L03: K-Nearest Neighbors & K-Means

Overview

Methods and Algorithms

MSc Data Science

Spring 2026

Outline

1. Problem
2. Method
3. Solution
4. Practice
5. Summary

Can a Machine Learn by Looking at Its Neighbors?

Situation: A bank has millions of past transactions labeled as fraud or legitimate, and millions of customers with no predefined segments.

Complication: Traditional rule-based systems miss novel fraud patterns, and marketing teams lack data-driven customer segments.

Question: Can we classify new transactions by similarity to past ones, and discover natural customer groups without labels?

XKCD #1838 by Randall Munroe (CC BY-NC 2.5) – Today: KNN for classification, K-Means for clustering

What Is the Difference Between Classification and Clustering?

Two distinct problems banks face every day

Classification (Supervised Learning)

- Labeled training data: we know the right answers
- Predict fraud vs legitimate, approve vs deny

Clustering (Unsupervised Learning)

- No labels: discover natural groups in the data
- Find customer segments for targeted products

KNN = classification with labels, K-Means = clustering without labels

Why Do Similar Things Behave Similarly?

Human intuition: "similar things behave similarly"

- Medicine: patients with similar symptoms receive similar diagnoses
- Finance: borrowers with similar profiles have similar default rates
- Retail: customers who bought similar products want similar recommendations

KNN formalizes this reasoning: to predict an outcome, find the most similar past cases and use their known outcomes.

KNN formalizes our natural reasoning: look at similar past cases to predict new ones

Why Do Banks Need Customer Segments?

Business motivation for clustering

- **Targeted products:** Premium clients get wealth management, students get starter accounts
- **Risk management:** Group loans by risk profile for portfolio diversification
- **Regulatory compliance:** Differentiated treatment by risk tier (Basel requirements)

Segmentation transforms raw customer data into actionable business strategy.

Segmentation transforms raw customer data into actionable business insights

Learning Objectives

By the end of this lecture, you will be able to:

1. **Analyze** the bias-variance tradeoff in KNN as a function of K
2. **Prove** K-Means convergence and evaluate initialization strategies
3. **Evaluate** cluster validity using silhouette analysis and Gap statistic
4. **Compare** distance metrics and their impact in high dimensions

Finance Applications: Customer segmentation, fraud detection

Bloom's Level 4–5: Analyze, Evaluate, Prove, Compare

What Problem Are We Solving?

Classification vs Clustering in Practice

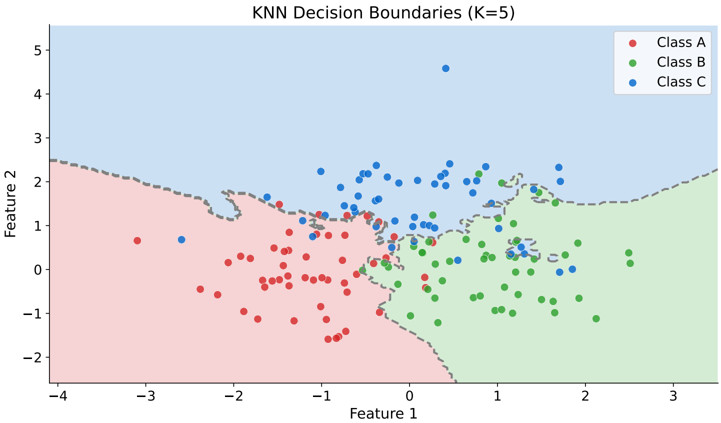
- **Fraud detection** (classification): labeled historical transactions, predict new ones
- **Customer segmentation** (clustering): no predefined groups, discover structure
- **Beware the "K":** K in KNN = number of neighbors; K in K-Means = number of clusters

When Supervised vs Unsupervised?

Labels available $\rightarrow$ KNN (or other classifiers)
No labels, seek structure $\rightarrow$ K-Means (or other clustering)

Same letter "K" means fundamentally different things in each algorithm

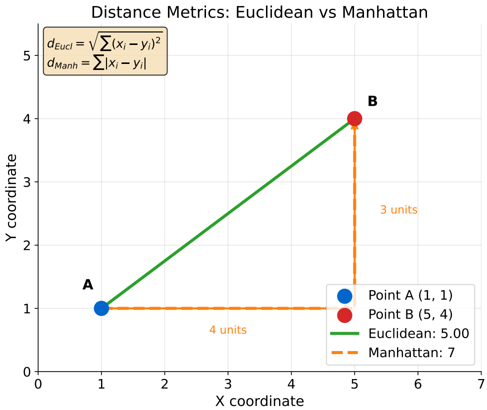
How Does KNN Draw Boundaries?



