به نام خدا

امیرمحمد کمیجانی 99522032

گزارش تمرین سری اول

1)

در این سوال برای مقایسه عنصر به عنصر دو آرایه با اندازه مساوی از توابع آماده موجود در کتابخانه **numpy** استفاده میکنیم.برای تمامی حالات خواسته شده تابع آماده تعبیه شده است.
توابع آماده موجود در نامپای:
greater,greater_equal,less,less_equal

در خروجی هم آرایه ای هم اندازه با آرایه های ورودی داریم که حاصل مقایسه عناصر را به صورت **True/False** نمایش داده ایم.

2)

در رابطه با ضرب دو ماتریس با استفاده از کتابخانه نامپای سه روش وجود دارد:
np.dot (a np.matmul (b np.multiply (c

تابع multiply ضرب ماتریس ها را به صورت عنصر به عنصر انجام میدهد که یکی از خواسته های سوال است.

برای ضرب ماتریسی از تابع **matmul** استفاده میکنیم.در رابطه با این تابع باید ذکر کنیم که این تابع خروجی نهایی را به صورت آرایه **دوبعدی** محاسبه و خروجی میدهد اگر یک بعدی یا سه بعدی و بیشتر باشد هم به دوبعدی تبدیل میکنید و بعد محاسبه و خروجی انجام میگیرد.

در رابطه با ضرب ماتریسی تابع **dot** هم خروجی مدنظر را در این سوال به ما میدهد و مکانیزم آن به صورت ضرب **اسکالر** میباشد.

3)

برای اضافه کردن دو ماتریس به صورت **row-wise** ، از تابع **np.add** استفاده میکنیم

در متد **column-wise** ابتدا ماتریس دوم را به صورت **1*n** در آورده سپس از تابع **np.add** استفاده میکنیم.

4)

ابتدا با استفاده از تابع **np.random.randint** یک آرایه 4*4 با مقادیر صحیح بین 1تا10 تولید میکنیم.
با استفاده از فرمول زیر دیتایی که دارین را نورمالایز میکنیم.

Normalized_data = (data – min(data)) / (max(data) – min(data))

5)

برای حل این سوال از کتابخانه **pandas** استفاده کردم و ابتدا فایل را میخوانیم

در این سوال با توجه به اینکه از مقدار بازده روزانه زیاد استفاده کردیم ستونی را اضافه کردیم که این مقادیر را حساب کند(طبق قرمول گفته شده)

** در رابطه با اولین دیتای داده شده چون نمیتوانیم میزان بازده روزانه آن را از روز قبل بدست آوریم مقدار آنرا در ستون بازده روزانه با استفاده از تابع **np.nan** برابر **NaN** قرار دادیم

پارت 1) بعد از اضافه کردن ستون بازده روزانه آنرا پرینت میگیریم

پارت2) میانگین را با استفاده از تابع **mean** بدست می آوریم

پارت3) انحراف معیار را از تابع **std** بدست می آوریم.

پارت 4 و 5) نمایش نمودار قیمت بسته شده و بازده روزانه را با استفاده از تابع **plot** در کتابخانه **matplotlib** انجام میدهیم

پارت6 و 7) برای بدست آوردن بیشترین و کمترین مقدار موجود در یک ستون از توابع **min , max** استفاده میکنیم.

6)

در این سوال در روش **for loop feed forward** از دو حلقه تو در تو **for** استفاده کرده و مقادیر مربوط به هر دو ماتریس در هم ضرب میکنیم

برای روش **vectorization** از کتابخانه موجود در نامپای یعنی تابع **dot** استفاده میکنیم

در خروجی مربوط به این سلول در نوت بوک زمان صرف شده با استفاده از هر روش مشخص است که استفاده از روش **vectorization** **سریعتر** است و **زمان کمتری** صرف میکند

** در روش **for_loop** بعد از کد اصلی ، کد دیگری کامنت شده که کاملتر از کد فعلی است و برای زمانی است که ماتریس w اندازه ای به مقدار **n * k** داشته باشد اما چون در این سوال مقدار k برابر 1 میباشد از نوشتن یک حلقه for دیگر پرهیز کردیم.

7)

با استفاده از عملیات بر روی ایندکس های آرایه، مقادیری که کمتر از **threshold** هستند را صفر و مقادیر بیشتر از آنرا برابر 1 قرار دادیم.

8)

*تابع **shape_comparison** : این تابع را خودم اضافه کردم. در این تابع شکل دو تابع را مقایسه میکنیم که در توابع بعدی مورد استفاده قرار میگیرد

تابع is_equal : در این تابع ابتدا با استفاده از تابع قبلی شکل دو آرایه را مقایسه میکنیم اگر برابر نباشند یعنی دو آرایه برابر نیستند.
سپس با استفاده از دو حلقه **for** ، بر روی دو آرایه پیمایش میکنیم و عنصر به عنصر مقایسه انجام میدهیم در صورتی که تمام عناصر دو آرایه برابر بودند ؛ این دو آرایه را برابر اعلام میکنیم.

تابع **is_higher_elementwise** :
ابتدا برای مقایسه دو ماتریس برابر بودن شکل آنها را بررسی میکنیم.

سپس برای ماتریس خروجی ماتریسی به نام **res** تعریف میکنیم و تمام مقادیر آنرا برابر **false** قرار میدهیم.
سپس با استفاده از حلقه**for** و پیمایش روی هر دو ماتریس و مقایسه عنصر به عنصر عناصری که بزرگتر از مقادیر متناظر در ماتریس دیگر هستند را در آرایه **res** برابر **true** قرار میدهیم و در غیر اینصورت برابر **false** باقی میماند

تابع is_subset :

ابتدا دو متغیر res,idxرا برای حالات مختلف ماتریکس به اندازه شکل ماتریس کوچکتر و مورد بررسی قرار میدهیم.

سپس با استفاده از حلقه های forی که تعریف کردیم پیمایش انجام میدهیم:
اگر مقادیر داده شده را در ماتریکس بزرگتر و اصلی یافتیم مقدار آنرا در **res**،**true** میکنیم سطر مقدار پیدا شده را در ماتریکس بزرگتر پیدا میکنیم و در ماتریکس **idx** قرار میدهیم

توضیح حالات مختلف:

اگر مقادیر در **res** همگی **true** نباشند مشخصا زیرمجموعه ماتریکس بزرگتر نیستند چون در این ماتریکس مقداری از ماتریکس دوم قرار ندارد.

اگر مقادیر **idx** در هر **element** از آن دارای دو مقدار برابر نباشند متوجه میشویم که این دو مقدار در یک عنصر از ماتریکس بزرگتر قرار ندارند(**باگی که در متغیر matrix6 داریم و باعث میشود زیر مجموعه نباشد)**

تابع **dot_product** :

این تابع ضرب دو ماتریس را با استفاده از حلقه های for انجام میدهیم.ابتدا ضرب ستون و سطر را با هم جمع میکنیم سپس در یک لیست دیگر قرار میدهیم تا با همین سطر ستون دیگری را ضرب کنیم و باز در لیست قرار دهیم و در لیست خروجی نهایی قرار دهیم سپس لیست را خالی میکنیم و این کار را تکرار میکنیم.