

450c - Binary Goodness of Fit, Principal Components Analysis

Apoorva Lal

April 14, 2022

Stanford

Binary Goodness of Fit

Cross Validation

PCA: Math

PCA: Mechanics and Implementation

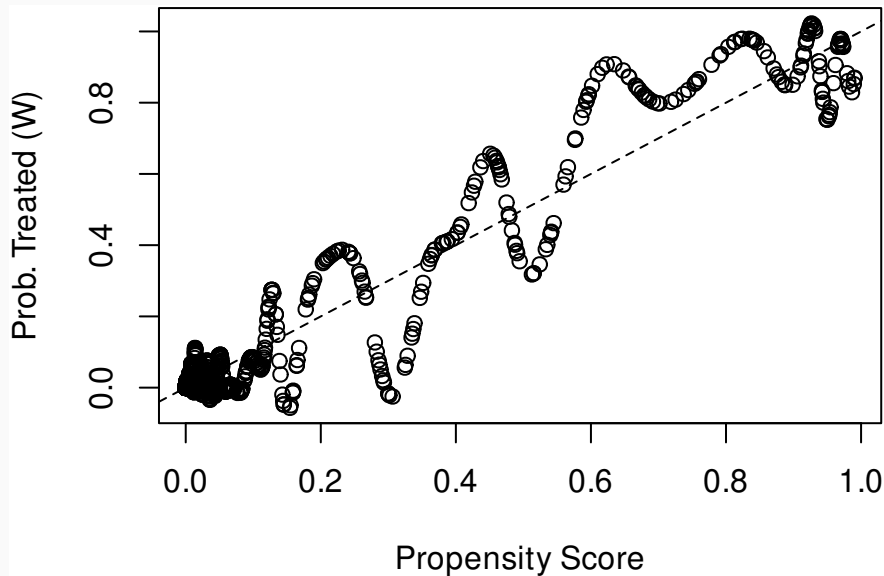
Binary Goodness of Fit

Binary Goodness of Fit

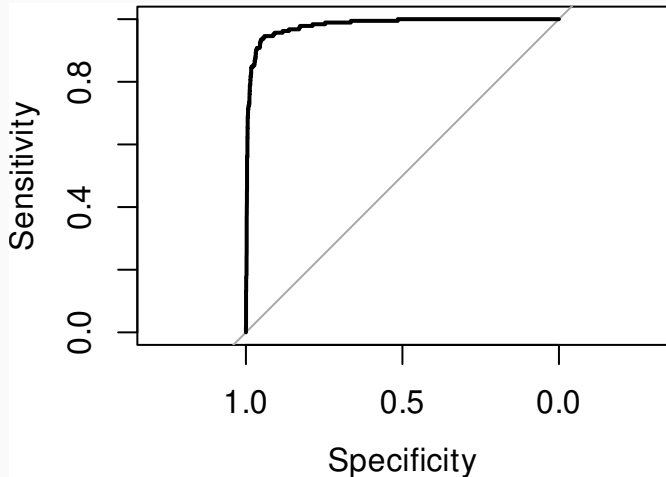
		True condition			
		Total population	Condition positive	Condition negative	
Predicted condition	Predicted condition positive	True positive	False positive, Type I error	Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$ False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	False negative, Type II error	True negative	False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$ F ₁ score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$	

```
data(lalonde.psid)
calib_curve = function(true, fitted){
  plot(smooth.spline(x = pscore, y = lalonde.psid[[w]], df = 4),
       xlab = "Propensity Score", ylab = "Prob. Treated (W)")
  abline(0, 1, lty="dashed")
}
y = 're78'; w = 'treat'; x = setdiff(names(lalonde.psid), c(y, w))
pscore = glm(formula_stitcher(w, x), lalonde.psid,
  family = binomial())$fitted.values
```

