CMPSC 448: Machine Learning

Lecture 5. Linear Regression & Ordinary Least Squares

Rui Zhang
Fall 2021

PennState

Outline

- Linear regression: scalar and multivariate
- Ordinary Least Squares (OLS) and its (closed-form) solution
- Existence and uniqueness of OLS solution

- Computational issues of closed-form OLS solution
- GD and SGD for OLS

- A geometric interpretation and hat matrix

- A statistical interpretation and MLE estimator

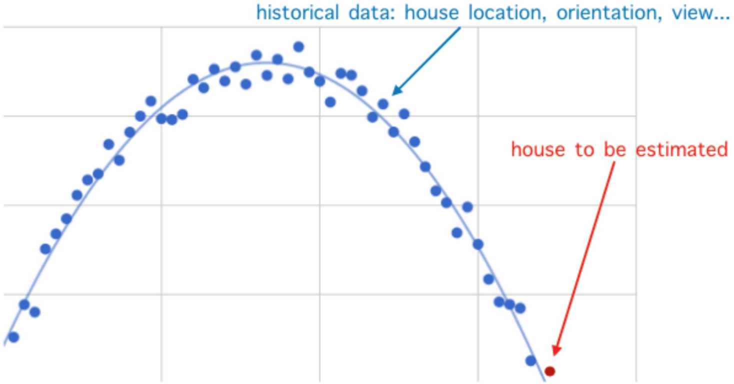
Next lecture: regularization for linear regression: Ridge and Lasso

Supervised learning

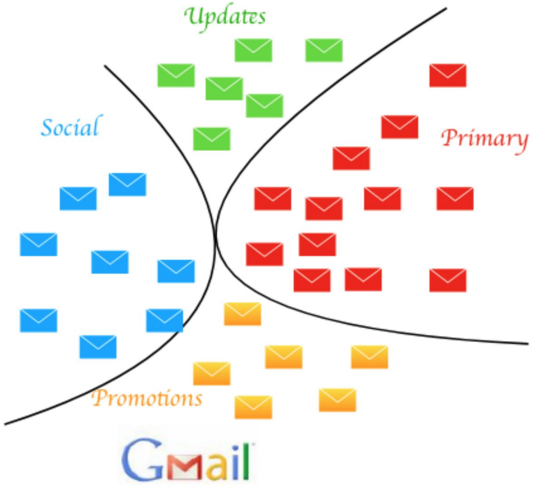
- In supervised learning, the learner gets to learn from **examples with their true labels**.
- The learner then needs to devise a **prediction rule**.
- The prediction rule will be used to **predict the labels of future examples**.
- The **success of the prediction rule** is measured by *how accurately it predicts the labels of future examples*.

Regression vs Classification

- Tracking a mortgage
- Houses data
- Banks: Estimate the loan based on location, orientation, view, etc
- Output values: **continuous**



- Spam classification
- Incoming emails
- How to group email in categories
- Output values: **discrete**, **categorical**



Regression vs Classification

- Ordinary Least Squares
- Ridge Regression
- Lasso Regression

Regression vs Classification

- Ordinary Least Squares
- Ridge Regression
- Lasso Regression

- Nearest Neighbor
- Artificial Neural Networks: Perceptron and Deep Neural Networks
- Logistic Regression
- Decision Trees
- Support Vector Machines (SVMs)
- Bagging: Random Forests
- Boosting: AdaBoost and Gradient Boosted Trees

Regression vs Classification

Regression

- Ordinary Least Squares
- Ridge Regression
- Lasso Regression

Classification

- Nearest Neighbor
- Artificial Neural Networks: Perceptron and Deep Neural Networks
- Logistic Regression
- Decision Trees
- Support Vector Machines (SVMs)
- Bagging: Random Forests
- Boosting: AdaBoost and Gradient Boosted Trees

- Algorithms covered in this course!

- There exist regression versions of classification methods, e.g., Support Vector Regression (SVR), Gradient Boosted Trees, DNNs, etc

- Today: one of the oldest, but still most popular/important methods: **Linear regression based on squared error**

Regression

The goal of regression is to make **quantitative (real valued) predictions** on the basis of a (vector of) **features or attributes**

Regression

The goal of regression is to make **quantitative (real valued) predictions** on the basis of a (vector of) **features or attributes**

- Does number of **lung cancer deaths** change with **number of cigarettes**?

Regression

The goal of regression is to make **quantitative (real valued) predictions** on the basis of a (vector of) **features or attributes**

- Does number of **lung cancer deaths** change with **number of cigarettes**?
- Does number of **skin cancer deaths** change with **latitude**?

Regression

The goal of regression is to make **quantitative (real valued) predictions** on the basis of a (vector of) **features or attributes**

- Does number of **lung cancer deaths** change with **number of cigarettes**?
- Does number of **skin cancer deaths** change with **latitude**?
- Does number of **gun deaths** change with **gun ownership**?

Regression

The goal of regression is to make **quantitative (real valued) predictions** on the basis of a (vector of) **features or attributes**

- Does number of **lung cancer deaths** change with **number of cigarettes**?
- Does number of **skin cancer deaths** change with **latitude**?
- Does number of **gun deaths** change with **gun ownership**?
- Can we predict the **price of a stock**?

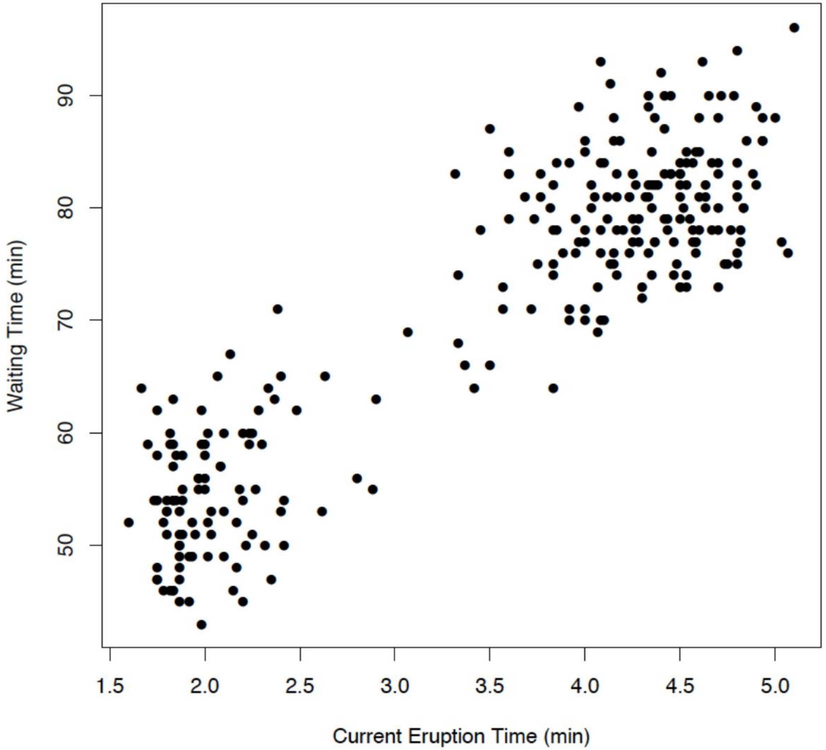
Example: Old faithful at Yellowstone

Old Faithful erupts every 35-120 minutes for 1.5-5 minutes to a height of 90-184 feet. Visitors to the park try to arrive at the geyser site to see it erupt without waiting too long

Waiting time between eruptions seems to be related to duration of previous eruption.

Example: Old faithful at Yellowstone

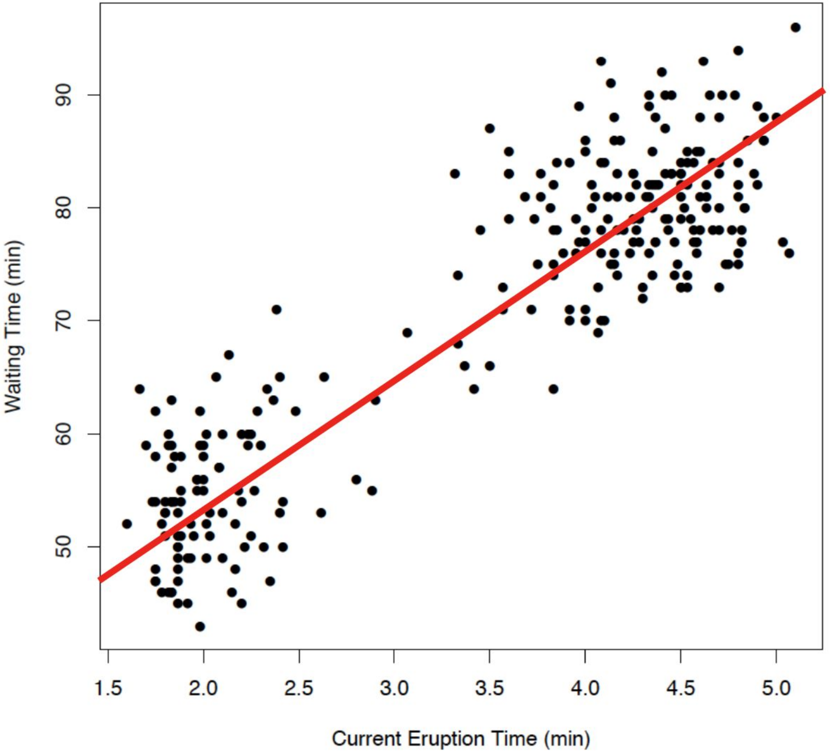
Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.



The 222 interruption times taken during August 1978 and August 1979

Example: Old faithful at Yellowstone

Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.

