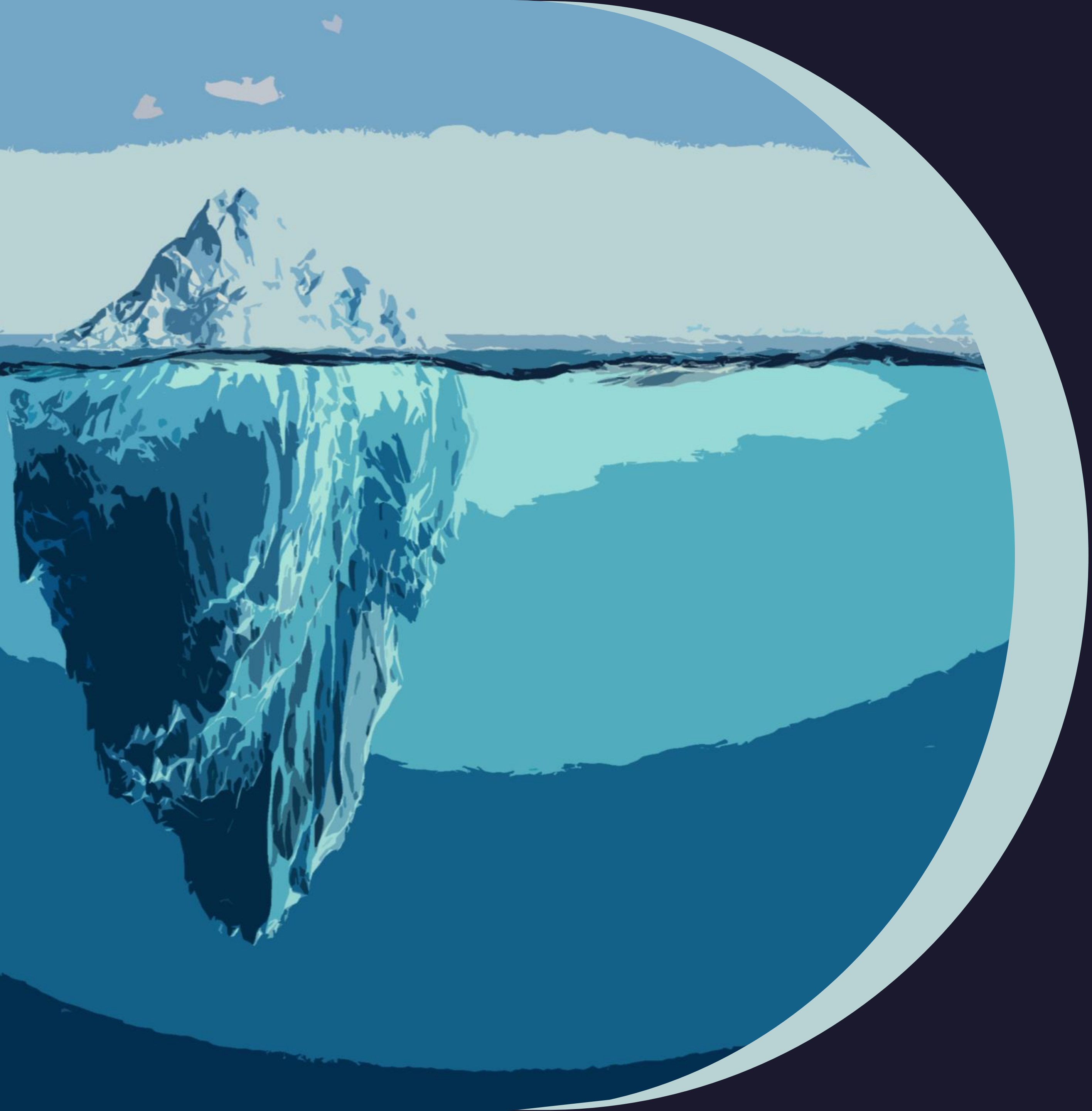


IEEE FCI - Helwan Student Branch

Logistic Regression

Hossam Ahmed

Ziad Waleed

Mario Mamdouh

Classification

definition

In machine learning, classification is a supervised learning task where the goal is to predict the category or class of a given input data point. The input data point can be represented as a set of features or attributes, and the output is a class label that belongs to a predefined set of possible labels.

CLASSIFICATION MODEL TRAINED ON LABELED DATASET WHERE EACH DATA POINT HAS A CLASS LABEL.

MODEL LEARNS PATTERNS AND RELATIONSHIPS BETWEEN INPUT FEATURES AND CLASS LABELS.

MODEL USES THIS KNOWLEDGE TO PREDICT CLASS LABEL OF NEW, UNSEEN DATA POINTS.

SPAM FILTERING: CLASSIFY EMAILS AS SPAM OR NOT SPAM BASED ON THEIR CONTENT AND METADATA.

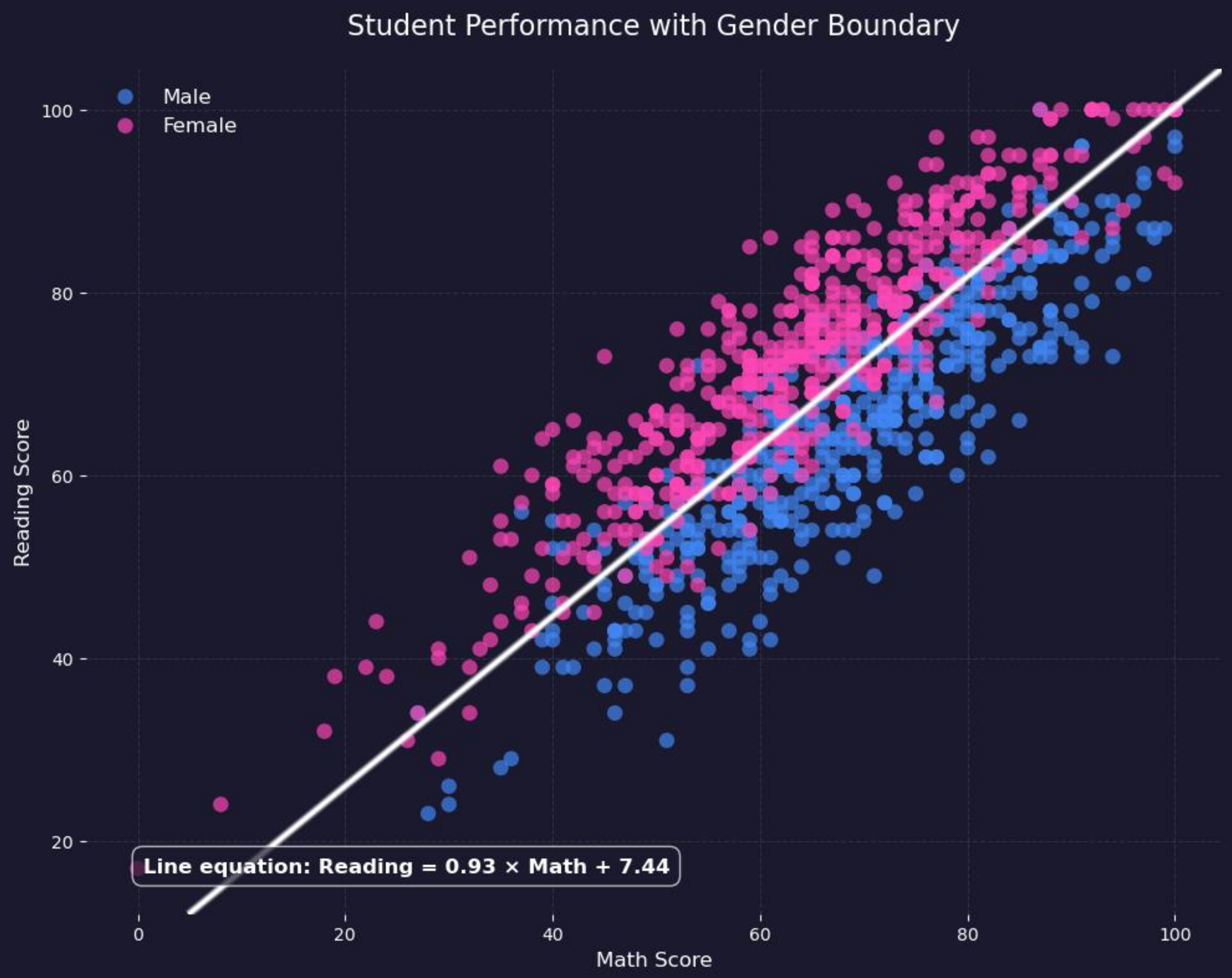
IMAGE RECOGNITION: CLASSIFY IMAGES INTO DIFFERENT CATEGORIES BASED ON THEIR VISUAL FEATURES, SUCH AS IDENTIFYING OBJECTS OR FACES IN AN IMAGE.

