ARTIFICIAL INTELLIGENCE NEEDS REAL INTELLIGENCE

Classification II

Professor: Tianyuan ZHANG
tianyuan.zhang@kedgebs.com

KEDGE BUSINESS SCHOOL

Recap of Previous Session

- Classification
  - Supervised learning that predict which category the input data belongs to
  - Three types of classification tasks

Binary Classification

- Spam
- Not spam

Multiclass Classification

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

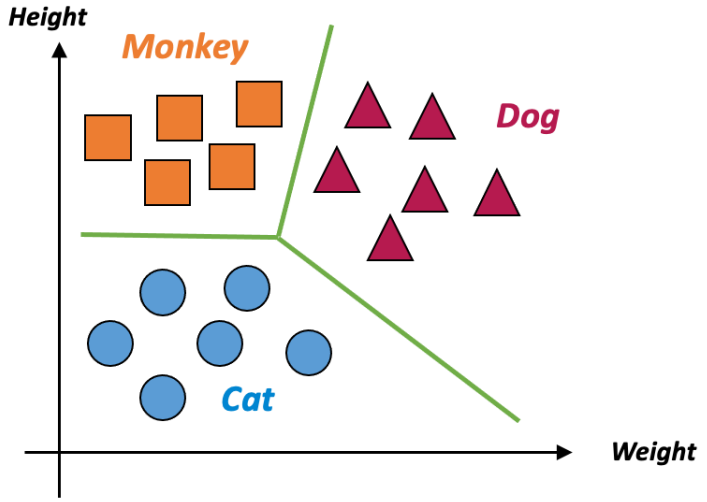
Multi-label Classification

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

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

Recap of Previous Session

- Classification
  - A classifier learned a way to divided the feature space into different parts
  - Each part represents a category
  - A classifier outputs:
    - Discrete categories
      - 0 → Not spam email
      - 1 → Spam email
    - The probability that the input belongs to a category
      - The probability that this email is spam is 0.95



Convert probabilities to categories by setting thresholds

Recap of Previous Session

- Logistic Regression
  - $y = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_N x_N)}}$
  - Mapping the linear combination of input features to the predicted probability through the sigmoid function
  - Predict the probability of an example belongs to a category
  - Guarantee the predicted value lies within 0 and 1 range

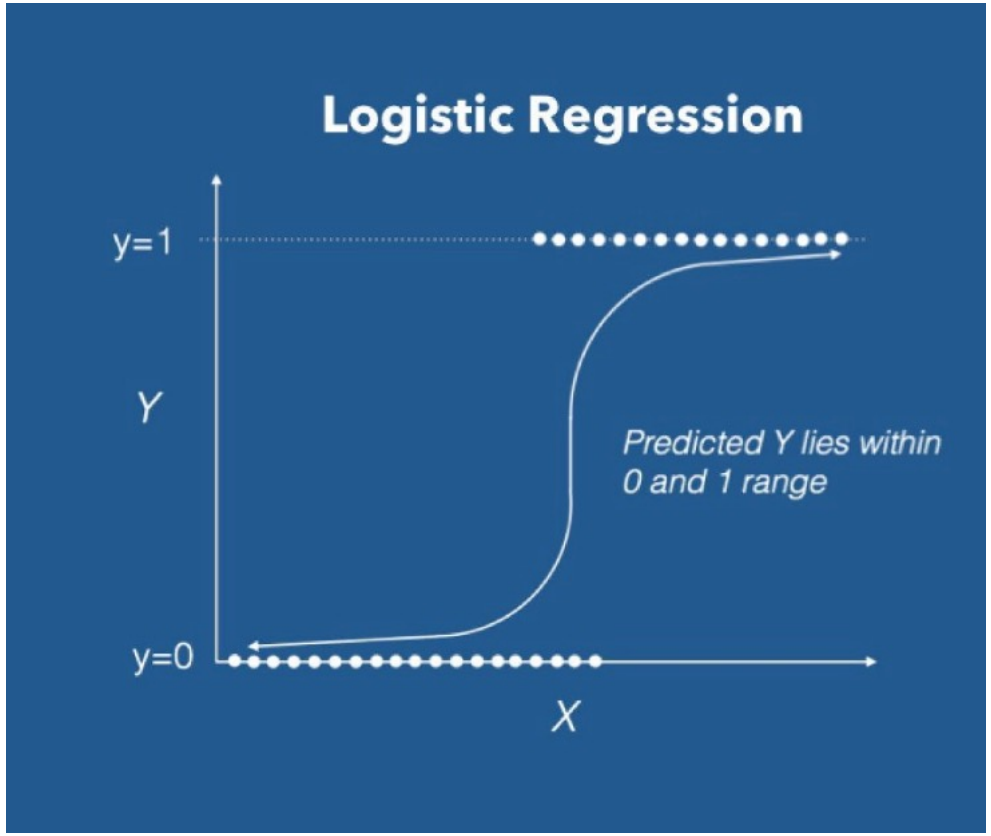


Image source: https://medium.com/@maithilijoshi6/a-comparison-between-linear-and-logistic-regression-8aea40867e2d

Recap of Previous Session

- Evaluation metrics for classification
  - Confusion matrix
    - 0 → Negative class
    - 1 → Positive class

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

Recap of Previous Session

- Evaluation metrics for classification
  - Confusion matrix

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$
- $\text{Precision} = \frac{TP}{TP+FP}$
- $\text{Recall} = \frac{TP}{TP+FN}$
- $\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

Proportion of true predictions to all predictions.

Recap of Previous Session

- Evaluation metrics for classification
  - Confusion matrix

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$
- $\text{Precision} = \frac{TP}{TP+FP}$
- $\text{Recall} = \frac{TP}{TP+FN}$
- $\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

Proportion of true predictions to all predictions.

Proportion of true ***positive*** predictions to all ***positive*** predictions.

What is the probability of being correct for a positive prediction?

Recap of Previous Session

- Evaluation metrics for classification
  - Confusion matrix

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$ — Proportion of true predictions to all predictions.
- $\text{Precision} = \frac{TP}{TP+FP}$ — Proportion of true positive predictions to all positive predictions.
- $\text{Recall} = \frac{TP}{TP+FN}$ — Proportion of true positive predictions to all ***positive targets***.
- $\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

What is the probability that a positive target is recognized?

Recap of Previous Session

- Evaluation metrics for classification
  - Confusion matrix

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- $\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$ — Proportion of true predictions to all predictions.
- $\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}$ — Proportion of true positive predictions to all positive predictions.
- $\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$ — Proportion of true positive predictions to all positive targets.
- $\text{F1-score} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$ — A ***hormonic mean*** of precision and recall, balancing the two.