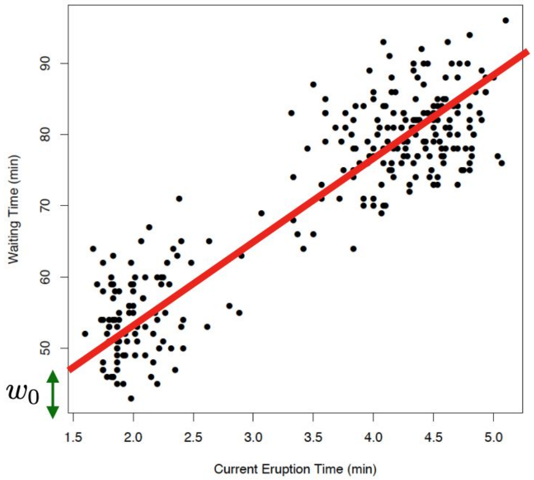


Linear regression in 1 dimension

Assume we only have 1 feature (d = 1)

Our linear model, a line with parameters $[w_0, w_1] \in \mathbb{R}^2$ in 1 dimension is given by:

$$\text{(wait time)} = w_0 + {\color{red}w_1} \times \text{(last duration)} + \text{(error)}$$



$$\underbrace{y}_{\text{response variable}} = \underbrace{w_0}_{\text{intercept (bias)}} + \underbrace{w_1}_{\text{slope}}\underbrace{x}_{\text{covariate}} + \underbrace{\epsilon}_{\text{noise}}$$

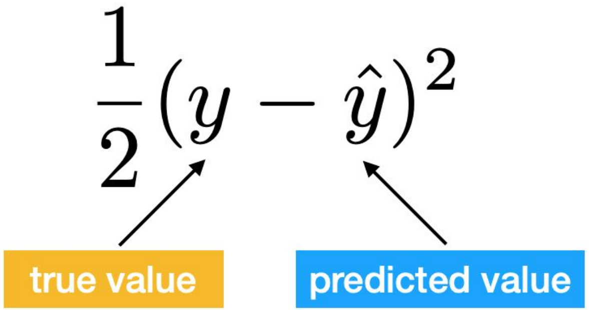
What should count as a good regression?

Let’s assume you already have a guess for the parameters of the linear model, say (w_0, w_1)
The prediction of your model on an input covariate x would be $\hat{y} = w_0 + w_1 x$
How should we measure the performance of a regression model?

What should count as a good regression?

Let’s assume you already have a guess for the parameters of the linear model, say (w_0, w_1)
The prediction of your model on an input covariate x would be $\hat{y} = w_0 + w_1 x$
How should we measure the performance of a regression model?

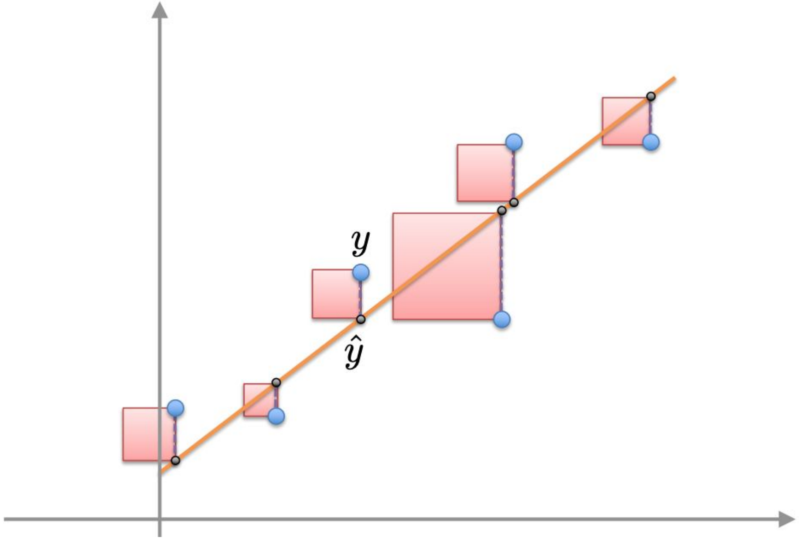
A widely used performance measure involves the **squared loss function**:



$$\frac{1}{2}(y - \hat{y})^2$$

true value

predicted value

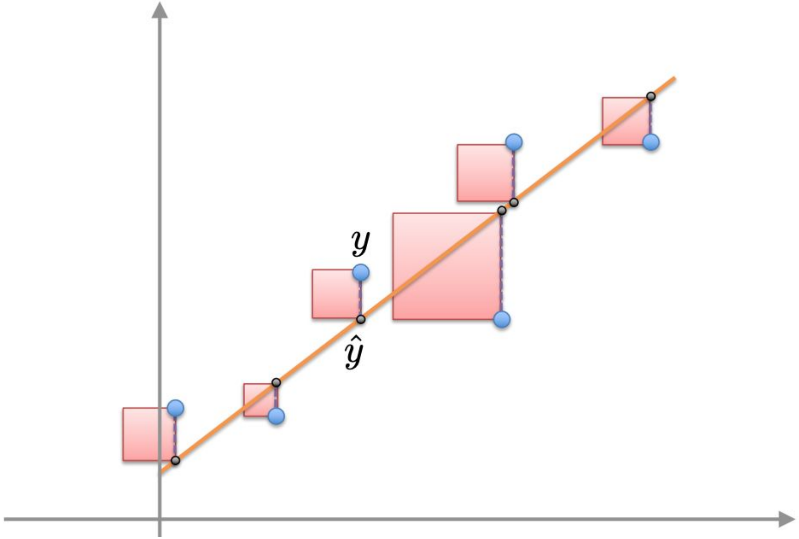


What should count as a good regression?

Let’s assume you already have a guess for the parameters of the linear model, say (w_0, w_1)
The prediction of your model on an input covariate x would be $\hat{y} = w_0 + w_1 x$
How should we measure the performance of a regression model?

A widely used performance measure involves the **squared loss function**:

$$\frac{1}{2}(\underset{\text{true value}}{y} - \underset{\text{predicted value}}{\hat{y}})^2$$



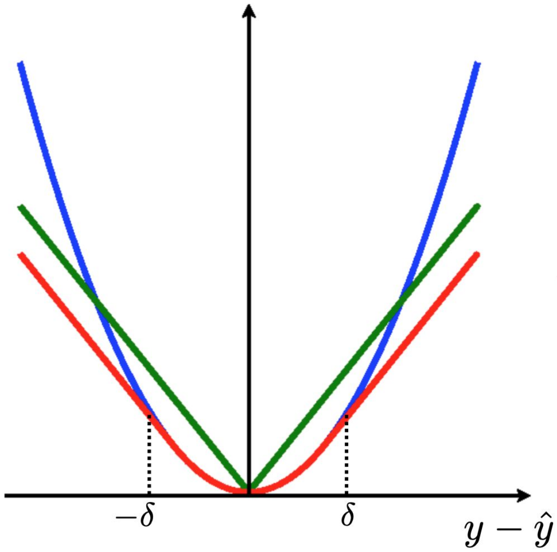
There are some justifications for this choice (a probabilistic interpretation is coming later in the lecture)

But usually, it is done because it is easy to minimize.

Is squared error the only option?

A widely used performance measure involves the **squared loss function**:

$$\ell(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$



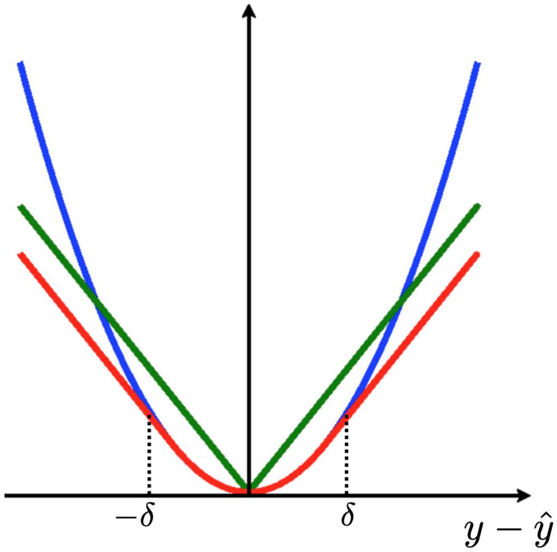
Is squared error the only option?

A widely used performance measure involves the **squared loss function**:

$$\ell(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

Another alternative is to use **absolute error loss**:

$$\ell(y, \hat{y}) = |y - \hat{y}|$$



Is squared error the only option?

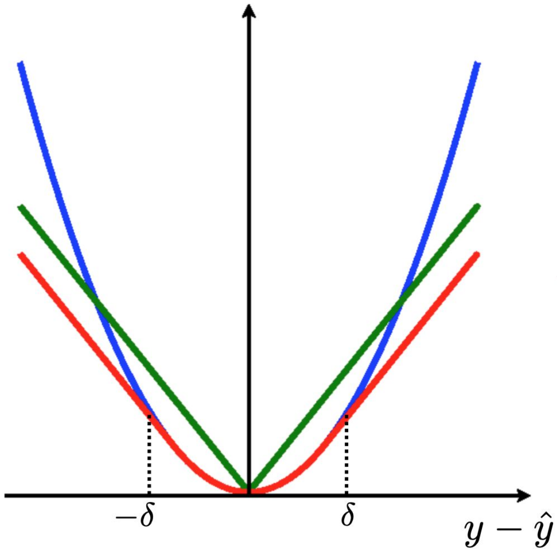
A widely used performance measure involves the **squared loss function**:

$$\ell(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

Another alternative is to use **absolute error loss**:

$$\ell(y, \hat{y}) = |y - \hat{y}|$$

Yet another option is to use **Huber loss**:



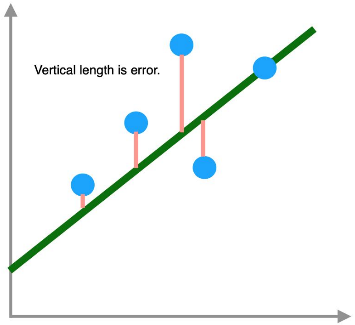
$$\ell(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \leq \delta, \\ \delta(|y - \hat{y}| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$$

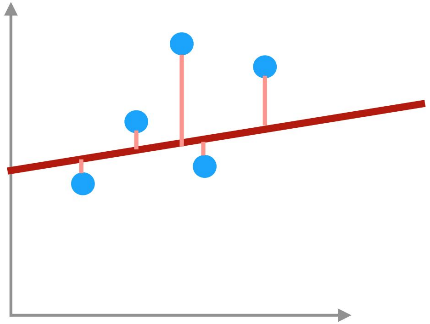
Least Squares

The **training error** (sum of squared errors) is:

$$f(w_0, w_1) = \sum_{i=1}^{n} \frac{1}{2}(y_i - \hat{y}_i)^2$$
$$= \sum_{i=1}^{n} \frac{1}{2}(y_i - w_0 - w_1 x_i)^2$$

We need to find parameters of a linear model that minimizes training error.





