Lecture 3: Linear Predictors Cont.

Gonzalo De La Torre Parra, Ph.D.

Fall 2021

Linear Predictors: Notations

- Given n training data points for supervised ML

$$(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n),$$

where $x \in \mathbb{R}^p$ and $y \in \mathbb{R}$, we collect them in the matrix $\mathbb{X}$ and vector $\mathbb{Y}$.

- **Matrix** $\mathbb{X}$: We collect the x values of the n training samples in a matrix

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

Thus, we assume that each x value is a p-dimensional vector:

$$\begin{aligned} x_1 &= (x_{11}, x_{12}, \ldots, x_{1p}) \\ x_2 &= (x_{21}, x_{21}, \ldots, x_{2p}) \\ &\vdots \\ x_n &= (x_{n1}, x_{12}, \ldots, x_{np}). \end{aligned}$$

- **Vector** $\mathbb{Y}$: We collect the y values of the n training samples in a vector

$$\mathbb{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

Thus, we assume that each y value is a real number.

The **diabetes data** contains data from 442 patients. It has 10 variables. We need to use those 10 variables (I have shown only 6 of them below) to predict a quantitative measure of disease (diabetes) progression one year after baseline.

```
Y              AGE          SEX          BMI          BP          S1          S2
151.0       [ 0.03807591  0.05068012  0.06169621  0.02187235 -0.0442235  -0.03482076]
75.0        [-0.00188202 -0.04464164 -0.05147406 -0.02632783 -0.00844872 -0.01916334]
141.0       [ 0.08529891  0.05068012  0.04445121 -0.00567061 -0.04559945 -0.03419447]
206.0       [-0.08906294 -0.04464164 -0.01159501 -0.03665645  0.01219057  0.02499059]
135.0       [ 0.00538306 -0.04464164 -0.03638469  0.02187235  0.00393485  0.01559614]
97.0        [-0.09269548 -0.04464164 -0.04069594 -0.01944209 -0.06899065 -0.07928784]
138.0       [-0.04547248  0.05068012 -0.04716281 -0.01599922 -0.04009564 -0.02480001]
63.0        [ 0.06350368  0.05068012 -0.00189471  0.06662967  0.09061988  0.10891438]
110.0       [ 0.04170844  0.05068012  0.06169621 -0.04009932 -0.01395254  0.00620169]
310.0       [-0.07090025 -0.04464164  0.03906215 -0.03321358 -0.01257658 -0.03450761]
```

For this example, $n = 442$ and $p = 10$. As a result, the matrix $\mathbb{X}$ will be a 442×10 matrix and the vector $\mathbb{Y}$ will be a 442×1 vector or matrix.

Linear Predictors

Definition

A linear predictor is a linear function of its variables: Let u and β be p-dimensional vectors $u = (u_1, u_2, \ldots, u_p)$ and $\beta = (\beta_1, \beta_2, \ldots, \beta_p)$. Then a linear predictor as a function of u is defined as

$$h(u_1, u_2, \ldots, u_p) = \beta_1 u_1 + \beta_2 u_2 + \cdots + \beta_p u_p$$
$$h(u) = \beta^T u.$$

Examples:

- Given a data point $x_1 = (x_{11}, x_{12}, \ldots, x_{1p})$, the linear predictor maps x_1 to

$$h(x_1) = h(x_{11}, x_{12}, \ldots, x_{1p}) = \beta_1 x_{11} + \beta_2 x_{12} + \cdots + \beta_p x_{1p} = \beta^T x_1.$$

- Given a data point $x_2 = (x_{21}, x_{22}, \ldots, x_{2p})$, the linear predictor maps x_2 to

$$h(x_2) = h(x_{21}, x_{22}, \ldots, x_{2p}) = \beta_1 x_{21} + \beta_2 x_{22} + \cdots + \beta_p x_{2p} = \beta^T x_2.$$

- Training using linear predictors is trying to predict the vector $\mathbb{Y}$ using the matrix $\mathbb{X}$:

$$\mathbb{Y} \longleftarrow \mathbb{X}\beta.$$

- In detail, this equation is equal to

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \longleftarrow \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}.$$

- Element-wise, this looks like

$$\begin{aligned} y_1 &\longleftarrow \beta_1 x_{11} + \beta_2 x_{12} + \cdots + \beta_p x_{1p} \\ &\vdots \\ y_n &\longleftarrow \beta_1 x_{n1} + \beta_2 x_{n2} + \cdots + \beta_p x_{np} \end{aligned}$$

- **Remember that you are using the same β vector on every training data. This is because you fix a predictor and see its performance on the entire training data. If you are not happy with the performance you change your predictor. In case of linear predictors, you change the vector β.**

Hypothesis Class of Linear Predictors

- Choosing a linear predictor amounts to choosing the coefficients of the linear function $\beta = (\beta_1, \ldots, \beta_p)$.
- Note that applying a predictor means applying the **same** function h to all the training data. So, when we move from one training point to another, the x vector changes, but the βs remain the same.
- The class of linear predictors is the simplest class of predictors you can choose. Motivated by this, let us define

$$\mathcal{H} = \{\text{Class of linear predictors}\}.$$

- Training process then becomes (for regression and squared error loss)

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

Linear Regression: Special Case of $p = 1$

- **Data**: We are given n training points

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n).$$

  Here x and y variables are real numbers. We have n training points n. We are again asked to learn a linear predictor using the data.

