foundations of data science for everyone

VI: Logistic Regression

AUTHOR AND LECTURER: Farid Qamar

this slide deck: https://slides.com/faridqamar/fdfse_6

O

optimizing the objective function

recall:

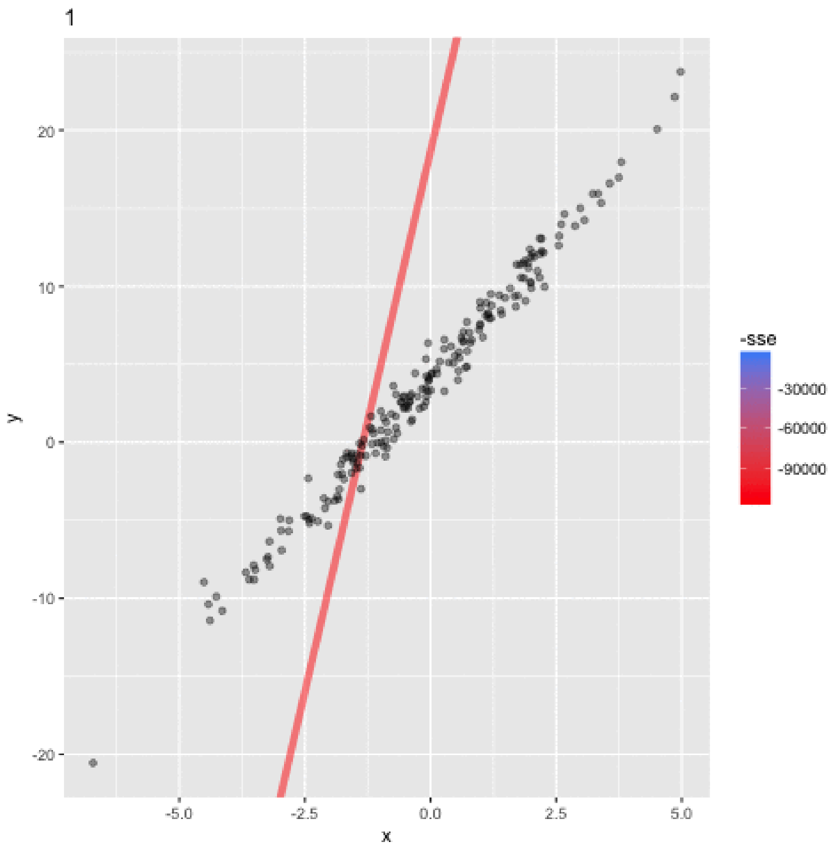
what is a model?

in the ML context:

a model is a low dimensional representation
of a higher dimensionality dataset

what is a machine learning?

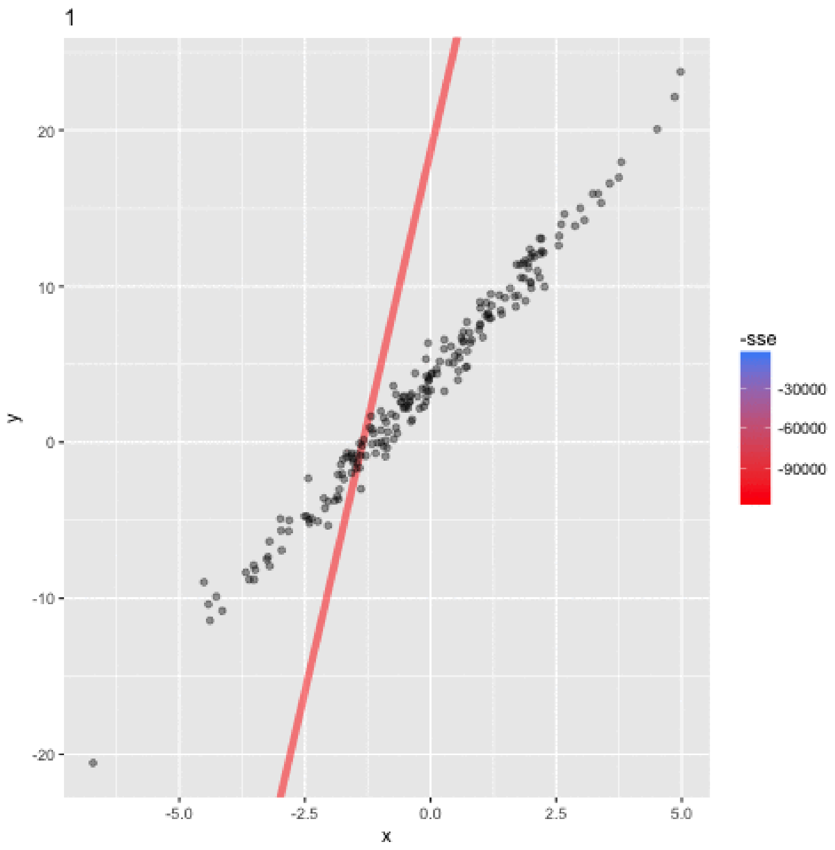
ML: Any model with parameters learned from the data



what is a machine learning?

ML: Any model with parameters learned from the data

ML models are a parameterized representation of "reality" where the parameters are learned from finite sets (*samples*) of realizations of that reality (*population*)



how do we model?

1

Choose the model:

a mathematical formula to represent the behavior in the data

parameters

example: line model $y = a\,x + b$

how do we model?

1

Choose the model:

a mathematical formula to represent the behavior in the data

example: line model y = a x + b

parameters

Choose the hyperparameters:

parameters chosen **before** the learning process, which govern the model and training process

example: the *degree N* of the polynomial $y = \sum_{i=0}^{N} c_i x^i$

how do we model?

2 Choose an objective function:

in order to find the "best" parameters of the model: we need to "optimize" a function.

We need something to be either **MINIMIZED** or **MAXIMIZED**

parameters

example:

line model: y = a x + b

objective function: sum of residual squared (least square fit method)

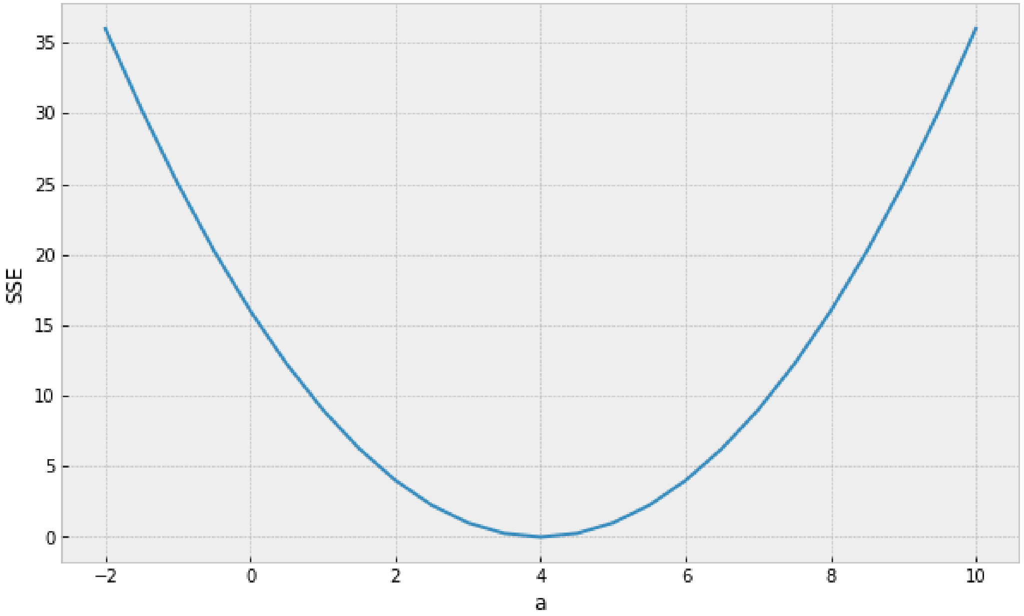
$$SSE = \sum (y_{i,observed} - y_{i,predicted})^2$$
$$SSE = \sum (y_{i,observed} - (ax_i + b))^2$$

we want to **minimize** *SSE* as much as possible

Optimizing the Objective Function

assume a simpler line model $y = ax$

($b = 0$) so we only need to find the "best" parameter a



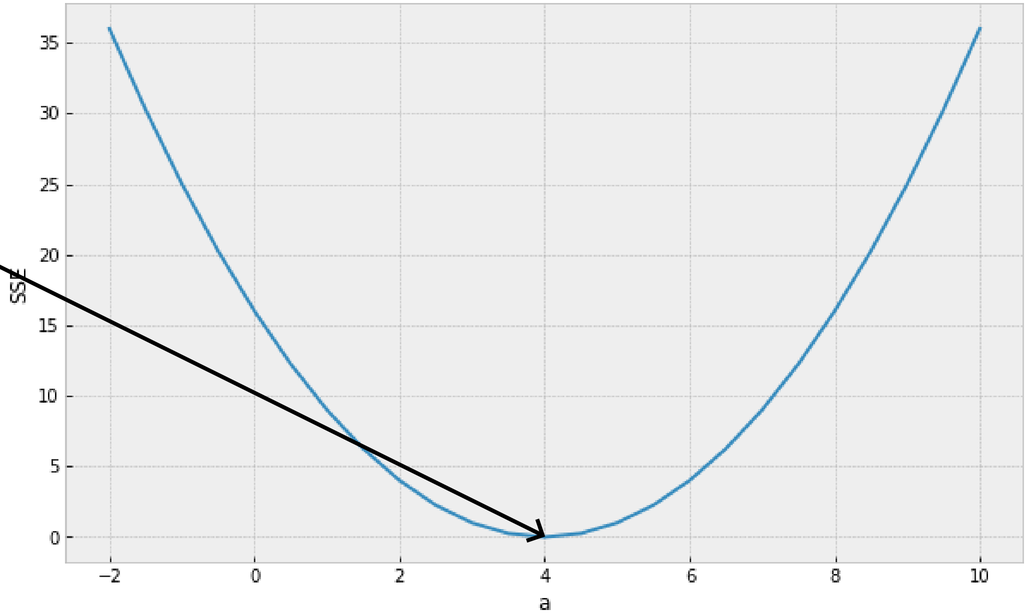
Optimizing the Objective Function

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

Minimum (optimal) SSE

➡ a = 4



Optimizing the Objective Function

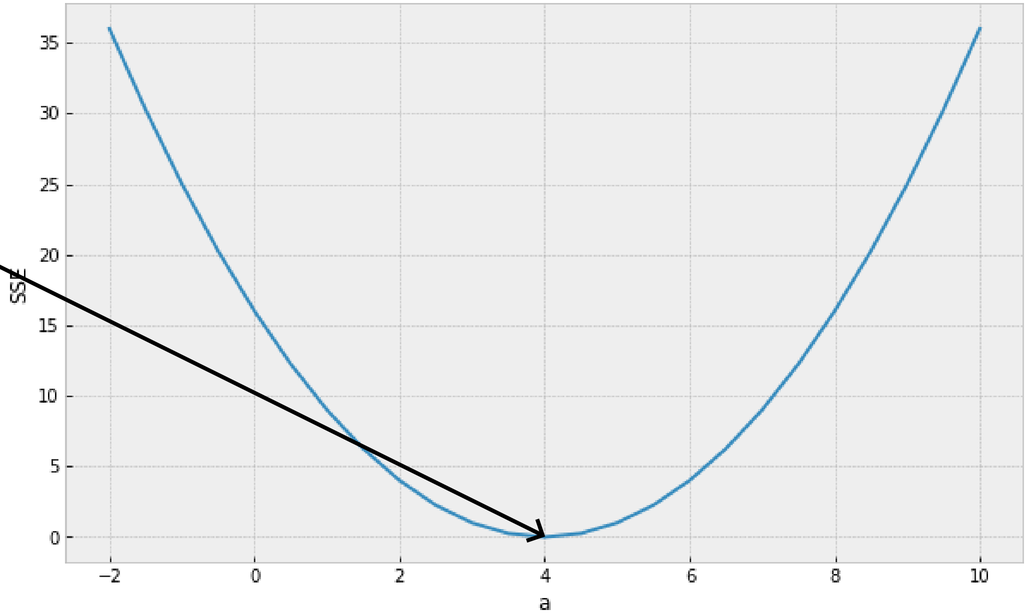
assume a simpler line model $y = ax$

($b = 0$) so we only need to find the "best" parameter a

Minimum (optimal) SSE

➡ a = 4

How do we find the minimum if we do not know beforehand how the SSE curve looks like?



1

regression vs classification

General purpose for ML

- to understand the structure of feature space
- **regression:** to predict unknown values based on known examples
- **classification:** to identify unknown classes based on known examples
- feature importance: to understand which features are important for the success of the model

Regression example:

Linear Regression

find the optimal parameters (**slope/coefficients** and **intercept**) of a linear model that best combine the **features** (independent variables) to describe the **target** (dependent variable)

Regression example:

Linear Regression

find the optimal parameters (**slope/coefficients** and **intercept**) of a linear model that best combine the **features** (independent variables) to describe the **target** (dependent variable)

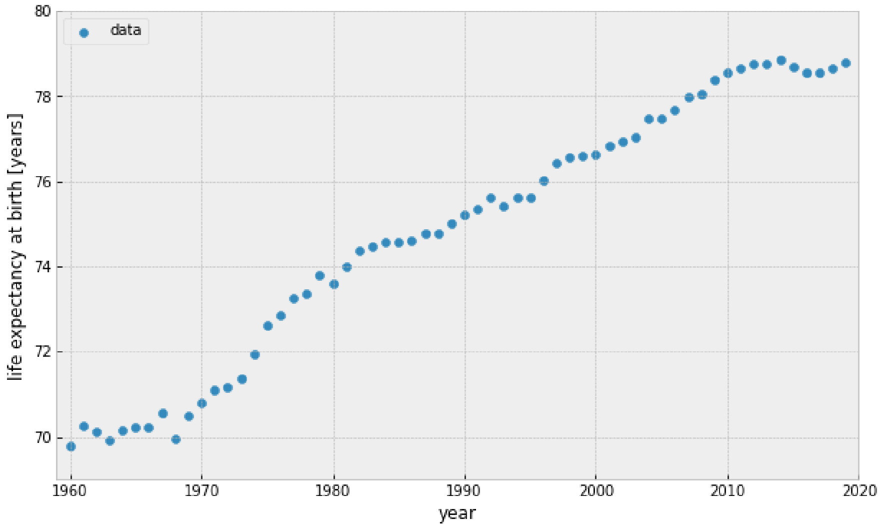
World Bank: Life expectancy at birth in the US

	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
	⋮	
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

Regression example:

Linear Regression

find the optimal parameters (**slope/coefficients** and **intercept**) of a linear model that best combine the **features** (independent variables) to describe the **target** (dependent variable)

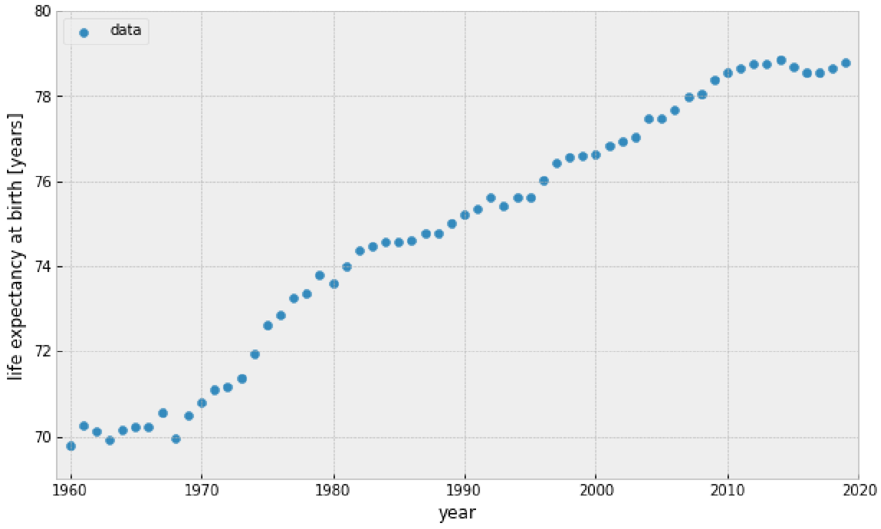


Regression example:

Linear Regression

find the optimal parameters (**slope/coefficients** and **intercept**) of a linear model that best combine the **features** (independent variables) to describe the **target** (dependent variable)

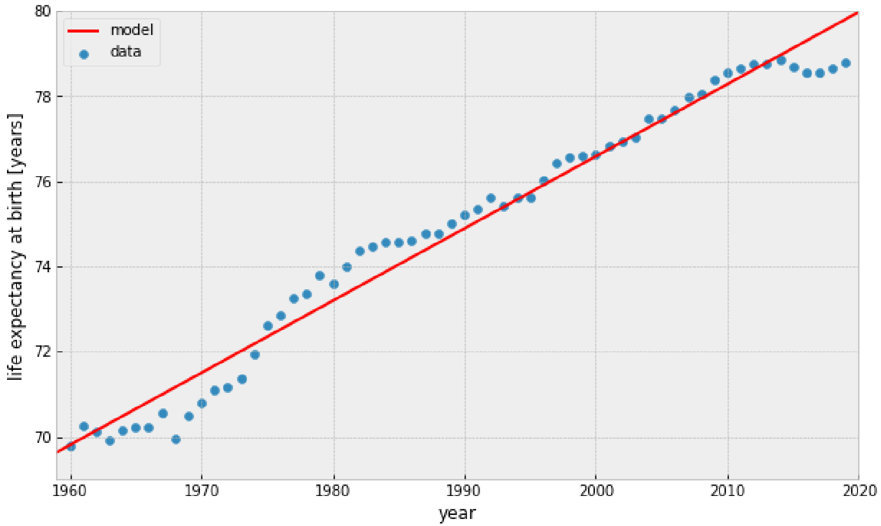
line model y = ax + b



Regression example:

Linear Regression

line model y = ax + b

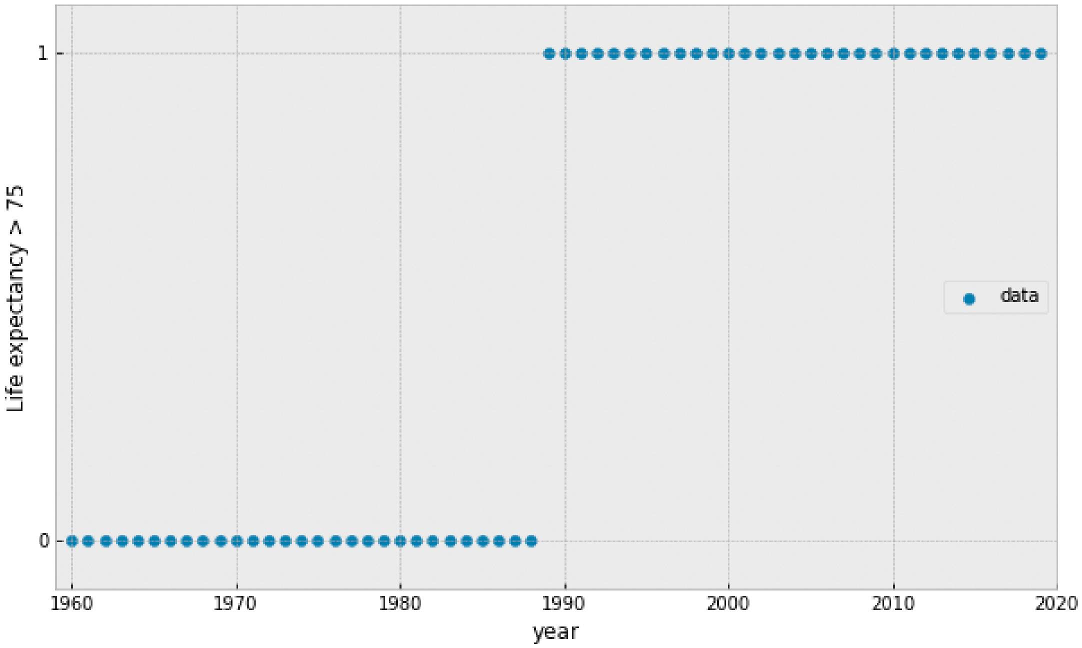


find the optimal parameters (**slope/coefficients** and **intercept**) of a linear model that best combine the **features** (independent variables) to describe the **target** (dependent variable)

what if our data is represented by a binary value?

