# مقدمه انجام کار گروهی:

هدف ابتدایی گروه، این بود که یک پروژه در databricks ایجاد کنیم و به صورت یکپارچه بر روی امکانات apache spark کار کنیم. اما بدلیل خطای 403 فیلترشکن در پلتفرم databricks اکثریت اعضا گروه نتوانستیم به این سایت دسترسی داشته باشیم. لذا تصمیم بر این شد که بصورت محلی(local) کار را پیش ببریم و کد را در github برای دسترسی همه اعضا به اشتراک بگذاریم.

**مراحل نصب و راه اندازی spark**

ابتدا ابزارهای زیر را به ترتیب نصب کردیم:

Java.SE.Development.Kit.17.0.5.x64

Git-2.39.1-64-bit

GitHubDesktopSetup-x64

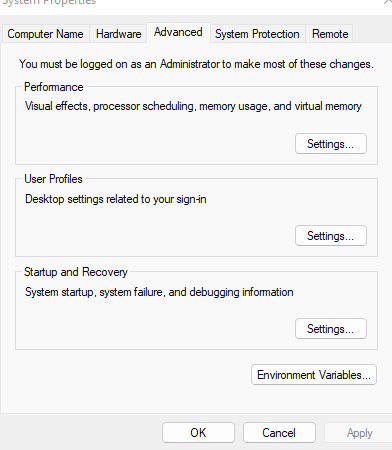
vscode

فایل زیپ زیر را فقط در مسیر درایو c کپی و استخراج می‌کنیم.

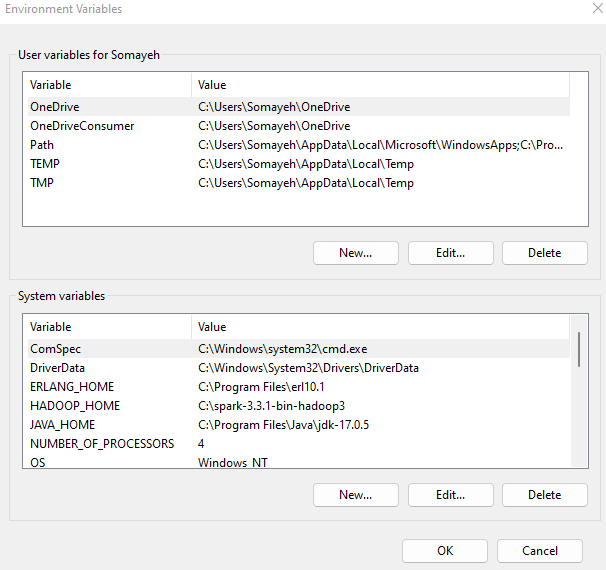
spark-3.3.1-bin-hadoop3.rar

سپس مسیرهای پیش فرض را در EnvironMent Variables اضافه میکنیم برای این کار مراحل زیر را انجام میدهیم:

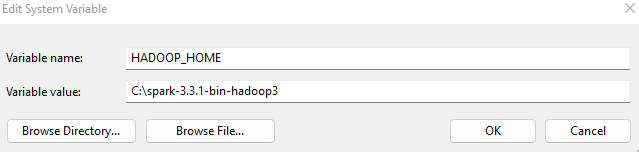
ابتدا system properties را باز میکنیم

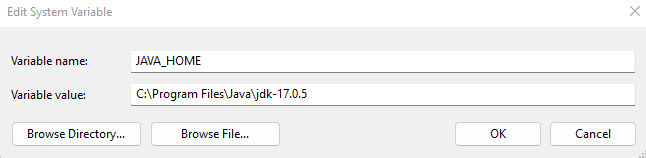


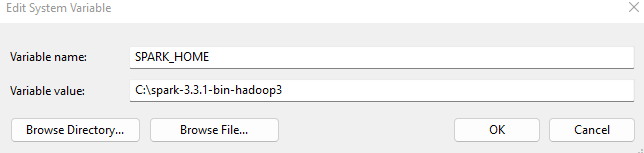
روی دکمه Environment Variables کلیک کرده در قسمت پایین دکمه new را میزنیم:



و سپس مسیر ها را به صورت زیر اضافه میکنیم:







نکته variable value ها با توجه به مسیر نصب برای هر کس ممکن است متفاوت باشد.

برای استفاده از پوسته اسپارک در Command Prompt دستور spark-shell را اجرا میکنیم و سپس دستور pip install pyspark را در همین محیط، برای نصب pyspark اجرا میکنیم.

