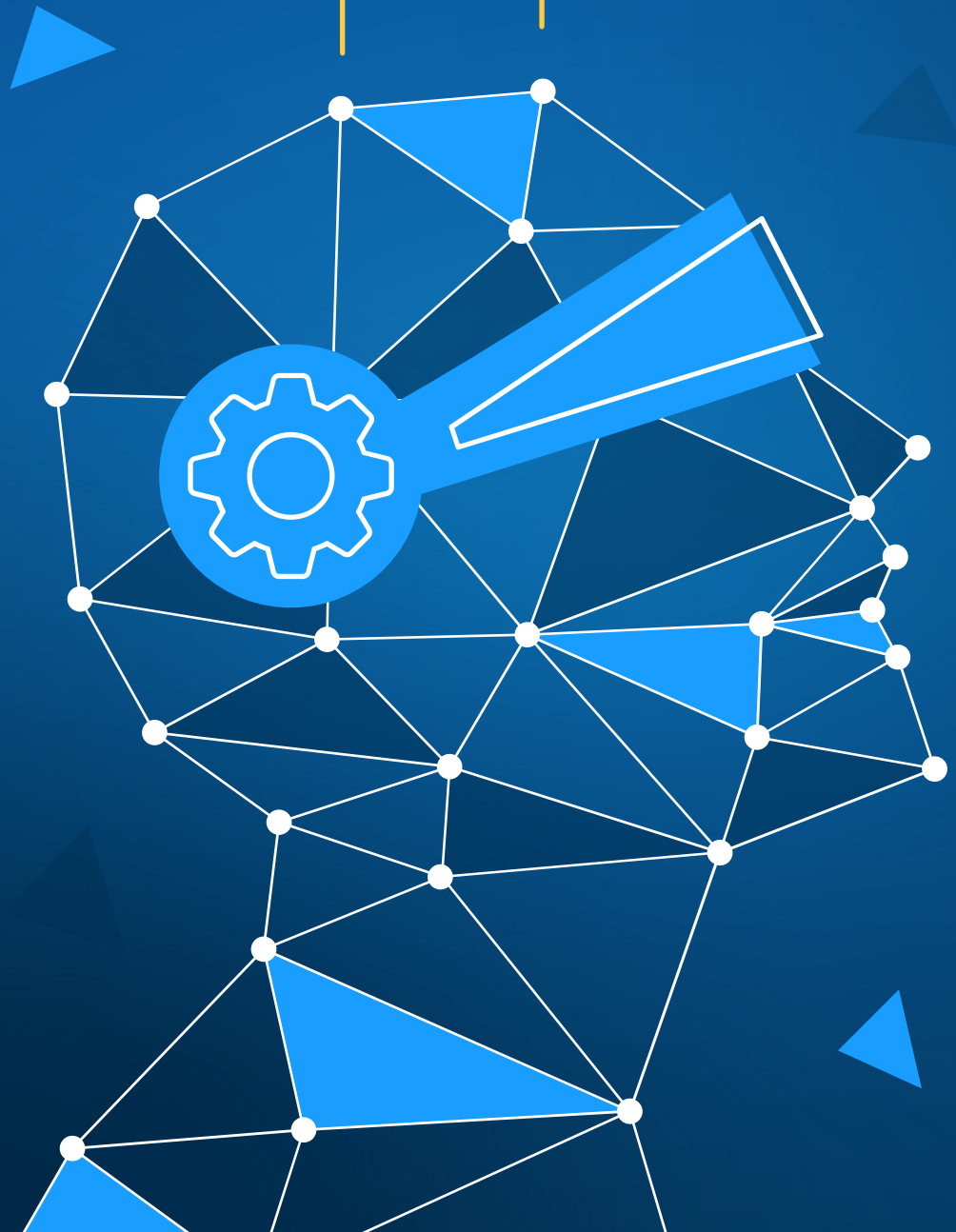
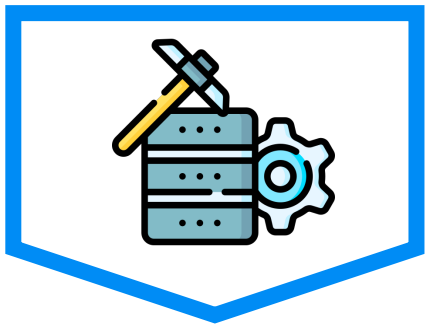


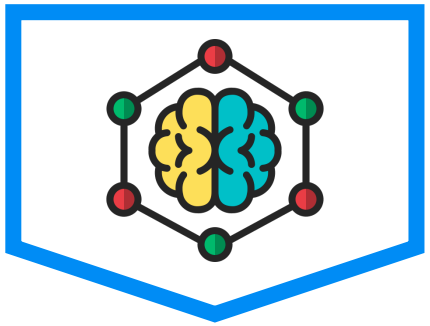
Data Preprocessing

The Machine Learning Process

The Machine Learning Process



Data Pre-Processing
- Import the data
- Clean the data
- Split into training & test sets
- Feature Scaling



Modelling
- Build the model
- Train the model
- Make predictions

Evaluation
- Calculate performance metrics
- Make a verdict

Training Set & Test Set

Training Set & Test Set

~

Train
80%

Test
20%

$$\hat{y} = b_0 + b_1 X_1 + b_2 X_2$$

Predicted values $\hat{y}$

V.S.

Actual values y

Feature Scaling

Feature Scaling

X1	X2	X3	X4
$ 179.43	56.784	34.6181	3.55
$ 641.87	62.054	47.7306	1.692
$ 556.30	64.13	55.596	1.559
$ 578.47	63.377	52.7121	1.679
$ 591.16	61.553	46.1315	1.984
$ 242.03	58.29	39.2952	2.942
$ 364.66	59.93	42.4628	2.494
$ 190.68	57.271	36.2725	3.419
$ 547.23	63.763	54.1971	1.634
$ 359.69	59.375	41.5105	2.128
$ 438.08	60.484	43.493	2.47
$ 637.17	62.525	49.428	1.725

Feature Scaling

Normalization

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

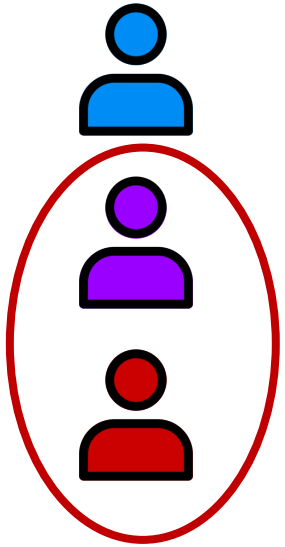
[0 ; 1]

Standardization

$$X' = \frac{X - \mu}{\sigma}$$

[-3 ; +3]

Feature Scaling



70,000 $

10,000

60,000 $

8,000

52,000 $

45 yrs

1

44 yrs

4

40 yrs

Feature Scaling

Normalization

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

$$[0 ; 1]$$

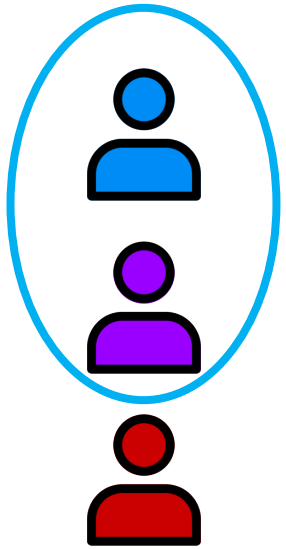
Feature Scaling

	70,000 $	45 yrs
	60,000 $	44 yrs
	52,000 $	40 yrs

Feature Scaling

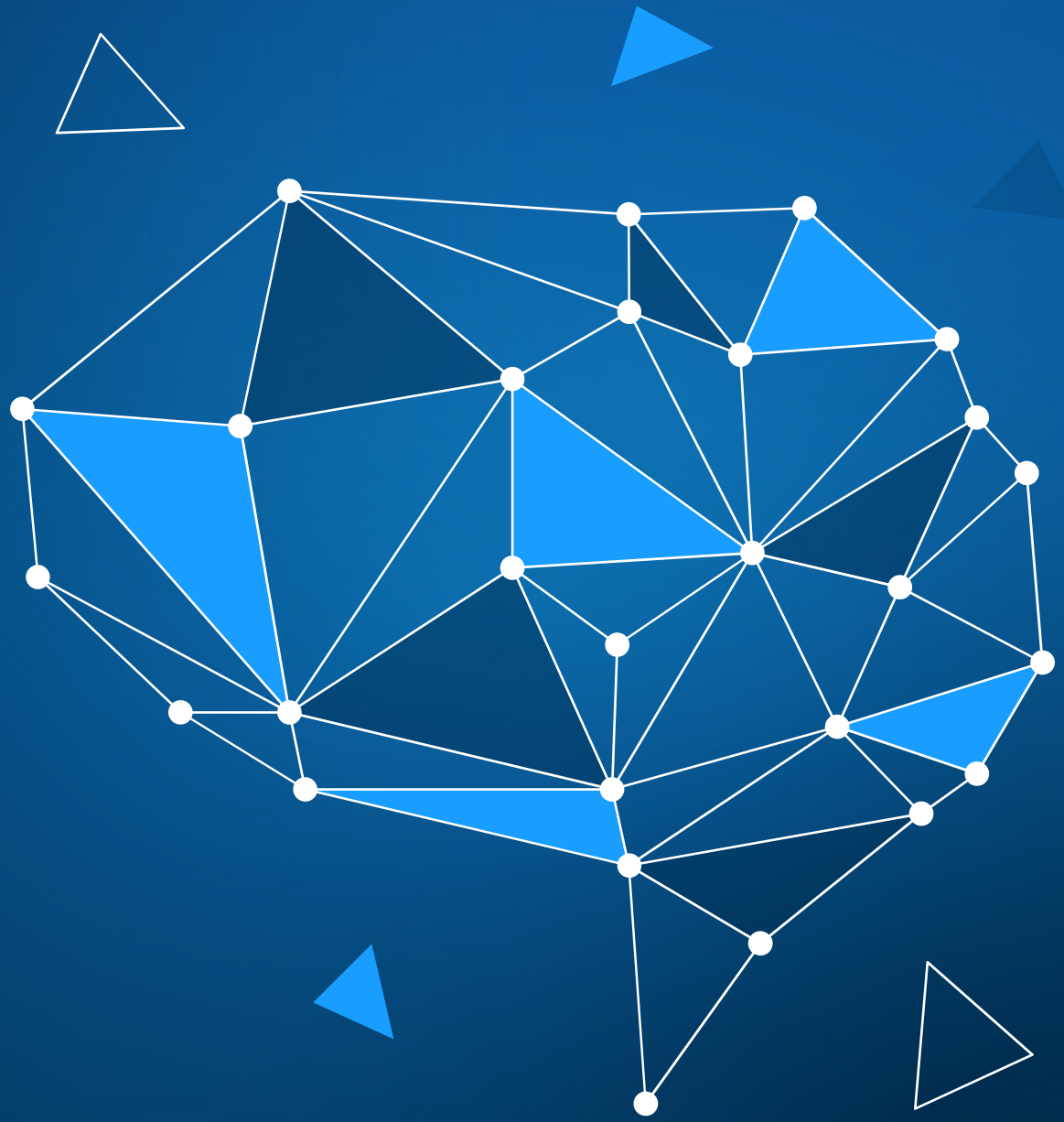
	1	45 yrs
	0.444	44 yrs
	0	40 yrs

Feature Scaling



1	1
0.444	0.75
0	0

Regression



Simple Linear Regression

Simple Linear Regression

$$\hat{y} = b_0 + b_1 X_1$$

Dependent variable

y-intercept (constant)

Slope coefficient

Independent variable

Simple Linear Regression

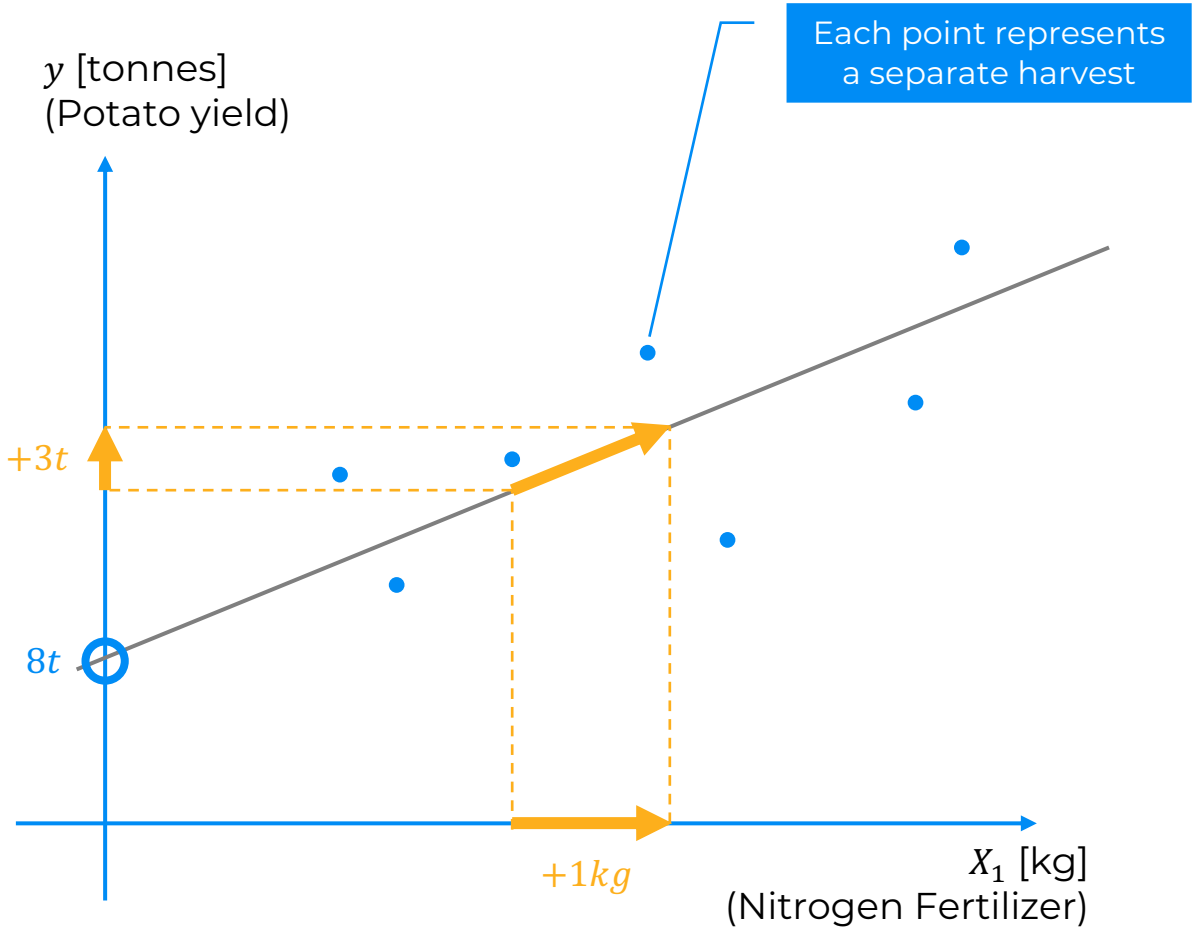
$\sim$

$$\hat{y} = b_0 + b_1 X_1$$

$$Potatoes[t] = b_0 + b_1 \times Fertilizer[kg]$$

$$b_0 = 8[t]$$

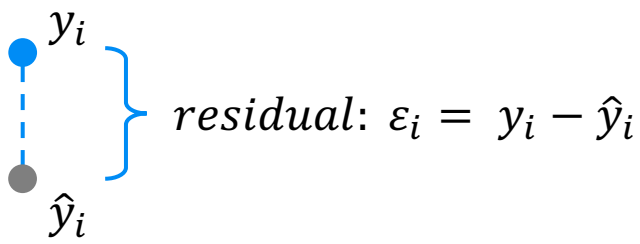
$$b_1 = 3[\frac{t}{kg}]$$



Ordinary Least Squares

Simple Linear Regression

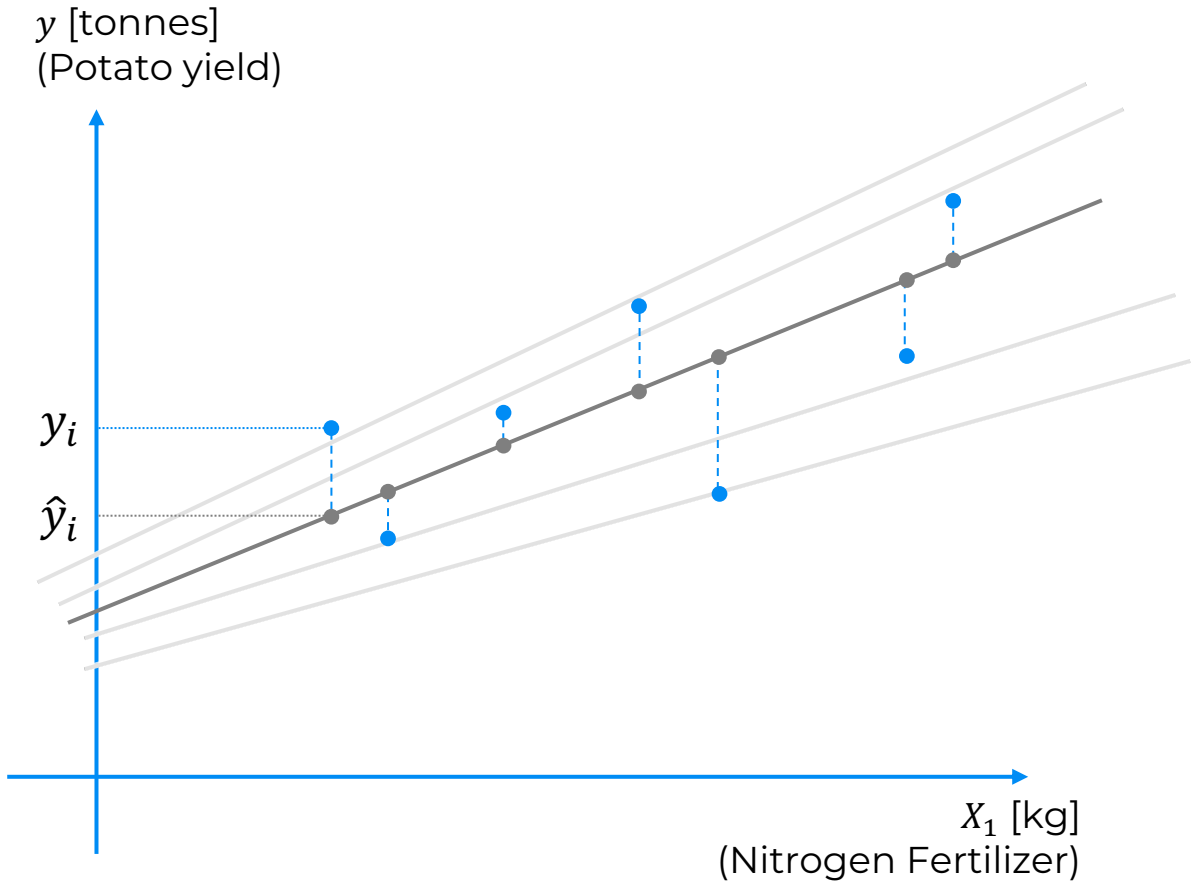
Ordinary Least Squares:



$\hat{y} = b_0 + b_1 X_1$

b_0, b_1 such that:

$SUM(y_i - \hat{y}_i)^2$ is minimized



Multiple Linear Regression

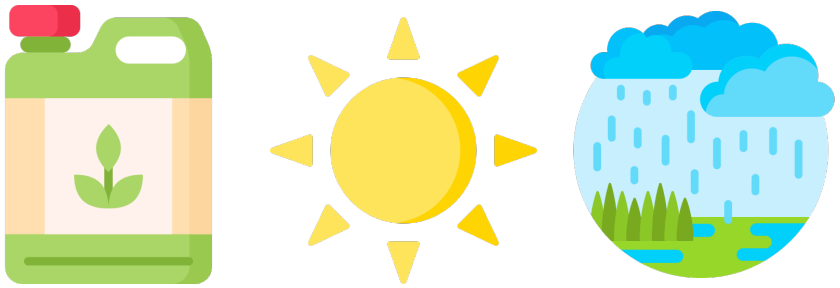
Multiple Linear Regression

$$\hat{y} = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_n X_n$$

- $\hat{y}$: Dependent variable
- b_0: y-intercept (constant)
- b_1: Slope coefficient 1
- X_1: Independent variable 1
- b_2: Slope coefficient 2
- X_2: Independent variable 2
- b_n: Slope coefficient n
- X_n: Independent variable n

Multiple Linear Regression

$\sim$



$$Potatoes[t] = 8t + 3\frac{t}{kg} \times Fertilizer[kg] - 0.54\frac{t}{°C} \times AvgTemp[°C] + 0.04\frac{t}{mm} \times Rain[mm]$$

Additional Reading

The Application of Multiple Linear Regression and Artificial Neural Network Models for Yield Prediction of Very Early Potato Cultivars before Harvest

Magdalena Piekutowska et. al. (2021)

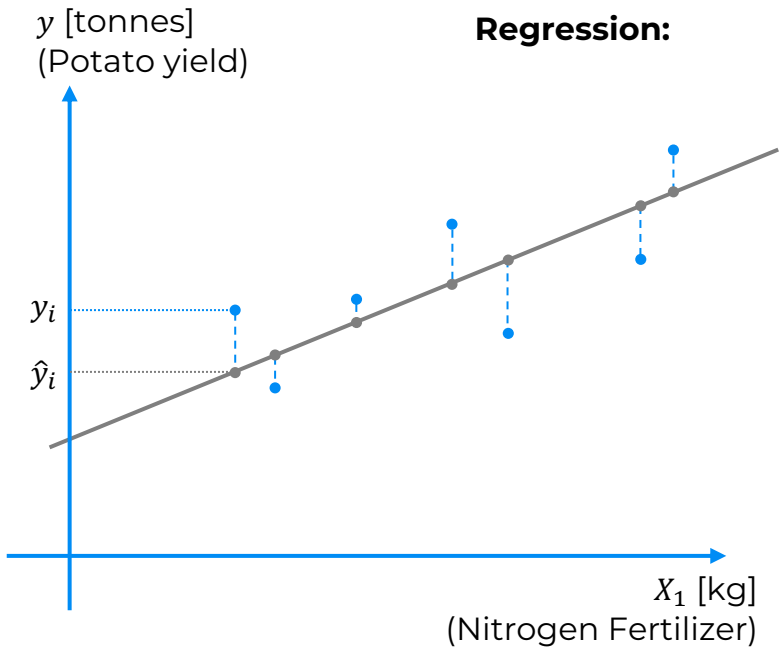
Link:

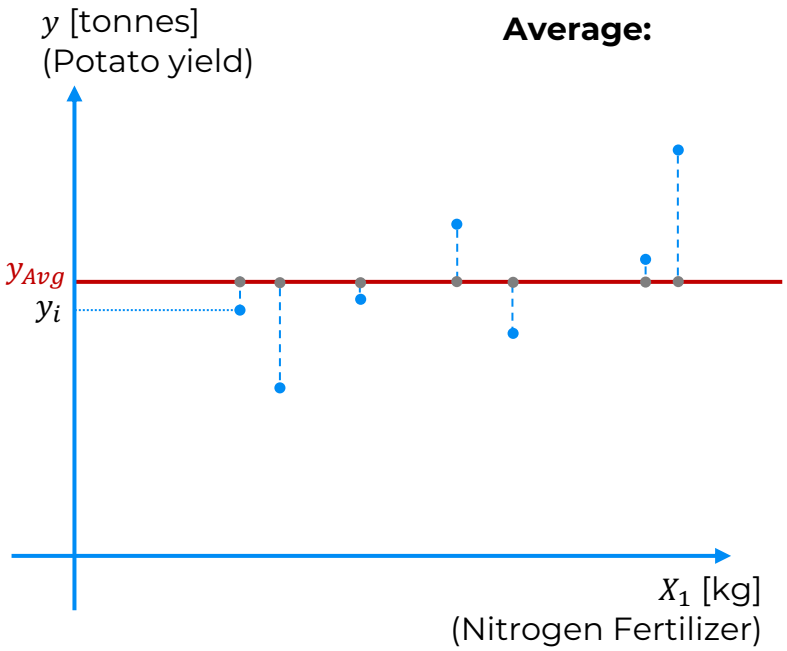
https://www.mdpi.com/2073-4395/11/5/885

Quantitative Yield Forecast		
Models RY1 and NY1	Yield Forecast before Harvest (40 Days from Full Emergence)	Data Range
INSO	insolation sum [h] in the periods: planting—June 20,	275.3–711.7
TEMP	average daily air temperature [°C] in the periods: planting—20 June	10.8–15.7
PREC	precipitation [mm] in the periods: planting—20 June	38.7–258.2
NITRO	sum of nitrogen fertilization [kg·ha^{-1}] in the periods: planting—20 June	80–155
PHOSP	sum of phosphorus fertilization [kg·ha^{-1}]	28.2–150
POTAS	sum of potassium fertilization [kg·ha^{-1}]	80–306.5
PLANT	planting date [number of days since the beginning of the year]	107–127
EMERG	date of emergence [number of days since the beginning of the year], yield forecast 20th of June	130–151
DENST	densification [plants/plot], yield forecast June 20	35–60
PH	Soil pH [in 1 mol KCl]	5.8–7
SFERTP	soil fertility in phosphorus [mg P_2O_5·100 g^{-1} soil]	14–26.2
SFERTK	soil fertility in potassium [mg K_2O·100 g^{-1} soil]	11.7–19.2
SFERTM	soil fertility in magnesium [mg Mg·100 g^{-1} soil]	3–9.1
YIELDP1	tuber yield [t·ha^{-1}], harvest 40 days from full emergence	11.6–41.3

R Squared

R Squared





$$SS_{res} = SUM(y_i - \hat{y}_i)^2$$

$$SS_{tot} = SUM(y_i - y_{avg})^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Rule of thumb (for our tutorials)*:
1.0 = Perfect fit (suspicious)
~0.9 = Very good
<0.7 = Not great
<0.4 = Terrible
<0 = Model makes no sense for this data

**This is highly dependent on the context*

Adjusted R Squared

Adjusted R Squared

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

R^2 – Goodness of fit
(greater is better)

Problem:

$\hat{y} = b_0 + b_1X_1 + b_2X_2$ ← $+ b_3X_3$

$$SS_{res} = SUM(y_i - \hat{y}_i)^2$$

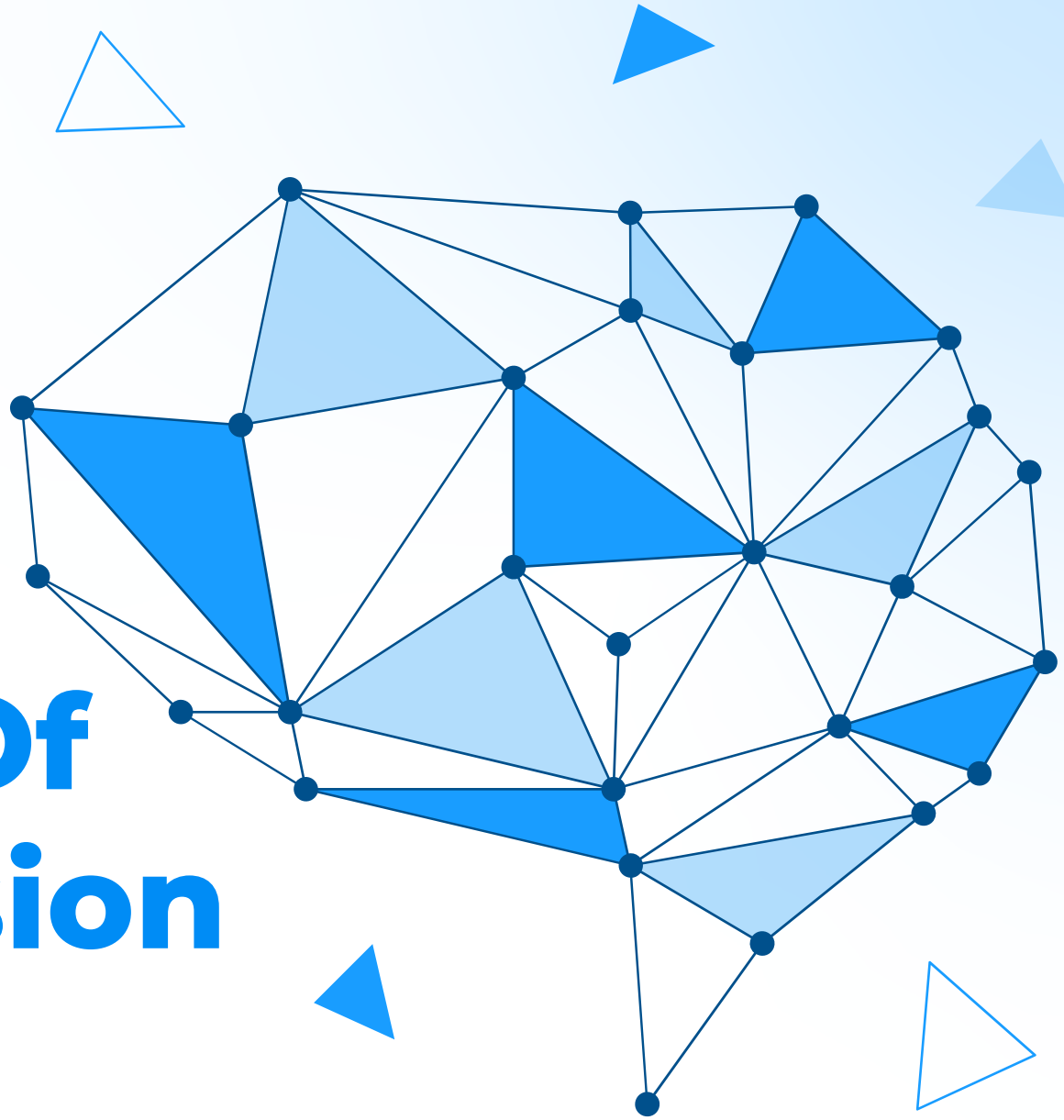
SS_{tot} doesn't change
SS_{res} will decrease or stay the same *(This is because of Ordinary Least Squares: SS_{res}-> Min)*

Solution:

$$Adj\ R^2 = 1 - (1 - R^2) \times \frac{n - 1}{n - k - 1}$$

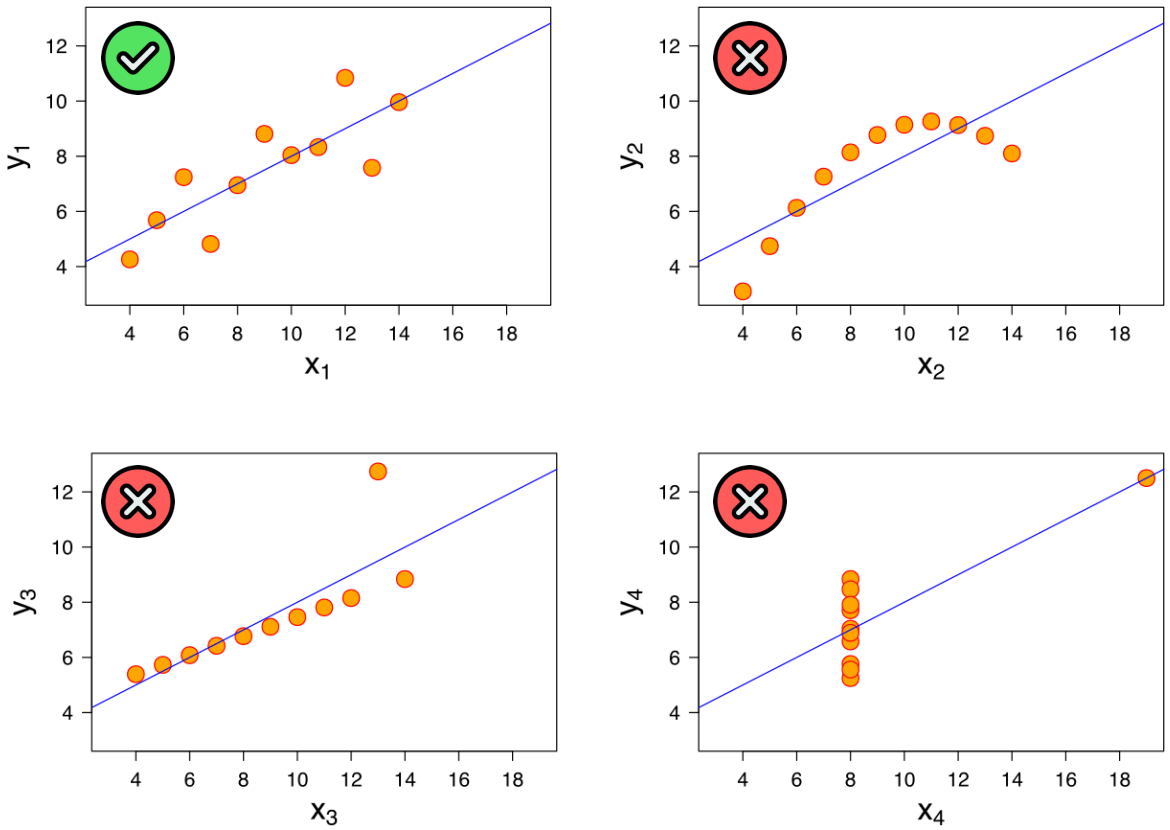
k – number of independent variables
n – sample size

Assumptions Of Linear Regression



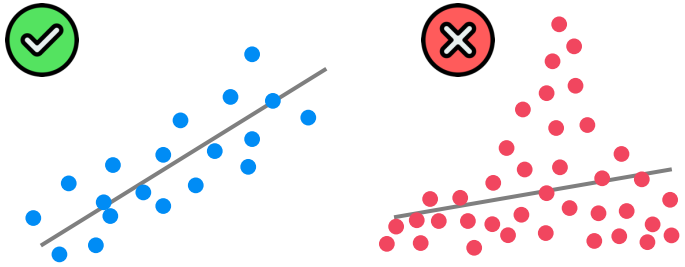
Assumptions of Linear Regression

Anscombe's quartet (1973):

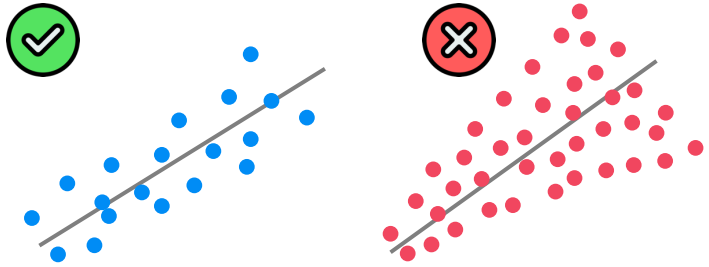


Assumptions of Linear Regression

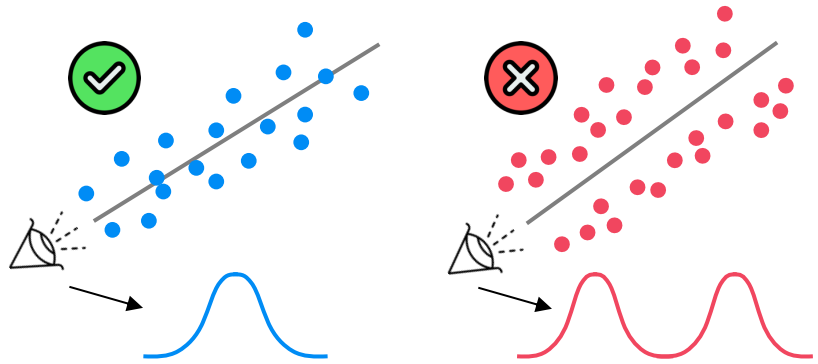
1. Linearity
(Linear relationship between Y and each X)



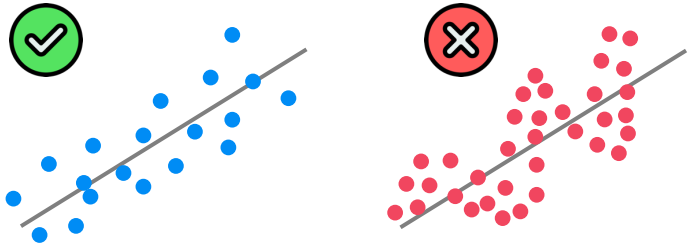
2. Homoscedasticity
(Equal variance)



3. Multivariate Normality
(Normality of error distribution)



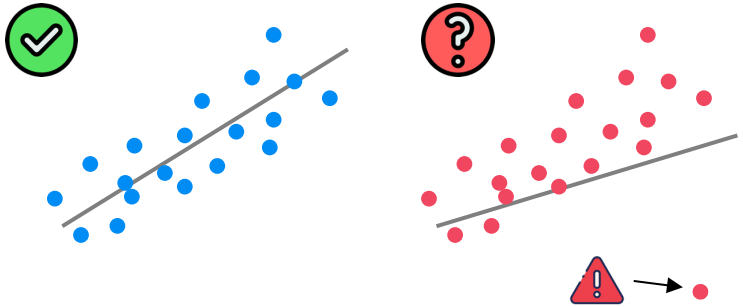
4. Independence
(of observations. Includes “no autocorrelation”)



5. Lack of Multicollinearity
(Predictors are not correlated with each other)

$X_1 \not\sim X_2$ $\quad X_1 \sim X_2$

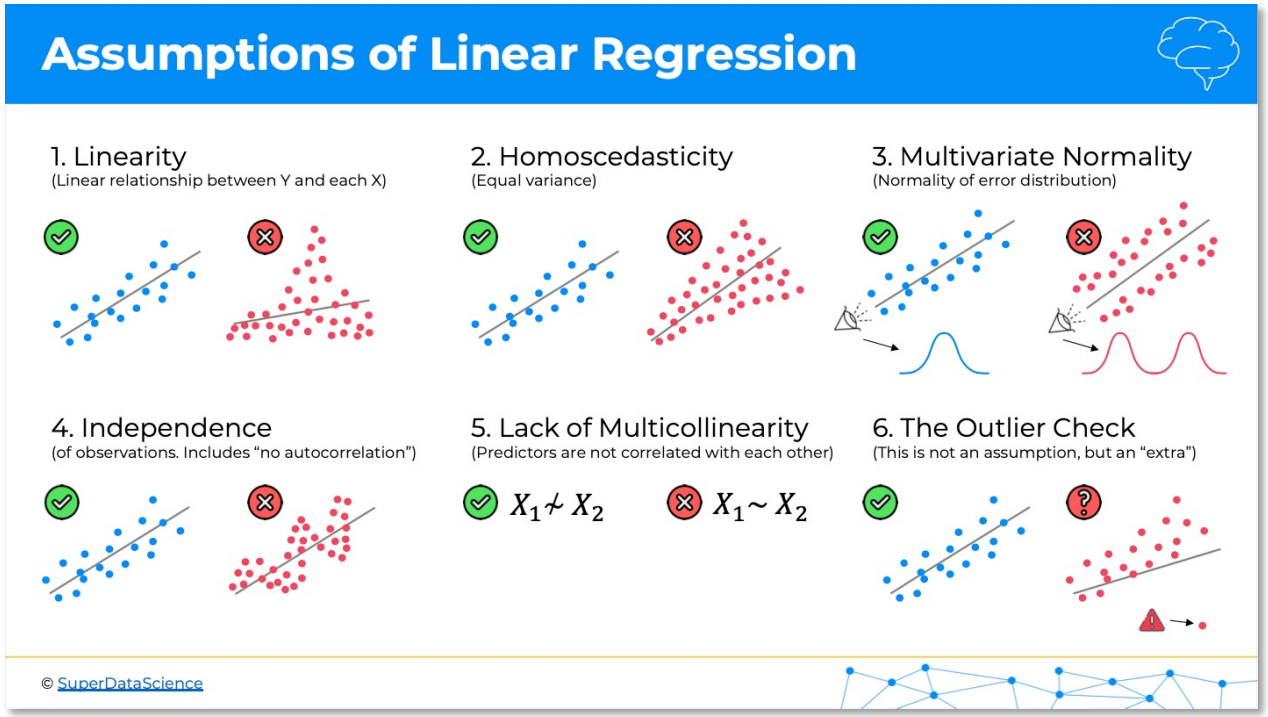
6. The Outlier Check
(This is not an assumption, but an “extra”)



Bonus

Download the Assumptions poster at:

superdatascience.com/assumptions



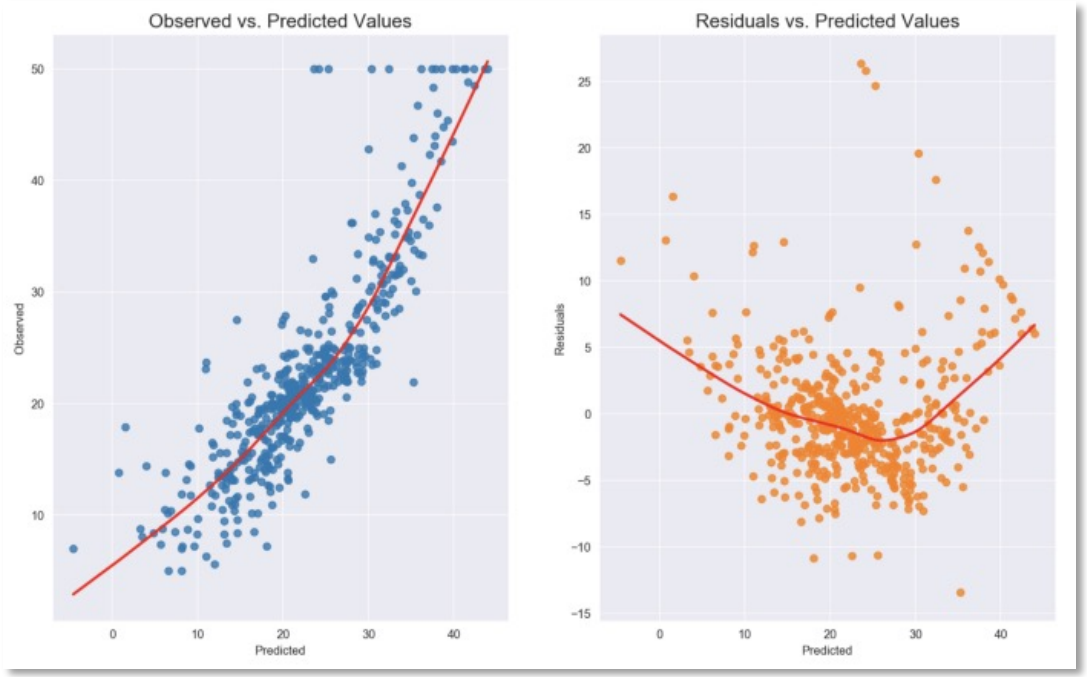
Additional Reading

Verifying the Assumptions of Linear Regression in Python and R

Eryk Lewinson (2019)

Link:

towardsdatascience.com/verifying-the-assumptions-of-linear-regression-in-python-and-r-f4cd2907d4c0



Dummy Variables

Dummy Variables

Profit	R&D Spend	Admin	Marketing	State
192,261.83	165,349.20	136,897.80	471,784.10	New York
191,792.06	162,597.70	151,377.59	443,898.53	California
191,050.39	153,441.51	101,145.55	407,934.54	California
182,901.99	144,372.41	118,671.85	383,199.62	New York
166,187.94	142,107.34	91,391.77	366,168.42	California

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + b_3 * x_3 + \ ???$$

Dummy Variables

Profit	R&D Spend	Admin	Marketing	State
192,261.83	165,349.20	136,897.80	471,784.10	New York
191,792.06	162,597.70	151,377.59	443,898.53	California
191,050.39	153,441.51	101,145.55	407,934.54	California
182,901.99	144,372.41	118,671.85	383,199.62	New York
166,187.94	142,107.34	91,391.77	366,168.42	California

Dummy Variables

New York	California
1	0
0	1
0	1
1	0
0	1

$$y = b_0 + b_1*x_1 + b_2*x_2 + b_3*x_3 + b_4*D_1$$

Dummy Var. Trap

Dummy Variable Trap

Profit	R&D Spend	Admin	Marketing	State
192,261.83	165,349.20	136,897.80	471,784.10	New York
191,792.06	162,597.70	151,377.59	443,898.53	California
191,050.39	153,441.51	101,145.55	407,934.54	California
182,901.99	144,372.41	118,671.85	383,199.62	New York
166,187.94	142,107.34	91,391.77	366,168.42	California

Dummy Variables

New York	California
1	0
0	1
0	1
1	0
0	1

$$y = b_0 + b_1*x_1 + b_2*x_2 + b_3*x_3 + b_4*D_1$$

Dummy Variable Trap

Profit	R&D Spend	Admin	Marketing	State
192,261.83	165,349.20	136,897.80	471,784.10	New York
191,792.06	162,597.70			nia
191,050.39	153,441.51			nia
182,901.99	144,372.41			ork
166,187.94	142,107.34			nia

Dummy Variables

New York	California
1	0
0	1
0	1
1	0
0	1

$$D_2 = 1 - D_1$$

$$y = b_0 + b_1*x_1 + b_2*x_2 + b_3*x_3 + b_4*D_1 + b_5*D_2$$

Dummy Variable Trap

Dummy Variables

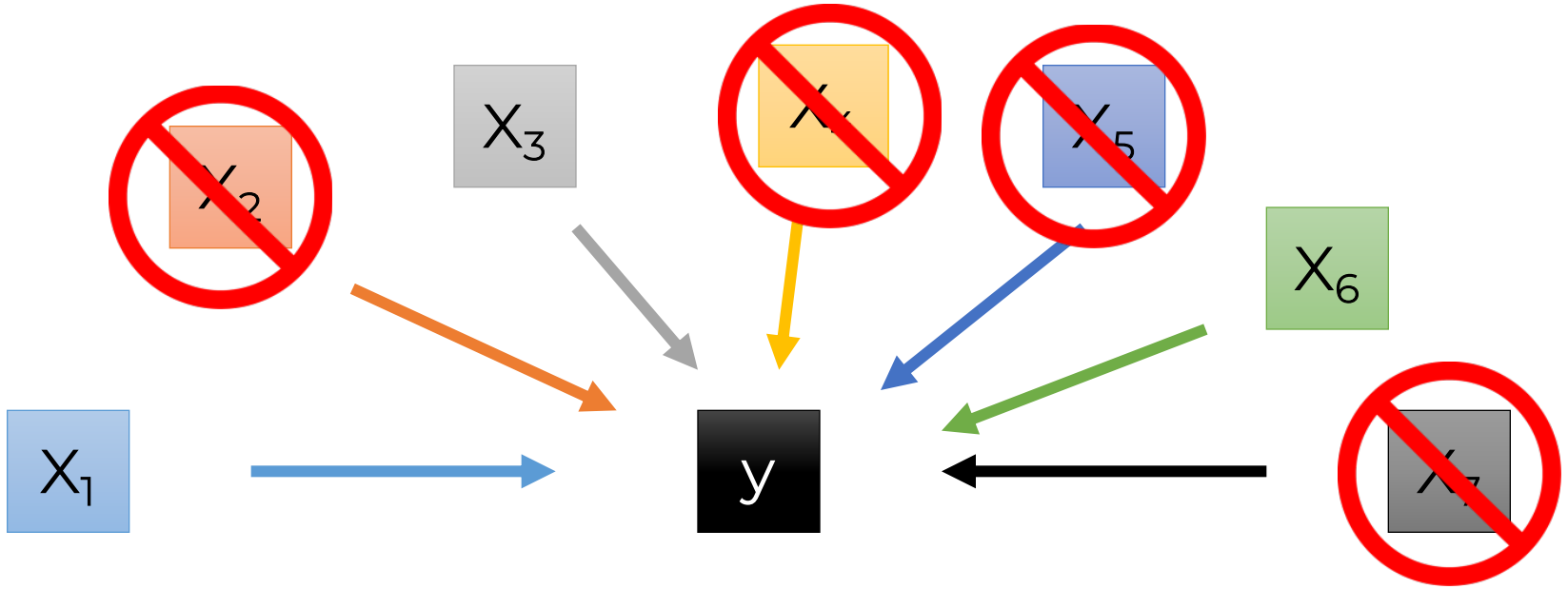
Profit	R&D Spend	Admin	Marketing	State	New York	California
192,261.83	165,349.20	136,897.80	471,784.10	New York	1	0
191,792.06	162,597.70	151,377.59	443,898.53	California	0	1
191,050.39	153,441.51	101,145.55	407,934.54	California	0	1
182,901.99	144,372.41	118,671.85	383,199.62	New York	1	0
166,187.94	142,107.34	91,391.77	366,168.42	California	0	1

$$y = b_0 + b_1*x_1 + b_2*x_2 + b_3*x_3 + b_4*D_1 + b_5*D_2$$

Always omit one dummy variable

Building A Model (Step-By-Step)

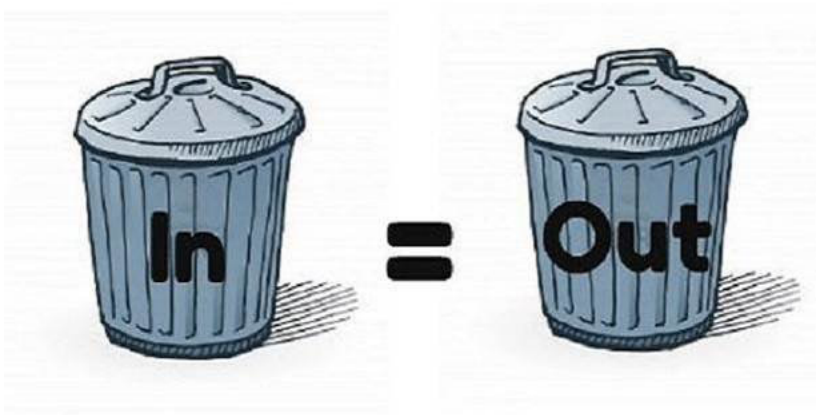
Building A Model



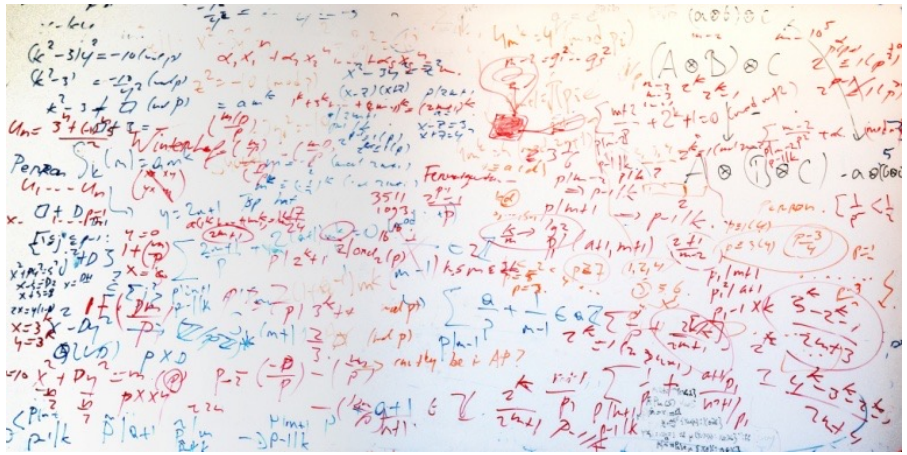
Why?