- **Training**: The training for a linear predictor looks the same:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 = \min_{\beta_1} \frac{1}{n} \sum_{i=1}^{n} (\beta_1 x_i - y_i)^2$$

  Here, $\beta = (\beta_1, \ldots, \beta_p) = \beta_1$. For $p = 1$, this is fitting a line to a set of points.

- **Solving the optimization problem**: How do you solve the optimization problem to learn the best β_1?
  - This is a *convex* function with a unique minimum (more on this in the coming lectures).
  - It is also differentiable.
  - The optimal point can be obtained by taking the derivative with respect to β_1 and setting the derivative to zero.

- **Convex Functions**: A convex function looks like as shown in the figure below (Taken from Boyd's book).

Figure 3.1 Graph of a convex function. The chord (*i.e.*, line segment) between any two points on the graph lies above the graph.

- **Optimization of a convex function**: Suppose $f(\beta)$ is a convex function and we want to solve

$$\min_{\beta} f(\beta).$$

If a minimum exists, it is unique and can be obtained by setting the derivative of $f(\beta)$ with respect to β to zero.

- **Solving for** β_1: We solve the optimization problem

$$\min_{\beta_1} \frac{1}{n} \sum_{i=1}^{n} (\beta_1 x_i - y_i)^2$$

  by again taking the derivative with respect to β_1 and setting it to zero.

- **The derivative**: The derivative is

$$\frac{d}{d\beta_1} \frac{1}{n} \sum_{i=1}^{n} (\beta_1 x_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} 2(\beta_1 x_i - y_i)(x_i).$$

  Simplification gives

$$2 \left(-\frac{1}{n} \sum_{i=1}^{n} x_i y_i + \beta_1 \frac{1}{n} \sum_{i=1}^{n} x_i^2 \right).$$

- **The solution**: Setting this to zero gives us the solution to the ordinary lease square or training problem for general n:

$$\beta_{ols} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}.$$

- **The learned predictor**: Note that your learned predictor as a function of a variable u is

$$h_s(u) = \beta_{ols} u = \frac{\sum_{i=1}^n x_i y_i}{\sum_{i=1}^n x_i^2} u.$$

  So, it is obviously an explicit function of your training data.

- **Application to New Data**: How do you use the learned predictor on new data? Given a new data point x^* (remember, no corresponding y^* will be given), your predicted value at this new x^* will be

$$\hat{y^*} = h_s(x^*) = \frac{\sum_{i=1}^n x_i y_i}{\sum_{i=1}^n x_i^2} x^*.$$

- **Generalization Error**: If I now give you a new (possibly infinite) set of $x - y$ labels to test your predictor (in this case you will be given the y values) $(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m})$, the generalization error would be

$$\begin{aligned}\frac{1}{m}\sum_{j=1}^m \ell(h_s(x_{n+j}), y_{n+j}) &= \frac{1}{m}\sum_{j=1}^m \left(y_{n+j} - \beta_{ols} x_{n+j}\right)^2. \\ &= \frac{1}{m}\sum_{j=1}^m \left(y_{n+j} - \frac{\sum_{i=1}^n x_i y_i}{\sum_{i=1}^n x_i^2} x_{n+j}\right)^2.\end{aligned}$$

Linear Regression: Special Case with $p = 2$

- **Data**: We are given n training points

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n).$$

Now, each x_i is a vector of dimension $p = 2$:

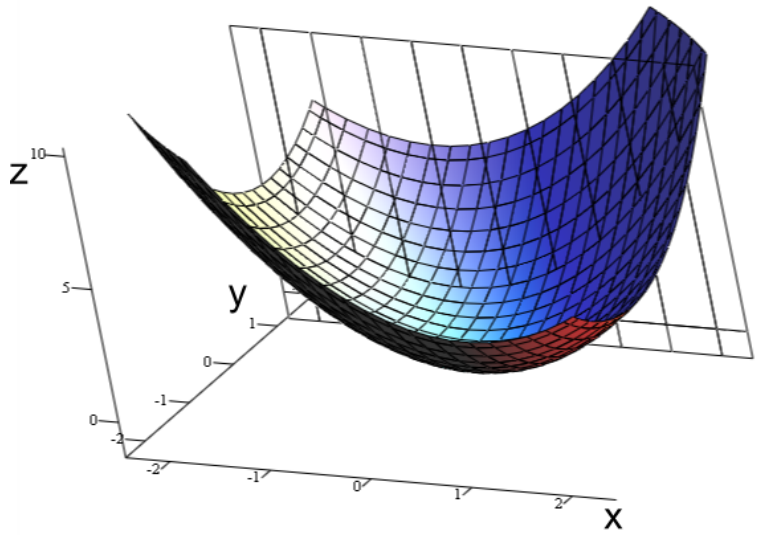
$$\begin{aligned} x_1 &= (x_{11}, x_{12}) \\ x_2 &= (x_{21}, x_{22}) \\ &\vdots \\ x_n &= (x_{n1}, x_{n2}). \end{aligned}$$

- **Training**: The training process for a linear predictor is:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 = \min_{\beta_1, \beta_2} \frac{1}{n} \sum_{i=1}^{n} (\beta_1 x_{i1} + \beta_2 x_{i2} - y_i)^2$$

Here, $\beta = (\beta_1, \ldots, \beta_p) = (\beta_1, \beta_2)$. **Depending on the problem, this can be fitting a line (with intercept) or fitting a plane**.

