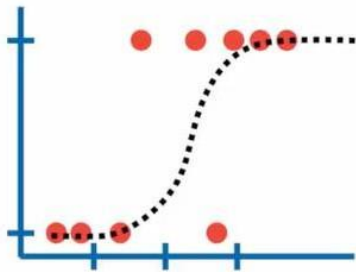




Materiale didattico per partecipante al corso **"TECNICO ESPERTO NELL'ANALISI E NELLA VISUALIZZAZIONE DEI DATI"** – Rif.P.A. 2021-15998/RER – approvata con DGR n. 1263 del 02/08/2021 di IFOA – Istituto Formazione Operatori Aziendali

# CONFUSION MATRIX

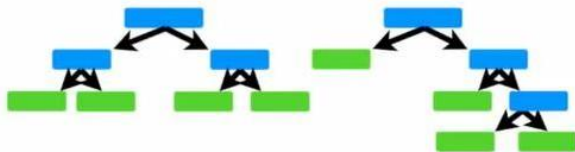
To do this, we could use  
**Logistic Regression...**



...or **K-Nearest Neighbors...**



...or a **Random Forest...**



...or some other method.  
There are tons to choose from.

How do we decide which one  
works best with our data?

We start by dividing the  
data into **Training** and  
**Testing** sets...

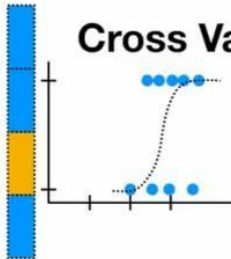
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
...				

**Training Data**

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No
...				

**Testing Data**

**NOTE:** This would be an  
excellent opportunity to use  
**Cross Validation.**



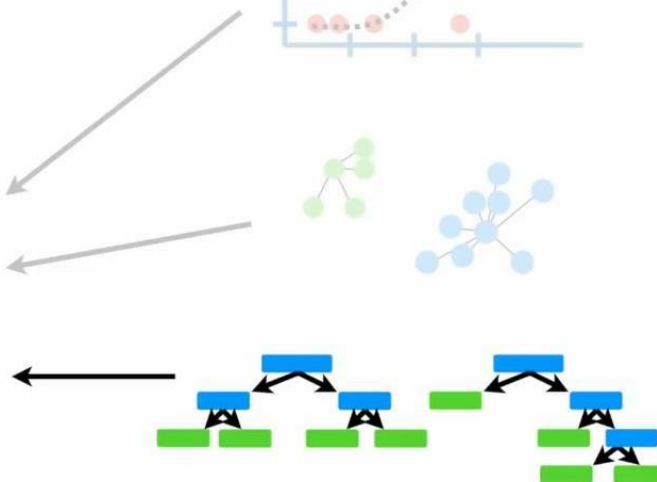
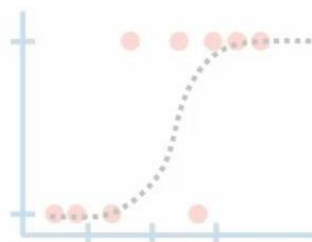
**Cross Validation....**

...it's no  
big deal!!!

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Training Data				
...	...	...	...	...

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No
Testing Data				
...	...	...	...	...