Calibration Figure



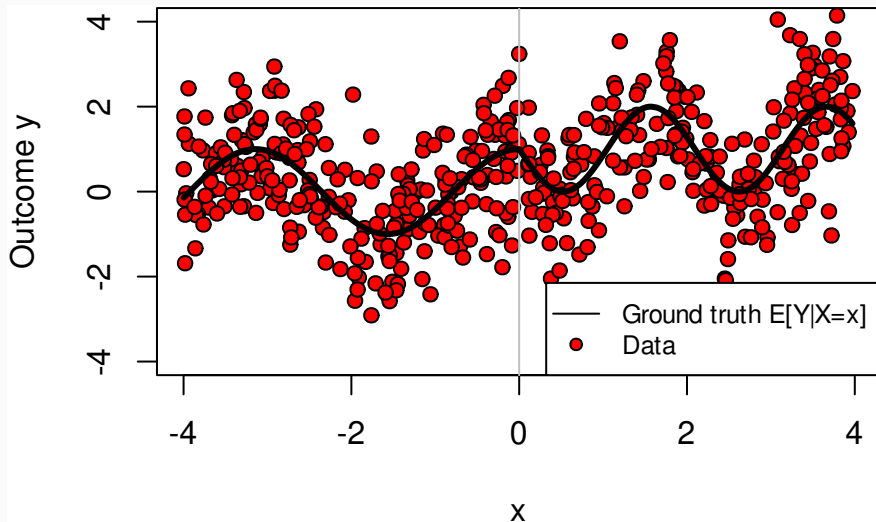
```
roc(lalonde.psid[[w]], pscore) |> plot()
```



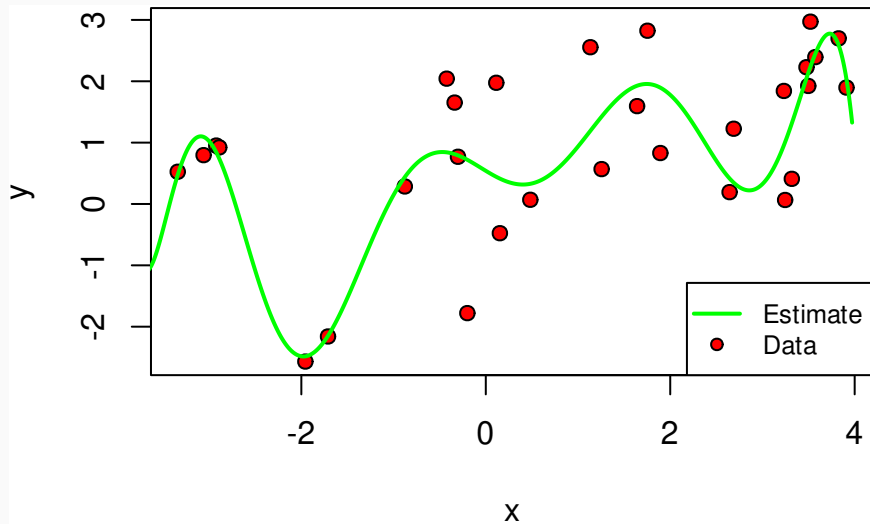
Cross Validation

$$y = \begin{cases} x < 0 & \cos(2x) \\ x \geq 0 & 1 - \sin(3x) \end{cases} + \varepsilon ; \varepsilon \sim \mathcal{N}(0, 1)$$

