Prédire le cours de la bourse

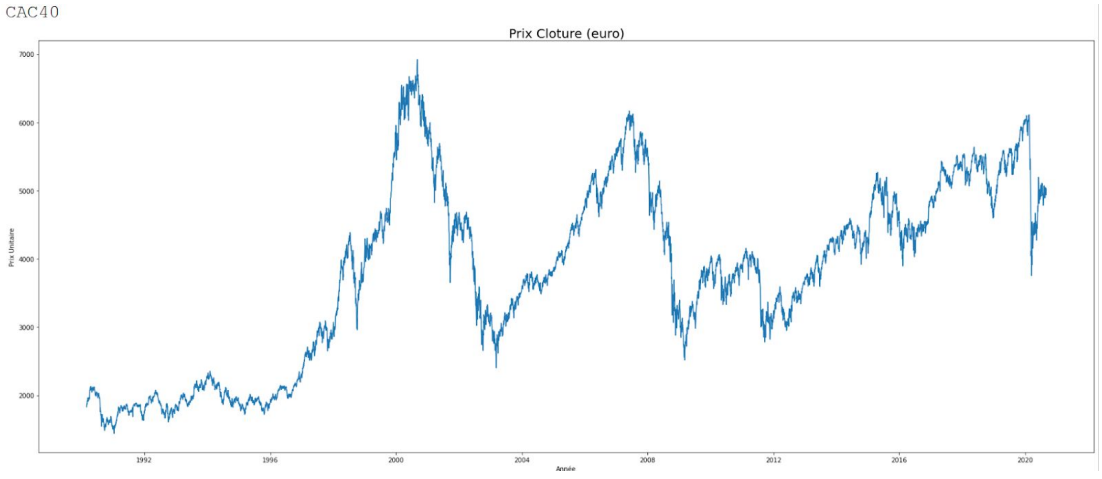
Introduction :

Le projet a été réalisé dans le cadre scolaire lors du cours de modélisations mathématiques. Le but du cours était de réaliser un projet mettant en œuvre une IA.

Nous avons choisi ce projet parmi une liste de projets proposée par M. Samir Chafik. L'objectif de ce projet est de prédire le cours de la bourse pour le Cac 40.

Nous avons choisi ce sujet car il nous a paru intéressant et utile vu qu'il s'agit de prédire des données dans le temps .
La problématique étant comment prédire des valeurs dans le temps et donc comment prédire des séries temporelles(Forecasting).
Pour ce faire, nous avons exploré 2 possibilités, le modèle LSTM (La mémoire à long terme) et le modèle SARIMAX (Seasonal Autoregressive Integrated Moving Average Exogenous model).

Les séries temporelles servent à utiliser un modèle donné qui est basé sur des données temporelles pour prédire les valeurs temporelles futures en fonction des valeurs qu'on leur a fournies, comme par exemple la prédiction du chiffre d'affaires d'une entreprise, ou dans notre cas, la prédiction du cours d'une bourse.



Description des modèles utilisés :

Nous avons décidé d'utiliser un notebook pour ce projet car cela nous permet de faciliter la gestion des erreurs. Ici, c'est la dernière cellules exécutée qui va finalement démarrer la partie principale du projet, car elle contient le menu.

Le menu propose d'afficher le dataset Cac40, de prédire la moyenne mobile à l'aide de SARIMAX, ou de prédire plus précisément le cours du Cac 40 à l'aide de LSTM. Le dernier choix proposé nous permet de quitter le programme.

```
#########################################################################
#                                  MENU                                 #
# 1. Afficher Cac40                                                     #
# 2.SARIMAX Prediction                                                  #
# 3.LSTM Prediction                                                     #
# 4.Exit/Quit                                                           #
#########################################################################
```

LSTM :

La mémoire à long terme (LSTM) est une architecture considérée comme une extension du réseau neuronal récurrent artificiel (RNN) utilisée dans le domaine de l'apprentissage profond . Il est notamment utilisé lors de la réalisation de prévisions basées sur des données de séries temporelles (forecasting).

Une fois sélectionné la prévision LSTM, le programme utilise d'abord la fonction "LSTM_pred". En effet, cette fonction est découpée en trois parties :

- La première est la préparation des données. La cellule va d'abord rentrer les données du CSV dans une matrice, pour ensuite normaliser les données de cette matrice afin d'accélérer la vitesse d'exécution de la cellule, ainsi que la performance, grâce au MinMaxScaler. La cellule va ensuite préparer le dataset en séparant le jeu de données en deux parties : celle vouée à l'entraînement et celle vouée à l'évaluation.
- La deuxième est l'entraînement. La cellule va utiliser la fonction "Sequencial" puis intégrer le LSTM. Enfin, elle va lancer l'entraînement avec un ".fit" sur la partie du dataset voué à l'entraînement.

```
#LSTM model
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(50, return_sequences= False))
model.add(Dense(25))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')

#Entrainement du model
model.fit(x_train, y_train, batch_size=1, epochs=1)
```

- La troisième est l'évaluation des données. La cellule va récupérer les données du model entraîné, puis va l'utiliser pour lancer la prédiction, grace au "model. predict".