Recap of Previous Session

- Evaluation metrics for classification
  - Confusion matrix

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$
- $\text{Precision} = \frac{TP}{TP+FP}$
- $\text{Recall} = \frac{TP}{TP+FN}$
- $\text{F1-score} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$

- *Changing classification threshold will change the confusion matrix:*
  - *Change accuracy, precision, recall, and F1-score*
  - *Can't improve both precision & recall at the same time*
  - *Trade-off between precision & recall based on needs*

Recap of Previous Session

- Evaluation metrics for classification
  - Metrics regardless threshold
    - ROC curve
      - Receiver Operating Characteristic curve
      - TPR vs. FPR at different thresholds
      - TPR = TP / (TP + FN)
      - FPR = FP / (FP + TN)
    - AUC
      - Area under the ROC curve
      - Range from 0 to 1
      - Higher than 0.5 → Better than a random classifier

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

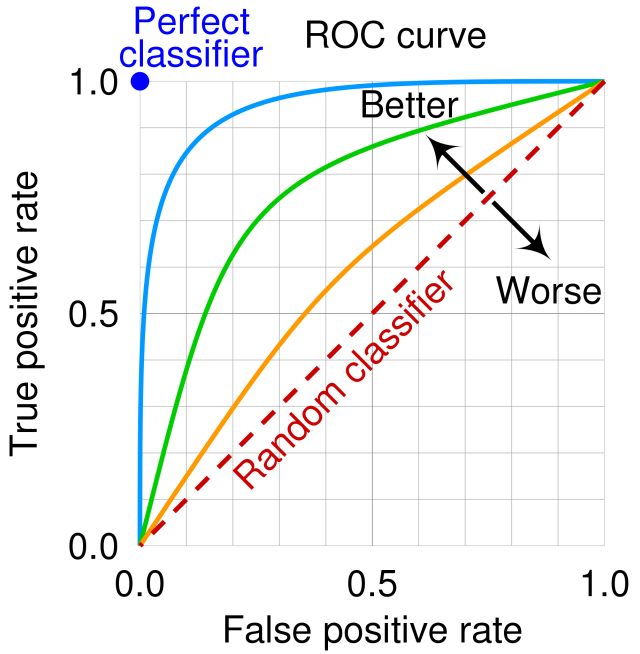


Image source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Outline

- **Metrics for Multi-class Classification**
- K-Nearest Neighbors
- Support Vector Machine

Metrics for Multi-class Classification

- Evaluation metrics
  - $\text{Precision} = \frac{TP}{TP+FP}$
  - $\text{Recall} = \frac{TP}{TP+FN}$
  - $\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- *Except accuracy, other metrics we've seen are derived form the confusion matrix:*
  - *Extend confusion matrix to multi-class*
  - *The equation for each metric remains the same*

Metrics for Multi-class Classification

- Evaluation metrics
  - $\text{Precision} = \frac{TP}{TP+FP}$
  - $\text{Recall} = \frac{TP}{TP+FN}$
  - $\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- *For binary classification*
  - *Class 1 → Positive class*
  - *Class 0 → Negative class*
- *The confusion matrix respect to this assumption*
  - *Metrics is calculated based on this assumption*

Metrics for Multi-class Classification

- Evaluation metrics
  - $\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$
  - $\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$
  - $\text{F1-score} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision+Recall}}$

	Target 0	Target 1
Predict 0	TN	FN
Predict 1	FP	TP

- *For binary classification*
  - *Class 1 → Positive class*
  - *Class 0 → Negative class*
- *For multi-class classification*
  - *Select a class → Positive class*
  - *All other classes → Negative class*

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Select a class → Positive class
  - All other classes → Negative class

➡

- **Monkey** → Positive class
- **Cat & Dog** → Negative class

	Target Cat	Target Dog	Target Monkey
Predict Cat	3	1	1
Predict Dog	1	4	2
Predict Monkey	2	1	5

➡

	Target Not Monkey	Target Monkey
Predict Not Monkey	TN 9	FN 3
Predict Monkey	FP 3	TP 5

Precision = 0.625 *Recall = 0.625*

F1-score = 0.625

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Select a class → Positive class
  - All other classes → Negative class

➡

- **Dog** → Positive class
- **Cat & Monkey** → Negative class

	Target Cat	Target Dog	Target Monkey
Predict Cat	3	1	1
Predict Dog	1	4	2
Predict Monkey	2	1	5

➡

	Target Not Dog	Target Dog
Predict Not Dog	TN 11	FN 2
Predict Dog	FP 3	TP 4

Precision = 0.57 *Recall = 0.67*

F1-score = 0.615

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Select a class → Positive class
  - All other classes → Negative class

➡

- **Cat** → Positive class
- **Dog & Monkey** → Negative class

	Target Cat	Target Dog	Target Monkey
Predict Cat	3	1	1
Predict Dog	1	4	2
Predict Monkey	2	1	5

➡

	Target Not Cat	Target Cat
Predict Not Cat	TN 12	FN 3
Predict Cat	FP 2	TP 3

Precision = 0.6 *Recall = 0.5*

F1-score = 0.545

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Precision, recall, and F1-score are class-wise metrics in multi-class case

	Target Cat	Target Dog	Target Monkey
Predict Cat	3	1	1
Predict Dog	1	4	2
Predict Monkey	2	1	5

	Precision	Recall	F1-socre	Support
Cat	0.6	0.5	0.545	6
Dog	0.57	0.67	0.615	6
Monkey	0.625	0.625	0.625	8

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Precision, recall, and F1-score are class-wise metrics in multi-class case
  - **Support** is the number of actual occurrences of the class in the dataset
    - How many instances of the specific class are tested to support the class-wise metrics
    - Compare the support values of difference classes to identify imbalanced classes

	Target Cat	Target Dog	Target Monkey
Predict Cat	*3*	1	1
Predict Dog	*1*	4	2
Predict Monkey	*2*	1	5

	Precision	Recall	F1-socre	*Support*
Cat	0.6	0.5	0.545	*6*
Dog	0.57	0.67	0.615	6
Monkey	0.625	0.625	0.625	8

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Precision, recall, and F1-score are class-wise metrics in multi-class case
  - Support is the number of actual occurrences of the class in the dataset
  - Overall performance metrics can be represented by the average of class-wise metrics:
    - Macro Average: unweighted

	Precision	Recall	F1-socre	Support
Cat	0.6	0.5	0.545	6
Dog	0.57	0.67	0.615	6
Monkey	0.625	0.625	0.625	8
Macro Average	0.599	0.597	0.595	20

$$\text{Precision}_{\text{Macro}} = \frac{\text{Precision}_{\text{Cat}} + \text{Precision}_{\text{Dog}} + \text{Precision}_{\text{Monkey}}}{3} = 0.599$$

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Precision, recall, and F1-score are class-wise metrics in multi-class case
  - Support is the number of actual occurrences of the class in the dataset
  - Overall performance metrics can be represented by the average of class-wise metrics:
    - Macro Average: unweighted
    - Weighted Average: use support as weight

