Classification I — KNN & Naïve Bayes

Faculty of DS & AI
Autumn semester, 2026

Trong-Nghia Nguyen

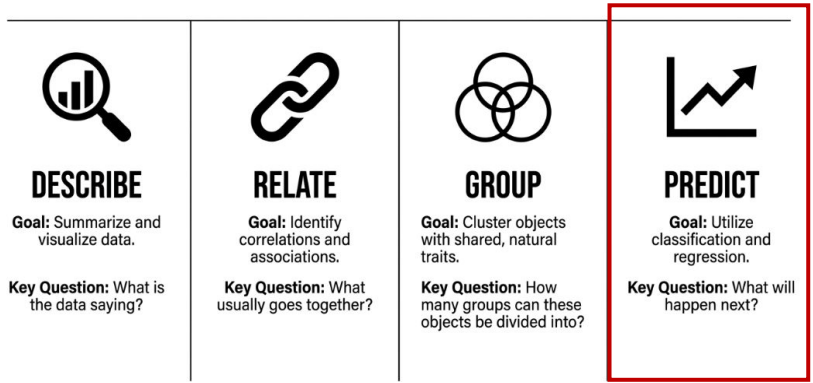
Business AI Lab

Content

- **Classification**
- k-Nearest Neighbors
- Naive Bayes
- Model Evaluation
- Tools
- Example

Classification

Four major problem groups in data mining



Today we start the **PREDICT** group: learn from **labelled history** and predict the **class** of new records.

Classification

Definition

- **Clustering** (Week 7): no labels → *unsupervised*.
- **Classification**: labels are known in past data → *supervised*; the target is a **category**.
- **Regression** (Week 11): supervised, but the target is a **number**.

Classification (Tan et al., 2019, Ch. 3)

- Given a **training set** of records (x, y): x is the **attribute set**, y the **class label**.
- **Task:** learn a model that maps each attribute set x to *one* of the *predefined* class labels y.
- **Goal:** *previously unseen* records are assigned a class as accurately as possible (measured on a test set).

$$D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\}$$
$$y^{(j)} \in \mathcal{C} = \{c_1, c_2, \ldots, c_K\}$$

Training set of n labelled records; $\mathcal{C}$ is a finite, predefined set of K classes.

$$f : \mathcal{X} \longrightarrow \mathcal{C}, \qquad \hat{y} = f(x), \quad x = (x_1, \ldots, x_m)$$

The classifier f maps the m features of a record to exactly one class $\hat{y}$.

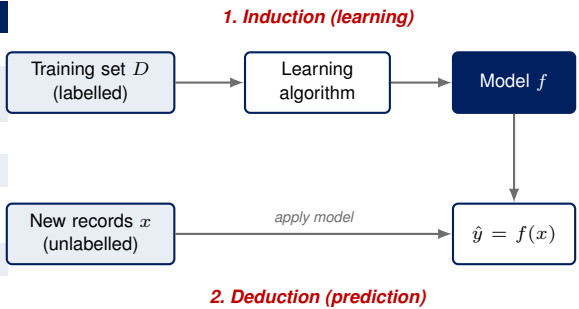
Three signs of a classification problem

1. The target is **categorical**: a finite set of known classes.
2. Historical data are **labelled** (supervised learning).
3. Classes are **mutually exclusive**: one class per record.

Classification

Components & two phases

Component	Symbol	Churn example
Record / instance	a row	one customer
Attributes (features)	x	app usage, satisfaction, contract
Class label (target)	y	`Churn`
Set of classes	$\mathcal{C}$	{Yes, No}
Training set	D	last year's customers, outcome known
Classifier (model)	f	a KNN or Naive Bayes model
Prediction	$\hat{y} = f(x)$	churn forecast for an active customer



North calls the new, unlabelled records **scoring data**.

Algorithm $\neq$ model: the same algorithm (e.g. Naive Bayes) learns *different* models from different data.

Classification

Output & problem types

- **Hard output:** a label $\hat{y} \in \mathcal{C}$, e.g. “Churn = Yes”.
- **Soft output:** a probability $\hat{P}(y = c \mid x)$ for every class (they sum to 1).

$$\hat{y} = \arg\max_{c \in \mathcal{C}} \hat{P}(y = c \mid x)$$

Default rule: choose the most probable class. With two classes this equals a 0.5 cutoff.

Probabilities let the business **rank** customers by risk. KNN (vote share) and Naive Bayes (posterior) both give them.

