

Statistical Machine Learning

Lecture 1

Statistical Setup

Horacio Gómez-Acevedo
Department of Biomedical Informatics

January 29, 2022



Introduction

We will fill the technical gaps of the Chapter 2 from Géron's book. The objective is: Your model should learn from the provided data and be able to predict the median housing price in any district, given all the other metrics.

Rough methodology:

- ▶ It is a supervised learning model
- ▶ Regression task
- ▶ There is no continuous data flow
- ▶ Sample size is small enough to handle it locally, thus a **plain batch training** is appropriate.

Formal Setup

The book uses the following description for the data setup. Note however that at this point, the data is not a matrix yet (entries of features are not necessarily real numbers), thus we will call it a **data frame**.

$$\mathbf{X} = \begin{matrix} & \textit{feature 1} & \textit{feature 2} & \cdots & \textit{feature } n \\ \begin{pmatrix} \mathbf{x}^{(1)} \end{pmatrix}^t \\ \begin{pmatrix} \mathbf{x}^{(2)} \end{pmatrix}^t \\ \vdots \\ \begin{pmatrix} \mathbf{x}^{(m)} \end{pmatrix}^t \end{matrix} \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

and the response variable as

$$\mathbf{Y} = (y^1, y^2, \dots, y^m)$$

Matrix representation

Once we have transformed our data frame into a real matrix \mathbf{X} of size $m \times n$ (m observations and n numerical features or predictors), the regression problem reduces to the representation of the response vector Y as a function (or hypothesis) h as follows

$$Y = h(X) + \varepsilon \quad (1)$$

where ε is a random error term.

Important points

- ▶ We do not know h beforehand
- ▶ It is customary to consider that the error rate averages zero
- ▶ Prediction of Y can be done by using

$$\hat{Y} = \hat{h}(X)$$

where \hat{h} is the estimate for h , and \hat{Y} represents the resulting prediction of Y .

Performance measure

In a typical statistical framework, we measure the quality of fit to assess how well our approximation "fits" the data. Residuals are normally used to determine how well our model fits the data by the so-called **Residual Standard Error (RSE)**

$$RSE = \sqrt{\frac{1}{m-2} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2} \quad (2)$$

A small variation of this formula is called **Root Mean Squared Error (RMSE)**

$$RMSE = \sqrt{\frac{1}{m} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2} \quad (3)$$

RMSE or (MSE) are normally used to determine the "performance" of our models.

Norms in \mathbb{R}^n

So far the model has used the norm (distance) between two vectors as the Euclidean norm (ℓ_2) as

$$\ell_2: \|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$$

There are more "norms" (distances) that we can use

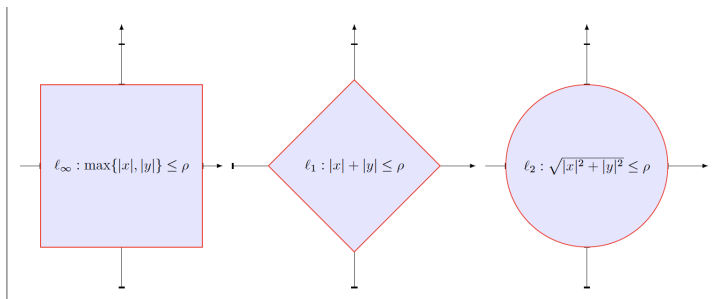
$$\ell_1: \|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$$

$$\ell_p: \|\mathbf{x}\|_p = \sqrt[p]{\sum_{i=1}^n |x_i|^p}$$

$$\ell_\infty: \|\mathbf{x}\|_\infty = \max_{i=1,\dots,n} |x_i|$$

Norms in the plane

When we consider a distance (norm), we want to visualize points that are close to zero up to a distance ρ . These sets are called (closed) balls of radius ρ .



These shapes will be important when we use Lasso techniques.

Creating a Test Dataset

After we downloaded the data and roughly observed their distribution (through histograms). We proceed with a very important steps in any machine learning algorithm.

- ▶ Create a training set (typically around 80% of the whole dataset)
- ▶ The rest will be the test set.
- ▶ **Never ever peak at the test set.** This is a good scientific practice.
- ▶ Use stratification and randomization to get proper representatives of the sample structure.

Expectation

Informally: Suppose W represents the outcome of an experiment (say tossing a die). The **expectation of W** (denoted by $E(W)$) can be perceived as the arithmetic mean of the outcome of W when repeating the experiment many times.

