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 see our notebook.

Table of contents

1

Data Collection

2

Data Preprocessing

3

Data Visualisation

4

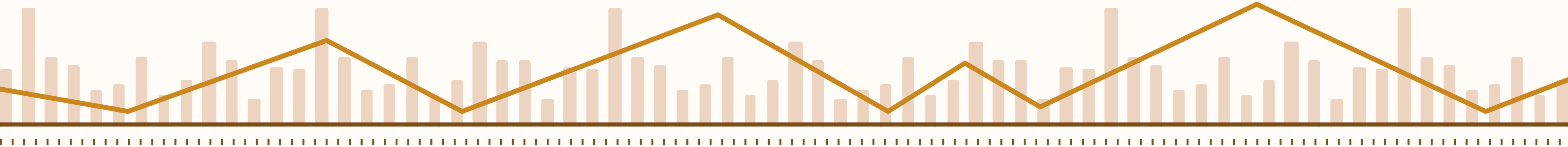
Data Analysis

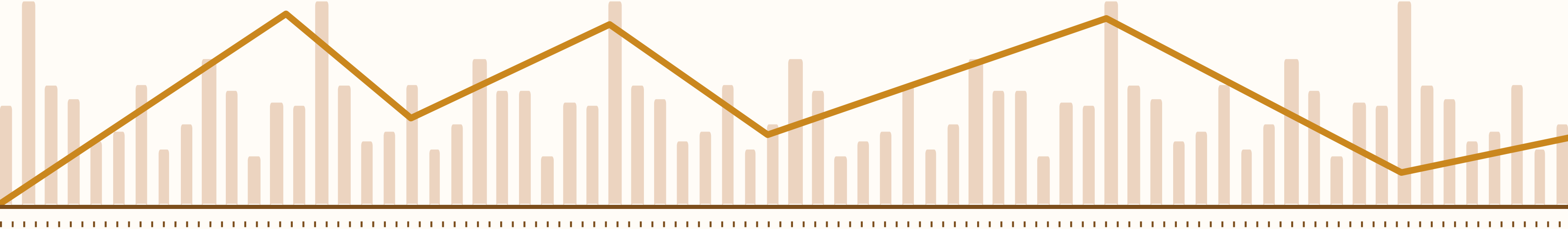
5

Accuracy Rate

6

Conclusion





1

Data Collection

yahoo! finance

Three step to get the Dataset

Yahoo Finance

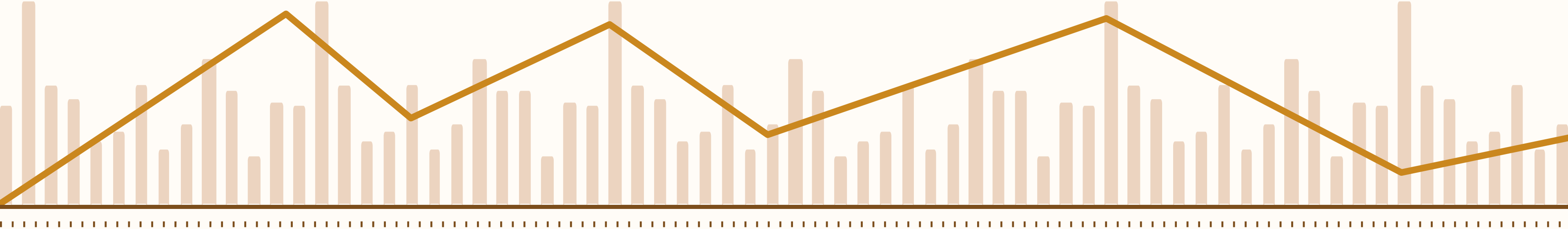
Go to Yahoo Finance and Search “USD/KHR(KHR=X)” or click this link.

Navigation

After landing on Yahoo Finance, navigate to Historical Data.

Download

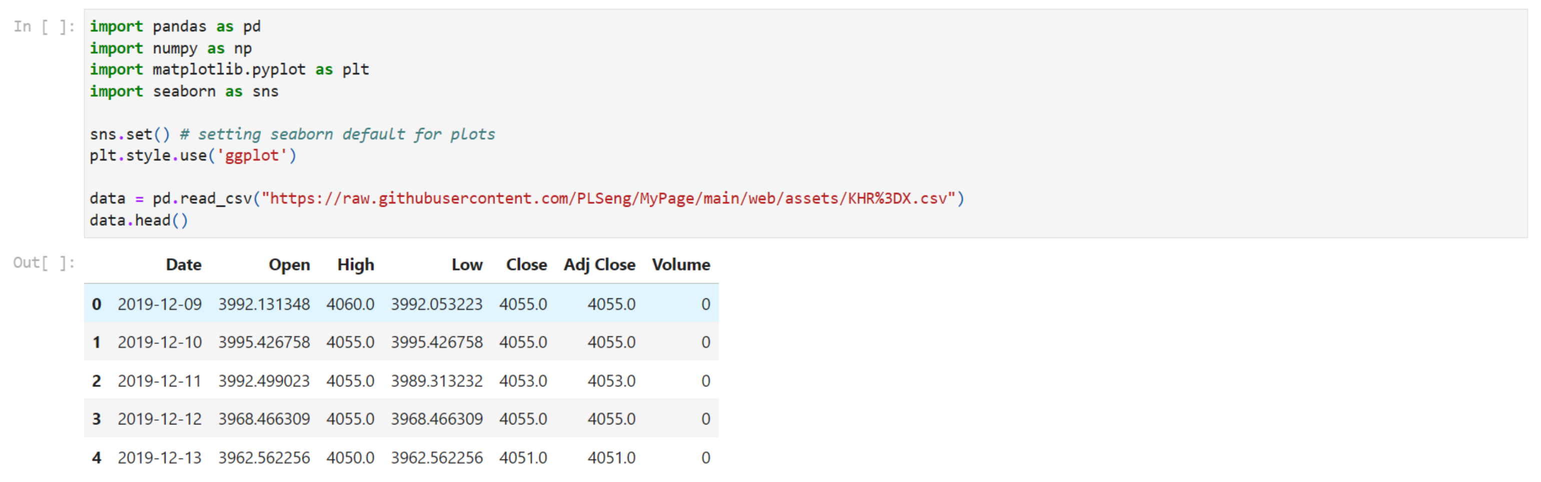
Select the criteria you want then click on the download button.



