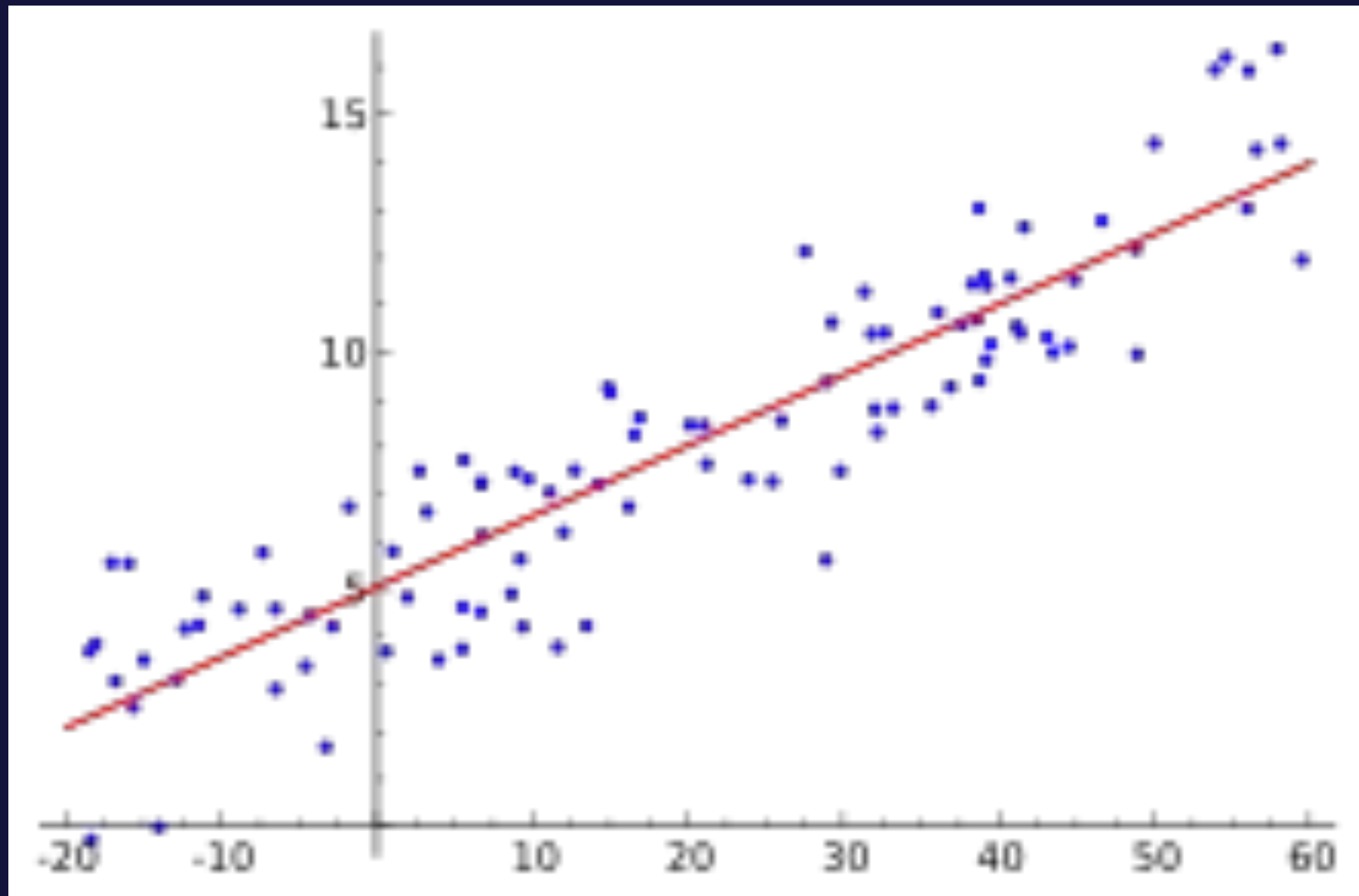
Introduction to linear regression

Lecturer: Yangchen Pan, Department of Engineering Science

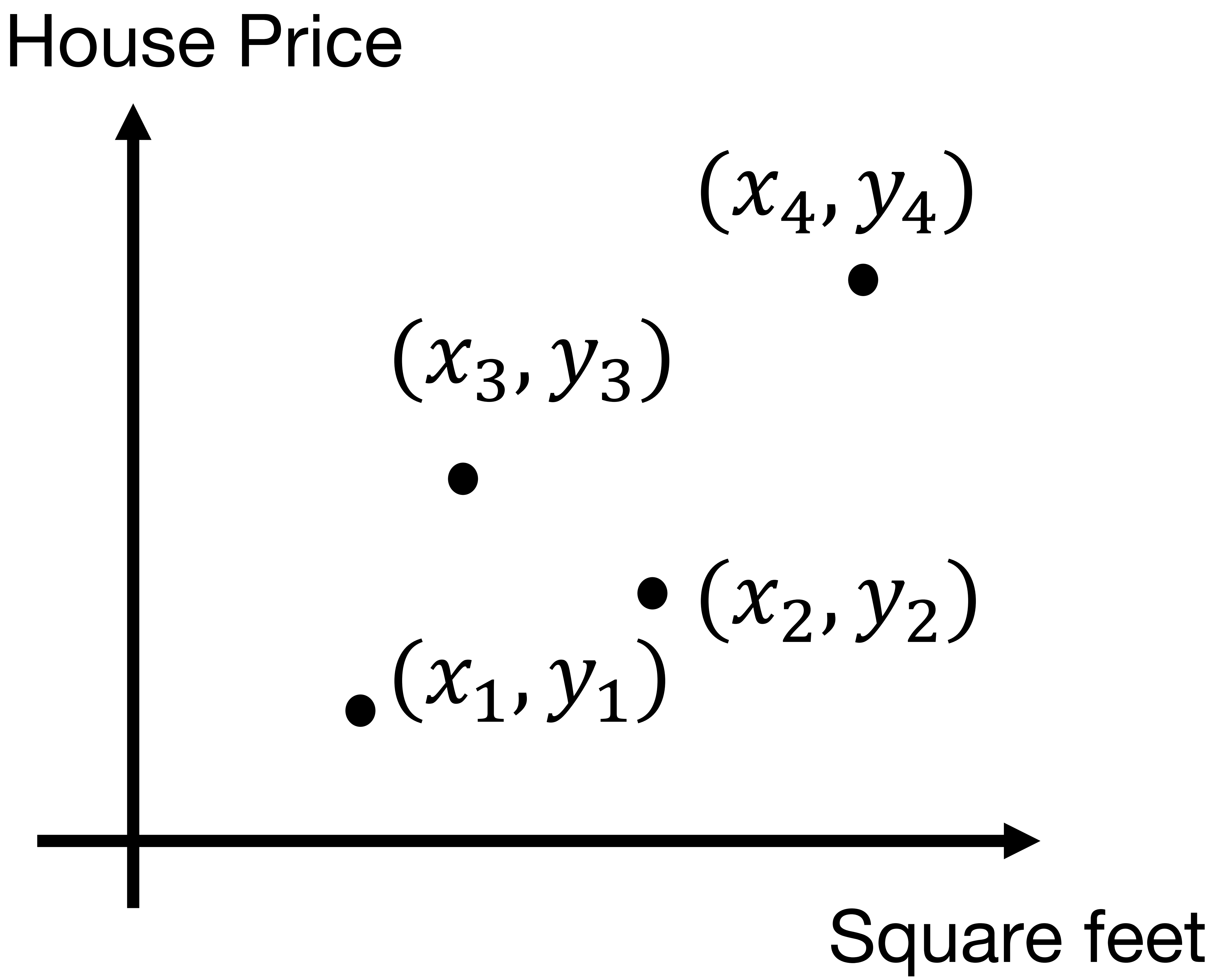


A Motivating Example

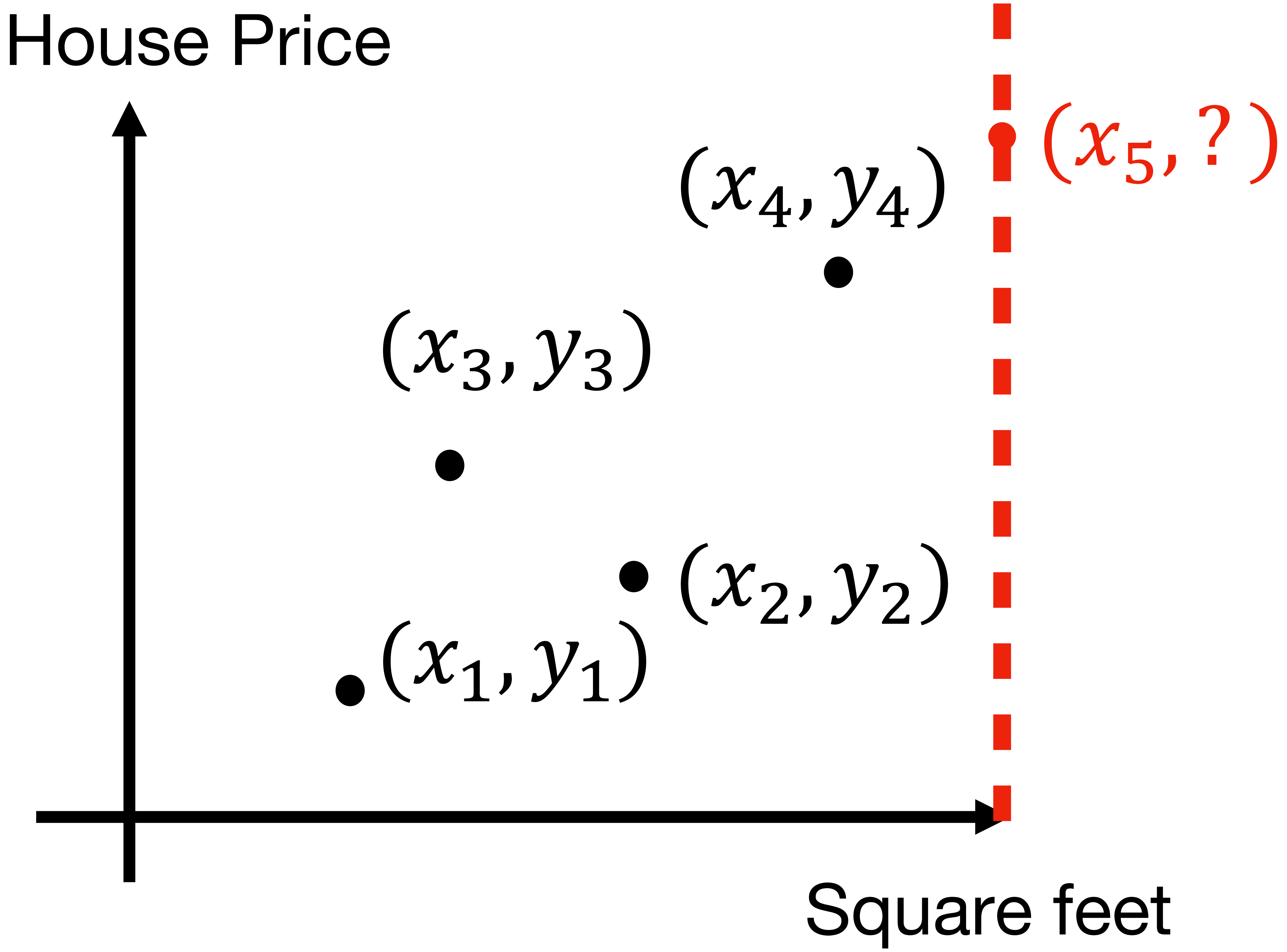
Dataset: we have some houses' prices and their areas in square feet

Goal: once we have a new house's square feet, can we make a reasonable guess/prediction for its price