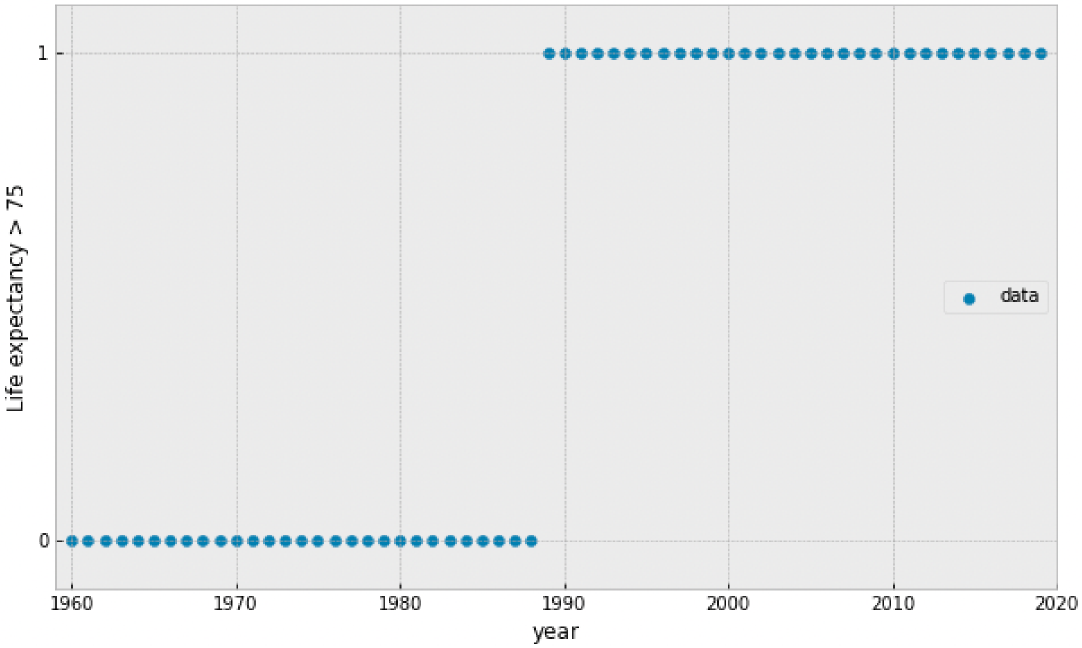
	year	75+
0	1960	0
1	1961	0
2	1962	0
3	1963	0
4	1964	0
5	1965	0
6	1966	0
⋮		
54	2014	1
55	2015	1
56	2016	1
57	2017	1
58	2018	1
59	2019	1

what if our data is represented by a binary value?



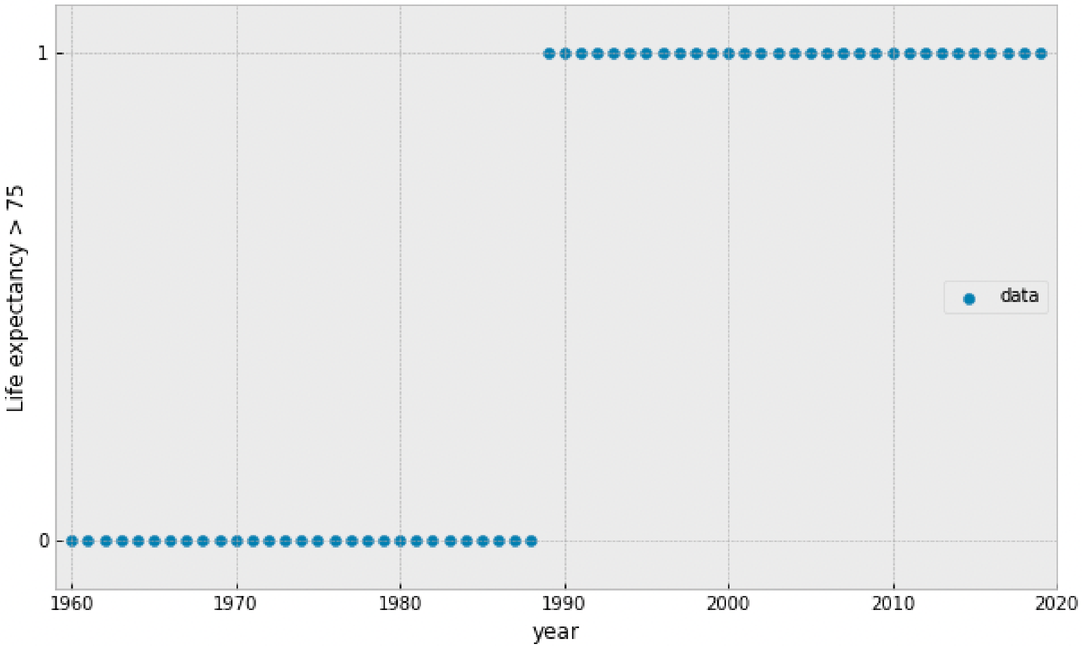
what if our data is represented by a <u>binary</u> value?

try fitting a linear model...



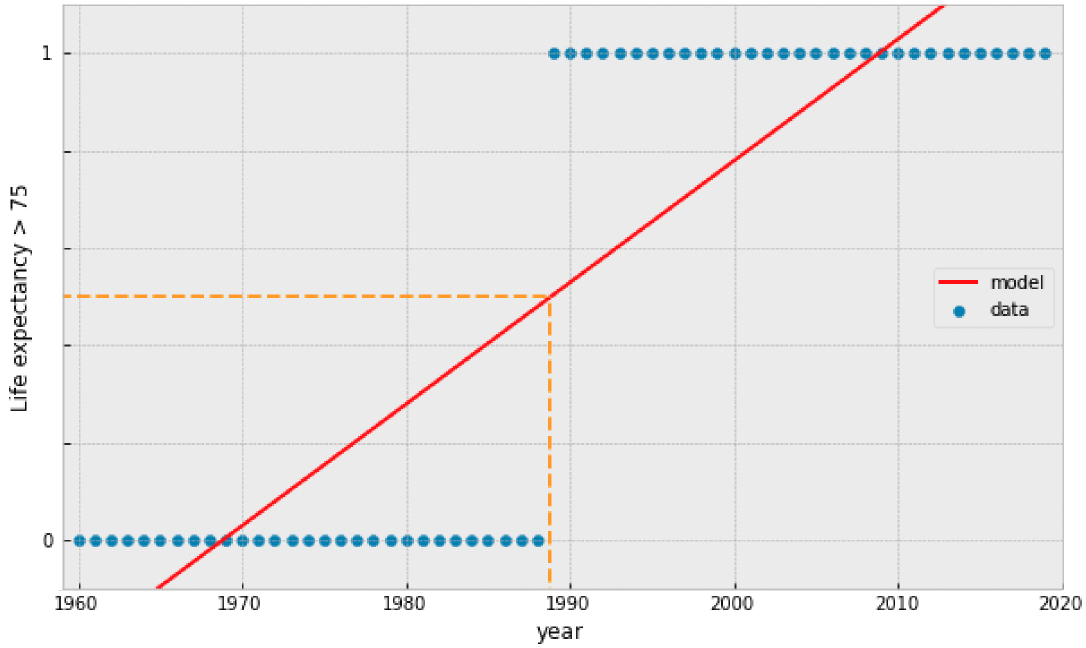
what if our data is represented by a binary value?

try fitting a linear model...



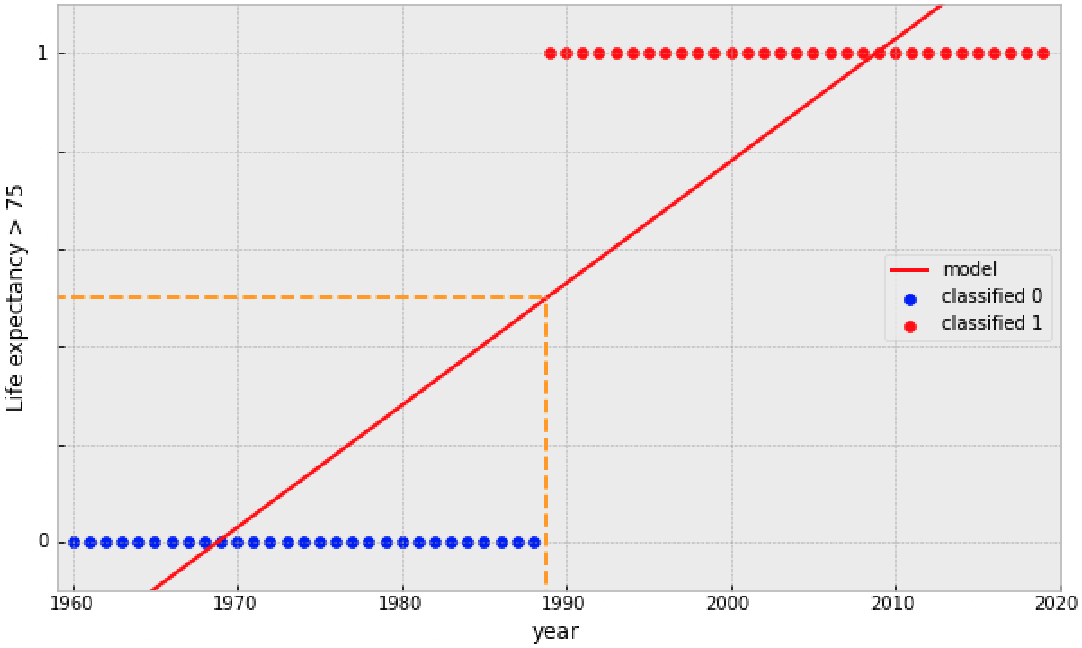
what if our data is represented by a <u>binary</u> value?

try fitting a linear model...



what if our data is represented by a <u>binary</u> value?

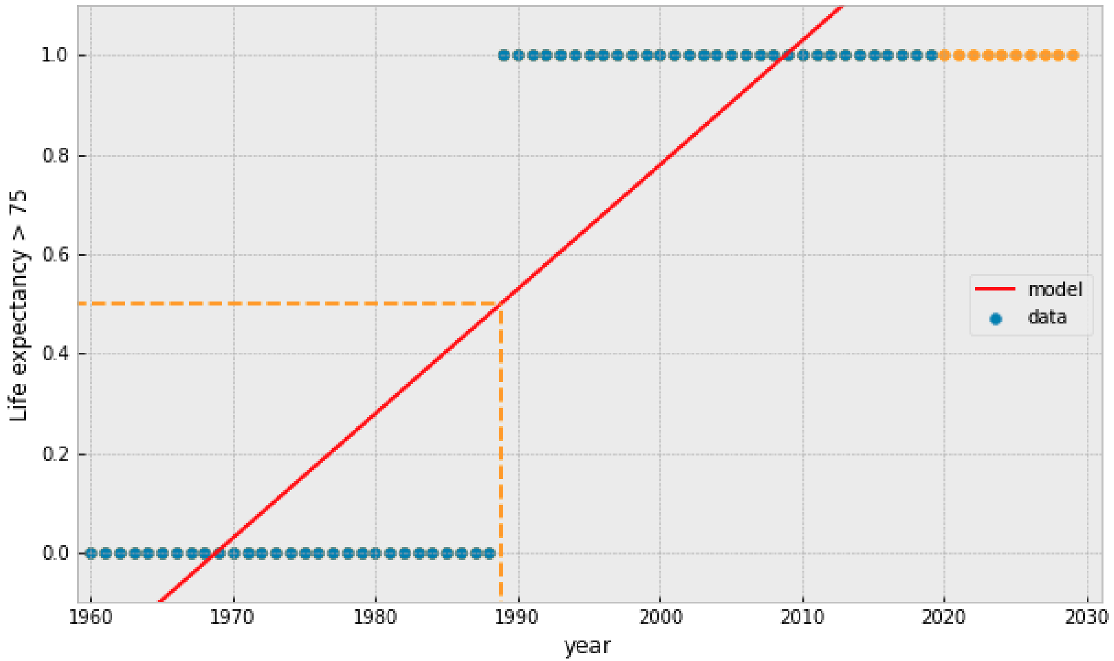
try fitting a linear model...



what if our data is represented by a binary value?

try fitting a linear model...

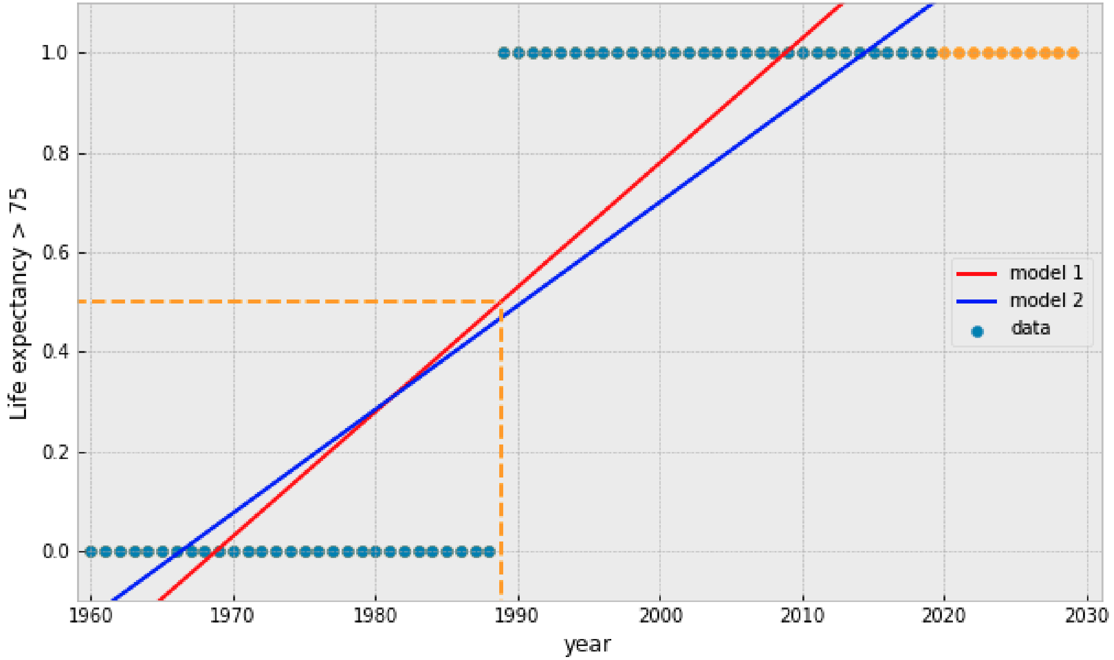
what if we add more data points?



what if our data is represented by a <u>binary</u> value?

try fitting a linear model...

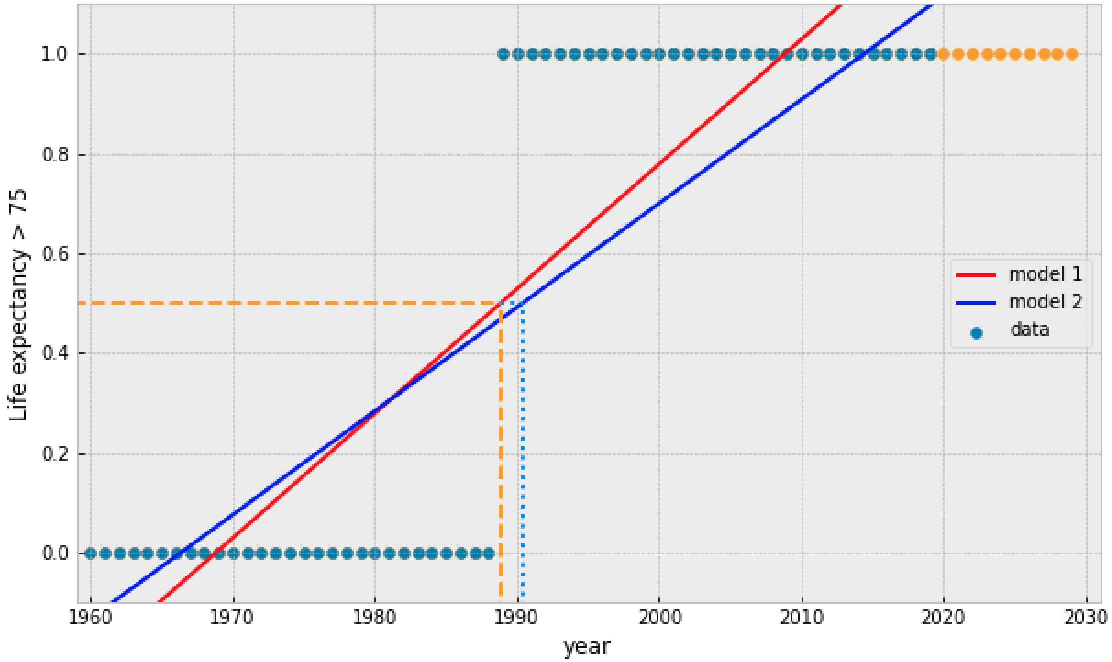
what if we add more data points?



what if our data is represented by a binary value?

try fitting a linear model...

what if we add more data points?



2

logistic regression

the **Logistic Function**:

$$f(x) = \frac{1}{1+e^{-z}} \quad ; \quad z = ax + b$$

interpreted as the
probability that the target
is True (= 1)

Objective Function:

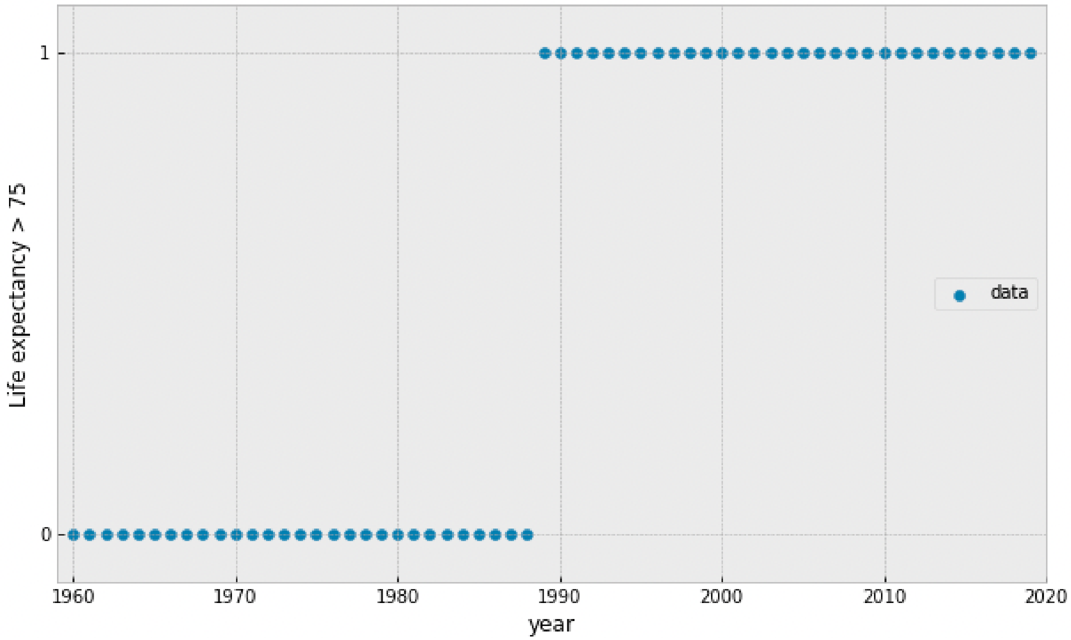
Log-likelihood

$$\log(\mathscr{L}) = \sum (y_i \log(f) + (1 - y_i) \log(1 - f))$$

the **Logistic Function**:

$$f(x) = \frac{1}{1+e^{-z}} \quad ; \quad z = ax + b$$

interpreted as the probability that the target is True (= 1)



Objective Function:

