# Introduction to Data Science 4

# Overview

Train, test and validation sets

The number of features and the Curse of Dimensionality

Precision, Recall, F1 score

Principal Component Analysis

# Machine Learning Procedure

**TRAIN set**: is used for creating the model

**VALIDATION set**: is used for evaluating the model

Train en validation set may be part of a cross-validation procedure

If the model is optimised by selecting the “best” parameter (e.g., “k” in case of kNN), then you are effectively overfitting. In this case you need a

**TEST set: is** used for evaluating the optimised model

# Example

Suppose you have 1000 instances. You divide these in 500+250+250 instances.

- Training set: 500 instances
- Validation set: 250 instances

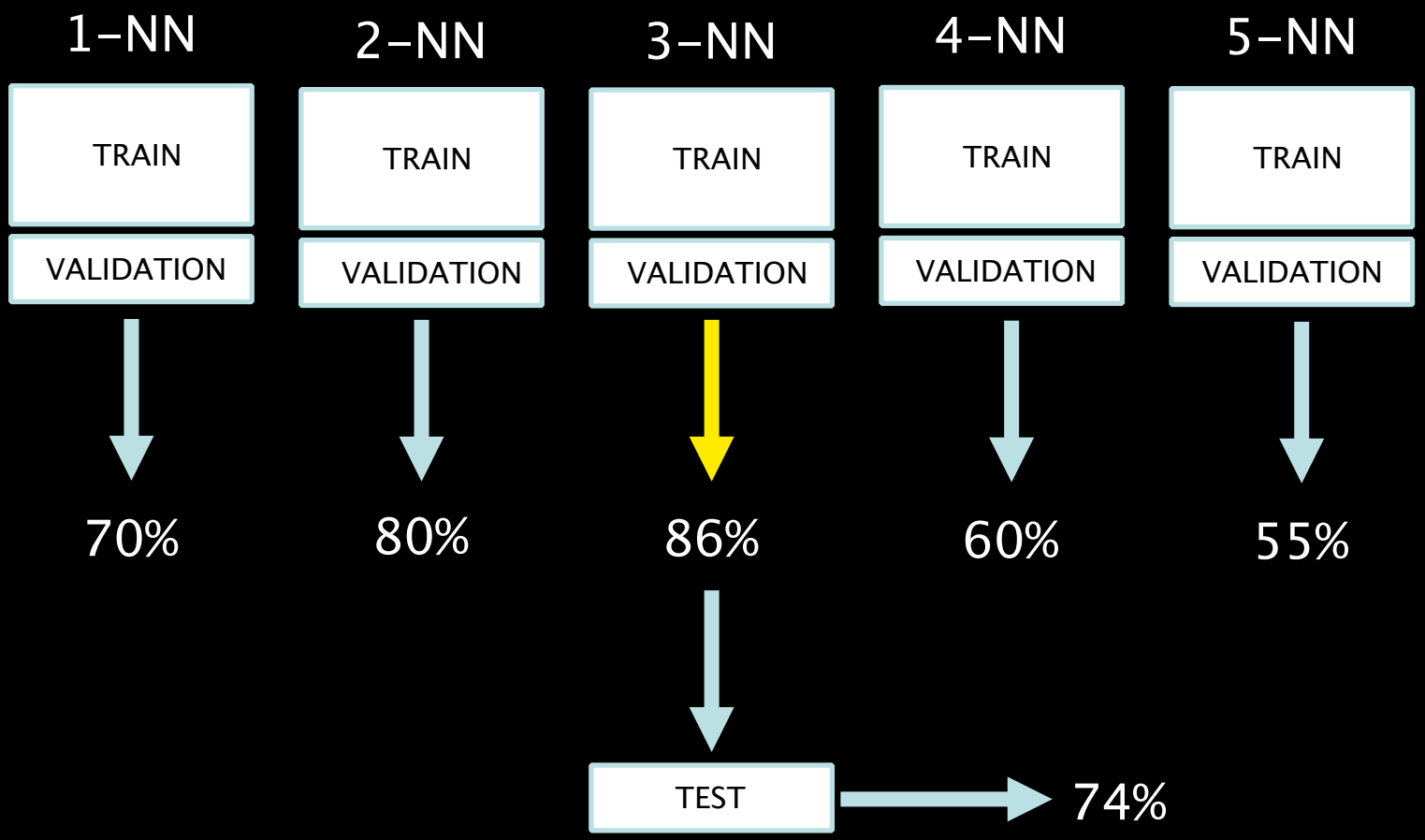
(you may perform cross validation)

With these sets you find the best k value.

- Test set: 250 instances

The performance of the test set is an estimate of the prediction performance on unseen instances.

# Parameter optimisation (example, see also p.149 WEKA book)



# Classification and regression tasks

How many features are needed?

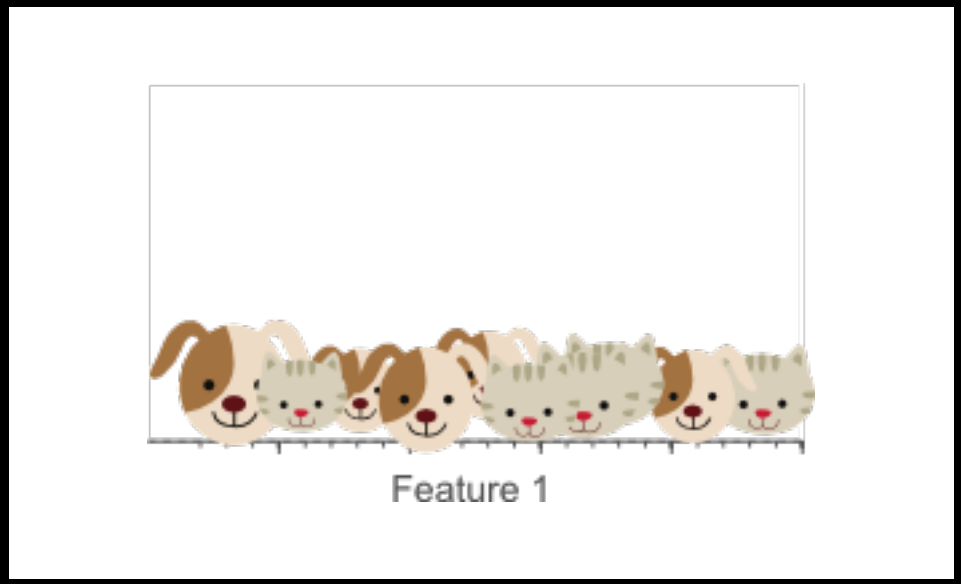
More features capture more information about the task

