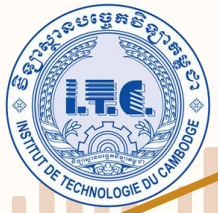


Institute of Technology of Cambodia
Department of Applied Mathematics and Statistics

Currency Exchange Rate Prediction

Group 1

KOUM Soknan
e20211754

PAV Limseng
e20211548

PEL Bunkhloem
e20201314

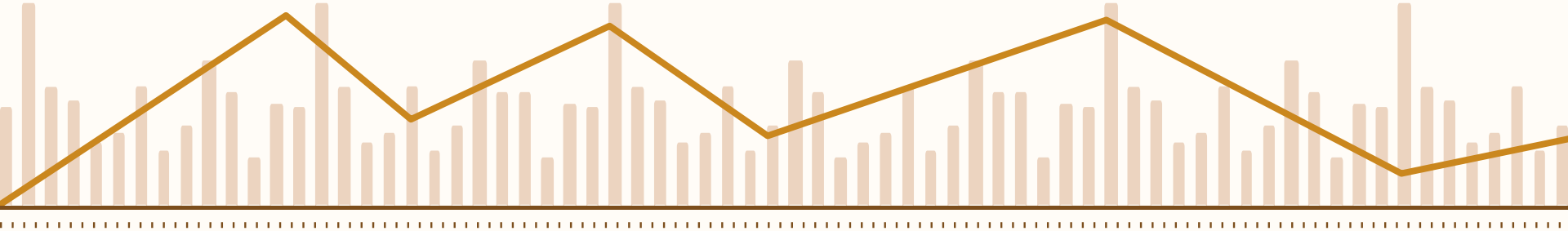
MA Ousa
e20210359

KHON Khengmeng
e20210176

Why do we chose this topic?

We chose **Currency Exchange Rate Prediction** due to its real-world impact and the complexity of the problem. It's crucial for **businesses** and **individuals** in **international trade** or **travel**. The project provides an opportunity to apply data science skills to a multifaceted problem influenced by various factors. The potential outcome could be a tool aiding in **financial planning** and **strategy**. Beside that we also want to know the **Trend** and the **Behavior** on the rate between the 2 currency. Click here to view our notebook.

Table of contents



1

Data Collection

yahoo! finance

Three step to get the Dataset

Yahoo Finance

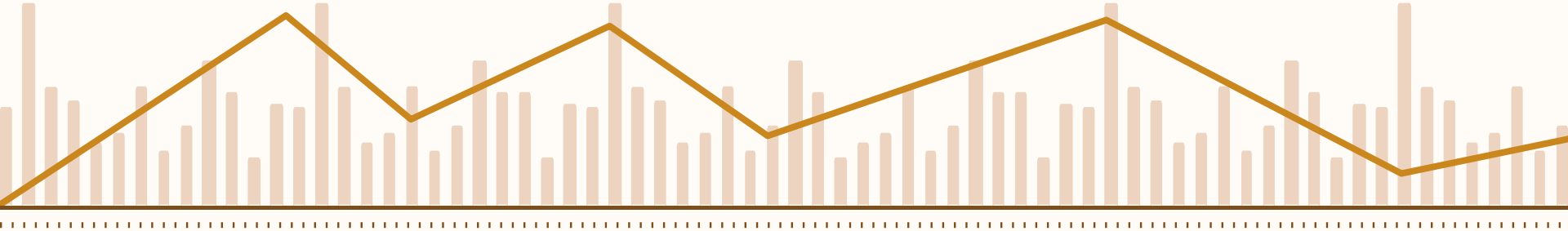
Go to Yahoo Finance and Search “USD/KHR(KHR=X)” or click this link.

Navigation

After landing on Yahoo Finance, navigate to Historical Data.

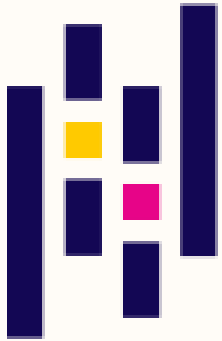
Download

Select the criteria you want then click on the download button.



