ARTIFICIAL INTELLIGENCE NEEDS REAL INTELLIGENCE

Classification I

Professor: Tianyuan ZHANG
tianyuan.zhang@kedgebs.com

KEDGE BUSINESS SCHOOL

Recap of Previous Session

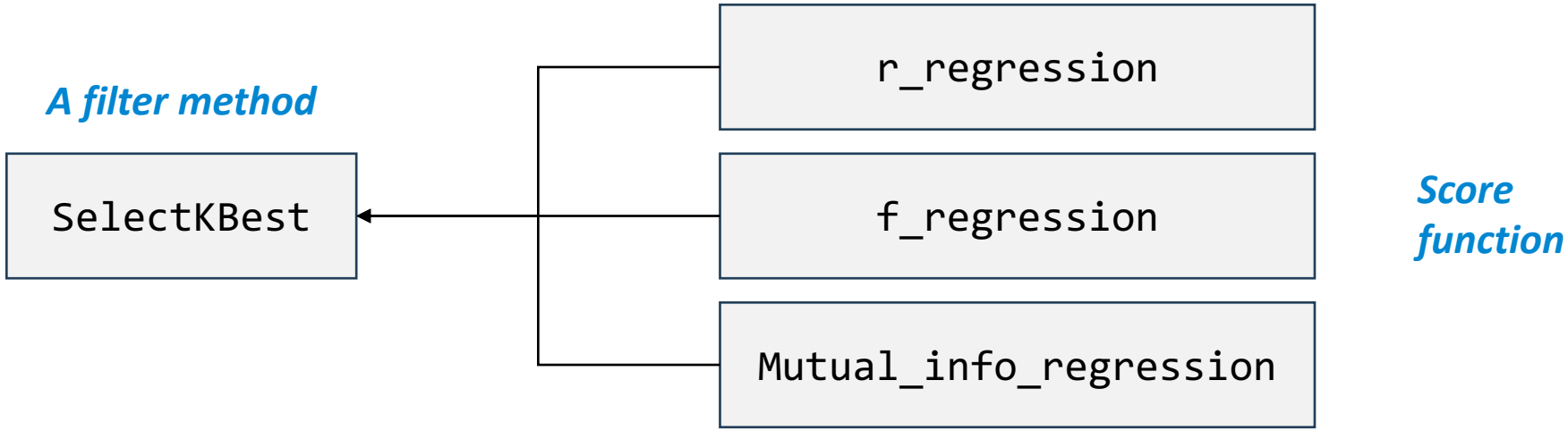
- Regression
  - Supervised learning algorithms
  - Learn the relationships between features $(x_1, x_2, \dots, x_n)$ and label y
  - Predict continuous numeric values based on the learned relationships
- Linear regression
  - Simple linear regression $y = \beta_0 + \beta_1 x$
  - Multiple linear regression $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_N x_N$
  - Polynomial regression $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_n x^n$

Recap of Previous Session

- Evaluation metrics for regression
  - Measure the model performance on unseen data
  - Residual, $\epsilon_i = y_i - \hat{y}_i$
  - Mean squared error, $\mathrm{MSE} = \frac{1}{n}\sum_{i=0}^{n} \epsilon_i^2$
  - Root mean squared error, $\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$
  - Mean absolute error, $\mathrm{MAE} = \frac{1}{n}\sum_{i=0}^{n} |\epsilon_i|$
  - Coefficient of determination, $R^2 = 1 - \frac{\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n}(y_i - \bar{y})^2}$, $\bar{y} = \frac{1}{n}\sum_{i=0}^{n} y_i$

Recap of Previous Session

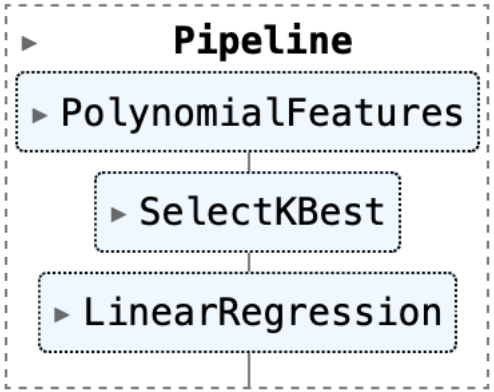
- Feature selection
  - Select a subset of relevant features for model construction
  - Improve model performance, reduce model complexity, reduce training time
  - Perform feature selection with the training dataset, not the entire dataset



Recap of Previous Session

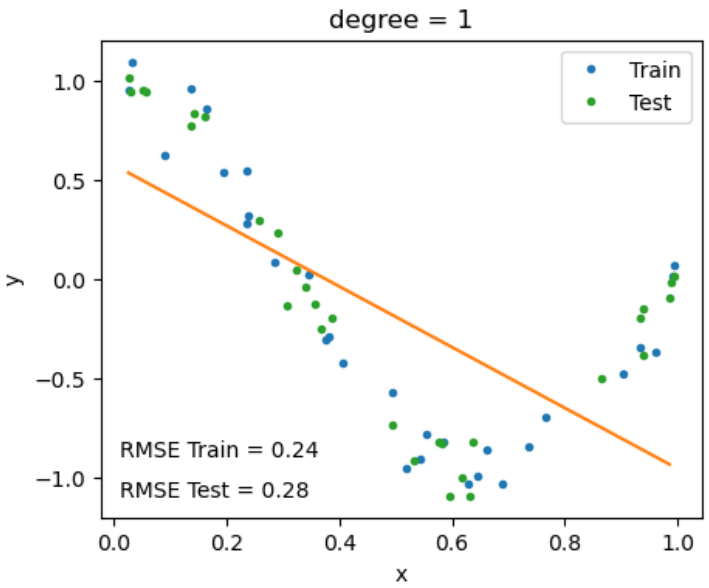
- Pipeline in `sklearn`
  - `sklearn.pipeline.Pipeline`
    - Define a pipeline of a list of transforms with a final estimator
      - Transforms: Feature selection, Polynomial features generation
      - Estimator: Linear regression model

```
# construct pipeline
pipeline = Pipeline(
    [
        ('poly', PolynomialFeatures(degree = degree, include_bias = False)),
        ('selector', SelectKBest(score_func = f_regression, k = 8)),
        ('mlr', LinearRegression(fit_intercept = True))
    ]
)
```

