Machine Learning

CMPT 726

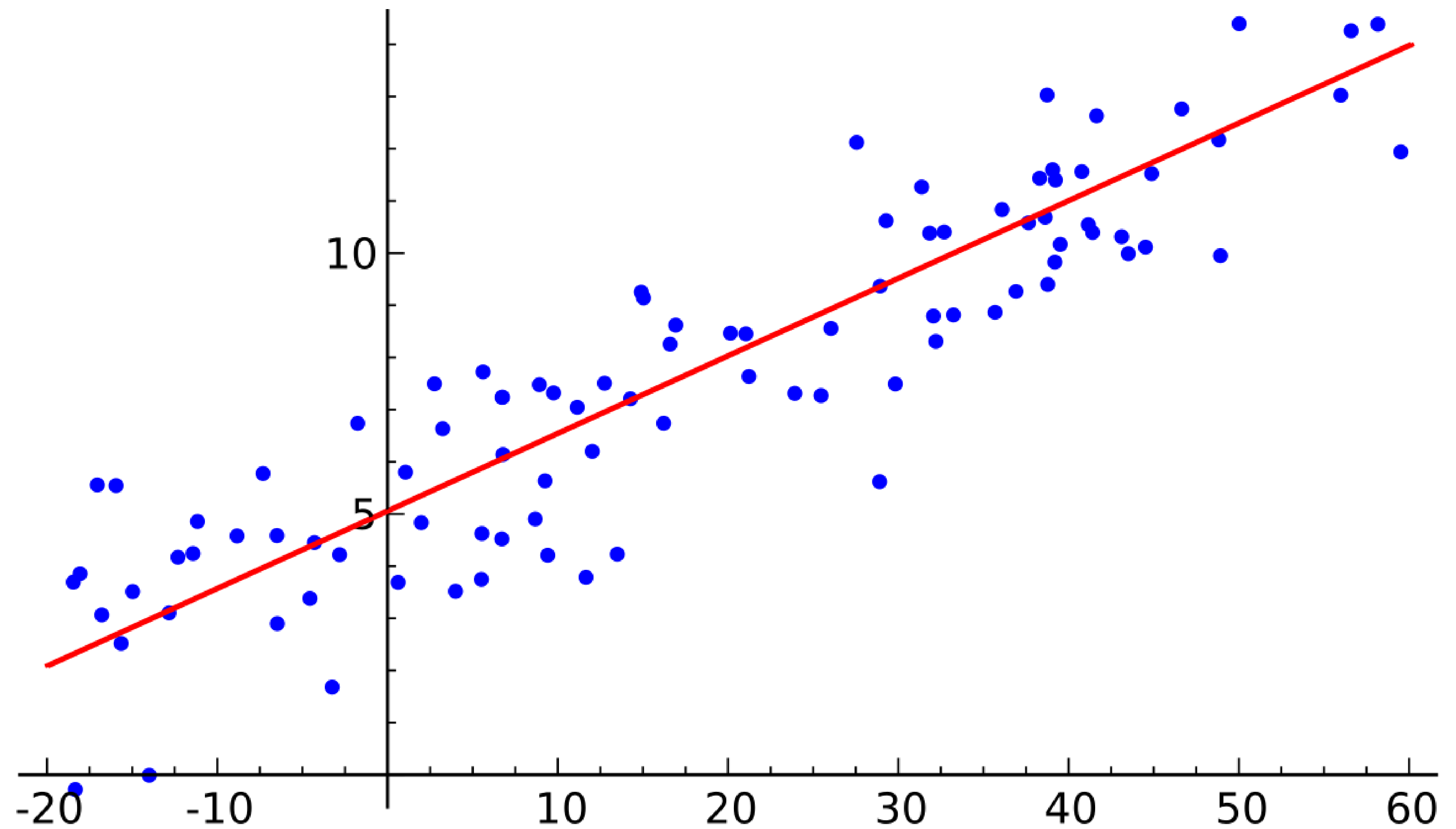
Mo Chen
SFU School of Computing Science

2021-09-27

Linear Regression

Recall: Machine Learning

- In modern terms:
  - Child machine: *Model*
  - Blank sheets: *Model parameters*
  - Teacher: *Loss function*