If the error is small, our predictions are close to the input targets (how it is going to perform on test data, i.e., generalization error, is another story)

Ordinary least squares solution

The objective is convex, so we can set the gradient to zero to obtain the optimal solution:

The optimal solution of one dimensional regression becomes as:

Ordinary least squares solution

The objective is convex, so we can set the gradient to zero to obtain the optimal solution:

The optimal solution of one dimensional regression becomes as:

$$\frac{\partial f}{\partial w_0} = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)(-1) = 0$$

$$\rightarrow \quad \sum_{i=1}^{n} w_0 = \sum_{i=1}^{n} y_i - w_1 x_i$$

$$\rightarrow \quad w_0 = \frac{\sum_{i=1}^{n} y_i - w_1 x_i}{n}$$

Ordinary least squares solution

The objective is convex, so we can set the gradient to zero to obtain the optimal solution:

The optimal solution of one dimensional regression becomes as:

$$\frac{\partial f}{\partial w_0} = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)(-1) = 0$$

$$\rightarrow \quad \sum_{i=1}^{n} w_0 = \sum_{i=1}^{n} y_i - w_1 x_i$$

$$\rightarrow \quad w_0 = \frac{\sum_{i=1}^{n} y_i - w_1 x_i}{n}$$

$$\frac{\partial f}{\partial w_1} = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)(-x_i) = 0$$

$$\rightarrow \quad w_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} (y_i - w_0) x_i$$

$$\rightarrow \quad w_1 = \frac{\sum_{i=1}^{n} (y_i - w_0) x_i}{\sum_{i=1}^{n} x_i^2}$$

Ordinary least squares solution

The objective is convex, so we can set the gradient to zero to obtain the optimal solution:
The optimal solution of one dimensional regression becomes as:

$$\frac{\partial f}{\partial w_0} = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)(-1) = 0$$

$$\rightarrow \quad \sum_{i=1}^{n} w_0 = \sum_{i=1}^{n} y_i - w_1 x_i$$

$$\rightarrow \quad w_0 = \frac{\sum_{i=1}^{n} y_i - w_1 x_i}{n}$$

$$\frac{\partial f}{\partial w_1} = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)(-x_i) = 0$$

$$\rightarrow \quad w_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} (y_i - w_0) x_i$$

$$\rightarrow \quad w_1 = \frac{\sum_{i=1}^{n} (y_i - w_0) x_i}{\sum_{i=1}^{n} x_i^2}$$

This is a **closed-form** solution!
Recall in last lecture we used iterative optimization methods, such as gradient descent (GD), to find the optimal solution.

Multivariate linear regression

In many applications the number of covariates that might affect the outcome is more than one.

- Smoking is not the only contributor to lung cancer, e.g., environmental factors like exposure to asbestos
- Zillow's Zestimate home valuation competition ($1M prize) has about 65 features!

Featured Prediction Competition

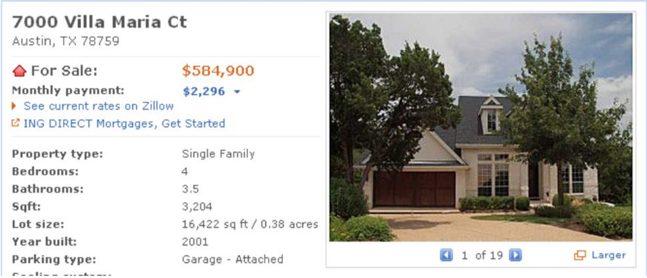
Zillow Prize: Zillow's Home Value Prediction (Zestimate)

Can you improve the algorithm that changed the world of real estate?

$1,200,000

Prize Money

Zillow · 3,779 teams · 9 months ago



7000 Villa Maria Ct

Austin, TX 78759

For Sale:	$584,900
Monthly payment:	$2,296

See current rates on Zillow

ING DIRECT Mortgages, Get Started

Property type:	Single Family
Bedrooms:	4
Bathrooms:	3.5
Sqft:	3,204
Lot size:	16,422 sq ft / 0.38 acres
Year built:	2001
Parking type:	Garage - Attached

In some other applications the number of covariates (**features**) might go up to millions!

How can we model the combined effect of multiple covariates (features) on output (e.g., the joint effect of smoking and asbestos on lung cancer)?

Multivariate Regression

Let’s assume the number of covariates is d

The input attributes are given as fixed length vectors, $\boldsymbol{x} = [x_1, x_2, \ldots, x_d]^\top \in \mathbb{R}^d$

A generic multivariate regression problem

Let's assume the number of covariates is d

The input attributes are given as fixed length vectors, $\boldsymbol{x} = [x_1, x_2, \ldots, x_d]^\top \in \mathbb{R}^d$ where each component maybe discrete or real valued.

The outputs are assumed to be real valued $y \in \mathbb{R}$ (the values of actual outputs such as prices may be more restricted)

We have access to a set of n training examples, $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$ sampled independently at random from some fixed but unknown distribution.

The goal is to minimize the **prediction error/loss** on new examples drawn at random from the same unknown distribution

Multivariate Regression

A linear multivariate predictor in d dimensions can be represented as a vector:

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^\top \in \mathbb{R}^d$$

where w_i shows the importance of i-th covariate.

Multivariate Regression

A linear multivariate predictor in d dimensions can be represented as a vector:

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^\top \in \mathbb{R}^d$$

where w_i shows the importance of i-th covariate.

The prediction of a multivariate input based on a linear model becomes:

$$\boldsymbol{w}^\top \boldsymbol{x} = \sum_{i=1}^{d} w_i x_i$$

Multivariate Regression

A linear multivariate predictor in d dimensions can be represented as a vector:

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^\top \in \mathbb{R}^d$$

where w_i shows the importance of i-th covariate.

The prediction of a multivariate input based on a linear model becomes:

$$\boldsymbol{w}^\top \boldsymbol{x} = \sum_{i=1}^{d} w_i x_i$$

Similar to one dimensional setting, we can use the squared loss to measure the discrepancy between actual response and predicted response:

$$\frac{1}{2}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2$$

Multivariate Regression

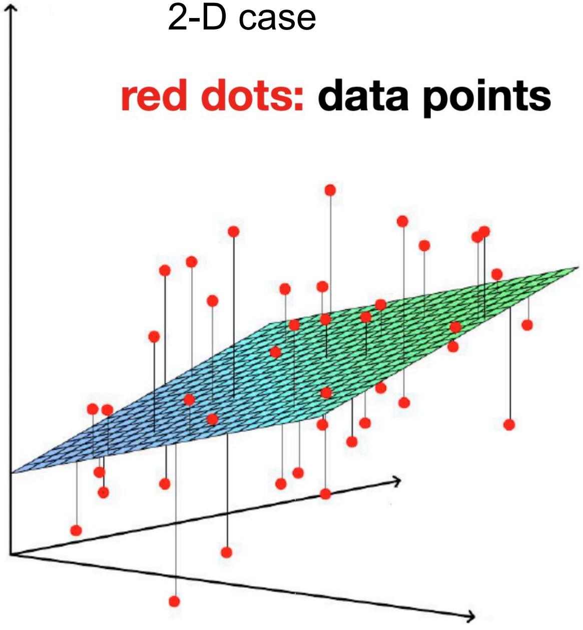
A linear multivariate predictor in d dimensions can be represented as a vector:

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^\top \in \mathbb{R}^d$$

where w_i shows the importance of i-th covariate.

2-D case

red dots: data points



The prediction of a multivariate input based on a linear model becomes:

$$\boldsymbol{w}^\top \boldsymbol{x} = \sum_{i=1}^{d} w_i x_i$$

Similar to one dimensional setting, we can use the squared loss to measure the discrepancy between actual response and predicted response:

$$\frac{1}{2}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2$$

Ordinary Least Squares (OLS)

The regression problem when solved with a linear model and squared loss as measure is called **Ordinary Least Squares (OLS)** regression.

$$\begin{aligned} &\frac{1}{2}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \\ &= \frac{1}{2}\sum_{i=1}^{n}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2 \end{aligned}$$

Ordinary Least Squares (OLS)

The regression problem when solved with a linear model and squared loss as measure is called **Ordinary Least Squares (OLS)** regression.

$$
\begin{aligned}
&\frac{1}{2}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \\
&= \frac{1}{2}\sum_{i=1}^{n}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2
\end{aligned}
$$

We can get rid of intercept (*b*) using a simple trick: $\boldsymbol{w}^\top \boldsymbol{x} + b = [\boldsymbol{w}, b]^\top [\boldsymbol{x}, 1]$

i.e., Just add a feature of value 1 to every data point and increase the number of parameters by one!