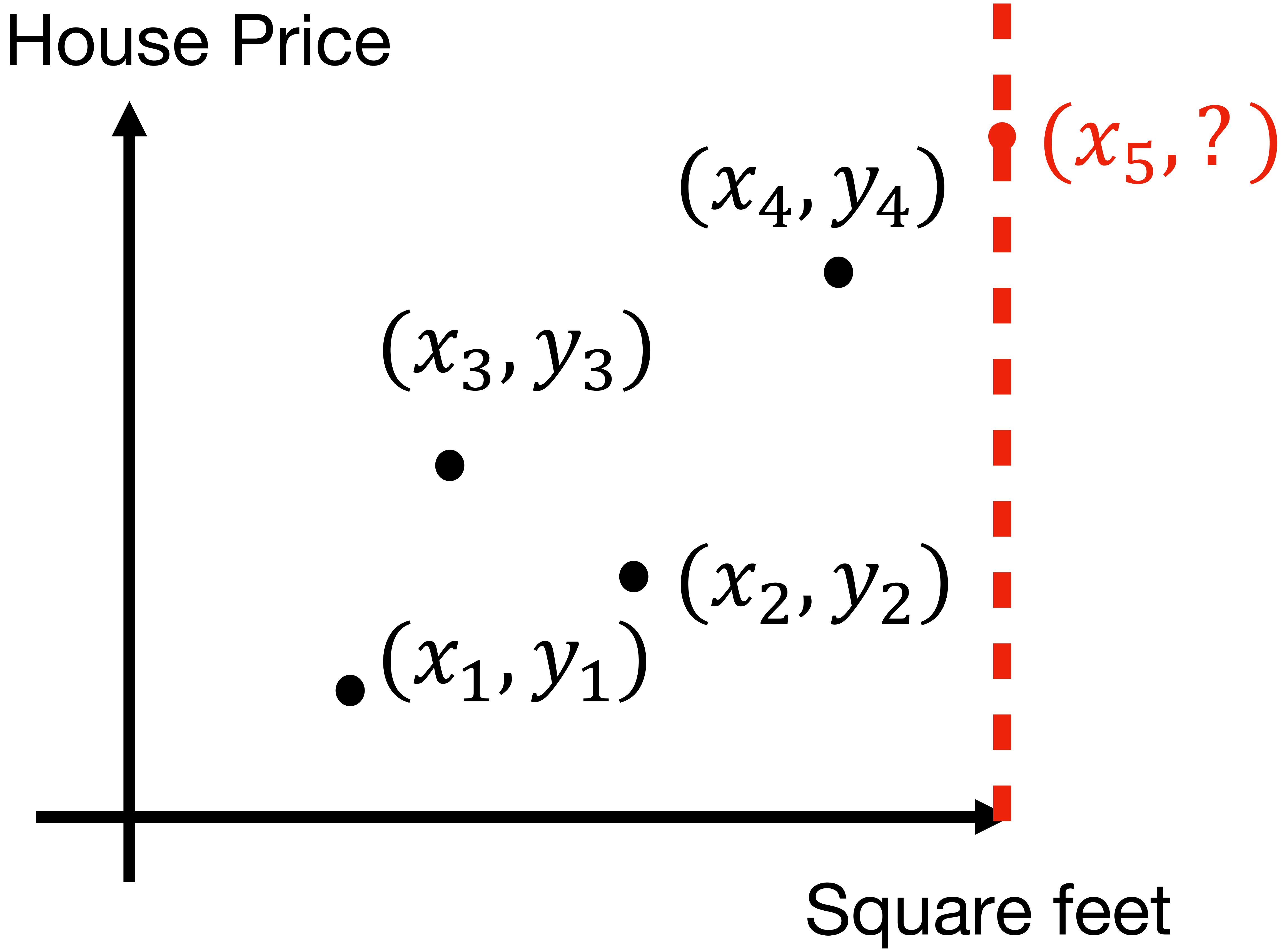
A Motivating Example



A Motivating Example



Ordinary Least Squares Regression



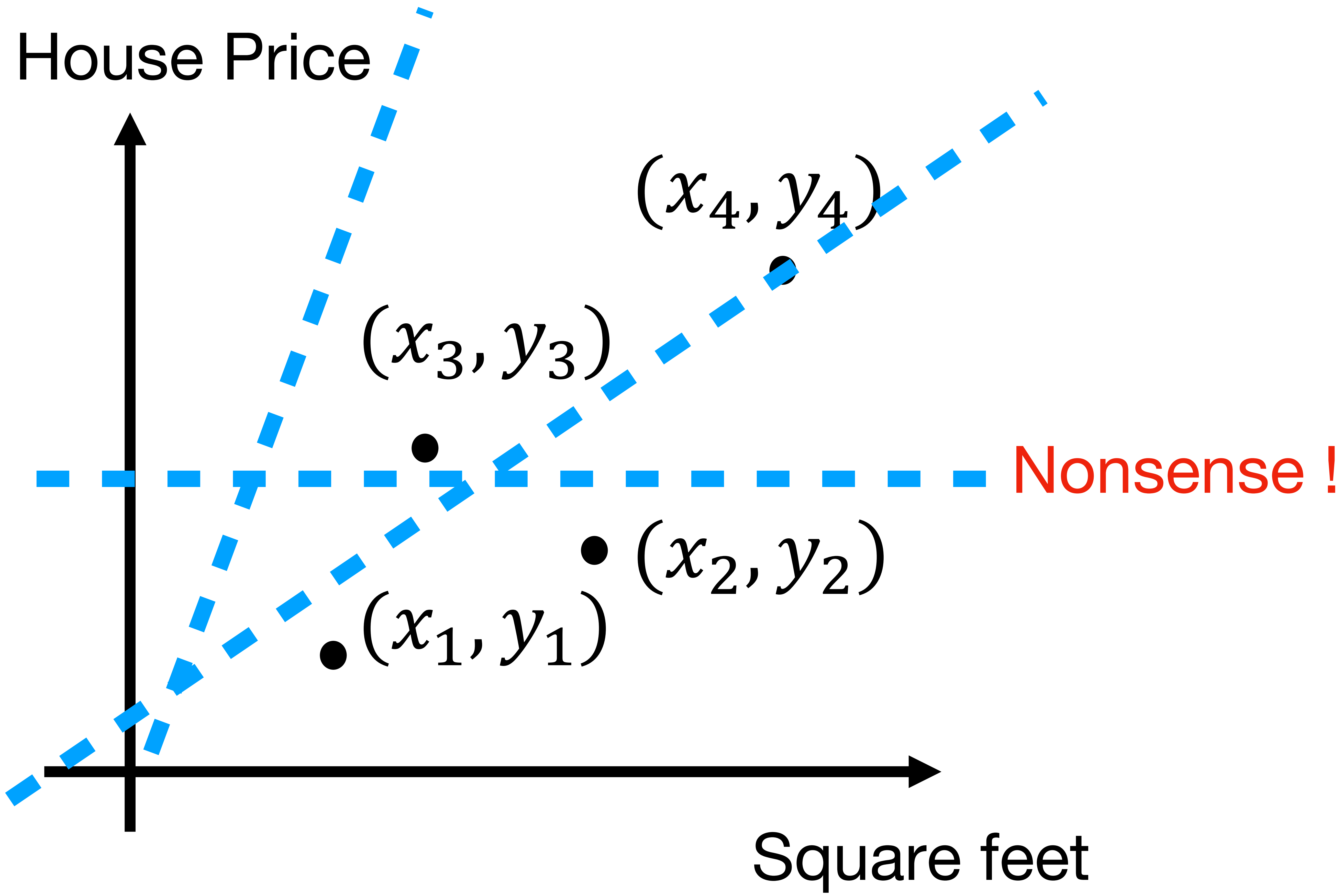
Regression: capturing the relationship between a dependent (outcome/response) variable and one or more independent variables

Independent variable: square feet

Dependent variable: house price