Input

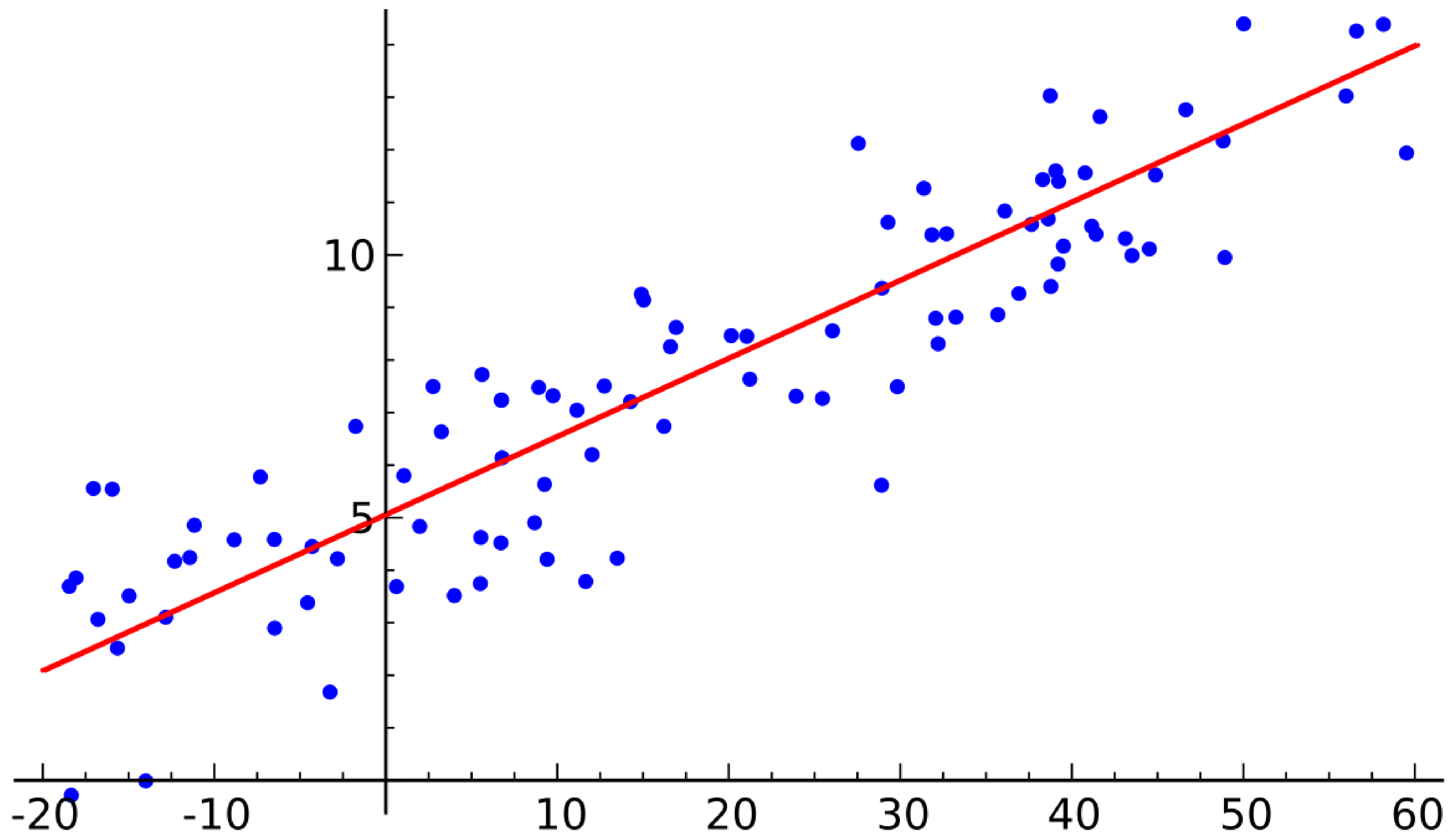
Predicted Output $\rightarrow$ $$\hat{y} = wx + b$$

$$L = (y - \hat{y})^2$$

Desired Output

Recall: Machine Learning

- In modern terms:
  - Child machine: *Model*
  - Blank sheets: *Model parameters*
  - Teacher: *Loss function*



