تابع get_data_from_mt5:

تابع `get_data_from_mt5` یک تابع است که برای دریافت داده‌های تاریخی بازار از پلتفرم MetaTrader5 استفاده میکند. با استفاده از این تابع، می‌توانید داده‌های تاریخی بازار را دریافت کرده و در قالب یک DataFrame از پکیج pandas ذخیره کنید.

در ابتدا، تابع با استفاده از اطلاعات ورودی، به پلتفرم MetaTrader5 متصل می‌شود و وارد حساب کاربری می‌شود. سپس، فریم‌های زمانی معتبر را تعریف می‌کند.

با استفاده از تابع `copy_rates_from_pos` و با استفاده از پارامترهای نماد ارزی، فریم زمانی و محدوده داده‌ها، داده‌های تاریخی بازار را از پلتفرم دریافت می‌کند. سپس، این داده‌ها را در یک DataFrame ذخیره می‌کند.

مراحلی از پردازش و تمیز کردن داده‌ها نیز انجام می‌شود. به عنوان مثال، ستون زمان به فرمت مناسب تبدیل می‌شود، ستون زمان به فرمت زمانی UTC تنظیم می‌شود و ستون‌های غیرضروری مثل 'spread'، 'time' و 'real_volume' حذف می‌شوند. همچنین، نام ستون‌های موجود در DataFrame نیز تغییر می‌کند.

در نهایت، دو ستون جدید به نام‌های 'Mean' و 'diff' به DataFrame اضافه می‌شوند. ستون 'Mean' میانگینی از ستون‌های 'High'، 'Low' و 'Close' است و ستون 'diff' تفاوت ستون 'Mean' با مقدار قبلی آن است. همچنین، ستونی به نام 'info' با مقدار 'نماد ارزی_فریم زمانی' به DataFrame اضافه می‌شود.

در نهایت، DataFrame حاصل به عنوان خروجی تابع برگردانده می‌شود.

مثال استفاده:

```
initialize = [123456, 'password', 'MetaTraderServer']

Ticker = 'EURUSD'

TimeFrame = '1h'

data = get_data_from_mt5(initialize, Ticker, TimeFrame)
```

تابع `get_data_from_mt5` به شما امکان می‌دهد داده‌های تاریخی بازار از پلتفرم MetaTrader5 را دریافت کرده و برای تحلیل و پردازش به صورت ساده‌تر و کارآمدتری استفاده کنید.

تابع clean_investing_data:

تابع `clean_investing_data` یک تابع است که برای پاک‌سازی و پردازش داده‌های مربوط به سایت investing استفاده می‌شود.

این تابع یک DataFrame را به عنوان ورودی دریافت می‌کند و دستوراتی را برروی آن اعمال می‌کند تا داده‌ها تمیز شده و آماده استفاده تحلیلی شوند.

در ابتدا، ستون تاریخ را به عنوان نمایه (index) بر روی DataFrame تنظیم می‌کند. برای این کار از تابع `pd.to_datetime` استفاده می‌کند.

سپس، DataFrame را بر اساس نمایه مرتب می‌کند. در اینجا، بر اساس تاریخ به صورت صعودی مرتب می‌شود.

ستون 'Price' را به نام 'Close' تغییر نام می‌دهد تا سازگاری در نام‌گذاری ستون‌ها ایجاد شود.

ستون‌های غیرضروری مانند 'Change %'، 'Date' و 'Vol.' را از DataFrame حذف می‌کند. در صورت وجود ستون 'Unnamed: 0' نیز آن را حذف می‌کند.

داده‌ها را پاکسازی می‌کند. ابتدا، کاماها را از اعداد حذف می‌کند و سپس اعداد را به فرمت اعشاری تبدیل می‌کند.

ستون 'Mean' را با میانگین ستون‌های 'High'، 'Low' و 'Close' محاسبه می‌کند و ستون 'diff' را به عنوان تفاوت میانگین فعلی و مقدار قبلی آن محاسبه می‌کند.

در نهایت، سطرهای تکراری را حذف کرده و سطرهایی که حاوی مقادیر NaN هستند را نیز حذف می‌کند.

DataFrame نهایی را به عنوان خروجی تابع برمی‌گرداند.

مثال استفاده:

```
cleaned_data = clean_investing_data(df)
print(cleaned_data.head())
```

تابع `clean_investing_data` به شما امکان می‌دهد داده‌های مربوط به سایت investing را پاک‌سازی و آماده استفاده تحلیلی کنید.