	Precision	Recall	F1-socre	Support
Cat	0.6	0.5	0.545	6
Dog	0.57	0.67	0.615	6
Monkey	0.625	0.625	0.625	8
Macro Average	0.599	0.597	0.595	20
Weighted Average	0.601	0.6	0.598	20

$$\text{Recall}_{\text{Weighted}} = \frac{\text{Recall}_{\text{Cat}} \times \text{Support}_{\text{Cat}} + \text{Recall}_{\text{Dog}} \times \text{Support}_{\text{Dog}} + \text{Recall}_{\text{Monkey}} \times \text{Support}_{\text{Monkey}}}{\text{Support}_{\text{Cat}} + \text{Support}_{\text{Dog}} + \text{Support}_{\text{Monkey}}} = 0.6$$

Metrics for Multi-class Classification

- Confusion matrix in multi-class case
  - Accuracy is not class-wise metric
    - Respect to the definition: The proportion of correct predictions to all predictions made

	Target Cat	**Target** Dog	**Target** Monkey
Predict Cat	3	1	1
Predict Dog	1	4	2
Predict Monkey	2	1	5

➡

- *Only the predictions on the diagonal are correct*
- *All other predictions are wrong*
- *Accuracy = 12 / 20 = 0.6*
- *There is only one accuracy for a confusion matrix*

Metrics for Multi-class Classification

- ROC curve & AUC in multi-class case
  - Convert multi-class classification problems to several binary classification problems
    - One-vs-Rest approach
    - ~~One-vs-One approach (less intuitive)~~
  - Compute class-wise ROC curve using One-vs-Rest confusion matrix

	Target Cat	Target Dog	Target Monkey
Predict Cat	3	1	1
Predict Dog	1	4	2
Predict Monkey	2	1	5

	Target Not Monkey	Target Monkey
Predict Not Monkey	TN 9	FN 3
Predict Monkey	FP 3	TP 5

	Target Not Dog	Target Dog
Predict Not Dog	TN 11	FN 2
Predict Dog	FP 3	TP 4

	Target Not Cat	Target Cat
Predict Not Cat	TN 12	FN 3
Predict Cat	FP 2	TP 3

Metrics for Multi-class Classification

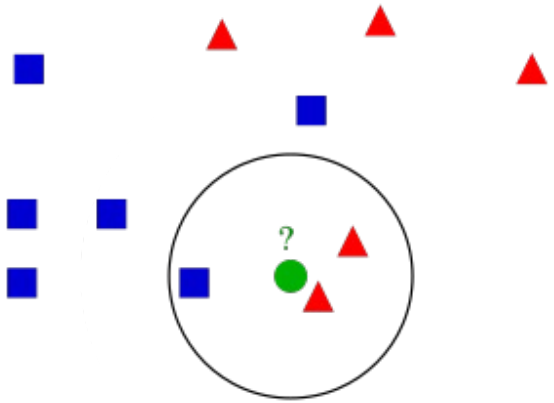
- ROC curve & AUC in multi-class case
  - One-vs-Rest class-wise ROC curve
    - `sklearn.metrics.roc_curve` is restricted to the binary classification task
    - Use `sklearn.metrics.RocCurveDisplay` instead.
  - One-vs-Rest class-wise AUC
    - `sklearn.metrics.roc_curve_score` can by used with multi-class classification
  - Find example codes in the solution of Exercise 04 - Challenge

Outline

- Metrics for multi-class classification
- **K-Nearest Neighbors**
- Support Vector Machine

K-Nearest Neighbors

- KNN
  - One of the simplest algorithm can be used to solve classification tasks
  - Classify a data points based on the classes of its nearest neighbors
  - K represents the number of nearest neighbors to consider



Class A *Class B* *To be classified*

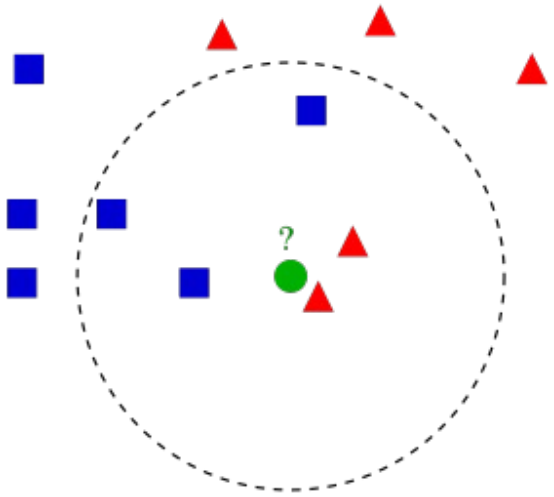
*If **K = 3**, consider 3 nearest neighbors:*

- *One is class A*
- *Two are class B*

The prediction is class B

K-Nearest Neighbors

- KNN
  - One of the simplest classification algorithm
  - Classify a data points based on the classes of its nearest neighbors
  - K represents the number of nearest neighbors to consider



Class A *Class B* *To be classified*

*If **K = 5**, consider 5 nearest neighbors:*

- *Three are class A*
- *Two are class B*

The prediction is class A

K-Nearest Neighbors

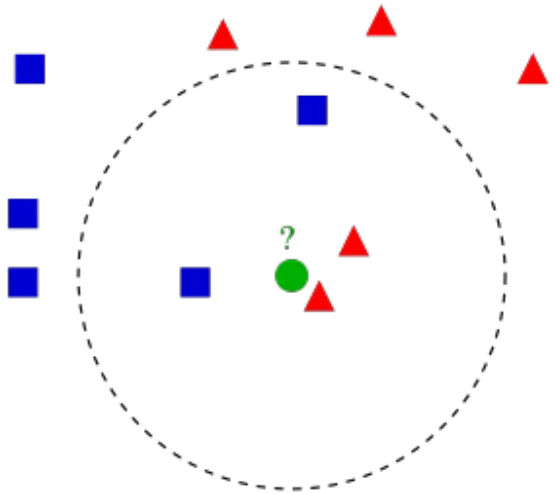
- KNN
  - Non-parametric
    - There is no parameter such as slope or intercept in a KNN model
  - Instance-based
    - Store all instances in the training dataset and make predictions based on that
  - A voting system
    - Each nearest neighbor votes on the class of the new data point
    - The prediction is made by the voting results of the nearest neighbors

K-Nearest Neighbors

- KNN → A voting system
  - How many voters in the system? → Determine the value of K
  - How to identify the voters? → Identify NNs by distance
  - How to compute the voting result? → Generate prediction

K-Nearest Neighbors

- KNN → A voting system
  - How many voters in the system? → Determine the value of K
    - K is a constant specified by us
    - The best choice of K depends on the dataset.
    - Better to choose an odd number to avoid tied votes for binary classification