- **Convex Function in Two Variables**: A convex function in two variables $f(\beta_1, \beta_2)$ looks like as shown in the figure below (taken from Wikipedia):



- **Optimization of a convex function**: We want to solve

$$\min_{\beta_1, \beta_2} f(\beta_1, \beta_2).$$

If a minimum exists, it is unique and can be obtained by setting the gradient

$$\nabla_\beta f(\beta_1, \beta_2) := \begin{bmatrix} \frac{\partial}{\partial \beta_1} f(\beta_1, \beta_2) \\ \frac{\partial}{\partial \beta_2} f(\beta_1, \beta_2) \end{bmatrix}$$

to zero.

- **The gradient**: The gradient is computed as follows.

$$\frac{\partial}{\partial \beta_1} \frac{1}{n} \sum_{i=1}^{n} (\beta_1 x_{i1} + \beta_2 x_{i2} - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} 2(\beta_1 x_{i1} + \beta_2 x_{i2} - y_i)(x_{i1})$$

$$\frac{\partial}{\partial \beta_2} \frac{1}{n} \sum_{i=1}^{n} (\beta_1 x_{i1} + \beta_2 x_{i2} - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} 2(\beta_1 x_{i1} + \beta_2 x_{i2} - y_i)(x_{i2}).$$

Simplification gives

$$\nabla_\beta f(\beta_1, \beta_2) = \begin{bmatrix} \frac{\partial}{\partial \beta_1} f(\beta_1, \beta_2) \\ \frac{\partial}{\partial \beta_2} f(\beta_1, \beta_2) \end{bmatrix}$$

$$= \begin{bmatrix} 2\left(-\frac{1}{n}\sum_{i=1}^{n} x_{i1} y_i \quad + \quad \beta_1 \frac{1}{n}\sum_{i=1}^{n} x_{i1}^2 \quad + \quad \beta_2 \frac{1}{n}\sum_{i=1}^{n} x_{i1} x_{i2}\right) \\ 2\left(-\frac{1}{n}\sum_{i=1}^{n} x_{i2} y_i \quad + \quad \beta_1 \frac{1}{n}\sum_{i=1}^{n} x_{i1} x_{i2} \quad + \quad \beta_2 \frac{1}{n}\sum_{i=1}^{n} x_{i2}^2\right) \end{bmatrix}$$

- **Setting the gradient to zero**: In order to find the optimal point, we need to set the gradient $\nabla_\beta f(\beta_1, \beta_2)$ to zero. This gives us

$$\begin{bmatrix} 2\left(-\frac{1}{n}\sum_{i=1}^n x_{i1}y_i + \quad \beta_1 \frac{1}{n}\sum_{i=1}^n x_{i1}^2 + \quad \beta_2 \frac{1}{n}\sum_{i=1}^n x_{i1}x_{i2}\right) \\ 2\left(-\frac{1}{n}\sum_{i=1}^n x_{i2}y_i + \quad \beta_1 \frac{1}{n}\sum_{i=1}^n x_{i1}x_{i2} + \quad \beta_2 \frac{1}{n}\sum_{i=1}^n x_{i2}^2\right) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

  The above equation is equivalent to (remove scaling by 2 and $\frac{1}{n}$)

$$\begin{bmatrix} \beta_1 \sum_{i=1}^n x_{i1}^2 + \quad \beta_2 \sum_{i=1}^n x_{i1}x_{i2} \\ \beta_1 \sum_{i=1}^n x_{i1}x_{i2} + \quad \beta_2 \sum_{i=1}^n x_{i2}^2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n x_{i1}y_i \\ \sum_{i=1}^n x_{i2}y_i \end{bmatrix}.$$

  This in turn can be written in the matrix format:

$$\begin{bmatrix} \sum_{i=1}^n x_{i1}^2 & \sum_{i=1}^n x_{i1}x_{i2} \\ \sum_{i=1}^n x_{i1}x_{i2} & \sum_{i=1}^n x_{i2}^2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n x_{i1}y_i \\ \sum_{i=1}^n x_{i2}y_i \end{bmatrix}.$$

- **The solution**: The OLS solution for $p = 2$ is given by (assuming invertibility of the matrix on the left above)

$$\beta_{ols} = \begin{bmatrix} \beta_{1,ols} \\ \beta_{2,ols} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n x_{i1}^2 & \sum_{i=1}^n x_{i1}x_{i2} \\ \sum_{i=1}^n x_{i1}x_{i2} & \sum_{i=1}^n x_{i2}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^n x_{i1}y_i \\ \sum_{i=1}^n x_{i2}y_i \end{bmatrix}.$$