But too many features hamper generalisation performance

# More features may be good

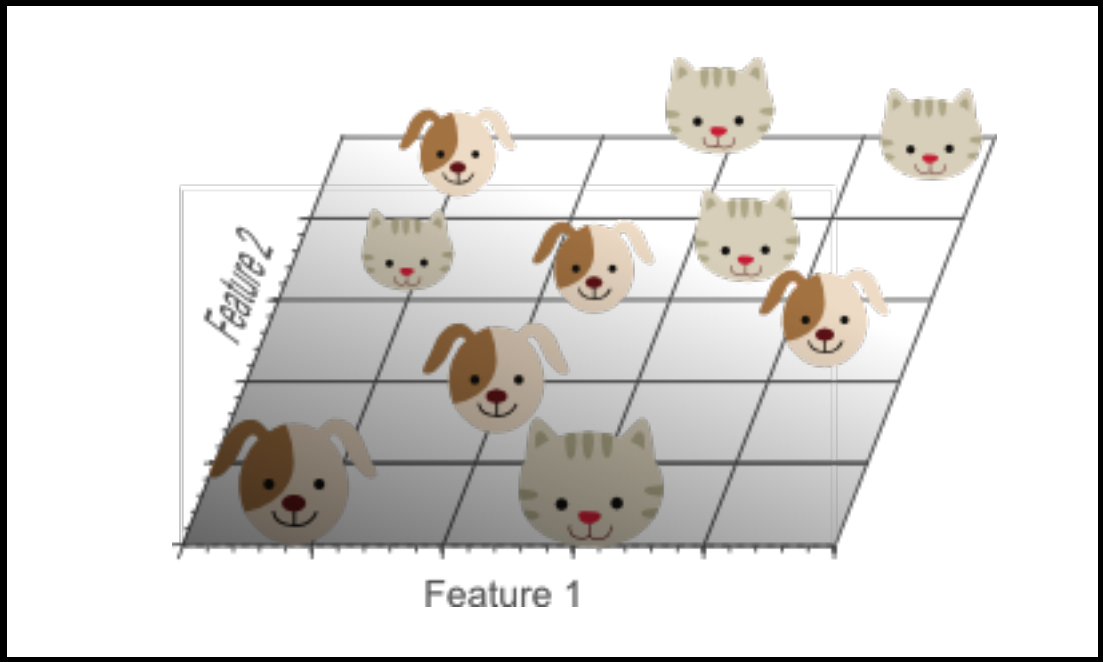
We consider a simple linear decision boundary to solve the CAT-DOG classification task…