Building A Model

1)



2)



Building A Model

5 methods of building models:

1. All-in
2. Backward Elimination
3. Forward Selection
4. Bidirectional Elimination
5. Score Comparison

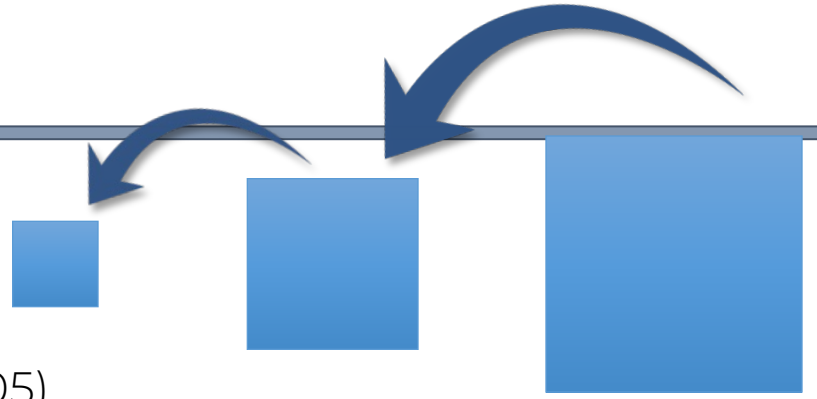
} Stepwise Regression

Building A Model

"All-in" – cases:

- Prior knowledge; OR
- You have to; OR
- Preparing for Backward Elimination

Building A Model



Backward Elimination

STEP 1: Select a significance level to stay in the model (e.g. SL = 0.05)

STEP 2: Fit the full model with all possible predictors

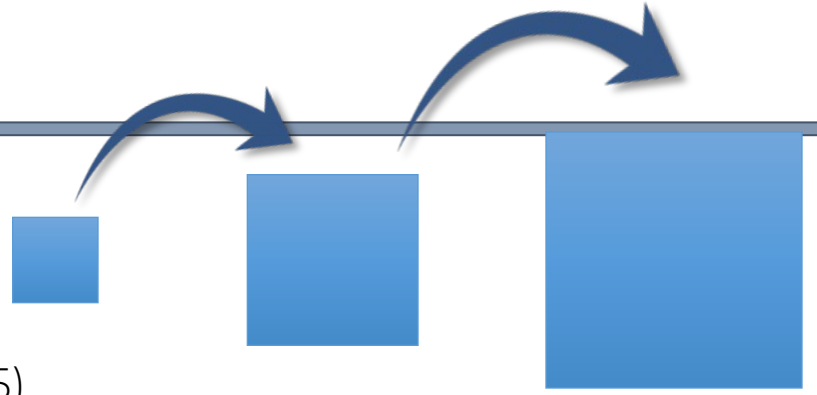
STEP 3: Consider the predictor with the highest P-value. If P > SL, go to STEP 4, otherwise go to FIN

STEP 4: Remove the predictor

STEP 5: Fit model without this variable*

FIN: Your Model Is Ready

Building A Model



Forward Selection

STEP 1: Select a significance level to enter the model (e.g. SL = 0.05)

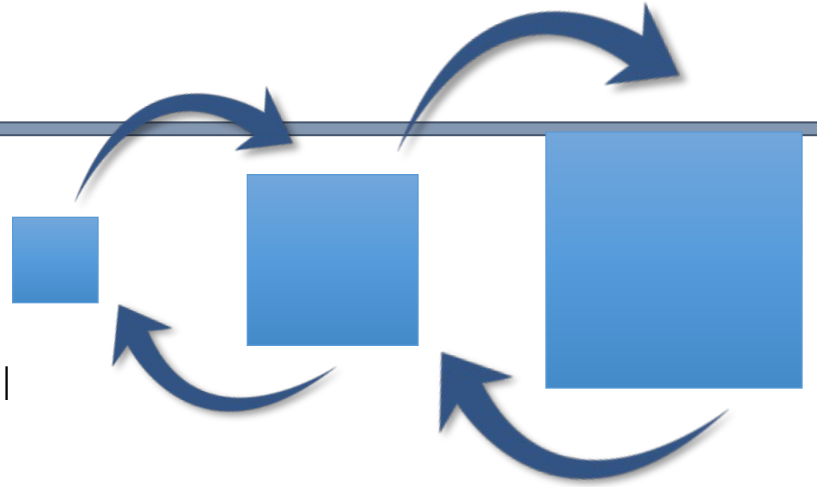
STEP 2: Fit all simple regression models $y \sim x_n$ Select the one with the lowest P-value

STEP 3: Keep this variable and fit all possible models with one extra predictor added to the one(s) you already have

STEP 4: Consider the predictor with the lowest P-value. If P < SL, go to STEP 3, otherwise go to FIN

FIN: Keep the previous model

Building A Model



Bidirectional Elimination

STEP 1: Select a significance level to enter and to stay in the model
e.g.: SLENTER = 0.05, SLSTAY = 0.05

STEP 2: Perform the next step of Forward Selection (new variables must have: P < SLENTER to enter)

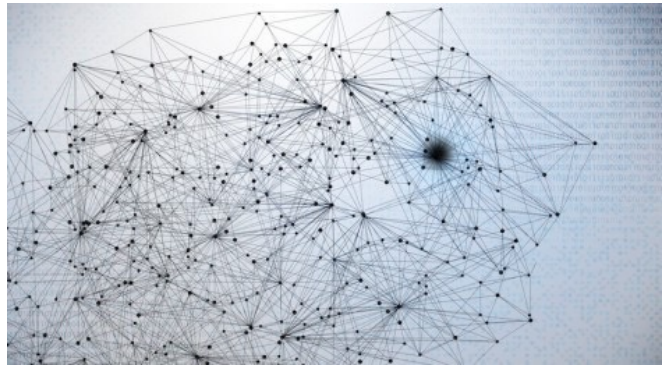
STEP 3: Perform ALL steps of Backward Elimination (old variables must have P < SLSTAY to stay)

STEP 4: No new variables can enter and no old variables can exit

FIN: Your Model Is Ready

Building A Model

All Possible Models



STEP 1: Select a criterion of goodness of fit (e.g. Akaike criterion)

STEP 2: Construct All Possible Regression Models: 2^N-1 total combinations

STEP 3: Select the one with the best criterion

FIN: Your Model Is Ready

Example:
10 columns means
1,023 models

Building A Model

5 methods of building models:

1. All-in
2. Backward Elimination
3. Forward Selection
4. Bidirectional Elimination
5. Score Comparison

Section Recap

Section Recap

In this section we learned:

1. How to create dummies for categorical IVs
2. How to avoid the dummy variable trap
3. Backward, Forward, Bidirectional, All Possible
4. We actually built a model. Step-By-Step!!
5. How to use adjusted R-squared in modelling
6. How to interpret coefficients of a MLR

Polynomial Regression

Regressions

Simple Linear Regression

$$y = b_0 + b_1 x_1$$

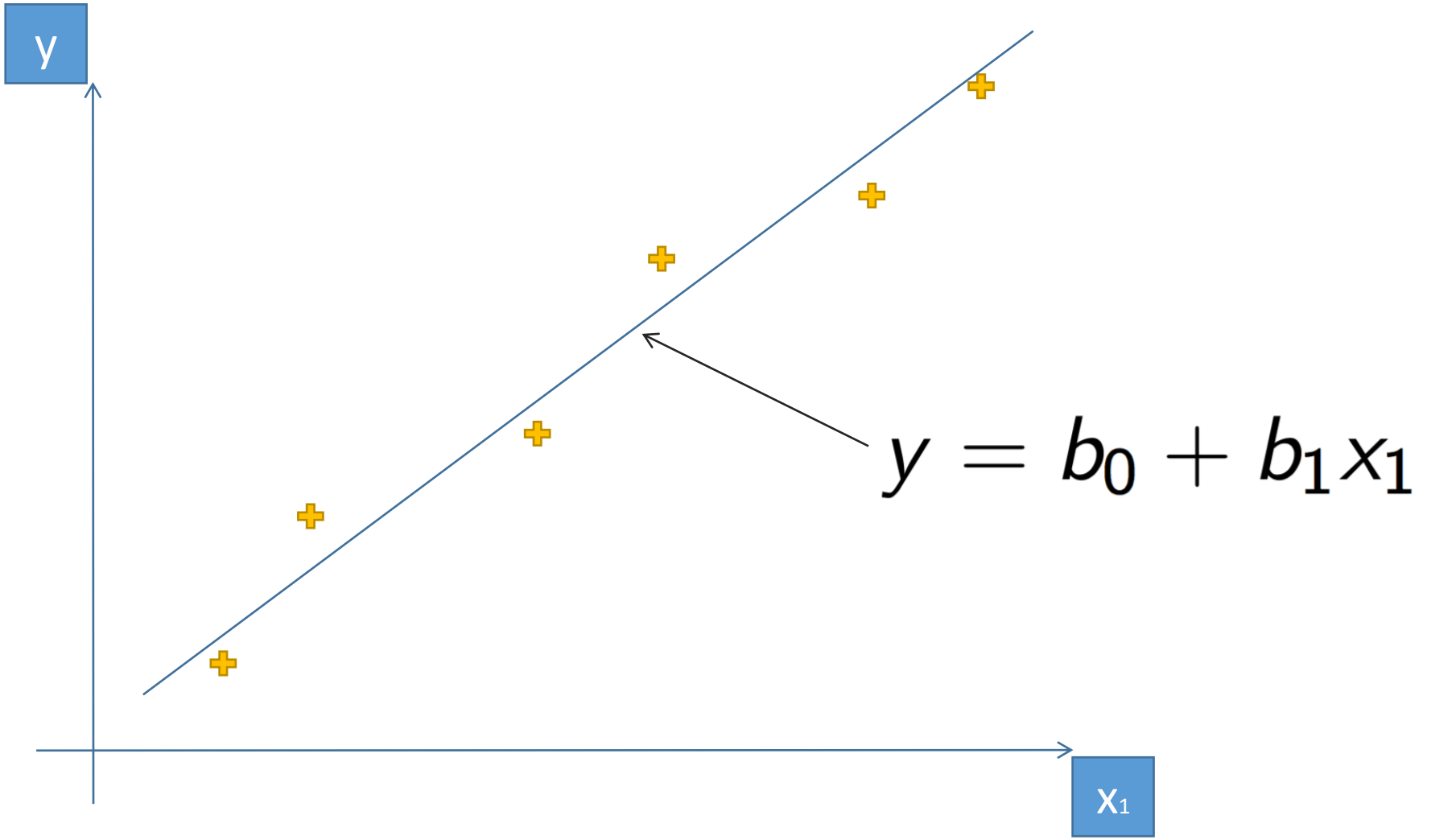
Multiple Linear Regression

$$y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$$

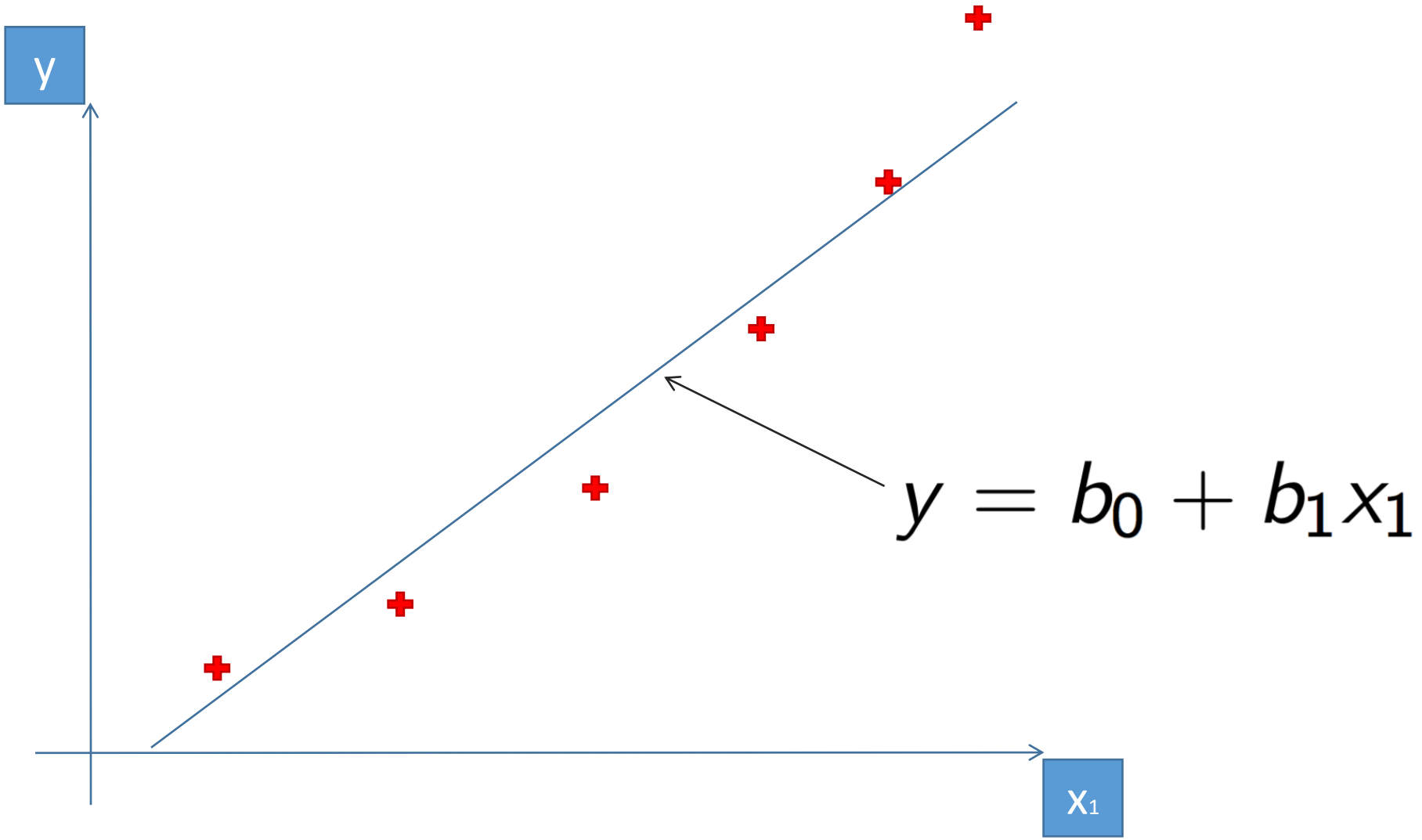
Polynomial Linear Regression

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \ldots + b_n x_1^n$$

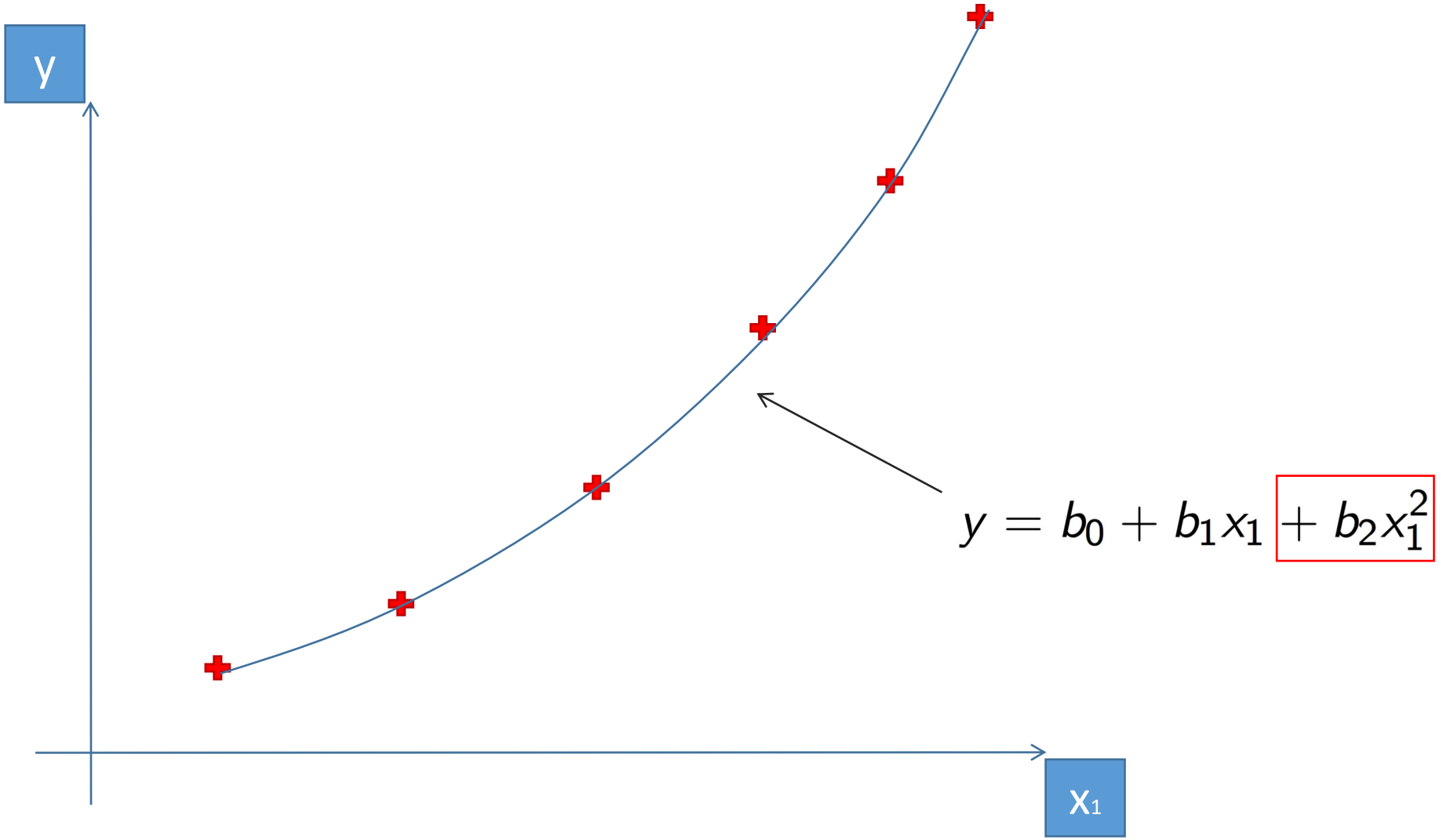
Simple Linear Regression



Simple Linear Regression



Polinomial Regression



Polinomial Regression

One Question: Why "Linear"?

Polinomial Regression

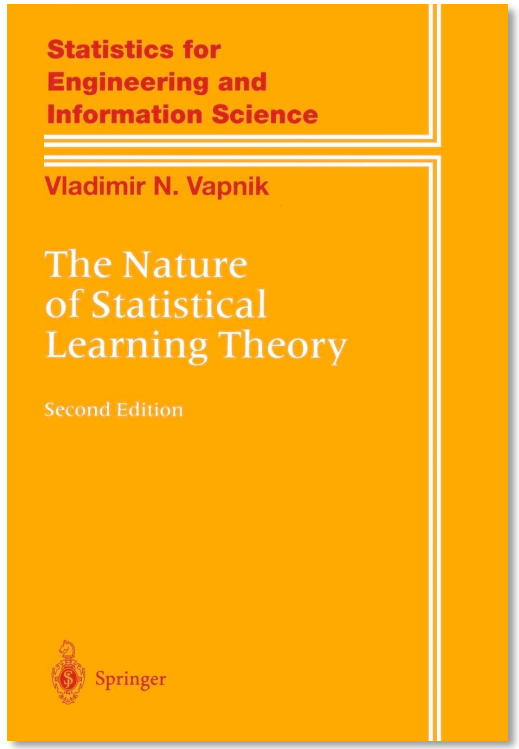
Polynomial Linear Regression

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \ldots + b_n x_1^n$$

SVR Intuition

SVR Intuition

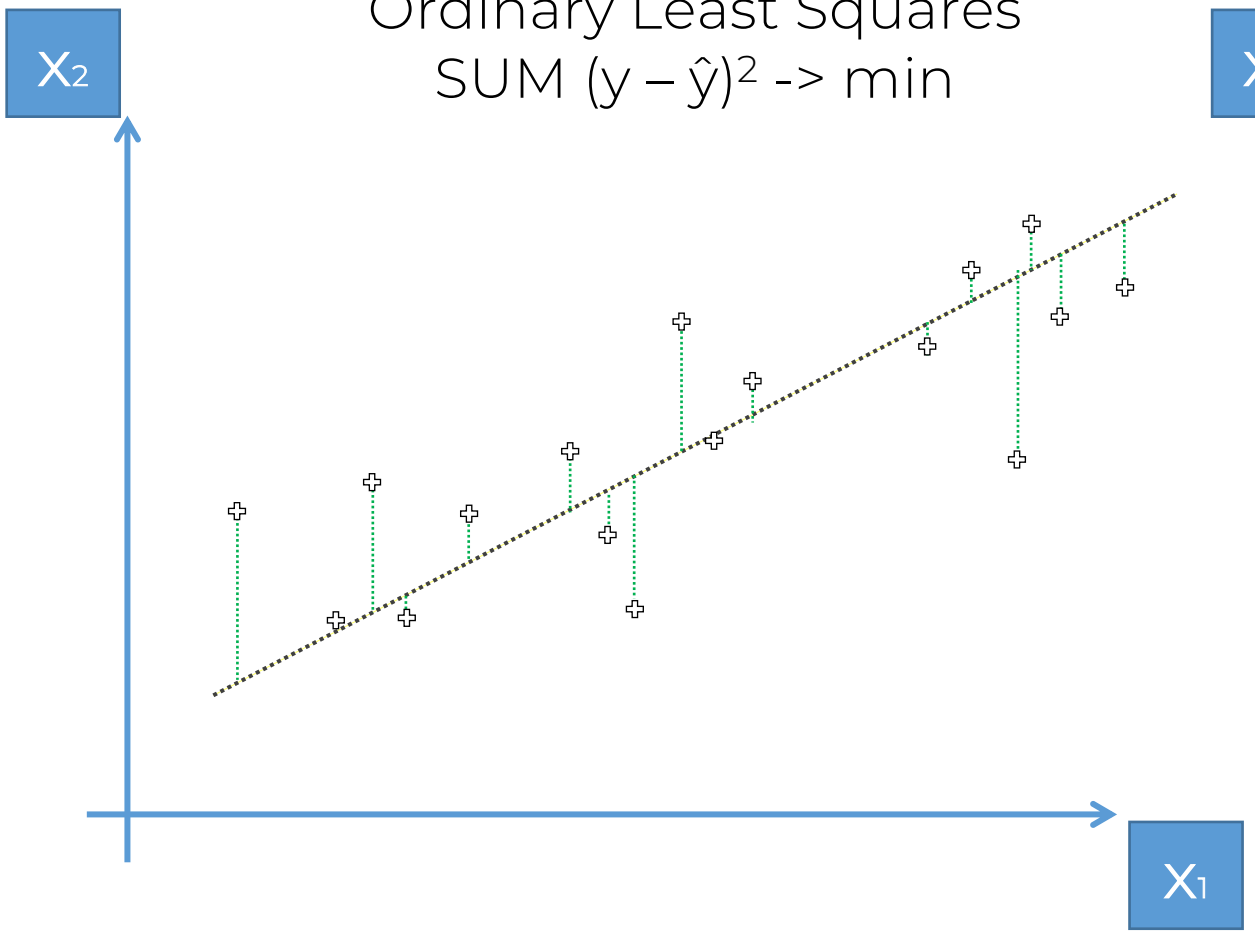
Vladimir Vapnik



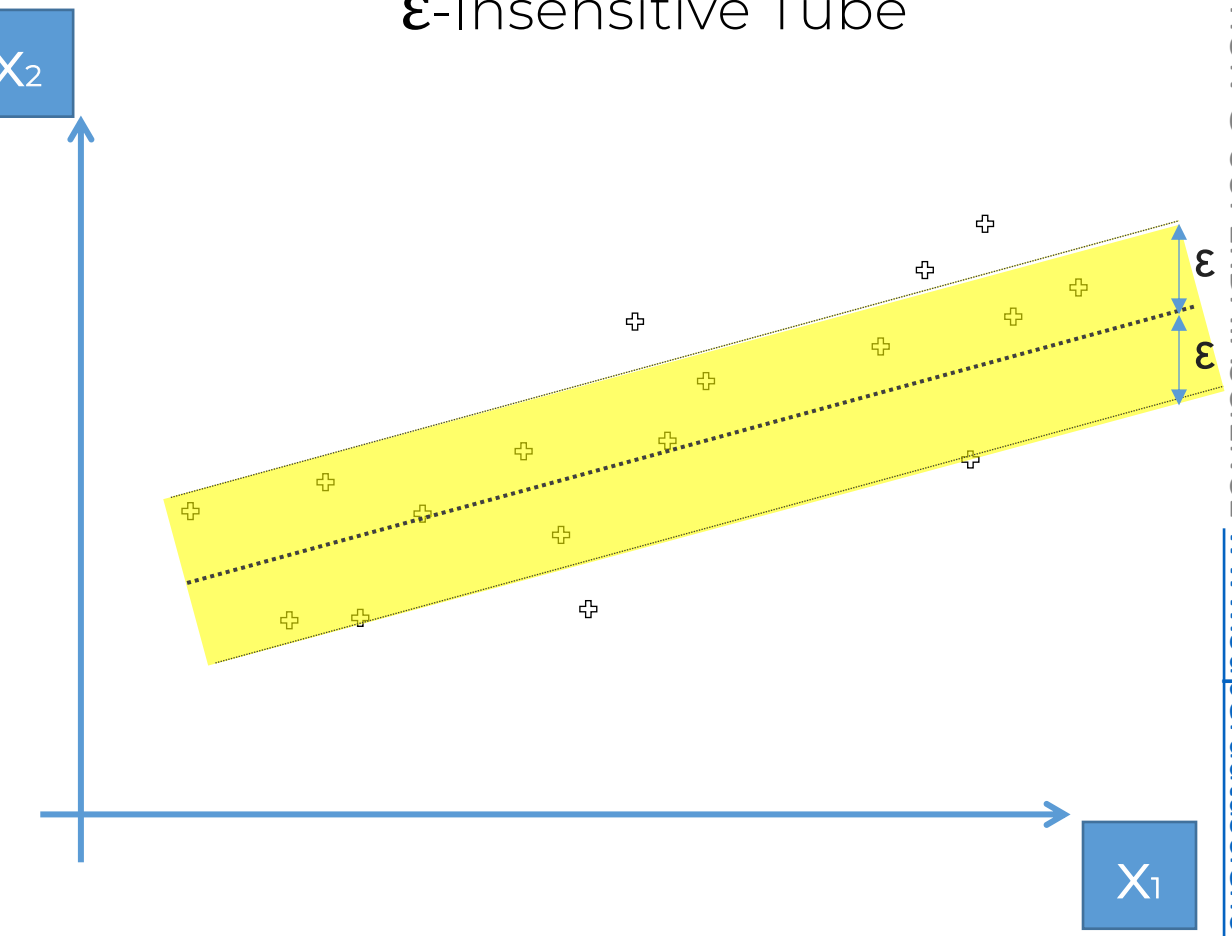
1992

SVR Intuition

Ordinary Least Squares
SUM $(y - \hat{y})^2$ -> min



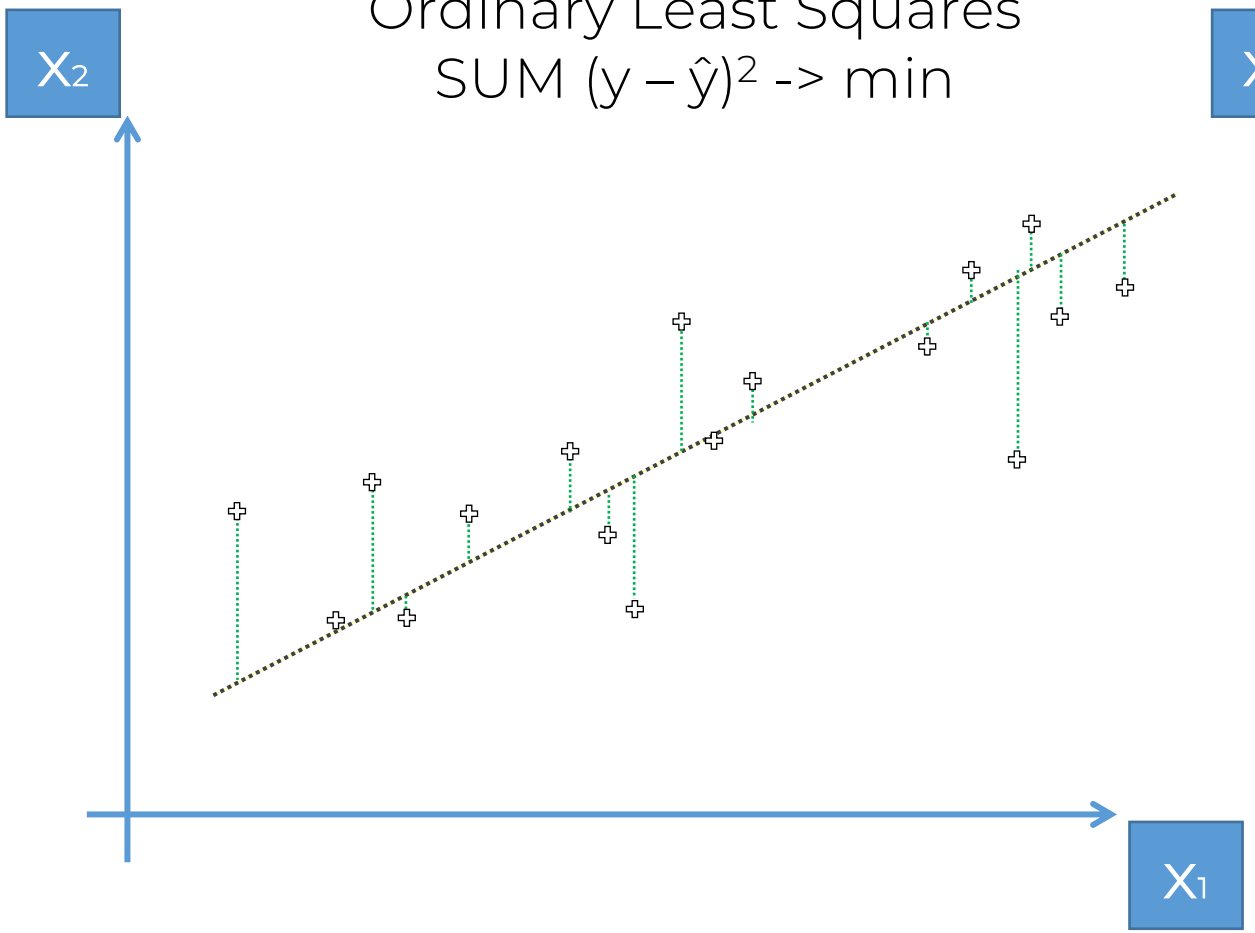
ε-Insensitive Tube



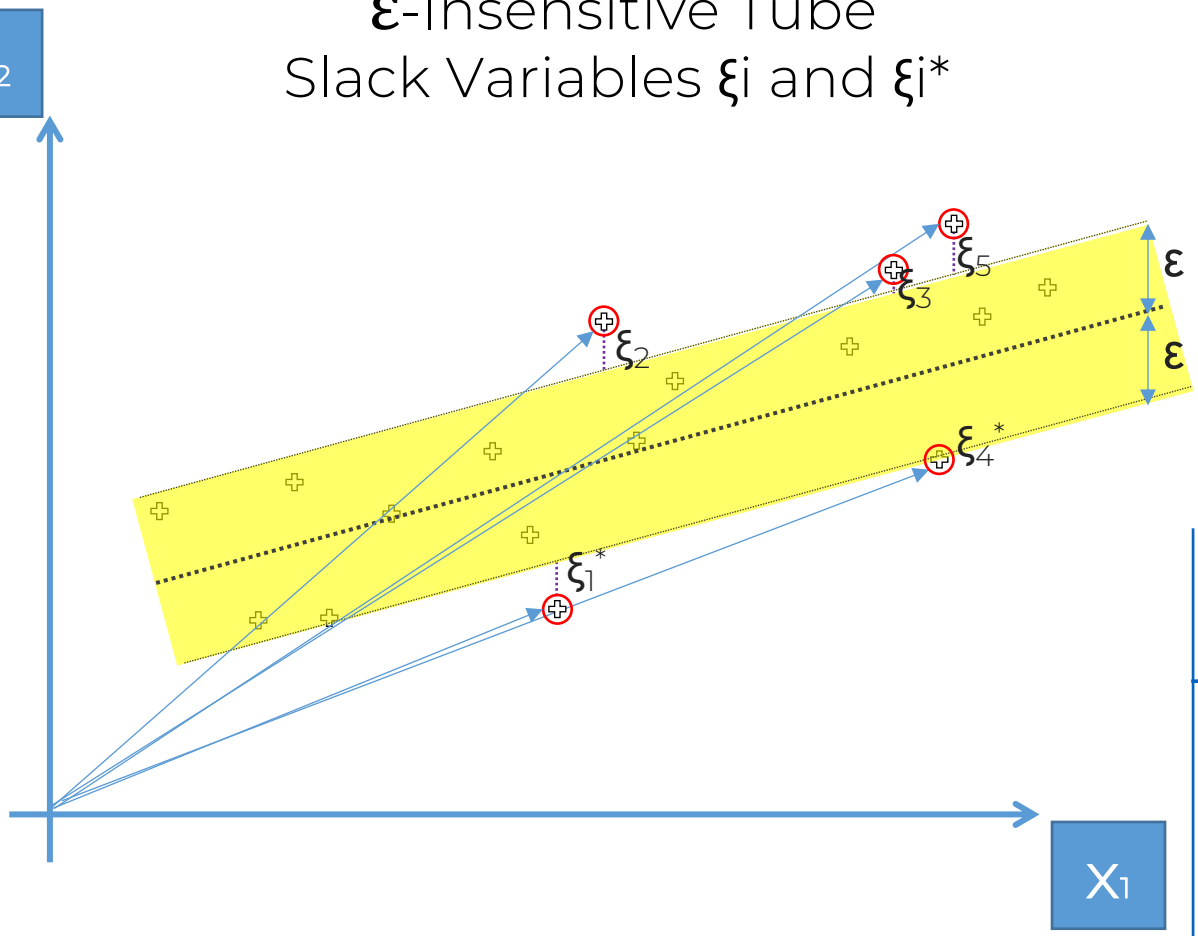
SVR Intuition

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*) \rightarrow min$$

Ordinary Least Squares
SUM $(y - \hat{y})^2$ -> min



ε-Insensitive Tube
Slack Variables ξi and ξi*



SVR Intuition

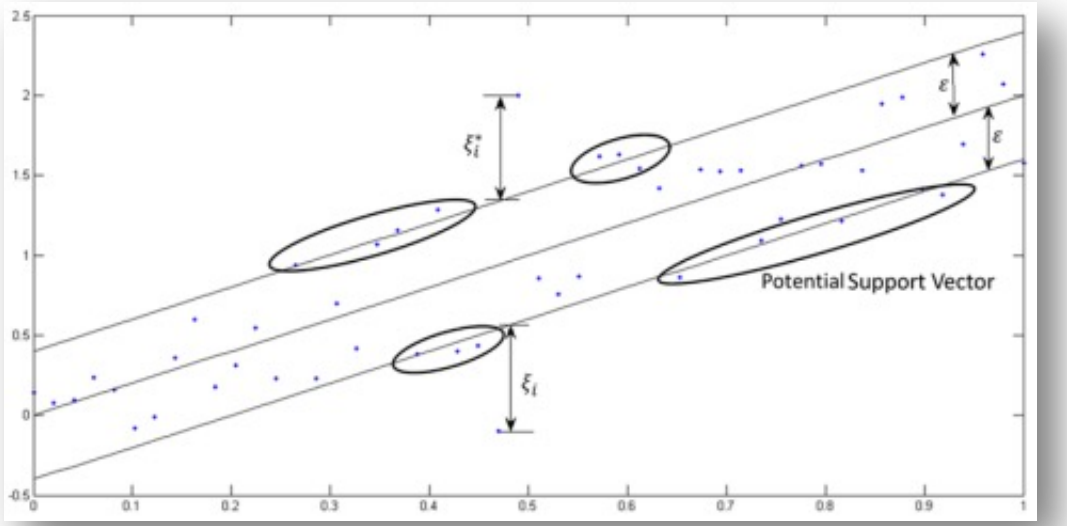
Additional Reading:

Chapter 4 – Support Vector Regression (from: Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers)

By Mariette Awad & Rahul Khanna (2015)

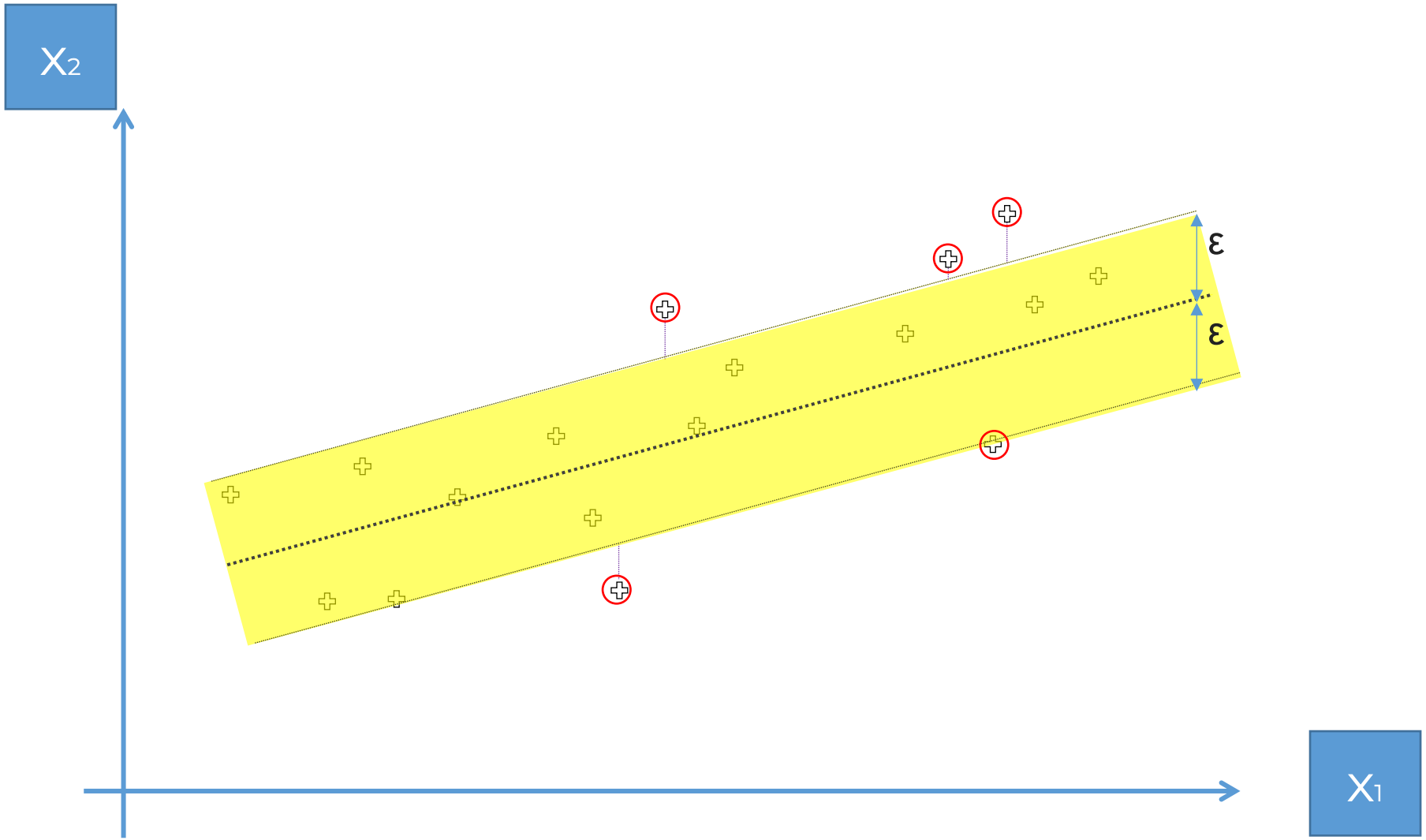
Link:

https://core.ac.uk/download/pdf/81523322.pdf



Heads-up about Non-Linear SVR

SVR Intuition



Python - Google Drive | support_vector_regression.ipynb | data_preprocessing_template.ipy | data_preprocessing_tools.ipynb | polynomial_regression.ipynb - C | Copy of support_vector_regressi

colab.research.google.com/drive/1oL2xxfVBcfvEuCNjRpTpywg6hSk81HKu#scrollTo=w_AG3rTAaVqQ

Copy of support_vector_regression.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

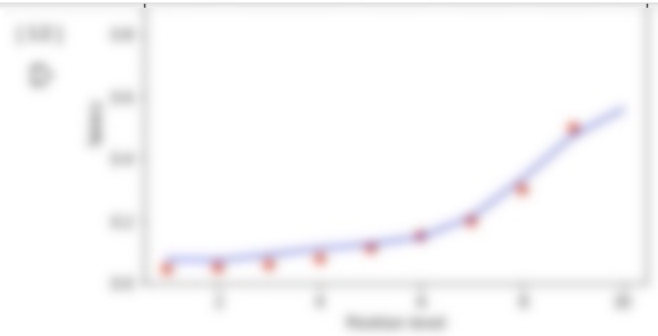
Files

Upload Refresh Mount Drive

..