Ordinary Least Squares (OLS)

The regression problem when solved with a linear model and squared loss as measure is called **Ordinary Least Squares (OLS)** regression.

$$\frac{1}{2}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$
$$= \frac{1}{2}\sum_{i=1}^{n}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2$$

We can get rid of intercept (*b*) using a simple trick: $\boldsymbol{w}^\top \boldsymbol{x} + b = [\boldsymbol{w}, b]^\top [\boldsymbol{x}, 1]$

i.e., Just add a feature of value 1 to every data point and increase the number of parameters by one!

So, we can simplify the objective as follows:

$$f_{\text{OLS}}(\boldsymbol{w}) = \frac{1}{2}\sum_{I=1}^{n}(\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)^2$$

The OLS solution

The goal is to solve the following d dimensional **convex optimization** problem:

$$\boldsymbol{w}^*_{\texttt{OLS}} = \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left[f_{\texttt{OLS}}(\boldsymbol{w}) \equiv \frac{1}{2}\sum_{i=1}^{n} (\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)^2 \right]$$

The OLS solution

The goal is to solve the following d dimensional **convex optimization** problem:

$$\boldsymbol{w}^*_{\texttt{OLS}} = \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left[f_{\texttt{OLS}}(\boldsymbol{w}) \equiv \frac{1}{2}\sum_{i=1}^{n} (\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)^2 \right]$$

We take the derivative w.r.t. to unknown vector $\boldsymbol{w}$ and set it to zero:

$$\frac{f_{\texttt{OLS}}(\boldsymbol{w})}{\partial \boldsymbol{w}} =$$

The OLS solution

The goal is to solve the following d dimensional **convex optimization** problem:

$$\boldsymbol{w}^*_{\texttt{OLS}} = \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left[f_{\texttt{OLS}}(\boldsymbol{w}) \equiv \frac{1}{2}\sum_{i=1}^{n} (\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)^2 \right]$$

We take the derivative w.r.t. to unknown vector $\boldsymbol{w}$ and set it to zero:

$$\begin{aligned}
\frac{f_{\texttt{OLS}}(\boldsymbol{w})}{\partial \boldsymbol{w}} &= \sum_{i=1}^{n} (\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)\boldsymbol{x}_i \\
&= \sum_{i=1}^{n} \left(\boldsymbol{w}^\top \boldsymbol{x}_i\right) \boldsymbol{x}_i - \sum_{i=1}^{n} y_i \boldsymbol{x}_i \\
&= \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{w} - \sum_{i=1}^{n} y_i \boldsymbol{x}_i \\
&= \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right) \boldsymbol{w} - \sum_{i=1}^{n} y_i \boldsymbol{x}_i = 0
\end{aligned}$$

The closed-form OLS solution

We have:

$$\left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right) \boldsymbol{w} - \sum_{i=1}^{n} y_i \boldsymbol{x}_i = 0$$

By rearranging the terms we have the optimal OLS solution as:

$$\boxed{\boldsymbol{w}_* = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right)^{-1} \sum_{i=1}^{n} y_i \boldsymbol{x}_i}$$

Be careful: you can not divide by a matrix, rather multiply both side by inverse of the matrix!

Computation

$$\boldsymbol{w}_* = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right)^{-1} \sum_{i=1}^{n} y_i \boldsymbol{x}_i$$

Computing above closed-form solution requires computing the inverse of a matrix (assuming it exists) of size $d \times d$

The time complexity is $O\left(d^3\right)$

Not feasible for high-dimensional problems!

Can we do better?

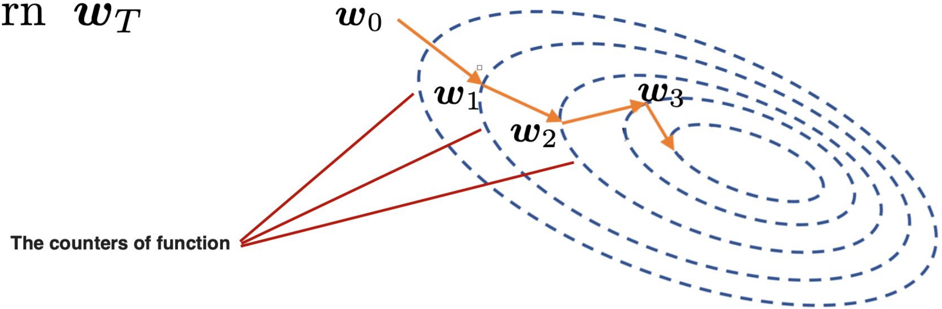
Gradient Descent (GD)

Our goal is to minimize the following differentiable convex function:

$$\text{minimize}_{\boldsymbol{w}\in\mathbb{R}^d} \quad f(\boldsymbol{w})$$

The gradient descent (**GD**) algorithm:

```
$\boldsymbol{w}_0 = \mathbf{0}$
for $t = 1, 2, \ldots, T$
    $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \nabla f(\boldsymbol{w}_t)$
end for
return $\boldsymbol{w}_T$
```

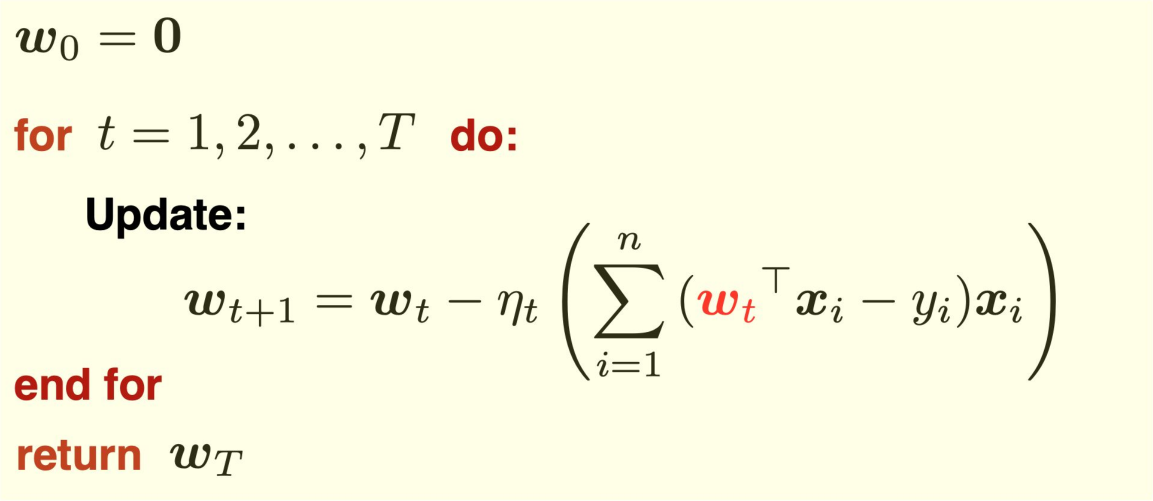


The GD traced back to Augustin-Louis Cauchy

Gradient Descent (GD) for OLS

$$\boldsymbol{w}^*_{\texttt{OLS}} = \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left[f_{\texttt{OLS}}(\boldsymbol{w}) \equiv \frac{1}{2}\sum_{i=1}^{n} (\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)^2 \right]$$

Let solve the optimization with GD:



```
$\boldsymbol{w}_0 = \mathbf{0}$
for $t = 1, 2, \ldots, T$ do:
    Update:
        $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \left( \sum_{i=1}^{n} (\boldsymbol{w}_t{}^\top \boldsymbol{x}_i - y_i) \boldsymbol{x}_i \right)$
end for
return $\boldsymbol{w}_T$
```

The time complexity is: $O\left(Tnd\right)$

Stochastic gradient descent (SGD) for OLS

Finding the solution with GD is more efficient than computing the closed-form solution!

But each iteration of GD requires computing n gradients! (not feasible when the number of training data is huge)

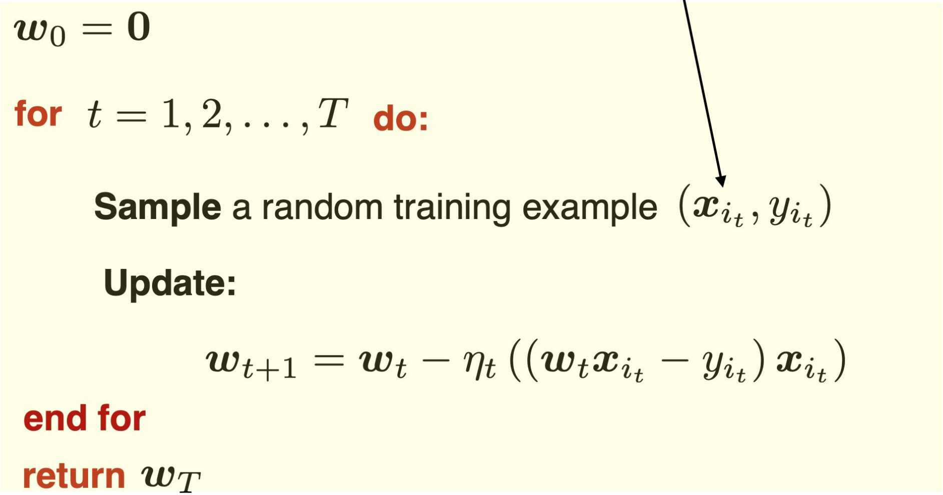
An idea: instead of computing gradient over all training data, randomly sample a small subset of training data uniformly at random (called **mini-batch**) and compute gradient just on those samples!

Stochastic gradient descent (SGD) for OLS

Let’s assume the mini-batch size is 1 for simplicity.

