Naïve Bayes: Step 2

#4 Posterior Probability

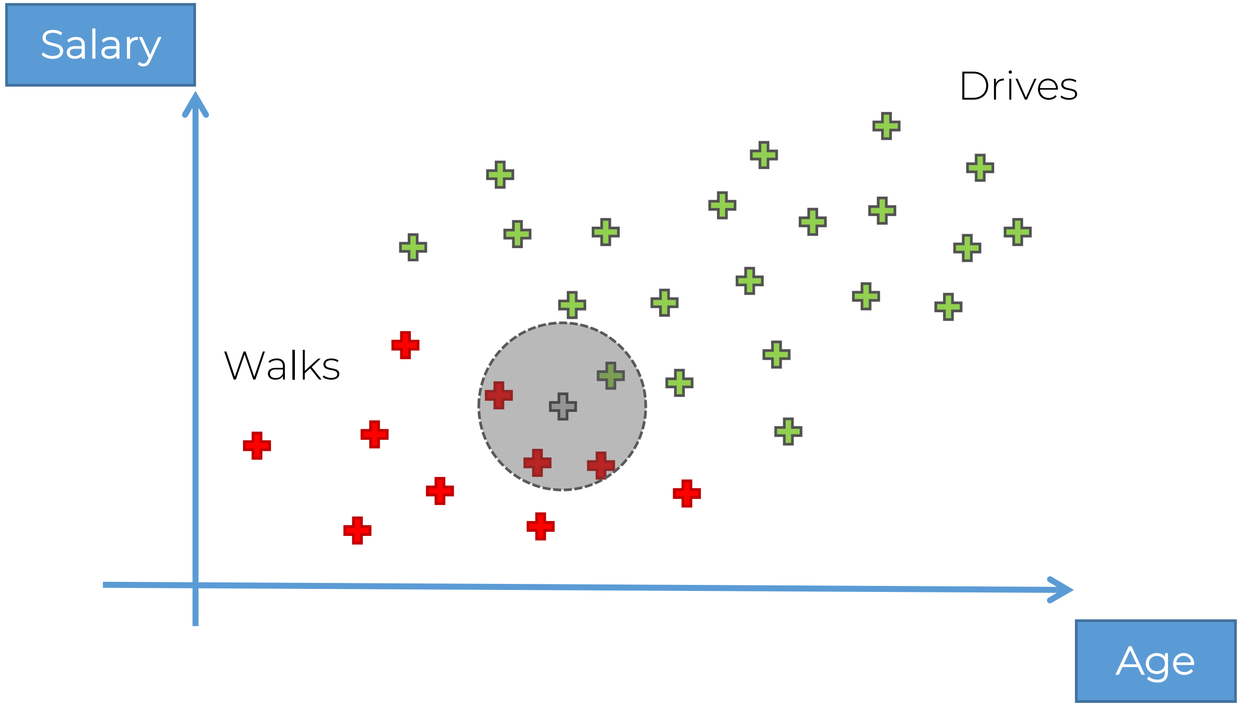
#3 Likelihood

#1 Prior Probability

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

#2 Marginal Likelihood

Naïve Bayes: Step 2



#3. P(X|Drives)

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

$$P(X|Drives) = \frac{1}{20}$$

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

#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

Step 2 – Done.

Naïve Bayes Classifier Additional Comments

Naïve Bayes

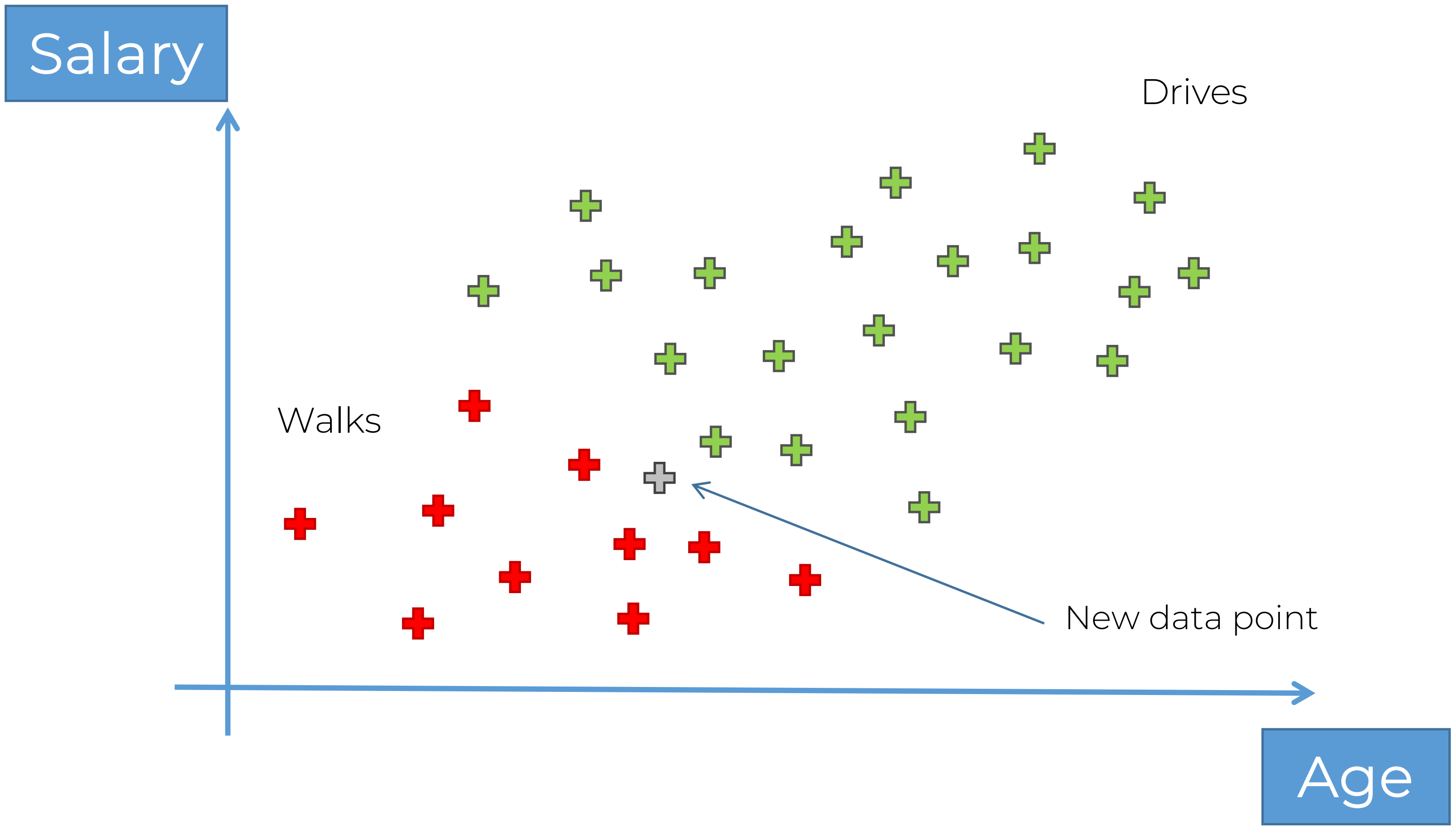
1. Q: Why "Naïve"?
2. P(X)
3. More than 2 features

Naïve Bayes

Q: Why "Naïve"?

A: Independence assumption

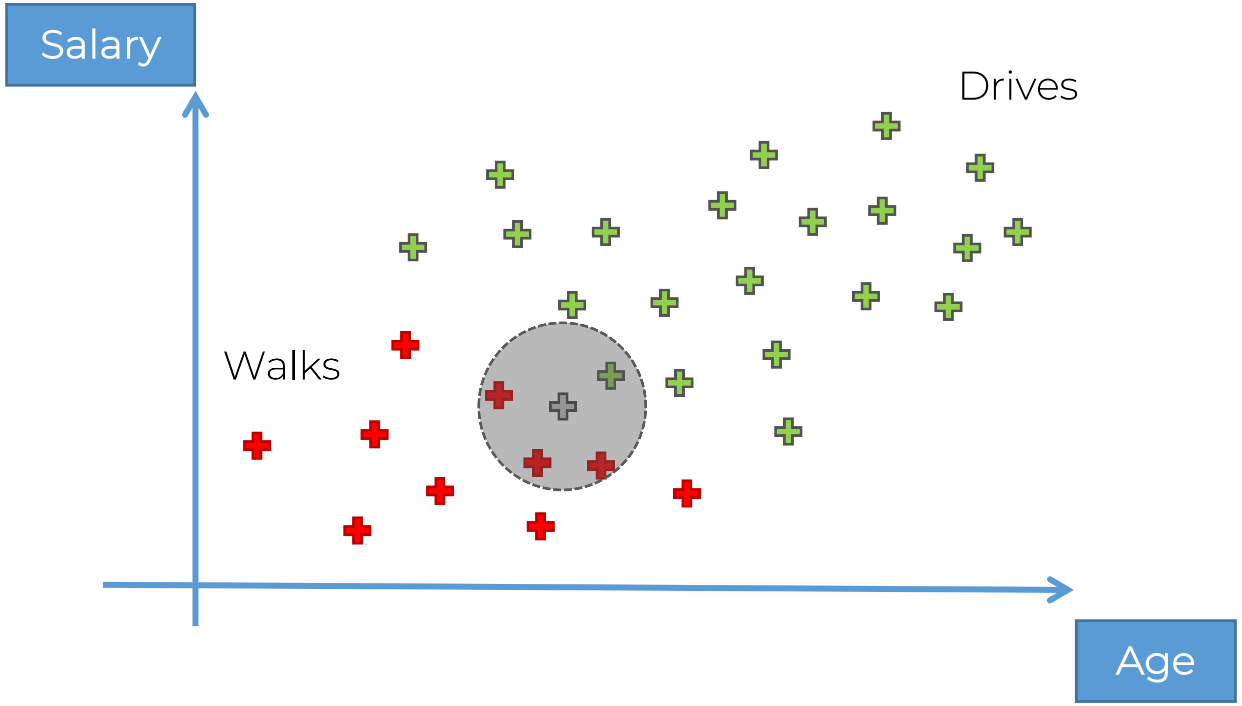
Naïve Bayes



Naïve Bayes

$$P(X)$$

Naïve Bayes: Step 2



#2. P(X)

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

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

NOTE: Same both times

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

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

Step 3

$$\frac{P(X|Walks) * P(Walks)}{\cancel{P(X)}} \quad v.s. \quad \frac{P(X|Drives) * P(Drives)}{\cancel{P(X)}}$$

More than 2 classes

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

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

Decision Tree Intuition

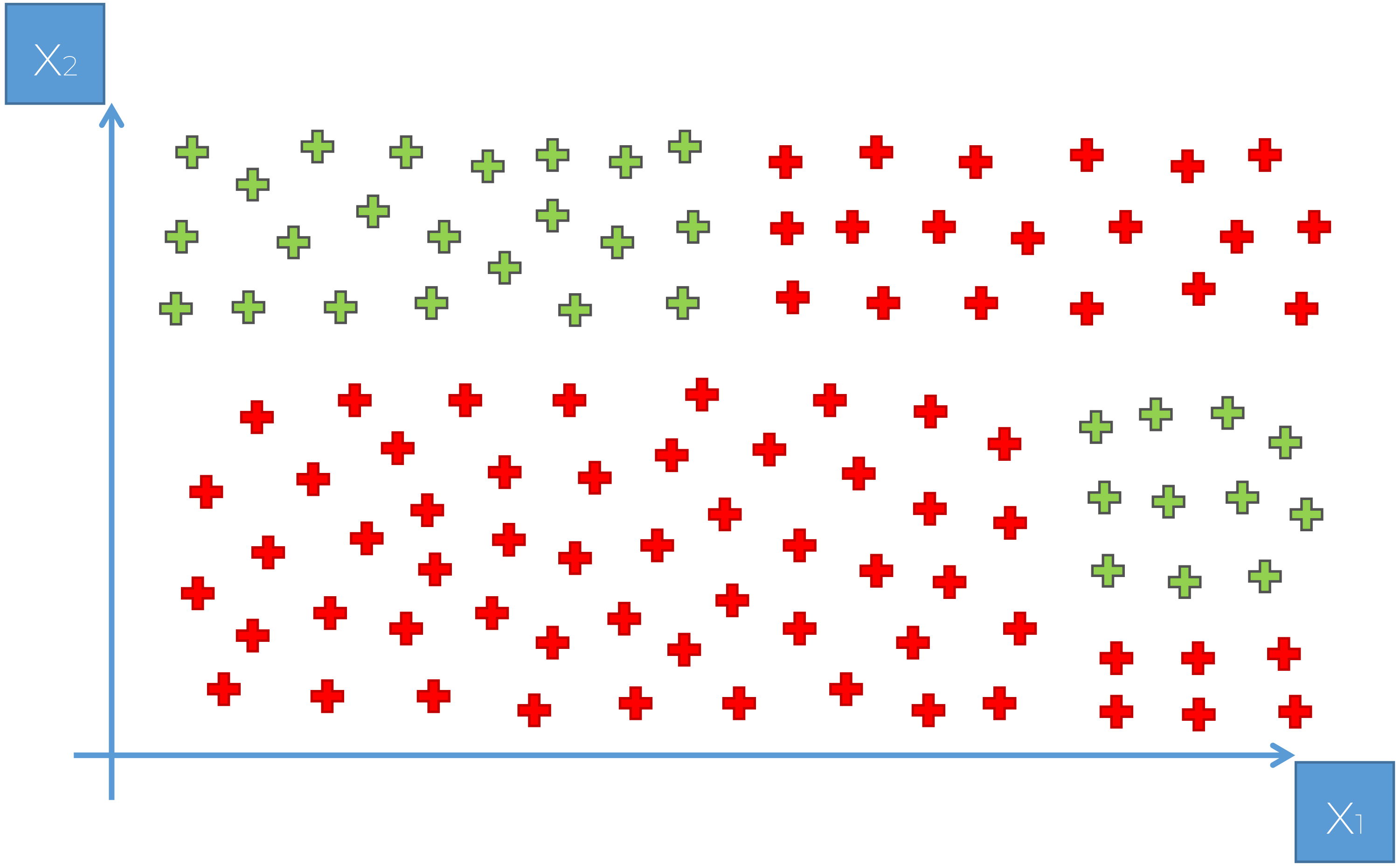
Decision Tree Intuition

CART

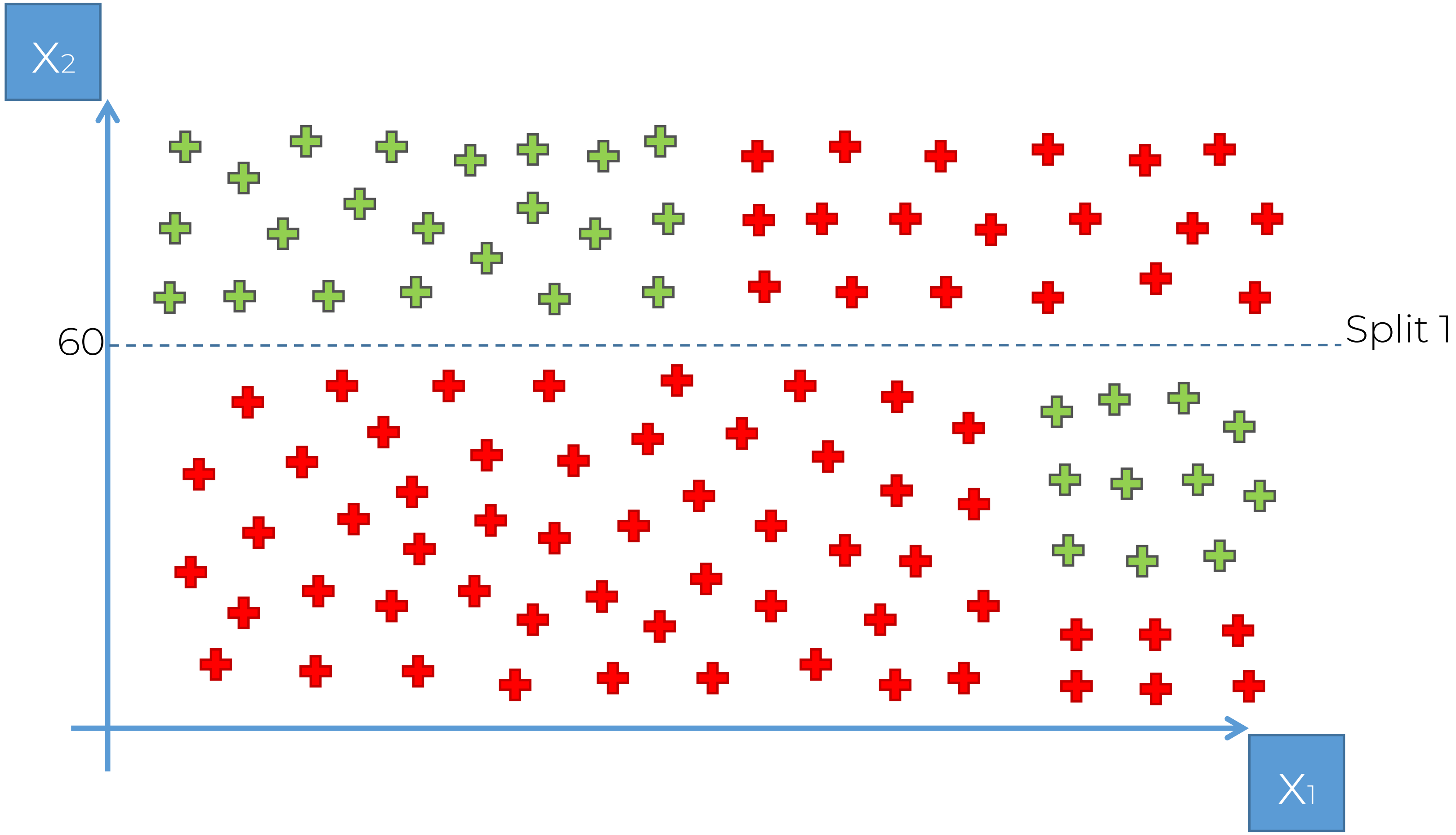
Classification Trees

Regression Trees

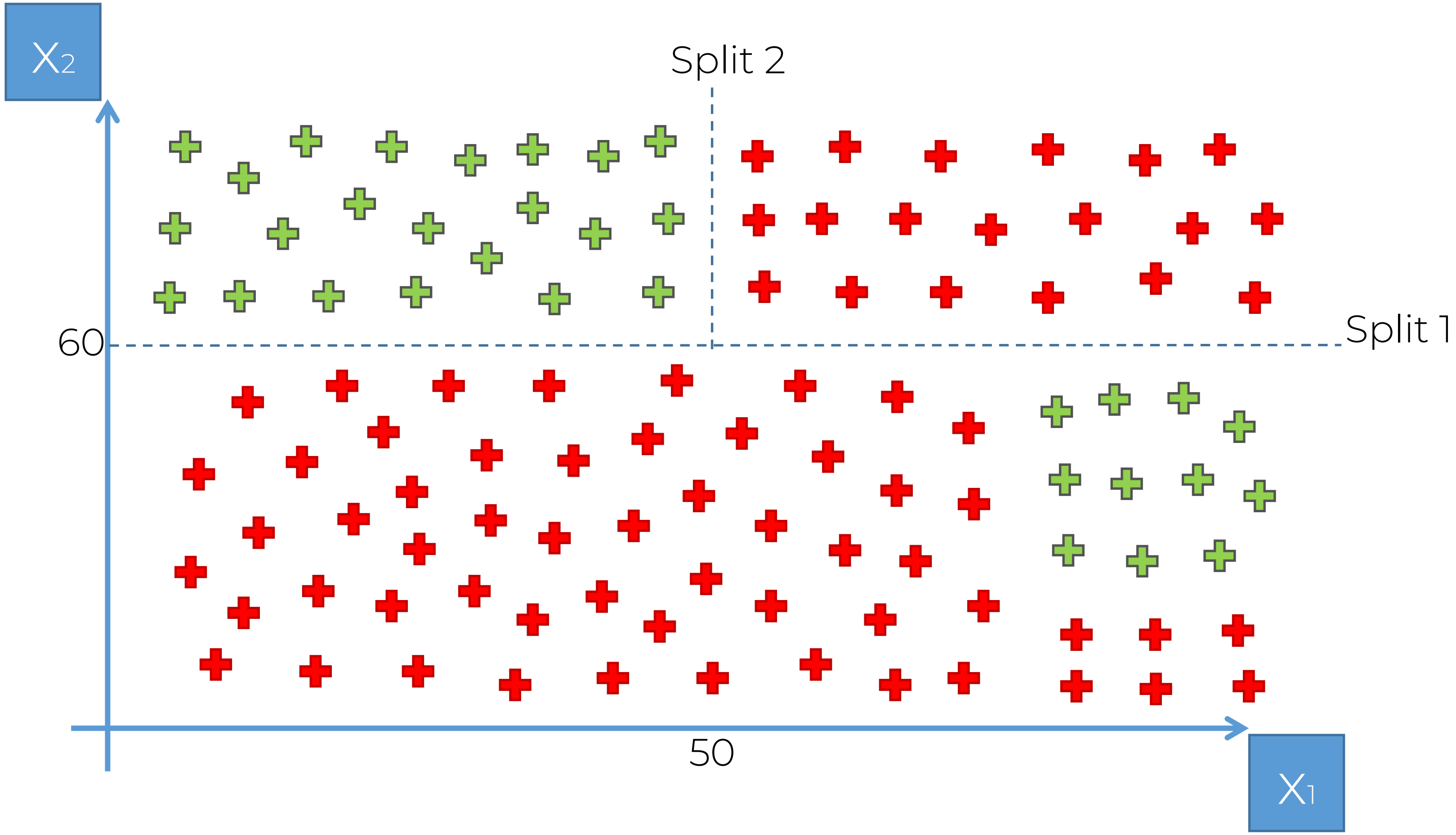
Decision Tree Intuition



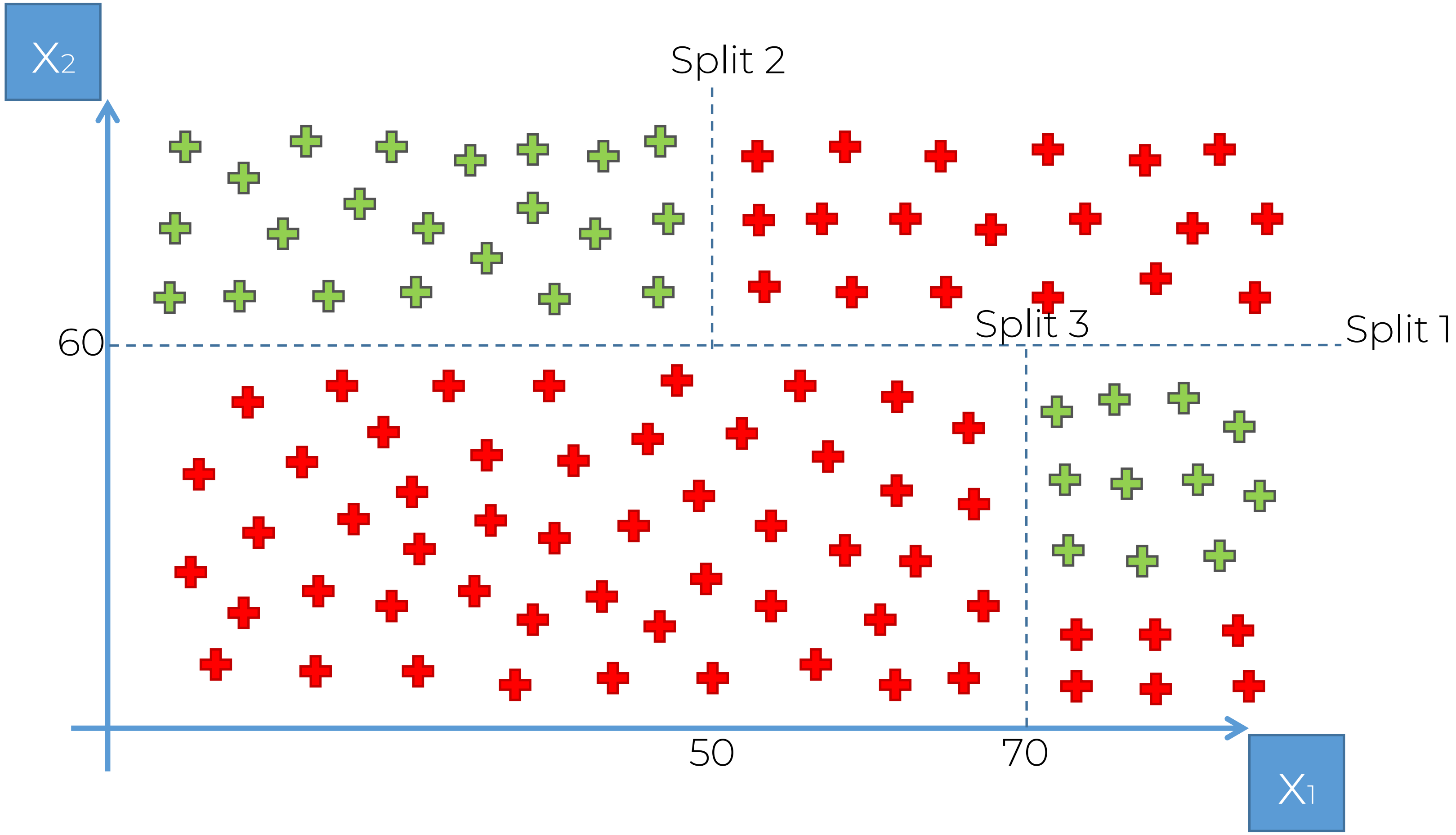
Decision Tree Intuition



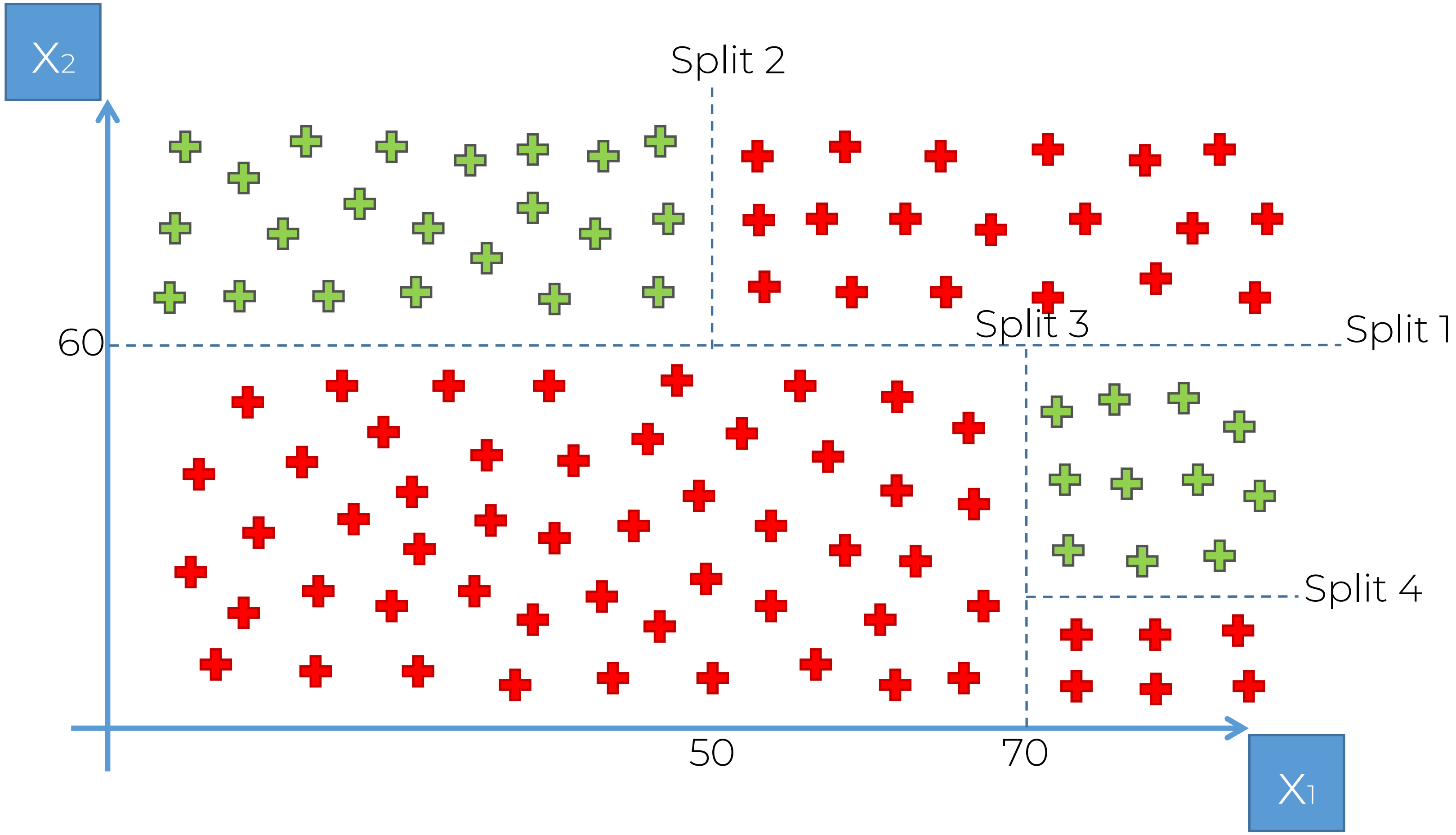
Decision Tree Intuition



Decision Tree Intuition



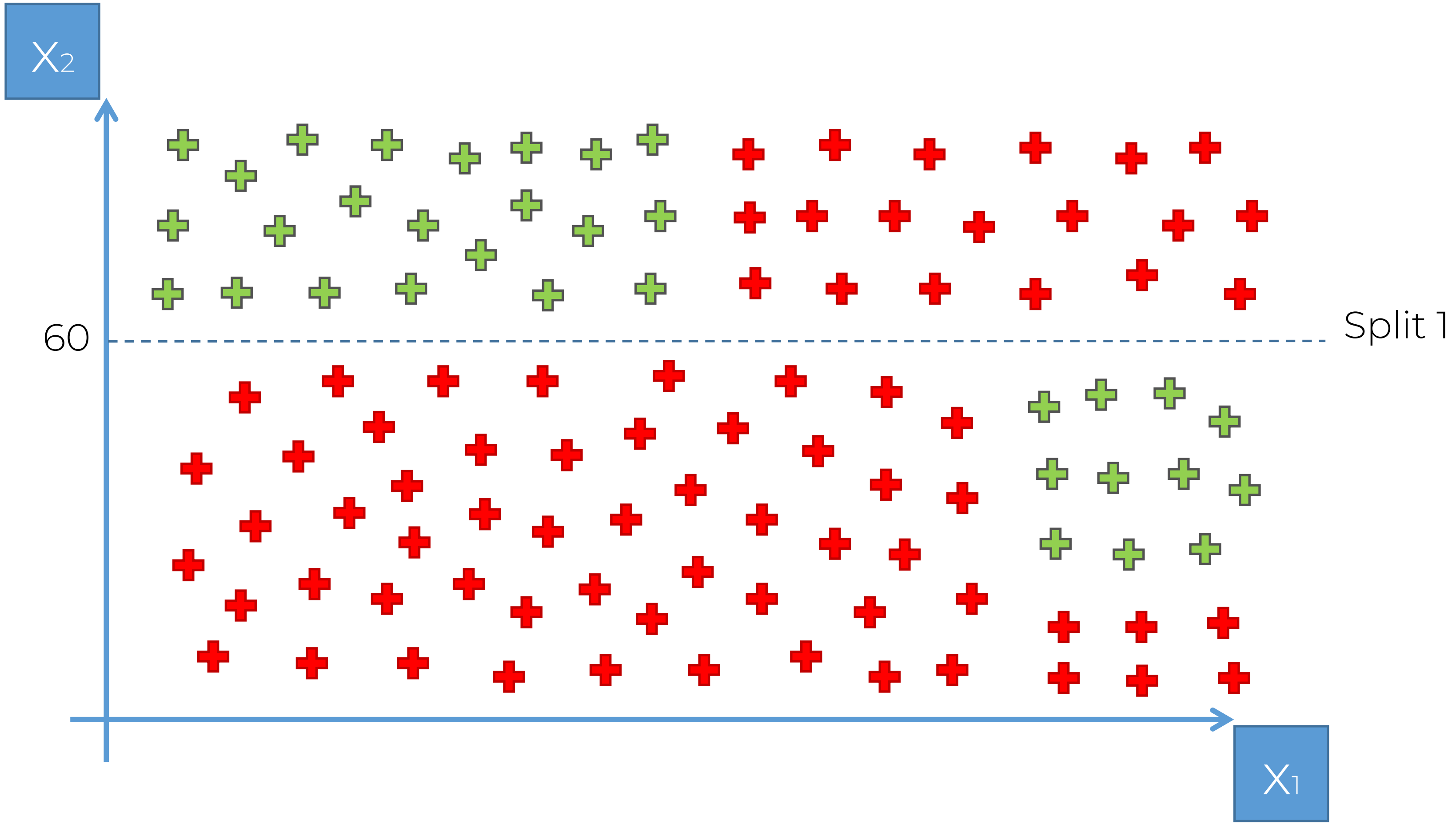
Decision Tree Intuition



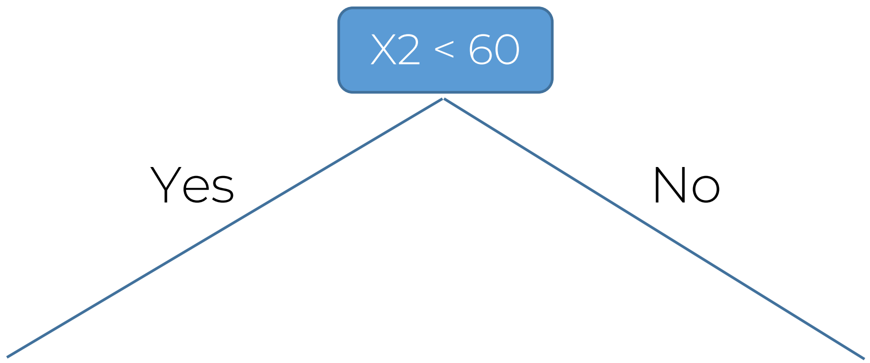
Decision Tree Intuition

Rewind…

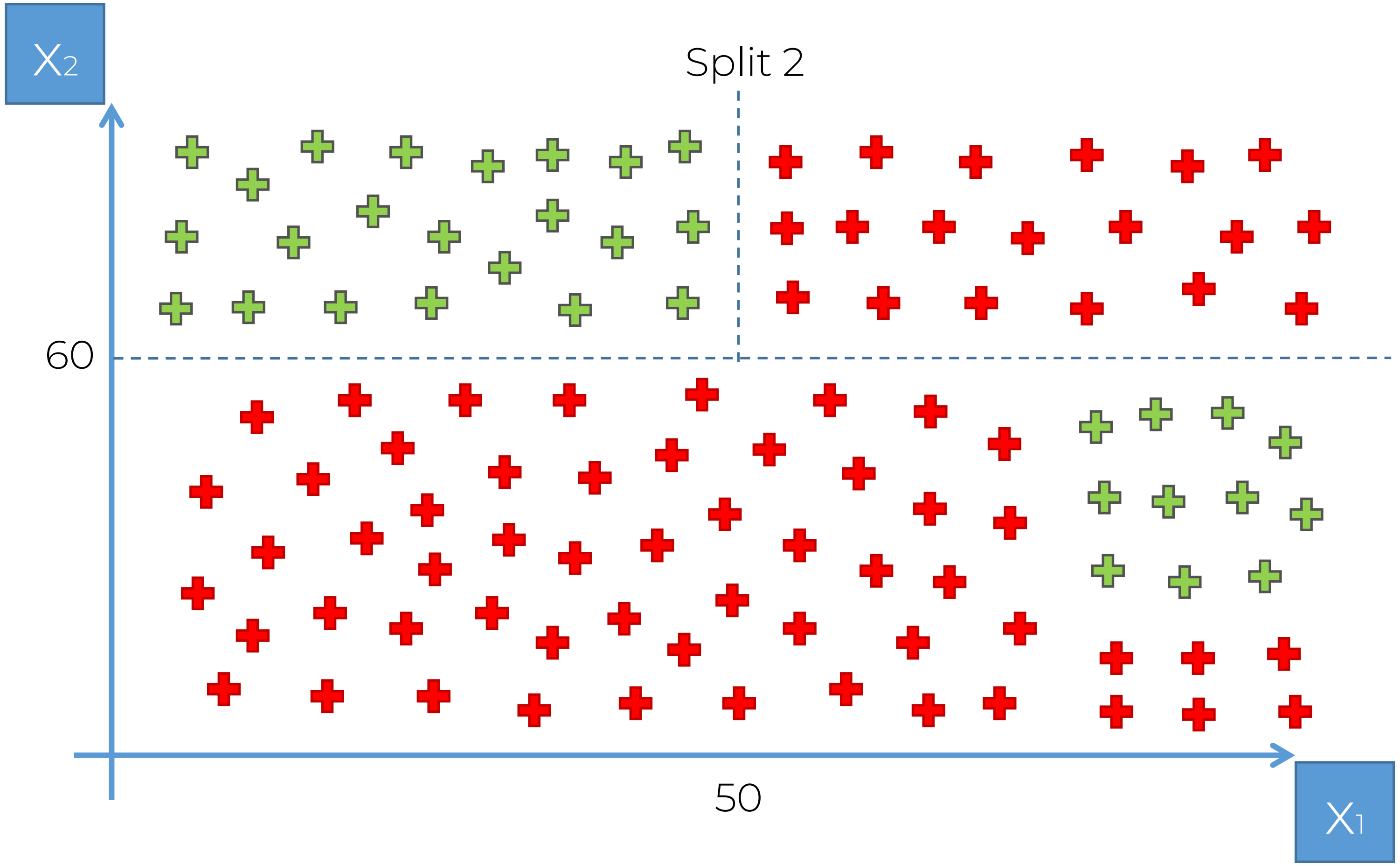
Split 1



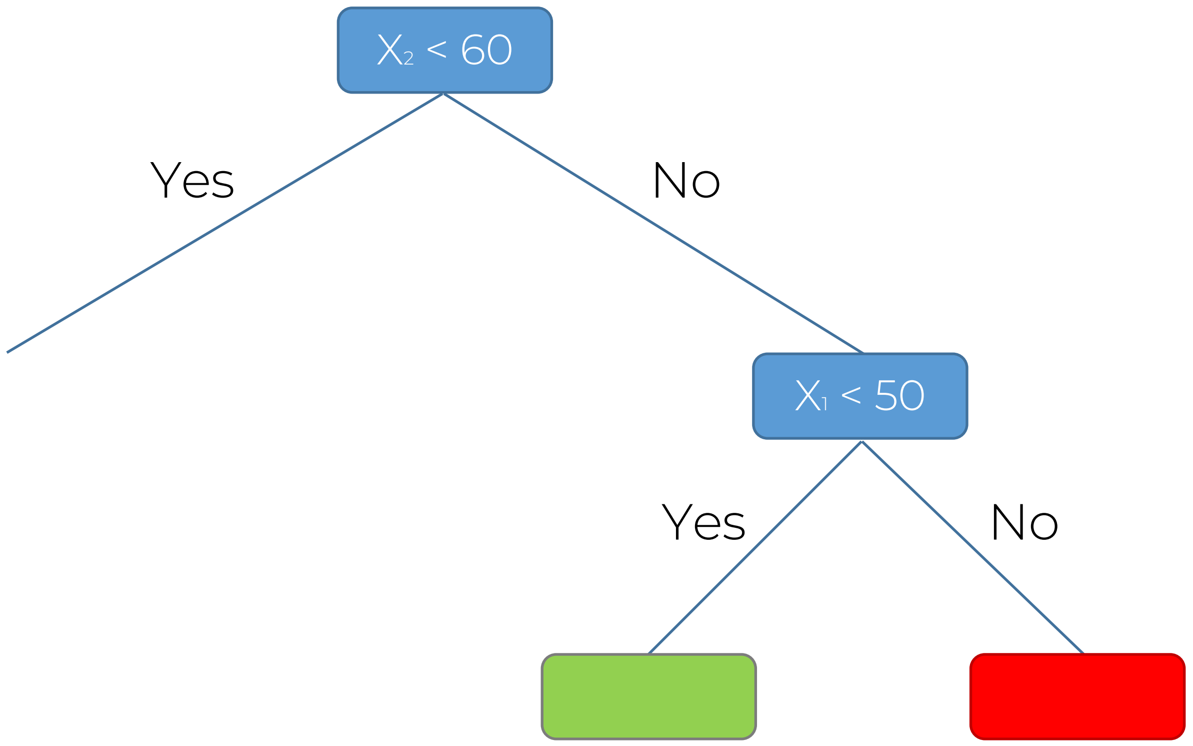
Split 1



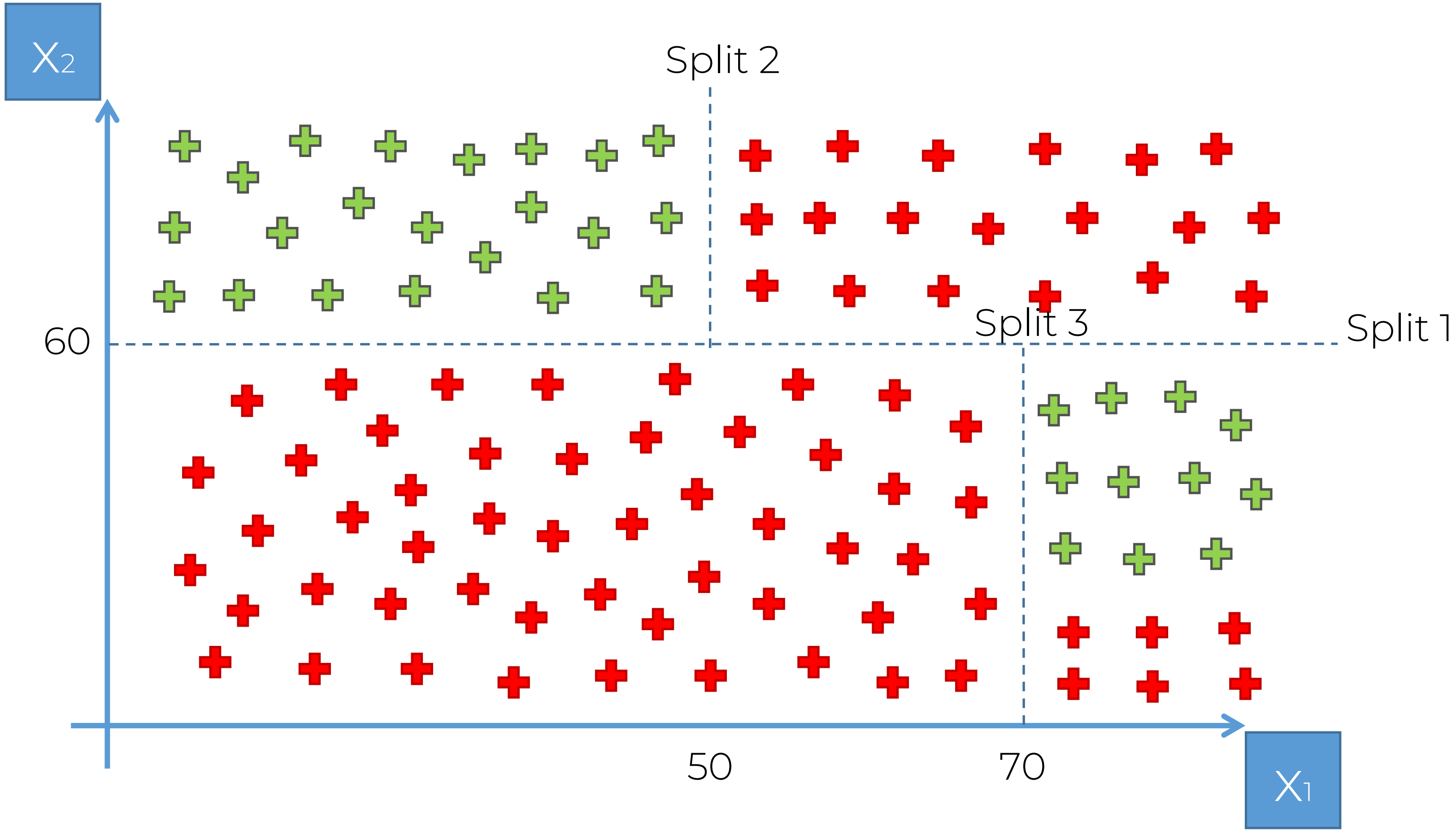
Split 2



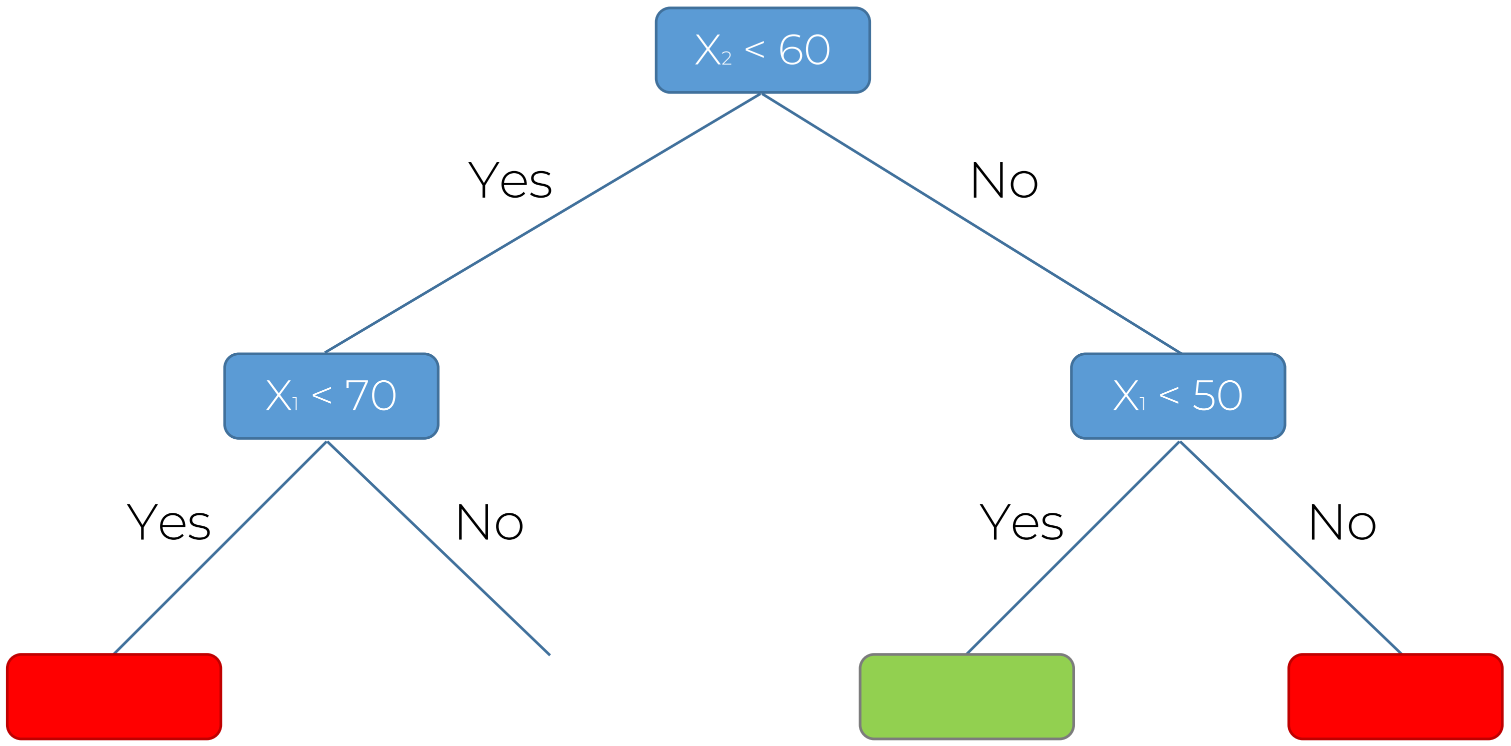
Split 2



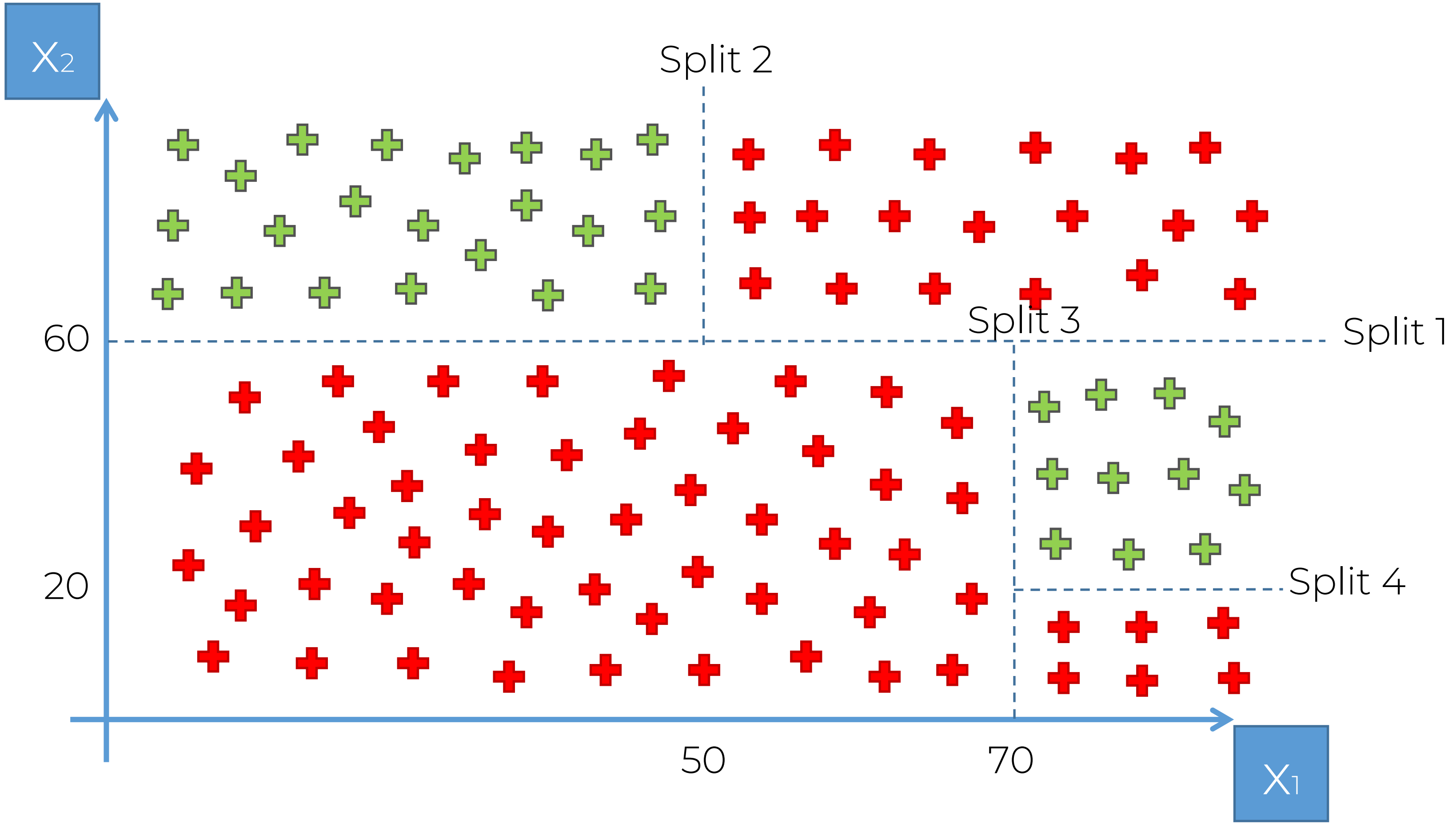
Split 3



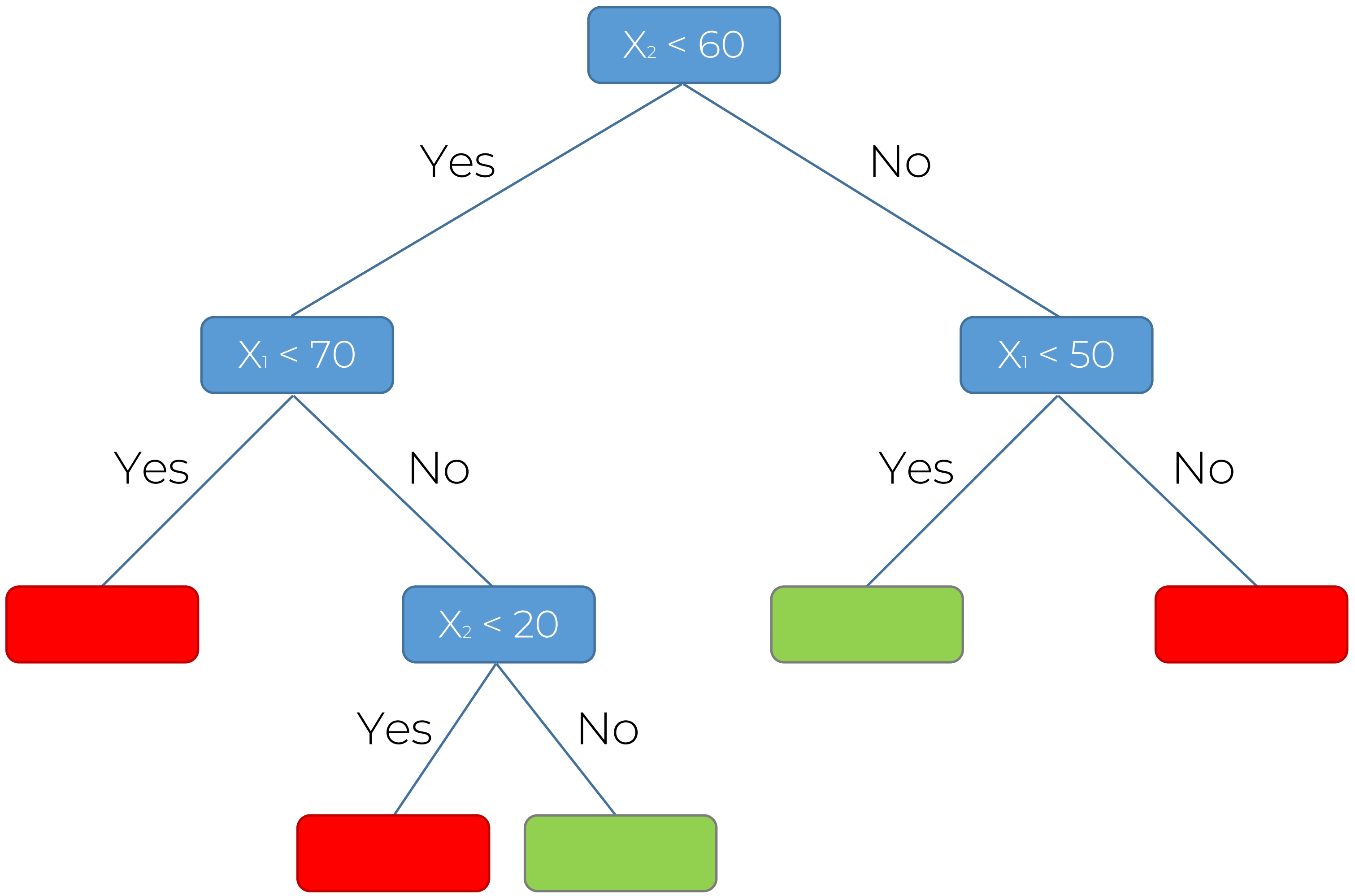
Split 3



Split 4



Split 4



Decision Trees

- Old Method
- Reborn with upgrades
- Random Forest
- Gradient Boosting
- etc.

