



ROYAUME DU MAROC  
UNIVERSITE ABDELMALEK ESSAÂDI FACULTE  
DES SCIENCES ET TECHNIQUES  
DEPARTEMENT GENIE INFORMATIQUE



# Rapport

## Projet de Fin de Module.

Réalisé par :

**SAMADI Sohaib**

**BENAMAR Zaid**

Encadré par :

**Pr. Lotfi el AACHAK**

<u>1-Introduction</u> .....	3
<u>2-Les Outils</u> .....	3
<u>3-Description du Projet</u> .....	4
<u>4-Création du Model</u> .....	5
4.1-importation .....	5
4.2-Data Pre-processing.....	5
4.3-Creation d'un model CNN.....	7
<u>5-Model Déploiement</u> .....	7
5.1-backend .....	8
5.2-Front end .....	9
<u>Conclusion</u> .....	11

## 1-Introduction

Dans le monde du sport professionnel, il est souvent important de connaître l'âge des joueurs afin de mieux comprendre leurs capacités actuelles et prévoir leur potentiel à l'avenir. Dans ce contexte, nous avons participé à une compétition Kaggle visant à prédire l'âge d'un joueur en utilisant des données de vitesse et d'accélération. Nous avons utilisé un modèle de réseau de neurones convolutionnels (CNN) en apprentissage profond pour résoudre ce problème de prédiction. Nous avons comparé plusieurs architectures de CNN et avons optimisé nos résultats en ajustant les hyperparamètres et en mettant en œuvre des techniques de traitement des données. Dans ce rapport, nous présenterons notre approche et nos résultats obtenus

## 2-Les Outils



Framework open source écrit en JavaScript qui permet la création d'applications Web et plus particulièrement de ce qu'on appelle des Single Page Applications.



FastAPI est un Framework Python pour développer rapidement des services Web asynchrones, tels que des APIs REST.



Pandas est un outil d'analyse et de manipulation de données open source rapide, puissant, flexible et facile à utiliser, construit sur le langage de programmation Python.

Jupyter est un environnement de développement interactif (IDE) open source utilisé principalement dans le domaine de la science des données. Il permet de créer et de partager des documents qui contiennent du code exécutable.



TensorFlow est une bibliothèque de logiciels open source gratuite destinée à l'apprentissage automatique et à l'intelligence artificielle.



Docker est un outil conçu pour simplifier la création, le déploiement et l'exécution d'applications en utilisant des conteneurs.



Kubernetes est une plate-forme open-source extensible et portable pour la gestion de charges de travail (workloads) et de services conteneurisés.



GitHub est un service d'hébergement Open-Source, permettant aux programmeurs et aux développeurs de partager le code informatique de leurs projets afin de travailler dessus de façon collaborative.

### 3-Description du Projet

L'objectif de ce projet est de développer et déployer une application web de prédiction de l'âge des joueurs de sport basée sur leurs performances physiques (vitesse et accélération).

Nous utiliserons un modèle de réseau de neurones convolutionnels (CNN) en apprentissage profond pour obtenir une prédiction précise. La partie frontend de l'application sera développée avec Angular et la partie backend avec FastAPI. Nous utiliserons Git, Docker et Kubernetes pour suivre les modifications apportées au code, travailler de manière collaborative avec l'équipe et gérer facilement la scalabilité et la disponibilité de l'application.

## 4-Création du Model

### 4.1-importation

Tout d'abord avant de rien faire on doit importer les bibliothèques nécessaires pour un model Deep Learning et aussi définir les sources de données.

```
import matplotlib.pyplot as plt
from pathlib import Path
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
import seaborn as sns
from datetime import datetime, date
from sklearn.preprocessing import MinMaxScaler

df_game = pd.read_csv('../input/nfl-big-data-bowl-2023/games.csv')
df_scout = pd.read_csv('../input/nfl-big-data-bowl-2023/pffScoutingData.csv')
df_player = pd.read_csv('../input/nfl-big-data-bowl-2023/players.csv')
df_play = pd.read_csv('../input/nfl-big-data-bowl-2023/plays.csv')
df_week1 = pd.read_csv('../input/nfl-big-data-bowl-2023/week1.csv')
df_week2 = pd.read_csv('../input/nfl-big-data-bowl-2023/week2.csv')
```

Dans notre cas la majorité de notre travail sera avec **df\_player**.

