Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary

Pre-Processing
Problem statement
Offset
Scaling

Seminar in Data Science

Lecture 8: Kernel Methods; Data Normalization

Laurent El Ghaoui

Seminar in Data Science and Information Technology, Summer 2020
TBSI – UC Berkeley

7/29/2020

Outline

Kernels
- Motivations
- Kernel trick
- Examples
- Kernels and sparsity
- Summary

Pre-Processing
- Problem statement
- Offset
- Scaling

Outline

Kernels
- Motivations
- Kernel trick
- Examples
- Kernels and sparsity
- Summary

Pre-Processing
- Problem statement
- Offset
- Scaling

Motivation

A linear regression problem

Linear (second-order) auto-regressive model for time-series: y_t linear function of y_{t-1}, y_{t-2}

$$y_t = w_1 + w_2 y_{t-1} + w_3 y_{t-2}, \quad t = 3, \ldots, T.$$

This writes $y_t = w^T x_t$, with x_t the "feature vectors"

$$x_t := (1, y_{t-1}, y_{t-2}), \quad t = 3, \ldots, T.$$

Define $3 \times (T-2)$ matrix X with t-th column x_t, and $(T-2) \times 1$ vector $y = (y_3, \ldots, y_T)$.

Model fitting via least-squares:

$$\min_w \|X^T w - y\|_2^2$$

Prediction rule : at time $T+1$, $\hat{y}_{T+1} = w_1 + w_2 y_T + w_3 y_{T-1} = w^T x_{T+1}$.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Nonlinear regression

Nonlinear auto-regressive model for time-series: y_t *quadratic* function of y_{t-1}, y_{t-2}

$$y_t = w_1 + w_2 y_{t-1} + w_3 y_{t-2} + w_4 y_{t-1}^2 + w_5 y_{t-1} y_{t-2} + w_6 y_{t-2}^2.$$

This writes $y_t = w^T \phi(x_t)$, with $\phi(x_t)$ the augmented feature vectors

$$\phi(x_t) := \left(1, y_{t-1}, y_{t-2}, y_{t-1}^2, y_{t-1} y_{t-2}, y_{t-2}^2\right).$$

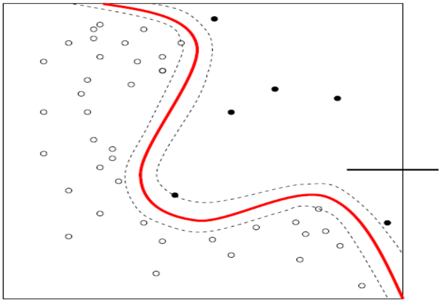
Prediction rule is $\hat{y}_{T+1} = w^T \phi(x_{T+1})$.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary

Pre-Processing
Problem statement
Offset
Scaling

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Nonlinear classification



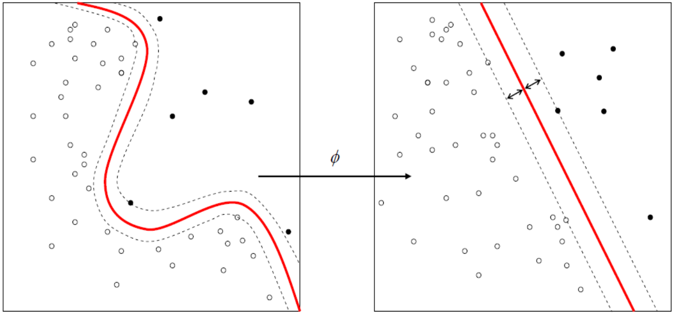
Non-linear (*e.g.*, quadratic) decision boundary

$$w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_1 x_2 + w_5 x_2^2 + b = 0.$$

Writes $w^T \phi(x) + b = 0$, with $\phi(x) := (x_1, x_2, x_1^2, x_1 x_2, x_2^2)$.

Challenges

In principle, it seems we can always augment the dimension of the feature space to make the data linearly separable. (See the video at http://www.youtube.com/watch?v=3liCbRZPrZA)



How do we do it in a computationally efficient manner?

