Machine learning

Hugo Maruri Aguilar - Kostas Papafitsoros

11/04/2024

Important concepts (from the notes)

Preliminaries

Eigenvalues and eigenvectors of a **square** matrix; the reproducing property $\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i$ and the decomposition $\mathbf{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{-1}$.

The **Karhunen-Loeve** (spectral) decomposition of a square symmetric matrix $\mathbf{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T = \sum_{i=1}^{p} \lambda_i \mathbf{v}_i \mathbf{v}_i^T$. The **singular value** decomposition $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \sum_{i=1}^{\min(n,p)} d_i \mathbf{u}_i \mathbf{v}_i^T$ of a matrix $\mathbf{X}$.

At a concept level and operative level, understand and be able to calculate **derivatives** of linear and quadratic forms. The concept of **Lagrange** multipliers.

`R`: Machine learning without data manipulation is an oxymoron

Using whichever version of `R` you feel comfortable with (`R` GUI, `RStudio`, online `R` "snippets", command version), you need to be familiar with data loading and manipulation. Basic functions for the preliminaries are `var`, `scale`, `apply`; `eigen` and `svd`.

Machine learning

The taxonomy of Machine Learning. Within it, different types of learning and problems within types of learning.

Unsupervised learning

Understand the sample variance-covariance matrix of a centered data set $\boldsymbol{\Sigma} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X}$ and be able to compute it either by hand in small example or numerically using software.

Principal component analysis

Understand and compute numerically Principal Components Analysis in its two versions: as Karhunen-Loeve decomposition of $\boldsymbol{\Sigma}$ and using the Singular value decomposition of $\mathbf{X}$. That is, become familiar with the following two factorisations

$$\boldsymbol{\Sigma} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X} = \underbrace{\mathbf{A}\boldsymbol{\Lambda}\mathbf{A}^T}_{\mathbf{1}\text{: KL-based PCA}} = \underbrace{\mathbf{V}\left(\frac{1}{n-1}\mathbf{D}^2\right)\mathbf{V}^T}_{\mathbf{2}\text{: SVD-based PCA}}.$$

On the operative side, use **3**: `prcomp` and process its output.

PC analysis has three elements: variances, loadings and scores. Become familiar with them, identify them and understand how to compute them; determine number of components to select; interpret PC loadings, use scatterplots of PC scores and the biplot.

Compute and interpret PC loadings $\mathbf{A} = \mathbf{V}$ and PC scores $\mathbf{XA} = \mathbf{UD}$.

Understand the difference in PCA when data has/not been centered and when data has/not been scaled.

Repeat all these `prcomp` results with `svd` and `eigen`.

Cluster analysis

Become familiar with the procedure of **agglomerative** clustering. Become familiar with one general method and there is not need to memorize e.g. nine apparently different methods (three distances and three linkages).

Become familiar with the K-means algorithm in its variants: **K-means** and **K-medoids**. In order to suggest a number of clusters, become familiar with scatterplots and auxiliary plots of ESS and the **silhouette**.

Use `agnes` from library `cluster` to do agglomerative clustering; also `pam` and `kmeans` to do K-means clustering of data. Plot and interpret results.

Supervised learning

Become familiar with the concept of **splitting** the data in training and testing for evaluation of a model. Ditto the concept of k-fold cross-validation and leave-one-out cross-validation. Use the function `cvFolds`.

Classification

The concept of **classification** with **binary** response (positives, negatives) and the mistakes **misclassifications** that may occur. Definition of the confusion matrix and computations with it for a given data set, including the figures TN, FP, FN, TP, P, N.

The summary measures TPR and FPR, aka as **sensitivity** and (1-)**specificity**. Computations of summary measures with a given data set, thresholding and plotting them and interpreting them in the **ROC graph**. Compute and interpret the performance of a classifier using **ROC curve** and the **AUC** where relevant.

Become familiar with classifiers: **linear**, **logistic** and **KNN**; also classification **tree** and linear **discriminant** analysis.

`R` functions for comparisons of classifiers are `table`, `roc` and `auc`; for building classifiers use `lm`, `glm`, `predict` and `knn`; `tree` and `lda`.

Regularization

Understand the concept of **penalising** a regression by the coefficient size as posing a **dual objetive** problem. In this setting, understand the elastic net penalization and its relation with the lasso and ridge penalizations. Also understand the concept of penalized likelihood as the general instance of regularization.

For both the ridge trace and the lasso path, understand the different versions the can be built and the computations involved in each case. Recognize the differences between ridge regression, lasso and elastic net.

Become familiar about recovering coefficients in the path in each of **lasso**, **ridge** and **elastic net**.

Important functions for this analysis are `lars`, `ridge` (in house) and `glmnet`, as well as the `cv.-` and `coef.-` utileries of them to recover path coefficients and doing cross-validation.