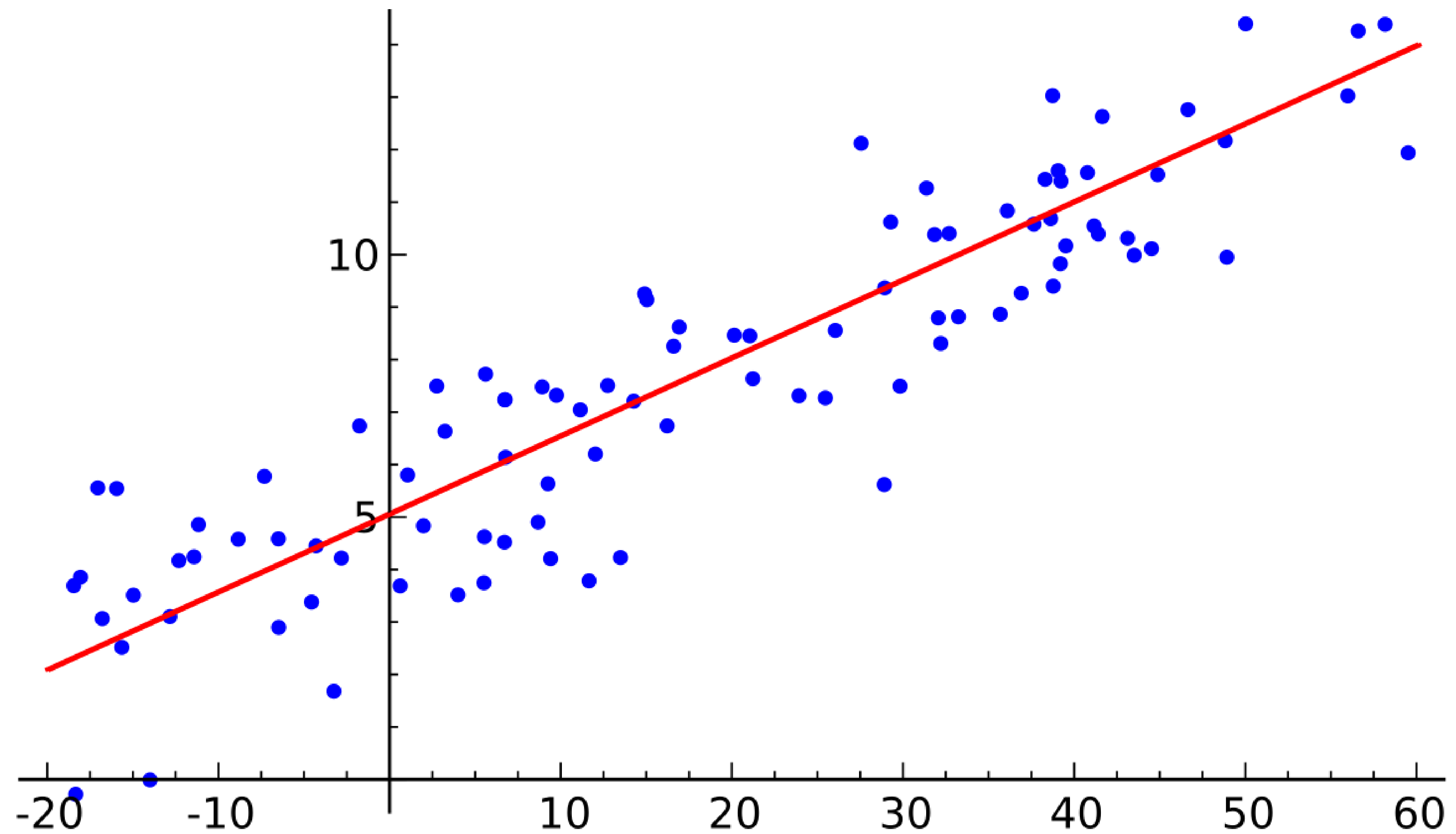
Parameters

Model $\longrightarrow$ $\boxed{\hat{y} = wx + b}$

Loss Function $\longrightarrow$ $L = (y - \hat{y})^2$

Recall: Machine Learning

- In modern terms:
  - Child machine: *Model*
  - Blank sheets: *Model parameters*
  - Teacher: *Loss function*

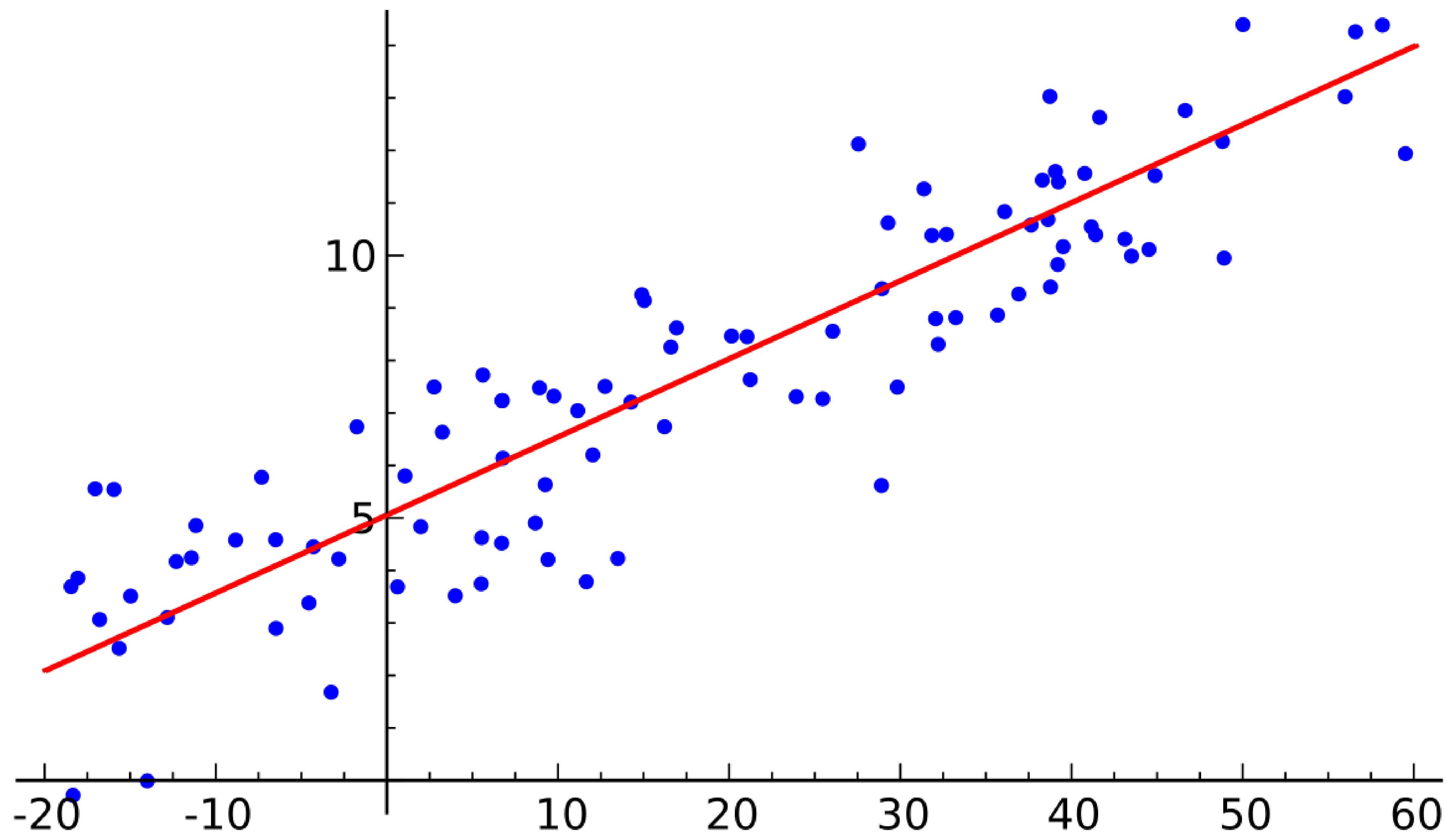


We want to find the parameter values that minimize the loss:

