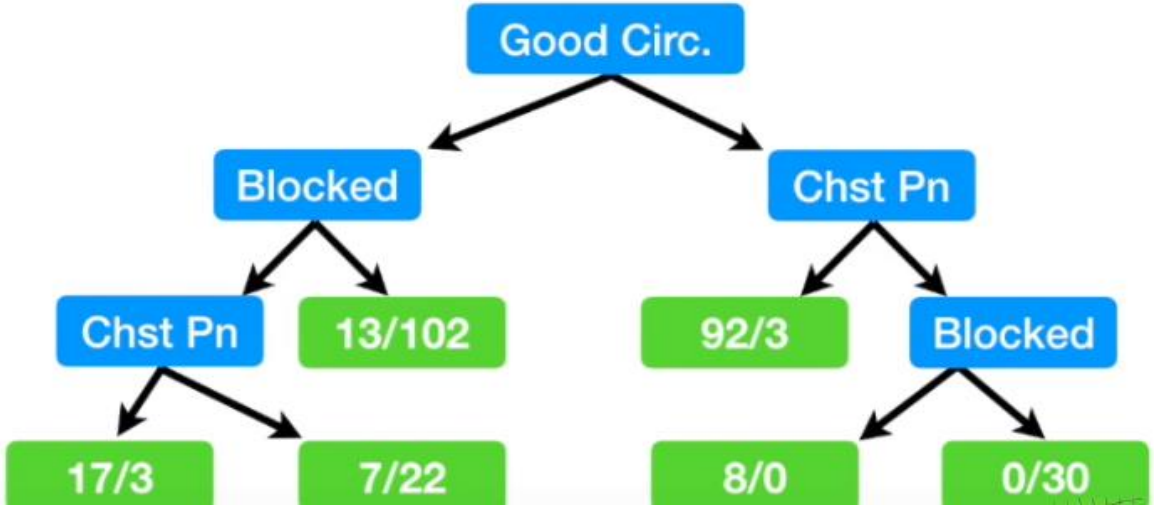
CSE 445
Lecture 13

Decision Tree (part 2)

Feature selection

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



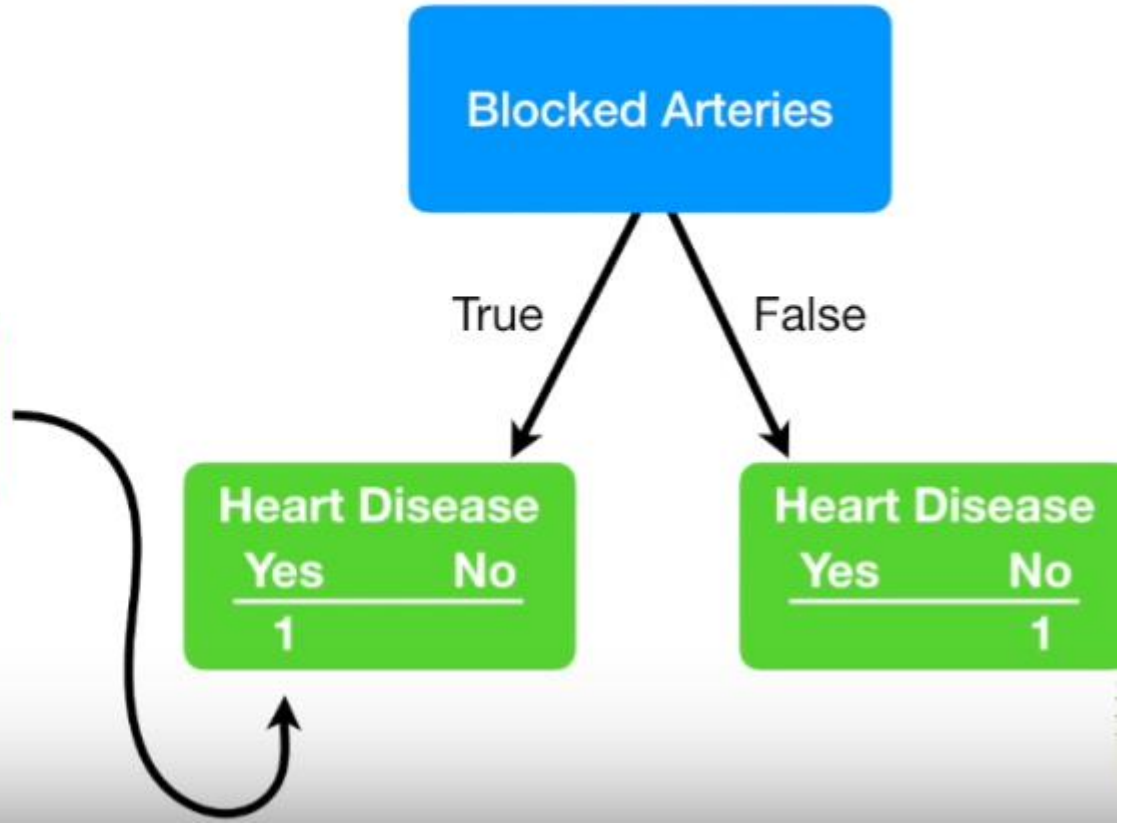
Feature selection

- We use impurity to decide if a feature is important
  - Larger impurity means the feature is not good
  - Gives hints about the importance of the feature for classification
- We can set a threshold for keeping a feature
- Thus we can use decision tree for selecting features

Dealing with missing values

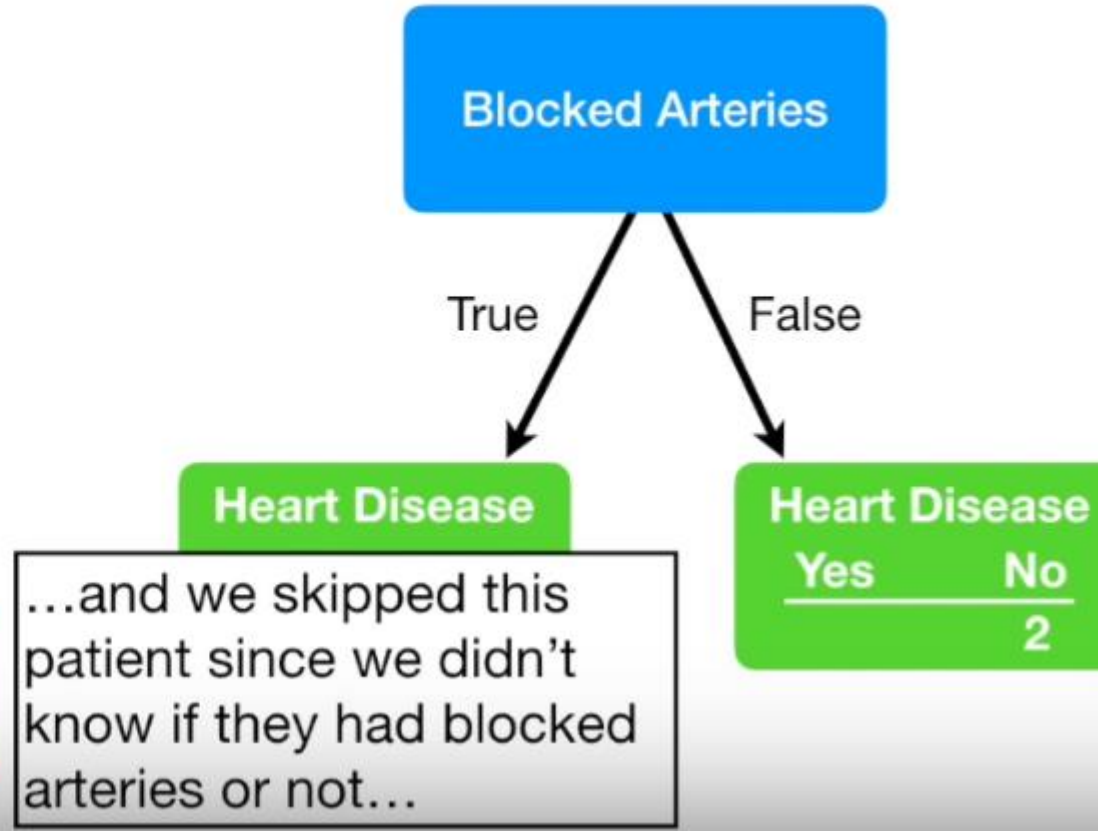
In the first video on decision trees, we calculated impurity for blocked arteries…

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes



Dealing with missing values

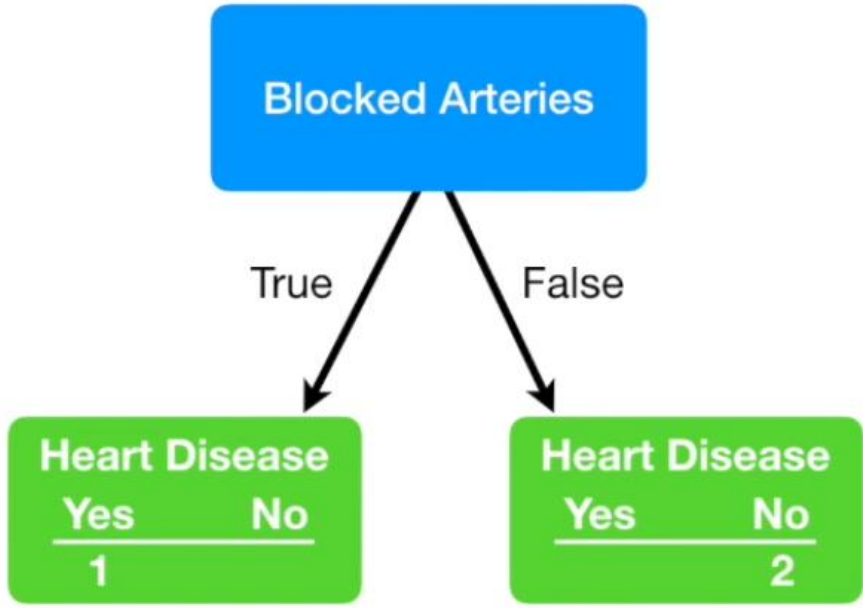
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes



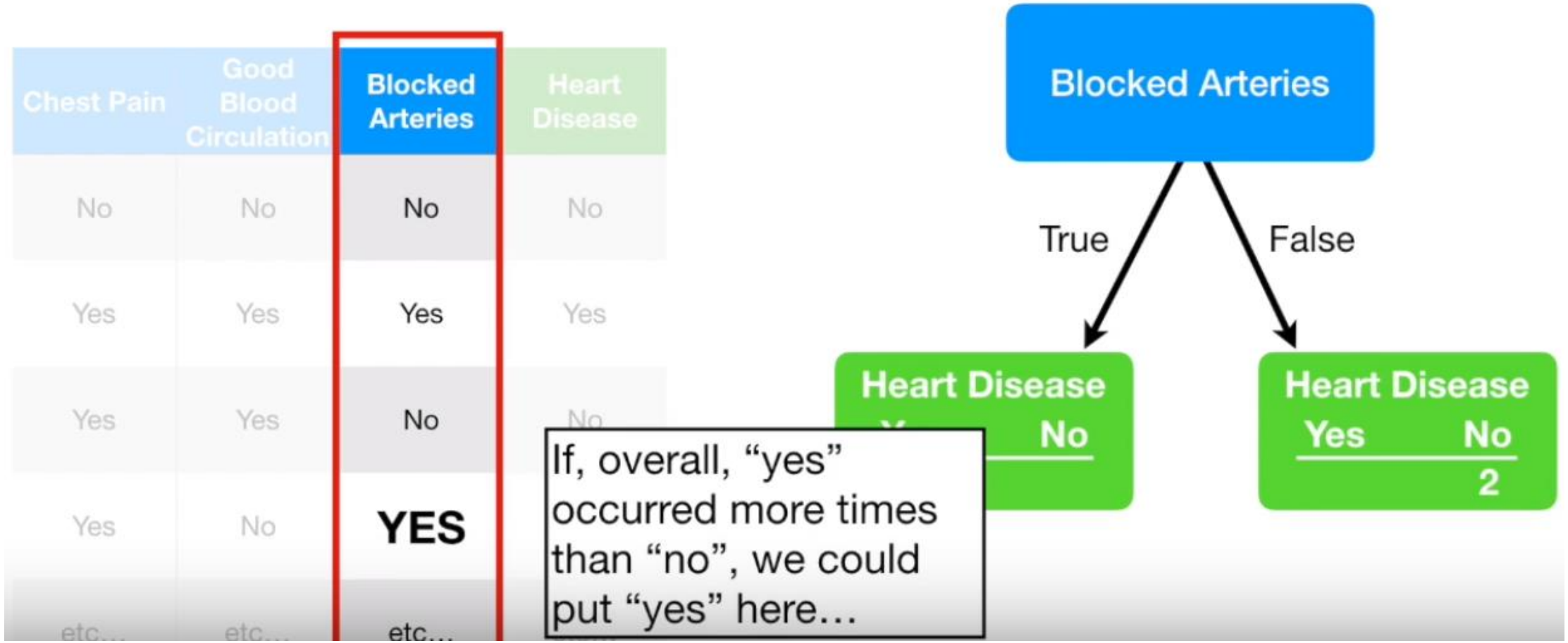
Dealing with missing values

We could pick the most common option…

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes

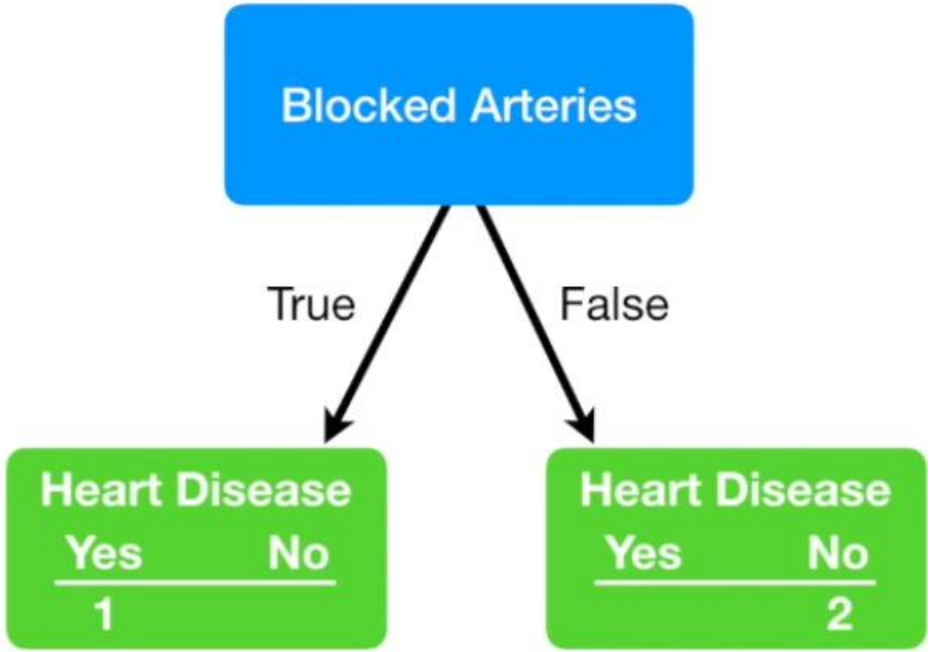


Dealing with missing values



Alternatively, we could find another column that has the highest correlation with blocked arteries and use that as a guide.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
No	Yes	No	No
Yes	No	???	Yes
etc	etc	etc	etc

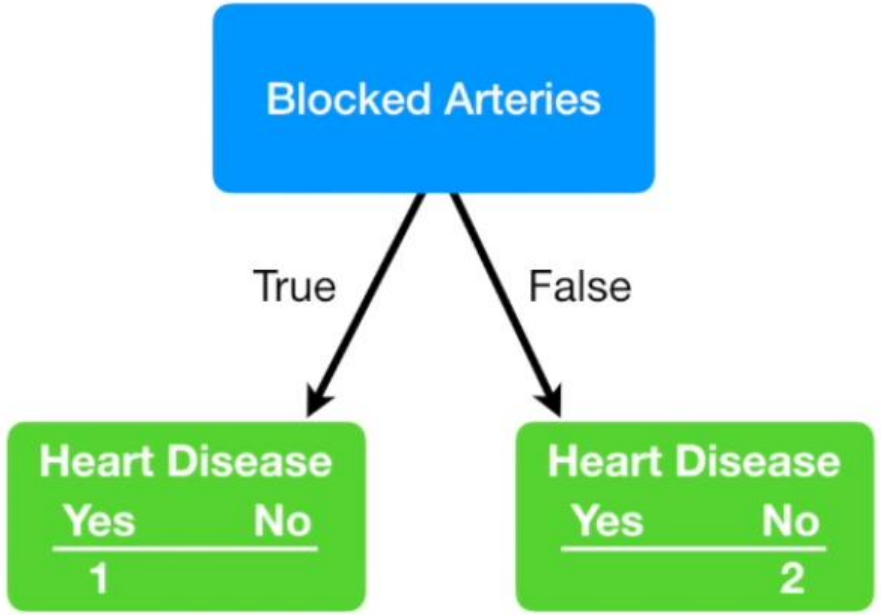


Dealing with missing values

In this case, Chest Pain and Blocked Arteries are often very similar.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	
Yes	Yes	Yes	Yes
No	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

Both are "No"

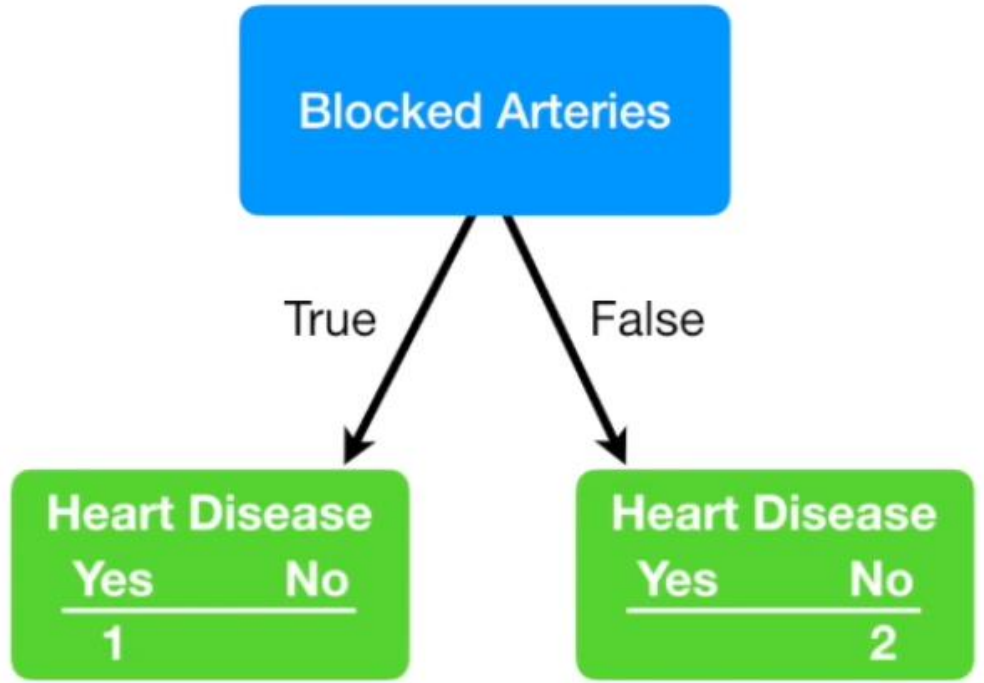


Dealing with missing values

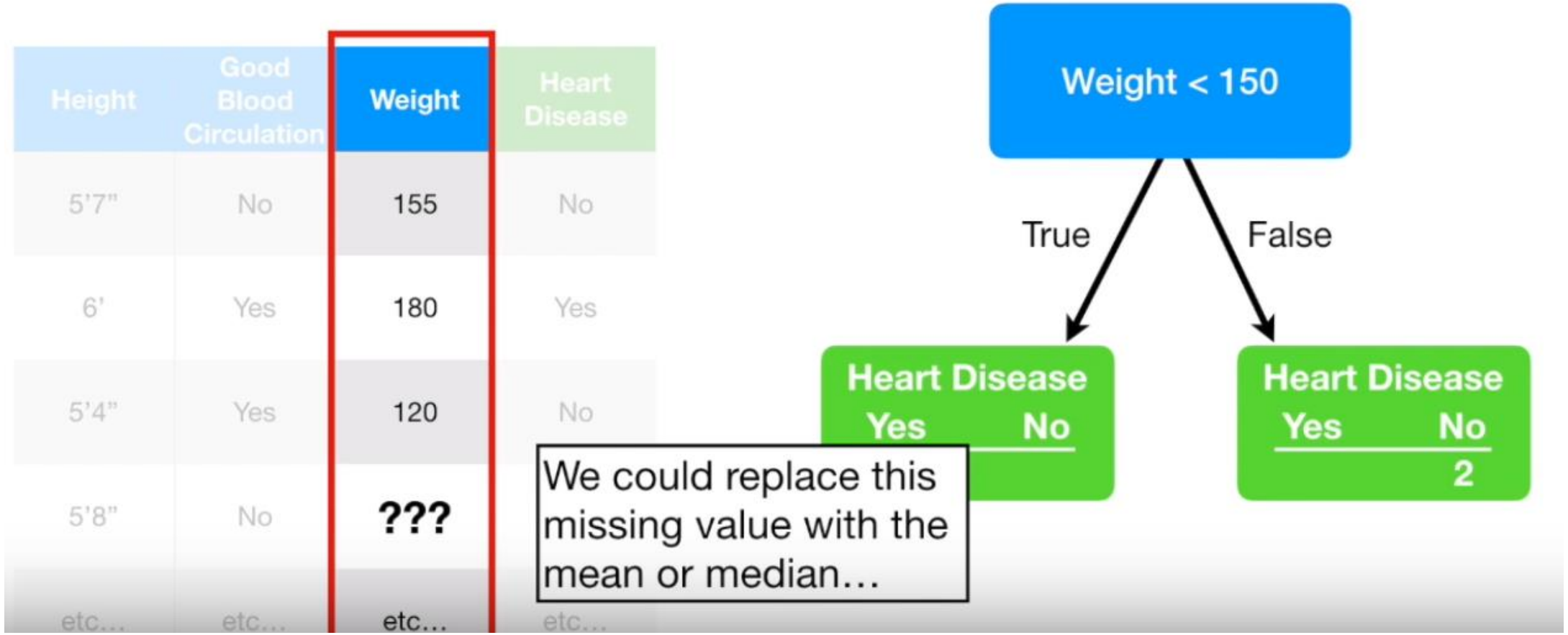
In this case, Chest Pain and Blocked Arteries are often very similar.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
No	Yes	No	No
Yes	No	???	
etc...	etc...	etc...	etc...

Since Chest Pain is "Yes"…



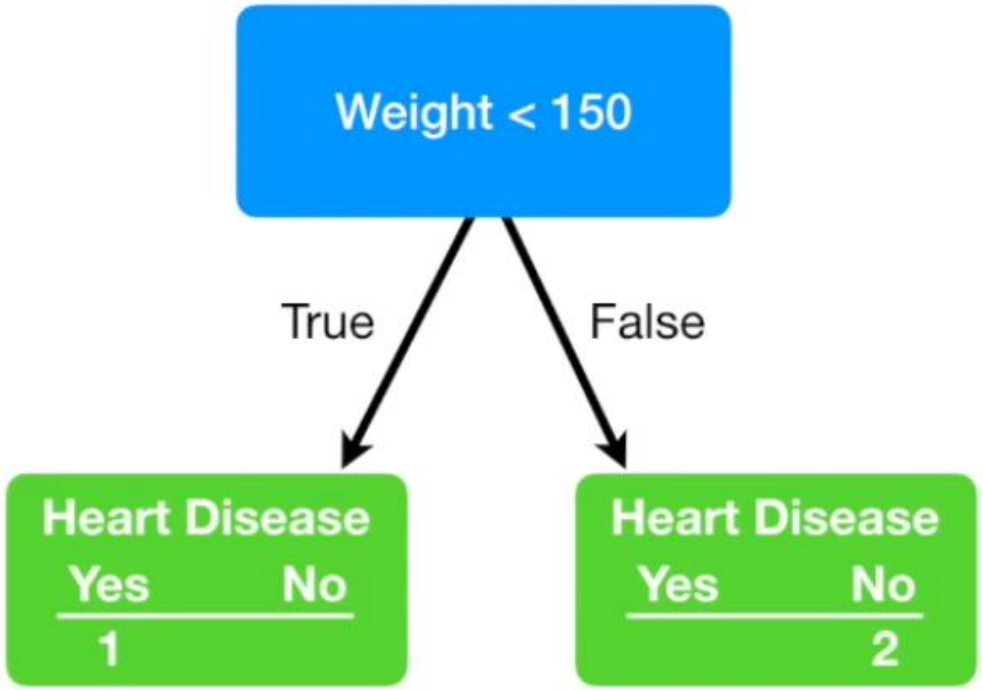
Dealing with missing values



Dealing with missing values

Alternatively, we could find another column that has the highest correlation with weight…

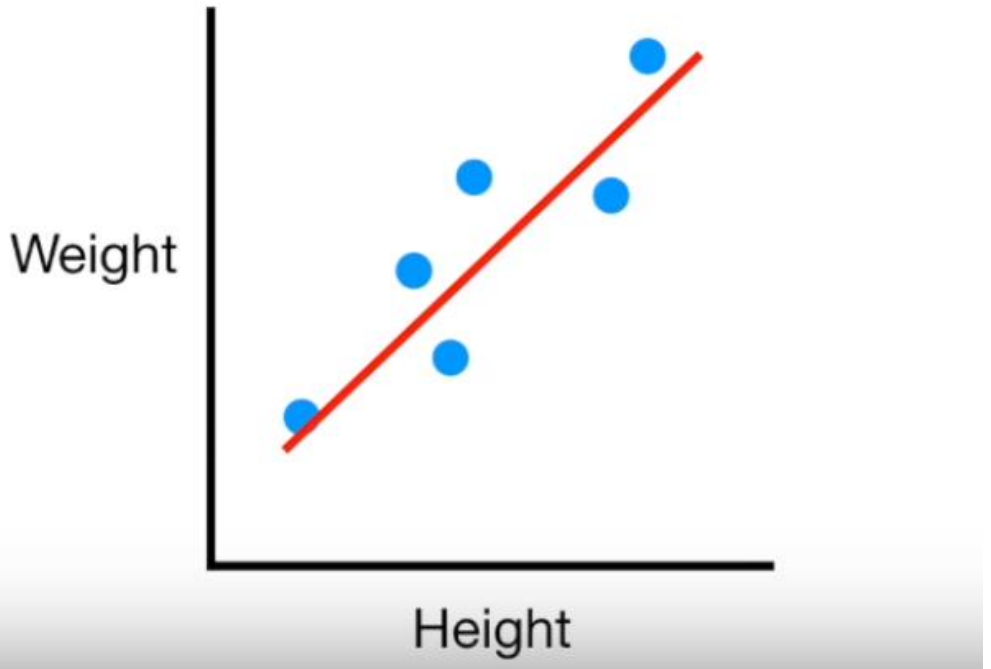
Height	Good Blood Circulation	Weight	Heart Disease
5'7"	No	155	No
6'	Yes	180	Yes
5'4"	Yes	120	No
5'8"	No	???	Yes
etc…	etc…	etc…	etc…



Dealing with missing values

Height	Good Blood Circulation	Weight	Heart Disease
5'7"	No	155	No
6'	Yes	180	Yes
5'4"	Yes	120	No
5'8"	No	???	Yes
etc...	etc...	etc...	etc...

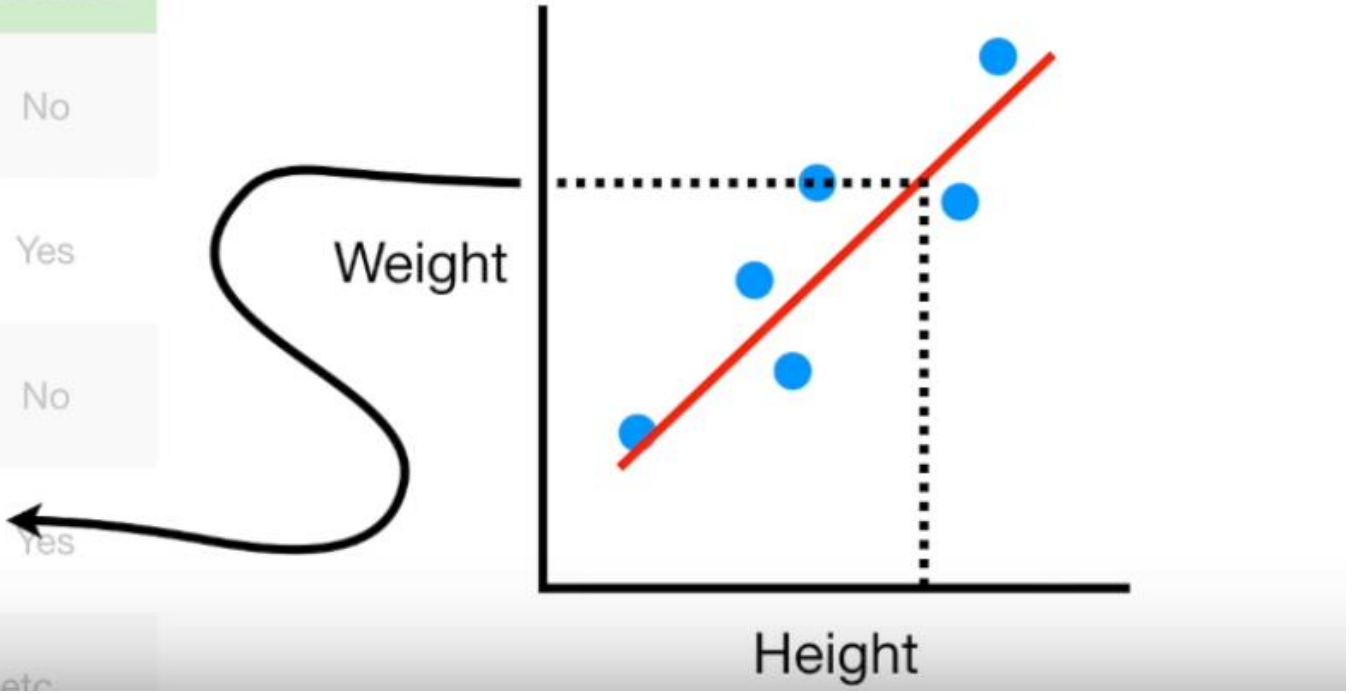
…and do a linear regression on the two columns…



Dealing with missing values

Height	Good Blood Circulation	Weight	Heart Disease
5'7"	No	155	No
6'	Yes	180	Yes
5'4"	Yes	120	No
5'8"	No	**168**	Yes
etc	etc	etc	etc

…and use the least squares line to predict the value for weight.



CART Algorithm

- Scikit-Learn uses the *Classification And Regression Tree* (CART) algorithm to train Decision Trees
- The idea is really quite simple:
- The algorithm first splits the training set in two subsets using a single feature k and a threshold T_k (e.g., "petal length ≤ 2.45 cm")
- How does it choose k and T_k? (We discussed it in the last class)
  - It searches for the pair (k, tk) that produces the purest subsets (weighted by their size)
- Once it has successfully split the training set in two, it splits the subsets using the same logic, then the sub-subsets and so on, recursively
- It stops the recursion once it reaches the maximum depth (defined by the max_depth hyper-parameter), or if it cannot find a split that will reduce impurity

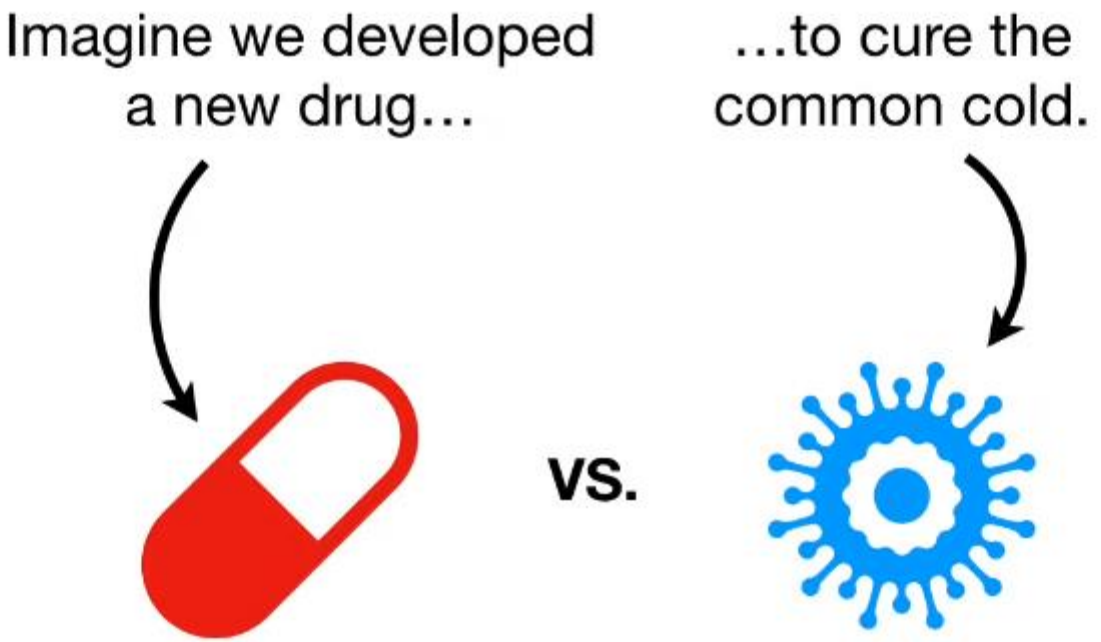
Complexity of the algorithm

- Suppose we have n examples and m features
- Decision Trees are approximately balanced, so traversing the Decision Tree requires going through roughly $O(log2(m))$ nodes – the height of the tree
  - So predictions are very fast, even when dealing with large training sets
- However, the training algorithm compares all features
- This results in a training complexity of $O(n \times m\ log(m))$
- For small training sets (less than a few thousand instances), Scikit-Learn can speed up training by presorting the data (set presort=True), but this slows down training considerably for larger training sets

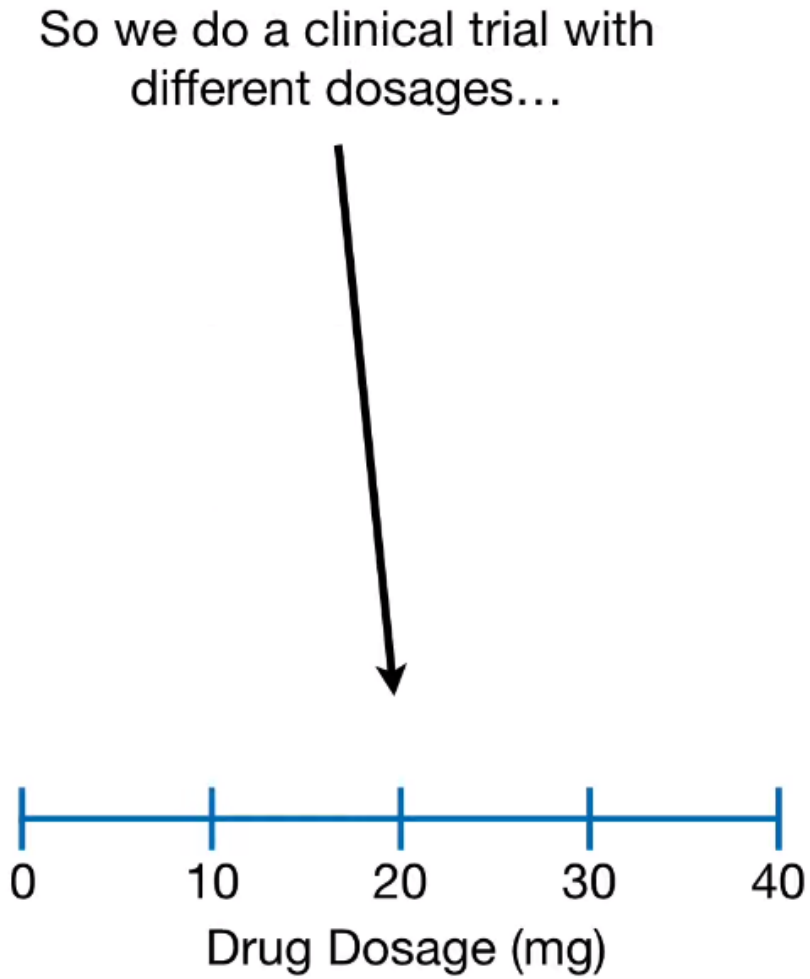
Regularization

- Decision Trees make very few assumptions about the training data (as opposed to linear models, which assume that the data is linear)
- If left unconstrained, the tree structure will adapt itself to the training data, fitting it very closely, and most likely overfitting it
- The DecisionTreeClassifier class has a few parameters that similarly restrict the shape of the Decision Tree:
  - max_depth – maximum depth of the Decision Tree
  - min_samples_split – the minimum number of samples a node must have before it can be split)
  - min_samples_leaf – the minimum number of samples a leaf node must have)
  - min_weight_fraction_leaf – same as min_samples_leaf but expressed as a fraction of the total number of weighted

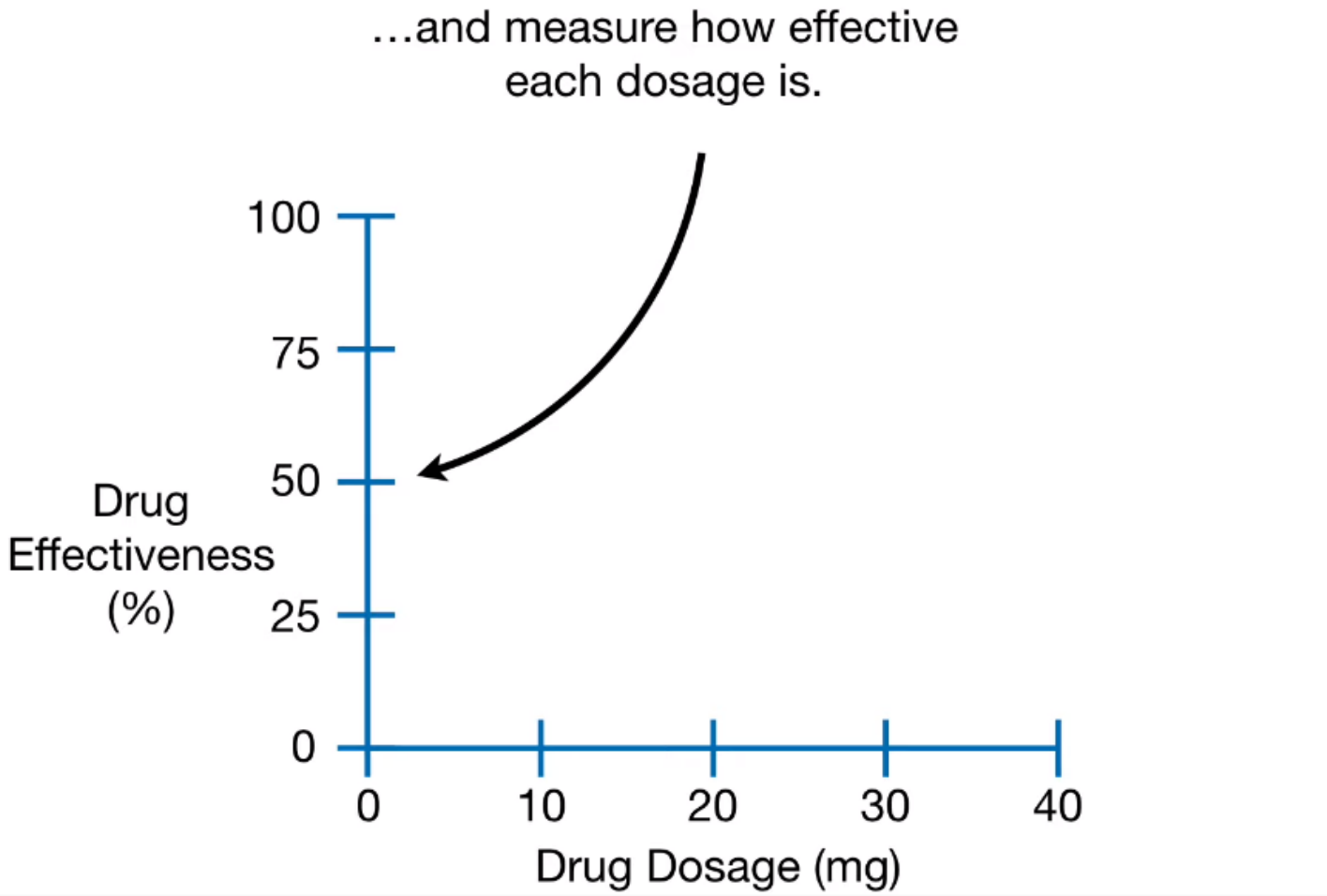
Decision Tree Regression



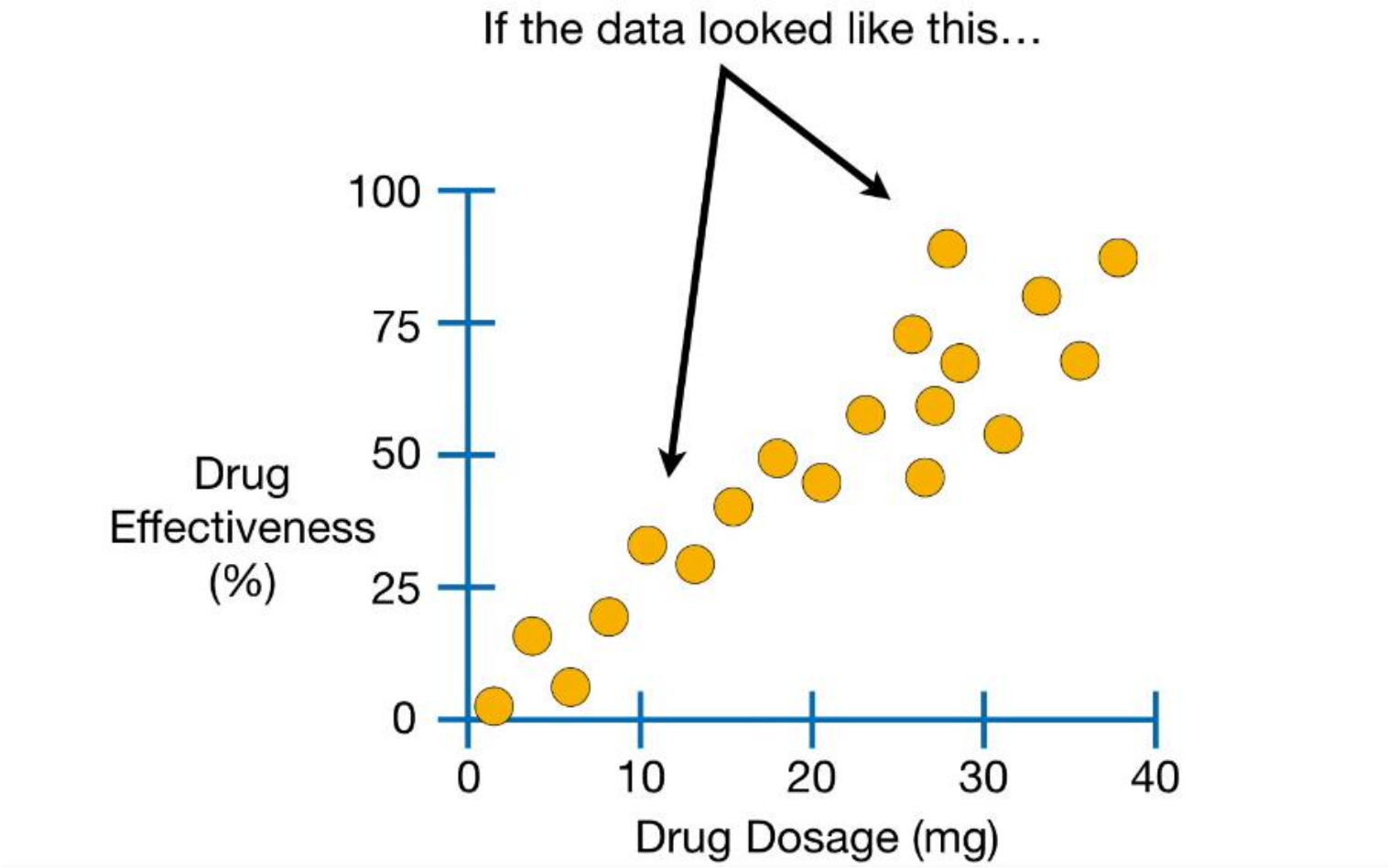
Decision Tree Regression



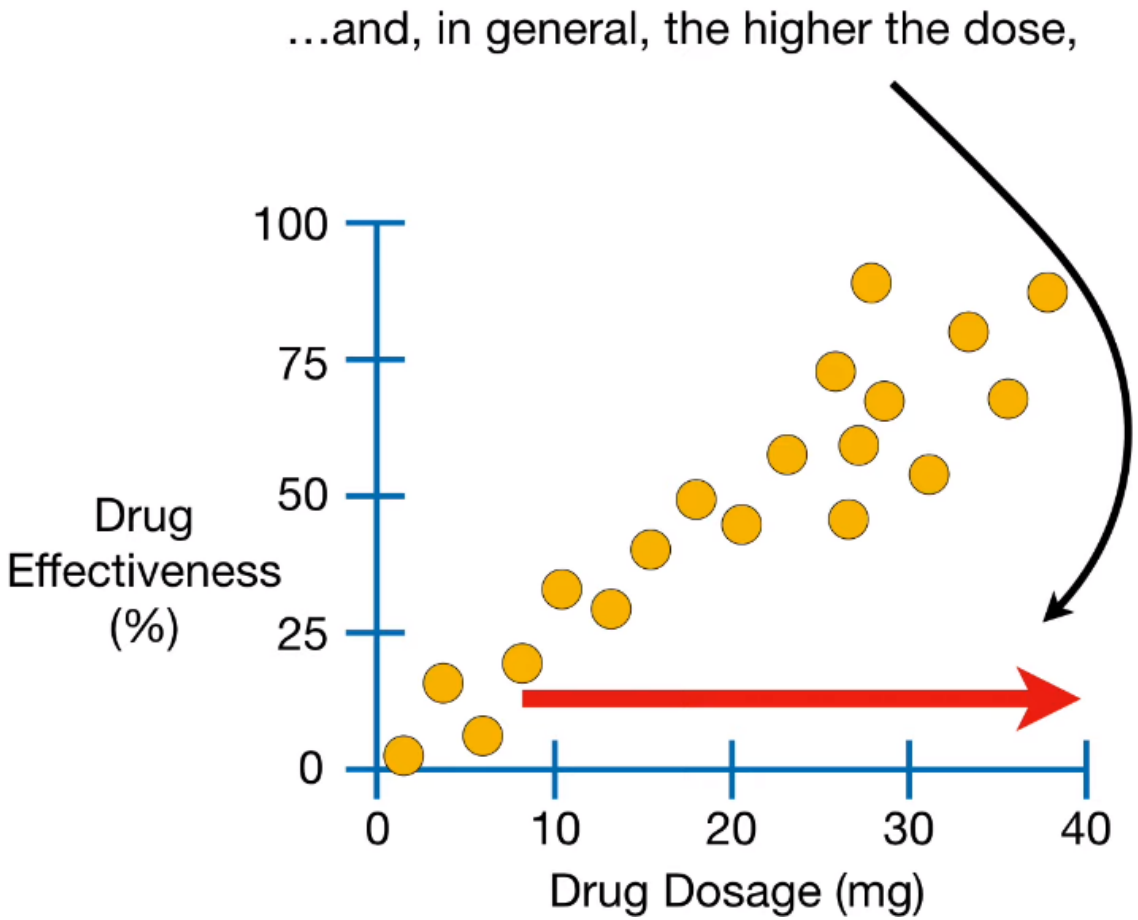
Decision Tree Regression



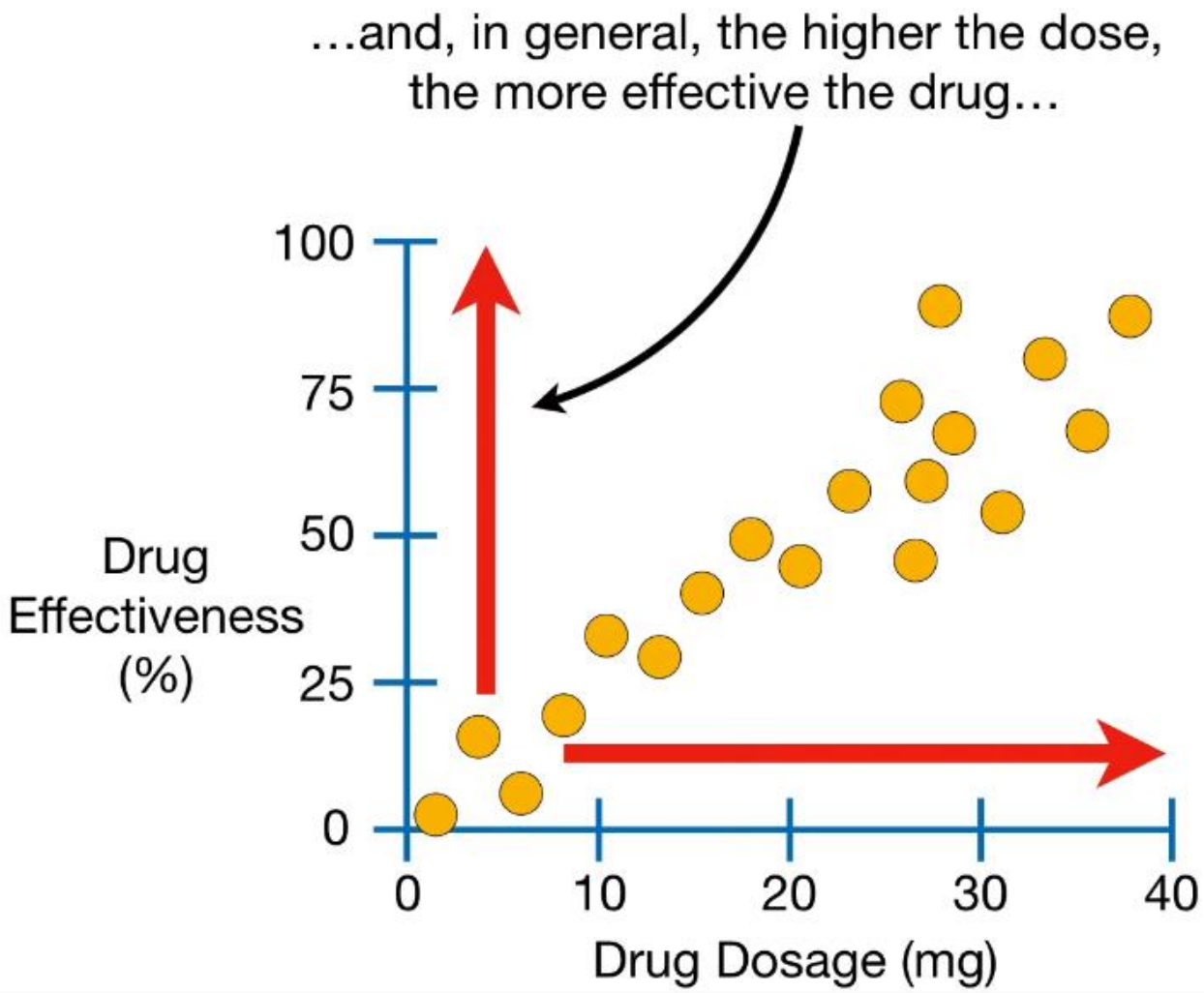
Decision Tree Regression



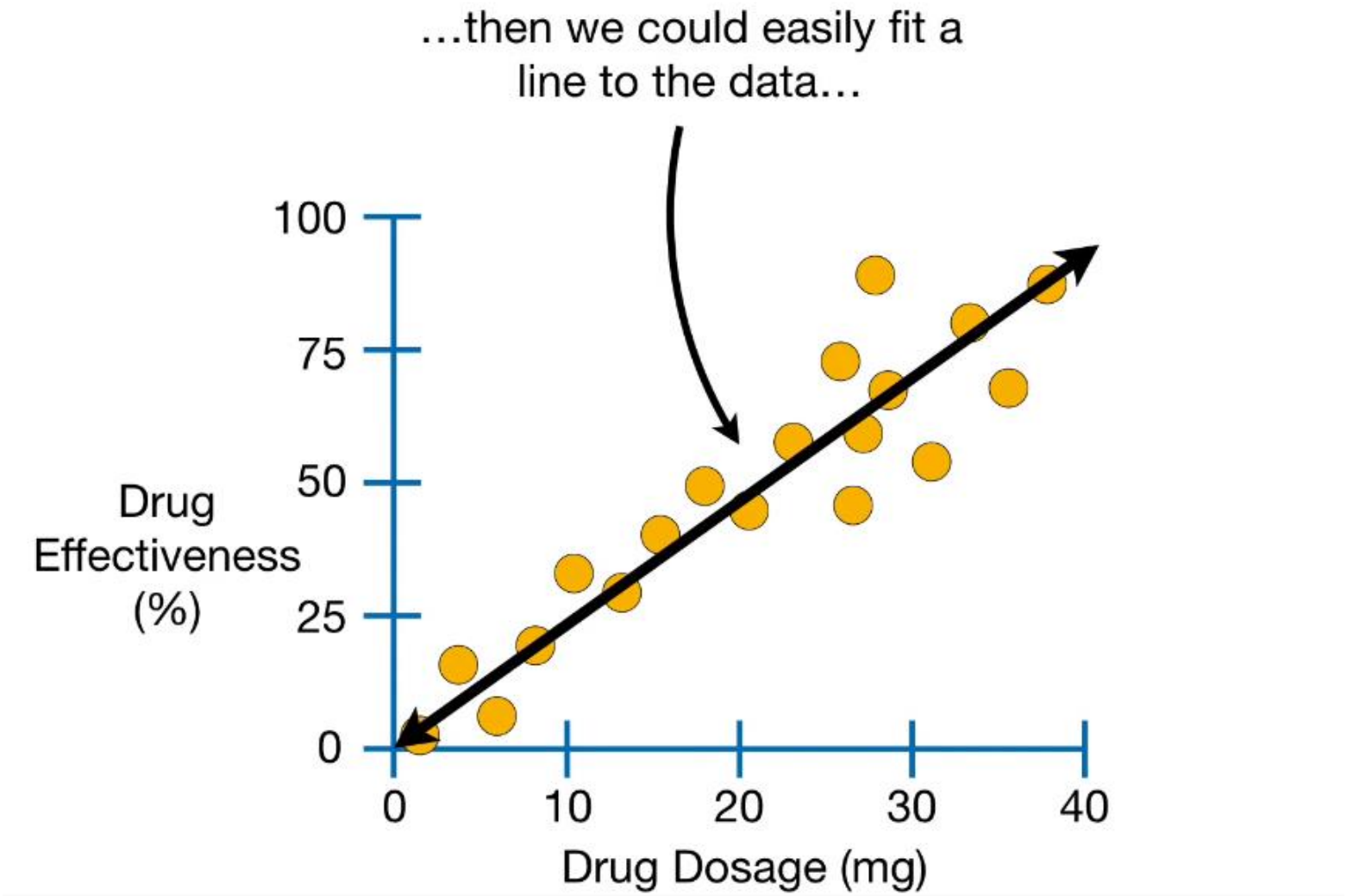
Decision Tree Regression



Decision Tree Regression

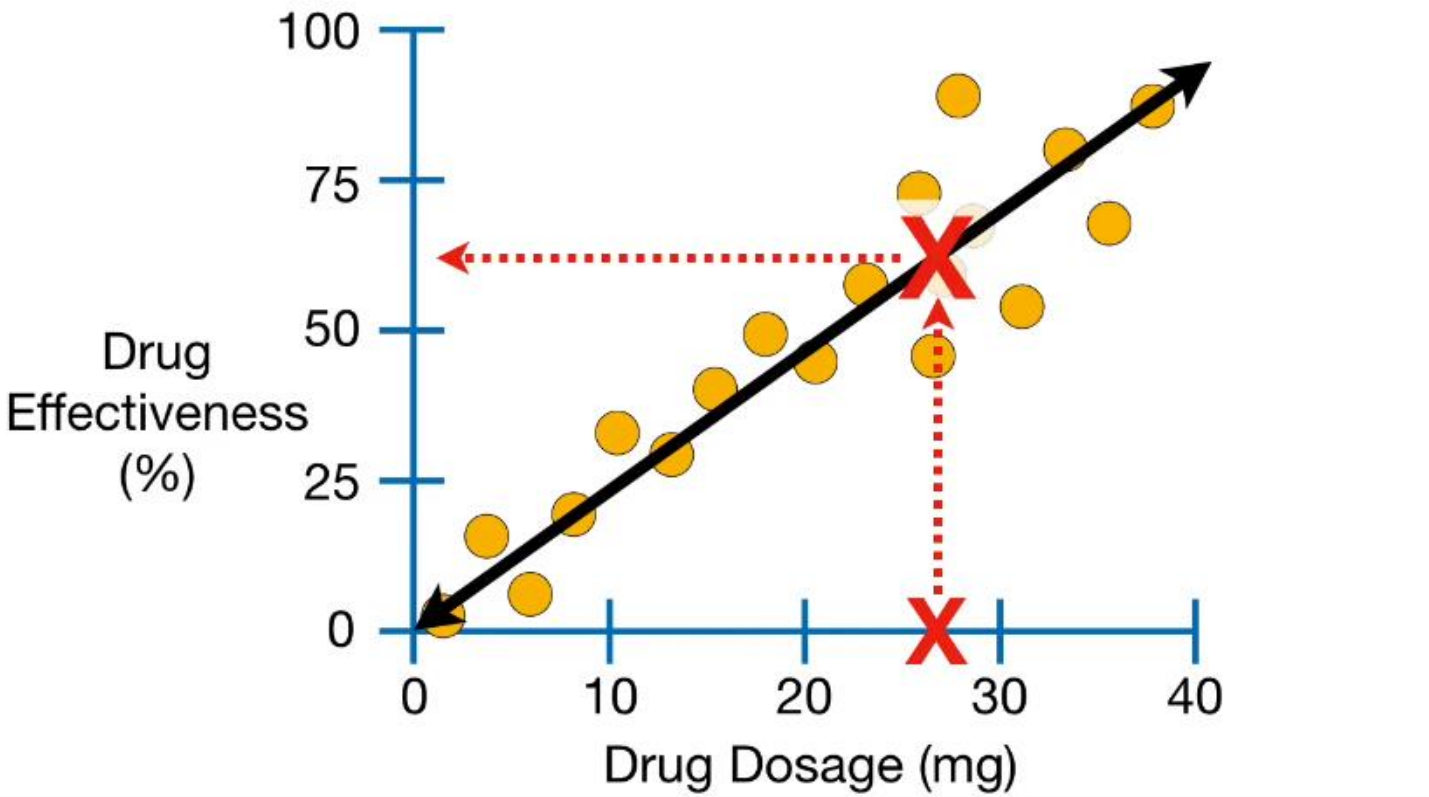


Decision Tree Regression

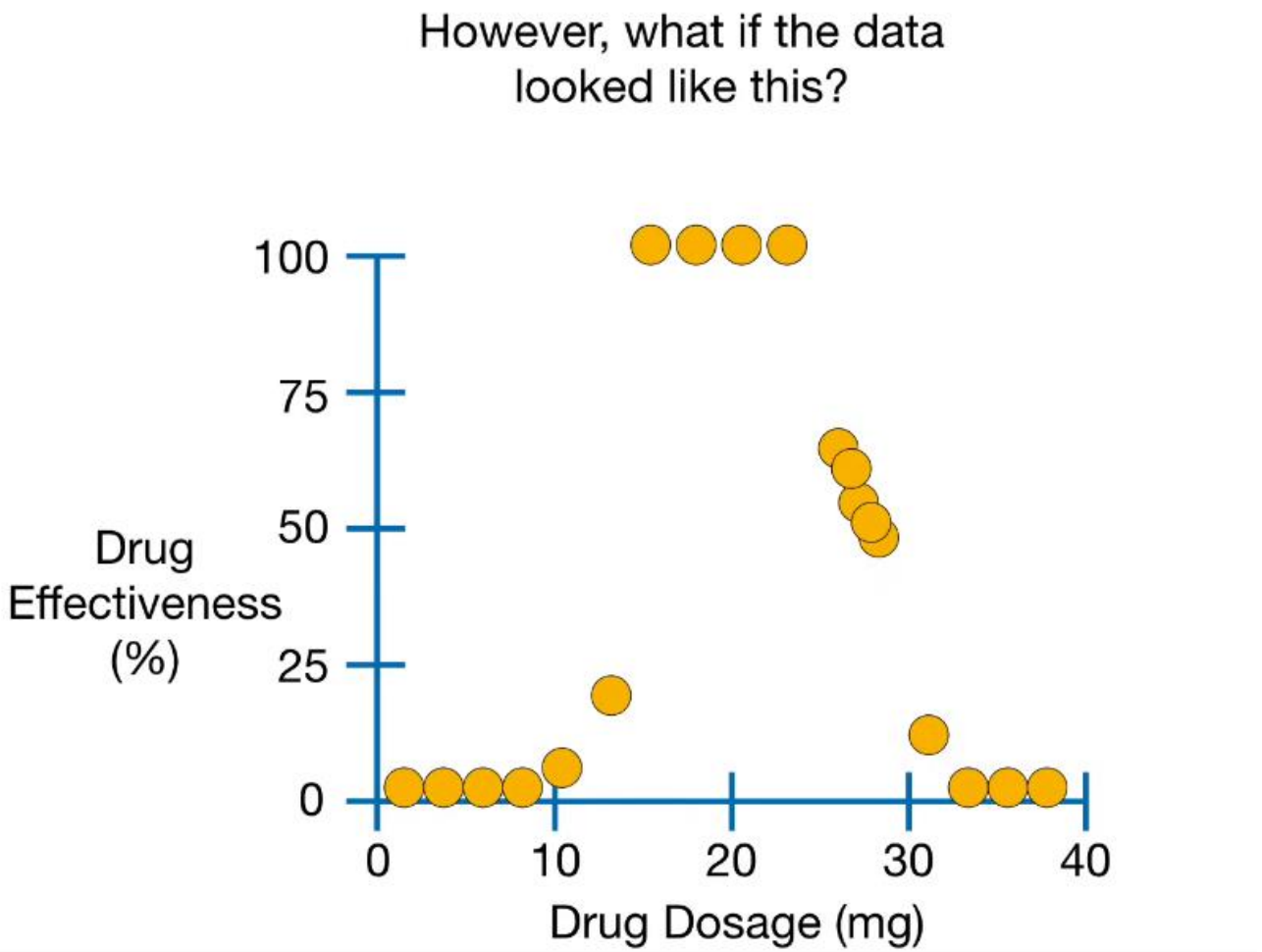


Decision Tree Regression

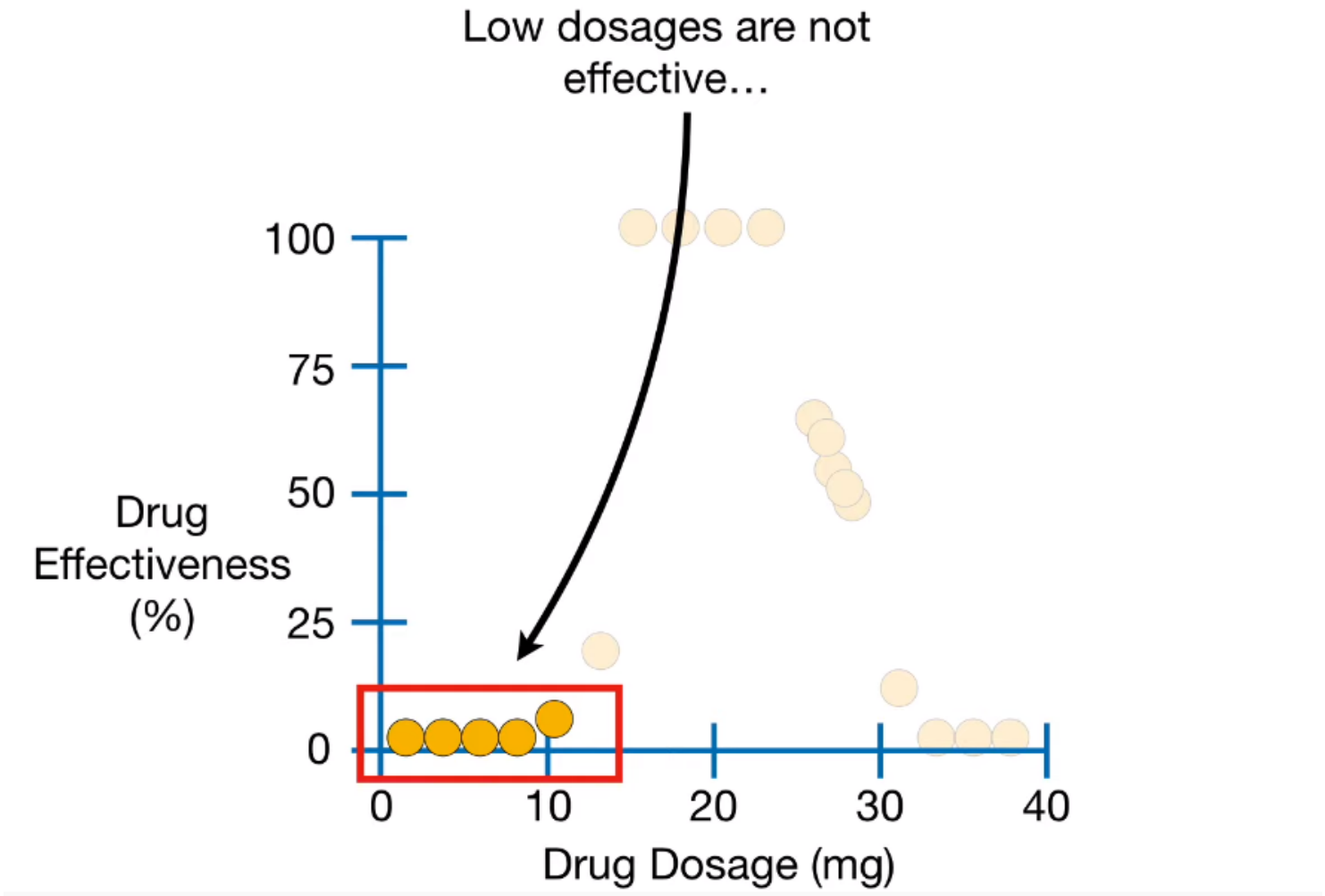
…we could use the line to predict that a **27 mg Dose** should be **62% Effective**.



