Machine Learning Fundamentals:
Lecture 2 – Dimensionality Reduction

Myriam Tami

myriam.tami@centralesupelec.fr

Mention IA - 3A, 2025 – 2026

CentraleSupélec

Imagine you are playing pictionary

The game rules,

- You have to draw a random word
- Example of random word: "camel"
- You win if your friends guess well

Who volunteers to draw a camel?

Imagine you are playing pictionary

Which drawing would you make of these two?

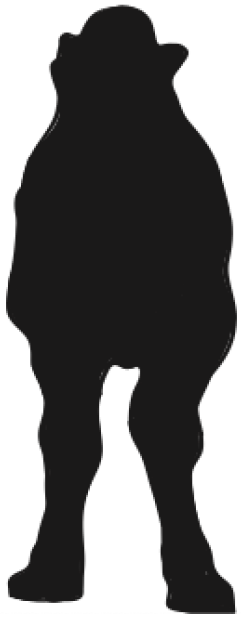
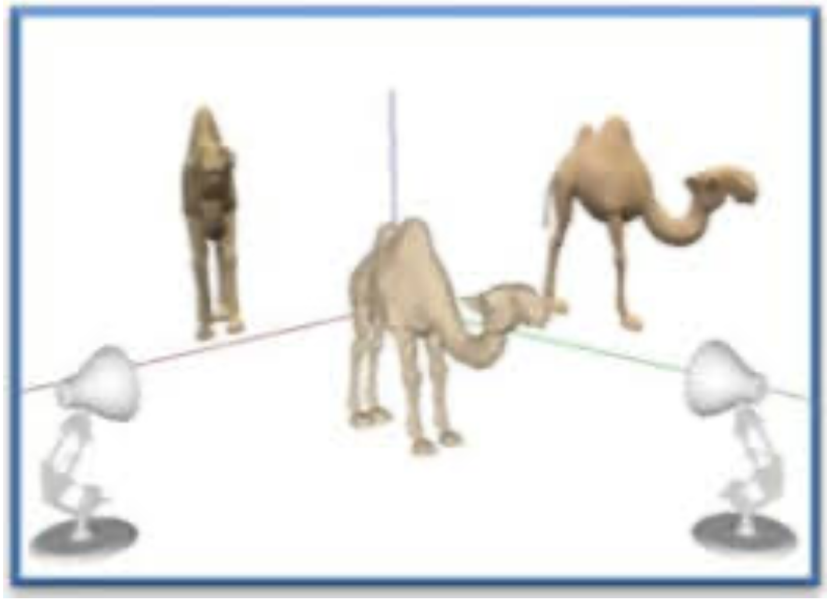


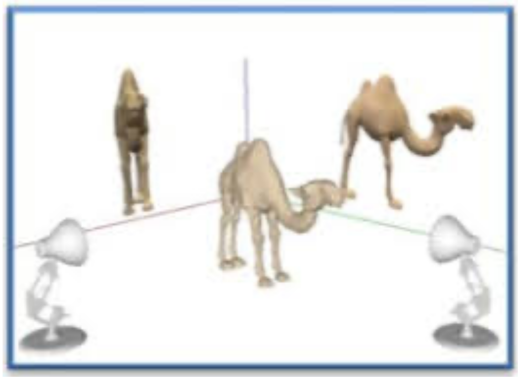
Figure: A

Figure: B



- By doing this, you have summarized a 3D object in a 2D (a plane)
- You projected down from 3 dimensions to 2 (a plane)
- You did a dimensionality reduction

Figure: A



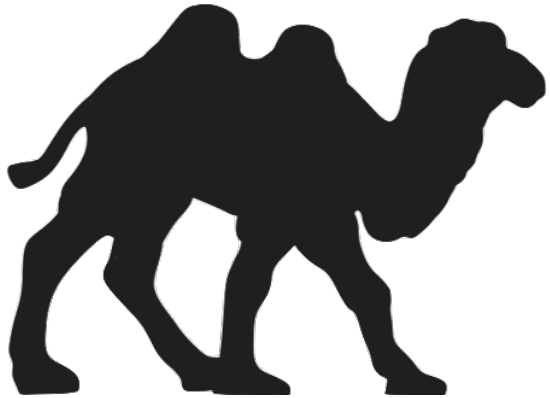


Figure: B

- Figure B summarizes better the camel 3D object than Figure A
- In general, we lose information about the data when we project down from a high dimenion space to a low one

Natural questions

- How choose the best lower dimension space of projection?
- What is the lower dimension space allowing to lose as less as possible information from the given data?
- How to keep useful information?
- How much of the information in the data set is lost by projecting the observations onto the lower dimension space?
- Why reduce dimension? What goes wrong in high dimensions?

1. Context and Motivations
   - High-Dimensional Data
   - What goes wrong in High Dimensions?
   - Illustration of the curse of dimensionality
   - High dimensions and model interpretability
2. Subset selection approaches
   - Best subset model selection procedure
   - Stepwise model selection procedure
   - How choose the best model?
3. Dimension reduction methods
   - Motivations
   - PCA
   - Illustration of the use of PCA on a dataset
   - Others methods
4. Conclusion

- Most traditional approaches for both regression and classification tasks are intended for **low-dimensional** data sets: $n >> d$
  i.e. the number of **observations/instances**, is much greater than the number of **predictors/features**
- Because throughout most of the field's history, the bulk of scientific problems have been low-dimensional
- In the past decades, new technologies allow to collect unlimited number of feature measurements
- If d can be very large, n is often limited due to the cost or sample availability, etc.

☞ By dimension we are referring to the size of d

Definition (High-dimensional data)

Data sets containing more features than observations ($d >> n$) are referred to **high-dimensional**

Example (High-dimensional data set in medicine field)

- task: predict blood pressure
- $n \approx 200$ patients
- features: age, gender, body mass index and measurements for half a million single nucleotide polymorphisms (individual DNA mutations that are relatively common in the population)
  i.e. $d \approx 500,000$

Classical approaches are not appropriate in this setting ($d >> n$), such as

- Least squares linear regression
- logistic regression
- linear discriminant analysis
- some classification algorithms
- ...

Definition (The curse of dimensionality)

The curse of dimensionality **introduces sparseness of the training data**. The more features we use, the more the volume of the space increase fastly and the more sparse the data becomes such that obtaining a good prediction accuracy (of the learned predictor or classifier) becomes difficult.

▸ Mathematical-References

The curse of dimensionality

- In order to obtain a good prediction accuracy (i.e. less sparsity to guarantee the result provided by the use of statistical learning approaches), the amount of data needed often grows exponentially with the dimensionality.
- Another effect of the curse of dimensionality, is that this sparseness is not uniformly distributed over the data space. In fact, data around the origin (at the center of the hypercube) is much more sparse than data in the corners of the space. ▸ References

☞ A serious consequence of the curse of dimensionality is overfitting

Example 1

Example (A simple linear classification)

- data: a set of images depicting a cat or a dog
- task: design a classifier that is able to distinguish dogs from cats
- approach (or algorithm or class of function) and model: linear classifier

```
if 0.5*red + 0.3*green + 0.2*blue > 0.6 then
    return: cat;
else
    return: dog;
end if
```

Algorithm 1: A linear classifier combining three features noted: red, green, blue

- **naive idea: add some features** based on color or texture of the image **to obtain a perfect classifier**

▸ Reference

Example 1

Can we obtain a perfect classification by carefully defining a few hundred of these features and adding them to the linear model?

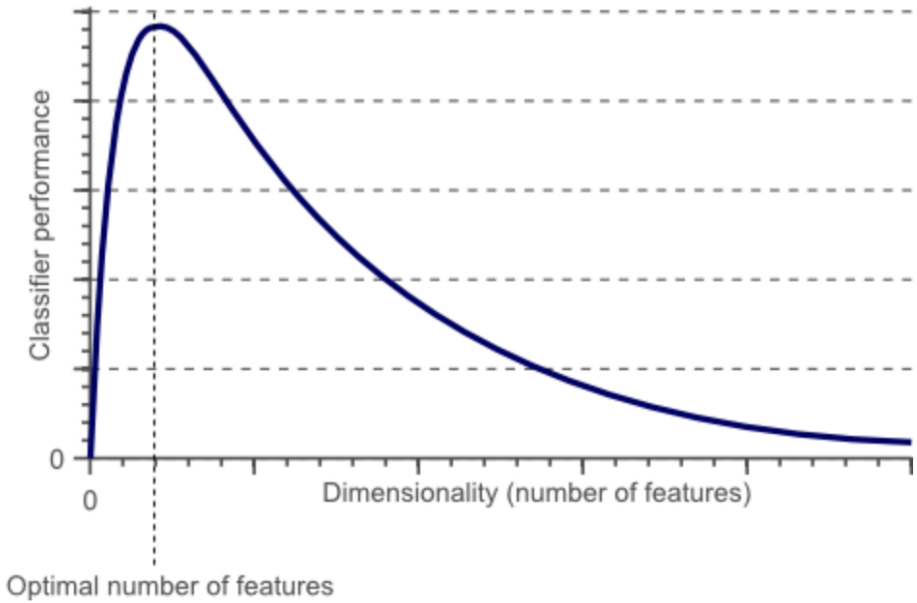


Figure: As the dimensionality increases, the classifier's performance increases until the optimal number of features is reached. **Further increasing the dimensionality without increasing the number of training samples results in a decrease in classifier performance**.

↪ Counter-intuitively: no we can not!

Example 1

- train data: 10 pictures of cats and dogs
- goal: train a classifier based on these training instances, that is able to correctly classifiy new pictures of cats and dogs (test data)
- Model: a linear classifier

Figure: One feature does not result in a perfect separation of our training data

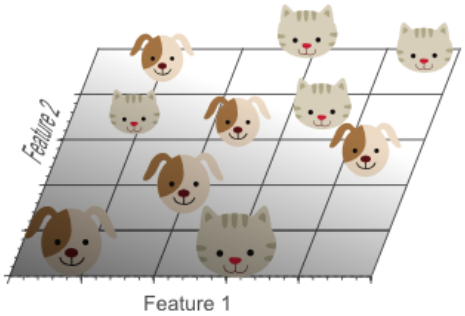


Figure: Adding a feature does not result in a linearly separable classification problem

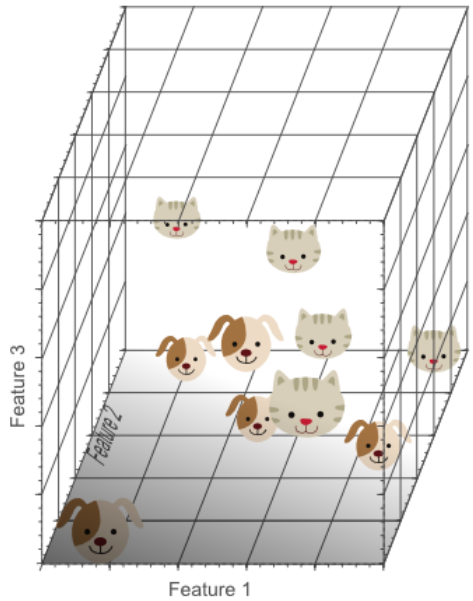


Figure: Adding a third feature results in a linearly separable classification problem

Example 1

$\Rightarrow$ A plane exists that perfectly separates dogs from cats:

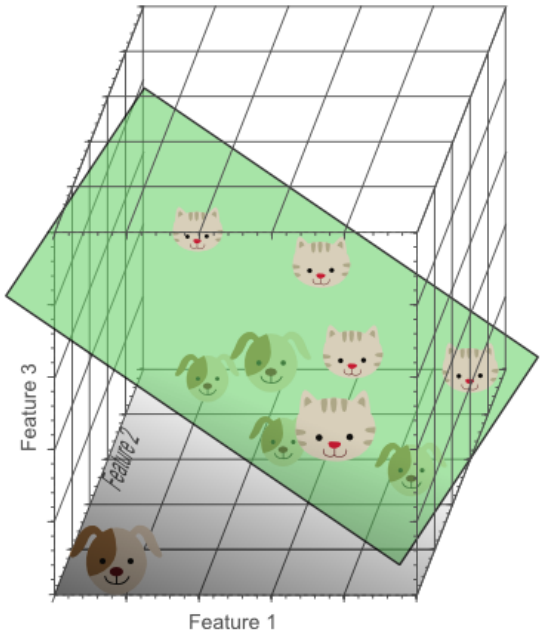


Figure: The more features we use, the higher the likelihood that we can successfully separate the classes perfectly.

