Unsupervised Learning

ACTL3142 & ACTL5110 Statistical Machine Learning for Risk Applications

Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani

UNSW SYDNEY

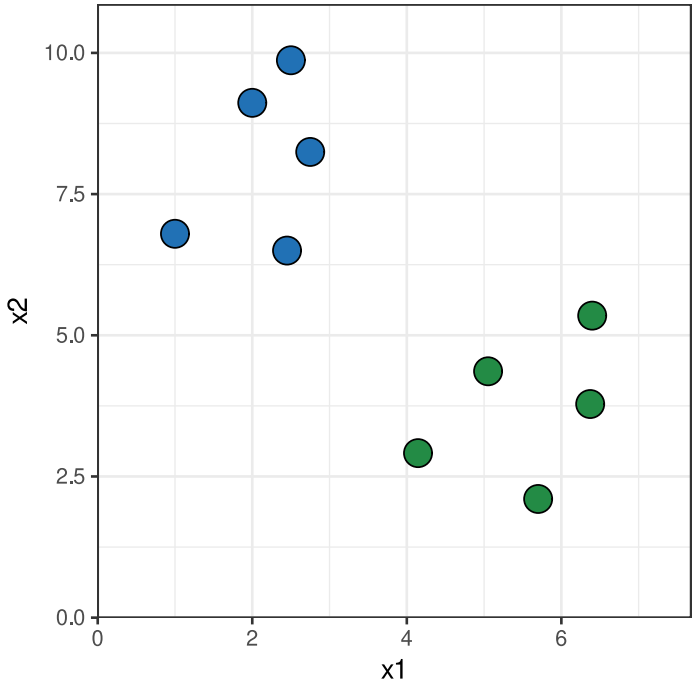
Lecture Outline

- **Unsupervised Learning**
- K-Means Clustering
- Demo: MNIST
- Hierarchical Clustering
- Dimension Reduction
- Demo: PCA on MNIST

UNSW SYDNEY

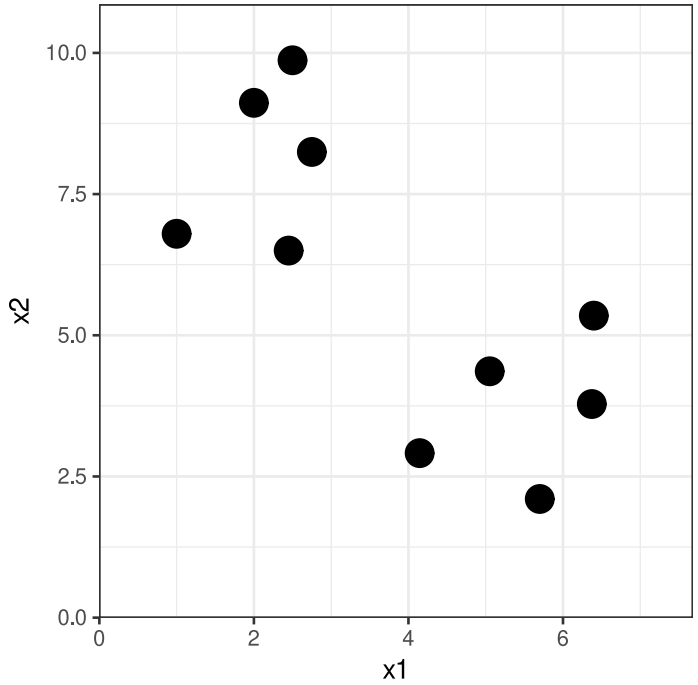
Supervised vs Unsupervised Learning

Supervised



- Data: $X_1, X_2, \ldots, X_p, Y$
- Goal: Predict Y using $X_1, X_2, \ldots, X_p$

Unsupervised



- Data: $X_1, X_2, \ldots, X_p$
- Goal: Discover interesting things using $X_1, X_2, \ldots, X_p$

Challenge of Unsupervised Learning

- Typical questions
  - Is there an informative way to visualize the data?
  - Can we discover subgroups among the variables?
- More subjective than supervised learning
  - There's no simple goal for the analysis
- Hard to assess the results obtained from unsupervised learning methods
  - There's no way to validate your results on an independent data set

Clustering vs. PCA

- Both seek to simplify the data via a small number of summaries
- Different mechanisms
  - Clustering: find homogeneous subgroups among the observations
  - PCA: find a low-dimensional representation of the observations that explain a good fraction of the variance
- Both useful for visualisation

UNSW SYDNEY

Clustering Methods

- A very broad set of techniques for finding subgroups, or clusters, in a data set
- The observations within each group are quite similar to each other
- Need to specify what it means for two or more observations to be similar or different (often domain-specific)
- Two clustering methods
  - K-means clustering
    - partition the observations into a pre-specified number of clusters
  - Hierarchical clustering
    - do not know in advance how many clusters we want
    - creates a *dendrogram*, a tree representation of clusters (for $K = 1, 2, 3, \ldots, n$)

Applications of Clustering

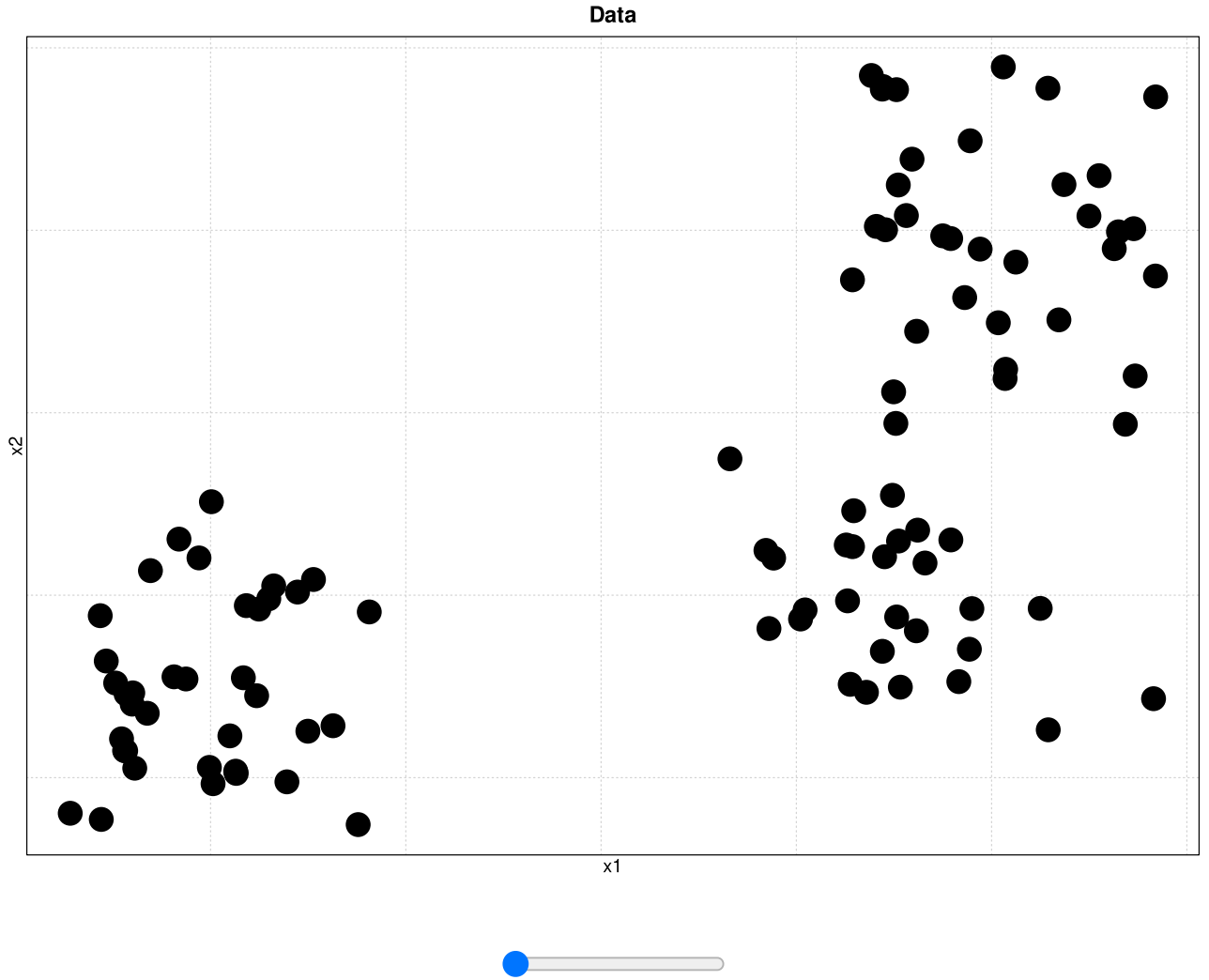
- Market segmentation
- Fraud detection
- Group patients by medical condition (e.g types of diabetes)
- Clustering of documents by type
- Compression of information (e.g. representative policies in a portfolio)

Lecture Outline

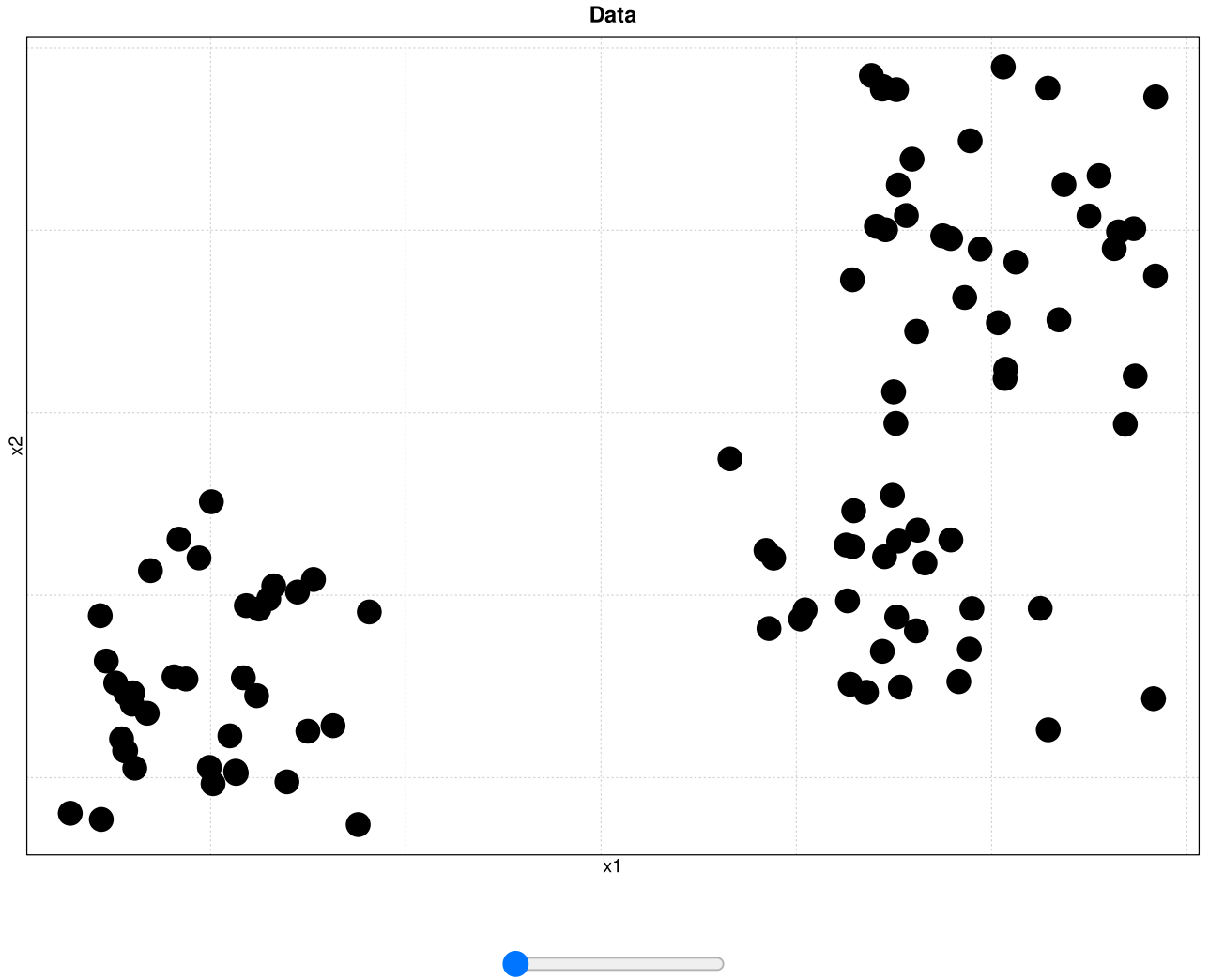
- Unsupervised Learning
- **K-Means Clustering**
- Demo: MNIST
- Hierarchical Clustering
- Dimension Reduction
- Demo: PCA on MNIST

UNSW SYDNEY

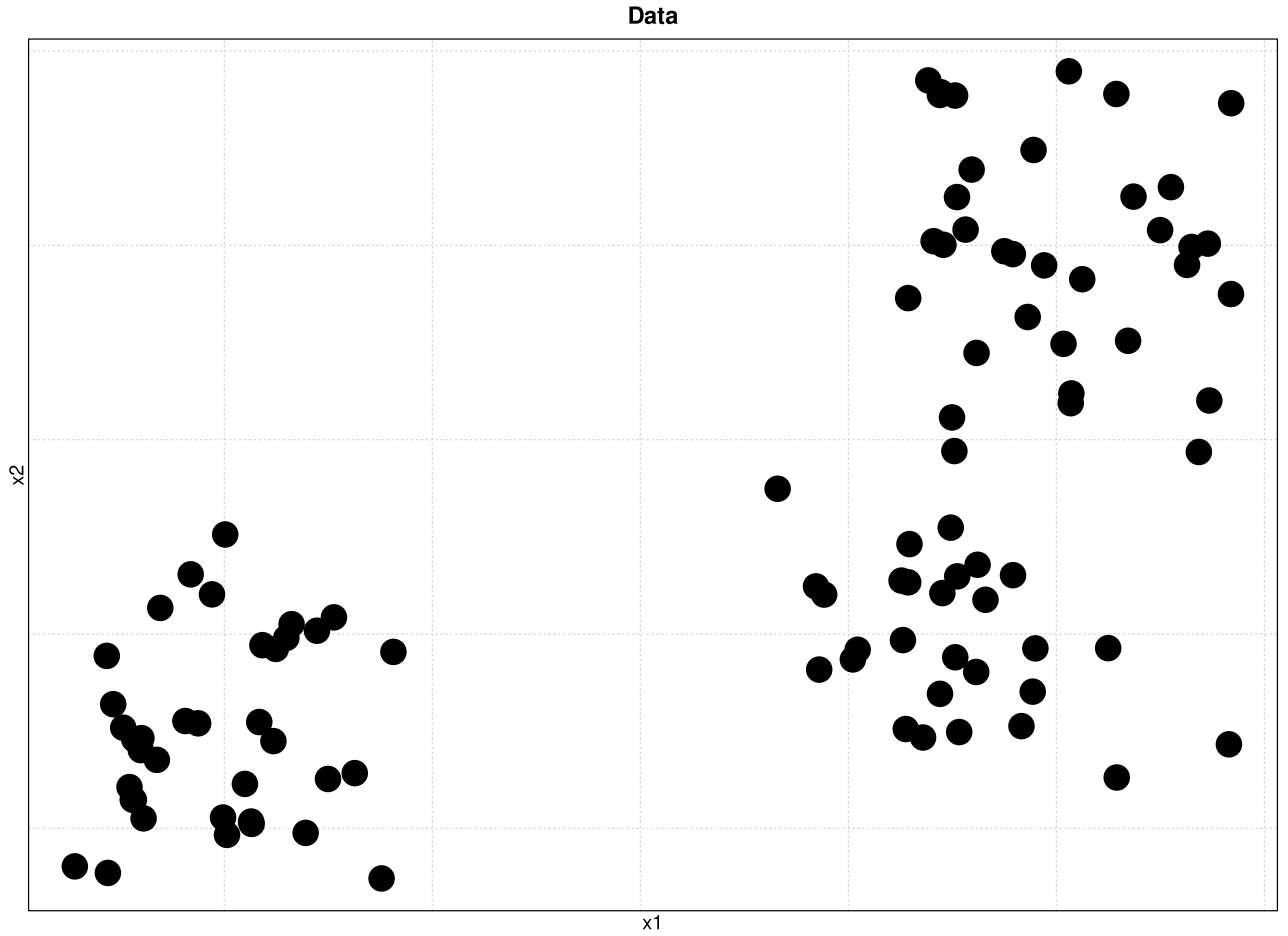
K-means clustering: Demonstration I



K-means clustering: Demonstration II



K-means clustering: Demonstration III



UNSW SYDNEY

K-Means Clustering

Denote $C_1, \ldots, C_K$ as the sets containing the indices of the observations in each cluster.

Each observation belongs to at least one of the K clusters

$$C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \ldots, n\}.$$

The clusters are non-overlapping; no observation belongs to more than one cluster

$$C_k \cap C_{k'} = \emptyset \text{ for all } k \neq k'.$$

Mathematical formulation: Clustering

A good clustering is one for which the within-cluster variation is as small as possible

$$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} W(C_k).$$

The most common choice of $W(\cdot)$

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

where $|C_k|$ is the number of observations in the kth cluster.

It turns out that

$$W(C_k) = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \bar{x}_{kj})^2$$

where $\bar{x}_k$ is the mean of the observations in the kth cluster, a.k.a. the *centroid*.