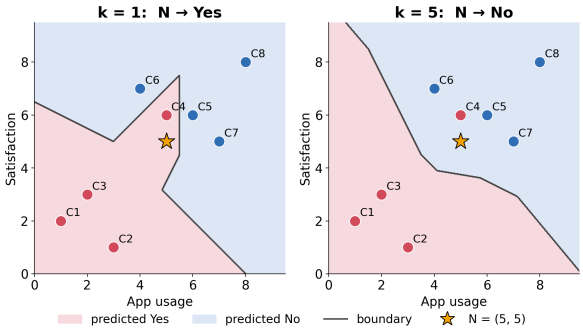
Type	Definition	Business example
Binary	$K = 2$; one class is the **positive** class (the one to detect)	churn Yes/No; loan default; fraud
Multi-class	$K > 2$; each record in *exactly one* class	credit grade A/B/C/D; plan chosen
Multi-label	a record may carry *several* labels at once	topics of a product review *(out of scope)*

Class imbalance

One class is rare (churn 10–20%, fraud $< 1\%$) — and it is usually the positive class. It changes how we **split** the data (stratify) and how we **evaluate** (beyond accuracy).

Classification

Decision boundary & algorithm families



A classifier splits the feature space into class regions; their border is the ***decision boundary****. Small k gives a jagged boundary that follows single points.*

Family	Idea	Example (week)
Nearest neighbour	compare with known examples	**KNN** (today)
Probabilistic	class probability via Bayes' theorem	**Naive Bayes** (today)
Separating function	a line / surface that splits the classes	logistic regression (11); discriminant analysis (North Ch. 7)
Trees & networks	if–then questions; layered transformations	decision tree, neural network (13)

Eager learner: builds the model in advance (Naive Bayes, trees, logistic regression).

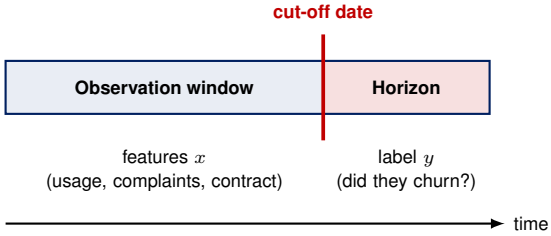
Lazy learner: only stores the data and works at prediction time (**KNN**).

Classification

Defining the problem in business

CRISP-DM **Business Understanding**: define the problem *before* choosing an algorithm.

- **Record:** one *active* customer at the cut-off date.
- **Label:** e.g. “Churn = Yes if the customer does not renew within 30 days after the plan expires”. A vague label gives a useless model.
- **Prediction horizon:** next month? next quarter?
- **Positive class & action:** what we detect, and what we do next (offer, call).
- **Features available at prediction time:** only information known *before* the outcome.



Data leakage

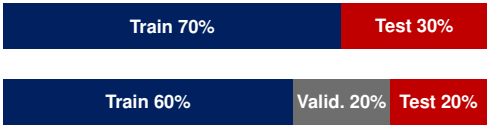
Never use information that only exists *after* the outcome. Using “contract cancellation date” to predict churn gives near-100% accuracy on paper and is useless in deployment.

Classification

Train / test split

A model must be scored on data it has **never seen** — a new exam, not the solved one.

- **Training set** (70–80%): the model learns from it.
- **Test set** (20–30%): hidden, used *once* for the final evaluation.
- **Stratified** split: keep the class proportions in both sets (vital when churners are only $\sim$20%).
- To **choose a parameter** such as k: train / **validation** / test — tune on validation, report on test.
- **Overfitting:** excellent on training data, poor on test data (the model memorised noise). **Underfitting:** poor on both.



Why never evaluate on the training set?

With **1-NN**, the nearest neighbour of every training point is *itself*, so training accuracy is always **100%** (when no two identical points have different labels). It says nothing about new customers.

Content

- Classification
- **k-Nearest Neighbors**
- Naive Bayes
- Model Evaluation
- Tools
- Example

k-Nearest Neighbors

Idea & formula

"Tell me who your neighbours are, and I will tell you who you are."

To classify a new customer, find the k **most similar customers** in the training set and take the **majority label**.

1. Choose k and a distance measure.
2. Compute the distance from the new point to **every** training point.
3. Keep the k nearest points.
4. **Majority vote** $\rightarrow$ predicted class.