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L03_KNN_KMeans/01_knn_boundaries

KNN creates non-linear, flexible decision boundaries that adapt to local data density

What Do Different Distance Metrics Look Like?



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L03_KNN_KMeans/02_distance_metrics

- Euclidean (circle), Manhattan (diamond), and Chebyshev (square) define different neighborhoods
- The choice of distance metric directly shapes which points are considered nearest neighbors

Different metrics create different decision boundaries – choose based on your data structure

How Do We Measure Similarity?

Euclidean distance (most common):

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{\sum_{j=1}^{p} (x_j - x'_j)^2}$$

Manhattan distance: $d(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{p} |x_j - x'_j|$

- **Feature scaling is critical:** salary in thousands vs age in decades
- Always standardize features before computing distances
- Choice of metric affects which points are "nearest"

Unscaled features let high-magnitude variables dominate the distance calculation

How Does KNN Classify New Points?

Majority vote among K nearest neighbors:

$$\hat{y}(\mathbf{x}) = \arg\max_c \sum_{i \in \mathcal{N}_K(\mathbf{x})} \mathbf{1}(y_i = c)$$

Weighted vote (optional): closer neighbors get more influence

- $K = 1$**:** perfectly fits training data, high variance (overfitting)
- K **large:** smoother boundaries, high bias (underfitting)
- **Sweet spot:** use cross-validation to select K

The bias-variance tradeoff: small K = complex boundary, large K = smooth boundary

What Does K-Means Optimize?

Minimize within-cluster sum of squares (WCSS):

$$\min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K} \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in C_k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2$$

- Each point assigned to its **nearest centroid**
- Centroids are the **mean** of their assigned points
- NP-hard in general, but Lloyd's algorithm finds good local minima

---

WCSS measures how tightly packed points are within each cluster

How Does K-Means Work Step by Step?

Lloyd's Algorithm (3 Steps):

1. **Initialize:** choose K starting centroids
2. **Assign:** each point to its nearest centroid
3. **Update:** recompute centroids as cluster means

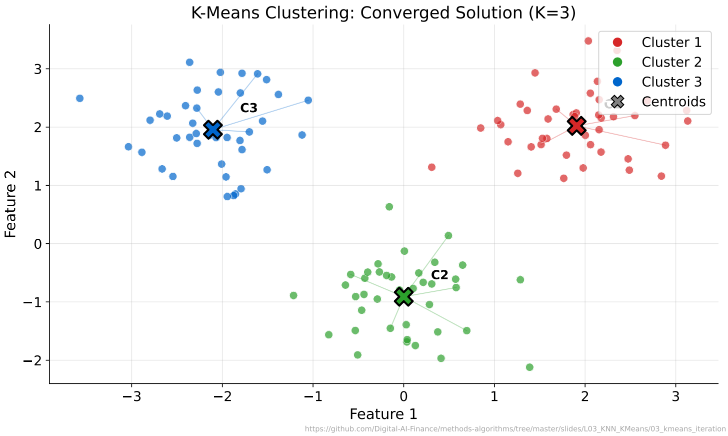
Repeat steps 2–3 until assignments stop changing.

Convergence Guarantee

WCSS decreases (or stays equal) at every step $\Rightarrow$ guaranteed to converge in finite iterations.

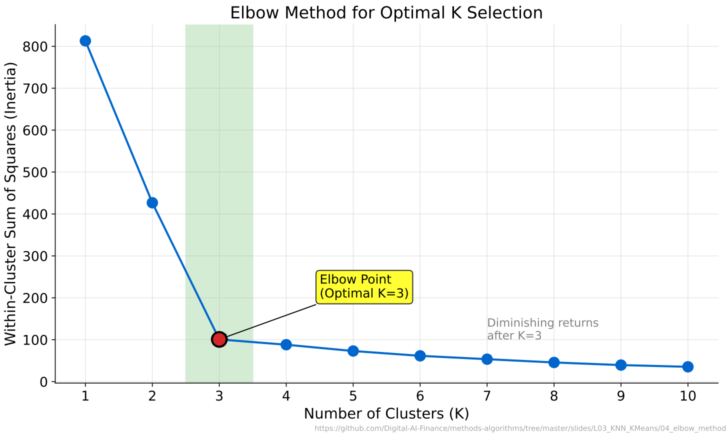
Convergence is guaranteed but only to a local minimum – initialization matters

What Does One K-Means Iteration Look Like?



Iteratively assign points to nearest centroid, then update centroids until convergence

How Do We Choose the Right K?



Look for the "elbow" where adding more clusters gives diminishing returns in WCSS reduction

Why Does Initialization Matter?

