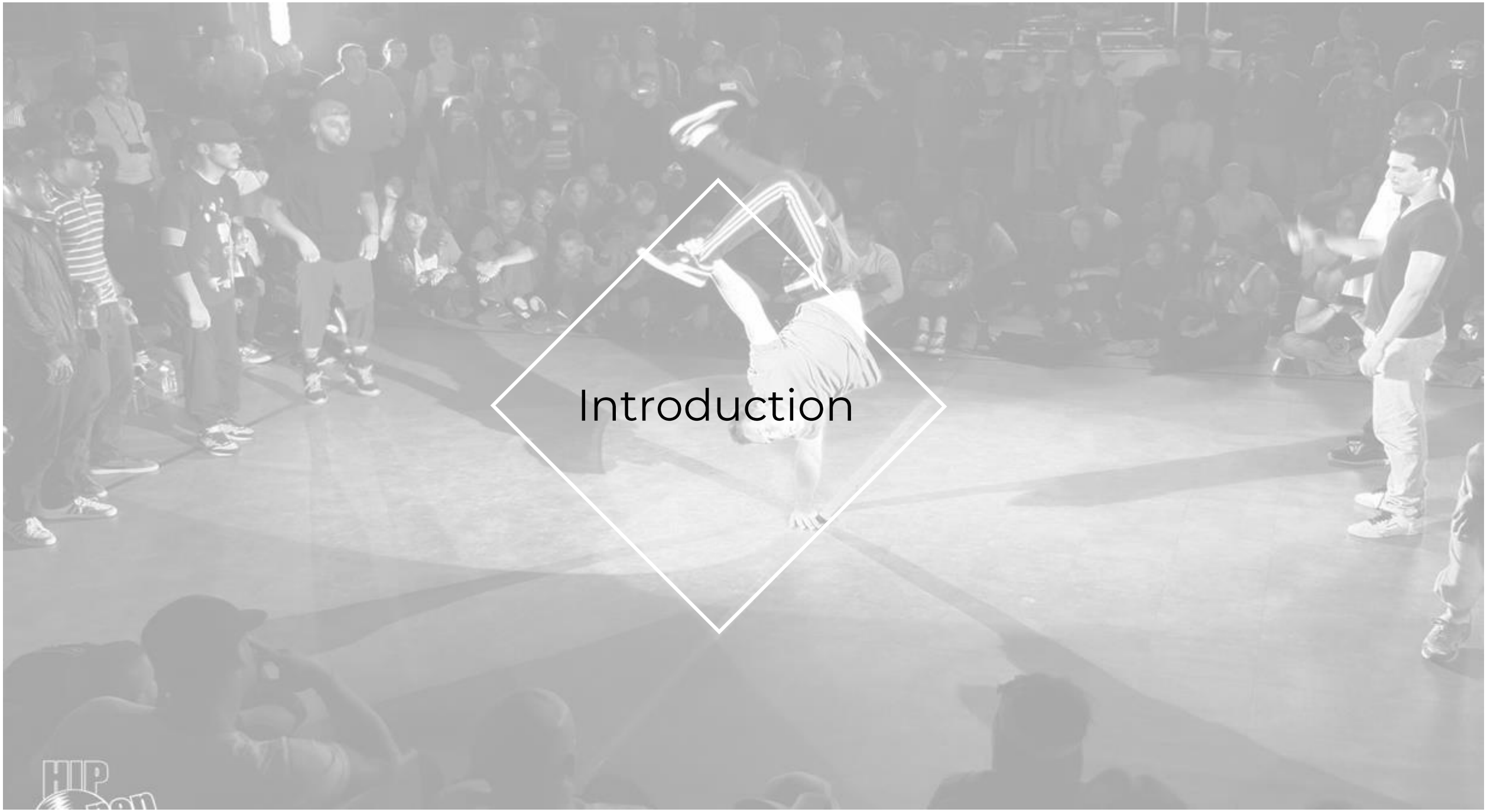
Disclaimer: These slides are not to be shared as some references are missing. The presented slides are a combination of various elements extracted from my papers, my codes but also coming from the net. The copyrights belong to their respective owners.

AI for Credit Scoring

Paris, June 14th 2022

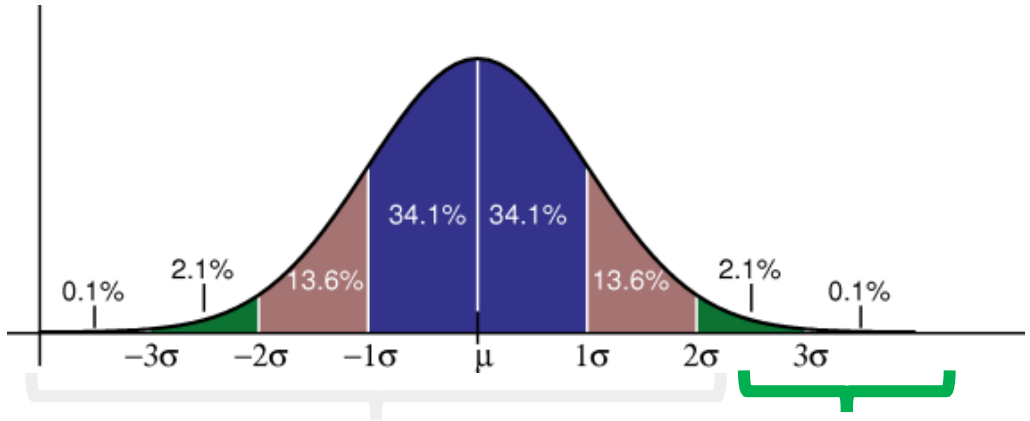
Bertrand K. Hassani



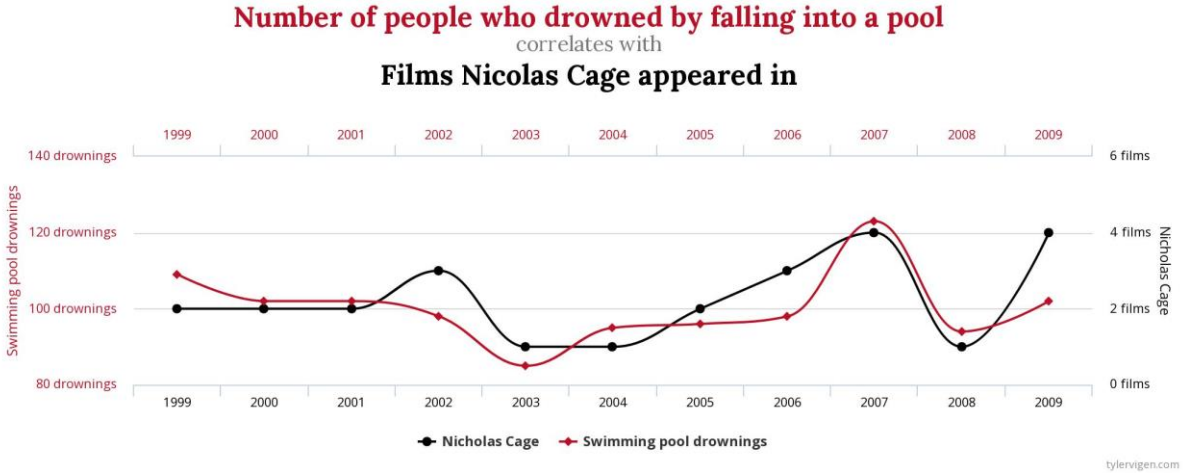
Introduction

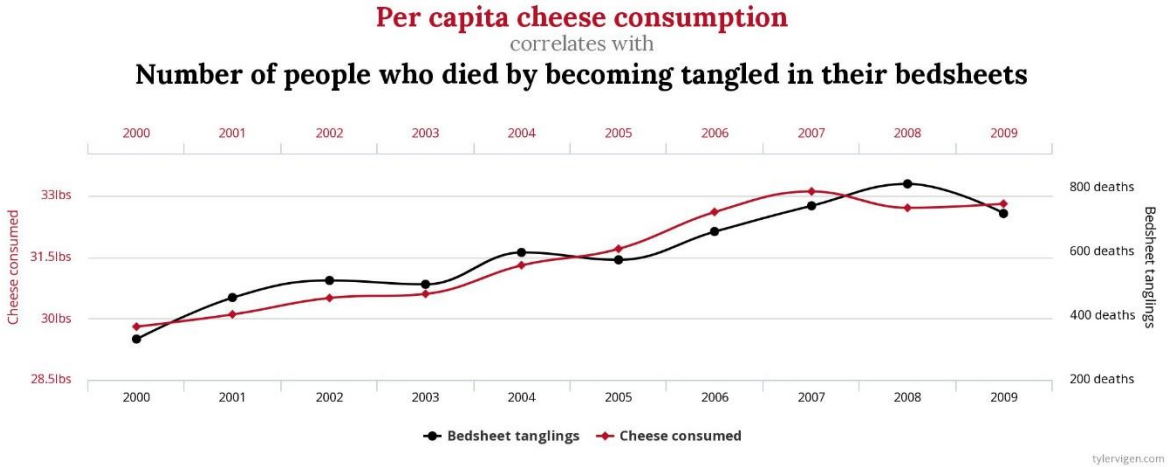
Fun Facts and Poll

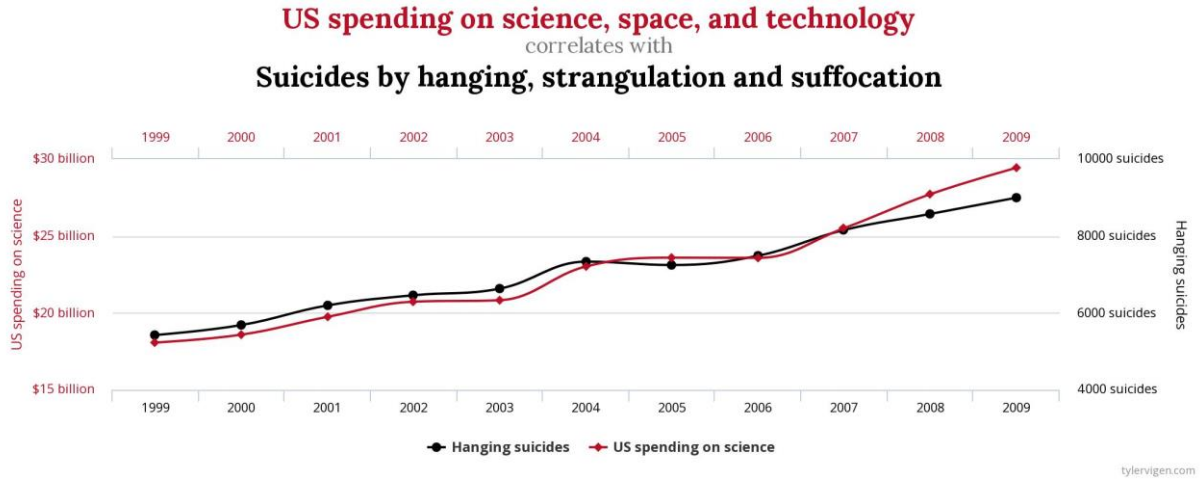
- $0^0 = 0$ or $= 1$?
- $1^3 + 2^3 + \ldots + n^3 =$ $(1 + 2 + \ldots + n)^2$
- **Euler Characteristic: Draw any number of dots on your page. Now connect the dots with lines, subject to the following rules: lines may not cross each other as they move from dot to dot, and every dot on your page must be connected to every other dot through a sequence of lines. Now count the number dots (D), lines (L), and regions separated by lines (R). (Don't forget to count the outside as a region too, i.e. add 1 to R)**
- **Expected Value of a Prayer.**
- **Condorcet Voting Paradox**



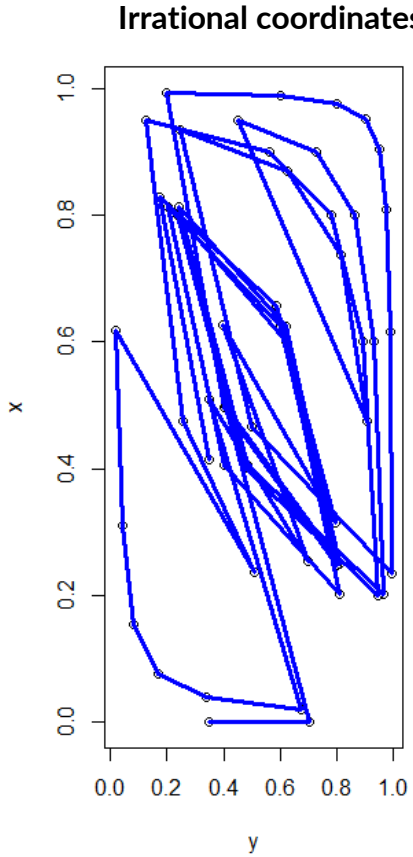
Bullshit is around the corner !

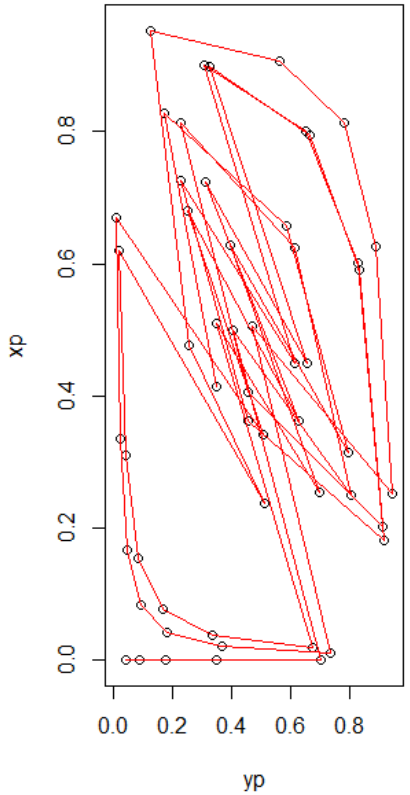


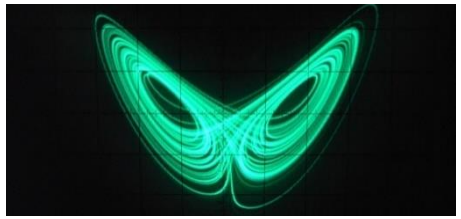


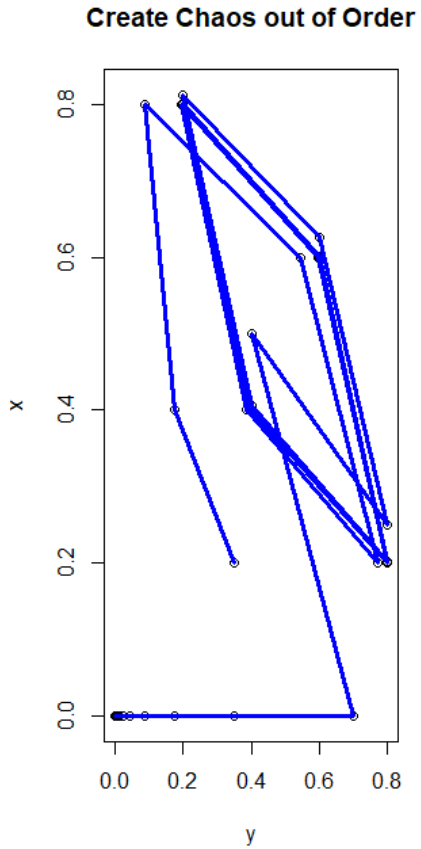


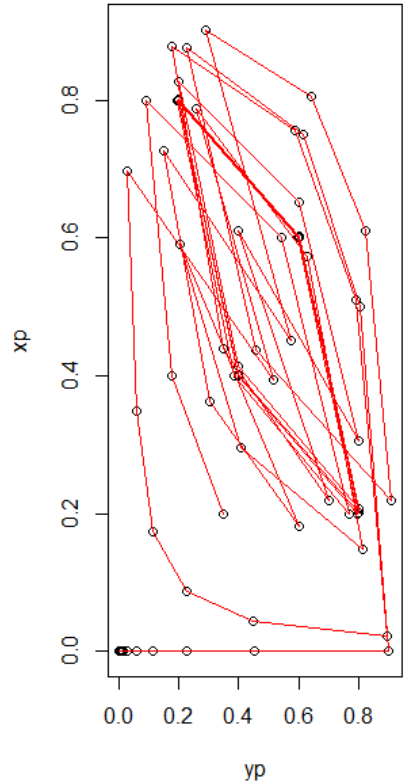
If you think you are right, you are wrong…. My little butterfly…









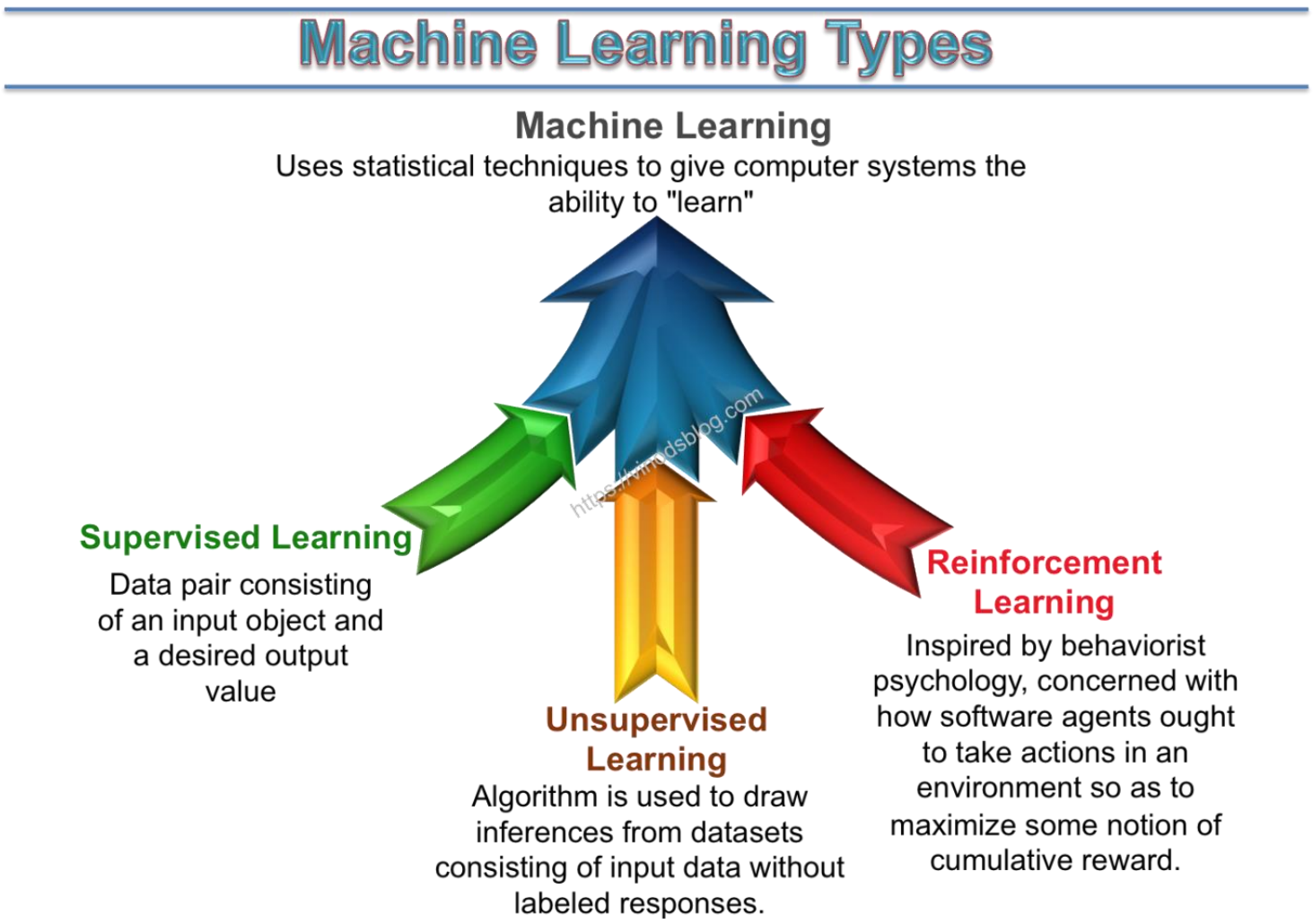


Data Science – Let's try to structure it

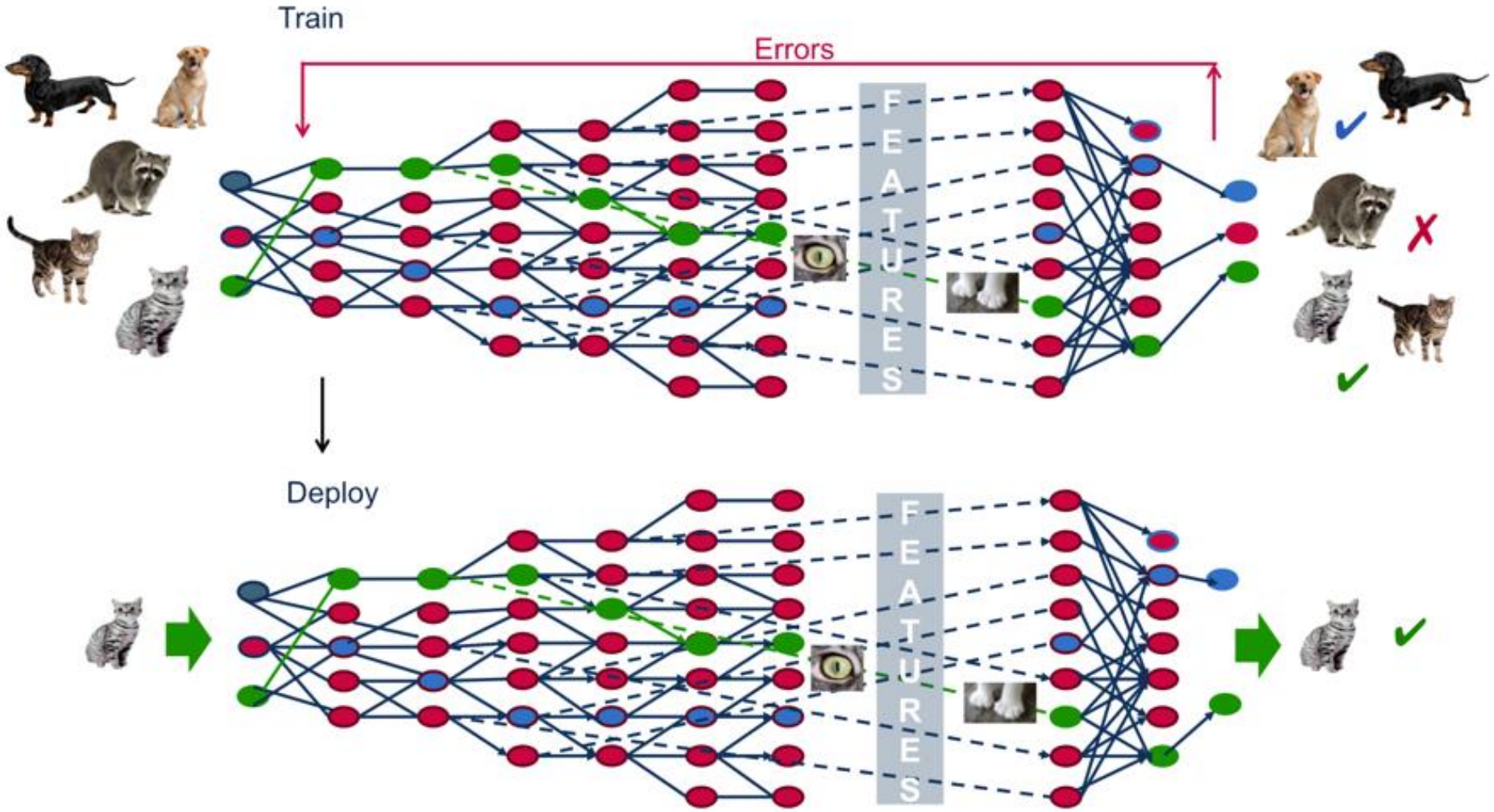
		Data Mining *(weak A.I.)*	**Machine Learning** *(weak A.I.)*	**Artificial Intelligence** *(strong A.I.)*
Different objectives…		**Analyse and order a data set** Pattern detection and recognition and features identification	**Learn how to order a data set in a closed environment** Learn classification and regression	**Learn how to order a data set in an opened environment and dynamically adapt to changes without human intervention**
… But same methodologies applied, often combined between themselves	**Natural Language Processing (NLP)**	**Understand and process Natural Language** data through statistical models (e.g. Conditional Random Forest)	**Text mining**: Text categorization, text clustering, concept/entity extraction	**Natural Language Understanding**: Analysis of concepts, entities, keywords, categories, relations, sentiment analysis
			Natural Language Generation: Generation of natural language from a machine representation system such as a knowledge base or a logical form	**Optical Character Recognition**: Conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo
			Cognitive Robotics *(same output but transformed in movement)*: Endowing a robot with intelligent behavior by providing it with a processing architecture to allow it to learn and reason about how to behave in response to complex goals	
	Deep Learning	**Get deep and granular learning on data by filtering data** through multiple layers of neural networks (**Recurrent Neural Network** or **Convolutional Neural Network**) to improve classification	**Speech recognition**: Recognition and translation of spoken language into text by computers	**Computer vision**: High-level understanding from digital images or videos
			Pattern Detection: Focuses on the recognition of patterns and regularities	**Pattern Prediction**: Prediction of patterns and weak signals

Adoption level
- Growing adoption
- Scarce adoption

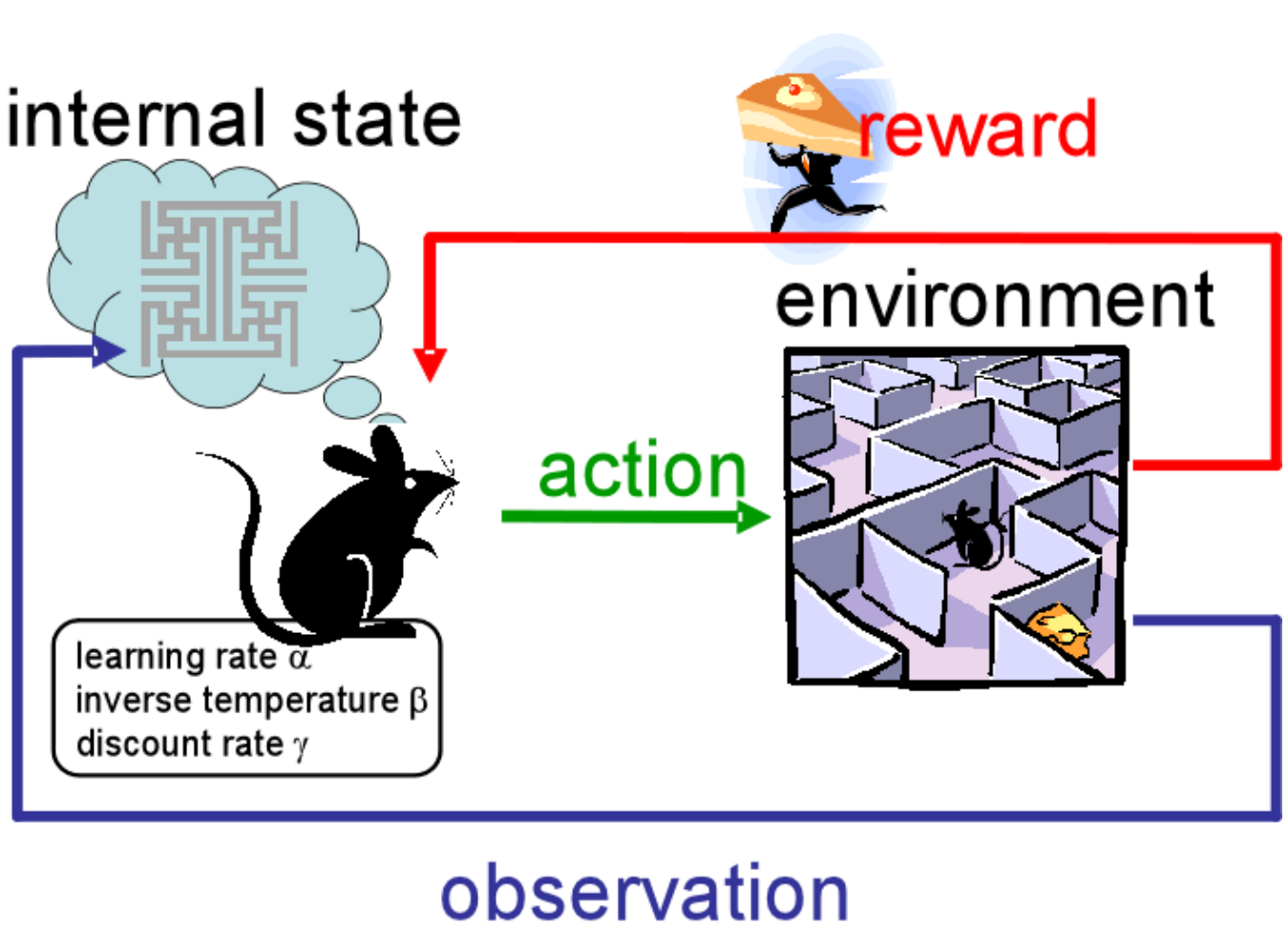
Quick Details

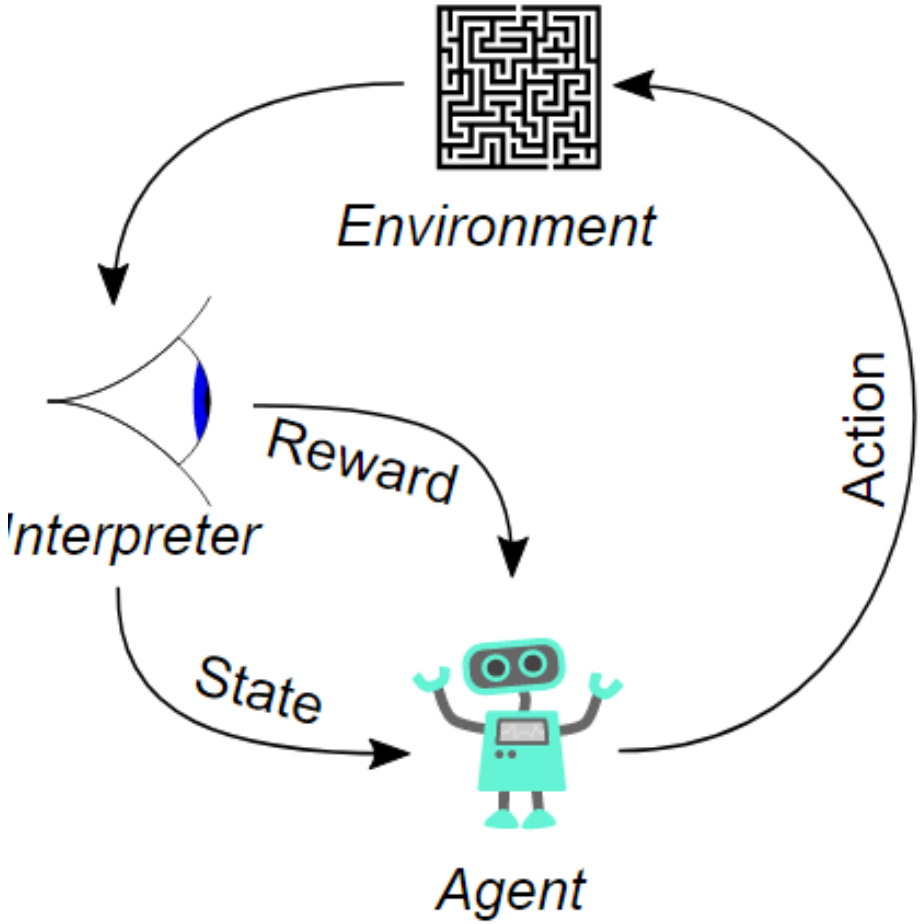


Deep Learning – Deep What?



Reinforcement Learning – Humans call it experience





Performance Indicators (1/3)

condition positive (P)
the number of real positive cases in the data

condition negative (N)
the number of real negative cases in the data

---

true positive (TP)
A test result that correctly indicates the presence of a condition or characteristic

true negative (TN)
A test result that correctly indicates the absence of a condition or characteristic

false positive (FP)
A test result which wrongly indicates that a particular condition or attribute is present

false negative (FN)
A test result which wrongly indicates that a particular condition or attribute is absent

---

sensitivity, recall, hit rate, or true positive rate (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$$

specificity, selectivity or true negative rate (TNR)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{FPR}$$

precision or positive predictive value (PPV)

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = 1 - \mathrm{FDR}$$

negative predictive value (NPV)

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} = 1 - \mathrm{FOR}$$

miss rate or false negative rate (FNR)

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} = 1 - \mathrm{TPR}$$

fall-out or false positive rate (FPR)

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} = 1 - \mathrm{TNR}$$

false discovery rate (FDR)

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}} = 1 - \mathrm{PPV}$$

false omission rate (FOR)

$$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TN}} = 1 - \mathrm{NPV}$$

Positive likelihood ratio (LR+)

$$\mathrm{LR+} = \frac{\mathrm{TPR}}{\mathrm{FPR}}$$

Negative likelihood ratio (LR-)

$$\mathrm{LR-} = \frac{\mathrm{FNR}}{\mathrm{TNR}}$$

prevalence threshold (PT)

$$\mathrm{PT} = \frac{\sqrt{\mathrm{TPR}(-\mathrm{TNR}+1)} + \mathrm{TNR} - 1}{(\mathrm{TPR} + \mathrm{TNR} - 1)} = \frac{\sqrt{\mathrm{FPR}}}{\sqrt{\mathrm{TPR}} + \sqrt{\mathrm{FPR}}}$$

threat score (TS) or critical success index (CSI)

$$\mathrm{TS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}$$

Prevalence

$$\frac{\mathrm{P}}{\mathrm{P} + \mathrm{N}}$$

accuracy (ACC)

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

balanced accuracy (BA)

$$\mathrm{BA} = \frac{TPR + TNR}{2}$$

F1 score

is the harmonic mean of precision and sensitivity:

$$\mathrm{F_1} = 2 \times \frac{\mathrm{PPV} \times \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

phi coefficient (φ or r_φ) or Matthews correlation coefficient (MCC)

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}$$

Fowlkes–Mallows index (FM)

$$\mathrm{FM} = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}} = \sqrt{PPV \times TPR}$$

informedness or bookmaker informedness (BM)

$$\mathrm{BM} = \mathrm{TPR} + \mathrm{TNR} - 1$$

markedness (MK) or deltaP (Δp)

$$\mathrm{MK} = \mathrm{PPV} + \mathrm{NPV} - 1$$

Diagnostic odds ratio (DOR)

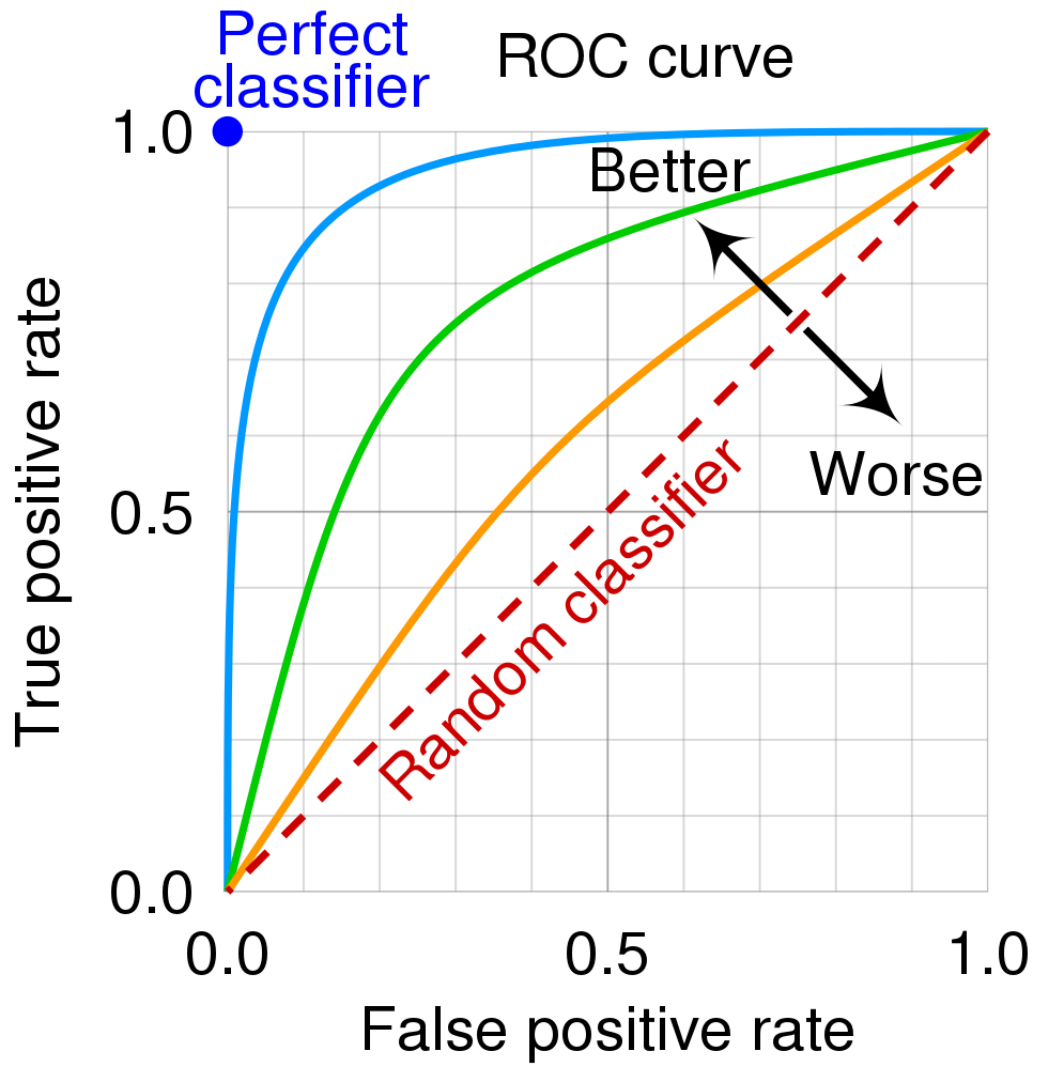
$$\mathrm{DOR} = \frac{\mathrm{LR+}}{\mathrm{LR-}}$$

Performance Indicators (2/3)

Sources: [15][16][17][18][19][20][21][22] view · talk · edit

		Predicted condition			
	Total population = P + N	**Positive (PP)**	**Negative (PN)**	Informedness, bookmaker informedness (BM) = TPR + TNR − 1	Prevalence threshold (PT) = $\frac{\sqrt{TPR \times FPR} - FPR}{TPR - FPR}$
Actual condition	**Positive (P)**	**True positive (TP),** hit	**False negative (FN),** type II error, miss, underestimation	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power = $\frac{TP}{P}$ = 1 − FNR	False negative rate (FNR), miss rate = $\frac{FN}{P}$ = 1 − TPR
	Negative (N)	**False positive (FP),** type I error, false alarm, overestimation	**True negative (TN),** correct rejection	False positive rate (FPR), probability of false alarm, fall-out = $\frac{FP}{N}$ = 1 − TNR	True negative rate (TNR), specificity (SPC), selectivity = $\frac{TN}{N}$ = 1 − FPR
	Prevalence = $\frac{P}{P + N}$	Positive predictive value (PPV), precision = $\frac{TP}{PP}$ = 1 − FDR	False omission rate (FOR) = $\frac{FN}{PN}$ = 1 − NPV	Positive likelihood ratio (LR+) = $\frac{TPR}{FPR}$	Negative likelihood ratio (LR−) = $\frac{FNR}{TNR}$
	Accuracy (ACC) = $\frac{TP + TN}{P + N}$	False discovery rate (FDR) = $\frac{FP}{PP}$ = 1 − PPV	Negative predictive value (NPV) = $\frac{TN}{PN}$ = 1 − FOR	Markedness (MK), deltaP (Δp) = PPV + NPV − 1	Diagnostic odds ratio (DOR) = $\frac{LR+}{LR-}$
	Balanced accuracy (BA) = $\frac{TPR + TNR}{2}$	F_1 score = $\frac{2PPV \times TPR}{PPV + TPR}$ = $\frac{2TP}{2TP + FP + FN}$	Fowlkes–Mallows index (FM) = $\sqrt{PPV \times TPR}$	Matthews correlation coefficient (MCC) = $\sqrt{TPR \times TNR \times PPV \times NPV}$ − $\sqrt{FNR \times FPR \times FOR \times FDR}$	Threat score (TS), critical success index (CSI), Jaccard index = $\frac{TP}{TP + FN + FP}$

Performance Indicators (3/3)

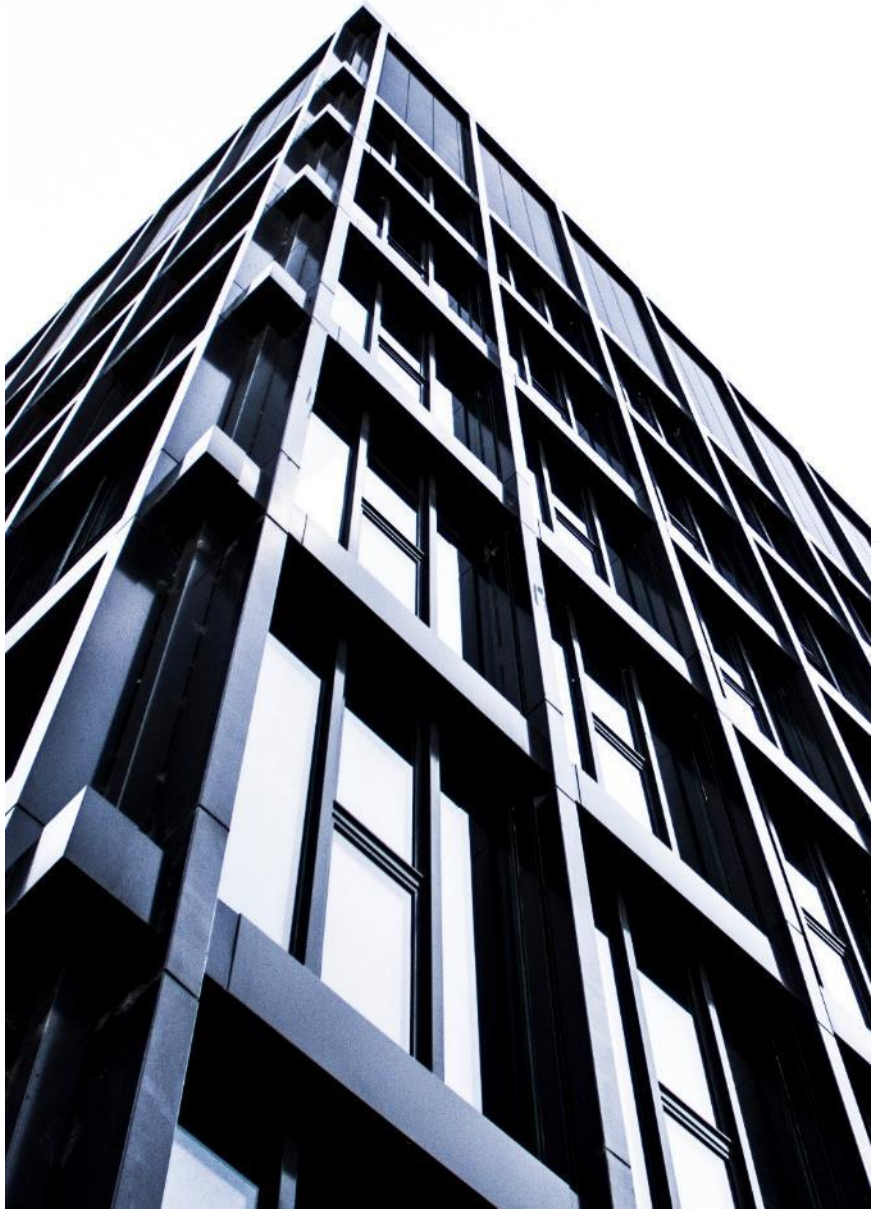


Gini coefficient
= 2 X AUC - 1

Model Interpretability ..... Towards Actionability

When we have a large number of predictors in the model, there will generally be many that have little or no effect on the response.

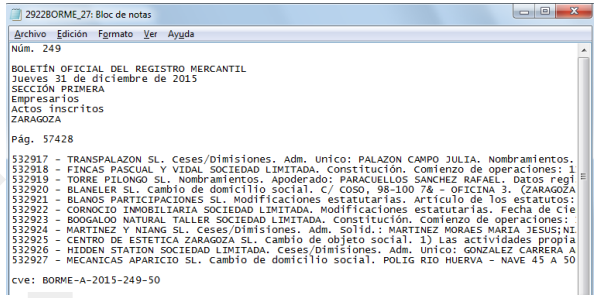
Including such irrelevant variable leads to unnecessary complexity.

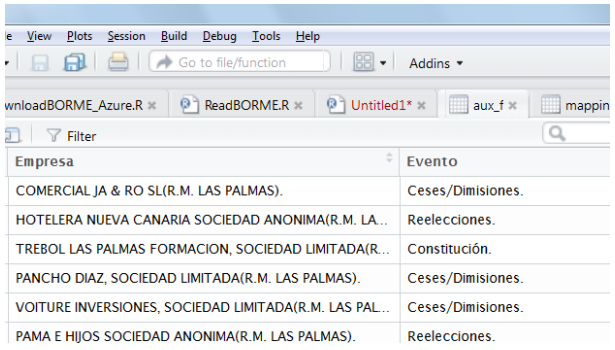


Leaving these variables in the model makes it harder to see the effect of the important variables.

The model would be easier to interpret by removing (i.e. setting the coefficients to zero) the unimportant variables.

Parsing, Text Mining, NLP – Watson's Dead?





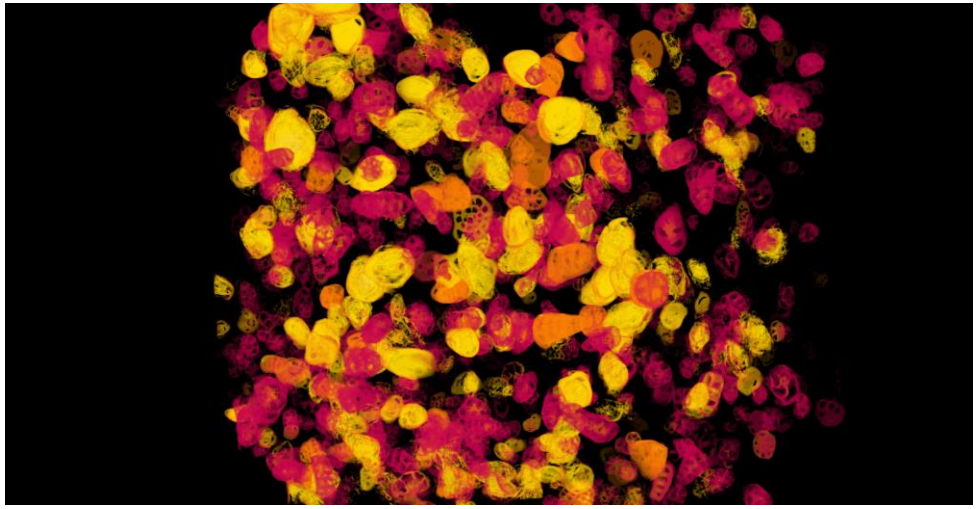
Getting new data into models is key to increase discriminatory power of models. E.g. doing bank accounts webscrapping of new to bank customer can increase GINI and profit between 10%-20% for this segment.

glassdoor™

THE TIMES

Risk Culture and Reputational Risk: Unstructured data coming from social network and online version of traditional medias are used to measure both how embedded is the risk culture and how we are perceived on the market.