No real training step: KNN only stores the data $\rightarrow$ a **lazy learner**.

$$d\big(x, x^{(j)}\big) = \sqrt{\sum_{i=1}^{m} \big(x_i - x_i^{(j)}\big)^2}$$

Euclidean distance (as in k-Means, Week 7) between the new point x and training point $x^{(j)}$.

$$\hat{y} = \arg\max_{c} \sum_{j \in N_k(x)} \mathbf{1}\big(y^{(j)} = c\big)$$

Majority vote; $N_k(x)$ is the set of the k nearest neighbours of x.

$$\hat{P}(\text{Yes} \mid x) = \frac{\#\,\text{neighbours with label Yes}}{k}$$

The vote share is also a probability estimate.

k-Nearest Neighbors

Example

Steps 1–2: distances to the new customer $N = (5, 5)$

Training data: 8 customers, both features on a 0–10 scale (no extra scaling needed).

ID	AppUse	Satisf.	Churn
C1	1	2	Yes
C2	3	1	Yes
C3	2	3	Yes
C4	5	6	Yes
C5	6	6	No
C6	4	7	No
C7	7	5	No
C8	8	8	No

Step 1 — distances, for example:

$$d(N, C4) = \sqrt{(5-5)^2 + (5-6)^2} = \sqrt{1} = 1$$

$$d(N, C5) = \sqrt{(5-6)^2 + (5-6)^2} = \sqrt{2} \approx 1.414$$

$$d(N, C3) = \sqrt{(5-2)^2 + (5-3)^2} = \sqrt{13} \approx 3.606$$

k-Nearest Neighbors

Example

Steps 1–2: distances to the new customer $N = (5, 5)$

Training data: 8 customers, both features on a 0–10 scale (no extra scaling needed).

ID	AppUse	Satisf.	Churn
C1	1	2	Yes
C2	3	1	Yes
C3	2	3	Yes
C4	5	6	Yes
C5	6	6	No
C6	4	7	No
C7	7	5	No
C8	8	8	No

Step 1 — distances, for example:

$$d(N, C4) = \sqrt{(5-5)^2 + (5-6)^2} = \sqrt{1} = 1$$

$$d(N, C5) = \sqrt{(5-6)^2 + (5-6)^2} = \sqrt{2} \approx 1.414$$

$$d(N, C3) = \sqrt{(5-2)^2 + (5-3)^2} = \sqrt{13} \approx 3.606$$

Step 2 — sort from nearest to farthest

Rank	1	2	3	4	5	6	7, 8
Customer	C4	C5	C7	C6	C3	C8	C2, C1
d	1.000	1.414	2.000	2.236	3.606	4.243	4.472, 5.000
Churn	Yes	No	No	No	Yes	No	Yes, Yes

k-Nearest Neighbors

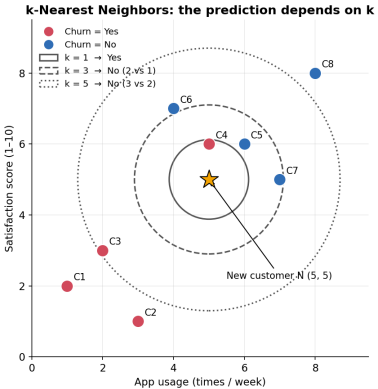
Example

Steps 3–4: majority vote for different k

k	Neighbours	Votes	Prediction
1	C4	1 Yes	**Yes**
3	C4, C5, C7	1 Yes – 2 No	**No**
5	C4, C5, C7, C6, C3	2 Yes – 3 No	**No**

- C4 is an **exception**: satisfied but still churned (e.g. moved city).
- With $k = 1$ the model follows this single noisy point.
- With $k = 3$ or 5 most neighbours stayed $\rightarrow$ **No**, with $\hat{P}(\text{Yes}) = 1/3$ or $2/5$.

Results verified with scikit-learn.



k-Nearest Neighbors

Choosing k & normalization

Choosing k

- **Small** k (e.g. 1): follows every point, sensitive to noise $\rightarrow$ overfitting.
- **Large** k: smoother, may ignore local structure $\rightarrow$ underfitting. If $k = n$, the model always predicts the majority class.
- Two classes: use an **odd** k to avoid ties.
- Pick k with the best result on the **validation set** (or cross-validation) — never on the test set.
- **Distance-weighted vote**: closer neighbours count more, $w_j = 1/d_j$ (Orange kNN: *Weight: Distance*).

Normalization is mandatory

- KNN is distance-based: the **Golden Rule** of Week 7 applies. Income (millions) next to complaints (0–5) $\rightarrow$ the distance is driven by income only.
- **Orange:** the kNN widget does **not** normalize (it only encodes categories and imputes) $\rightarrow$ add **Preprocess** $\rightarrow$ **Normalize**.
- **Python:** fit `MinMaxScaler` on the **training set only**, then apply it to the test set (fitting on all data leaks test information).
- Categorical features must be one-hot encoded before computing distances (Orange does it automatically).

