CLASSIFICATION PROBLEM

Applying Linear Regression to classification problems is not a good idea. Consider a data set of patients having cancer (malignant or benign). Let's say after applying Linear Regression we got line l_1. If a straight vertical line v_1 is drawn crossing the threshold 0.5 then we can say points to the left of v_1 are benign and points to the right are malignant.

Suppose a new data d_{new} is added to the training set and because of that l_1 has changed and v_1 also has changed. The above inference that points to the left of v_1 are benign and points to the right are malignant is not correct anymore. So Linear Regression for classification problems is not a good fit. In classification problems, $y = 1$ or $y = 0$ but in Linear Regression $y \geq 1$ or $y \leq 0$.

LOGISTIC REGRESSION

It's an algorithm used to solve classification problems. In classification problems, we want $0 \leq h(x) \leq 1$, so we define logistic function or sigmoid function as $g(z) = \frac{1}{1+e^{-z}}$.

> $h(x) = g(\theta^T X) = \frac{1}{1+e^{-\theta^T X}}$

Say for $x = [x_0,\ x_1] = [1,\ 10]$, hypothesis function $h(x) = 0.7$. That means probability that $y = 1$ for given x is 0.7 or there's a 70 chance that $y = 1$. Hypothesis function gives the probability of $y = 1$ for given x, parameterized by θ.
$h(x) = P(y = 1|x; \theta)$
$P(y = 1) + P(y = 0) = 1$
$\begin{bmatrix} y = 1 \ \ when\ h(x) \geq 0.5 \\ y = 0 \ \ when\ h(x) < 0.5 \end{bmatrix}$ And $\begin{bmatrix} g(z) \geq 0.5 \ \ when\ z \geq 0 \\ g(z) < 0.5 \ \ when\ z < 0 \end{bmatrix}$

> So, $\begin{bmatrix} h(x) = g(\theta^T X) \geq 0.5 \ \ when\ \theta^T X \geq 0 \ \ then\ y = 1 \\ h(x) = g(\theta^T X) < 0.5 \ \ when\ \theta^T X < 0 \ \ then\ y = 0 \end{bmatrix}$

DECISION BOUNDARY

Let $h(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
$\theta_0 = -3, \theta_1 = 1$ and $\theta_2 = 1$
So, $\theta^T X = -3 + x_1 + x_2$
$y = 1$ if $h(x) \geq 0.5$ or $\theta^T X \geq 0$
$-3 + x_1 + x_2 \geq 0$ Or $x_1 + x_2 \geq 3$
When we plot this straight line on training set we get a decision boundary. It's the property of hypothesis function not the property of data set.

For Linear Regression $J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h(x) - y)^2$ where $h(x) = \theta^T X$. For Logistic Regression $h(x) = \frac{1}{1+e^{-\theta^T X}}$ the above cost function gives a non-convex function, so gradient descent is not guaranteed to converge to global minimum. So a different cost function is used to get convex plot for Logistic Regression.

$$cost(h(x), y) = \begin{bmatrix} -log(h(x)) \ \ if \ y = 1 \\ -log(1 - h(x)) \ \ if \ y = 0 \end{bmatrix}$$

Above expression can be expressed as, $cost(h(x), y) = -ylog(h(x)) - (1 - y)log(1 - h(x))$

> $J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [ylog(h(x)) + (1 - y)log(1 - h(x))]$

To minimize $J(\theta)$ simultaneously update θ as below (Partial derivatives of cost function)

> $\theta_j = \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} [h(x^i) - y^i] x_j^i$ where $j = 0, 1, 2 ... n$

MULTI-CLASS CLASSIFICATION

In multi-class classification y can take values $0, 1, 2, 3$ etc depending on number of classes. For example, weather: sunny $y = 1$, cold $y = 2$, snow $y = 3$. We define three classifiers $h(x) = P(y = i|x; \theta)$ where $i = 1, 2, 3$.

$h(x)$ will run for all three classifiers then the classifier with highest $h(x)$ will be the right prediction. For example, $h(x) = [0.7, 0.4, 0.2]$. Maximum $h(x) = 0.7$ for $i = 1$. So prediction is $y = 1$.

OVER-FITTING PROBLEM

- Underfit/High bias: Failure to learn the relationship in training data, assumptions about the model lead to ignore training data.
- Overfit/High variance: Too much reliance on training data, model changes significantly based on training data.

When there are too many features, hypothesis may fit the training set very well. Cost function may be very close to zero or may be even zero exactly. But then it fails to generalize to new examples or fails to predict on new examples. This is overfitting. This can be avoided by reducing number of features or by using Regularization.

Higher order polynomial expressions like $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ give overfitting problems. If we make θ_3 and θ_4 very small then fourth degree polynomial expression becomes, $\theta_0 + \theta_1 x + \theta_2 x^2 + very\ small\ value + very\ small\ value$

When there are many fetaures say 100 then we don't know which parameters to pick and try to shrink. So in Regularization we modify the cost function to shrink all the parameters.

> $J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h(x) - y)^2 + \lambda \sum_{j=1}^{n} \theta_j^2$ where λ is regularization parameter added to inflate the cost of θ.

Say $h(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$. By using the above regularized cost function, we can reduce θ_3 and θ_4 close to zero. Then we get a quadratic plot, good fit. If λ is too large then we end up penalizing $\theta_1, \theta_2, \theta_3, \theta_4$ very highly. So we end up getting rid of $\theta_1, \theta_2, \theta_3, \theta_4$. Then $h(x) = \theta_0$. This is the example of underfit.

	Without Regularization	**With Regularization**
Cost Function	$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [ylog(h(x) + (1-y)log(1-h(x))]$	$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [ylog(h(x) + (1-y)log(1-h(x))] + \frac{\lambda}{2m} \sum_{i=1}^{n} \theta_j^2$
Gradient Descent	$\theta_j = \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} [h(x^i) - y^i] x_j^i$	$\theta_j = \theta_j - \alpha \sum_{i=1}^{m} [h(x^i) - y^i] x_0^i$ where $j = 0$ $\theta_j = \theta_j - \alpha [\frac{1}{m} \sum_{i=1}^{m} [h(x^i) - y^i] x_j^i + \frac{\lambda}{m} \theta_j]$ where $j = 1, 2 ... n$