Classwork 05: Logistics Regression CPE 490/590 ST

Instructor: Rahul Bhadani

40 points

1 Logistics Regression with One Predictor Variable

1.1 Background

Logistics regression establishes the numerical relationship between two variables (predictor and response). The response is assumed to be binary (either 0 or 1, or -1 and 1).

Note: Logistics Regression is actually a supervised classification problem.

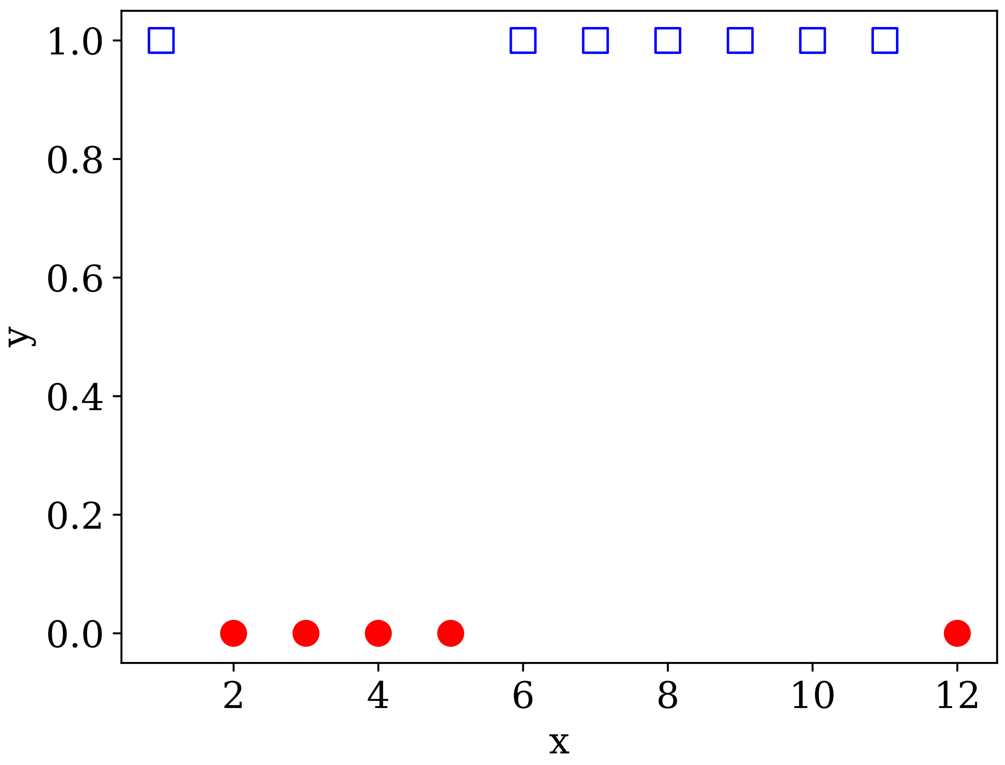
1.2 Assumptions about Logistics Regression

1. **Linearity:** Logistics regression fits a logistics curve to binary data. The curve can be interpreted as the probability associated with each outcome across independent variable values. Logistics regression assumes that the relationship between the natural log of these probabilities (when expressed as odds) and your predictor variable is linear.
2. **No Outliers:** Logistics regression is very sensitive to outliers.
3. **Independence:** Each observation point should be independent of each other.