Linear Regression: express the outcome/dependent variable in terms of the independent variable as a linear function

Ordinary Least Squares Regression

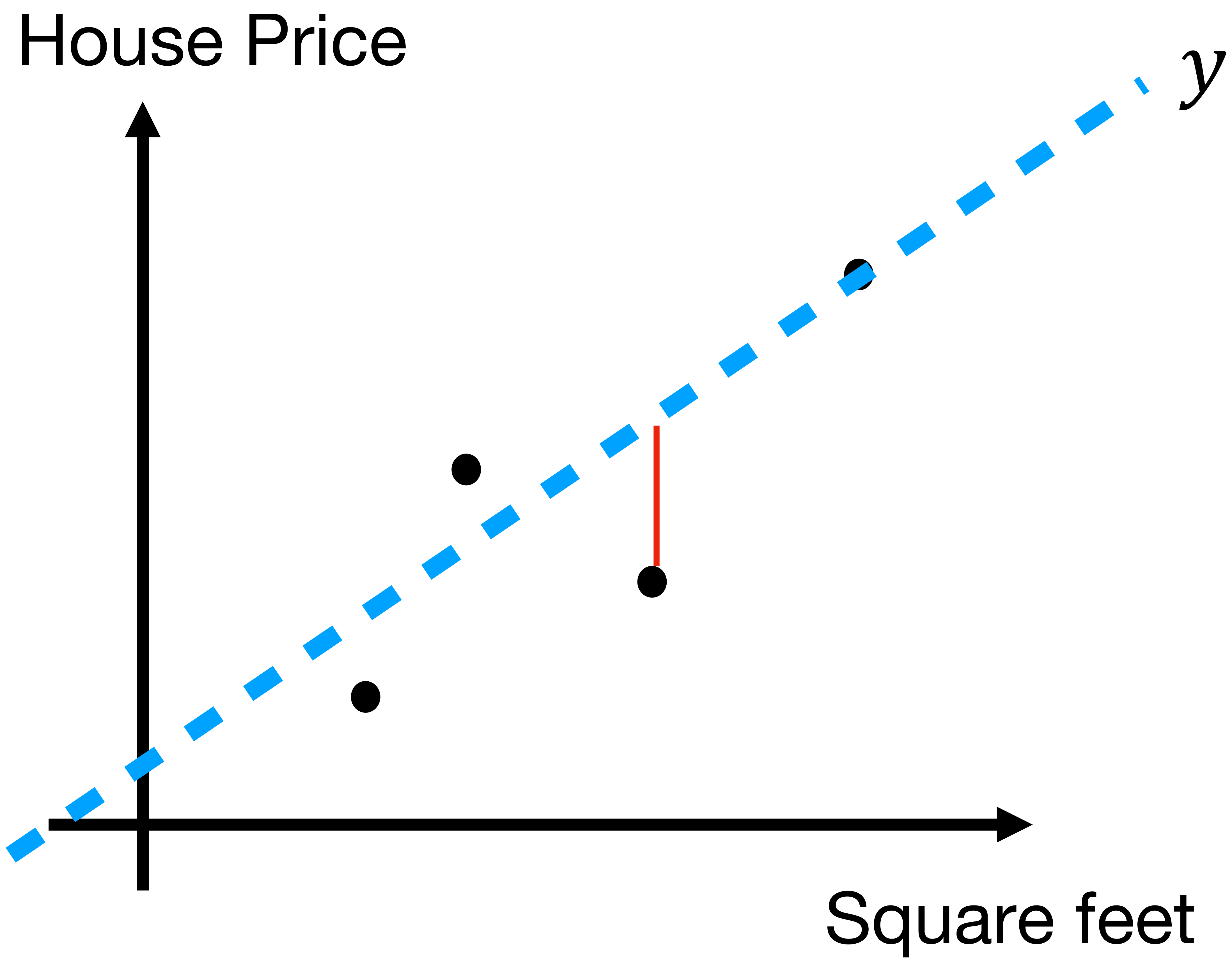


Least Squares: a specific & reasonable way to choose a specific line to describe the relationship between the dependent variable and independent variable