2

Data Preprocessing

Import Data and Necessary Library



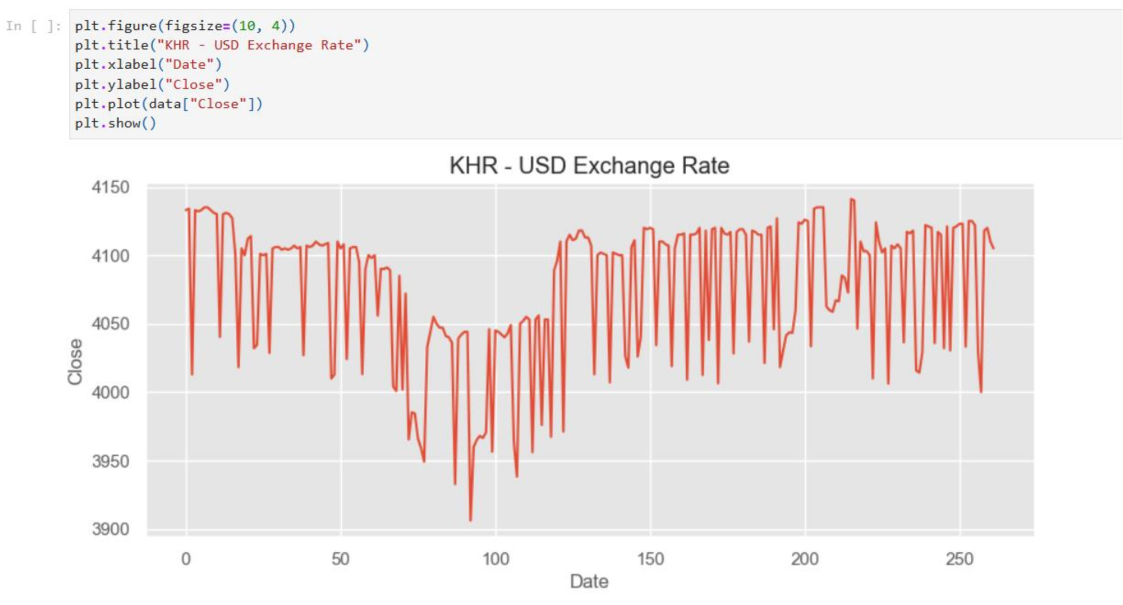
In [ ]:

```python
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[ ]:

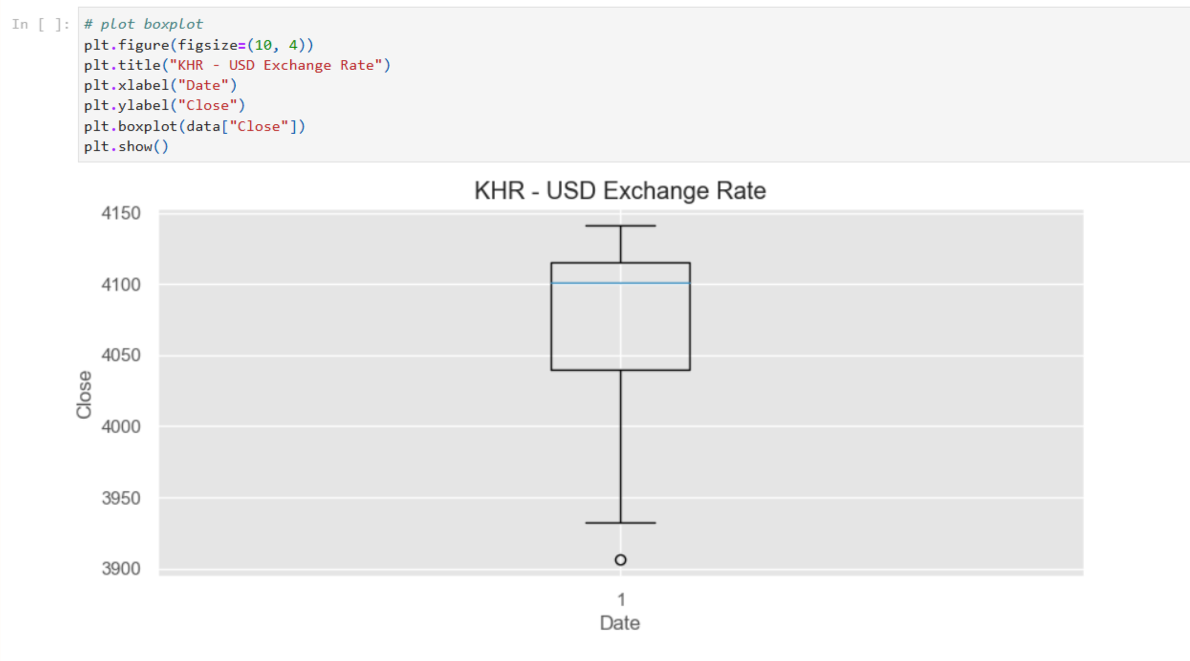
	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-12-09	3992.131348	4060.0	3992.053223	4055.0	4055.0	0
1	2019-12-10	3995.426758	4055.0	3995.426758	4055.0	4055.0	0
2	2019-12-11	3992.499023	4055.0	3989.313232	4053.0	4053.0	0