Consider the dataset given in the table:

x	1	2	3	4	5	6	7	8	9	10	11	12
y	1	0	0	0	0	1	1	1	1	1	1	0

A scatter plot looks like this:



Q1: Please comment on why linear regression is not a suitable choice for making a distinction between red dots and blue squares **(5 Points)**:

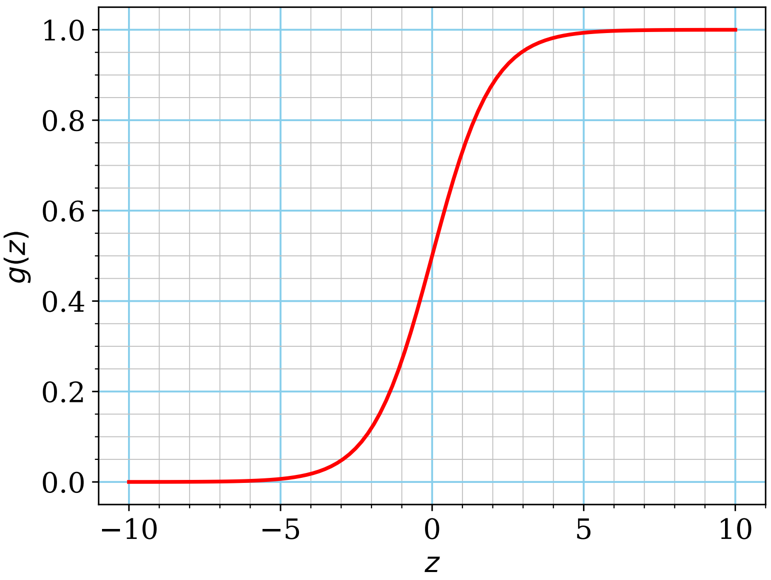
Answer:

Linear Regression will predict continuous value and not 0 or 1.

1.3 Sigmoid Function

By looking at the data above, we think we can classify or correctly label the data points if we have a function that is low (0) for some low values of x, and high (1)for some values of x. One such function is **Sigmoid Function** or **Logistics Function**.

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



From the above, we see that

$$\lim_{z \to -\infty} g(z) = 0$$

and

$$\lim_{z \to \infty} g(z) = 1$$

It means, that for a very high positive value of x, the function approaches 1, and for a very high negative value of x, the function approaches 0.

1.4 Generalized Sigmoid/Logistics Function

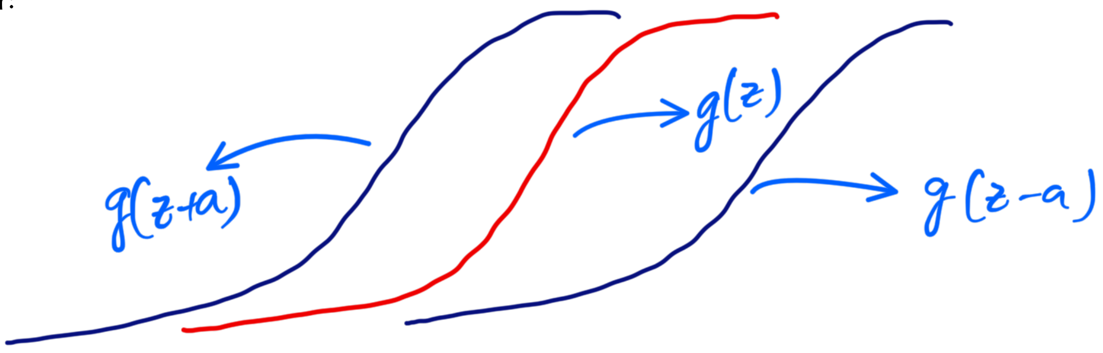
But hey, data x are all positive. So what do we do then?

Can we somehow modify the $g(z)$ function using some transformation techniques we studied in the Signals & Systems class?

What if we have some parameter a so that we get $g(z - a)$?

Q2: Draw $g(z-a)$ for a positive and negative a **(5 Points)**.

