Classifiying Categorical Data

Week 11

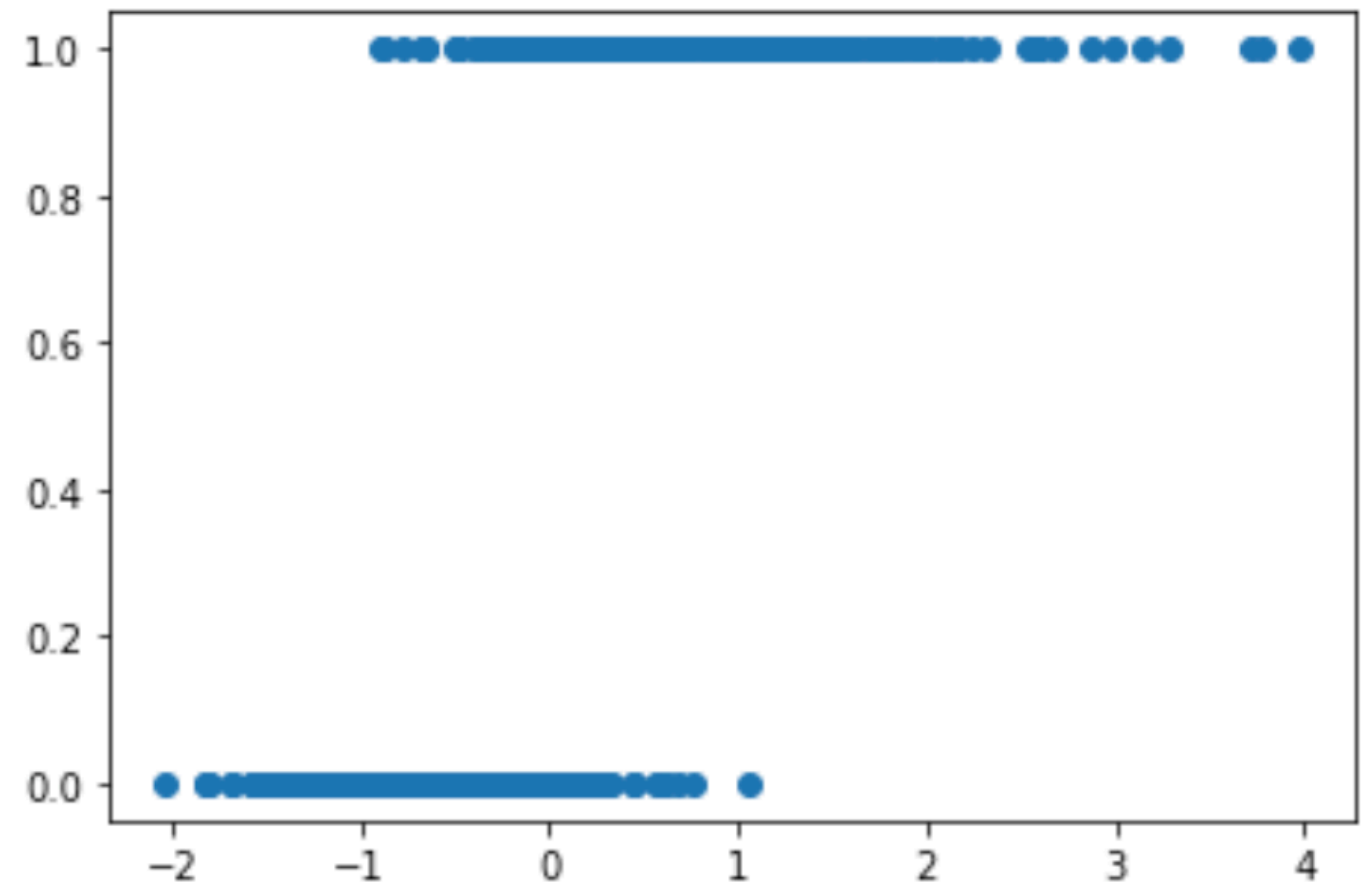
Natalie Agus

Learning Objectives

- Write **cost** function of logistic regression
- Use logistic regression to **calculate** probabilities of binary classification
- **Train** logistic regression model
- **Split** data into training, validation, and testing set
- Calculate **confusion** matrix, **precision**, and **recall**.