Random Forest Intuition

Ensemble Learning

Random Forest Intuition

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

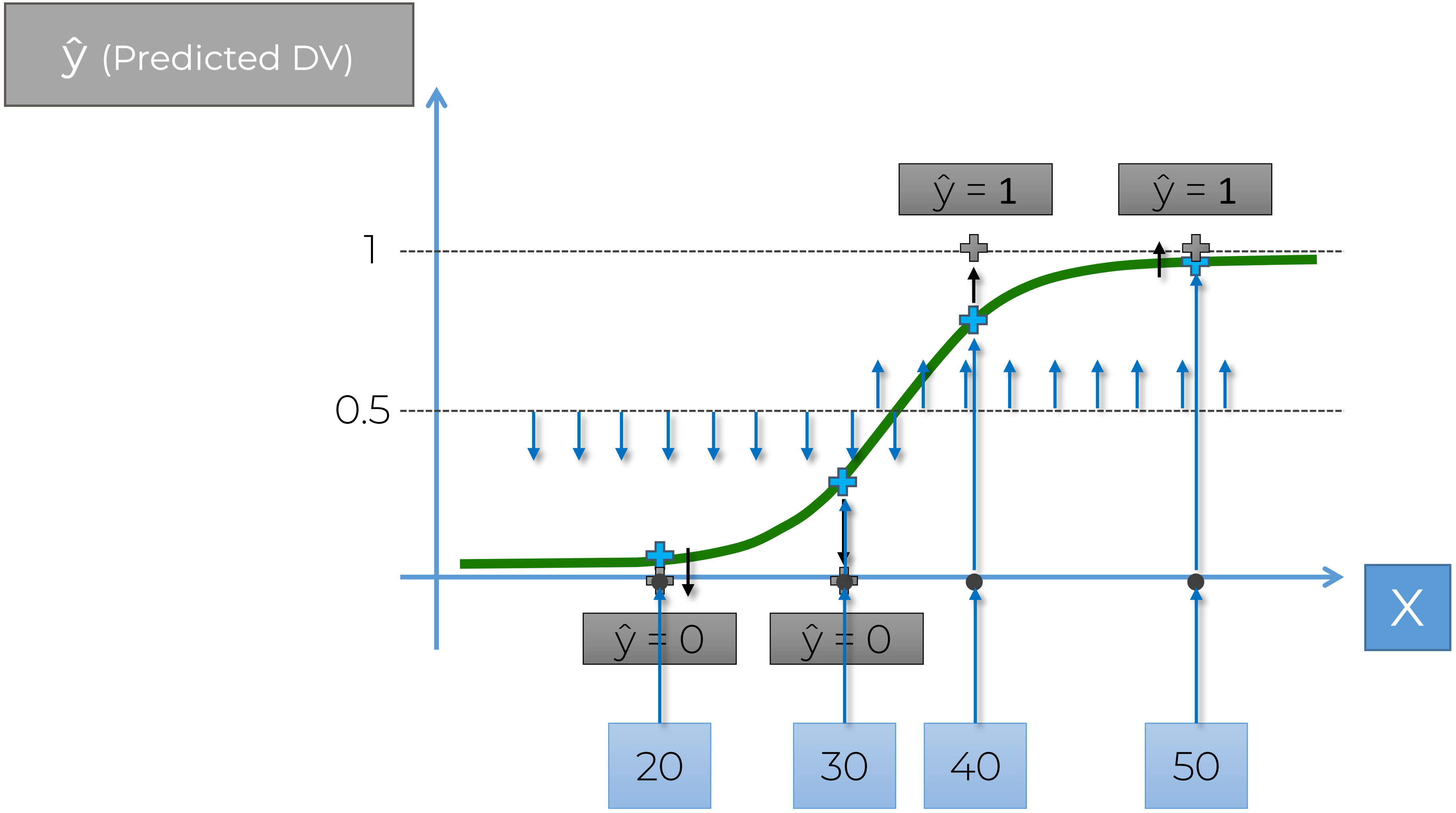
STEP 2: Build the Decision Tree associated to these K data points.

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

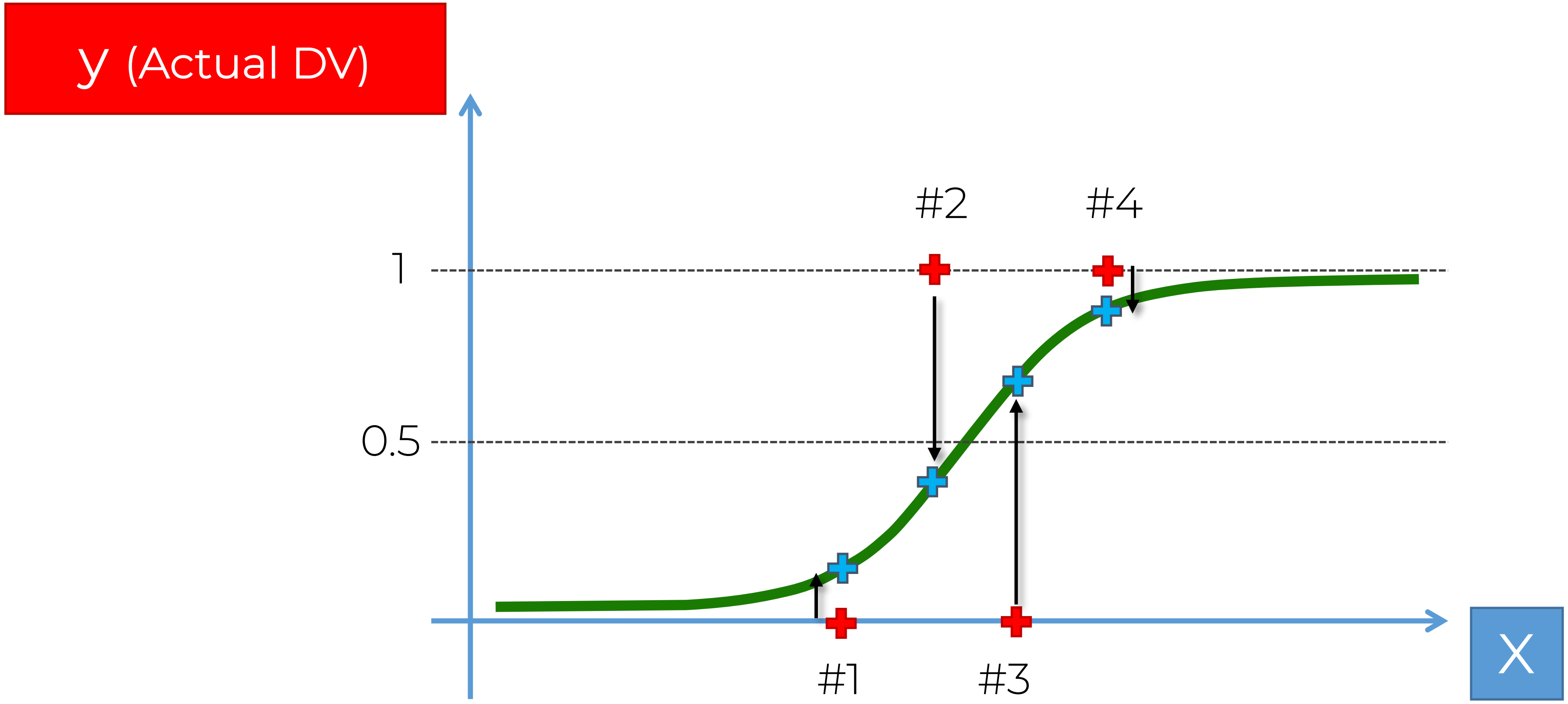
STEP 4: For a new data point, make each one of your Ntree trees predict the category to which the data points belongs, and assign the new data point to the category that wins the majority vote.

False Positives
False Negatives

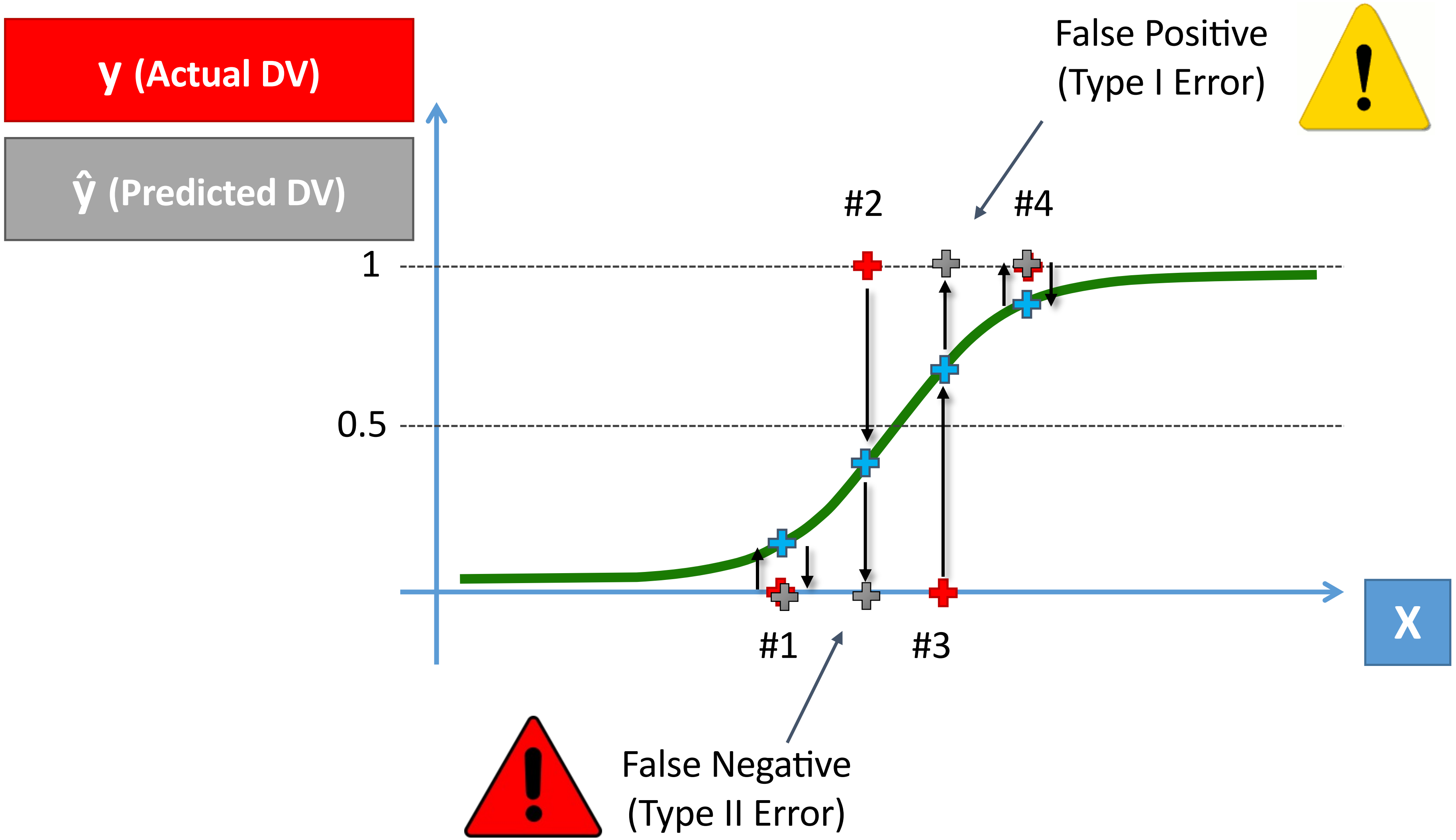
False Positives & Negatives



False Positives & Negatives



False Positives & Negatives



Fin.

Confusion Matrix & Accuracy

Confusion Matrix & Accuracy

		Prediction	
		NEG	POS
Actual	NEG	TRUE NEG	FALSE POS
	POS	FALSE NEG	TRUE POS

Type II Error
(False Negatives)

Type I Error
(False Positives)

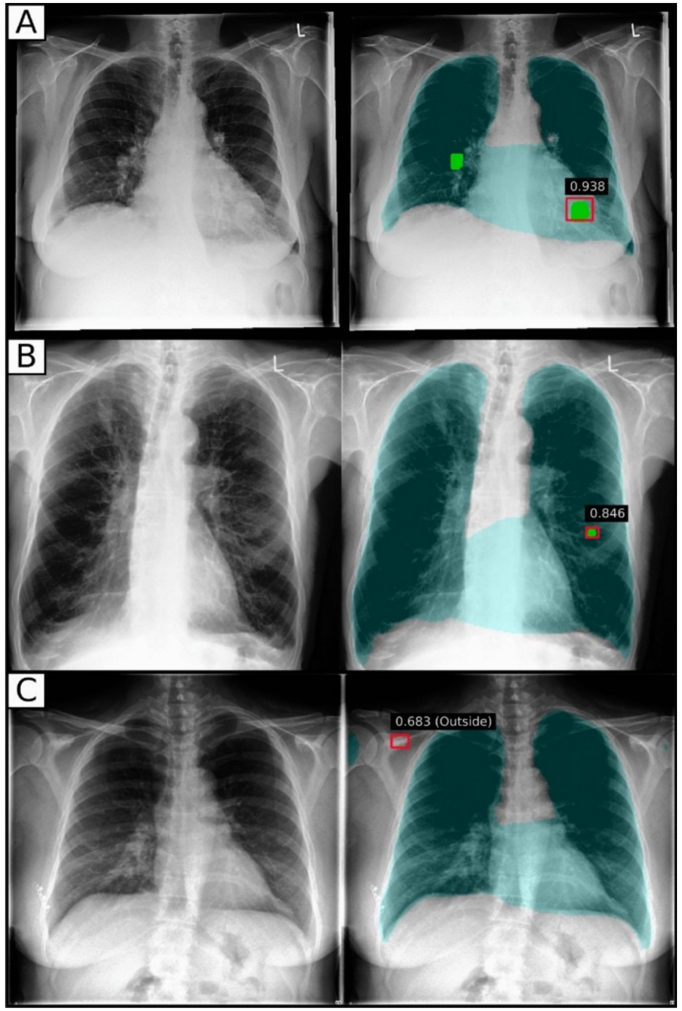


Image source: nature.com

Confusion Matrix & Accuracy

		Prediction	
		NEG	POS
Actual	NEG	43	12
	POS	4	41

Type II Error
(False Negatives)

Type I Error
(False Positives)

Accuracy Rate and Error Rate:

$$AR = \frac{Correct}{Total} = \frac{TN + TP}{Total} = \frac{84}{100} = 84\%$$

$$ER = \frac{Incorrect}{Total} = \frac{FP + FN}{Total} = \frac{16}{100} = 16\%$$

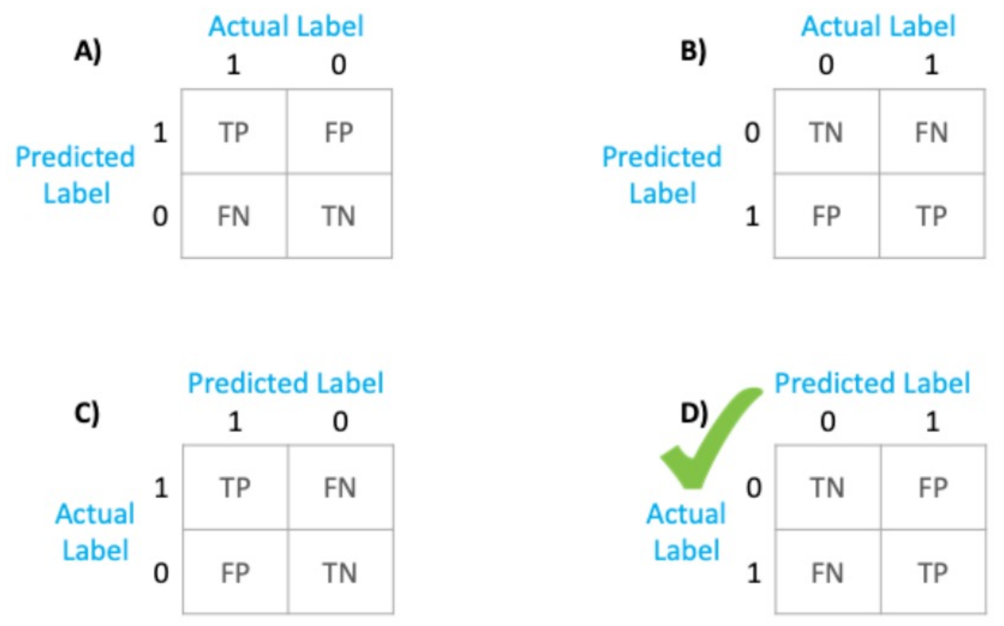
Additional Reading

Understanding the Confusion Matrix from Scikit learn

Samarth Agrawal (2021)

Link:

https://towardsdatascience.com/understanding-the-confusion-matrix-from-scikit-learn-c51d88929c79



Referring back to original image. Option D is the default output. Image by Author

Accuracy Paradox

Accuracy Paradox

		ŷ (Predicted DV)	
		0	1
y (Actual DV)	0	9,700	150
	1	50	100

<u>Scenario 1:</u>

Accuracy Rate = Correct / Total
AR = 9,800/10,000 = 98%

Accuracy Paradox

		$\hat{y}$ (Predicted DV)	
		0	1
y (Actual DV)	0	9,850	0
	1	150	0

Scenario 1:

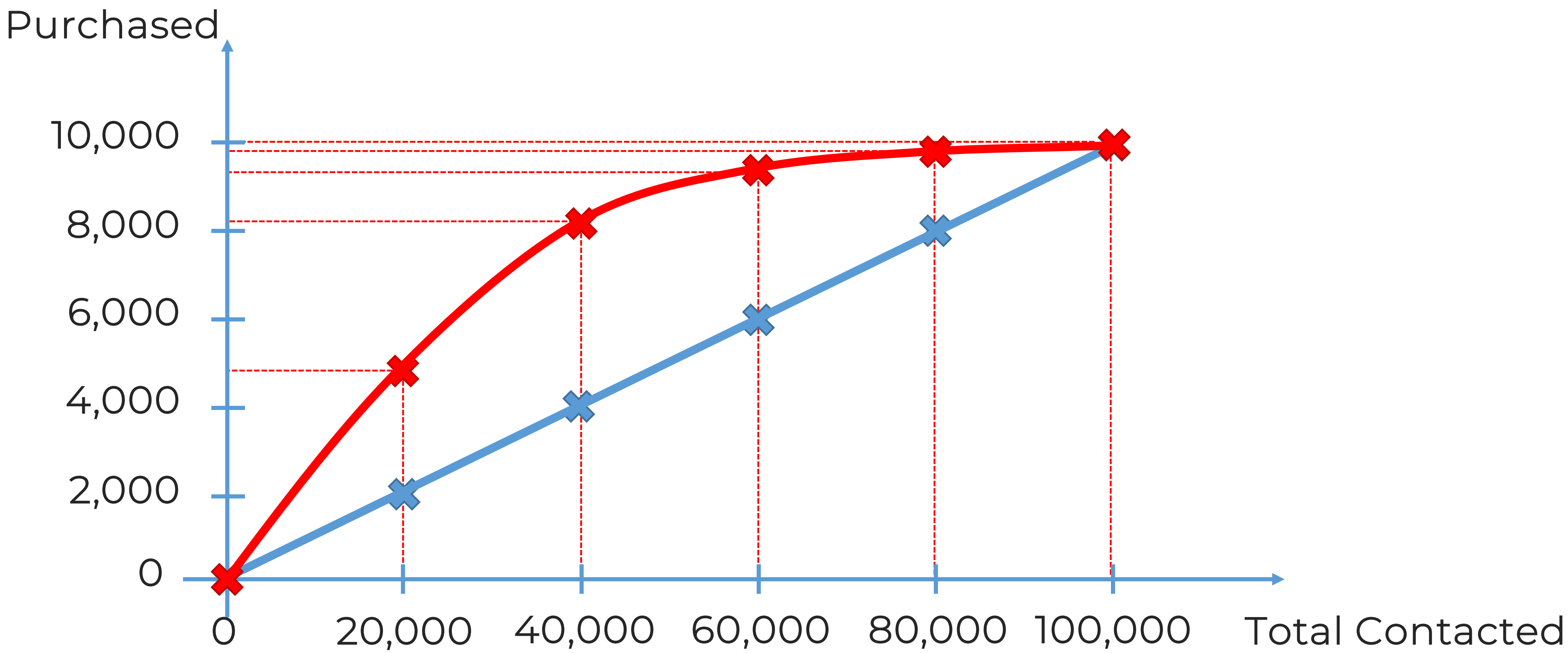
Accuracy Rate = Correct / Total
AR = 9,800/10,000 = 98%

Scenario 2:

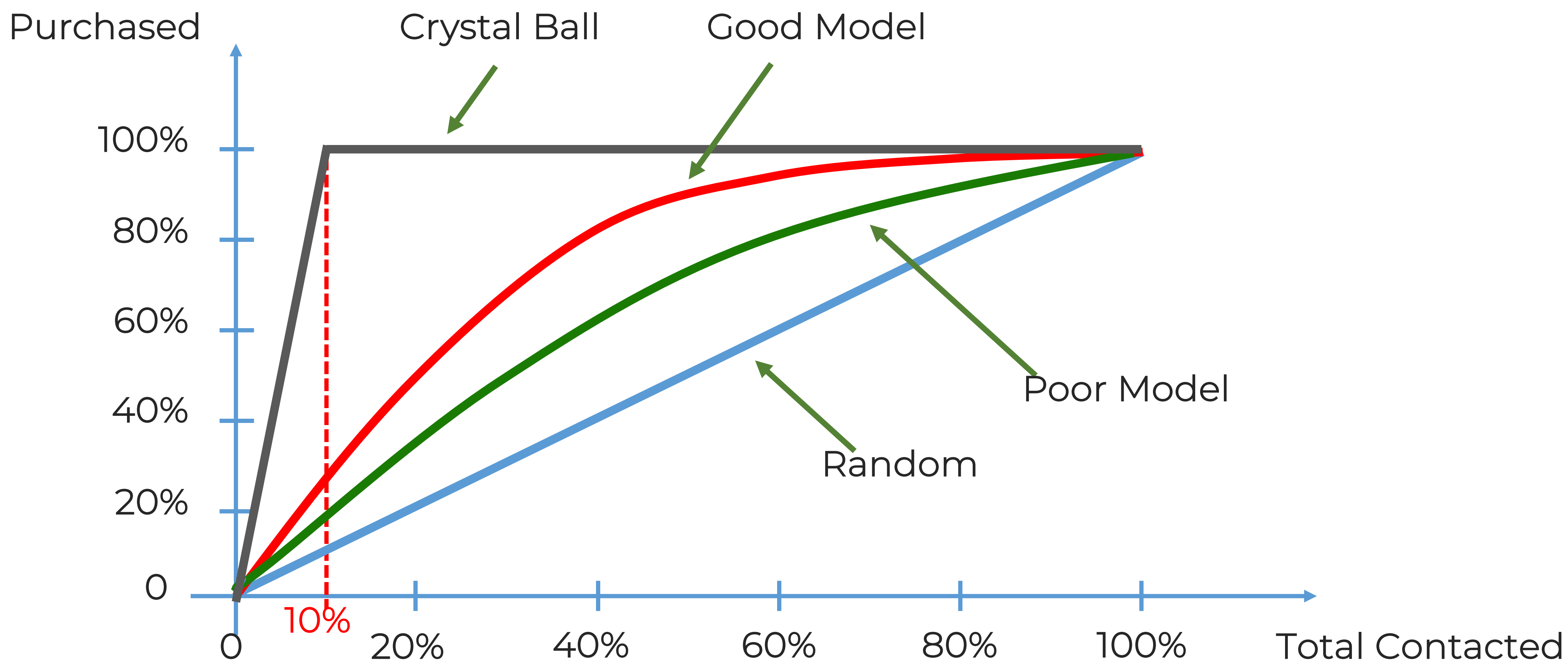
Accuracy Rate = Correct / Total
AR = 9,850/10,000 = 98.5%

Cumulative Accuracy Profile

CAP


Purchased
10,000
8,000
6,000
4,000
2,000
0
0
20,000
40,000
60,000
80,000
100,000
Total Contacted

CAP


Purchased
Crystal Ball
Good Model
100%
80%
60%
40%
20%
0
Poor Model
Random
10%
0
20%
40%
60%
80%
100%
Total Contacted

CAP

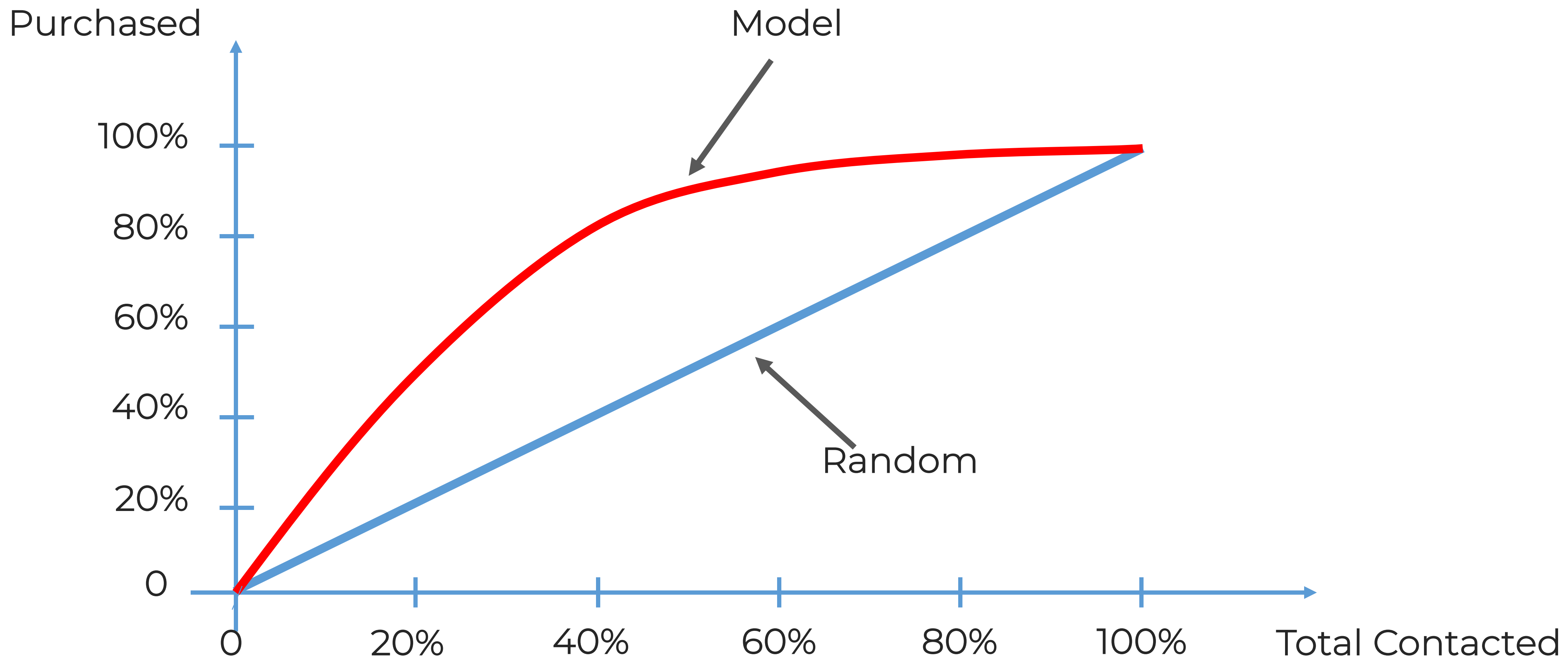
Note:

CAP = Cumulative Accuracy Profile

$\neq$

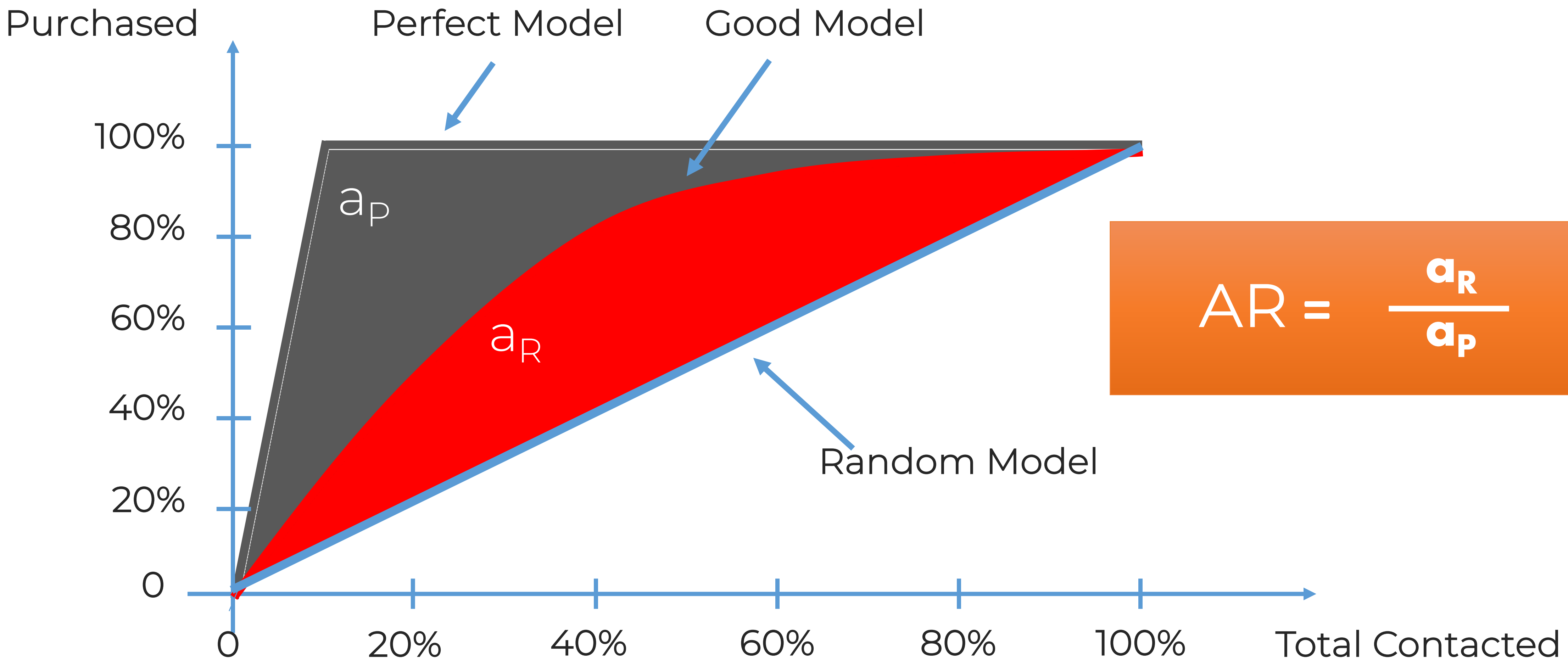
ROC = Receiver Operating Characteristic

CAP


Purchased
Model
100%
80%
60%
40%
20%
0
Random
0
20%
40%
60%
80%
100%
Total Contacted

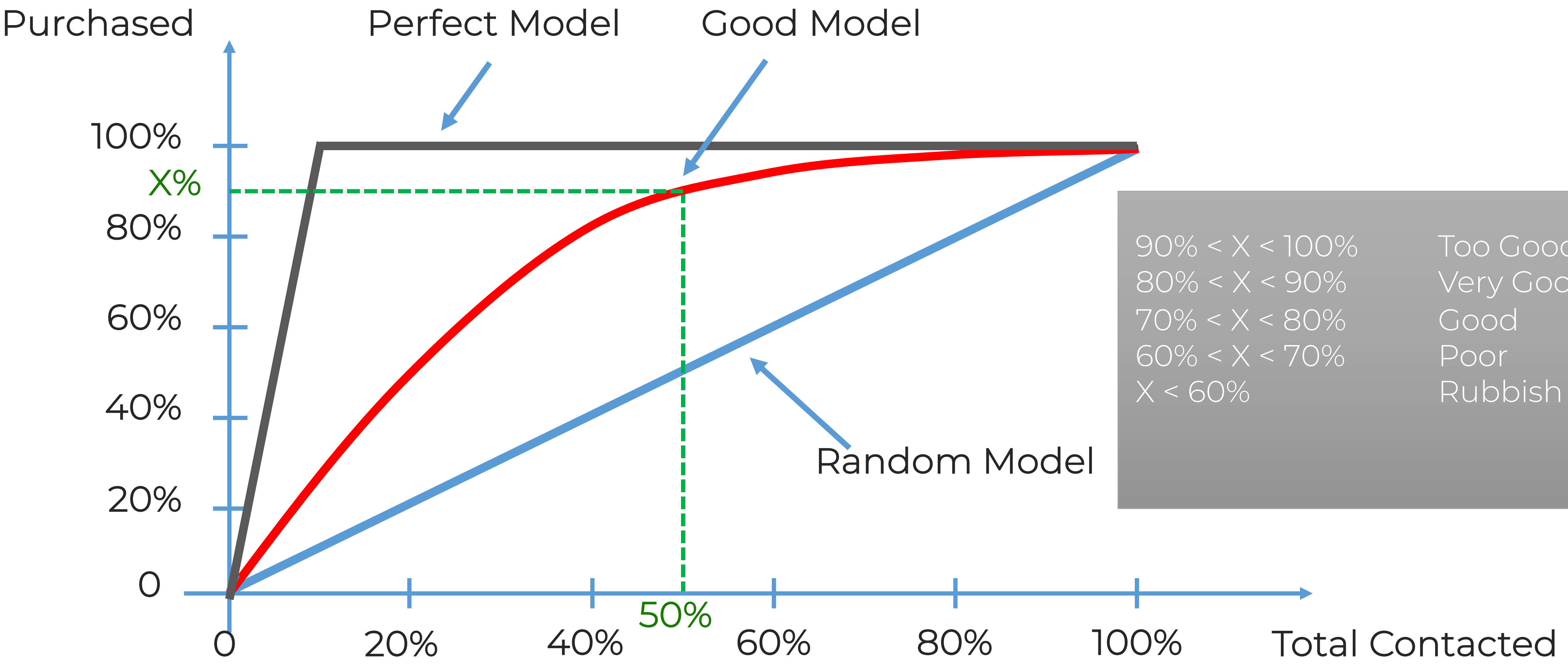
CAP Analysis

CAP Analysis



$$AR = \frac{a_R}{a_P}$$

CAP Analysis



90% < X < 100%	Too Good
80% < X < 90%	Very Good
70% < X < 80%	Good
60% < X < 70%	Poor
X < 60%	Rubbish

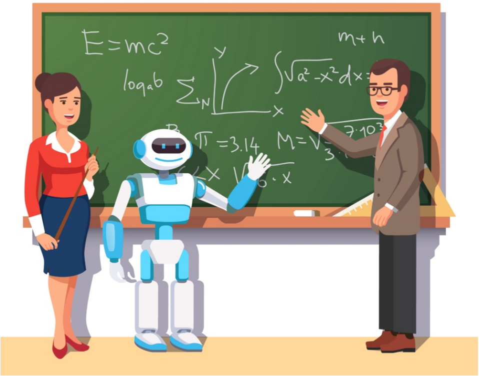
Clustering

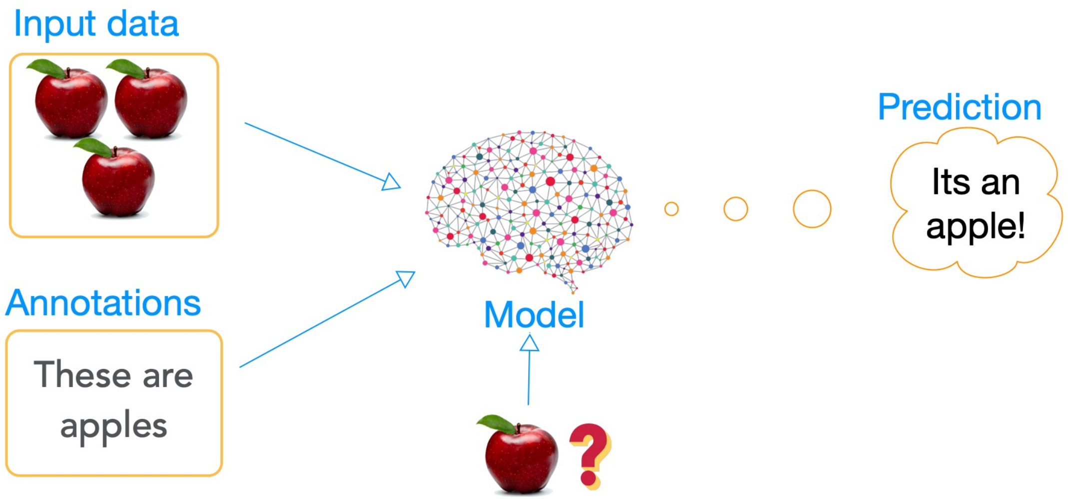
What is Clustering?

Clustering – grouping unlabelled data

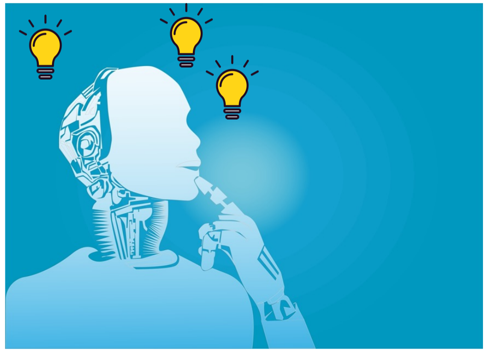
What is Clustering?

Supervised Learning
(e.g. Regression, Classification)





Unsupervised Learning
(e.g. Clustering)



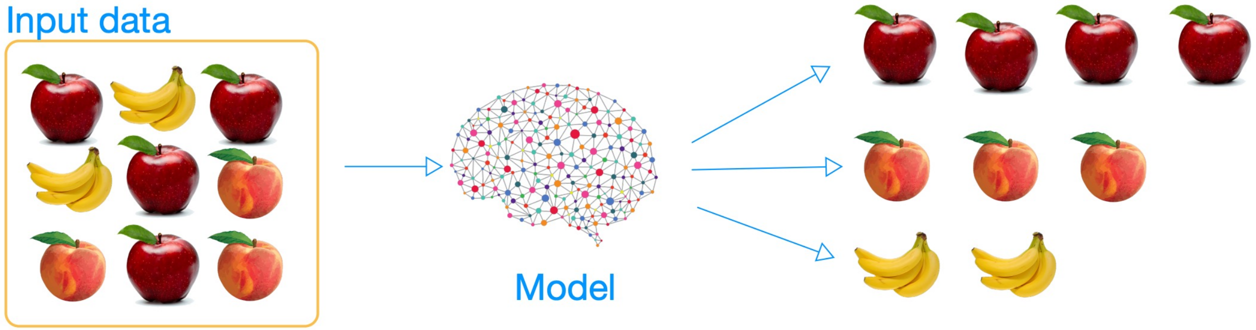
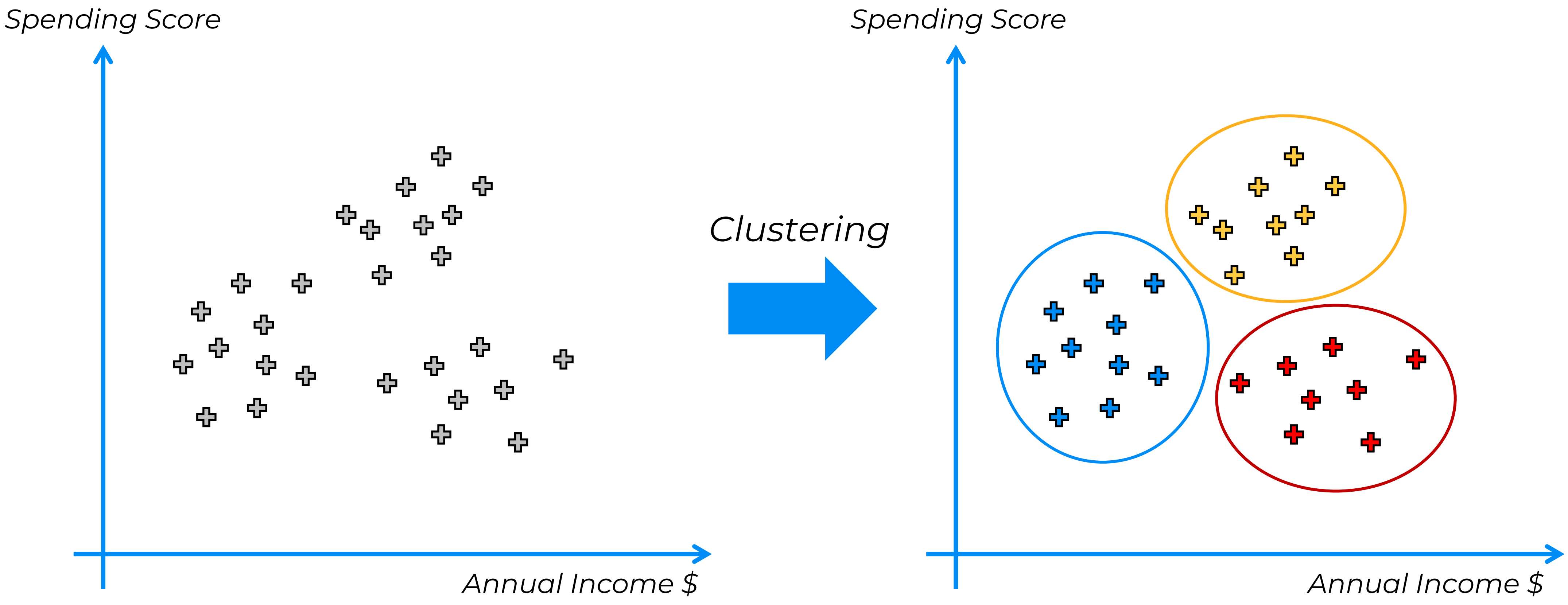


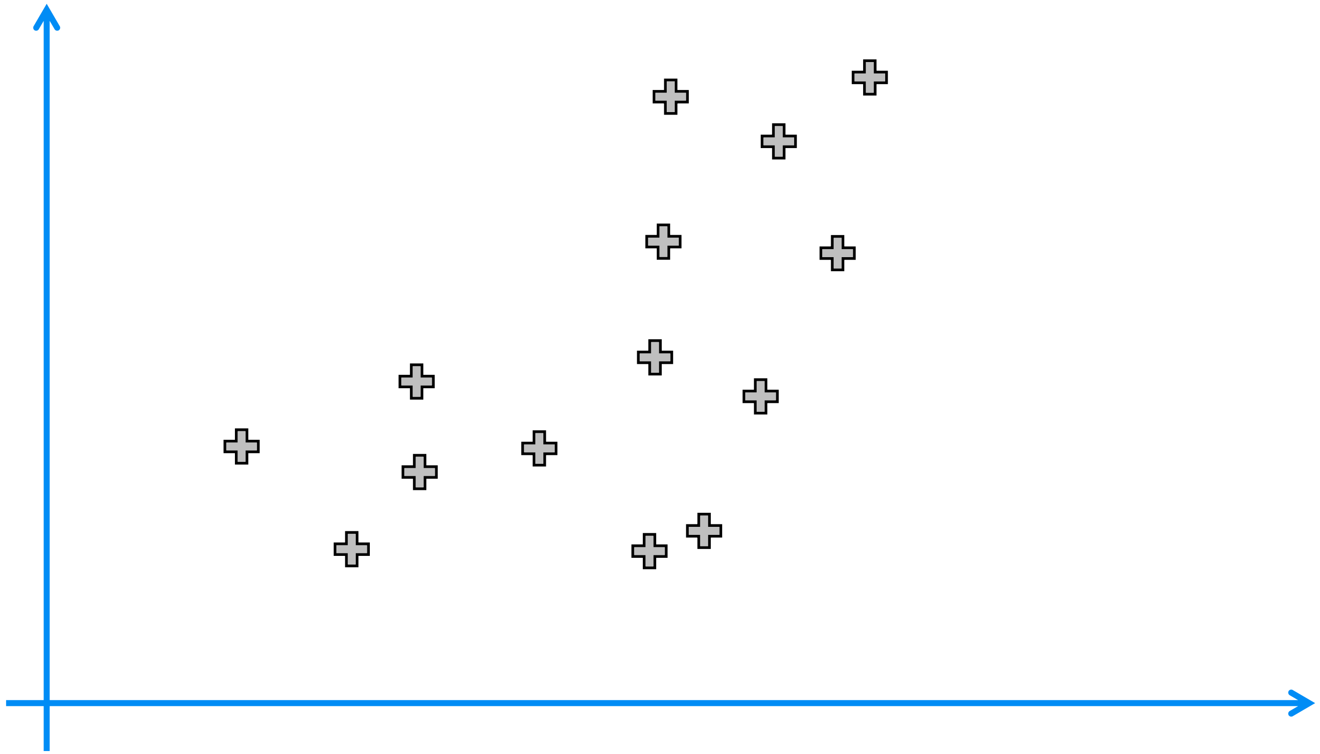
Image source: mdpi.com/2073-8994/10/12/734

What is Clustering?