سپس برای استفاده از کد موجود در github از آن یک شبیه سازی(Clone) در درایو c ایجاد میکنیم. با اجرای دستور cd c:/ در Command Prompt خط دستور(Command Line) را به درایو c منتقل میکنیم و دستور زیر را در آن اجرا مکنیم:

Git clone ssh address in github for this project

سپس برای استفاده از کد شبیه سازی شده در درایو c دستورات زیر را به ترتیب در Command Prompt اجرا میکنیم:

c:\>cd spark\_project

c:\spark\_project>code .

و برای استفاده از github desktop دستور زیر را اجرا میکنیم:

c:\spark\_project>github .

مقایسه پلتفرم‌ها

Databricks یک شرکت نرم افزاری آمریکایی است که توسط سازندگان آپاچی اسپارک تأسیس شده است. Databricks یک پلت فرم مبتنی بر وب برای کار با Spark ایجاد می کند که مدیریت خودکار خوشه و نوت بوک های سبک IPython را ارائه می دهد.

این شرکت که در زمینه داده های بزرگ(BigData) فعالیت می کند، عمده ابزار هایی که توسعه می دهد را به صورت عمومی منتشر کرده است.

انبارهای داده (data warehouses) برای داده های ساختاریافته ایده آل هستند، اما برای رسیدگی به داده های بدون ساختار، داده های نیمه ساختار یافته و داده هایی با تنوع، سرعت و حجم بالا، دریاچه های داده (Data Lakes) معرفی شده اند.

پلتفرم Databricks Lakehouse بهترین ویژگی های دریاچه های داده و انبارهای داده را برای ارائه قابلیت اطمینان، حاکمیت قوی، انعطاف پذیری و پشتیبانی از یادگیری ماشینی، ترکیب می کند.

اما پلتفرم Apache Zeppelin یک web based notebook مشابه databricks است که ویژگی‌هایی مانند data exploration, visualization, sharing and collaboration را فراهم می‌کند. همچنین از زبان‌های python, scala, hive, sparksql, shell, markdown پشتیبانی می‌کند. مزیت اصلی آن منبع‌باز بودن است که می‌توان آن را بر روی سرور به صورت اختصاصی مستقر کرد.

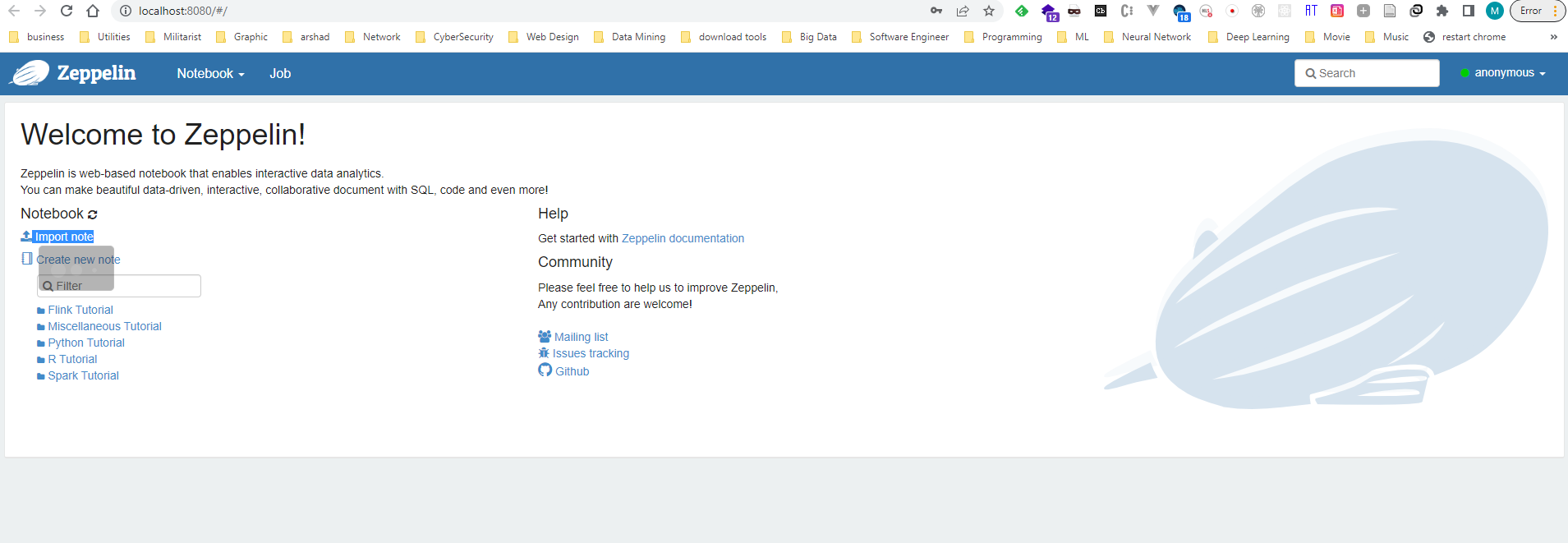
Apache Spark یک سیستم محاسباتی خوشه ای سریع و همه منظوره است که APIهای سطح بالا را که از نمودارهای اجرای عمومی پشتیبانی می کند در جاوا، اسکالا، پایتون و R و یک موتور بهینه سازی شده ارائه می دهد. Apache Spark در Zeppelin پشتیبانی می شود.