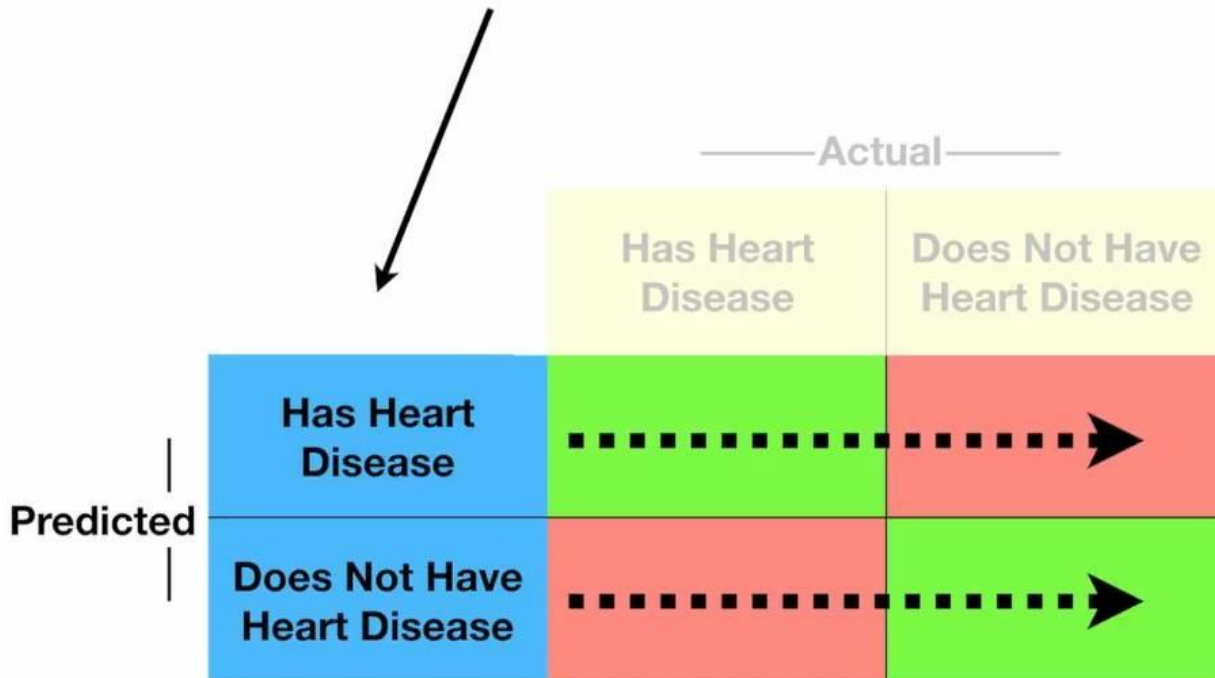
Now we need to summarize how each method performed on the **Testing** data.



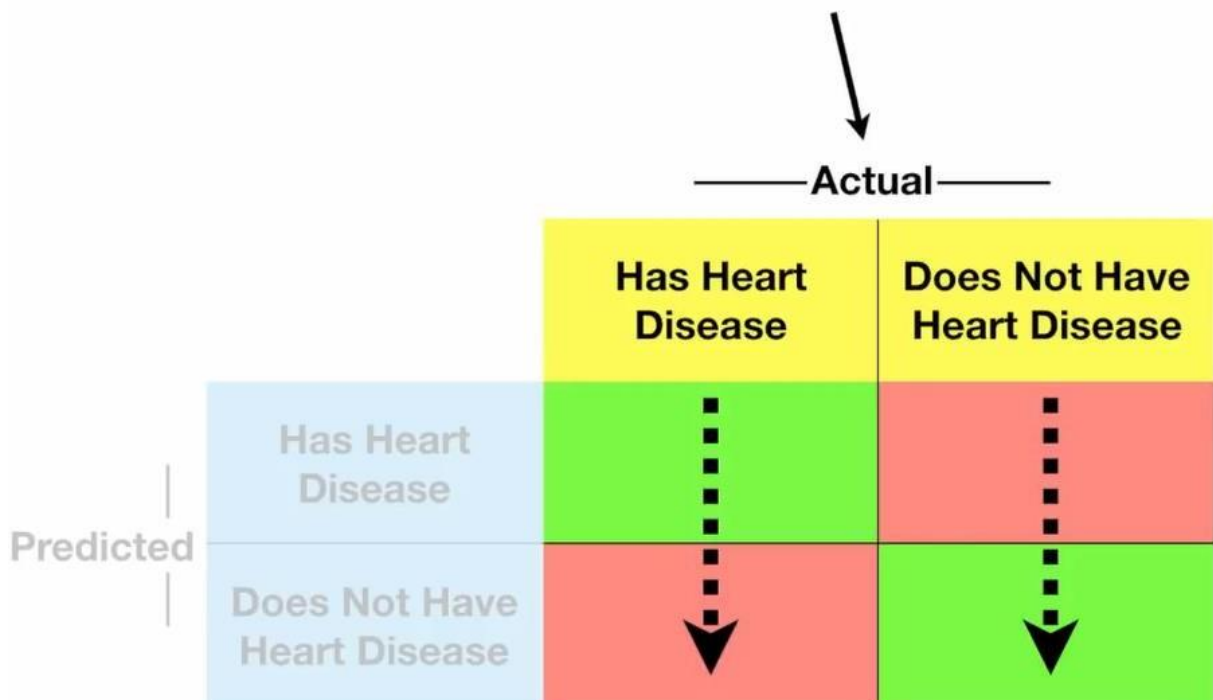
One way to do this is by creating a **Confusion Matrix** for each method.

