# Deep Learning (CS 470, CS 570)

## Module 2, Lecture 1: Regression

# Machine Learning Sub-branches

- Supervised Learning
  - Classification : PLA
  - Regression
- Unsupervised Learning
  - Clustering : k-mean
  - Dimensionality reduction
- Semi-supervised Learning
- Reinforcement Learning
- Deep Learning: different ML approach
  - Supervised
  - Unsupervised
  - Semi-supervised

# Linear Regression

**Problem:** Given customer income vs. credit card spending data for **N** customers, a credit card company wants to predict credit limits for the future customers.

**Table:** Customer income vs. spending

| Customer ID | X (income) | Y (spending) |
|---|---|---|
| 1 | 90,000 | 40,000 |
| 2 | 1,50,000 | 72,000 |
| … | … | … |
| N | 67,000 | 33,000 |

**How?**

- Learn if there is any relation between input and output variables.
- Use it for prediction in future.

Some ML problems require to predict a value of a continuous variable in the output. A continuous variable can take any fraction value such as time. These set of problems are called regression. Above is an example of a regression problem.

# Machine Algorithm Steps

- **Collect data:** $(\boldsymbol{x}^1, y^1), (\boldsymbol{x}^2, y^2), \ldots, (\boldsymbol{x}^{\mathrm{N}}, y^{\mathrm{N}})$

- **Select model:** Defining Hypothesis set,

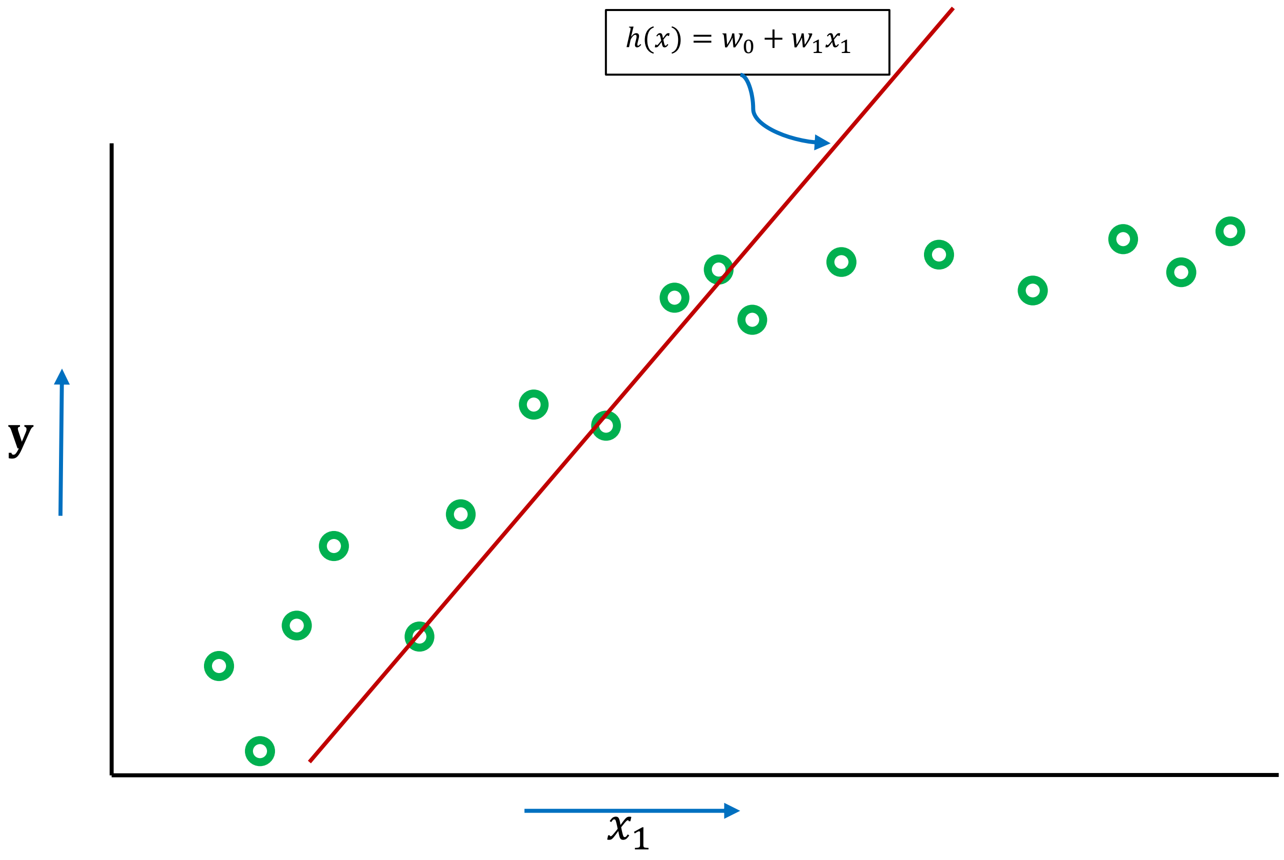
  Get to know number of parameters of the function and structure of the function

