A Basic Machine Learning Kit

Lorenzo Rosasco,
- Universita’ di Genova
- Istituto Italiano di Tecnologia

August, 14th 2018 - BMM Summer School, Woods Hole
GOAL: Introduce key algorithms that you can use and complicate when needed
PART I
• Local methods
• Bias-Variance
• Cross Validation

PART II
• Linear Least Squares
• Features and Kernels
• Deep Neural Nets

PART III
• Variable Selection: OMP
• Dimensionality Reduction: PCA

PART IV
• Matlab practical session

Morning

Afternoon
PART I

- Local methods
- Bias-Variance
- Cross Validation

**GOAL:** Investigate the trade-off between stability and fitting starting from simple machine learning approaches
The goal of supervised learning is to find an underlying input-output relation
\[
f(x_{\text{new}}) \sim y,
\]
given data.

The data, called training set, is a set of \( n \) input-output pairs,
\[
S = \{(x_1, y_1), \ldots, (x_n, y_n)\}.
\]
\[ X_n = \begin{pmatrix} x_1^1 & \ldots & \ldots & \ldots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^1 & \ldots & \ldots & \ldots & x_n^p \end{pmatrix} \]

\[ Y_n = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \]
Local Methods: Nearby points have similar labels

**Nearest Neighbor**

Given an input $\bar{x}$, let

$$i' = \arg \min_{i=1,\ldots,n} \| \bar{x} - x_i \|^2$$

and define the nearest neighbor (NN) estimator as

$$\hat{f}(\bar{x}) = y_{i'}.$$
Demo
We describe a simple yet efficient class of algorithms, the so-called memory-based learning algorithms, based on the principle that nearby input points should have the same/similar output.

2.1. Nearest Neighbor

Consider a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$. Given an input $\bar{x}$, let $i_0 = \text{arg min}_{i = 1, \ldots, n} \|\bar{x} - x_i\|^2$ and define the nearest neighbor (NN) estimator as

$$\hat{f}(\bar{x}) = y_{i_0}.$$ 

Every new input point is assigned the same output as its nearest input in the training set. We add few comments. First, while in the above definition we simply considered the Euclidean norm, the method can be promptly generalized to consider other measures of similarity among inputs. For example, if the inputs are binary strings, i.e., $X = \{0, 1\}$, one could consider the Hamming distance $d_H(\bar{x}, x_i) = \sum_{j=1}^D \left[ x_{ij} \neq \bar{x}_{ij} \right]$ where $x_{ij}$ is the $j$-th component of a string $x_i$.

Second, the complexity of the algorithm for predicting any new point is $O(nD)$ – recall that the complexity of multiplying two $D$-dimensional vectors is $O(D)$. Finally, we note that NN can be fairly sensitive to noise.

2.2. K-Nearest Neighbor

Consider $d_{\bar{x}} = (\|\bar{x} - x_i\|^2)_{i=1}^n$ the array of distances of a new point $\bar{x}$ to the input points in the training set. Let $s_{\bar{x}}$ be the above array sorted in increasing order and $I_{\bar{x}}$ the corresponding vector of indices, and

$$K_{\bar{x}} = \{I_{\bar{x}}^1, \ldots, I_{\bar{x}}^K\}$$

be the array of the first $K$ entries of $I_{\bar{x}}$. The $K$-nearest neighbor estimator (KNN) is defined as,

$$\hat{f}(\bar{x}) = \sum_{i' \in K_{\bar{x}}} y_{i'},$$

or

$$\hat{f}(\bar{x}) = \frac{1}{K} \sum_{i' \in K_{\bar{x}}} y_{i'}.$$
Demo
Remarks:

**Generalization I:** closer points should count more

\[
\hat{f}(\bar{x}) = \frac{\sum_{i=1}^{n} y_i k(\bar{x}, x_i)}{\sum_{i=1}^{n} k(\bar{x}, x_i)}, \quad \text{Gaussian} \quad k(x', x) = e^{-\|x-x'\|^2/2\sigma^2}.
\]

**Parzen Windows**

**Generalization II:** other metric/similarities

\[
X = \{0, 1\}^D, \quad d_H(x, \bar{x}) = \frac{1}{D} \sum_{j=1}^{D} 1[x^j \neq \bar{x}^j]
\]

There is one parameter controlling fit/stability
How do we choose it?

Is there an optimal value?

Can we compute it?
Is there an optimal value?