Log-likelihood

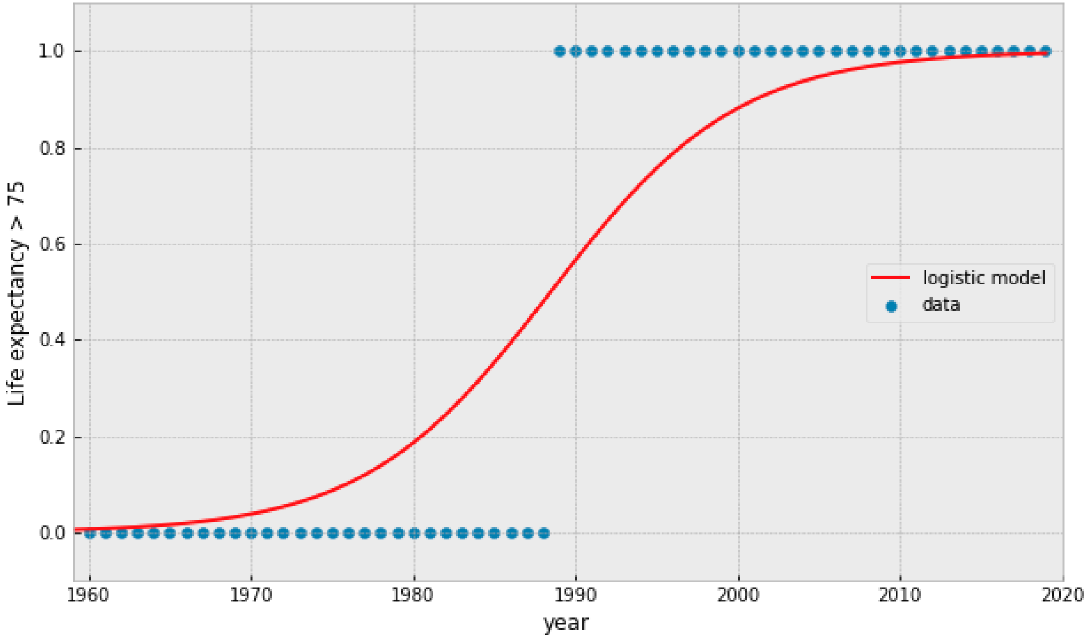
$$\log(\mathscr{L}) = \sum (y_i \log(f) + (1 - y_i) \log(1 - f))$$

the **Logistic Function**:

(Sigmoid)

$$f(x) = \frac{1}{1+e^{-z}} \quad ; \quad z = ax + b$$

interpreted as the probability that the target is True (= 1)



Objective Function:

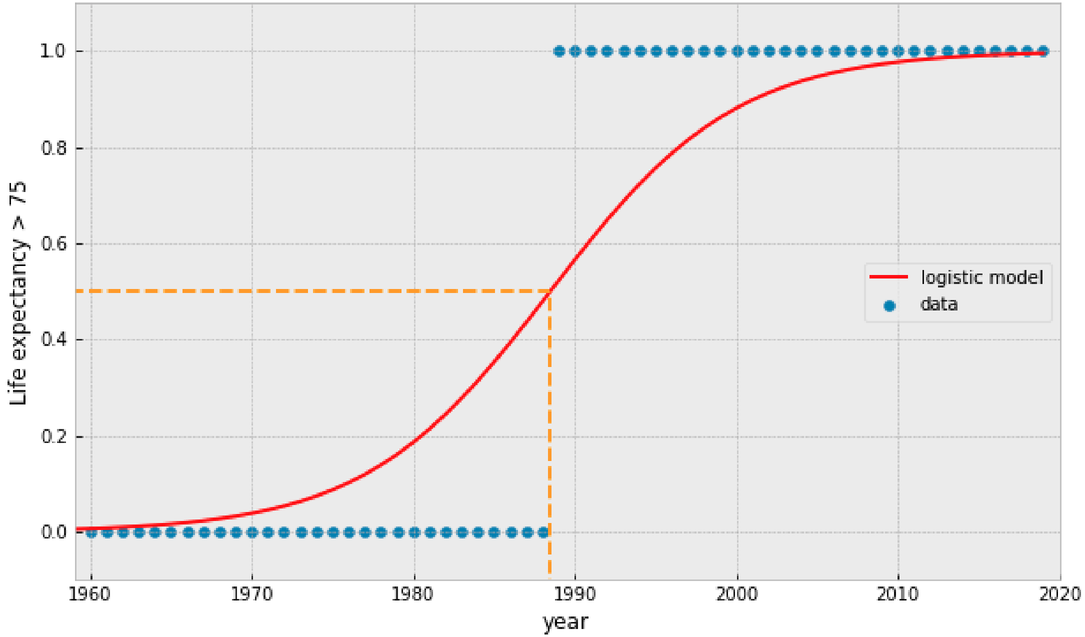
Log-likelihood

$$\log(\mathscr{L}) = \sum(y_i \log(f) + (1 - y_i)\log(1 - f))$$

the **Logistic Function**:

$$f(x) = \frac{1}{1+e^{-z}} \quad ; \quad z = ax + b$$

interpreted as the probability that the target is True (= 1)



Objective Function:

Log-likelihood

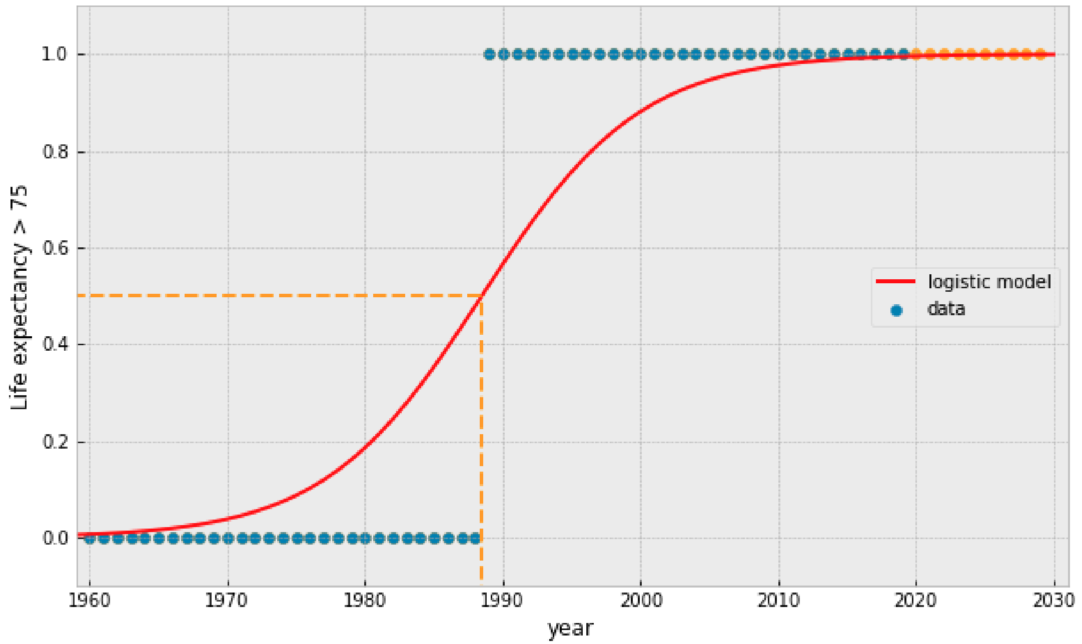
$$\log(\mathscr{L}) = \sum (y_i \log(f) + (1 - y_i) \log(1 - f))$$

the **Logistic Function**:

$$f(x) = \frac{1}{1+e^{-z}} \quad ; \quad z = ax + b$$

interpreted as the probability that the target is True (= 1)

what if we add more data points?

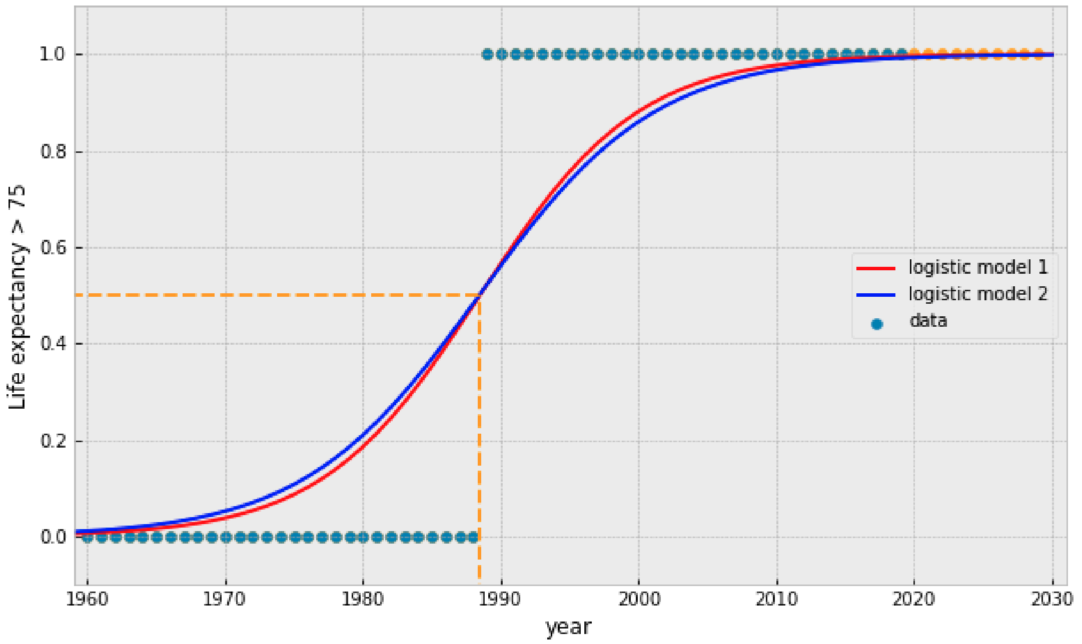


the **Logistic Function**:

$$f(x) = \frac{1}{1+e^{-z}} \quad ; \quad z = ax + b$$

interpreted as the probability that the target is True (= 1)

what if we add more data points?



3 classification model evaluation

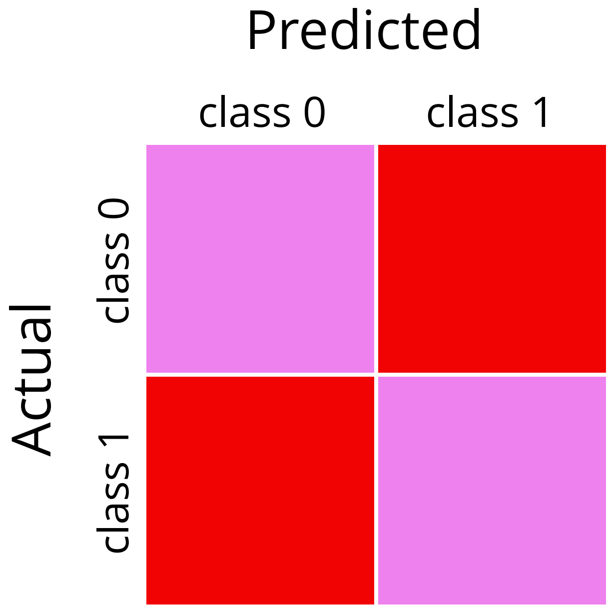
Confusion Matrix

indicates the model's "confusion" between classification outcomes

smaller off-diagonal elements & larger diagonal elements

=

model more effective at correctly labeling classes



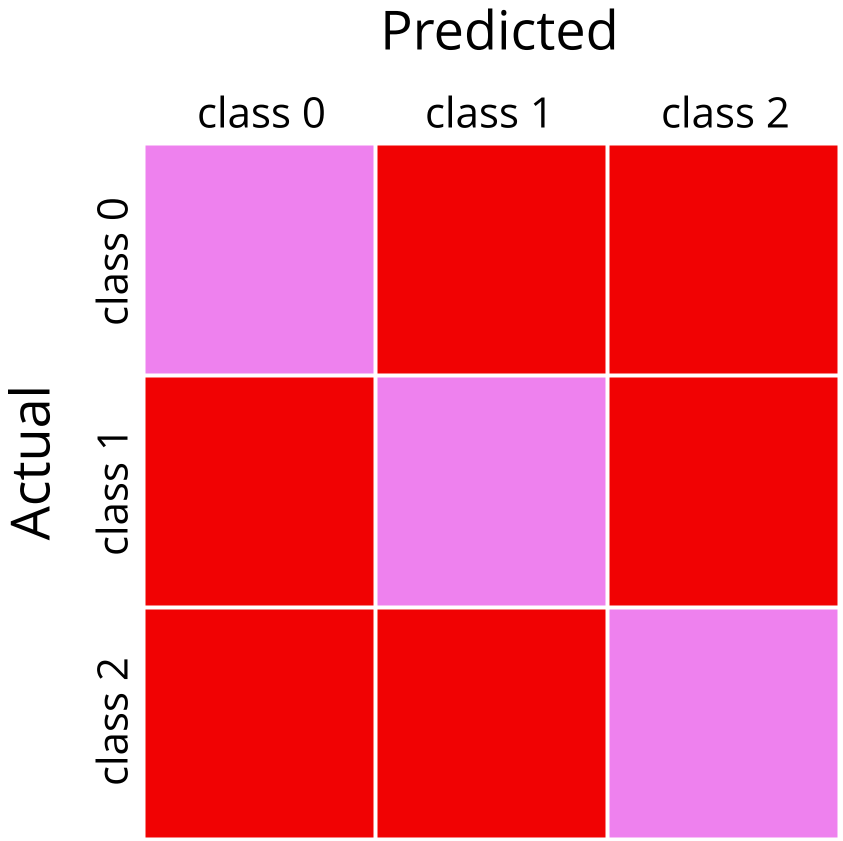
Confusion Matrix

indicates the model's "confusion" between classification outcomes

smaller off-diagonal elements & larger diagonal elements

=

model more effective at correctly labeling classes



Confusion Matrix

for example...

model predicting 500 objects:

indicates the model's "confusion" between classification outcomes

smaller off-diagonal elements &

larger diagonal elements

=

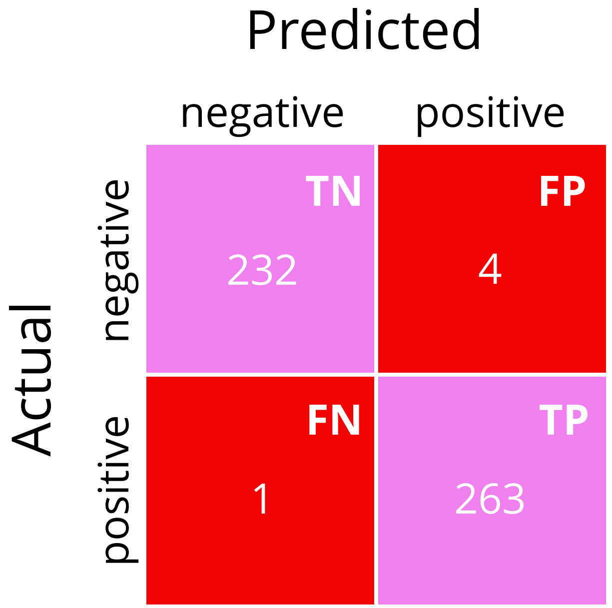
model more effective at correctly labeling classes

		Predicted	
		negative	positive
Actual	negative	232	4
	positive	1	263

True/False Positives/Negatives

Classification outcomes:

- true positives (TP) : "+" correctly labeled as "+"
- true negatives (TN) : "-" correctly labeled as "-"
- false positives (FP) : "-" incorrectly labeled as "+"
- false negatives (FN) : "+" incorrectly labeled as "-"



Accuracy

Classification outcomes:

- true positives (TP) : "+" correctly labeled as "+"
- true negatives (TN) : "-" correctly labeled as "-"
- false positives (FP) : "-" incorrectly labeled as "+"
- false negatives (FN) : "+" incorrectly labeled as "-"

accuracy: $\frac{TP+TN}{N} = \frac{TP+TN}{TP+FP+TN+FN}$

		Predicted	
		negative	positive
Actual	negative	TN 232	FP 4
	positive	FN 1	TP 263

accuracy = $\frac{232+263}{500} = 99\%$

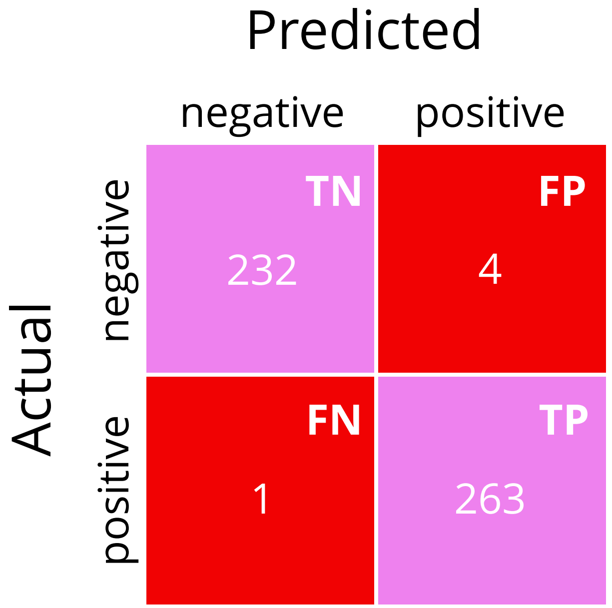
Precision and Recall

Classification outcomes:

- true positives (TP) : "+" correctly labeled as "+"
- true negatives (TN) : "-" correctly labeled as "-"
- false positives (FP) : "-" incorrectly labeled as "+"
- false negatives (FN) : "+" incorrectly labeled as "-"

precision: $\frac{TP}{TP+FP}$

recall: $\frac{TP}{TP+FN}$



precision = $\frac{263}{263+4} = 98.5\%$

recall = $\frac{263}{263+1} = 99.6\%$

Precision and Recall

precision:
(or specificity)

$$\frac{TP}{TP+FP}$$

Fraction of objects you think are positive that actually are positive

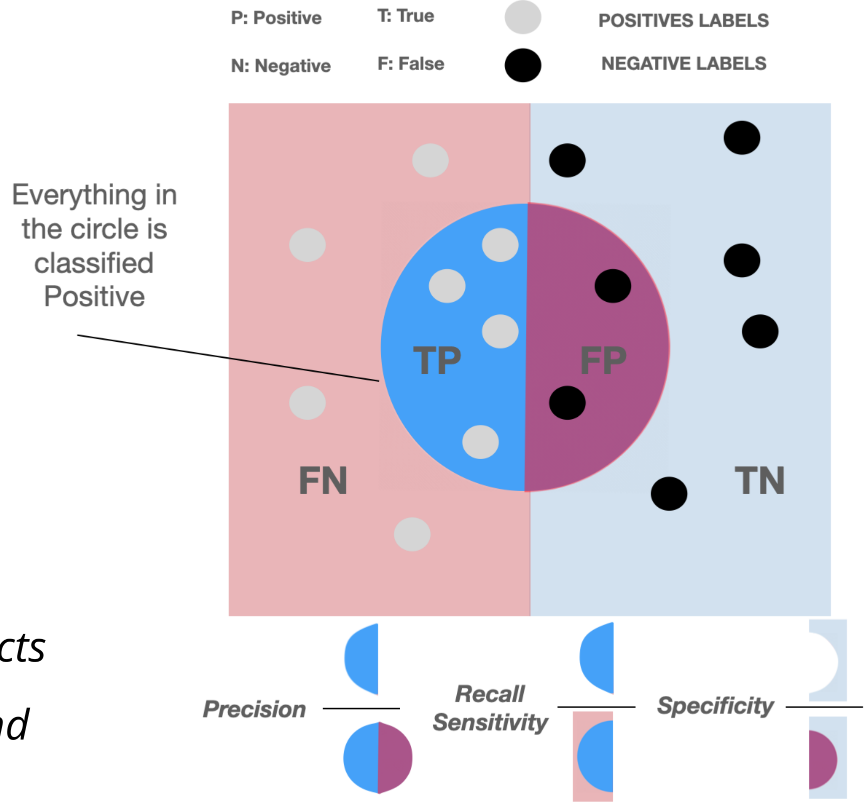
recall:
(or sensitivity)

$$\frac{TP}{TP+FN}$$

Fraction of positive objects that you were able to find

F1-score:

$$\frac{2\times\ \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$



https://en.wikipedia.org/wiki/Precision_and_recall

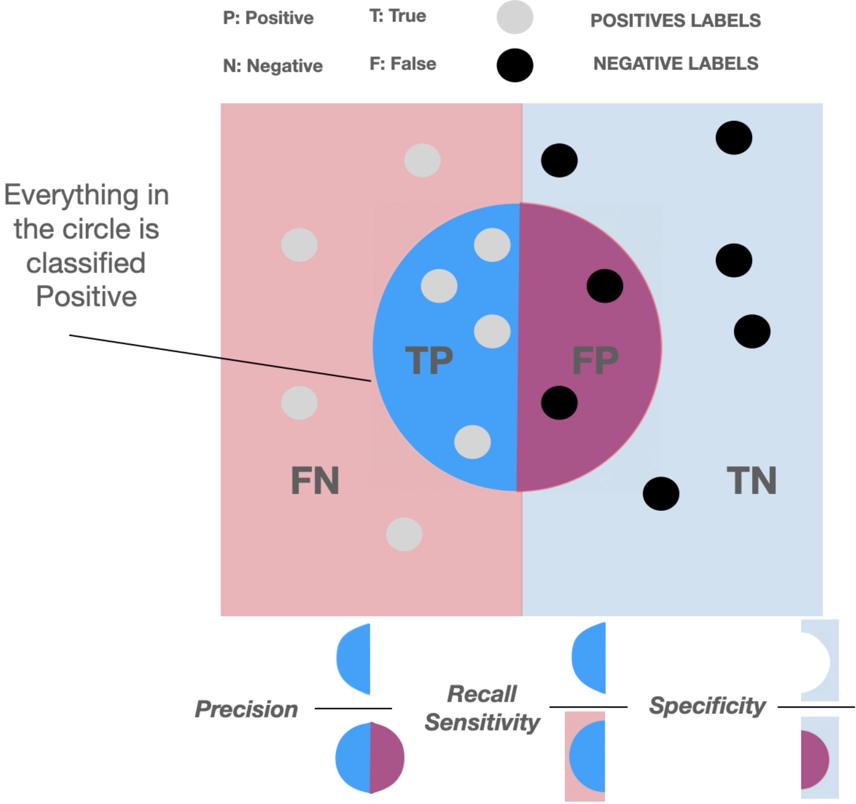
Current classifier accuracy: 50%

Precision?