- Note how the density of the training samples decreased exponentially when we increased the dimensionality of the problem

Example 1

- If we would keep adding features, the feature space becomes sparser and sparser
- It becomes more easy to find a separable hyperplane
- However, **if we project the highly dimensional classification result back to a lower dimensional space**, a huge problem associated with this approach appears: too complex model is now observed!

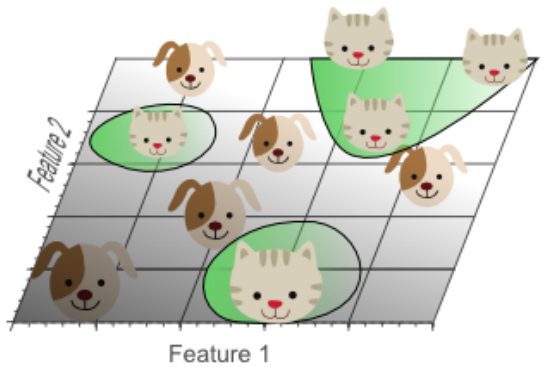


Figure: In a lower feature space, we observe using too many features results in **overfitting**.

Example 1

Definition (overfitting)

The predictor (classifier or regressor) starts **learning exceptions that are specific to the training data** and **do not generalize well** when new unseen data from the testing or validation set is encountered

- The previous figure illustrates by the projection of the 3D classification results onto a 2D feature space that although the data was linearly separable in the 3D space, this is not the case in a lower one
- Adding dimensions to obtain perfect classification results correspond to use a complicated non-linear classifier in the lower feature space
- As a result, the classifier is overfitting

Example 1

- The concept of overfitting is a direct result of the curse of dimensionality
- A linear classifier that has been trained using less features could give better results in terms of ability to generalization on new data

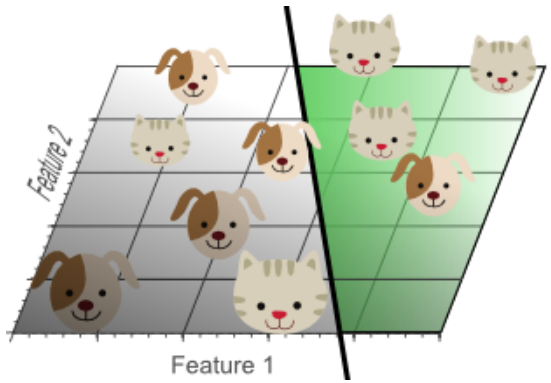


Figure: Although the training data is not classified perfectly, this classifier achieves better results on unseen data than the one from the previous figure

☞ By using less features, the curse of dimensionality was avoided such that the classifier did not overfit the training data

Example 2

Example (Least squares regression)

- data: a set of n observations and $d = 1$ feature
- task: design a regressor that is able to predict the value of a target variable Y (quantitative and continuous)
- approach and model: least squares regression, standard linear model

$$Y = \theta_0 + \theta_1 X_1 + \varepsilon \tag{1}$$

commonly used to describe a linear and additive relationship between a target Y and a set of features $X_1, X_2, \ldots, X_d$ (here only X_1)

- goal of this example: illustrate what can goes wrong when $d >> n$ by decreasing n value

Example 2

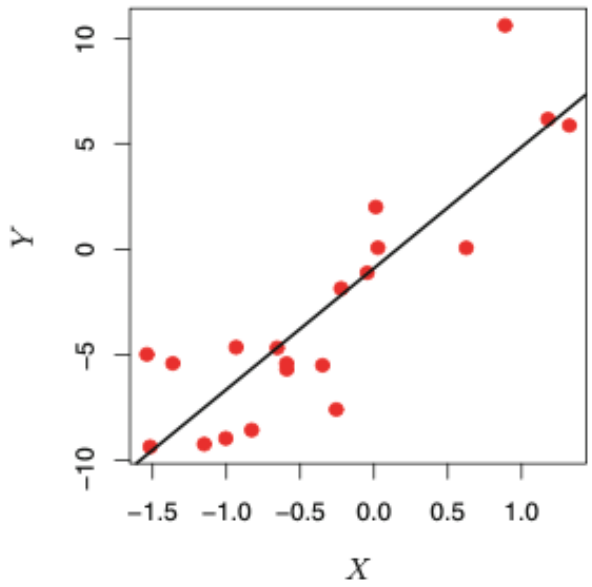


Figure: Case 1: Least squares regression when $n = 20 >> d = 1$. The linear model learned does not perfectly fit the data but the regression line seeks to approximate the observations as well as possible

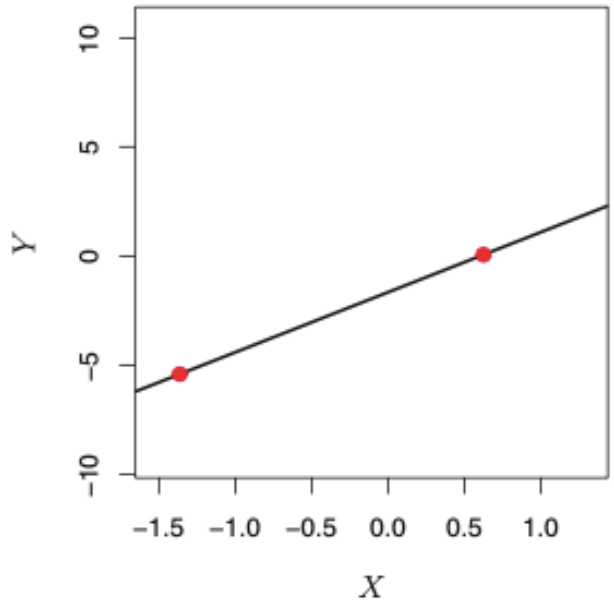


Figure: Case 2: Least squares regression when $n = 2 \approx d = 1$. The regression line fits the data exactly and leads to overfitting of the data.

Example 2

- Although it is possible to perfectly fit the training data (low training error) in the high-dimensional setting, the resulting linear model will perform extremely poorly on an independent test set (high testing error)
- Therefore this kind of learned regressor does not constitute a useful model

☞ It's important to apply extra care when analyzing data sets with a large number of features, and of always evaluating model performance on an independent test set

Example 2

It's easy to imagine a last case 3 corresponding to $d >> n = 1$

- there is no longer a unique least squares coefficient estimate
- As a result, the method cannot be used at all
- We need more constraints

- Some features from a given data set could be not associated with the target variable
- It's preferable to don't use these *irrelevant variable* in a multiple regression model (unnecessary complexity of the model)
- Advantage: by removing these variables, the model obtained is more easily interpreted

To tackle the problem associated to the curse of dimensionality and hight-dimensional data:

1. In this course, we see some approaches for automatically performing **feature selection** in the goal to exclude irrelevant variables and reduce the set of variables
2. Then we see some **dimension reduction methods** in the goal to reduce the dimension of an $n \times d$ data matrix into an $n \times M$ new one by computing M new features, such as $M << d$

☞ We will **keep the linear regression model as a common thread**.
But all concepts or methods presented are available for the classification models.

To perform **best subset selection**,

- We fit a separate least squares regression for each possible combination of the d features
  - We fit all d models containing exactly one feature
  - Then, fit $\binom{d}{2} = d(d-1)/2$ models containing 2 features
  - and so forth...
  - Don't forget the model containing no features and only the intercept

  $\hookrightarrow 2^d$ models
- We look at all of the resulting models, with the goal to identify the best one (in terms of accuracy)

Accuracy and measures for selection

Accuracy measures seen in the previous lecture:

- MSE, denoted $J()$ in the previous lecture:

$$J(\theta_0, \theta_1) = \frac{1}{2n} \sum_{i=1}^{n} \Big(h(x_i) - y_i \Big)^2 \tag{2}$$

  Question: What is h formula associated to the model (1)?
  $\hookrightarrow$ MSE could be computed for the train set (training error) or the test set (test error)

- Cross-validated prediction error
  Question: Who can recall how we compute a cross-validate prediction error for 5 folds?
  $\hookrightarrow$ associated to the test error

Accuracy and measures for selection

Accuracy measures probably seen during statistical lectures:

- $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{RSS}{TSS}$
  $\hookrightarrow$ an indication about the amount variance of Y explained by the model: **higher is R^2 better is the model**.
  $\hookrightarrow$ associated to training error
- Adjusted-$R^2 = 1 - \frac{RSS/(n-k-1)}{TSS/(n-1)}$
  $\hookrightarrow$ R^2 adjusted to the number of features $k \in \{1, \cdots, d\}$ into the model

☞ The training error tends to underestimate the test error: while **RSS always decreases as the number of features in the model increases**, $RSS/(n-k-1)$ may increase or decrease, due to the presence of k in the denominator.
It's better to use Adjusted-R^2 than R^2.

Reminder: Residual Sum of Squares

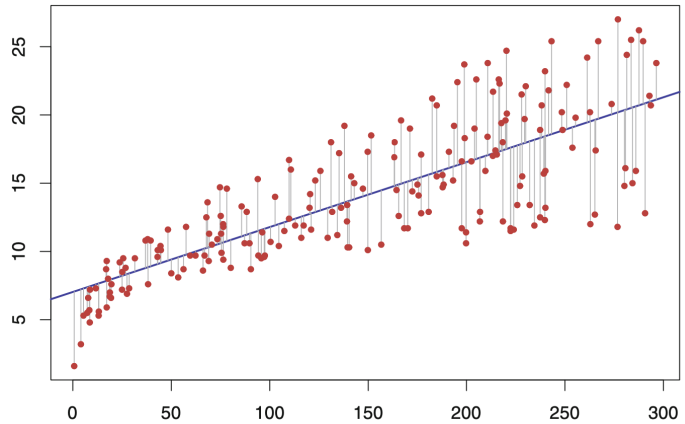


Figure: For these data (x-axis associated to a feature and y-axis to the target variable) the least squares fit for the regression of the feature onto the target is shown. The fit is found by minimizing the sum of squared errors. Each gray line segment represents an error, and the fit makes a compromise by averaging their squares (RSS is minimized). In this case a linear fit captures the essence of the relationship.