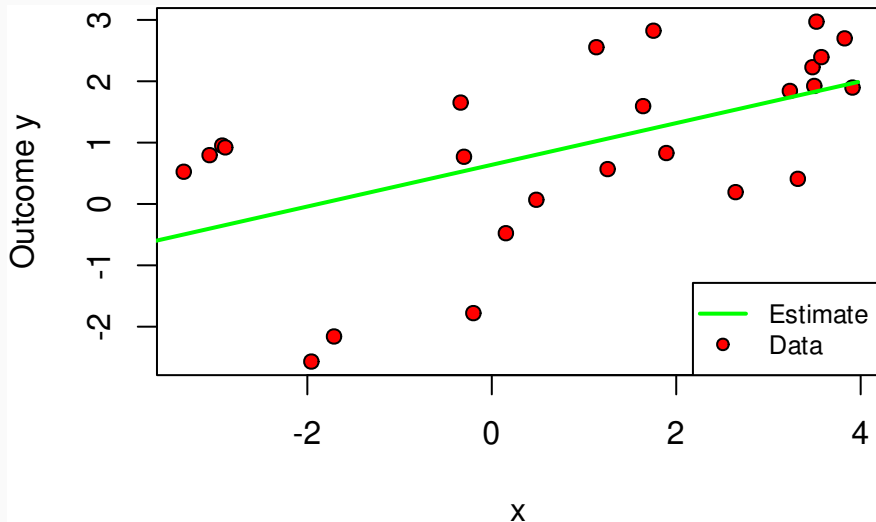
Population Regression Function



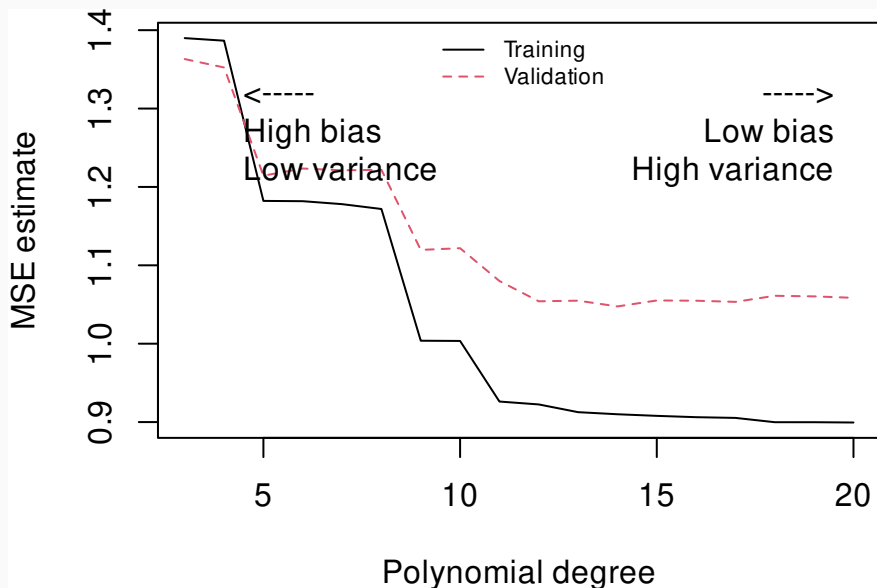
Overfit



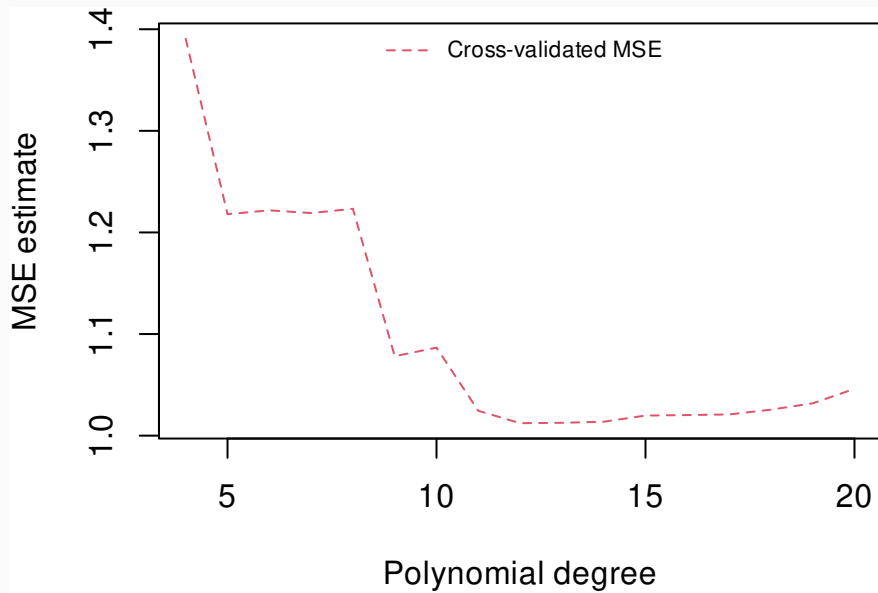
Underfit



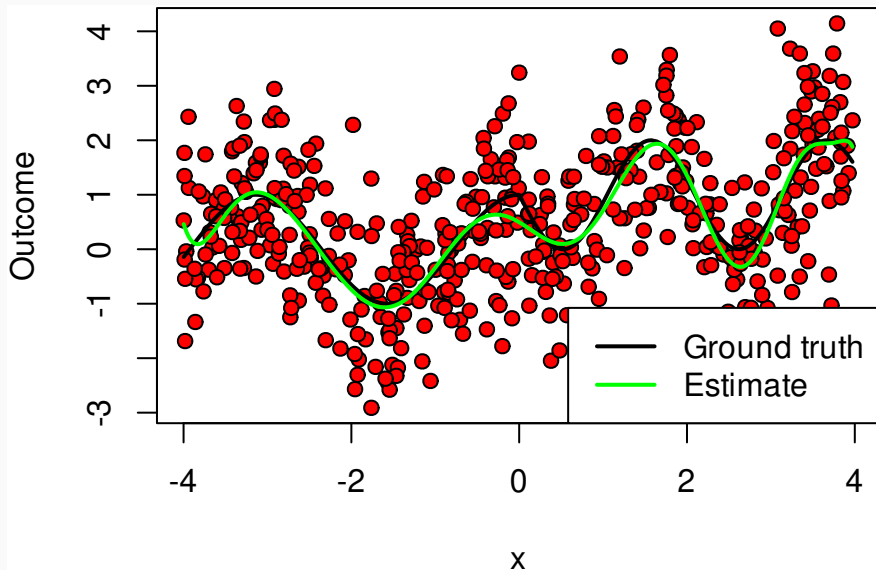
Bias Variance Tradeoff : MSE (test-train split)



