Lecture 1: Introduction to Machine Learning

Gonzalo De La Torre Parra, Ph.D.

Fall 2021

What does the matrix $\mathbb{X}^T\mathbb{X}$ looks like?

- **Columns of $\mathbb{X}$**: Recall that

$$\mathbb{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}.$$

Now, define the p vectors for the columns of matrix $\mathbb{X}$:

$$x^{(1)} = \begin{bmatrix} x_{11} \\ \vdots \\ x_{n1} \end{bmatrix}, \quad x^{(2)} = \begin{bmatrix} x_{12} \\ \vdots \\ x_{n2} \end{bmatrix}, \quad \ldots, \quad x^{(p)} = \begin{bmatrix} x_{1p} \\ \vdots \\ x_{np} \end{bmatrix}.$$

For the diabetes data example, $x^{(1)}$ is the collection of the 'AGE' attribute for all the 442 patients; $x^{(2)}$ is 'SEX' attribute, $x^{(3)}$ is 'BMI', etc. With this new notation, we have

$$\mathbb{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} = \begin{bmatrix} x^{(1)} & x^{(2)} & \ldots & x^{(p)} \end{bmatrix}.$$

We say that $\mathbb{X}$ has n rows and p columns.

- $\mathbb{X}^T\mathbb{X}$ **for the general case**: For arbitrary p, the matrix $\mathbb{X}^T\mathbb{X}$ has a similar expression (you should try to verify it directly, it is quite easy):

$$\mathbb{X}^T\mathbb{X} = \begin{bmatrix} \langle x^{(1)}, x^{(1)} \rangle & \langle x^{(1)}, x^{(2)} \rangle & \dots & \langle x^{(1)}, x^{(p)} \rangle \\ \langle x^{(2)}, x^{(1)} \rangle & \langle x^{(2)}, x^{(2)} \rangle & \dots & \langle x^{(2)}, x^{(p)} \rangle \\ \vdots & & & \\ \langle x^{(p)}, x^{(1)} \rangle & \langle x^{(p)}, x^{(2)} \rangle & \dots & \langle x^{(p)}, x^{(p)} \rangle \end{bmatrix}.$$

  Note that it is a $p \times p$ symmetric matrix.

- Now that we know what the matrix $\mathbb{X}^T\mathbb{X}$ looks like, we can investigate its properties.

When is the matrix $\mathbb{X}^T\mathbb{X}$ invertible?

We state this in the form of a theorem. We will see a sketch of the proof soon.

Theorem

The matrix $\mathbb{X}^T\mathbb{X}$ is invertible if and only if the columns of $\mathbb{X}^T\mathbb{X}$ are linearly independent if and only if the columns $x^{(1)}, x^{(2)}, \ldots, x^{(p)}$ of $\mathbb{X}$ are linearly independent.

- **Linear Independence**: Let us first understand what linear independence is: in words, *a collection of vectors are called linearly independent if they cannot be written as a linear function of each other*.
- **Examples**:
  - Vectors $a = [1, 1]$ and $b = [2, 2]$ are **not** linearly independent vectors in $\mathbb{R}^2$ because we can write $b = 2a$.
  - Vectors $a = [1, 0, 1]$ and $b = [3, 0, 3]$ are **not** linearly independent vectors in $\mathbb{R}^3$ because we can write $b = 3a$.
  - Vectors $a = [1, 0, 1]$, $b = [3, 0, 3]$, $c = [1, 0, 0]$ are **not** linearly independent vectors in $\mathbb{R}^3$ because a and b are not linearly independent.

Definition

A set of vectors $v_1, v_2, \ldots, v_p$ in $\mathbb{R}^n$ are called **linearly independent** if

$$c_1 v_1 + c_2 v_2 + \cdots + c_p v_p = [0, 0, \ldots, 0] \quad \text{(the vector of } n \text{ zeros)}.$$

implies that all the constants are zero: $c_1 = c_2 = \cdots = c_p = 0$.

- **Remark**: If we can find non-zero constants for which the above relation is true then we will be able to write (assuming $c_1 \neq 0$)

$$c_1 v_1 = -c_2 v_2 - \cdots - c_p v_p,$$

and v_1 will be a linear function of $v_2, \ldots, v_p$.