$$w^*, b^* = \arg\min_{w,b} L$$

Ordinary Least Squares

Given a dataset of input-output pairs (a.k.a. "observations"), $\{(x_i, y_i)\}_{i=1}^{N}$



We construct a model to predict output y ("labels") from input x ("data")

We choose a linear model:
$\hat{y} = wx + b$, where $\hat{y}$ is the predicted output ("prediction")

Denote this model as $f(x; w, b) \coloneqq wx + b$, where w and b are model parameters

We choose the following loss function: $L(w, b) = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ known as "square loss")

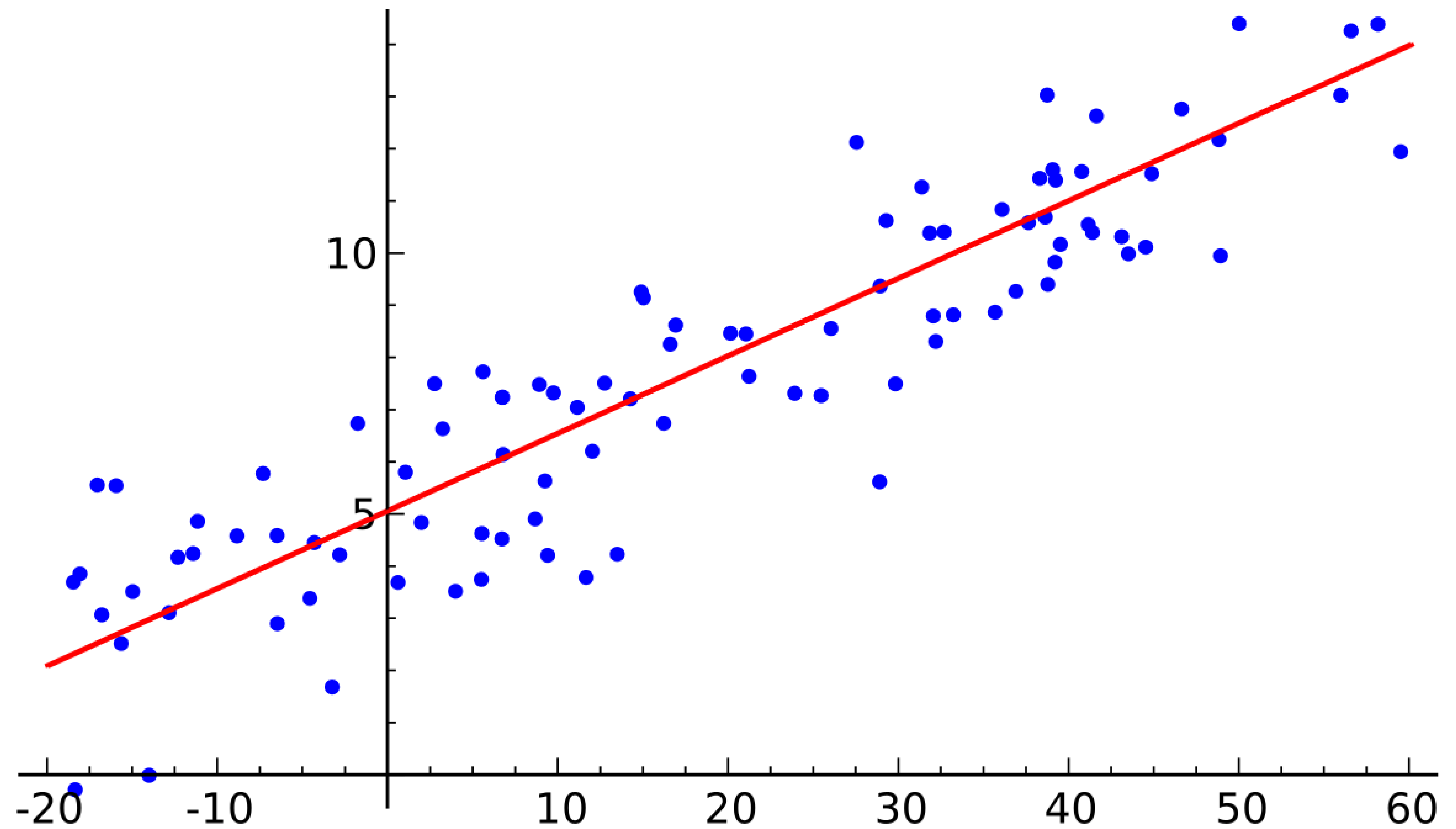
Ordinary Least Squares

Goal: Find the model parameters that minimizes the loss, i.e.:

$$\min_{w,b} L(w,b) = \min_{w,b} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

$$= \min_{w,b} \sum_{i=1}^{N} \big(y_i - f(x_i; w, b)\big)^2$$

$$= \min_{w,b} \sum_{i=1}^{N} \big(y_i - (wx_i + b)\big)^2$$



Ordinary Least Squares

What if the input has multiple dimensions?