MSE: K-fold cross validation

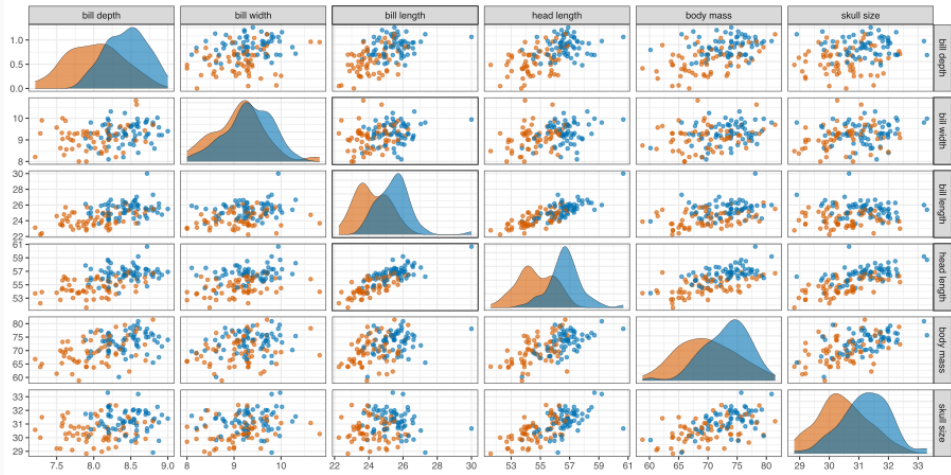


Fit (just right)



PCA: Math

Scatterplot Matrices can get unwieldy



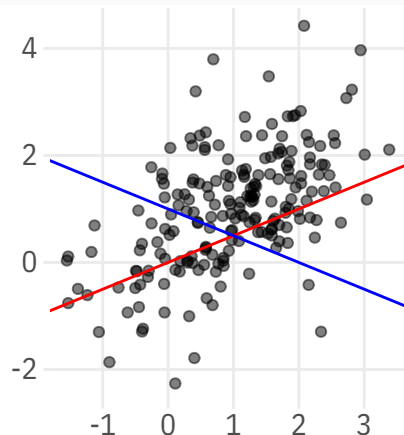
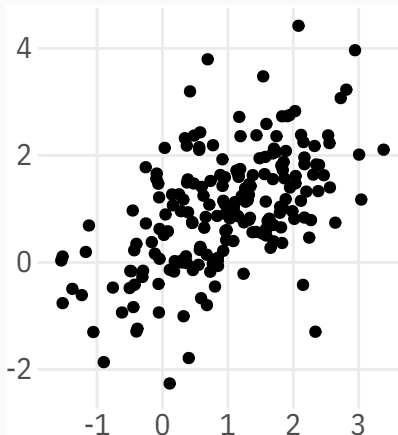
Source: Claus Wilke's data-viz course