مقایسه zeppelin و Databricks:

|  |  |  |
| --- | --- | --- |
|  | zeppelin | Databricks |
| **setup** | خود میزبان(Self-hosted) | خود میزبان یا کاملا مدیریت شده |
| **JUPYTER COMPATIBILITY** | ندارد | سازگار با ژوپیتر(JUPYTER) |
| **PROGRAMMING LANGUAGES** | Python, SQL, Spark | JUPYTER |
| **DATA VISUALIZATION** | ناشناخته | مصور سازی با UI |
| **COLLABORATIVE EDITING** | مبتنی بر فایل یا نا همزمان | بلادرنگ |
| **PRICING** | رایگان | رایگان و پولی |
| **LICENSE** | منبع باز | اختصاصی |

در این پروژه ما از طریق docker، این پلتفرم را با دستور زیر بر روی سیستم محلی(local) نصب و مستقر کردیم:

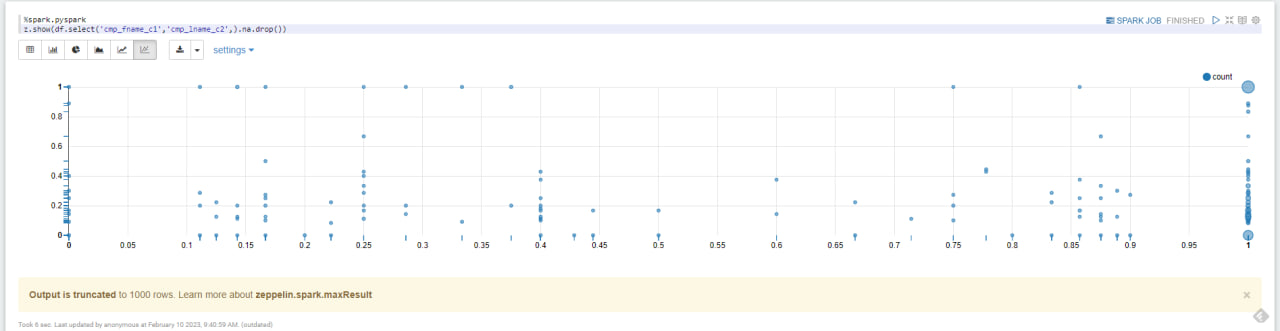
docker run -p 8080:8080 --rm --name zeppelin apache/zeppelin:0.10.1

پس از استقرار با مراجعه به آدرس localhost:8080 با صفحه اصلی zeppelin مواجه می‌شویم:

شما می‌توانید notebook ایجاد کنید و یا با گزینه import از notebook قبلا ساخته شده استفاده کنید. همچنین به دلیل اینکه از چندین زبان مختلف پشتیبانی می‌کند، در خط اول هر cell بهتر است نوع پردازش آن سلول را با % مشخص کنید.



پس از کار با zeppelin، مشاهده شد که مصورسازی با محدودیت 1000 ردیف مواجه است که با تغییر تنظیمات این مورد برطرف شد. اما با توجه به منابع محدودی که سیستم داشت، خروجی نمودارها به صورت truncated براساس بایت نمایش داده می‌شد.



به دلیل محدودیت‌های مصورسازی، به دنبال رفع مشکل 403 پلتفرم databricks اقدام کرده و به عنوان مرجعی برای مصورسازی از آن استفاده کردیم. کدهای عملیاتی تا قبل از این مرحله انجام شده و خروجی‌ها کامل بود.