Independent variable: square feet

Dependent variable: house price

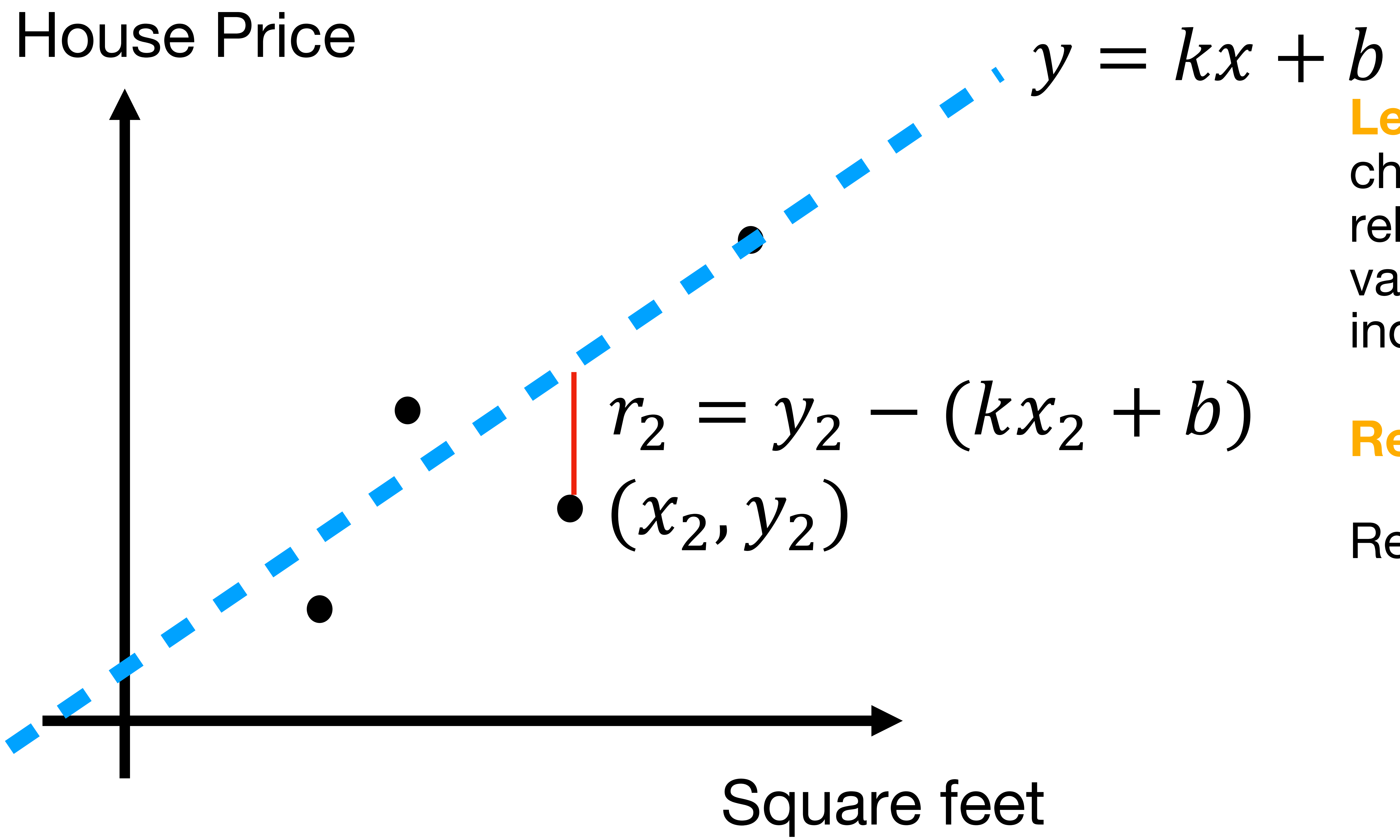
Ordinary Least Squares Regression



$$y = kx + b$$

Least Squares: a specific way to choose a specific line to describe the relationship between the dependent variable (house price) and the independent variable (square feet)

Ordinary Least Squares Regression

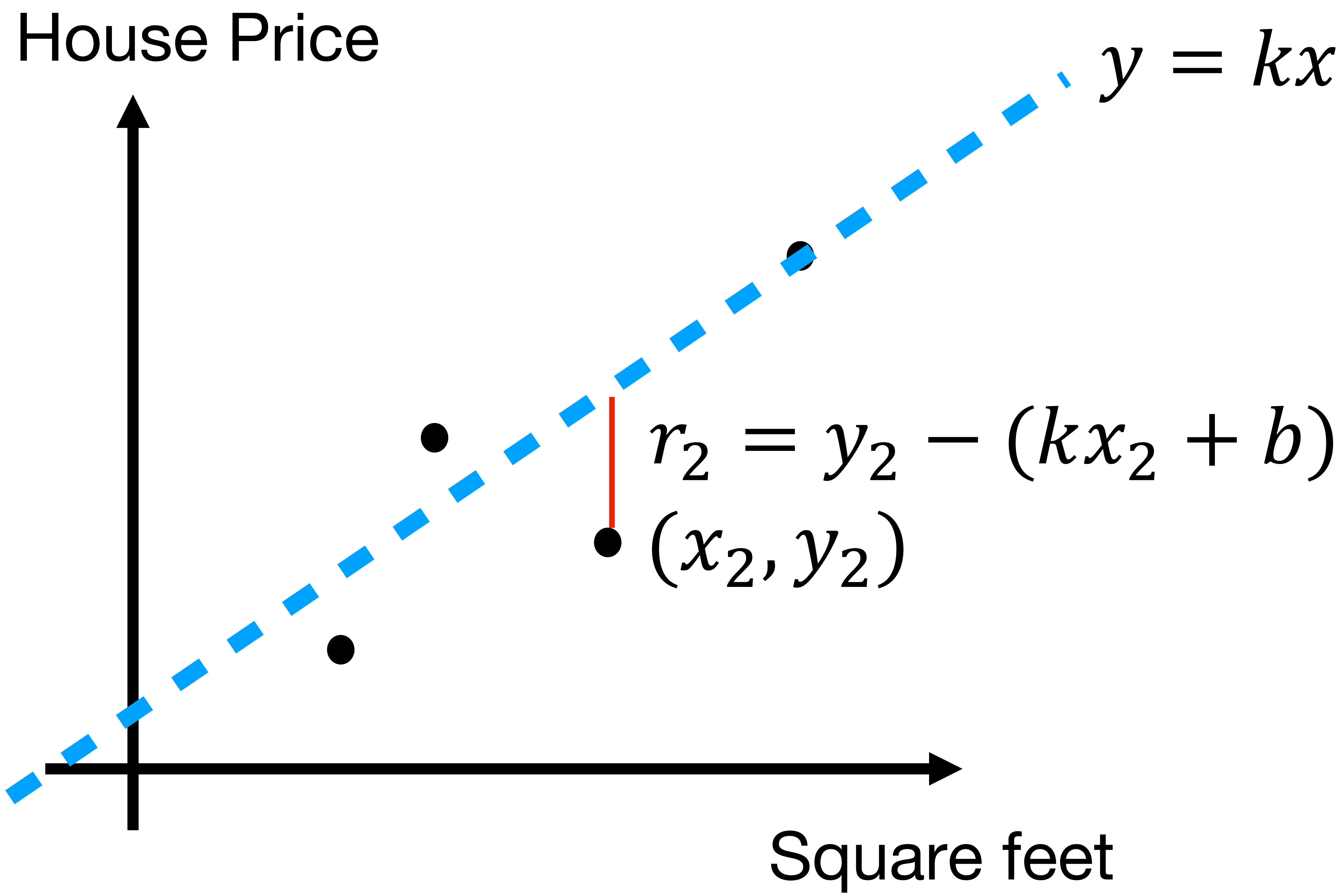


Least Squares: a specific way to choose a specific line to describe the relationship between the dependent variable (house price) and the independent variable (square feet)

Residual: $r_2 = y_2 - (kx_2 + b)$

Residual = observation - model's output

Ordinary Least Squares Regression



Least Squares: choosing the line (i.e. specify the value of the slope and intercept) to minimize the sum of squares of **residuals** of all the 4 points

Formally, we want: $\min_{k,b} \sum_{i=1}^{4} (y_i - (kx_i + b))^2$

Ordinary Least Squares Regression (OLS)

OLS: $\min_{k,b} \sum_{i=1}^{4} (y_i - (kx_i + b))^2$ **(we have 4 houses' area and price)**

In addition to square feet, if we have additional information about the house, such as the age, the number of bedrooms, location information (whether it is close to shopping/transportation centers), etc., we hopefully make better predictions.

In math, we use a vector to summarize all relevant information: $x_i \in \mathbb{R} \to \mathbf{x}_i \in \mathbb{R}^d, k \in \mathbb{R} \to \mathbf{k} \in \mathbb{R}^d$

Ordinary Least Squares Regression (OLS)

OLS: $\min_{k,b} \sum_{i=1}^{4} (y_i - (kx_i + b))^2$ **(we have 4 houses' area and price)**

In addition to square feet, if we have additional information about the house, such as the age, the number of bedrooms, location information (whether it is close to shopping/transportation centers), etc., we hopefully make better predictions.

In math, we use a vector to summarize all relevant information: $x_i \in \mathbb{R} \rightarrow \mathbf{x}_i \in \mathbb{R}^d, k \in \mathbb{R} \rightarrow \mathbf{k} \in \mathbb{R}^d$

Then the model output of a house i becomes:
$\mathbf{k}^\top \mathbf{x}_i + b$

Ordinary Least Squares Regression (OLS)

OLS: $\min_{k,b} \sum_{i=1}^{4} (y_i - (kx_i + b))^2$ **(we have 4 houses' area and price)**

In addition to square feet, if we have additional information about the house, such as the age, the number of bedrooms, location information (whether it is close to shopping/transportation centers), etc., we hopefully make better predictions.