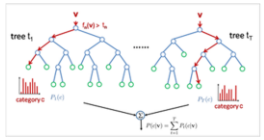
Image Recognition



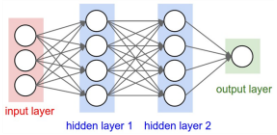
AI for Controls – “How did we let that happen?”

Preventative Controls

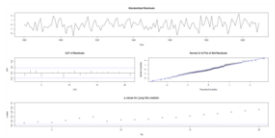
Random Forests



Neural Networks

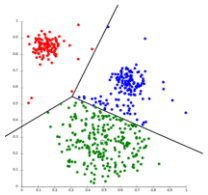


Time Series

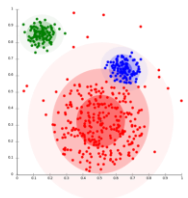


Detective Controls

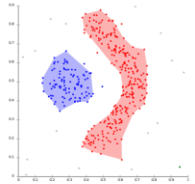
K-Means



EM Partitioning

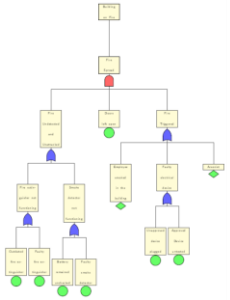


Density Approach

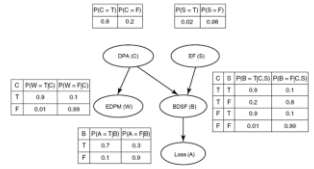


Corrective Controls

Fault Trees



Bayesian Networks



Operational risk is "the risk of a change in value caused by the fact that actual losses, incurred for inadequate or failed internal processes, people and systems, or from external events (including legal risk), differ from the expected losses".

Risk

Control

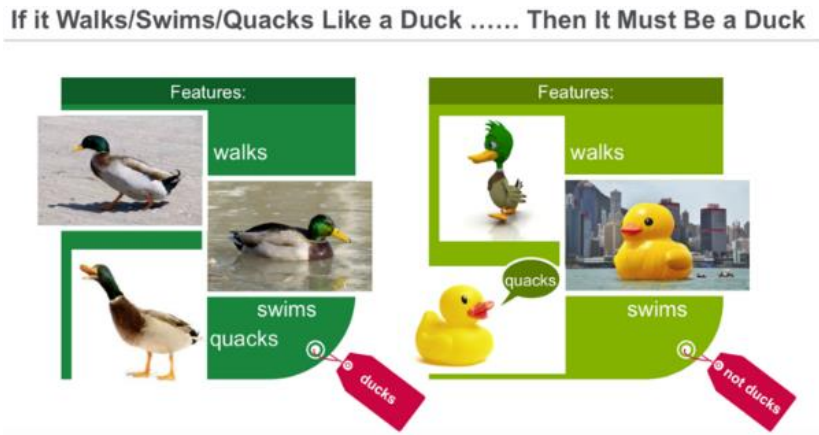
Failure

AI or more for Credit Scoring

```
if(!is.element("adabag", installed.packages()[, 1]))
      install.packages("adabag")
if(!is.element("glmnet", installed.packages()[, 1]))
      install.packages("glmnet")
if(!is.element("randomForest", installed.packages()[, 1]))
      install.packages("randomForest")
if(!is.element("e1071", installed.packages()[, 1]))
      install.packages("e1071")
```

Modelling	GINI
Classic model (logistic regression)	51.36
Lasso	53.98
Random Forest / Ranger	57.84
Boosting	44.16
SVM	28.38
Neural Network	51.94
Deep learning	44.92

+12.6%



ORIGINATION

PORTFOLIO MANAGEMENT

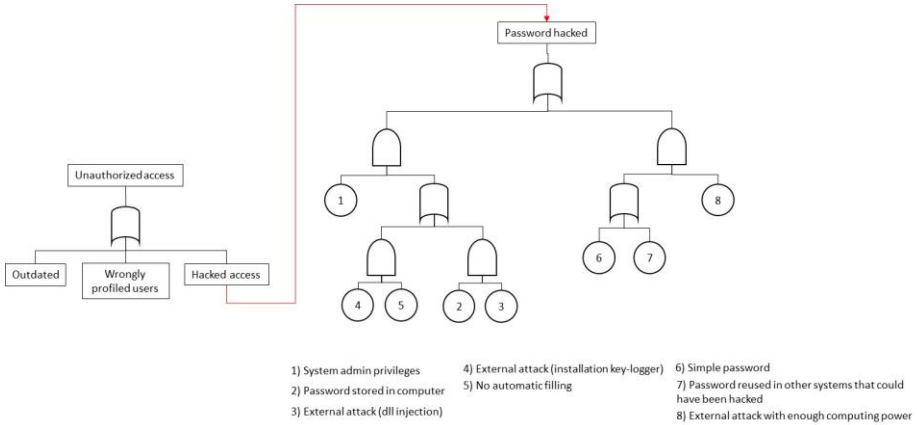
RECOVERY

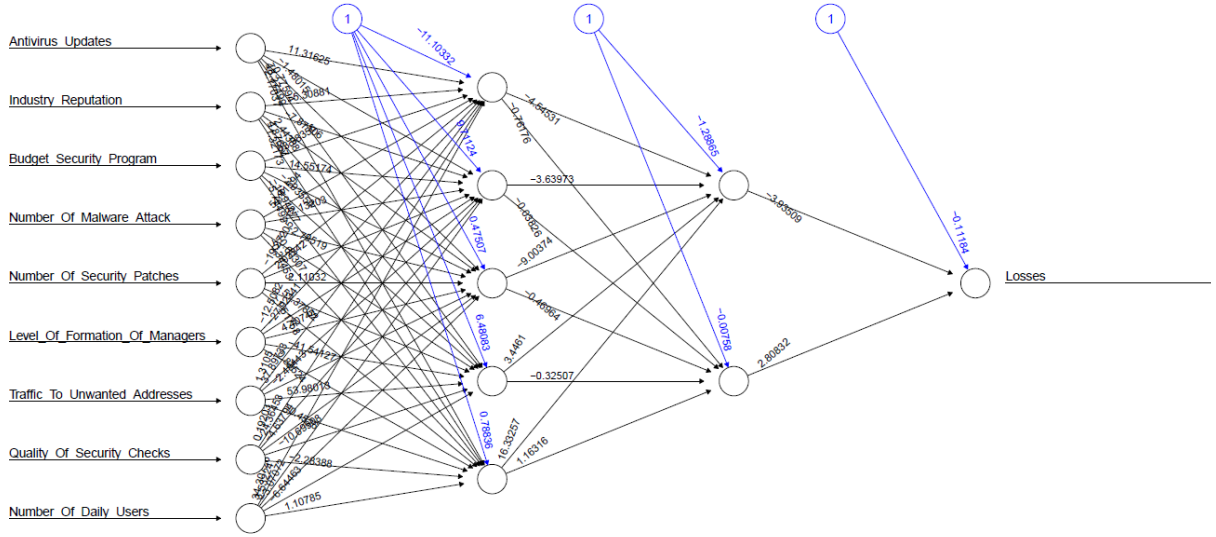
These “re-newed” technics owe their success to the increase of data availability and the improvement of IT infrastructures.

Cyber Security

	Resounding successful cyber-attacks 16H1				Information theft 16H1	
Type of attack	**# attacks**	**%**	**Relevant targets**	**Main authors**	**# attacks**	**%**
Account Hijacking	72	15%	Donald Trump Mar Zuckerberg FBI Standford Ringo Star Snapchat	Anonymous Isis World Hacker Team Penis AKA @DotGovs OurMine Team	53	74%
DDos	53	11%	Nissan Taiwanese Prison System Ku Klux Klan website	Anonymous World Hacker Team Ghost Squad	0	0%
Targeted attack	49	10%	US Government	Sofacy	38	78%
SQLi	48	10%	Spanish Police Department 33 Turkish Hospitals	@FkPoliceAnonOps Anonymous	47	98%
Defacement	28	6%	Dell	MuhmadEmad	1	4%
Malware	21	4%	Nasa South Korea	Anonsec North Korea	12	57%
Malversiting	14	3%	MSN Skype	Unknown	4	29%
Brute Force	4	1%	US Internal Revenue Service Spotify	Unknown	4	100%
Others	206	42%	Cases with less tahtn 4 types of attack in the considered period		183	89%
TOTAL	495	100%			342	69%

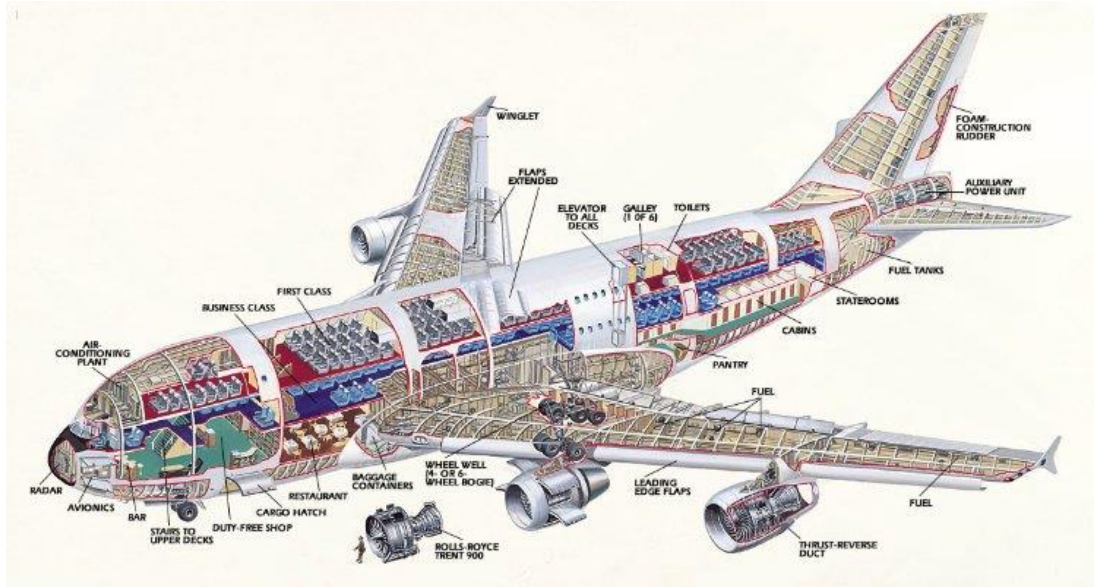
"Cybercrime is a fast-growing area of crime. More and more criminals are exploiting the speed, convenience and anonymity of the Internet to commit a diverse range of criminal activities that know no borders, either physical or virtual, cause serious harm and pose very real threats to victims worldwide." - Interpol





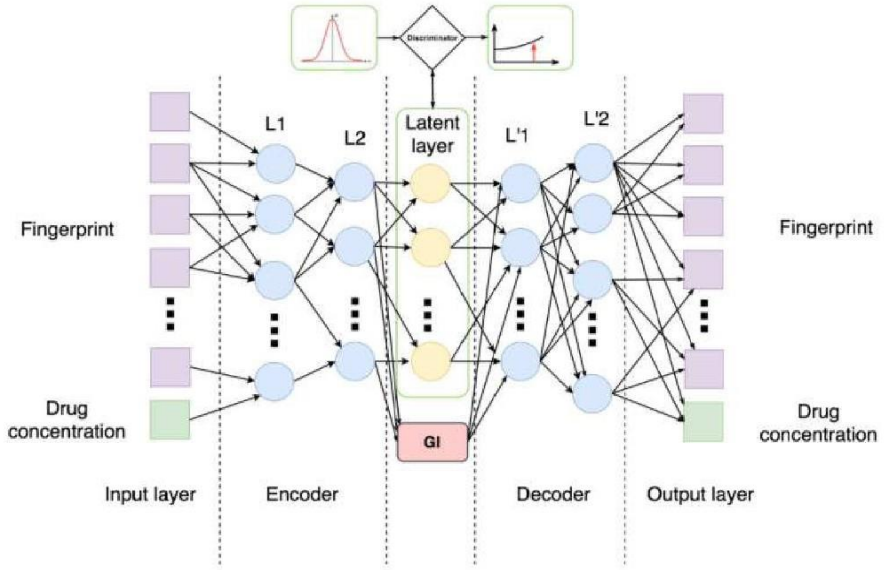
Predictive Maintenance

- Integrated Digital Deployment and Upscaling: act before something fails to ensure business continuity….
- Data a “mean to an end” strategy: internal services oriented
- Transitional Support
- Application: Any Factory, Solar Plant, Automotive, Aeronotics



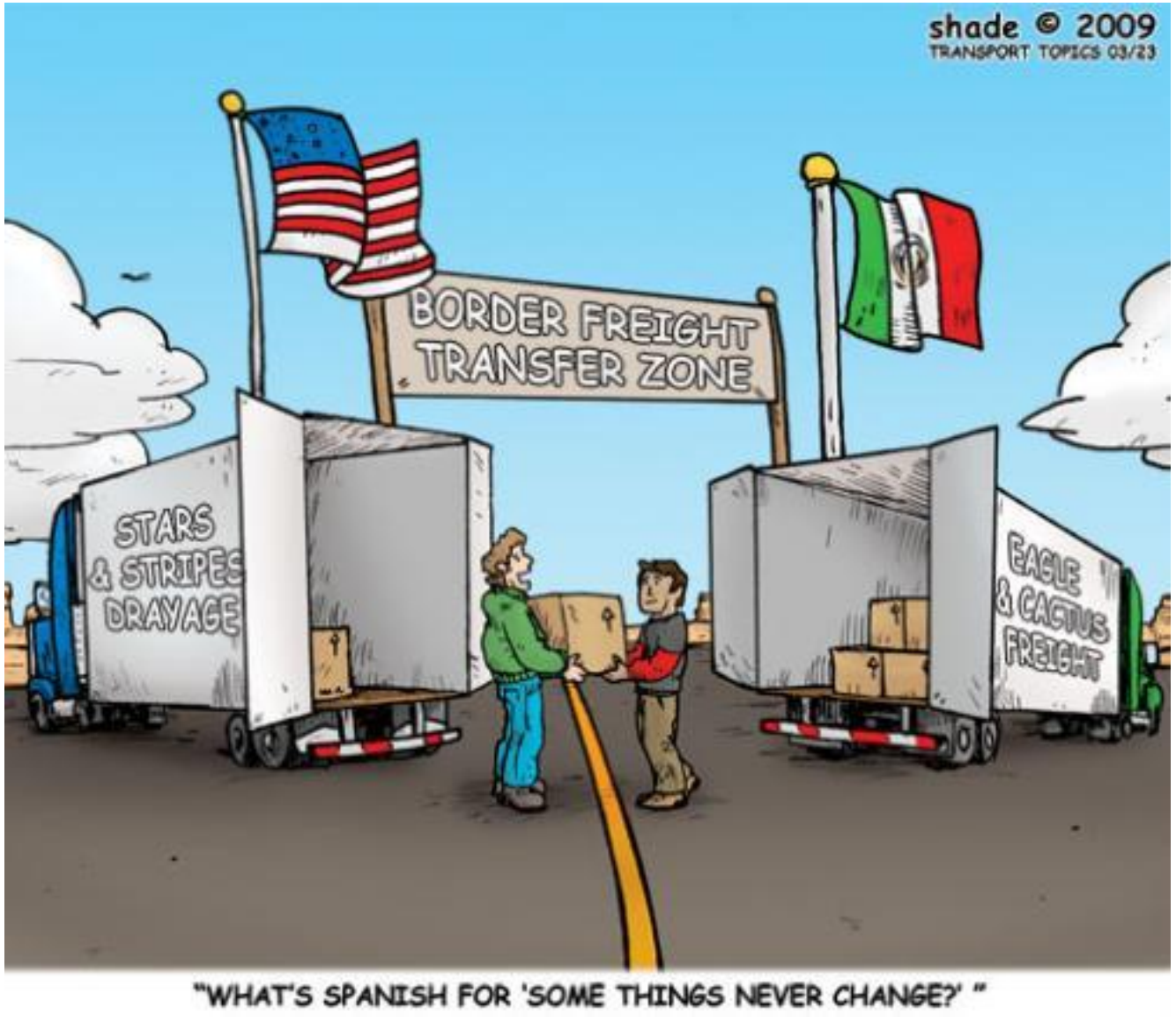
Pharmaceutical

- 14th Feb 2017: Russian scientists deploy GANs to develop Anti-Cancer Drugs
- 72 million molecules “fingerprinted”. Mapping of known anti-cancer drugs
- **Hundreds of new anti-cancer drugs generated, including known ones that were not yet in the training**



- According to Alex Zhavoronkov, founder of Insilico Medicine: *“GANs were proposed only recently (in 2014) and scientists are still exploring its power. The pace of progress is accelerating and soon we are likely to see tremendous advances stemming from combinations of GANs with other methods”*

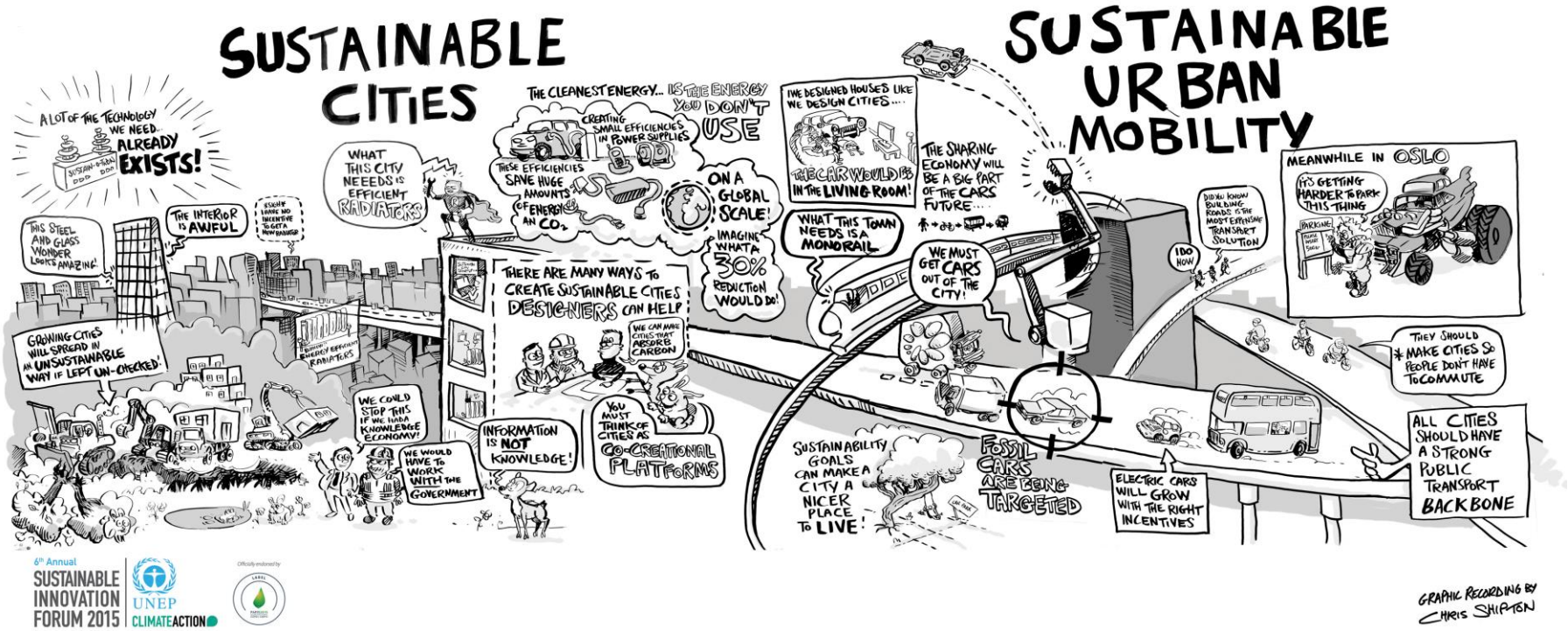
Logistics


shade © 2009
TRANSPORT TOPICS 03/23
BORDER FREIGHT TRANSFER ZONE
STARS & STRIPES DRAYAGE
EAGLE & CACTUS FREIGHT
"WHAT'S SPANISH FOR 'SOME THINGS NEVER CHANGE?' "

Mobility


UBER

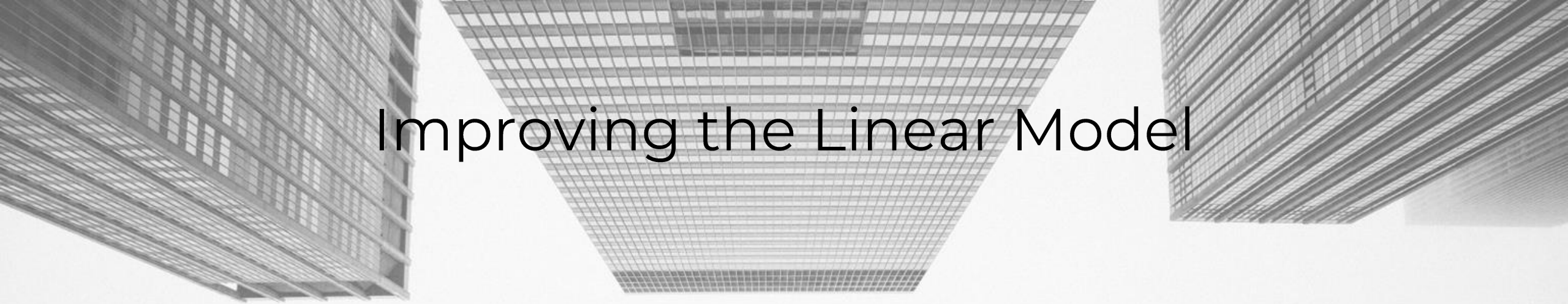

Closed
Client Number xxxxx
Wait Time 11min
Closed
Taxi Number 2567
Capacity 2/5


SUSTAINABLE CITIES
A LOT OF THE TECHNOLOGY WE NEED ALREADY EXISTS!
THIS STEEL AND GLASS WONDER LOOKS AMAZING!
THE INTERIOR IS AWFUL
GROWING CITIES WILL SPREAD IN AN UNSUSTAINABLE WAY IF LEFT UN-CHECKED!
WHAT THIS CITY NEEDS IS EFFICIENT RADIATORS
THE CLEANEST ENERGY... IS THE ENERGY YOU DON'T USE
CREATING SMALL EFFICIENCIES IN POWER SUPPLIES
THESE EFFICIENCIES SAVE HUGE AMOUNTS OF ENERGY AND CO2
ON A GLOBAL SCALE!
IMAGINE WHAT A 30% REDUCTION WOULD DO!
THERE ARE MANY WAYS TO CREATE SUSTAINABLE CITIES DESIGNERS CAN HELP
WE CAN MAKE CITIES THAT ABSORB CARBON
WE COULD STOP THIS IF WE HAD A KNOWLEDGE ECONOMY!
INFORMATION IS NOT KNOWLEDGE!
YOU MUST THINK OF CITIES AS CO-CREATIONAL PLATFORMS
WE WOULD HAVE TO WORK WITH THE GOVERNMENT
SUSTAINABLE URBAN MOBILITY
WE DESIGNED HOUSES LIKE WE DESIGN CITIES....
THE CAR WOULD BE IN THE LIVING ROOM!
THE SHARING ECONOMY WILL BE A BIG PART OF THE CARS FUTURE...
WHAT THIS TOWN NEEDS IS A MONORAIL
WE MUST GET CARS OUT OF THE CITY!
DID YOU KNOW BUILDING ROADS IS THE MOST EXPENSIVE TRANSPORT SOLUTION
I DO NOW
MEANWHILE IN OSLO
IT'S GETTING HARDER TO PARK THIS THING
PARKING
THEY SHOULD MAKE CITIES SO PEOPLE DON'T HAVE TO COMMUTE
ALL CITIES SHOULD HAVE A STRONG PUBLIC TRANSPORT BACKBONE
SUSTAINABILITY GOALS CAN MAKE A CITY A NICER PLACE TO LIVE!
FOSSIL CARS ARE BEING TARGETED
ELECTRIC CARS WILL GROW WITH THE RIGHT INCENTIVES
6th Annual SUSTAINABLE INNOVATION FORUM 2015
UNEP CLIMATEACTION
GRAPHIC RECORDING BY CHRIS SHIPTON

Linear Regression, Lasso and Ridge

Lesson Goals:

- Understand best subset selection and stepwise selection methods for reducing the number of predictor variables in regression.
- Indirectly estimate test error by adjusting training error to account for bias due to overfitting (C_p, AIC, BIC, adjusted R^2).
- Directly estimate test error using validation set approach and cross-validation approach.
- Understand and know how to perform ridge regression and the lasso as shrinkage (regularization) methods.
- Understand and know how to perform principal components regression and partial least squares as dimension reduction methods.
- Learn considerations for high-dimensional settings.



Improving the Linear Model

- We may want to improve the simple linear model by replacing OLS estimation with some alternative fitting procedure.

- Why use an alternative fitting procedure?
  - Prediction Accuracy
  - Model Interpretability

Prediction Accuracy

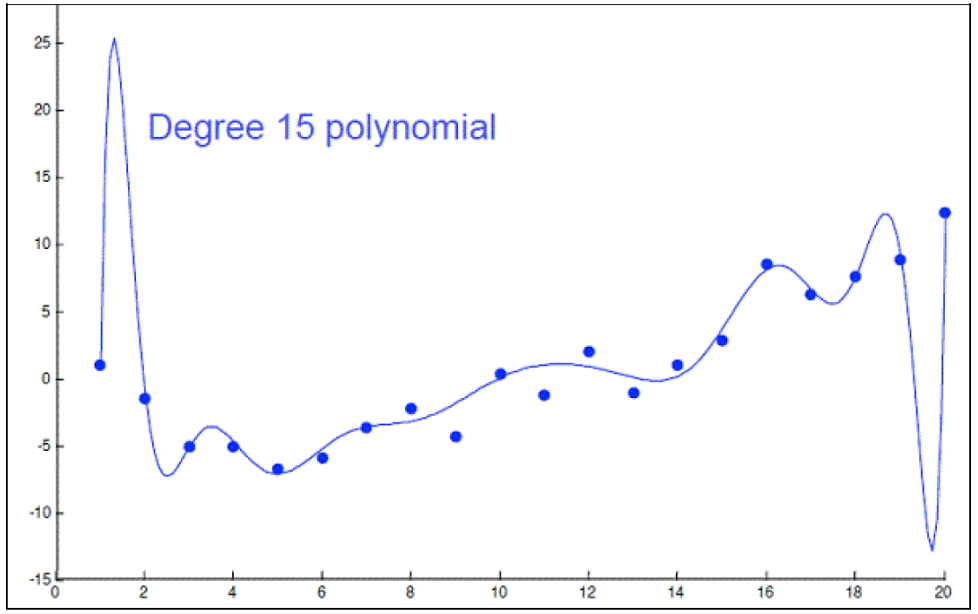
- The OLS estimates have relatively low bias and low variability especially when the relationship between the response and predictors is linear and $n >> p$.

- If n is not much larger than p, then the OLS fit can have high variance and may result in over fitting and poor estimates on unseen observations.

- If $p > n$, then the variability of the OLS fit increases dramatically, and the variance of these estimates in infinite.

Feature/Variable Selection

- Carefully selected features can improve model accuracy, but adding too many can lead to overfitting.
  - Overfitted models describe random error or noise instead of any underlying relationship.
  - They generally have poor predictive performance on test data.



- For instance, we can use a 15-degree polynomial function to fit the following data so that the fitted curve goes nicely through the data points.
- However, a brand new dataset collected from the same population may not fit this particular curve well at all.

Feature/Variable Selection (cont.)

- Subset Selection
  - Identify a subset of the *p* predictors that we believe to be related to the response; then, fit a model using OLS on the reduced set.
  - Methods: best subset selection, stepwise selection
- Shrinkage (Regularization)
  - Involves shrinking the estimated coefficients toward zero relative to the OLS estimates; has the effect of reducing variance and performs variable selection.
  - Methods: ridge regression, lasso
- Dimension Reduction
  - Involves projecting the *p* predictors into a *M*-dimensional subspace, where $M < p$, and fit the linear regression model using the *M* projections as predictors.
  - Methods: principal components regression, partial least squares

Best Subset Selection

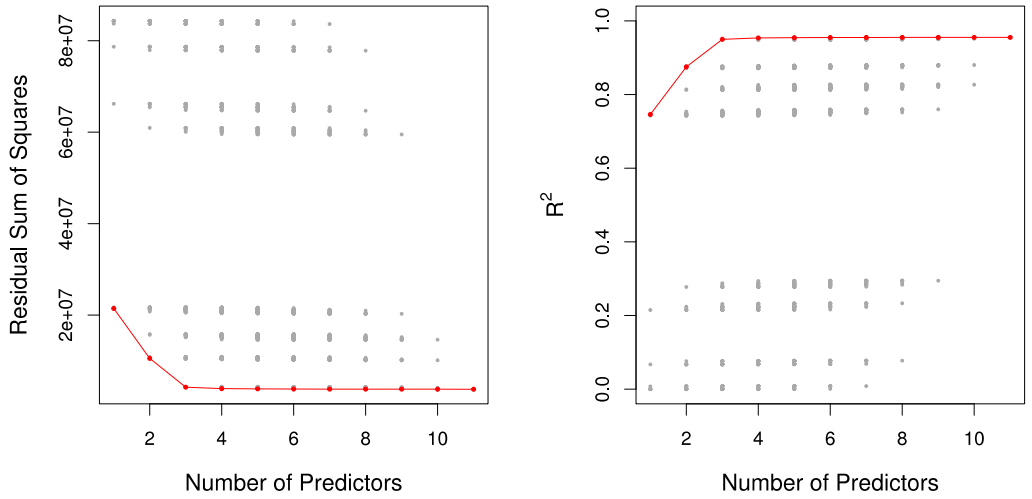
- We fit a separate OLS regression for each possible combination of the p predictors:

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 1, 2, \ldots p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Best Subset Selection (cont.)

- The RSS (R^2) will always decline (increase) as the number of predictors included in the model increases, so they are not very useful statistics for selecting the *best* model.

- The red line tracks the best model for a given number of predictors, according to RSS and R^2



Best Subset Selection (cont.)

- While best subset selection is a simple and conceptually appealing approach, it suffers from computational limitations.
- The number of possible models that must be considered grows rapidly as *p* increases.
- Best subset selection becomes computationally *infeasible* for value of *p* greater than around 40.

Stepwise Selection

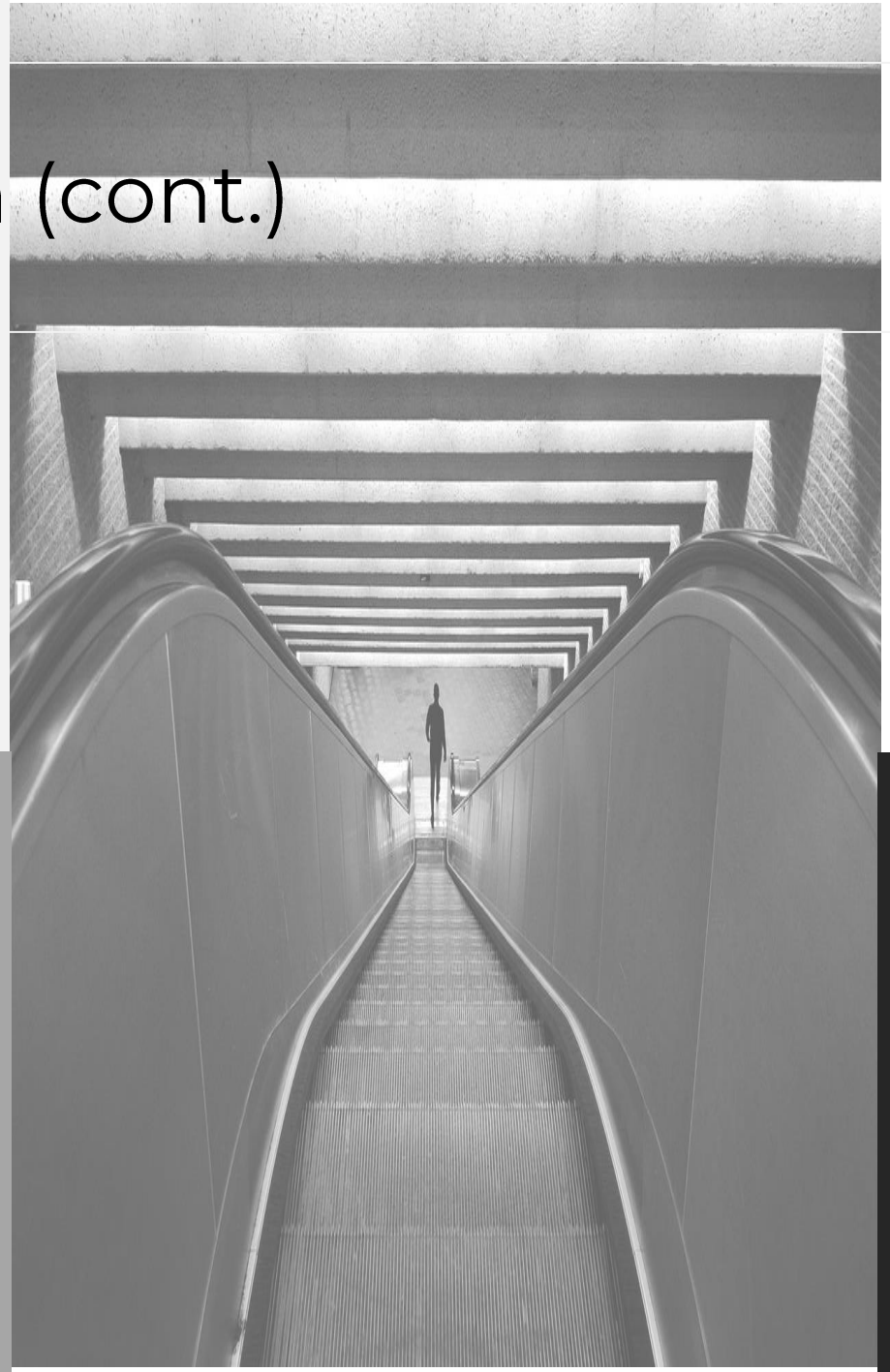
- For computational reasons, best subset selection cannot be applied with very large *p*.

- The larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data.

- An enormous search space can lead to overfitting and high variance of the coefficient estimates.

Stepwise Selection (cont.)

More attractive methods include:



Forward Stepwise Selection

Begins with a null OLS model containing no predictors, and then adds one predictor at a time that improves the model the most until no further improvement is possible.

Backward Stepwise Selection

Begins with a full OLS model containing all predictors, and then deletes one predictor at a time that improves the model the most until no further improvement is possible.

Forward Stepwise Selection

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   - 2.1 Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   - 2.2 Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Backward Stepwise Selection

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   - 2.1 Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   - 2.2 Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

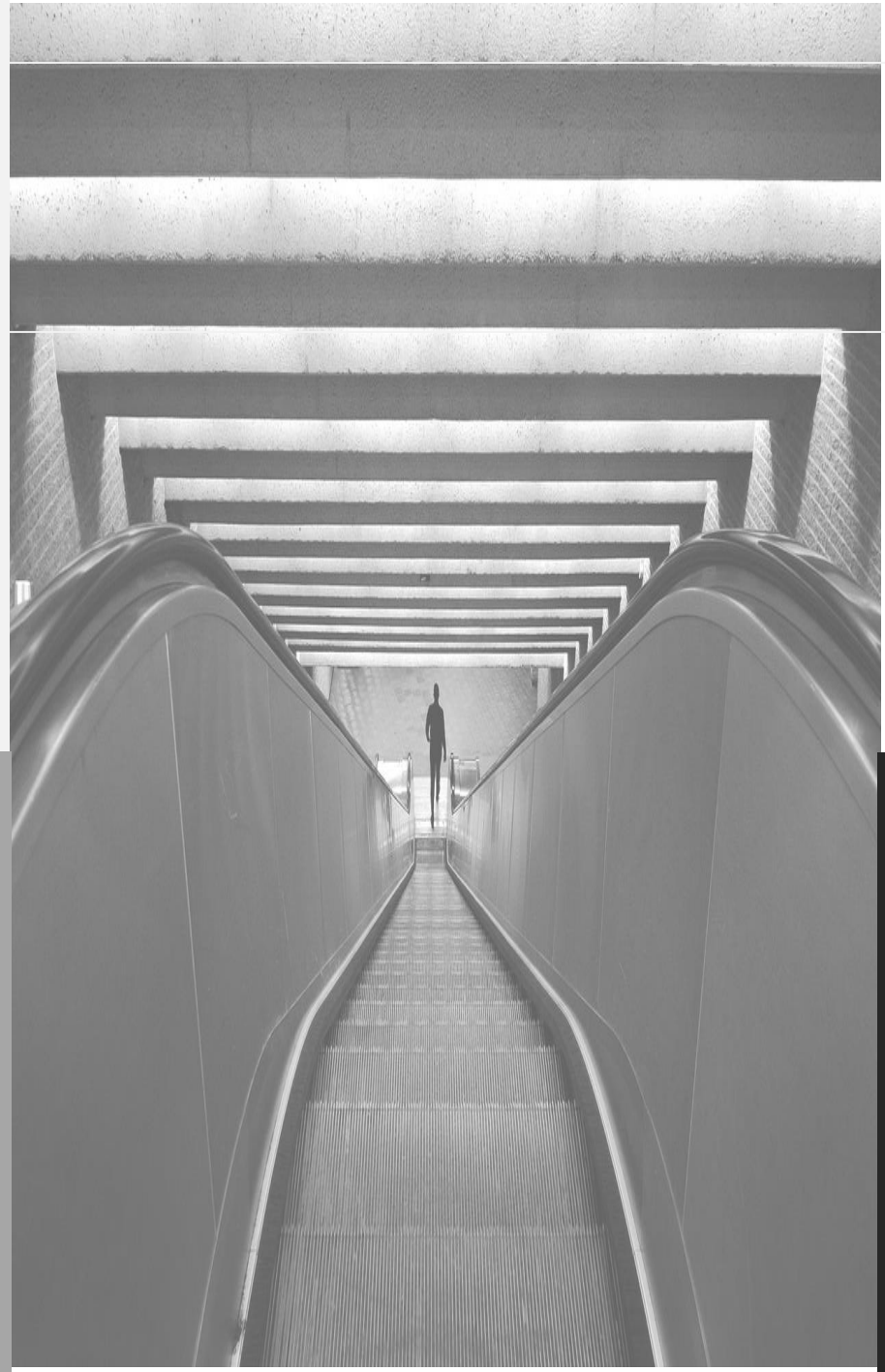
Stepwise Selection (cont.)

- Both forward and backward stepwise selection approaches search through only $1 + p(p + 1)/2$ models, so they can be applied in settings where p is too large to apply best subset selection.
- Both of these stepwise selection methods are *not* guaranteed to yield the best model containing a subset of the p predictors.
- Forward stepwise selection can be used even when $n < p$, while backward stepwise selection requires that $n > p$.
- There is a *hybrid* version of these two stepwise selection methods.

Choosing the Optimal Model

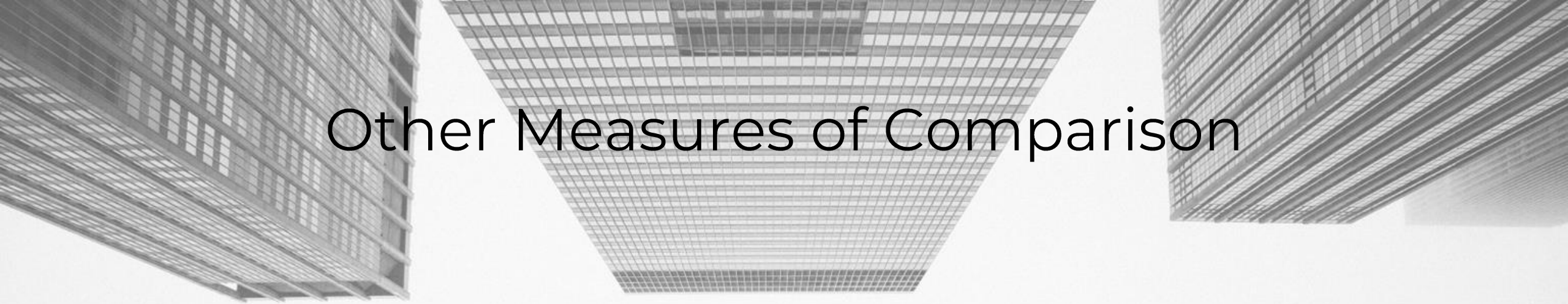
- The model containing all the predictors will always have the smallest RSS and the largest R^2, since these quantities are related to the training error.
- We wish to choose a model with low test error, not a model with low training error. Recall that training error is usually a poor estimate of test error.
- Thus, RSS and R^2 are not suitable for selecting the *best* model among a collection of models with different numbers of predictors.

Estimating Test Error



- We can indirectly estimate test error by making an *adjustment* to the training error to account for the bias due to overfitting.

- We can *directly* estimate the test error, using either a validation set approach or a cross-validation approach.

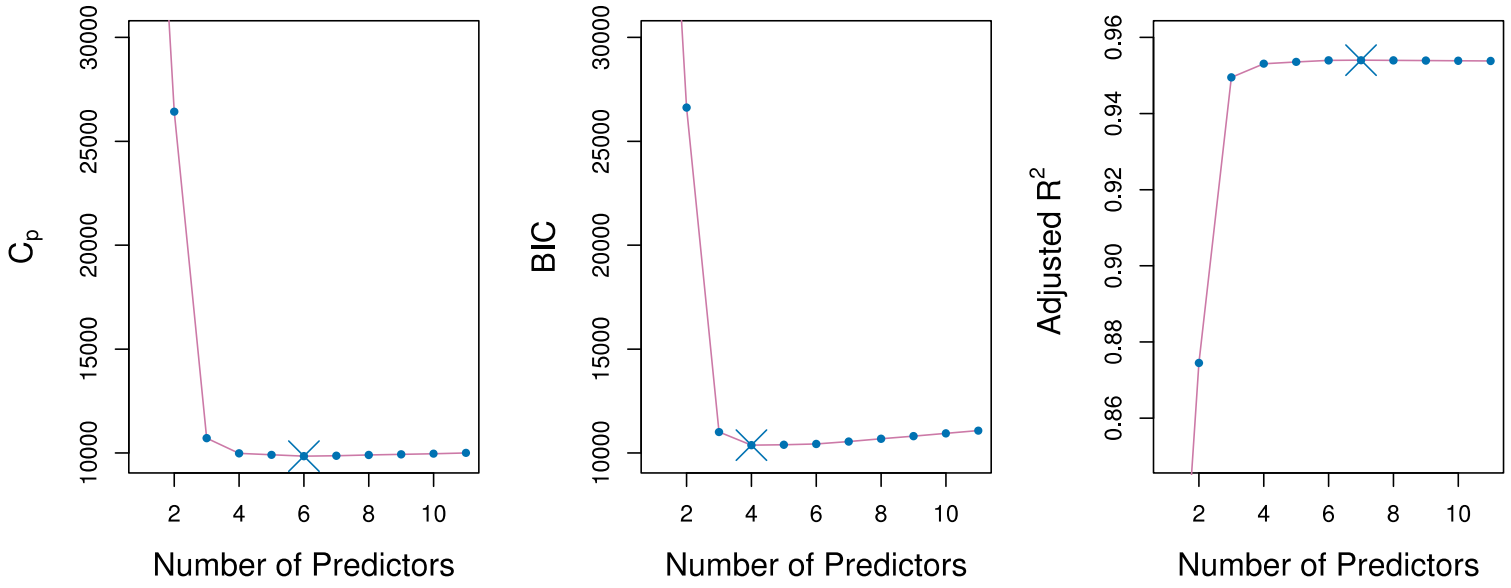


Other Measures of Comparison

- To compare different models, we can use other approaches:
  - Adjusted R^2
  - AIC (Akaike information criterion)
  - BIC (Bayesian information criterion)
  - Mallow's C_p (equivalent to AIC for linear regression)
- These techniques adjust the training error for the model size, and can be used to select among a set of models with different numbers of variables.
- These methods add penalty to RSS for the number of predictors in the model.

Credit Data: C_p, BIC, and Adjusted R^2

- A small value of C_p and BIC indicates a low error, and thus a better model.
- A large value for the Adjusted R^2 indicates a better model.



Mallow's C_p

- For a fitted OLS model containing *d* predictors, the C_p estimate of test MSE:

$$C_p = \frac{1}{n}\left(\mathrm{RSS} + 2d\hat{\sigma}^2\right)$$

where $\hat{\sigma}^2$ is an estimate of the variance of the error ε associated with each response measurement.

- Here, a penalty is added to the training RSS in order to adjust for the fact that the training error tends to underestimate the test error.

Akaike Information Criterion (AIC)

- Defined for a large class of models fit by maximum likelihood.

$$\mathrm{AIC} = -2 \log L + 2 \cdot d$$

where *L* is the maximized value of the likelihood function for the estimated model.

- In the case of the linear model with Gaussian errors, MLE and OLS are the same things; thus, C_p and AIC are equivalent.

Bayesian Information Criterion (BIC)

- BIC will tend to take on a small value for a model with a low test error, and so generally we select the model that has the lowest BIC value.

$$\text{BIC} = \frac{1}{n}\left(\text{RSS} + \log(n) d\hat{\sigma}^2\right)$$

- Since log *n* > 2 for an *n* > 7, the BIC statistic generally places a heavier penalty on models with many variables, and hence results in the selection of smaller models than C_p.

- Notice that BIC replaces the $2d\hat{\sigma}^2$ used by C_p with a $\log(n)d\hat{\sigma}^2$ term, where *n* is the number of observations.

Adjusted R^2

- For an OLS model with *d* variables, the adjusted R^2 is calculated:

$$\text{Adjusted } R^2 = 1 - \frac{\text{RSS}/(n-d-1)}{\text{TSS}/(n-1)}$$

where TSS is the total sum of squares.

- Unlike the other statistics, a large value of adjusted R^2 indicates a model with a small test error.

- The adjusted R^2 statistics *pays a price* for the inclusion of unnecessary variables in the model.