Need a different model:

$$\hat{y} = \left(\sum_{j=1}^{n} w_j x_j\right) + b$$

In vector notation:

$$\hat{y} = \vec{w}^\top \vec{x} + b,$$

where $\vec{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix}$ and $\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$

More compactly:

$\hat{y} = \vec{w}^\top \vec{x}$, where $\vec{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_n \\ b \end{pmatrix}$ and $\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ 1 \end{pmatrix}$

Loss function remains the same:

$$L(\vec{w}) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{N} (y_i - \vec{w}^\top \vec{x}_i)^2$$

Ordinary Least Squares

We find the critical point by setting the gradient to zero:

$$\nabla_{\vec{w}} L(\vec{w}) = \frac{\partial L}{\partial w_j}(\vec{w}) = \vec{0}$$

For all $j \in \{1, \dots, n\}$, we have

$$\frac{\partial L}{\partial w_j} = 2\sum_{i=1}^{N}(y_i - \vec{w}^\top \vec{x}_i)\left(-x_{ij}\right) = 0$$

$$2\sum_{i=1}^{N}\left(-y_i x_{ij} + (\vec{w}^\top \vec{x}_i)x_{ij}\right) = 0$$

$$\sum_{i=1}^{N}(\vec{w}^\top \vec{x}_i)x_{ij} = \sum_{i=1}^{N} y_i x_{ij}$$

$$\sum_{i=1}^{N}\sum_{k=1}^{n} w_k x_{ik}\, x_{ij} = \sum_{i=1}^{N} y_i x_{ij}$$

$$\sum_{k=1}^{n} w_k \sum_{i=1}^{N} x_{ik} x_{ij} = \sum_{i=1}^{N} y_i x_{ij}$$

$$\underbrace{\begin{pmatrix} x_{1,1} & \cdots & x_{N,1} \\ \vdots & \ddots & \vdots \\ x_{1,n} & \cdots & x_{Nn} \end{pmatrix}}_{X^\top}\underbrace{\begin{pmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{Nn} \end{pmatrix}}_{X}\underbrace{\begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix}}_{\vec{w}} = \underbrace{\begin{pmatrix} x_{1,1} & \cdots & x_{N,1} \\ \vdots & \ddots & \vdots \\ x_{1,n} & \cdots & x_{Nn} \end{pmatrix}}_{X^\top}\underbrace{\begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}}_{\vec{y}}$$

So, this can be expressed concisely in matrix form: $X^\top X\vec{w} = X^\top\vec{y}$,

where $X = \begin{pmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_N^\top \end{pmatrix}, \vec{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$.

This linear system of equations is known as the “normal equations”

Hence, if X is full rank,

$$\vec{w} = (X^\top X)^{-1}X^\top\vec{y}$$

This is a critical point and might be a global minimum (still need to check this).

Ordinary Least Squares

It is often more convenient to derive this directly in matrix form.

Recall: the loss function is

$$L(\vec{w}) = \sum_{i=1}^{N} (y_i - \vec{y}_i)^2 = \sum_{i=1}^{N} (y_i - \vec{w}^\top \vec{x}_i)^2$$

Let $X = \begin{pmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_N^\top \end{pmatrix}, \vec{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$. We can rewrite it in terms of matrices:

$L(\vec{w}) = \|\vec{y} - X\vec{w}\|_2^2$
$= (\vec{y} - X\vec{w})^\top (\vec{y} - X\vec{w})$
$= \vec{y}^\top y - (X\vec{w})^\top \vec{y} - \vec{y}^\top (X\vec{w}) + (X\vec{w})^\top (X\vec{w})$
$= \vec{y}^\top y - 2\vec{y}^\top (X\vec{w}) + (\vec{w}^\top X^\top)(X\vec{w})$
$= \vec{y}^\top y - (2\vec{y}^\top X)\vec{w} + \vec{w}^\top (X^\top X)\vec{w}$

Aside: Matrix Derivatives

It is possible to take partial derivatives w.r.t. entire vectors or matrices at once. These are known as matrix derivatives.

Common Identities:

$$\frac{\partial(\vec{a}^\top\vec{x})}{\partial\vec{x}} := \begin{pmatrix} \frac{\partial}{\partial x_1}(\vec{a}^\top\vec{x}) \\ \vdots \\ \frac{\partial}{\partial x_n}(\vec{a}^\top\vec{x}) \end{pmatrix} = \begin{pmatrix} \frac{\partial}{\partial x_1}\left(\sum_{i=1}^{n} a_i x_i\right) \\ \vdots \\ \frac{\partial}{\partial x_n}\left(\sum_{i=1}^{n} a_i x_i\right) \end{pmatrix} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} = \vec{a}$$

$$\frac{\partial(A\vec{x})}{\partial\vec{x}} := \begin{pmatrix} \frac{\partial}{\partial x_1}(A\vec{x})_1 & \cdots & \frac{\partial}{\partial x_1}(A\vec{x})_m \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_n}(A\vec{x})_1 & \cdots & \frac{\partial}{\partial x_n}(A\vec{x})_m \end{pmatrix} = \begin{pmatrix} \frac{\partial}{\partial x_1}\left(\sum_{i=1}^{n} a_{1i} x_i\right) & \cdots & \frac{\partial}{\partial x_1}\left(\sum_{i=1}^{n} a_{mi} x_i\right) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_n}\left(\sum_{i=1}^{n} a_{1i} x_i\right) & \cdots & \frac{\partial}{\partial x_n}\left(\sum_{i=1}^{n} a_{mi} x_i\right) \end{pmatrix} = \begin{pmatrix} a_{11} & \cdots & a_{m1} \\ \vdots & \ddots & \vdots \\ a_{1n} & \cdots & a_{mn} \end{pmatrix} = A^\top$$