تابع get_country_index_from_investing:

تابع `get_country_index_from_investing` یک تابع است که برای دریافت شاخص کشور از داده‌های بازار سرمایه استفاده می‌شود. این تابع نام کشور را به عنوان ورودی دریافت می‌کند و بر اساس آن، داده‌های مربوط به شاخص کشور مورد نظر را از داده‌های بازار سرمایه دریافت می‌کند و پاکسازی می‌کند.

ابتدا، داده‌های مربوط به شاخص دلار آمریکا (US Dollar Index) را از فایل CSV می‌خواند و با استفاده از تابع `clean_investing_data` آن را پاکسازی می‌کند. سپس، اگر نام کشور ورودی برابر با 'USD' باشد، شاخص کشور را برابر با داده‌های شاخص دلار آمریکا قرار می‌دهد.

در غیر اینصورت، اگر نام کشور ورودی در لیست ['CAD', 'JPY', 'SEK', 'CHF'] وجود داشت، نماد ارزی مربوط به کشور مورد نظر را با 'USD' ترکیب می‌کند و داده‌های مربوط به آن را از فایل CSV می‌خواند و پاکسازی می‌کند. ستون‌های 'Open'، 'High'، 'Low" و 'Close' را از داده‌ها استخراج کرده و در متغیر `df` قرار می‌دهد. سپس، شاخص کشور را با تقسیم شاخص دلار آمریکا بر داده‌های شاخص مربوط به کشور محاسبه می‌کند.

در غیر اینصورت، نماد ارزی مربوط به کشور را با 'USD' ترکیب می‌کند و داده‌های مربوط به آن را از فایل CSV می‌خواند و پاکسازی می‌کند. ستون‌های 'Low'، 'High'، 'Open' و 'Close' را از داده‌ها استخراج کرده و در متغیر `df` قرار می‌دهد. سپس، شاخص کشور را با ضرب داده‌های شاخص مربوط به کشور در شاخص دلار آمریکا محاسبه می‌کند.

ستون 'Mean' را با میانگین ستون‌های 'High'، 'Low' و 'Close' محاسبه می‌کند و ستون 'diff' را به عنوان تفاوت میانگین فعلی و مقدار قبلی آن محاسبه می‌کند.

در نهایت، سطرهایی که حاوی مقادیر NaN هستند را حذف می‌کند و DataFrame حاصل را به عنوان خروجی تابع برمی‌گرداند.

مثال استفاده:

```
country_index = get_country_index_from_investing('CAD')
print(country_index.head())
```

تابع `get_country_index_from_investing` به شما امکان می‌دهد شاخص کشور مورد نظر را از داده‌های بازار سرمایه دریافت و پاکسازی کنید.

تابع get_csv_files:

تابع `get_csv_files` یک تابع است که لیستی از فایل‌های CSV در دایرکتوری مربوط به کشور مورد نظر را بازمی‌گرداند.

این تابع یک رشته به نام `country` را به عنوان ورودی دریافت می‌کند که نام کشور مورد نظر را مشخص می‌کند.

ابتدا، مسیر کامل دایرکتوری مربوط به کشور را تشکیل می‌دهد. سپس، مسیر کنونی را با استفاده از تابع `os.getcwd()` دریافت می‌کند و مسیر دایرکتوری مربوط به کشور را به آن اضافه می‌کند. همچنین، پسوند فایل را برابر با 'csv' تعیین می‌کند.

سپس، با استفاده از تابع `os.chdir()`، مسیر کنونی را به مسیر دایرکتوری مربوط به کشور تغییر می‌دهد.

با استفاده از تابع `glob.glob()`، فایل‌های با پسوند 'csv' در دایرکتوری را خوانده و نام فایل‌ها را بدون پسوند '.csv' در لیست `csv_files` قرار می‌دهد.

در نهایت، با استفاده از تابع `os.chdir()`، مسیر کنونی را به مسیر قبلی تغییر می‌دهد و لیست `csv_files` را به عنوان خروجی تابع برمی‌گرداند.

مثال استفاده:

```
files = get_csv_files('USD')
print(files)
```

تابع `get_csv_files` به شما امکان می‌دهد لیست فایل‌های CSV در دایرکتوری مربوط به کشور مورد نظر را دریافت کنید.

تابع clean_news:

تابع `clean_news` یک تابع است که داده‌های مربوط به ستون 'News' در DataFrame را پاکسازی می‌کند و نتایج را در لیست `news` قرار می‌دهد.