- **The learned predictor**: Note that your learned predictor is (with $u = (u_1, u_2)$)

$$\begin{aligned} h_s(u) &= \beta_{ols}^T u = \beta_{1,ols} u_1 + \beta_{2,ols} u_2 \\ &= \left(\begin{bmatrix} \sum_{i=1}^n x_{i1}^2 & \sum_{i=1}^n x_{i1} x_{i2} \\ \sum_{i=1}^n x_{i1} x_{i2} & \sum_{i=1}^n x_{i2}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^n x_{i1} y_i \\ \sum_{i=1}^n x_{i2} y_i \end{bmatrix} \right)^T \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \end{aligned}$$

- **Application to New Data**: Prediction on new $x^* = (x_1^*, x_2^*)$:

$$\begin{aligned} \hat{y^*} &= h_s(x^*) = \beta_{ols}^T x^* = \beta_{1,ols} x_1^* + \beta_{2,ols} x_2^* \\ &= \left(\begin{bmatrix} \sum_{i=1}^n x_{i1}^2 & \sum_{i=1}^n x_{i1} x_{i2} \\ \sum_{i=1}^n x_{i1} x_{i2} & \sum_{i=1}^n x_{i2}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^n x_{i1} y_i \\ \sum_{i=1}^n x_{i2} y_i \end{bmatrix} \right)^T \begin{bmatrix} x_1^* \\ x_2^* \end{bmatrix}. \end{aligned}$$

- **Generalization Error**: the generalization error for new data $(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m})$ would be

$$\frac{1}{m} \sum_{j=1}^m \left(y_{n+j} - \beta_{ols}^T x_{n+j} \right)^2.$$

Affine Functions

- **Affine Function of one variable**: An affine function of one variable is defined as

$$h(u) = b + au$$

  for real numbers a and b. This is also a line with an intercept b. It passes through the origin only if $b = 0$.

- **Affine functions of two variables**: An affine function of two variables is defined as

$$h(u_1, u_2) = a_1 u_1 + a_2 u_2 + b$$

  for some constants a_1, a_2, b. It is a plane with an intercept.

- **Affine function of many variables**: More generally, we define anaffine function of many variables as

$$h(u_1, u_2, \ldots, u_p) = a_1 u_1 + a_2 u_2 + \cdots + a_p u_p + b.$$

Training With Affine Predictors

- **Training an affine precdictor**: This looks like

$$\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2 = \min_{\beta_0 \in \mathbb{R}, \beta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} (\beta^T x_i + \beta_0 - y_i)^2.$$

  Here,

$$\beta = (\beta_1, \ldots, \beta_p).$$

- **Not a new problem**: This is exactly like a OLS problem but in dimension $p+1$: for each x data point x_i, add a 1 to the first component

$$(1, x_{i1}, \cdots, x_{ip}).$$

  Then, train a linear predictor in dimension $p+1$, i.e., search for the optimal $\beta \in \mathbb{R}^{p+1}$ where now the β is modified to include β_0:

$$\beta = (\beta_0, \beta_1, \cdots, \beta_p).$$

Short Tutorial on Inverse and Transpose

- **Matrix inverse**: The inverse of a $p \times p$ square matrix A is any matrix B (of same dimensions) such that
$$AB = BA = I_p,$$
where I_p is the $p \times p$ identity matrix. Such a matrix B is denoted by A^{-1}.
  - If a matrix is not square, its inverse is not defined.
  - Not every matrix is invertible.
  - If you can express any column of a matrix as a linear function of other columns, the matrix is not invertible. The same logic applies to rows.
  - It is not at all trivial to find the inverse of a matrix by just looking at it.
  - Note the obvious fact that
$$A^{-1} \neq \frac{1}{A}.$$
The left hand side may not exist. But, the right hand side is not even well-defined.
  - Computing the matrix inverse is computationally costly.
- **Matrix transpose**: If A is a $n \times p$ matrix (not necessarily square) with elements a_{ij}, then the transpose of A is obtained by replacing a_{ij} with a_{ji}, for all i, j. For example,
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$
- **Symmetric matrix**: A square matrix is called symmetric if $A = A^T$. Note that a rectangular matrix (one which is not square) can never be symmetric.

Linear Regression for any p, n.

- **Data**: We are given n training point

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n).$$

  Now, $x \in \mathbb{R}^p$ and $y \in \mathbb{R}$.

- **Training**: The training process for the search for the best linear predictor is given by the optimization problem

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

  Here, $\beta = (\beta_1, \ldots, \beta_p)$.

  We now want to understand how to solve this more general optimization problem over the p-dimensional space $\mathbb{R}^p$. The approach is again to take the gradient with respect to β and set the p-dimensional gradient vector to zero. In the next few slides, we look at a cleaner way to do this using matrix algebra.