- **Model training:** Learning the function parameters. **Optimization**

  Learning function $g: \boldsymbol{x} \rightarrow y,$ g$\in H$ , where $g$ is an approximation of $f$

- **Model validation:** Testing accuracy of the model on the test data

# Linear Regression




Given data: $(\boldsymbol{x}^1, y^1), (\boldsymbol{x}^2, y^2), \ldots, (\boldsymbol{x}^N, y^N)$

Predict: $y$ from the value of $x$

Learn: $h(.)$ such that $y \leftarrow h(\boldsymbol{x})$

Borrowing the equation of a line from PLA classifier:

$$h(\boldsymbol{x}) = w_0 + w_1 x_1$$

In high dimension: $h(\boldsymbol{x}) = \sum_{i=0}^{d} w_i x_i$

In matrix form: $h(\boldsymbol{x}) = \boldsymbol{w}^T \mathbf{x}$

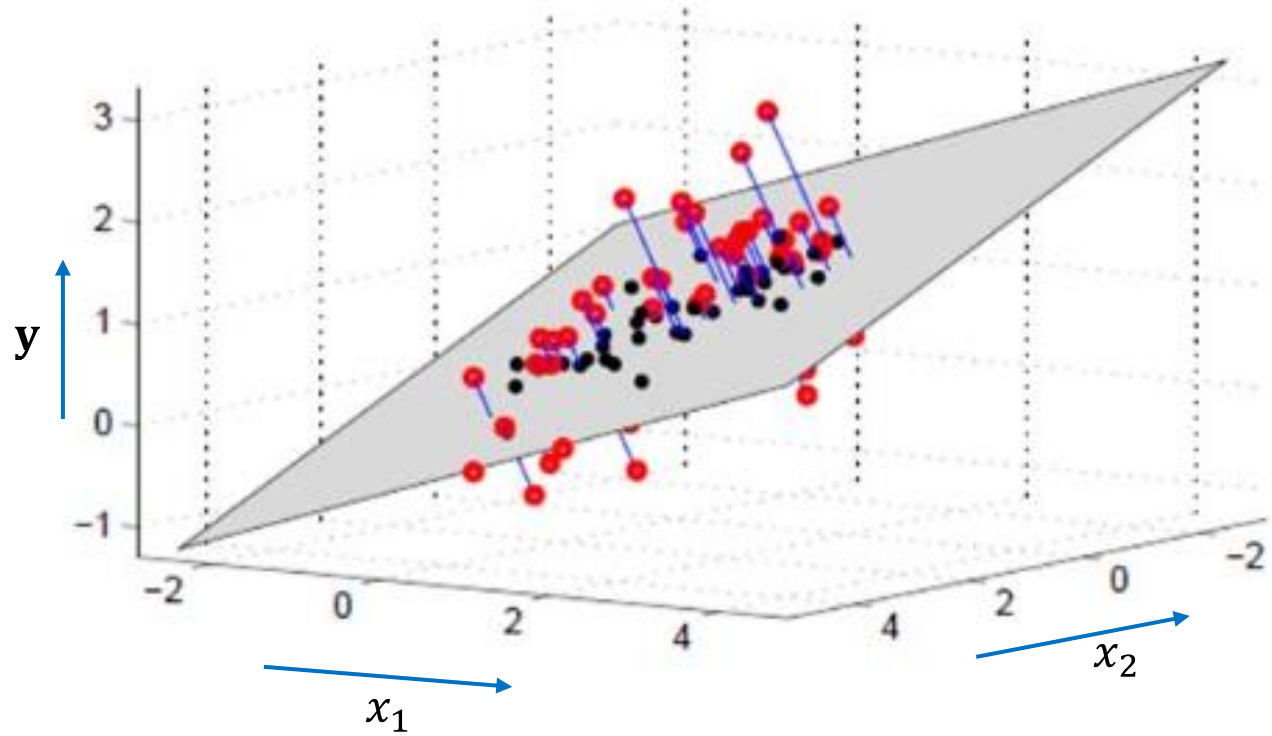
$\boldsymbol{x}$ is the independent, and $y$ is the dependent variable