Answer:

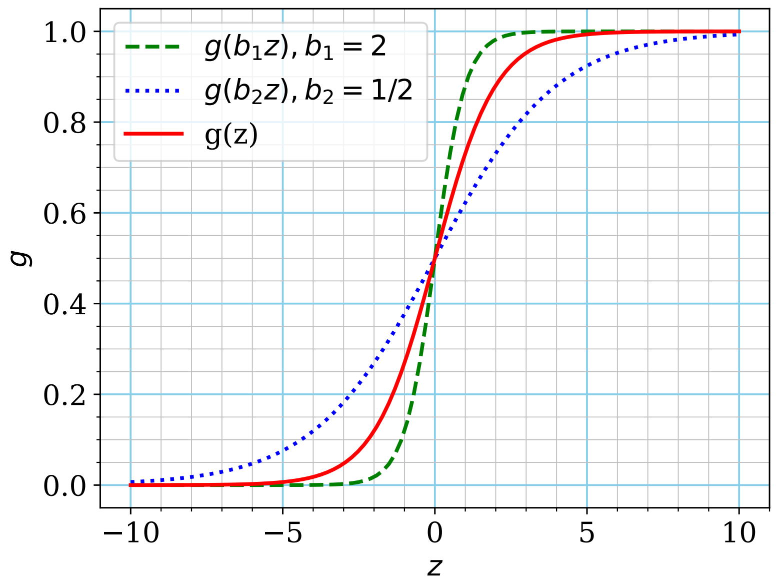


Now, what if we want the slope to change faster or slower? How should we modify the Sigmoid function?

Q3: Write down a modified Sigmoid function for the slope to change faster or slower **(5 Points)**.

Answer:

$$g(z;b) = \frac{1}{1+e^{-zb}}$$



In the above example b_1, b_2 and $\pm a$ are learnable parameters that we are going to learn from data the same way we learned w_1 and w_0 in the linear regression. Thus, we consider a parameter vector $\boldsymbol{\theta}$ that consists of a vector of parameters to be learned using data $\mathbf{x}$ that we can use to write a generalized version of sigmoid that is data-dependent by setting $z = \boldsymbol{\theta}^\top \mathbf{x}$

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g(\boldsymbol{\theta}^\top \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^\top \mathbf{x}}}$$

Here, the overall goal becomes finding $P(Y|X)$, which is given information about the random variable X that gives the distribution of the predictor variable, finding the probability that the random variable Y takes a certain value.

In the above formulation, $h_{\boldsymbol{\theta}}$ estimates the probability that the data points belong $Y = 1$, and hence we can write it as

$$P(Y = 1|X = x) = h_{\boldsymbol{\theta}}(\mathbf{x})$$

(5 Points)

$$P(Y = 0|X = x) = \boxed{1 - h_\theta(\vec{x})}$$

1.5 Cost Function for Logistics Regression

Just like linear regression, we need a cost function based on which we want to estimate $\boldsymbol{\theta}$.

Here, we have two scenarios:

1.5.1 $y_i = 1$

1. if $h_{\boldsymbol{\theta}}(\mathbf{x}) \approx 1$, cost ≈ 0 (no penalty)
2. if $h_{\boldsymbol{\theta}}(\mathbf{x}) \approx 0$, cost $\approx$ very high

1.5.2 $y_i = 0$

1. if $h_{\boldsymbol{\theta}}(\mathbf{x}) \approx 1$, cost $\approx$ very high
2. if $h_{\boldsymbol{\theta}}(\mathbf{x}) \approx 0$, cost ≈ 0 (no penalty)

Then as the outcome is 0 or 1, it follows the Bernoulli Distribution (The Bernoulli distribution is a special case of the binomial distribution where a single trial is conducted (so n would be 1 for such a binomial distribution).

The formula for pmf associated with a Bernoulli random variable over possible outcomes x is given as follows **(5 Points)**:

Answer:

$$f(x;p) = \begin{cases} p & ; \text{ if } x = 1 \\ 1-p & ; \text{ if } x = 0 \end{cases}$$

$$\Rightarrow f(x;p) = p^{x} (1-p)^{1-x}$$

In the case of logistics regression, that probability is given by the logistics function.

Q4: How will we modify the pmf for the case of the logistics function **(5 Points)**?