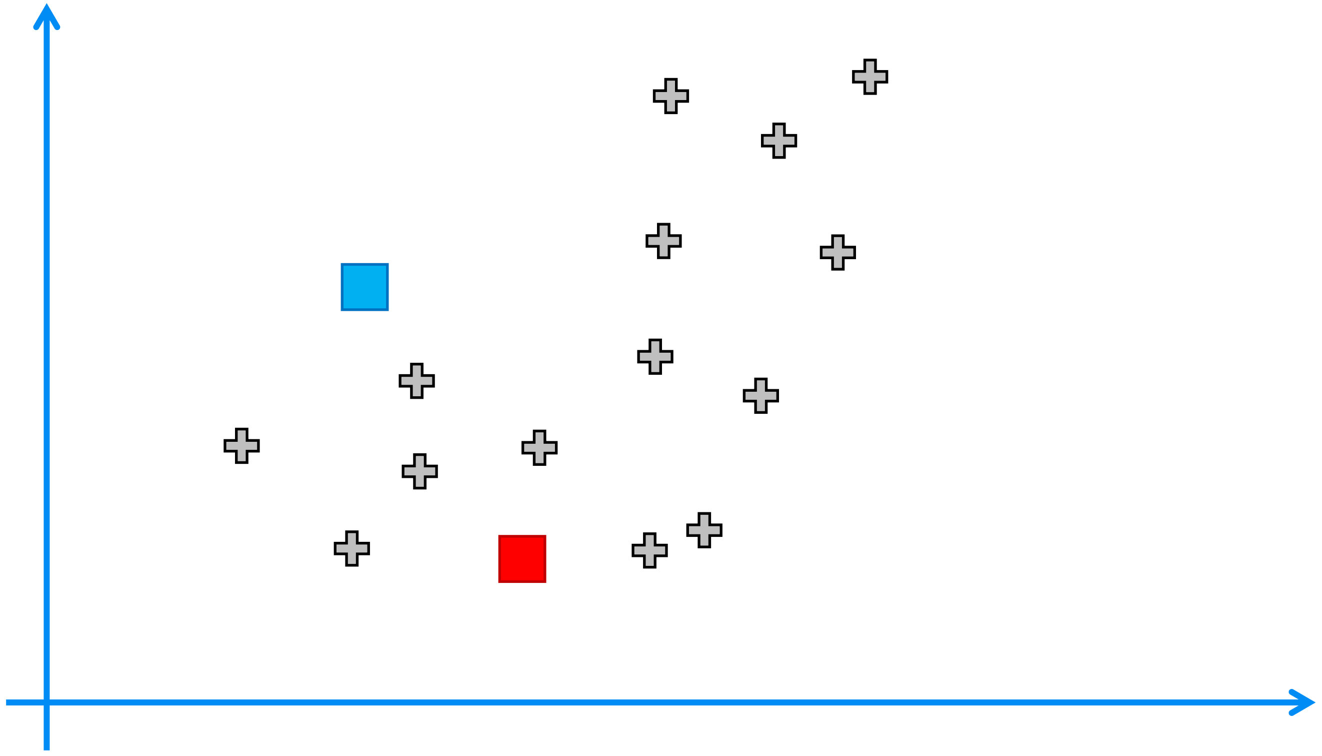


K-Means Clustering

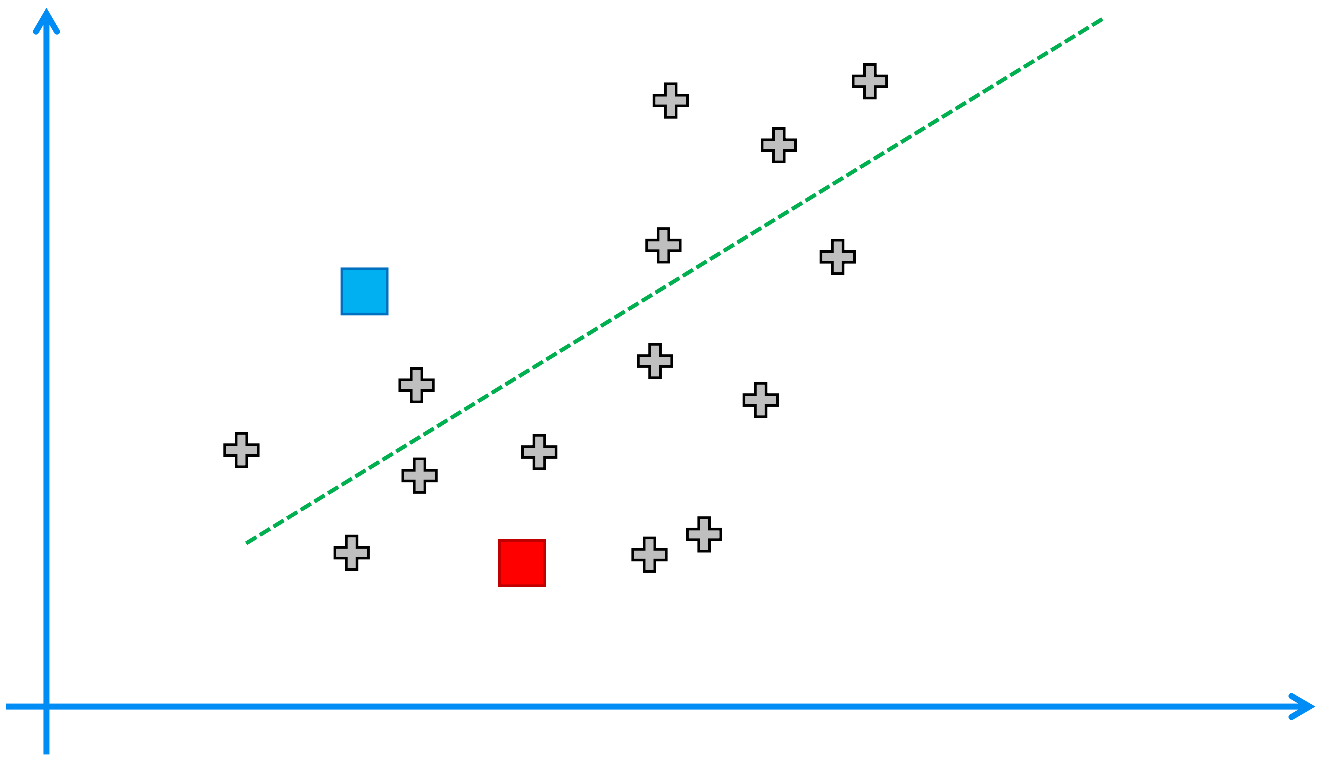
K-Means Clustering



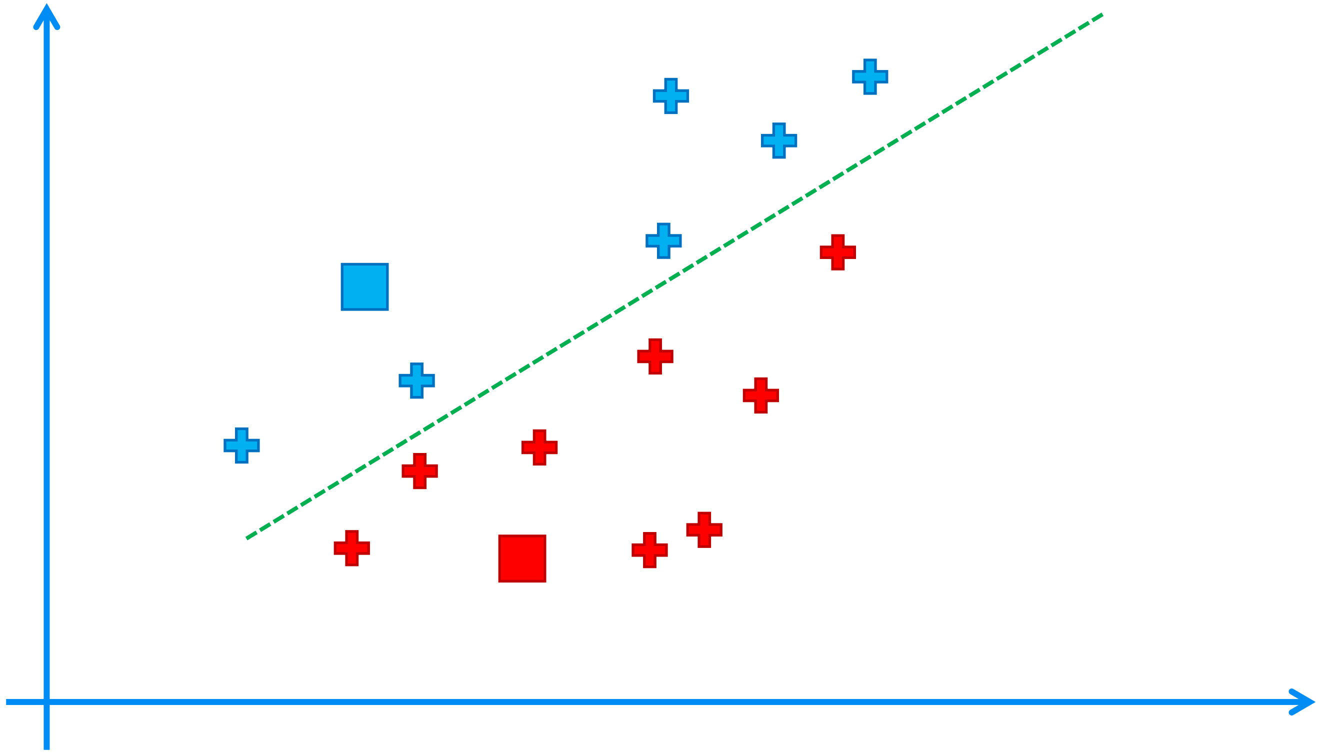
K-Means Clustering



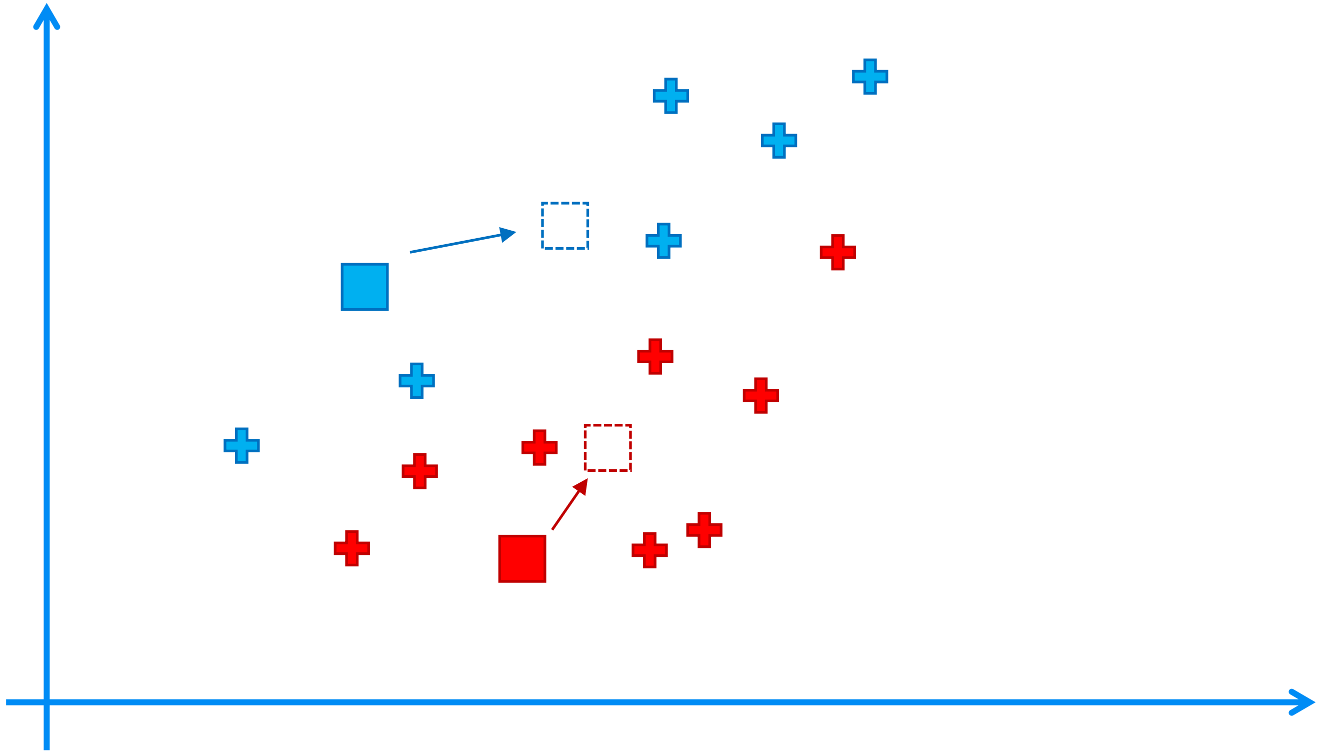
K-Means Clustering



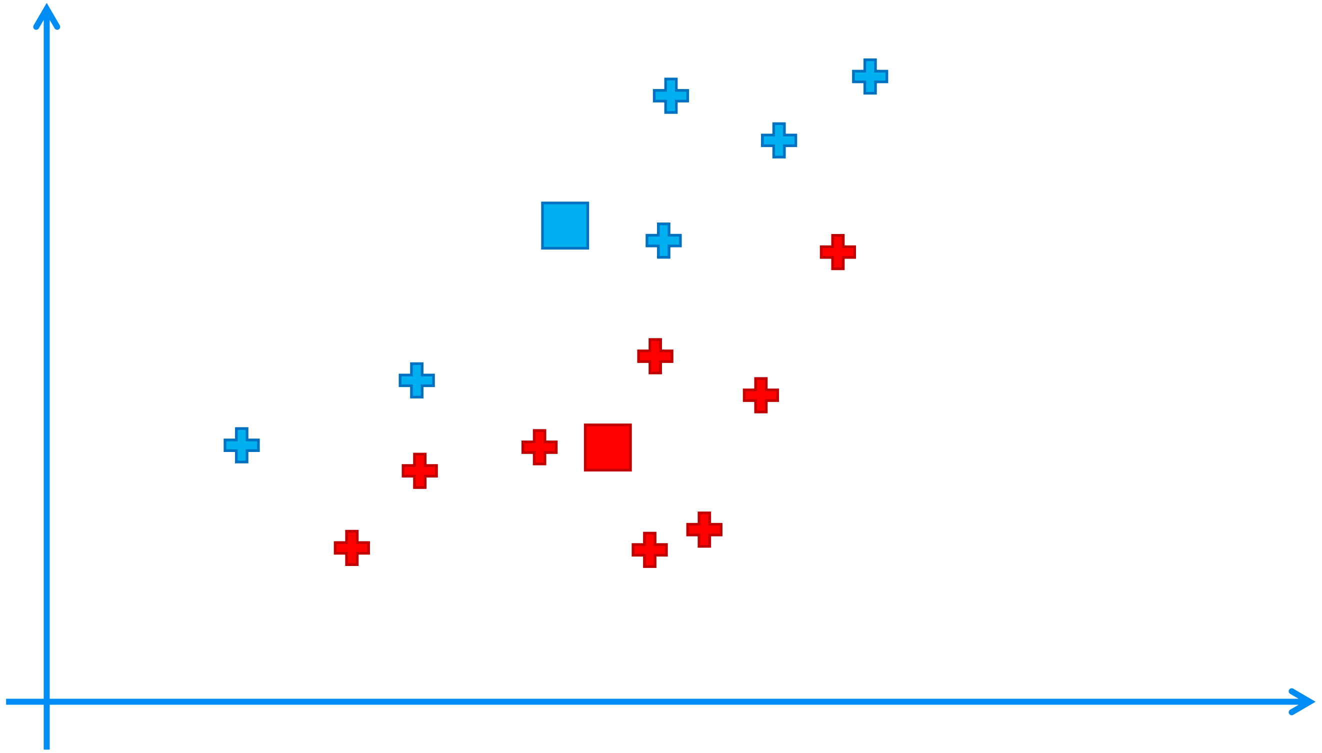
K-Means Clustering



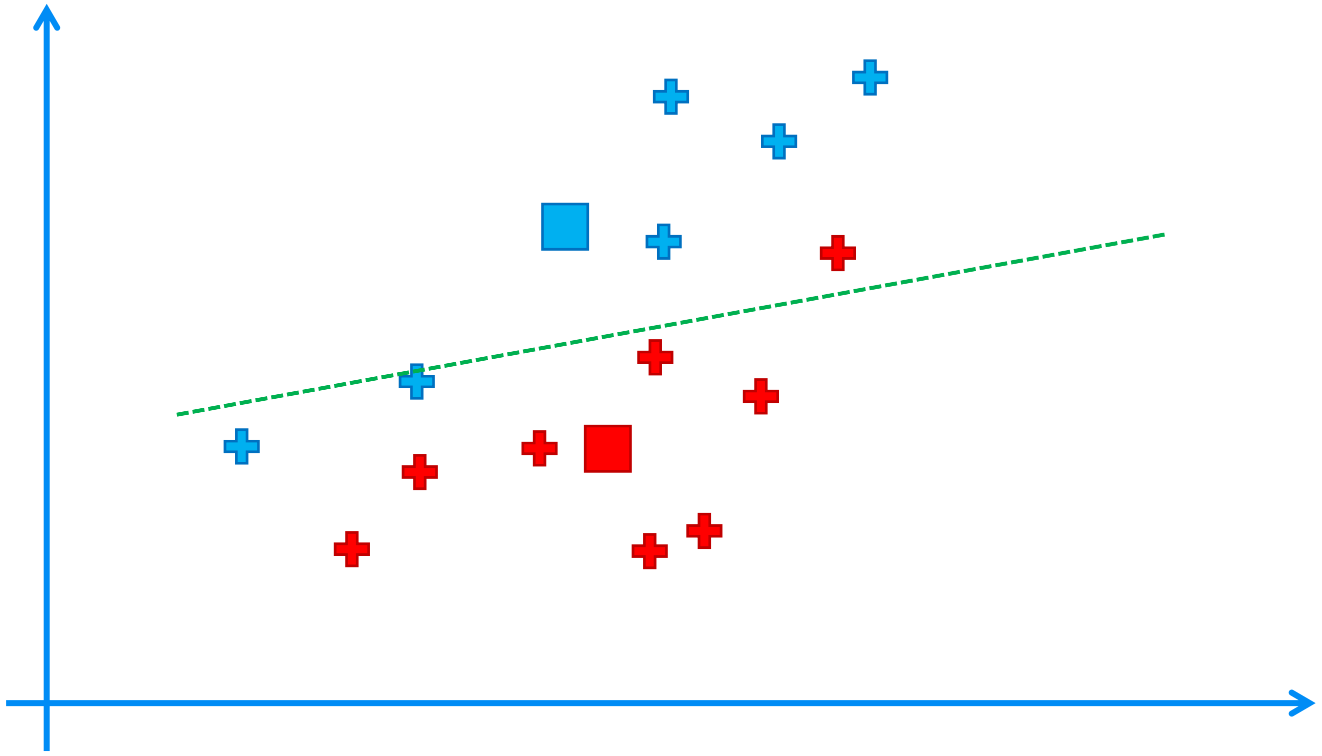
K-Means Clustering



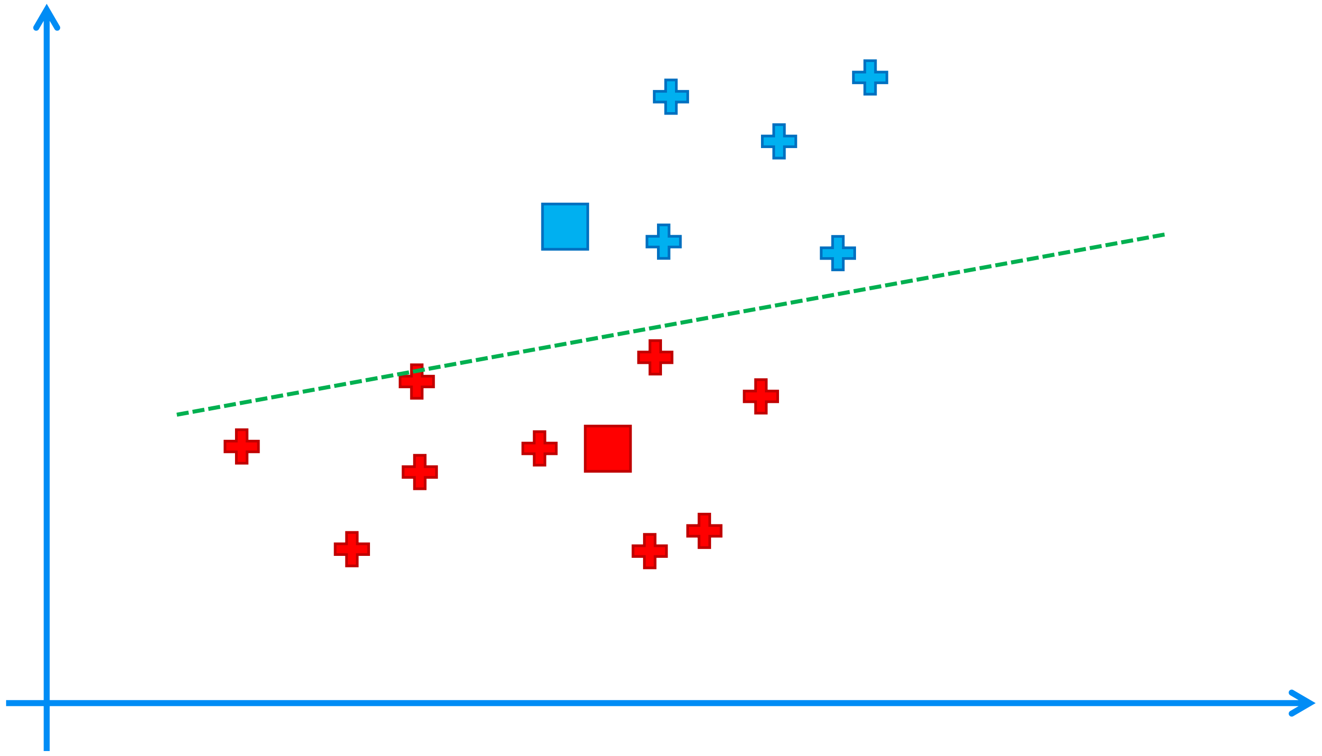
K-Means Clustering



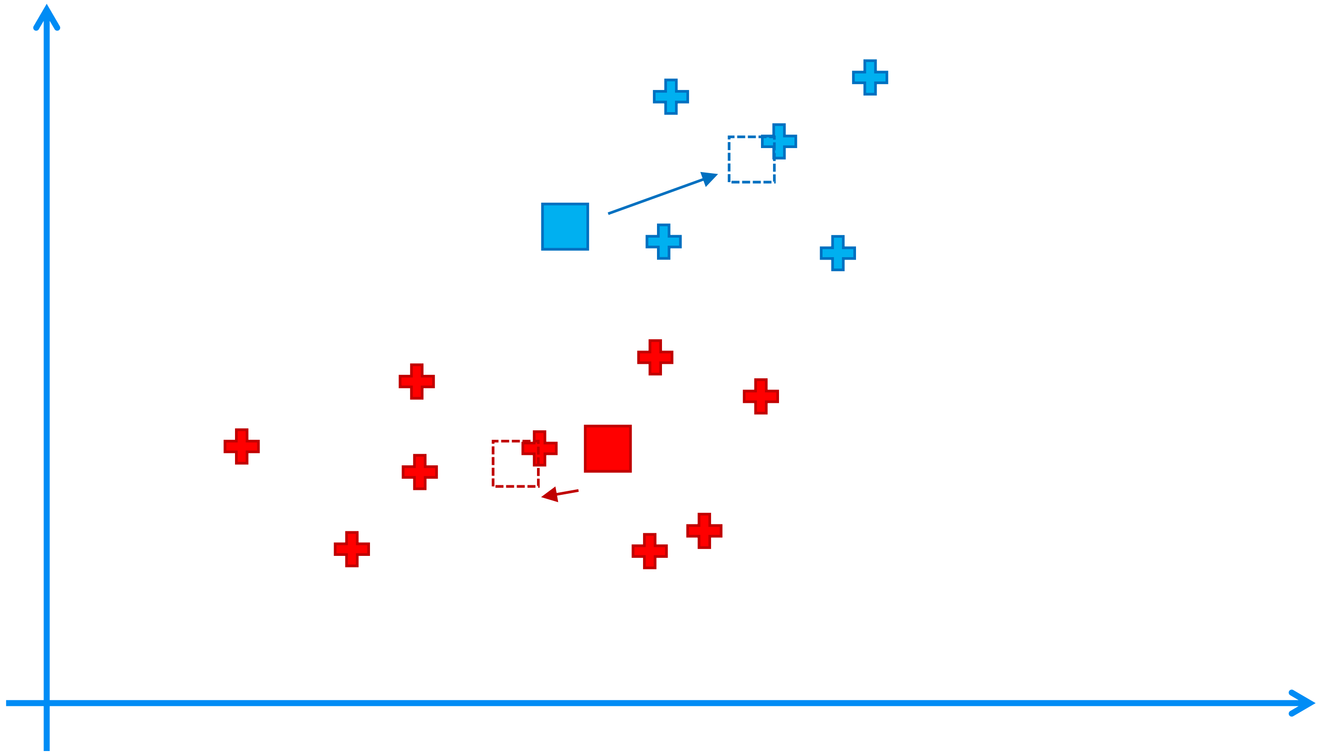
K-Means Clustering



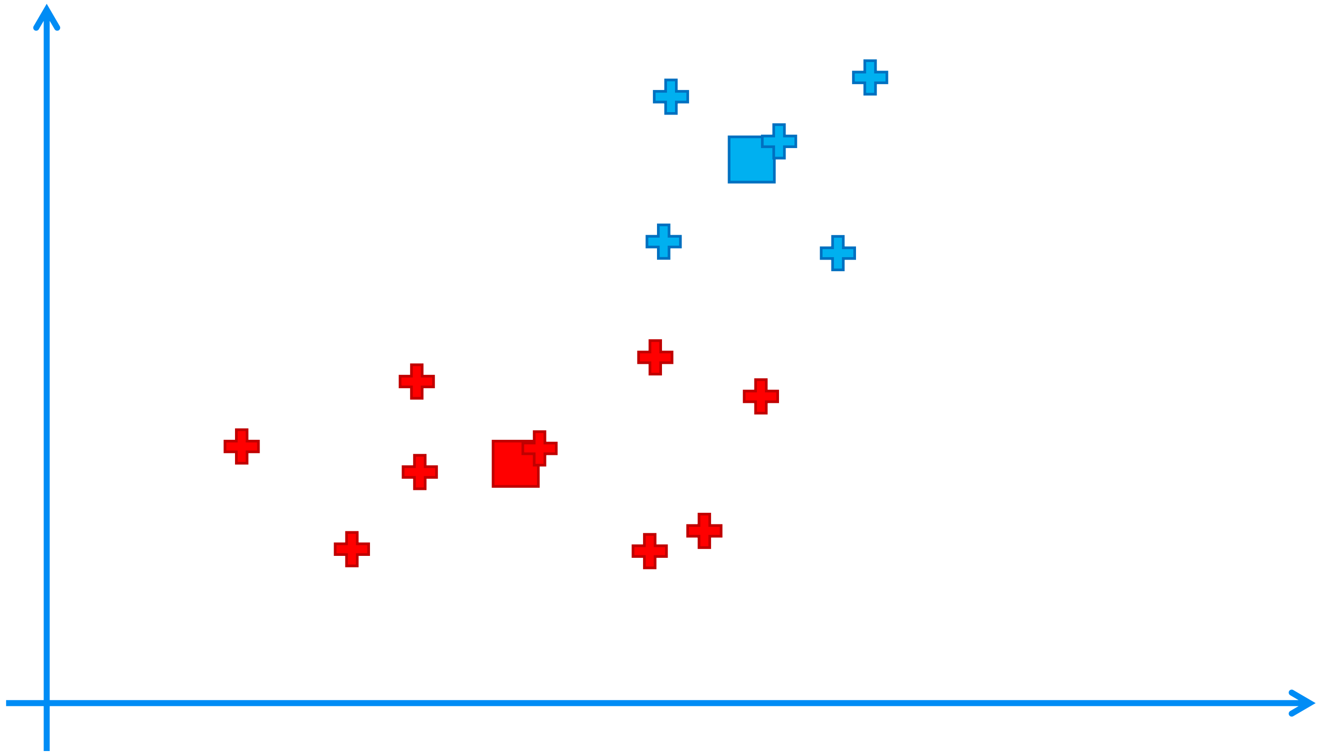
K-Means Clustering



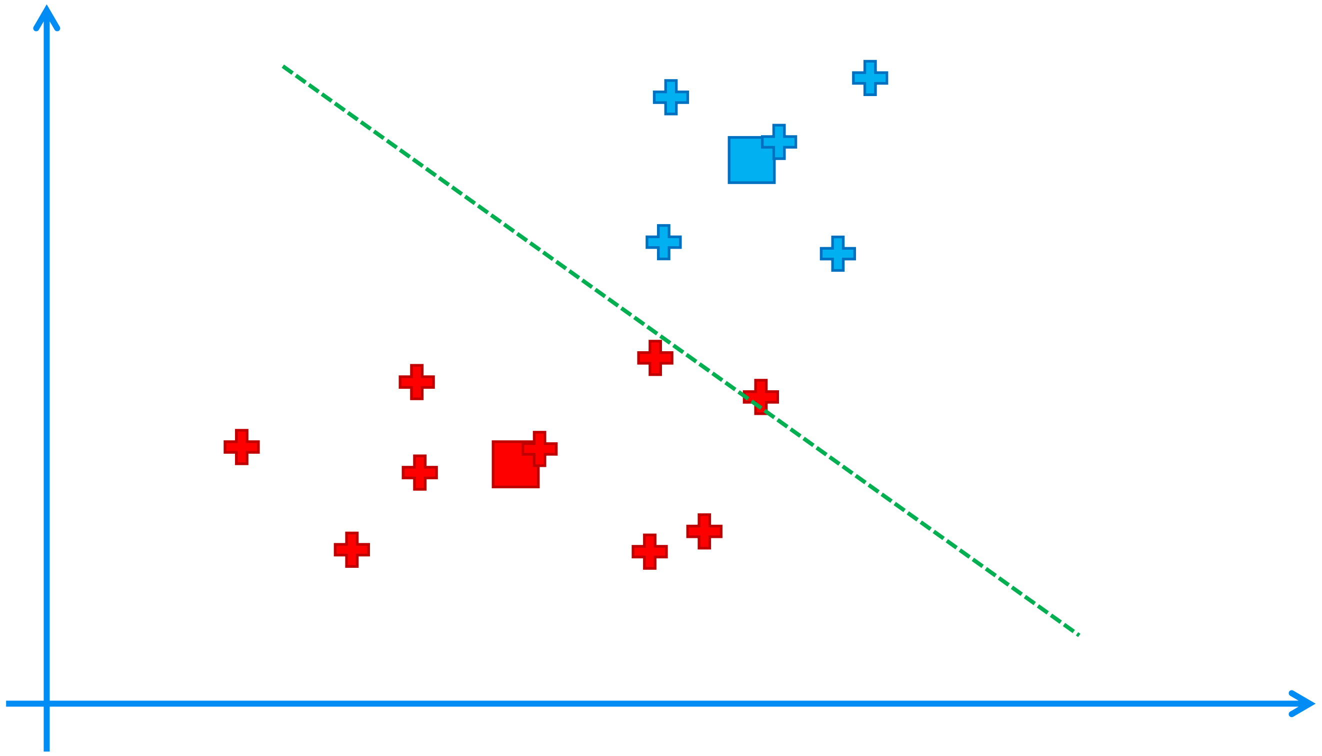
K-Means Clustering



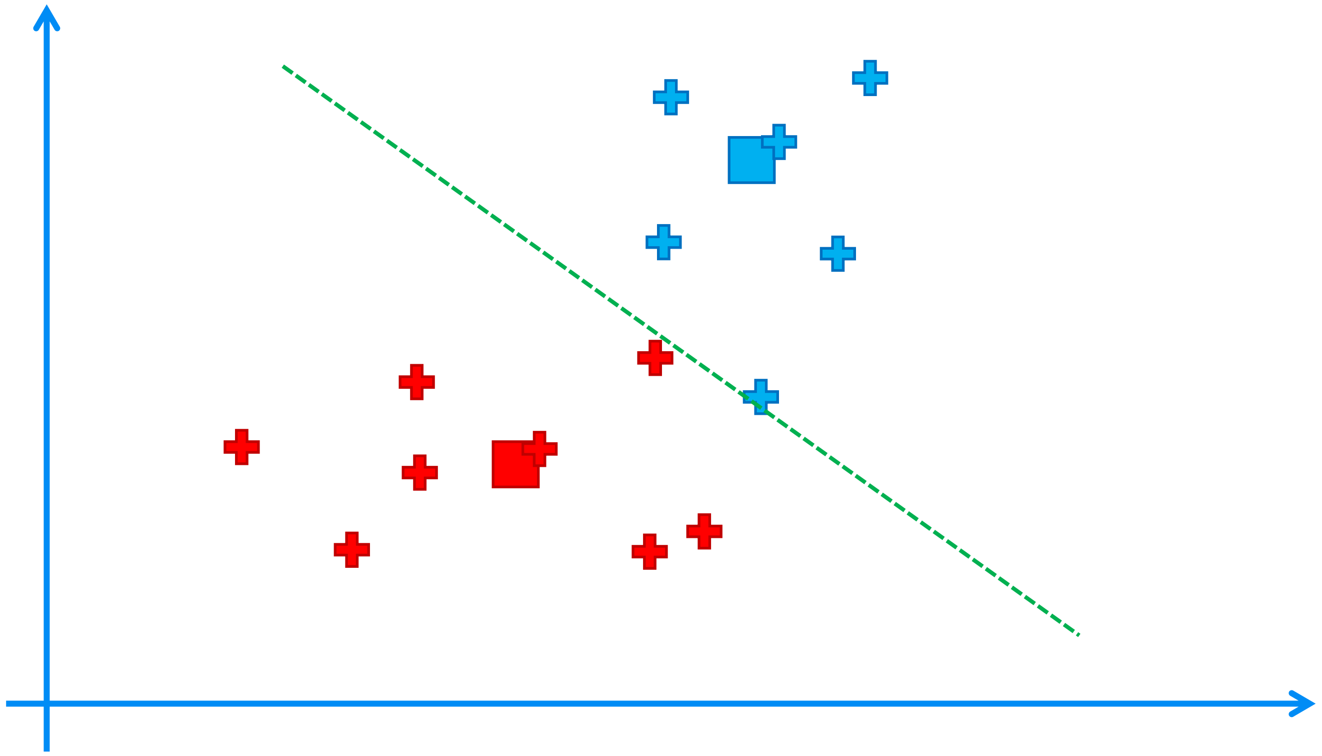
K-Means Clustering



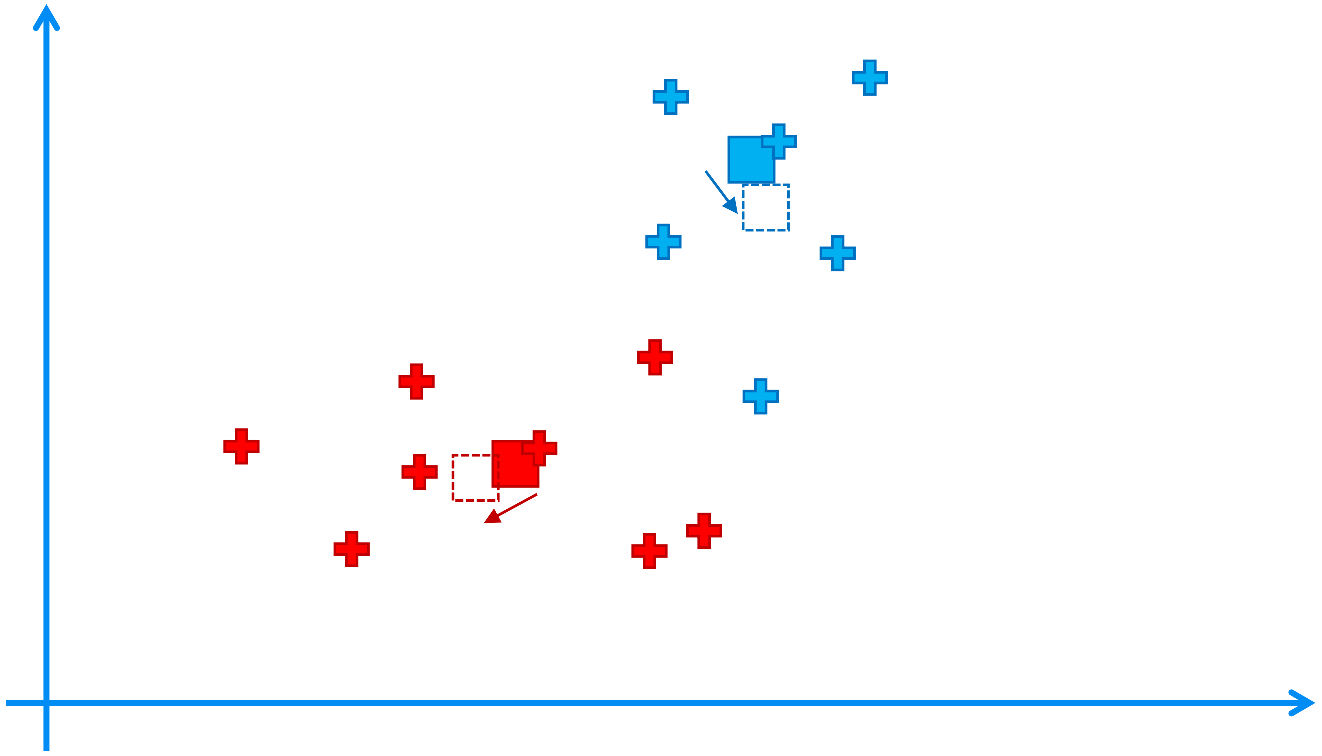
K-Means Clustering



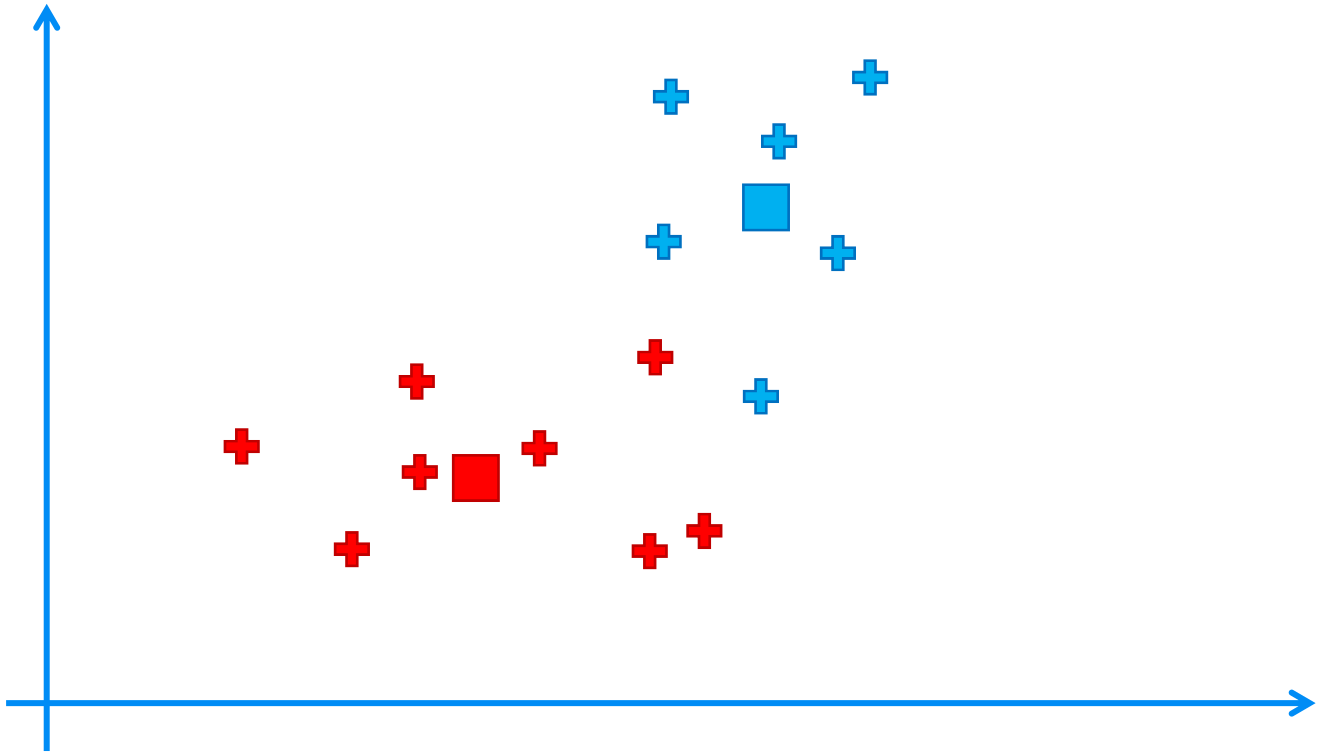
K-Means Clustering



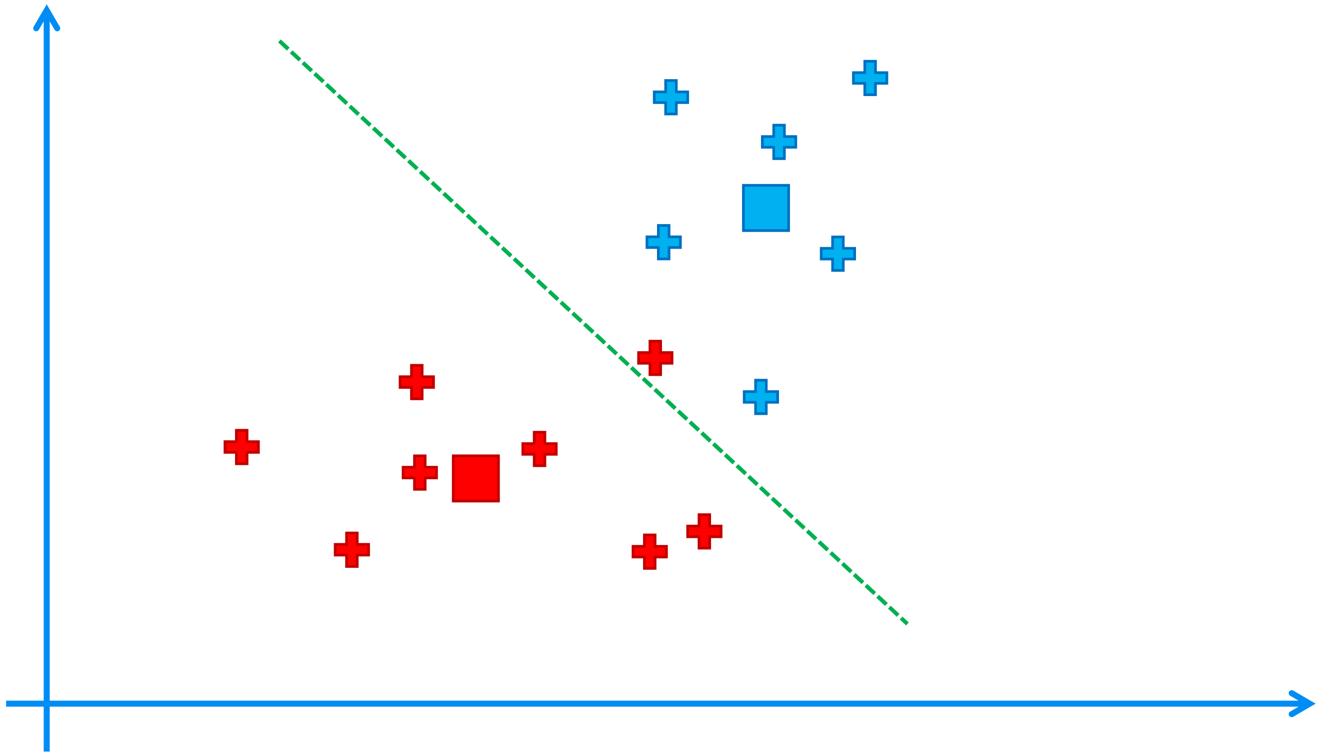
K-Means Clustering



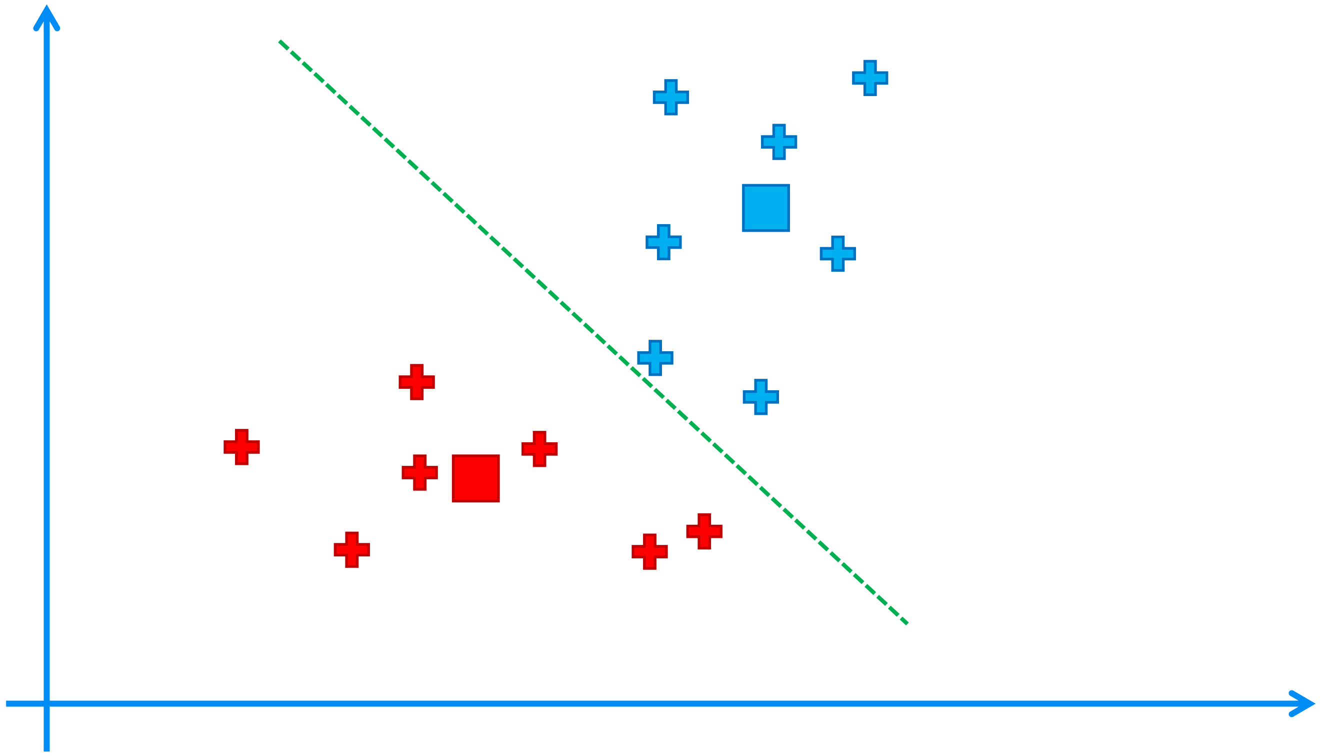
K-Means Clustering



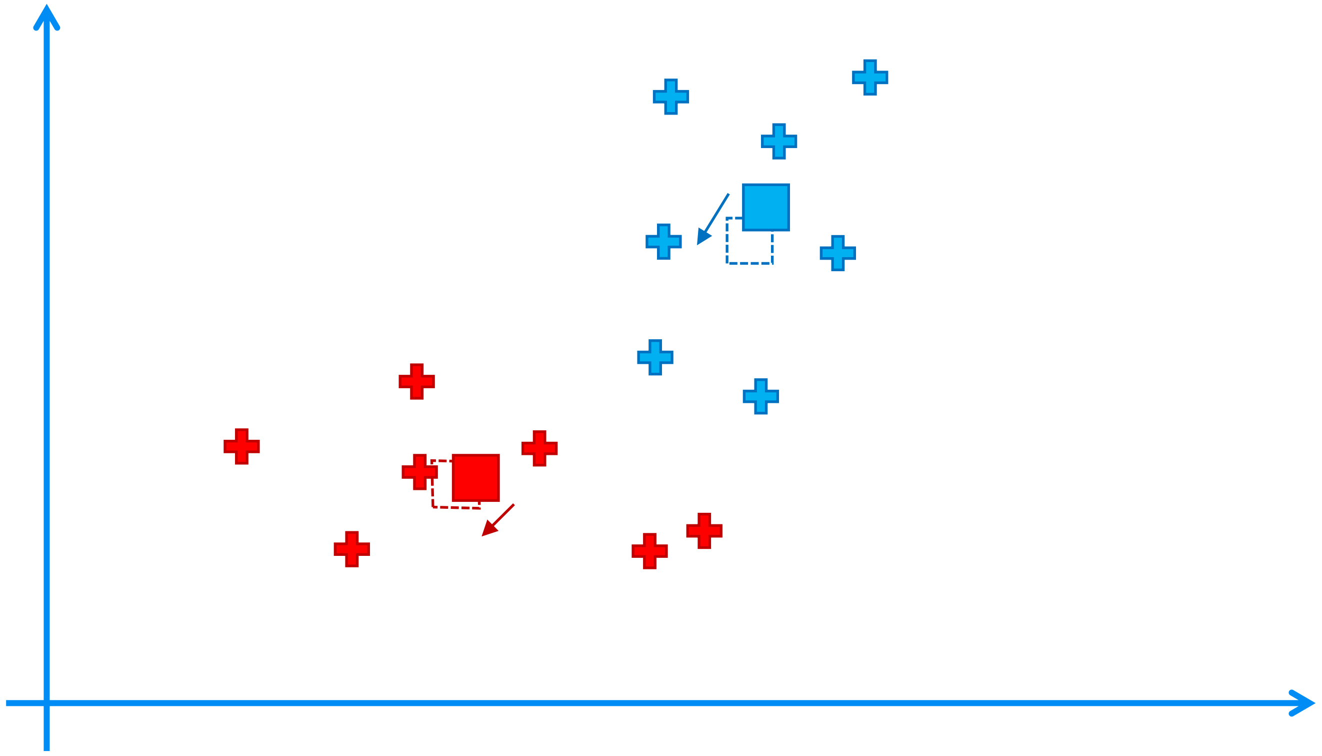
K-Means Clustering



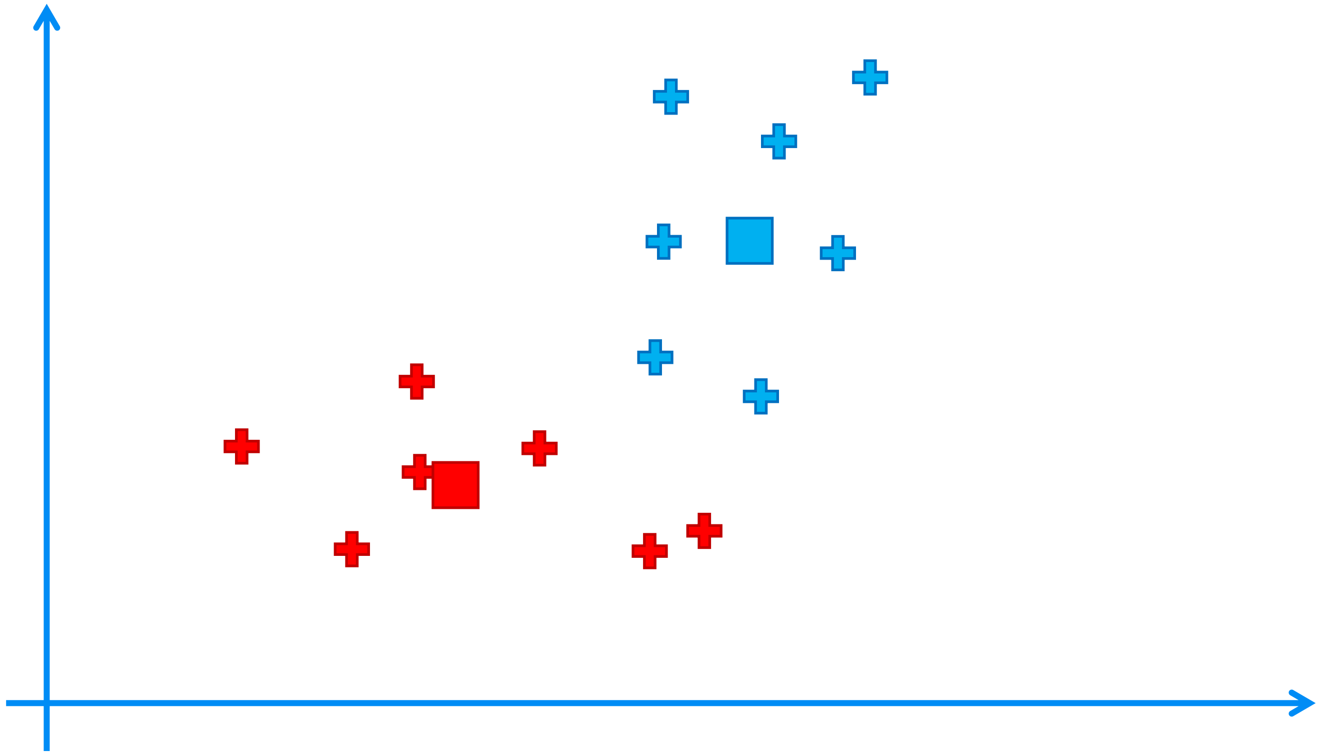
K-Means Clustering



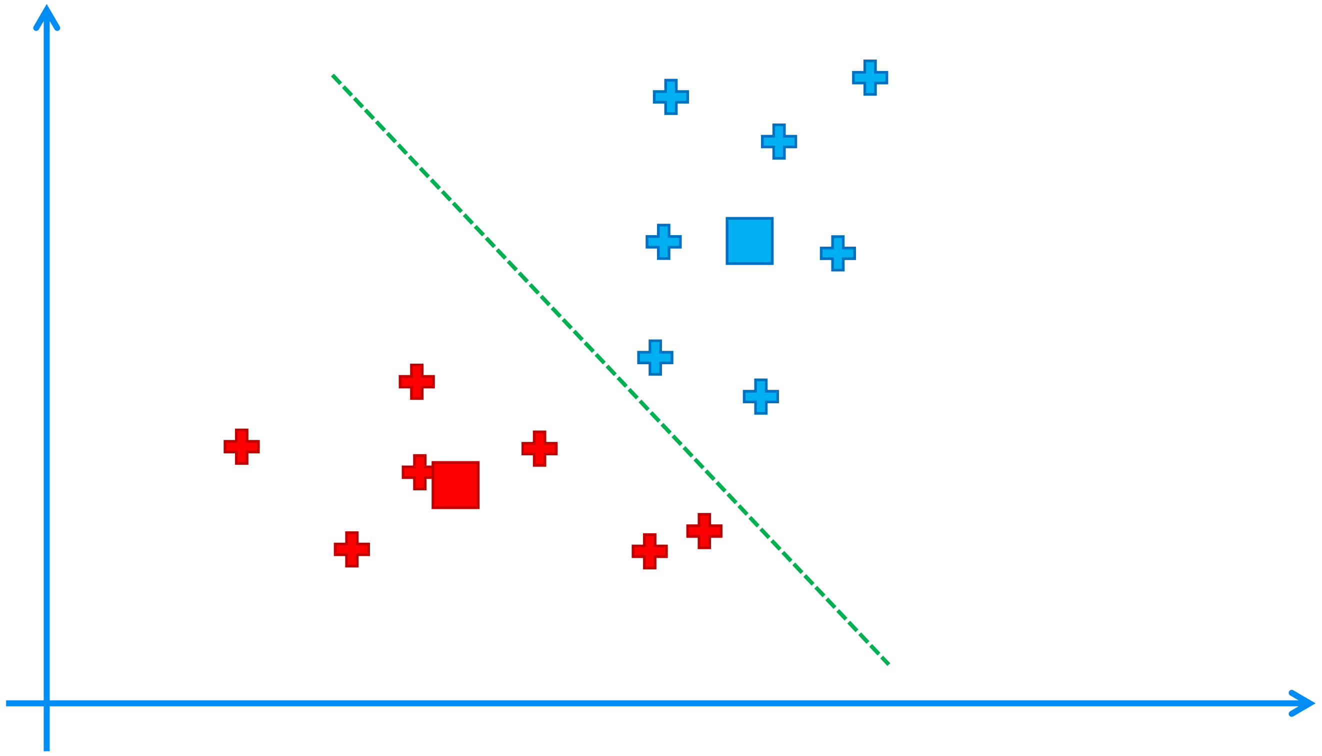
K-Means Clustering



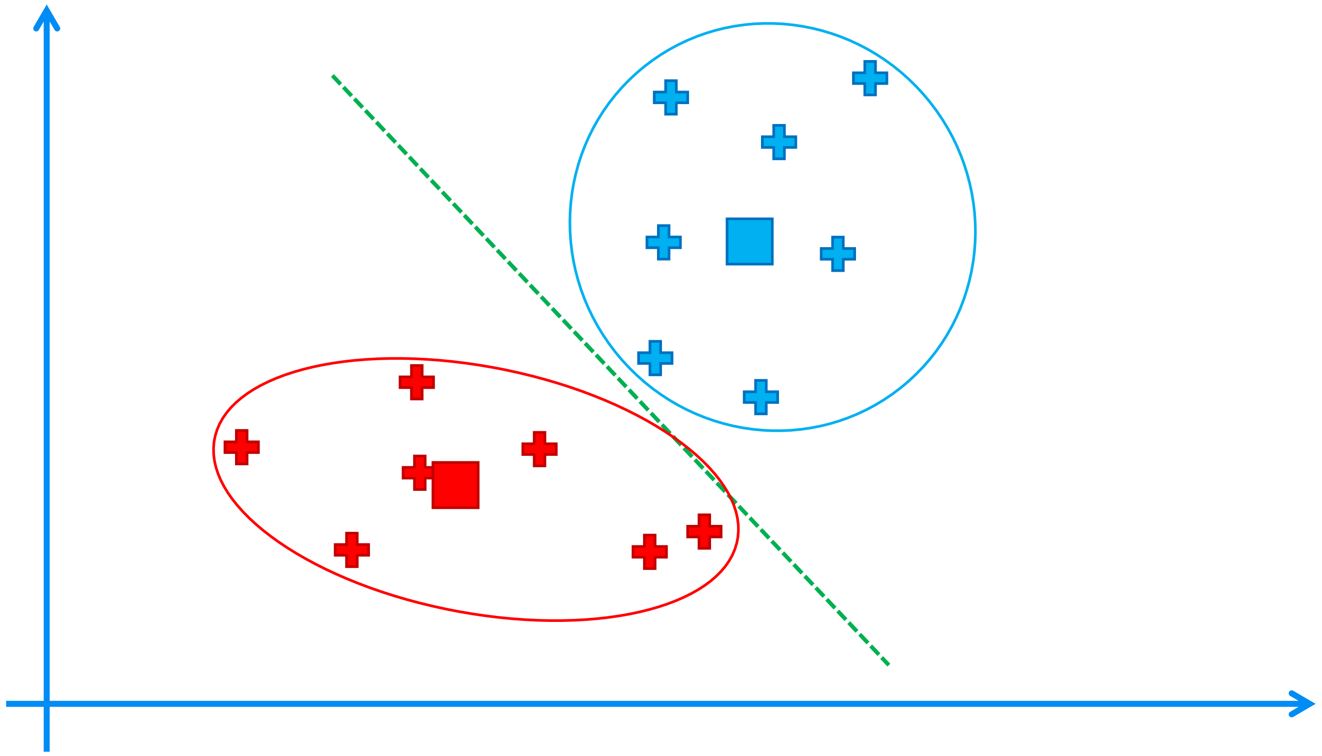
K-Means Clustering



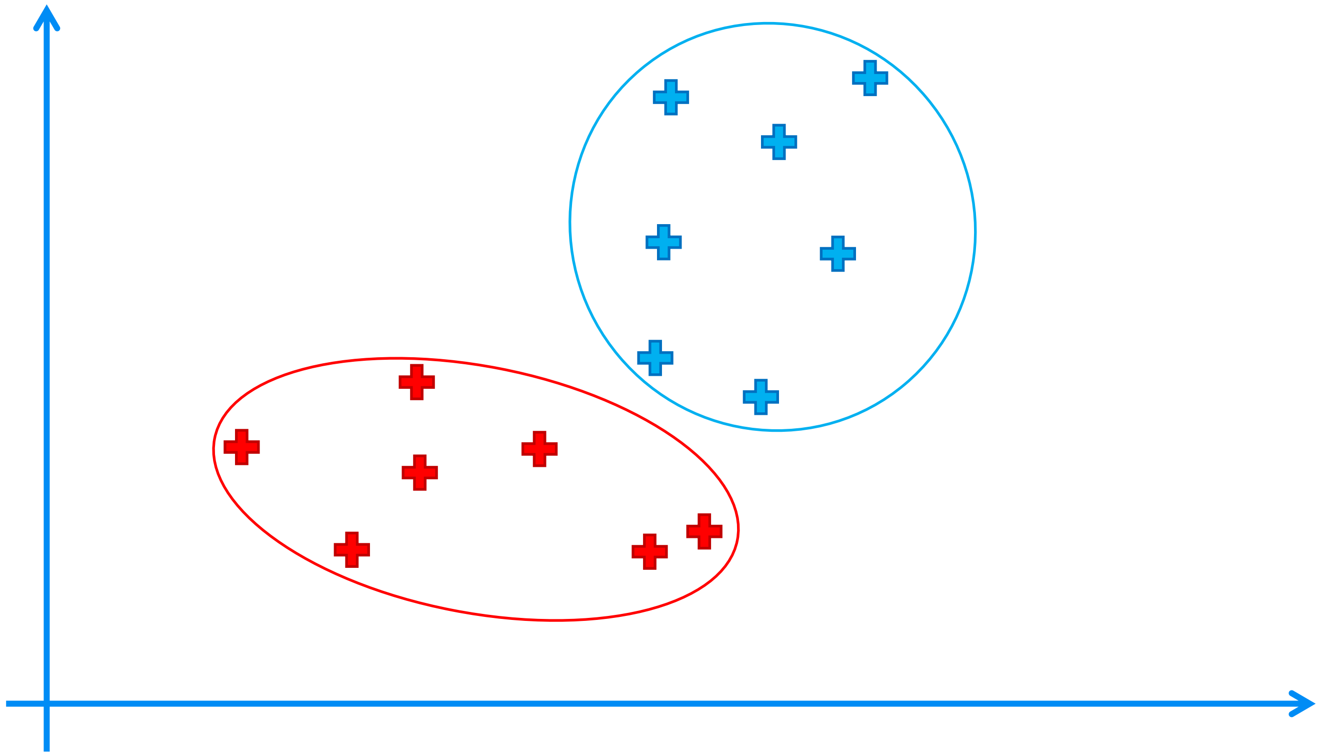
K-Means Clustering



K-Means Clustering

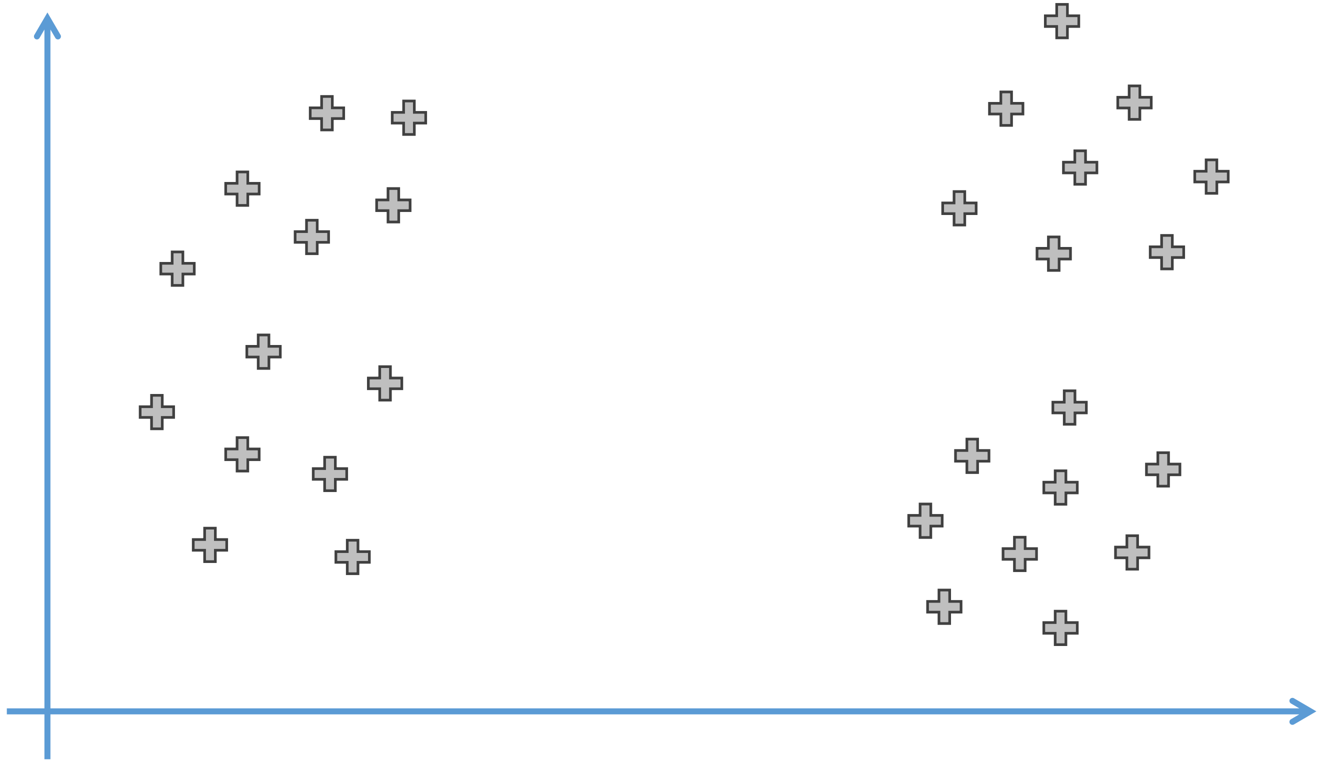


K-Means Clustering



The Elbow Method

The Elbow Method



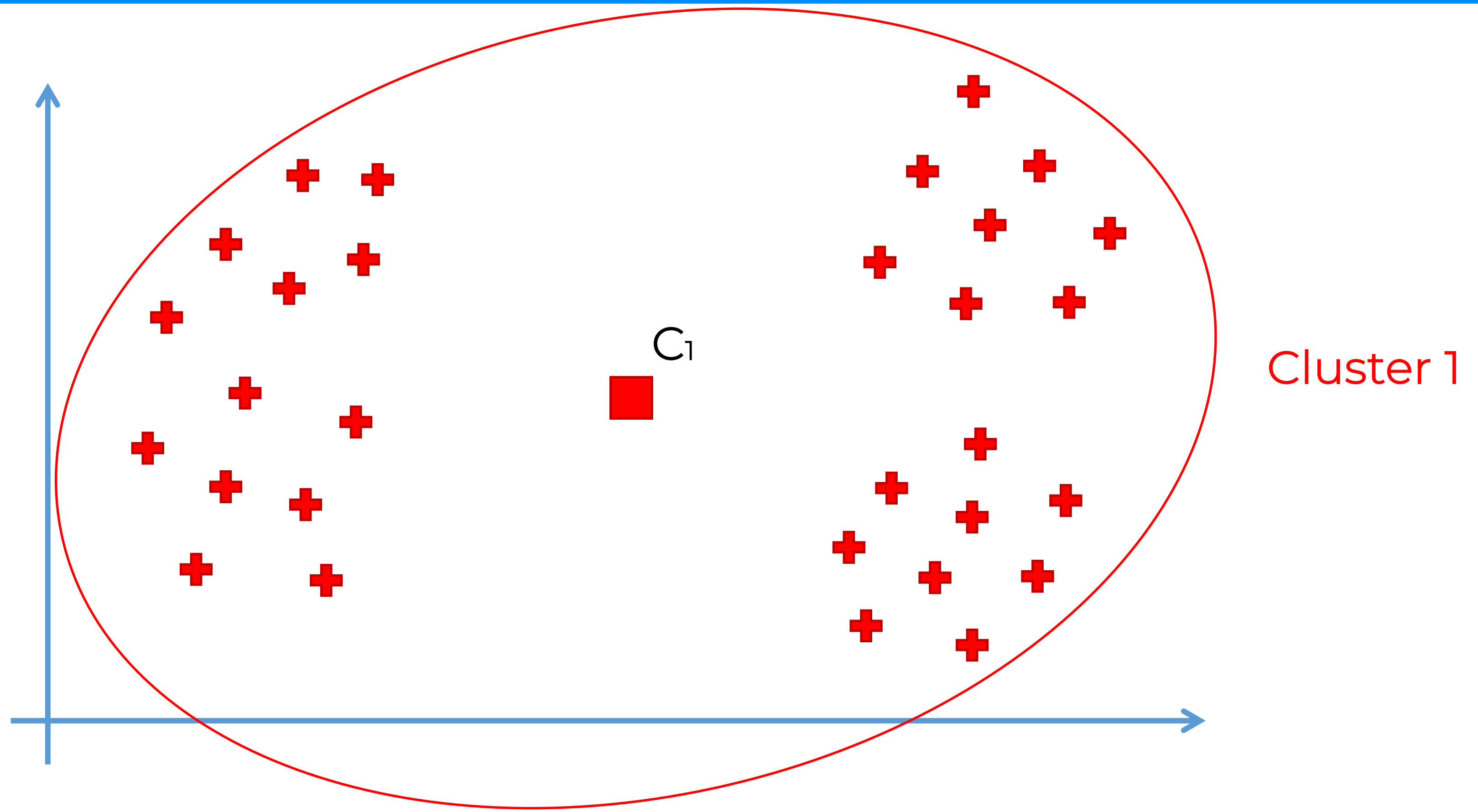
The Elbow Method

Within Cluster Sum of Squares:

$$\mathrm{WCSS} = \sum_{\mathrm{P_i\ in\ Cluster\ 1}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_1})^2 + \sum_{\mathrm{P_i\ in\ Cluster\ 2}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_2})^2 + \ \ldots$$

