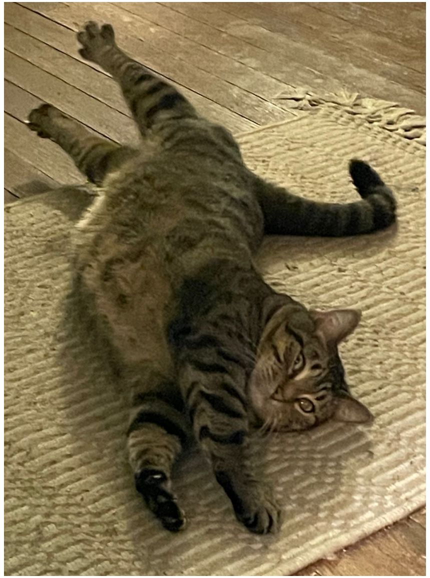
INFO 2950: Intro to Data Science

Lecture 26
2023-11-29

Agenda

1. Python Review

2. Midterm Review
   (Part 1)



When to use what languages?

- **Python**
  - Data manipulation
  - ML models

When to use what languages?

- **Python**
  - Data manipulation
  - ML models
- **SQL**
  - Merging data, group by's

In Python, (with documentation/Googling) you should now be able to do the following with dataframes:

In Python, (with documentation/Googling) you should now be able to do the following with dataframes:

- select rows, columns
- do arithmetic transforms of a column
- generate new rows / columns
- merges, sorting, data transformations
- aggregate statistics / group by
- run linear and logistic regressions
- run hypothesis tests / generate distributions
- run advanced models: NB, SVD, neural nets…

Python refresher / R intro

- We'll review basic syntax from the first few lectures
- ...and introduce how to do the same things in R!

https://www.business-science.io/business/2021/02/18/R-is-for-research.html

Python refresher / R intro

You won't need to know R for the final exam, but we want to show you that much of what you learned is *transferable knowledge!*

https://www.business-science.io/business/2021/02/18/R-is-for-research.html

R

- Designed by statisticians, not CS
- Harder to use as general-purpose programming language, but easier for stats-oriented work
- Lagging Python for recent NNs
- Base R is ok, "tidyverse" packages are 😻

https://en.wikipedia.org/wiki/R_%28programming_language%29

https://www.jetbrains.com/help/pycharm/r-plugin-support.html

Python : Loading libraries

```
import numpy as __
import pandas as __
import seaborn
```

Python vs R: Loading libraries

```python
import numpy as np
import pandas as pd
import seaborn
```

```r
library(tidyverse)
```

```
── Attaching packages ─ tidyverse 1.3.1 ──
✔ ggplot2 3.3.6     ✔ purrr   0.3.4
✔ tibble  3.1.7     ✔ dplyr   1.0.9
✔ tidyr   1.2.0     ✔ stringr 1.4.0
✔ readr   2.1.2     ✔ forcats 0.5.1
── Conflicts ──── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
```

Python : Reading a file

```
countries =
      pd.______("countries.tsv",
      delimiter="__")
```

Python vs R: Reading a file

```python
countries =
      pd.read_csv("countries.tsv",
      delimiter="\t")
```

```r
countries <- read_tsv("countries.tsv")
```

Rows: 230 Columns: 63
── Column specification──
Delimiter: "\t"
chr (34): Country, Continent, Region, Location, Highest Point, Langua...
dbl (28): Area, Borders, Length, Coastline, Height, Temperature, Popu...

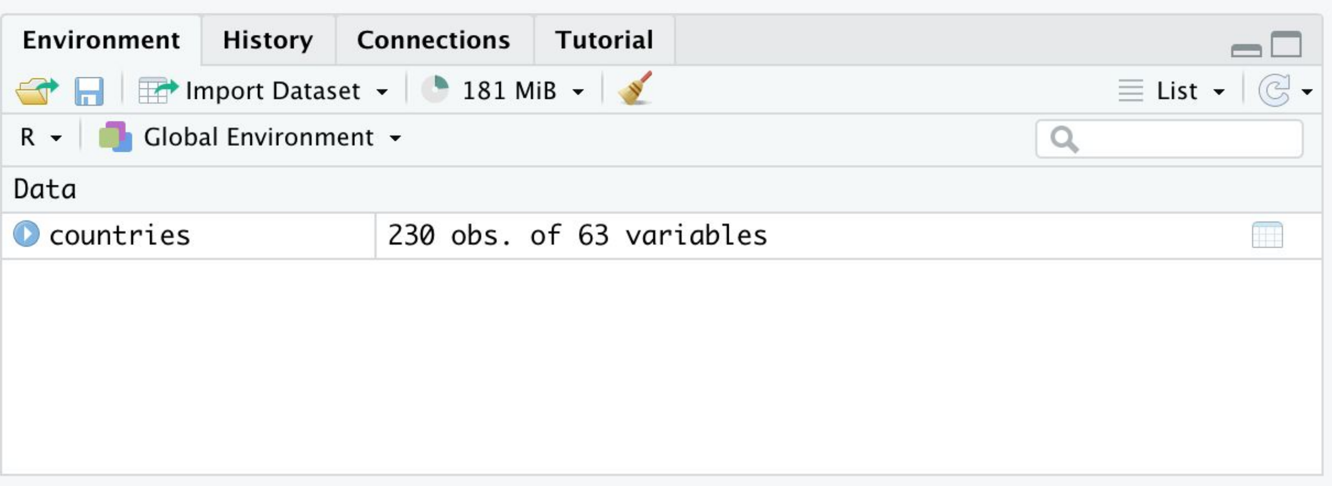
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.

Python vs R: Reading a file

```
countries =
      pd.read_csv("countries.tsv",
      delimiter="\t")
```

```
countries <- read_tsv("countries.tsv")
```

RStudio keeps track of defined variables and their sizes



Python : Show the first 3 rows

```
countries.____(3)
```

Python vs R: Show the first rows

```python
countries.head(3)
```

```r
head(countries, 3)
```

or

```r
countries %>% head(3)
```

Python vs R: Show the first rows

```
countries.head(3)
```

```
head(countries, 3)
```

or

```
countries %>% head(3)
```

"pipe" operator

value on the left becomes input to the right

Python : Show two columns

```
countries__"Country",
      "Population"__
```

Python vs R: Show two columns

```python
countries[["Country",
     "Population"]]
```

```r
countries %>% select(Country,
     Population)
```

Python vs R: Show two columns

```
countries[["Country",
         "Population"]]
```

```
countries %>% select(Country,
      Population)
```

R can figure out that "Population" is a column name, so it doesn't need quotes

Python : Select rows by condition

```
countries.____
      countries["Population"] >
      100000000 _
```

Python vs R: Select rows by condition

```python
countries.loc[
    countries["Population"] >
    100000000 ]
```

```r
countries %>%
  filter(Population >
  100000000)
```

Research Question:
do more urbanized countries have larger populations?

Python : Sort rows by a column

```
countries.__________
      (by="Urban")[["Country",
      "Urban"]]
```

Python vs R: Sort rows by a column

```
countries.sort_values
      (by="Urban")[["Country",
      "Urban"]]
```

```
countries %>% arrange(Urban)
      %>% select(Country, Urban)
```

Pipes allow us to combine multiple operations

Python vs R: Sort rows by a column

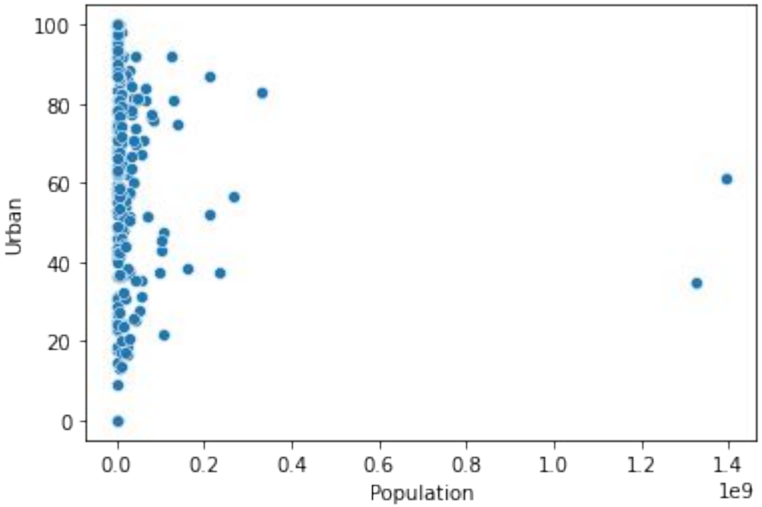
```
countries.sort_values
      (by="Urban")[["Country",
      "Urban"]]
```

	Country	Urban
227	Wallis and Futuna (France)	0.0
226	Tokelau (New Zealand)	0.0
224	Montserrat (United Kingdom)	9.1
68	Papua New Guinea	13.3
202	Burundi	13.7
...	...	...
175	Macau (China)	100.0
182	Sint Maarten (Netherlands)	100.0
7	Singapore	100.0
229	Holy See	100.0
194	Kosovo	NaN