ابتدا، یک لیست خالی به نام `news` تعریف می‌شود.

سپس، برای هر ردیف در ستون 'News'، مراحل پاکسازی را انجام می‌دهد. ابتدا، رشته را براساس فاصله‌ها جدا می‌کند و در لیست `news_parts` قرار می‌دهد. سپس، از تابع `list.filter()` با استفاده از عملگر نهادی "__ne__" برای حذف فضاهای خالی در لیست `news_parts` استفاده می‌کند.

سپس، برای هر عنصر در `news_parts`، مراحل پاکسازی دیگر را انجام می‌دهد. اگر عنصر حاوی '(' باشد و برابر با "(YoY)" یا "(MoM)" باشد، عنصر را به لیست `parts` اضافه می‌کند. در غیر اینصورت، اگر عنصر برابر با "\xa0" باشد، عنصر "Flash" را در ابتدای لیست `parts` قرار می‌دهد. در غیر اینصورت، عنصر را به لیست `parts` اضافه می‌کند.

در نهایت، لیست `parts` را با استفاده از فاصله‌ها به یک رشته تبدیل کرده و در لیست `news` قرار می‌دهد.

در نهایت، لیست `news` را به عنوان خروجی تابع برمی‌گرداند.

تابع `fix_dataframe` یک تابع است که DataFrame مربوطه را پاکسازی می‌کند و بازمی‌گرداند.

در ابتدا، DataFrame ورودی را در متغیر `df` قرار می‌دهد.

ابتدا، سطرهایی را که در ستون 'currency' خالی هستند را حذف می‌کند.

سپس، سطرهایی که در ستون 'time' برابر با 'Tentative' هستند را حذف می‌کند.

همچنین، سطرهایی که در ستون 'time' برابر با 'All Day' هستند را حذف می‌کند.

ستون 'Date_Time' را با ترکیب ستون‌های 'date' و 'time' و با فرمت 'dd/mm/yyyy HH:MM' محاسبه می‌کند و ستون 'Forecast'، 'Actual'، 'Country'، 'News'، 'Impact' و 'Previous' را با ترتیب ستون‌های 'event'، 'importance'، 'forecast'، 'actual'، 'zone" و 'previous' تنظیم می‌کند.

ستون‌هایی که مورد نیاز نیستند را حذف می‌کند.

سپس، ستون 'Forecast'، 'Actual' و 'Previous' را به عنوان ورودی به تابع `clean_news` می‌دهد و ستون 'Country' را به حروف بزرگ تغییر می‌دهد.

در ادامه، ستون‌های مورد نیاز را انتخاب کرده و به ترتیب 'Forecast'، 'Actual'، 'Country'، 'News'، 'Date_Time' و 'Previous' قرار می‌دهد.

در نهایت، DataFrame نهایی را به عنوان خروجی تابع برمی‌گرداند.

مثال استفاده:

```
import pandas as pd

data = {
    'currency': ['USD', 'EUR', 'GBP'],
    'time': ['10:00', 'Tentative', 'All Day'],
```

```
    'date': ['01/01/2023', '02/01/2023', '03/01/2023'],
    'importance': ['High', 'Medium', 'Low'],
    'event': ['News 1', 'News 2', 'News 3'],
    'zone': ['Zone 1', 'Zone 2', 'Zone 3'],
    'actual': ['1.23', '2.34', '3.45'],
    'forecast': ['1.25', '2.36', '3.47'],
    'previous': ['1.21', '2.32', '3.43']
}
df = pd.DataFrame(data)
cleaned_df = fix_dataframe(df)
print(cleaned_df)
```

خروجی:

```
            Date_Time    News Country  Actual  Forecast  Previous
0 2023-01-01 10:00:00  News 1  Zone 1    1.23      1.25      1.21
2 2023-01-03 00:00:00  News 3  Zone 3    3.45      3.47      3.43
```

تابع convert_to_gmt:

تابع `convert_to_gmt` یک تابع است که زمان دریافتی به شکل GMT را تبدیل می‌کند و نتیجه را به عنوان رشته برمی‌گرداند.

ابتدا، آفست زمان دریافتی را با استفاده از متد `strftime` و الگوی `%z` استخراج می‌کند. آفست زمان به صورت یک رشته شامل 4 رقم عددی است که نمایانگر اختلاف زمانی به دقیقه در مقایسه با GMT است.