2

Data Preprocessing



Import Data and Necessary Library

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

sns.set() # setting seaborn default for plots
plt.style.use('ggplot')

data = pd.read_csv("https://raw.githubusercontent.com/PLSeng/MyPage/main/web/assets/KHR%3DX.csv")
data.head()
```

Out[ ]:

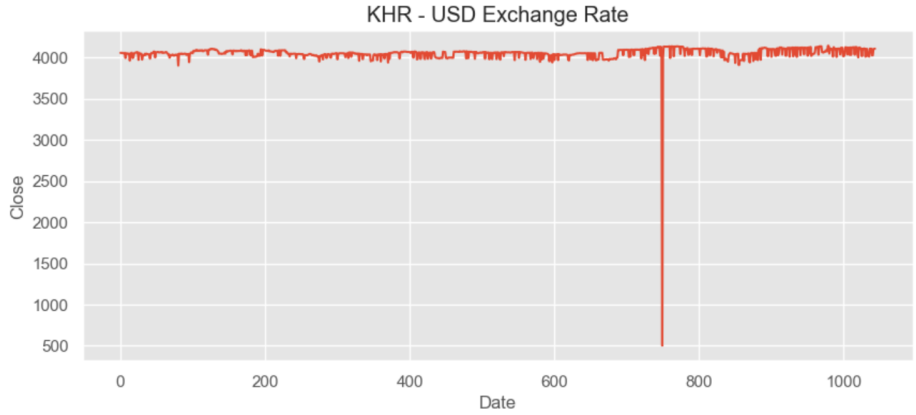
	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-12-09	3992.131348	4060.0	3992.053223	4055.0	4055.0	0
1	2019-12-10	3995.426758	4055.0	3995.426758	4055.0	4055.0	0
2	2019-12-11	3992.499023	4055.0	3989.313232	4053.0	4053.0	0
3	2019-12-12	3968.466309	4055.0	3968.466309	4055.0	4055.0	0
4	2019-12-13	3962.562256	4050.0	3962.562256	4051.0	4051.0	0

Import Data and Necessary Library

In [ ]:
```
data.shape
```

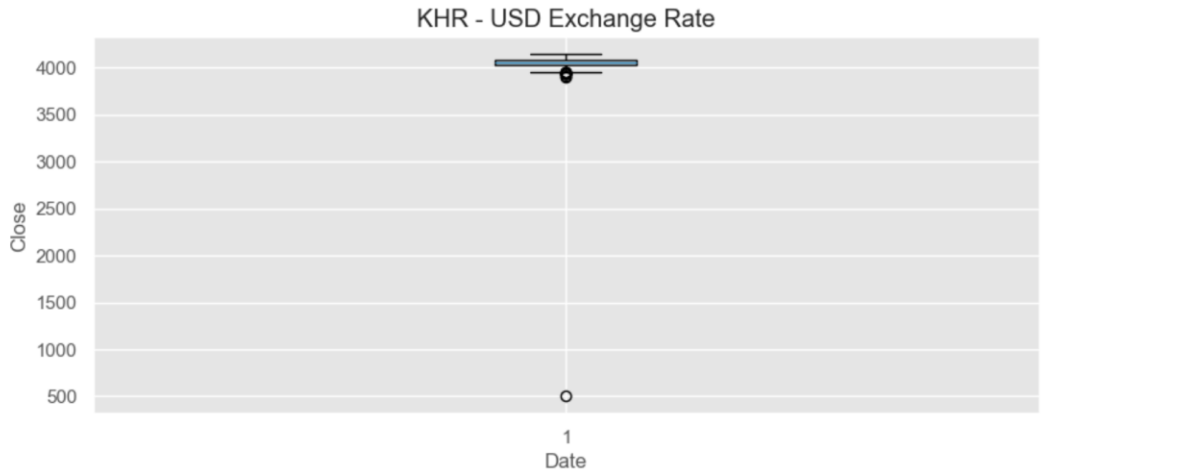
Out[ ]: (1044, 7)

In [ ]:
```
plt.figure(figsize=(10, 4))
plt.title("KHR - USD Exchange Rate")
plt.xlabel("Date")
plt.ylabel("Close")
plt.plot(data["Close"])
plt.show()
```



Plot Boxplot to see outliers

In [ ]:
```
# plot boxplot
plt.figure(figsize=(10, 4))
plt.title("KHR - USD Exchange Rate")
plt.xlabel("Date")
plt.ylabel("Close")
plt.boxplot(data["Close"])
plt.show()
```

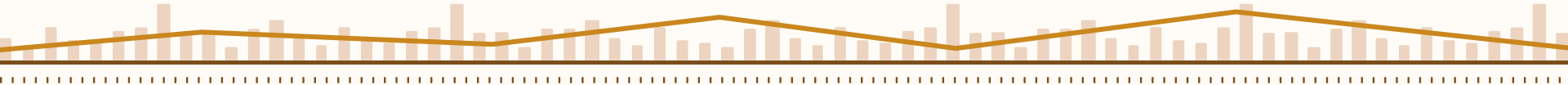


Drop Empty Column

In [ ]:
```
# preprocessing