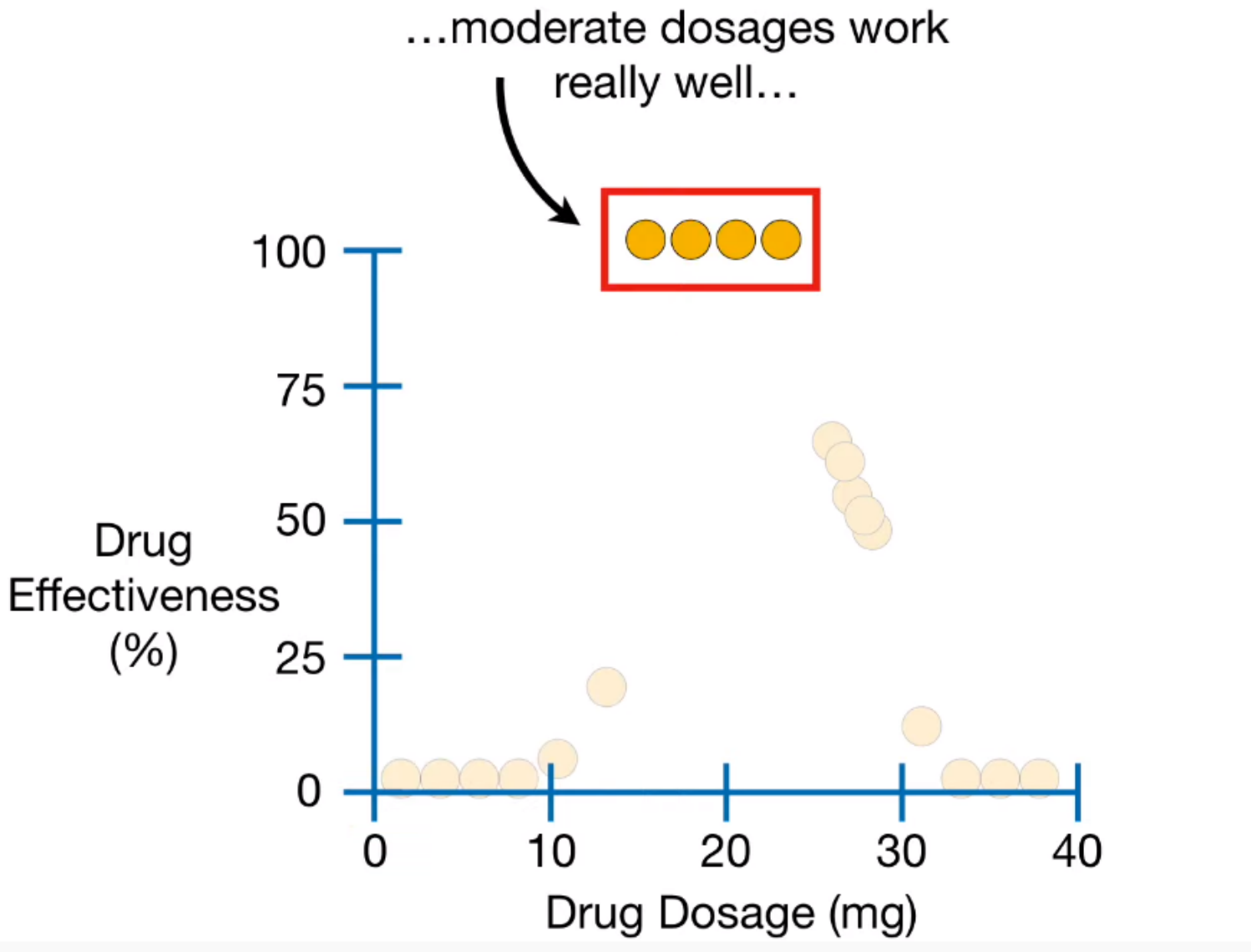
Decision Tree Regression



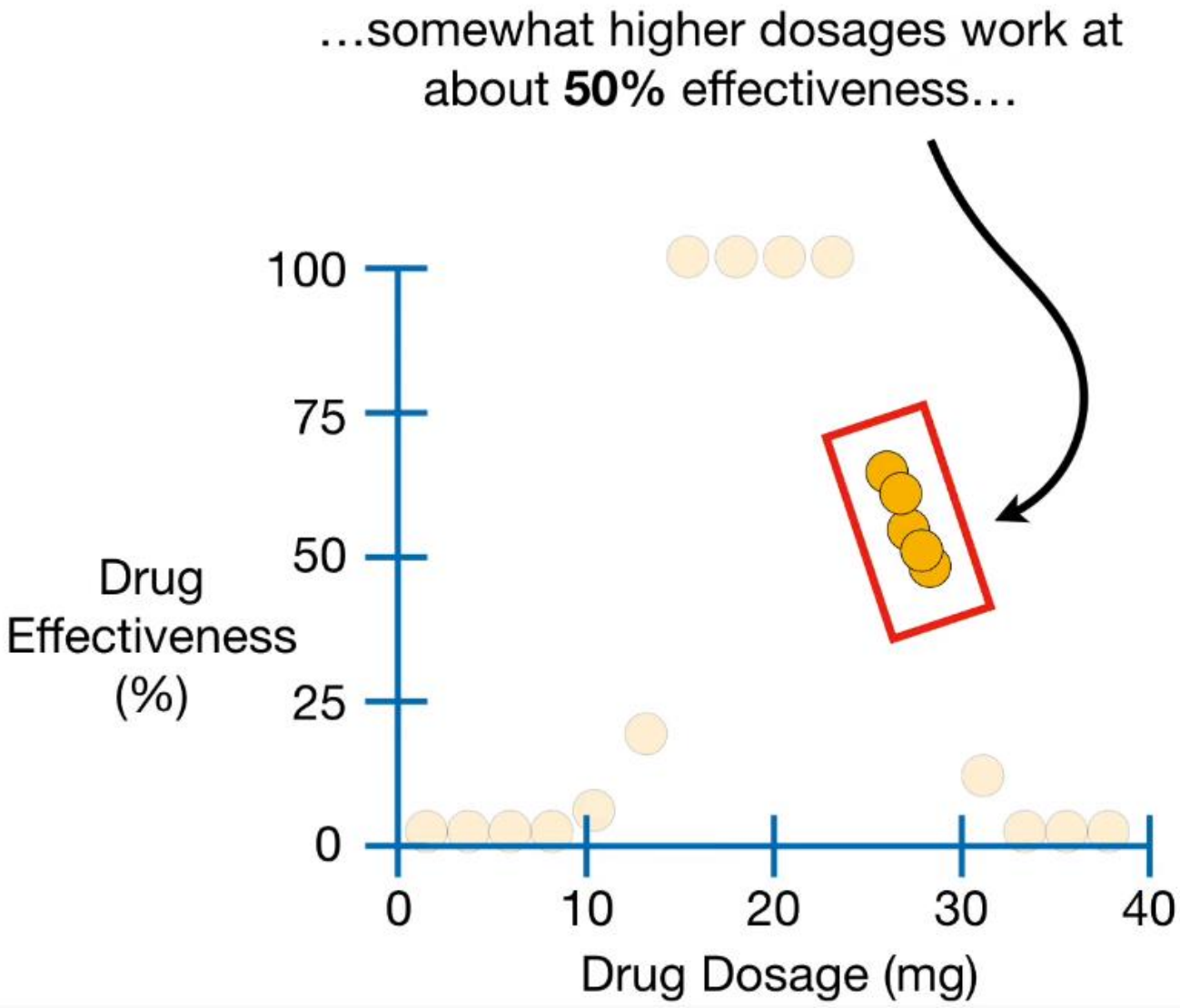
Decision Tree Regression



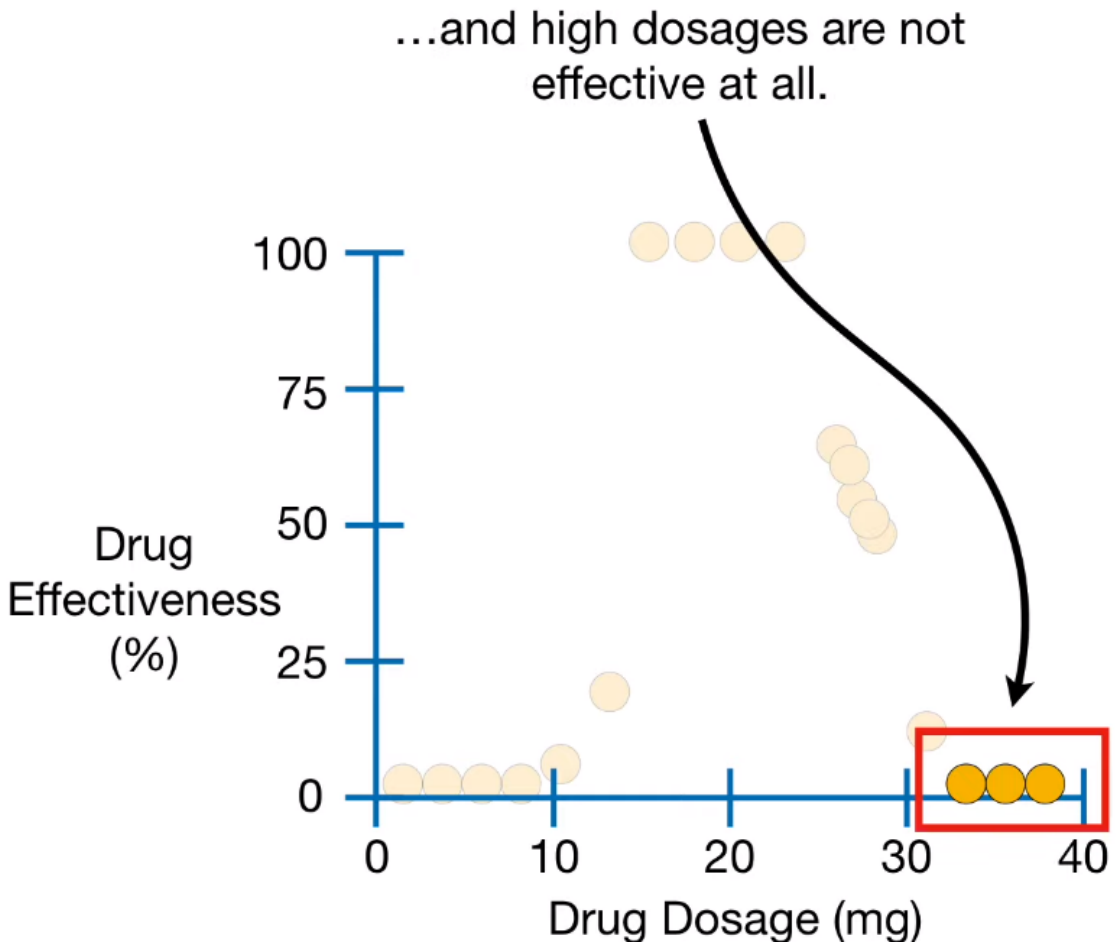
Decision Tree Regression



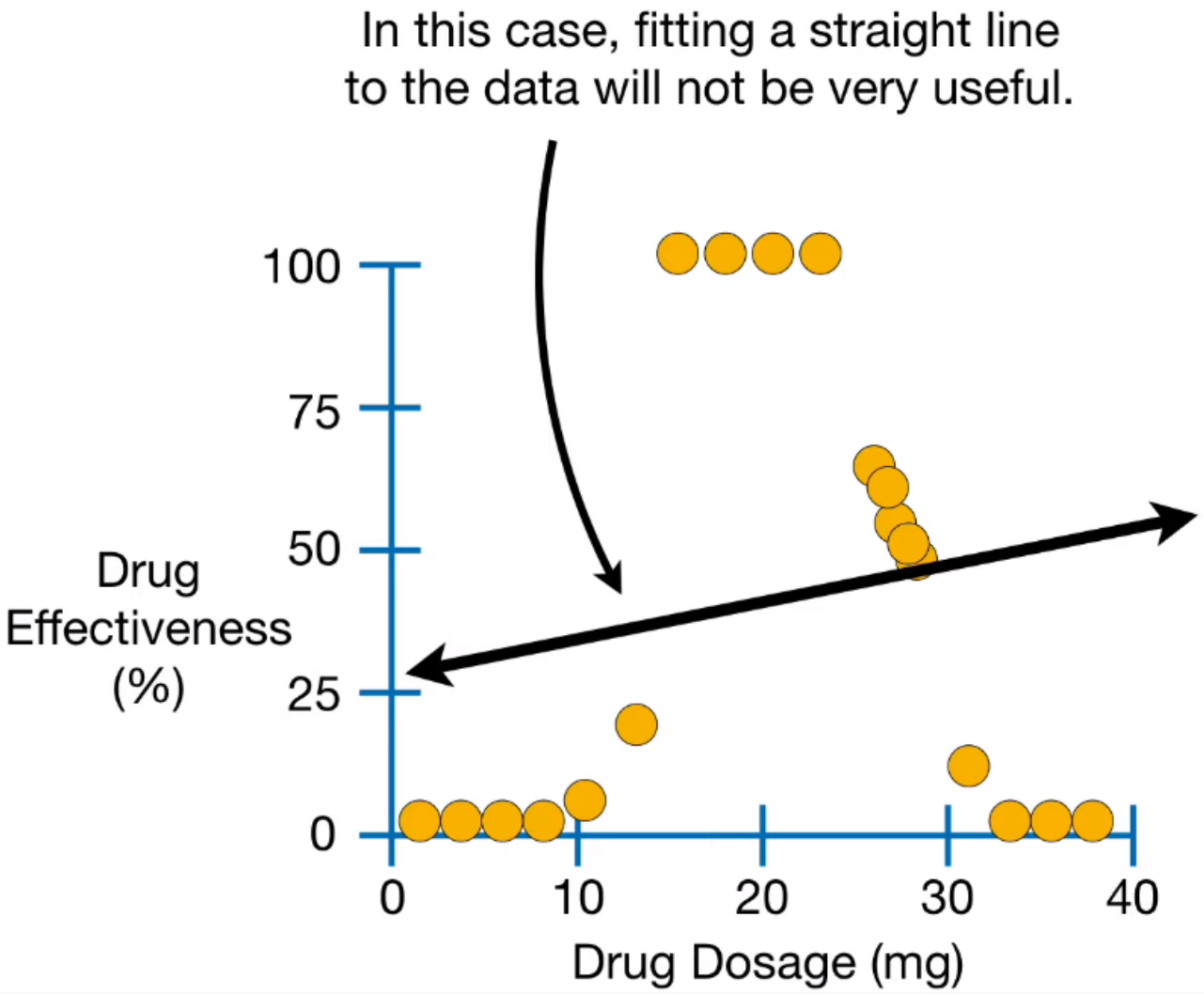
Decision Tree Regression



Decision Tree Regression

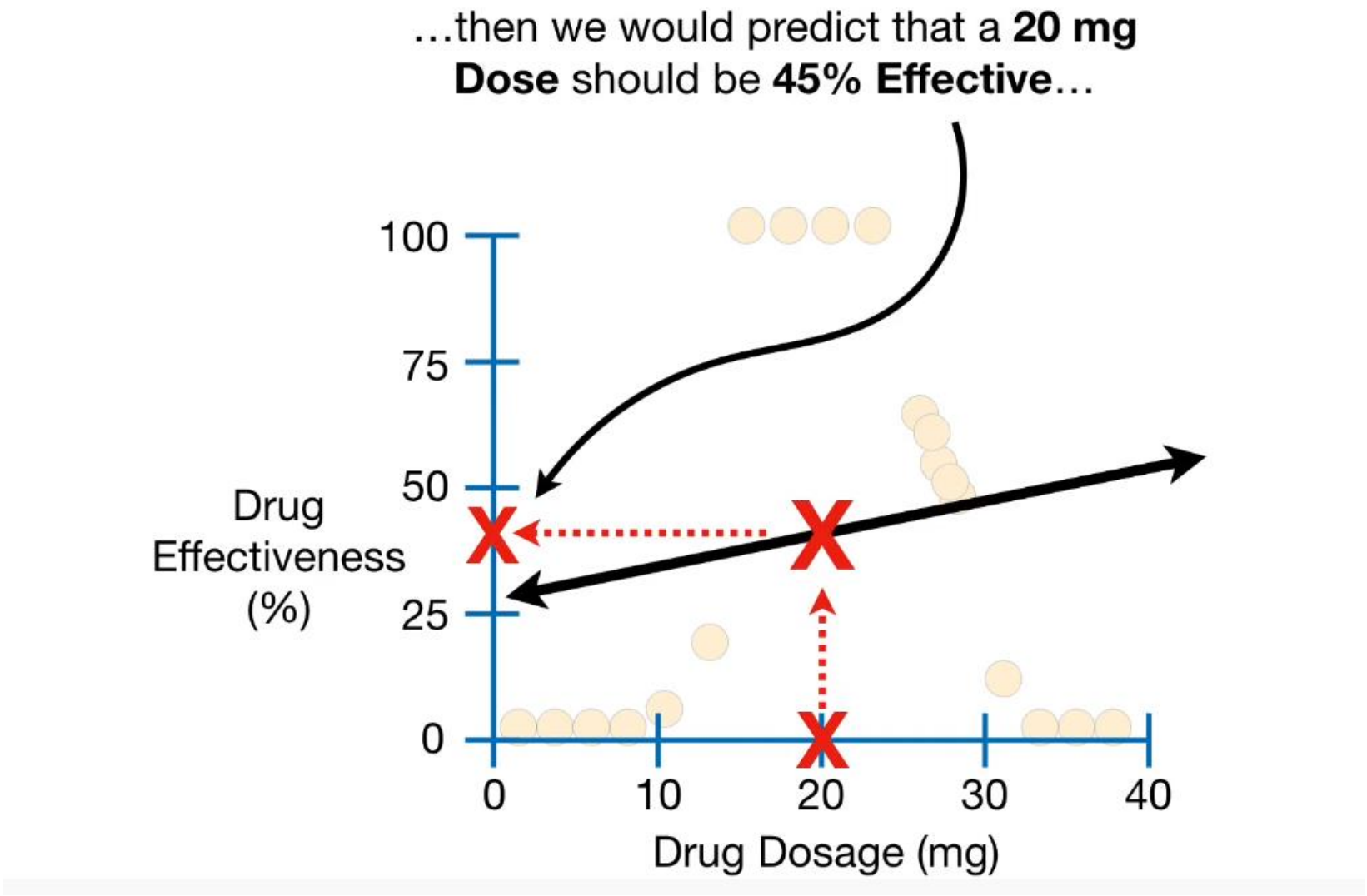


Decision Tree Regression

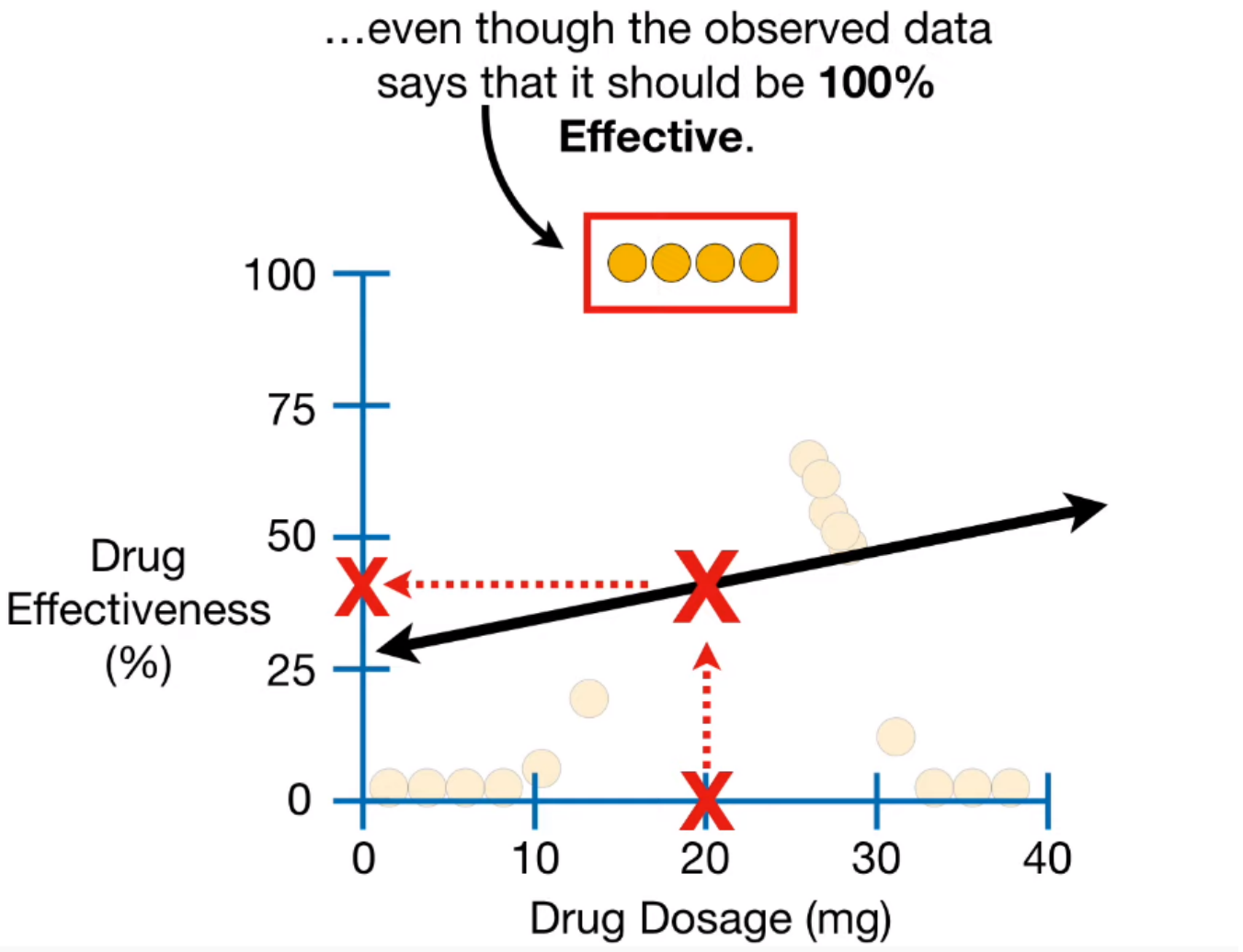


Decision Tree Regression

Decision Tree Regression



Decision Tree Regression

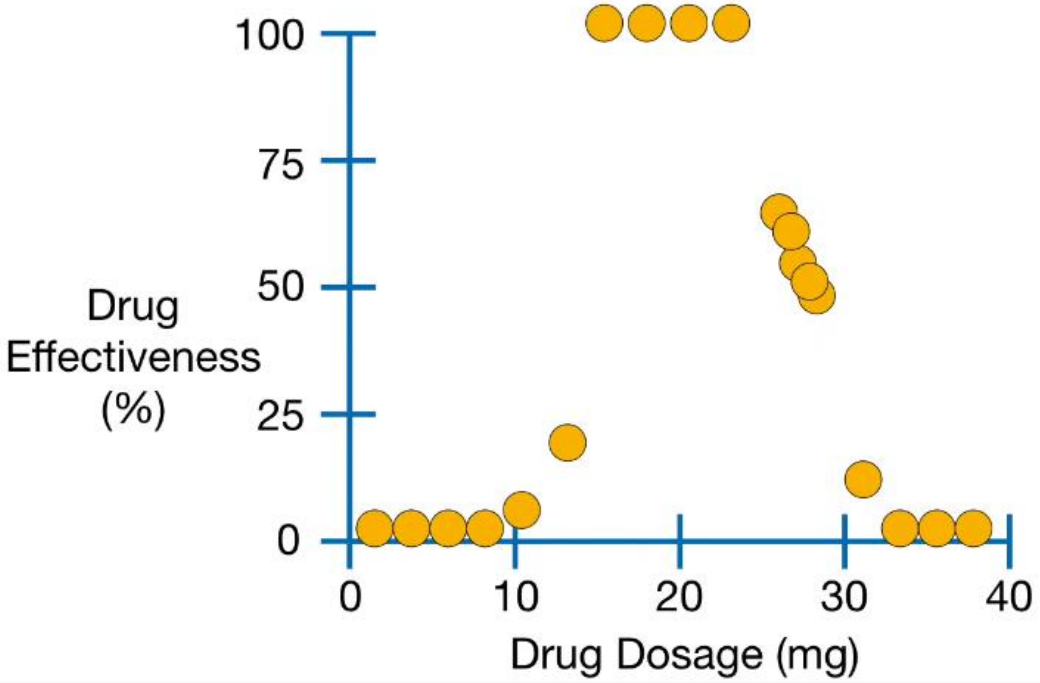


Decision Tree Regression

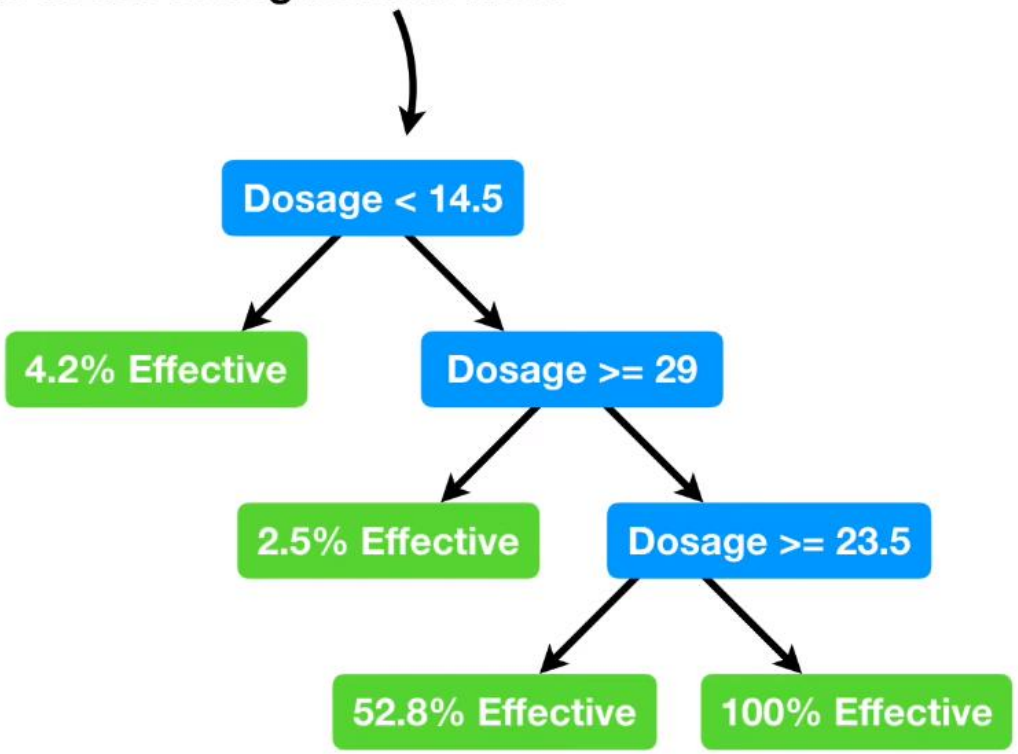
- What do we do?
  - Use SVM?
    - Very expensive for large datasets
  - Linear Regression?
    - Inaccurate
  - Something else?
    - Lets see…

Decision Tree Regression

However, what if the data looked like this?

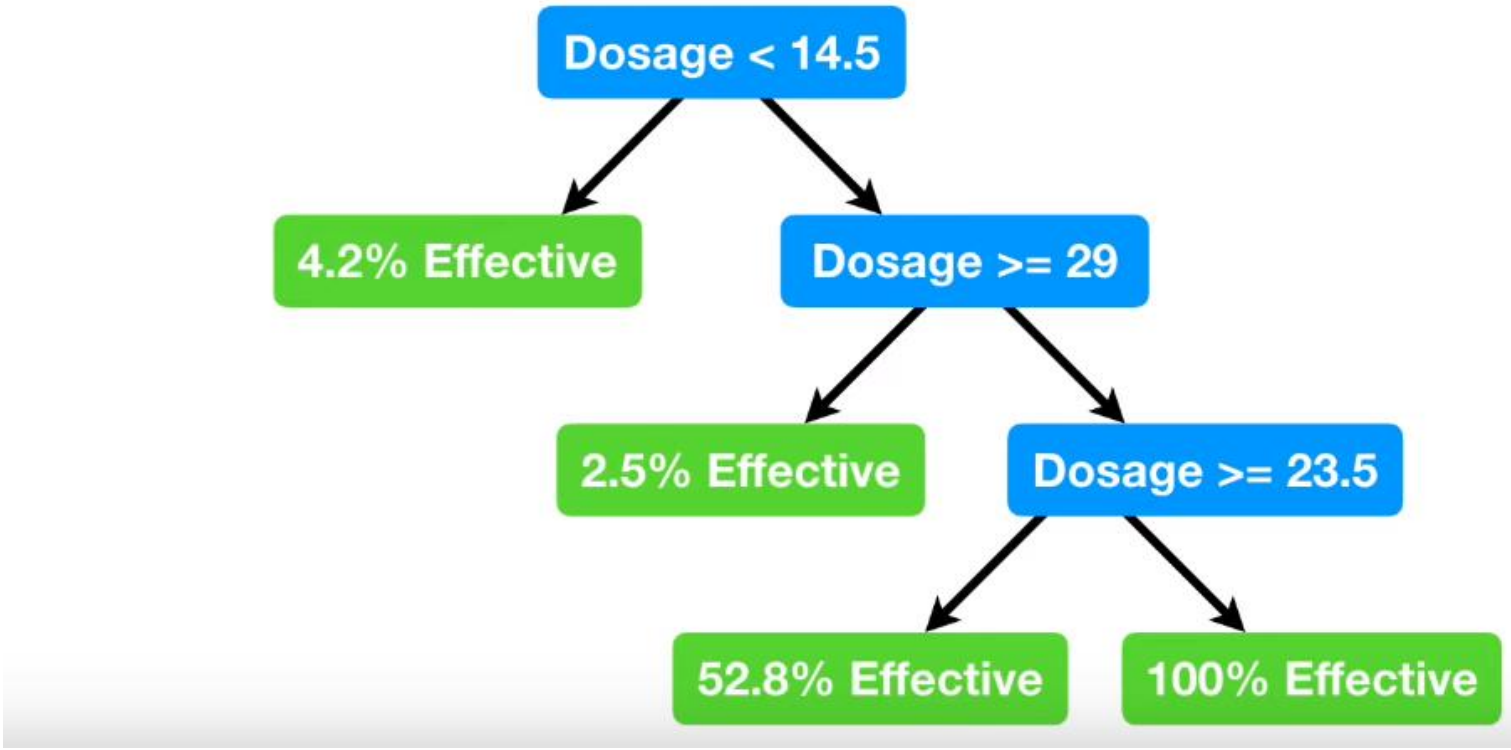


One option is to use a **Regression Tree.**

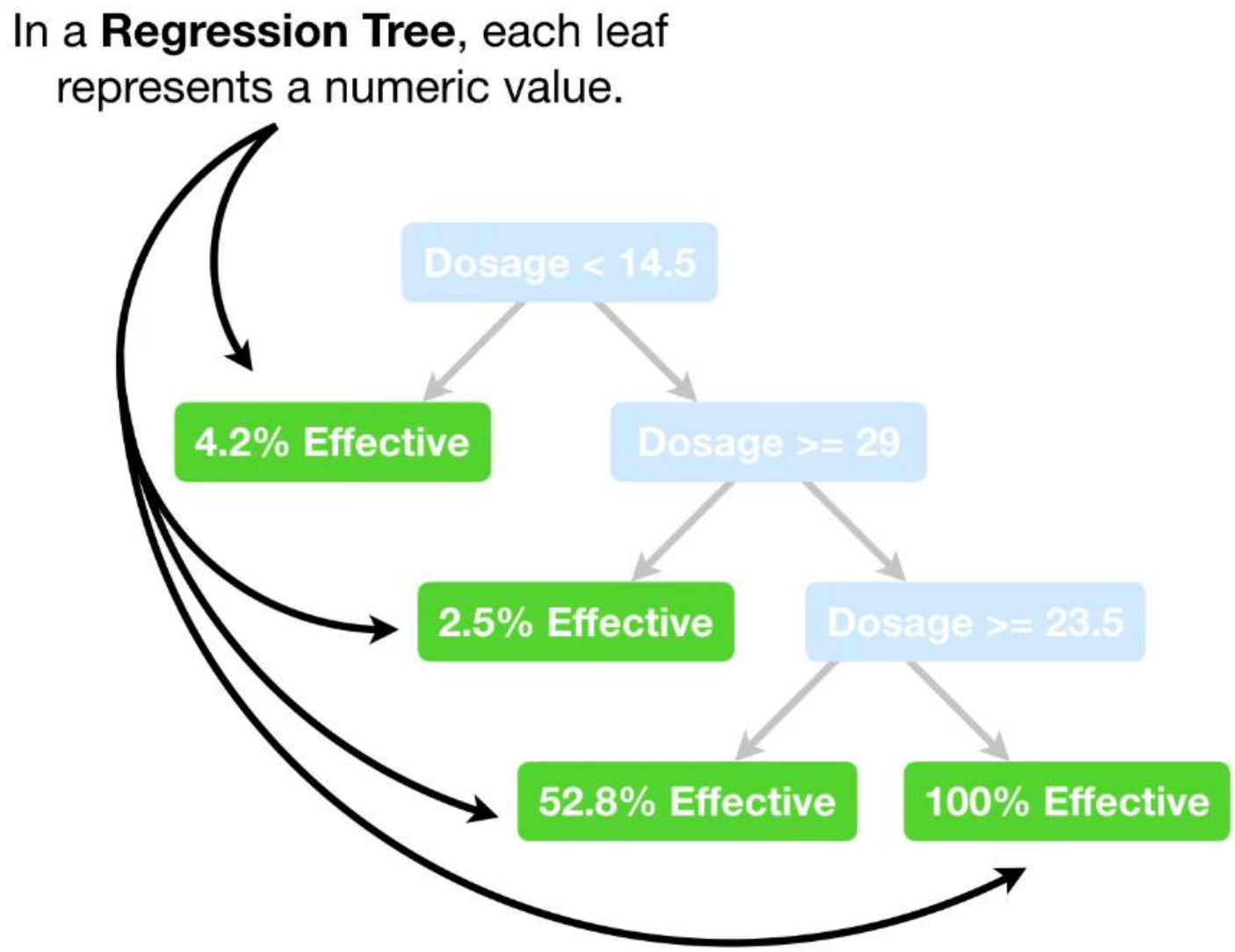


Decision Tree Regression

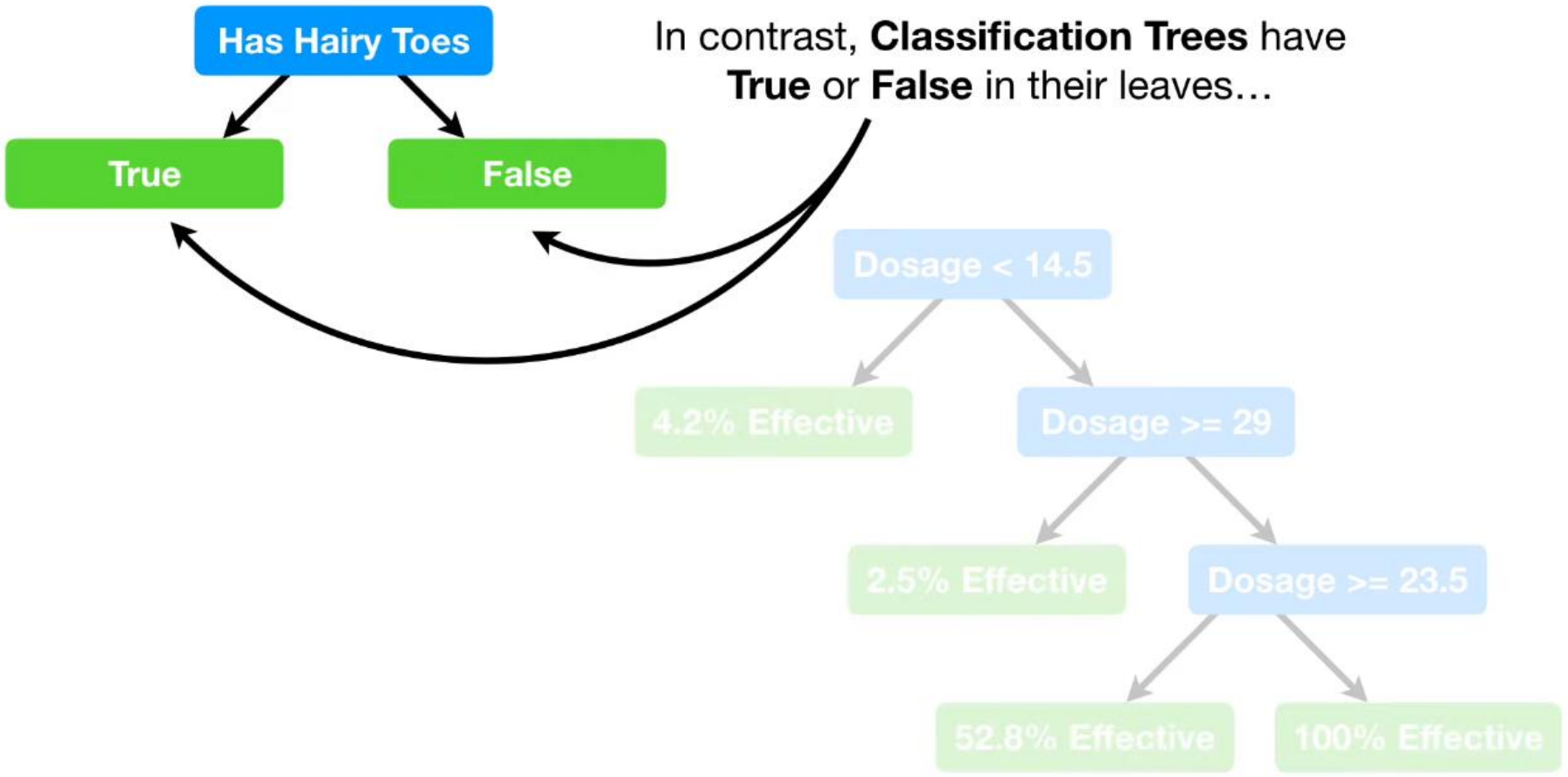
Regression Trees are a type of **Decision Tree.**



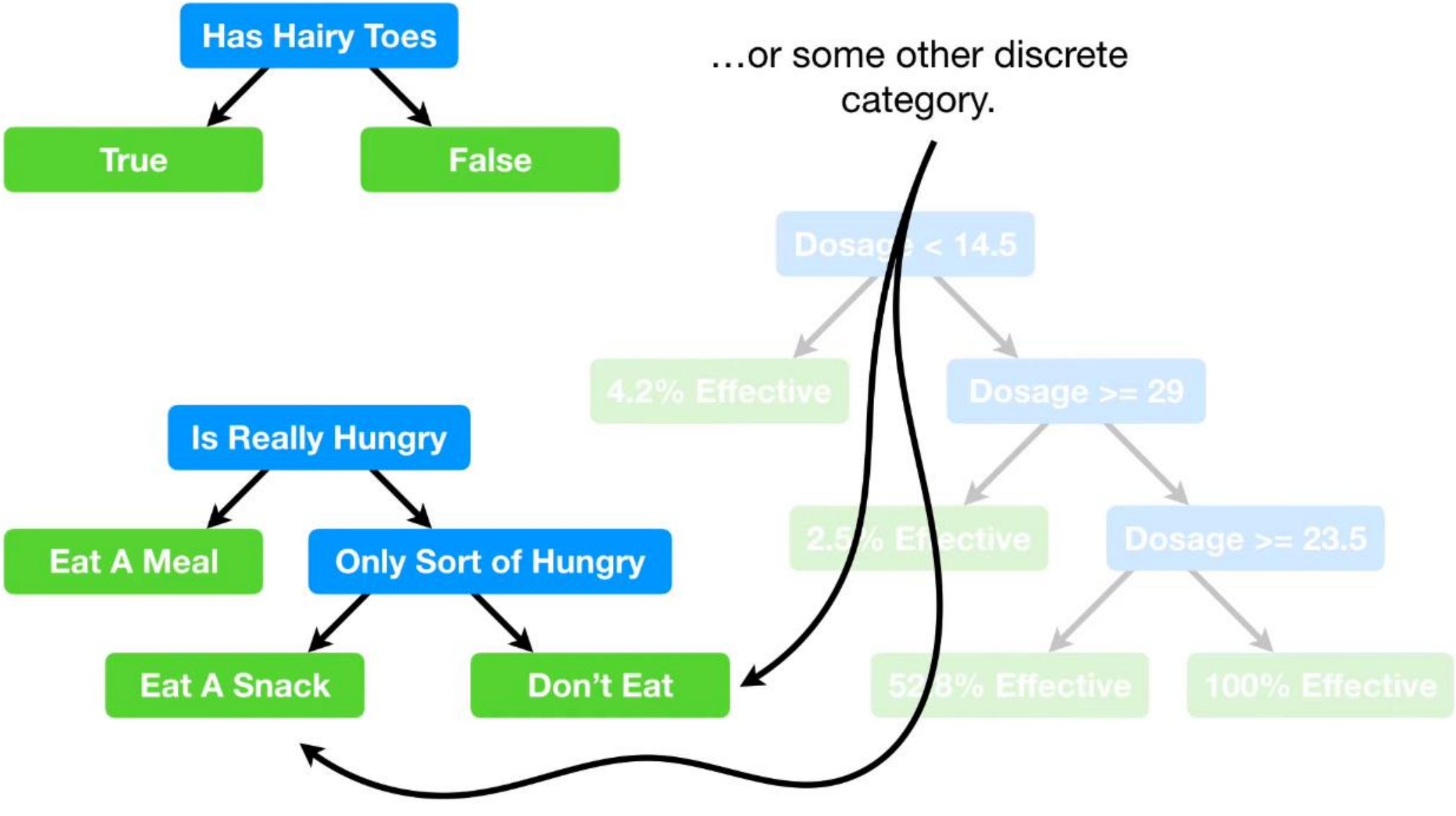
Decision Tree Regression



Decision Tree Regression

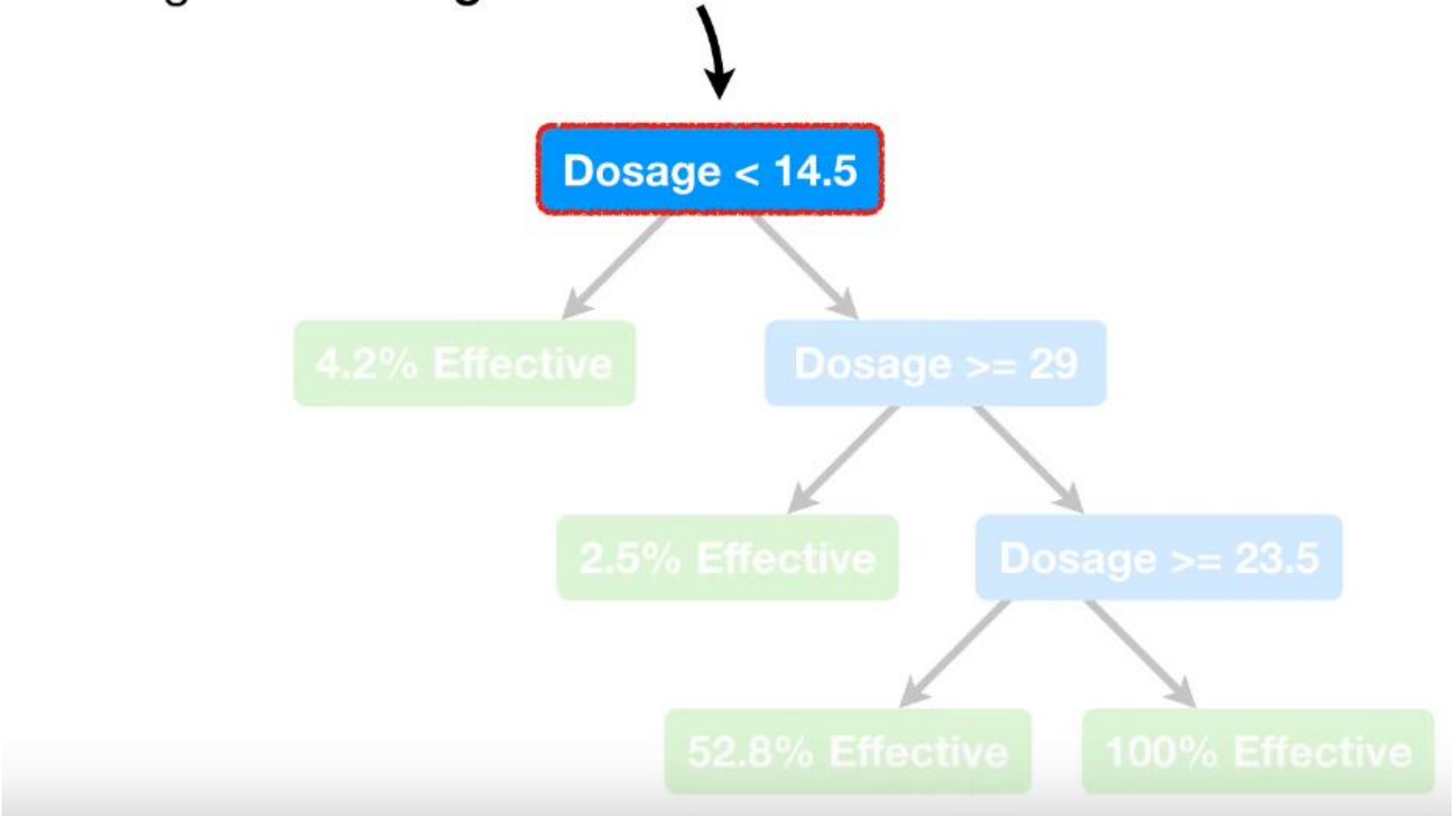


Decision Tree Regression



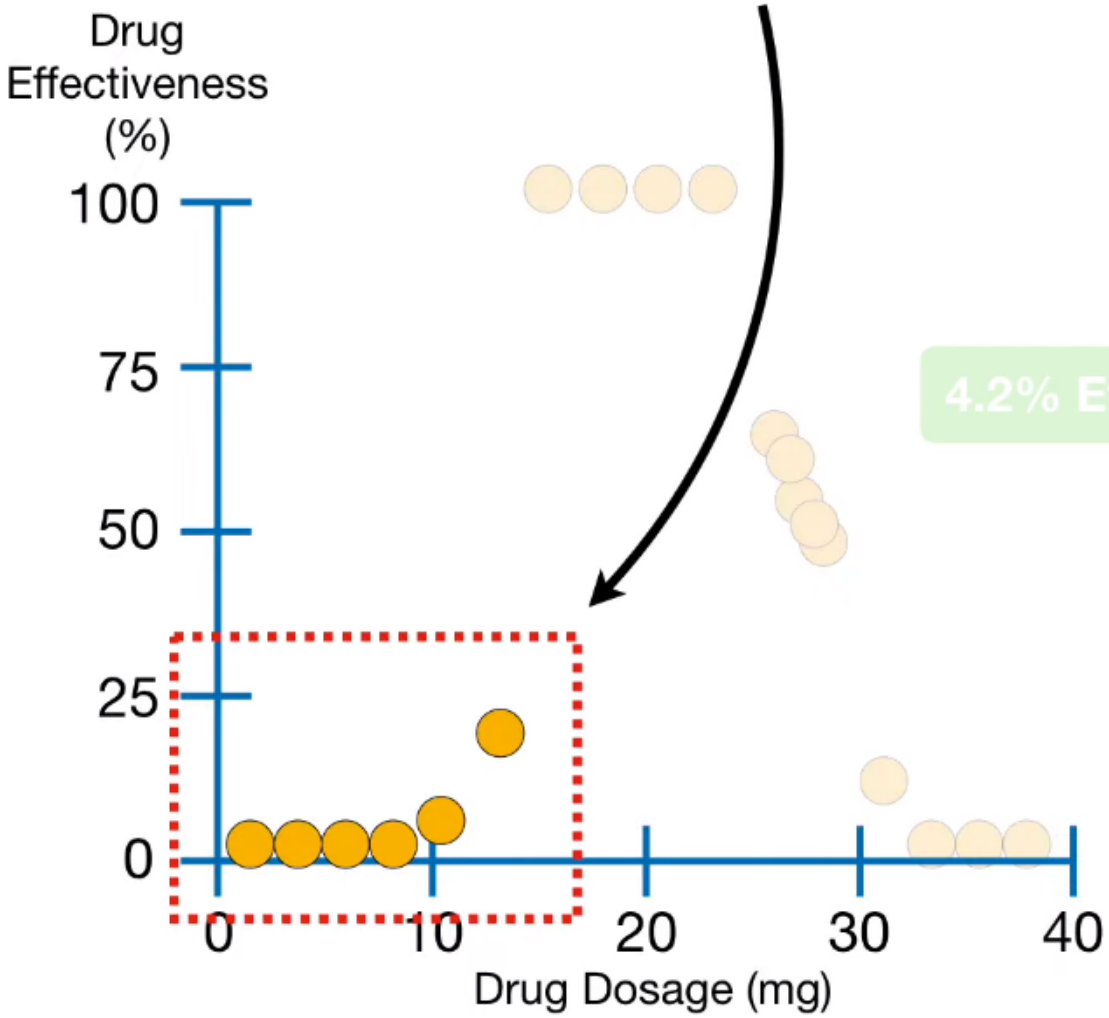
Decision Tree Regression

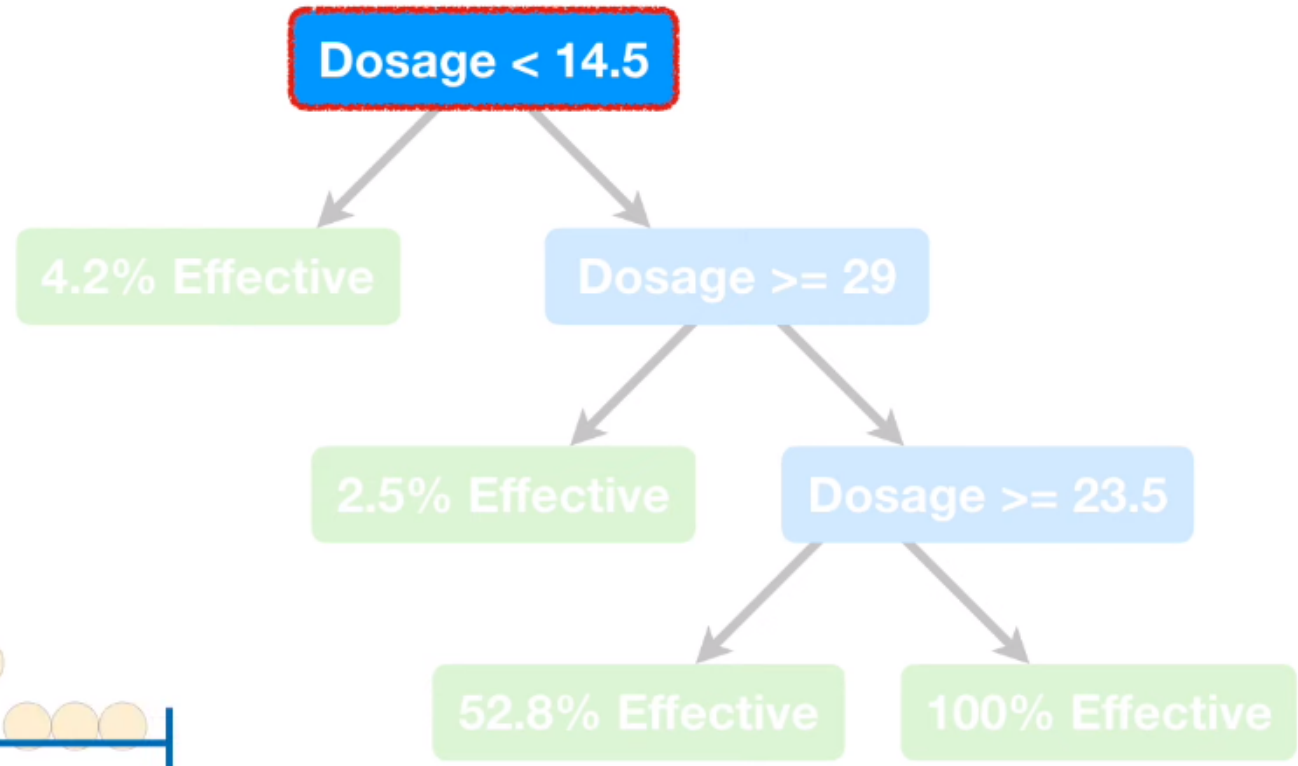
With *this* **Regression Tree**, we start by asking if the **Dosage** is less than **14.5**.