Validation and Cross-Validation

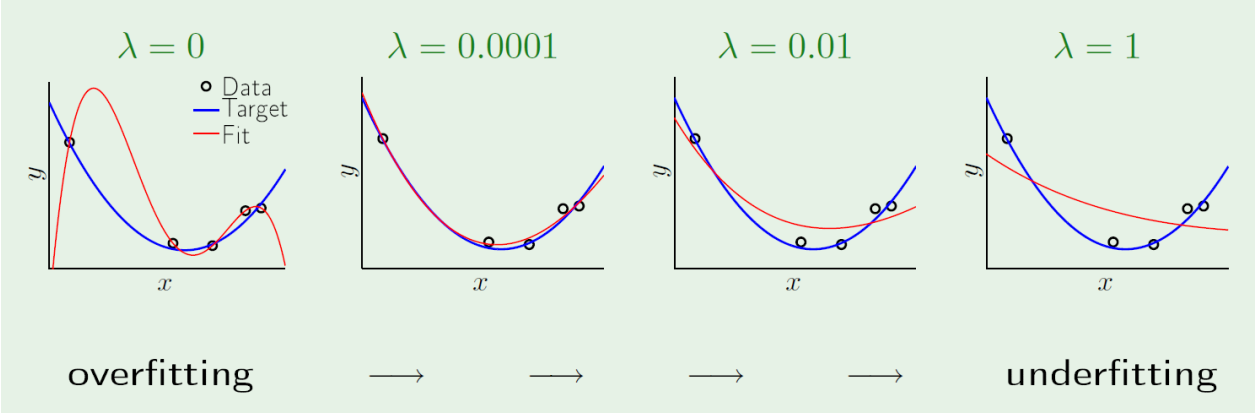
- Each of the procedures returns a sequence of models indexed by model size k = 0, 1, 2, … Our job here is to select $\hat{k}$.
- We compute the validation set error or the CV error for each model under consideration, and then select the k for which the resulting estimated test error is smallest.
- This procedure provides a direct estimate of the test error, and it can also be used in a wider range of model selection tasks.
- We can also select a model using the *one-standard-error rule*.

Shrinkage (Regularization) Methods

- The subset selection methods use OLS to fit a linear model that contains a subset of the predictors.
- As an alternative, we can fit a model containing all *p* predictors using a technique that constrains or *regularizes* the coefficient estimates (i.e. *shrinks* the coefficient estimates towards zero).
- It may not be immediately obvious why such a constraint should improve the fit, but it turns out that *shrinking* the coefficient estimates can significantly reduce their variance.

Shrinkage (Regularization) Methods (cont.)

- Regularization is our first weapon to combat overfitting.
- It constrains the machine learning algorithm to improve out-of-sample error, especially when noise is present.
- Look at what a little regularization can do:



Ridge Regression

- Recall that the OLS fitting procedure estimates the beta coefficients using the values that minimize:

$$\text{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

- Ridge regression is similar to OLS, except that the coefficients are estimated by minimizing a slightly different quantity:

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \text{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2$$

where $\lambda \geq 0$ is a *tuning parameter*, to be determined separately.

Ridge Regression (cont.)

- Note that $\lambda \geq 0$ is a complexity parameter that controls the amount of shrinkage.
- The idea of penalizing by the sum-of-squares of the parameters is also used in neural networks, where it is known as *weight decay*.
- An equivalent way to write the ridge problem is:

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\text{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 \leq t,$$

Ridge Regression (cont.)

- The effect of this equation is to add a shrinkage penalty of the form

$$\lambda \sum_{j=1}^{p} \beta_j^2,$$

where the tuning parameter λ is a positive value.

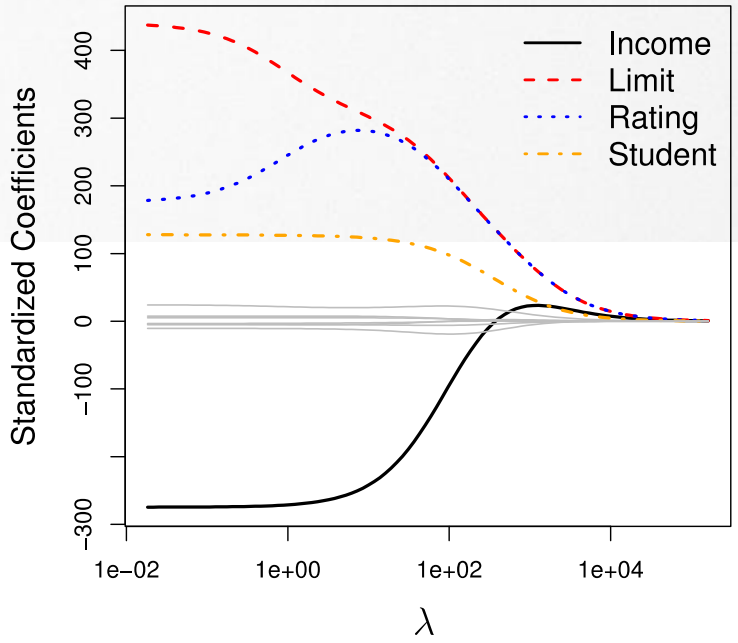
- This has the effect of shrinking the estimated beta coefficients towards zero. It turns out that such a constraint should improve the fit, because shrinking the coefficients can significantly reduce their variance.

- Note that when λ = 0, the penalty term as no effect, and ridge regression will procedure the OLS estimates. Thus, selecting a good value for λ is critical (can use cross-validation for this).

Ridge Regression (cont.)

- As λ increases, the standardized ridge regression coefficients shrinks towards zero.

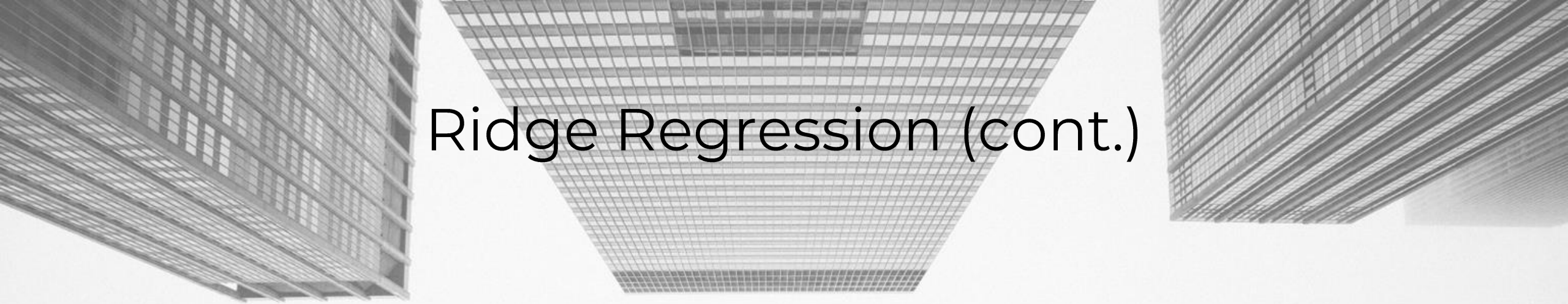
- Thus, when λ is extremely large, then all of the ridge coefficient estimates are basically zero; this corresponds to the *null model* that contains no predictors.



Ridge Regression (cont.)

- The standard OLS coefficient estimates are *scale equivariant*.
- However, the ridge regression coefficient estimates can change *substantially* when multiplying a given predictor by a constant, due to the sum of squared coefficients term in the penalty part of the ridge regression objective function.
- Thus, it is best to apply ridge regression after *standardizing the predictors*:

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}$$



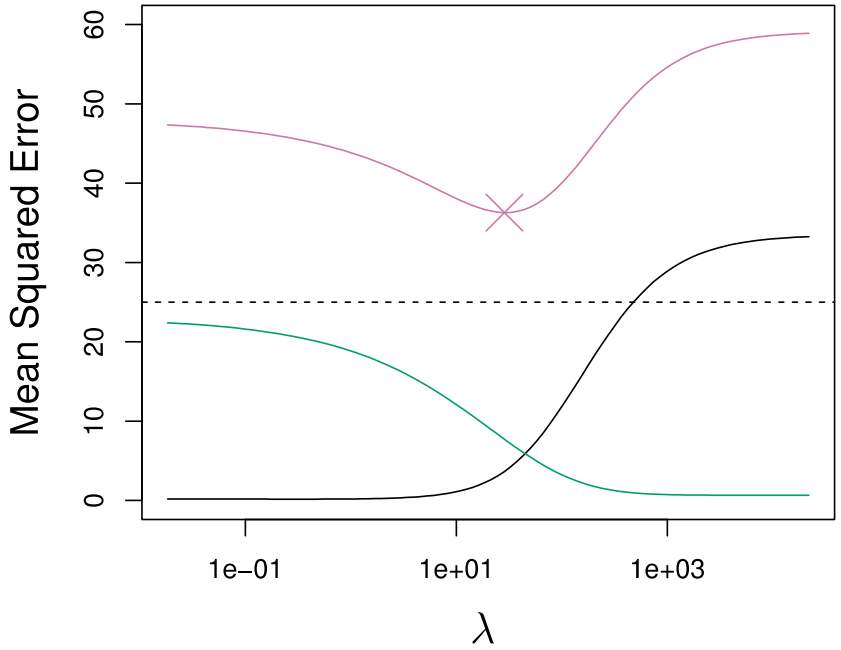
Ridge Regression (cont.)

- It turns out that the OLS estimates generally have low bias but can be highly variable. In particular when n and p are of similar size or when $n < p$, then the OLS estimates will be extremely variable

- The penalty term makes the ridge regression estimates *biased* but can also substantially reduce variance

- As a result, there is a bias/variance trade-off.

Ridge Regression (cont.)



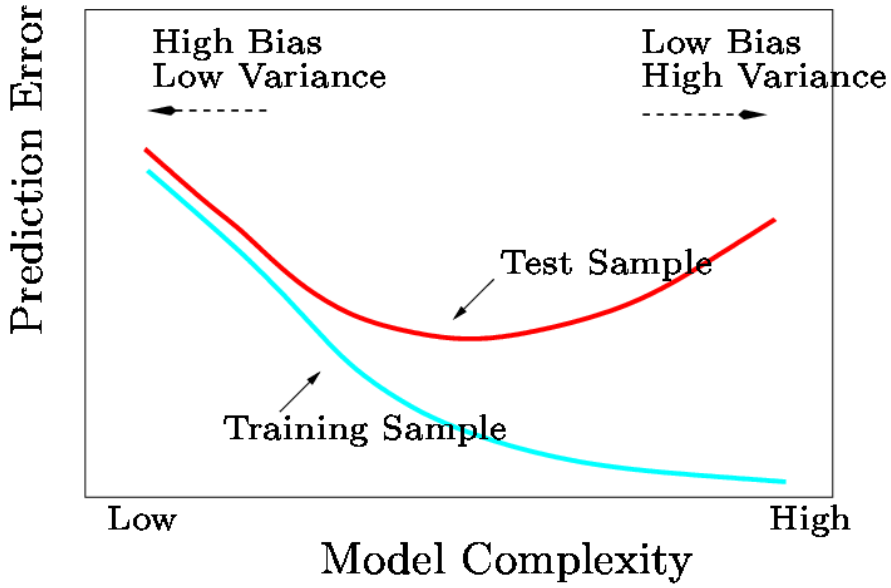
- Black = Bias
- Green = Variance
- Purple = MSE

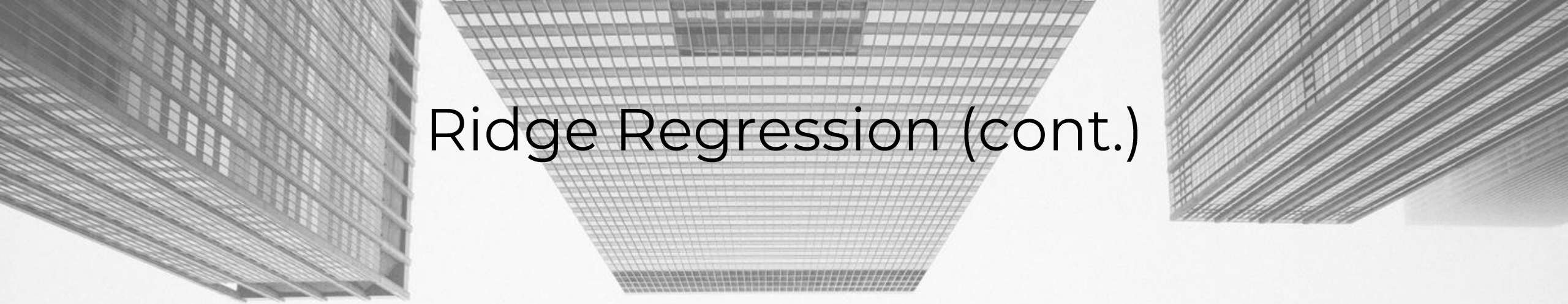
- Increased λ leads to increased bias but decreased variance

Ridge Regression (cont.)

- In general, the ridge regression estimates will be more biased than the OLS ones but have lower variance.

- Ridge regression will work best in situations where the OLS estimates have high variance.





Ridge Regression (cont.)

Computational Advantages of Ridge Regression

- If p is large, then using the best subset selection approach requires searching through enormous numbers of possible models.

- With ridge regression, for any given λ we only need to fit one model and the computations turn out to be very simple.

- Ridge regression can even be used when $p > n$, a situation where OLS fails completely (i.e. OLS estimates do not even have a unique solution).

Ridge Regression (cont.)

- In matrix form:

$$\mathrm{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta,$$

the ridge regression solutions are easily seen to be

$$\hat{\beta}^{\mathrm{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y},$$

- The solution adds a positive constant to the diagonal of $\mathbf{X}^T\mathbf{X}$ before inversion (making the problem non-singular).

- The *singular value decomposition* (SVD) of the centered matrix **X** gives us some additional insight into the nature of ridge regression.

Ridge Regression (cont.)

- The SVD of the *N* x *p* matrix **X** has the form $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$
- Here, **U** and **V** are *N* x *p* and *p* x *p* orthogonal matrices, with the columns of **U** spanning the column space of **X**, and the columns of **V** spanning the row space.
- **D** is a *p* x *p* diagonal matrix, with diagonal entries $d_1 \geq d_2 \geq \cdots d_p \geq 0$ called singular values of **X**.
- If one or more values $d_j = 0$, **X** is singular.

Ridge Regression (cont.)

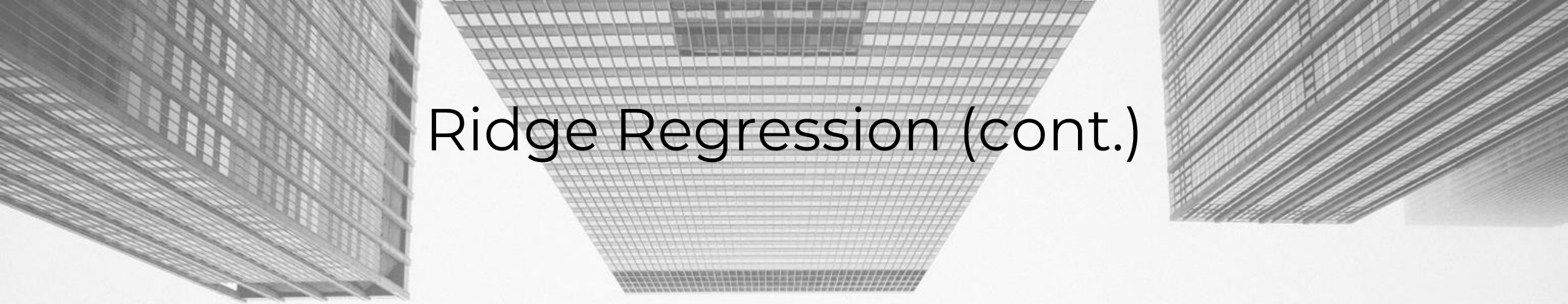
- Using SVD, we can write the OLS fitted vector as:

$$\begin{aligned}\mathbf{X}\hat{\beta}^{\text{ls}} &= \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\ &= \mathbf{U}\mathbf{U}^T\mathbf{y},\end{aligned}$$

- The ridge regression solutions are:

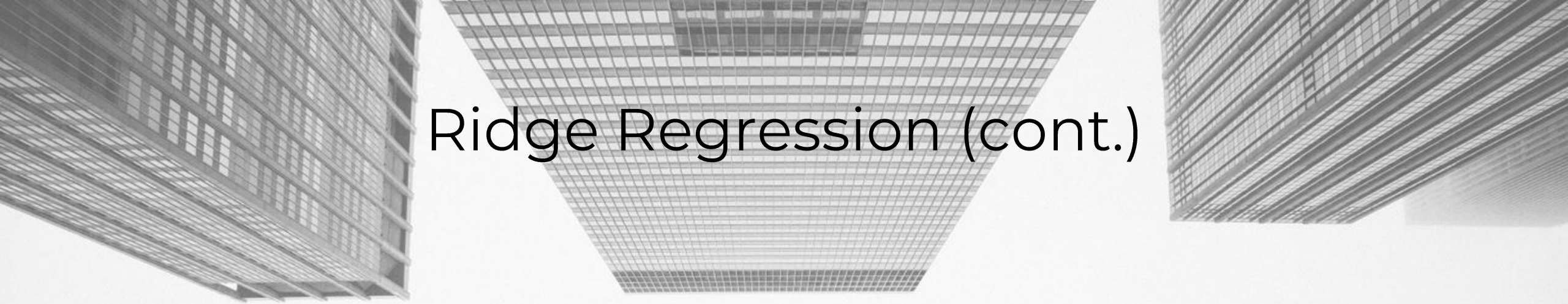
$$\begin{aligned}\mathbf{X}\hat{\beta}^{\text{ridge}} &= \mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} \\ &= \mathbf{U}\,\mathbf{D}(\mathbf{D}^2+\lambda\mathbf{I})^{-1}\mathbf{D}\,\mathbf{U}^T\mathbf{y} \\ &= \sum_{j=1}^{p}\mathbf{u}_j\frac{d_j^2}{d_j^2+\lambda}\mathbf{u}_j^T\mathbf{y},\end{aligned}$$

where $\mathbf{u}_j$ are the columns of $\mathbf{U}$.



Ridge Regression (cont.)

- Like linear regression, ridge regression computes the coordinates of **y** with respect to the orthonormal basis **U**.
- It then *shrinks* these coordinates by the factor $\frac{d_j^2}{d_j^2+\lambda}$.
- This means that a greater amount of shrinkage is applied to the coordinates of basis vectors with smaller d_j^2.
- The SVD of the centered matrix **X** is another way of expressing the *principal components* of the variables in **X**.



Ridge Regression (cont.)

- Thus, we have $\mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{D}^2\mathbf{V}^T$, which is the *eigen decomposition* of $\mathbf{X}^T\mathbf{X}$.
- The eigenvectors v_j (columns of **V**) are also called the *principal components* directions of **X**.
- The first principal component direction v_1 has the property that $\mathbf{z}1 = \mathbf{X}v_1$ has the largest sample variance amongst all normalized linear combinations of the columns of **X**.
- The small singular values d_j correspond to directions in the column space of **X** having small variance, and ridge regression shrinks these directions the most.

The Lasso

- One significant problem of ridge regression is that the penalty term will never force any of the coefficients to be exactly zero.
- Thus, the final model will include all *p* predictors, which creates a challenge in model interpretation
- A more modern machine learning alternative is the *lasso*.
- The lasso works in a similar way to ridge regression, except it uses a different penalty term that shrinks some of the coefficients exactly to zero.

The Lasso (cont.)

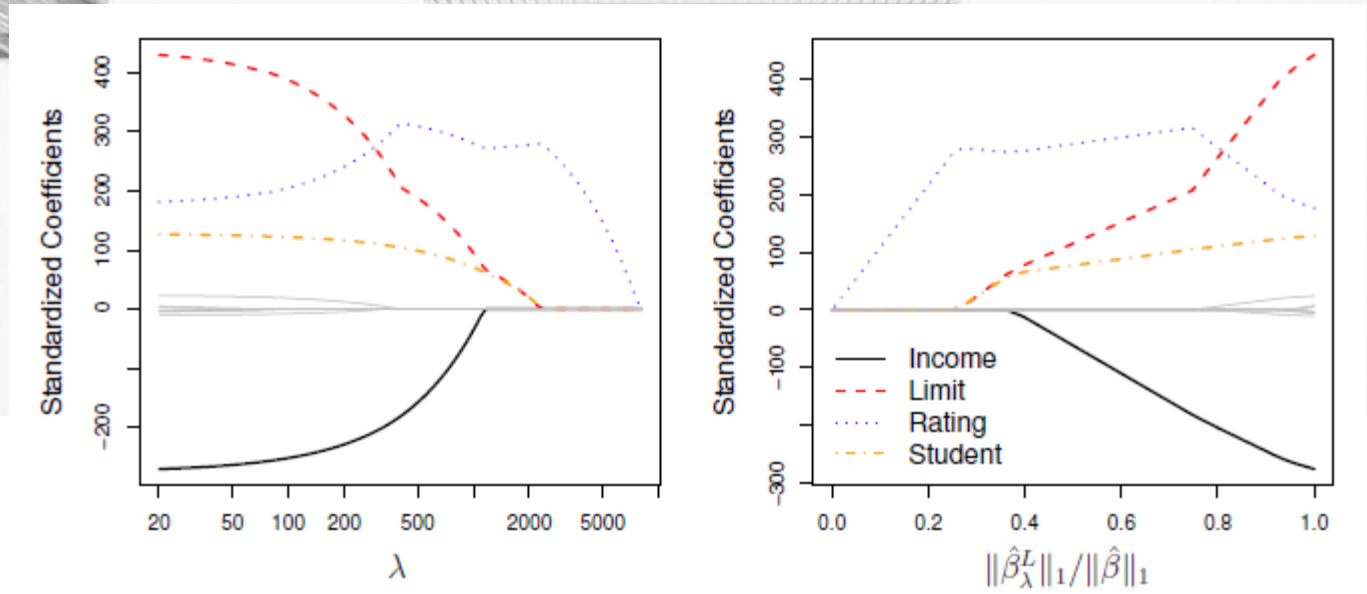
- The lasso coefficients minimize the quantity:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|$$

- The key difference from ridge regression is that the lasso uses an ℓ_1 penalty instead of an ℓ_2, which has the effect of forcing some of the coefficients to be exactly equal to zero when the tuning parameter λ is sufficiently large.

- Thus, the lasso performs variable/feature selection.

The Lasso (cont.)



- When λ = 0, then the lasso simply gives the OLS fit.
- When λ becomes sufficiently large, the lasso gives the null model in which all coefficient estimates equal zero.

The Lasso (cont.)

- One can show that the lasso and ridge regression coefficient estimates solves the problems:

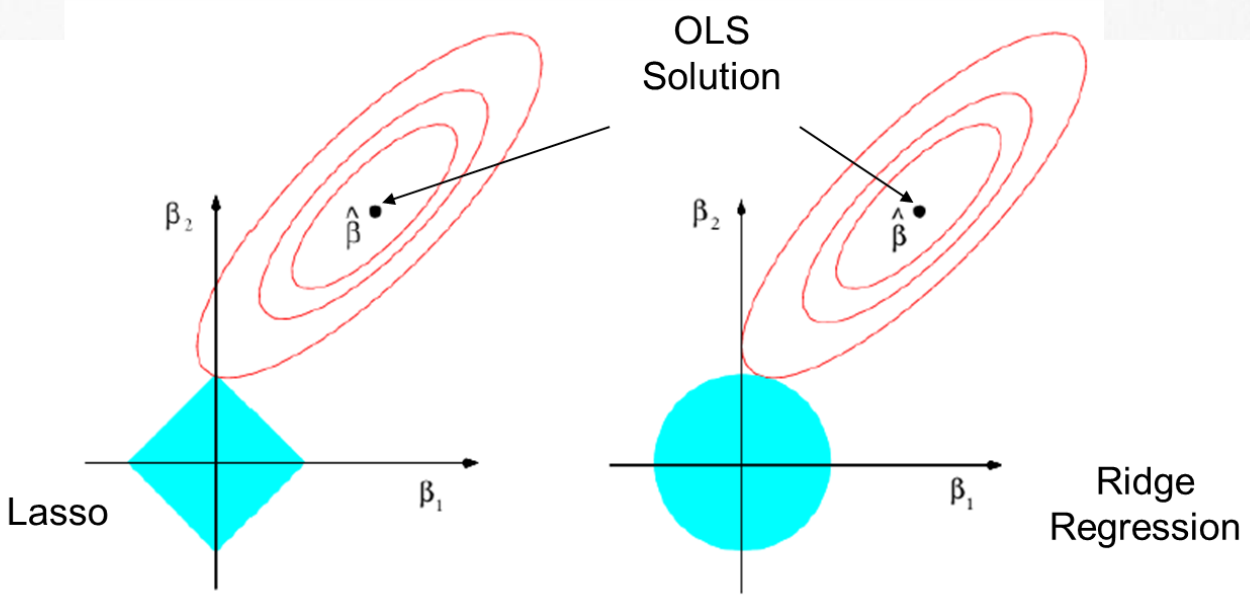
$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

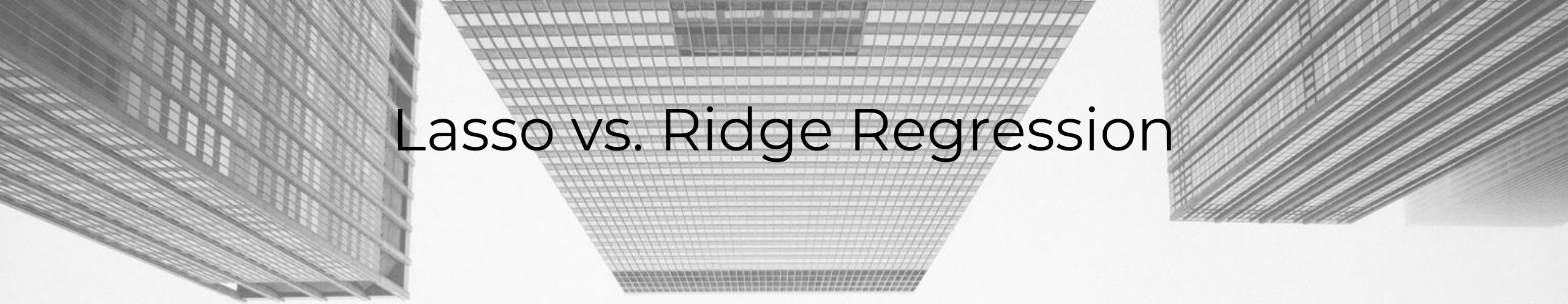
and

$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s,$$

The Lasso (cont.)

- The lasso and ridge regression coefficient estimates are given by the first point at which an ellipse contacts the constraint region.





Lasso vs. Ridge Regression

- The lasso has a major advantage over ridge regression, in that it produces simpler and more interpretable models that involved only a subset of predictors.
- The lasso leads to qualitatively similar behavior to ridge regression, in that as λ increases, the variance decreases and the bias increases.
- The lasso can generate more accurate predictions compared to ridge regression.
- Cross-validation can be used in order to determine which approach is better on a particular data set.

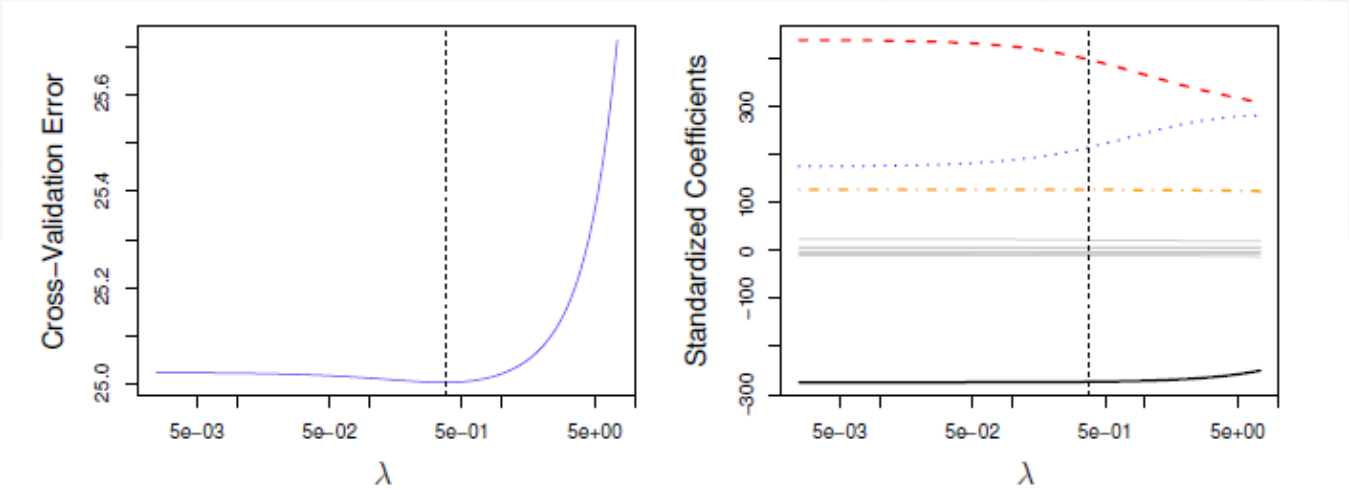
Selecting the Tuning Parameter λ

- As for subset selection, for ridge regression and lasso we require a method to determine which of the models under consideration in best; thus, we required a method selecting a value for the tuning parameter λ or equivalently, the value of the constraint s.

- Select a grid of potential values; use cross-validation to estimate the error rate on test data (for each value of λ) and select the value that gives the smallest error rate.

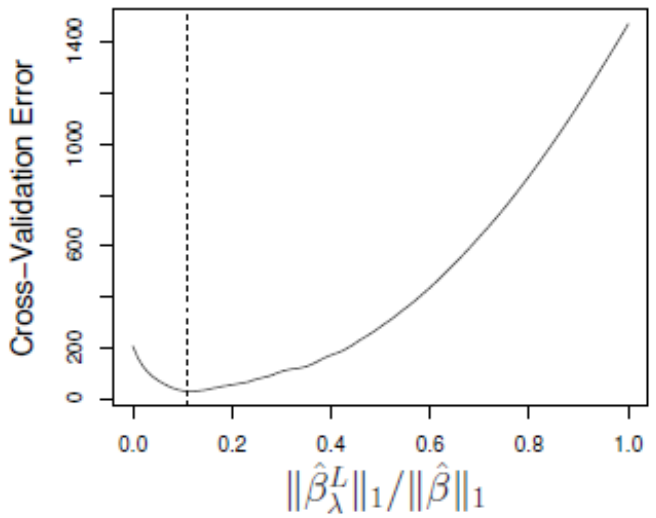
- Finally, the model is re-fit using all of the variable observations and the selected value of the tuning parameter λ.

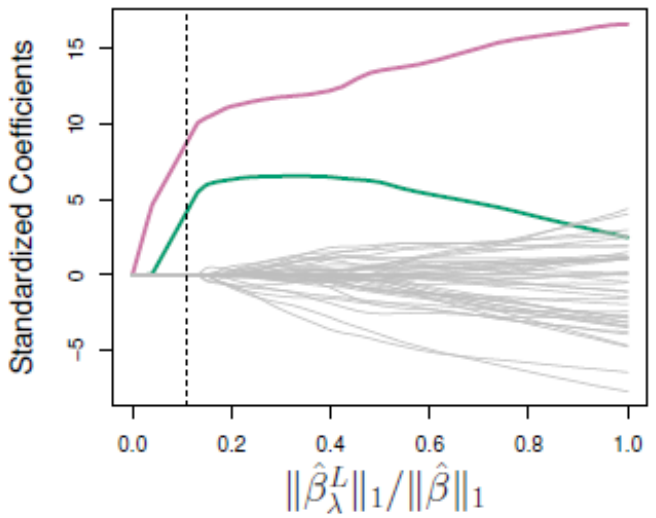
Selecting the Tuning Parameter λ: Credit Data Example



Left: *Cross-validation errors that result from applying ridge regression to the* `Credit` *data set with various values of* λ.
Right: *The coefficient estimates as a function of* λ. *The vertical dashed lines indicates the value of* λ *selected by cross-validation.*

Selecting the Tuning Parameter λ: Simulated Data Example





Left: *Ten-fold cross-validation MSE for the lasso, applied to the sparse simulated data set from Slide 39.* Right: *The corresponding lasso coefficient estimates are displayed. The vertical dashed lines indicate the lasso fit for which the cross-validation error is smallest.*

Dimension Reduction

- The methods we have discussed so far have involved fitting linear regression models, via OLS or a shrunken approach, using the original predictors.
- We now explore a class of approaches that *transform* the predictors and then fit an OLS model using the transformed variables.
- We refer to these techniques as *dimension reduction* methods.

Dimension Reduction (cont.)

- Let Z_1, Z_2,…,Z_M represent $M < p$ *linear combination* of our original p predictors. That is,

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$$

for some constants $\phi_{m1}, \ldots, \phi_{mp}$.

- We can then fit an OLS linear regression model,

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i, \quad i = 1, \ldots, n,$$

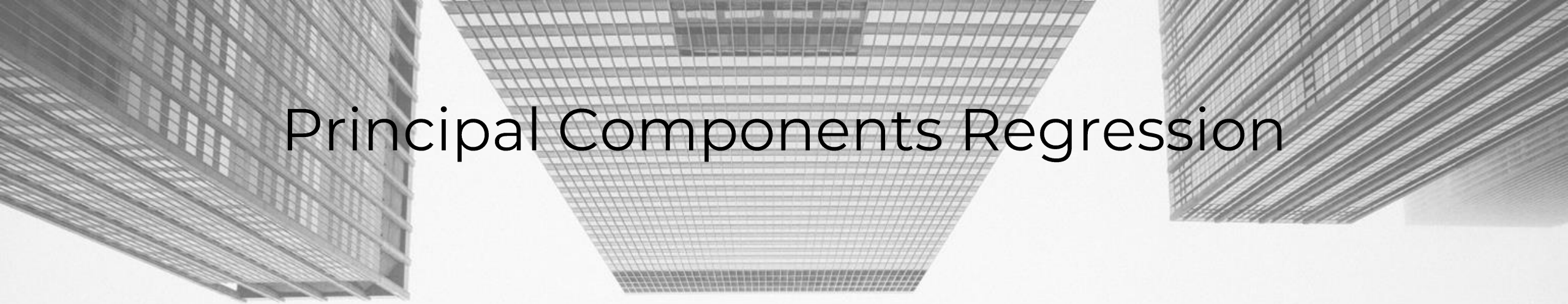
Dimension Reduction (cont.)

- If the constants $\phi_{m1}, \ldots, \phi_{mp}$ are chosen wisely, then such dimension reduction approaches can outperform OLS regression.

- The term *dimension reduction* comes from the fact that this approach reduces the problem of estimating the $p + 1$ coefficients $\beta_0, \ldots, \beta_p$ to the simpler problem of estimating the $M + 1$ coefficients $\theta_0, \ldots, \theta_M$, where $M < p$.

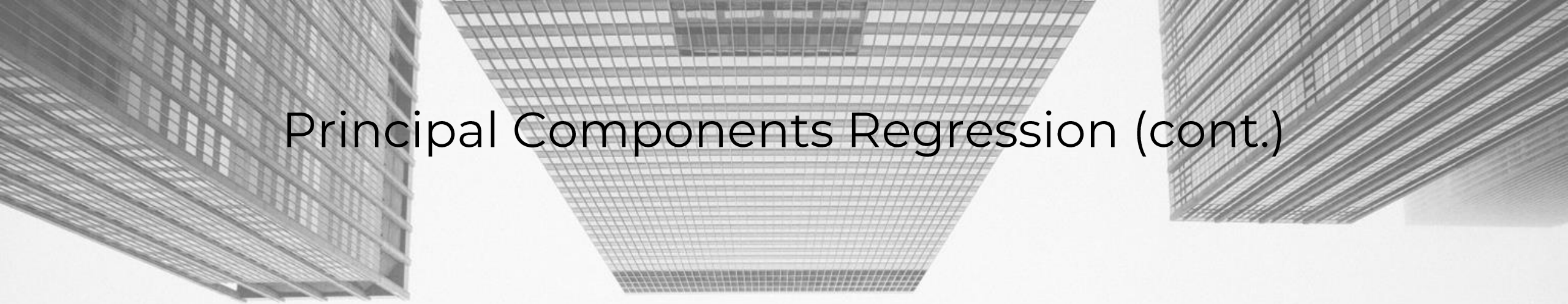
$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{mj} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{mj} x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij}, \qquad \beta_j = \sum_{m=1}^{M} \theta_m \phi_{mj}$$

- This method serves to constrain the estimated β_j coefficients.



Principal Components Regression

- Here, we apply principal components analysis (PCA) to define the linear combinations of the predictors, for use in the regression.
- The *first principal component* is that (normalized) linear combination of the variables with the largest variances.
- The *second principal component* has largest variance, subject to being uncorrelated with the first….etc.
- Thus, with many correlated variables, we replace them with a small set of principal components that capture their joint variation.



Principal Components Regression (cont.)

- The *principal components regression* (PCR) approach involves constructing the first *M* principal components, and then using these components as the predictors in an OLS linear regression model.
- The key idea is that often a small number of principal components suffice to *explain* most of the variability in the data, as well as the relationship with the response.
- We assume that the directions in which $X_1,...,X_p$ show the most variation are the directions that are associated with Y.
- When performing PCR, predictors should be *standardized* prior to generating the principal components.

Principal Components Regression (cont.)

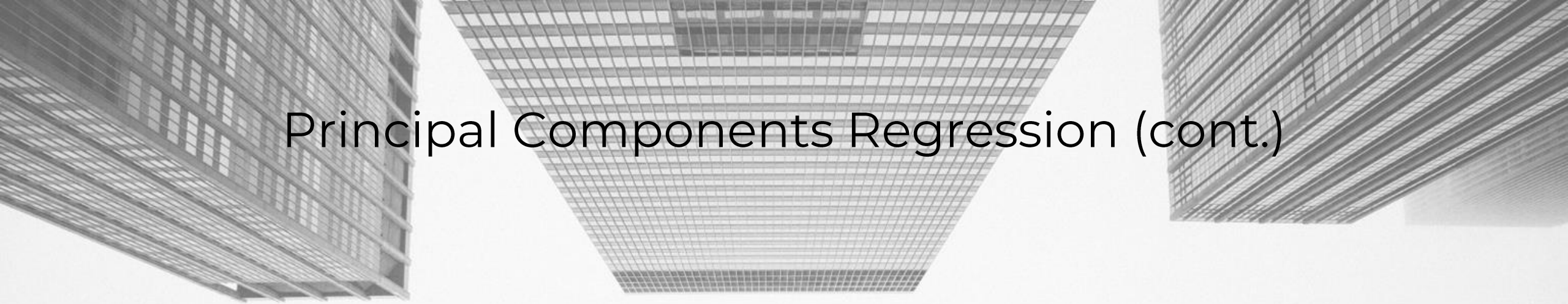
- PCR forms the derived input columns $\mathbf{z}_m = \mathbf{X}v_m$, and then regresses $\mathbf{y}$ on $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_M$ for some $M \leq p$. Since the $\mathbf{z}_m$ are orthogonal, this regression is just a sum of univariate regressions:

$$\hat{\mathbf{y}}^{\text{pcr}}_{(M)} = \bar{y}\mathbf{1} + \sum_{m=1}^{M} \hat{\theta}_m \mathbf{z}_m$$

where $\hat{\theta}_m = \langle \boldsymbol{z}_m, \boldsymbol{y} \rangle / \langle \boldsymbol{z}_m, \boldsymbol{z}_m \rangle$. Since $\mathbf{z}_m$ are each linear combinations of the original $\mathbf{x}_j$, we can express the solution in terms of coefficients of the $\mathbf{x}_j$.

$$\hat{\beta}^{\text{pcr}}(M) = \sum_{m=1}^{M} \hat{\theta}_m v_m$$

- PCR discards the $p - M$ smallest eigenvalue components.

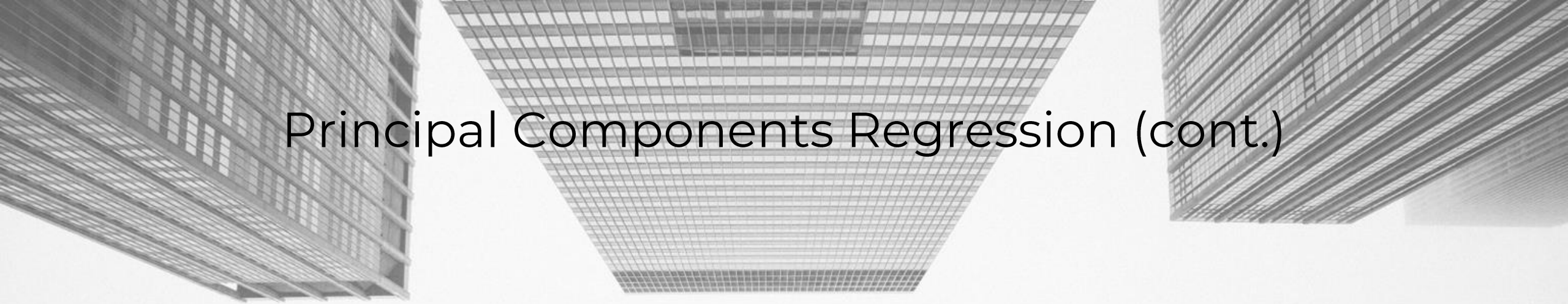


Principal Components Regression (cont.)

- By manually setting the projection onto the principal component directions with small eigenvalues set to 0 (i.e., only keeping the large ones), dimension reduction is achieved.

- PCR is very similar to ridge regression in a certain sense.

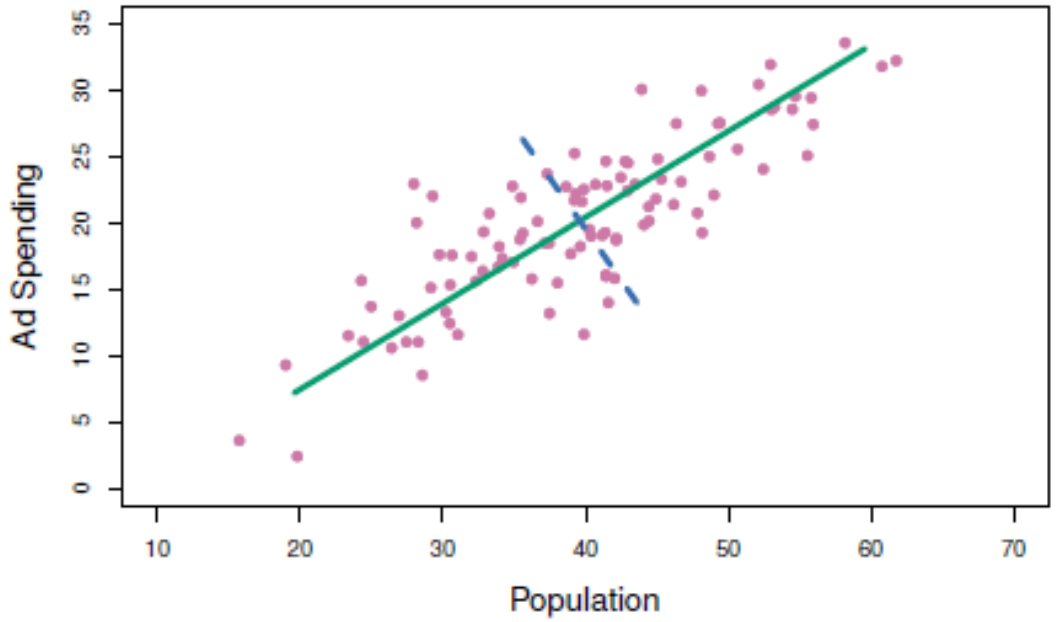
- Ridge regression can be viewed conceptually as projecting the *y* vector onto the principal component directions and then shrinking the projection on each principal component direction.



Principal Components Regression (cont.)

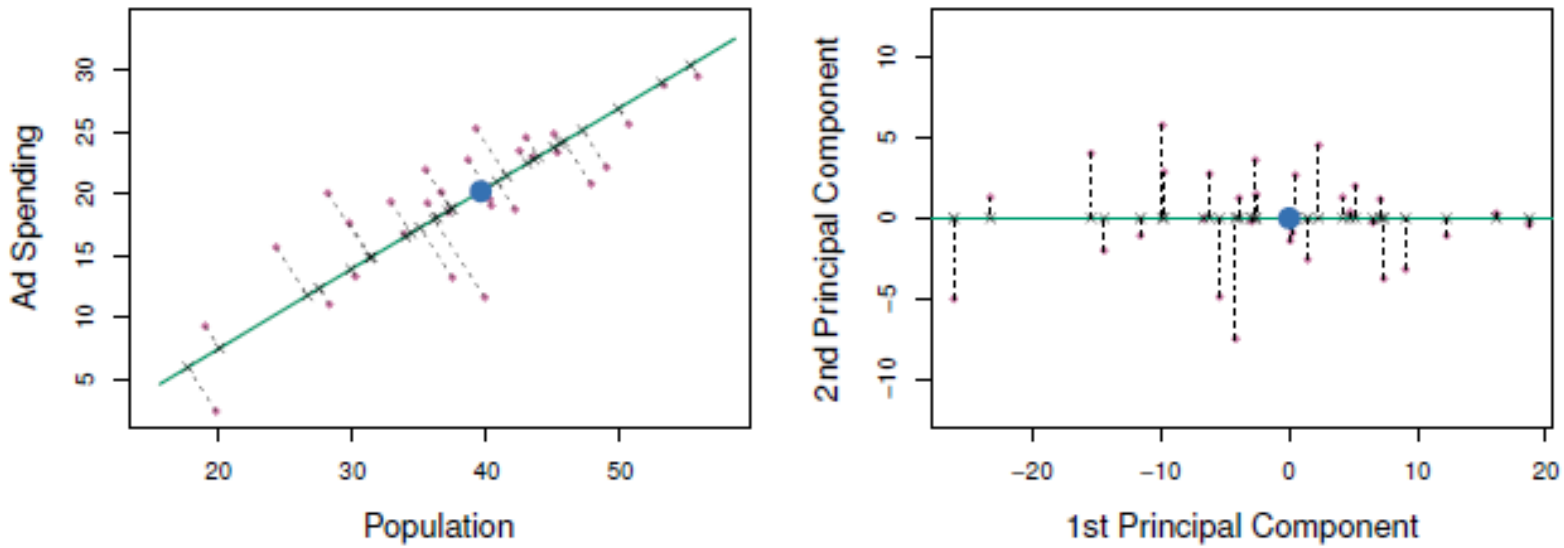
- The amount of shrinkage depends on the variance of that principal component.
- Ridge regression shrinks everything, but it never shrinks anything to zero.
- By contrast, PCR either does not shrink a component at all or shrinks it to zero.

Principal Components Regression (cont.)