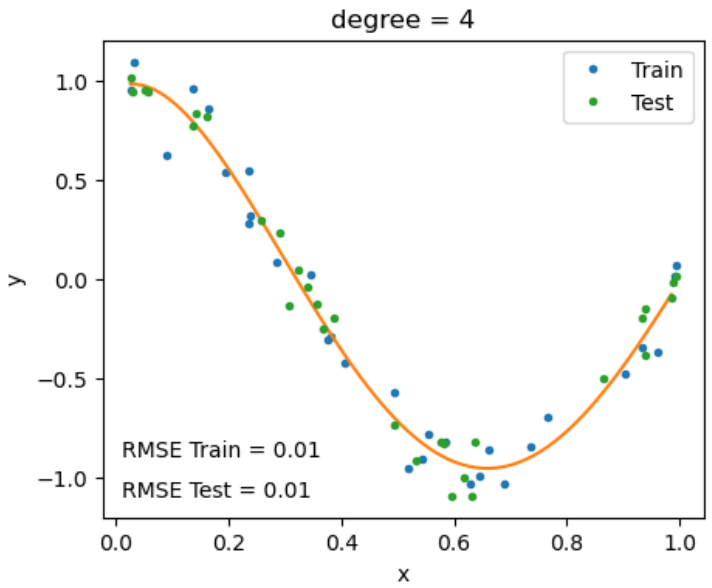


Recap of Previous Session

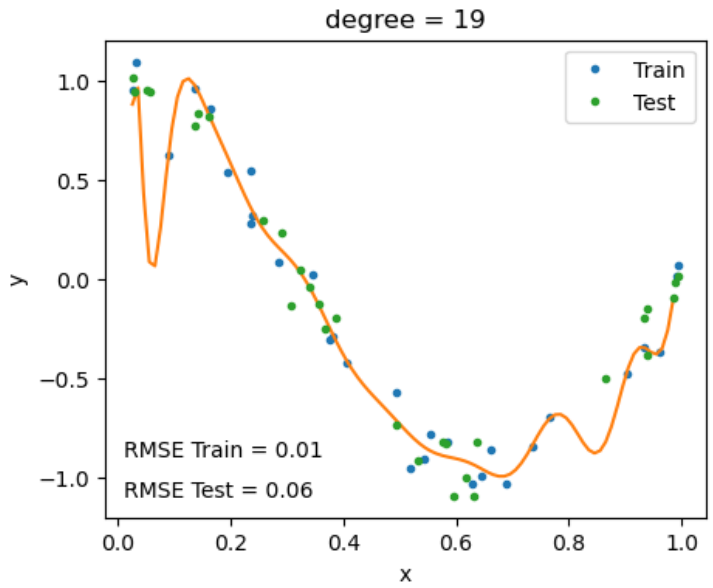
- Under-fitting & Over-fitting



Under-fitting



Well-fitting



Over-fitting

Outline

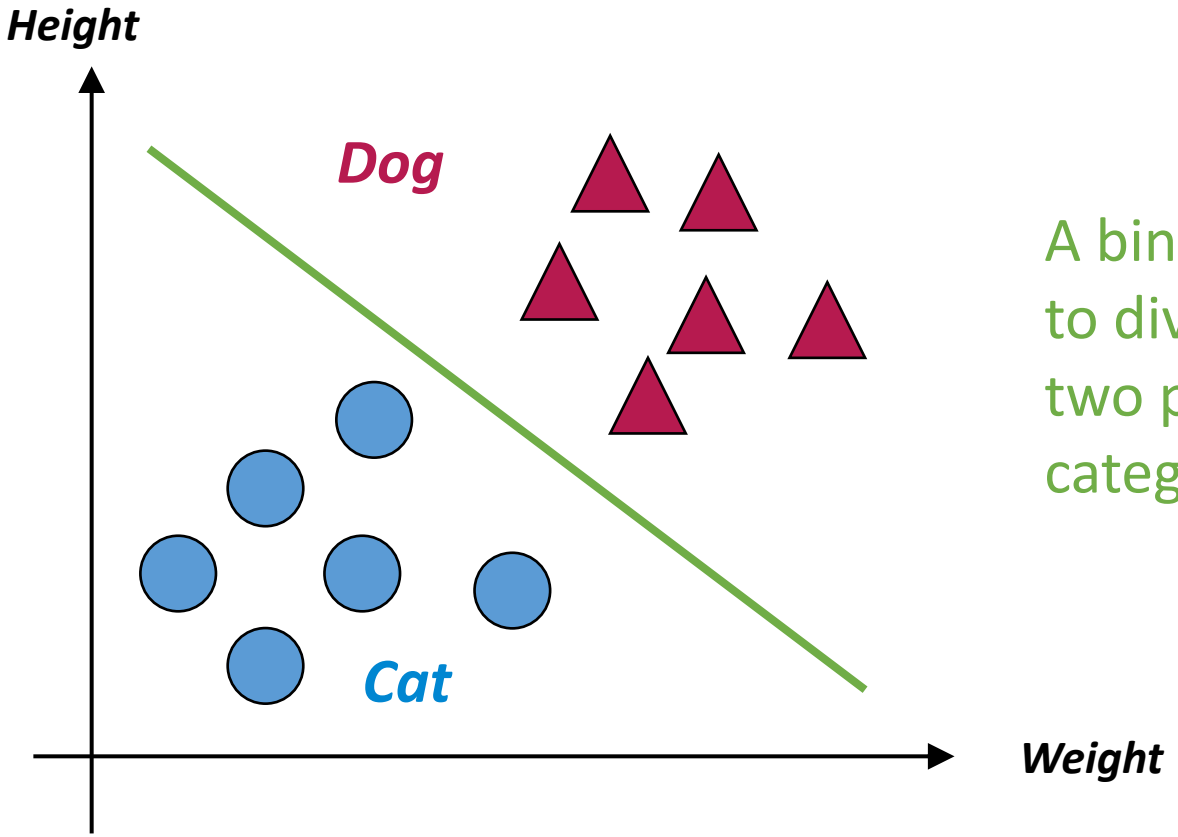
- **Introduction to Classification**
- Logistic Regression
- Evaluation Metrics for Classification

Introduction to Classification

- In statistics
  - Classification is the problem of identifying which of a set of categories an observation belongs to.
- In machine learning
  - Supervised learning algorithms that predict which category the input instance of data belongs to.
  - Three types of classification tasks
    - Binary classification
    - Multi-class classification
    - Multi-label classification

