$X$ X_train: $m \times n$

$$\begin{matrix} & n \\ m & \begin{bmatrix} & \\ & \end{bmatrix} \end{matrix}$$

→ row = a training data point

There're $m$ training data points

↑ column = a feature

$\vec{w}$ weights: $n \times 1$ a parameter for each feature

$\vec{y}$ y_train: $m \times 1$ a output / label for each training data pt.

$\vec{e}$ error: $m \times 1$ same size as y_train

$\hat{\vec{y}}$ output = $X\vec{w} + \vec{b}$ $\vec{b}$: $m \times 1$ bias for each prediction

$$\vec{e} = \frac{1}{m}(\hat{\vec{y}} - \vec{y})^2$$ mean square error, vector square is $y^T y$

$$= \frac{1}{m}(\hat{\vec{y}} - \vec{y})^T(\hat{\vec{y}} - \vec{y})$$ $\hat{\vec{y}}^T\vec{y} = \vec{y}^T\hat{\vec{y}}$ b/c outcome is scalar

$$= \frac{1}{m}(\hat{\vec{y}}^T\hat{\vec{y}} - 2\vec{y}^T\hat{\vec{y}} + \vec{y}^T\vec{y})$$ $\hat{\vec{y}} = X\vec{w} + \vec{b}$

$$= \frac{1}{m}(\underbrace{\vec{w}^T X^T X \vec{w}}_{\downarrow\ 2X^TX\vec{w}} + \underbrace{\vec{w}^T X^T \vec{b}}_{\downarrow\ X^T\vec{b}} + \underbrace{\vec{b}^T X \vec{w}}_{\downarrow\ X^T\vec{b}} + \underbrace{\vec{b}^T\vec{b}}_{\downarrow\ 0} - \underbrace{2\vec{y}^T X\vec{w}}_{\downarrow\ -2X^T\vec{y}} - \underbrace{2\vec{y}^T\vec{b}}_{\downarrow\ 0})$$

$$\nabla_{\vec{w}}\vec{e} = \frac{1}{m}(2X^T(X\vec{w} + \vec{b} - \vec{y}))$$

$$= \frac{2}{m}X^T\vec{e} \quad (n \times 1)$$

$\nabla_{\vec{w}}(\vec{x}^T\vec{w}) = x$

$\nabla_{\vec{w}}(\vec{w}^T x) = x$

$\nabla_{\vec{w}}(\vec{w}^T A\vec{w}) = (A + A^T)\vec{w}$

$$\nabla_{\vec{b}}\vec{e} = \frac{1}{m}(X\vec{w} + X\vec{w} + 2\vec{b}) = \frac{2}{m}(X\vec{w} + b)$$

$\hat{\vec{y}} = X'\vec{w}$ $\qquad \vec{e} = X'\vec{w} - \vec{y}$

$\vec{e} = \frac{1}{m}\left(\vec{w}^T X' X'^T \vec{w} - 2\vec{y}^T X'\vec{w}\right)$

$\nabla_{\vec{w}} \vec{e} = \frac{1}{m}\left(2X'^T X'\vec{w} - 2X'^T\vec{y}\right)$

$= \frac{2}{m}\left(2X'^T\left(X'\vec{w} - \vec{y}\right)\right)$

$= \frac{2}{m} X'^T \vec{e}$ $\qquad n+1 \times 1$

$X' = m \times (n+1)$ last col $\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$

$\vec{w} = (n+1) \times 1$

the $\vec{b}$ is just $\vec{w}$

$\left[ X \mid \vec{b} \right] \left[ \vec{w} \right]$

Takeaway:

- weights correspond to the dimensions in training data. bias is the same across all training data. it's the y-intercept
- stack a column of 1s at the end of matrix X to easily calculate $X\vec{w} + \vec{b}$ with $X\vec{w}$

$n$ $\left[ \vec{X} \mid \begin{matrix} 1 \\ \vdots \\ 1 \end{matrix} \right]$ $n \times 1$

$\vec{X}$: each dimension / feature. → each training data

$\begin{matrix} 1 \\ \vdots \\ 1 \end{matrix}$: bias is the same across all training data.

- $\vec{w}$ has length = $n+1$. $\nabla\vec{w}$ finds gradient for each $w_i$ and gradient for $b$. $w_i$ stands for slope for each dimension, $b$ is y intercept
- vector calculus:

$\nabla_{\vec{w}} (\vec{X}^T \vec{w}) = \vec{X}$

$\nabla_{\vec{w}} (\vec{w}^T \vec{X}) = \vec{X}$

} b/c $\vec{X}^T\vec{w}$ is a scalar, $\vec{X}^T\vec{w} = \vec{w}^T\vec{X}$

$\nabla_{\vec{w}} (\vec{w}^T A \vec{w}) = (A + A^T)\vec{w}$

- Use mean square error for loss, b/c it gives us a convex function that we can optimize over by gradient descent

- Math for Linear Regression w Gradient Descent

$X_{train}$ = matrix of training data $m \times n$

$X$ = $X_{train}$ stack w/ a column of 1s $m \times (n+1)$

$\vec{w}$ = length $n \times 1$

$\hat{\vec{y}} = X\vec{w}$ $\qquad \vec{e} = \vec{y} - \hat{\vec{y}}$ error = actual - predicted

$$\ell = \frac{1}{m}(\vec{y} - \hat{\vec{y}})^2 = \frac{1}{m}(\vec{y} - X\vec{w})^2 = \frac{1}{m}(\vec{y} - X\vec{w})^T(\vec{y} - X\vec{w})$$

$$= \frac{1}{m}\left(\vec{w}^T X^T X\vec{w} - 2\vec{w}^T X^T \vec{y} + \vec{y}^T\vec{y}\right)$$

$$\nabla_{\vec{w}}\ell = \frac{1}{m}\left[(X^TX + XX^T)\vec{w} - 2X^T\vec{y} + 0\right]$$

$$= \frac{1}{m}(2X^TX\vec{w} - 2X^T\vec{y})$$

$$= \frac{2}{m}X^T(X\vec{w} - \vec{y}) = -\frac{2}{m}X^T\vec{e}$$

$\nabla_{\vec{w}}\ell$ is used for gradient descent! step opposite of gradient direction

- Recall Linear Regression Normal form Equation

$X = \left[\boxed{\text{data}} \;\middle|\; \begin{matrix}1\\ \vdots \\ 1\end{matrix}\right]$ $\qquad \vec{y}$ = labels

- $\vec{w} = (X^TX)^TX^T\vec{y}$

High Level Notes:

- Regression Problems: $y$ are continuous values → Lin Reg

  Classification problems: $y$ are binary → Logic Regression

- Lasso & Ridge Regularization are applied to linear Regression so that the weights are minimized.

  Regularization prevents overfitting of the training data

## Lasso vs. Ridge Regularization

- Penalty is added to the loss function

  ($L_1$) Lasso: $\alpha \sum_i |w_i|$ → sparse weights, mostly zeros

  $$\nabla_{\vec{w}} \vec{\ell} = -\frac{2}{m} X^T \vec{e} + 2\alpha * \text{sign}(\vec{w})$$

  ($L_2$) Ridge: $\alpha \sum_i w_i^2$ → small but distributed weights

  $$\nabla_{\vec{w}} \vec{\ell} = -\frac{2}{m} X^T \vec{e} + 2\alpha \vec{w}$$ → gradient w/ ridge