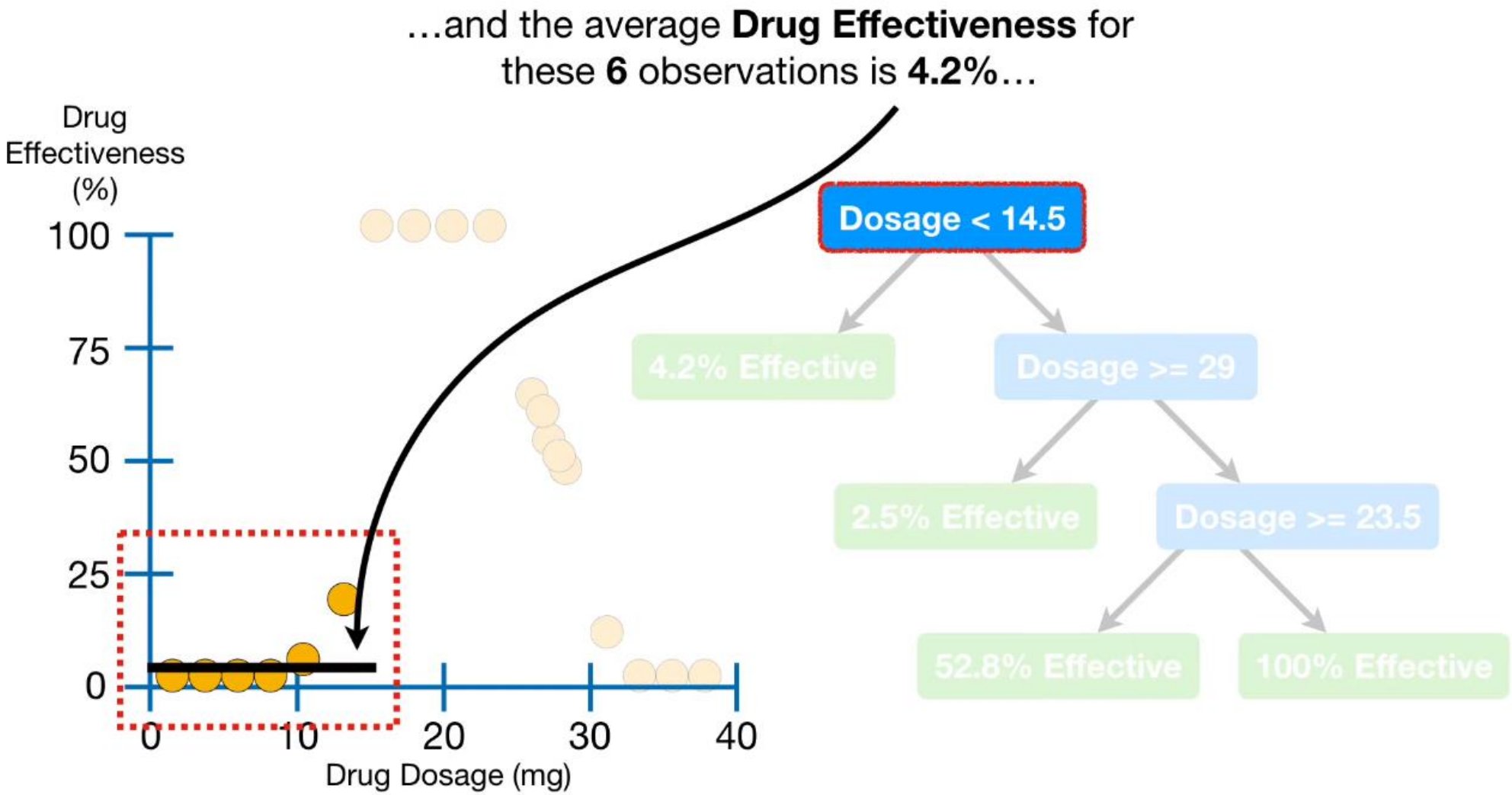
Decision Tree Regression

…if so, then we are talking about these **6** observations in the training data…



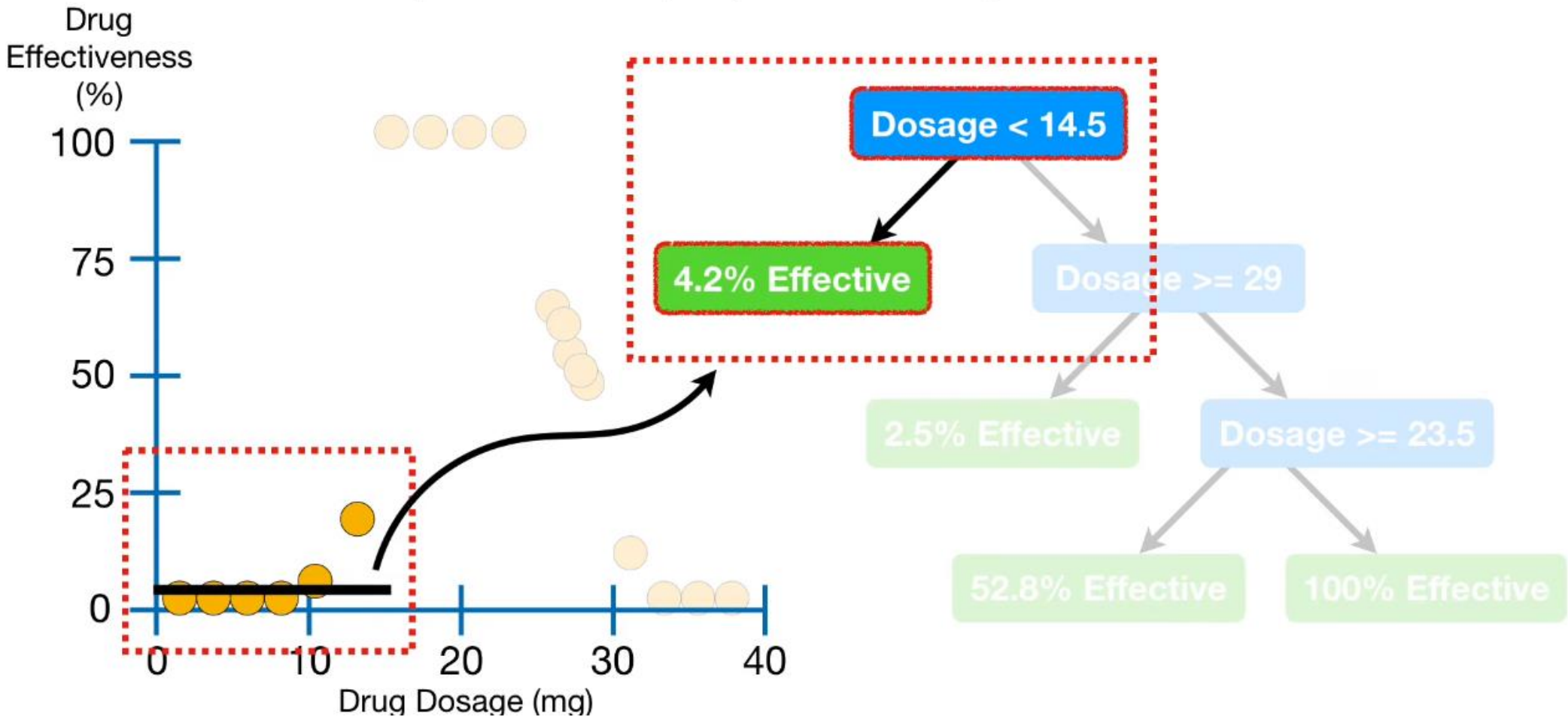


Decision Tree Regression

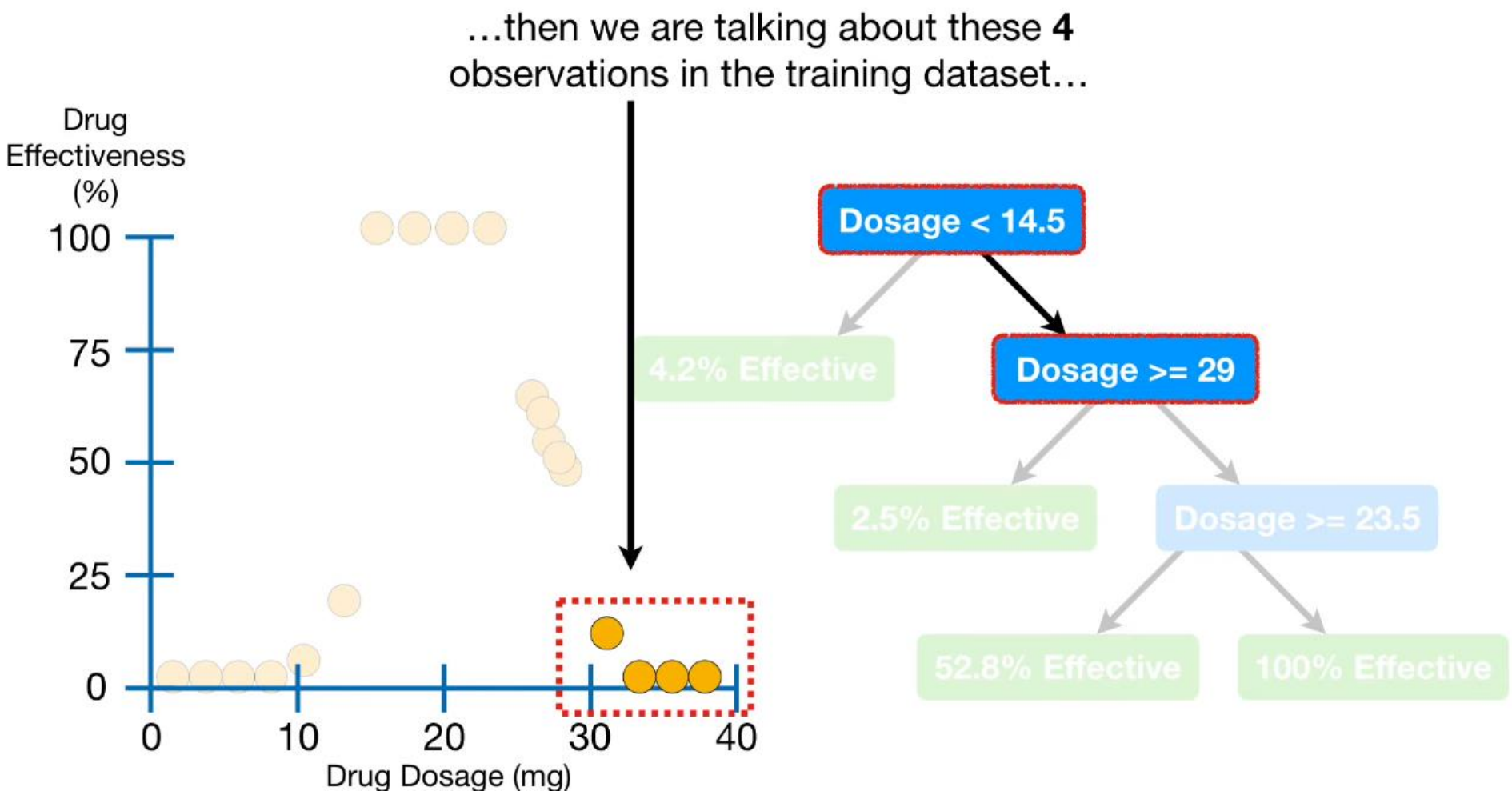


Decision Tree Regression

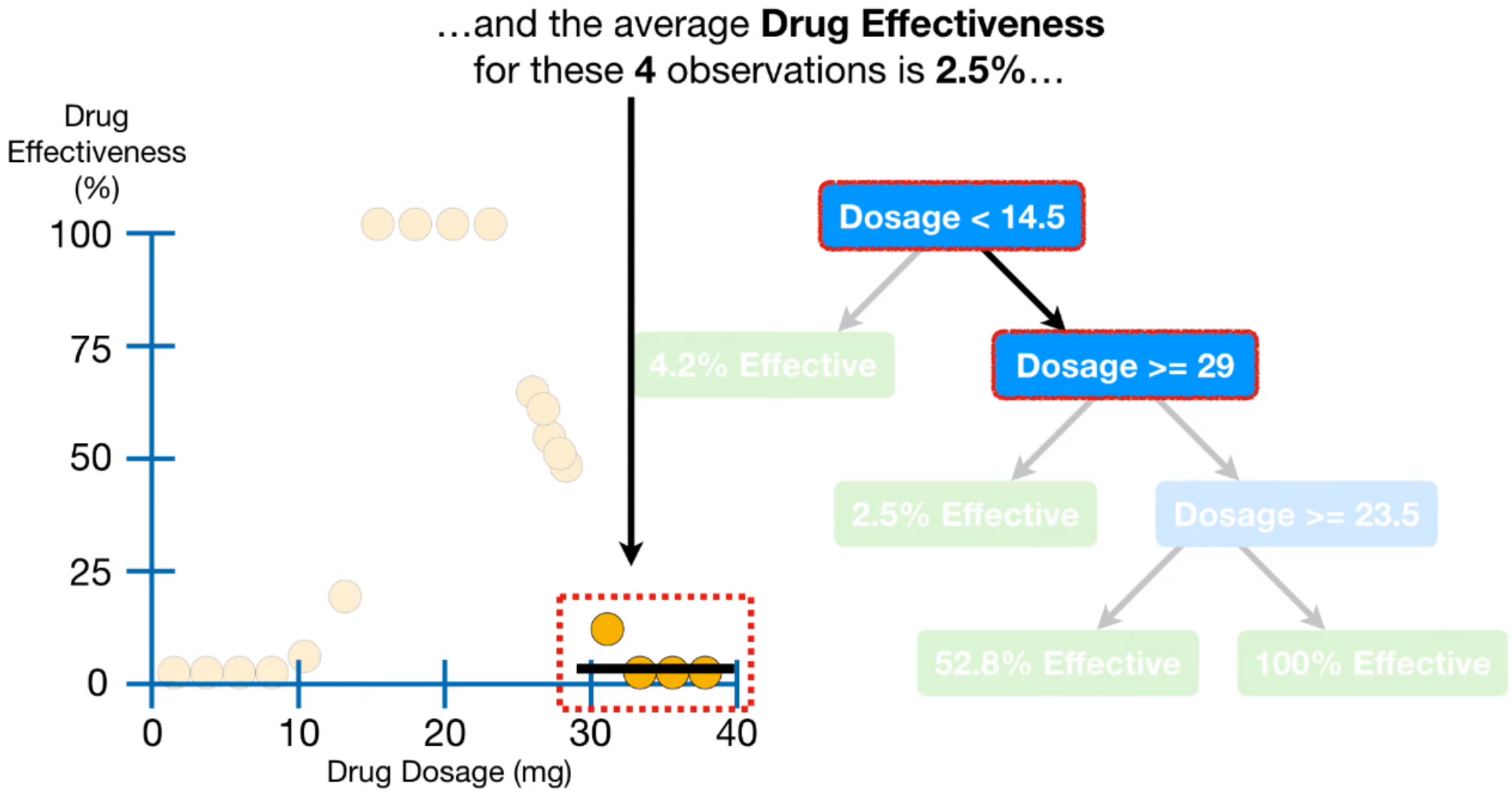
…so the tree uses the average value, **4.2%**, as its prediction for people with **Dosages < 14.5**.



Decision Tree Regression

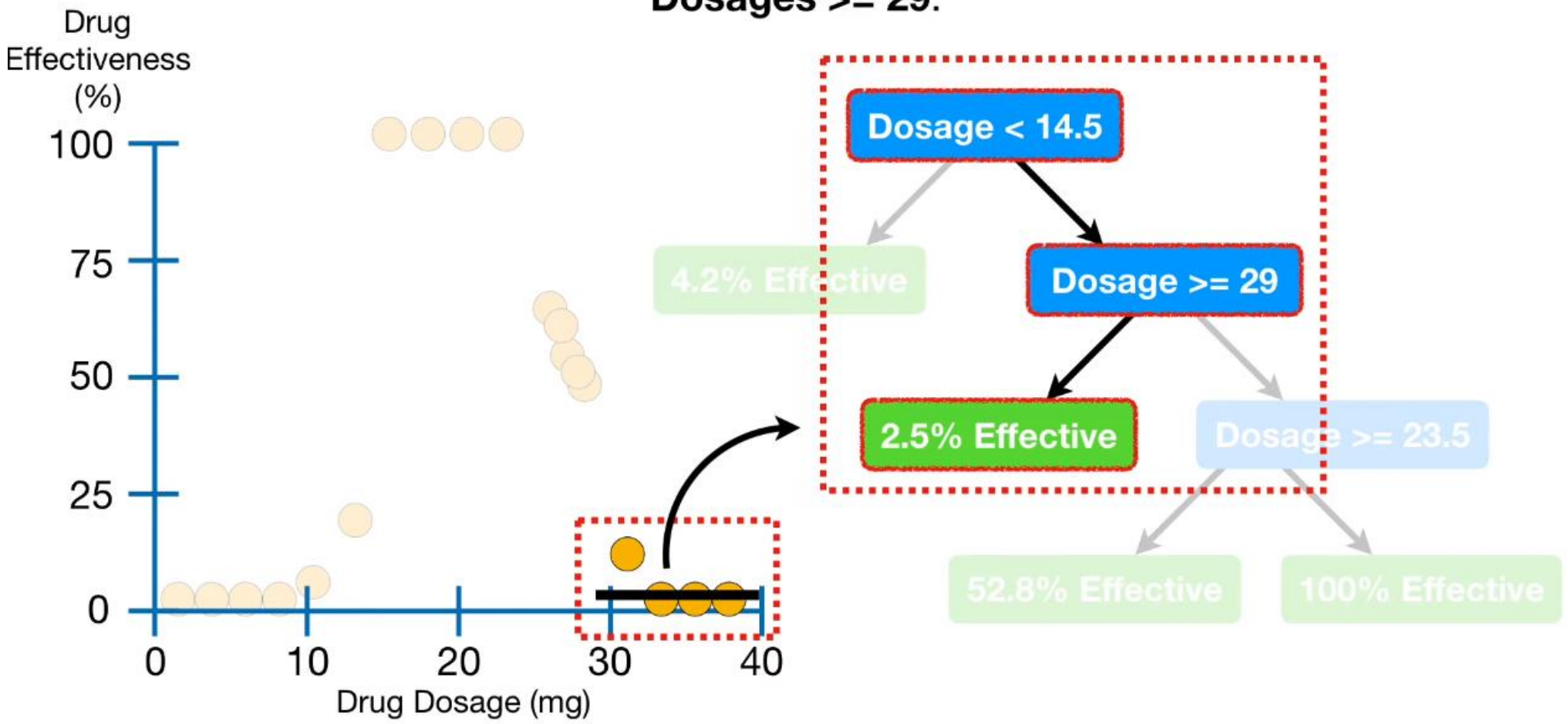


Decision Tree Regression

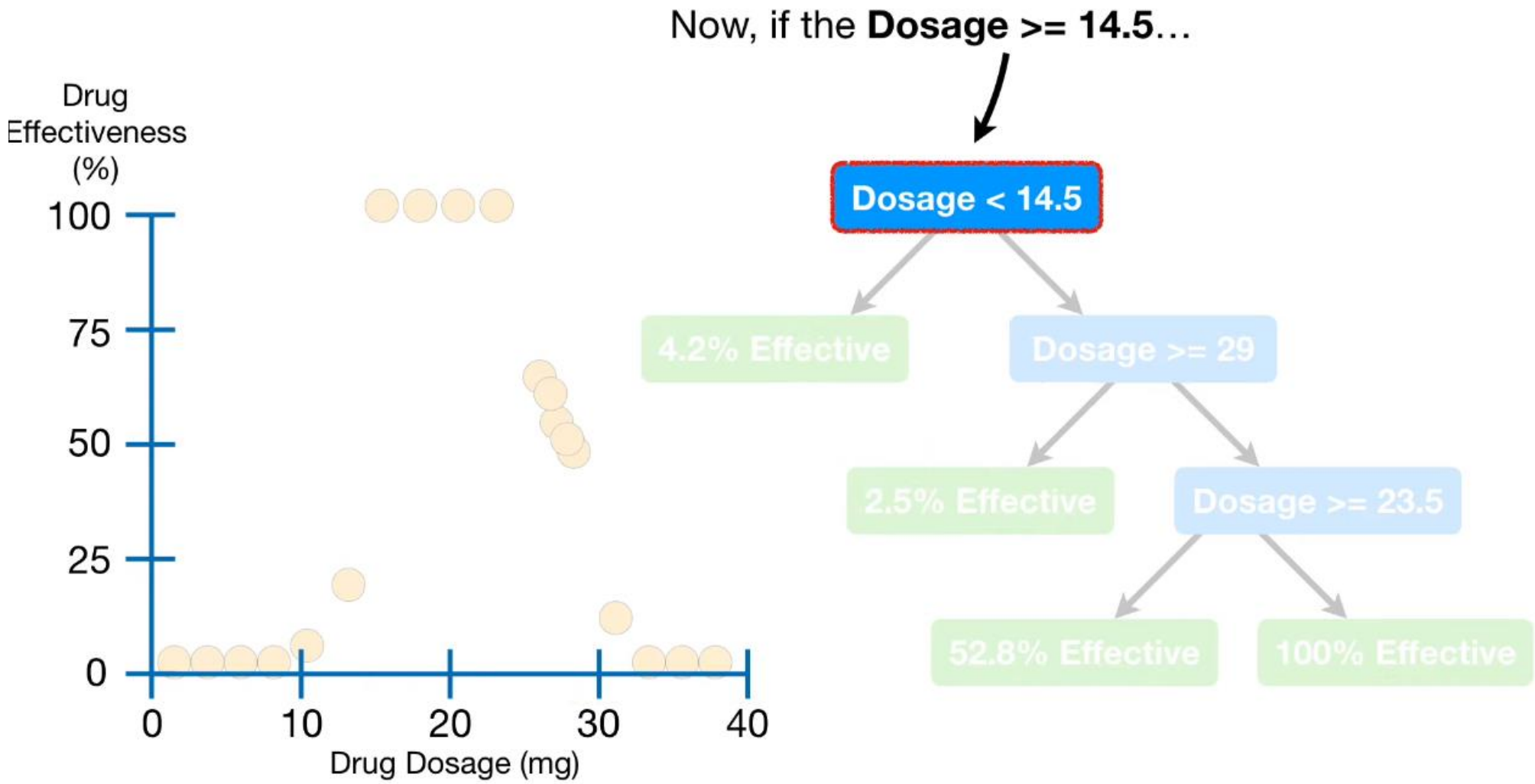


Decision Tree Regression

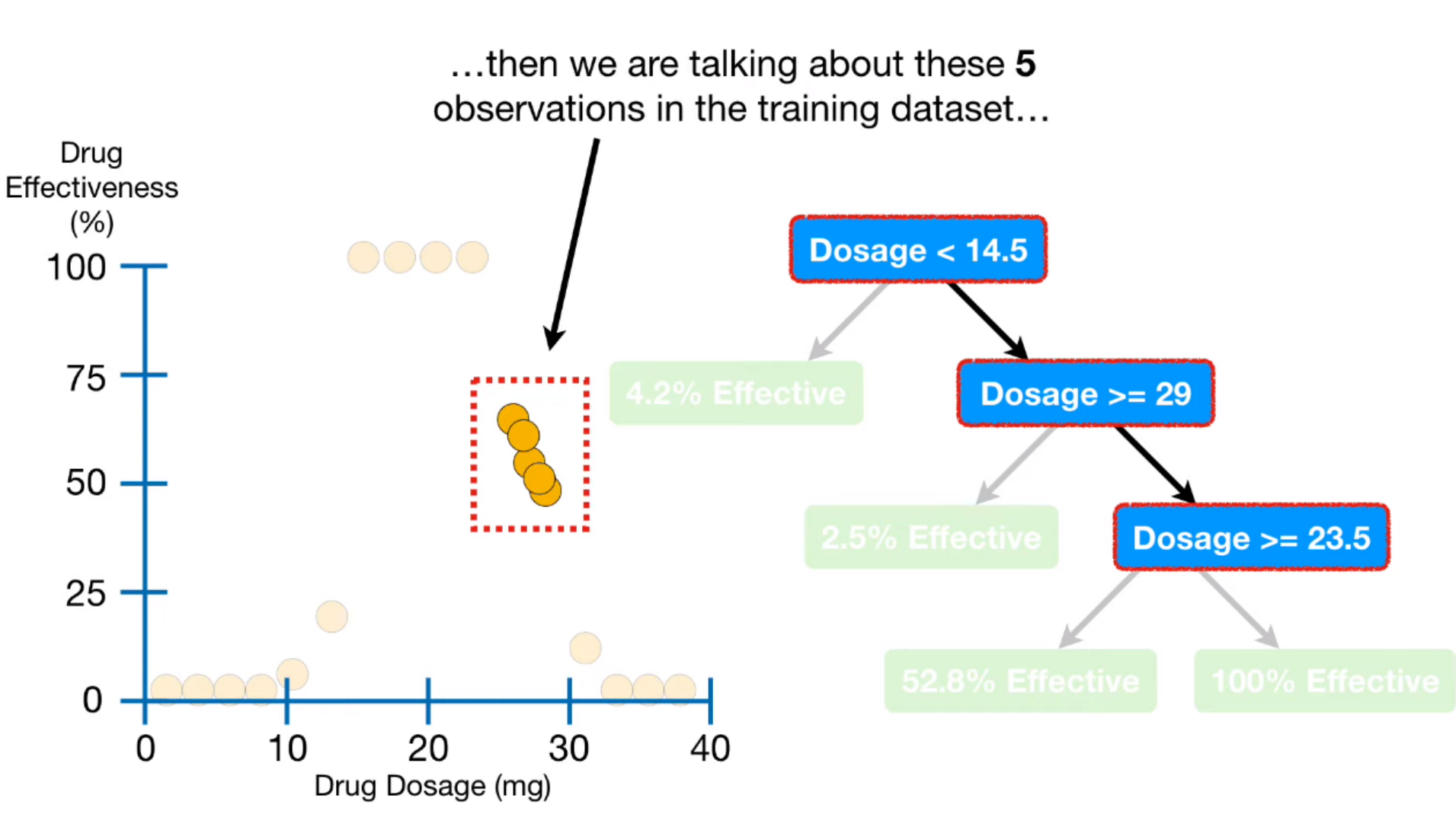
…so the tree uses the average value, **2.5%**, as its prediction for people with **Dosages >= 29**.



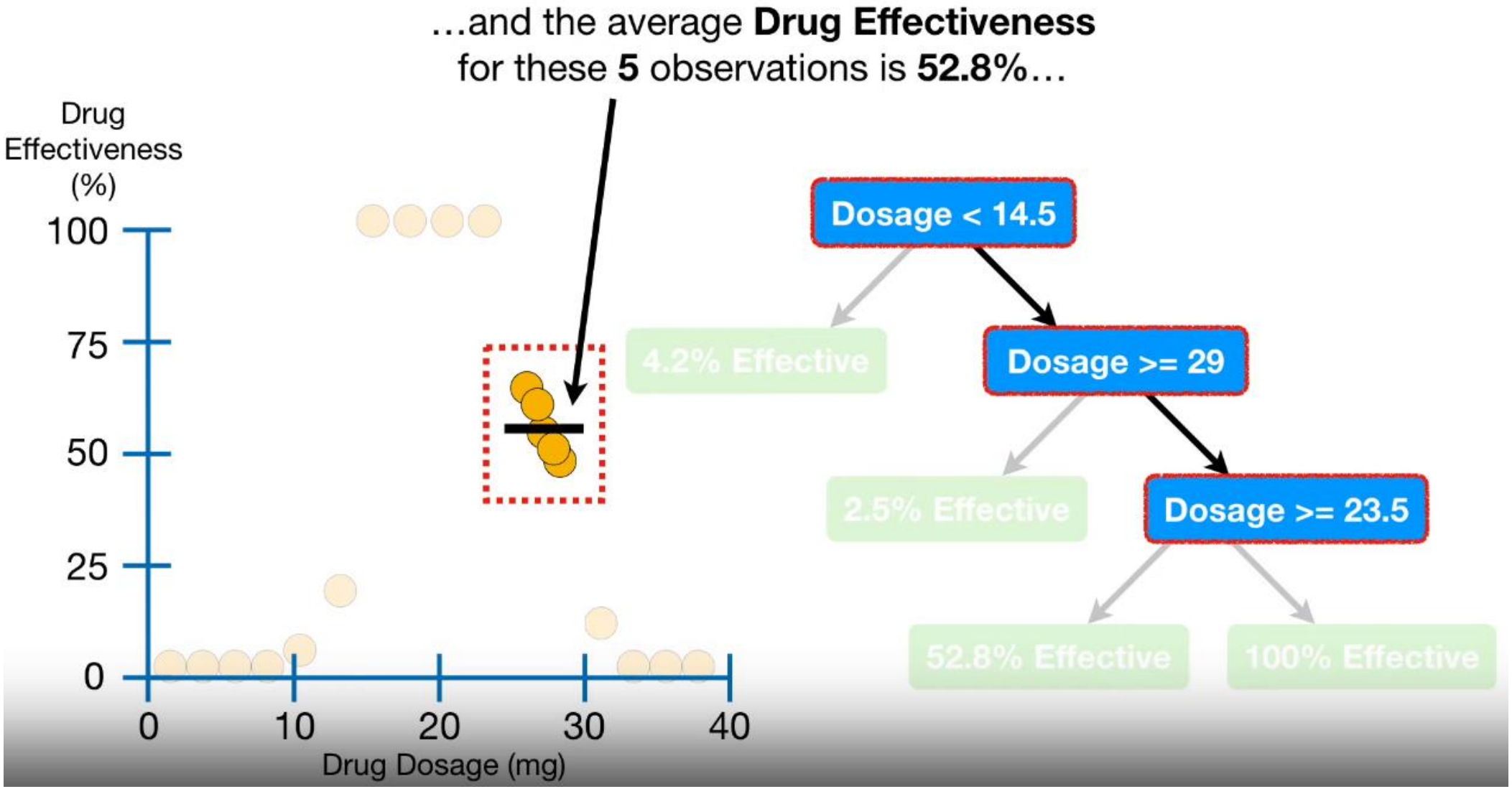
Decision Tree Regression



Decision Tree Regression

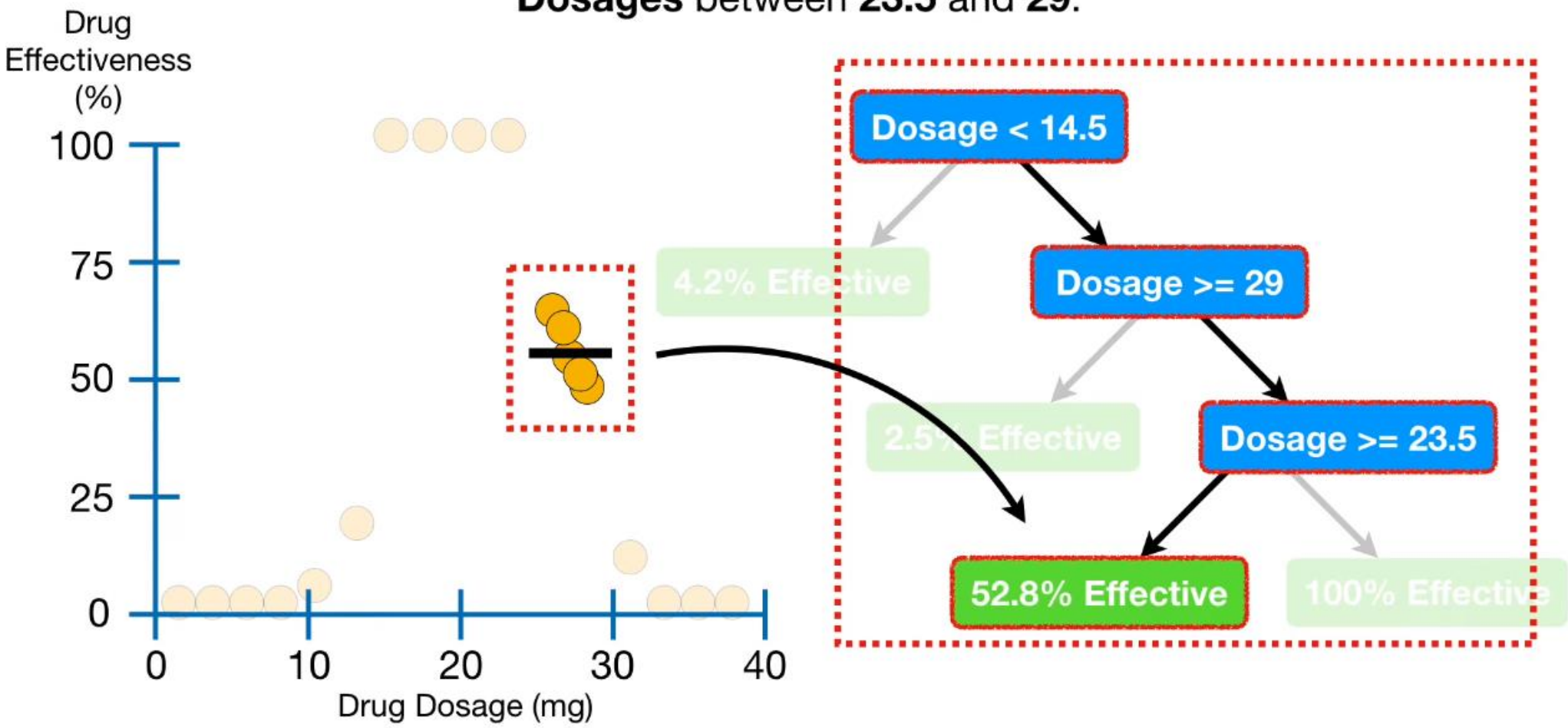


Decision Tree Regression

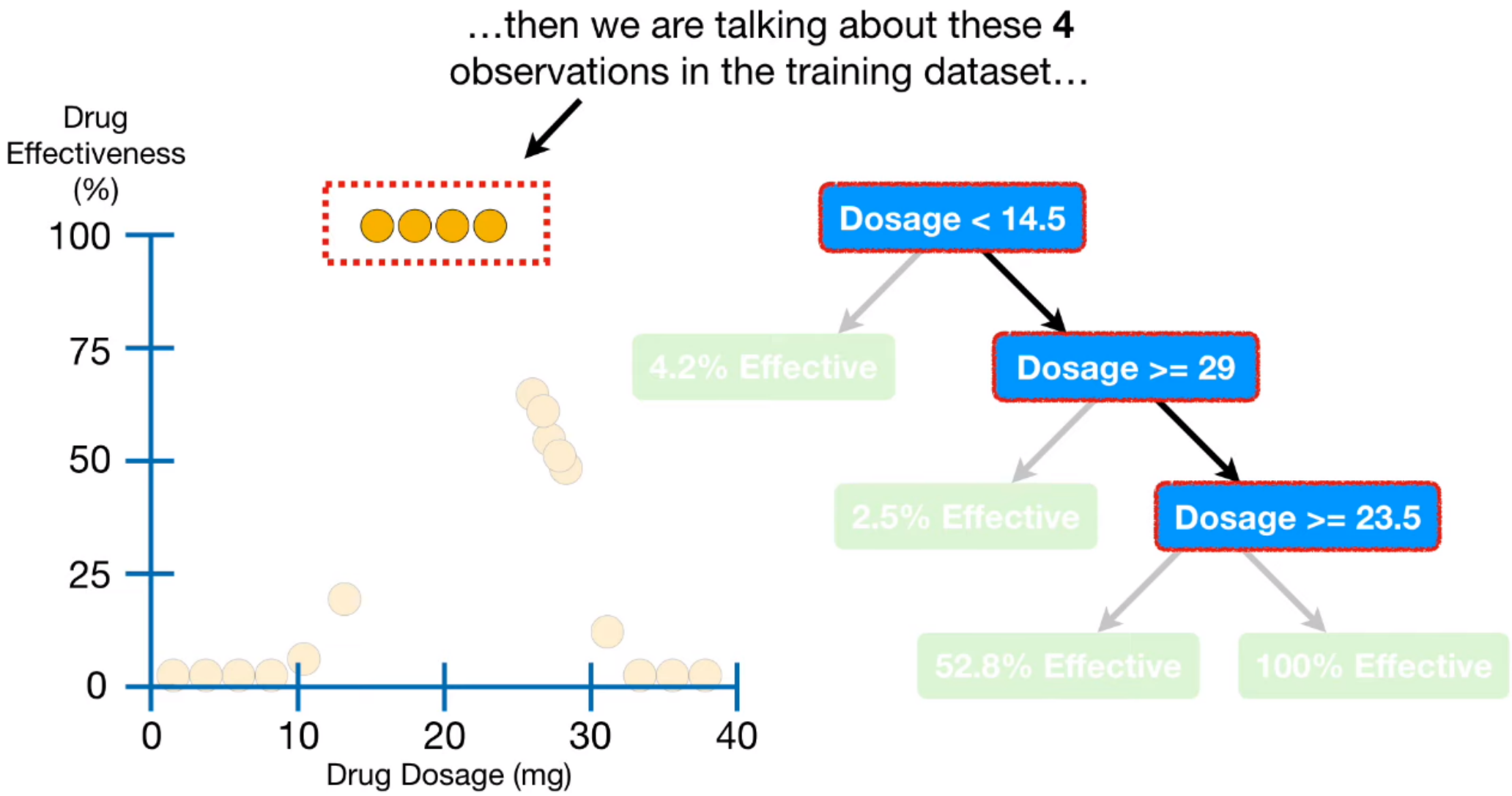


Decision Tree Regression

…so the tree uses the average value, **52.8%**, as its prediction for people with **Dosages** between **23.5** and **29**.

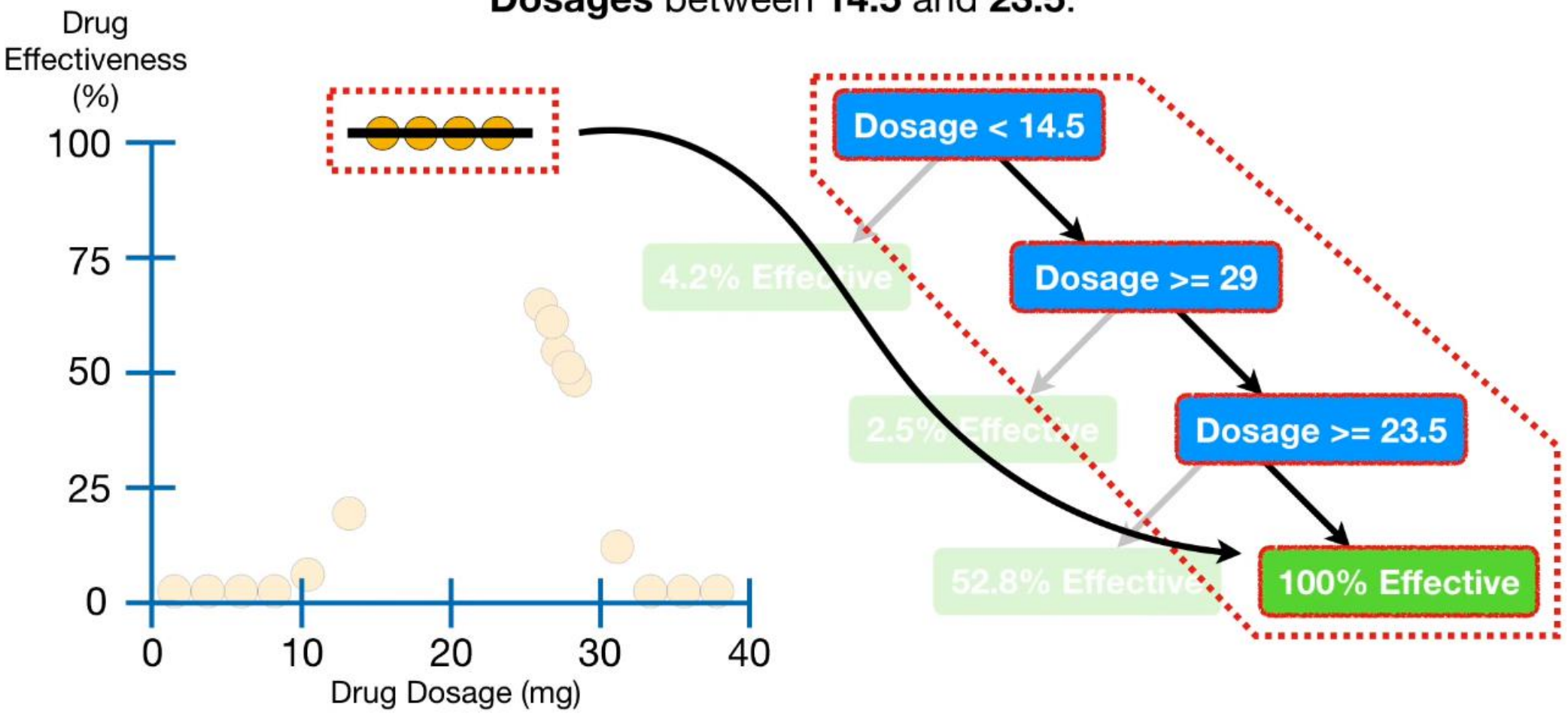


Decision Tree Regression



Decision Tree Regression

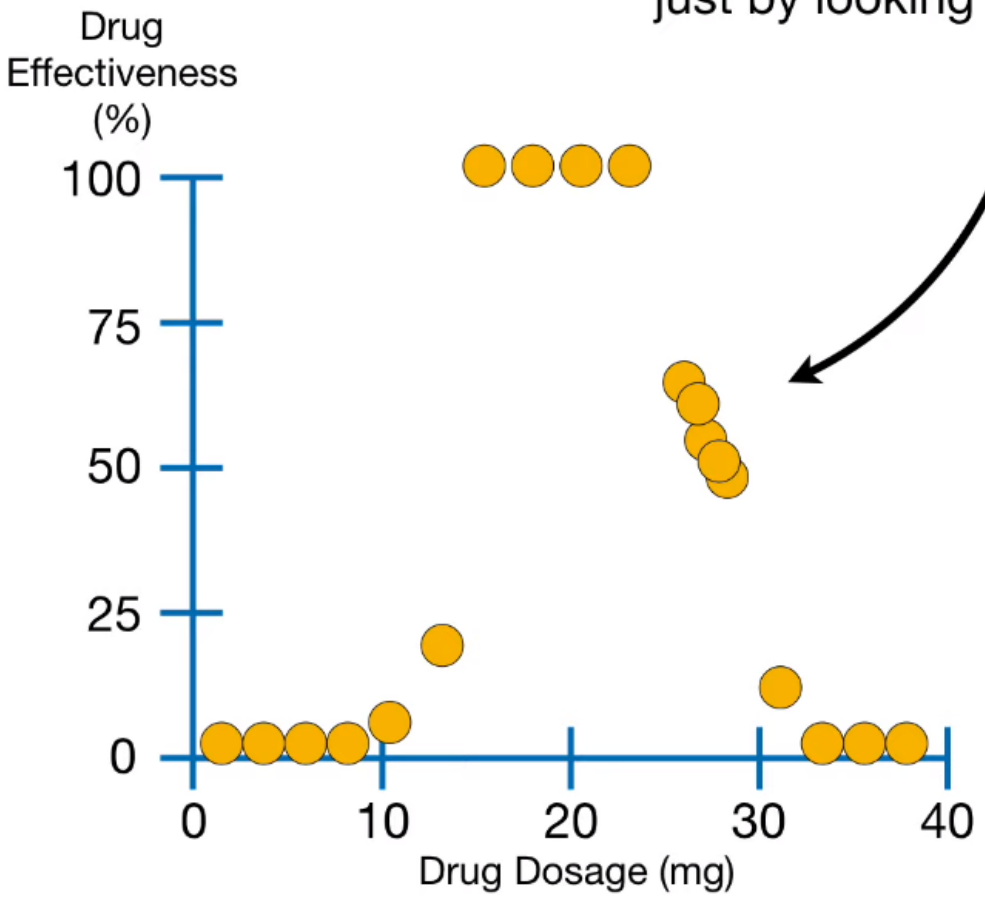
…so the tree uses the average value, **100%**, as its prediction for people with **Dosages** between **14.5** and **23.5**.



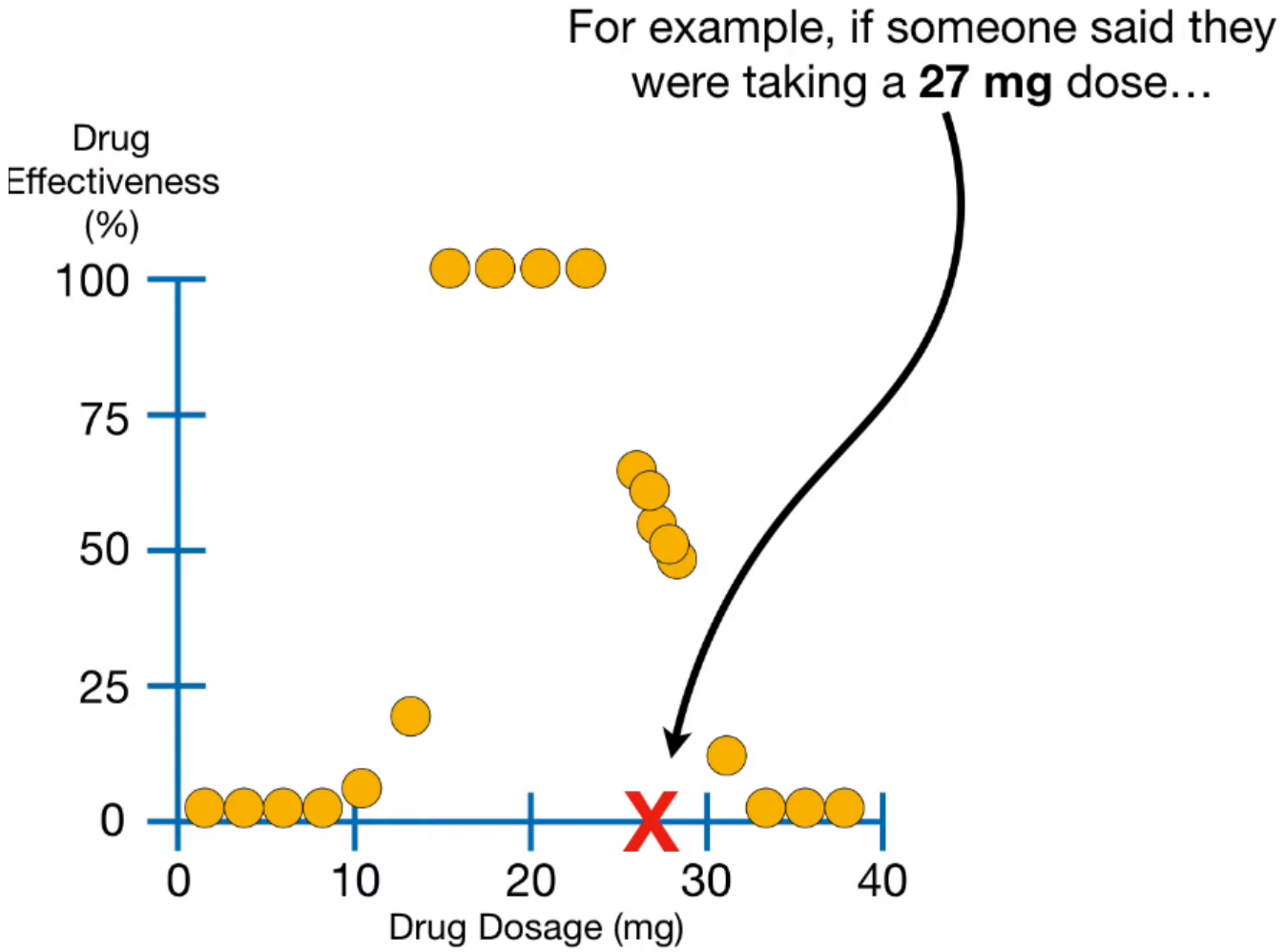
Decision Tree Regression

Decision Tree Regression

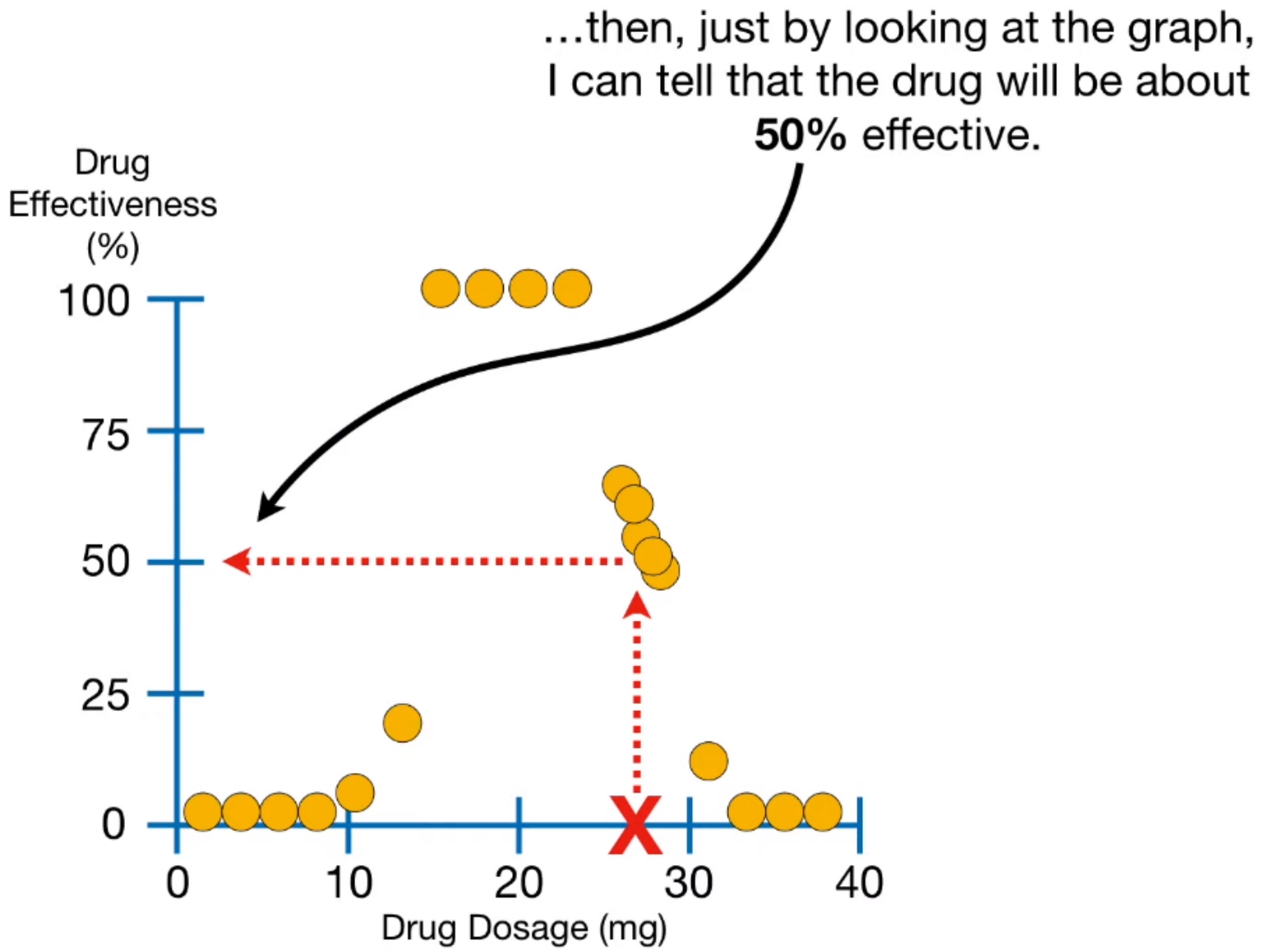
At this point you might be thinking, “The **Regression Tree** is cool, but I can also predict **Drug Effectiveness** just by looking at the graph…”



Decision Tree Regression

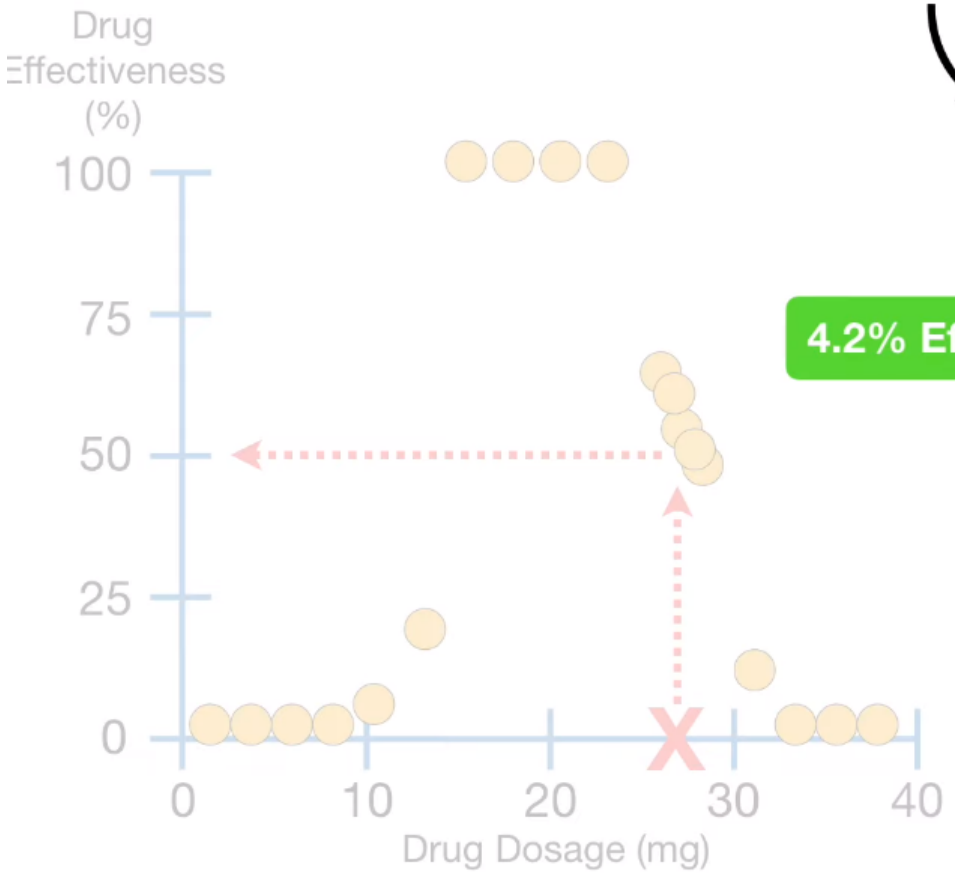


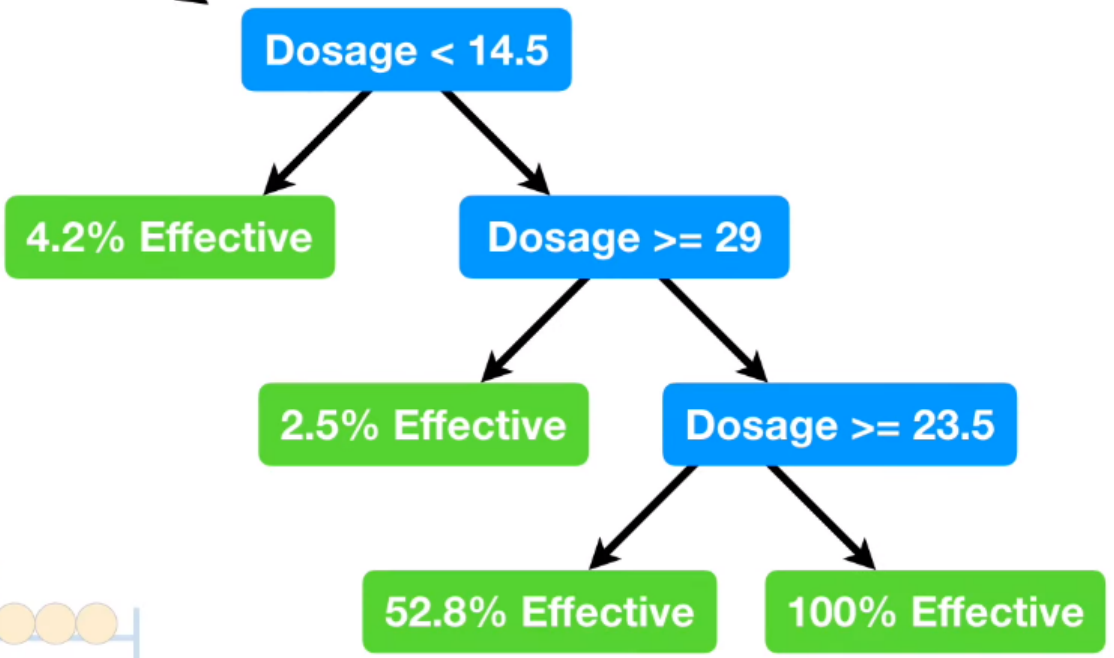
Decision Tree Regression



Decision Tree Regression

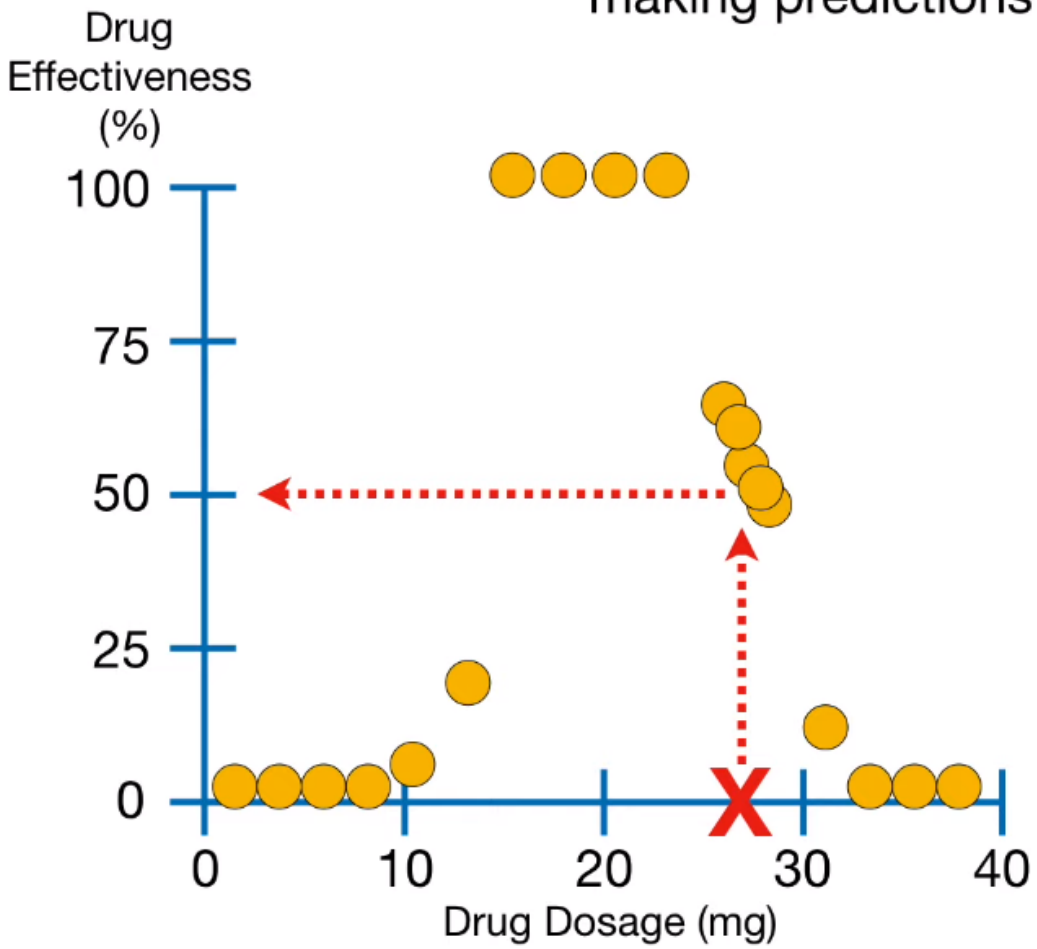
So why make a big deal about the **Regression Tree**?