In math, we use a vector to summarize all relevant information: $x_i \in \mathbb{R} \rightarrow \mathbf{x}_i \in \mathbb{R}^d, k \in \mathbb{R} \rightarrow \mathbf{k} \in \mathbb{R}^d$

Then the model output of a house i becomes:
$\mathbf{k}^\top \mathbf{x}_i + b$

If you have n houses and their prices, then:

$$\min_{\mathbf{k},b} \sum_{i=1}^{n} (y_i - (\mathbf{k}^\top \mathbf{x}_i + b))^2$$

For notation convenience, define a few new notations:

$\mathbf{X} \in \mathbb{R}^{n\times d}, \mathbf{y} \in \mathbb{R}^n$

$\mathbf{w} = (k^\top, b)^\top, \mathbf{x}_i \leftarrow (\mathbf{x}_i, 1), y_i = \mathbf{x}_i^\top \mathbf{w}$

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 .$$

Ordinary Least Squares Regression (OLS) - a brief summary

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,\ldots,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,.$$

This is the cost function for linear regression.

$$\mathbf{w}_{\mathrm{MLE}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}.$$

This is called least-squares solution.

Ordinary Least Squares Regression (OLS) - a brief summary

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,\ldots,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,.$$

This is the cost function for linear regression.

$$\mathbf{w}_{\mathrm{MLE}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

This is called least-squares solution.

What if we don't have additional info, how to make the model more powerful?

Learning with nonlinear features

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,...,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,.$$

What if we don't have additional info, how to make the model more powerful?

- Given a sample x, we can construct some functions to expand the feature: $\phi_i(x), i = 1, 2, \ldots$

Example 1. polynomial basis

$$\Phi = \begin{bmatrix} 1 & x_1 & \cdots & x_1^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_t & \cdots & x_t^d \end{bmatrix} \in \mathbb{R}^{t\times(d+1)}$$

Learning with nonlinear features

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,\ldots,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 .$$

But, how do we choose such functions/basis to expand feature? – similar examples share similar features

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right)$$

1. If we use this similarity function as feature function and choose some fixed instances $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_l \in \mathcal{X}$, the new representation would be

$$\boldsymbol{x} \mapsto \varphi(\boldsymbol{x}) = [\kappa(\boldsymbol{x}, \boldsymbol{\mu}_1), \cdots, \kappa(\boldsymbol{x}, \boldsymbol{\mu}_l)]^\top .$$

2. If we place the centres $\boldsymbol{\mu}_j$ on the inputs $\boldsymbol{x}_i$ themselves, then the new data matrix Φ would be: $\Phi_{i,j} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Note that this Φ is symmetric since $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \kappa(\boldsymbol{x}_j, \boldsymbol{x}_i)$. Moreover, if Φ is invertible, the solution to the learning problem (1.1) with L_2 error will be $\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}} \|\Phi\boldsymbol{w} - \boldsymbol{y}\|_2^2 = \Phi^{-1}\boldsymbol{y}$.

Notes from Junfeng

Learning with nonlinear features

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,\ldots,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,.$$

Other methods to get data representation: other similarity functions, neural networks, etc.

Issues – overfitting

Overfitting:

1) models become overly fitted into the (often noisy) training data;

2) but performs poorly on new, unseen data.

Issues – overfitting

Overfitting:

1) models become overly fitted into the (often noisy) **training data**;

2) but performs poorly on **new, unseen data**.

A brief distraction: training, validation, and testing datasets

Training: you use to train your model/update your parameters, the data your model will directly fit

Validation: used to do model selection/hyper-parameter tuning, but not for training your model

Testing: used to evaluate your model's performance **on unseen data**. It should not be used for training or model selection.

In your exercises, you can simply treat validation & testing the same.

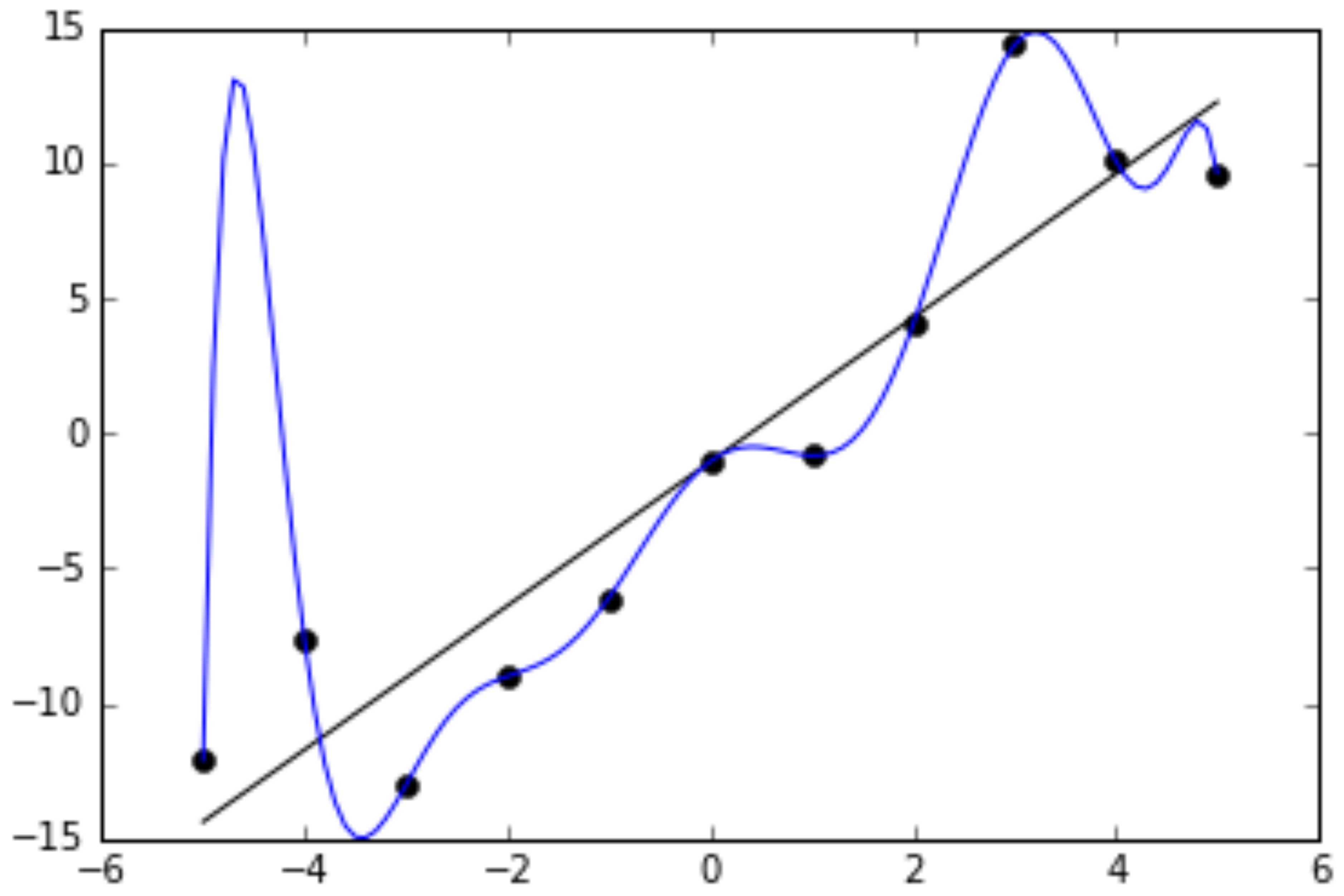
Issues – overfitting

Overfitting:

1) models become overly fitted into the (often noisy) training data;

2) but performs poorly on new, unseen data.

Question: if we add more and more polynomial degrees to the feature matrix, what would happen to the performance on the training and validation set respectively?



Issues – overfitting

Regularization:

Add a loss to penalize weight vector. Intuitively, they push the magnitude of the weights to be small or even zero, and so one might expect some features are no longer that useful, hence regularization can reduce the model complexity.

Here are two types of popular regularization, minimizing the following two objective functions:

L1/lasso: $$(\mathbf{X}\mathbf{w}-\mathbf{y})^\top(\mathbf{X}\mathbf{w}-\mathbf{y})+\lambda\|\mathbf{w}\|_1$$

L2/ridge: $$(\mathbf{X}\mathbf{w}-\mathbf{y})^\top(\mathbf{X}\mathbf{w}-\mathbf{y})+\lambda\mathbf{w}^\top\mathbf{w}$$

Issues – overfitting

Overfitting:

1) models become overly fitted into the (often noisy) training data;

2) but performs poorly on new, unseen data.

Regularization:

Add a loss to penalize weight vector. Intuitively, they push the magnitude of the weights to be small or even zero, and so one might expect some features are no longer that useful, hence regularization can reduce the model complexity.

Consider an example of why regularization may work:

You have a feature vector: $\mathbf{x} = (1, x, x^2, x^3, x^4, \dots x^{19}, x^{20})$

Your linear model's output is: $y = \mathbf{x}^\top \mathbf{w} = \sum_{i=0}^{20} x^i \mathbf{w}(i)$

Regularization (i.e., penalty on large weights) may push those w(i)s multiplying high degree feature to be small, so these features are no longer useful.