Introduction to Classification

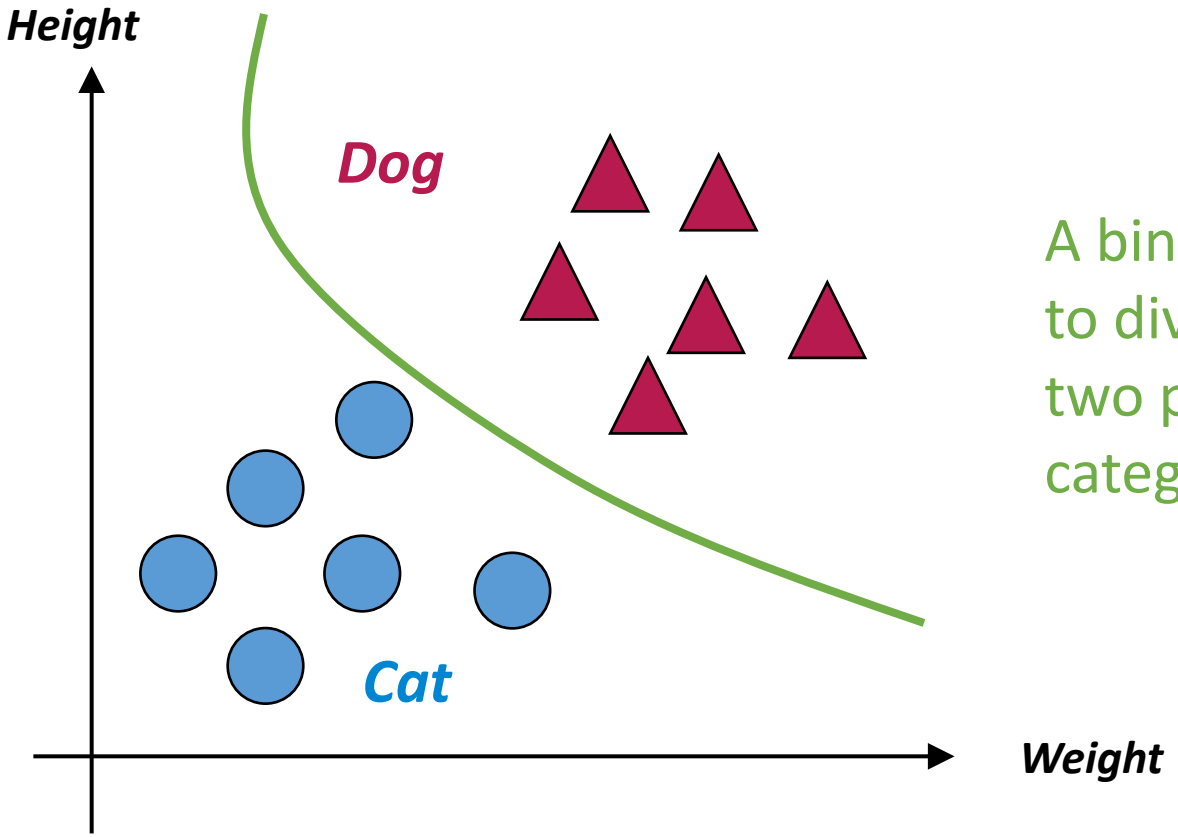
- Binary classification



A binary classifier learned a way to divide the feature space into two parts, each represents a category.

Introduction to Classification

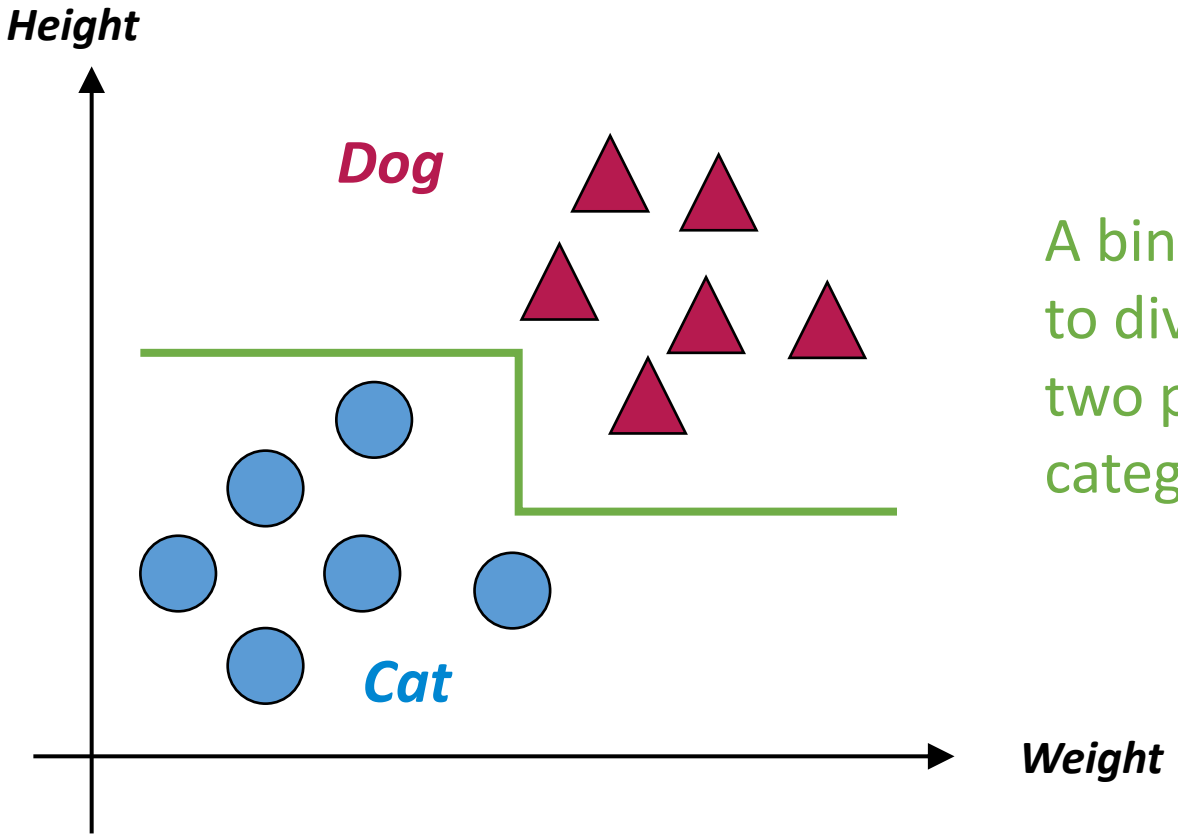
- Binary classification



A binary classifier learned a way to divide the feature space into two parts, each represents a category.