Recall?

Specificity?

Sensitivity?



https://en.wikipedia.org/wiki/Precision_and_recall

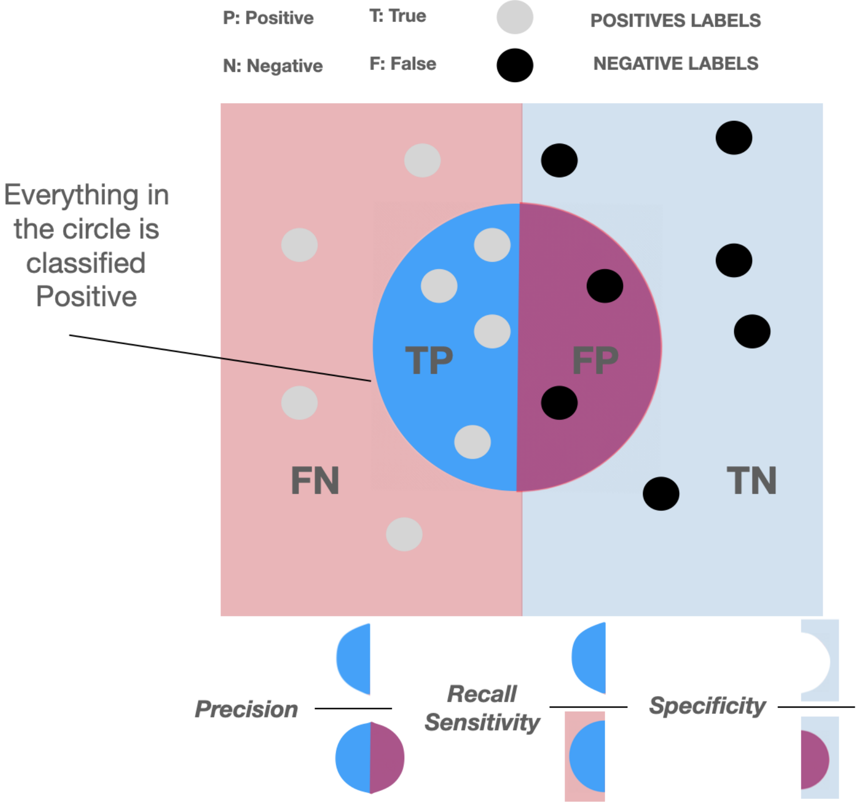
Current classifier accuracy: 50%

Precision = 4/6 = 0.7

Recall = 4/8 = 0.5

Specificity = 0.7

Sensitivity = 0.5



https://en.wikipedia.org/wiki/Precision_and_recall

4

encoding categorical variables

Categorical Variable

variable that can take a finite number of values.

spicies	**age**	**weight**
dog	7	32.3
bird	1	0.3
cat	3	8.1

Categorical Variable

variable that can take a finite number of values.

spicies	age	weight
dog	7	32.3
bird	1	0.3
cat	3	8.1

continuous

Categorical Variable

variable that can take a finite number of values.

spicies	age	weight
dog	7	32.3
bird	1	0.3
cat	3	8.1

ordinal

continuous

Categorical Variable

variable that can take a finite number of values.

spicies	age	weight
dog	7	32.3
bird	1	0.3
cat	3	8.1

continuous

categorical

ordinal

numerical encoding

change categorical to
(integer) numerical

spicies	age	weight
1	7	32.3
2	1	0.3
3	3	8.1

one-hot encoding

change each category to a binary

cat	bird	dog	age	weight
0	0	1	7	32.3
0	1	0	1	0.3
1	0	0	3	8.1

implies an order that does not exist

numerical encoding

change categorical to (integer) numerical

spicies	**age**	**weight**
1	7	32.3
2	1	0.3
3	3	8.1

dog=1, bird=2, cat=3

...dog < bird < cat... ??

ignores covariance between features

increases the dimensionality

problematic if you are interested in feature importance

↑

one-hot encoding

change each category to a binary

cat	**bird**	**dog**	**age**	**weight**
0	0	1	7	32.3
0	1	0	1	0.3
1	0	0	3	8.1

Definitely Preferred!

implies an order that does not exist

↑

numerical encoding

change categorical to (integer) numerical

spicies	age	weight
1	7	32.3
2	1	0.3
3	3	8.1

ignores covariance between features

increases the dimensionality

problematic if you are interested in feature importance

↑

one-hot encoding

change each category to a binary

cat	bird	dog	age	weight
0	0	1	7	32.3
0	1	0	1	0.3
1	0	0	3	8.1

5 normalization

Data can have covariance (and it almost always does!)

COVARIANCE = correlation / variance

axis 0 -> observations

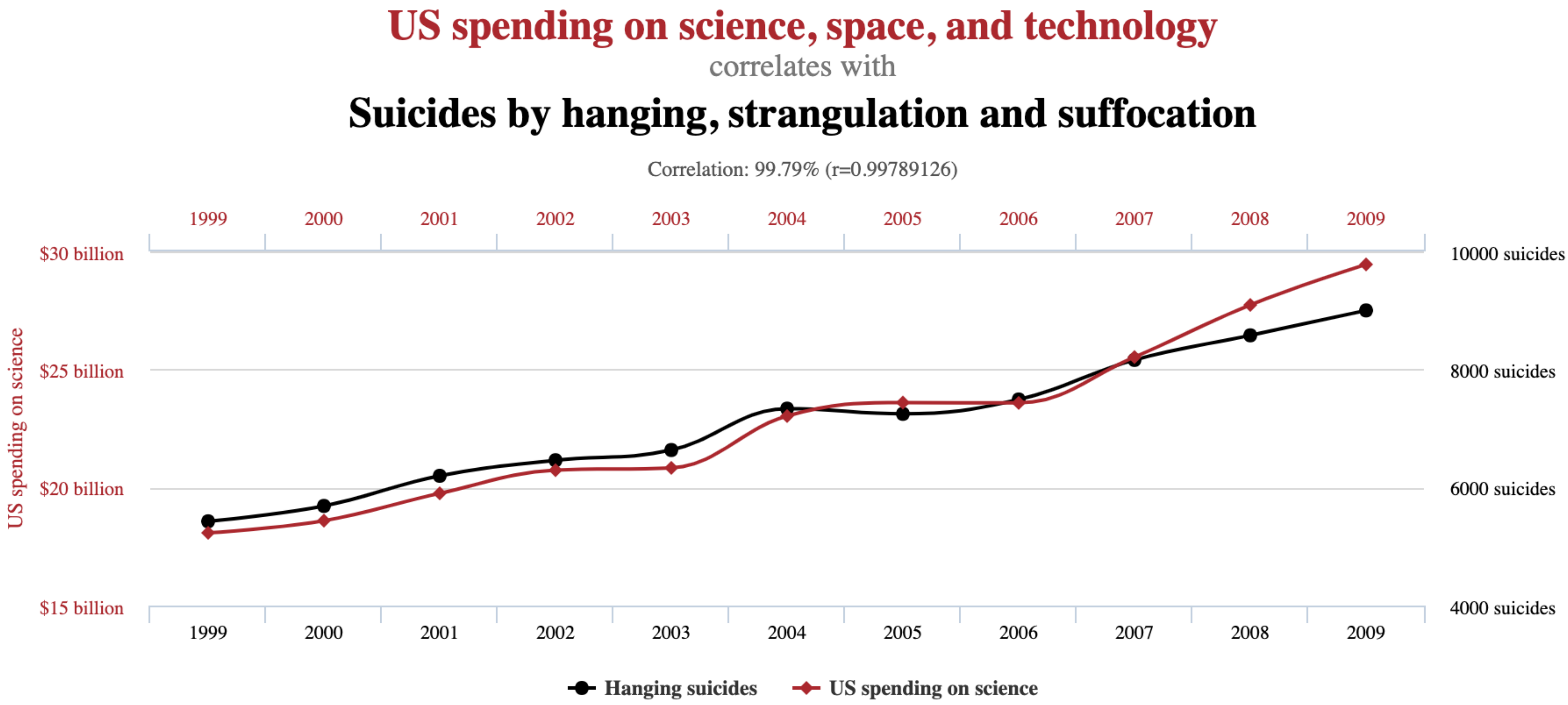
Clicca qui Meteo Italia oggi mercoledì 18 settembre

18/09/2019	TEMPO	PROB.PRECIP.	TEMP.(°C)	UMIDITÀ(%)	VENTO(KM/H)	RAFFICHE(KM/H)
02:00		-	21	57	7	20
05:00		-	20	57	7	16
08:00		-	20	61	5	14
11:00		-	23	55	7	16
14:00		50%	26	58	7	20
17:00		50%	22	62	22	50
20:00		10%	18	77	16	45
23:00		-	17	75	12	32

Bacheca – Parma domani mercoledì 18 settembre – meteoweek.com

axis 1 -> features

Data can have covariance (and it almost always does!)

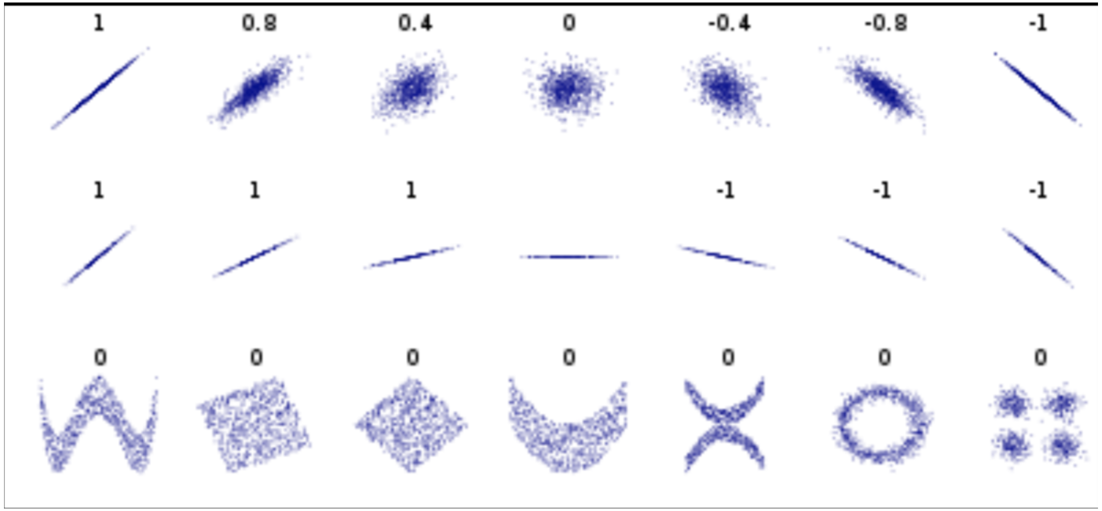


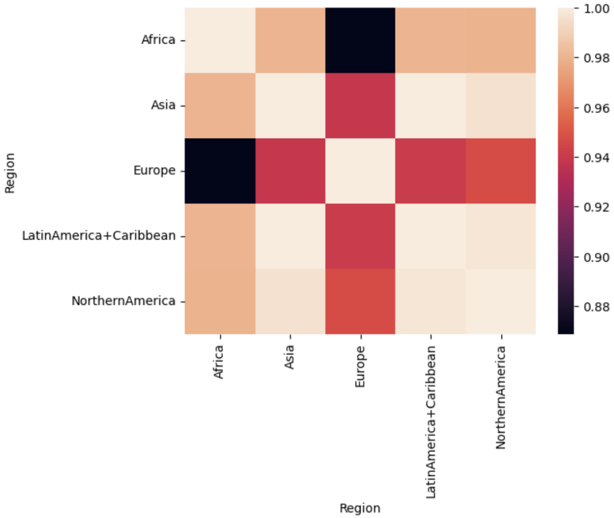
https://www.tylervigen.com/spurious-correlations

Data can have covariance (and it almost always does!)

Pearson's correlation (linear correlation)

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

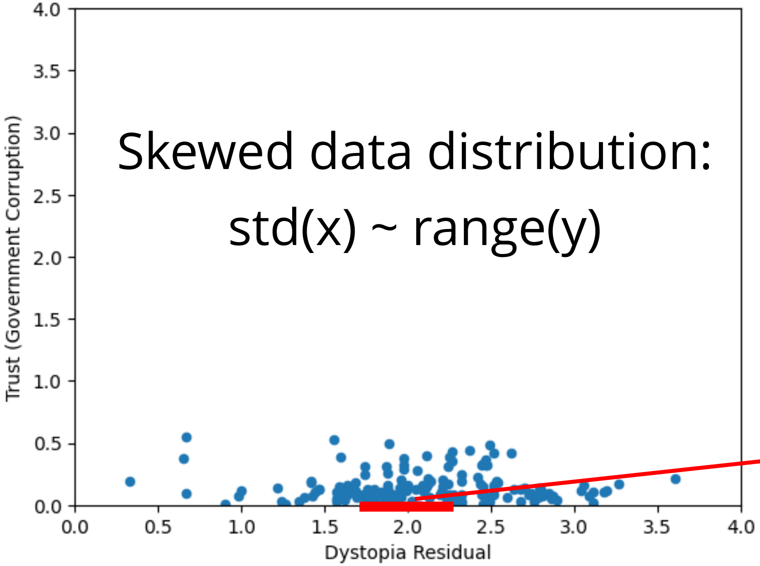




Generic preprocessing… WHY??

Worldbank Happyness Dataset https://github.com/fedhere/MLPNS_FBianco/blob/main/clustering/happiness_solution.ipynb

	Country	Region	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dy Re
0	Switzerland	Western Europe	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2
1	Iceland	Western Europe	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2
2	Denmark	Western Europe	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.



Skewed data distribution:

std(x) ~ range(y)

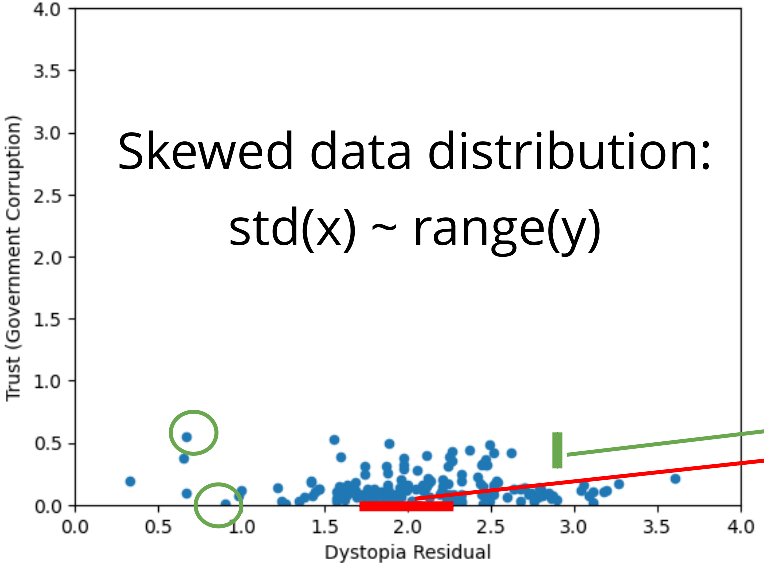
```
happiness15.describe()
```

	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual	year
count	160.000000	158.000000	160.000000	158.000000	160.000000	160.000000	160.000000	160.000000	158.000000	160.000000
mean	5.365756	0.047885	0.842979	0.991046	0.628037	0.428151	0.143023	0.236448	2.098977	2015.050000
std	1.141280	0.017146	0.402840	0.272369	0.246332	0.149803	0.119492	0.126605	0.553550	0.445805
min	2.839000	0.018480	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.328580	2015.000000
25%	4.517750	0.037268	0.539453	0.856823	0.437897	0.328630	0.061067	0.148800	1.759410	2015.000000
50%	5.203000	0.043940	0.901085	1.029510	0.695745	0.434635	0.107220	0.216130	2.095415	2015.000000
75%	6.193250	0.052300	1.155523	1.214405	0.809837	0.547057	0.179565	0.307547	2.462415	2015.000000
max	7.587000	0.136930	1.690420	1.402230	1.025250	0.669730	0.551910	0.795880	3.602140	2019.000000

Generic preprocessing... WHY??

Worldbank Happyness Dataset https://github.com/fedhere/MLPNS_FBianco/blob/main/clustering/happiness_solution.ipynb

	Country	Region	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dy Re
0	Switzerland	Western Europe	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2
1	Iceland	Western Europe	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2
2	Denmark	Western Europe	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.



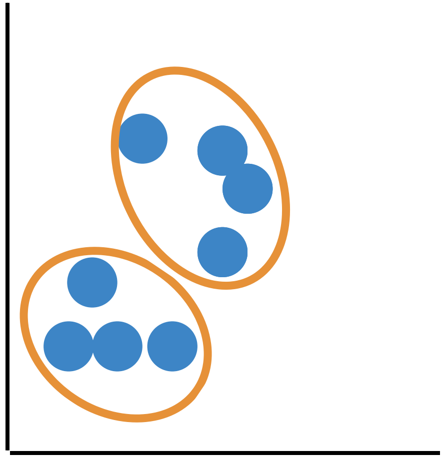

```
happiness15.describe()
```

	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual	year
count	160.000000	158.000000	160.000000	158.000000	160.000000	160.000000	160.000000	160.000000	158.000000	160.000000
mean	5.365756	0.047885	0.842979	0.991046	0.628037	0.428151	0.143023	0.236448	2.098977	2015.050000
std	1.141280	0.017146	0.402840	0.272369	0.246332	0.149803	0.119492	0.126605	0.553550	0.445805
min	2.839000	0.018480	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.328580	2015.000000
25%	4.517750	0.037268	0.539453	0.856823	0.437897	0.328630	0.061067	0.148800	1.759410	2015.000000
50%	5.203000	0.043940	0.901085	1.029510	0.695745	0.434635	0.107220	0.216130	2.095415	2015.000000
75%	6.193250	0.052300	1.155523	1.214405	0.809837	0.547057	0.179565	0.307547	2.462415	2015.000000
max	7.587000	0.136930	1.690420	1.402230	1.025250	0.669730	0.551910	0.795880	3.602140	2019.000000

unsupervised vs supervised learning

Unsupervised learning

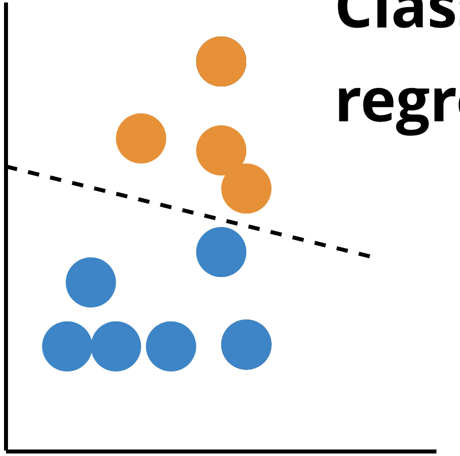
- understanding structure
- anomaly detection
- dimensionality reduction



Clustering

Supervised learning

- classification
- prediction
- feature selection

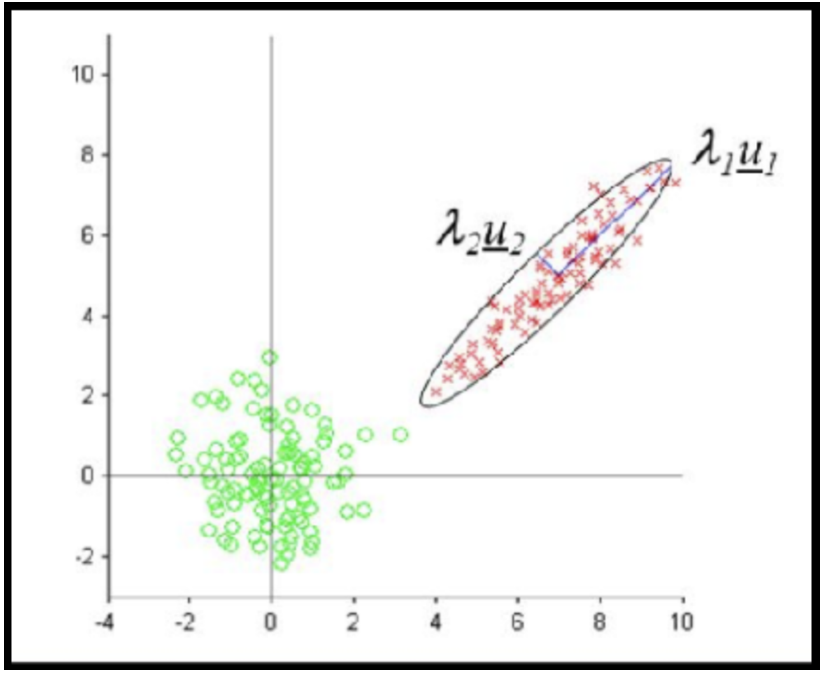


Classifying & regression

Generic preprocessing

Data can have covariance (and it almost always does!)

