Today

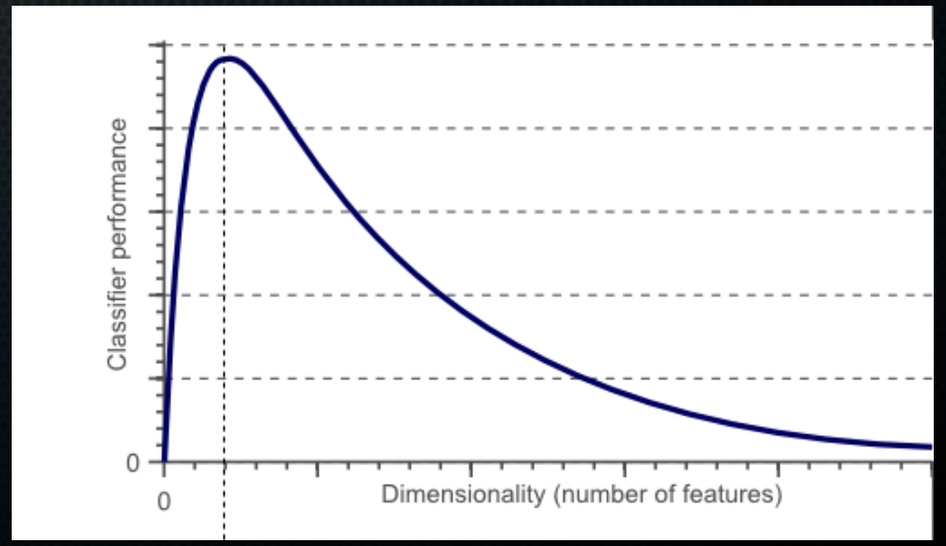
- Recap yesterday
- Problems with high-dimensional data
- Feature extraction
- Dimension reduction
  - Principal Coordinate Analysis (PCA)

Yesterday

- Clustering attempts to find structure in non-labeled data
- Different clustering approaches:
  - K-means clustering: form K clusters from the data using an iterative procedure. Prototype method.
  - Hierarchical clustering: start with every observation its own cluster, combine until everything is in one cluster → builds a tree of observations. Cutting the tree at any level → clusters
  - Some intricacies of hierarchical tree building in phylogeny
- There is no true clustering, only a dialogue with the data and what is best for your purposes.

Problems with high-dimensional data: curse of dimensionality

- A naive idea would be: if I cannot classify well with 10 features, then surely with 100 or 1000 or 10.000 features (or expression of 20.000 genes?) I can!
- However: wonky things happen in high dimensions that our brains are not well-equipped to handle. The counterintuitive result is that the above is completely untrue!



Source: https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/

Problems with high-dimensional data: curse of dimensionality

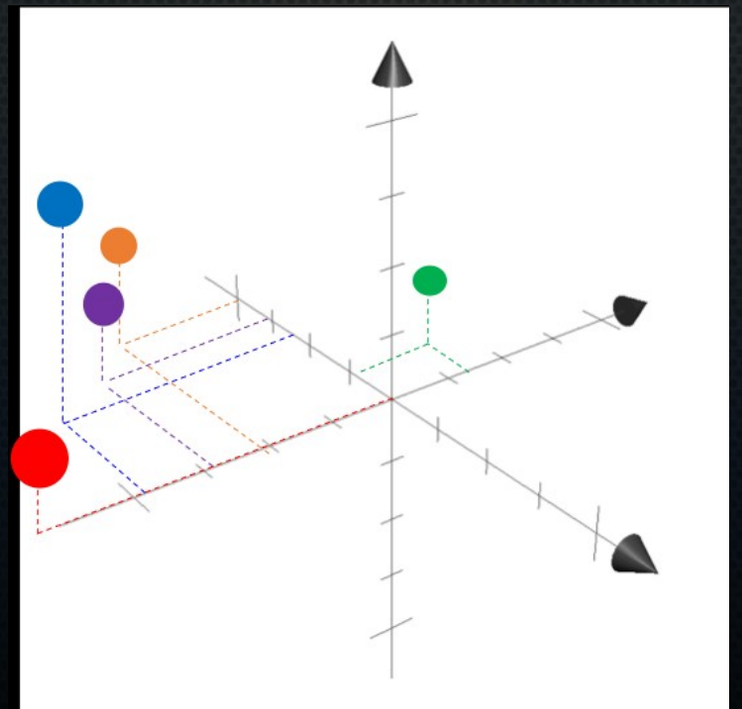
- What is going on?
- There's four main issues:
  - For higher dimensions, you need exponentially more data to have good coverage of the possibility space (data sparsity rises exponentially with dimension of data)
  - In higher dimensions, you can easily make models with 100% accuracy Unfortunately, they will still be garbage → Automatic overfitting.
  - In higher dimensions, every data point is an extreme point
  - In higher dimensions, distances between points become meaningless

Problems with high-dimensional data: curse of dimensionality

- Before we examine each of these, we need to define dimensionality, and extrinsic and intrinsic dimensionality.

Problems with high-dimensional data: curse of dimensionality

- Before we examine each of these, we need to define dimensionality, and extrinsic and intrinsic dimensionality.
- Dimensionality = how many pieces of information you use to describe a point. In 3D space, you use x, y, and z to define a specific point.



Source: https://mbernste.github.io/posts/intrinsic_dimensionality/

Problems with high-dimensional data: curse of dimensionality

- Before we examine each of these, we need to define dimensionality, and extrinsic and intrinsic dimensionality.
- Dimensionality = how many pieces of information you use to describe a point. In 3D space, you use x, y, and z to define a specific point.
- Let’s look at some expression data examples to think about intrinsic and extrinsic dimensionality.

Problems with high-dimensional data: curse of dimensionality

- Intrinsic and extrinsic dimensionality

	Gene A expression	Gene B expression
Sample 1	0.22	-3
Sample 2	8	4
Sample 3	-0.86	1.2

2D data

Problems with high-dimensional data: curse of dimensionality

- Intrinsic and extrinsic dimensionality

	Gene A expression	Gene B expression	Gene A expression2	Gene B expression2	Gene A expression3	Gene B expression3
Sample 1	0.22	-3	0.22	-3	0.22	-3
Sample 2	8	4	8	4	8	4
Sample 3	-0.86	1.2	-0.86	1.2	-0.86	1.2

6D data?

Problems with high-dimensional data: curse of dimensionality

- Intrinsic and extrinsic dimensionality

	Gene A expression	Gene B expression	Gene C expression
Sample 1	0.22	-3	4.82
Sample 2	8	4	-2.5
Sample 3	-0.86	1.2	1.1

3D data?

Problems with high-dimensional data: curse of dimensionality

- Intrinsic and extrinsic dimensionality

	Gene A expression	Gene B expression	Gene A expression2	Gene B expression2	Gene A expression3	Gene B expression3
Sample 1	0.22	-3	0.22	-3	0.22	-3
Sample 2	8	4	8	4	8	4
Sample 3	-0.86	1.2	-0.86	1.2	-0.86	1.2

- Even when we now have 6 variables describing our data, we wouldn't say it is really 6-dimensional because 4 columns are outright copies of the first 2.

	Gene A expression	Gene B expression	Gene C expression
Sample 1	0.22	-3	4.82
Sample 2	8	4	-2.5
Sample 3	-0.86	1.2	1.1

- Adding a gene C would make it 3-dimensional. → to describe a sample in the space of these 3 expression values you cannot miss expression of gene C!

Problems with high-dimensional data: curse of dimensionality

- Intrinsic and extrinsic dimensionality

	Gene A expression	Gene B expression	Gene A expression2	Gene B expression2	Gene A expression3	Gene B expression3
Sample 1	0.22	-3	0.22	-3	0.22	-3
Sample 2	8	4	8	4	8	4
Sample 3	-0.86	1.2	-0.86	1.2	-0.86	1.2

Extrinsic dimensionality: 6
Intrinsic dimensionality : 2

- Even when we now have 6 variables describing our data, we wouldn't say it is really 6-dimensional because 4 columns are outright copies of the first 2.

	Gene A expression	Gene B expression	Gene C expression
Sample 1	0.22	-3	4.82
Sample 2	8	4	-2.5
Sample 3	-0.86	1.2	1.1

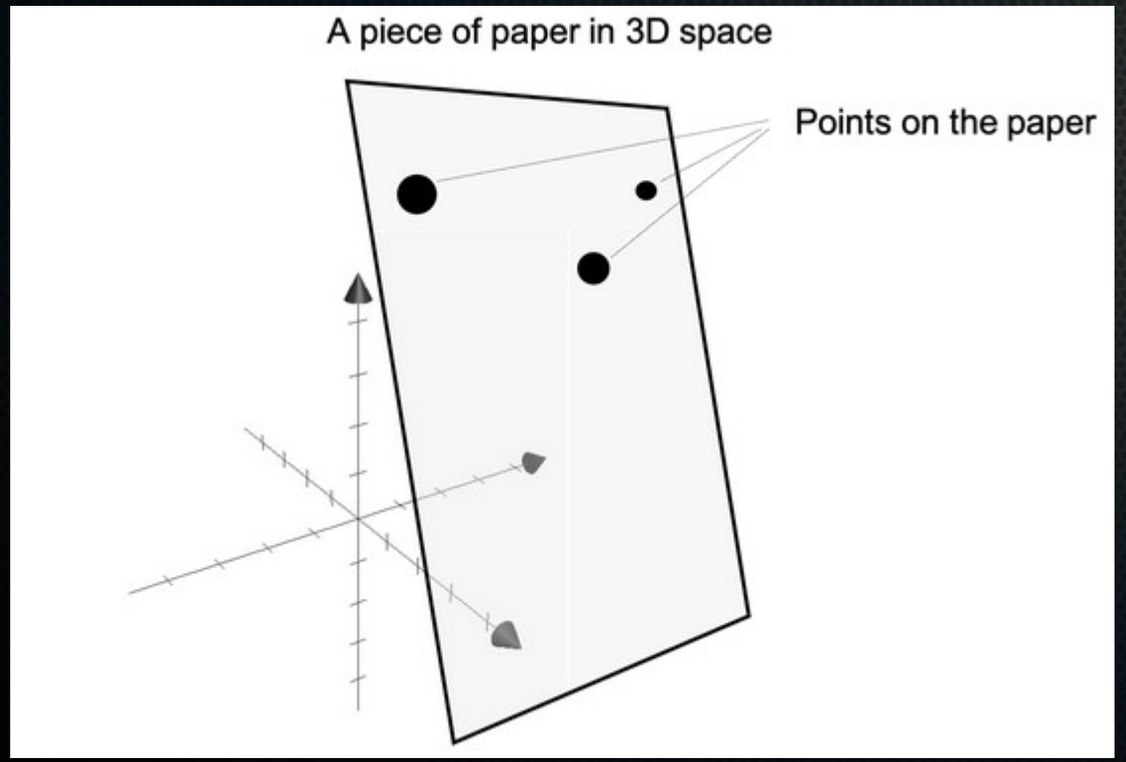
Extrinsic dimensionality: 3
Intrinsic dimensionality : 3

- Adding a gene C would make it 3-dimensional. → to describe a sample in the space of these 3 expression values you cannot miss expression of gene C!

Problems with high-dimensional data: curse of dimensionality

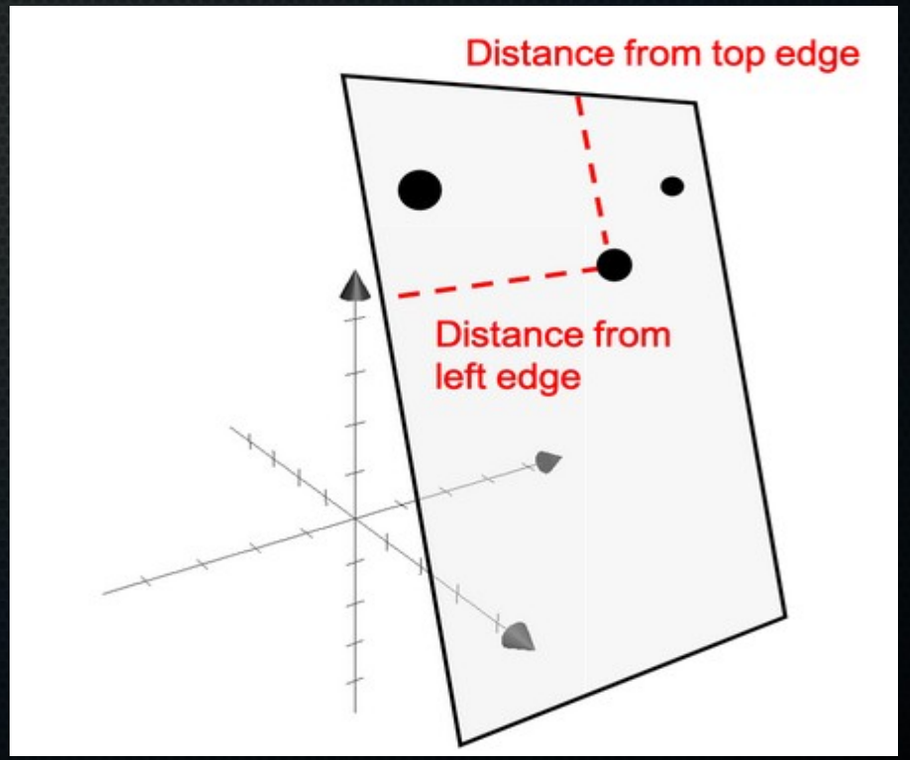
- A visual example:

Extrinsic dimensionality: 3



Source: https://mbernste.github.io/posts/intrinsic_dimensionality/

Intrinsic dimensionality : 2

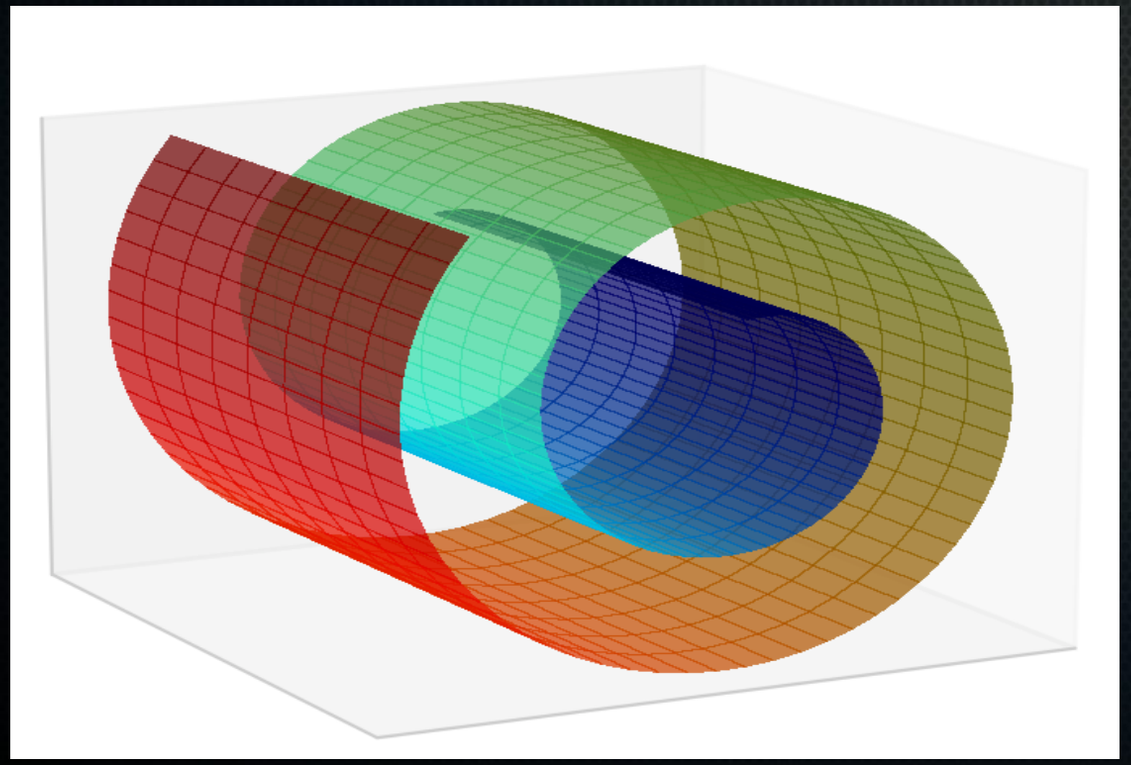


Source: https://mbernste.github.io/posts/intrinsic_dimensionality/

Problems with high-dimensional data: curse of dimensionality

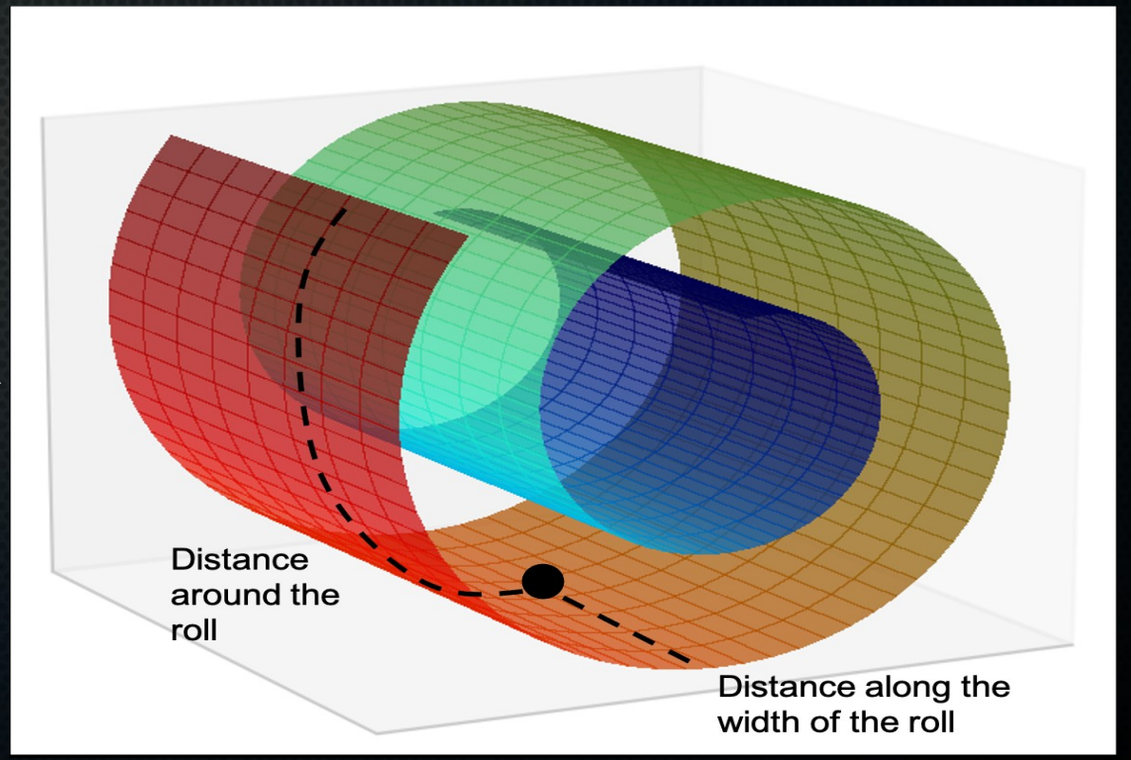
- A visual example:

Extrinsic dimensionality: 3



Source: https://mbernste.github.io/posts/intrinsic_dimensionality/

Intrinsic dimensionality : 2



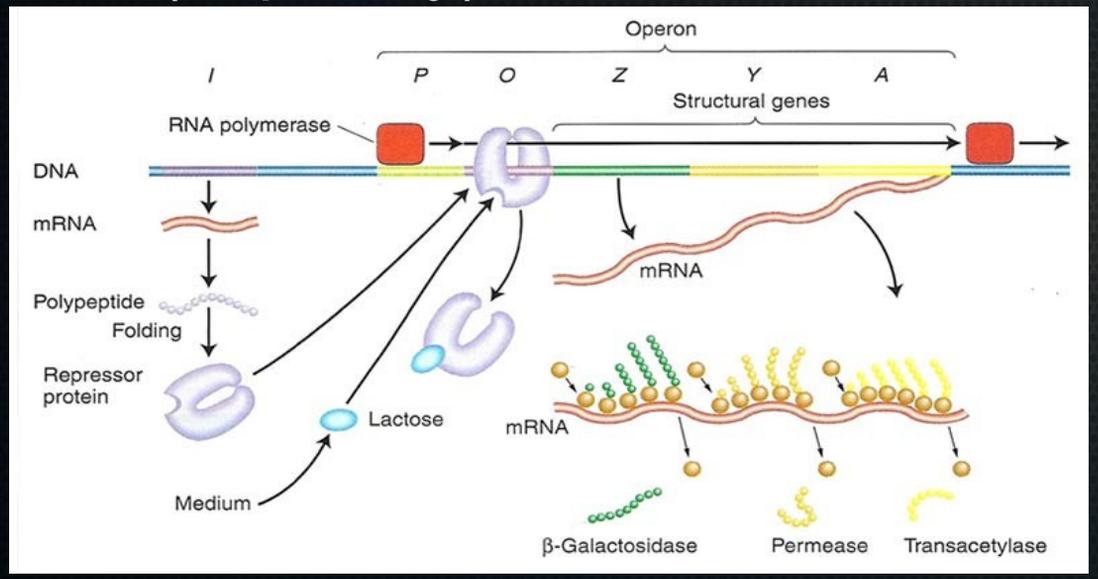
Source: https://mbernste.github.io/posts/intrinsic_dimensionality/

Problems with high-dimensional data: curse of dimensionality

- What about real data?
- Will (hopefully) not find features outright repeated.

Problems with high-dimensional data: curse of dimensionality

- What about real data?
- Will (hopefully) not find features outright repeated.



Source: https://microbenotes.com/wp-content/uploads/2018/09/Lac-Operon.jpg

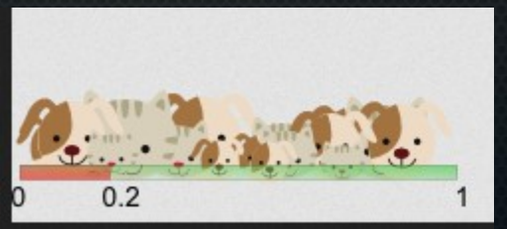
- If you are making a metabolic classifier (Lac-metabolising or not) → no extra information Permease and Transacetylase. However: they will have different (but correlated) expression values (e.g. due to different mRNA properties)

Problems with high-dimensional data: curse of dimensionality

- What about real data?
- Will (hopefully) not find features outright repeated.
- Alternative in humans: if an oncogenic mutation in the promoter of a transcription factor causes it to become highly expressed, both it *and* many of its downstream targets will be different between patients with or without cancer. → don't want to include all of these genes for the classification. One will suffice!
- In other words: if features give *mostly* the same information, you get dimensionality problems without benefits. → search collinearity/multicollinearity for extra reading. And see here or here.

Problems with high-dimensional data: data sparsity

- We want to classify images of cats and dogs. We are naive so we think this might work using average red, green, and blue intensity values of the images.
- Let's say there's 100 cats and dogs on the planet. We really did our best and got 20 images to train on, but for some reason we only have the red channel to train on right now.



Source: https://www.visiondummy.com/wp-content/uploads/2014/04/curseofdimensionality.png

Problems with high-dimensional data: data sparsity

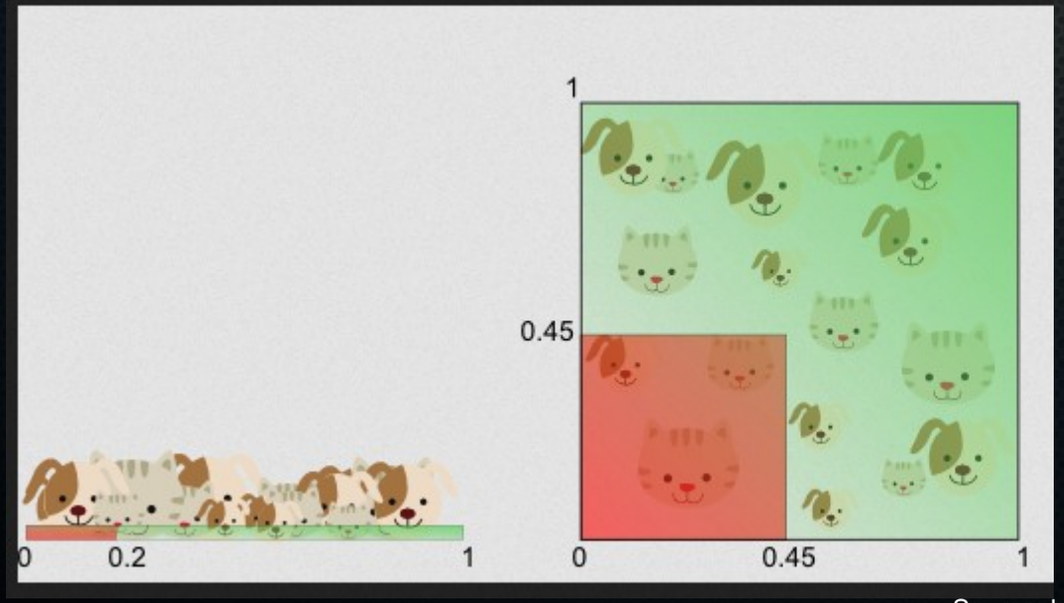
- We want to classify images of cats and dogs. We are naive so we think this might work using average red, green, and blue intensity values of the images.
- Let's say there's 100 cats and dogs on the planet. We really did our best and got 20 images to train on, but for some reason we only have the red channel to train on right now.

Source: https://www.visiondummy.com/wp-content/uploads/2014/04/curseofdimensionality.png

We cover 20% of all possible average red values

Problems with high-dimensional data: data sparsity

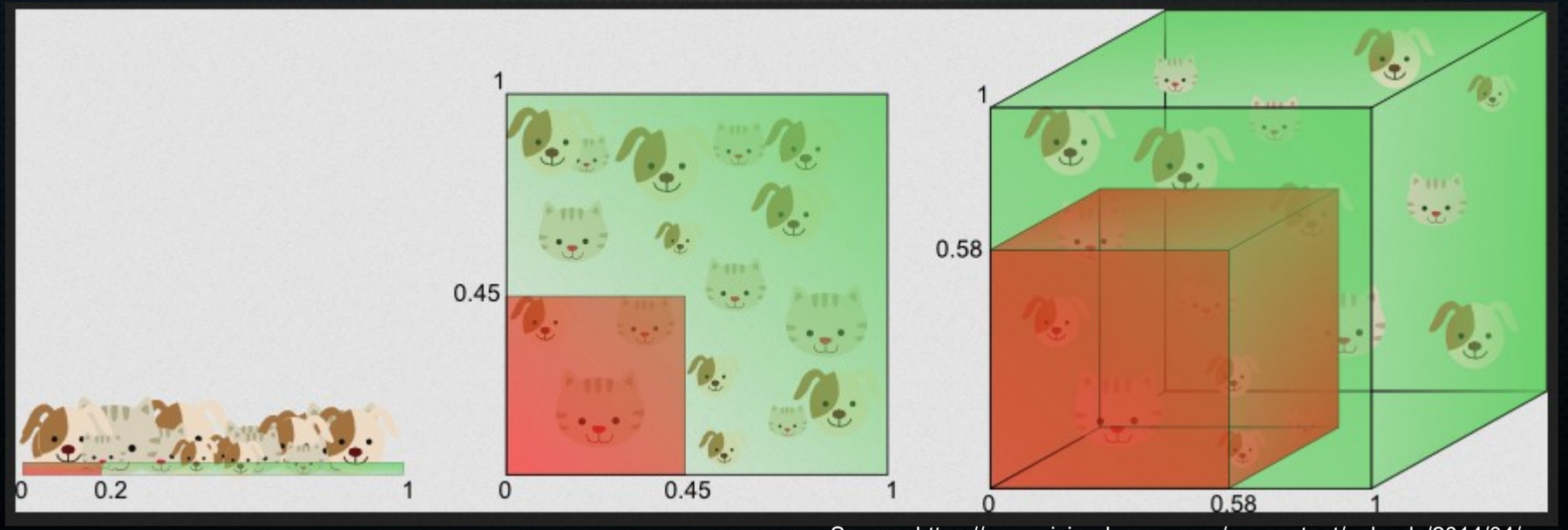
- Now we get our data for the green channel. How many cats and dogs do we need to sample to still cover 20% of the possibilities?
- → You need to sample 45% of the possible red values and 45% of the possible green values, since 0.45^2 ~= 0.2.