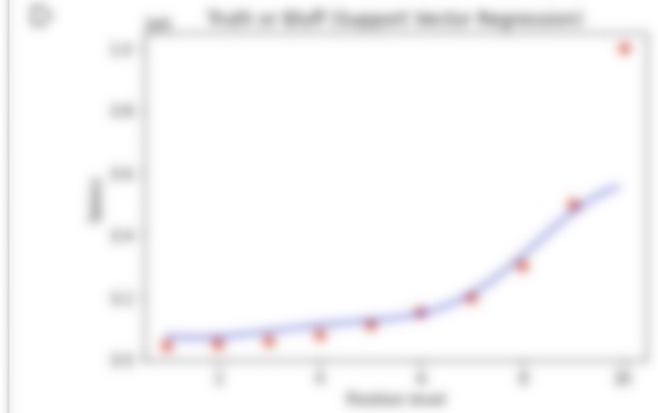
sample_data

Position_Salaries.csv



Visualising the SVR results (for higher resolution and smoother curve)

```
1 X_grid = np.arange(min(sc_X.inverse_transform(X)), max(sc_X.inverse_transform(X)), 0.1)
2 X_grid = X_grid.reshape((len(X_grid), 1))
3 plt.scatter(sc_X.inverse_transform(X), sc_y.inverse_transform(y), color = 'red')
4 plt.plot(X_grid, sc_y.inverse_transform(regressor.predict(sc_X.transform(X_grid))), color = 'blue')
5 plt.title('Truth or Bluff (Support Vector Regression)')
6 plt.xlabel('Position level')
7 plt.ylabel('Salary')
8 plt.show()
```



Position_Salaries.csv

1 to 10 of 10 entries Filter

Position	Level	Salary
Business Analyst	1	45000
Junior Consultant	2	50000
Senior Consultant	3	60000
Manager	4	80000
Country Manager	5	110000
Region Manager	6	150000
Partner	7	200000
Senior Partner	8	300000
C-level	9	500000
CEO	10	1000000

Show 10 per page

Disk 76.87 GB available

Heads-up about Non-Linear SVR

Section on SVM:

- SVM Intuition

Section on Kernel SVM:

- Kernel SVM Intuition
- Mapping to a higher dimension
- The Kernel Trick
- Types of Kernel Functions
- Non-linear Kernel SVR

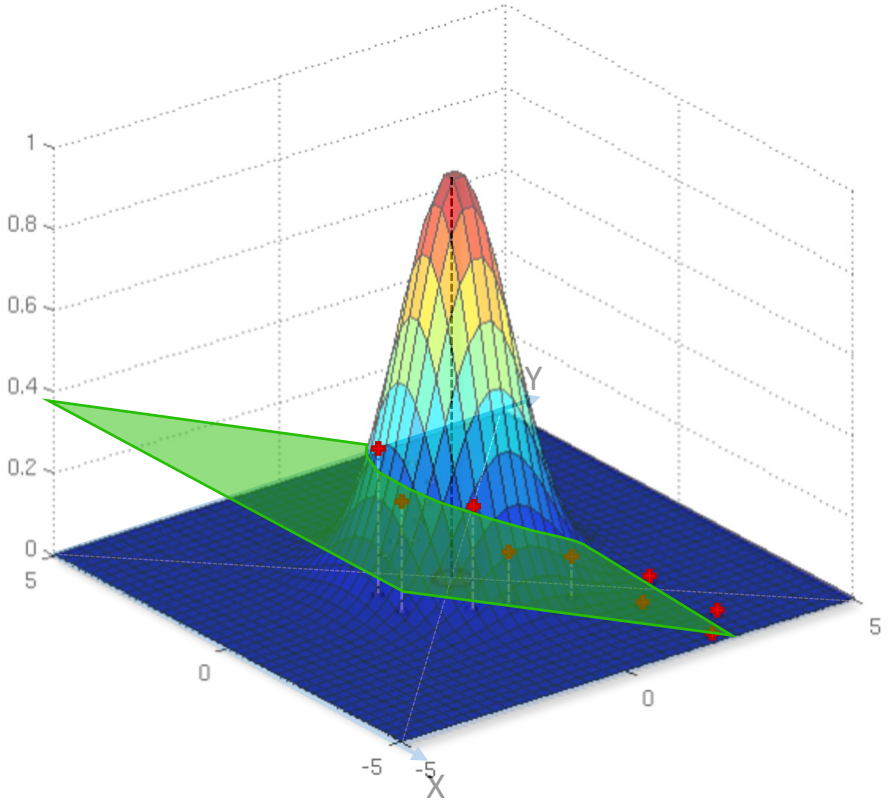
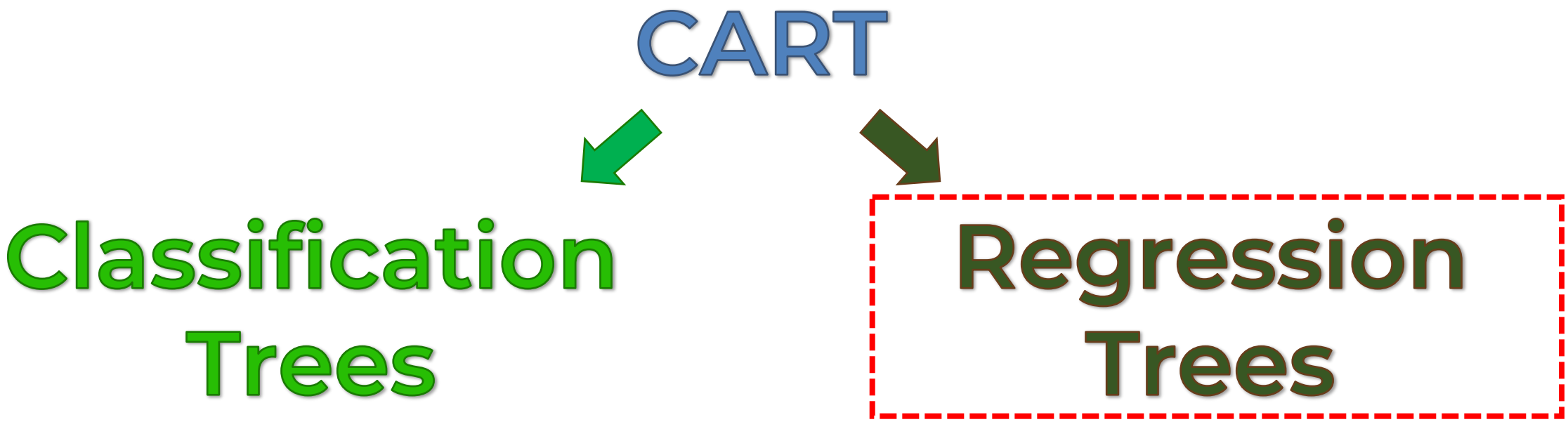


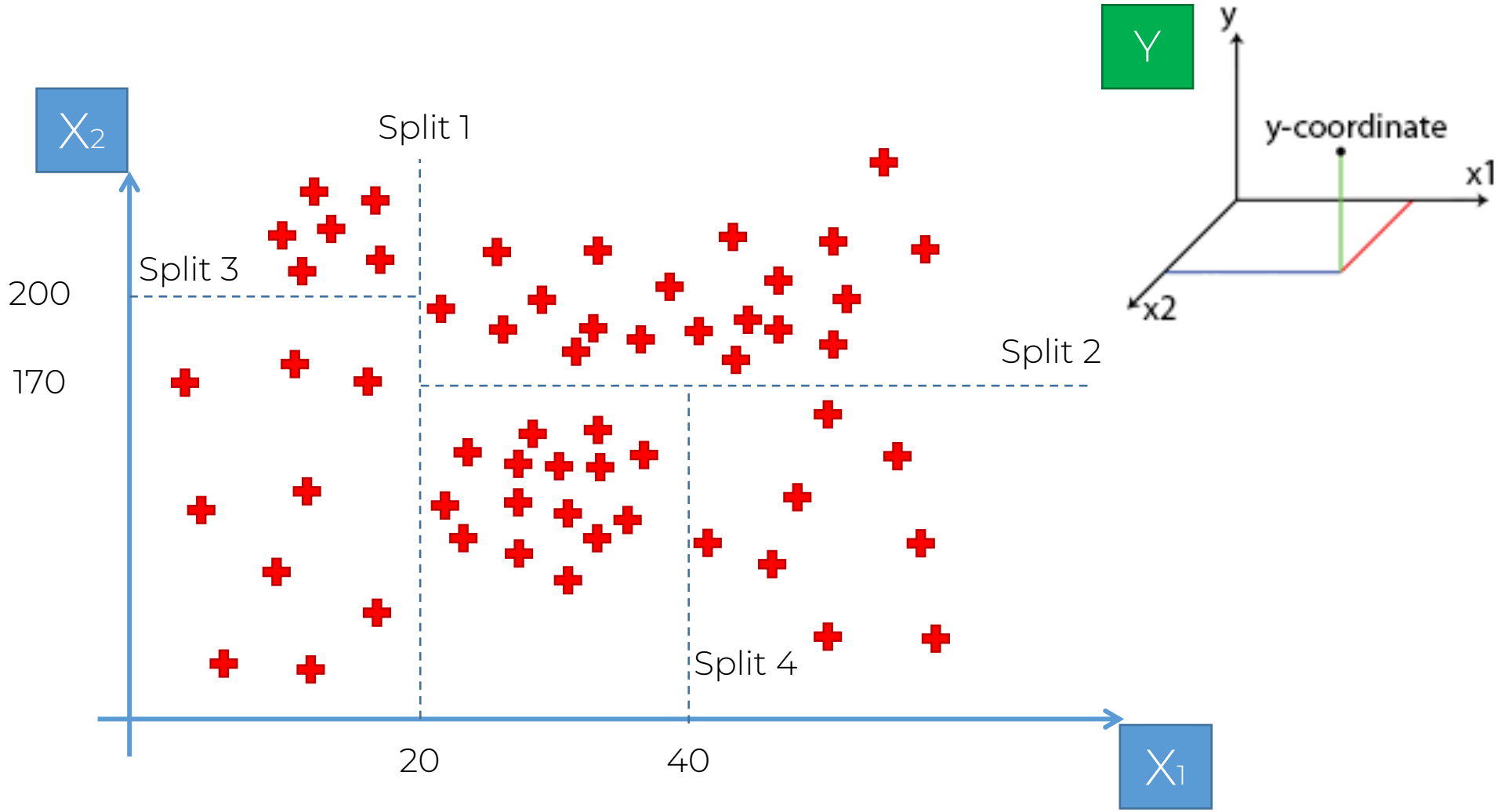
Image source: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html

Decision Tree Intuition

Decision Tree Intuition



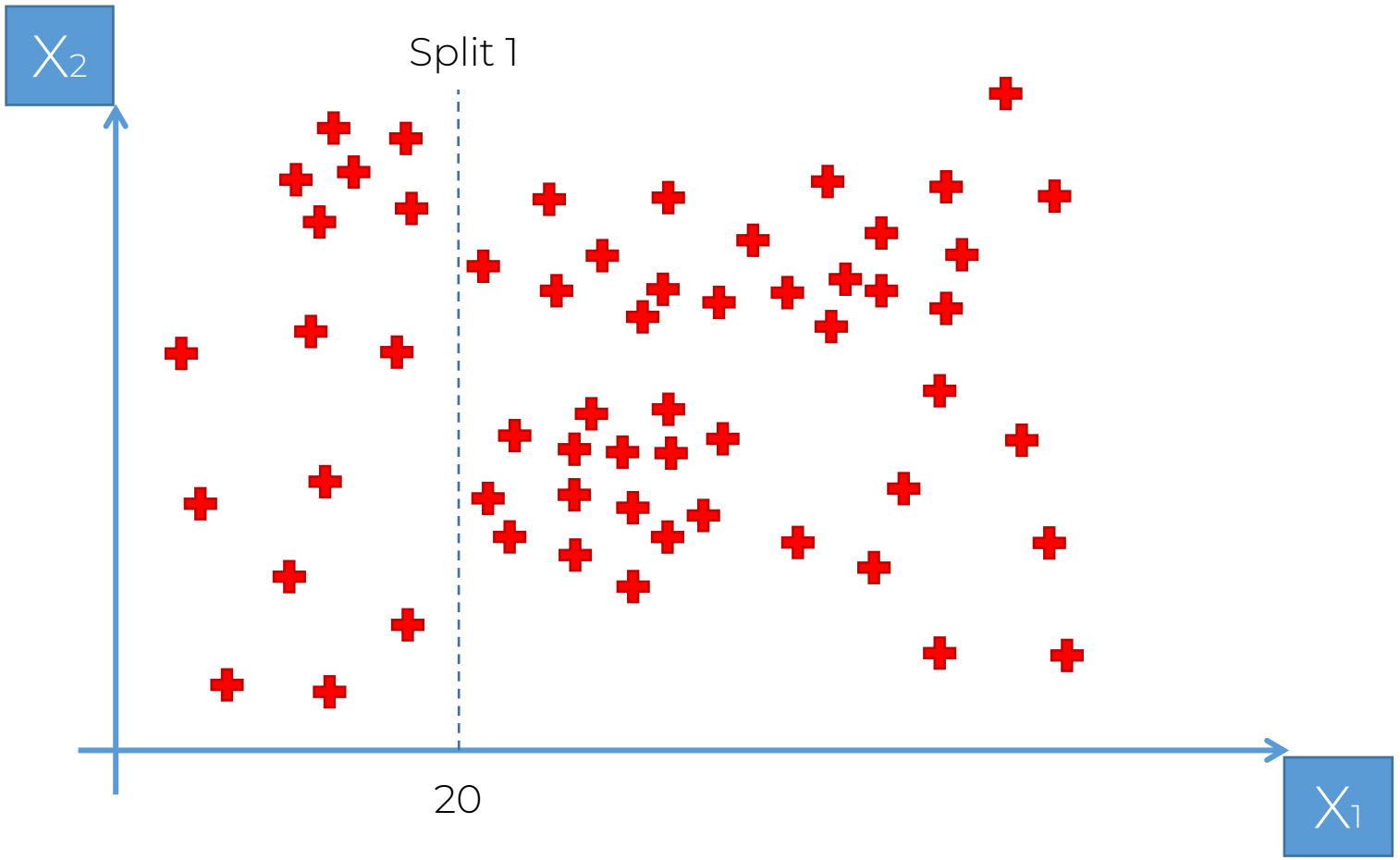
Decision Tree Intuition



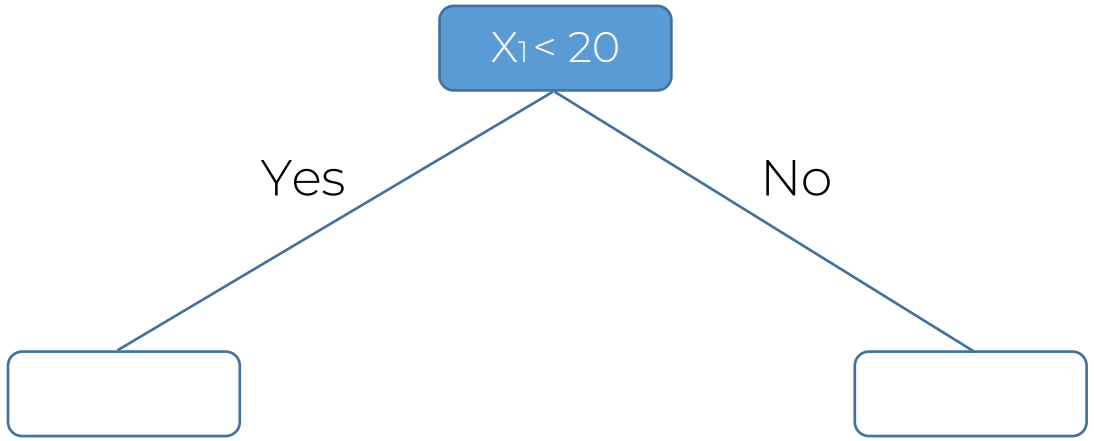
Decision Tree Intuition

Rewind...