Reminder: Residual Sum of Squares

Example (For the model (1),)

$$RSS = \sum_{i=1}^{n} (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_{i1})^2 \tag{3}$$

Interpretation: RSS measures the amount of variability that is left unexplained after performing the regression

Accuracy and measures for selection (Brief listing)

Other measures:

- $C_p = \frac{1}{n}(RSS + 2k\hat{\sigma}^2)$
  called **Mallows' C_p** and k is the number of model's predictors and $\hat{\sigma}^2$ an estimate of the variance of the error ε associated with each target measurement in the model
- AIC= $\frac{1}{n\hat{\sigma}^2}(RSS + 2k\hat{\sigma}^2)$
  The **Akaike Information Criterion** criterion is defined for a large class of models fit by maximum likelihood
- BIC= $\frac{1}{n\hat{\sigma}^2}(RSS + log(n)k\hat{\sigma}^2)$
  Bayesian information Criterion

1: Let $\mathcal{M}_0$ denote the null model containing no features and simply predicting the sample mean for each instance.
2: **for** $k = 1, 2, \ldots, d$ **do**
3: Fit (**on the train set**) all $\binom{d}{k}$ models that contains exactly k features
4: Pick the best among these models, and call it $\mathcal{M}_k$. Here, the best means a smallest training error measurement[a] or a largest R^2.
5: **end for**
6: Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_d$ **using cross-validated prediction error** (**or prediction error on the test set**), C_p, AIC, BIC or Adjusted-R^2

Algorithm 2: Best subset selection algorithm

[a] for example RSS (Residual Sum of Squares) for least squares, deviance for logistic regression,...

- The row 2 of the Algorithm 2 allows to reduce the problem from one of 2^d possible models to one of $d + 1$ (cf. row 6)
- If using R^2 or RSS for choosing the best model among $\mathcal{M}_0, \ldots, \mathcal{M}_d$, be careful because of their different dimensions
  $\hookrightarrow$ The R^2 increases (low training error) as the number of features of the model
- "A low training error by no means guarantees a low test error". It's better to use the test error and select the model with the low one

- We can directly estimate the test error, using either a validation set approach or a cross-validation approach
- Or, we can indirectly estimate test error by making an adjustment to the training error by using C_p, AIC, BIC, Adjusted R^2 statistics

Advantages of Algorithm 2:

- Simple
- Conceptually appealing

Disadvantages:

- Computation time limitation
- The number of possible models grows quickly as d increases
  $\hookrightarrow$ If $d = 10$, there are $1,000$ possible models
- Best subset selection technique becomes computationally infeasible for values of d greater than 40, even with extremely fast modern computers

We need computationally efficient alternative methods

Stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

We have three stepwise algorithms:

- Forward Stepwise Selection
- Backward Stepwise Selection
- Hybrid Approaches

Forward stepwise selection

1: Let $\mathcal{M}_0$ denote the null model containing no features and simply predicting the sample mean for each instance.
2: **for** $k = 0, 1, \ldots, d-1$ **do**
3: Consider all $d-k$ models that augment the features in $\mathcal{M}_k$ with one additional feature.
4: Pick the best among these $d-k$ models, and call it $\mathcal{M}_{k+1}$. Here, the best means a smallest RSS (training error) or a largest R^2 (on the train set too).
5: **end for**
6: Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_d$ using cross-validated prediction error, C_p, AIC, BIC or adjusted R^2

Algorithm 3: Forward stepwise selection algorithm

Backward stepwise selection

1: Let $\mathcal{M}_d$ denote the full model containing all d features.
2: **for** $k = 1, \ldots, d-1, d$ **do**
3: Consider all k model(s) that contain all but one of the features in $\mathcal{M}_k$, for a total of $k-1$ features
4: Pick the best among these k models, and call it $\mathcal{M}_{k-1}$. Here, the best means a smallest RSS (training error) or a largest R^2 (on the train set too).
5: **end for**
6: Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_d$ using cross-validated prediction error, C_p, AIC, BIC or adjusted R^2

Algorithm 4: Backward stepwise selection algorithm

Hybride approaches

- The previous algorithms generally give similar but not identical models
- Hybrid versions of forward and backward stepwise selection are available:
- **Features are added to the model sequentially but after adding each new variable, the method may also remove any variables that no longer provide an improvement in the model fit**
- Such an approach attempts to more closely mimic best subset selection while retaining the computational advantages of forward and backward stepwise selection.

Both subset and stepwise model selection:

- Result in the creation of a set of models containing a subset of d features
- Need a way to determine the best one

But,

- the model containing all of the features will always have the smallest RSS and the largest R^2 (which are related to training error)

And,

- the training error can be a poor estimate of the test error

Therefore,

- For selecting the best model among a collection of models with different numbers of features, we need a way to estimate the test error

There are two common approaches:

1. We can indirectly estimate test error by making an adjustment to the training error: Adjusted R^2, C_p, AIC, BIC
2. We can directly estimate the test error using: a validation set approach, a cross-validation approach

Measures making an adjustment to the training error

Concerning Adjusted R^2,

- Adjusted R^2 allows an adjustment by dividing RSS with $n - k - 1$
- Thus, the following behavior "RSS always decrease as the number of features in the model increases" is penalized

For, C_p, AIC and BIC another kind of penalization:

- A penalty is added to the RSS error
- The penalty is proportional to the number of features in the model

Example (Mallows' C_p penalty)

$$C_p = \frac{1}{n}(RSS + \underbrace{2k\hat{\sigma}^2}_{penalty})$$

Validation set approach

- **Universal method**: implementation for most of the estimation procedures

Validation set Principle

1. Separate data into training data and test data
2. Build the estimator on the training sample
3. Use the test sample to compute the error of prediction (model assessment)

- It's possible to repeat several times and average the errors of prediction obtained

Cross validation approach

K-Fold Cross Validation Principle

1. Partition of data into K subsets
2. Each subset serves successively as a test sample, the rest as a training sample

In practice: K between 5 and 10

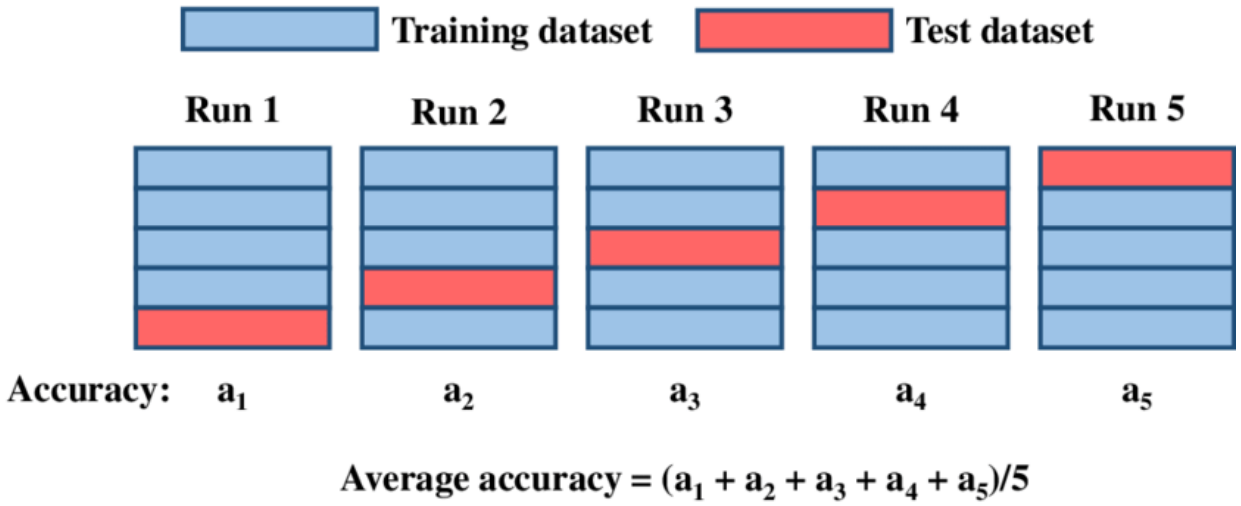


Figure: Diagram of a 5-fold Cross Validation ▸ Reference

Cross Validation (CV) approach and model selection

K-fold CV is a technique widely used for tuning parameters (a kind of model selection)

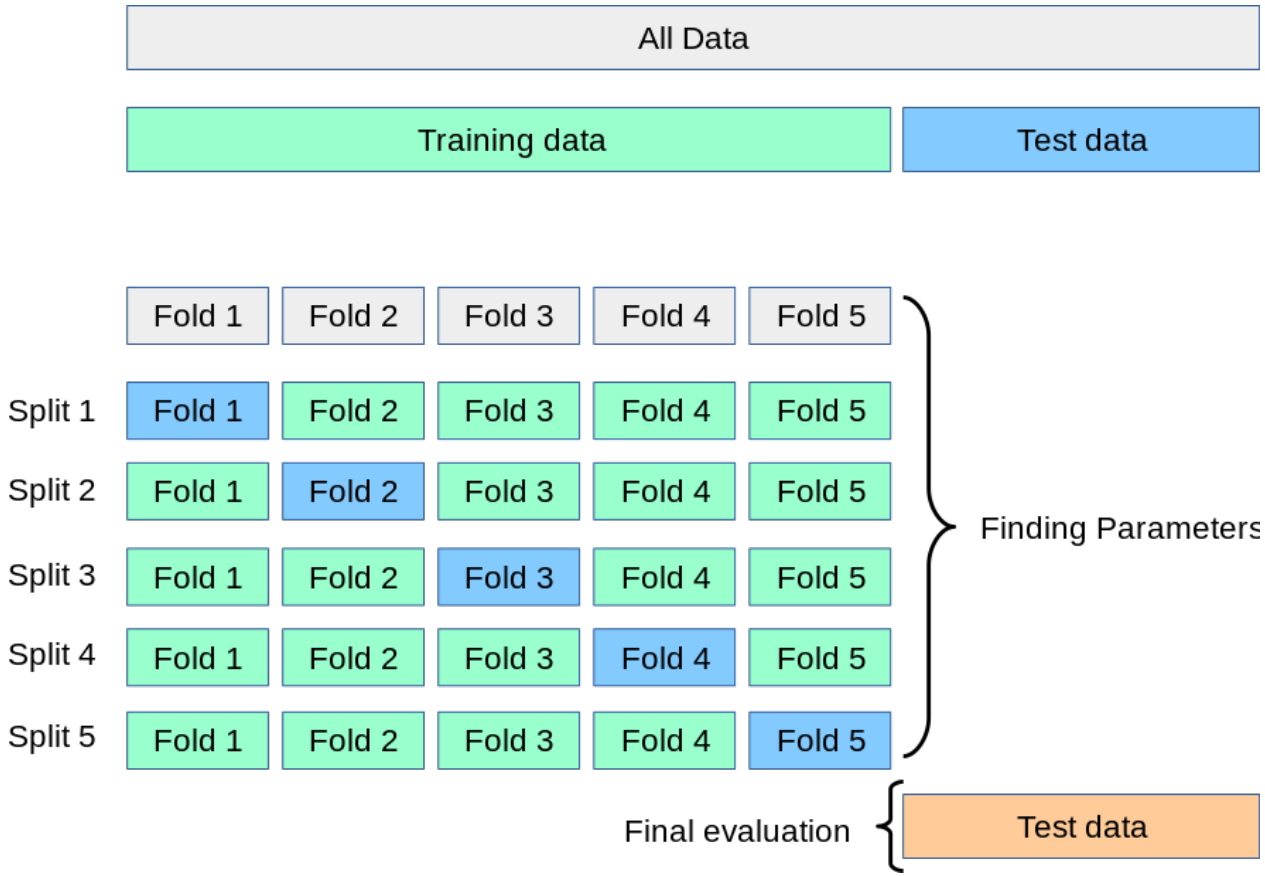


Figure: Diagram of a 5-fold CV evaluating estimator perf. and allowing model selection (hyperparameter=nb features) or tuning another parameters ▸ Reference