Source: https://www.visiondummy.com/wp-content/uploads/2014/04/curseofdimensionality.png

Problems with high-dimensional data: data sparsity

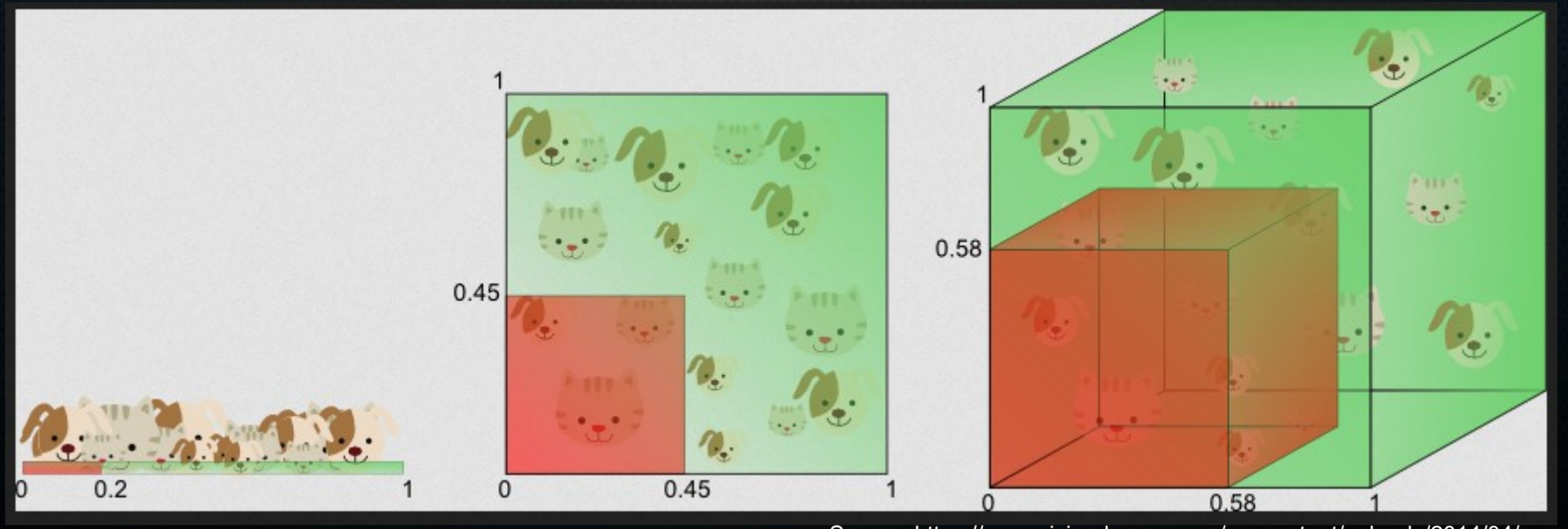
- Now we get our data for the blue channel as well. How many cats and dogs do we need to sample to still cover 20% of the possibilities?
- → You need to sample 58% of the possible red, green and blue values, since 0.58^3 ~= 0.2.



Source: https://www.visiondummy.com/wp-content/uploads/2014/04/curseofdimensionality.png

Problems with high-dimensional data: data sparsity

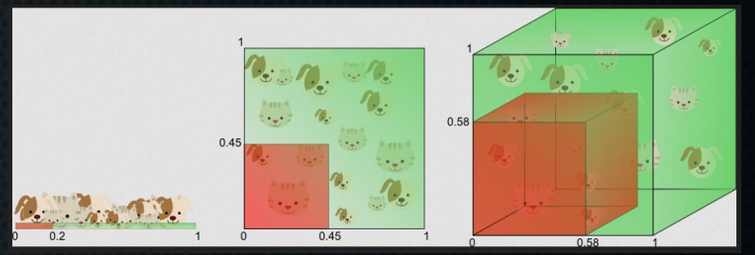
- Exponential increase in data needed to get the same coverage of the space. In other words: given some training set size, data quickly becomes extremely sparse the more features you use!



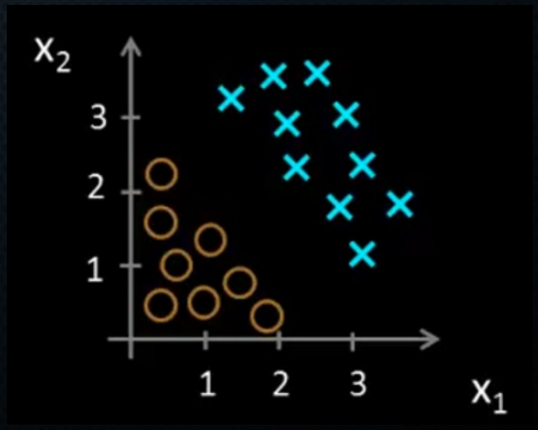
Source: https://www.visiondummy.com/wp-content/uploads/2014/04/curseofdimensionality.png

Problems with high-dimensional data: data sparsity

- Exponential increase in data needed to get the same coverage of the space. In other words: given some training set size, data quickly becomes extremely sparse the more features you use!
- Bad news for ML:
  - ML is statistical: we count correct and incorrect classifications. If the observations become extremely sparse, then for much of the problem space we have no idea.

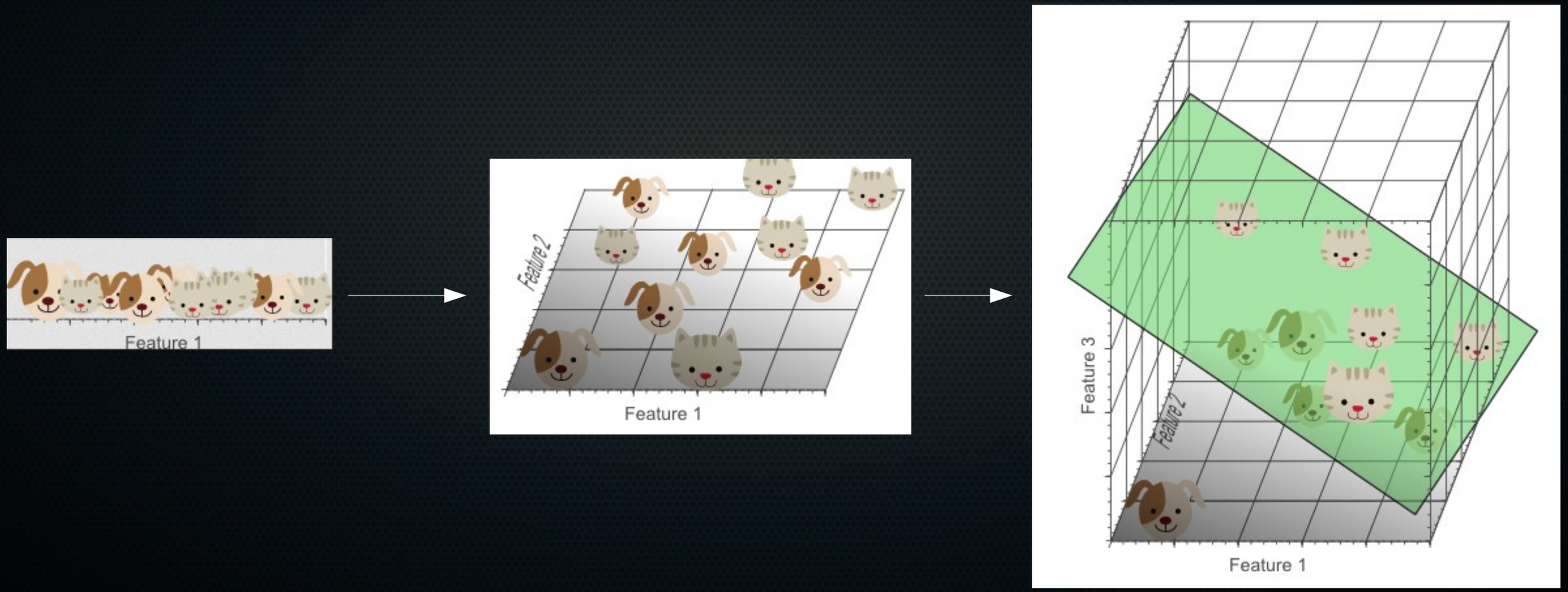


Source: https://www.visiondummy.com/wp-content/uploads/2014/04/curseofdimensionality.png



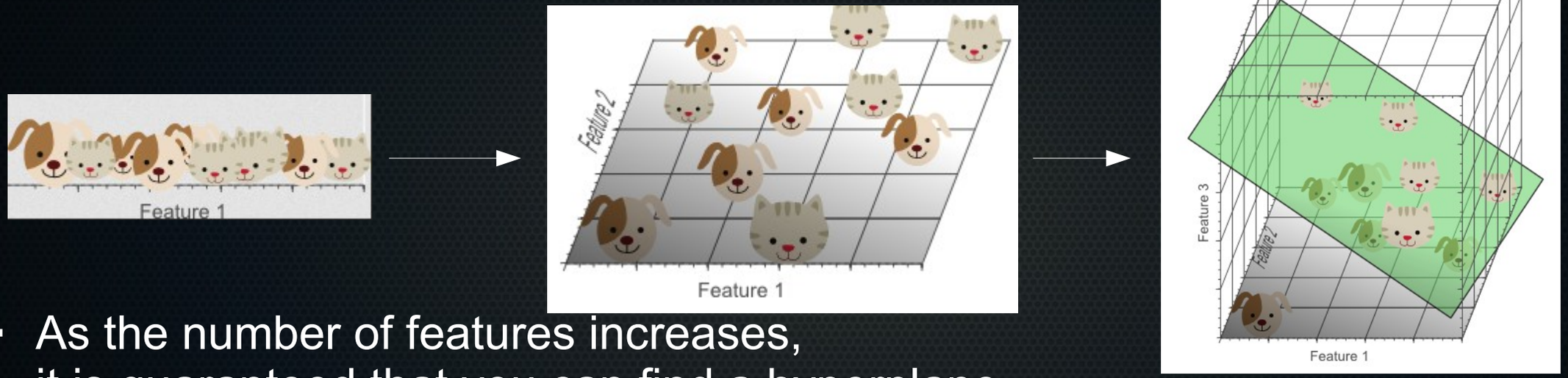
Problems with high-dimensional data: overfitting

- Directly related is the problem of overfitting.



Problems with high-dimensional data: overfitting

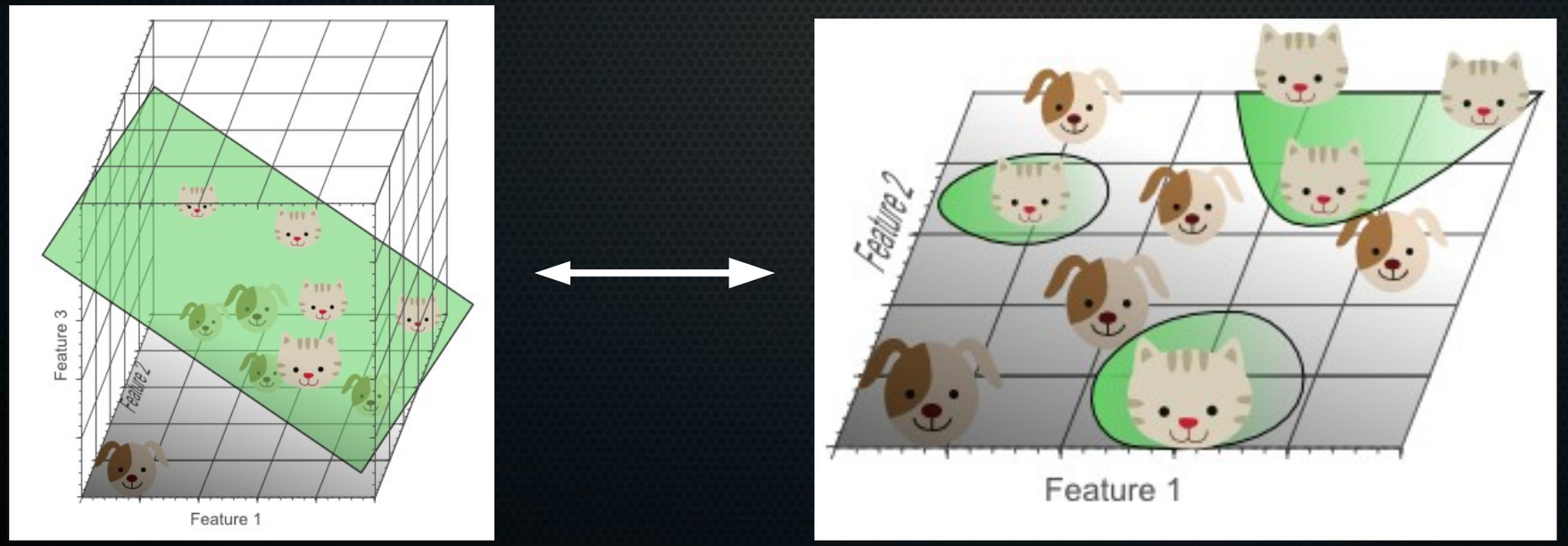
- Directly related is the problem of overfitting.