Decision Tree Regression

When the data are super simple and we are only using one predictor, **Dosage**, to predict **Drug Effectiveness**, making predictions by eye isn't terrible.



Dosage	Drug Effect.
10	58
20	60
35	57
5	44
etc…	etc…

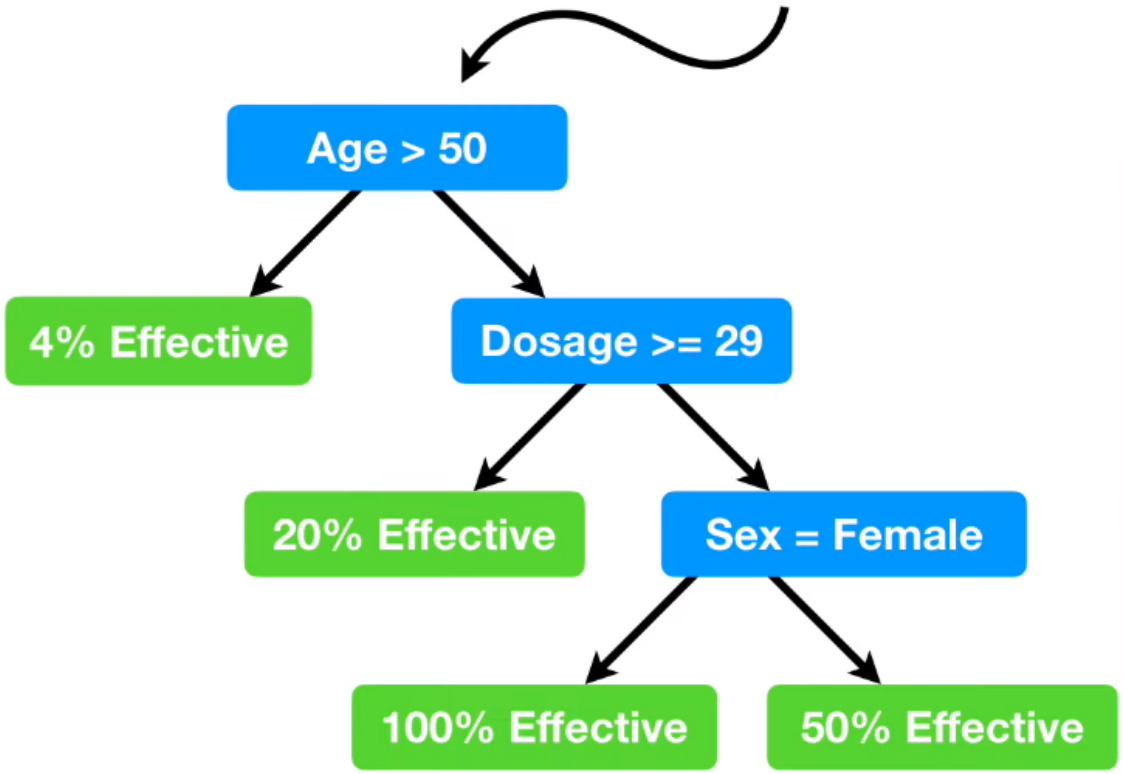
Decision Tree Regression

But when we have **3** or more predictors, like **Dosage**, **Age** and **Sex**, to predict **Drug Effectiveness**, drawing a graph is very difficult, if not impossible.

Dosage	Age	Sex	Etc.	Drug Effect.
10	25	Female	…	98
20	73	Male	…	0
35	54	Female	…	100
5	12	Male	…	44
etc…	etc…	etc…	etc…	etc…

Decision Tree Regression

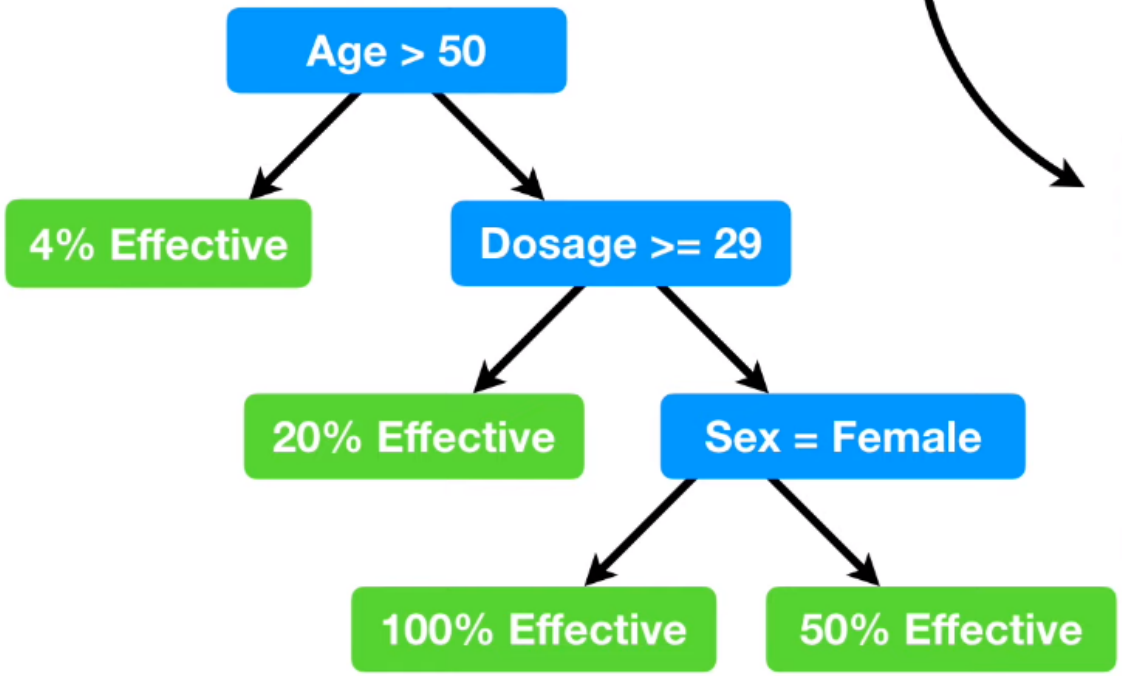
In contrast, a **Regression Tree** easily accommodates the additional predictors.



Dosage	Age	Sex	Etc.	Drug Effect.
10	25	Female	…	98
20	73	Male	…	0
35	54	Female	…	100
5	12	Male	…	44
etc…	etc…	etc…	etc…	etc…

Decision Tree Regression

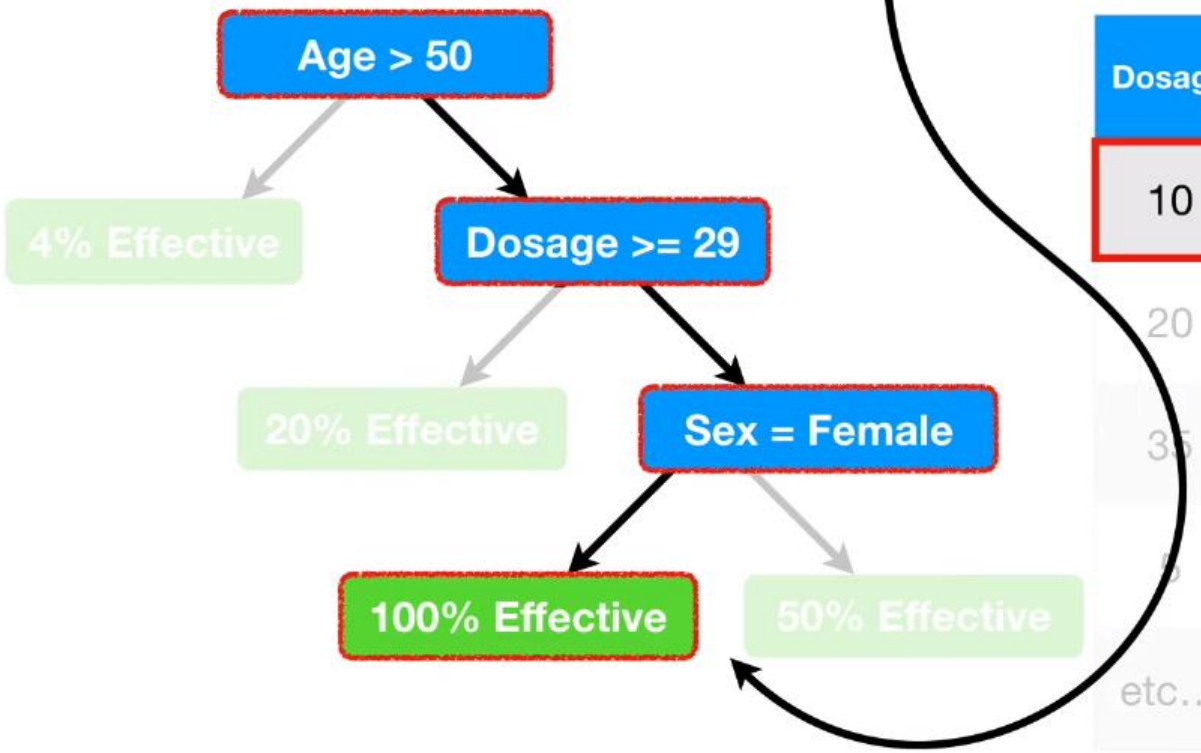
For example, if we wanted to predict the **Drug Effectiveness** for this patient…



Dosage	Age	Sex	Etc.	Drug Effect.
10	25	Female	…	98
20	73	Male	…	0
35	54	Female	…	100
5	12	Male	…	44
etc…	etc…	etc…	etc…	etc…

Decision Tree Regression

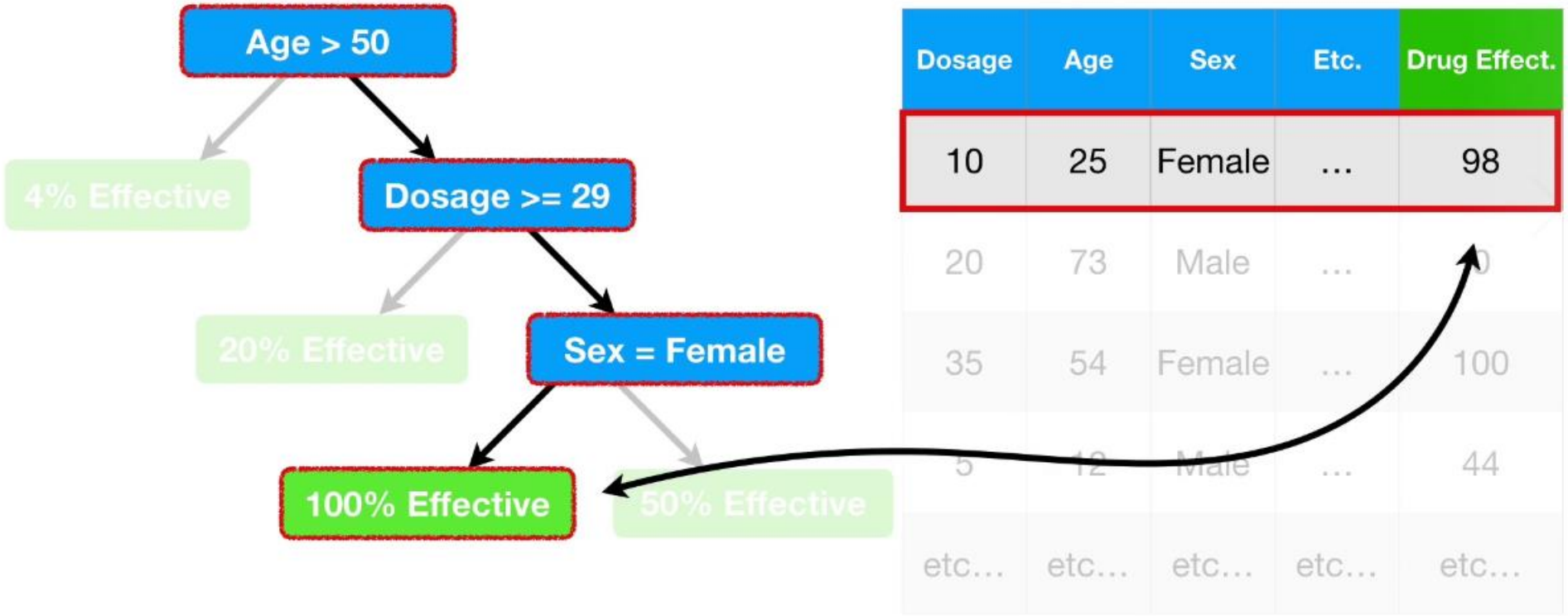
…and since they *are* **Female**, we follow the branch on the *left* and predict that the dosage will be **100% Effective**…



Dosage	Age	Sex	Etc.	Drug Effect.
10	25	Female	…	98
20	73	Male	…	0
35	54	Female	…	100
5	12	Male	…	44
etc…	etc…	etc…	etc…	etc…

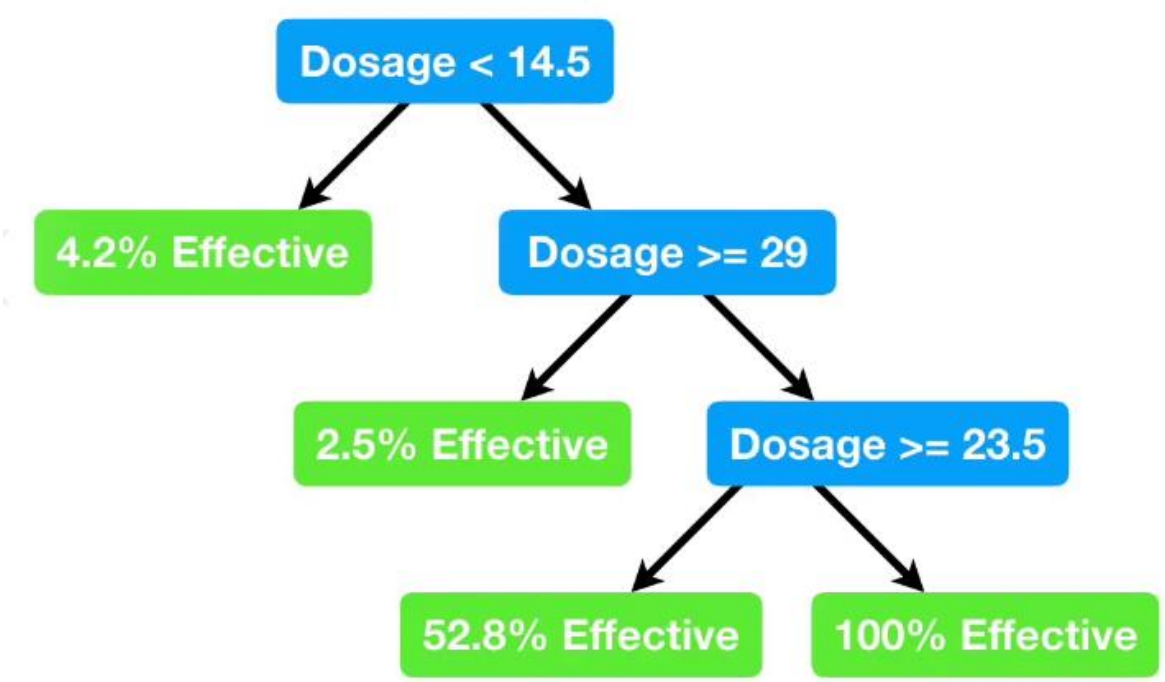
Decision Tree Regression

…and that's not too far off from the truth, **98%**.



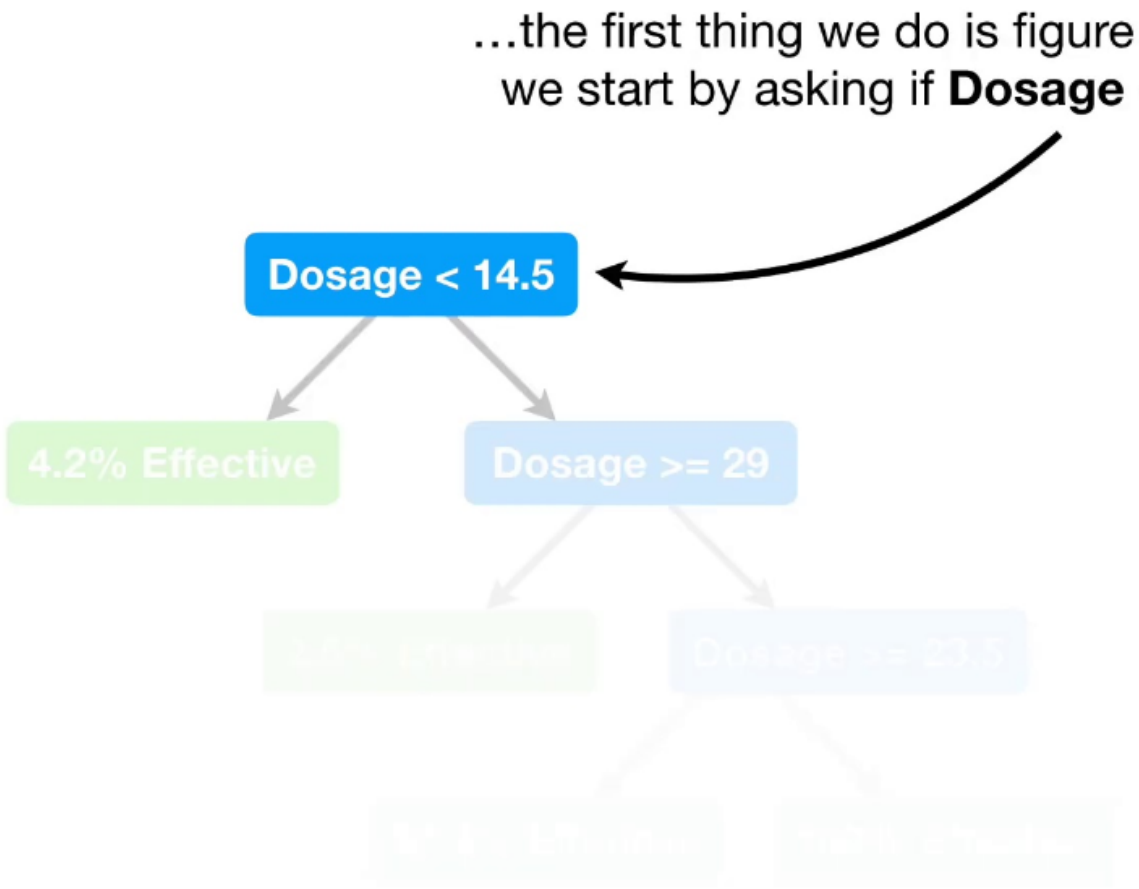
Decision Tree Regression

…and talk about how to build this **Regression Tree** from scratch…



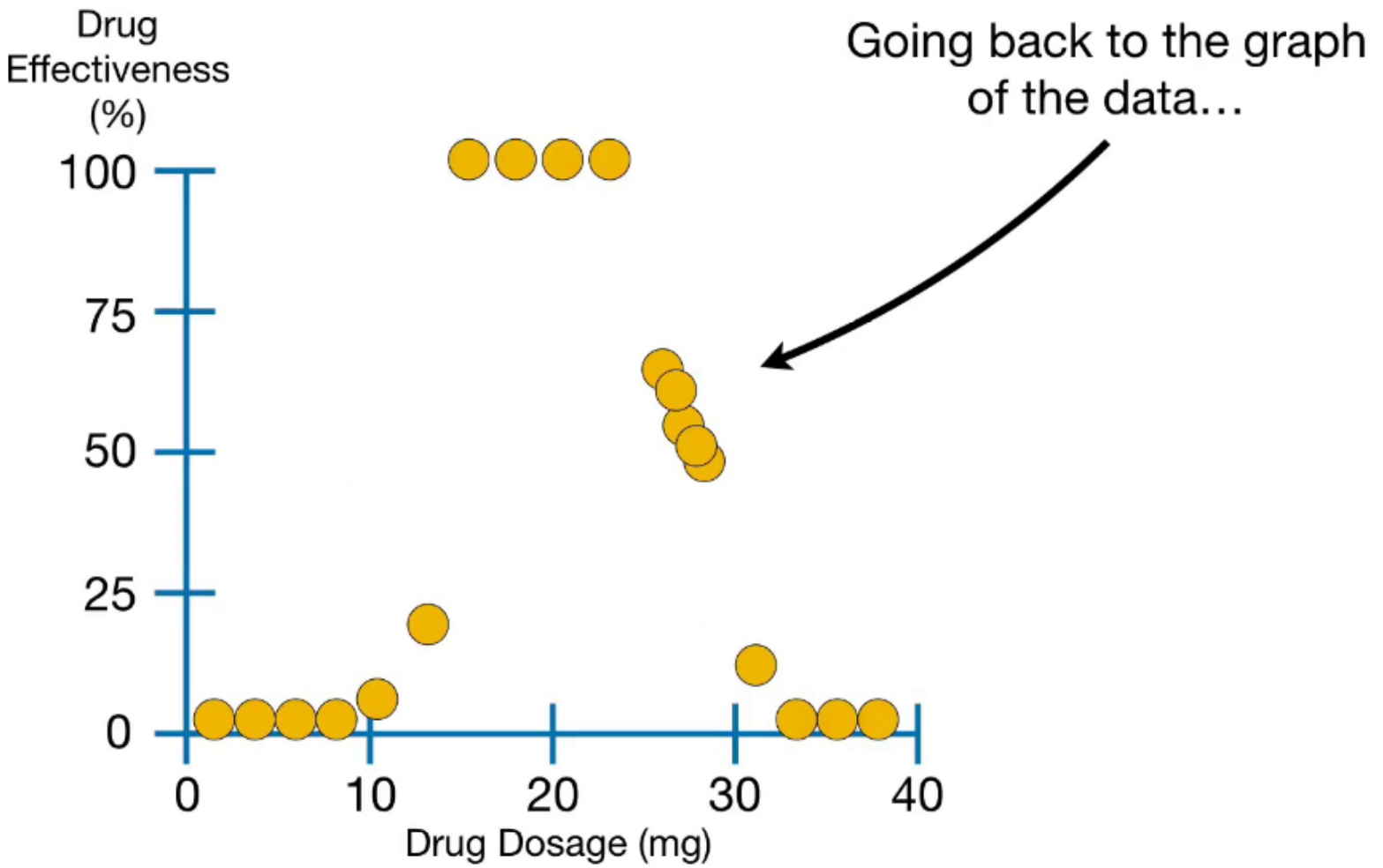
Dosage	Drug Effect.
10	58
20	60
35	57
5	44
etc…	etc…

Decision Tree Regression



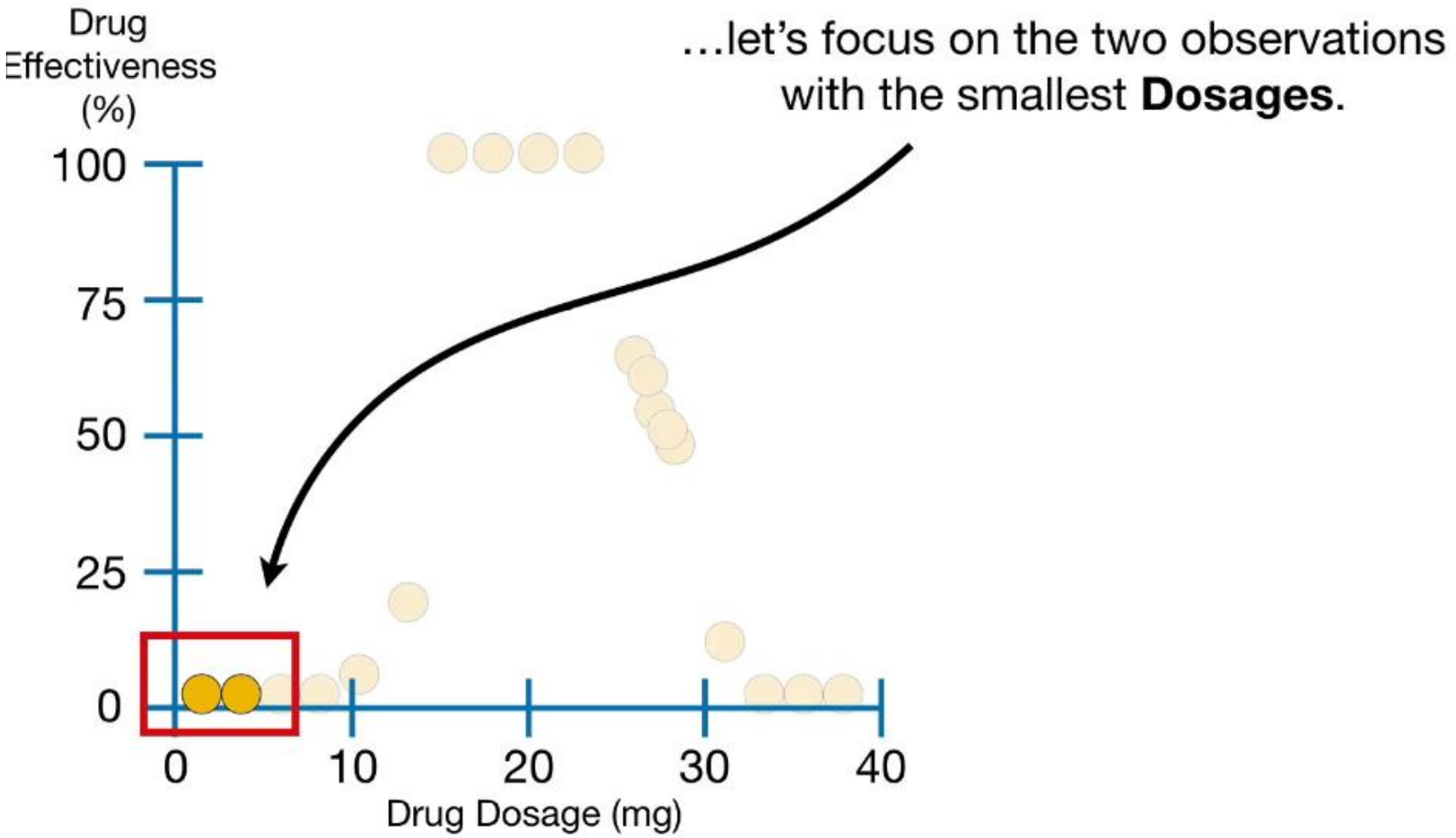
Dosage	Drug Effect.
10	58
20	60
35	57
5	44
etc…	etc…

Decision Tree Regression

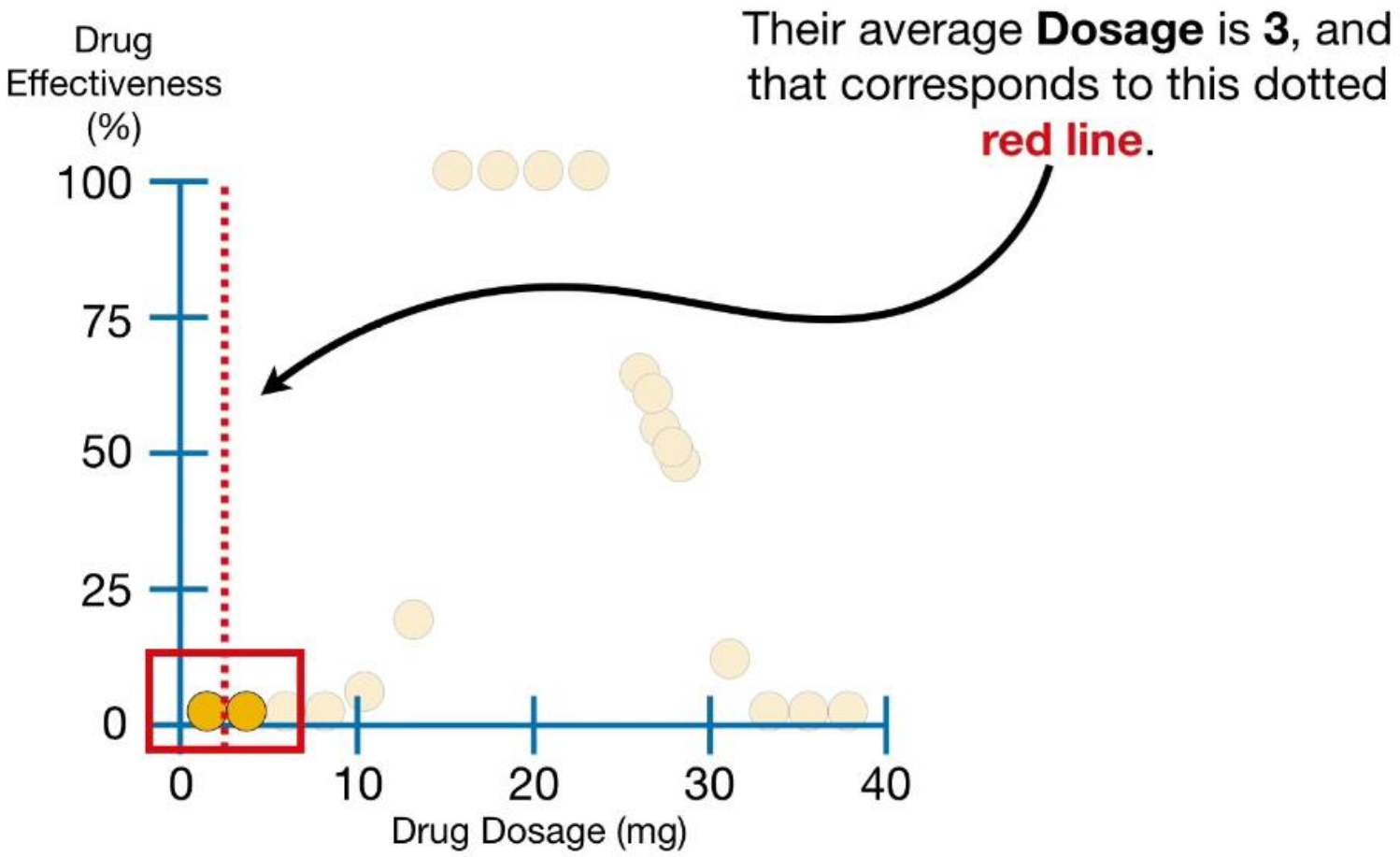


Dosage	Drug Effect.
10	58
20	60
35	57
5	44
etc…	etc…

Decision Tree Regression

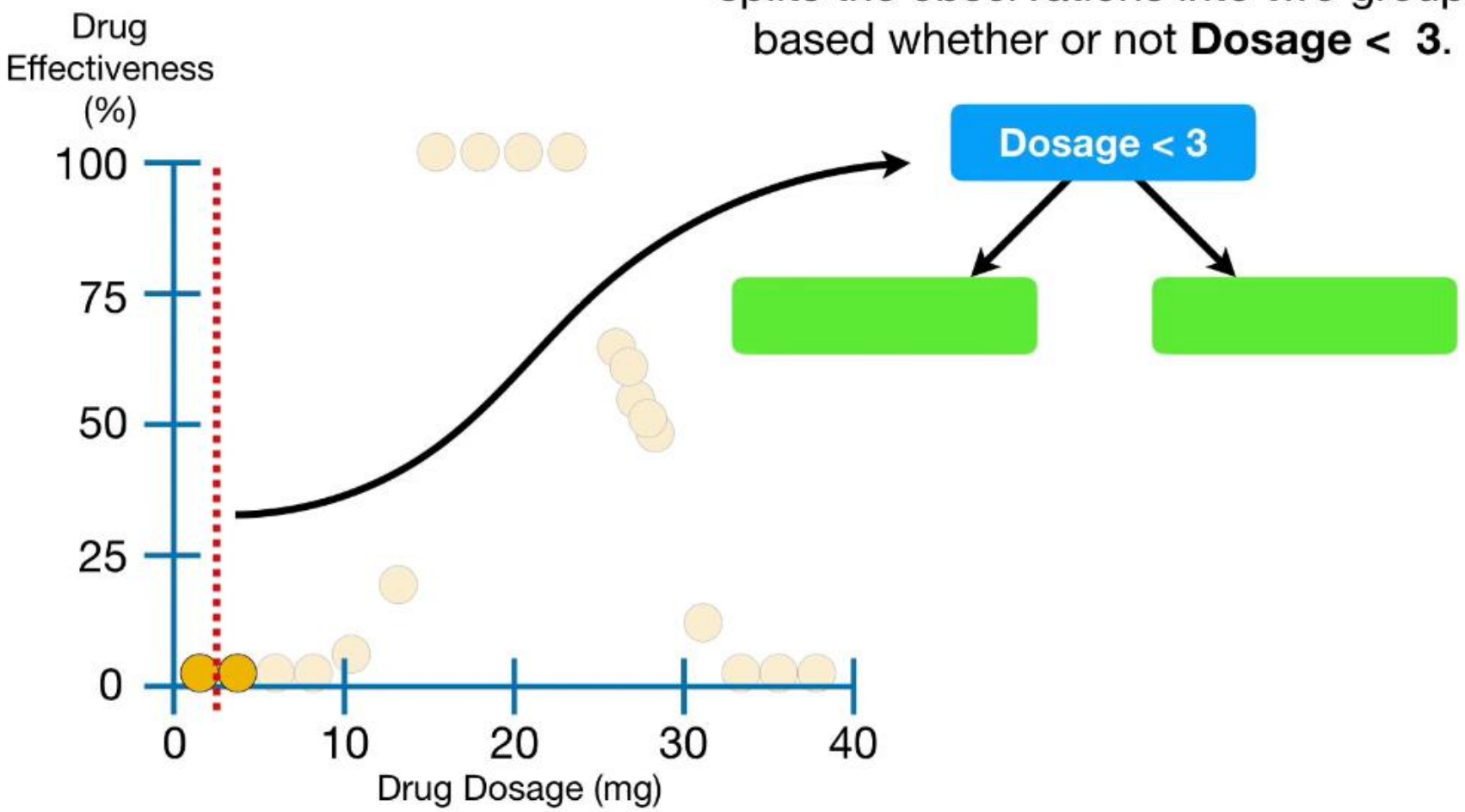


Decision Tree Regression

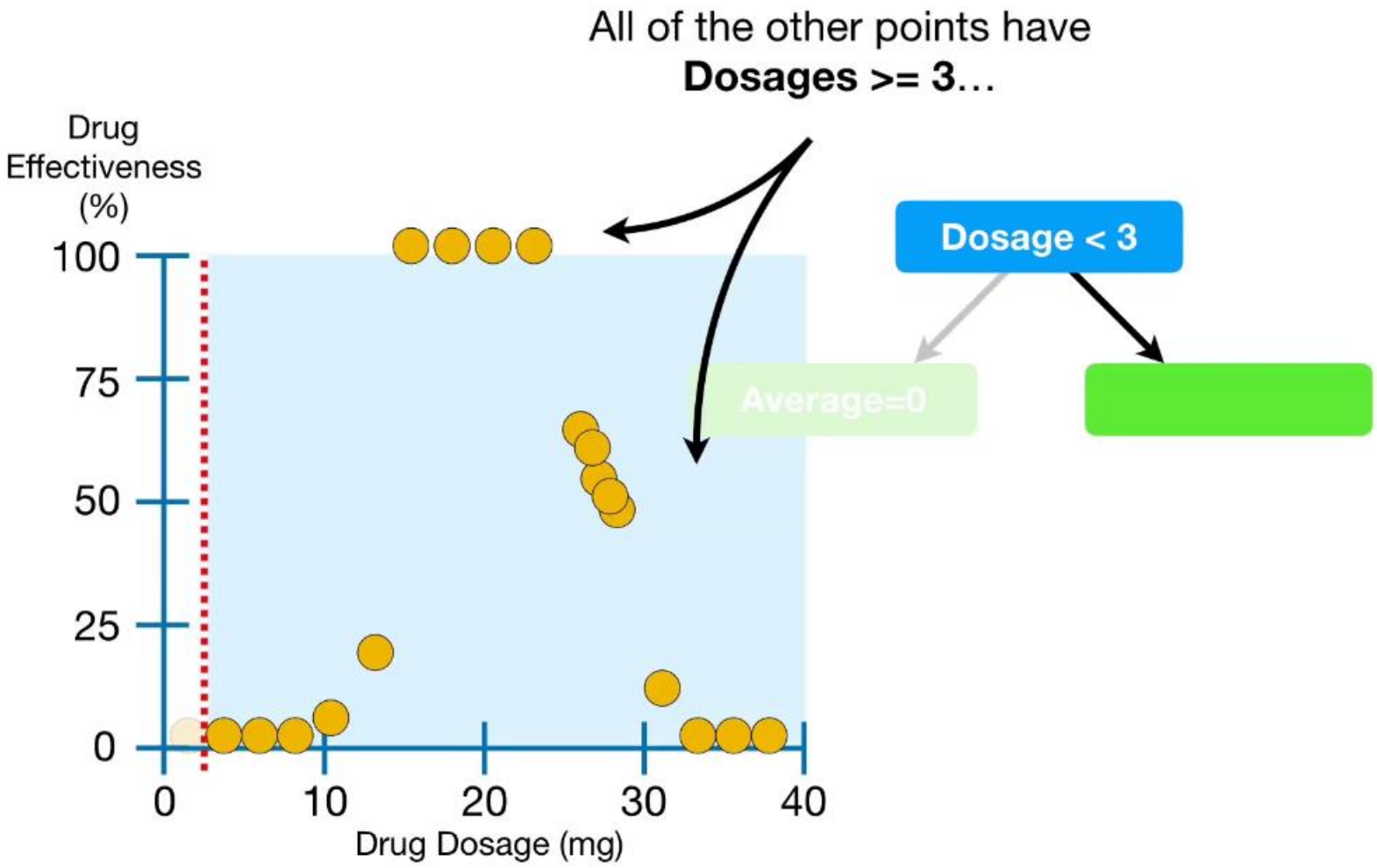


Decision Tree Regression

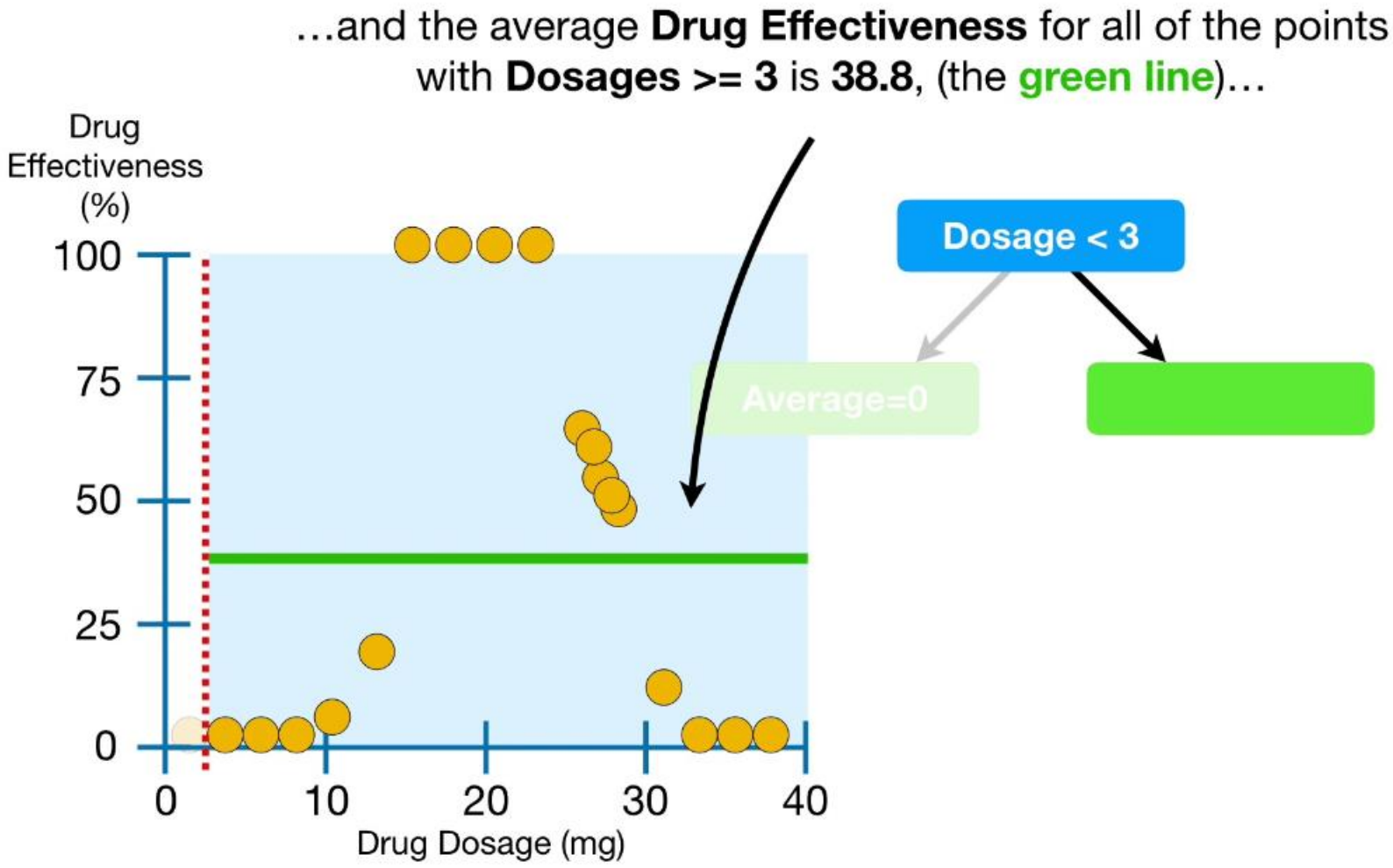
Now we can build a very simple tree that splits the observations into two groups based whether or not **Dosage < 3**.



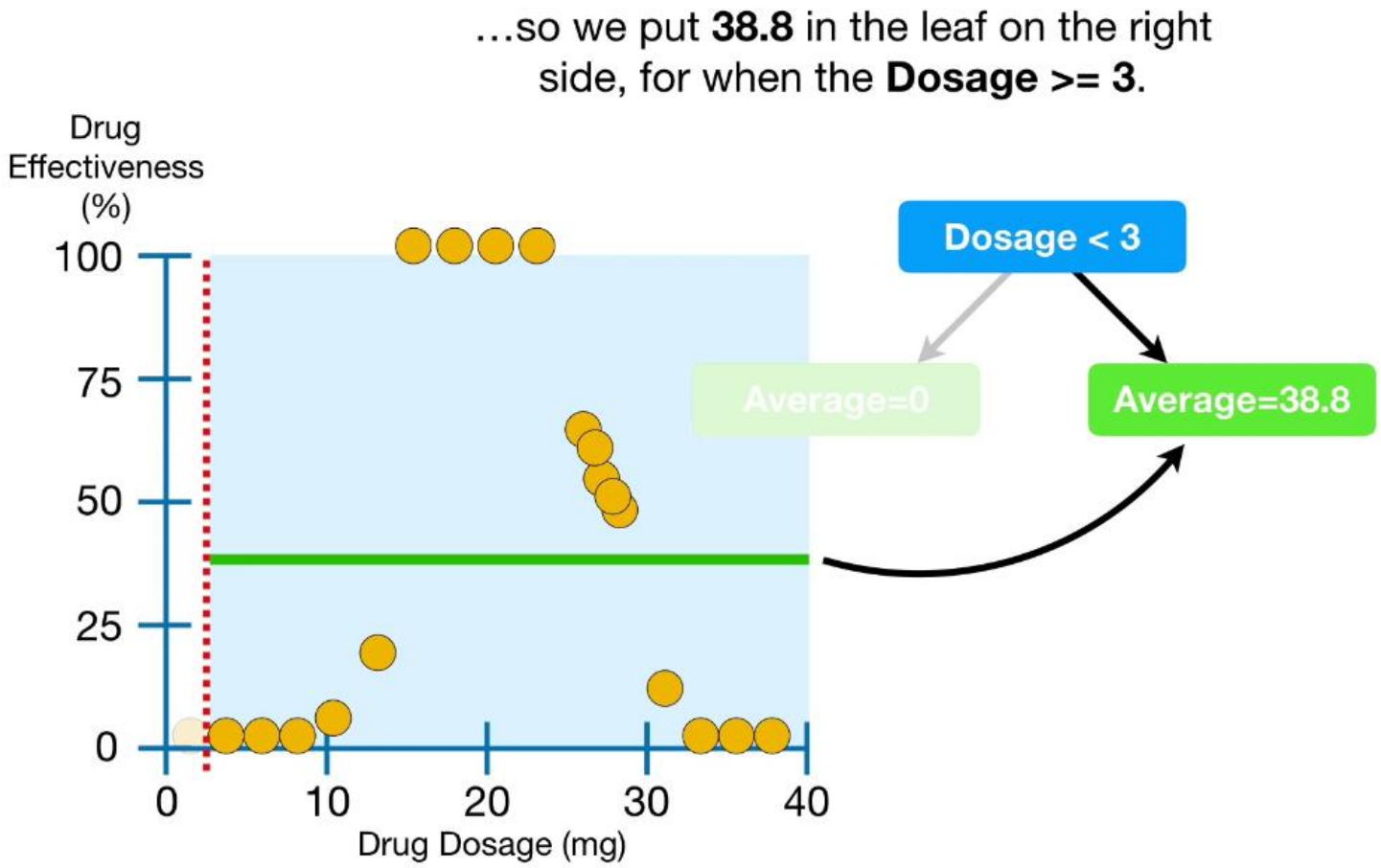
Decision Tree Regression



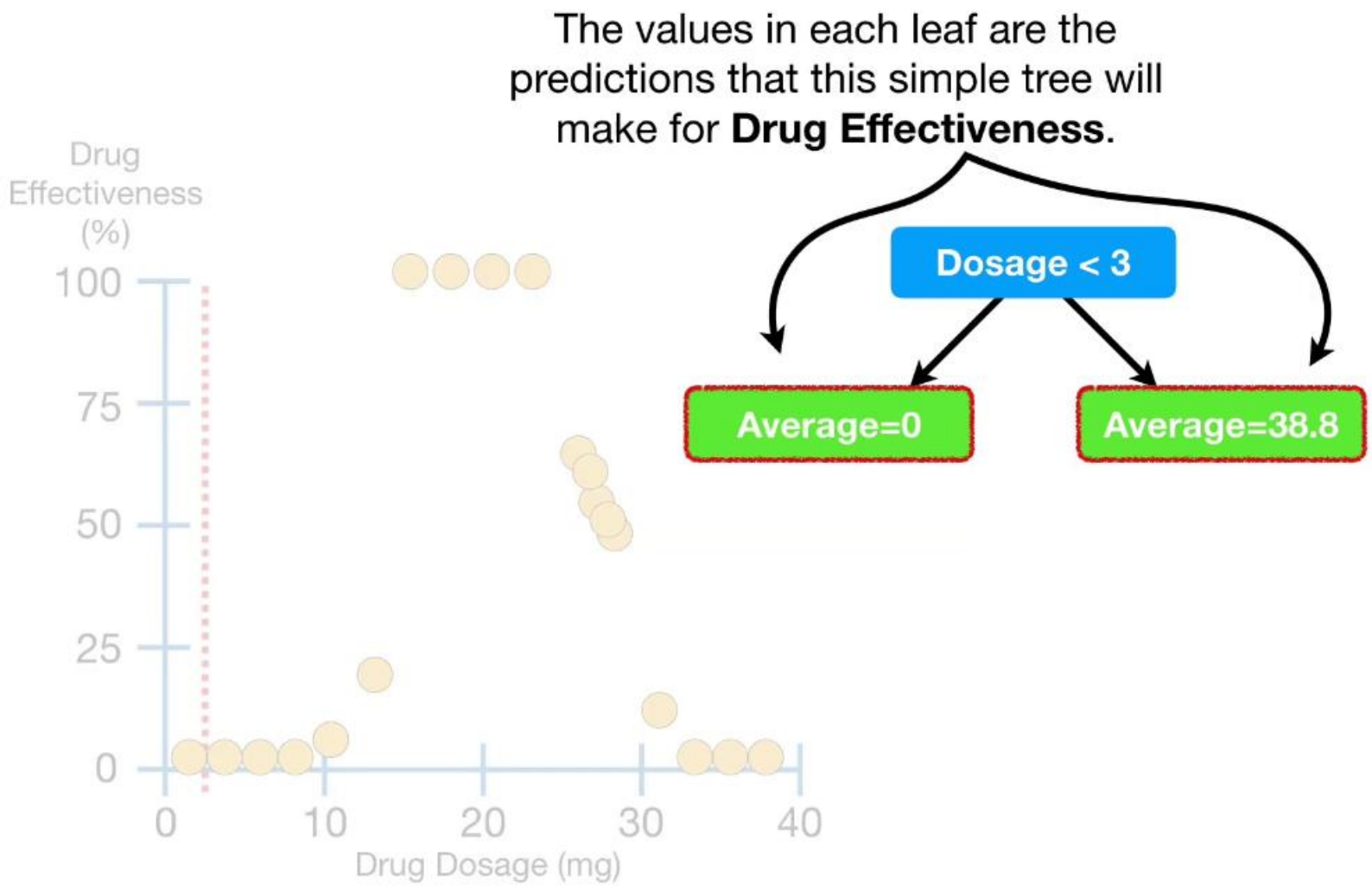
Decision Tree Regression



Decision Tree Regression

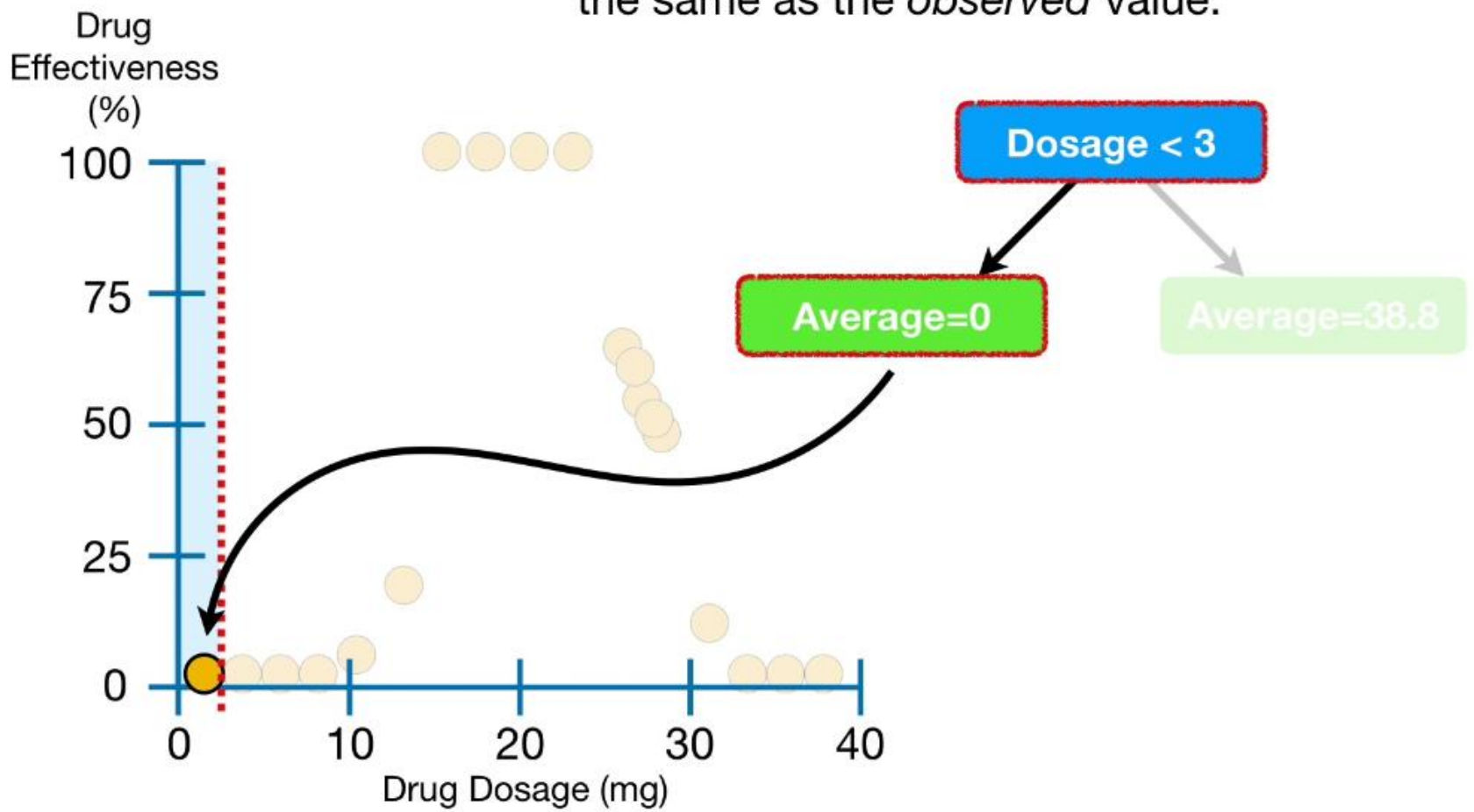


Decision Tree Regression

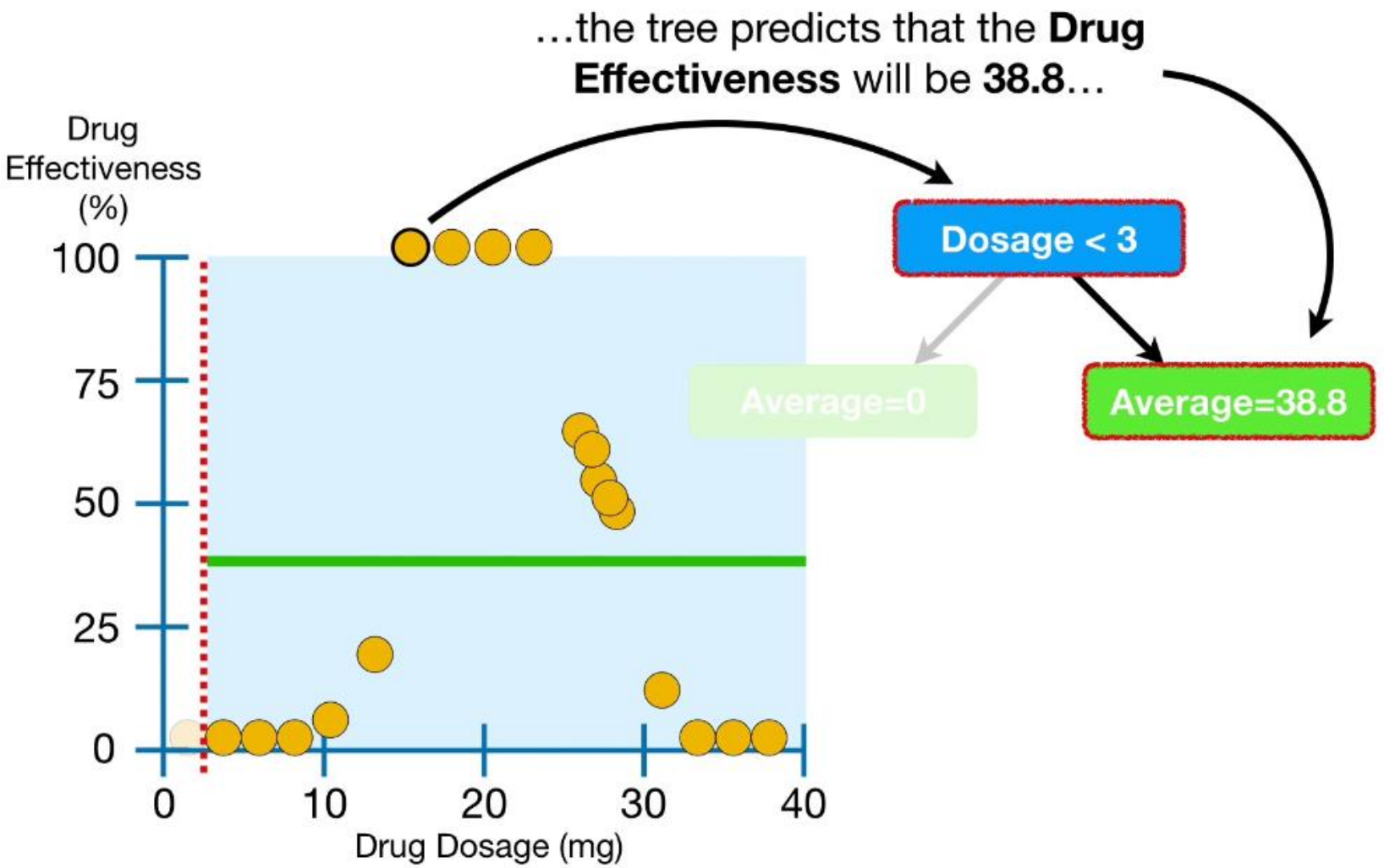


Decision Tree Regression

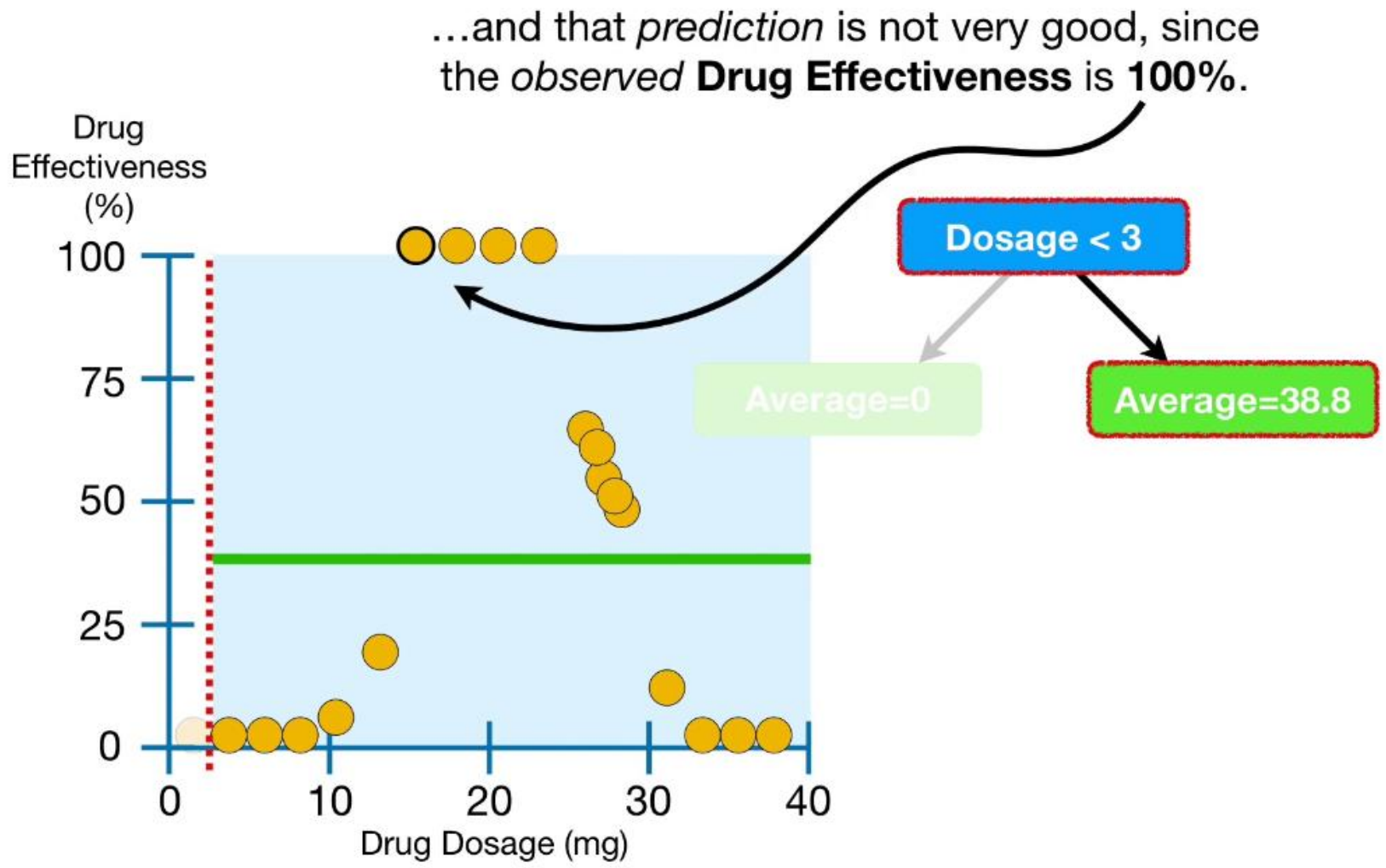
The *prediction* for this point, **Drug Effectiveness = 0**, is pretty good since it is the same as the *observed* value.