Issues – overfitting

Regularization:

Add a loss to penalize weight vector. Intuitively, they push the magnitude of the weights to be small or even zero, and so one might expect some features are no longer that useful, hence regularization can reduce the model complexity.

Here are two types of popular regularization, minimizing the following two objective functions:

L1/lasso: $(\mathbf{X}\mathbf{w}-\mathbf{y})^\top(\mathbf{X}\mathbf{w}-\mathbf{y})+\lambda\|\mathbf{w}\|_1$ For L1, no closed-form solution, sparse sol

L2/ridge: $(\mathbf{X}\mathbf{w}-\mathbf{y})^\top(\mathbf{X}\mathbf{w}-\mathbf{y})+\lambda\mathbf{w}^\top\mathbf{w}$ For L2, closed-form solution is:

$$\mathbf{w}_{\mathrm{MAP}}=(\mathbf{X}^\top\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}.$$

Without regularization, lambda=0, so the solution is: $\mathbf{w}_{\mathrm{MLE}}=(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}.$

How regularization makes model more robust

You try to minimize $(\mathbf{X}\mathbf{w}-\mathbf{y})^\top(\mathbf{X}\mathbf{w}-\mathbf{y})+\boxed{\lambda\mathbf{w}^\top\mathbf{w}}$

(ridge regression): $\mathbf{w}_{\mathrm{MAP}}=(\mathbf{X}^\top\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}.$

Choosing the lambda here leads to the so-called bias-variance tradeoff problem.

With regularization: $\mathbf{w}_{\mathrm{MAP}}=(\mathbf{X}^\top\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}.=\sum_{j=1}^{d}\frac{\sigma_j\mathbf{u}_j^\top\mathbf{y}}{\sigma_j^2+\lambda}\mathbf{v}_j$

Without regularization: $\mathbf{w}=(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}=\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^\top\mathbf{y}=\sum_{j=1}^{d}\frac{\mathbf{u}_j^\top\mathbf{y}}{\sigma_j}\mathbf{v}_j$

Geometric View for OLS

Now, let's take a look at the OLS's geometric interpretation. To start, we rewrite the objective into a matrix form:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,.$$

$$\begin{aligned}\mathrm{Err}(\mathbf{w}) &= (\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) \\ &= \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,,\end{aligned}$$

$$\nabla \mathrm{Err}(\mathbf{w}) = 2\mathbf{X}^\top \mathbf{X}\mathbf{w} - 2\mathbf{X}^\top \mathbf{y}$$

Geometric View for OLS

Now, let's take a look at the OLS's geometric interpretation. To start, we rewrite the objective into a matrix form:

$$\mathbf{w}_{\text{MLE}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 .$$

$$\begin{aligned}\text{Err}(\mathbf{w}) &= (\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) \\ &= \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 ,\end{aligned}$$

Set this gradient equal to zero, we find:

$$\mathbf{w}_{\text{MLE}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}.$$

We can now express the predicted target values as:

$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{X}\mathbf{w}_{\text{MLE}} \\ &= \boxed{\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top}\mathbf{y}.\end{aligned}$$

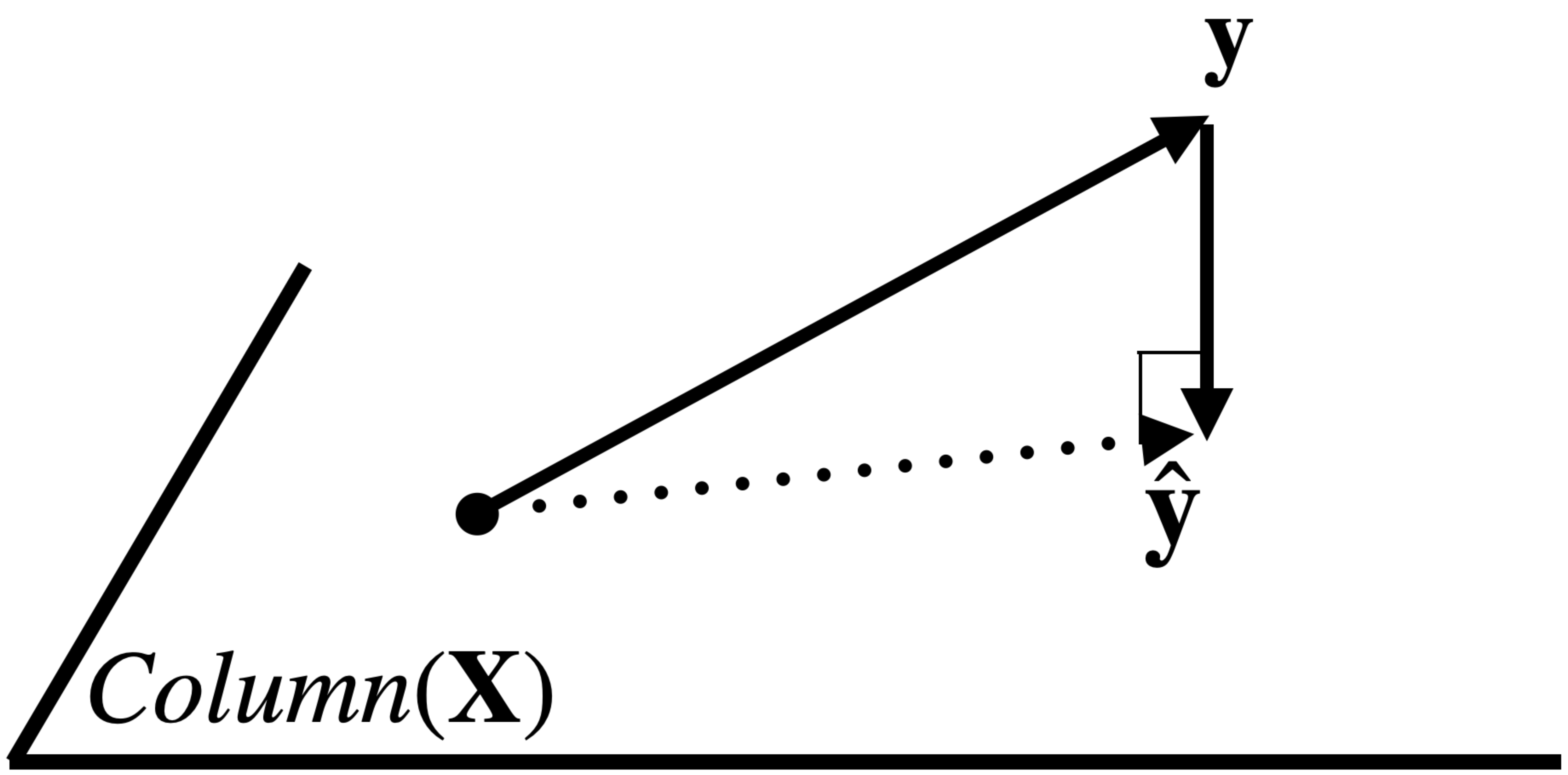
This matrix is called a projection.

Geometric View for OLS

We can now express the predicted target values as:

$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{X}\mathbf{w}_{\mathrm{MLE}} \\ &= \boxed{\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top}\mathbf{y}.\end{aligned}$$

This matrix is called a projection.



This is saying: when the columns of X (i.e. the independent variables/features of a house) are not powerful enough to linearly express the y vector; OLS solution would try to express the one closest to y, which is the orthogonal projection of y onto the column space of X.

Probability Perspective

Ordinary Least Squares Regression (OLS) - a brief summary

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,\ldots,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\min_{w} \sum_{i=1}^{n} (y_i - w^\top \mathbf{x}_i)^2$$

These seem to be intuitive & reasonable choices. Do we have a mathematical derivation/justification for these choices?