Big Picture: Unsupervised Learning

- Difference between supervised and unsupervised learning
- **supervised:** predict y using x
 - regression
 - random forests
 - LASSO
 - support vector machines
 - neural networks
- **unsupervised:** characterise X
 - no response/label y ; only have a big data matrix X
 - categorise and cluster data (based on substantive knowledge)
 - principal components analysis
 - factor analysis - [link](#)
 - k -means clustering
 - scaling

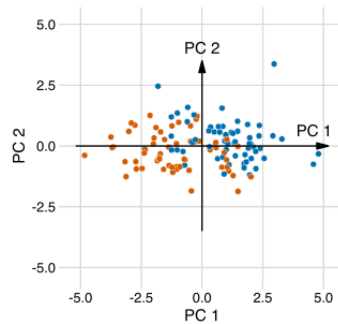
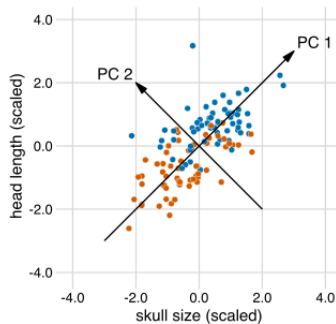
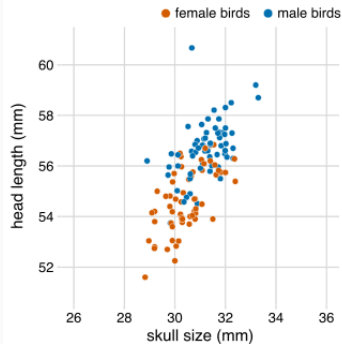
- Introductory **unsupervised** technique for *Dimension reduction*
- **Examples**
 - How can we order Democratic congressmen from most liberal to most conservative?
 - How can we rank vice-presidential candidates on different dimensions?
 - How can we classify speeches or votes?
- We don't have *ys* to construct a regression model.
- Need to infer *latent* structure

- Suppose we have the following cloud, and want to reduce it to one dimension:



- The core idea behind PCA is to pick the vector through the dimensions along which most the variance in the data is represented
- This amounts to fitting a hyperplane that minimises distance to each point
- That way, we retain as much information as possible
- Conversely, we minimise the reconstruction error – because we maximise the amount of information that we retain.

Basic Exercise Visualised

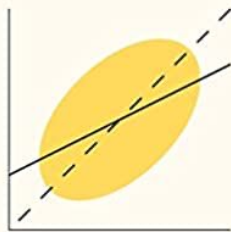


Source: Fundamentals of Data Visualisation

Statistical Models

Theory and Practice

REVISED EDITION



David A. Freedman

PCA: Mechanics and Implementation

- Matrix \mathbf{X} with dimensions $n \times p$.
- Objective is to reduce matrix to K dimensions
- PCA dimensions denoted by \mathbf{w}_k
- Each data point reconstructed by observation-specific weight (*score*) z_{ik} on dimensions \mathbf{w}_k .

$$\tilde{\mathbf{x}}_i = \sum_{k=1}^K z_{ik} \mathbf{w}_k$$

- Matrix \mathbf{X} with dimensions $n \times p$.
- Objective is to reduce matrix to K dimensions
- PCA dimensions denoted by \mathbf{w}_k
- Each data point reconstructed by observation-specific weight (*score*) z_{ik} on dimensions \mathbf{w}_k .

$$\tilde{\mathbf{x}}_i = \sum_{k=1}^K z_{ik} \mathbf{w}_k$$

- **Objective Function**

- Pick $\boldsymbol{\theta} := \mathbf{w}_k, z_{ik}$ as to minimise avg. reconstruction error:

$$\min_{\mathbf{w}, z_{ik}} \frac{1}{N} \sum_{i=1}^N \left\| \mathbf{x}_i - \sum_{k=1}^K z_{ik} \mathbf{w}_k \right\|^2$$

- Taking the FOC and simplifying yields the solution $\mathbf{w}_k^T \Sigma \mathbf{w}_k$.
- where Σ is the empirical covariance matrix $\frac{1}{n} \mathbf{X}^\top \mathbf{X}$
- Minimising reconstruction error \equiv Maximising variance of projected data [Slide 18]

$$\mathbf{w}_k^T \Sigma \mathbf{w}_k = \lambda_k$$

- \mathbf{w}_k^* is equal to the k th eigenvector of Σ , and $z_{ik}^* = \mathbf{w}_k^T \mathbf{x}_i$.