### 4.2-Data Pre-processing

il est nécessaire de réunir un jeu de données contenant des mesures de vitesse et d'accélération ainsi que l'âge des joueurs pour lesquels ces mesures ont été prises. Il faut ensuite nettoyer et préparer ces données, par exemple en remplaçant les valeurs manquantes ou en normalisant les valeurs des différentes caractéristiques

```

# Extract age from birthday

def age(birthdate):

    today = date.today()
    age = today.year - pd.to_datetime(birthdate).year - ((today.month, today.day) > pd.to_datetime(birthdate).month, pd.to_datetime(birthdate).day)
    return age

df_player['Age'] = df_player['birthDate'].apply(age)

# now after we got age, we no longer need the birthday column
df_player = df_player.drop(['birthDate'], axis=1)

# Checking Missing values
df_player.isna().sum()

```


Bien Sur notre data set contient des valeurs null c'est pour ca on doit les remplir par l'âge moyen de notre dataset

```

# Filling missing values
mean_age = df_player['Age'].mean()
df_player['Age'] = df_player['Age'].fillna(mean_age).round()
df_player.isna().sum()

```

On a rempli 232 valeurs null par l'âge moyen

nflId	0		nflId	0
height	0		height	0
weight	0		weight	0
collegeName	224		collegeName	224
officialPosition	0		officialPosition	0
displayName	0		displayName	0
Age	232		Age	0
dtype: int64		dtype: int64		

### 4.3-Creation d'un model CNN

il faut choisir un type de modèle de machine Learning, comme un réseau de neurones convolutionnels (CNN), et le configurer en spécifiant les hyperparamètres. Le modèle doit être entraîné sur le jeu de données dans notre les données d'entraînement représente 80% de notre dataset.

```
model = keras.Sequential([
    keras.Input(shape=(X_train.shape[1], 1)),
    layers.Conv1D(32, kernel_size=1, padding='same', activation="relu"),
    layers.MaxPooling1D(pool_size=1),
    layers.Conv1D(64, kernel_size=1, padding='same', activation="relu"),
    layers.MaxPooling1D(pool_size=1),
    layers.Flatten(),
    layers.Dropout(0.5),
    layers.Dense(150, activation="relu"),
    layers.Dense(100, activation="relu"),
    layers.Dense(50, activation="relu"),
    layers.Dense(1, activation="relu"),
])

model.summary()
```

## 5-Model Déploiement

Après que notre modèle a été entraîné et évalué, il peut être déployé pour être utilisé dans l'application web. Pour ce faire, il peut être nécessaire de convertir le modèle en un format compatible avec l'application. Dans notre cas, nous avons enregistré notre modèle avec l'extension **.h5**, ce qui signifie qu'il peut être facilement chargé et utilisé avec différentes bibliothèques de deep Learning.

### 5.1-backend

-Après l'installation et la configuration de fastapi, On a ajouté un fichier **cnn.h5** qui contient notre model, et on a créé un autre fichier **cnn.py**. Ce dernier contient la définition de la fonction de la prédiction qui sera appelée lorsque l'API sera utilisée. Cette fonction devra charger le modèle entraîné et prédire l'âge en utilisant les données de vitesse et d'accélération passées en entrée.

```
def predict(a,s) :
    path = os.path.join(MODELS, "cnn.h5")
    my_model = keras.models.load_model(path)

    array = np.array([[a,s]])
    array = np.expand_dims(array, -1)

    pred = my_model.predict(array)
    pred = int(normalizer(pred)[0])

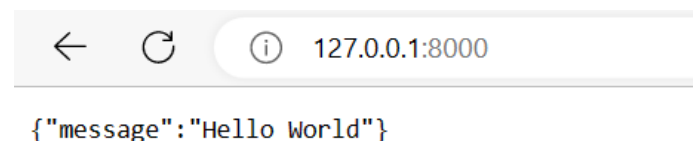
    return pred
```

- apres on a Créé un fichier Python qui définit l'API et la route associée à la prédiction du modèle. Cela est fait en utilisant la décoration @app.post et en spécifiant le chemin de la route.

```
@app.post("/pred",response_model=PredictionOut)
def makepred2(payload: TextIn):
    # a = np.asarray(payload.a).astype('float32')
    # s = np.asarray(payload.s).astype('float32')
    Age = predict(payload.a,payload.s)
    return {"Age": Age}
```

- Configurer uvicorn et Gunicorn pour exécuter l'application web. Cela peut être fait en utilisant les commandes "uvicorn main:app --reload" et "gunicorn main:app" respectivement.

```
(env) PS C:\Users\vinox\Desktop\NewFastApi\src> python -m uvicorn main:app --reload
INFO: Will watch for changes in these directories: ['C:\\Users\\vinox\\Desktop\\NewFastApi\\src']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [8360] using StatReload
INFO: Started server process [15152]
INFO: Waiting for application startup.
INFO: Application startup complete.
```



The screenshot shows a web browser address bar with the URL `127.0.0.1:8000`. Below the address bar, the JSON response `{"message": "Hello World"}` is displayed.