Content

- Classification
- k-Nearest Neighbors
- **Naive Bayes**
- Model Evaluation
- Tools
- Example

Naive Bayes

Bayes' theorem & the naive assumption

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

$$P(C \mid X) = \frac{P(X \mid C)\, P(C)}{P(X)}$$

Conditional probability (top); Bayes' theorem for a class C and observed data X (bottom).

Term	Name	Meaning (churn)
$P(C)$	prior	overall churn rate
$P(X \mid C)$	likelihood	share of churners who show profile X
$P(C \mid X)$	posterior	churn probability of *this* customer — what we want
$P(X)$	evidence	same for all classes; only normalizes

With many features $X = (x_1, \ldots, x_m)$, estimating $P(X \mid C)$ directly needs huge data. Naive Bayes assumes the **features are independent given the class**:

$$P(X \mid C) = \prod_{i=1}^{m} P(x_i \mid C)$$

$$\text{score}(C) = P(C) \prod_{i=1}^{m} P(x_i \mid C)$$

$P(X)$ is common to all classes, so comparing scores is enough: $\hat{y} = \arg\max_C$ *score*(C).

$$P(C_j \mid X) = \frac{\text{score}(C_j)}{\sum_l \text{score}(C_l)}$$

Normalize the scores to obtain probabilities.

Naive Bayes

Example

Training data: 12 customers, 3 categorical features

ID	Contract	Complained	AutoPay	Churn
K1	Monthly	Yes	No	**Yes**
K2	Monthly	Yes	No	**Yes**
K3	Monthly	Yes	Yes	**Yes**
K4	Monthly	No	No	**Yes**
K5	Monthly	No	No	**Yes**
K6	Monthly	Yes	No	**No**
K7	Monthly	No	Yes	**No**
K8	Annual	No	Yes	**No**
K9	Annual	No	Yes	**No**
K10	Annual	No	Yes	**No**
K11	Annual	No	No	**No**
K12	Annual	No	Yes	**No**

Step 1 — count by class (5 churners, 7 non-churners)

Feature value	Churn = Yes (5)	Churn = No (7)
Contract = Monthly	5	2
Contract = Annual	0	5
Complained = Yes	3	1
Complained = No	2	6
AutoPay = Yes	1	5
AutoPay = No	4	2

New customer A

Contract = **Monthly**, Complained = **Yes**, AutoPay = **No**. Will A churn?

Naive Bayes

Example

Customer A = (Monthly, Complained = Yes, AutoPay = No)

Step 2 — score of each class

$$\text{score(Yes)} = \underbrace{\frac{5}{12}}_{P(\text{Yes})} \times \underbrace{\frac{5}{5}}_{P(\text{Monthly}|\text{Yes})} \times \underbrace{\frac{3}{5}}_{P(\text{Compl.=Yes}|\text{Yes})} \times \underbrace{\frac{4}{5}}_{P(\text{AutoPay=No}|\text{Yes})} = 0.2000$$

Naive Bayes

Example

Customer A = (Monthly, Complained = Yes, AutoPay = No)

Step 2 — score of each class

$$\text{score(Yes)} = \underbrace{\frac{5}{12}}_{P(\text{Yes})} \times \underbrace{\frac{5}{5}}_{P(\text{Monthly}|\text{Yes})} \times \underbrace{\frac{3}{5}}_{P(\text{Compl.=Yes}|\text{Yes})} \times \underbrace{\frac{4}{5}}_{P(\text{AutoPay=No}|\text{Yes})} = 0.2000$$

$$\text{score(No)} = \underbrace{\frac{7}{12}}_{P(\text{No})} \times \frac{2}{7} \times \frac{1}{7} \times \frac{2}{7} \approx 0.0068$$

Naive Bayes

Example

Customer A = (Monthly, Complained = Yes, AutoPay = No)

Step 2 — score of each class

$$\text{score(Yes)} = \underbrace{\frac{5}{12}}_{P(\text{Yes})} \times \underbrace{\frac{5}{5}}_{P(\text{Monthly}|\text{Yes})} \times \underbrace{\frac{3}{5}}_{P(\text{Compl.=Yes}|\text{Yes})} \times \underbrace{\frac{4}{5}}_{P(\text{AutoPay=No}|\text{Yes})} = 0.2000$$

$$\text{score(No)} = \underbrace{\frac{7}{12}}_{P(\text{No})} \times \frac{2}{7} \times \frac{1}{7} \times \frac{2}{7} \approx 0.0068$$

Step 3 — normalize

$$P(\text{Yes} \mid A) = \frac{0.2000}{0.2000 + 0.0068} \approx \mathbf{0.967}$$

$\Rightarrow$ Predict **Yes**. A monthly-contract customer who complained and has no auto-payment is at **very high risk** — prioritise retention.