- Taking the FOC and simplifying yields the solution $\mathbf{w}_k^T \Sigma \mathbf{w}_k$.
- where Σ is the empirical covariance matrix $\frac{1}{n} \mathbf{X}^\top \mathbf{X}$
- Minimising reconstruction error \equiv Maximising variance of projected data [Slide 18]

$$\mathbf{w}_k^T \Sigma \mathbf{w}_k = \lambda_k$$

- \mathbf{w}_k^* is equal to the k th eigenvector of Σ , and $z_{ik}^* = \mathbf{w}_k^T \mathbf{x}_i$.
- Remember that $\mathbf{A}\mathbf{w} = \lambda\mathbf{w}$.

- Taking the FOC and simplifying yields the solution $\mathbf{w}_k^T \Sigma \mathbf{w}_k$.
- where Σ is the empirical covariance matrix $\frac{1}{n} \mathbf{X}^\top \mathbf{X}$
- Minimising reconstruction error \equiv Maximising variance of projected data [Slide 18]

$$\mathbf{w}_k^T \Sigma \mathbf{w}_k = \lambda_k$$

- \mathbf{w}_k^* is equal to the k th eigenvector of Σ , and $z_{ik}^* = \mathbf{w}_k^T \mathbf{x}_i$.
- Remember that $\mathbf{A}\mathbf{w} = \lambda\mathbf{w}$.
 - The eigenvector \mathbf{w} points out the vector in multidimensional space along which most of the variance-covariance matrix (Σ) can be captured.

- Taking the FOC and simplifying yields the solution $\mathbf{w}_k^T \Sigma \mathbf{w}_k$.
- where Σ is the empirical covariance matrix $\frac{1}{n} \mathbf{X}^\top \mathbf{X}$
- Minimising reconstruction error \equiv Maximising variance of projected data [Slide 18]

$$\mathbf{w}_k^T \Sigma \mathbf{w}_k = \lambda_k$$

- \mathbf{w}_k^* is equal to the k th eigenvector of Σ , and $z_{ik}^* = \mathbf{w}_k^T \mathbf{x}_i$.
- Remember that $\mathbf{A}\mathbf{w} = \lambda\mathbf{w}$.
 - The eigenvector \mathbf{w} points out the vector in multidimensional space along which most of the variance-covariance matrix (Σ) can be captured.
 - we're rotating the coordinate system as to remove the correlation between the covariates.

Singular Value Decomposition (SVD)

- Every matrix \mathbf{X} can be written as

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}$$

where $\mathbf{U} \in \mathbb{R}^{n \times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n \times p}$, $\mathbf{V} \in \mathbb{R}^{p \times p}$.

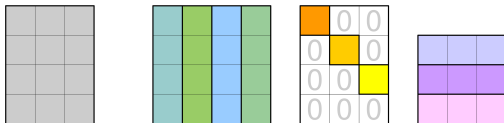
Singular Value Decomposition (SVD)

- Every matrix \mathbf{X} can be written as

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

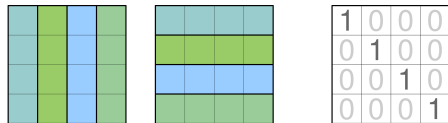
where $\mathbf{U} \in \mathbb{R}^{n \times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n \times p}$, $\mathbf{V} \in \mathbb{R}^{p \times p}$.