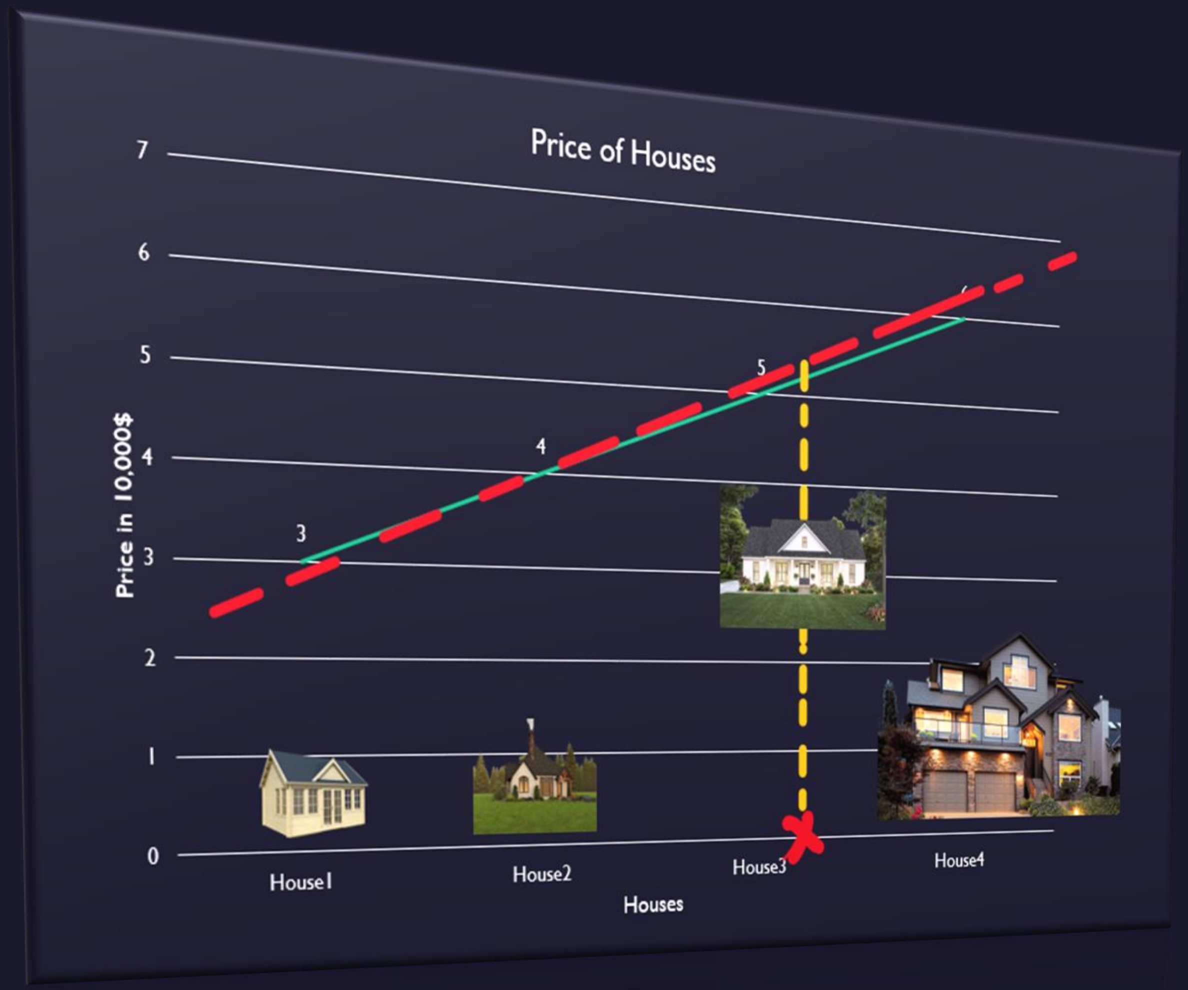
SENTIMENT ANALYSIS: CLASSIFY TEXT AS POSITIVE, NEGATIVE, OR NEUTRAL BASED ON ITS SENTIMENT OR EMOTION.

FRAUD DETECTION: CLASSIFY TRANSACTIONS AS FRAUDULENT OR LEGITIMATE BASED ON THEIR CHARACTERISTICS AND PATTERNS.

MEDICAL DIAGNOSIS: CLASSIFY PATIENTS INTO DIFFERENT DISEASE CATEGORIES BASE D ON THEIR SYMPTOMS AND MEDICAL HISTORY.

A line give quantities not classes !





- We input some features of the house
- We got a quantity the Price 💵
- Not classifying or answering a Yes , No question
- Like Is It a GOOD house or BAD house?
- A classification problem here can be is this house Big or small (e.g., from images)
- Or Is this house luxury or not to which degree
- Low , Mid , High
- So, classification can be more than one class (discrete outcome)

Probabilities is what we need!

- Our model need to give us probabilities
- Probability of being Class1 is 0.6 so we can classify this observation to be Class1 based on the input features

How to make a line classify ?

Input features give us in a linear model a quantity like the price

f(X{🏠, 🛖, 🏚️, 🏡, 🏘️}) = price 💵

How to make a line classify ?

price 💵 can be helpful in answering a classification question Is this house Cheap or Expensive !! We classified the houses into two categories based on the output of linear mode a quantity !

IEEE FCI - Helwan Student Branch

How to make a line classify ?

So, we need a function that use the linear model to give us a probabilities

Model = P(y=$Class_i$ |given features)

How to make a line classify ?

Fun($w1x + w2$)

P(class1) Expensive

P(class3) Cheap

How to make a line classify ?