"Jacobian"

Aside: Matrix Derivatives

Sum Rule:

$$\frac{\partial\left(\vec{f}(\vec{x})+\vec{g}(\vec{x})\right)}{\partial\vec{x}}=\frac{\partial\vec{f}}{\partial\vec{x}}+\frac{\partial\vec{g}}{\partial\vec{x}}$$

Product Rule (for inner products):

$$\frac{\partial\left(\vec{f}(\vec{x})^\top\vec{g}(\vec{x})\right)}{\partial\vec{x}}=\frac{\partial\vec{f}}{\partial\vec{x}}\vec{g}(\vec{x})+\frac{\partial\vec{g}}{\partial\vec{x}}\vec{f}(\vec{x})$$

Hence,

$$\frac{\partial(\vec{x}^\top A\vec{x})}{\partial\vec{x}}=\frac{\partial}{\partial\vec{x}}\left(\vec{x}^\top(A\vec{x})\right)=\frac{\partial\vec{x}}{\partial\vec{x}}(A\vec{x})+\frac{\partial(A\vec{x})}{\partial\vec{x}}\vec{x}=A\vec{x}+A^\top\vec{x}=(A+A^\top)\vec{x}$$

More available at https://en.wikipedia.org/wiki/Matrix_calculus#Identities (no need to memorize all of them, but helpful to remember the common ones like $\frac{\partial\left(\vec{a}^\top\vec{x}\right)}{\partial\vec{x}}$, $\frac{\partial(A\vec{x})}{\partial\vec{x}}$ and $\frac{\partial\left(\vec{x}^\top A\vec{x}\right)}{\partial\vec{x}}$.

Ordinary Least Squares

Recall:

$$\begin{aligned} L(\vec{w}) &= \|\vec{y} - X\vec{w}\|_2^2 \\ &= (\vec{y} - X\vec{w})^\top(\vec{y} - X\vec{w}) \\ &= \vec{y}^\top y - (X\vec{w})^\top\vec{y} + \vec{y}^\top(X\vec{w}) + (X\vec{w})^\top(X\vec{w}) \\ &= \vec{y}^\top y - 2\vec{y}^\top(X\vec{w}) + (\vec{w}^\top X^\top)(X\vec{w}) \\ &= \vec{y}^\top y - (2\vec{y}^\top X)\vec{w} + \vec{w}^\top(X^\top X)\vec{w} \end{aligned}$$

$$\frac{\partial L}{\partial \vec{w}} = \frac{\partial(\vec{y}^\top\vec{y})}{\partial \vec{w}} - \frac{\partial\left((2X^\top\vec{y})^\top\vec{w}\right)}{\partial \vec{w}} + \frac{\partial(\vec{w}^\top(X^\top X)\vec{w})}{\partial \vec{w}} = 0$$

$$\begin{aligned} &0 - 2X^\top\vec{y} + (X^\top X + (X^\top X)^\top)\vec{w} = 0 \\ &-2X^\top\vec{y} + 2(X^\top X)\vec{w} = 0 \\ &2(X^\top X)\vec{w} = 2X^\top\vec{y} \\ &(X^\top X)\vec{w} = X^\top\vec{y} \end{aligned}$$