230 rows × 2 columns

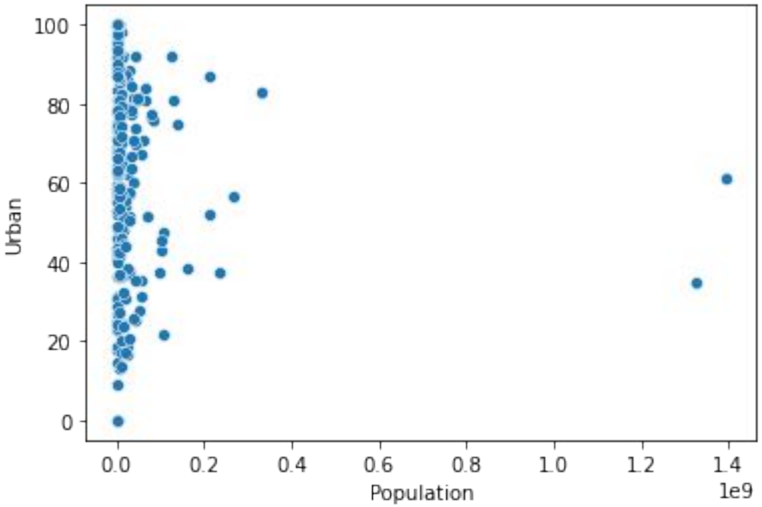
Python : plot data as points

```
seaborn.________(data=countries,
      x="________", y="________")
```

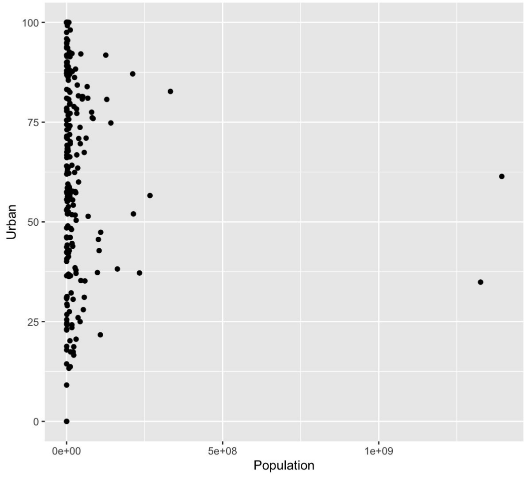


Python vs R: plot data as points

```
seaborn.scatterplot(data=countries,
    x="Population", y="Urban")
```




```
ggplot(data=countries,
    aes(Population, Urban)) +
    geom_point()
```

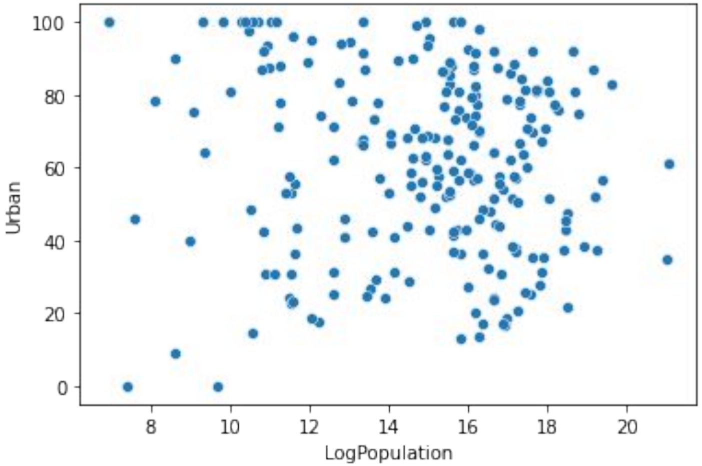


Python : Add a "LogPopulation" column

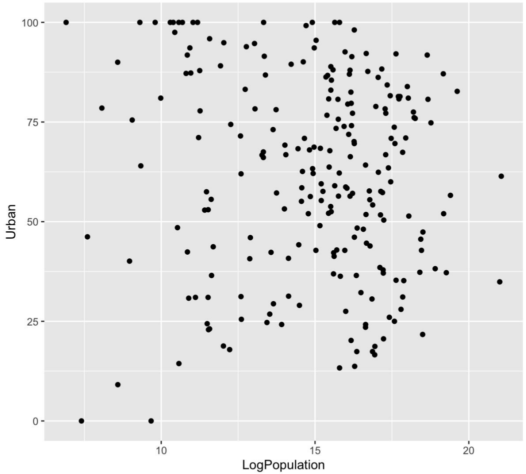
```
countries[__________] =
      np.___(_______[_______])
```

Python vs R: Add a "LogPopulation" column

```python
countries["LogPopulation"] =
      np.log(countries["Population"])
```



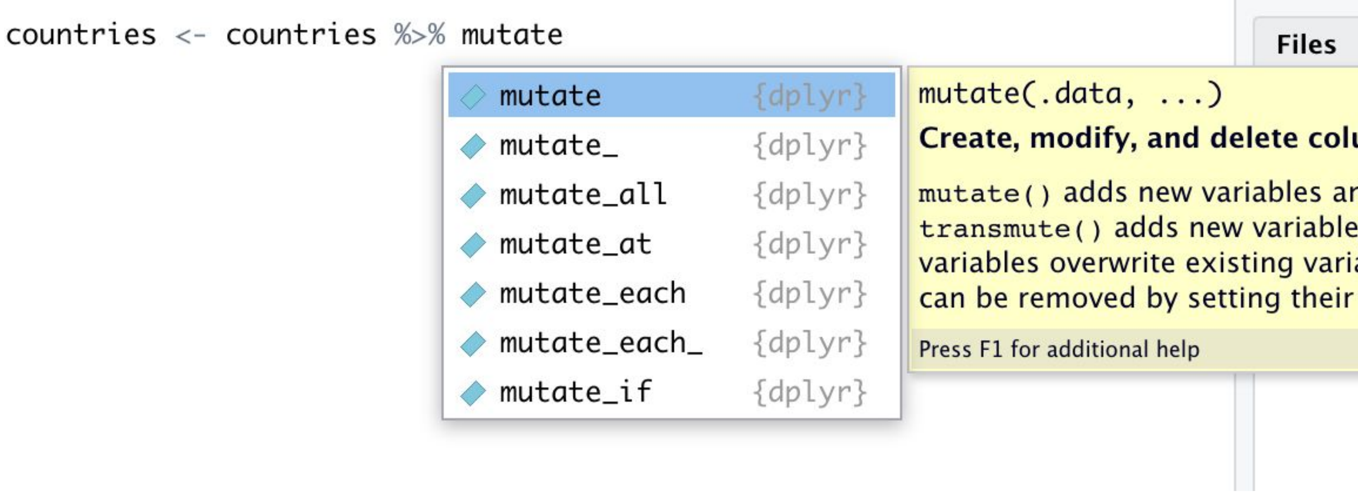

```r
countries <- countries %>%
      mutate(LogPopulation =
      log(Population))
```



Python vs R: Add a "LogPopulation" column

```python
countries["LogPopulation"] =
     np.log(countries["Populat
     ion"])
```

```r
countries <- countries %>%
     mutate(LogPopulation =
     log(Population))
```



RStudio helps with API hints (also Google Colab and VSCode for Python)

Python: Predict population from Urban%

```
urban_pop_model = LinearRegression()
    .___(countries[["Urban"]], countries["LogPopulation"])

urban_pop_model._______, urban_pop_model.________

    (array([-0.0051409]), 16.475302332927527)
```

Python: Predict population from Urban%

```
urban_pop_model = LinearRegression()
    .fit(countries[["Urban"]], countries["LogPopulation"])

urban_pop_model.coef_, urban_pop_model.intercept_

    (array([-0.0051409]), 16.47530233292752 7)
```

R: Predict population from Urban%

```
urban_pop_model <- lm(LogPopulation ~ Urban, data=countries)
summary(urban_pop_model)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-6.8100 -0.9732  0.1155  1.2519  4.8958

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
      (Intercept) 16.475302   0.479951  34.327   <2e-16 ***
      Urban       -0.005141   0.007324  -0.702    0.484
      ---
      Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

      Residual standard error: 1.97 on 142 degrees of freedom
      Multiple R-squared:  0.003458,   Adjusted R-squared:  -0.00356
      F-statistic: 0.4927 on 1 and 142 DF,  p-value: 0.4839
```

R: Predict population from Urban%

```
urban_pop_model <- lm(LogPopulation ~ Urban, data=countries)
summary(urban_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max
-6.8100 -0.9732  0.1155  1.2519  4.8958

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
     (Intercept) 16.475302   0.479951  34.327   <2e-16 ***
     Urban       -0.005141   0.007324  -0.702    0.484
     ---
     Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Residual standard error: 1.97 on 142 degrees of freedom
     Multiple R-squared:  0.003458,   Adjusted R-squared:  -0.00356
     F-statistic: 0.4927 on 1 and 142 DF,  p-value: 0.4839
```

R uses "~" to build a formula for linear models

R: Predict population from Urban%

```
urban_pop_model <- lm(LogPopulation ~ Urban, data=countries)
summary(urban_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max
-6.8100 -0.9732  0.1155  1.2519  4.8958

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
     (Intercept) 16.475302   0.479951  34.327   <2e-16 ***
     Urban       -0.005141   0.007324  -0.702    0.484
     ---
     Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Residual standard error: 1.97 on 142 degrees of freedom
     Multiple R-squared:  0.003458,   Adjusted R-squared:  -0.00356
     F-statistic: 0.4927 on 1 and 142 DF,  p-value: 0.4839
```

Is our coefficient > or < 0?