3	2019-12-12	3968.466309	4055.0	3968.466309	4055.0	4055.0	0
4	2019-12-13	3962.562256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

Import Data and Necessary Library



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

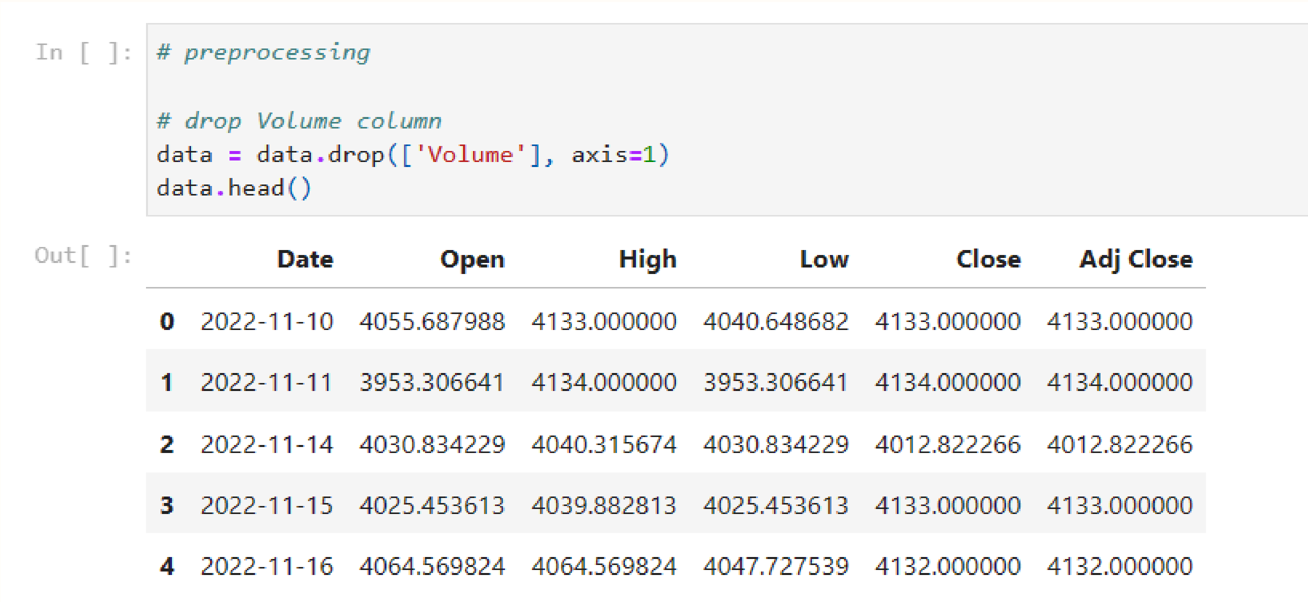
Plot Boxplot to see outliers



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



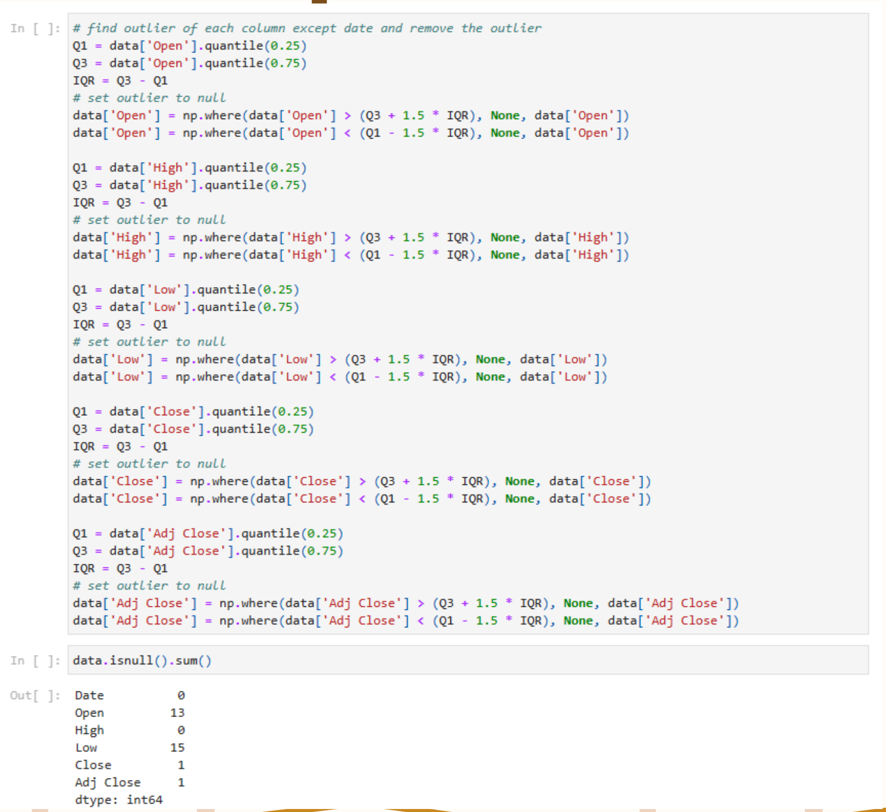
In [ ]:

```
# preprocessing

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

Out[ ]:

	Date	Open	High	Low	Close	Adj Close
0	2022-11-10	4055.687988	4133.000000	4040.648682	4133.000000	4133.000000
1	2022-11-11	3953.306641	4134.000000	3953.306641	4134.000000	4134.000000
2	2022-11-14	4030.834229	4040.315674	4030.834229	4012.822266	4012.822266
3	2022-11-15	4025.453613	4039.882813	4025.453613	4133.000000	4133.000000
4	2022-11-16	4064.569824	4064.569824	4047.727539	4132.000000	4132.000000

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'])
```

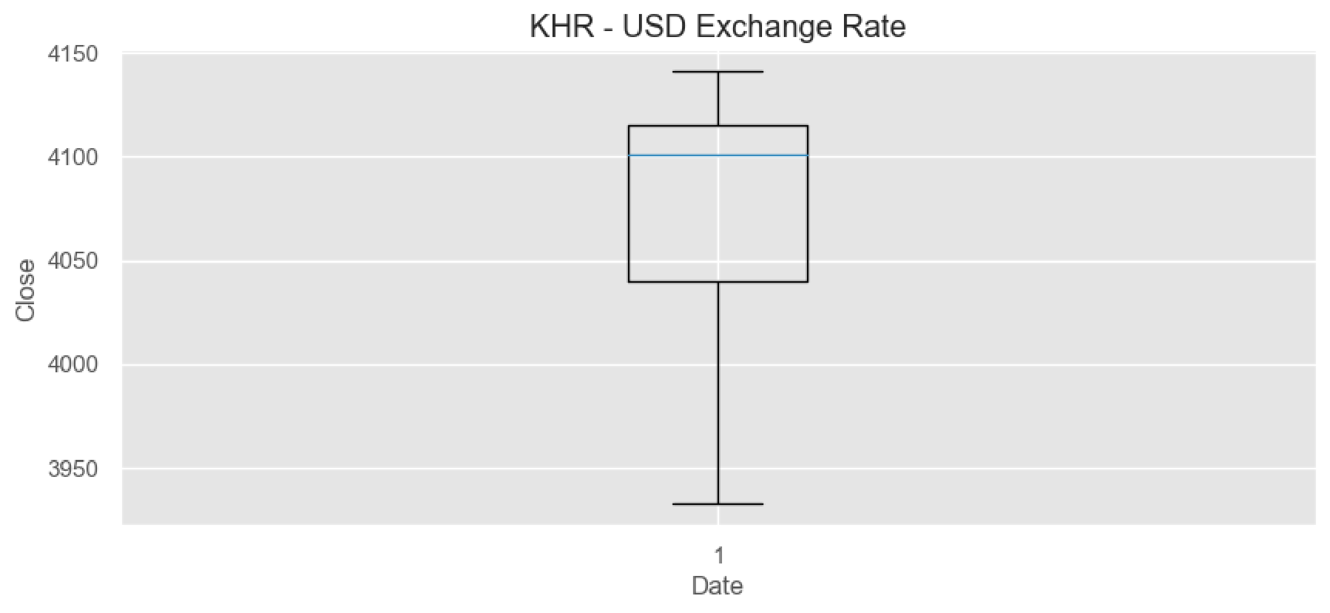
In [ ]:

```
data.isnull().sum()
```

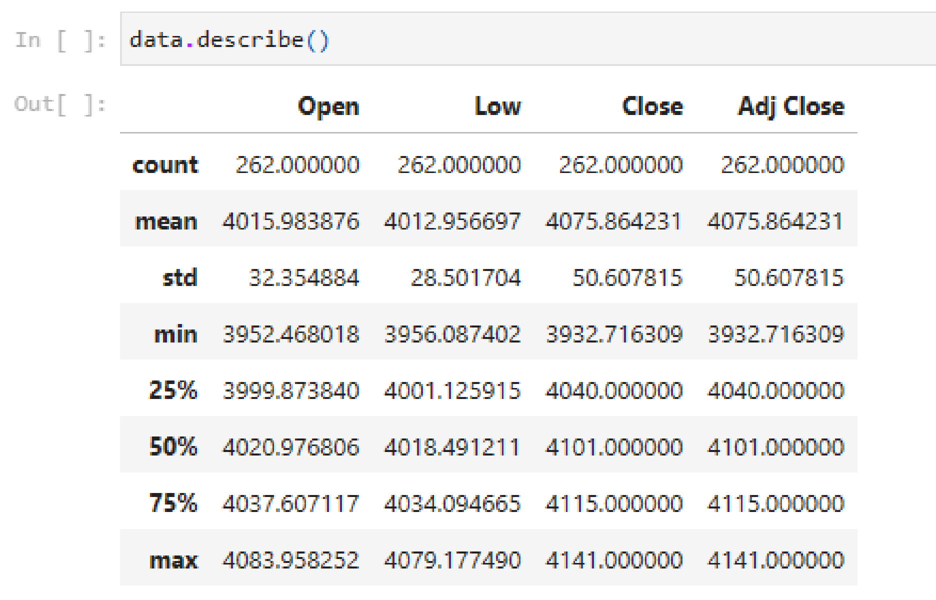
Out[ ]:

```
Date          0
Open         13
High          0
Low          15
Close         1
Adj Close     1
dtype: int64
```

Fill Empty cell with min value



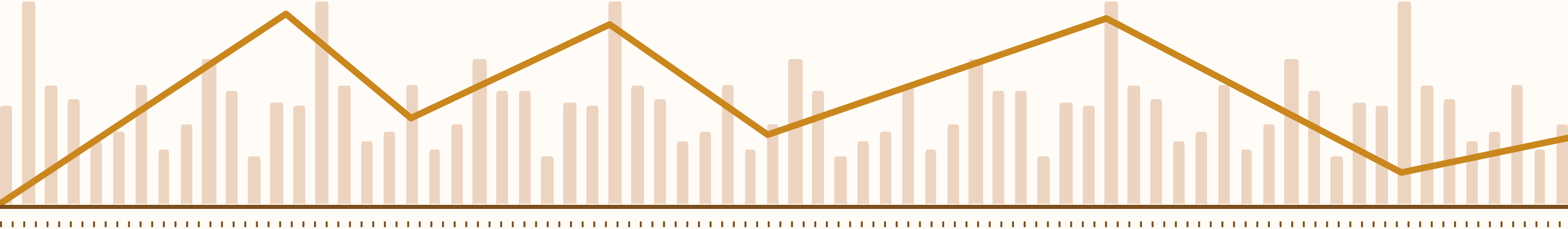
Descriptive Statistics



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

Out[ ]:

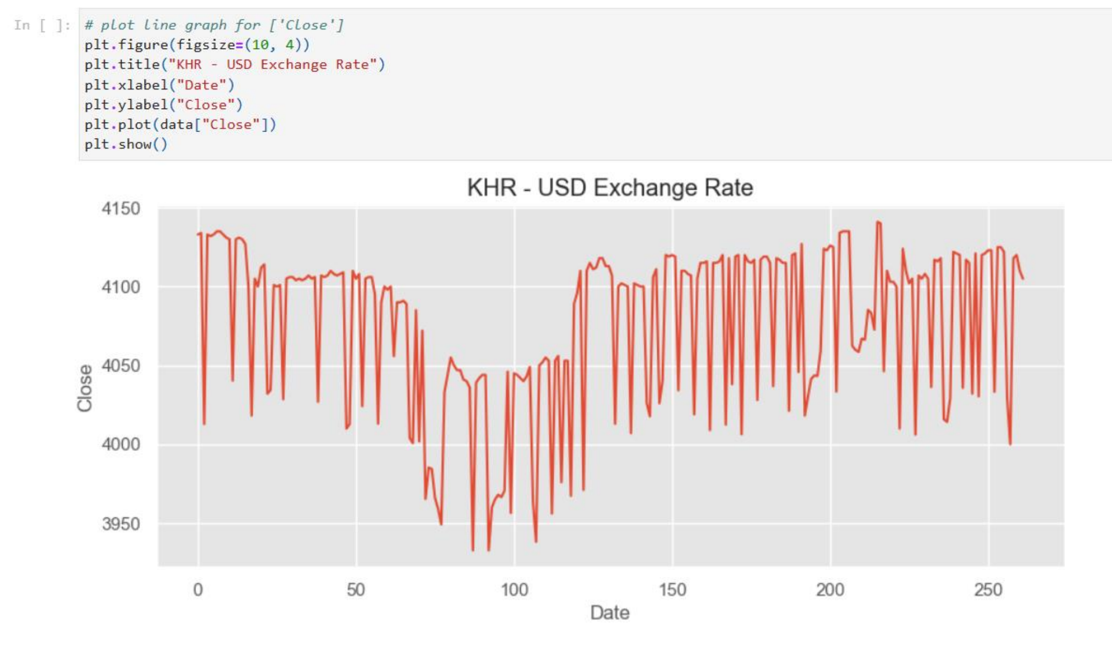
	Open	Low	Close	Adj Close
count	262.000000	262.000000	262.000000	262.000000
mean	4015.983876	4012.956697	4075.864231	4075.864231
std	32.354884	28.501704	50.607815	50.607815
min	3952.468018	3956.087402	3932.716309	3932.716309
25%	3999.873840	4001.125915	4040.000000	4040.000000
