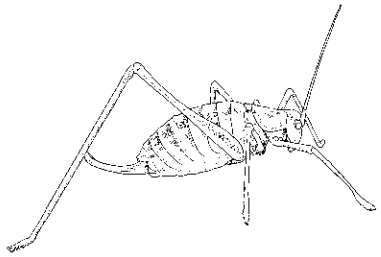
Let’s do Machine Learning

The Classification Problem

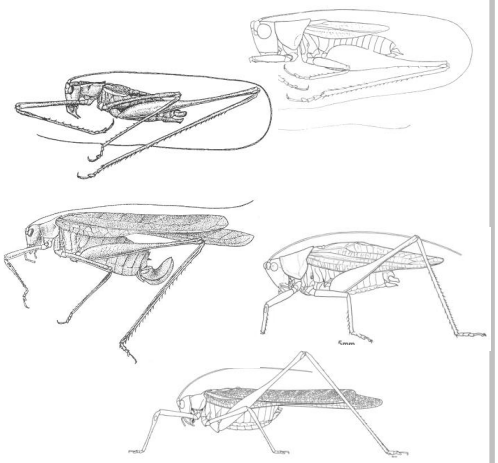
(informal definition)

Given a collection of annotated data. In this case 5 instances **Katydids** of and five of **Grasshoppers**, decide what type of insect the unlabeled example is.

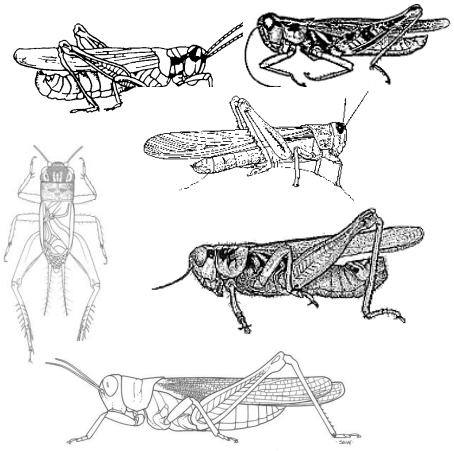


Katydid or **Grasshopper**?

Katydids



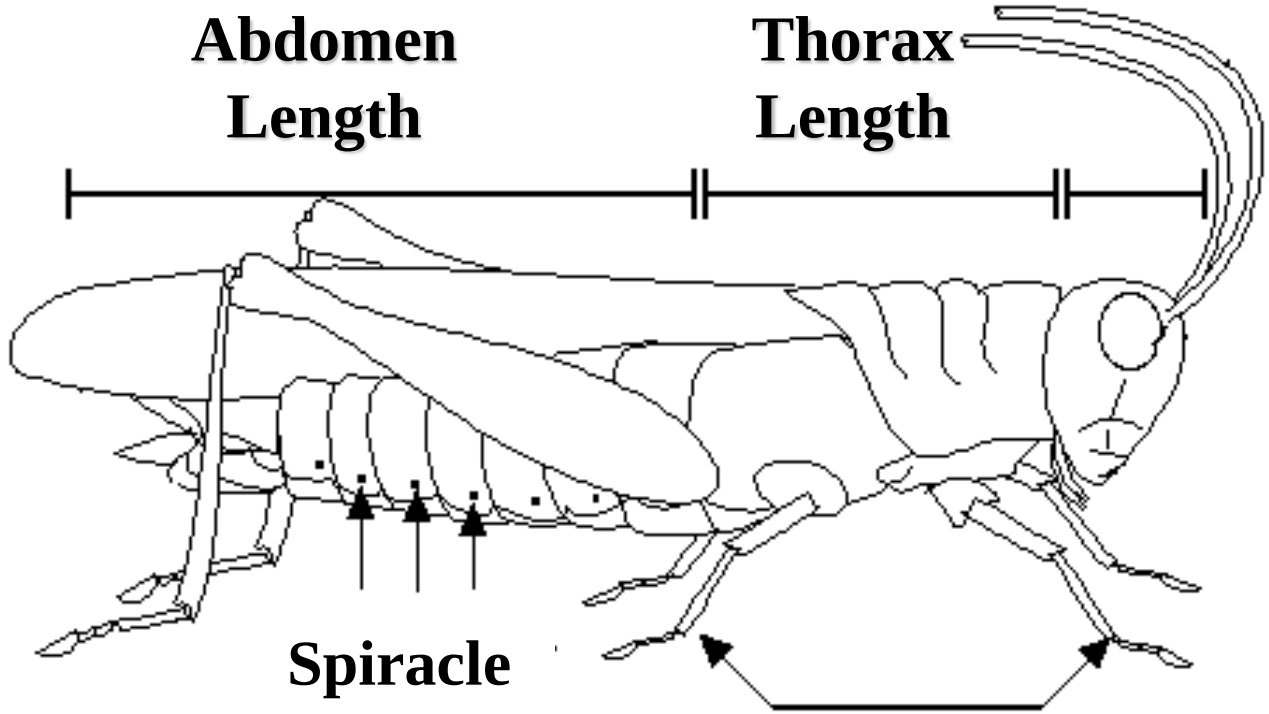
Grasshoppers



For any domain of interest, we can measure *features*

Color {Green, Brown, Gray, Other}

Has Wings?



We can store features in a database.

The classification problem can now be expressed as:

• Given a training database (**My_Collection**), predict the **class** label of a previously unseen instance

My_Collection

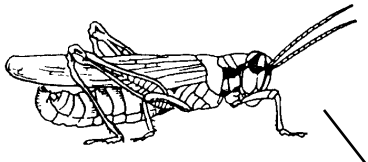
Insect ID	Abdomen Length	Antennae Length	Insect Class
1	2.7	5.5	**Grasshopper**
2	8.0	9.1	**Katydid**
3	0.9	4.7	**Grasshopper**
4	1.1	3.1	**Grasshopper**
5	5.4	8.5	**Katydid**
6	2.9	1.9	**Grasshopper**
7	6.1	6.6	**Katydid**
8	0.5	1.0	**Grasshopper**
9	8.3	6.6	**Katydid**
10	8.1	4.7	**Katydids**

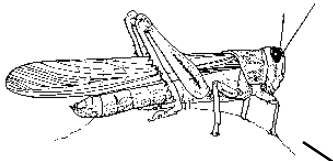
previously unseen instance =

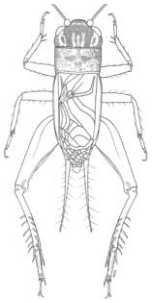
11	5.1	7.0	???????

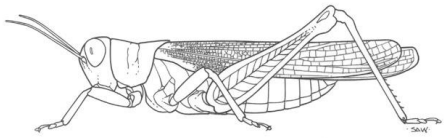
Grasshoppers

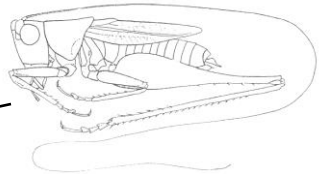
Katydids

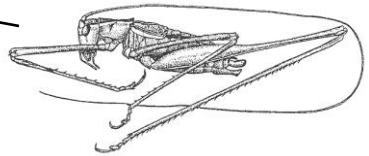


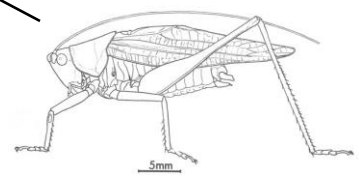


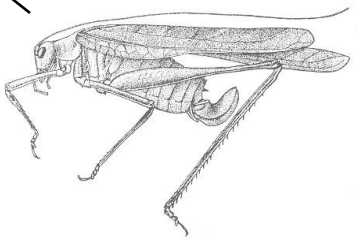


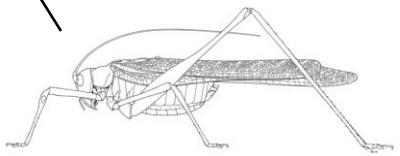


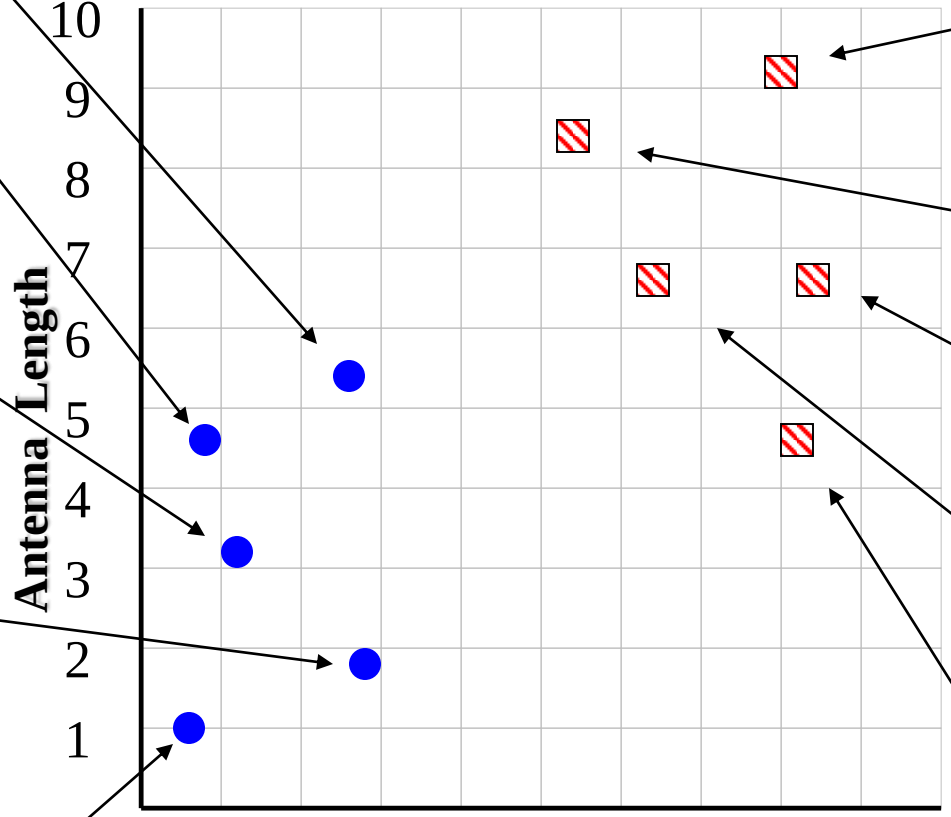




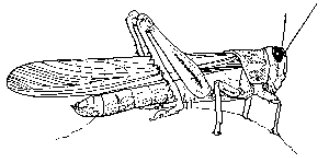




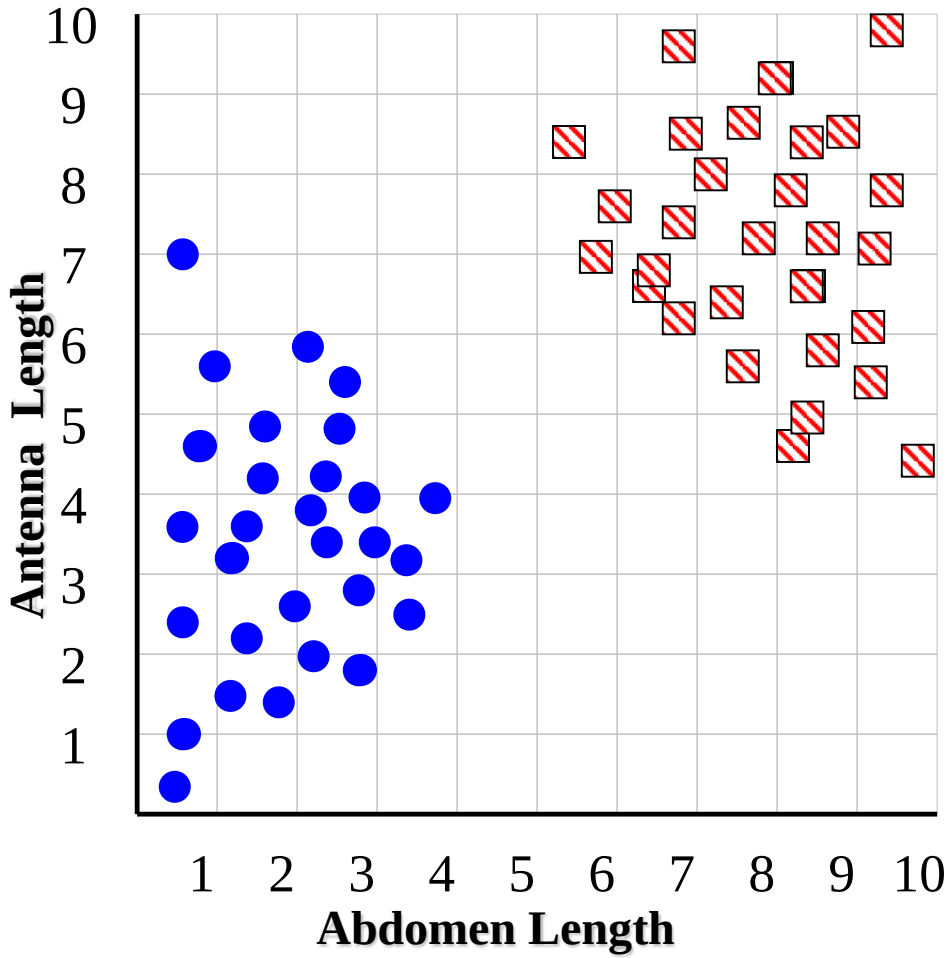




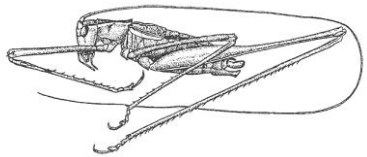
Grasshoppers



We will also use this lager dataset as a motivating example…



Katydids



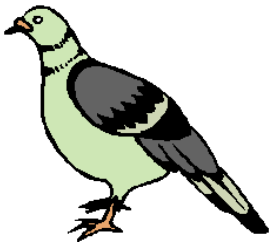
Each of these data objects are called…

- exemplars
- (training) examples
- instances
- tuples

We will return to the previous slide in two minutes. In the meantime, we are going to play a quick game.