# More features may be good (1)



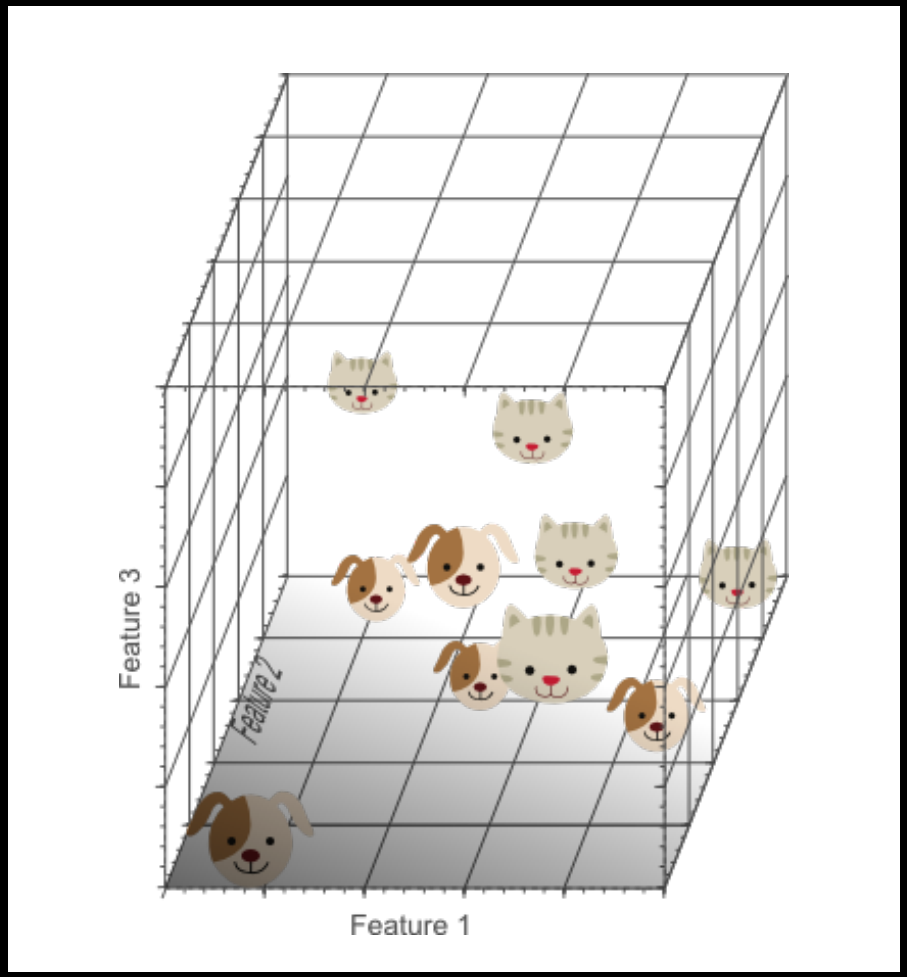

1 feature is insufficient to separate cats and dogs

# More features may be good (2)



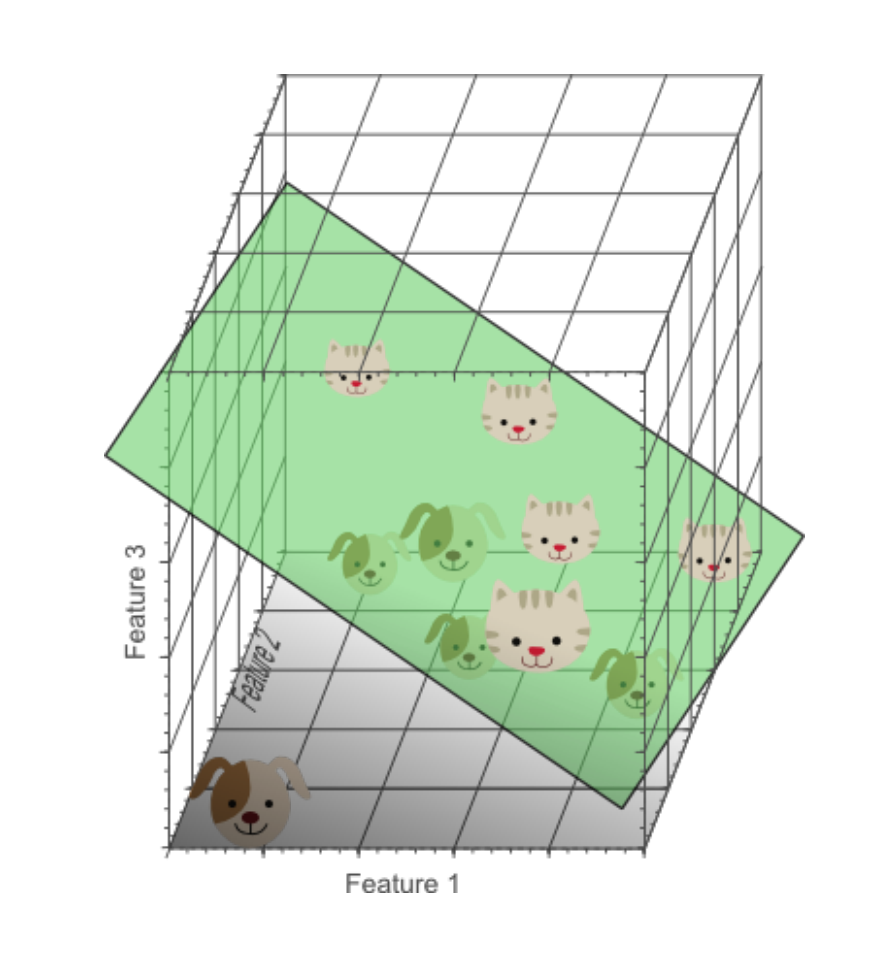

2 features are insufficient to separate cats and dogs

# More features may be good (3)




3 features are sufficient to separate cats and dogs

# More features may be good




3 features are sufficient to separate cats and dogs

# More features may be bad

We assume that the CAT-DOG classification task requires 20% of the data for training…

# Cats versus Dogs

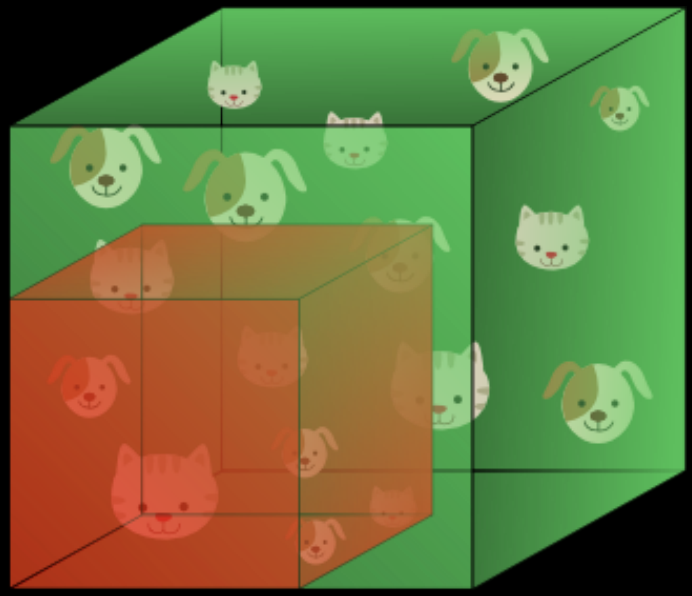
The amount of training data needed to cover 20% of the feature ran

1 feature

(1-dimensional feature space)