Ideally we would like to choose $K$ that minimizes the expected error

$$E_S E_{x,y} (y - \hat{f}_K(x))^2.$$ 

Next: Characterize corresponding minimization problem to uncover one of the most fundamental aspect of machine learning.
For the sake of simplicity we consider a regression model

\[ y_i = f_*(x_i) + \delta_i, \quad \mathbb{E}\delta_I = 0, \mathbb{E}\delta_i^2 = \sigma^2 \quad i = 1, \ldots, n \]

further let

\[ f_K(x) = \mathbb{E}\hat{f}_K(x) = \frac{1}{K} \sum_{\ell \in K_x} f_*(x_\ell) \]

**Error decomposition**

\[
\mathbb{E}(y - \hat{f}_K(x))^2 = \mathbb{E}(y - f_*(x))^2 + \mathbb{E}(f_*(x) - f_K(x))^2 + \mathbb{E}(f_K(x) - \hat{f}_K(x))^2
\]

\[
\sigma^2 \quad \frac{\sigma^2}{Kn} \quad \mathbb{E}(f_*(x) - \frac{1}{K} \sum_{\ell \in K_x} f_*(x_\ell))^2
\]

Irreducible error  
Bias  
Variance
5.2 The Bias Variance Trade-Off

We are ready to discuss the behavior of the (point-wise) expected loss of the KNN algorithm as a function of $K$. As it is clear from the above equation, the variance decreases with $K$. The bias is likely to increase with $K$, if the function $f_* \mid x \rangle$ is suitably smooth. Indeed, for small $K$ the few closest neighbors to $x$ will have values close to $f_* \mid x \rangle$, so their average will be close to $f_* \mid x \rangle$. Whereas, as $K$ increases neighbors will be further away and their average might move away from $f_* \mid x \rangle$. A larger bias variance is preferred when data are few/noisy to achieve a better control of the variance, whereas the bias can be decreased as more data become available, hence reducing the variance. For any given training set, the best choice for $K$ would be the one striking the optimal trade-off between bias and variance (that is the value minimizing their sum).

5.3 Cross Validation

While instructive, the above analysis is not directly useful in practice since the data distribution, hence the expected loss, is not accessible. In practice, data driven procedures are used to find a proxy for the expected loss. The simplest such procedure is called hold-out cross validation. Part of the training set is hold-out, to compute a (holdout) error to be used as a proxy of the expected error. An empirical bias variance trade-off is achieved choosing the value of $K$ that achieves minimum hold-out error. When data are scarce the hold-out procedure, based on a simple "two ways split" of the training set, might be unstable. In this case, so called $V$-fold cross validation is preferred, which is based on multiple data splitting. More precisely, the data are divided in $V$ (non overlapping) sets. Each set is hold-out and used to compute an hold-out error which is eventually averaged to obtained the final $V$-fold cross validation error. The extreme case where $V = n$ is called leave-one-out cross validation.

5.3.1 Conclusions: Beyond KNN

Most of the above reasonings hold for a large class of learning algorithms beyond KNN. Indeed, many (most) algorithms depend on one or more parameter controlling the bias.

---

$$
\mathbb{E}(y - \hat{f}_K(x))^2 = \mathbb{E}(y - f_*(x))^2 + \mathbb{E}(f_*(x) - f_K(x))^2 + \mathbb{E}(f_K(x) - \hat{f}_K(x))^2
$$

---

Is there an optimal value? YES!

Can we compute it?
Not quite...

\[
E(y - \hat{f}_K(x))^2 = E(y - \hat{f}_*(x))^2 + E(\hat{f}_*(x) - f_K(x))^2 + E(f_K(x) - \hat{f}_K(x))^2
\]

Irreducible error

\[
\sigma^2
\]

Bias

\[
E(\hat{f}_*(x) - \frac{1}{K} \sum_{\ell \in K_x} f_*(x_\ell))^2
\]

Variance

\[
\frac{\sigma^2}{Kn}
\]

…enter Cross Validation

Split data: train on some, tune on some other
Hold-Out
Cross Validation Flavors

V-Fold, (V=n is Leave-One-Out)
Actual protocol

Training - Validation - Test
Perils of data mining
End of PART I

- Local methods
- Bias-Variance
- Cross Validation

Stability - Overfitting - Bias/Variance - Cross-Validation

End of the Story?
tell me the length of the edge of a cube containing 1% of the volume of a cube with edge 1

Curse of dimensionality!
PART II

• Linear Least Squares
• Features and Kernels
• Deep Neural Nets

GOAL: Introduce the basic (global) regularization methods with linear and non linear models

Going Global + Impose Smoothness
Of all the principles which can be proposed for that purpose, I think there is none more general, more exact, and more easy of application, that of which we made use in the preceding researches, and which consists of rendering the sum of squares of the errors a minimum.

