Linear Regression: One-Dimensional Case

- **Given**: a set of \( N \) input-response pairs

- The inputs \((x)\) and the responses \((y)\) are one dimensional scalars

- **Goal**: Model the relationship between \( x \) and \( y \)
Let’s assume the relationship between $x$ and $y$ is linear.
Let’s assume the relationship between $x$ and $y$ is linear.

Linear relationship can be defined by a straight line with parameter $w$.

Equation of the straight line: $y = wx$. 
The line may not fit the data *exactly*
The line may not fit the data *exactly*

But we can try making the line a *reasonable approximation*
Linear Regression: One-Dimensional Case

- The line may not fit the data \textit{exactly}
- But we can try making the line a \textit{reasonable approximation}
- Error for the pair \((x_i, y_i)\) pair: \(e_i = y_i - wx_i\)
The line may not fit the data \textit{exactly}.

But we can try making the line a \textit{reasonable approximation}.

\textbf{Error} for the pair \((x_i, y_i)\) pair: \(e_i = y_i - wx_i\)

The \textbf{total squared error}: \(E = \sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - wx_i)^2\)
Linear Regression: One-Dimensional Case

- The line may not fit the data exactly.
- But we can try making the line a reasonable approximation.
- Error for the pair \((x_i, y_i)\) pair: \(e_i = y_i - wx_i\)
- The total squared error: \(E = \sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - wx_i)^2\)
- The best fitting line is defined by \(w\) minimizing the total error \(E\)
Linear Regression: One-Dimensional Case

- The line may not fit the data exactly.
- But we can try making the line a reasonable approximation.
- Error for the pair \((x_i, y_i)\) pair: \(e_i = y_i - wx_i\)
- The total squared error: \(E = \sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - wx_i)^2\)
- The best fitting line is defined by \(w\) minimizing the total error \(E\).
- Just requires a little bit of calculus to find it (take derivative, equate to zero..)
Analogy to line fitting: In higher dimensions, we will fit hyperplanes.

For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.
Linear Regression: In Higher Dimensions

- **Analogy to line fitting**: In higher dimensions, we will fit hyperplanes.
- For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.

Many planes are possible. Which one is the best?
Linear Regression: In Higher Dimensions

- **Analogy to line fitting:** In higher dimensions, we will fit hyperplanes.
- For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.

Many planes are possible. Which one is the best?

- **Intuition:** Choose the one which is (on average) closest to the responses $Y$. 
Analogy to line fitting: In higher dimensions, we will fit hyperplanes.

For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.

Many planes are possible. Which one is the best?

Intuition: Choose the one which is (on average) closest to the responses \( Y \).

Linear regression uses the sum-of-squared error notion of closeness.
**Analogy to line fitting:** In higher dimensions, we will fit hyperplanes. For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.

Many planes are possible. Which one is the best?

**Intuition:** Choose the one which is (on average) closest to the responses $Y$.

- Linear regression uses the sum-of-squared error notion of closeness.
- Similar intuition carries over to higher dimensions too.
**Analogy to line fitting:** In higher dimensions, we will fit hyperplanes.

For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.

Many planes are possible. Which one is the best?

**Intuition:** Choose the one which is (on average) closest to the responses $Y$.

- Linear regression uses the sum-of-squared error notion of closeness.
- Similar intuition carries over to higher dimensions too.
  - Fitting a $D$-dimensional hyperplane to the data.
**Linear Regression: In Higher Dimensions**

- **Analogy to line fitting:** In higher dimensions, we will fit hyperplanes.
- For 2-dim. inputs, linear regression fits a 2-dim. plane to the data.

Many planes are possible. Which one is the best?

- **Intuition:** Choose the one which is (on average) closest to the responses $Y$.
  - Linear regression uses the sum-of-squared error notion of closeness.
- Similar intuition carries over to higher dimensions too.
  - Fitting a $D$-dimensional hyperplane to the data.
  - Hard to visualize in pictures though..
Analogy to line fitting: In higher dimensions, we will fit hyperplanes. For 2-dim. inputs, linear regression fits a 2-dim. plane to the data. Many planes are possible. Which one is the best? Intuition: Choose the one which is (on average) closest to the responses $Y$. Linear regression uses the sum-of-squared error notion of closeness. Similar intuition carries over to higher dimensions too. Fitting a $D$-dimensional hyperplane to the data. Hard to visualize in pictures though.