2 features

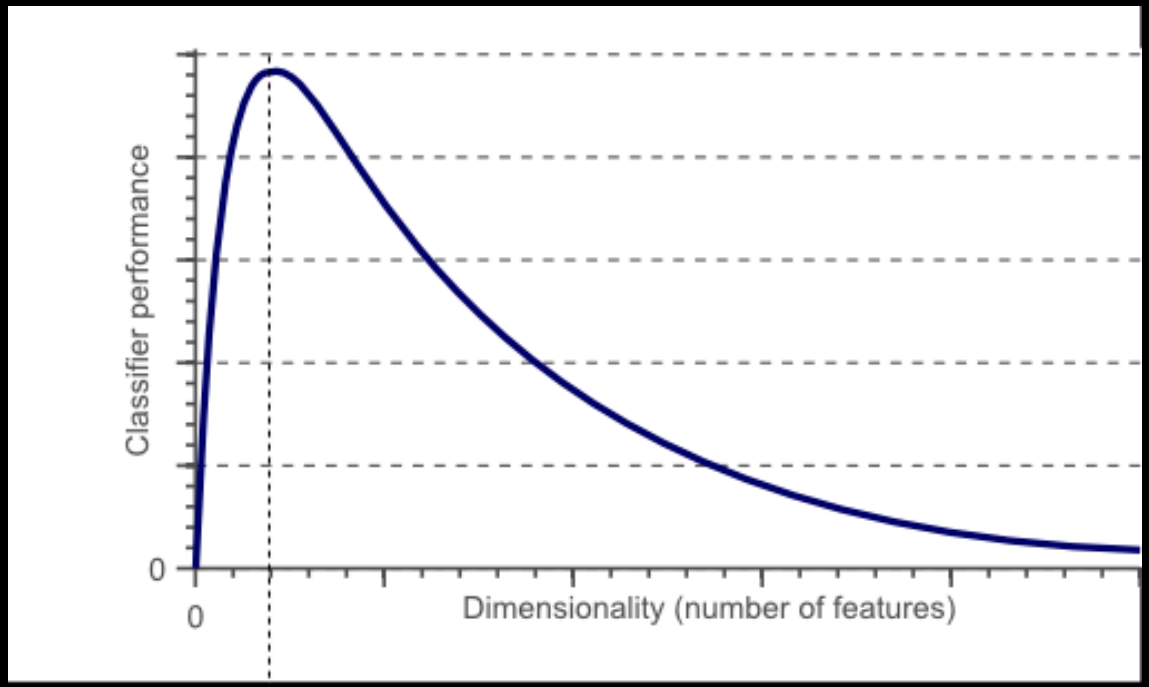
(2-dimensional feature space)



3 features

(3-dimensional feature space)

# Curse of Dimensionality




- If the number of features becomes too large, generalisation performance drops

# Machine Learning Trade-off

We want to increase the number features that may contain relevant information for the classification or regression task

versus

We want to reduce the number of features to counter the curse of dimensionality

# How to counter the curse?

Feature selection

Dimensionality reduction (PCA)

# Feature Selection

Can be performed automatically. e.g., in decision trees (“pruning”), or

Can be performed manually:

- using domain knowledge
(which may be wrong!)
- or common sense
(which may also be wrong!)

# Confusion Table

| | predicted POSITIVE | predicted NEGATIVE |
|---|---|---|
| actual POSITIVE | TRUE POSITIVE | FALSE NEGATIVE |
| actual NEGATIVE | FALSE POSITIVE | TRUE NEGATIVE |

# PRECISION =

| | predicted POSITIVE | predicted NEGATIVE |
|---|---|---|
| actual POSITIVE | TRUE POSITIVE | FALSE NEGATIVE |
| actual NEGATIVE | FALSE POSITIVE | TRUE NEGATIVE |

$$\frac{\text{TRUE POSITIVES}}{\text{TRUE POSITIVES} + \text{FALSE POSITIVES}}$$

Precision is a number between 0 and 1

# RECALL =

| | predicted POSITIVE | predicted NEGATIVE |
|---|---|---|
| actual POSITIVE | TRUE POSITIVE | FALSE NEGATIVE |
| actual NEGATIVE | FALSE POSITIVE | TRUE NEGATIVE |