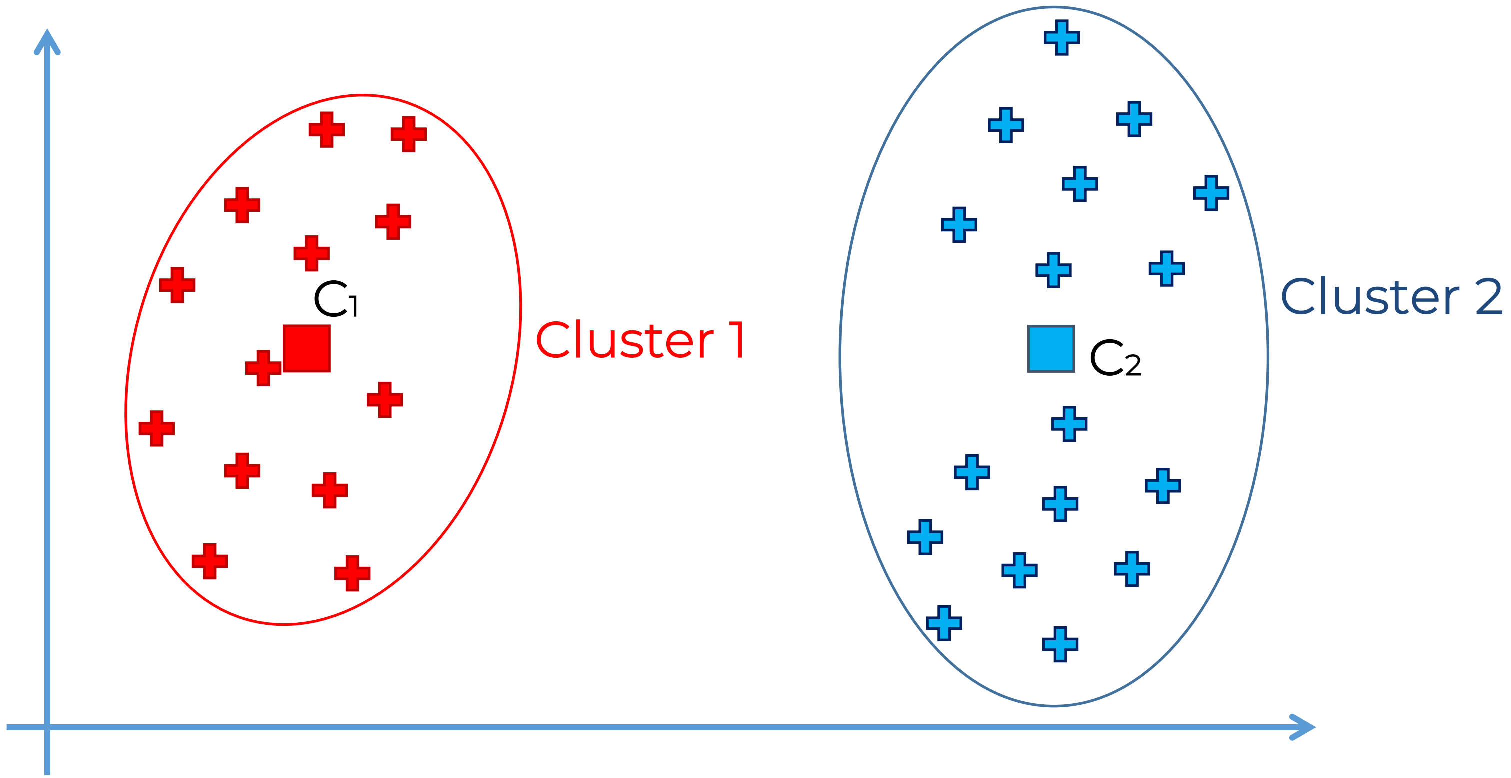
The Elbow Method

The Elbow Method



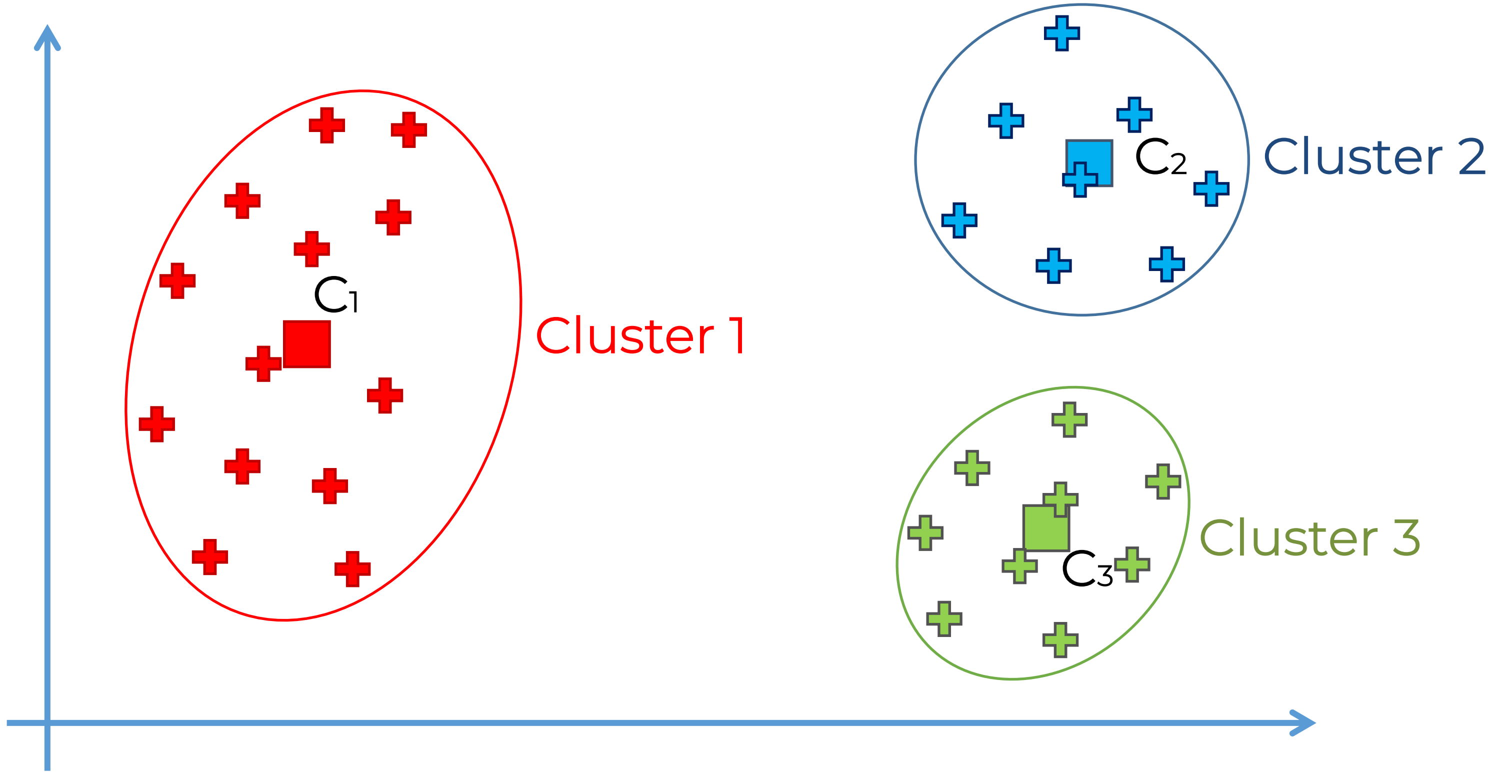
$$\text{WCSS} = \sum_{P_i \text{ in Cluster } 1} \text{distance}(P_i, C_1)^2$$

The Elbow Method



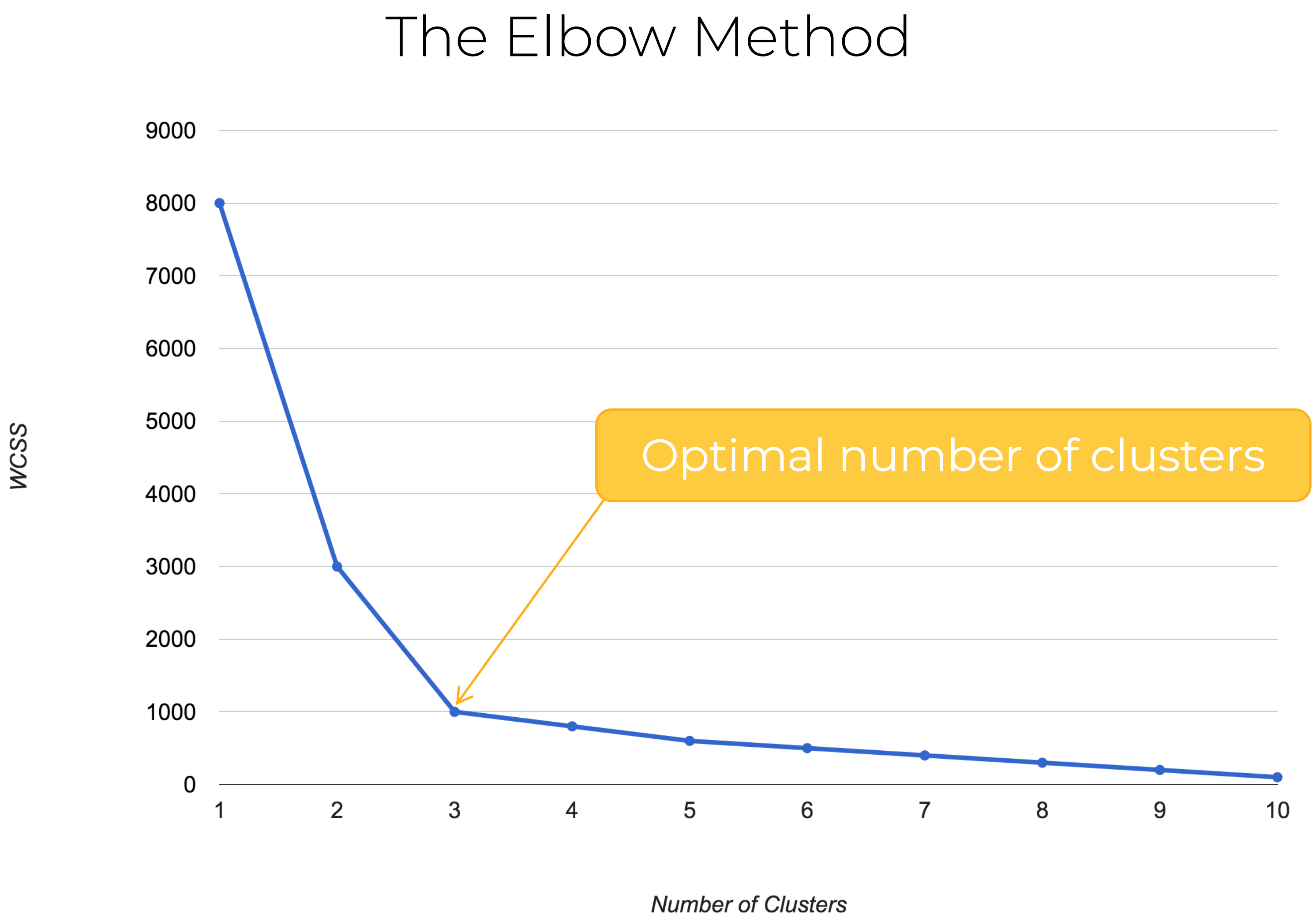
$$\mathrm{WCSS} = \sum_{P_i \text{ in Cluster } 1} \mathrm{distance}(P_i, C_1)^2 + \sum_{P_i \text{ in Cluster } 2} \mathrm{distance}(P_i, C_2)^2$$

The Elbow Method



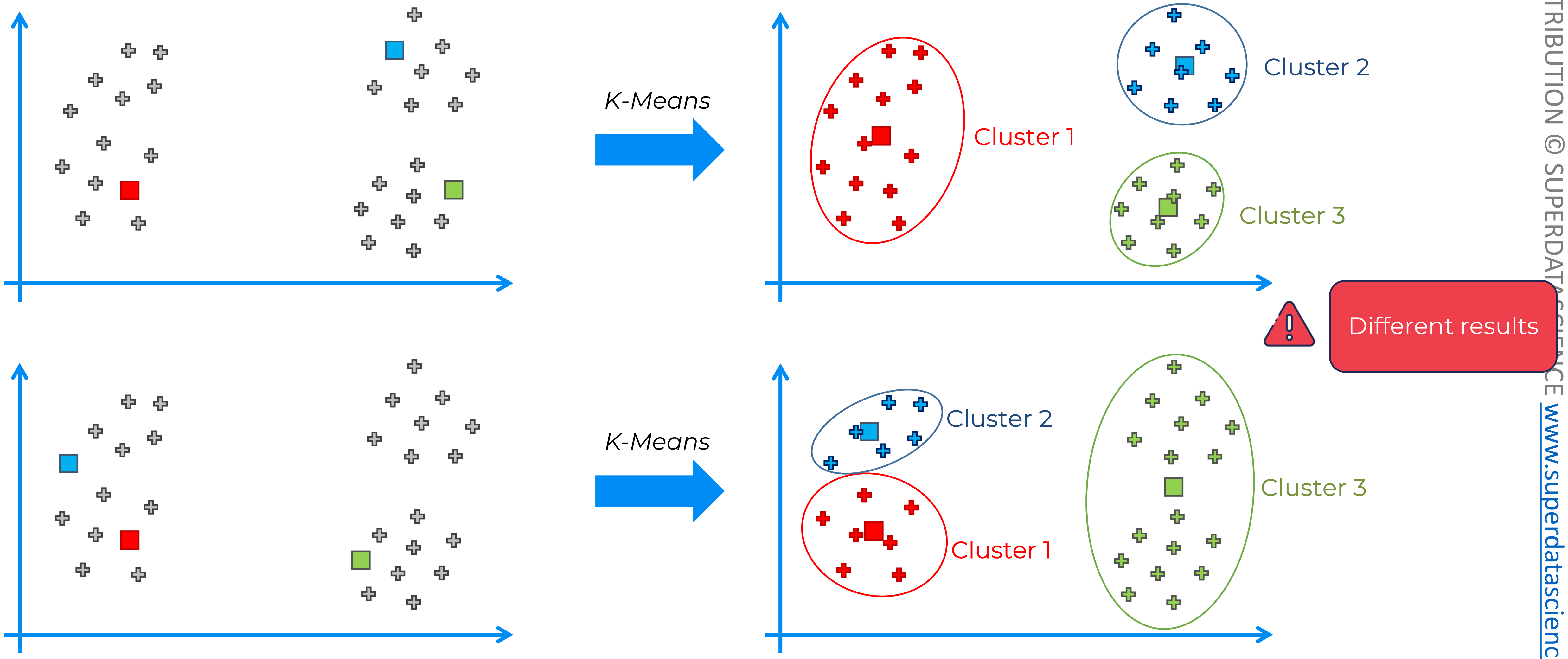
$$\text{WCSS} = \sum_{P_i \text{ in Cluster 1}} \text{distance}(P_i, C_1)^2 + \sum_{P_i \text{ in Cluster 2}} \text{distance}(P_i, C_2)^2 + \sum_{P_i \text{ in Cluster 3}} \text{distance}(P_i, C_3)^2$$

The Elbow Method



K-Means++

K-Means++


K-Means
Cluster 1
Cluster 2
Cluster 3
Different results
K-Means
Cluster 2
Cluster 1
Cluster 3

K-Means++

K-Means++ Initialization Algorithm:

Step 1: Choose first centroid at random among data points

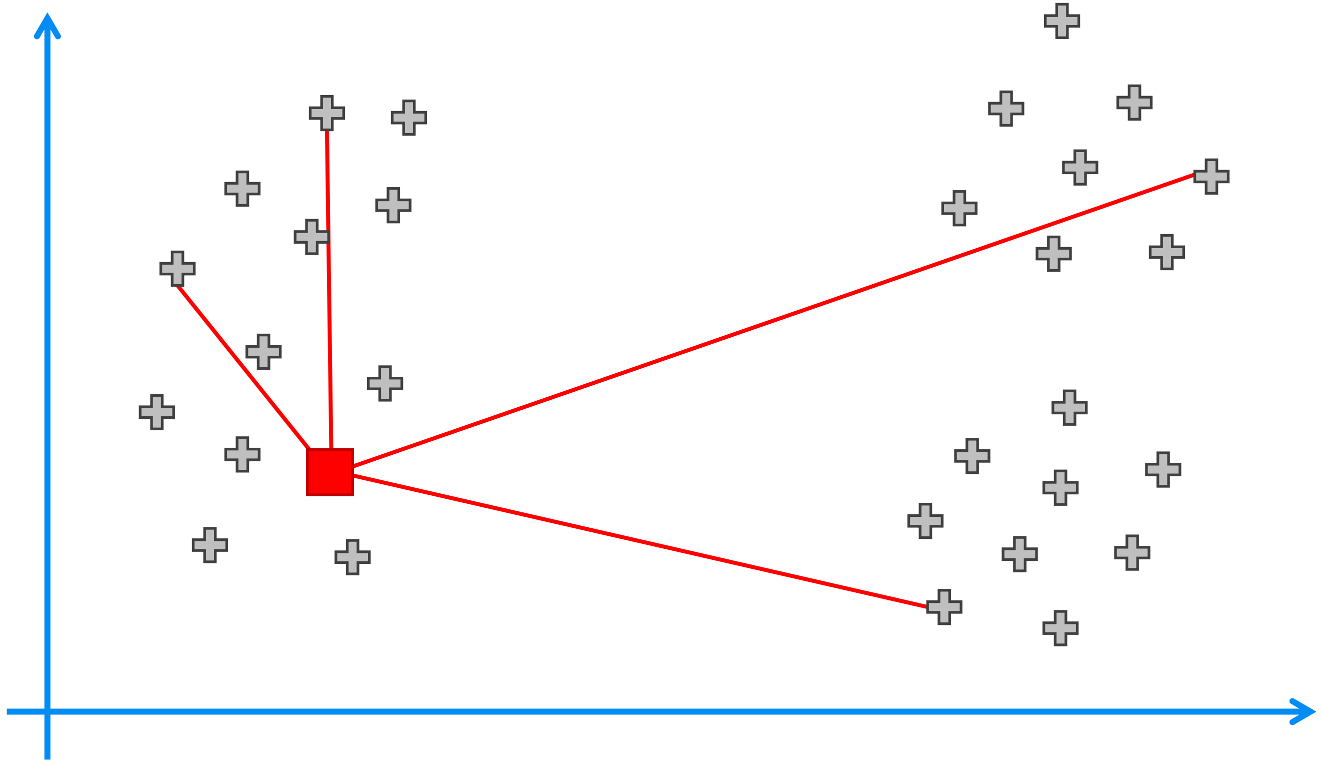
Step 2: For each of the remaining data points compute the distance (D) to the nearest out of already selected centroids

Step 3: Choose next centroid among remaining data points using weighted random selection – weighted by D^2

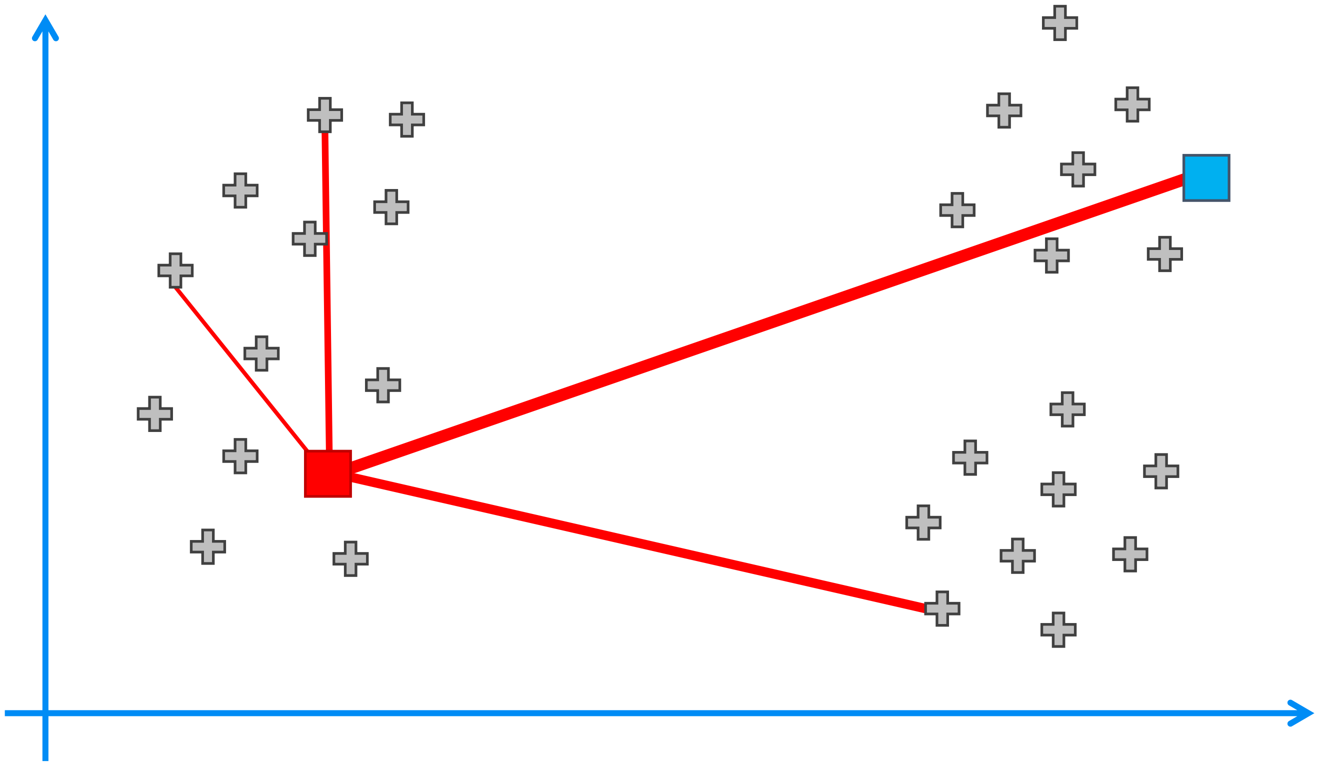
Step 4: Repeat Steps 2 and 3 until all k centroids have been selected

Step 5: Proceed with standard k-means clustering

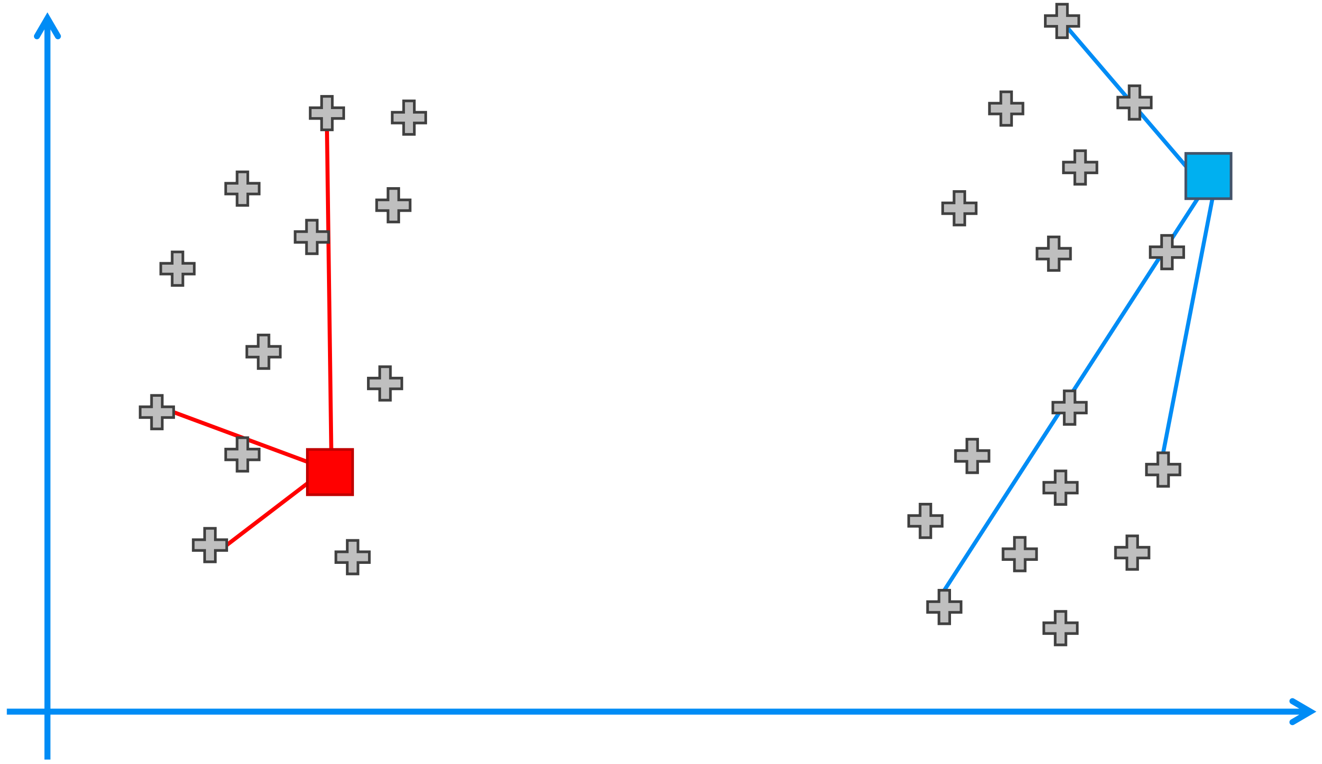
K-Means++



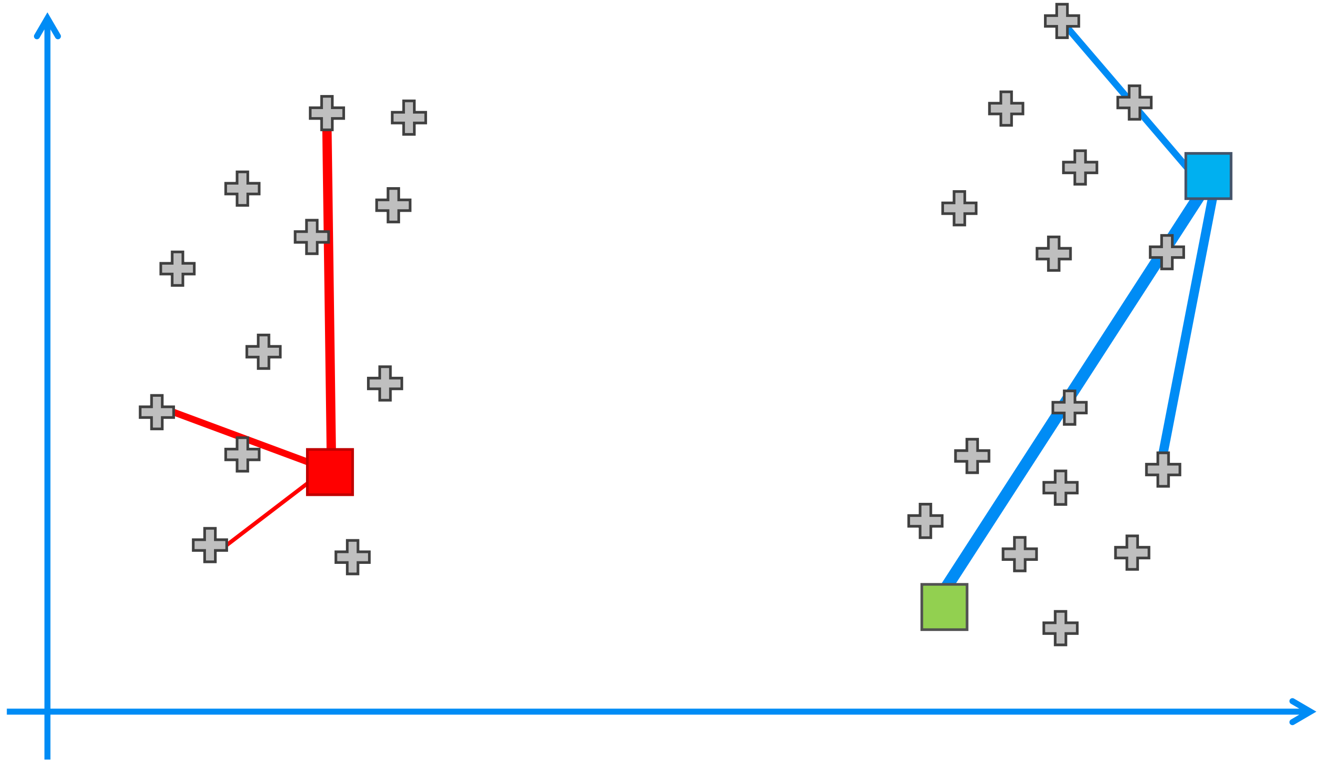
K-Means++



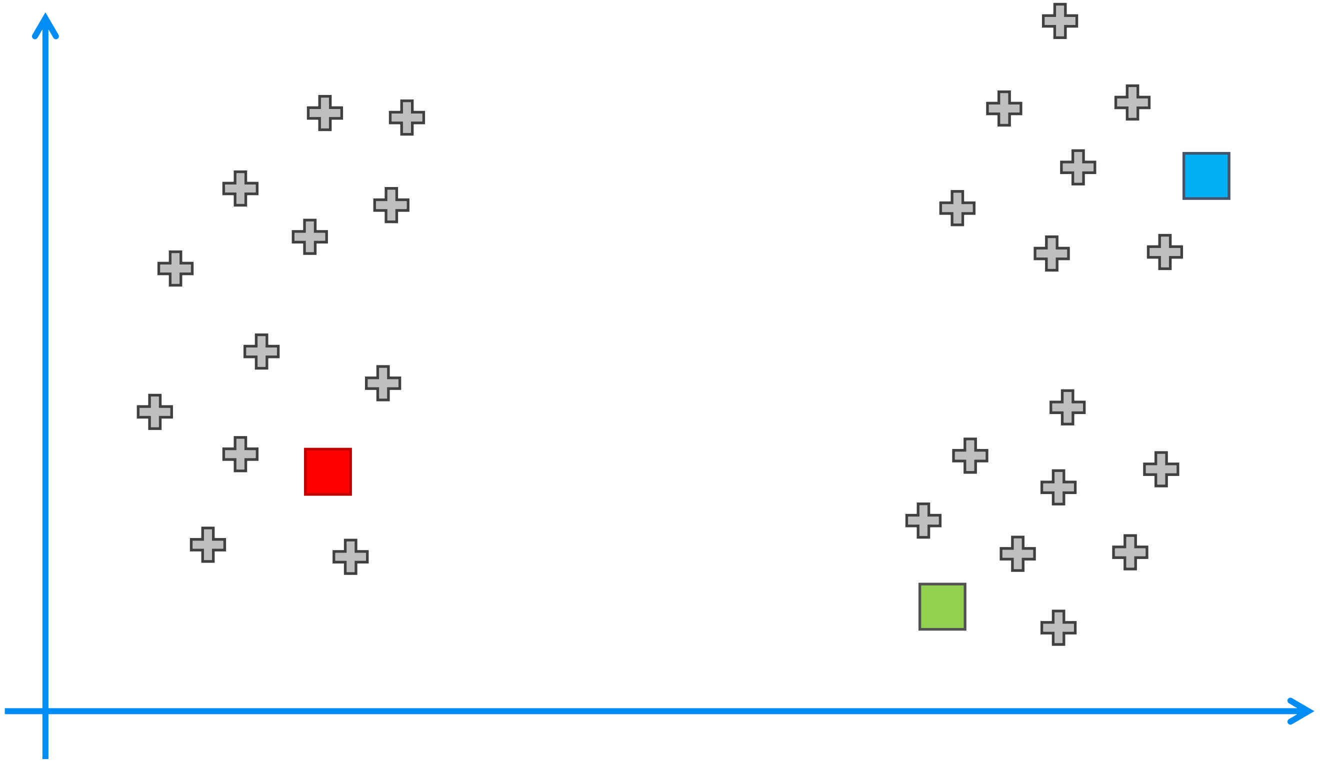
K-Means++



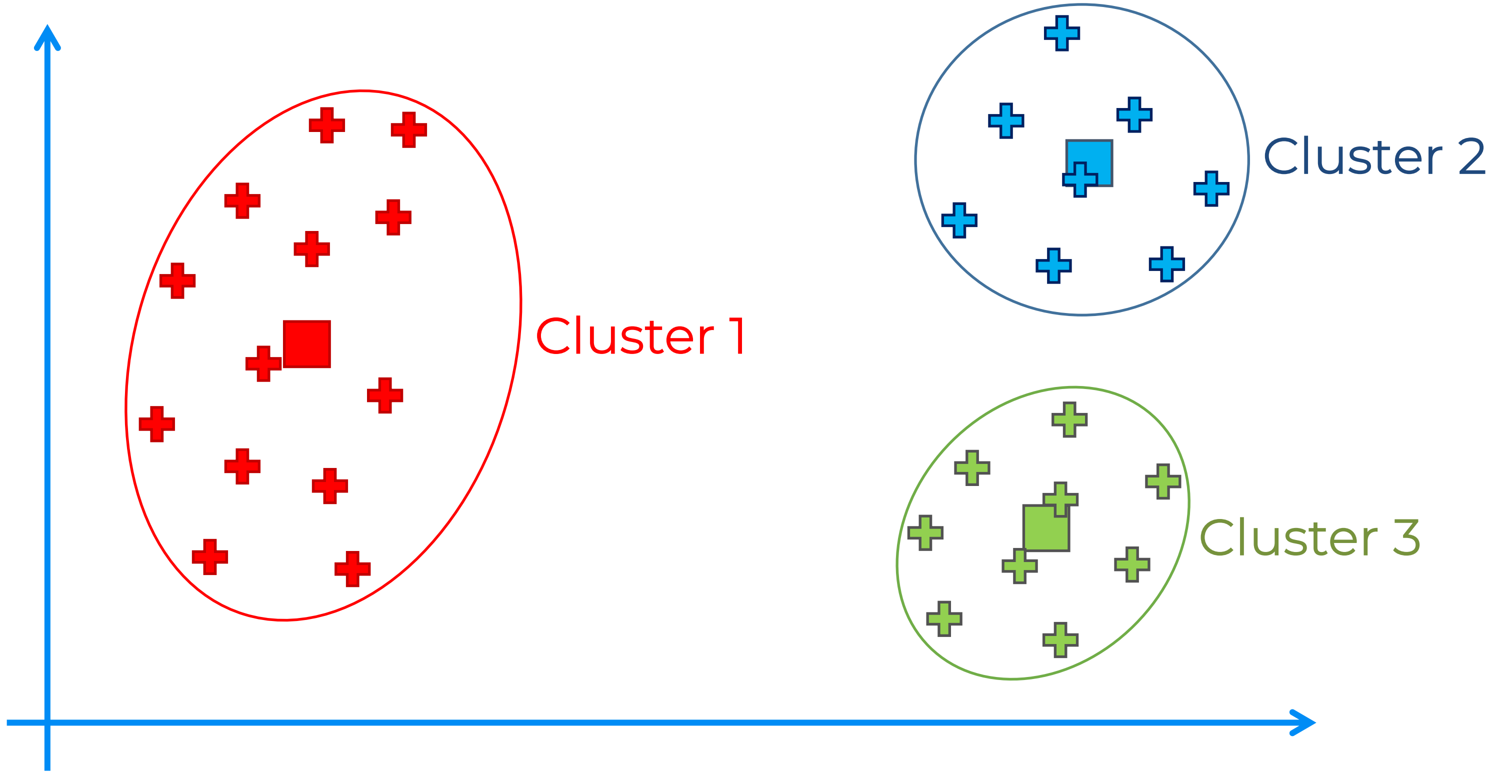
K-Means++



K-Means++

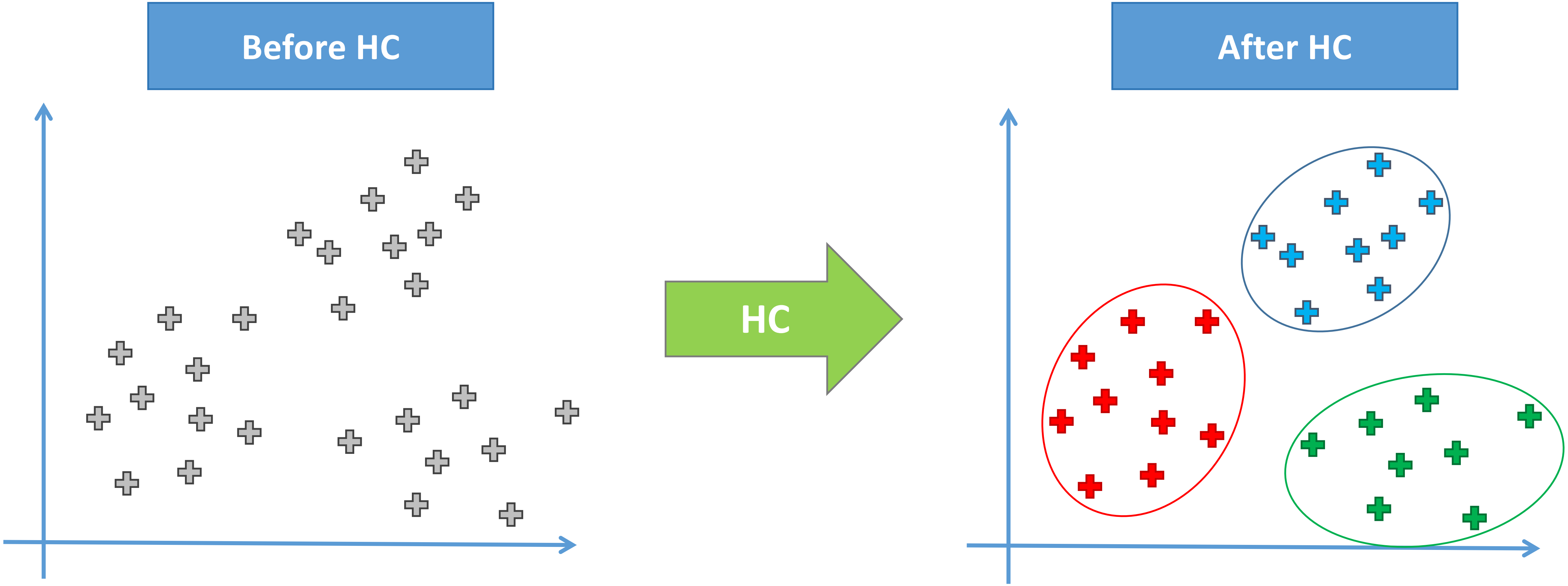


K-Means++



HC Intuition: Understanding HC

What HC does for you



Same as K-Means but different process

NOTE: Agglomerative & Divisive

Agglomerative HC

STEP 1: Make each data point a single-point cluster ➡ That forms N clusters

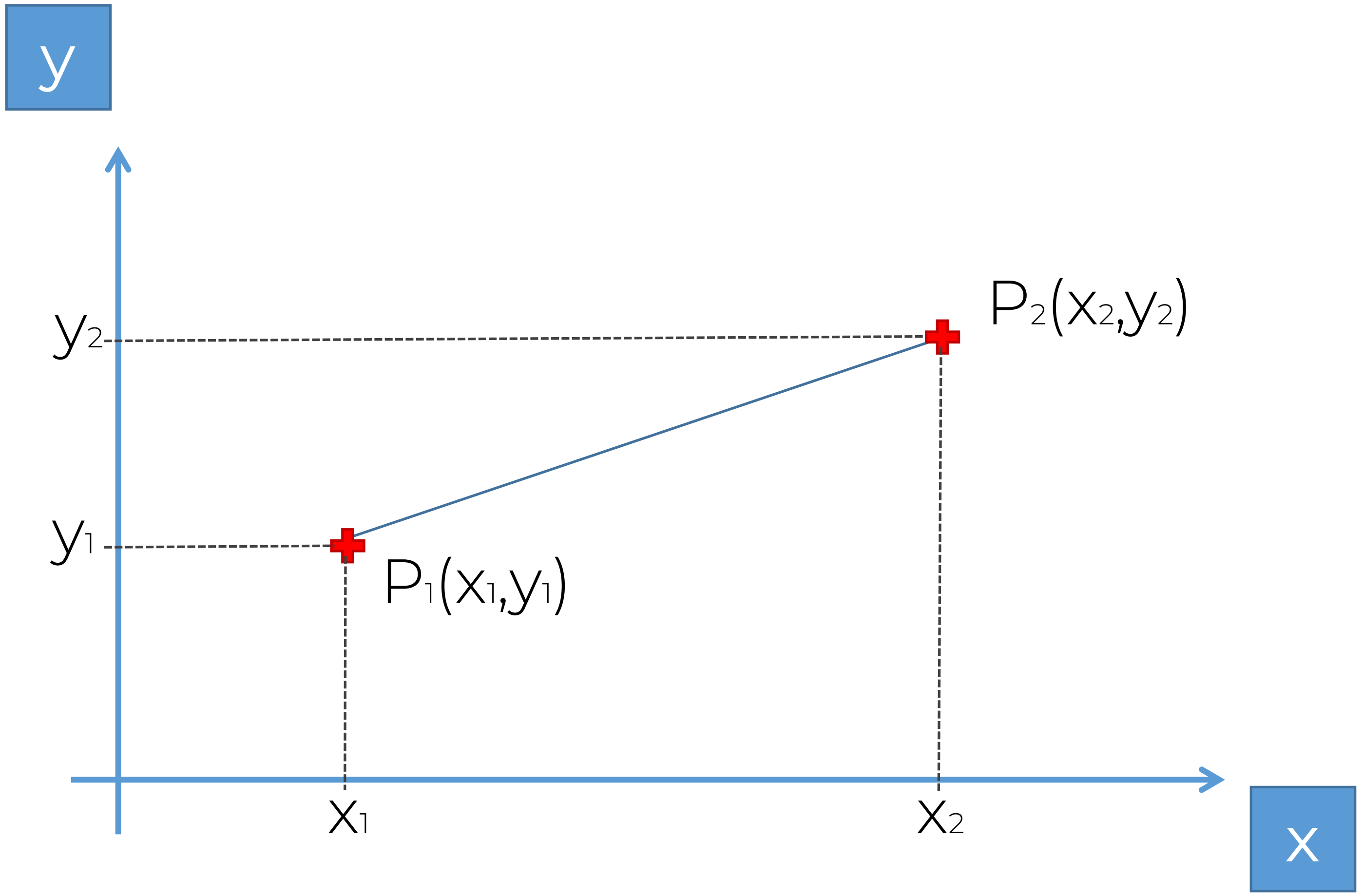
STEP 2: Take the two closest data points and make them one cluster ➡ That forms N-1 clusters

STEP 3: Take the two closest clusters and make them one cluster ➡ That forms N - 2 clusters

STEP 4: Repeat **STEP 3** until there is only one cluster

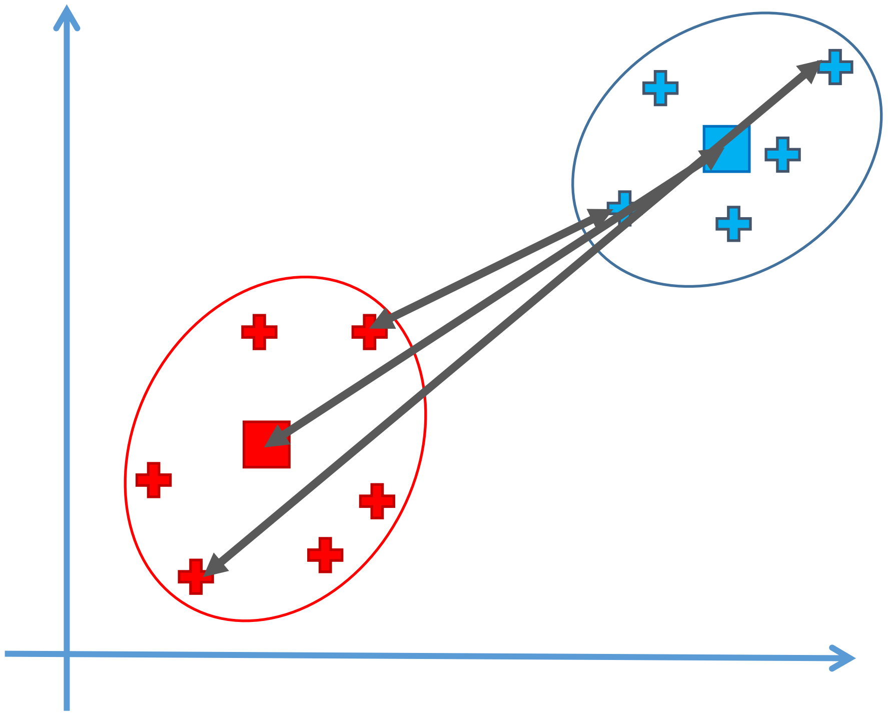
FIN

Euclidean Distance



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

Distance Between Clusters



Distance Between Two Clusters:

- Option 1: Closest Points
- Option 2: Furthest Points
- Option 3: Average Distance
- Option 4: Distance Between Centroids

Agglomerative HC

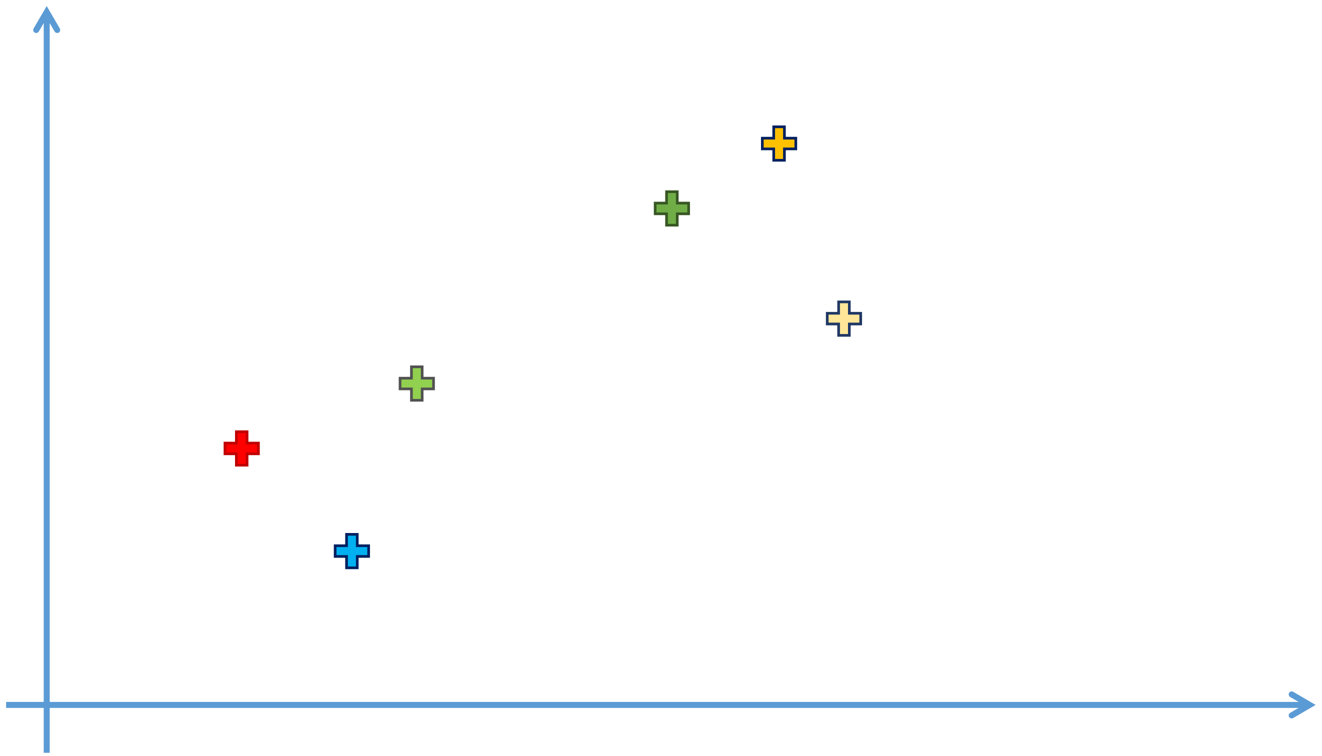
Consider the following dataset of N = 6 data points

Agglomerative HC

STEP 1: Make each data point a single-point cluster ➡ That forms 6 clusters

Agglomerative HC

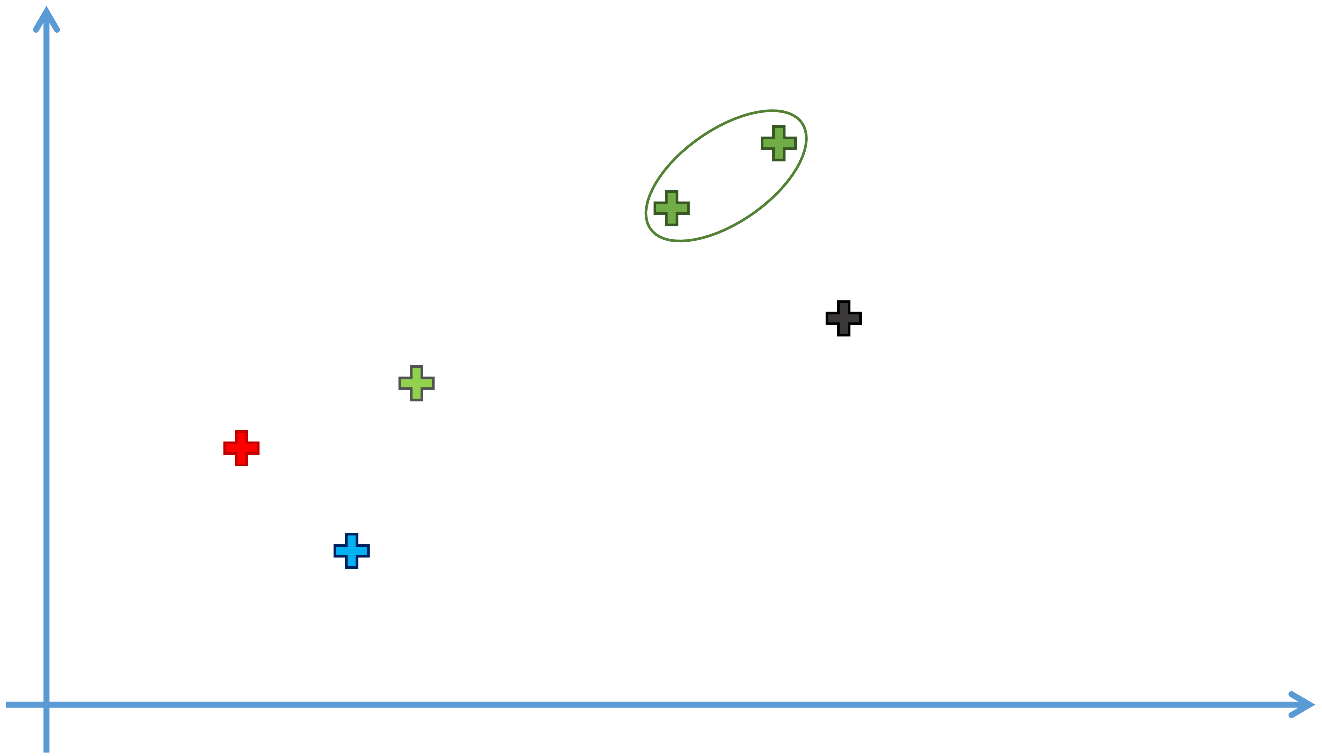
STEP 1: Make each data point a single-point cluster ➡ That forms 6 clusters



Agglomerative HC

STEP 2: Take the two closest data points and make them one cluster

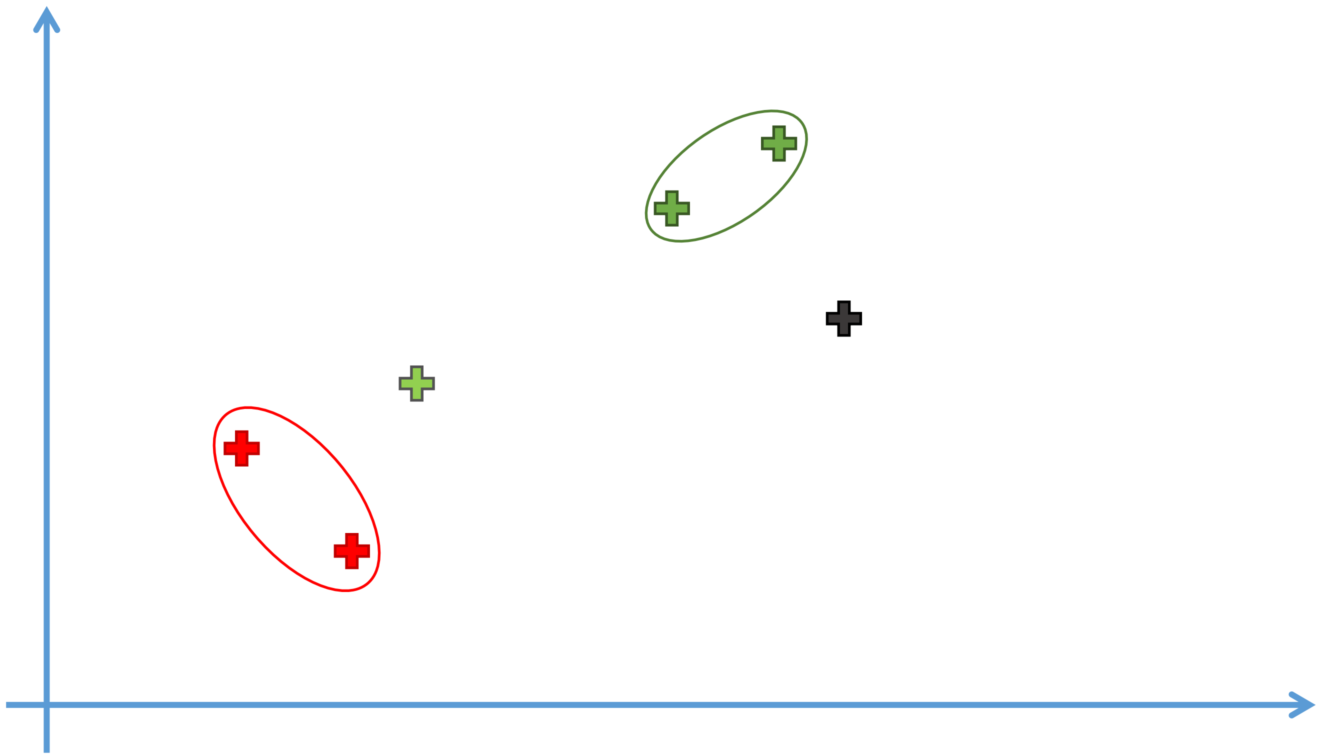
➡ That forms 5 clusters



Agglomerative HC

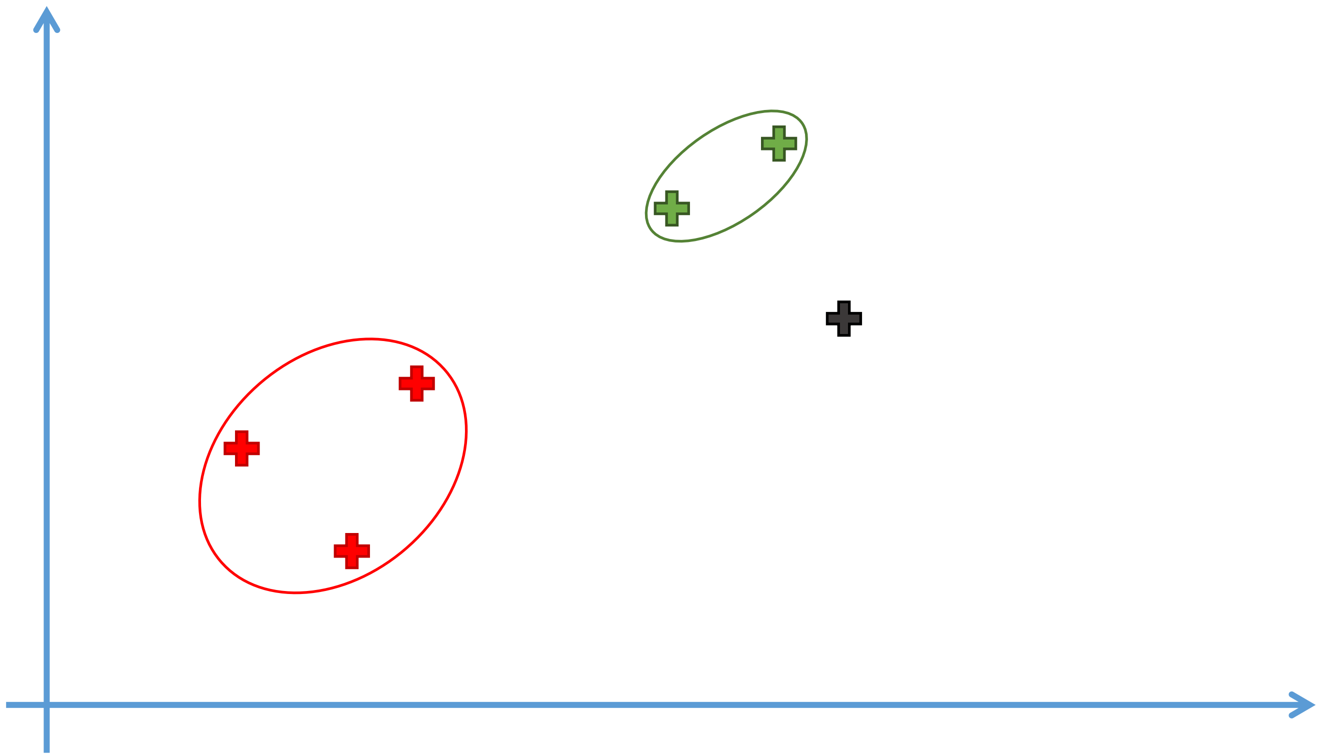
STEP 3: Take the two closest clusters and make them one cluster