Logistic Regression
Hypothesis Function

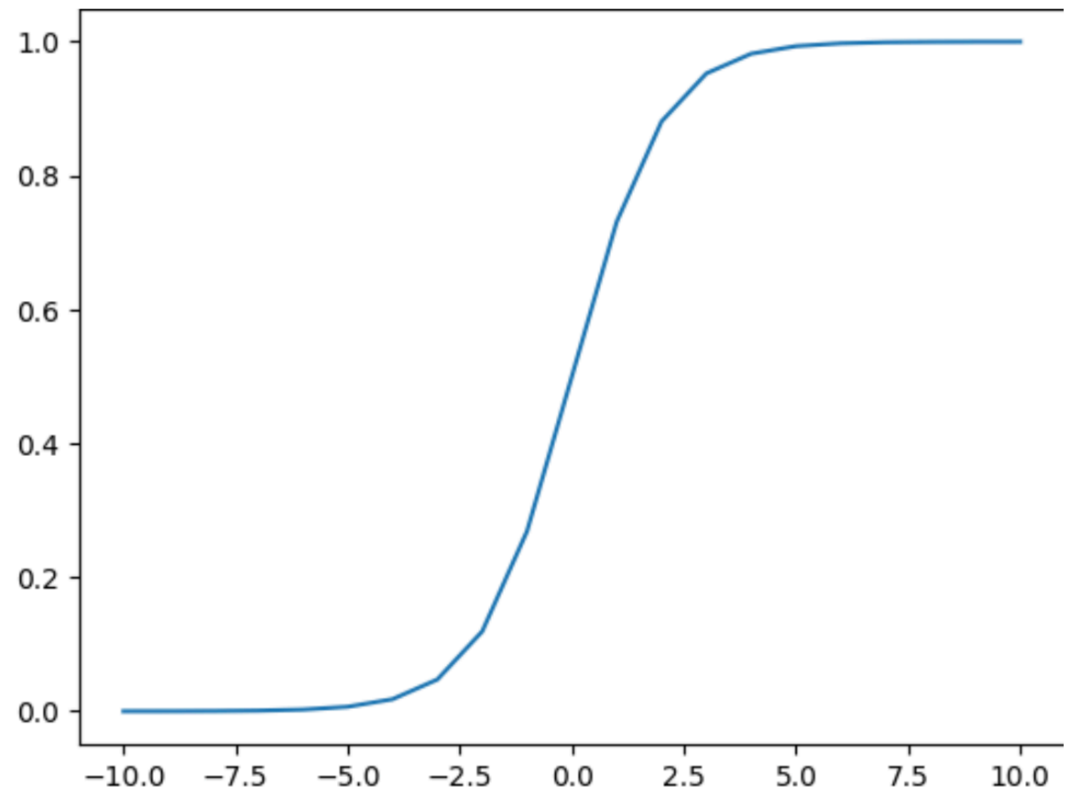
- Model and predict the probability of a **binary** outcome based on one or more independent variables
- Need a function that can map **continuous** data into **discrete** values (like 0 or 1)



Logistic Function

$$y = \frac{1}{1 + e^{-z}}$$

- Logistic function y plot
- We use y as p(x)
- Parameters present in z(x)
- z(x): linear regression
- We **map the value of linreg** to p(x)
  - p(x): estimated probability that y=1 on input x



$$p(x) = \frac{1}{1 + e^{-z(x)}}$$

$$z(x) = \beta_0 x_0 + \beta_1 x_1 + \ldots + \beta_n x_n$$

What does the parameters represent?

- Definition: **odds**

$$\text{Odds} = \frac{P(Y=1)}{P(Y=0)}$$

- Take log of both sides:

$$\text{Log-Odds} = \log\left(\frac{P(Y=1)}{P(Y=0)}\right)$$

- Then: $\text{Odds} = \frac{P(Y=1)}{P(Y=0)} = e^{\text{Log-Odds}}$

- Getting P(Y=1):

$$P(Y=1) = \frac{1}{1+e^{-\text{Log-Odds}}}$$

- We have:

$$p(x) = \frac{1}{1+e^{-z(x)}}$$

- Hence, z(x) is log-odds

$$\text{Log-Odds} = \log\left(\frac{P(Y=1)}{P(Y=0)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

What do the parameters represent?

- log-odds range between $-\infty$ to $+\infty$

$$\text{Log-Odds} = \log\left(\frac{P(Y=1)}{P(Y=0)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

- Plugging log-odds into logistic function gives us exactly P(Y=1), and maps it back to 0 to 1:

$$p(x) = \frac{1}{1 + e^{-z(x)}}$$

$$P(Y=1) = \frac{1}{1 + e^{-\text{Log-Odds}}}$$

What do the parameters represent?