Example

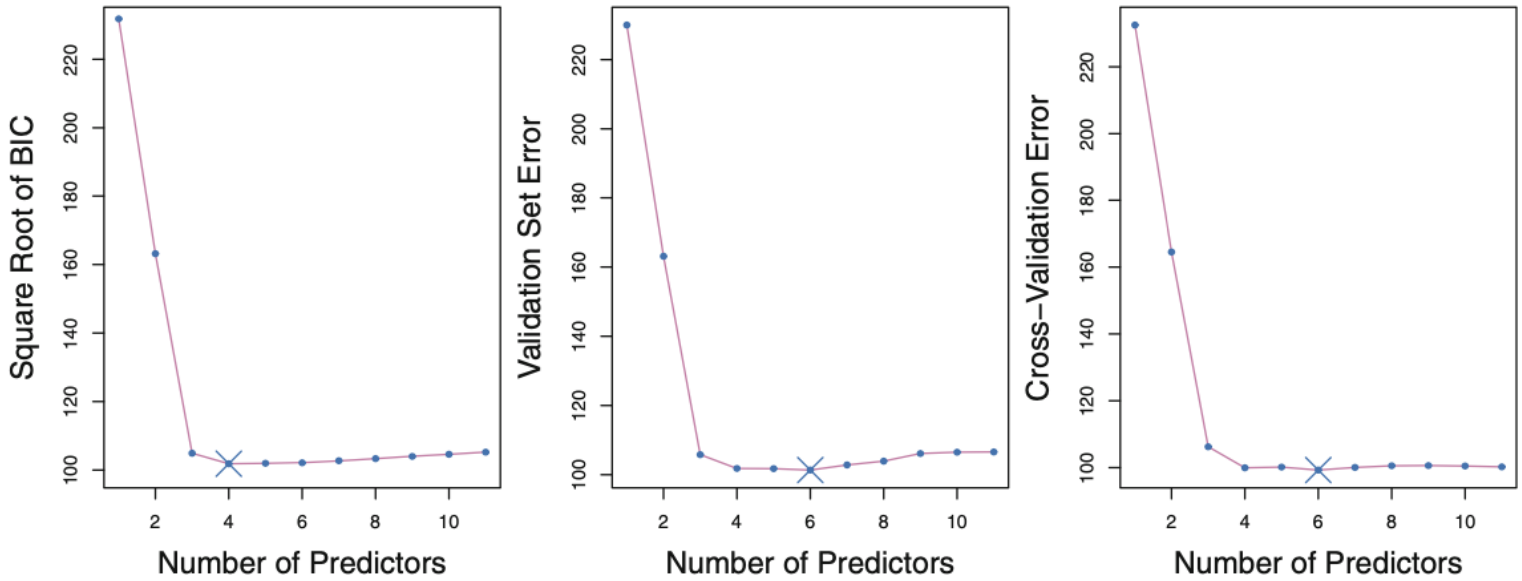


Figure: As a function of d the features called here "Predictors" ($d \in [1, \ldots, 11]$), **the BIC, validation set error** (3/4 of the data as the training set) **and 10-fold cross-validation errors on a data set are displayed for the best k-feature model**. The best model, based on each of these errors, is shown as a blue cross.

For much details about this example, the associated data set you can refer to ▸ this book

Example

We can select a model using the **one-standard-error rule**:

The one-standard-error rule

Select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve.
If a set of models appear to be more or less equally good, then we might as well choose the simplest model i.e: the model with the smallest number of features.

In the **previous example**, applying the one-standard-error rule to the validation set or cross-validation approach **leads to selection of the four-feature model**.

subset selection summary

- **Subset selection** approaches involve to **identify a subset of the *d* features** that we believe to be linked to the target variable
- We can then fit the model (e.g. a linear model) on the reduced set of features
- To select this subset of features we can use **best subset selection** method but for computational reasons it's better to consider **stepwise selection** techniques
- The selection of this subset of features is a kind of **model selection** or **hyperparameter tuning** allowing to **reduce the dimension**
- To select the best model, we can **compare their predictive error on a test set**
- Thus, we can directly use validation set approach or K-fold cross validation method which are much better than a statistical estimation of the test error (allowed by Mallows' C_p, AIC, BIC statistics)

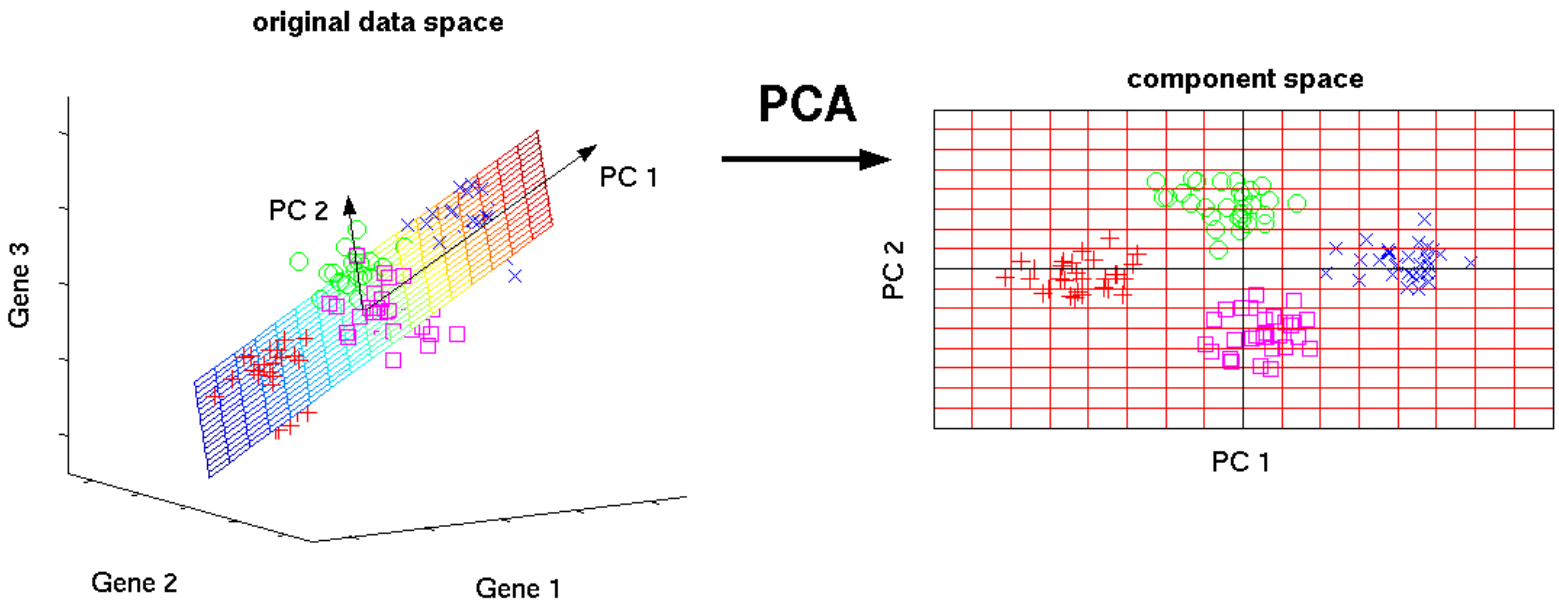
If you are more curious ▸ Reference

An alternative of subset selection methods are shrinkage methods (Ridge regression, Lasso)

- Shrinkage methods fit a model containing all d features using a technique that constrains or regularizes the coefficient estimates
- In other words: shrinkage methods shrinks some coefficients of the model towards zero

1. Context and Motivations
   - High-Dimensional Data
   - What goes wrong in High Dimensions?
   - Illustration of the curse of dimensionality
   - High dimensions and model interpretability
2. Subset selection approaches
   - Best subset model selection procedure
   - Stepwise model selection procedure
   - How choose the best model?
3. Dimension reduction methods
   - Motivations
   - PCA
   - Illustration of the use of PCA on a dataset
   - Others methods
4. Conclusion

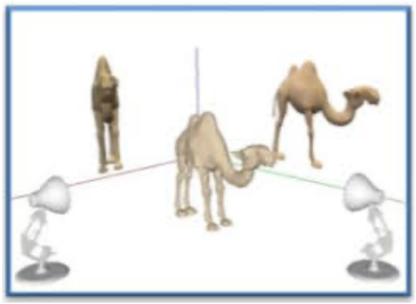
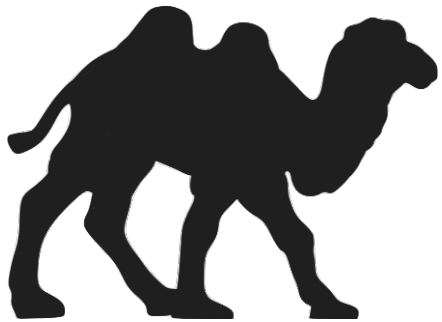
- All the previous methods reduce the number of features and use the original features $X_1, \ldots, X_d$
- We now present a **new class of methods allowing dimension reduction** that **transform the features** and then **fit a model using these transformed variables**
- These approaches are referred as **dimension reduction methods**



▸ Reference

Dimension reduction methods idea

- Certain views of high-dimensional data are more informative than others

- Can you find a low-dimensional representation with as much variation as possible (which minimizes the information distortion)?

Main question: Can you find a low-dimensional representation **with as much variation as possible**?

To implement the idea:

- What will be the **candidate** family of **low-dimensional representations**?
- How will we **choose** one of the many candidates?

Candidates: Linear combinations

Main question: Can you find a **low-dimensional representation** with as much variation as possible?
For the **candidate representation** aspect,

- Consider $Z_1, \ldots, Z_m, \ldots, Z_M$ represent $M < d$ **linear combinations** of our original d features (high-dimensional vectors),

$$Z_m = \sum_{j=1}^{d} \phi_{jm} X_j \tag{4}$$

or,

$$z_{i,m} = \sum_{j=1}^{d} \phi_{jm} x_{i,j} \tag{5}$$

- where, $\phi_m^T = (\phi_{1m}, \ldots, \phi_{dm})$ is a free parameter

Selection criteria: Maximal variance

Main question: Can you find a low-dimensional representation with **as much variation as possible**?