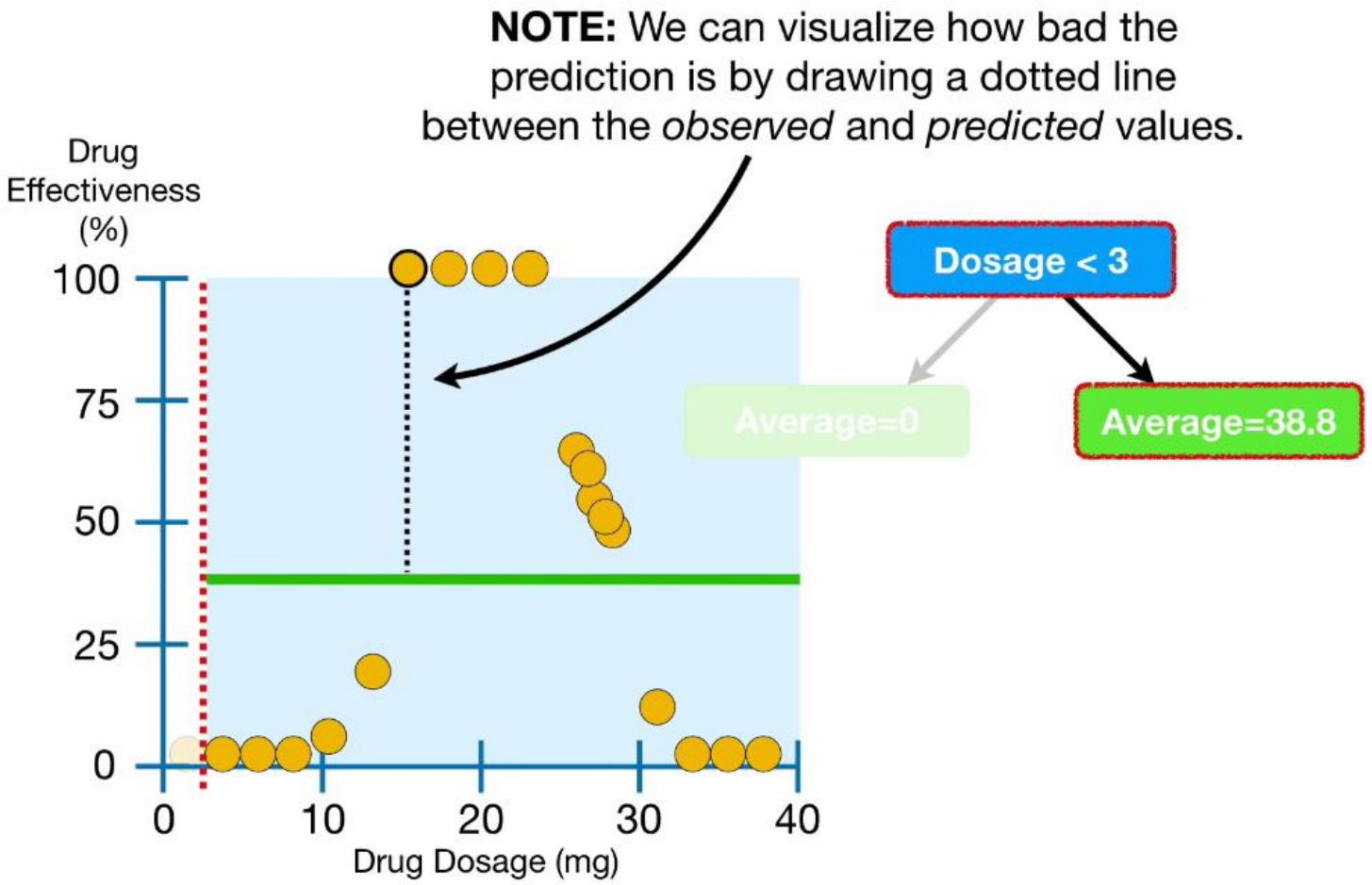
Decision Tree Regression



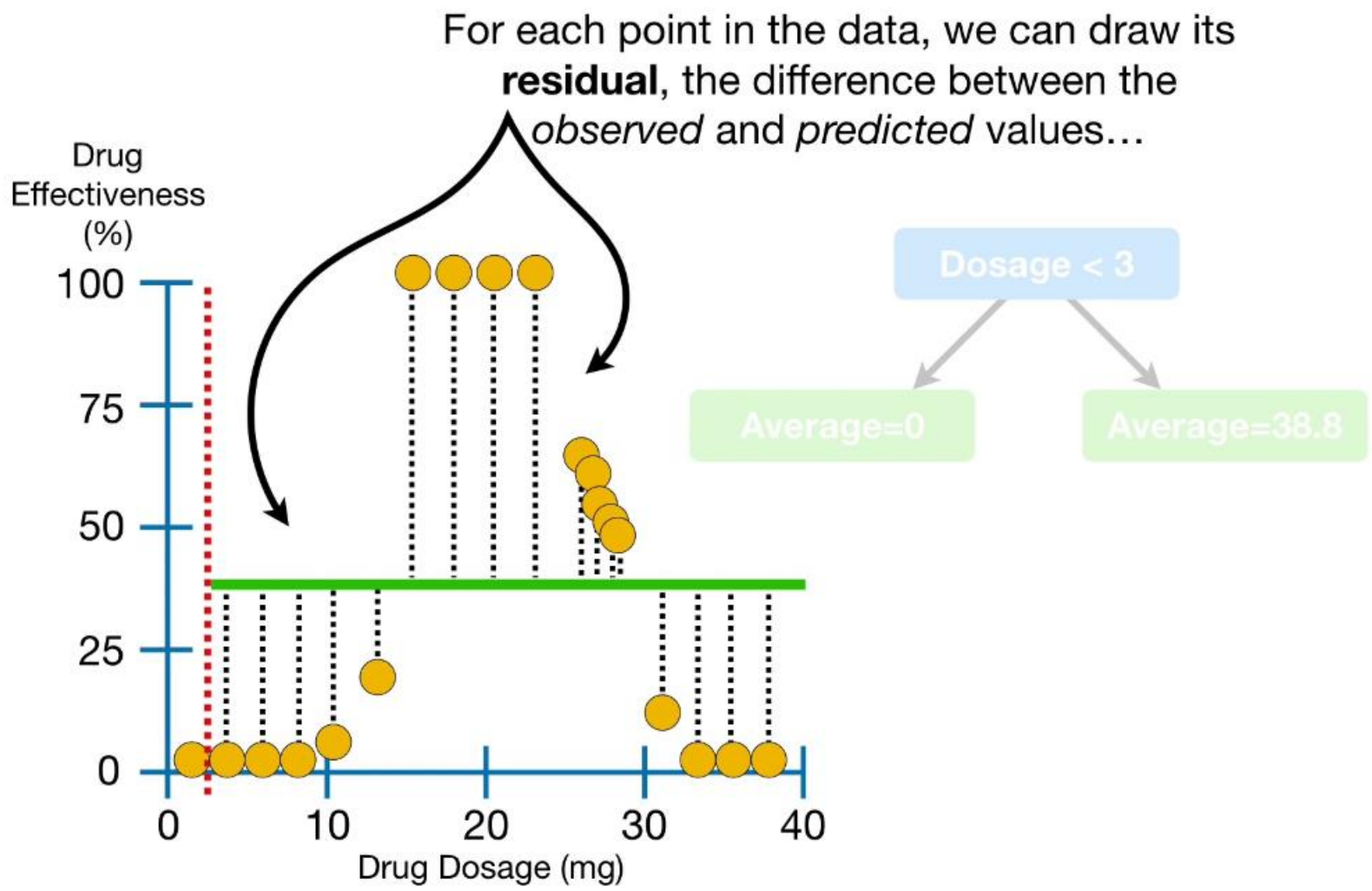
Decision Tree Regression



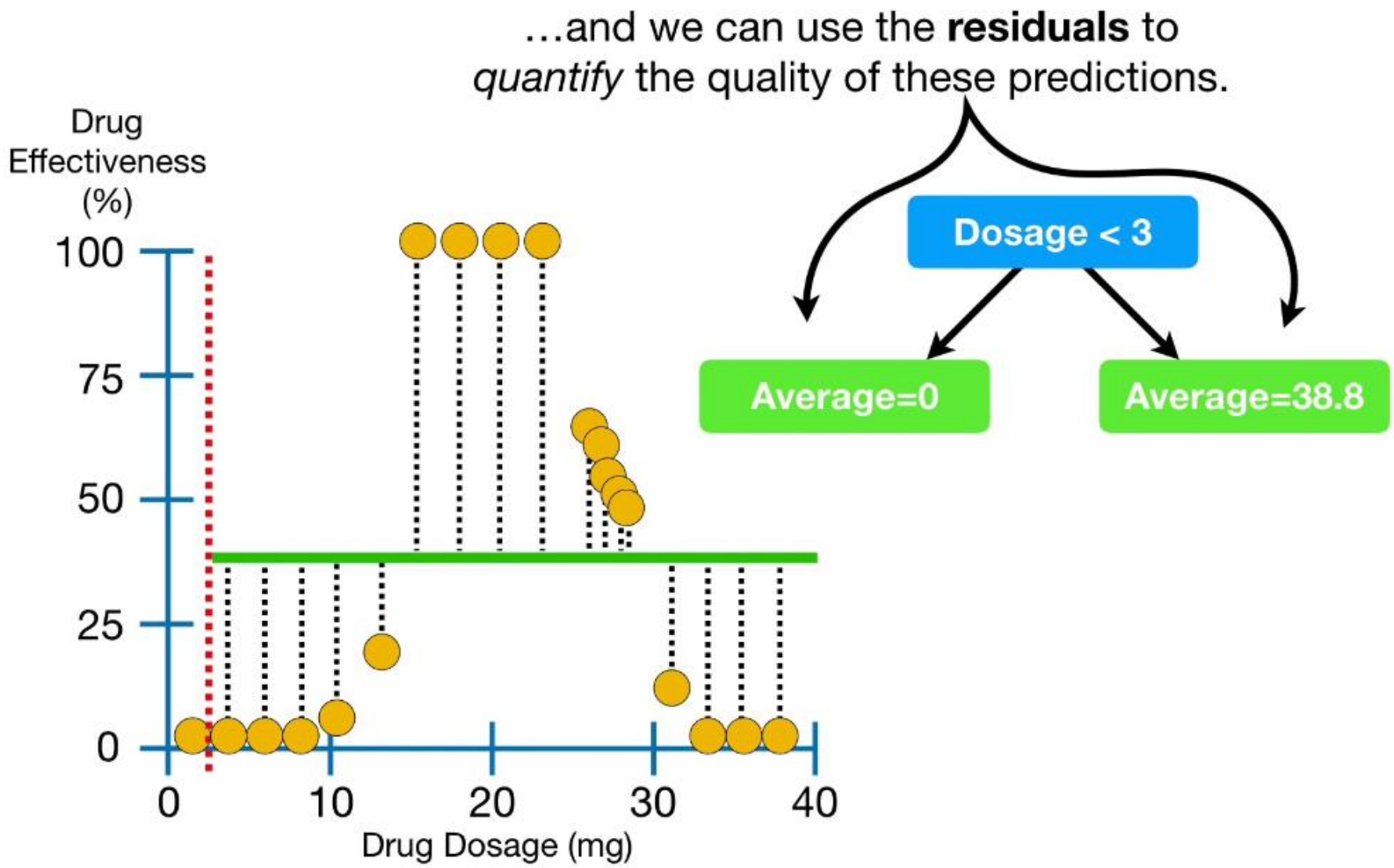
Decision Tree Regression



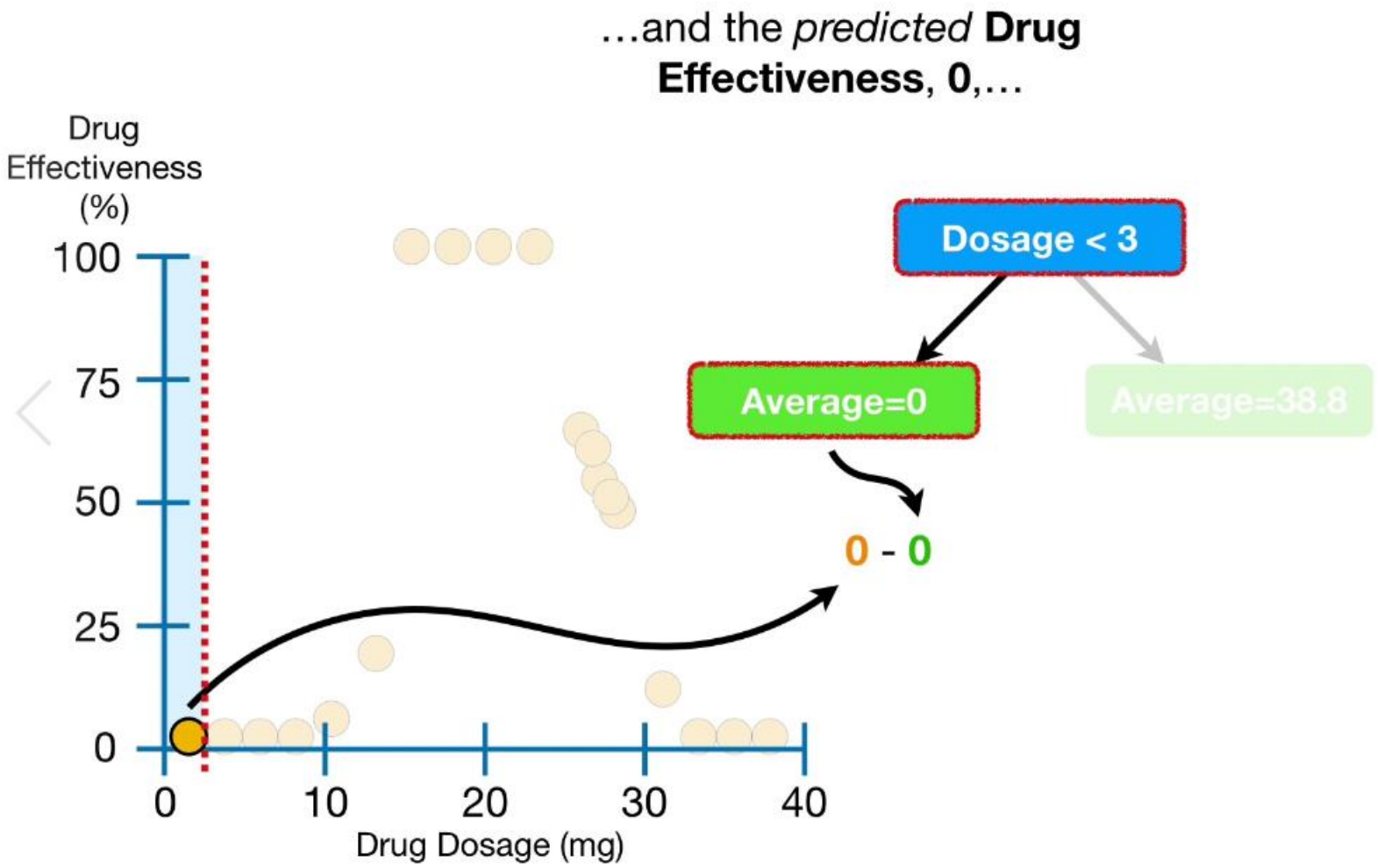
Decision Tree Regression



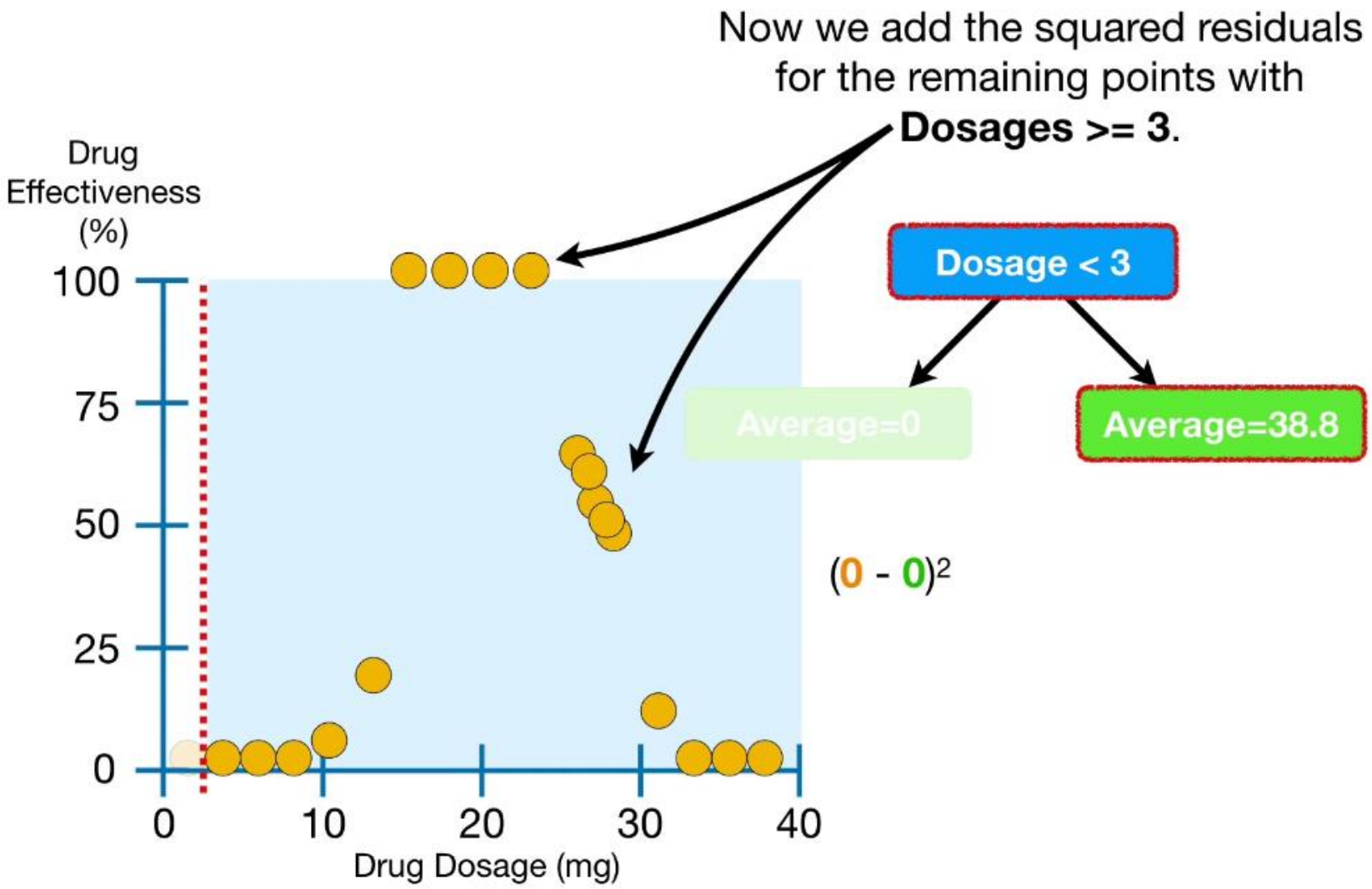
Decision Tree Regression



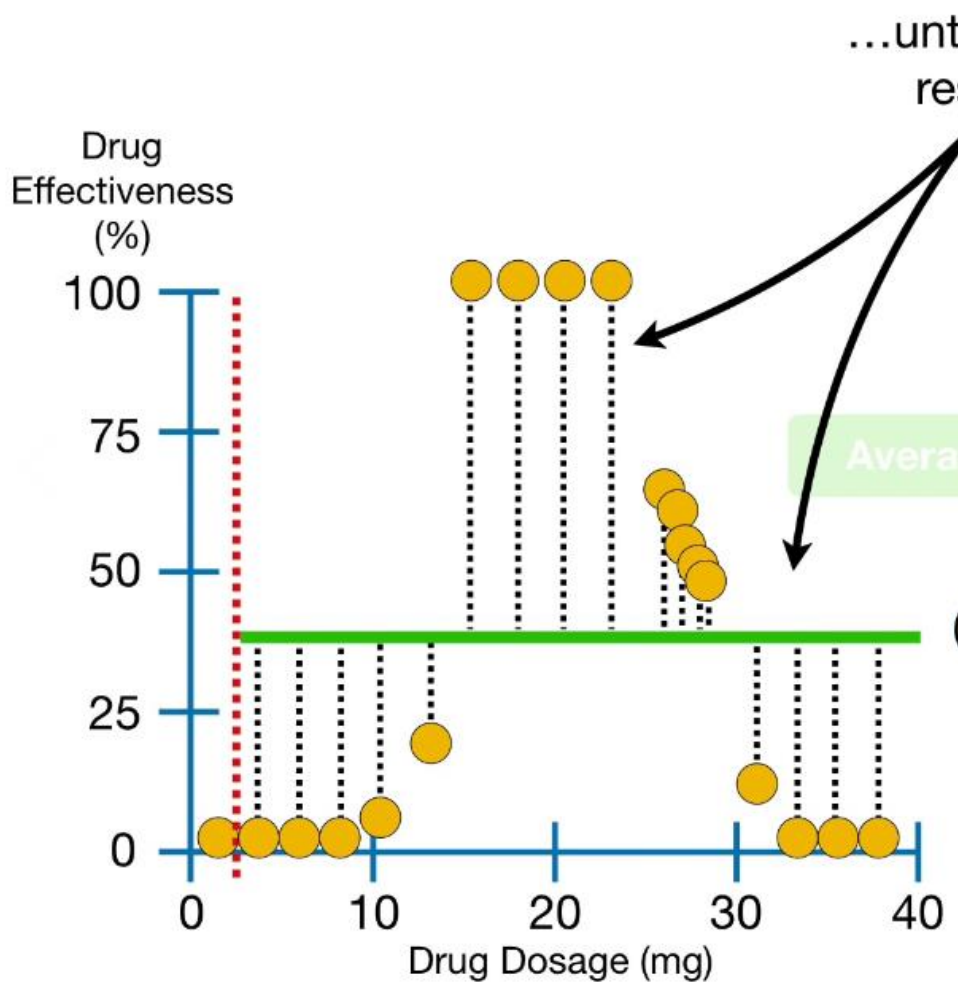
Decision Tree Regression

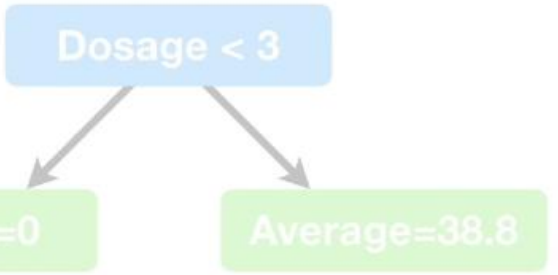


Decision Tree Regression



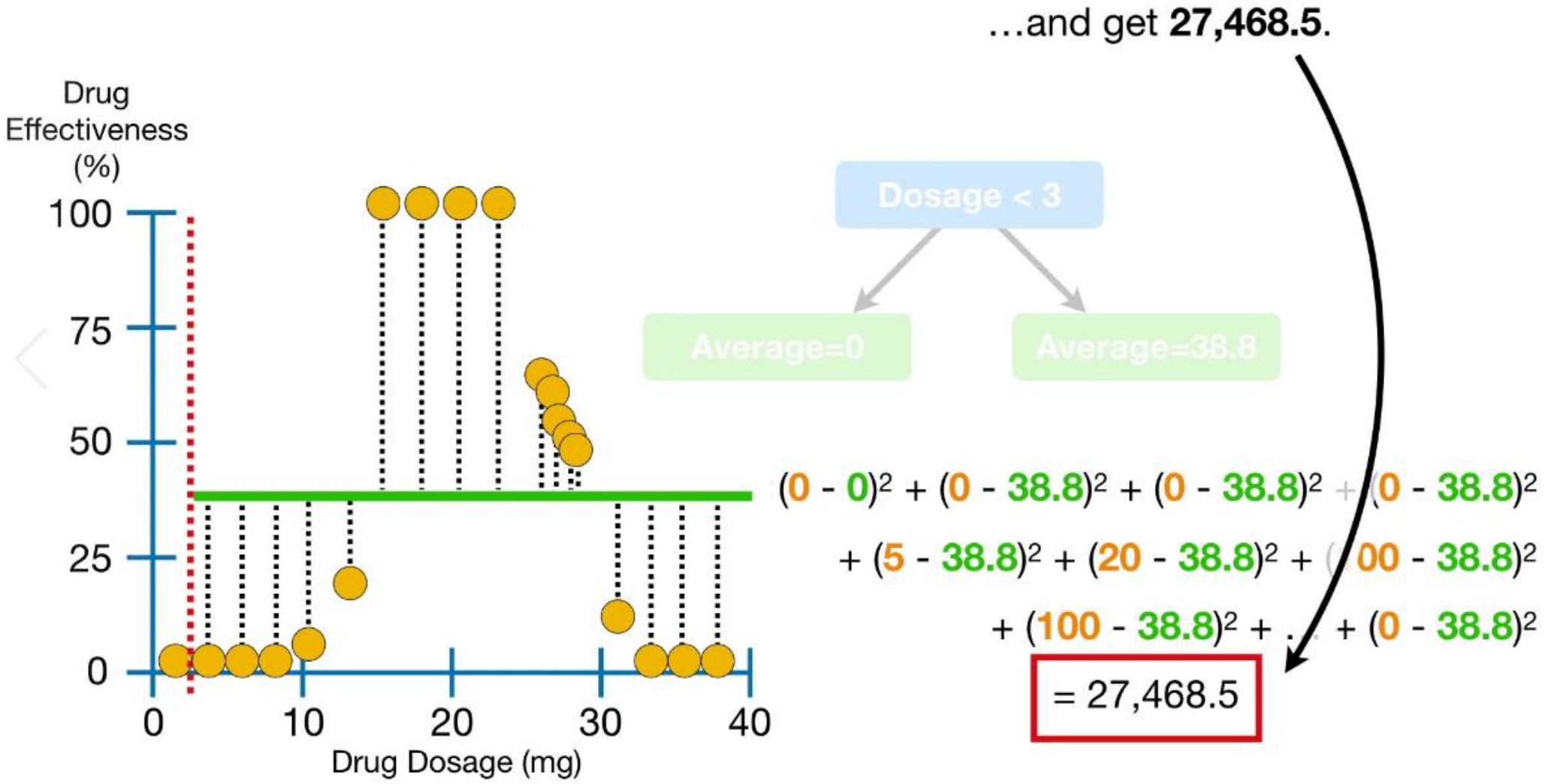
Decision Tree Regression



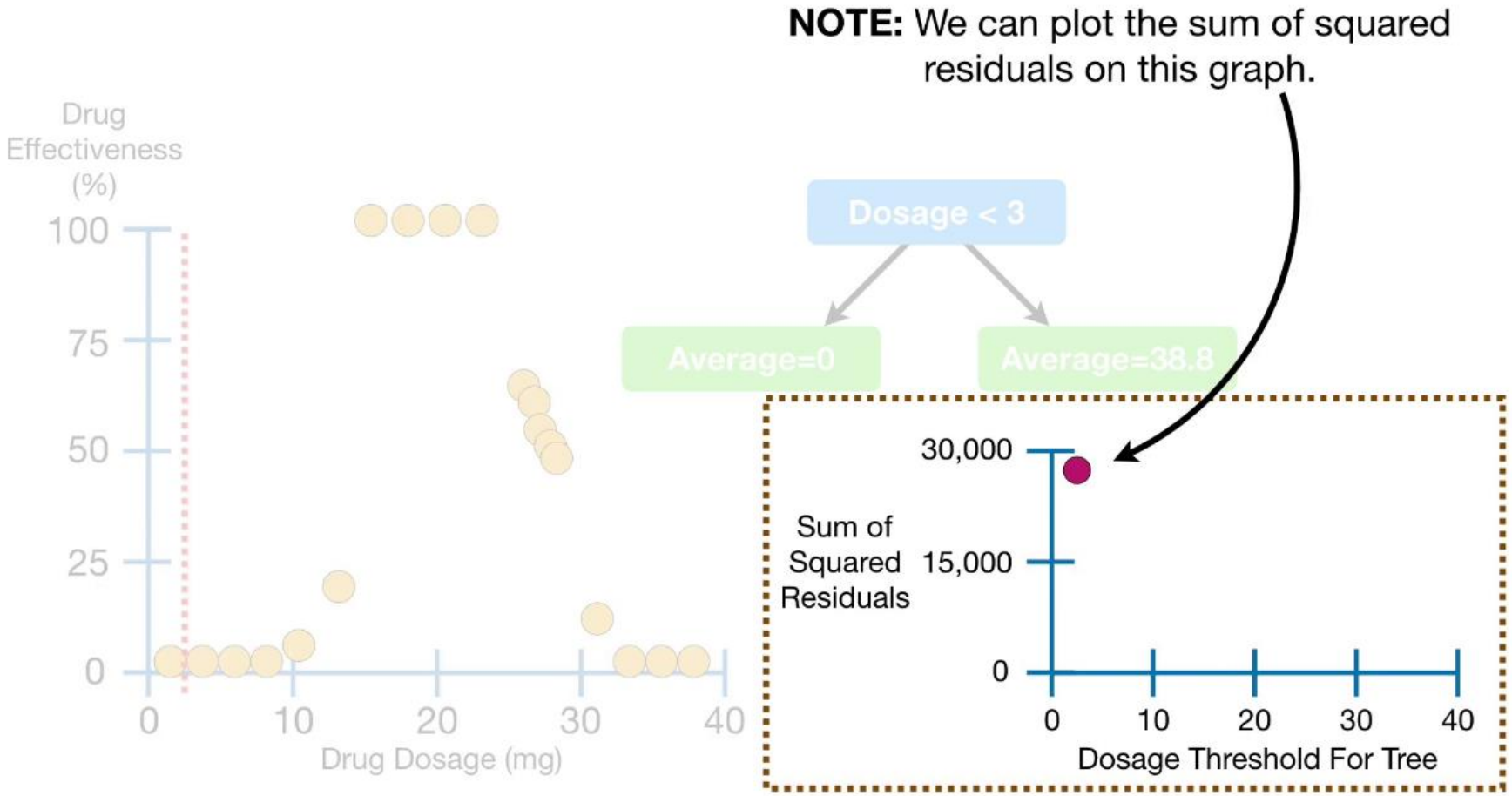


$$(0 - 0)^2 + (0 - 38.8)^2 + (0 - 38.8)^2 + (0 - 38.8)^2$$
$$+ (5 - 38.8)^2 + (20 - 38.8)^2 + (100 - 38.8)^2$$
$$+ (100 - 38.8)^2 + \ldots + (0 - 38.8)^2$$