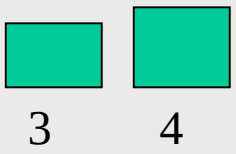
I am going to show you some classification problems which were shown to pigeons!

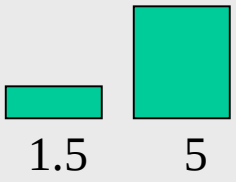
Let us see if you are as smart as a pigeon!

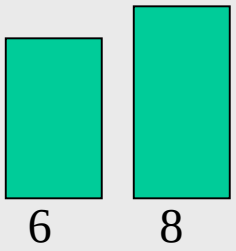


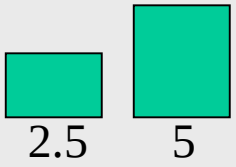
Pigeon Problem 1

Examples of class A

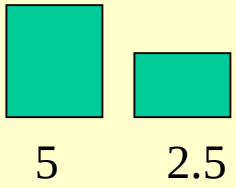


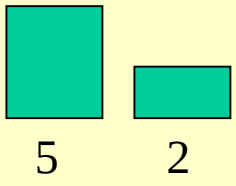


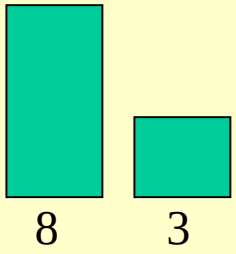


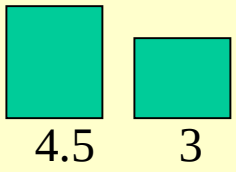


Examples of class B



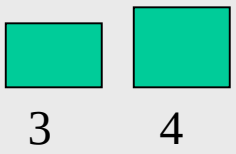


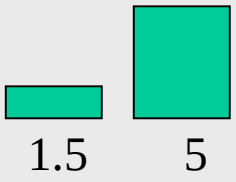


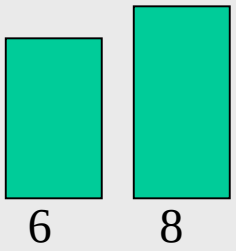


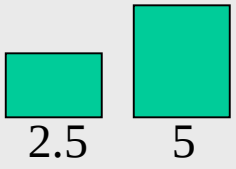
Pigeon Problem 1

Examples of class A

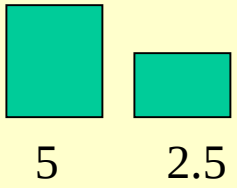


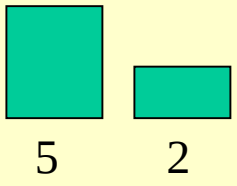


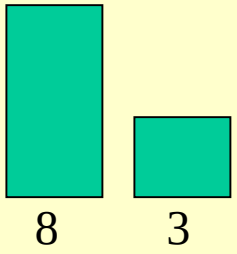


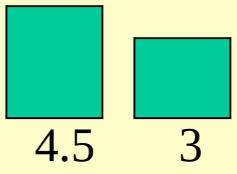


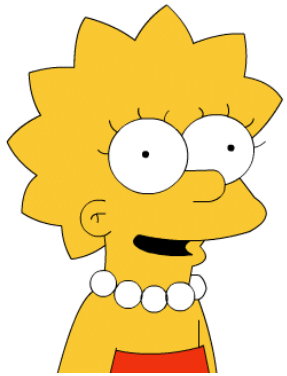
Examples of class B



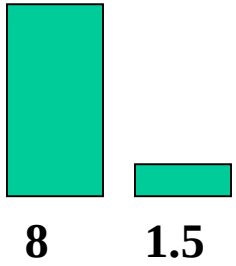




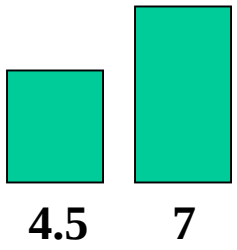




What class is this object?

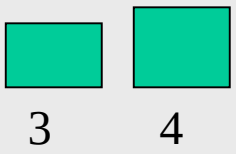


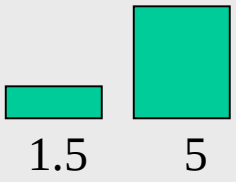
What about this one, A or B?

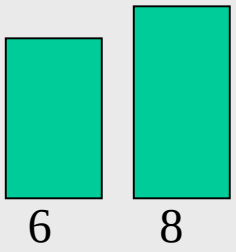


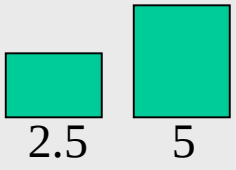
Pigeon Problem 1

Examples of class A

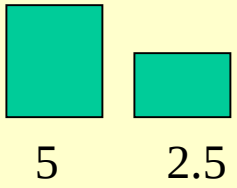


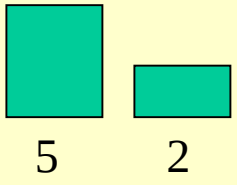


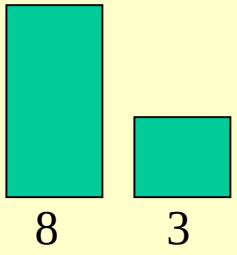


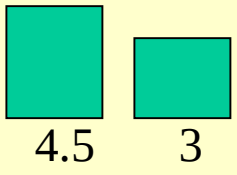


Examples of class B

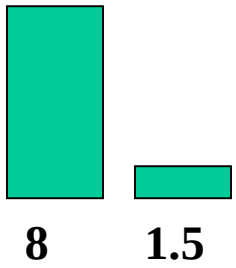








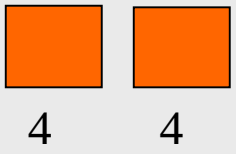
This is a B!

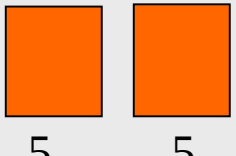


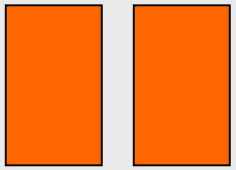
Here is the rule.
If the left bar is smaller than the right bar, it is an A, otherwise it is a B.

Pigeon Problem 2

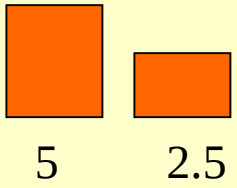
Examples of class A

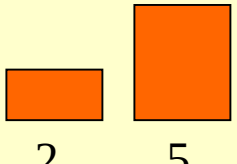


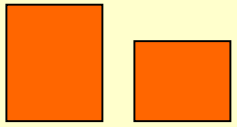


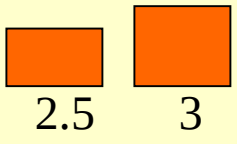


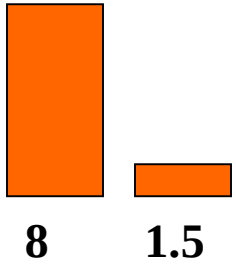
Examples of class B

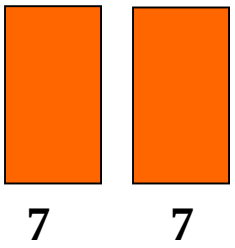






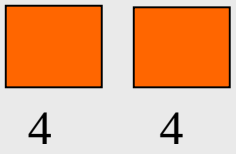


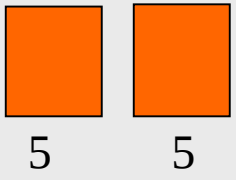


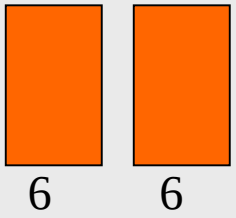


Pigeon Problem 2

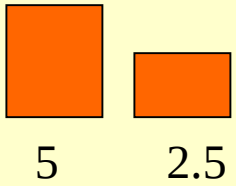
Examples of class A

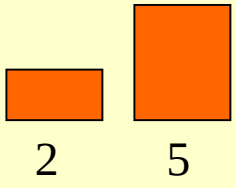


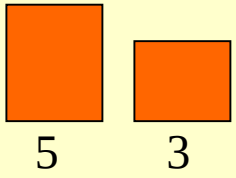


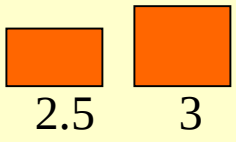


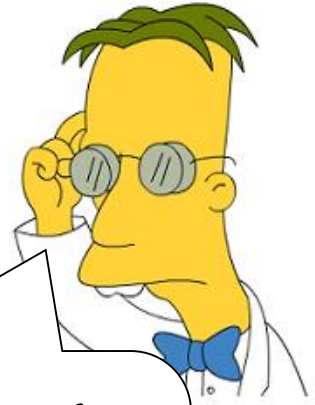
Examples of class B





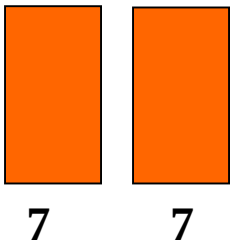






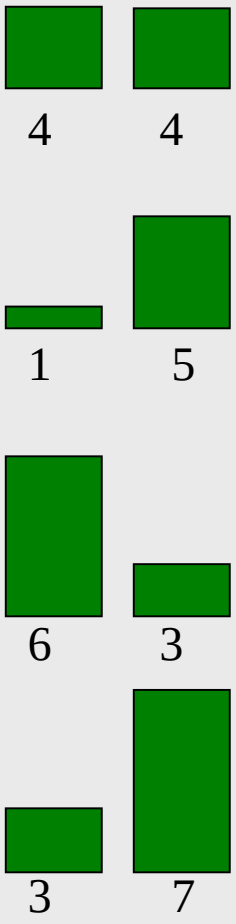
The rule is as follows, if the two bars are equal sizes, it is an A. Otherwise it is a B.

So this one is an A.

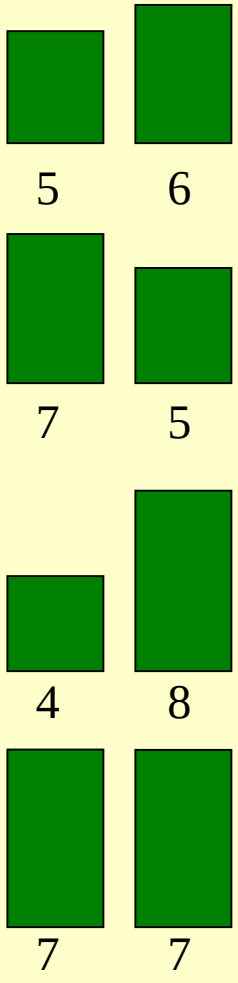


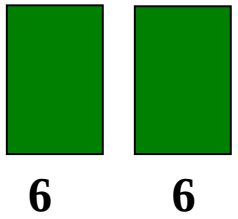
Pigeon Problem 3

Examples of class A



Examples of class B

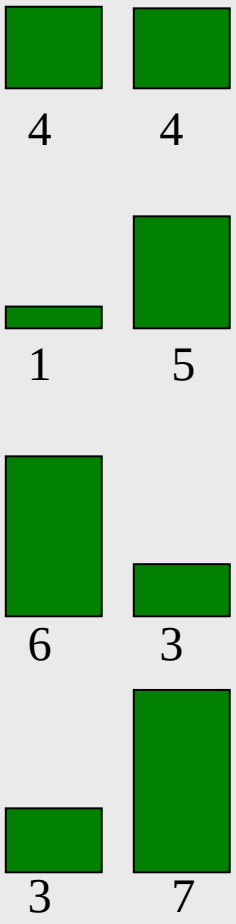




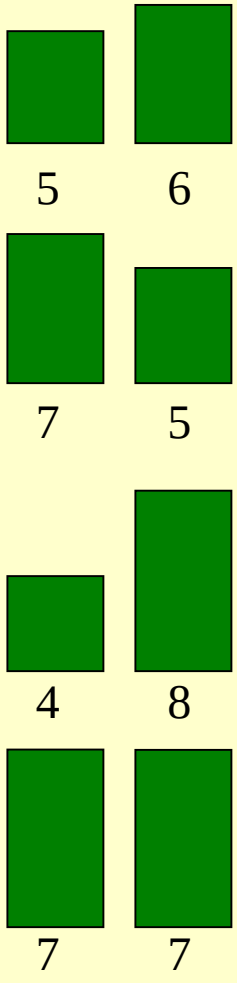
This one is really hard!
What is this, A or B?