For the **"as much variation as possible"** aspect,

- For each $m \in [1, M]$, we want $z_{i,m}$ that has a largest sample variance
- Subject to the **normalization constraint** that for each m, $\sum_{j=1}^{d} \phi_{j,m}^2 = 1$

  $\hookrightarrow$ It is necessary to consider only linear combinations subject to this constraint, since otherwise we could increase $\phi_{1,m}, \ldots, \phi_{d,m}$ arbitrarily in order to increase the variance

Example

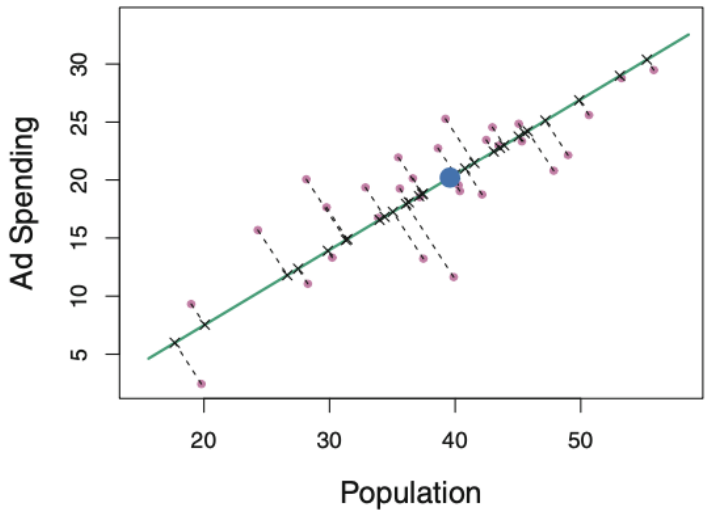


Figure: The *Population Size* (X_1) and *Ad Spending* (X_2) for $n = 30$ different cities are shown as purple circles (x_i). **The green solid line indicates the first principal component Z_1. The vector ϕ_1 gives the Z_1 direction.** The distances from each instance to Z_1 are represented using the black dashed line segments. The blue dot represents the mean $(\bar{X}_1, \bar{X}_2)$.

- The purple circles correspond to the instances x_i from the matrix X
- **The projections** of the purple circles on the green line correspond to $z_{11}, \ldots, z_{n1}$ called the **principal component scores**

Example

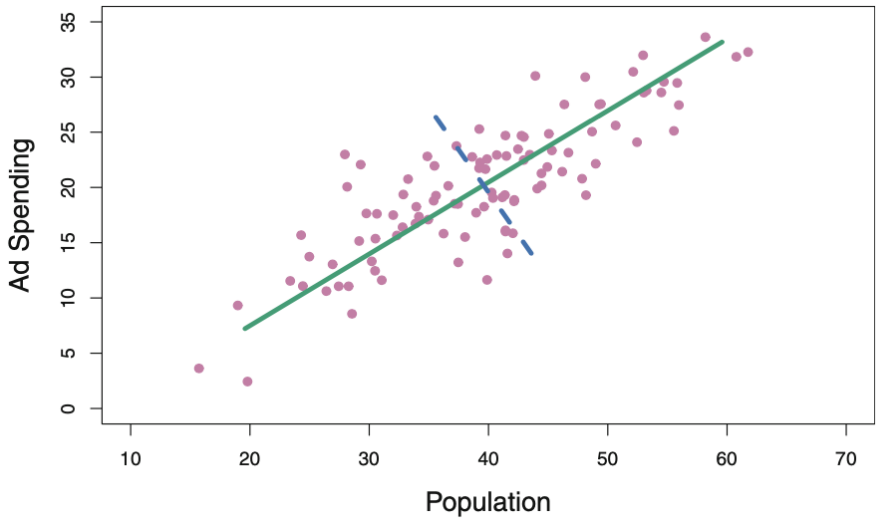


Figure: The same example with more instances $n = 100$. Z_1, shown in green, is the dimension along which the data vary the most, and it also defines the line that is closest to all n the observations (x_i). The blue dashed line indicates the second principal component Z_2.

- **Z_2 is a linear combination of the variables that is uncorrelated with Z_1, and has largest variance subject to this constraint**
- This zero correlation condition is equivalent to the condition that the direction of Z_2 must be orthogonal, to the Z_1 direction

Example

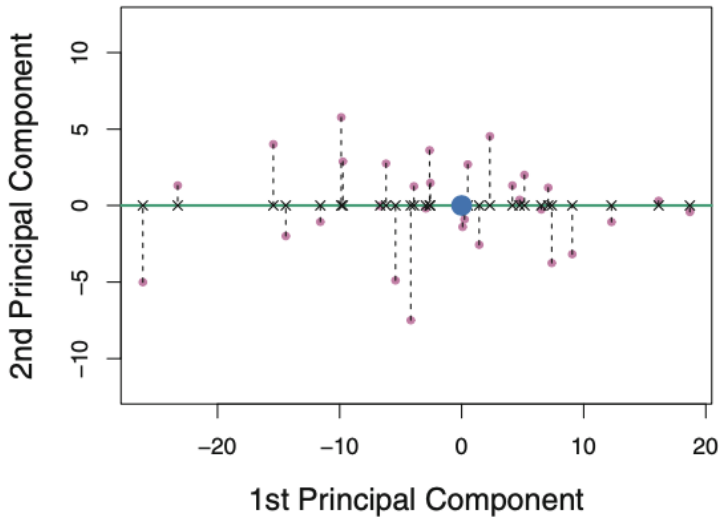


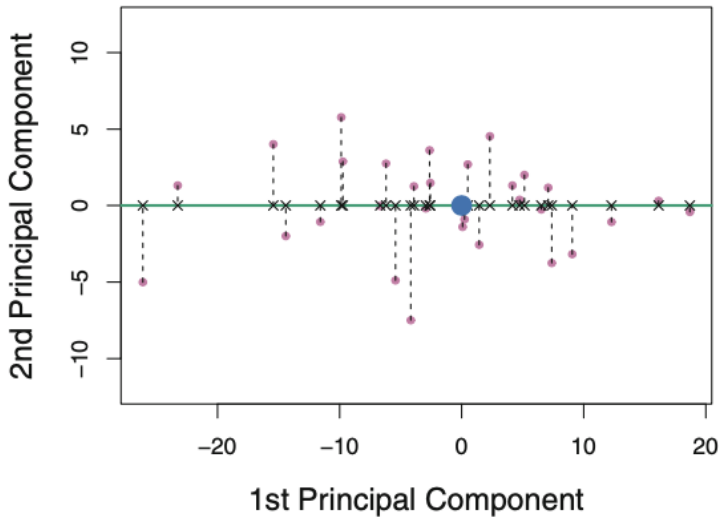
Figure: The same example with a subset of the instances $n = 30$. However, the panel has been rotated so that the first principal component direction Z_1 coincides with the x-axis.

From the equations (4) or (5),

$$z_{i1} = 0.839 \times (x_{i,1} - \bar{X}_1) + 0.544 \times (x_{i,2} - \bar{X}_2) \quad (6)$$

where $\phi_{11} = 0.839$ and $\phi_{21} = 0.544$ are the *principal component loadings*

Example



Interpretation:

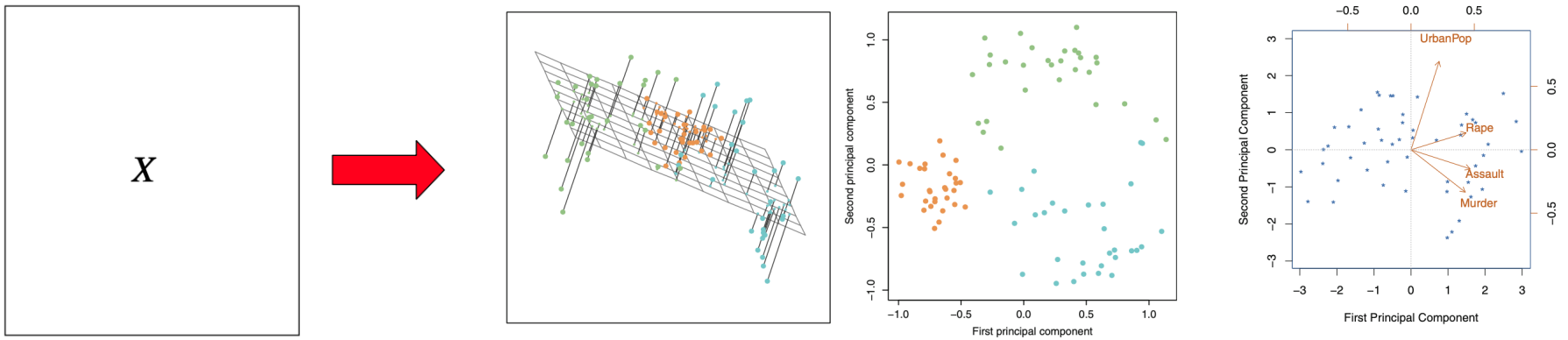
- The principal component **scores** z_{i1} **summary of the joint** X_1 **and** X_2 for each location (due to the linear relationship)
- **The sign of the score gives information**
  - If $z_{i1} < 0$, then this indicates a city with **below-average population size** and **below-average ad spending**
  - A positive score suggests the opposite

Motivations recap

- **Dimension reduction methods** involve projecting the d features from a high d-dimensional data matrix X into a M-dimensional subspace, where $M < d$
- This **low-dimensional representation** is achieved by computing successively M linear combinations/projections of the features subject to some constraints (higher variance in the projected data, normalization of ϕ_m, components uncorrelated)
- Principal components can interpreted as mathematical summary of the data matrix X features
- Then these M new features (princ. comp.) could be used to fit a model as **a solution to the problem linked to the curse of dimensionality**
- PCA is a popular example of dimension reduction methods

The data, the problem and its solution

- A **data matrix** X, dimensions: (n, d)
- **The problem**: look for "best" possible representation of observations and input variables (features) in a small-dimensional space generated by linear combinations of the original features
- These linear combinations are the orthogonal principal components
- **The solution**: PCA method



What are Principal Components Analysis?

- PCA is the **analysis of the structure of the variance-covariance matrix** i.e. variability, dispersion of data[a]
- **Goal**: **describe** using $M < d$ components a **maximum of this variability**
- PCA allows:
  - a **reduction** of the data to M new descriptors
  - a **visualization** of the data in 2 or 3 dimensions (if M = 2 or 3)
  - an **interpretation** of the data: inter-variable links
- PCA is an **intermediate step** often used **before other analyzes of the data**

[a] Except if one of the features can be expressed as a function of others, we need the all d features to take into account all the variability of the system

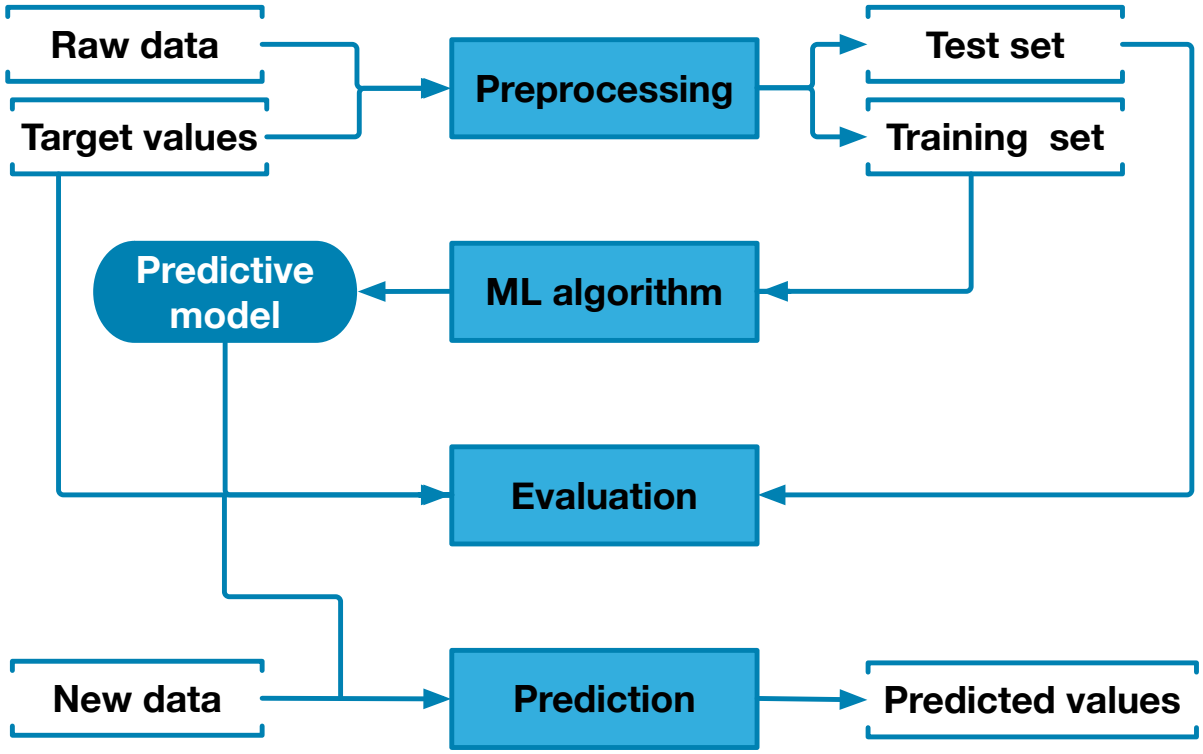


Figure: Machine learning pipeline from the lecture 1

PCA is a dimension reduction method useful during data preprocessing

- To obtain informations about the quality of the data
- Identify the more informative features and no-informative ones
- Identify outliers or a group of atypical observations