سپس، با استفاده از رشته حاصل، رشته نهایی را تشکیل می‌دهد که شامل عبارت "GMT" و زمان رشته آفست است. برای تشکیل زمان رشته نهایی، از ارقام رشته آفست استفاده می‌کند. اگر رقم اول از صفر متفاوت باشد، آن رقم را در رشته نهایی قرار می‌دهد. سپس، رقم دوم را در رشته نهایی قرار می‌دهد. سپس، دو نقطه اضافه می‌کند و رقم سوم و چهارم را در رشته نهایی قرار می‌دهد.

در نهایت، رشته نهایی را به عنوان خروجی تابع برمی‌گرداند.

مثال استفاده:

```
from datetime import datetime

requested_time = datetime(2023, 1, 1, 10, 30)

converted_time = convert_to_gmt(requested_time)

print(converted_time)
```

خروجی:

```
GMT +05:30
```

تابع get_today_calendar:

تابع `get_today_calendar` یک تابع است که تقویم اقتصادی برای روز جاری بر اساس منطقه زمانی و کشورهای مشخص شده دریافت می‌کند و نتیجه را به عنوان یک DataFrame برمی‌گرداند.

ابتدا، زمان فعلی را با استفاده از `datetime.now()` و منطقه زمانی مشخص شده دریافتی محاسبه می‌کند و در متغیر `now` قرار می‌دهد.

سپس، زمان GMT فعلی را با استفاده از تابع `convert_to_gmt` برای زمان `now` محاسبه می‌کند و در متغیر `gmt_format` قرار می‌دهد.

سپس، با استفاده از متد `investpy.news.economic_calendar()`، تقویم اقتصادی برای منطقه زمانی GMT و کشورهای مشخص شده دریافتی را در بازه زمانی از زمان فعلی تا 24 ساعت بعد از آن دریافت می‌کند و در متغیر `df` قرار می‌دهد.

سپس، DataFrame حاصل را با استفاده از تابع `fix_dataframe` پاکسازی می‌کند و در همان متغیر `df` قرار می‌دهد.

در نهایت، DataFrame نهایی را به عنوان خروجی تابع برمی‌گرداند.

توجه: برای استفاده از این تابع، باید ماژول `investpy` را نصب کنید.

تابع make_folder:

تابع `make_folder` یک تابع است که یک پوشه با مسیر مشخص شده را ایجاد می‌کند، در صورتی که پوشه در مسیر وجود نداشته باشد.

ابتدا، تابع با استفاده از `os.path.exists` بررسی می‌کند که آیا پوشه با مسیر مشخص شده در مسیر فعلی وجود دارد یا خیر.

در صورتی که پوشه وجود نداشته باشد، با استفاده از `os.mkdir` پوشه را ایجاد می‌کند.

توجه: برای استفاده از این تابع، باید ماژول `os` را وارد کنید.

تابع merge_dataframes:

تابع `merge_dataframes` یک تابع است که فایل‌های CSV موجود در یک مسیر را با هم ترکیب کرده و یک DataFrame حاوی اطلاعات تمام فایل‌ها را برمی‌گرداند.

در ابتدا، تمام فایل‌های CSV موجود در مسیر مشخص شده را با استفاده از `glob` و تکه‌های کد `pd.read_csv(file,]` `[(index_col=0) for file in glob(path` می‌خواند و در لیست `li` قرار می‌دهد. هر فایل CSV به صورت یک DataFrame خوانده می‌شود و ستون اول آن به عنوان ستون شاخص (index) استفاده می‌شود.

سپس، از تابع `pd.concat` برای ترکیب تمام DataFrameها در لیست `li` با هم استفاده می‌شود. ترکیب این DataFrameها باعث ایجاد یک DataFrame جدید با تمام ردیف‌ها می‌شود. پارامتر `axis=0` برای ترکیب بر اساس ستون‌ها (ردیف به ردیف) استفاده می‌شود. پارامتر `ignore_index=True` نیز برای بازنشانی شماره ردیف‌ها در DataFrame جدید استفاده می‌شود.

سپس، با استفاده از `sort_values`، ردیف‌ها بر اساس ستون "Date_Time" به صورت صعودی مرتب می‌شوند. سپس، با استفاده از `reset_index`، شماره ردیف‌ها به صورت مجدد تنظیم می‌شوند و ستون اضافی "index" حذف می‌شود.

در نهایت، DataFrame نهایی را به عنوان خروجی تابع برمی‌گرداند.

