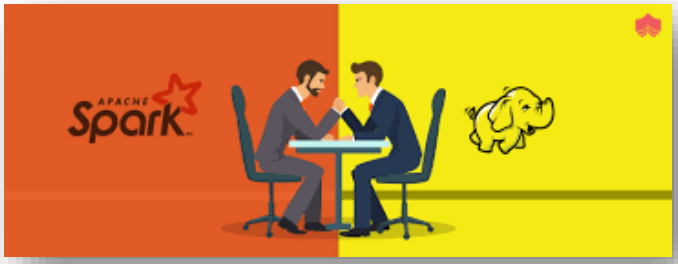
بسمه تعالی

مبانی داده کاوی

تمرین اول

الهام دهقانی _ 220797049



هدوپ چیست؟

Apache Hadoop پلتفرمی است که مجموعه داده های بزرگ را به صورت توزیع شده مدیریت می کند. این چارچوب از MapReduce برای تقسیم داده‌ها به بلوک‌ها و تخصیص تکه‌ها به گره‌ها در یک خوشه استفاده می‌کند. MapReduce داده ها را به صورت موازی در هر گره پردازش می کند تا یک خروجی منحصر به فرد تولید کند.

هر ماشین در یک خوشه هم داده ها را ذخیره و هم پردازش می کند. Hadoop داده ها را با استفاده از HDFS در دیسک ذخیره می کند. این نرم افزار گزینه های مقیاس پذیری یکپارچه را ارائه می دهد. می‌توانید با یک دستگاه شروع کنید و سپس به هزاران دستگاه گسترش دهید و هر نوع سخت‌افزار شرکتی یا کالایی را اضافه کنید.

اکوسیستم هدوپ بسیار مقاوم در برابر خطا است. Hadoop برای دستیابی به دسترسی بالا به سخت افزار وابسته نیست. در هسته خود، Hadoop برای جستجوی شکست در لایه برنامه ساخته شده است. با تکثیر داده‌ها در یک خوشه، زمانی که یک قطعه سخت‌افزاری از کار می‌افتد، چارچوب می‌تواند بخش‌های گمشده را از مکان دیگری بسازد.

پروژه Apache Hadoop از چهار ماژول اصلی تشکیل شده است:

- HDFS – سیستم فایل توزیع شده Hadoop. این سیستم فایلی است که ذخیره مجموعه بزرگی از داده ها را در یک خوشه Hadoop مدیریت می کند. HDFS می تواند داده های ساختاریافته و بدون ساختار را مدیریت کند. سخت افزار ذخیره سازی می تواند از هر HDD درجه مصرف کننده تا درایوهای سازمانی متغیر باشد.
- MapReduce. جزء پردازشی اکوسیستم هدوپ قطعات داده را از HDFS به جداسازی وظایف نقشه در خوشه اختصاص می دهد. MapReduce تکه ها را به صورت موازی پردازش می کند تا قطعات را به نتیجه دلخواه ترکیب کند.

- YARN. یک مذاکره کننده منابع دیگر. مسئول مدیریت منابع محاسباتی و زمان بندی کار.
- Hadoop Common. مجموعه ای از کتابخانه ها و ابزارهای کمکی رایج که سایر ماژول ها به آنها وابسته هستند. نام دیگر این ماژول هسته Hadoop است، زیرا از سایر اجزای Hadoop پشتیبانی می کند.

ماهیت Hadoop آن را برای هر کسی که به آن نیاز دارد قابل دسترسی می کند. جامعه منبع باز بزرگ است و مسیر را برای پردازش کلان داده در دسترس هموار کرده است.

اسپارک چیست؟

آپاچی اسپارک یک ابزار متن باز است. این فریم ورک می تواند در حالت مستقل یا روی یک مدیر ابری یا خوشه ای مانند Apache Mesos و سایر پلتفرم ها اجرا شود. برای عملکرد سریع طراحی شده است و از RAM برای ذخیره و پردازش داده ها استفاده می کند.

Spark انواع مختلفی از حجم کاری کلان داده را انجام می دهد. این شامل پردازش دسته‌ای شبیه MapReduce، و همچنین پردازش جریان بی‌درنگ، یادگیری ماشین، محاسبه نمودار، و پرس‌وجوهای تعاملی است. با استفاده آسان از API های سطح بالا، Spark می تواند با بسیاری از کتابخانه های مختلف از جمله PyTorch و TensorFlow ادغام شود.

موتور Spark برای بهبود کارایی MapReduce و حفظ مزایای آن ایجاد شده است. حتی اگر Spark سیستم فایل خود را ندارد، می تواند به داده ها در بسیاری از راه حل های ذخیره سازی مختلف دسترسی داشته باشد. ساختار داده ای که Spark استفاده می کند، Resilient Distributed Dataset یا RDD نامیده می شود.

پنج جزء اصلی آپاچی اسپارک وجود دارد:

- Apache Spark Core. اساس کل پروژه. Spark Core وظایف ضروری مانند زمان‌بندی، ارسال وظایف، عملیات ورودی و خروجی، بازیابی خطا و غیره را بر عهده دارد. سایر قابلیت‌ها در بالای آن ساخته شده‌اند.
- Spark streaming. این مؤلفه پردازش جریان های داده زنده را امکان پذیر می کند. داده ها می توانند از بسیاری از منابع مختلف، از جمله کافکا، کینزیس، فلوم و غیره سرچشمه بگیرند.
- Spark SQL . Spark از این مؤلفه برای جمع آوری اطلاعات در مورد داده های ساختار یافته و نحوه پردازش داده ها استفاده می کند.
- کتابخانه یادگیری ماشین (MLlib). این کتابخانه از بسیاری از الگوریتم های یادگیری ماشین تشکیل شده است. هدف MLlib مقیاس‌پذیری و در دسترس‌تر کردن یادگیری ماشینی است.
- GraphX. مجموعه ای از API که برای تسهیل وظایف تجزیه و تحلیل گراف استفاده می شود.

بخش‌های زیر به تشریح تفاوت‌ها و شباهت‌های اصلی بین این دو چارچوب می‌پردازند. ما نگاهی به Hadoop vs Spark از زوایای مختلف خواهیم داشت.

جدول زیر یک نمای کلی از نتیجه گیری های انجام شده در بخش های بعدی را ارائه می دهد.

مقایسه هدوپ و اسپارک

	هدوپ	اسپارک
عملکرد	عملکرد کندتر، از دیسک ها برای ذخیره سازی استفاده می کند و به سرعت خواندن و نوشتن دیسک بستگی دارد.	عملکرد سریع در حافظه با کاهش عملیات خواندن و نوشتن دیسک.
هزینه	یک پلت فرم منبع باز، هزینه کمتر برای اجرا. از سخت افزار مصرف کننده مقرون به صرفه استفاده می کند. یافتن متخصصان آموزش دیده Hadoop آسان تر است.	یک پلت فرم منبع باز، اما برای محاسبات به حافظه متکی است، که به طور قابل توجهی هزینه های جاری را افزایش می دهد.
پردازش داده	بهترین برای پردازش دسته ای. از MapReduce برای تقسیم یک مجموعه داده بزرگ در یک خوشه برای تجزیه و تحلیل موازی استفاده می کند.	مناسب برای تجزیه و تحلیل داده های تکراری و پخش زنده. برای اجرای عملیات با RDD و DAG کار می کند.
تحمل خطا	یک سیستم بسیار مقاوم در برابر خطا. داده ها را در سراسر گره ها تکرار می کند و در صورت بروز مشکل از آنها استفاده می کند.	فرآیند ایجاد بلوک RDD را ردیابی می کند، و سپس می تواند یک مجموعه داده را در صورت خرابی یک پارتیشن بازسازی کند. Spark همچنین می تواند از DAG برای بازسازی داده ها در سراسر گره ها استفاده کند.
مقیاس پذیری	با افزودن گره ها و دیسک ها برای ذخیره سازی به راحتی مقیاس پذیر است. از ده ها هزار گره بدون محدودیت شناخته شده پشتیبانی می کند.	مقیاس کردن کمی چالش برانگیزتر است زیرا برای محاسبات به RAM متکی است. از هزاران گره در یک خوشه پشتیبانی می کند.
امنیت	بسیار امن. از LDAP، ACL، Kerberos، SLA و غیره پشتیبانی می کند.	ایمن نیست. به طور پیش فرض، امنیت خاموش است. برای دستیابی به سطح امنیتی لازم به ادغام با Hadoop متکی است.
سهولت استفاده و پشتیبانی زبان	استفاده از آن با زبان های کمتر پشتیبانی شده دشوارتر است. از جاوا یا پایتون برای برنامه های MapReduce استفاده می کند.	کاربر پسندتر به حالت پوسته تعاملی اجازه می دهد. API ها را می توان در Java، Scala، R، Python، Spark SQL نوشت.
یادگیری ماشین	کندتر از Spark قطعات داده می توانند خیلی بزرگ باشند و گلوگاه ایجاد کنند. Mahout کتابخانه اصلی است.	با پردازش در حافظه بسیار سریعتر. از MLlib برای محاسبات استفاده می کند.

برنامه ریزی و مدیریت منابع	از راه حل های خارجی استفاده می کند. YARN رایج ترین گزینه برای مدیریت منابع است. Oozie برای برنامه ریزی گردش کار در دسترس است.	دارای ابزارهای داخلی برای تخصیص منابع، زمان‌بندی و نظارت است.

کارایی

وقتی نگاهی به Hadoop در مقابل Spark از نظر نحوه پردازش داده ها بیندازیم، ممکن است مقایسه عملکرد این دو چارچوب طبیعی به نظر نرسد.

با دسترسی به داده های ذخیره شده محلی در HDFS، Hadoop عملکرد کلی را افزایش می دهد. با این حال، برای پردازش درون حافظه اسپارک مناسب نیست. طبق ادعای آپاچی، اسپارک در هنگام استفاده از رم برای محاسبات ۱۰۰ برابر سریعتر از Hadoop با MapReduce است.

تسلط با مرتب سازی داده ها روی دیسک ها باقی ماند. Spark ۳ برابر سریعتر بود و برای پردازش ۱۰۰ ترابایت داده در HDFS به ۱۰ برابر گره های کمتری نیاز داشت. این معیار برای ثبت رکورد جهانی در سال ۲۰۱۴ کافی بود.

دلیل اصلی این برتری Spark این است که داده های میانی را روی دیسک نمی خواند و نمی نویسد بلکه از RAM استفاده می کند. Hadoop داده ها را در بسیاری از منابع مختلف ذخیره می کند و سپس داده ها را به صورت دسته ای با استفاده از MapReduce پردازش می کند.

همه موارد فوق ممکن است اسپارک را به عنوان برنده مطلق معرفی کند. با این حال، اگر اندازه داده ها بزرگتر از RAM موجود باشد، Hadoop انتخاب منطقی تری است. نکته دیگری که باید به آن توجه کرد هزینه اجرای این سیستم ها است.

هزینه

مقایسه Hadoop در مقابل Spark با در نظر گرفتن هزینه، باید بیشتر از قیمت نرم افزار جستجو کنیم. هر دو پلتفرم منبع باز و کاملا رایگان هستند. با این وجود، هزینه های زیرساخت، نگهداری و توسعه باید در نظر گرفته شود تا هزینه کل مالکیت تقریبی (TCO) به دست آید.

مهمترین عامل در مقوله هزینه، سخت افزار زیربنایی است که برای اجرای این ابزارها نیاز دارید. از آنجایی که Hadoop برای پردازش داده ها به هر نوع ذخیره سازی دیسک متکی است، هزینه اجرای آن نسبتا پایین است.

از سوی دیگر، Spark به محاسبات درون حافظه برای پردازش داده ها در زمان واقعی بستگی دارد. بنابراین، چرخش گره‌ها با رم زیاد، هزینه مالکیت را به میزان قابل توجهی افزایش می‌دهد.

نگرانی دیگر توسعه اپلیکیشن است. Hadoop بیشتر از Spark وجود داشته است و یافتن توسعه دهندگان نرم افزار کمتر چالش برانگیز است.

نکات بالا نشان می دهد که زیرساخت هدوپ مقرون به صرفه تر است. در حالی که این جمله درست است، باید یادآور شویم که Spark داده ها را بسیار سریعتر پردازش می کند. از این رو، برای انجام همان کار به تعداد کمتری ماشین نیاز دارد.

پردازش داده ها

این دو چارچوب داده ها را به روش های کاملا متفاوتی مدیریت می کنند. اگرچه Hadoop با MapReduce و Spark با RDD داده ها را در یک محیط توزیع شده پردازش می کنند، Hadoop برای پردازش دسته ای مناسب تر است. در مقابل، Spark با پردازش بلادرنگ می درخشد.

هدف Hadoop ذخیره داده ها بر روی دیسک ها و سپس تجزیه و تحلیل موازی آن ها به صورت دسته ای در یک محیط توزیع شده است. MapReduce برای مدیریت حجم وسیعی از داده ها به مقدار زیادی RAM نیاز ندارد. Hadoop برای ذخیره سازی به سخت افزار روزمره متکی است و برای پردازش داده های خطی مناسب ترین است.

Apache Spark با مجموعه داده های توزیع شده انعطاف پذیر (RDD) کار می کند. RDD مجموعه ای توزیع شده از عناصر است که در پارتیشن های روی گره ها در سراسر خوشه ذخیره می شود. اندازه یک RDD معمولاً برای یک گره بزرگ است. بنابراین Spark، RDD ها را به نزدیک ترین گره ها پارتیشن بندی می کند و عملیات را به صورت موازی انجام می دهد. این سیستم تمام اقدامات انجام شده بر روی یک RDD را با استفاده از یک گراف غیر چرخشی جهت دار (DAG) ردیابی می کند.

با محاسبات درون حافظه و API های سطح بالا، Spark به طور موثر جریان های زنده داده های بدون ساختار را کنترل می کند. علاوه بر این، داده ها در تعداد از پیش تعریف شده پارتیشن ذخیره می شوند. یک گره می تواند به تعداد مورد نیاز پارتیشن داشته باشد، اما یک پارتیشن نمی تواند به گره دیگری گسترش یابد.

تحمل خطا

صحبت از Hadoop در مقابل Spark در رده تحمل خطا، می‌توان گفت که هر دو سطح قابل قبولی از خرابی‌ها را ارائه می‌دهند. همچنین، می توان گفت که نحوه برخورد آنها با تحمل خطا متفاوت است.

Hadoop تحمل خطا را به عنوان اساس عملکرد خود دارد. این داده ها را بارها در سراسر گره ها تکرار می کند. در صورت بروز مشکل، سیستم با ایجاد بلوک‌های گمشده از مکان‌های دیگر، کار را از سر می‌گیرد. گره های اصلی وضعیت همه گره های برده را دنبال می کنند. در نهایت، اگر یک گره slave به پینگ های یک master پاسخ ندهد، master job های معلق را به گره برده دیگری اختصاص می دهد.

Spark از بلوک های RDD برای دستیابی به تحمل خطا استفاده می کند. این سیستم نحوه ایجاد مجموعه داده تغییرناپذیر را ردیابی می کند. سپس در صورت بروز مشکل می تواند فرآیند را مجدداً راه اندازی کند. Spark می تواند داده ها را در یک خوشه با استفاده از ردیابی DAG از گردش کار بازسازی کند. این ساختار داده اسپارک را قادر می‌سازد تا خرابی‌های یک اکوسیستم پردازش داده توزیع‌شده را مدیریت کند.

مقیاس پذیری

خط بین Hadoop و Spark در این بخش تار می شود. Hadoop از HDFS برای مقابله با داده های بزرگ استفاده می کند. هنگامی که حجم داده ها به سرعت رشد می کند، Hadoop می تواند به سرعت مقیاس شود تا تقاضا را برآورده کند. از آنجایی که Spark سیستم فایل خود را ندارد، زمانی که داده ها برای مدیریت آن بسیار بزرگ هستند، باید به HDFS تکیه کند.

خوشه ها می توانند به راحتی با افزودن سرورهای بیشتر به شبکه، قدرت محاسباتی را گسترش داده و افزایش دهند. در نتیجه، تعداد گره ها در هر دو چارچوب می تواند به هزاران عدد برسد. هیچ محدودیت قطعی برای اینکه چه تعداد سرور می توانید به هر خوشه اضافه کنید و چه مقدار داده می توانید پردازش کنید وجود ندارد.

برخی از اعداد تایید شده شامل ۸۰۰۰ ماشین در محیط Spark با پتابایت داده است. وقتی صحبت از خوشه‌های هدوپ می‌شود، به خوبی شناخته شده‌اند که ده‌ها هزار ماشین و نزدیک به یک اگزابایت داده را در خود جای می‌دهند.

سهولت استفاده و پشتیبانی از زبان برنامه نویسی

Spark ممکن است فریمورک جدیدتری باشد که به اندازه Hadoop متخصصان در دسترس نیست، اما کاربرپسندتر شناخته شده است. در مقابل، Spark از چندین زبان در کنار زبان اصلی (Scala) پشتیبانی می کند: جاوا، پایتون، R و Spark SQL. این به توسعه دهندگان اجازه می دهد تا از زبان برنامه نویسی مورد علاقه خود استفاده کنند.

چارچوب Hadoop بر اساس جاوا است. دو زبان اصلی برای نوشتن کد MapReduce جاوا یا پایتون هستند. Hadoop حالت تعاملی برای کمک به کاربران ندارد. با این حال، با ابزارهای Pig و Hive یکپارچه می شود تا نوشتن برنامه های پیچیده MapReduce را تسهیل کند.

علاوه بر پشتیبانی از API در چندین زبان، Spark در بخش سهولت استفاده با حالت تعاملی خود برنده است. می توانید از پوسته Spark برای تجزیه و تحلیل داده ها به صورت تعاملی با اسکالا یا پایتون استفاده کنید. این پوسته بازخورد فوری به درخواست‌ها ارائه می‌کند، که استفاده از Spark را نسبت به Hadoop MapReduce آسان‌تر می‌کند.

چیز دیگری که به Spark برتری می دهد این است که برنامه نویسان می توانند در صورت لزوم از کدهای موجود مجددا استفاده کنند. با انجام این کار، توسعه دهندگان می توانند زمان توسعه برنامه را کاهش دهند. داده های تاریخی و جریانی را می توان برای مؤثرتر ساختن این فرآیند ترکیب کرد.

امنیت

با مقایسه Security، Hadoop و Spark، Hadoop برنده واضح است. مهمتر از همه، امنیت Spark به طور پیش فرض خاموش است. این بدان معناست که اگر با این مشکل مقابله نکنید، تنظیمات شما لو می‌شود.

می توانید با معرفی احراز هویت از طریق راز مشترک یا ثبت رویداد، امنیت Spark را بهبود بخشید. با این حال، این برای حجم کاری تولید کافی نیست.

در مقابل، Hadoop با چندین روش احراز هویت و کنترل دسترسی کار می کند. سخت‌ترین روش اجرای احراز هویت Kerberos است. اگر Kerberos بیش از حد قابل کنترل باشد، Hadoop همچنین از Ranger، LDAP، ACLs، رمزگذاری بین گره، مجوزهای استاندارد فایل در HDFS و مجوز سطح سرویس پشتیبانی می کند.

با این حال، Spark می تواند با ادغام با Hadoop به سطح مناسبی از امنیت برسد. به این ترتیب Spark می تواند از تمام روش های موجود برای Hadoop و HDFS استفاده کند. علاوه بر این، وقتی Spark روی YARN اجرا می‌شود، می‌توانید از مزایای دیگر روش‌های احراز هویت که در بالا به آن اشاره کردیم استفاده کنید.

فراگیری ماشین

یادگیری ماشینی یک فرآیند تکراری است که با استفاده از محاسبات درون حافظه بهترین عملکرد را دارد. به همین دلیل، اسپارک راه حل سریع تری در این زمینه بود.

دلیل این امر این است که Hadoop MapReduce مشاغل را به وظایف موازی تقسیم می کند که ممکن است برای الگوریتم های یادگیری ماشین خیلی بزرگ باشند. این فرآیند مشکلات عملکرد I/O را در این برنامه های Hadoop ایجاد می کند.

کتابخانه Mahout اصلی ترین پلتفرم یادگیری ماشین در خوشه های Hadoop است. Mahout برای انجام خوشه بندی، طبقه بندی و توصیه به MapReduce متکی است. Samsara جایگزین این پروژه شد.

Spark با یک کتابخانه پیش‌فرض یادگیری ماشین، MLlib، عرضه می‌شود. این کتابخانه محاسبات تکراری ML در حافظه را انجام می دهد. این شامل ابزارهایی برای انجام رگرسیون، طبقه بندی، ماندگاری، ساخت خط لوله، ارزیابی و بسیاری موارد دیگر است.

ثابت شد که Spark با MLlib۹ برابر سریعتر از Apache Mahout در یک محیط مبتنی بر دیسک Hadoop است. هنگامی که به نتایج کارآمدتر از آنچه Hadoop ارائه می دهد نیاز دارید، Spark انتخاب بهتری برای یادگیری ماشین است.

برنامه ریزی و مدیریت منابع

Hadoop یک زمانبندی داخلی ندارد. از راه حل های خارجی برای مدیریت منابع و برنامه ریزی استفاده می کند. با Resource Manager و Node Manager، YARN مسئول مدیریت منابع در یک کلاستر Hadoop است. یکی از ابزارهای موجود برای زمانبندی گردش کار، Oozie است.

YARN با مدیریت دولتی برنامه های کاربردی سر و کار ندارد. فقط قدرت پردازش موجود را تخصیص می دهد.

Hadoop MapReduce با افزونه هایی مانند Capacity Scheduler و Fair Scheduler کار می کند. این زمان‌بندی‌ها تضمین می‌کنند که برنامه‌ها منابع ضروری را در صورت نیاز دریافت می‌کنند و در عین حال کارایی یک خوشه را حفظ می‌کنند. Fair Scheduler منابع لازم را در اختیار برنامه‌ها قرار می‌دهد در حالی که پیگیری می‌کند که در نهایت، همه برنامه‌ها تخصیص منابع یکسانی را دریافت می‌کنند.

از طرف دیگر اسپارک این عملکردها را در خود دارد. زمانبند DAG مسئول تقسیم اپراتورها به مراحل است. هر مرحله دارای وظایف متعددی است که DAG برنامه ریزی می کند و Spark باید آنها را اجرا کند.

Spark Scheduler و Block Manager برنامه ریزی، نظارت و توزیع منابع کار و وظایف را در یک خوشه انجام می دهند.

در موارد زیر از هدوپ در مقابل اسپارک استفاده کنید

با نگاهی به Hadoop در مقابل Spark در بخش‌های ذکر شده در بالا، می‌توانیم چند مورد استفاده برای هر چارچوب استخراج کنیم.

موارد استفاده از Hadoop عبارتند از:

- پردازش مجموعه داده های بزرگ در محیط هایی که اندازه داده ها از حافظه موجود بیشتر است.
- ساخت زیرساخت تجزیه و تحلیل داده ها با بودجه محدود.
- تکمیل کارهایی که نیازی به نتایج فوری نیست و زمان نیز عامل محدود کننده ای نیست.
- پردازش دسته ای با وظایفی که از عملیات خواندن و نوشتن دیسک بهره می برند.
- تجزیه و تحلیل داده های تاریخی و آرشیوی.

با Spark، ما می توانیم موارد استفاده زیر را که در آن هدوپ بهتر عمل می کند، جدا کنیم:

- تجزیه و تحلیل داده های جریان در زمان واقعی
- هنگامی که زمان اهمیت دارد، Spark نتایج سریعی را با محاسبات درون حافظه ارائه می دهد.
- مقابله با زنجیره های عملیات موازی با استفاده از الگوریتم های تکرار شونده.
- پردازش موازی نمودار برای مدل سازی داده ها.
- همه برنامه های یادگیری ماشین