➡That forms 4 clusters



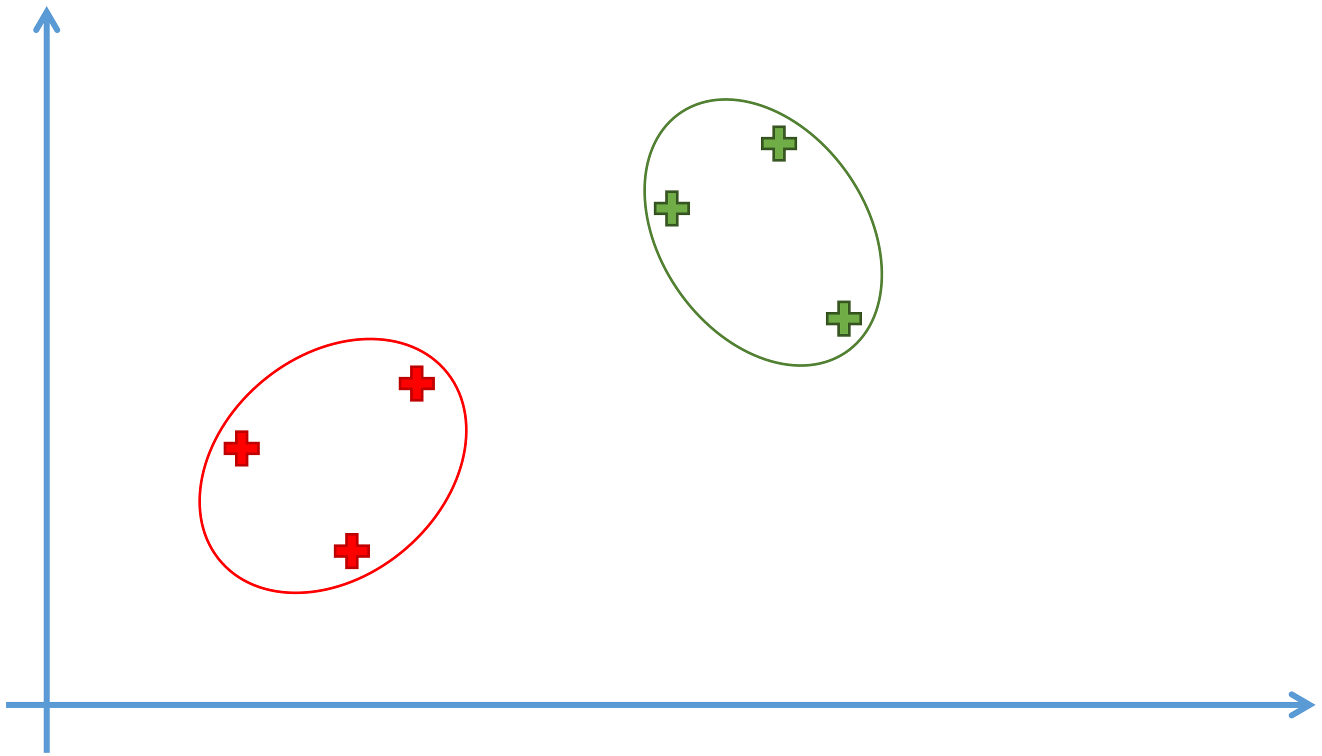
Agglomerative HC

STEP 4: Repeat STEP 3 until there is only one cluster



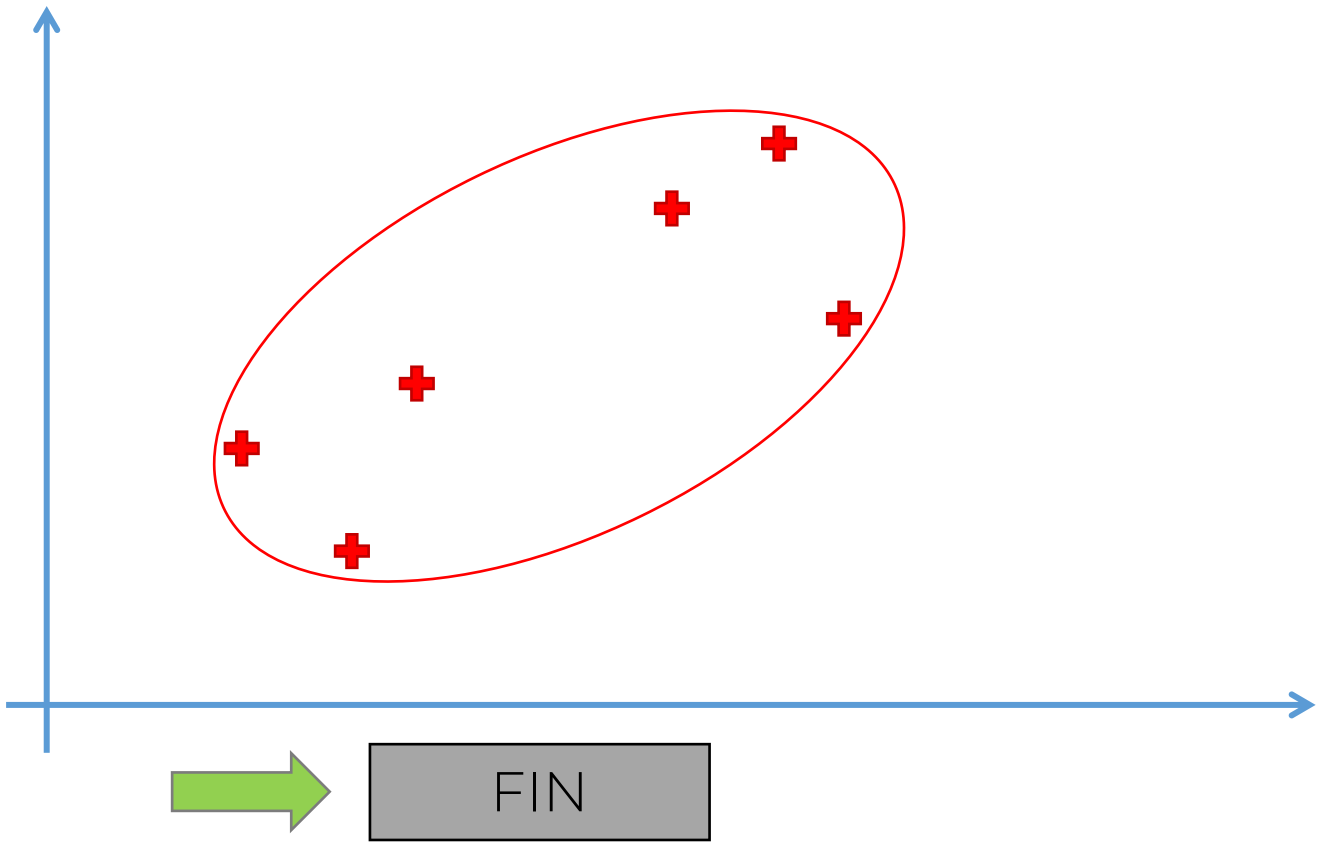
Agglomerative HC

STEP 4: Repeat STEP 3 until there is only one cluster



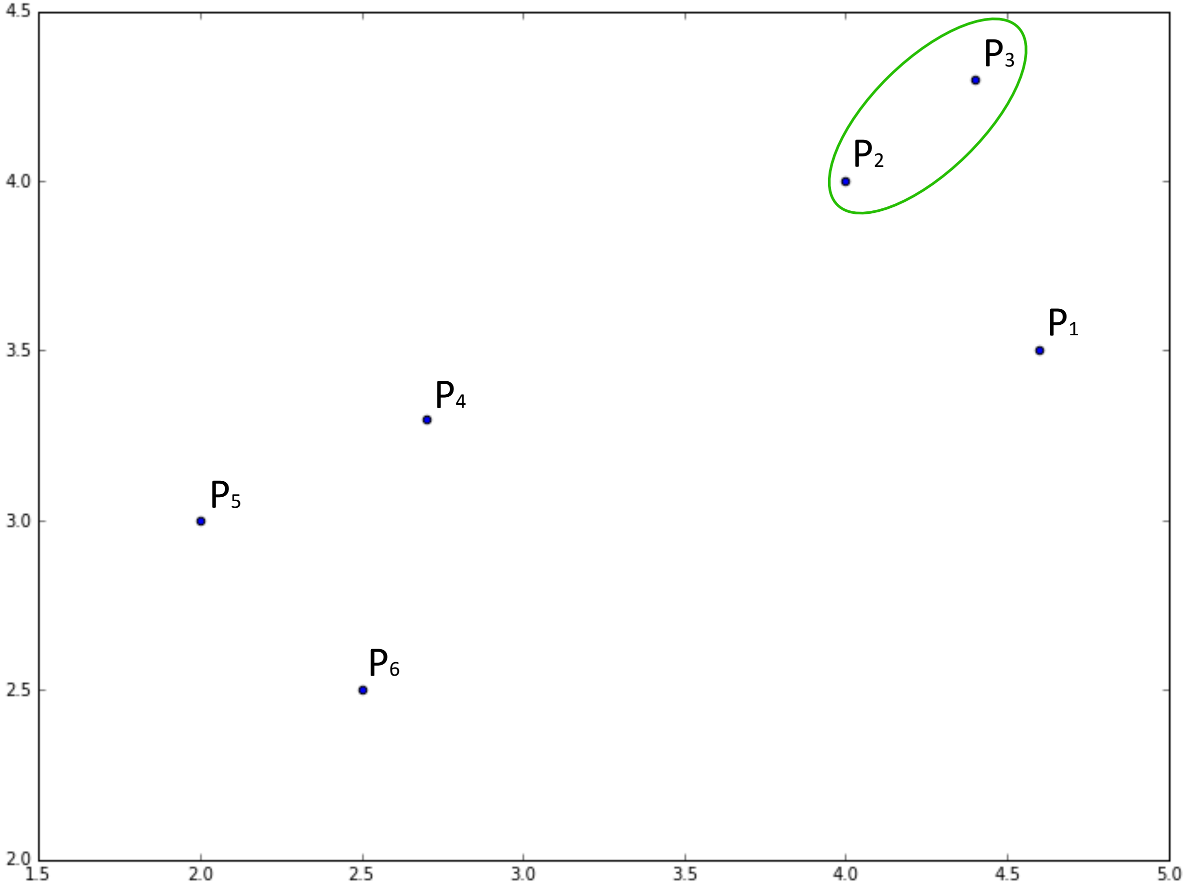
Agglomerative HC

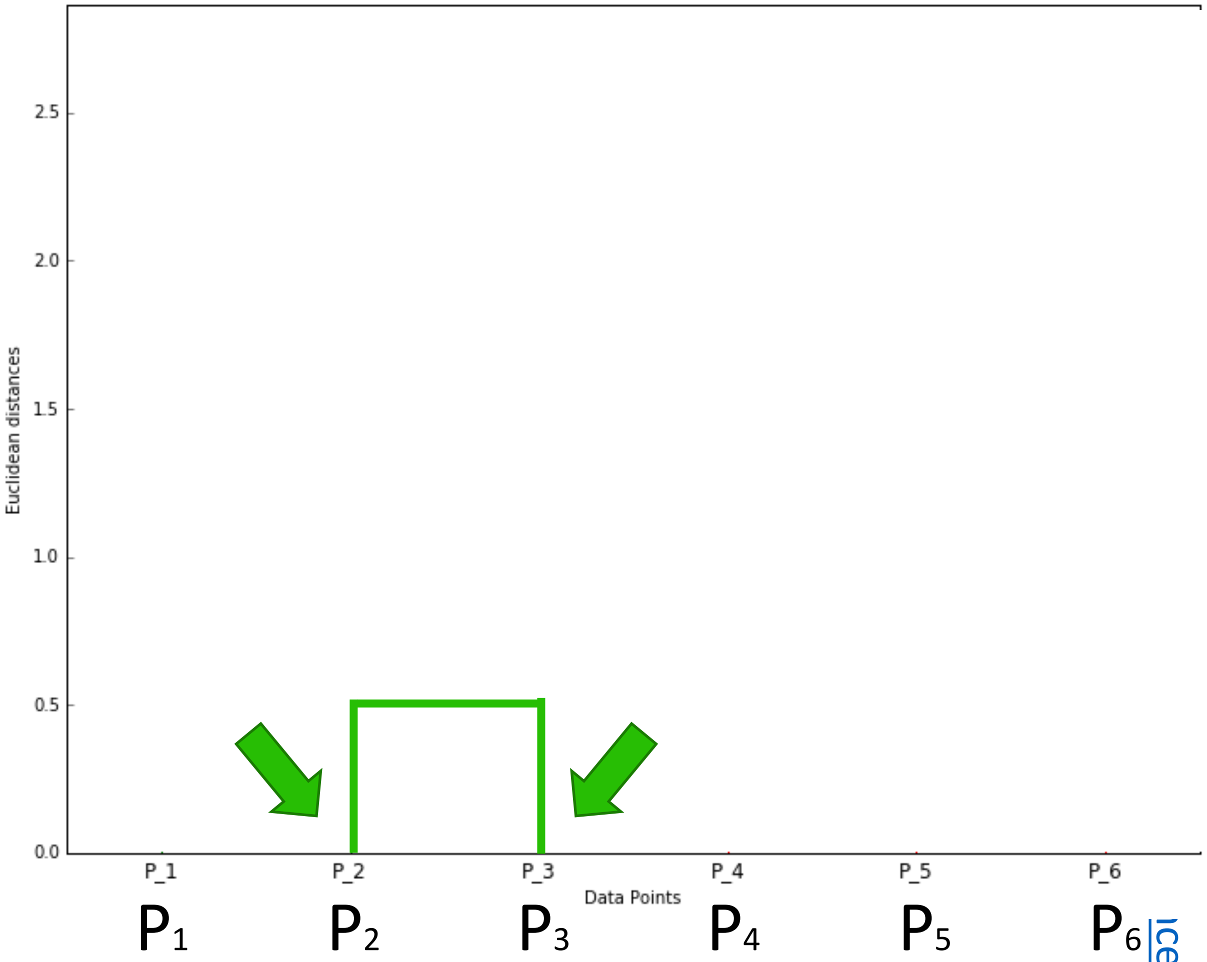
STEP 4: Repeat STEP 3 until there is only one cluster



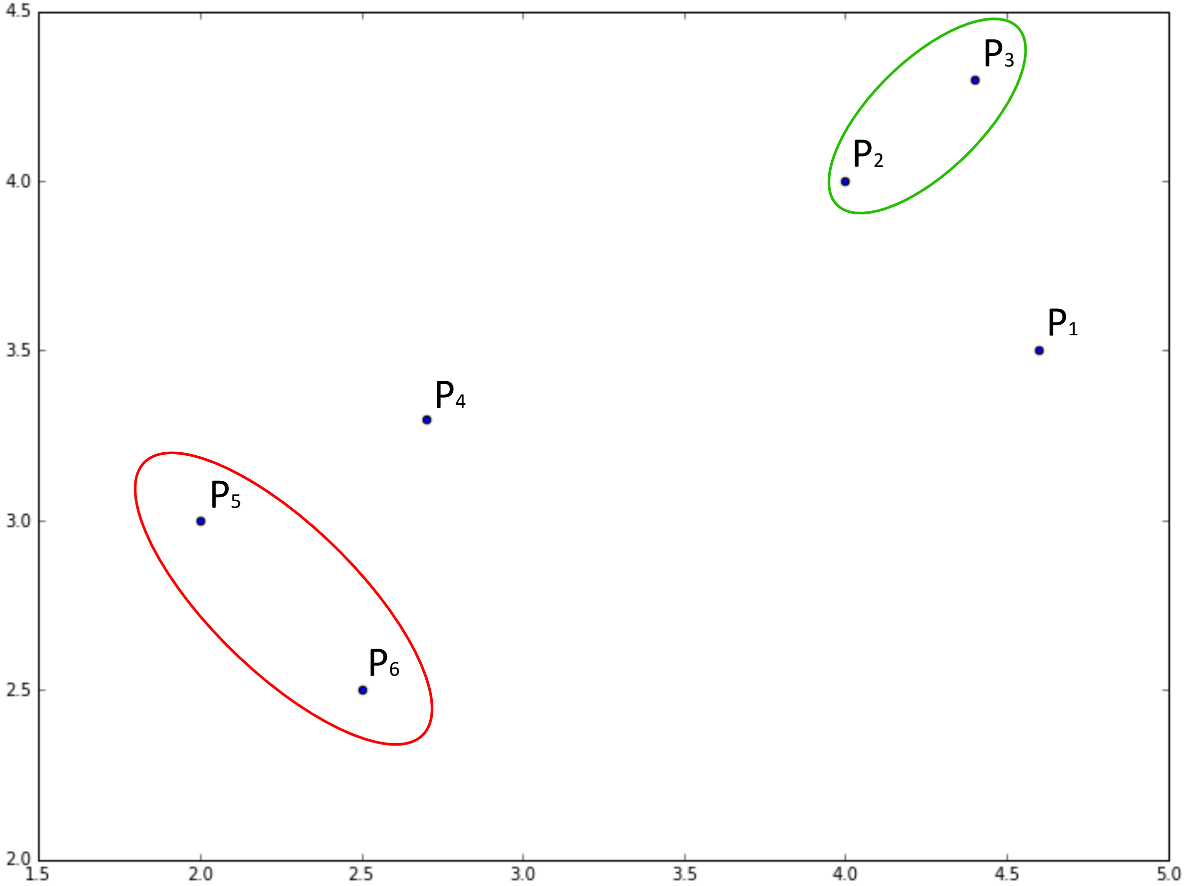
HC Intuition: How Do Dendograms Work?

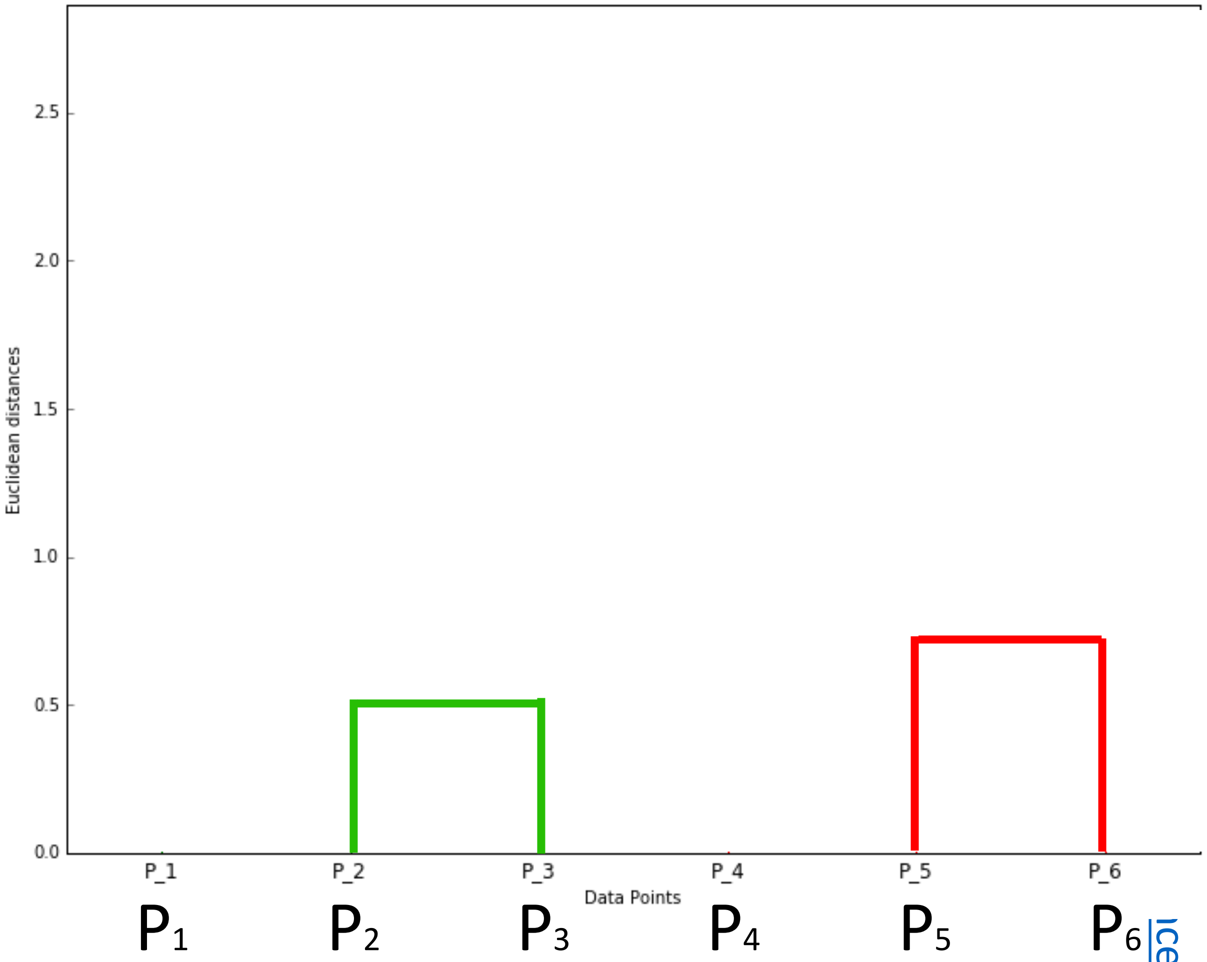
How Do Dendograms Work?



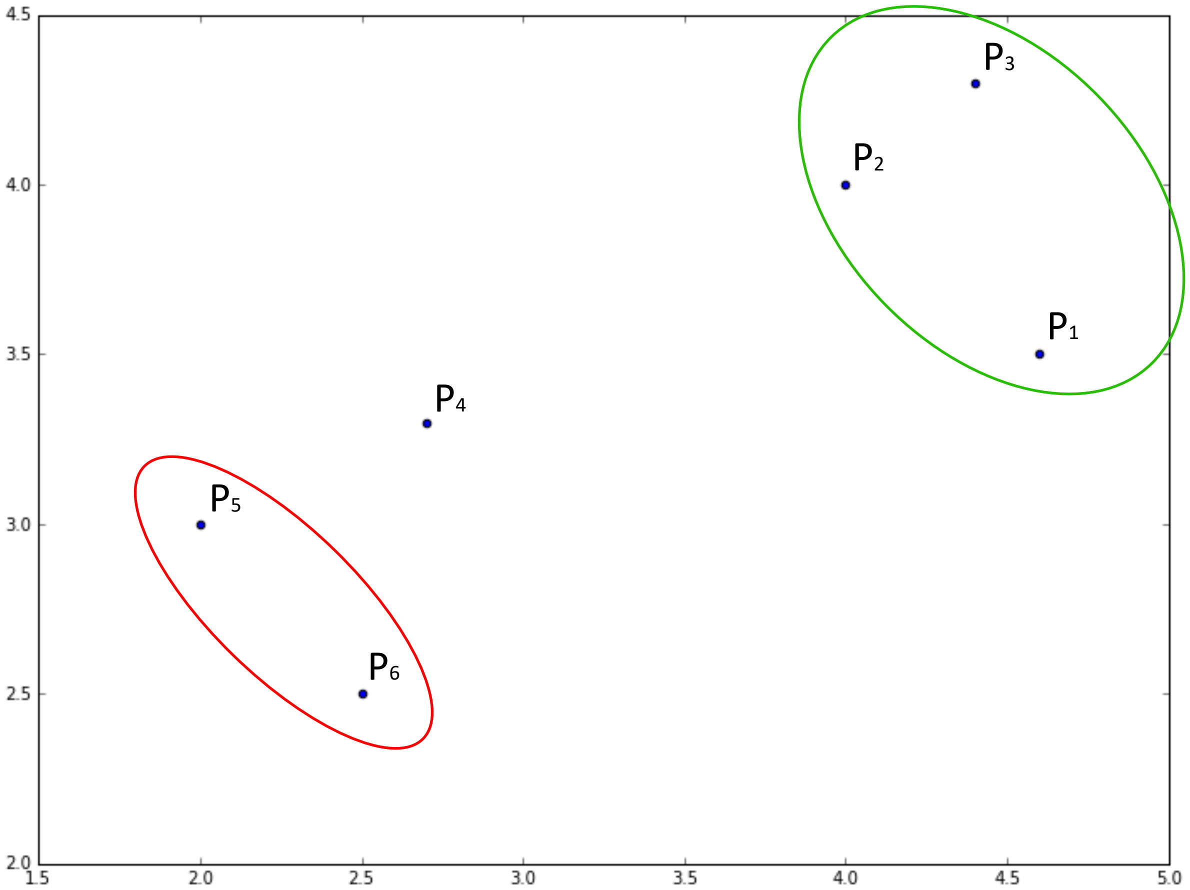


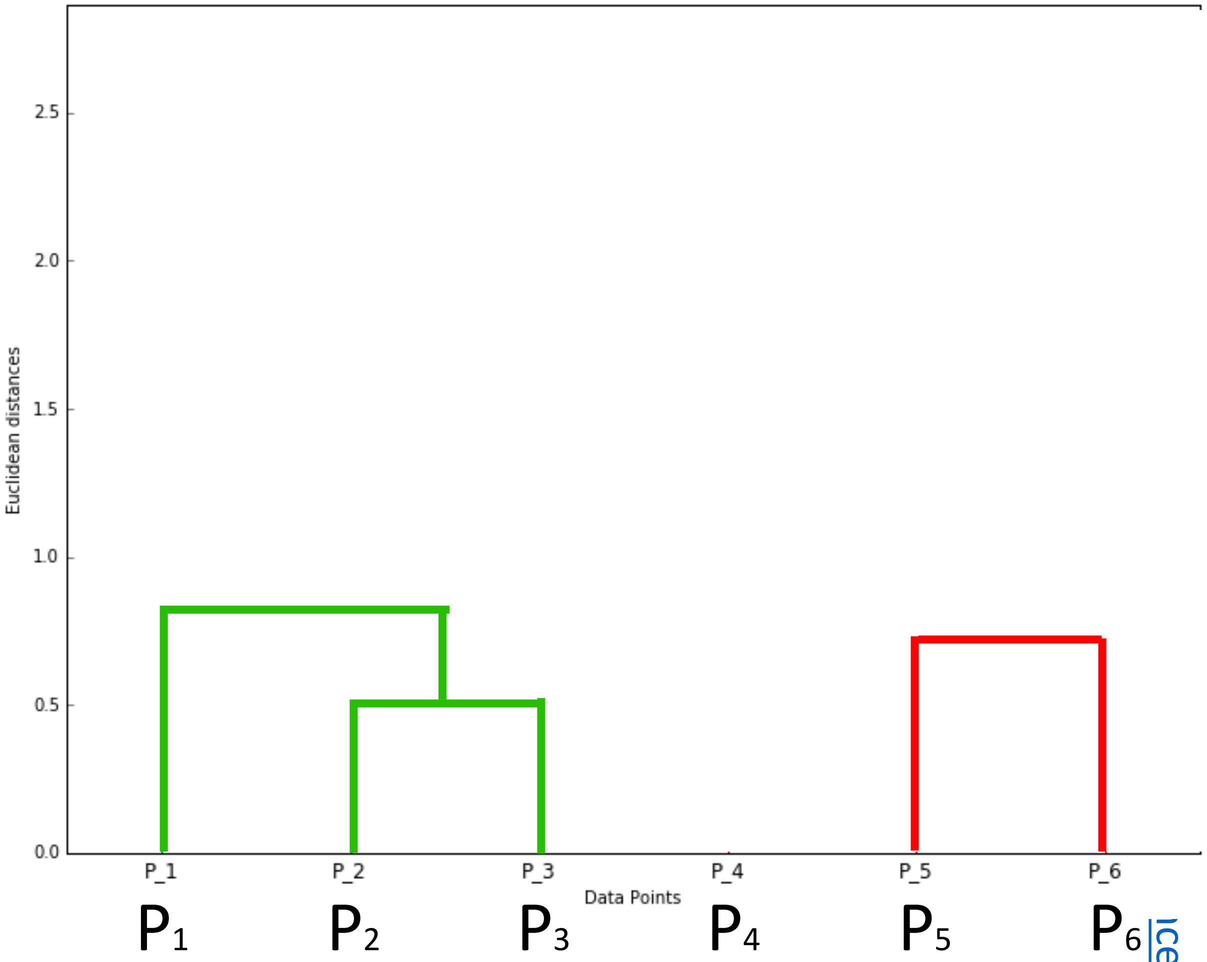
How Do Dendograms Work?



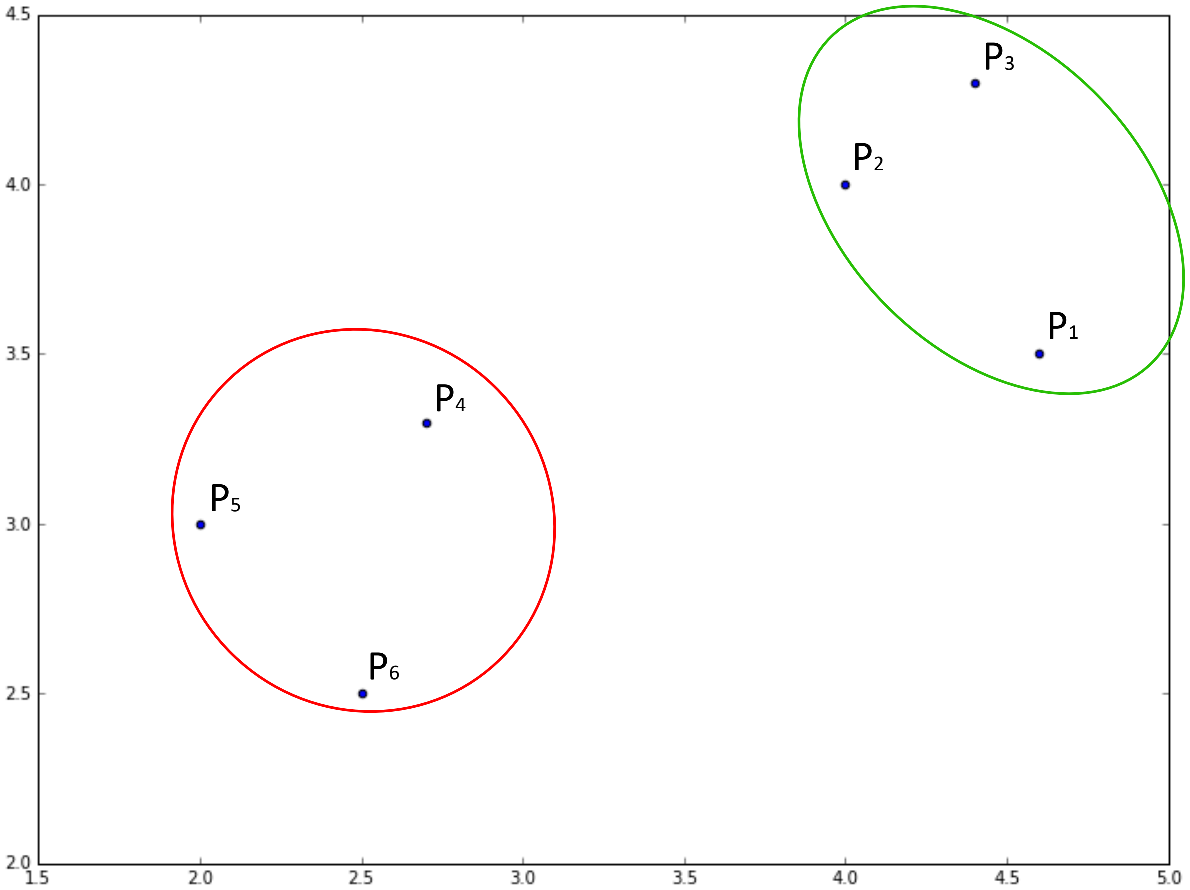


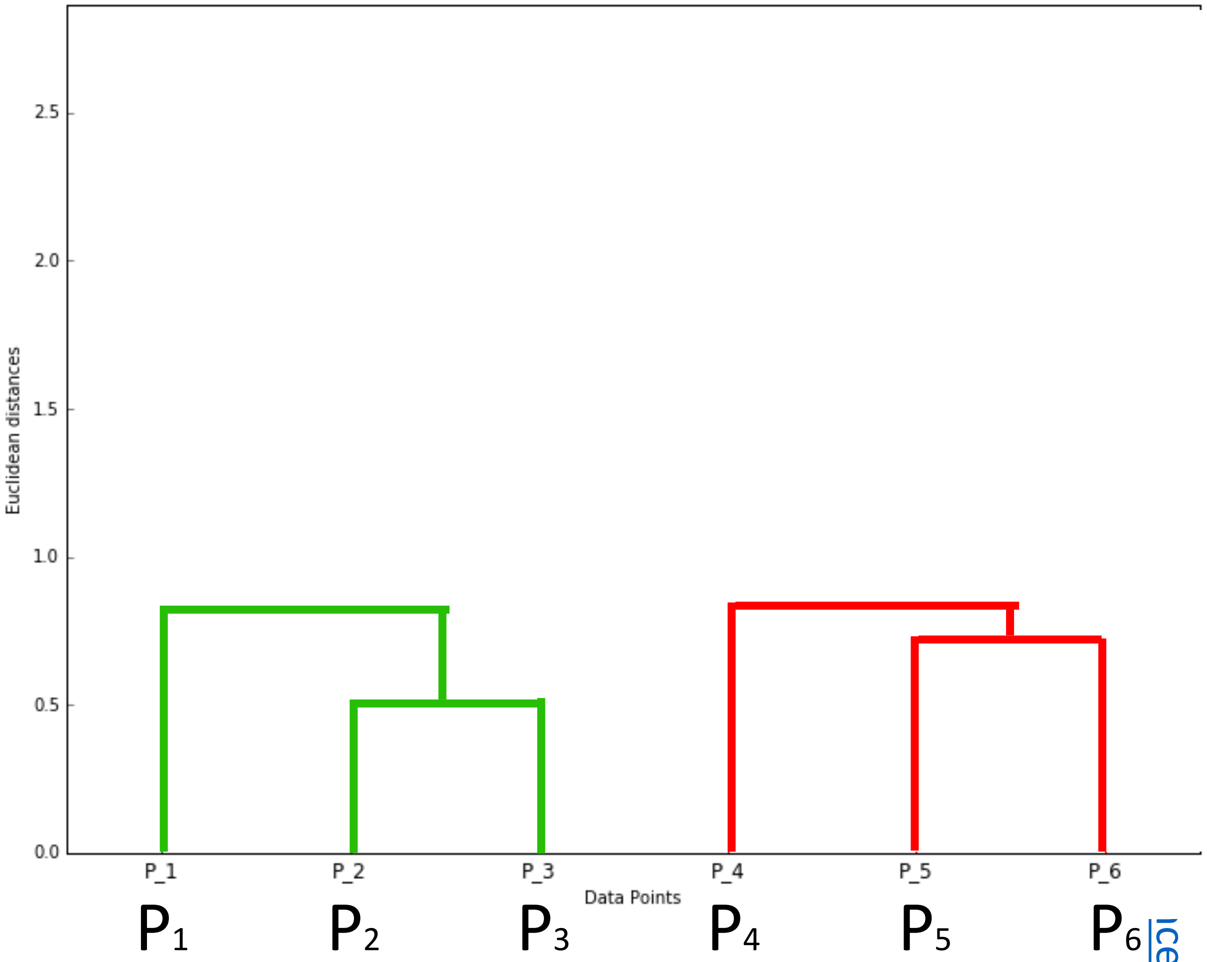
How Do Dendograms Work?



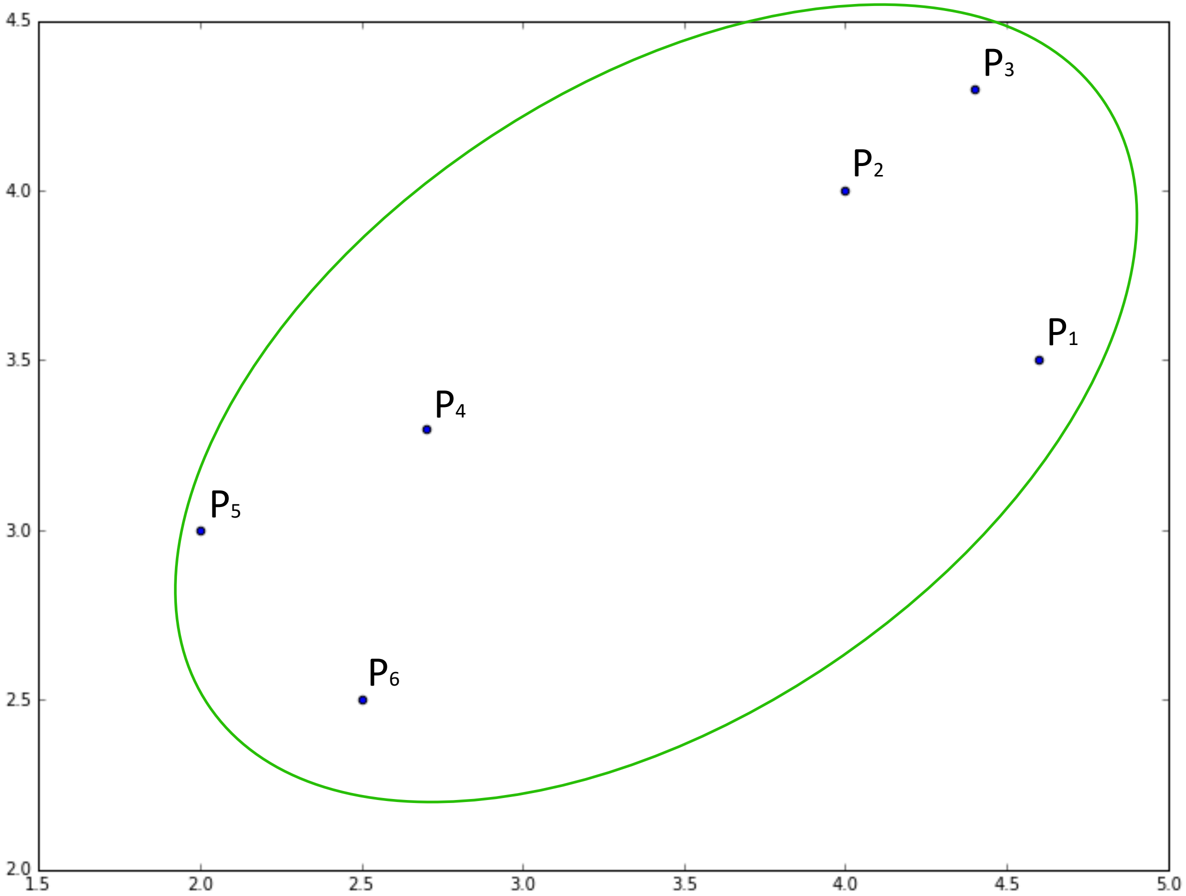


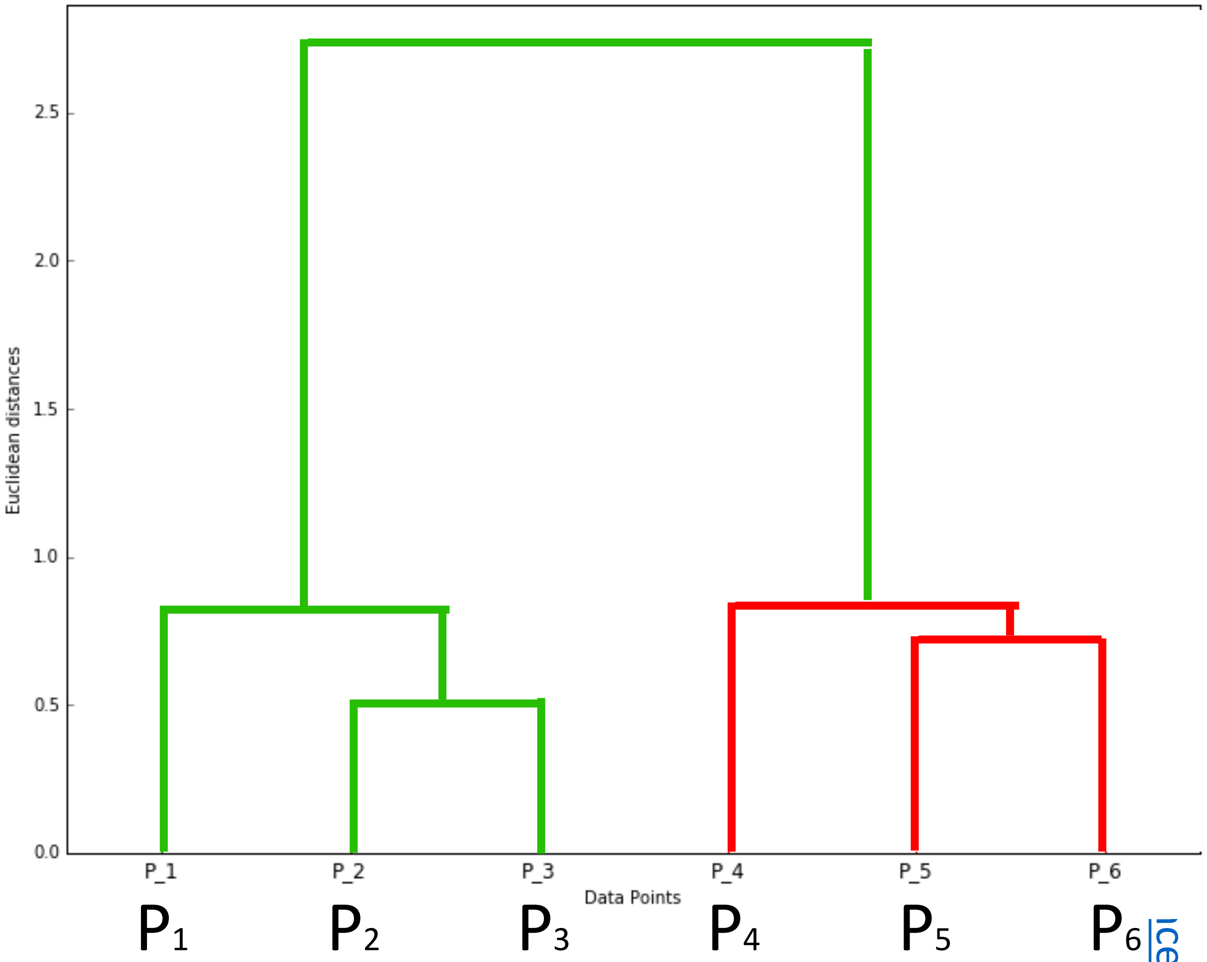
How Do Dendograms Work?



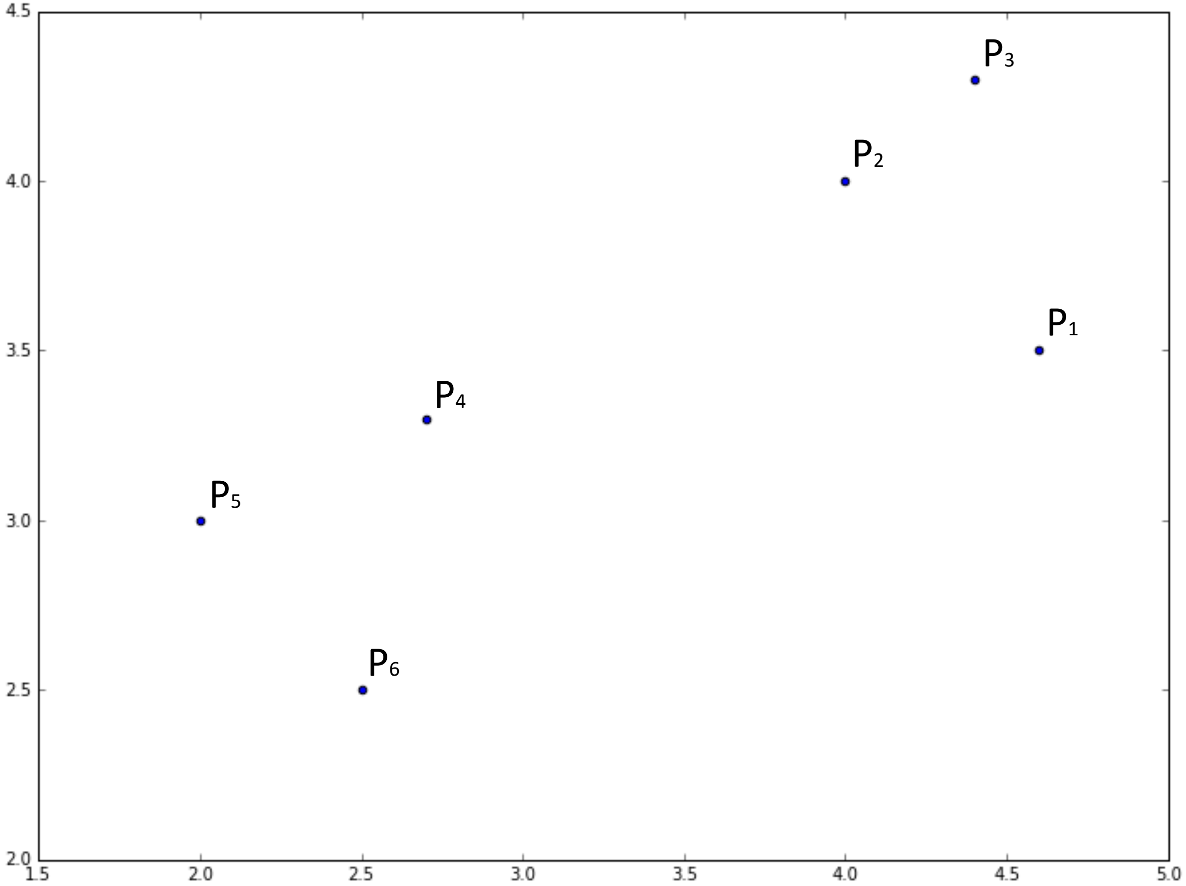


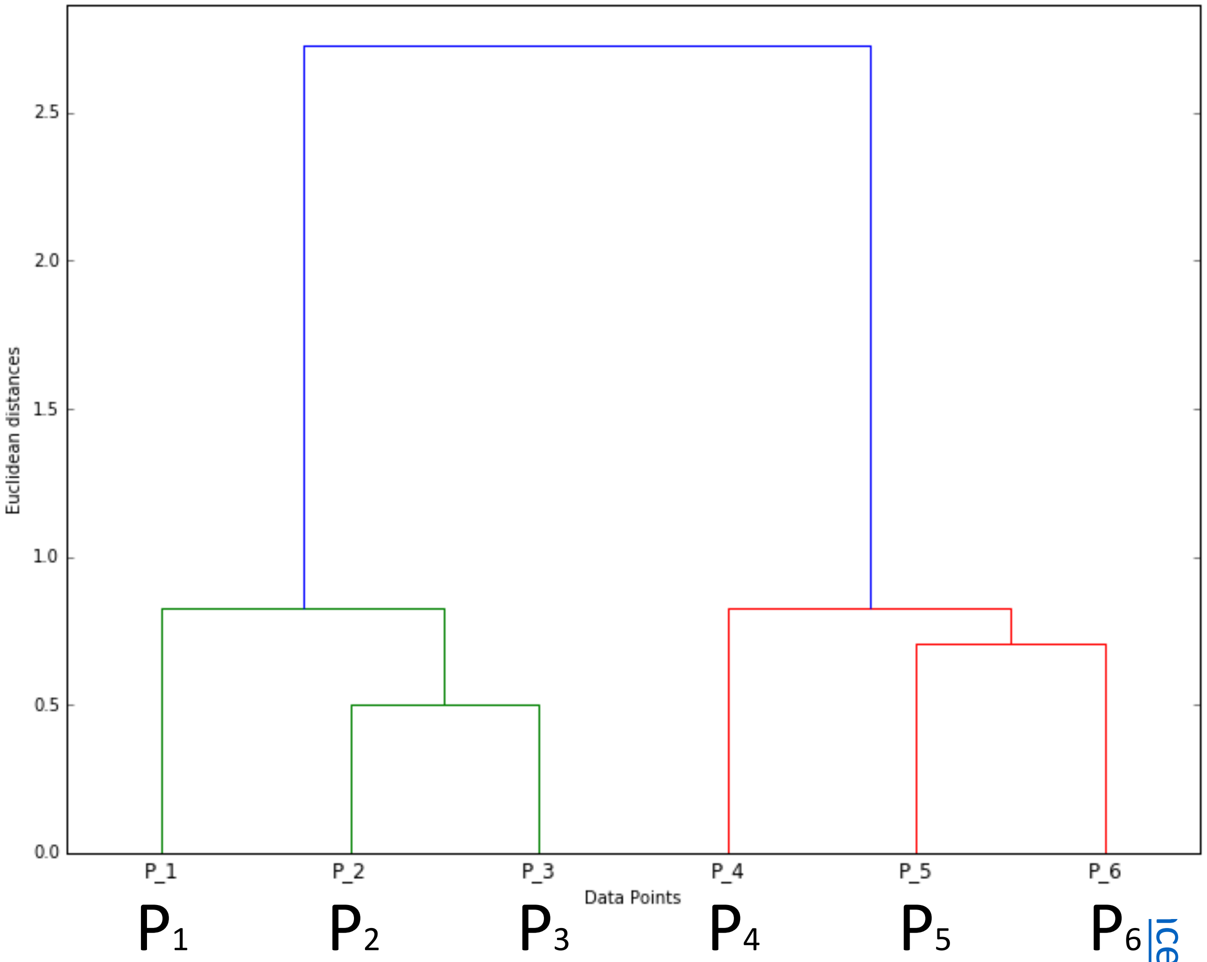
How Do Dendograms Work?





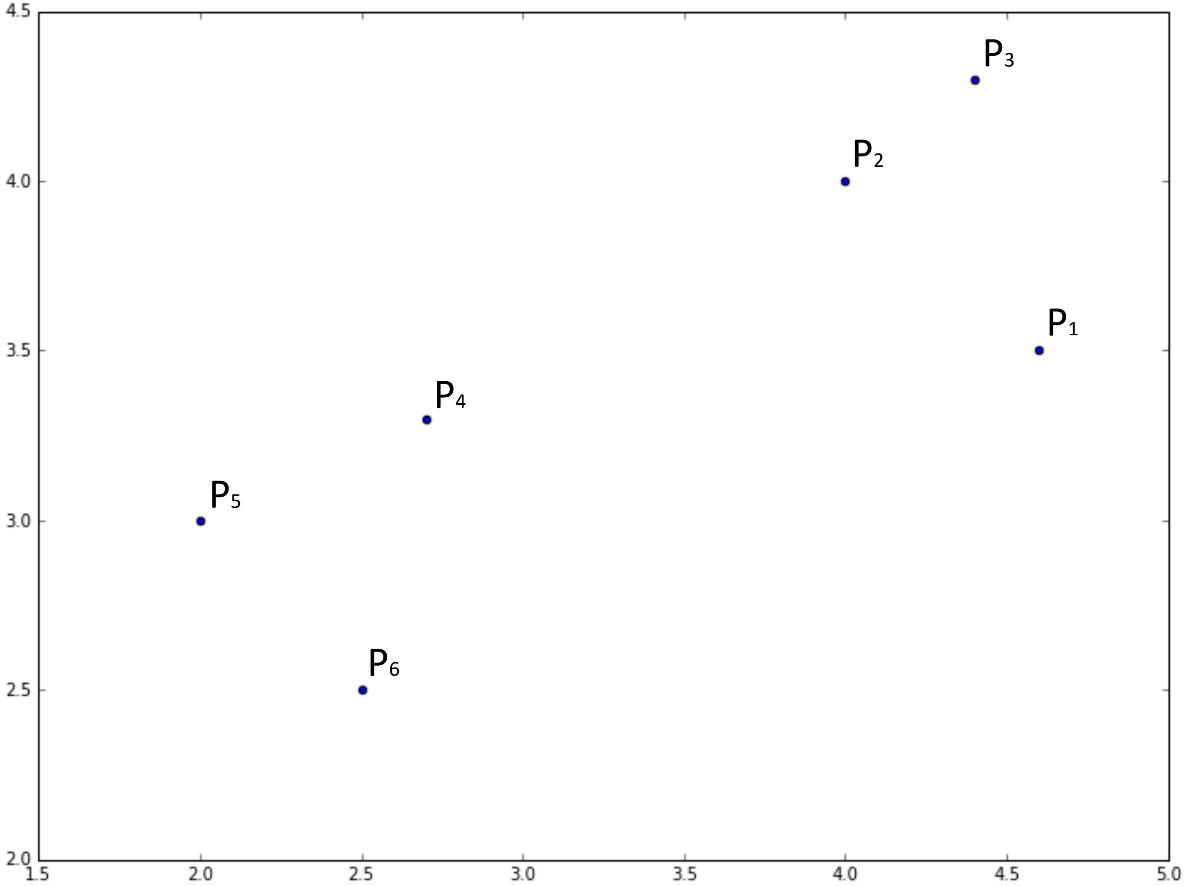
How Do Dendograms Work?