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease		
	Does Not Have Heart Disease		

The rows in a **Confusion Matrix** correspond to what the machine learning algorithm predicted...



...and the columns correspond to the known truth.



...then the top left corner contains  
**True Positives.**

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	<b>True Positives</b>	
	Does Not Have Heart Disease		

The **True Negatives** are in the bottom right-hand corner.

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	
	Does Not Have Heart Disease		<b>True Negatives</b>

These are the patients that *did not have heart disease* that were correctly identified by the algorithm.

The bottom left-hand corner contains the  
**False Negatives...**

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	
	Does Not Have Heart Disease	<b>False Negatives</b>	True Negatives

Lastly, the top right-hand corner contains the  
**False Positives...**

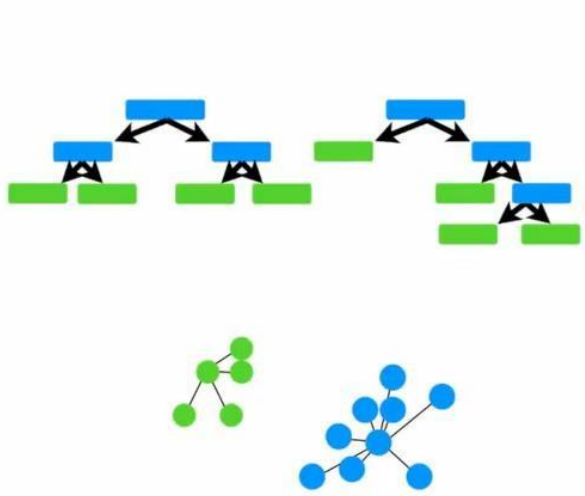
		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	<b>False Positives</b>
	Does Not Have Heart Disease	False Negatives	True Negatives

**False Positives** are patients that do not have heart disease, but the algorithm says they do.



...and the algorithm misclassified 22 patients that *did not* have heart disease by saying that they *did* (**False Positives**).

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	142	22
	Does Not Have Heart Disease	29	110



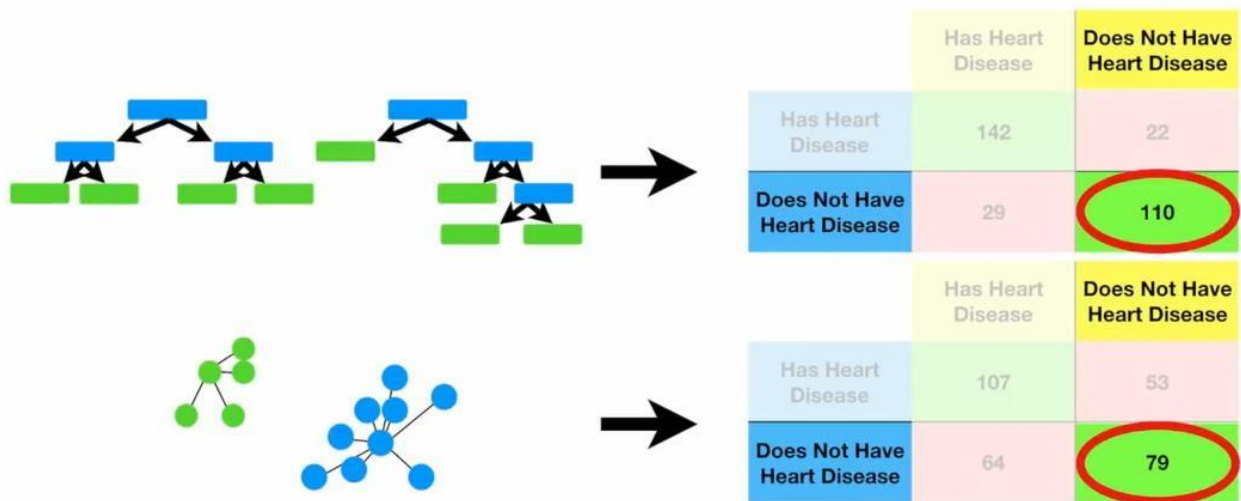
The diagram illustrates the K-Nearest Neighbors (K-NN) model structure and its application to heart disease classification. The top part shows a hierarchical tree structure with blue and green nodes, representing the model's internal logic. The bottom part shows two clusters of data points (green and blue) being classified based on their proximity to the training set. An arrow points from the clusters to a confusion matrix.