R: Predict population from Urban%

```
urban_pop_model <- lm(LogPopulation ~ Urban, data=countries)
summary(urban_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.8100 -0.9732  0.1155  1.2519  4.8958 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 16.475302   0.479951  34.327   <2e-16 ***
Urban       -0.005141   0.007324  -0.702    0.484    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.97 on 142 degrees of freedom
Multiple R-squared:  0.003458,	Adjusted R-squared:  -0.00356 
F-statistic: 0.4927 on 1 and 142 DF,  p-value: 0.4839
```

Urban coefficient slightly < 0

R: Predict population from Urban%

```
urban_pop_model <- lm(LogPopulation ~ Urban, data=countries)
summary(urban_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max
-6.8100 -0.9732  0.1155  1.2519  4.8958

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
     (Intercept) 16.475302   0.479951  34.327   <2e-16 ***
     Urban       -0.005141   0.007324  -0.702    0.484
     ---
     Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Residual standard error: 1.97 on 142 degrees of freedom
     Multiple R-squared:  0.003458,   Adjusted R-squared:  -0.00356
     F-statistic: 0.4927 on 1 and 142 DF,  p-value: 0.4839
```

Is our coefficient "statistically significant"?

R: Predict population from Urban%

```
urban_pop_model <- lm(LogPopulation ~ Urban, data=countries)
summary(urban_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max
-6.8100 -0.9732  0.1155  1.2519  4.8958

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 16.475302   0.479951  34.327   <2e-16 ***
Urban       -0.005141   0.007324  -0.702    0.484
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.97 on 142 degrees of freedom
Multiple R-squared:  0.003458,   Adjusted R-squared:  -0.00356
F-statistic: 0.4927 on 1 and 142 DF,  p-value: 0.4839
```

Nope!
The intercept is *definitely* not zero, but Urban% looks random

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.4715 -0.9001  0.2607  1.1208  3.6736 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146    
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651 
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

Include multiple predictors with +

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max
-6.4715 -0.9001  0.2607  1.1208  3.6736

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

1. Is Urban coefficient statistically significant at the 0.001 level?
2. Is Area coefficient statistically significant at the 0.001 level?

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.4715 -0.9001  0.2607  1.1208  3.6736 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146    
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651 
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

1. No 2. Yes

Urban% is still not significant, but lower p value.
Land area is highly significant!

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.4715 -0.9001  0.2607  1.1208  3.6736 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146    
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651 
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

Which coefficient has larger magnitude, Urban or Area?

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.4715 -0.9001  0.2607  1.1208  3.6736 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146    
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651 
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

Magnitude means absolute value; Urban's is bigger: 0.009844 > 0.0000003765

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.4715 -0.9001  0.2607  1.1208  3.6736 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146    
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651 
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

1 unit (percentage point) increase in Urban% corresponds to a _____ [increase/decrease] in log population.

1 unit (square miles) increase in Area corresponds to a _____ [increase/decrease] in log population.

R: Predict population from Urban% + Area

```
ur_ar_pop_model <- lm(LogPopulation ~ Urban + Area, data=countries)
summary(ur_ar_pop_model)
Residuals:
    Min      1Q  Median      3Q     Max 
-6.4715 -0.9001  0.2607  1.1208  3.6736 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.644e+01  4.378e-01  37.544  < 2e-16 ***
Urban       -9.844e-03  6.736e-03  -1.462    0.146    
Area         3.765e-07  6.911e-08   5.448 2.21e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.797 on 141 degrees of freedom
Multiple R-squared:  0.1768,	Adjusted R-squared:  0.1651 
F-statistic: 15.14 on 2 and 141 DF,  p-value: 1.108e-06
```

1 unit (percentage point) increase in Urban% corresponds to a .0098 decrease in log population.

1 unit (square miles) increase in Area corresponds to a 0.0000003765 increase in log population. Small but significant!

Python: Predict population from Urban%

```
outputs,  predictors = patsy.dmatrices("LogPopulation ~ Urban",
    data=countries, return_type="dataframe")
sm_urban_pop_model = sm.OLS(outputs, predictors).fit()
sm_urban_pop_model.summary()
```

Using `statsmodels` package

OLS Regression Results

Dep. Variable:	LogPopulation	**R-squared:**	0.003
Model:	OLS	**Adj. R-squared:**	-0.004
Method:	Least Squares	**F-statistic:**	0.4927
Date:	Wed, 30 Nov 2022	**Prob (F-statistic):**	0.484
Time:	10:56:43	**Log-Likelihood:**	-300.98
No. Observations:	144	**AIC:**	606.0
Df Residuals:	142	**BIC:**	611.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	16.4753	0.480	34.327	0.000	15.527	17.424
Urban	-0.0051	0.007	-0.702	0.484	-0.020	0.009

Omnibus:	16.672	**Durbin-Watson:**	1.709
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	21.589
Skew:	-0.684	**Prob(JB):**	2.05e-05
Kurtosis:	4.315	**Cond. No.**	192.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Python: Predict population from Urban%

```
outputs,  predictors = patsy.dmatrices("LogPopulation ~ Urban",
    data=countries, return_type="dataframe")
sm_urban_pop_model = sm.OLS(outputs, predictors).fit()
sm_urban_pop_model.summary()
```

OLS Regression Results

Dep. Variable:	LogPopulation	**R-squared:**	0.003
Model:	OLS	**Adj. R-squared:**	-0.004
Method:	Least Squares	**F-statistic:**	0.4927
Date:	Wed, 30 Nov 2022	**Prob (F-statistic):**	0.484
Time:	10:56:43	**Log-Likelihood:**	-300.98
No. Observations:	144	**AIC:**	606.0
Df Residuals:	142	**BIC:**	611.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	16.4753	0.480	34.327	0.000	15.527	17.424
Urban	-0.0051	0.007	-0.702	0.484	-0.020	0.009

dmatrices uses the same formula syntax as R

Python: Predict population from Urban%

```
outputs,  predictors = patsy.dmatrices("LogPopulation ~ Urban",
    data=countries, return_type="dataframe")
sm_urban_pop_model = sm.OLS(outputs, predictors).fit()
sm_urban_pop_model.summary()
```

OLS Regression Results

Dep. Variable:	LogPopulation	**R-squared:**	0.003
Model:	OLS	**Adj. R-squared:**	-0.004
Method:	Least Squares	**F-statistic:**	0.4927
Date:	Wed, 30 Nov 2022	**Prob (F-statistic):**	0.484
Time:	10:56:43	**Log-Likelihood:**	-300.98
No. Observations:	144	**AIC:**	606.0
Df Residuals:	142	**BIC:**	611.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	16.4753	0.480	34.327	0.000	15.527	17.424
Urban	-0.0051	0.007	-0.702	0.484	-0.020	0.009

statsmodels does not include intercept by default, dmatrices function adds an intercept

Python: Predict population from Urban%

```
outputs,  predictors = patsy.dmatrices("LogPopulation ~ Urban",
    data=countries, return_type="dataframe")
sm_urban_pop_model = sm.OLS(outputs, predictors).fit()
sm_urban_pop_model.summary()
```

Same output!