Formally: If W is a random variable, then

- ▶ W is discrete

$$E(W) = \sum_{\alpha} \alpha P(W = \alpha)$$

- ▶ W is continuous with probability density function $f(w)$

$$E(W) = \int_{-\infty}^{\infty} sf(s)ds$$

Example

Suppose W is the outcome when we roll a die. What is $E(W)$?

Solution. Since the die is fair

$$p(W = 1) = p(W = 2) = \cdots = p(W = 6) = \frac{1}{6}$$

$$\begin{aligned} E(W) &= 1 \left(\frac{1}{6} \right) + 2 \left(\frac{1}{6} \right) + 3 \left(\frac{1}{6} \right) + 4 \left(\frac{1}{6} \right) + 5 \left(\frac{1}{6} \right) + 6 \left(\frac{1}{6} \right) \\ &= \frac{7}{2} \end{aligned}$$

Note. Not all random variables have expected value. For instance if W is Cauchy distributed with parameters $\alpha = 0$ and $\beta = 1$.

Expectation properties

Let's suppose that W and Z have expectations

- ▶ Linearity: $E(aW + bZ) = aE(W) + bE(Z)$ where $a, b \in \mathbb{R}$
- ▶ Independence condition: If W and Z are (statistically) independent $E(WZ) = E(W)E(Z)$.

If the expectation of $E(W^2)$ exists, we define the **variance** of W as

$$\text{Var}(W) = E(W^2) - E(W)^2$$

If the variance of Z also exists, we define the **covariance** of W and Z as

$$\text{Cov}(W, Z) = E(WZ) - E(W)E(Z)$$

Variance and Covariance Properties

- ▶ Since $\text{Var}(W) \geq 0$, we define the **standard deviation** of W as $\sigma(W) = \sqrt{\text{Var}(W)}$
- ▶ $\text{Var}(W + Z) = \text{Var}(W) + \text{Var}(Z)$
- ▶ $\text{Var}(aW) = a^2 \text{Var}(W)$ for $a \in \mathbb{R}$
- ▶ If W and Z are (statistically) independent $\text{Cov}(W, Z) = 0$.
- ▶ $\text{Cov}(W + a, Z + b) = \text{Cov}(W, Z)$

If W is a random variable with existing variance, the **standardize form of** W is the variable \tilde{W} defined by

$$\tilde{W} = \frac{W - E(W)}{\sigma(W)}$$

It can be shown that

$$E(\tilde{W}) = 0 \quad \text{and} \quad \text{Var}(\tilde{W}) = 1$$

Variance Covariance Matrix

Let's go back and consider X as a matrix of size $m \times p$. Thus, $X = (X_1, X_2, \dots, X_p)$ where each X_i is a column vector. We can calculate the so-called Variance-Covariance matrix

$$\begin{pmatrix} \text{Var}(X_1) & \text{Cov}(X_1, X_2) & \cdots & \text{Cov}(X_1, X_p) \\ \text{Cov}(X_2, X_1) & \text{Var}(X_2) & \cdots & \text{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_p, X_1) & \text{Cov}(X_p, X_2) & \cdots & \text{Var}(X_p) \end{pmatrix}$$

- ▶ All entries in the matrix are **estimates** of the true variance and covariance.
- ▶ The matrix is symmetric.

Pearson's Correlation Coefficient

If W and Z are random variables with existing variance and if neither $\sigma(W)$ nor $\sigma(Z)$ are zero. The **Pearson's correlation coefficient** ρ between W and Z is defined as

$$\rho(W, Z) = \frac{\text{Cov}(W, Z)}{\sigma(W)\sigma(Z)}$$

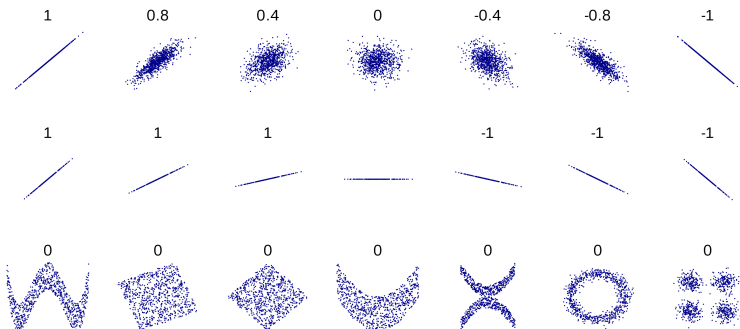
If W and Z are random variables with existing correlation coefficient, then

$$-1 \leq \rho(W, Z) \leq 1$$

When $\rho(W, Z) > 0$ we say that W and Z are positively correlated. And similarly when $\rho(W, Z) < 0$ we called the variables negatively correlated.