Answer:

$$P(Y=y|X=x) = h_\theta(\vec{x})^y [1-h_\theta(\vec{x})]^{1-y}$$

Now, we write taking all data into account (all data points are independent):

$$L(\theta) = \Pi_{i=1}^n P(Y=y_i|X=\mathbf{x}_i) = \Pi_{i=1}^n h_{\boldsymbol{\theta}}(\mathbf{x}_i)^{y_i}\left[1-h_{\boldsymbol{\theta}}(\mathbf{x}_i)\right]^{(1-y_i)}$$

This is called **likelihood function**.

Now, take the log of the likelihood function, which is called the log-likelihood function:

$$LL(\boldsymbol{\theta}) = \sum_{i=1}^n y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) + (1-y_i)\log\left[1-h_{\boldsymbol{\theta}}(\mathbf{x}_i)\right]$$

Write down the log of the Likelihood function **(5 Points)**:

$$LL(\boldsymbol{\theta}) = \log(L(\boldsymbol{\theta})) =$$

$$\sum y_i \log(h_\theta(\vec{x}_i)) + (1-y_i)\log[1-h_\theta(\vec{x}_i)]$$

1.6 The Cost Function

In the logistics regression, our goal becomes maximizing the log-likelihood by choosing the appropriate $\boldsymbol{\theta}$. For this, we take the partial derivative of $LL(\theta)$ with respect to $\boldsymbol{\theta}$.

Derive the expression for

$$\frac{\partial LL(\boldsymbol{\theta})}{\partial \theta_j}$$

for $j - th$ component of $\boldsymbol{\theta}$. **(5 Points)**

$$\frac{\partial LL(\boldsymbol{\theta})}{\partial \theta_j} = \sum_{i=1}^{n} \Big[y_i - h_{\boldsymbol{\theta}}(\mathbf{x}_i) \Big] x_{ij}$$

Answer:

$$\frac{\partial LL(\theta)}{d\theta_j} = \frac{\partial}{\partial \theta_j} y \log h_\theta(\vec{x}) + \frac{\partial}{\partial \theta}(1-y)\log\left[1-h_\theta(\vec{x})\right]$$

$$= \frac{y}{h_\theta(\vec{x})} - \frac{1-y}{1-h_\theta(\vec{x})} \cdot \frac{\partial}{\partial \theta_j} h_\theta(\vec{x})$$

$$= \left[\frac{y}{h_\theta(\vec{x})} - \frac{1-y}{1-h_\theta(\vec{x})}\right] h_\theta(\vec{x})\left(1-h_\theta(\vec{x})\right) x_j$$

$$= \left[\frac{y - y \cdot h_\theta(\vec{x}) - h_\theta(\vec{x}) + y\, h_\theta(\vec{x})}{\cancel{h_\theta(\vec{x})}\cancel{\left(1-h_\theta(\vec{x})\right)}}\right] \cdot \cancel{h_\theta(\vec{x})}\cancel{\left(1-h_\theta(\vec{x})\right)} x_j$$

$$= \left(y - h_\theta(\vec{x})\right) x_j$$

If $p = h_\theta(\vec{x}) = g(\theta^T \vec{x})$

Let $z = \theta^T \vec{x}$

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \frac{\partial LL(\theta)}{\partial p} \cdot \frac{\partial p}{\partial \theta_j}$$

$$= \frac{\partial LL(\theta)}{\partial p} \cdot \frac{\partial p}{\partial z} \cdot \frac{\partial z}{\partial \theta_j}$$

$$\frac{\partial h_\theta(\vec{x})}{\partial \theta_j} = h_\theta(\vec{x})(1 - h_\theta(\vec{x}))$$