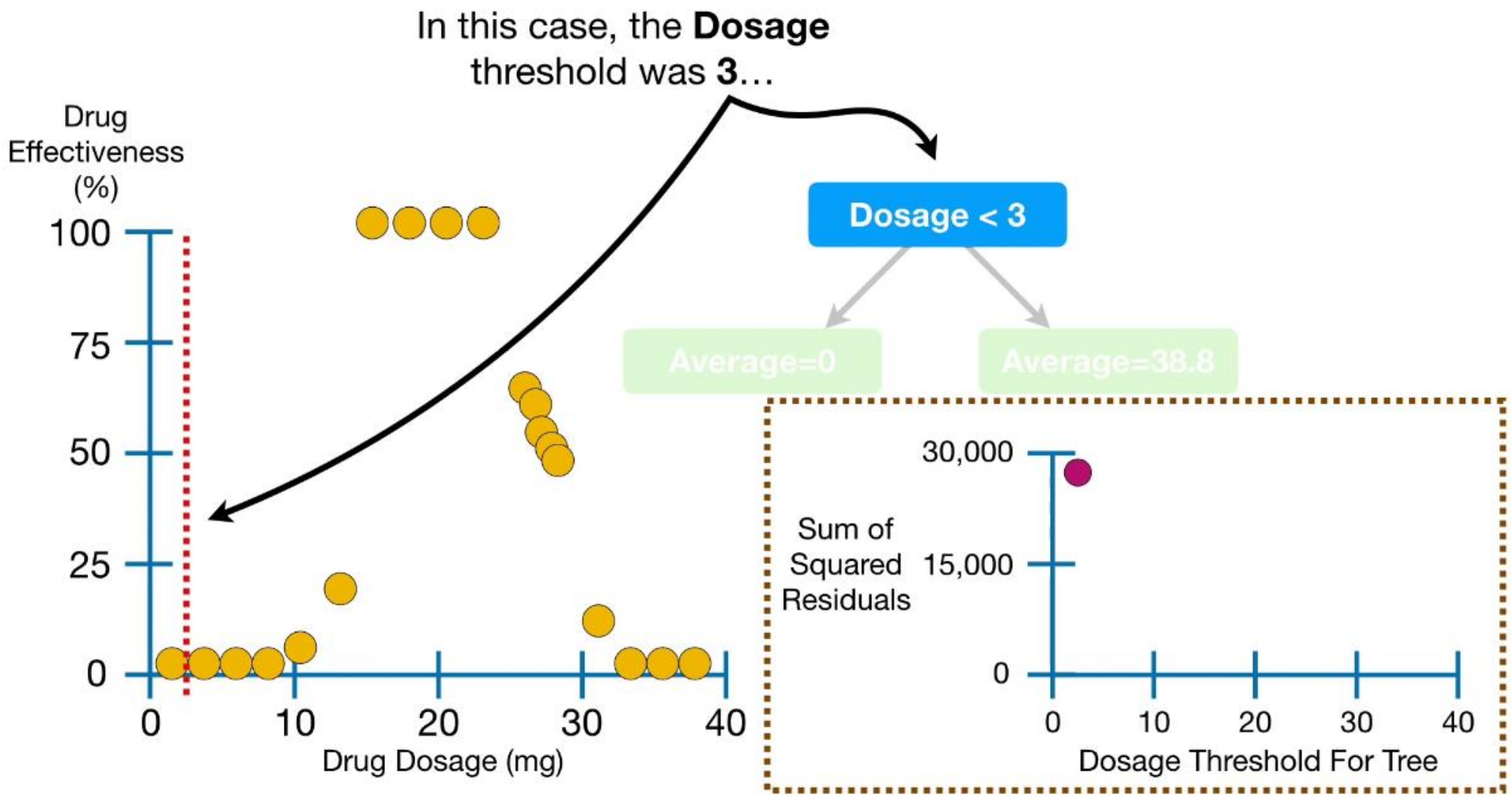
Decision Tree Regression



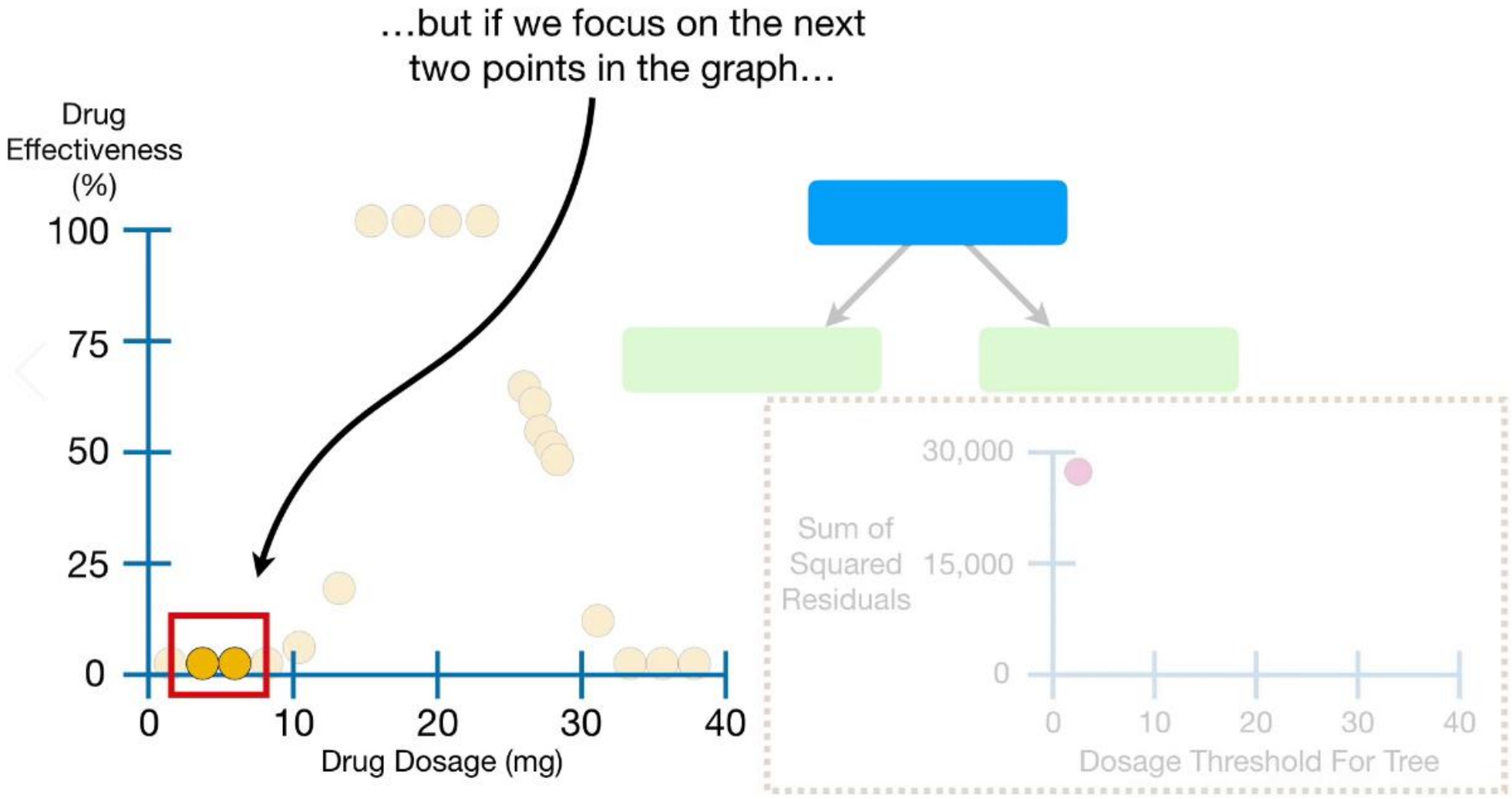
Decision Tree Regression



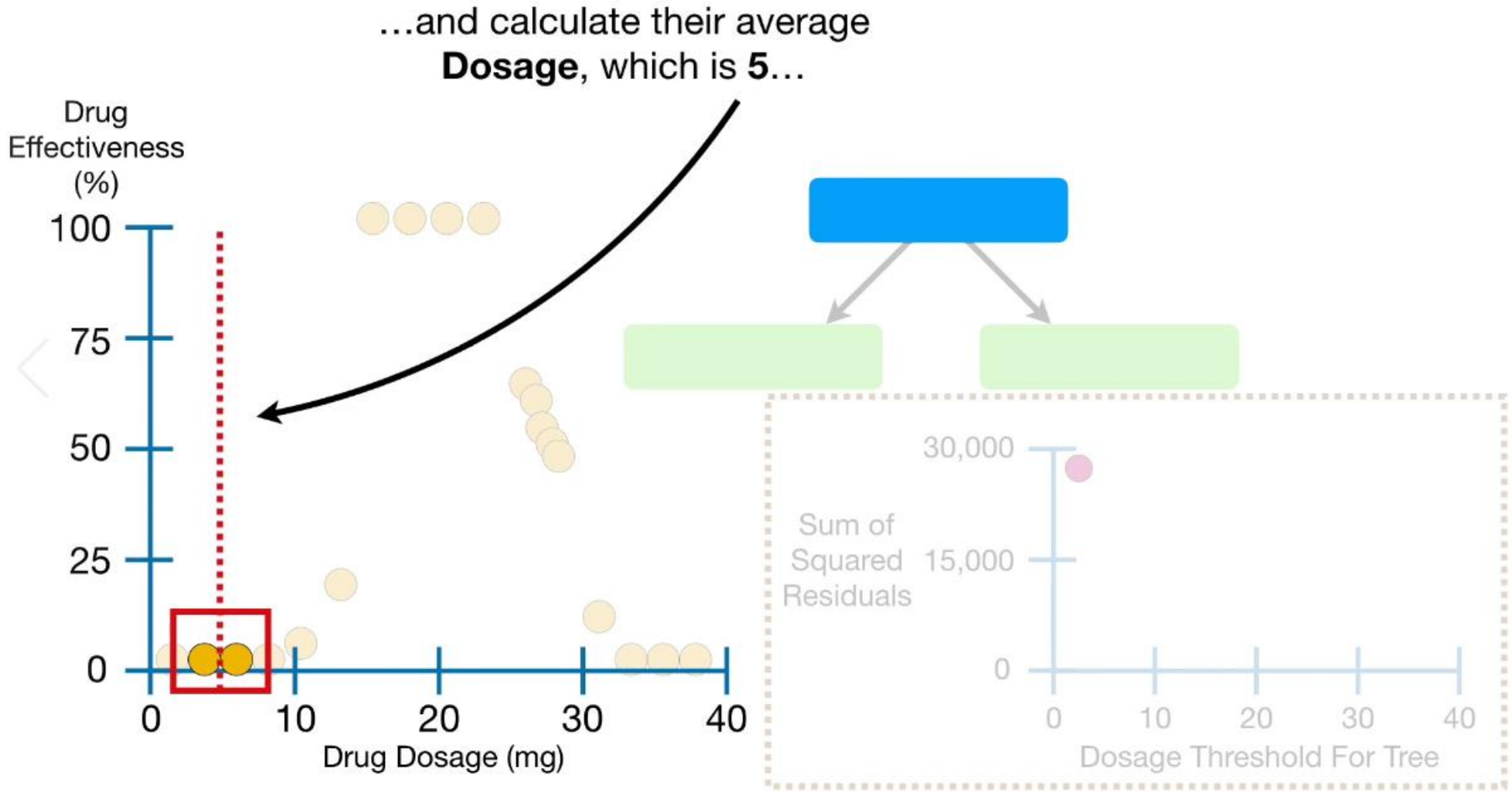
Decision Tree Regression



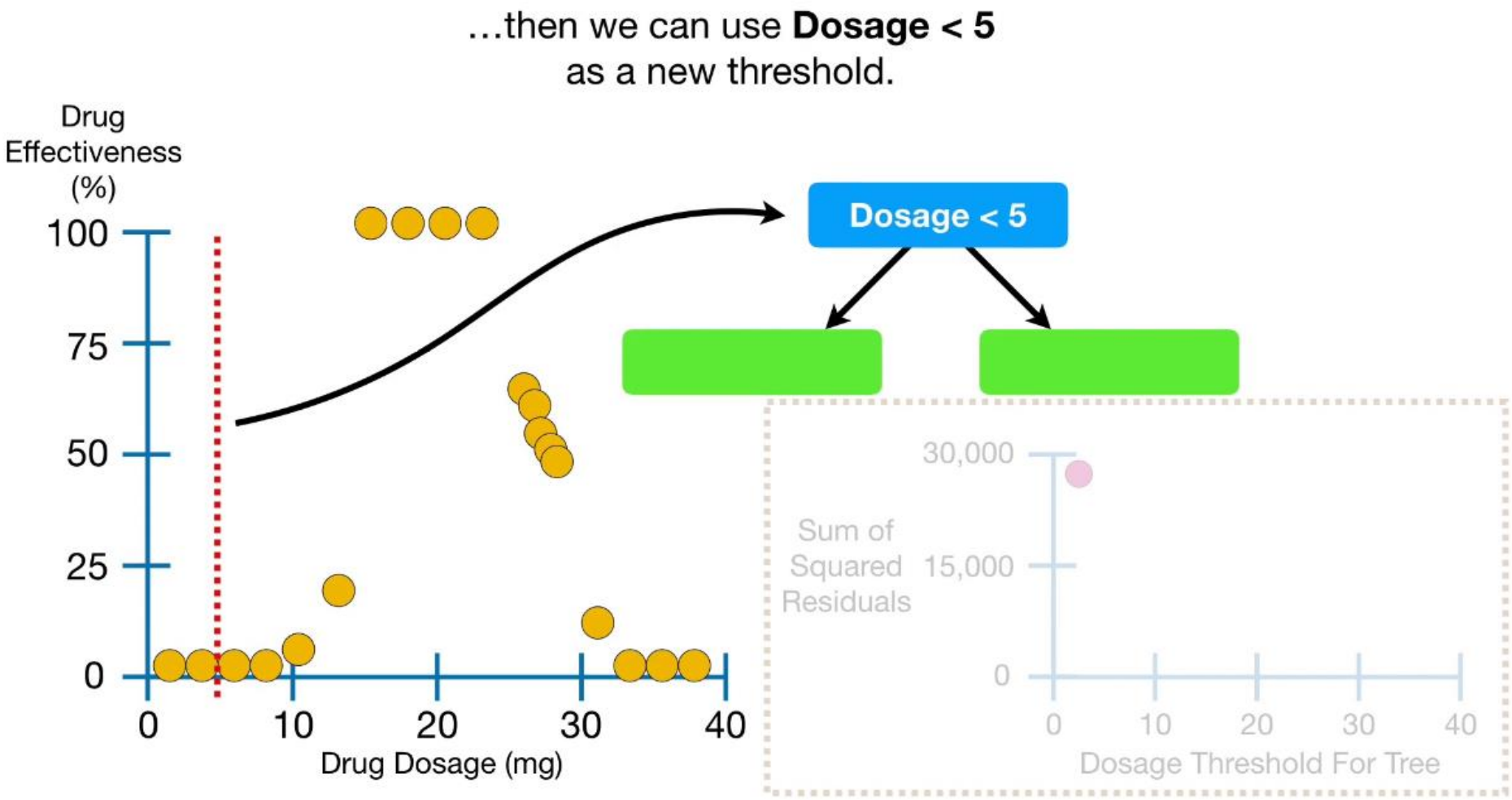
Decision Tree Regression



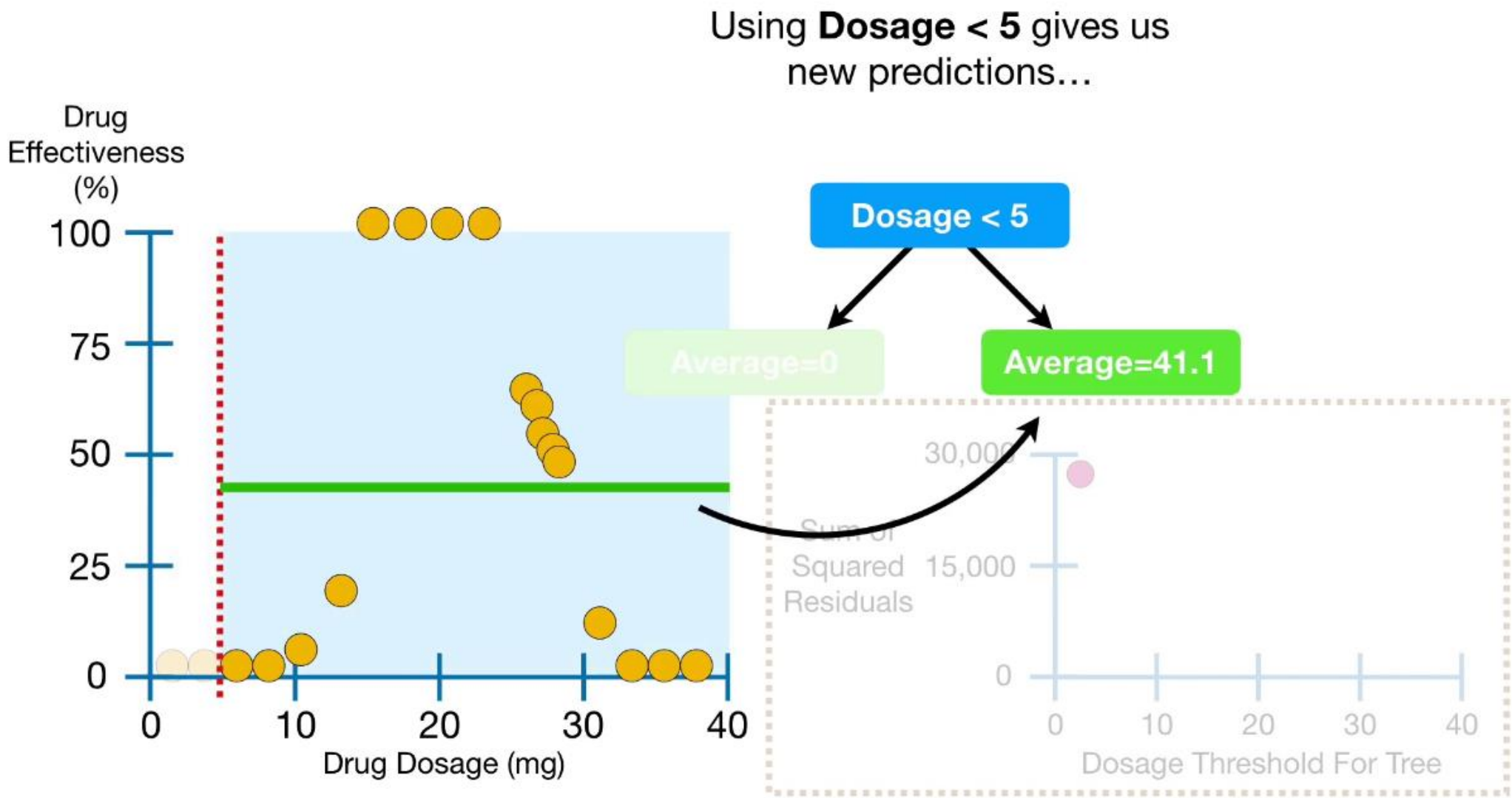
Decision Tree Regression



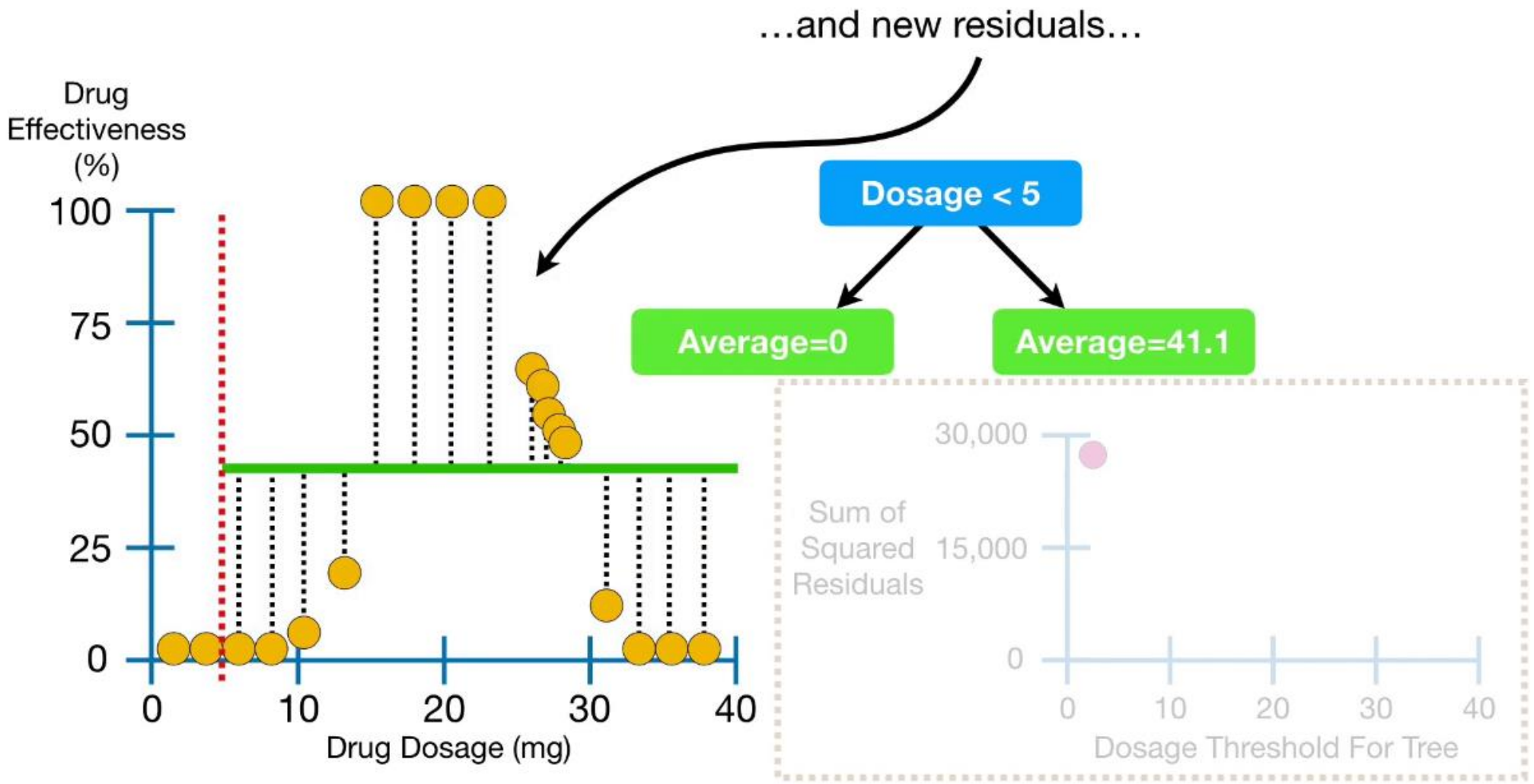
Decision Tree Regression



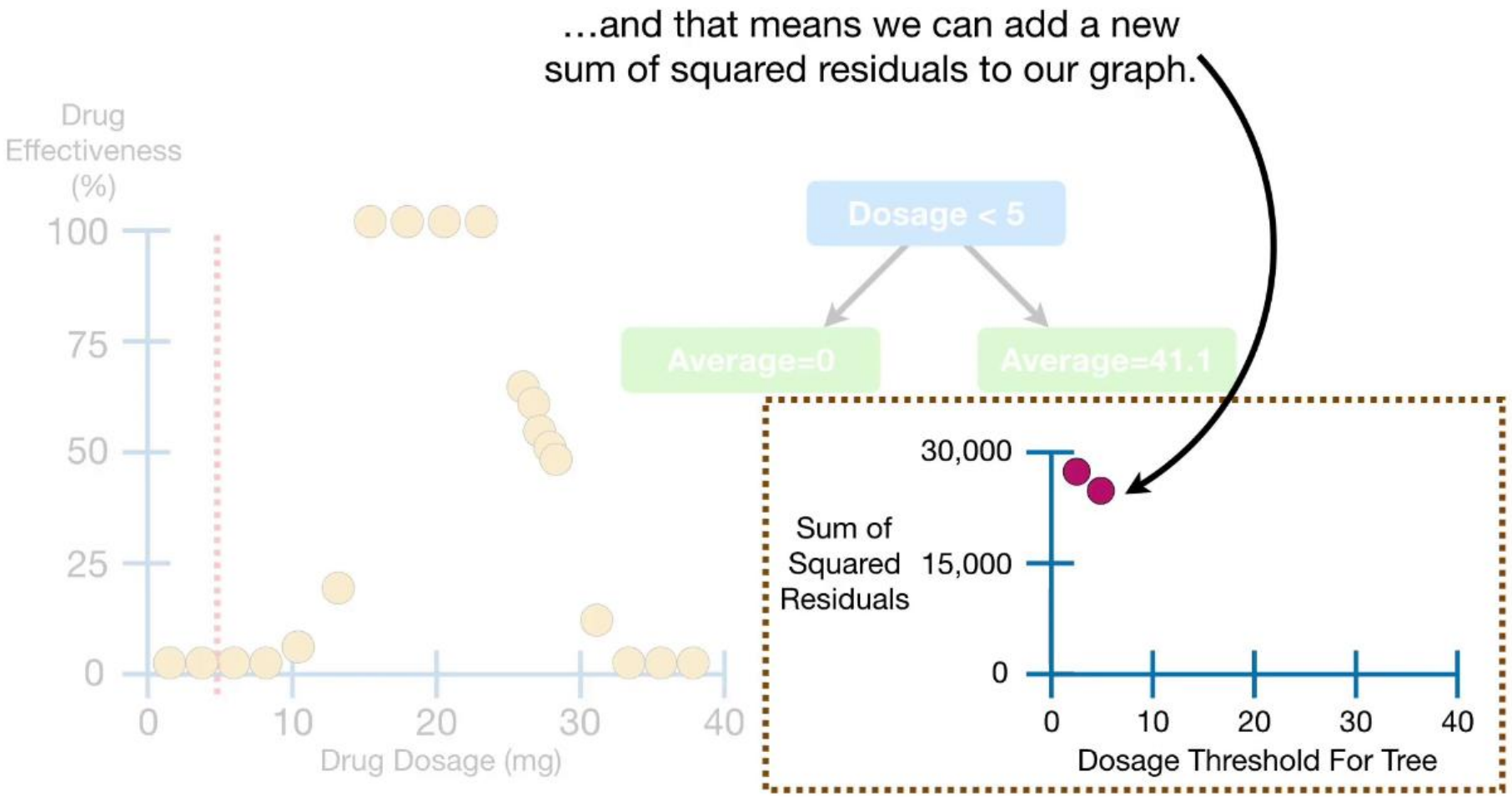
Decision Tree Regression



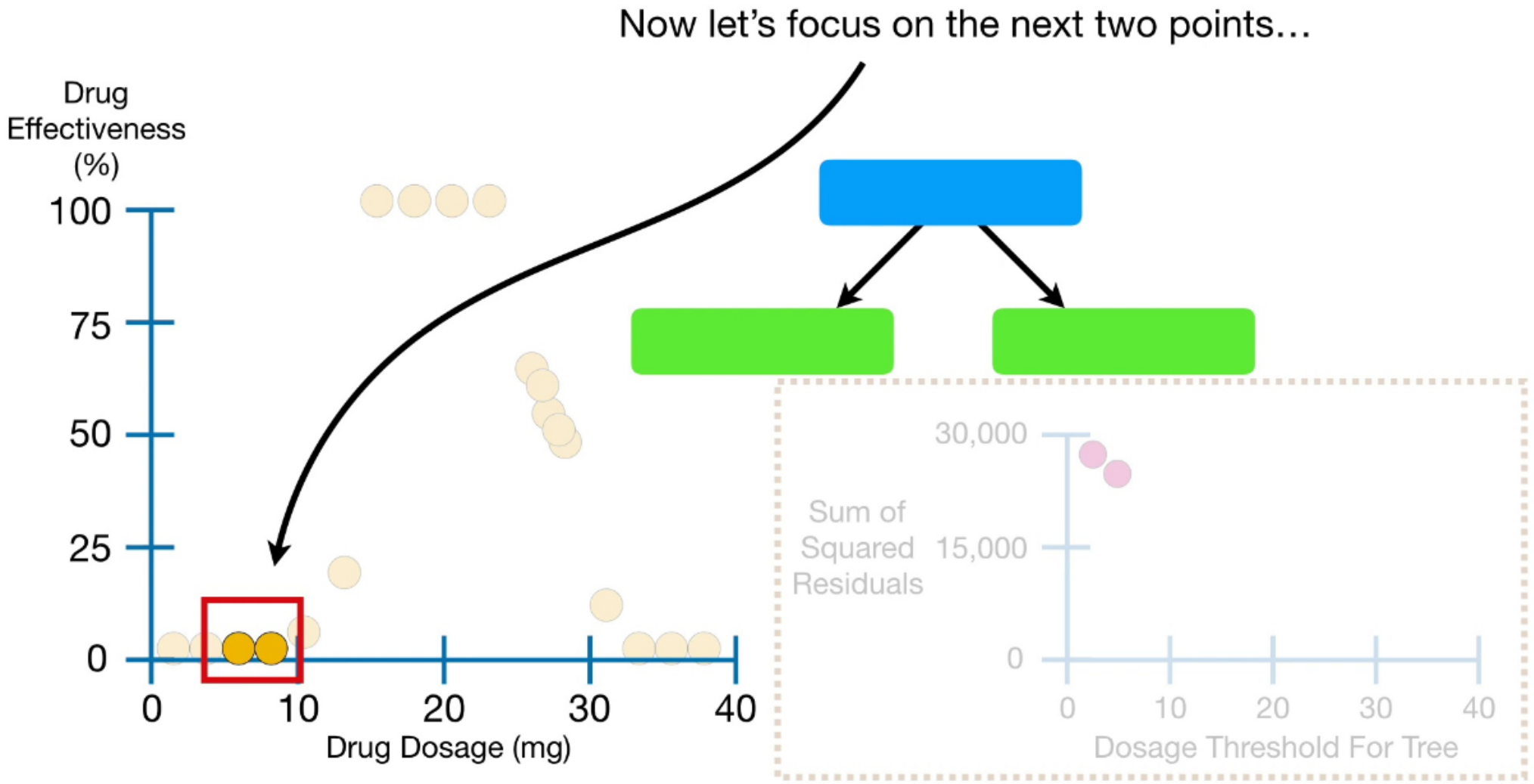
Decision Tree Regression



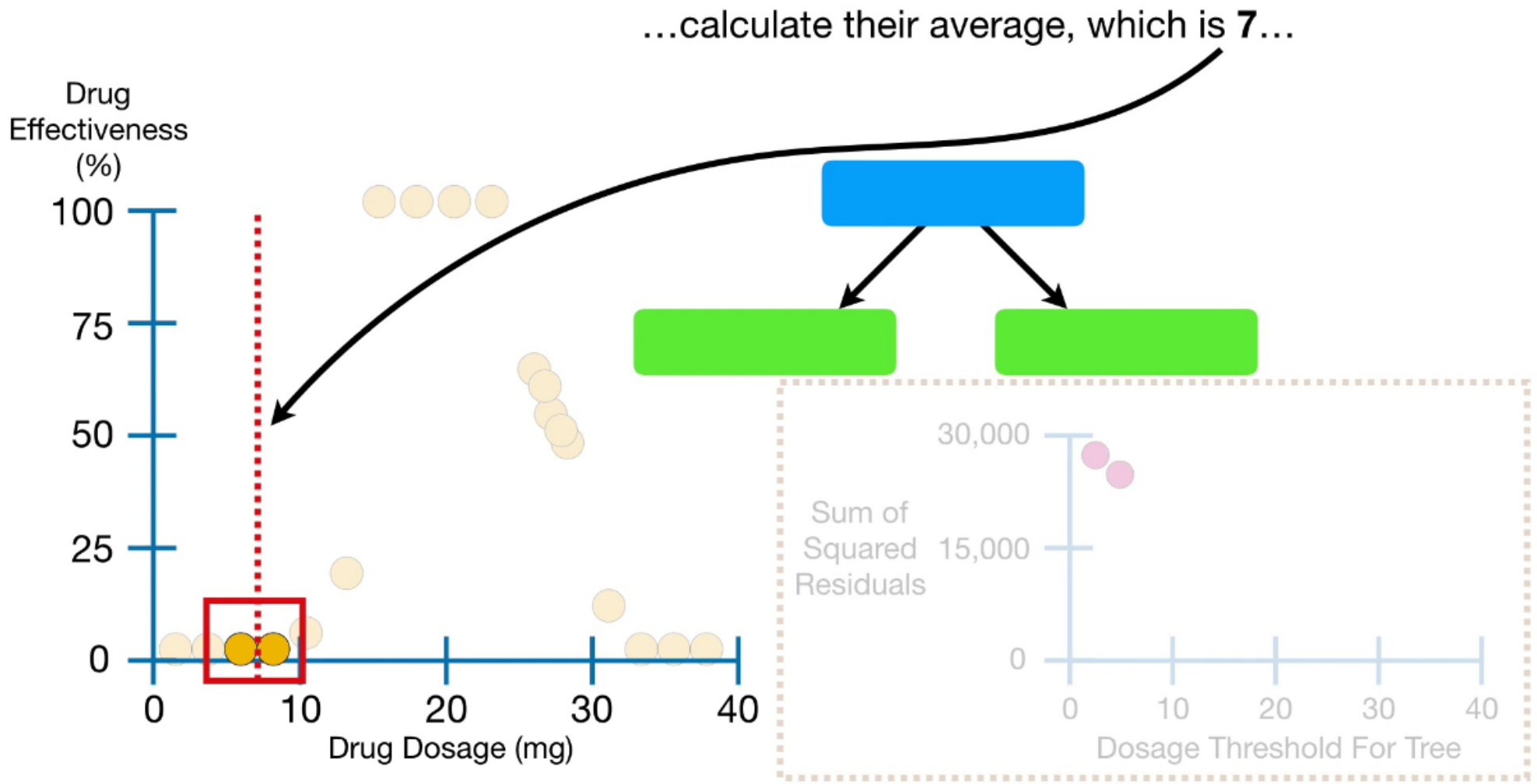
Decision Tree Regression



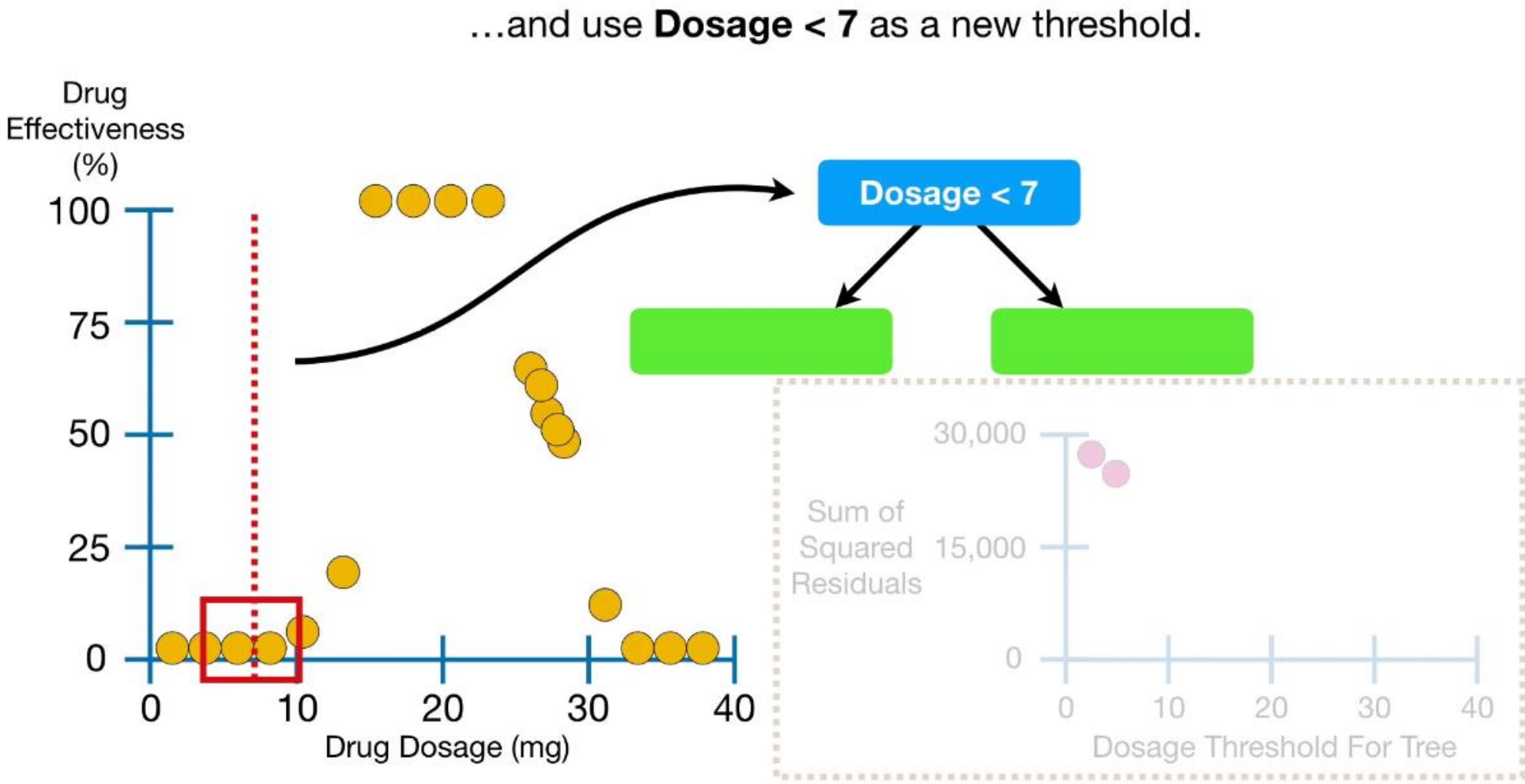
Decision Tree Regression



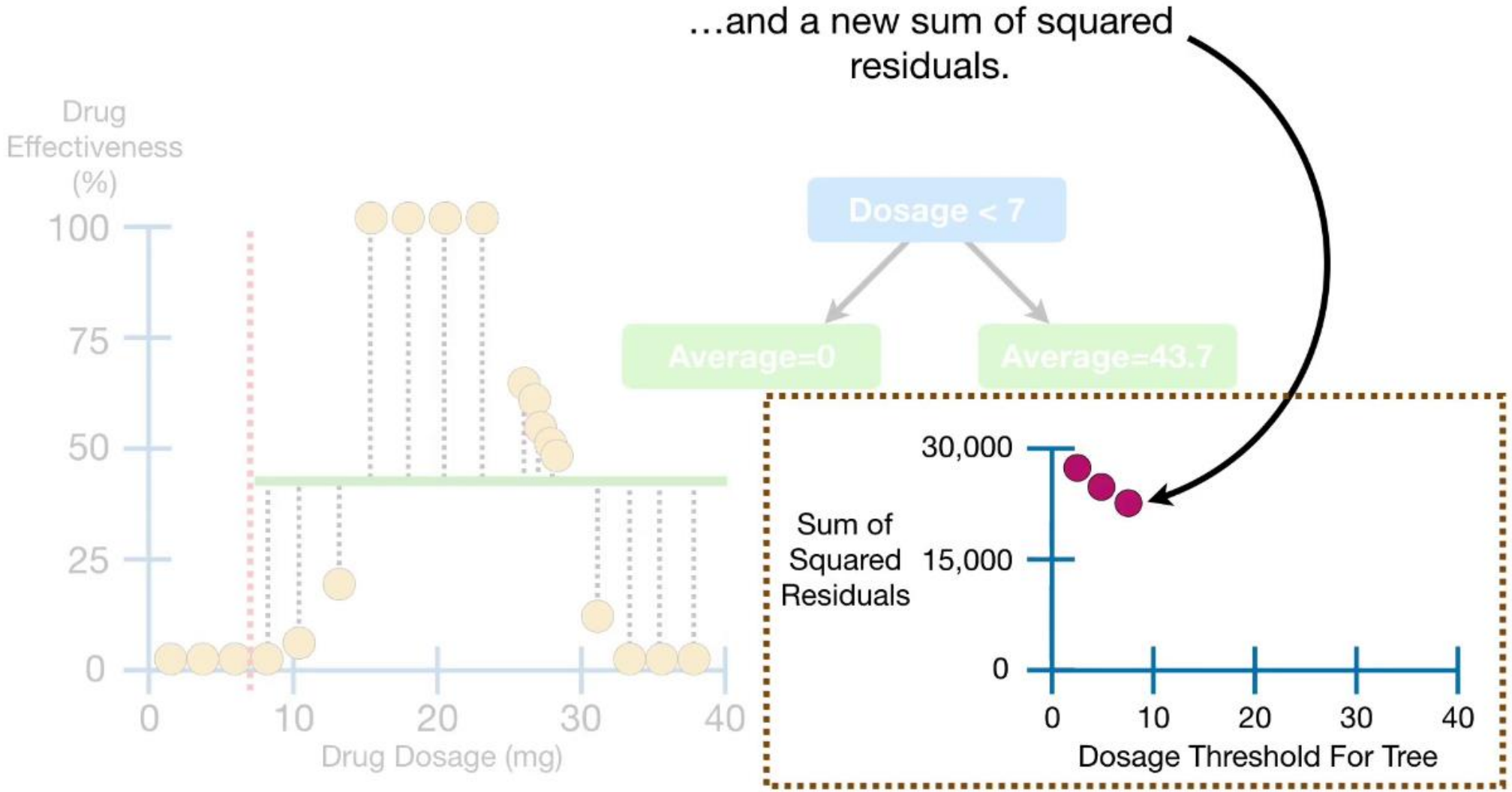
Decision Tree Regression



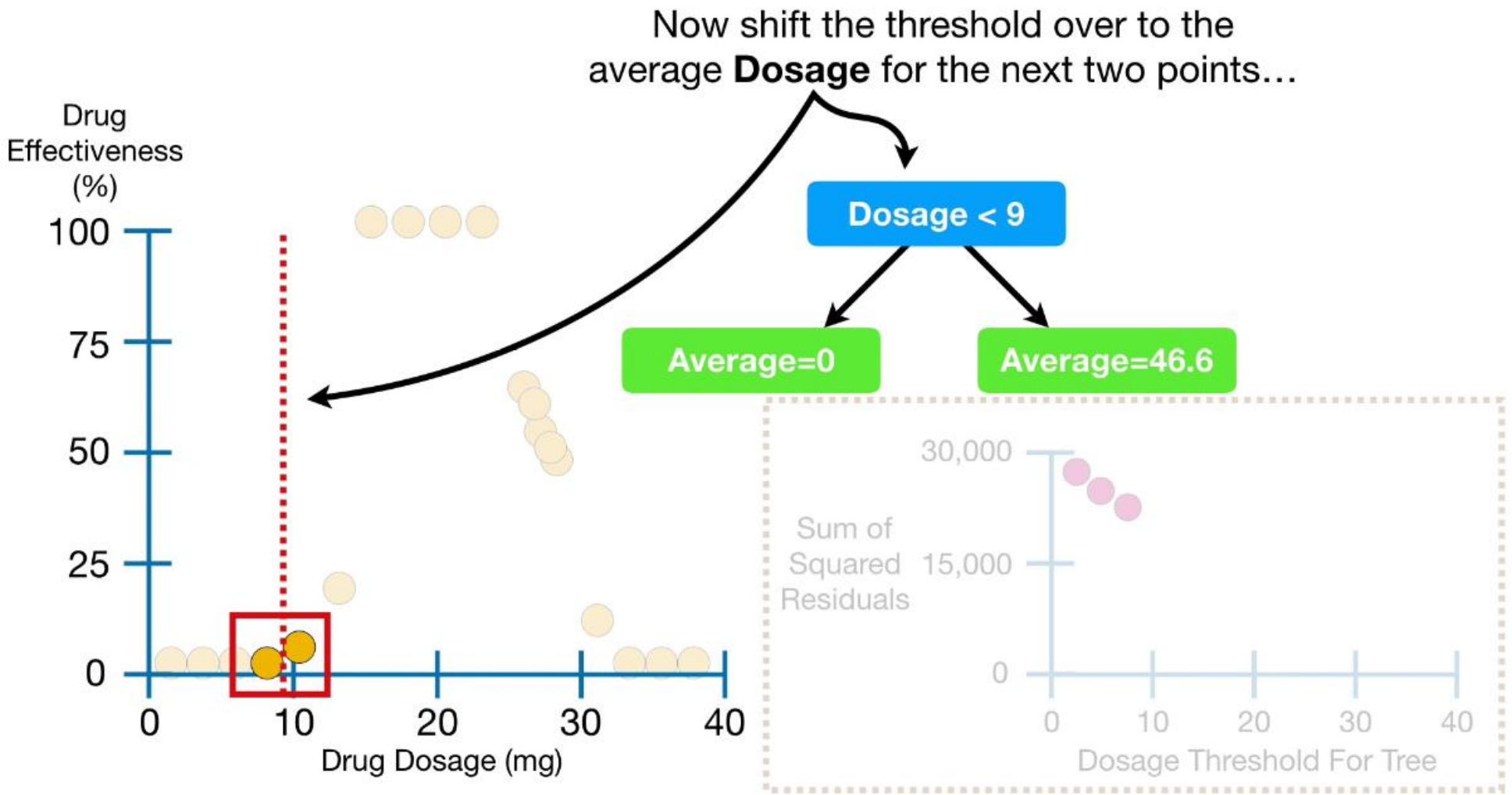
Decision Tree Regression



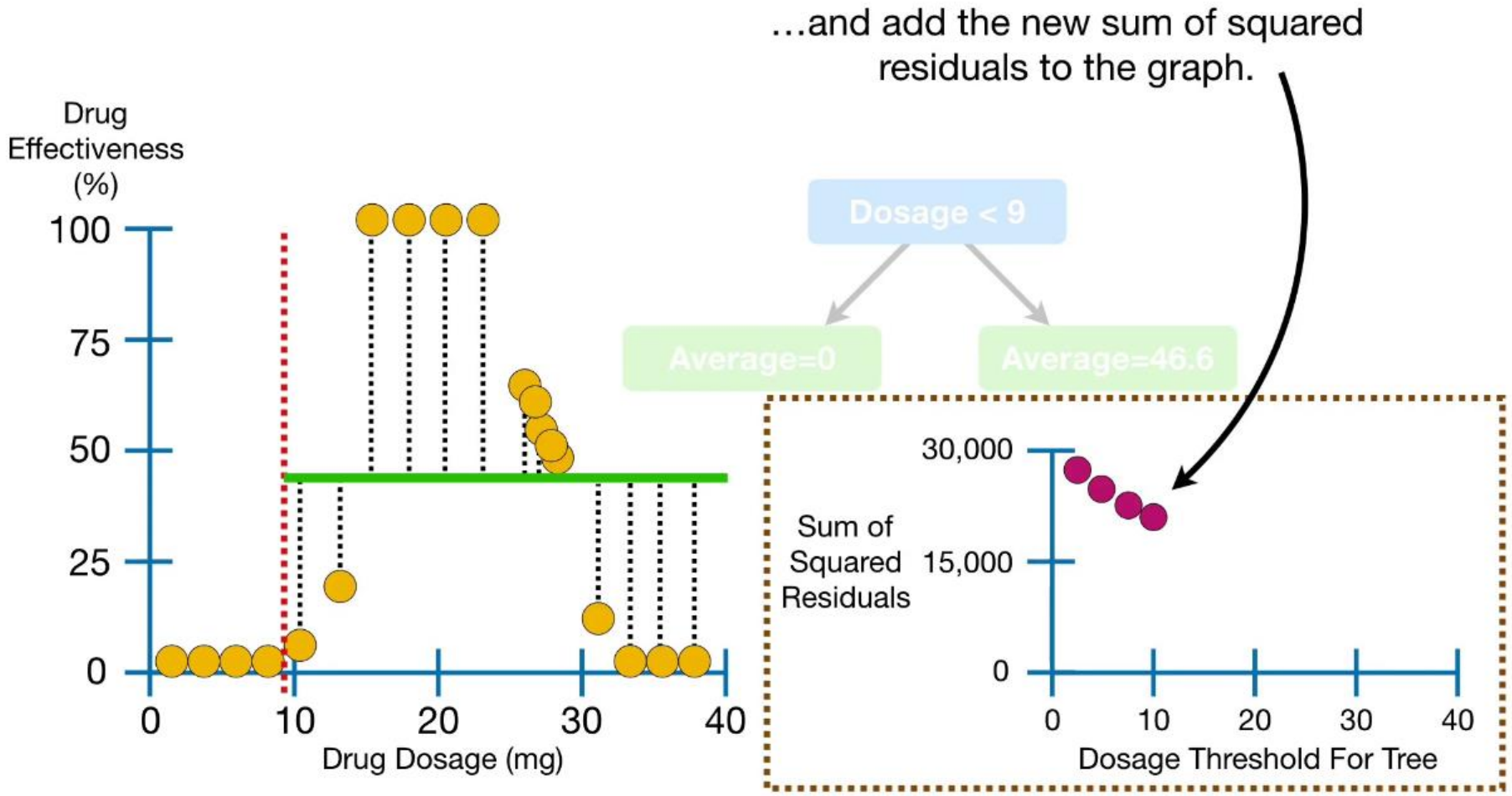
Decision Tree Regression



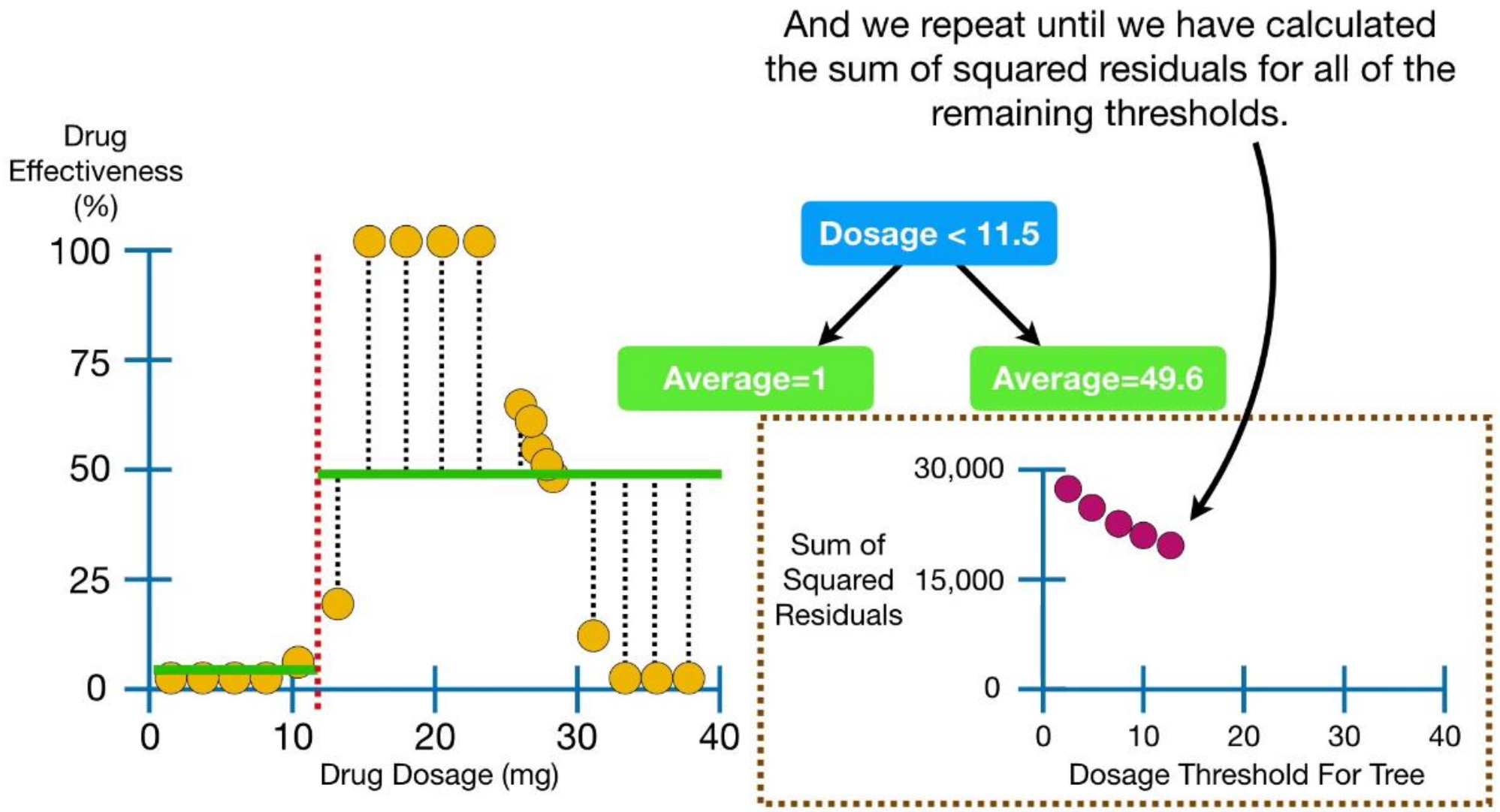
Decision Tree Regression



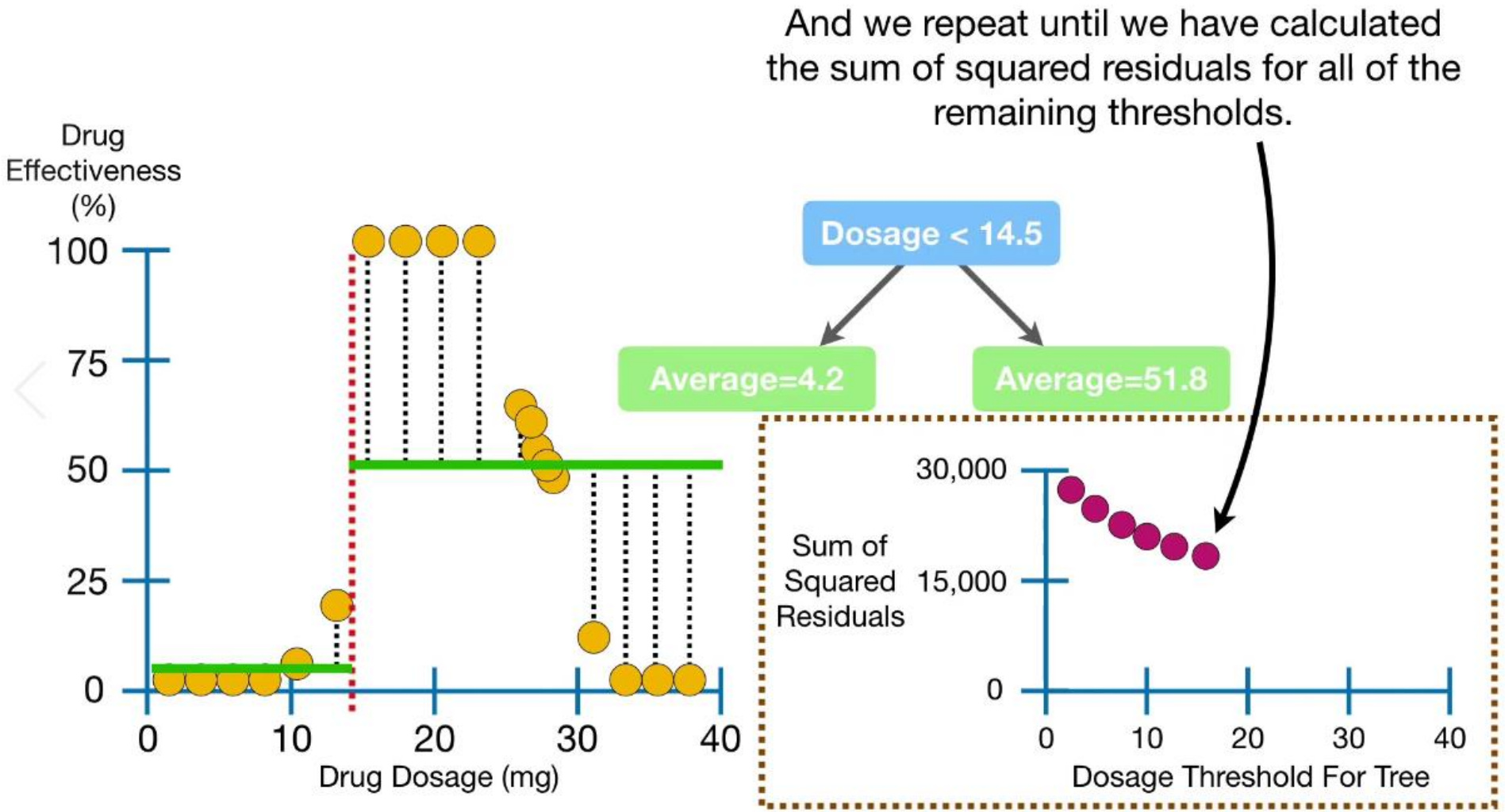
Decision Tree Regression



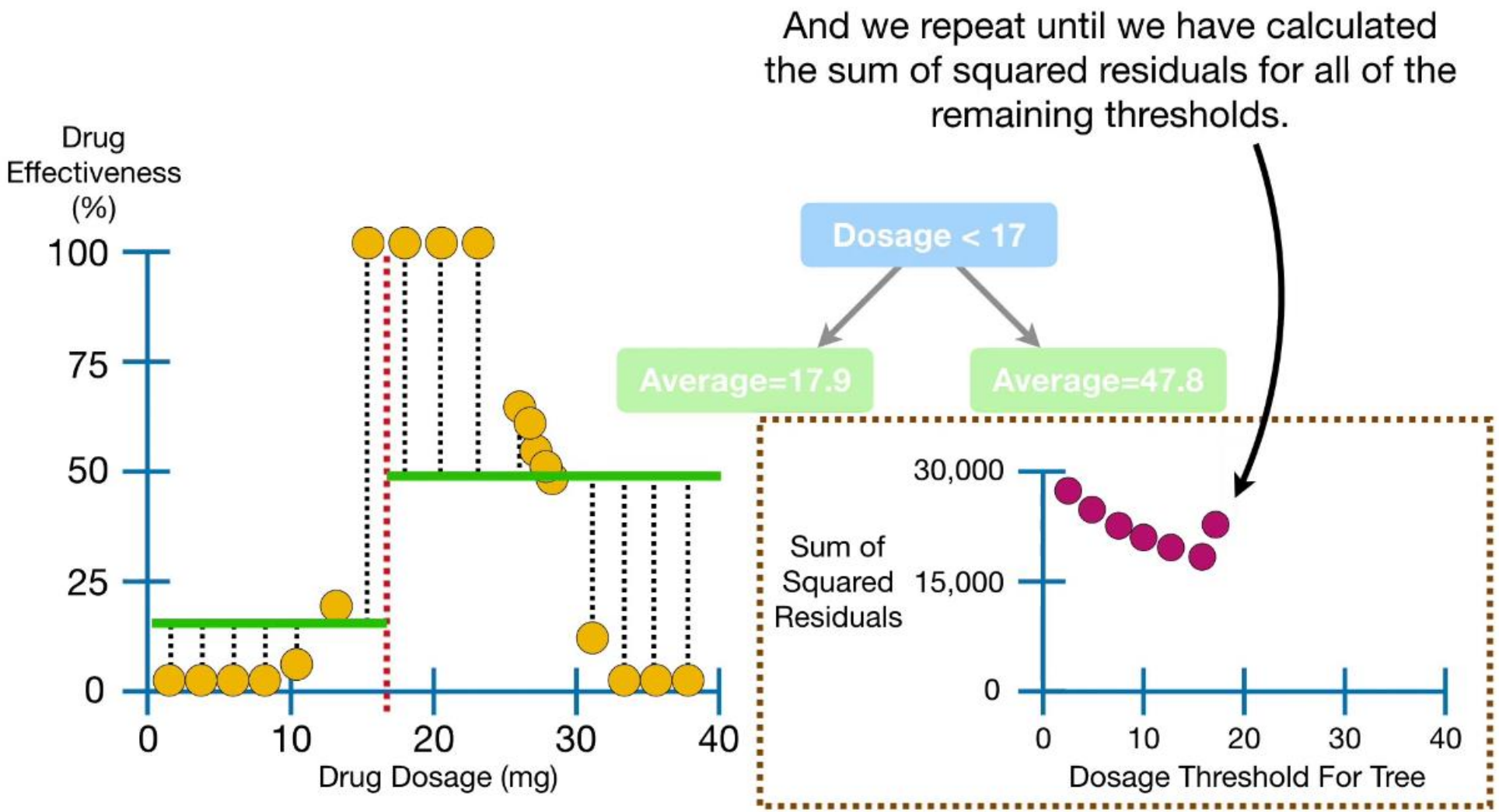
Decision Tree Regression



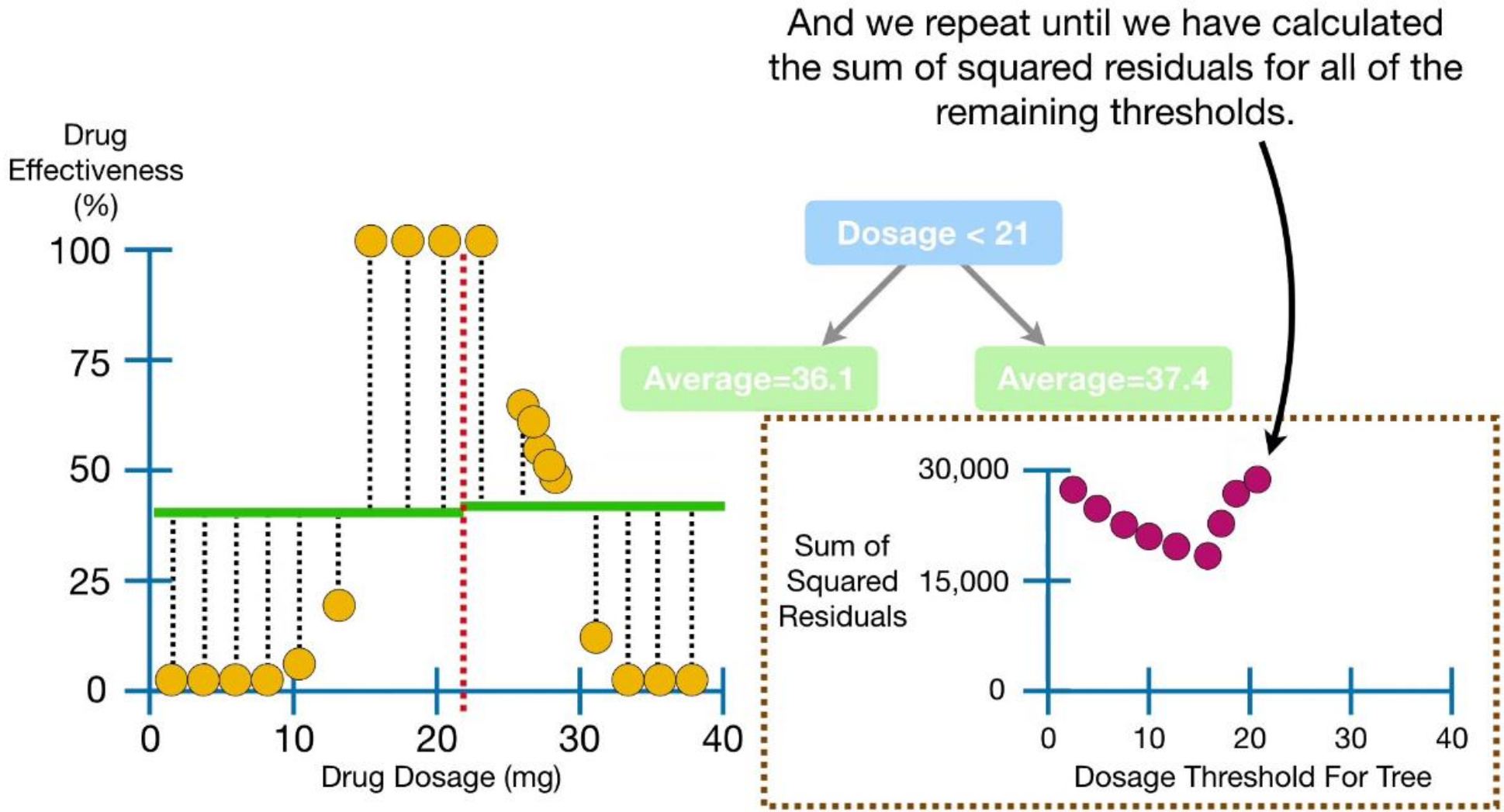
Decision Tree Regression



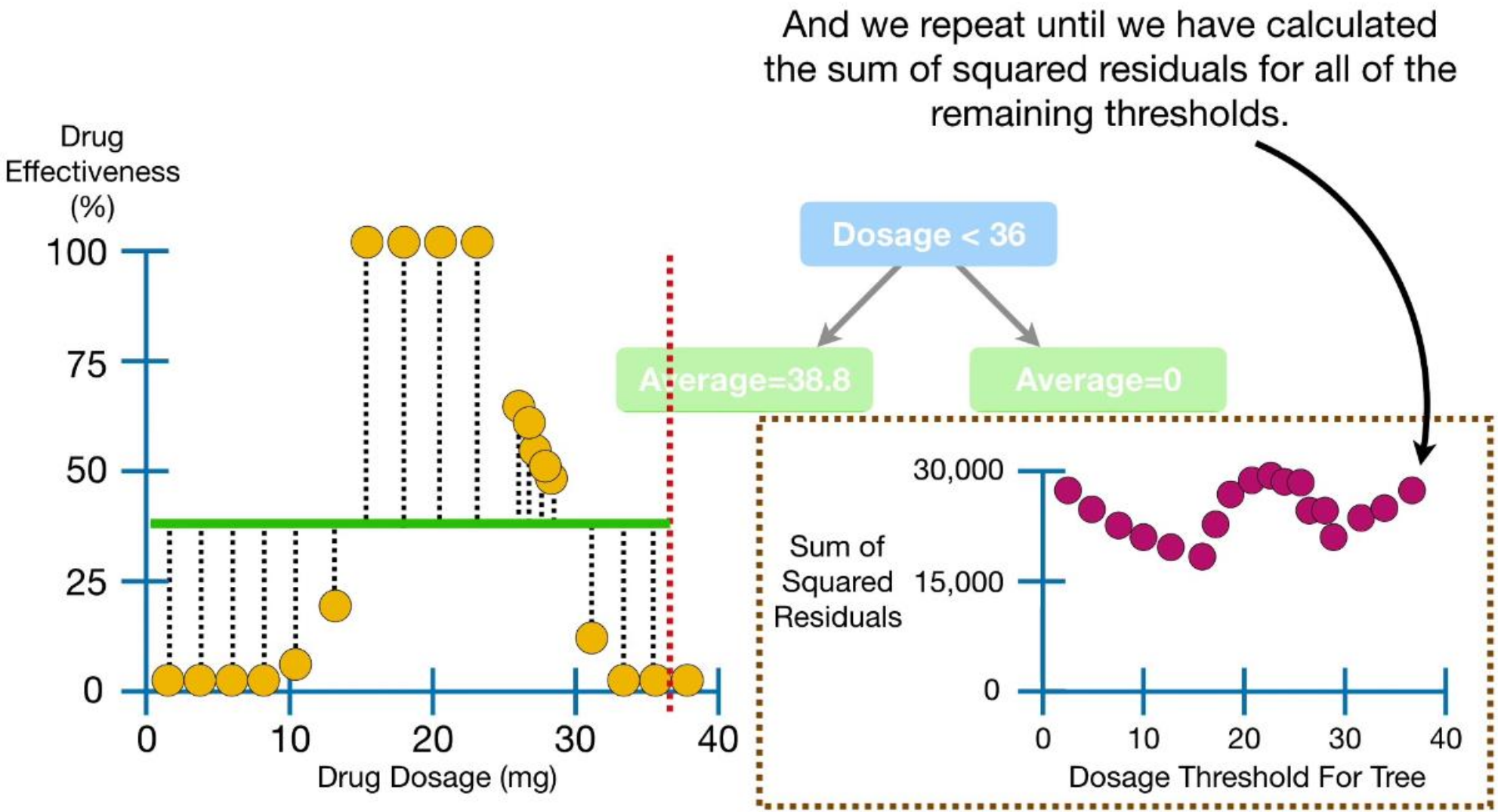
Decision Tree Regression



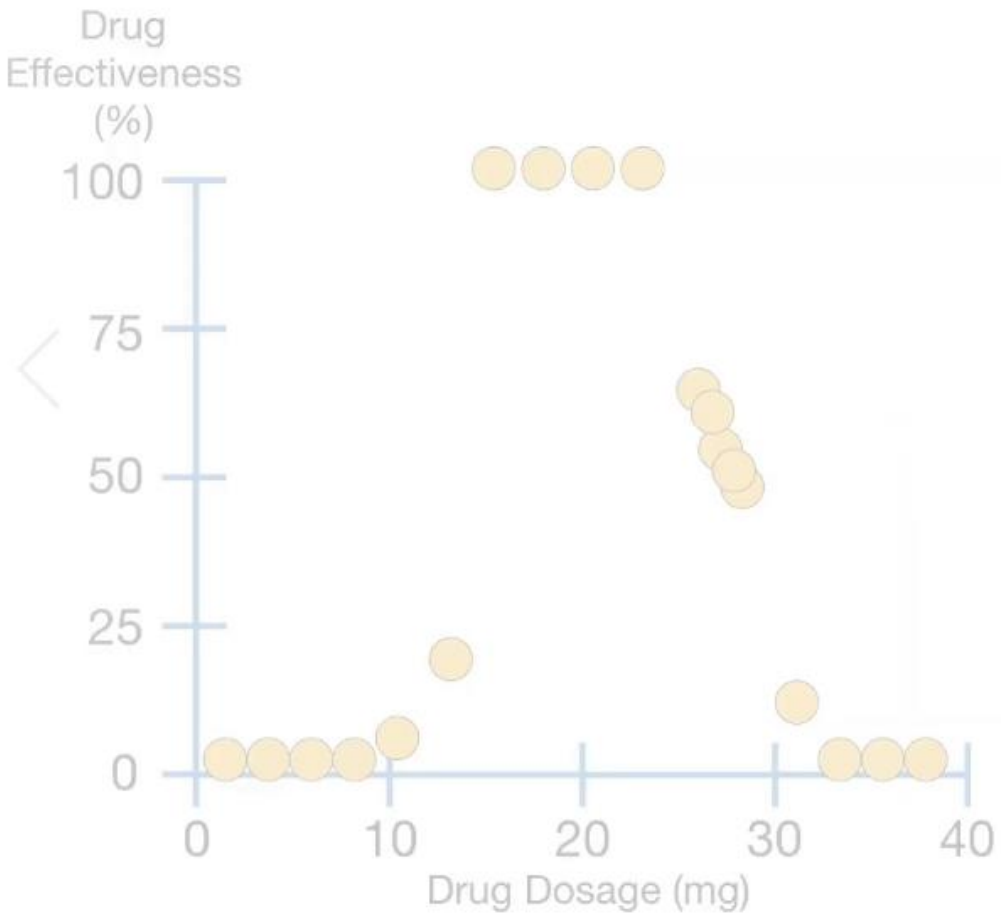
Decision Tree Regression



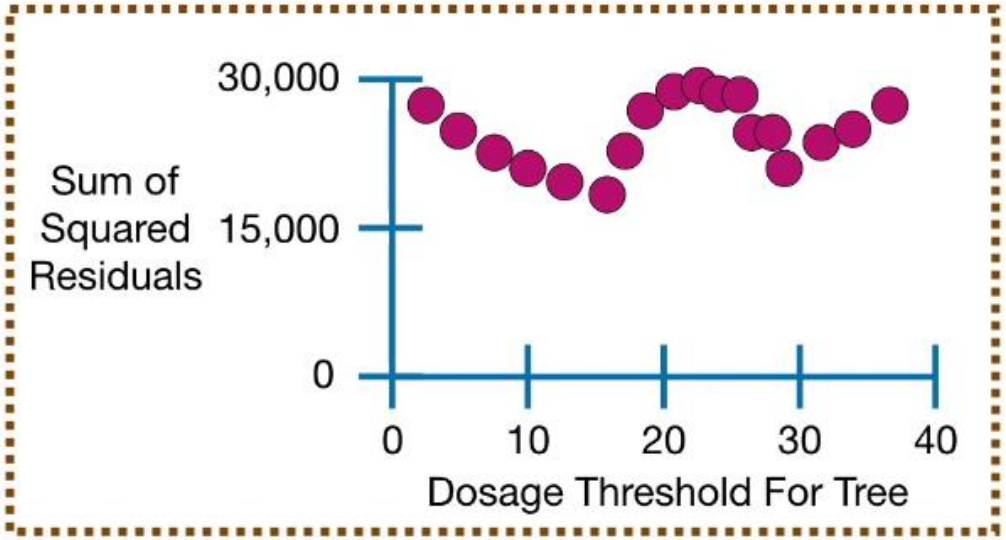
Decision Tree Regression



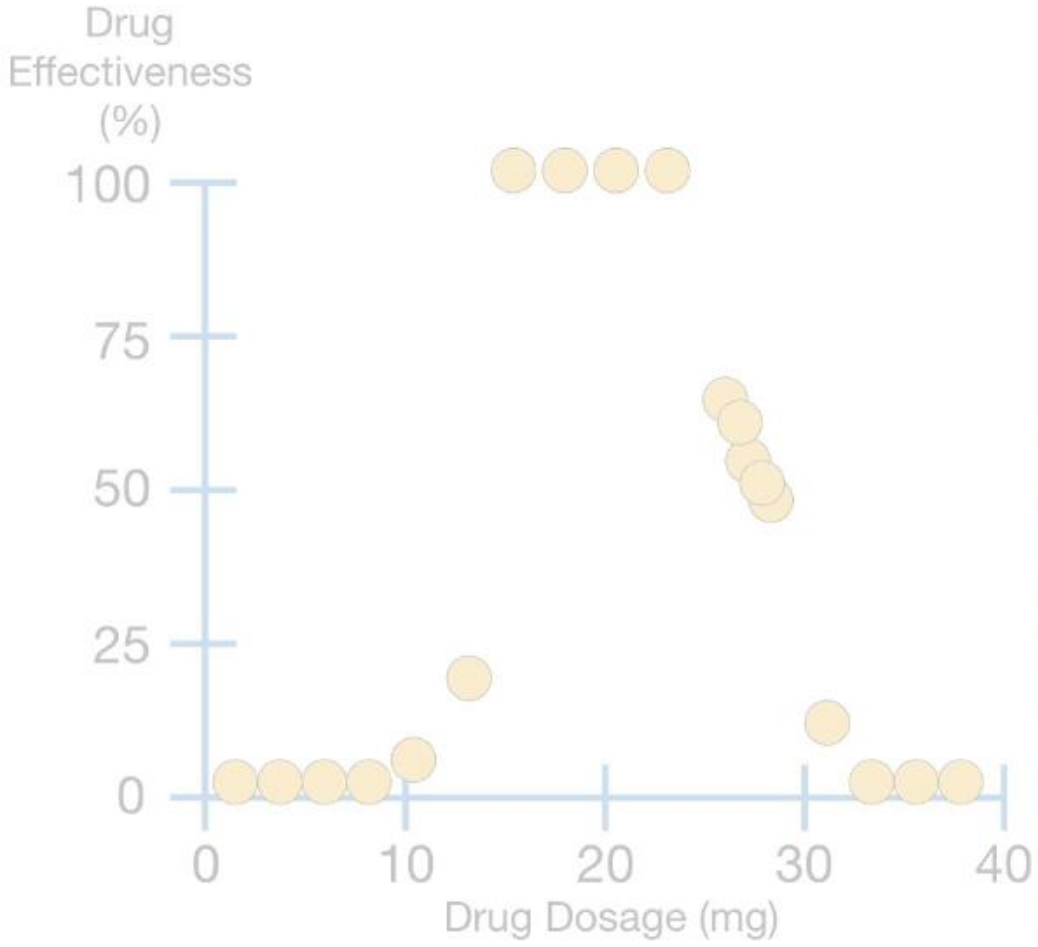
Decision Tree Regression

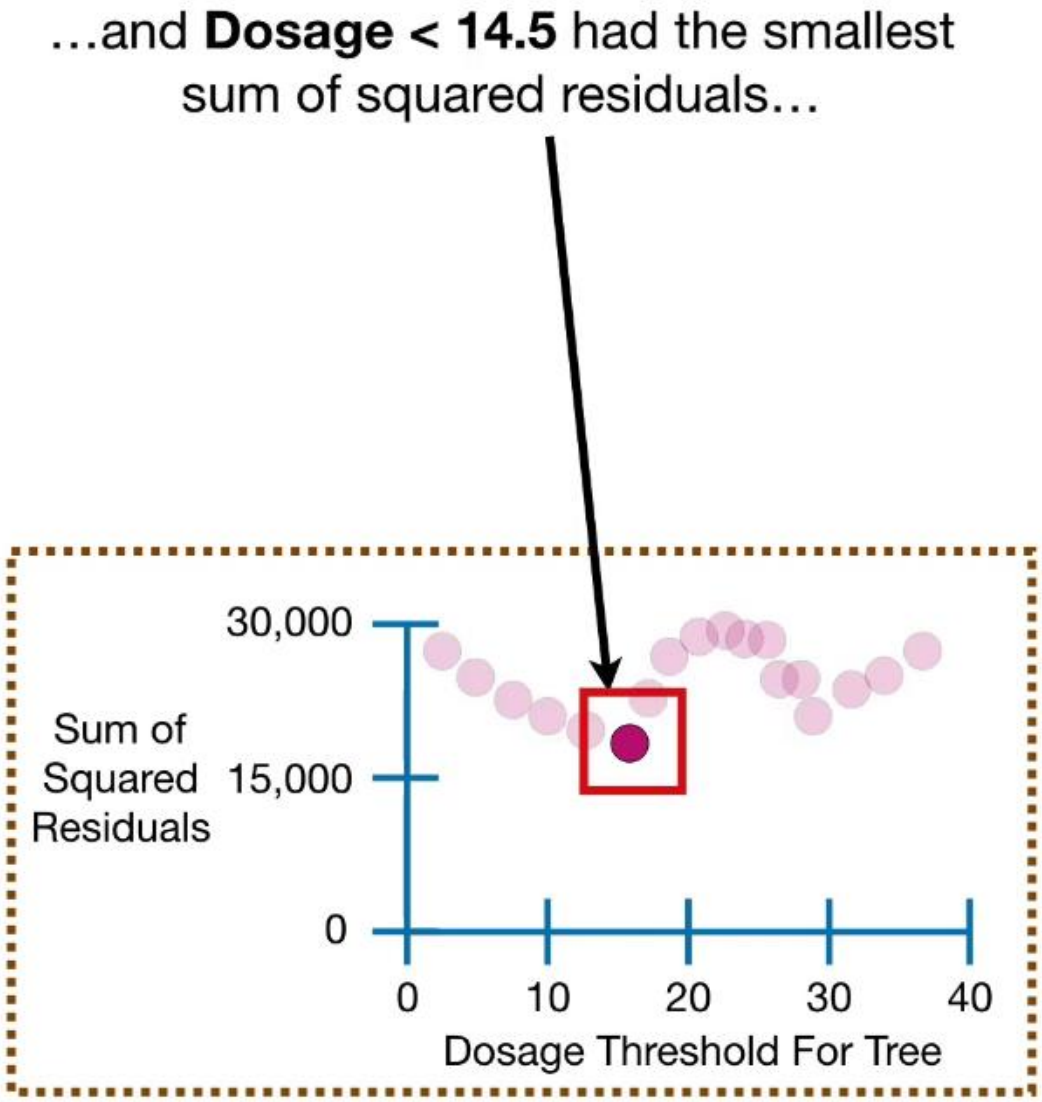


Now we can see the sum of squared residuals for all of the thresholds…

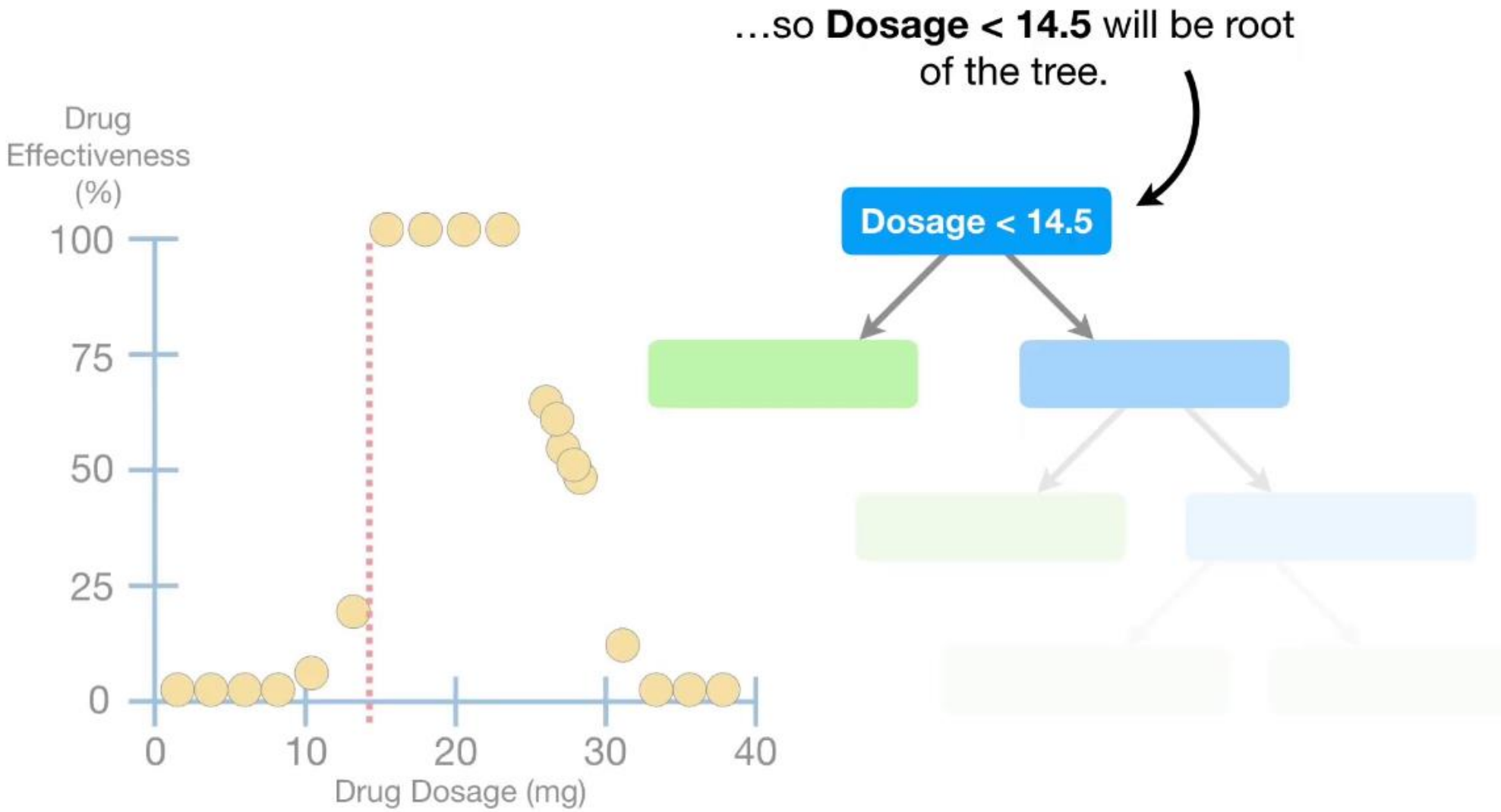


Decision Tree Regression

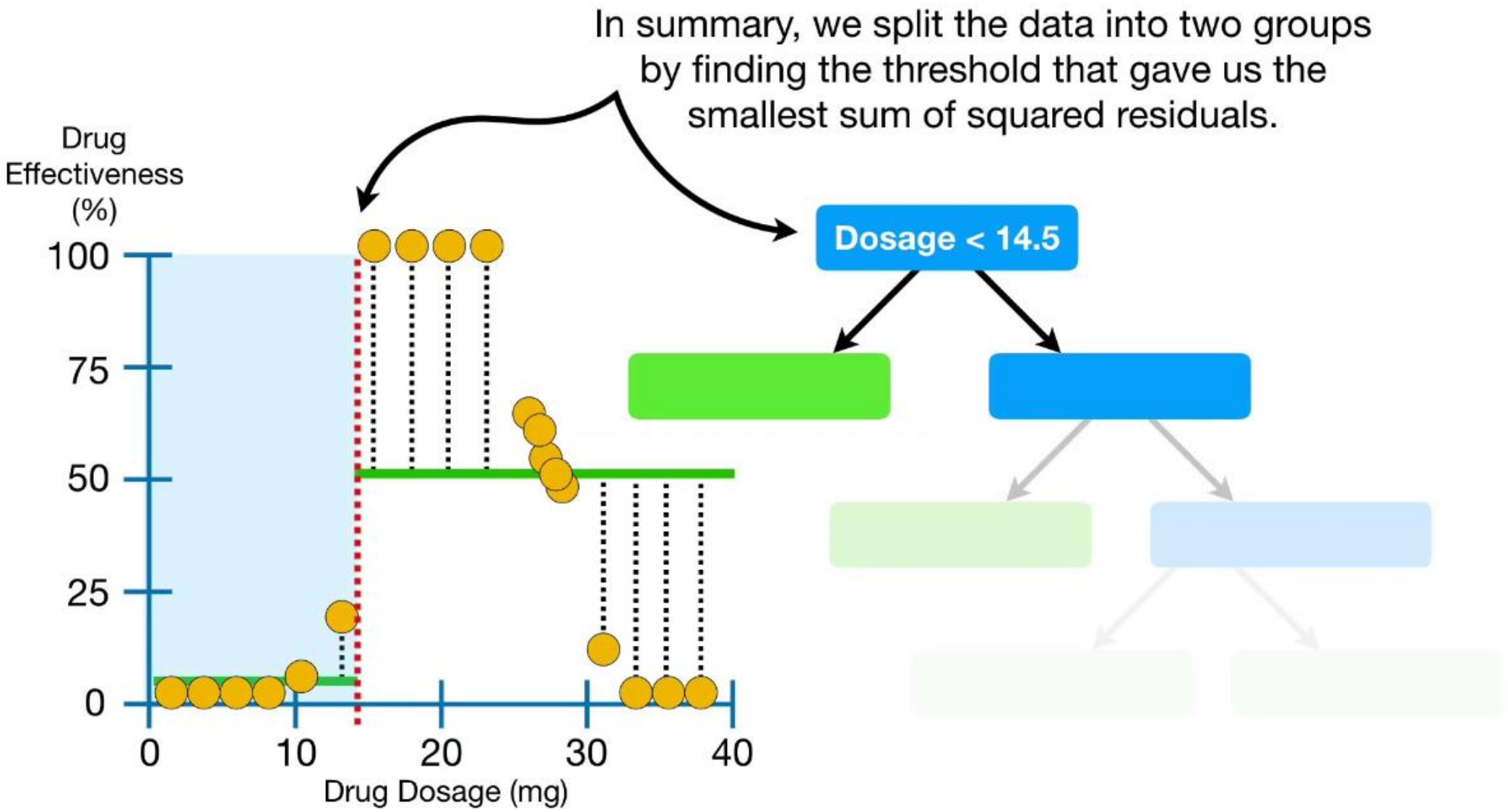