Class A *Class B* *To be classified*

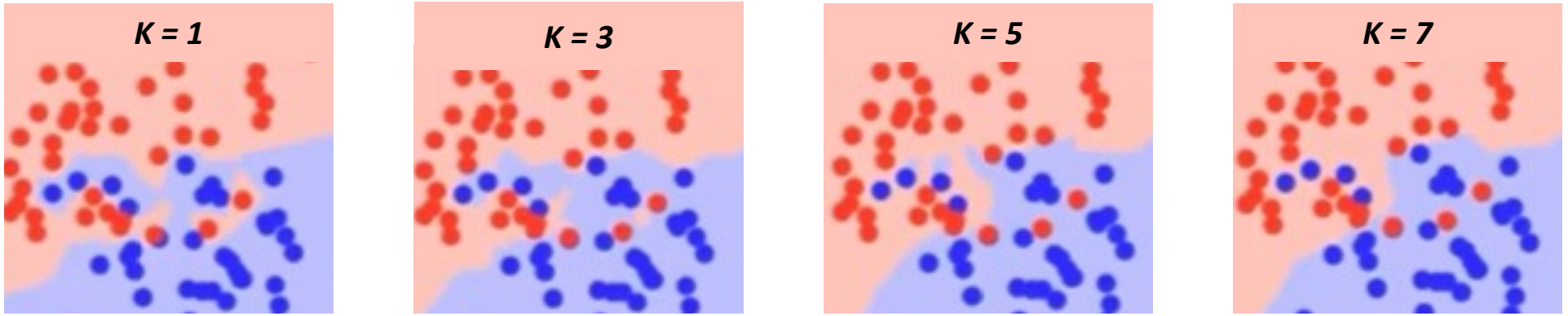
*If **K = 4**, consider 4 nearest neighbors:*

- *Two are class A*
- *Two are class B*

We got tied votes here.

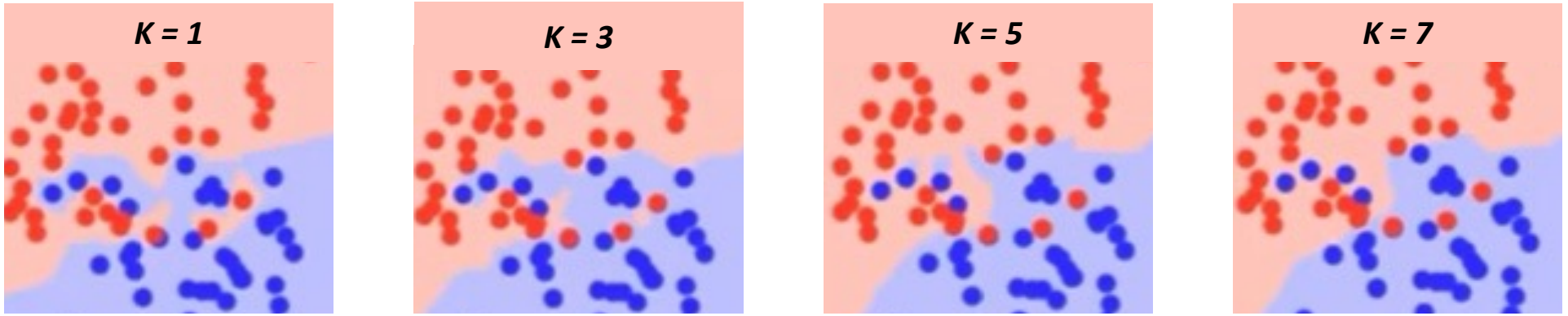
K-Nearest Neighbors

- KNN → A voting system
  - How many voters in the system? → Determine the value of K
    - The effect of the value of K



K-Nearest Neighbors

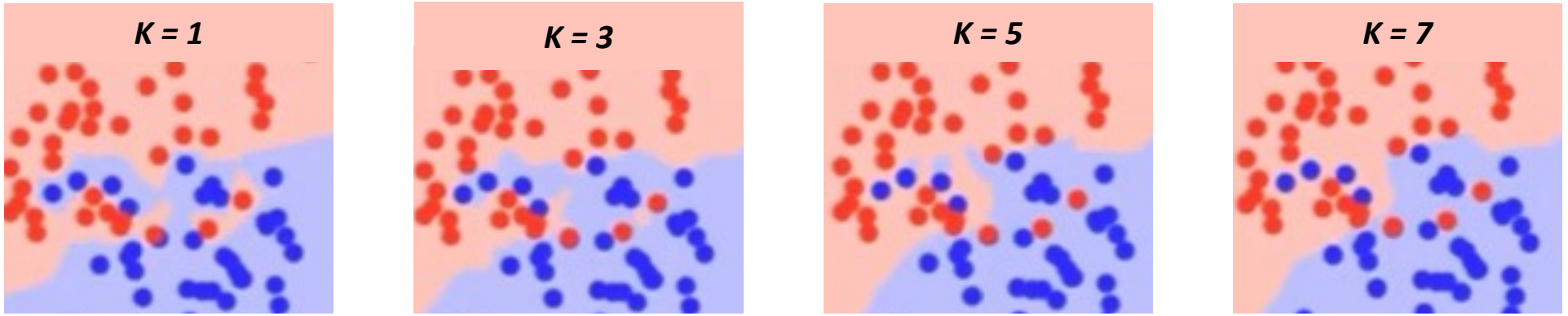
- KNN → A voting system
  - How many voters in the system? → Determine the value of K
    - The effect of the value of K



    - The boundary becomes **smoother (simpler)** when increasing the value of K
    - A large K might smoothen the boundary too much, the model failed to capture the complexity of data → **Under-fitting**

K-Nearest Neighbors

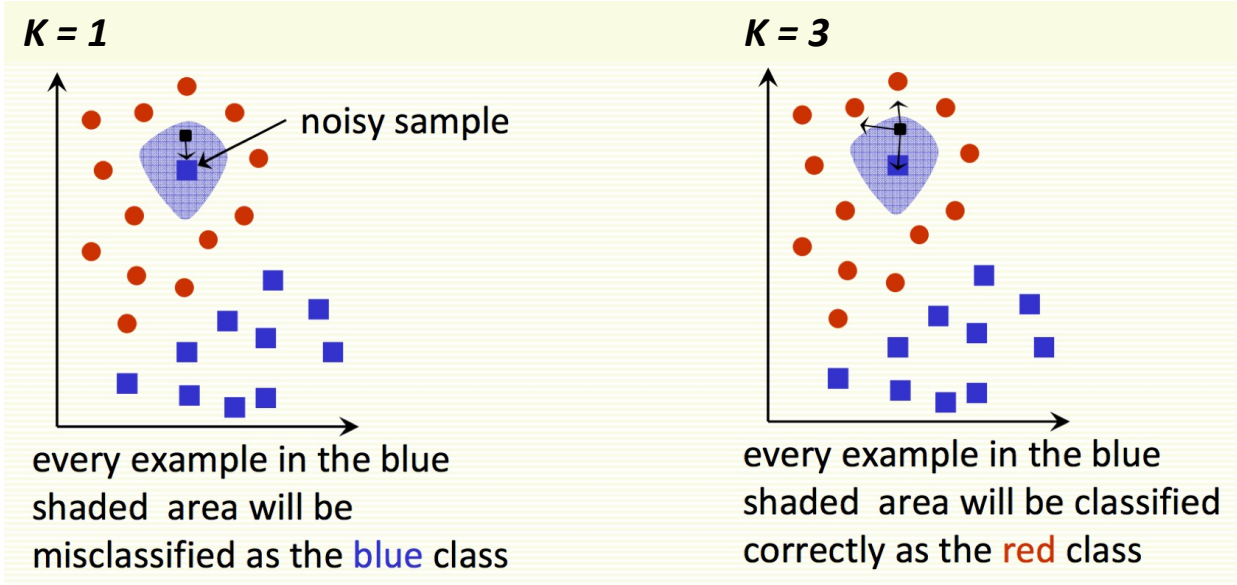
- KNN → A voting system
  - How many voters in the system? → Determine the value of K
    - The effect of the value of K



