Pitfalls of Linear Regression Model in Classification

- Linear regression model outputs continuous values, and consequently extra steps (based on unknown decision rules) are needed to convert these values to the desired discrete labels.

- The predicted values are not probabilistic not letting us make meaningful interpretations.

Previous Classification Models

- **Perceptron**
  - Generates a random separating hyperplane
  - Intended for linearly separable data
- **Support Vector Machine (SVM)**
  - Finds a separating hyperplane that has maximum margin to the data points
  - Fails in the presence of nonlinear decision boundaries

Logistic Regression Theory

- Binary logistic regression:
  - The categorical response has only two possible outcomes
  - This model predicts the conditional probability that the binary response Y takes value 1 given the features $x \in \mathbb{R}^d$:

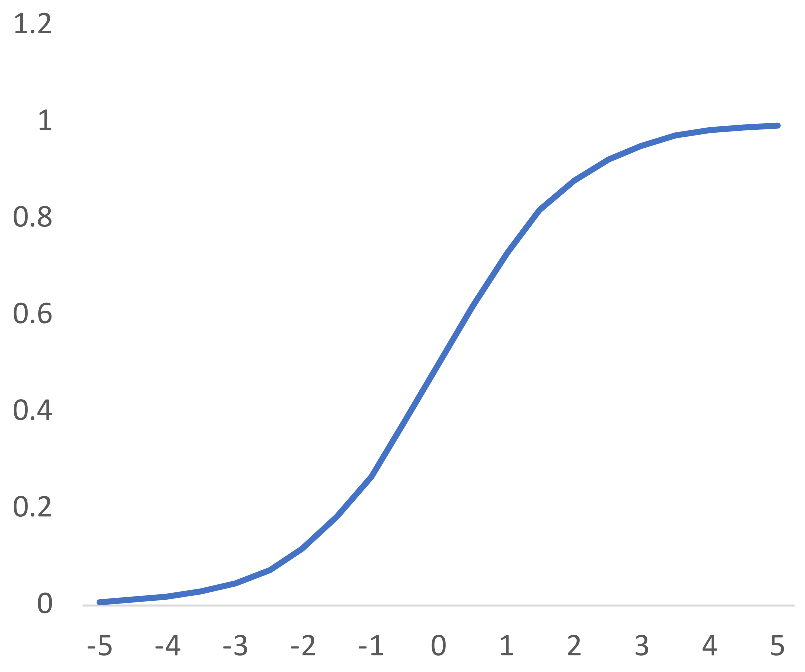
$$P(Y = 1|x) = f_\theta(x) = \frac{1}{1 + \exp(-\theta^T x)}$$

  - The goal is to learn the weight vector $\theta \in \mathbb{R}^d$

Logistic Activation Function

- Note that $f_\theta(x) = s(\theta^T x)$, where s is the sigmoid function:

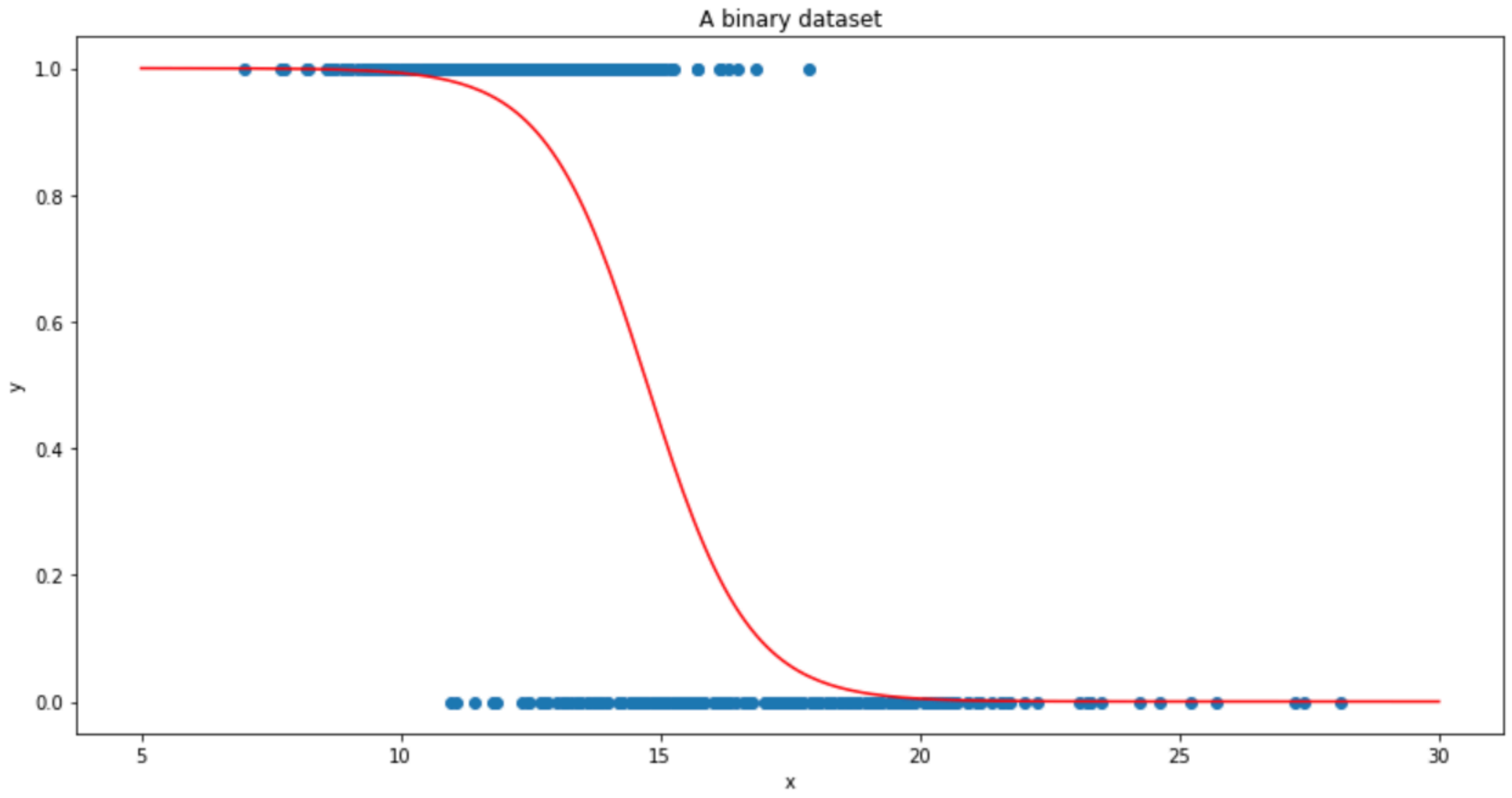
$$s(t) = \frac{1}{1+\exp(-t)}$$



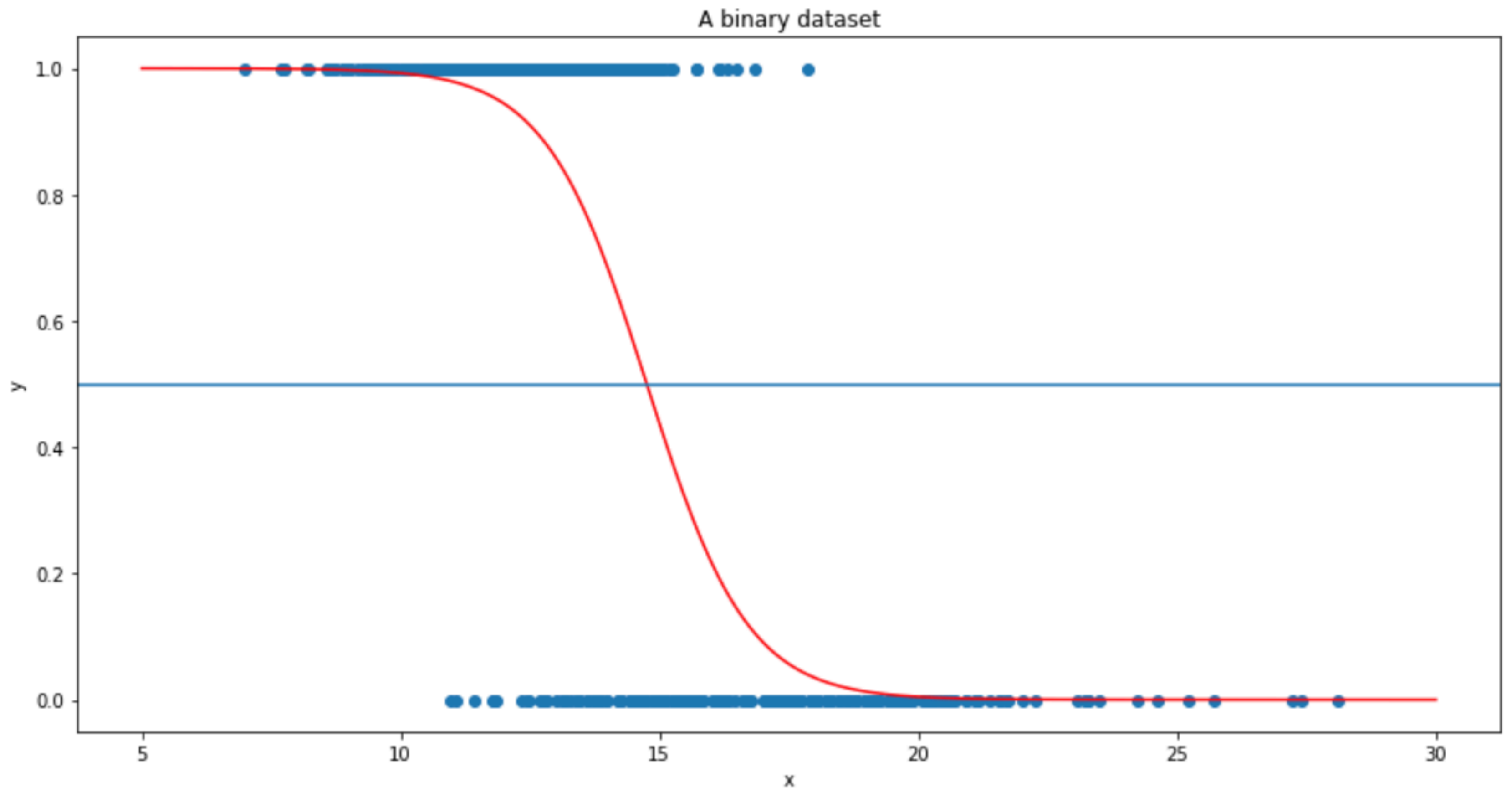
- Some of the properties of sigmoid:
  - Domain: $-\infty < t < \infty$
  - Range: $0 < s(t) < 1$
  - Reflection and symmetry: $s(-t) = 1 - s(t)$
  - Derivative: $s'(t) = s(t)\big(1 - s(t)\big)$

Decision Rule

- Assuming the learned weight vector is $\hat{\theta}$, for a given data point $x \in \mathbb{R}^d$:
  - The model predicts it belongs to class 1 if $s(\hat{\theta}^T x) \geq \frac{1}{2}$.
  - The model predicts it belongs to class 0 if $s(\hat{\theta}^T x) < \frac{1}{2}$.
- Above, we are setting the threshold to be $\frac{1}{2}$. This could be considered as a hyperparameter to be tuned.



- Here’s an example of a logistic regression model fit to a binary class dataset. You can see the sigmoid shape trying to fit to the blue data points.



- The horizontal blue line at y = 0.5 is the probability threshold, and it intersects with x at around x = 15. So any point left of x = 15 would be classified as 1, while the rest would be classified as 0.

Loss Function

- For logistic regression, we use **cross entropy** loss. Given training points $(x_i, y_i),\ i = 1, \ldots, n$, cross entropy loss is defined as

$$L(\theta, X, y) = -\sum_{i=1}^{n} y_i \log\big(f_\theta(x_i)\big) + (1 - y_i)\log\big(1 - f_\theta(x_i)\big)$$

- It is derived by taking the logarithm of likelihood $f_\theta(x_i)^{y_i}\big(1 - f_\theta(x_i)\big)^{1-y_i}$, which we want to be maximized. Because of this, the loss function above is also called the **negative log-likelihood function**.

- Note that

$$\nabla_\theta L(\theta, X, y) = -\frac{1}{n}\sum_{i=1}^{n}\big(y_i - f_\theta(x_i)\big)\ x_i$$

Training a Logistic Regression Model (1)

- In order to minimize the cross-entropy loss $L(\theta, X, y)$, we use some variants of the Gradient Descent algorithm:

  - **Batch Gradient Descent:** We use the gradient computed over all the training data points.

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_\theta L\left(\theta^{(t)}, X, y\right)$$

Batch Gradient Descent is computationally expensive at each iteration.