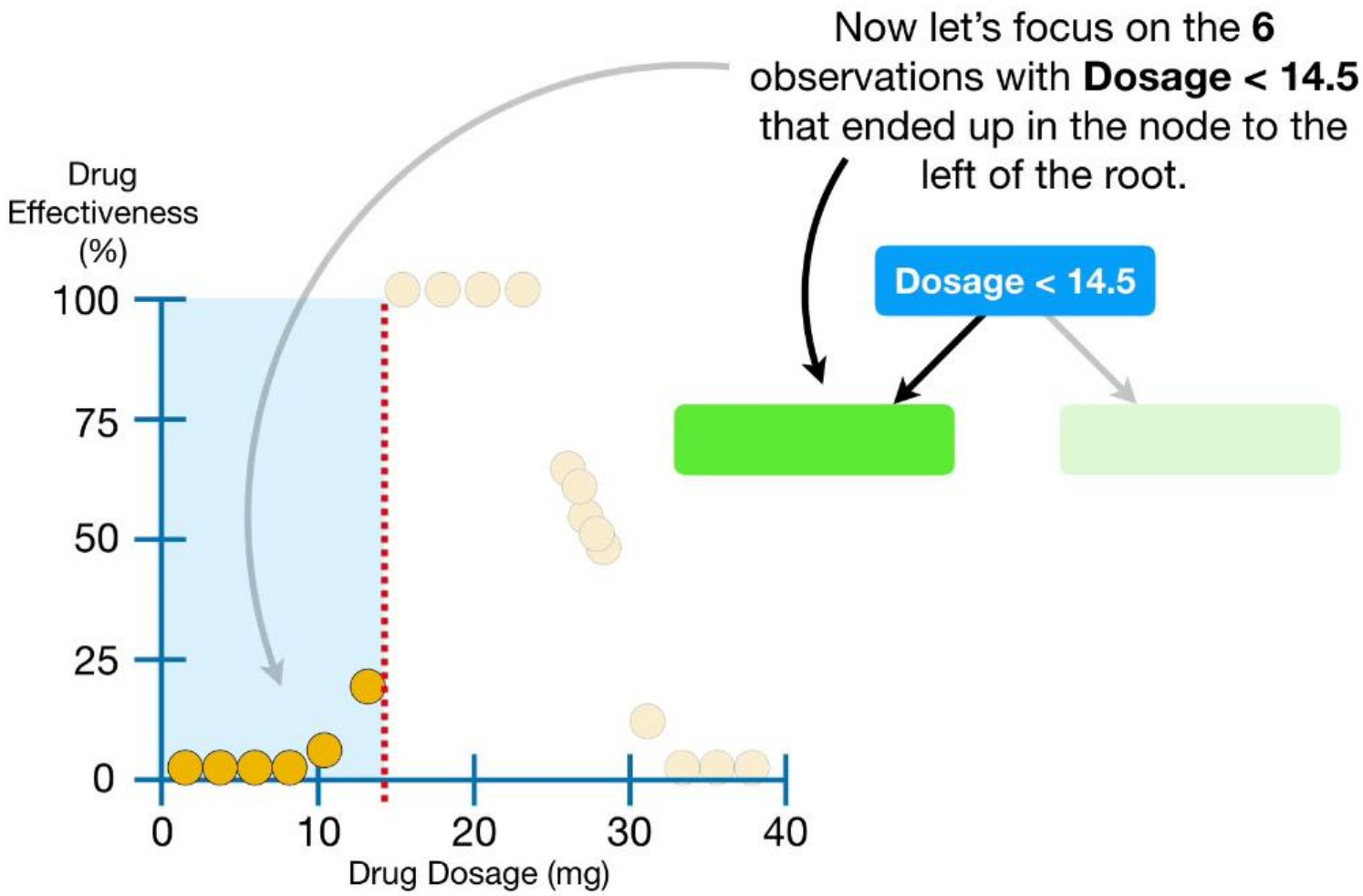
Decision Tree Regression



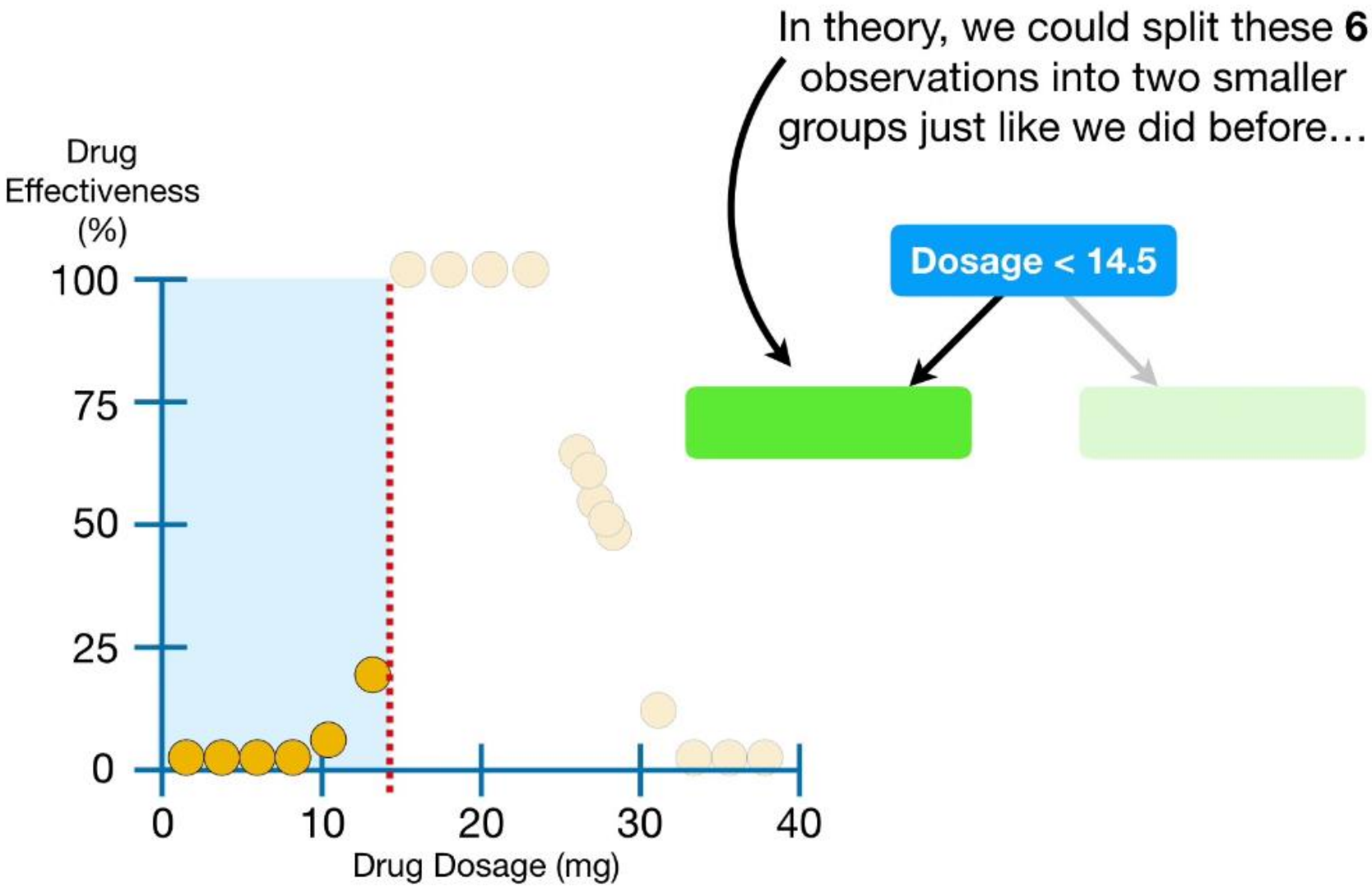
Decision Tree Regression



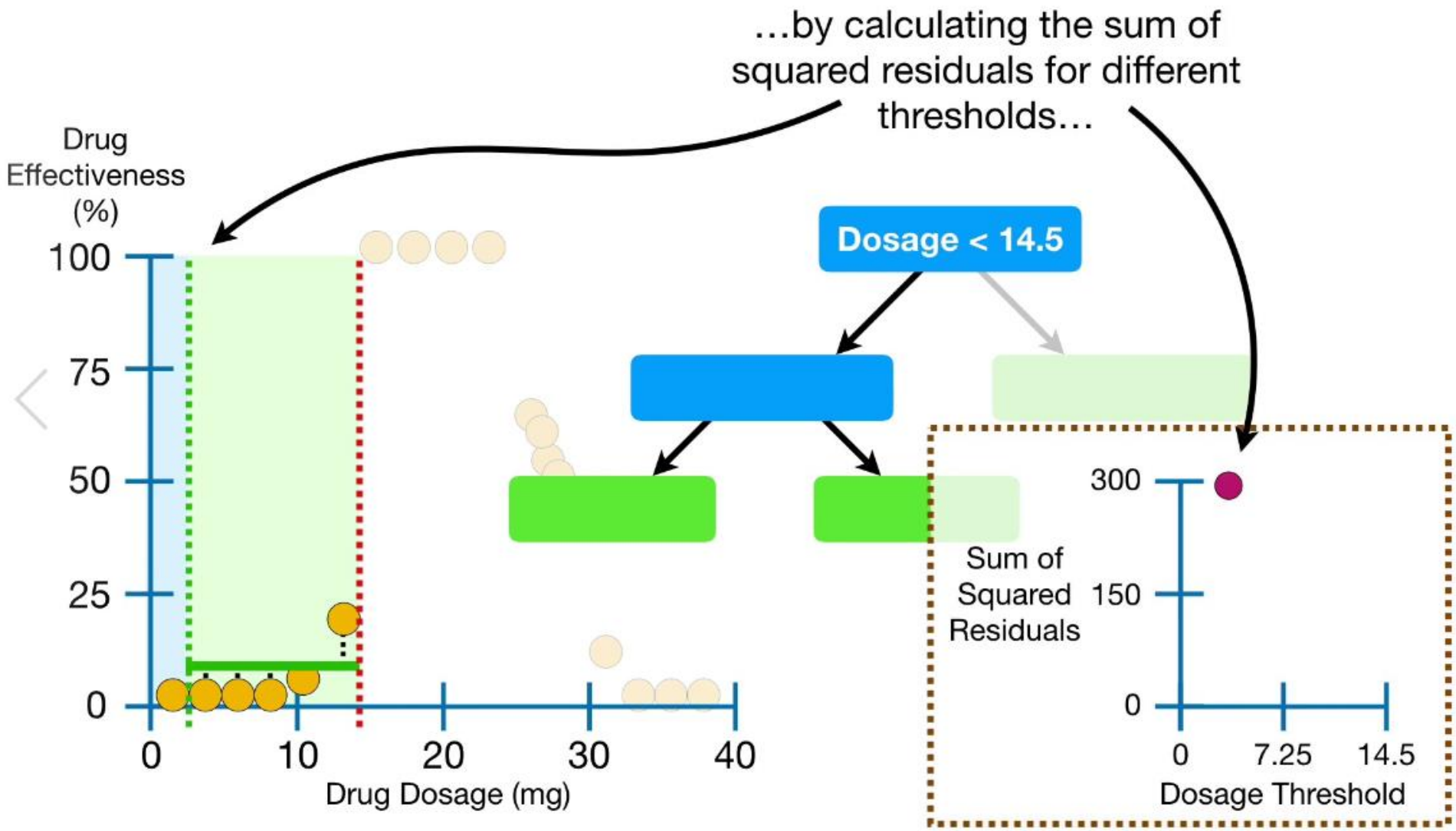
Decision Tree Regression



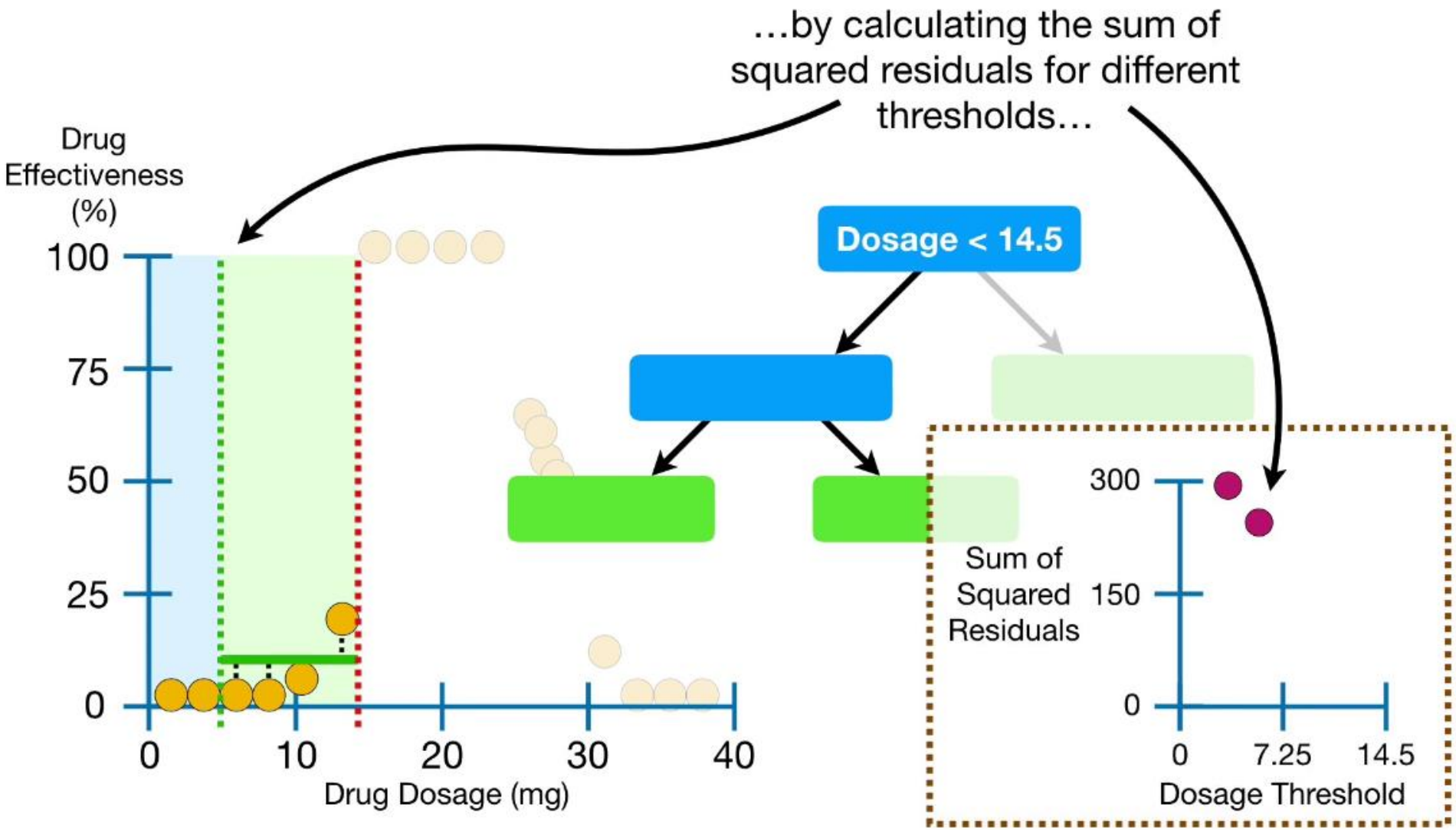
Decision Tree Regression



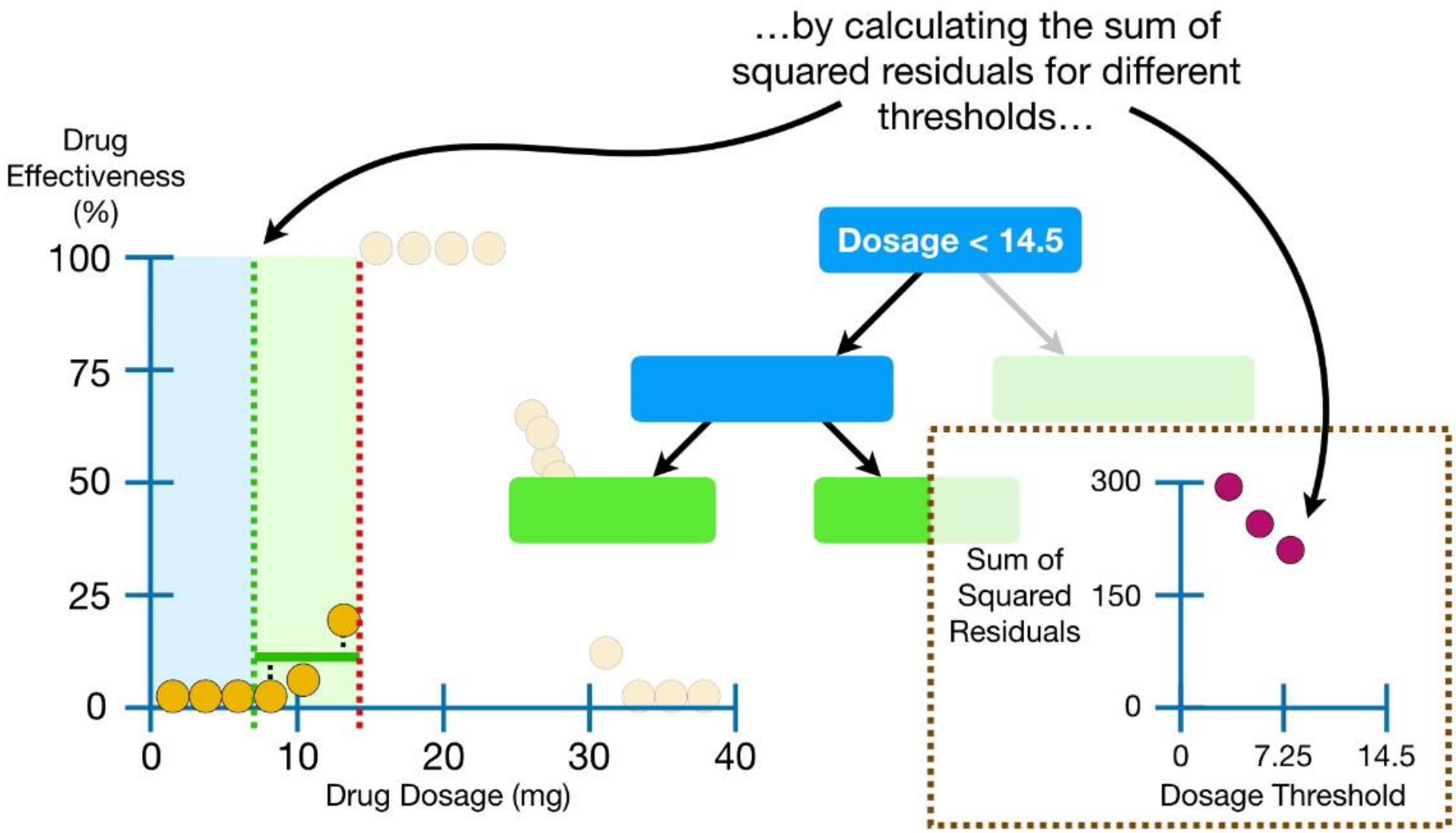
Decision Tree Regression



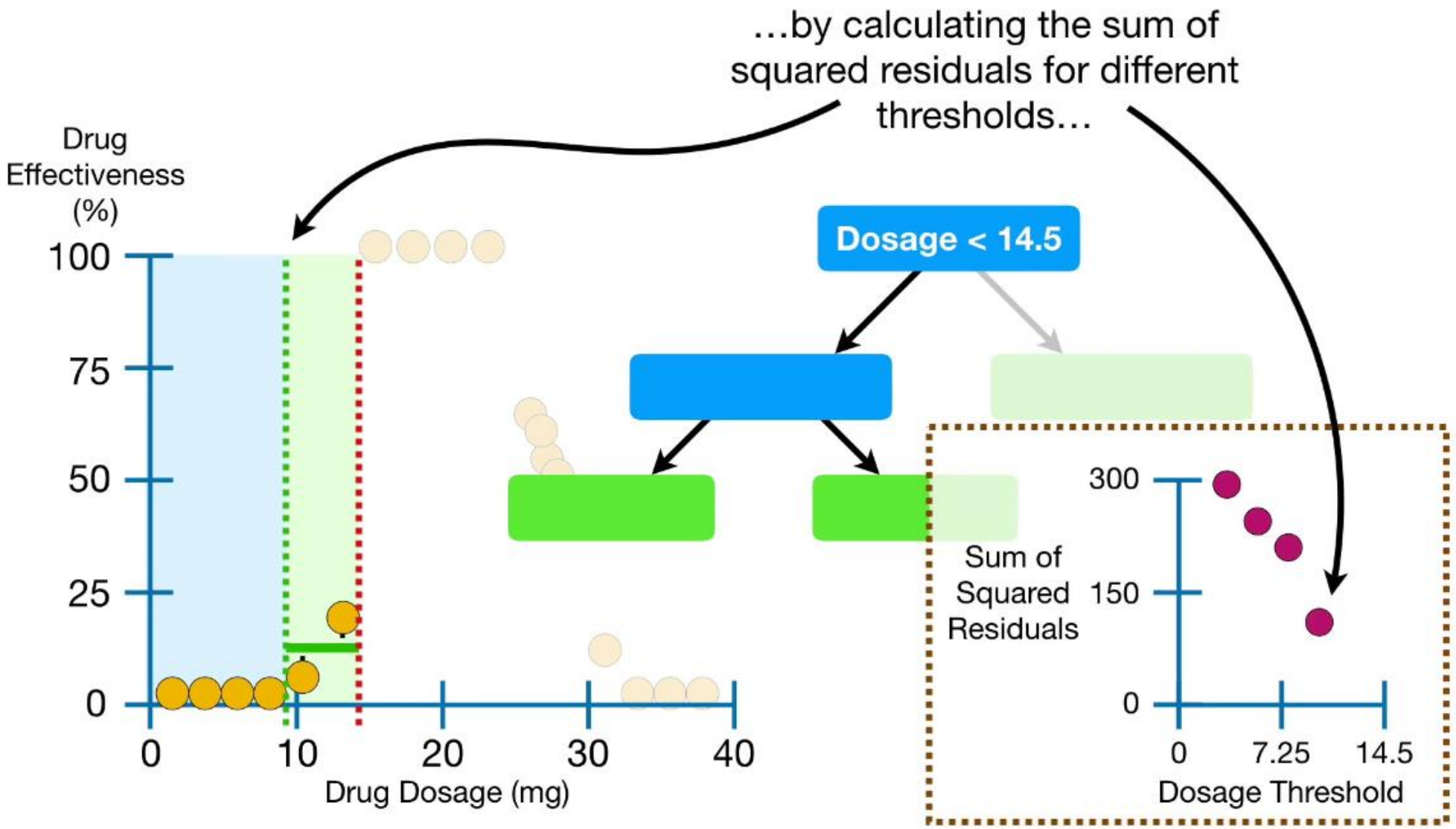
Decision Tree Regression



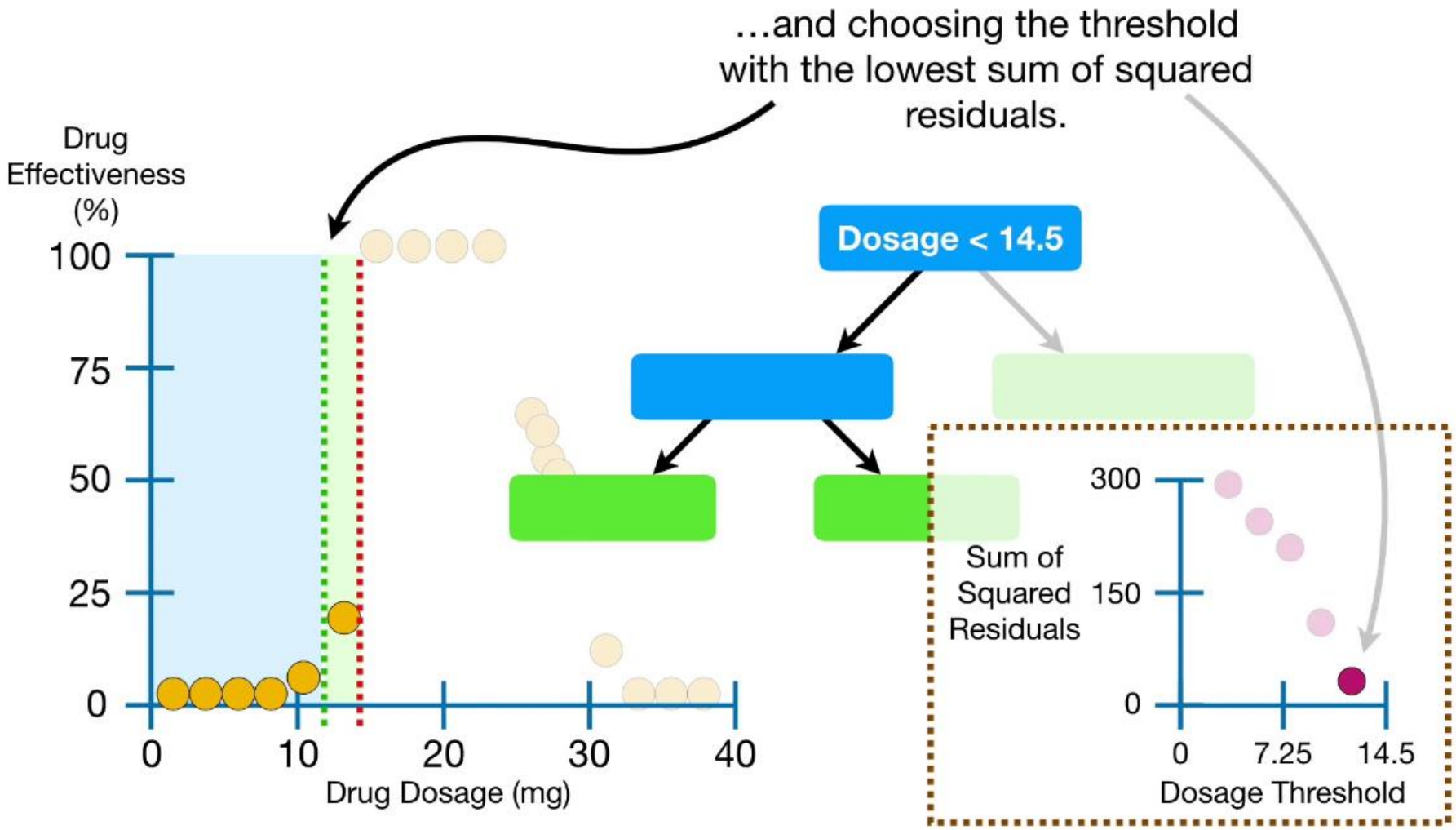
Decision Tree Regression



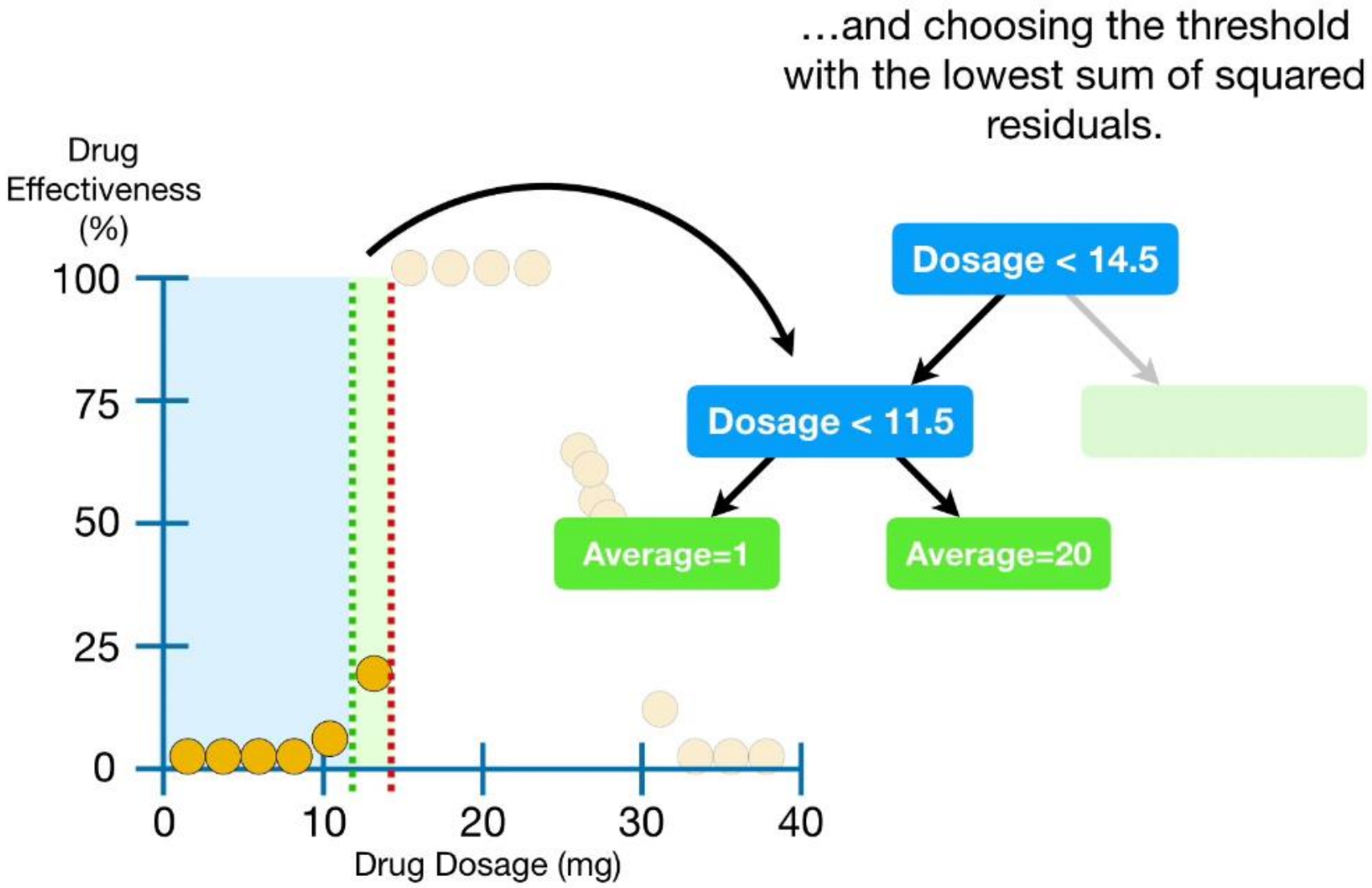
Decision Tree Regression



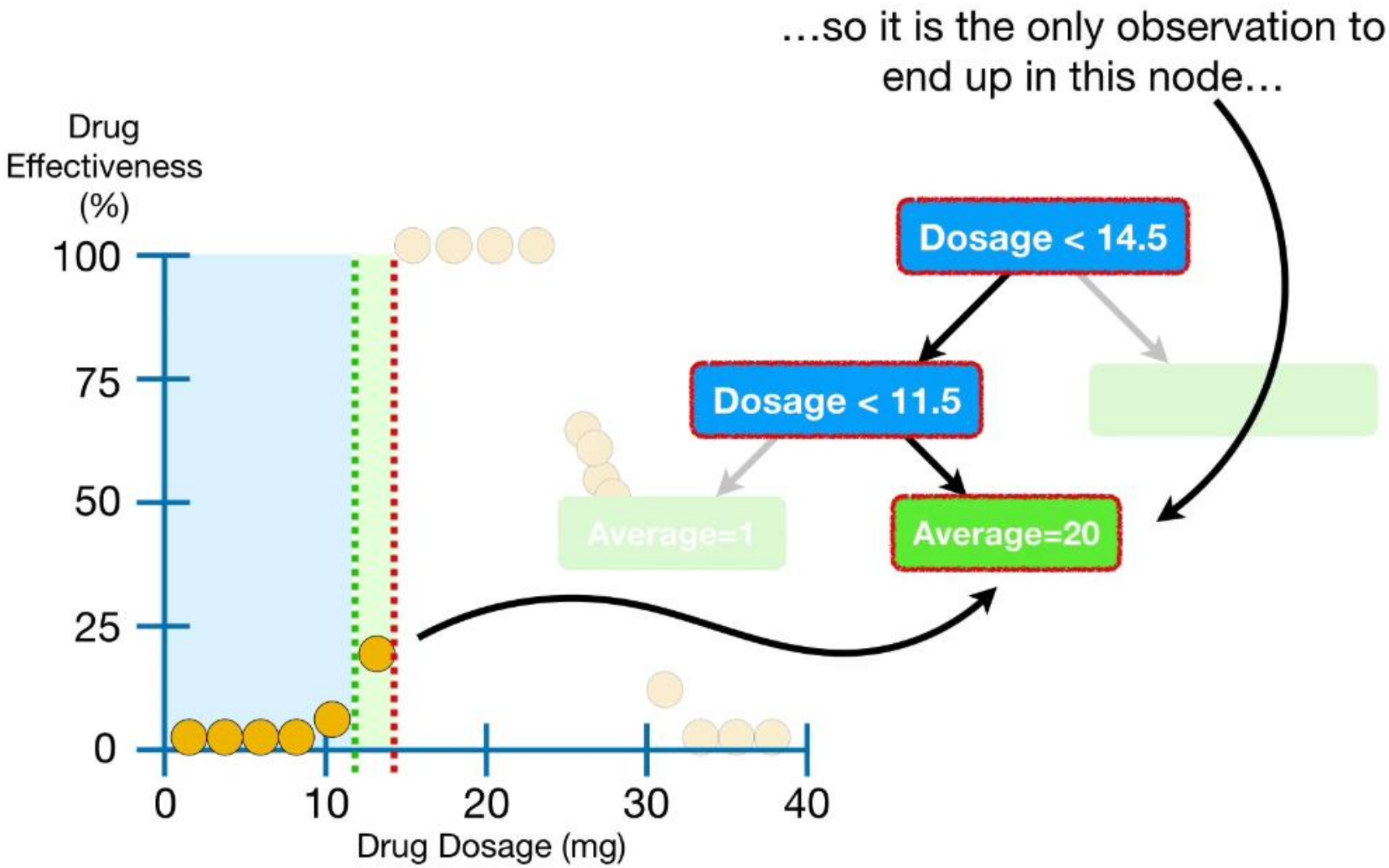
Decision Tree Regression



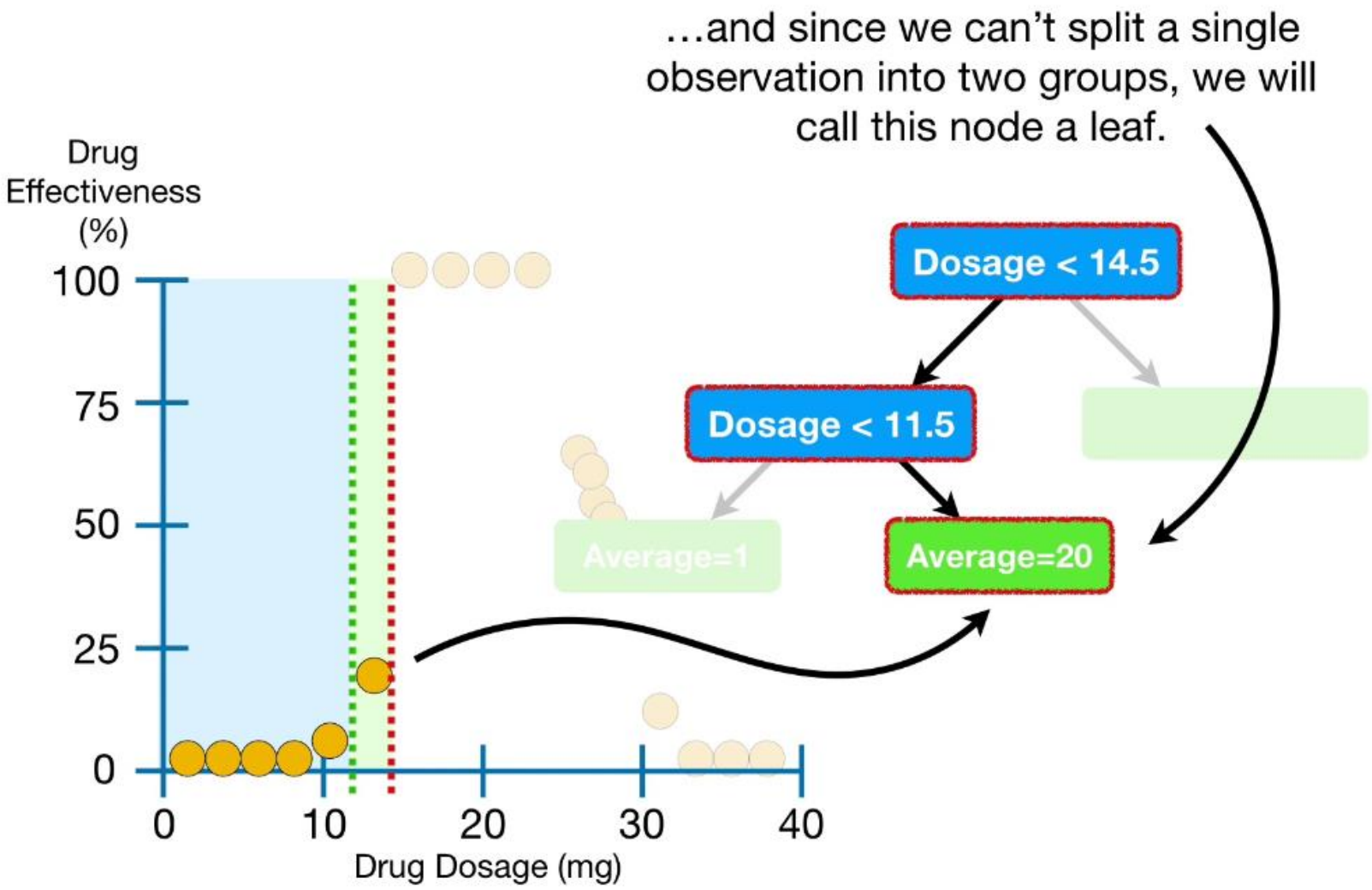
Decision Tree Regression



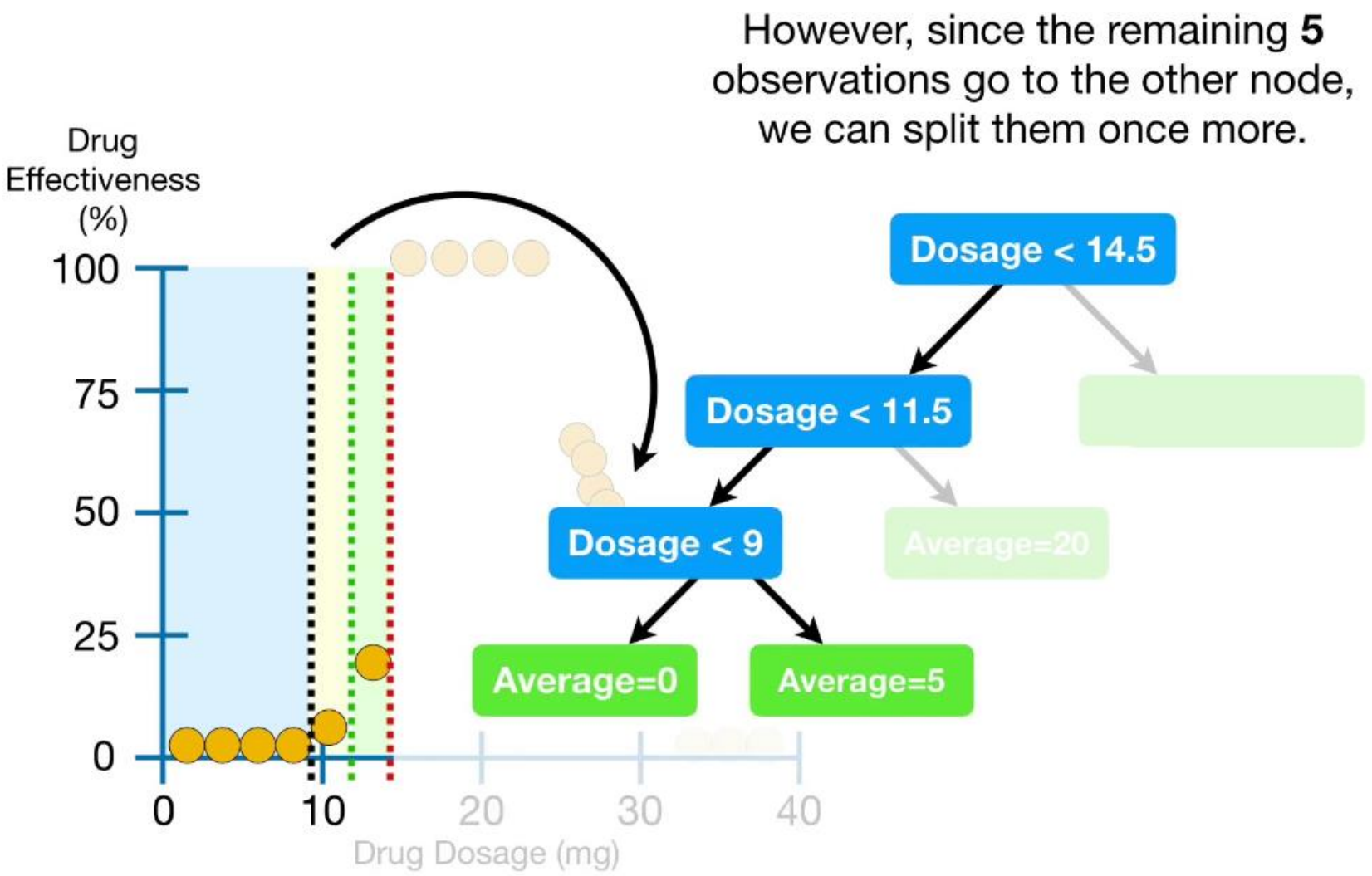
Decision Tree Regression



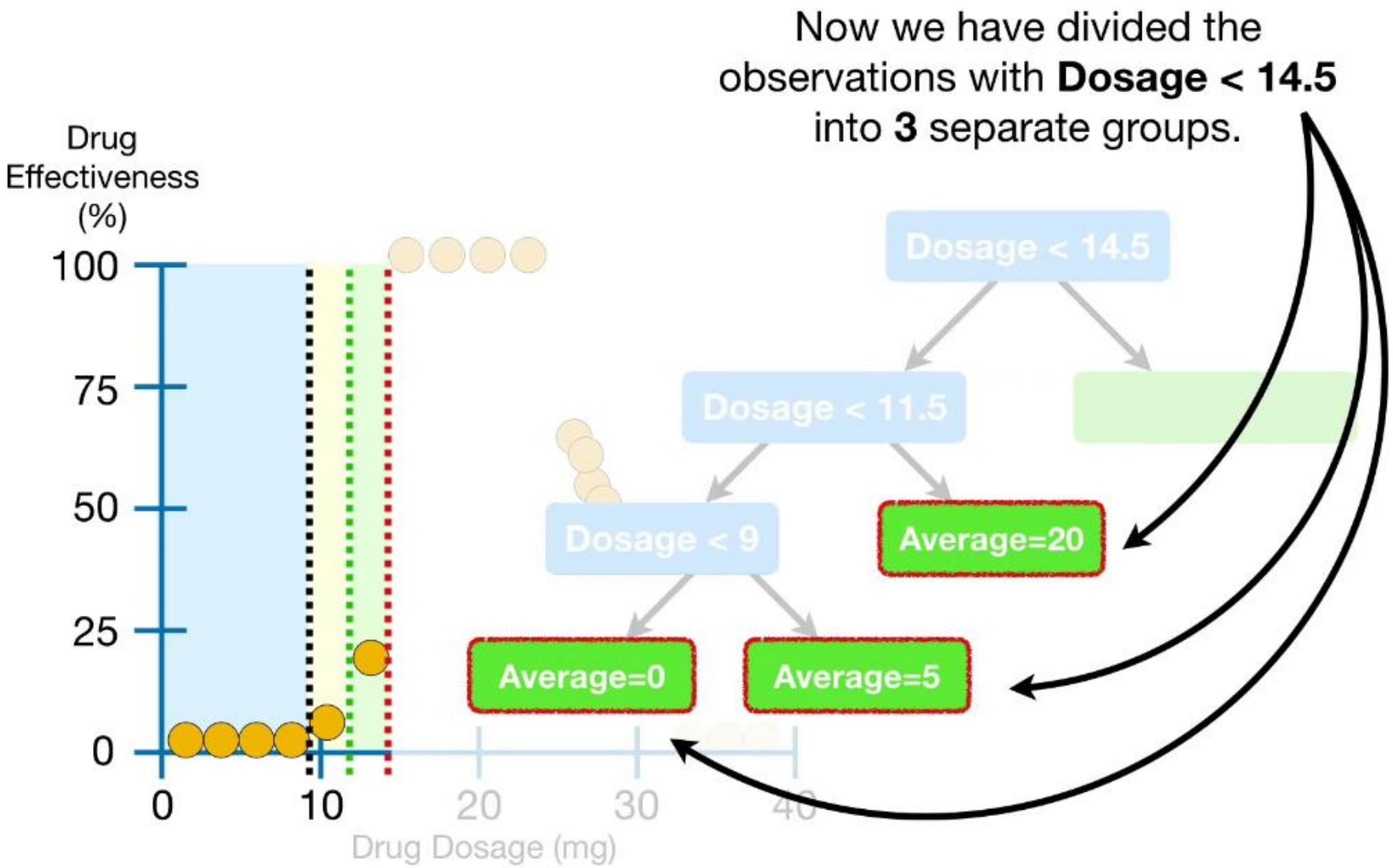
Decision Tree Regression



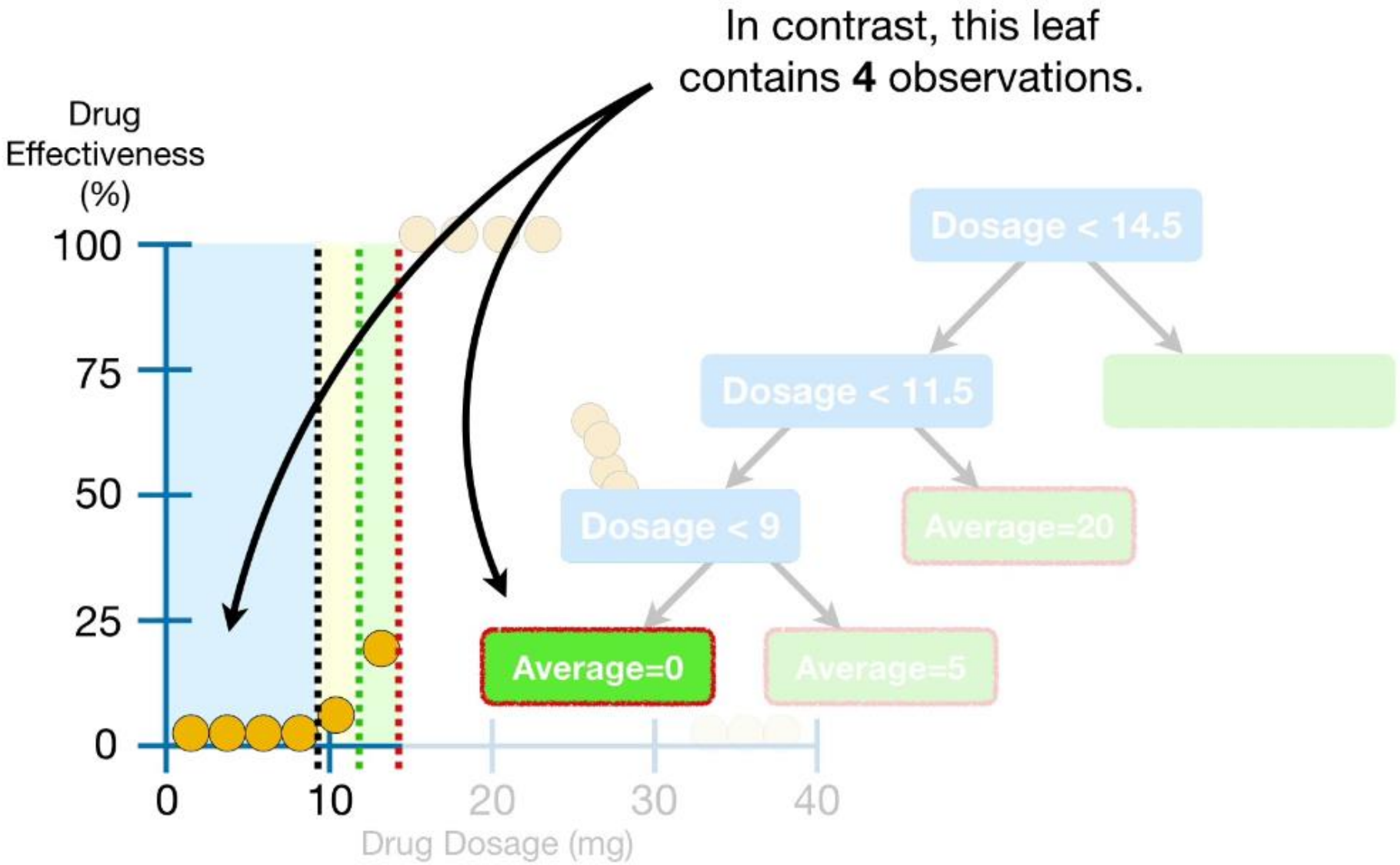
Decision Tree Regression



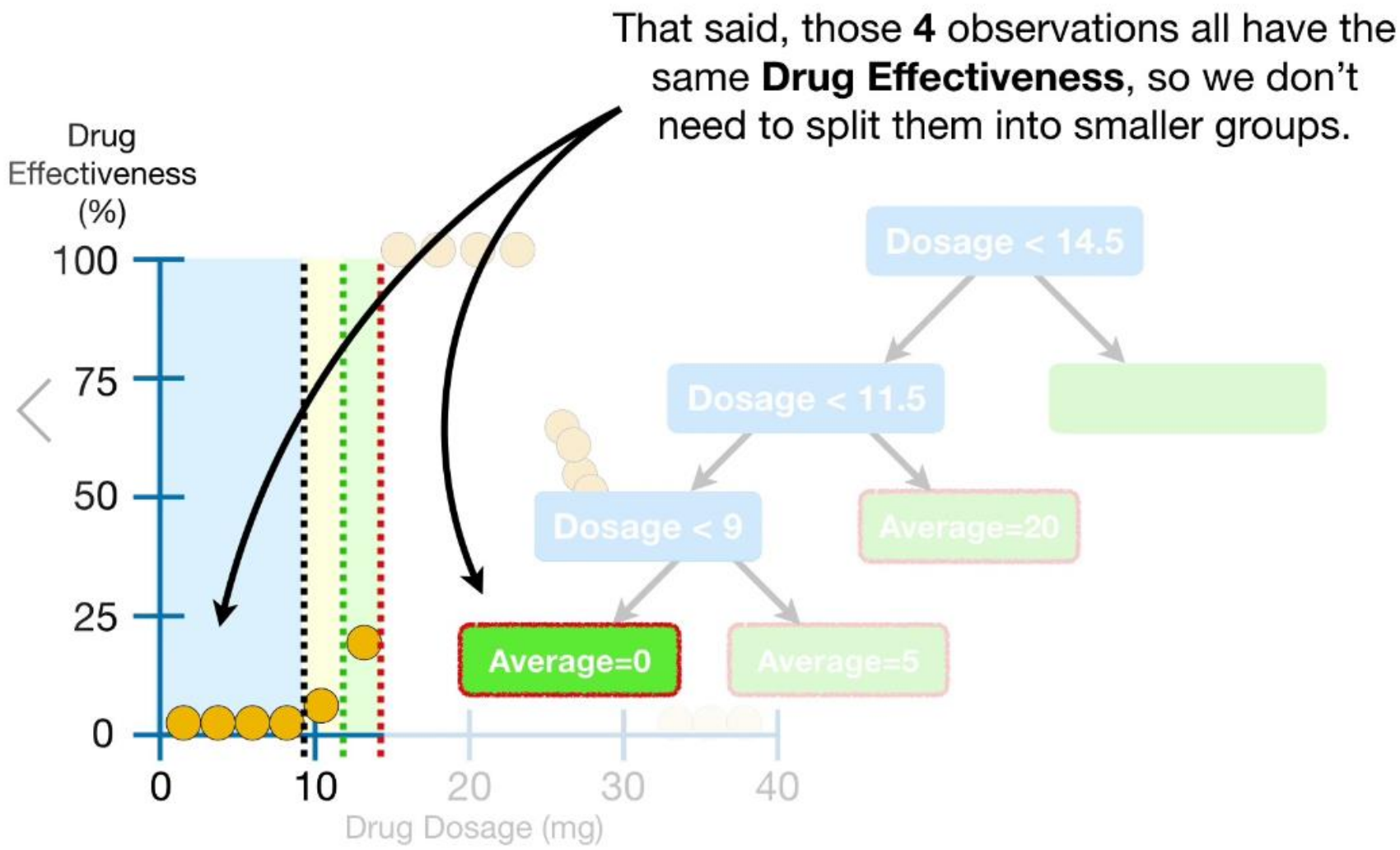
Decision Tree Regression



Decision Tree Regression

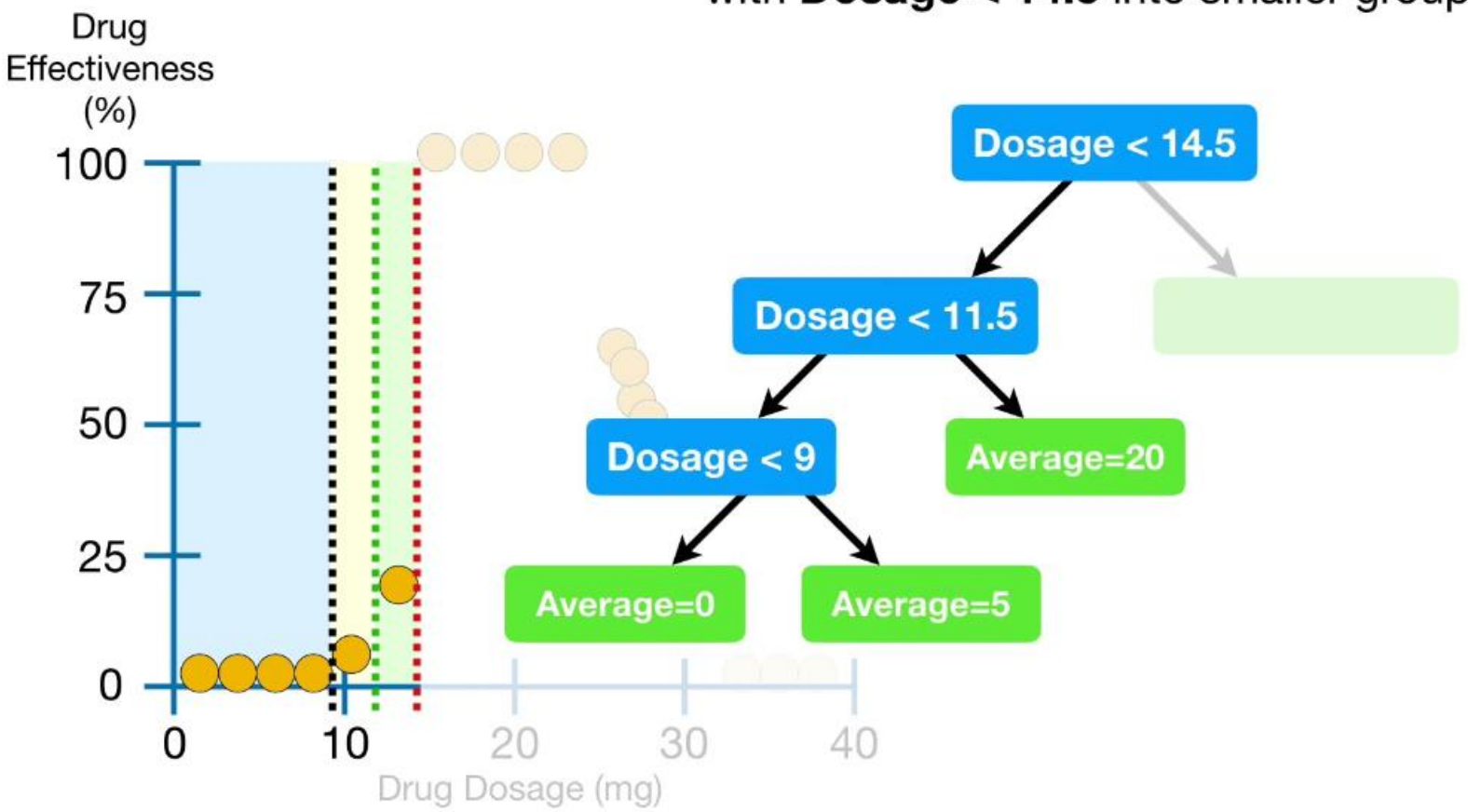


Decision Tree Regression

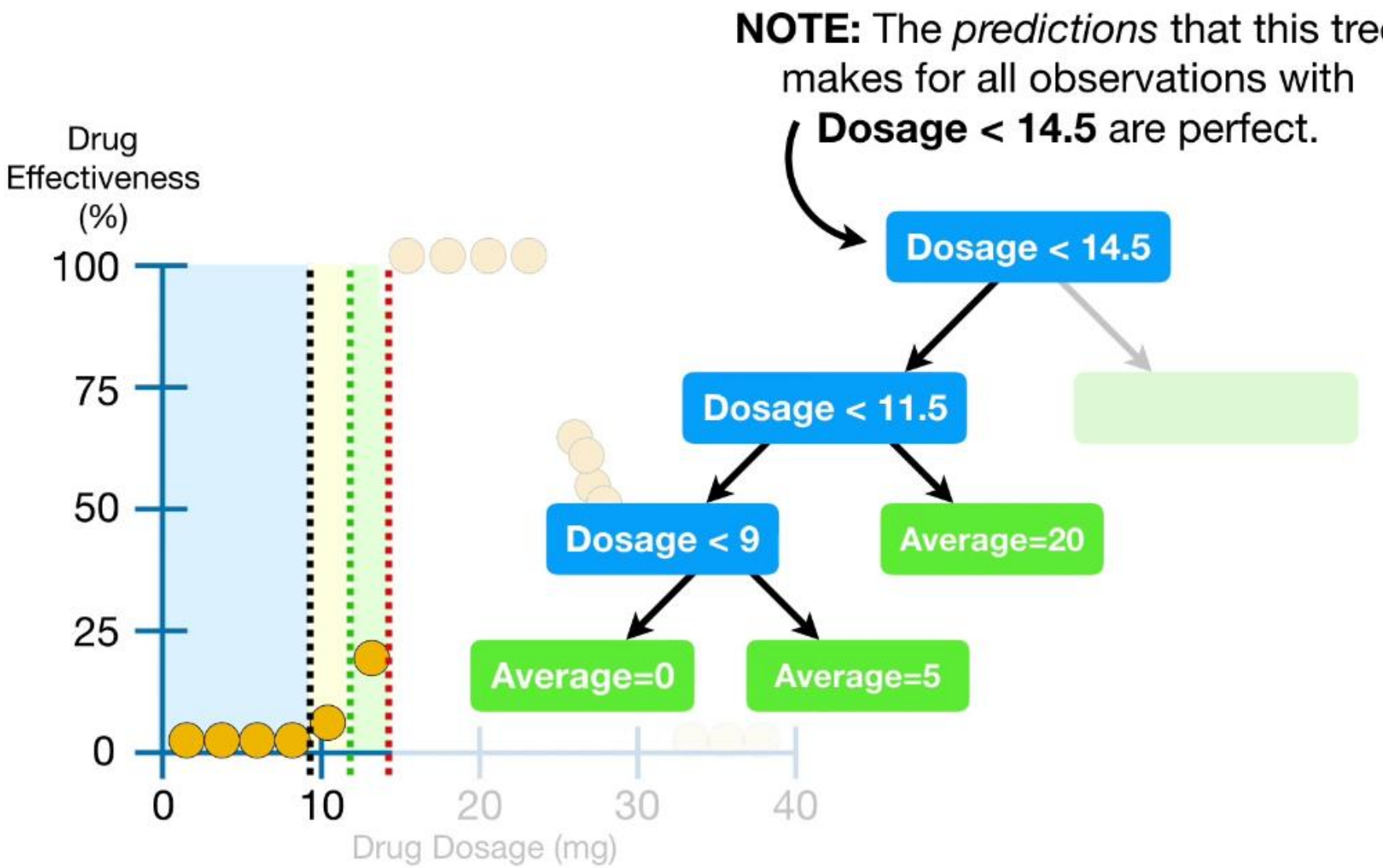


Decision Tree Regression

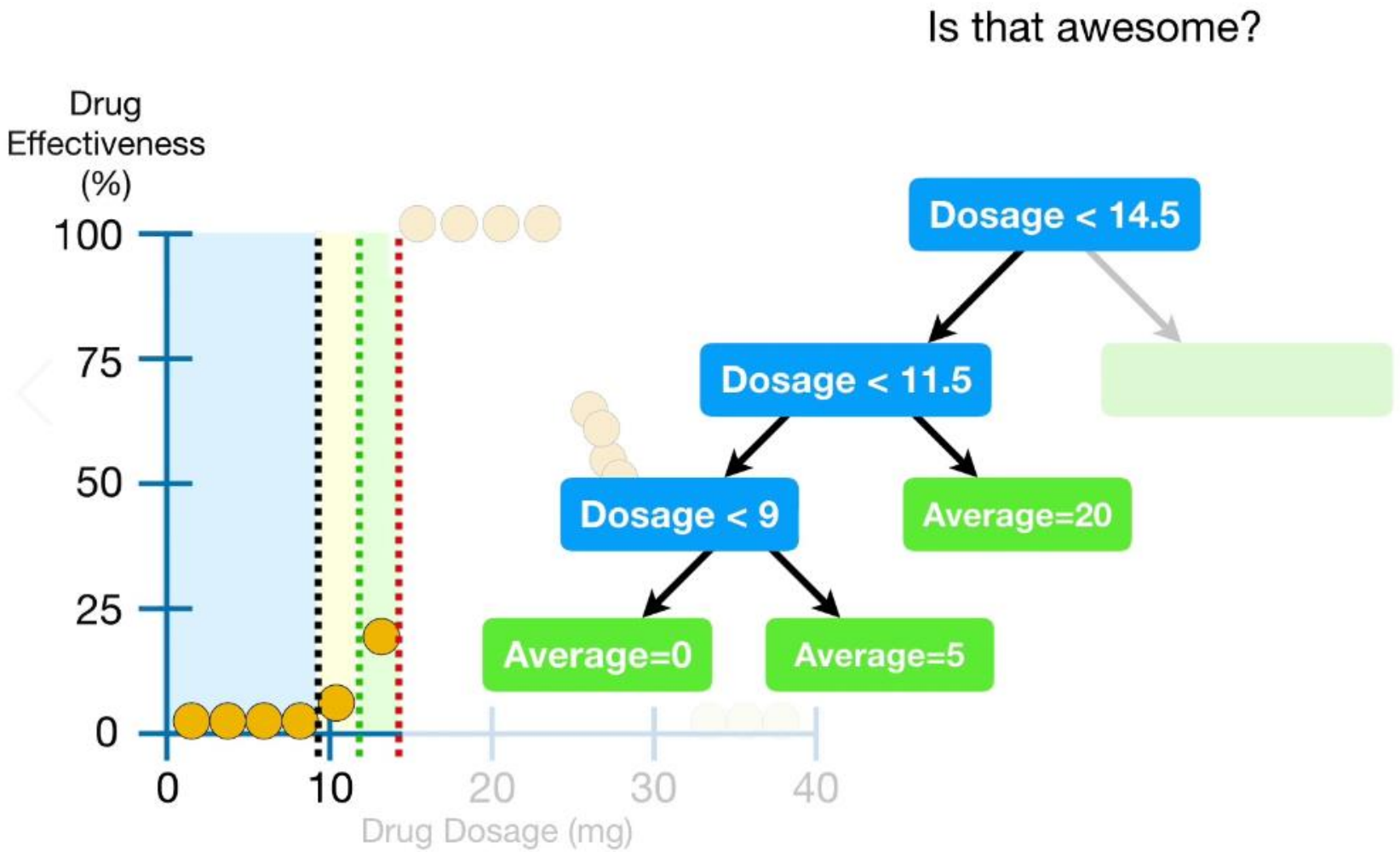
So we are done splitting the observations with **Dosage < 14.5** into smaller groups.



Decision Tree Regression

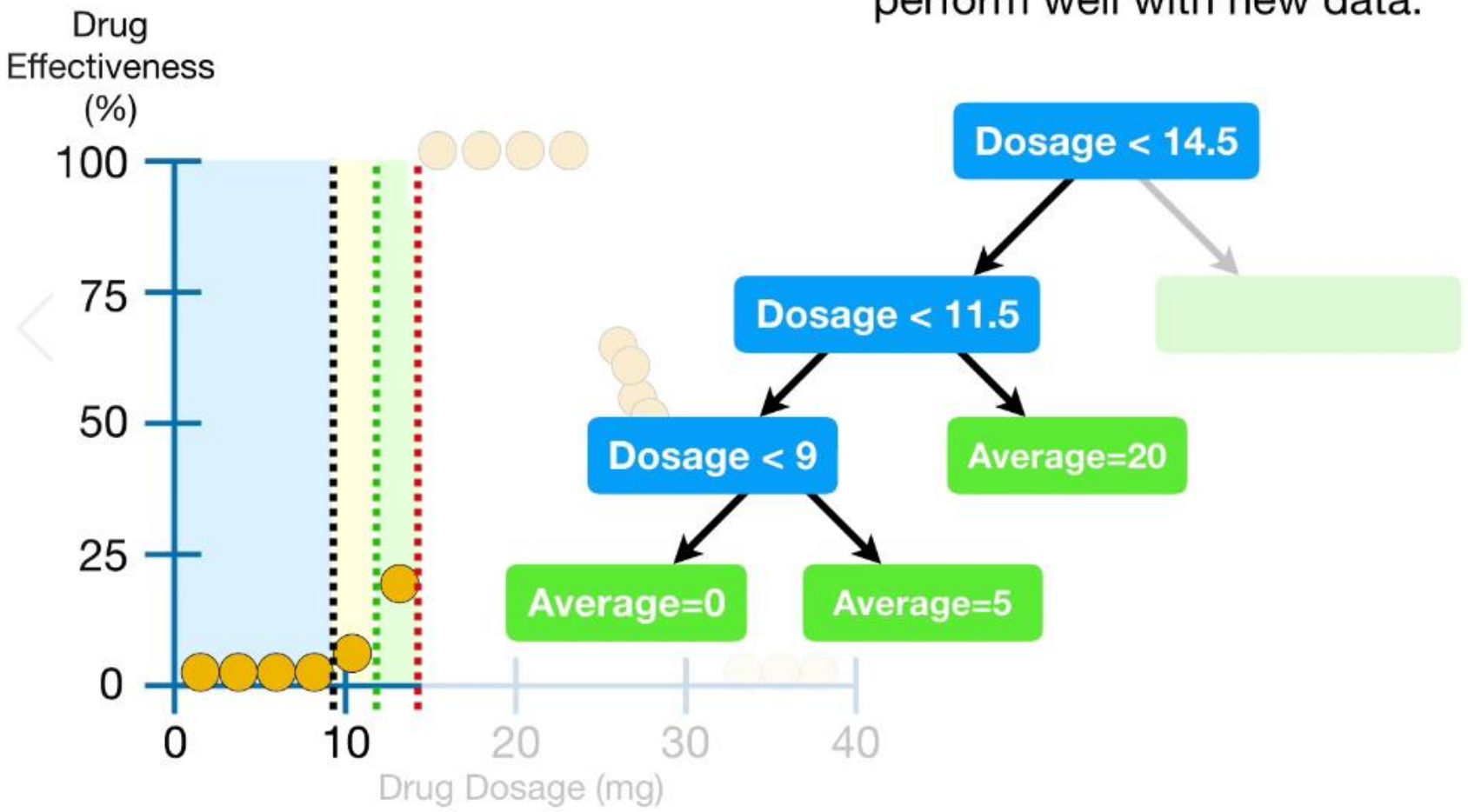


Decision Tree Regression



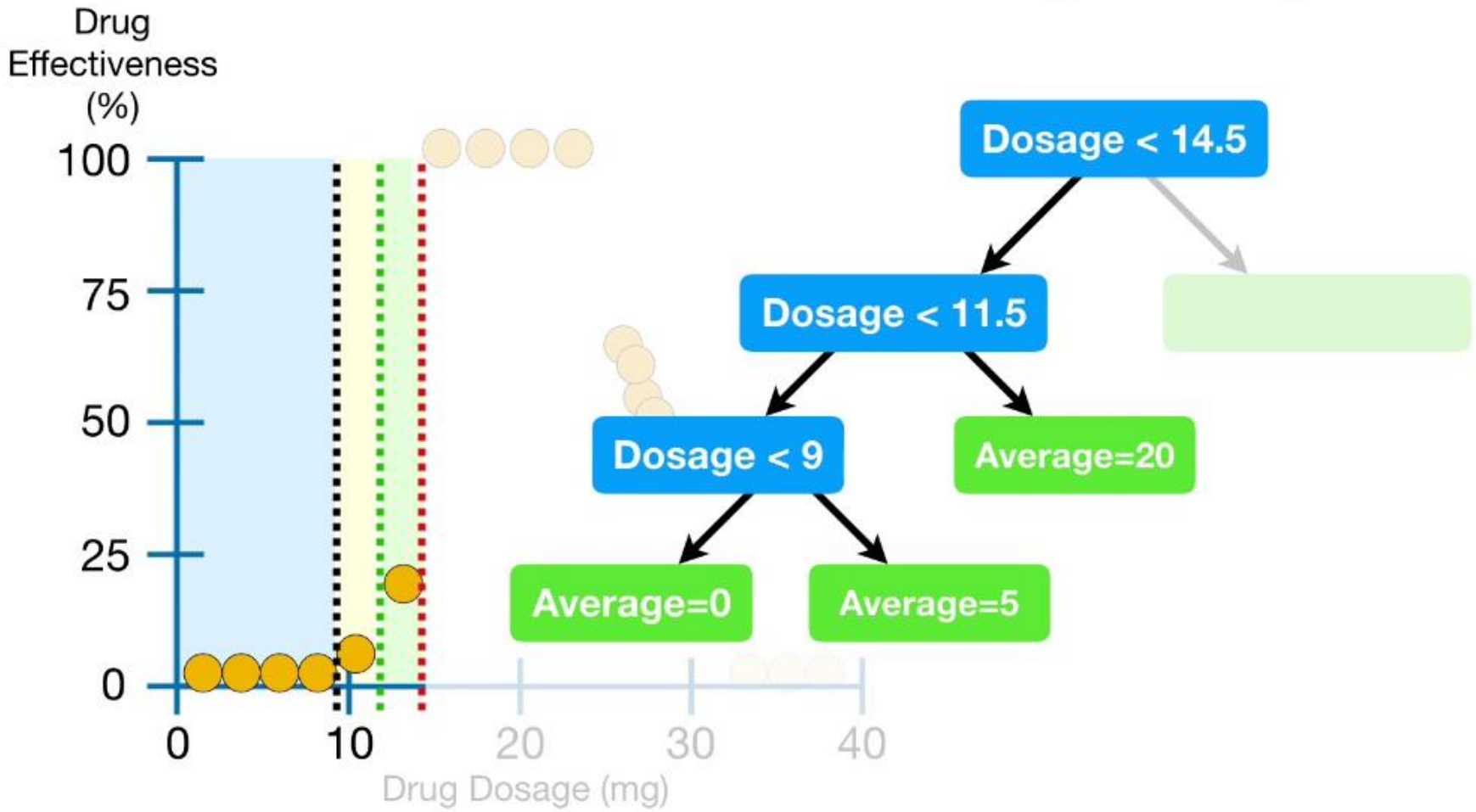
Decision Tree Regression

When a model fits the training data perfectly, it probably means it is overfit and will not perform well with new data.



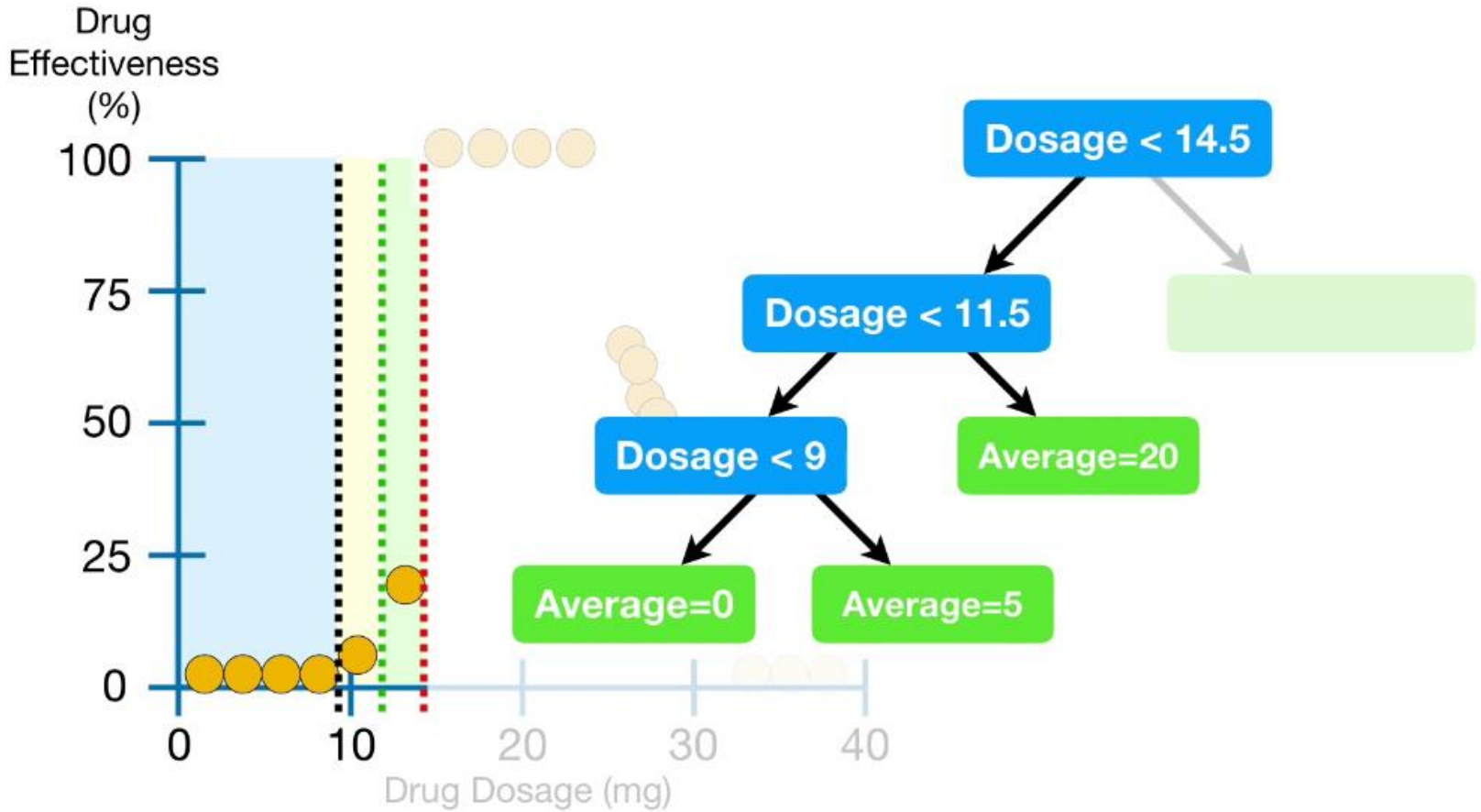
Decision Tree Regression