    - With K increasing to infinity, the figure becomes all blue or all red depending on the majority
    - When $K \to +\infty$, the model will always return the majority class in the training dataset

K-Nearest Neighbors

- KNN → A voting system
  - How many voters in the system? → Determine the value of K



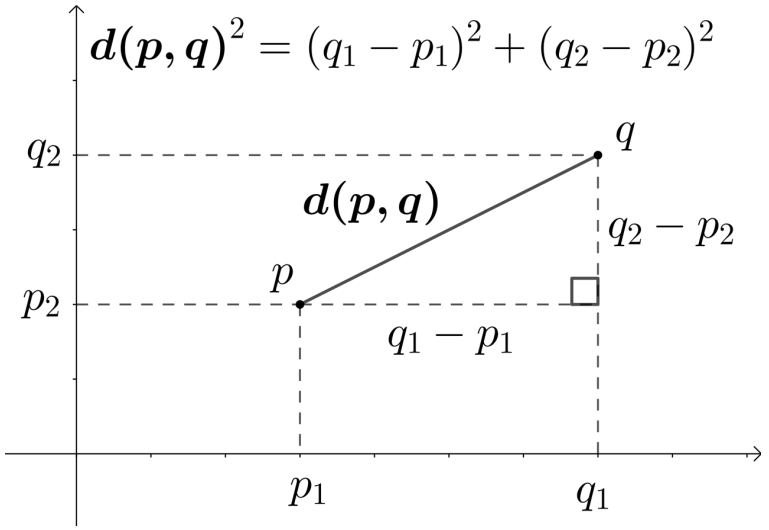
- A small K leads to models that are complex and sensitive to noise → **Over-fitting**

Image source: https://www.cs.toronto.edu/~urtasun/courses/CSC411_Fall16/05_nn.pdf

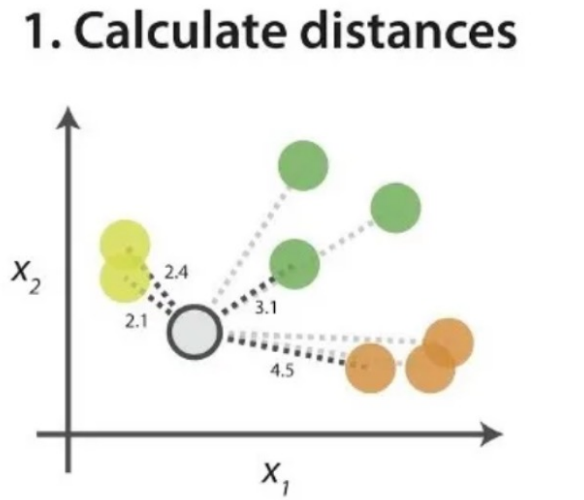
K-Nearest Neighbors

- KNN → A voting system
  - How to identify the voters? → Identify NNs by distance

Euclidean Distance



- `sklearn.metrics.DistanceMetric`



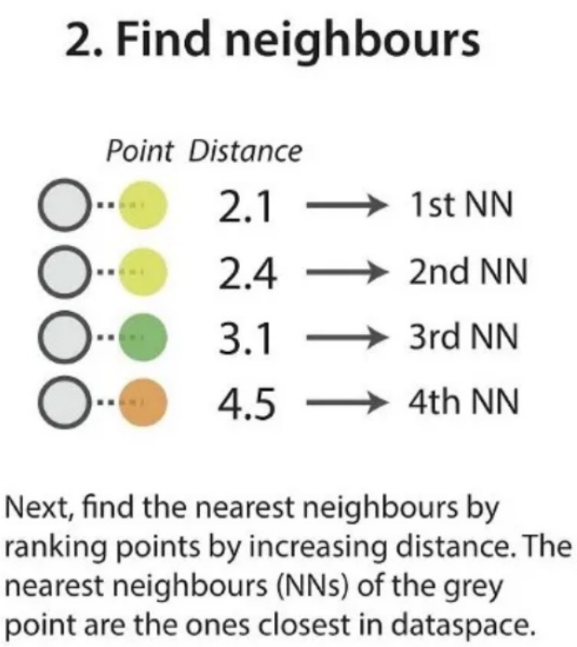


Image source: https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb

K-Nearest Neighbors

- KNN → A voting system
  - How to identify the voters? → Identify NNs by distance
    - Scale the data before calculating the distance