(Legendre 1805)

\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda w^T w, \quad \lambda \geq 0
\]
Of all the principles which can be proposed for that purpose, I think there is none more general, more exact, and more easy of application, that of which we made use in the preceding researches, and which consists of rendering the **sum of squares of the errors** a minimum.

(Legendre 1805)

\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda w^T w, \quad \lambda \geq 0
\]
\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda w^T w, \quad \lambda \geq 0
\]
\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda w^T w, \quad \lambda \geq 0
\]

Computations?

\[
-\frac{2}{n} X_n^T (Y_n - X_n w), \quad \text{and}, \quad 2w
\]

...to zero

\[
(X_n^T X_n + \lambda n I)w = X_n^T Y_n
\]

OK, but what is this doing?
Interlude: Linear Systems

\[ Ma = b, \]

- If \( M \) is a diagonal \( M = \text{diag}(\sigma_1, \ldots, \sigma_D) \) where \( \sigma_i \in (0, \infty) \) for all \( i = 1, \ldots, D \), then
  \[ M^{-1} = \text{diag}(1/\sigma_1, \ldots, 1/\sigma_D), \quad (M + \lambda I)^{-1} = \text{diag}(1/(\sigma_1 + \lambda), \ldots, 1/(\sigma_D + \lambda)) \]

- If \( M \) is symmetric and positive definite, then considering the eigendecomposition
  \[ M^{-1} = V \Sigma V^T, \quad \Sigma = \text{diag}(\sigma_1, \ldots, \sigma_D), \quad V V^T = I, \]
  then
  \[ M^{-1} = V \Sigma^{-1} V^T, \quad \Sigma^{-1} = \text{diag}(1/\sigma_1, \ldots, 1/\sigma_D), \]
  and
  \[ (M + \lambda I)^{-1} = V \Sigma_\lambda V^T, \quad \Sigma_\lambda = \text{diag}(1/(\sigma_1 + \lambda), \ldots, 1/(\sigma_D + \lambda)) \]
\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda w^T w, \quad \lambda \geq 0
\]

Statistics?

\[
(X^T_n X_n + \lambda n I) w = X^T_n Y_n
\]

---

*another story that shall be told another time*

(Stein '56, Tikhonov'61)

\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda w^T w, \quad \lambda \geq 0
\]

\[
f_w(x) = w^T x = \sum_{i=1}^{v} w^j x^j
\]

Shrinkage - Regularization
Demo
Why a linear decision rule?
Dictionaries

\[ x \mapsto \tilde{x} = (\phi_1(x), \ldots, \phi_p(x)) \in \mathbb{R}^p \]

\[ f(x) = w^T \tilde{x} = \sum_{j=1}^p \phi_j(x) w^j \]

\[ \Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3 \]

\[ (x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2) \]
A direct computation shows that the gradient with respect to $w$ of the empirical risk and the regularizer are respectively

$$2nX_n^T n(Y_nX_nw),$$

and

$$2w.$$

Then, setting the gradient to zero, we have that the solution of regularized least squares solves the linear system

$$(X_n^T X_n + \lambda nI)w = X_n^T Y_n$$

$$\iff (\tilde{X}_n^T \tilde{X}_n + \lambda nY_n)w = \tilde{X}_n^T Y_n$$

What About Computational Complexity?
Complexity Vademecum

$M$ $n$ by $p$ matrix and $v, v'$ $p$ dimensional vectors

- $v^T v' \rightarrow O(p)$
- $M v' \rightarrow O(np)$
- $M M^T \rightarrow O(np^2)$
- $(M M^T)^{-1} \rightarrow O(n^3)$
\[(X_n^T X_n + \lambda n I)w = X_n^T Y_n \quad \leftrightarrow \quad (\tilde{X}_n^T \tilde{X}_n + \lambda n Y)w = \tilde{X}_n^T Y_n\]

**What About Computational Complexity?**

\[O(p^3) + O(p^2 n)\]

**What if \(p\) is much larger than \(n\)?**

\[(X_n^T X_n + \lambda n I)^{-1} X_n^T = X_n^T (X_n X_n^T + \lambda n I)^{-1}\]

\[w = X_n^T \left( X_n X_n^T + \lambda n I \right)^{-1} Y_n = \sum_{i=1}^{n} x_i^T c_i\]
The result follows noting that the following equality holds,

\[ (X_n^T X_n + \lambda n I)^{-1} X_n = X_n^T (X_n X_n^T + \lambda n I)^{-1} \]

\[ w = X_n^T (X_n X_n^T + \lambda n I)^{-1} Y_n = \sum_{i=1}^{n} x_i^T c_i \]