The hyperplane is defined by parameters $w$ (a $D \times 1$ weight vector).
Given training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$

- Inputs $x_i$: $D$-dimensional vectors ($\mathbb{R}^D$), responses $y_i$: scalars ($\mathbb{R}$)
Given training data $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$

- Inputs $x_i$: $D$-dimensional vectors ($\mathbb{R}^D$), responses $y_i$: scalars ($\mathbb{R}$)

- The linear model: response is a linear function of the model parameters

$$y = f(x, w) = b + \sum_{j=1}^{M} w_j \phi_j(x)$$
Given training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$

Inputs $x_i$: $D$-dimensional vectors ($\mathbb{R}^D$), responses $y_i$: scalars ($\mathbb{R}$)

The linear model: response is a linear function of the model parameters

$$y = f(x, w) = b + \sum_{j=1}^{M} w_j \phi_j(x)$$

$w_j$’s and $b$ are the model parameters ($b$ is an offset)

Parameters define the mapping from the inputs to responses
Linear Regression: In Higher Dimensions (Formally)

- Given training data \( \mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \)

- Inputs \( x_i \): \( D \)-dimensional vectors (\( \mathbb{R}^D \)), responses \( y_i \): scalars (\( \mathbb{R} \))

- The linear model: response is a linear function of the model parameters

\[
y = f(x, w) = b + \sum_{j=1}^{M} w_j \phi_j(x)
\]

- \( w_j \)'s and \( b \) are the model parameters (\( b \) is an offset)
  - Parameters define the mapping from the inputs to responses

- Each \( \phi_j \) is called a basis function
  - Allows change of representation of the input \( x \) (often desired)
Linear Regression: In Higher Dimensions

The linear model:

\[ y = b + \sum_{j=1}^{M} w_j \phi_j(x) = b + w^T \phi(x) \]

- \( \phi = [\phi_1, \ldots, \phi_M] \)
- \( w = [w_1, \ldots, w_M] \), the weight vector (to learn using the training data)
The linear model:

\[ y = b + \sum_{j=1}^{M} w_j \phi_j(x) = b + w^T \phi(x) \]

- \( \phi = [\phi_1, \ldots, \phi_M] \)
- \( w = [w_1, \ldots, w_M] \), the weight vector (to learn using the training data)
- We consider the simplest case: \( \phi(x) = x \)
  - \( \phi_j(x) \) is the \( j \)-th feature of the data (total \( D \) features, so \( M = D \))
The linear model:

\[ y = b + \sum_{j=1}^{M} w_j \phi_j(x) = b + w^T \phi(x) \]

- \[ \phi = [\phi_1, \ldots, \phi_M] \]
- \[ w = [w_1, \ldots, w_M] \], the weight vector (to learn using the training data)
- We consider the simplest case: \( \phi(x) = x \)
  - \( \phi_j(x) \) is the \( j \)-th feature of the data (total \( D \) features, so \( M = D \))
- The linear model becomes

\[
 y = b + \sum_{j=1}^{D} w_j x_j = b + w^T x
\]
The linear model:

\[ y = b + \sum_{j=1}^{M} w_j \phi_j(x) = b + w^T \phi(x) \]

- \( \phi = [\phi_1, \ldots, \phi_M] \)
- \( w = [w_1, \ldots, w_M] \), the weight vector (to learn using the training data)
- We consider the simplest case: \( \phi(x) = x \)
  - \( \phi_j(x) \) is the \( j \)-th feature of the data (total \( D \) features, so \( M = D \))
- The linear model becomes

\[ y = b + \sum_{j=1}^{D} w_j x_j = b + w^T x \]

Note: Nonlinear relationships between \( x \) and \( y \) can be modeled using suitably chosen \( \phi_j \)'s (more when we cover Kernel Methods)
Given training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$

Fit each training example $(x_i, y_i)$ using the linear model

$$y_i = b + w^T x_i$$
Given training data \( \mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \)

Fit each training example \((x_i, y_i)\) using the linear model

\[
y_i = b + \mathbf{w}^T \mathbf{x}_i
\]

A bit of notation abuse: write \(\mathbf{w} = [b, \mathbf{w}]\), write \(\mathbf{x}_i = [1, \mathbf{x}_i]\)

\[
y_i = \mathbf{w}^T \mathbf{x}_i
\]