$$\frac{\text{TRUE POSITIVES}}{\text{TRUE POSITIVES + FALSE NEGATIVES}}$$

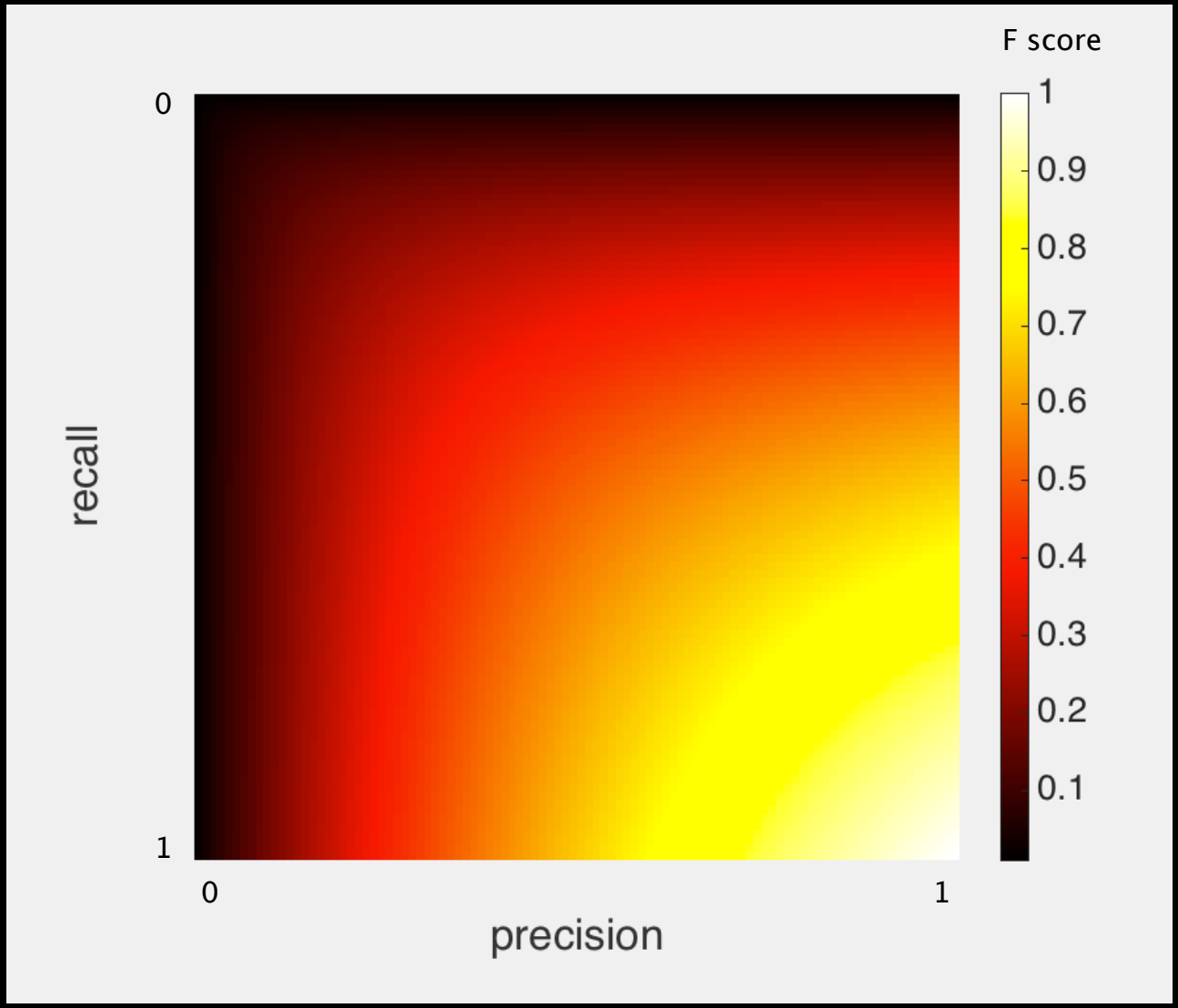
Recall is a number between 0 and 1

# F1 score (or F score)

$$2 \frac{\text{PRECISION} \times \text{RECALL}}{\text{PRECISION} + \text{RECALL}}$$

F1 score is a number between 0 and 1

# F score



# PCA

- PCA operates on the features
- Each feature is an axis
- Two features: XY coordinate system
- PCA rotates the XY coordinate system