OLS Regression Results

Dep. Variable:	LogPopulation	**R-squared:**	0.003
Model:	OLS	**Adj. R-squared:**	-0.004
Method:	Least Squares	**F-statistic:**	0.4927
Date:	Wed, 30 Nov 2022	**Prob (F-statistic):**	0.484
Time:	10:56:43	**Log-Likelihood:**	-300.98
No. Observations:	144	**AIC:**	606.0
Df Residuals:	142	**BIC:**	611.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	16.4753	0.480	34.327	0.000	15.527	17.424
Urban	-0.0051	0.007	-0.702	0.484	-0.020	0.009

Omnibus:	16.672	**Durbin-Watson:**	1.709
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	21.589
Skew:	-0.684	**Prob(JB):**	2.05e-05
Kurtosis:	4.315	**Cond. No.**	192.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Python & R: data reshaping

	Python	SQL	R
Wide to long	melt(), stack(), …	CROSS APPLY(), UNPIVOT()	melt(), pivot_longer(), gather(), …
Long to wide	pivot(), pivot_table(), unstack(), ...	PIVOT()	pivot_wider(), dcast(), spread(), …

1 minute break & attendance!

tinyurl.com/ey7w79k5

Python : inner join of df1 and df2 (on column ‘id’)

(Using Pandas merge!)

```
___.merge(___, how=___,
      on=___)
```

Python : inner join of df1 and df2 (on column ‘id’)

(Using Pandas merge!)

```
df1.merge(df2, how=’inner’,
      on=‘id’)
```

SQL : inner join of df1 and df2 (on column 'id')

(Using Pandas merge!)

```
df1.merge(df2, how='inner',
      on='id')
```

(Using SQL merge!)

```
SELECT *
      FROM df1
      _____ _____ ___
      ___ df1.id ___ df2.id
```

SQL : inner join of df1 and df2 (on column ‘id’)

```
df1.merge(df2, how='inner',
      on='id')

SELECT *
      FROM df1
      INNER JOIN df2
      ON df1.id = df2.id
```

Python & SQL & R: inner join of df1 and df2 (on column ‘id’)

```
df1.merge(df2, how=‘inner’,
      on=‘id’)
-----------------------------------
SELECT *
      FROM df1
      INNER JOIN df2
      ON df1.id = df2.id
```

Using Base R:

```
merge(df1, df2, by=‘id’)
```

Using dplyr package:

```
df1 %>% inner_join(df2,by=‘id’)
```

Python : get average ‘price’ grouped by ‘product’ from dataframe ‘df’

(Using Pandas!)

```
__.groupby(______)[______].____()
```

Python : get average ‘price’ grouped by ‘product’ from dataframe ‘df’

(Using Pandas!)

```
df.groupby(‘product’)[‘price’].mean()
```

SQL : get average ‘price’ grouped by ‘product’ from dataframe ‘df’

(Using Pandas!)

```
df.groupby(‘product’)[‘price’].mean()
```

(Using SQL!)

```
SELECT ________
___ df
______ ___ ______
```

SQL : get average ‘price’ grouped by ‘product’ from dataframe ‘df’

(Using Pandas!)

```
df.groupby(‘product’)[‘price’].mean()
```

(Using SQL!)

```
SELECT AVG(price)
FROM df
GROUP BY product
```

Python & SQL & R: get average ‘price’ grouped by ‘product’ from dataframe ‘df’

(Using Pandas!)

```
df.groupby(‘product’)[‘price’].
      mean()
```

(Using SQL!)

```
SELECT AVG(price)
FROM df
GROUP BY product
```

(Using R tidyverse)

```
df %>% group_by(product)
   %>% summarise(mean(price))
```

Numpy stats in 1-D

```
>>> a = np.array([[1, 2], [3, 4]])
>>> __.______                       (2,2)
>>> np.mean(a, axis=0)              ?
>>> np.median(a, axis=1)            ?
```

https://numpy.org/doc/stable/reference/generated/numpy.mean.html

Numpy stats in 1-D

```
>>> a = np.array([[1, 2], [3, 4]])
>>> np.shape                                (2,2)
>>> np.mean(a, axis=0)                      array([2., 3.])
>>> np.median(a, axis=1)                    array([1.5, 3.5])
```

https://numpy.org/doc/stable/reference/generated/numpy.mean.html

Numpy stats in 1-D

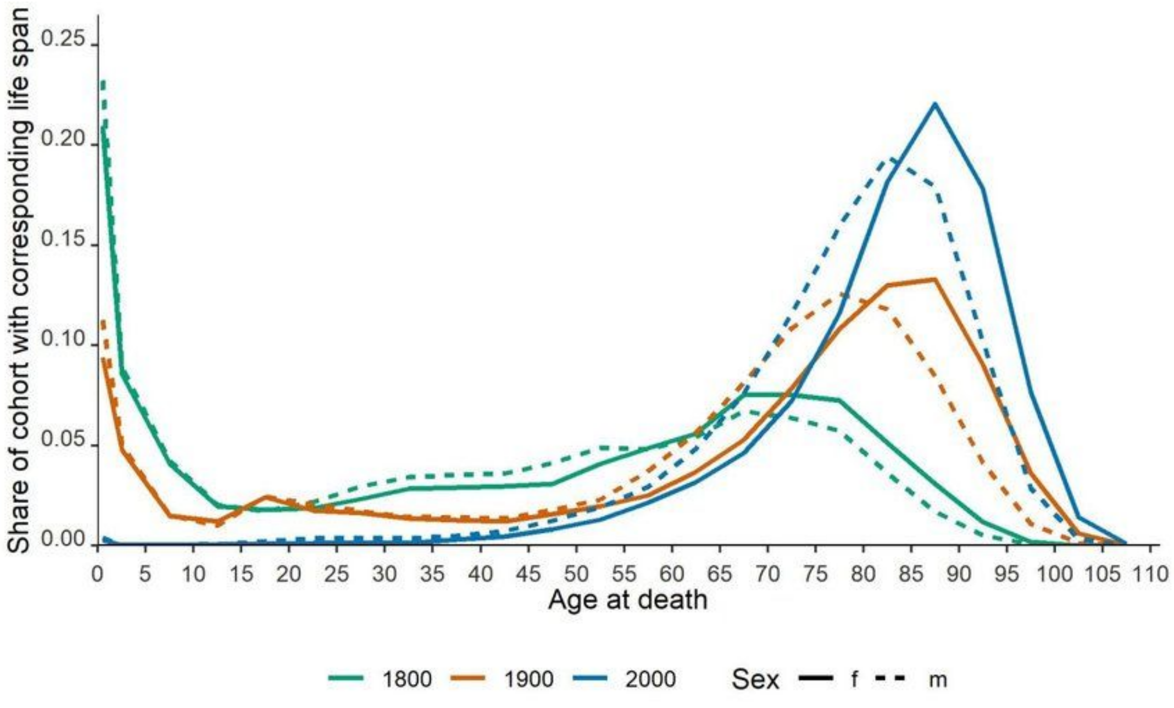
```
>>> a = np.array([[1, 2], [3, 4]])
>>> __.______                        (2,2)
>>> np.mean(a, axis=0)               array([2., 3.])
>>> np.median(a, axis=1)             array([1.5, 3.5])
```

Remember: median minimizes sum of “absolute distances” but the Python default is the midpoint!

https://numpy.org/doc/stable/reference/generated/numpy.mean.html

Swedish ages at death

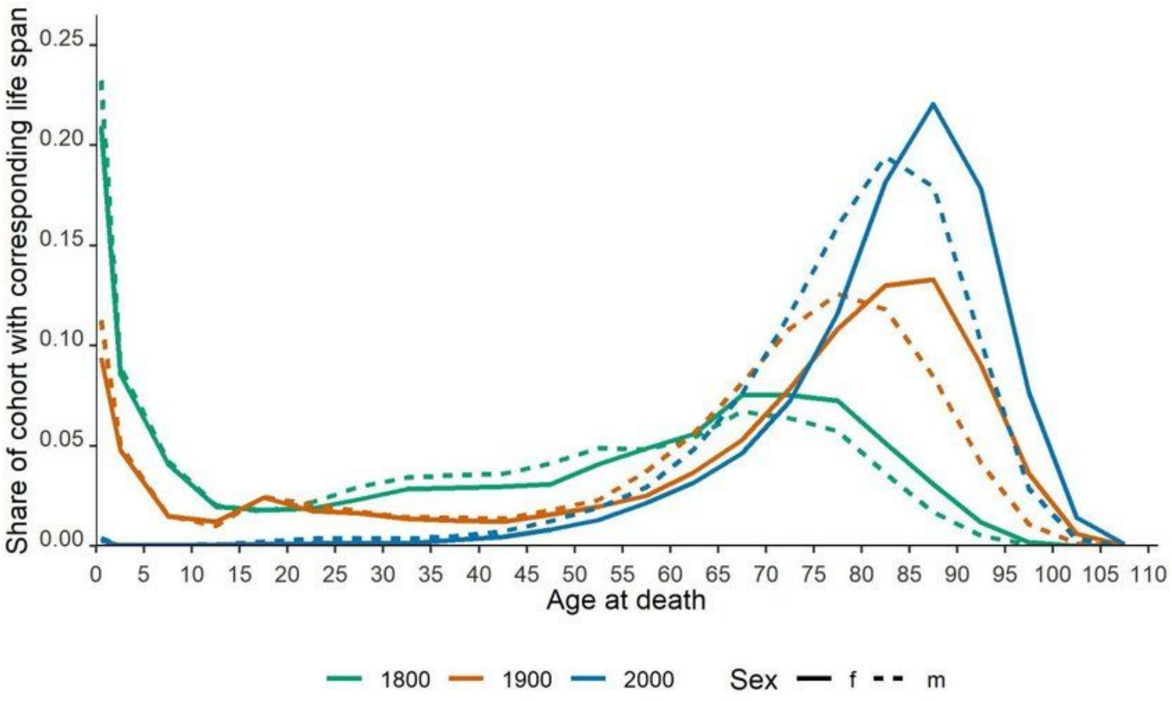
1. Is mean meaningful?
2. Is median meaningful?



https://www.researchgate.net/publication/339721309_Exceptional_lives_extraordinary_families_Familial_clustering_of_longevity_in_the_19th_and_early_20th_centuries

Swedish ages at death

Nope & nope, consider using a metric like average adult age of death for bimodal data →



https://www.researchgate.net/publication/339721309_Exceptional_lives_extraordinary_families_Familial_clustering_of_longevity_in_the_19th_and_early_20th_centuries

What are these called?

$$\frac{\Sigma_i (X_i - \bar{X})^2}{N}$$

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

1.