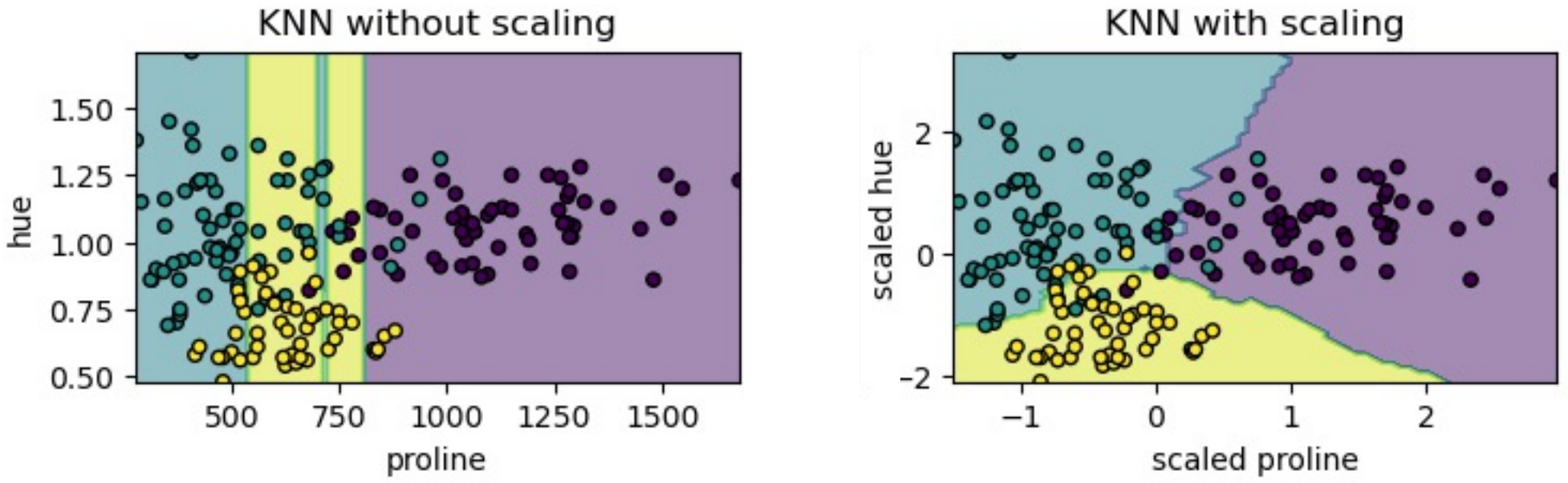


Image source: https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb

K-Nearest Neighbors

- KNN → A voting system
  - How to identify the voters? → Identify NNs by distance
    - Scale the data before calculating the distance
    - `sklearn.preprocessing.StandardScaler`
      - Standardize features by removing the mean and scaling to unit variance
        - Feature x
        - Scaled feature $x_{scaled} = \frac{x-u}{s}$
        - u is the mean value of feature x of the training dataset
        - s is the standard deviation of feature x of the training dataset

Image source: https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb

K-Nearest Neighbors

- KNN → A voting system
  - How to compute the voting result? → Generate prediction
    - KNN generates predictions based on the nearest neighbors
      - Are these nearest neighbors equally important?
      - Equally important → A simple majority voting
      - Closer neighbors are more important → Weight neighbors by the inverse of their distance

	Class	Distance
Neighbor 1	B	0.25
Neighbor 2	A	0.5
Neighbor 3	A	0.8

- *Majority voting*
  - *Class A: 1 + 1 = 2*
  - *Class B: 1*
- *Weighted voting*
  - *Class A: 1 / 0.5 + 1 / 0.8 = 3.25*
  - *Class B: 1 / 0.25 = 4*

Image source: https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb

K-Nearest Neighbors

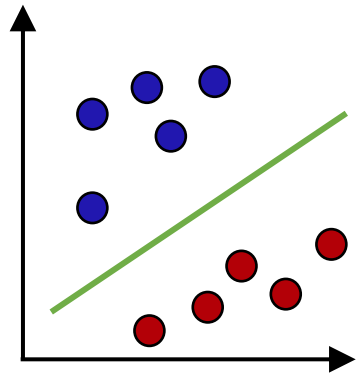
- Pros
  - Simplicity and intuitiveness
  - Doesn’t require training time (simply store the instances)
  - Can be used for multi-class classification
- Cons
  - Computationally intensive during prediction
    - Computing distance for all stored instances, which is time-consuming
  - Poor performance with imbalanced classes
    - The majority class may dominate the majority voting
  - Sensitivity to noise and outliers
    - Increasing the value of K may help

Outline

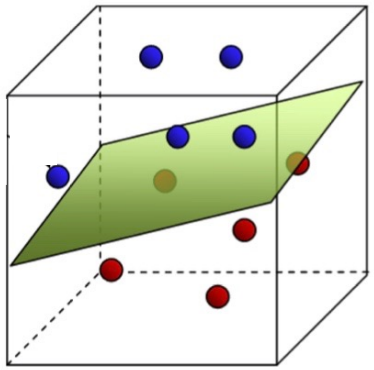
- Metrics for multi-class classification
- Nearest Neighbors
- **Support Vector Machine**

Support Vector Machine

- SVM
  - Supervised learning algorithm can be used for classification
  - For binary classification, SVM classifier finds a hyperplane that divides the feature space into two parts, each represents a category



2-dimensional
Line



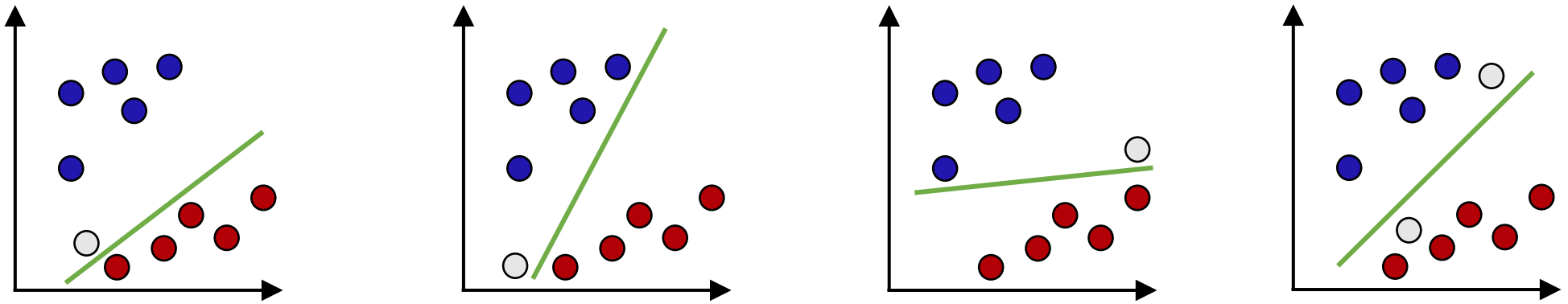
3-dimensional
Plane

?

High-dimensional
Hyperplane

Support Vector Machine

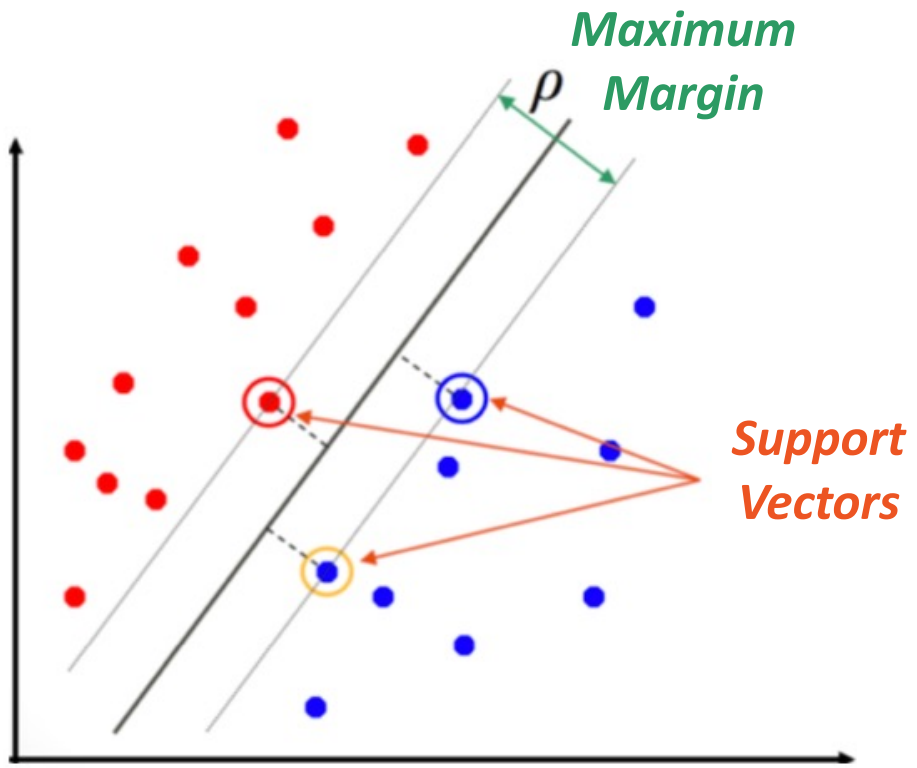
- SVM
  - There are multiple ways (hyperplanes) that can separate data points based on their classes.