**Computational Complexity:** \( O(p^3) + O(p^2 n) \)
\[
(X_n^T X_n + \lambda n I)^{-1} X_n^T = X_n^T (X_n X_n^T + \lambda n I)^{-1}
\]

\[
w = X_n^T (X_n X_n^T + \lambda n I)^{-1} Y_n = \sum_{i=1}^{n} x_i^T c_i
\]

**Kernels**

\[
w = \sum_{j=1}^{n} x_i c_i \implies f(x) = x^T w = \sum_{j=1}^{n} x_j^T x_i c_i
\]

\[
(K_n + \lambda n I) c = Y_n, \quad (K_n)_{i,j} = K(x_i, x_j)
\]

- the linear kernel \(K(x, x') = x^T x'\),
- the polynomial kernel \(K(x, x') = (x^T x' + 1)^d\),
- the Gaussian kernel \(K(x, x') = e^{-\frac{||x-x'||^2}{2\sigma^2}}\),
\[ \hat{f}(x) = \sum_{i=1}^{n} K(x_i, x)c_i. \]

---

*things I won’t tell you about*

- Reproducing Kernel Hilbert Spaces
- Gaussian Processes
- Integral Equations
- Sampling Theory/Inverse Problems

- Loss functions- SVM, Logistic…
- Multi - task, labels, outputs, classes
Demo
6. FEATURE, KERNELS AND REPRESENTER THEOREM

Figure 1. A pictorial representation of the potential effect of considering a feature map in a simple two dimensional example.

where $x_1, \ldots, x_n$ are the inputs in the training set and $c = (c_1, \ldots, c_n)$ a set of coefficients. The above result is an instance of the so called representer theorem. We first discuss this result in the context of RLS.

6.2.1. Representer Theorem for RLS.

The result follows noting that the following equality holds,

$$\left( X^T X + nI \right)^{-1} = X^T (X X^T + nI)^{-1} X,$$

so that we have,

$$w = X^T (X X^T + nI)^{-1} Y,$$

Equation (6.3) follows from considering the SVD of $X$, that is $X = U \Sigma V^T$. Indeed we have $X^T = V \Sigma U^T$ so that

$$\left( X^T X + nI \right)^{-1} X = V \Sigma^{-1} U^T \left( \Sigma^2 + nI \right)^{-1} U \Sigma V^T X = V \Sigma^{-1} \left( \Sigma^2 + nI \right)^{-1} \Sigma U^T X.$$

6.2.2. Representer Theorem Implications.

Using Equation (7.2) it possible to show how the vector of coefficients can be computed considering different loss functions. In particular, for the square loss the vector of coefficients satisfies the following linear system

$$K_n c = Y_n,$$

where $K_n$ is the $n$ by $n$ matrix with entries $(K_n)_{i,j} = x_i^T x_j$. The matrix $K_n$ is called the kernel matrix and is symmetric and positive semi-definite.

6.3. Kernels

One of the main advantages of using the representer theorem is that the solution of the problem depends on the input points only through inner products $x_i^T x_0$. Kernel methods can be seen as replacing the inner product with a more general function $K(x, x_0)$. In this case, the representer theorem (7.2), that is $f_w(x) = w^T \sum_{j=1}^p \phi_j(x) w^j$, becomes

$$\hat{f}(x) = \sum_{i=1}^n K(x_i, x) c_i.$$

Neural Networks

$$f(x) = \sum_{j=1}^p \beta_j \sigma(w_j^T x + b_j)$$
Neural Networks

\[ f(x) = \sum_{j=1}^{p} \beta_j \sigma(w_j^T x + b_j) \]

\[ \sigma(w_j^T x + b_j) = \sigma(\sum_{k=1}^{d} w_j^k x^k + b_j) \]
Deep Neural Networks

\[ f_W(x) = \beta^T \sigma(W_L \sigma(W_{L-1} \ldots \sigma(W_1 x))) \]
Newton method/Gradient descent

\[
\min_W \sum_{i=1}^{n} (f_W(x_i) - y_i)^2
\]

\[
W_{t+1} = W_t - \gamma \nabla_W \sum_{i=1}^{n} (f_{W_t}(x_i) - y_i)^2
\]
Stochastic gradient

\[ W_{t+1} = W_t - \gamma \nabla W \sum_{i=1}^{n} (f_{W_t}(x_i) - y_i)^2 \]

\[ W_{t+1} = W_t - \gamma \nabla W (f_{W_t}(x_t) - y_t)^2 \]

linear regression/logistic/svm

neural nets
Demo
End of PART II

• Linear Least Squares
• Kernel and features
• Deep Neural Nets
PART III

• a) Variable Selection: OMP
• b) Dimensionality Reduction: PCA

GOAL: To introduce methods that allow to learn *interpretable* models from data
$n$ patients $p$ gene expression measurements

\[
X_n = \begin{pmatrix}
  x^1_1 & \cdots & \cdots & \cdots & x^p_1 \\
  \vdots & \ddots & \ddots & \ddots & \vdots \\
  x^1_n & \cdots & \cdots & \cdots & x^p_n \\
\end{pmatrix};
Y_n = \begin{pmatrix}
y_1 \\
\vdots \\
y_n
\end{pmatrix}
\]
Which variables are important for prediction?