	Has Heart Disease	Does Not Have Heart Disease
Has Heart Disease	142	22
Does Not Have Heart Disease	29	110

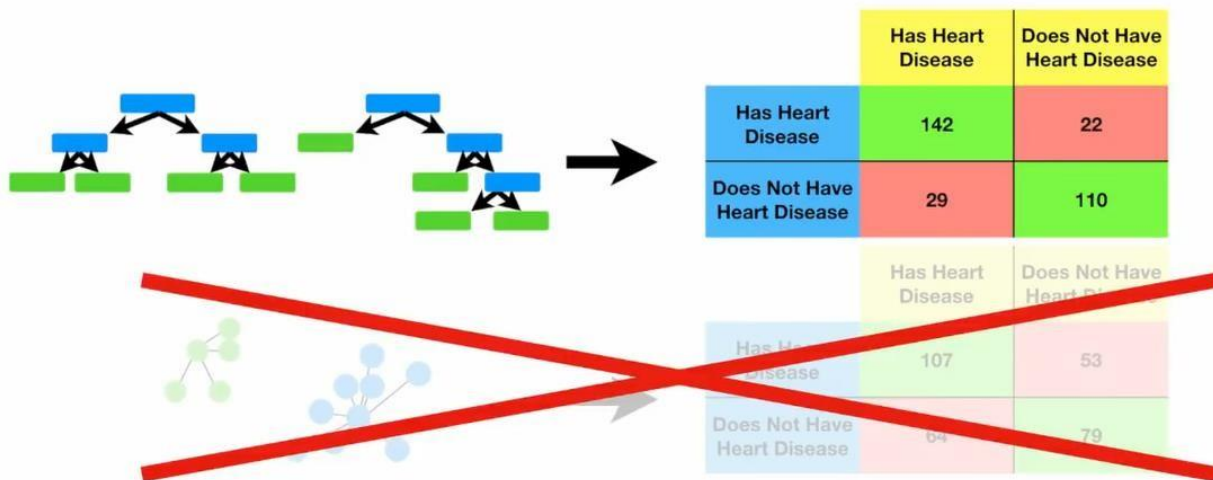
  

	Has Heart Disease	Does Not Have Heart Disease
Has Heart Disease	107	53
Does Not Have Heart Disease	64	79

**K-Nearest Neighbors** was worse than the **Random Forest** at predicting patients *with* Heart Disease (**107** vs **142**)...

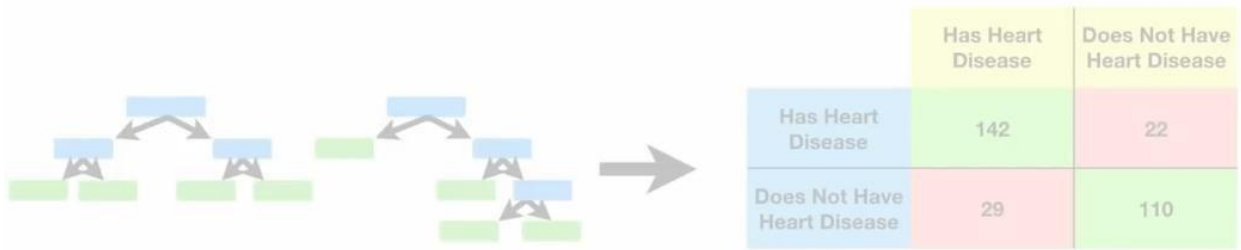


...and worse at predicting patients *without* Heart Disease (79 vs 110)...