- As the number of features increases, it is guaranteed that you can find a hyperplane that separates the data perfectly.
- However...

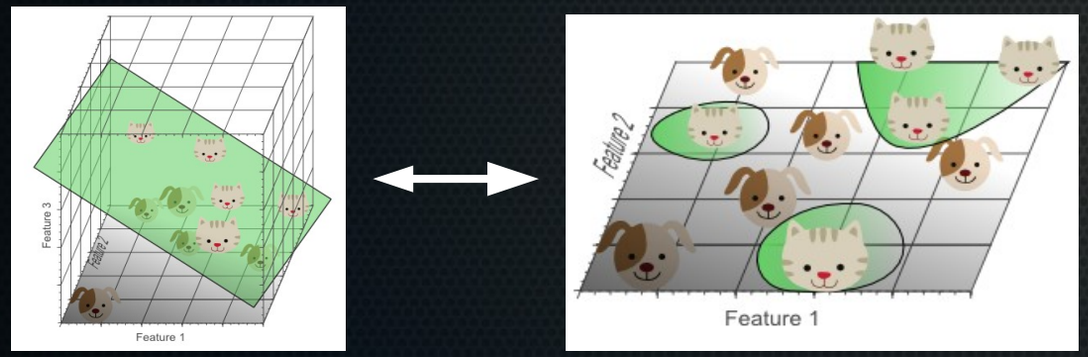
Problems with high-dimensional data: overfitting

- If you map it back to fewer dimensions: clearly see that this is equal to overfitting → far too complex model that has adapted completely to the training data (so to noise).

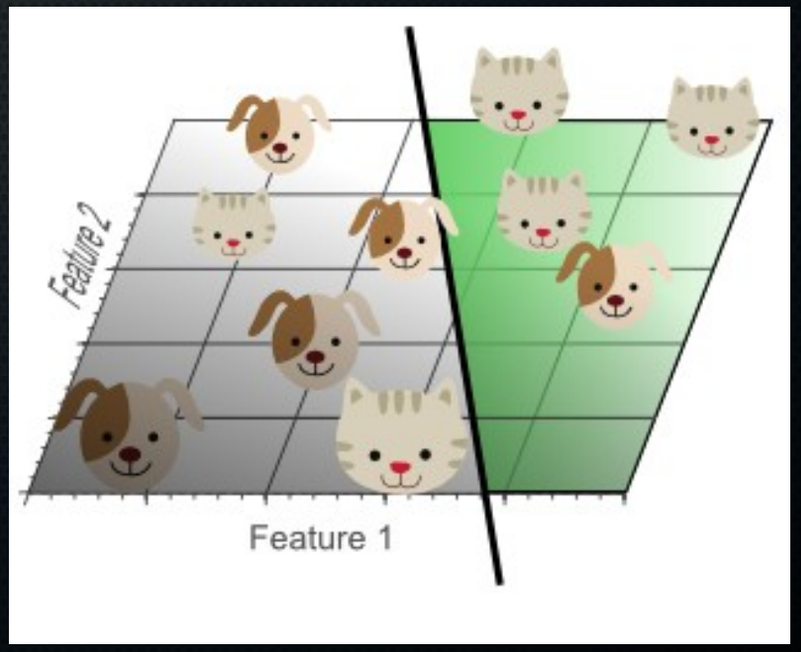


Problems with high-dimensional data: overfitting

- If you map it back to fewer dimensions: clearly see that this is equal to overfitting → far too complex model that has adapted completely to the training data (so to noise).

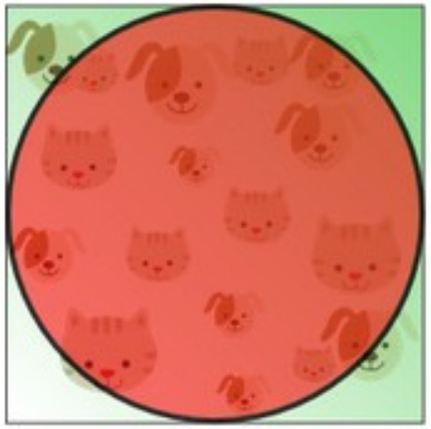


- What we'd want is something like this:
  → Not perfect, but generalises much better



Problems with high-dimensional data: Extreme points

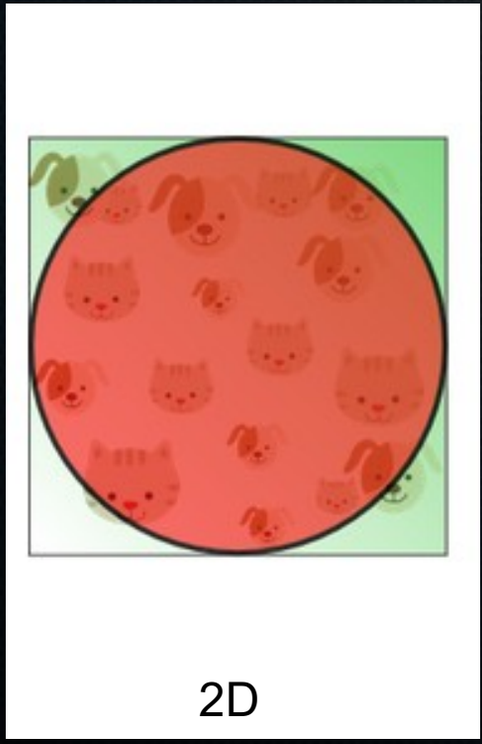
- As the number of dimensions increases, all data points become extremes/outliers → none are meaningfully ‚close‘ to another.



2D

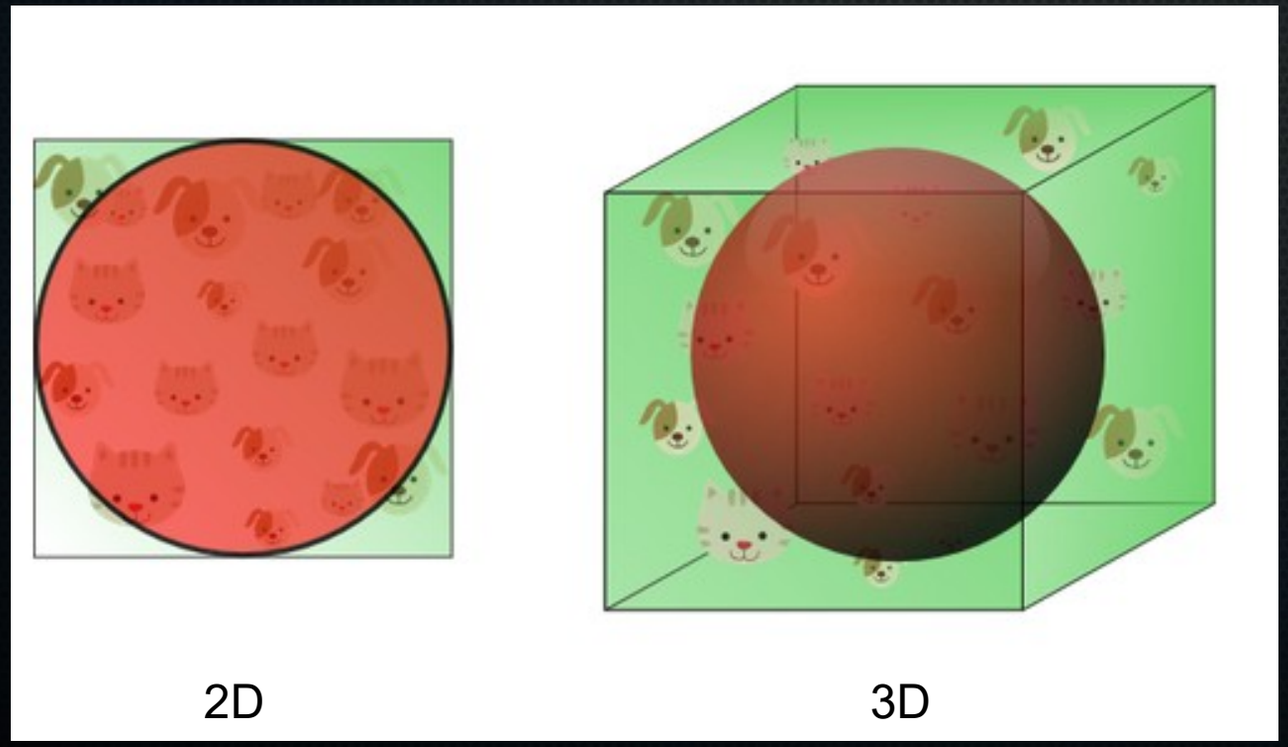
Problems with high-dimensional data: Extreme points

- Here: unit square (1*1) with inscribed unit circle (radius of 0.5).
- See that to be extreme you have to be very close to 0 or 1 in feature 1 or feature 2. Most data points are average (in the circle)



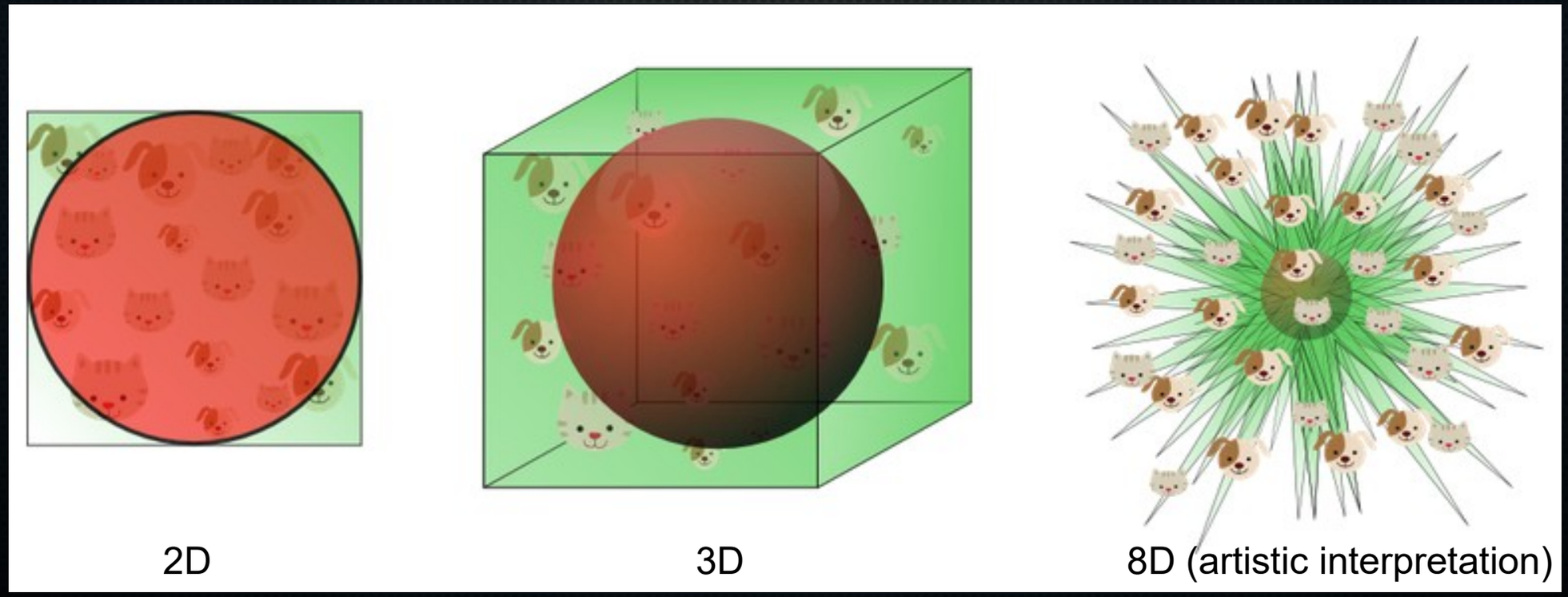
Problems with high-dimensional data: Extreme points

- In 3 dimensions: unit cube remains same volume. Sphere volume *shrinks*! → more points near the edge of the space.