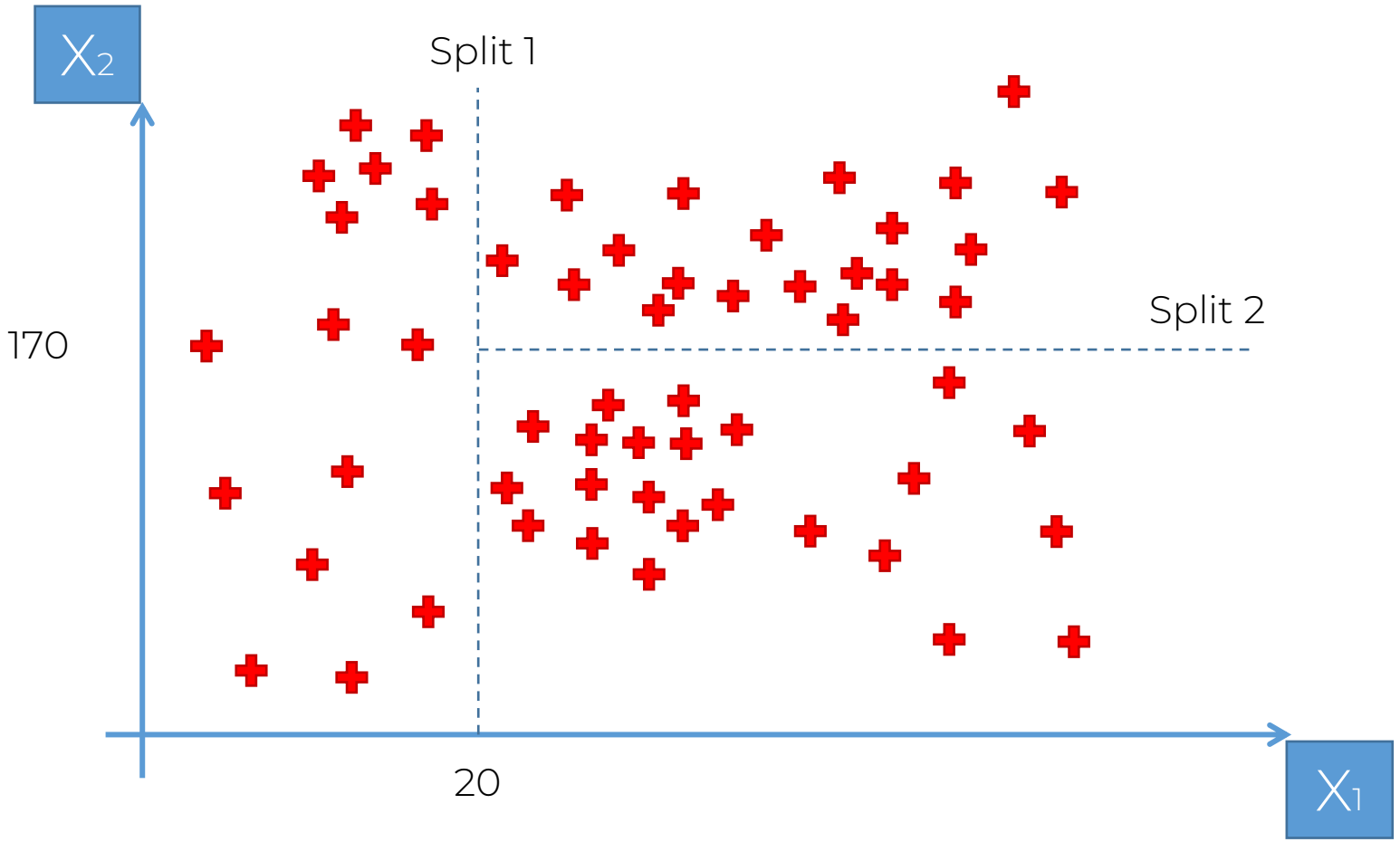
Decision Tree Intuition



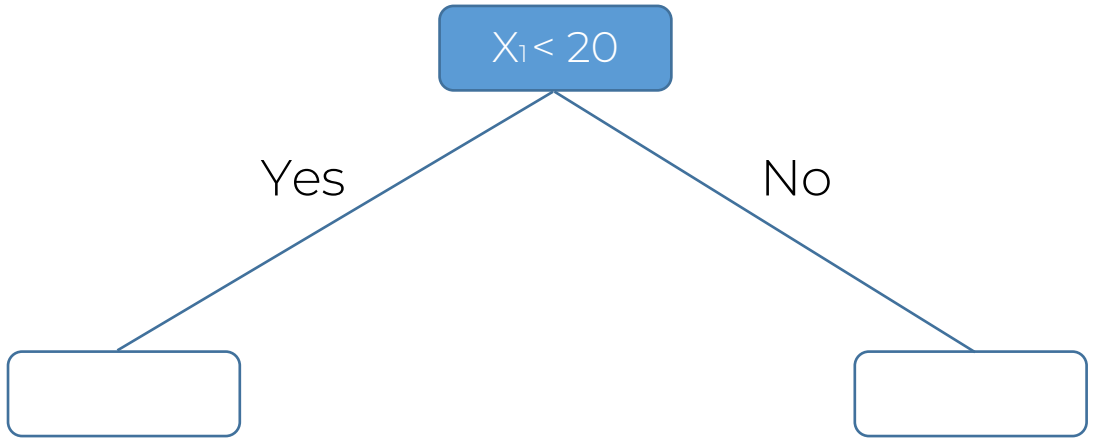
Decision Tree Intuition



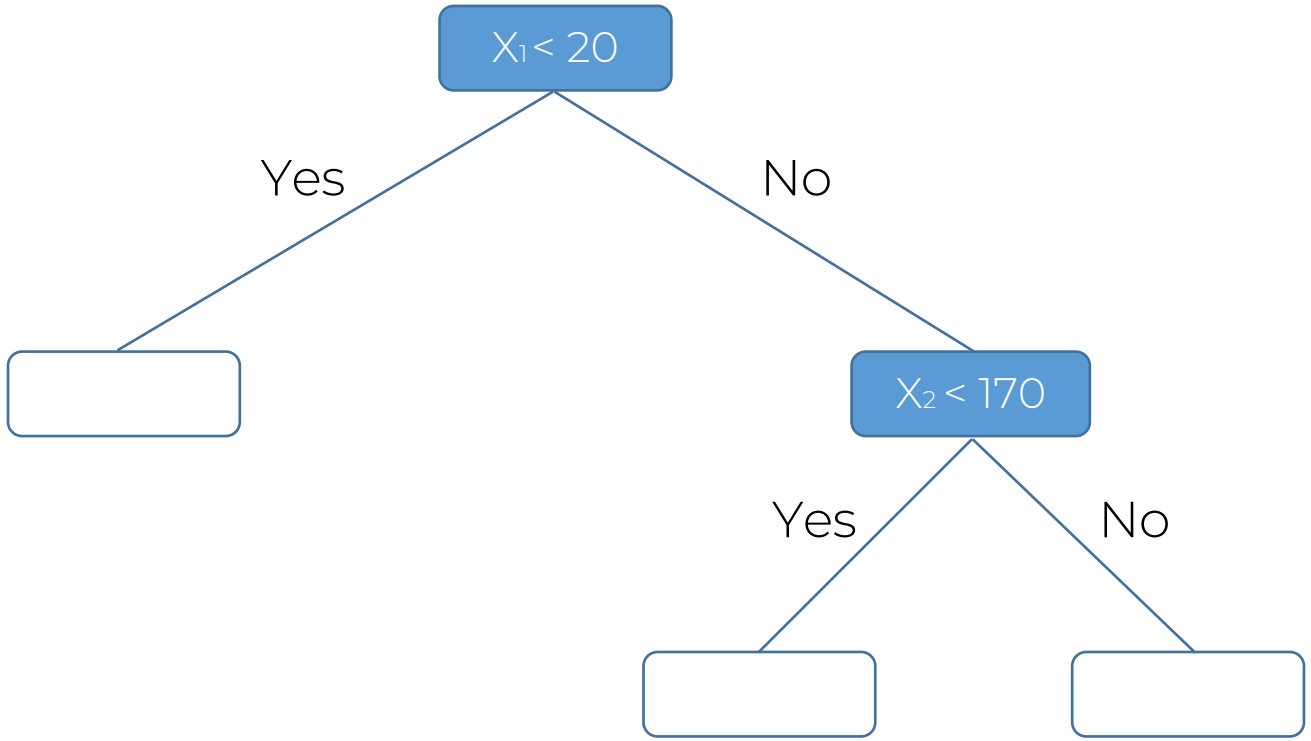
Decision Tree Intuition



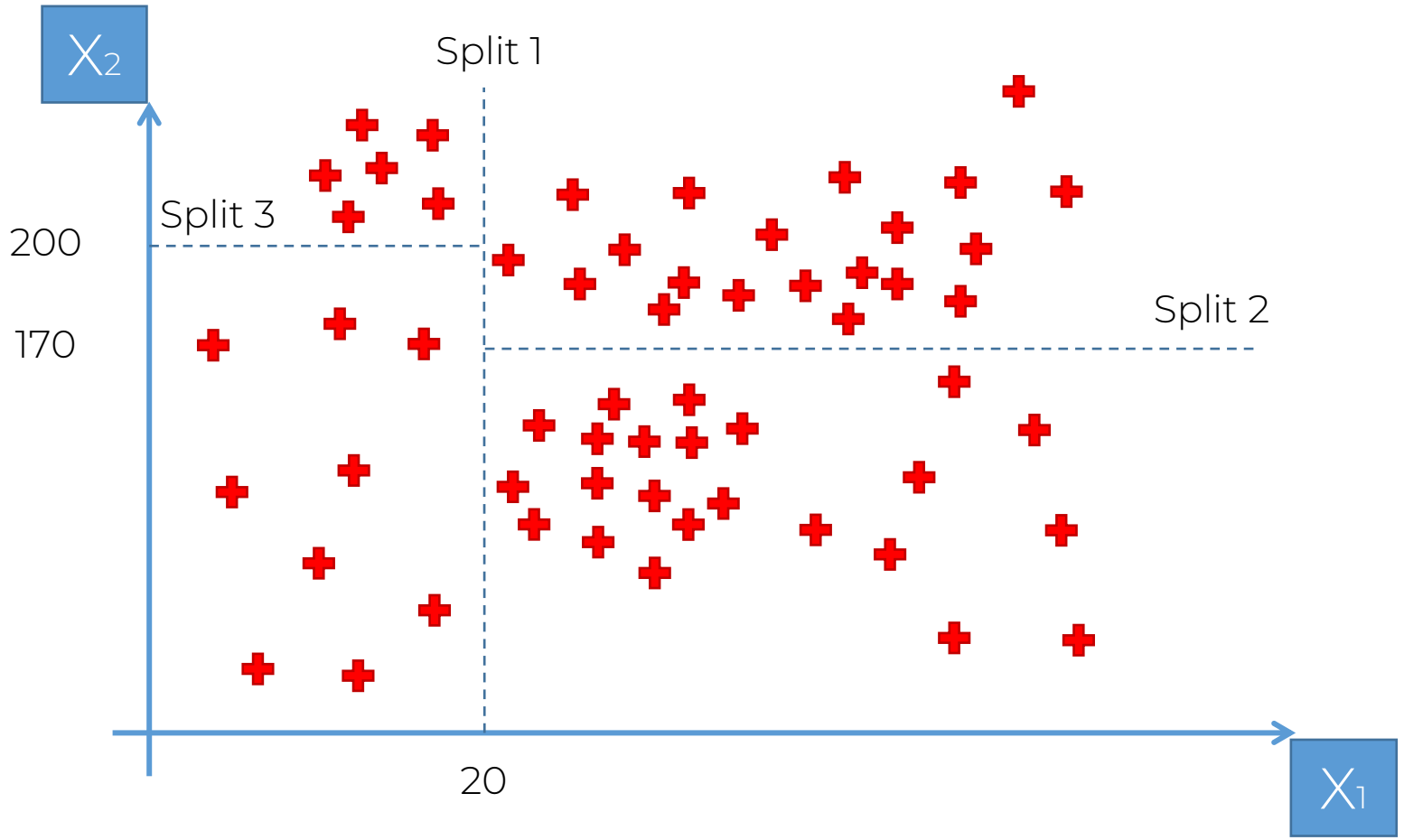
Decision Tree Intuition



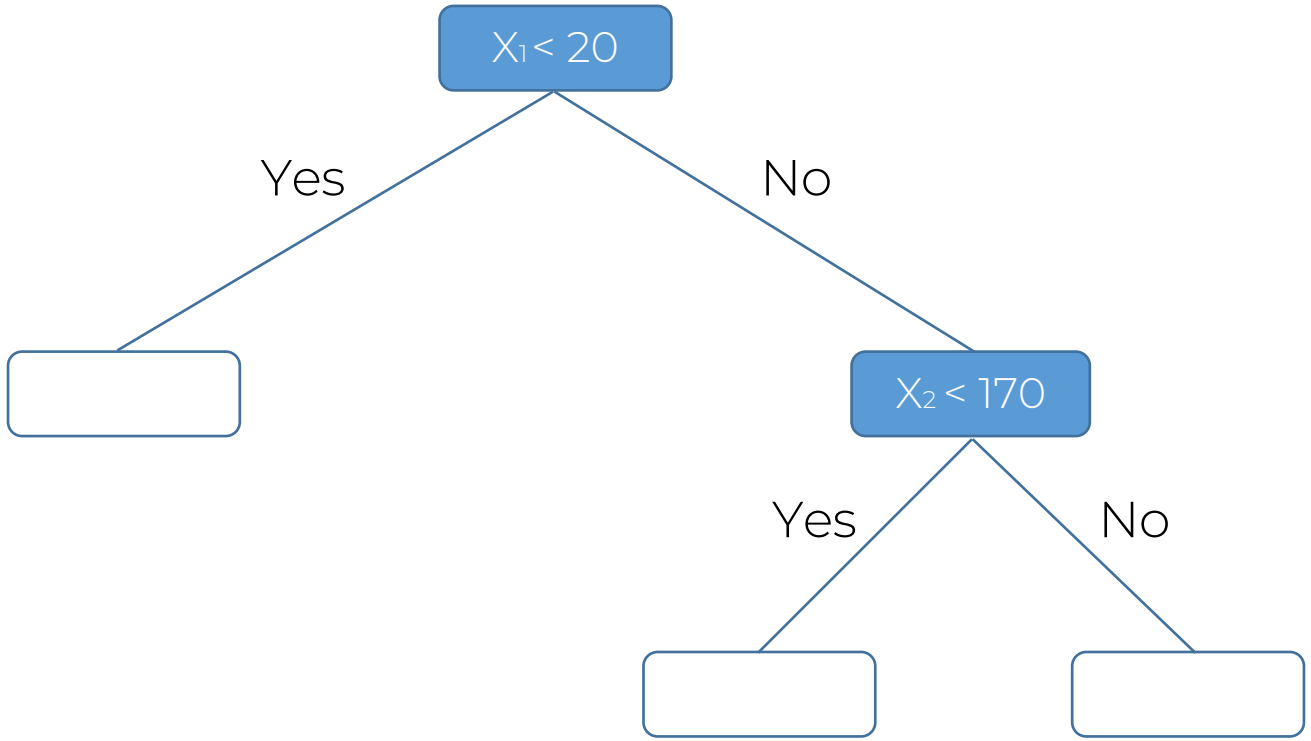
Decision Tree Intuition



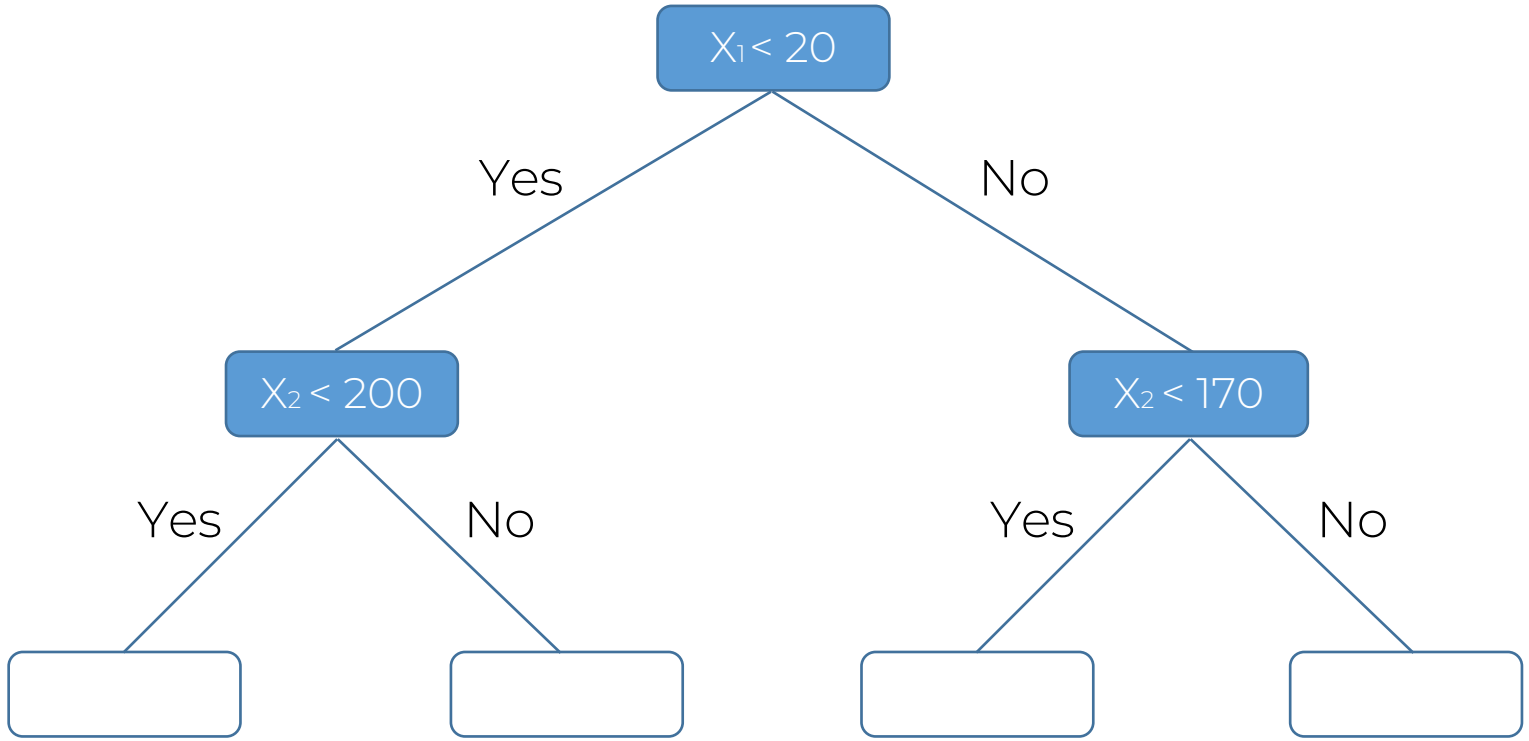
Decision Tree Intuition



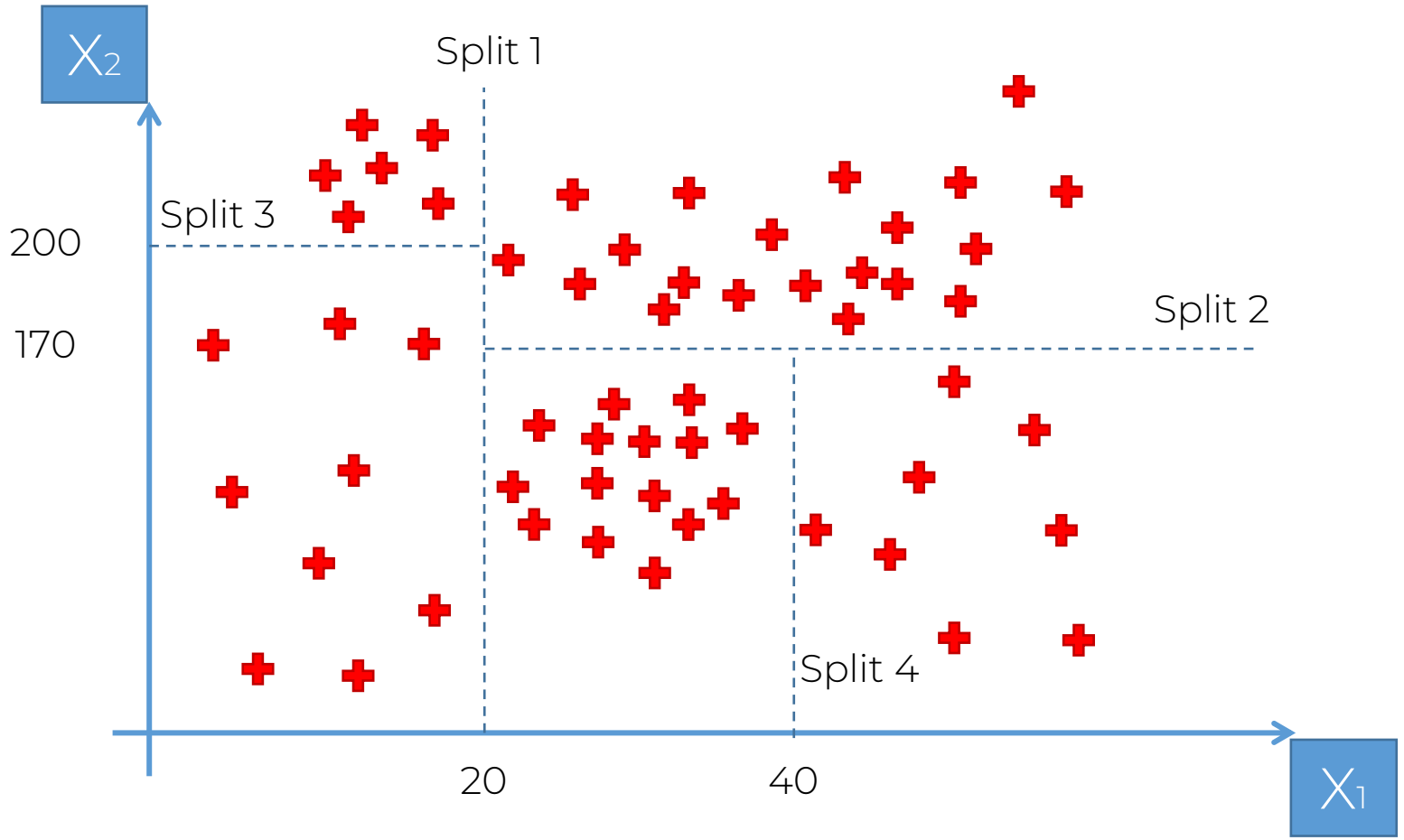
Decision Tree Intuition



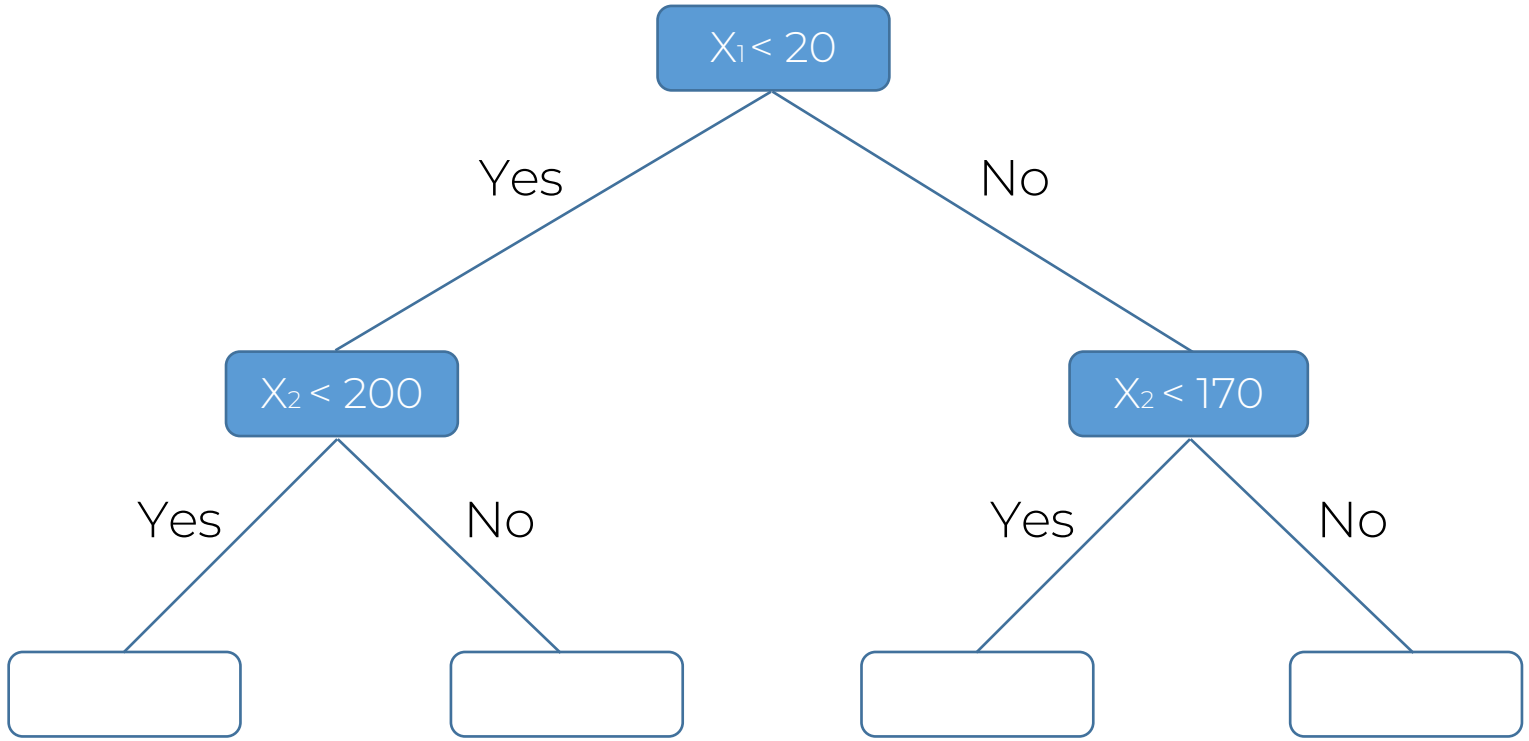
Decision Tree Intuition



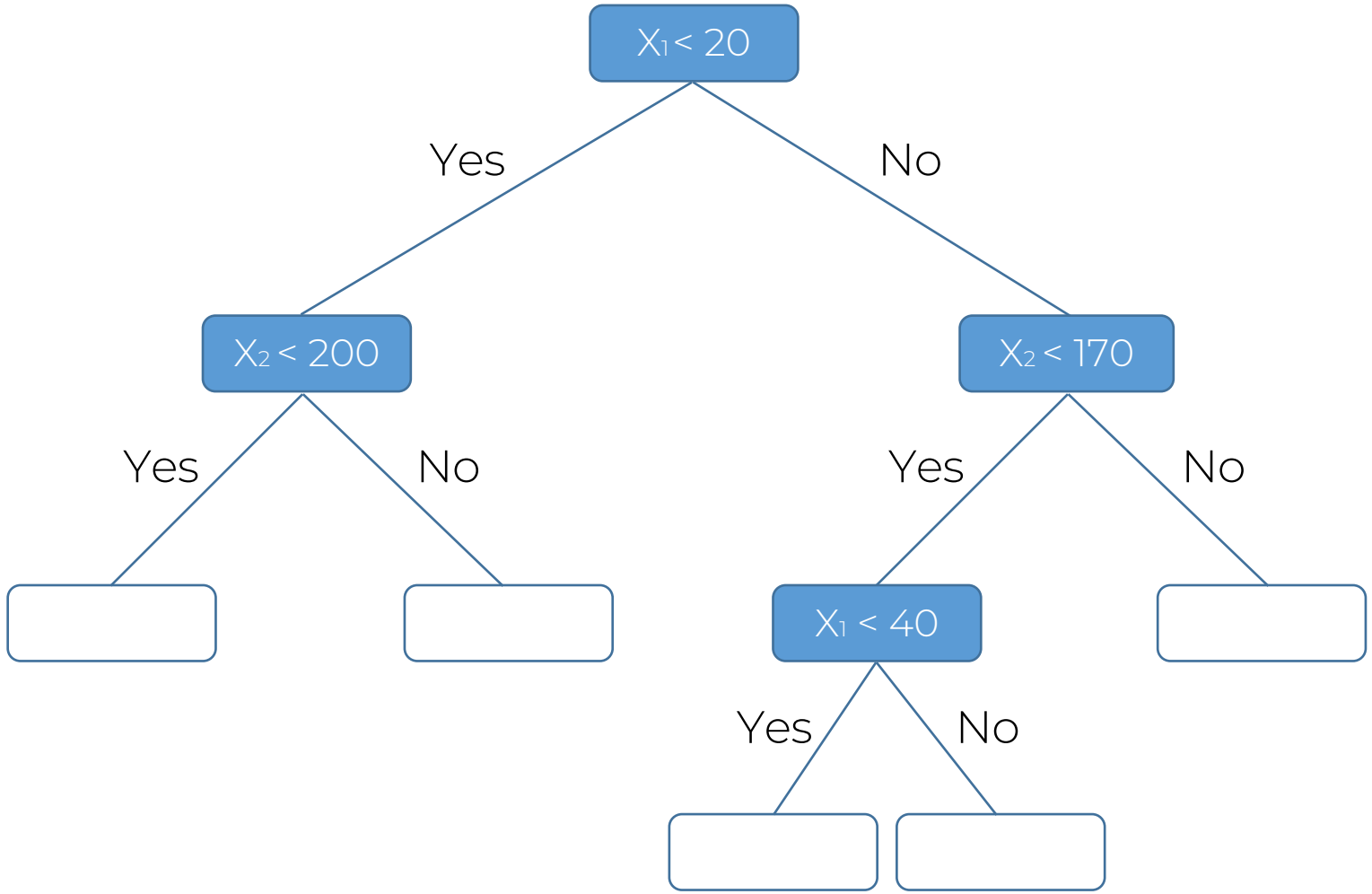
Decision Tree Intuition



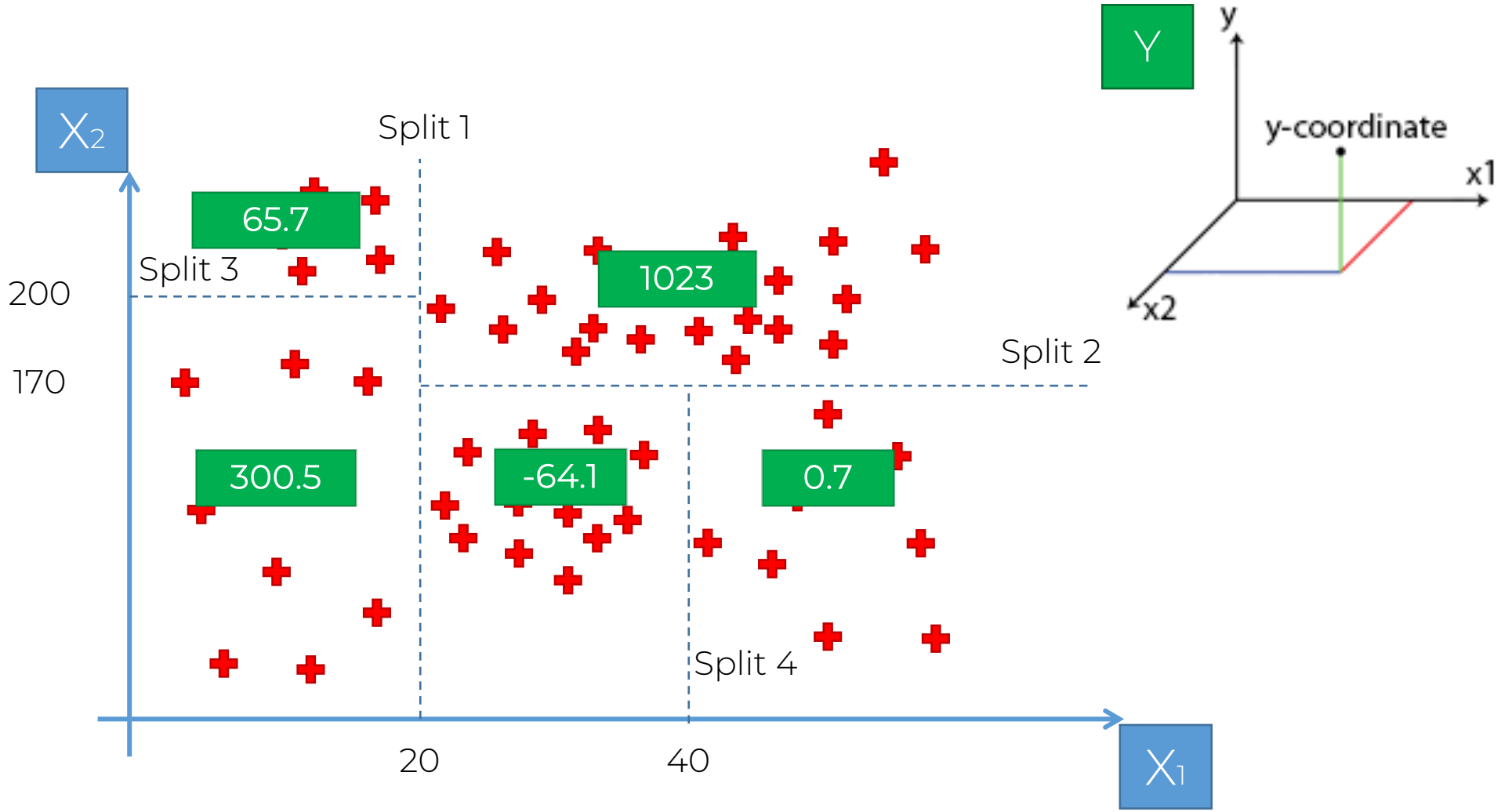
Decision Tree Intuition



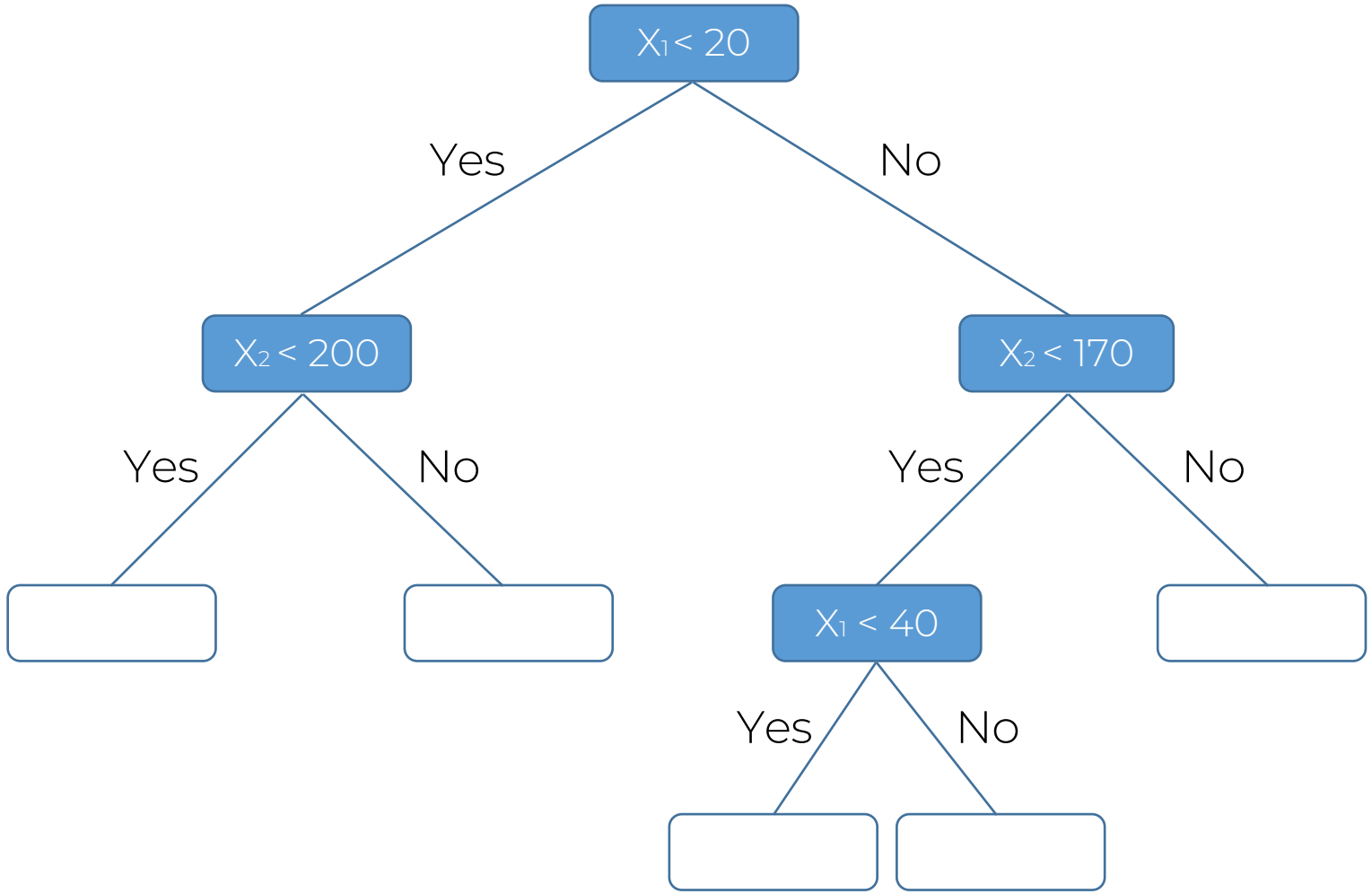
Decision Tree Intuition



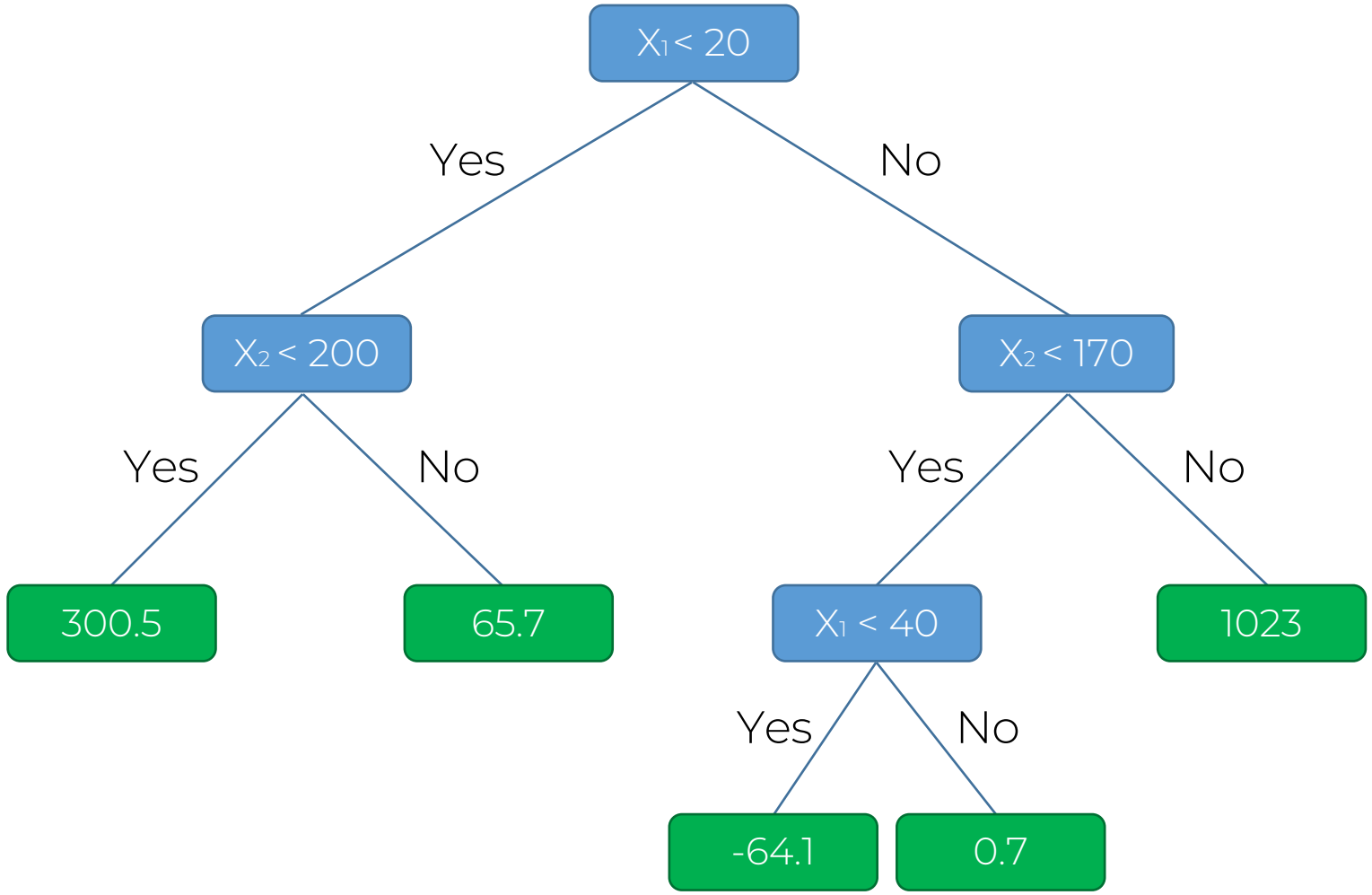
Decision Tree Intuition



Decision Tree Intuition



Decision Tree Intuition



Random Forest Intuition

Random Forest Intuition

Ensemble Learning

Random Forest Intuition

STEP 1: Pick at random K data points from the Training set.

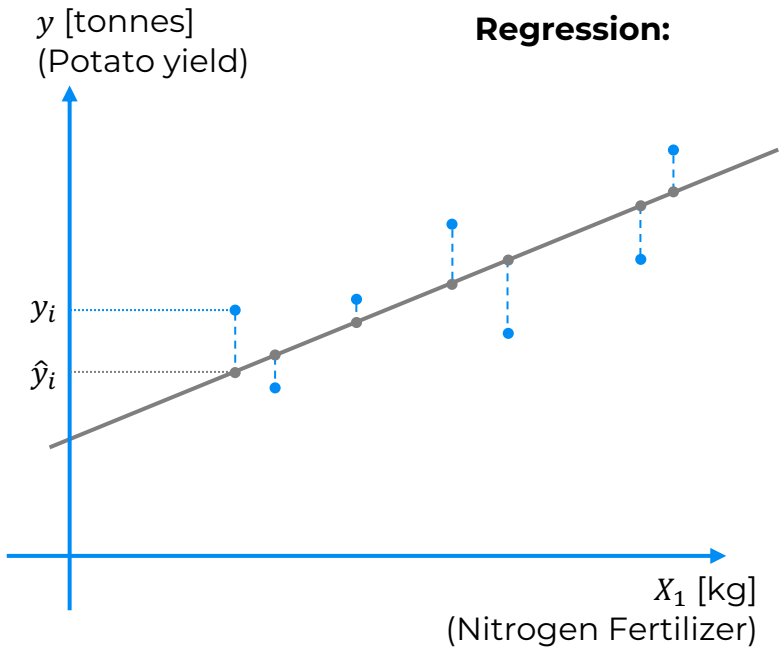
STEP 2: Build the Decision Tree associated to these K data points.

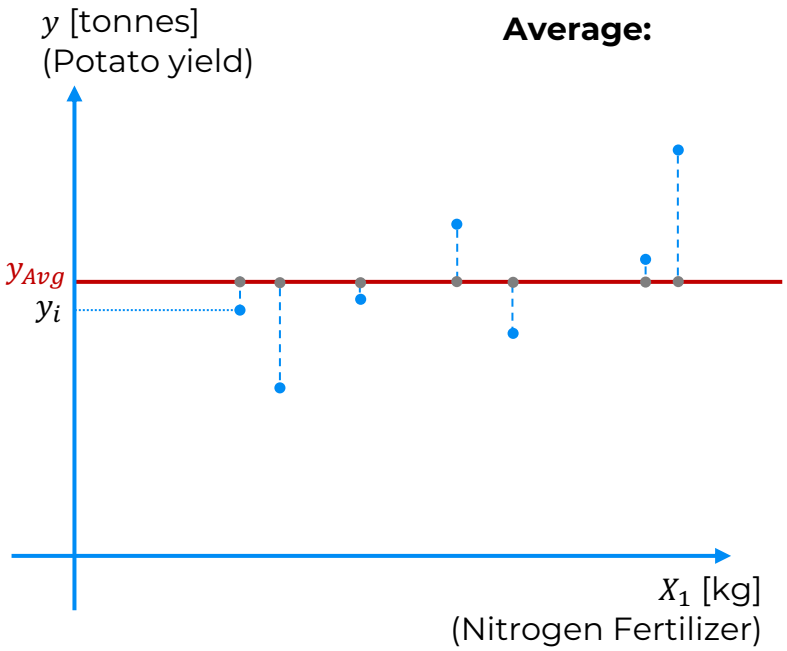
STEP 3: Choose the number Ntree of trees you want to build and repeat STEPS 1 & 2

STEP 4: For a new data point, make each one of your Ntree trees predict the value of Y to for the data point in question, and assign the new data point the average across all of the predicted Y values.

R Squared

R Squared





$$SS_{res} = SUM(y_i - \hat{y}_i)^2$$

$$SS_{tot} = SUM(y_i - y_{avg})^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Rule of thumb (for our tutorials)*:
1.0 = Perfect fit (suspicious)
~0.9 = Very good
<0.7 = Not great
<0.4 = Terrible
<0 = Model makes no sense for this data

**This is highly dependent on the context*

Adjusted R Squared

Adjusted R Squared

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

R^2 – Goodness of fit
(greater is better)

Problem:

$\hat{y} = b_0 + b_1X_1 + b_2X_2$ ← $+ b_3X_3$

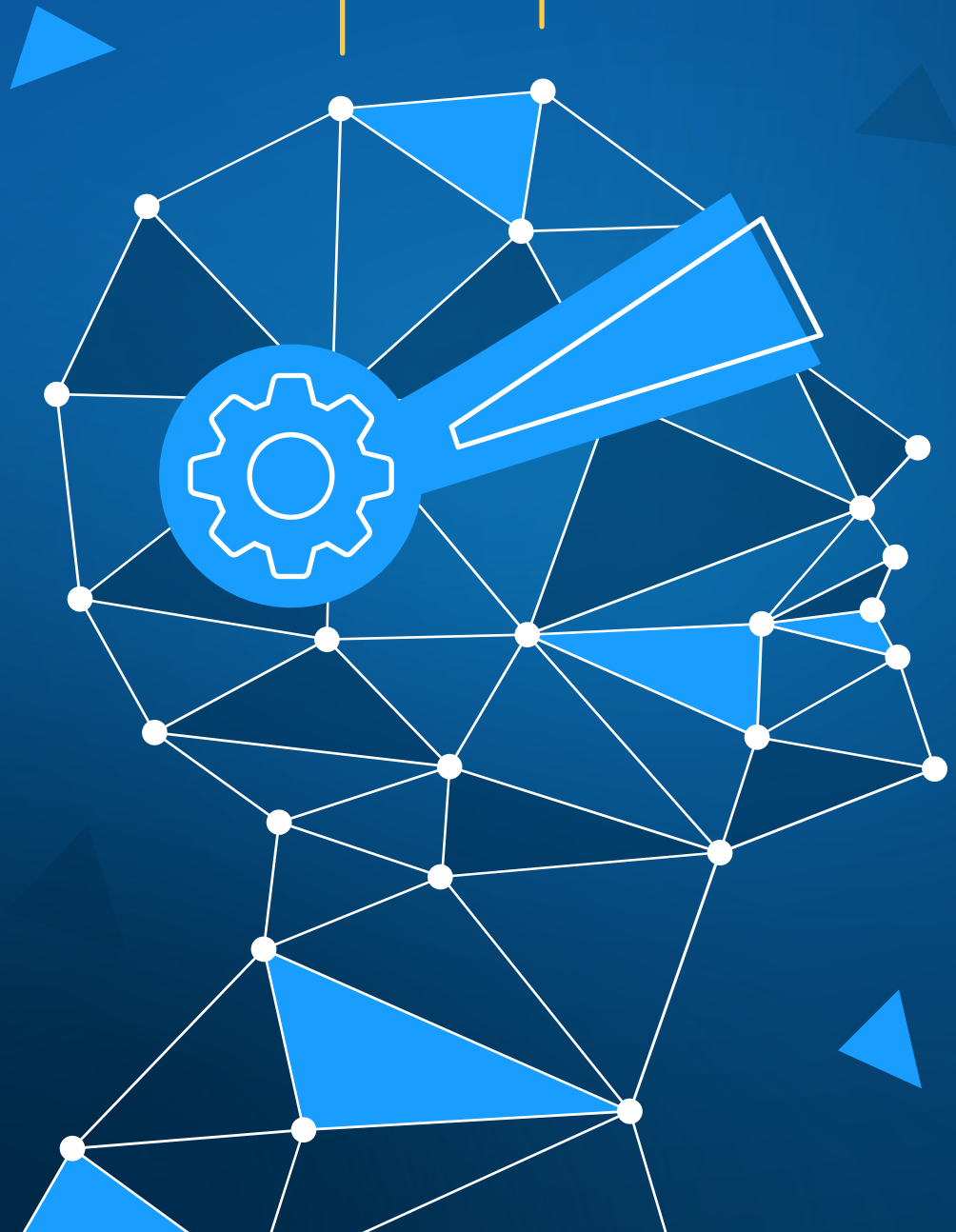
$$SS_{res} = SUM(y_i - \hat{y}_i)^2$$

SS_{tot} doesn't change
SS_{res} will decrease or stay the same *(This is because of Ordinary Least Squares: SS_{res}-> Min)*

Solution:

$$Adj\ R^2 = 1 - (1 - R^2) \times \frac{n - 1}{n - k - 1}$$

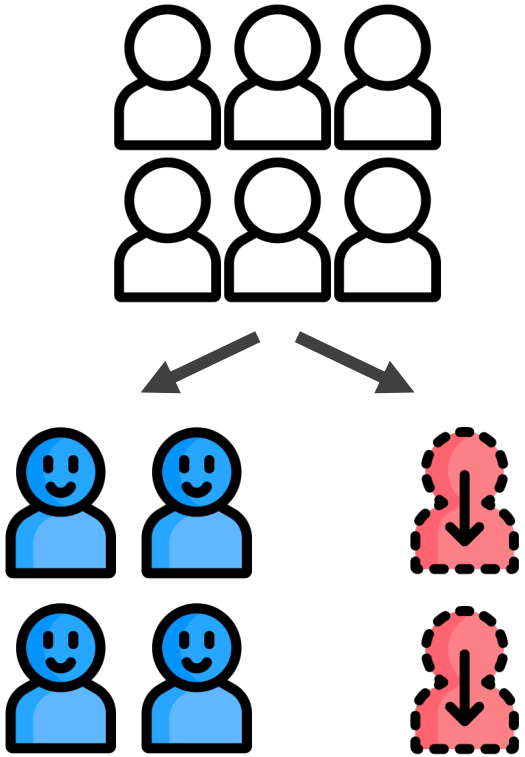
k – number of independent variables
n – sample size



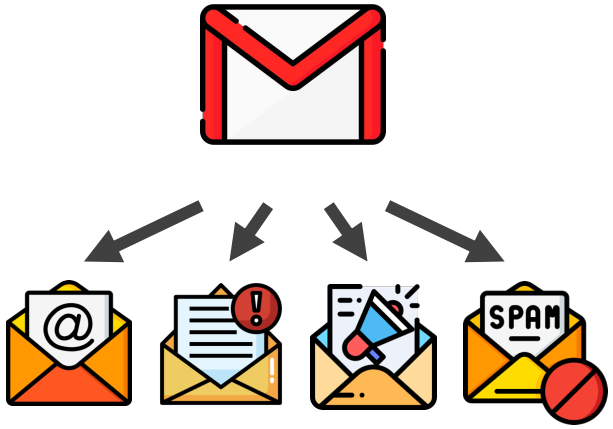
Classification

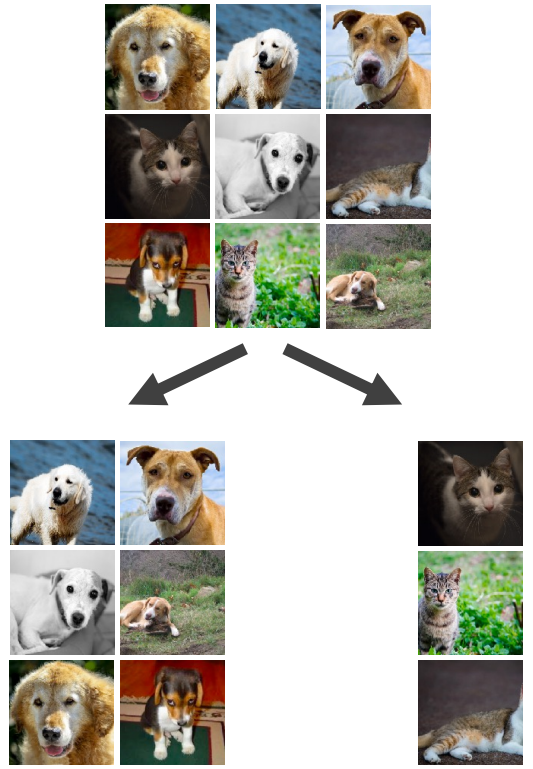
What is Classification?

Classification: a Machine Learning technique to identify the category of new observations based on training data.



Likely to stay *Likely to leave*





Dogs *Cats*

Logistic Regression

Logistic Regression

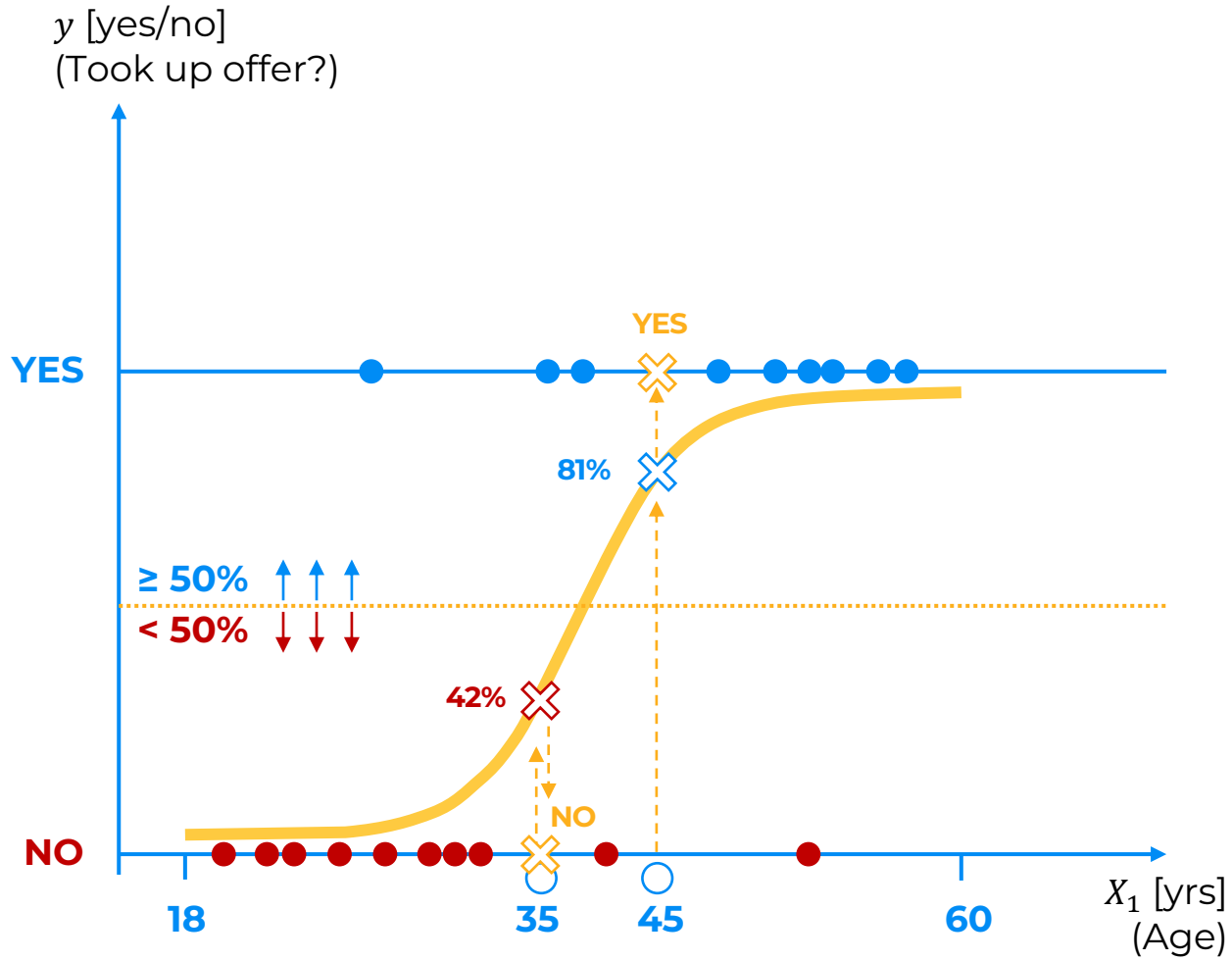
Logistic regression: predict a categorical dependent variable from a number of independent variables.

Will purchase health insurance: Yes / No

$\sim$

Age

$$\ln \frac{p}{1-p} = b_0 + b_1 X_1$$



Logistic Regression

Will purchase health insurance: Yes / No

$\sim$

Age

Income

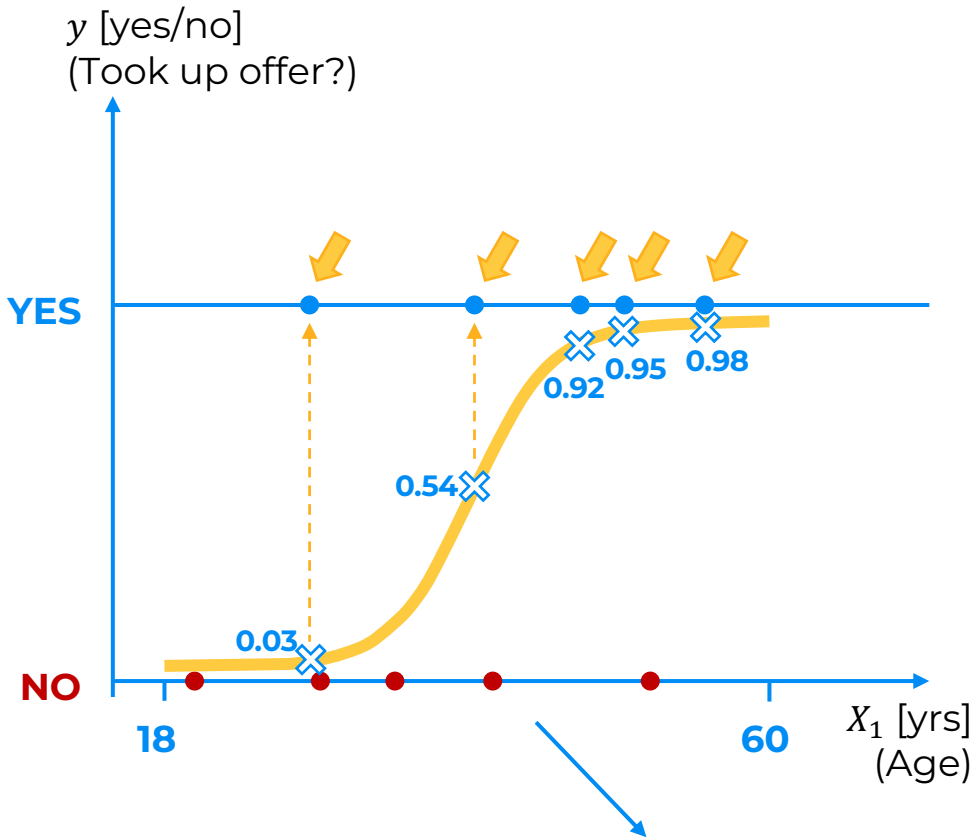
Level of Education

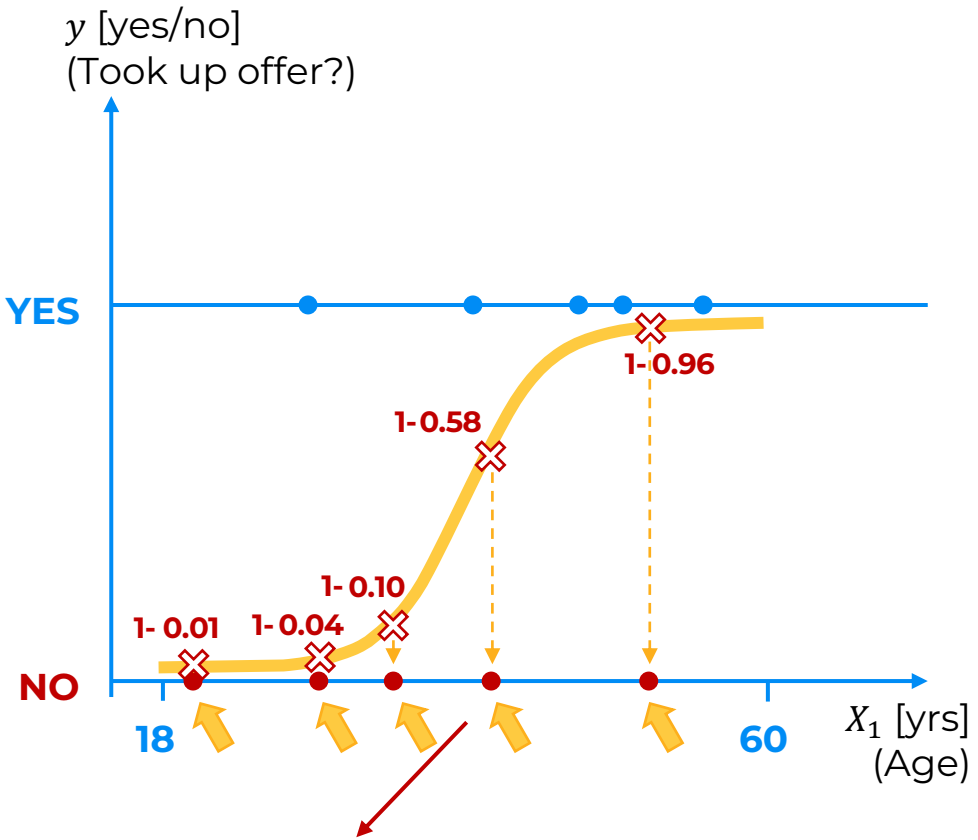
Family or Single

$$\ln \frac{p}{1-p} = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + b_4X_4$$

Maximum Likelihood

Maximum Likelihood

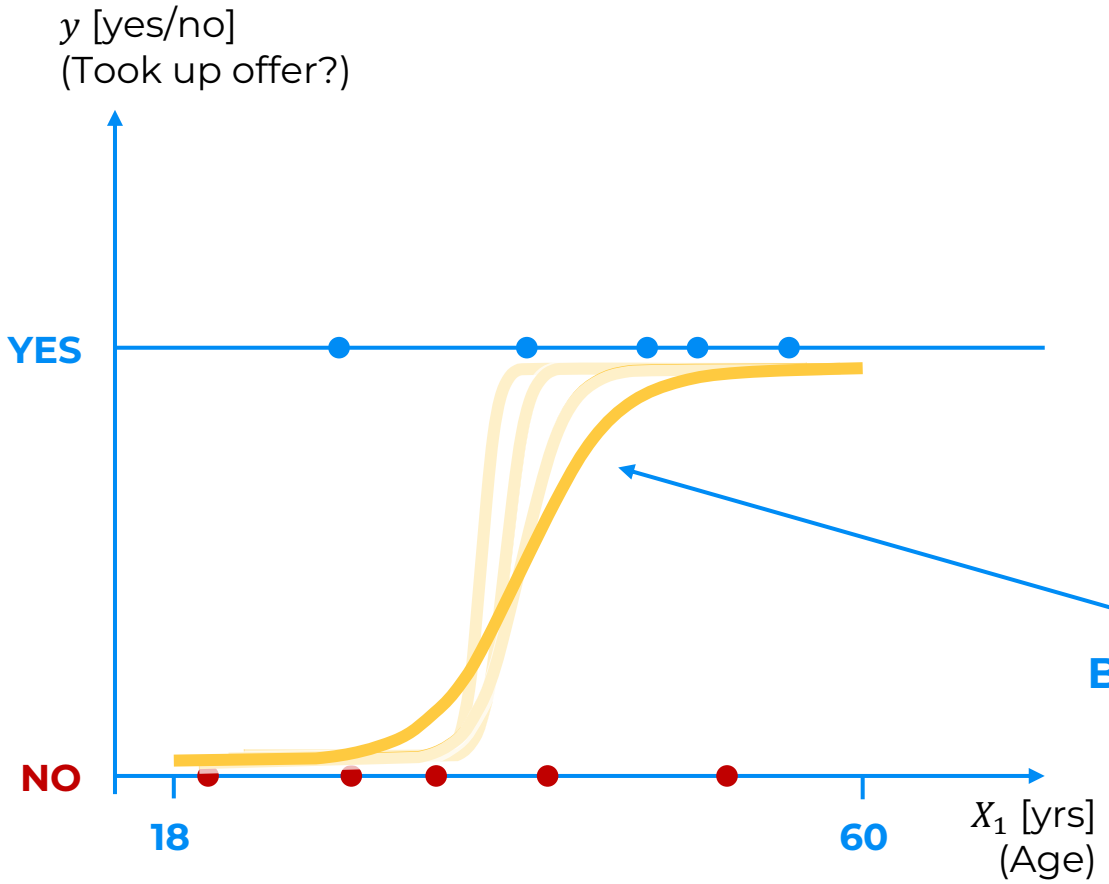




Likelihood = 0.03 x 0.54 x 0.92 x 0.95 x 0.98 x (1 – 0.01) x (1 – 0.04) x (1 – 0.10) x (1 – 0.58) x (1 – 0.96)

Likelihood = **0.00019939**

Maximum Likelihood



Likelihood = 0.00007418

Likelihood = 0.00012845

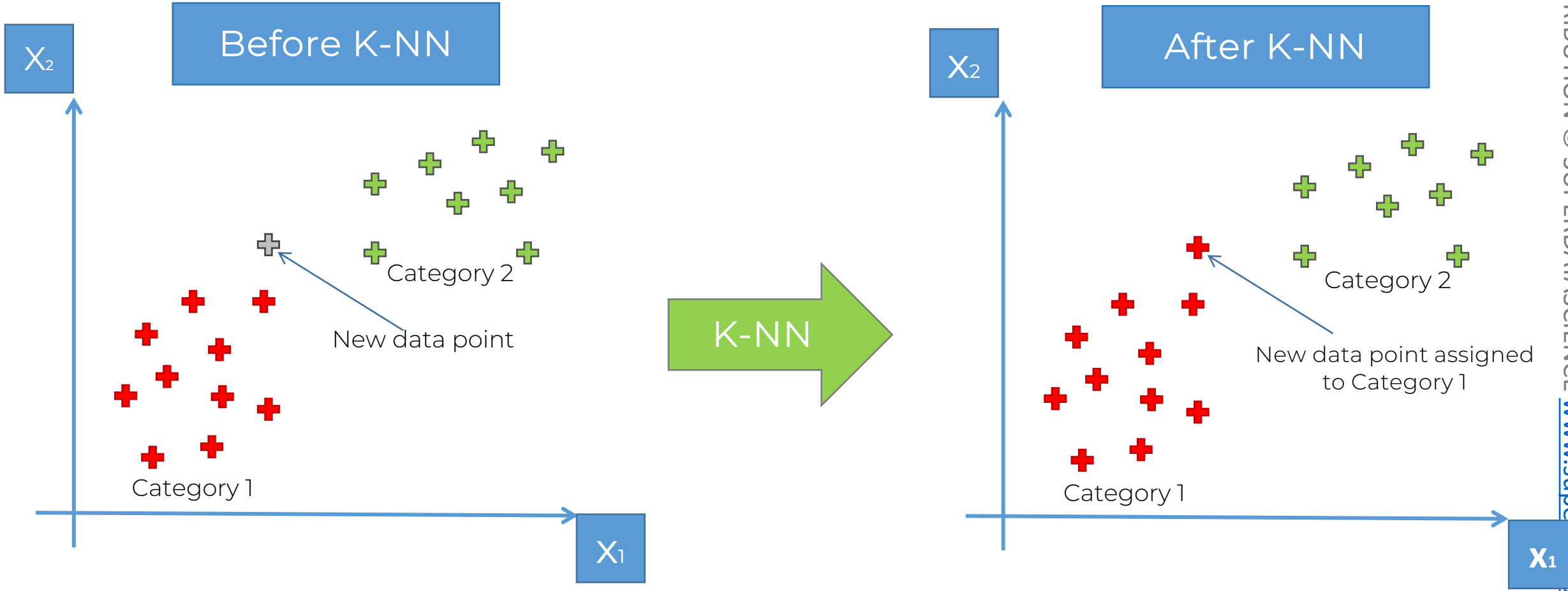
Likelihood = 0.00016553

Likelihood = 0.00019939

Best Curve <= Maximum Likelihood

K-NN Intuition

What K-NN does for you



How did it do that ?