Naive Bayes

Zero frequency & Laplace smoothing

Customer B = (**Annual**, Complained = Yes, AutoPay = No).
No churner has an annual contract:
$P(\text{Annual} \mid \text{Yes}) = 0/5 = 0.$

$$\text{score(Yes)} = \tfrac{5}{12} \times \tfrac{0}{5} \times \tfrac{3}{5} \times \tfrac{4}{5} = 0 \;\Rightarrow\; P(\text{Yes} \mid B) = 0$$

One zero count **wipes out** all other evidence (B complained and has no auto-payment).

$$P(x_i = v \mid C) = \frac{n_{C,v} + 1}{n_C + K_i}$$

Laplace (+1) smoothing; K_i = number of values of feature i (here $K_i = 2$).

Why $+K_i$? The smoothed probabilities of a feature must still sum to 1 (Contract, class Yes: Monthly + Annual):

$$\tfrac{5+1}{5+2} + \tfrac{0+1}{5+2} = \tfrac{6}{7} + \tfrac{1}{7} = 1$$

Recompute for B with smoothing

$$\text{score(Yes)} = \tfrac{5}{12} \times \tfrac{1}{7} \times \tfrac{4}{7} \times \tfrac{5}{7} \approx 0.0243$$

$$\text{score(No)} = \tfrac{7}{12} \times \tfrac{6}{9} \times \tfrac{2}{9} \times \tfrac{3}{9} \approx 0.0288$$

$$P(\text{Yes} \mid B) = \tfrac{0.0243}{0.0243+0.0288} \approx \mathbf{0.458}$$

$\Rightarrow$ still **No**, but a **close call** (46%) instead of 0%.

- With smoothing, customer A gives $P(\text{Yes} \mid A) \approx 0.910$ — still Yes.
- Only the conditional probabilities are smoothed; priors stay $5/12$ and $7/12$ (as in scikit-learn's `CategoricalNB`).
- Smoothing pulls extreme probabilities towards the middle. It matters most with **little data**; with thousands of records the $+1$ is negligible.

Naive Bayes

Numeric features

Counting works only for categories. For a numeric feature (tenure, income) almost every customer has a different value, so there is nothing to count. Two options:

- **Discretize** into bins (Low / Medium / High), then count as before. Simple, but information inside a bin is lost and the result depends on the bins. Orange's **Naive Bayes** widget does this automatically (and applies +1 smoothing), so its numbers can differ slightly from a hand calculation.
- **Gaussian Naive Bayes:** assume the feature is normally distributed within each class.

$$P(x \mid C) = \frac{1}{\sqrt{2\pi}\,\sigma_C} \exp\left(-\frac{(x-\mu_C)^2}{2\sigma_C^2}\right)$$

μ_C, σ_C: mean and standard deviation of the feature within class C.

Example — tenure (months): churners $\mu = 4,\ \sigma = 2$; stayers $\mu = 20,\ \sigma = 8$. A new customer has **6 months**.

$$P(6 \mid \text{Yes}) = \frac{1}{\sqrt{2\pi}\cdot 2} \exp\left(-\frac{(6-4)^2}{2\cdot 2^2}\right) = 0.1995 \times e^{-0.5} \approx \mathbf{0.121}$$

$$P(6 \mid \text{No}) = \frac{1}{\sqrt{2\pi}\cdot 8} \exp\left(-\frac{(6-20)^2}{2\cdot 8^2}\right) = 0.0499 \times e^{-1.531} \approx \mathbf{0.011}$$

- $0.121/0.011 \approx 11$: a 6-month tenure looks about **11 times** more like a churner. The priors and the other features are then multiplied in as usual.
- These are **densities**, not probabilities: they can exceed 1. Only their **ratio** between classes matters, because the scores are normalized at the end.

Naive Bayes

Why “naive” still works

The assumption is usually wrong

- Independence rarely holds: monthly-contract customers also tend to skip auto-payment.
- Naive Bayes multiplies both likelihoods as if they were separate evidence, so it **counts the same evidence twice**.
- Consequence: the **probabilities** are often exaggerated (too close to 0 or 1). The 0.967 of customer A is probably over-confident.