Principal components

Definition

The principal components (princ. comp.) are linear combinations of the original features $X_1, \ldots, X_d$ and denoted Z_m such as,

1. $$Z_m = \sum_{j=1}^{d} \phi_{jm} X_j \tag{7}$$

   where, $\phi_m^T = (\phi_{1m}, \ldots, \phi_{dm})$ is a free parameter to identify, $m \in [1, \ldots, M]$ and $M << d$
2. $\forall m$, Z_m are mutually uncorrelated
3. $\forall m$, Z_m have a maximal variance
4. Z_m are of decreasing importance

Thus, Z_1 is the first princ. comp. and must have the maximal variance

The First principal component Z_1

Geometrically, Z_1 determines a new direction in the data point cloud that follows the axis of maximum elongation (stretch) of the cloud

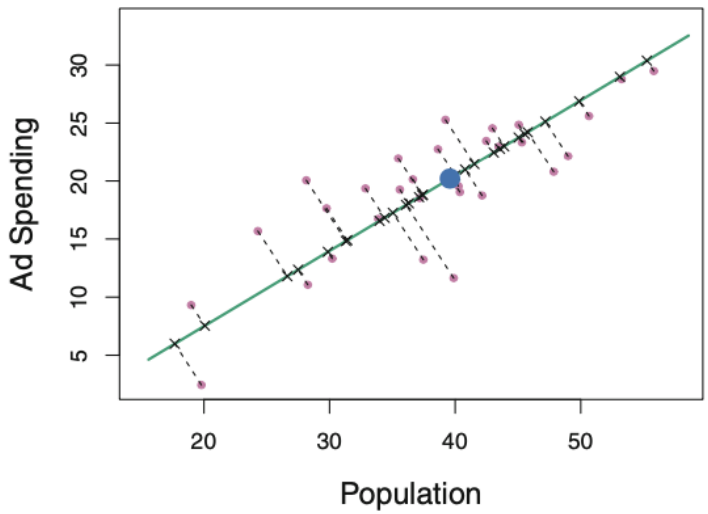


Figure: The green solid line indicates the first principal component Z_1.

We denote $z_{i,1} = \sum_{j=1}^{d} \phi_{j1} x_{i,j}$ the coordinate of data point i on the Z_1-axis projection of x_i on Z_1

The First principal component Z_1

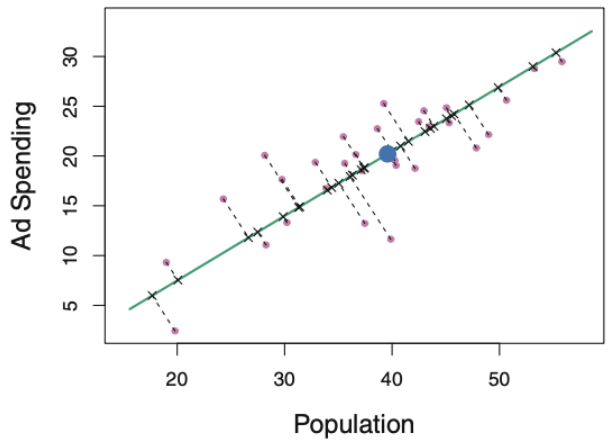


Figure: The green solid line indicates the first principal component Z_1.

- Z_1 has the maximal variance $\Rightarrow$ the projections z_{i1} are as dispersed as possible
- To fix the line, **we impose that it passes through the center of gravity** (otherwise all of the parallel lines are suitable)
- Thus, Z_1 is the line allowing the best possible adjustment of the cloud, i.e., which best preserves the distance between the data points (after projection)
- Then, the projection line Z_1 is ensuring minimal distortion

The Second principal component Z_2

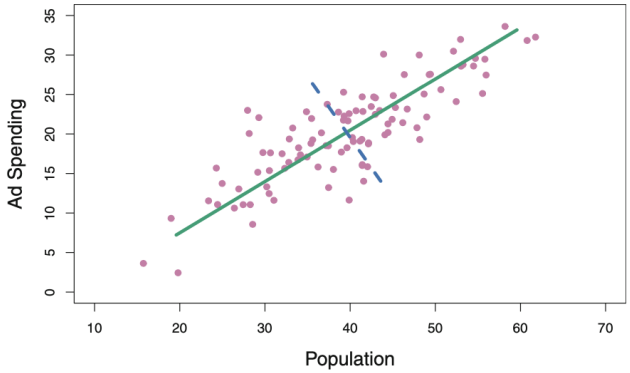


Figure: The green line indicates the first princ. comp. Z_1 and the blue dashed line corresponds to the second princ. comp. Z_2.

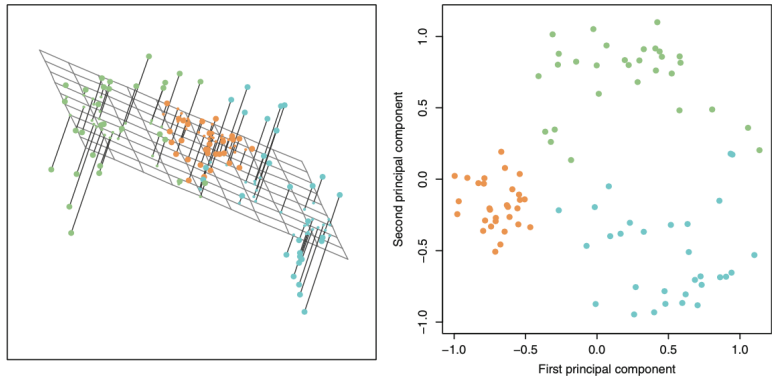


Figure: Illustration of a best projection plan provided by both first and second princ. comp..

- Z_2 is the second princ. comp., **orthogonal to Z_1, and ensuring maximal variance**
- Geometrically, Z_2 determines a line orthogonal to Z_1 (at the center of gravity) of maximum elongation
- Z_1 **and** Z_2 **provide the principal plan that are the best projection plan** (ensuring minimal distorsion)

- One can continue the process of determining another princ. comp.
- Z_3 is the line orthogonal to both Z_1 and Z_2 (at the center of gravity) such as the data points' variance is maximal
- ... until $m = M$ fixed

How compute Z_1?

Given a (n, d) data matrix X,

- Assumption: each feature X_j has been centered to have a zero mean
- We look for,

$$\underbrace{Z_1}_{\in \mathbb{R}^n} = \underbrace{X}_{\in \mathbb{R}^{n \times d}} \underbrace{\phi_1}_{\in \mathbb{R}^d} = \sum_{j=1}^{d} \underbrace{\phi_{jm}}_{\in \mathbb{R}} \underbrace{X_j}_{\in \mathbb{R}^n} \tag{8}$$

that has largest sample variance, subject to the constraint that $\sum_{j=1}^{d} \phi_{j1}^2 = 1$

How compute Z_1?

Thus, we compute the variance,

$$
\begin{aligned}
V(Z_1) &= V(X\Phi_1) \\
&= \Phi_1^T V(X)\Phi_1 \quad \text{because } \Phi_1 \text{ is a constant vector} \\
&\propto \Phi_1^T X^T X \Phi_1 \quad \text{because } X_j \text{ are centered and } V(X) = (n-1)^{-1} X^T X \\
&= \Phi_1^T \Sigma \Phi_1
\end{aligned}
$$

- Σ is the variance-covariance matrix of X
- Σ is a symmetric and positive definite matrix

How compute Z_1?

As a result, we have to solve the following optimization problem,

Constrained optimization problem to find Z_1

$$\begin{cases} \max_{\Phi_1}(\Phi_1^T \Sigma \Phi_1) \\ \Phi_1^T \Phi_1 = 1 \end{cases} \tag{9}$$

Thus, we write the Lagrange function,

$$\mathcal{L} = \Phi_1^T \Sigma \Phi_1 + \lambda(1 - \Phi_1^T \Phi_1) \tag{10}$$

Compute the necessary optimality condition,

$$\partial_{\Phi_1} \mathcal{L} = 0 \tag{11}$$

How compute Z_1?

We obtain the eigenvalue equation,

$$\Sigma\Phi_1 = \lambda\Phi_1 \tag{12}$$

- Since, Σ is a symmetric and positive definite matrix, the eigen values are real and positive
- The eigenvectors can be chosen to be orthonormal

Therefore, the solution of the constrained optimization problem (9) is to

- **chose Φ_1 as the unitary Σ's eigenvector** associated with the greatest λ eigenvalue **obtained by the diagonalization of Σ**
- i.e. to project the data on the eigenvector having the highest eigenvalue λ

How compute Z_2?

To find the second axis of maximum variance, we look for,

Constrained optimization problem to find Z_2

$$\begin{cases} \max_{\Phi_2}(\Phi_2^T \Sigma \Phi_2) \\ \Phi_2^T \Phi_2 = 1 \\ \Phi_2^T \Phi_1 = 0 \end{cases} \tag{13}$$

- where, Φ_1 is the first eigenvector having the highest eigenvalue λ
- As the eigenvectors of Σ are naturally orthonormal, **the solution is to choose Φ_2 as the second eigenvector of Σ** (with second highest eigenvalue)

One can continue the process to determine the next princ. comp.

Interpretation of eigenvalues

- The sum of the eigenvalues corresponds to the total variance

$$tr(\Sigma) = V(X) = \sum_{j=1}^{d} \lambda_j$$

- **Each eigenvalue measures the amount of variance explained by the corresponding princ. comp. axis**

PCA recap

- **Perform a PCA of X is compute Z_m $\forall m \in [1, \ldots, M]$** (princ. comp.)
  - that means identify Φ_m (parameters),
  - i.e., determine λ (eigen values)
- Φ_m are provided by a diagonalization of Σ
- $tr(\Sigma) = \sum_{j=1}^{d} \lambda_j$ gives information about **the amount of explained variance** or cumulative explained variance
- **In practice, X_j are often centered and reduced**

PCA recap

- **PCA replace d original features X_j** (having different variances and correlation between each other) **by M new components Z_m** ($M << d$):
  - mutually orthogonal, i.e. $Cov(Z_m, Z_{m'}) = 0, \forall m \neq m'$
  - having highest variance
- $V(Z_1) \geq V(Z_2) \geq \ldots \geq V(Z_M)$ **decreasing importance**
- $M << d$ to **reduce dimension** of the original data space
- **Highlighting linear relationships in data**

Assumption:

- In "reality", the data live in a subspace of reduced dimensions $M < d$ and PCA help to find it
- The maximum number M of princ. comp. correspond to the intrinsic dimension of the data

How many principal component to use/choose?