K-Means algorithm

The optimisation problem that defines K-means clustering is

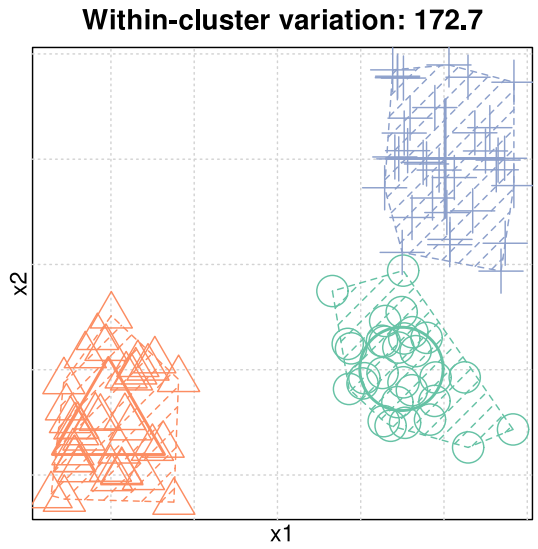
$$\min_{C_1,\ldots,C_k} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2.$$

It's a difficult problem to solve precisely, but we have a very simple algorithm that provides a local optimum:

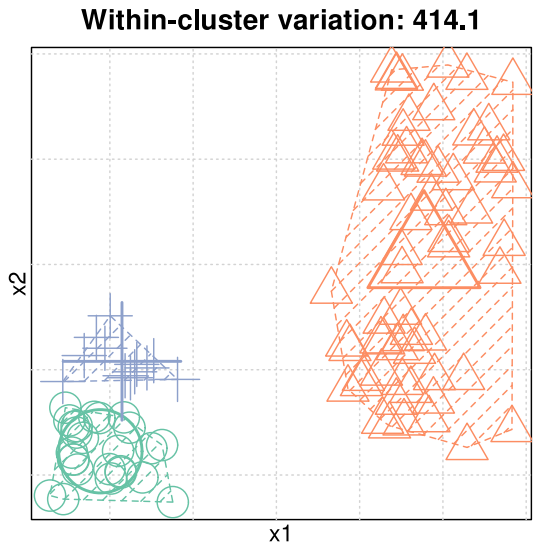
1. Randomly initialise K cluster centres/centroids
2. Assign each observation to the cluster whose centroid is closest
   - "Closest" is defined using Euclidean distance
3. For each of the K clusters, compute the cluster centroid
   - The "centroid" is the vector of the means for the observations in the kth cluster
4. Repeat 2 & 3 until convergence

UNSW SYDNEY

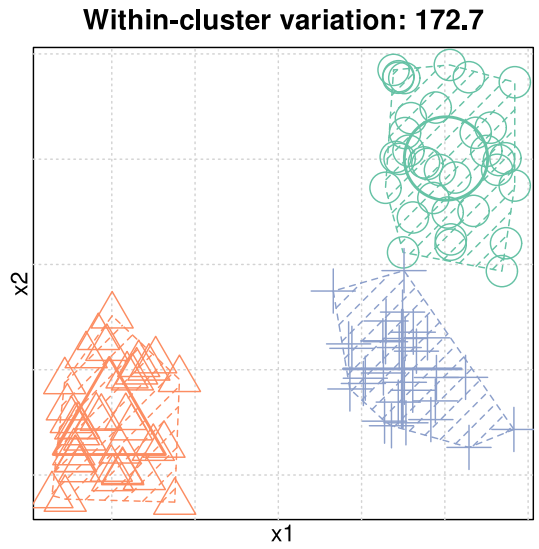
K-means clustering: Local Optima



Clusters after a seed of 1

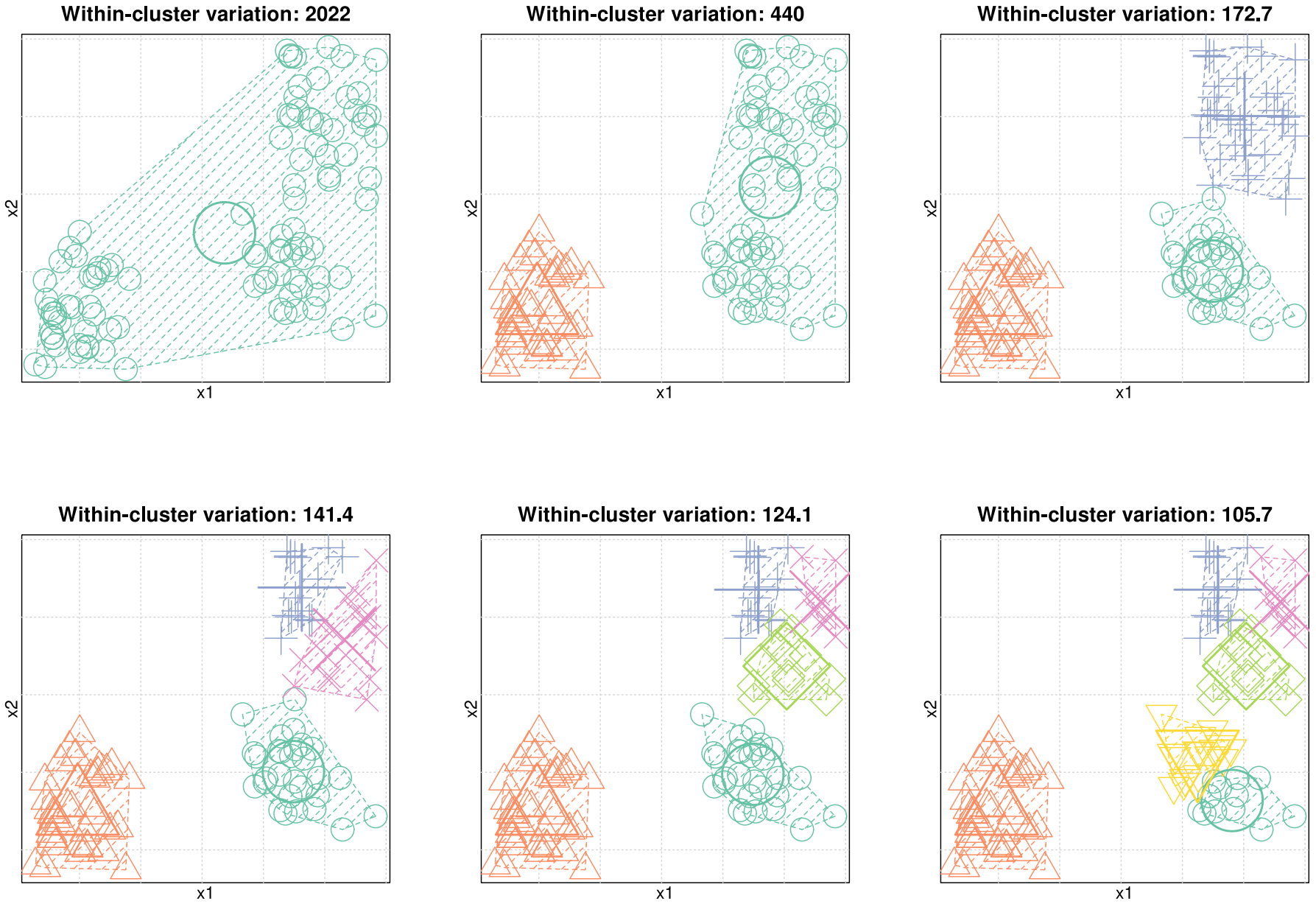


Clusters after a seed of 2

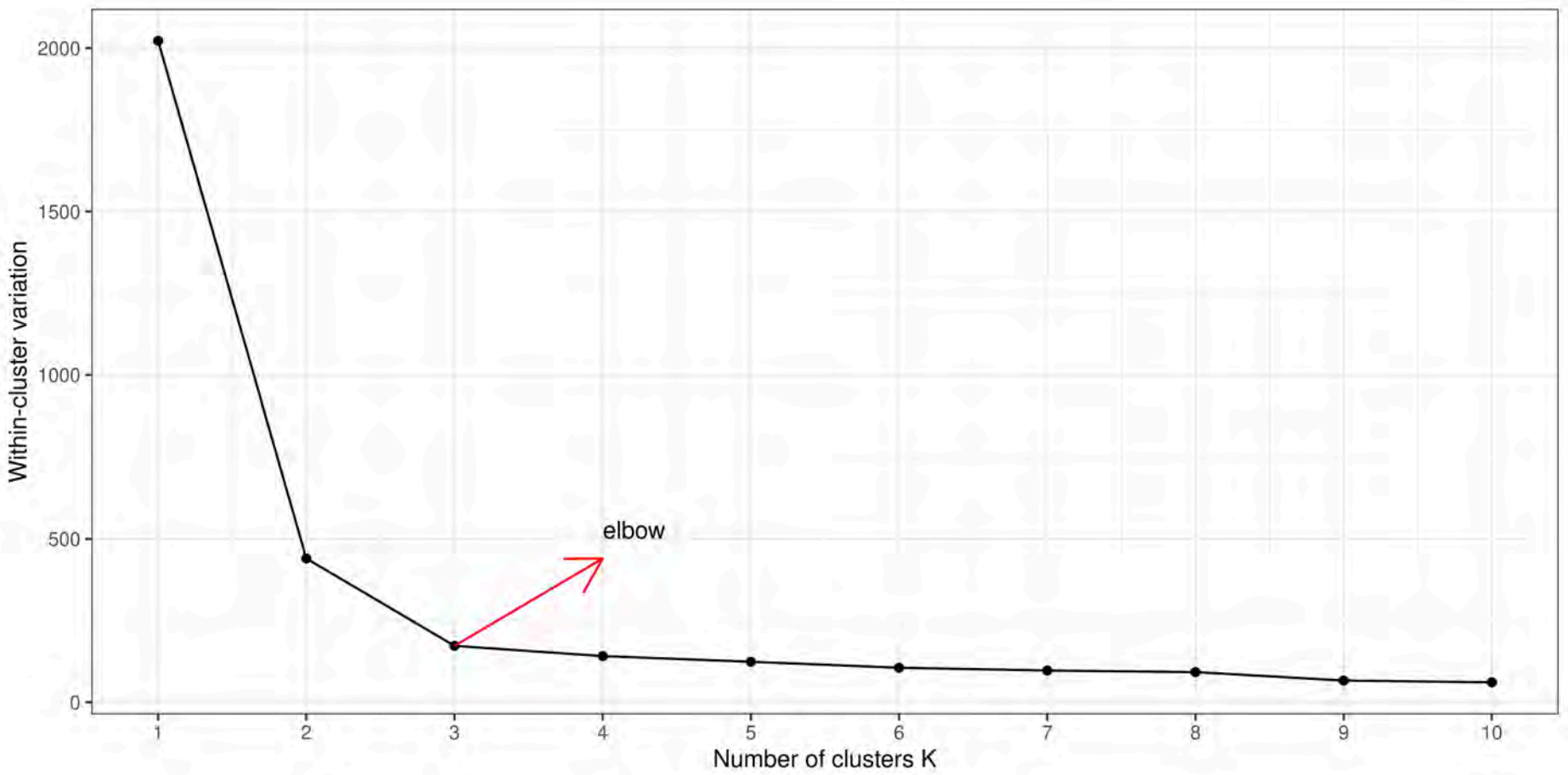


Clusters after a seed of 9

- The algorithm finds a local rather than a global optimum
- Results depend on initial centroids used
- Important to run the algorithm multiple times and select the best solution (minimum within-cluster variation)


Within-cluster variation: 2022
Within-cluster variation: 440
Within-cluster variation: 172.7
Within-cluster variation: 141.4
Within-cluster variation: 124.1
Within-cluster variation: 105.7
x1
x2

What is the right value of K?



Lecture Outline

- **Unsupervised Learning**
- K-Means Clustering
- Demo: MNIST
- Hierarchical Clustering
- Dimension Reduction
- Demo: PCA on MNIST

UNSW SYDNEY

The data

R | Python

```r
train_df <- read.csv("mnist_train.csv")
train_df
```

```
# A tibble: 60,000 × 785
      X0    X1    X2    X3    X4    X5    X6    X7    X8    X9   X10   X11   X12
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1     0     0     0     0     0     0     0     0     0     0     0     0     0
 2     0     0     0     0     0     0     0     0     0     0     0     0     0
 3     0     0     0     0     0     0     0     0     0     0     0     0     0
 4     0     0     0     0     0     0     0     0     0     0     0     0     0
 5     0     0     0     0     0     0     0     0     0     0     0     0     0
 6     0     0     0     0     0     0     0     0     0     0     0     0     0
 7     0     0     0     0     0     0     0     0     0     0     0     0     0
 8     0     0     0     0     0     0     0     0     0     0     0     0     0
 9     0     0     0     0     0     0     0     0     0     0     0     0     0
10     0     0     0     0     0     0     0     0     0     0     0     0     0
# ℹ 59,990 more rows
# ℹ 772 more variables: X13 <dbl>, X14 <dbl>, X15 <dbl>, X16 <dbl>, X17 <dbl>,
#   X18 <dbl>, X19 <dbl>, X20 <dbl>, X21 <dbl>, X22 <dbl>, X23 <dbl>,
#   X24 <dbl>, X25 <dbl>, X26 <dbl>, X27 <dbl>, X28 <dbl>, X29 <dbl>,
#   X30 <dbl>, X31 <dbl>, X32 <dbl>, X33 <dbl>, X34 <dbl>, X35 <dbl>,
#   X36 <dbl>, X37 <dbl>, X38 <dbl>, X39 <dbl>, X40 <dbl>, X41 <dbl>,
#   X42 <dbl>, X43 <dbl>, X44 <dbl>, X45 <dbl>, X46 <dbl>, X47 <dbl>, …
```

Only 19% of the data is non-zero.

UNSW SYDNEY

Which is the odd one out?

```
  X479 X480 X481       X482      X483      X484      X485      X486      X487
4    0    0    0 0.00000000 0.0000000 0.0000000 0.0000000 0.1882353 0.8666667
7    0    0    0 0.00000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
6    0    0    0 0.01960784 0.5294118 0.9882353 0.9882353 0.7058824 0.0627451
       X488       X489      X490      X491      X492      X493      X494
4 0.9843137 0.98431370 0.6745098 0.0000000 0.0000000 0.0000000 0.0000000
7 0.0000000 0.99215686 0.9882353 0.9882353 0.9882353 0.1647059 0.0000000
6 0.0000000 0.08235294 0.7960784 0.9921569 0.9686274 0.5058824 0.6784314
       X495      X496      X497      X498      X499
4 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
7 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
6 0.9882353 0.9882353 0.7215686 0.2588235 0.1921569
```

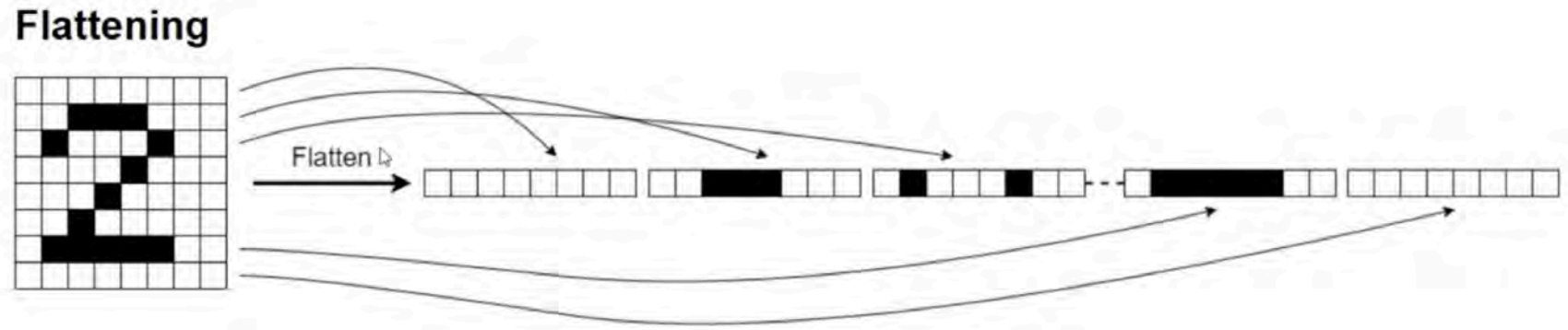
UNSW SYDNEY

MNIST Dataset

The MNIST dataset.

Source: Wikipedia, MNIST database.

The data is flattened



An image turned into a vector.

Preparation

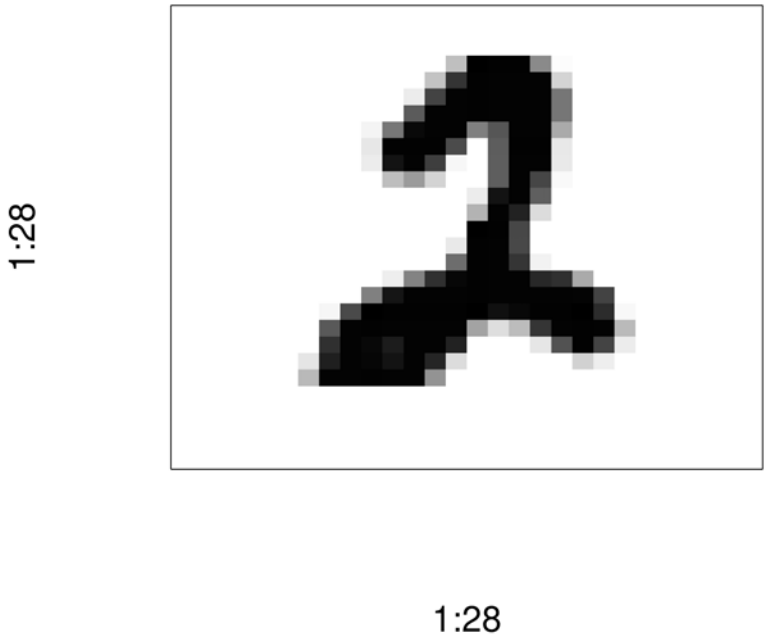
Take just a fraction of the data, and make a plotting function.

R Python

```r
# Separate the features and the labels
x <- as.matrix(train_df[, -785])
y <- train_df$label

# Split the data into train/validation and te
set.seed(88)

test_indices <- sample(1:nrow(x),
    size = 0.2*nrow(x))
x_test <- x[test_indices,]
y_test <- y[test_indices]
x_train_val <- x[-test_indices,]
y_train_val <- y[-test_indices]

train_indices <- sample(1:nrow(x_train_val),
    size = 0.75*nrow(x_train_val))
x_train <- x_train_val[train_indices,]
y_train <- y_train_val[train_indices]
x_val <- x_train_val[-train_indices,]
y_val <- y_train_val[-train_indices]
```

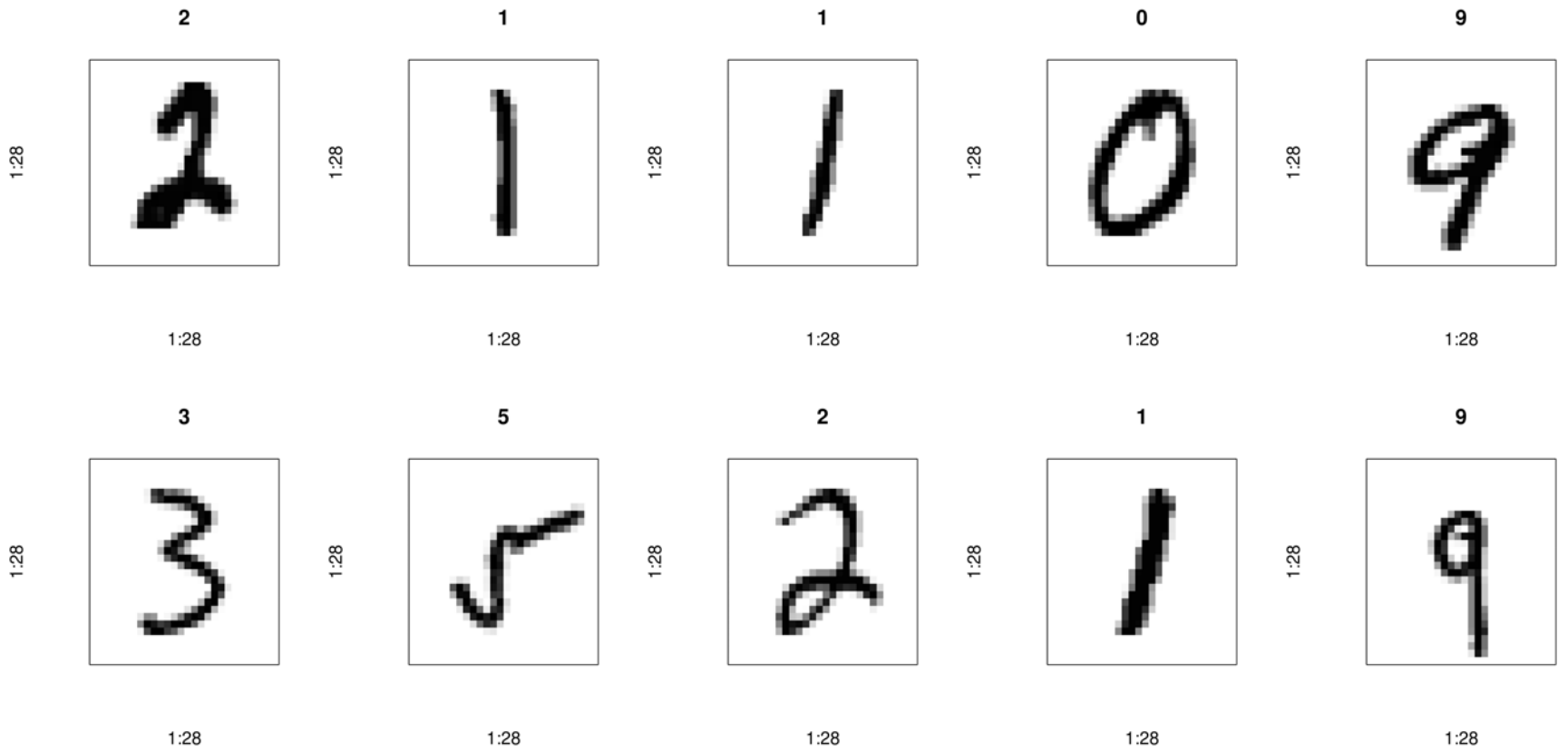
R Python

```r
plot_digit(x_train[1,])
```