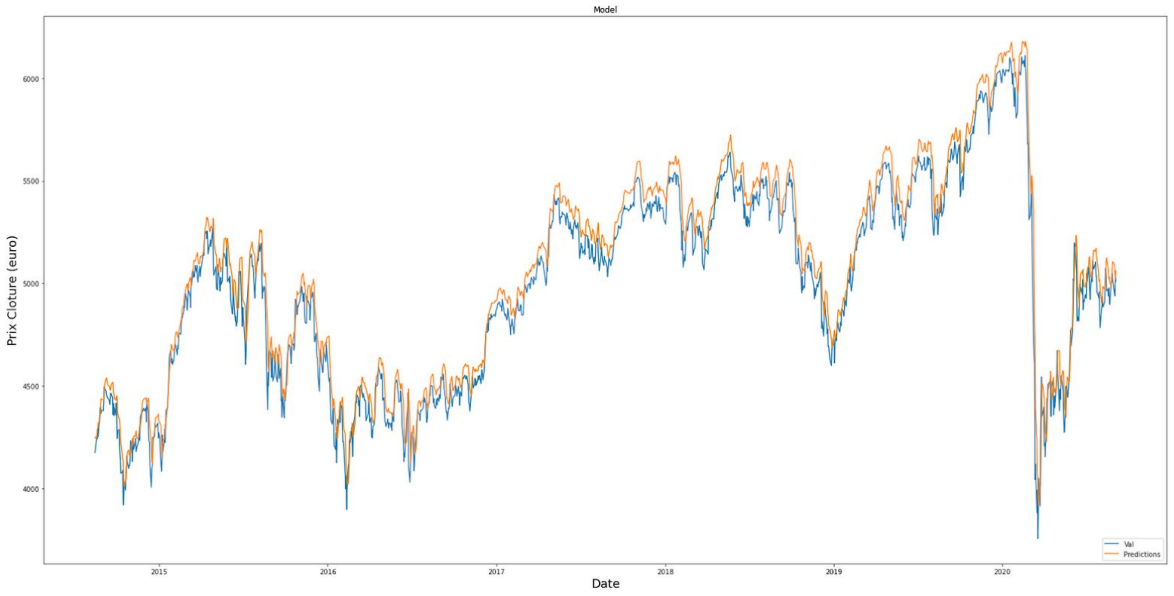
```
#Creation du Dataset servant à l'evalutaion de l'entraînement
test_data = scaled_data[training_data_len - 60: , :]
#Création des deux dataset x_test et y_test séparant le test_data en deux
x_test = []
for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])
y_test = close[training_data_len:, :]

# Convertion numpy array
x_test = np.array(x_test)

# Reshape du  data
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1 ))

# Evaluation et récuperation du jeu de donée prédites
global predictions
predictions = model.predict(x_test)
```

Les résultats :



SARIMAX :

La moyenne mobile autorégressive intégrée (Seasonal Autoregressive Integrated Moving Average Exogenous model en Anglais) ou appeler SARIMAX et un dérivé du modèle ARIMA qui permet de faire de la prédiction sur des séries temporelle en déterminants des tendances. Le modèle SARIMAX prend en compte en plus la saisonnalité des tendances . On a choisi d'utiliser un modèle de plus, car on estimait que l'utilisation des courbes de tendances (appelées aussi moyennes mobiles), était une thématique intéressante, car elles sont beaucoup utilisées dans le monde du trading.

Une fois sélectionné la prévision SARIMAX, le programme utilise d'abord la fonction "SARIMAX_pred". En effet, cette fonction est elle aussi découpée en trois parties (mais plus petites) :

- Une récupération des données.
- L'entraînement de celles-ci.
- Enfin, l'évaluation des données entrainées.

```
#Prediction utilisant SARIMAX
def sarimax_pred():
  y_train = Cac40.iloc[:training_data_len]
  y_test = Cac40.iloc[training_data_len:]
  sarima = SARIMAX(y_train, order=(1, 1, 1), seasonal_order=(1, 1, 1, 45),mle_regression=True)\
  .fit(disp=False)
  y_sarima = sarima.predict(6000,7738,typ='levels')
  y_sarima.index = y_test.index
  mse(y_sarima,y_test)
```

La limite de ce modèle SARIMAX, est le temps de prédiction. Il peut prendre jusqu'à 30 minutes. Ce qui est loin de la prédiction LSTM qui est estimée à 3 minutes au maximum.