Correlations (Important Notes)

- ▶ Even when the variables W and Z are perfectly correlated (i.e., $\rho(W, Z) = \pm 1$) this statement does not mean anything about *causation*.
- ▶ Neither is true that $\rho(W, Z) = 0$ implies (statistical) independence of W and Z .
- ▶ Correlations do not catch non-linear relationships.



The problem of collinearity

Highly correlated attributes (features) represent a problem for finding an appropriate model for our data.

The simplest way to do it (even not the best) is to select one of the highly correlated variable.

There are more robust techniques to deal with collinearity (at least from the linear regression setup)

- ▶ Regression on principal components
- ▶ Ridge regression

Data Imputation

When we face incomplete information or missing data our statistical tools do not normally work (or work poorly). One remedy is "guess" the best possible value is a process called **Data Imputation**.

Let's consider a simple multilinear regression model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon$$

One simple method is the *case-wise deletion*, which means that if we have a missing value in any of our variables we drop that data point. This is not a good alternative since it can introduce bias in the calculations of β_i . It also triggers the following vicious cycle: sample size reduction \Rightarrow standard errors increase \Rightarrow confidence intervals widen \Rightarrow test of the lack of fit decrease.

Data Imputation Strategies

- ▶ Missing values can be filled in by sampling nonmissing values of the variable, or by using a constant such as median or mean nonmissing value.
- ▶ You can gather alternative information from external sources.
- ▶ Imputation can use relationships among the X_i and between X_i and Y .
- ▶ We can consider the reason for nonresponse if known.

Rough Guidelines for Imputation

- ▶ If the proportion of missing is ≤ 0.05 . For continuous variables imputing missings with the median nonmissing value is adequate; for categorical predictors the most frequent category can be used.
- ▶ If the proportion of missings is between 0.05 and 0.15. If a predictor is unrelated to all of the other predictors, imputation as above should be used. If the predictor is correlated with other predictors, develop a customized model to predict the values based on the other predictors. For categorical variables, classification trees are suggested.
- ▶ For cases where the proportion of missings is ≥ 0.15 , it may be necessary to refit the model on the subset of observations for which that predictor is not missing.

Text and Categorical Variables

Pandas provides a simple method called **factorization**, in which words are uniquely mapped to a number. Similar method is available in R, when you call a feature a "factor".

One simple methodology is to create **one-hot encoding** in which one attribute will be equal to 1 (hot) and the rest zero.

Variable Transformation

We use variable transformation for one of the following reasons:

- ▶ To stabilize the variance of (one) of the X_i (in case the homoscedasticity assumption is violated)
- ▶ To normalize one of the variables X_i .
- ▶ To linearize the regression model.

Common transformations

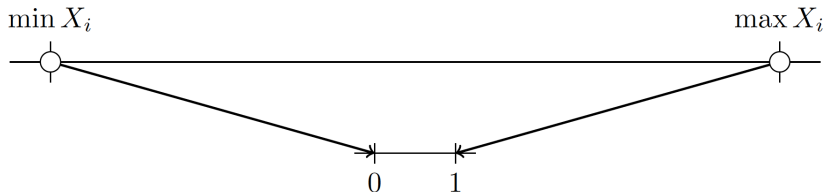
- ▶ The *log transformation* so $\tilde{Y} = \log Y$ provided that Y is positive. It is used when the variance increases markedly with increasing Y .
- ▶ The *square root transformation* so $\tilde{Y} = \sqrt{Y}$ (provided Y is positive). If the variance is proportional to the mean of Y .
- ▶ The *square transformation* so $\tilde{Y} = Y^2$ if the variance decreases with the mean of Y .