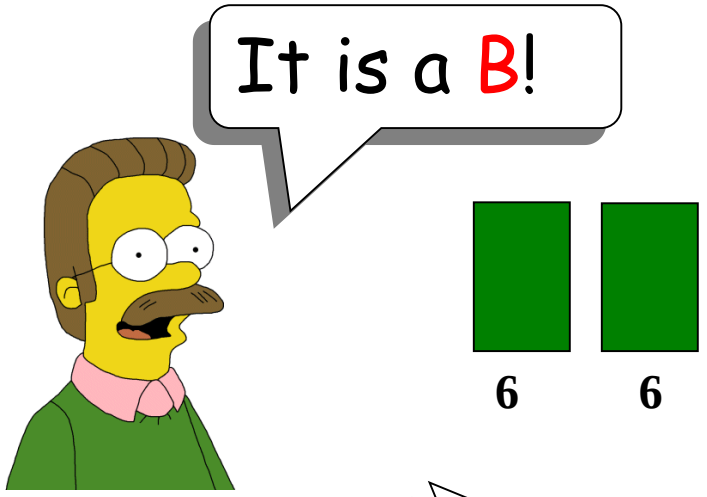
Pigeon Problem 3

Examples of class A



Examples of class B

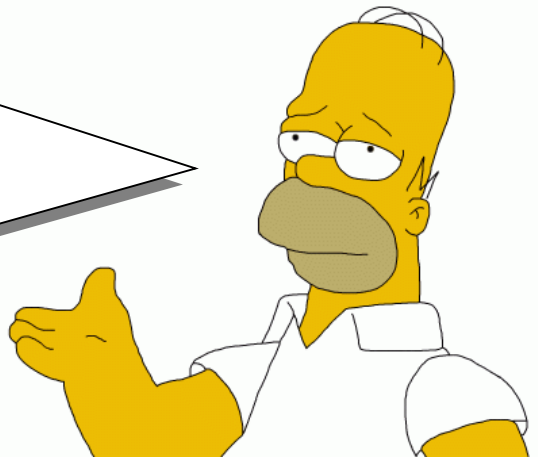




The rule is as follows, if the square of the sum of the two bars is less than or equal to 100, it is an A. Otherwise it is a B.

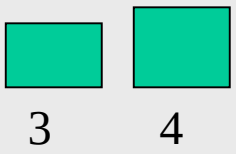
Why did we spend so much time with this game?

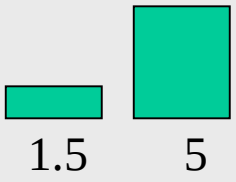
Because we wanted to show that almost all classification problems have a geometric interpretation, check out the next 3 slides…

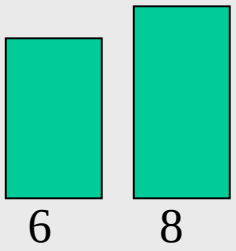


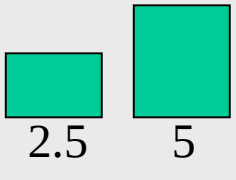
Pigeon Problem 1

Examples of class A

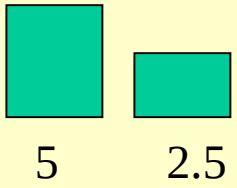


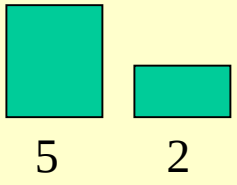


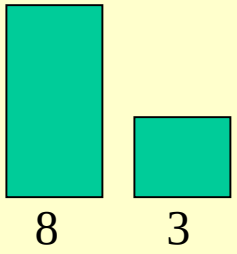


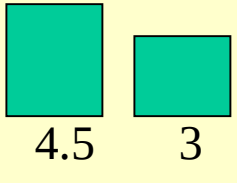


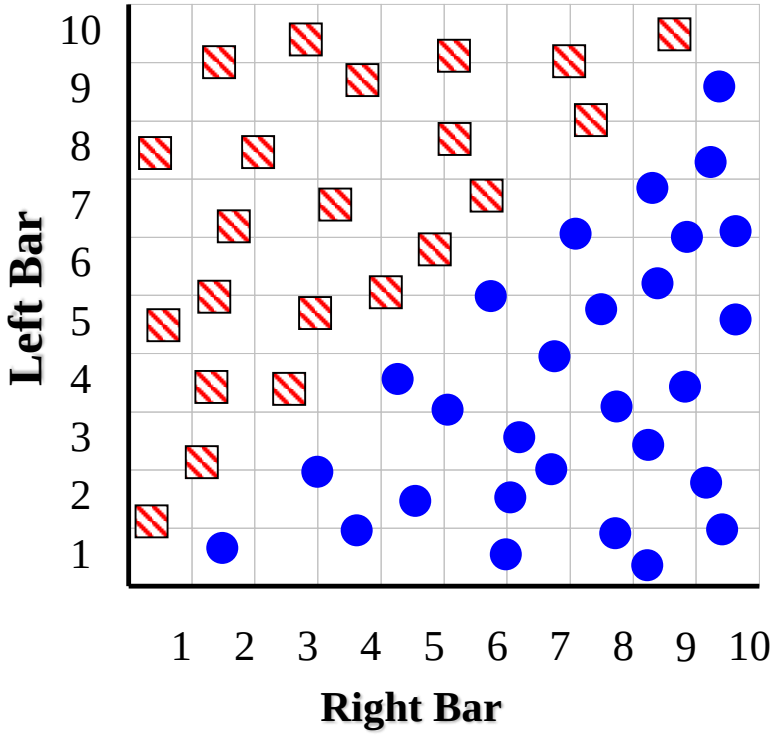
Examples of class B







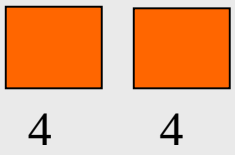


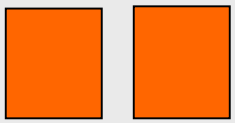


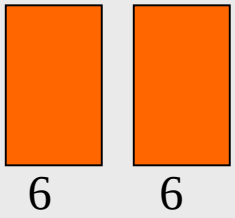
Here is the rule again.
If the left bar is smaller than the right bar, it is an A, otherwise it is a B.

Pigeon Problem 2

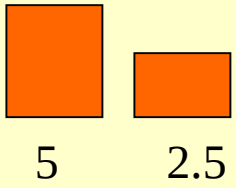
Examples of class A

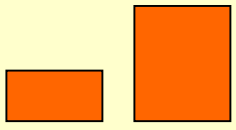


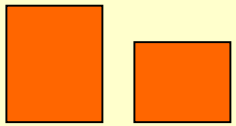


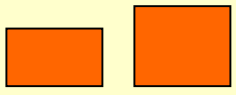


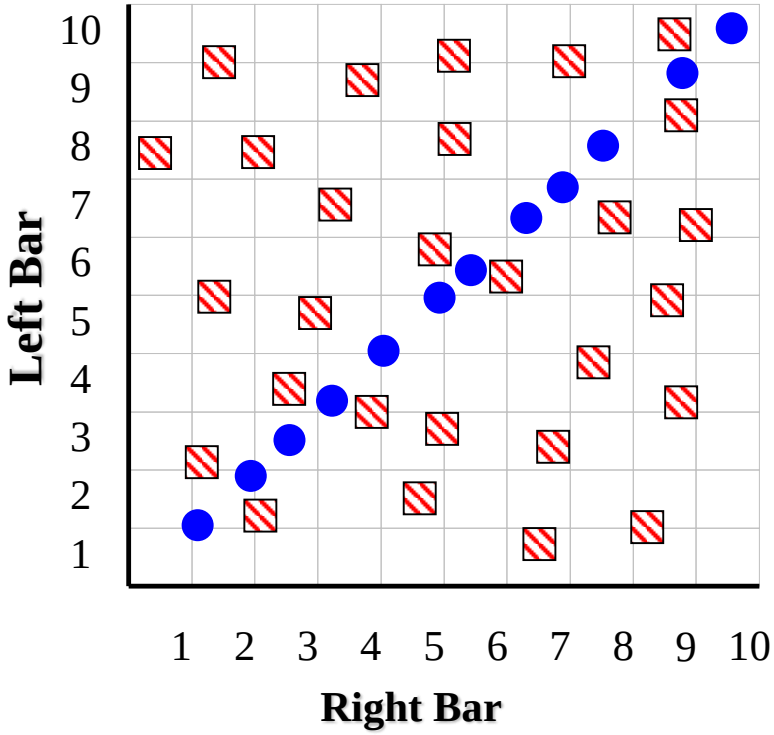
Examples of class B







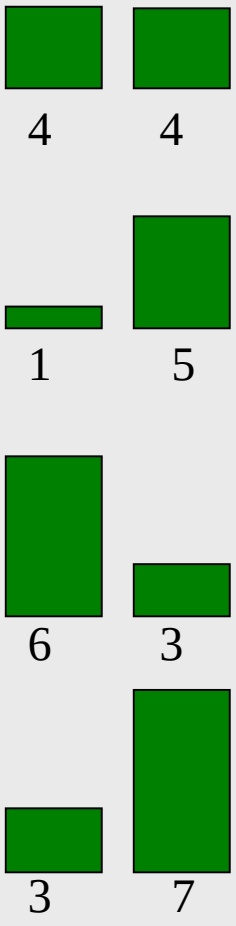




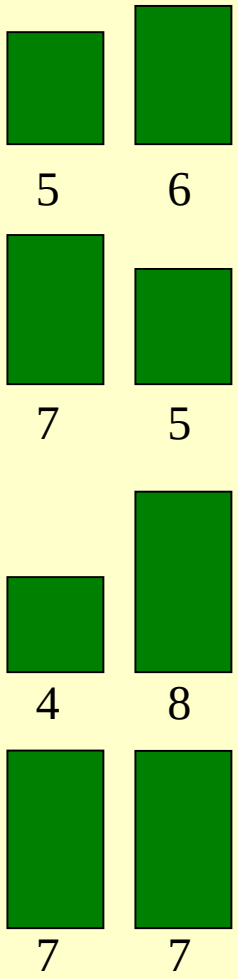
Let me look it up... here it is.. the rule is, if the two bars are equal sizes, it is an A. Otherwise it is a B.