Is there a way to prevent our tree from overfitting the training data?



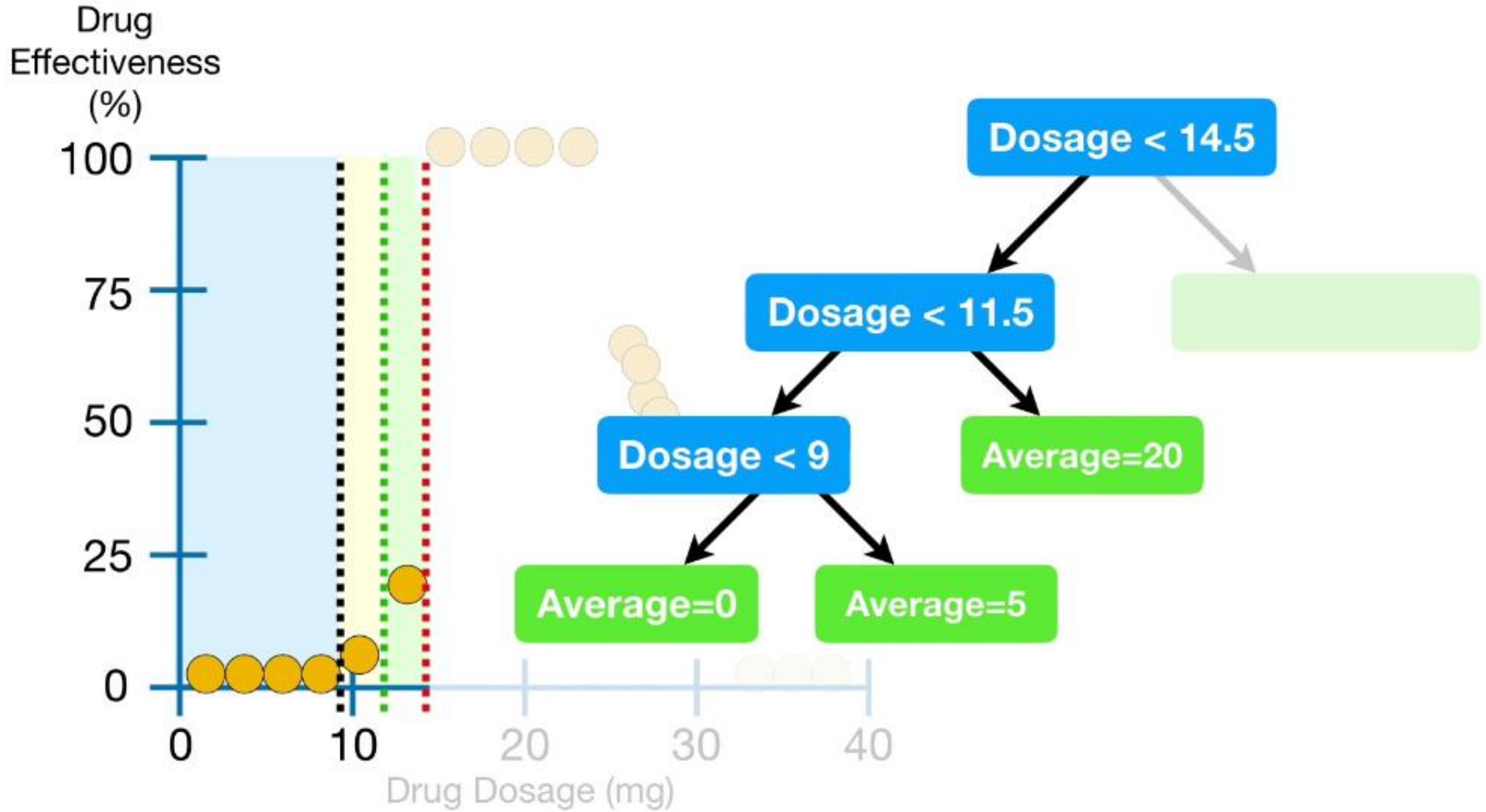
Decision Tree Regression

The simplest is to only split observations when there are more than some minimum number.



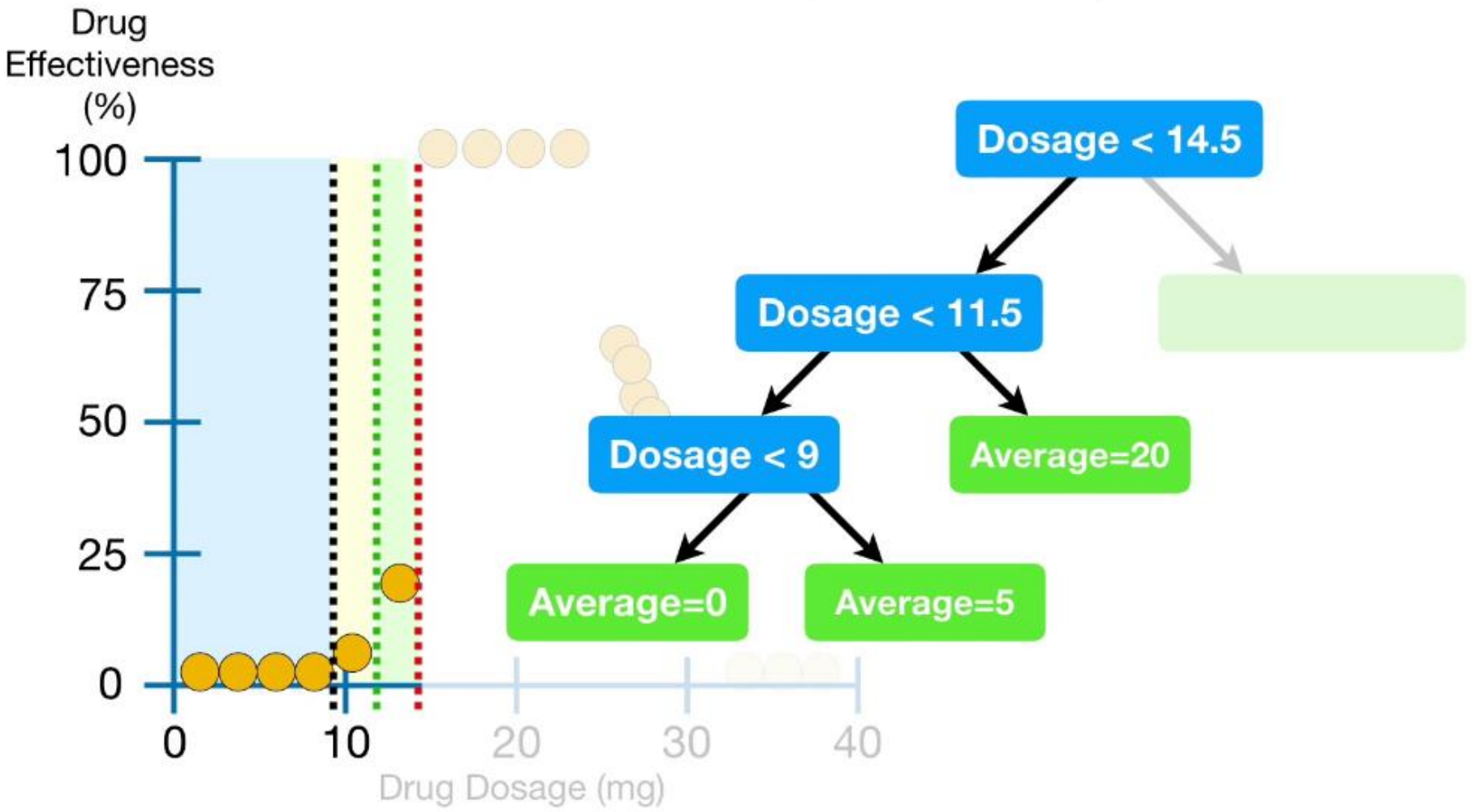
Decision Tree Regression

Typically, the minimum number of observations to allow for a split is **20**.

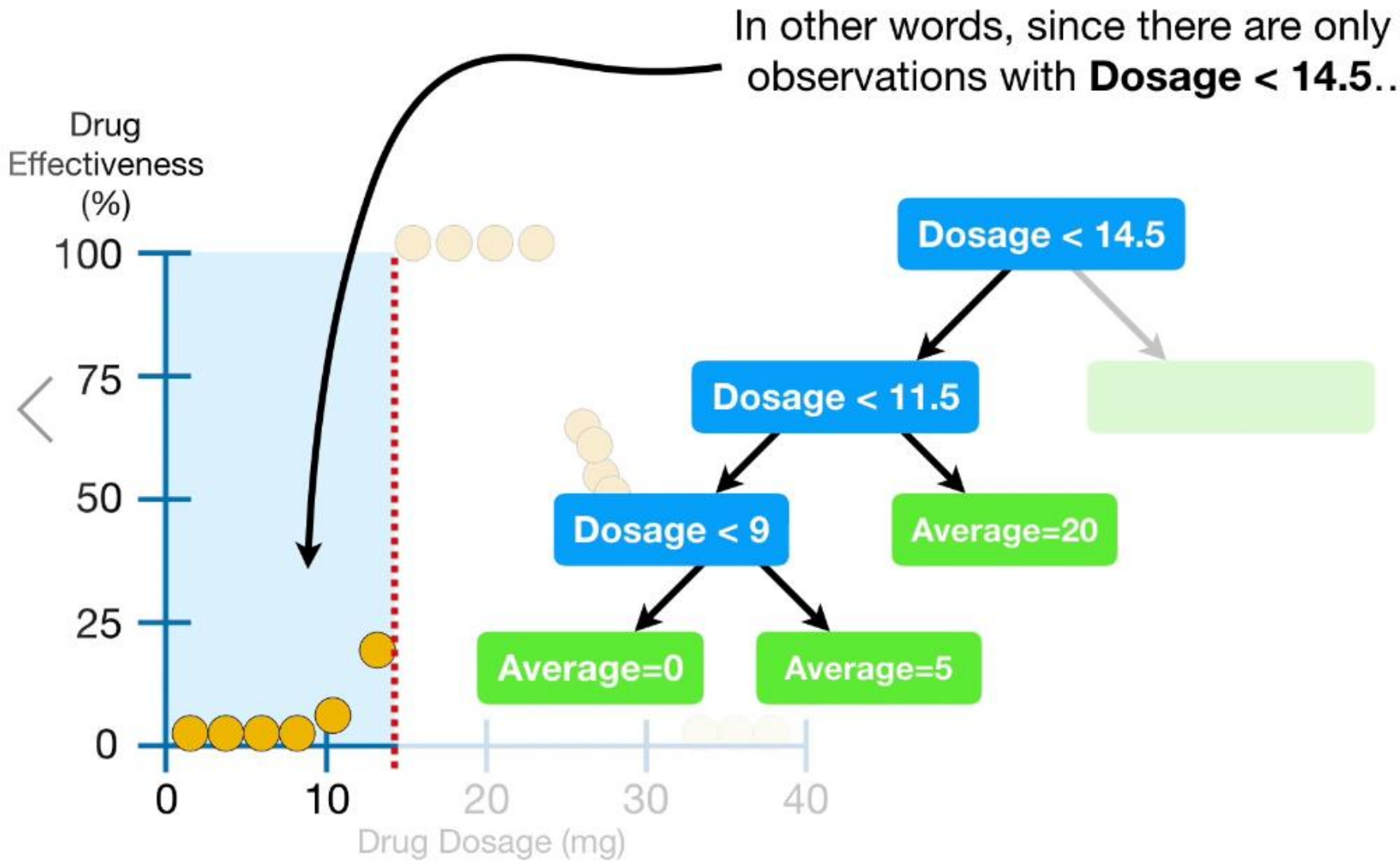


Decision Tree Regression

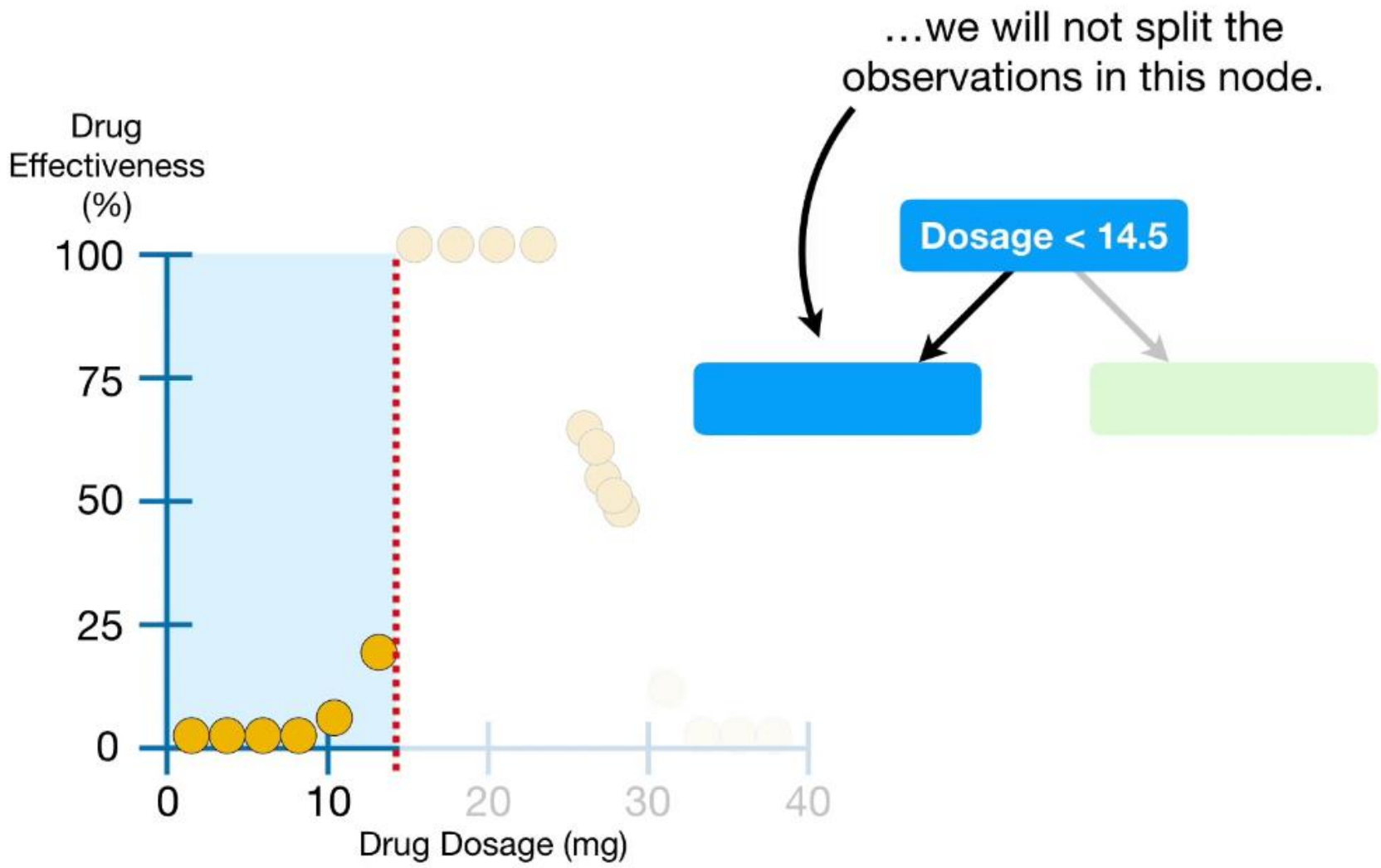
However, since this example doesn't have many observations, I set the minimum to **7**.



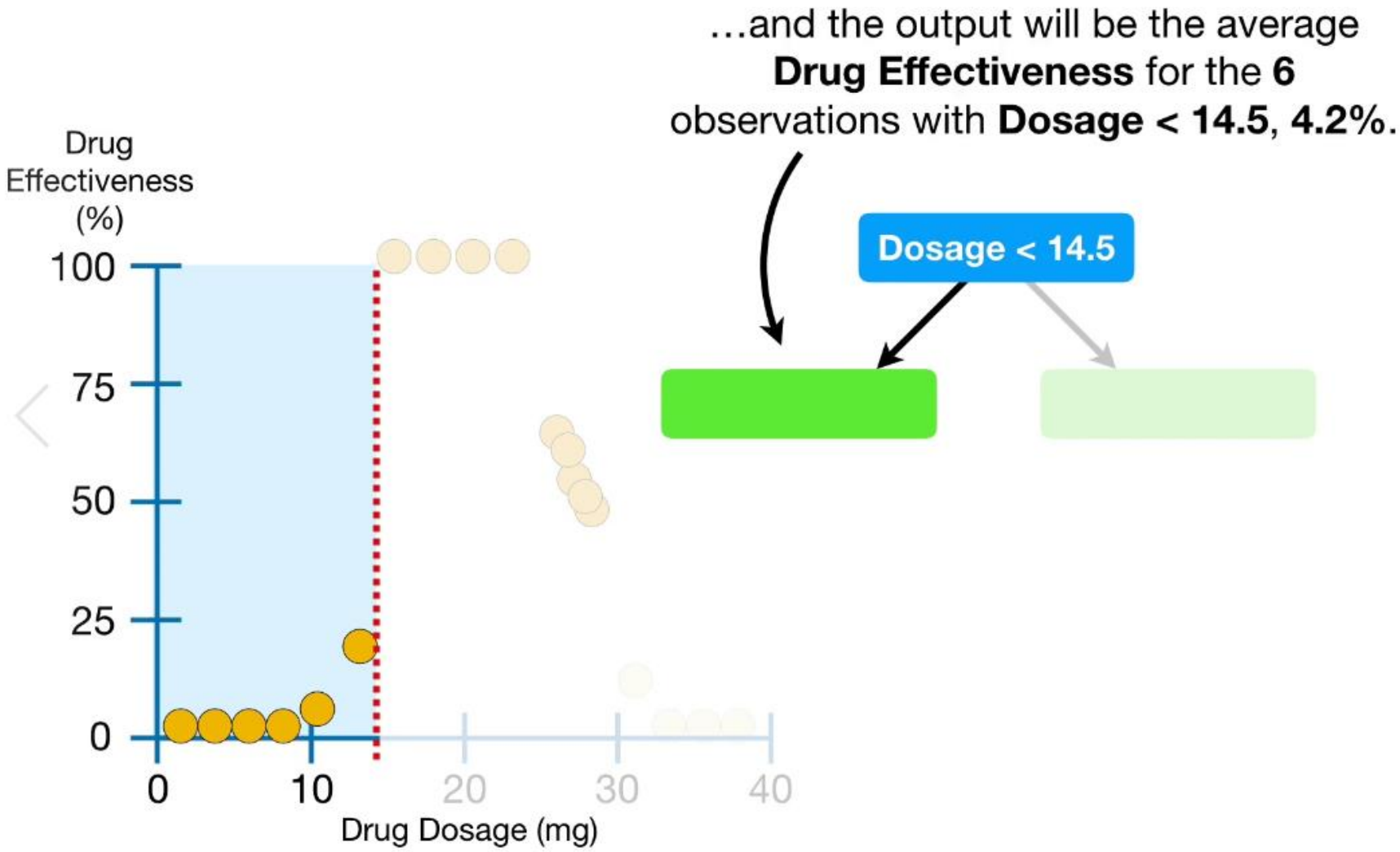
Decision Tree Regression



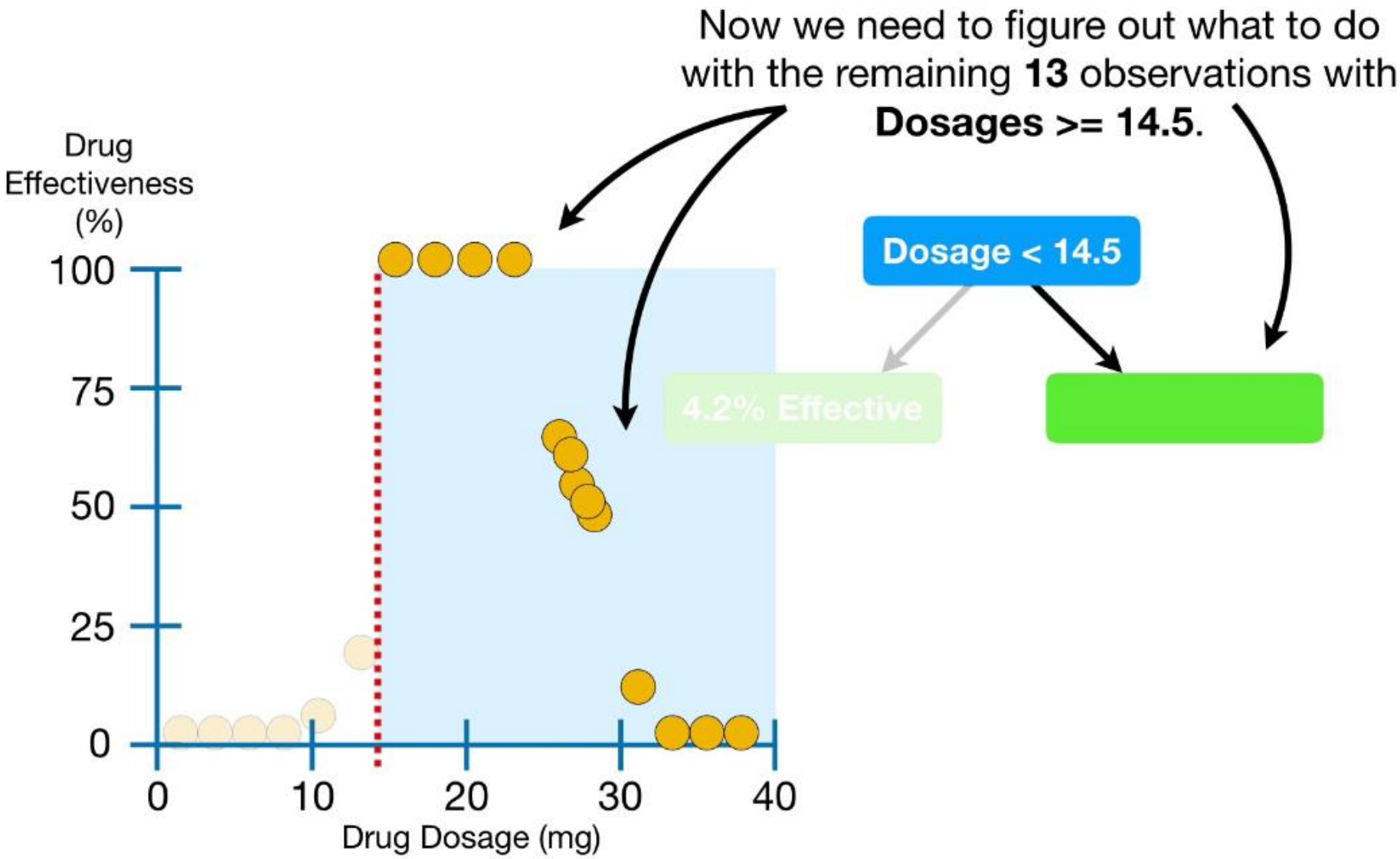
Decision Tree Regression



Decision Tree Regression

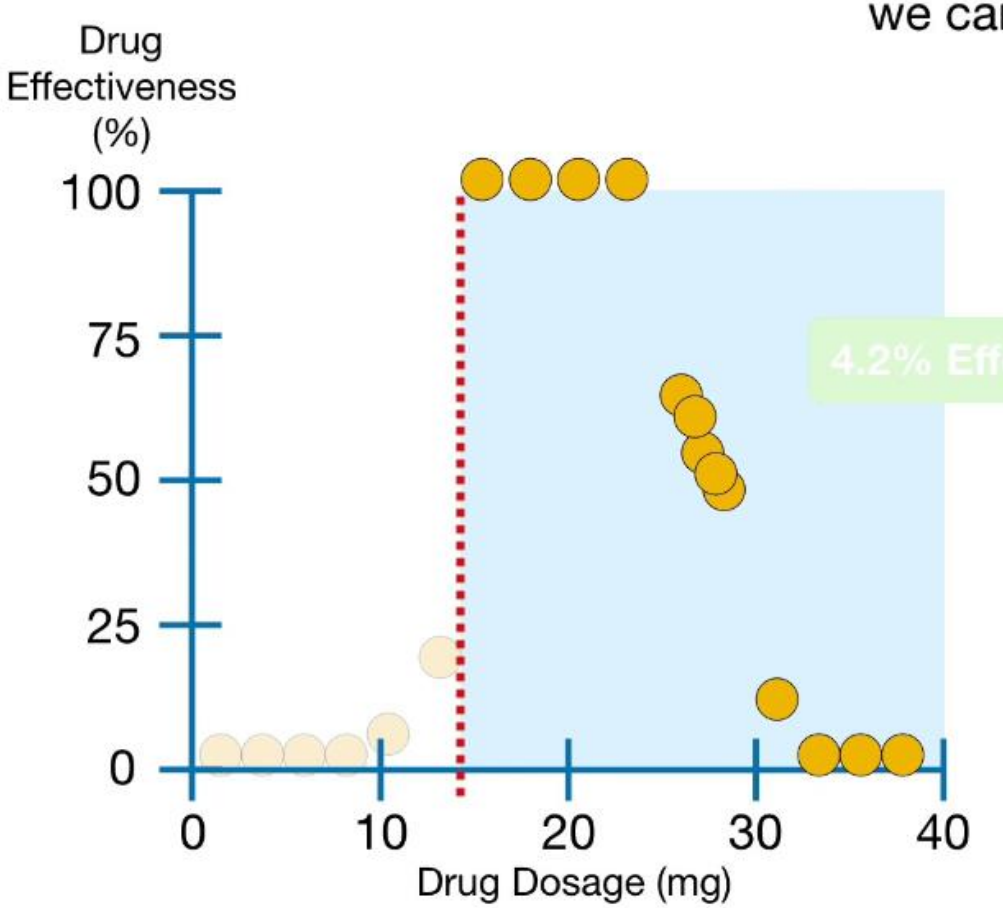


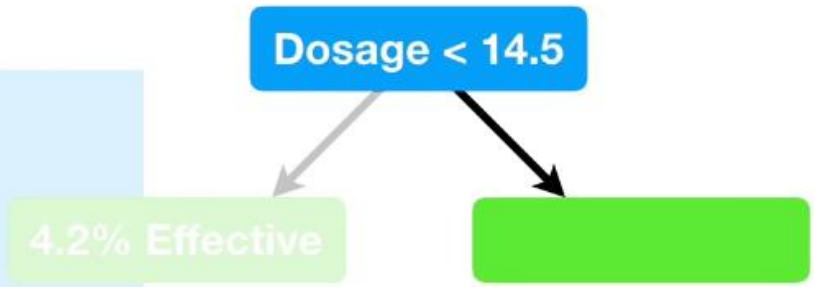
Decision Tree Regression



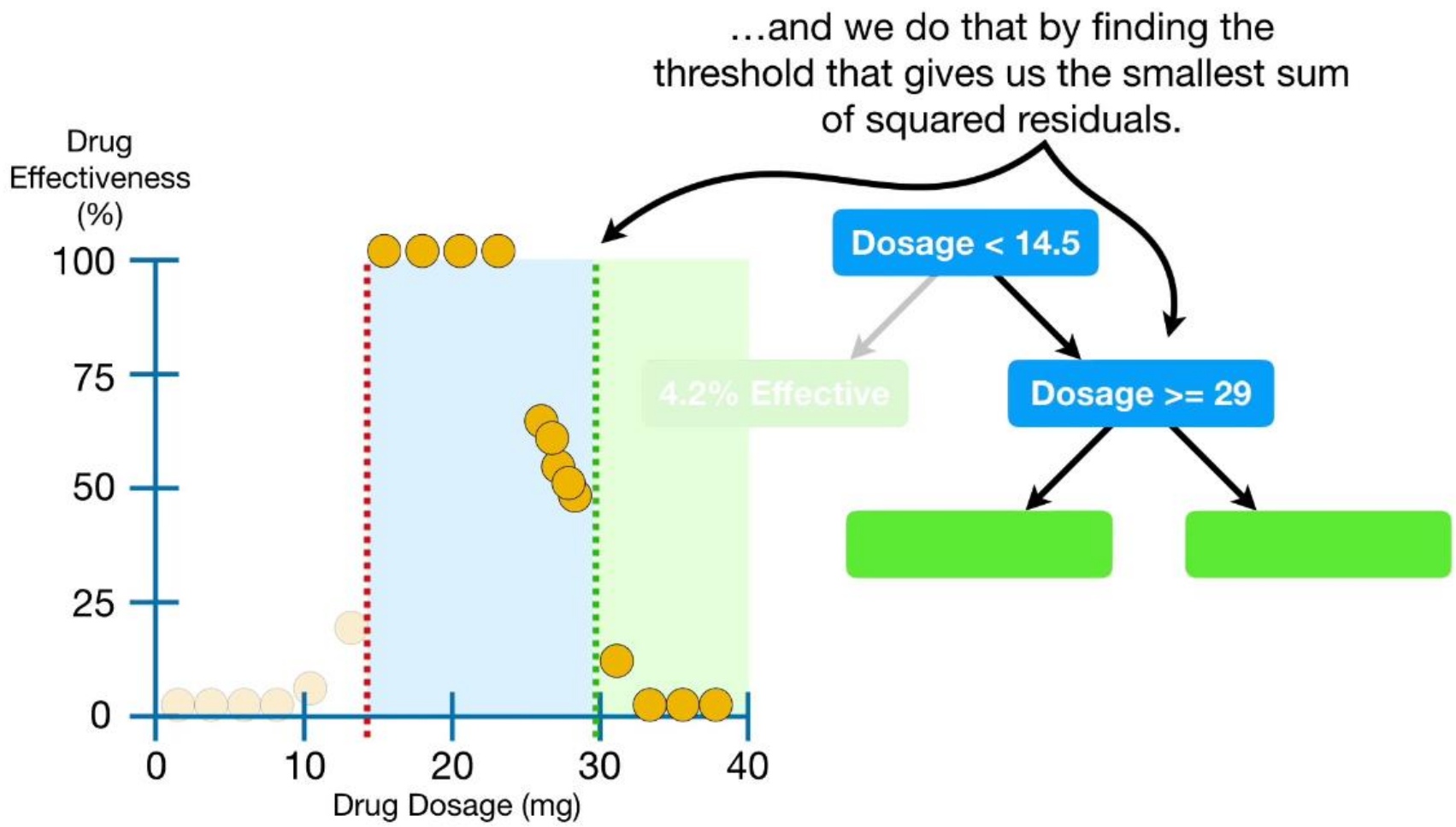
Decision Tree Regression

Since we have more than **7** observations on the right side (with **Dosage >= 14.5**), we can split them into two groups…

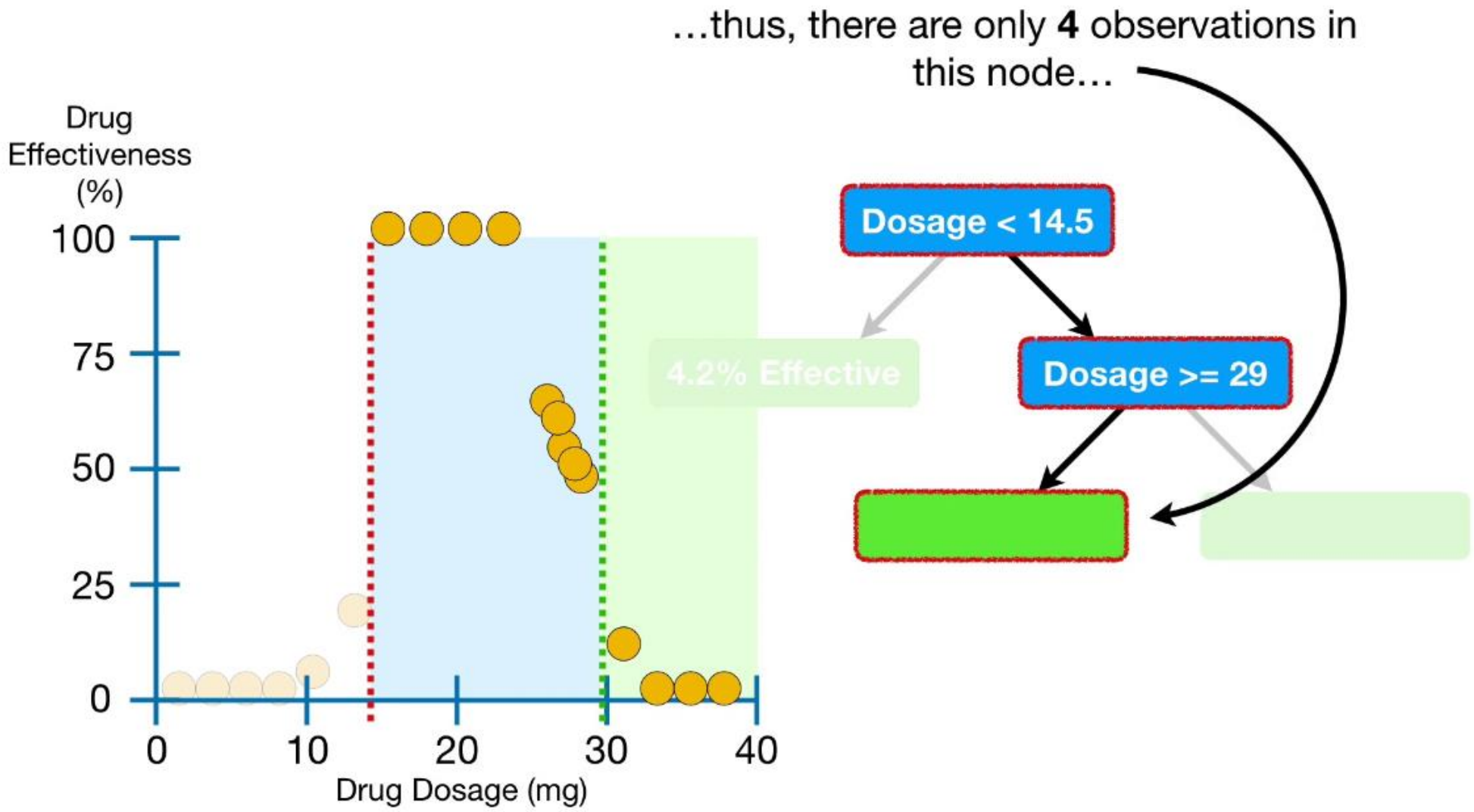




Decision Tree Regression



Decision Tree Regression



Problems of decision tree

- Decision trees can not easily capture additive structure
- For example, as seen on next slide on the left, a simple decision boundary of the form $x1 + x2$ could only be approximately modeled through the use of many splits, as each split can only consider one of $x1$ or $x2$ at a time
- A linear model on the other hand could directly derive this boundary, as shown below right
- Over fit is a terrible and common issue!!
  - We will discuss how to fix that in the next lectures

Problems of decision tree