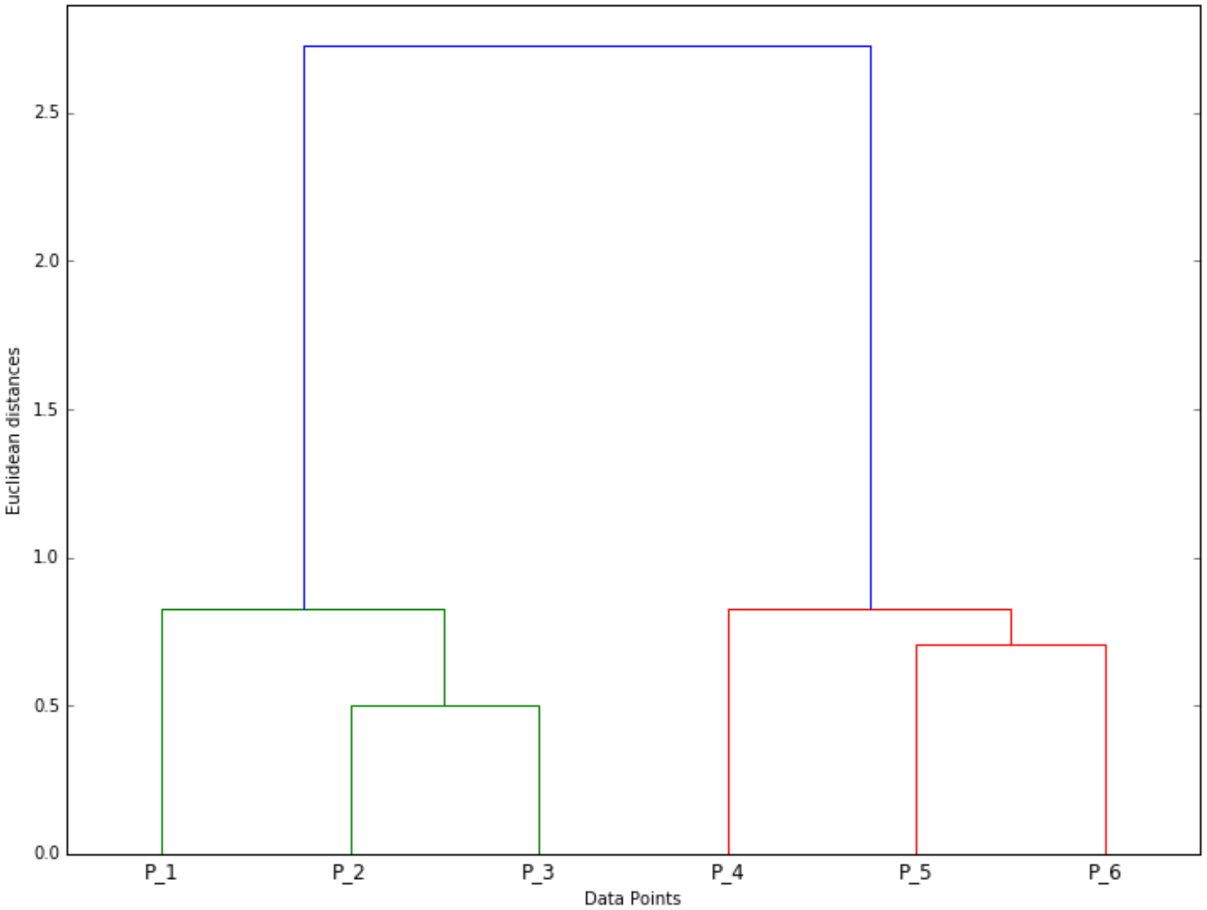




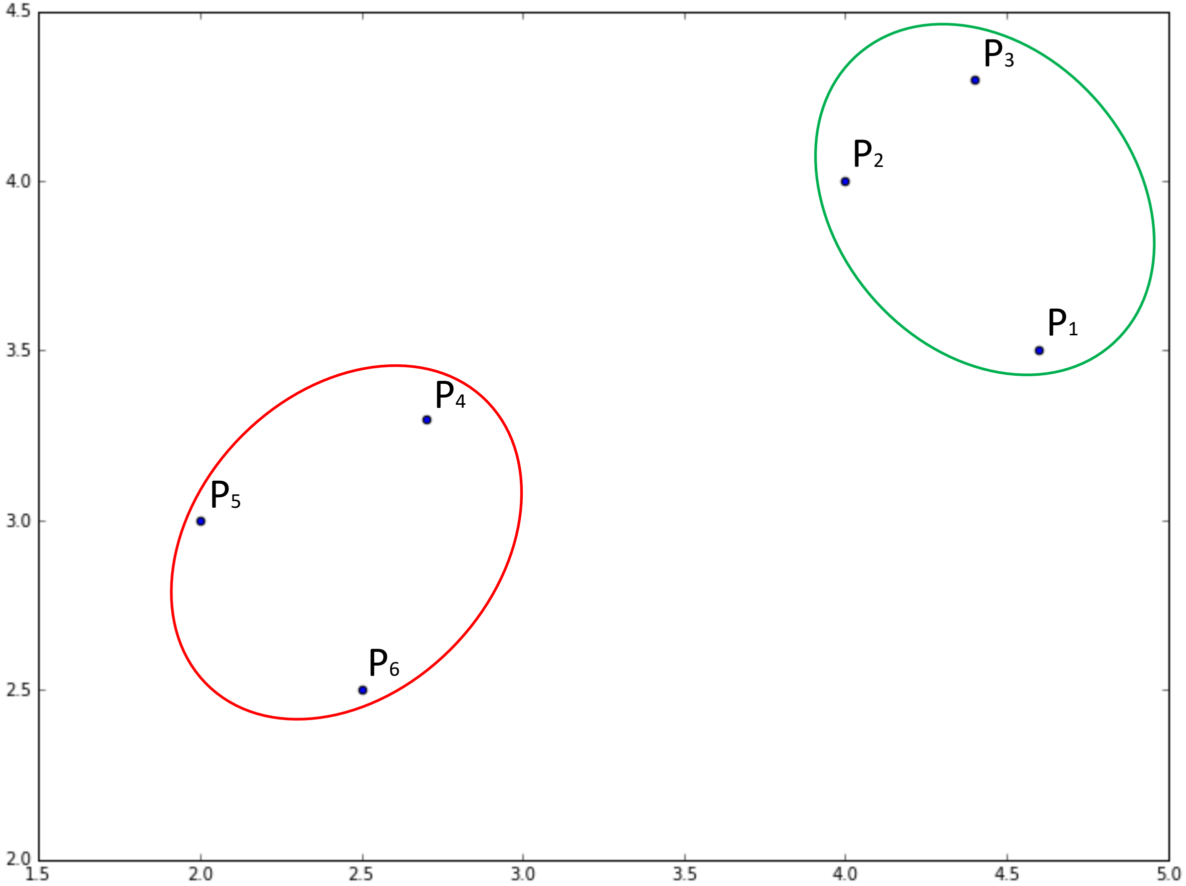
HC Intuition:
Using Dendrograms

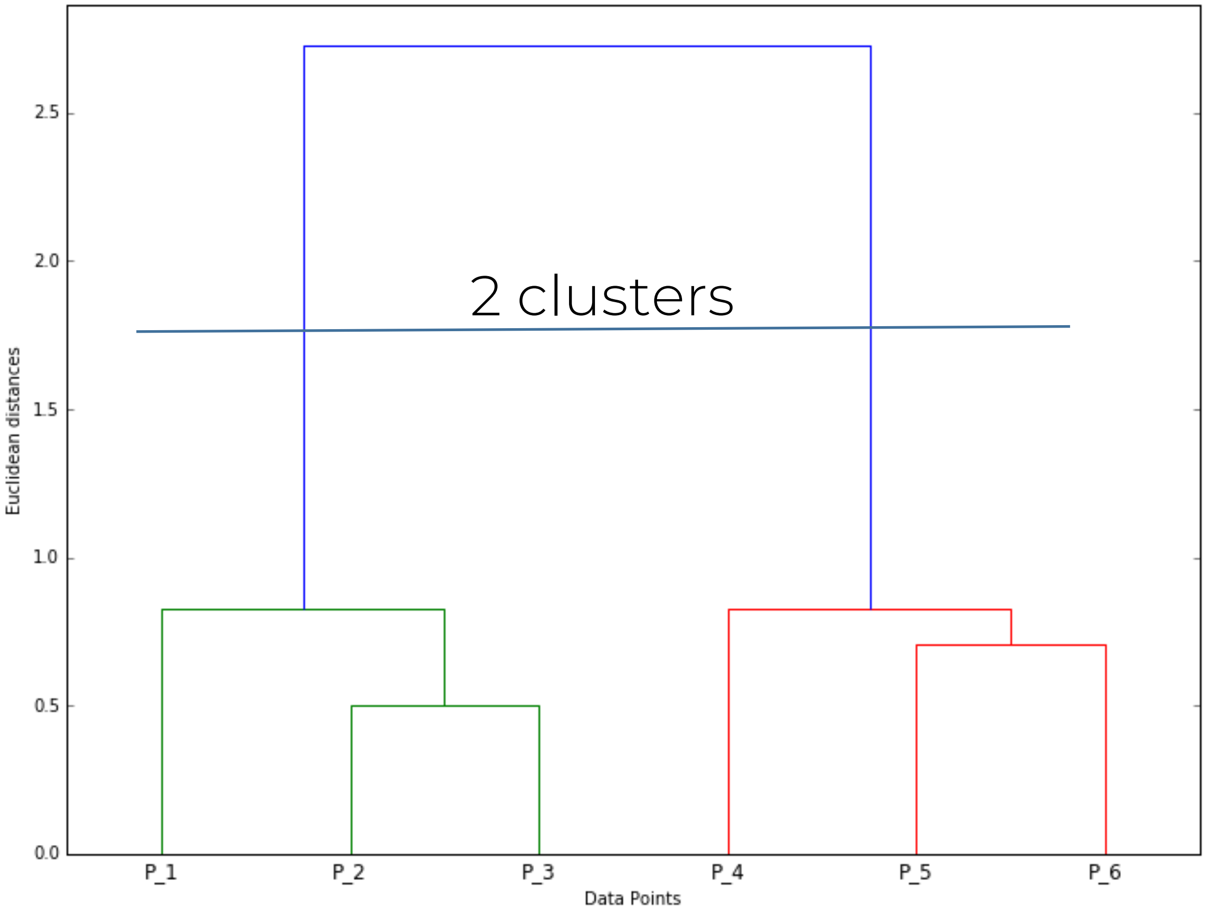
Dendrograms – Two Clusters



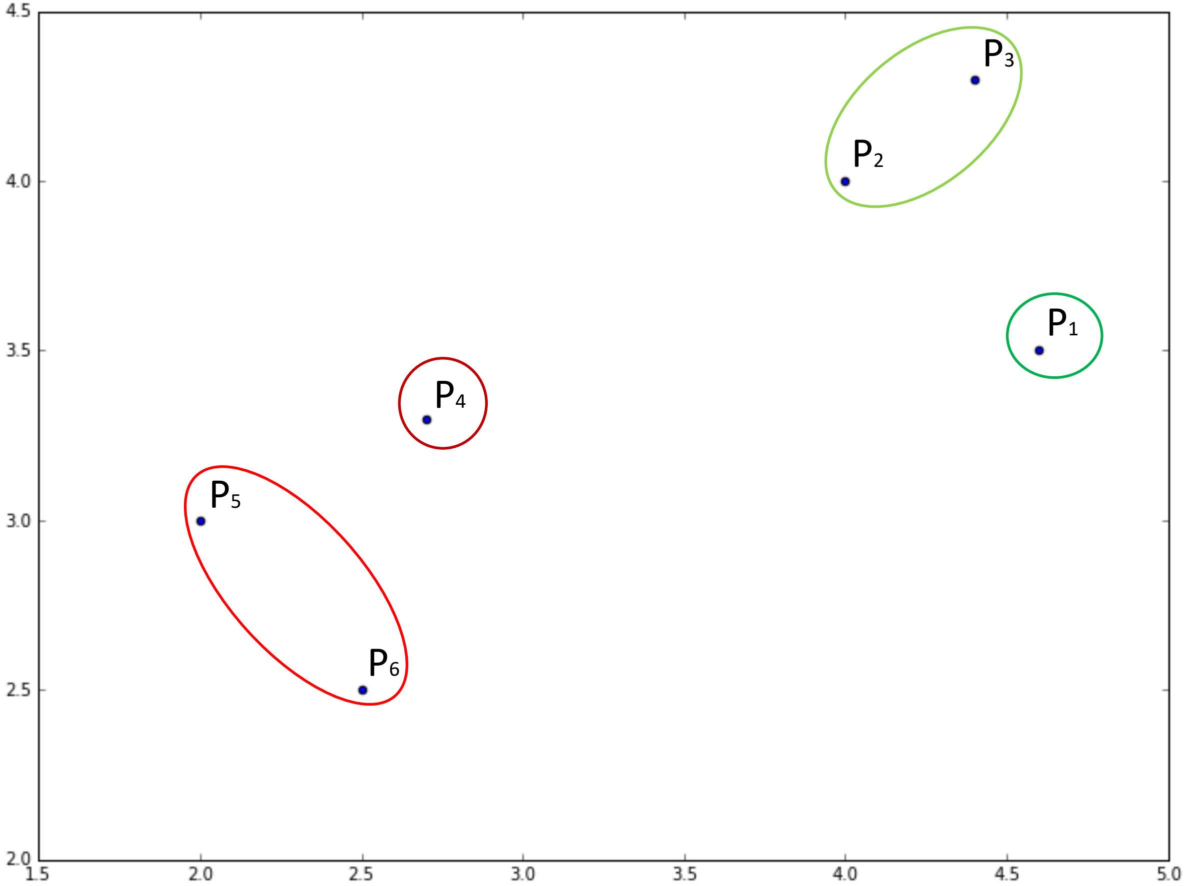


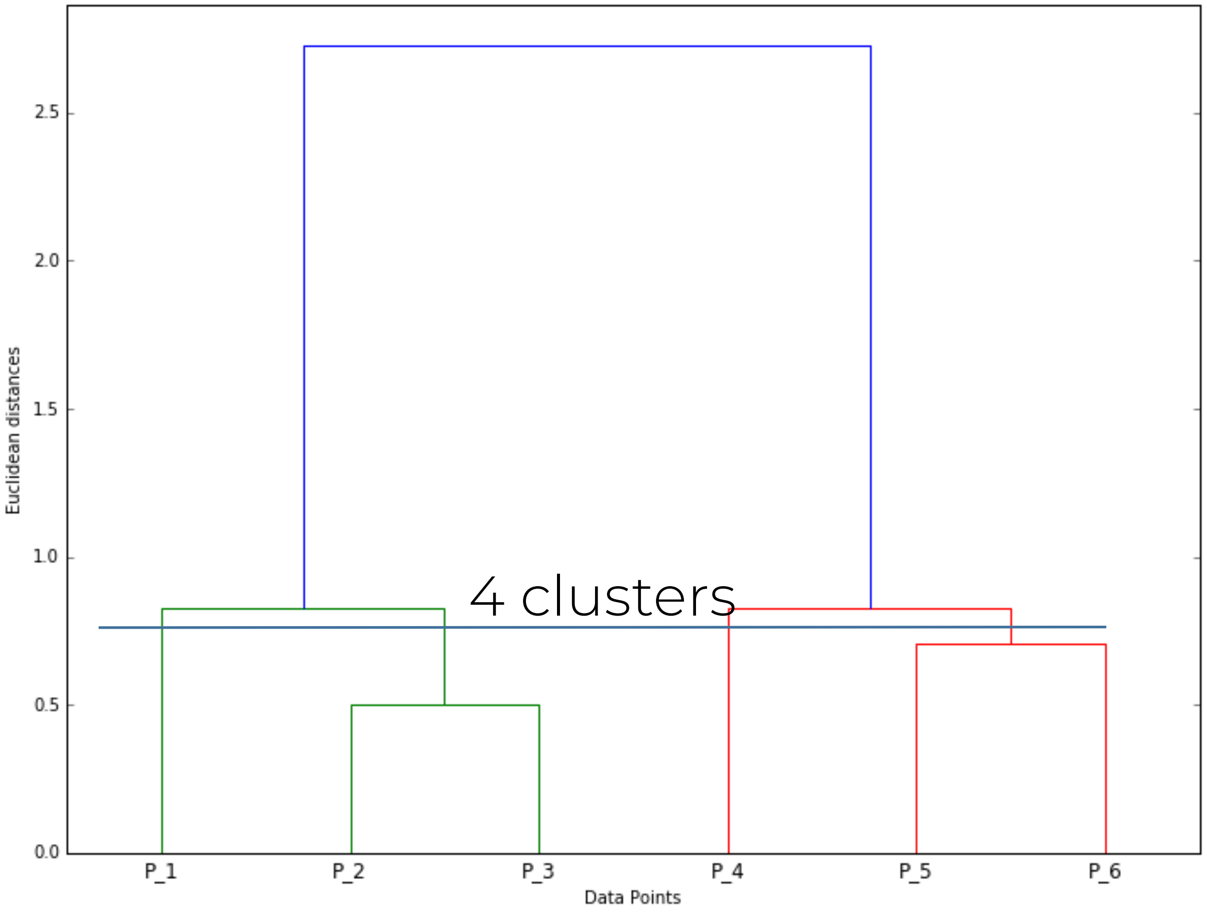
Dendrograms – Two Clusters



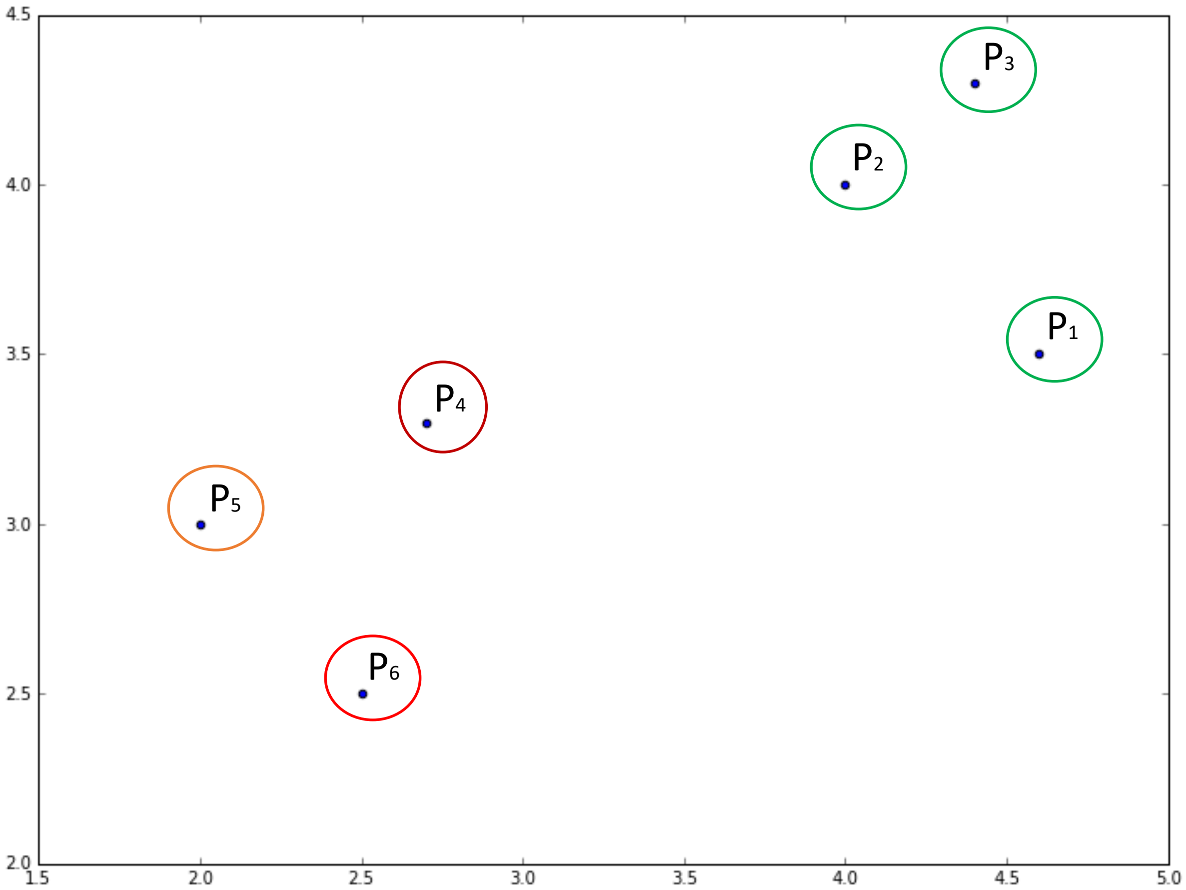


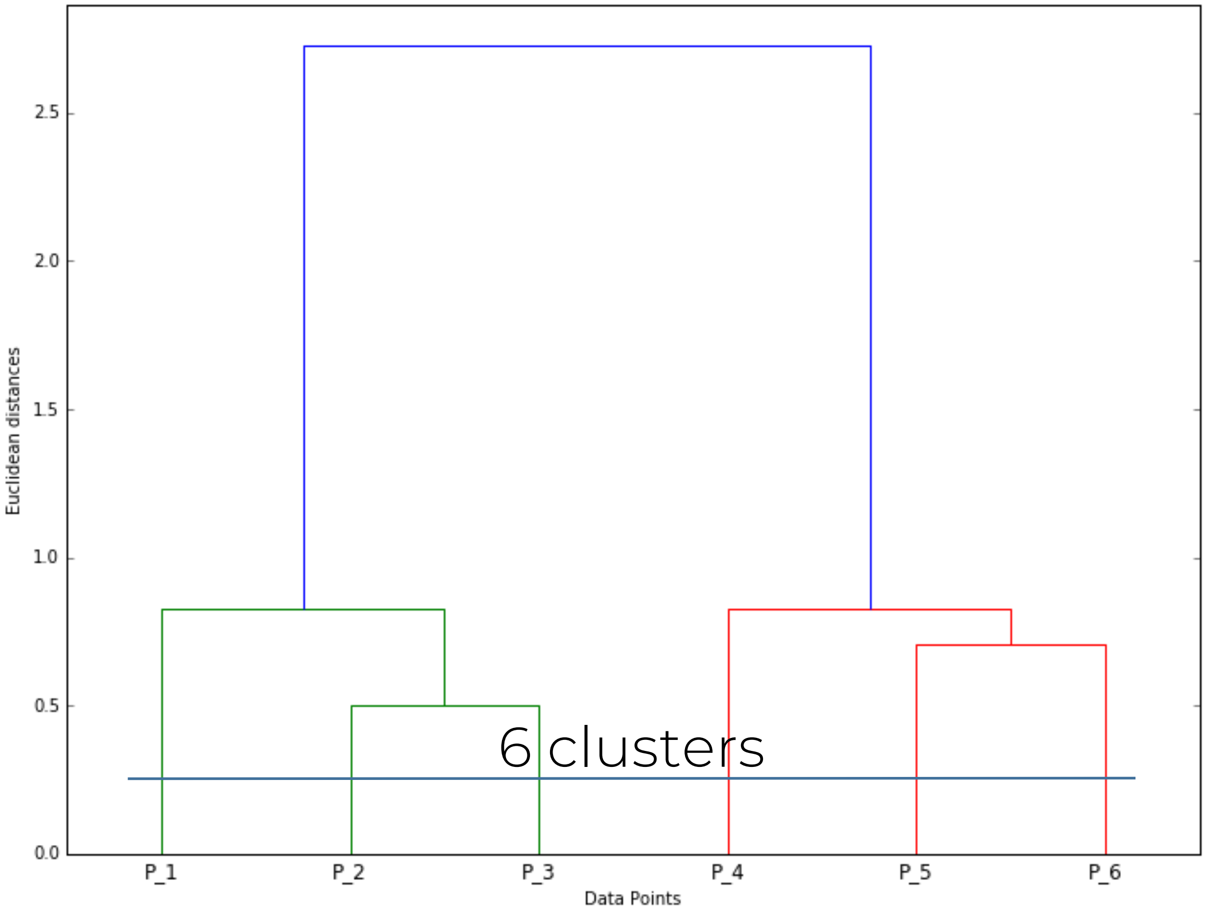
Dendrograms – Four Clusters



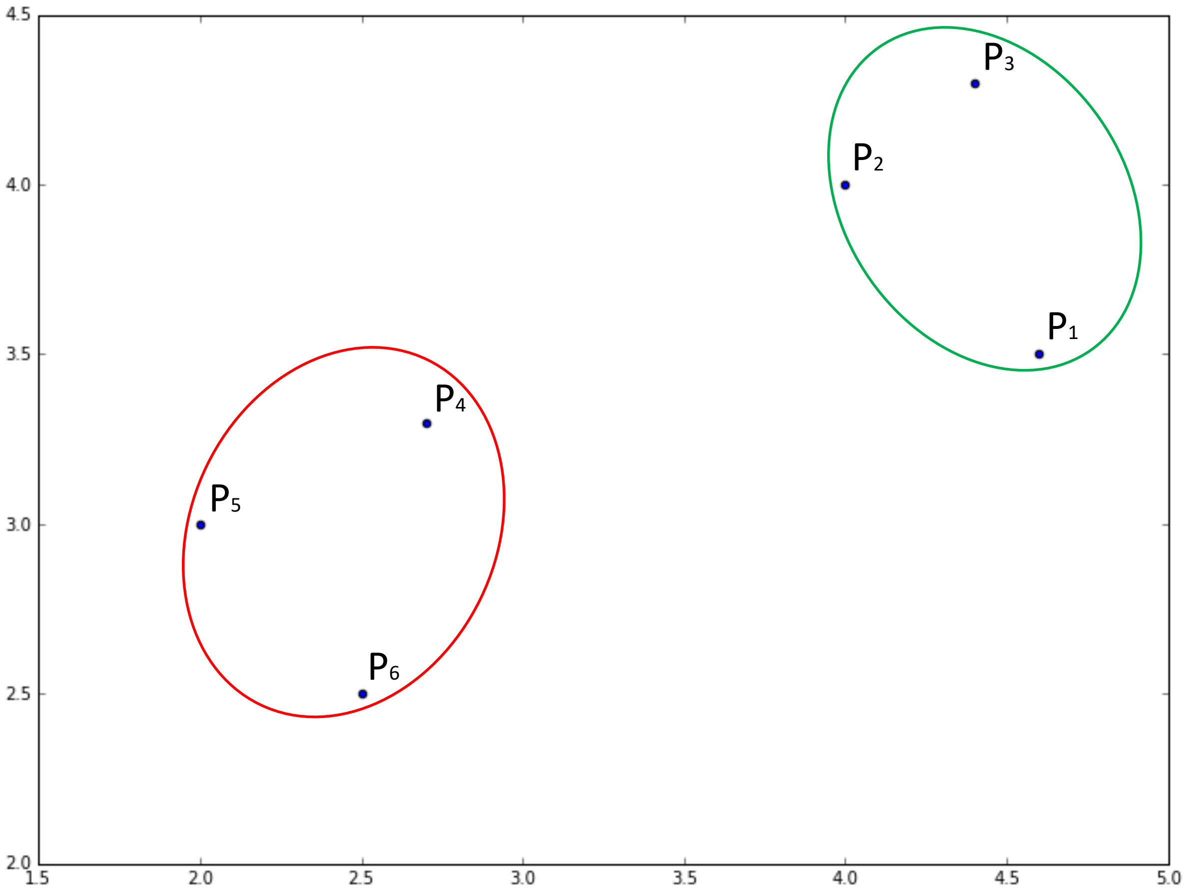


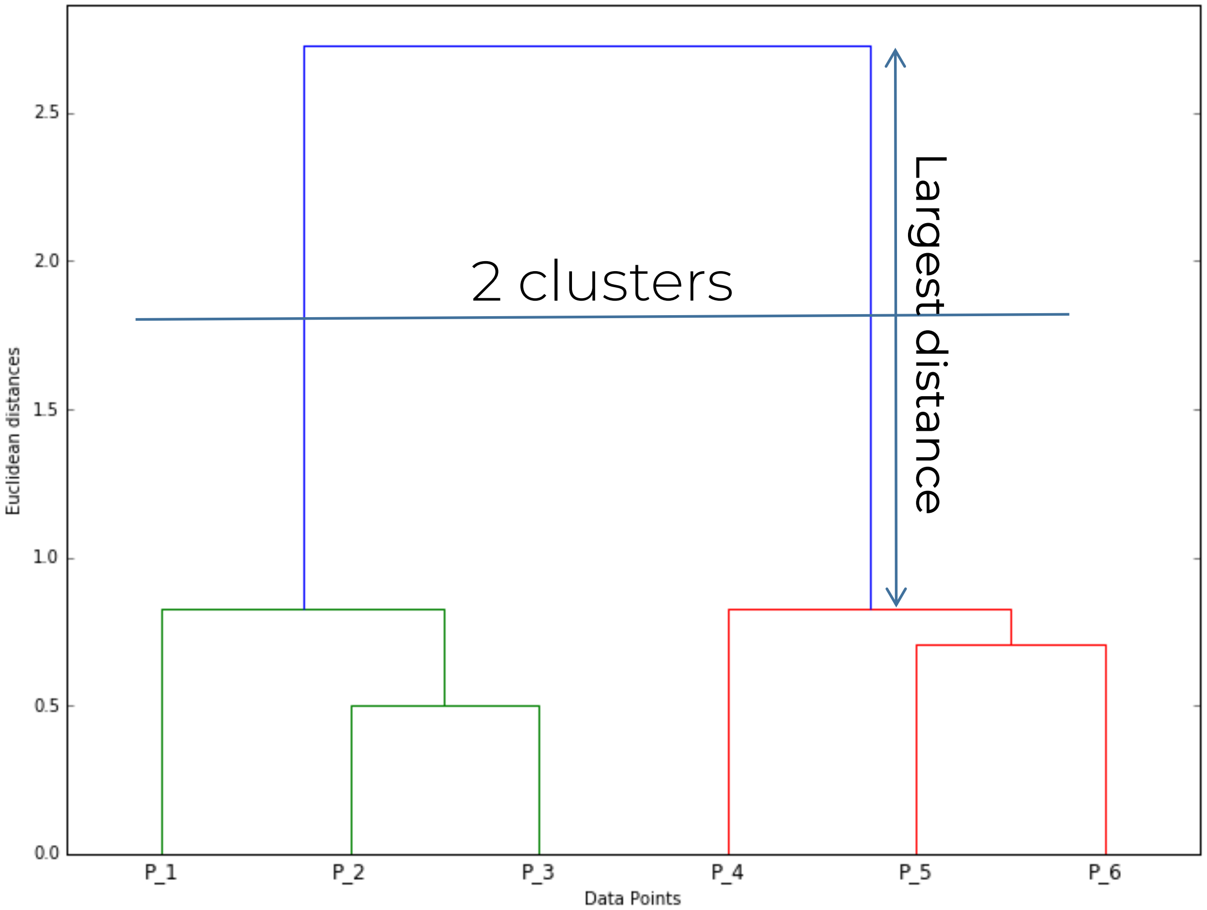
Dendrograms – Six Clusters





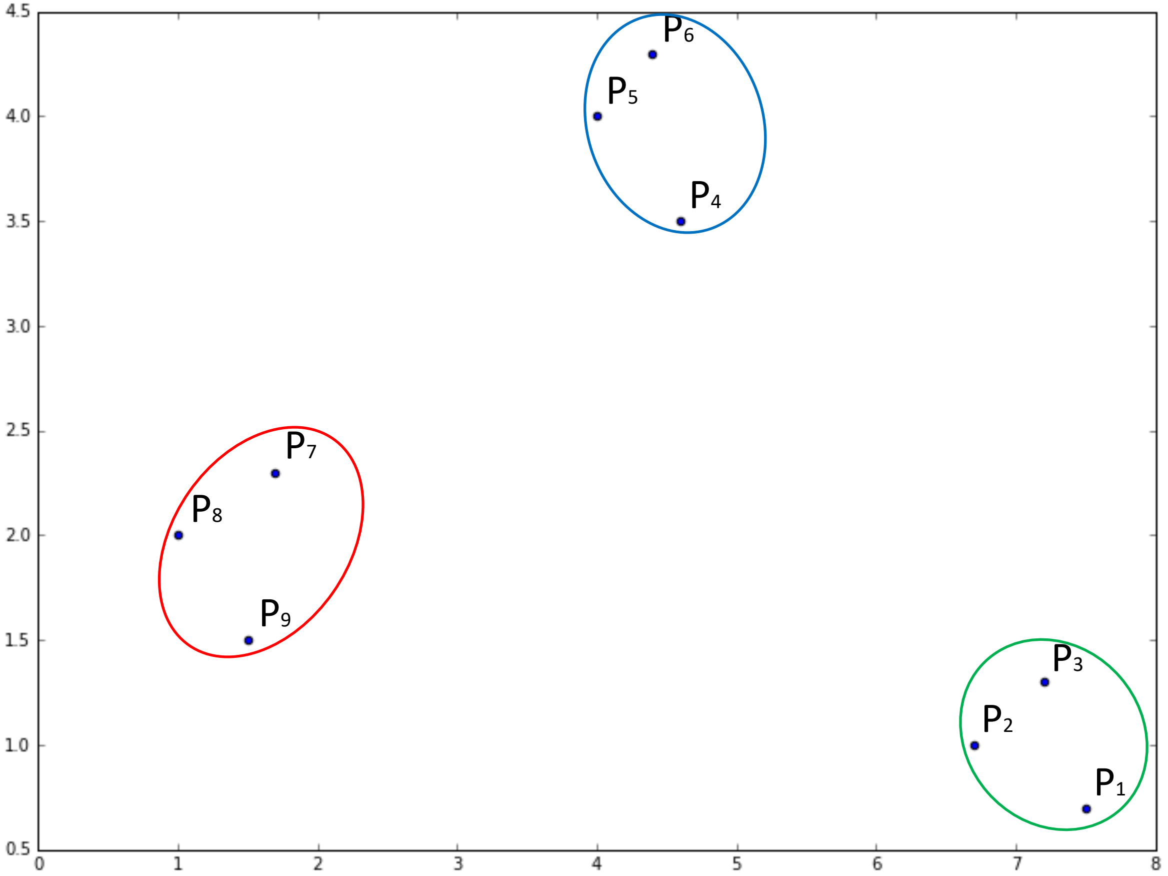
Dendrograms – Optimal # of Clusters

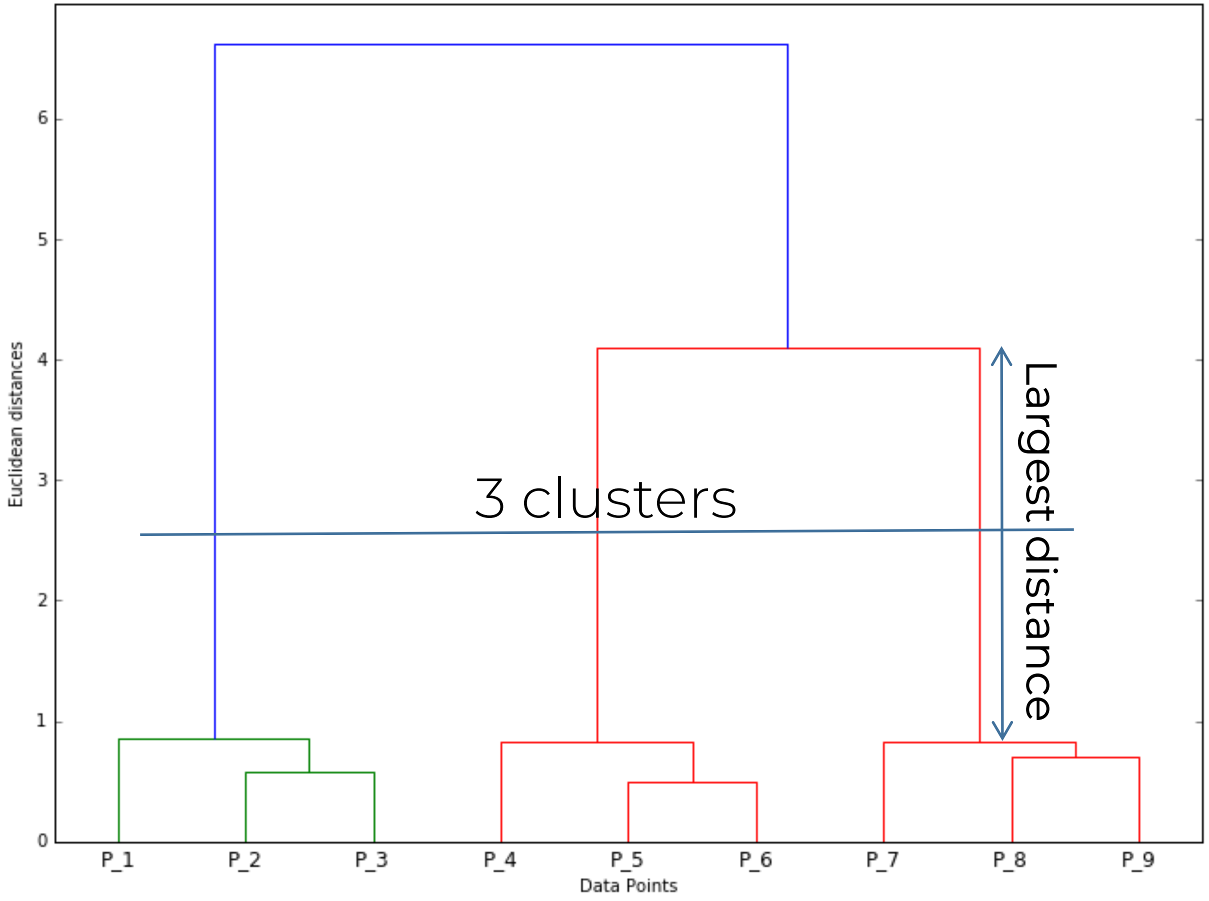




Knowledge Test

Dendrograms – Knowledge Test





Association Rule Learning
Apriori Intuition

ARL - What is it all about ?

ARL - What is it all about ?

People who bought also bought …

ARL- Movie Recommendation

User ID	Movies liked
46578	Movie1, Movie2, Movie3, Movie4
98989	Movie1, Movie2
71527	Movie1, Movie2, Movie4
78981	Movie1, Movie2
89192	Movie2, Movie4
61557	Movie1, Movie3

Potential Rules:

Movie1 $\Rightarrow$ Movie2

Movie2 $\Rightarrow$ Movie4

Movie1 $\Rightarrow$ Movie3

ARL- Market Basket Optimisation

Transaction ID	Products purchased
46578	Burgers, French Fries, Vegetables
98989	Burgers, French Fries, Ketchup
71527	Vegetables, Fruits
78981	Pasta, Fruits, Butter, Vegetables
89192	Burgers, Pasta, French Fries
61557	Fruits, Orange Juice, Vegetables
87923	Burgers, French Fries, Ketchup, Mayo

Potential Rules:

Burgers $\Rightarrow$ French Fries

Vegetables $\Rightarrow$ Fruits

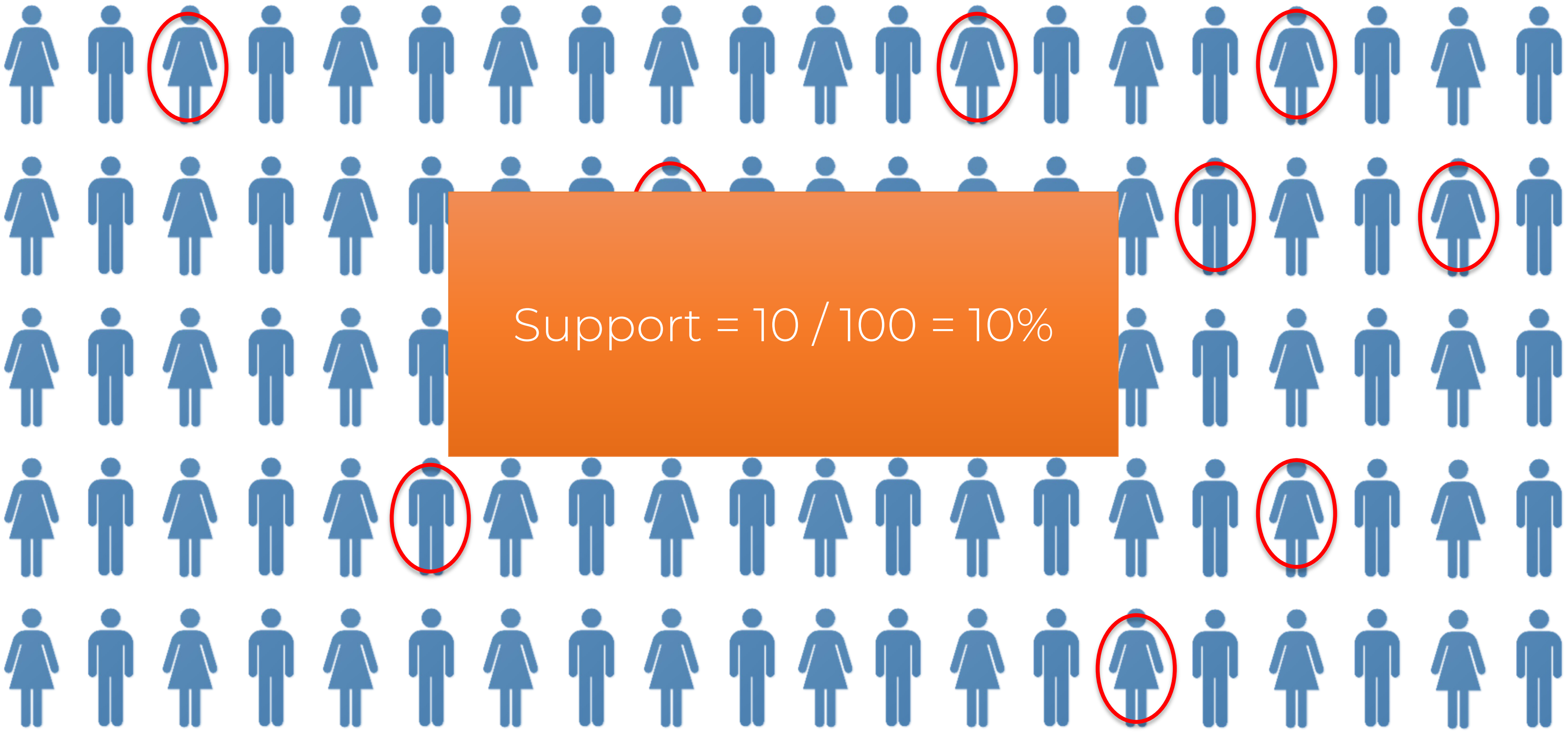
Burgers, French Fries $\Rightarrow$ Ketchup

Apriori - Support

Movie Recommendation: $$\text{support}(\boldsymbol{M}) = \frac{\#\ \text{user watchlists containing}\ \boldsymbol{M}}{\#\ \text{user watchlists}}$$

Market Basket Optimisation: $$\text{support}(\boldsymbol{I}) = \frac{\#\ \text{transactions containing}\ \boldsymbol{I}}{\#\ \text{transactions}}$$

Apriori - Support



Apriori- Confidence

Movie Recommendation: $$\text{confidence}(\boldsymbol{M_1} \rightarrow \boldsymbol{M_2}) = \frac{\#\ \text{user watchlists containing } \boldsymbol{M_1} \text{ and } \boldsymbol{M_2}}{\#\ \text{user watchlists containing } \boldsymbol{M_1}}$$

Market Basket Optimisation: $$\text{confidence}(\boldsymbol{I_1} \rightarrow \boldsymbol{I_2}) = \frac{\#\ \text{transactions containing } \boldsymbol{I_1} \text{ and } \boldsymbol{I_2}}{\#\ \text{transactions containing } \boldsymbol{I_1}}$$

Apriori - Confidence

Apriori - Confidence

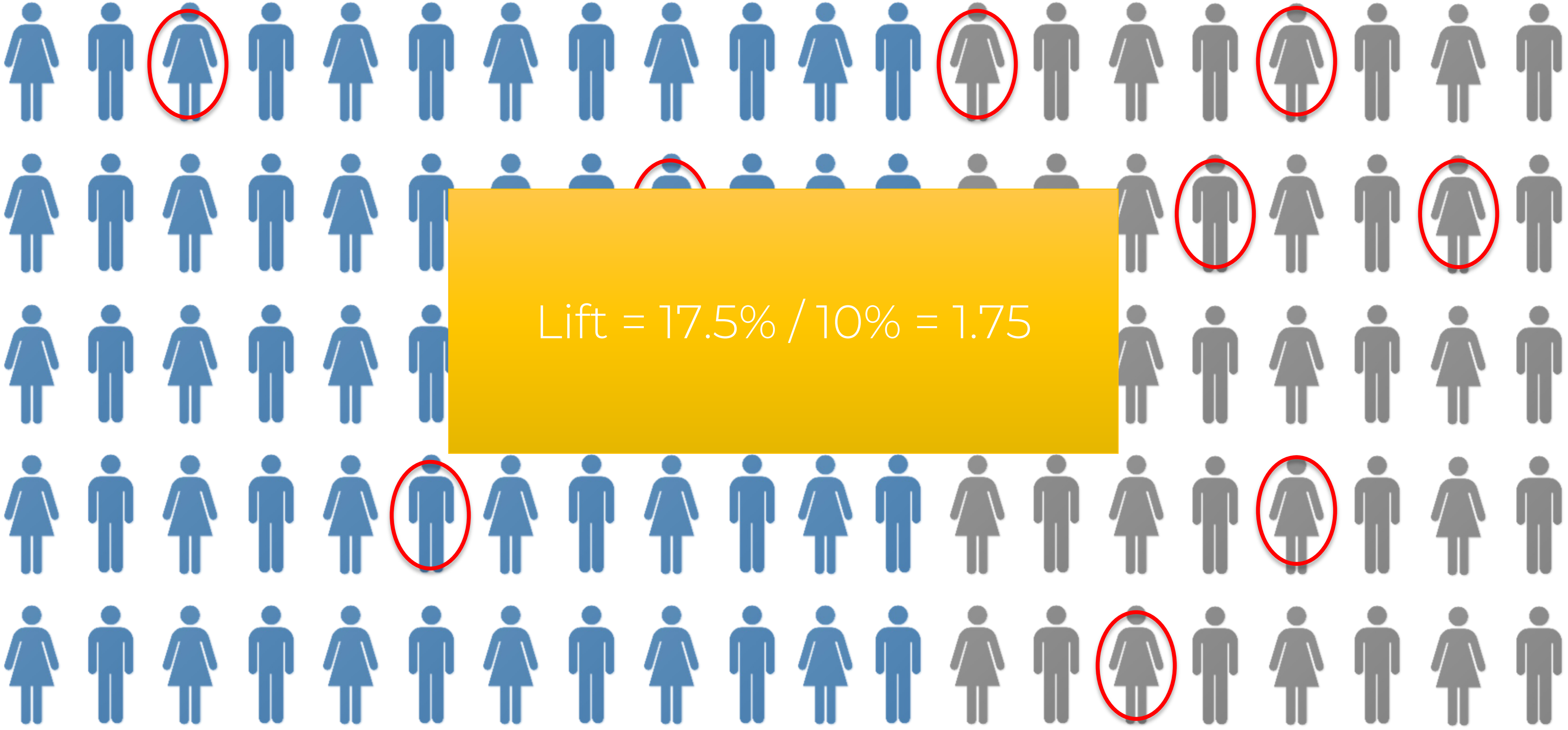
Confidence = 7 / 40 = 17.5%

Apriori - Lift

Movie Recommendation: $$\text{lift}(M_1 \rightarrow M_2) = \frac{\text{confidence}(M_1 \rightarrow M_2)}{\text{support}(M_2)}$$

Market Basket Optimisation: $$\text{lift}(I_1 \rightarrow I_2) = \frac{\text{confidence}(I_1 \rightarrow I_2)}{\text{support}(I_2)}$$

Apriori- Lift



Apriori - Algorithm

Step 1: Set a minimum support and confidence

Step 2: Take all the subsets in transactions having higher support than minimum support

Step 3: Take all the rules of these subsets having higher confidence than minimum confidence

Step 4: Sort the rules by decreasing lift

Association Rule Learning
Eclat Intuition

ARL- What is it all about ?

People who bought also bought …

ARL - Movie Recommendation

User ID	Movies liked
46578	Movie1, Movie2, Movie3, Movie4
98989	Movie1, Movie2
71527	Movie1, Movie2, Movie4
78981	Movie1, Movie2
89192	Movie2, Movie4
61557	Movie1, Movie3

Potential Rules:

Movie1 ⇒ Movie2

Movie2 ⇒ Movie4

Movie1 ⇒ Movie3

ARL - Market Basket Optimisation

Transaction ID	Products purchased
46578	Burgers, French Fries, Vegetables
98989	Burgers, French Fries, Ketchup
71527	Vegetables, Fruits
78981	Pasta, Fruits, Butter, Vegetables
89192	Burgers, Pasta, French Fries
61557	Fruits, Orange Juice, Vegetables
87923	Burgers, French Fries, Ketchup, Mayo

Potential Rules:

Burgers → French Fries

Vegetables → Fruits

Burgers, French Fries → Ketchup

Eclat - Support

Movie Recommendation: $$\text{support}(\boldsymbol{M}) = \frac{\#\ \text{user watchlists containing } \boldsymbol{M}}{\#\ \text{user watchlists}}$$

Market Basket Optimisation: $$\text{support}(\boldsymbol{I}) = \frac{\#\ \text{transactions containing } \boldsymbol{I}}{\#\ \text{transactions}}$$

Eclat- Algorithm

Step 1: Set a minimum support

Step 2: Take all the subsets in transactions having higher support than minimum support

Step 3: Sort these subsets by decreasing support

Reinforcement Learning

The Multi-Armed Bandit Problem

The Multi-Armed Bandit Problem

D1

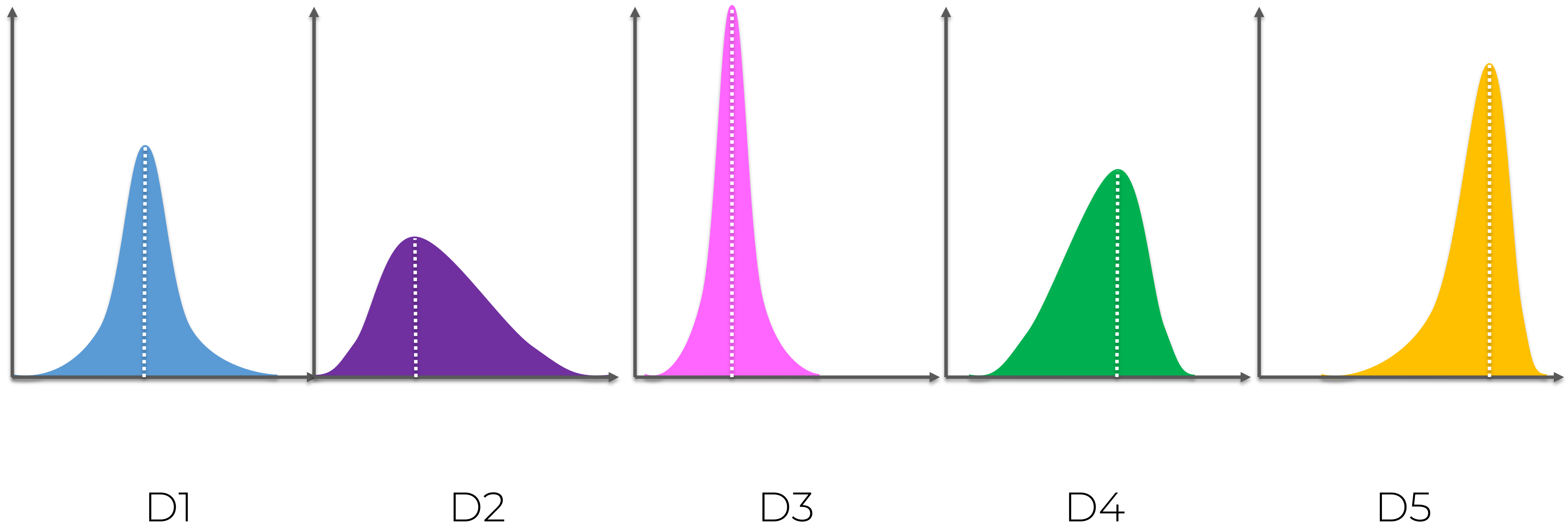
D2

D3

D4

D5

The Multi-Armed Bandit Problem

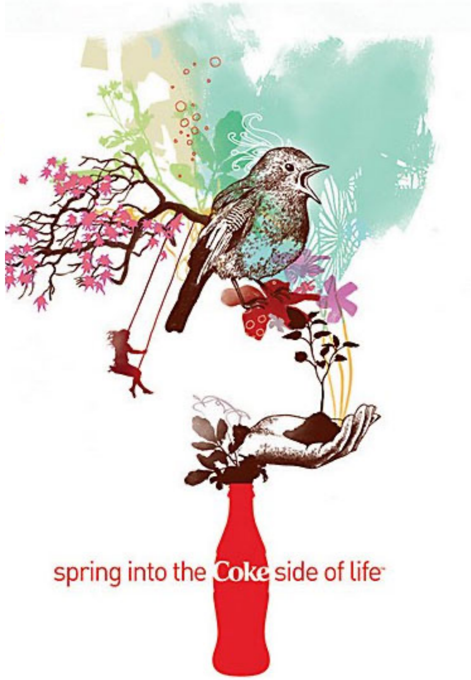


The Multi-Armed Bandit Problem

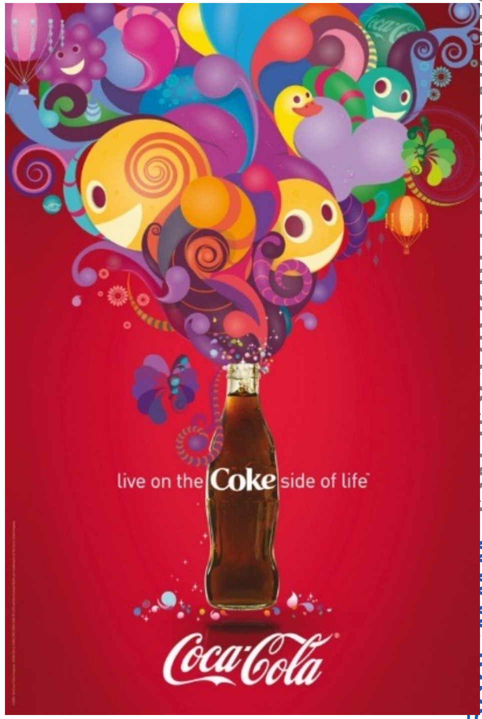
D1

D2

D3



D4



D5

Examples used for educational purposes. No affiliation with Coca-Cola

Upper Confidence Bound Intuition (UCB)

The Multi-Armed Bandit Problem

D1

D2

D3

D4

D5

The Multi-Armed Bandit Problem

- We have d arms. For example, arms are ads that we display to users each time they connect to a web page.
- Each time a user connects to this web page, that makes a round.
- At each round n, we choose one ad to display to the user.
- At each round n, ad i gives reward $r_i(n) \in \{0, 1\}$: $r_i(n) = 1$ if the user clicked on the ad i, 0 if the user didn't.
- Our goal is to maximize the total reward we get over many rounds.

Upper Confidence Bound Algorithm

Step 1. At each round n, we consider two numbers for each ad i:

- $N_i(n)$ - the number of times the ad i was selected up to round n,
- $R_i(n)$ - the sum of rewards of the ad i up to round n.

Step 2. From these two numbers we compute:

- the average reward of ad i up to round n

$$\bar{r}_i(n) = \frac{R_i(n)}{N_i(n)}$$

- the confidence interval $[\bar{r}_i(n) - \Delta_i(n), \bar{r}_i(n) + \Delta_i(n)]$ at round n with

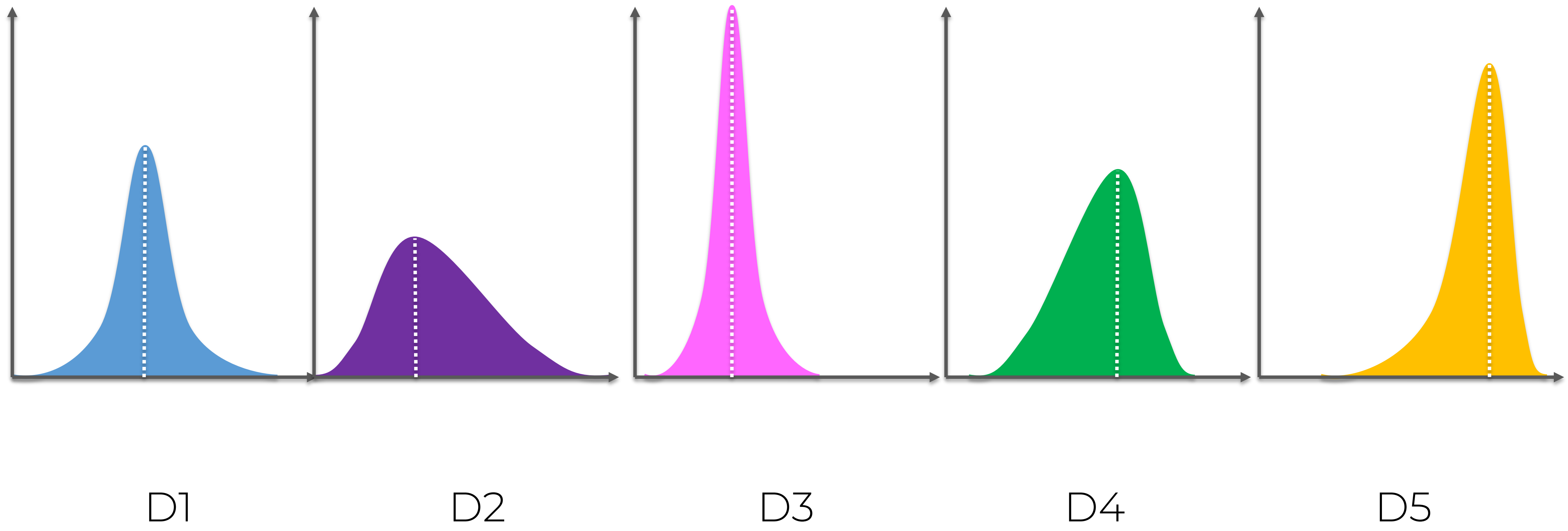
$$\Delta_i(n) = \sqrt{\frac{3}{2} \frac{\log(n)}{N_i(n)}}$$

Step 3. We select the ad i that has the maximum UCB $\bar{r}_i(n) + \Delta_i(n)$.