  - Which hyperplane is the optimal one?

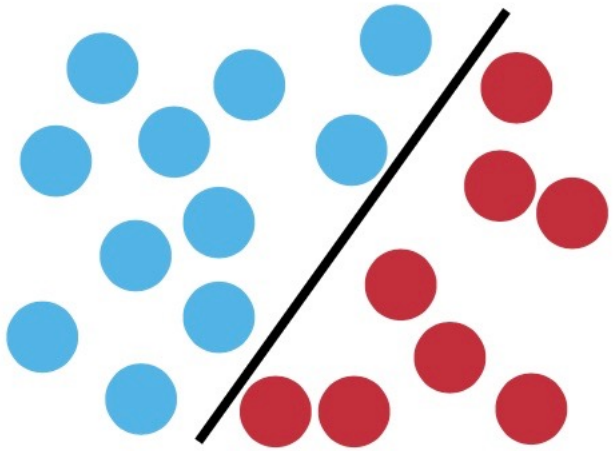
Support Vector Machine

- The optimal hyperplane
  - SVM select the hyperplane that separates data points with the maximum margin
  - Data points closet to the hyperplane are support vectors
  - Margin ρ of a hyperplane is the sum of the distances between support vectors and the hyperplane.

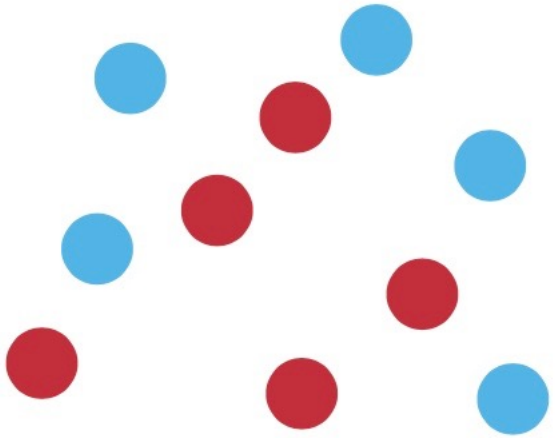


Support Vector Machine

- What if the data points are not linearly separable?



Linearly separable

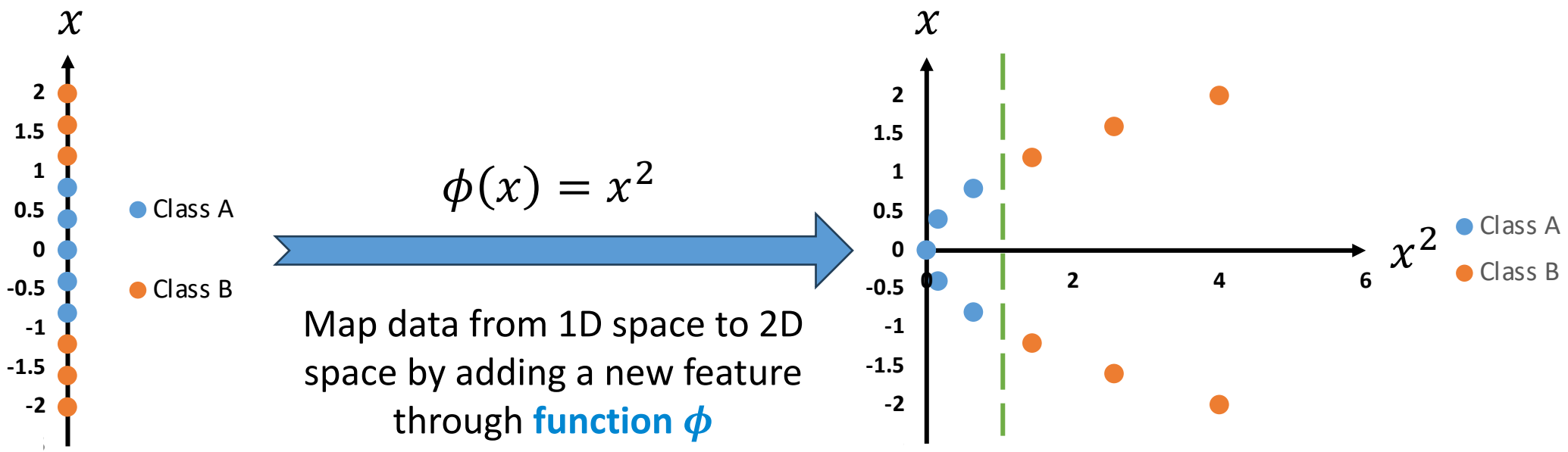


Not linearly separable

- Map data points to a high-dimensional linear-separable feature space

Support Vector Machine

- A dataset with only one feature x and two classes (A and B)