We are trying to fit a straight line to predict the value of $y$

**How to get the best fit of the strait line on the given data set ?**

**Optimization !**

# Linear Regression: High Dimension



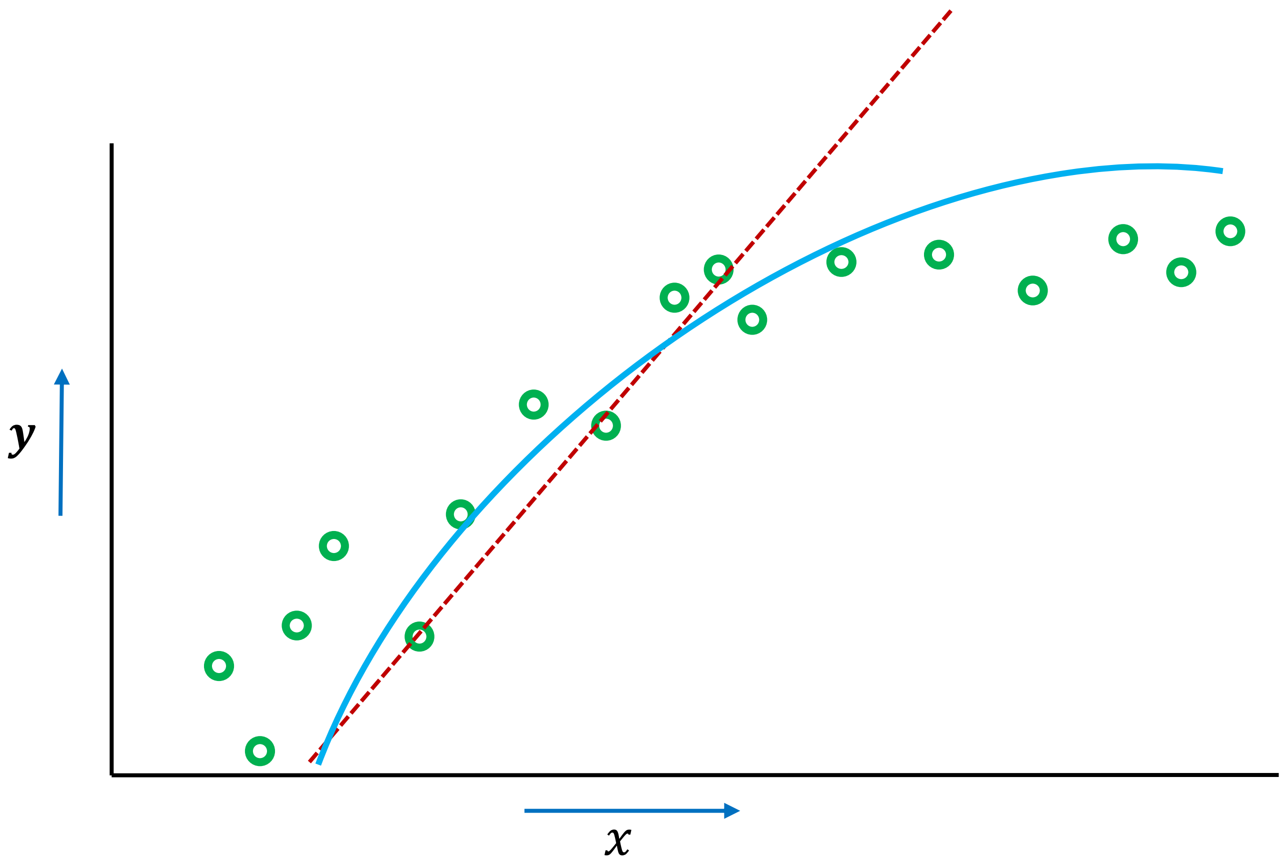

For 2 dimensional input the equation becomes:

$$h(x) = w_0 + w_1 x_1 + w_2 x_2$$

In two dimensional case where each input point is defined by two variables $\{x_1, x_2\}$, the regression function $h(\,)$ becomes a plane in a 3D space where the three axis of the 3D space are $x_1$, $x_2$, and $y$. In the generic $d$ dimensional case $h(\,)$ is a $d - 1$ dimensional hyper-plane.