- **Examples**:
  - The vectors $e_1 = [1, 0, 0]$, $e_2 = [0, 1, 0]$, $e_3 = [0, 0, 1]$ are linearly independent. Because for constants c_1, c_2, c_3,

$$c_1 e_1 + c_2 e_2 + c_3 e_3 = [c_1, c_2, c_3] = [0, 0, 0]$$

    if and only if $c_1 = 0$, $c_2 = 0$, and $c_3 = 0$ (violating the non-zero condition in the definition).
  - Check that $v_1 = [1, 1, 0]$, $v_2 = [0, 1, 1]$, $v_3 = [1, 0, 1]$ are also linearly independent!

Lemma

A square matrix is invertible if and only if its columns are linearly independent.

Although the proof of this is not that difficult, we skip this and take this for granted for the rest of the course. Otherwise, this ML course will become a course in linear algebra.

- We saw that the vectors $e_1 = [1, 0, 0]$, $e_2 = [0, 1, 0]$, $e_3 = [0, 0, 1]$ are linearly independent. This means the identity matrix given below is invertible because e_1, e_2, e_3 are the columns of it:

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

- Similarly, because $v_1 = [1, 1, 0]$, $v_2 = [0, 1, 1]$, $v_3 = [1, 0, 1]$ are linearly independent, the following matrix is invertible:

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

- **Can you think of a matrix that is not invertible?**

Linear Combinations of Columns

- Recall that

$$\mathbb{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} = \begin{bmatrix} x^{(1)} & x^{(2)} & \cdots & x^{(p)} \end{bmatrix}.$$

where $x^{(1)}, x^{(2)}, \ldots, x^{(p)}$ are the columns of matrix $\mathbb{X}$:

$$x^{(1)} = \begin{bmatrix} x_{11} \\ \vdots \\ x_{n1} \end{bmatrix}, \quad x^{(2)} = \begin{bmatrix} x_{12} \\ \vdots \\ x_{n2} \end{bmatrix}, \quad \ldots, \quad x^{(p)} = \begin{bmatrix} x_{1p} \\ \vdots \\ x_{np} \end{bmatrix}.$$

- Then, the matrix multiplication $\mathbb{X}\beta$ is a linear combination of the columns of $\mathbb{X}$:

$$\mathbb{X}\beta = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} = \begin{bmatrix} x_{11} \\ \vdots \\ x_{n1} \end{bmatrix} \beta_1 + \begin{bmatrix} x_{12} \\ \vdots \\ x_{n2} \end{bmatrix} \beta_2 + \ldots \begin{bmatrix} x_{1p} \\ \vdots \\ x_{np} \end{bmatrix} \beta_p.$$

Proof of the Theorem

Theorem

The matrix $\mathbb{X}^T\mathbb{X}$ is invertible if and only if the columns $x^{(1)}, x^{(2)}, \ldots, x^{(p)}$ of $\mathbb{X}$ are linearly independent.

- **Proof**: We will prove that the columns of $\mathbb{X}$ are linearly dependent if and only if the columns of $\mathbb{X}^T\mathbb{X}$ are linearly dependent.

  If the columns of $\mathbb{X}$ are linearly dependent, then there exists a non-zero vector $c = (c_1, \cdots, c_p)$ such that

  $$\mathbb{X}c = 0.$$

  Multiply by $\mathbb{X}^T$ to get

  $$\mathbb{X}^T\mathbb{X}c = 0.$$

  This proves that the columns of $\mathbb{X}^T\mathbb{X}$ are linearly dependent and that it is not invertible (based on the lemma).

- **Proof Continued**: Now if the columns of $\mathbb{X}^T\mathbb{X}$ are linearly dependent, then there exists a non-zero vector $c = (c_1, \cdots, c_p)$ such that

$$\mathbb{X}^T\mathbb{X}c = 0.$$

Multiple on the left by c^T to get

$$c^T\mathbb{X}^T\mathbb{X}c = 0.$$

But,

$$c^T\mathbb{X}^T\mathbb{X}c = \langle \mathbb{X}c, \mathbb{X}c \rangle.$$

This means that

$$\mathbb{X}c = 0$$

because the inner product of a vector with itself is the sum of square of the components, and if the sum of square of the components is zero, then each of the component must be zero. This means that the columns of $\mathbb{X}$ are linearly dependent. Note that $\mathbb{X}$ is not (or need not be) a square matrix, so the lemma on invertibility does not apply to $\mathbb{X}$.