  - **Stochastic Gradient Descent:** We use the gradient of the loss due to one single random training data point.

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_\theta l\left(\theta^{(t)}, x_i, y_i\right),$$

where $l(\theta, x_i, y_i) = -y_i \log\left(f_\theta(x_i)\right) - (1 - y_i) \log\left(1 - f_\theta(x_i)\right)$.

Stochastic Gradient Descent may take more iterations to converge compared to the Batch Gradient Descent.

Training a Logistic Regression Model (2)

- **Mini-Batch Gradient Descent:** We use the gradient of loss due to a random subset of the training data points with fixed size.

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_\theta L^B\left(\theta^{(t)}, X, y\right),$$

where $L^B(\theta, X, y) = \frac{1}{|B|} \sum_{i \in B} l(\theta, x_i, y_i)$.

Mini-batch gradient descent compromise between time complexity and accuracy of batch gradient descent and stochastic gradient descent.

Evaluation Metrics of Classification Models (1)

- **Accuracy**

  The accuracy of the classifier is defined as the ratio of its correct predictions to the number of test point.

- **Confusion Matrix**

  The confusion matrix compares what the model predicts with the actual counts in each class.

Evaluation Metrics of Classification Models (2)

- **Precision and Recall**
  - **Precision:**

$$Precision = \frac{True\ Positives}{True\ Positive\ +\ False\ Positive} = \frac{True\ Positives}{Predicted\ True}$$

  - **Recall:**

$$Recall = \frac{True\ Positives}{True\ Positive + False\ Negative} = \frac{True\ Positive}{Actually\ Positive}$$

Evaluation Metrics of Classification Models (3)