or

Torture the data until they confess

Sparsity: only some of the coefficients are non zero
Brute Force Approach

check all individual variables, then all couple, triplets…..

\[
\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_w(x_i))^2 + \lambda \|w\|_0,
\]

\[
\|w\|_0 = |\{ j \mid w^j \neq 0 \}|
\]
CHAPTER 11

Variable Selection

In many practical situations beyond predictions it is important to obtain interpretable results. Interpretability is often determined by detecting which factors have determined our prediction. We look at this question from the perspective of variable selection.

Consider a linear model

\[ f_w(x) = w^T x = \sum_{j=1}^{D} w_j x_j. \]

Here we can think of the components \( x_j \) of an input of specific measurements: pixel values in the case of images, dictionary words counting in the case of texts, etc. Given a training set the goal of variable selection is to detect which variables are important for prediction. The key assumption is that the best possible prediction rule is sparse, that is only few of the coefficients in (11.1) are different from zero.

11.1. Subset Selection

A brute force approach would be to consider all the training sets obtained considering all the possible subset of variables. More precisely we could begin by considering only the training set where we retain only the first variable of each input points. Then the one where we retain only the second, and so on and so forth. Next, we could pass to consider training set with pairs of variables, then triplet etc. For each training set one would solve the learning problem and eventually end selecting the variables for which the corresponding training set achieve the best performance.

The approach described has an exponential complexity and becomes unfeasible already for relatively small \( D \). If we consider the square loss, it can be shown that the corresponding problem could be written as

\[ \min_{w} \sum_{i=1}^{n} (y_i - f_w(x_i))^2 + \|w\|_0, \]

where \( \|w\|_0 = \{j | w_j \neq 0\} \) is called the \( \| \cdot \|_0 \) norm and counts the number of non zero components in \( w \). In the following we focus on the least squares loss and consider different approaches to find approximate solution to the above problem, namely greedy methods and convex relaxation.

11.2. Greedy Methods: (Orthogonal) Matching Pursuit

Greedy approaches are often considered to find approximate solution to problem (11.2). This class of approaches to variable selection generally encompasses the following steps:

1. initialize the residual, the coefficient vector, and the index set,
2. find the variable most correlated with the residual,
3. update the index set to include the index of such variable,
4. update/compute coefficient vector,
5. update residual.

The simplest such procedure is called forward stage-wise regression in statistics and matching pursuit (MP) in signal processing. To describe the procedure we need some notation. Let \( X \) be the \( n \times D \) data matrix and \( x_j \) be the columns of \( X \). Let \( Y \) be the output vector. Let \( r, w, I \) denote the residual, the coefficient vector, and index set, respectively.

The MP algorithm starts by initializing the residual \( r \), the coefficient vector \( w \), and the index set \( I \):

\[ r \equiv y, \quad w \equiv 0, \quad I \equiv \{1, \ldots, D\}. \]

(1) initialize the residual, the coefficient vector, and the index set,
(2) find the variable most correlated with the residual,
(3) update the index set to include the index of such variable,
(4) update/compute coefficient vector,
(5) update residual.
\[ r_0 = Y_n, \quad w_0 = 0, \quad I_0 = \emptyset. \]

for \( i = 1, \ldots, T - 1 \)

\[ k = \arg \max_{j=1,\ldots,D} a_j, \quad a_j = \frac{(r_{i-1}^T X_j)^2}{\|X_j\|^2}, \quad \bigcirc \]

\[ I_i = I_{i-1} \cup \{k\}. \]

\[ w_i = w_{i-1} + w_k, \quad w_k k = v_k e_k \]

\[ r_i = r_{i-1} - X w^k. \]

end

Matching Pursuit

(Mallat Zhang '93)

\[ \bigcirc v^j = \frac{r_{i-1}^T X_j}{\|X_j\|^2} = \arg \min_{v \in \mathbb{R}} \|r_{i-1} - X^j v\|^2, \quad \|r_{i-1} - X^j v^j\|^2 = \|r_{i-1}\|^2 - a_j \]
In many practical situations beyond predictions it is important to obtain interpretable results. Interpretability is often determined by detecting which factors have determined our prediction. We look at this question from the perspective of variable selection.