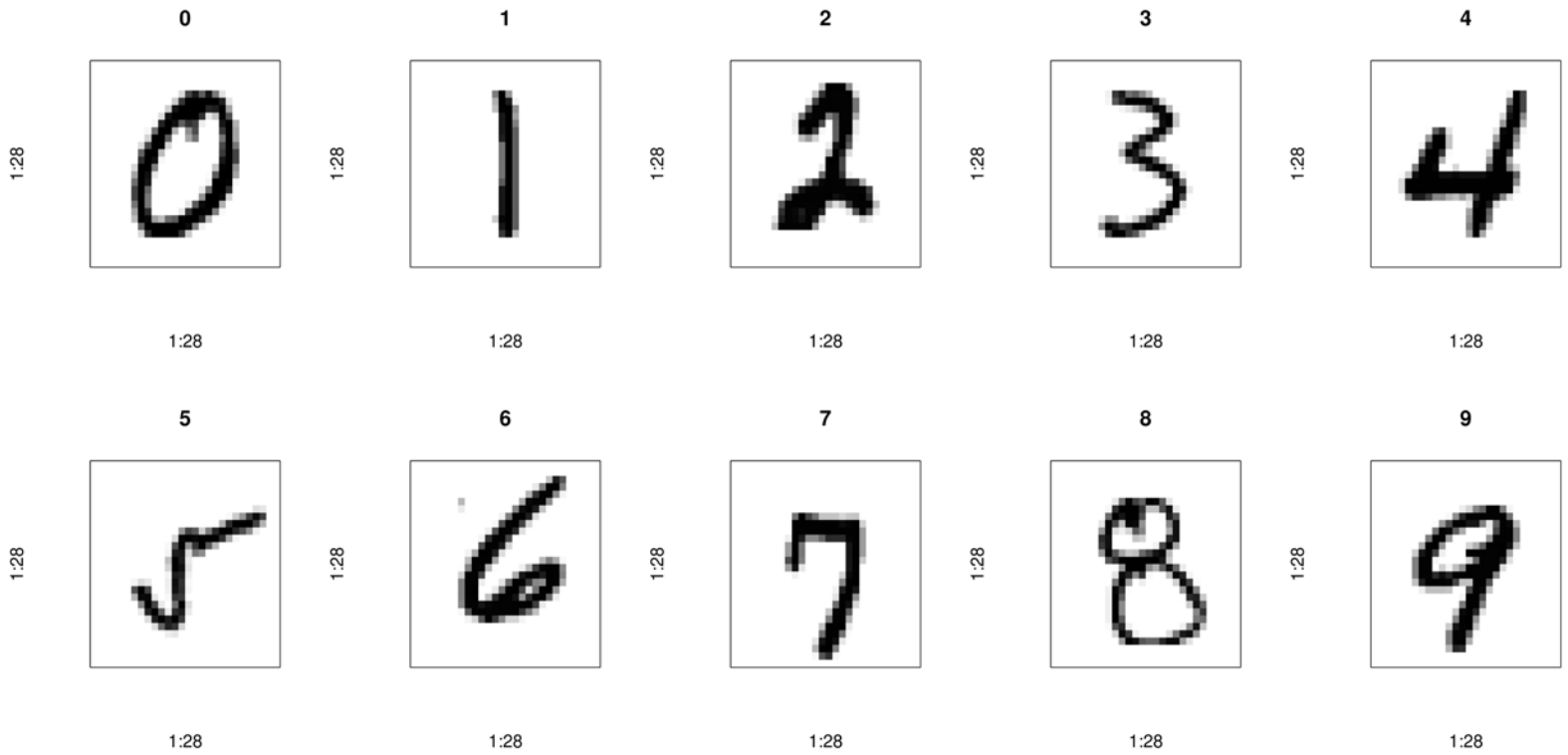
Plotting the data

R | Python



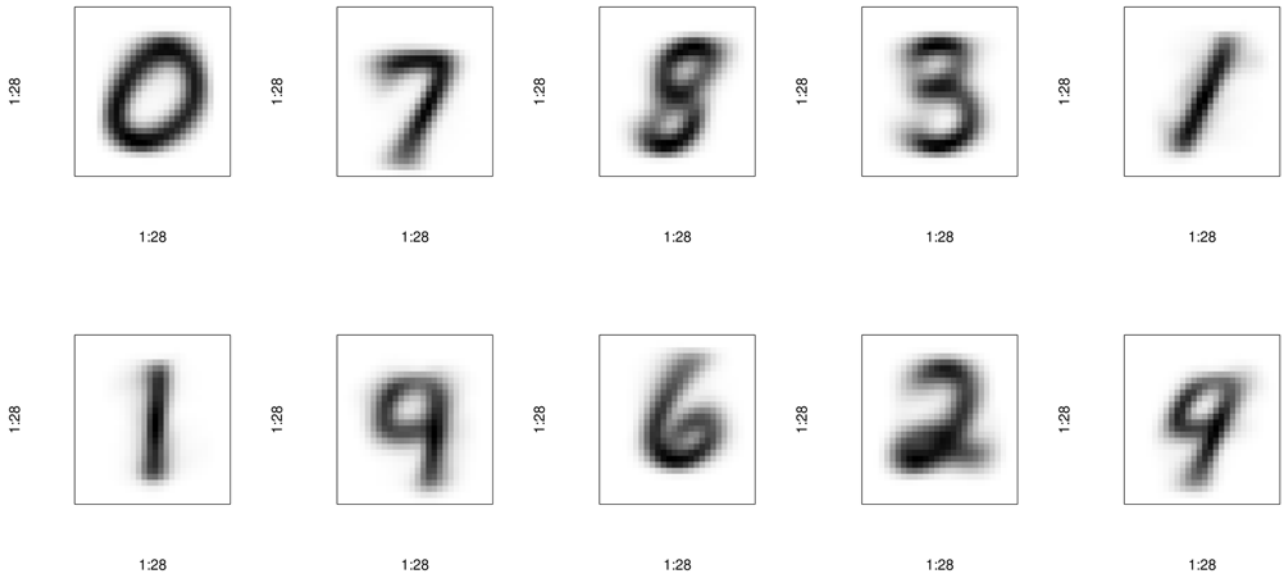
There are 10 natural clusters

```r
# Plot one of each digit
par(mfrow = c(2,5))
for (i in 0:9) {
  plot_digit(x_train[y_train == i,][1,])
  title(i)
}
```



K-means Clustering on MNIST I

```
set.seed(1)
kmeans_out <- kmeans(x_train, centers = 10)
```



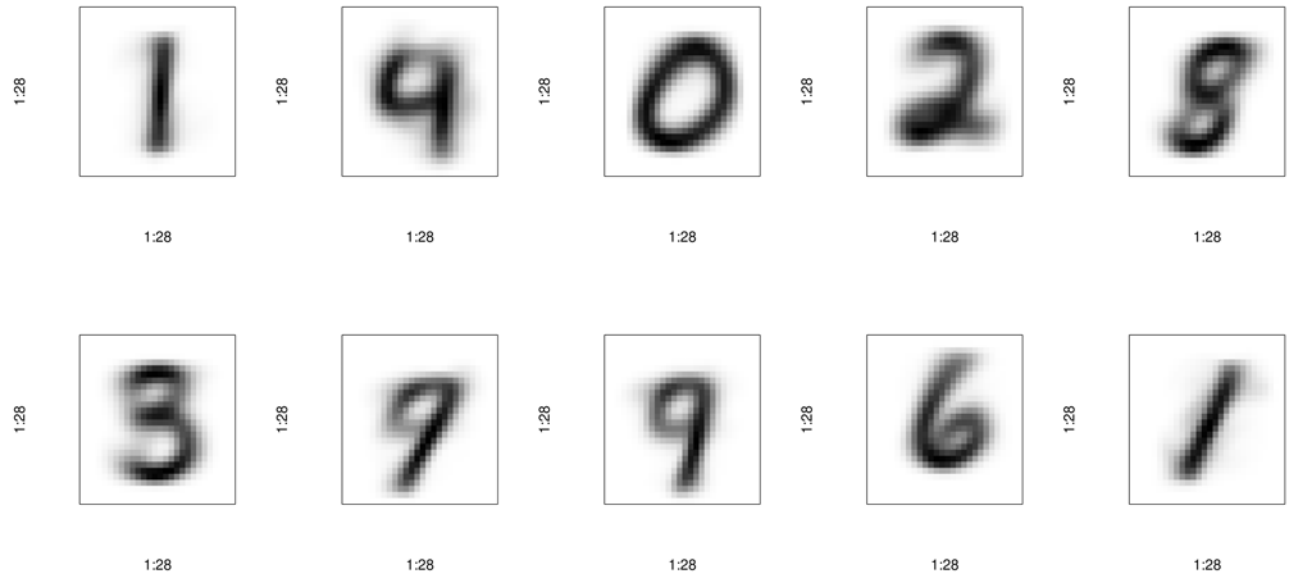
The within-cluster variation is

```
kmeans_out$tot.withinss
```

```
[1] 1413762
```

K-means Clustering on MNIST II

```r
set.seed(2)
kmeans_out <- kmeans(x_train, centers = 10)
```



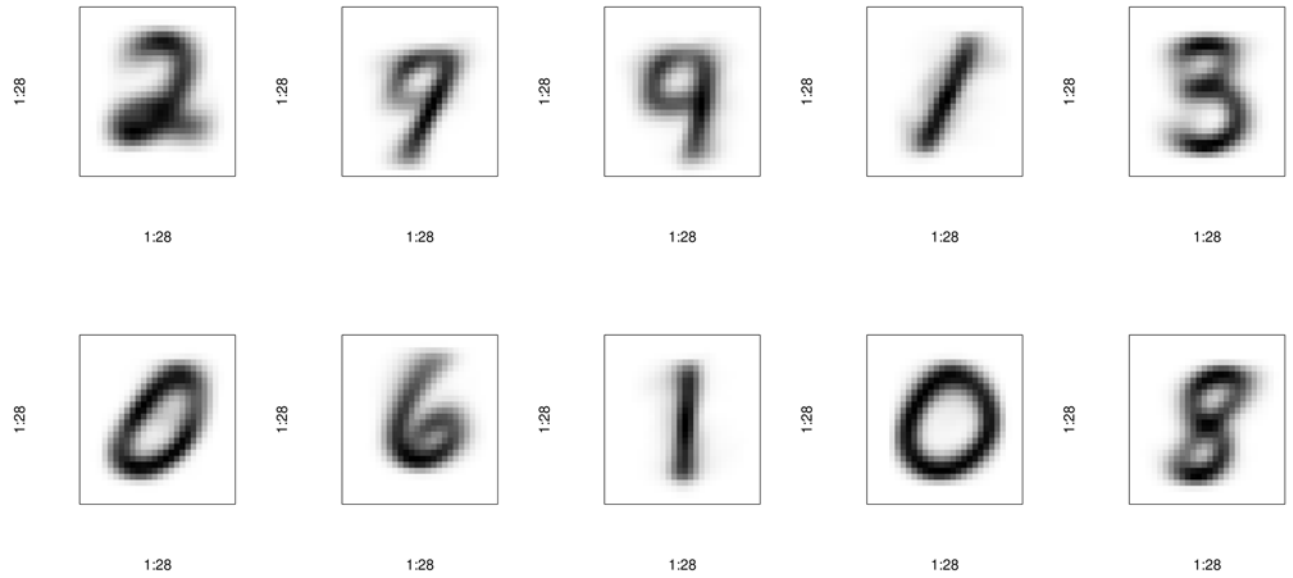
The within-cluster variation is

```r
kmeans_out$tot.withinss
```

```
[1] 1410923
```

K-means Clustering on MNIST III

```
set.seed(3)
kmeans_out <- kmeans(x_train, centers = 10)
```



The within-cluster variation is

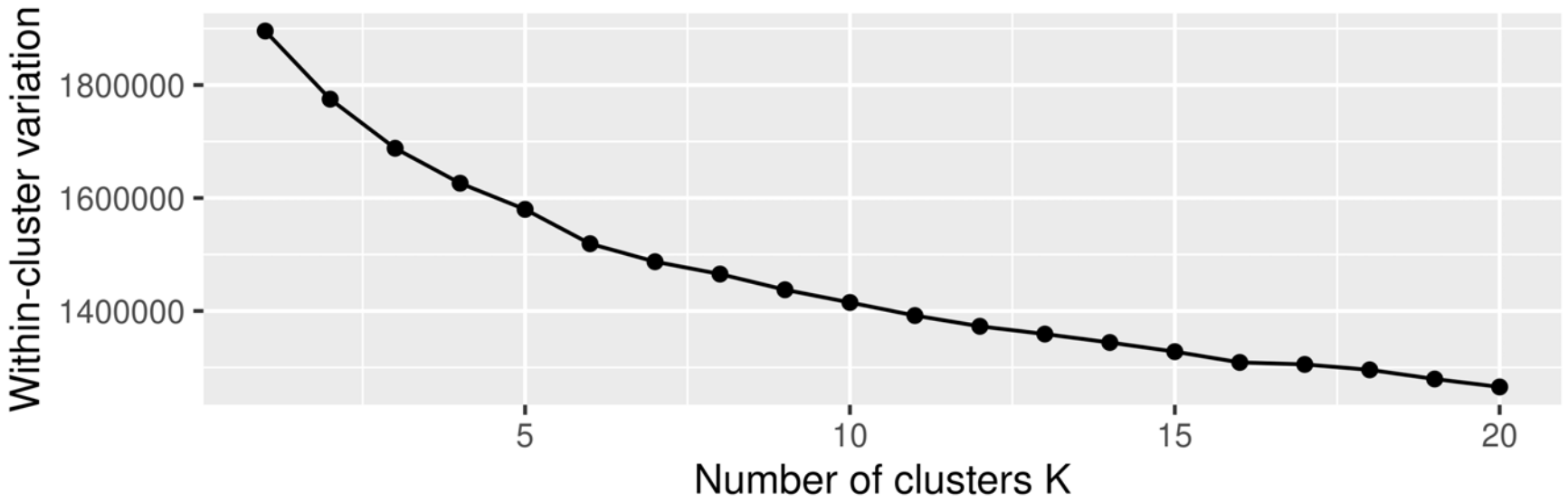
```
kmeans_out$tot.withinss
```

```
[1] 1410441
```

Elbow method MNIST I

R | Python

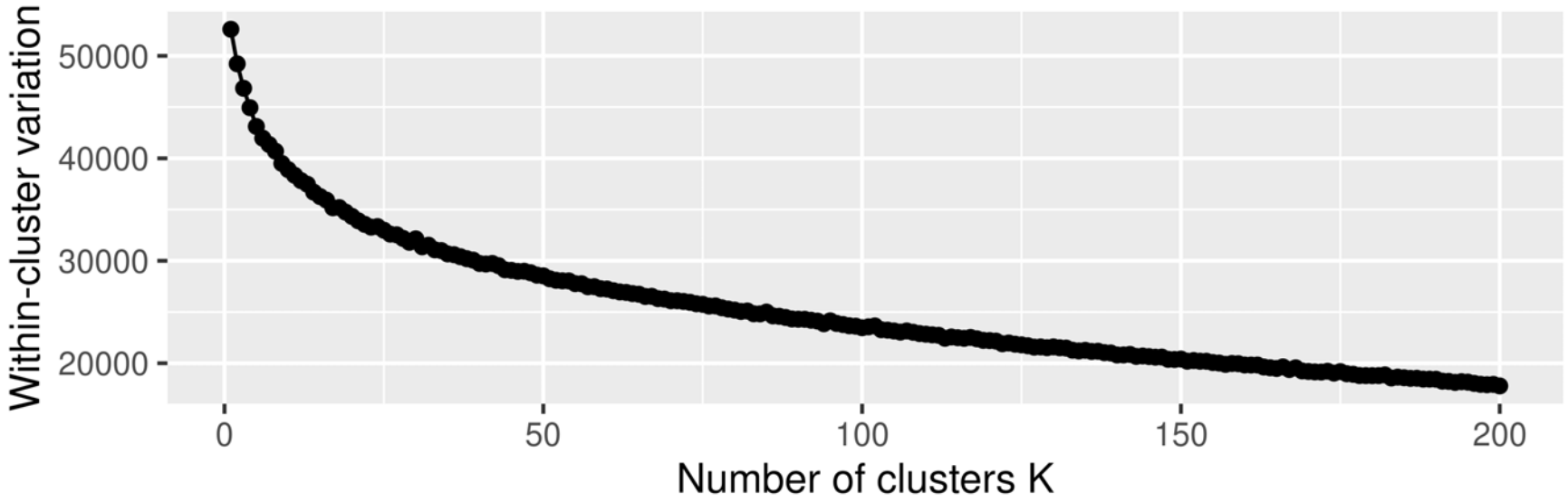
```r
wss <- rep(0, 20)
for (k in 1:20) {
  kmeans_out <- kmeans(x_train, centers = k)
  wss[k] <- kmeans_out$tot.withinss
}
```



Elbow method MNIST II

R | Python

```r
wss <- rep(0, 200)
x_tiny_subset <- x_train[1:1000,]
for (k in 1:200) {
  kmeans_out <- kmeans(x_tiny_subset, centers = k)
  wss[k] <- kmeans_out$tot.withinss
}
```



Lecture Outline

- Unsupervised Learning
- K-Means Clustering
- **Demo: MNIST**
- Hierarchical Clustering
- Dimension Reduction
- Demo: PCA on MNIST

Hierarchical Clustering

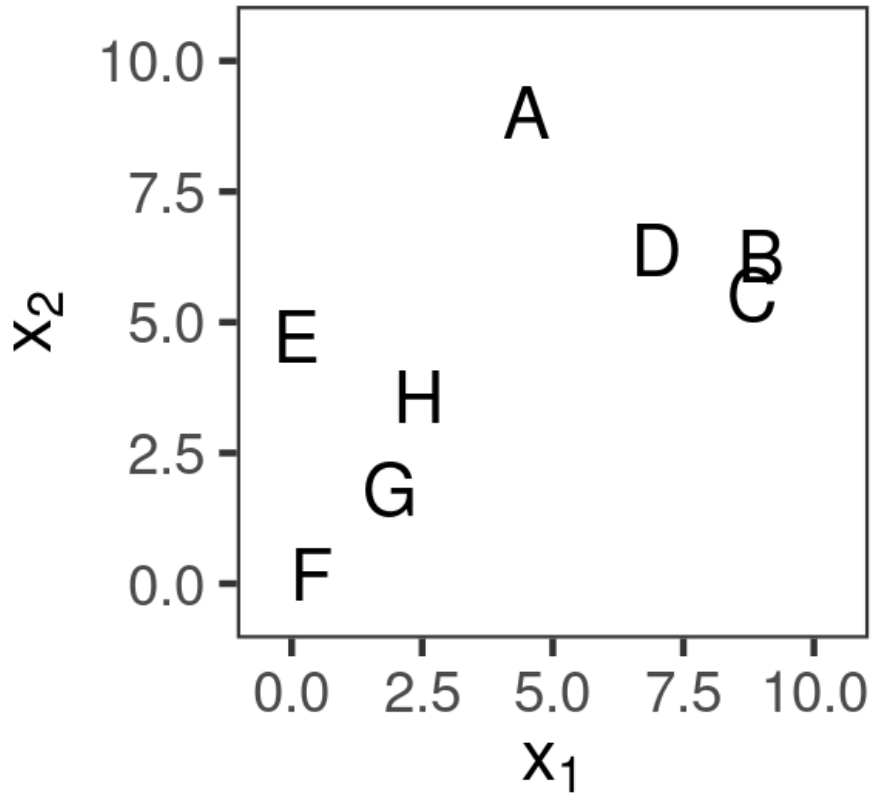
- No need to specify the number of clusters K
- Result is a tree-based representation, called a dendrogram
- Allows user to choose any distance metric
  - K-means restricted us to Euclidean distance
- Focus on bottom-up or agglomerative clustering
  - start from the leaves
  - combine the clusters up to the trunk