2.

What are these called?

$$\frac{\Sigma_i (X_i - \bar{X})^2}{N}$$

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

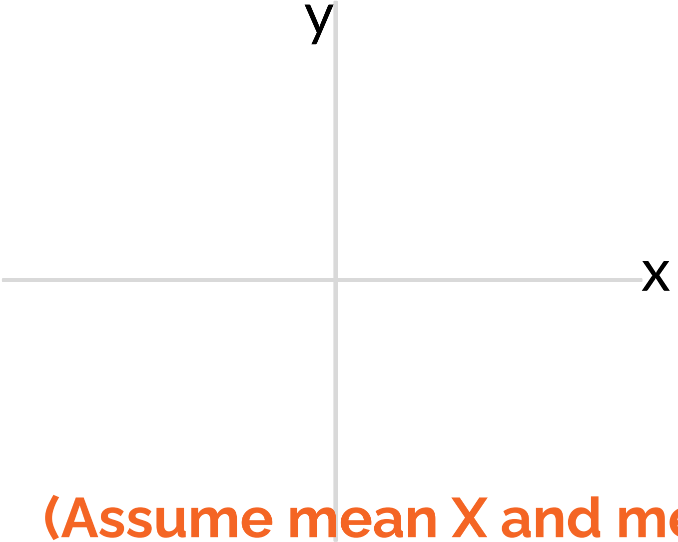
1. Variance

2. Covariance

Covariance

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

What signs will covariance take in each quadrant?



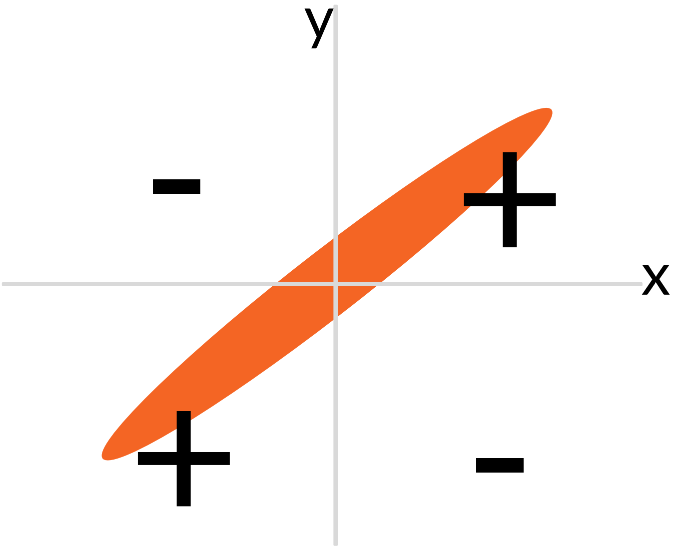
(Assume mean X and mean Y = 0)

Covariance

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

What signs will covariance take in each quadrant?

most positive covariance: close to the diagonal, in the positive quadrants



Covariance

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

What is the covariance of x with itself?

Covariance

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

What is the covariance of x with itself? Variance!

$$\frac{\Sigma_i (X_i - \bar{X})^2}{N}$$

Normalization

What is this called?

$$(X_i - \overline{X})/\sigma_x$$

Normalization

What is this called?

$$(X_i - \bar{X})/\sigma_x$$

"z-score"

Covariance calculated with z-scores?

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

$$(X_i - \bar{X})/\sigma_x$$

What is this called?

$$\frac{\Sigma_i (X_i - X)(Y_i - \bar{Y})/(\sigma_x \sigma_y)}{N}$$

Covariance calculated with z-scores?

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})}{N}$$

$$(X_i - \bar{X})/\sigma_x$$

(Pearson) Correlation:

$$\frac{\mathrm{cov}(X, Y)}{\sigma_x \sigma_y}$$

What is correlation of X with itself?

$$\frac{\Sigma_i (X_i - \bar{X})(Y_i - \bar{Y})/(\sigma_x \sigma_y)}{N}$$

What is correlation of X with itself?

$$\frac{\Sigma_i(X_i - \bar{X})(X_i - \bar{X})/(\sigma_x\sigma_x)}{N}$$

$$= \mathrm{Var}(X) / (\sigma_x\sigma_x)$$

$$= 1$$

Covariance, Correlation

- Correlation is normalized and measures both strength and direction of linear relationship between X and Y

- Covariance just measures direction of linear relationship between X and Y

When is it useful to look at correlation matrices?

When is it useful to look at correlation matrices?

- Make a visual summary of lots of data to understand patterns

- Check for multicollinearity when deciding on regression inputs

Why is multicollinearity bad?

- Do you get similar coefficients on **ad** if you run...
  - sales ~ price + ad + loc + volume
  - sales ~ price + ad + loc

	sales	price	ad	loc	volume
sales	1.00	-0.70	0.12	0.01	0.39
price	-0.70	1.00	0.00	0.00	-0.18
ad	0.12	0.00	1.00	0.00	-0.74
loc	0.01	0.00	0.00	1.00	-0.04
volume	0.39	-0.18	-0.74	-0.04	1.00

https://www.edupristine.com/blog/detecting-multicollinearity

Why is multicollinearity bad?

- Collinear inputs can make the regression coefficients very unstable

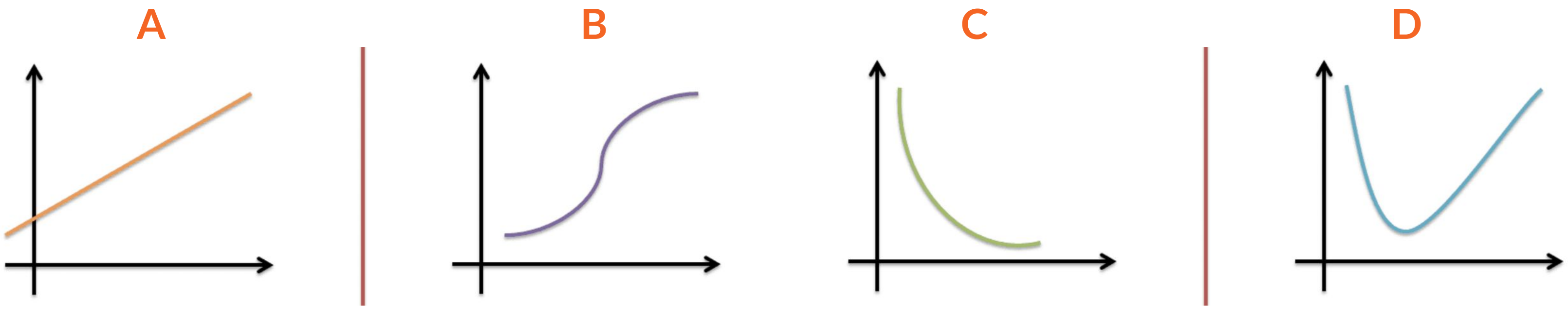
	Estimate
(Intercept)	125.931
price	-11.836
ad	131.283
loc	7.768
volume	11.870

	Estimate
(Intercept)	662.733
price	-15.100
ad	20.500
loc	1.833

https://www.edupristine.com/blog/detecting-multicollinearity

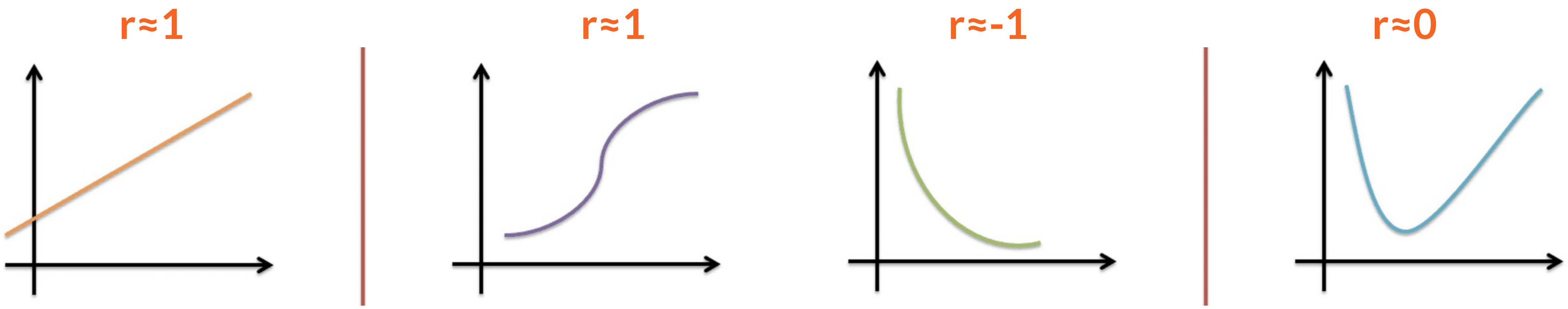
Rank (spearman) correlations

- Used to understand x and y monotonicity instead of linearity
- **Which is/are non-monotone?**



