X} \pm t_{n-1,0.975} \cdot \frac{S}{\sqrt{n}}$$

is a 95% CI for μ

- For Two-Samples T-Test, equal variances assumed:

$$\left[(\overline{X} - \overline{Y}) + t_{n_x+n_y-2;0.025}\ \sqrt{\frac{S_p^2}{n_x} + \frac{S_p^2}{n_y}}\ ,\ (\overline{X} - \overline{Y}) + t_{n_x+n_y-2;0.975}\ \sqrt{\frac{S_p^2}{n_x} + \frac{S_p^2}{n_y}}\right] \text{ or}$$

$$(\overline{X} - \overline{Y}) \pm t_{n_x+n_y-2,0.975}\ \sqrt{\frac{S_p^2}{n_x} + \frac{S_p^2}{n_y}}$$

is a 95% CI for $\mu_x - \mu_y$

Intro to Data Science

Impressionist and Realist Paintings - CI for Means Difference

Recall we got: $\overline{X} - \overline{Y} = 15;\ n_x = 30;\ n_y = 30; S_p^2 = 1824$

95% CI for $\mu_x - \mu_y$:

```
LB = (np.mean(impr_red) - np.mean(real_red)) - stats.t.ppf(0.975, n_x + n_y - 2) * np.s
UB = (np.mean(impr_red) - np.mean(real_red)) + stats.t.ppf(0.975, n_x + n_y - 2) * np.s

print(f'[{LB: .1f}, {UB: .1f}]')
```

```
[-6.8,  37.4]
```

Power

INTRODUCTION

TO DATA SCIENCE

Power

- If we take the “true” means difference of the red level pixel in impressionst paintings “population” vs. realist:

```
true_mean_diff = np.mean(impr_red_all) - np.mean(real_red_all)
print(f'True means difference: {true_mean_diff: .2f}')
```

```
True means difference:  15.35
```

- Surprise: it *is* in fact higher, by 15 points.
- Let’s assume for the time being that this is “interesting” (is it, though?)
- It turns out the null hypothesis, which we have not rejected, was wrong, impressionist paintings *are* “redder”
- Why did we fail to reject the null hypothesis, and would likely fail again?
- We lacked **Statistical Power**: the probability of correctly rejecting the null hypothesis when the null is false, when the alternative is actually the case.

Impressionist and Realist Paintings - Power Analysis - Simulation

$\pi = P(\text{reject} H_0 | H_1 \text{ true}) = ?$

Let's simulate before we compute. We have the "population", we can just take many impressionist and realist $n = 30$ samples, perform a T-test and see the percentage of times we reject the null hypothesis, i.e. p-value $< \alpha$:

```
def random_sample_t_test(alpha, n):
    real_red_sample = np.random.choice(real_red_all, n, replace=False)
    impr_red_sample = np.random.choice(impr_red_all, n, replace=False)
    t_test = stats.ttest_ind(impr_red_sample, real_red_sample, alternative='greater')
    return (impr_red_sample.mean() > real_red_sample.mean() and t_test[1] < alpha)

n_simulations = 10000
n = 30
alpha = 0.05

rejections = [random_sample_t_test(alpha, n) for i in range(n_simulations)]

print(f'Power = P(reject H0 | H1 ) = {np.mean(rejections): .2f}')
```

```
Power = P(reject H0 | H1 ) =  0.45
```

With 45% power, it seems only about 1 in 2 samples would have caught our 15 points difference and reject H_0!

Impressionist and Realist Paintings - Power Analysis - Computation

- In a realistic situation we do not know that H_0 should be rejected. We do not know the 15 points "true" difference. And we cannot sample 10,000 times from the population.
- A common approach is to estimate statistical power for different "true" differences or "effect size".
- Usually in standard deviation terms: $\frac{\mu_{H1} - \mu_{H0}}{\sigma}$ (e.g. 0.5 means "half a standard deviation")
- If we assume that H_0 should be rejected and the true difference is 15 points:

$$\pi = P(\text{reject } H_0 | H_1 \text{ true}) = P(\text{ getting p-value } < \alpha | \text{ true mean difference is 15 po}$$

$$P\left(\frac{\overline{X}-\overline{Y}-0}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} > t_{n_x+n_y-2;0.95} \middle| \mu_x-\mu_y=15\right) =$$