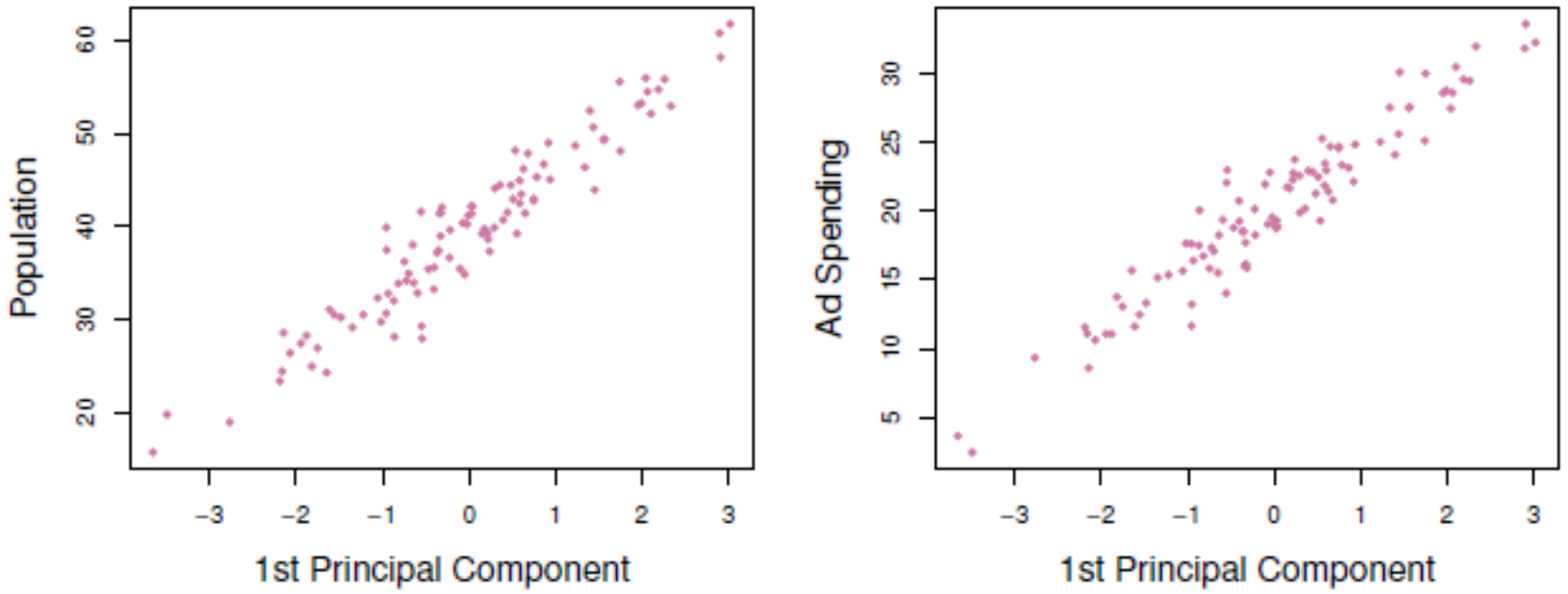
The population size (pop) and ad spending (ad) for 100 *different cities are shown as purple circles. The green solid line indicates the first principal component, and the blue dashed line indicates the second principal component.*

Principal Components Regression (cont.)



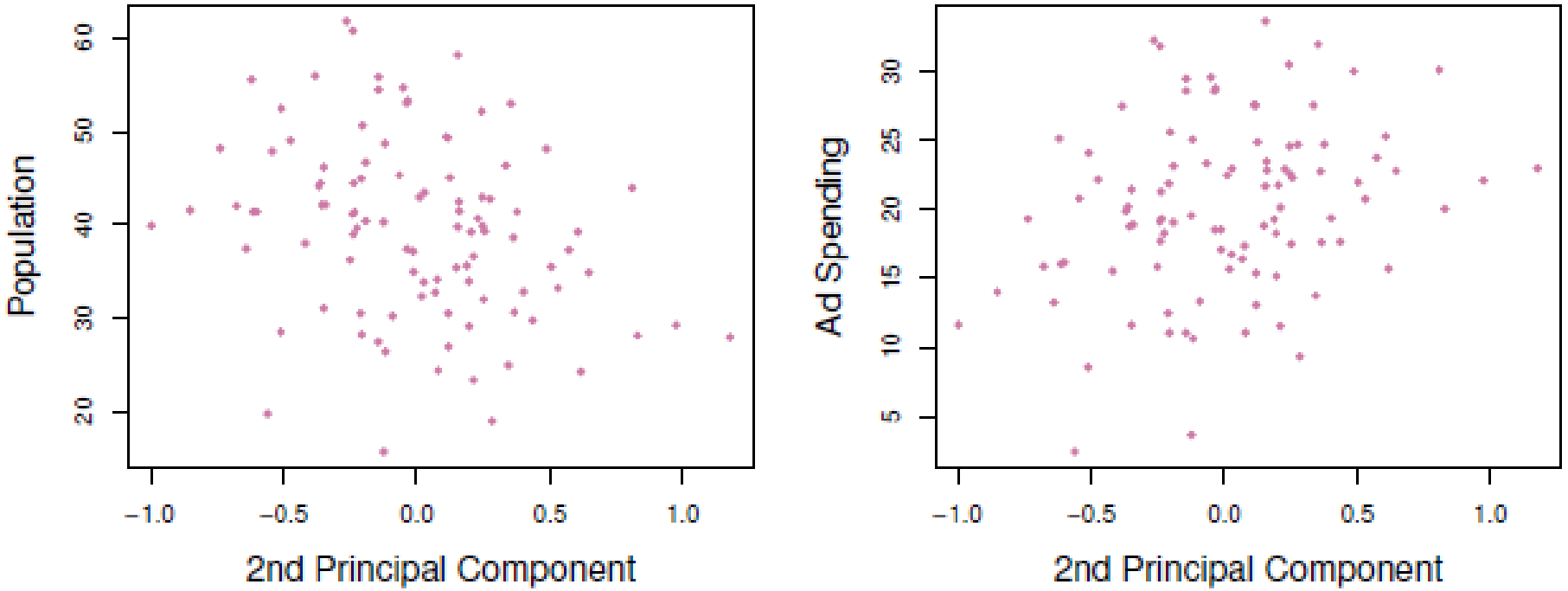
A subset of the advertising data. Left: The first principal component, chosen to minimize the sum of the squared perpendicular distances to each point, is shown in green. These distances are represented using the black dashed line segments. Right: The left-hand panel has been rotated so that the first principal component lies on the x-axis.

Principal Components Regression (cont.)



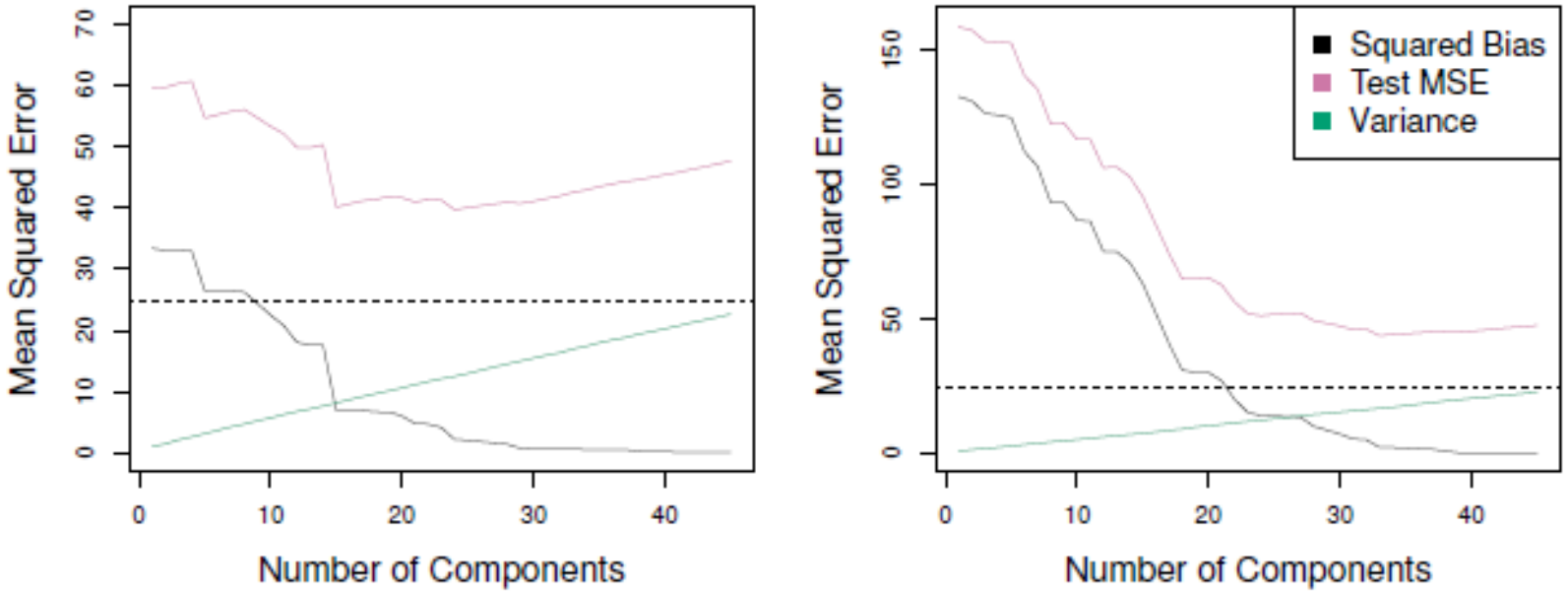
Plots of the first principal component scores z_{i1} versus pop *and* ad. *The relationships are strong.*

Principal Components Regression (cont.)



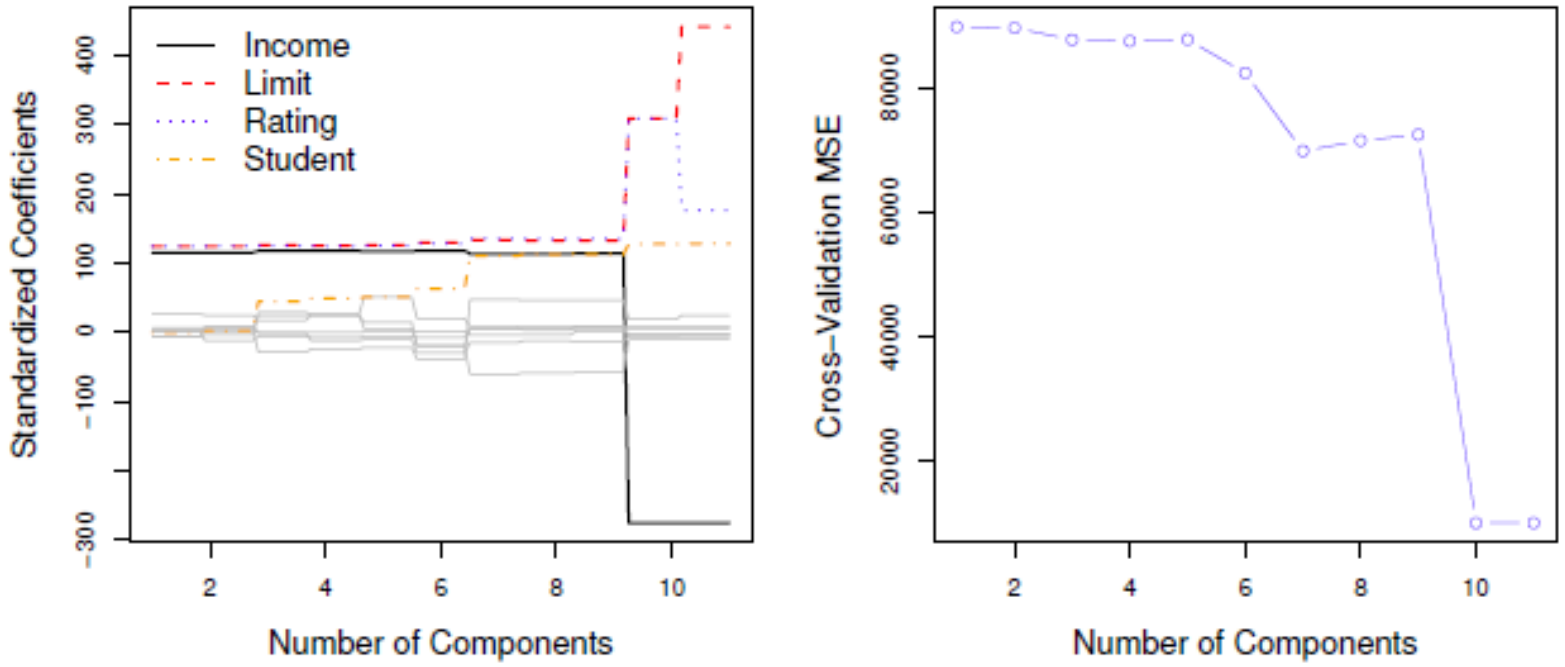
Plots of the second principal component scores z_{i2} versus pop *and* ad. *The relationships are weak.*

Principal Components Regression (cont.)

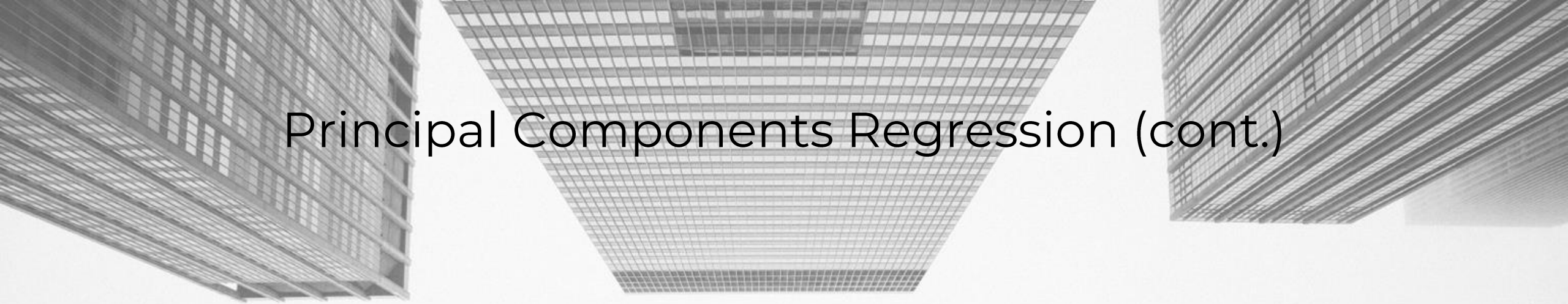


PCR was applied to two simulated data sets. The black, green, and purple lines correspond to squared bias, variance, and test mean squared error, respectively. Left: *Simulated data from slide 32.* Right: *Simulated data from slide 39.*

Principal Components Regression (cont.)



Left: *PCR standardized coefficient estimates on the* `Credit` *data set for different values of* M. Right: *The* 10*-fold cross validation MSE obtained using PCR, as a function of* M.



Principal Components Regression (cont.)

- As more principal components are used in the regression model, the bias decreases but the variance increases.
- PCR will tend to do well in cases when the first few principal components are sufficient to capture most of the variation in the predictors as well as the relationship with the response.
- We note that even though PCR provides a simple way to perform regression using $M < p$ predictors, it *is not* a feature selection method.
- In PCR, the number of principal components is typically chosen by cross-validation.

Partial Least Squares

- PCR identifies linear combinations, or *directions*, that best represents the predictors.
- These directions are identified is an *unsupervised* way, since the response *Y* is not used to help determine the principal component directions.
- That is, the response does not *supervise* the identification of the principal components.
- PCR suffers from a potentially serious drawback: there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response.

Partial Least Squares (cont.)

- Like PCR, *partial least squares* (PLS) is a dimension reduction method, which first identifies a new set of features $Z_1,\ldots,Z_M$ that are linear combinations of the original features.
- Then PLS fits an OLS linear model using these M new features.
- Unlike PCR, PLS identifies these new features in a *supervised* way; PLS makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are *related to the response*.
- The PLS approach attempts to find directions that help explain both the response and the predictors.

Partial Least Squares (cont.)

- After standardizing the p predictors, PLS computes the first partial least squares direction Z_1 by setting each ϕ_{1j} in

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$$

equal to the coefficient from the simple linear regression of Y onto X_j.

- One can show that this coefficient is proportional to the correlation between Y and X_j.

Partial Least Squares (cont.)

- Hence, in computing $Z_1 = \sum_{j=1}^{p} \phi_{1j} X_j$, PLS places the highest weight on the variables that are most strongly related to the response.
- Subsequent directions are found by taking residuals and then repeating the above prescription.
- As with PCR, the number *M* of PLS directions used in PLS is a tuning parameters that is typically chosen by cross-validation.
- While the supervised dimension reduction of PLS can reduce bias, it also has the potential to increase variance.

Partial Least Squares (cont.)

Algorithm 3.3 *Partial Least Squares.*

1. Standardize each $\mathbf{x}_j$ to have mean zero and variance one. Set $\hat{\mathbf{y}}^{(0)} = \bar{y}\mathbf{1}$, and $\mathbf{x}_j^{(0)} = \mathbf{x}_j,\ j = 1, \ldots, p$.

2. For $m = 1, 2, \ldots, p$

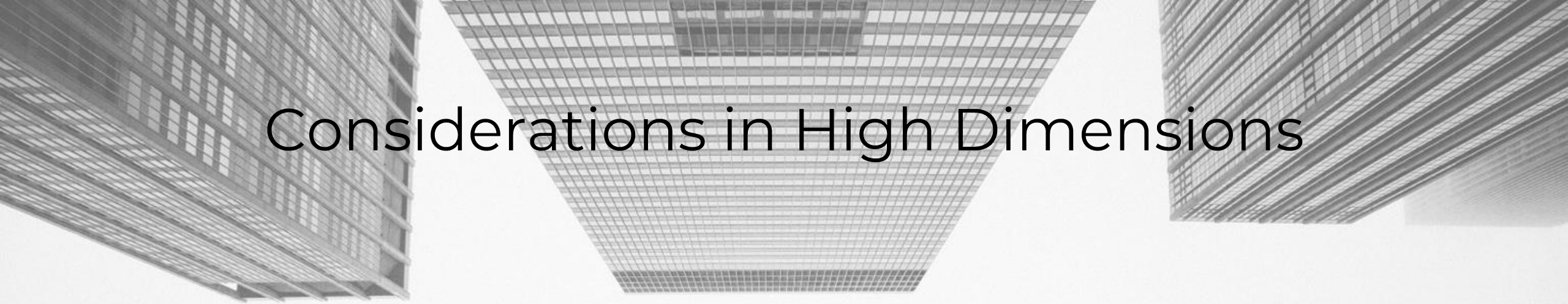
   (a) $\mathbf{z}_m = \sum_{j=1}^{p} \hat{\varphi}_{mj}\mathbf{x}_j^{(m-1)}$, where $\hat{\varphi}_{mj} = \langle \mathbf{x}_j^{(m-1)}, \mathbf{y} \rangle$.

   (b) $\hat{\theta}_m = \langle \mathbf{z}_m, \mathbf{y} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle$.

   (c) $\hat{\mathbf{y}}^{(m)} = \hat{\mathbf{y}}^{(m-1)} + \hat{\theta}_m \mathbf{z}_m$.

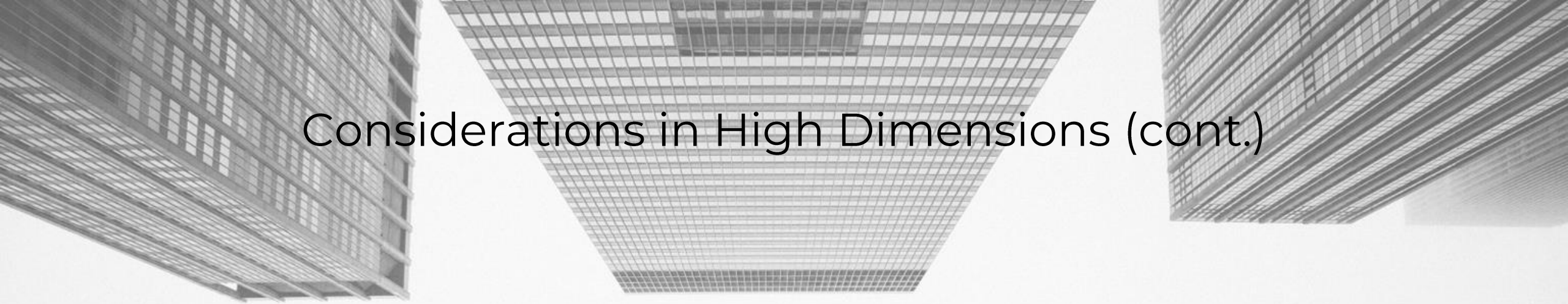
   (d) Orthogonalize each $\mathbf{x}_j^{(m-1)}$ with respect to $\mathbf{z}_m$: $\mathbf{x}_j^{(m)} = \mathbf{x}_j^{(m-1)} - [\langle \mathbf{z}_m, \mathbf{x}_j^{(m-1)} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle]\mathbf{z}_m,\ j = 1, 2, \ldots, p$.

3. Output the sequence of fitted vectors $\{\hat{\mathbf{y}}^{(m)}\}_1^p$. Since the $\{\mathbf{z}_\ell\}_1^m$ are linear in the original $\mathbf{x}_j$, so is $\hat{\mathbf{y}}^{(m)} = \mathbf{X}\hat{\beta}^{\text{pls}}(m)$. These linear coefficients can be recovered from the sequence of PLS transformations.



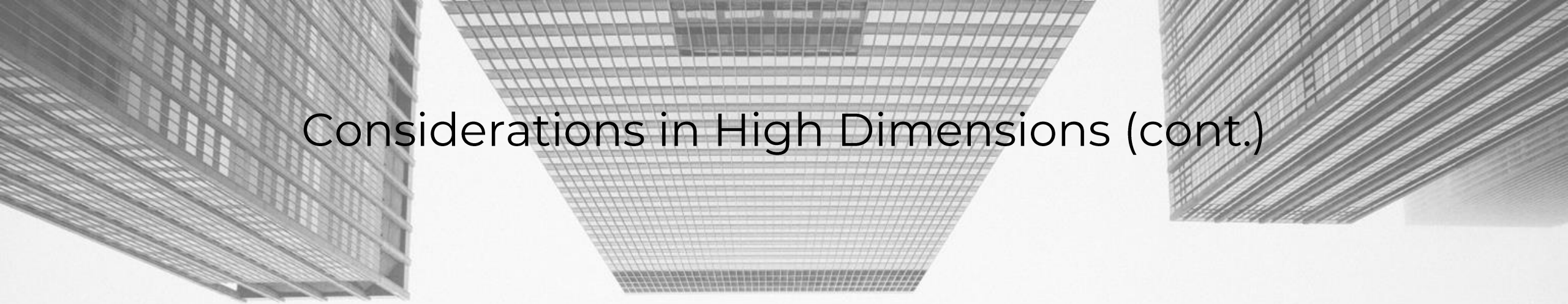
Considerations in High Dimensions

- While *p* can be extremely large, the number of observations *n* is often limited due to cost, sample availability, etc.
- Data sets containing more features than observations are often referred to a *high-dimensional*.
- When the number of features *p* is as large as, or larger than, the number of observations *n*, OLS should not be performed.
  - It is too *flexible* and hence overfits the data.
- Forward stepwise selection, ridge regression, lasso, and PCR are particularly useful for performing regression in the high-dimensional setting.



Considerations in High Dimensions (cont.)

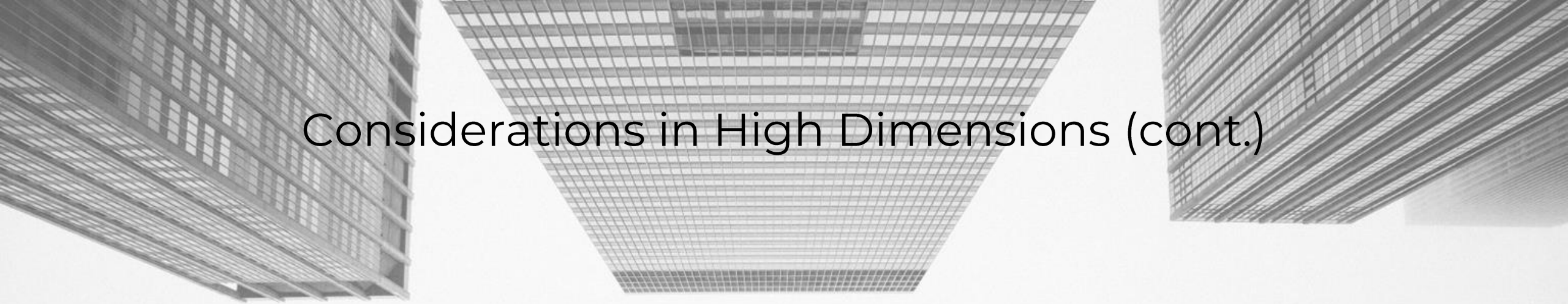
- Regularization or shrinkage plays a key role in high-dimensional problems.
- Appropriate tuning parameter selection is crucial for good predictive performance.
- The test error tends to increase as the dimensionality of the problem (i.e. the number of features or predictors) increases, unless the additional features are truly associated with the response.
  - Known as the *curse of dimensionality*



Considerations in High Dimensions (cont.)

- *Curse of dimensionality*
  - Adding additional *signal* features that are truly associated with the response will improve the fitted model, in the sense of leading to a reduction in test set error.
  - Adding *noise* features that are not truly associated with the response will lead to a deterioration in the fitted model, and consequently an increased test set error.

- Noise features increase the dimensionality of the problem, exacerbating the risk of overfitting without any potential upside in terms of improved test set error.



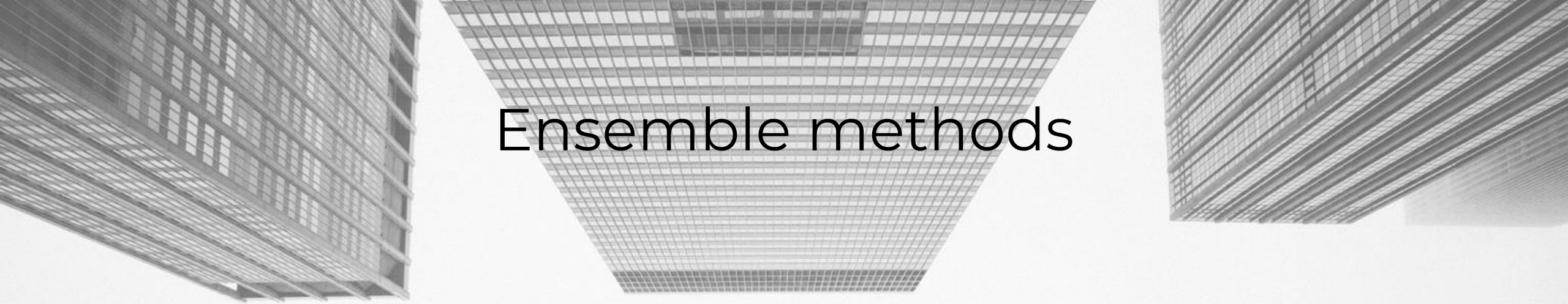
Considerations in High Dimensions (cont.)

- In the high-dimensional setting, the multicollinearity problem is extreme: any variable in the model can be written as a linear combination of all of the other variables in the models.
- It is also important to be particularly careful in reporting errors and measures of model fit in the high-dimensional setting.
- One should *never* use sum of squared errors, p-values, R^2 statistics, or other traditional measures of model fit on the *training data* as evidence of good model fit in the high-dimensional setting.
- It is important to report results on an independent test set, or cross-validation errors.

Summary

- Best subset selection and stepwise selection methods.
- Estimate test error by adjusting training error to account for bias due to overfitting.
- Estimate test error using validation set approach and cross-validation approach.
- Ridge regression and the lasso as shrinkage (regularization) methods.
- Principal components regression and partial least squares.
- Considerations for high-dimensional settings.

Tree Learning: Random Forest



Ensemble methods

- A single decision tree does not perform well
- But, it is super fast
- What if we learn multiple trees?

We need to make sure they do not all just learn the same

Bagging

If we split the data in random different ways, decision trees give different results, **high variance.**

Bagging: Bootstrap **agg**regat**ing** is a method that result in low variance.

If we had multiple realizations of the data (or multiple samples) we could calculate the predictions multiple times and take the average of the fact that averaging multiple onerous estimations produce less uncertain results

Bagging

Say for each sample *b*, we calculate $f^b(x)$, then:

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x)$$

How?

Bootstrap

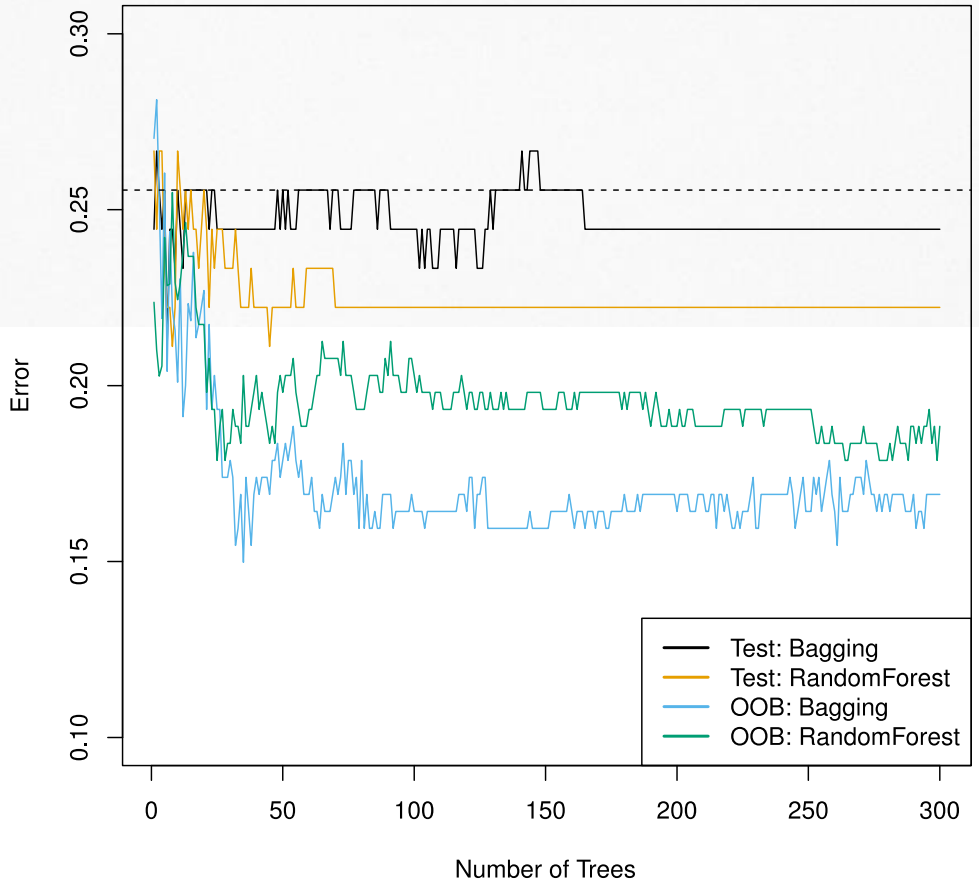
Construct B (hundreds) of trees (no pruning)

Learn a classifier for each bootstrap sample and average them

Very effective

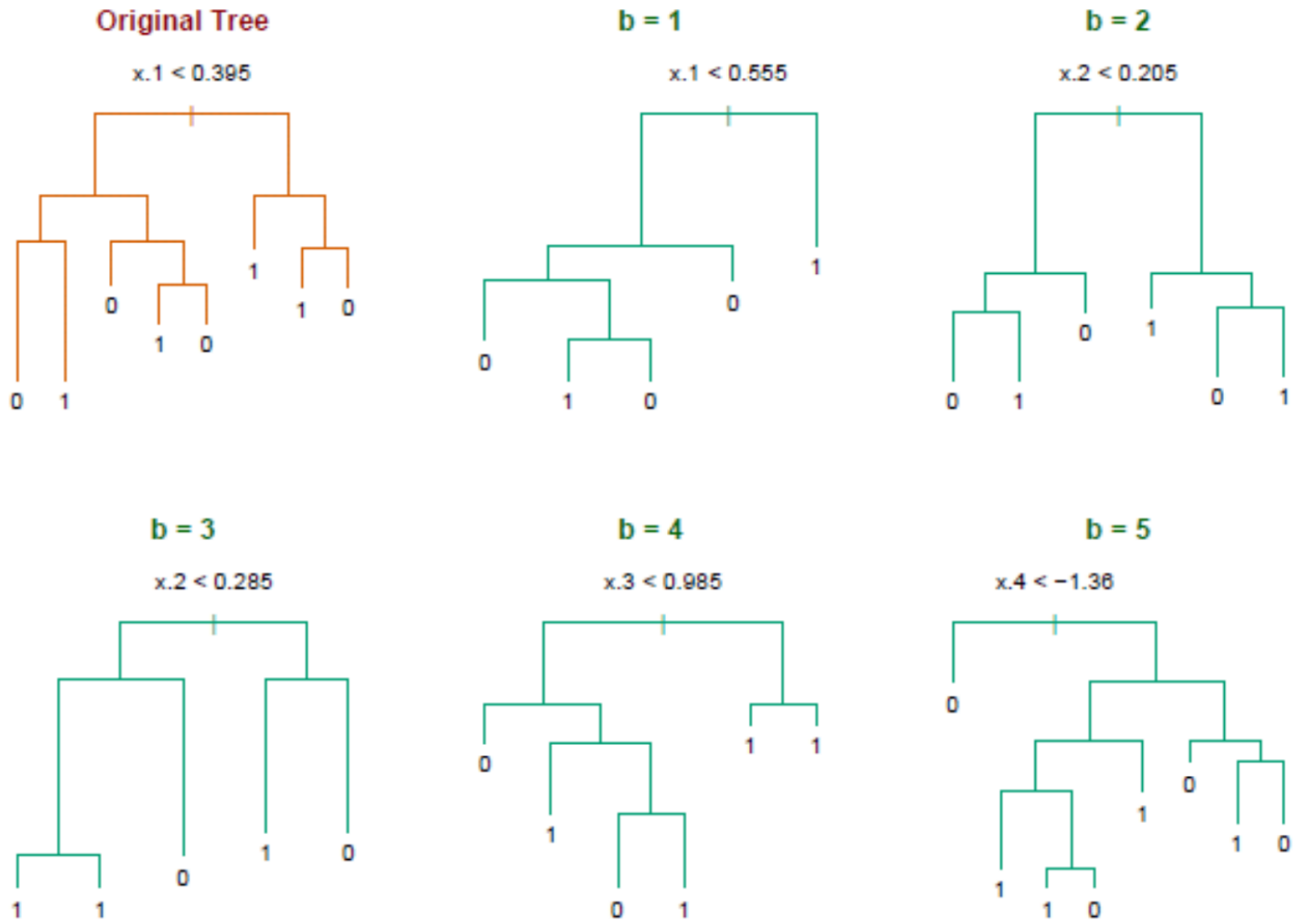
Bagging for classification: Majority vote

NO OVERFITTING



Test error

Bagging decision trees



Hastie et al.,"The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer (2009)

Out-of-Bag Error Estimation

- No cross validation?
- Remember, in bootstrapping we sample with replacement, and therefore **not all observations are used for each bootstrap sample**. On average 1/3 of them are not used!
- We call them out-of-bag samples (OOB)
- We can predict the response for the *i-th* observation using each of the trees in which that observation was OOB and do this for *n* observations
- Calculate overall OOB MSE or classification error

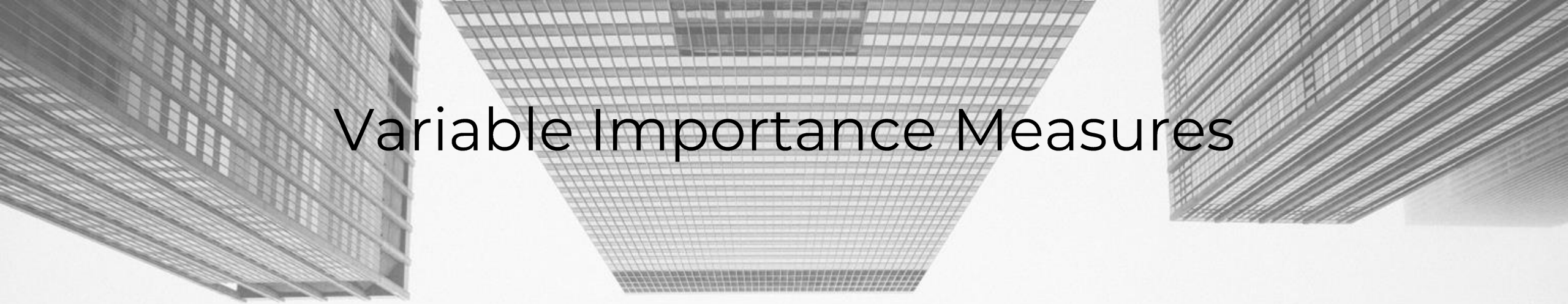
Bagging

Reduces overfitting (variance)

Normally uses one type of classifier

Decision trees are popular

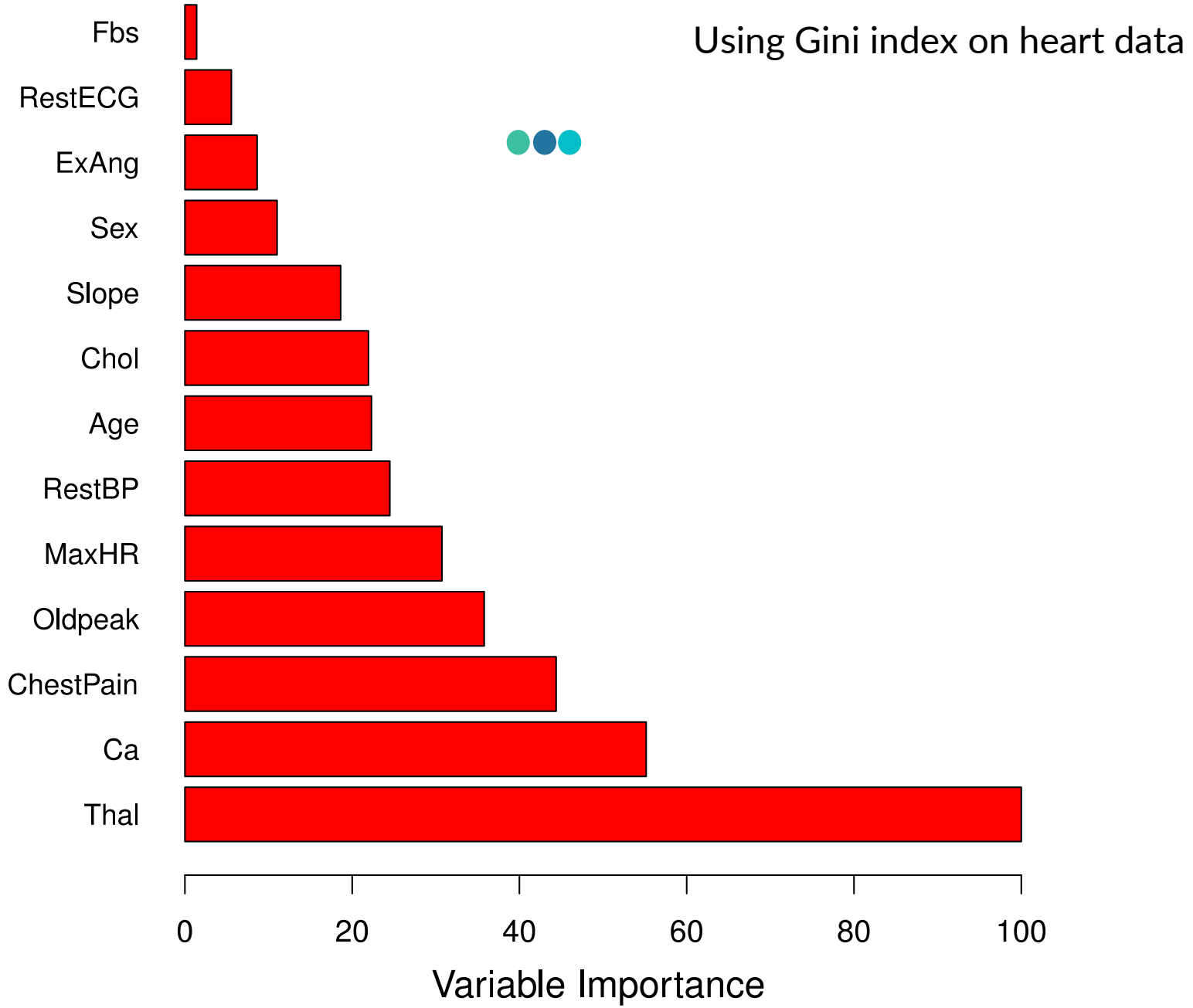
Easy to parallelize

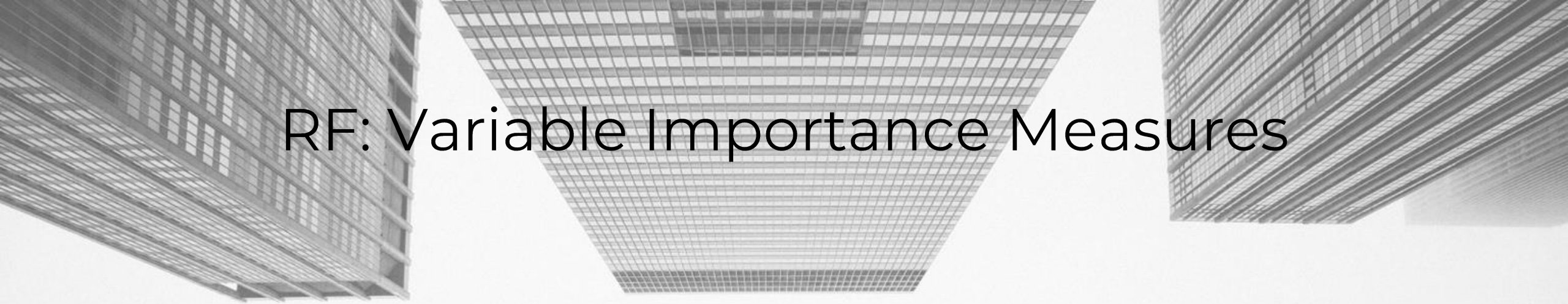


Variable Importance Measures

- Bagging results in improved accuracy over prediction using a single tree
- Unfortunately, difficult to interpret the resulting model. Bagging improves prediction accuracy at the expense of interpretability.

Calculate the total amount that the RSS or Gini index is decreased due to splits over a given predictor, averaged over all B trees.


Using Gini index on heart data
Fbs
RestECG
ExAng
Sex
Slope
Chol
Age
RestBP
MaxHR
Oldpeak
ChestPain
Ca
Thal
0
20
40
60
80
100
Variable Importance

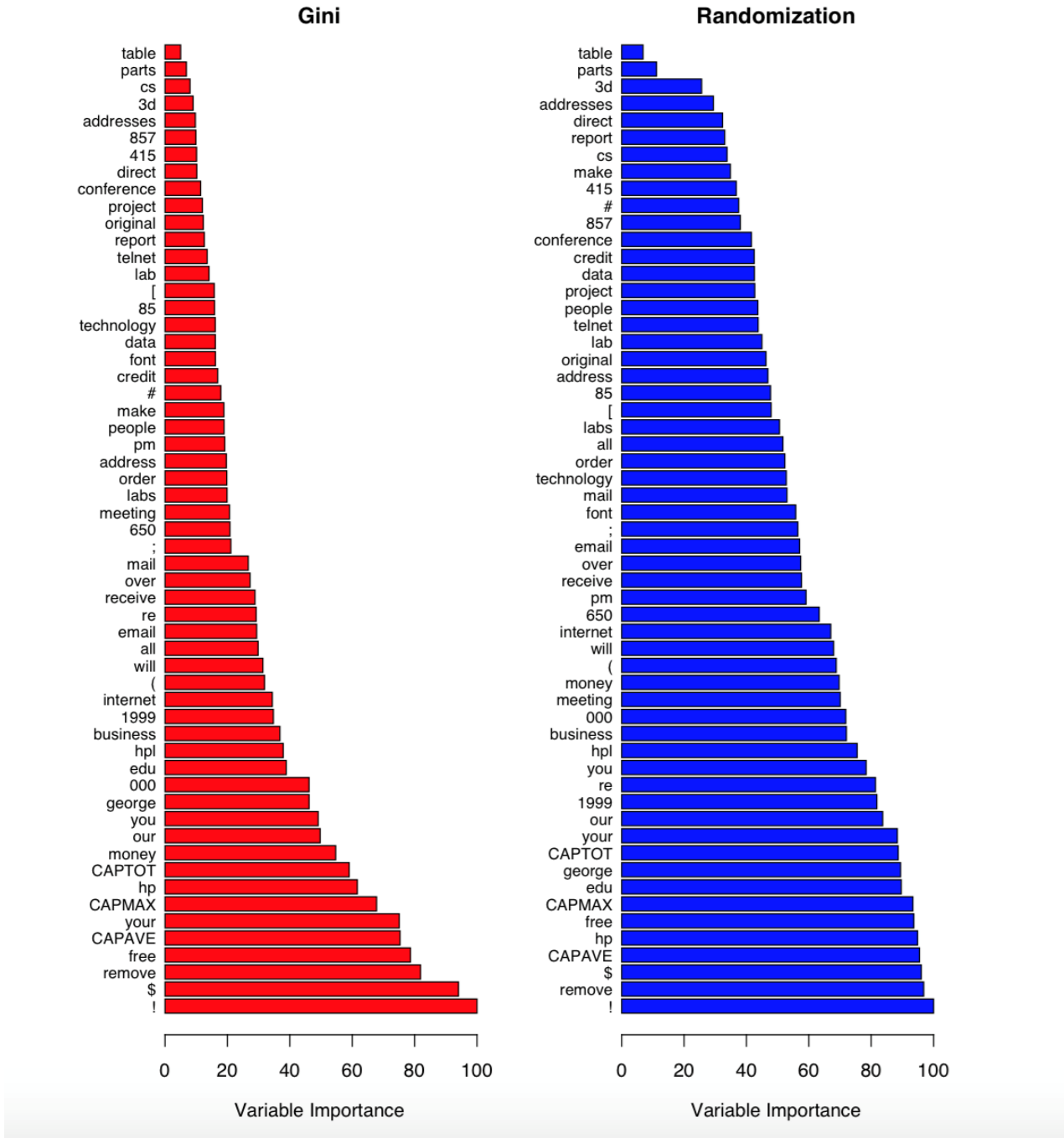


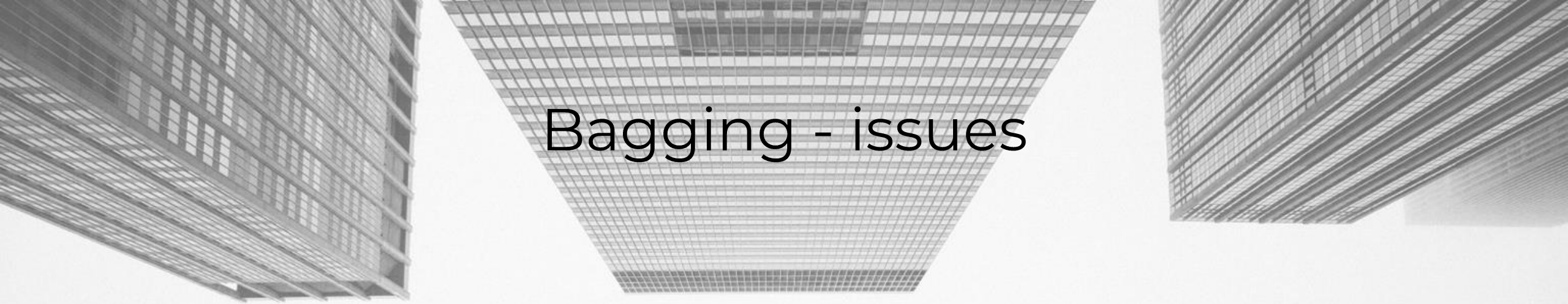
RF: Variable Importance Measures

Record the prediction accuracy on the oob samples for each tree

Randomly permute the data for column *j* in the oob samples the record the accuracy again.