50%	4020.976806	4018.491211	4101.000000	4101.000000
75%	4037.607117	4034.094665	4115.000000	4115.000000
max	4083.958252	4079.177490	4141.000000	4141.000000



3

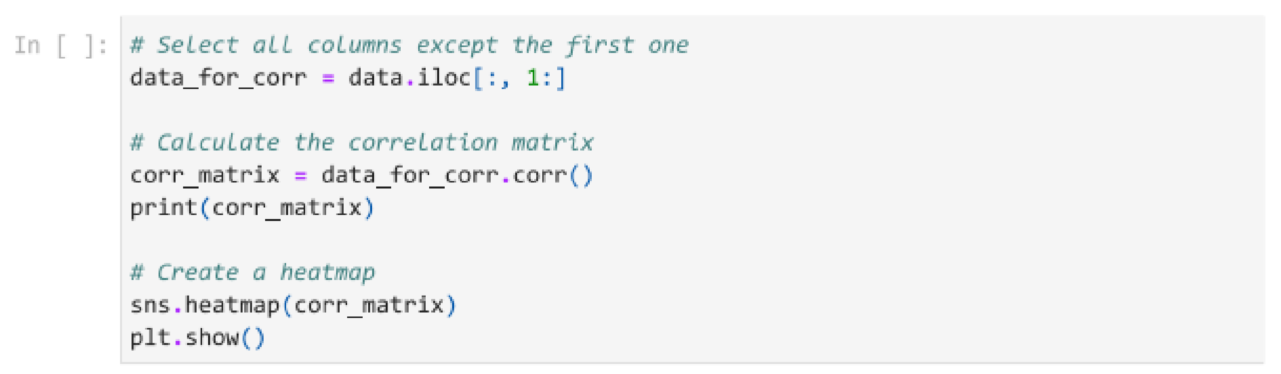
Data Visualisation

Plot the time series graph



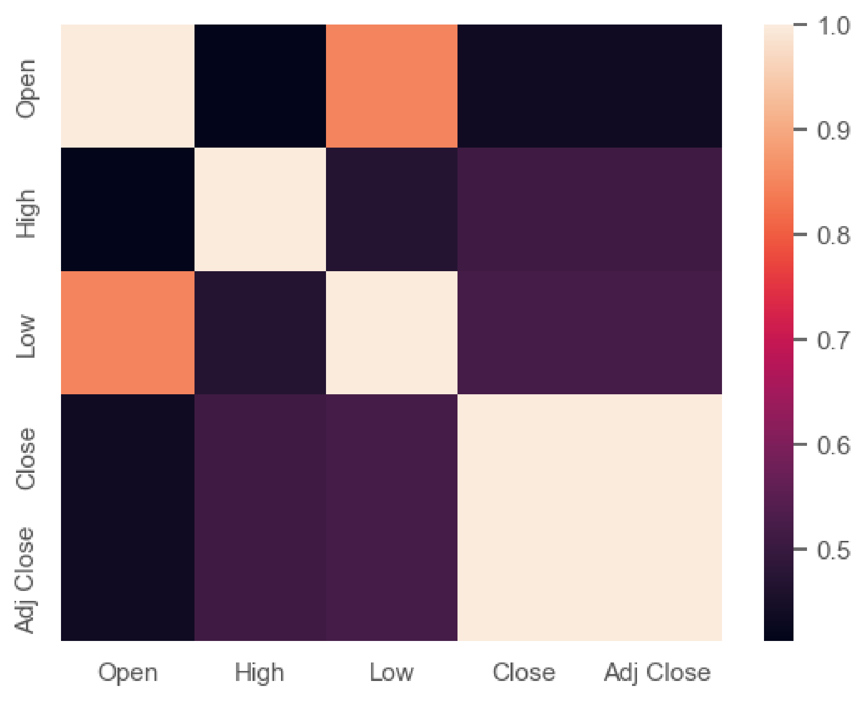
```
In [ ]: # plot line graph for ['Close']
        plt.figure(figsize=(10, 4))
        plt.title("KHR - USD Exchange Rate")
        plt.xlabel("Date")
        plt.ylabel("Close")
        plt.plot(data["Close"])
        plt.show()
```

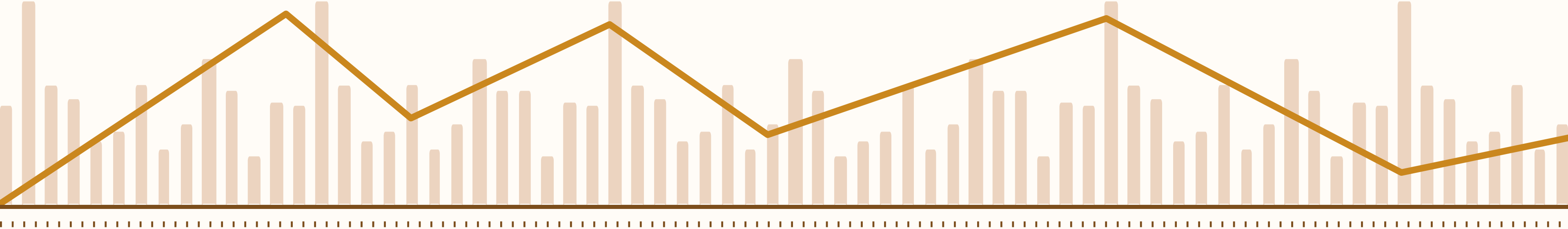
Plot a heatmap to see the correlation