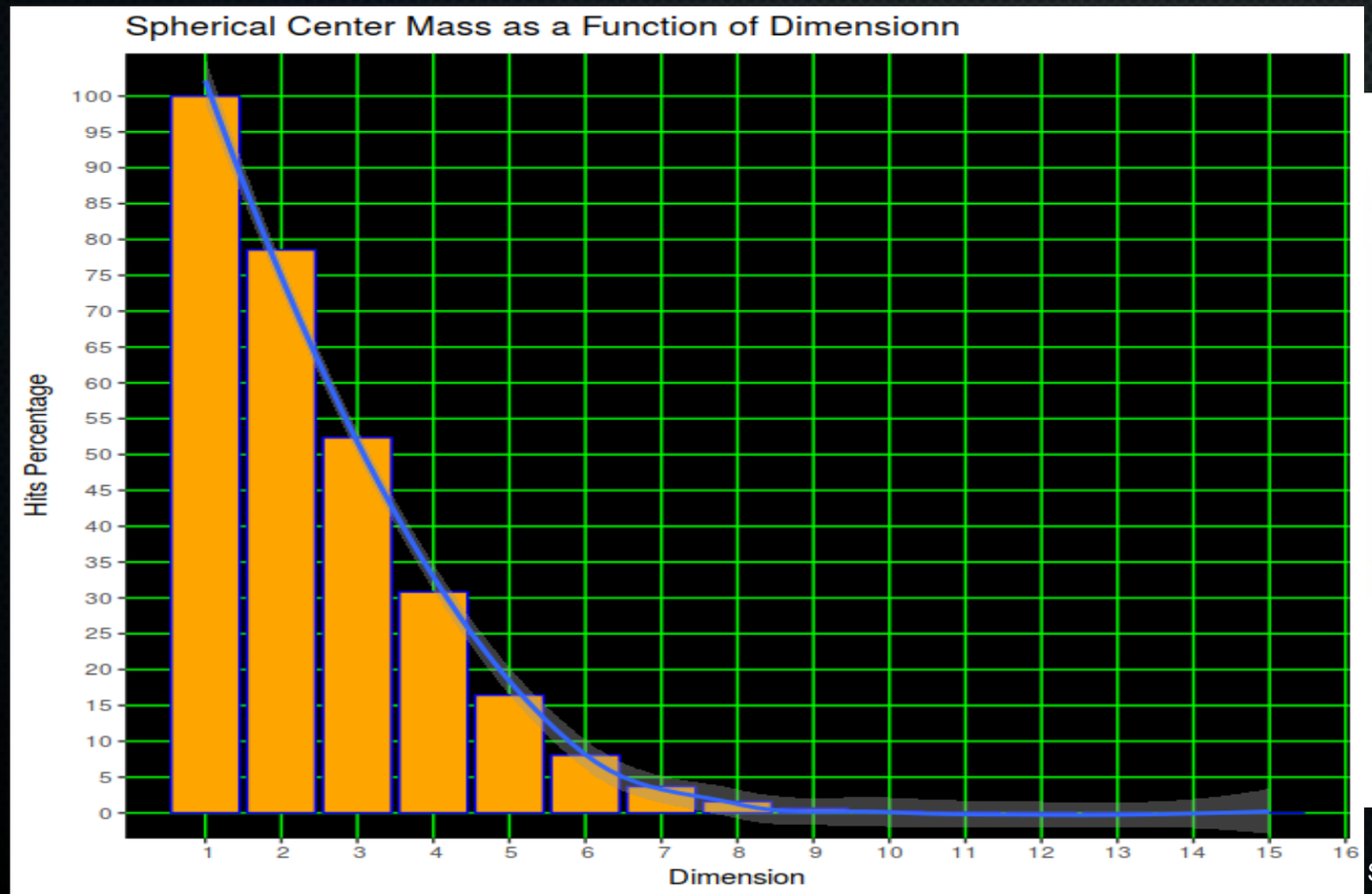
Problems with high-dimensional data: Extreme points

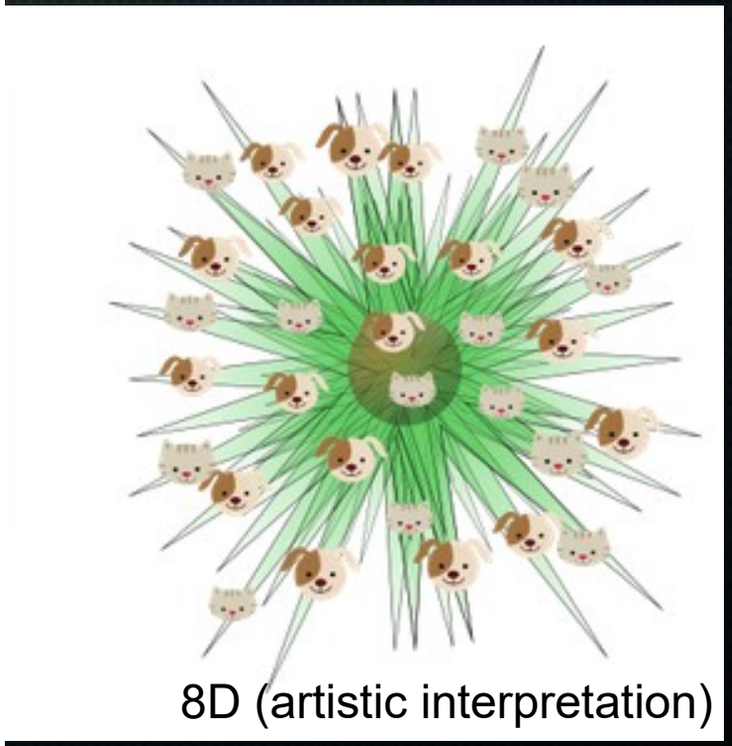
- In 8 dimensions: hypercube remains same volume. Sphere volume *shrinks further*.



Problems with high-dimensional data: Extreme points

- In 8 dimensions: hypercube remains same volume. Sphere volume *shrinks further*. → 98% is outside the middle hypersphere!





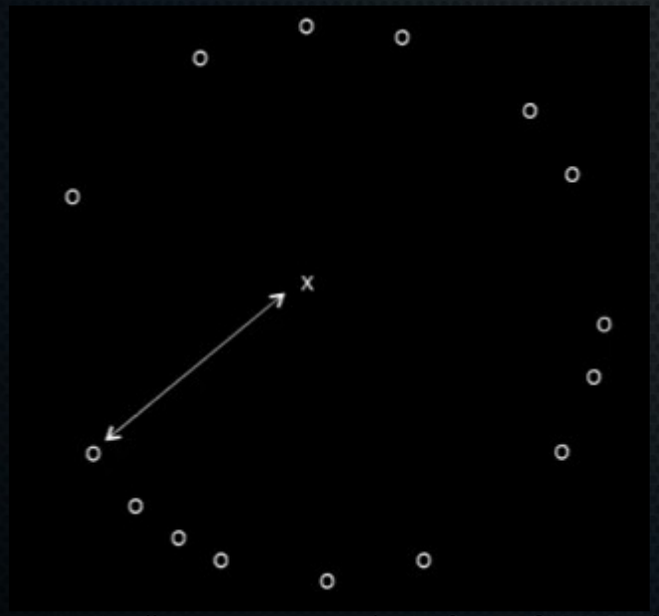
Source: https://blog.midnightmechanism.com/post/dimensionality-curse/

Problems with high-dimensional data: Extreme points

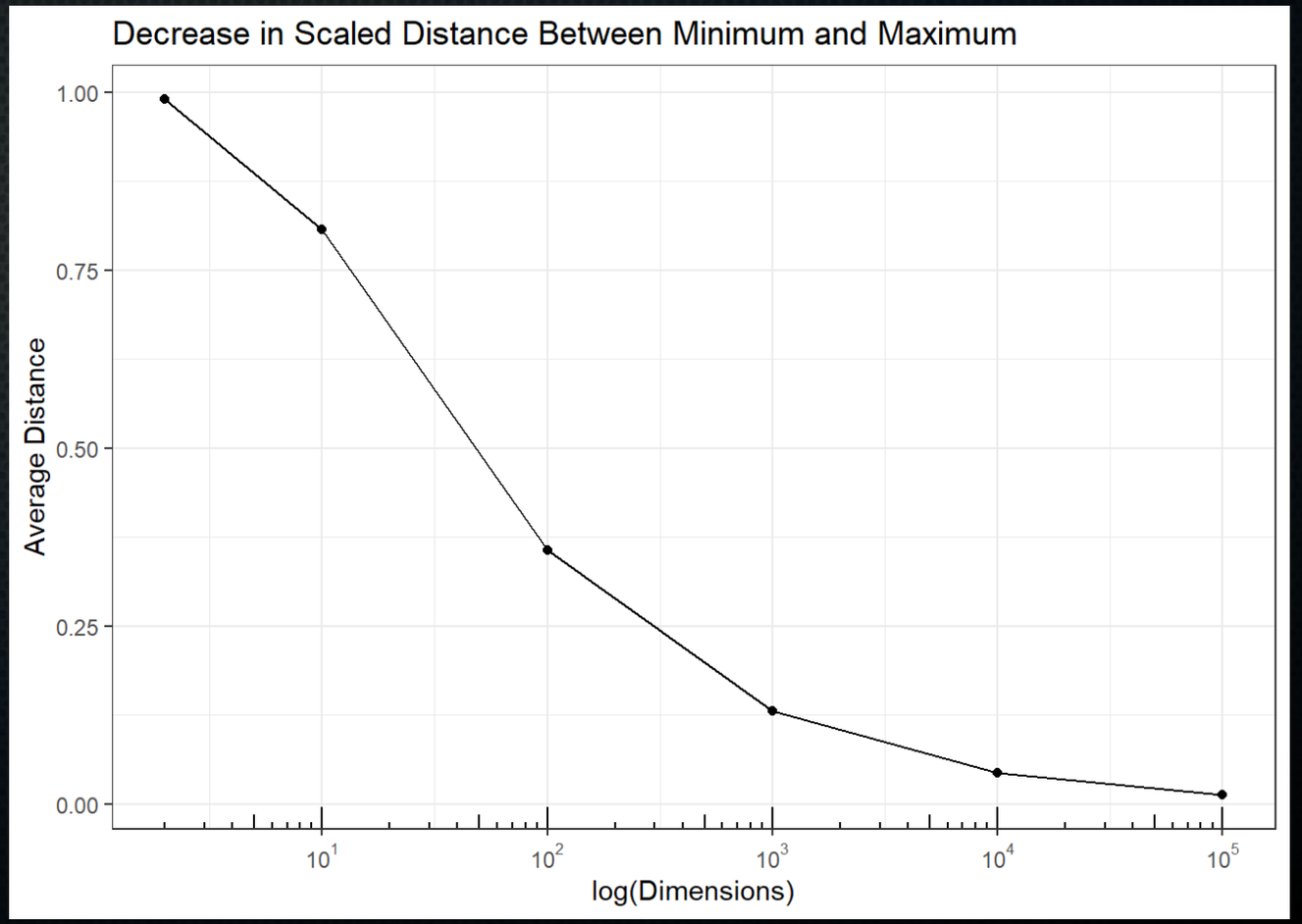
- In 8 dimensions: hypercube remains same volume. Sphere volume *shrinks further*. → 98% is outside the middle hypersphere!
- Intuitively: if just one feature is an outlier, then the whole point becomes an outlier.
  - So with increasing #features → no point is average anymore, they are all extreme in their own way, and unlikely to share similarities.

Problems with high-dimensional data: Extreme points

Strangely enough: because everything is extreme, distances become *meaningless*.



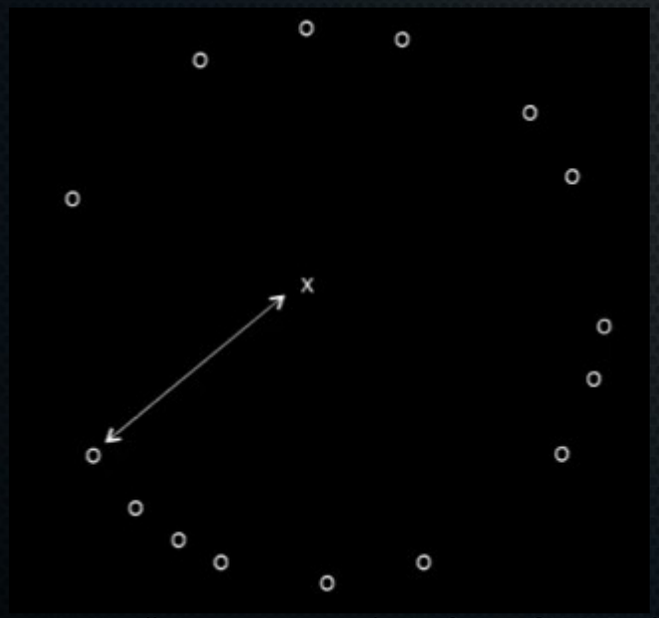
Source: https://www.youtube.com/watch?v=dZrGXYty3qc&list=PL-DP1aeg73q2vMsAgwguwMUQn0JQ3DalV&index=2



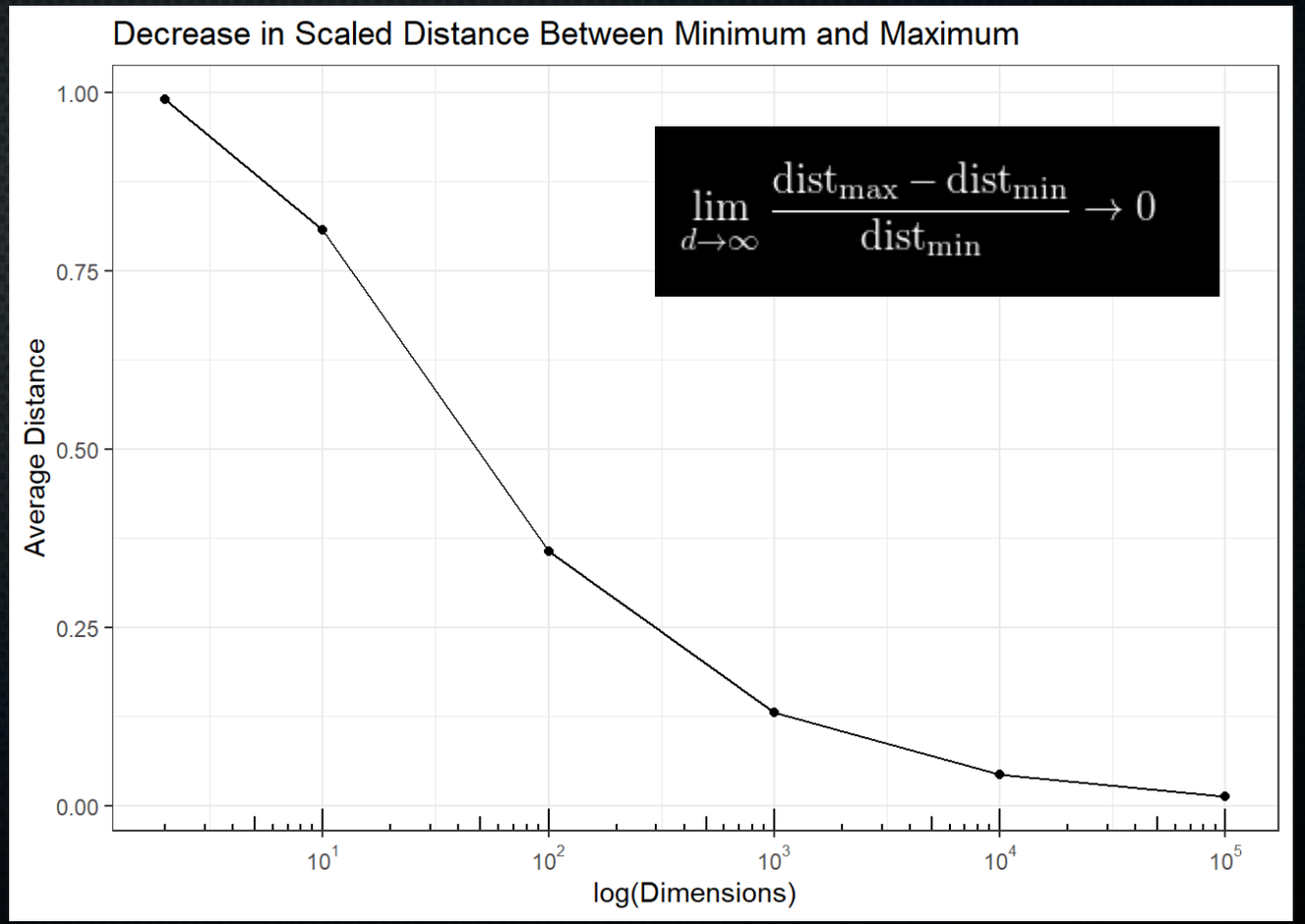
Source: https://blog.dominodatalab.com/the-curse-of-dimensionality/

Problems with high-dimensional data: Extreme points

Strangely enough: because everything is extreme, distances become *meaningless*.