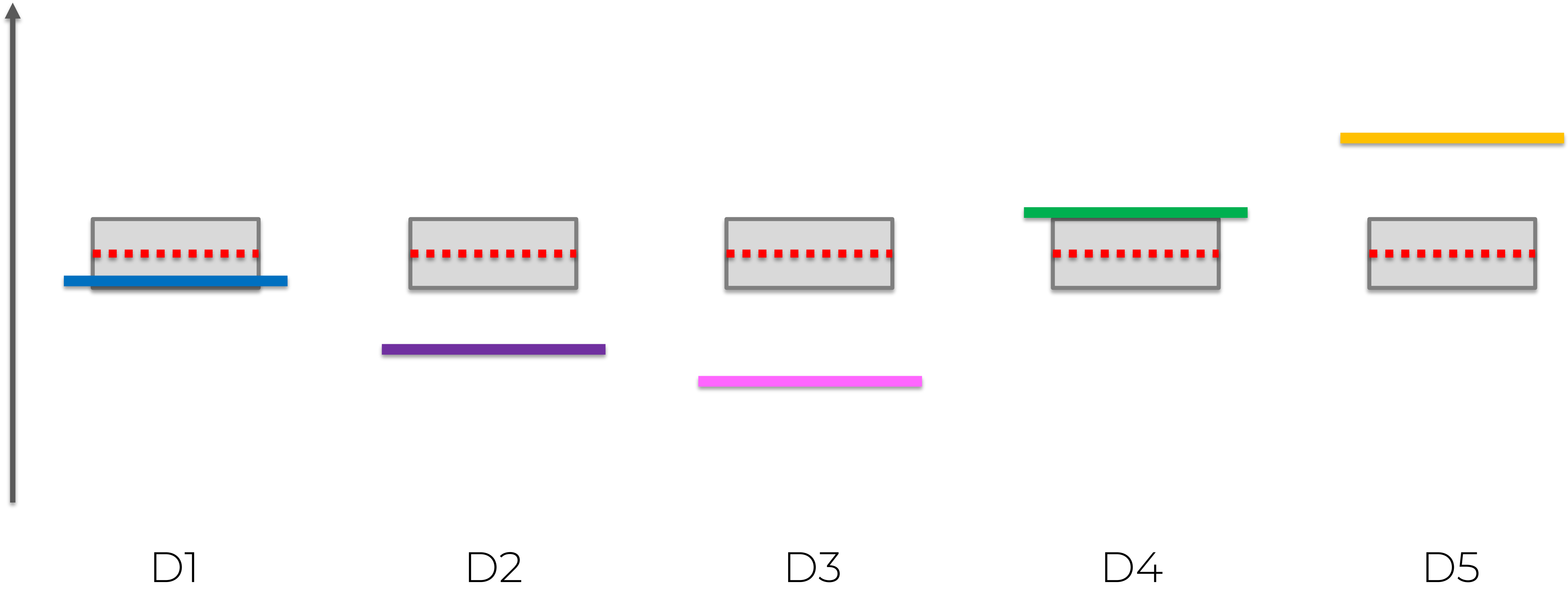
Upper Confidence Bound Algorithm

D1 D2 D3 D4 D5

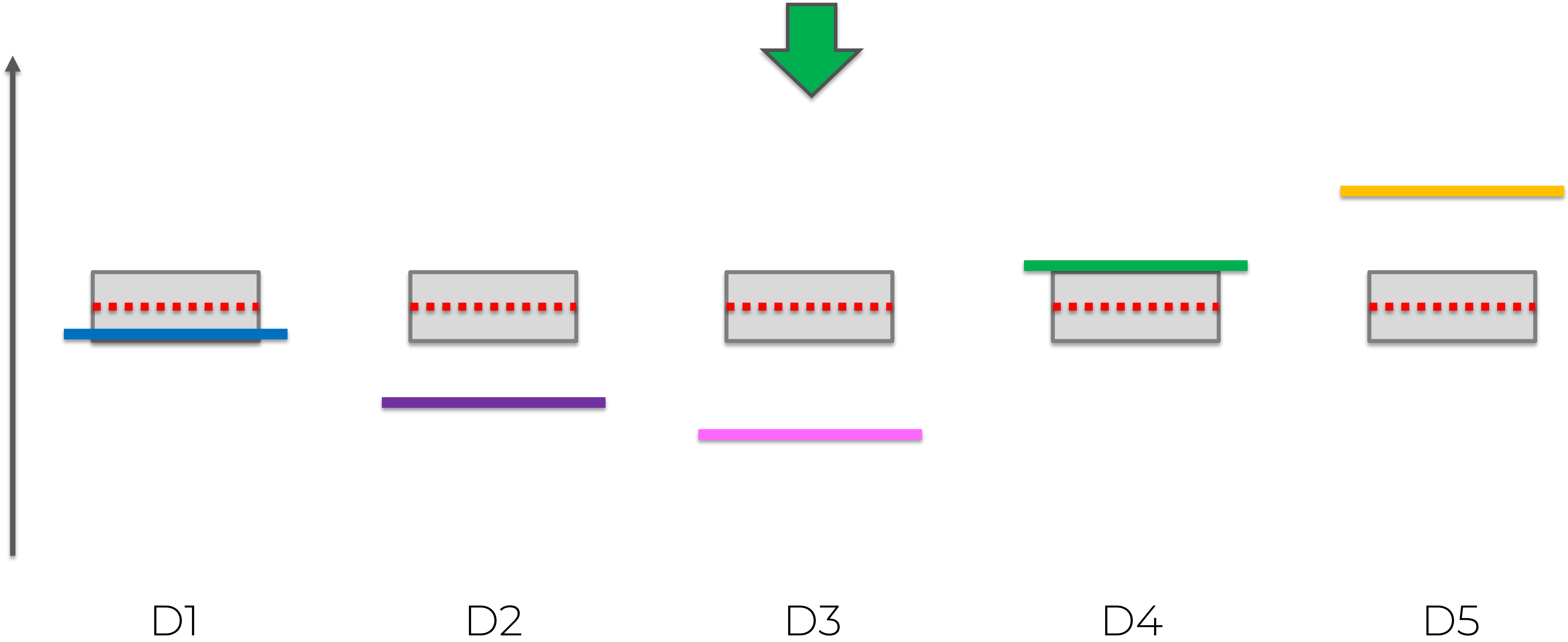
Upper Confidence Bound Algorithm



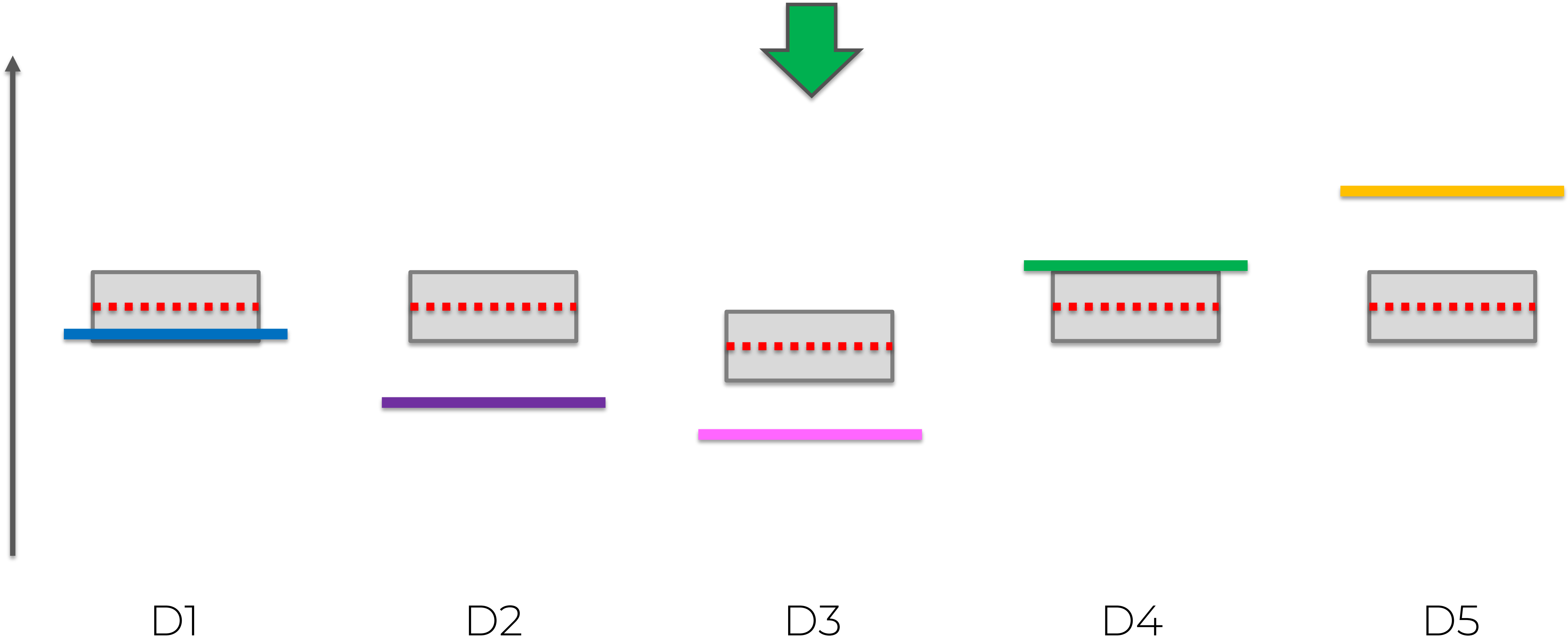
Upper Confidence Bound Algorithm



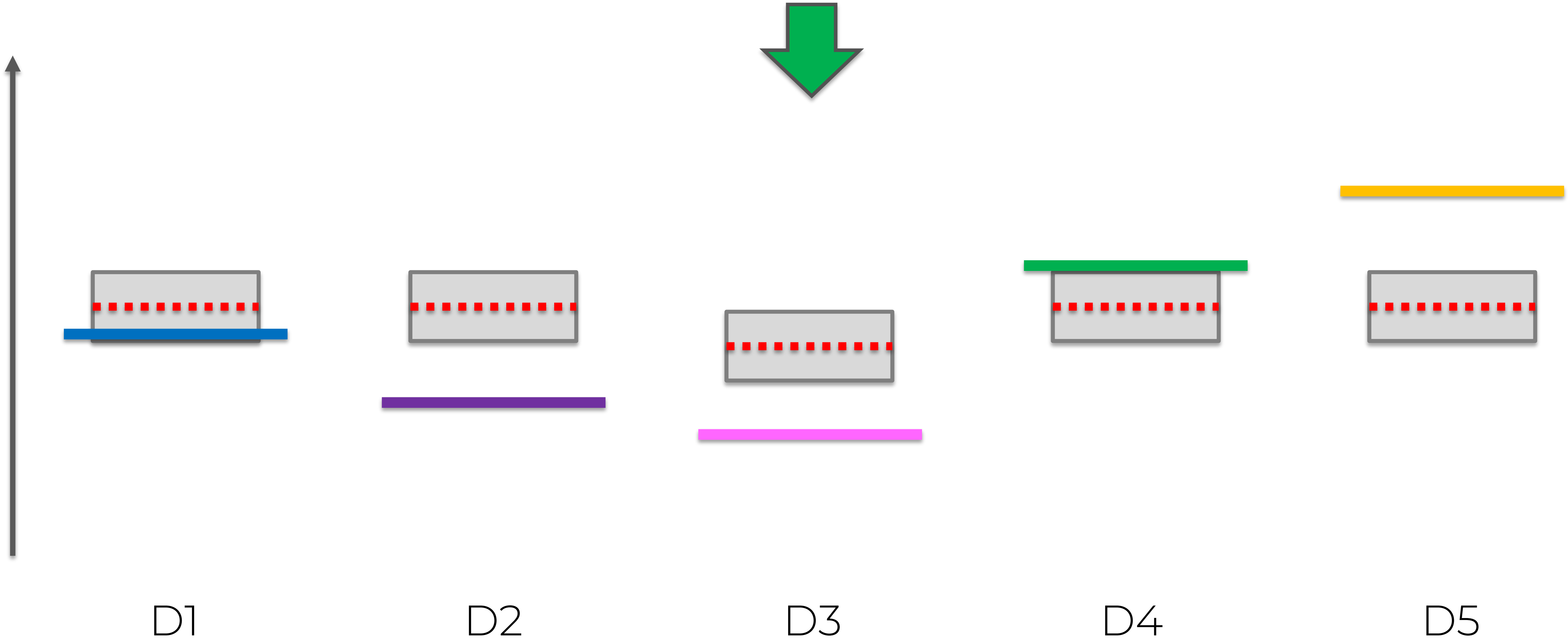
Upper Confidence Bound Algorithm



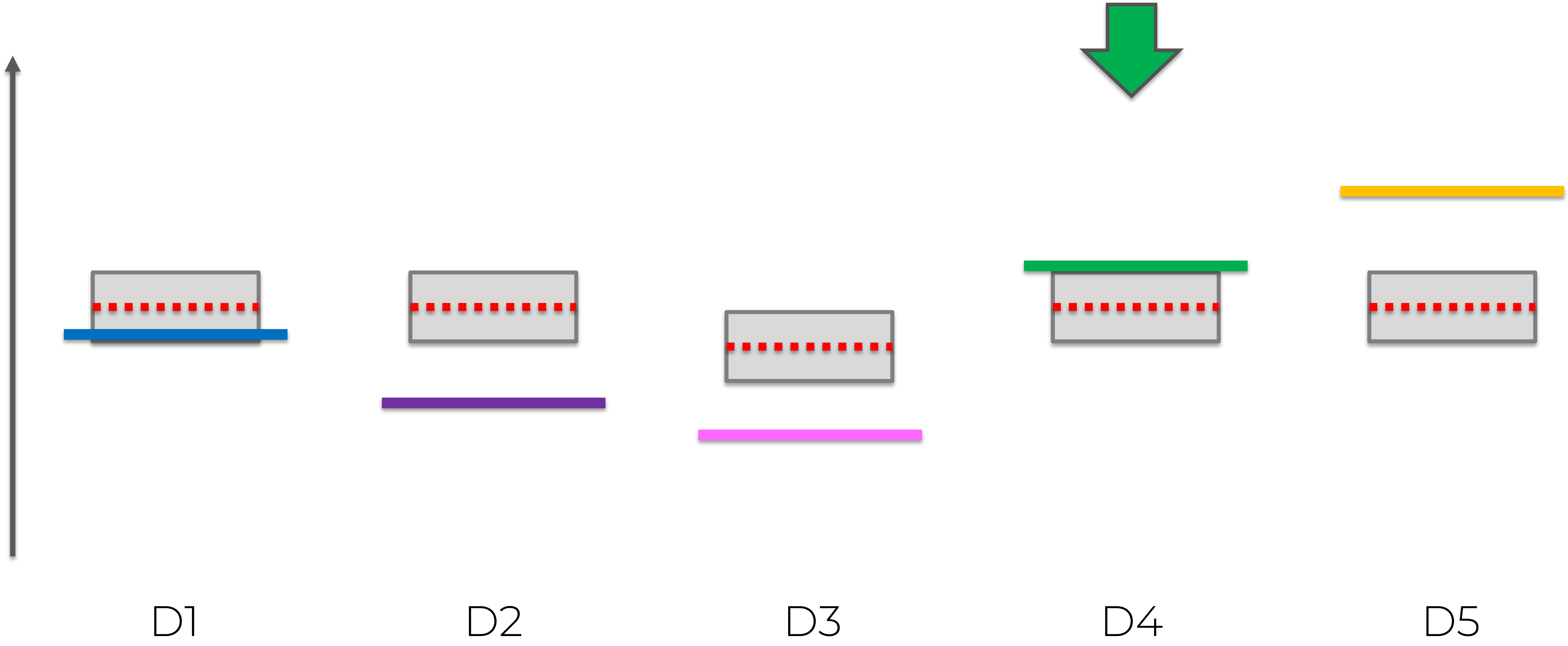
Upper Confidence Bound Algorithm



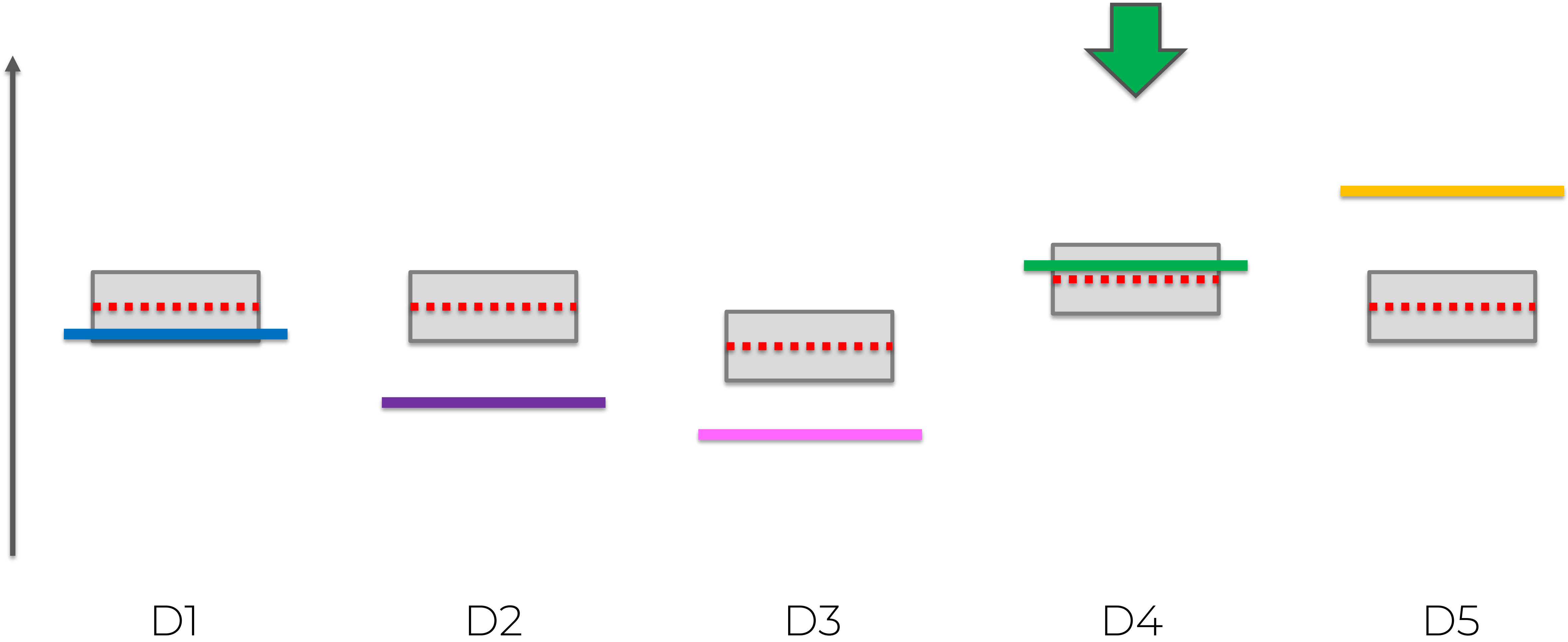
Upper Confidence Bound Algorithm



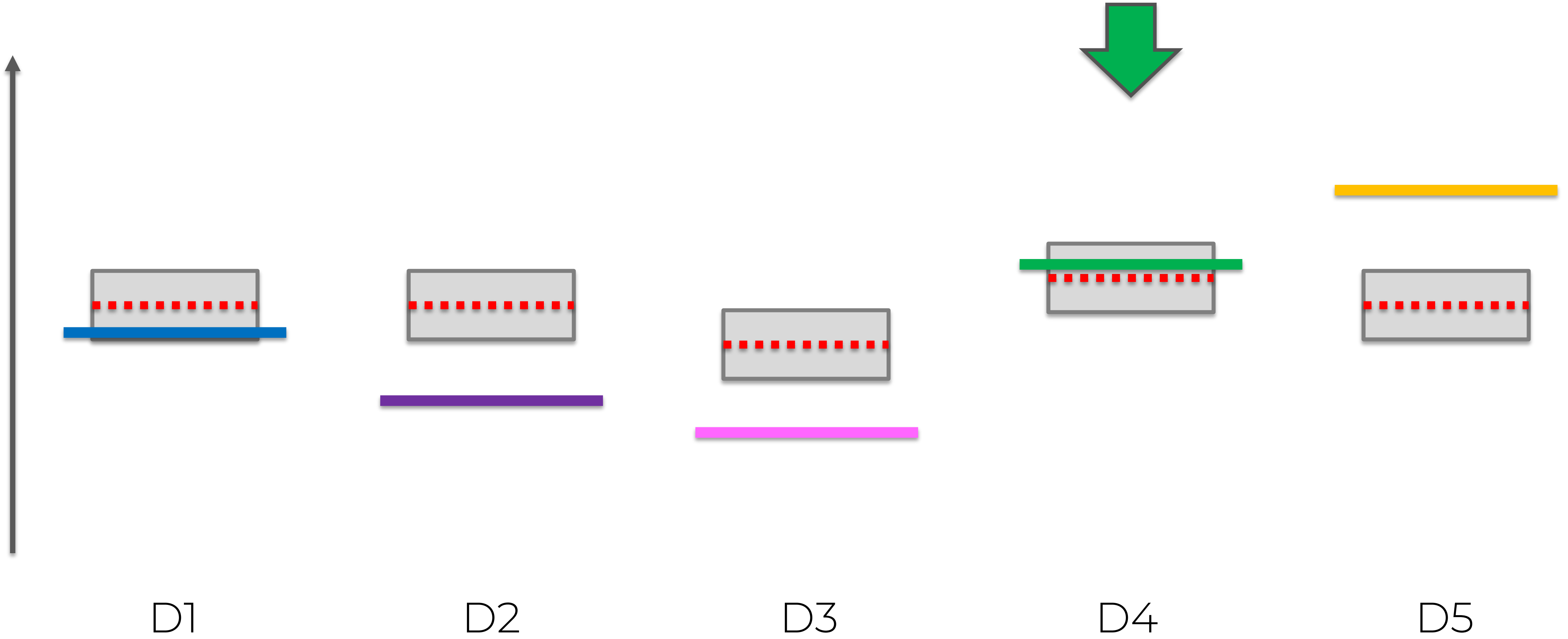
Upper Confidence Bound Algorithm



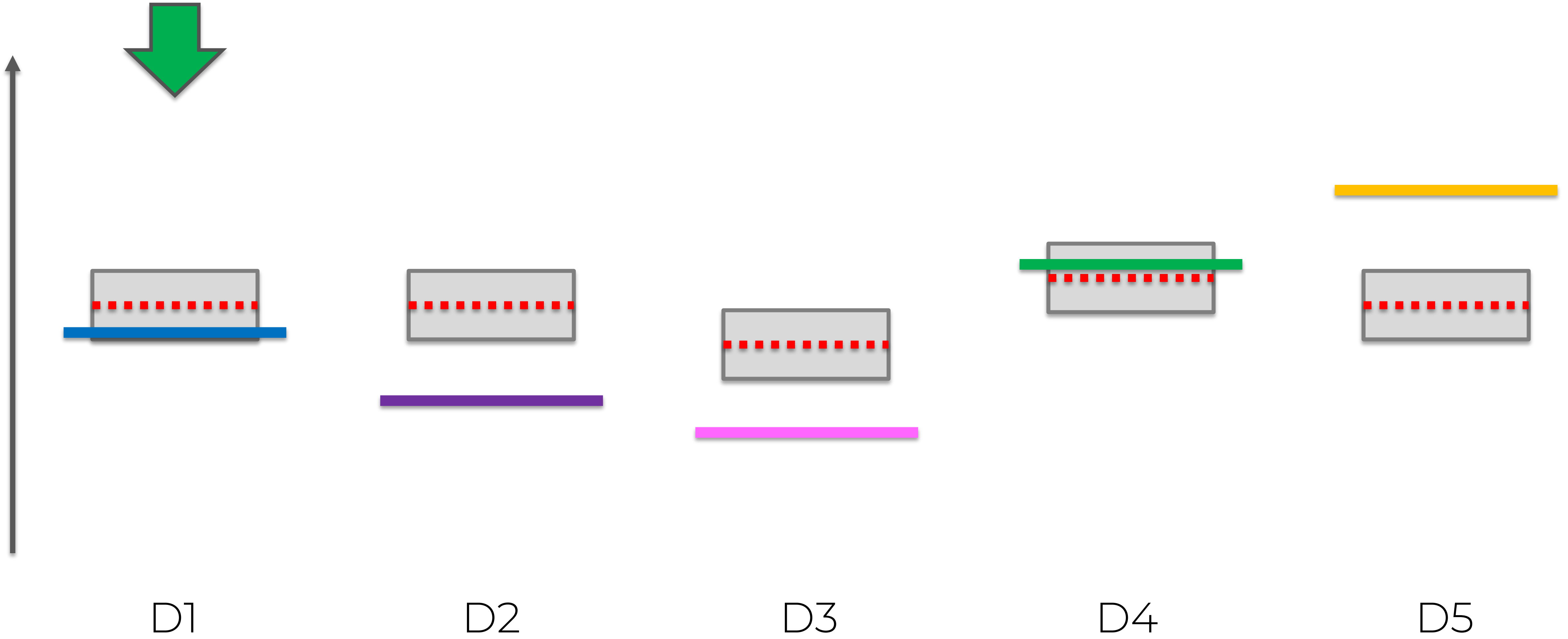
Upper Confidence Bound Algorithm



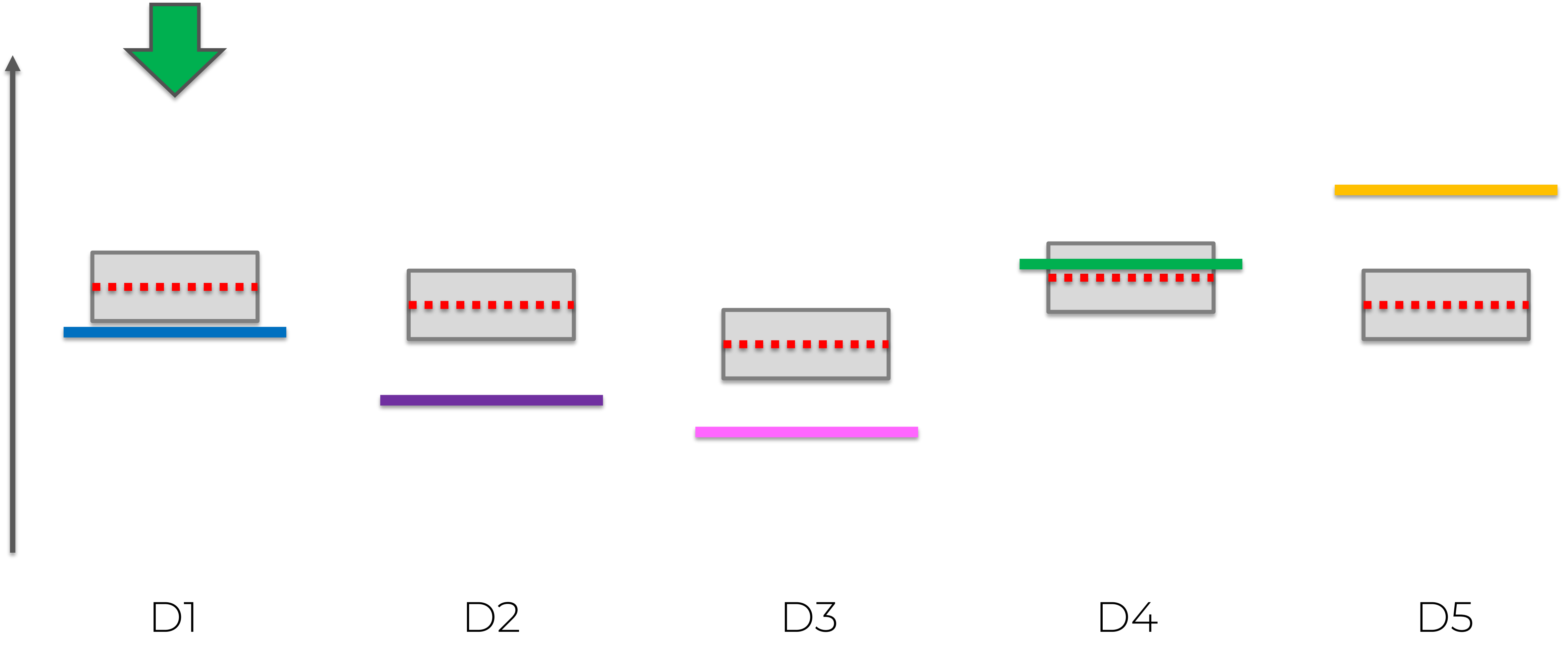
Upper Confidence Bound Algorithm



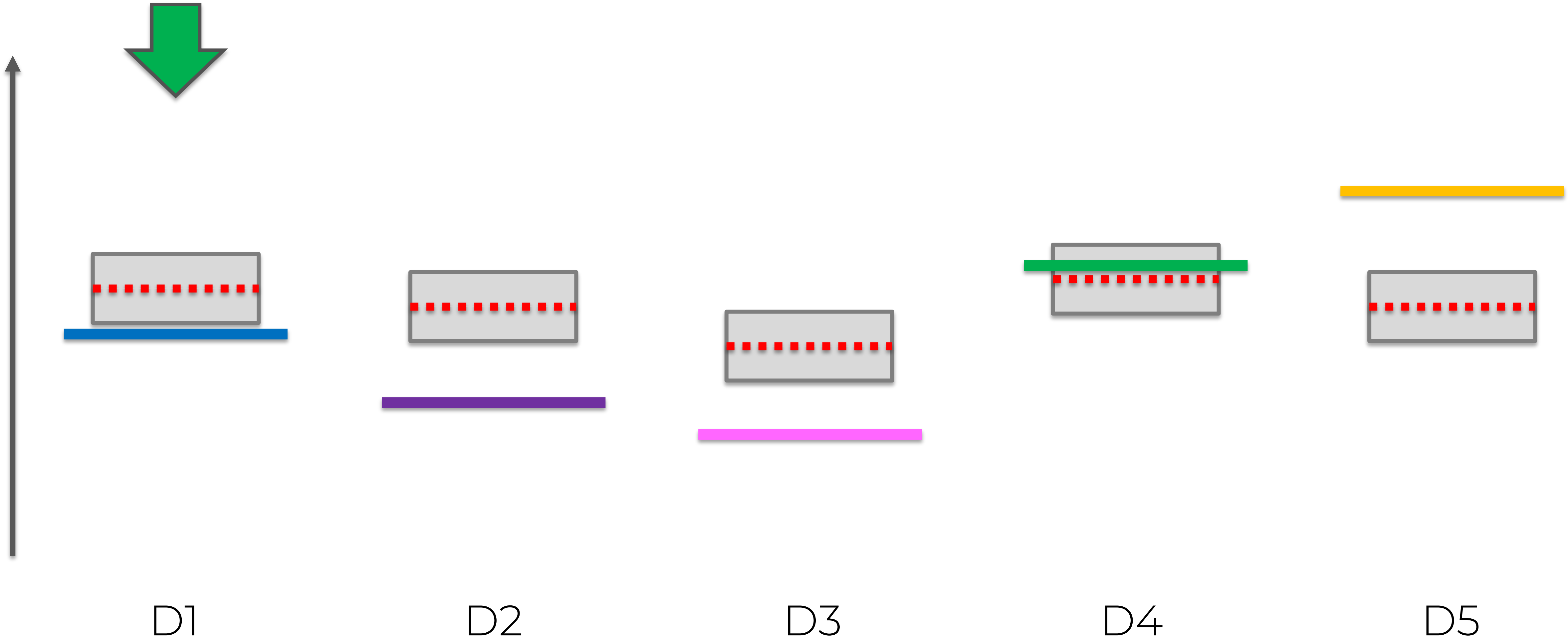
Upper Confidence Bound Algorithm



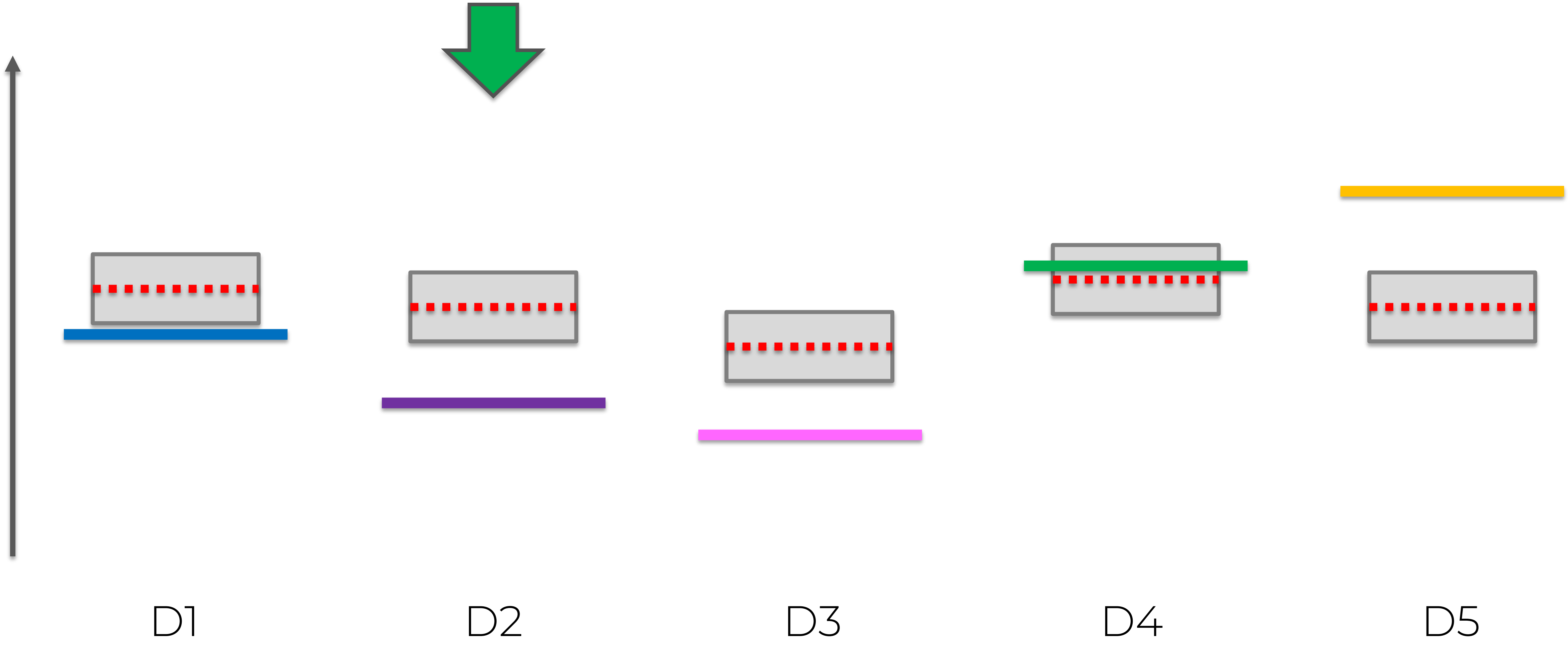
Upper Confidence Bound Algorithm



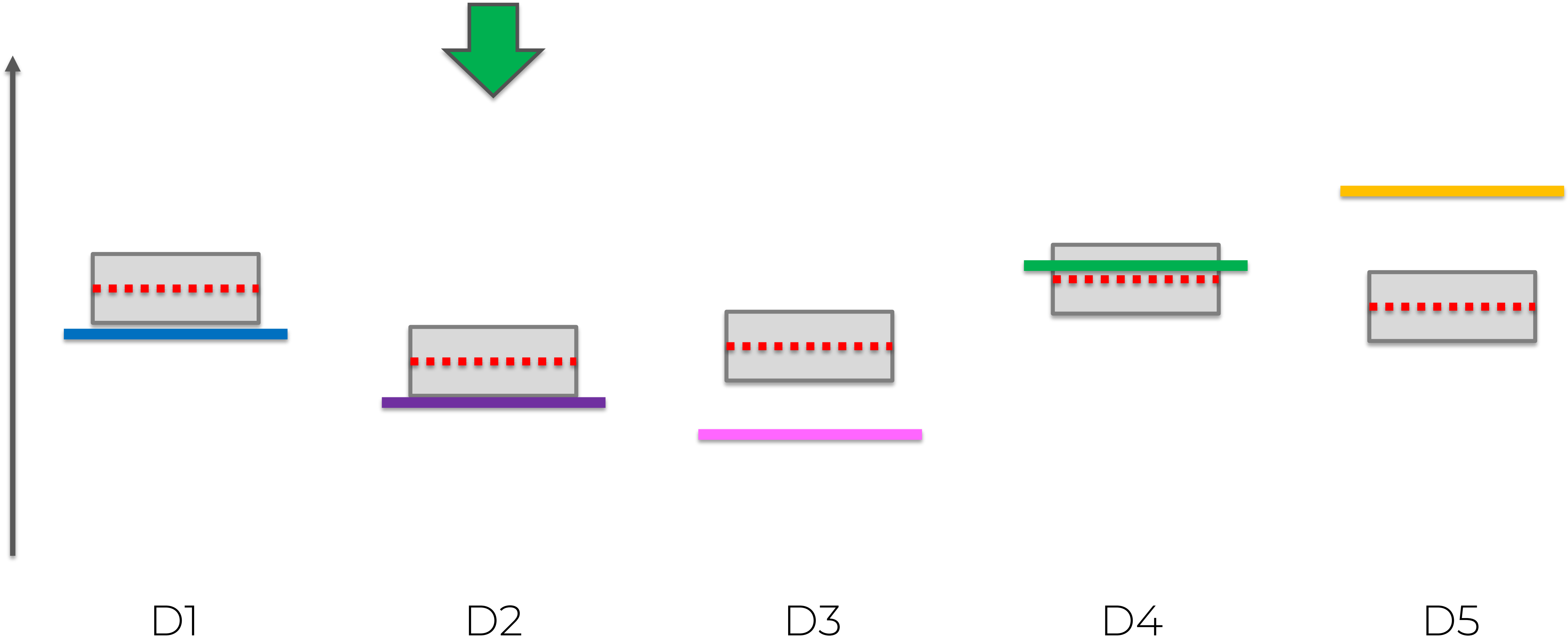
Upper Confidence Bound Algorithm



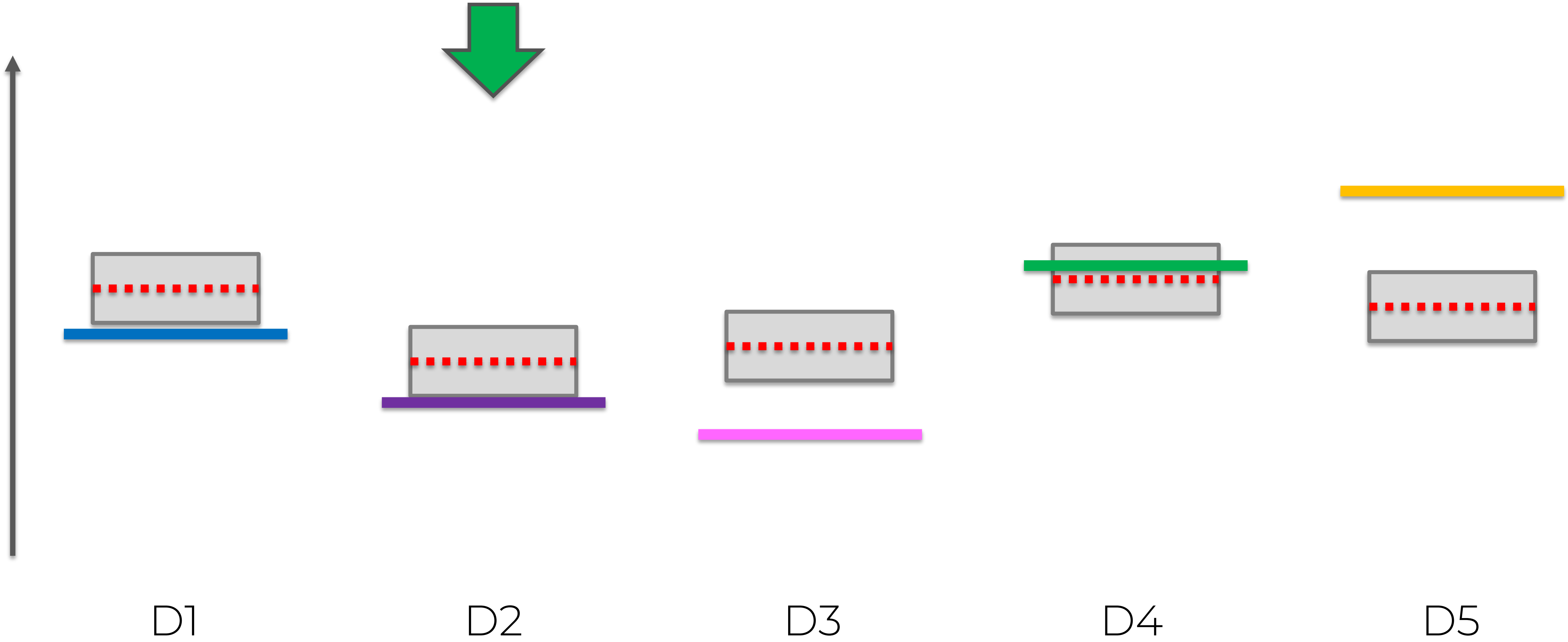
Upper Confidence Bound Algorithm



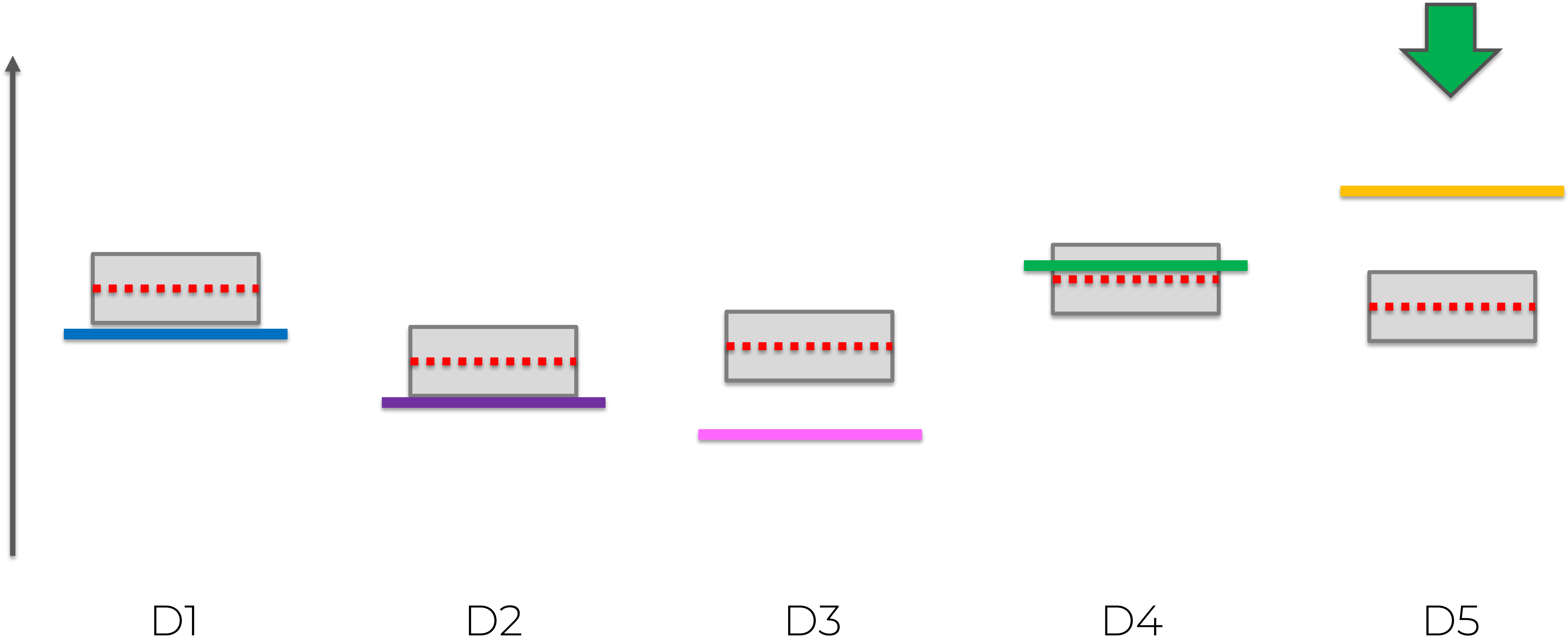
Upper Confidence Bound Algorithm



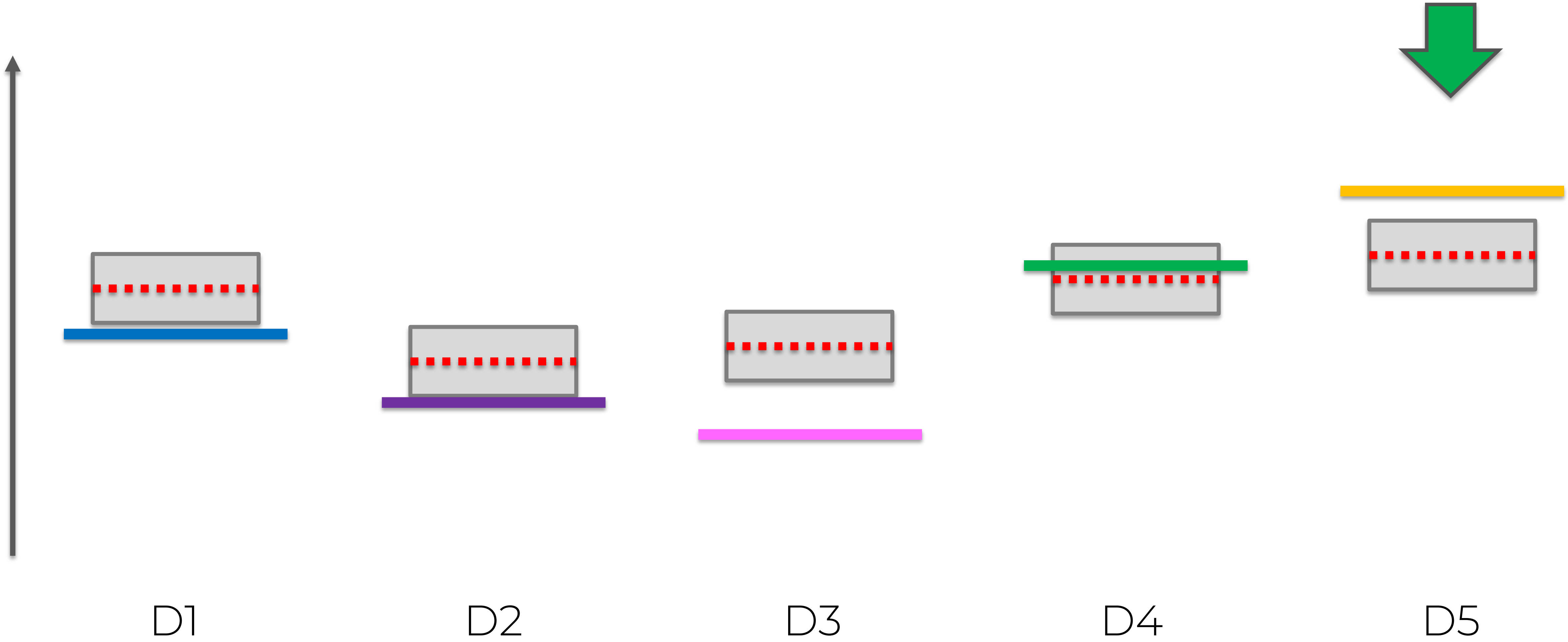
Upper Confidence Bound Algorithm



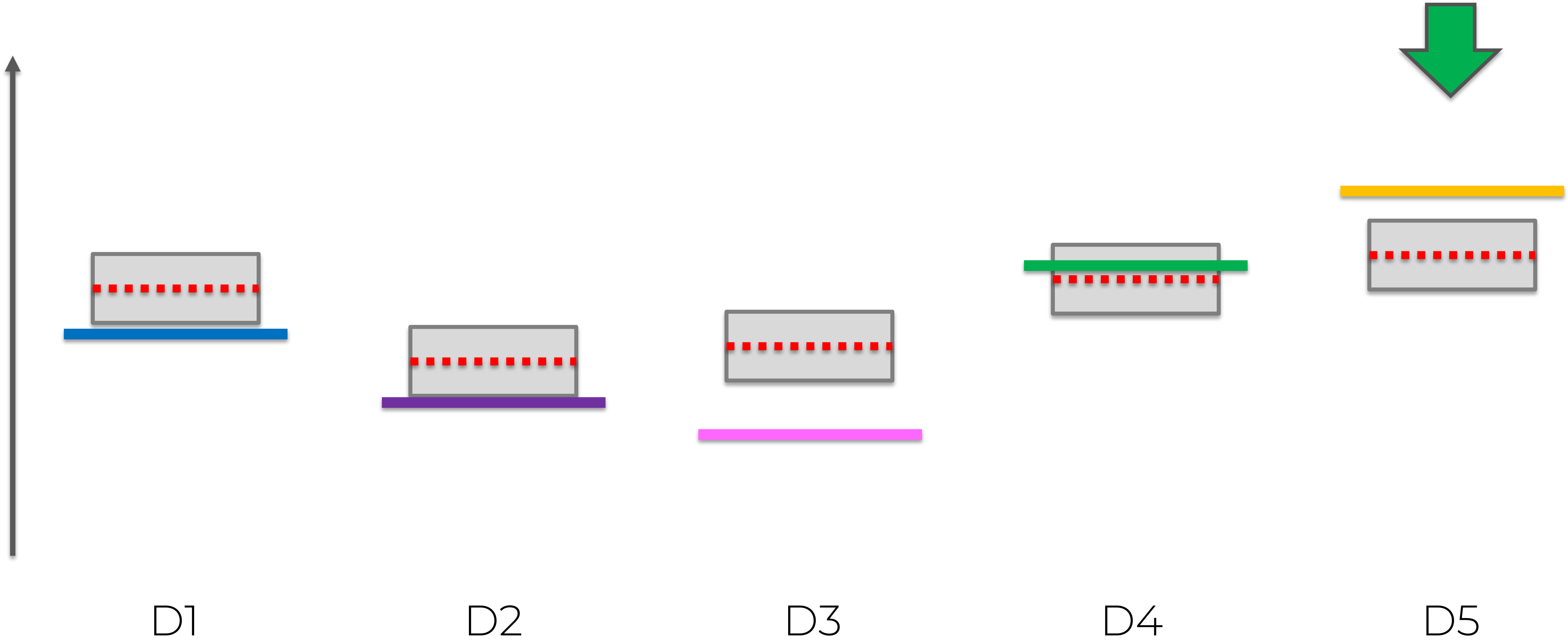
Upper Confidence Bound Algorithm



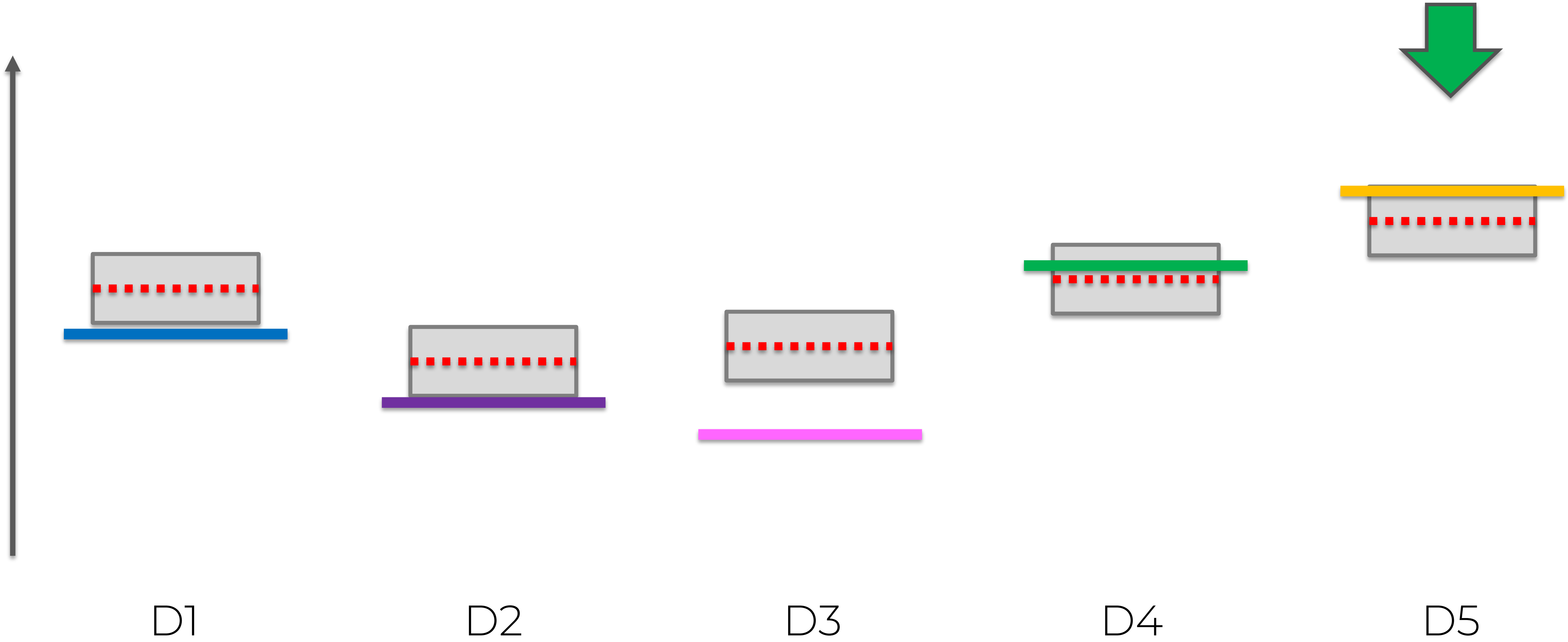
Upper Confidence Bound Algorithm



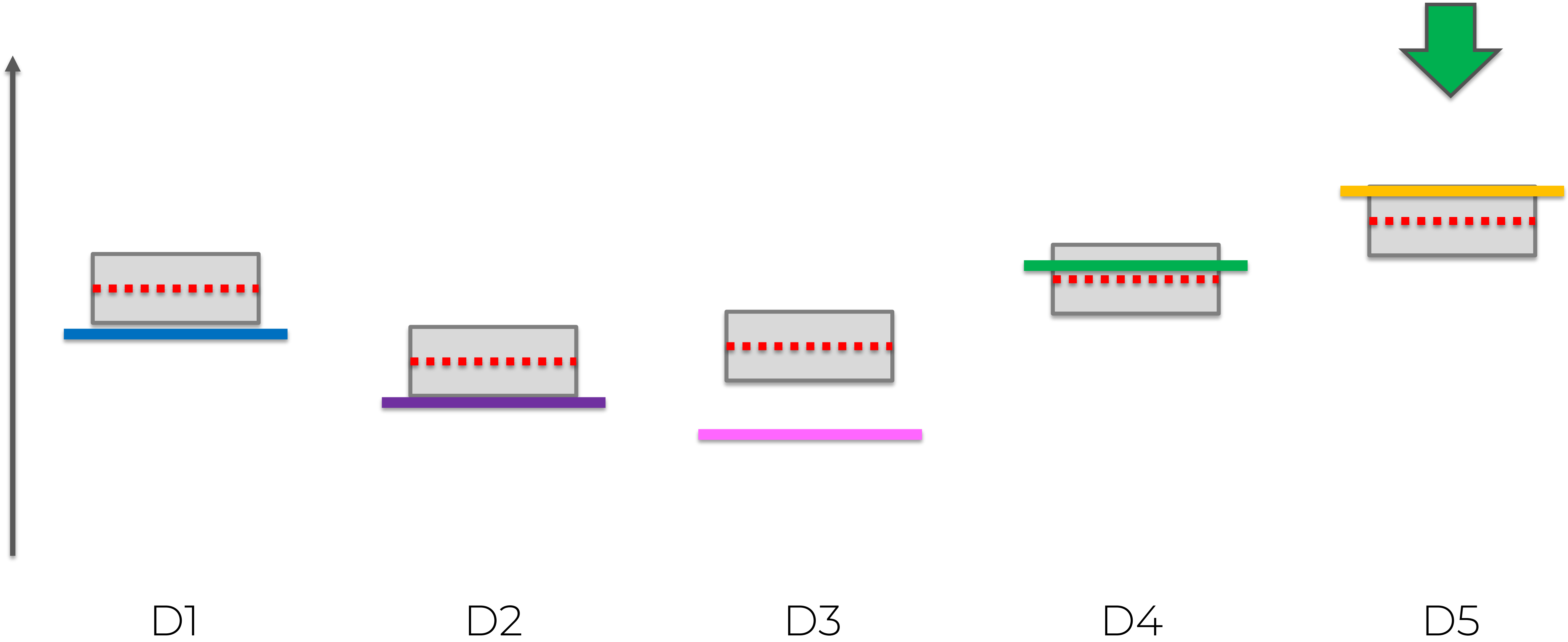
Upper Confidence Bound Algorithm



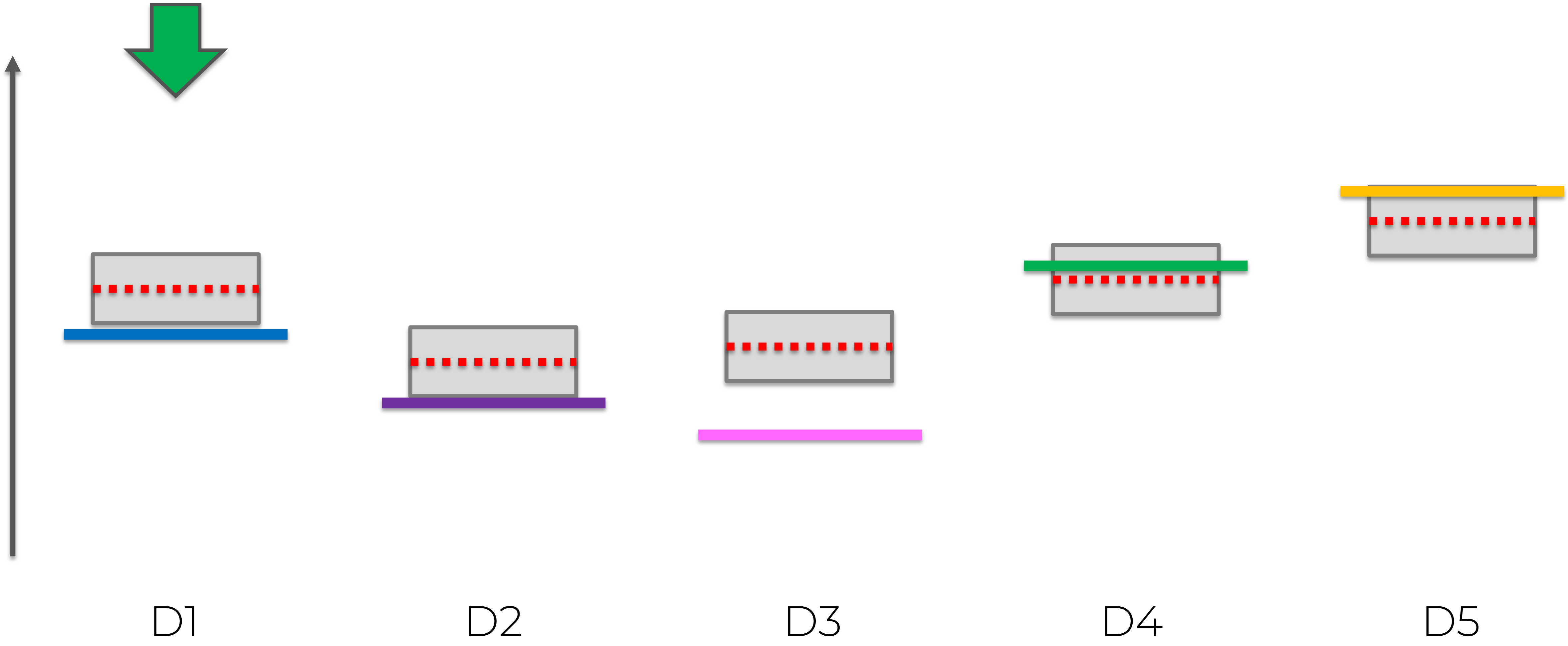
Upper Confidence Bound Algorithm