. . . but the decision is often right

- To classify, the model only needs to know **which class scores higher**, not the exact probability.
- The **ranking** (who is riskier) is often still good — exactly what a retention campaign needs to choose whom to contact.
- Fast, needs little data, handles many features: the classic choice for **spam filtering**, where words are far from independent, yet it works well.

Take-away

Wrong assumption, often the right decision: trust the **ranking** more than the exact probability.

Further reading: Domingos & Pazzani (1997), On the optimality of the simple Bayesian classifier under zero-one loss, Machine Learning 29, 103–130.

Content

- Classification
- k-Nearest Neighbors
- Naive Bayes
- **Model Evaluation**
- Tools
- Example

Model Evaluation

Confusion matrix & metrics

Choose the **positive class** = the class we want to detect: **Churn = Yes**. "Positive" does not mean "good".

	Pred. Yes	Pred. No
Actual Yes	**TP** churn predicted, churned	**FN** missed a churner
Actual No	**FP** false alarm	**TN** stay predicted, stayed

Rows = actual, columns = predicted (as in Orange's Confusion Matrix).

$$\text{Accuracy} = \frac{TP + TN}{N} \qquad \text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN} \qquad \text{Specificity} = \frac{TN}{TN + FP}$$

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$N = TP + TN + FP + FN$ *(all test records);* *Error rate* $= 1 -$ *Accuracy.*

- **Precision:** of the customers we *flag*, how many really churn?
- **Recall (sensitivity):** of the real churners, how many do we catch?
- **Specificity:** of the customers who stay, how many are recognised?
- $\mathbf{F_1}$: harmonic mean — high only if *both* precision and recall are high.

Model Evaluation

Example

Test set: 200 customers, 40 actually churned

	Pred. Yes	Pred. No	Total
Actual Yes	TP = 30	FN = 10	40
Actual No	FP = 20	TN = 140	160
Total	50	150	200

$$\text{Accuracy} = \frac{30+140}{200} = 0.85$$

$$\text{Precision} = \frac{30}{30+20} = 0.60$$

$$\text{Recall} = \frac{30}{30+10} = 0.75$$

$$\text{Specificity} = \frac{140}{140+20} = 0.875$$

$$F_1 = \frac{2 \times 0.60 \times 0.75}{0.60+0.75} \approx 0.667$$

Model Evaluation

Example

Test set: 200 customers, 40 actually churned

	Pred. Yes	Pred. No	Total
Actual Yes	TP = 30	FN = 10	40
Actual No	FP = 20	TN = 140	160
Total	50	150	200

Reading: the model catches **75%** of the churners; of the flagged customers **60%** really churn — 3 of every 5 retention vouchers reach the right person.

$$\text{Accuracy} = \frac{30+140}{200} = 0.85$$

$$\text{Precision} = \frac{30}{30+20} = 0.60$$

$$\text{Recall} = \frac{30}{30+10} = 0.75$$

$$\text{Specificity} = \frac{140}{140+20} = 0.875$$

$$F_1 = \frac{2\times 0.60\times 0.75}{0.60+0.75} \approx 0.667$$

Model Evaluation

Example

Test set: 200 customers, 40 actually churned

	Pred. Yes	Pred. No	Total
Actual Yes	TP = 30	FN = 10	40
Actual No	FP = 20	TN = 140	160
Total	50	150	200

Reading: the model catches **75%** of the churners; of the flagged customers **60%** really churn — 3 of every 5 retention vouchers reach the right person.

$$\text{Accuracy} = \frac{30+140}{200} = 0.85$$

$$\text{Precision} = \frac{30}{30+20} = 0.60$$

$$\text{Recall} = \frac{30}{30+10} = 0.75$$

$$\text{Specificity} = \frac{140}{140+20} = 0.875$$

$$F_1 = \frac{2\times 0.60\times 0.75}{0.60+0.75} \approx 0.667$$

The accuracy paradox

A lazy model that **always predicts No**: Accuracy $= \frac{0+160}{200} =$ **0.80**, but Recall $= \frac{0}{0+40} =$ **0** (precision is undefined, $0/0$). With a rare positive class, high accuracy proves nothing — always check precision, recall and F_1.

Model Evaluation

Choosing the metric

Problem	Costlier error	Focus
Churn	**FN**: a lost customer and all future revenue; an FP only costs a voucher	**Recall**
Spam filter	**FP**: an important e-mail lands in spam	**Precision**
Card fraud	**FN**: a fraudulent transaction passes	**Recall**, with enough precision

Discuss

The company can only afford vouchers for **50 customers per month**. Should the cutoff favour precision or recall? Why?