The decrease in accuracy as a result of this permuting is averaged over all trees, and is used as a measure of the importance of variable j in the random forest.


Gini
table
parts
cs
3d
addresses
857
415
direct
conference
project
original
report
telnet
lab
[
85
technology
data
font
credit
#
make
people
pm
address
order
labs
meeting
650
;
mail
over
receive
re
email
all
will
(
internet
1999
business
hpl
edu
000
george
you
our
money
CAPTOT
hp
CAPMAX
your
CAPAVE
free
remove
$
!
0
20
40
60
80
100
Variable Importance
Randomization
table
parts
3d
addresses
direct
report
cs
make
415
#
857
conference
credit
data
project
people
telnet
lab
original
address
85
[
labs
all
order
technology
mail
font
;
email
over
receive
pm
650
internet
will
(
money
meeting
000
business
hpl
you
re
1999
our
your
CAPTOT
george
edu
CAPMAX
free
hp
CAPAVE
$
remove
!
0
20
40
60
80
100
Variable Importance



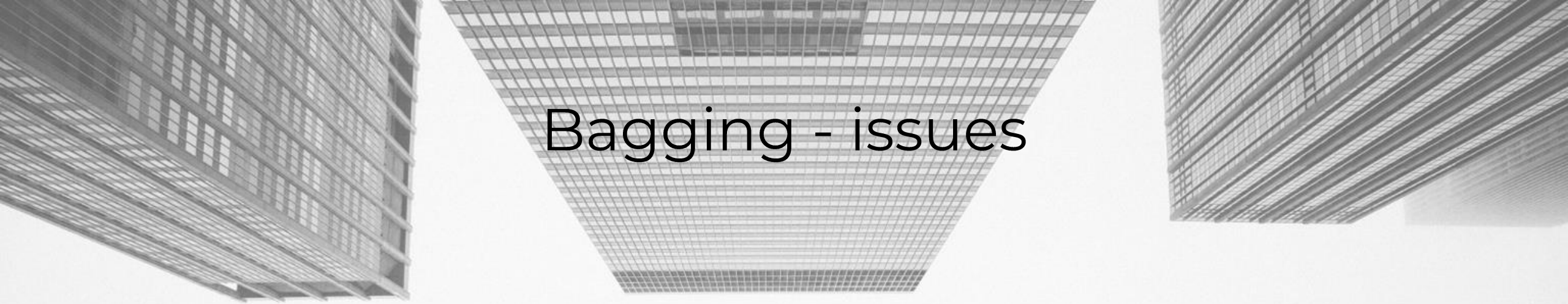
Bagging - issues

Each tree is identically distributed (i.d.)

➔ the expectation of the average of B such trees is the same as the expectation of any one of them

➔ the bias of bagged trees is the same as that of the individual trees

i.d. and not i.i.d



Bagging - issues

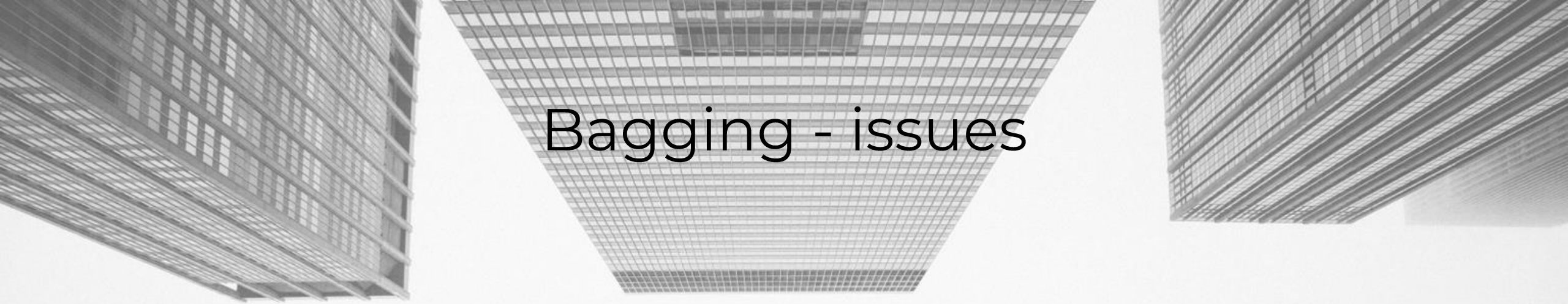
An average of *B* i.i.d. random variables, each with variance σ^2, has variance: σ^2/B

If i.d. (identical but not independent) and pair correlation ρ is present, then the variance is:

$$\rho\,\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

As *B* increases the second term disappears but the first term remains

Why does bagging generate correlated trees?

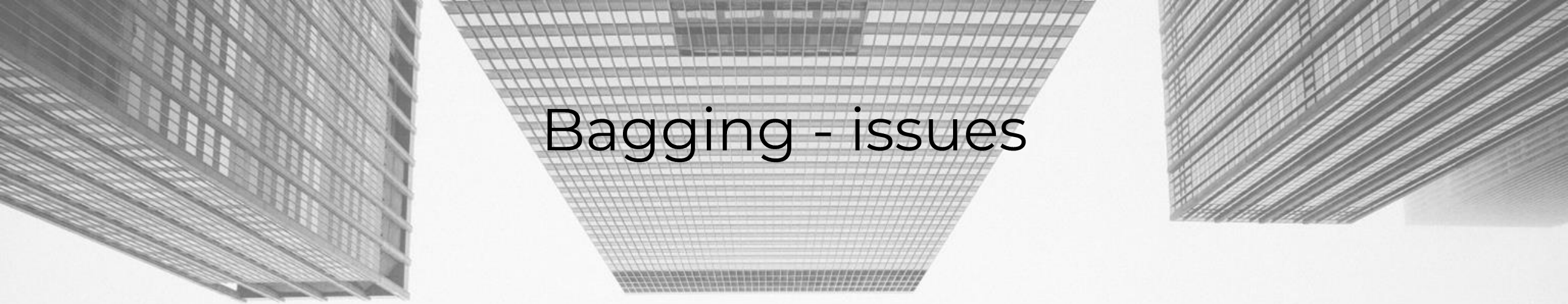


Bagging - issues

Suppose that there is one very strong predictor in the data set, along with a number of other moderately strong predictors.

Then all bagged trees will select the strong predictor at the top of the tree and therefore all trees will look similar.

How do we avoid this?

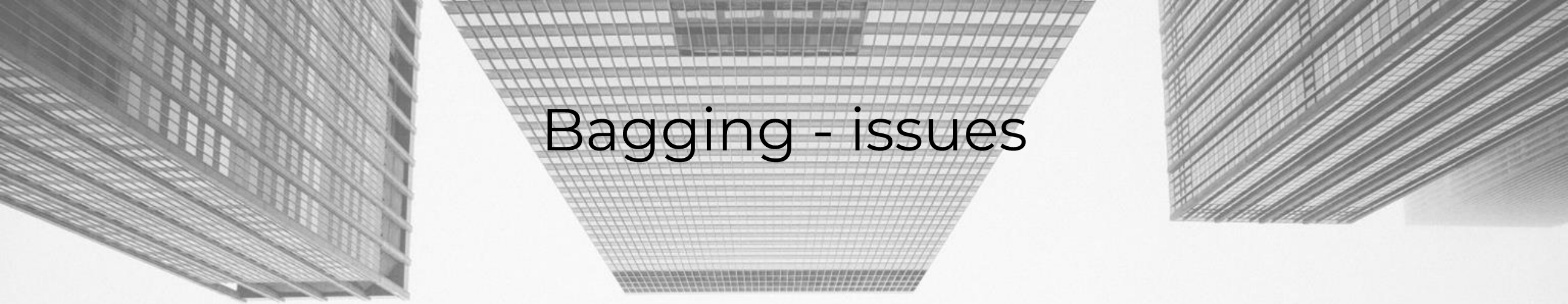


Bagging - issues

We can penalize the splitting (like in pruning) with a penalty term that depends on the number of times a predictor is selected at a given length

We can restrict how many times a predictor can be used

We only allow a certain number of predictors



Bagging - issues

Remember we want i.i.d such as the bias to be the same and variance to be less?

Other ideas?

What if we consider only a subset of the predictors at each split?

We will still get correlated trees unless ….

we **randomly** select the subset !

Random Forests

As in bagging, we build a number of decision trees on bootstrapped training samples each time a split in a tree is considered, a random sample of *m* predictors is chosen as split candidates from the full set of p predictors.

Note that if $m = p$, then this is bagging.

Random Forests

Random forests are popular. Leo Breiman's and Adele Cutler maintains a random forest website where the software is freely available, and of course it is included in every ML/STAT package

http://www.stat.berkeley.edu/~breiman/RandomForests/

Random Forests Algorithm

For b = 1 to B:

(a) Draw a bootstrap sample Z^* of size N from the training data.

(b) Grow a random-forest tree to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size $\boldsymbol{n_{min}}$ is reached.

i. Select $\boldsymbol{m}$ variables at random from the p variables.

ii. Pick the best variable/split-point among the m.

iii. Split the node into two daughter nodes.

Output the ensemble of trees.

To make a prediction at a new point x we do:

For regression: average the results

For classification: majority vote

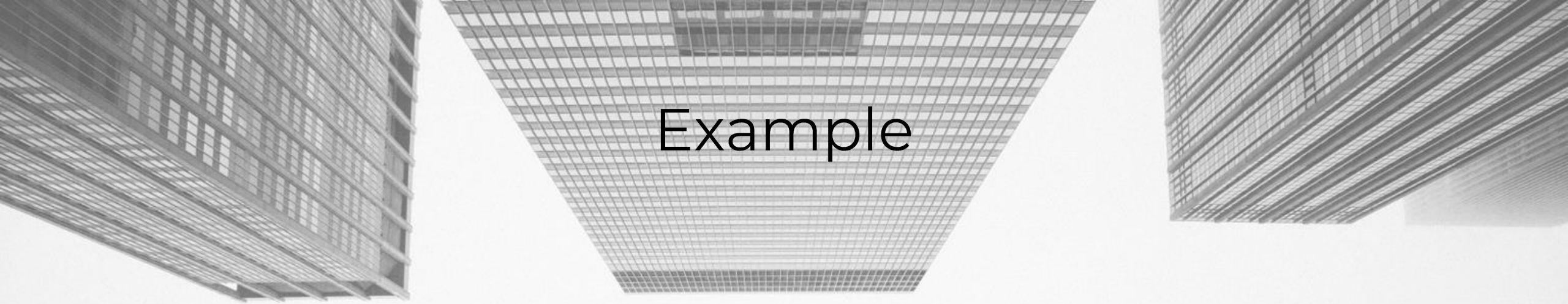
Random Forests Tuning

The inventors make the following recommendations:

- For classification, the default value for m is $\sqrt{p}$ and the minimum node size is one.
- For regression, the default value for m is $p/3$ and the minimum node size is five.

In practice the best values for these parameters will depend on the problem, and they should be treated as tuning parameters.

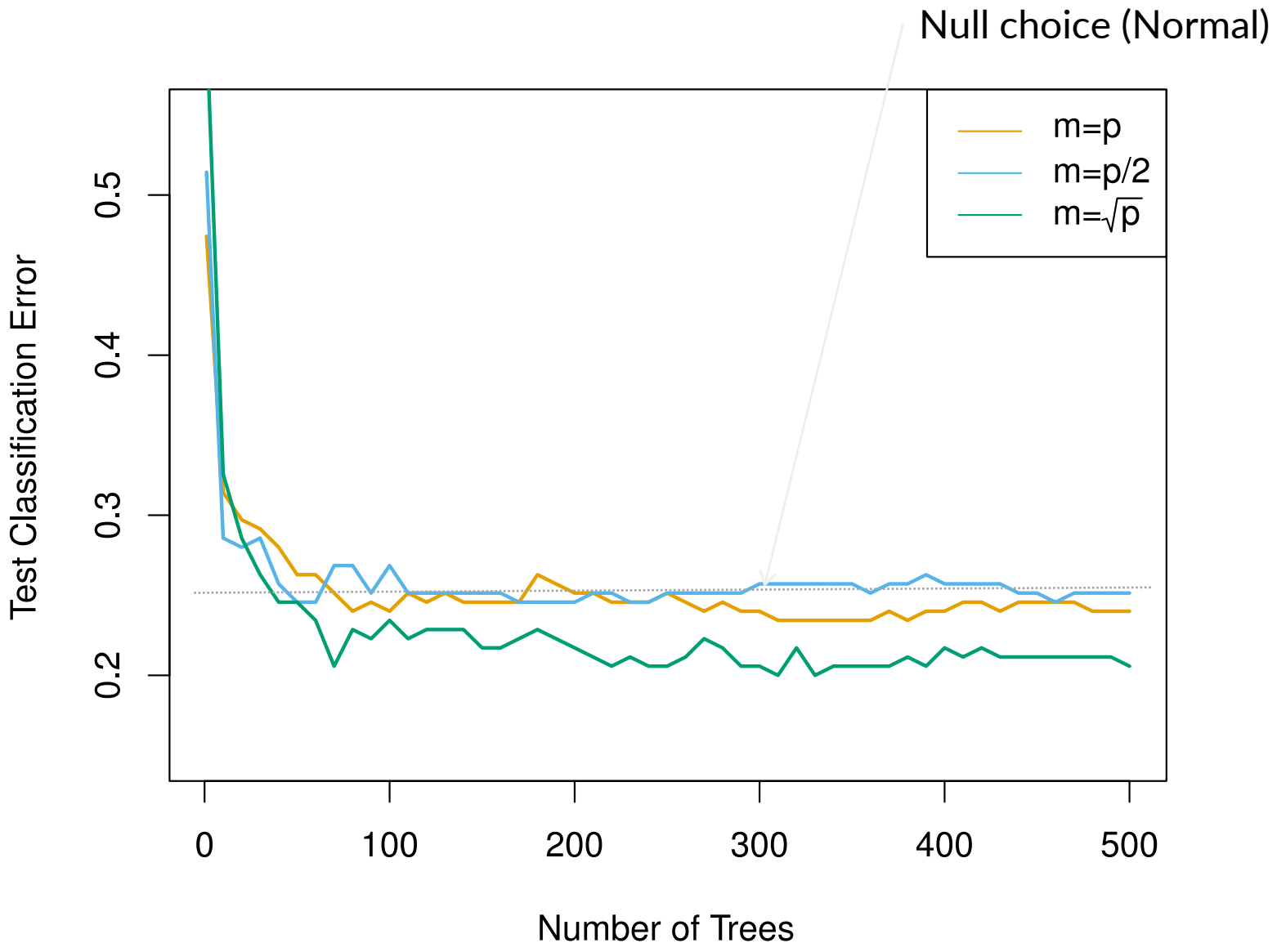
Like with Bagging, we can use OOB and therefore RF can be fit in one sequence, with cross-validation being performed along the way. Once the OOB error stabilizes, the training can be terminated.



Example

- 4,718 genes measured on tissue samples from 349 patients.
- Each gene has different expression
- Each of the patient samples has a qualitative label with 15 different levels: either normal or 1 of 14 different types of cancer.

Use random forests to predict cancer type based on the 500 genes that have the largest variance in the training set.


Null choice (Normal)
m=p
m=p/2
m=√p
Test Classification Error
0.2
0.3
0.4
0.5
0
100
200
300
400
500
Number of Trees

Random Forests Issues

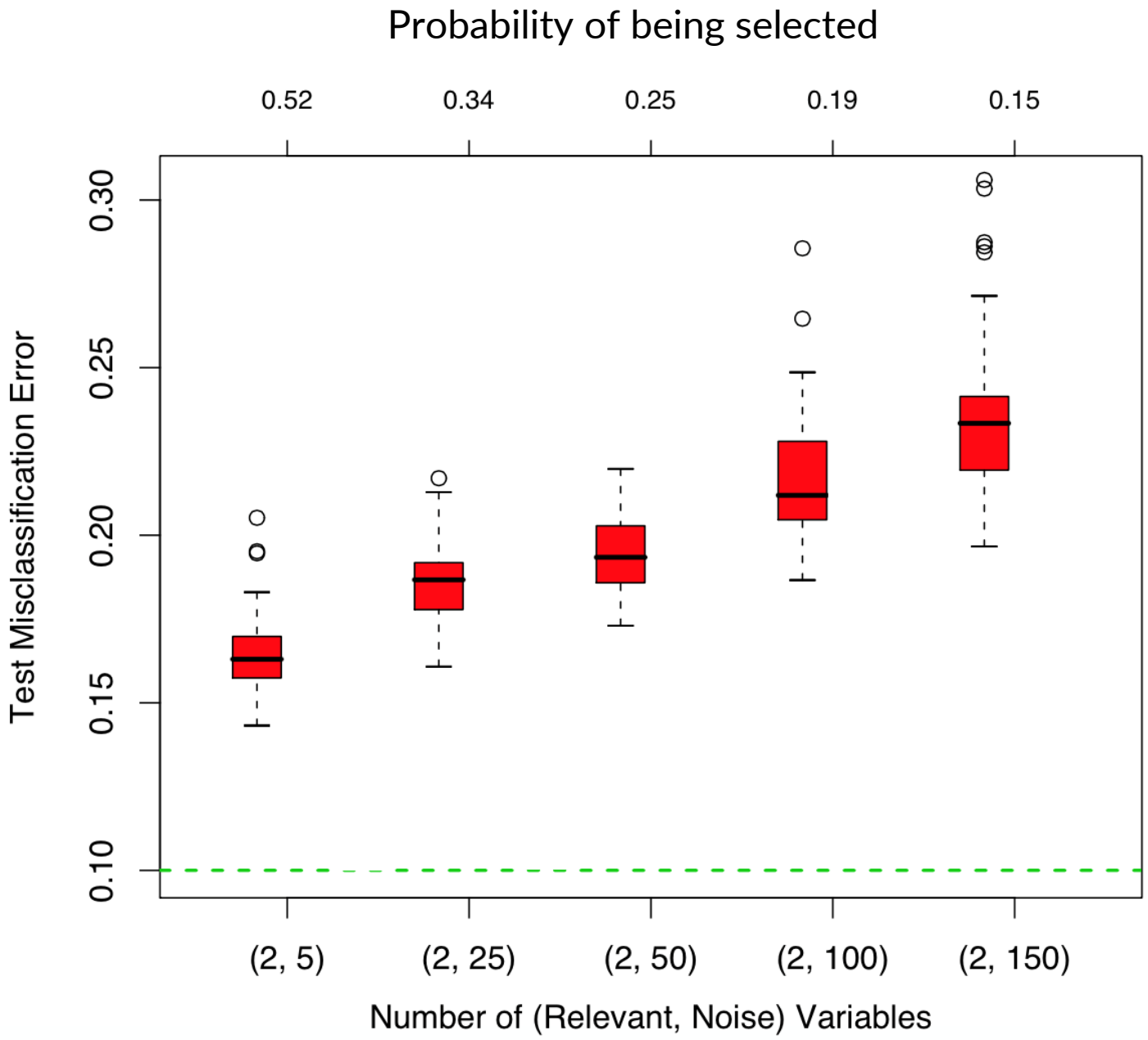
When the number of variables is large, but the fraction of relevant variables is small, random forests are likely to perform poorly when *m* is small

Why?

Because:

At each split the chance can be small that the relevant variables will be selected

For example, with 3 relevant and 100 not so relevant variables the probability of any of the relevant variables being selected at any split is ~0.25


Probability of being selected
0.52
0.34
0.25
0.19
0.15
Test Misclassification Error
0.10
0.15
0.20
0.25
0.30
(2, 5)
(2, 25)
(2, 50)
(2, 100)
(2, 150)
Number of (Relevant, Noise) Variables

Can RF overfit?

Random forests “cannot overfit” the data wrt to number of trees.

Why?

The number of trees, *B* does not mean increase in the flexibility of the model

Boosting

Boosting is a general approach that can be applied to many statistical learning methods for regression or classification.

Bagging: Generate multiple trees from bootstrapped data and average the trees.

Recall bagging results in i.d. trees and not i.i.d.

RF produces i.i.d (or more independent) trees by randomly selecting a subset of predictors at each step

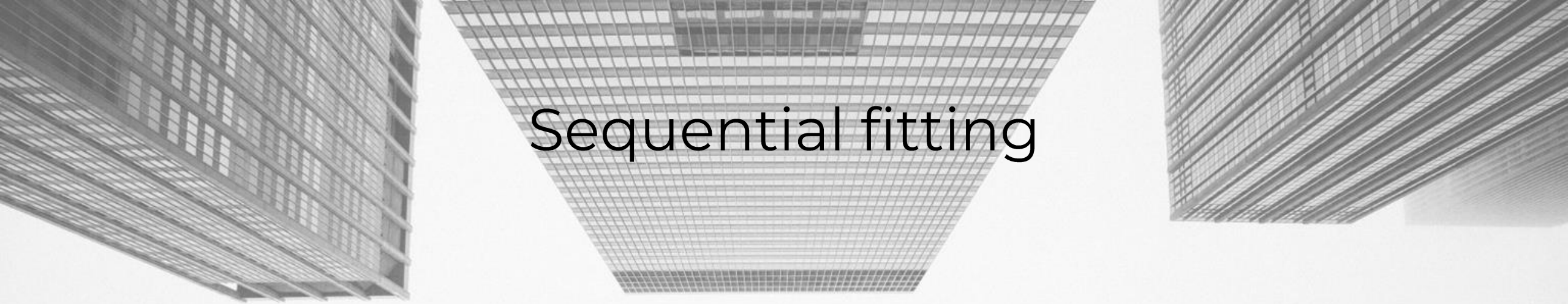
Boosting

Boosting works very differently.

1. Boosting does not involve bootstrap sampling

2. Trees are grown sequentially: each tree is grown using information from previously grown trees

3. Like bagging, boosting involves combining a large number of decision trees, f^1, . . . , f^B



Sequential fitting

Given the current model,

- we fit a decision tree to the **residuals** from the model. Response variable now is the residuals and not Y
- We then add this new decision tree into the fitted function in order to update the residuals
- The learning rate has to be controlled

Boosting for regression

1. Set $f(x)=0$ and $r_i = y_i$ for all i in the training set.

2. For $b=1,2,\ldots,B$, repeat:

   a. Fit a tree with d splits(+1 terminal nodes) to the training data (X, r).

   b. Update the tree by adding in a shrunken version of the new tree:

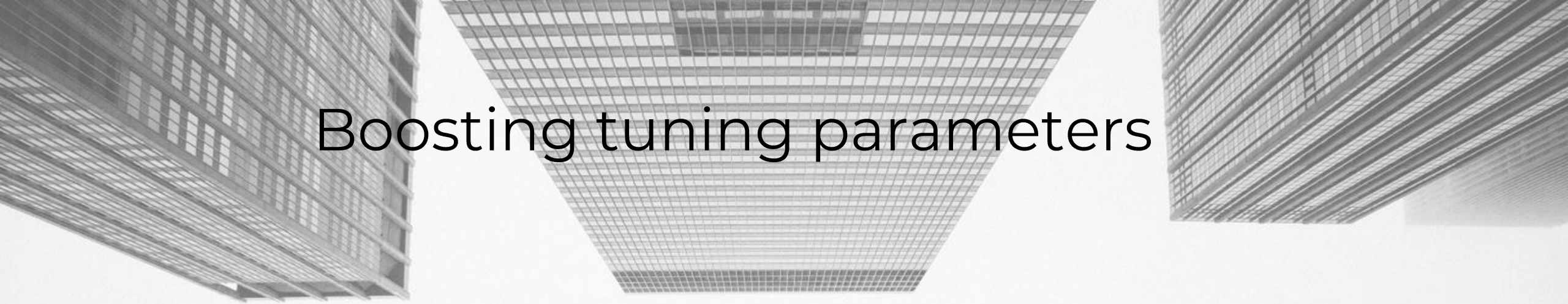
$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

   c. Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

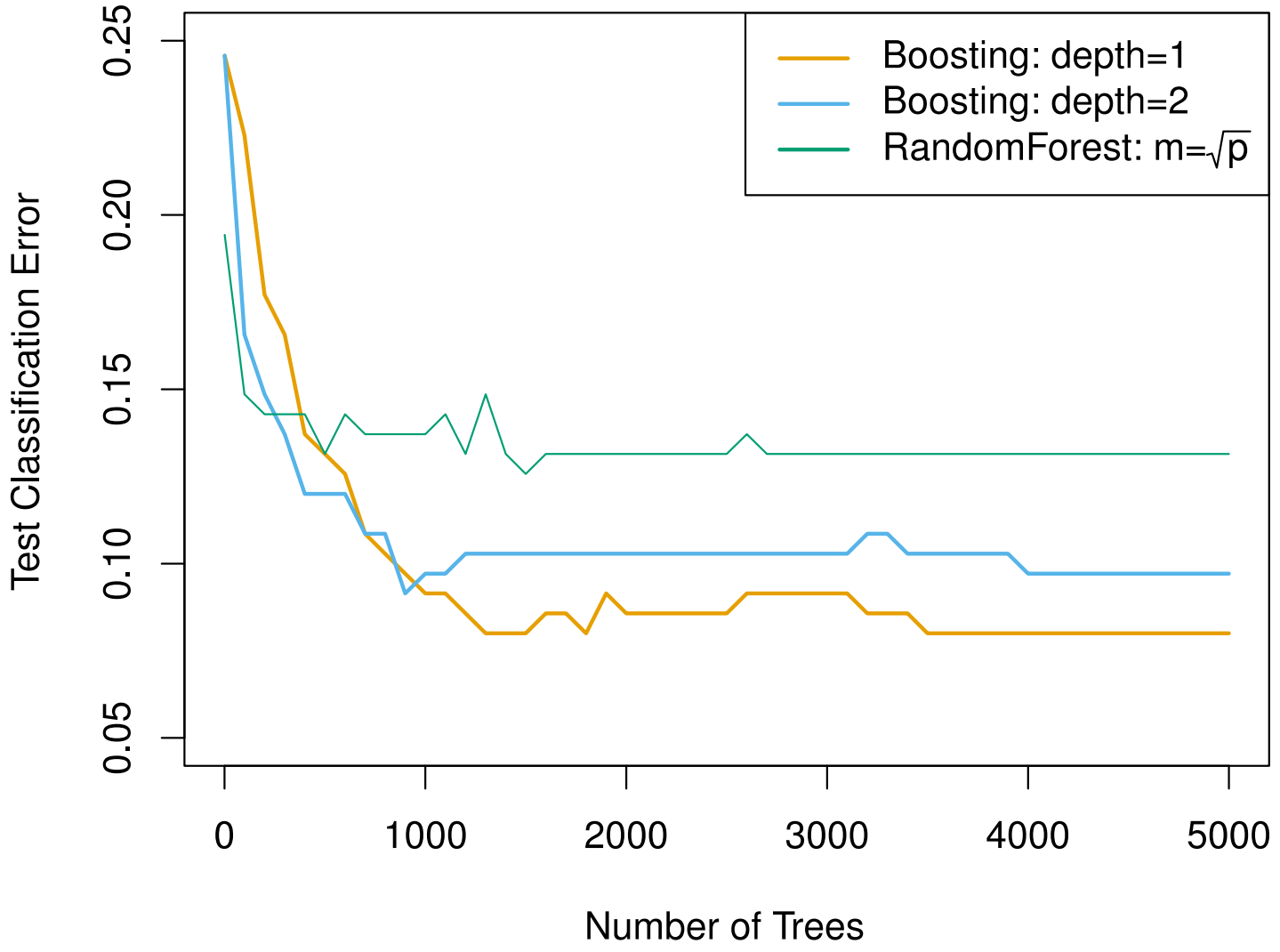


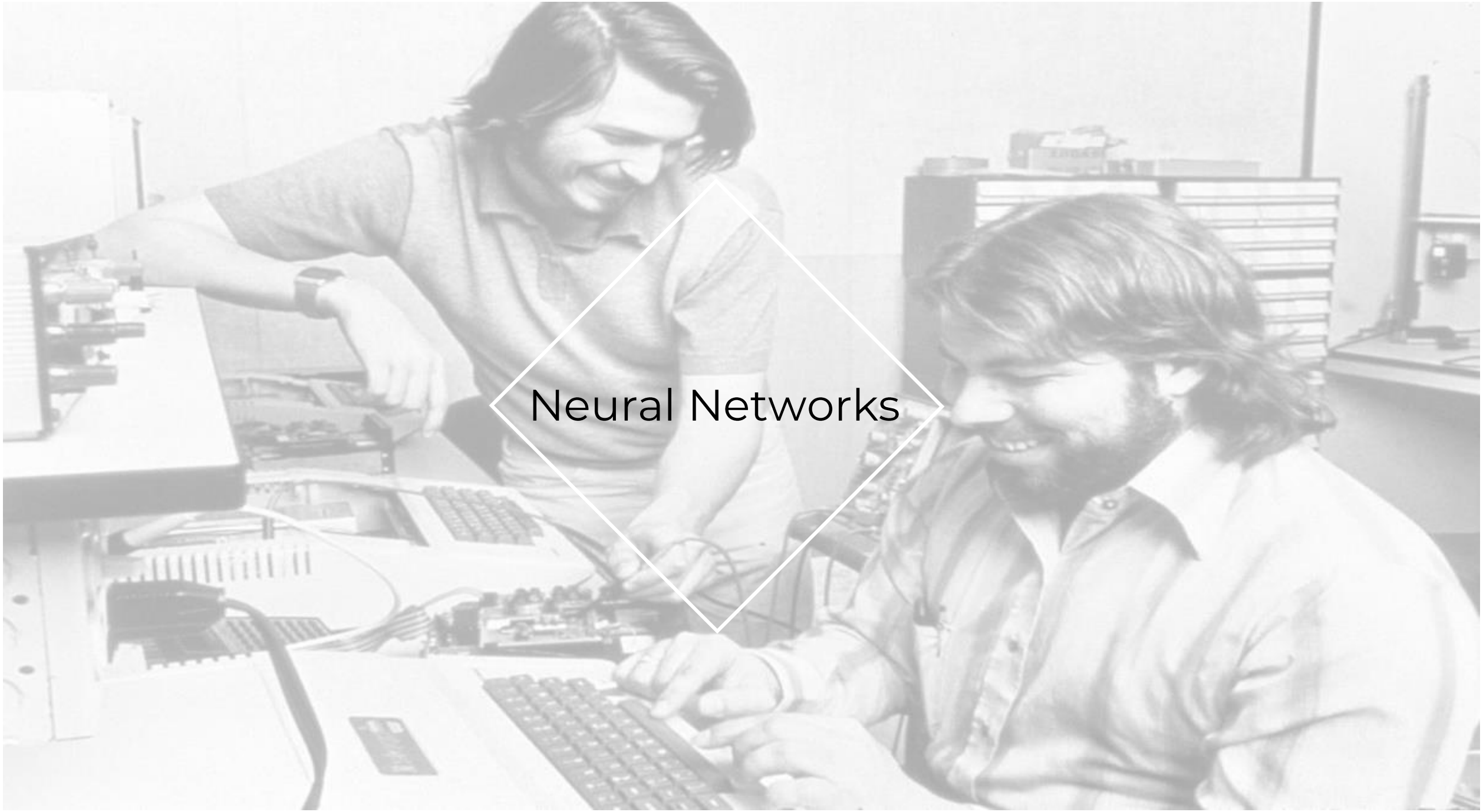
Boosting tuning parameters

- The number of trees B. RF and Bagging do not overfit as B increases. Boosting can overfit! **Cross Validation**

- The shrinkage parameter λ, a small positive number. Typical values are 0.01 or 0.001 but it depends on the problem. λ only controls the learning rate

- The number d of splits in each tree, which controls the complexity of the boosted ensemble. Stumpy trees, d = 1 works well.


Boosting: depth=1
Boosting: depth=2
RandomForest: m=√p
Test Classification Error
0.05
0.10
0.15
0.20
0.25
0
1000
2000
3000
4000
5000
Number of Trees


Neural Networks

Background and Motivation

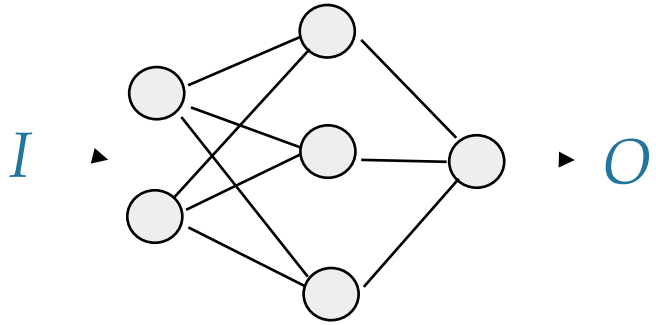
ITEM	COMPUTER	BRAIN
Complexity	ordered structure serial processor	10^10 neuron processors 10^4 connections
Processor Speed	10,000,000 operations per second	100 operations per second
Computational Power	one operation at a time 1 or 2 inputs	millions of operations at a time thousands of inputs

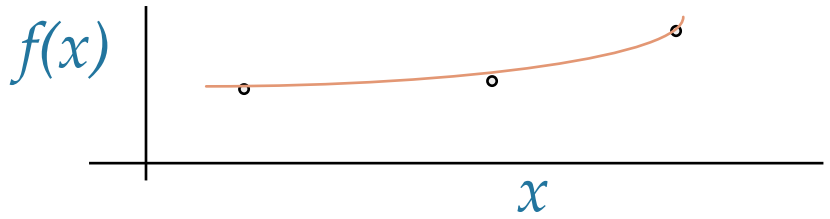
Background and Motivation

Inherent Advantages of the Brain:

"distributed processing and representation"

- Parallel processing speeds
- Fault tolerance
- Graceful degradation
- Ability to generalize





Background and Motivation

History of Artificial Neural Networks

- Creation:
  1890: William James - defined a neuronal process of learning
- Promising Technology:
  1943: McCulloch and Pitts - earliest mathematical models
  1954: Donald Hebb and IBM research group - earliest simulations
  1958: Frank Rosenblatt - The Perceptron
- Disenchantment:
  1969: Minsky and Papert - perceptrons have severe limitations
- Re-emergence:
  1985: Multi-layer nets that use back-propagation
  1986: PDP Research Group - multi-disciplined approach

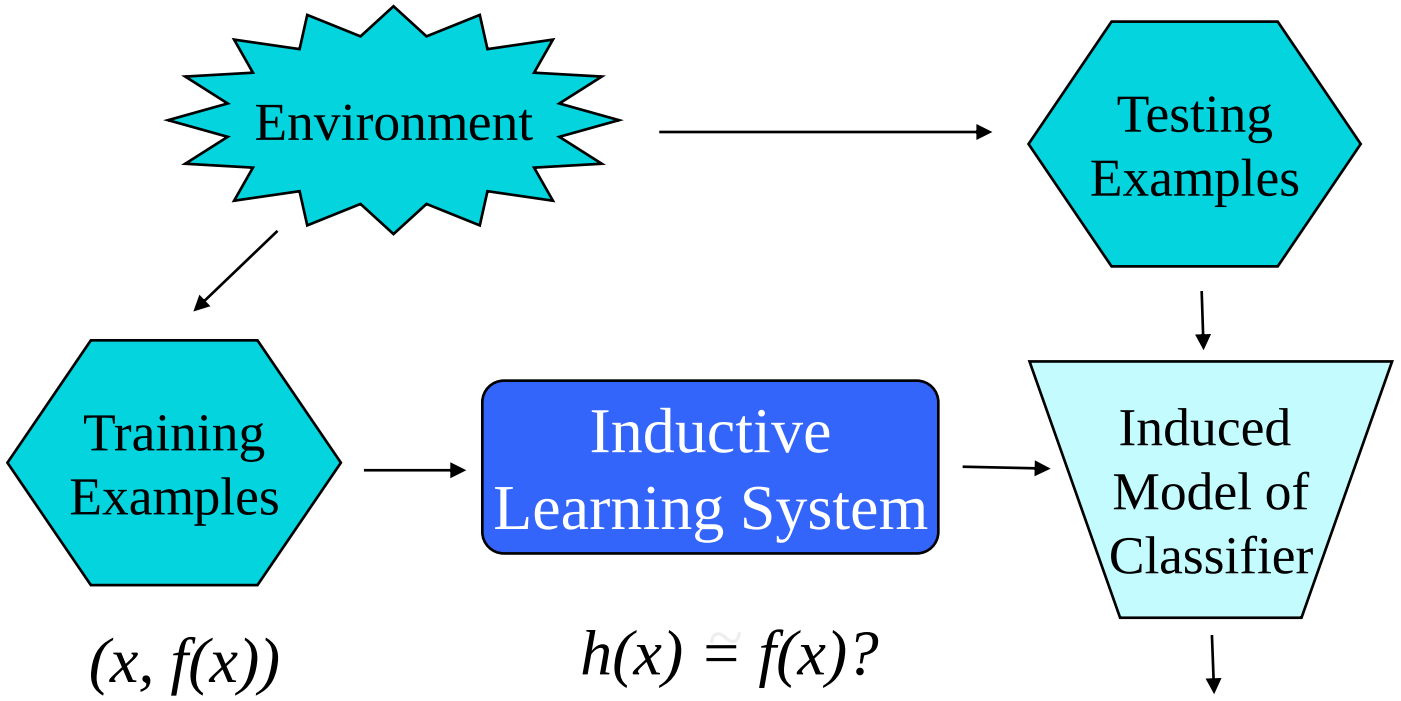
Background and Motivation

ANN application areas …

- Science and medicine: modeling, prediction, diagnosis, pattern recognition
- Manufacturing: process modeling and analysis
- Marketing and Sales: analysis, classification, customer targeting
- Finance: portfolio trading, investment support
- Banking & Insurance: credit and policy approval
- Security: bomb, iceberg, fraud detection
- Engineering: dynamic load schedding, pattern recognition

Classification Systems and Inductive Learning

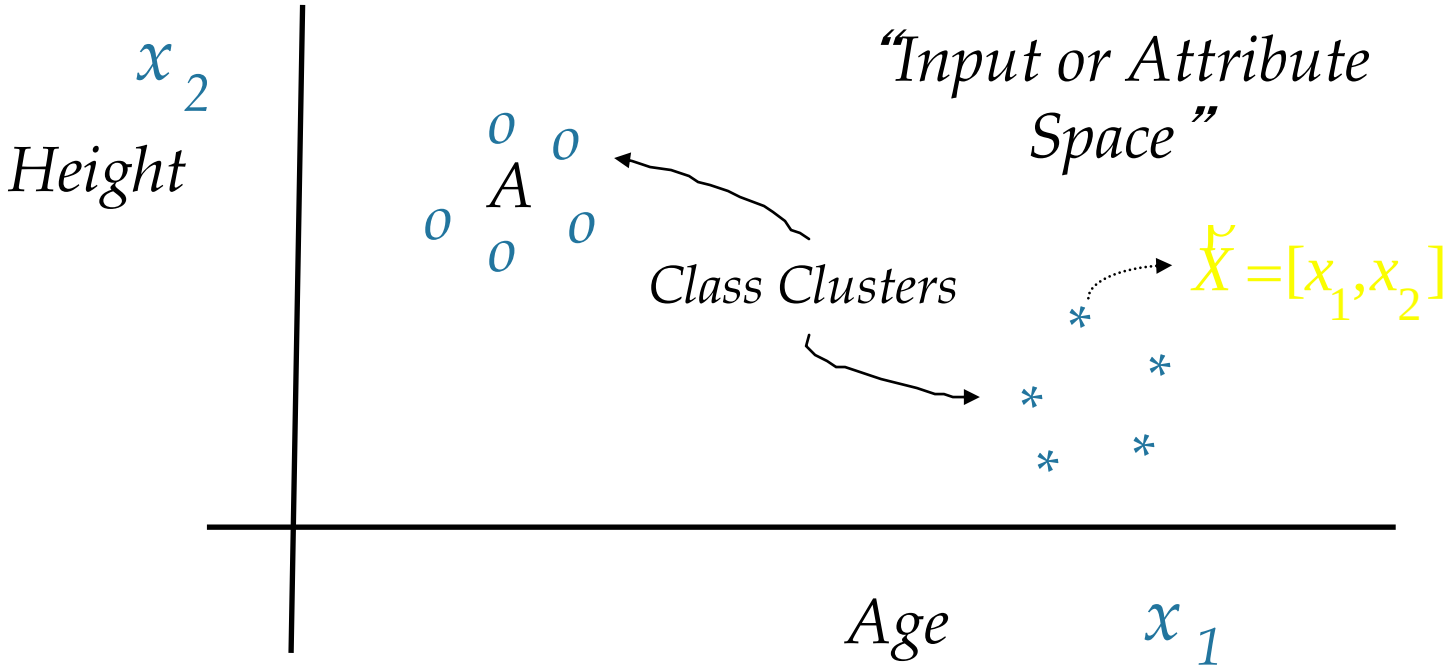
Basic Framework for Inductive Learning



A problem of representation and search for the best hypothesis, h(x).

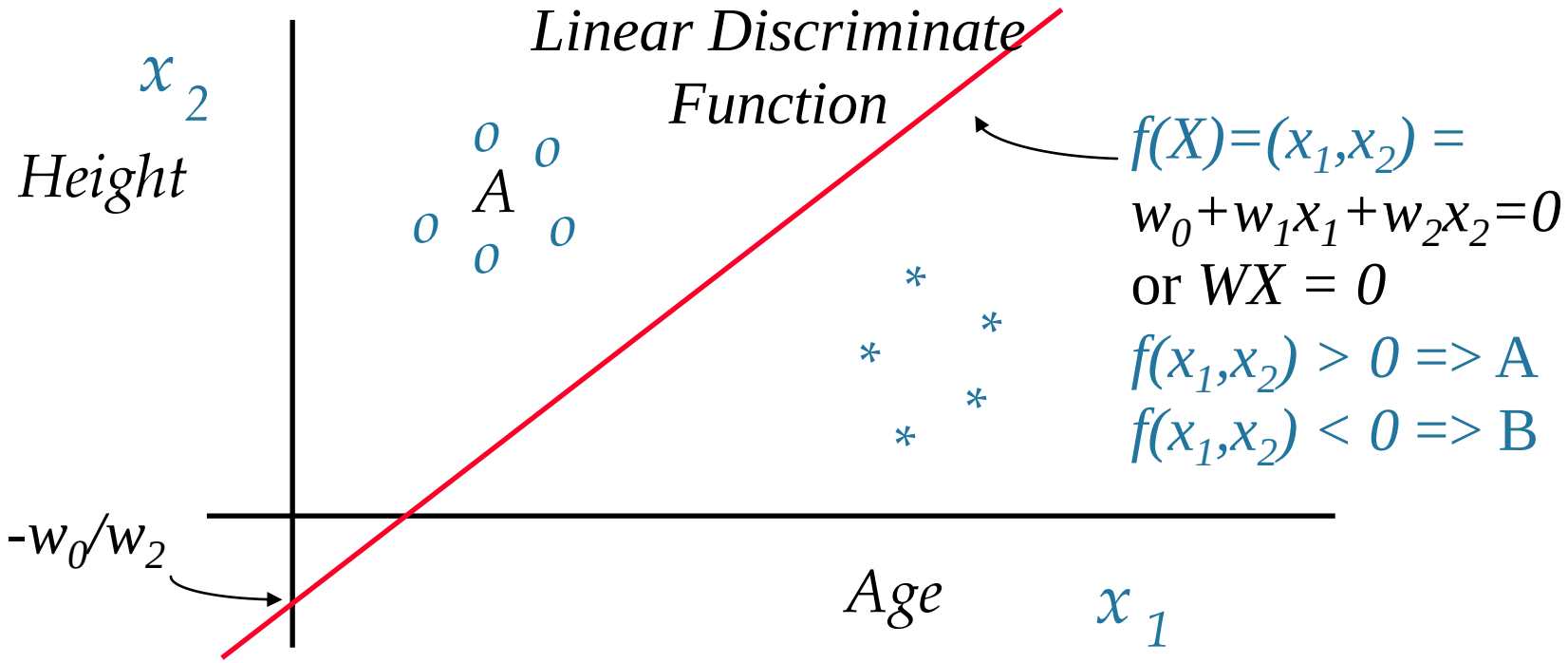
Classification Systems and Inductive Learning

Vector Representation & Discriminate Functions



Classification Systems and Inductive Learning

Vector Representation & Discriminate Functions



Classification Systems and Inductive Learning

- $f(X) = WX = 0$ will discriminate class A from B,
- BUT … we do not know the appropriate values for :

$$w_0, w_1, w_2$$

Classification Systems and Inductive Learning

We will consider one family of neural network classifiers:

- continuous valued input
- feed-forward
- supervised learning
- global error

From Biological to Artificial Neurons

The Neuron - A Biological Information Processor

- *dentrites* - the receivers
- *soma* - neuron cell body (sums input signals)
- *axon* - the transmitter
- *synapse* - point of transmission
- neuron activates after a certain *threshold* is met

Learning occurs via electro-chemical changes in effectiveness of *synaptic junction*.

