In the name of God



استاد : دکتر هراتی

دانشجو : توحید حقیقی سیس

شماره دانشجویی : 830598021

موضوع : تمرین اول

**تمرین اول :**

**در این تمرین چون ستون های مشترک ندارند تبدیل به ضرب میشود و عمل ضرب به جای natural join روی داده ها انجام میشود .**

**این مسئله را به 2 روش انجام دادم که روش اول در کلاستر ها قابل انجام است :**

**روش 1 :**

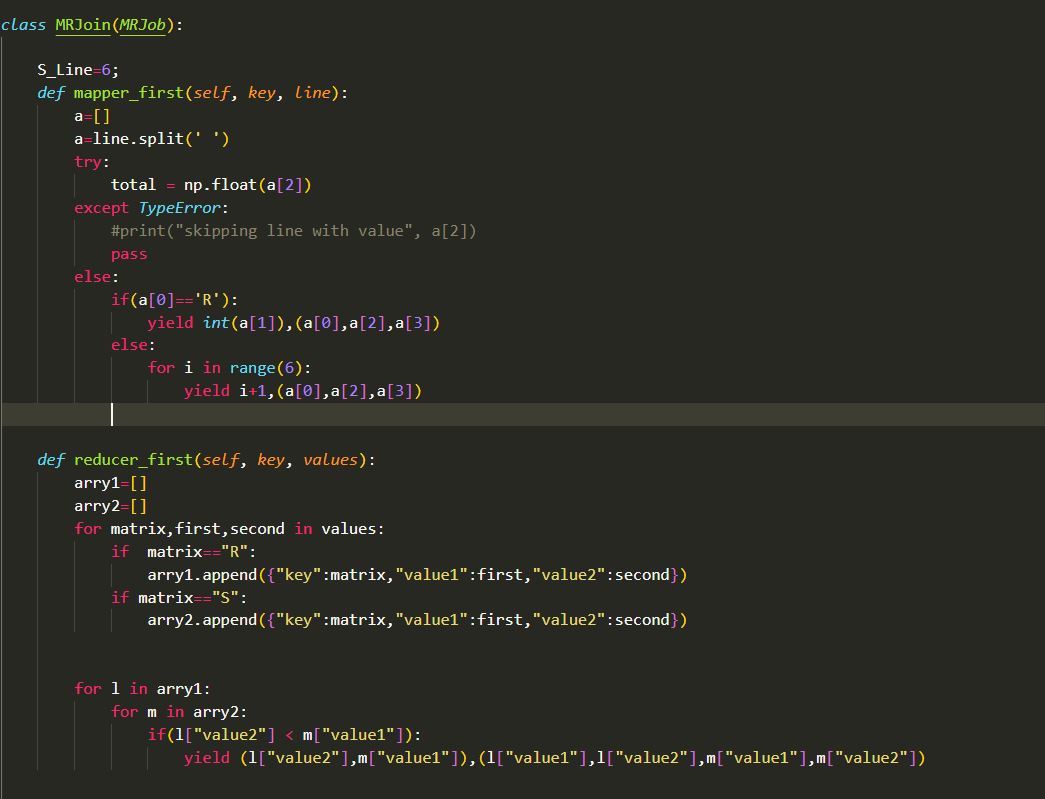
**در 1 مرحله map و reducer قابل انجام است .**

**به این صورت که ما 2 جدول R , S داریم و میخواهیم ان ها را در هم ضرب کنیم و بعد شرط را روی ان اعمال کنیم .**

|  |  |  |
| --- | --- | --- |
| **Value2** | **Value1** | **R** |
| **0** | **1** | **1** |
| **1** | **4** | **2** |
| **0** | **9** | **3** |
| **1** | **1** | **4** |
| **6** | **3** | **5** |
| **4** | **9** | **6** |

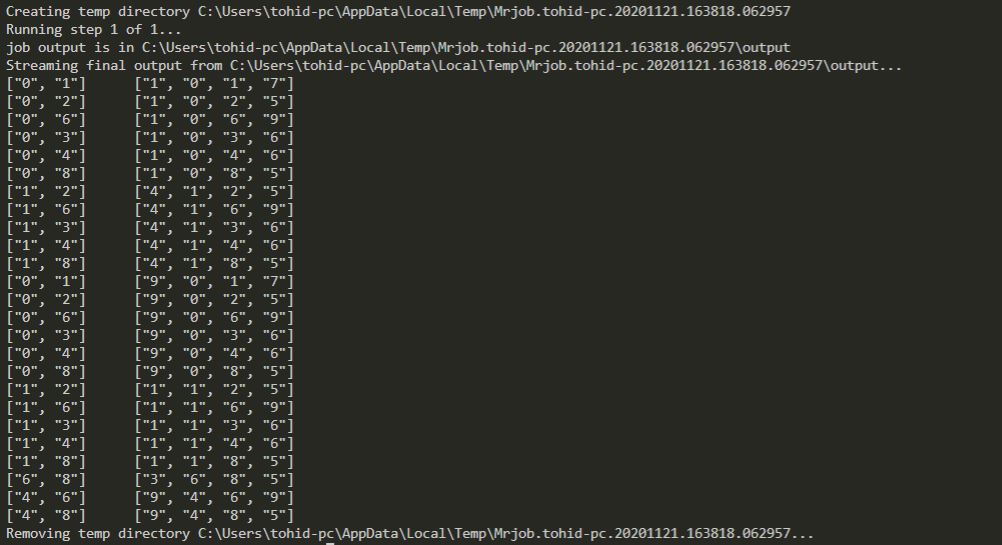
|  |  |  |
| --- | --- | --- |
| **Value2** | **Value1** | **S** |
| **7** | **1** | **1** |
| **5** | **2** | **2** |
| **9** | **6** | **3** |
| **6** | **3** | **4** |
| **6** | **4** | **5** |
| **5** | **8** | **6** |

**کد برنامه به صورت زیر میباشد :**

****

**در Map اول به این صورت عمل میکنیم که چون در ضرب یک سطر از R در کل سطر های S ضرب میشود پس باید کل سطر های S همراه با یک سطر از R به یک Reducer ارسال شود .**

**برای این کار کل سطر های S را با تمام کلید های 1 تا 6 کپی بر میداریم در این صورت همه ان ها به یک Reducer میرسد و میتوانیم ان ها را در هم ضرب کنیم و در خروجی نشان دهیم برای مثال دو جدول بالا را به کد نوشته شده داده ام و جواب زیر را به من داد .**

****

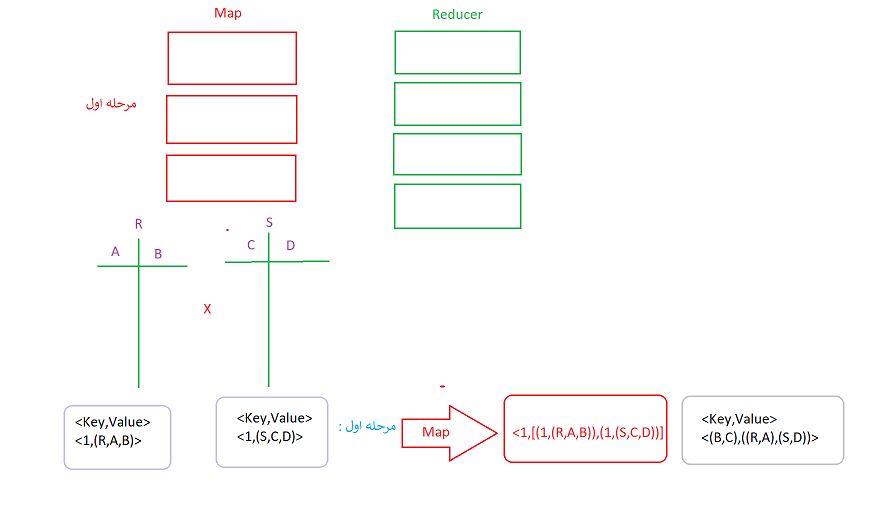
**وقتی در Reducer اول عمل ضرب رو انجام دادیم بعد شرط B<C را روی ان اعمال میکنیم و هر کدام این شرط در ان ها صادق بود را Yeild میکنیم .**

**در این حالت به تعداد سطر های جدول R و Reducer داریم و تمام اعمال سربار یک Reducer نمیشود .**

**[کد را اینجا ببینید](https://github.com/tohidhaghighy/MrjobRecommander/blob/main/Mrjob.py)**

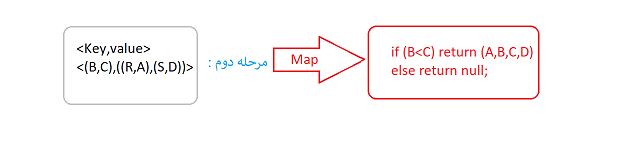
**روش 2 : این روش به نظر من مشکل دارد چون فعالیت ها به یک reducer منتقل میشود .**

**در 2 مرحله Map و Reduce انجام میشود و به این صورت است که در مرحله اول عمل ضرب انجام میشود و در مرحله دوم شرط روی داده ها اعمال میشود .**

****

**مرحله اول به صورت بالا میباشد که یک کلید مشترک میگیریم که همه مقادیر را برای اعمال ضرب وارد یک Reduce کنیم پس کلید را <1> میگیریم و در جدول اول value را (R,A,B) و در جدول دوم value را (S,C,D) میگیریم .**

**و همه را به یک Reudcer منتقل میکنیم در اینجا در این مرحله میتوانیم عمل ضرب را انجام دهیم ولی چون یک شرط نیز برای آن وجود دارد ما باید مقادیر را با کلید مقادیر شرط به یک Reduce دیگر منتقل کنیم .**

****

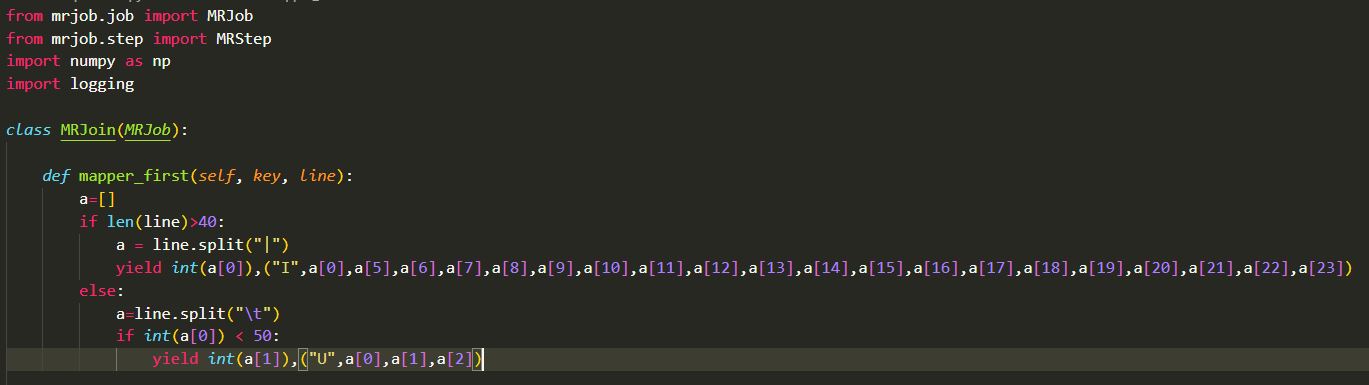
**و در Reduce دوم شرط را بررسی میکنیم اگر شرط برقرار بود سطر جدول را بر میگردانیم .**

**تمرین دوم :**

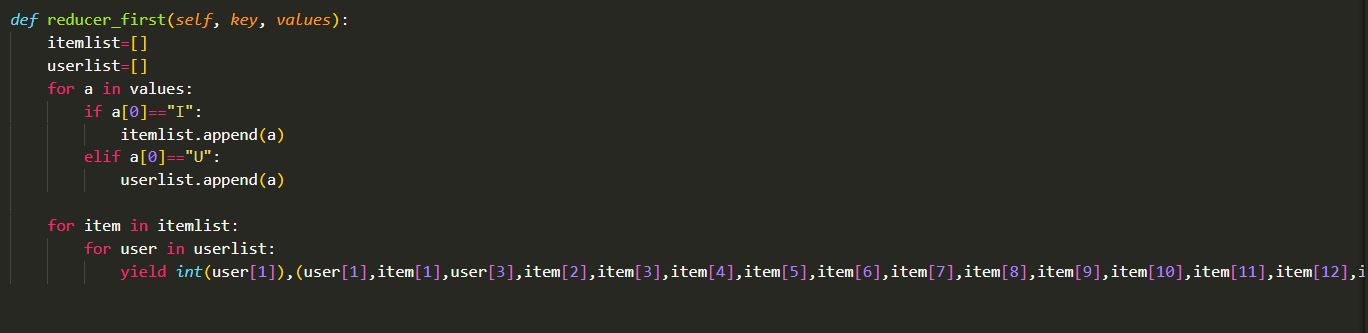
**بر اساس داده های موجود در جدول u.data و u.item میتوانیم با 2 مرحله Map-reduce جواب مورد نظر را پیدا کنیم .**

**در مرحله اول میتوانیم کلید را itemid بگیریم که در جدول یوزر در ستون دوم و در جدول ویدیو ها در ستون اول قرار دارد و خروجی این مرحله این خواهد بود که natural join بین دو جدول u.data,u.item بر اساس itemid**

**در مرحله اول کل اطلاعات فیلم را به جدول یوزر اضافه میکند و در مرحله دوم میتوانیم کلید را userid بگذاریم و اطلاعات مورد نظر هر یوزر در هر reducer جدا قرار میگیرد و میتوانیم ببینیم بیشتر به چه ganre فیلم امتیاز داده است و به چه نوع فیلم هایی علاقه دارد .**

****

**در mapper اول بر اساس کلید itemid ان ها را natural join میکنیم تا اطلاعات یوزر و فیلم به هم بچسبند و بتوانیم از ان استفاده کنیم از جدول یوزر ّا دستور splite(“\t”) میتوانیم سطر ها را جدا کنیم و ستون دوم را کلید بگیریم و از جدول دوم که جدول فیلم است با دستور splite(“|”) سطر ها را جدا میکنیم و ستون اول را کلید میگیریم .**

****

**در قسمت قبل itemid کلید بود در reducer اول کلید را یوزر گیریم و به ازای هر یوزر یک reduce اختصاص میدهیم .**

**در Reducer بعدی 2 دیکشنری تعریف میکنیم یکی تعداد ژانر ها و دیگری ژانر \* rating را در خود نگه میدارد .**

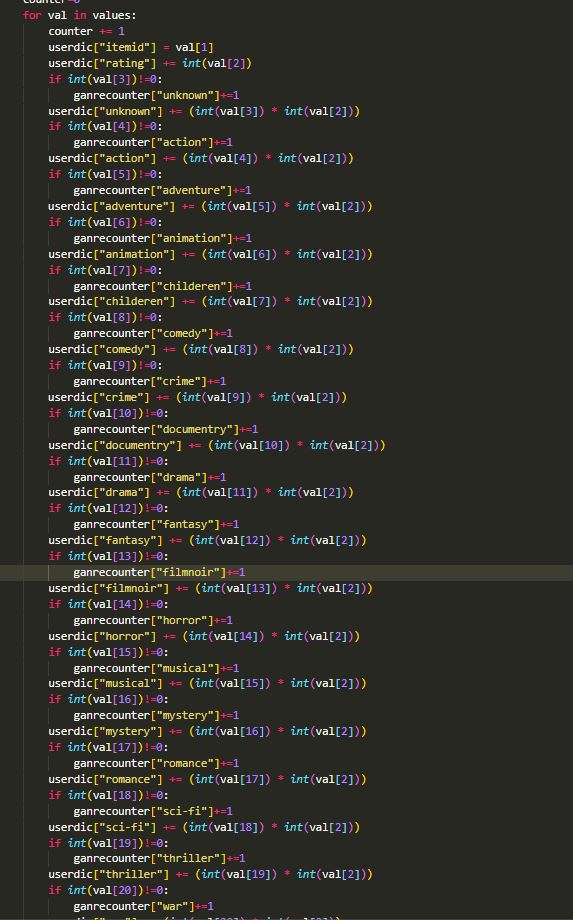
 userdic = {"key":key,"itemid":0,"rating":0,"unknown":0,"action":0,"adventure":0,"animation":0,"childeren":0,"comedy":0,"crime":0,"documentry":0,"drama":0,"fantasy":0,"filmnoir":0,"horror":0,"musical":0,"mystery":0,"romance":0,"sci-fi":0,"thriller":0,"war":0,"western":0,}

        ganrecounter = {"key":key,"itemid":0,"rating":0,"unknown":0,"action":0,"adventure":0,"animation":0,"childeren":0,"comedy":0,"crime":0,"documentry":0,"drama":0,"fantasy":0,"filmnoir":0,"horror":0,"musical":0,"mystery":0,"romance":0,"sci-fi":0,"thriller":0,"war":0,"western":0,}

**داخل یک For تمام مقادیر ارسالی از مرحله قبل را محاسبه چون هر reducer مربوط به یک یوزر است .**

**میتوانیم میزان علاقه هر یوزر به ژانر ها را محاسبه کنیم .**

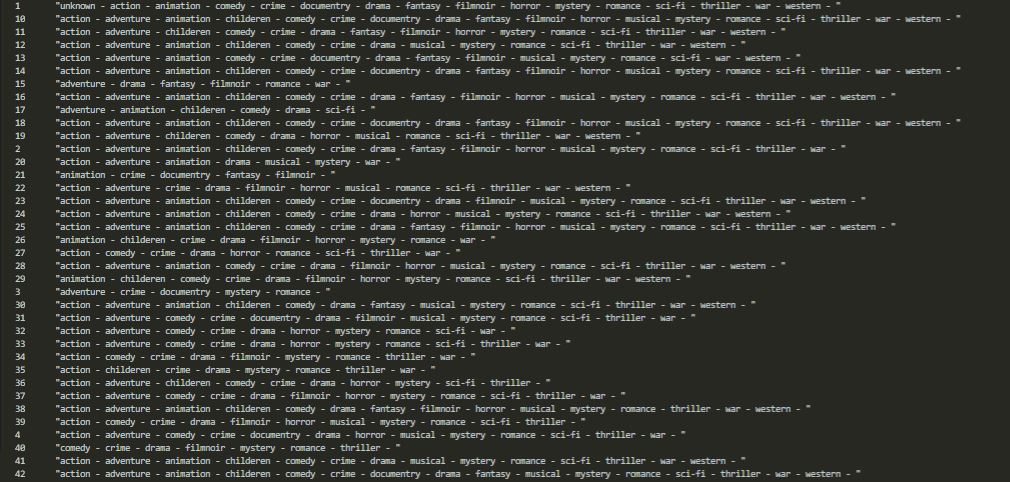
**و بعد از محاسبه مقادیر و تقسیم ان ها بر تعداد ژانر ها میتوانیم اگر از 3 بیشتر شد ان را به عنوان ژانر مورد علاقه کاربر برمیگردانیم .**

****

****

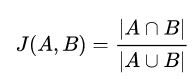
**خروجی این برنامه به صورت زیر خواهد بود .**

[**کد را اینجا ببینید**](https://github.com/tohidhaghighy/MrjobRecommander/blob/main/Recommander/Recomander/RecommanderMapReduce.py)

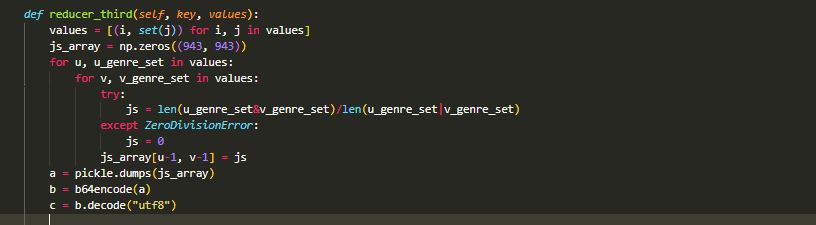
****

**در بخش دوم این سوال** حال باید کابران مشابه را بر اساس این ژانرها پیدا کنیم **برای کاهش تعداد مقایسات هر کاربر با فقط یک کاربر دیگر بصورت تصادفی مقایسه میشود.** برای اینکار از شباهت جاکارد استفاده میکنیم. کاربرانی را مشابه فرض میکنیم که شباهت آنها برابر یا بزرگتر از 0.5 باشد .

برای حل این بخش از قسمت قبل استفاده میکنیم با این تفاوت که قسمت قبل برای 50 یوزر بود ولی این بخش از کل یوزر ها که 496 تا هست استفاده میکنم خروجی قسمت قبل رو تبدیل به Set در پایتون میکنم تا بتوانم روش راحت تر اجتماع و اشتراک بگیرم چون شباهت ژاکارد از فرمول زیر به دست می آید .

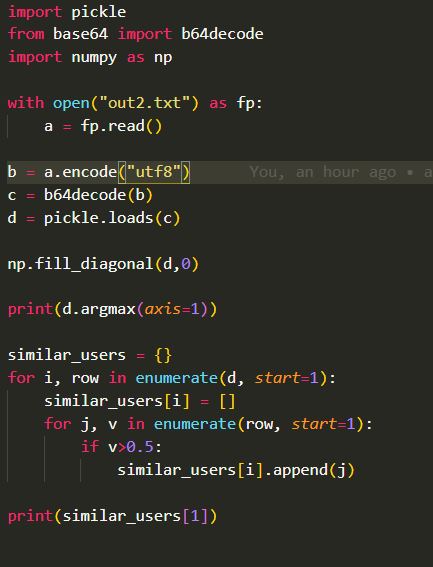


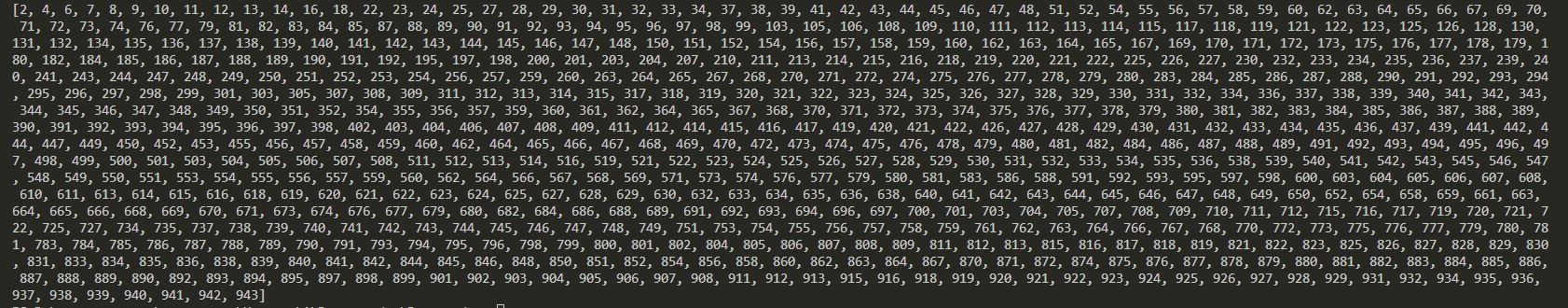
**همه داده های reducer هر یوزر را وارد یک reducer میکنم و روی ان شباهت ژاکارد میگیرم و یک خروجی 496\*496 میشود چون همه کاربران را با هم شباهت ژاکارد میگیرم .**

****

[**کد را اینجا ببینید**](https://github.com/tohidhaghighy/MrjobRecommander/blob/main/Recommander/Recomander/Recommander.py)

**در قسمت قبل ماتریس numpy شباهت ژاکارد را تشکیل داده ام و در out.txt ذخیره کرده ام .**

****

****

**تمرین سوم :**

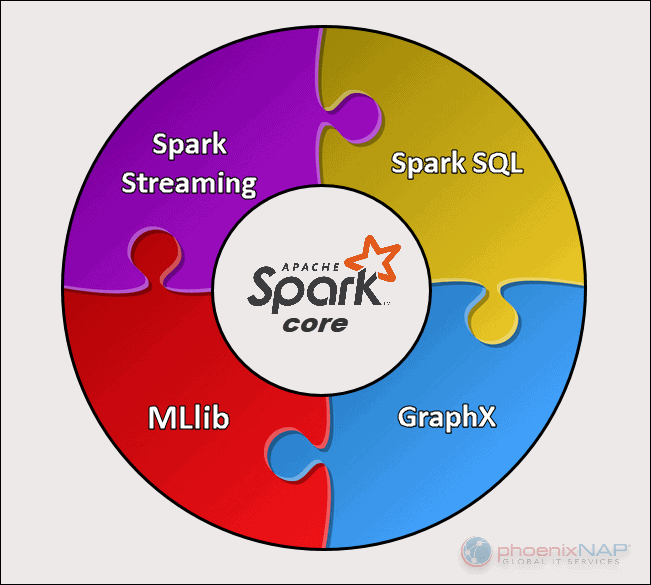
در یک یا دو صفحه Spark و Hadoop را با هم مقایسه کنید. همچنین بطور مختصر Apache  
Storm و Apache Kafka را معرفی کنید. با درج رفرنس در کنار مطالب خود، آنها را به ما نیز معرفی کنید .

اسپارک :

به عنوان موتور سریع و عمومی برای پردازش داده های در مقیاس بزرگ ، توسط توسعه دهندگان Apache Spark شناخته می شود. اگر بخواهیم این دو محیط برنامه نویسی را با هم مقایسه کنیم ، در آنجا که Hadoop به نظر می رسد 800 لیتر گوریل باشد ، Spark یوزپلنگ بزرگ 130 پوند است. استنباط شده است که از نظر جرقه سریعتر از نظر پردازش در حافظه ، در مقایسه با Hadoop و MapReduce قرار دارد. اما بسیاری بر این باورند که ممکن است سرعت پردازش در فضای دیسک خیلی سریع نباشد. آنچه Spark در حقیقت از آن برتری دارد ، بی زحمت بودن پرس و جوهای تعاملی ، بار کاری و از همه مهمتر یادگیری ماشین است.

اسپارک دارای 5 ماژول اصلی میباشد :

* Apache Spark Core
* Spark Streaming
* Spark Sql
* MLLib
* GraphX



هدوپ :

یک پروژه Apache.org است ، که به راحتی با استفاده از مدل های برنامه نویسی ، قادر به توزیع نرم افزاری است که مجموعه داده های بزرگ را برای تعدادی از خوشه های رایانه پردازش می کند. Hadoop یکی از این نرم افزارهاست که قادر است از یک سیستم محاسباتی منفرد تا نزدیک به هزاران سیستم کالایی که به آنها امکان ذخیره سازی محلی و قدرت رایانه را می دهد ، مقیاس کند. به بیان ساده تر ، می توانید از Hadoop به عنوان گوریل داده بزرگ 800 پوند در فضای بزرگ تحلیل داده استفاده کنید. این یکی از دلایل محبوبیت استفاده از این برنامه نرم افزاری خاص در بین تحلیلگران داده است.

هدوپ دارای 4 ماژول اصلی میباشد :

* HDFS - [Hadoop Distributed File System](https://phoenixnap.com/kb/what-is-hdfs)

یک فایل سیستمی است که مدیریت داده های بزرگ را در کلاستر ها را بر عهده دارد.

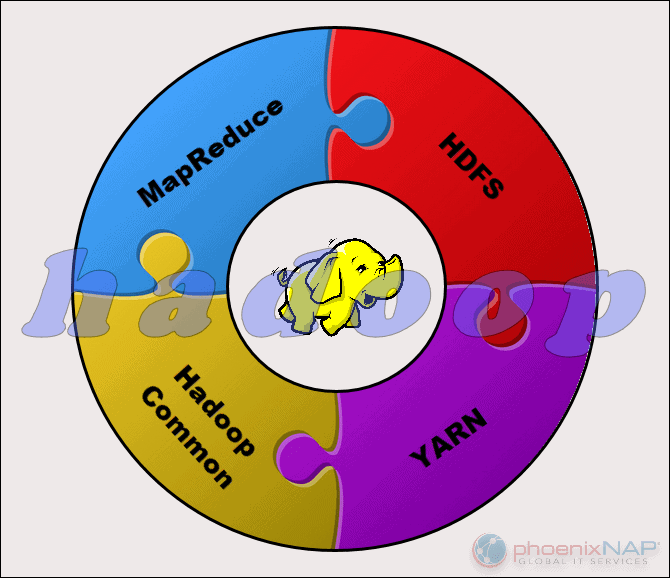
* MapReduce

الگوریتمی است که برای توزیع داده ها در کلاستر ها استفاده میشود .

* Yarn - Yet Another Resource Negotiator

مدیریت منابع محاسباتی و تقسیم بندی وظایف

* Hadoop Common



امروزه با رشد روز افزون ابزارهایی مانند شبکه های اجتماعی و ظهور مفاهیمی همچون وب معنایی حجم داده ها و پردازش آنها بطور شگفت انگیزی افزایش یافته است . برای نمونه یک موتور جستجو در کسری از ثانیه نتایج جستجوهای کاربر را آماده می کند که این ناشی از تحلیلی کارآمد روی داده های حجیم جمع آوری شده از سطح وب می باشد ، بنابراین وجود مکانیزمی برای پردازش داده های حجیم با هزینه ای مقرون به صرفه بسیار پر اهمیت است . در این مقاله قصد داریم الگوریتم های ترکیبی در چارچوب ماهوت براساس مدل Hadoop- Mapreduce و Spark را مورد مقایسه قرارداده و نقاط ضعف و قوت هریک را بررسی نماییم . نتایج مقایسات نشان می دهد که اسپارک علاوه بر داشتن تمامی نقاط هدوپ نقاط ضعف آن را نیز توسط پردازش درون حافظه ای ، سرعت بالاتر ، دارا بودن توابع API فراوان که این امر کار را برای برنامه نویسان و استفاده کنندگان راحت تر می کند ، قابلیت برنامه نویسی در پایتون ، اسکالا ، جاوا و حتی زبانR و موتور پردازش متحد و یکپارچه نیز بخشی از نقاط قوت اسپارک می باشد . حال آن که Mapreduce یک مدل پردازش داده با مقیاس پذیری آسان پردازش داده روی چندین گره محاسباتی است این پاردایم برنامه را در دو فاز نگاشت و کاهش اجرا می کند.

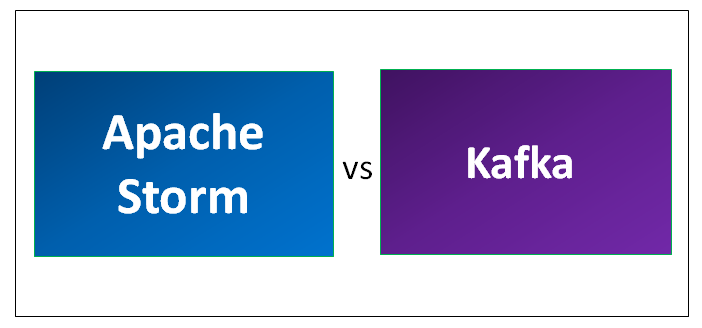
Hadoop و Spark هر دو برای پردازش های موازی به وجود امده اند spark نقاط ضعف Hadoop را رفع کرده اما با این وجود هنوز شرکت های زیادی از Hadoop استفاده میکنند الگوریتم mapreduce در Hadoop برای تقسیم داده ها در کلاستر ها کاربرد دارد به این صورت که هر داده بر اساس کلید خود به reducer مخصوص خود ارسال میشود و در انجا پردازش های لازم روی ان انجام میشود .

Refrences:

* <https://logz.io/blog/hadoop-vs-spark/>
* <https://www.educba.com/hadoop-vs-spark/>
* <https://phoenixnap.com/kb/hadoop-vs-spark>

**بخش دوم :**

**تفاوت بین Apache storm vs Apache spark :**

****

**کافکا یک صف توزیع شده برای داده های بزرگ است که میتواند داده ها با حجم زیاد را کنترل کند و پیام ها را از یک نقطه به نقطه دیگر منتقل میکند ولی storm صف نیست و سیستمی است که توانایی پردازش در زمان واقعی را میتواند انجام دهد به این صورت که داده هایی که به یک سیستم ارسال میشود اول کافکا ان ها را گرفته و وارد سیستم میکند و storm رفته و از کافکا و یا هر سیتم دیگری داده ها را میگیرد و پردازش های لازم را روی ان ها انجام میدهد و در صورت نیاز ان ها را به Nosql ارسال میکند .**

**Refrences :**

* [**https://stackoverflow.com/questions/21808529/apache-kafka-vs-apache-storm**](https://stackoverflow.com/questions/21808529/apache-kafka-vs-apache-storm)
* [**https://www.educba.com/apache-storm-vs-kafka/**](https://www.educba.com/apache-storm-vs-kafka/)