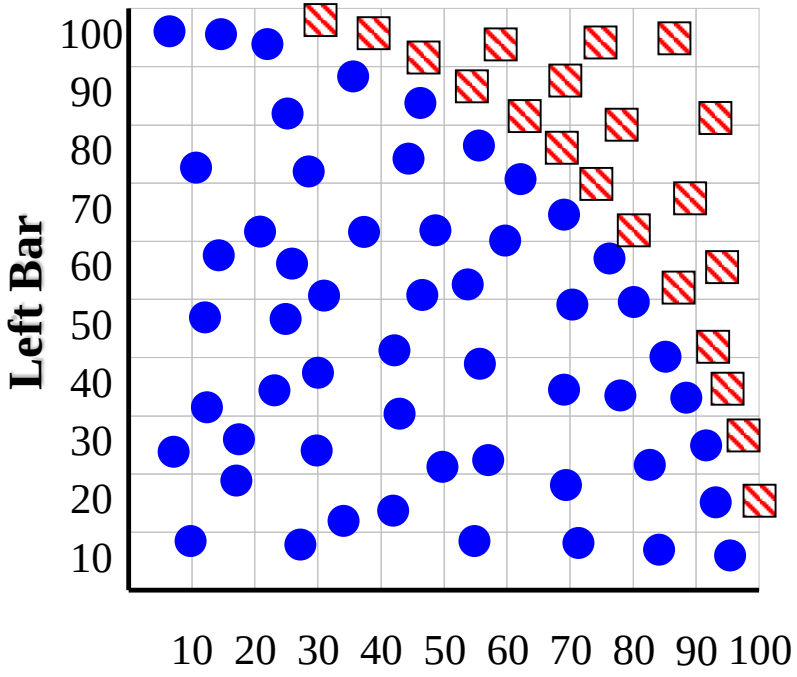
Pigeon Problem 3

Examples of class A



Examples of class B

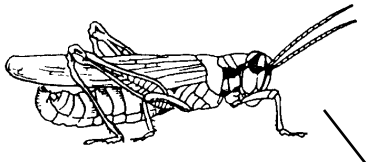


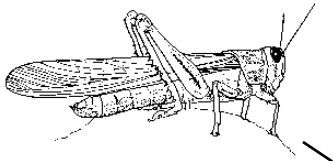


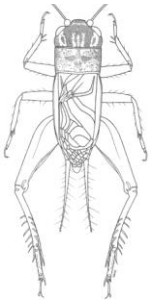
The rule again:
if the square of the sum of the two bars is less than or equal to 100, it is an A. Otherwise it is a B.

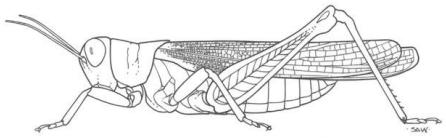
Grasshoppers

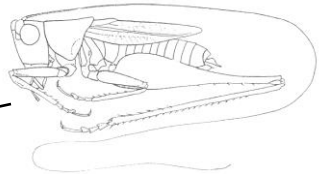
Katydids

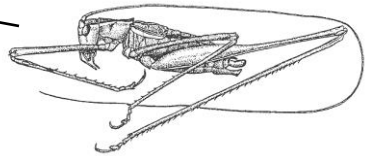


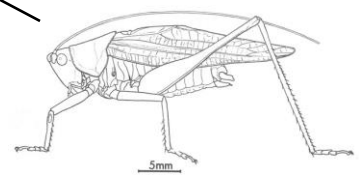


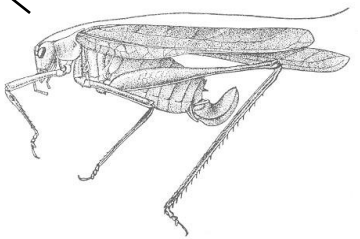


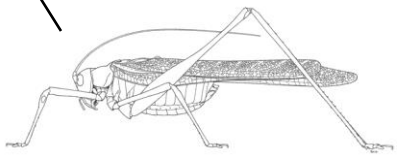


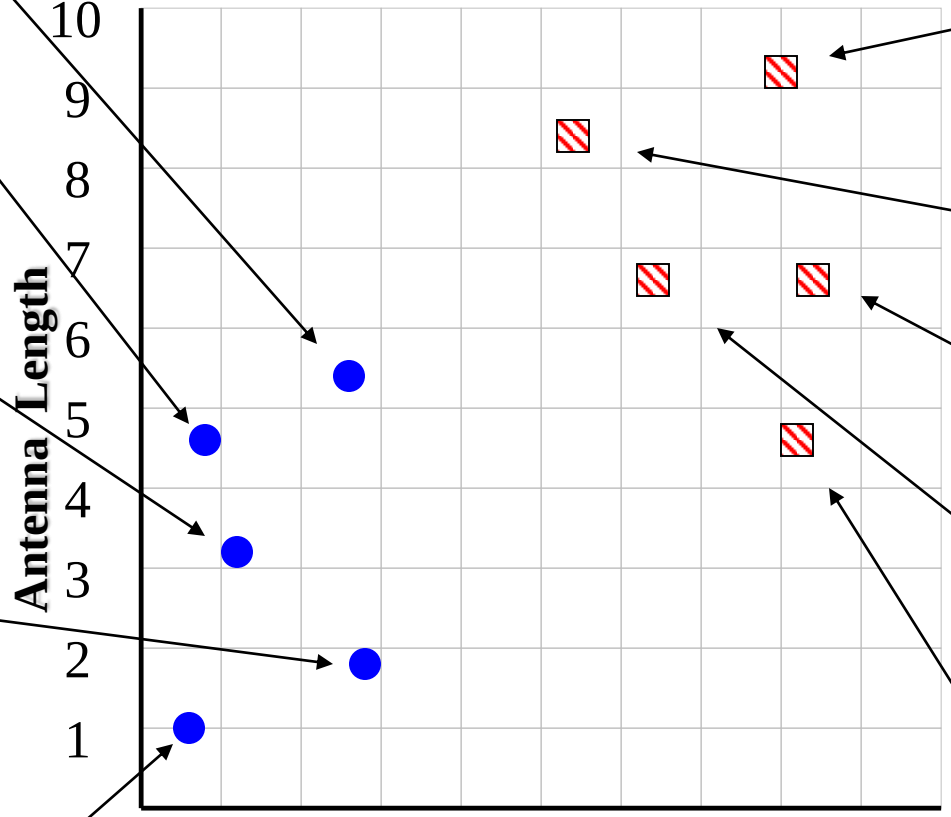




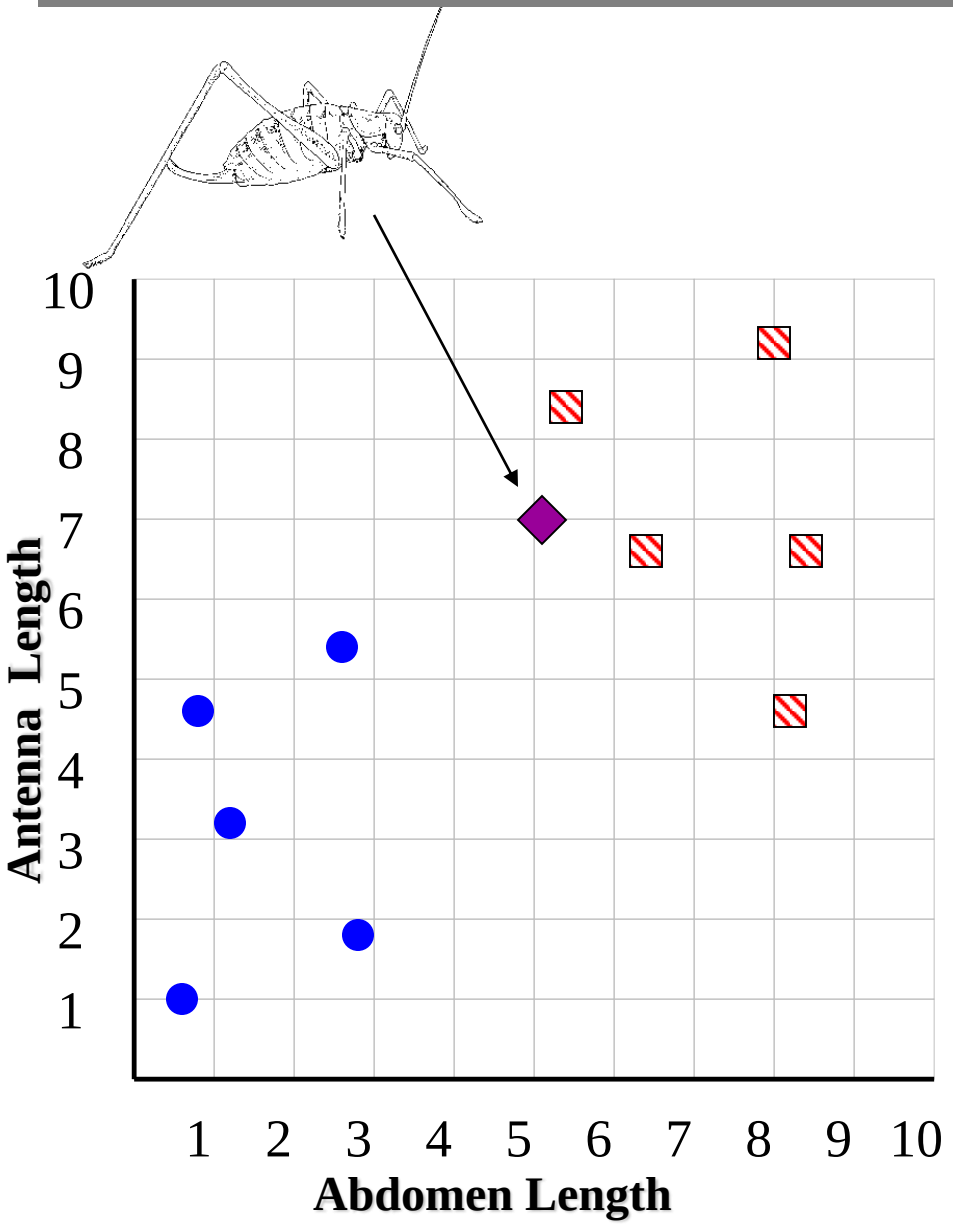








previously unseen instance =	*11*	**5.1**	**7.0**	**???????**



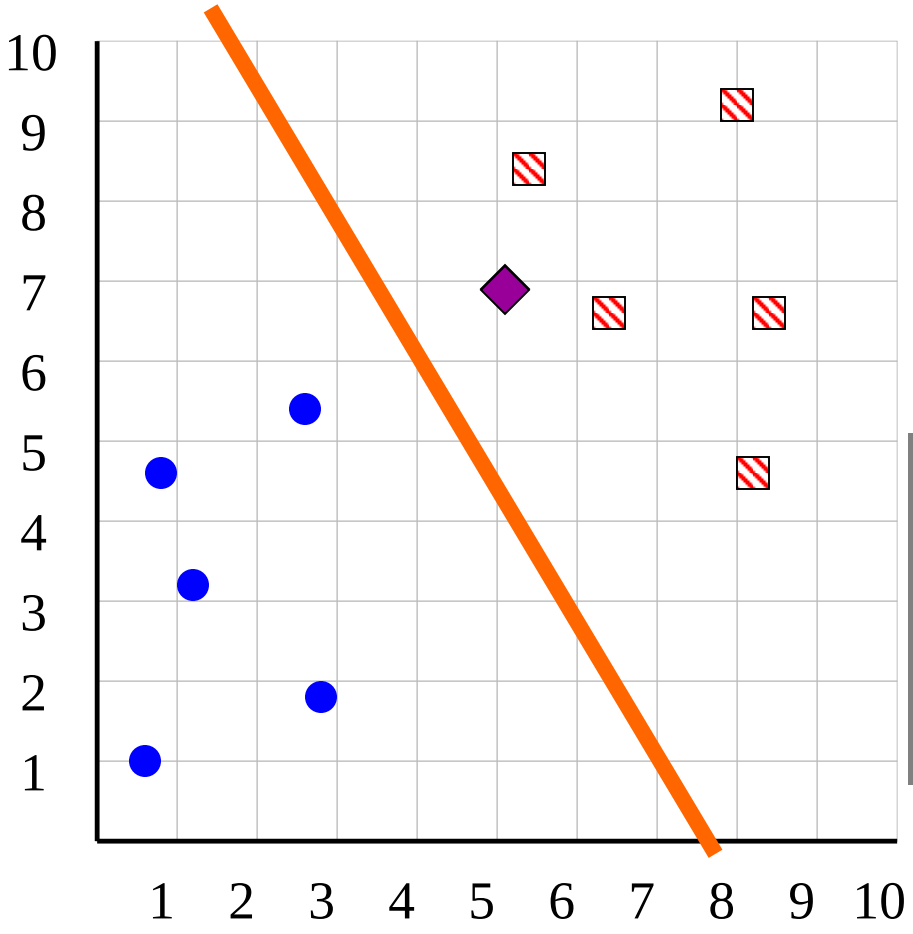
- We can "project" the previously unseen instance into the same space as the database.

- We have now abstracted away the details of our particular problem. It will be much easier to talk about points in space.