From Biological to Artificial Neurons

An Artificial Neuron - The Perceptron

- simulated on hardware or by software
- input connections - the receivers
- *node*, *unit*, or *PE* simulates neuron body
- output connection - the transmitter
- *activation function* employs a threshold or *bias*
- *connection weights* act as synaptic junctions

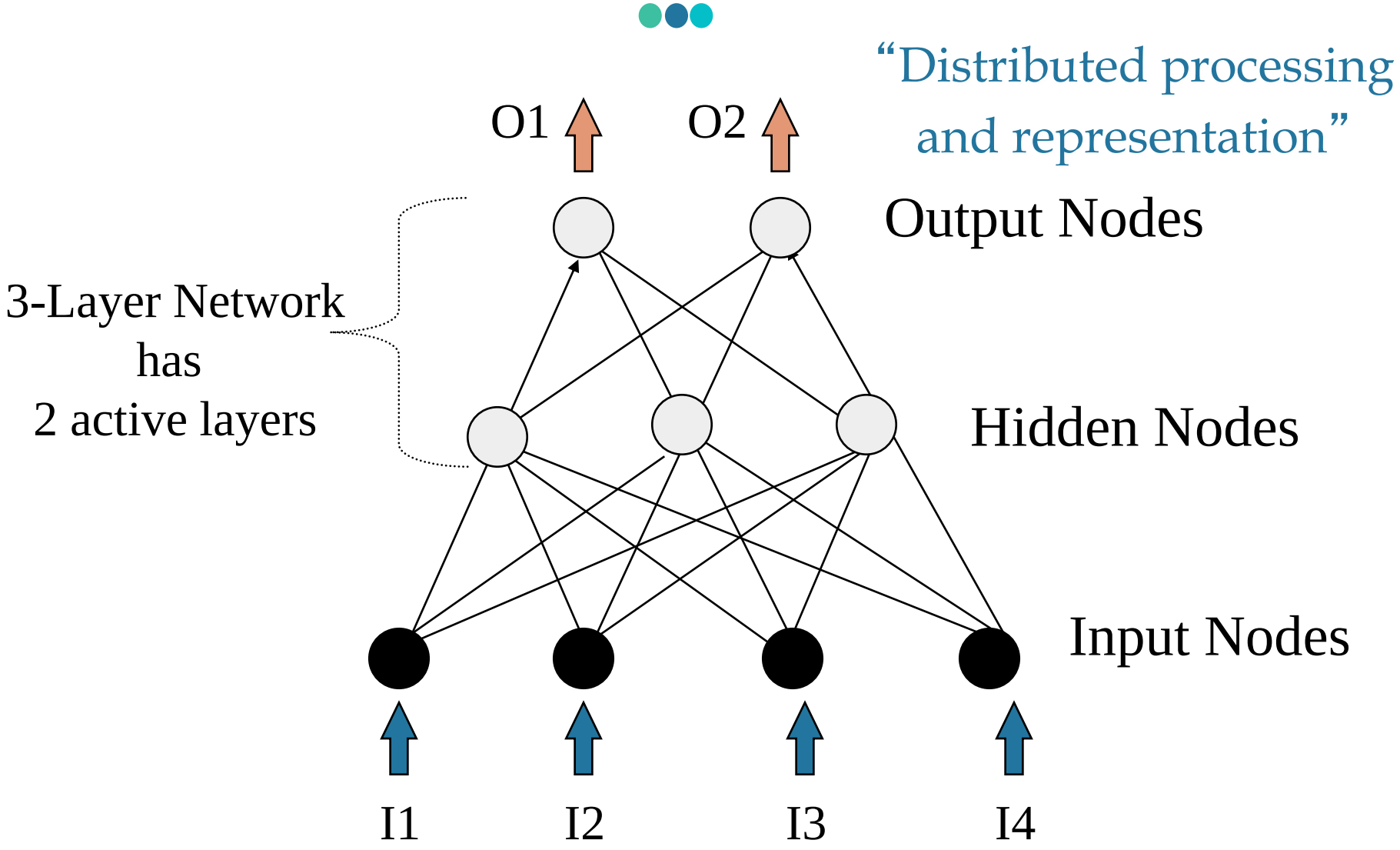
Learning occurs via changes in value of the connection weights.

From Biological to Artificial Neurons

An Artificial Neuron - The Perceptron

- Basic function of neuron is to sum inputs, and produce output given sum is greater than threshold
- ANN node produces an output as follows:
  1. Multiplies each component of the input pattern by the weight of its connection
  2. Sums all weighted inputs and subtracts the threshold value => *total weighted input*
  3. Transforms the total weighted input into the output using the activation function

From Biological to Artificial Neurons



From Biological to Artificial Neurons

Behaviour of an artificial neural network to any particular input depends upon:

- structure of each node (activation function)
- structure of the network (architecture)
- weights on each of the connections

.... these must be learned !

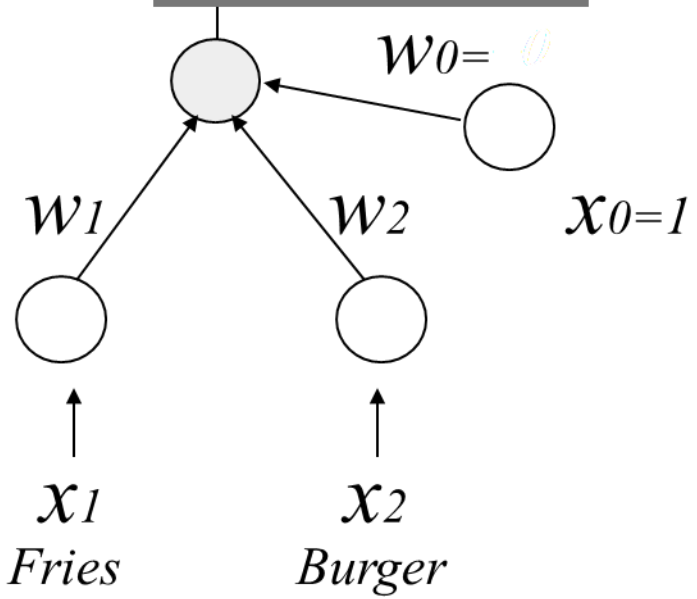
Learning in a Simple Neuron

"Full Meal Deal"

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

$$y = f\left[\sum_{i=0}^{2} w_i x_i\right]$$

where f(a) is the step function, such that: $f(a)=1,\ a > 0$; $f(a)=0,\ a <= 0$



$$H = \{W | W \in R^{(n+1)}\}$$

Learning in a Simple Neuron

Perceptron Learning Algorithm:

1. Initialize weights
2. Present a pattern and target output
3. Compute output :
4. Update weights :

$$y = f\left[\sum_{i=0}^{2} w_i x_i\right]$$

$$w_i(t+1) = w_i(t) + \Delta w_i$$

Repeat starting at 2 until acceptable level of error

Learning in a Simple Neuron

Widrow-Hoff or Delta Rule for Weight Modification

$$w_i(t+1)=w_i(t)+\Delta w_i \quad ; \quad \Delta w_i=\eta d x_i(t)$$

Where:

η = *learning rate* (*o* < η <= *1*), typically set = *0.1*

d = *error signal* = desired output - network output

= *t* - *y*

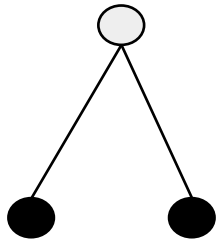
Limitations of Simple Neural Networks

What is a Perceptron doing when it learns?

- We will see it is often good to visualize network activity
- A discriminate function is generated
- Has the power to map input patterns to output class values
- For 3-dimensional input, must visualize 3-D space and 2-D *hyper-planes*

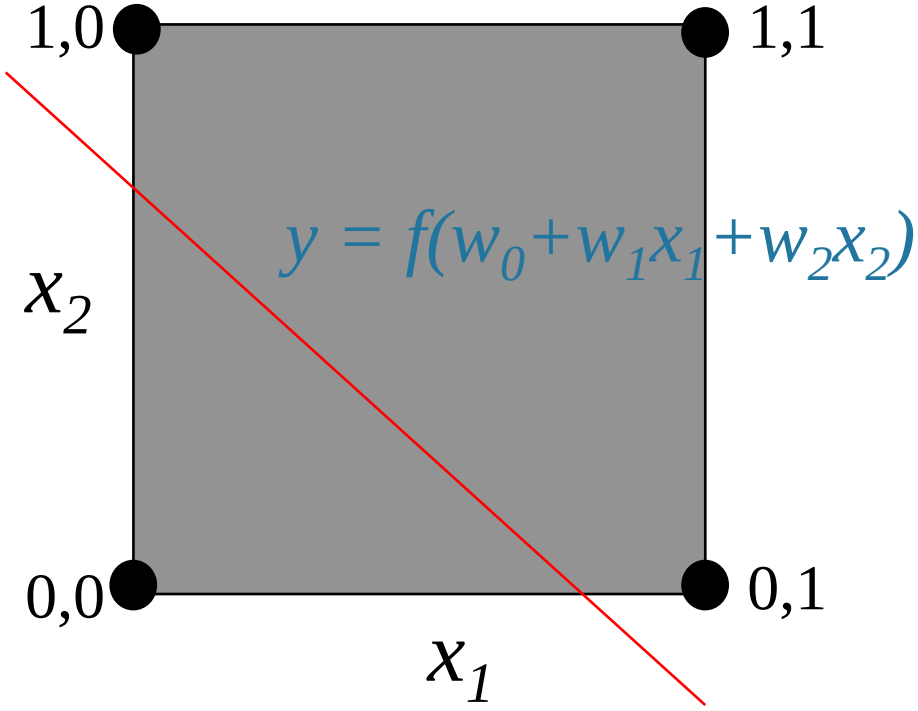
EXAMPLE

Simple
Neural Network



Logical OR
Function

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



What is an artificial neuron doing when it learns?

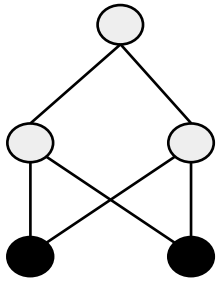
Limitations of Simple Neural Networks

The Limitations of Perceptrons (Minsky and Papert, 1969)

- Able to form only *linear discriminate functions*; *i.e.* classes which can be divided by a line or hyper-plane
- Most functions are more complex; *i.e.* they are *non-linear* or not *linearly separable*
- This crippled research in neural net theory for 15 years ....

EXAMPLE

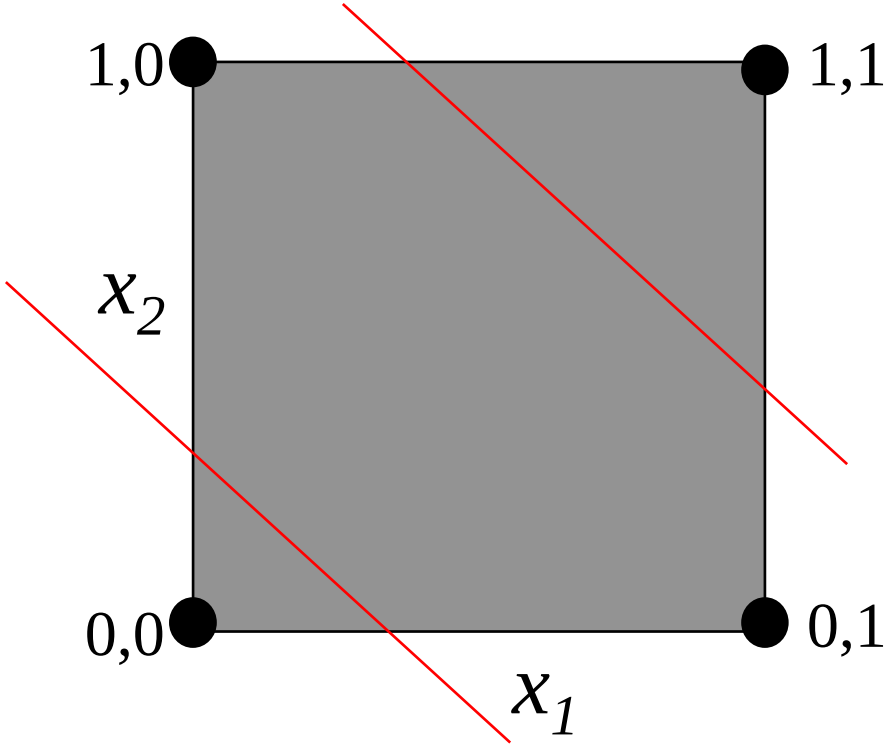
Multi-layer
Neural Network



Hidden layer
of neurons

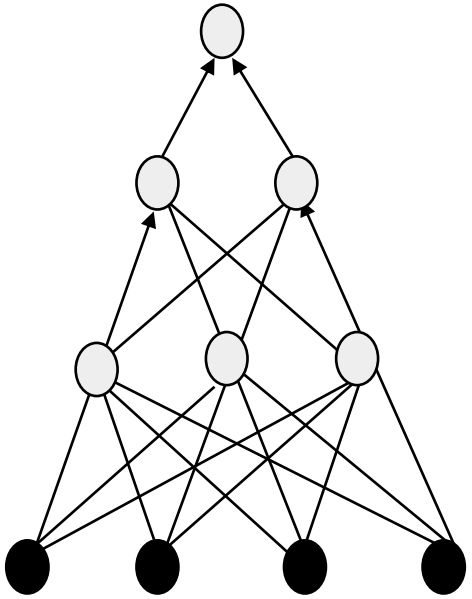
Logical XOR
Function

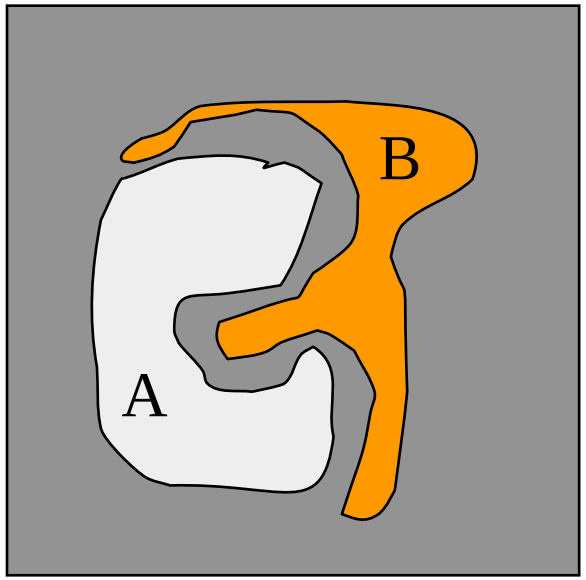
x1	*x2*	*y*
0	0	0
0	1	1
1	0	1
1	1	0



Two neurons are need! Their combined results can produce good classification.

EXAMPLE





More complex multi-layer networks are needed to solve more difficult problems.

Multi-layer Feed-forward ANNs

Over the 15 years (1969-1984) some research continued ...

- *hidden layer* of nodes allowed combinations of linear functions
- *non-linear activation functions* displayed properties closer to real neurons:
  - output varies continuously but not linearly
  - differentiable .... *sigmoid*

⇨ non-linear ANN classifier was possible

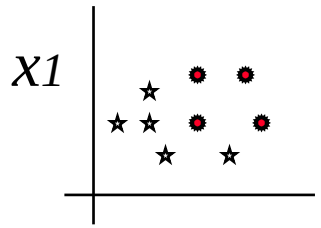
Multi-layer Feed-forward ANNs

However … there was no learning algorithm to adjust the weights of a multi-layer network - weights had to be set by hand.

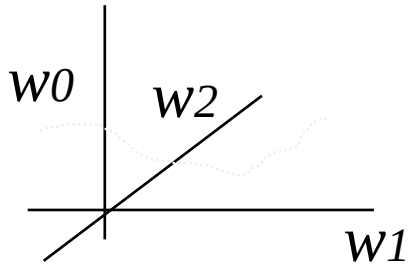
How could the weights below the hidden layer be updated?

Visualizing Network Behaviour

- Pattern Space



- Weight Space



- Visualizing the process of learning
  function surface in weight space
  error surface in weight space

The Back-propagation Algorithm

- 1986: the solution to multi-layer ANN weight update rediscovered
- Conceptually simple - the global error is backward propagated to network nodes, weights are modified proportional to their contribution
- Most important ANN learning algorithm
- Become known as *back-propagation* because the error is send back through the network to correct all weights

The Back-propagation Algorithm

- Like the Perceptron - calculation of error is based on difference between target and actual output:

$$E = \frac{1}{2}\sum_j (t_j - o_j)^2$$

- However in BP it is the rate of change of the error which is the important feedback through the network

generalized delta rule

$$\Delta w_{ij} = -\eta \frac{\delta E}{\delta w_{ij}}$$

- Relies on the sigmoid activation function for communication

The Back-propagation Algorithm

Objective: compute w_{ij} for all $\frac{\delta E}{\delta w_{ij}}$

Definitions:

x_j = weight from node i to node j

w_{ij} = totaled weighted input of node $= \sum_{i=0}^{n} w_{ij} o_i$

o_j = output of node $= f(x_j) = 1/(1+e^{-x_j})$

E = error for 1 pattern over all output nodes

The Back-propagation Algorithm

Objective: compute $\frac{\delta E}{\delta w_{ij}}$ for all w_{ij}

Four step process:

1. Compute how fast error changes as output of node *j* is changed
2. Compute how fast error changes as total input to node *j* is changed
3. Compute how fast error changes as weight w_{ij} coming into node *j* is changed
4. Compute how fast error changes as output of node *i* in previous layer is changed

The Back-propagation Algorithm

On-Line algorithm:

1. Initialize weights
2. Present a pattern and target output
3. Compute output : $o_j = f[\sum_{i=0}^{n} w_{ij} o_i]$
4. Update weights : $w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}$

where $\Delta w_{ij} = -\eta \frac{\delta E}{\delta w_{ij}}$

Repeat starting at 2 until acceptable level of error

The Back-propagation Algorithm

Where:

$$\Delta w_{ij} = \eta d_j o_i = -\eta \frac{\partial E}{\partial w_{ij}} = -\eta E W_{ij}$$

For output nodes:

$$d_j = EI_j = EA_j o_j (1 - o_j) = o_j (1 - o_j)(t_j - o_j)$$

For hidden nodes:

$$d_i = EI_i = EA_i o_i (1 - o_i) = o_i (1 - o_i) \sum_j EI_j w_{ij}$$

The Back-propagation Algorithm

Visualizing the bp learning process:

The bp algorithm performs a *gradient descent* in weights space toward a minimum level of error using a fixed *step size* or learning rate η

The gradient is given by : $\frac{\delta E}{\delta w_{ij}}$

= rate at which error changes as weights change

The Back-propagation Algorithm

Momentum Descent:

- Minimization can be speed-up if an additional term is added to the update equation:

$$\alpha[w_{ij}(t) - w_{ij}(t-1)]$$

where:

$$o < \alpha < 1$$

- Thus:

$$\Delta w_{ij}(t) = \eta d_j o_i + \alpha \Delta w_{ij}(t-1)$$

- Augments the effective learning rate η to vary the amount a weight is updated
- Analogous to momentum of a ball - maintains direction
- Rolls through small local minima
- Increases weight upadte when on stable gradient

The Back-propagation Algorithm

Line Search Techniques:

- Steepest and momentum descent use only gradient of error surface
- More advanced techniques explore the weight space using various *heuristics*
- Most common is to search ahead in the direction defined by the gradient

The Back-propagation Algorithm

On-line *vs.* Batch algorithms:

- Batch (or cumulative) method reviews a set of training examples known as an epoch and computes global error:

$$E = \frac{1}{2} \sum_{p} \sum_{j} (t_j - o_j)^2$$

- Weight updates are based on this cumulative error signal
- On-line more stochastic and typically a little more accurate, batch more efficient

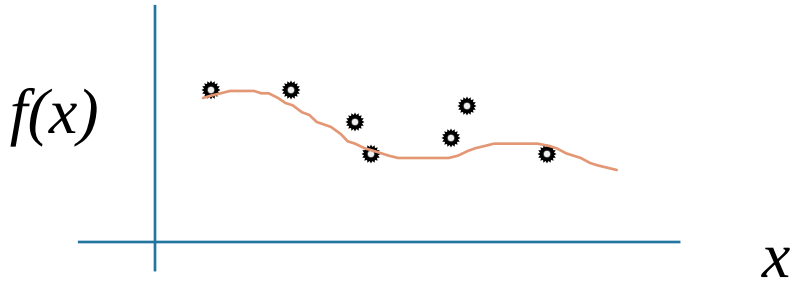
The Back-propagation Algorithm

Several Interesting Questions:

- What is BP's inductive bias?
- Can BP get stuck in local minimum?
- How does learning time scale with size of the network & number of training examples?
- Is it biologically plausible?
- Do we have to use the sigmoid activation function?
- How well does a trained network *generalize* to unseen test cases?

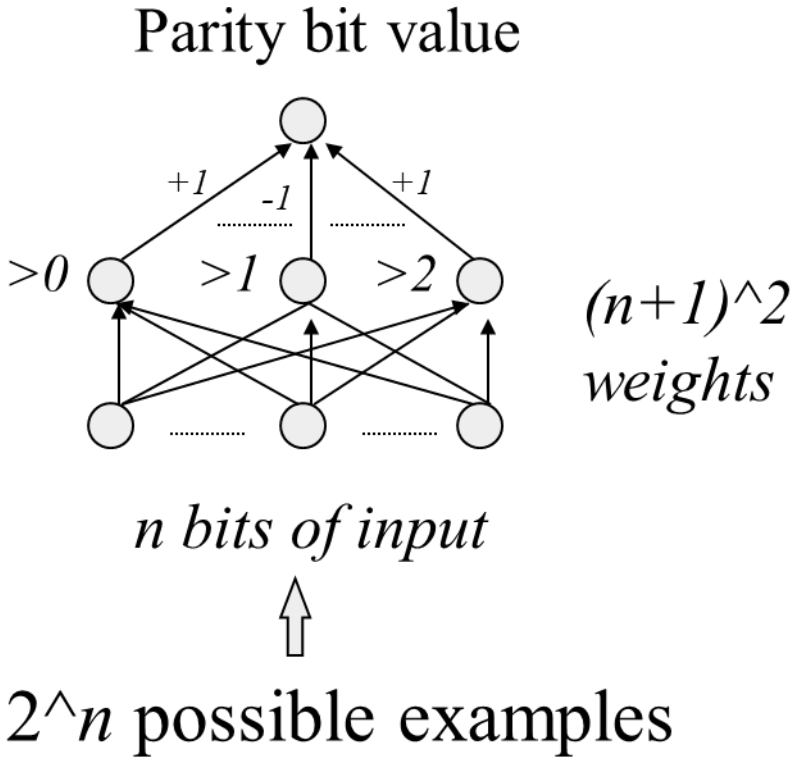
Generalization

- The objective of learning is to achieve good *generalization* to new cases, otherwise just use a look-up table.
- Generalization can be defined as a mathematical *interpolation* or *regression* over a set of training points:



Generalization

An Example: Computing Parity

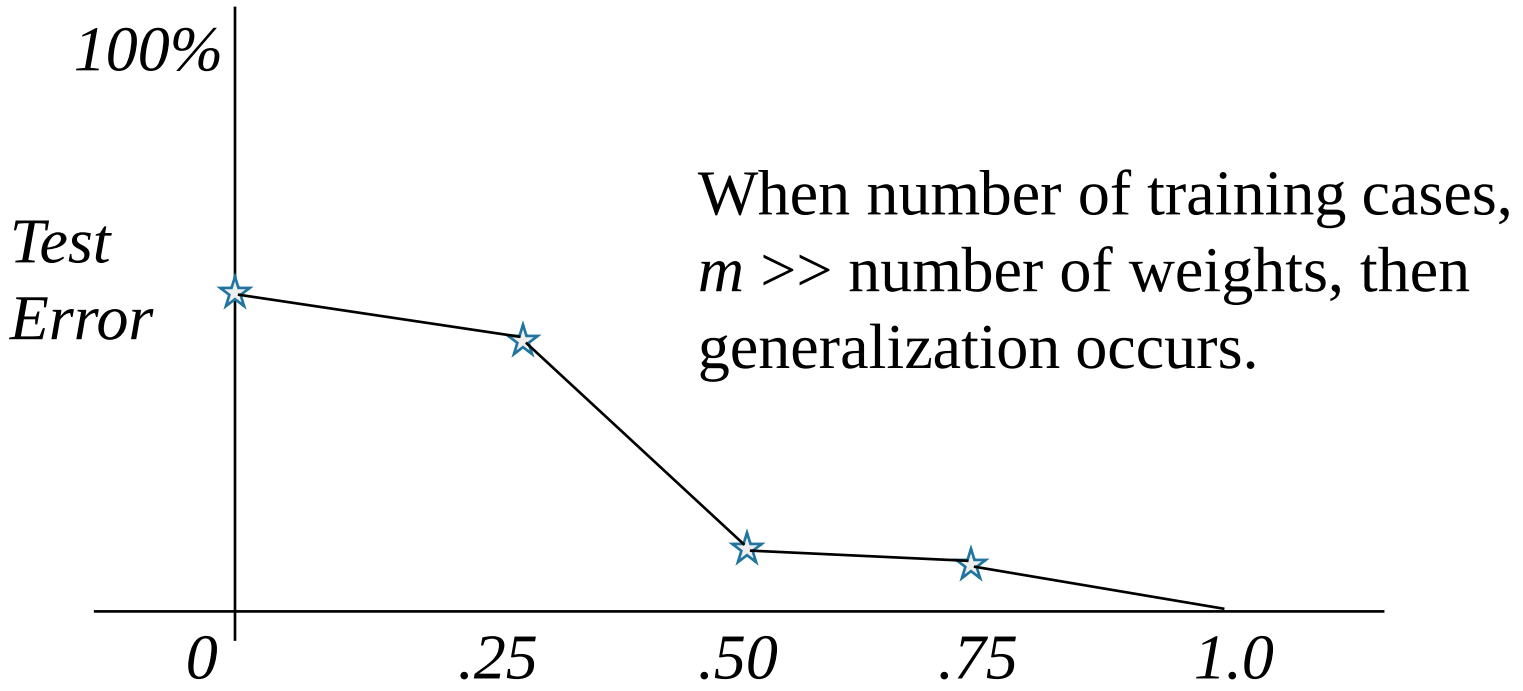


Can it learn from *m* examples to generalize to all 2^*n* possibilities?

Generalization

Network test of 10-bit parity

(Denker *et. al.*, 1987)



Generalization

A Probabilistic Guarantee

N = # hidden nodes m = # training cases

W = # weights $\mathcal{E}$ = error tolerance (< 1/8)

Network will generalize with 95% confidence if:

1. Error on training set < $\varepsilon / 2$
2. $$m > O(\frac{W}{\varepsilon}\log_2\frac{N}{\varepsilon}) \approx m > \frac{W}{\varepsilon}$$

Based on PAC theory => provides a good rule of practice.

Generalization

Consider 20-bit parity problem:

- 20-20-1 net has 441 weights
- For 95% confidence that net will predict with , we need $\leq \varepsilon = 0.1$

$$m > \frac{W}{\varepsilon} = \frac{441}{0.1} = 4410$$ training examples

- Not bad considering $2^{20} = 1{,}048{,}576$

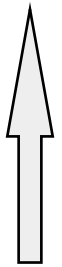
Generalization

Training Sample & Network Complexity

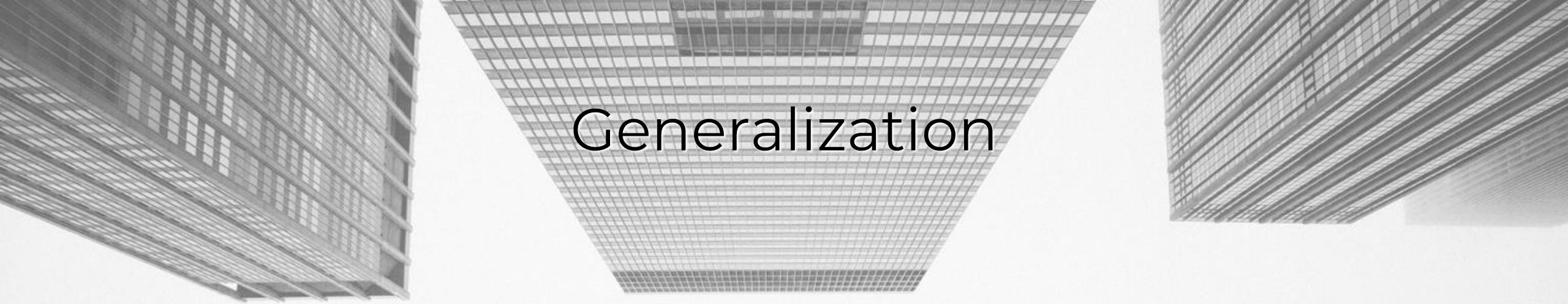
Based on $m > \frac{W}{\varepsilon}$

W - *to reduced size of training sample*

Optimum W => Optimum # Hidden Nodes



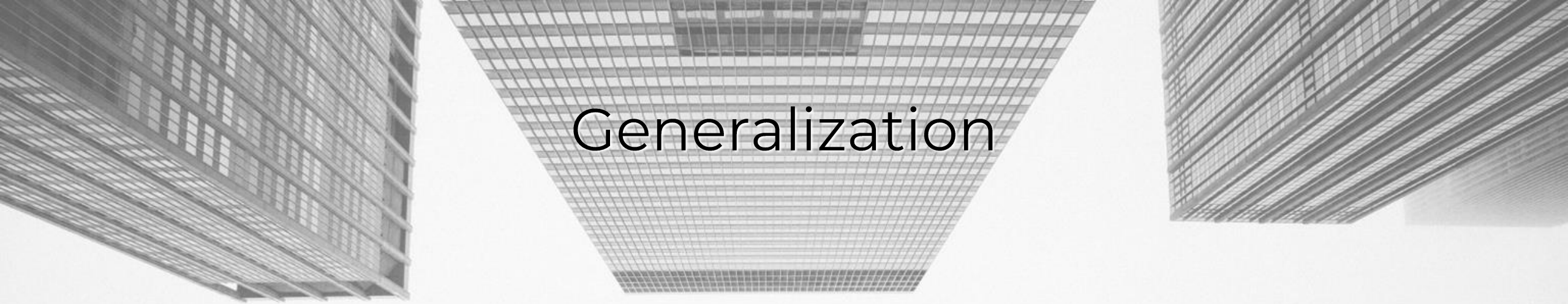
W - *to supply freedom to construct desired function*



Generalization

How can we control number of effective weights?

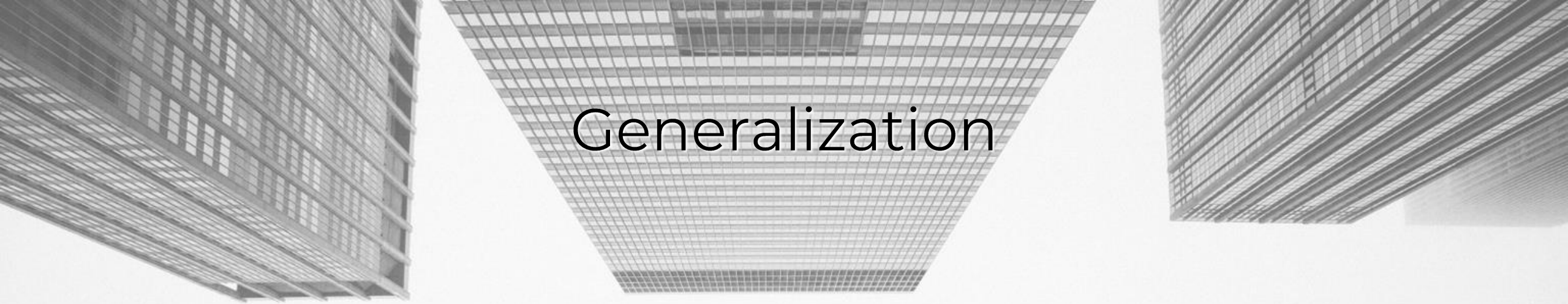
- Manually or automatically select optimum number of hidden nodes and connections
- Prevent *over-fitting* = *over-training*
- Add a *weight-cost* term to the bp error equation



Generalization

Over-Training

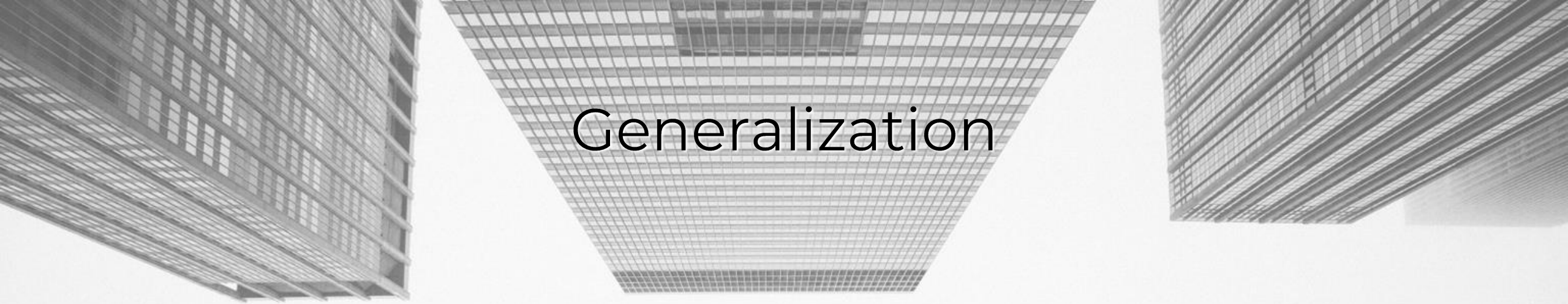
- Is the equivalent of over-fitting a set of data points to a curve which is too complex
- Occam's Razor (1300s) : *"plurality should not be assumed without necessity"*
- The simplest model which explains the majority of the data is usually the best



Generalization

Preventing Over-training:

- Use a separate *test* or *tuning set* of examples
- Monitor error on the test set as network trains
- Stop network training just prior to over-fit error occurring - *early stopping* or *tuning*
- Number of effective weights is reduced
- Most new systems have automated early stopping methods



Generalization

Weight Decay: an automated method of effective weight control

- Adjust the bp error function to penalize the growth of unnecessary weights:

$$E = \frac{1}{2}\sum_j (t_j - o_j)^2 + \frac{\lambda}{2}\sum_i w_{ij}^2 \quad \Rightarrow \quad \Delta w_{ij} = \Delta w_{ij} - \lambda w_{ij}$$

where: w_{ij} = weight -cost parameter

λ is decayed by an amount proportional to its magnitude; *those not reinforced => 0*

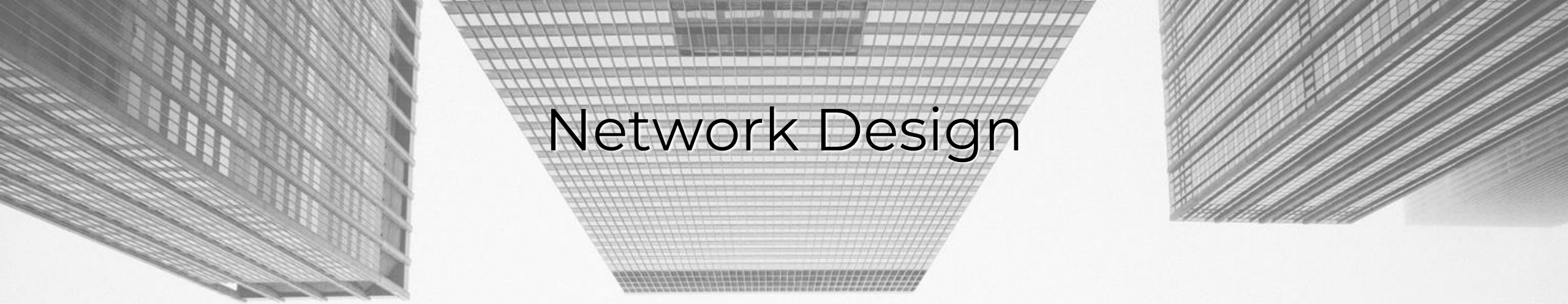
Network Design & Training Issues

Design:

- Architecture of network
- Structure of artificial neurons
- Learning rules

Training:

- Ensuring optimum training
- Learning parameters
- Data preparation
- and more ….



Network Design

Architecture of the network: How many nodes?

- Determines number of network weights
- How many layers?
- How many nodes per layer?

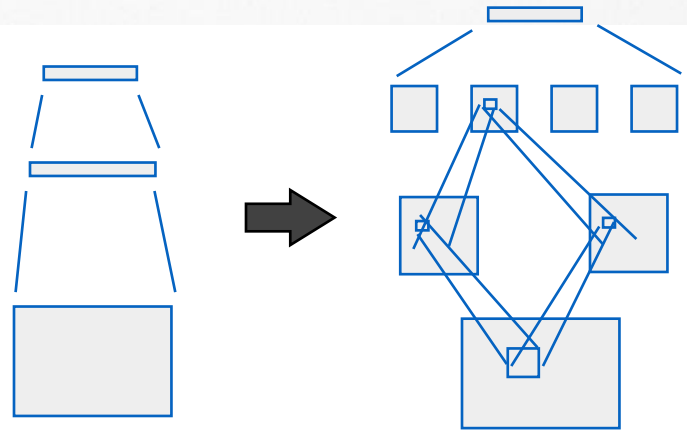
Input Layer Hidden Layer Output Layer

- Automated methods:
  - augmentation (cascade correlation)
  - weight pruning and elimination

Network Design

Architecture of the network: Connectivity?

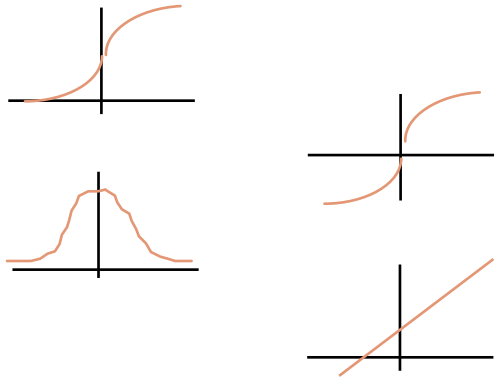
- Concept of model or *hypothesis* space
- Constraining the number of hypotheses:
  - selective connectivity
  - shared weights
  - recursive connections

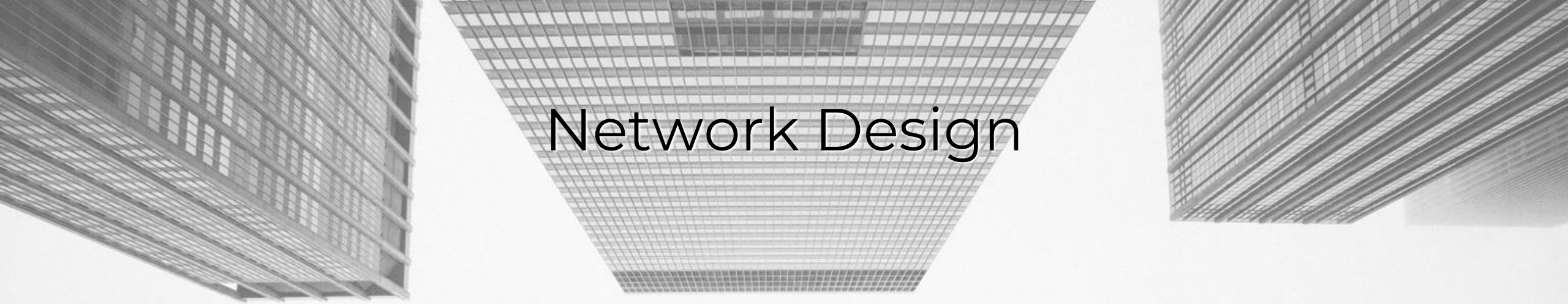


Network Design

Structure of artificial neuron nodes

- Choice of input integration:
  - summed, squared and summed
  - multiplied
- Choice of activation (transfer) function:
  - sigmoid (logistic)
  - hyperbolic tangent
  - Gaussian
  - linear
  - soft-max



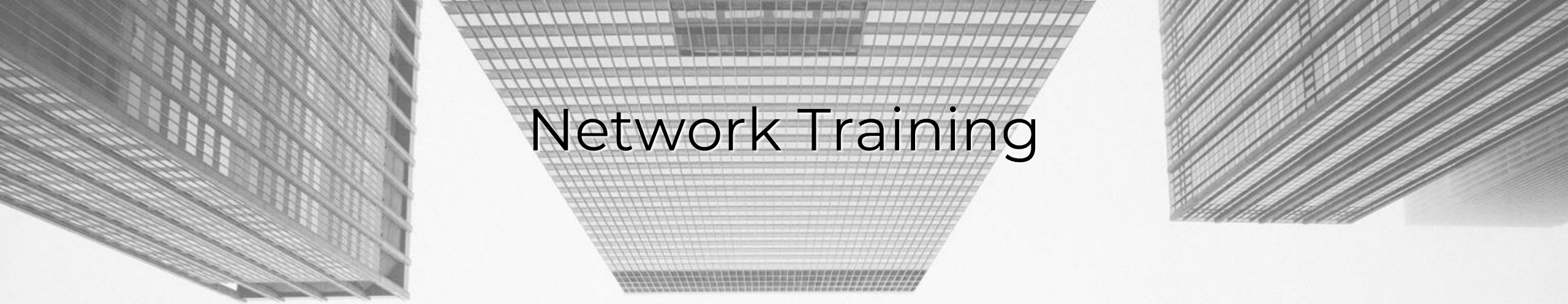


Network Design

Selecting a Learning Rule

- Generalized delta rule (steepest descent)
- Momentum descent
- Advanced weight space search techniques
- Global Error function can also vary

  - normal - quadratic - cubic



Network Training

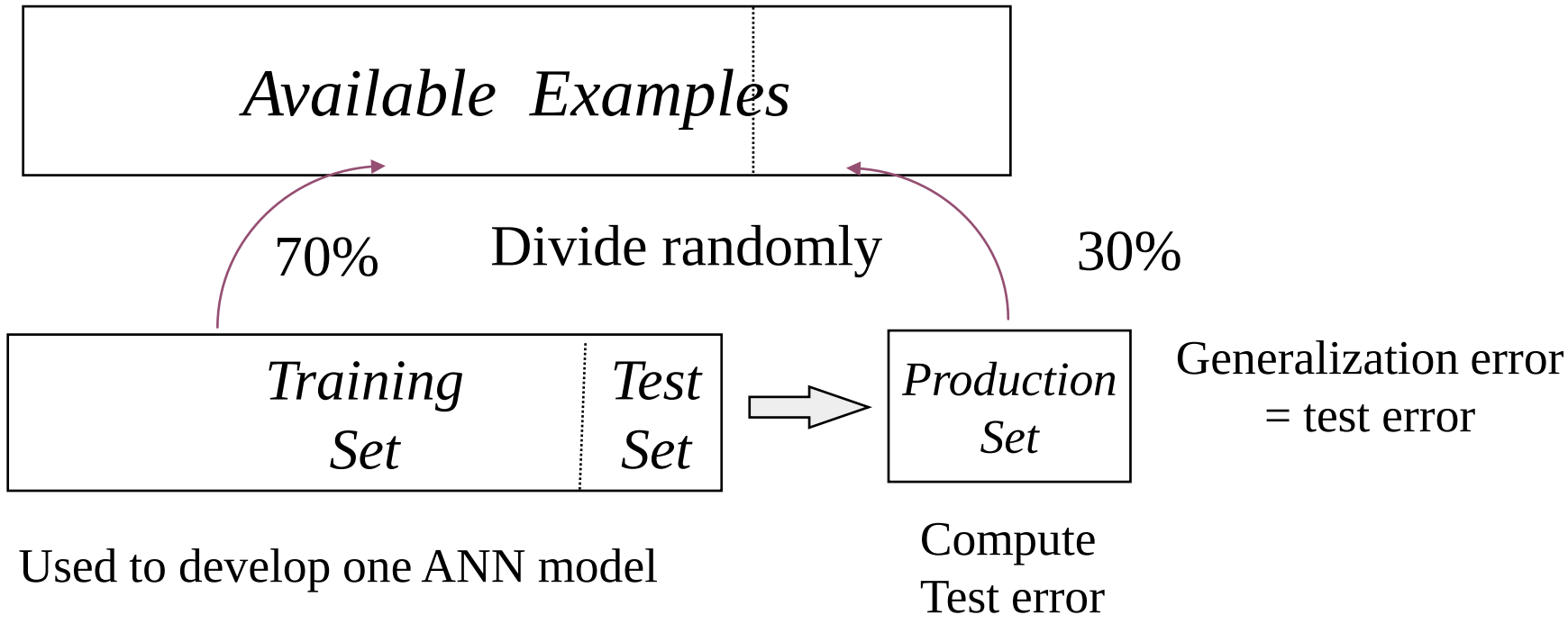
How do you ensure that a network has been well trained?

- Objective: To achieve good generalization
  accuracy on new examples/cases
- Establish a maximum acceptable error rate
- Train the network using a *validation test set* to tune it
- Validate the trained network against a separate test set which is usually referred to as a *production test set*

Network Training

Approach #1: Large Sample

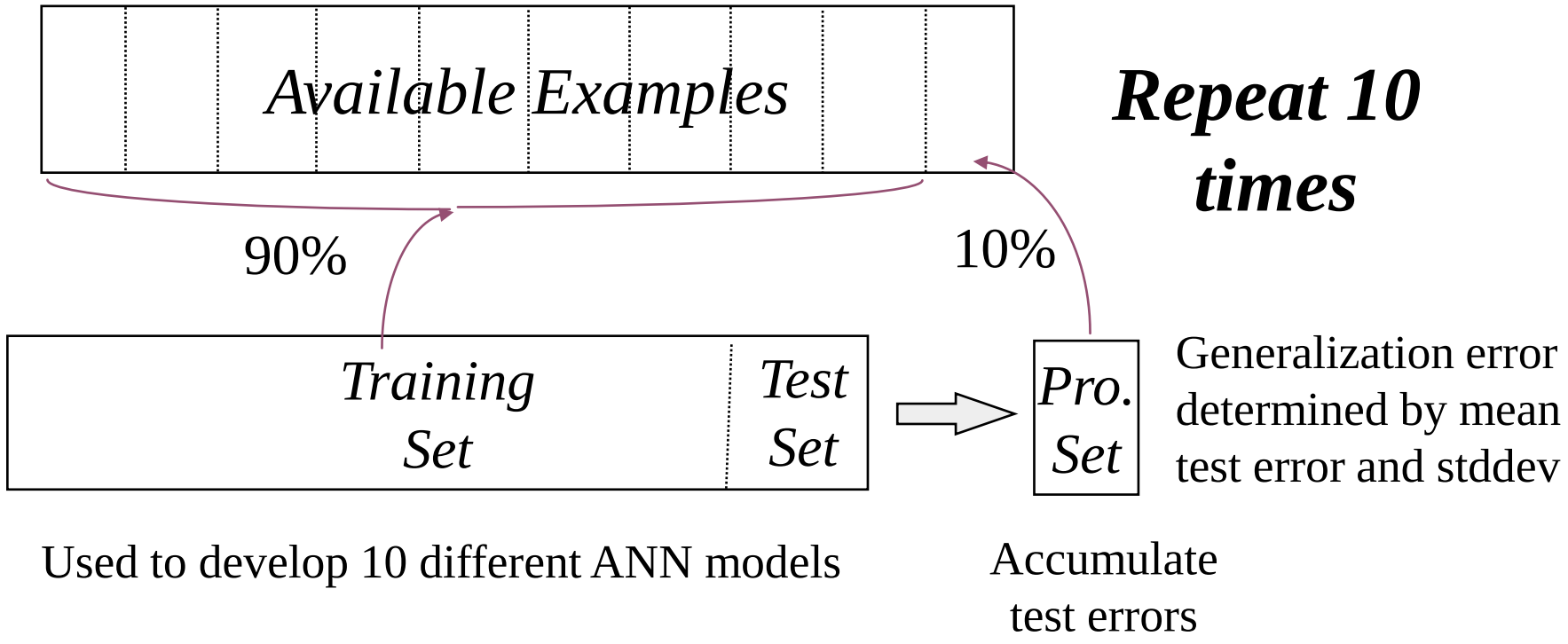
When the amount of available data is large ...