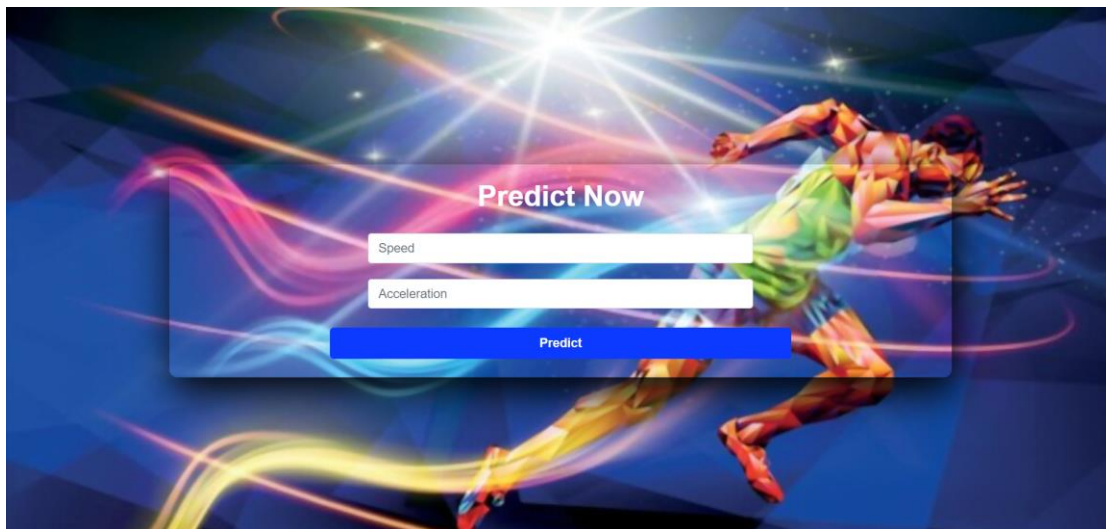
## 5.2-Front end

-Après l'installation de angular Cli en utilisant **npm**, et la création de notre projet et le component il nous reste seulement de configurer le service Angular qui enverra les requêtes HTTP à l'API de prédiction de l'âge, cette requête sera déclencher lors l'utilisateur click sur le Button de prédiction.

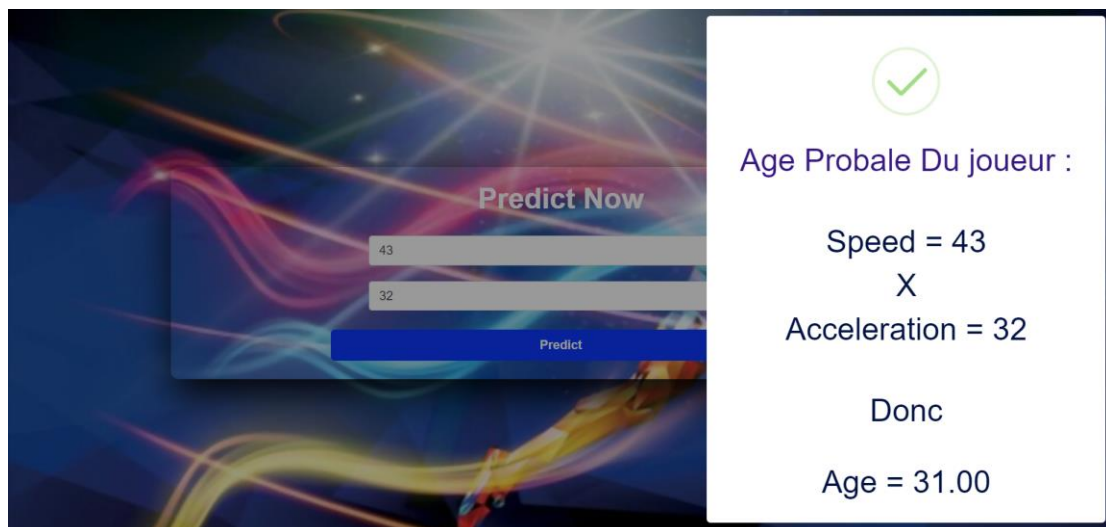
```
submit(): void{  
  this.http.post('http://localhost:8000/pred',this.form.getRawValue()).subscribe(res =>{  
  
    console.log(res);  
    const resu= Object.values(res)
```

Lage sera stocker sur la variable **resu**

-notre Dashboard assez simple contient 2 inputs texte pour le speed et l'accélération et un Button de prédiction