Rank (spearman) correlations

- D is non-monotone

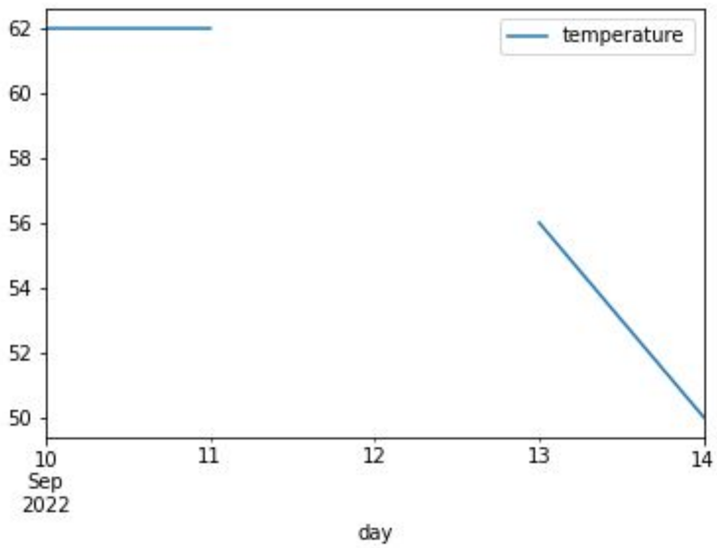


https://discover.hubpages.com/education/Differences-Between-Spearman-Rank-Correlation-and-Pearson-Correlation

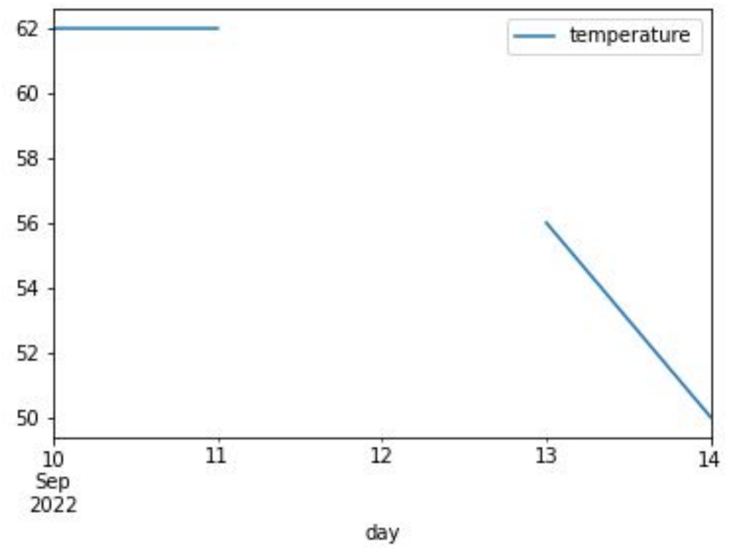
Time Series

- Data where one column denotes time (*datetime* format)
- Most meaningful when data is aggregated so that each "time step":
    - Is regularly spaced (e.g. daily, monthly, quarterly data) chronologically
    - Has corresponding data per time step
    - Is unique
    - Deals with missing values

Time Series: why is this happening?



Time Series: why is this happening?



Missing data (has NaNs)!

Regression motivations

1. Make predictions
2. Summarize relationship between variables
3. Inspect outliers and other oddities

Regression motivations

Input

Output

1. Make predictions
2. Summarize relationship between variables
3. Inspect outliers and other oddities

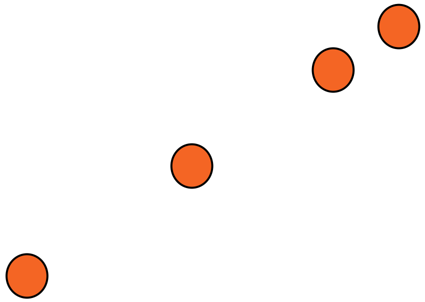
- $y = \alpha + \beta \cdot x$
- $y \sim x$

Regression motivations

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?

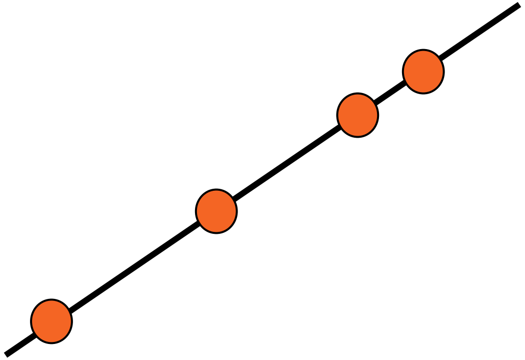
Regression motivations

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?



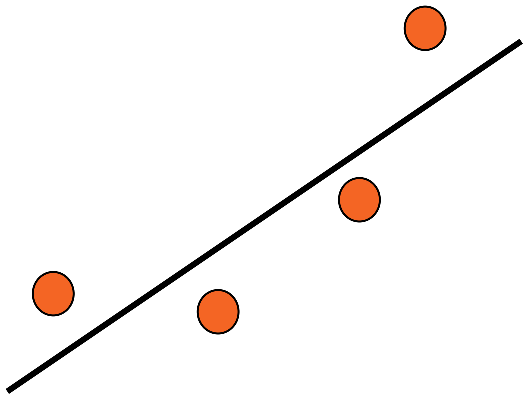
Regression motivations

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?



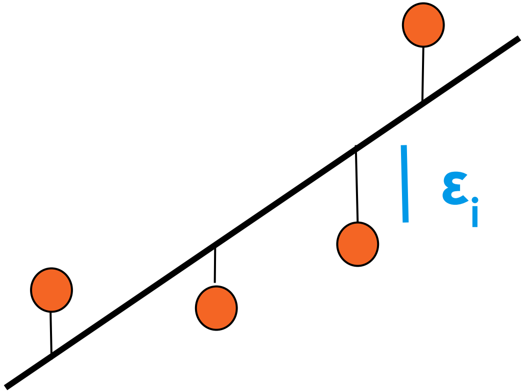
Regression motivations

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?



Regression motivations

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?



Regression motivations

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?

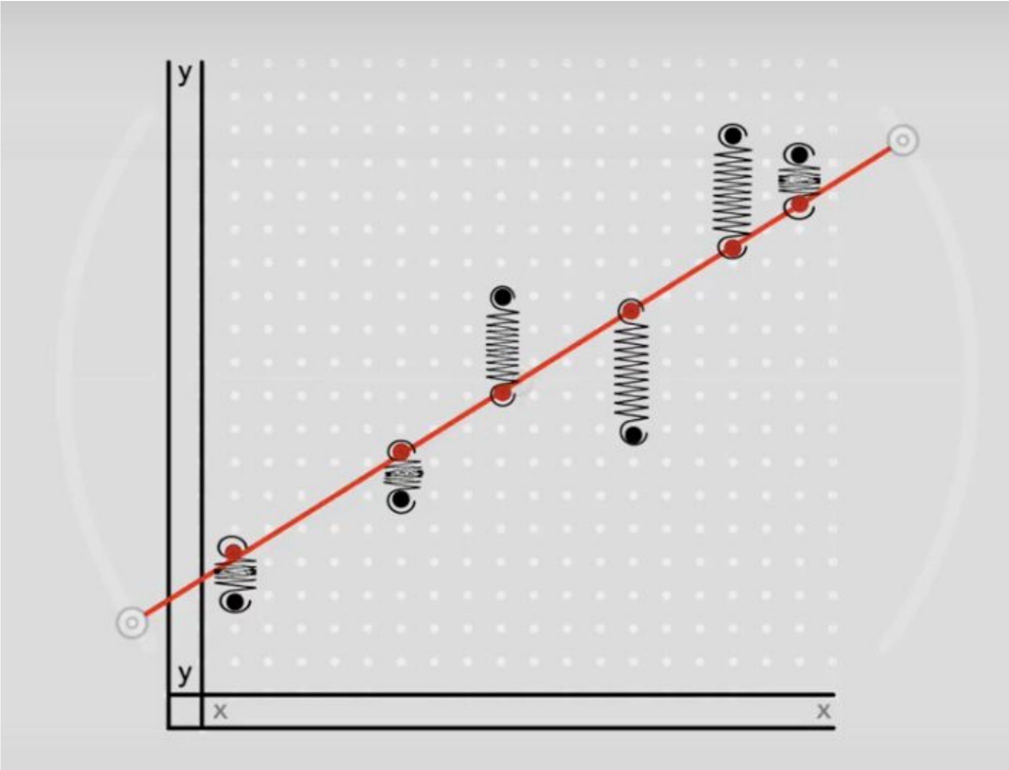
No, need error term (residual) corresponding to each input *i* to represent each individual dot: $y_i = \alpha + \beta x_i + \varepsilon_i$

Regression (OLS)

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?