The derivation comes from the fact that the logistics function has a special property:

$$\frac{\partial g(z)}{\partial z} = g(z)[1 - g(z)]$$

Equating

$$\frac{\partial LL(\boldsymbol{\theta})}{\partial \theta_j} = 0$$

will give optimal θ_j^*. However, the expression doesn't have any closed form, so we resort to using numerical methods for solving the optimization, and hence, in this case, we use **Gradient Ascent Optimization** which is equivalent to minimizing the negative of the log-likelihood function, for which we can use already discussed **Gradient Descent Optimization**.

$$\theta_j \leftarrow \theta_j - \eta \frac{\partial LL(\boldsymbol{\theta})}{\partial \theta_j}$$

$$\theta_j \leftarrow \theta_j - \eta \sum_{i=1}^{n} \Big[y_i - h_{\boldsymbol{\theta}}(\mathbf{x}_i) \Big] x_{ij}$$

1.7 Logistics Regression Using Sklearn

```python
import numpy as np
from sklearn.linear_model import LogisticRegression

# Create an instance of a Logistic Regression Classifier and fit the data.
logreg = LogisticRegression()
logreg.fit(x, y.ravel())

# Predict
y_hat = logreg.predict(x)

print("Predictions:", y_hat)
```

1.8 Assessing the Logistics Regression

1.8.1 Accuracy

Note that the example in this notebook is a very rudimentary example without the data being split into the training and test sets. So for an accuracy score, we just pass the original dataset. However, in practice, we need to split the data into the training and test sets and only use the test set for making assessment.

```
logreg.score(x,y)
```

1.8.2 Confusion Matrix

The confusion matrix displays how many were classified correctly and incorrectly.

```
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y, y_hat)
print(confusion_matrix)
```

1.8.3 Precision, Recall, F-measure, Support

- **True Positive (tp)**: Arpita has cancer. She takes a medical test for cancer, and the test result is positive. This is a true positive because the test correctly identified that Arpita has cancer.
- **True Negative (tn)**: Fatima does not have cancer. She takes the same medical test, and the test result is negative. This is a true negative because the test correctly identified that Fatima does not have cancer.
- **False Positive (fp)**: Niu does not have cancer. However, when she takes the medical test, the test result is positive. This is a false positive because the test incorrectly identified that Niu has cancer.
- **False Negative (fn)**: Diana has cancer. However, when she takes the medical test, the test result is negative. This is a false negative because the test incorrectly identified that Diana does not have cancer.

Precision is the ratio

$$\text{Precision} = \frac{tp}{tp + fp}$$

This means the classifier cannot label a sample as positive if it is negative.

Recall is

$$\text{Recall} = \frac{tp}{tp + fn}$$

It is the ability of the classifier to find all the positive samples.

F-score is the weighted harmonic mean of the precision and recall.

$$F_{score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Support is the number of occurrences of each class in the test set.

```
from sklearn.metrics import classification_report
print(classification_report(y, y_hat))
```

1.8.4 ROC Curve

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate (TPR)
- False Positive Rate (FPR)

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.

1.8.5 Classification Threshold

In the context of binary classification, the classification threshold is a value that determines the decision boundary of the predicted probabilities obtained from a model. If the predicted probability for a certain observation is above this threshold, the model classifies the observation as the positive class; if it's below, the model classifies it as the negative class.

For example, in logistic regression, the default threshold is typically 0.5. If the predicted probability of an observation belonging to the positive class is greater than or equal to 0.5, the observation is classified as positive; otherwise, it's classified as negative.

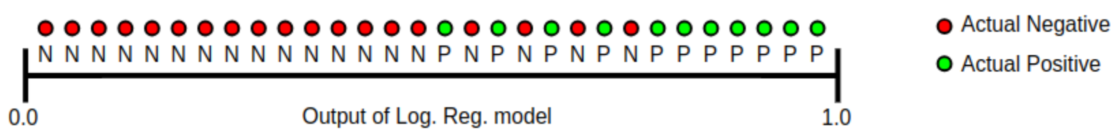
In a ROC curve, the classification threshold varies. Each point on the ROC curve represents a different threshold value, ranging from 0 to 1. Lowering the classification threshold classifies more items as positive, which increases both False Positives and True Positives.