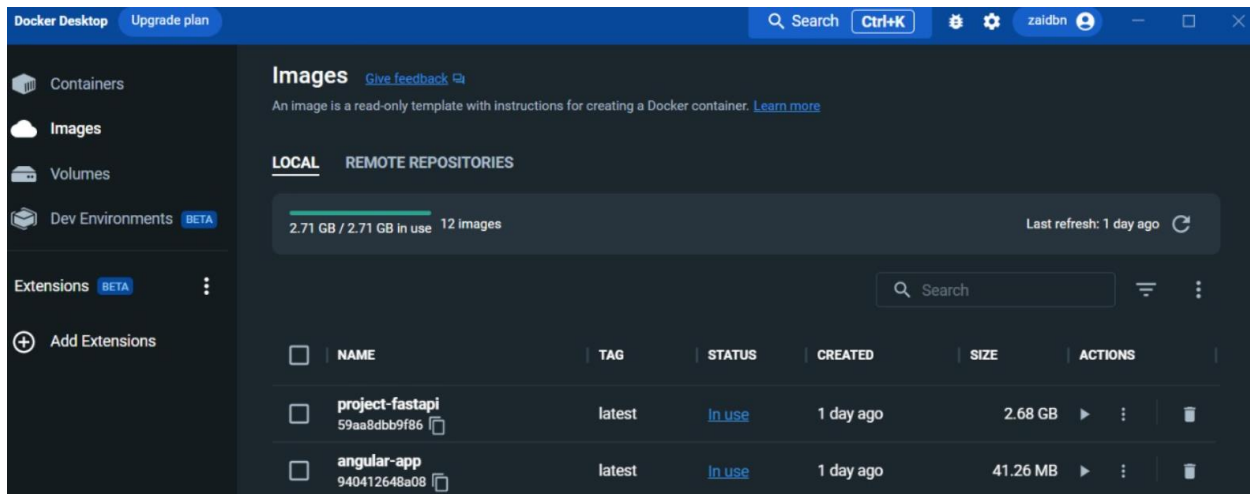


Et voici un petit exemple de la prédiction :



-enfin nous avons utilisé Docker et Kubernetes.

Avec Docker, nous avons créé des images de notre application et de notre modèle de CNN, ce qui nous a permis de déployer facilement ces composants sur différents serveurs ou environnements. Nous avons également utilisé Docker Compose pour définir et exécuter plusieurs conteneurs de manière coordonnée.



En utilisant Kubernetes, nous avons pu gérer l'exécution de nos conteneurs sur un cluster de serveurs et assurer la disponibilité de l'application en cas de panne de l'un des serveurs.

```
D:\Work\Master\S3\Deep Learning\PFM\back&front\frontt\dash>kubectl apply -f deployment.yaml
deployment.apps/angular-deployment created
```

```
D:\Work\Master\S3\Deep Learning\PFM\back&front\frontt\dash>kubectl apply -f deployment.yaml
deployment.apps/angular-deployment unchanged
service/angular-service created
```

```
D:\Work\Master\S3\Deep Learning\PFM\back&front\frontt\dash>kubectl get all
NAME                                READY    STATUS    RESTARTS   AGE
pod/hello-world-65d7f54d4c-rp8tn    1/1      Running   0           2m47s

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)
service/hello-world-service         NodePort      10.100.15.109 <none>       90:307
service/kubernetes                   ClusterIP     10.96.0.1     <none>       443/TCP

NAME                                READY    UP-TO-DATE  AVAILABLE   AGE
deployment.apps/hello-world          1/1      1           1           2m47s

NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/hello-world-65d7f54d4c 1          1          1        2m47s

D:\Work\Master\S3\Deep Learning\PFM\back&front\frontt\dash>kubectl port-forward service/hello-world-service 80:90
Forwarding from 127.0.0.1:80 -> 80
Forwarding from [::1]:80 -> 80
Handling connection for 80
Handling connection for 80
```

## CONCLUSION :

Ce travail a pour objectif de concevoir et faire face aux différentes techniques de la préparation et l'analyse des données et conception des modèles deeplearning, qui permet de prédire l'âge d'un joueur à partir l'accélération et vitesse de ce dernier comme paramètres d'entrée notre modèle.

Pour pouvoir compléter notre projet, nous avons détaillé les différentes étapes de conception, de déploiement dans docker et kubernetes.

Durant ces parties, nous avons présenté la conception détaillée, à la fois, de notre projet, d'autre part nous avons aussi présenté les tâches qui a été accompli pour chaque partie, ensuite, j'ai présenté quelques captures d'écran montrant le bon fonctionnement du notre projet.