Algorithm:

1. Treat each of the n observations as its own cluster
2. For $i = n, n-1, \ldots, 2$:
   1. Compute the pairwise inter-cluster dissimilarities among the i clusters
   2. Identify the pair of clusters that are least dissimilar and merge them

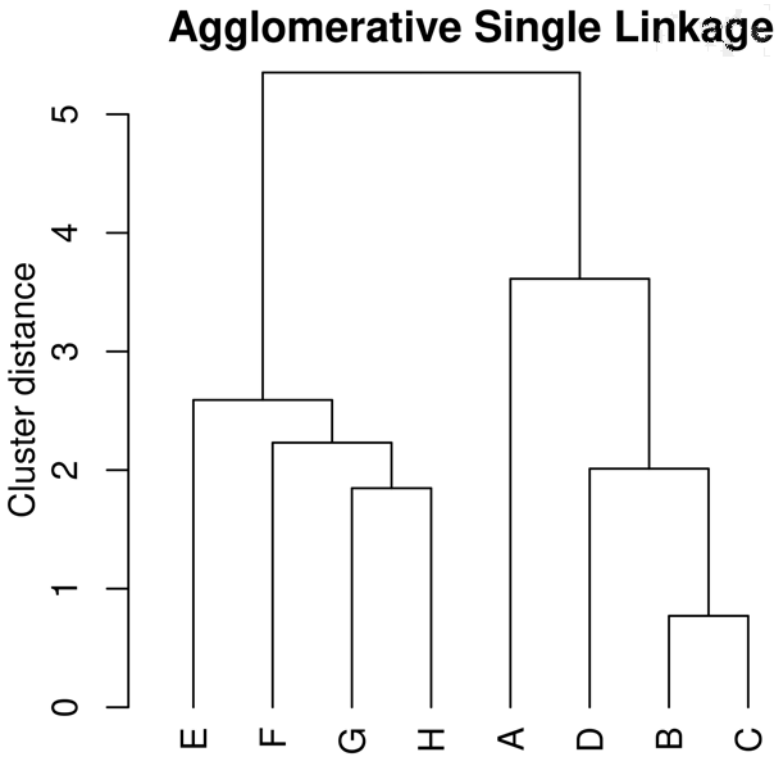
UNSW SYDNEY

Hierarchical Agglomerative Clustering

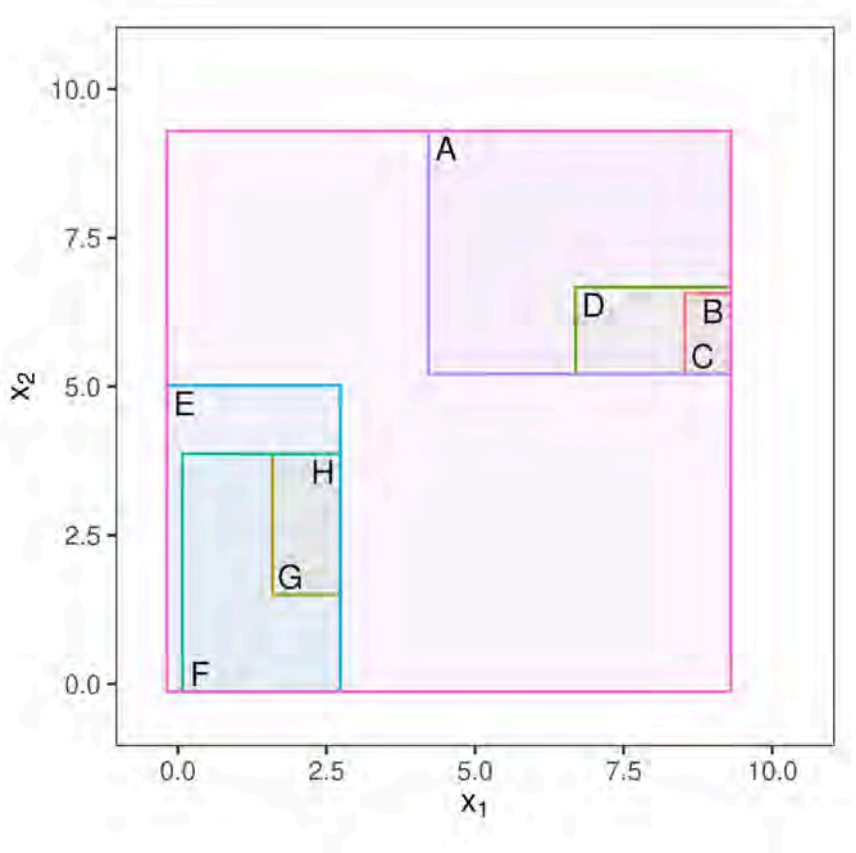


Example of agglomerative clustering (with single linkage).

Hierarchical Clustering: The dendogram

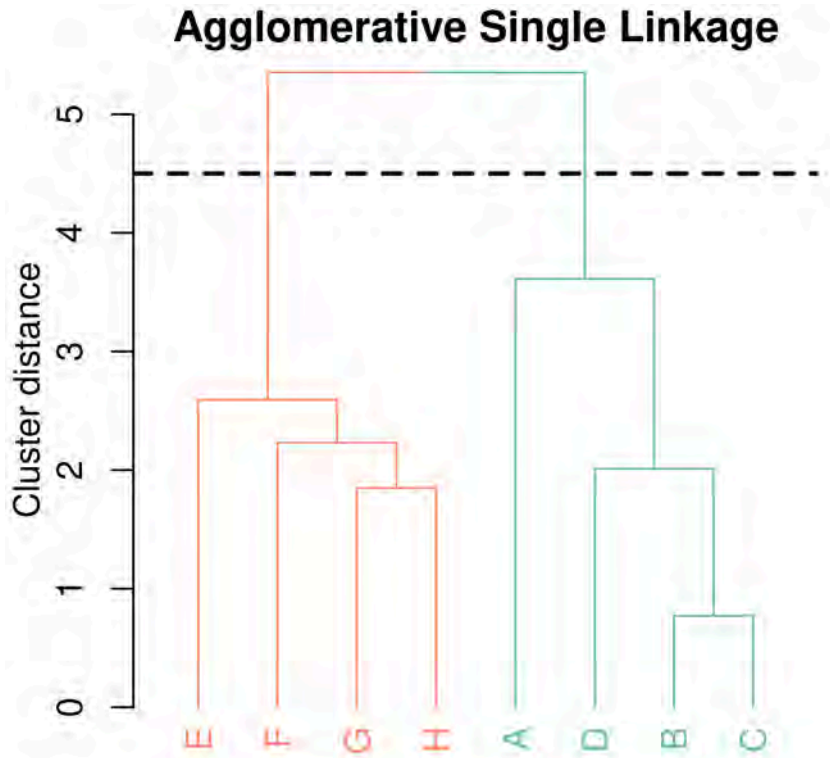


Dendogram

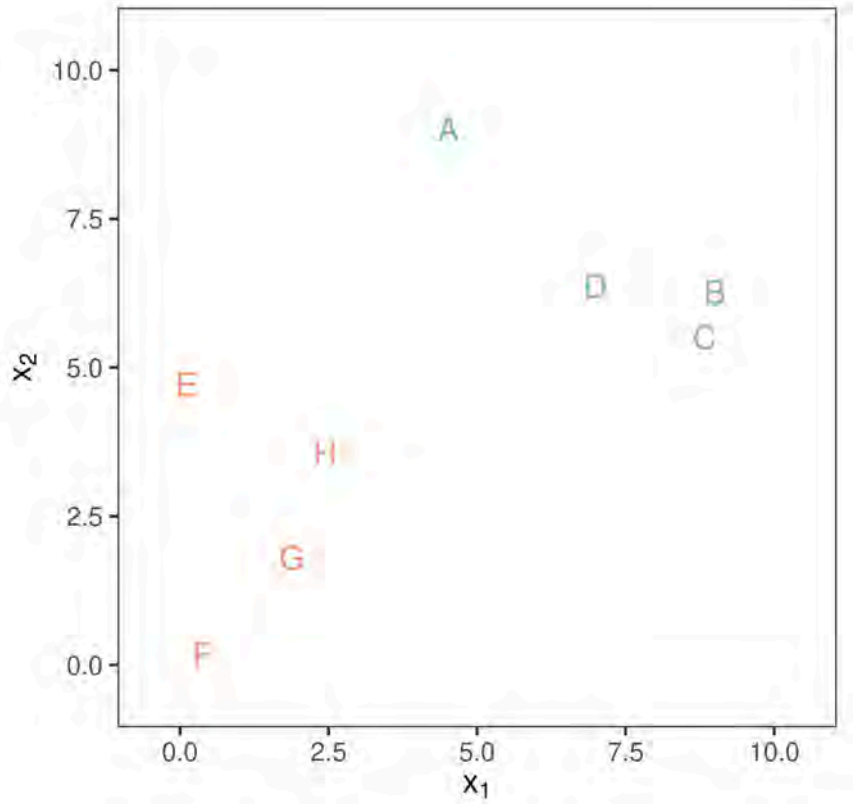


Clusters

Choose a max distance I

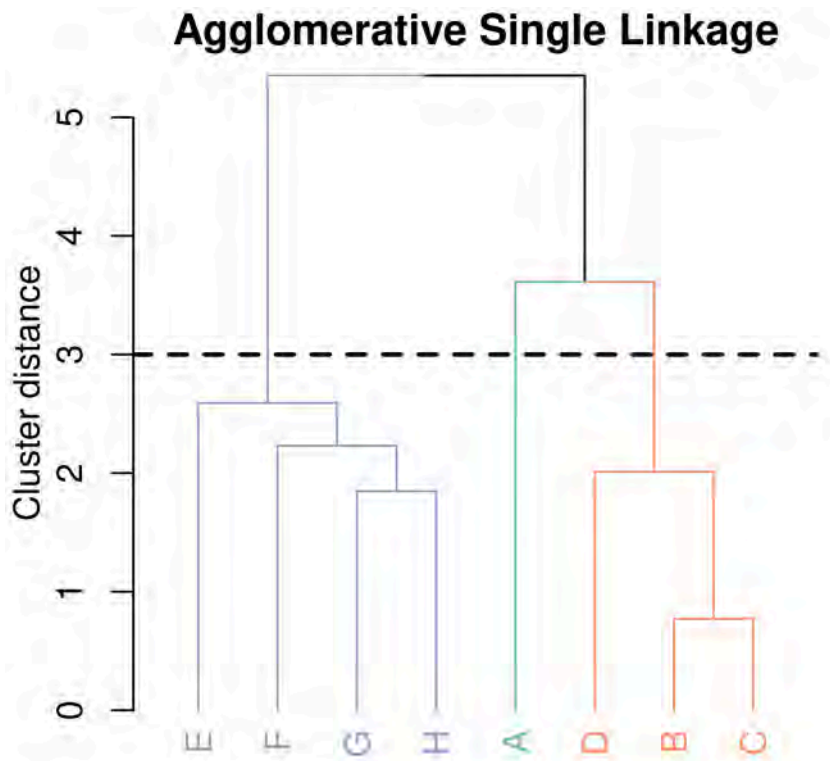


Cut at K = 2

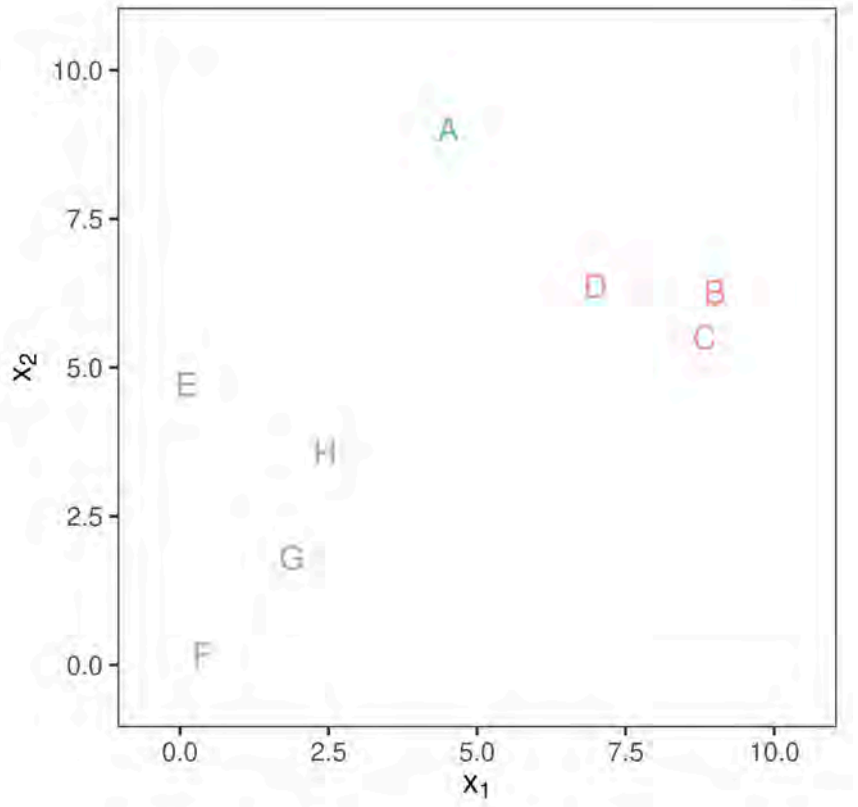


K = 2 clusters

Choose a max distance II

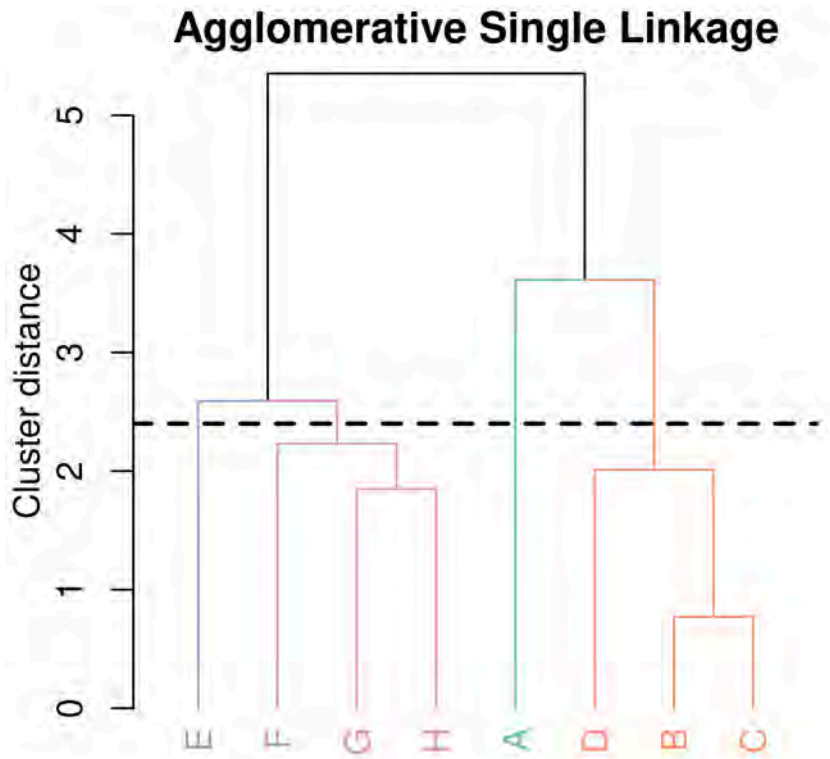


Cut at K = 3

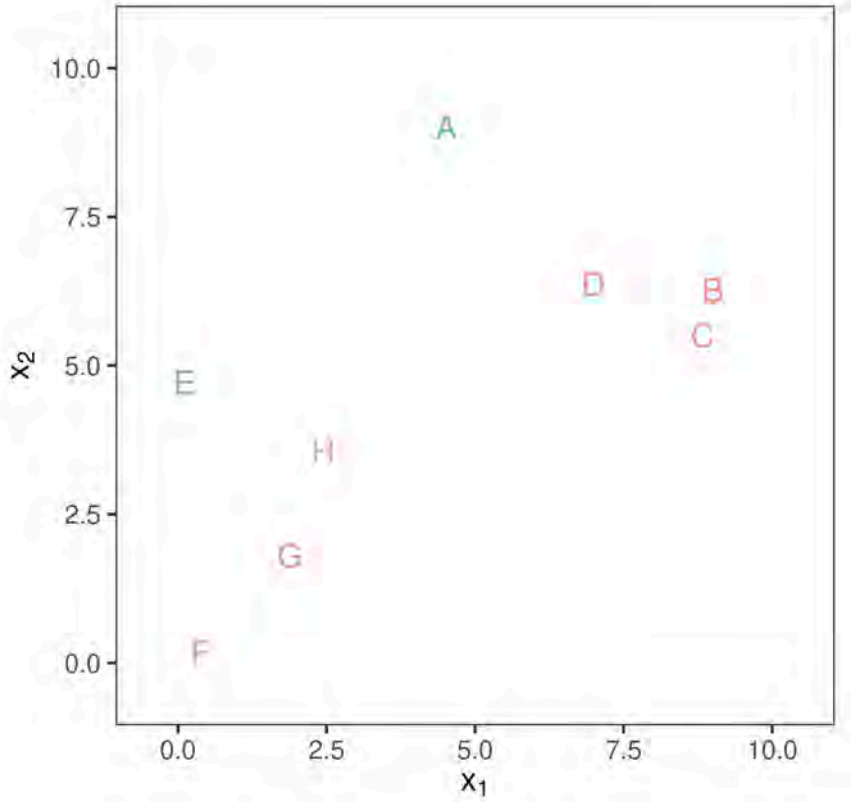


K = 3 clusters

Choose a max distance III



Cut at K = 4



K = 4 clusters

Choice of Dissimilarity Measure

- Euclidean distance

$$\sqrt{\sum_{j=1}^{p}(x_{ij} - x_{i'j})^2}$$

- Simple matching

$$\frac{1}{p}\sum_{j=1}^{p} I(x_{ij} \neq x_{i'j})$$

- Manhattan distance

$$\sum_{j=1}^{p} |x_{ij} - x_{i'j}|$$

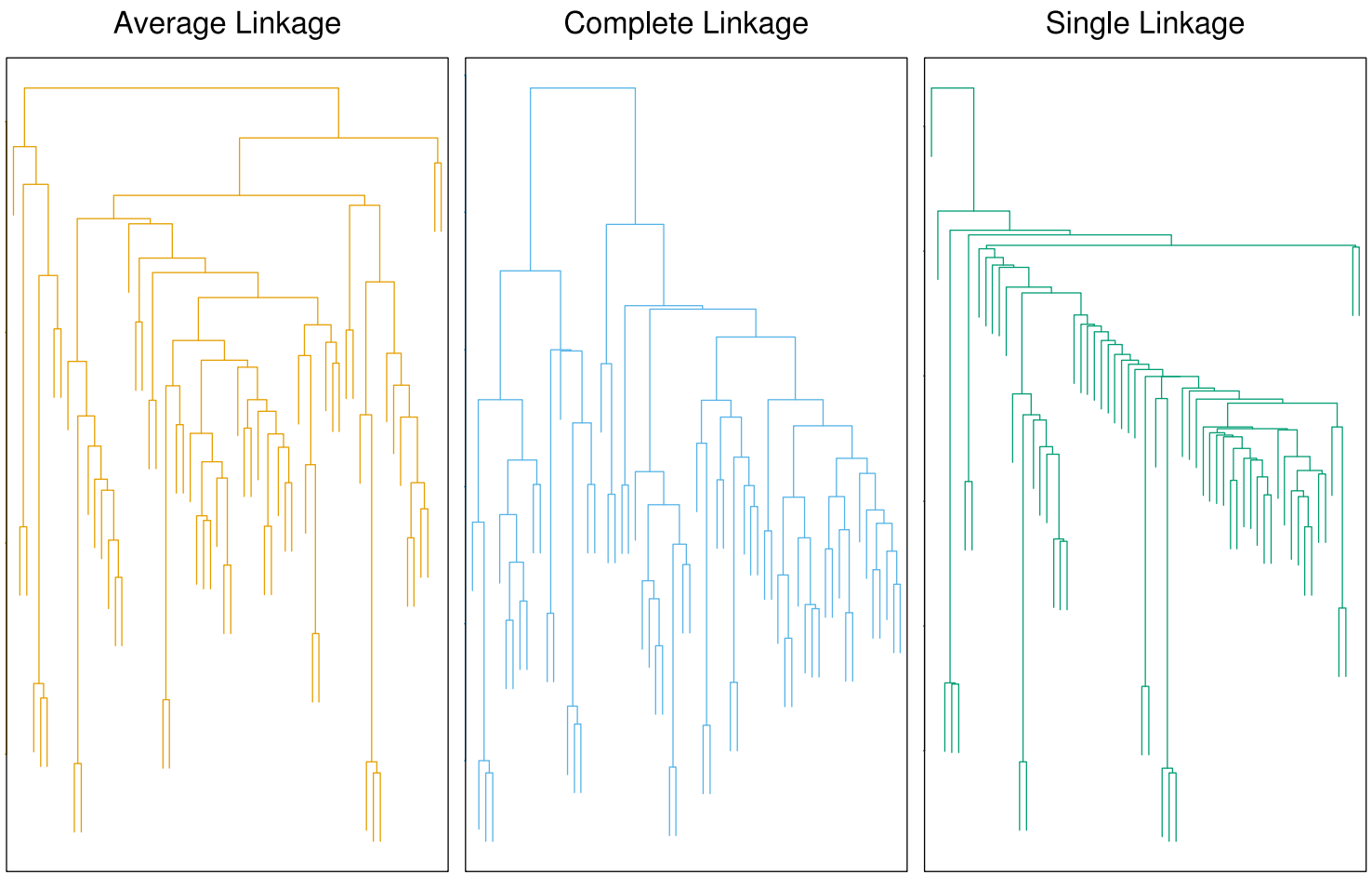
- Combination of numerical and categorical?