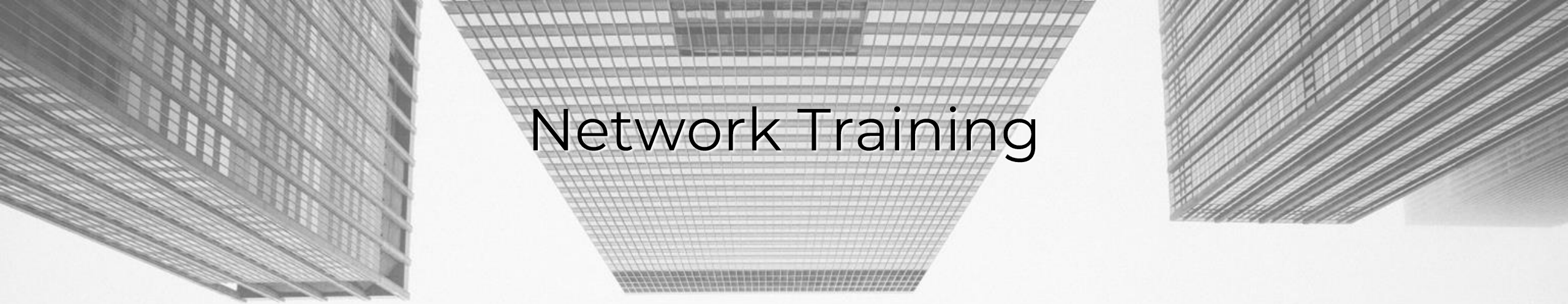


Network Training

Approach #2: Cross-validation

When the amount of available data is small …





Network Training

How do you select between two ANN designs ?

- A statistical test of hypothesis is required to ensure that a significant difference exists between the error rates of two ANN models
- If Large Sample method has been used then apply McNemar's test*
- If Cross-validation then use a paired *t* test for difference of two proportions

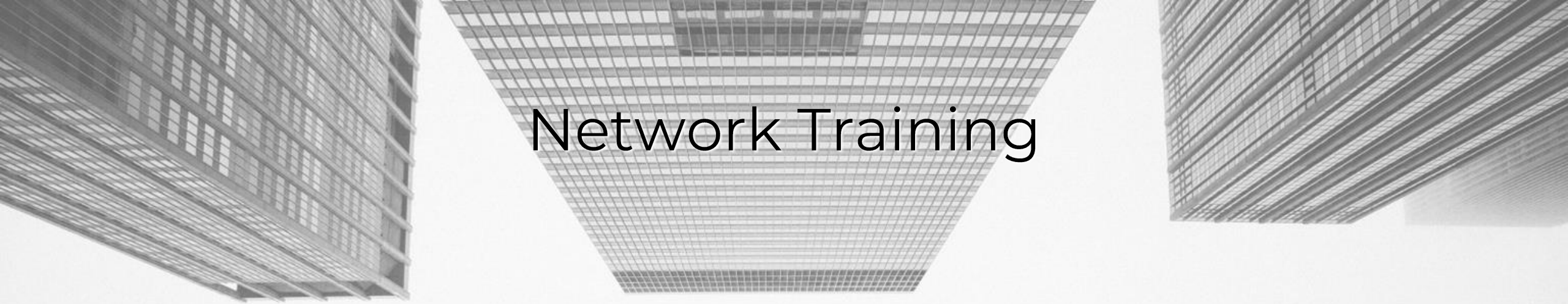
*We assume a classification problem, if this is function approximation then use paired *t* test for difference of means

Network Training

Mastering ANN Parameters

		Typical	Range
learning rate -	η	0.1	0.01 - 0.99
momentum -	α	0.8	0.1 - 0.9
weight-cost -	λ	0.1	0.001 - 0.5

Fine tuning : - adjust individual parameters at each node and/or connection weight automatic adjustment during training



Network Training

Network weight initialization

- Random initial values +/- some range
- Smaller weight values for nodes with many incoming connections
- Rule of thumb: initial weight range should be approximately

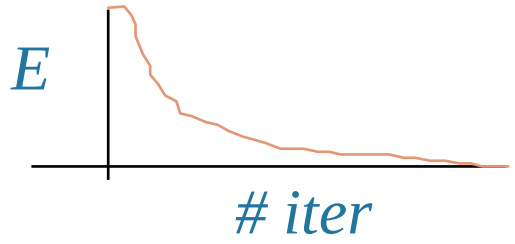
$$\pm \frac{1}{\#\,weights}$$

  coming into a node

Network Training

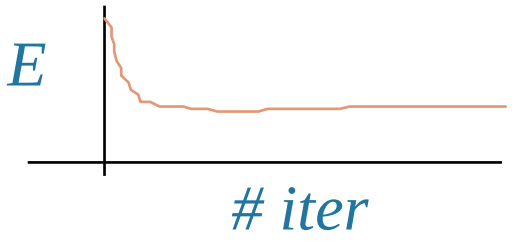
Typical Problems During Training

Would like:

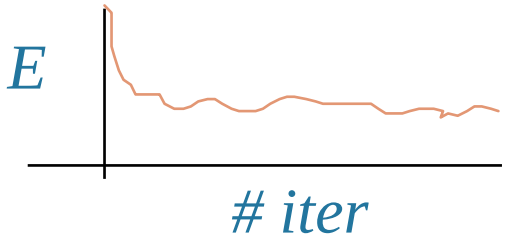


Steady, rapid decline in total error

But sometimes:



Seldom a local minimum - reduce learning or momentum parameter



Reduce learning parms. - may indicate data is not learnable

Data Preparation

Garbage in ⇨ Garbage out

- The quality of results relates directly to quality of the data
- 50%-70% of ANN development time will be spent on data preparation
- The three steps of data preparation:
  - Consolidation and Cleaning
  - Selection and Preprocessing
  - Transformation and Encoding

Data Preparation

Data Types and ANNs

- Four basic data types:
  - *nominal* discrete symbolic (blue,red,green)
  - *ordinal* discrete ranking (1st, 2nd, 3rd)
  - *interval* measurable numeric (-5, 3, 24)
  - *continuous* numeric (0.23, -45.2, 500.43)
- bp ANNs accept only continuous numeric values (typically 0 - 1 range)

Data Preparation

Consolidation and Cleaning

- Determine appropriate input attributes
- Consolidate data into working database
- Eliminate or estimate missing values
- Remove *outliers* (obvious exceptions)
- Determine prior probabilities of categories and deal with *volume bias*

Data Preparation

Selection and Preprocessing

- Select examples ⇨ random sampling

  Consider number of training examples?

$$m > \frac{W}{\varepsilon}$$

- Reduce attribute dimensionality
  - remove redundant and/or correlating attributes
  - combine attributes (sum, multiply, difference)
- Reduce attribute value ranges
  - group symbolic discrete values
  - quantize continuous numeric values

Data Preparation

Transformation and Encoding

Nominal or Ordinal values

- Transform to discrete numeric values
- Encode the value 4 as follows:
  - one-of-N code (0 1 0 0 0) - five inputs
  - thermometer code (1 1 1 1 0) - five inputs
  - real value (0.4)* - one input if ordinal
- Consider relationship between values
  - (*single*, *married*, *divorce*) *vs.* (*youth*, *adult*, *senior*)

** Target values should be 0.1 - 0.9 , not 0.0 - 1.0 range*

Data Preparation

Transformation and Encoding

Interval or continuous numeric values

- De-correlate example attributes via normalization of values:
  - Euclidean: *n = x/sqrt(sum of all x^2)*
  - Percentage: *n = x/(sum of all x)*
  - Variance based: *n = (x - (mean of all x))/variance*
- Scale values using a linear transform if data is uniformly distributed or use non-linear (log, power) if skewed distribution

Data Preparation

Transformation and Encoding

Interval or continuous numeric values

Encode the value 1.6 as:

- Single real-valued number (0.16)* - OK!
- Bits of a binary number (010000) - BAD!
- one-of-N quantized intervals (0 1 0 0 0)
  - NOT GREAT! - discontinuities
- distributed (fuzzy) overlapping intervals
  (0.3 0.8 0.1 0.0 0.0) - BEST!

* Target values should be 0.1 - 0.9 , not 0.0 - 1.0 range

Post-Training Analysis

Examining the neural net model:

- Visualizing the constructed model
- Detailed network analysis

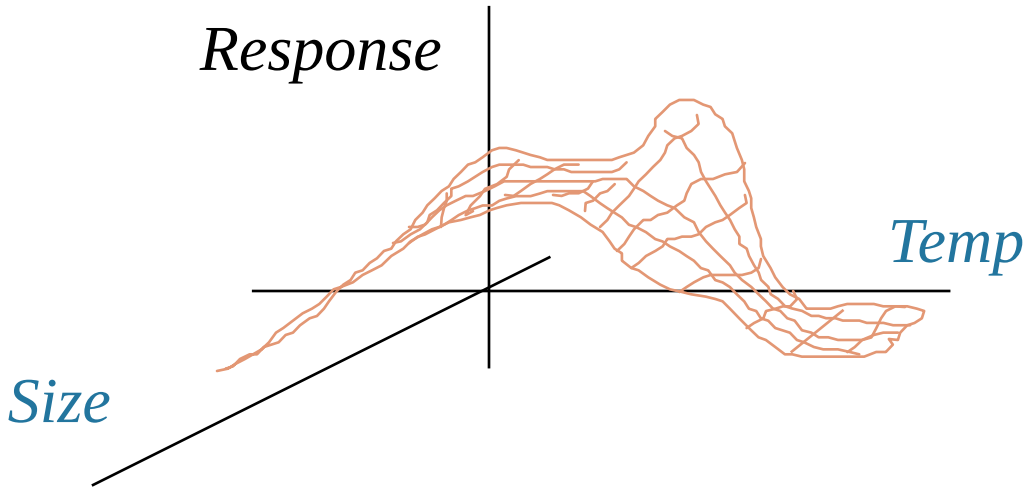
Sensitivity analysis of input attributes:

- Analytical techniques
- Attribute elimination

Post-Training Analysis

Visualizing the Constructed Model

- Graphical tools can be used to display output response as selected input variables are changed



Post-Training Analysis

Detailed network analysis

- Hidden nodes form internal representation
- Manual analysis of weight values often difficult - graphics very helpful
- Conversion to equation, executable code
- Automated ANN to symbolic logic conversion is a hot area of research

Post-Training Analysis

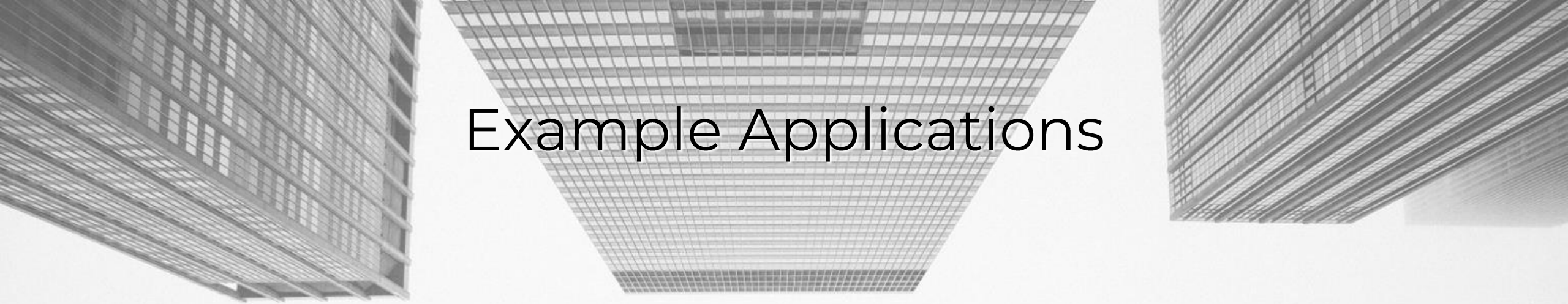
Sensitivity analysis of input attributes

- Analytical techniques
  - factor analysis
  - network weight analysis
- Feature (attribute) elimination
  - forward feature elimination
  - backward feature elimination

The ANN Application Development Process

Guidelines for using neural networks

1. Try the best existing method first
2. Get a **big** training set
3. Try a net without hidden units
4. Use a sensible coding for input variables
5. Consider methods of constraining network
6. Use a test set to prevent over-training
7. Determine confidence in generalization through cross-validation



Example Applications

- Pattern Recognition *(reading zip codes)*
- Signal Filtering *(reduction of radio noise)*
- Data Segmentation *(detection of seismic onsets)*
- Data Compression *(TV image transmission)*
- Database Mining *(marketing, finance analysis)*
- Adaptive Control *(vehicle guidance)*

Pros and Cons of Back-Prop

Cons:

- Local minimum - but not generally a concern
- Seems biologically implausible
- Space and time complexity:
  lengthy training times $O(W^3)$
- It's a black box! *I can't see how it's making decisions?*
- Best suited for supervised learning
- Works poorly on dense data with few input variables

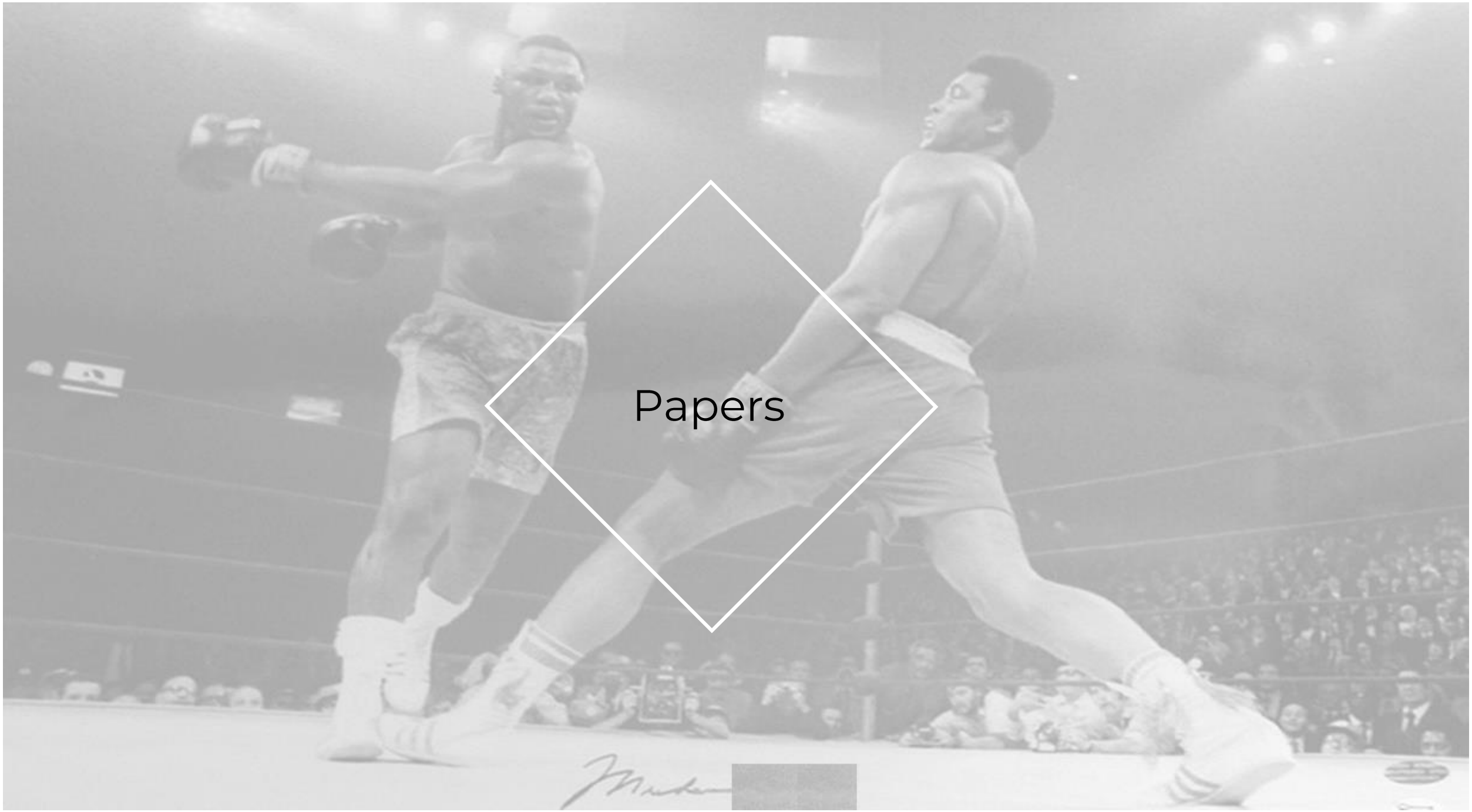
Pros and Cons of Back-Prop

Pros:

- Proven training method for multi-layer nets
- Able to learn any arbitrary function (XOR)
- Most useful for non-linear mappings
- Works well with noisy data
- Generalizes well given sufficient examples
- Rapid recognition speed
- Has inspired many new learning algorithms

Other Networks and Advanced Issues

- Variations in feed-forward architecture
  - jump connections to output nodes
  - hidden nodes that vary in structure
- Recurrent networks with feedback connections
- Probabilistic networks
- General Regression networks
- Unsupervised self-organizing networks



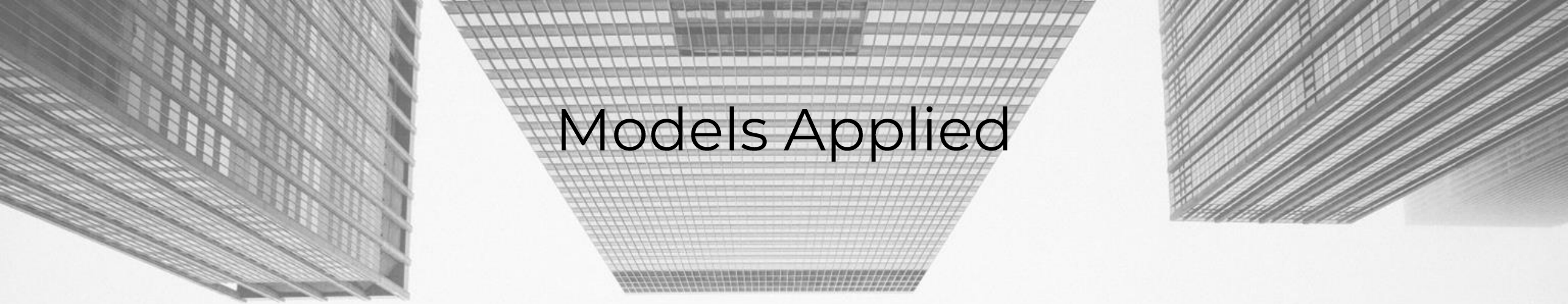
Papers

Objectives

- Added value of machine learning for risk management
- Computational learning theory behind
- Comparison of models for Credit Risk scoring
- Is that possible for authorities to supervise these models?

Data Used

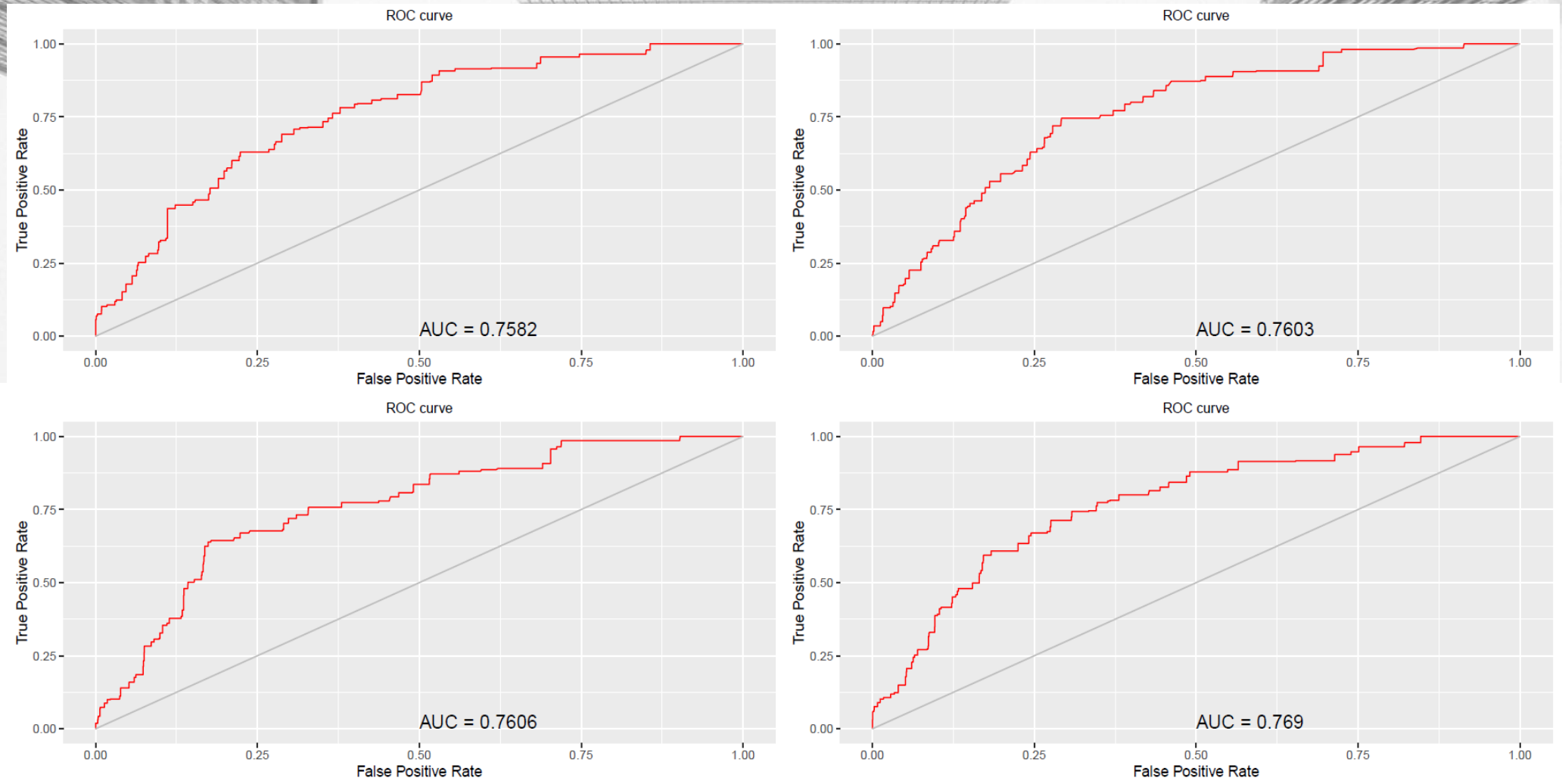
- 12 544 companies considered in the Sample
- European Perimeter
- 343 variables characterizing companies – turnover, margin etc. (Labelled data)
- 1 binary factor indicating if the company defaulted or not



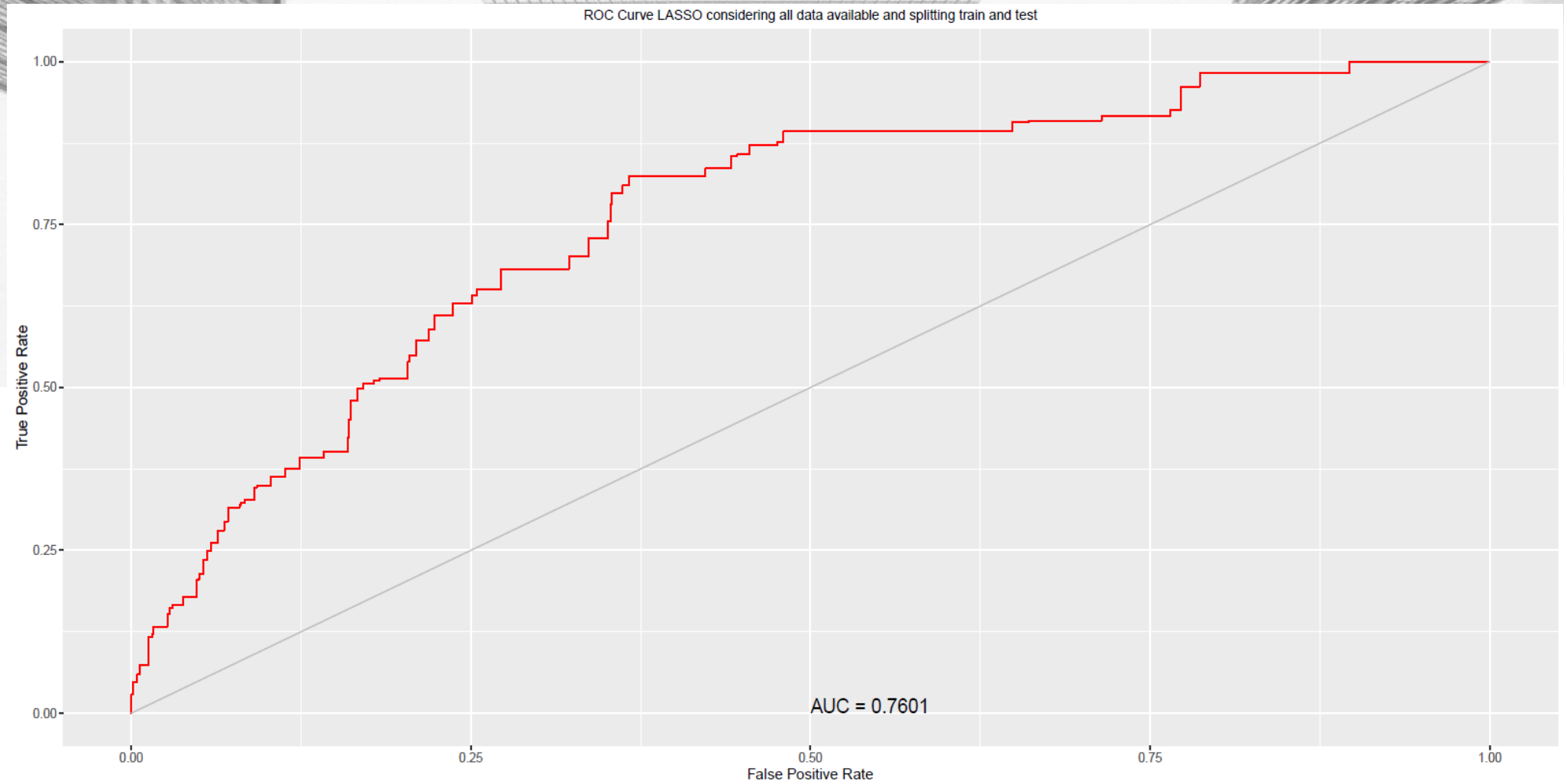
Models Applied

- Logistic Regression
- LASSO
- Random Forrest
- Gradiant Boosting(s)
- SVM
- Neural Network
- Deep Learning

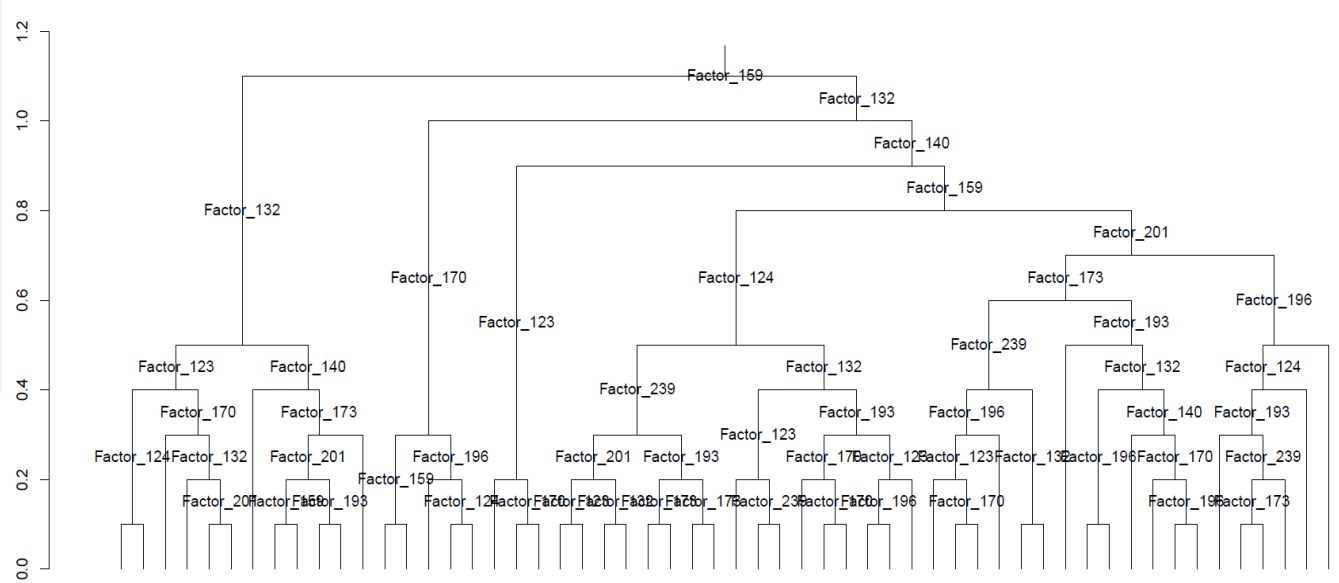
Logistic Regression (ROC Evolution)

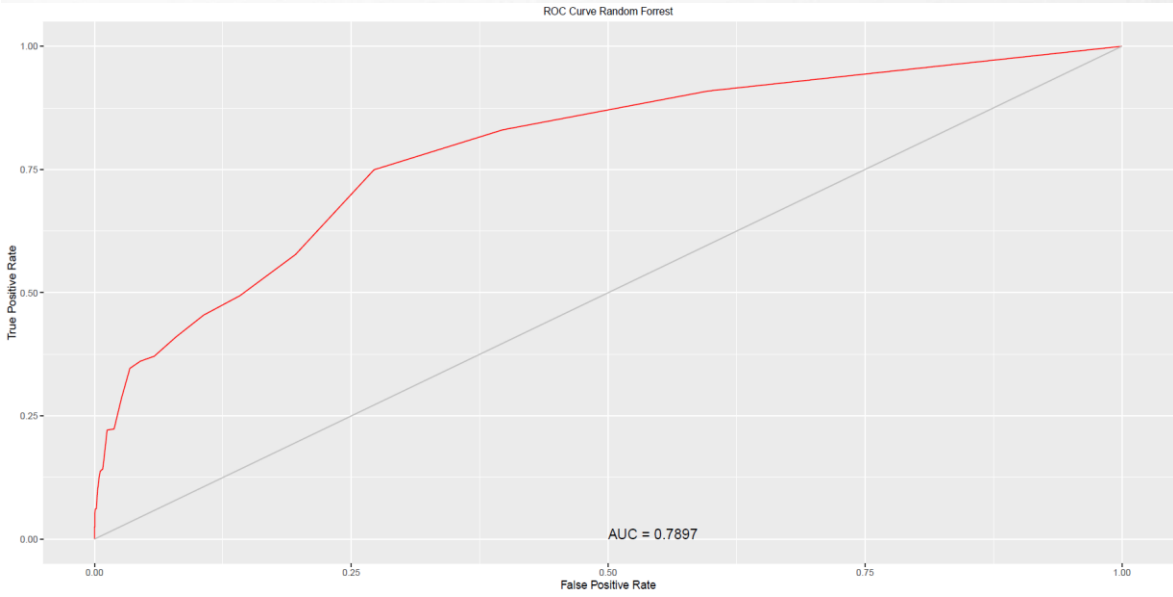


LASSO (ROC)

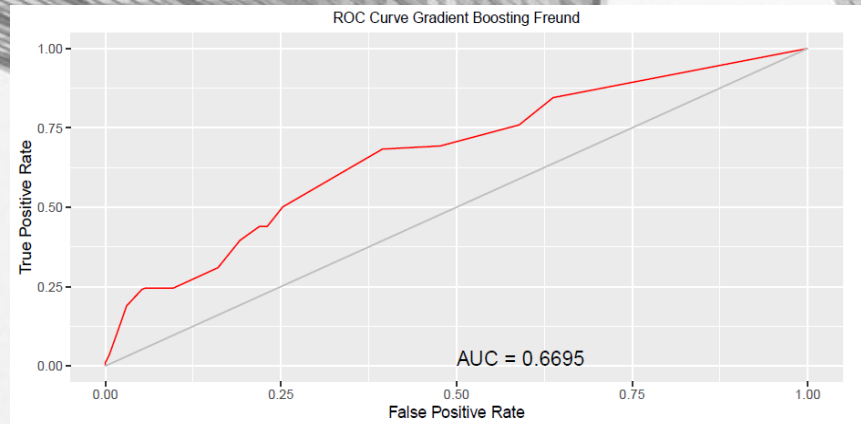

ROC Curve LASSO considering all data available and splitting train and test
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.7601
0.00
0.25
0.50
0.75
1.00
False Positive Rate

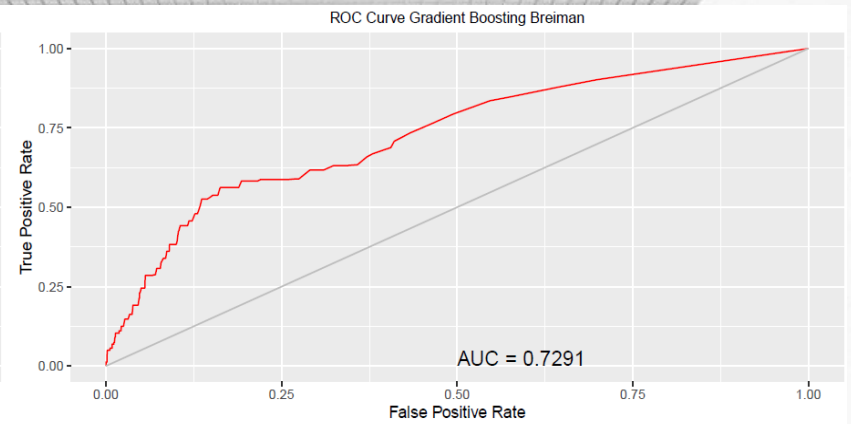
Random Forest (Extraction and ROC)

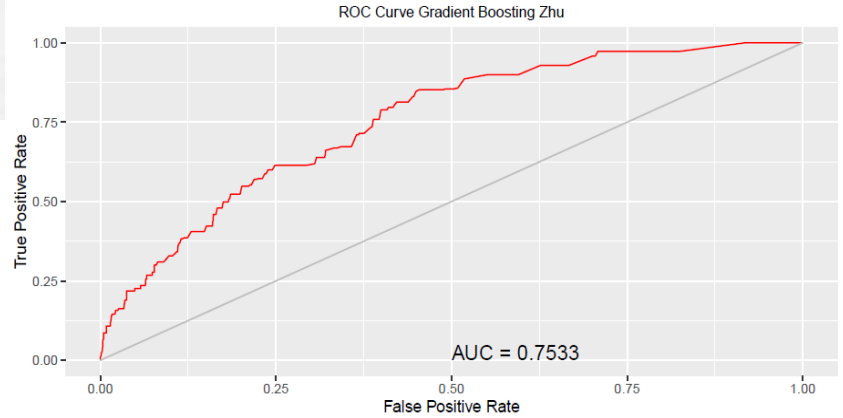


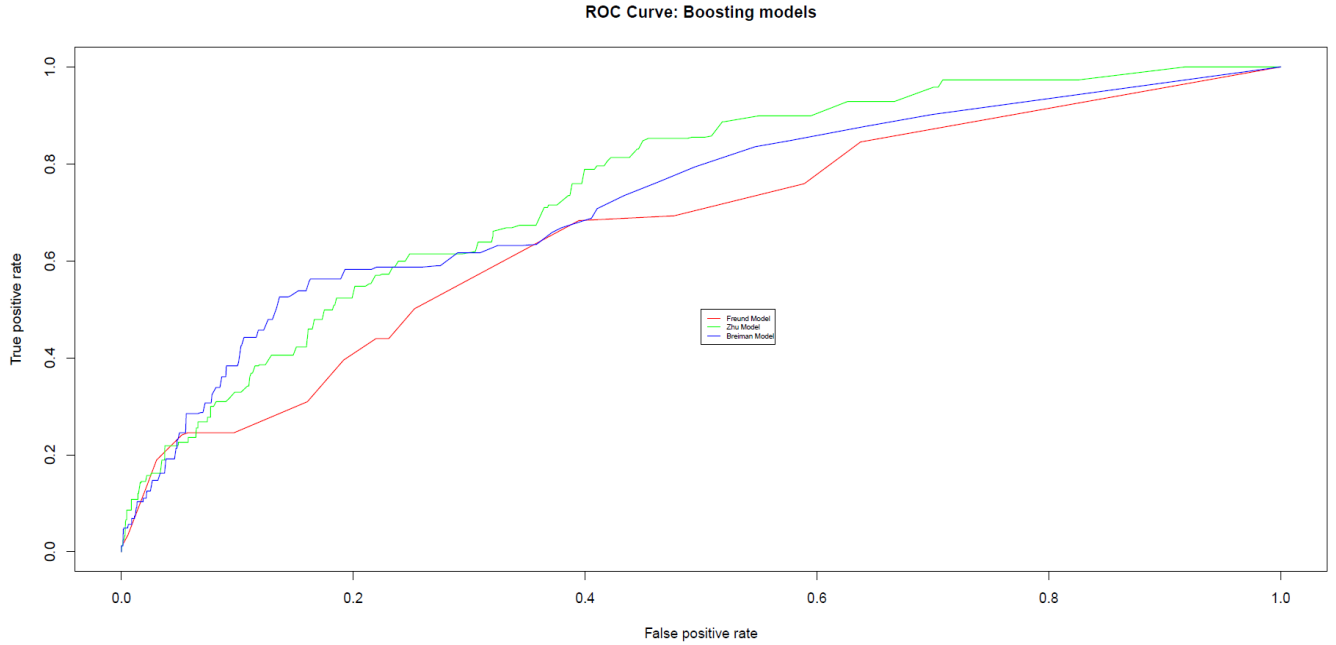


Gradient Boosting (ROC comparison)

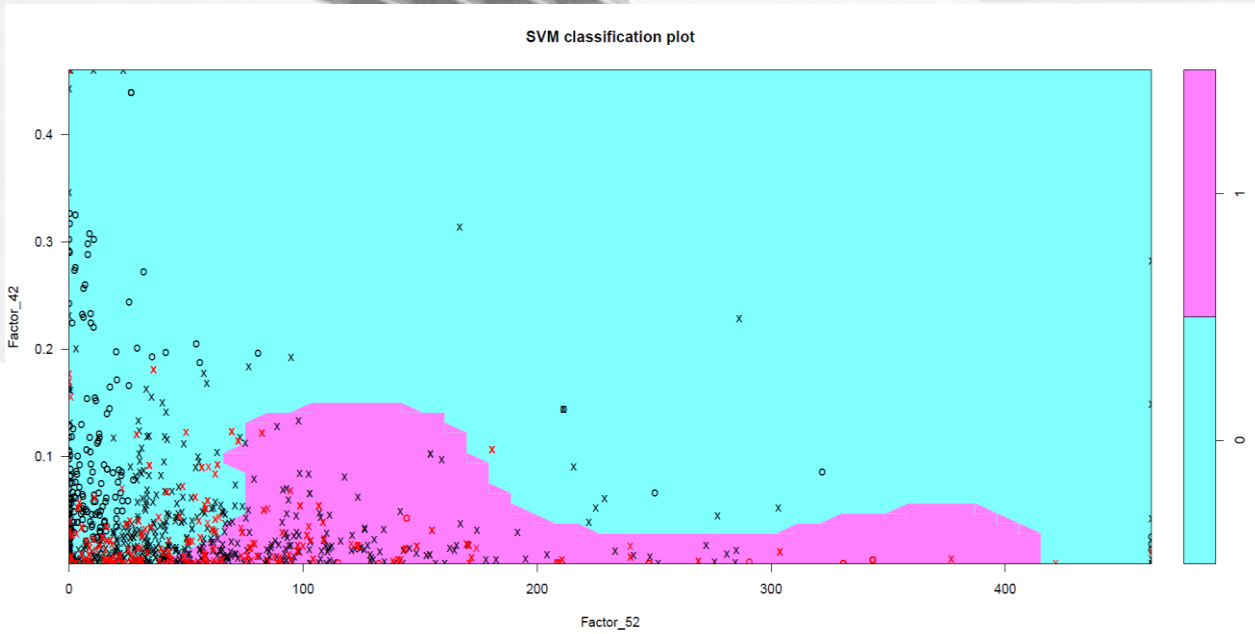


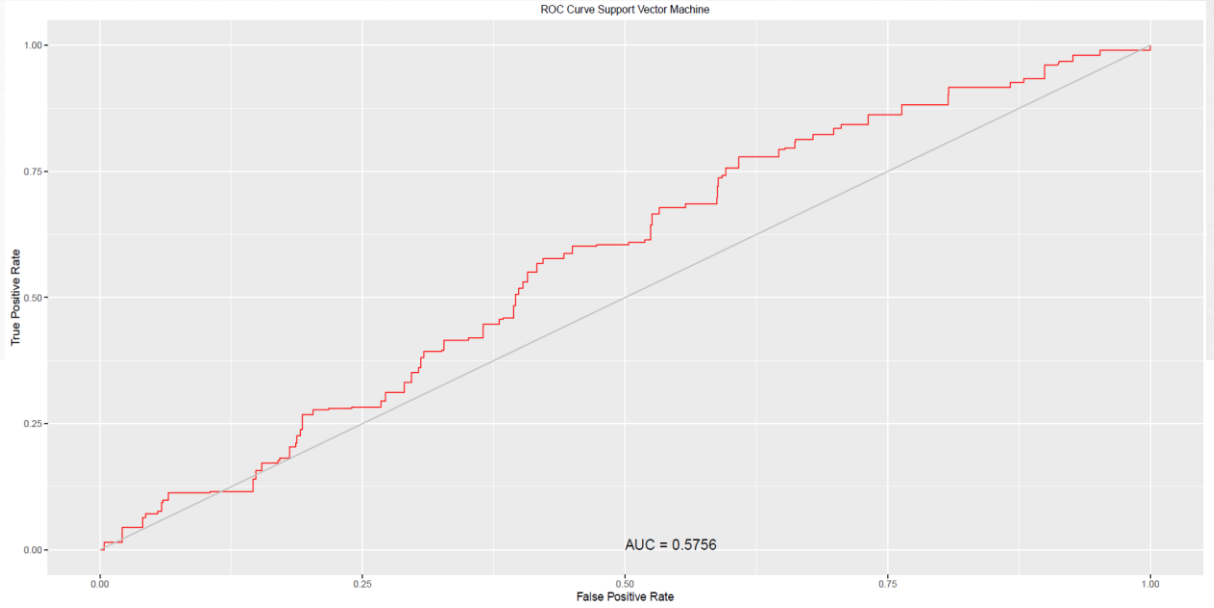




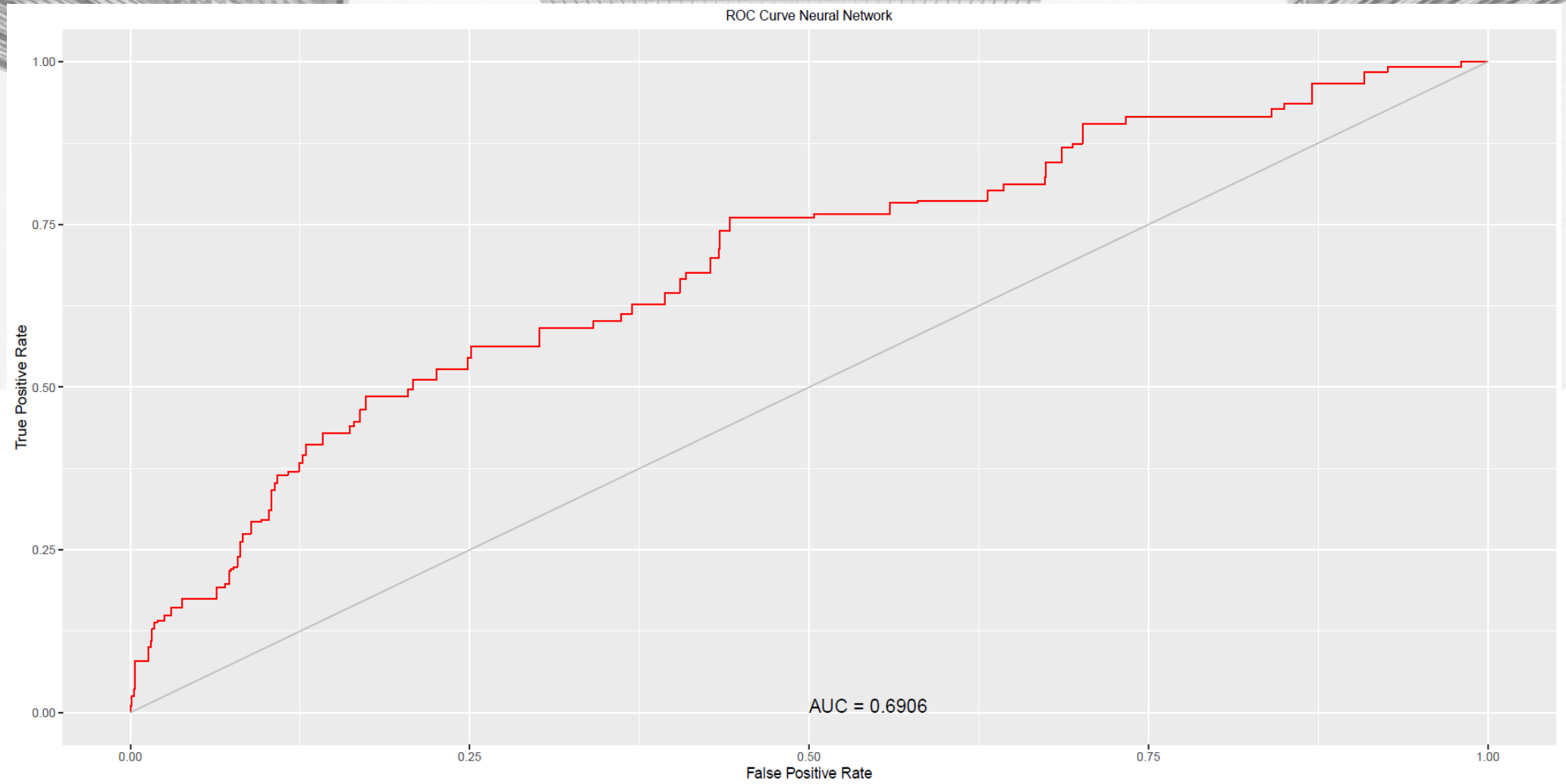


SVM (Representation and ROC)

