Logistic Regression

Oliver Gregory

Odds and Log Odds

Odds and probabilities are related but not the same. The odds of an event, Y, are the ratio of the event happening to the event not happening. For example, if the odds of Edinburgh Rugby between the Glasgow Warriors are 5 to 3, then that means if it were to be played 8 times, Edinburgh would win 5 of those times and the odds would be $5/3 \approx 1.7$. Probability on the other hand, is the ratio of the event happening to everything, i.e., the number of games Edinburgh Rugby win, to the number of total games they play, which is $5/8 = 0.625$.

Probabilities are always between 0 and 1, while odds are always within the range of 0 and infinity.

Calculating the odds from probabilities

The odds of an event happening can be calculated from the ratio of the probabilites of each event.

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

Back to the Edinburgh Rugby example, where $Y = 1$ is the event where Edinburgh Rugby beats the Glasgow Warriors,

$$\begin{aligned} P(Y=1) &= \frac{5}{8} = 0.625 \\ P(Y=0) &= 1 - P(Y=1) = \frac{3}{8} = 0.375 \\ \text{Odds of Edinburgh Winning} &= \frac{P(Y=1)}{P(Y=0)} = \frac{5}{8} \Big/ \frac{3}{8} = \frac{5}{3} \approx 1.7 \end{aligned}$$

Log Odds

When the odds are against an event happening, i.e., it is less likely to happen than not happen, the odds will be between 0 and 1. Conversely, when the odds are in favour of an event happening, the odds will be between 1 and infinity. If the odds of an event are against 1 to 6, then the odds are approximately 1.7. But if the odds are in favour 6 to 1, then the odds are 6. The magnitude of the odds against look much smaller than the odds in favour, but taking the log of these odds makes it all symmetrical.

$$\log(1/6) = -\log(6) = -1.79$$

The log function makes the absolute distance from 0 to the odds the same whether it is against or in favour. Log Odds are normally distributed.

Logistic Function (or Sigmoid Function)

The logit function is defined as the log of the ratio of probabilities, i.e., the log odds.

$$z = \log\left(\frac{p}{1-p}\right), \text{ where } p = P(Y=1)$$

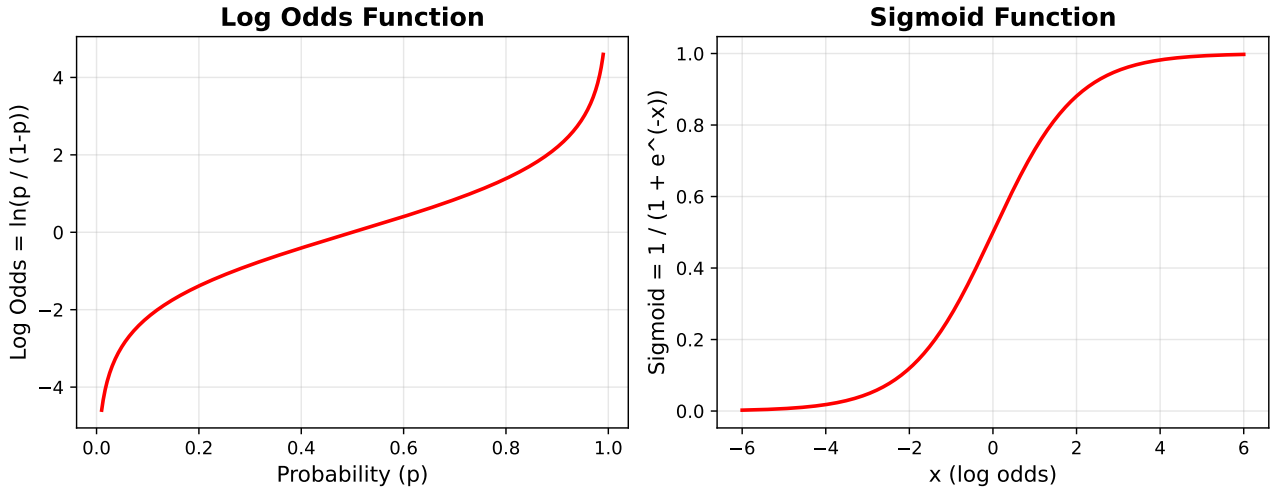
The logistic function, or the sigmoid function, is the inverse of the logit function.

$$
\begin{aligned}
e^{z} &= \frac{p}{1-p} \\
e^{-z} &= \frac{1}{p} - 1 \\
1 + e^{-z} &= \frac{1}{p} \\
\sigma(z) = p &= \frac{1}{1+e^{-z}} \text{ where } z \text{ is the log odds}
\end{aligned}
$$

This is the logistic function:

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

This means that the logistic function backs out the probability of an event happening, from the log odds of the event happening.