Note that we need to consider how to compare groups as well.

Distance between clusters (linkage)

- Complete
  - maximal inter-cluster dissimilarity
  - compute all pairwise dissimilarities between clusters A and B and take **largest**.
- Single
  - minimal inter-cluster dissimilarity
  - compute all pairwise dissimilarities between clusters A and B and take **smallest**.
- Average
  - mean inter-cluster dissimilarity
  - compute all pairwise dissimilarities between clusters A and B and take **average**.
- Centroid
  - dissimilarity between the centroid for cluster A (a mean vector of length p) and the centroid for cluster B
  - an inversion can occur

Same Data, Different Linkage



Average, complete, and single linkage applied to an example data set. Average and complete linkage tend to yield more balanced clusters.

Source: James et al. (2021), An Introduction to Statistical Learning, Figure 12.14.

Practical Issues

- Should the observations / features be standardised in some way?
- Hierarchical clustering
  - dissimilarity measure?
  - type of linkage?
  - where to cut the dendrogram?
- K-means clustering
  - how many clusters?
- Validate the clusters obtained
  - does the clusters represent true subgroups in the data?
- Robustness
  - Don’t rely on one single answer
  - Try different assumptions/data and check consistency of message

Lecture Outline

- Unsupervised Learning
- K-Means Clustering
- Demo: MNIST
- **Hierarchical Clustering**
- Dimension Reduction
- Demo: PCA on MNIST

UNSW SYDNEY

Can you memorise these in 30 secs?

112358132134

248163264128

203048154248

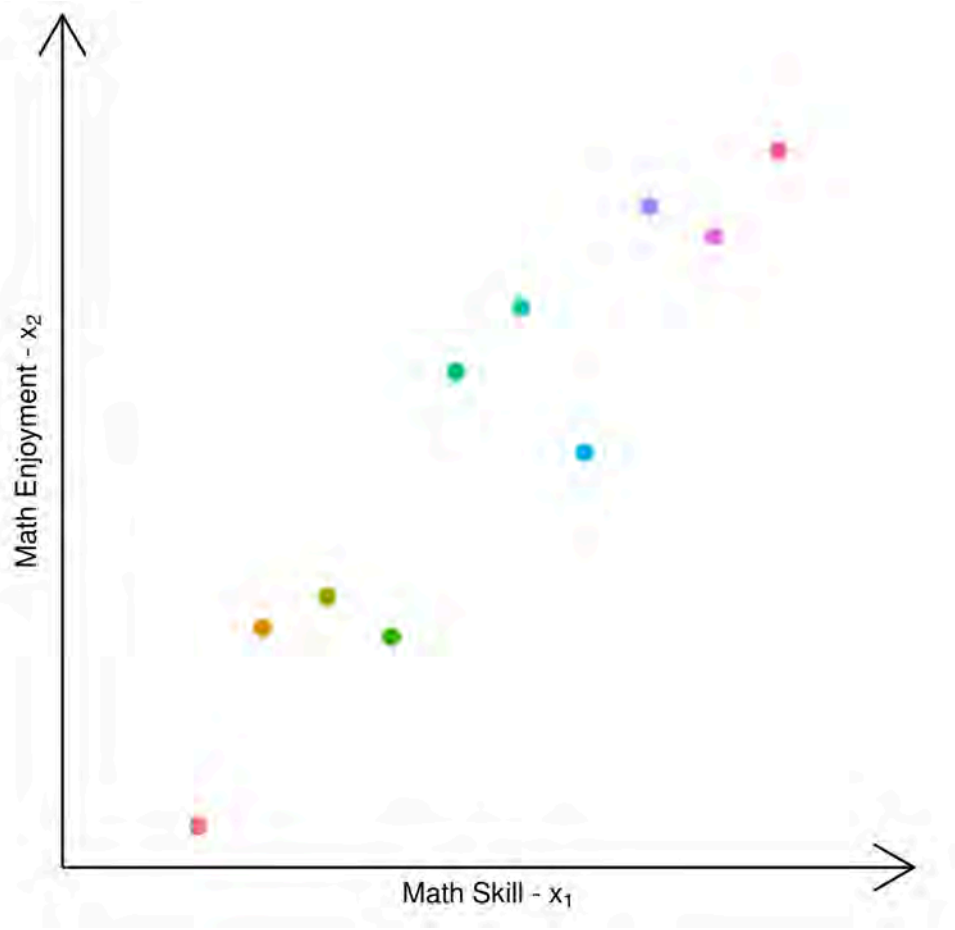
Principal Components Analysis

- Produce derived variables for supervised learning
  - of smaller size than the original data set (i.e. dimension reduction)
  - explain most of the variability in the original set
  - mutually uncorrelated
- A tool for data visualisation

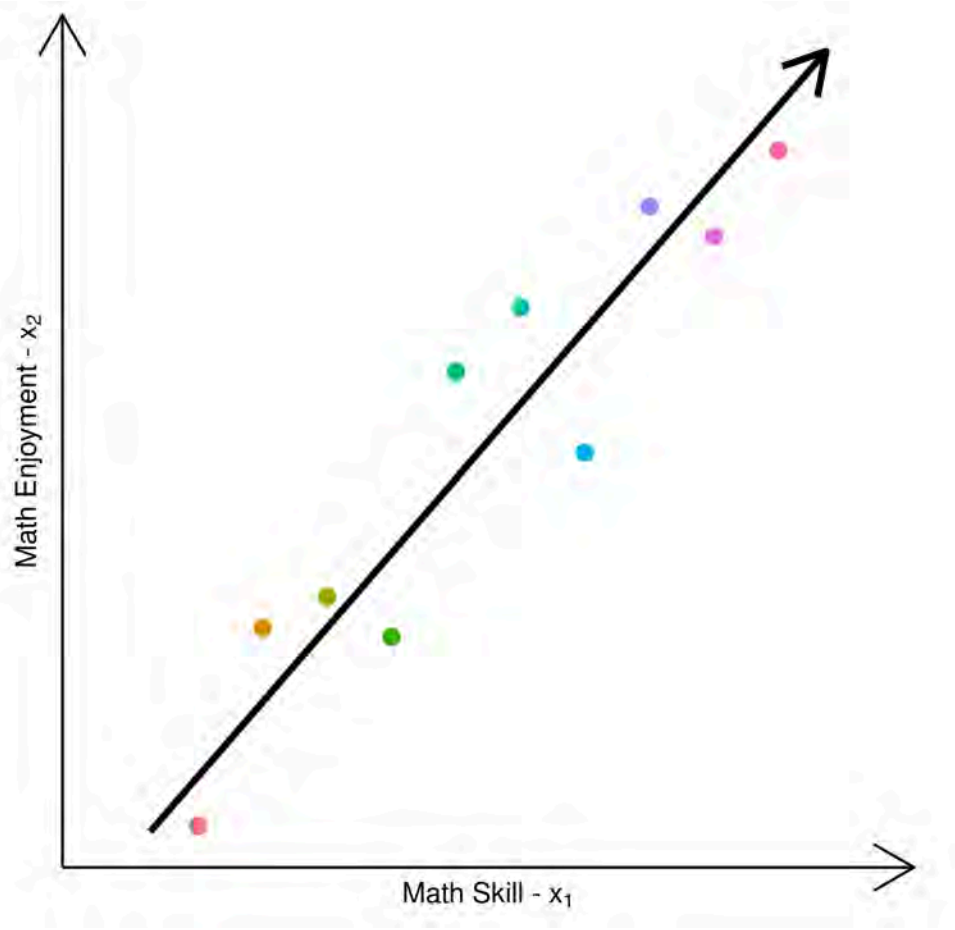
> "… our brains are sort of bad at looking at columns of numbers, but absolutely ace at locating patterns and information in a two-dimensional field of vision" Jordan Ellenberg

Source: Jordan Ellenberg (2014), How Not to Be Wrong: The Power of Mathematical Thinking

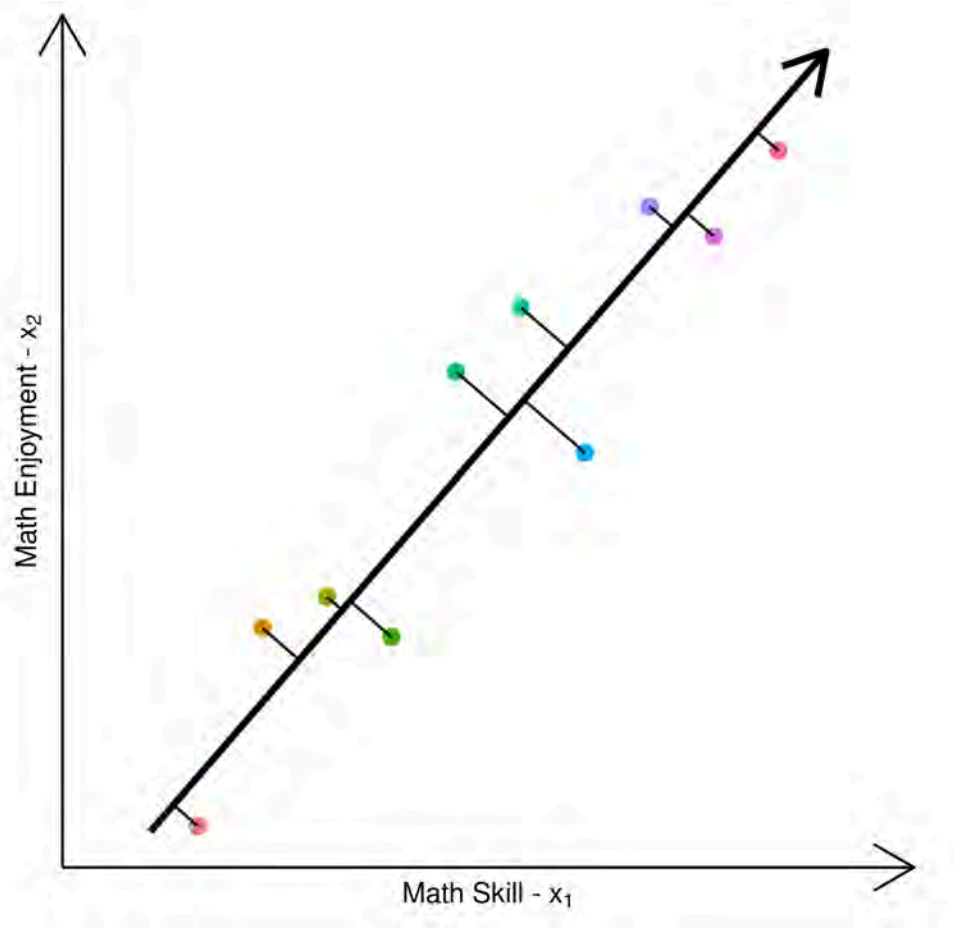
PCA Motivation: Data compression I



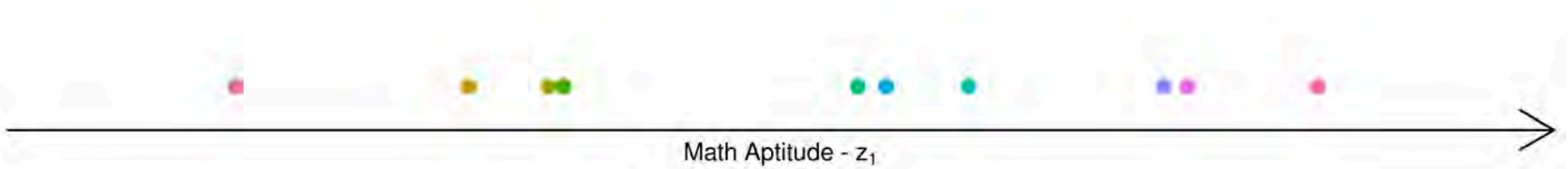
PCA Motivation: Data compression II



PCA Motivation: Data compression III



PCA Motivation: Data compression IV



PCA Motivation: Data compression V

Reduce data from 2D to 1D

$$x^{(1)} \in \mathcal{R}^2 \rightarrow z^{(1)} \in \mathcal{R}$$

$$x^{(2)} \in \mathcal{R}^2 \rightarrow z^{(2)} \in \mathcal{R}$$

$$\vdots$$

$$x^{(n)} \in \mathcal{R}^2 \rightarrow z^{(n)} \in \mathcal{R}$$

Principal Components

The first principal component of a set of features $X_1, X_2, \ldots, X_p$ is the normalised linear combination of the features

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \cdots + \phi_{p1} X_p, \qquad \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

that has the largest variance

$\phi_{11}, \ldots, \phi_{p1}$	loadings of the first principal component
$\phi_1 = (\phi_{11}, \ldots, \phi_{p1})^T$	principal component loading vector
$\sum_{j=1}^{p} \phi_{j1}^2 = 1$	constraint to prevent an arbitrarily large variance

Try out this interactive demo or this demo.

Further Principal Components

The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance and are uncorrelated with Z_1

$$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \cdots + \phi_{p2} x_{ip}, \quad i = 1, 2, \ldots, n$$

$\phi_2 = (\phi_{12}, \ldots, \phi_{p2})^T$ is the second principal component loading vector

Geometry of PCA

- The loading vector ϕ_1 defines a direction in feature space along which the data vary the most
- The projection of the n data points $x_1, \ldots, x_n$ onto this direction are the principal component scores $z_{11}, \ldots, z_{n1}$

Is PCA the same as linear regression? Why or why not?

UNSW SYDNEY

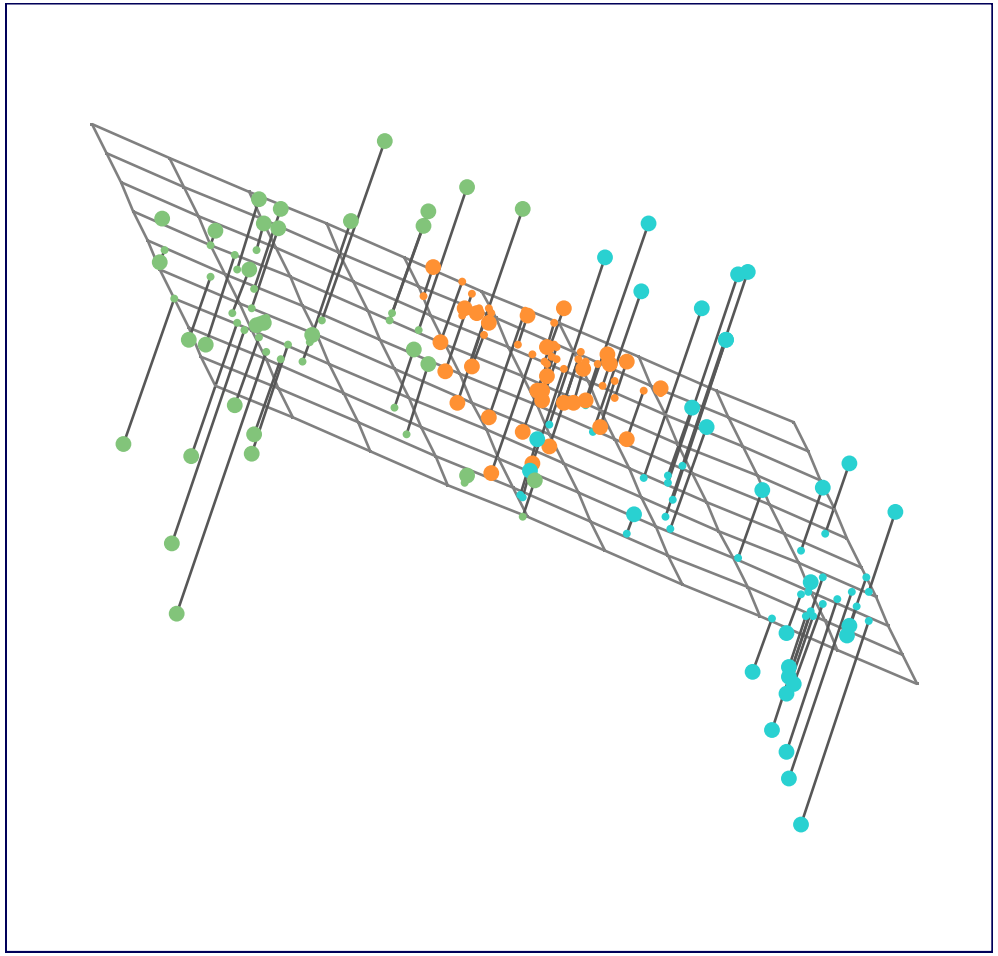
Another Interpretation of PC

- The first principal component loading vector
  - the line in p-dimensional space that is closest to the n observations
- Extends beyond the first principal component
  - the first two principal components of a data set span the plane that is closest to the n observations
  - the first three principal components of a data set span the hyperplane that is closest to the n observations
  - and so forth

In 3 dimensions, first two PCs:

- Plane spans the first two principal component directions.
- Minimises the sum of square distances from each point to the plane.