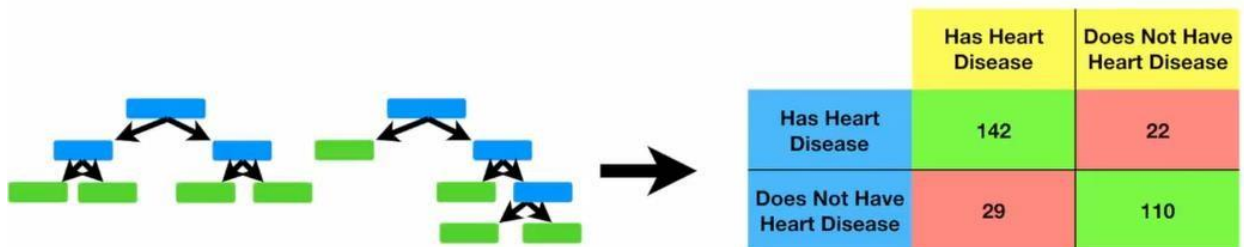
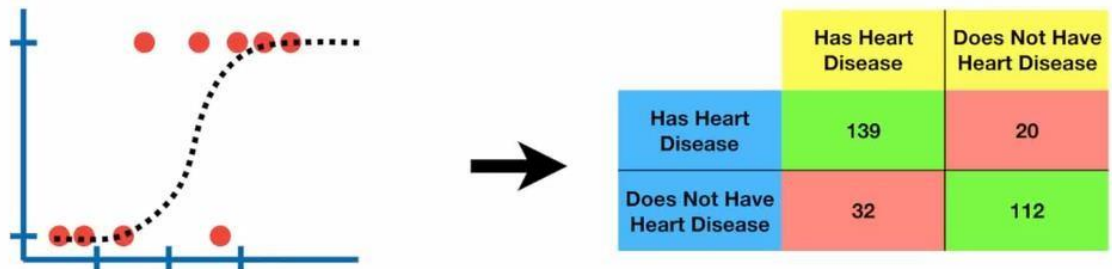


...so if we had to choose between using the **Random Forest** and **K-Nearest Neighbors**, we would choose the **Random Forest**.

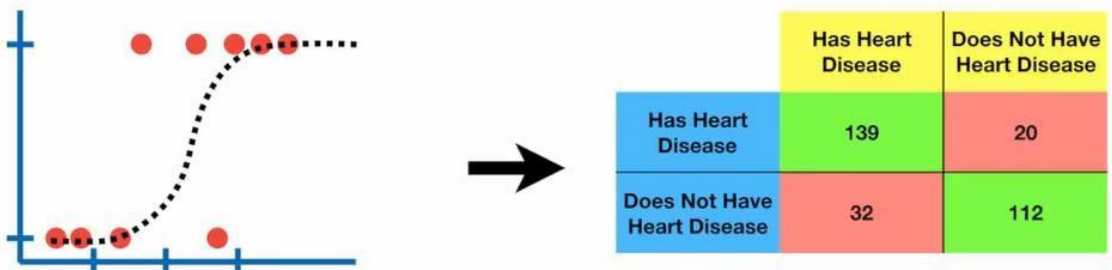




Lastly, we can apply **Logistic Regression** to the **Testing Dataset** and create a **Confusion Matrix**.



These two **Confusion Matrices** are very similar and make it hard to choose which machine learning method is a better fit for this data.



**Sensitivity, Specificity, ROC and AUC**

		Actual		
		Troll 2	Gore Police	Cool as Ice
Predicted	Troll 2	12	102	93
	Gore Police	112	23	77
	Cool as Ice	83	92	17

...and if we had 40 things to choose from, we get a confusion matrix with 40 rows and 40 columns.



In summary, a **Confusion Matrix** tells you what your machine learning algorithm did right...

...and what it did wrong.

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	False Positives
	Does Not Have Heart Disease	False Negatives	True Negatives

Once we've filled out the **Confusion Matrix**, we can calculate two useful metrics:  
**Sensitivity** and **Specificity**.