Sigmoid($w1x + w2$)

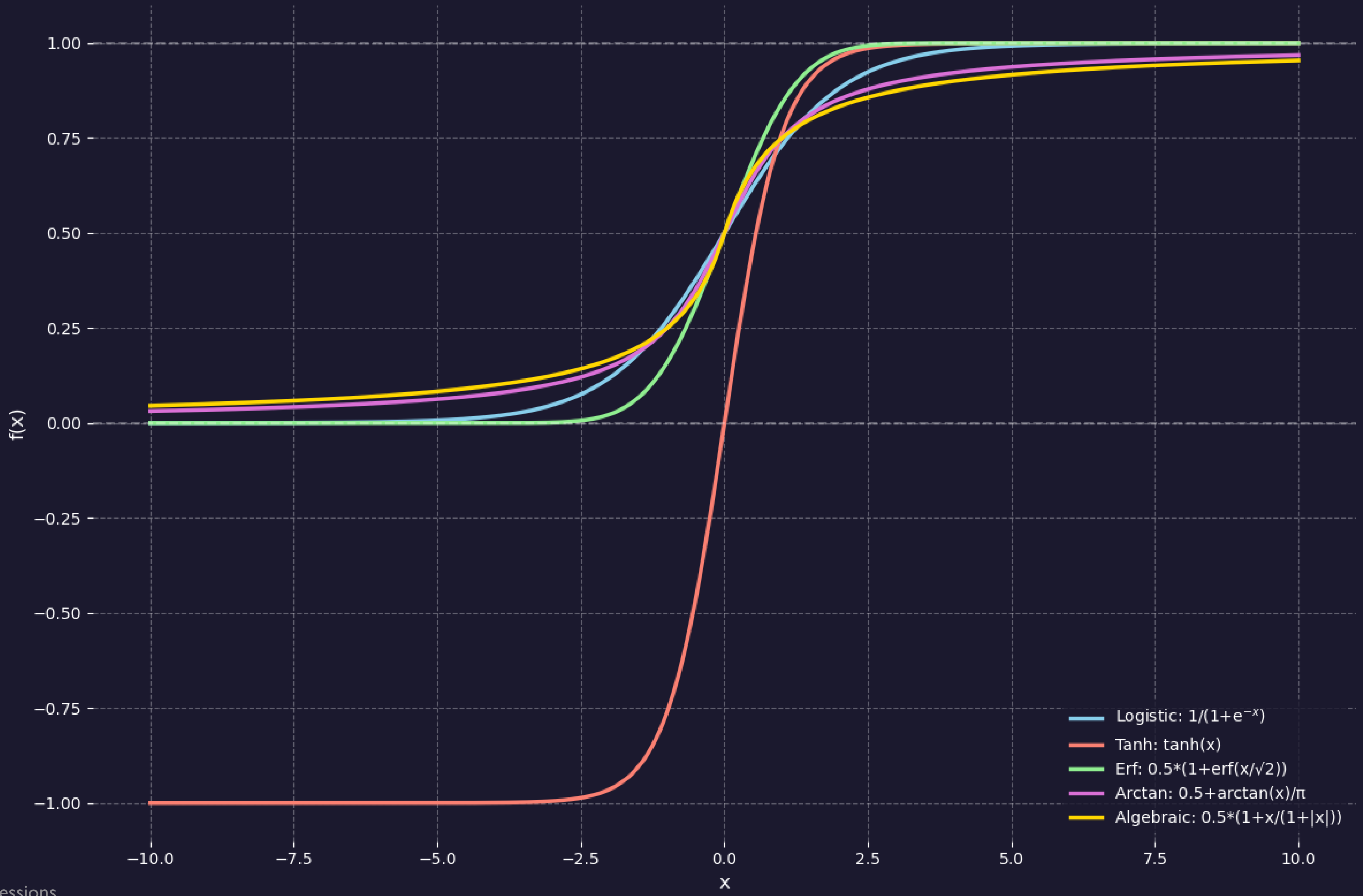
P(class1)
Expensive

P(class3)
Cheap

What is the sigmoid function?

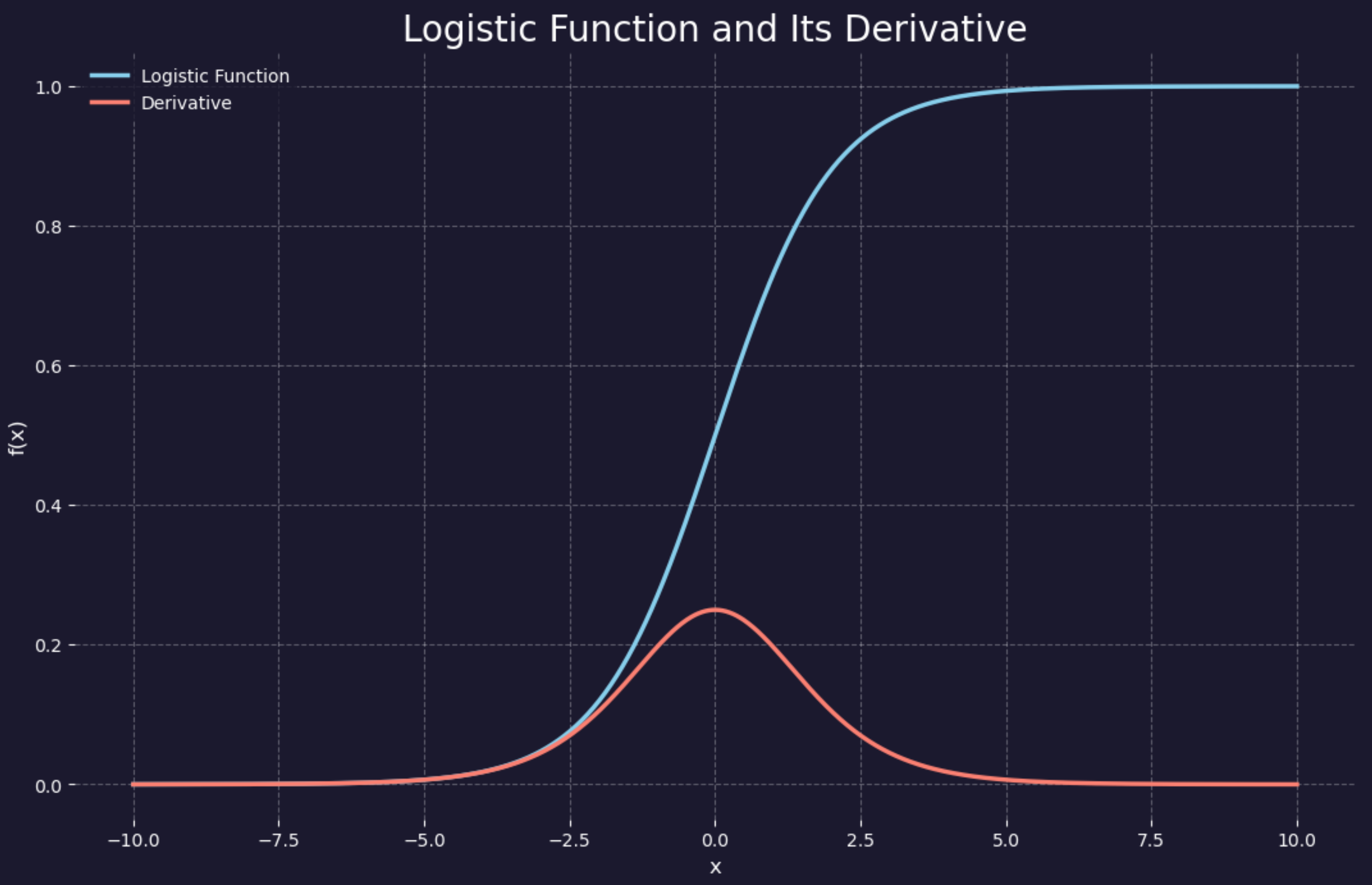
Sigmoid : are general mathematical functions that share similar properties: have S-shaped curves, map any real number to a probability between 0 and 1

What is the sigmoid function?



