



Dr. Christian Tsoungui Obama

Fiche mémo ML #1

Votre guide pratique en 7 étapes de l'idée au modèle de régression déployable

Guide fondamental pour aspirants Data Scientists

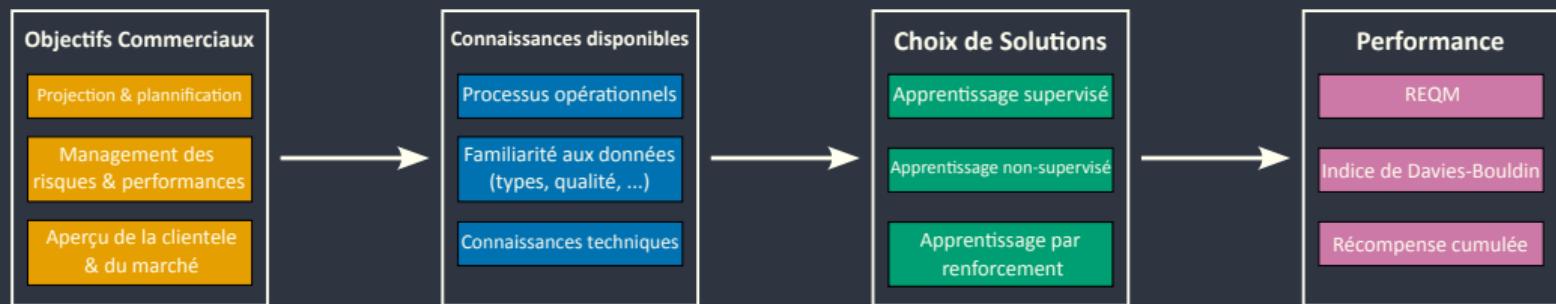
Basé sur
"Machine Learning avec Scikit-learn, Keras & TensorFlow"
par

Aurélien Géron

Étape 1 : Bien définir le problème

- Imaginez que vous collectez des données météorologiques avec un capteur. Vous vous demandez : « Si je connais l'humidité, puis-je prédire la température ? »
- Pensez maintenant à un magasin qui suit son budget publicitaire et ses ventes. Quelqu'un demande : « Si nous modifions notre budget publicitaire, pouvons-nous prédire nos ventes ? »

L'utilisation de modèles d'apprentissage automatique (ML) pour répondre à de telles questions nécessite une définition adéquate du problème et de son contexte.



Feuille de route d'un projet de ML : illustration des points clés à considérer lors de la définition du problème

Les étapes pour construire des modèles de régression sont décrites ci-après.

Étape 2 : Acquérir les données

Sources de données possibles :

- données propriétaires (ex. : bases de données de l'entreprise, données opérationnelles)
- sources en ligne (ex. : web scraping, API)
- dépôts de données publics et ouverts (ex. : Kaggle, UCI)

```
1 import kagglehub
2 from kagglehub import KaggleDatasetAdapter
3
4 # Chargement d'un jeu de donnees depuis Kaggle
5
6 adsSales = kagglehub.dataset_load(
7         KaggleDatasetAdapter.PANDAS,
8         "thorgodofthunder/tvradionewspaperadvertising/versions/1",
9         "Advertising.csv",
10        )
```

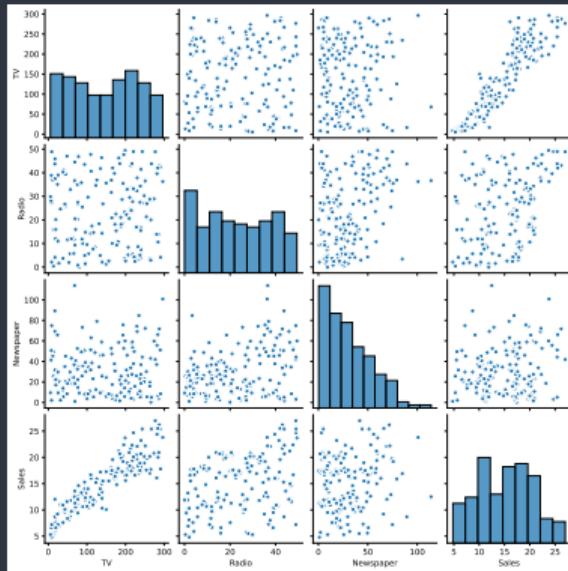
Exemple de code pour importer le jeu de données "Advertising" depuis Kaggle¹