STEP 1: Choose the number K of neighbors

STEP 2: Take the K nearest neighbors of the new data point, according to the Euclidean distance

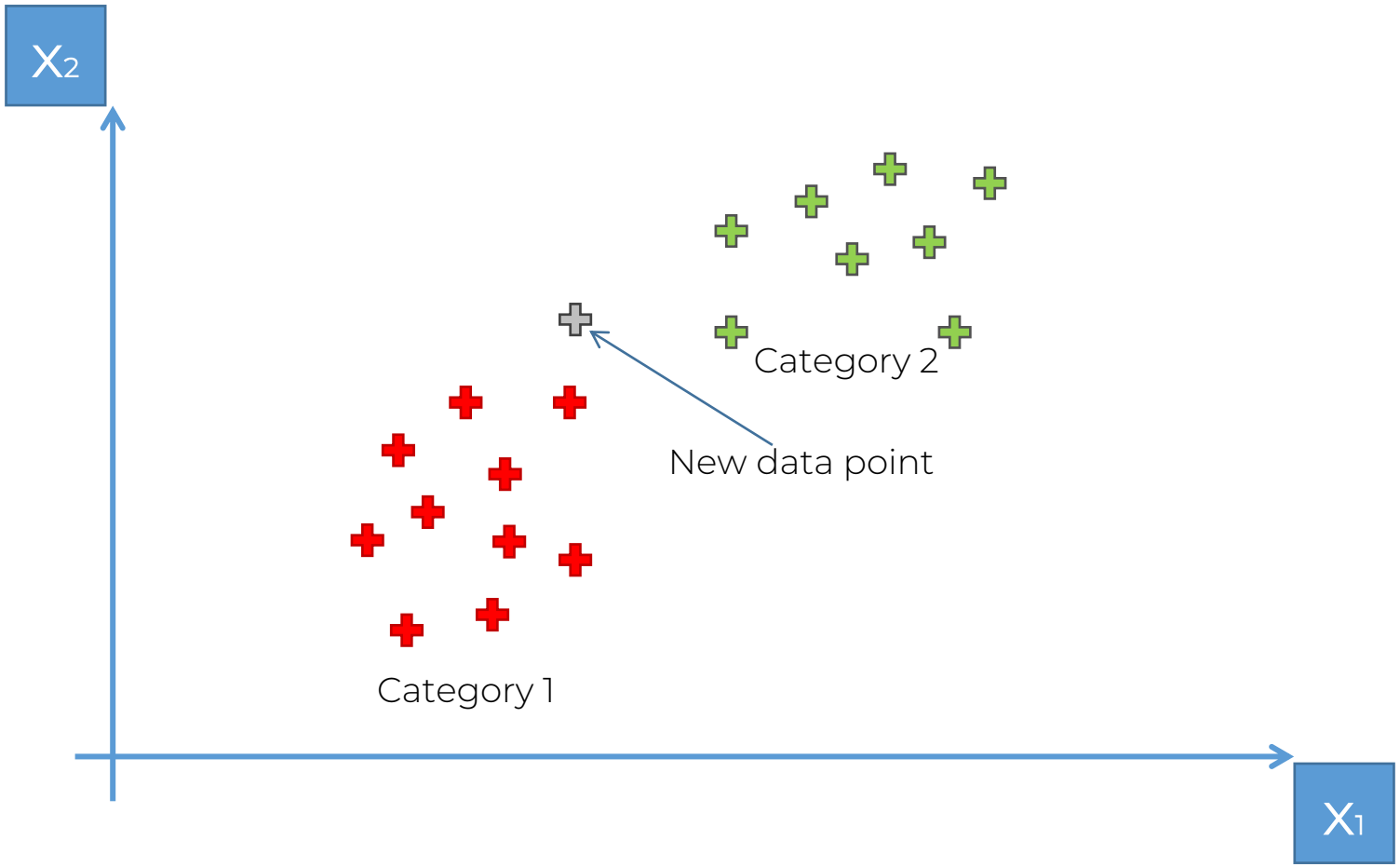
STEP 3: Among these K neighbors, count the number of data points in each category

STEP 4: Assign the new data point to the category where you counted the most neighbors

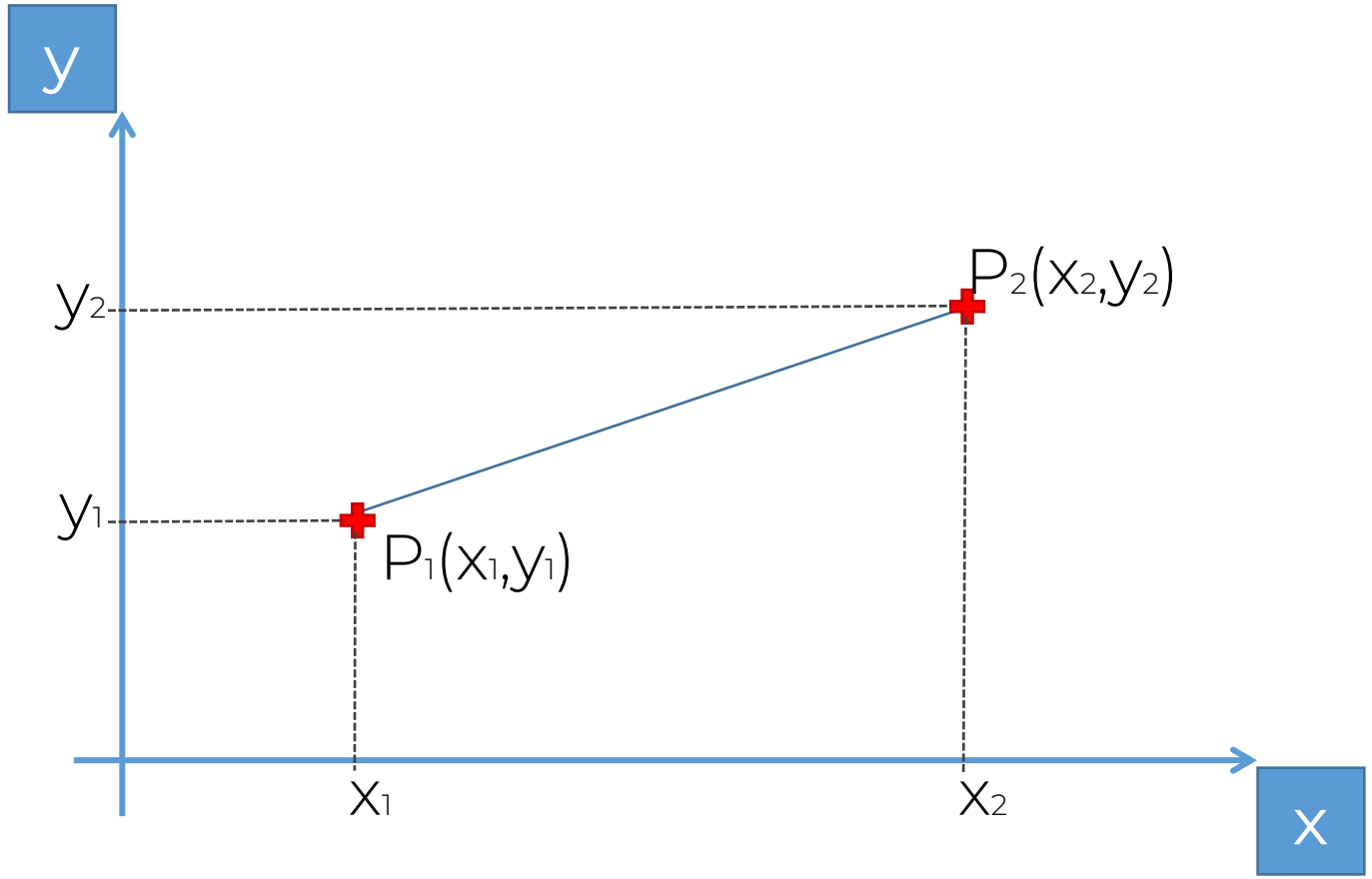
Your Model is Ready

K-NN algorithm

STEP 1: Choose the number K of neighbors: K = 5

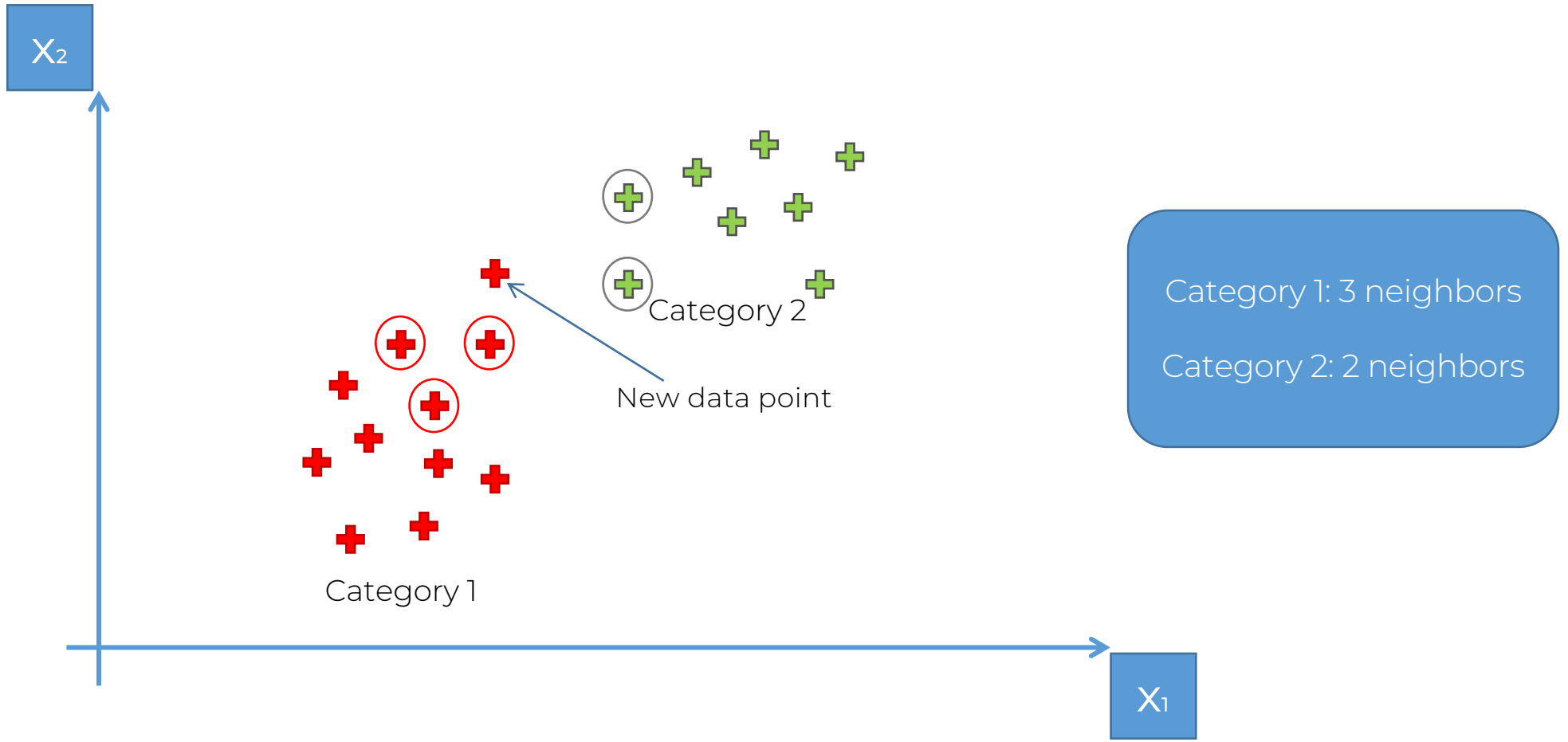


Euclidean Distance



$$\text{Euclidean Distance between } P_1 \text{ and } P_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

K-NN algorithm

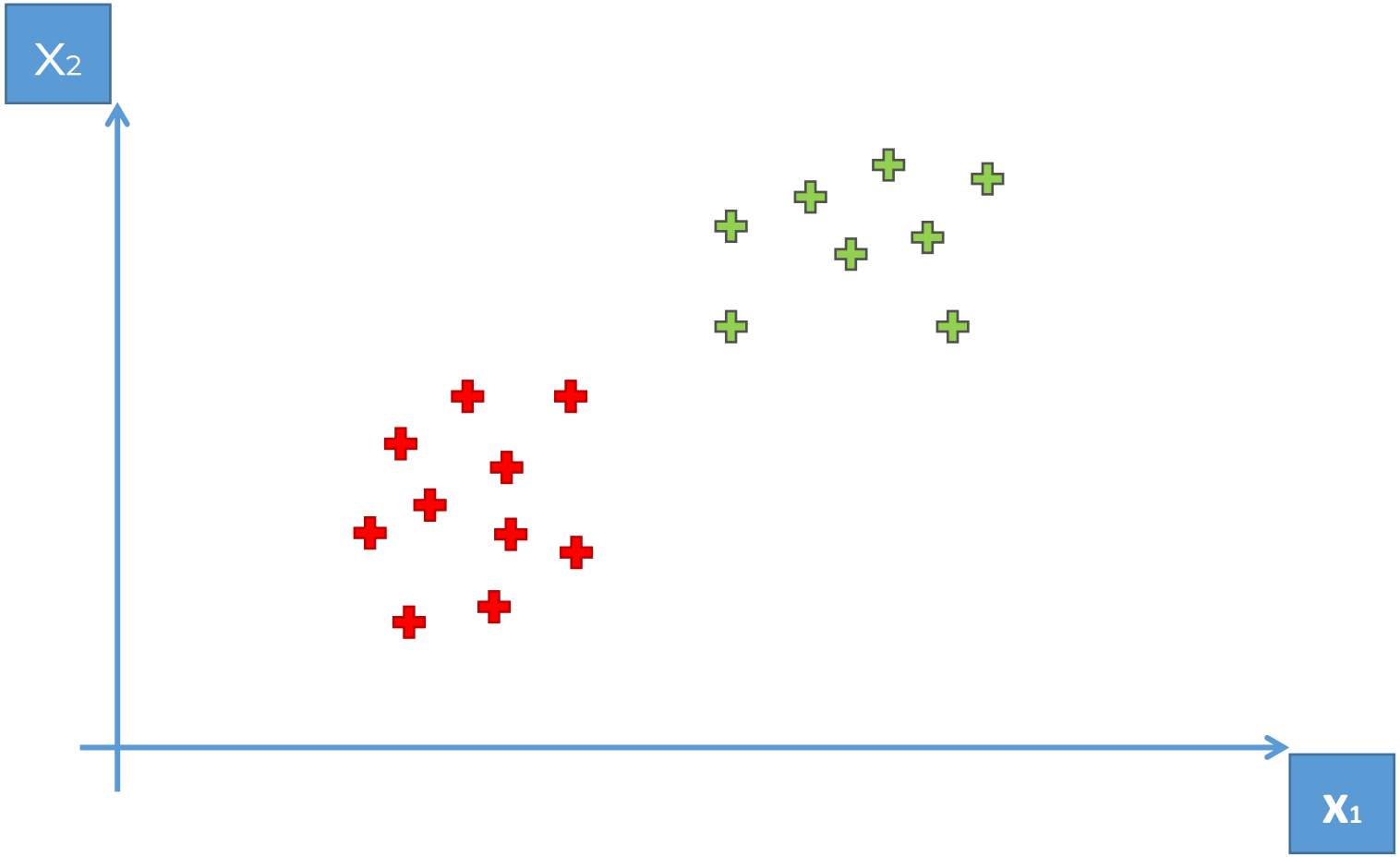


Category 1: 3 neighbors

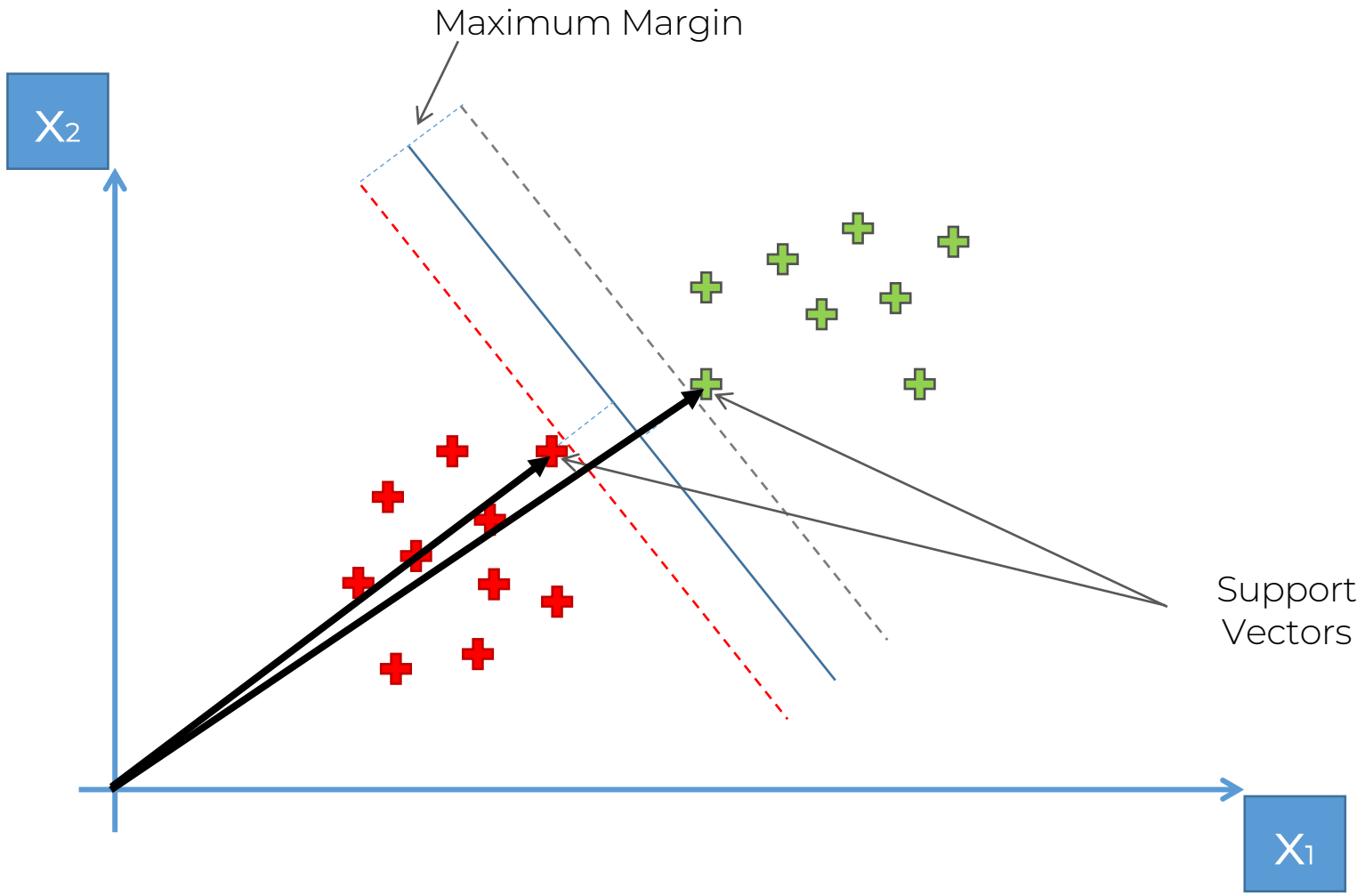
Category 2: 2 neighbors

SVM Intuition

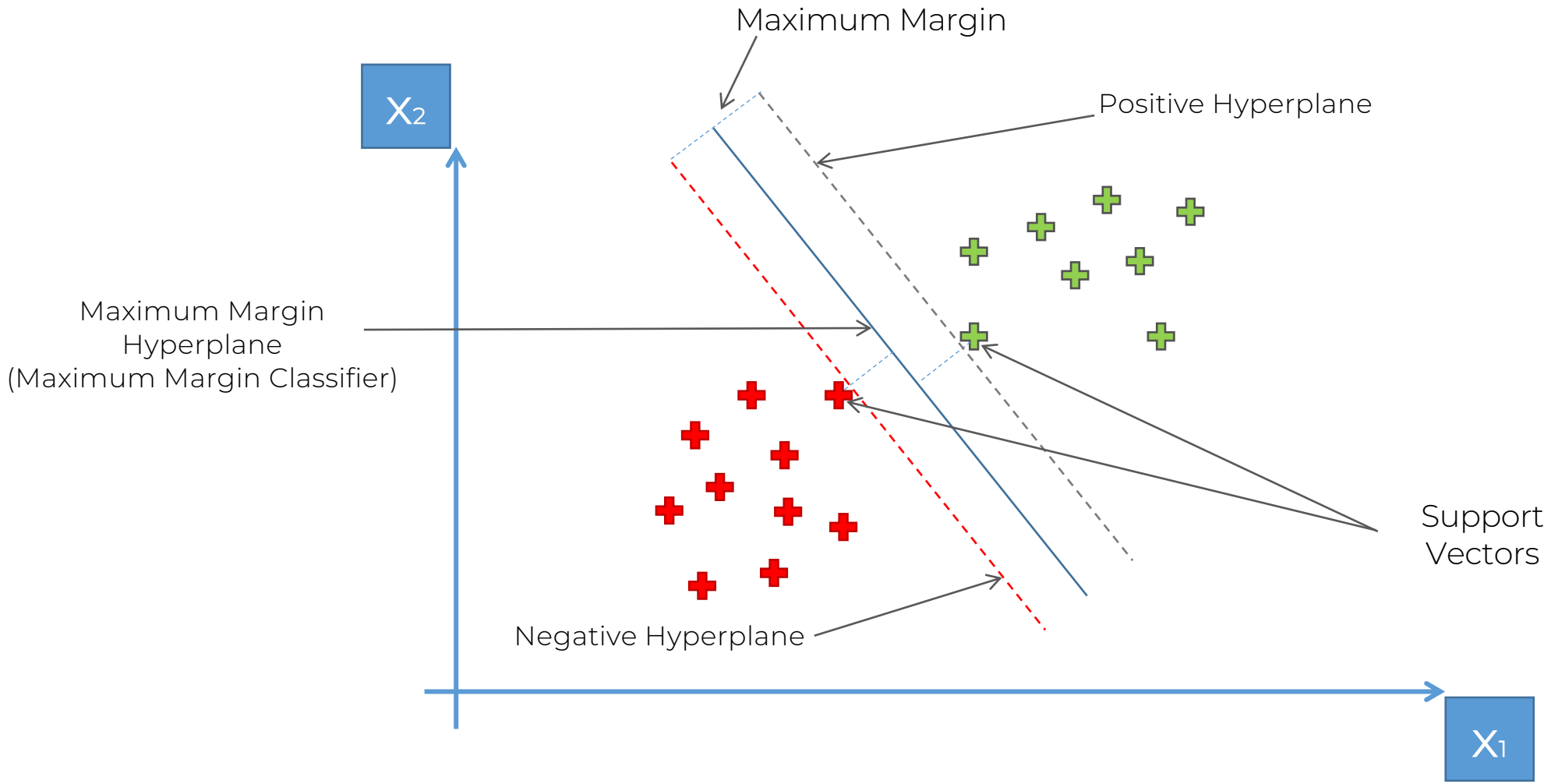
How to separate these points ?



Support Vectors

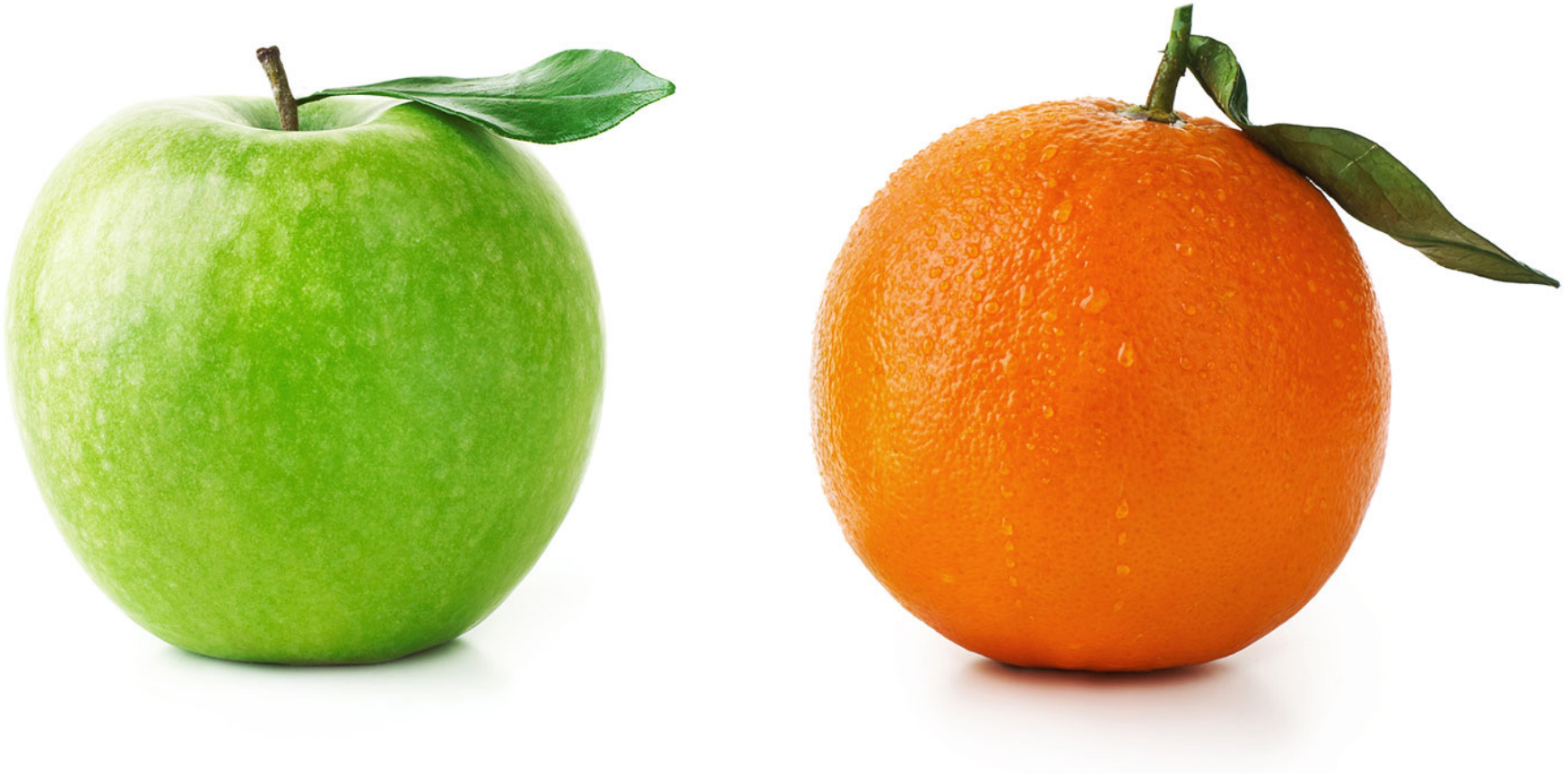


Hyperplanes

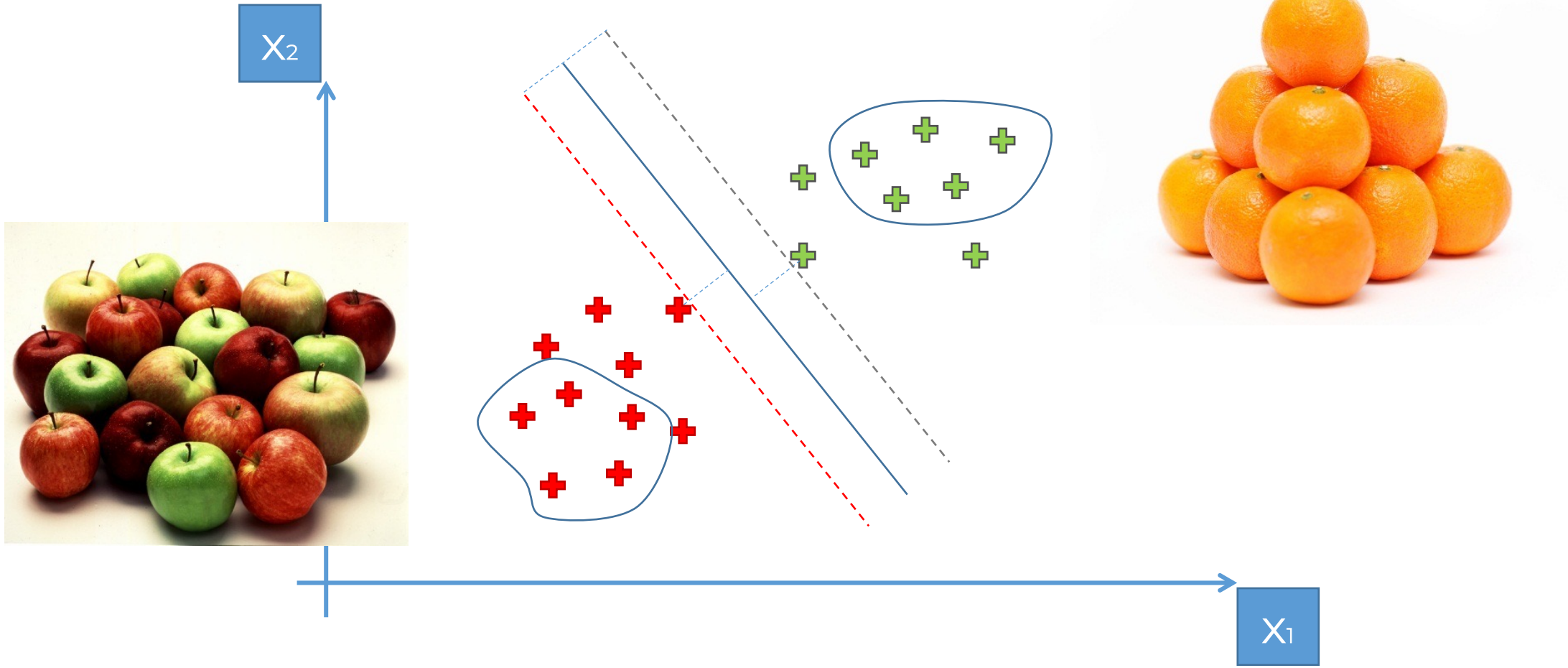


What's So Special About SVMs?

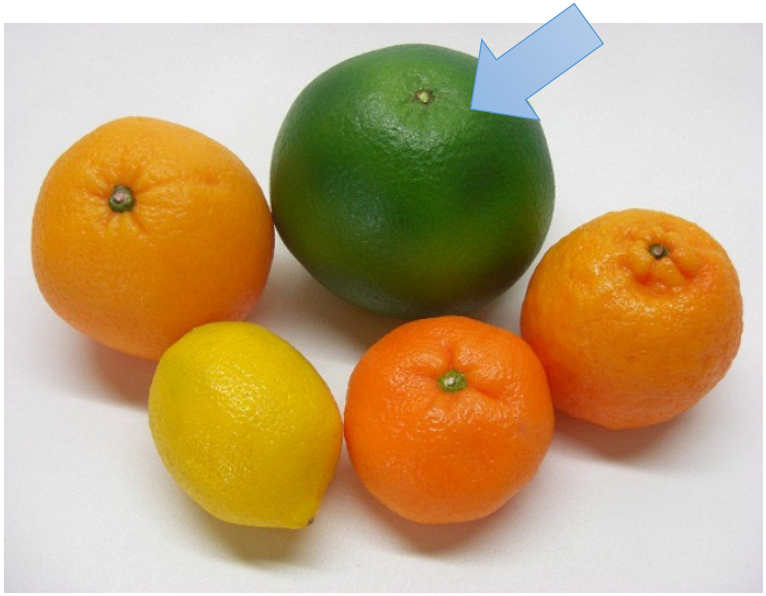
What's So Special About SVMs?

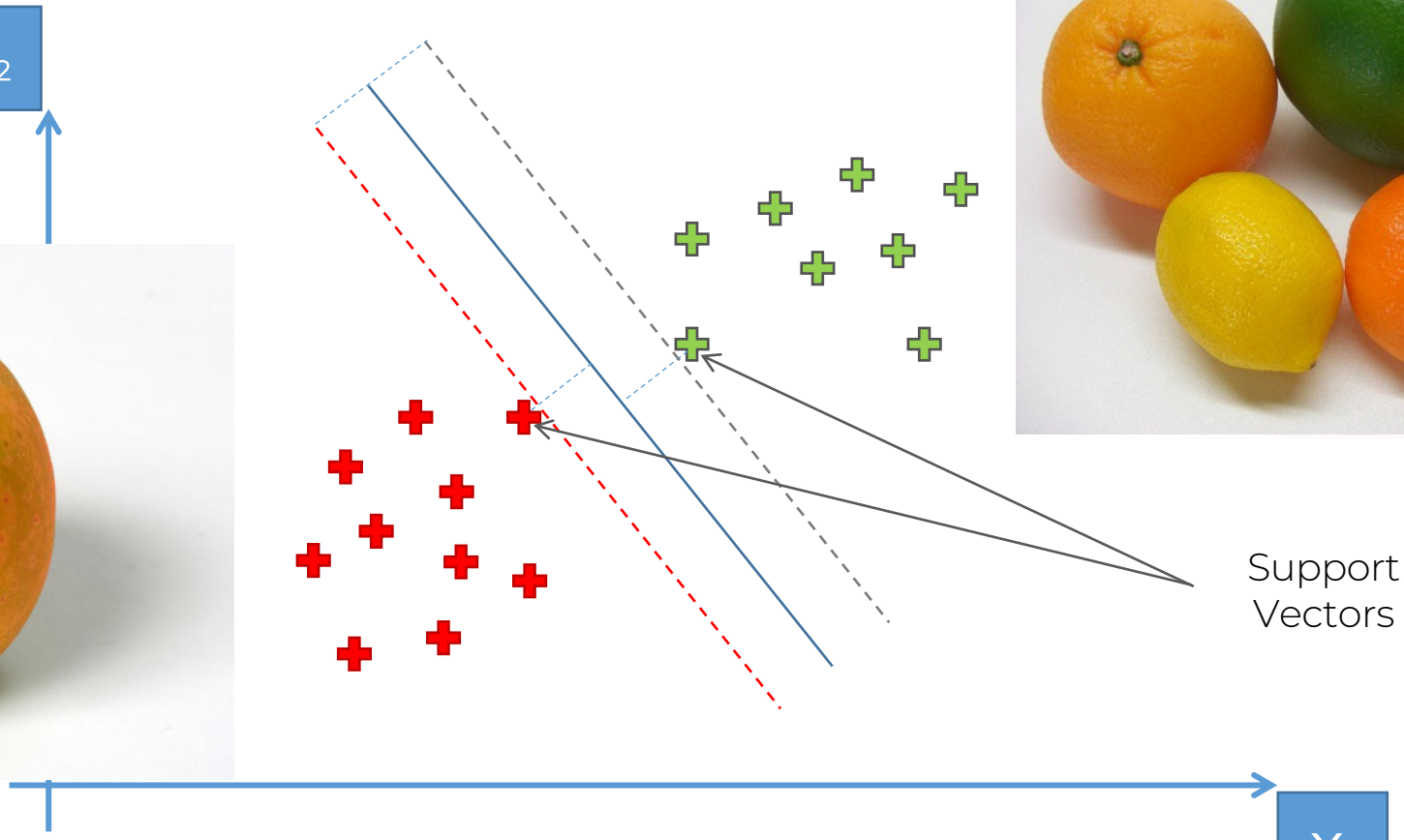


What's So Special About SVMs?



What's So Special About SVMs?

