**Etapes de création d’un model de Machine**

**En rouge, les méthodes/fonctions utilisées par Pandas ou Numpy**

**En vert, les méthodes/fonctions utilisées par Scikit-Learn**

**Travail sur le jeu de donnée :**

* Etape 1 : Identifier le type de model à utiliser (Classification, Regression)  
  cf : tableau excel analyse algorythme
* Etape 2 : Effectuer le data management (traitement des données, imputation des NaN, features engeneering…)
* Fillna() = remplacement des valeurs manquantes
* Isna() = determine si une valeur est Nan
* Dropna() = supprimer les valeurs manquantes
* Dropt(columns=[]) = supprimer les colonnes présente dans la liste
* Replace({}) = remplacer les valeurs par d’autre (utile dans l’encodage, ou feature engeenering)
* To\_numeric = passer une colonne d’un dataframe en timestamp
* Merge(df1, df2, (LEFT, RIGHT, HOW, ON)) & concat([df1, df2], axis=0ou1) = merging et concatenation
* Apply(function) = permet d’appliquer une fonction sur une valeur d’une colonne (attention, sans parametre)
* Filtre = df[df[‘colonne’] > condition ]
* Value\_counts() = compte le nombre d’occurrence unique
* Df.columns = liste des colonnes du dataframe
* Enumerate = prend deux variable pour boucler, index + valeur

**Travail avant de splitter votre jeu de donnée :**

* Etape 3 : encodage des variables catégorielles (si nécessaire)
* Replace()
* Get\_dummies()
* OrdinalEncoder()
* OneHotEncoder
* Etape 4 : Mise à l’échelle du jeu de donnée (standardisation, normalisation)
* StandardScaler() = standardisation au z score
* MixMaxScaler() = standardisation au min max
* Etape 5 : Evaluer les fréquences des classes (classification)
* Value\_counts() / len(df)
* Réechantillonage si classe déséquilibré (1/3-2/3 début du danger)
  + Sous-echantillonage
  + Sur-echantillonage (SMOTE)
  + Gestion des poids (class\_weight)

**Entrainement du model***(boucle iterative, je test, je recommence, je test, je recommence…):*

* Etape 5 : Sélection des variables explicatives (table de corrélation)
* Df.corr()
* Scipy bibliotheque (pearson, student, fisher, ANOVA…)
* Etape 6 : Instanciation du model

On instancie le model dans une variable

* Etape 7 : Split du jeu de donnée en train et test
* Train\_test\_split(X, y, test\_size=0.3, random\_state=42)

Avec X = features explicative, y = target, test\_size = % donnée test, rs = fixé a un int

* Etape 8 : Entrainement du jeu de donnée sur le train set

Model.fit()

* Etape 9 : Prédiction du jeu de donnée sur le test set
* Regression : model.predict(y\_test, y\_pred)
* Classification : model.predict(y\_test, y\_pred) / model.predict\_proba(y\_test,y\_pred)
* Etape 10 : Evaluation du model
* Regression : r2\_score, mean\_squared\_error, median\_squared\_error …
* Classification : confusion\_matrix, classification\_report
* Etape 11 : Validation croisé du model
* Kfold
* StratifieldKfold

**Autre : Optimisation des paramètres**

* **GridSearchCV : permet de passer une grille de paramètre et d’effectuer des cross validation pour évaluer une combinaison de paramètre sur un model (cv par default=5, param\_grid={}).**

Regarder **:**

**best\_params\_dict = dictionnaire meilleur parametre**

**best\_score\_ =meilleur score**

**best\_index = meilleur score (index de la ligne)**

**best\_estimator = meilleur model (instance du model)**

* **Integration DeepLearning via API roboflow**