**Source:** https://slideplayer.com/slide/11494650/

# Linear Regression: Non Linear Cases



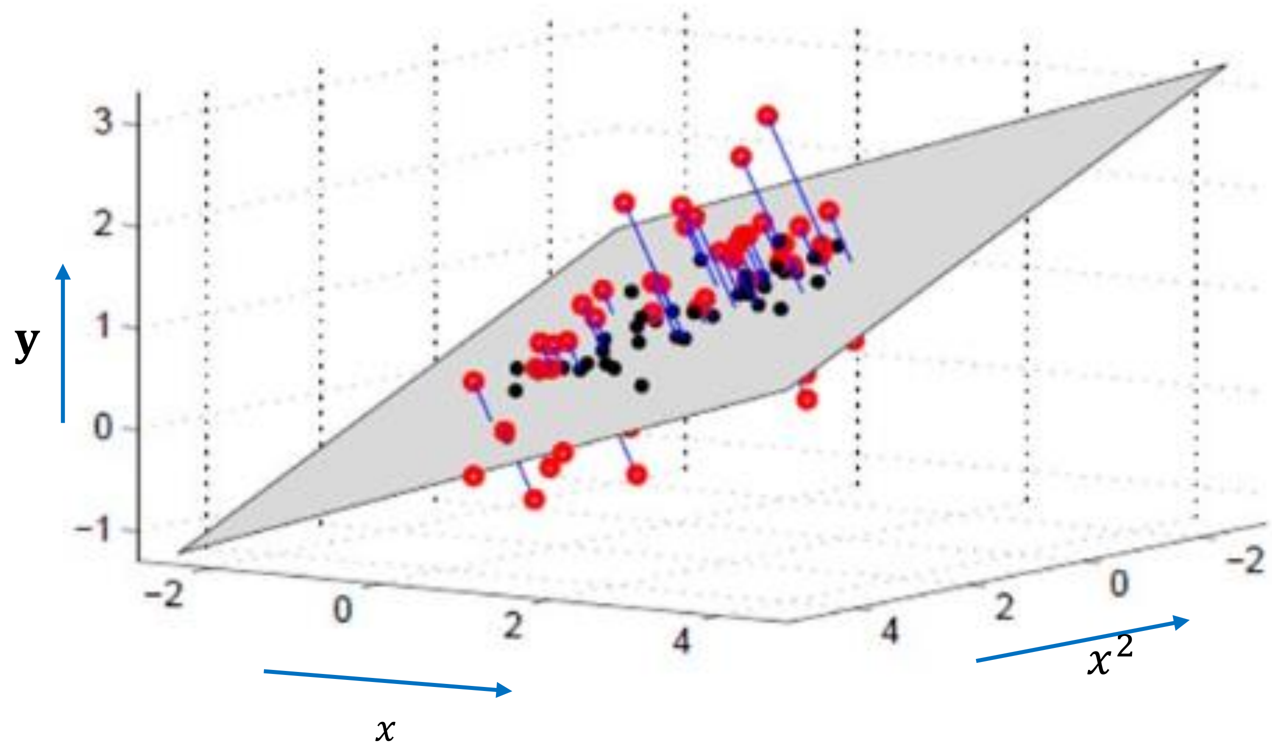

Transform the data to non linear space:

$$h(x) = w_0 + w_1 x + w_2 x^2$$

In some situations linear $h(\,)$ is not a good approximation of the data point distribution such as the case shown in this slide. In this example a better way approximate the green points is the blue curve which cannot be represented by a linear equation. In these cases, we need a nonlinear equation where there is at least one higher order term (square, cube etc.) for at least one of the input variables. The above equation is an example of a nonlinear equation.

**Source:** https://slideplayer.com/slide/11494650/

# Linear Regression: Non Linear Cases




Transform the data points to a nonlinear space:

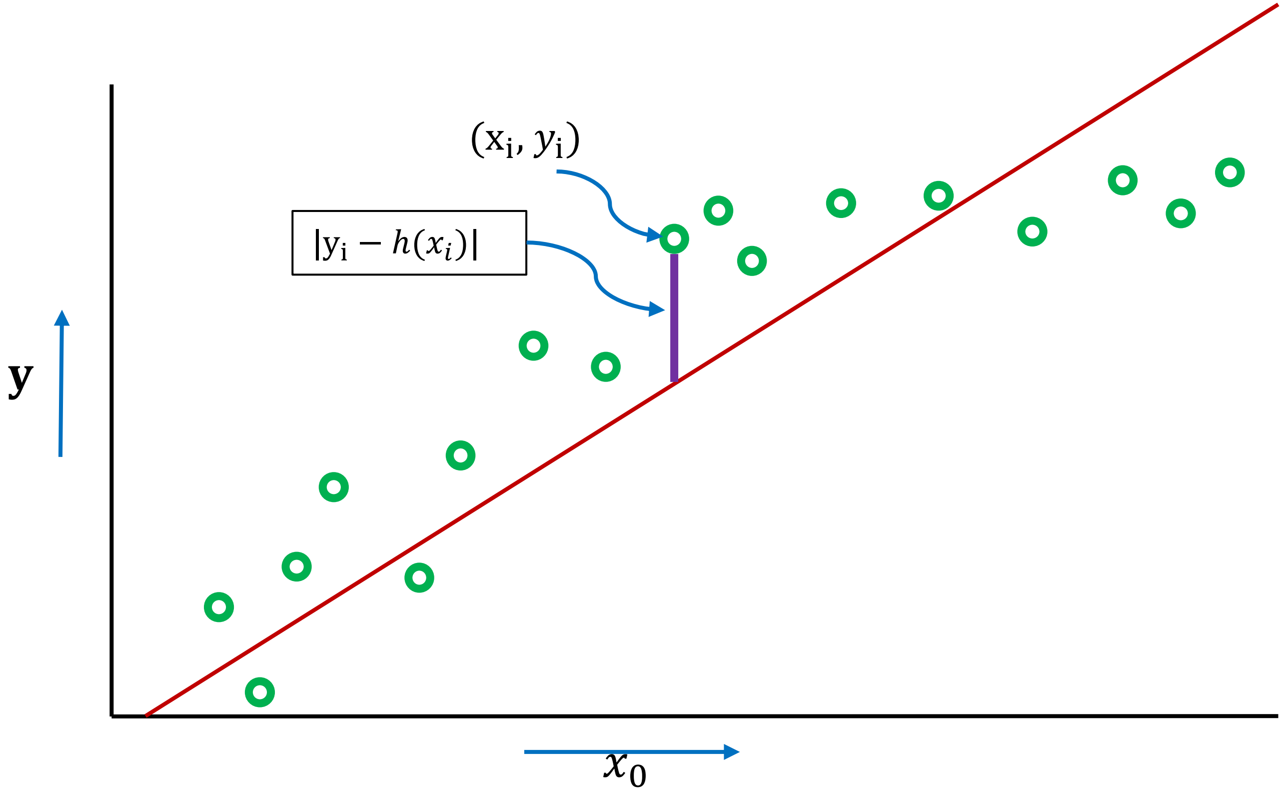
$$h(x) = w_0 + w_1 x + w_2 x^2$$

Linear regression in a nonlinear space

Non linear equation can be handled by projecting the data point in a space define by the nonlinear variables and solving a linear regression problem in that space. The next slides discuss a technique to solve linear regression problem.

**Source:** https://slideplayer.com/slide/11494650/

# Linear Regression: Optimization




Prediction error for $i_{th}$ point: $|y_i - h(x_i)|$

Prediction error for **N** points: $\sum_{i=1}^{N} |y_i - h(x_i)|$

Sum Square Error (SSE): $\frac{1}{N}\sum_{i=1}^{N} (y_i - h(x_i))^2$

Root Mean Square Error(RMSE): $\sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - h(x_i))^2}$

**Optimization:** minimize SSE or RMSE

# Linear Regression: Optimization

**SSE:** $$E = \frac{1}{N}\sum_{\mathrm{i=1}}^{\mathrm{N}}(h(x_i) - y_i)^2$$

$$E = \frac{1}{N}\sum_{\mathrm{i=1}}^{\mathrm{N}}(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}_i - y_i)^2$$ , as $h(x) = \boldsymbol{w}^T\mathbf{x}$

$$\mathbf{X} = \begin{bmatrix} \leftarrow & x_0 & \rightarrow \\ \leftarrow & x_1 & \rightarrow \\ & \vdots & \\ \leftarrow & x_{\mathrm{N}} & \rightarrow \end{bmatrix} \qquad \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