Introduction to Classification

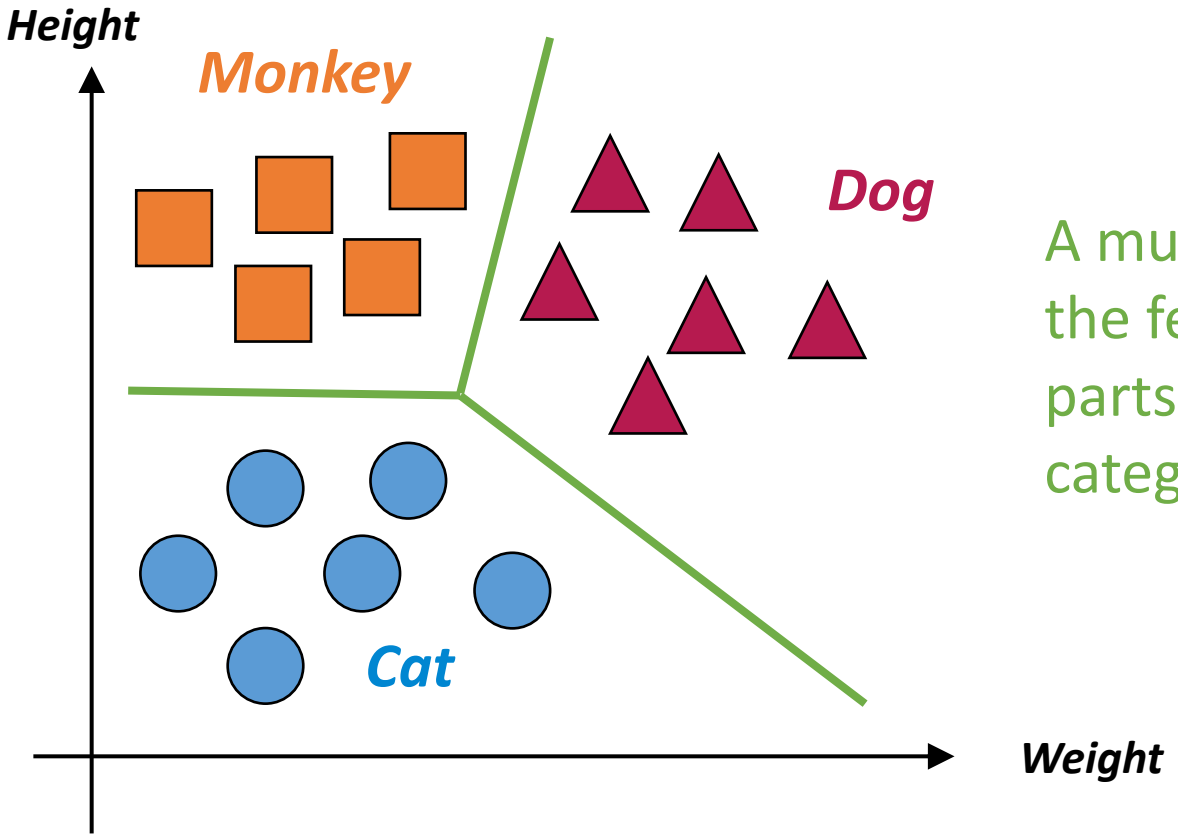
- Binary classification



A binary classifier learned a way to divide the feature space into two parts, each represents a category.

Introduction to Classification

- Multi-class classification



A multi-class classifier divided the feature space into several parts, each represents a category.

Introduction to Classification

- Multi-class classification
  - Some classification algorithms can directly make multi-class predictions
    - 0 → Monkey
    - 1 → Cat
    - 2 → Dog
  - Some can only make binary predictions
    - Solve multi-class problem by training multiple binary classifiers
      - One vs One strategy
        - Binary classifier 1: Monkey vs Cat
        - Binary classifier 2: Monkey vs Dog
        - Binary classifier 3: Cat vs Dog

Introduction to Classification

- Multi-class classification
  - Some classification algorithms can directly make multi-class predictions
    - 0 → Monkey
    - 1 → Cat
    - 2 → Dog
  - Some can only make binary predictions
    - Solve multi-class problem by training multiple binary classifiers
      - One vs Rest strategy
        - Binary classifier 1: Monkey vs Not monkey (cat or dog)
        - Binary classifier 2: Cat vs Not cat (monkey or dog)
        - Binary classifier 3: Dog vs Not dog (monkey or cat)

Introduction to Classification

- Multi-label classification
  - Try to predict zero or more classes for each input example.
  - No mutual exclusion because the input example can have more than one label.

Introduction to Classification

Three Type of Classification Tasks

YAHOO! JAPAN

Binary Classification

- Spam
- Not spam

Multiclass Classification

- Dog
- Cat
- Horse
- Fish
- Bird
- ...

Multi-label Classification

- Dog
- Cat
- Horse
- Fish
- Bird
- ...

KEDGE BUSINESS SCHOOL

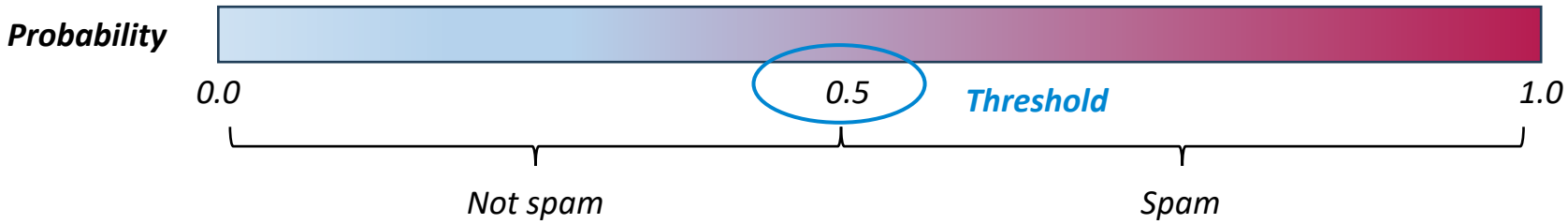
Image source: Source: https://www.microsoft.com/en-us/research/uploads/prod/2017/12/40250.jpg

Introduction to Classification

- The output of a classifier can be:
  - Discrete categories
    - 0 → Not spam email; 1 → Spam email
  - The probability that the input instance belongs to a category
    - The probability that this email is spam is 0.95

Convert predicted probabilities to discrete categories by setting a threshold

- Use a regression model as a classifier
  - Predict the probability of an instance belongs to a category
  - Continuous numerical values → Discrete categories



Outline

- Introduction to Classification
- **Logistic Regression**
- Evaluation Metrics for Classification

Logistic Regression

- Logistic regression model predicts the probability of an example belongs to a category.
  - Probability is a continuous number between 0 and 1
    - 0 → Impossibility
    - 1 → Certainty
- By setting a threshold to the predicted probability, logistic regression model can act as a binary classifier.
- Why not use linear regression?

Logistic Regression

- Why not use linear regression?
  - Input features: Weight and height of the pet
  - Output: The probability that the pet is a cat

$$P_{cat} = -0.1 \times Weight + 2 \times Height$$

The pet is too heavy → $P_{cat} < 0$

The pet is too tall → $P_{cat} > 1$

Logistic Regression

- Why not use linear regression?
  - $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_N x_N$
  - $y \in (-\infty, +\infty)$
  - Can't guarantee the output P is between 0 and 1.
- We could find a function that map the output of linear regression to P