Normal equations

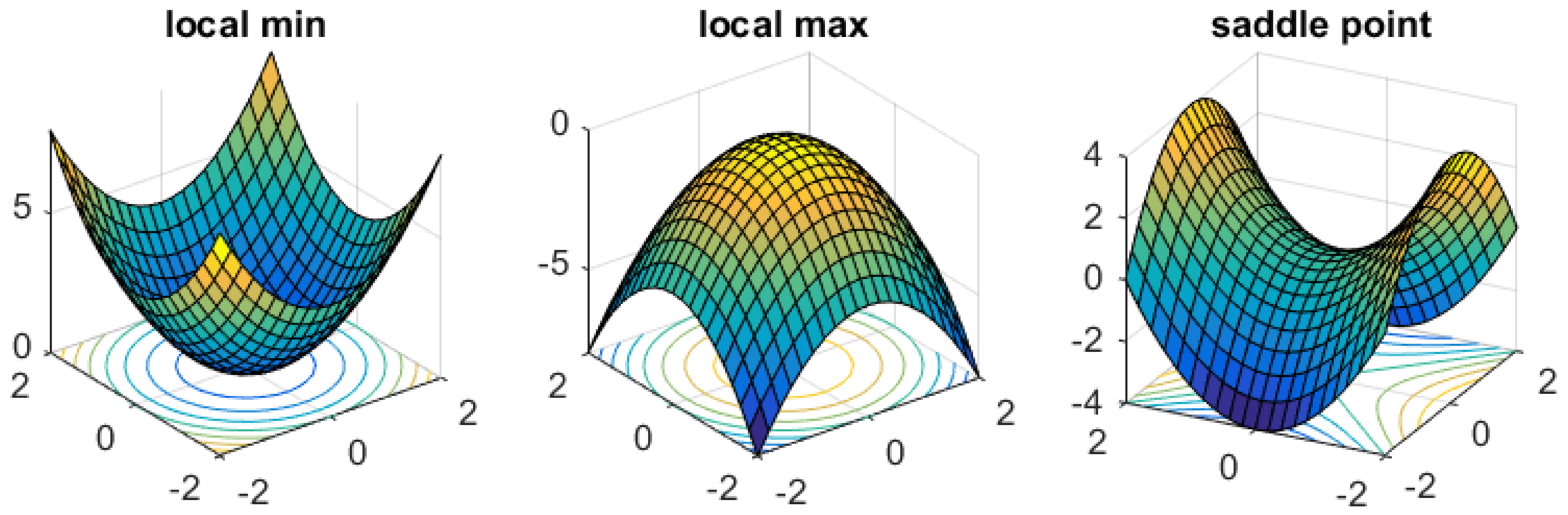
If X is full-rank,

$$\vec{w} = (X^\top X)^{-1}X^\top\vec{y}$$

Same as before

Ordinary Least Squares

Now we need to check if $\vec{w} = (X^\top X)^{-1} X^\top \vec{y}$ is indeed a global minimum.



To do so, it suffices to show that the loss function is convex (in which case any critical point is a local and global minimum).

Ordinary Least Squares

Recall: $\frac{\partial L}{\partial \vec{w}} = -2X^\top \vec{y} + 2(X^\top X)\vec{w}$

$$\frac{\partial^2 L}{\partial \vec{w} \partial \vec{w}^\top} = \frac{\partial}{\partial \vec{w}}\left(\frac{\partial L}{\partial \vec{w}}\right)$$

$$\begin{aligned}
&= \frac{\partial}{\partial \vec{w}}(-2X^\top \vec{y} + 2(X^\top X)\vec{w}) \\
&= \frac{\partial}{\partial \vec{w}}(-2X^\top \vec{y}) + \frac{\partial}{\partial \vec{w}}(2(X^\top X)\vec{w}) \\
&= 0 + 2(X^\top X)^\top \\
&= 2X^\top X \\
&\succcurlyeq 0
\end{aligned}$$

Since the Hessian is always positive semi-definite, the loss function is convex.

So any critical point is a local and global minimum, so the critical point $\vec{w} = (X^\top X)^{-1} X^\top \vec{y}$ minimizes the loss function, which will now be denoted as $\vec{w}^*$.

Ordinary Least Squares: Summary

Model: $\hat{y} = \vec{w}^\top \vec{x}$

Parameters: $\vec{w}$

Loss function: $L(\vec{w}) = \sum_{i=1}^{N} (y_i - \vec{y}_i)^2 = \sum_{i=1}^{N} (y_i - \vec{w}^\top \vec{x}_i)^2$

where $X = \begin{pmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_N^\top \end{pmatrix}, \vec{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$

Optimal parameters: $\vec{w}^* \coloneqq \arg\min_{\vec{w}} L(\vec{w}) = (X^\top X)^{-1} X^\top \vec{y}$

The process of finding the optimal parameters is known as **training**.

The process of computing predictions $\hat{y}$ for new input $\vec{x}$ is known as **testing**.

Pseudoinverse

The matrix $(X^\top X)^{-1}X^\top$ is known as the pseudoinverse (or Moore-Penrose inverse) of X, and is usually denoted as $X^\dagger$.

Rationale:

- When # of data points (N) = # of input dimensions (n), can fit data perfectly.
  - In other words, $\min_{\vec{w}} \|\vec{y} - X\vec{w}\|_2^2$, so finding the optimal $\vec{w}$ amounts to solving the linear system of equations $X\vec{w} = \vec{y}$. If X is full-rank, $\vec{w}^* = X^{-1}\vec{y}$.
  - X^{-1} can be interpreted as an operator that solves a linear system of equations.
- When # of data points (N) > # of input dimensions (n), in general cannot fit data perfectly.
  - As shown earlier, $\vec{w}^* = (X^\top X)^{-1}X^\top\vec{y} = X^\dagger\vec{y}$. If we compare these formulas $\vec{w}^* = X^{-1}\vec{y}$ vs. $\vec{w}^* = X^\dagger\vec{y}$, $X^\dagger$ can be seen as a generalization of X^{-1} to the case where data cannot be fit perfectly.
- $X^\dagger$ can be interpreted as an operator that finds the best possible solution to a linear system of equations that has no solution.

Pseudoinverse

How to compute the pseudoinverse $X^\dagger = (X^\top X)^{-1} X^\top$?