- **Goal**: keep as much information as possible from the original data X
- **Measurement of this information: the cumulative Proportion of Variance Explained via a PCA with M princ. comp.** (unit: percentage):

Definition (The Proportion of Variance Explained (PVE))

The Proportion of Variance Explained (PVE) by the mth princ. comp. Z_m is given by,

$$\frac{V(Z_m)}{\sum_{j=1}^{d} V(X_j)} \tag{14}$$

- The PVE of each principal component is a positive quantity
- In total, there are $min(n-1, d)$ princ. comp., and their PVEs sum to one (i.e. cumulative PVE equals to one)
- Note: if the original variables are strongly correlated with each other, a reduced number of comp. can explain 80% to 90% of the variance

How many principal component to use?

Some empirical **rules** about the value of M,

- **cumulative inertia percentage greater than** 80% (for n and d "large enough")
- **rupture in the histogram of the eigenvalues**
- **last eigenvalue greater than 1** (Kaiser criterion, only for centered and reduced data)

- Once we have computed the princ. comp., we can plot them against each other in order to produce low-dimensional views of the data
- We can plot the score vector Z_1 against Z_2, Z_1 against Z_3, Z_2 against Z_3, and so forth
- Geometrically, this amounts to projecting the original data down onto the subspace spanned by Φ_1, Φ_2, or Φ_1, Φ_3 or Φ_2, Φ_3 and plotting the projected points.

Example

The Population Size and Ad Spending data for different cities

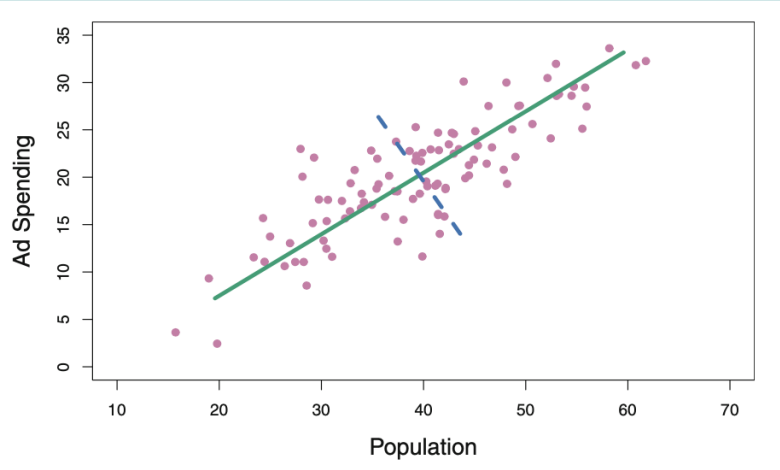


Figure: The green line indicates the first princ. comp. Z_1 and the blue dashed line corresponds to the second princ. comp. Z_2.

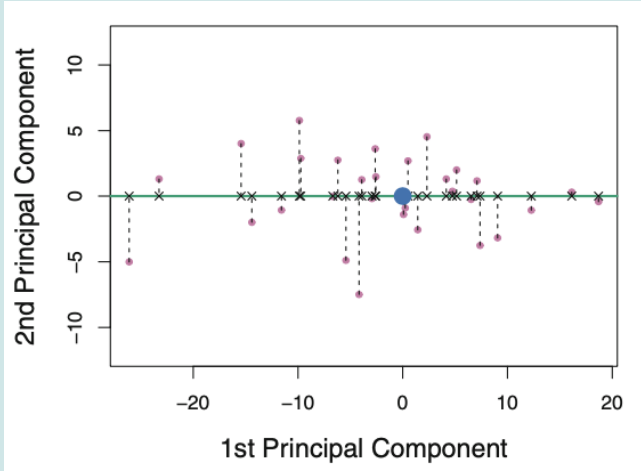


Figure: The same example with a subset of $n = 30$. The panel is rotated so that the Z_1 direction coincides with the x-axis. This subspace is spanned by Φ_1, Φ_2.

$$z_{i1} = 0.839 \times (x_{i,1} - \bar{X}_1) + 0.544 \times (x_{i,2} - \bar{X}_2)$$

where, $\phi_{11} = 0.839$ and $\phi_{21} = 0.544$ are the *principal component loadings*

Illustration of the use of PCA on the USArrests data set

▸ Reference

- **USArrests data**: for each of the $n = 50$ **states (instances)** in the US, the data set contains **the number of arrests per** $100,000$ **residents for each of three crimes**: Assault, Murder, and Rape. We **also record** UrbanPop (the percent of the population in each state living in urban areas); i.e., $d = 4$.

About the PCA:

- The princ. comp. score vectors have length $n = 50$
- **We choose the princ. comp.** loading vectors have **the maximal length** $M = 4$ (it's not usual but it's a pedagogical illustration)
- PCA was performed after standardizing each feature X_j to have mean zero and standard deviation one.
  Thus, each data instance x_{ij} is transformed by,

$$\frac{x_{ij} - \bar{X}_j}{\sqrt{V(X_j)}}$$

	PC1	PC2
Murder	0.5358995	−0.4181809
Assault	0.5831836	−0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

Figure: The princ. comp. loading vectors of Φ_1 and Φ_2, for the USArrests data. **Each loading coefficient ϕ_{jm} corresponds to the contribution of X_j to Z_m** (denoted PCm).

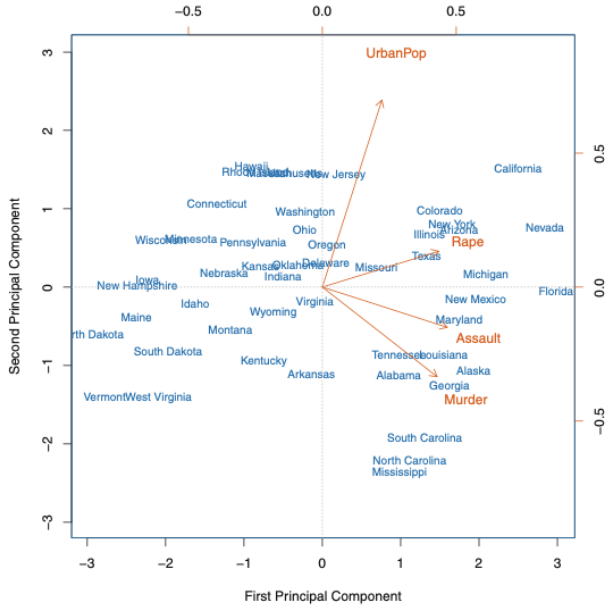


Figure: The first two princ. comp. for the USArrests data. The **blue state names represent the scores for Z_1 and Z_2. The orange arrows indicate the** features in the first two princ. comp. loading vectors Φ_1 **and** Φ_2 space.

- Together, the **first two princ. comp. explain almost** 87% **of the variance in the data**, and the last two princ. comp. explain only 13 % of the variance

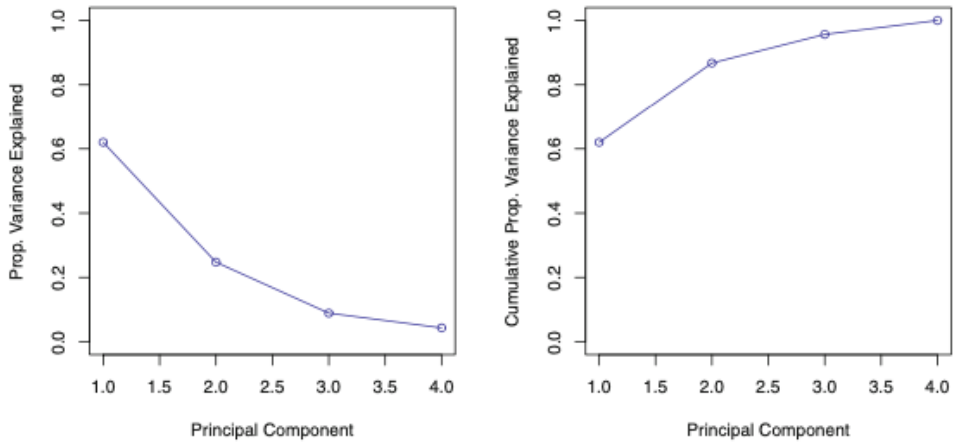
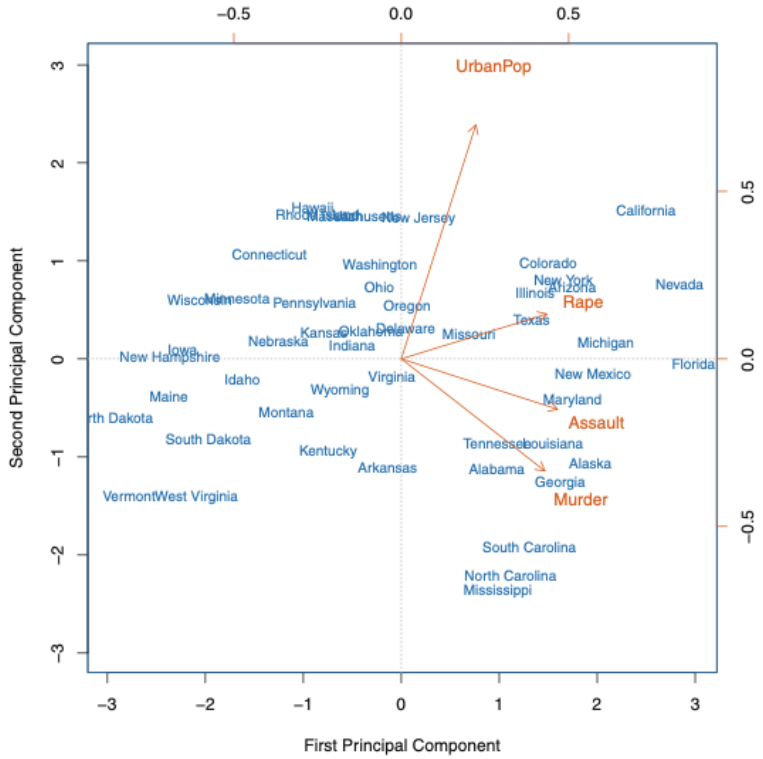


Figure: Left: a histogram plot depicting the PVE by each of the four princ. comp. in the USArrests data. Right: the cumulative PVE by the four princ. comp.