Probabilistic view

Probabilistic View for Linear Regression

If you have n data points in the form of $\{(\mathbf{x}_i, y_i)\}_{i=1,2,\ldots,n}$, and you want to find a linear relationship between x and y, then OLS tells us to minimize the sum of squared residuals:

$$\min_{w} \sum_{i=1}^{n} (y_i - w^\top \mathbf{x}_i)^2$$

Assumption: the observed p$(y_i|x_i)$ *is Gaussian with mean* $w^T\mathbf{x_i}$ *and constant variance* σ^2.

$$p(y_i|\mathbf{x}_i, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(y_i - \mathbf{x}_i^\top \mathbf{w})^2}{2\sigma^2})$$

Then we are ready to formulate our maximum likelihood estimation (MLE) problem:

Probabilistic View for Linear Regression

Then we are ready to formulate our maximum likelihood estimation (MLE) problem:

$$\mathbf{w}_{\mathrm{MLE}} = \arg\max_{\mathbf{w}} \{p(\mathbf{y}|\mathbf{X}, \mathbf{w})\}$$

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^{n} p(y_i|\mathbf{x}_i, \mathbf{w}) \qquad (1)$$

$$\forall i \in \{1, \ldots, n\}, p(y_i|\mathbf{x}_i, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(y_i - \mathbf{x}_i^\top \mathbf{w})^2}{2\sigma^2})$$

Recall the assumption: $y_i \sim \mathcal{N}(y; \mathbf{x_i}^\top \mathbf{w}, \sigma^2)$.

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

$$\ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = -\sum_{i=1}^{n} \log(\sqrt{2\pi\sigma^2}) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \mathbf{w})^2$$

Probabilistic View for Linear Regression

Since maximizing the log-likelihood is equivalent to minimizing the negative log-likelihood, then

$$\begin{aligned}
\mathbf{w}_{MLE} &= \arg\max_{\mathbf{w}} \ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}) \\
&= \arg\min_{\mathbf{w}} - \ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}) \\
&= \arg\min_{\mathbf{w}} \sum_{i=1}^{n} \log(\sqrt{2\pi\sigma^2}) + \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \mathbf{w})^2 \\
&= \arg\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \mathbf{w})^2
\end{aligned}$$

Then we derived our Ordinary Least Squares Regression objective from the probabilistic perspective!

In-class exercise: MLE for Linear regression

Three steps:

1. Make an assumption on p(y|x) with some unknown parameter (recall: gaussian assumption)

2. Define the mean of the gaussian as linear function of x

3. Write down the log-likelihood function and simplify it

In-class exercise: MLE for Logistic regression

What if the target variables are binary, i.e., 0, 1? – a classification problem

Naturally, we can use Bernoulli distribution to model such target variable. That is,

$$p(y) = \mu^y (1 - \mu)^{1-y}$$

What should be p(y|x, w) then?

$$p(y|x, w) = (x^\top w)^y (1 - x^\top w)^{1-y}$$

Can we do this?

In-class exercise: MLE for Logistic regression

What if the target variables are binary, i.e., 0, 1? – a classification problem

Naturally, we can use Bernoulli distribution to model such target variable. That is,

$$p(y) = \mu^y (1 - \mu)^{1-y}$$

What should be p(y|x, w) then?

$$p(y|x, w) = (sigmoid(x^\top w))^y (1 - sigmoid(x^\top w))^{1-y}$$

Then, you follow the regular procedure as we do in linear regression, to write down the log-likelihood function.

In-class exercise: MLE for Logistic regression

What if the target variables are binary, i.e., 0, 1? – a classification problem

Naturally, we can use Bernoulli distribution to model such target variable. That is,

$$p(y) = \mu^y (1 - \mu)^{1-y}$$

What should be p(y|x, w) then?

$$p(y|x, w) = (sigmoid(x^\top w))^y (1 - sigmoid(x^\top w))^{1-y}$$

How to do prediction?

Generalized Linear Models

So far, we learned linear regression, logistic regression, let's summarize the general steps:

1. We specify a probability distribution to model target variable;
2. Parameterize the mean of the distribution (link function: link the linear output and the mean of the probability distribution you assume)
3. Do MLE to find the objective function
4. Use E[Y|X=x*] to do prediction

Regularization - probabilistic view

$$\mathbf{w}_{\text{MLE}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

As you may already note, the inverse may not exist or is "ill-conditioned"; then we get an unstable solution, unstable in the sense that a slight difference in the data would cause a significant difference in the solution.

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

$$\mathbf{X}^\top\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}^\top\mathbf{U}^\top\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top = \mathbf{V}\boldsymbol{\Sigma}_d^2\mathbf{V}^\top$$

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} = \mathbf{V}\boldsymbol{\Sigma}^{-1}\mathbf{U}^\top\mathbf{y} = \sum_{j=1}^{d} \frac{\mathbf{u}_j^\top \mathbf{y}}{\sigma_j} \mathbf{v}_j$$

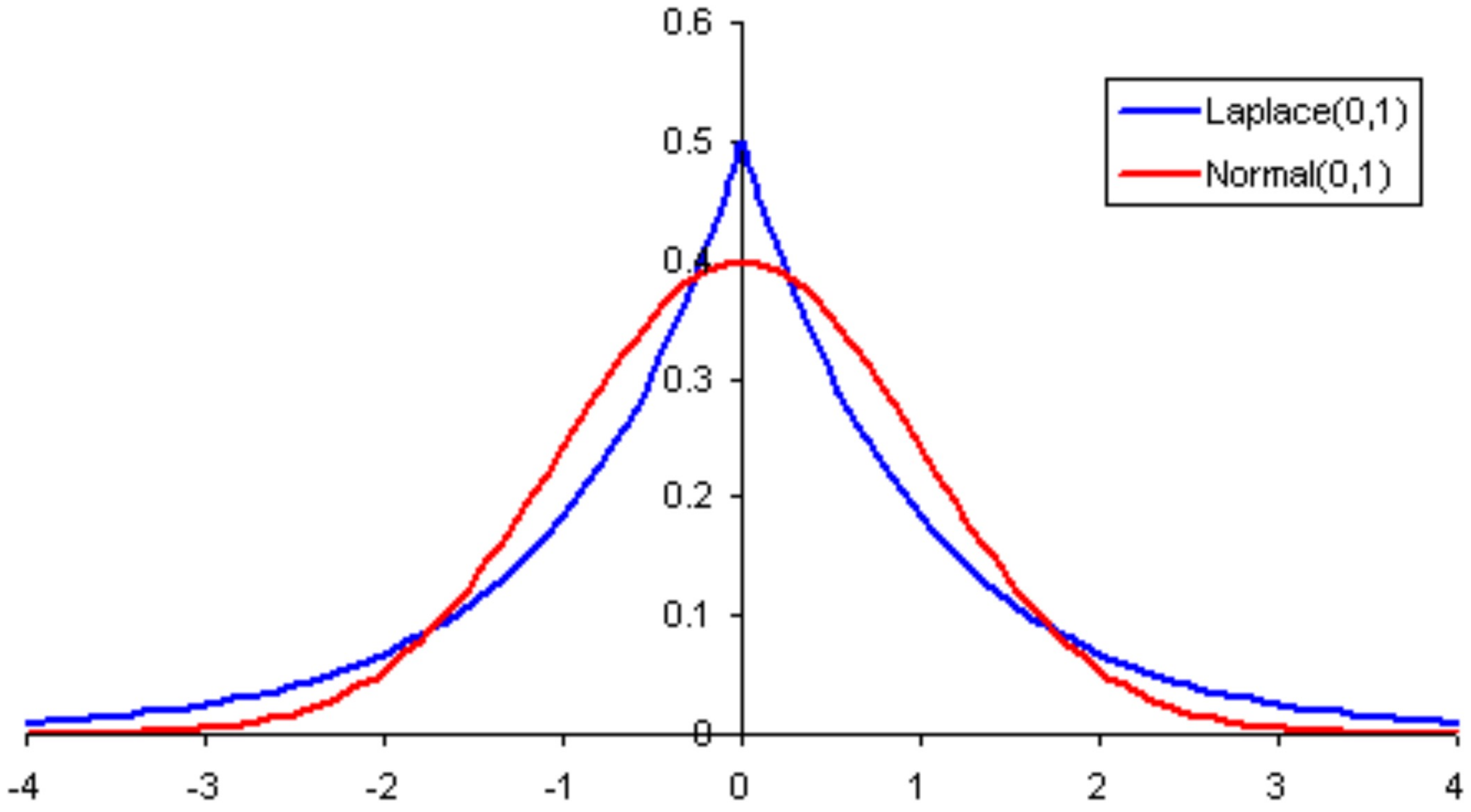
=1e-8?

where $\mathbf{u}_j, \mathbf{v}_j$ are the jth vectors in $\mathbf{U}, \mathbf{V}$, σ_j is the jth singular value in the diagonal matrix Σ

Regularization - probabilistic view

A hint: think about what happened to the solution (i.e., the weight vector) when some singular values are very small, and then think about what we can do to the probabilistic assumptions behind OLS.