- $\mathbf{U}^\top \mathbf{U} = \mathbf{I}_n$ are left singular vectors
- $\mathbf{V}^\top \mathbf{V} = \mathbf{I}_p$ are right singular vectors
- $\mathbf{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_{\min\{n,p\}})$, where $\sigma_i > \sigma_{i+1}$
- Note: computers often store only $\min\{n, p\}$ dimensions
- *Principal components of $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}$*

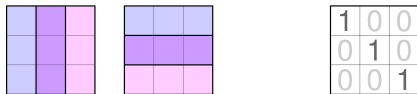


$$\mathbf{M} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^*$$

$m \times n$ $m \times m$ $m \times n$ $n \times n$

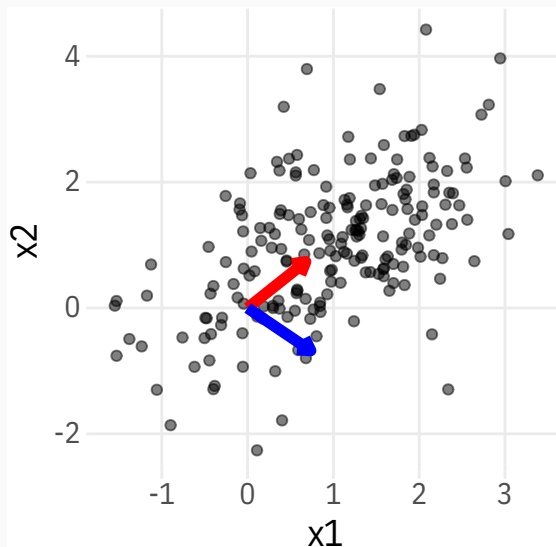


$$\mathbf{U} \mathbf{U}^* = \mathbf{I}_m$$



$$\mathbf{V} \mathbf{V}^* = \mathbf{I}_n$$

PCA: Viz with simulated data

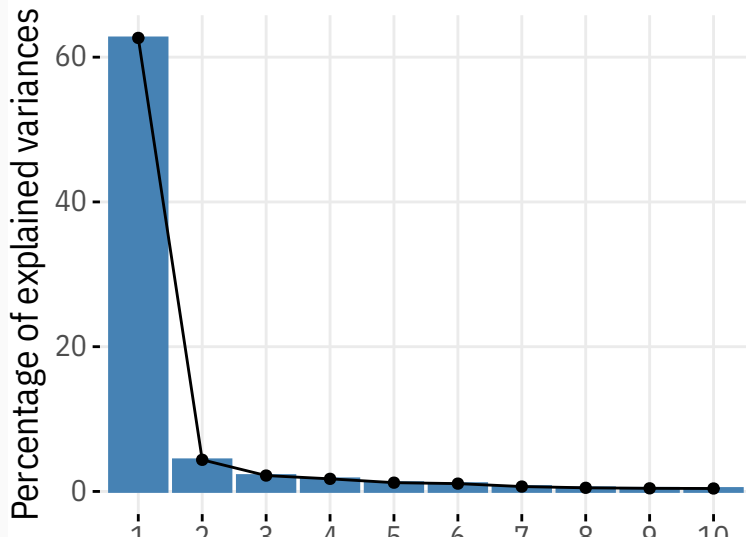


PCA: Implementation on Voting data

```
load("house_113.RData")
votes <- rc$votes[-1, ] %>% as_tibble() # removes Obama
votes <- votes %>% mutate_all(~ case_when(. %in% 1:3 ~ 1, TRUE ~ 0))
house <- rc$legis.data[-1,] %>% as_tibble()
pca <- prcomp(votes)
pca %>% glimpse
```

```
## List of 5
## $ sdev      : num [1:444] 12.52 3.31 2.34 2.09 1.74 ...
## $ rotation: num [1:1202, 1:444] -0.0367 -0.0374 -0.0379 0.0371 -0.038 ...
## ..- attr(*, "dimnames")=List of 2
## .. ..$ : chr [1:1202] "Vote 1" "Vote 2" "Vote 3" "Vote 4" ...
## .. ..$ : chr [1:444] "PC1" "PC2" "PC3" "PC4" ...
## $ center   : Named num [1:1202] 0.495 0.505 0.511 0.437 0.514 ...
## ..- attr(*, "names")= chr [1:1202] "Vote 1" "Vote 2" "Vote 3" "Vote 4" ...
## $ scale    : logi FALSE
## $ x        : num [1:444, 1:444] -0.984 -3.705 -11.935 -12.019 -10.986 ...
## ..- attr(*, "dimnames")=List of 2
## .. ..$ : NULL
## .. ..$ : chr [1:444] "PC1" "PC2" "PC3" "PC4" ...
## - attr(*, "class")= chr "prcomp"
```