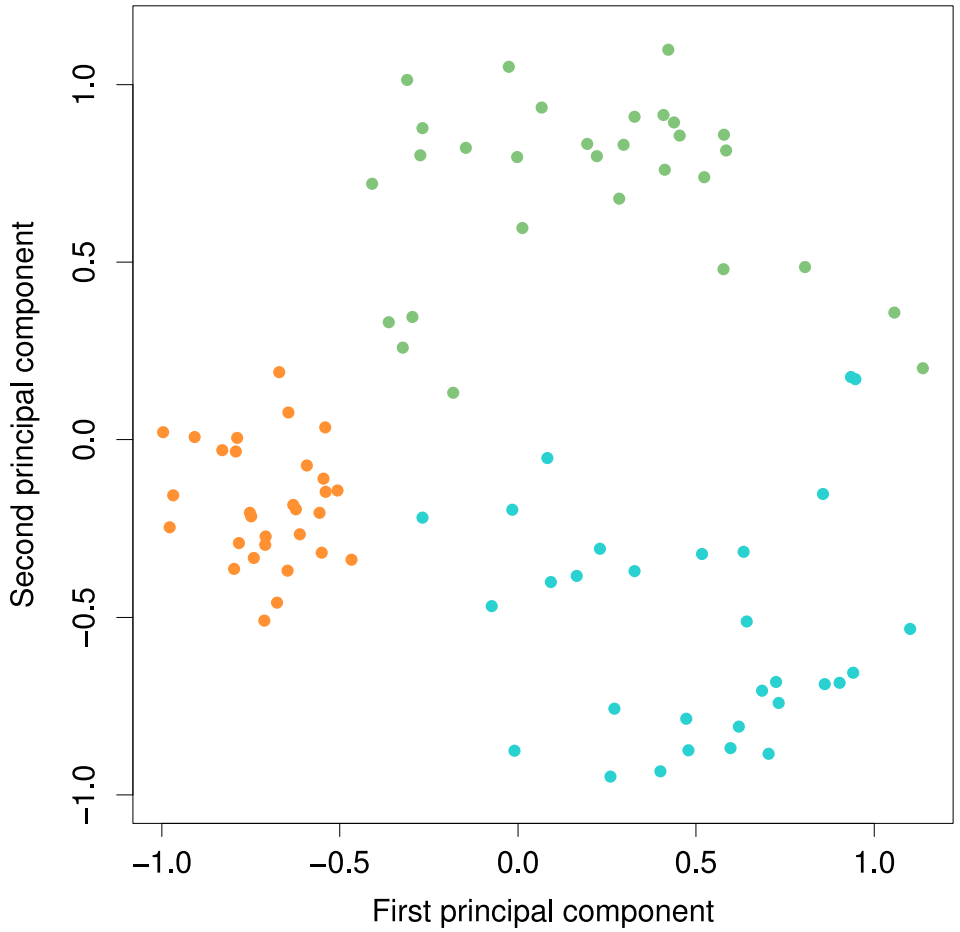
2 principal component directions I



Ninety observations simulated in three dimensions. The observations are displayed in color for ease of visualization. The first two principal component directions span the plane that best fits the data. The plane is positioned to minimize the sum of squared distances to each point.

Source: James et al. (2021), An Introduction to Statistical Learning, Figure 12.2a.

2 principal component directions II



The first two principal component score vectors give the coordinates of the projection of the 90 observations onto the plane.

Source: James et al. (2021), An Introduction to Statistical Learning, Figure 12.2b.

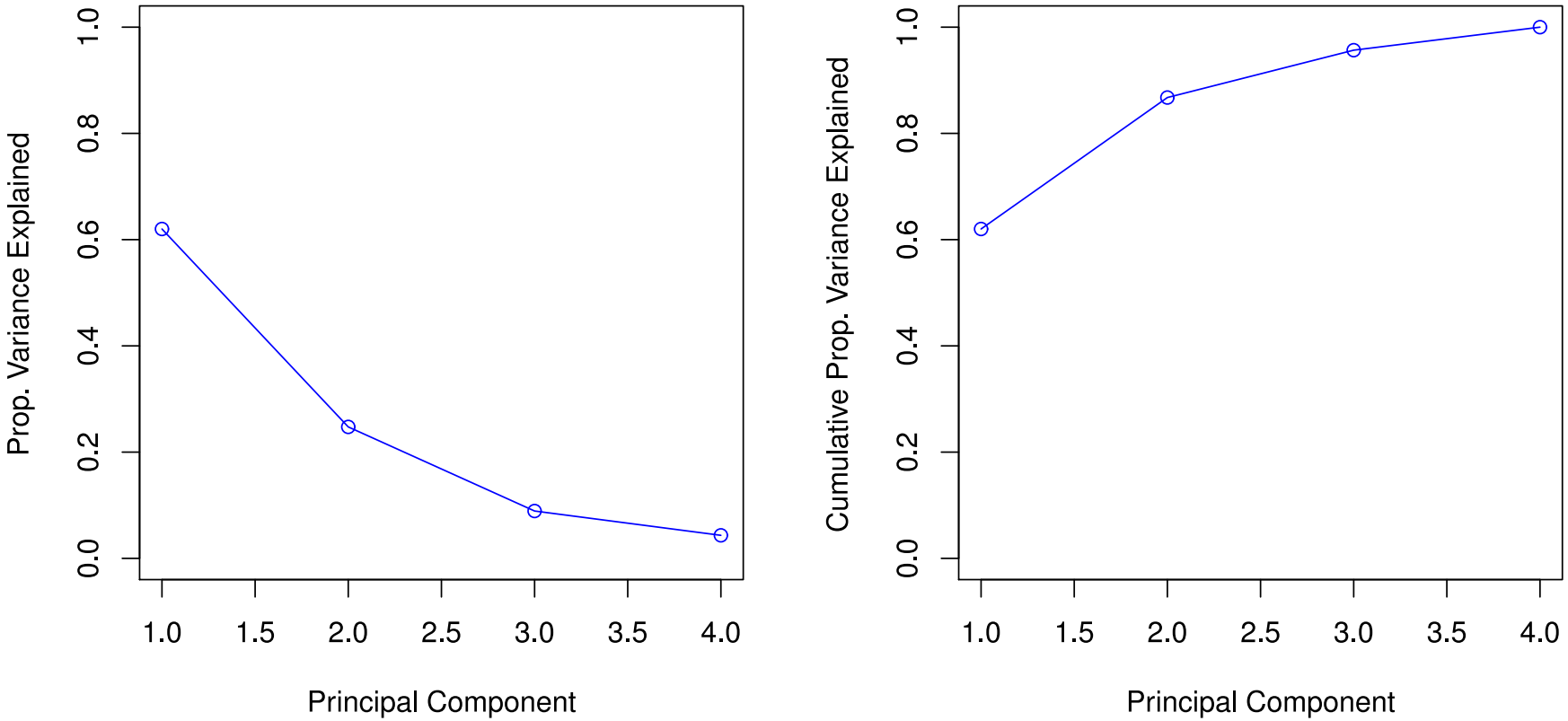
More on PCA

- Scaling the variables
  - typically scale each variable to have standard deviation one before performing PCA
  - may not be necessary if variables are measured in the same units
- Uniqueness of the principal components
  - each principal component loading vector is unique up to a sign flip
- The proportion of variance explained (PVE)
  - the PVE of the mth principal component is given by

$$\frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{jm}x_{ij}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2}$$

UNSW SYDNEY

How many principal components to use?



Left: a scree plot depicting the proportion of variance explained by each of the four principal components in the USArrests data. Right: the cumulative proportion of variance explained by the four principal components in the USArrests data.

Source: James et al. (2021), An Introduction to Statistical Learning, Figure 12.3.

Lecture Outline

- **Unsupervised Learning**
- K-Means Clustering
- Demo: MNIST
- Hierarchical Clustering
- Dimension Reduction
- Demo: PCA on MNIST

PCA on MNIST (failed attempt)

```r
pr_comp_train <- prcomp(x_train, scale = TRUE)
```

```
Error in prcomp.default(x_train, scale = TRUE): cannot rescale a constant/zero column to unit variance
```

R | Python

```r
# Calculate the column std devs
col_std_devs <- apply(x_train, 2, sd)
col_std_devs
```

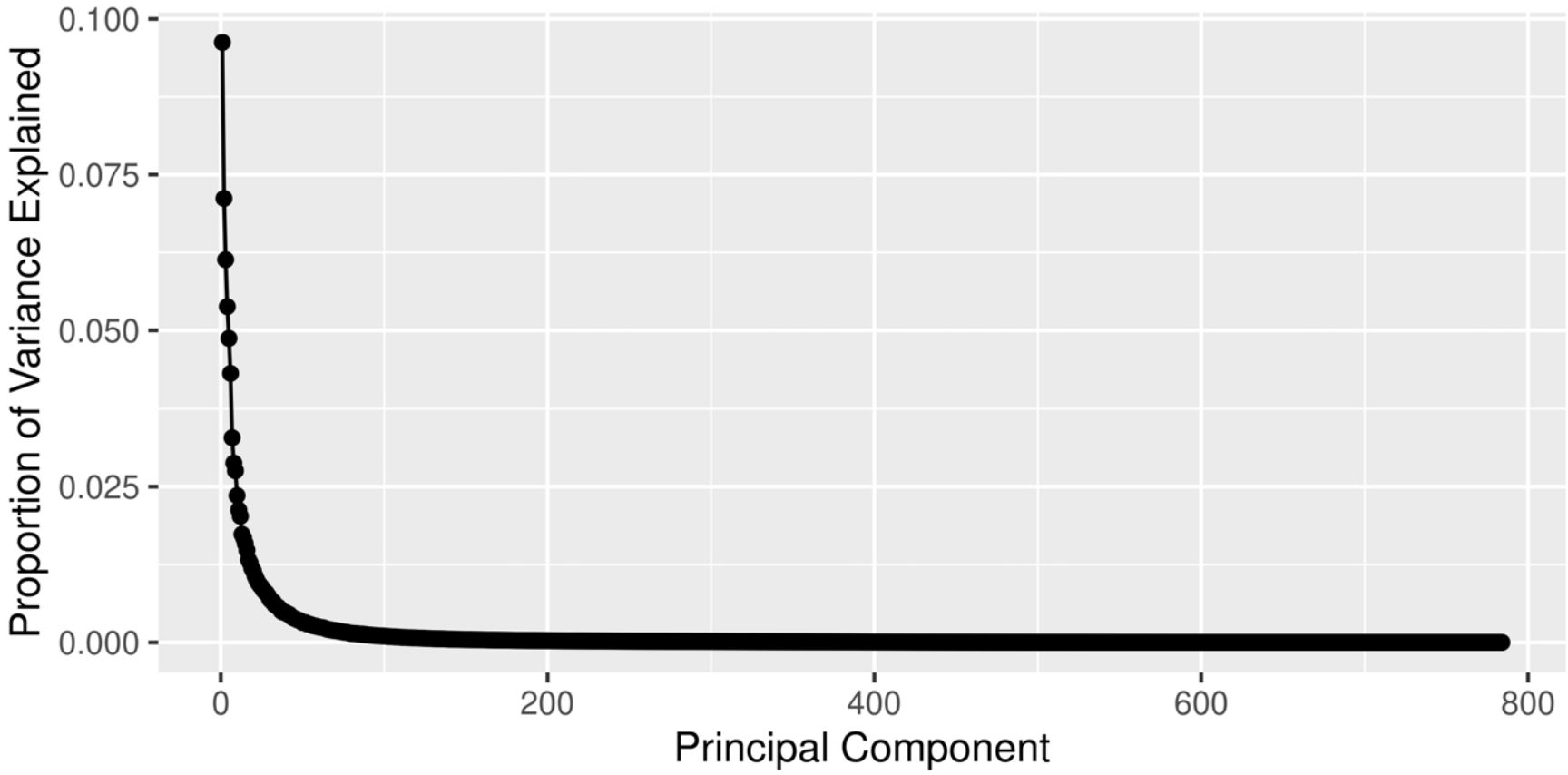
```
          X0           X1           X2           X3           X4           X5
0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
          X6           X7           X8           X9          X10          X11
0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
         X12          X13          X14          X15          X16          X17
0.0023975438 0.0052497943 0.0000000000 0.0000000000 0.0000000000 0.0000000000
         X18          X19          X20          X21          X22          X23
0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
         X24          X25          X26          X27          X28          X29
0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
         X30          X31          X32          X33          X34          X35
0.0000000000 0.0000000000 0.0000000000 0.0001653479 0.0036899614 0.0073450035
         X36          X37          X38          X39          X40          X41
0.0145456588 0.0161569550 0.0211853119 0.0213073459 0.0208507875 0.0195538213
         X42          X43          X44          X45          X46          X47
0.0229472506 0.0243856500 0.0230354157 0.0211948495 0.0151041369 0.0109502265
         X48          X49          X50          X51          X52          X53
0.0111429515 0.0074043871 0.0041586589 0.0040716909 0.0000000000 0.0000000000
         X54          X55          X56          X57          X58          X59
0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0013227829 0.0005993860
         X60          X61          X62          X63          X64          X65
0.0015278703 0.0021872664 0.0112660837 0.0184871514 0.0260346078 0.0377863192
```

UNSW SYDNEY

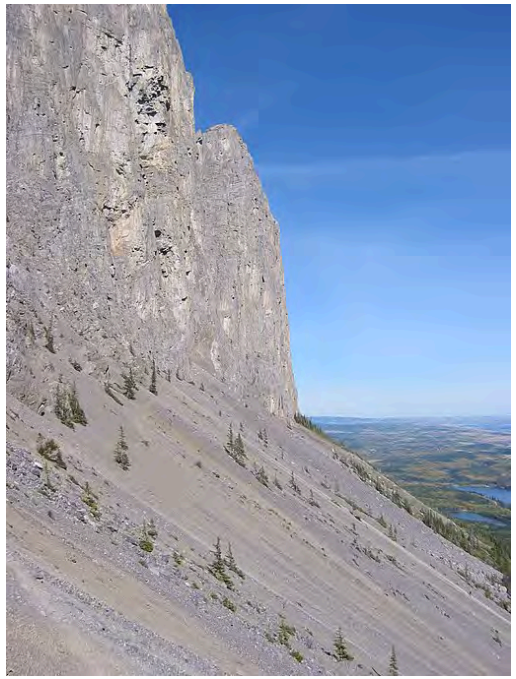
PCA on MNIST

R | Python

```r
pr_comp_train <- prcomp(x_train)
pve <- pr_comp_train$sdev^2 / sum(pr_comp_train$sdev^2)
```



Scree plot



A scree

A scree plot is a line plot of the eigenvalues of principal components on the y-axis and the factors on the x-axis.

The scree plot is used to determine the number of factors to retain in a principal components analysis.

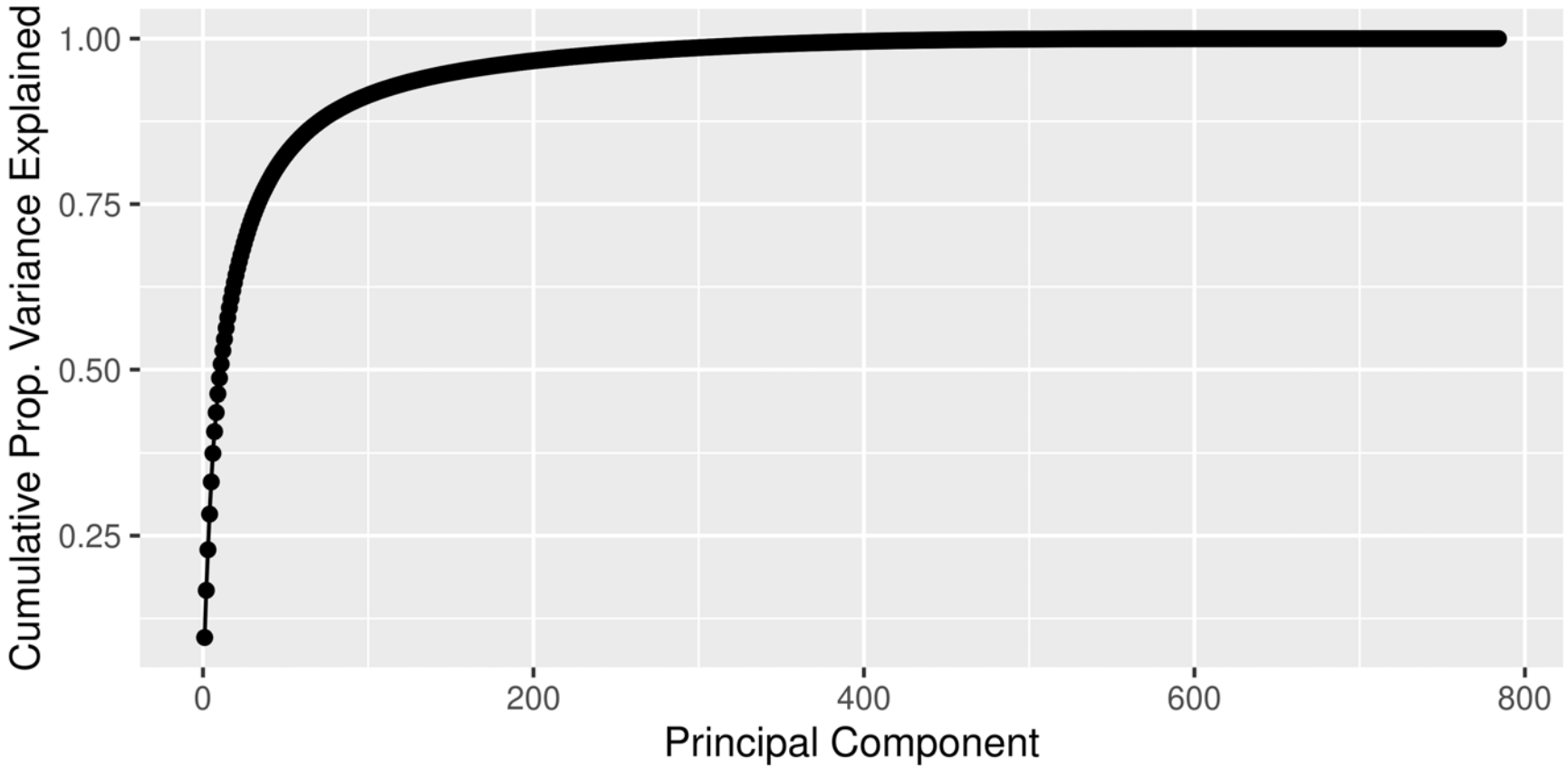
The point at which the line starts to level off is the number of factors to retain.

Source: Wikimedia Commons.