¹ www.kaggle.com/datasets/thorgodofthunder/tvradionewspaperadvertising

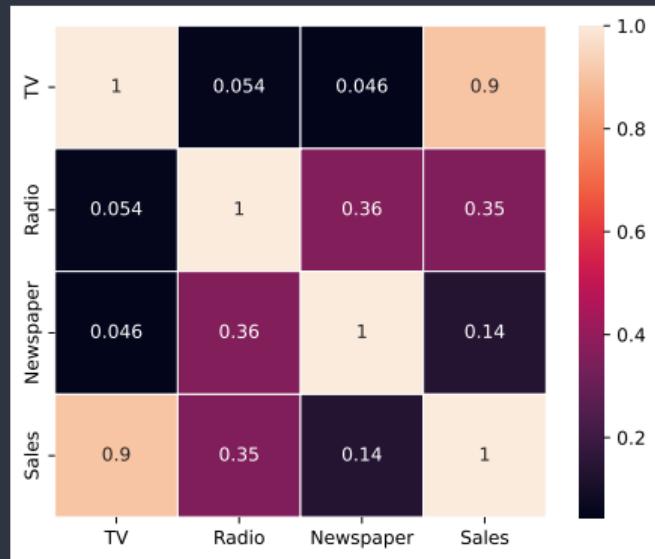
Étape 3 : Explorer les données

Analyse exploratoire des données (AED)

1. diviser les données en ensembles d'entraînement et de test
2. examiner la corrélation linéaire (attributs numériques)
3. transformer les données à l'aide de pipelines (ex: imputation, one-hot-encoding)



Matrice de nuages de points : le nuage de points ressemble à une ligne droite pour les attributs linéairement corrélés



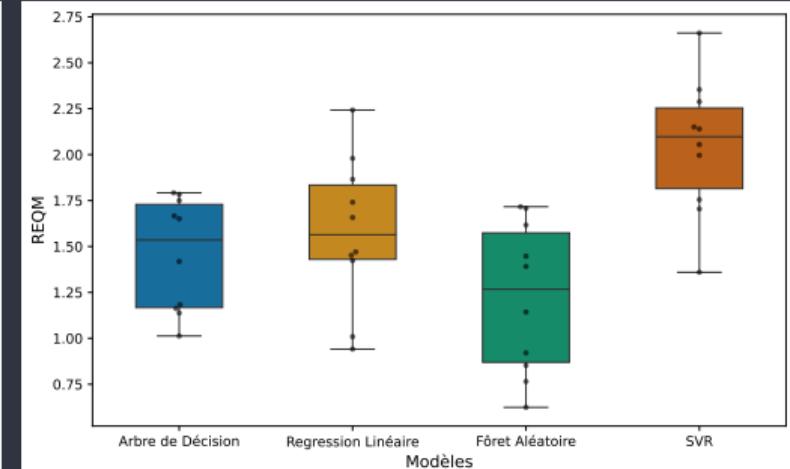
Matrice de corrélation : les valeurs quantifient la force et la direction de la corrélation linéaire

Étape 4 : Entrainer et évaluer les modèles candidats

Quelques modèles candidats pour la régression :