The SGD algorithm for OLS:

i_t : the index of example sampled at iteration t



```
$\boldsymbol{w}_0 = \mathbf{0}$
for $t = 1, 2, \ldots, T$ do:
    Sample a random training example $(\boldsymbol{x}_{i_t}, y_{i_t})$
    Update:
        $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \left( \left( \boldsymbol{w}_t \boldsymbol{x}_{i_t} - y_{i_t} \right) \boldsymbol{x}_{i_t} \right)$
end for
return $\boldsymbol{w}_T$
```

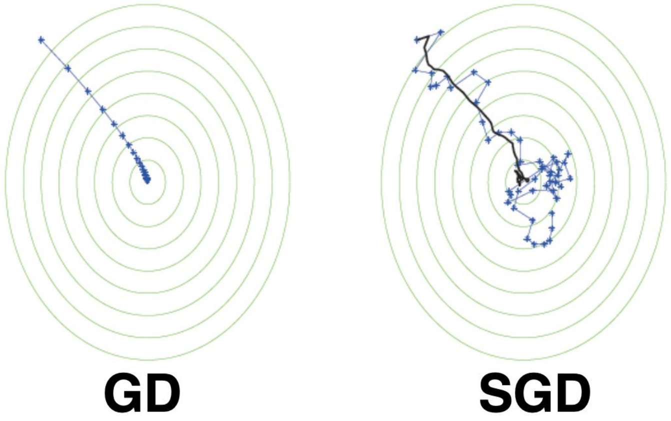
The time complexity is: $O\left(Td\right)$

GD versus SGD

SGD needs much more iterations than GD to converge!

SGD has cheaper iterations: O(1) versus O(n) gradient computations

With larger mini-batch sizes in SGD we can take advantages of both



Linear Regression: Matrix Notation

We can express the solution a bit more generally by resorting to a matrix notation

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}, \qquad \boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}, \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Linear Regression: Matrix Form

We can express the solution a bit more generally by resorting to a matrix notation

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}, \qquad \boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}, \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

So we have $f_{\mathrm{OLS}}(\boldsymbol{w}) = \frac{1}{2} \sum_{I=1}^{n} (\boldsymbol{w}^\top \boldsymbol{x}_i - y_i)^2$

$$= \frac{1}{2} \left\| \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \right\|_2^2$$

$$= \frac{1}{2} \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2$$

Recall

For vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, and matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ we have:

$$\boldsymbol{a}^\top \boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{a}$$

$$\|\boldsymbol{a}\|_2^2 = \boldsymbol{a}^\top \boldsymbol{a}$$

$$(\boldsymbol{A} + \boldsymbol{B})^\top = \boldsymbol{A}^\top + \boldsymbol{B}^\top$$

$$(\boldsymbol{A} + \boldsymbol{B})(\boldsymbol{A} + \boldsymbol{B}) = \boldsymbol{A}\boldsymbol{A} + \boldsymbol{B}\boldsymbol{A} + \boldsymbol{A}\boldsymbol{B} + \boldsymbol{B}\boldsymbol{B}$$

$$\boldsymbol{a}^\top \boldsymbol{A}\boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{A}^\top \boldsymbol{a}$$

Sanity check: always make sure the dimensions match; otherwise you did something wrong

Optimal solution: Matrix Form

$$\boldsymbol{a}^\top \boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{a}$$

$$\|\boldsymbol{a}\|_2^2 = \boldsymbol{a}^\top \boldsymbol{a}$$

$$(\boldsymbol{A} + \boldsymbol{B})^\top = \boldsymbol{A}^\top + \boldsymbol{B}^\top$$

$$(\boldsymbol{A} + \boldsymbol{B})(\boldsymbol{A} + \boldsymbol{B}) = \boldsymbol{A}\boldsymbol{A} + \boldsymbol{B}\boldsymbol{A} + \boldsymbol{A}\boldsymbol{B} + \boldsymbol{B}\boldsymbol{B}$$

$$\boldsymbol{a}^\top \boldsymbol{A} \boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{A}^\top \boldsymbol{a}$$

$$\begin{aligned} &\frac{1}{2}\left\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\right\|_2^2 \\ &= \frac{1}{2}\left(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\right)^\top \left(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\right) \end{aligned}$$

Sanity check the dimension of all multiplications match!

Optimal solution: Matrix Form

$$\boldsymbol{a}^\top \boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{a}$$
$$\|\boldsymbol{a}\|_2^2 = \boldsymbol{a}^\top \boldsymbol{a}$$
$$(\boldsymbol{A}+\boldsymbol{B})^\top = \boldsymbol{A}^\top + \boldsymbol{B}^\top$$
$$(\boldsymbol{A}+\boldsymbol{B})(\boldsymbol{A}+\boldsymbol{B}) = \boldsymbol{A}\boldsymbol{A} + \boldsymbol{B}\boldsymbol{A} + \boldsymbol{A}\boldsymbol{B} + \boldsymbol{B}\boldsymbol{B}$$
$$\boldsymbol{a}^\top \boldsymbol{A} \boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{A}^\top \boldsymbol{a}$$

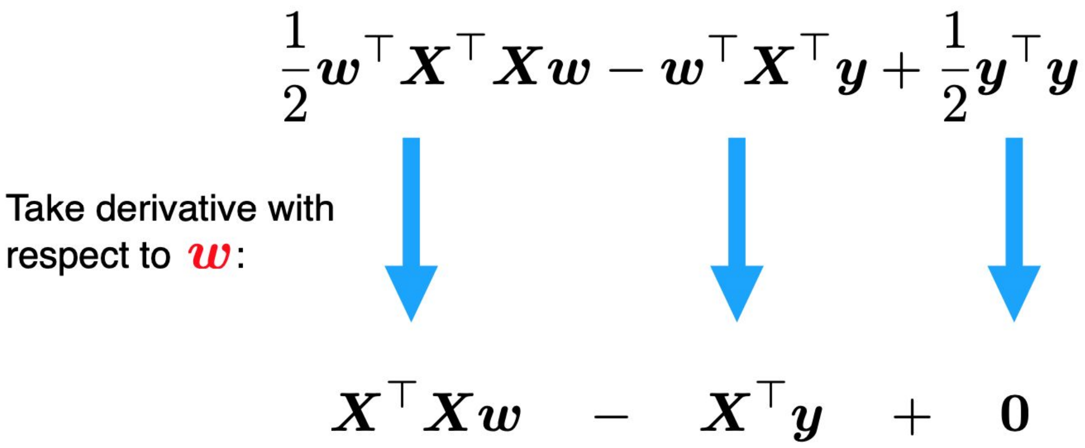
$$\begin{aligned}
&\frac{1}{2}\|\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}\|_2^2 \\
&= \frac{1}{2}(\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y})^\top(\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}) \\
&= \frac{1}{2}\left((\boldsymbol{X}\boldsymbol{w})^\top-\boldsymbol{y}^\top\right)(\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}) \\
&= \frac{1}{2}\left(\boldsymbol{w}^\top\boldsymbol{X}^\top-\boldsymbol{y}^\top\right)(\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}) \\
&= \frac{1}{2}\left(\boldsymbol{w}^\top\boldsymbol{X}^\top(\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y})-\boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y})\right) \\
&= \frac{1}{2}\left(\boldsymbol{w}^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{w}-\boldsymbol{w}^\top\boldsymbol{X}^\top\boldsymbol{y}-\boldsymbol{y}^\top\boldsymbol{X}\boldsymbol{w}+\boldsymbol{y}^\top\boldsymbol{y}\right) \\
&= \frac{1}{2}\boldsymbol{w}^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{w}-\boldsymbol{w}^\top\boldsymbol{X}^\top\boldsymbol{y}+\frac{1}{2}\boldsymbol{y}^\top\boldsymbol{y}
\end{aligned}$$

Sanity check the dimension of all multiplications match!

Optimal solution: Matrix Form

How to get the derivative?

Check HW1



$$\frac{1}{2}\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} - \boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^\top \boldsymbol{y}$$

Take derivative with respect to $\boldsymbol{w}$:

$$\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} \quad - \quad \boldsymbol{X}^\top \boldsymbol{y} \quad + \quad \boldsymbol{0}$$

Optimal solution: Matrix Form

$$\frac{1}{2}\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} - \boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^\top \boldsymbol{y}$$

How to get the derivative?
Check HW1

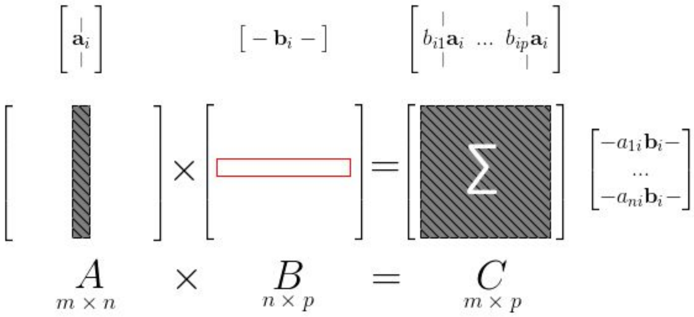
Take derivative with respect to $\boldsymbol{w}$:

$$\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} \quad - \quad \boldsymbol{X}^\top \boldsymbol{y} \quad + \quad \boldsymbol{0}$$

Setting derivative to zero gives:

$$\boldsymbol{w}_* = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right)^{-1} \sum_{i=1}^{n} y_i \boldsymbol{x}_i \quad \Longleftrightarrow \quad \boxed{\boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}}$$