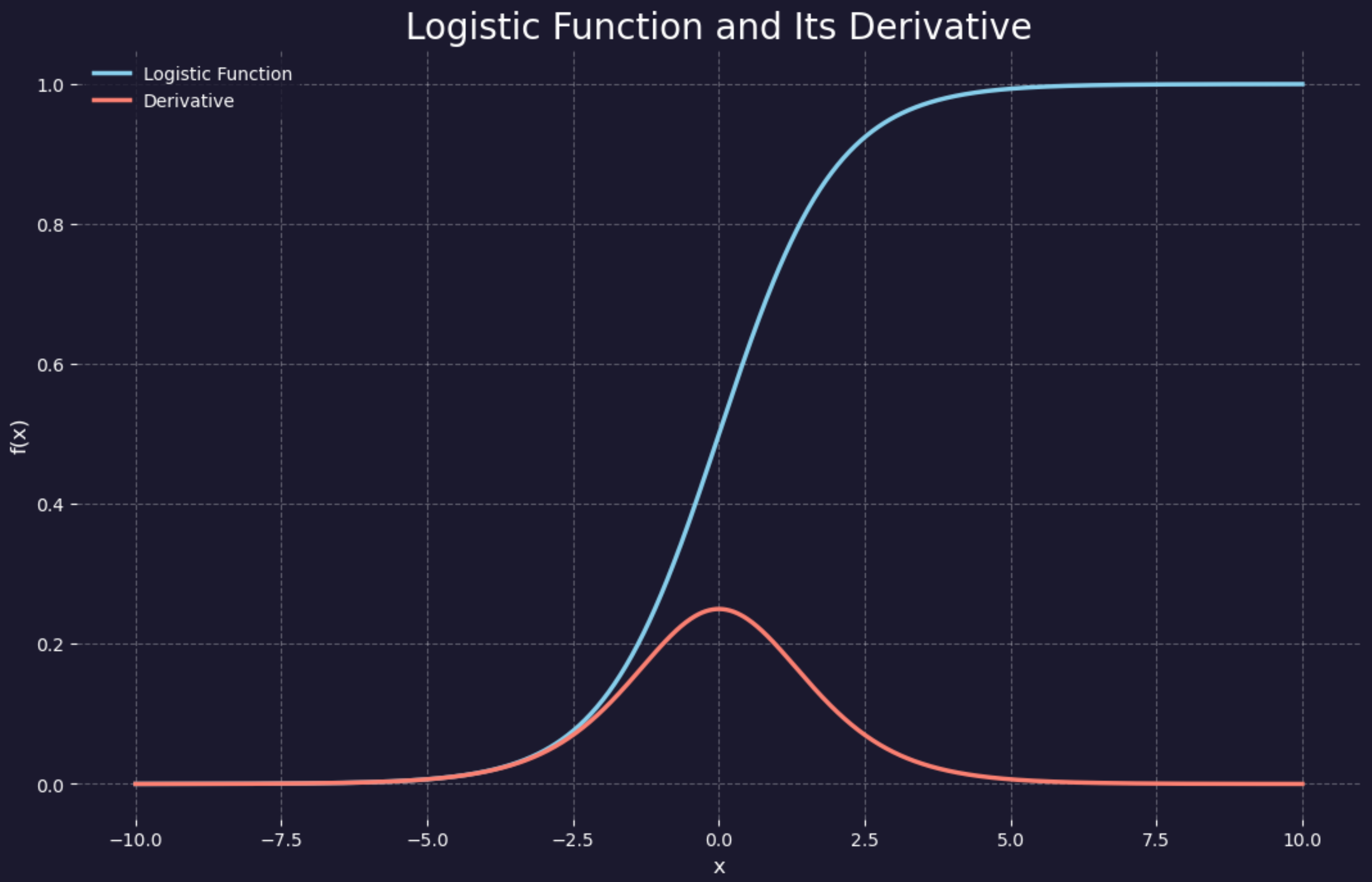
Logistic function

- Logistic function $\sigma(x)$:
  - $\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^{x}}{e^{x}+1}$
  - $\sigma(x) = 1 - \sigma(-x)$
- Logistic function derivative $\frac{\mathrm{d}}{\mathrm{dx}}\sigma(x)$:
  - $\frac{\mathrm{d}}{dx}\sigma(x) = \sigma(x)(1-\sigma(x))$
- Logistic function range is $(0,1)$, its domain is $\mathbb{R}$, it's symmetric around $x = 0$, it's differentiable everywhere.

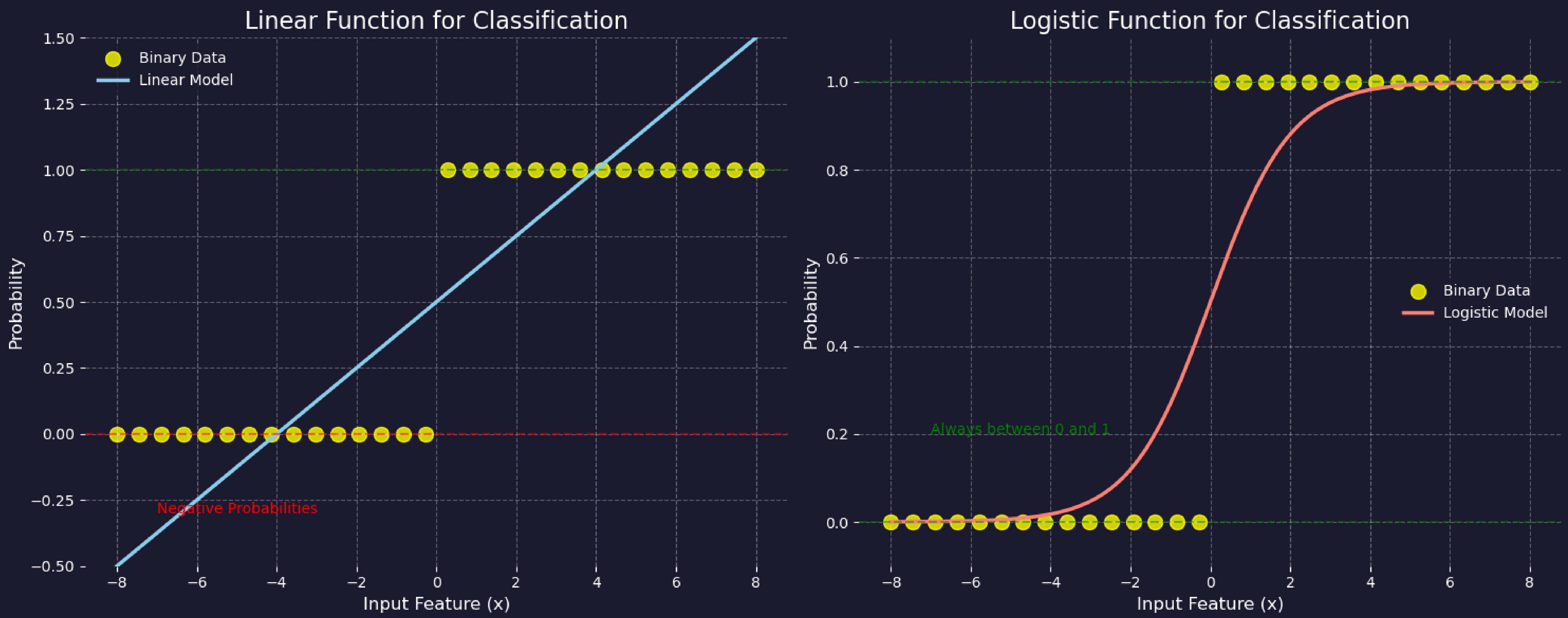


Logistic function

- Logistic function is monotonic always increasing.
  - $\lim_{x \to \infty} \sigma(x) = 1$
  - $\lim_{x \to -\infty} \sigma(x) = 0$
- Logistic function derivative $\sigma'(x)$ range is $(0, \frac{1}{4})$, meaning its maximum derivative is 0.25, and it's positive everywhere, and it's maximum around $x = 0$