# drop Volume column
data = data.drop(['Volume'], axis=1)
data.head()
```

Out[ ]:

	Date	Open	High	Low	Close	Adj Close
0	2019-12-09	3992.131348	4060.0	3992.053223	4055.0	4055.0
1	2019-12-10	3995.426758	4055.0	3995.426758	4055.0	4055.0
2	2019-12-11	3992.499023	4055.0	3989.313232	4053.0	4053.0
3	2019-12-12	3968.466309	4055.0	3968.466309	4055.0	4055.0
4	2019-12-13	3962.562256	4050.0	3962.562256	4051.0	4051.0



Drop Empty Column

In [ ]:
```
# preprocessing

# drop Volume column
data = data.drop(['Volume'], axis=1)
data.head()
```

Out[ ]:

	Date	Open	High	Low	Close	Adj Close
0	2019-12-09	3992.131348	4060.0	3992.053223	4055.0	4055.0
1	2019-12-10	3995.426758	4055.0	3995.426758	4055.0	4055.0
2	2019-12-11	3992.499023	4055.0	3989.313232	4053.0	4053.0
3	2019-12-12	3968.466309	4055.0	3968.466309	4055.0	4055.0
4	2019-12-13	3962.562256	4050.0	3962.562256	4051.0	4051.0

Drop Outliers

In [ ]:
```
# find outlier of each column except date and remove the outlier
Q1 = data['Open'].quantile(0.25)
Q3 = data['Open'].quantile(0.75)
IQR = Q3 - Q1
# set outlier to null
data['Open'] = np.where(data['Open'] > (Q3 + 1.5 * IQR), None, data['Open'])
data['Open'] = np.where(data['Open'] < (Q1 - 1.5 * IQR), None, data['Open'])

Q1 = data['High'].quantile(0.25)
Q3 = data['High'].quantile(0.75)
IQR = Q3 - Q1
# set outlier to null
data['High'] = np.where(data['High'] > (Q3 + 1.5 * IQR), None, data['High'])
data['High'] = np.where(data['High'] < (Q1 - 1.5 * IQR), None, data['High'])

Q1 = data['Low'].quantile(0.25)
Q3 = data['Low'].quantile(0.75)
IQR = Q3 - Q1
# set outlier to null
data['Low'] = np.where(data['Low'] > (Q3 + 1.5 * IQR), None, data['Low'])
data['Low'] = np.where(data['Low'] < (Q1 - 1.5 * IQR), None, data['Low'])

Q1 = data['Close'].quantile(0.25)
Q3 = data['Close'].quantile(0.75)
IQR = Q3 - Q1
# set outlier to null
data['Close'] = np.where(data['Close'] > (Q3 + 1.5 * IQR), None, data['Close'])
data['Close'] = np.where(data['Close'] < (Q1 - 1.5 * IQR), None, data['Close'])

