Cheat Sheet **Keras :** Modèle de base

Définition

Keras est une bibliothèque open source de Deep Learning permettant la génération de réseaux de neurones artificiels.
Elle permet une expérimentation efficace de ces derniers de par sa simple utilisation et ses nombreuses possibilités de déploiement grâce à son intégration avec la librairie Tensorflow.

Modèles

MLP

Binary Classification

```
from tensorflow.keras.layers import Dense
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

Multi-Class Classification

```
from tensorflow.keras.layers import Dropout
model.add(Dense(512, activation='relu', input_dim=784))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(512, activation='softmax'))
```

Régression

```
model.add(Dense(64, activation='relu', input_dim=nb_features))
model.add(Dense(1))
```

CNN

```
from tensorflow.keras.layers import Activation,Conv2D,
MaxPooling2D,Flatten
model.add(Conv2D(32, (3,3), input_shape=(150,150,3) activation='relu'))
model.add(Conv2D(32, (3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

Preprocessing

Une fois le jeu de données importé sous le format d'un aray numpy, il peut être nécessaire de le préparer avant l'application du modèle.

Sequence Padding

```
from tensorflow.keras.preprocessing import squence
X_train = sequence.pad_sequences(X_train, maxlen)
X_test = sequence.pad_sequences(X_test, maxlen)
```

Ensembles de test et d'entraînement

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size, random_state)
```

Encodage des variables catégorielles
(One-hot-Encoding des variables cibles)

```
from tensorflow.keras.utils import to_categorical
y_train = to_categorical(y_train, num_classes)
y_test = to_categorical(y_train, num_classes)
```

Standardisation, Normalisation des variables

```
from sklearn.model_selection import StandardScaler
scaler = StandardScaler().fit(X_train)
X_train_norm = scaler=transform(X_train)
X_test_norm = scaler_transform(X_test)
```