Source: https://www.youtube.com/watch?v=dZrGXYty3qc&list=PL-DP1aeg73q2vMsAgwguwMUQn0JQ3DalV&index=2

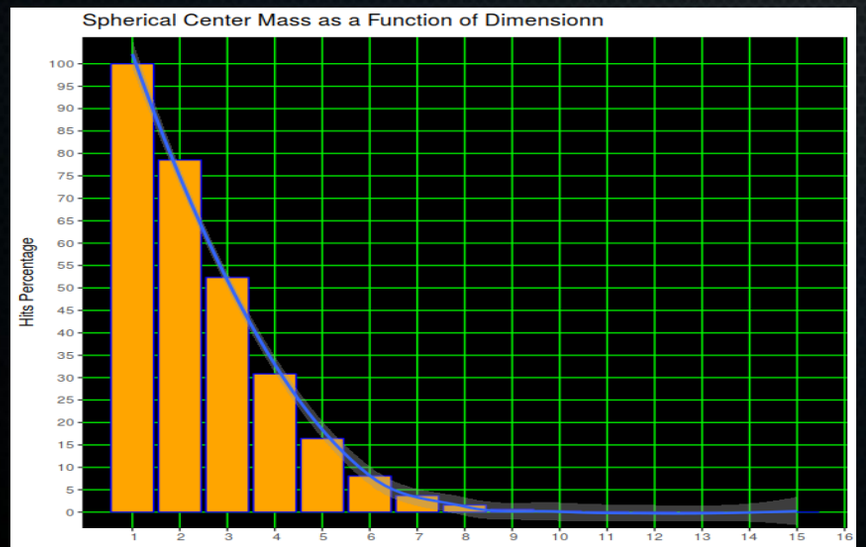


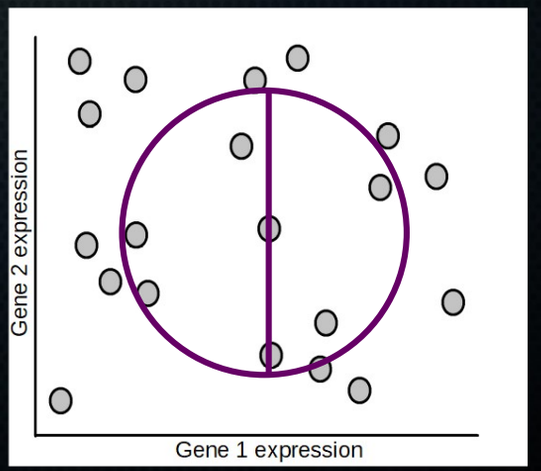
Source: https://blog.dominodatalab.com/the-curse-of-dimensionality/

Problems with high-dimensional data: Extreme points

Strangely enough: because everything is extreme, distances become *meaningless*.

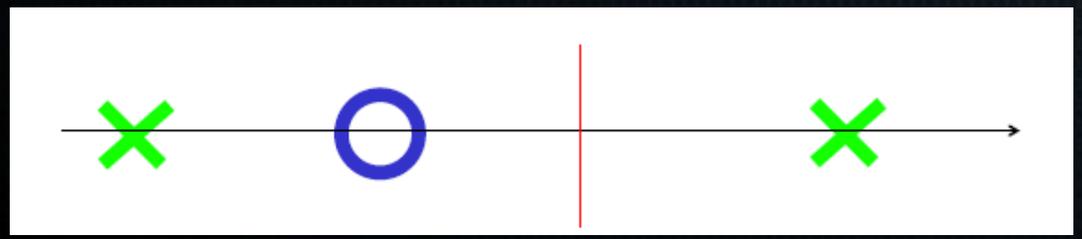
- Intuitively: say you want all neighbours in a certain range. In low-D, that works fine. In high-D, since multidimensional spheres rapidly decrease in volume, no other point is close anymore. All points in high-dimensional space are extremely dissimilar.

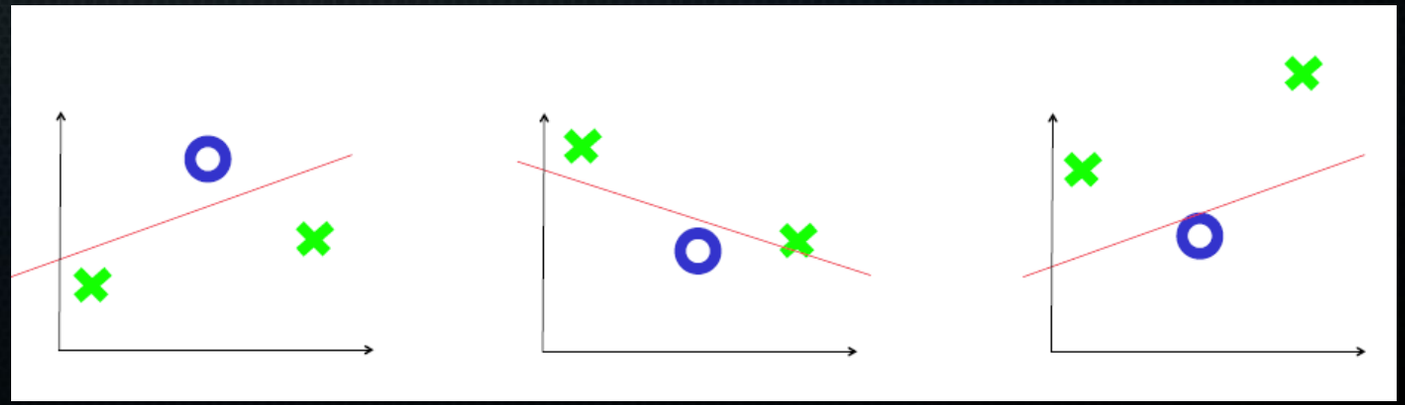




Problems with high-dimensional data: Summary

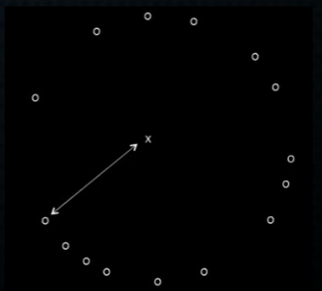
- There's a discrepancy between our low-D intuitions and how high-D data works:
  - More features won't automatically give better performance, because unless you get infeasible amounts of training samples, data scarcity increases exponentially, meaning you can make perfect classifiers, fit to noise.





Problems with high-dimensional data: Summary

- There's a discrepancy between our low-D intuitions and how high-D data works:
  - More features won't automatically give better performance, because unless you get infeasible amounts of training samples, data scarcity increases exponentially, meaning you can make perfect classifiers, fit to noise.
  - With more features, every data point becomes extreme. Therefore, data points are not ‚close' anymore: the very *concept* of distance and well-defined neighbouring points ceases to work (some work for more dimensions than others/there may be some exceptions).



Solutions high-dimensional data

- What can we do?
  - Feature engineering
  - Feature selection
  - Dimensionality reduction

Feature engineering

- Won’t go into detail, since domain-specific
- Example:
  - Risk for diabetes type 2. Data includes height in cm, weight in kg, age, ethnicity, and sex. → collapse this into bmi categories (corrected for ethnicity) → 1 highly informative feature from 5 separate ones.

Feature engineering

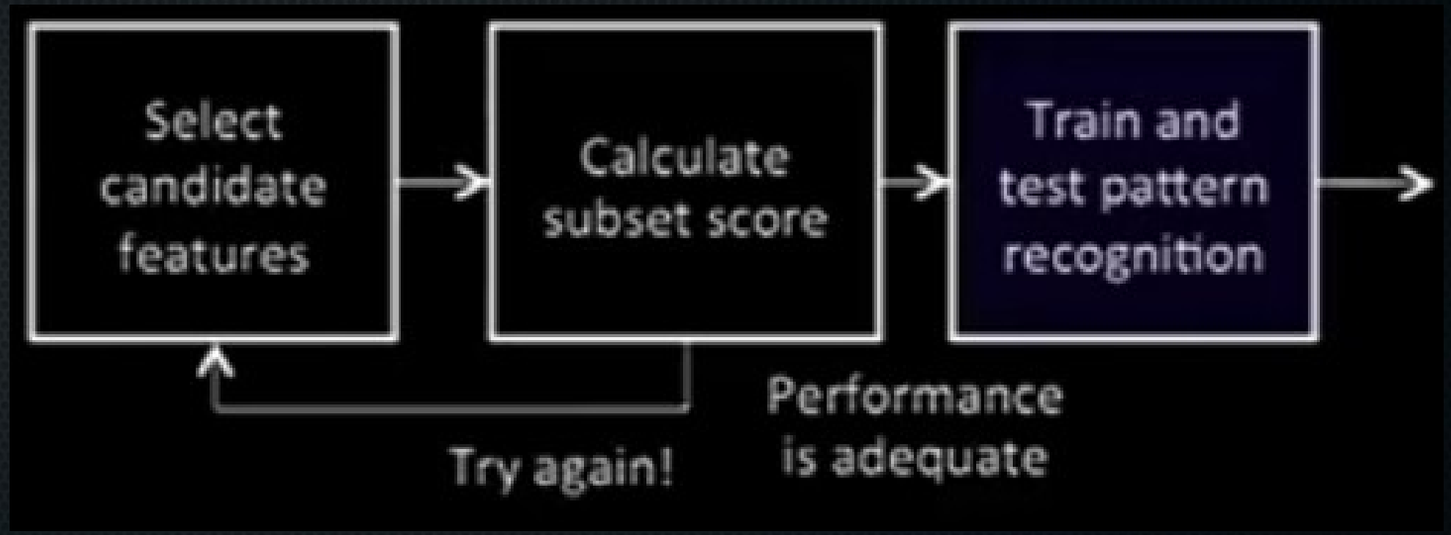
- Won’t go into detail, since domain-specific
- Example:
  - Risk for diabetes type 2. Data includes height in cm, weight in kg, age, ethnicity, and sex. → collapse this into bmi categories (corrected for ethnicity) → 1 highly informative feature from 5 separate ones. → manual dimensionality reduction.
- Requires that you already know factors that play a role. Often not the case or at least incomplete knowledge.