Q1 = data['Adj Close'].quantile(0.25)
Q3 = data['Adj Close'].quantile(0.75)
IQR = Q3 - Q1
# set outlier to null
data['Adj Close'] = np.where(data['Adj Close'] > (Q3 + 1.5 * IQR), None, data['Adj Close'])
data['Adj Close'] = np.where(data['Adj Close'] < (Q1 - 1.5 * IQR), None, data['Adj Close'])
```

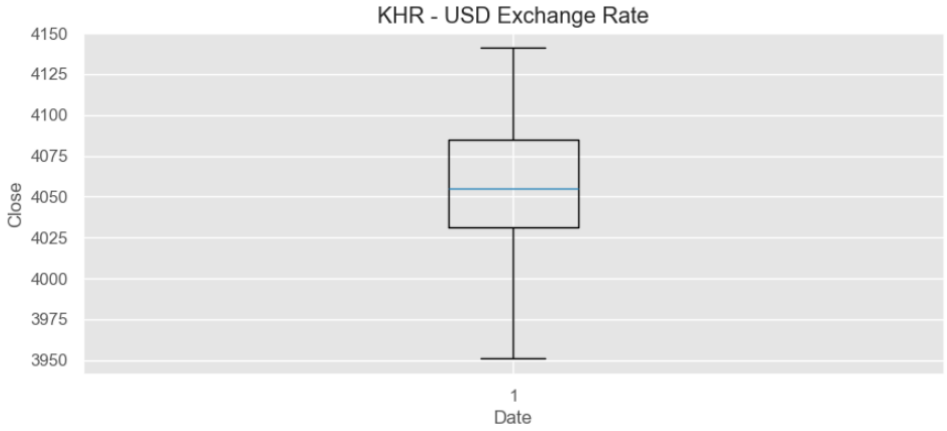
In [ ]:
```
data.isnull().sum()
```

Out[ ]:
```
Date          0
Open         13
High         16
Low           4
Close        11
Adj Close    11
dtype: int64
```

Fill Empty cell with min value

In [ ]:
```
# fill null value with min value
data['Open'].fillna(data['Open'].min(), inplace=True)
data['High'].fillna(data['High'].min(), inplace=True)
data['Low'].fillna(data['Low'].min(), inplace=True)
data['Close'].fillna(data['Close'].min(), inplace=True)
data['Adj Close'].fillna(data['Adj Close'].min(), inplace=True)
```

In [ ]:
```
# plot boxplot again
plt.figure(figsize=(10, 4))
plt.title("KHR - USD Exchange Rate")
plt.xlabel("Date")
plt.ylabel("Close")
plt.boxplot(data["Close"])
plt.show()
```

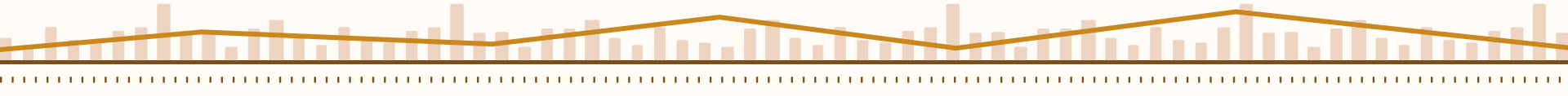