- **The PVE** of each princ. comp., as well as the cumulative PVE are displayed and **show a rupture in the histogram** indicating to **chose** $M = 2$
- Note that the eigenvalues and PVE values are proportional
- Thus, the previous slide's Figure provides a pretty accurate summary of the data using just two dimensions ($M = 2$)

Interpretation: features' representation



- The **first loading vector places approximately equal weight on Assault, Murder, and Rape**, with much less weight on UrbanPop
  $\hookrightarrow$ Z_1 **corresponds to** a measure of overall rates of **serious crimes**
- Z_2 **places most of its weight on UrbanPop**: Z_2 corresponds to the **level of urbanization** of the state

Interpretation: features' representation

- The crime-related features (Murder, Assault, and Rape) are located close to each other (close to Z_1), and the UrbanPop variable is far from the others (close to Z_2)
- This indicates that
  - the crime-related features are correlated with each other and contribute to Z_1
    $\hookrightarrow$ **interpretation**: **states with high murder rates tend to have high assault and rape rates**
  - t**he UrbanPop variable is less correlated with the other three and contribute to** Z_2

Interpretation: instances' representation

Now, what about the states?

- The **states with large positive scores on** Z_1, such as California, Nevada and Florida, have **high crime rates**
- While **states** like North Dakota, **with negative scores on** Z_1, have **low crime rates**
- California also has a **high score on** Z_2, **indicating a high level of urbanization**
- While the opposite is true for states like Mississippi
- **States close to zero on both components**, such as Indiana, have approximately **average levels of both crime and urbanization**

Scaling the features

Why before PCA is performed, the features have been scaled (to have a standard deviation one)?

- If some original features are very dispersed, they will dominate (take precedence) over the others
- Consequence: the princ. comp. will essentially try to explain the variance due to these variables
- When we scale (or standardize) all features will have the same "importance" (are on the same scale)
- Then, **the variance-covariance matrix Σ contains the intrinsic structure of the data and the PCA explains this structure**

Scaling the features (illustration)

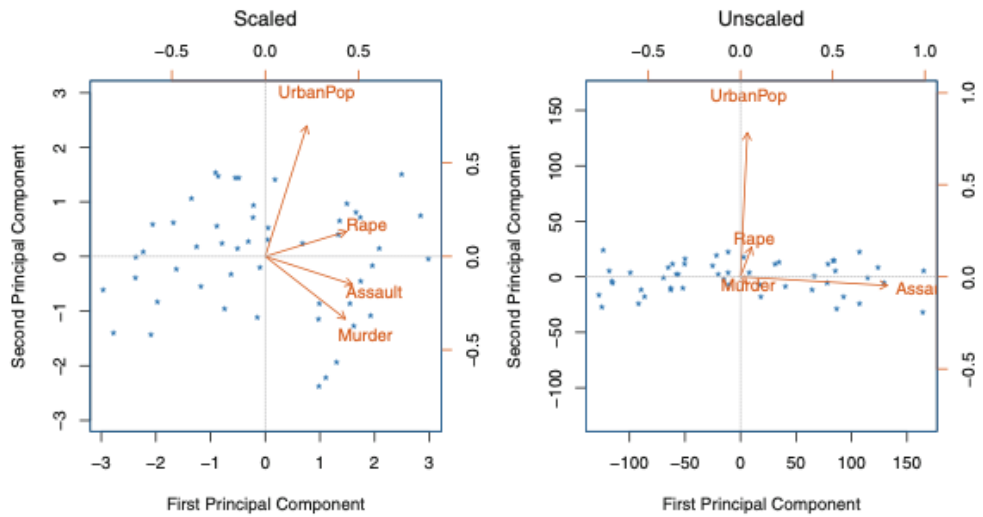
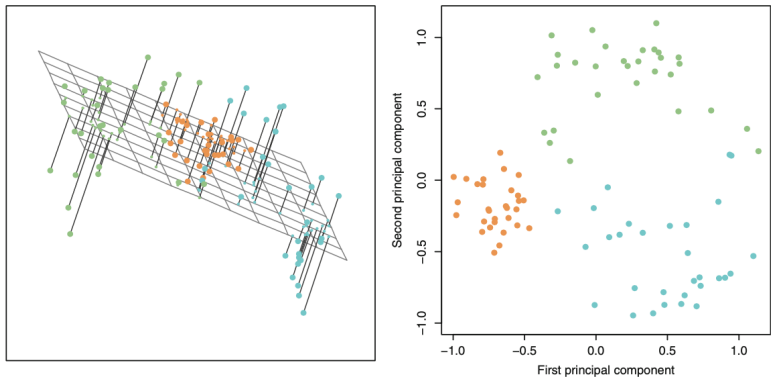


Figure: Two princ. comp. **biplots** for the USArrests data. **Left: with the scaled features. Right: with unscaled data.**

- **Cause 1**. Assault has by far the largest loading on Z_1 because it has the highest variance among the four features: V(Murder)=18.97, V(Rape)=87.73, **V(Assault)=6945.16**, and V(UrbanPop)=209.5 $\hookrightarrow$ Φ_1 places almost all of its weight on Assault
- **Cause 2.** Another possibility: the features are measured in different units (but here it's not the cause)

Interpretation Recap: instances' representation

- **Instances' representation in the biplot can highlight instances' groups having similarities**



- Beware of abusive proximity due to projections
- **The representation is only valid if the percentage of variance explained by Z_1 and Z_2 is sufficiently large** (cloud quite flattened on the plane)
- Think to check if the proximities are maintained in other projection planes: (O, Z_1, Z_3), (O, Z_2, Z_3), etc.
- The best represented instances are the closest to the plane (not very important projection)

Interpretation Recap: features' representation

- Plotting the correlations between the princ. comp. and the original features allows to identify the groups of highest correlated variables ($corr = 1$ or $corr = -1$)
- PCA correlation circle allows interpretation of Z_1 and Z_2
- Each plotted feature are in a circle of radius one with the coordinates $(corr(Z_1, X_j), corr(Z_2, X_j))$
- We can then identify the groups of related or opposite features
- If the features are close to the circumference: they are well represented by the 2 princ. comp.

PLS: a supervised alternative to PCA

- Now, we know that PCA involves identifying linear combinations, or directions, that best represent the original features
- These directions are identified in an **unsupervised way**, since **a target variable Y is not used to help determine the princ. comp. directions**
- Consequently, **a PCA's drawback: there is no guarantee that the directions that best explain the features will also be the best directions to use for predicting Y**
- Partial Least Squares (**PLS**) method is **an supervised alternative to PCA**

Remark: A supervised PCA could help for tuning M as a hyperparameter by a cross-validation way

FCA

- Factorial Correspondence Analysis (FCA) is conceptually similar to PCA, but applies to **analyze the association between two qualitative variables**
- FCA is performed on a contingency table
- FCA creates orthogonal components and, for each item in a table, a set of scores (sometimes called factor scores)

Example (A contingency table)

Couleur des yeux	Couleur des cheveux				
	Fair	Red	Medium	Dark	Black
BLUE	326	38	241	110	3
LIGHT	688	116	584	188	4
MEDIUM	343	84	909	412	26
DARK	98	48	403	681	85

Figure: R.A. Fisher's data (1940) provide, for 5387 children in Caithness (Scotland), the joint distribution (number) for hair and eye color.

FCA

Example (FCA's results on R.A. Fisher's data)

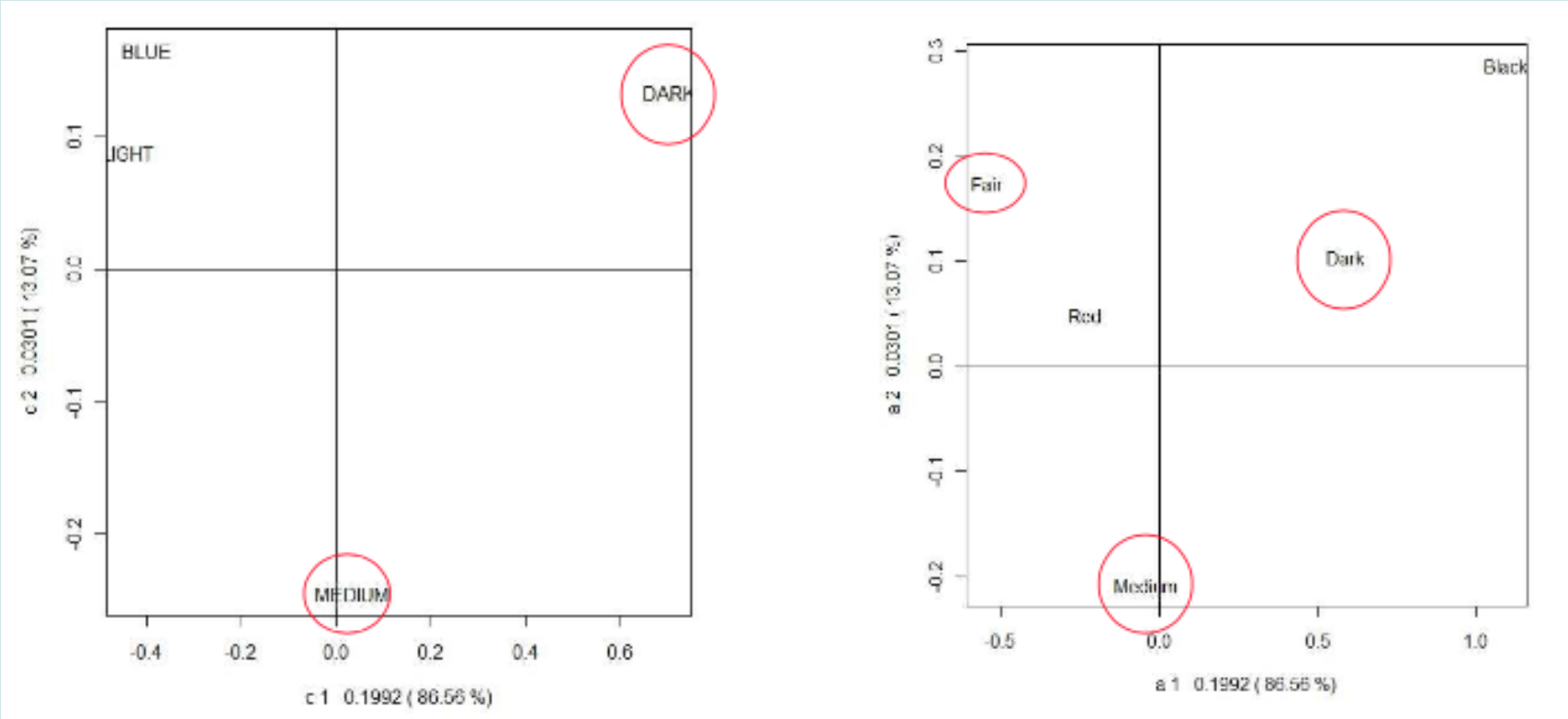


Figure: Representations of row and column modalities

MCA

- Multiple Correspondence Analysis (MCA) is conceptually similar to PCA, but applies to categorical rather than continuous data
- MCA is an extension of the simple FCA **for summarizing and visualizing a data table containing more than two categorical variables**
- MCA is generally used to analyze a data set from survey. The goal is to identify:
  - A group of individuals with similar profile in their answers to the questions
  - The associations between variable categories

Conclusion

- Now, new technologies provide essentially high-dimensional data
- Traditional (statistical) approaches are not intended for High dimensional data:
- **Overfitting is a serious problem/consequence of the curse of dimensionality**
- It's possible to use subset selection approaches to reduce the number of features from a data matrix
- But some computational limitations encourage the use of **stepwise selection algorithms** rather than best subset selection
- **Dimension reduction methods are useful to both tackle the problem due to the curse of dimensionality and preprocess the data**
- The most popular dimension reduction approach is PCA
- Sometimes, for a regression task, PCA allows to obtain better accurate results than using original data