No, need error term (residual) corresponding to each input *i* to represent each individual dot: $y_i = \alpha + \beta x_i + \boxed{\varepsilon_i}$

KEY INSIGHT: we want to minimize this error across all of our points *i*



https://www.jmp.com/en_us/statistics-knowledge-portal/what-is-regression/the-method-of-least-squares.html

Comparing linear models

Is $y = \alpha$ a linear regression model?

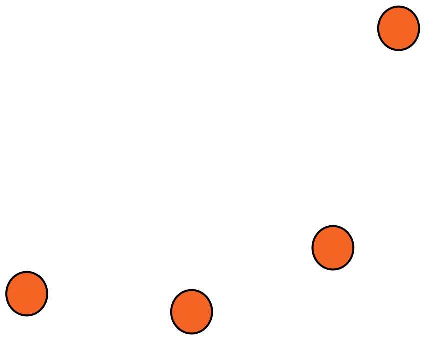
Comparing linear models

Is $y = \alpha$ a linear regression model? YES!

Comparing linear models

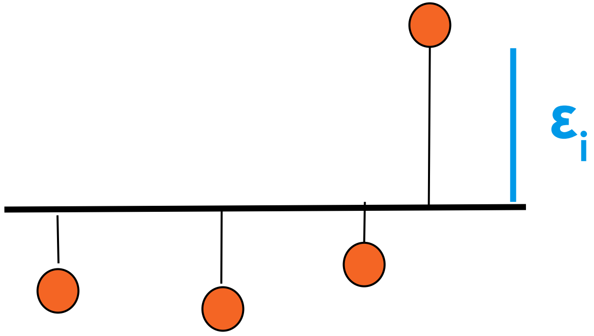
Is $y = \alpha$ a linear regression model? YES!

Draw the line for this model if $\alpha = \bar{y}$:



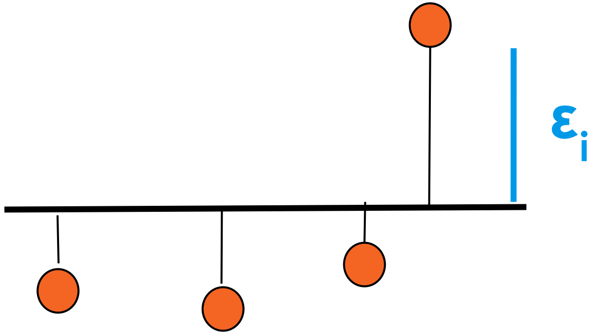
Comparing linear models

Draw the line for $y = \alpha = \bar{y}$:



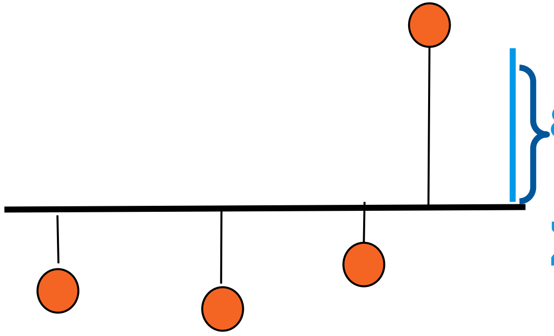
Comparing linear models

What statistic is the sum of squared residuals ε_i for $y = a = \bar{y}$ equal to?



Comparing linear models

What statistic is the sum of squared residuals ε_i for $y = \alpha$ equal to? N*Var(Y)



$\varepsilon_i = y_i - \bar{y}$

$\Sigma\varepsilon_i^2 = \Sigma(y_i - \bar{y})^2$

$= N * (\Sigma(y_i - \bar{y})^2/N)$

Regression (OLS)

Does this regression equation $y = \alpha + \beta \cdot x$ pass through all input data?

No, need error term (residual) corresponding to each input *i* to represent each individual dot: $y_i = \alpha + \beta x_i + \boxed{\varepsilon_i}$

KEY INSIGHT: we want to minimize this error across all of our points *i*

How to get α, β?

- **Calculus** (minimize sum of squared error)
- **Stochastic gradient descent** (need this method for complicated models)
- **Use Python**
  - Fit linear regression
  - Call `intercept_` and `coef_`

How to get α, β?

- **Calculus** (minimize sum of squared error)
- **Stochastic gradient descent** (need this method for complicated models)
- **Use Python**
    - Fit linear regression
    - Call intercept_ and coef_

Linear Regression β formula

$$\widehat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Linear Regression β formula

N * ?

$$\widehat{\beta} = \frac{\boxed{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}}{\boxed{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$$

N * ?

Linear Regression β formula

N * cov(x,y)

$$\widehat{\beta} = \frac{\boxed{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}}{\boxed{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$$

N * var(x)

Which is tall, which is wide?

input_x	is_high	output_y
2022-09-19	high	77
2022-09-19	low	58
2022-09-20	high	73
2022-09-20	low	55
2022-09-21	high	80
2022-09-21	low	57

input_x	high	low
2022-09-19	77	58
2022-09-20	73	55
2022-09-21	80	57

Tall

input_x	is_high	output_y
2022-09-19	high	77
2022-09-19	low	58
2022-09-20	high	73
2022-09-20	low	55
2022-09-21	high	80
2022-09-21	low	57

Wide

input_x	high	low
2022-09-19	77	58
2022-09-20	73	55
2022-09-21	80	57

- Which df would you use to run a regression?
- Write the model with ~ and variable names.

Tall

input_x	is_high	output_y
2022-09-19	high	77
2022-09-19	low	58
2022-09-20	high	73
2022-09-20	low	55
2022-09-21	high	80
2022-09-21	low	57

Wide

input_x	high	low
2022-09-19	77	58
2022-09-20	73	55
2022-09-21	80	57

- Tall
- output_y ~ input_x + is_high

Tall

input_x	is_high	output_y
2022-09-19	high	77
2022-09-19	low	58
2022-09-20	high	73
2022-09-20	low	55
2022-09-21	high	80
2022-09-21	low	57

Wide

input_x	high	low
2022-09-19	77	58
2022-09-20	73	55
2022-09-21	80	57

- What keyword do you use to go from tall to wide?

Tall

input_x	is_high	output_y
2022-09-19	high	77
2022-09-19	low	58
2022-09-20	high	73
2022-09-20	low	55
2022-09-21	high	80
2022-09-21	low	57

Wide

input_x	high	low
2022-09-19	77	58
2022-09-20	73	55
2022-09-21	80	57

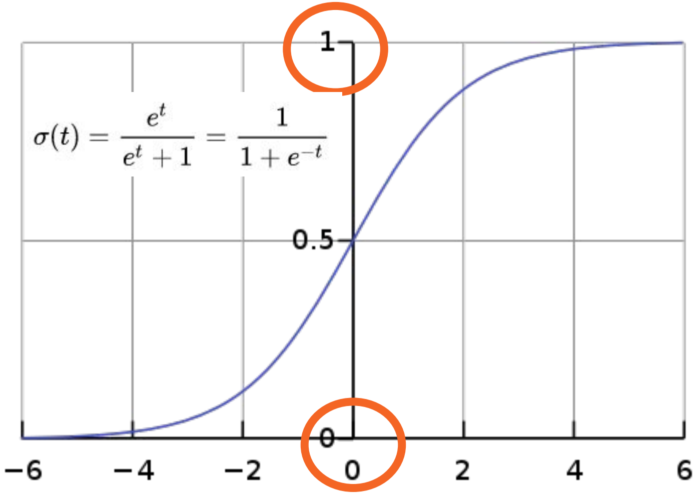
- **What keyword do you use to go from tall to wide? Pivot**

Logistic regression

- How do we “smoosh” $\alpha + \beta \cdot x$ predictions into a [0,1] probability range?

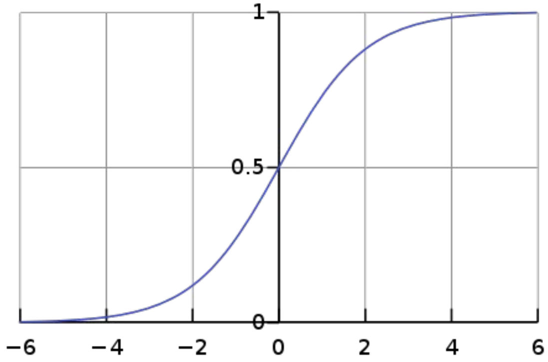
Logistic regression

- How do we "smoosh" α + β · x predictions into a [0,1] probability range?