$$y \in (-\infty, +\infty)$$

$$P \in [0, 1]$$

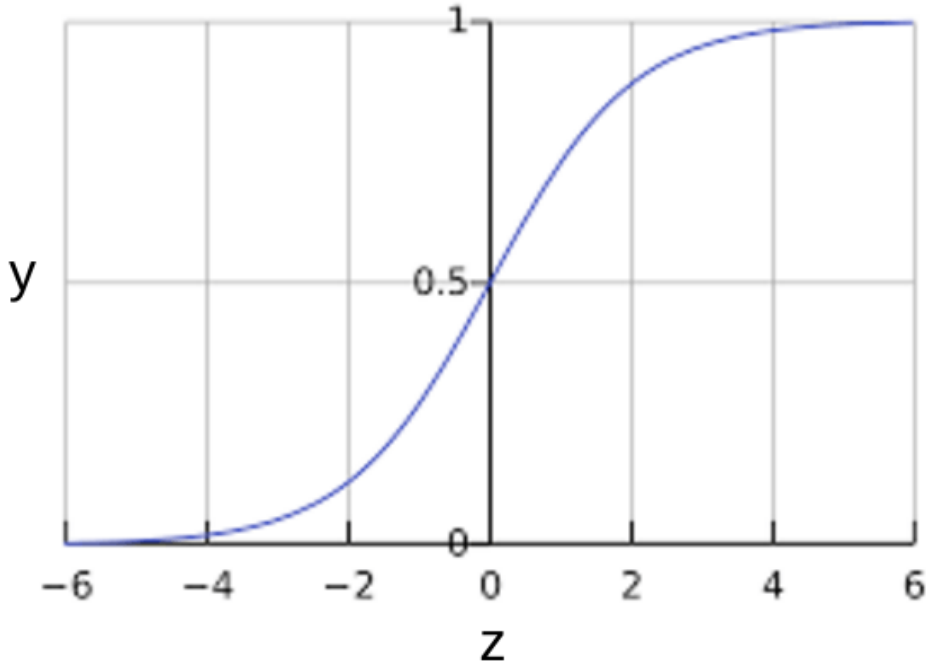
Logistic Regression

- Logistic Regression
  - The sigmoid function

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

  - Map the linear combination of input features to the predicted probability

$$z = \beta_0 + \beta_1 x_1 + \cdots + \beta_N x_N$$



Logistic Regression

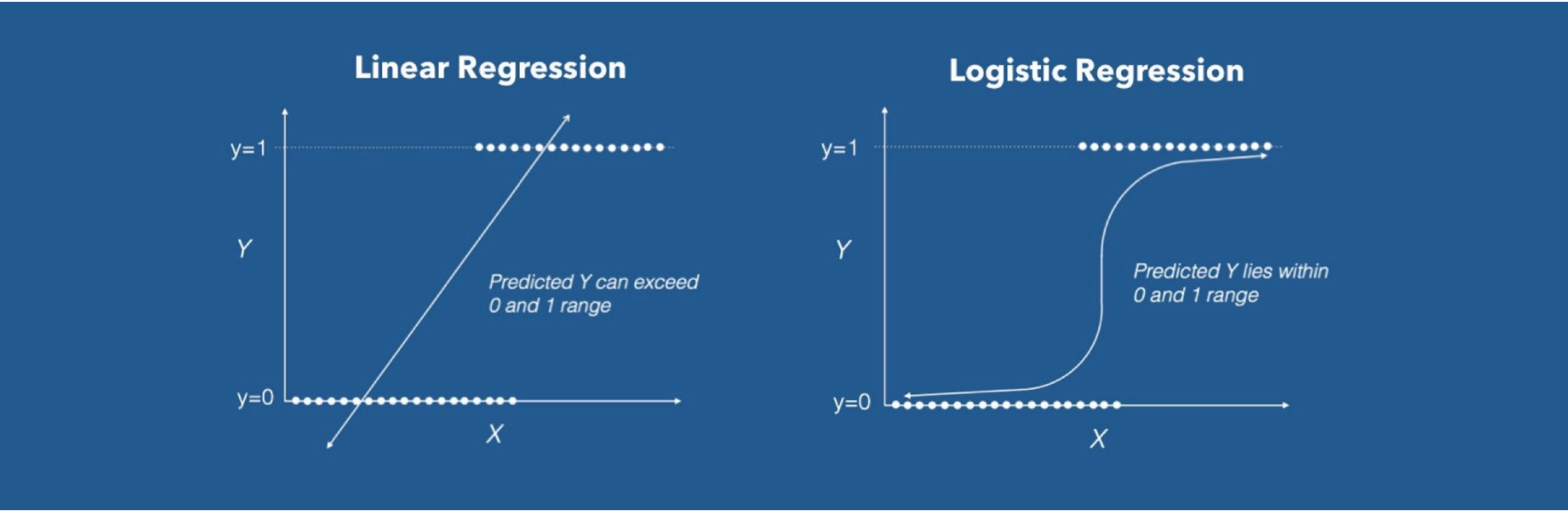


Image source: https://medium.com/@maithilijoshi6/a-comparison-between-linear-and-logistic-regression-8aea40867e2d

Logistic Regression

- Input features → $x_1, x_2, \ldots, x_n$
- Label variable → y is a binary variable (either 0 or 1)

- Model → $y = \dfrac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_N x_N)}}$

- Logistic regression convert the output of a linear regression model to probabilities through a sigmoid function.
  - `sklearn.linear_model.LogisticRegression`

Outline

- Introduction to Classification
- Logistic Regression
- **Evaluation Metrics for Classification**

Evaluation Metrics for Classification

- A classifier predicts:
  - Discrete categories
  - Continuous probability + threshold
- Evaluation goal:
  - Whether the predicted category is correct or not → for a single prediction

Target value	1 → Wolf
Predicted value	0 → No wolf
Correctness	False

Target value	1 → Wolf
Predicted value	1 → Wolf
Correctness	True

  - We need metrics to evaluate the overall performance of classifier

Evaluation Metrics for Classification

- Accuracy
  - The proportion of correct predictions among all predictions the model made
  - The fraction of predictions our model got right

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

- 'Wolf – No wolf' example:
  - 0 → No wolf
  - 1 → Wolf

Target	0	*0*	0	0	0	0	1	0	*1*	0
Predict	0	*1*	0	0	0	0	1	0	*0*	0
Correct	T	*F*	T	T	T	T	T	T	*F*	T

$$\text{Accuracy} = \frac{8}{10} = 0.8$$

Evaluation Metrics for Classification

- Accuracy
  - 'Wolf – No wolf' example:
    - 0 → No wolf
    - 1 → Wolf
  - Limitations:
    - Imbalanced classes
      - 'No wolf' examples counts for 80% in the testing dataset
      - 'Wolf' examples only counts for 20%
      - The goal of this classifier is to detect wolfs and alert accordingly
      - The classifier failed 50% for 'Wolf' examples → not acceptable
    - Accuracy doesn't reflect the actual performance when classes are imbalanced in data

Target	0	*0*	0	0	0	0	1	0	*1*	0
Predict	0	*1*	0	0	0	0	1	0	*0*	0
Correct	T	*F*	T	T	T	T	T	T	*F*	T

$$\text{Accuracy} = \frac{8}{10} = 0.8$$

Not bad

Evaluation Metrics for Classification

- Accuracy
  - 'Wolf – No wolf' example:
    - 0 → No wolf
    - 1 → Wolf
  - Limitations:
    - Neglect the cost of different errors
      - Different errors have different consequences
      - Some errors are more unacceptable than others
    - Accuracy treats all errors equally, ignoring the fact that the cost of different errors might be very different.

Target	0	*0*	0	0	0	0	1	0	*1*	0
Predict	0	*1*	0	0	0	0	1	0	*0*	0
Correct	T	*F*	T	T	T	T	T	T	*F*	T

Fake alert, money lost

No alert, wolf attacked people

Evaluation Metrics for Classification

- Accuracy
  - 'Wolf – No wolf' example:
    - 0 → No wolf
    - 1 → Wolf

Target	0	*0*	0	0	0	0	1	0	*1*	0
Predict	0	*1*	0	0	0	0	1	0	*0*	0
Correct	T	*F*	T	T	T	T	T	T	*F*	T

  - Limitations:
    - Accuracy doesn't tell the full story of the performance of a classifier.
      - How well the classifier performs on each class separately?
      - What is the distribution (type and amount) of errors?

Evaluation Metrics for Classification

- Confusion matrix
  - 'Wolf – No wolf' example:
    - 0 → No wolf → Negative class
    - 1 → Wolf → Positive class
  - The confusion matrix is:

Target	0	*0*	0	0	0	0	1	0	*1*	0
Predict	0	*1*	0	0	0	0	1	0	*0*	0
Correct	T	*F*	T	T	T	T	T	T	*F*	T

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	True Negative TN = 7	False Negative FN = 1
Predict 1 Positive class	False Positive FP = 1	True Positive TP = 1

Evaluation Metrics for Classification

- Confusion matrix
  - 'Wolf – No wolf' example:
    - 0 → No wolf → Negative class
    - 1 → Wolf → Positive class
  - The confusion matrix is:

TP means the classifier correctly predict the positive class (Wolf).