```
# 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

Scikit-Learn

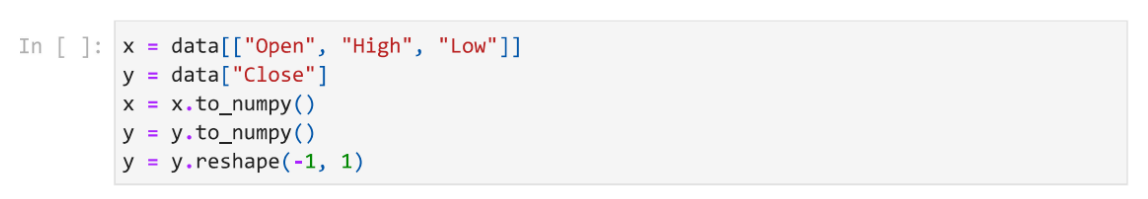
train_test_split

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.

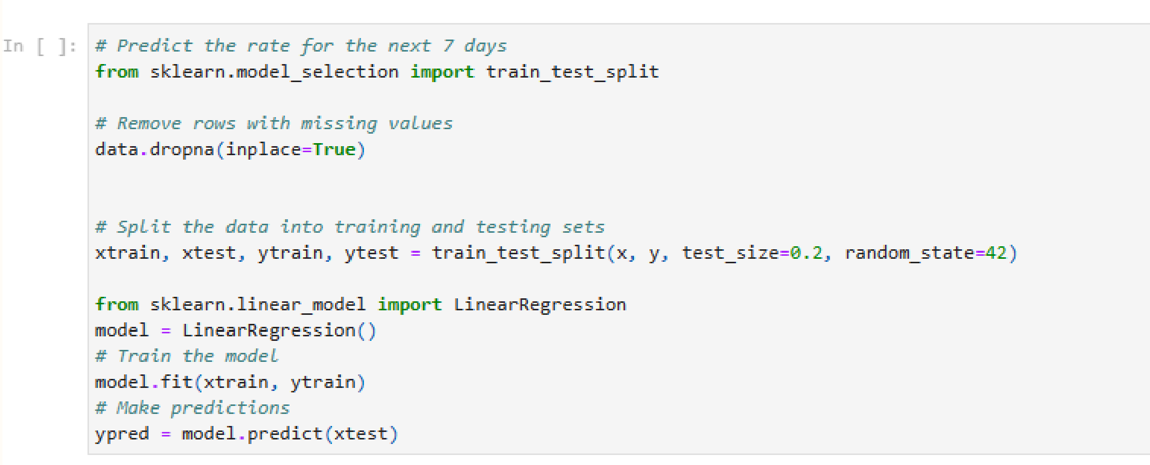
LinearRegression

Linear regression in scikit-learn is a method used for predictive modeling. Similar to how the **`DecisionTreeRegressor`** works by **splitting data** and using parameters like criterion and **max_depth,** linear regression follows its own process. Linear regression is implemented through the **`LinearRegression`** class.

Regression

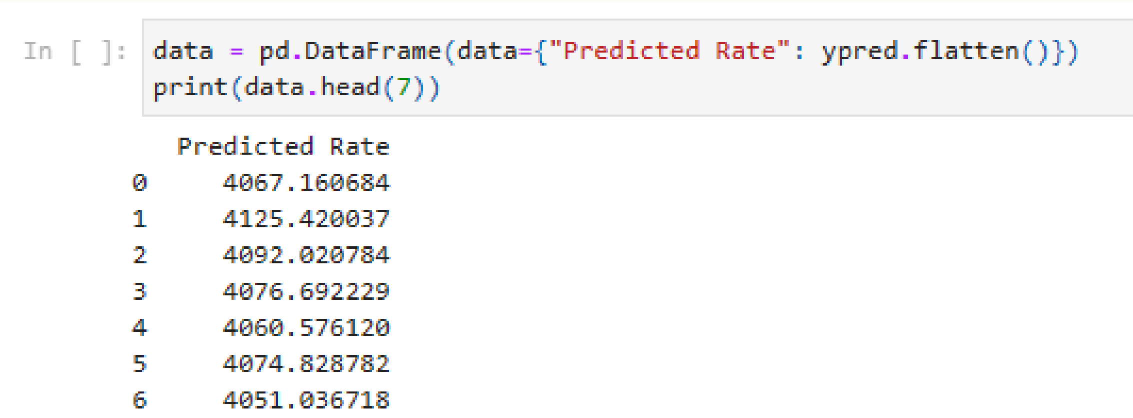


```
In [ ]: x = data[["Open", "High", "Low"]]
        y = data["Close"]
        x = x.to_numpy()
        y = y.to_numpy()
        y = y.reshape(-1, 1)
```



