■ Fiche Résumé : Hyperparamètres Tuning pour Modèles de Régression

■ Decision Tree

• `criterion` : Fonction de perte (ex: 'squared_error')
• `max_depth` : Profondeur maximale de l’arbre
• `min_samples_split` : Min d’échantillons pour diviser un nœud
• `min_samples_leaf` : Min d’échantillons dans une feuille
• `max_features` : Nombre max de variables testées par split
• `splitter` : Méthode de division ('best', 'random')

■ Random Forest

• `n_estimators` : Nombre d’arbres
• `max_depth` : Profondeur max
• `max_features` : Nb de variables considérées par split
• `bootstrap` : Utilise l’échantillonnage bootstrap

■ Gradient Boosting

• `n_estimators` : Nombre d’arbres
• `learning_rate` : Taux d’apprentissage
• `subsample` : Fraction d’échantillons utilisés
• `max_depth` : Profondeur des arbres
• `loss` : Fonction de perte

■ AdaBoost

• `n_estimators` : Nombre d’estimateurs faibles
• `learning_rate` : Influence de chaque nouvel arbre
• `loss` : Fonction de perte (ex: 'linear')

■ XGBoost

• `n_estimators`, `learning_rate`, `max_depth`, `subsample`
• `colsample_bytree` : Fraction de variables utilisées
• `gamma` : Réduction des splits peu utiles
• `reg_alpha`, `reg_lambda` : Régularisation L1/L2

■ CatBoost

• `iterations` : Nombre d’arbres
• `learning_rate` : Taux d’apprentissage
• `depth` : Profondeur
• `l2_leaf_reg` : Régularisation L2

■ KNeighbors

• `n_neighbors` : Nb de voisins

• `weights` : Uniforme ou pondéré
• `algorithm` : Méthode de recherche
• `p` : Type de distance (1 = manhattan, 2 = euclidienne)

■ Linear Regression

• `fit_intercept` : Apprend-on l’ordonnée ?
• Régularisation possible : `Ridge`, `Lasso`, `ElasticNet`

■ Conseils de tuning

• Commencer avec `n_estimators`, `max_depth`, `learning_rate`
• Utiliser `RandomizedSearchCV` pour les grands espaces
• Vérifier les scores de validation pour éviter l'overfitting