Descriptive Statistics

In [ ]:
```
data.describe()
```

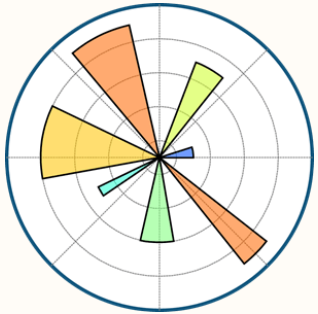
Out[ ]:

	Open	High	Low	Close	Adj Close
count	1044.000000	1044.000000	1044.000000	1044.000000	1044.000000
mean	3996.967278	4055.307647	3991.401142	4053.830659	4053.830659
std	31.126075	36.957627	28.416155	42.866399	42.866399
min	3915.555176	3967.952637	3908.131104	3951.013184	3951.013184
25%	3976.076599	4035.000000	3970.887512	4031.000000	4031.000000
50%	3994.783081	4055.000000	3988.388428	4055.000000	4055.000000
75%	4019.030213	4080.000000	4014.008545	4085.000000	4085.000000
max	4083.958252	4141.000000	4068.570313	4141.000000	4141.000000



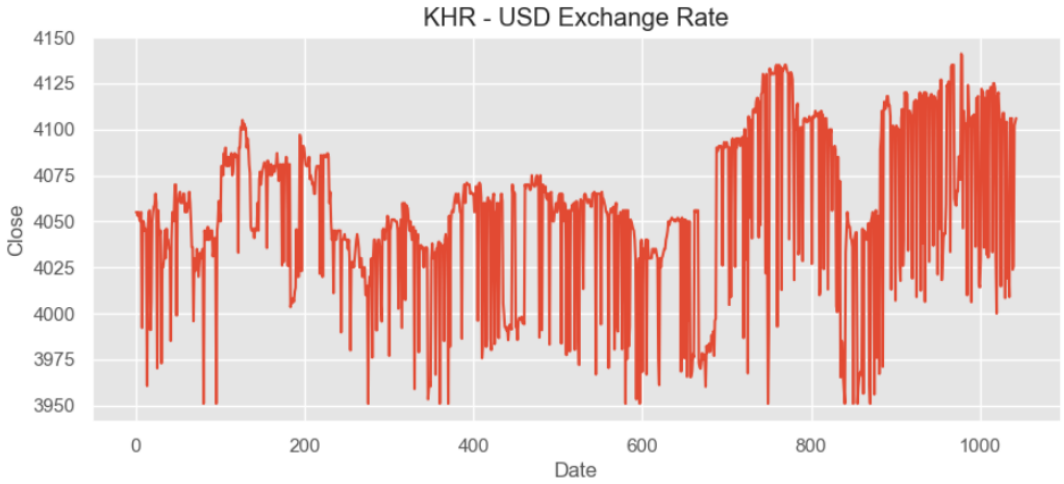
3

Data Visualisation



Plot the time series graph

In [ ]:
```
# plot line graph for ['Close']
plt.figure(figsize=(10, 4))
plt.title("KHR - USD Exchange Rate")
plt.xlabel("Date")
plt.ylabel("Close")
plt.plot(data["Close"])
plt.show()
```

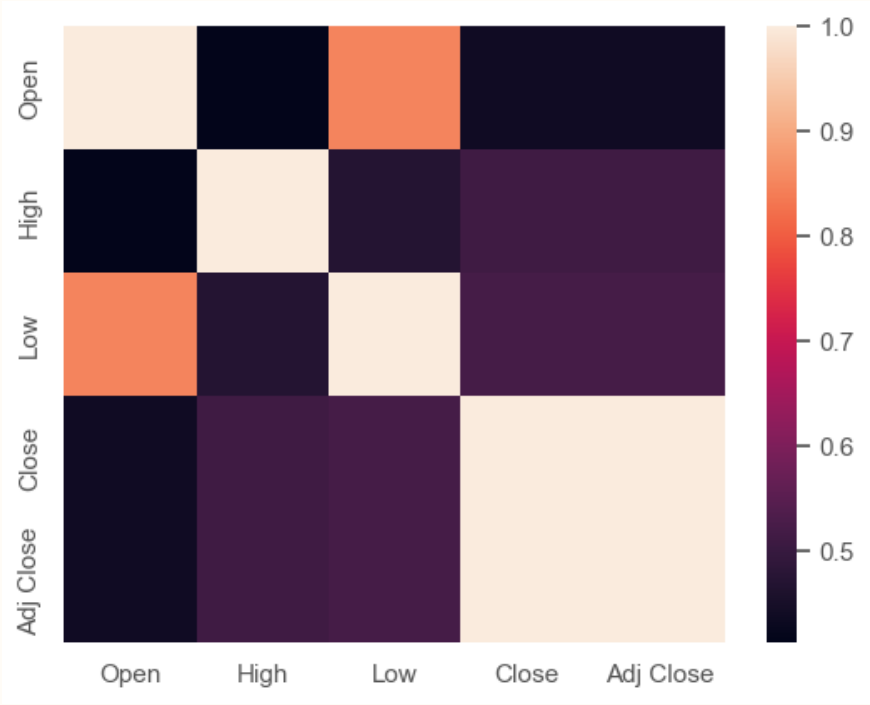


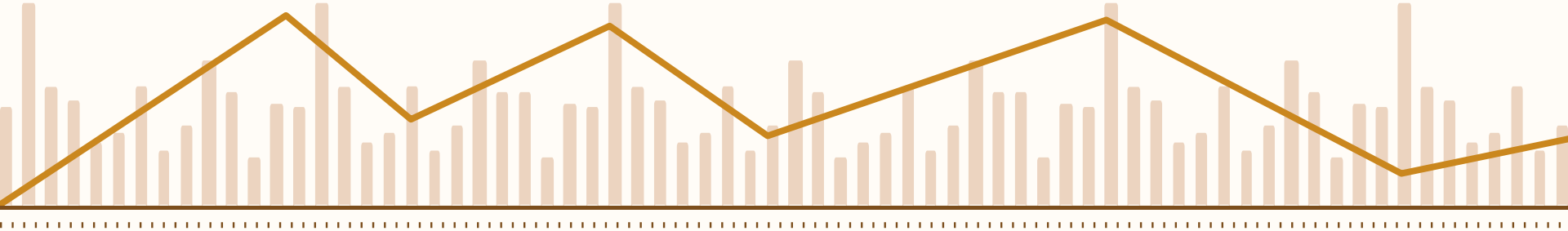
Plot a heatmap to see the correlation

In [ ]:

```
# Select all columns except the first one
data_for_corr = data.iloc[:, 1:]

# Calculate the correlation matrix
corr_matrix = data_for_corr.corr()
print(corr_matrix)

# Create a heatmap
sns.heatmap(corr_matrix)
plt.show()
```

Plot a heatmap to see the correlation





4

Data Analysis

Scikit-Learn

train_test_split

train_test_split in **scikit-learn** splits data into random train and test subsets. It takes arrays as inputs and you can specify the size of the test and train datasets. It also allows for reproducible output and stratified splitting.

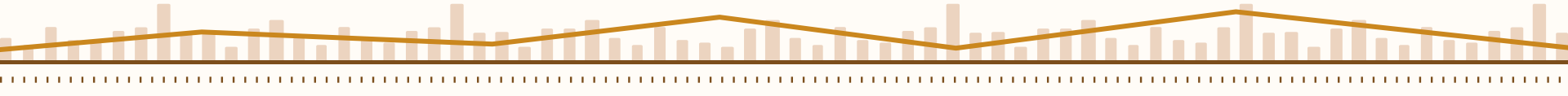
DecisionTreeRegressor

DecisionTreeRegressor in **scikit-learn** is a decision tree regressor. It splits data into train and test subsets based on various parameters like **criterion**, **splitter**, **max_depth**, etc. It's used for creating **models** that **predict** the value of a **target variable**.

Regression

In [ ]:
```
x = data[["Open", "High", "Low"]]
y = data["Close"]
x = x.to_numpy()
y = y.to_numpy()
y = y.reshape(-1, 1)
```

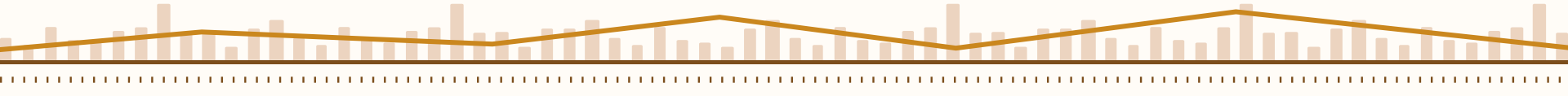
In [ ]:
```
# Predict the rate for the next 7 days
from sklearn.model_selection import train_test_split

# Remove rows with missing values
data.dropna(inplace=True)


# Split the data into training and testing sets
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2, random_state=4

from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor()
# Train the model
model.fit(xtrain, ytrain)
# Make predictions
ypred = model.predict(xtest)
```

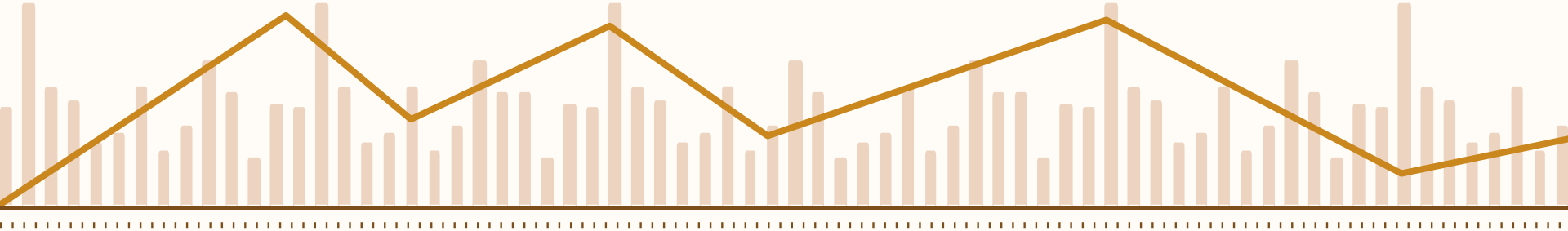


Prediction

In [ ]:
```
data = pd.DataFrame(data={"Predicted Rate": ypred.flatten()})
print(data.head(7))
```