Decision threshold (cutoff)

- KNN and Naive Bayes output probabilities; by default *Yes* if $P(\text{Yes}) \geq 0.5$.
- **Lower** the cutoff (e.g. 0.3): more customers flagged $\rightarrow$ recall $\uparrow$, precision usually $\downarrow$. **Raise** it: the opposite.
- The cutoff is a **business decision** (budget). ROC curve and AUC: next week's lab.

Cross-validation (preview)

- Split the data into k folds; each fold serves once as the test set; average the results — more stable with little data.
- Orange **Test and Score** uses 5-fold cross-validation by default. Details in Week 13.

Content

- Classification
- k-Nearest Neighbors
- Naive Bayes
- Model Evaluation
- **Tools**
- Example

Tools

Python

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, classification_report

X = df[["AppUse", "Satisfaction"]]          # features
y = df["Churn"]                              # class label
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, stratify=y, random_state=42)

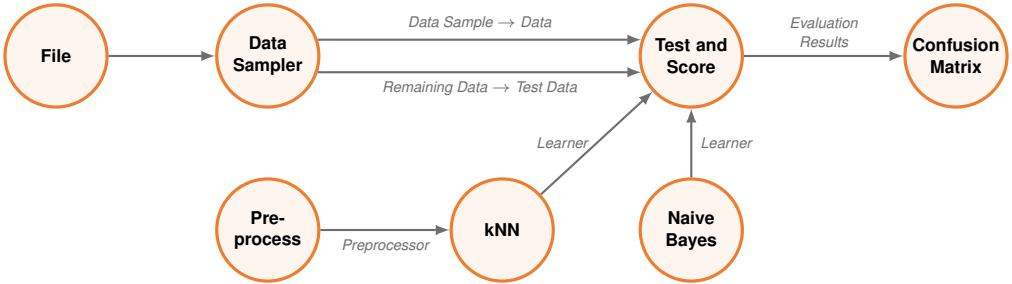
scaler = MinMaxScaler().fit(X_train)         # fit on TRAIN only
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(scaler.transform(X_train), y_train)  # stores the data
y_pred = knn.predict(scaler.transform(X_test))