▨ **Katydids**

● **Grasshoppers**

Simple Linear Classifier

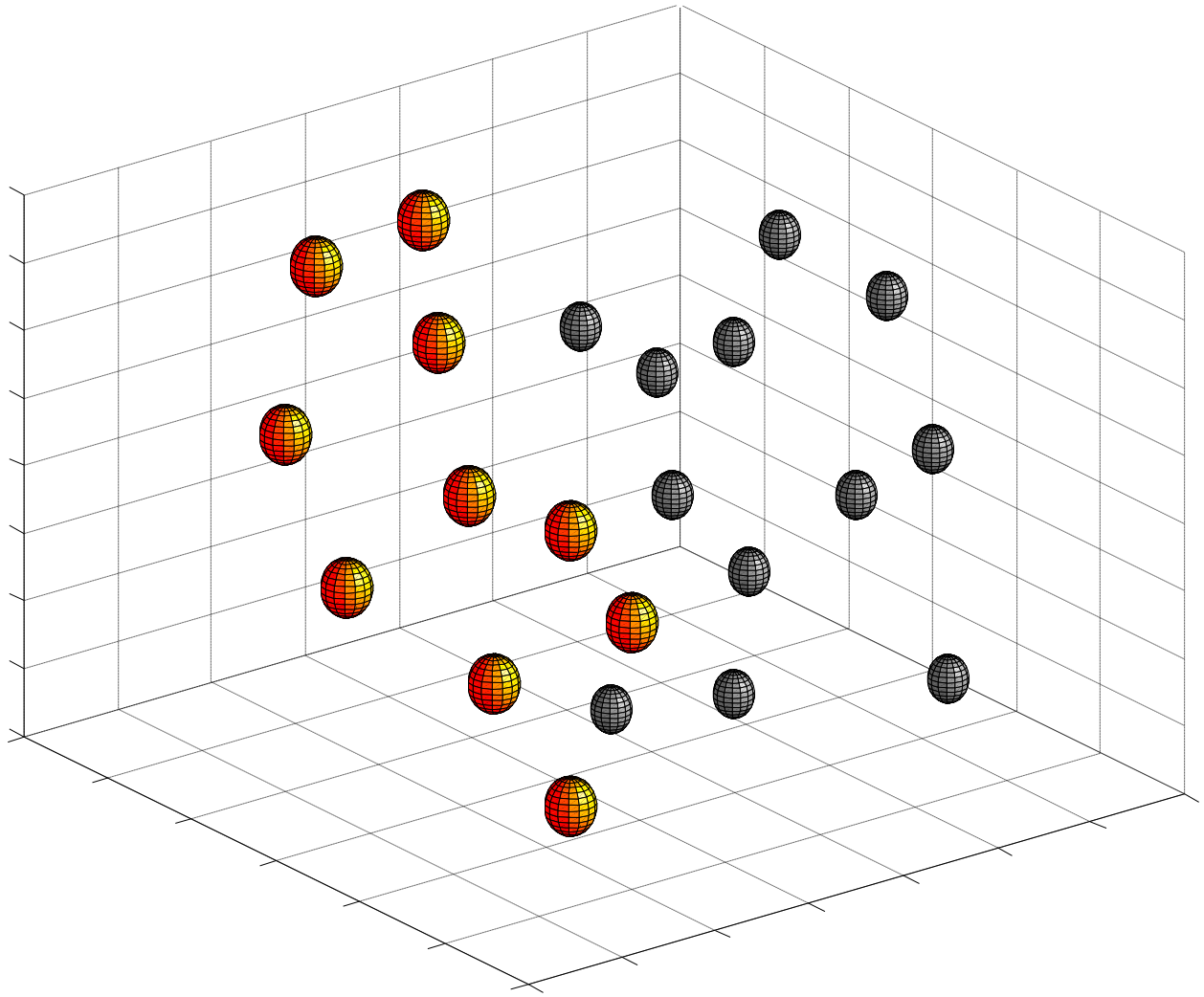


R.A. Fisher
1890-1962

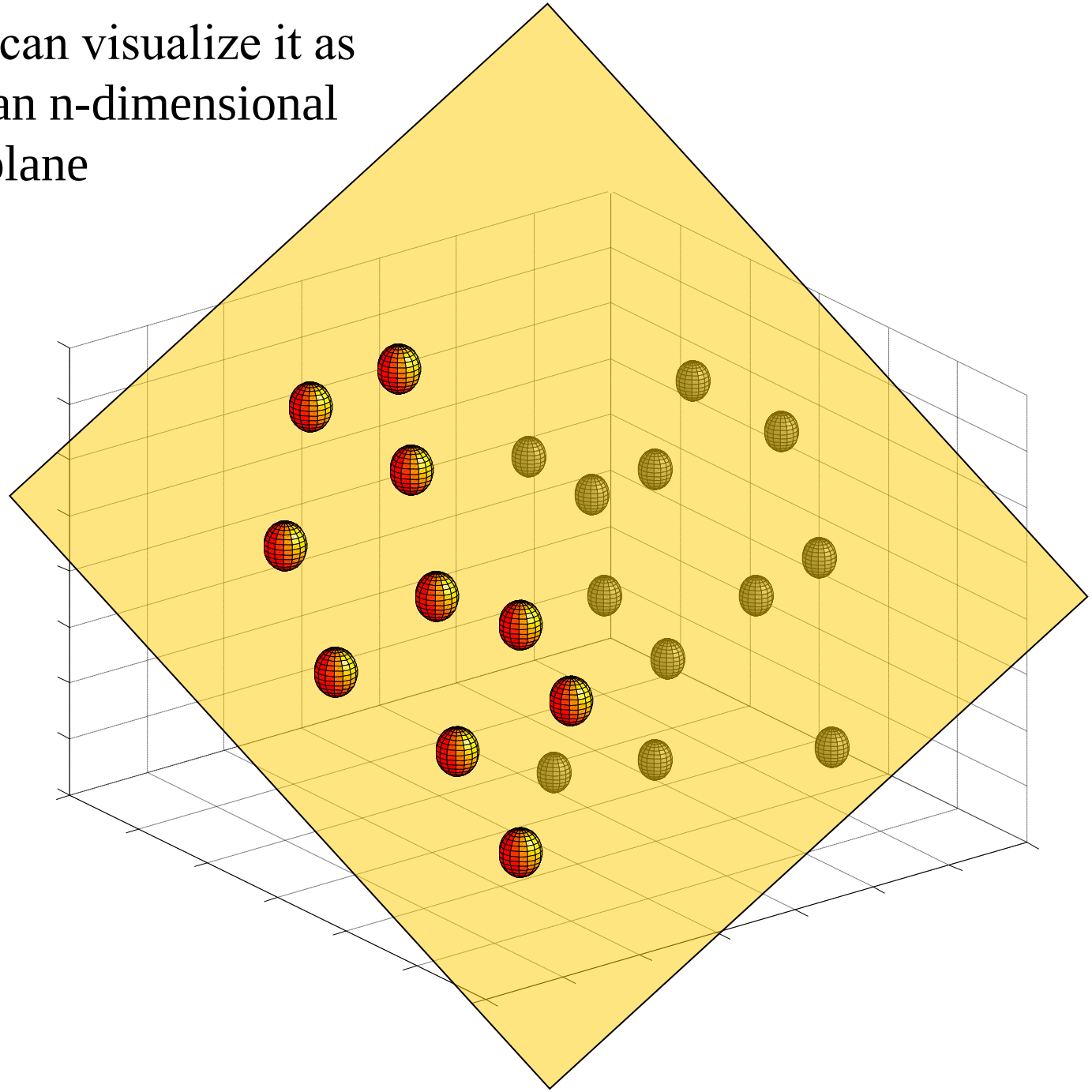
If previously unseen instance above the line
then
class is **Katydid**
else
class is **Grasshopper**

Katydids
Grasshoppers

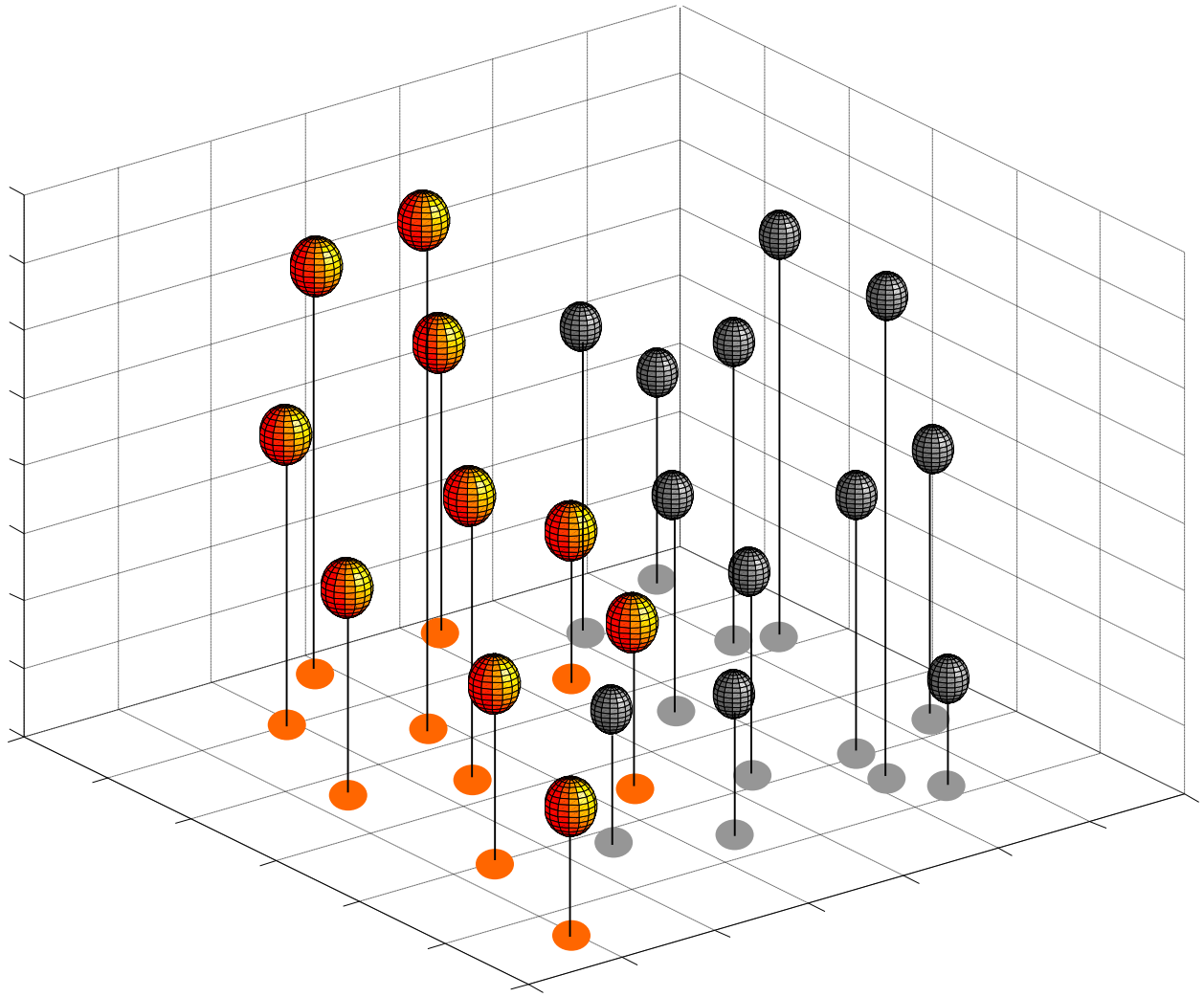
The simple linear classifier is defined for higher dimensional spaces…