Trigger alert

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	True Negative TN = 7	False Negative FN = 1
Predict 1 Positive class	False Positive FP = 1	***True Positive*** ***TP = 1***

Evaluation Metrics for Classification

- Confusion matrix
  - 'Wolf – No wolf' example:
    - 0 → No wolf → Negative class
    - 1 → Wolf → Positive class
  - The confusion matrix is:

TN means the classifier correctly predict the negative class (No wolf).

Nothing to do

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	***True Negative*** ***TN = 7***	False Negative FN = 1
Predict 1 Positive class	False Positive FP = 1	True Positive TP = 1

Evaluation Metrics for Classification

- Confusion matrix
  - 'Wolf – No wolf' example:
    - 0 → No wolf → Negative class
    - 1 → Wolf → Positive class
  - The confusion matrix is:

FP means the classifier incorrectly predict the positive class (Wolf).

Trigger fake alert

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	True Negative TN = 7	False Negative FN = 1
Predict 1 Positive class	***False Positive*** ***FP = 1***	True Positive TP = 1

Evaluation Metrics for Classification

- Confusion matrix
  - 'Wolf – No wolf' example:
    - 0 → No wolf → Negative class
    - 1 → Wolf → Positive class
  - The confusion matrix is:

__FN__ means the classifier incorrectly predict the negative class (No-wolf).

No alert, people got attacked

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	True Negative TN = 7	***False Negative*** ***FN = 1***
Predict 1 Positive class	False Positive FP = 1	True Positive TP = 1

Evaluation Metrics for Classification

- Confusion matrix
  - 'Wolf – No wolf' example:
    - 0 → No wolf → Negative class
    - 1 → Wolf → Positive class
  - The confusion matrix is:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} = \frac{1 + 7}{1 + 7 + 1 + 1} = 0.8$$

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	True Negative TN = 7	False Negative FN = 1
Predict 1 Positive class	False Positive FP = 1	True Positive TP = 1

Evaluation Metrics for Classification

- Confusion matrix

	Target 0 Negative class	Target 1 Positive class
Predict 0 Negative class	True Negative TN = 7	False Negative FN = 1
Predict 1 Positive class	False Positive FP = 1	True Positive TP = 1

- Precision
  - The ratio of TP to all positive predictions

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{1}{1+1} = 0.5$$

- Recall
  - The ratio of TP to all actual positives

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{1}{1+1} = 0.5$$

Evaluation Metrics for Classification

Precision

- Of all the situations predicted as ‘Wolf’, how many were actually ‘Wolf’?
- When the classifier predicts a wolf, it is correct 50% of the time
- High precision indicates a low rate of fake alerts (i.e. a low number of False Positive)

• Recall

- Of all the situations there is actually ‘Wolf’, how many were correctly detected?
- The classifier correctly identifies 50% of all ‘Wolf’ situations.
- High recall means a classifier correctly identified most of the wolves, but some situations without wolf might be misclassified, resulting in fake alerts. (the number of False Positive might by high)

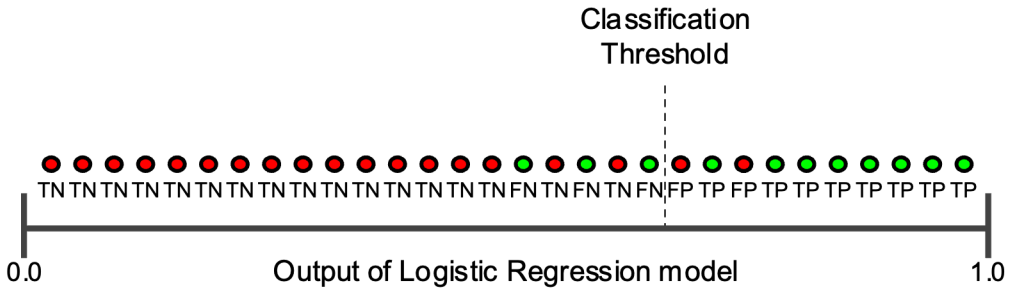
Evaluation Metrics for Classification

- Precision vs Recall
  - A classifier for detecting spam emails
    - 0 → Not spam → Negative class
    - 1 → Spam → Positive class

    - What does Precision = 0.6 mean? → 40% of emails in the spam folder are actually normal emails.
    - What does Recall = 0.8 mean? → 20% of spam emails are not detected.
    - Which do we prefer?
      - High Precision & Low Recall → Most of spam emails won't be detected.
      - Low Precision & High Recall → A lot of normal emails are put in spam folder by mistake.

Evaluation Metrics for Classification

- Precision vs Recall
  - The ideal classifier has both high precision and recall
  - But improving precision and improving recall are often in conflicts
  - We need to make a trade-off between precision and recall



Actually not spam
Actually spam

TN = 17	FN = 3
FP = 2	TP = 8

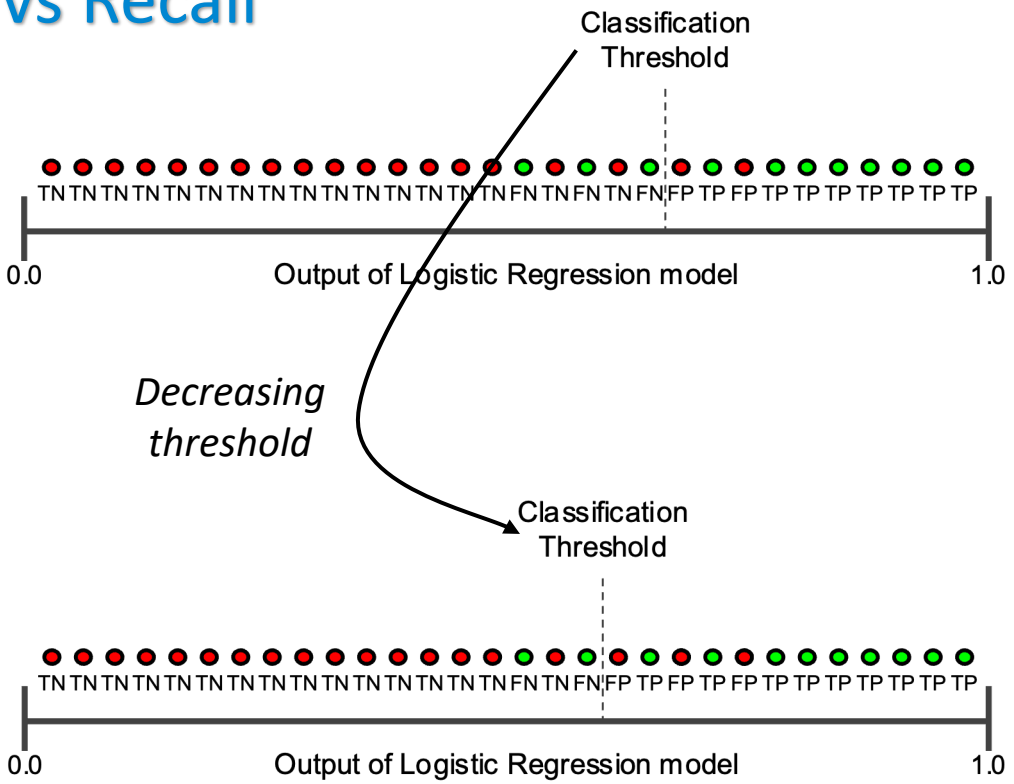
$$\text{Precision} = \frac{8}{8+2} = 0.8$$