# PCA example

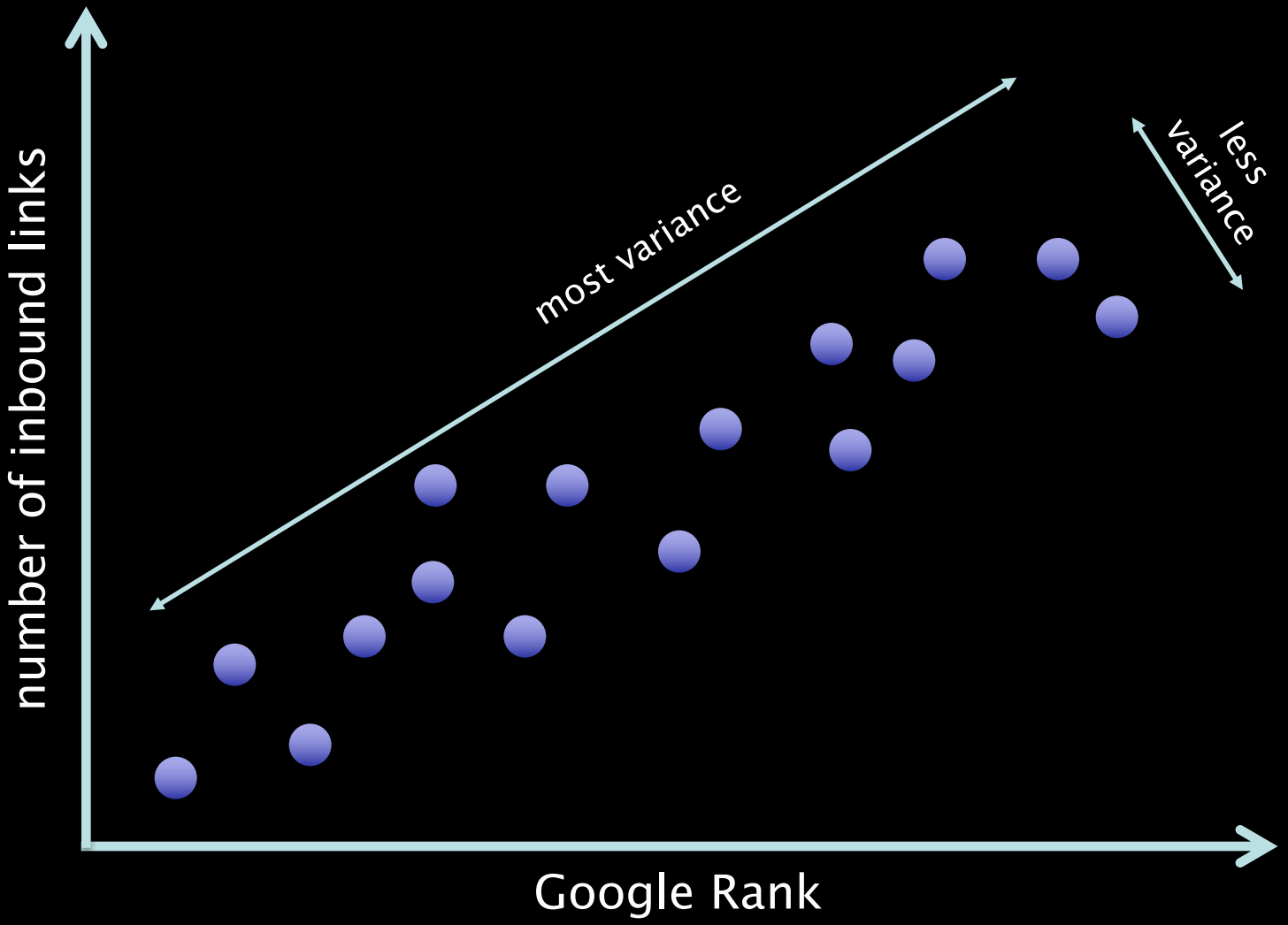
- Two features describing a website

1. Number of inbound links
2. Google Rank

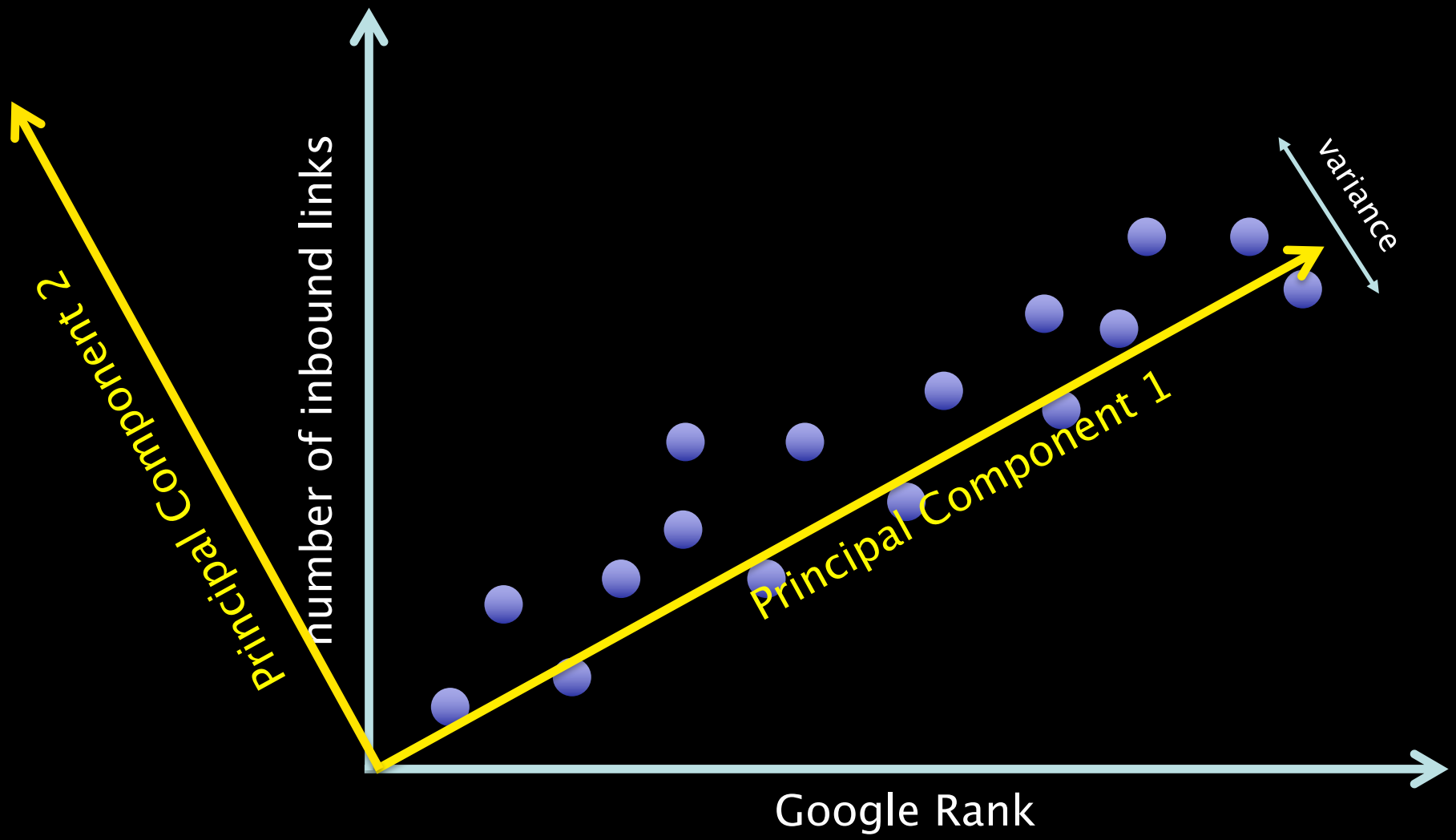


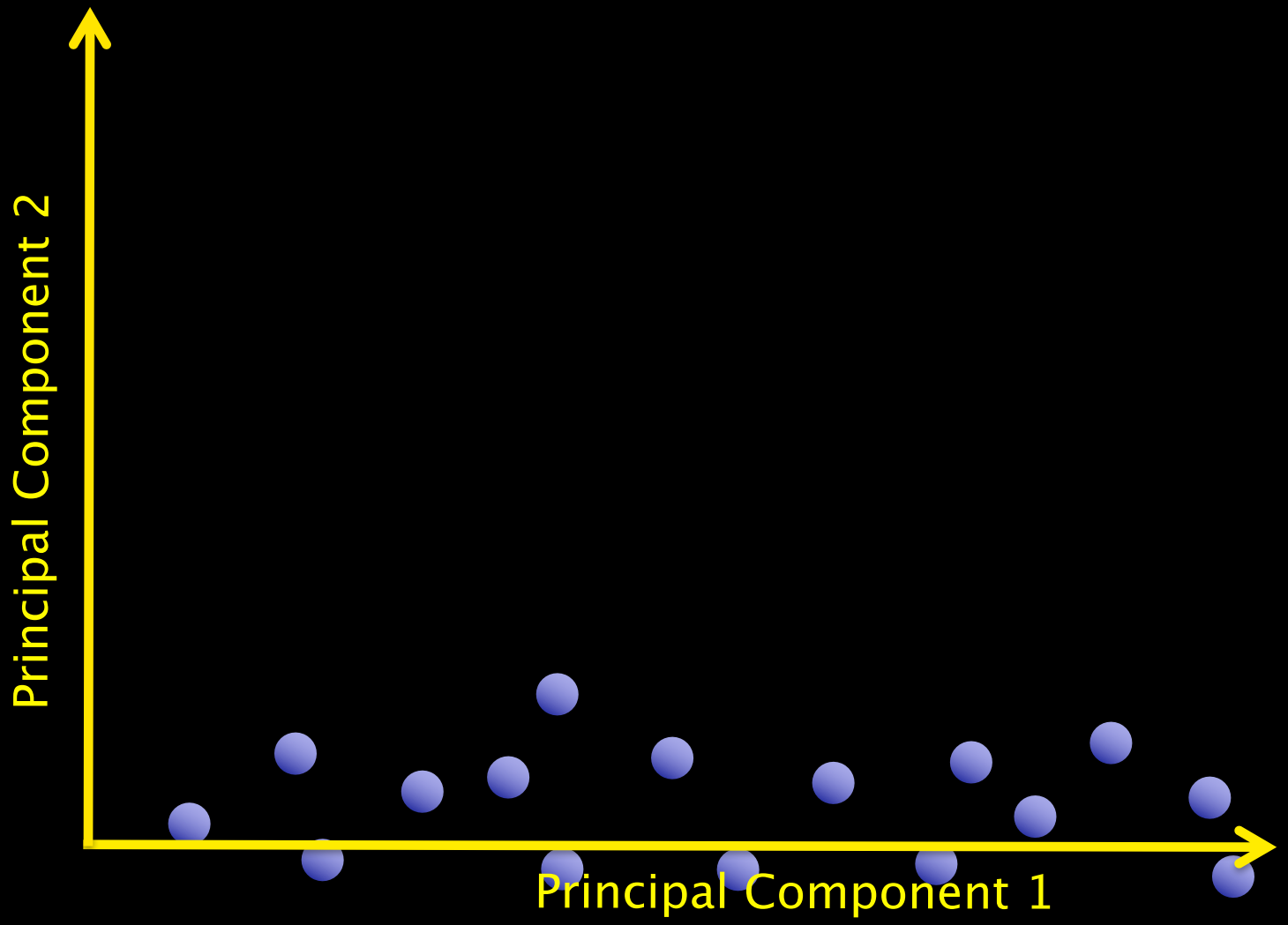
- These features are correlated and therefore **redundant**