PCA on MNIST: Cumulative

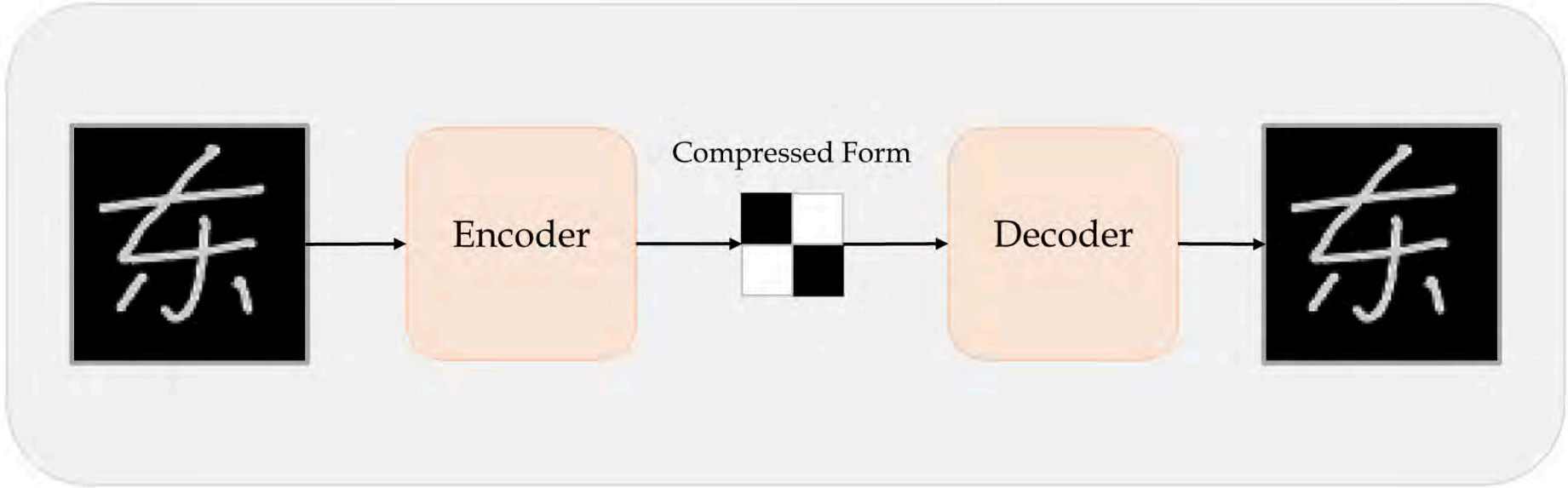
R | Python

```r
cve <- cumsum(pve)
```



Autoencoder

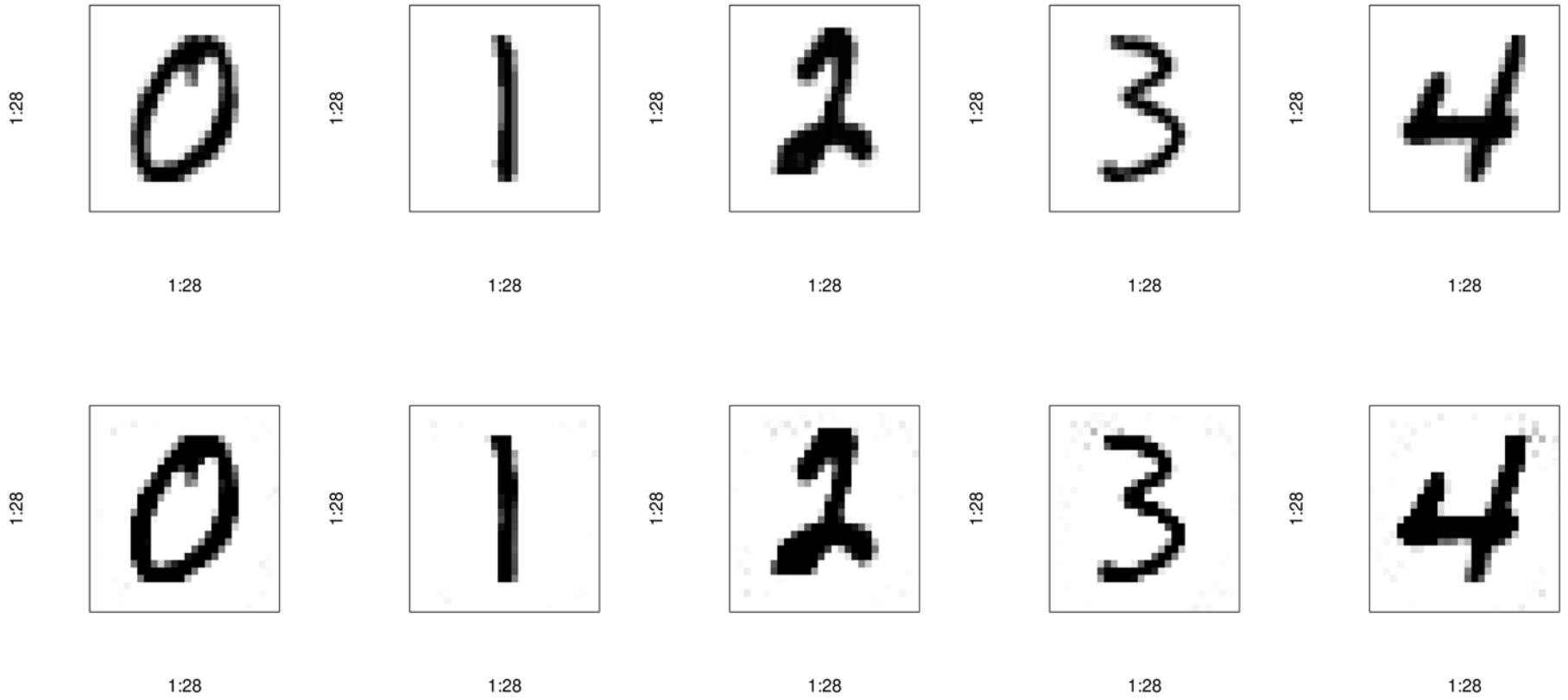
An autoencoder takes an observation, maps it to a latent space via an encoder module, then decodes it back to an output with the same dimensions via a decoder module.



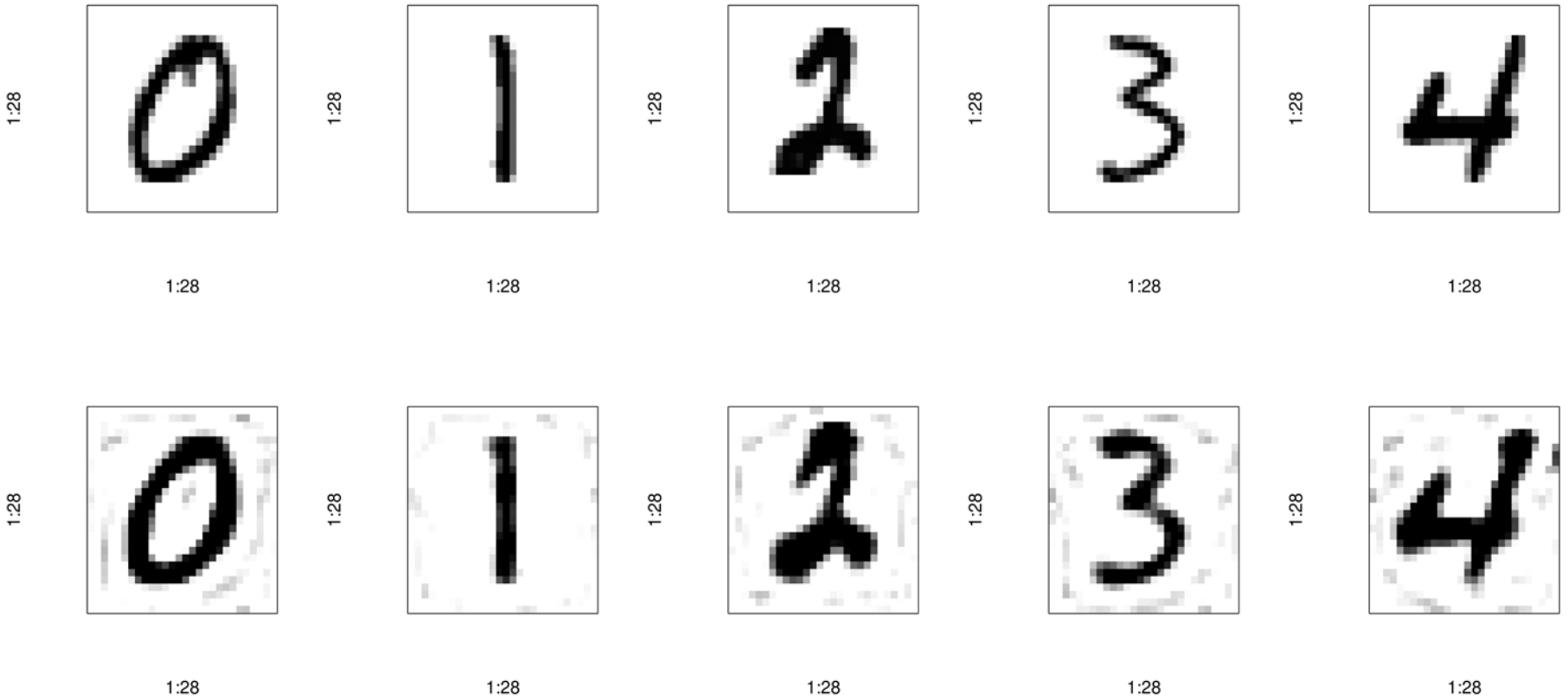
Schematic of an autoencoder.

Source: Marcus Lautier (2022).

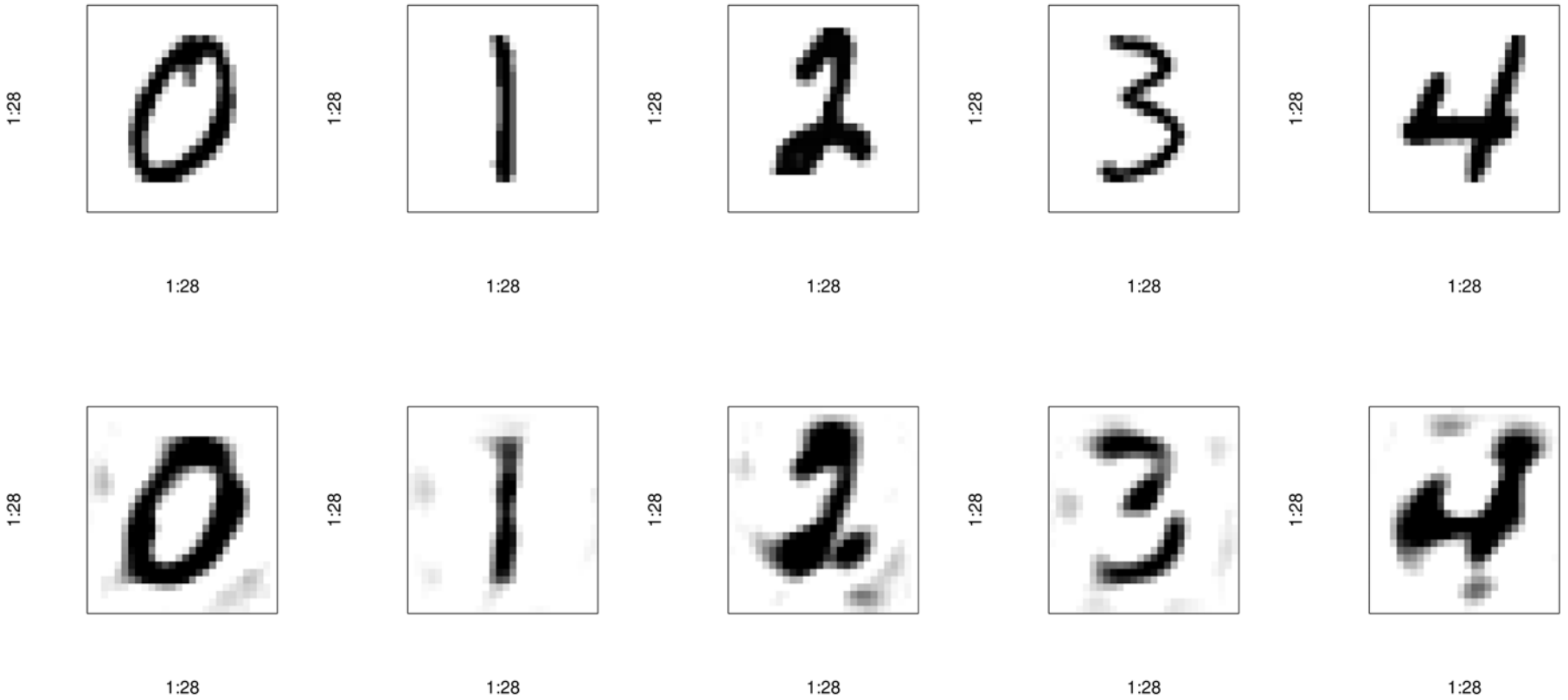
PCA on MNIST: Reconstructed with 400 PCs



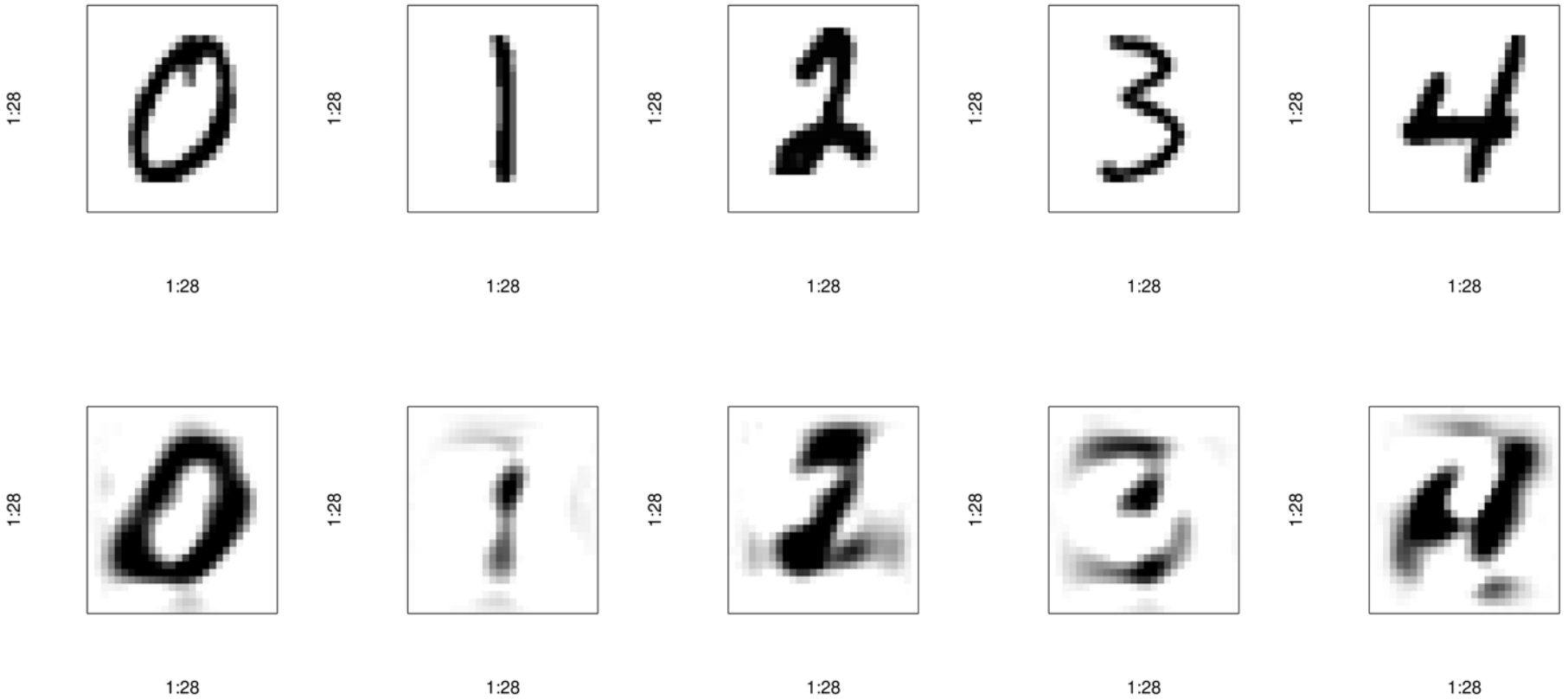
PCA on MNIST: Reconstructed with 100 PCs



