03 · Data Prep & Feature Engineering

EDA in five passes

Pass	pandas	Red flag
shape & types	`df.info()`	numbers stored as `object`
missing	`df.isna().mean()`	> 30% missing; missing only in one class
spread	`df.describe()`	impossible min/max
duplicates & ids	`df.duplicated().sum()`, `df.nunique()`	ids, repeated rows
target	`y.value_counts(normalize=True)`	rare positive class
signal	`df.groupby(target).mean()`	a column that predicts perfectly → leak?

Leak test for every column: **would I know this value at prediction time?**

Missing values

Strategy	When	scikit-learn
drop rows	few, random	`df.dropna(subset=[...])`
median	skewed numbers	`SimpleImputer(strategy="median")`
most frequent	categories	`SimpleImputer(strategy="most_frequent")`
constant	"unknown" is meaningful	`SimpleImputer(strategy="constant", fill_value="unknown")`
+ indicator	gap is informative (MNAR)	`SimpleImputer(add_indicator=True)`
model-based	strong column relationships	`KNNImputer`, `IterativeImputer`

Outliers

- IQR = Q3 − Q1; flag `< Q1 - 1.5·IQR` or `> Q3 + 1.5·IQR`.
- Error → fix or drop. Real extreme → `clip(upper=x.quantile(0.99))`, `np.log1p(x)`, or a tree model.

Encoding & scaling

Tool	Use
`OneHotEncoder(handle_unknown="ignore")`	nominal categorie
`OrdinalEncoder(categories=[[...]])`	truly ordered categorie
target encoding (`TargetEncoder`)	high cardinalit cross-fitted onl
`StandardScaler`	mean 0, std 1
`MinMaxScaler`	range [0, 1], outlier sensitive
`RobustScaler`	median / IQR, outlier-robust

Scale for linear models, k-NN, k-means, SVM, neural nets. Trees don't need it.

Splits & leakage

```
X_tr, X_te, y_tr, y_te = train_test_split(X,
y, test_size=0.2, stratify=y,
random_state=42)
pre = ColumnTransformer([
    ("num", Pipeline([("imp",
SimpleImputer(strategy="median")), ("sc",
tandardScaler())]), num_cols),
    ("cat", Pipeline([("imp",
mputer(strategy="most_frequent")),
                      ("oh",
eHotEncoder(handle_unknown="ignore"))]),
cat_cols),
])
model = Pipeline([("pre", pre), ("clf",
LogisticRegression(max_iter=1000))])
cross_val_score(model, X_tr, y_tr, cv=5,
scoring="roc_auc")
```

Data	Split
classification	`stratify=y` / `StratifiedKFold`
several rows per customer	`GroupKFold`
time-ordered	`TimeSeriesSplit` (train past, test future)

Rules: split first · fit on train only · test set used once · preprocessing inside the Pipeline.