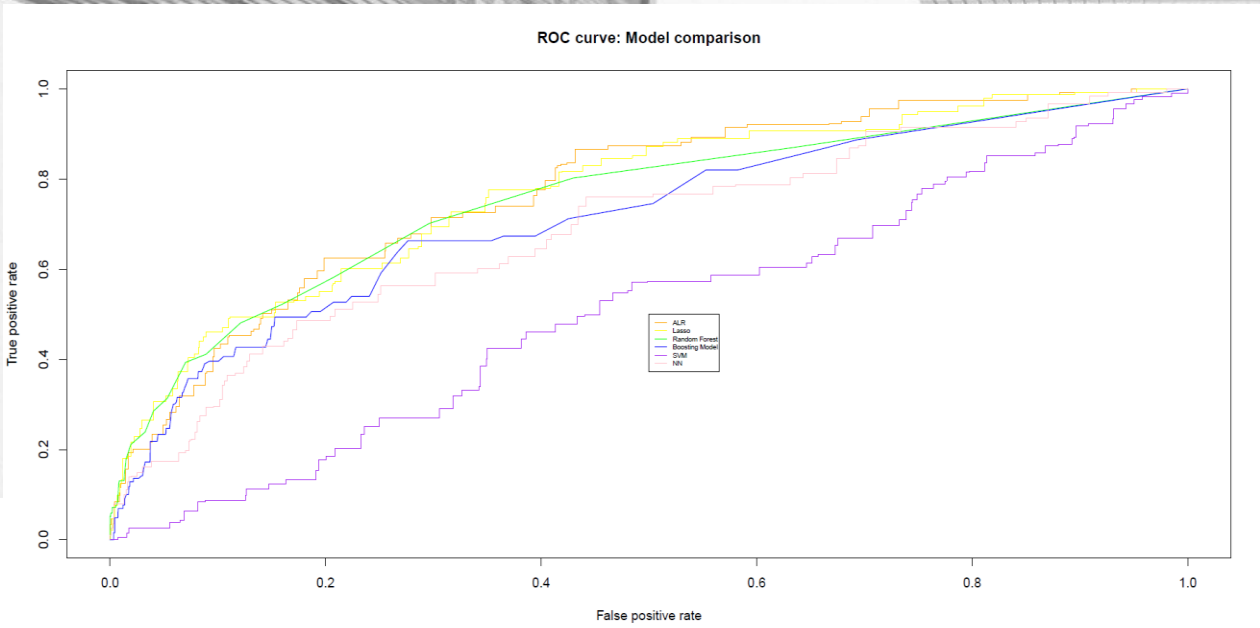


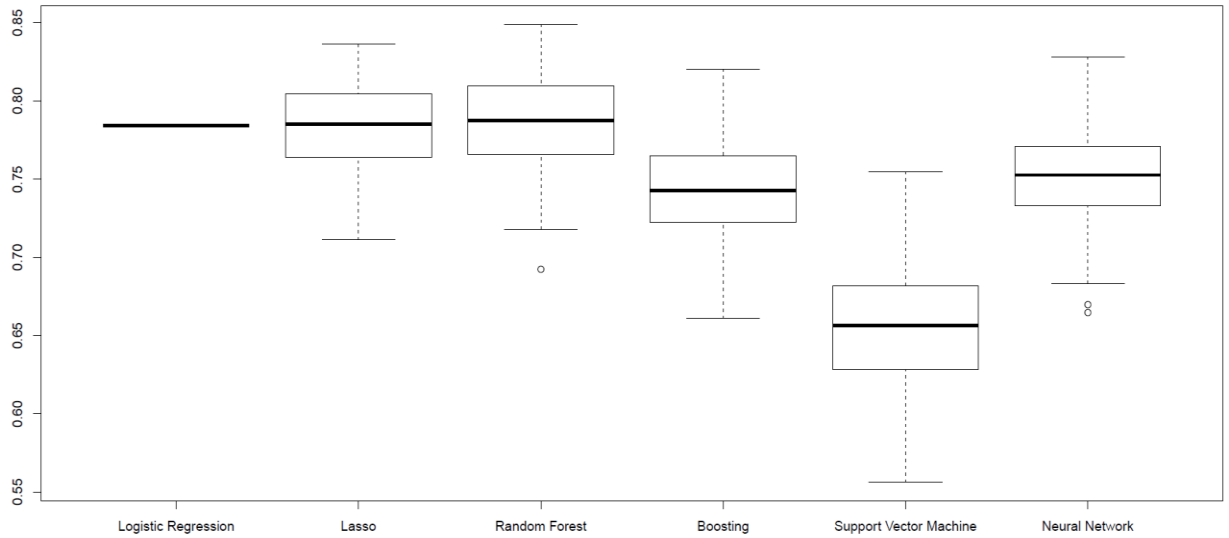


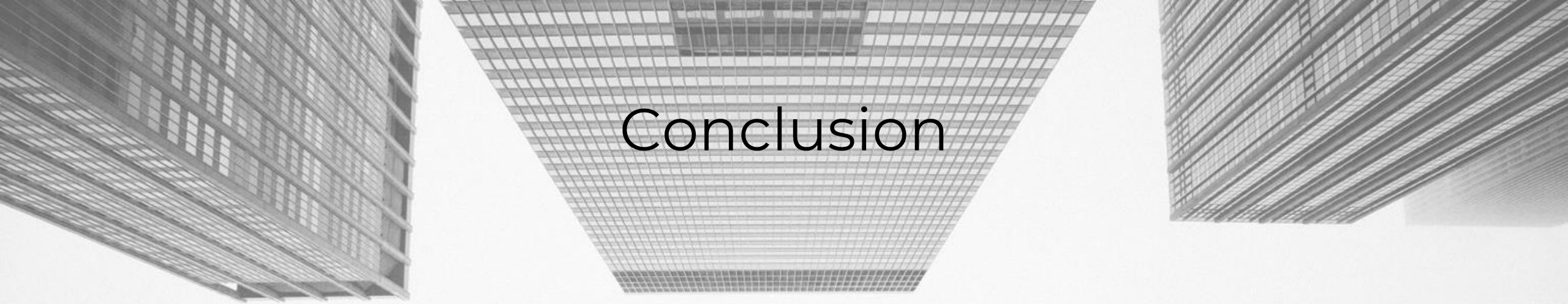
Neural Network (ROC)



Comparison


ROC curve: Model comparison
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0
ALR
Lasso
Random Forest
Boosting Model
SVM
NN


0.55
0.60
0.65
0.70
0.75
0.80
0.85
Logistic Regression
Lasso
Random Forest
Boosting
Support Vector Machine
Neural Network



Conclusion

- Only dynamic frameworks are viable
- Advance machine learning models make sense only if new data flows are captured
- Model risk management and governance is curently inadapted
- Computational learning theory is key
- It might be challenging for central banks to supervise these models

UCL

Societal biases reinforcement through Machine Learning – A credit scoring perspective

Bertrand K. Hassani

UCL Computer Science

Université Paris 1 Panthéon-Sorbonne

MAKING AN IMPACT THAT MATTERS since 1845

Summary

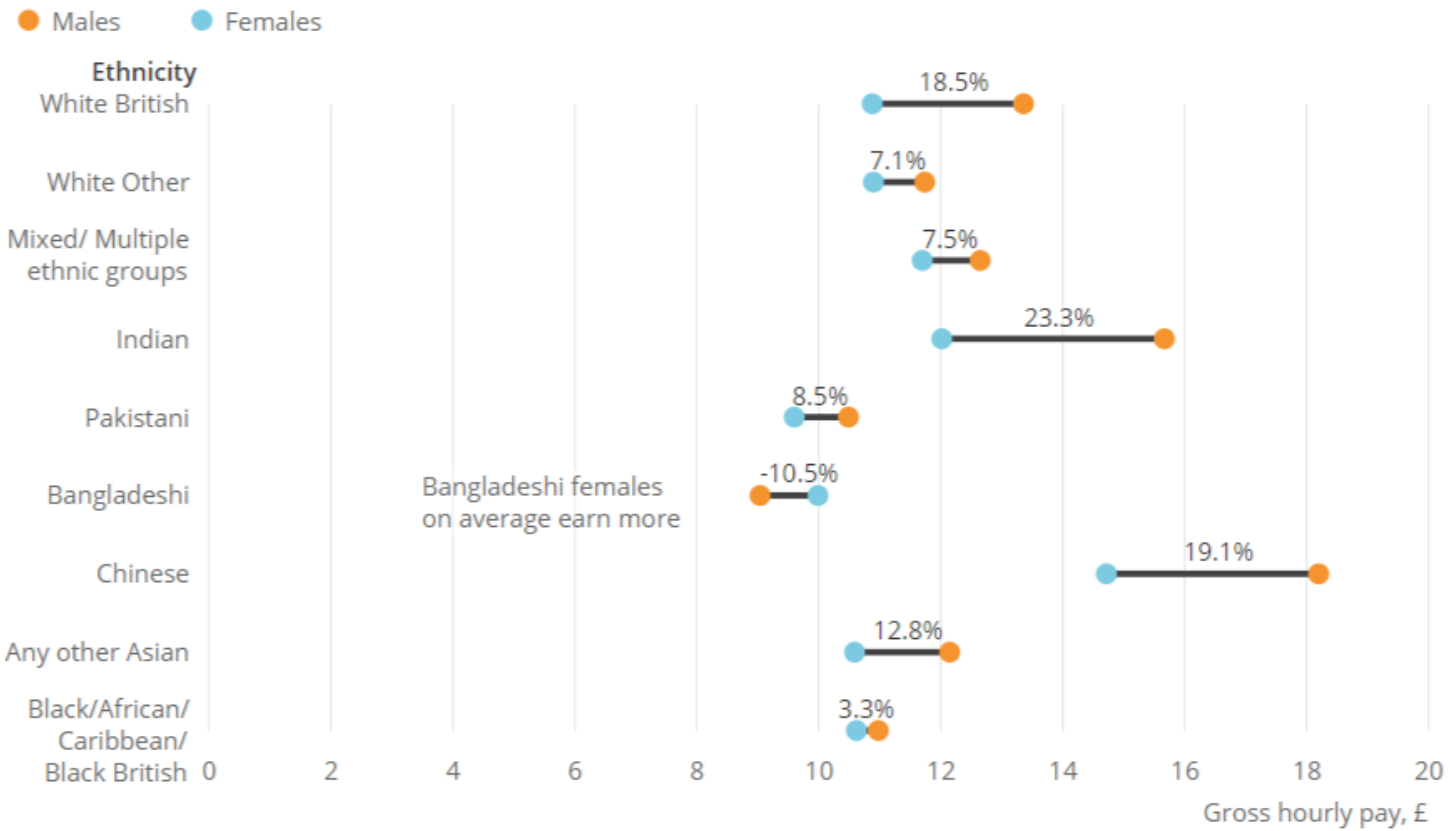
- Problematic: does machine learning and AI ensure that social biases thrive ?
- Presentation of Social and Societal Bias Evidence
- Presentation of the Data Sets
- Presentation of the Methodology
- Results

Societal Biases Evidence

Social and Societal Biases

Social Biases becomes societal biases when they become the norm.

Median gross hourly earnings for all employees by sex, Great Britain, 2018

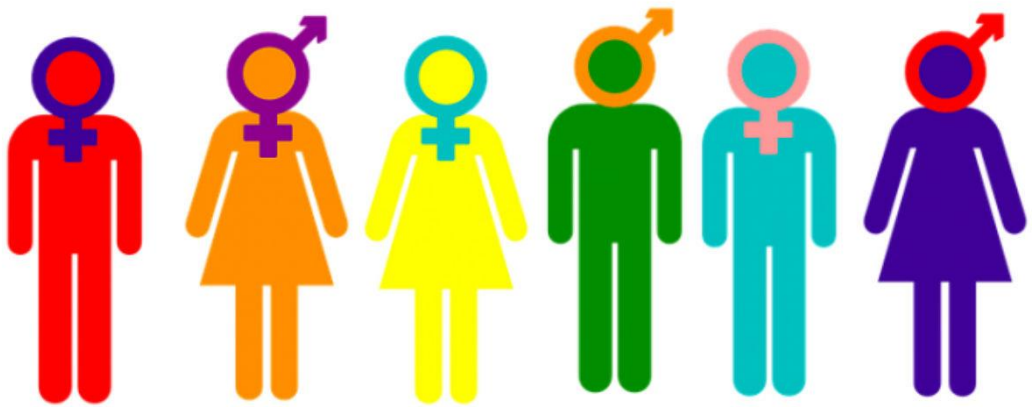


https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/earningsandworkinghours/articles/ethnicitypaygapsingreatbritain/2018

Data Sets

Data Sets: Gender

```
    Gender    Married     Dependents          Education
Female:113   No :209   Min.    :0.0000   Graduate     :467
Male  :484   Yes:388   1st Qu.:0.0000   Not Graduate:130
                       Median :0.0000
                       Mean   :0.7621
                       3rd Qu.:2.0000
                       Max.   :3.0000
  LoanAmount   Loan_Amount_Term Credit_History
Min.    :  9   Min.    : 12.0   Min.   :0.0000
1st Qu.:101   1st Qu.:360.0    1st Qu.:1.0000
Median :129   Median :360.0    Median :1.0000
Mean   :147   Mean   :342.3    Mean   :0.7755
3rd Qu.:166   3rd Qu.:360.0    3rd Qu.:1.0000
Max.   :700   Max.   :480.0    Max.   :1.0000
     Feat3            Feat4
Min.    :   70   Min.    :0.002595
1st Qu.: 1708   1st Qu.:0.057600
Median : 2718   Median :0.088823
Mean   : 3806   Mean   :0.102575
3rd Qu.: 4300   3rd Qu.:0.122909
Max.   :63337   Max.   :2.400000
```



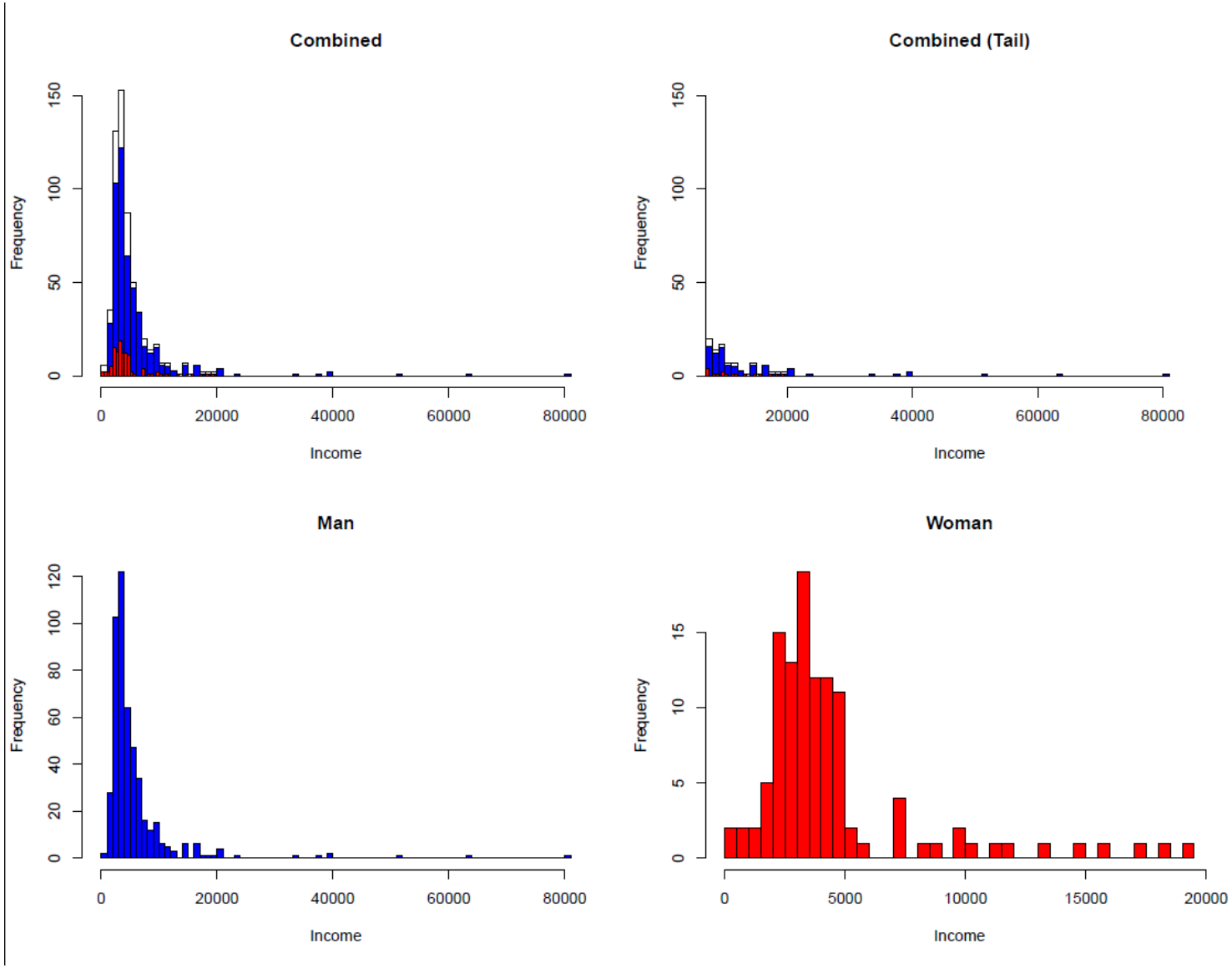
```
Self_Employed ApplicantIncome CoapplicantIncome
No :517       Min.    :  150   Min.    :    0
Yes: 80       1st Qu.: 2873   1st Qu.:    0
              Median : 3800   Median : 1229
              Mean   : 5418   Mean   : 1639
              3rd Qu.: 5818   3rd Qu.: 2306
              Max.   :81000   Max.   :41667
  Property_Area Loan_Status      Feat1              Feat2
Rural    :176   N:186        Min.    :    0.01   Min.   :0.003016
Semiurban:227   Y:411        1st Qu.:    1.34   1st Qu.:0.023500
Urban    :194                Median :    3.83   Median :0.031017
                             Mean   : 3196.91   Mean   :0.038635
                             3rd Qu.: 4547.00   3rd Qu.:0.043177
                             Max.   :81000.00   Max.   :0.900000
```

Data Sets: Ethnicity

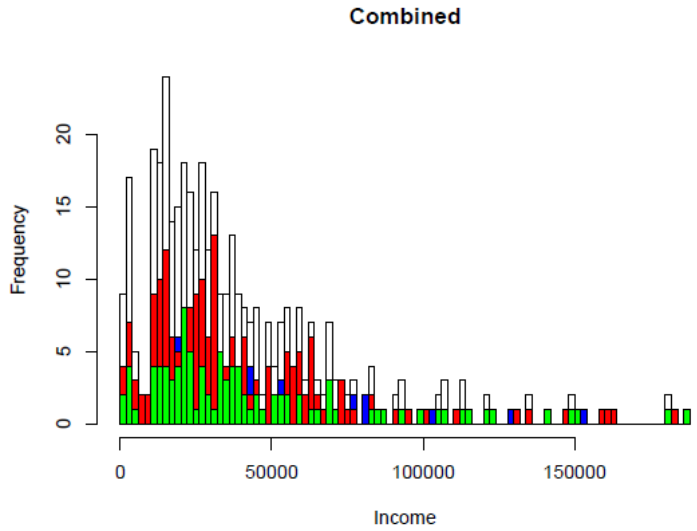
```
  ï..Income          Limit          Rating          Cards
Min.   :    12   Min.   :  855   Min.   : 93.0   Min.   :1.000
1st Qu.: 16481   1st Qu.: 3088   1st Qu.:247.2   1st Qu.:2.000
Median : 29715   Median : 4622   Median :344.0   Median :3.000
Mean   : 40672   Mean   : 4736   Mean   :354.9   Mean   :2.958
3rd Qu.: 53778   3rd Qu.: 5873   3rd Qu.:437.2   3rd Qu.:4.000
Max.   :186634   Max.   :13913   Max.   :982.0   Max.   :9.000
Married                Ethnicity      Balance          Ethnic
No :155    African American: 99   Min.   :   0.00   Caucasian:199
Yes:245    Asian           :102   1st Qu.:  68.75   Other    :201
           Caucasian       :199   Median : 459.50
                                  Mean   : 520.01
                                  3rd Qu.: 863.00
                                  Max.   :1999.00
```

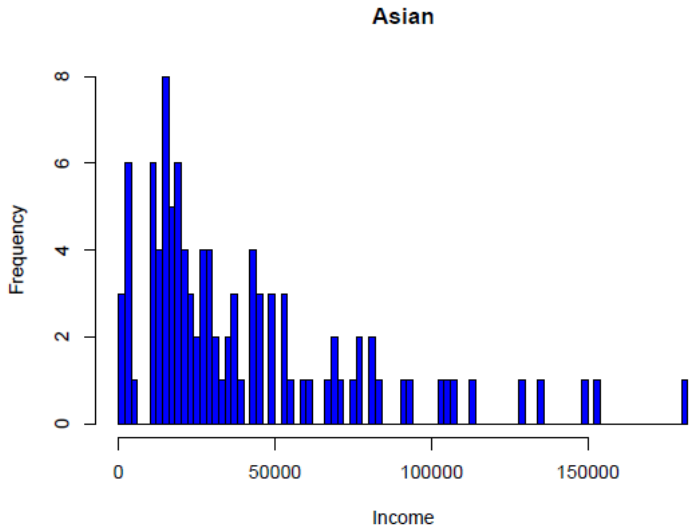
```
      Age          Education       Gender    Student
 Min.   :23.00   Min.   : 5.00   Female:207   No :360
 1st Qu.:41.75   1st Qu.:11.00   Male  :193   Yes: 40
 Median :56.00   Median :14.00
 Mean   :55.67   Mean   :13.45
 3rd Qu.:70.00   3rd Qu.:16.00
 Max.   :98.00   Max.   :20.00
     RatingN
  Min.   :0.0000
  1st Qu.:0.1735
  Median :0.2823
  Mean   :0.2946
  3rd Qu.:0.3872
  Max.   :1.0000
```

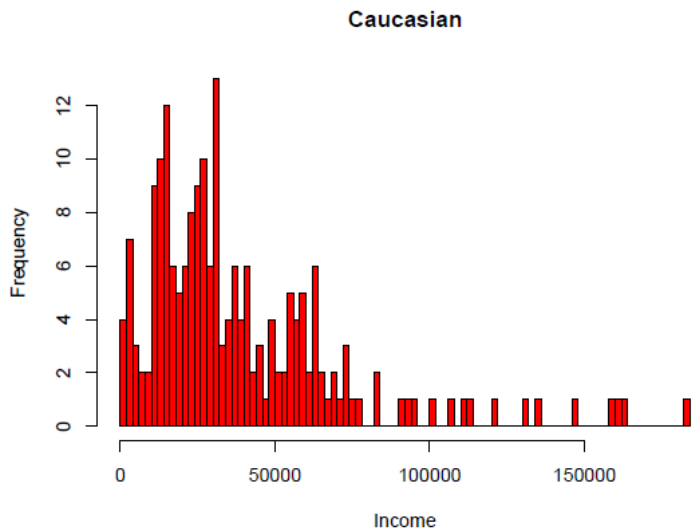
Data Sets: Gender

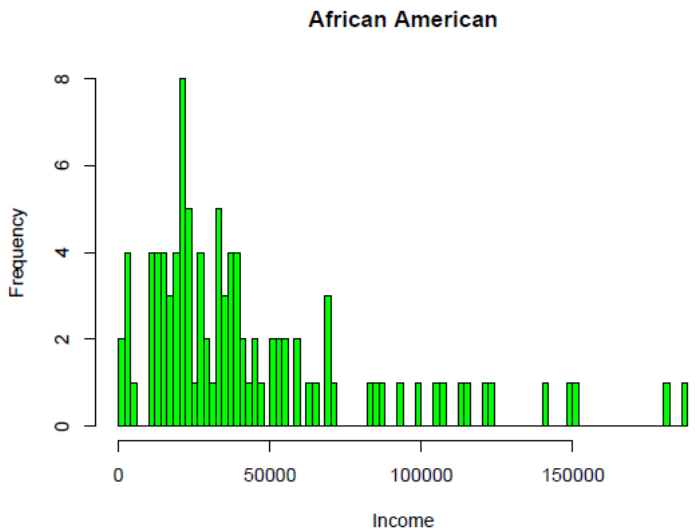


Data Sets: Ethnicity (Initial)

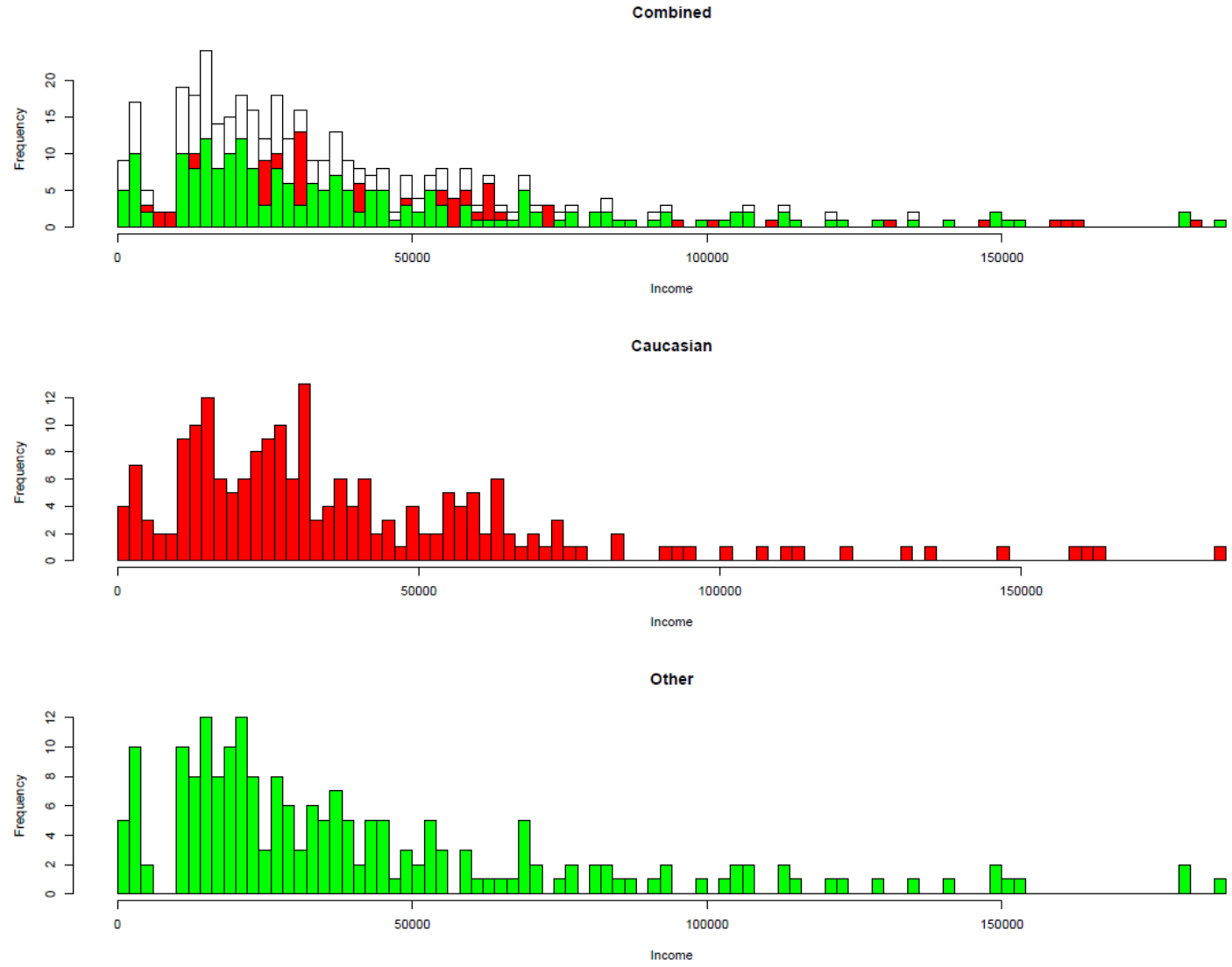




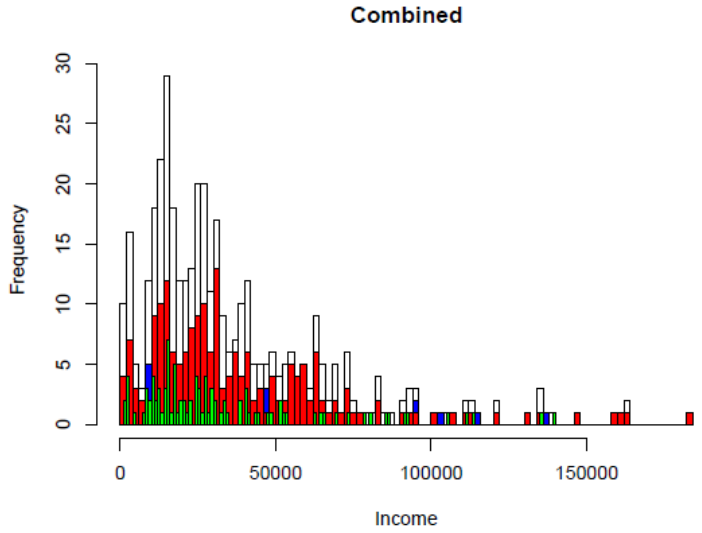


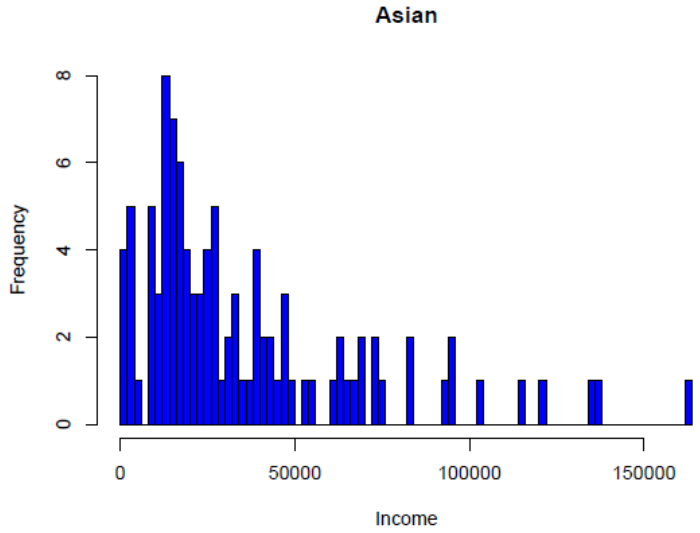


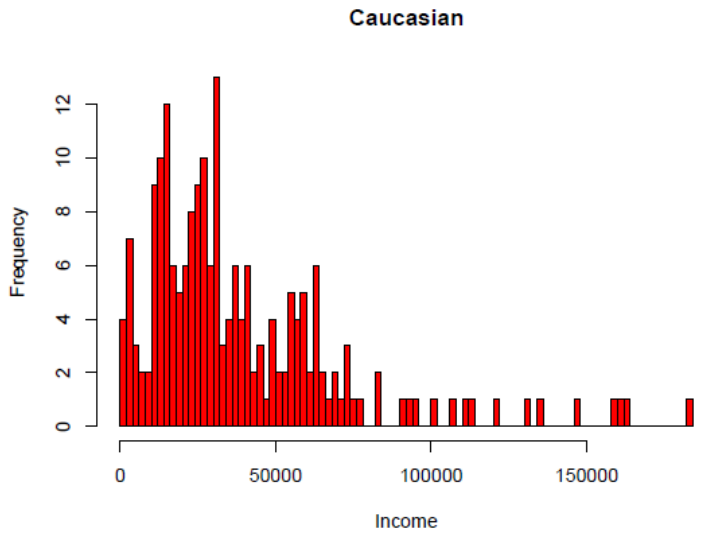
Data Sets: Ethnicity (2 Categories)

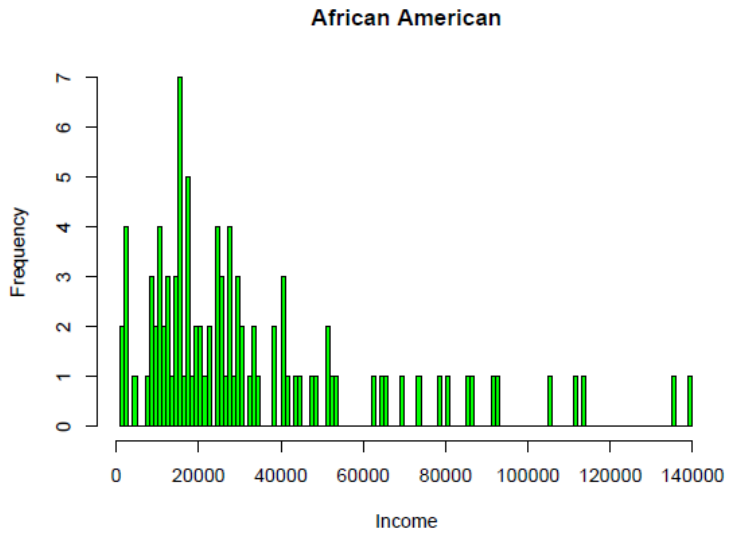


Data Sets: Ethnicity (Modified)









Methodology

Experimentation

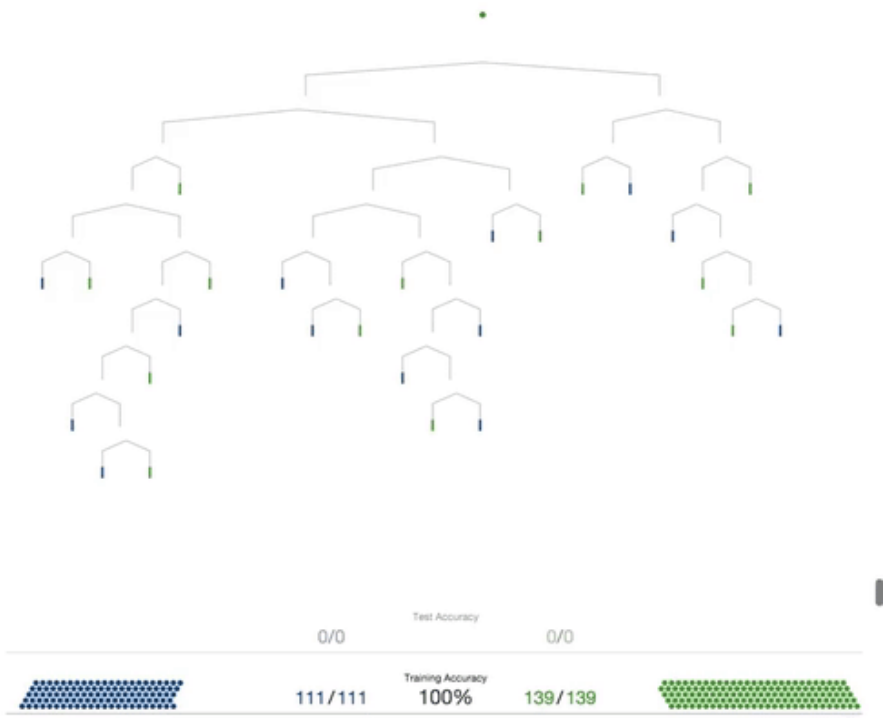
1. We need to ensure that the data set used is actually suitable for scoring purposes

2. Will these data set be useful to predict the gender or the ethnic group of the customer ?

3. So what ?

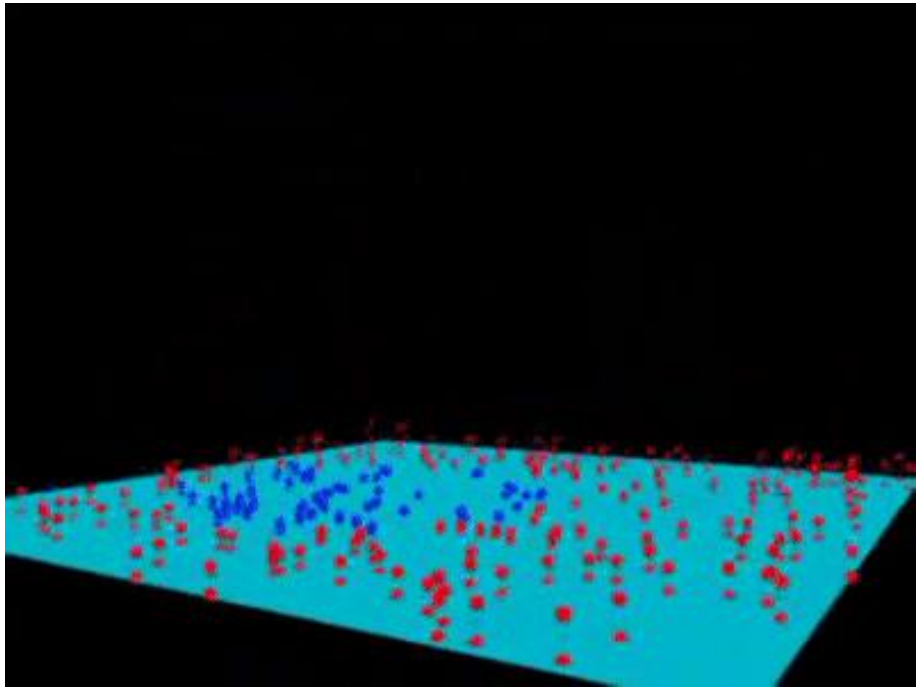
Random Forest and SVM

Random Forest



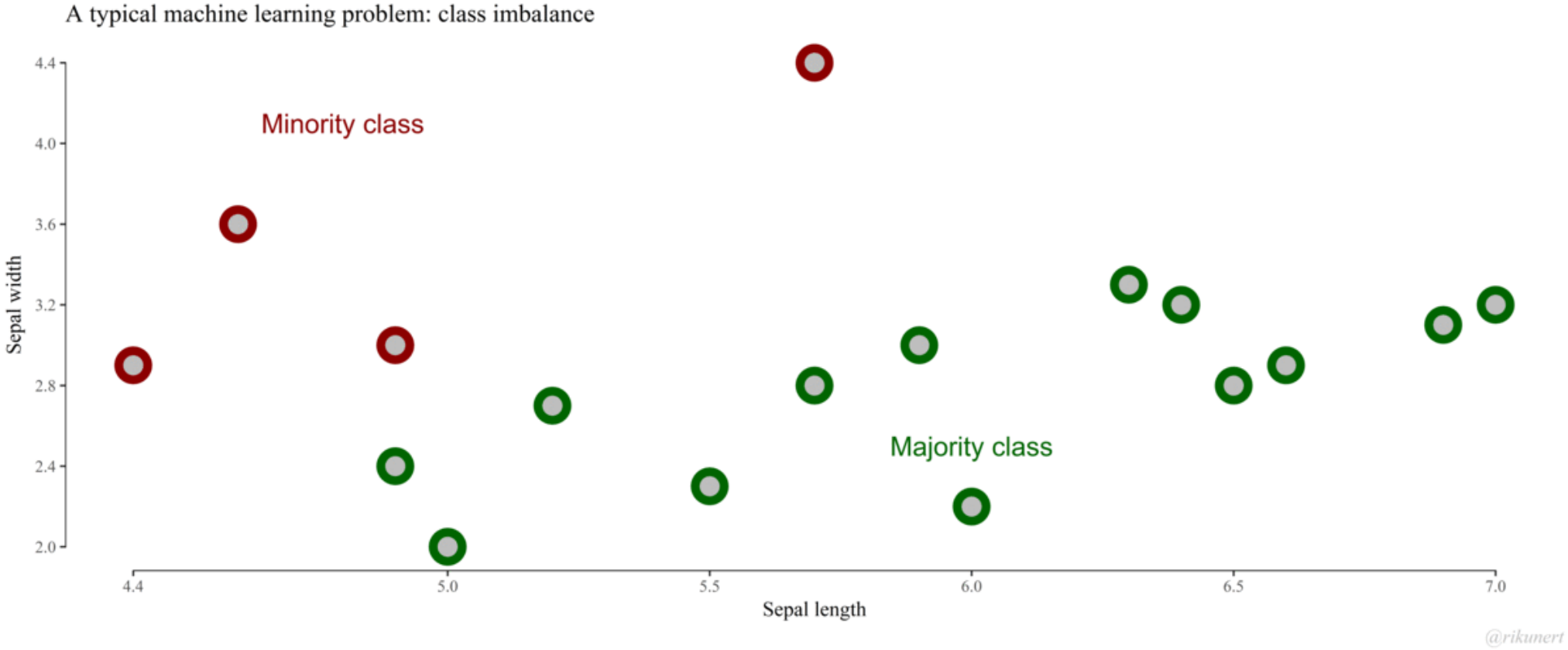
https://gfycat.com/fr/rigidfantasticblackfly

Support Vector Machine



https://towardsdatascience.com/a-friendly-introduction-to-support-vector-machines-svm-925b68c5a079

SMOTE: Synthetic Minority Over-sampling Technique



https://towardsdatascience.com/how-i-handled-imbalanced-text-data-ba9b757ab1d8

Results

Validity of the data sets from a credit scoring point of view

Ethnicity Set

Data as provided	0.0008210515
Data "2 categories"	0.0008903203

Table 4: This table presents the mean squared error obtained for the random forest regression performed using the "Ethnicity Set" for credit scoring purposes.

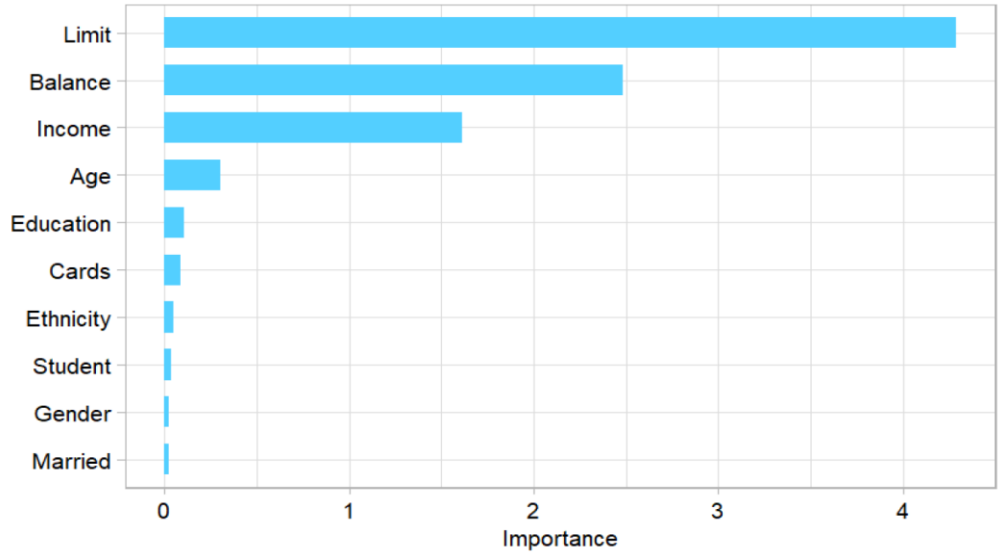
Gender Set

	Random Forest	Support Vector Machine
Data as provided	0.5052632	0.3030303
Features Engineered	0.5	0.2790698
Smote	0.8295189	0.7883529
Smote Features Engineered	0.843418	0.8054146

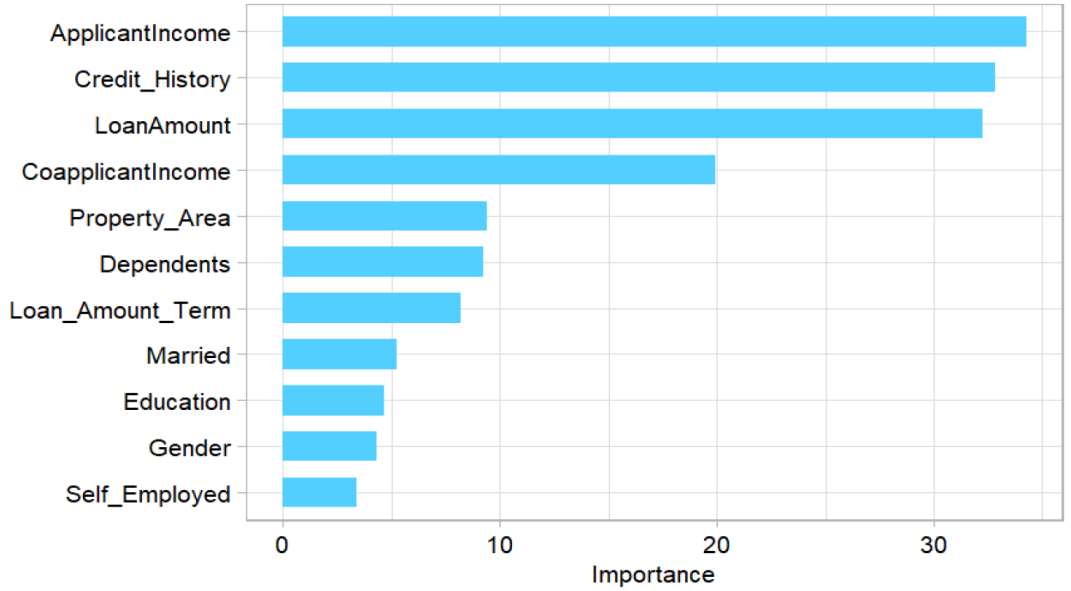
Table 6: This table presents the F1-Score obtained for the random forest classification performed using the "Gender Set" for Loan prediction purposes.

Variable Importance

Ethnicity Set



Gender Set



Validity of the data sets from a Ethnicity background and Gender prediction point of view

Ethnicity Set

Data as provided	0.6507937
Data "2 categories"	0.5154639
Data Modified	0.7
Data Modified Smote	0.9863014

Table 5: This table presents the F1-Score obtained for the random forest classification performed using the "Ethnicity Set" for ethnicity prediction purposes.

Gender Set

	Random Forest	Support Vector Machine
Data as provided	0.3333333	0.5
Features Engineered	0.3666667	0.4507042
Smote	0.8583765	0.7854478
Smote Features Engineered	0.8773748	0.7989691

Table 7: This table presents the F1-Score obtained for the random forest classification performed using the "Gender Set" for Gender prediction purposes.

Conclusions

Conclusion

- Problematic: does machine learning and AI ensure that social biases thrive ?
  - Yes, it does by replicating patterns it learns
- Does bank profit generating paradygm and prudential rules ensure that the issue cannot be tackled ?
  - Unfortunately, it seems that way
- So what ?
  - Further research, as it seems that if we work by homogeneous intermediate subsamples, we might have a solution to overcome the problem.

Bibliography

Bibliography

- [1] Ariane Hegewisch and Heidi Hartmann. The gender wage gap: 2018; earnings differences by race and ethnicity. Institute for Women's Policy Research, 7, 2018.

- [2] Ethnicity pay gaps in Great Britain: 2018. Office of National Statistics, United Kingdom, 2019.

- [3] Tamara E. Holmes. Credit card race, age, gender statistics. creditcards.com, 2019.

- [4] Peter Martey Addo, Dominique Guegan, and Bertrand Hassani. Credit risk analysis using machine and deep learning models. Risks, 6(2):38, 2018.

- [5] Leo Breiman. Random forests. Machine learning, 45(1):5{32}, 2001