Logistic function Vs Linear function



Formulation

$$Logistic\ Regression = Logistic\ function\ (Linear\ model)$$

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

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

$$\sigma(w_1x + w_0) = \frac{1}{1 + e^{-(w_1x+w_0)}}$$

$$\sigma(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1 + e^{-(\boldsymbol{w}^T\boldsymbol{x})}}$$

Probability & odds

- Likelihood refers to the process of determining the best data distribution given a specific situation in the data.
  - $P(distribution\,|data)$

- Logistic regression models ***the probability of a class given some input features.*** It can be interpreted through the lens of likelihood: given the features of a data point, ***what is the most likely class (0 or 1) that could have generated it?***

- Logistic regression estimates the probability that a binary outcome $y \in \{0,1\}$ occurs, given input features $\boldsymbol{x}$. It does so by modeling the **conditional probability**

$$p = P(y = 1\,|x)$$
$$\mathrm{q} = 1 - p = P(y = 0\,|x)$$

Probability & odds

- In logistic regression the target variable Y follows a Bernoulli distribution

$$p = P(y = 1 \,|x)$$
$$\mathrm{q} = 1 - p = P(y = 0 \,|x)$$

- A Bernoulli distribution is a discrete probability distribution for a **single** trial that has exactly **two possible outcomes**: success (1) or failure (0).
  - Success p, or failure $q = 1 - p$

- The PMF of this distribution, over possible outcomes k is
  - can be written as $f(k;p) = p^k \times q^{1-k} = \boldsymbol{p^k \times (1-p)^{1-k}}$ for $k \in \{0,1\}$

- We can express the **likelihood probability** of one data point as

$$P(Y = y|X = x) = \sigma(\boldsymbol{w}^T\boldsymbol{x})^y.[1 - \sigma(\boldsymbol{w}^T\boldsymbol{x})]^{1-y}$$

Probability & odds

- The meaning of ***ODDS*** is the probability that one thing is so or will happen rather than

$$Odds = \frac{p}{q} = \frac{P(y=1|X)}{P(y=0|X)} = \frac{P(y=1|X)}{1-P(y=1|X)}$$

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

$$= \frac{\frac{e^x}{\cancel{e^x+1}}}{\frac{1}{\cancel{e^x+1}}} = e^x$$

Logit function

$$Odds = \frac{\sigma(x)}{1 - \sigma(x)} = e^{x}$$

Let's take the ln (natural logarithm) of both sides of the equation

$$\ln(Odds) = \ln(e^{x})$$

$$\text{logit} = \ln(Odds) = x$$

The natural logarithm of the odds give us the **logit function (log-odd)** which is the inverse function of the logistic function, ***so if you have a logistic model, you can use the logit to get its parameters***.

$$logit\big(\sigma(w^{T}x)\big) = w^{T}x$$

Log likelihood

- Logistic regression models ***the probability of a class given some input features***. It can be interpreted through the lens of likelihood: given the features of a data point, ***what is the most likely class (0 or 1) that could have generated it?***

- The likelihood of all data points is the product of all the likelihoods, and we know the probability mass function follows a **Bernoulli distribution**

$$\boldsymbol{L}(\boldsymbol{w}) = \prod_{i=1}^{n} P(Y = y_i | X = x_i)$$

$$\boldsymbol{L}(\boldsymbol{w}) = \prod_{i=1}^{n} \sigma(\boldsymbol{w}^T \boldsymbol{x}_i)^{y_i} . [1 - \sigma(\boldsymbol{w}^T \boldsymbol{x}_i)]^{1-y_i}$$

Log likelihood

- Let's take the log (or ln) of this function and recall that $\log(ab) = \log(a) + \log(b)$ and $\log(x^p) = p.\log(x)$ *logarithmic product and power rules.*

$$\boldsymbol{LL(w)} = \boldsymbol{log}\ \boldsymbol{L(w)} = \boldsymbol{log} \prod_{\boldsymbol{i=1}}^{\boldsymbol{n}} P(Y = y_i | X = x_i)$$

$$\boldsymbol{LL(w)} = \boldsymbol{log} \prod_{\boldsymbol{i=1}}^{\boldsymbol{n}} \sigma(\boldsymbol{w}^T \boldsymbol{x_i})^{y_i} . [1 - \sigma(\boldsymbol{w}^T \boldsymbol{x_i})]^{1-y_i}$$

$$\boldsymbol{LL(w)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} y_i\ \boldsymbol{log}\ \sigma(\boldsymbol{w}^T \boldsymbol{x_i}) + (1 - y_i) \boldsymbol{log} (1 - \sigma(\boldsymbol{w}^T \boldsymbol{x_i}))$$

Log likelihood

- We can simplify the notation and refer to this equation as the **log-likelihood**. By maximizing it (MLE), we find the optimal weights for the logistic regression model

$$LL(w) = log\ L(w) = log \prod_{i=1}^{n} P(Y = y_i | X = x_i)$$

$$LL(w) = log \prod_{i=1}^{n} p^{y_i}.[q]^{1-y_i}$$

$$LL(w) = \sum_{i=1}^{n} y_i\ log\ p + (1 - y_i) log(q)$$

Log likelihood interpretation

- Log likelihood LL_i for a data point i, $LL_i = y_i \, \boldsymbol{log} \, p + (1 - y_i)\boldsymbol{log}(q)$
  - *And for all datapoints* $\boldsymbol{LL(w)} = \sum_{i=1}^{n} y_i \, \boldsymbol{log} \, p + (1 - y_i)\boldsymbol{log}(q)$

- Log likelihood LL_i for a data point i,$LL_i = y_i \, \boldsymbol{log} \, p + (1 - y_i)\boldsymbol{log}(q)$
  - *If* $p = 0$ prediction, and it was a True prediction $y_i = 0$
    - $LL_i = y_i \, \boldsymbol{log} \, p + (1 - y_i)\boldsymbol{log}(q) = 0 \, \boldsymbol{log} \, 0 + (1 - 0)\boldsymbol{log}(1)$
    - $\boldsymbol{log(0) = -\infty, log(1) = 0}$
    - $p \neq 0$ *exactly (it can't) but it's approaching zero, logistic function can't exactly equals zero neither 1, but it can approach them.*
    - *So* $\boldsymbol{log(0)}$ *a large penalty that approach* $-\infty$
    - $0 \, \boldsymbol{log} \, 0$ *means we got <u>zero penalty</u> (don't plug zero but maybe 0.0001)*
    - $(1 - 0)\boldsymbol{log}(1) = \boldsymbol{1 * 0 = 0}$ *here also we got <u>zero penalty</u> because* $\log(1)$ ***is zero,*** and it's a number to close too one not exactly 1.

Log likelihood interpretation

- Log likelihood LL_i for a data point i,$LL_i = y_i\, \boldsymbol{log}\, p + (1 - y_i)\boldsymbol{log}(q)$
  - *If* $p = 0$ prediction, and it was a False prediction $y_i = 1$
    - $LL_i = y_i\, \boldsymbol{log}\, p + (1 - y_i)\boldsymbol{log}(q) = 1\, \boldsymbol{log}\, 0 + (1 - 1)\boldsymbol{log}(1)$
    - $\boldsymbol{log}(\mathbf{0}) = -\infty, \boldsymbol{log}(\mathbf{1}) = \mathbf{0}$
    - *So* $\boldsymbol{log}(\mathbf{0})$ *a large penalty that approach* $-\infty$
    - $1\, \boldsymbol{log}\, 0$ *means we got a large penalty* $\approx -\infty$
    - $(1 - 1)\boldsymbol{log}(1) = \mathbf{0} * \mathbf{0} = \mathbf{0}$ *here also we got zero penalty because* $\log(1)$ ***is zero,*** and it's a number to close too one not exactly 1.
    - ***The reason the second term in both cases were neglectable is because it's designed to work when $p = 1$, the prediction of the other class.***

- We got more negative values as penalties the more our prediction is far from the true label, you can think of the log likelihood as a score function, the higher the score the better, so the higher the sum of scores (you grades 🎓 🎉) the model is better.