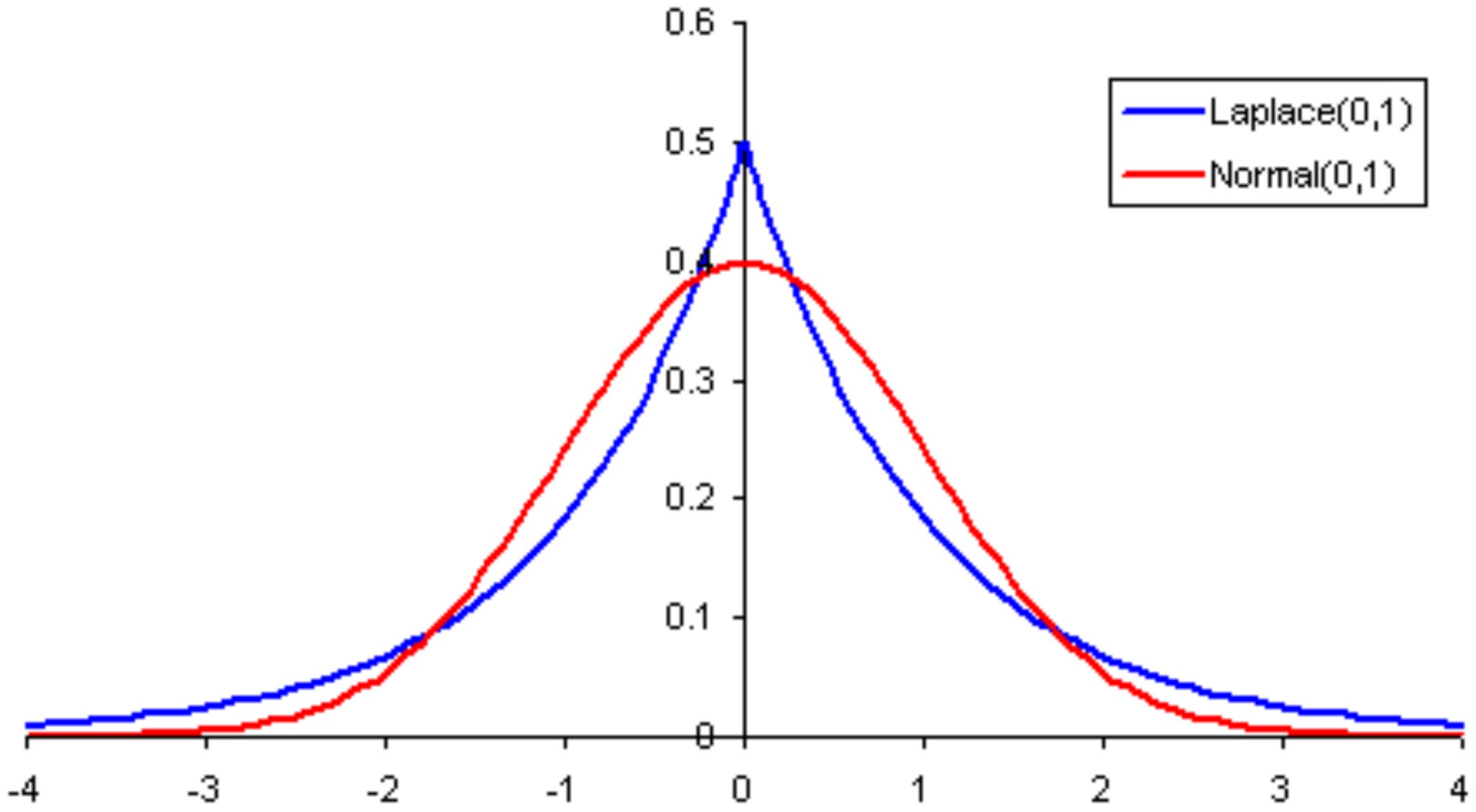
Enforce probabilistic priors on the weight vector to keep all its entries small!



Regularization - probabilistic view

A hint: think about what happened to the solution (i.e., the weight vector) when some singular values are very small, and then think about what we can do to the probabilistic assumptions behind OLS.

Enforce probabilistic priors on the weight vector to keep all its entries small!



The two types of priors result in the well-known L1 (lasso) and L2 (ridge) regularization (regression).

Regularization - probabilistic view

Instead of using MLE, now we need to use M**aximize A Posteriori** (MAP) estimation to derive our objective function since we assumed some prior on the weight vector we want to estimate.

A brief review of MAP:

Posterior distribution of **w** when given dataset $\mathbf{D}$:

$$p(\mathbf{w}|\mathbf{D}) = p(\mathbf{D}|\mathbf{w})p(\mathbf{w})/p(\mathbf{D}) \propto p(\mathbf{D}|\mathbf{w})p(\mathbf{w})$$

Maximize this posterior is equivalent to maximize its logarithm; hence, we note that MAP results in maximizing:

MLE's log-likelihood function + log of p(w)

Regularization - probabilistic view

Instead of using MLE, now we need to use M**aximize A Posteriori** (MAP) estimation to derive our objective function since we assumed some prior on the weight vector we want to estimate.

A brief review of MAP:

Posterior distribution of **w** when given dataset $\mathbf{D}$:

$$p(\mathbf{w}|\mathbf{D}) = p(\mathbf{D}|\mathbf{w})p(\mathbf{w})/p(\mathbf{D}) \propto p(\mathbf{D}|\mathbf{w})p(\mathbf{w})$$

Maximize this posterior is equivalent to maximize its logarithm; hence, we note that MAP results in maximizing:

MLE's log-likelihood function + log of p(w)

Which is equivalent to minimizing:

- MLE's log-likelihood function - log of p(w)

Regularization - probabilistic view

Instead of using MLE, now we need to use M**aximize A Posteriori** (MAP) estimation to derive our objective function since we assumed some prior on the weight vector we want to estimate.

A brief review of MAP:

Posterior distribution of **w** when given dataset $\mathbf{D}$:

$$p(\mathbf{w}|\mathbf{D}) = p(\mathbf{D}|\mathbf{w})p(\mathbf{w})/p(\mathbf{D}) \propto p(\mathbf{D}|\mathbf{w})p(\mathbf{w})$$

Maximize this posterior is equivalent to maximize its logarithm; hence, we note that MAP results in maximizing:

MLE's log-likelihood function + log of p(w)

Which is equivalent to minimizing:

- MLE's log-likelihood function - log of p(w)

This is the original OLS objective

Regularization - probabilistic view

What is logarithm of p(w)?

Consider Gaussian prior first (the MAP of Laplace case can be derived in the similar way):

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; 0, \lambda^{-1} I)$$

$$-\ln p(\mathbf{w}) = \ln(2\pi|\lambda^{-1}\mathbf{I}|) + \frac{\mathbf{w}^\top \mathbf{w}}{2\lambda^{-1}} = \underline{\ln(2\pi) - d\ln(\lambda)} + \frac{\lambda}{2}\mathbf{w}^\top \mathbf{w}.$$

Will be dropped off when taking gradient
w.r.t. w. So it does not affect optimization.

Regularization - probabilistic view

What is logarithm of p(w)?

Consider Gaussian prior first (the MAP of Laplace case can be derived in the similar way):

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; 0, \lambda^{-1} I)$$

$$-\ln p(\mathbf{w}) = \ln(2\pi|\lambda^{-1}\mathbf{I}|) + \frac{\mathbf{w}^\top \mathbf{w}}{2\lambda^{-1}} = \ln(2\pi) - d\ln(\lambda) + \frac{\lambda}{2}\mathbf{w}^\top \mathbf{w}.$$

Then computing - MLE's log-likelihood function - log of p(w), we finally arrive the objective function:

$$(\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^\top \mathbf{w}$$

Regularization - probabilistic view

What is logarithm of p(w)?

Consider Gaussian prior first (the MAP of Laplace case can be derived in the similar way):

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; 0, \lambda^{-1} I)$$

$$-\ln p(\mathbf{w}) = \ln(2\pi|\lambda^{-1}\mathbf{I}|) + \frac{\mathbf{w}^\top \mathbf{w}}{2\lambda^{-1}} = \ln(2\pi) - d\ln(\lambda) + \frac{\lambda}{2}\mathbf{w}^\top \mathbf{w}.$$

Then computing - MLE's log-likelihood function - log of p(w), we finally arrive the objective function:

$$(\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^\top \mathbf{w}$$

Closed-form solution is: $\mathbf{w}_{\mathrm{MAP}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}.$

Regularization - probabilistic view

If we choose a Laplace prior, we can derive the loss function in the same way:

$$(\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \|\mathbf{w}\|_1$$

Then we no longer have a closed-form solution; gradient descent has to be used. This objective function is not differentiable at zero.

Finish exercise 1-3.