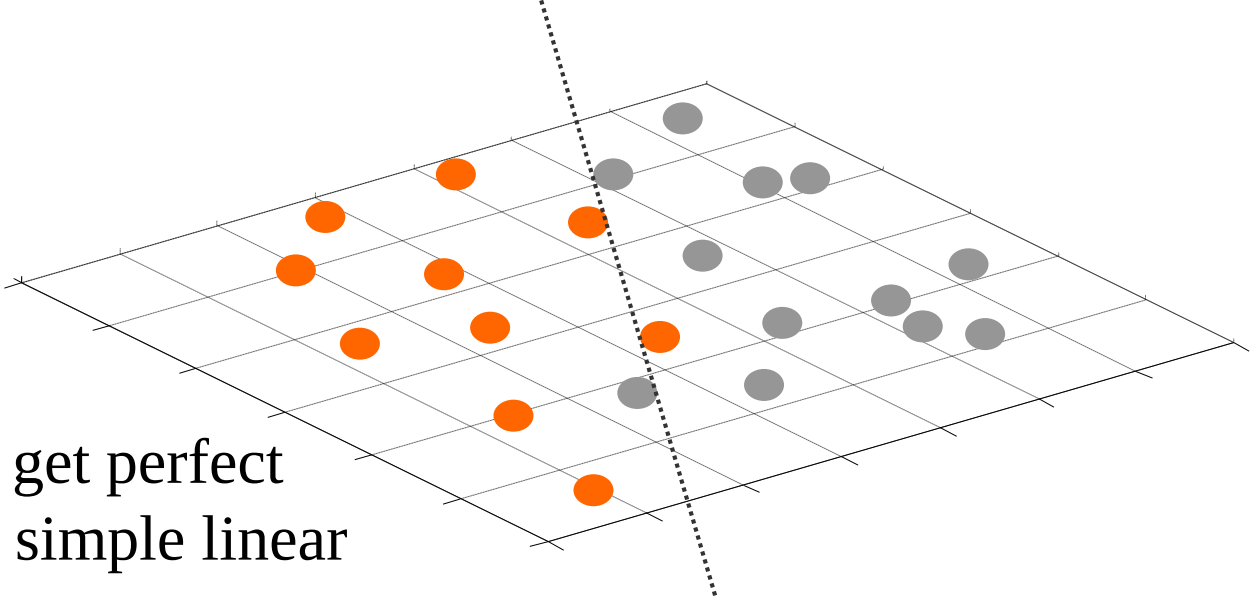


… we can visualize it as being an n-dimensional hyperplane



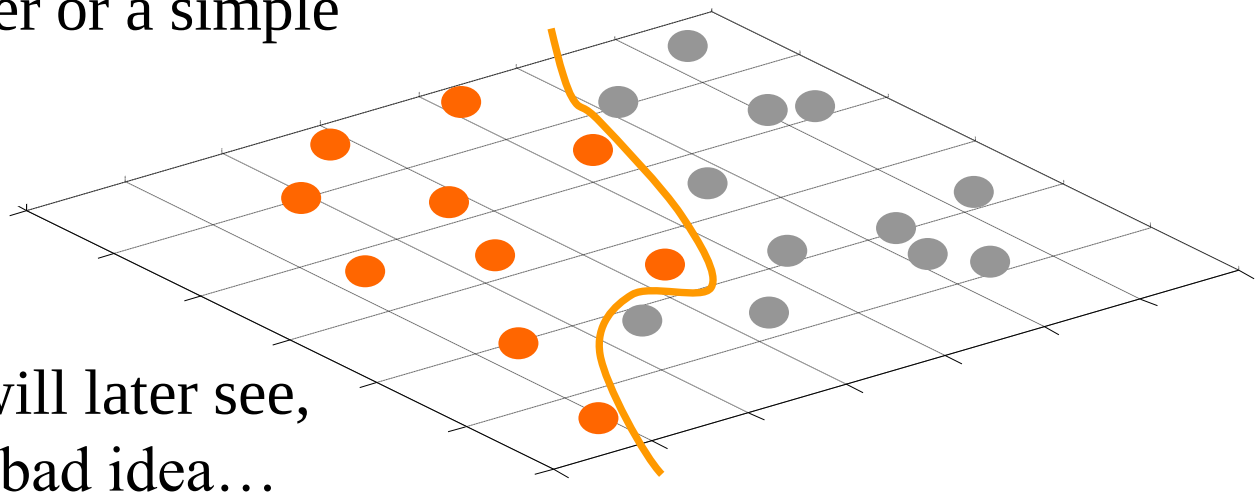
It is interesting to think about what would happen in this example if we did not have the 3rd dimension…





We can no longer get perfect accuracy with the simple linear classifier…

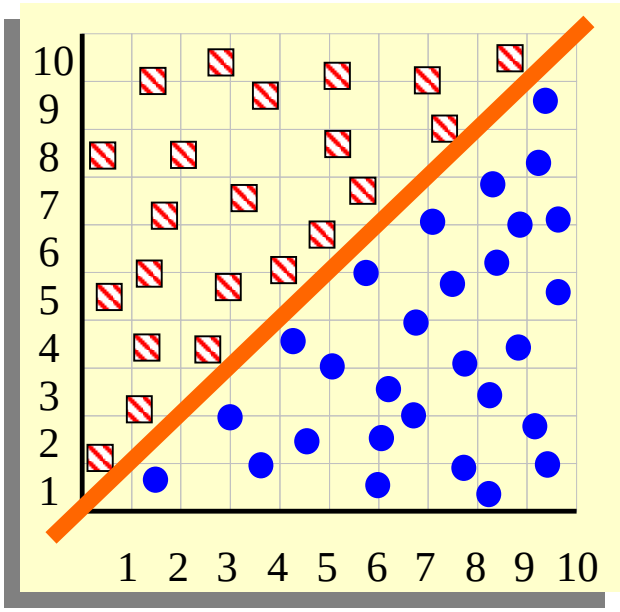
We could try to solve this problem by user a simple *quadratic* classifier or a simple *cubic* classifier..

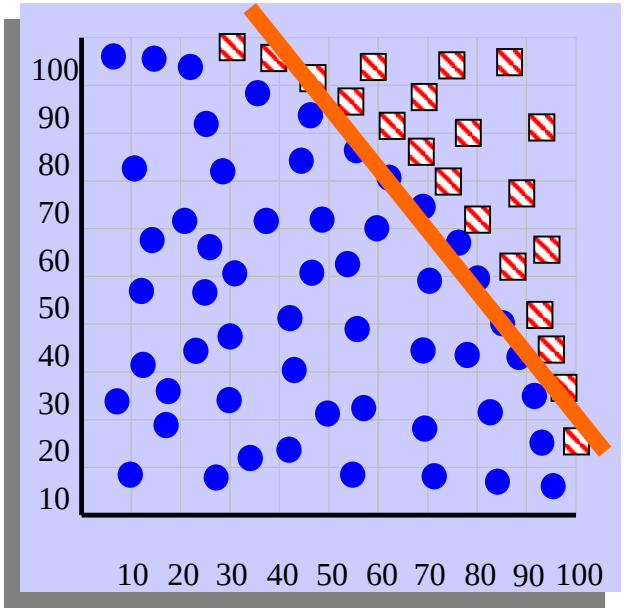


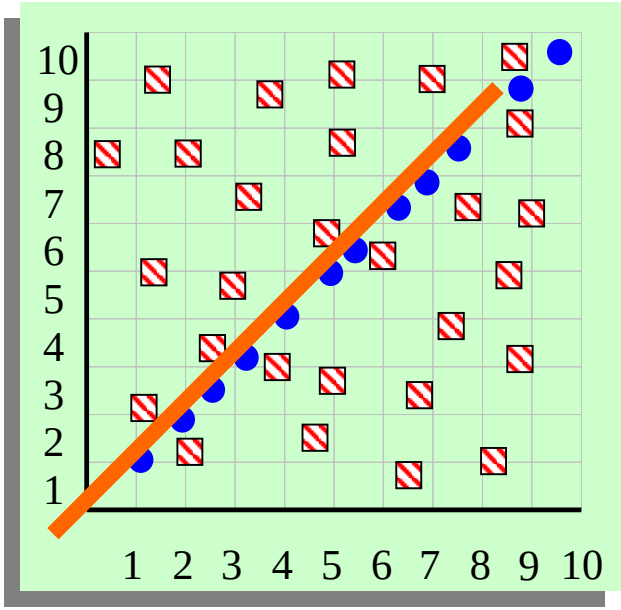
However, as we will later see, this is probably a bad idea…

Which of the “Pigeon Problems” can be solved by the Simple Linear Classifier?

1) Perfect
2) Useless
3) Pretty Good







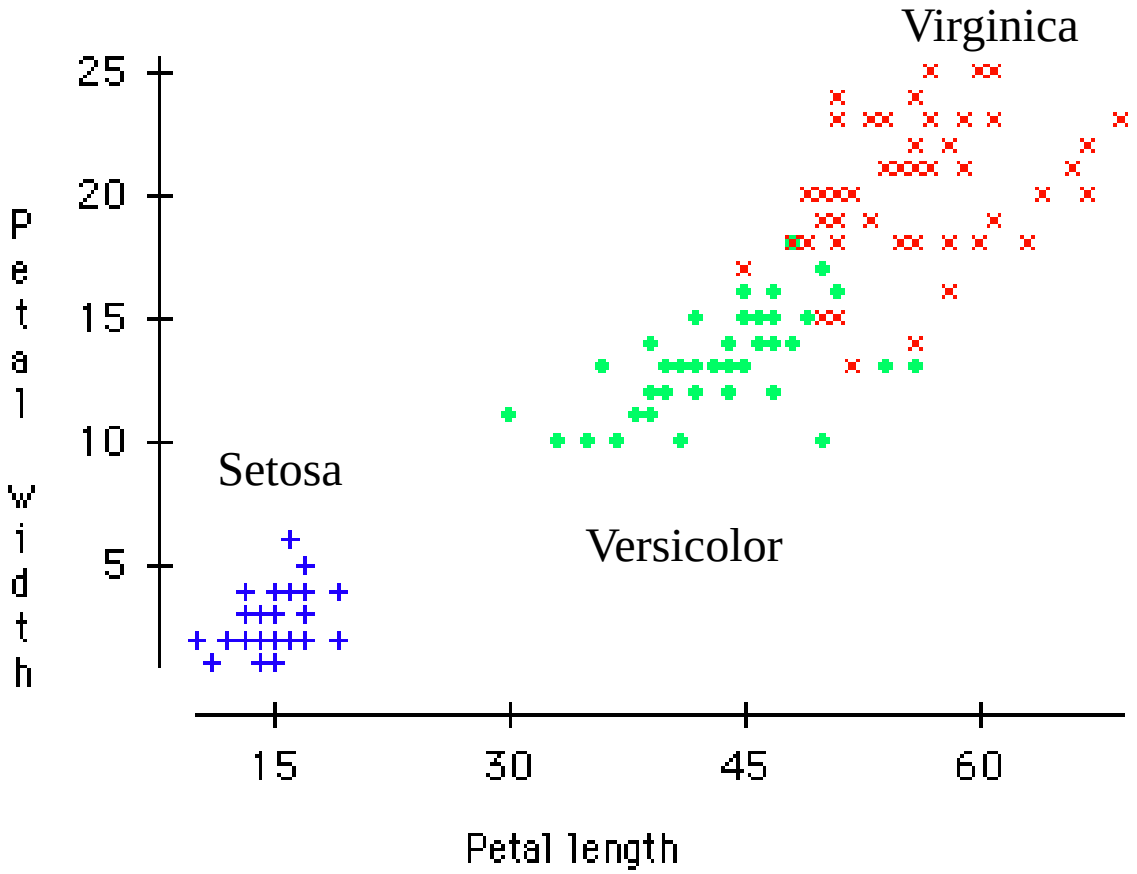
Problems that can be solved by a linear classifier are call **linearly separable**.

A Famous Problem

R. A. Fisher's Iris Dataset.

- 3 classes
- 50 of each class

The task is to classify Iris plants into one of 3 varieties using the Petal Length and Petal Width.

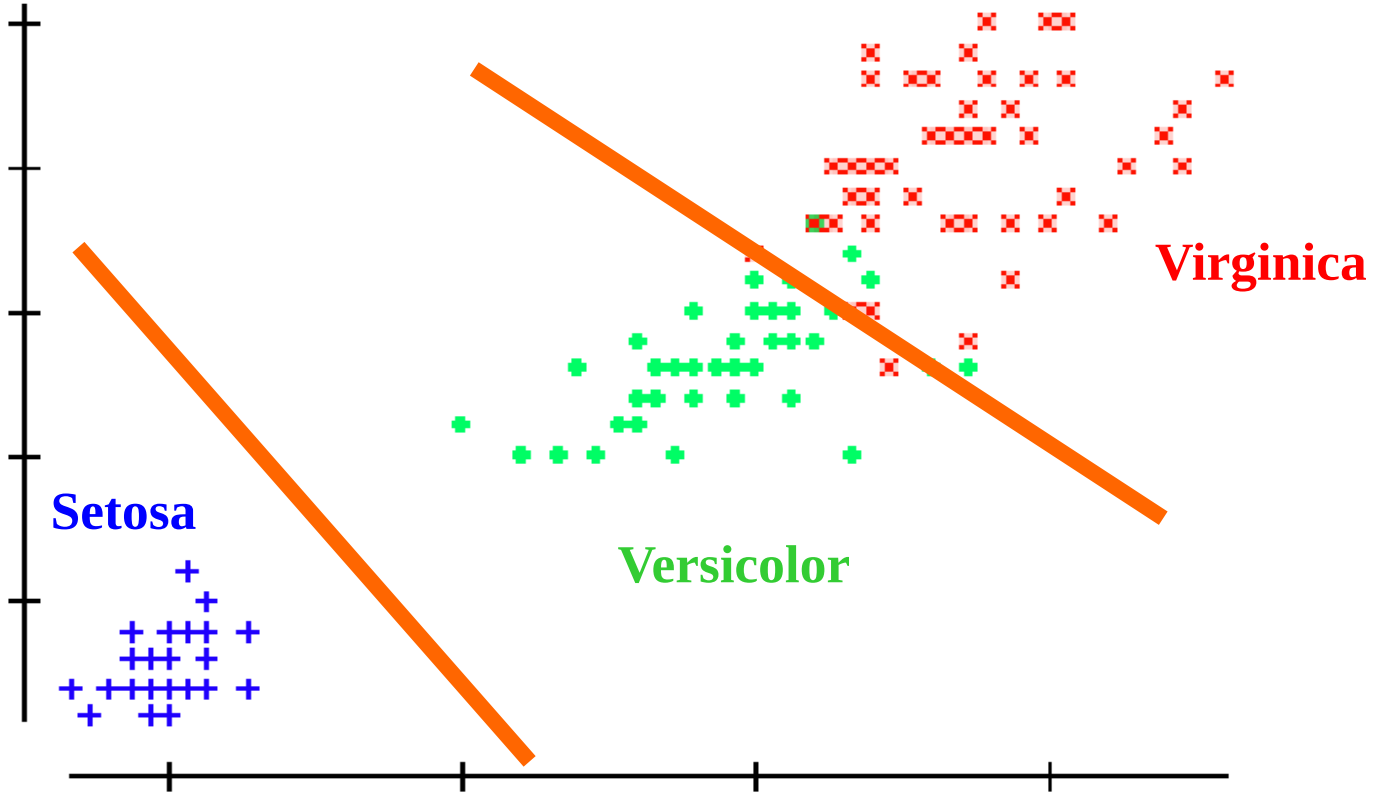


Iris Setosa

Iris Versicolor

Iris Virginica

We can generalize the piecewise linear classifier to N classes, by fitting N-1 lines. In this case we first learned the line to (perfectly) discriminate between **Setosa** and **Virginica**/**Versicolor**, then we learned to approximately discriminate between **Virginica** and **Versicolor**.