Problem: random initialization can lead to poor local minima.

K-Means++ strategy:

1. Pick first centroid uniformly at random
2. Pick next centroid with probability proportional to d^2 from nearest existing centroid
3. Repeat until K centroids chosen

- **Guarantee:** $O(\log K)$-competitive with optimal clustering
- **Default** in scikit-learn's `KMeans(init='k-means++')`

Arthur & Vassilvitskii (2007): K-Means++ spreads initial centroids apart for better convergence

How Do KNN and K-Means Compare?

Property	**KNN**	**K-Means**
Task	Classification	Clustering
Learning	Supervised	Unsupervised
K means	Neighbors	Clusters
Training	None (lazy)	Iterative
Prediction	$O(np)$ per query	$O(Kp)$ per query
Output	Class label	Cluster ID
Complexity	Scales with data	Scales with K

Despite sharing "K", these algorithms solve fundamentally different problems

How Do Banks Segment Customers?

RFM Analysis with K-Means

- **Features:** Recency (last transaction), Frequency (transaction count), Monetary (total spend)
- **Standardize** all features before clustering (different scales)
- **Result:** actionable segment profiles (e.g., high-value loyal, at-risk churners)

Business Impact

Each segment receives tailored products, pricing, and communication strategies.

RFM segmentation is a standard technique in retail banking and CRM analytics

Can KNN Detect Fraud?

The class imbalance challenge: fraud is rare ($<1\%$ of transactions).

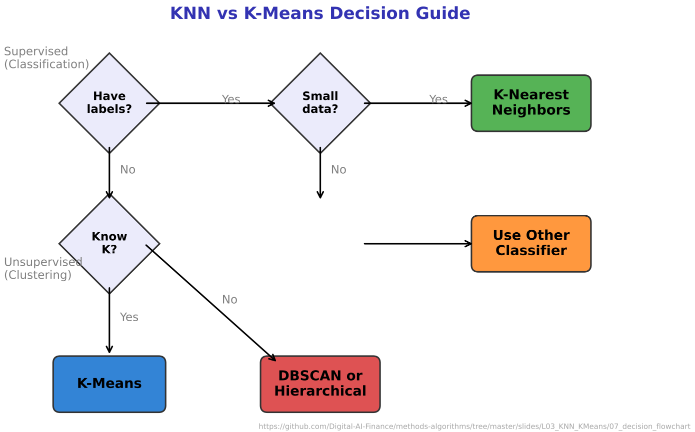
- **Problem:** accuracy is misleading (99% accuracy by predicting "no fraud")
- **Solutions:** SMOTE oversampling, distance-weighted KNN, cost-sensitive learning
- **Metric:** use Precision-Recall AUC, not accuracy

Why KNN Works Here

Fraudulent transactions cluster in feature space – KNN detects them by proximity to known fraud cases.

In fraud detection, false negatives (missed fraud) are far costlier than false positives

Which Method Should You Choose?



KNN for labeled data classification, K-Means for unlabeled data clustering

Hands-on Exercise

Open the Colab Notebook

1. **KNN from scratch:** implement distance calculation and majority vote
2. **Customer segmentation:** apply K-Means to RFM banking data, interpret clusters
3. **Model selection:** compare distance metrics and K values using cross-validation

Link: See course materials for Colab notebook

Exercises progress from implementation to application to critical evaluation

What Should You Remember?

KNN (Classification)

- Non-parametric, lazy learner – no training phase
- Small K: flexible but noisy; large K: smooth but biased

K-Means (Clustering)

- Iterative algorithm with guaranteed convergence (to local minimum)
- K-Means++ initialization avoids poor starting points

Common Considerations

- Feature scaling is essential for both algorithms
- Choosing K requires validation (cross-validation or elbow/silhouette)

Both algorithms depend critically on distance – get the distance right, get the result right

Until Next Time...

"Even K-Means would struggle to cluster the ways students misuse K-Means."

With KNN and K-Means, you can now classify the known and discover the unknown.

Next Session: L04 – Random Forests (from distance-based to tree-based methods)

XKCD #2731 callback – clustering is easy, knowing when to cluster is the hard part

References

Core Textbooks

- James et al., *ISLR* (2021), Chapters 2 & 12
- Hastie et al., *Elements of Statistical Learning* (2009), Chapters 13 & 14

Key Papers

- Arthur & Vassilvitskii (2007), "K-Means++: The Advantages of Careful Seeding"
- Cover & Hart (1967), "Nearest Neighbor Pattern Classification"

Next Lecture: L04 – Random Forests: ensemble methods and tree-based learning

ISLR Chapter 2 covers KNN; Chapter 12 covers clustering including K-Means