# Correlated Features



# PCA


number of inbound links
Google Rank
Principal Component 1
Principal Component 2
variance


Principal Component 2
Principal Component 1

# PCA

- Takes 2 features as input

1. Number of inbound links
2. Google Rank

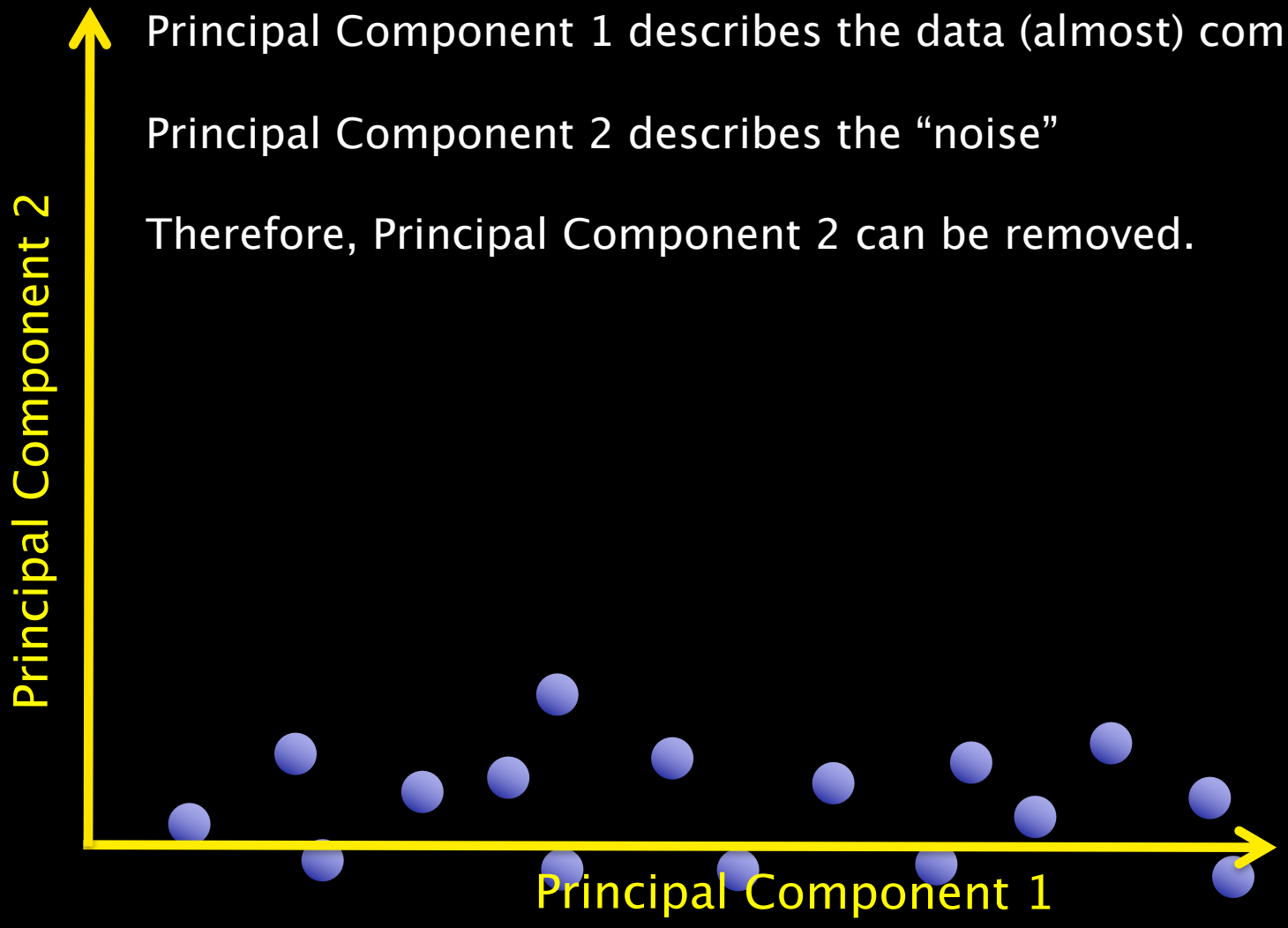

- … and gives 2 new features as output

1. Principal Component 1
2. Principal Component 2

Principal Component 1 describes the data (almost) completely

Principal Component 2 describes the “noise”

Therefore, Principal Component 2 can be removed.