Consider a linear model

$$f_w(x) = w^T x = \sum_{i=1}^{n} w_i x_i.$$  \hspace{1cm} (20.1)

Here we can think of the components $x_j$ of an input of specific measurements: pixel values in the case of images, dictionary words counting in the case of texts, etc. Given a training set the goal of variable selection is to detect which variables are important for prediction. The key assumption is that the best possible prediction rule is sparse, that is only few of the coefficients in (20.1) are different from zero.

### 20.1 Subset Selection

A brute force approach would be to consider all the training sets obtained considering all the possible subset of variables. More precisely we could begin by considering only the training set where we retain only the first variable of each input points. Then the one where we retain only the second, and so on and so forth. Next, we could pass to consider training set with pairs of variables, then triplet etc. For each training set one would solve the learning problem and eventually end selecting the variables for which the corresponding training set achieve the best performance.

The approach described has an exponential complexity and becomes unfeasible already for relatively small $D$. If we consider the square loss, it can be shown that the corresponding problem could be written as

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_w(x_i))^2 + \lambda \|w\|_0,$$  \hspace{1cm} (20.2)

where $\|w\|_0 = \{|j| \mid w_j \neq 0\}$ is called the `0 norm and counts the number of non zero components in $w$. In the following we focus on the least squares loss and consider different approaches to find approximate solution to the above problem, namely greedy methods and convex relaxation.

Problem is now convex and can be solved using convex optimization, in particular so called proximal methods.
- Solving underdetermined systems
- Sampling theory
- Compressed Sensing
- Structured Sparsity
- From vector to matrices- from sparsity to low rank
End of PART III a)

- a) Variable Selection: OMP
- b) Dimensionality Reduction: PCA

Interpretability - Sparsity - Greedy & Convex Relaxation Approaches
PART III b)

- a) Variable Selection: OMP
- b) Dimensionality Reduction: PCA

**GOAL:** To introduce methods that allow to reduce data dimensionality in absence of labels, namely *unsupervised learning*
Dimensionality Reduction for Data Visualization
In many practical applications it is of interest to reduce the dimensionality of the data. In particular, this is useful for data visualization, or for investigating the "effective" dimensionality of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map $M: \mathbb{R}^D \rightarrow \mathbb{R}^k$, $k \ll D$.

**19.1 PCA & Reconstruction**

PCA is arguably the most popular dimensionality reduction procedure. It is a data driven procedure that given an (unsupervised) sample $S = (x_1, \ldots, x_n)$ derives dimensionality reduction defined by a linear map $M$. PCA can be derived from several prospective and here we give a geometric/analytical derivation.

We begin by considering the case where $k = 1$. We are interested in finding the single most relevant dimension according to some suitable criterion. Recall that, if $w \in \mathbb{R}^D$ with $k w = 1$, then the (orthogonal) projection of a point $x$ on $w$ is given by $(w^T x) w$. Consider the problem of finding the direction $p$ which allows the best possible average reconstruction of the training set, that is the solution of the problem

$$\min_{w \in \mathbb{S}^D} \frac{1}{n} \sum_{i=1}^n k x_i - (w^T x_i) w^2,$$

where $\mathbb{S}^D = \{w \in \mathbb{R}^D | k w = 1\}$ is the sphere in $D$ dimensions. The norm $k x_i - (w^T x_i) w^2$ measures how much we lose by projecting $x$ along the direction $w$, and the solution $p$ to problem (19.1) is called the first principal component of the data. A direct computation shows that $k x_i - (w^T x_i) w^2 = k x_i (w^T x_i)^2$, so that problem (19.1) is equivalent to

$$\max_{w \in \mathbb{S}^D} \frac{1}{n} \sum_{i=1}^n (w^T x_i)^2.$$  

(19.2)

This latter observation is useful for two different reasons that we discuss in the following.

**19.2 PCA and Maximum Variance**

If the data are centered, that is $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = 0$, problem (19.2) has the following interpretation: we look for the direction along which the data have (on average) maximum variance.
Dimensionality Reduction

\[ M : X = \mathbb{R}^D \rightarrow \mathbb{R}^k, \quad k \ll D, \]

Consider first \( k = 1 \)
Dimensionality Reduction

\[ M : X = \mathbb{R}^D \rightarrow \mathbb{R}^k, \quad k \ll D, \]

Consider first \( k = 1 \)

\[
\text{PCA} \quad \min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \| x_i - (w^T x_i) w \|^2,
\]

\[ w^T w = 1 \]

Computations? Statistics?
In many practical applications it is of interest to reduce the dimensionality of the data. In particular, this is useful for data visualization, or for investigating the “effective” dimensionality of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map $M : \mathbb{R}^D = \mathbb{R}^k$, $k \ll D$, according to some suitable criterion.