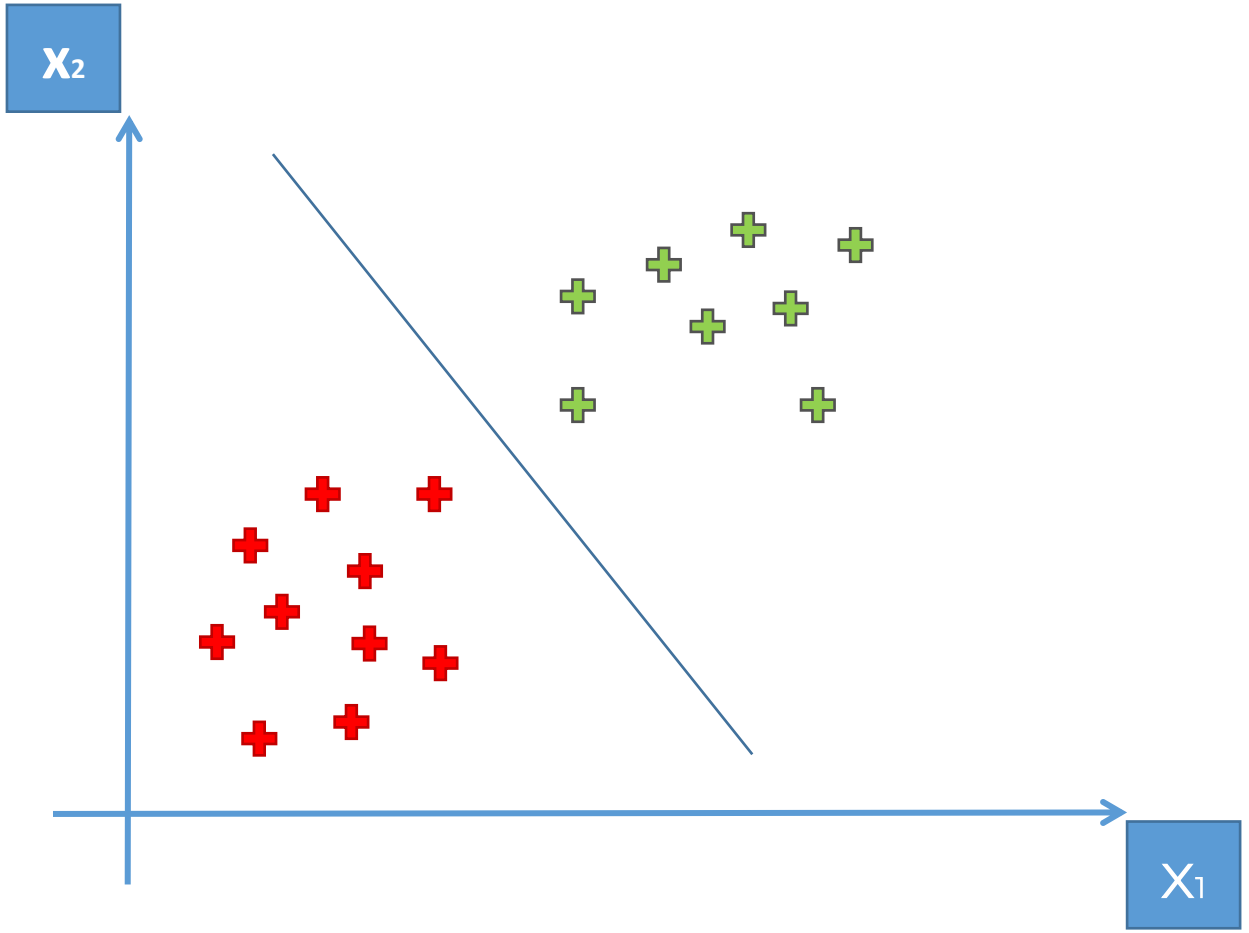




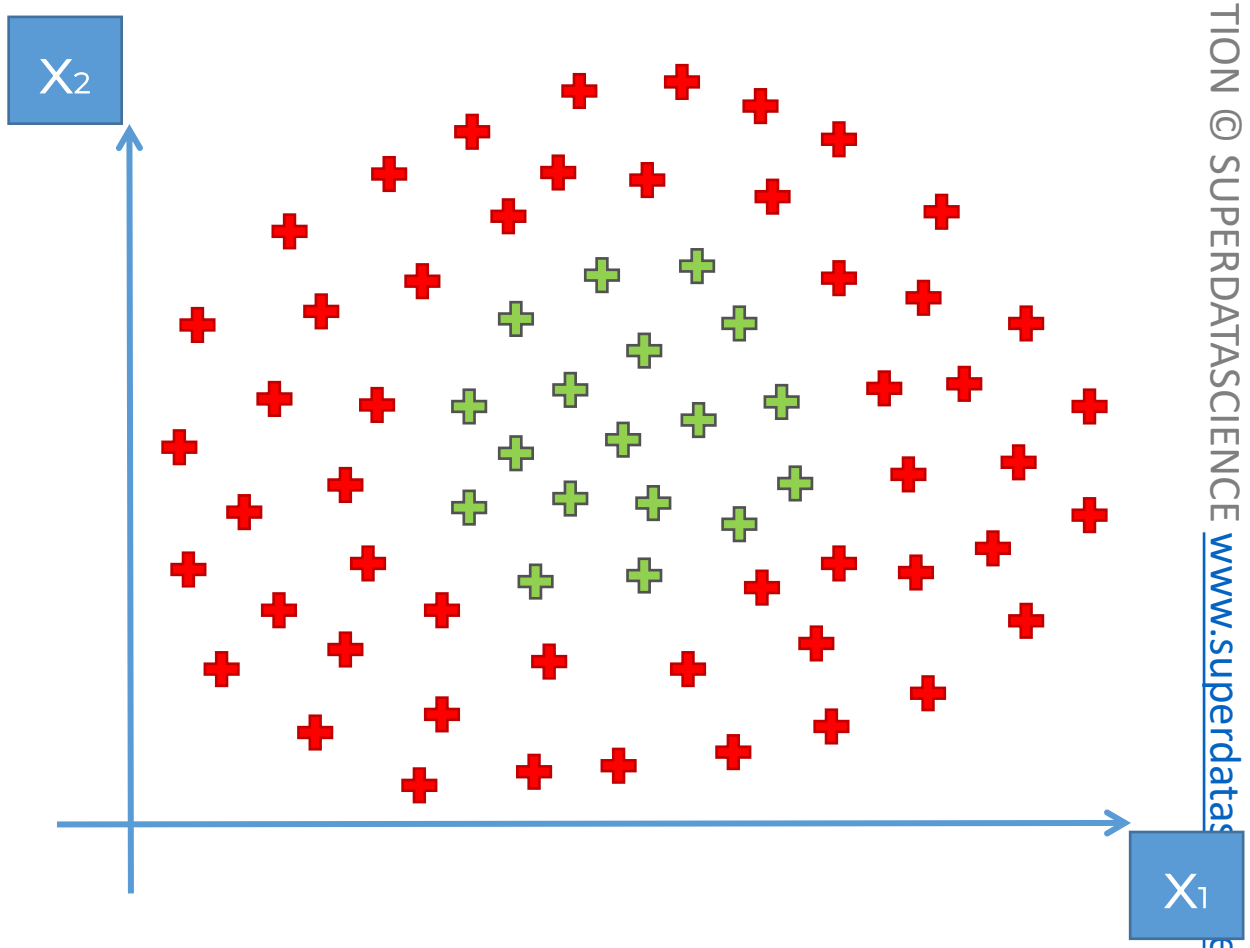
Kernel SVM Intuition

Linear Separability

Linearly Separable



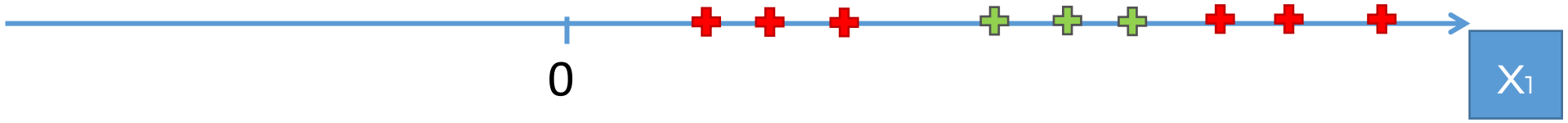
Not Linearly Separable



A Higher-Dimensional Space

Mapping to a Higher Dimension

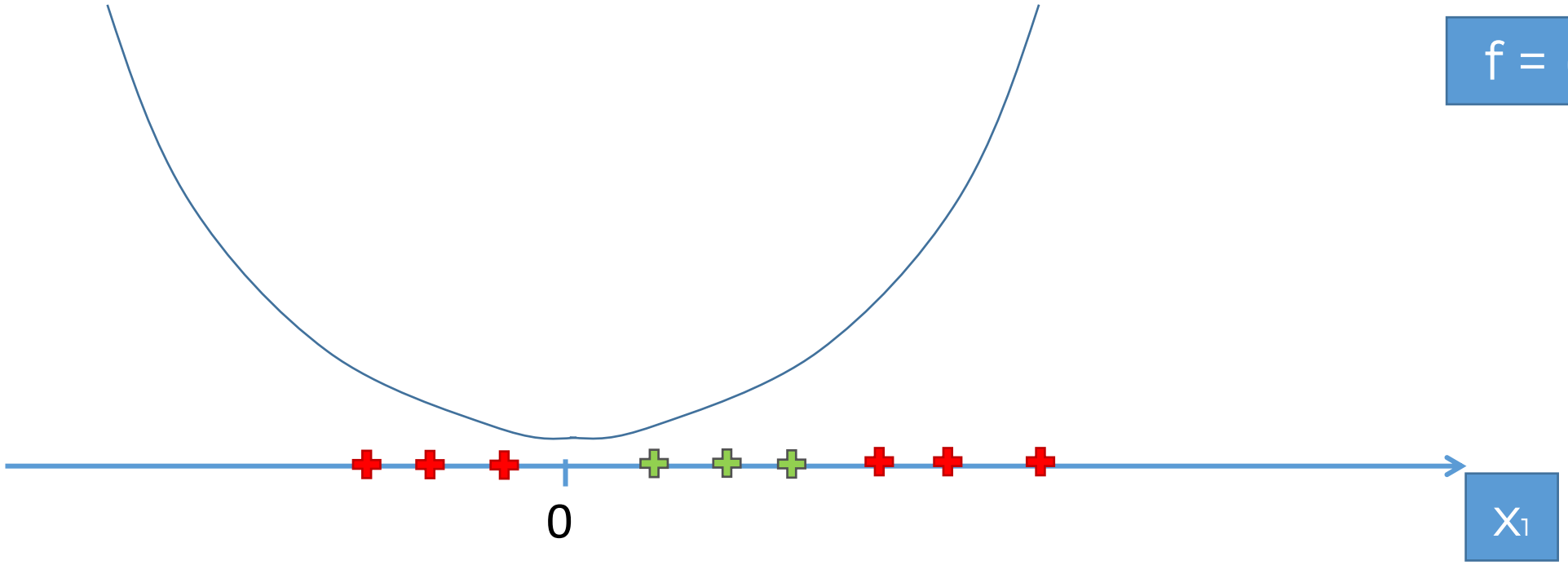
$f = x - 5$



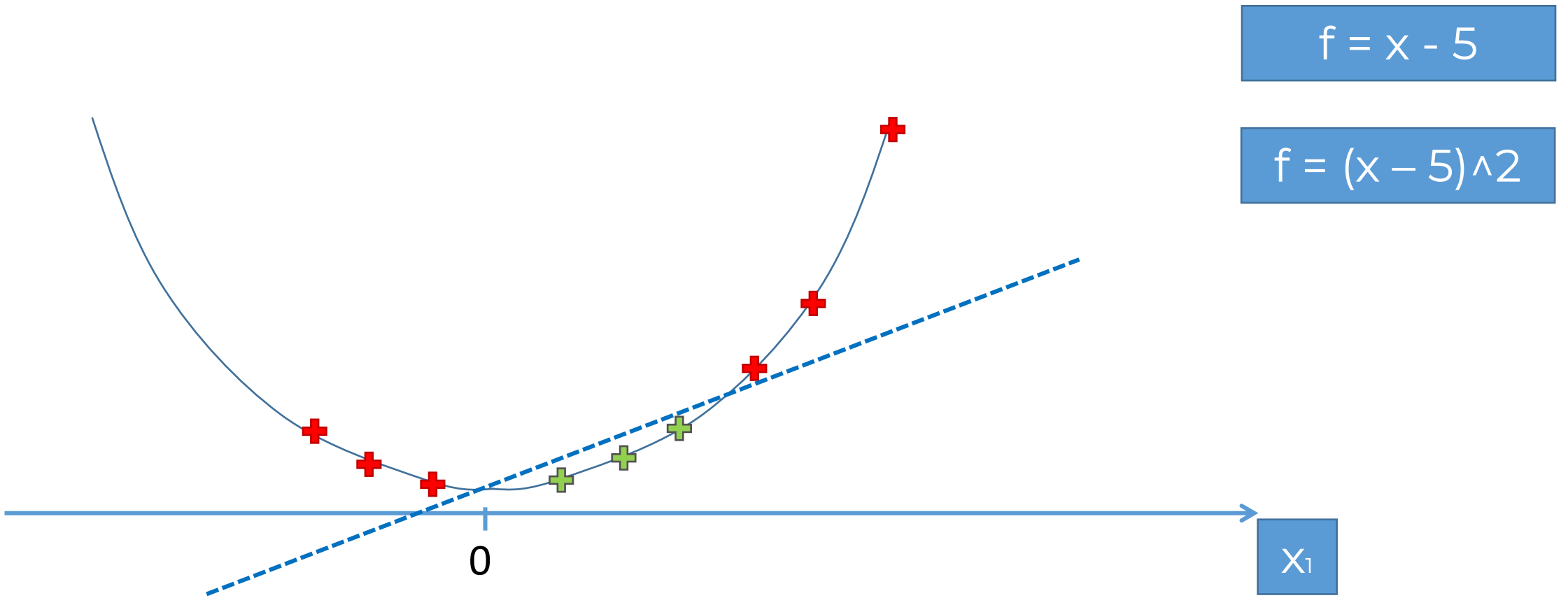
Mapping to a Higher Dimension

$f = x - 5$

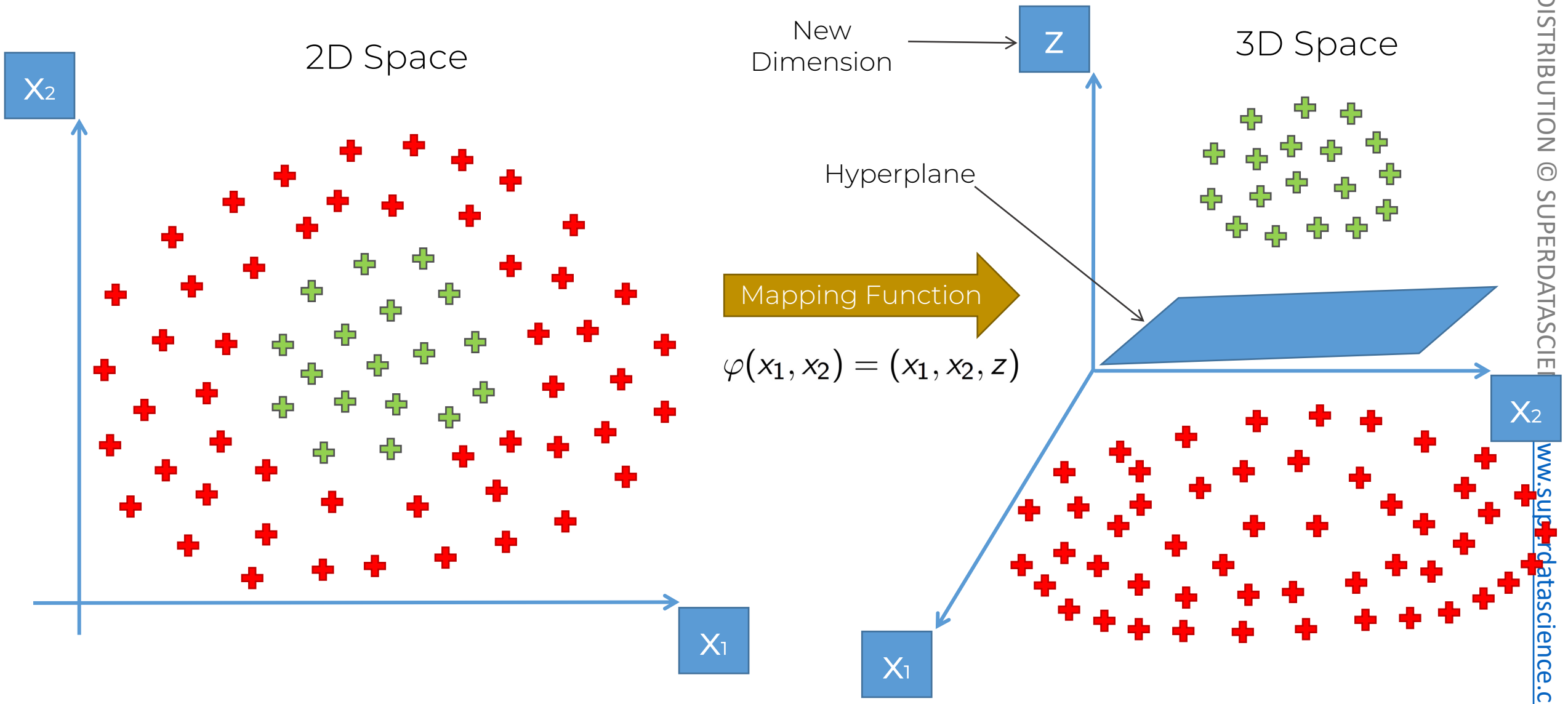
$f = (x - 5)^2$



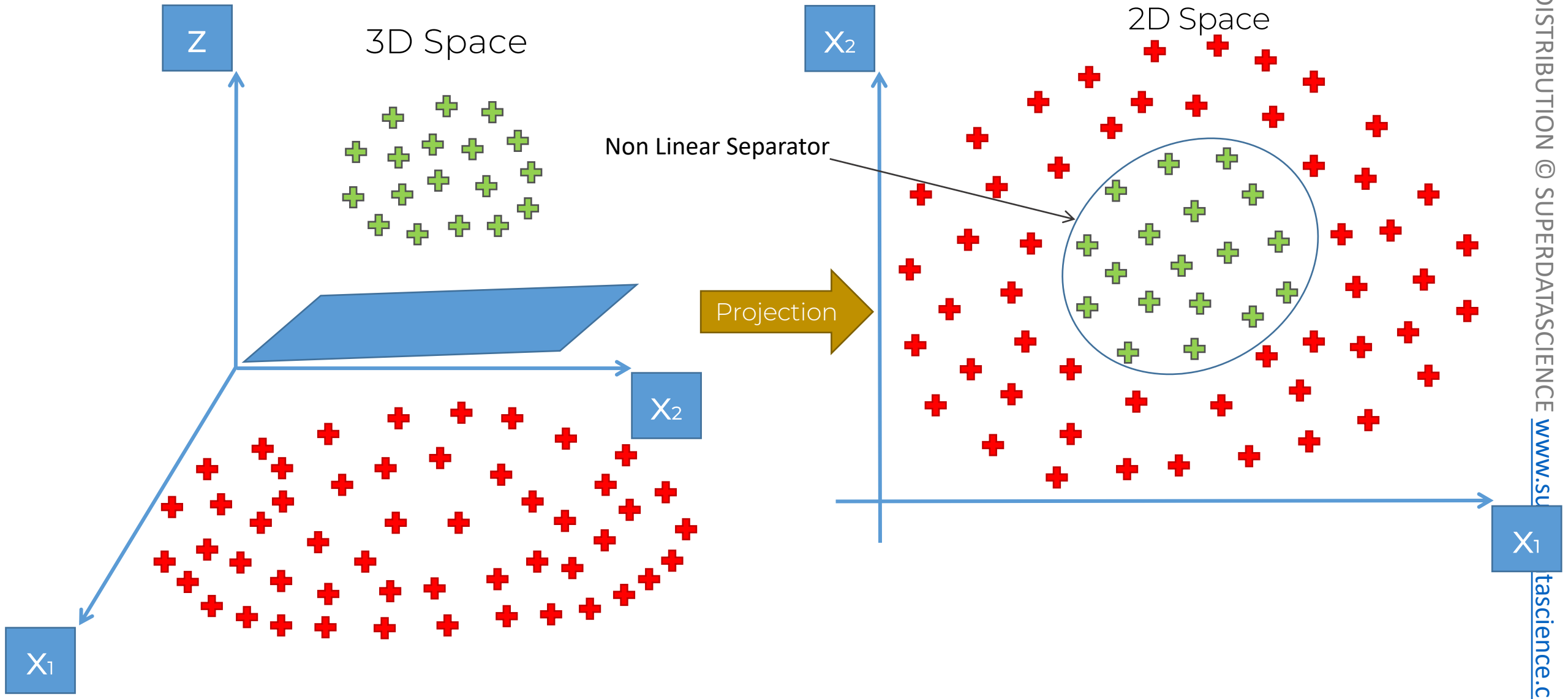
Mapping to a Higher Dimension



Mapping to a Higher Dimension



Projecting back to 2D Space



But there is a catch...

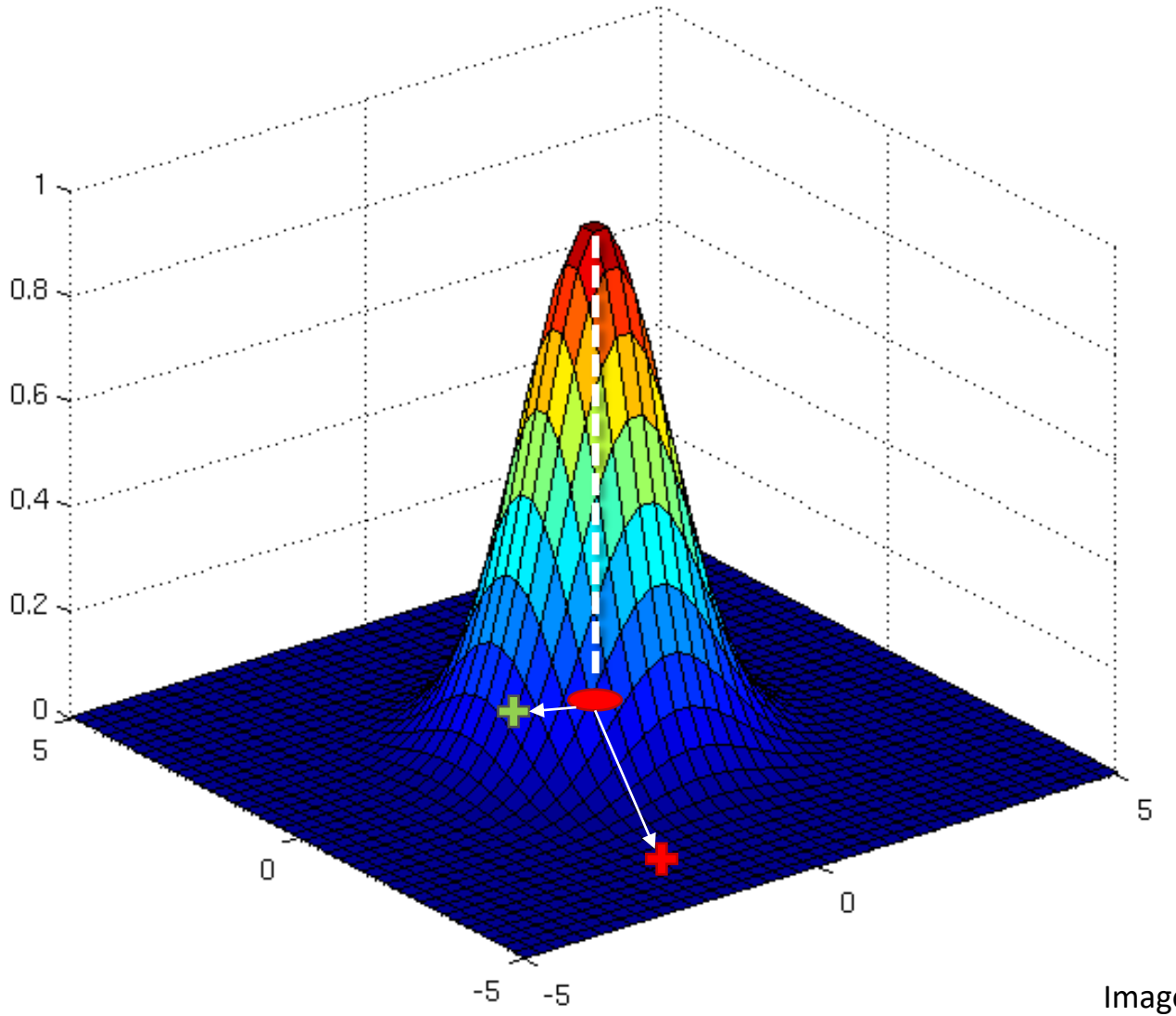
Mapping to a Higher Dimensional Space can be highly compute-intensive

The Kernel Trick

The Gaussian RBF Kernel

$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

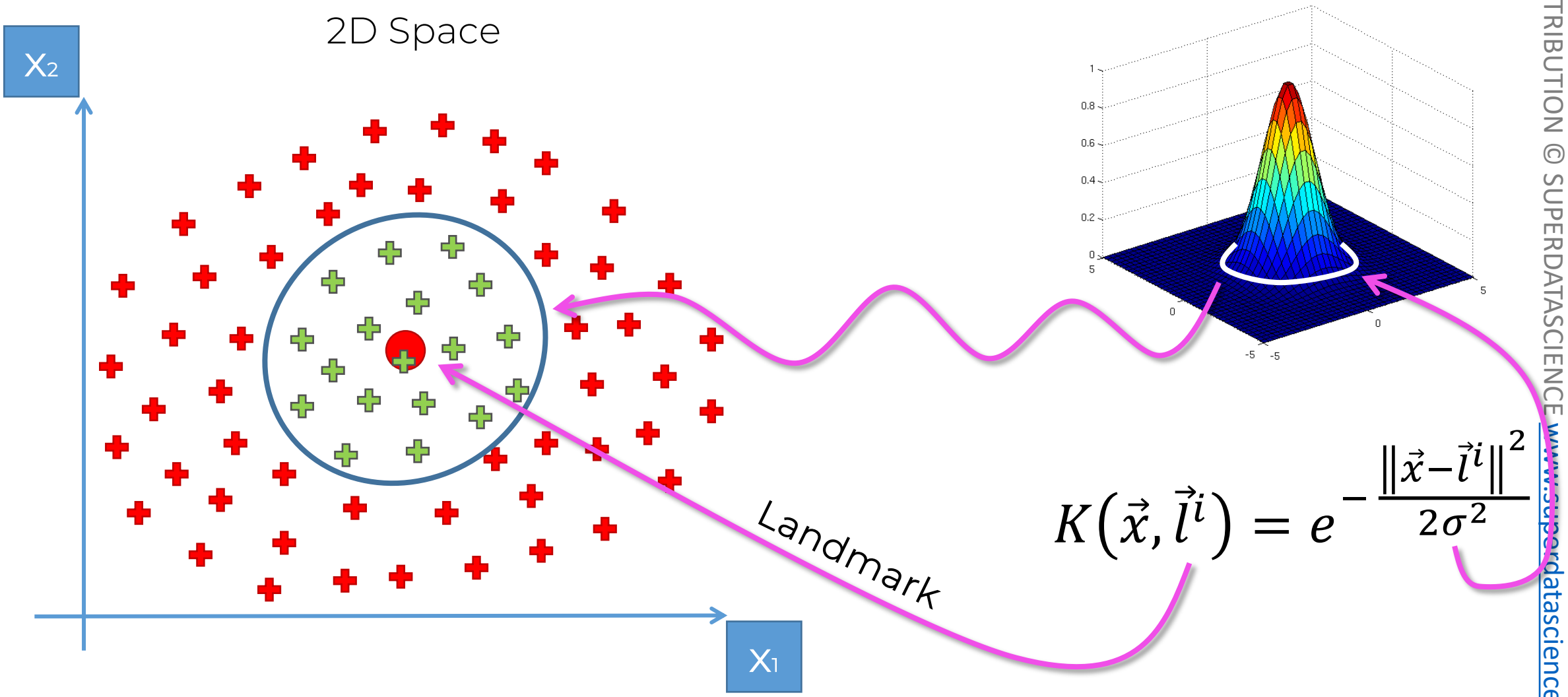
The Gaussian RBF Kernel



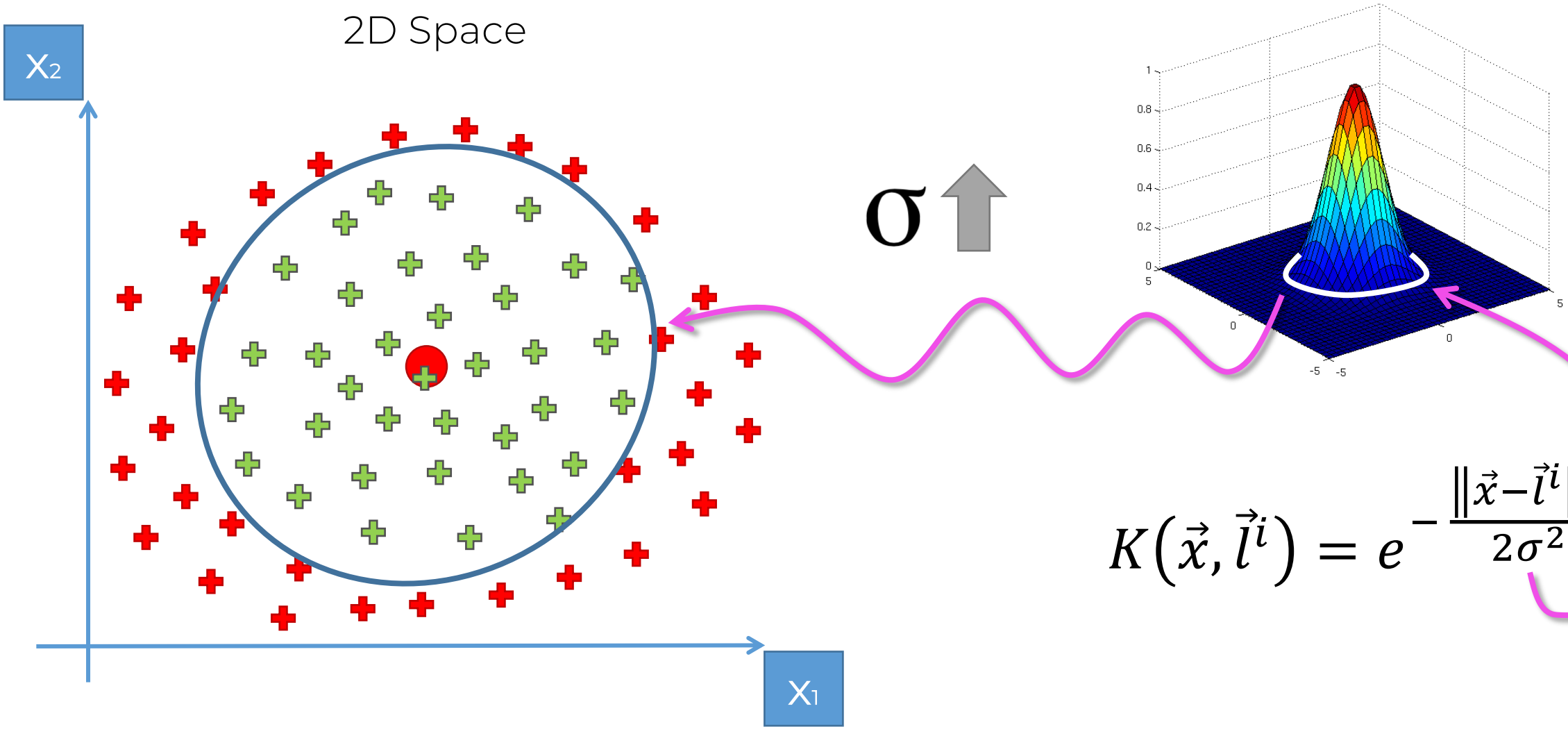
$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

Image source: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html

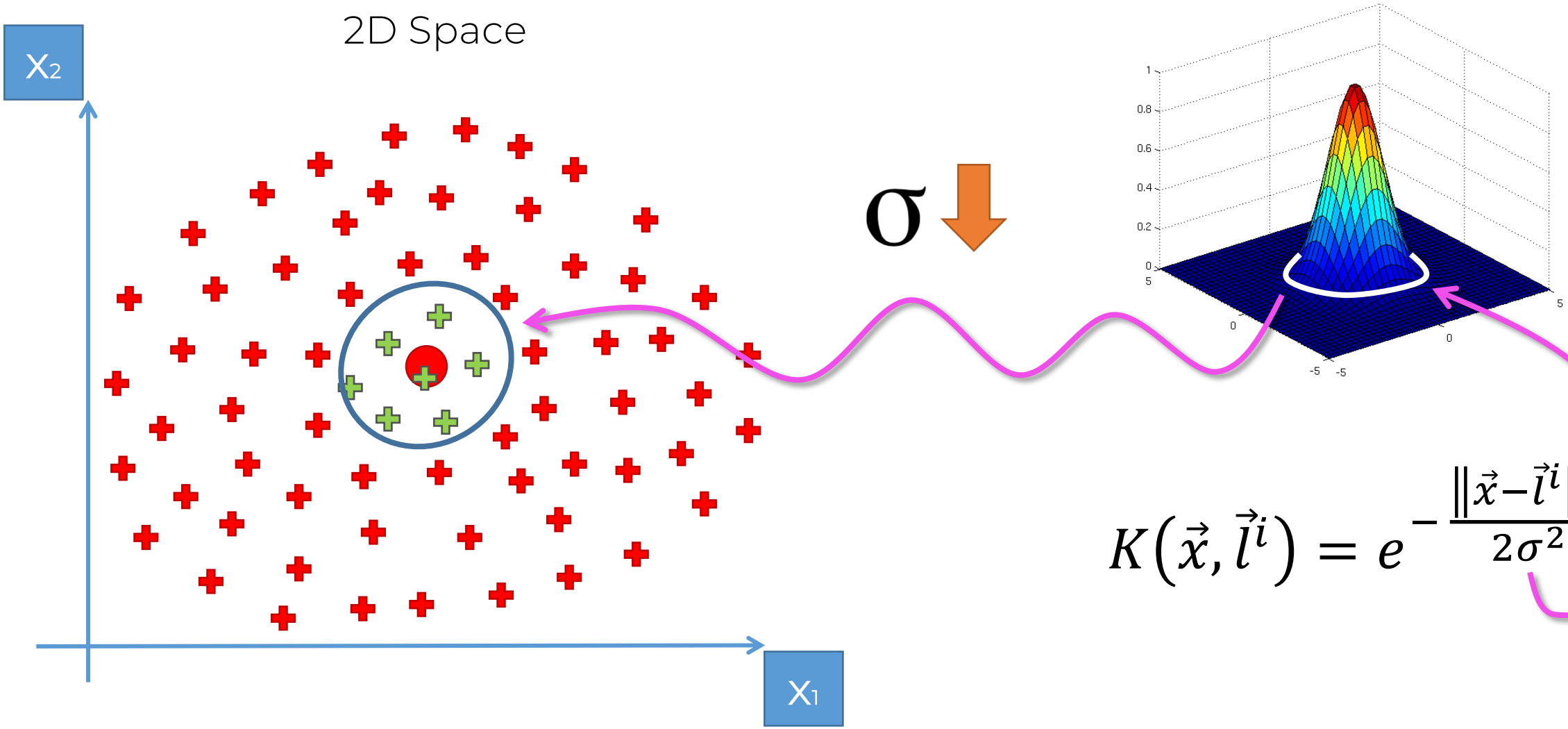
The Gaussian RBF Kernel



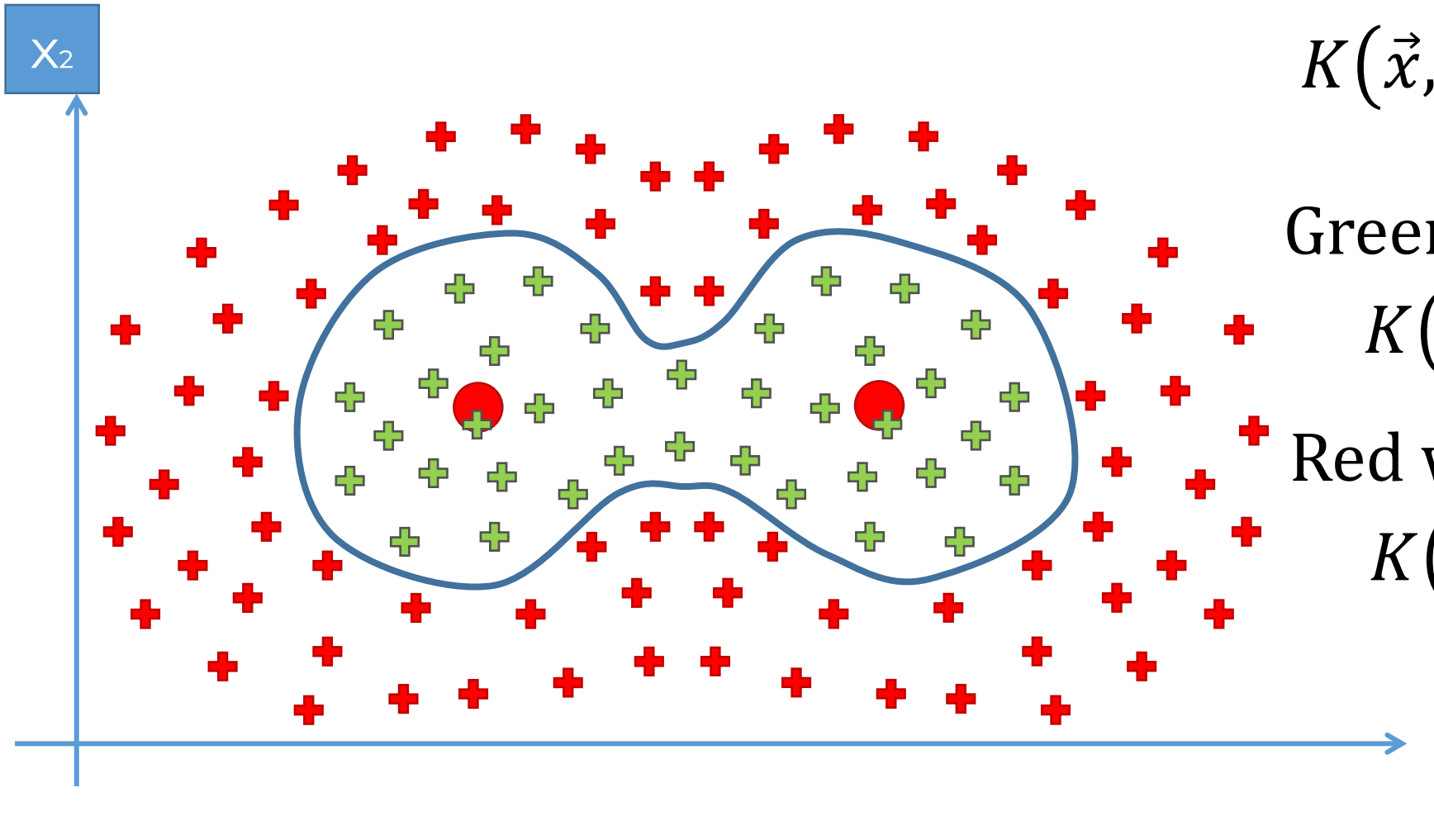
The Gaussian RBF Kernel



The Gaussian RBF Kernel



The Gaussian RBF Kernel



$$K(\vec{x}, \vec{l}^1) + K(\vec{x}, \vec{l}^2)$$

(Simplified Formula)

Green when:

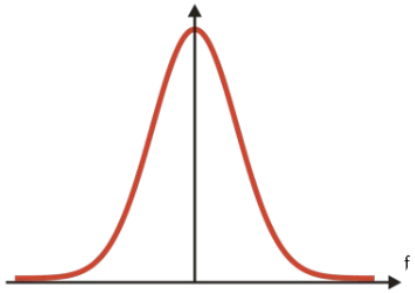
$$K(\vec{x}, \vec{l}^1) + K(\vec{x}, \vec{l}^2) >$$

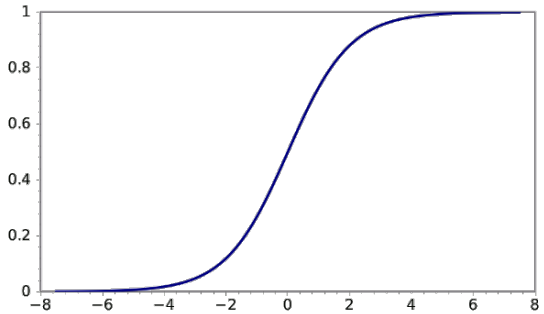
Red when:

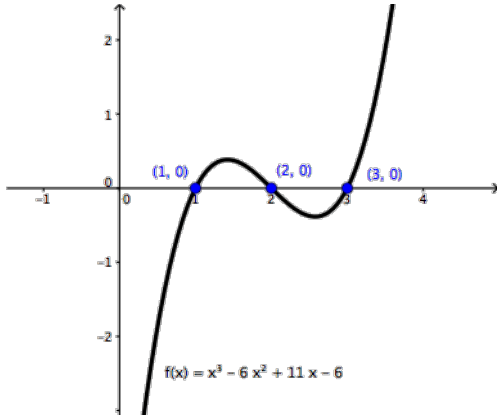
$$K(\vec{x}, \vec{l}^1) + K(\vec{x}, \vec{l}^2) =$$

Types of Kernel Functions

Types of Kernel Functions







Gaussian RBF Kernel

$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^2}{2\sigma^2}}$$

Sigmoid Kernel

$$K(X, Y) = \tanh(\gamma \cdot X^T Y + r)$$

Polynomial Kernel

$$K(X, Y) = (\gamma \cdot X^T Y + r)^d, \gamma > 0$$

Non-Linear SVR (Advanced)

Heads-up about Non-Linear SVR

Section on SVR:

- SVR Intuition ✓

Section on SVM:

- SVM Intuition ✓

Section on Kernel SVM:

- Kernel SVM Intuition ✓
- Mapping to a higher dimension ✓
- The Kernel Trick ✓
- Types of Kernel Functions ✓
- Non-linear Kernel SVR ←

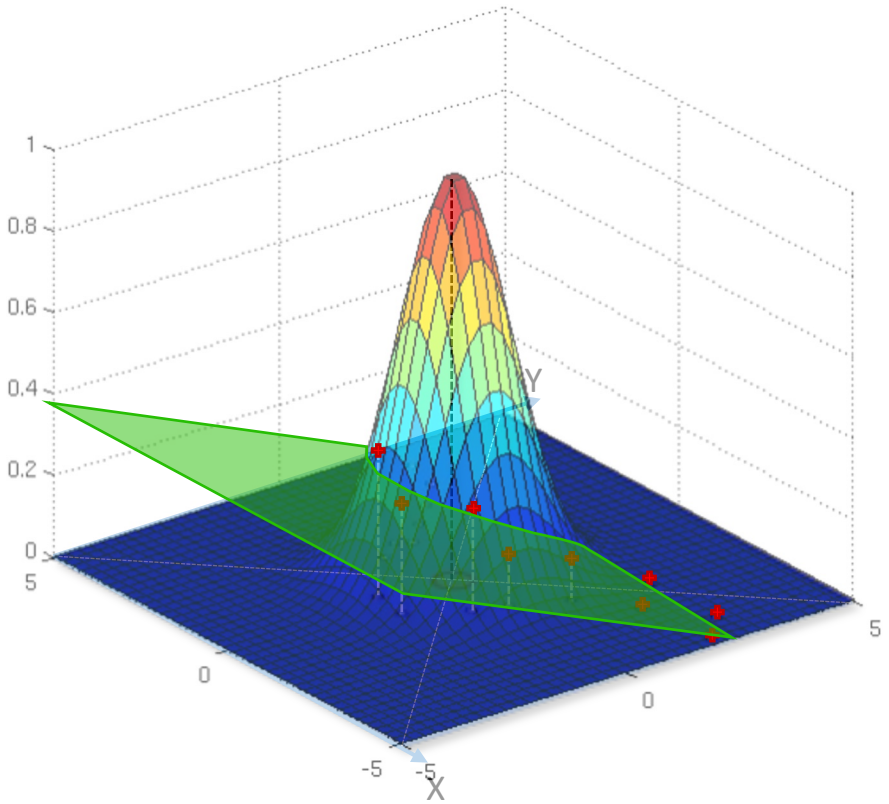
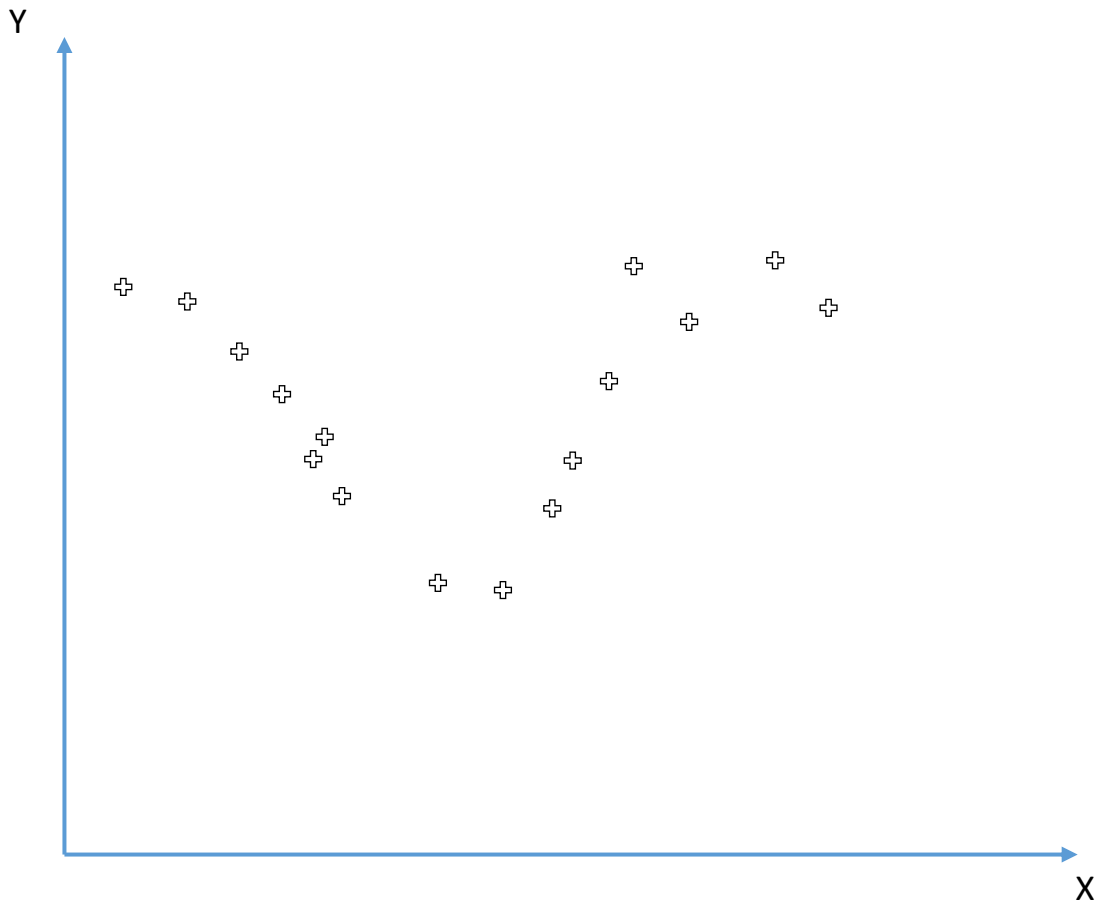
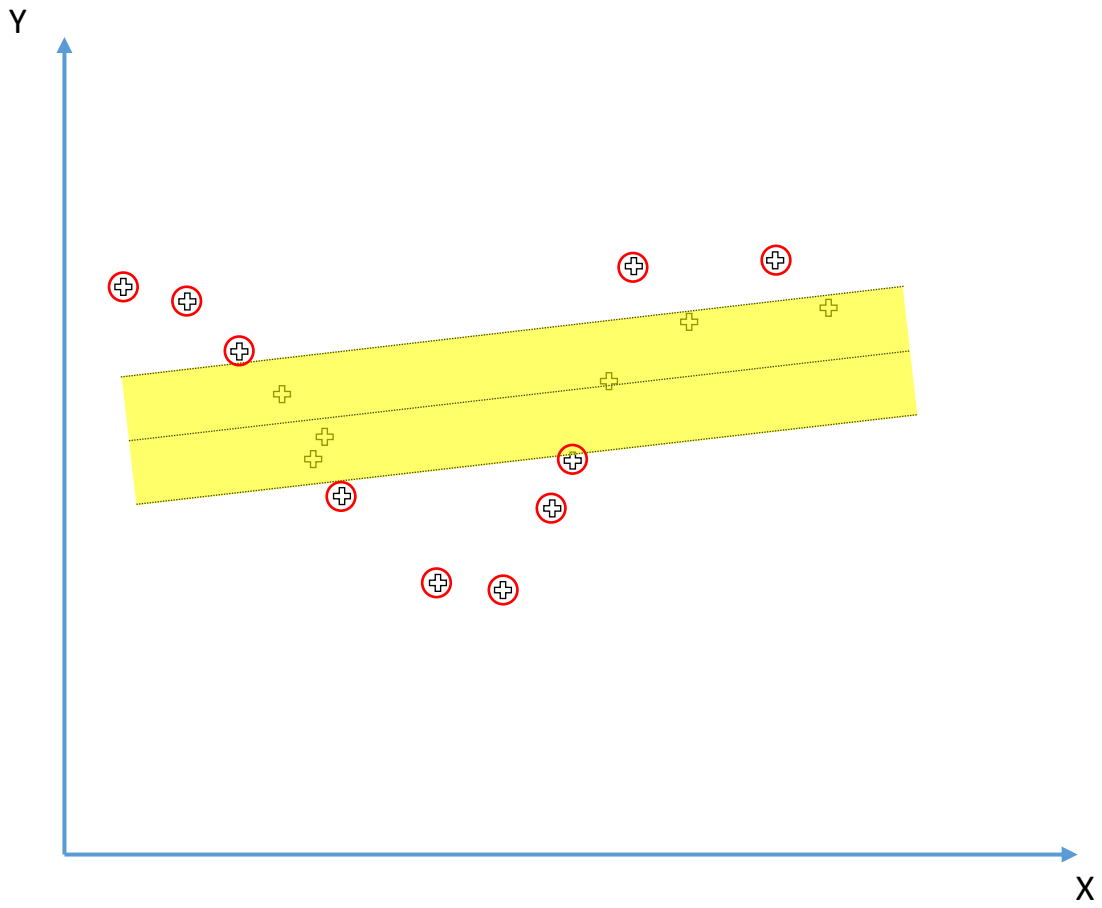