Matrix Multiplication - different views



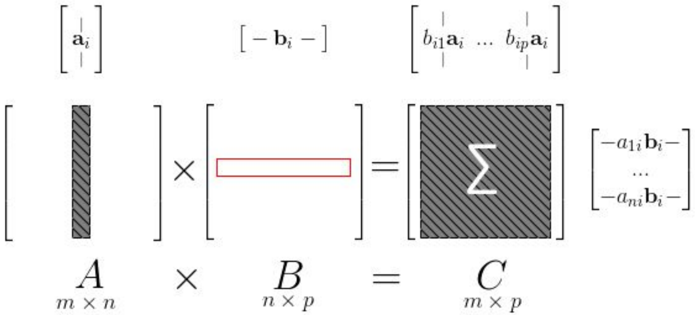
$\mathbf{a}_i$
$[-\mathbf{b}_i-]$
$[b_{i1}\mathbf{a}_i \dots b_{ip}\mathbf{a}_i]$
×
=
Σ
$-a_{1i}\mathbf{b}_i-$
$\dots$
$-a_{ni}\mathbf{b}_i-$
$\underset{m\times n}{A} \times \underset{n\times p}{B} = \underset{m\times p}{C}$

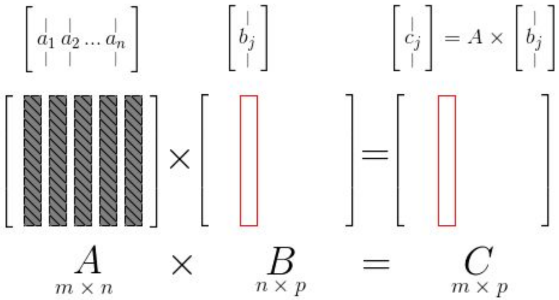
Sum of Outer Products $C = AB = \sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i^T$

$$\boldsymbol{w}_* = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right)^{-1} \sum_{i=1}^{n} y_i \boldsymbol{x}_i \quad \Longleftrightarrow \quad \boxed{\boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}}$$

http://mlwiki.org/index.php/Matrix-Matrix_Multiplication

Matrix Multiplication - different views





Sum of Outer Products $C = AB = \sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i^T$

Column at a Time

$$\boldsymbol{w}_* = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top \right)^{-1} \sum_{i=1}^{n} y_i \boldsymbol{x}_i \Longleftrightarrow \boxed{\boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X} \right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}}$$

http://mlwiki.org/index.php/Matrix-Matrix_Multiplication

OLS issues

- It assumes linear relationship between features and repossess (outputs) [real world relationships tend to be more complicated than simple lines or planes, meaning that even with an infinite number of training points (and hence perfect information about what the optimal choice of plane is) linear methods will often fail to do a good job at making predictions]

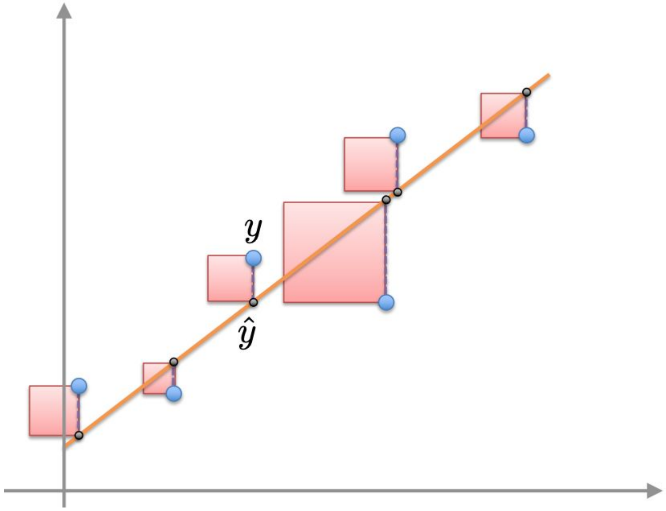
OLS issues

- It assumes linear relationship between features and repossess (outputs) [real world relationships tend to be more complicated than simple lines or planes, meaning that even with an infinite number of training points (and hence perfect information about what the optimal choice of plane is) linear methods will often fail to do a good job at making predictions]
- It always uses all features

OLS issues

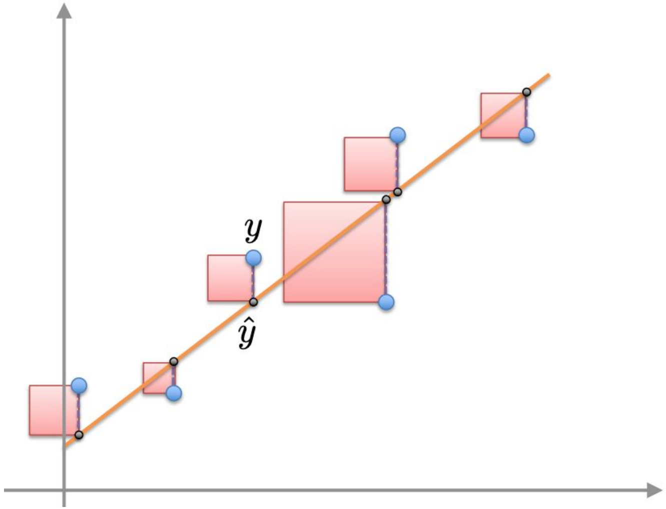
- It assumes linear relationship between features and repossess (outputs) [real world relationships tend to be more complicated than simple lines or planes, meaning that even with an infinite number of training points (and hence perfect information about what the optimal choice of plane is) linear methods will often fail to do a good job at making predictions]
- It always uses all features
- Data might be so big and we can not store

OLS issues



- It assumes linear relationship between features and repossess (outputs) [real world relationships tend to be more complicated than simple lines or planes, meaning that even with an infinite number of training points (and hence perfect information about what the optimal choice of plane is) linear methods will often fail to do a good job at making predictions]
- It always uses all features
- Data might be so big and we can not store
- It is sensitive to outliers [some points in the training data have excessively large or small values for the dependent variable compared to the rest of the training data]

OLS issues



- It assumes linear relationship between features and repossess (outputs) [real world relationships tend to be more complicated than simple lines or planes, meaning that even with an infinite number of training points (and hence perfect information about what the optimal choice of plane is) linear methods will often fail to do a good job at making predictions]
- It always uses all features
- Data might be so big and we can not store
- It is sensitive to outliers [some points in the training data have excessively large or small values for the dependent variable compared to the rest of the training data]
- Solution might not be unique [as soon as the number of features used exceeds the number of training data points, the least squares solution will not be unique]

Non-uniqueness of OLS solution

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$$

Why isn't solution unique?

- Imagine we have two features that are identical for all examples.

$$\hat{y}_i = w_0 + w_1 x_{i,1} + w_2 x_{i,1} = w_0 + (w_1 + w_2) x_{i,1} + 0 x_{i,1}$$

Thus, if $[w_0, w_1, w_2]$ is a solution, then $[w_0, w_1 + w_2, 0]$ is also a solution.

Non-uniqueness of OLS solution

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$$

Why isn't solution unique?

- Imagine we have two features that are identical for all examples.

$$\hat{y}_i = w_0 + w_1 x_{i,1} + w_2 x_{i,1} = w_0 + (w_1 + w_2) x_{i,1} + 0 x_{i,1}$$

Thus, if $[w_0, w_1, w_2]$ is a solution, then $[w_0, w_1 + w_2, 0]$ is also a solution.

- This is special case of features being "**collinear**": One feature is a linear function of another.
- I can increase weight on one feature, and decrease it on the other, without changing predictions.

But, any $\boldsymbol{w}$ where $\nabla f_{\text{OLS}}(\boldsymbol{w}) = 0$ is a global optimum, due to convexity.

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

Least squares solution is unique **if and only if** all columns of $\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$ are **linearly independent**.

- No column can be written as a “linear combination” of the others.

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

Least squares solution is unique **if and only if** all columns of $\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$ are **linearly independent**.

- No column can be written as a “linear combination” of the others.
- Many equivalent conditions: Since $\boldsymbol{X}\boldsymbol{v} = 0 \iff \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v} = 0$

$\boldsymbol{X}$ has “full column rank”, $\boldsymbol{X}^\top \boldsymbol{X}$ is invertible,

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$$

Least squares solution is unique **if and only if** all columns of $\boldsymbol{X}$ are **linearly independent**.

- No column can be written as a “linear combination” of the others.
- Many equivalent conditions: Since $\boldsymbol{X}\boldsymbol{v} = 0 \iff \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v} = 0$

$$\boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

$\boldsymbol{X}$ has “full column rank”, $\boldsymbol{X}^\top \boldsymbol{X}$ is invertible,

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$$

Least squares solution is unique **if and only if** all columns of $\boldsymbol{X}$ are **linearly independent**.

- No column can be written as a “linear combination” of the others.
- Many equivalent conditions: Since $\boldsymbol{X}\boldsymbol{v} = 0 \iff \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v} = 0$

$$\boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

$\boldsymbol{X}$ has “full column rank”, $\boldsymbol{X}^\top \boldsymbol{X}$ is invertible,

$$\det\left(\boldsymbol{X}^\top \boldsymbol{X}\right) > 0$$

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

Least squares solution is unique **if and only if** all columns of $\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$ are **linearly independent**.

- No column can be written as a "linear combination" of the others.
- Many equivalent conditions: Since $\boldsymbol{X}\boldsymbol{v} = 0 \iff \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v} = 0$ $\quad \boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$

$\boldsymbol{X}$ has "full column rank", $\boldsymbol{X}^\top \boldsymbol{X}$ is invertible, $\boldsymbol{X}^\top \boldsymbol{X}$ has non-zero eigenvalues, $\det\left(\boldsymbol{X}^\top \boldsymbol{X}\right) > 0$

$\boldsymbol{X}^\top \boldsymbol{X}$ is positive definite

When does OLS have a unique solution?

We said that least squares solution is not unique if we have repeated columns.

But there are other ways it could be non-unique:

- One column is a scaled version of another column.
- One column could be the sum of 2 other columns.
- One column could be three times one column minus four times another.

Least squares solution is unique **if and only if** all columns of $\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$ are **linearly independent**.