19.1 PCA & Reconstruction

PCA is arguably the most popular dimensionality reduction procedure. It is a data driven procedure that given an (unsupervised) sample $S = (x_1, \ldots, x_n)$ derive dimensional reduction defined by a linear map $M$. PCA can be derived from several prospective and here we give a geometric/analytical derivation.

We begin by considering the case where $k = 1$. We are interested in finding the single most relevant dimension according to some suitable criterion. Recall that, if $w \in \mathbb{R}^D$ with $k w = 1$, then the (orthogonal) projection of a point $x$ on $w$ is given by $(w^T x) w$. Consider the problem of finding the direction $p$ which allows the best possible average reconstruction of the training set, that is the solution of the problem

$$\min_{w \in S^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \|x_i - (w^T x_i) w\|^2,$$

Statistics?
shows that problem (19.1) is called the first principal component of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map which allows the best possible average reconstruction of the original data. It can be derived from several prospective and here we give a geometric/analytical derivation.

In many practical applications it is of interest to reduce the dimensionality of the data. We begin by considering the case where \(k=1\), then the (orthogonal) projection of a point \(x_i\) is given by \(\frac{1}{n} \sum_{i=1}^{n} (w^T x_i)w\) which allows the best possible average reconstruction along the direction \(w\) with norm one.

The norm \(\|x_i\| - (w^T x_i)^2\) relates to the geometric/analytical derivation of PCA.

\[
\min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \|x_i - (w^T x_i)w\|^2,
\]

Statistics?
\[
\min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \|x_i - (w^T x_i)w\|^2,
\]

\[\Rightarrow \max_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} (w^T x_i)^2.\]

Statistics?
The problem (19.1) is called the first principal component of the data. A direct computation measures how much we lose by projecting where $w$ can be seen as the problem of defining a map $S^n ightarrow S^{n-1}$.

In particular, this is useful for data visualization, or for investigating the "eigenstructure" of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map $S^n ightarrow S^{n-1}$.

If the data are centered, that is $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i = 0$, then computing the first principal component of the data reduced to computing the biggest eigenvalue.

Indeed, using the symmetry of the inner product we have

$$\min_{w \in S^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \| x_i - (w^T x_i)w \|^2,$$

which allows the best possible average reconstruction of the data. In fact, it is possible to show that the best possible average reconstruction of the data is given by $\frac{1}{n} \sum_{i=1}^{n} x_i w$, so that problem (19.1) is equivalent to

$$(19.2) \text{max} \frac{1}{n} \sum_{i=1}^{n} (w^T x_i)^2.$$

A simple further manipulation allows to write problem (19.2) as an eigenvalue problem.

We begin by considering the case where $X$ is symmetric and positive semi-definite. If the data are centered the matrix $C = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T$ is symmetric and positive definite. Then computing the first principal component of the data reduced to computing the biggest eigenvalue of $C$.

If $u$ is the solution of the problem (19.1) it is easy to see that this corresponds to considering $\max_{w \in S^{D-1}} \frac{1}{n} \sum_{i=1}^{n} (w^T x_i)^2$. This latter observation is useful for two different reasons that we discuss in the following.

Consider $w$ that maximizes $\frac{1}{n} \sum_{i=1}^{n} (w^T x_i)^2$. The norm $\| x_i - (w^T x_i)w \|^2 = \| x_i \|^2 - (w^T x_i)^2$.

$$\Rightarrow \max_{w \in S^{D-1}} \frac{1}{n} \sum_{i=1}^{n} (w^T x_i)^2.$$

$$\Rightarrow \max_{w \in S^{D-1}} \frac{1}{n} \sum_{i=1}^{n} (w^T (x_i - \bar{x})^2),$$

where the latter quantity is the so-called Rayleigh quotient. Note that, if $w$ is a solution of problem (19.2), then $w^T x_i = \bar{x}^T w$, and $w^T (x_i - \bar{x}) = 0$.

\[ \min_{w \in S^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \| x_i - (w^T x_i)w \|^2, \]
In many practical applications it is of interest to reduce the dimensionality of the data. In particular, this is useful for data visualization, or for investigating the "effective" dimensionality of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map \( M : \mathbb{R}^D \rightarrow \mathbb{R}^k \), \( k \ll D \), according to some suitable criterion.

