SUPERVISED MACHINE LEARNING: REGRESSION

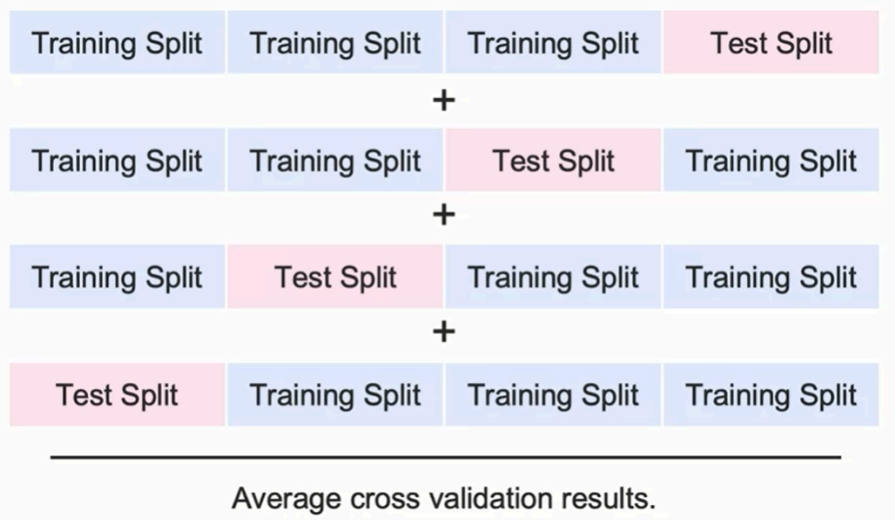
MODULE 3:

CROSS VALIDATION

TABLE OF CONTENTS

K-Fold Cross-Validation..... 2
Types of K-Fold Cross-Validation..... 2
Standard K-Fold..... 2
Stratified K-Fold.....3
Group K-Fold..... 4
Model Complexity and Choice of K.....4
Key Takeaways.....5

K-Fold Cross-Validation



- Cross-validation is a method to **assess how well a model generalizes** to unseen data.
- Instead of one train-test split, the dataset is divided into multiple parts (*folds*).
- The model is trained on some folds and validated on the remaining ones, and this process is repeated to obtain an **average performance metric** that reduces random bias.
- In **K-Fold Cross-Validation**, the dataset is split into *k* folds:
    - Train the model on $k - 1$ folds and validate it on the remaining fold.
    - Repeat *k* times so each fold serves once as validation.
    - The final score is the **average across all folds**.

Types of K-Fold Cross-Validation

Standard K-Fold

- Randomly splits data into *k* folds of equal size.
- Works well for balanced, independent datasets.

```
from sklearn.model_selection import KFold, cross_val_score
from sklearn.linear_model import Ridge

# Define model
ridge = Ridge(alpha=1.0)

# Standard 10-fold CV
kf = KFold(n_splits=10, shuffle=True, random_state=1)
scores = cross_val_score(ridge, X, y, cv=kf,
scoring='neg_mean_squared_error')

print("MSE for each fold:", -scores)
print("Average MSE:", -np.mean(scores))
```

Stratified K-Fold

- Maintains the same class distribution in each fold.
- Suitable for **imbalanced classification problems**.

```
from sklearn.model_selection import StratifiedKFold,
cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_breast_cancer

# Load data
data = load_breast_cancer()
X, y = data.data, data.target

# Define model
model = LogisticRegression(max_iter=5000)

# Stratified 5-fold CV
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
scores = cross_val_score(model, X, y, cv=skf, scoring='accuracy')

print("Accuracy per fold:", scores)
print("Average accuracy:", scores.mean())
```

Group K-Fold

- Keeps **related samples** (e.g., from the same group or person) in the same fold.
- Prevents **data leakage** between training and validation.

```
from sklearn.model_selection import GroupKFold
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import numpy as np

# Create dummy groups
groups=np.array([i//10 for i in range(100)])# 10 samples per group

# Define model and Group K-Fold
model = LinearRegression()
gkf = GroupKFold(n_splits=5)

for fold, (train_idx, val_idx) in enumerate(gkf.split(X, y, groups)):
    model.fit(X[train_idx], y[train_idx])     # Train on groups
    preds = model.predict(X[val_idx])         # Validate on unseen groups
    print(f"Fold {fold+1} MSE:", mean_squared_error(y[val_idx], preds))
```

Model Complexity and Choice of K

Model Type	Description	Common Issue
Low Complexity	Simple, few parameters	Underfitting
High Complexity	Many features, very flexible	Overfitting

- **Smaller k (e.g., 2–3):** higher variance, less stable results.
- **Larger k (e.g., 10–20):** lower variance, more stable but computationally expensive.

Key Takeaways

- **Cross-Validation**
  Evaluates model stability by rotating train-test splits for fair performance estimation.
- **K-Fold Variants**
  - *Standard:* Random splits.
  - *Stratified:* Keeps class proportions.
  - *Group:* Prevents leakage across related samples.