Further Understanding of the Ridge and LASSO Solutions

- **Constraint version of ridge**: Solving the unconstrained version of ridge problem given by

$$\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 + \lambda \sum_{i=1}^{p} \beta_i^2$$

  is equivalent to solving the following constraint version

$$\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2$$

$$\text{subject to} \sum_{i=1}^{p} \beta_i^2 \leq t_\lambda .$$

  Note that the constraint is a circular region because of the squares.

- Choosing a large λ in the unconstrained version is equivalent to choosing a smaller t_λ in the constrained version.

- **Constraint version of LASSO**: Solving the unconstrained version of LASSO problem given by

$$\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 + \lambda \sum_{i=1}^{p} |\beta_i|$$

is equivalent to solving the following constraint version

$$\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2$$

$$\text{subject to} \sum_{i=1}^{p} |\beta_i| \leq t_\lambda.$$

Note that the constraint is a diamond-shaped region because of the magnitude.

- Choosing a large λ in the unconstrained version is equivalent to choosing a smaller t_λ in the constrained version.

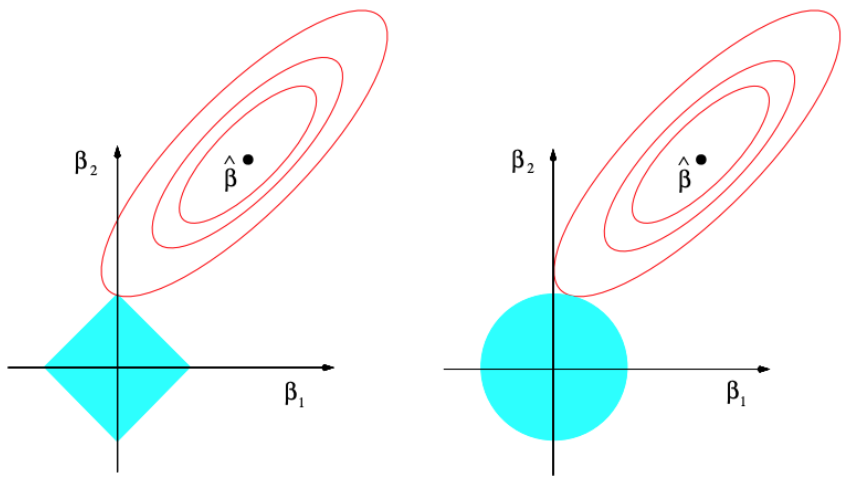
Understanding the Solutions using Level Sets

Definition

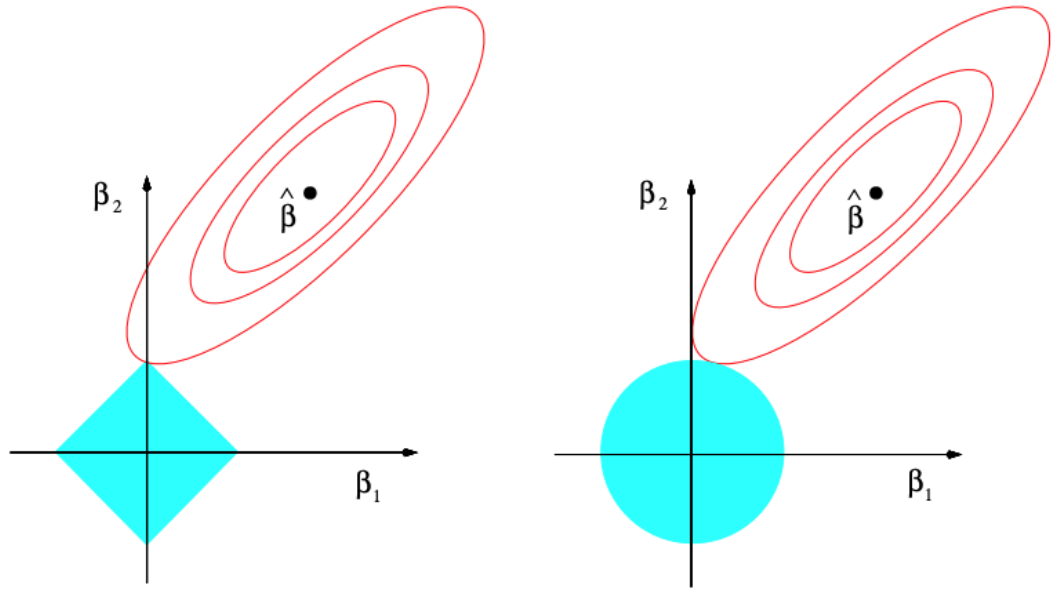
Given a function $f(\beta)$, a level set is defined for any $c \in \mathbb{R}$ as the set of all β such that $f(\beta) = c$. Mathematically,