Maximizing Log likelihood

- We want to find the model whose weights give us the highest sum of scores over all the data points.

- Log likelihood is the score function, who we want to maximize :

$$\max_{\mathbf{w}} \sum_{i=1}^{n} y_i \, log \; p + (1 - y_i) log(q)$$

$$\max_{\mathbf{w}} \sum_{i=1}^{n} y_i \, log \; \sigma(\boldsymbol{w}^T \boldsymbol{x}_i) + (1 - y_i) log(1 - \sigma(\boldsymbol{w}^T \boldsymbol{x}_i))$$

- Let's get its derivative, <u>if</u> we can find the derivative and make it equal to zero, we would find the parameters that maximize the score (because we found the critical points).

Maximizing Log likelihood

$$\frac{\partial LL(w)}{\partial w_j} = \frac{\partial LL(w)}{\partial p} * \frac{\partial p}{\partial w_j}$$

$$\frac{\partial LL(w)}{\partial w_j} = \frac{\partial LL(w)}{\partial p} * \frac{\partial p}{\partial z} * \frac{\partial z}{\partial w_j}$$

$$\boxed{\begin{gathered} p = \sigma(z) = \sigma(w^T x) \\ z = w^T x \end{gathered}}$$

We now need to find each partial derivative of this chain.

Maximizing Log likelihood

$$\frac{\partial LL(w)}{\partial p} = \frac{\partial}{\partial p}\left(y \log p + (1-y)\log(1-p)\right)$$

$$\frac{\partial LL(w)}{\partial p} = y\frac{\partial}{\partial p}\log p + (1-y)\frac{\partial}{\partial p}\log(1-p)$$

$$\boxed{\frac{\partial LL(w)}{\partial p} = \frac{y.1}{p} + \frac{(1-y).(-1)}{1-p} = \frac{y}{p} - \frac{1-y}{1-p}}$$

Maximizing Log likelihood

$$\frac{\partial p}{\partial z} = \frac{\partial}{\partial z}(\sigma(z))$$

$$\frac{\partial z}{\partial w_j} = \frac{\partial z}{\partial w_j}(w^T x)$$

$$\boxed{\frac{\partial p}{\partial z} = \sigma(z)\big(1 - \sigma(z)\big)}$$

$$\boxed{\frac{\partial z}{\partial w_j} = x_j \qquad only\ x_j\ interact\ with\ w_j}$$

$$\boxed{\frac{\partial LL(w)}{\partial p} = \frac{y}{p} - \frac{1-y}{1-p}}$$

$$\boxed{\frac{\partial LL(w)}{\partial w_j} = \frac{\partial LL(w)}{\partial p} * \frac{\partial p}{\partial z} * \frac{\partial z}{\partial w_j}}$$

Maximizing Log likelihood

$$\frac{\partial LL(w)}{\partial w_j} = \frac{\partial LL(w)}{\partial p} * \frac{\partial p}{\partial z} * \frac{\partial z}{\partial w_j}$$

$$\frac{\partial LL(w)}{\partial w_j} = \left(\frac{y}{p} - \frac{1-y}{1-p}\right) * \left(\sigma(z)\left(1-\sigma(z)\right)\right) * (x_j)$$

$$\frac{\partial LL(w)}{\partial w_j} = \left(\frac{y}{p} - \frac{1-y}{1-p}\right) * \left(p(1-p)\right) * (x_j)$$

Maximizing Log likelihood

$$\frac{\partial LL(\boldsymbol{w})}{\partial \boldsymbol{w}_j} = \left(\cancel{\boldsymbol{p}}(\boldsymbol{1}-\boldsymbol{p})\frac{\boldsymbol{y}}{\cancel{\boldsymbol{p}}} - \boldsymbol{p}\cancel{(\boldsymbol{1}-\boldsymbol{p})}\frac{\boldsymbol{1}-\boldsymbol{y}}{\cancel{\boldsymbol{1}-\boldsymbol{p}}}\right) * (\boldsymbol{x}_j)$$

$$\frac{\partial LL(\boldsymbol{w})}{\partial \boldsymbol{w}_j} = (\boldsymbol{y}(\boldsymbol{1}-\boldsymbol{p}) - (\boldsymbol{1}-\boldsymbol{y})\boldsymbol{p}) * (\boldsymbol{x}_j)$$

$$\frac{\partial LL(\boldsymbol{w})}{\partial \boldsymbol{w}_j} = (\boldsymbol{y} - \cancel{\boldsymbol{y}\boldsymbol{p}} - \boldsymbol{p} + \cancel{\boldsymbol{y}\boldsymbol{p}}) * (\boldsymbol{x}_j)$$

Maximizing Log likelihood

$$\frac{\partial LL(\boldsymbol{w})}{\partial \boldsymbol{w}_j} = (\boldsymbol{y} - \boldsymbol{p}) * (\boldsymbol{x}_j)$$

$$\boxed{\frac{\partial LL(\boldsymbol{w})}{\partial \boldsymbol{w}_j} = \left(\boldsymbol{y} - \boldsymbol{\sigma}(\boldsymbol{w}^T \boldsymbol{X})\right) * (\boldsymbol{x}_j)}$$

This equation $\frac{\partial LL(w)}{\partial w_j} = \mathbf{0}$ can't be solved algebraically (analytically) as we can't separate the x alone, it's a transcendental equation, so we must solve it with ***iterative optimization method.***

Log likelihood optimization

- We can use an iterative method like gradient ascent or newton method to maximize the score function, by finding better weights.

- Gradient Ascent (GA)

$$w_j^{new} = w_j^{old} + \eta \frac{\partial LL(\boldsymbol{w})}{\partial \boldsymbol{w_j}}$$

$$w_j^{new} = w_j^{old} + \eta \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} [\boldsymbol{y_i} - \boldsymbol{\sigma}(\boldsymbol{w^T x_i})]\boldsymbol{x_j^i}$$

Log likelihood optimization

- We can reverse any optimization problem, from maximization to minimization or vise versa by multiplying the objective function by (-1).
  - $\boldsymbol{f}(\boldsymbol{x}^{*}) \geq \boldsymbol{f}(\boldsymbol{x})\ \forall x \Rightarrow -\boldsymbol{f}(\boldsymbol{x}^{*}) \leq -\boldsymbol{f}(\boldsymbol{x})\ \forall x$, where $f(x^{*})$ the optimal value.

- We can use Gradient descent (GD) if we reversed the objective function which is the log likelihood LL, to be the **negative log likelihood** NLL (cross entropy)

$$NLL = -LL$$

$$-\left(\sum_{i=1}^{n} y_i\, \boldsymbol{log}\ p + (1 - y_i)\boldsymbol{log}(q)\right)$$

Log likelihood optimization

- Gradient Descent

$$w_j^{new} = w_j^{old} - \eta \frac{\partial NLL(\boldsymbol{w})}{\boldsymbol{\partial w_j}}$$

$$w_j^{new} = w_j^{old} - \eta \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} [-\boldsymbol{y_i} + \boldsymbol{\sigma}(\boldsymbol{w^T x_i})]\boldsymbol{x_j^i}$$

Classification Model Evaluation

- P/N refer to the **predicted class** — whether the model predicted the sample as **positive (P)** (e.g., class A) or **negative (N)** (e.g., any class other than A).