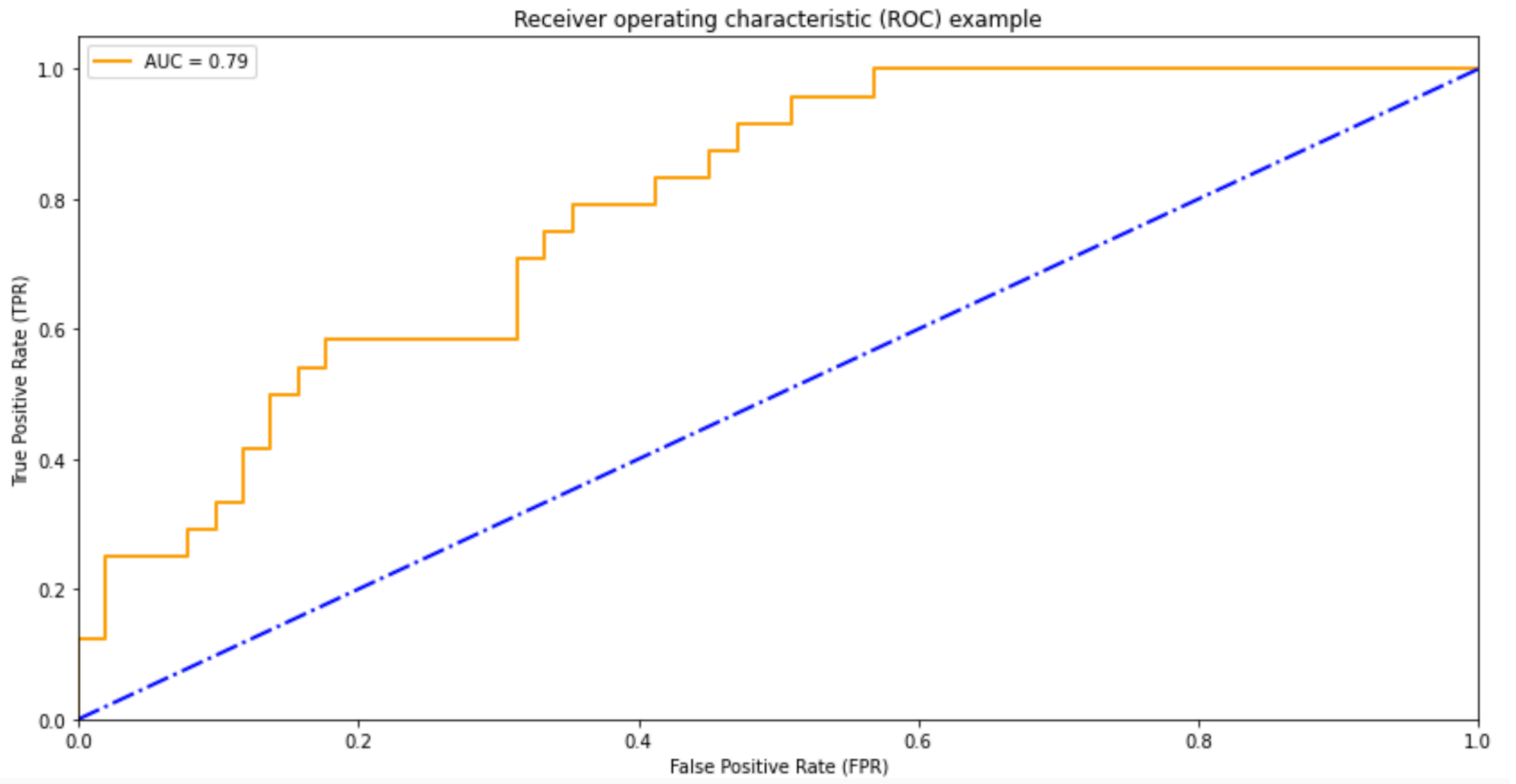
- **ROC Curve and AUC**
  - **TPR:**

$$TPR = \frac{True\ Positives}{True\ Positives\ + False\ Negatives} = \frac{True\ Positives}{Actually\ Positive}$$

  - **FNR:**

$$FNR = \frac{True\ Negatives}{True\ Negatives\ +\ False\ Positives} = \frac{True\ Negatives}{Actually\ Negative}$$

Compute TPR and FNR at different thresholds ranging from 0 to 1. Then, ROC curve is obtained by plotting TPR vs FNR. The higher the area under ROC curve (AUC), the better the model performs.



- Here’s a ROC Curve example. The closer to the top right corner the curve is (and therefore the closer to 1 the AUC is) the more robust the model

From Binary to Multiclass Classification

- When we have K classes, we consider K different weight vectors $\theta_1, \dots, \theta_K$ corresponding to each class.

- The probability that the data point x_i belongs to the class j is defined to be

$$\frac{\exp\left(\theta_j^T x_i\right)}{\exp(\theta_1^T x_i)+\exp(\theta_2^T x_i)+\cdots+\exp(\theta_K^T x_i)}.$$

Above, we have used the **softmax function**, which is a generalization of the logistic function.

- For predication, we say that the data point x belongs to class $i \in \{1, \dots, K\}$, if the probability of x belonging to that class is the highest.

Bonus: Probabilistic Interpretation (1)

- **KL Divergence:** Given probability distribution P and Q defined on the same sample space, the KL divergence from Q to P is given by

$$D_{KL}(P||Q) = \sum_x P(x) \ln \frac{P(x)}{Q(x)}.$$

- In binary classification, KL divergence quantifies the difference between the distribution $P_{\hat{\theta}}$ computed by the logistic model and the actual distribution P on the dataset.

Bonus: Probabilistic Interpretation (2)

- For each data point (x, y), KL divergence between P and P_θ is given by

$$D_{KL}(P(y)||P_\theta(\hat{y})) \; = \; P(y=1|x)\ln\frac{P(y=1|x)}{P_\theta(\hat{y}=1|x)} + \big(1-P(y=1|x)\big)\ln\frac{1-P(y=1|x)}{1-P_\theta(\hat{y}=1|x)}.$$

- In binary classification, the goal is to find $\hat{\theta}$ in logistic model that minimizes sum of KL divergence of all (x_i, y_i) in the dataset, which is given by

$$\sum_{i=1}^{n} D_{KL}\left(P(y_i)||P_\theta(\widehat{y_i})\right)$$

Bonus: Probabilistic Interpretation (3)

- Set $y_i = P(y_i = 1 | x_i)$ and $\hat{y_i} = P_\theta\left(\widehat{y_i} = 1 \middle| x_i\right)$.
- Removing the terms independent of θ, the objective function becomes

$$\sum_{i=1}^{n} -y_i \ln(\hat{y}_i) - (1 - y_i) \ln(1 - \hat{y}_i) \,,$$

which is exactly the sum of cross-entropy loss between observed empirical distribution P and predicted probability distribution P_θ given by logistic regression model.