- Recall that

$$\mathbb{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \qquad \mathbb{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}_{n \times p}$$

where each x_i is collected in the rows of the matrix $\mathbb{X}$:

$$\begin{aligned} x_1 &= (x_{11}, x_{12}, \ldots, x_{1p}) \\ &\vdots \\ x_n &= (x_{n1}, x_{12}, \ldots, x_{np}). \end{aligned}$$

- Now note that the vector obtained by the product $\mathbb{X}\beta$ can be written as

$$\mathbb{X}\beta = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} = \begin{bmatrix} x_1^T \beta \\ \vdots \\ x_n^T \beta \end{bmatrix}$$

- The difference of two vectors $\mathbb{Y} - \mathbb{X}\beta$ can be written as

$$\mathbb{Y} - \mathbb{X}\beta = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1^T\beta \\ \vdots \\ x_n^T\beta \end{bmatrix} = \begin{bmatrix} y_1 - x_1^T\beta \\ \vdots \\ y_n - x_n^T\beta \end{bmatrix}$$

- **Important observation**: Now note that the training or sample error is nothing but a scaled version of the dot product of the vector $\mathbb{Y} - \mathbb{X}\beta$ with itself:

$$\frac{1}{n}\sum_{i=1}^{n}(\beta^T x_i - y_i)^2 = \frac{1}{n}\langle \mathbb{Y} - \mathbb{X}\beta, \mathbb{Y} - \mathbb{X}\beta \rangle = \frac{1}{n}\left(\mathbb{Y} - \mathbb{X}\beta\right)^T\left(\mathbb{Y} - \mathbb{X}\beta\right).$$

To understand the above equation, recall that the inner product or the dot product between two n-dimensional vectors:

$$\langle a, b \rangle = a^T b = \sum_{i=1}^{n} a_i b_i \qquad \langle a, a \rangle = a^T a = \sum_{i=1}^{n} a_i^2.$$

- **The optimization problem**: The OLS optimization problem is then given by

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

- In order to solve the problem, we expand the expression for the objective function and obtain

$$\frac{1}{n} (\mathbb{Y} - \mathbb{X}\beta)^T (\mathbb{Y} - \mathbb{X}\beta) = \frac{1}{n} \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right).$$

- **Optimal** β: As discussed earlier, we can show that this is a convex function and the optimal solution can be obtained by setting the gradient with respect to β to zero:

$$\nabla_\beta \frac{1}{n} \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right) = 0.$$

  Here the 0 on the right is the vector of all zeros with p components.

- **But, how do we find gradient of this expression?**

Gradient Calculations for Quadratic Forms

- The gradient ∇_β of a real valued function $f(\beta)$ of β is the column vector of partials:

$$\nabla_\beta f(\beta) = \begin{bmatrix} \frac{\partial f(\beta)}{\partial \beta_1} \\ \frac{\partial f(\beta)}{\partial \beta_2} \\ \vdots \\ \frac{\partial f(\beta)}{\partial \beta_p} \end{bmatrix}$$

- For any square matrix A,

$$\nabla_\beta \left(\beta^T A \, \beta\right) = (A + A^T)\beta.$$

- To verify this, let us try a 2×2 example:

$$\beta^T A \, \beta = \begin{bmatrix} \beta_1 & \beta_2 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \beta_1 & \beta_2 \end{bmatrix} \begin{bmatrix} a\beta_1 + b\beta_2 \\ c\beta_1 + d\beta_2 \end{bmatrix}$$

$$= \beta_1(a\beta_1 + b\beta_2) + \beta_2(c\beta_1 + d\beta_2).$$

- Thus, we have

$$\beta^T A\ \beta = \beta_1(a\beta_1 + b\beta_2) + \beta_2(c\beta_1 + d\beta_2).$$

- The gradient is given by

$$\begin{aligned}
\nabla_\beta\ (\beta^T A\ \beta) &= \begin{bmatrix} \frac{\partial(\beta^T A\ \beta)}{\partial \beta_1} \\ \frac{\partial(\beta^T A\ \beta)}{\partial \beta_2} \end{bmatrix} = \begin{bmatrix} 2a\beta_1 + b\beta_2 + c\beta_2 \\ b\beta_1 + c\beta_1 + 2d\beta_2 \end{bmatrix} \\
&= \begin{bmatrix} 2a & b+c \\ b+c & 2d \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} \\
&= \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} a & c \\ b & d \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} \\
&= A\beta + A^T\beta.
\end{aligned}$$

- What happens when A is symmetric, i.e, $A = A^T$? We get

$$\nabla_\beta\ (\beta^T A\ \beta) = (A + A^T)\beta = 2A\beta.$$

- What happens when $A = I$, the identity matrix? We get

$$\nabla_\beta\ (\beta^T A\ \beta) = \nabla_\beta\ (\beta^T \beta) = 2A\beta = 2\beta.$$

- For any column vector $a = [a_1, \cdots, a_p]^T$,

$$a^T \beta = \beta^T a = \sum_{i=1}^{p} a_i \beta_i$$

- What is the gradient of $a^T \beta$? To obtain this, just take partial derivatives of $a^T \beta = \sum_{i=1}^{p} a_i \beta_i$ with respect to components of β and arrange them in a column vector:

$$\nabla_\beta \left(a^T \beta \right) = \begin{bmatrix} \frac{\partial (a^T \beta)}{\partial \beta_1} \\ \frac{\partial (a^T \beta)}{\partial \beta_2} \\ \vdots \\ \frac{\partial (a^T \beta)}{\partial \beta_p} \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix}.$$