- T/F refer to whether the prediction is **correct** or **incorrect** compared to the actual label.

- $\boldsymbol{P}$ (positive) it's all the samples that are Truly (actually) positive, that include :
  - $\boldsymbol{TP}$ Positive samples that were correctly classified as positive.
  - $\boldsymbol{FN}$ Positive samples that were incorrectly classified as negative.
  - $\boldsymbol{P = TP + FN}$ all positive samples in the dataset.

- $\boldsymbol{N}$ (negative) it's all the samples that are Truly (actually) negative, that include :
  - $\boldsymbol{TN}$ Negative samples that were correctly classified as negative.
  - $\boldsymbol{FP}$ Negative samples that were incorrectly classified as positive.
  - $\boldsymbol{P = TN + FP}$ all negative samples in the dataset.

Classification Model Evaluation

- **Accuracy** is the proportion of **correct predictions** made by a classification model out of **all predictions**.
  - All predictions $P + N$
  - Correct prediction $TP + TN$
  - Accuracy $\frac{TP+TN}{P+N}$
- Accuracy can be misleading in **imbalanced datasets**, where one class heavily outweighs the other.
  - Imagine a dataset for detecting a rare disease where:
  - 95% are healthy
  - 5% have the disease
  - A model that always predicts “healthy” would be **95% accurate**, but it **misses all actual disease cases**, making it useless for detection.

Classification Model Evaluation

IEEE FCI - Helwan Student Branch

- During WWII, the U.S. Air Force analyzed damaged planes to determine where to add armor.
  - Data: They looked at planes that returned from missions, focusing on where they were damaged
  - **Wald's Insight:** Statistician **Abraham Wald** realized the planes that returned were not a representative sample.
  - **Key Finding:** Planes that didn't return likely sustained critical damage in areas not visible in the data (e.g., engines, cockpit).
  - **Recommendation:** Reinforce areas with **less damage** on the returning planes, like the **wings**, as these were the most vulnerable.

Just as the wrong evaluation of plane damage led to faulty conclusions, using the wrong **evaluation metric** can be **deceptive** and lead to incorrect decisions.

Classification Model Evaluation

- **True positive rate (TPR)**
  - Recall
  - Sensitivity
  - Hit rate
  - $TPR = 1 - FNR = \frac{TP}{P} = \frac{TP}{TP+FN}$
- **False negative rate (FNR)**
  - Miss rate
  - Type II error
  - $FNR = 1 - TPR = \frac{FN}{P} = \frac{FN}{TP+FN}$

$$\boldsymbol{P = TP + FN}$$

Let’s divide both side by $\boldsymbol{P}$

$$\boldsymbol{1 = \frac{TP + FN}{P} = \frac{TP}{P} + \frac{FN}{P}}$$

$$\boldsymbol{1 = \frac{TP}{P} + \frac{FN}{P} = TPR + FNR}$$

Classification Model Evaluation

- **Recall** (TPR) is the proportion of actual positive instances that are correctly identified by the model.
  - $TPR = 1$, Perfect recall — the model correctly identifies all positive instances.
  - $TPR = 0$, The model fails to identify any positive instances (it predicts all negatives).
- TPR measures how good the model is at **capturing positive examples**.
  - In medical diagnostics, for example, this would represent how well the test detects **sick individuals**.

- The **False Negative Rate (FNR)** is the proportion of **actual positive instances** that the model incorrectly labels as negative.
  - $FNR = 0$, No false negatives — the model correctly identifies all positive instances.
  - $TPR = 0$, All positive instances are missed by the model (it incorrectly predicts them as negative).

Classification Model Evaluation

- **True negative rate (TNR)**
  - Specificity
  - $TNR = 1 - FPR = \frac{TN}{N} = \frac{TN}{TN+FP}$
  - $TNR = 1$, Perfect specificity, the model correctly identifies all negative instances as negative.
- **False positive rate (FNR)**
  - Probability of false alarm
  - Fall out
  - Type I error
  - $FPR = 1 - TNR = \frac{FP}{N} = \frac{FP}{TN+FP}$
  - $FPR = 0$, good no false positive

$$\boldsymbol{N = TN + FP}$$

Let's divide both side by $\boldsymbol{N}$

$$\boldsymbol{1 = \frac{TN + FP}{N} = \frac{TN}{N} + \frac{FP}{N}}$$

$$\boldsymbol{1 = \frac{TN}{N} + \frac{FP}{N} = TNR + FPR}$$

Classification Model Evaluation

FCI - Helwan
Student Branch

- PP (predicted positive) it's the total number of predicted positive samples, that include :
  - TP Positive samples that were correctly classified as positive.
  - FP Negative samples that were incorrectly classified as positive.
  - $PP = TP + FP$ all the predicted positive samples in the dataset, whether they are True of False.

- PN (predicted negative) it's all the total number of predicted negative samples, that include :
  - TN Negative samples that were correctly classified as negative.
  - FN Positive samples that were incorrectly classified as negative.
  - $PN = TN + FN$ all the predicted negative samples in the dataset, whether they are True of False.

Classification Model Evaluation

- **Positive predictive value (PPV)**
  - Precision 🌟
  - $PPV = 1 - FDR = \frac{TP}{PP} = \frac{TP}{TP+FP}$
  - $PPV = 1$, perfect precision, all positive predictions made by the model are correct, no false positive.
  - But FN can exist.
- **False discovery rate (FDR)**
  - $FDR = 1 - PPV = \frac{FP}{PP} = \frac{FP}{TP+FP}$
  - $FDR = 0$ no false discoveries.
  - $FDR = 1$ all predictive + are false.