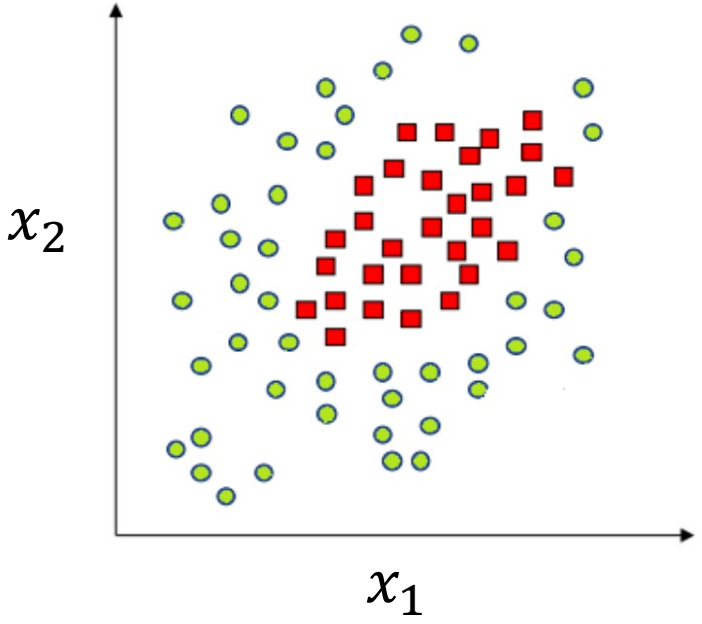
Not linearly separable

Linearly separable

Support Vector Machine

- A dataset with only two features x_1 and x_2

$$\phi(x_1, x_2) = (x_1 - a)^2 + (x_2 - b)^2$$



function ϕ

Map data from 2D space to 3D space

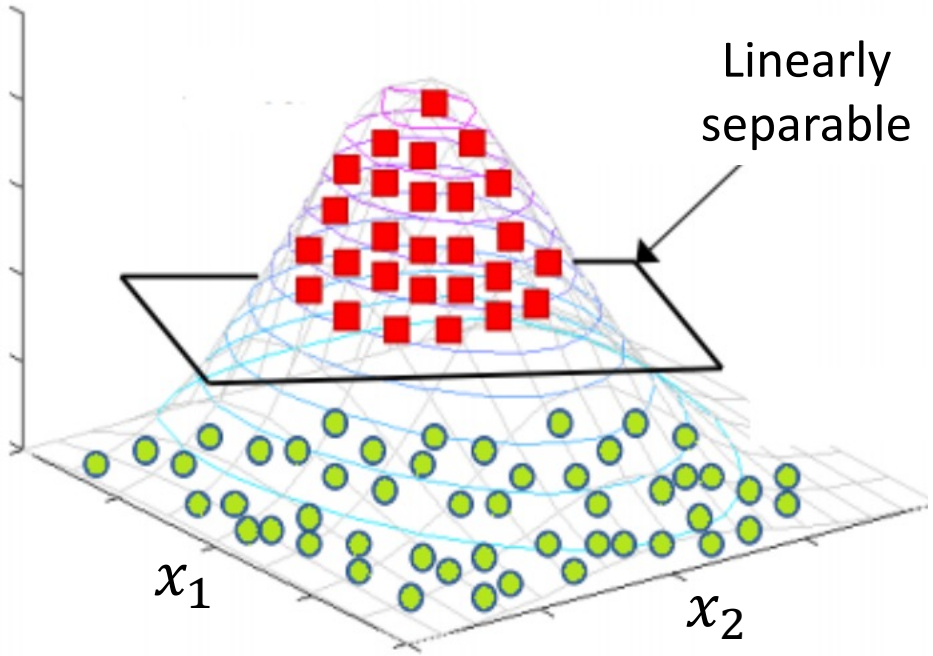


Image source: https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/

Support Vector Machine

- The kernel trick
  - Function ϕ is a **kernel** that maps the original data to a high-dimensional feature space
  - Convert nonlinearly separable low-dimensional space to a linearly separable high-dimensional space
  - Allow SVM to find a separating hyperplane in the transformed high-dimensional space

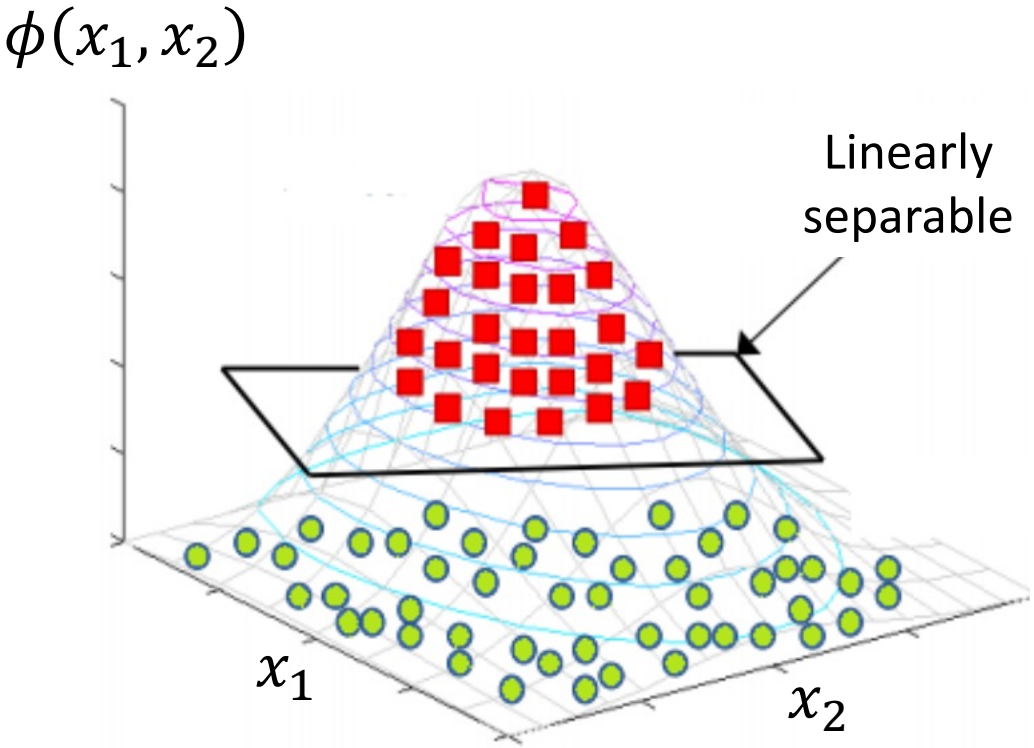


Image source: https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/

Support Vector Machine

- The kernel trick
  - Kernels provided by `sklearn`

1.4.6. Kernel functions

The *kernel function* can be any of the following:

- linear: $\langle x, x' \rangle$.
- polynomial: $(\gamma \langle x, x' \rangle + r)^d$, where d is specified by parameter `degree`, r by `coef0`.
- rbf: $\exp(-\gamma \|x - x'\|^2)$, where γ is specified by parameter `gamma`, must be greater than 0.
- sigmoid $\tanh(\gamma \langle x, x' \rangle + r)$, where r is specified by `coef0`.

Support Vector Machine

- Pros
  - Memory efficiency
    - Only use a subset of training data (support vectors) in the trained model
  - Robustness to over-fitting
    - A regularization parameter that penalizes the complexity of model

Parameters:

C : *float, default=1.0*

Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.

  - Can solve nonlinearly separable problem thanks to the kernel trick

kernel : *{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'} or callable, default='rbf'*

Specifies the kernel type to be used in the algorithm. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape `(n_samples, n_samples)`. For an intuitive visualization of different kernel types see Plot classification boundaries with different SVM Kernels.

Support Vector Machine

- Cons
  - Only directly applicable for binary classification
    - For multi-class problem
      - One-vs-One
      - One-vs-Rest
  - Sensitivity to the choice of kernel and regularization parameters
    - Tomorrow session
      - Cross validation
      - hyper-parameter tuning
  - Can’t directly provide probability predictions
  - Long training time on large datasets

Hands-on Exercise

- Exercise 05 Classification II