- Thus,

$$\nabla_\beta \left(a^T \beta \right) = a.$$

Linear Regression Solution

- **The optimization problem**: Recall that the OLS optimization problem is

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 = \min_{\beta \in \mathbb{R}^p} \frac{1}{n} \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right)$$

- We solve the above problem by setting

$$\nabla_\beta \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right) = 0.$$

  So, do we know how to compute the following gradient? Yes, because gradient (partial derivative or total derivatives) of a sum of functions is equal to the sum of partial derivatives of individual functions.

- **Gradient calculations for quadratic forms**: Also, recall that for any symmetric matrix A and vector a

$$\nabla_\beta \left(\beta^T A \, \beta \right) = (A + A^T)\beta = 2A\beta$$
$$\nabla_\beta \left(a^T \beta \right) = \nabla_\beta \left(\beta^T a \right) = a.$$

  Thus, setting $A = \mathbb{X}^T \mathbb{X}$ (which is symmetric) and $a = \mathbb{X}^T \mathbb{Y}$ we get

$$\nabla_\beta \left(\beta^T \mathbb{X}^T \mathbb{X} \, \beta \right) = 2\mathbb{X}^T \mathbb{X} \beta$$
$$\nabla_\beta \left((\mathbb{X}^T \mathbb{Y})^T \beta \right) = \nabla_\beta \left(\beta^T \mathbb{X}^T \mathbb{Y} \right) = \mathbb{X}^T \mathbb{Y}.$$

- This gives us

$$\begin{aligned}
&\nabla_\beta \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right) \\
&= \nabla_\beta \left(\mathbb{Y}^T \mathbb{Y} \right) + \nabla_\beta \left(\beta^T \mathbb{X}^T \mathbb{X} \beta \right) - 2\nabla_\beta \left(\beta^T \mathbb{X}^T \mathbb{Y} \right) \\
&= 2\mathbb{X}^T \mathbb{X} \beta - 2\mathbb{X}^T \mathbb{Y}.
\end{aligned}$$

- Setting $\nabla_\beta \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right) = 0$ we get

$$2\mathbb{X}^T \mathbb{X} \beta - 2\mathbb{X}^T \mathbb{Y} = 0,$$

  or

$$\mathbb{X}^T \mathbb{X} \beta = \mathbb{X}^T \mathbb{Y}.$$

- Multiplying from the left by $\left(\mathbb{X}^T \mathbb{X} \right)^{-1}$ (and assuming the inverse exists) both sides of the equation we get

$$\beta_{ols} = \left(\mathbb{X}^T \mathbb{X} \right)^{-1} \mathbb{X}^T \mathbb{Y}.$$

Understanding the Linear Regression Solution

We should ask the following questions:

1. What does the matrix $\mathbb{X}^T\mathbb{X}$ looks like?
2. When is it invertible?
3. What if it is not invertible?
4. We used convex optimization techniques without verifying if the results are really applicable. So, what is convex optimization and what is the mathematics behind it?
5. We found the solution of the linear regression or ordinary least square problem using calculus or convex optimization techniques. But, do we really understand the solution? How to interpret the results?
6. Are there other linear predictors apart from the one obtained using the linear regression or ordinary least square techniques?

Understanding the Linear Regression Solution

We should ask the following questions (**now with answers**):

1. What does the matrix $\mathbb{X}^T\mathbb{X}$ looks like? **Look at the $p = 2$ case.**
2. When is it invertible? **When columns of $\mathbb{X}$ are linearly independent. To be discussed later.**
3. What if it is not invertible? **Use other linear predictors or ML techniques. For example, use Ridge Regression.**
4. We used convex optimization techniques without verifying if the results are really applicable. So, what is convex optimization and what is the mathematics behind it? **We have seen the figures and intuition. We just have to express them in math. We will cover this next week.**
5. We found the solution of linear regression using convex optimization techniques. But, do we really understand the solution? How to interpret the results? **Keyword is projection: OLS solution is the projection of $\mathbb{Y}$ on the column space of $\mathbb{X}$. We will cover projects in a couple of weeks.**
6. Are there other linear predictors apart from the one obtained using the linear regression or ordinary least square techniques? **Of course, we will discuss RIDGE, LASSO, and ELASTIC NET**

Ridge Regression

- **Linear regression**:

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

$$= \min_{\beta \in \mathbb{R}^p} \frac{1}{n} \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} \right).$$

- **Ridge regression**: Fix $\lambda > 0$, remove scaling by $\frac{1}{n}$ and solve

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

$$= \min_{\beta \in \mathbb{R}^p} (\mathbb{Y} - \mathbb{X}\beta)^T (\mathbb{Y} - \mathbb{X}\beta) + \lambda \beta^T \beta$$

$$= \min_{\beta \in \mathbb{R}^p} \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} + \lambda \beta^T \beta \right).$$

Thus, in ridge regression there is an extra term. If we set $\lambda = 0$, then ridge regression reduces to linear regression.