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	False Positives
	Does Not Have Heart Disease	False Negatives	True Negatives

In this case, **Sensitivity** tells us what percentage of patients *with* heart disease were correctly identified.

$$\text{Sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	False Positives
	Does Not Have Heart Disease	False Negatives	True Negatives

**Specificity** tells us what percentage of patients *without* heart disease were correctly identified.

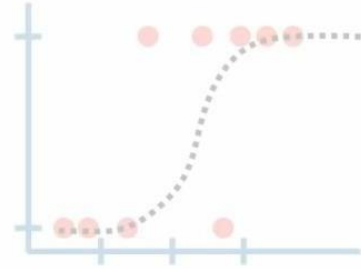
$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	True Positives	False Positives
	Does Not Have Heart Disease	False Negatives	True Negatives



$$\text{Sensitivity} = \frac{139}{139 + 32} = 0.81$$

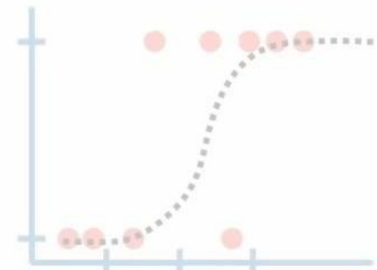
**Sensitivity** tells us that **81%** of the people *with* Heart Disease were correctly identified by the **Logistic Regression** model.



	Has Heart Disease	Does Not Have Heart Disease
Has Heart Disease	139	20
Does Not Have Heart Disease	32	112

$$\text{Specificity} = \frac{112}{112 + 20} = 0.85$$

**Specificity** tells us that **85%** of the people *without* Heart Disease were correctly identified by the **Logistic Regression** model.



	Has Heart Disease	Does Not Have Heart Disease
Has Heart Disease	139	20
Does Not Have Heart Disease	32	112

$$\text{Sensitivity} = \frac{142}{142 + 29} = 0.83$$

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$



		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	142	22
	Does Not Have Heart Disease	29	110