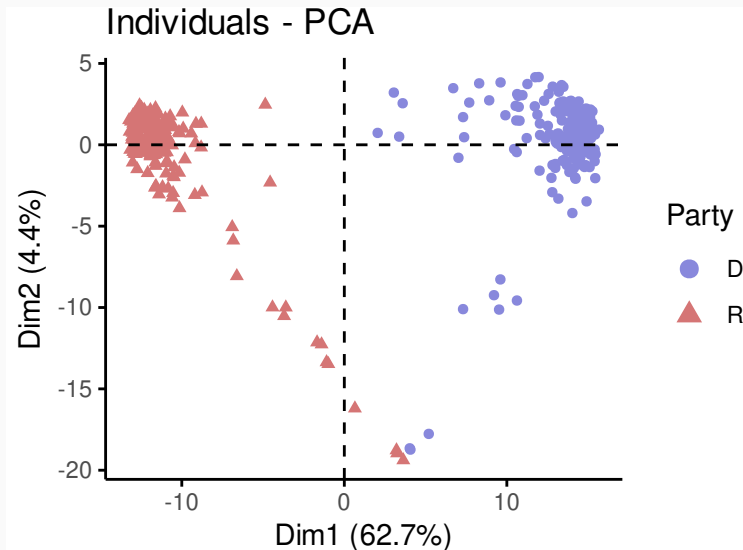
Scree plot



Generating PCs for units

```
house <- house %>% mutate(PC1 = pca$x[,1])  
house %>% arrange(PC1) %>% head(n = 5)
```

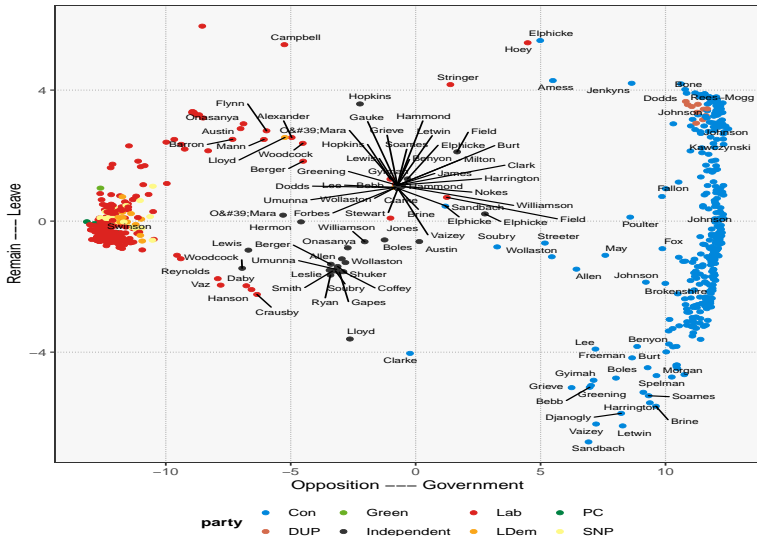
state	icpsrState	cd	icpsrLegis	party	partyCode	PC1
OH	24	1	29550	R	200	-13.24
TX	49	19	20353	R	200	-13.21
NC	47	13	21349	R	200	-13.17
OH	24	5	20755	R	200	-13.13
MO	34	7	21150	R	200	-13.10



Example 2: Analysing Brexit

- Roll-call example: We know that legislators in parliamentary systems predominantly vote along party lines
- But 2017-2019 UK Parliament was unusual: many, many rebellions with respect to Brexit
- Have a $n \times p$ votes matrix with n MPs and p divisions.
- Code an Aye vote as 1, a No vote as -1, and an abstention as 0.
- Use PCA to reduce dimensionality

PCA: Implementation (cont'd)



- For many text/ml applications, data is very large and very sparse
 - Netflix problem
- Use [irlba](#) in such settings

- PCA is a **dimension reduction** technique
- Convenient, but often not ideal:
 - Interpretation of principal components? Typically ad-hoc
 - Information loss
 - No easy way for categorical classification
- Next couple of weeks: more classification methods