If petal width > 3.272 – (0.325 * petal length) **then** class = **Virginica**
Elseif petal width…

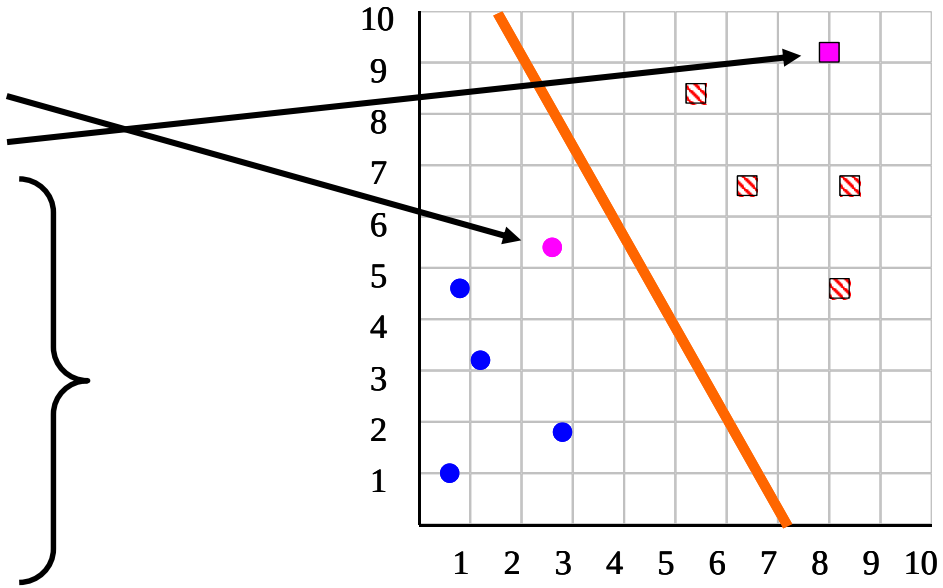
Predictive Accuracy I

- How do we *estimate* the **accuracy** of our classifier?

  We need a TESTING dataset that the classifier never saw before

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Number of instances in our database}}$$

Insect ID	Abdomen Length	Antennae Length	Insect Class
1	2.7	5.5	Grasshopper
2	8.0	9.1	Katydid
3	0.9	4.7	Grasshopper
4	1.1	3.1	Grasshopper
5	5.4	8.5	Katydid
6	2.9	1.9	Grasshopper
7	6.1	6.6	Katydid
8	0.5	1.0	Grasshopper
9	8.3	6.6	Katydid
10	8.1	4.7	Katydids



Predictive Accuracy II

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Number of instances in our database}}$$

Accuracy is a single number, we may be better off looking at a **confusion matrix**. This gives us additional useful information…

True label is...

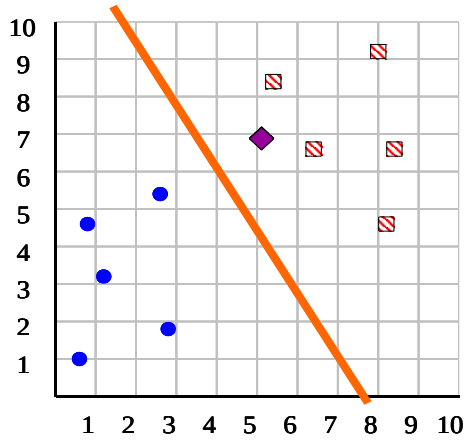
Classified as a…	Cat	Dog	Pig
Cat	**100**	**0**	**0**
Dog	**9**	**90**	**1**
Pig	**45**	**45**	**10**

Speed and Scalability I

We need to consider the time and space requirements for the two distinct phases of classification:

- Time to **construct** the classifier
  - In the case of the simpler linear classifier, the time taken to fit the line, this is linear in the number of instances.
- Time to **use** the model
  - In the case of the simpler linear classifier, the time taken to test which side of the line the unlabeled instance is. This can be done in constant time.

As we shall see, some classification algorithms are very efficient in one aspect, and very poor in the other.



Speed and Scalability II

For learning with small datasets, this is the whole picture →

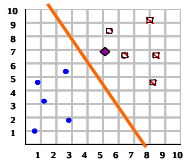
However, for data mining with massive datasets, it is not so much the (main memory) time complexity that matters, rather it is how many times we have to scan the database.

Speed and Scalability I

We need to consider the time and space requirements for the two distinct phases of classification:

- Time to **construct** the classifier
  - In the case of the simpler linear classifier, the time taken to fit the line, this is linear in the number of instances.
- Time to **use** the model
  - In the case of the simpler linear classifier, the time taken to test which side of the line the unlabeled instance is. This can be done in constant time.

As we shall see, some classification algorithms are very efficient in one aspect, and very poor in the other.



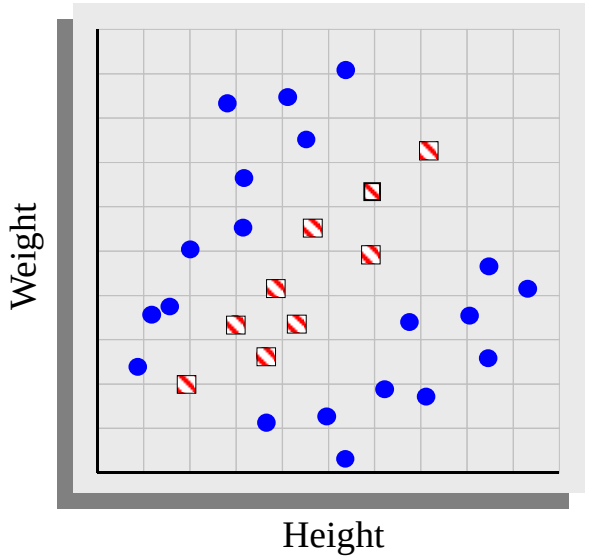
This is because for most data mining operations, disk access times completely dominate the CPU times.

For data mining, researchers often report the number of times you must scan the database.

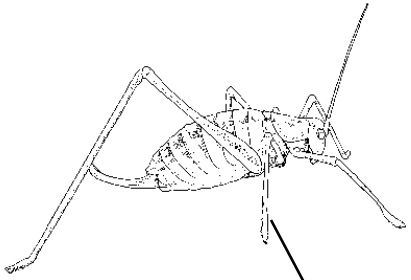
Interpretability

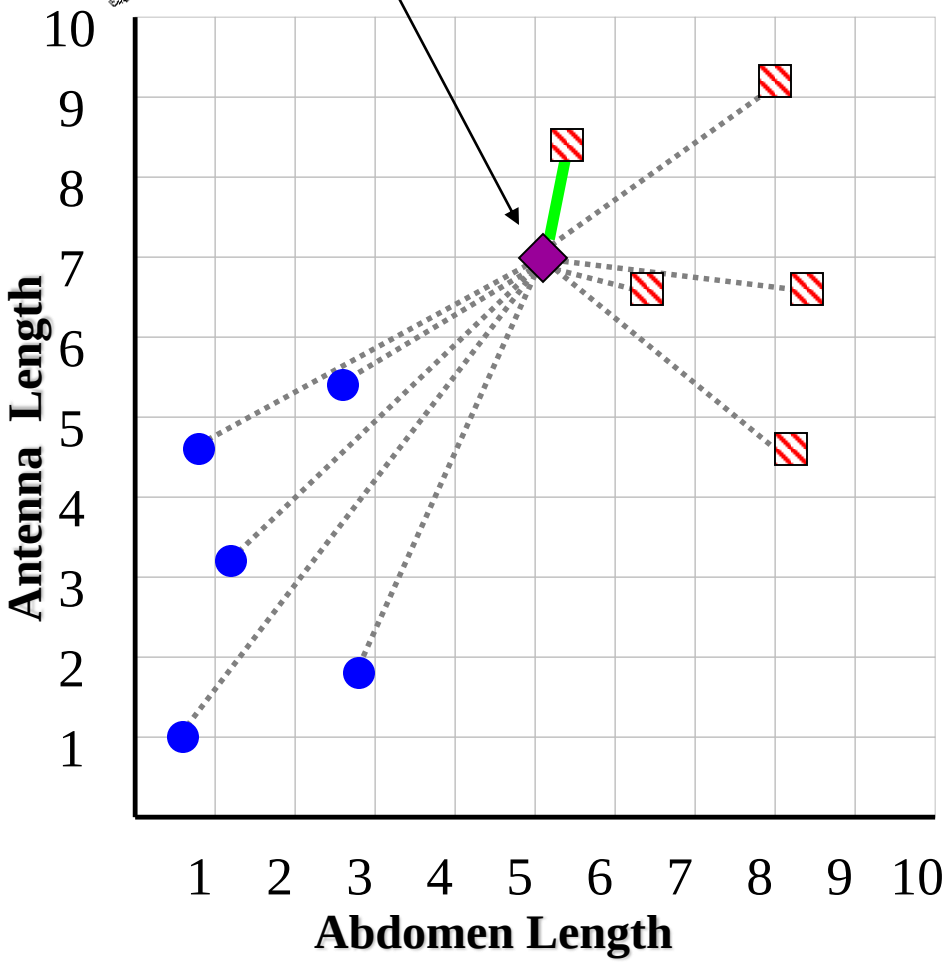
Some classifiers offer a *bonus* feature. The structure of the learned classifier tells use something about the domain.

As a trivial example, if we try to classify peoples health risks based on just their height and weight, we could gain the following insight (Based of the observation that a single linear classifier does not work well, but two linear classifiers do).
There are two ways to be unhealthy, being obese and being too skinny.