Image source: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html

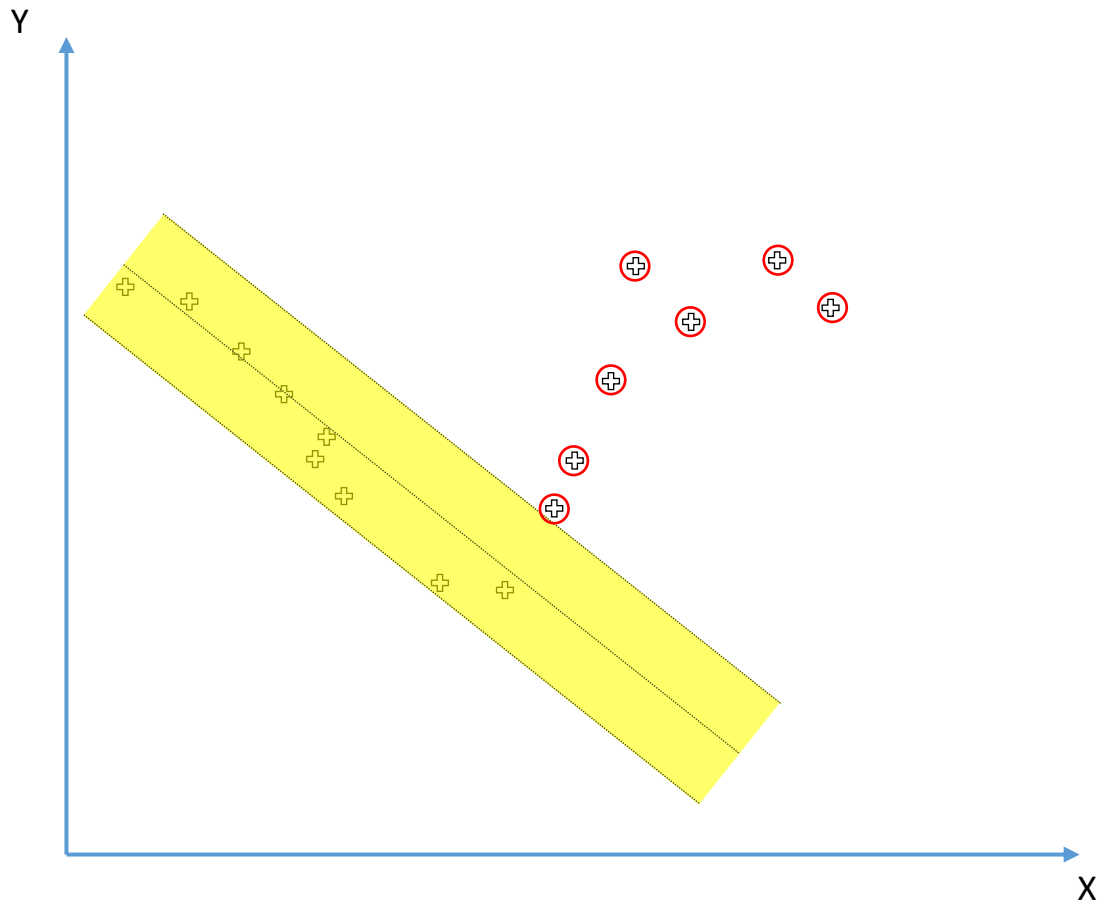
Non-Linear SVR



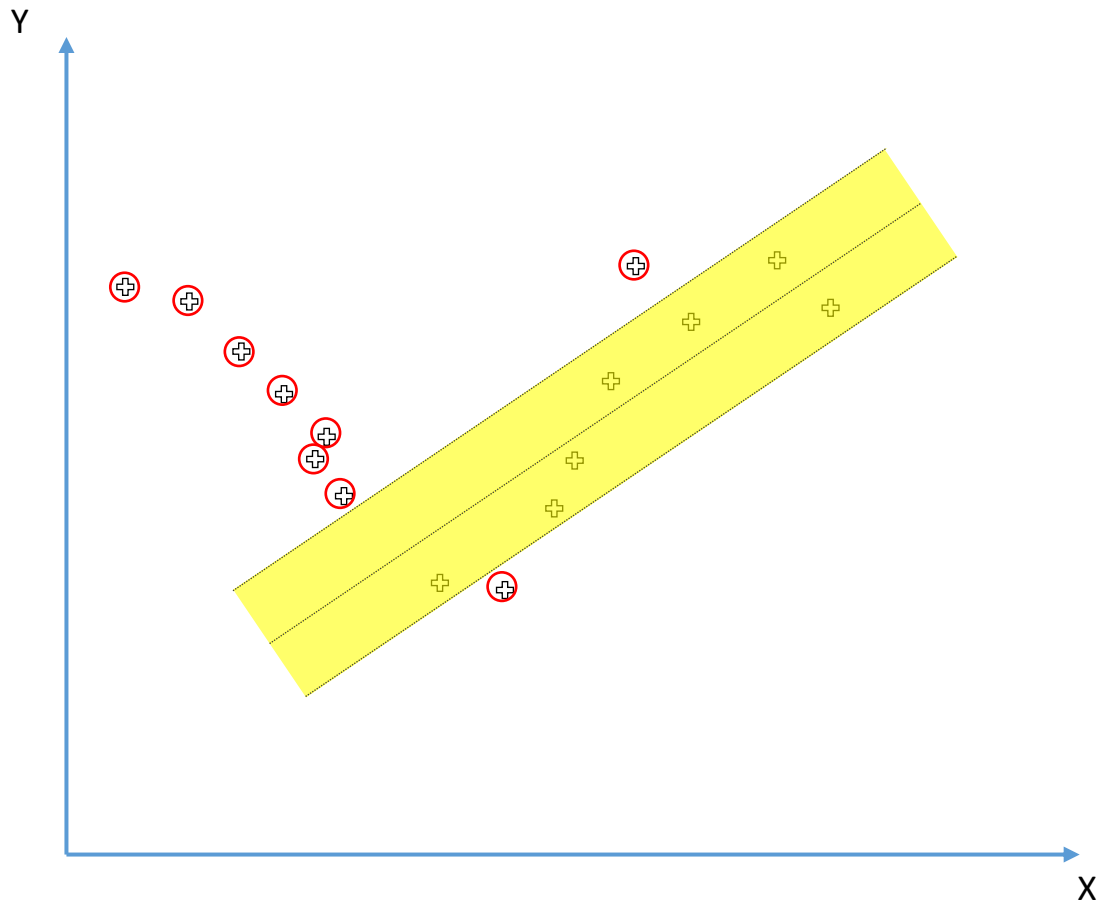
Non-Linear SVR



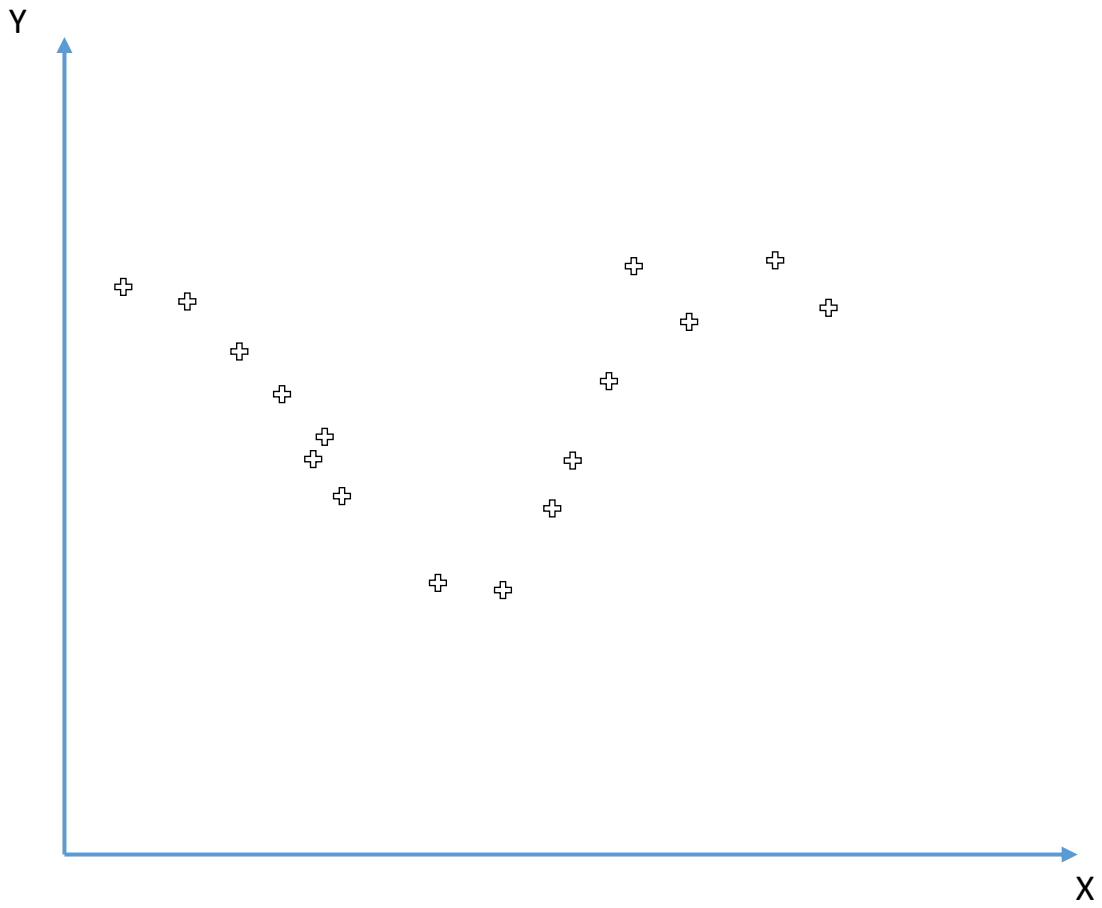
Non-Linear SVR



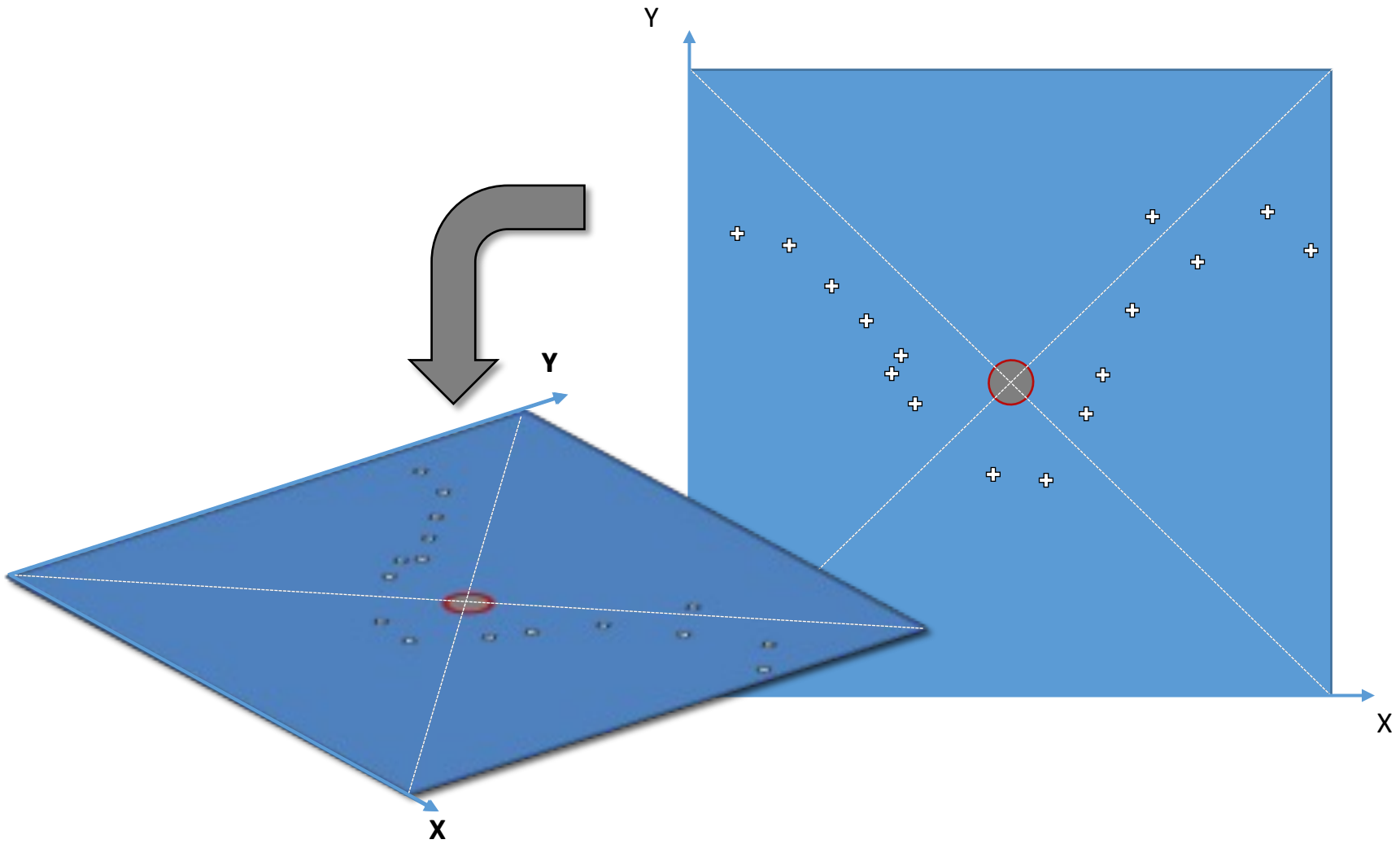
Non-Linear SVR



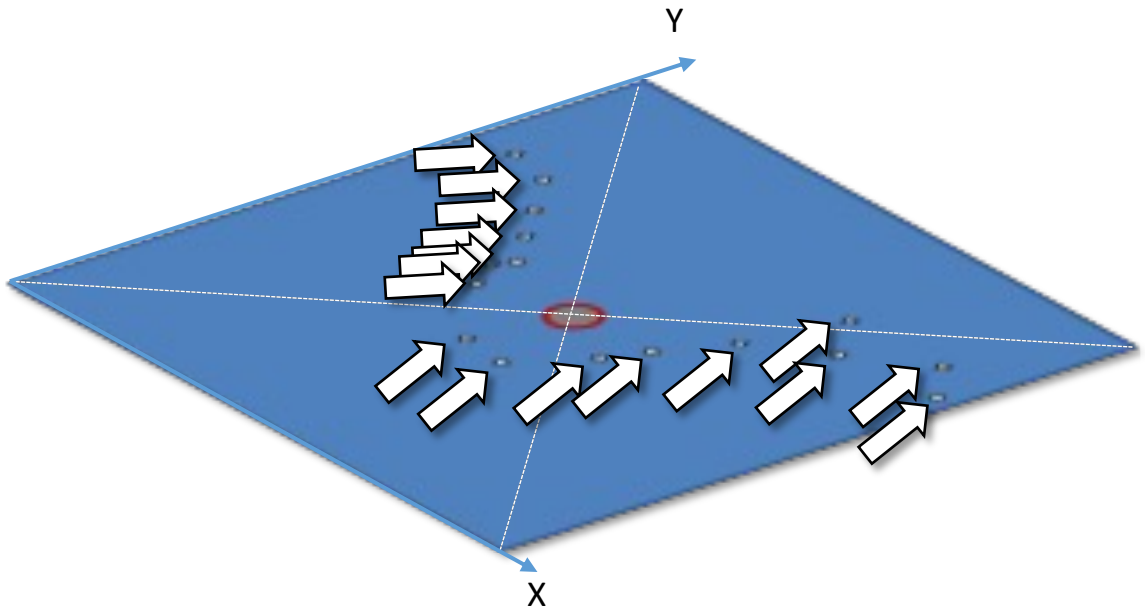
Non-Linear SVR



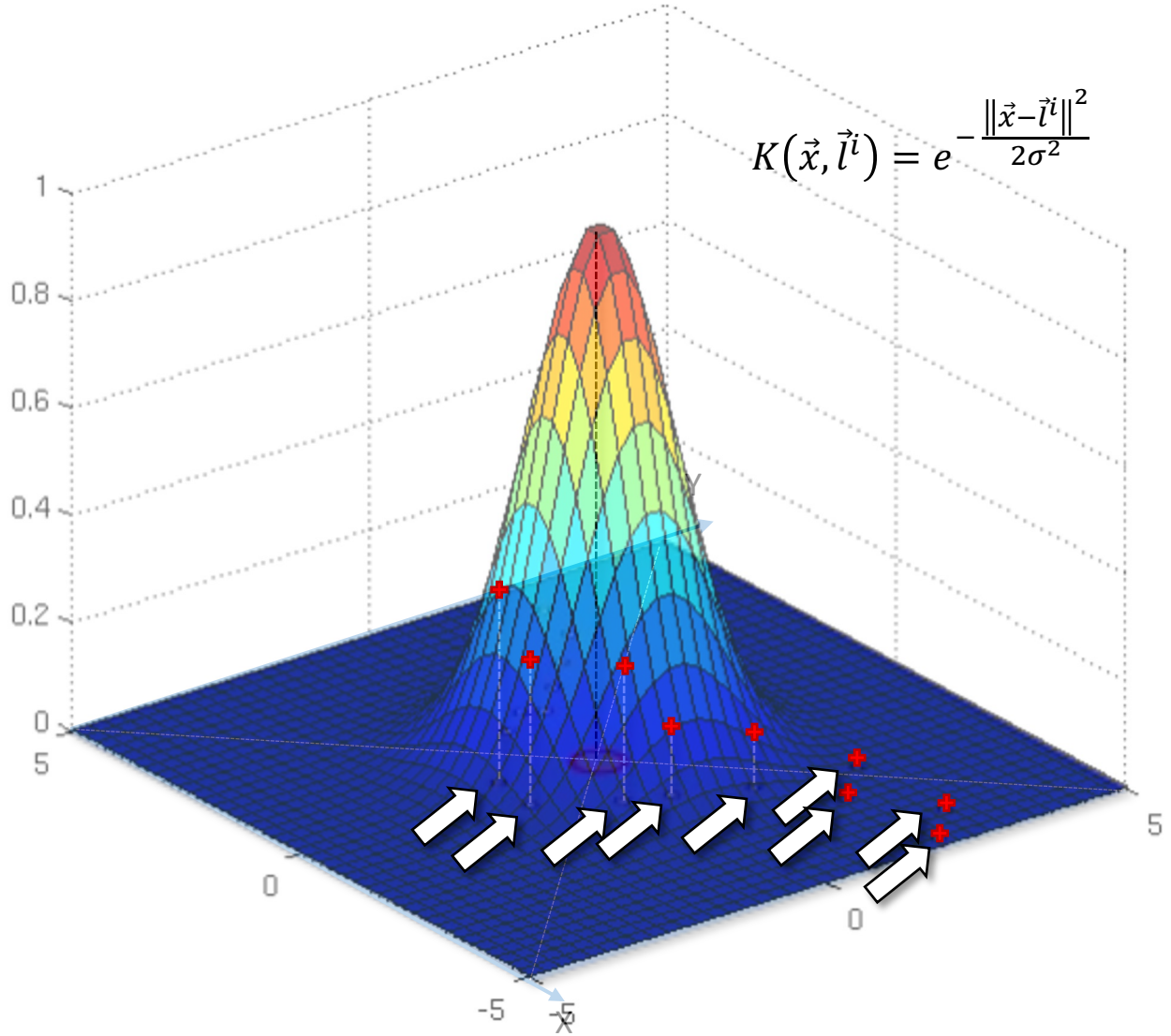
Non-Linear SVR



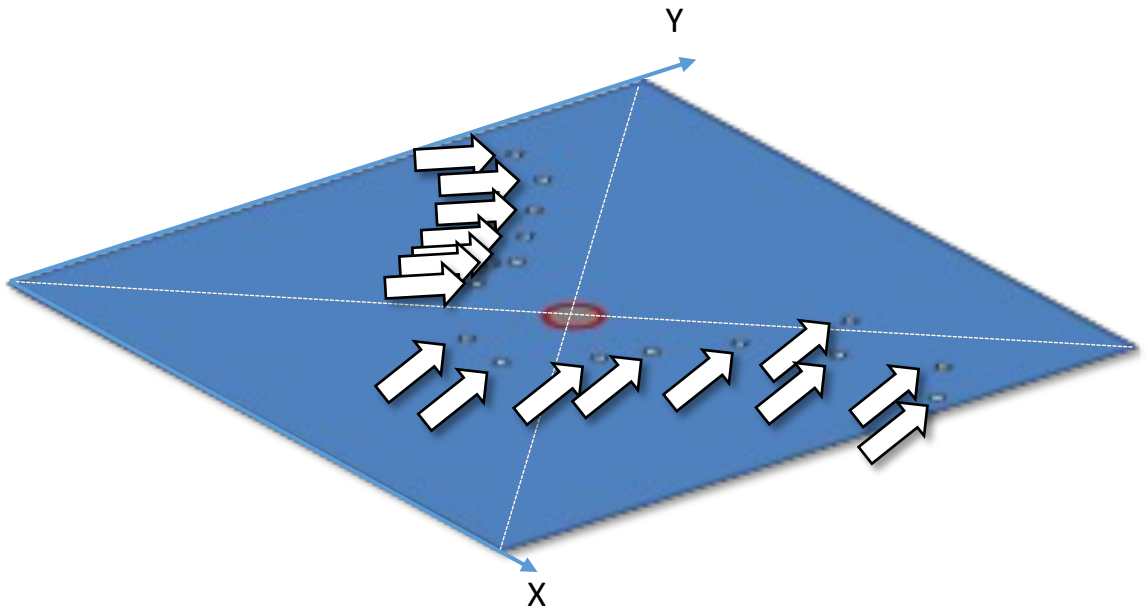
Non-Linear SVR

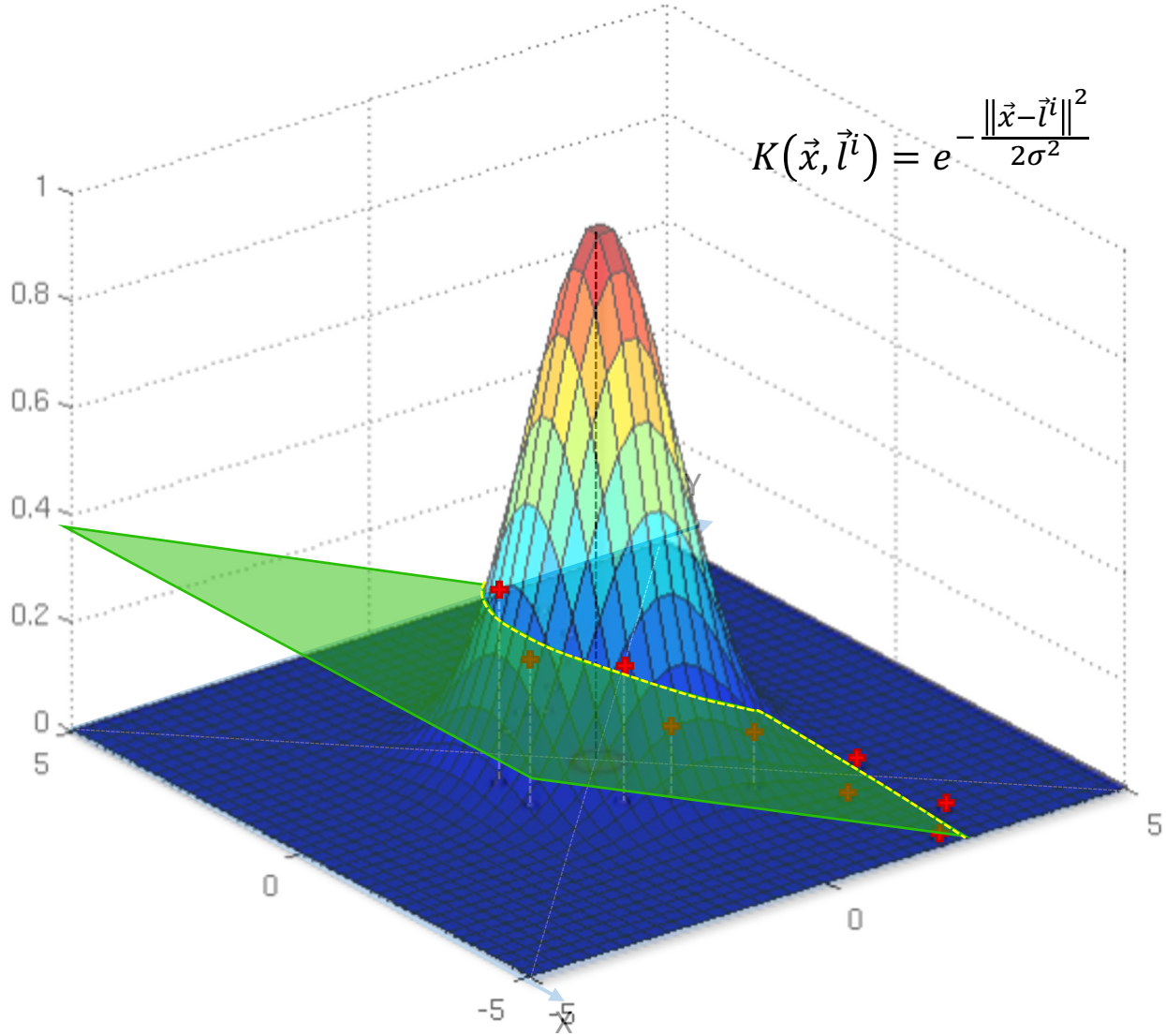


$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$$

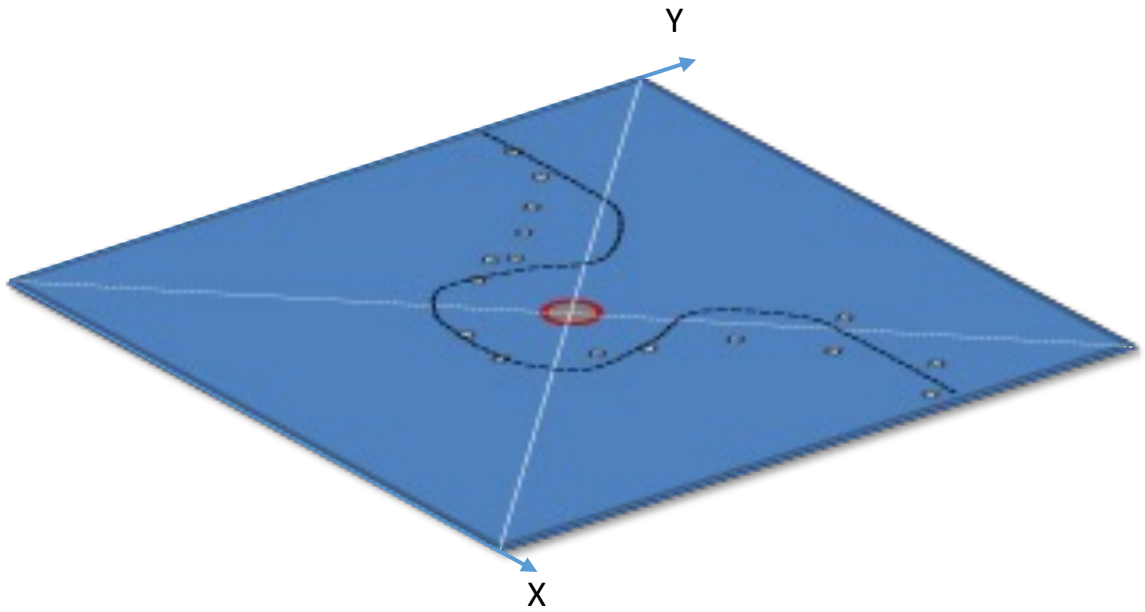


Non-Linear SVR

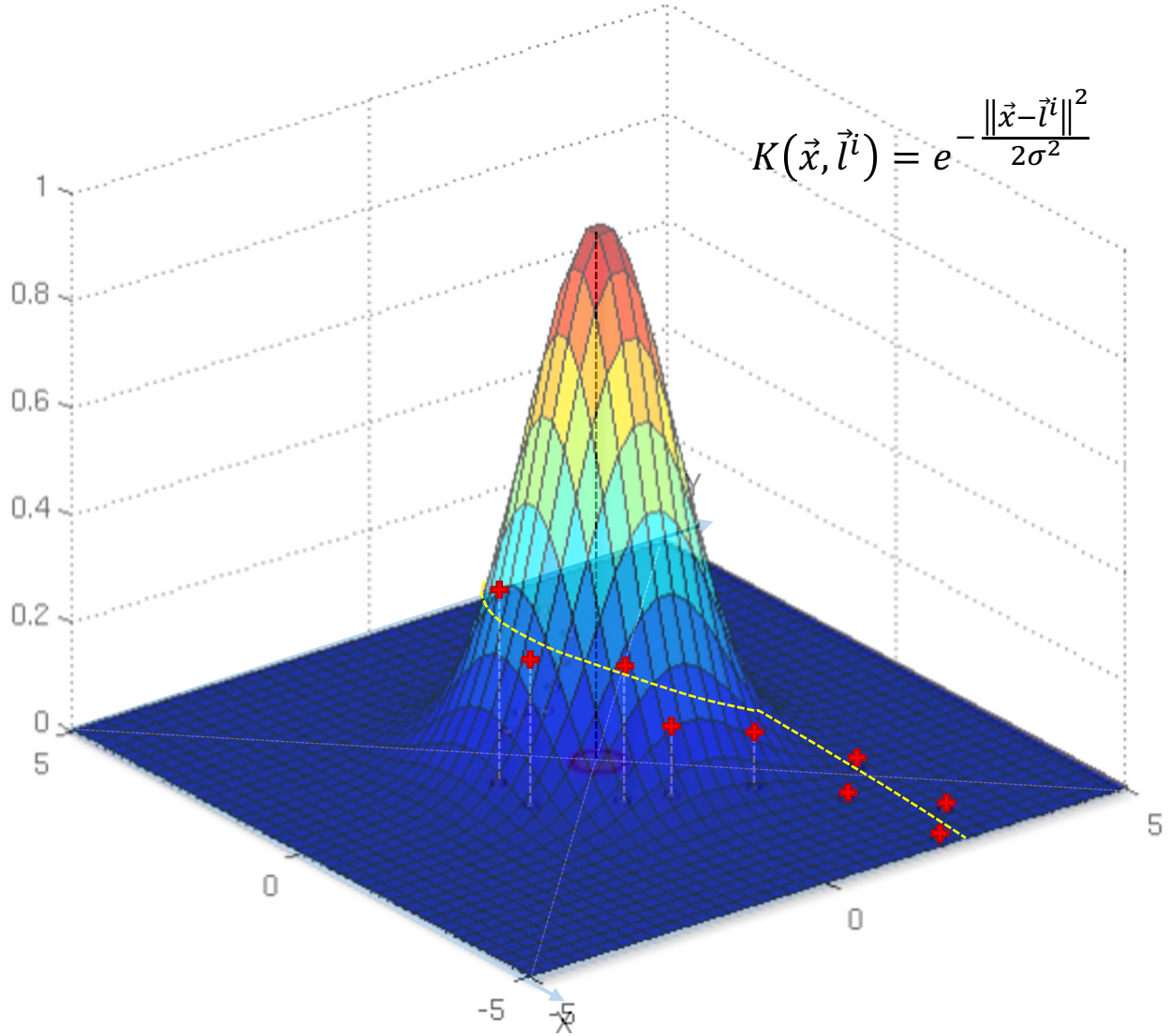




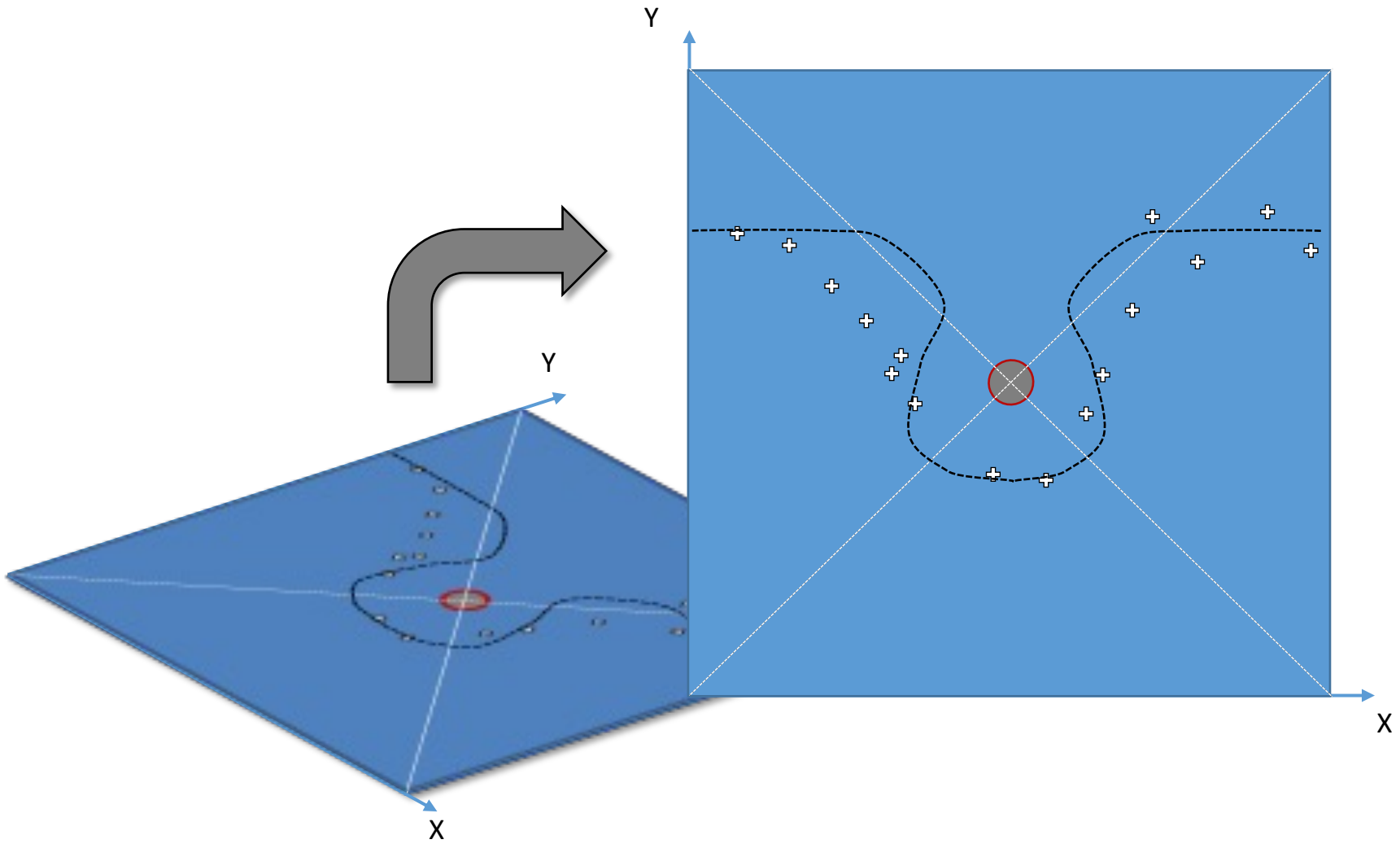
Non-Linear SVR



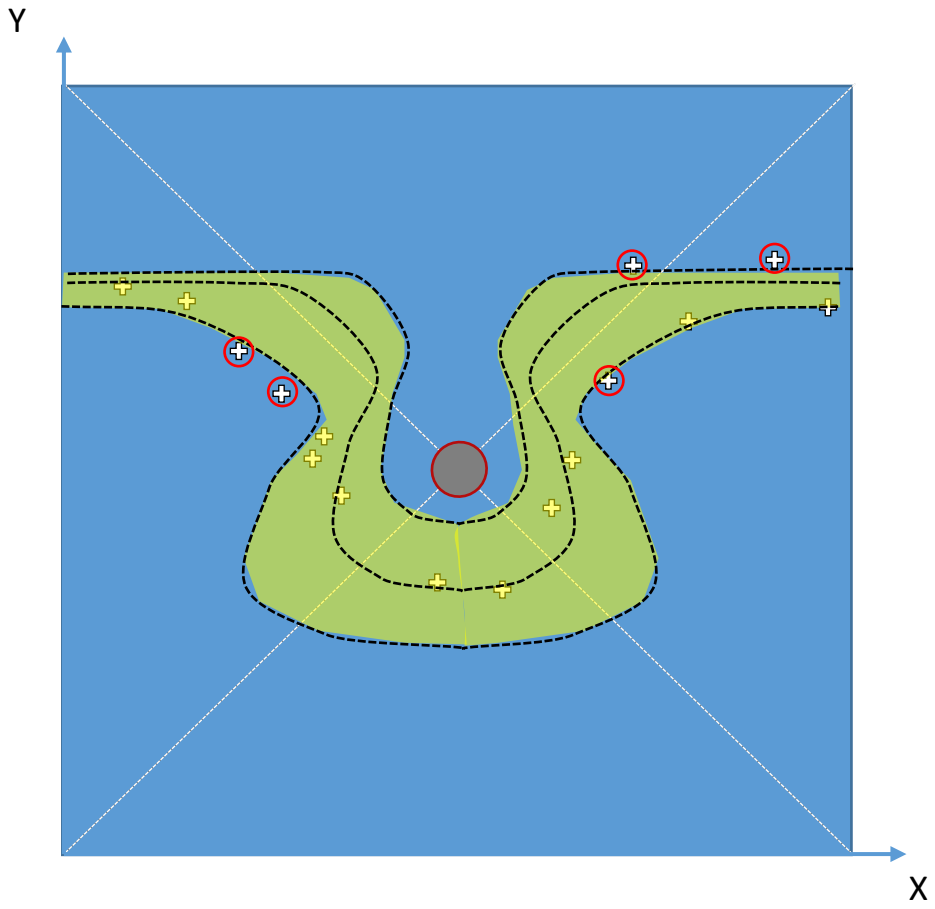
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$$

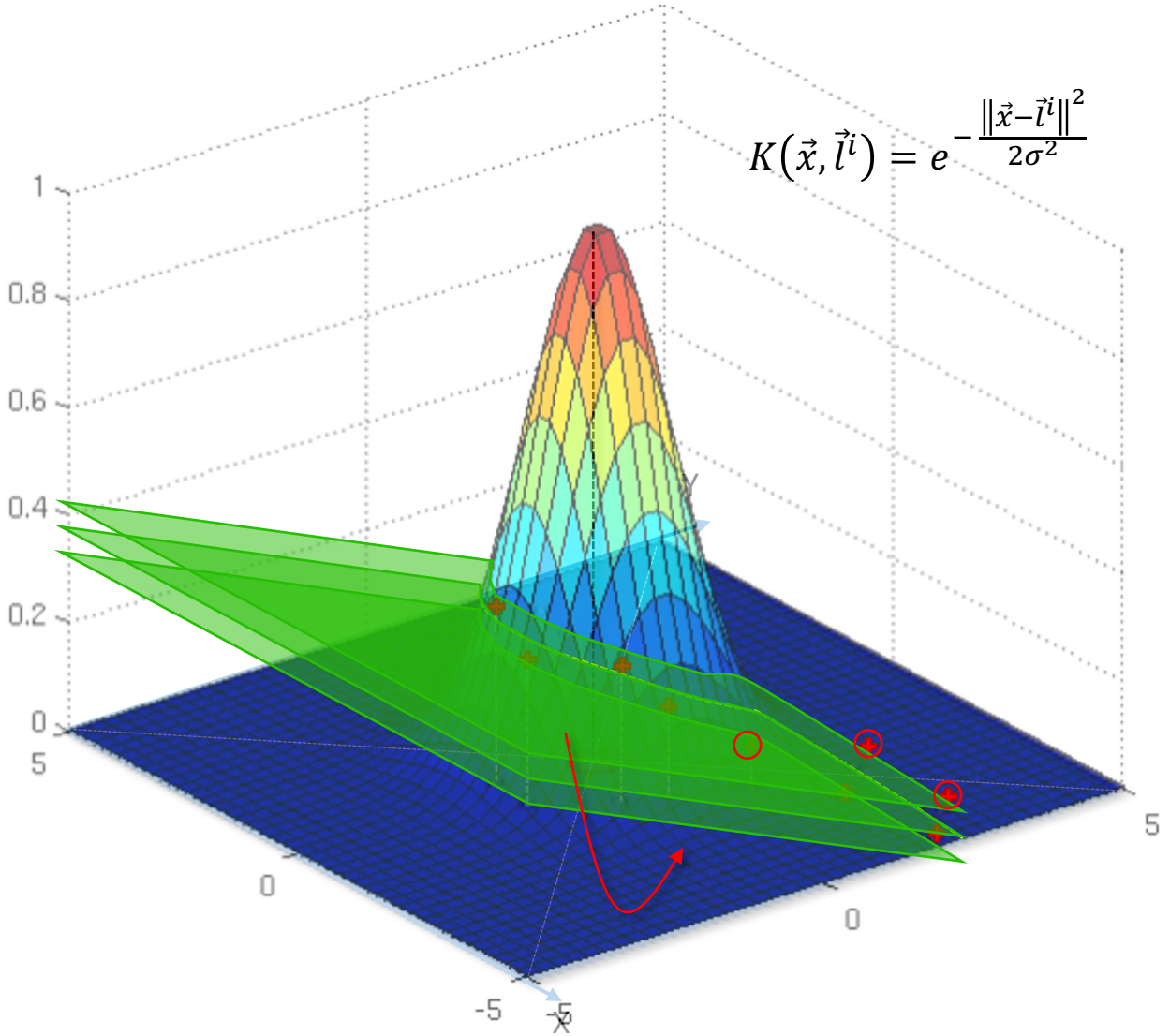


Non-Linear SVR



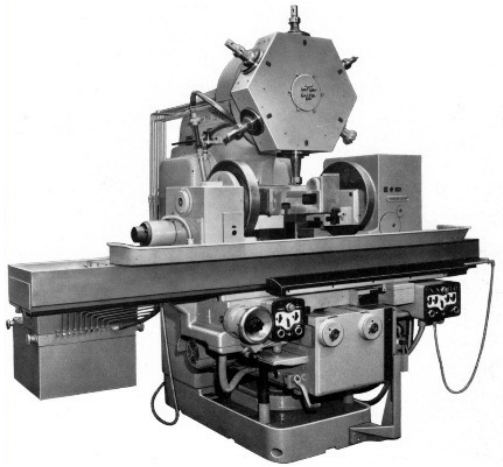
Non-Linear SVR

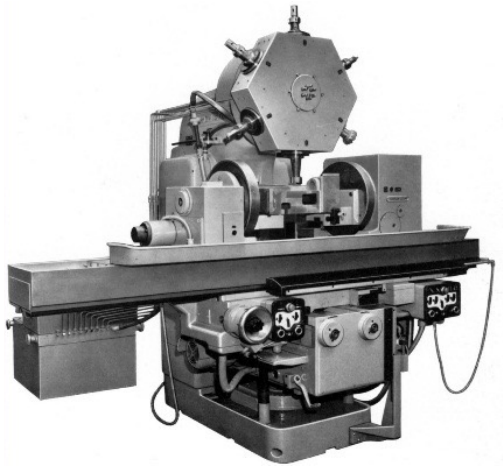




Bayes' Theorem

Bayes Theorem

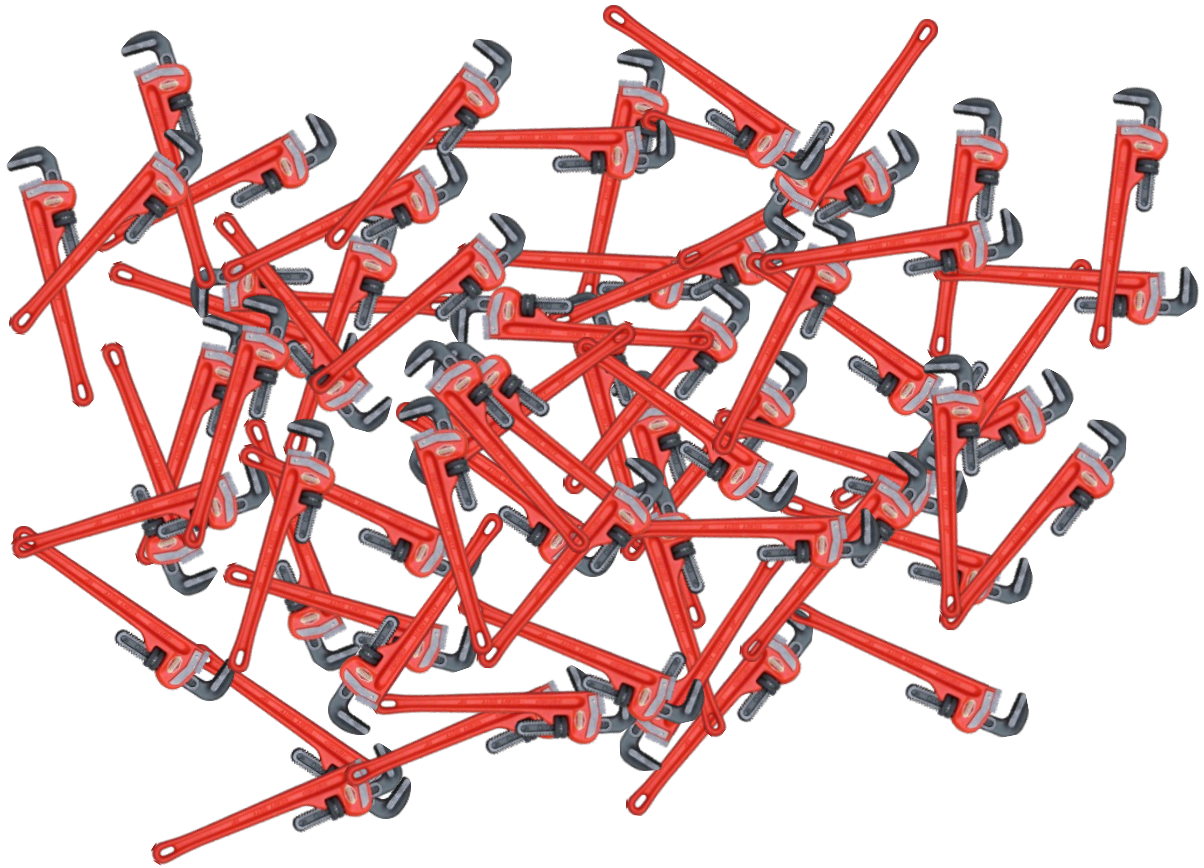




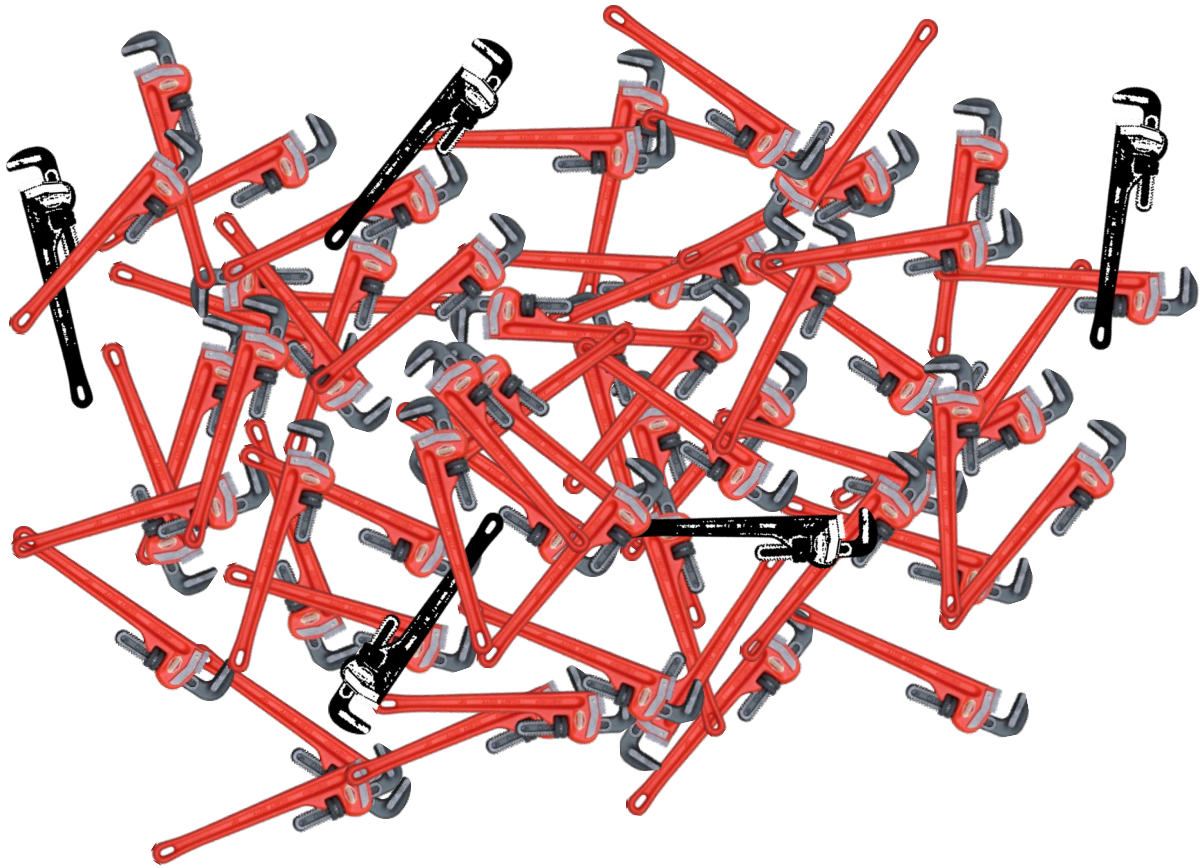
m1 m1 m1 m1 m1 m1 m1 m1 m1 m1 m1 m1 m1

m2 m2 m2 m2 m2 m2 m2 m2 m2 m2

Bayes Theorem

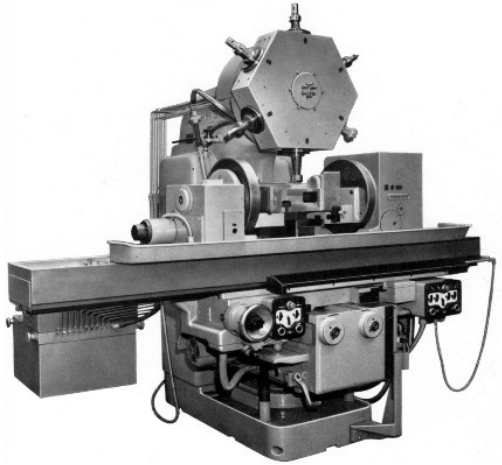


Bayes Theorem



Bayes Theorem

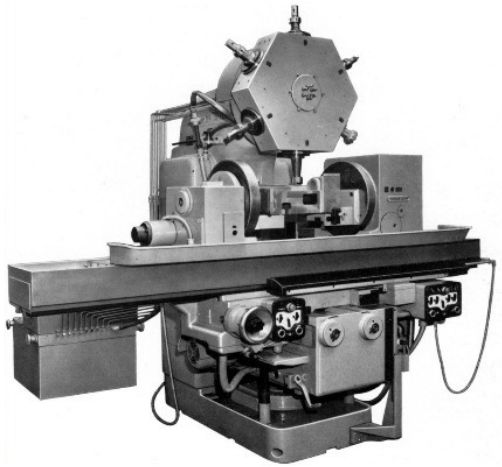
What's the probability?



m2

Bayes Theorem

What's the probability?



m2

Bayes Theorem

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

Out of all produced parts:
We can SEE that 1% are defective

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

Question:
What is the probability that a part
produced by mach2 is defective = ?