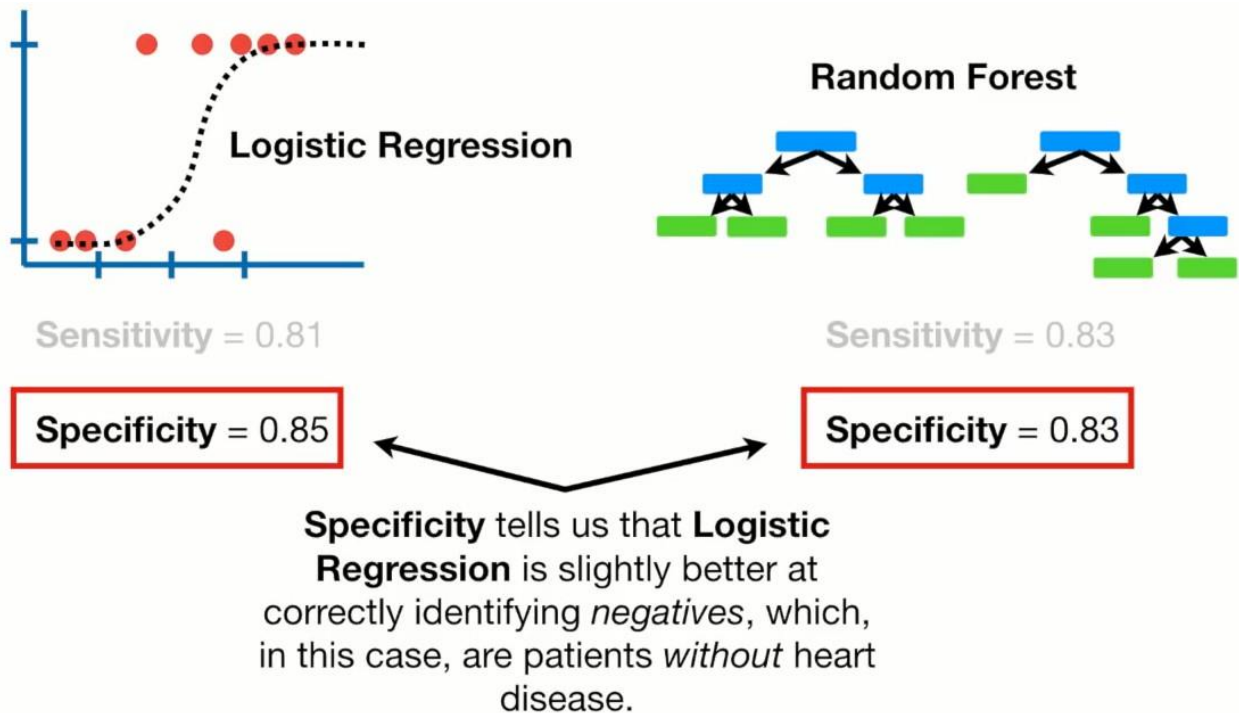
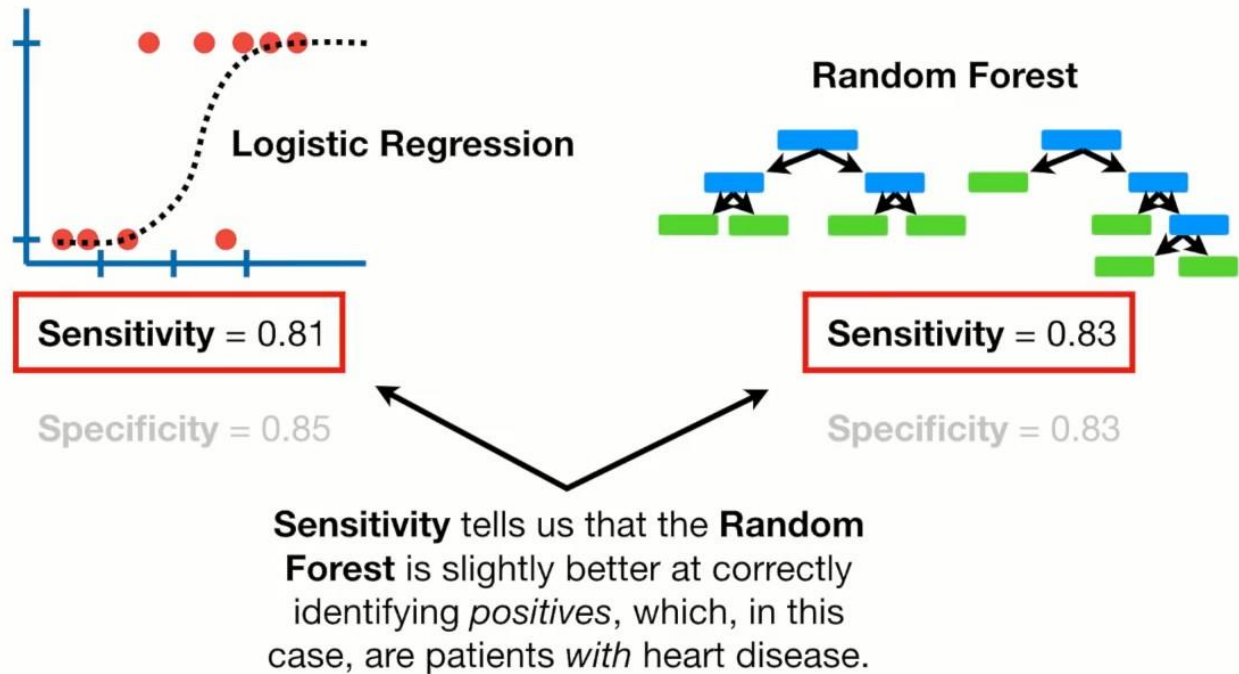
$$\text{Sensitivity} = \frac{142}{142 + 29} = 0.83$$

$$\text{Specificity} = \frac{110}{110 + 22} = 0.83$$



		Actual	
		Has Heart Disease	Does Not Have Heart Disease
Predicted	Has Heart Disease	142	22
	Does Not Have Heart Disease	29	110



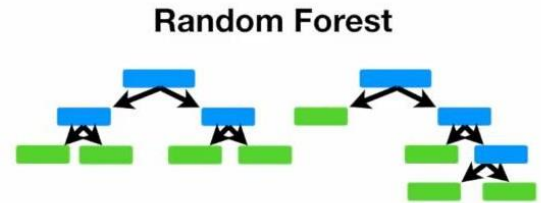




**Sensitivity** = 0.81

**Specificity** = 0.85

We would choose the **Logistic Regression** model if correctly identifying patients **without** heart disease was more important than correctly identifying patients **with** heart disease.



**Sensitivity** = 0.83

**Specificity** = 0.83

Alternatively, we would choose the **Random Forest** model if correctly identifying patients **with** heart disease was more important than correctly identifying patients **without** heart disease.

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

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

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

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

$$specificity = \frac{TN}{TN + FP}$$