$$P\left(\frac{\overline{X}-\overline{Y}-(\mu_x-\mu_y)+(\mu_x-\mu_y)}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} > t_{n_x+n_y-2;0.95} \middle| \mu_x-\mu_y=15\right) =$$

$$P\left(\frac{\overline{X}-\overline{Y}-(\mu_x-\mu_y)}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} > t_{n_x+n_y-2;0.95} - \frac{(\mu_x-\mu_y)}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} \middle| \mu_x-\mu_y=15\right) =$$

$$1-P\left(\frac{\overline{X}-\overline{Y}-(\mu_x-\mu_y)}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} < t_{n_x+n_y-2;0.95} - \frac{(\mu_x-\mu_y)}{\sqrt{\frac{S_p^2}{n_x}+\frac{S_p^2}{n_y}}} \middle| \mu_x-\mu_y=15\right) =$$

```
t_c = stats.t.ppf(0.95, n_x + n_y - 2)
power = 1 - stats.t.cdf(t_c - 15/np.sqrt(S2_p/n_x + S2_p/n_y), n_x + n_y - 2)
print(f'Power = P(reject H0 | H1) = {power: .2f}')
```

```
Power = P(reject H0 | H1) =  0.38
```

Intro to Data Science

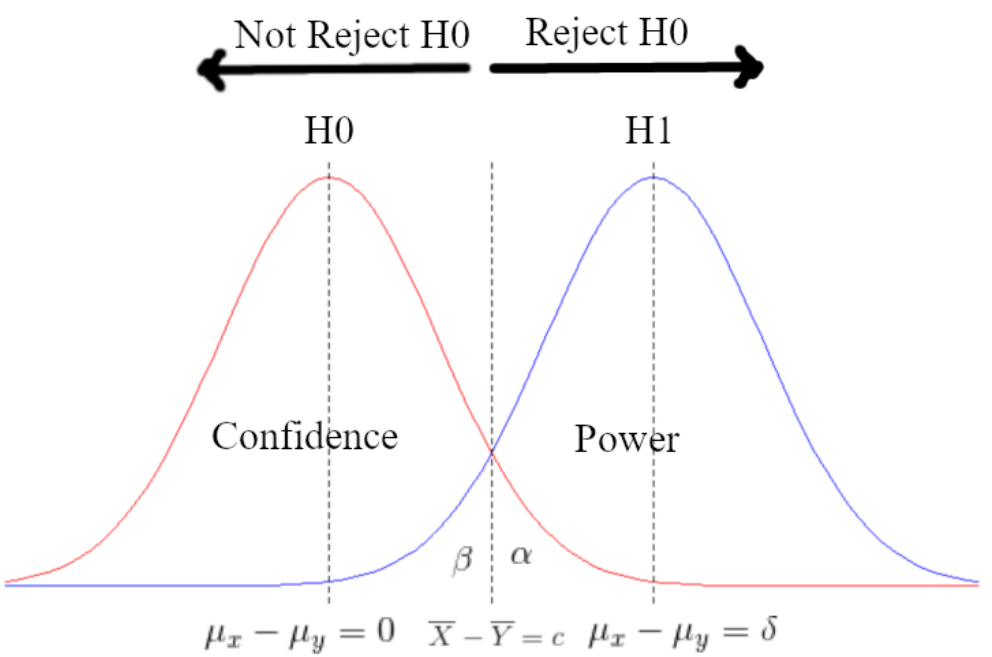
```
# with Python's built-in method:
from statsmodels.stats.power import TTestIndPower

effect = 15/np.sqrt(S2_p) # true means difference divided by S_p
power = TTestIndPower().power(effect_size = effect, nobs1 = 30, alpha = 0.05, ratio = 1

print(f'Effect size = {effect : .2f}  Power = P(reject H0 | H1) = {power: .2f}')
```

```
Effect size =  0.35  Power = P(reject H0 | H1) =  0.38
```

Reminder: Type I and Type II Errors

Reality\Decision	Not Reject H_0	Reject H_0
H_0	Confidence: $1-\alpha$	Type I Error: α
H_1	Type I Error: β	Power: $1-\beta$

Or in a graph:



Intro to Data Science

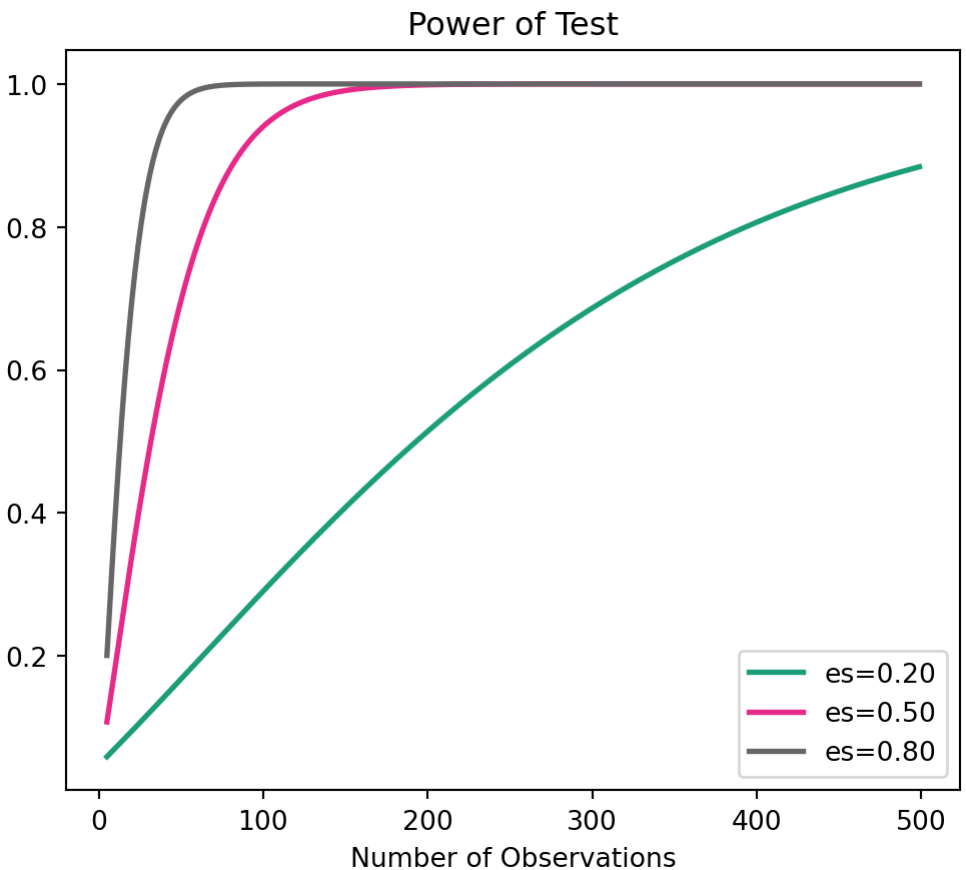
What affects Power?

From our final calculation:

$$1 - P\left(\frac{\overline{X} - \overline{Y} - (\mu_x - \mu_y)}{\sqrt{\frac{S_p^2}{n_x} + \frac{S_p^2}{n_y}}} < t_{n_x + n_y - 2; 1-\alpha} - \frac{(\mu_x - \mu_y)}{\sqrt{\frac{S_p^2}{n_x} + \frac{S_p^2}{n_y}}} | \mu_x - \mu_y = 15\right)$$

- True Difference $\mu_x - \mu_y = \delta$
- Standard deviation σ, as estimated by S_p (usually combined with true difference to create a standardized effect size: $\frac{\mu_x - \mu_y}{\sigma}$). In our case: $15/S_p = 0.35$.
- Sample size n_x, n_y (many times the go-to parameter for researchers to increase power)
- Type I Error α (usually untouched!)

```
effect_sizes = np.array([0.2, 0.5, 0.8])
sample_sizes = np.array(range(5, 500))
TTestIndPower().plot_power(dep_var = 'nobs', nobs = sample_sizes, effect_size = effect_
plt.show()
```



Caution!

- Statistical significance is not scientific significance. The tiniest, uninteresting, effect size, can be “discovered” with a large enough sample size.
- Example: Assume in our case study we have an overall difference of 0.5 in red value, instead of 15, so effect size is $0.5/S_p \approx 0.01$.

▶ Code

Intro to Data Science