ORIGINAL DATA



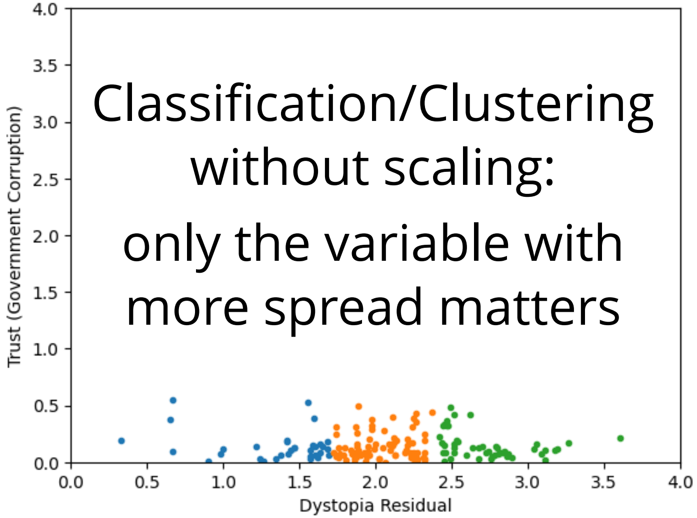
STANDARDIZED DATA

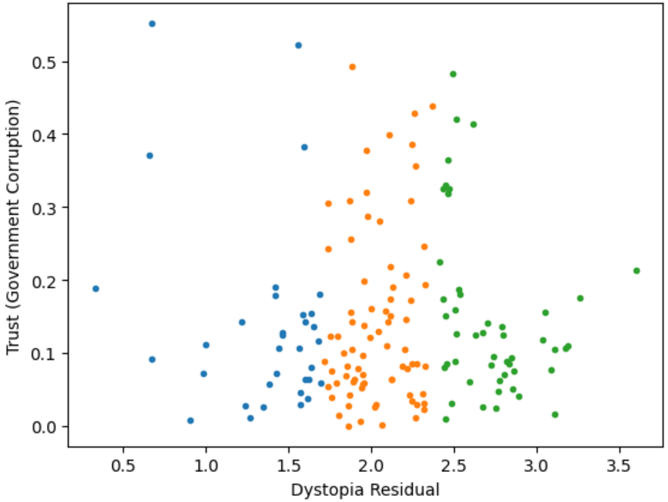
Data that is not correlated appear as a sphere in the Ndimensional feature space

Generic preprocessing... WHY??

Worldbank Happyness Dataset

	Country	Region	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity
0	Switzerland	Western Europe	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678
1	Iceland	Western Europe	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630
2	Denmark	Western Europe	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139

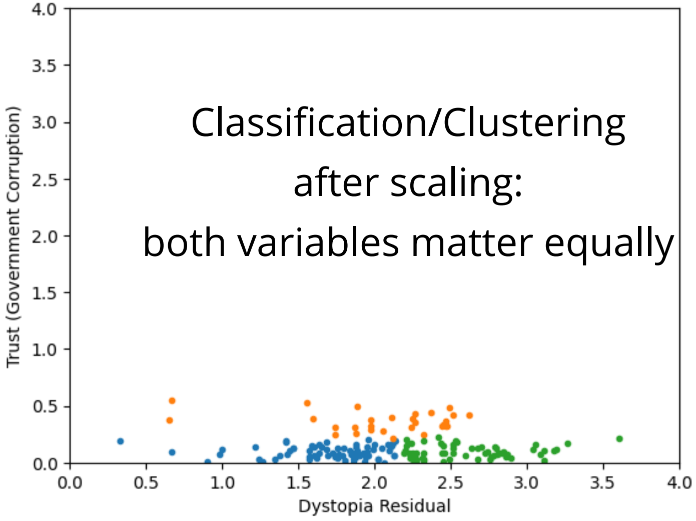


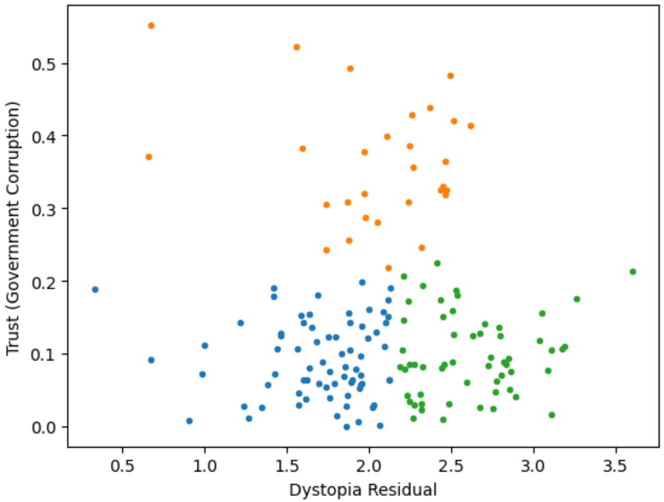


Generic preprocessing... WHY??

Worldbank Happyness Dataset

	Country	Region	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity
0	Switzerland	Western Europe	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678
1	Iceland	Western Europe	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630
2	Denmark	Western Europe	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139

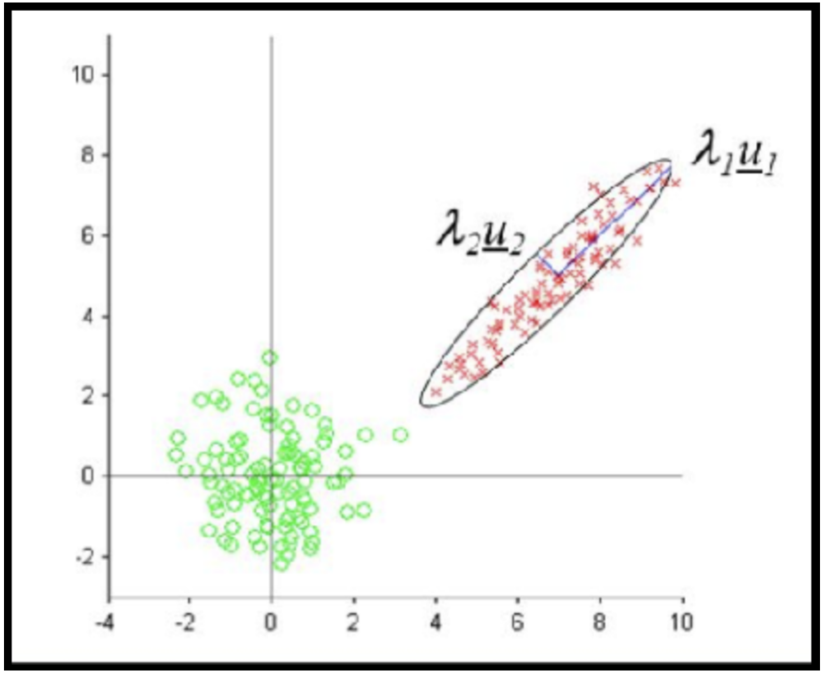




Generic preprocessing

Data can have covariance (and it almost always does!)

ORIGINAL DATA



STANDARDIZED DATA

Data that is not correlated appear as a sphere in the Ndimensional feature space

Generic preprocessing: most commo
will just correct for the spread and c

for each feature: divide by standard deviation and subtract mean

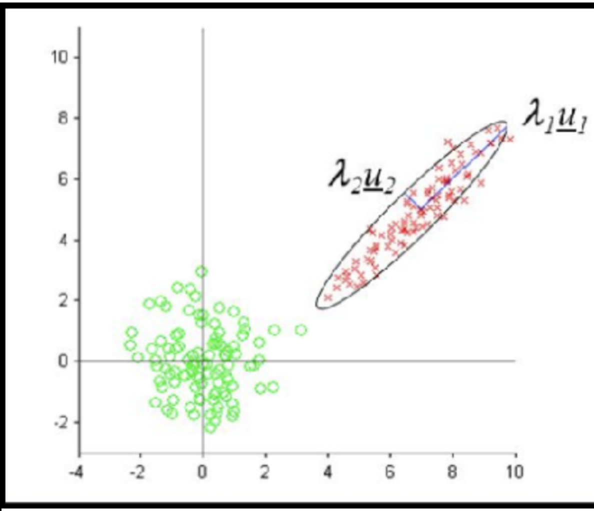
```
X = preprocessing.scale(X, axis=0)
```
Last executed 2018-12-12 09:35:39 in 46ms

```
X.mean(axis=0)
```
Last executed 2018-12-12 09:35:40 in 13ms

```
array([ 3.85590369e-16, -6.93196168e-17, -5.90549813e-16, -5.95882091e-16,
       -8.49165306e-16, -1.57568821e-15, -8.00508267e-16,  5.55890004e-16,
       -5.16564452e-16,  1.09378357e-15,  3.46598084e-16,  2.31954102e-16,
        2.78611537e-16, -2.51283611e-16,  8.66495210e-18,  3.03939858e-16,
       -3.66594127e-17, -9.27149875e-16, -6.39873386e-16,  2.93275302e-17,
        9.19817992e-17,  6.33208038e-18, -1.99960433e-17,  9.55144336e-16,
       -2.20623011e-16,  6.93196168e-17, -9.46479383e-17,  2.26621824e-16,
        6.93196168e-17,  2.32953905e-16])
```

```
X.std(axis=0)
```
Last executed 2018-12-12 09:36:28 in 19ms

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```



whitening

The term "whitening" refers to white noise,
i.e. noise with the same power at all
frequencies"

Data can have covariance (and it almost always does!)

PLUTO Manhattan data (42,000 x 15) correlation matrix

axis 0 -> observations

$$\Sigma = \begin{bmatrix} \mathrm{E}[(X_1-\mu_1)(X_1-\mu_1)] & \mathrm{E}[(X_1-\mu_1)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_1-\mu_1)(X_n-\mu_n)] \\ \mathrm{E}[(X_2-\mu_2)(X_1-\mu_1)] & \mathrm{E}[(X_2-\mu_2)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_2-\mu_2)(X_n-\mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n-\mu_n)(X_1-\mu_1)] & \mathrm{E}[(X_n-\mu_n)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_n-\mu_n)(X_n-\mu_n)] \end{bmatrix}.$$

axis 1 -> features

Data can have covariance (and it almost always does!)

PLUTO Manhattan data (42,000 x 15) correlation matrix

$$\Sigma = \begin{bmatrix} \mathrm{E}[(X_1-\mu_1)(X_1-\mu_1)] & 0 & \cdots & 0 \\ 0 & \mathrm{E}[(X_2-\mu_2)(X_2-\mu_2)] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & \mathrm{E}[(X_n-\mu_n)(X_n-\mu_n)] \end{bmatrix}.$$

A covariance matrix is diagonal if the data has no correlation

Full On Whitening

: remove covariance by dia
transforming the data with
diagonalizes the covaria

find the matrix W that diagonalized Σ

```
from zca import ZCA
import numpy as np
X = np.random.random((10000, 15)) # data
array
trf = ZCA().fit(X)
X_whitened = trf.transform(X)
X_reconstructed =
trf.inverse_transform(X_whitened)
```

Generic preprocessing: other common

for image processing (e.g. segmentation) often you need to mimmax preprocess

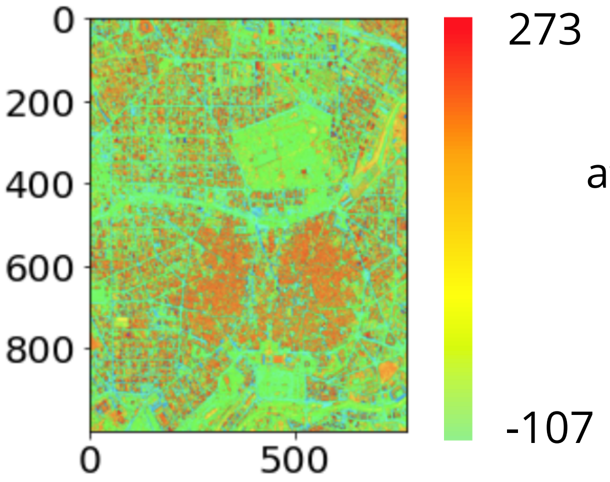
```
from sklearn import preprocessing

Xopscaled = preprocessing.minmax_scale(image_pixels.astype(float), axis=1)
Xopscaled.reshape(op.shape)[200, 700]
```

```
pl.imshow(Xopscaled.reshape(op.shape));
```

```
Clipping input data to the valid range for imshow with RGB data
```

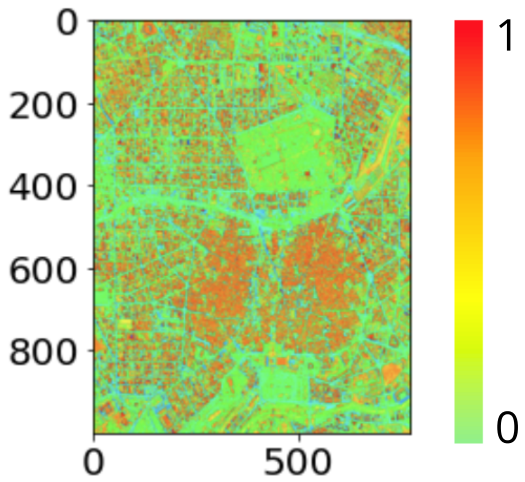
before



after (looks the same but colorbar different)

```
pl.imshow(Xopscaled.reshape(op.shape));
```

```
Clipping input data to the valid range fo
```



https://www.nature.com/articles/nmeth.3812.pdf

POINTS OF SIGNIFICANCE

Analyzing outliers: influential or nuisance?

Some outliers influence the regression fit more than others.

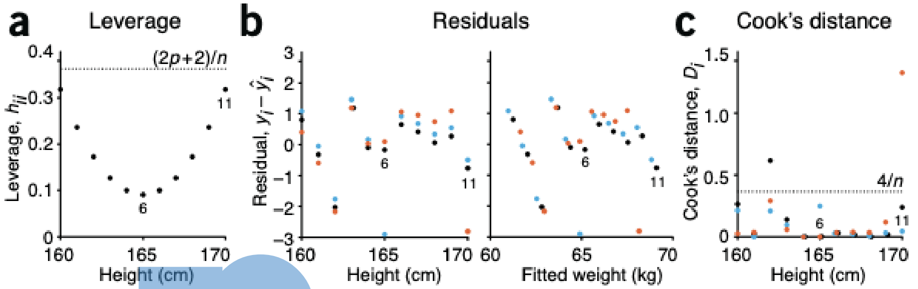


Figure 2 | The leverage, residual and Cook's distance of an observation are used to assess the robustness of the fit. (**a**) The leverage of an observation tells us about its potential to influence the fit and increases as the square

reading

https://github.com/fedhere/FDSFE_FBianco/blob/main/HW5/Multiple_Linear_Regression.ipynb

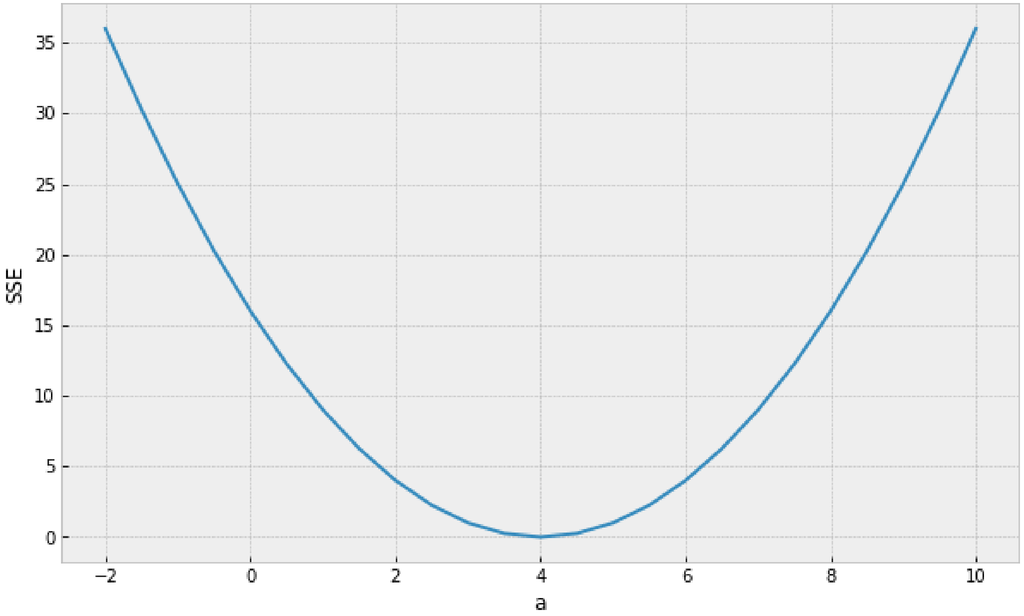
HW

1.1 stochastic gradient descent (SGD)