Feature selection

- Select a subset of features that gives high performance, avoid dimensionality problems.
- Two main ways:
  - Filter
  - Wrapper

Feature selection: filtering

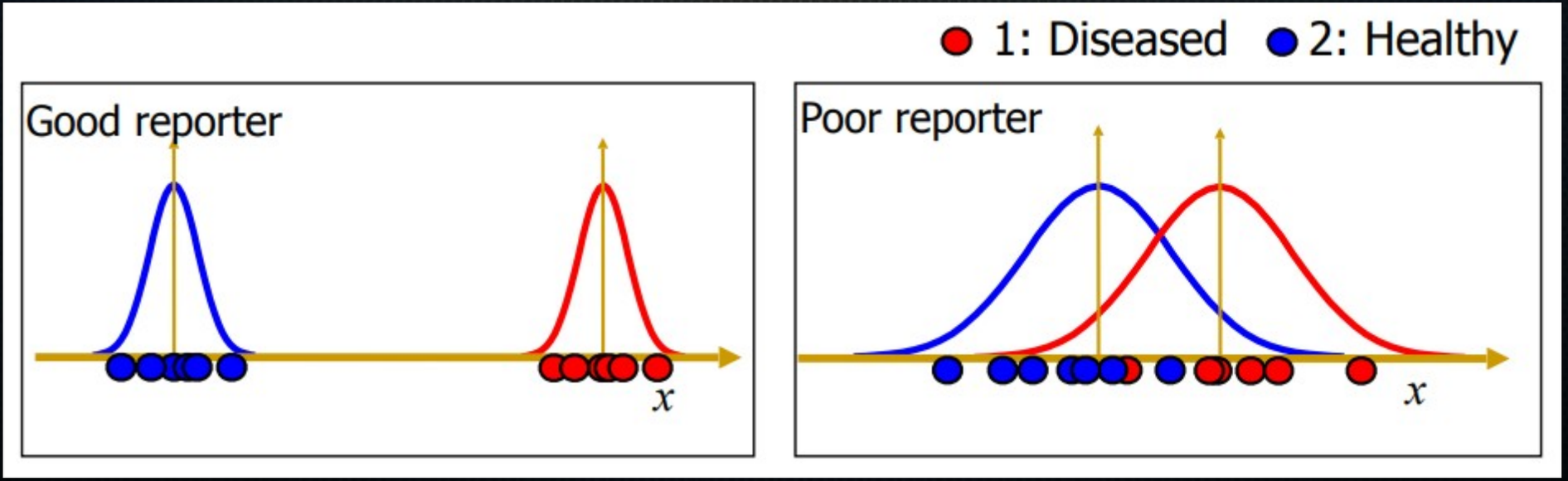
- Use some stand-alone metric to decide how informative each feature is → select *n* best ones → make your classifier



Source: https://www.youtube.com/watch?v=JA9W72UWR0c&list=PL-DP1aeg73q2vMsAgwguwMUQn0JQ3DalV&index=3

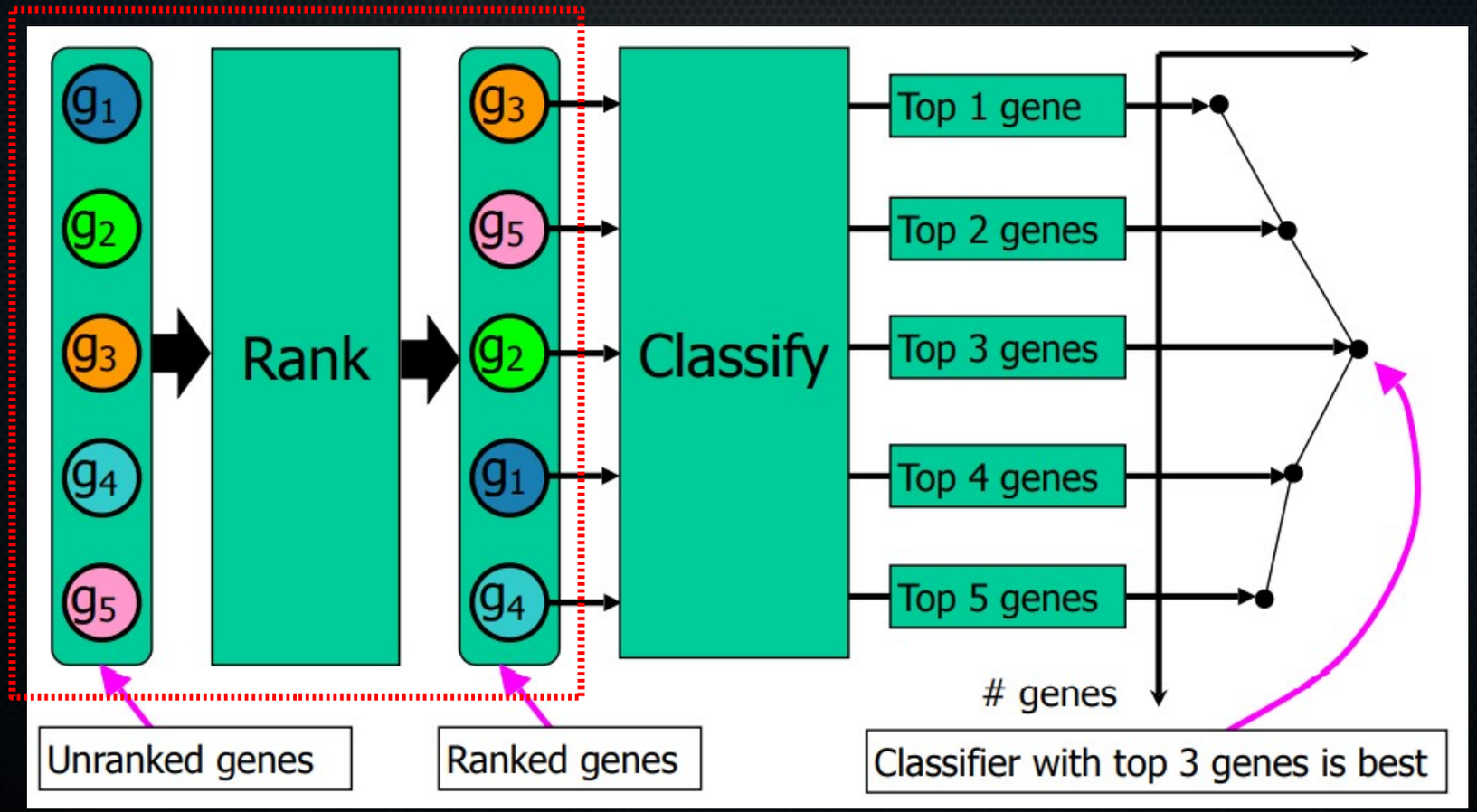
Feature selection: filtering

- Use some stand-alone metric to decide how informative each feature is → select *n* best ones → make your classifier
- Example: per-gene t-test on differences in mean expression between classes + multiple-testing correction



Feature selection: filtering

- Use some stand-alone metric to decide how informative each feature is → select *n* best ones → make your classifier

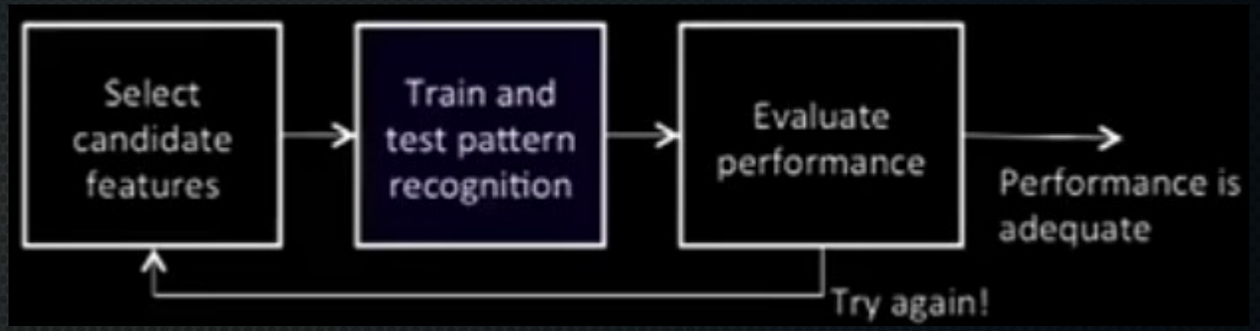


Feature selection: filtering

- Use some stand-alone metric to decide how informative each feature is → select *n* best ones → make your classifier. Filter before training.
- Many more options like mutual information, Pearson correlation, etc.

Feature selection: wrapper

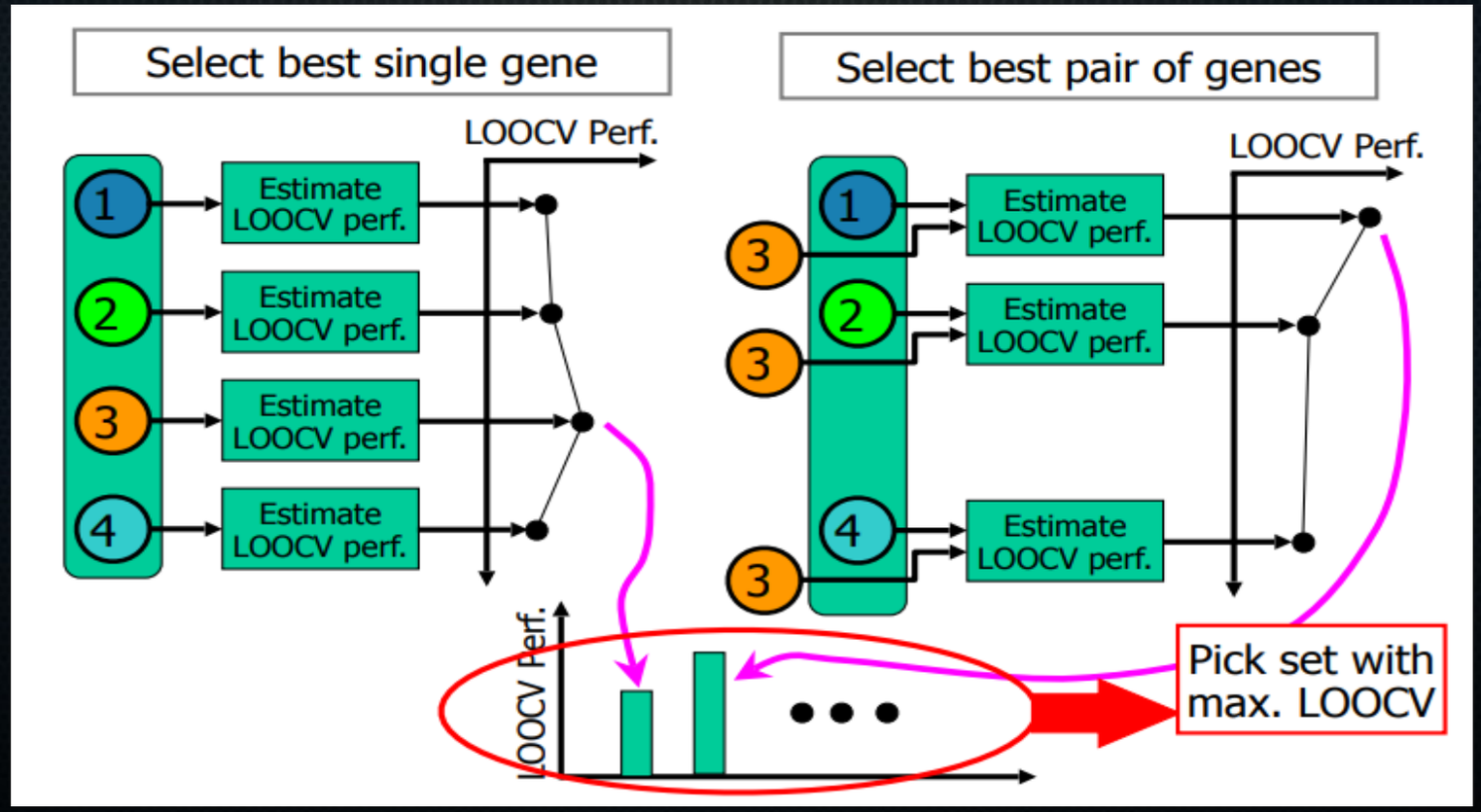
- Make a separate classifier for each feature and sets of features (up to some number) → pick features that make the best classifier.
- Wrapper around normal training procedure.



Source: https://www.youtube.com/watch?v=JA9W72UWR0c&list=PL-DP1aeg73q2vMsAgwguwMUQn0JQ3DalV&index=3

Feature selection: wrapper

- Make a separate classifier for each feature and sets of features (up to some number) → pick features that make the best classifier.



Feature selection regime comparison

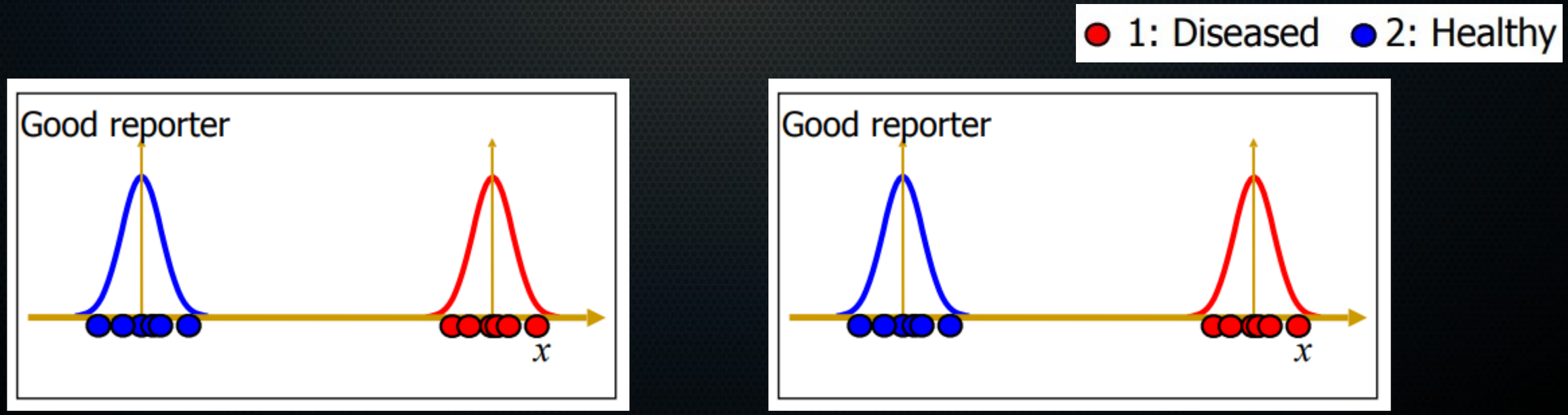
- Filtering assumes some proxy for classifier performance, wrapping directly tests how well a feature performs in your problem of interest.
- Filtering computationally feasible when wrapping is prohibitive (1000 genes, 1.27 * 10^31 possible combinations).
- Wrapping is specific: features you pick for a Random Forest might differ from those for SVM, which might differ from those for a logistic regression.

Problems greedy forward feature selection

- In the examples shown: combine top *n* features for final classifier by adding the next-best feature until performance decreases.
- This is called greedy forward feature selection.