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Generic form of regularized learning problem

Many classification and regression problems can be written

$$\min_{w} \mathcal{L}(X^T w, y) + \lambda \|w\|_2^2$$

where

- $X = [x_1, \ldots, x_m]$ is a $n \times m$ matrix of data points.
- $y \in \mathbf{R}^m$ contains a response vector (or labels).
- $w \in \mathbf{R}^n$ contains classifier or regression coefficients.
- $\mathcal{L}$ is a “loss” function that depends on the problem considered.
- $\lambda \geq 0$ is a regularization parameter.

Prediction/classification rule: depends only on $w^T x$, where $x \in \mathbf{R}^p$ is a new data point.

Note: here, we consider l_2-norm penalty only!

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Loss functions

- Squared loss: (for linear least-squares regression)

$$\mathcal{L}(z,y) = \|z-y\|_2^2 = \sum_{i=1}^{n}(x_i - y_i)^2.$$

- Hinge loss: (for SVMs)

$$\mathcal{L}(z,y) = \sum_{i=1}^{m} \max(0, 1 - y_i z_i)$$

- Logistic loss: (for logistic regression)

$$\mathcal{L}(z,y) = -\sum_{i=1}^{m} \log(1 + e^{-y_i z_i}).$$

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Key result

For the generic problem:

$$\min_{w} \mathcal{L}(X^T w, y) + \lambda \|w\|_2^2$$

the optimal w lies in the span of the data points $(x_1, \ldots, x_m)$:

$$w = Xv$$

for some vector $v \in \mathbf{R}^m$.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Proof

A linear algebra result

For any matrix $X \in \mathbf{R}^{n \times m}$: every $w \in \mathbf{R}^n$ can be written as the sum of two *orthogonal* vectors, one in the range of X and the other orthogonal to it:

$$w = Xv + r$$

where $v \in \mathbf{R}^m$, and $X^T r = 0$ (that is, r is in the nullspace $\mathcal{N}(X^T)$).

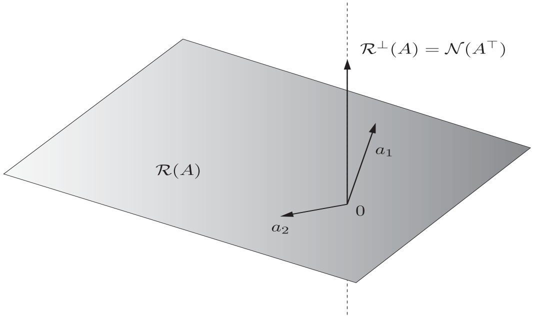


Figure shows the case $X = A = (a_1, a_2)$.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Consequence of key result

For the generic problem:

$$\min_{w} \mathcal{L}(X^T w, y) + \lambda \|w\|_2^2$$

the optimal w can be written as $w = Xv$ for some vector $v \in \mathbf{R}^m$.

Hence training problem depends only on the $m \times m$ (PSD) matrix $K := X^T X$:

$$\min_{v} \mathcal{L}(Kv, y) + \lambda v^T K v.$$

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Kernel matrix

The training problem depends only on the "kernel matrix" $K = X^T X$

$$K_{ij} = x_i^T x_j, \;\; 1 \leq i, j \leq n.$$

That is, K contains the scalar products between all data point pairs.

The prediction/classification rule depends on the scalar products between new point x and the training data points $x_1, \ldots, x_m$:

$$w^T x = v^T X^T x = v^T k, \;\; k := X^T x = (x^T x_1, \ldots, x^T x_m).$$

Computational advantage: Once K is formed (this takes $O(n^2 p)$), then the training problem has only n variables. When $p >> n$, this leads to a dramatic reduction in problem size.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

How about the nonlinear case?

In the nonlinear case, we simply replace the feature vectors x_i by some "augmented" feature vectors $\phi(x_i)$, with ϕ a non-linear mapping.

Example : in classification with quadratic decision boundary, we use

$$\phi(x) := (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2).$$

This leads to the modified kernel matrix

$$K_{ij} = \phi(x_i)^T \phi(x_j), \quad 1 \leq i, j \leq m.$$

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

The kernel function

The *kernel function* associated with mapping ϕ is

$$k(x, z) = \phi(x)^T \phi(z).$$

It provides information about the metric in the feature space, *e.g.*:

$$\|\phi(x) - \phi(z)\|_2^2 = k(x, x) - 2k(x, z) + k(z, z).$$

The computational effort involved in

- solving the training problem;
- making a prediction,

depends only on our ability to quickly evaluate such scalar products.