Nearest Neighbor Classifier





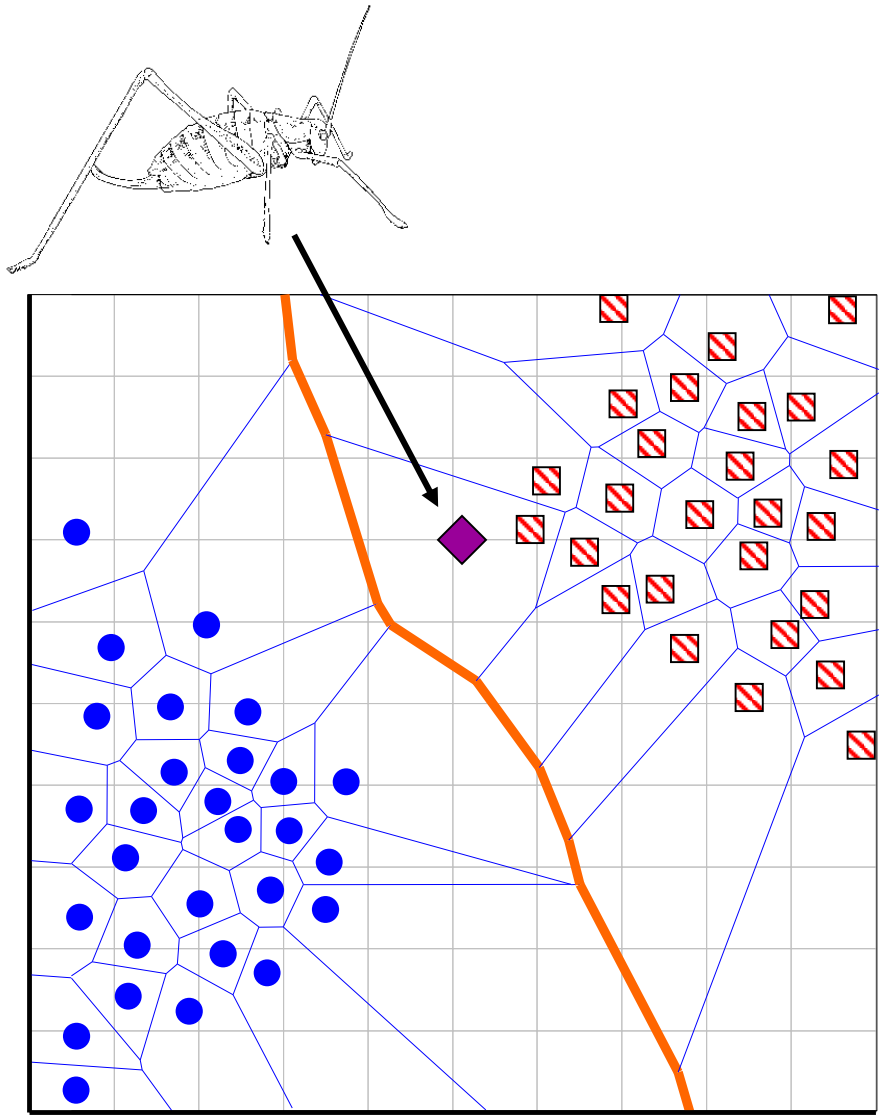
Evelyn Fix
1904-1965

Joe Hodges
1922-2000

If the **nearest** instance to the previously unseen instance **is a Katydid**
class is **Katydid**
else
class is **Grasshopper**

Katydids
Grasshoppers

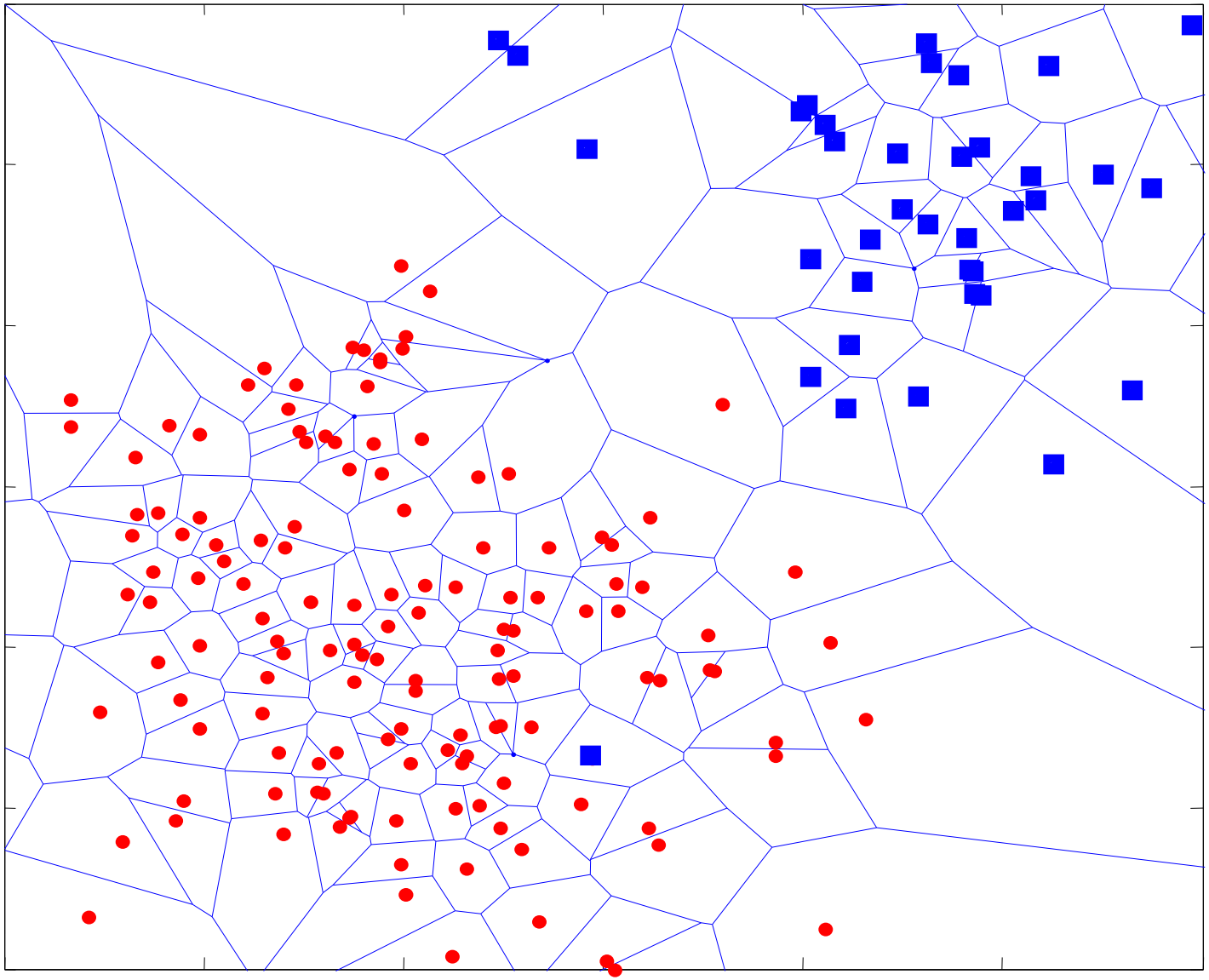
We can visualize the nearest neighbor algorithm in terms of a decision surface…



Note the we don't actually have to construct these surfaces, they are simply the implicit boundaries that divide the space into regions "belonging" to each instance.

This division of space is called Dirichlet Tessellation (or Voronoi diagram, or Theissen regions).

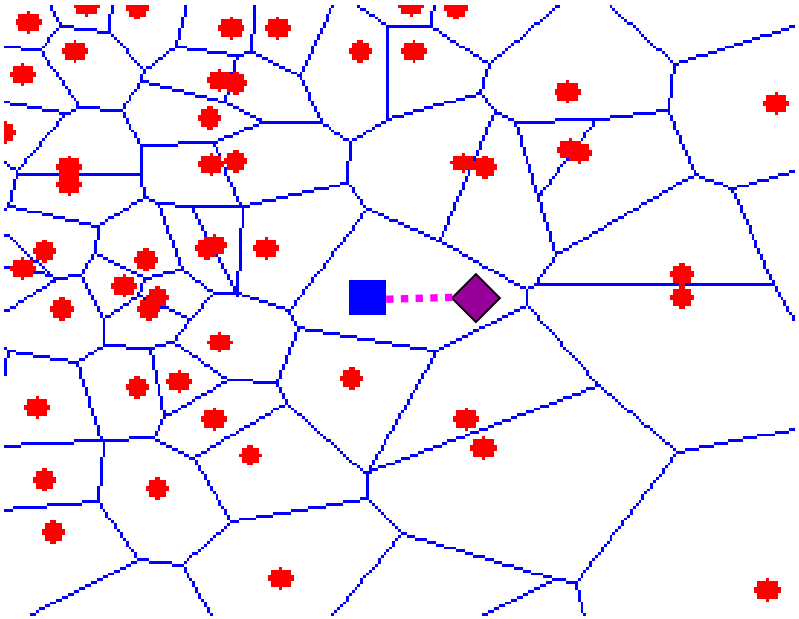
The nearest neighbor algorithm is sensitive to outliers…



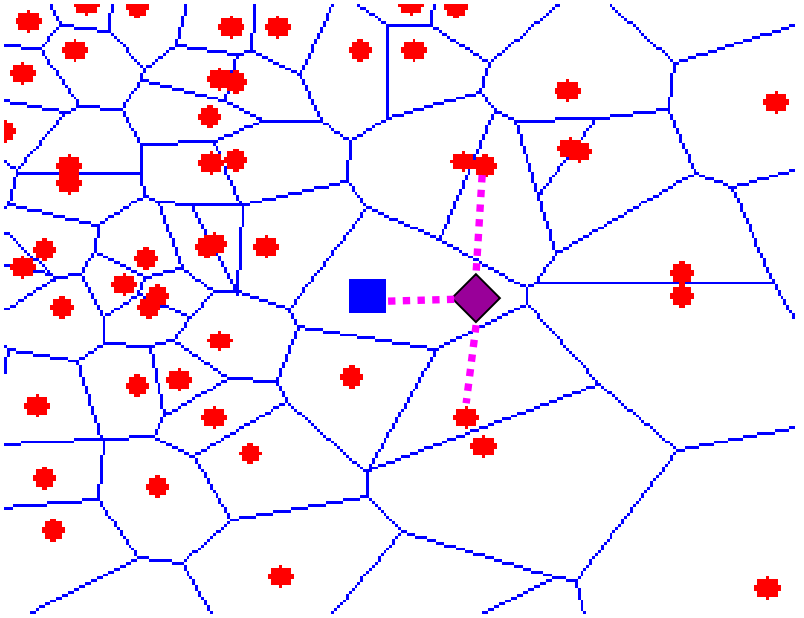
The solution is to…

We can generalize the nearest neighbor algorithm to the K- nearest neighbor (KNN) algorithm.

We measure the distance to the nearest K instances, and let them vote. K is typically chosen to be an odd number.



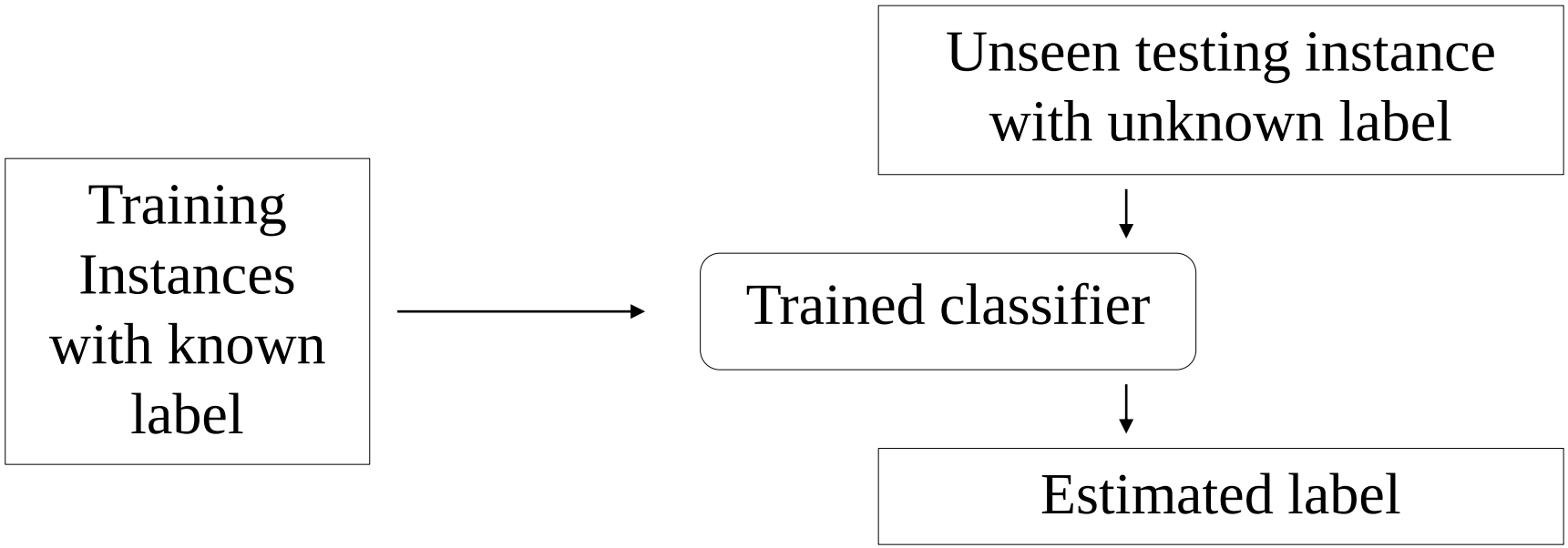
K = 1



K = 3

Real classifiers are more complex

- Based on various statistical techniques
- All classifiers can be seen as a black box



We now build our own classifier!

- Input: image of an hand
- Output: ROCK, PAPER, SCISSOR

First step is to build a dataset. Let's shoot many pictures!