the algorithm: **Stochastic Gradient Descent (SGD)**

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

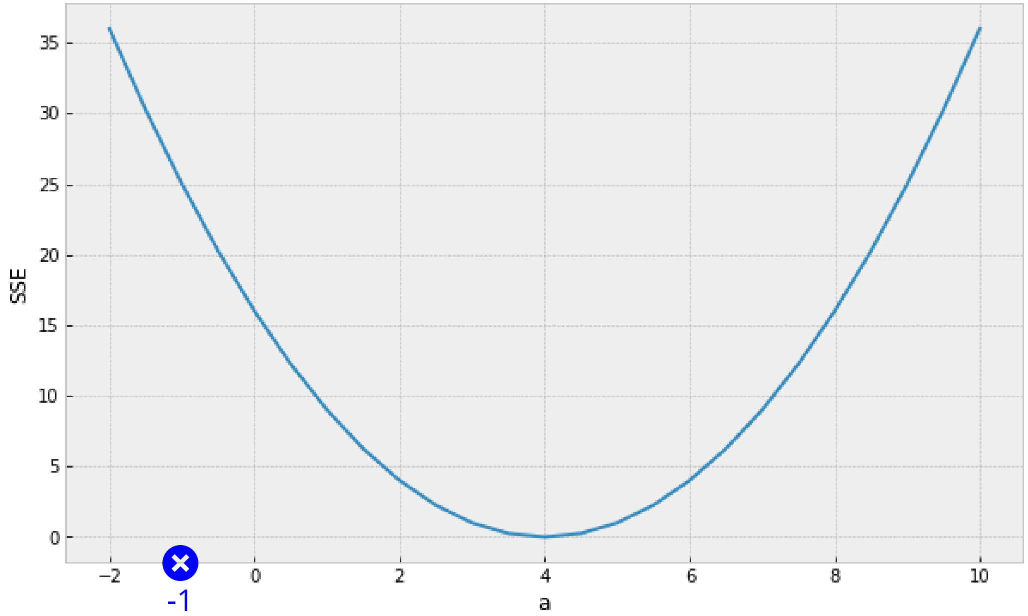


the algorithm: ~~Stochastic~~ **Gradient Descent**

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a

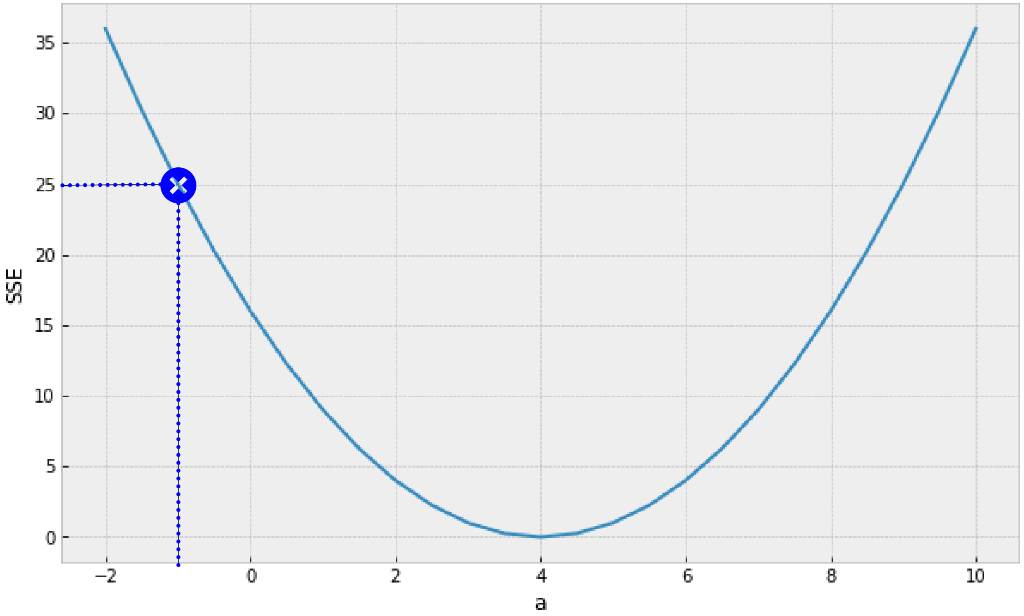


the algorithm: ~~Stochastic~~ **Gradient Descent**

assume a simpler line model $y = ax$

($b = 0$) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE

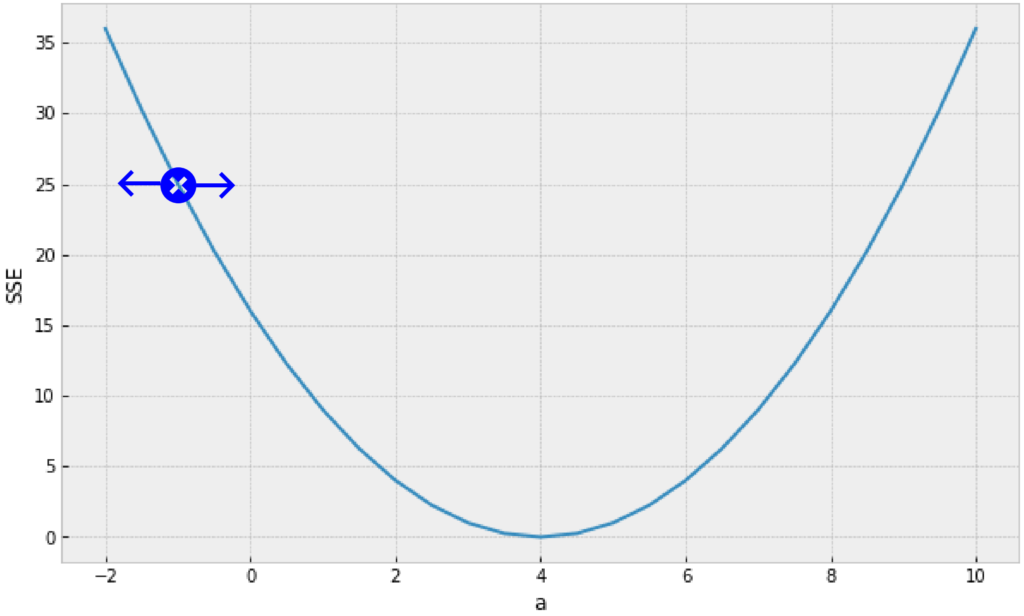


the algorithm: ~~Stochastic~~ Gradient Descent

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE

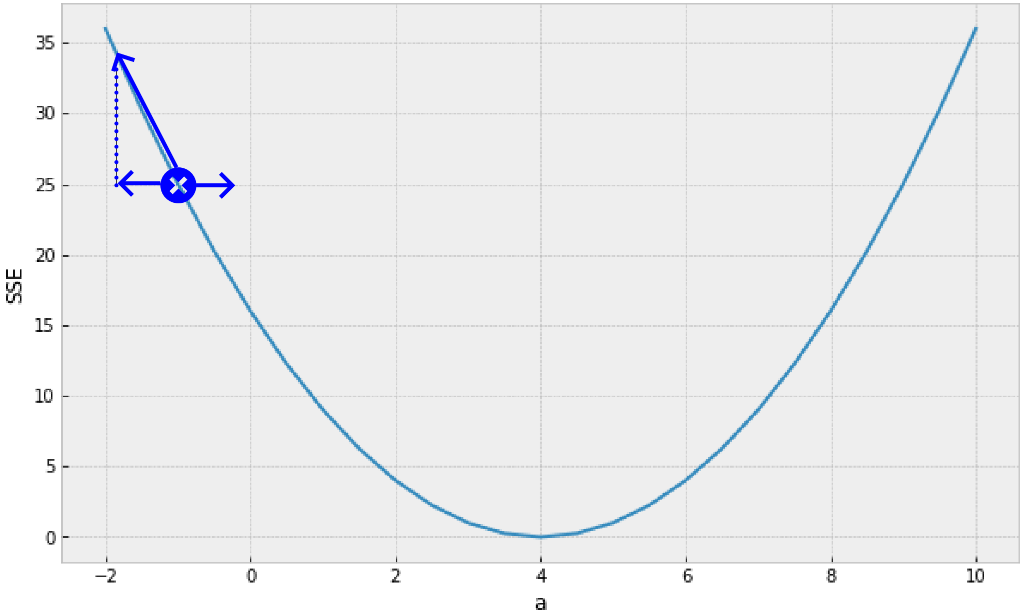


the algorithm: ~~Stochastic~~ **Gradient Descent**

assume a simpler line model $y = ax$

($b = 0$) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE

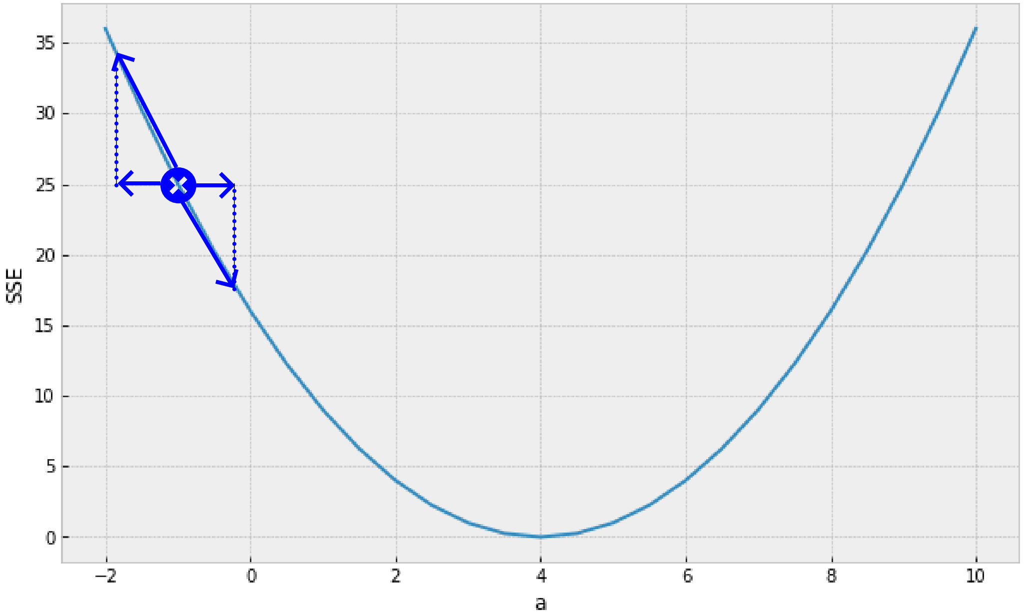


the algorithm: ~~Stochastic~~ **Gradient Descent**

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE

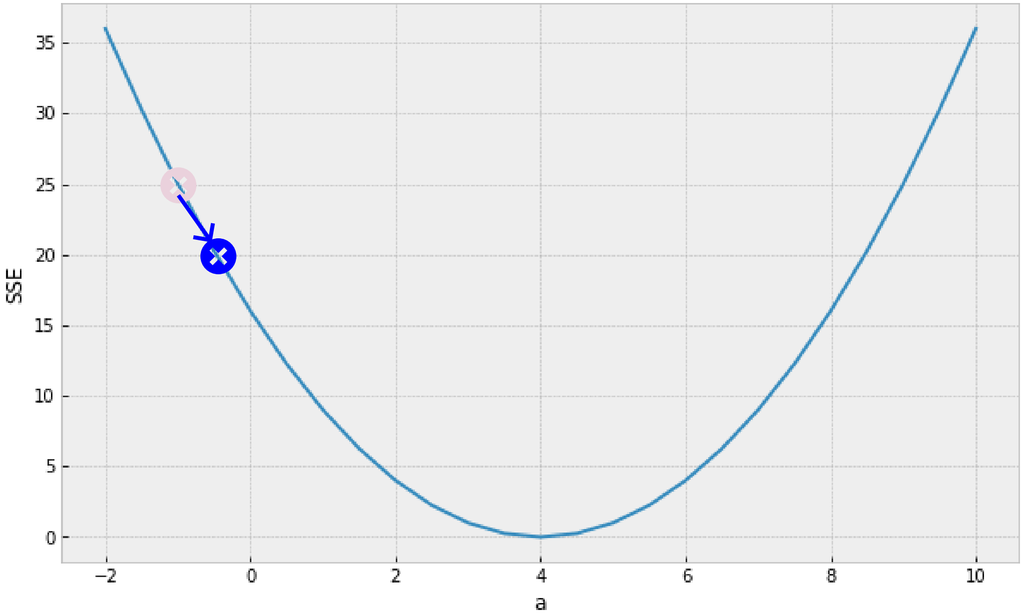


the algorithm: ~~Stochastic~~ **Gradient Descent**

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction

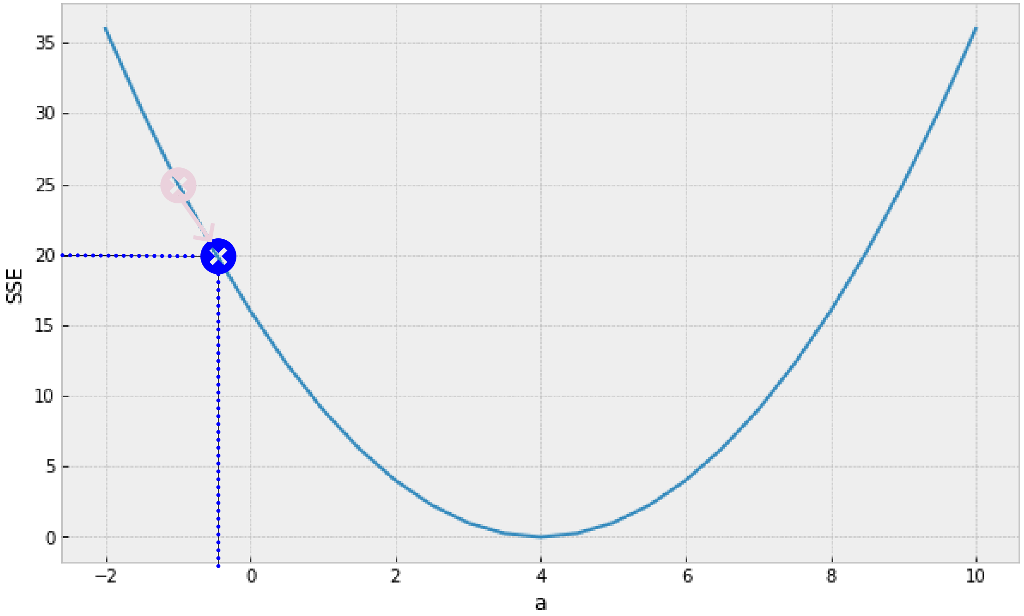


the algorithm: ~~Stochastic~~ **Gradient Descent**

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

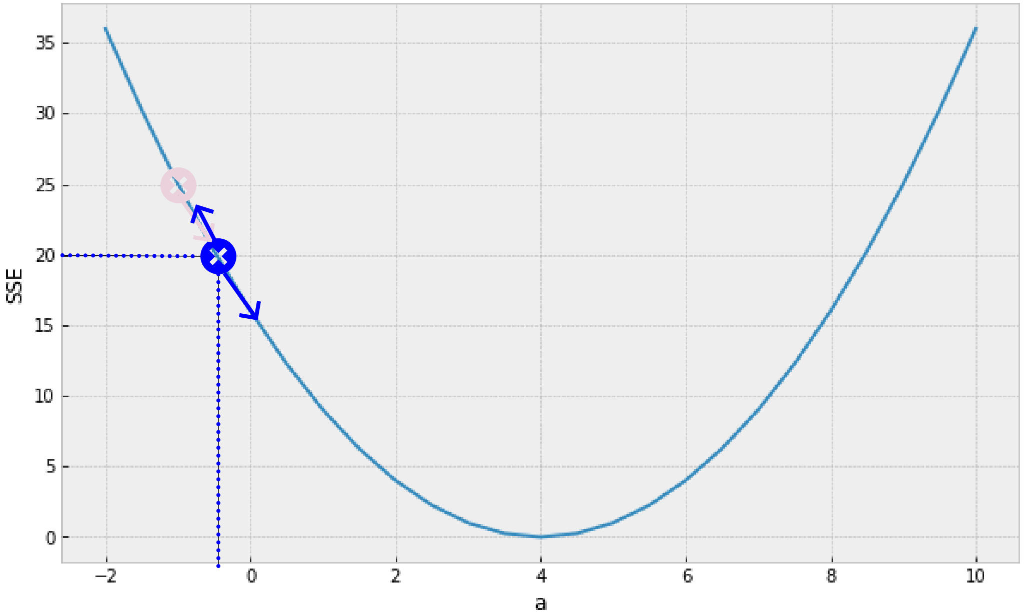


the algorithm: ~~Stochastic~~ Gradient Descent

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

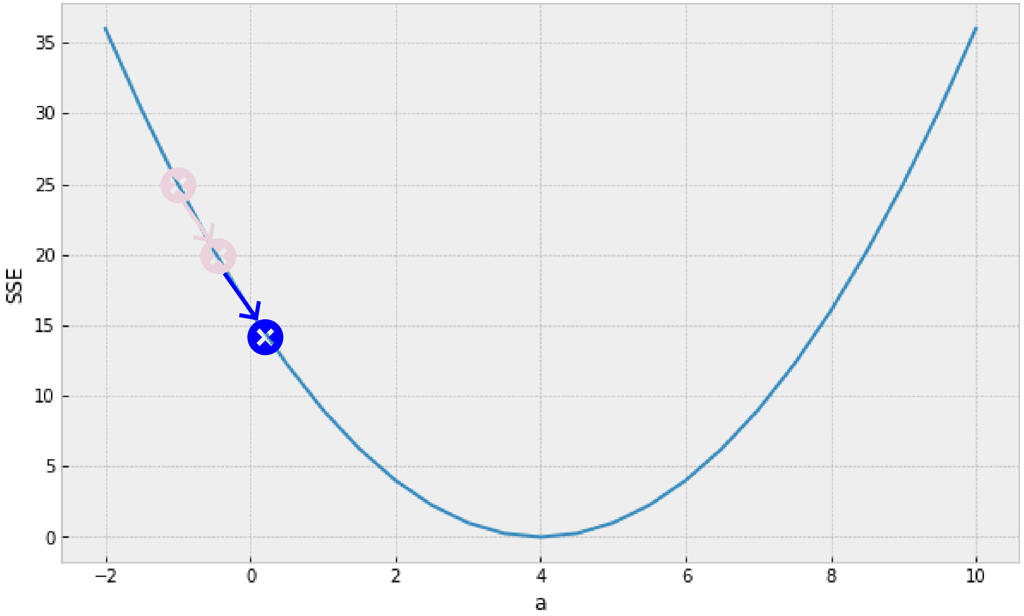


the algorithm: ~~Stochastic~~ Gradient Descent

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

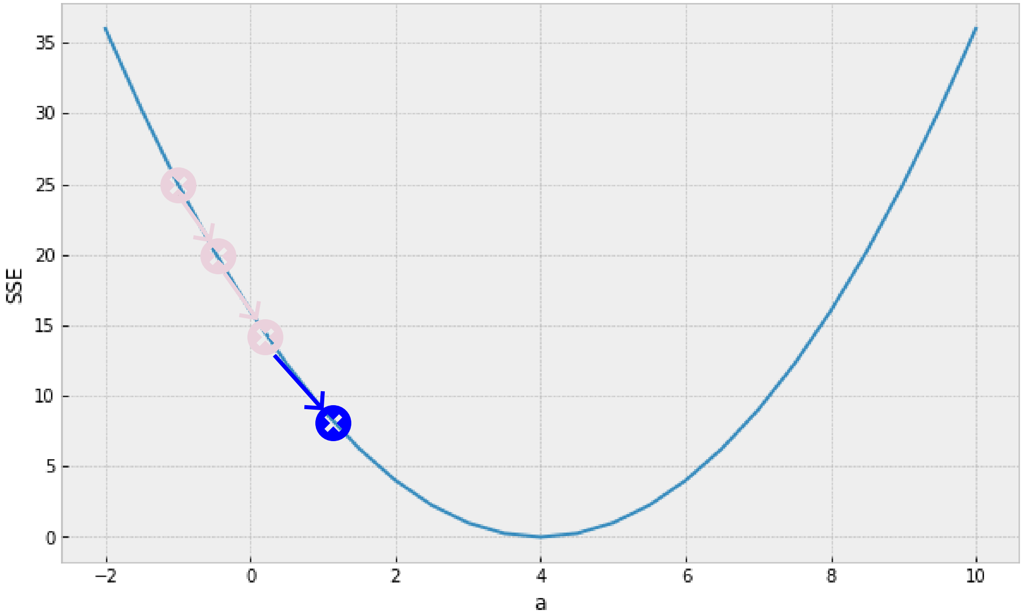


the algorithm: ~~Stochastic~~ Gradient Descent

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

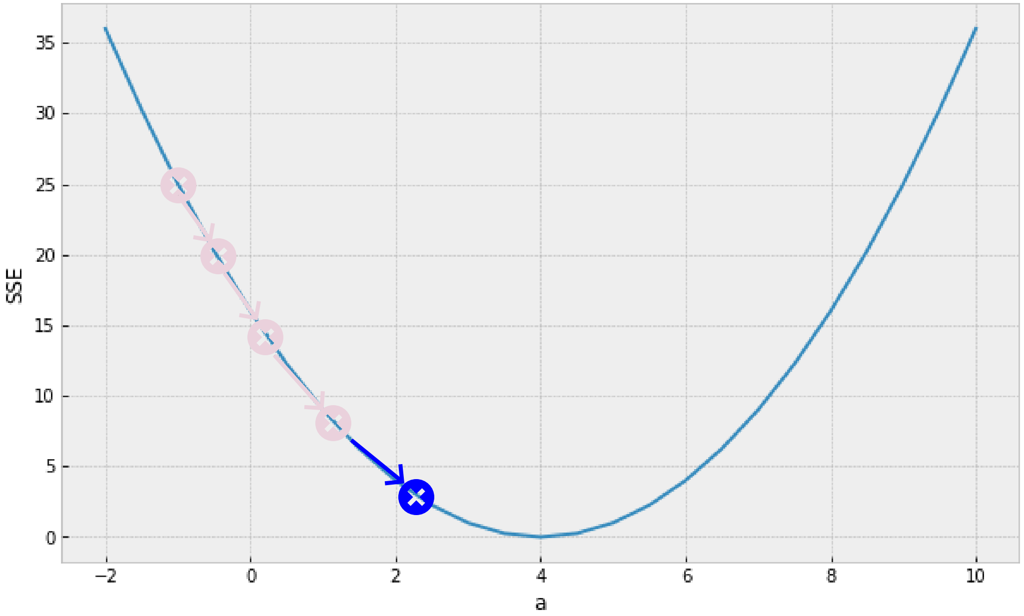


the algorithm: ~~Stochastic~~ Gradient Descent

assume a simpler line model $y = ax$

(b = 0) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

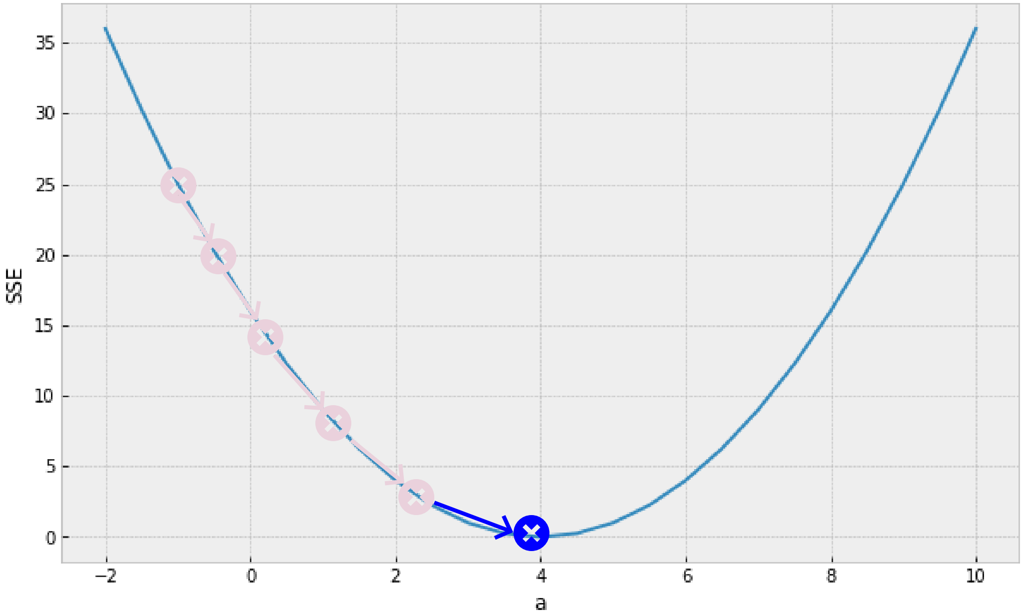


the algorithm: ~~Stochastic~~ Gradient Descent

assume a simpler line model $y = ax$

($b = 0$) so we only need to find the "best" parameter a

1. choose initial value for a
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