We can't choose k arbitrarily; it has to satisfy the above for some ϕ.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Quadratic kernels

Classification with quadratic boundaries involves feature vectors

$$\phi(x) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2).$$

Fact : given two vectors $x, z \in \mathbf{R}^2$, we have

$$\phi(x)^T \phi(z) = (1 + x^T z)^2.$$

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary

Pre-Processing
Problem statement
Offset
Scaling

Polynomial kernels

More generally when $\phi(x)$ is the vector formed with all the products between the components of $x \in \mathbf{R}^n$, up to degree d, then for any two vectors $x, z \in \mathbf{R}^n$,

$$\phi(x)^T \phi(z) = (1 + x^T z)^d.$$

Computational effort grows linearly in n.

This represents a dramatic reduction in speed over the “brute force” approach:

- Form $\phi(x)$, $\phi(z)$;
- evaluate $\phi(x)^T \phi(z)$.

Computational effort grows as n^d.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Other kernels

Gaussian kernel function:

$$k(x, z) = \exp\left(-\frac{\|x - z\|_2^2}{2\sigma^2}\right),$$

where $\sigma > 0$ is a scale parameter. Allows to ignore points that are too far apart. Corresponds to a non-linear mapping ϕ to infinite-dimensional feature space.

There is a large variety (a zoo?) of other kernels, some adapted to structure of data (text, images, etc).

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary

Pre-Processing
Problem statement
Offset
Scaling

Kernels and sparsity

Kernel methods essentially lift the space of features, to pose the original problem in higher-dimensional space.

Sparse methods seek to improve the interpretability of the results, by selecting a few features that are relevant to the prediction problem at hand.

Unfortunately kernels and sparsity don't mix well, since we loose the interpretation of the original features in the "lifting" process.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary

Pre-Processing
Problem statement
Offset
Scaling

Sparse methods: basic idea

In some cases we are interested in the *interpretability* of the regression parameters:

- Which few features are responsible for most of the predictive power?
- Can we find those features in a reliable way?

Sparse methods attempt to trade-off sparsity of the optimal regression or classification coefficients with predictive power.

In such methods, the l_1-norm plays an important role.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

A justification from regularized LS

Consider the *cardinality-constrained* regularized LS:

$$\min_w \|X^T w - y\|_2^2 + \lambda \|w\|_2^2 \ : \ \mathbf{Card}(w) = k$$

where $\lambda > 0$ is the regularization parameter, $\mathbf{Card}(w)$ denotes cardinality of w (number of nonzero elements in w), and k imposes a bound on it.

Problem above is very difficult to solve. We can approximate it as

$$\min_w \|X^T w - y\|_2^2 + \frac{\lambda}{k} \|w\|_1^2.$$

Proof: Cauchy-Schwartz inequality implies

$$\sqrt{\mathbf{Card}(w)} \cdot \|w\|_2 \geq \|w\|_1.$$

Relaxation
The LASSO

When λ varies, the set of solutions is the same as that of the "LASSO" problem

$$\min_{w} \|X^T w - y\|_2^2 + \mu \|w\|_1 ,$$

where $\mu > 0$ spans the positive real line.

- Not solvable by direct linear algebra methods.
- Many efficient algorithms have bee proposed for this type of convex problem
- When the number of variables is not too high, *coordinate descent* (solving the problem one variable at a time, all the others being fixed) works well.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Example: image of Microsoft in NYT headlines

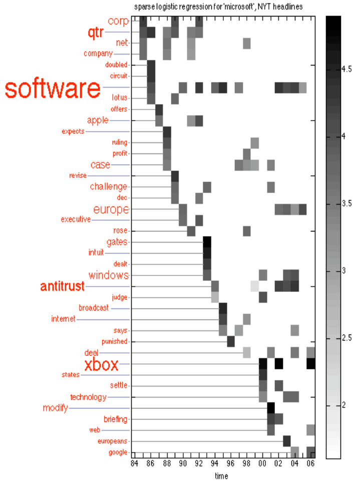


Image of `microsoft` in NYT headlines (1981-2007).