$$\text{Recall} = \frac{8}{8+3} = 0.73$$

Image source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

Evaluation Metrics for Classification

- Precision vs Recall



TN = 17	FN = 3
FP = 2	TP = 8

$\text{Precision} = 0.8$

$\text{Recall} = 0.73$

TN = 16	FN = 3
FP = 2	TP = 9

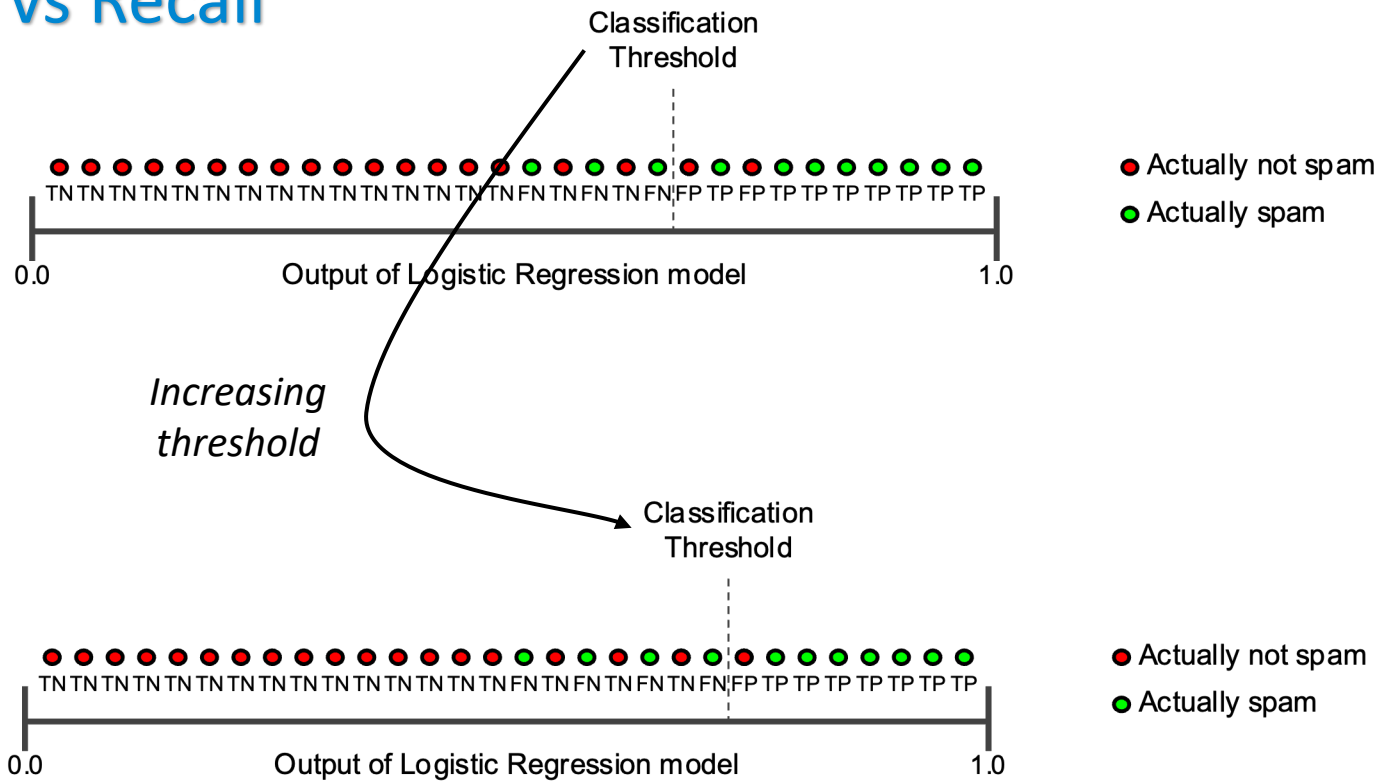
$\text{Precision} = 0.75 \searrow$

$\text{Recall} = 0.82 \nearrow$

Image source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

Evaluation Metrics for Classification

- Precision vs Recall



TN = 17	FN = 3
FP = 2	TP = 8

Precision = 0.8
Recall = 0.73

TN = 18	FN = 4
FP = 1	TP = 7

Precision = 0.88 ↗
Recall = 0.64 ↘

Image source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

Evaluation Metrics for Classification

- F1-score
  - A balance evaluation between precision and recall
  - The harmonic mean of precision and recall
  - $\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
  - Best being 1
    - Precision = 1 and Recall = 1
  - Worst being 0
    - Precision = 0 or Recall = 0
  - Better metric than accuracy when
    - Imbalanced classes
    - Different costs for different errors

Evaluation Metrics for Classification

- Changing the classification threshold will change the performance of the classifier:
  - Confusion Matrix
  - Precision / Recall
  - Accuracy / F1-score
- If the performance is poor, it may not be the problem with the classifier, but with an unreasonable threshold.
- How to evaluate the classifier's performance regardless the threshold?

Evaluation Metrics for Classification

- ROC curve (Receiver Operating Characteristic curve)
  - A curve showing the performance of a classifier at all thresholds
  - Each threshold results in a confusion matrix
    - True Positive Rate (TPR) = TP / (TP + FN)
    - False Positive Rate (FPR) = FP / (FP + TN)
  - An ROC curve plots TPR vs. FPR at different thresholds

TN	FN
FP	TP

Evaluation Metrics for Classification

- ROC curve
  - An ROC curve plots TPR vs. FPR at different thresholds
  - TPR = TP / (TP + FN)
  - FPR = FP / (FP + TN)
- The perfect classifier
  - TPR = 1
  - FPR = 0

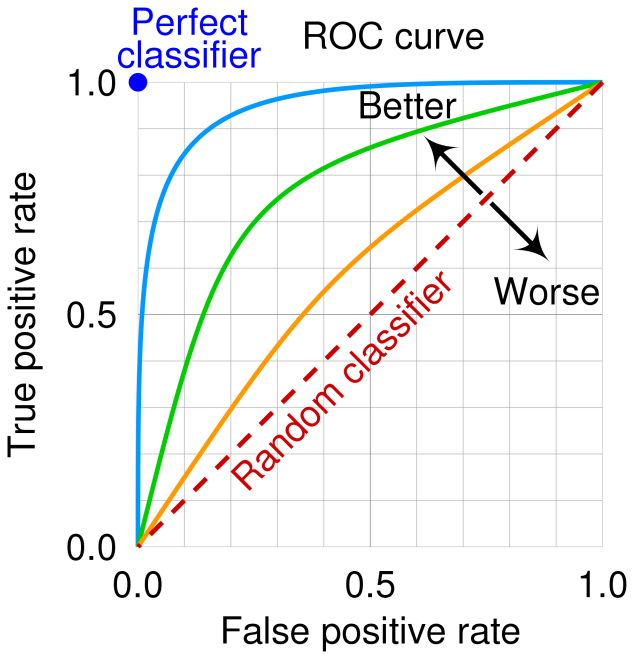


Image source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Evaluation Metrics for Classification