Maximum Likelihood Estimation

Logistic regression uses **Maximum Likelihood Estimation** to estimate the parameters that best fit the training data.

For a single observation, the likelihood of observing that observation depends on its class label:

- If $y = 1$: The probability is $P(y = 1|x) = \sigma(z)$.
- If $y = 0$: The probability is $P(y = 0|x) = 1 - \sigma(z)$.

This can be combined into a single expression:

$$P(y|x) = \sigma(z)^{y} \cdot (1 - \sigma(z))^{(1-y)}$$

For a dataset with m observations, the likelihood function, which gives the probability of observing a particular dataset, is the product of the individual observation probabilities for all the observations.

$$L(\beta) = \prod_{i=1}^{m} \sigma(z_i)^{y_i} \cdot (1 - \sigma(z_i))^{(1-y_i)}$$

where $z_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_n x_{in}$ is the linear combination for the i^{th} observation and y_i is the actual observation.

To simplify the maximisation computation, we can take the log of this, making it the log-likelihood function.

$$\log L(\beta) = \sum_{i=1}^{m} [y_i \log(\sigma(z_i)) + (1 - y_i) \log(1 - \sigma(z_i))]$$

The objective of MLE is to maximise the log-likelihood. This is the same as minimising the negative log-likelihood, which is also called the cross-entropy loss. Intuitively, when $y_i = 1$, the loss simplifies to $-\log \sigma(z_i)$, heavily penalising the model if it assigns a low probability to the positive class. When $y_i = 0$, the loss simplifies to $-\log(1 - \sigma(z_i))$, heavily penalising the model if it assigns a high probability to the positive class.

$$\mathcal{L}(\beta) = -\frac{1}{m} \sum_{i=1}^{m} [y_i \log(\sigma(z_i)) + (1 - y_i) \log(1 - \sigma(z_i))]$$

In general, there is no analytical solution to this minimisation, so numerical methods, like gradient descent, must be used to find the optimal parmeters of the model.

Interpreting Coefficients

Without scaling, a one unit increase in a variable x_i leads to a β_i units increase in z, also known as the log odds. For example, a coefficient of 0.5 corresponds to an increase in the log odds of 0.5. However, this is not easily interpretable. Instead, if we take the exponential, then we see that a coefficient of 0.5 corresponds to an increase in the *odds* of an event happening by $\exp(0.5) \approx 1.65$ meaning a one unit increase in the feature, leads to an increase in the odds of the positive class by 65%.

In code

We can use the `sklearn.linear_model` package to build logistic regression models using python.

```python
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LogisticRegression()
model.fit(X_train, y_train)

y_proba = model.predict_proba(X_test)
y_pred = (y_proba[:, 1] >= 0.5).astype(int)
```

Assumptions of logistic regression

1. Linearity: The relationship between the features and the log odds is linear.
2. Independent errors: The errors, or residuals, are independent.
3. No multicollinearity: The independent variables are not highly correlated with each other.
4. Sufficient sample size: Logistic regression requires a sufficiently sized and balanced dataset. A rule of thumb is 10-20 observations per feature.

Metrics for logistic regression

Accuracy

Accuracy measures the proportion of correctly classified predictions, both positive and negative, out of all predictions.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Accuracy is useful when the classes are **balanced** as a high accuracy means the model can effectively differentiate between outcomes; however, in imbalanced datasets, the model can achieve high accuracy by predicting the majority class all of the time.

Recall

Recall measures the proportion of actual positives that the model is able to identify.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall is vital when the cost of missing a true positive is high, such as in medicine where failing to identify a disease can have fatal consequences.

Precision

Precision measures the proportion of positive predictions that were actually correct.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision is critical when the cost of a false positive is high, for example, in spam email classification where important emails may be misclassified as spam.

F1-Score

The F1-Score is the harmonic mean of precision and recall which provides a metric that balances the tradeoff between them.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

It can be particularly useful when precision and recall both matter, i.e. when false positives and false negatives are equally important, such as in search engine results.

ROC Curve and AUC

The ROC (Receiver Operating Characteristic) curve plots the true positive rate (recall) against the false positive rate at different classification thresholds. The AUC (Area Under the Curve) quantifies the overall ability of the model to distinguish between classes. The true positive rate is all the proportion of true positives over all actual positives. The false positive rate is the proportion of false positives over all actual negatives.

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$