Logistic (sigmoid) function

Logistic regression



- This allows us to predict the probability that y=TRUE

$$p(x) = \frac{1}{1+e^{-(\alpha+\beta\cdot x))}}$$

- So, interpret logistic regression differently than linear regression

What is this called?

Probability y=1: $$p(x) = \frac{1}{1+e^{-(\alpha+\beta\cdot x))}}$$

Probability y=0: $$1 - p(x)$$

Pr(Magnus win) / Pr (Magnus lose) = $\mathbf{p(x) / (1\text{-}p(x))}$

The Odds Ratio

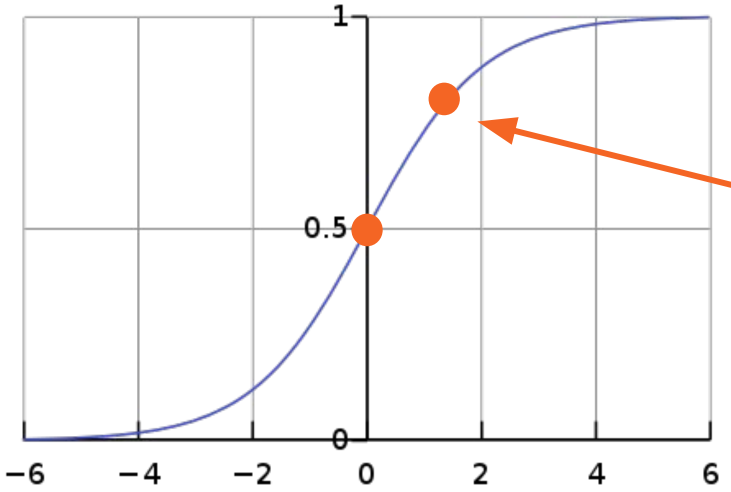
Probability y=1: $p(x) = \frac{1}{1+e^{-(\alpha+\beta \cdot x))}}$

Probability y=0: $1 - p(x)$

Pr(Magnus win) / Pr (Magnus lose) = p(x) / (1-p(x))

Intuition: log odds ratio

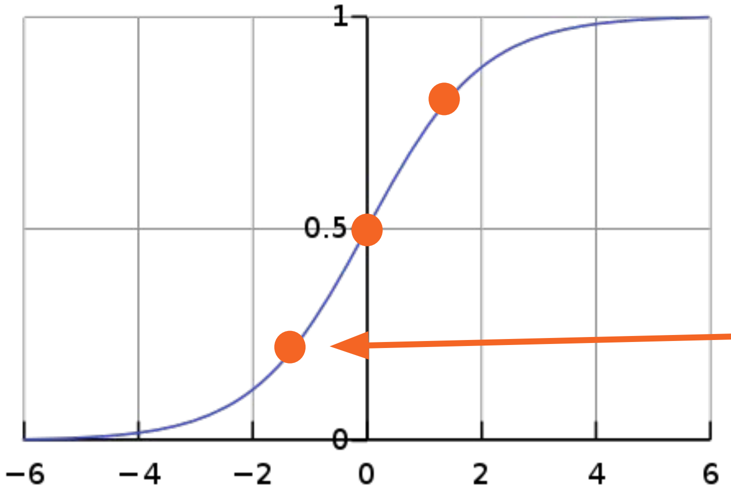
Logistic Function $\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$



Log odds	Probability	Odds $e^{1.38} = 4$
0.0	0.5	1:1
1.38	$\sigma(1.38)=0.8$ 0.8	$0.8 = 4/(4+1)$ 4:1
-1.38	?	1:4
-2.94	0.05	?
-4.59	0.01	?

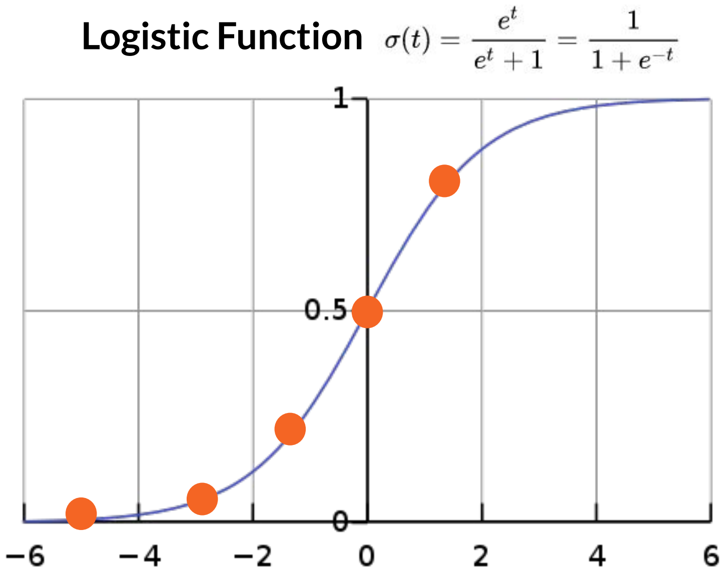
Intuition: log odds ratio

Logistic Function $\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$



Log odds	Probability	Odds $e^{-1.38} = 0.25$
0.0	0.5	1:1
1.38	0.8	4:1
-1.38	$\sigma(-1.38)=0.2$ 0.2	$0.2 = 1/(1+4)$ 1:4
-2.94	0.05	?
-4.59	0.01	?

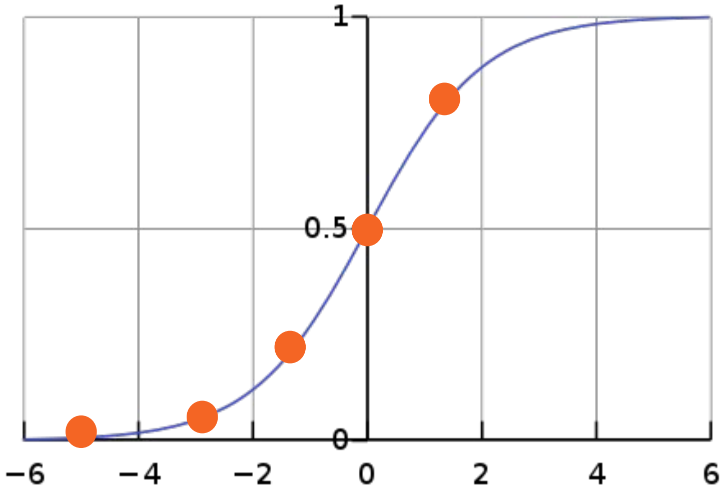
Intuition: log odds ratio



Log odds	Probability	Odds
0.0	0.5	1:1
1.38	0.8	4:1
-1.38	0.2	1:4
-2.94	0.05	?
-4.59	0.01	?

Intuition: log odds ratio

Logistic Function $\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$



Log odds	Probability	Odds
0.0	0.5	1:1
1.38	0.8	4:1
-1.38	0.2	1:4
-2.94	0.05	1:19
-4.59	0.01	1:99

Interpreting regressions: prediction (first line), summary (next lines)

Canvas > Modules > 2. Regression and Linear Models

Review how to apply in examples & derive the things in this table!

Midterm Fall 2023 - Review Topics.pdf

Download Midterm Fall 2023 - Review Topics.pdf (76.7 KB) | Alternative formats

Model	**Regression Interpretation**
Linear $y = \alpha + \beta x$	If $x=0$, $y = \alpha$ 1 unit change in x is associated with a β unit change in y
Linear-log $y = \alpha + \beta \ln(x)$	If $x=1$, $y = \alpha$ If x is multiplied by *e*, we expect a β unit change in y 1% change in x is associated with a $0.01*\beta$ unit change in y
Log-linear $\ln(y) = \alpha + \beta x$	If $x=0$, $y = e^{\alpha}$ For a 1 unit change in x, we expect y to be multiplied by e^{β} 1 unit change in x is associated with a $100*(\exp(\beta)-1)\%$ change in y
Log-log	If $x=1$, $y = e^{\alpha}$

Admin

- Phase 5 due Dec 4th

- Final exam on Dec 10th at 2pm
  - You may bring one double-sided 8x11" cheat sheet
  - No calculators will be allowed

Admin

- Canvas > Modules
  - Discussion 13: Final exam from Fall 2022 for practice
  - *4. Real World Applications*: List of topics in the 2nd half of the class to review
    - (in conjunction with the midterm review list, posted in *2. Regression and Linear Models*)

Goodbye from Instructor Thalken!

- Used 2950 skills to get a new paper published
- Will be on a free trip to Singapore to go to a conference during next lecture!

Next Class

- Final exam review, continued!