# PCA
## (highest component removed)

- Takes two features as input

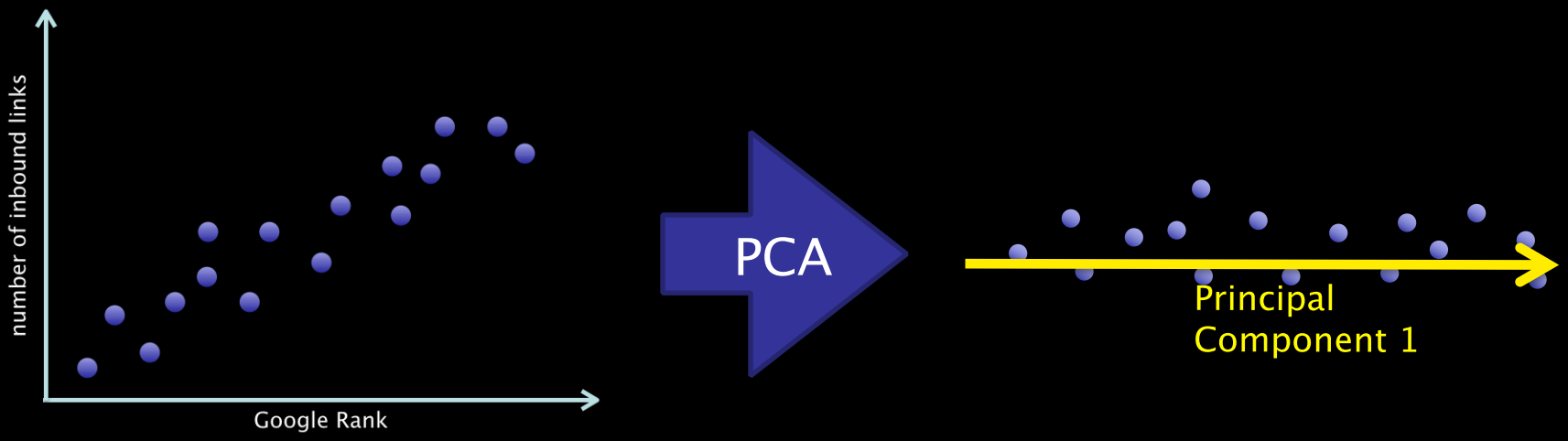
1. Number of inbound links
2. Google Rank


- … and gives **one** feature as output

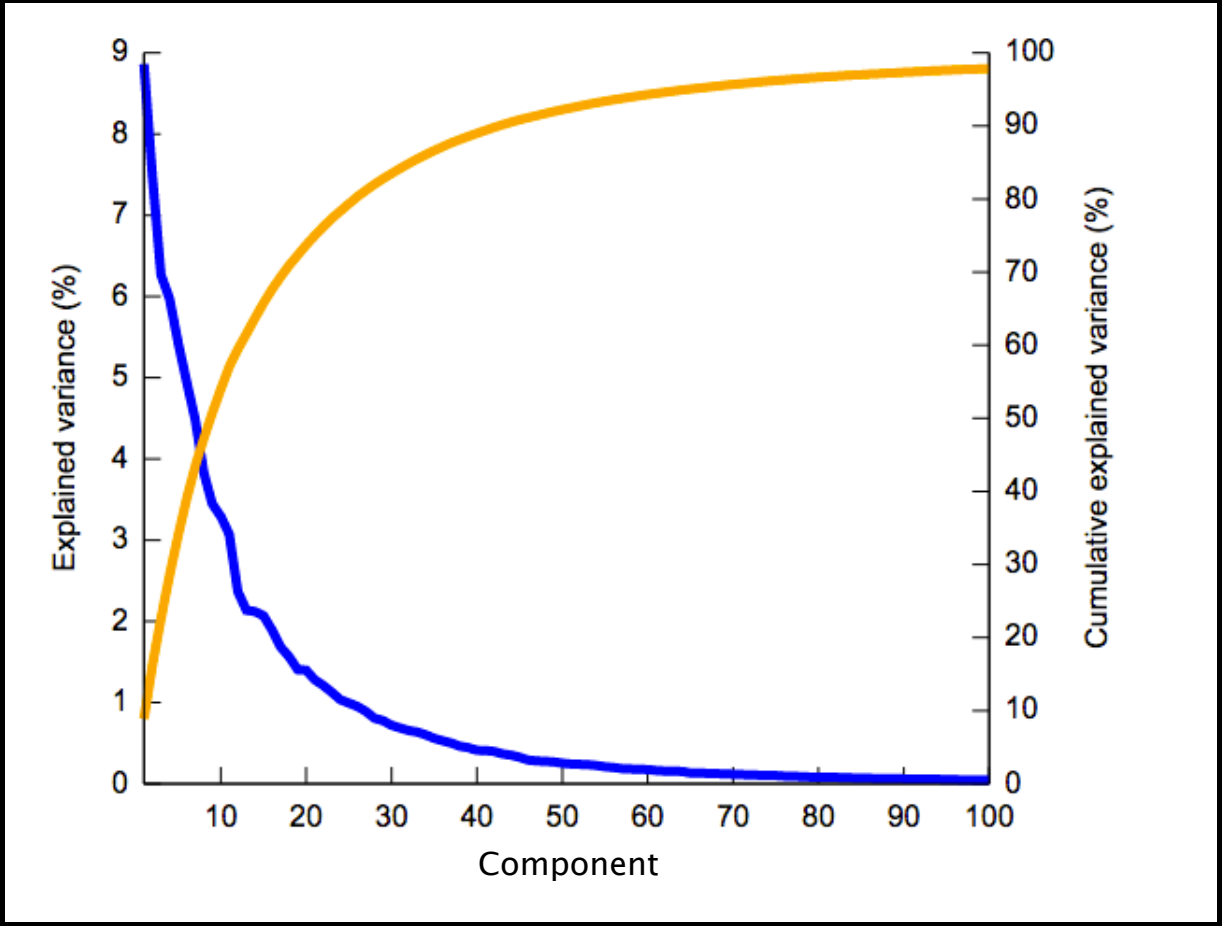
1. Principal Component 1

# Dimensionality Reduction

- PCA performs **dimensionality reduction**
- It reduces 2 dimensions (features) to 1



# PCA in general



# Required Reading

WEKA book:

Chapter 5, Section 5.7 up to pp. 177
(emphasis is on precision and recall)

Chapter 7, Section 7.3, pp. 324-326 (PCA)

Chapter 13, The Experimenter

(just read to understand how it works)