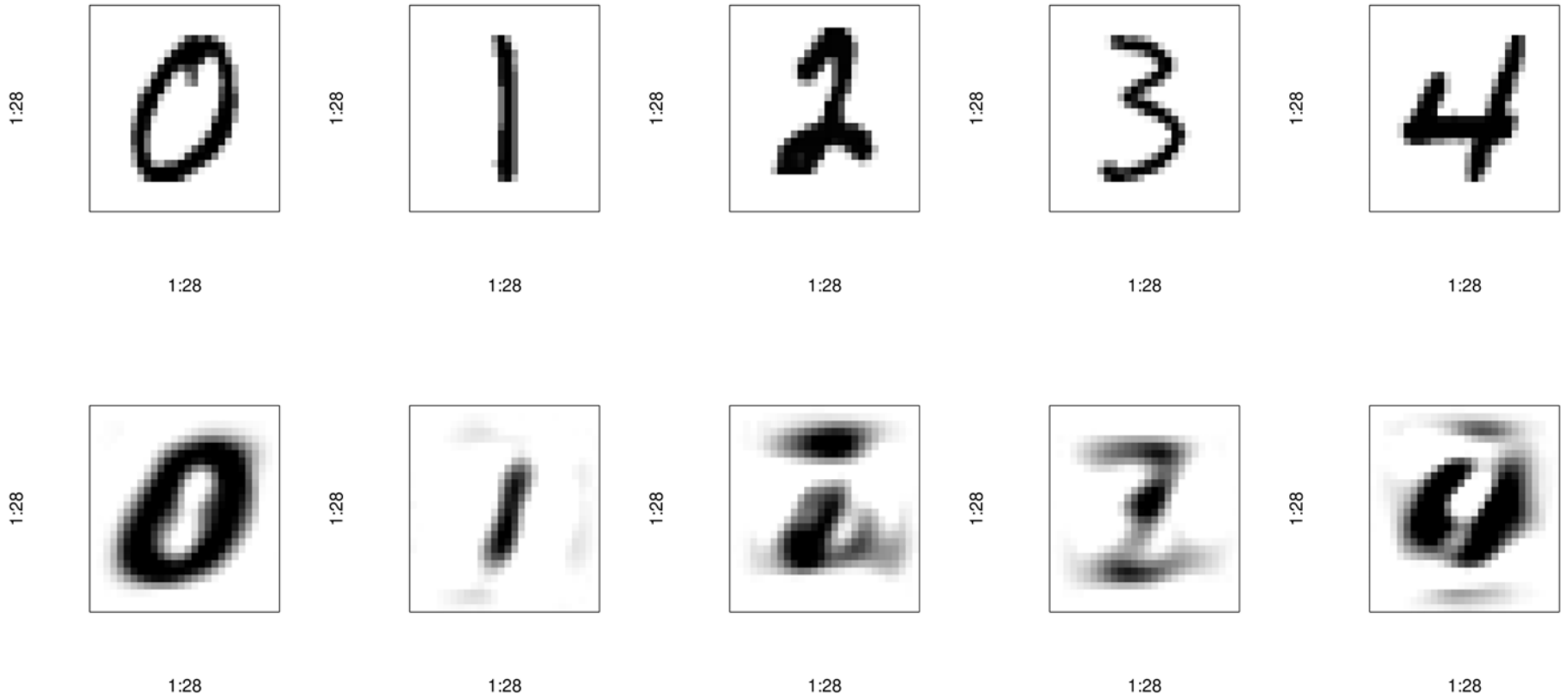
PCA on MNIST: Reconstructed with 25 PCs



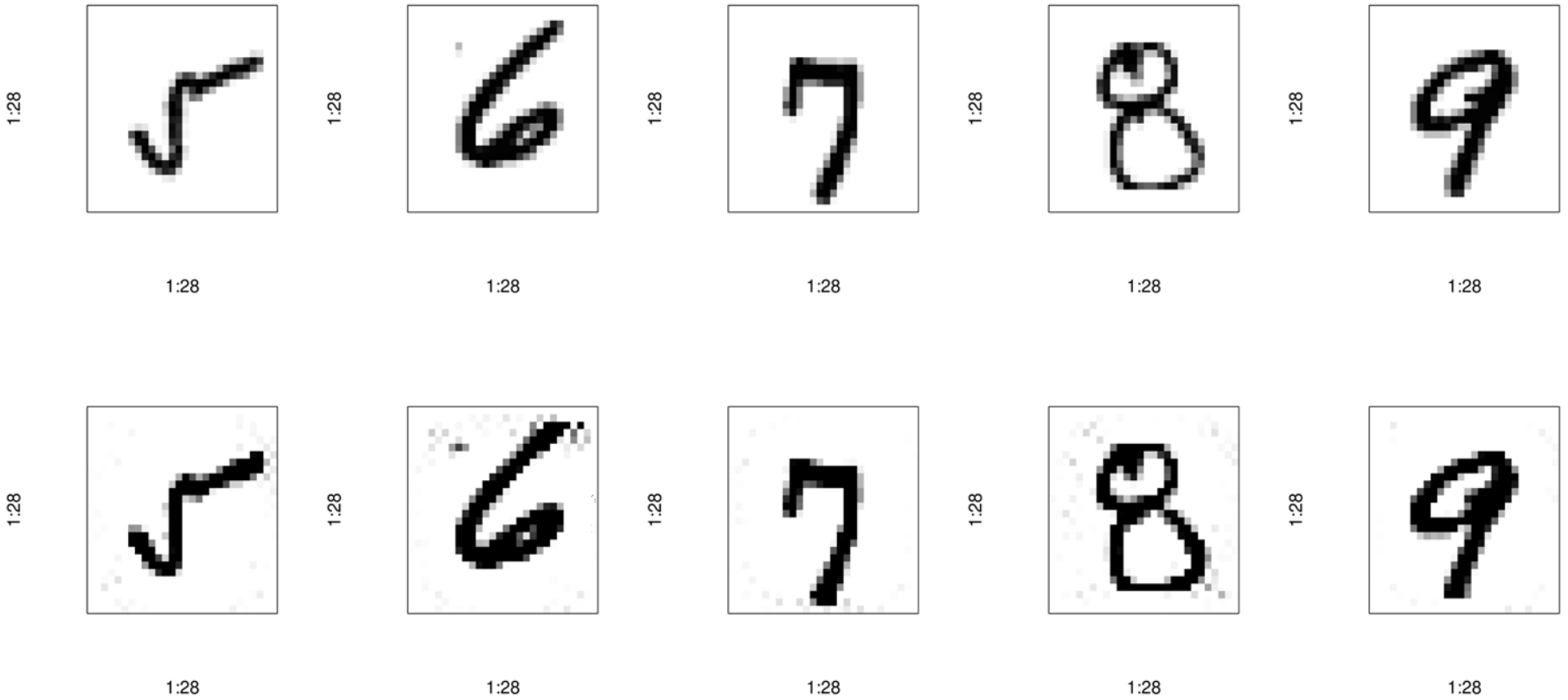
PCA on MNIST: Reconstructed with 10 PCs



PCA on MNIST: Reconstructed with 5 PCs

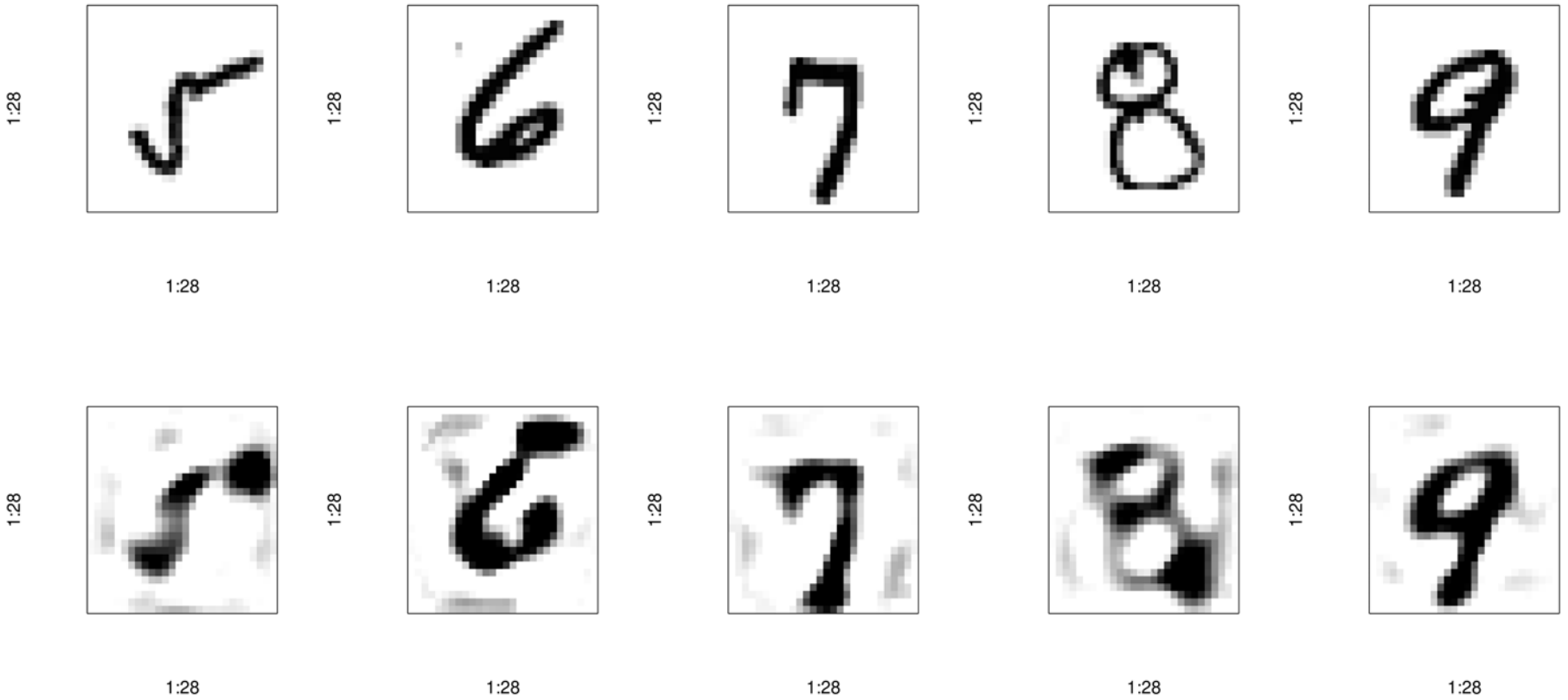


PCA on MNIST: Reconstructed with 400 PCs II

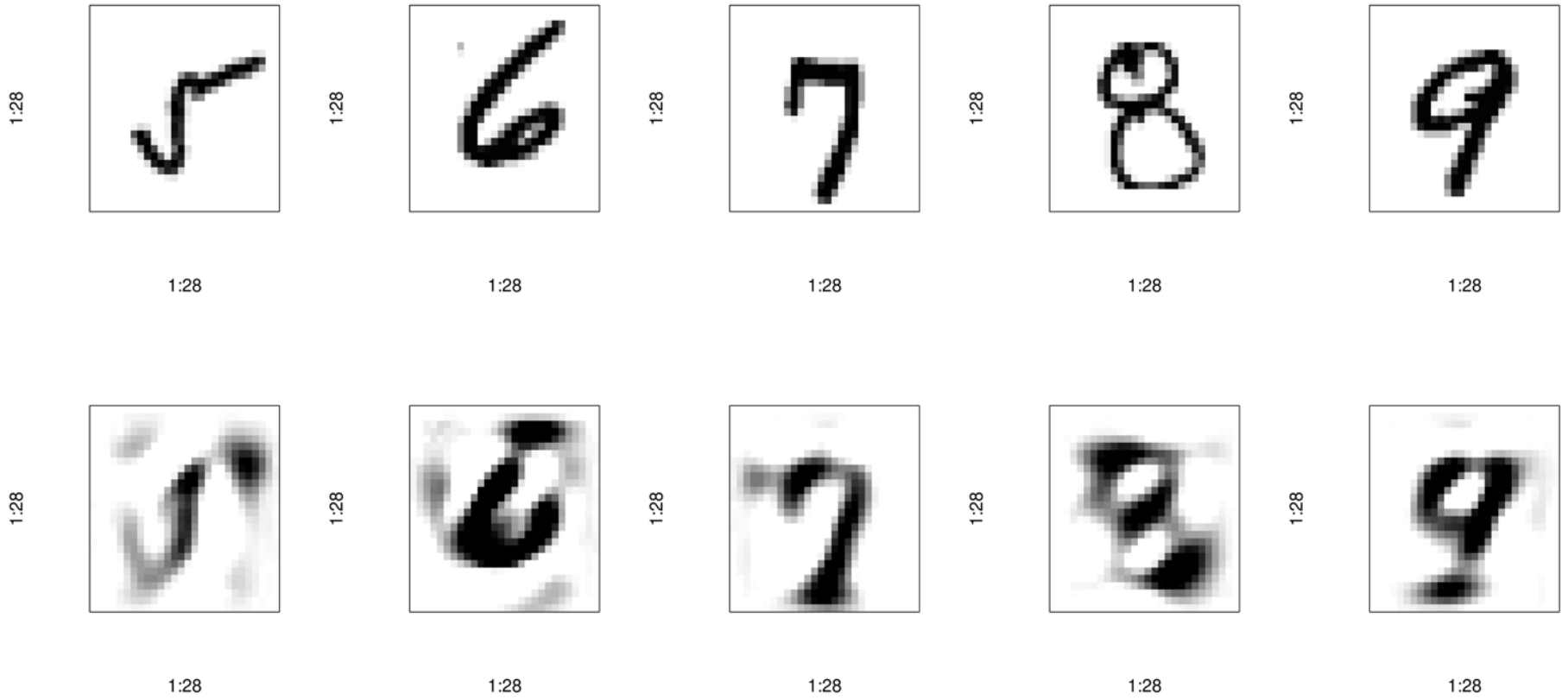


PCA on MNIST: Reconstructed with 100 PCs II

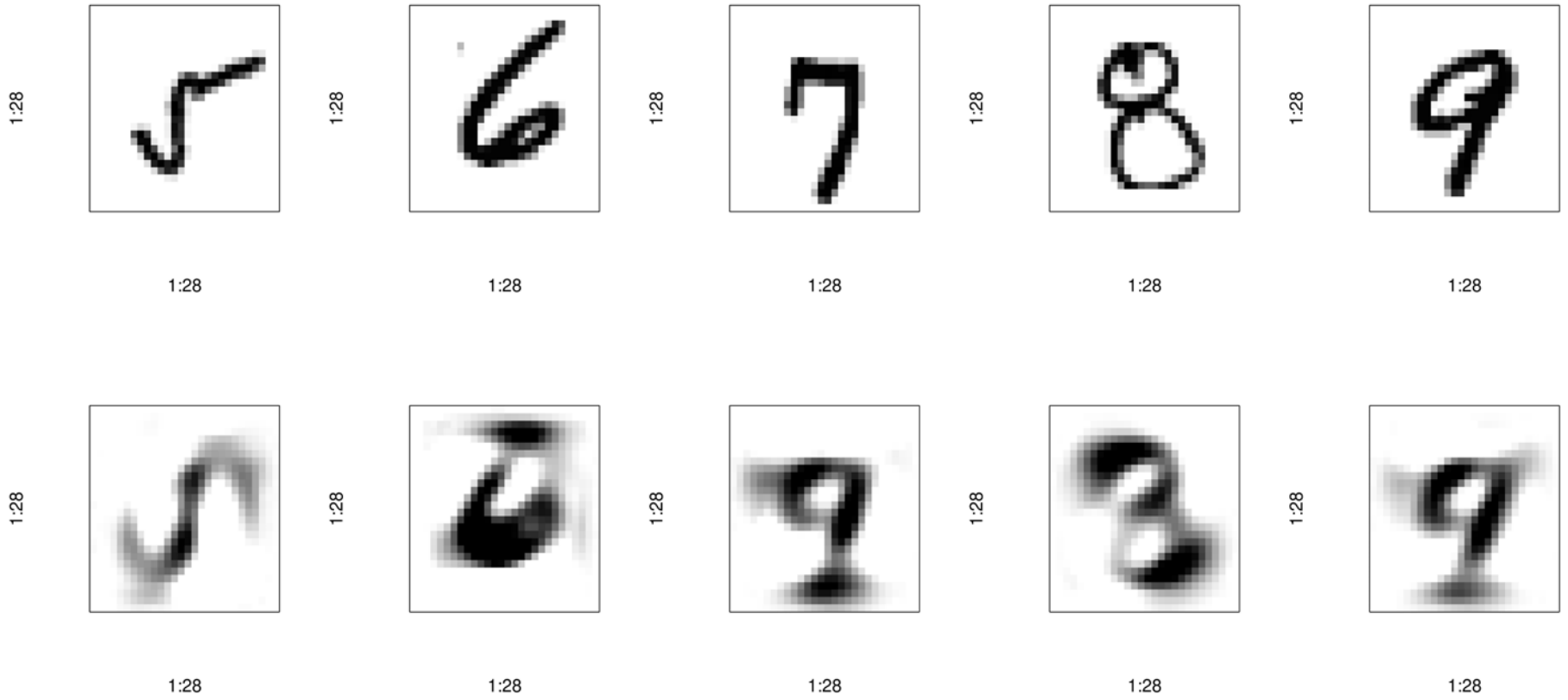
PCA on MNIST: Reconstructed with 25 PCs II



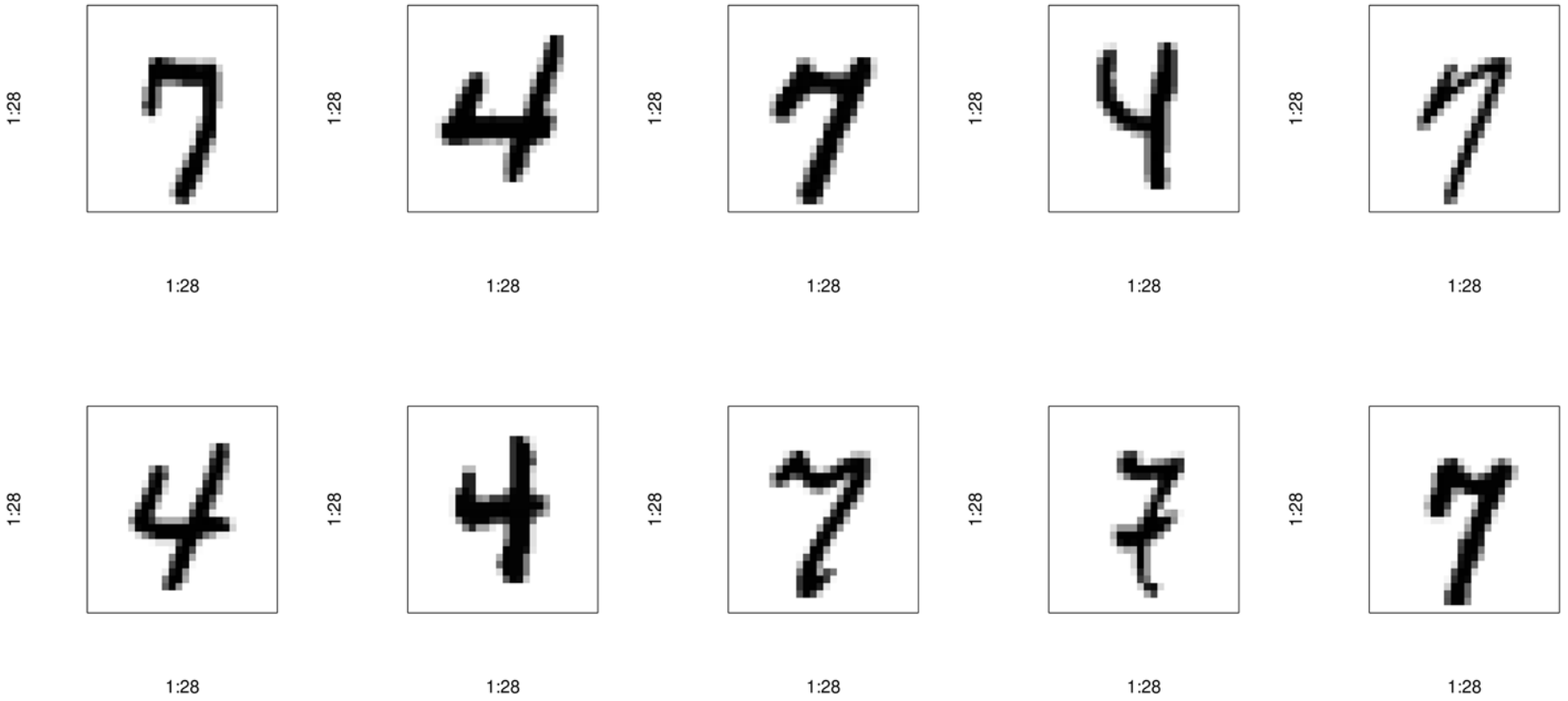
PCA on MNIST: Reconstructed with 10 PCs II



PCA on MNIST: Reconstructed with 5 PCs II



Just pull out the 4s and the 7s



Logistic regression on 4 vs 7

R | Python

```r
x_train_filtered <- as.data.frame(x_train[, col_std_devs > 0])
x_val_filtered <- as.data.frame(x_val[, col_std_devs > 0])
x_test_filtered <- as.data.frame(x_test[, col_std_devs > 0])

logistic_model_varying <- glm(y_train ~ ., data=x_train_filtered, family = binomial)
nrow(summary(logistic_model_varying)$coefficients)
```

```
[1] 629
```

UNSW SYDNEY

Logistic regression on first 50 PCs

R | Python

```r
pca_train <- as.data.frame(pr_comp_train$x[, 1:50])
logistic_model_pca <- glm(y_train ~ ., data=pca_train, family = binomial)
summary(logistic_model_pca)$coefficients
```

```
               Estimate Std. Error      z value     Pr(>|z|)
(Intercept)  1.198515947 0.21135576   5.67060924 1.422906e-08
PC1          3.647866454 0.19481760  18.72452246 3.124608e-78
PC2          2.851327556 0.17512018  16.28211881 1.322109e-59
PC3          0.677184850 0.09516631   7.11580437 1.112620e-12
PC4         -1.637208408 0.12421894 -13.18002196 1.143575e-39
PC5         -0.077537401 0.12214106  -0.63481847 5.255468e-01
PC6          0.502354456 0.11548195   4.35006906 1.360947e-05
PC7         -0.792560391 0.11826262  -6.70169838 2.060109e-11
PC8         -0.351994691 0.11458539  -3.07189847 2.127021e-03
PC9          0.075791383 0.12492815   0.60667978 5.440634e-01
PC10         0.008328826 0.14043146   0.05930884 9.527061e-01
PC11         0.128684822 0.14642117   0.87886762 3.794731e-01
PC12        -0.717080199 0.15831254  -4.52952237 5.911718e-06
PC13         0.278289184 0.16411883   1.69565667 8.995092e-02
PC14        -0.326632889 0.13879100  -2.35341547 1.860184e-02
PC15         1.260314977 0.22677140   5.55764509 2.734387e-08
PC16         0.517611645 0.19579969   2.64357748 8.203499e-03
PC17         0.203745210 0.17867643   1.14030267 2.541602e-01
PC18        -0.462318412 0.19603068  -2.35839822 1.835399e-02
PC19         1.687111767 0.21417667   7.87719665 3.348078e-15
PC20         0.464271371 0.19228580   2.41448599 1.575743e-02
```

UNSW SYDNEY

Compare models on validation accuracy

R | Python

```r
# Perform PCA on the validation set using the same rotation from the training set
pca_val <- as.data.frame(predict(pr_comp_train, newdata = x_val)[, 1:50])

# Calculate accuracy on validation data
y_pred <- predict(logistic_model_varying, x_val_filtered, type = "response") > 0.5
accuracy_varying <- mean(y_pred == y_val)
y_pred <- predict(logistic_model_pca, pca_val, type = "response") > 0.5
accuracy_pca <- mean(y_pred == y_val)

c(accuracy_varying, accuracy_pca)
```

```
[1] 0.9641089 0.9847360
```

Compression

> "A photograph, which used to be a pattern of pigment on a sheet of chemically coated paper, is now a string of numbers, each one representing the brightness and color of a pixel. An image captured on a 4-megapixel camera is a list of 4 million numbers-no small commitment of memory for the device shooting the picture. But these numbers are highly correlated with each other. If one pixel is bright green, the next one over is likely to be as well. The actual information contained in the image is much less than 4 million numbers' worth-and it's precisely this fact that makes it possible to have compression, the critical mathematical technology that allows images, videos, music, and text to be stored in much smaller spaces than you'd think." Jordan Ellenberg

Source: Jordan Ellenberg, How Not to Be Wrong: The Power of Mathematical Thinking.