```
   Predicted Rate
0     4043.325928
1     4025.000000
2     4060.000000
3     4052.000000
4     4050.000000
5     4036.320068
6     4046.296631
```



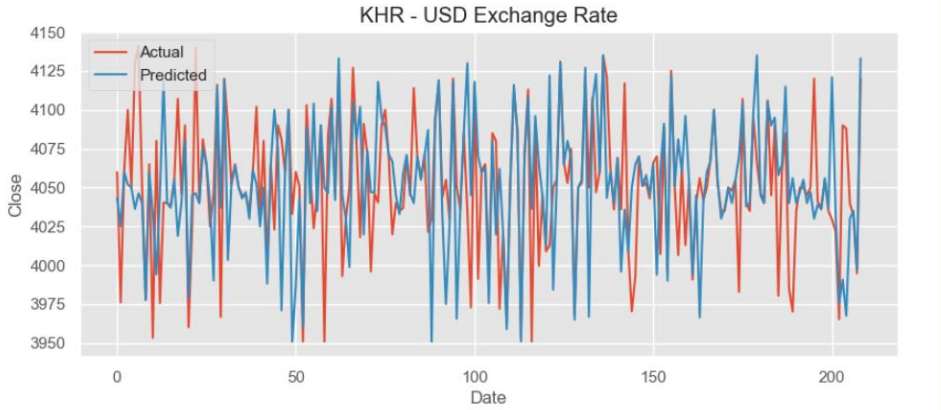


5

Accuracy Rate

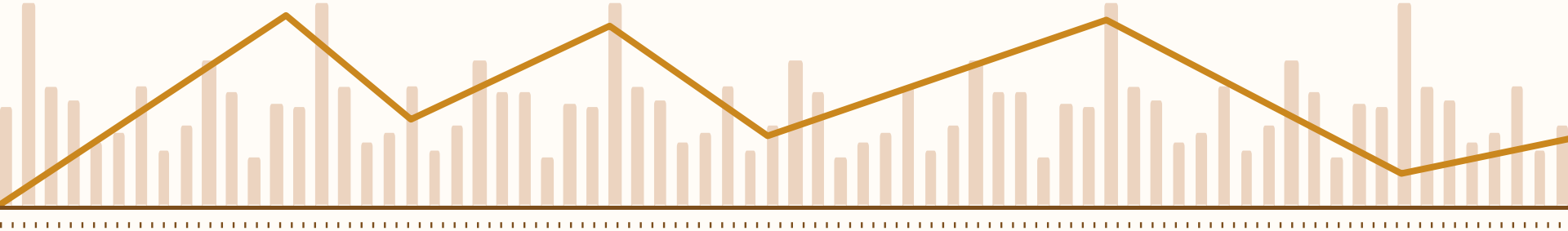
Trendline between Actual and Predicted Value

In [ ]:
```
# Plot the results
plt.figure(figsize=(10, 4))
plt.title("KHR - USD Exchange Rate")
plt.xlabel("Date")
plt.ylabel("Close")
plt.plot(ytest, label="Actual")
plt.plot(ypred, label="Predicted")
plt.legend()
plt.show()
```



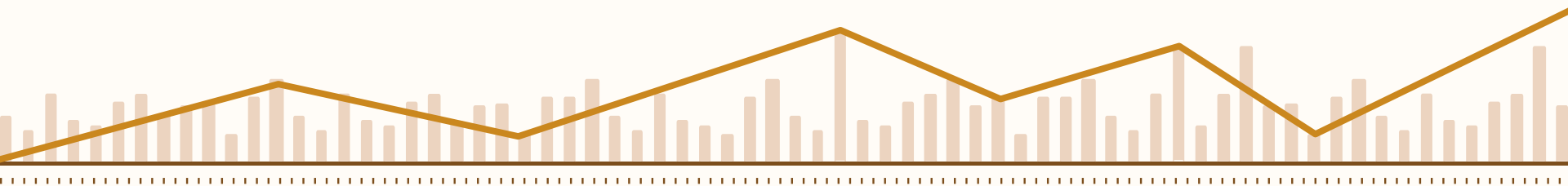
Get the accuracy rate

```
MAE: 41.36138803773583
MSE: 3526.4626259763036
RMSE: 59.38402669048558
R2: 0.7669175989842307
Adjusted R2: 0.8750962274934693
```



6

Conclusion



Thanks!

Do you have any questions?