19.1 PCA & Reconstruction

PCA is arguably the most popular dimensionality reduction procedure. It is a data driven procedure that given an (unsupervised) sample \( S = (x_1, \ldots, x_n) \) derives dimensional reduction defined by a linear map \( M \). PCA can be derived from several prospective and here we give a geometric/analytical derivation.

We begin by considering the case where \( k = 1 \). We are interested in finding the single most relevant dimension according to some suitable criterion. Recall that, if \( w \in \mathbb{R}^D \) with \( k_w = 1 \), then the (orthogonal) projection of a point \( x \) on \( w \) is given by \( (w^T x) w \). Consider the problem of finding the direction \( p \) which allows the best possible average reconstruction of the training set, that is the solution of the problem

\[
\min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \| x_i - (w^T x_i) w \|^2,
\]

\((19.1)\)

where \( \mathbb{S}^D \) is the sphere in \( D \) dimensions. The norm \( k x_i (w^T x_i) w \) measures how much we lose by projecting \( x \) along the direction \( w \), and the solution \( p \) to problem \((19.1)\) is called the first principal component of the data. A direct computation shows that

\[
\| x_i - (w^T x_i) w \|^2 = k x_i k \| w^T x_i \|^2,
\]

so that problem \((19.1)\) is equivalent to

\[
\max_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \| w^T x_i \|^2.
\]

\((19.2)\)

This latter observation is useful for two different reasons that we discuss in the following.

19.2 PCA and Maximum Variance

If the data are centered, that is \( \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i = 0 \), problem \((19.2)\) has the following interpretation: we are looking for the direction along which the data have (on average) maximum variance.

**Computations?**
shows that problem (19.1) is called the first principal component of the data. A direct computation
in particular, this is useful for data visualization, or for investigating the “effective” dimensionality of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map to a single principle component. The idea is simply to iterate the above reasoning to describe the most relevant dimension according to some suitable criterion. Recall that, if $\mathbf{x}$ is the centered sample, that is $\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$, problem (19.1) it is easy to see that this corresponds to considering the problem of finding an eigenvector of the covariance and the corresponding eigenvalue. Rayleigh quotient achieves its maximum at a vector which corresponds to the maximum eigenvalue of the covariance and the corresponding eigenvector. Note that, if $\mathbf{w}$ is an eigenvector of $\mathbf{C}$ with corresponding eigenvalue $\lambda$, then $\lambda = \frac{\mathbf{w}^T \mathbf{C} \mathbf{w}}{\mathbf{w}^T \mathbf{w}}$, or equivalently $\lambda = \frac{\mathbf{w}^T \mathbf{C} \mathbf{w}}{\mathbf{w}^T \mathbf{w}}$. Indeed, using the symmetry of the inner product we have $\lambda = \frac{\mathbf{w}^T \mathbf{C} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{w}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{w} = \mathbf{w}^T (\frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T) \mathbf{w}$.

**Computations?**

$$\min_{\mathbf{w} \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \| \mathbf{x}_i - (\mathbf{w}^T \mathbf{x}_i) \mathbf{w} \|^2,$$

$$\max_{\mathbf{w} \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i)^2.$$  
$$\Leftrightarrow \max_{\mathbf{w} \in \mathbb{S}^{D-1}} \mathbf{w}^T \mathbf{C}_n \mathbf{w}, \quad \mathbf{C}_n = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$$

$$\frac{1}{n} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i)^2 = \frac{1}{n} \sum_{i=1}^{n} \mathbf{w}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{w} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{w}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{w} = \mathbf{w}^T (\frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T) \mathbf{w}$$
Dimensionality Reduction

\[ M : X = \mathbb{R}^D \rightarrow \mathbb{R}^k, \quad k \ll D, \]

What about \( k = 2? \)

\[ \text{\( w_2 \) second eigenvector of } C_n \]

\[
\max_{\substack{w \in \mathbb{S}^{D-1} \\
w \perp w_1}} w^T C_n w, \quad C_n = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T.
\]
In many practical applications it is of interest to reduce the dimensionality of the data. In particular, this is useful for data visualization, or for investigating the "effective" dimensionality of the data. This problem is often referred to as dimensionality reduction and can be seen as the problem of defining a map \( M : X = \mathbb{R}^D \rightarrow \mathbb{R}^k, \ k \ll D, \)

things I won't tell you about

- **Random** Maps: Johnson-Linderstrauss Lemma
- **Non Linear** Maps: Kernel PCA, Laplacian/ Diffusion maps
End of PART III b)

- a) Variable Selection: OMP
- b) Dimensionality Reduction: PCA

Interpretability - Sparsity - Greedy & Convex Relaxation Approaches
The End

PART IV

• Matlab practical session

Afternoon