Modèles	Avantages	Inconvénients
Régression linéaire	simple, interprétable, rapide à entraîner	suppose la linéarité, sensible aux données aberrantes et à la multicolinéarité
Régresseur par arbre de décision	modélise la non-linéarité, facile à visualiser et à interpréter	sujet au surajustement, sensible aux changements de données et à l'asymétrie
Régression par machine à vecteurs de support (SVR)	modélise la non-linéarité, efficace pour les données de haute dimension et de taille moyenne	coûteux en calcul pour les larges jeux de données, difficile à interpréter et sensible au choix du noyau

```
1 # Entrainement d'un modèle de regression linéaire
2
3 from sklearn.linear_model import LinearRegression
4 from sklearn.metrics import mean_squared_error
5 import numpy as np
6
7 lin_reg = LinearRegression()
8 lin_reg.fit(adsSales_prepared, adsSales_labels)
9
10 adsSales_predictions = lin_reg.predict(
11     adsSales_prepared)
12 lin_mse = mean_squared_error(adsSales_labels,
13     adsSales_predictions)
14
15 lin_rmse = np.sqrt(lin_mse)
```



Performance des modèles : scores de racine d'erreur quadratique moyenne (REQM – le plus petit est le meilleur)

Étape 5 : Affiner le meilleur candidat (Forêts aléatoires)

Les hyperparamètres du modèle sont affinés pour améliorer ses performances :

1. explorer un large espace avec la recherche aléatoire
2. affiner la recherche avec la recherche par grille
3. utiliser la validation croisée dans les deux cas

```
1 # Réglage des hyperparamètres par recherche par grille
2 from sklearn.model_selection import GridSearchCV
3
4 param_grid = [
5     {'n_estimators': [3, 10, 30, 50], 'max_features': [6, 8, 10]},
6     {'bootstrap': [False], 'n_estimators': [3, 10], 'max_features': [2, 3, 4]},
7 ]
8
9 forest_reg = RandomForestRegressor()
10 grid_search = GridSearchCV(forest_reg, param_grid, cv=5,
11                             scoring='neg_mean_squared_error',
12                             return_train_score=True)
13
14 # Entrainer le modèle pour toutes les combinaisons de paramètres
15 grid_search.fit(adsSales_prepared, adsSales_labels)
16
17 # Sélection du meilleur modèle de la recherche par grille
18 final_model = grid_search.best_estimator_
```

Étape 6 : Évaluer la généralisation du modèle grâce au jeu de données de test

Préparer le jeu de données de test avec le même pipeline que pour les données d'entraînement

```
1 # Definition de le jeu de donnees de test
2 X_test = strat_test_set.drop("Sales", axis=1)
3 y_test = strat_test_set["Sales"].copy()
4
5 X_test_prepared = full_pipeline.transform(X_test)
```

Évaluer la performance du modèle

```
1 # Execution des predictions sur l'ensemble de test
2 final_predictions = final_model.predict(X_test_prepared)
3
4 final_mse = mean_squared_error(y_test, final_predictions)
5 final_rmse = np.sqrt(final_mse)
```

Étape 7 : Sauvegarder le modèle pour un déploiement éventuel

```
1 # Sauvegarde du modèle en tant que fichier joblib
2
3 def save_model(model, name="model", model_path="."):
4     """
5         Sauvegarde le modèle avec un nom donné à un emplacement spécifique.
6
7     Paramètres
8     -----
9     model :
10        Le modèle entraîné.
11     name:
12        Le nom de fichier sous lequel sauvegarder le modèle
13     model_path:
14        Le chemin dans lequel le modèle est sauvegardé
15     Exemple:
16         "./models/"
17
18     """
19     os.makedirs(model_path, exist_ok=True)
20     model_name = name + ".pkl"
21     pkl_path = os.path.join(model_path, model_name)
22     joblib.dump(model, pkl_path)
23
24 MODEL_PATH = os.path.join("models", "adsSales")
25 save_model(final_model, name="Foret_aleatoire_V1.1", model_path=MODEL_PATH)
```



Avez-vous trouvé cela utile ?

- ⌚ Obtenez le notebook complet : » Dépôt GitHub «
- 💬 Commentez ci-dessous : Comment choisissez-vous vos modèles candidats pour la régression ?
- 👤 Suivez-moi : pour la prochaine fiche mémo de cette série.
- ❤️ Aimez ce post : pour l'aider à atteindre plus de monde.

Restons en contact !

 /in/christiantsoungui  @TsounguiChris