- **Gradient**: The gradient is

$$\begin{aligned}
&\nabla_\beta \left(\mathbb{Y}^T \mathbb{Y} + \beta^T \mathbb{X}^T \mathbb{X} \beta - 2\beta^T \mathbb{X}^T \mathbb{Y} + \lambda \beta^T \beta \right) \\
&= 2\mathbb{X}^T \mathbb{X} \beta - 2\mathbb{X}^T \mathbb{Y} + 2\lambda\beta \\
&= 2\left(\mathbb{X}^T \mathbb{X} + \lambda I_p \right) \beta - 2\mathbb{X}^T \mathbb{Y}.
\end{aligned}$$

- **Ridge regression solution**: Setting gradient equal to zero we get

$$\left(\mathbb{X}^T \mathbb{X} + \lambda I_p \right) \beta = \mathbb{X}^T \mathbb{Y}.$$

  Thus, the Ridge regression coefficients are

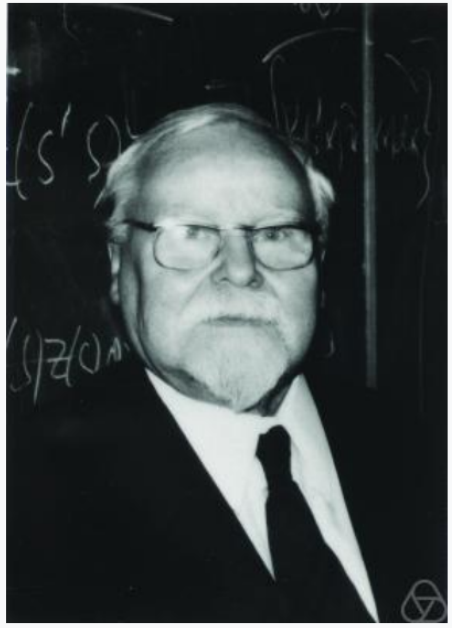
$$\beta_{\text{ridge}} = \left(\mathbb{X}^T \mathbb{X} + \lambda I_p \right)^{-1} \mathbb{X}^T \mathbb{Y}.$$

  It turns out that the matrix $\left(\mathbb{X}^T \mathbb{X} + \lambda I_p \right)$ it is always invertible! We will come back to Convexity, Optimization, Matrices and their Invertibility next week.

- **How to select λ?** Note that the optimal solution depends on the choice of λ. But, how do we select it? The answer is that we optimize over λ. We will discuss how to achieve this optimization when we discuss an data example in Python.

Ridge is also known as Tikhonov Regularization

Andrey Tikhonov



Born	17 October 1906 Gzhatsk, Russian Empire
Died	October 7, 1993 (aged 86) Moscow, Russia
Nationality	Russian
Alma mater	Moscow State University
Known for	Important contributions to topology, functional analysis, mathematical physics, ill-posed problems; Tychonoff spaces, Tychonoff's theorem, Tikhonov regularization, Tikhonov's theorem (dynamical systems), magnetotellurics geophysical method.

Why Ridge?

- Published in a Russian journal in the 1950-1960 (I do not know the exact citation).
- **Invertibility**: Originally introduced to guarantee existence of solution in case $\mathbb{X}^T\mathbb{X}$ is not invertible. (*idea popular in Signal Processing community*)
- **Regularization**: In modern ML, it is also seen as a way to execute Occam's razor: keep things simple. We can view λ as a penalty on the complexity of the solution β where the complexity is measured using $\beta^T\beta$. (*idea popular in Machine Learning community*)
- **Shrinkage**: It can be argued and shown in special cases that the coefficients β_{ridge} are shrunken version of β_{ols}. We will also see this in a Python example. The idea of shrinkage is one of the most profound ideas coming out of high-dimensional statistics. Shrinking your coefficients often leads to stability (and hence better prediction error). It is not at all intuitive why this result is true. Such an idea can only come out of rigorous math. (*Idea popular in Statistics community*)
- **Choice of** λ: While shrinkage of coefficients or regularization (penalizing size of β) can and often lead to better overall prediction performance, it is not clear how much shrinkage or regulalization is needed to get the best generalization error.