- ROC curve
  - An ROC curve plots TPR vs. FPR at different thresholds
  - TPR = TP / (TP + FN)
  - FPR = FP / (FP + TN)
- The random classifier
  - 50% - 50% chance
    - At any threshold, the chances of a positive prediction being true (TP) or false (FP) are the same.
    - The TPR and FPR will increase at the same rate.
  - A random classifier will locate at the line TPR = FPR

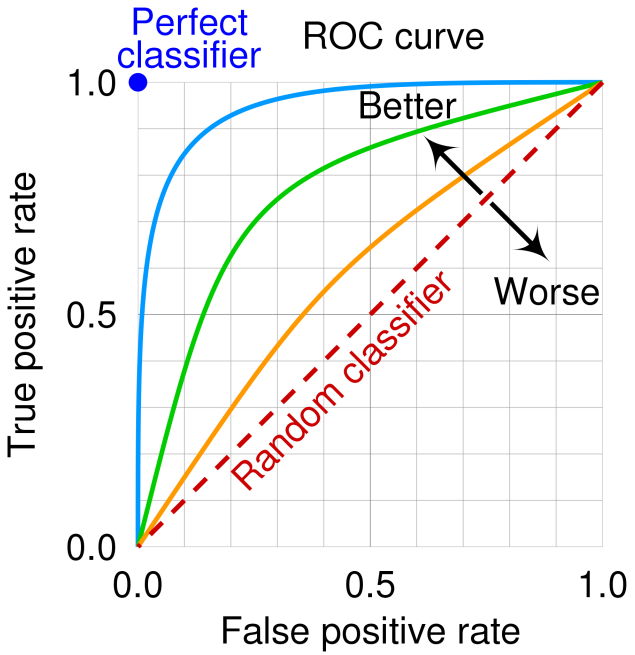


Image source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Evaluation Metrics for Classification

- ROC curve
  - An ROC curve plots TPR vs. FPR at different thresholds
  - TPR = TP / (TP + FN)
  - FPR = FP / (FP + TN)
- The closer a point (representing a classifier and a threshold) is to the upper left corner, the better the performance.
- A classifier with the ROC curve close to the upper left corner means we can find a threshold to get almost perfect predictions.

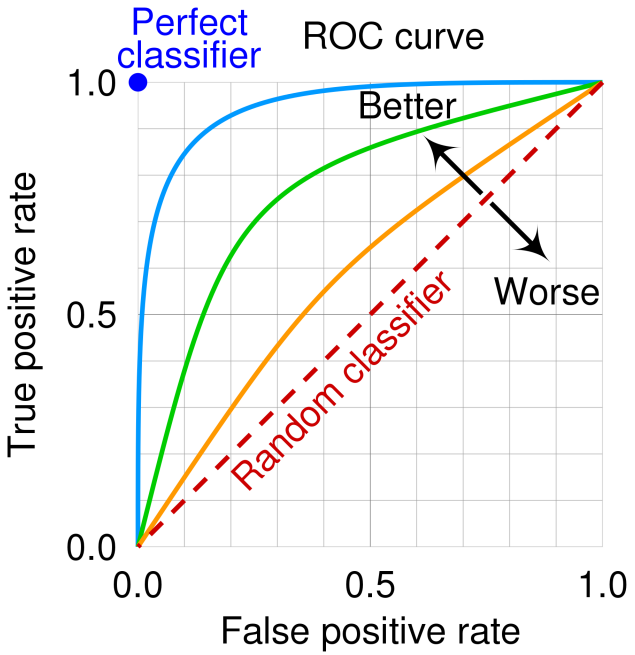
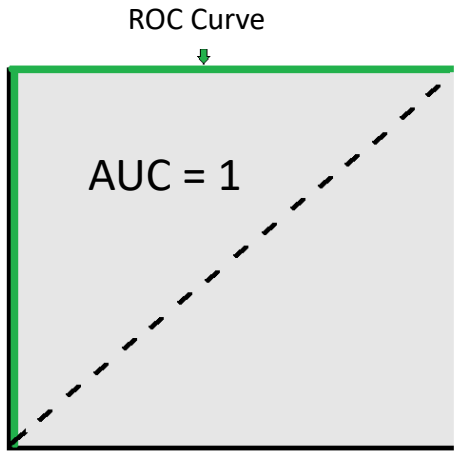


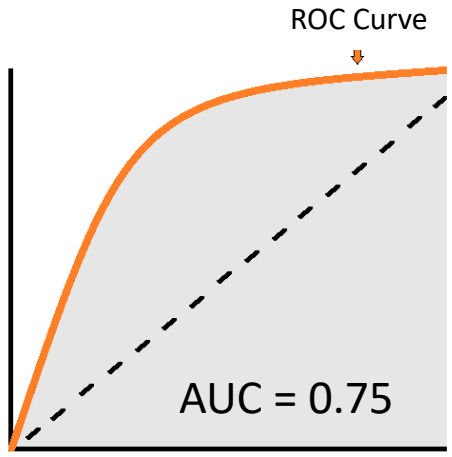
Image source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Evaluation Metrics for Classification

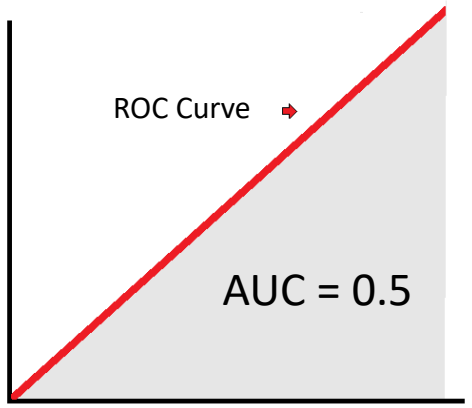
- AUC (Area under the ROC curve)
  - Range from 0 to 1



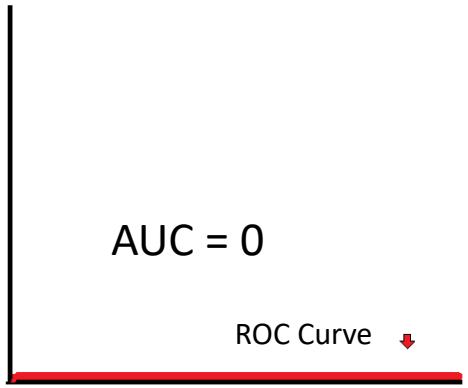
A classifier which is always correct, regardless the threshold.



A classifier which learned something and can predict better than a random one



A classifier which gives random predictions, 50% correct and 50% wrong



A classifier which is always wrong, regardless the threshold.

Image source: https://machine-learning.paperspace.com/wiki/auc-area-under-the-roc-curve

Hands-on Exercise

- Exercise 03 Logistic Regression