Feature Scaling

In theory, we can have features with much different scales (sometimes called "dynamic ranges"). It is a good practice to adjust the features. There are two main tricks normally used

- **min-max scaling.** This process maps all the values of the feature X_i into the interval $[0, 1]$

$$\tilde{X}_i = \frac{X_i - X_i^{\min}}{X_i^{\max} - X_i^{\min}}$$



Feature Standardization

If we have the variable X we can use the transformation

$$\tilde{X} = \frac{X - \mu}{\sigma},$$

where $\mu = E(X)$ and $\sigma = \sigma(X)$.

Note. The standardization relies on estimates of the mean and standard deviation.

Cross validation

Important Note

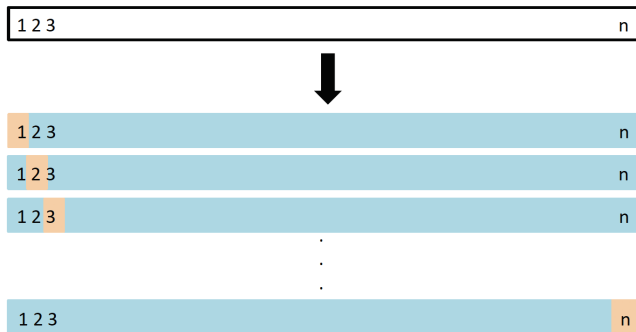
In this part, we will assume that we have access to the test dataset. Roughly speaking, we will consider methods that estimate the **test error rate** by *holding out* a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations.

Leave-one-out cross-validation

LOOCV involves splitting the set of observations into two parts. A single observation (x_1, y_1) is used for the validation set, and the remaining observations $\{(x_2, y_2), \dots, (x_n, y_n)\}$ make up for the training set. The statistical learning method is fit on the $n - 1$ training observations and a prediction \hat{y}_1 is made for the excluded observation. $MSE_1 = (y_1 - \hat{y}_1)^2$ provides an approximately unbiased estimate for the test error. We repeat this process for (x_2, y_2) and so forth, then the LOOCV estimate for the test MSE is

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^n MSE_i$$

LOOCV idea

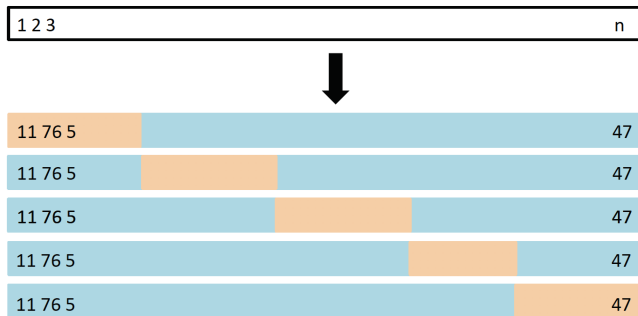


k -fold Cross Validation

This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining $k - 1$ folds. The mean squared error MSE_1 is then computed on the observations in the held-out fold. This process is repeated k times; each time, a different group of observations is treated as a validation set. This process results in k estimates of the test error. The k -fold CV estimate is computed by averaging

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^k MSE_i$$

k -fold CV idea








What have we learned?

- ▶ Measures of performance commonly used in statistical models (RSE, RSS, RMSE).
- ▶ Common practices for preparing datasets for ML.
- ▶ Revisiting Expectation, Variance, Correlations and Variance-Covariance matrices.
- ▶ Best practices for data imputation.
- ▶ Data transformation.
- ▶ Cross validation as an estimate of test error.

References

Materials and some of the pictures are from (Gareth et al., 2015).

-  Gareth, J. et al. (2015). *An Introduction to Statistical Learning*. 1st edition. Springer. ISBN: 978-1-4614-7137-0.
-  Harrell, F. E. Jr. (2001). *Regression Modeling Strategies. with Applications to Linear Models, Logistic Regression and Survival Analysis*. 1st. edition. Springer Series in Statistics. Springer. ISBN: 978-0-387-95232-1.
-  Hastie, T., R. Tibshirani, and J. Friedman (2001). *The Elements of Statistical Learning*. 1st Edition. Springer Series in Statistics. Springer. ISBN: 978-0-0-387-95284-0.
-  Kleinbau, D. et al. (2008). *Applied Regression Analysis and Other Multivariable Methods*. 4th edition. Duxbury. ISBN: 978-0-495-38496-0.
-  Pestman, W. R. (1998). *Mathematical Statistics. An Introduction*. De Gruyter. ISBN: 3-11.015356-4.

I have used some of the graphs by hacking TiKz code from StakExchange and other old tricks of T_EX