$$L_c = \{\beta : f(\beta) = c\}.$$

For $p = 2$, a level set is $L_c = \{(\beta_1, \beta_2) : f(\beta_1, \beta_2) = c\}$. A convex function in two variables is a cup shaped function. For such a function, the level sets are concentric as shown in the figure. In the figure, $\hat{\beta}$ is the linear regression solution.



Left figure is for LASSO and the right figure is for Ridge.



Optimizing a convex function is equivalent to finding the non-empty level set L_c with the minimum value of c.

When there is no constraint ($\lambda = 0$ or $t_\lambda = \infty$), or when the constraint is loose (small λ or large t_λ), then the optimal level is achieved at $\hat{\beta} = \beta_{ols}$.

When the constraint is significant (large λ or small t_λ), the optimal level has to be achieved while making sure that the constraint region is touched.
Note that the intuition is given using two-dimensional variables, but is valid more generally.

- **LASSO**: For LASSO, the level curves touch the triangular constraint region at the corners. As a result, some of the coefficients are exactly zero.
- **Ridge**: For ridge regression, the level curves can touch the circular constraint region at any point. As a result, the coefficients are not exactly zero, but are shrunked version of the linear regression coefficients.

Training, Testing, Validation, and Cross-Validation

- **Training process**: It is the process of finding the best predictor in a hypothesis class:

$$\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2.$$

Often this objective needs to be modifed to allow for regularization (as in Ridge, LASSO, Elastic Net):

$$\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2 + \lambda \, \text{Pen}(h).$$

Here $\text{Pen}(h)$ is a penalty term that penalizes the complexity of the predictor h.

- **Generalization error**: If we manage to solve such a problem, as we did for Linear regression, Ridge regression, LASSO, and Elastic Net, we may need to estimate the performance of the learned predictor h_s. The best way to check the performance is to obtain an entirely new dataset

$$(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m})$$

and estimate the quality by evaluating

$$L_G(h_s) = \frac{1}{m} \sum_{j=1}^{m} (h_s(x_{n+j}) - y_{n+j})^2.$$

Note: No penalty on complexity here!

- **Ideal generalization error**: Ideally, we need infinite amount of data for generalization error calculation and the true generalization error is really given by
$$\lim_{m\to\infty} \frac{1}{m} \sum_{j=1}^{m} (h_s(x_{n+j}) - y_{n+j})^2.$$
- **The reality**: In practice, we are often only given a dataset. There is not always a clear distinction between which data to use for training and which one to use for generalization error calculations. We need to decide these ourselves. This predicament leads to the concepts of **training, testing, and validation.**
- **Approach 1**: Given training data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, divide the n data points into three parts: one part for training, one part to help with the optimization of hyperparameters like λ (called a validation set), and the third for testing. Testing is a way to estimate the ideal generalization error from finite data.
- **Approach 2**: Divide the n data points into two parts: one for training and another for testing. Use *cross-validation* on the training data to estimate hyperparameters.
- **Important remark on how to divide the data**: There is no exisiting theory on how to deal with this issue. However, the same intuitive ideas apply: **If we need a large number of samples for training, then we also need a large number of samples to test and validate**.

- **So, what are the concepts of validation and testing?** Let us look at it through an example. Suppose we are given 100 data points:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_{100}, y_{100}).$$

Let us further assume that we wish to solve the Ridge problem.