**Matrix form:** $$E = \frac{1}{N}\|(\mathbf{Xw} - \boldsymbol{y})\|^2$$

This slide derives SSE for N points in the matrix form.

# Linear Regression: Optimization

**Matrix form:** $$E = \frac{1}{N}\|(\mathbf{X}\boldsymbol{w} - \boldsymbol{y})\|^2$$

**Derivative w.r.t. W and equating to zero:** $$\Delta E = \frac{2}{N}\mathbf{X}^T(\mathbf{X}\boldsymbol{w} - \boldsymbol{y}) = 0$$

$$\Rightarrow \mathbf{X}^T\mathbf{X}\boldsymbol{w} = \mathbf{X}^T\boldsymbol{y}$$

$$\Rightarrow \boldsymbol{w} = \boxed{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T}\boldsymbol{y}$$

**Pseudo inverse**

$$\boxed{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T}\mathbf{X} = \mathbf{I}$$

We know that when the derivative of a function equates to zero, it can represent the function minima. Therefore, we computed the derivative of the error function and solved for $\boldsymbol{w}$ when the derivative is zero. This gives us the $h(\,)$ that best approximates (lowest error) the training data points. Remember, we need to compute matrix derivation here. We learned this in the first section of the course.

# Additional Reading

Linear Regression