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

-> P(Mach1) = 30/50 = 0.6
-> P(Mach2) = 20/50 = 0.4

Out of all produced parts:
We can SEE that 1% are defective

-> P(Defect) = 1%

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

-> P(Mach1 | Defect) = 50%
-> P(Mach2 | Defect) = 50%

Question:
What is the probability that a part produced by mach2 is defective = ?

-> P(Defect | Mach2) = ?

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

~~-> P(Mach1) = 30/50 = 0.6~~
-> P(Mach2) = 20/50 = 0.4

Out of all produced parts:
We can SEE that 1% are defective

-> P(Defect) = 1%

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

~~-> P(Mach1 | Defect) = 50%~~
-> P(Mach2 | Defect) = 50%

Question:
What is the probability that a part produced by mach2 is defective = ?

-> P(Defect | Mach2) = ?

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr
Out of all produced parts:
We can SEE that 1% are defective
Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2
Question:
What is the probability that a part produced by mach2 is defective = ?

-> P(Mach2) = 20/50 = 0.4
-> P(Defect) = 1%
-> P(Mach2 | Defect) = 50%
-> P(Defect | Mach2) = ?

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr
Out of all produced parts:
We can SEE that 1% are defective
Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2
Question:
What is the probability that a part produced by mach2 is defective = ?

-> P(Mach2) = 20/50 = 0.4
-> P(Defect) = 1%
-> P(Mach2 | Defect) = 50%
-> P(Defect | Mach2) = ?

$$P(Defect \mid Mach2) = \frac{P(Mach2 \mid Defect) * P(Defect)}{P(Mach2)}$$

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr
Out of all produced parts:
We can SEE that 1% are defective
Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2
Question:
What is the probability that a part produced by mach2 is defective = ?

-> P(Mach2) = 20/50 = 0.4
-> P(Defect) = 1%
-> P(Mach2 | Defect) = 50%
-> P(Defect | Mach2) = ?

$$P(\text{Defect} \mid \text{Mach2}) = \frac{0.5 * 0.01}{0.4}$$

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr
Out of all produced parts:
We can SEE that 1% are defective
Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2
Question:
What is the probability that a part produced by mach2 is defective = ?

-> P(Mach2) = 20/50 = 0.4
-> P(Defect) = 1%
-> P(Mach2 | Defect) = 50%
-> P(Defect | Mach2) = ?

$$P(\text{Defect} \mid \text{Mach2}) = \frac{0.5 * 0.01}{0.4} = 0.0125 = 1.25\%$$

It's intuitive!

$$P(\text{Defect} \mid \text{Mach2}) = \frac{P(\text{Mach2} \mid \text{Defect}) * P(\text{Defect})}{P(\text{Mach2})} = 1.25\%$$

Let's look at an example:

- 1000 wrenches
- 400 came from Mach2
- 1% have a defect = 10
- of them 50% came from Mach2 = 5
- % defective parts from Mach2 = 5/400 = 1.25%

It’s intuitive!

Obvious question:
If the items are labeled, why couldn’t we just count the number of defective wrenches that came from Mach2 and divide by the total number that came from Mach2?

Bayes Theorem

Quick exercise:

P(Defect | Mach1) = ?

Bayes Theorem

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

-> P(Mach1) = 30/50 = 0.6
-> P(Mach2) = 20/50 = 0.4

Out of all produced parts:
We can SEE that 1% are defective

-> P(Defect) = 1%

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

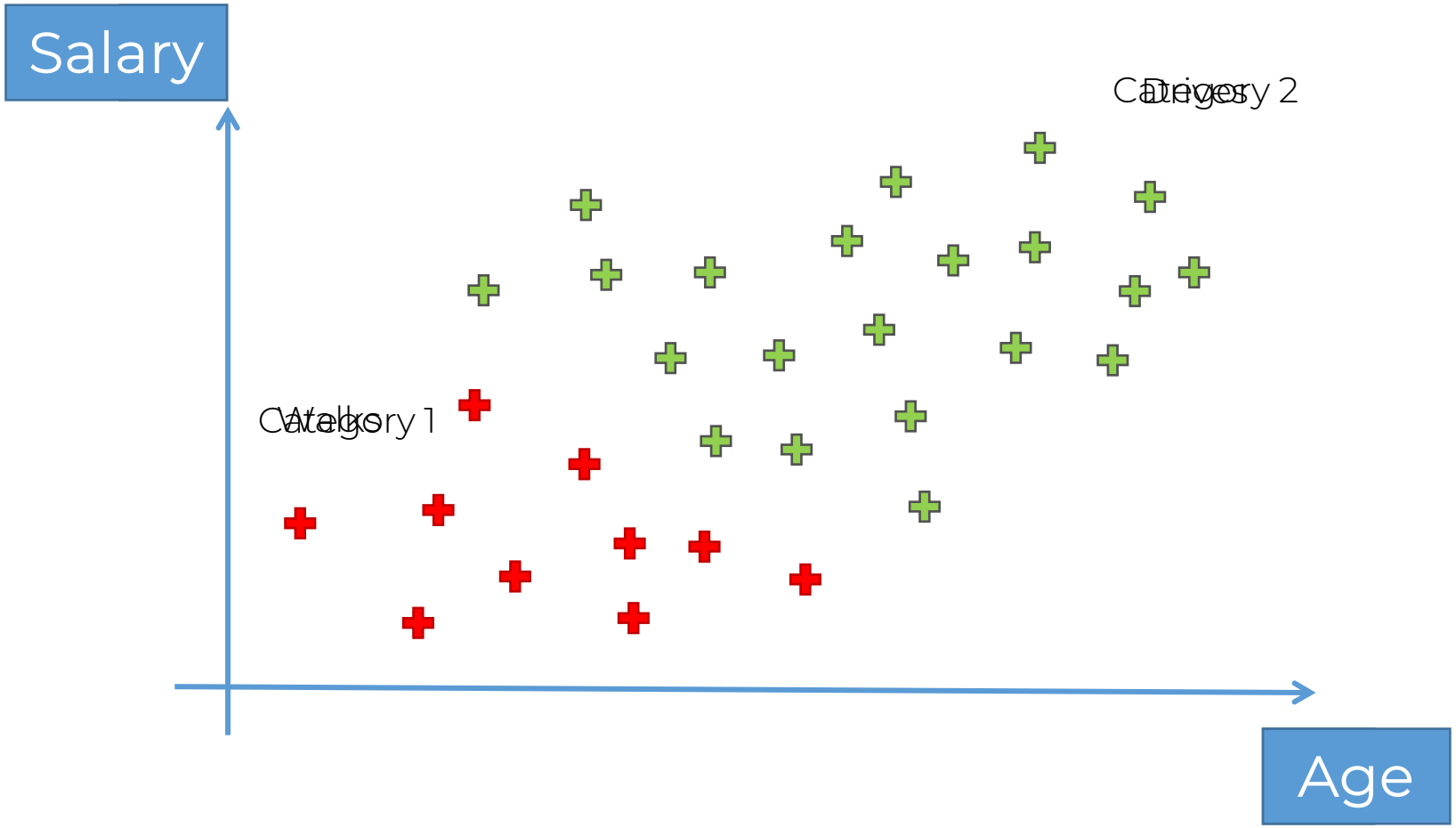
-> P(Mach1 | Defect) = 50%
-> P(Mach2 | Defect) = 50%

Naïve Bayes Classifier Intuition

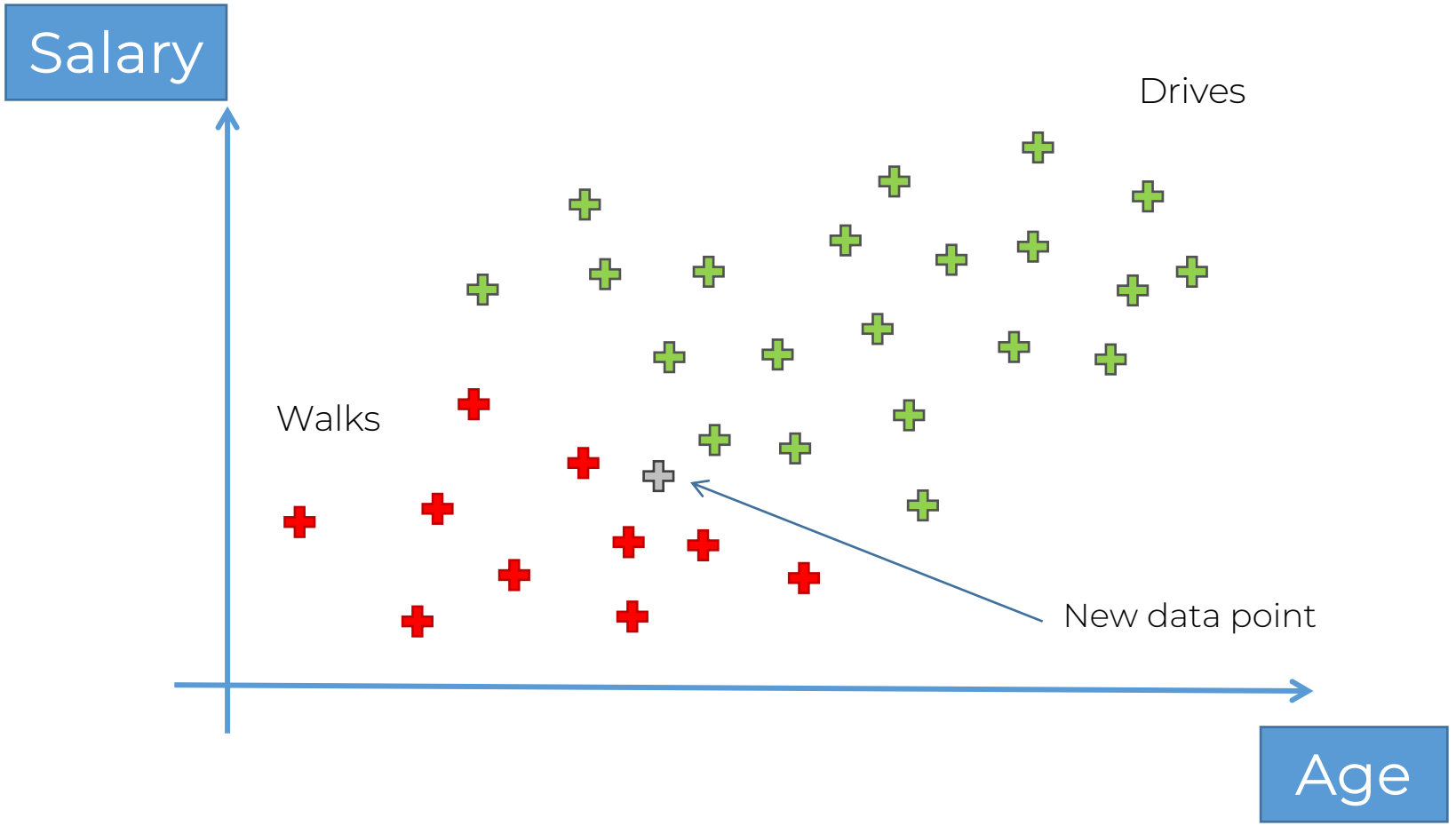
Naïve Bayes

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Naïve Bayes



Naïve Bayes



Naïve Bayes

Plan of Attack

Naïve Bayes

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Step 1

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

#2 Marginal Likelihood

Step 2

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Drives|X) = \frac{P(X|Drives) * P(Drives)}{P(X)}$$

#2 Marginal Likelihood

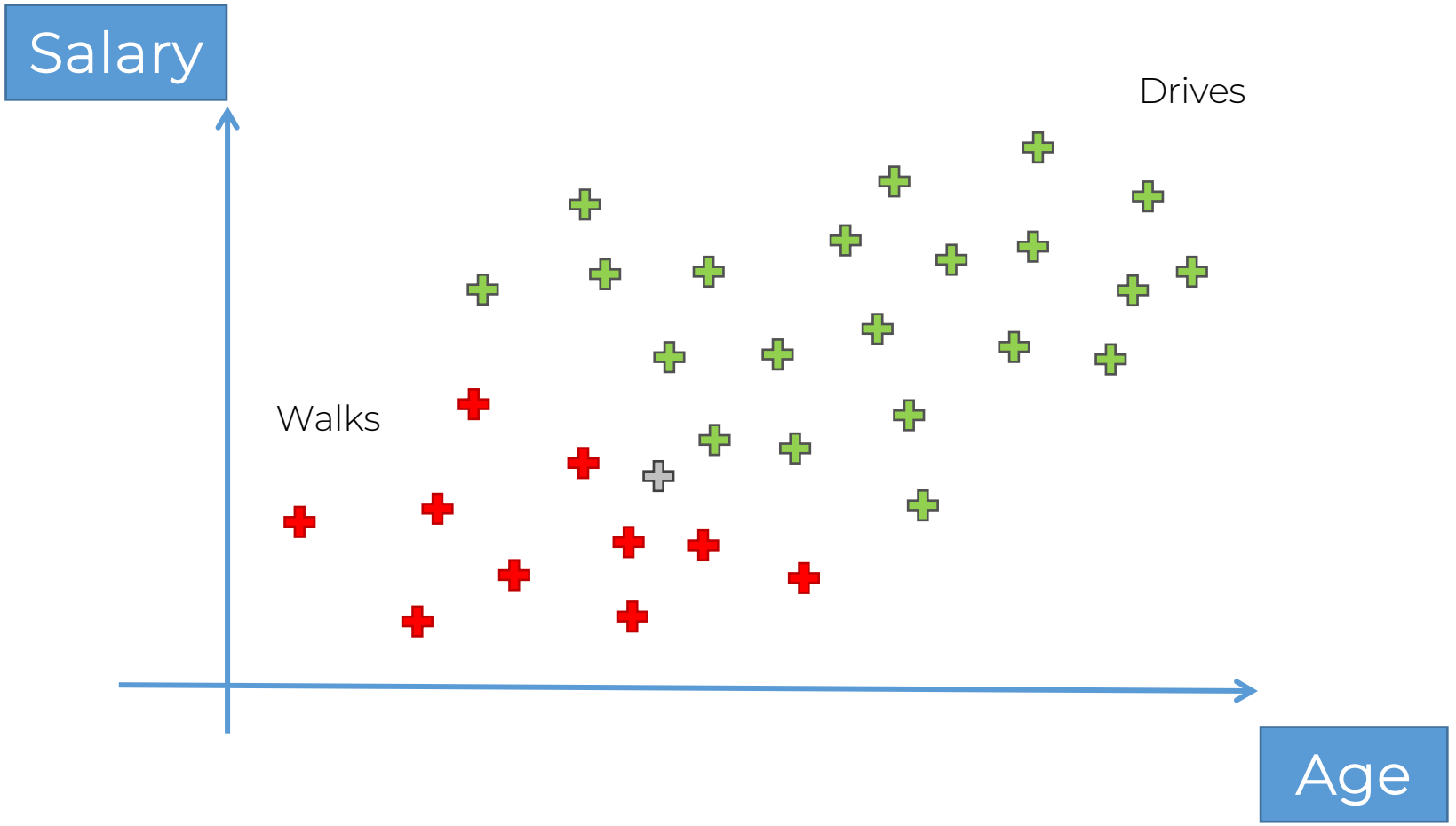
Step 3

$$P(Walks|X) \; v.s. \; P(Drives|X)$$

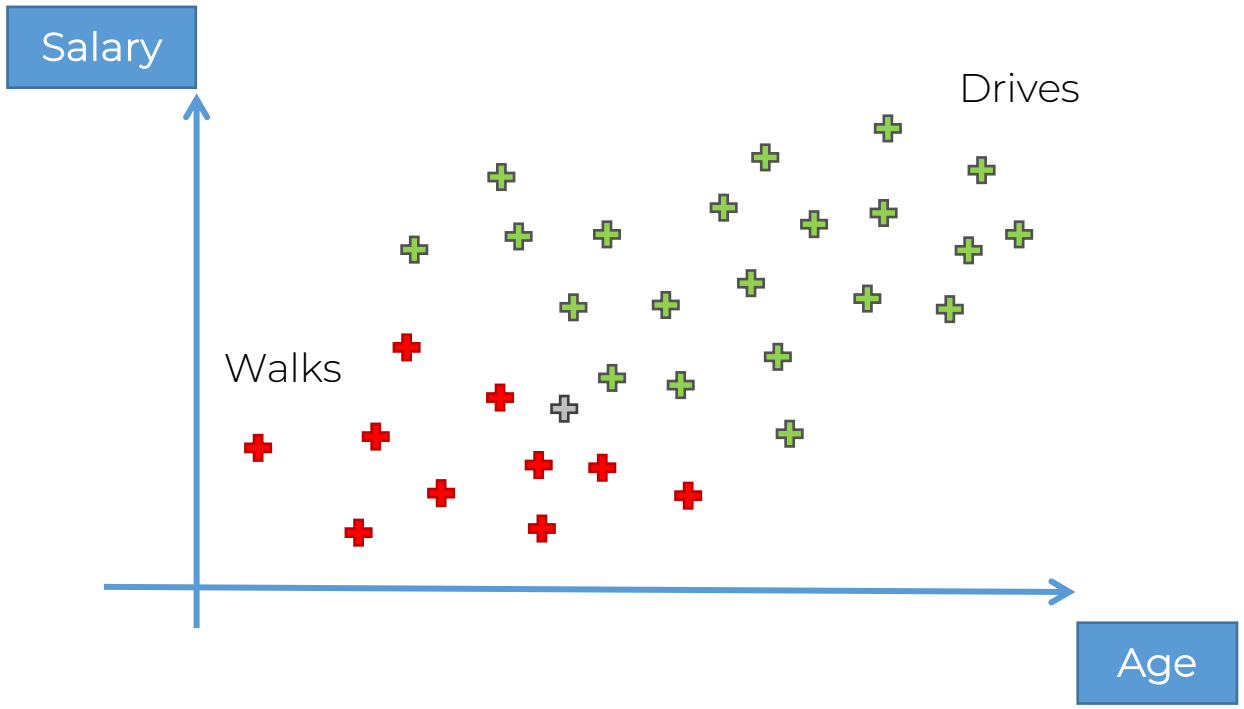
Naïve Bayes

Ready?

Naïve Bayes: Step 1



Naïve Bayes: Step 1



#1. P(Walks)

$$P(Walks) = \frac{Number\ of\ Walkers}{Total\ Observations}$$

$$P(Walks) = \frac{10}{30}$$

Naïve Bayes: Step 1

#4 Posterior Probability

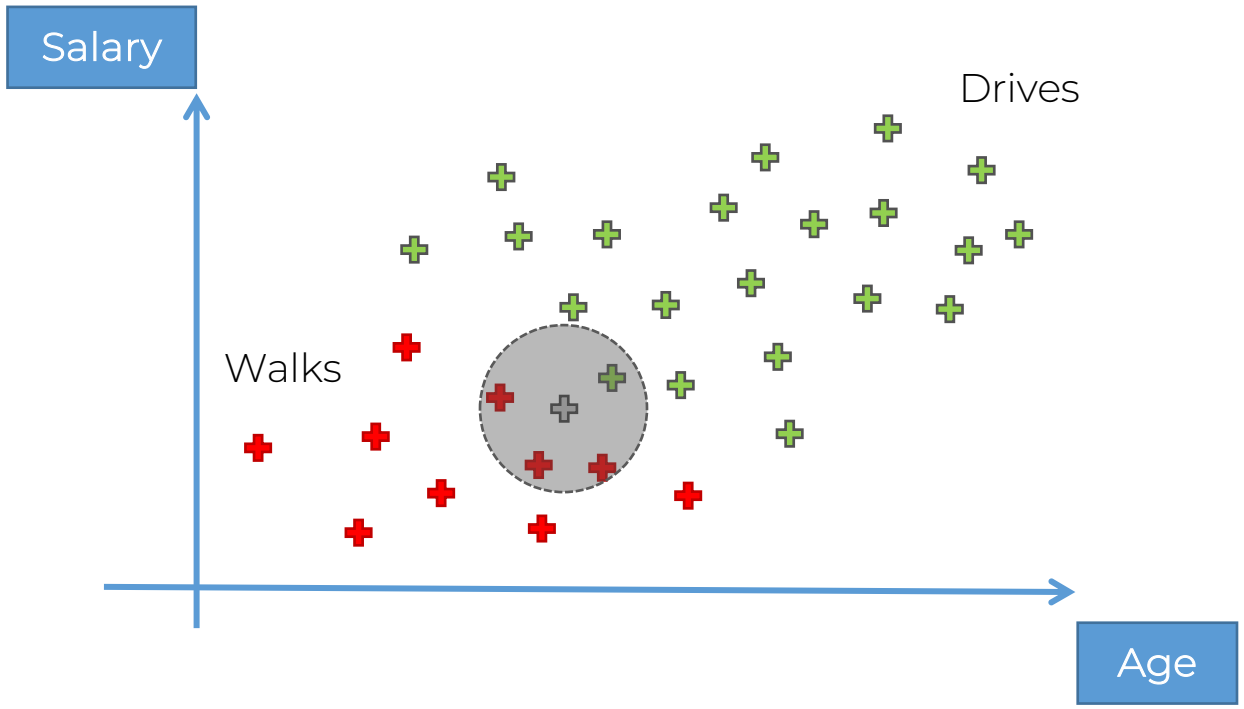
#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

#2 Marginal Likelihood

Naïve Bayes: Step 1



#2. P(X)

$$P(X) = \frac{Number\ of\ Similar\ Observations}{Total\ Observations}$$

$$P(X) = \frac{4}{30}$$

Naïve Bayes: Step 1

#4 Posterior Probability

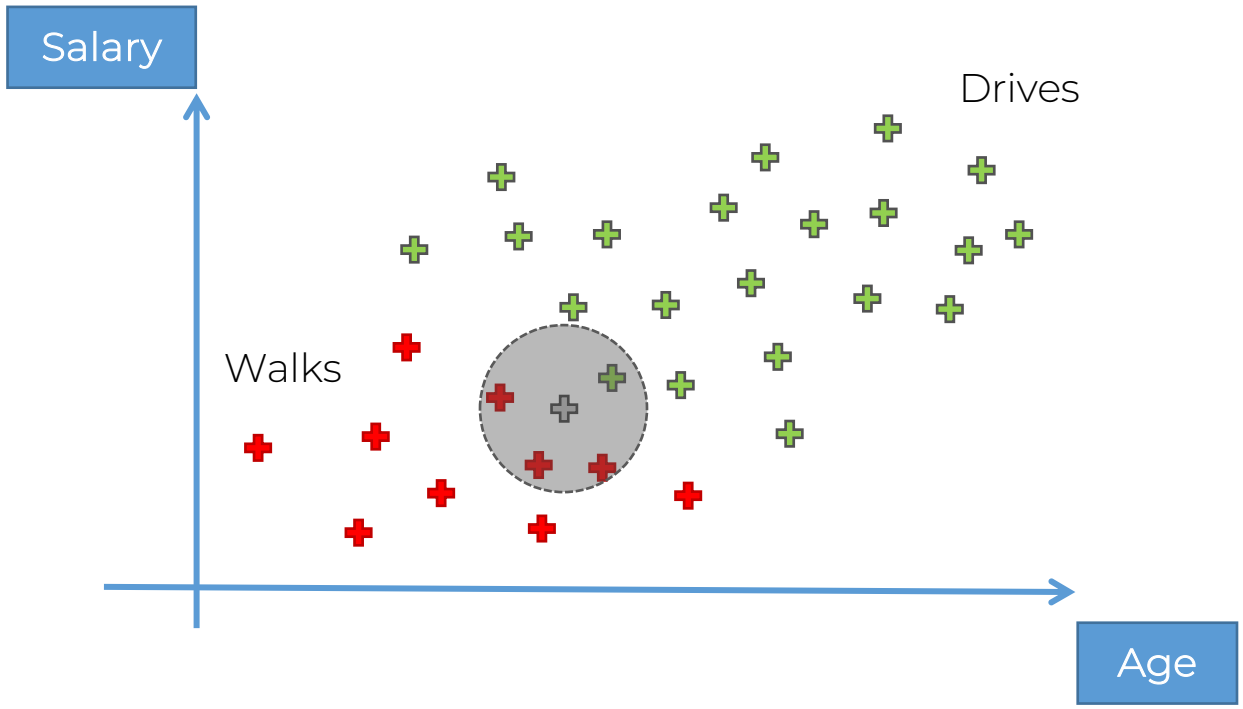
#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

#2 Marginal Likelihood

Naïve Bayes: Step 1



#3. P(X|Walks)

$$P(X|Walks) = \frac{Number\ of\ Similar\ Observations\ Among\ those\ who\ Walk}{Total\ number\ of\ Walkers}$$

$$P(X|Walks) = \frac{3}{10}$$

Naïve Bayes: Step 1

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

#2 Marginal Likelihood

Naïve Bayes: Step 1

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{\frac{3}{10} * \frac{10}{30}}{\frac{4}{30}} = 0.75$$

#2 Marginal Likelihood

Naïve Bayes

Step 1 – Done.

Step 2

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Drives|X) = \frac{P(X|Drives) * P(Drives)}{P(X)}$$

#2 Marginal Likelihood

Naïve Bayes: Step 2

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Drives|X) = \frac{\frac{1}{20} * \frac{20}{30}}{\frac{4}{30}} = 0.25$$

#2 Marginal Likelihood

Naïve Bayes

Step 2 – Done.

Step 3

$$P(Walks|X) \; v.s. \; P(Drives|X)$$

Step 3

$$0.75 \; v.s. \; 0.25$$

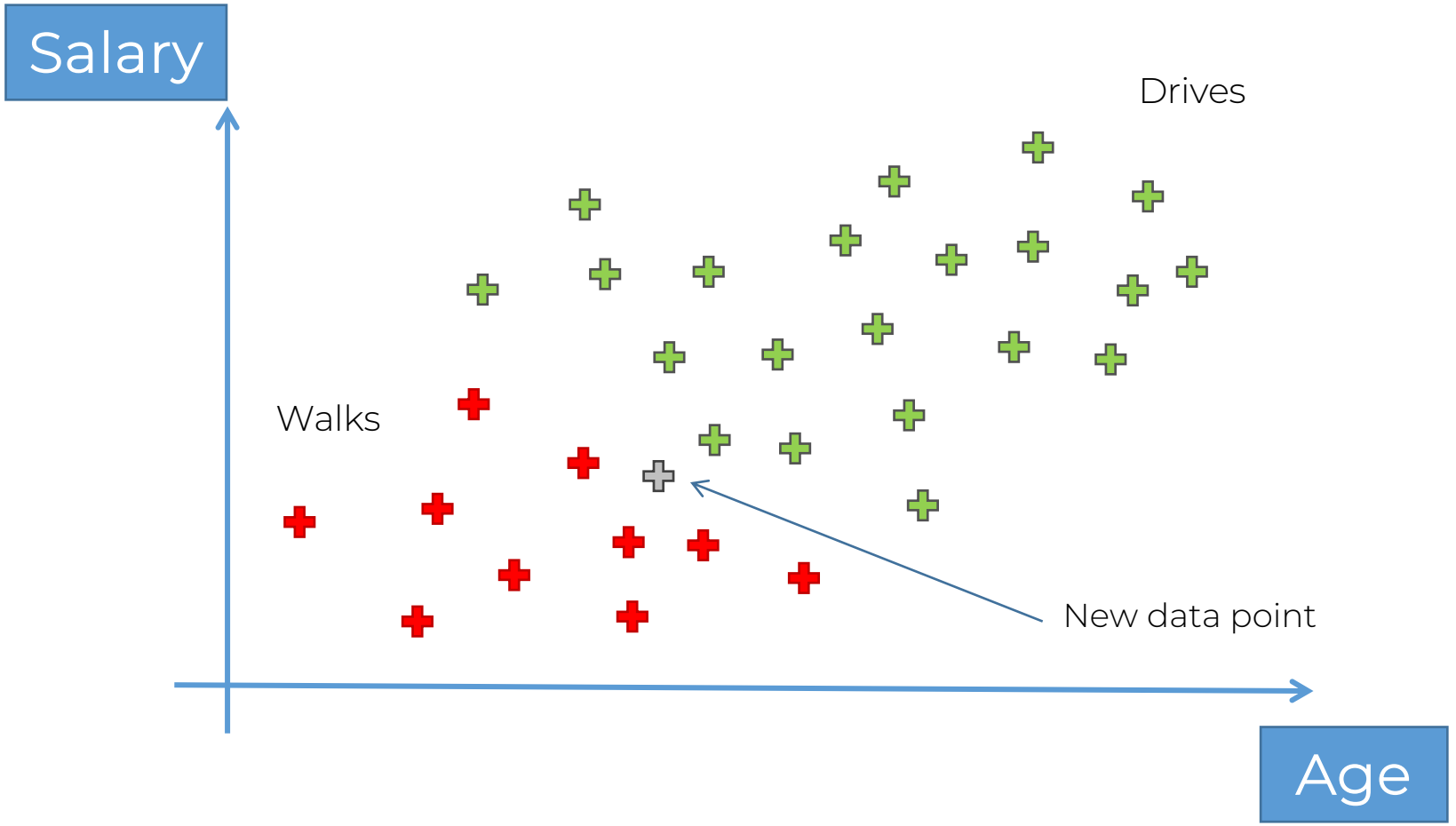
Step 3

$$0.75 > 0.25$$

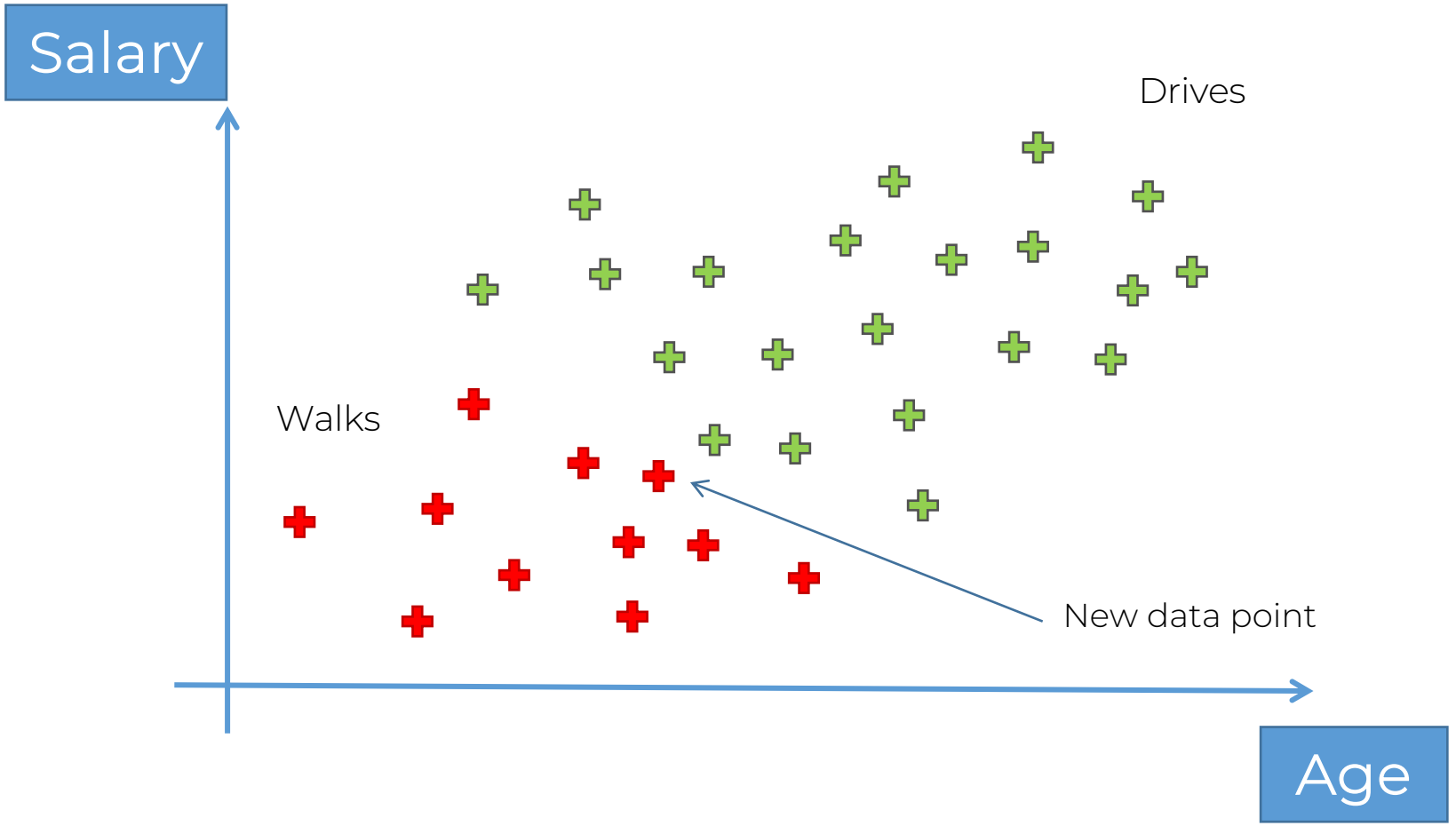
Step 3

$$P(Walks|X) > P(Drives|X)$$

Naïve Bayes



Naïve Bayes



Naïve Bayes Classifier Intuition (Challenge Reveal)

Step 2

#4 Posterior Probability

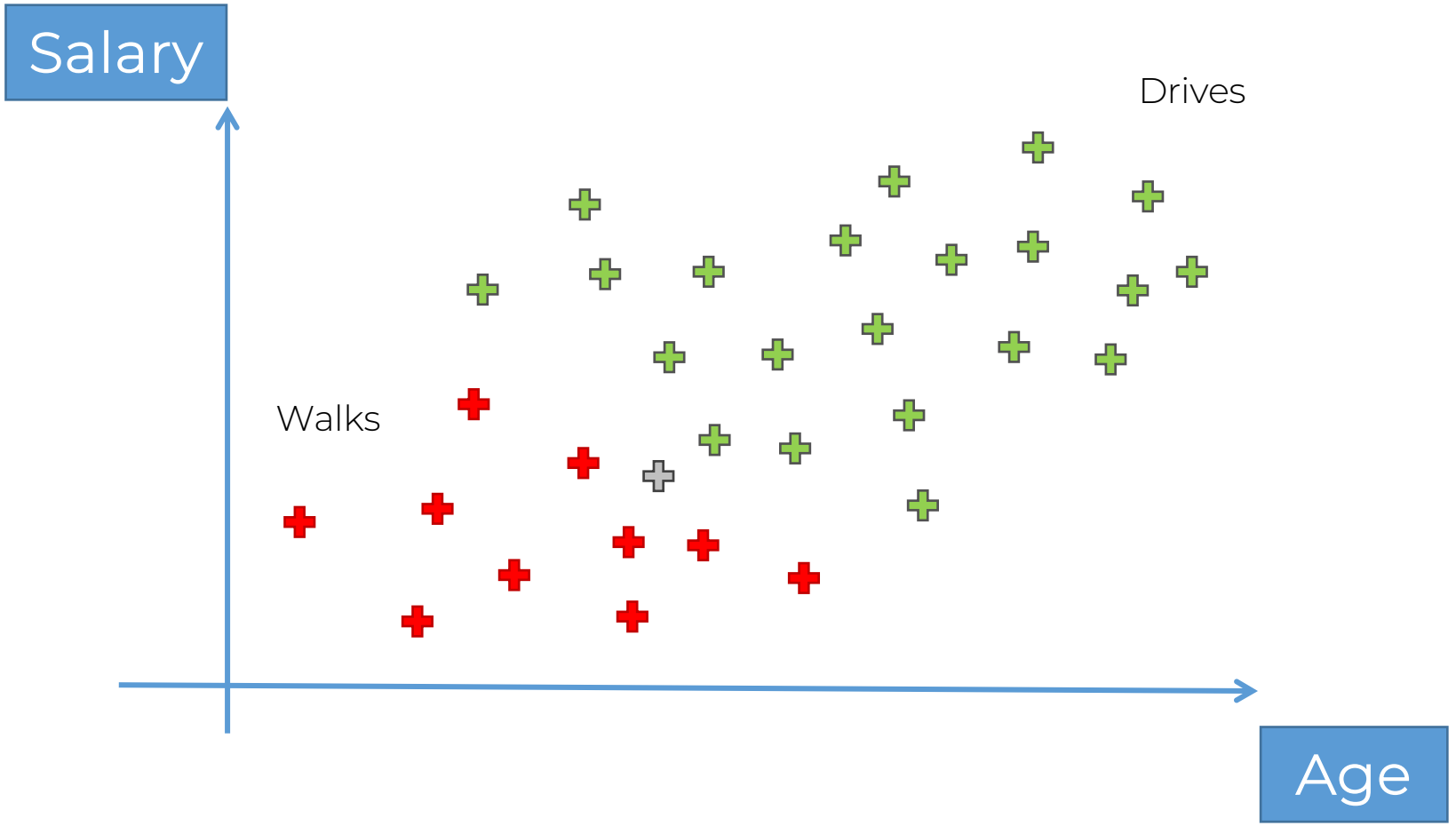
#3 Likelihood

#1 Prior Probability

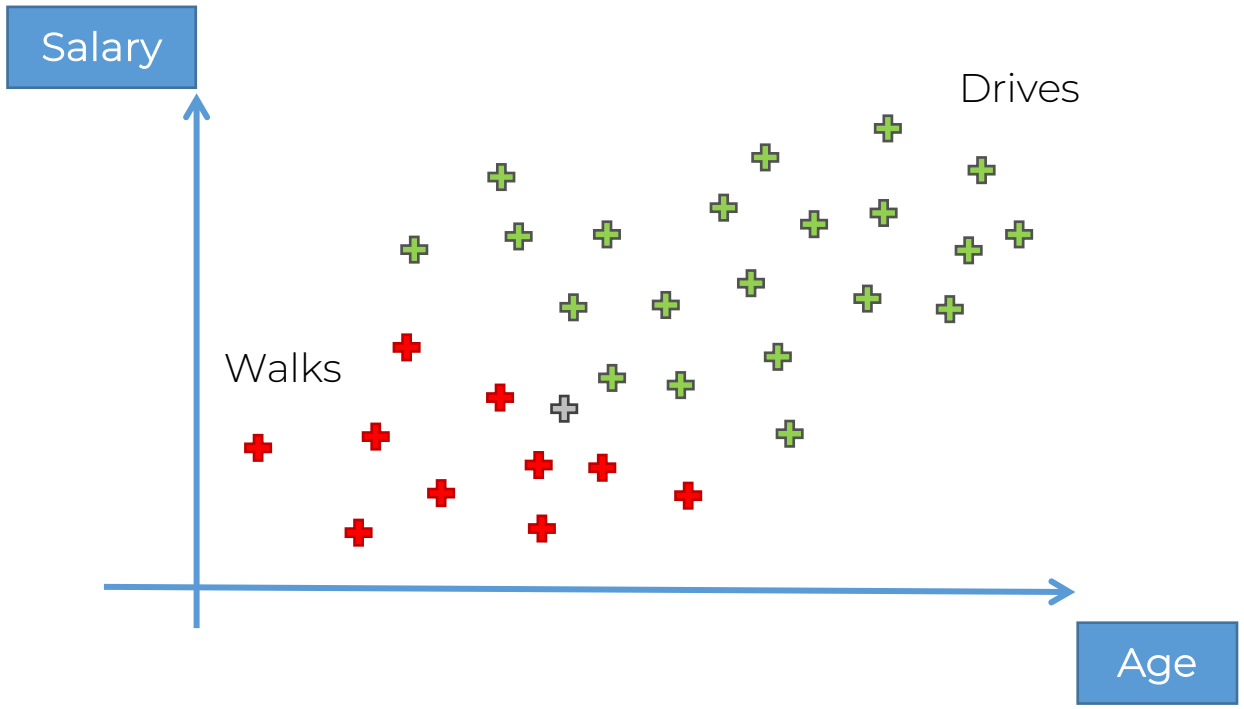
$$P(Drives|X) = \frac{P(X|Drives) * P(Drives)}{P(X)}$$

#2 Marginal Likelihood

Naïve Bayes: Step 2



Naïve Bayes: Step 2



#1. P(Drives)

$$P(Drives) = \frac{Number\ of\ Drivers}{Total\ Observations}$$

$$P(Drives) = \frac{20}{30}$$

Naïve Bayes: Step 2

#4 Posterior Probability

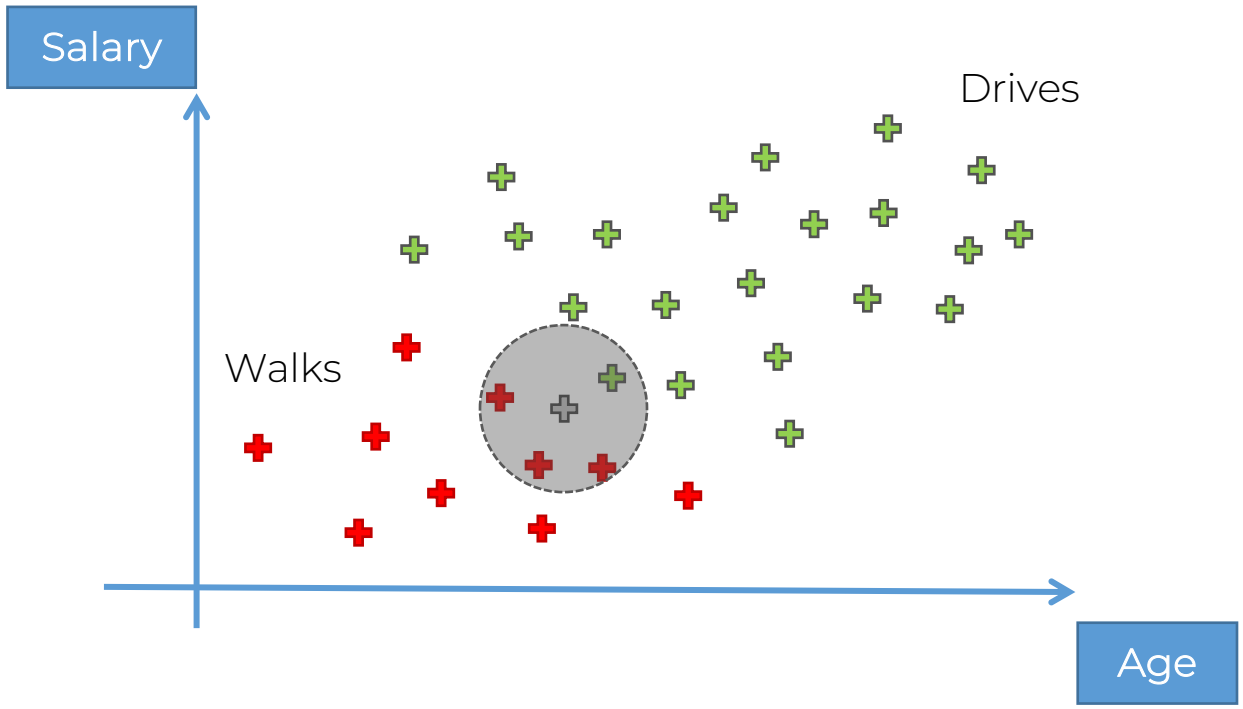
#3 Likelihood

#1 Prior Probability

$$P(Drives|X) = \frac{P(X|Drives) * P(Drives)}{P(X)}$$

#2 Marginal Likelihood

Naïve Bayes: Step 2



#2. P(X)

$$P(X) = \frac{Number\ of\ Similar\ Observations}{Total\ Observations}$$

$$P(X) = \frac{4}{30}$$