We can do so using SVD:

Let $U\Sigma V^\top$ be the singular value decomposition (SVD) of X.

$$
\begin{aligned}
X^\dagger &= (X^\top X)^{-1} X^\top \\
&= (V\Sigma U^\top U\Sigma V^\top)^{-1} V\Sigma U^\top \\
&= (V\Sigma\Sigma V^\top)^{-1} V\Sigma U^\top \\
&= \left((V^\top)^{-1}\Sigma^{-1}\Sigma^{-1}V^{-1}\right) V\Sigma U^\top \\
&= (V^\top)^{-1}\Sigma^{-1}\Sigma^{-1}\Sigma U^\top \\
&= (V^\top)^{-1}\Sigma^{-1} U^\top \\
&= (V^\top)^\top \Sigma^{-1} U^\top \\
&= V\Sigma^{-1} U^\top
\end{aligned}
$$

Multiple Output Linear Regression

What if the output has multiple dimensions as well?

$$\hat{\vec{v}} \coloneqq f(\vec{x}; W) = W\vec{x}, \text{ where } W \coloneqq \begin{pmatrix} \vec{w}_1^\top \\ \vdots \\ \vec{w}_m^\top \end{pmatrix} = \begin{pmatrix} w_{11} & \cdots & w_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ w_{m1} & \cdots & w_{mn} & b_m \end{pmatrix} \text{ and } \vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ 1 \end{pmatrix}$$

Note on terminology: This is different from "multiple linear regression", which refers to linear regression with multiple independent variables/regressors/input dimensions. In machine learning, the input is almost always multi-dimensional, so multiple linear regression is simply referred to as linear regression.

In statistics, known as the "general linear model".

Multiple Output Linear Regression

Because the output is now a vector, the loss must operate on vectors. The analogous loss to the square loss for vectors is the squared l_2 loss:

$$
\begin{aligned}
L(W) &= \sum_{i=1}^{N} \left\| \vec{v}_i - \hat{\vec{v}}_i \right\|_2^2 \\
&= \sum_{i=1}^{N} \| \vec{v}_i - f(\vec{x}_i; W) \|_2^2 \\
&= \sum_{i=1}^{N} \| \vec{v}_i - W\vec{x} \|_2^2
\end{aligned}
$$

Loss function in matrix form:

Let $X = \begin{pmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_N^\top \end{pmatrix}$ and $\vec{y} = \begin{pmatrix} \vec{v}_1^\top \\ \vdots \\ \vec{v}_N^\top \end{pmatrix} \coloneqq (\vec{y}_1 \quad \cdots \quad \vec{y}_m)$

$$L(W) = \|Y - XW^\top\|_F^2 = \sum_{i=1}^{m} \|\vec{y}_i - X\vec{w}_i\|_2^2$$

It turns out that the optimal parameters are similar (show this as an exercise):

$$W^{*\top} = (X^\top X)^{-1} X^\top Y$$

Compare to the single output case:

$$\vec{w}^* = (X^\top X)^{-1} X^\top \vec{y}$$

Linear Regression in Action

https://colab.research.google.com/drive/17DXTj9mp4J9Yi_YlFol4Nm5-UrtghmAT?usp=sharing

(This is an important part of the lecture - don’t skip over this if you are going over this after the lecture)

Features (a.k.a. Basis Functions)

Polynomial features are an example of the general notion of features.

In general, a feature map $\phi\colon \mathbb{R}^n \to \mathbb{R}^{n'}$ (where $n' \geq n$ typically) converts raw data into features, which are used in place of the raw data as input to a model.

Example in the context of linear regression:

$$\hat{y} = \vec{w}^\top \phi(\vec{x})$$

$$\phi(\vec{x}) \coloneqq \begin{pmatrix} \phi_1(\vec{x}) \\ \vdots \\ \phi_{n'}(\vec{x}) \end{pmatrix} = \begin{pmatrix} \log(x_1) \\ \vdots \\ \log(x_n) \end{pmatrix}$$

Each component function ϕ_i is known as a basis function.

Features allow the model prediction to depend *non-linearly* on the data without changing the training procedure.

Polynomial Features

Polynomial features map raw data to monomials of varying degrees.

E.g.: Polynomial features up to degree 2

$$\phi(\vec{x}) = (\underbrace{1}_{\text{degree 0}} \quad \underbrace{x_1 \quad \cdots \quad x_n}_{\text{degree 1}} \quad \underbrace{x_1^2 \quad x_1 x_2 \quad \cdots \quad x_1 x_n \quad x_2^2 \quad x_2 x_3 \quad \cdots \quad x_n^2}_{\text{degree 2 (don't forget the cross terms!)}})^\top$$

In general, perform binomial expansion of $(1 + x_1 + \cdots + x_n)^d$ and make each term (without the coefficient) a basis function.

Generalization

Training set or **train set**: set of observations used for training / fitting the model

Testing set or **test set**: set of new observations

Goal of machine learning is to achieve low loss / high accuracy on the **testing set**.

The testing set should approximate how much loss we expect the model to incur on unseen observations in the real world.

Never, ever train on the testing set! (Otherwise the testing set becomes pointless)

In fact, the testing set should not be used at any point during the development process. It should only be used **once** when you are completely satisfied with the model and are ready to deploy it.

Generalization

Training / train loss and **testing / test loss**: loss on the training set and testing set respectively.

Testing loss is generally higher than training loss.

When the model achieves low training loss and low testing loss, the model is said to **generalize**.

Generalization gap: Testing Loss − Training Loss

Overfitting: When the model is fitted *too* well to the data, i.e.: when testing loss is much higher than training loss

Underfitting: When the model is not fitted well enough, i.e.: when both training and testing loss are high and the generalization gap is small

Model capacity / expressive power: The flexibility of the model in fitting the data. A model that can fit more “shapes” of the data distribution is said to be more **expressive** / have more **model capacity / expressive power**.