The optimal threshold is the one that gives the best balance between True Positives and False Positives, depending on the specific context and the cost associated with misclassification. In some cases, such as when using ROC Curves, the best or optimal threshold for the classifier can be calculated directly. In other cases, it is possible to use a grid search to tune the threshold and locate the optimal value.

1.8.6 Area Under the ROC Curve (AUC)

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example.

For example, given the following examples, which are arranged from left to right in ascending order of logistic regression predictions:


N N N N N N N N N N N N N N N P N P N P N P N P P P P P P P
0.0
Output of Log. Reg. model
1.0
Actual Negative
Actual Positive

AUC represents the probability that a random positive (green) example is positioned to the right of a random negative (red) example.

AUC ranges in value from 0 to 1. A model whose predictions are 100

AUC is desirable for the following two reasons:

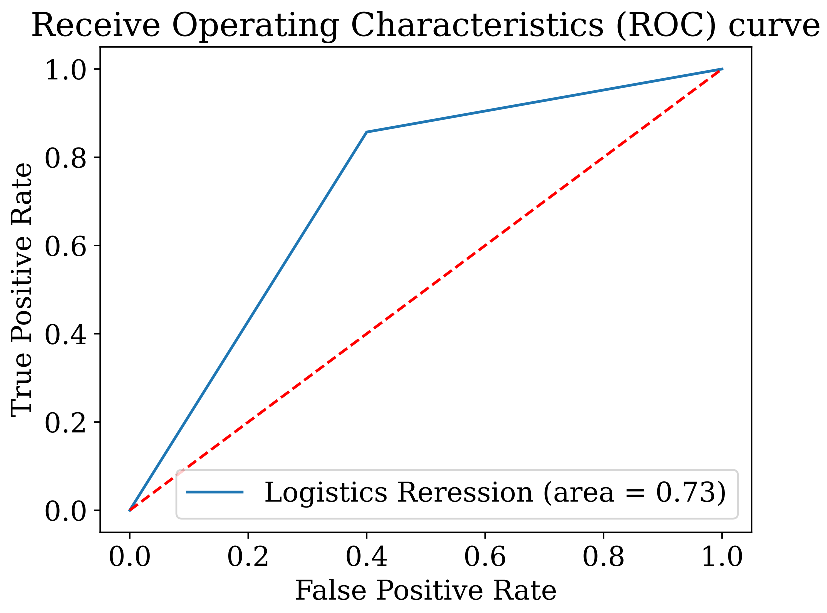
AUC is scale-invariant. It measures how well predictions are ranked, rather than their absolute values. AUC is classification-threshold-invariant. It measures the quality of the model's predictions irrespective of what classification threshold is chosen. However, both these reasons come with caveats, which may limit the usefulness of AUC in certain use cases:

Scale invariance is not always desirable. For example, sometimes we really do need well-calibrated probability outputs, and AUC won't tell us about that.

Classification-threshold invariance is not always desirable. In cases where there are wide disparities in the cost of false negatives vs. false positives, it may be critical to minimize one type of classification error. For example, when doing email spam detection, you likely want to prioritize minimizing false positives (even if that results in a significant increase of false negatives). AUC isn't a useful metric for this type of optimization.

```python
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(y, y_hat)
fpr, tpr, threshold = roc_curve(y, y_hat)

plt.figure()
plt.plot(fpr, tpr, label='Logistics Reression (area = %0.2f)'% logit_roc_auc)
plt.plot([0, 1], [0, 1], 'r--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receive Operating Characteristics (ROC) curve')
plt.legend(loc='lower right')
plt.show()
```


Receive Operating Characteristics (ROC) curve
True Positive Rate
False Positive Rate
Logistics Reression (area = 0.73)
0.0
0.2
0.4
0.6
0.8
1.0

§§§ The End §§§