- **log-odds**: is a dependent variable made up of linear combination of the model parameters) before applying the logistic function to convert it into probability.

$$\text{Log-Odds} = \log\left(\frac{P(Y=1)}{P(Y=0)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

- **Each parameter** (βj) represent the **change** in log-odds associated with *one unit increase* in the predictor Xj

- Since log is a positive function, a positive change in log-odds increases the odds and vice versa.

- Hence change in log-odds translates to **multiplicative** change in odds (with a factor)

$$\text{Odds}_{\text{new}} = \text{Odds}_{\text{old}} \cdot e^{\Delta}$$

Cost Function

- **Binary cross-entropy loss or log-loss**
- Can't use regular MSE
- We start by computing **likelihood**
- Then taking the logarithm of it (positive increasing function, no effect on maximisation)
  - We want to maximise likelihood
  - **Equivalent** to minimise negative log-likelihood
- Depending on value of y_i (0 or 1), only one of the term will bear a 'cost'

$$L(\beta) = \prod_{i=1}^{n} P(y_i | x_i; \beta)$$

$$\log L(\beta) = \sum_{i=1}^{n} [y_i \log(\hat{p}_i) + (1 - y_i) \log(1 - \hat{p}_i)]$$

$$\hat{p}_i = P(Y = 1 | x_i; \beta)$$

$$\mathrm{Cost}(\beta) = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log(\hat{p}_i) + (1 - y_i) \log(1 - \hat{p}_i)]$$

Gradient Descent

- **Minimise** cost function: $\frac{\partial}{\partial \beta_j} J(\beta) = \frac{1}{m} \Sigma_{i=1}^{m} \left(p(x) - y^i \right) x_j^i$
- **Update** function (matrix version):

$$\beta_j = \beta_j - \alpha \frac{1}{m} \Sigma_{i=1}^{m} \left(p(x) - y^i \right) x_j^i$$

$$\mathbf{p}(x) = \frac{1}{1 + e^{-\mathbf{Xb}}}$$

$$\mathbf{b} = \mathbf{b} - \alpha \frac{1}{m} \mathbf{X}^T (\mathbf{p} - \mathbf{y})$$

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^1 & x_2^1 & \dots & x_n^1 \\ 1 & x_1^2 & x_2^2 & \dots & x_n^2 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & x_1^m & x_2^2 & \dots & x_n^m \end{bmatrix}$$

Multi-class

- **N class**: N sets of parameters, train each case
- **One versus all technique**

feature_1	feature_2	cat	dog	fish	predicted class
x	x	**0.8**	0.2	0.3	cat
x	x	**0.9**	0.1	0.2	cat
x	x	0.5	**0.9**	0.4	dog
x	x	0.3	0.2	**0.8**	fish
x	x	0.1	**0.7**	0.5	dog

Metrics: Confusion Matrix

- Confusion matrix obtained from **test set**

actual\predicted	Positive Case	Negative Case
Positive Case	True Positives	False Negatives
Negative Case	False Positives	True Negatives

- **4 metrics: accuracy, precision, sensitivity, specificity**

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total Cases}}$$

$$\text{sensitivity} = \frac{\text{TP}}{\text{Total Actual Positives}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{precision} = \frac{\text{TP}}{\text{Total Predicted Positives}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{specificity} = \frac{\text{TN}}{\text{Total Actual Negatives}} = \frac{\text{TN}}{\text{FP} + \text{TN}}$$

Metrics: Confusion Matrix

- For multiple cases: let Mij --> row i column j

actual\predicted	cat	dog	fish
cat	11	1	2
dog	2	9	3
fish	1	1	8

$$\text{accuracy} = \frac{\sum_i M_{ii}}{\sum_i \sum_j M_{ij}}$$

$$\text{precision}_i = \frac{M_{ii}}{\sum_j M_{ji}}$$

$$\text{sensitivity}_i = \frac{M_{ii}}{\sum_j M_{ij}}$$

- Sensitivity: sum over **columns** j
- Precision: uses Mji, which sum over **rows** j

Learning Objectives

- Write **cost** function of logistic regression
- Use logistic regression to **calculate** probabilities of binary classification
- **Train** logistic regression model
- **Split** data into training, validation, and testing set
- Calculate **confusion** matrix, **precision**, and **recall**.