- No column can be written as a "linear combination" of the others.
- Many equivalent conditions: Since $\boldsymbol{X}\boldsymbol{v} = 0 \iff \boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{v} = 0$ $\quad \boldsymbol{w}_* = \left(\boldsymbol{X}^\top\boldsymbol{X}\right)^{-1}\boldsymbol{X}^\top\boldsymbol{y}$

$\boldsymbol{X}$ has "full column rank", $\boldsymbol{X}^\top\boldsymbol{X}$ is invertible, $\boldsymbol{X}^\top\boldsymbol{X}$ has non-zero eigenvalues, $\det\left(\boldsymbol{X}^\top\boldsymbol{X}\right) > 0$

$\boldsymbol{X}^\top\boldsymbol{X}$ is positive definite

Note that we cannot have independent columns if $d > n$ (why? recall the definition of rank, and rank(X) <= min(d,n))

The geometric interpretation

$$\mathbf{X} = \begin{array}{c} \begin{array}{cccc} \phi_1 & \phi_2 & & \phi_d \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \end{array}$$

The output of a matrix-vector multiplication is linear combination of columns of the matrix

$$\boldsymbol{X}\boldsymbol{w} = w_1\boldsymbol{\phi}_1 + w_2\boldsymbol{\phi}_2 + \ldots + w_d\boldsymbol{\phi}_d$$

The geometric interpretation

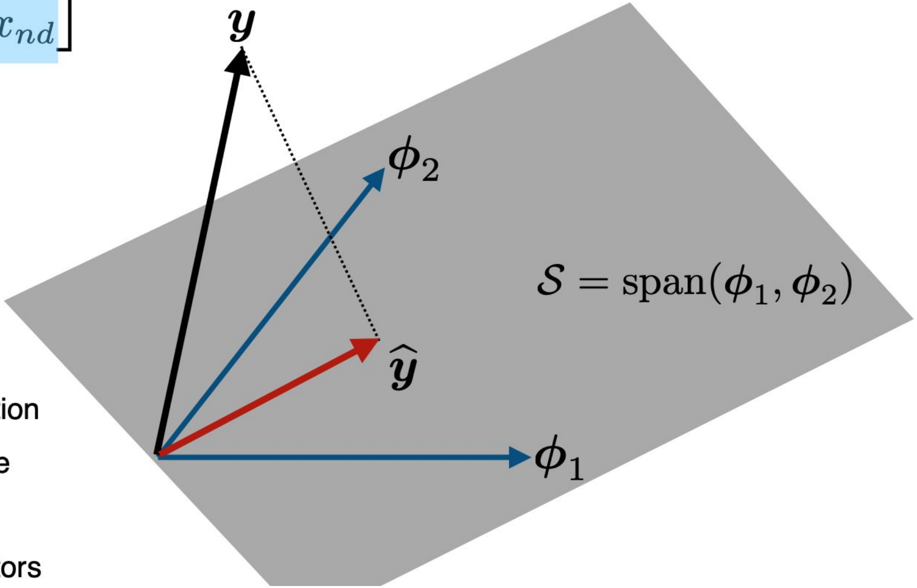
$$\mathbf{X} = \begin{array}{c} \begin{array}{cccc} \phi_1 & \phi_2 & & \phi_d \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \end{array}$$

The output of a matrix-vector multiplication is linear combination of columns of the matrix

$$\boldsymbol{X}\boldsymbol{w} = w_1\boldsymbol{\phi}_1 + w_2\boldsymbol{\phi}_2 + \ldots + w_d\boldsymbol{\phi}_d$$

$$\frac{1}{2}\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2$$

The solution is orthogonal projection of $\boldsymbol{y}$ to the space spanned by the column of data matrix $\boldsymbol{X}$, i.e., linear combination of feature vectors



$\boldsymbol{y}$
$\boldsymbol{\phi}_2$
$\mathcal{S} = \mathrm{span}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$
$\widehat{\boldsymbol{y}}$
$\boldsymbol{\phi}_1$

The hat matrix

The optimal solution is: $\boldsymbol{w}_* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$

The predicted labels' of training data are:

$$\widehat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{w}_* = \boldsymbol{X}\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

Define the following matrix:

$$\boldsymbol{H} = \boldsymbol{X}\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top$$

The matrix $\boldsymbol{H} \in \mathbb{R}^{n \times n}$ is called the HAT matrix, because it puts a hat on $\boldsymbol{y}$:

$$\widehat{\boldsymbol{y}} = \boldsymbol{H}\boldsymbol{y}$$

The difference between true outputs and predicted values are called residuals:

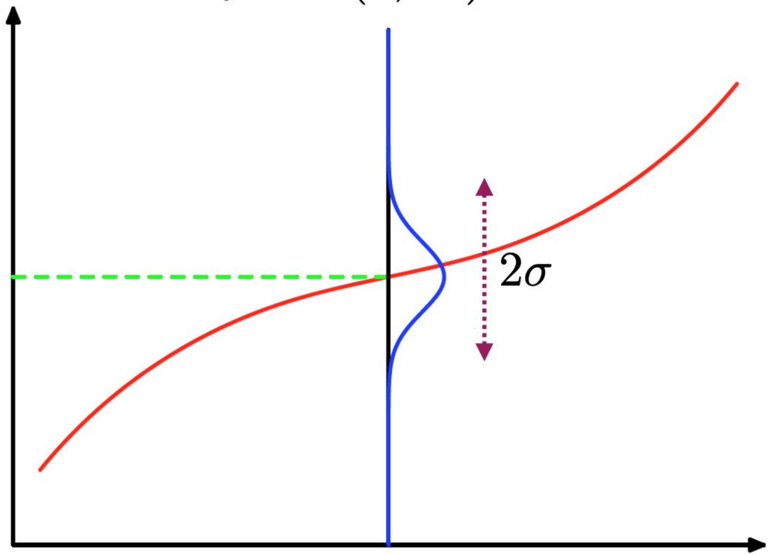
$$\boldsymbol{r} = \boldsymbol{y} - \widehat{\boldsymbol{y}} \in \mathbb{R}^n$$

Statistical estimation interpretation

We will now make a specific assumption on the conditional distribution of y given $\boldsymbol{x}$
Will see that estimating the parameters of that distribution from the training sample using maximum likelihood estimation (MLE)

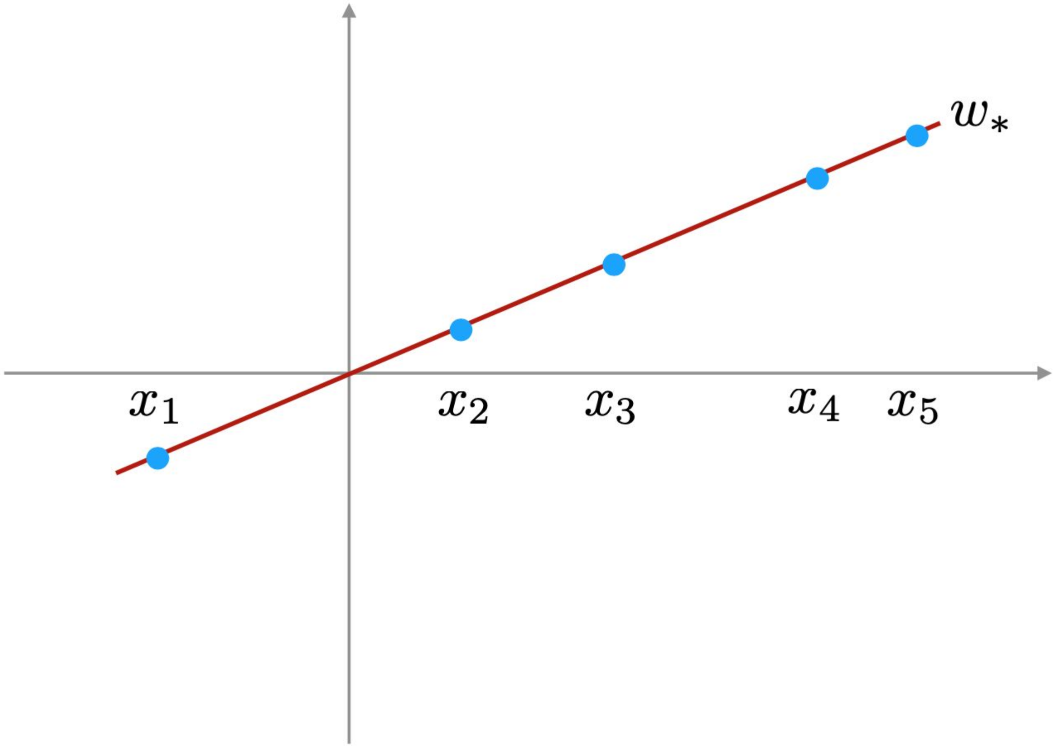
$$y_i = \boldsymbol{w}_*^\top \boldsymbol{x}_i + \epsilon_i$$