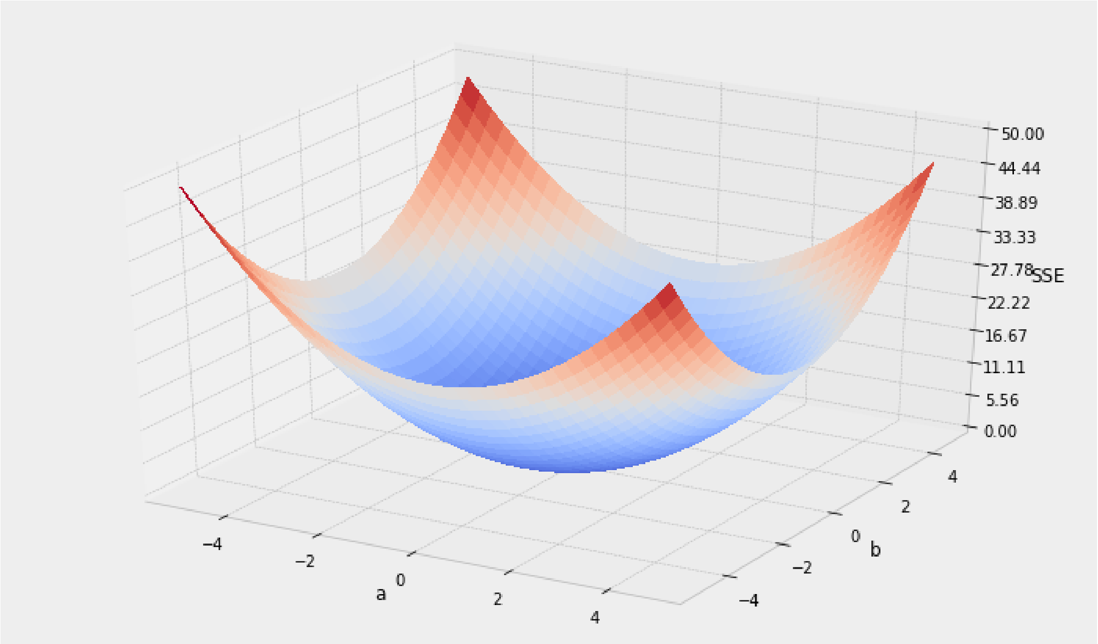


the algorithm: ~~Stochastic~~ **Gradient Descent**

for a line model $y = ax + b$

we need to find the "best" parameters a and b

1. choose initial value for a & b
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat

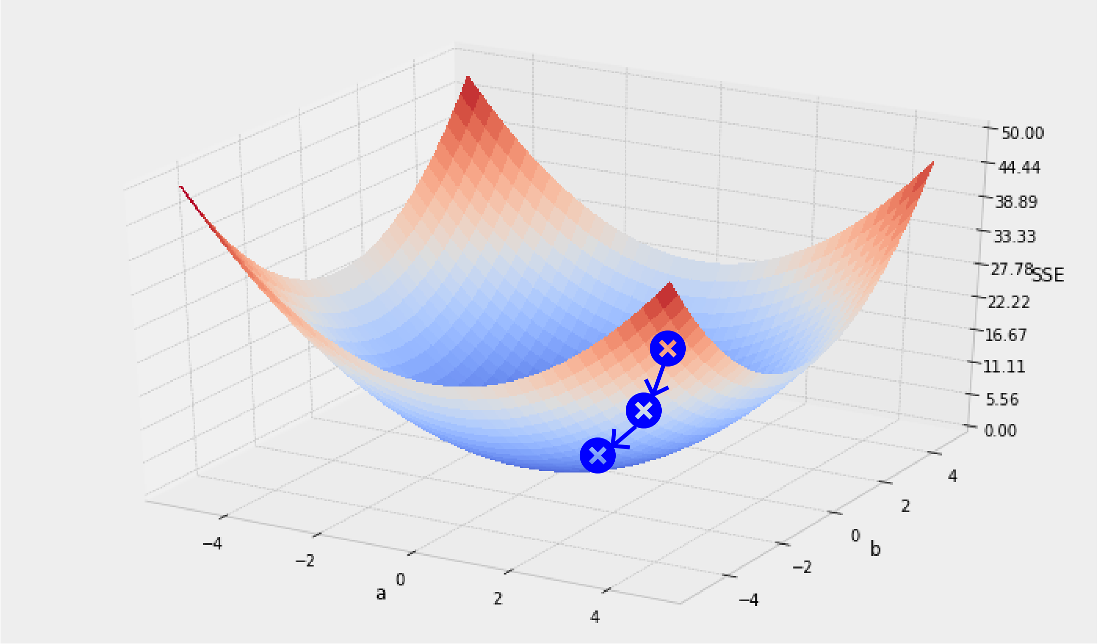


the algorithm: ~~Stochastic~~ Gradient Descent

for a line model $y = ax + b$

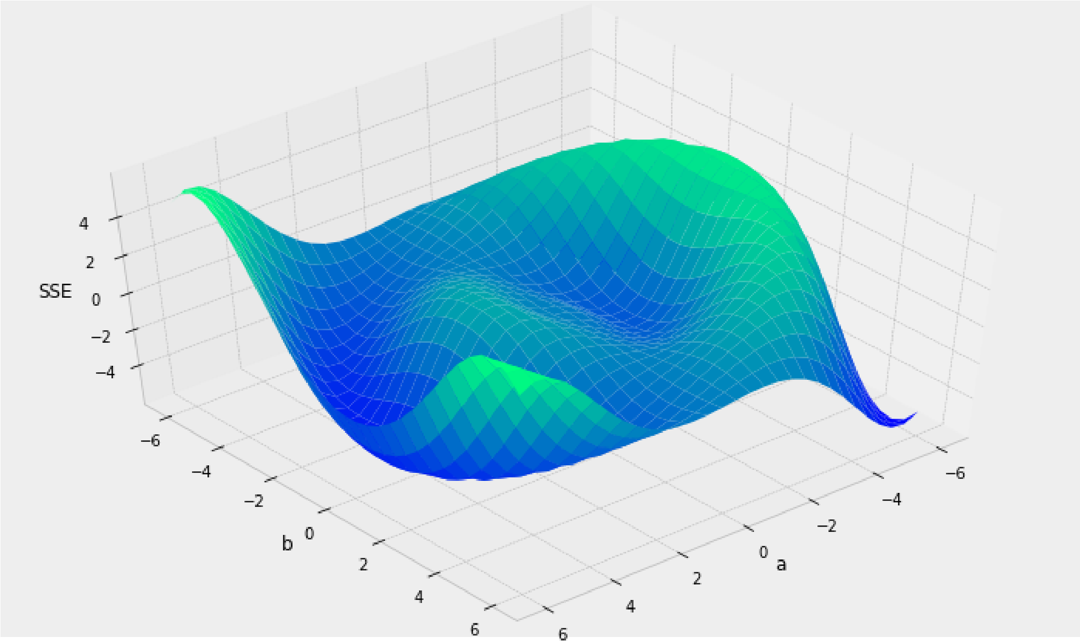
we need to find the "best" parameters a and b

1. choose initial value for a & b
2. calculate the SSE
3. calculate best direction to go to decrease the SSE
4. step in that direction
5. go back to step 2 and repeat



the algorithm: ~~Stochastic~~ Gradient Descent

Things to consider:

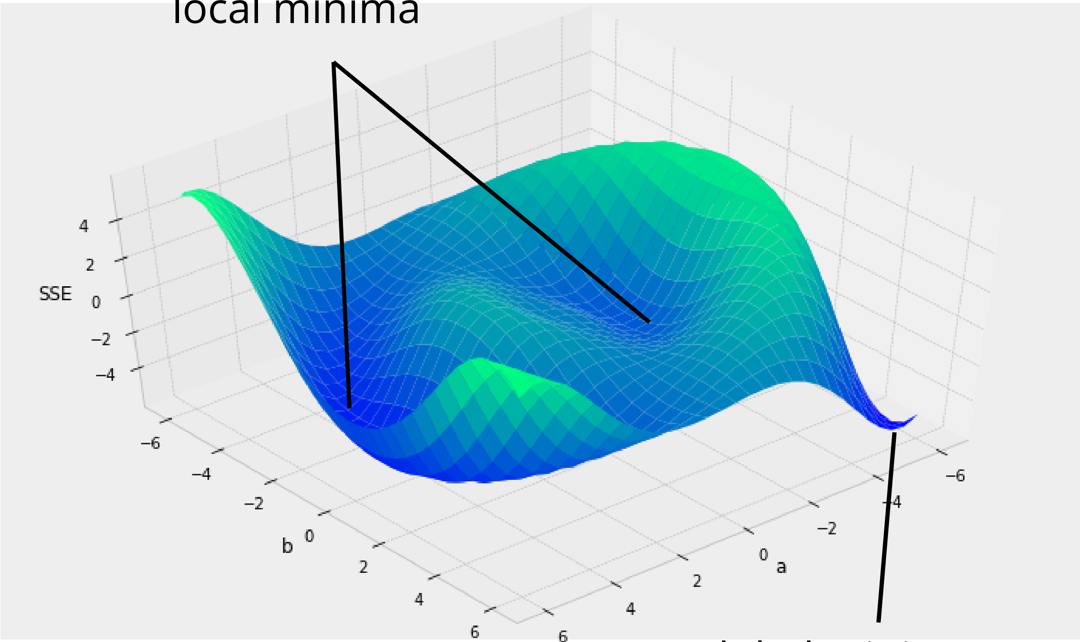


the algorithm: ~~Stochastic~~ Gradient Descent

Things to consider:

- local vs. global minima

local minima

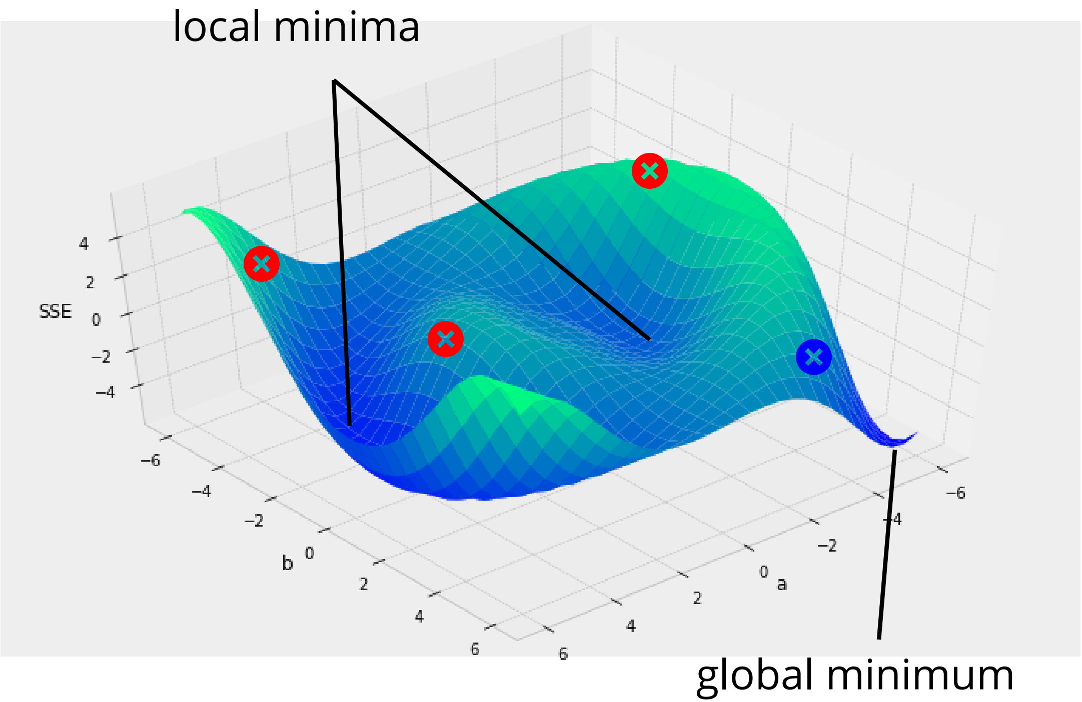


global minimum

the algorithm: ~~Stochastic~~ Gradient Descent

Things to consider:

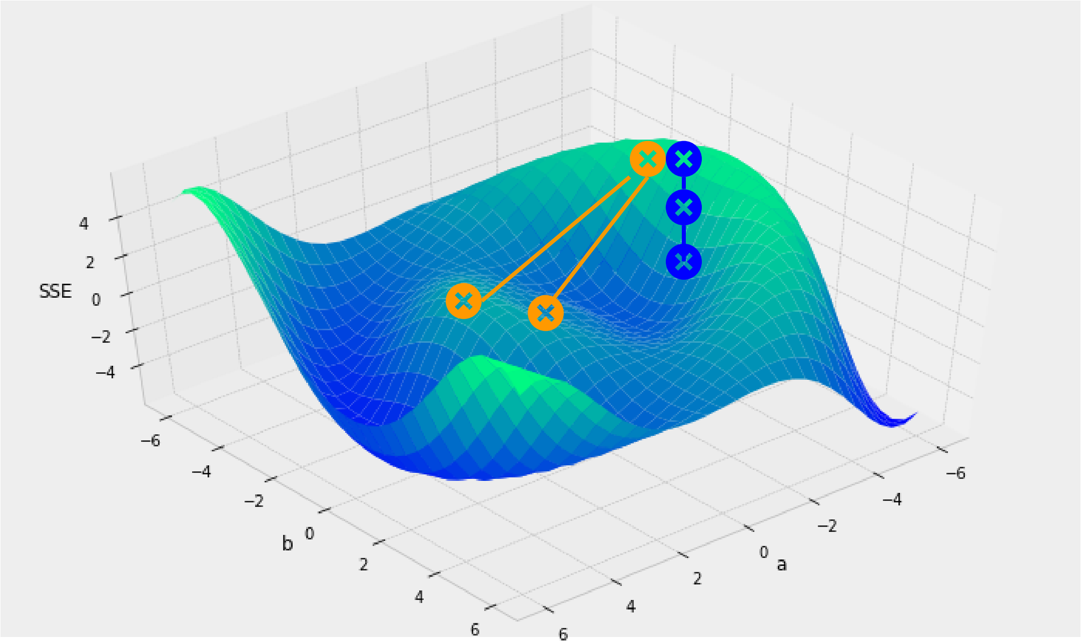
- local vs. global minima
- initialization: choosing starting spot?



the algorithm: ~~Stochastic~~ **Gradient Descent**

Things to consider:

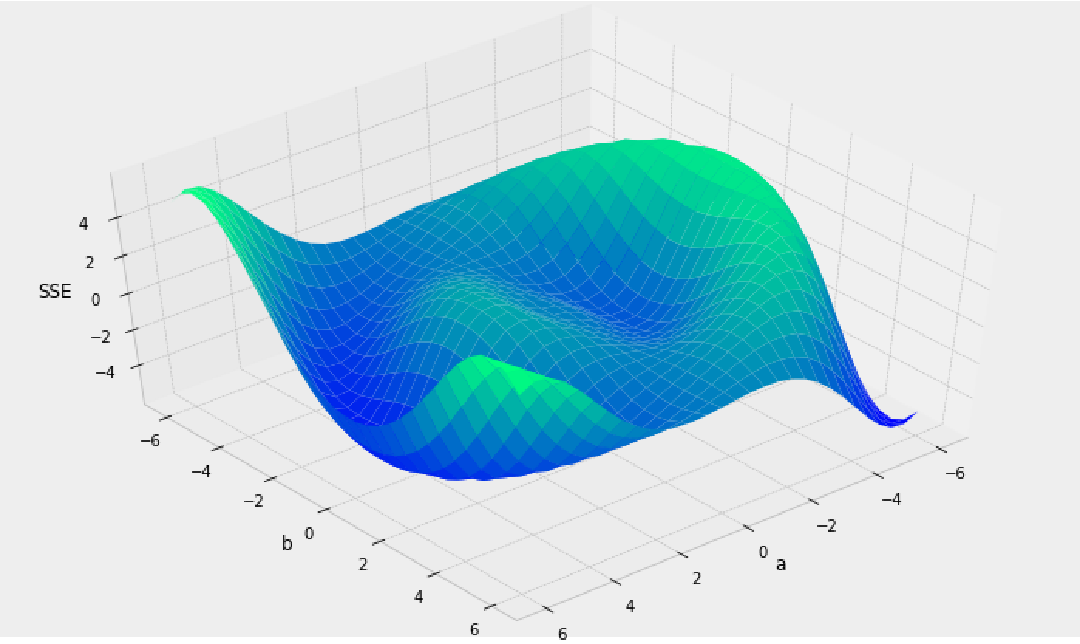
- local vs. global minima
- initialization: choosing starting spot?
- learning rate: how far to step?



the algorithm: ~~Stochastic~~ Gradient Descent

Things to consider:

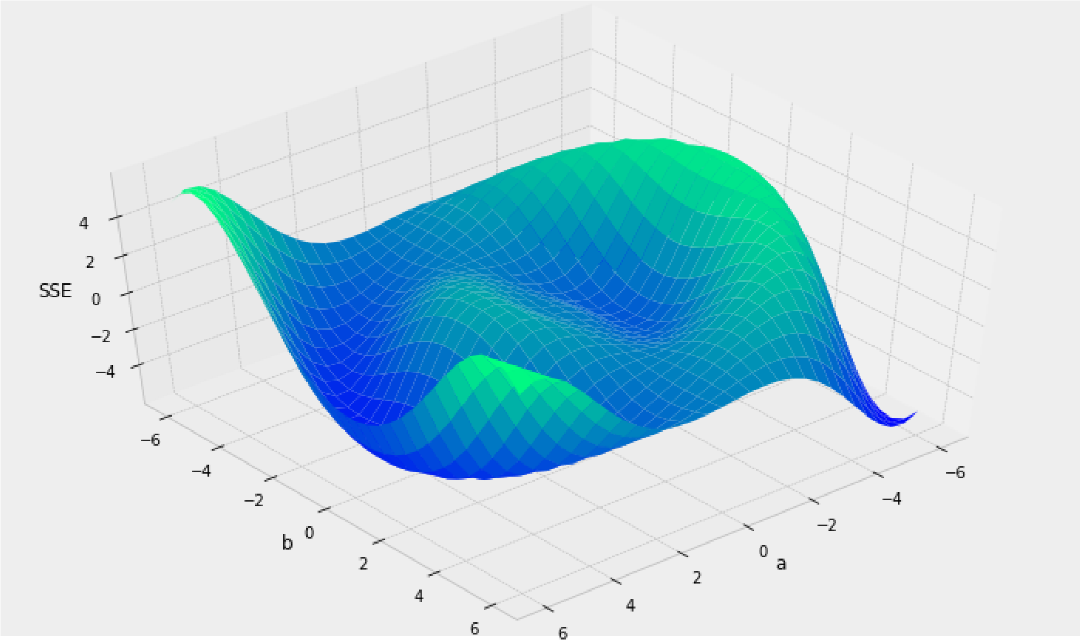
- local vs. global minima
- initialization: choosing starting spot?
- learning rate: how far to step?
- stopping criterion: when to stop?



the algorithm: **Stochastic Gradient Descent**

Things to consider:

- local vs. global minima
- initialization: choosing starting spot?
- learning rate: how far to step?
- stopping criterion: when to stop?