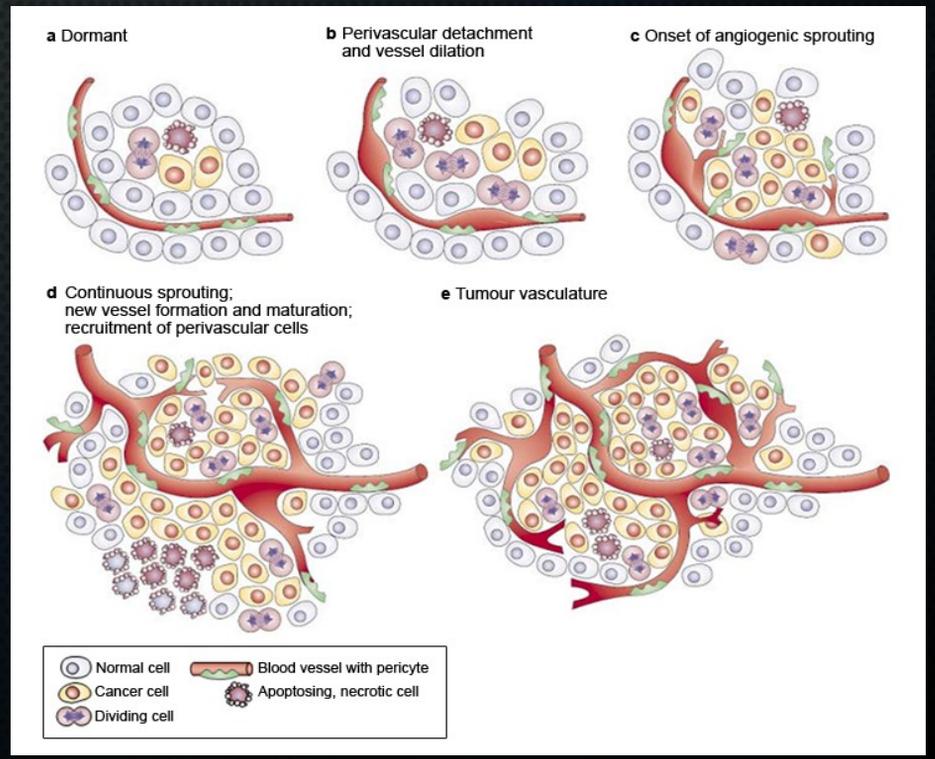
Problems greedy forward feature selection

- Not guaranteed to get the best *combination* of features.
- Will a cancer cell metastatise?
  - Selected features might be: p53 mutated, repair mechanism mutated



Problems greedy forward feature selection

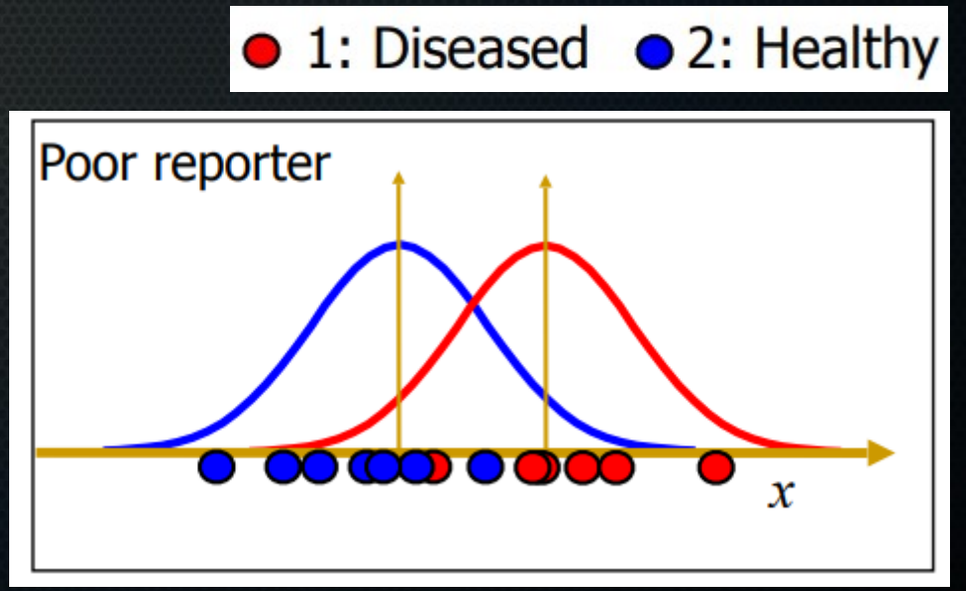
- Not guaranteed to get the best *combination* of features.
- Will a cancer cell metastatise?
  - Selected features might be: p53 mutated, repair mechanism mutated
  - Mechanistic → p53 allows more division, repair mechanism aberration leads to mutations in 5 key areas (neovascularisation, metabolic switching, immune evasion, cell motility, ECM modification)



Source: https://www.med-iq.com/files/cme/chapter/text-basedSuppliments/SA196/figure_2_lg.jpg

Problems greedy forward feature selection

- Not guaranteed to get the best *combination* of features.
- Will a cancer cell metastatise?
  - Selected features might be: p53 mutated, repair mechanism mutated
  - Mechanistic → p53 allows more division, repair mechanism aberration leads to mutations in 5 key areas (neovascularisation, metabolic switching, immune evasion, cell motility, ECM modification)
  - Each of the 5 separately:



Problems greedy forward feature selection

- Not guaranteed to get the best *combination* of features.
- Will a cancer cell metastatise?
  - Selected features might be: p53 mutated, repair mechanism mutated
  - Mechanistic → p53 allows more division,
    repair mechanism aberration leads to
    mutations in 5 key areas
    (neovascularisation, metabolic switching,
    immune evasion, cell motility,
    ECM modification)
  - 5 combined:

Problems greedy forward feature selection

- Not guaranteed to get the best *combination* of features.
- Will a cancer cell metastatise?
  - Selected features might be: p53 mutated, repair mechanism mutated
  - Mechanistic → p53 allows more division, repair mechanism aberration leads to mutations in 5 key areas (neovascularisation, metabolic switching, immune evasion, cell motility, ECM modification)
  - Greedy ffs will select p53 and repair mechanism, since repair mechanism is a proxy for all 5 key areas being mutated at once. Then, adding genes for one of these 5 key areas won't give a better score. However, all 5 key areas together would be perfect!

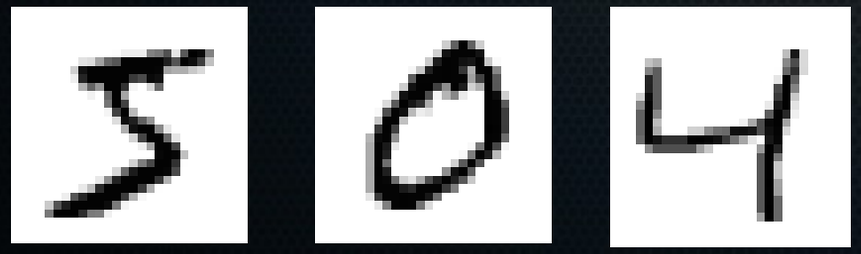
Summary so far

- You can manually combine features together with domain knowledge.
- You can select a subset of features to train on, either by pre-selection (filtering) or training many classifiers and picking the features that lead to the best-performing one(s) (wrapping).

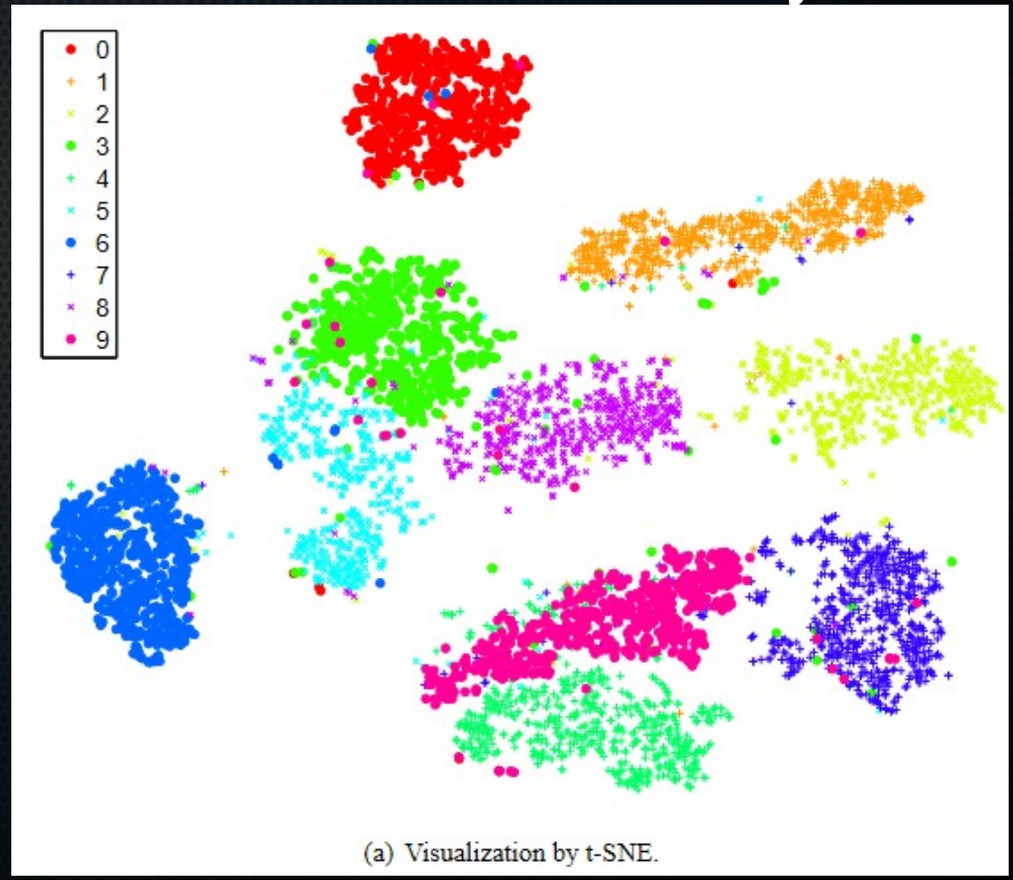
Dimensionality reduction

- Extrinsic dimensionality may be high, but the data we care about might lie in a specific subspace of lower dimensionality.

- MNIST:
  28*28 = 784-dimensional data



Source: http://colah.github.io/posts/2014-10-Visualizing-MNIST/

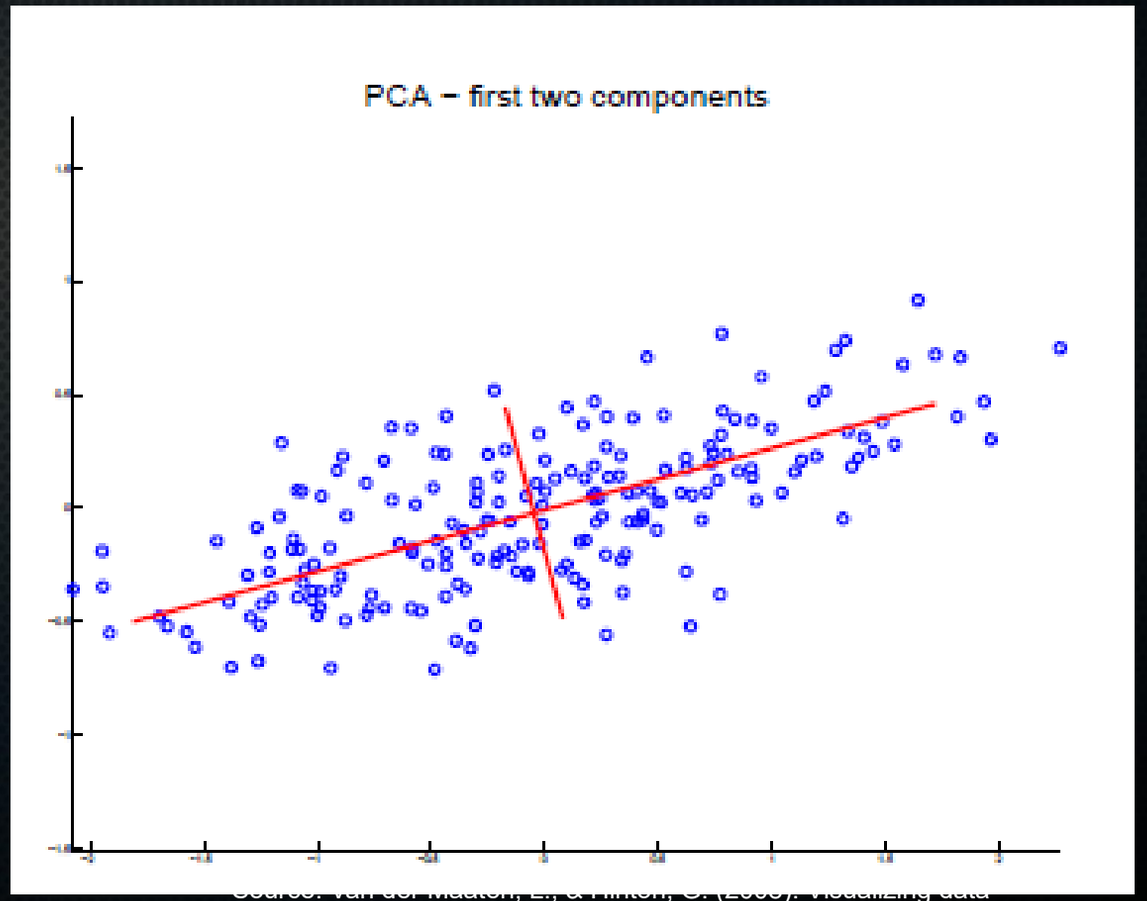


(a) Visualization by t-SNE.

Source: Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).

Dimensionality reduction

- We can make better use of our dimensions by redefining them in a way that maximises information in each dimension
- Here: linear manipulation (rotation) of the data such that each new dimension captures most of the variance → PCA



Source: van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).

More after the practical break!

BREAK FOR PRACTICAL

- Zie hoe goed een classifier wordt als je begint met correcte features en steeds meer dimensies met noise toevoegt.
- Zelf filteren op basis van t-test in dataset met expressie 100 ‚genen‘ (mag nepdata zijn) → 10 informatieve, 90 noisy bijv.
- Voorbeeld wrapper met 10 genen voor logistische regressie.