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

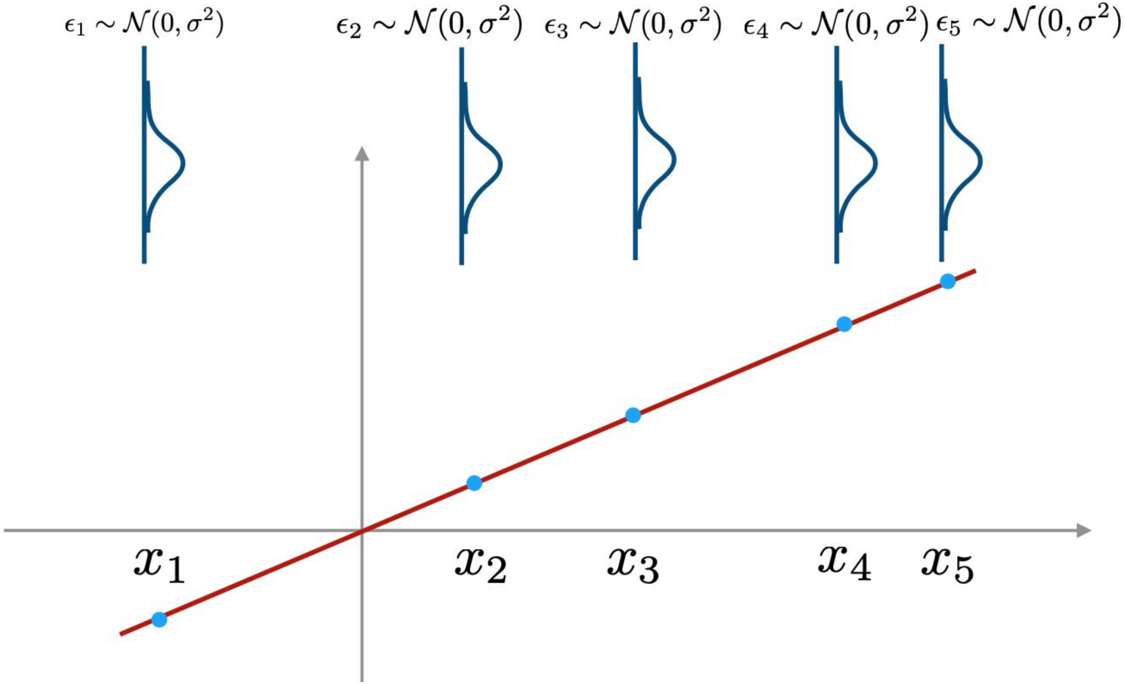


Example

Let consider a simple linear model with intercept 0 and 5 examples



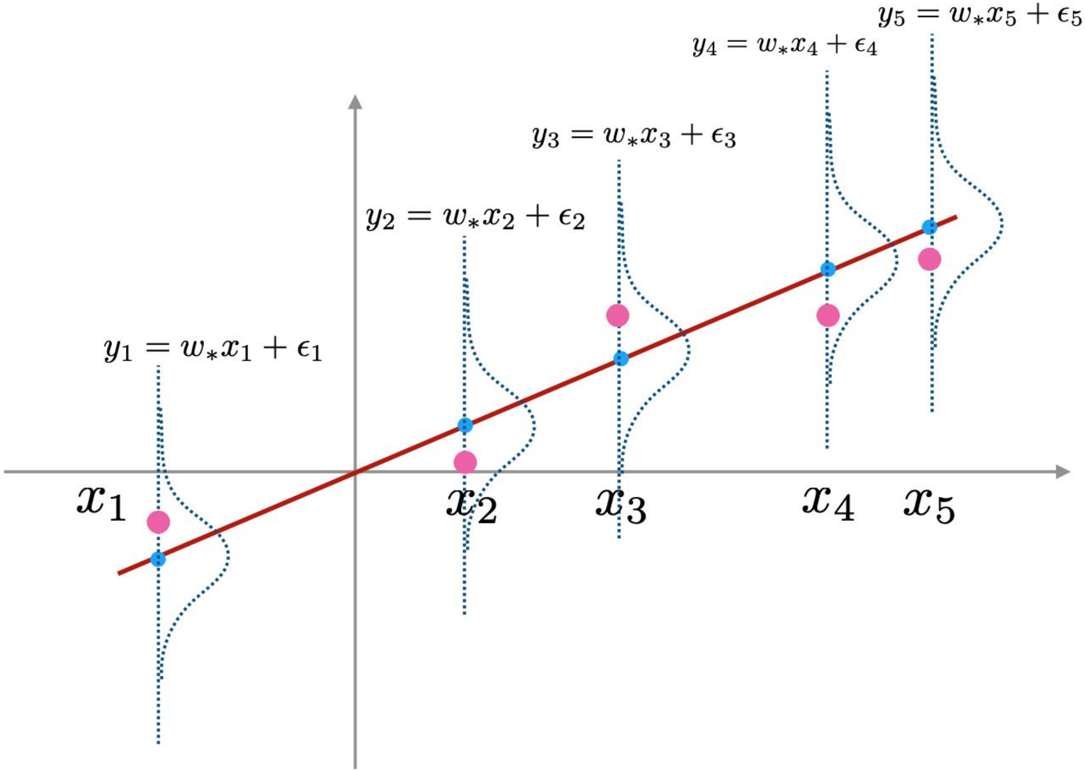
Example


$\epsilon_1 \sim \mathcal{N}(0, \sigma^2)$
$\epsilon_2 \sim \mathcal{N}(0, \sigma^2)$
$\epsilon_3 \sim \mathcal{N}(0, \sigma^2)$
$\epsilon_4 \sim \mathcal{N}(0, \sigma^2)$
$\epsilon_5 \sim \mathcal{N}(0, \sigma^2)$
x_1
x_2
x_3
x_4
x_5

The noises are independent of the samples

Example

This is what we observe, and noise + actual values:



$y_5 = w_* x_5 + \epsilon_5$
$y_4 = w_* x_4 + \epsilon_4$
$y_3 = w_* x_3 + \epsilon_3$
$y_2 = w_* x_2 + \epsilon_2$
$y_1 = w_* x_1 + \epsilon_1$
x_1
x_2
x_3
x_4
x_5

Recall

Maximum Likelihood Estimator (MLE)

Statistical estimation interpretation

What is the distribution of $y_i = \boldsymbol{w}_*^\top \boldsymbol{x}_i + \epsilon_i\,,\ \ \epsilon_i \sim \mathcal{N}(0, \sigma^2)$

The distribution is: $\mathbb{P}[y_i \mid \boldsymbol{x}_i; \boldsymbol{w}_*] \sim \mathcal{N}(\boldsymbol{w}_*^\top \boldsymbol{x}_i, \sigma^2)$

Let assume we are given sample from this distribution:

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$$

Can we estimate the unknown parameter vector: $\boldsymbol{w}_*$

Statistical estimation interpretation

Let’s compute the likelihood for a fixed parameter vector

$$\mathbb{P}[y_i \mid \boldsymbol{x}_i; \boldsymbol{w}_*] \sim \mathcal{N}(\boldsymbol{w}_*^\top \boldsymbol{x}_i, \sigma^2)$$

$$\mathcal{L}(\mathcal{D}|\boldsymbol{w}) = \prod_{i=1}^{n} \mathbb{P}[y_i|\boldsymbol{x}_i; \boldsymbol{w}]$$

Statistical estimation interpretation

Let’s compute the likelihood for a fixed parameter vector $\mathbb{P}[y_i \mid \boldsymbol{x}_i; \boldsymbol{w}_*] \sim \mathcal{N}(\boldsymbol{w}_*^\top \boldsymbol{x}_i, \sigma^2)$

$$\mathcal{L}(\mathcal{D}|\boldsymbol{w}) = \prod_{i=1}^{n} \mathbb{P}[y_i|\boldsymbol{x}_i; \boldsymbol{w}]$$

$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}\right)$$

Statistical estimation interpretation

Let’s compute the likelihood for a fixed parameter vector $\mathbb{P}[y_i \mid \boldsymbol{x}_i; \boldsymbol{w}_*] \sim \mathcal{N}(\boldsymbol{w}_*^\top \boldsymbol{x}_i, \sigma^2)$

$$\mathcal{L}(\mathcal{D}|\boldsymbol{w}) = \prod_{i=1}^{n} \mathbb{P}[y_i|\boldsymbol{x}_i; \boldsymbol{w}]$$

$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}\right)$$

Let’s take the log

$$\ln \mathcal{L}(\mathcal{D}|\boldsymbol{w}) = -\frac{n}{2}\ln(2\pi) - n\ln\sigma - \sum_{i=1}^{n} \frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}$$

Statistical estimation interpretation

The optimal solution is the one that maximizes the log-likelihood (known as MLE estimator):

$$\ln \mathcal{L}(\mathcal{D}|\boldsymbol{w}) = -\cancel{\frac{n}{2}\ln(2\pi)} - \cancel{n\ln\sigma} - \sum_{i=1}^{n} \frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}$$

$$\Downarrow$$

$$\arg\max_{\boldsymbol{w}} \quad -\sum_{i=1}^{n} \frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}$$

Statistical estimation interpretation

The optimal solution is the one that maximizes the log-likelihood (known as MLE estimator):

$$\ln \mathcal{L}(\mathcal{D}|\boldsymbol{w}) = \cancel{-\frac{n}{2}\ln(2\pi)} \cancel{- n\ln\sigma} - \sum_{i=1}^{n} \frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}$$

$$\Downarrow$$

$$\arg\max_{\boldsymbol{w}} \; -\sum_{i=1}^{n} \frac{\left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2}{2\sigma^2}$$

= (equivalent to)

$$\arg\min_{\boldsymbol{w}} \; \sum_{i=1}^{n} \left(y_i - \boldsymbol{w}^\top \boldsymbol{x}_i\right)^2$$

Wow! MLE estimator under Gaussian noise assumption is exactly same as OLS!

Terminology woes

Different fields use different terminology and symbols.

- Data points = **objects** = **examples**= rows = observations.
- Inputs = predictors = **features**= explanatory variables= regressors= independent variables = covariates = columns.
- Outputs = outcomes = targets = response variables = dependent variables (also called a “label” if it’s categorical).
- Regression coefficients = **weights** = parameters = betas.

With linear regression, the symbols are inconsistent too:

- In **ML**, the data is **X** and the weights are **w**, dimension is ***d***.
- In **statistics**, the data is **X** and the weights are **β**, dimension is ***p***.
- In **optimization**, the data is **A** and the weights are **x**.