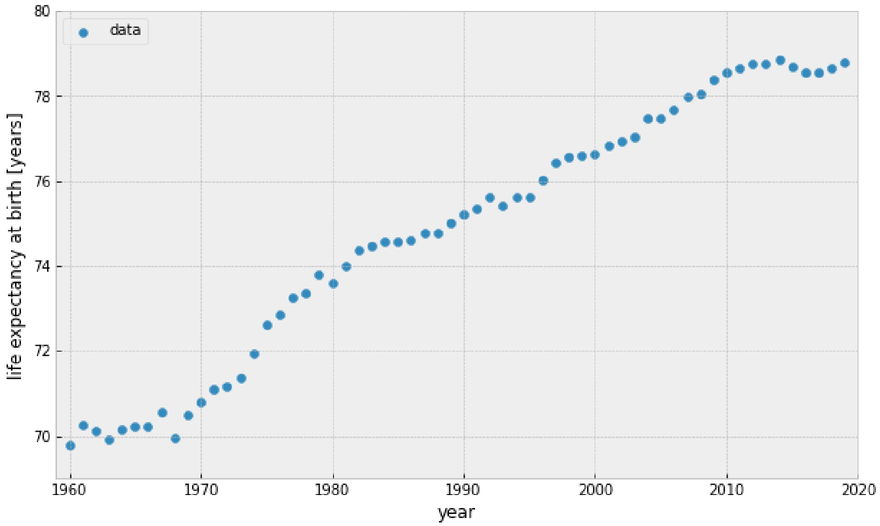


<u>Stochastic</u> Gradient Descent (SGD): use a different (random) sub-sample of the data at each iteration

2

multiple linear regression

ML terminology

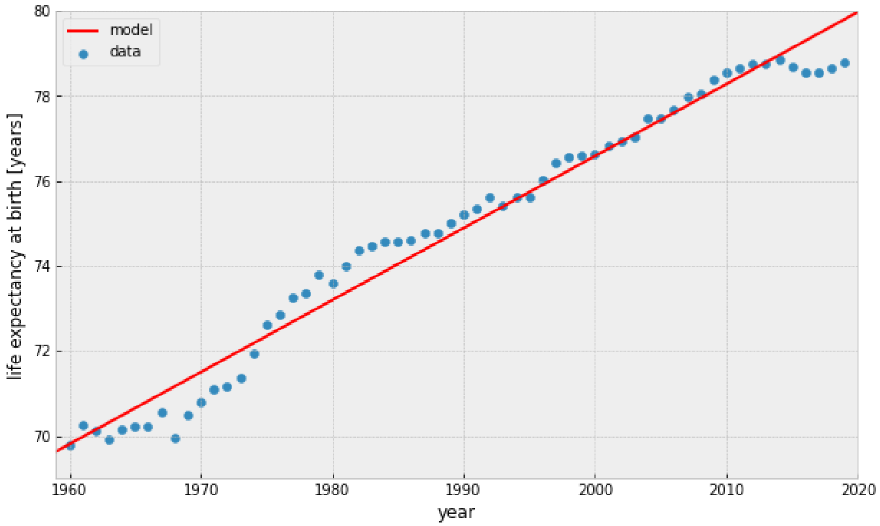


World Bank: Life expectancy at birth in the US

	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
	⋮	
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

ML terminology

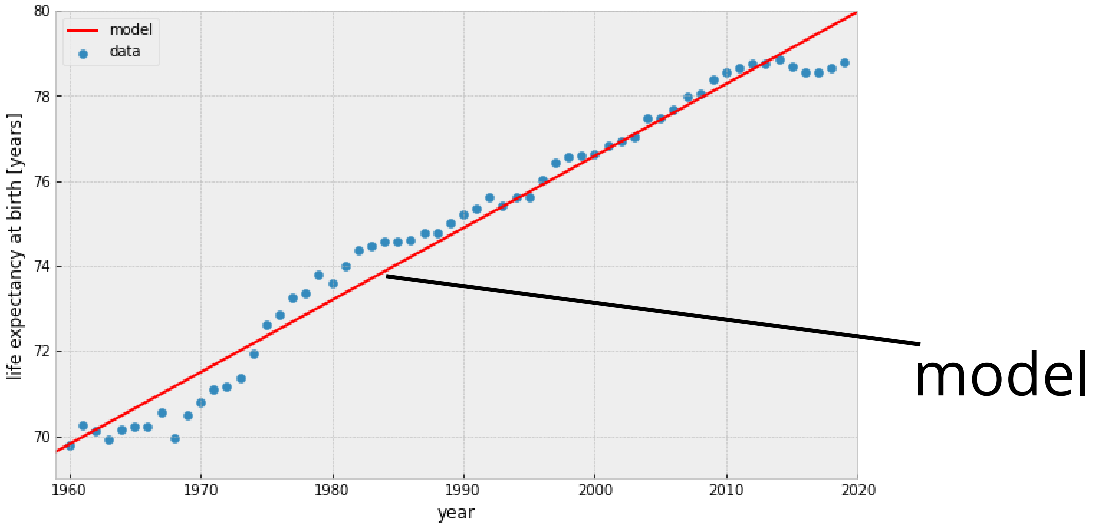
World Bank: Life expectancy at birth in the US



	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
⋮		
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

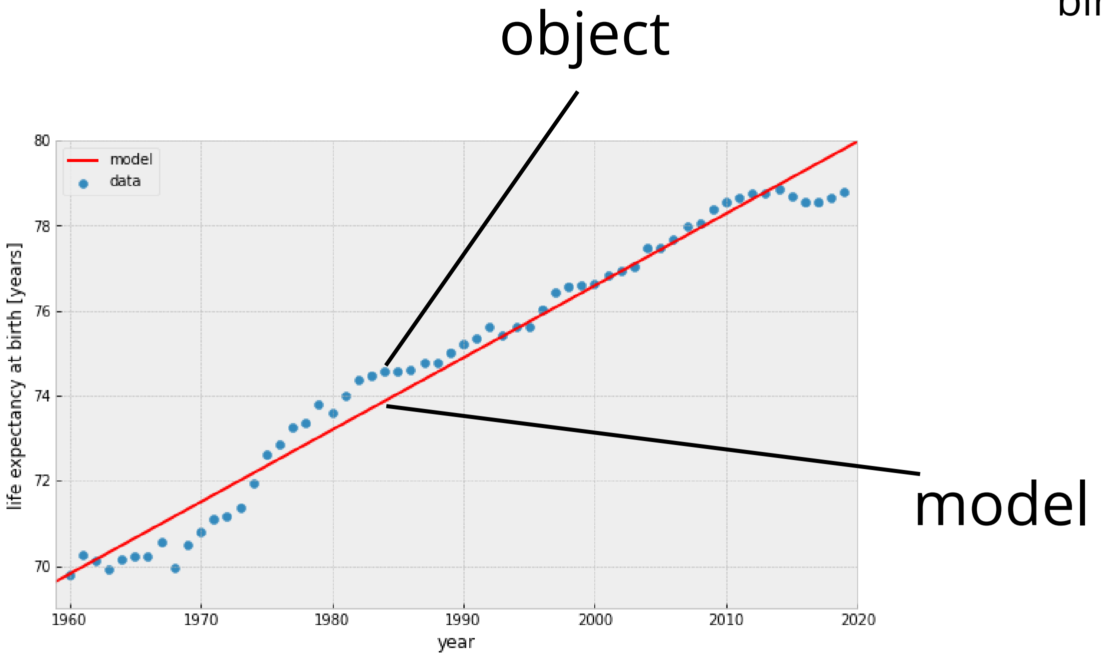
ML terminology

World Bank: Life expectancy at birth in the US



	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
⋮		
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

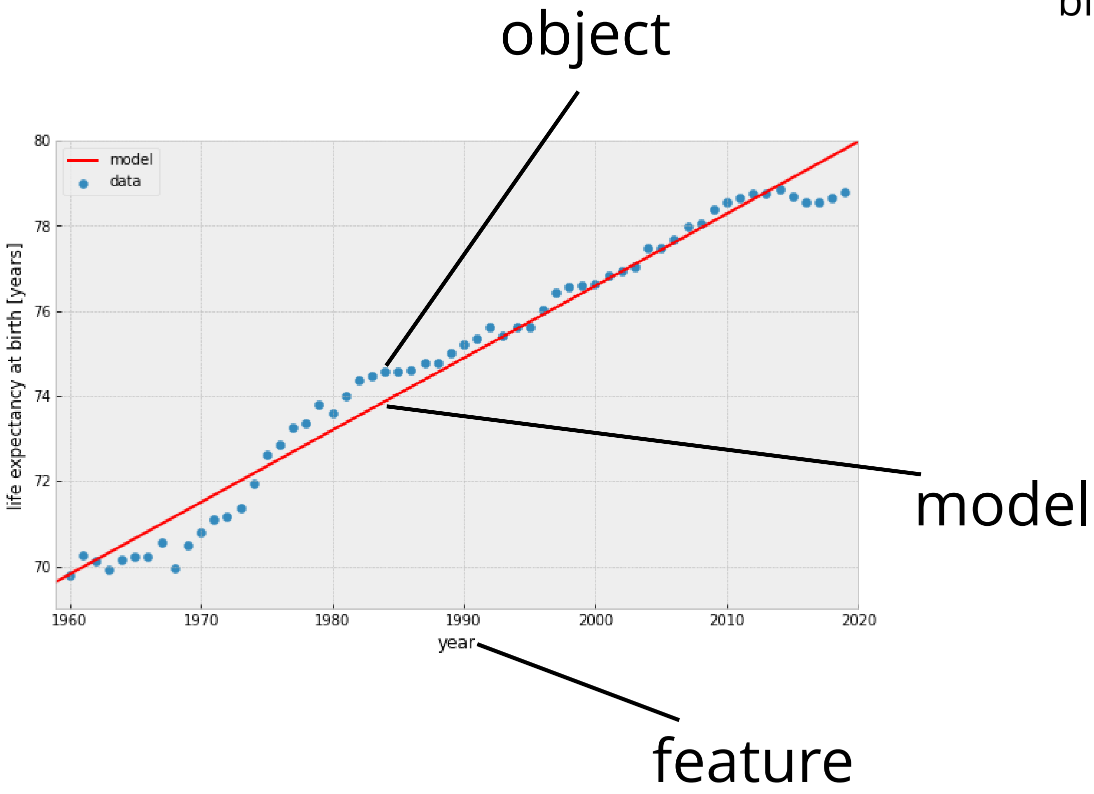
ML terminology



World Bank: Life expectancy at birth in the US

	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
⋮		
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

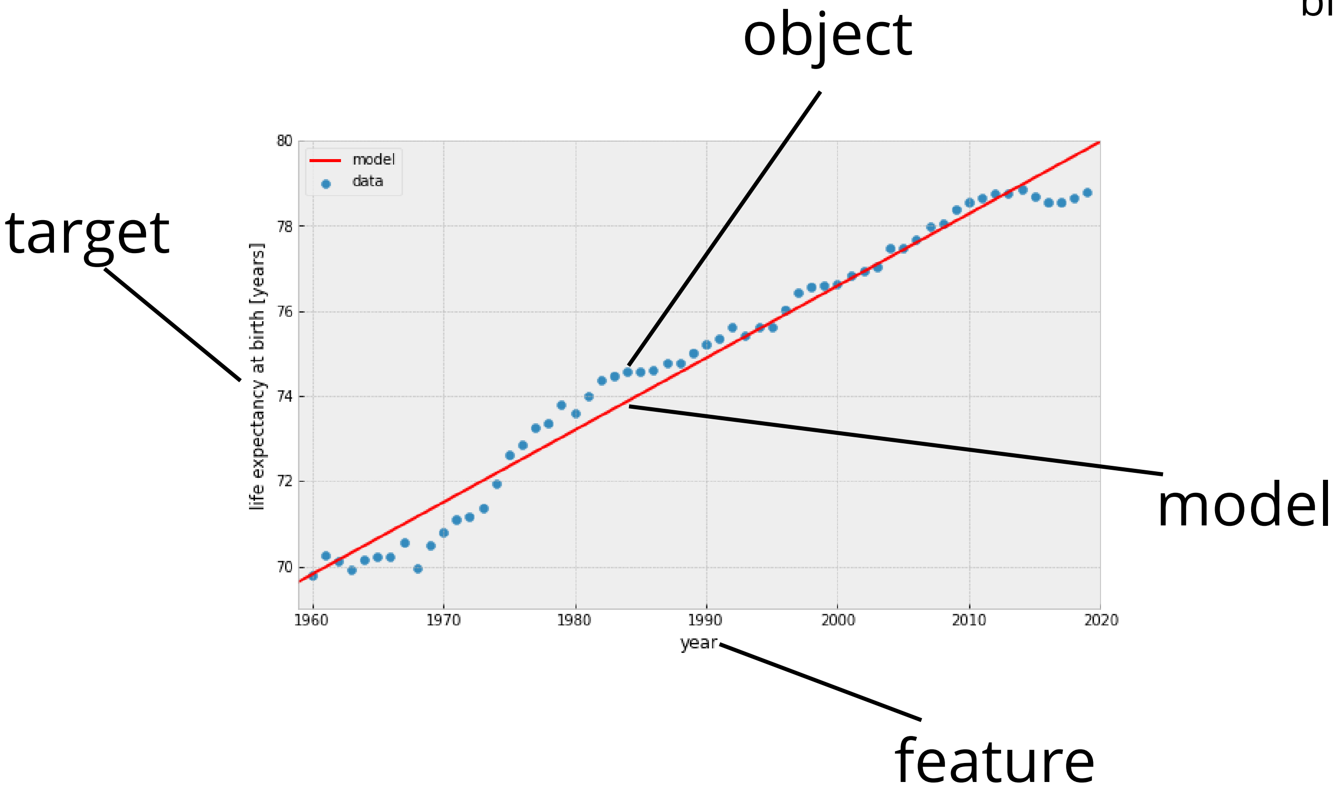
ML terminology



World Bank: Life expectancy at birth in the US

	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
	⋮	
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

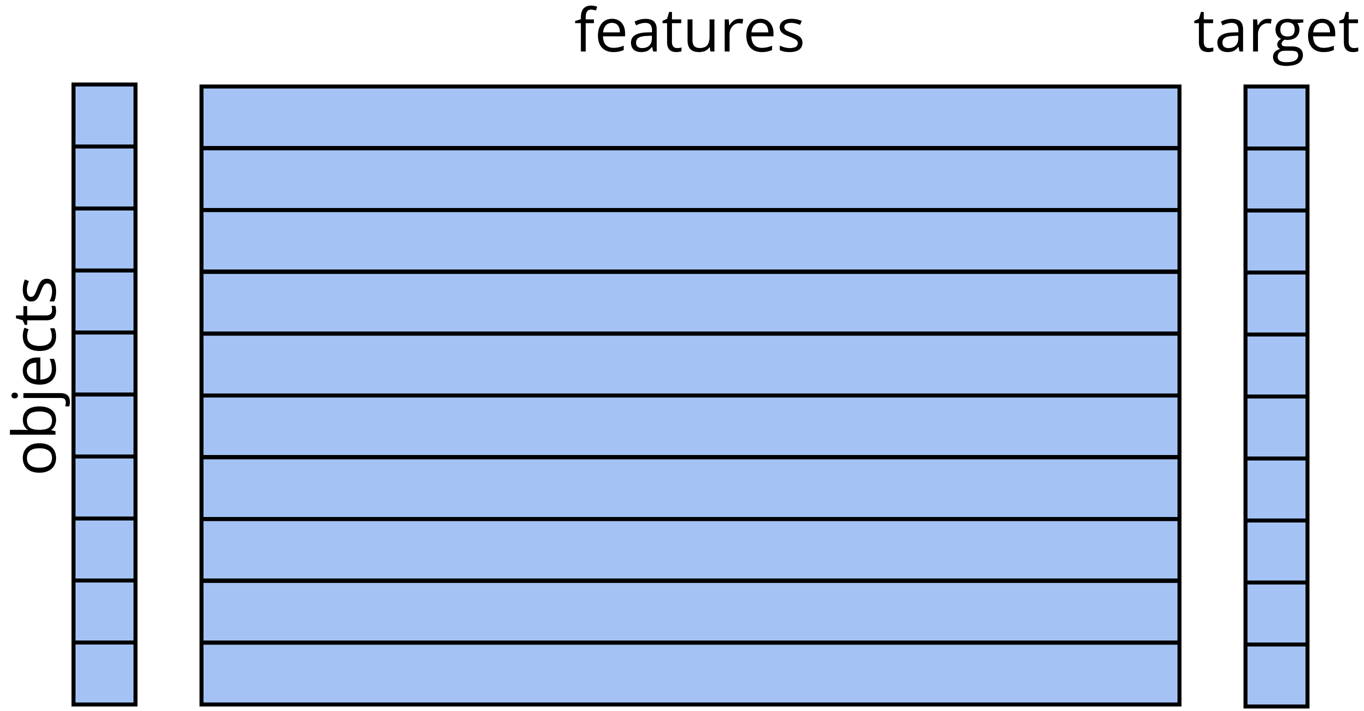
ML terminology



World Bank: Life expectancy at birth in the US

	year	leb
0	1960	69.770732
1	1961	70.270732
2	1962	70.119512
3	1963	69.917073
4	1964	70.165854
5	1965	70.214634
6	1966	70.212195
	⋮	
54	2014	78.841463
55	2015	78.690244
56	2016	78.539024
57	2017	78.539024
58	2018	78.639024
59	2019	78.787805

ML terminology



ML terminology

features target

objects

	transaction_date	house_age	distance_nearest_MRT_station	convenience_stores	house_price_unit_area
0	2012.917	32.0	84.87882	10	37.9
1	2012.917	19.5	306.59470	9	42.2
2	2013.583	13.3	561.98450	5	47.3
3	2013.500	13.3	561.98450	5	54.8
4	2012.833	5.0	390.56840	5	43.1
...	...	...	...	...	...
409	2013.000	13.7	4082.01500	0	15.4
410	2012.667	5.6	90.45606	9	50.0
411	2013.250	18.8	390.96960	7	40.6
412	2013.000	8.1	104.81010	5	52.5
413	2013.500	6.5	90.45606	9	63.9

Simple Linear Regression

1 feature

1 target

2 parameters

$$y = ax + b$$

Simple Linear Regression

1 feature

1 target

2 parameters

$$y = ax + b$$

$$y = \beta_0 + \beta_1 x_1$$

Simple Linear Regression

1 feature

1 target

2 parameters

$$y = ax + b$$

$$y = \beta_0 + \beta_1 x_1$$

Multiple Linear Regression

n features

1 target

Simple Linear Regression

1 feature
1 target
2 parameters

$$y = ax + b$$

$$y = \beta_0 + \beta_1 x_1$$

Multiple Linear Regression

n features
1 target
n+1 parameters

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_n x_n$$

Simple Linear Regression

1 feature
1 target
2 parameters

$$y = ax + b$$

$$y = \beta_0 + \beta_1 x_1$$

Multiple Linear Regression

n features
1 target
n+1 parameters

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_n x_n$$

$$y = \sum_{i=0}^{n} \beta_i x_i \;\; ; \;\; x_0 = \vec{1}$$