توجه: برای استفاده از این تابع، باید ماژول `pandas` را وارد کنید و نیز `glob` را از ماژول `glob` استوریج کنید.

تابع get_calendar_historical_data:

تابع `get_calendar_historical_data` یک تابع است که اطلاعات تقویم اقتصادی تاریخچه را برای یک یا چند کشور در بازه زمانی مشخص دریافت کرده و آن‌ها را در فایل‌های CSV ذخیره می‌کند.

توضیحات تابع به شرح زیر است:

1. تابع `get_calendar_historical_data` پارامترهای ورودی زیر را می‌پذیرد:

- `from_year`: سال شروع بازه زمانی مورد نظر برای دریافت اطلاعات تقویم اقتصادی. پیش‌فرض آن 2008 است.

- `to_year`: سال پایان بازه زمانی مورد نظر برای دریافت اطلاعات تقویم اقتصادی. پیش‌فرض آن 2022 است.

- `to_date`: تاریخ روز پایان بازه زمانی مورد نظر برای دریافت اطلاعات تقویم اقتصادی. پیش‌فرض آن "05/08" است.

- `save_path`: مسیر ذخیره‌سازی فایل‌های CSV. پیش‌فرض آن "static" است.

- `countries`: لیست کشورهای مورد نظر برای دریافت اطلاعات تقویم اقتصادی. پیش‌فرض آن شامل ["","United States" ""Euro Zone", "Canada", "united kingdom"] است.

2. ابتدا، تابع یک مسیر موقت به نام `temp_path` در مسیر ذخیره‌سازی (پوشه "static" یا مسیر دلخواه) ایجاد می‌کند.

3. سپس، برای هر کشور در لیست `countries`، یک مسیر مربوط به آن کشور در پوشه موقت ایجاد می‌کند.

4. سپس، با استفاده از یک حلقه، اطلاعات تقویم اقتصادی برای هر سال در بازه `from_year` تا `to_year-1` برای کشور مورد نظر دریافت می‌شود. این اطلاعات در یک دیتافریم قرار گرفته و در یک فایل CSV با نام `"{country}_{i}_{i+1}_all_news.csv"` ذخیره می‌شود، که `country` نام کشور و `i` و `i+1` سال‌های بازه زمانی هستند.

5. در صورت بروز خطا در دریافت اطلاعات تقویم اقتصادی برای یک سال، عملیات برای آن سال انجام نمی‌شود و ادامه حلقه صورت می‌گیرد.

6. سپس، اطلاعات تقویم اقتصادی برای سال `to_year` تا تاریخ `to_date` برای کشور مورد نظر دریافت می‌شود و در یک فایل CSV با نام `"{country}_{to_year}_all_news.csv"` ذخیره می‌شود.

7. سپس، تمام فایل‌های CSV مربوط به کشور در پوشه مربوطه در مسیر ذخیره‌سیزی که در مرحله قبل ایجاد شده‌اند، با استفاده از تابع `merge_dataframes` ترکیب شده و در یک فایل CSV با نام `"{country}_all_news.csv"` در مسیر ذخیره‌سازی اصلی ذخیره می‌شوند.

8. تابع به پایان می‌رسد و پیام "Done!" را نمایش می‌دهد.

مقدار پیش‌فرض برای پارامتر `save_path` در تابع این است که مسیر ذخیره‌سازی فایل‌ها "static" باشد. اگر مقدار پارامتر `save_path` خالی (`""`) یا نقطه (`"."`) باشد، مقدار `os.path.abspath('.')` برای `save_path` استفاده می‌شود.

در نهایت، تابع `get_calendar_historical_data` امکان می‌دهد اطلاعات تقویم اقتصادی تاریخچه را برای یک یا چند کشور در بازه زمانی مشخص دریافت کرده و آن‌ها را در فایل‌های CSV ذخیره کنید.

مثال استفاده:

```
get_calendar_historical_data(from_year=2019, to_year=2021, to_date="12/31", save_path="data", countries=["United States", "Canada"])
```

در این مثال، تابع get_calendar_historical_data برای کشورهای "United States" و "Canada"، اطلاعات تقویم اقتصادی بین سال‌های 2019 تا 2020 (شامل سال 2019 و سال 2020) را دریافت کرده و در فایل‌های CSV در پوشه "data" ذخیره می‌کند. تاریخ پایان بازه زمانی نیز به عنوان "12/31" تعیین شده است.