- Data matrix X records occurrences of words in documents.
- Response ± 1 vector y records presence/absence of term `microsoft`.
- Solve LASSO to regress occurrence of Microsoft against other word occurrences.
- Sparsity allows to find just a few words whose occurrence predicts that of the term “Microsoft” in headlines.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Can LASSO be kernelized?

Answer : NO.

This is due to the fact that the solution to the problem

$$\min_w \|X^T w - y\|_2^2 + \mu \|w\|_1,$$

does not depend only on $K := X^T X$, but on X itself. This is in contrast with the Ridge regression problem

$$\min_w \|X^T w - y\|_2^2 + \mu \|w\|_2^2.$$

The main reason is the lack of rotational invariance of the l_1-norm.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Kernel methods: summary

- Kernels need to be chosen by the user.
- Choice not always obvious; Gaussian or polynomial kernels are popular.
- Control over-fitting via cross validation (wrt say, scale parameter of Gaussian kernel, or degree of polynomial kernel).
- Kernel methods not well adapted to sparsity / l_1-norm regularization (seen next).

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Outline

Kernels
- Motivations
- Kernel trick
- Examples
- Kernels and sparsity
- Summary

Pre-Processing
- Problem statement
- Offset
- Scaling

Pre-processing
Problem statement

Before we start processing data via a supervised learning algorithm, we need to prepare data:

- Remove missing values;
- Remove outlier values;
- Normalize (offset and rescale) data points;
- Normalize features.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Offset: the LS case

Consider the penalized least-squares problem:

$$\min_{w} \|X^T w - y\|_2^2 + \rho^2 \|w\|_2^2. \tag{1}$$

The prediction rule for a new data point x is $\hat{y}(x) = w^T x$.

It is common to center the $n \times m$ data matrix $X = [x_1, \ldots, x_m]$ and response y, and modify it to $X^c := [x_1 - \bar{x}, \ldots, x_m - \bar{x}]$, $y^c = y - \bar{y}$ with

$$\bar{x} = \frac{1}{m} \sum_{i=1}^{m} x_i, \quad \bar{y} = \frac{1}{m} \sum_{i=1}^{m} y_i.$$

This is equivalent to solving the problem

$$\min_{w,b} \|X^T w + b\mathbf{1} - y\|_2^2 + \rho^2 \|w\|_2^2,$$

since the optimal b is $b = \bar{y} - \bar{x}^T w$.

The prediction rule is now $\hat{y}(x) = w^T x + b$.

In effect, for the LS case, centering implicitly allows to consider a more general *affine* rule (as opposed to linear, as in problem (1)).

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Offset: dealing with other loss functions

When other loss functions are involved, the previous centering method does not have a natural interpretation as an affine rule.

It is thus *always* better to work with affine rules (arguably more general than linear ones):

$$\min_{w,b} \mathcal{L}(X^T w + b\mathbf{1}, y) + p(w)$$

Such an approach provides an implicit "centering".

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Scaling features

Sometimes, scaling the features provides better performance. Just as with centering, scaling should be done differently with different loss functions.

- The standard approach is to center, and then normalize data by variance: $X \to D^{-1}(X - \hat{x}\mathbf{1}^T)$, where $D = \mathbf{diag}(\sigma_1, \dots, \sigma_n)$.
- The problem becomes

  $$\min_{w,b} \mathcal{L}((X - \hat{x}\mathbf{1}^T)^T D^{-1} w, y) + p(w) = \mathcal{L}(X^T D^{-1} w - \mathbf{1}(\hat{x}^T D^{-1} w), y) + p(w)$$

  which is equivalent, under the change of variable $w \to D^{-1}w$, to

  $$\min_{w,b} \; \mathcal{L}(X^T w + b\mathbf{1}, y) + p(Dw).$$

- One may use robust methods, where mean and standard deviation are replaced with median and mean absolute deviation.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary
Pre-Processing
Problem statement
Offset
Scaling

Scaling data points

In practice, it may be advisable to scale data points differently.

- In time-series prediction, we may use exponential smoothing in order to give more importance to recent time-series values than older ones.
- In imbalanced classification, we may need to scale the negatively labelled data differently from the positively labelled data.

Kernels
Motivations
Kernel trick
Examples
Kernels and sparsity
Summary

Pre-Processing
Problem statement
Offset
Scaling