**SelectK\_Classification – Documentation**

**What your code is trying to do:**

1. Load data →
2. One-hot encode categorical columns →
3. Split features (X) & target (y) →
4. Select top-k features (chi-square) →
5. Train/test split + scale →
6. Train many classifiers (LogReg, SVMs, KNN, NB, DT, RF) →
7. Predict on test set →
8. Collect Accuracy & Report →
9. Put results in a small summary table.

**Line-by-line (grouped) explanation**

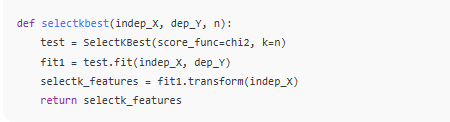
****

 pandas/numpy: For data handling

 scikit-learn: For splitting, scaling, feature selection, models, metrics

 matplotlib/pickle/time aren’t actually used here.

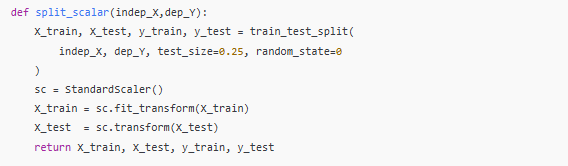
***1. selectkbest***



 Creates a **chi-square** selector and picks the **top n features** with the highest association to the target.

 **Important constraint:** chi2 requires **non-negative features** (e.g., 0/1 dummies, counts). If you feed negatives here, it will error.

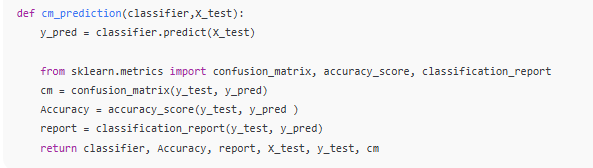
***2. split\_scalar***



 Splits data (75/25).

 **Standardizes** features (zero mean, unit variance) using parameters learned from **training only**, then applies to test. Good practice for SVM/KNN/LogReg.

***3.cm\_prediction***

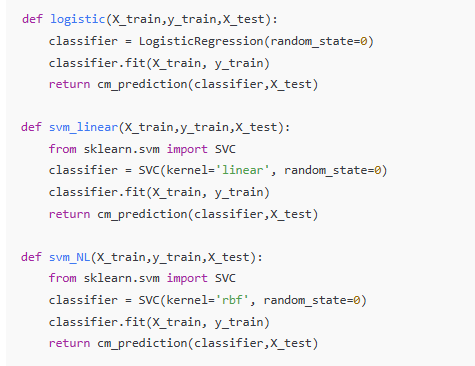


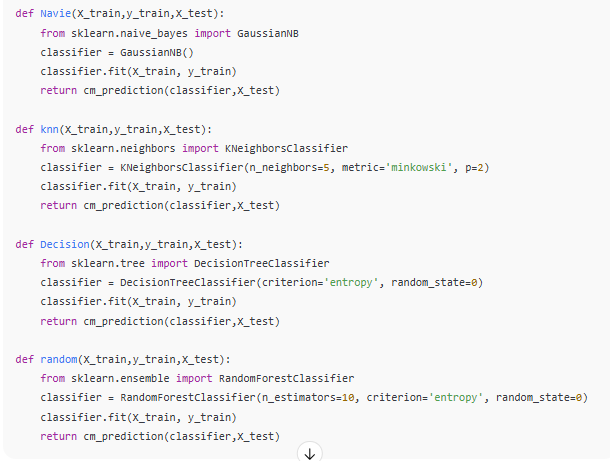
Uses the trained classifier to predict the test set, then computes:

* Confusion matrix
* Accuracy score
* Classification report (precision/recall/F1 by class)

***4.The model trainers***

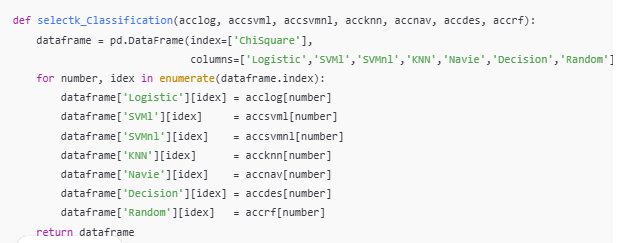
Each function trains a different classifier, then calls cm\_prediction to evaluate:





***5.Build the results table***

* You only create **one row** (“ChiSquare”), then fill each column with the first element of each accuracy list.



***Visual flow (at a glance)***

CSV → DataFrame

→ get\_dummies

→X/y split

→[LEAKAGE] SelectKBest(chi2) on FULL X,y

→ kbest\_X

→ train\_test\_split(kbest\_X, y)

→ StandardScaler (fit on train, transform test)

→ Train many models

→ Predict on X\_test → Accuracy/Report/CM

→ Summary DataFrame