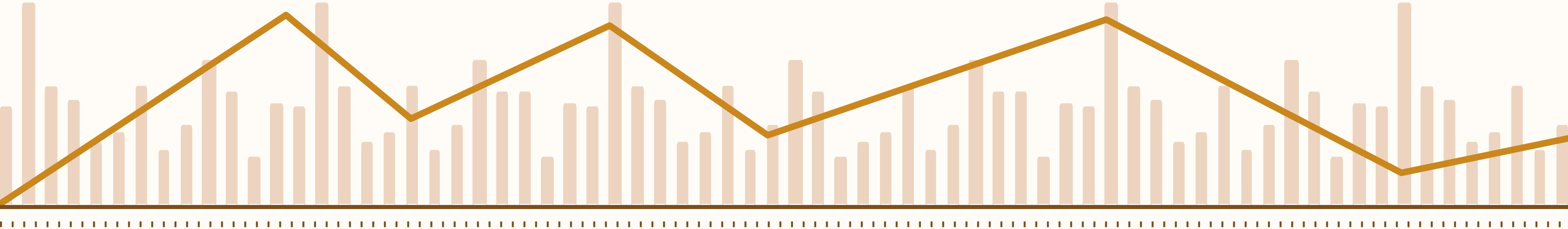
```
In [ ]: # Predict the rate for the next 7 days
        from sklearn.model_selection import train_test_split

        # 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=42)

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

Prediction



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

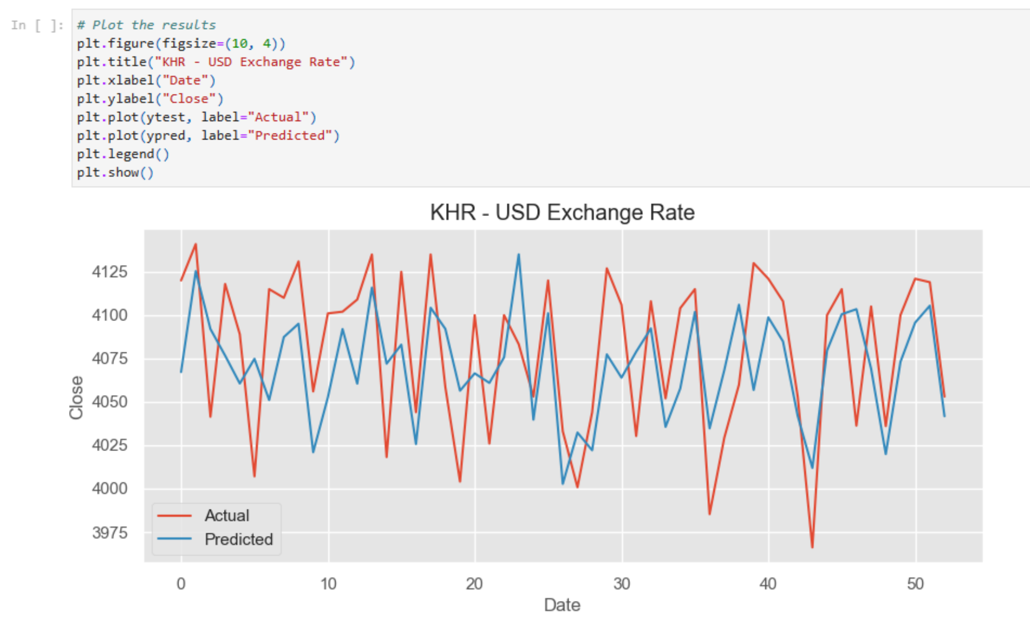
```
   Predicted Rate
0     4067.160684
1     4125.420037
2     4092.020784
3     4076.692229
4     4060.576120
5     4074.828782
6     4051.036718
```



5

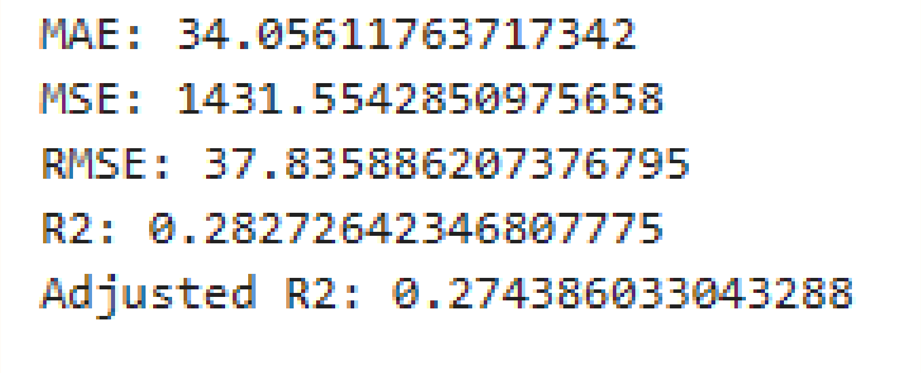
Accuracy Rate

Trendline between Actual and Predicted Value

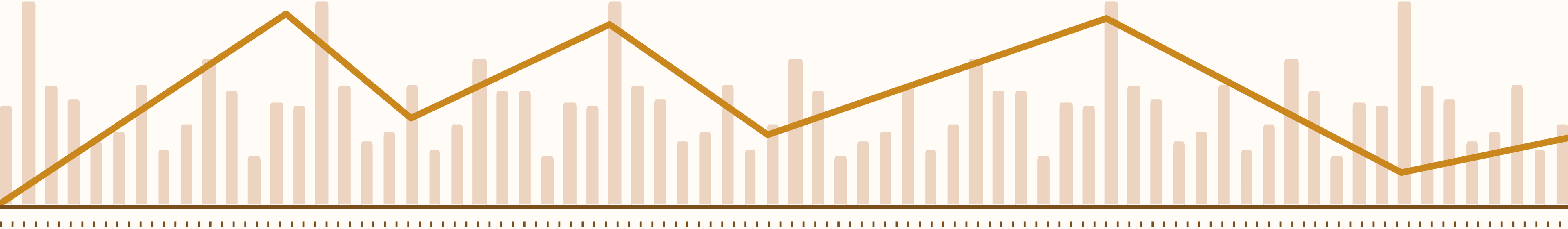


```
In [ ]: # Plot the results
        plt.figure(figsize=(10, 4))
        plt.title("KHR - USD Exchange Rate")
        plt.xlabel("Date")
        plt.ylabel("Close")
        plt.plot(ytest, label="Actual")
        plt.plot(ypred, label="Predicted")
        plt.legend()
        plt.show()
```

Get the accuracy rate

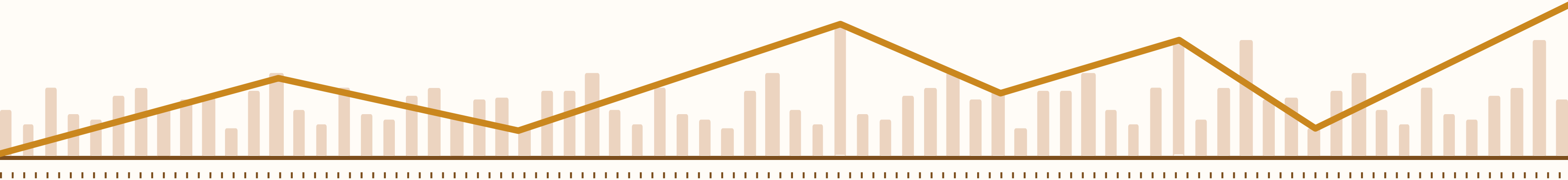


```
MAE: 34.05611763717342
MSE: 1431.5542850975658
RMSE: 37.835886207376795
R2: 0.28272642346807775
Adjusted R2: 0.274386033043288
```



6

Conclusion



Thanks!

Do you have any questions?

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution