Machine Learning

Model Selection

Mirco Mutti
Credits to Francesco Trovò

Definition of different models

What to do in the case the model you are considering is not performing well even by tuning properly the parameters (cross-validation)?

We have two opposite options:

- simplify the model (model selection)
- increase its complexity (next time)

Definition of different models

What to do in the case the model you are considering is not performing well even by tuning properly the parameters (cross-validation)?

We have two opposite options:

- simplify the model (model selection)
- increase its complexity (next time)

How to Select a Model

We already discussed how to evaluate a specific model (bias/variance dilemma)

- Model Selection
  - Feature selection: choose only a subset of significant features to use
  - Regularization (shrinkage): introduce some penalization for complex models in the loss function
  - Dimensionality reduction: project the features in a lower dimensional space
- Ensemble model
  - Bagging
  - Boosting

Model Selection

Model Selection

- Feature Selection
  - Filter methods
  - Embedded FS
  - Wrapper methods
    - Brute Force
    - Forward Step-wise Selection
    - **Backward Step-wise Selection**
- Feature Extraction
  - **PCA**
  - ICA
- Regularization
  - LASSO
  - Ridge
  - ...

Feature Selection: Brute Force

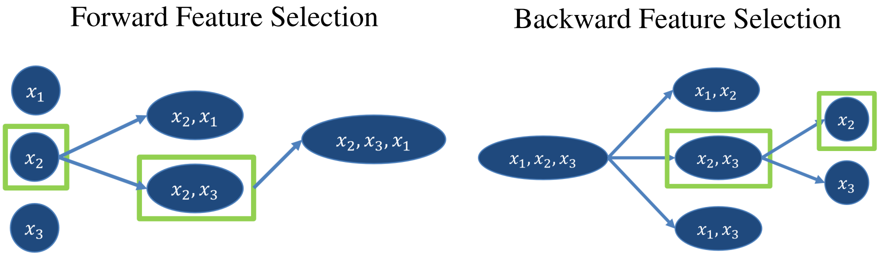
In principle one can do:

- For each feature x_k with $k \in \{1, \ldots, M\}$
  - Learn all the possible $\binom{M}{k}$ possible models with k inputs
  - Select the model with the smallest loss
- Select the k providing the model with the smallest loss

Problem: if M is large enough the computation of all the models is unfeasible

Wrapper Methods

We evaluate only a subset of the possible models



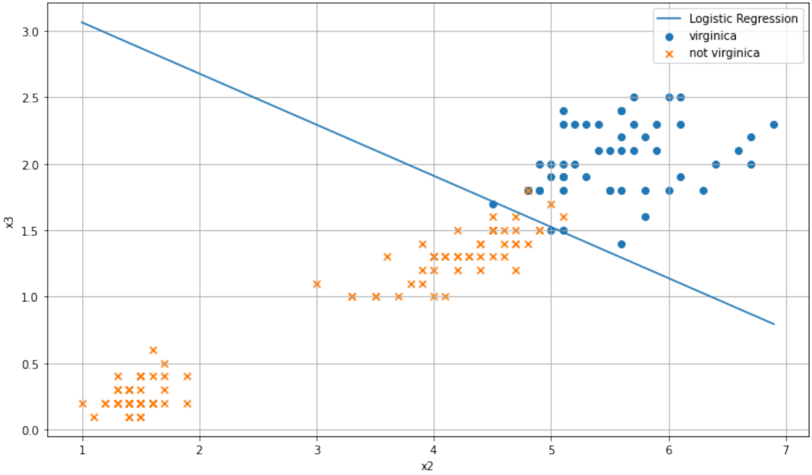
Backward Feature selection on the Iris Dataset (1)

- Assume the problem is to discriminate between Virginica and Non-Virginica iris
- We select a performance metric: validation accuracy on 20% of the data
- Train a model on the full data (x_1, x_2, x_3, x_4): Logistic regression
- Remove one of the features and check the error:
  - Model with (x_1, x_2, x_3): accuracy 1
  - Model with (x_1, x_3, x_4): accuracy 1
  - Model with (x_1, x_2, x_4): accuracy 1
  - Model with (x_2, x_3, x_4): accuracy 1
- Removing a single feature does not change the method performance

Backward Feature selection on the Iris Dataset (2)

- Let us remove one of the features at random x_4
- Remove another feature and check the error:
  - Model with (x_1, x_2): accuracy 0.96
  - Model with (x_1, x_3): accuracy 0.96
  - Model with (x_2, x_3): accuracy 1
- The model with (x_2, x_3) is performing better than the others
- Iterate *one more time*

Results on the Iris Dataset



Principal Component Analysis

Idea

Principal Component Analysis (PCA) is an unsupervised dimensionality reduction technique, i.e., which extract some low dimensional features from a dataset

We would like to perform a linear transformation of the original data X s.t. the largest variance lies on the first transformed feature, the second largest variance on the second transformed feature, ...

At last we only keep some of the features we extract

Procedure

- Translate the original data X to $\tilde{X}$ s.t. they have zero mean
- Compute the covariance matrix of $\tilde{X}$, $C = \tilde{X}^T \tilde{X}$
- The eigenvector e_1 corresponding to the largest eigenvalue λ_1 is the first principal component
- The eigenvector e_2 corresponding to the second largest eigenvalue λ_2 is the second principal component
- ...

Given a sample vector $\tilde{\mathbf{x}}$, its transformed version $\mathbf{t}$ can be computed using:

$$\mathbf{t} = \tilde{\mathbf{x}}\, W$$

where t_i is the i-th principal component

- loadings: W matrix of the weights
- scores: T transformation of the input dataset $\tilde{X}$
- variance: $(\lambda_1, \ldots, \lambda_M)$ vector of the variance of principal components

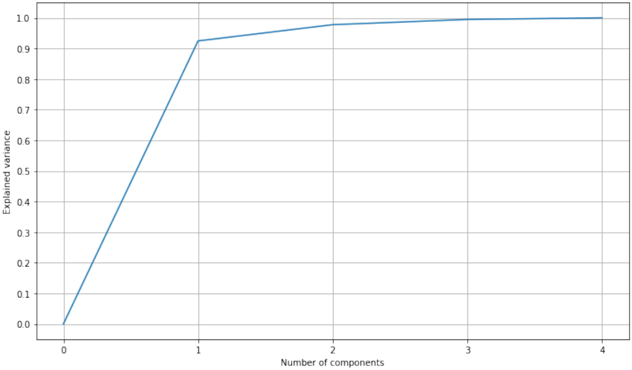
How Many Features

There are a few different methods to determine how many feature to choose

- Keep all the principal components until we have a cumulative variance of 90%-95%
- Keep all the principal components which have more than 5% of variance (discard only those which have low variance)
- Find the elbow in the cumulative variance

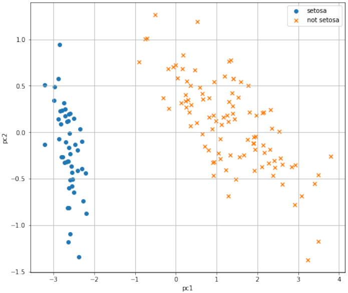
Cumulated Variance Plot

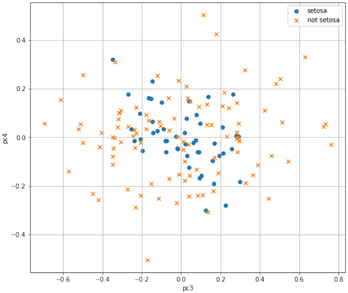
Using the Iris dataset inputs:



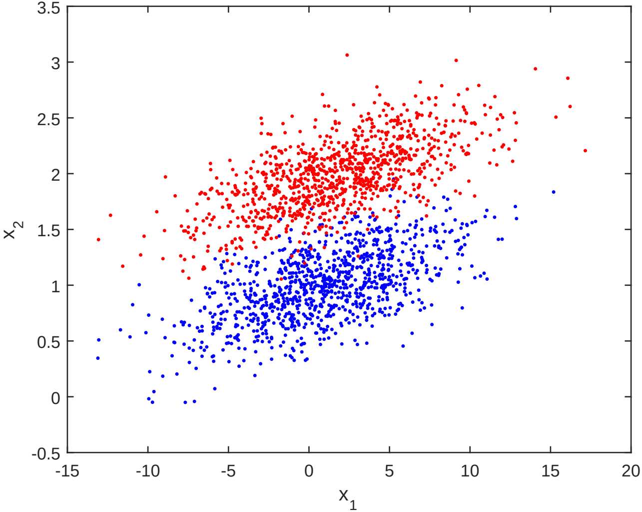
Principal Components

If we separate the first two components from the second twos:





Simpson's Paradox



PCA Different Purposes

- Feature Extraction: reduce the dimensionality of the dataset by selecting only the number of principal components retaining information about the problem
- Compression: the linear transformation W minimizes among the ones with k dimension $\min_{W_{red}} ||\tilde{X}(k)W_{red} - \tilde{X}||_2^2$, i.e., is the linear transformation minimizing the reconstruction error
- Data visualization: reduce the dimensionality of the input dataset to 2 or 3 to be able to visualize the data

Regularization

Regularization

Already known regularization procedure:

- Ridge:

$$L(\mathbf{w}) = \frac{1}{2}RSS(\mathbf{w}) + \frac{\lambda}{2}||\mathbf{w}||_2^2$$

- Lasso:

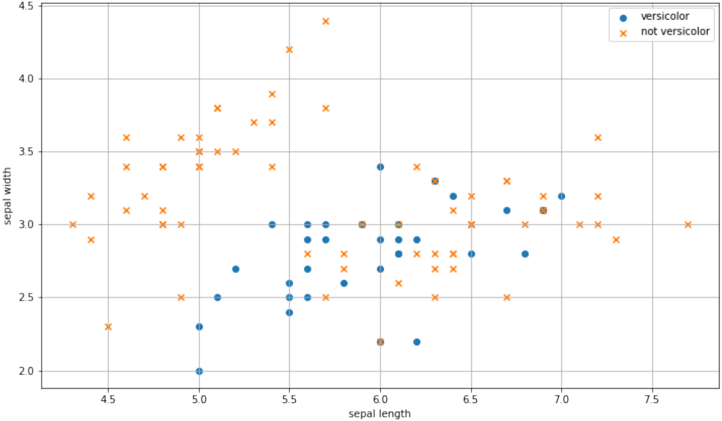
$$L(\mathbf{w}) = \frac{1}{2}RSS(\mathbf{w}) + \frac{\lambda}{2}||\mathbf{w}||_1$$

- Elastic net:

$$L(\mathbf{w}) = \frac{1}{2}RSS(\mathbf{w}) + \frac{\lambda_1}{2}||\mathbf{w}||_2^2 + \frac{\lambda_2}{2}||\mathbf{w}||_1$$

- They can be applied to the linear regression technique, it can be extended for other methods
- For classification we will see some specific methods

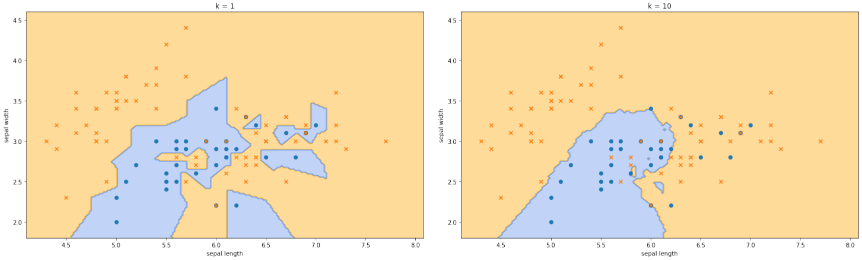
A hard problem



This problem cannot be solved with a linear classification technique

K-Nearest Neighbour

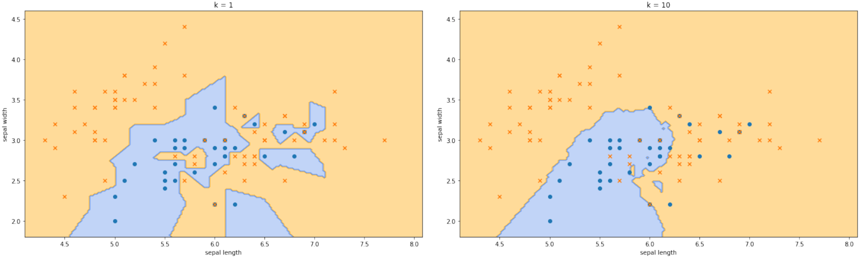
Different values of the K parameter



The larger the value of K, the more the model is regularized ($1/K$ acts as a regularization hyperparameter)

K-Nearest Neighbour

Different values of the K parameter



The larger the value of K, the more the model is regularized ($1/K$ acts as a regularization hyperparameter)