- **Following Approach 1**: Divide the data into three chunks with sizes $(50, 20, 30)$ (these choices are ad-hoc). Use the following approach:
  1. Fix λ:
  2. Train using $(x_1, y_1), (x_2, y_2), \ldots, (x_{50}, y_{50})$, i.e., solve

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{50} \sum_{i=1}^{50} (\beta^T x_i - y_i)^2 + \lambda \sum_{i=1}^{p} \beta_i^2.$$

Let us call the solution β_λ (because it depends on the chosen λ).

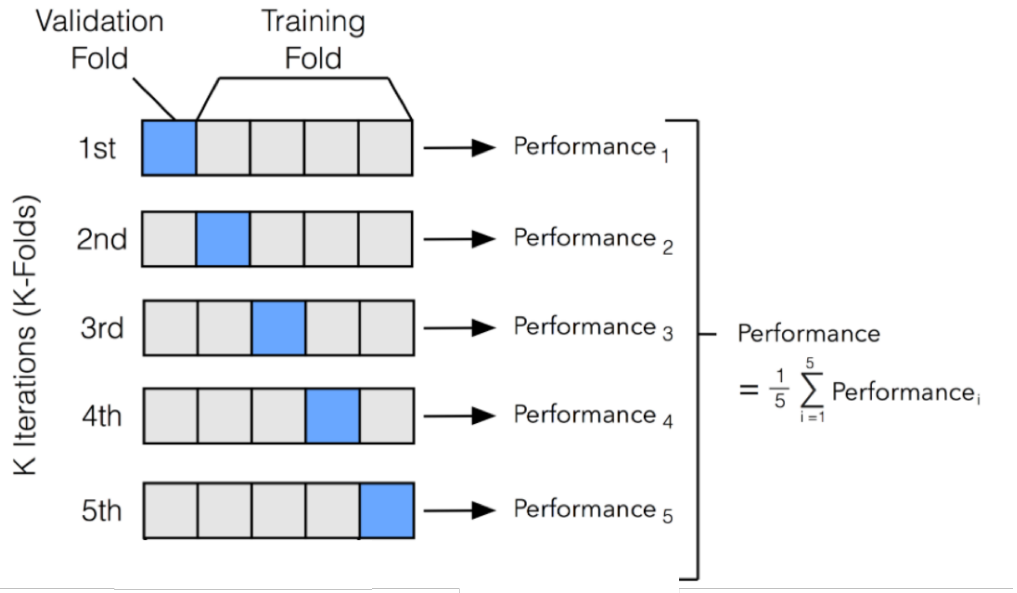
  3. Validate the accuracy using $(x_{51}, y_{51}), (x_{52}, y_{52}), \ldots, (x_{70}, y_{70})$, i.e., evaluate

$$\frac{1}{20} \sum_{i=51}^{70} (\beta_\lambda^T x_i - y_i)^2.$$

  4. Go back to step 1 and change λ.
  5. Repeat steps 1 through 4 for different values of λ and pick the one with the smallest validation error. Let us call the optimal λ, λ^*.
  6. Finally, evaluate the performance of the best predictor for coefficient β_{λ^*} using the testing set:

$$\frac{1}{30} \sum_{i=71}^{100} (\beta_{\lambda^*}^T x_i - y_i)^2.$$

- **Following Approach 2**: Divide the data into two chunks with sizes $(70, 30)$ (these choices are ad-hoc) and call them training and testing data, respectively. Use the following approach called $K-$fold cross-validation.



1. Divide the training data of size 70 into $K = 5$ chunks (say).
2. Fix λ.
3. Train using chunks 2, 3, 4, 5 and validate on chunk 1. The validation error here is your performance 1.
4. Repeat this for each chunk as shown in the figure and obtain Peformance 2,3,4,5. Note that the solution β_λ will be different for different chunks.
5. Average the validation performance over all 5 folds.
6. Go back to step 2 and change λ.
7. Repeat steps 2 through 6 for different values of λ and pick the one with the smallest cross-validation error. Let us call the optimal λ, λ^*.
8. Finally, evaluate the performance of the best predictor for coefficient β_{λ^*} using the testing set.

Remarks 1

Remarks 2

Remarks 3

Remarks 4