LASSO

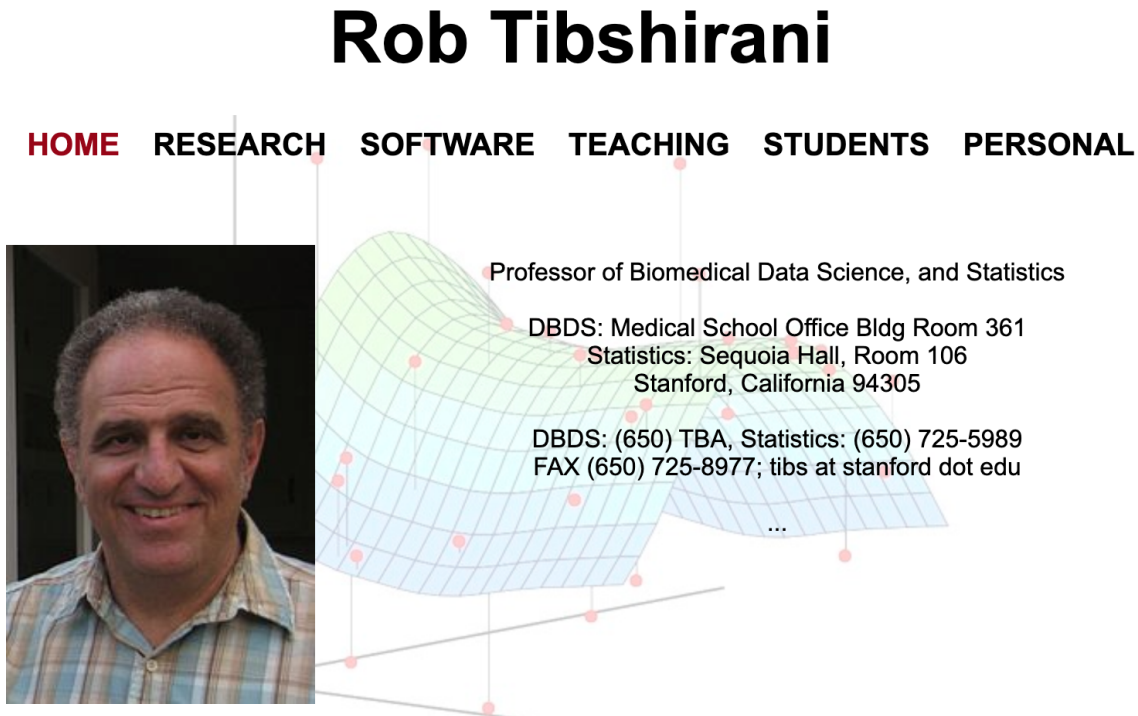
- **LASSO**: It stands for **least absolute shrinkage and selection operator**.
- **The LASSO problem**: In this we solve for $\lambda > 0$

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

  Compare this with
  - **Linear Regression**: $\min_{\beta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2$.
  - **Ridge Regression**: $\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 + \lambda \sum_{i=1}^{p} \beta_i^2$.

  Thus, the penalty is on the magnitude of the components of β.
- **LASSO solution**:
  - Except special cases, the optimal solution cannot be written in a closed form. One of the main reason being that the objective is **not differentiable**.
  - **LASSO SOLUTION IS OBTAINED USING NUMERICAL TECHNIQUES.**
  - **GOOD NEWS: LASSO OBJECTIVE FUNCTION IS CONVEX.** This ensures a unique minimum.

LASSO is due to Rob Tibshirani



- The original lasso paper:
  Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B., Vol. 58, No. 1, pages 267-288.

Shrinkage and Thresholding

- **Ridge solution is approximately shrinkage of OLS**:

$$\beta_{ridge} \approx \frac{\beta_{ols}}{1+\lambda}$$

  Note that this is exact only under special circumstances (to be done in a HW). Here, we use it only as a guide to understand the ridge solution.

- **LASSO solution is approximately soft-thresholding of OLS**:

$$\beta_{lasso,i} \approx \begin{cases} \beta_{ols,i} - \lambda, & \text{if } \beta_{ols,i} > \lambda \\ 0, & \text{if } |\beta_{ols,i}| < \lambda \\ \beta_{ols,i} + \lambda, & \text{if } \beta_{ols,i} < -\lambda \end{cases}$$

  Again, note that this is exact only under special circumstances. Here, we use it only as a guide to understand the LASSO solution.

- **Variable selection**: Note that Ridge solution shrinks the linear regression coefficients towards zero. But, the LASSO solution abruptly sets some of coefficients to zero. Thus, the LASSO solution has the added advantage that it helps with variable selection. That is, it helps us select those components of the x variable that are the most relevant for predicting the variable y. The parameter λ controls the variable selection and the optimal variable selection is obtained by optimizing over λ.

Elastic Net

- **Why not always use LASSO?** If LASSO can help us with variable selection, it seems we should always use LASSO. But, it has limitations.
- **Limitations of LASSO**:
  - **Limitations to variable selection**: If we have p components of x and n data points with $p \gg n$ (real case), then the LASSO selects at most n variables. We will understand the reasons behind this later.
  - **Dependent components**: If a group of components of x are highly correlated, LASSO often selects only one from the group. This is not ideal as we can improve the prediction by using all the variables in the group.
- **Elastic Net**: If you have dependent variables AND you want variable selection, the recommended approach is to use something called Elastic Net:

$$\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 + \lambda_1 \sum_{i=1}^{p} \beta_i^2 + \lambda_2 \sum_{i=1}^{p} |\beta_i|.$$

  Here we are combining magnitude as well as square penalty for the components of β. Because of the LASSO term here, the objective function is not differentiable and solution is obtained using numerical techniques.
- **Original paper**: Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. Journal of the royal statistical society: series B (statistical methodology), 67(2), 301-320.
- **Should we always use Elastic Net?** Note that you now have two additional parameters to train or optimize. This may lead to overfitting.