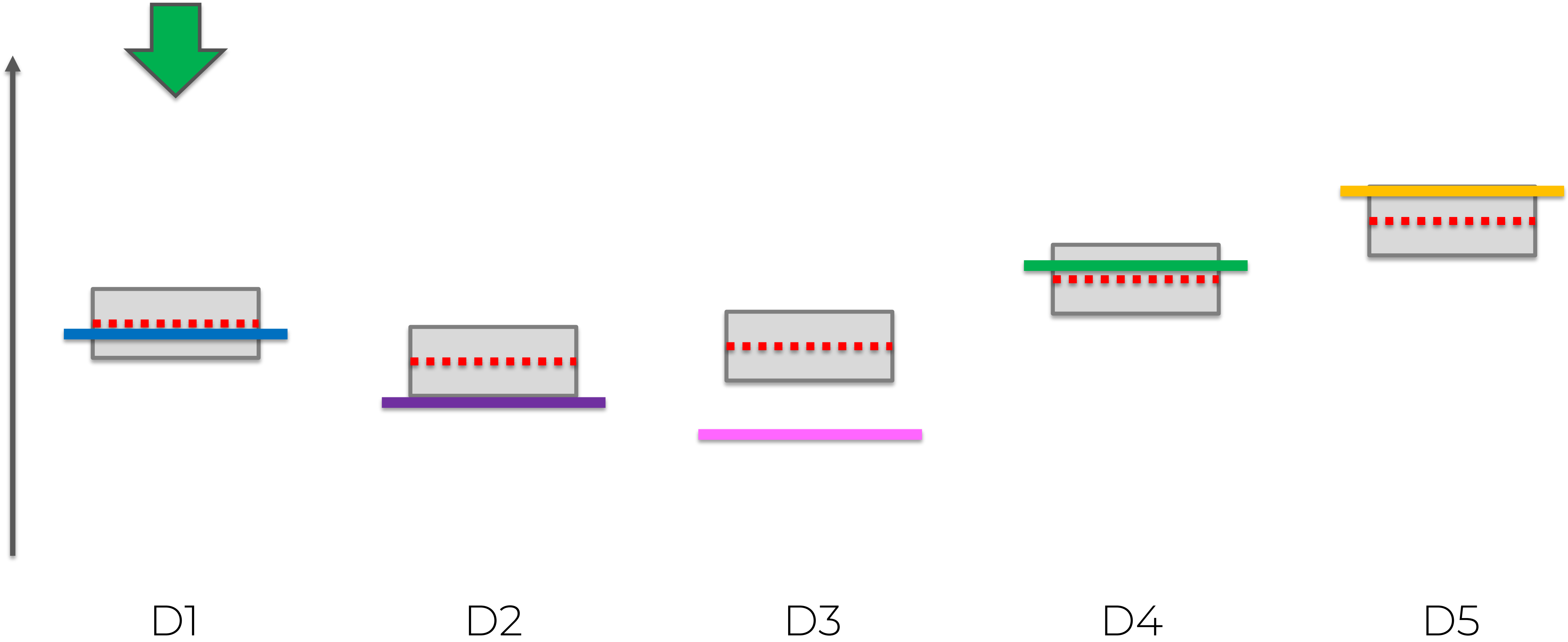
Upper Confidence Bound Algorithm



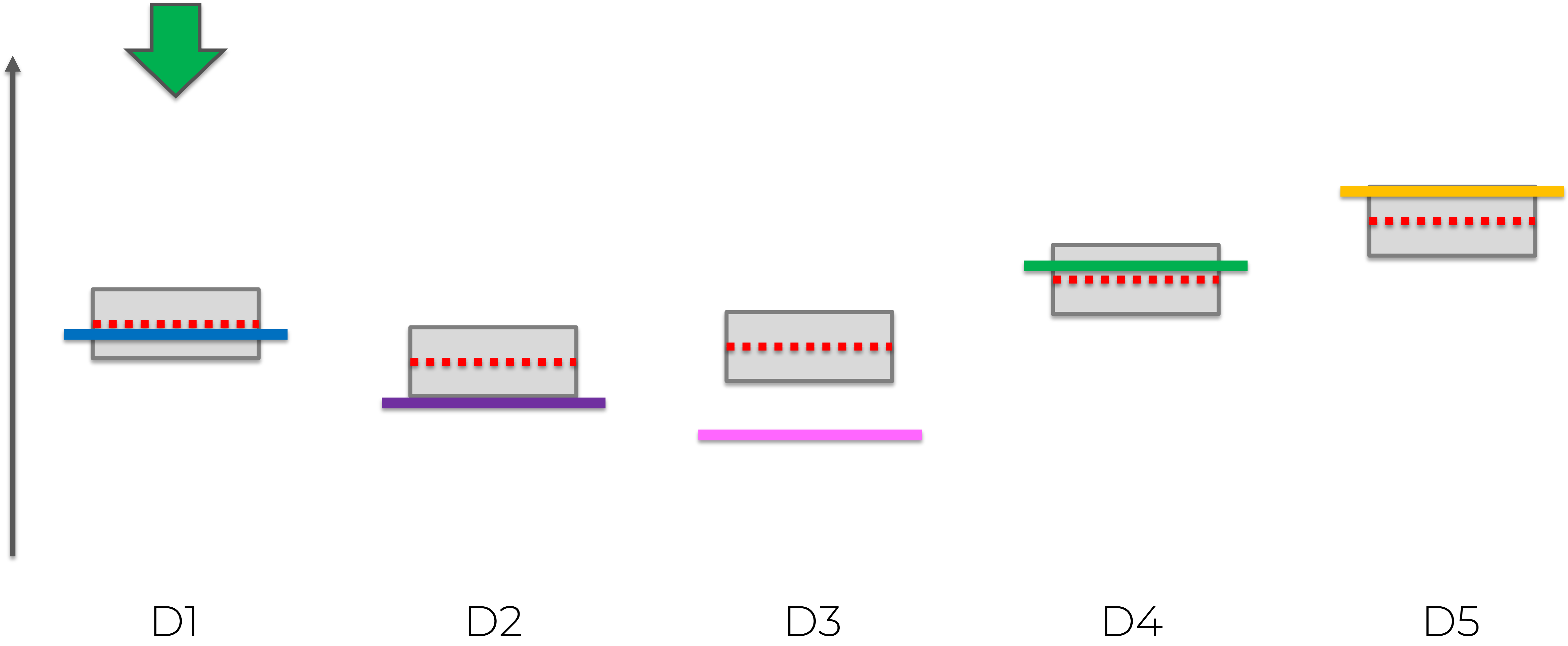
Upper Confidence Bound Algorithm



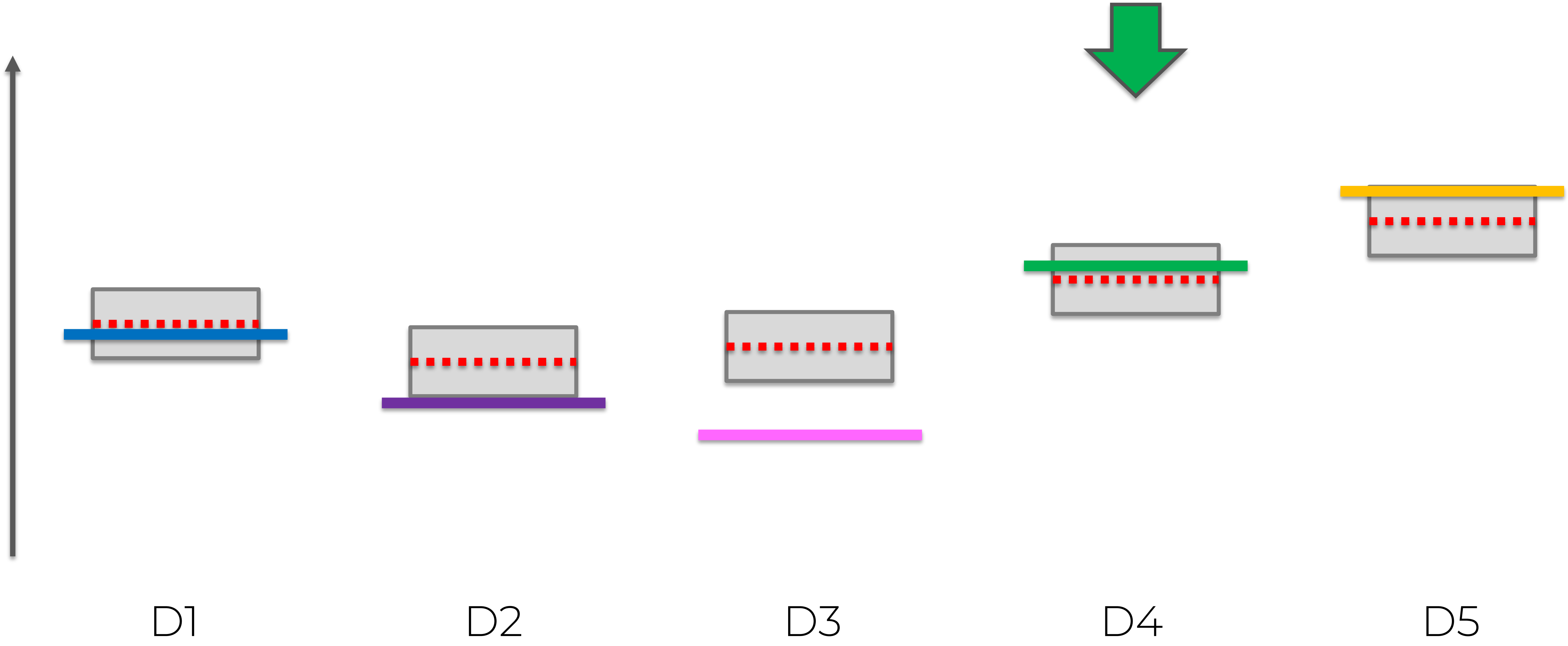
Upper Confidence Bound Algorithm



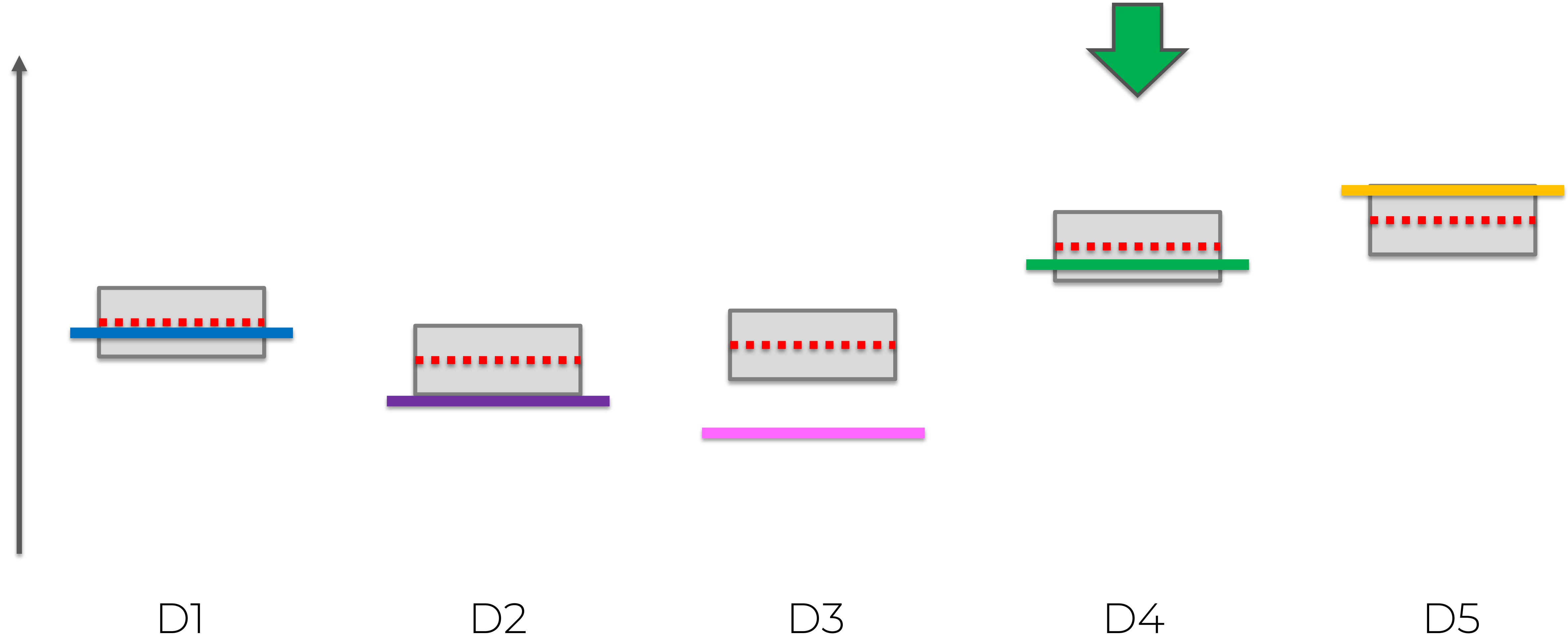
Upper Confidence Bound Algorithm



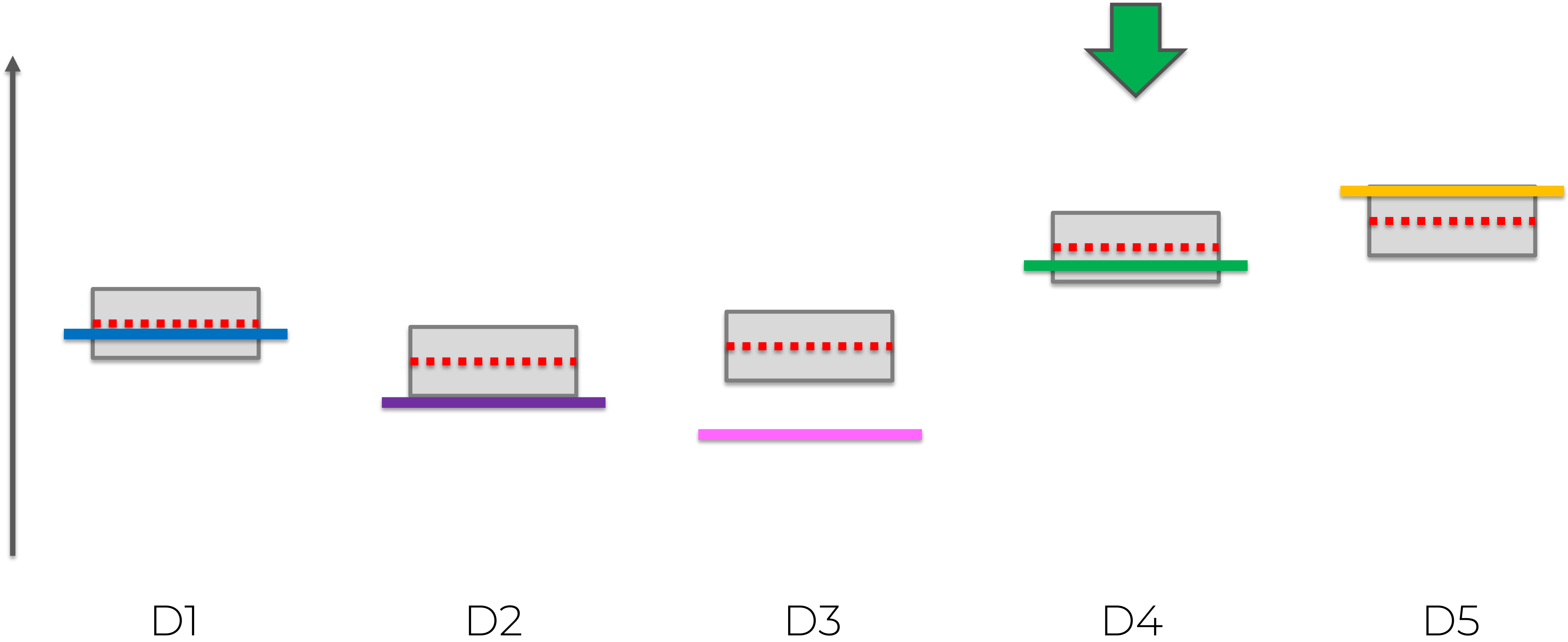
Upper Confidence Bound Algorithm



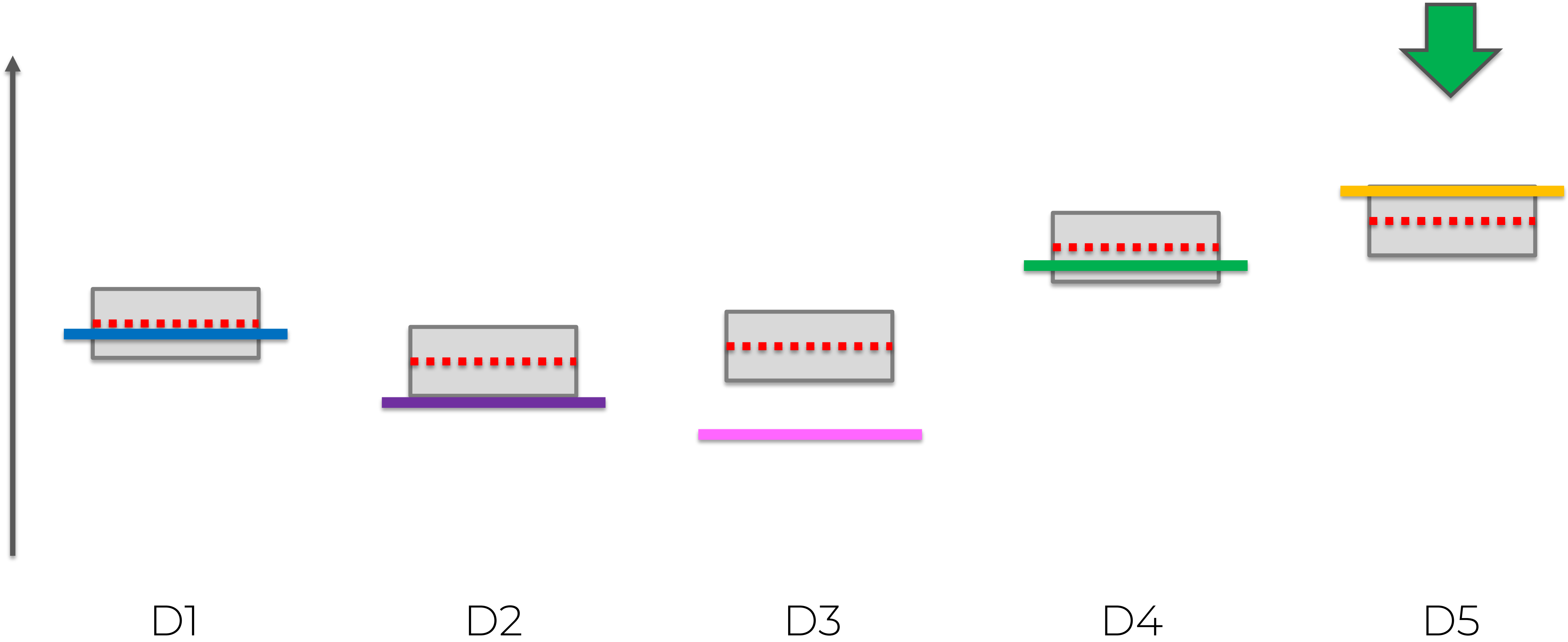
Upper Confidence Bound Algorithm



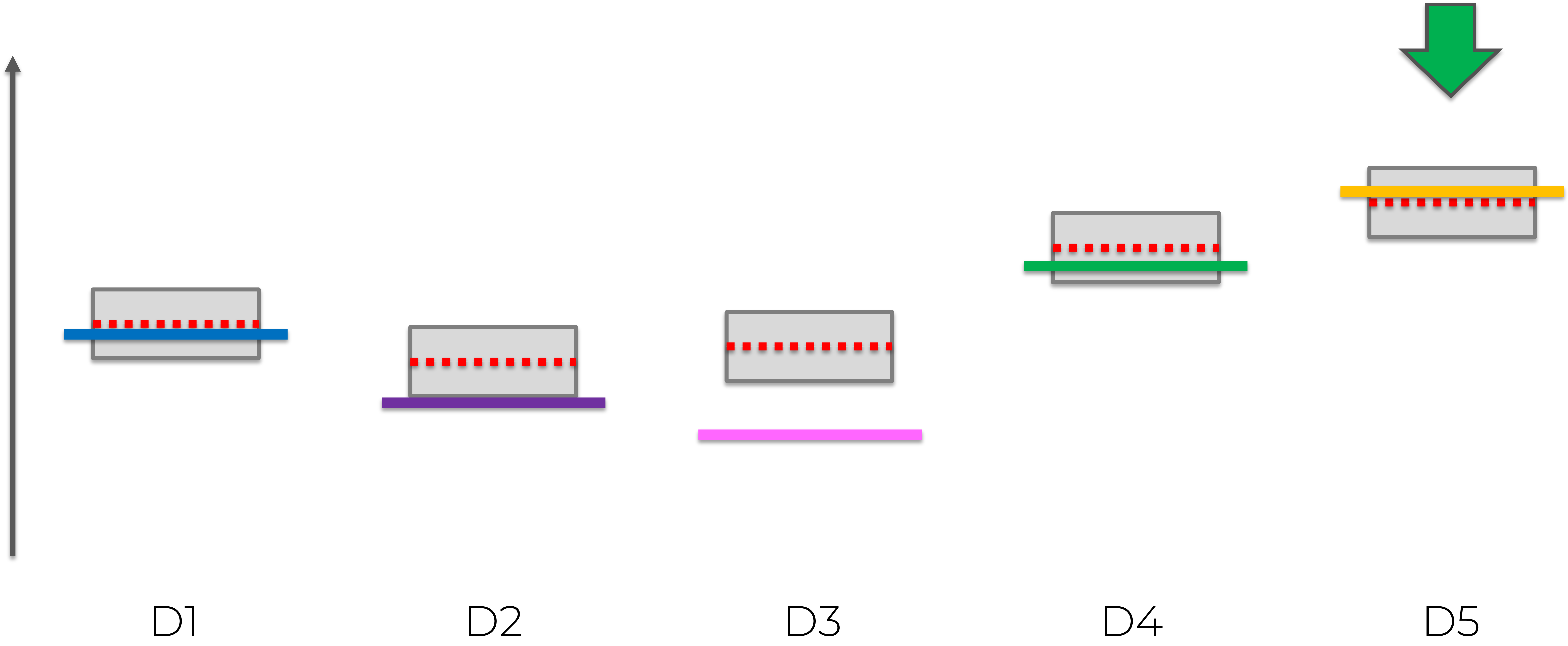
Upper Confidence Bound Algorithm



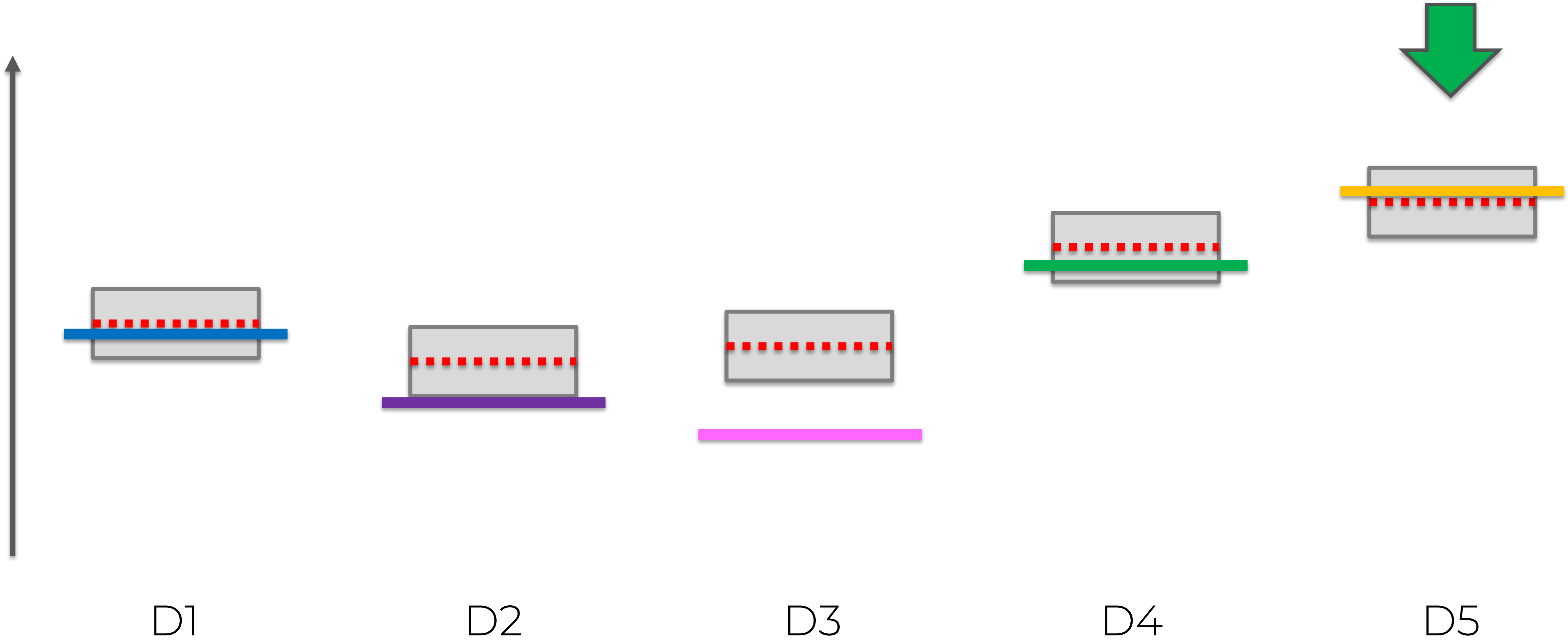
Upper Confidence Bound Algorithm



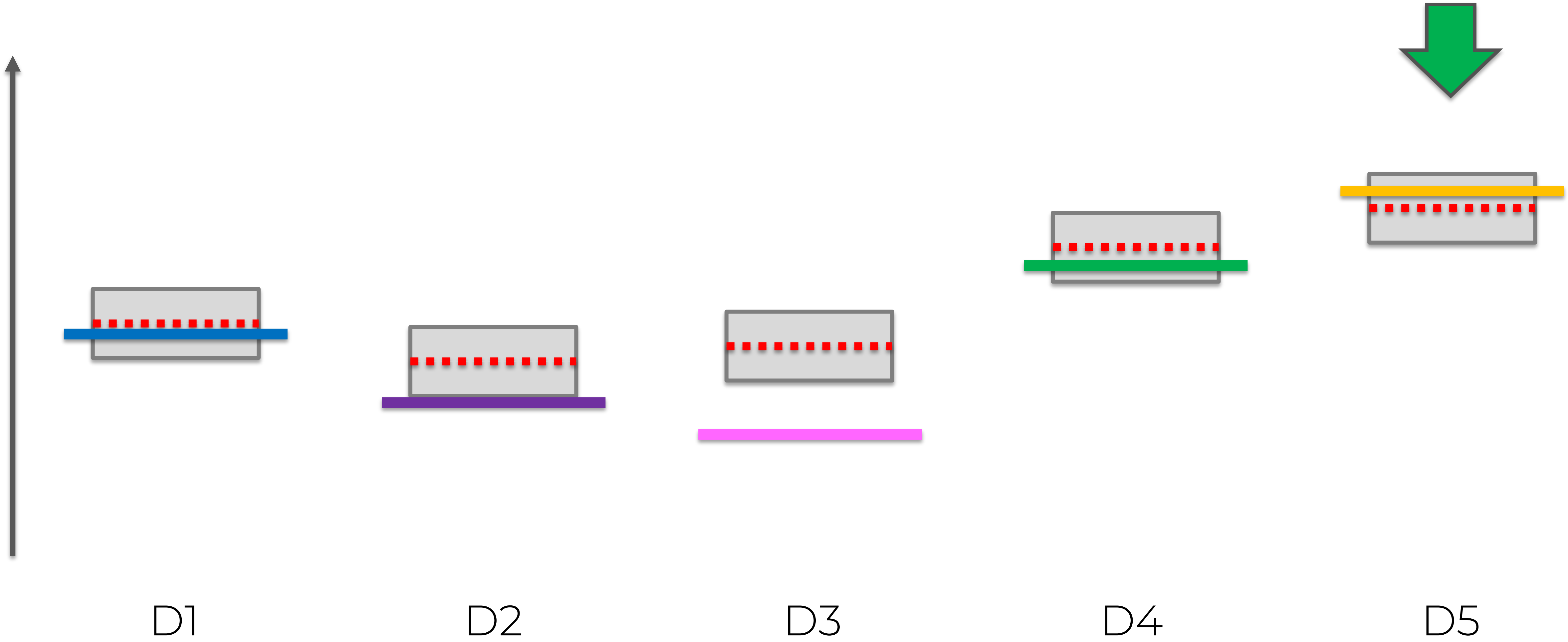
Upper Confidence Bound Algorithm



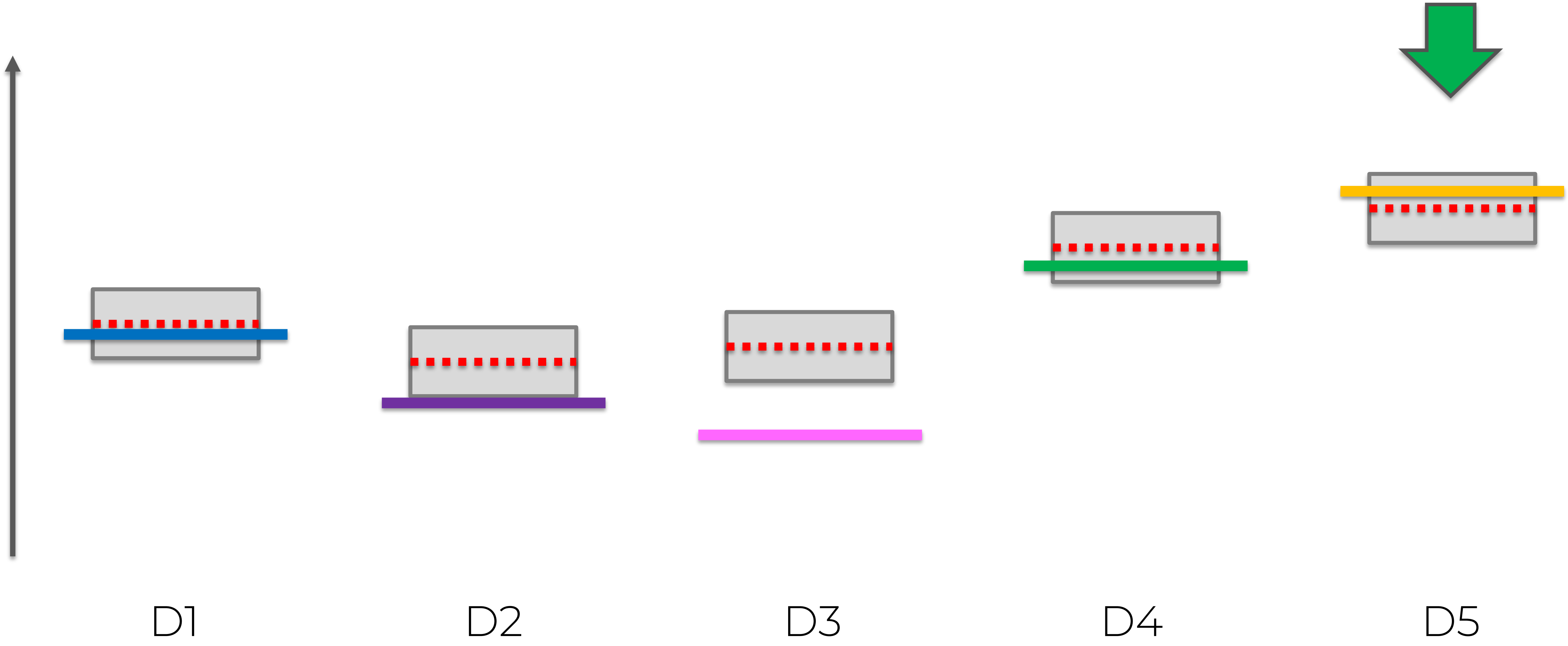
Upper Confidence Bound Algorithm



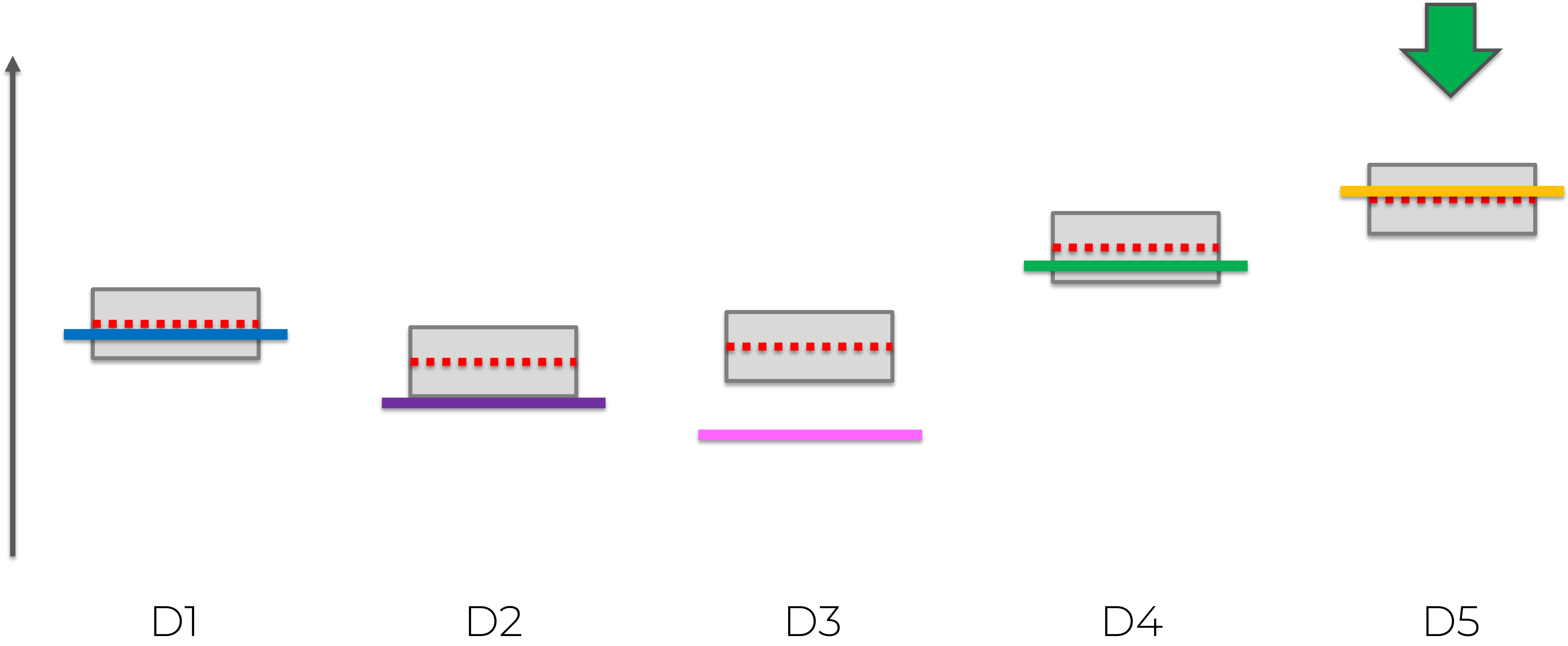
Upper Confidence Bound Algorithm



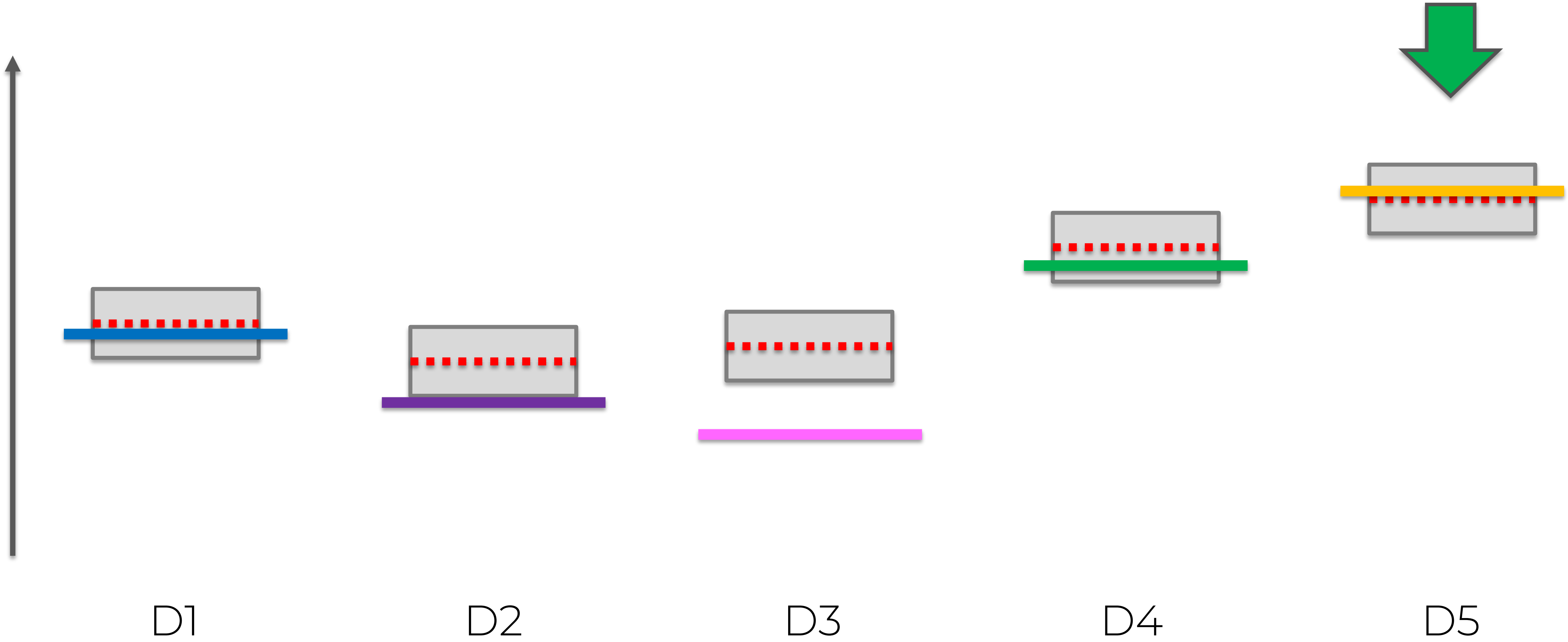
Upper Confidence Bound Algorithm



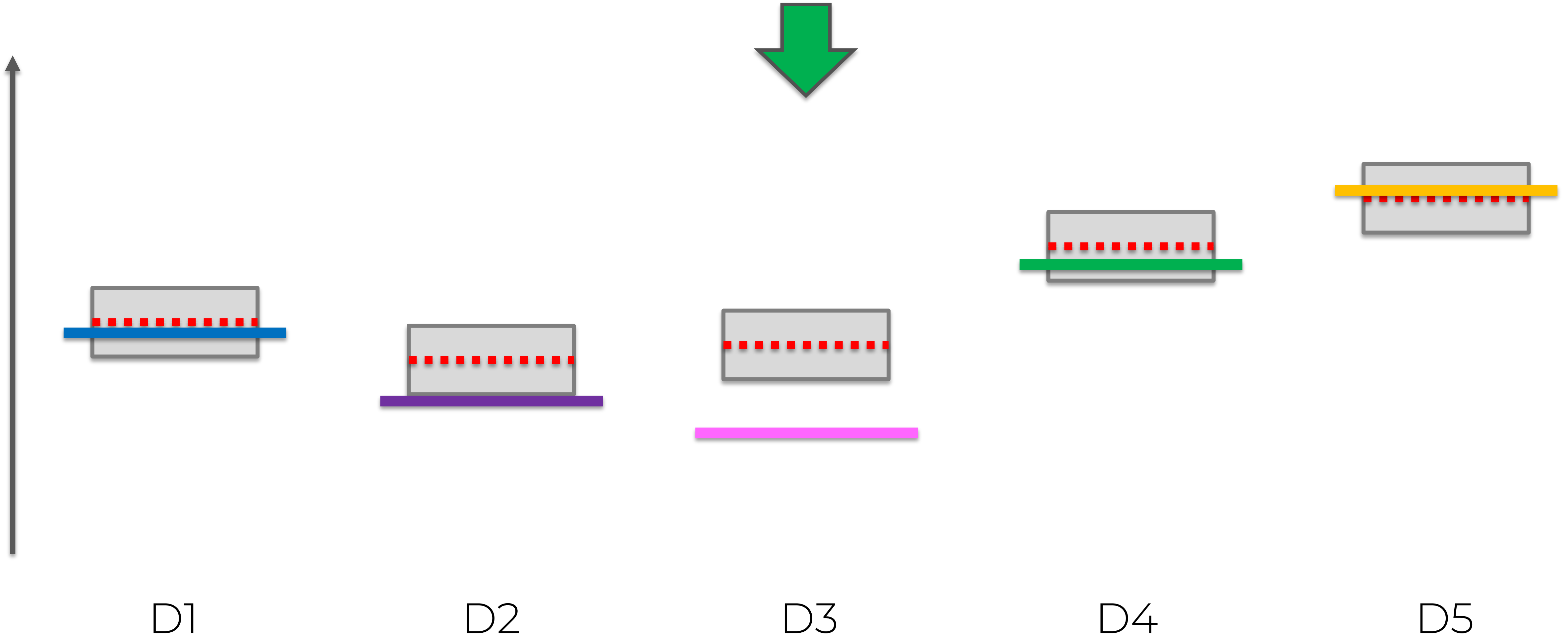
Upper Confidence Bound Algorithm



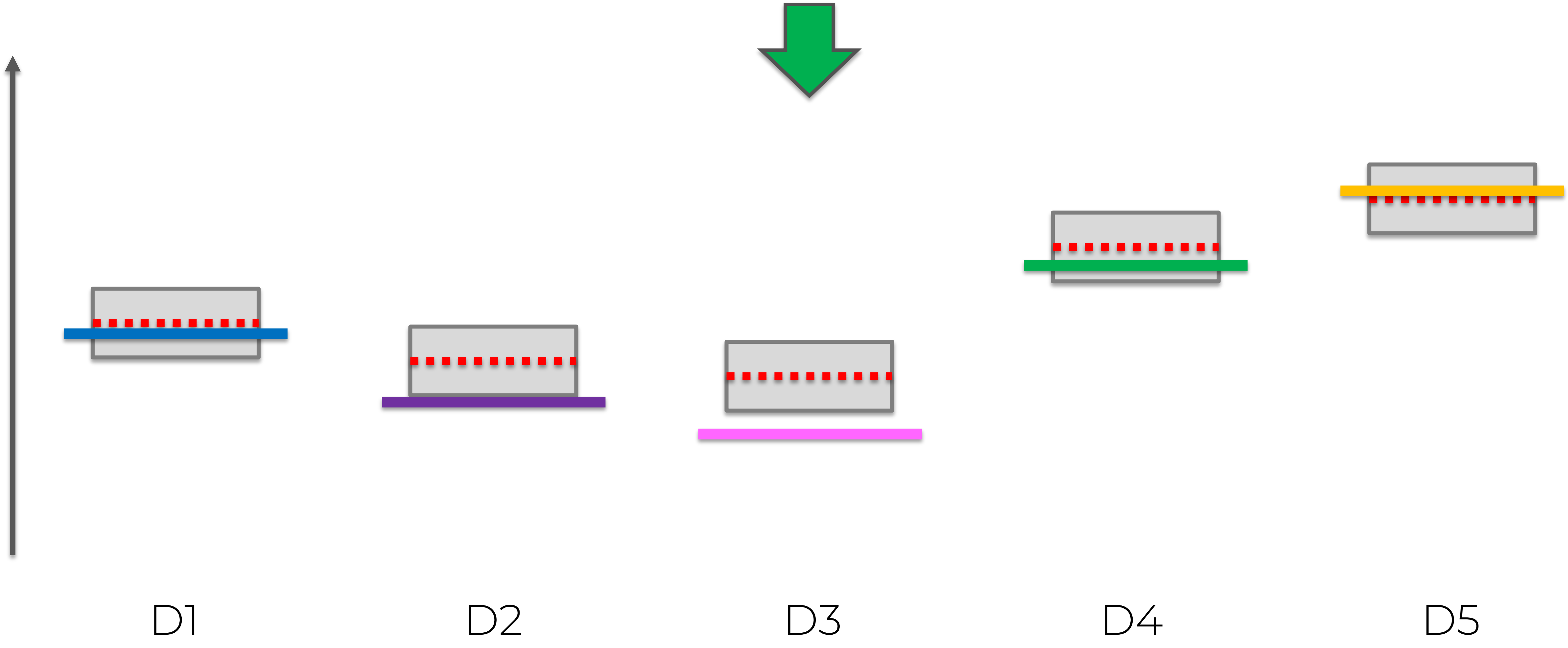
Upper Confidence Bound Algorithm



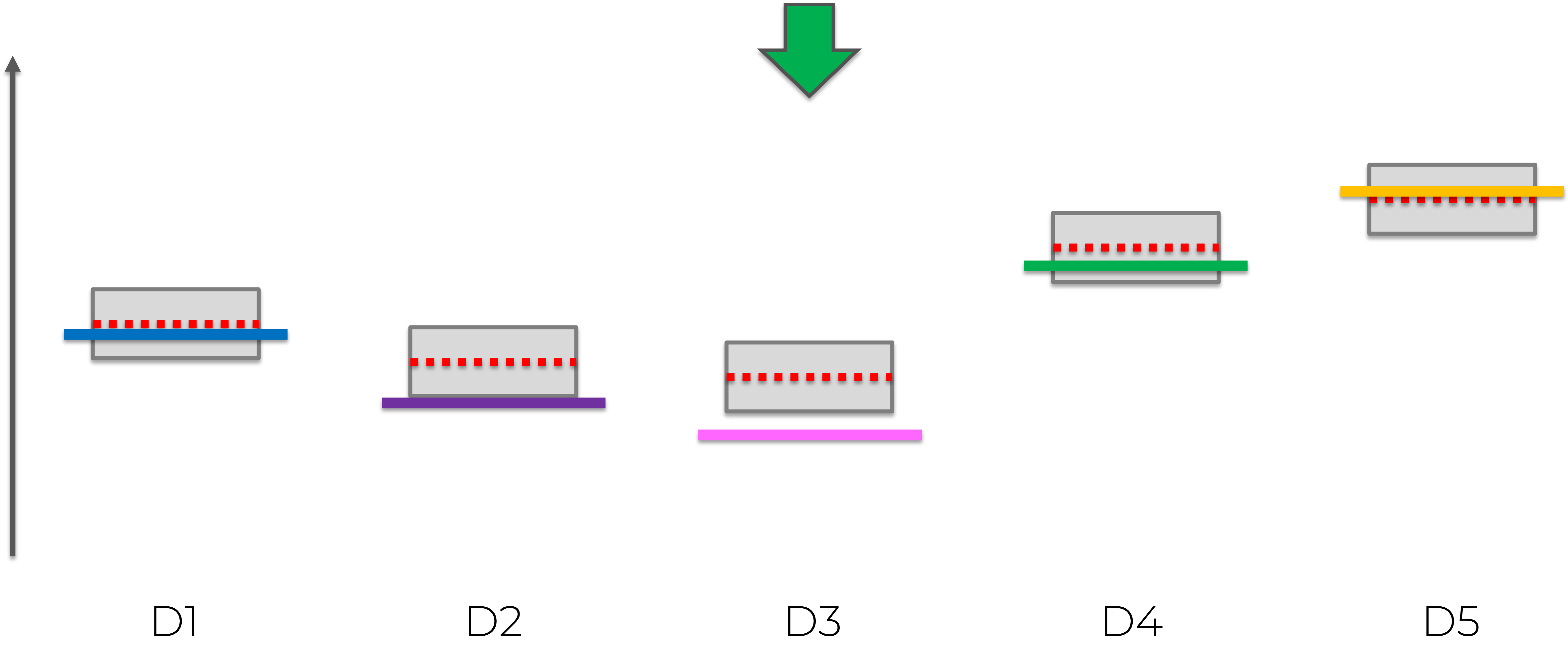
Upper Confidence Bound Algorithm



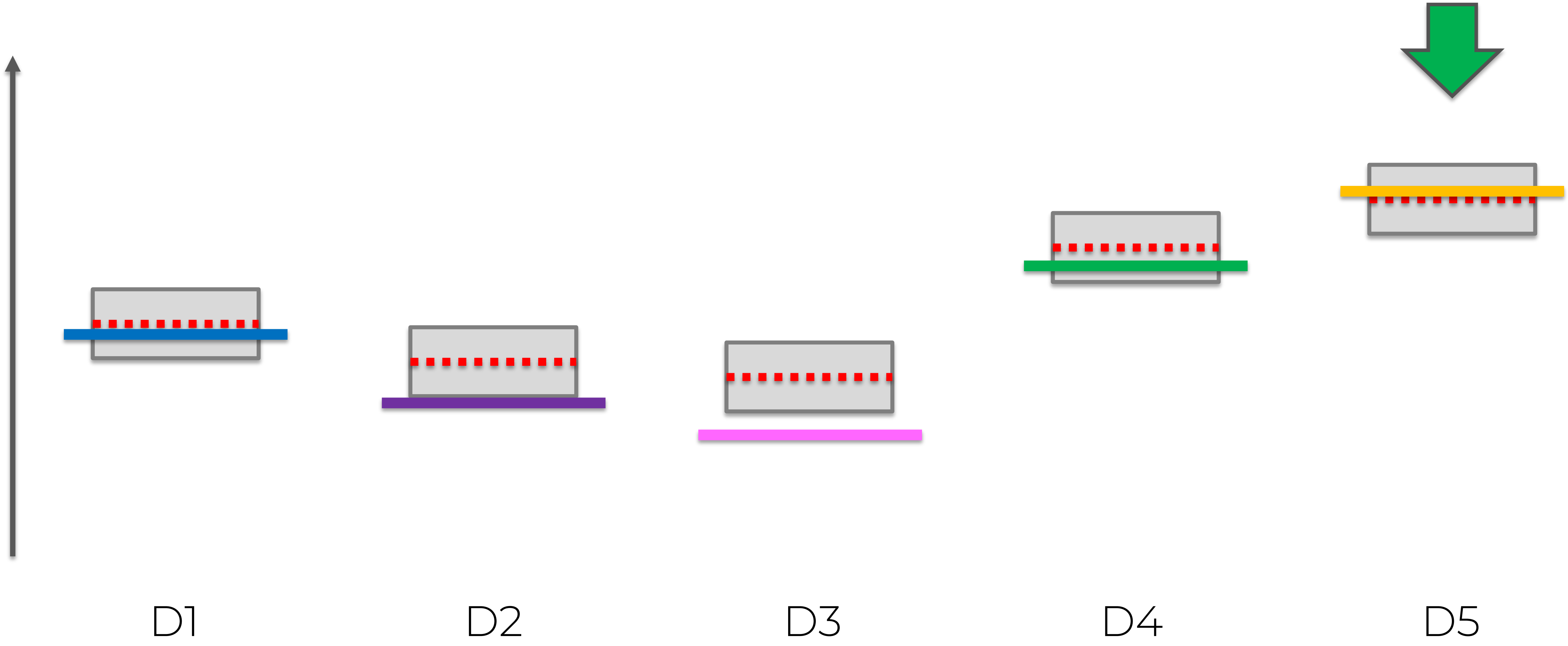
Upper Confidence Bound Algorithm



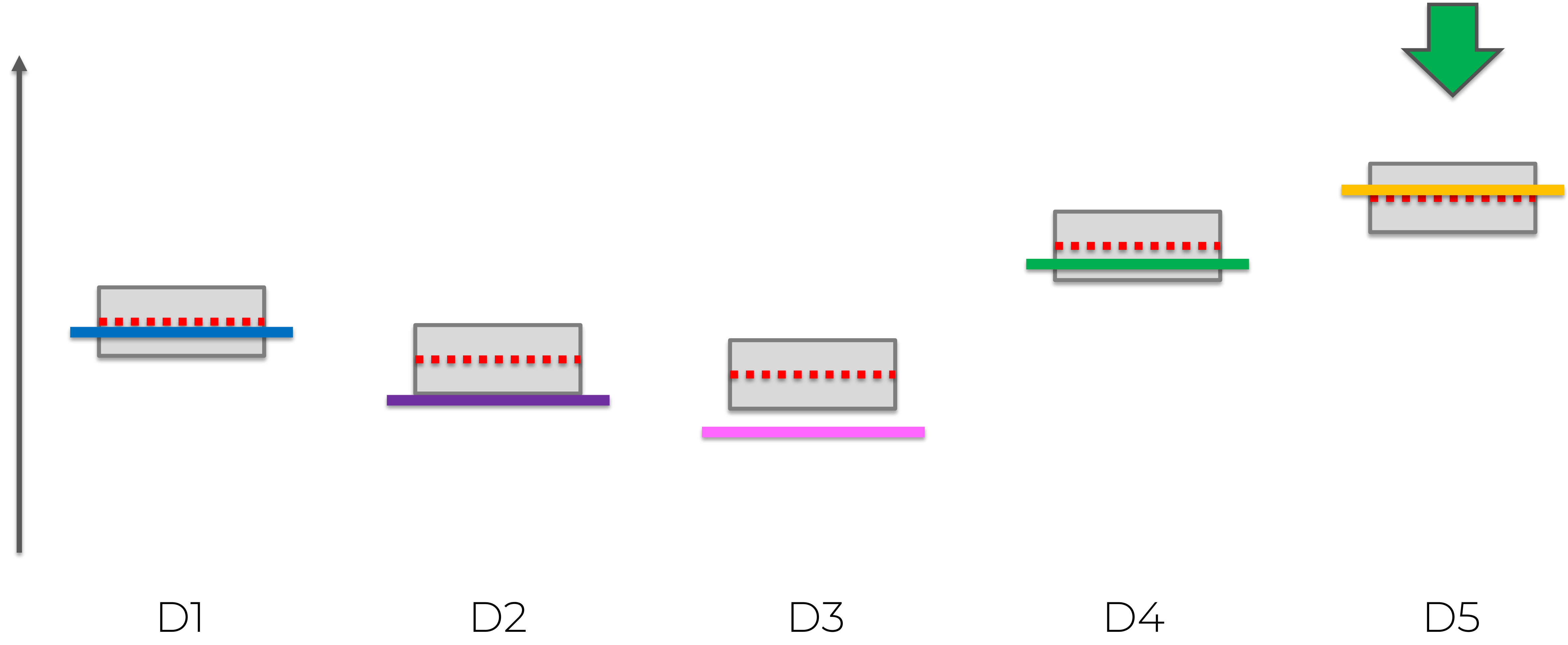
Upper Confidence Bound Algorithm



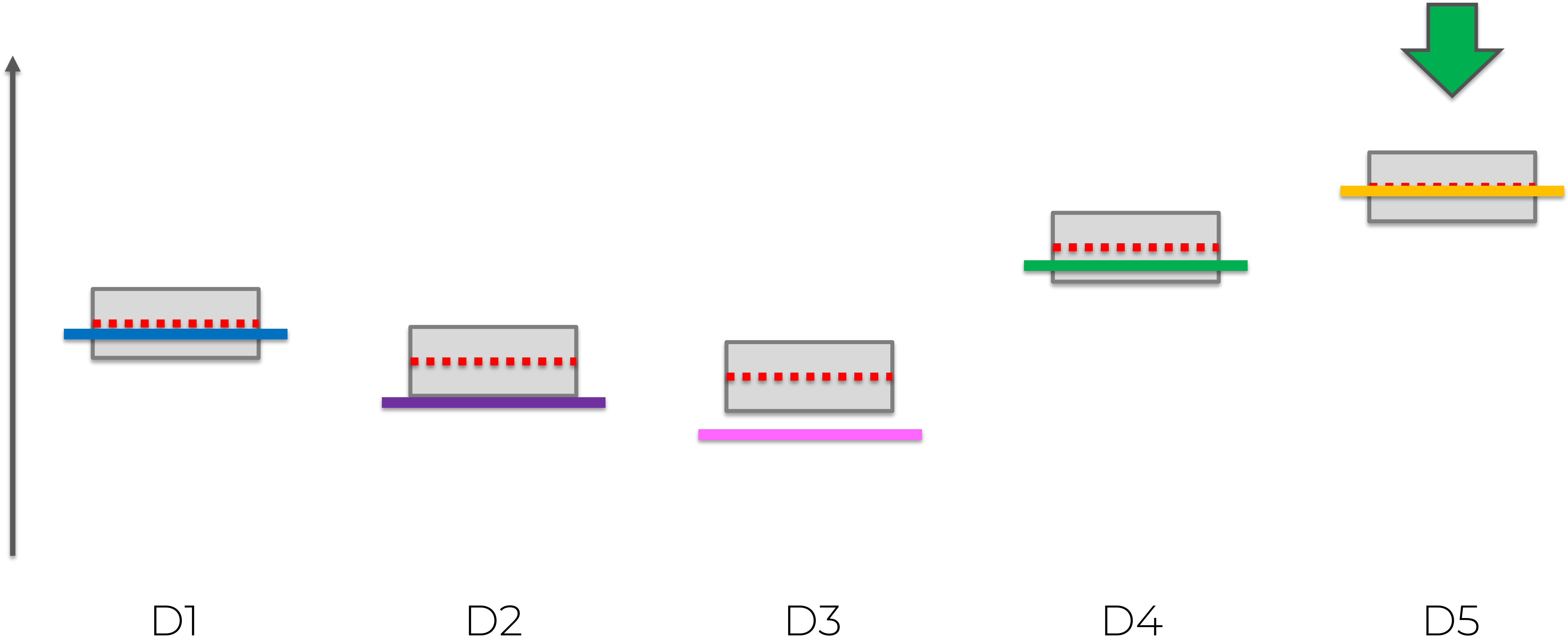
Upper Confidence Bound Algorithm



Upper Confidence Bound Algorithm



Upper Confidence Bound Algorithm



Upper Confidence Bound Algorithm