Hands-on: k-NN Regression with Python-powered Train/Test Split

Overview

In this 30 min session you will:

- Learn how to integrate Python code into R using the **reticulate** package.
- Use **scikit-learn**'s `train_test_split()` function (Python) to prepare datasets in R.
- Fit a custom **S3-based k-NN regression** model in R.
- Evaluate predictions and compare observed vs predicted values.

Setup

Download the S3 k-NN implementation script:

```
curl -O https://raw.githubusercontent.com/mmadoliat/WSoRT/refs/heads/main/R/knn_s3_formula.R
```

Source them in your R console:

```
source("knn_s3_formula.R")
```

Import Python packages

```
library(reticulate)
sklearn_model_selection <- import("sklearn.model_selection", convert = TRUE)
```

Custom helper: Split data in Python, return as list of R `data.frames`

```
py_train_test_split <- function(data, test_size = 0.3, seed = 42L) {
  set.seed(seed)
  X <- as.matrix(data[, -1, drop = FALSE])
  y <- data[[1]]

  split <- sklearn_model_selection$train_test_split(
    X, y,
    test_size = test_size,
    random_state = as.integer(seed)
  )

  # unpack results from Python tuple
  X_train <- split[[1]]
  X_test  <- split[[2]]
  y_train <- split[[3]]
  y_test  <- split[[4]]

  train_df <- as.data.frame(cbind(y_train, X_train))
  names(train_df) <- names(data)
  test_df <- as.data.frame(cbind(y_test, X_test))
  names(test_df) <- names(data)

  list(train = train_df, test = test_df)
}
```

Use Python to split mtcars into train/test

```
split_data <- py_train_test_split(mtcars[, c("mpg", "disp", "hp", "wt")], test_size = 0.3)

train_df <- split_data$train
test_df  <- split_data$test

# Fit the S3 k-NN model on train set
model <- knn_s3(mpg ~ disp + hp + wt, train_df, k = 5)
summary(model)

# Predict on test set
```

```r
preds <- predict(model, newdata = test_df, method = "R")
cbind(Observed = test_df$mpg, Predicted = round(preds, 2))
```

Use Python to split mtcars into train/test

```r
split_data <- py_train_test_split(mtcars[, c("mpg", "disp", "hp", "wt")], test_size = 0.3)

train_df <- split_data$train
test_df  <- split_data$test

# Fit the S3 k-NN model on train set
model <- knn_s3(mpg ~ disp + hp + wt, train_df, k = 5)
summary(model)

# Predict on test set
preds <- predict(model, newdata = test_df, method = "R")
cbind(Observed = test_df$mpg, Predicted = round(preds, 2))
```

Tasks

1. Change the train/test split ratio to 0.5 and observe changes in model performance.
2. Modify `py_train_test_split()` to return **NumPy** arrays instead of data frames — verify how this affects R code.
3. (Optional) Replace the R prediction with Python's `sklearn.neighbors.KNeighborsRegressor` and compare results.

Good luck!