print(confusion_matrix(y_test, y_pred, labels=["Yes", "No"]))
print(classification_report(y_test, y_pred))
```

- `stratify=y`: keeps the Yes/No ratio in both sets.
- The scaler is **fitted on the training set only** to avoid leakage.
- `confusion_matrix`: rows = actual, columns = predicted.
- `classification_report`: precision, recall, F_1 per class.
- Naive Bayes: `CategoricalNB()` (categorical features, Laplace smoothing by default) or `GaussianNB()` (numeric features).

Tools

Orange



- **Data Sampler:** fixed proportion 70% (default); tick *Stratify sample (when possible)*.
- **Preprocess** → *Normalize features* feeds kNN only; kNN does not normalize by itself (default: 5 neighbours, Euclidean, uniform weights).
- **Test and Score:** choose *Test on test data*. Without Data Sampler it runs 5-fold cross-validation by default.
- Set **Target class = Yes**; the default *(None, show average over classes)* averages precision/recall over both classes.

Tools

KNN vs Naive Bayes

Criterion	KNN	Naive Bayes
Idea	follow the label of the most similar neighbours	the class most likely to have produced the observed data
Foundation	**distance**	**probability** (Bayes' theorem)
Training	none (lazy): stores the data	very fast: counting frequencies
Prediction	slow on large data (distance to every point)	very fast
Normalization	**mandatory**	not needed
Best suited to	numeric features (categories must be encoded)	categorical features (numbers discretized or Gaussian)
Main parameter	k, distance measure	smoothing strength
Strengths	intuitive ("this customer resembles 5 churners"); flexible boundaries	needs little data; many features; gives probabilities
Weaknesses	sensitive to scale, irrelevant features, many dimensions	independence assumption; zero-frequency problem

No algorithm is always better: try both and compare them on the **same test set** with a metric that fits the business problem — that is next week's lab.

Content

- Classification
- k-Nearest Neighbors
- Naive Bayes
- Model Evaluation
- Tools
- **Example**

Example

Example 1

Is it a classification problem?

Problem. For each business question below, decide whether it is a classification problem. If it is, say whether it is **binary** or **multi-class**, and name the class label y and the set of classes $\mathcal{C}$.

(a) Predict next month's revenue of each store.

(b) Predict whether a customer will choose the *Basic*, *Standard* or *Premium* plan.

(c) Split customers into groups with similar purchasing behaviour.

(d) Predict whether a loan will become a bad debt.

Tasks (reason it out yourself):

- Use the three signs of a classification problem (categorical target, labelled history, one class per record).
- For the questions that are *not* classification, name the correct problem group.

Example

Example 2

KNN by hand: one more customer

Problem. Use the 8-customer training set of the KNN example and add one more customer: **C9 = (AppUse 5, Satisfaction 4)**, who **churned**. The new customer is still $N = (5, 5)$.

Tasks:

- Compute $d(N, C9)$ and insert C9 into the sorted distance table.
- Predict the class of N with $k = 3$. Does the prediction change compared with the original example (No)?
- Give $\hat{P}(\text{Yes} \mid N)$ for $k = 3$ and for $k = 5$.
- In one sentence: what does this tell you about how sensitive KNN is to a single new data point?

Example

Example 3

A fraud model with 99.5% accuracy

Problem. A bank tests a fraud-detection model on **10,000 transactions**, of which **0.5%** are fraudulent. The vendor reports an accuracy of **99.5%**.

Tasks:

- How many transactions in the test set are fraudulent?
- Build the confusion matrix of a "model" that labels *every* transaction as legitimate. What are its accuracy, recall and precision?
- Is the reported 99.5% convincing? Which metrics would you ask the vendor for, and why?
- Which error (FP or FN) is costlier for the bank? Which metric should the bank focus on?

Example

Example 4

Naive Bayes by hand: will an online visitor buy?

Problem. An online shop logged 10 past sessions. Predict **Buy** (Yes / No) for new visitors with Naive Bayes.

Session	Device	Source	Returning	Buy
S1	Desktop	Search	Yes	**Yes**
S2	Desktop	Ads	Yes	**Yes**
S3	Mobile	Search	Yes	**Yes**
S4	Desktop	Search	No	**Yes**
S5	Mobile	Social	No	**No**
S6	Mobile	Ads	No	**No**
S7	Mobile	Social	Yes	**No**
S8	Desktop	Social	No	**No**
S9	Mobile	Search	No	**No**
S10	Mobile	Ads	No	**No**

New visitors

- V = (Device = Mobile, Source = Social, Returning = No)
- W = (Device = Mobile, Source = Search, Returning = Yes)

Tasks:

- Build the frequency table of each feature value by class, and the priors $P(\text{Yes})$, $P(\text{No})$.
- Compute score(Yes), score(No) and $P(\text{Yes} \mid V)$ **without** smoothing. What goes wrong?
- Recompute for V with **Laplace smoothing**. Note: *Source* has $K = 3$ values, the other features $K = 2$.
- Compute $P(\text{Yes} \mid W)$ without and with smoothing. Predict the class of V and W.

Example

Example 5

Which model should run the retention campaign?

Problem. Two churn models are tested on the same **500 customers**, of whom **100 churned** (positive class = Churn).

Model A (KNN)	Pred. Yes	Pred. No
Actual Yes	70	30
Actual No	60	340

Model B (NB)	Pred. Yes	Pred. No
Actual Yes	55	45
Actual No	20	380

Business facts

- Every flagged customer receives a voucher costing **0.2 million VND**; a churner who receives it stays.
- Every churner the model misses is lost: **2.0 million VND** of revenue.

Tasks:

- For each model compute accuracy, precision, recall, specificity and F_1.
- Which model wins on accuracy and on F_1?
- Compute the total cost of each model: (missed churners $\times$ 2.0) + (flagged customers $\times$ 0.2).
- Which model should the company deploy? Explain why the answer differs from the accuracy ranking.

Summary

1. **Classification** = supervised learning: from records (x, y), learn a classifier f that assigns each record to *one* class of a finite, predefined set; part of the **PREDICT** group.
2. Two phases, **induction** (learn) and **deduction** (apply); output is a label or a probability; problems are binary, multi-class (or multi-label). Define the label, positive class and horizon first.
3. Always **split train** / **test** (stratified when classes are imbalanced); report only results on unseen data.
4. **KNN:** k nearest neighbours by distance, majority vote; k and **normalization** decide the quality.
5. **Naive Bayes:** $P(C \mid X) \propto P(C) \prod_i P(x_i \mid C)$; independence assumption; **Laplace smoothing** avoids zero probabilities.
6. **Confusion matrix** $\rightarrow$ accuracy, precision, recall, F_1. With a rare class **accuracy can mislead**; choose the metric by the **cost of errors**.

Readings: North, *Data Mining for the Masses*, Ch. 7 · Shmueli et al. (2020), *Data Mining for Business Analytics in Python*, Ch. 5, 7, 8 · Tan et al. (2019), *Introduction to Data Mining*, Ch. 3.

Thank you!