$$\boldsymbol{PP = TP + FP}$$

Let's divide both side by $\boldsymbol{PP}$

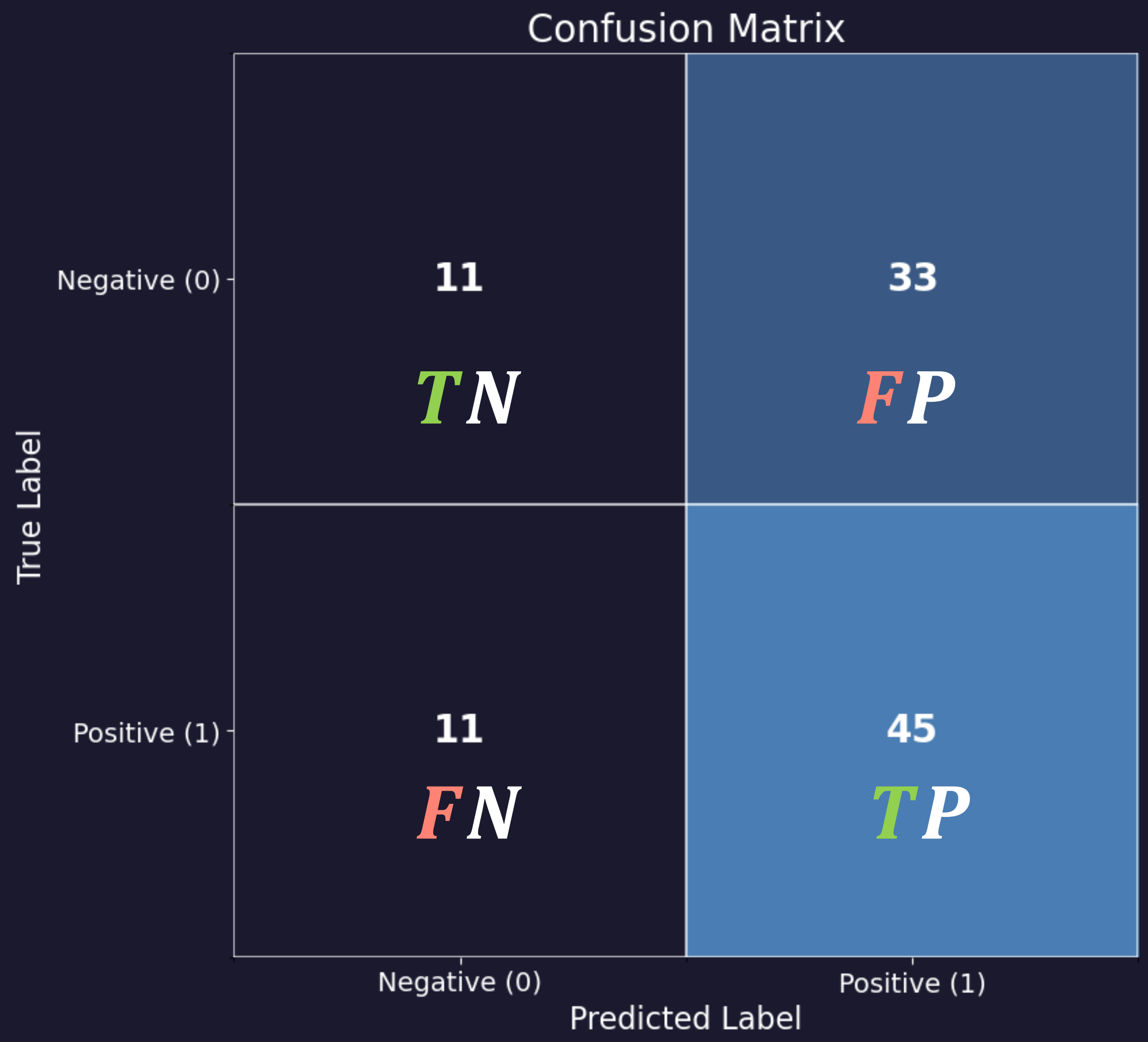
$$\boldsymbol{1 = \frac{TP + FP}{PP} = \frac{TP}{PP} + \frac{FP}{PP}}$$

$$\boldsymbol{1 = \frac{TP}{PP} + \frac{FP}{PP} = PPV + FDR}$$

Classification Model Evaluation

- **A confusion matrix** is a table used to evaluate the performance of classification model.
  - The main diagonal where True label match the prediction, is the correct predictions.
  - Any square outside this diagonal are misclassified samples.

- Calculate the metrics
  - Accuracy 0.56
  - Precision 0.57
  - Recall 0.80
  - Why the Recall is high?

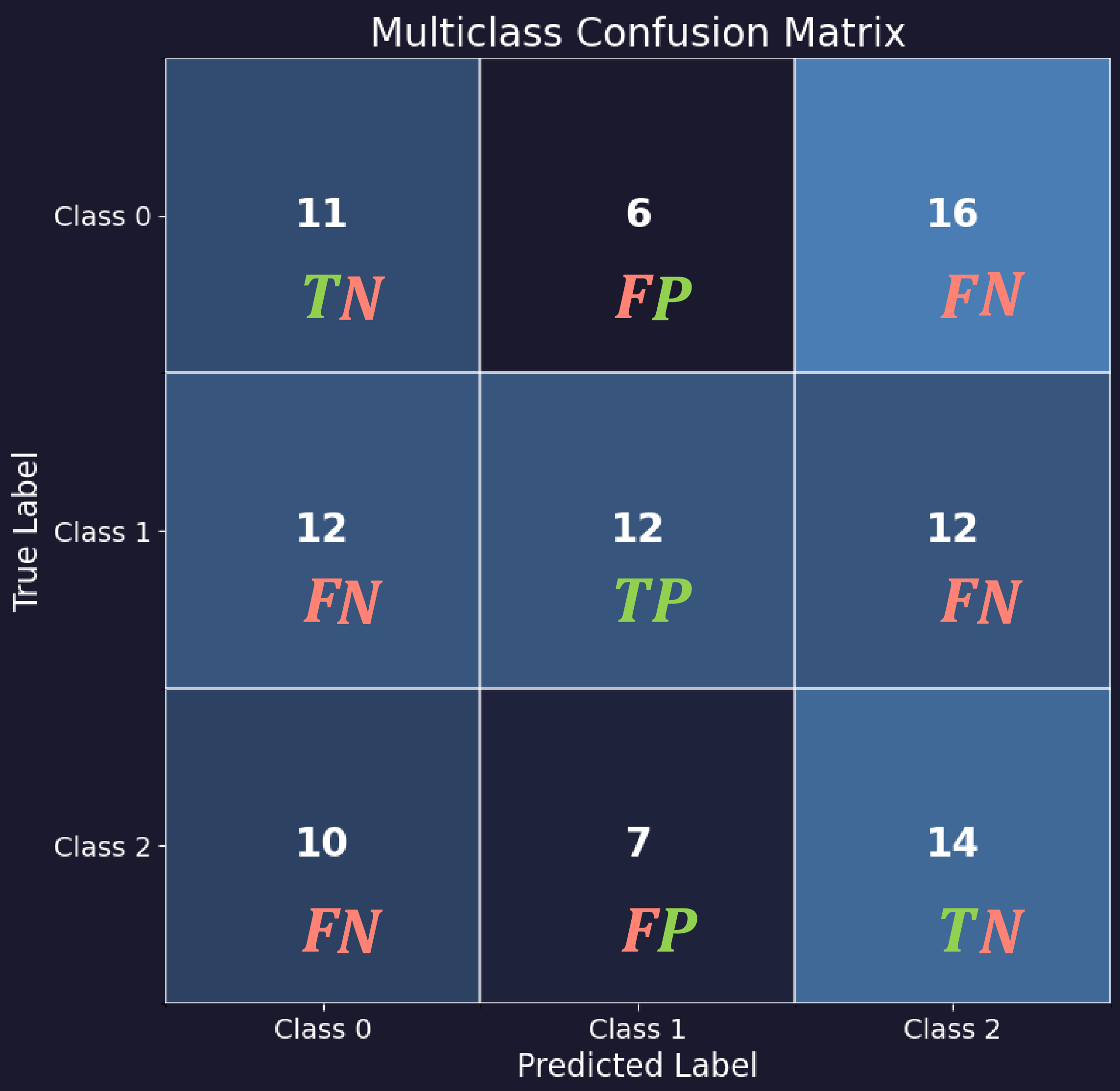


Classification Model Evaluation

- In multiclass confusion matrix, you calculate relative to some class, like are looping over the classes.

- If you are calculating the metrics for class 0, you would consider this class the positive class and any other classes are negative.

- Ex. Precision for class 1 $(PPV = \frac{TP}{TP+FP})$
  - $PPV_{class1} = \frac{12}{12+6+7} = 0.48$



References

- https://en.wikipedia.org/wiki/Logistic_regression
- https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a
- https://iopscience.iop.org/article/10.1088/1742-6596/1725/1/012014/pdf
- https://en.wikipedia.org/wiki/Newton%27s_method
- Jurafsky, D., & Martin, J. H. (2025, January 12). *Speech and language processing* (Draft of Chapter 5)
- Monroe, W. (2017, August 14). *CS 109: Logistic regression (Lecture notes #22).* Stanford University.

See 👀

- https://mlu-explain.github.io/logistic-regression/ [🌟 visual article]
- https://mlpocket.com/ml/supervised/logistic-regression [visual article]
- http://deeplearning.stanford.edu/tutorial/supervised/SoftmaxRegression/ [SoftMax regression]

Thank You

Time for code