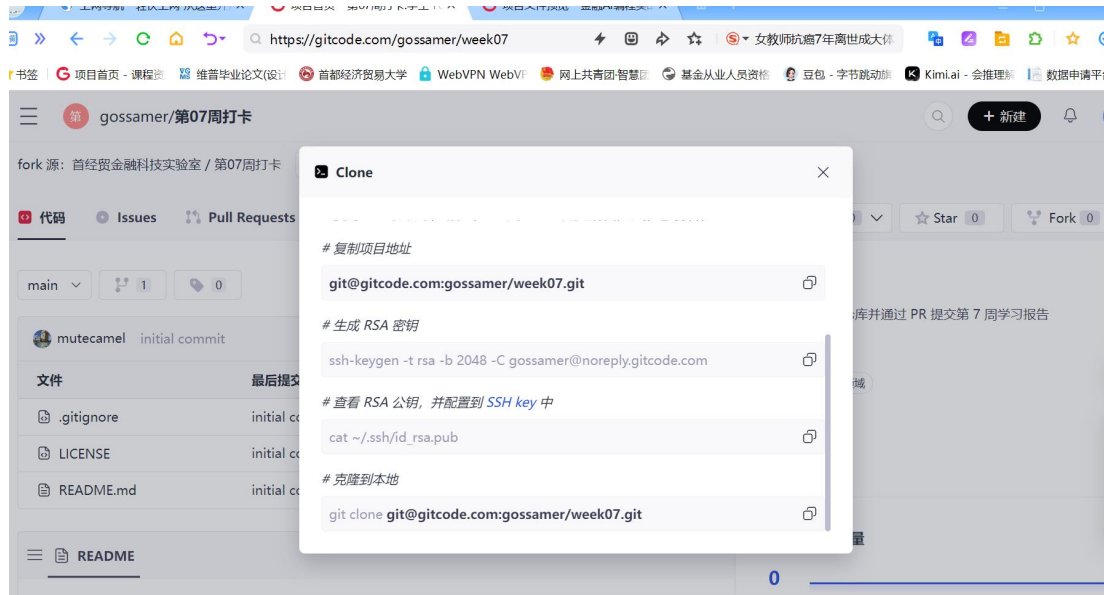
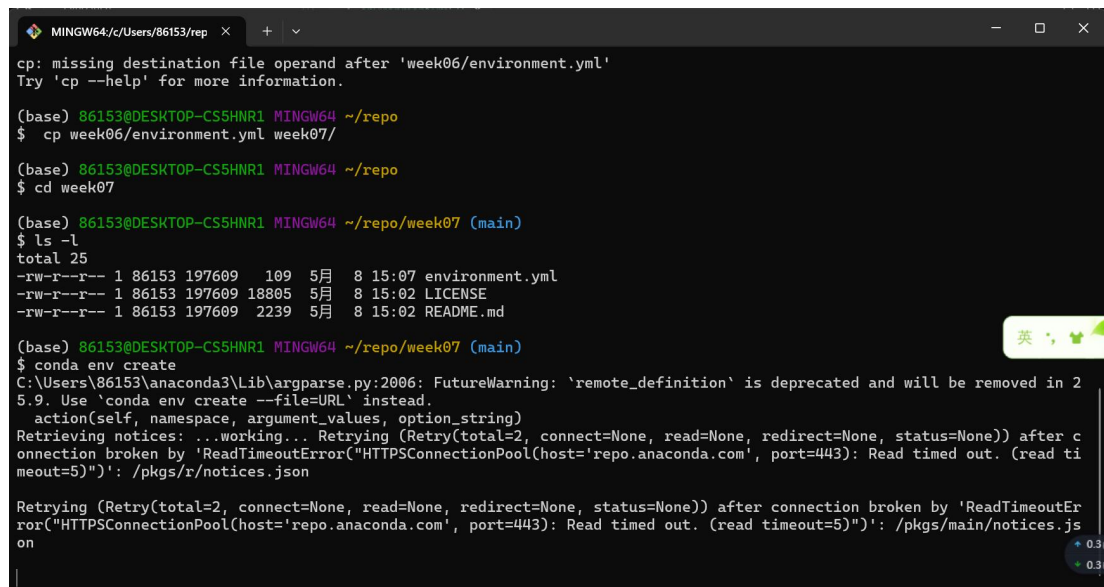


第七周学习报告

1. Fork [第 07 周打卡](https://gitcode.com/cueb-fintech/week07) 仓库至你的名下，然后将你名下的这个仓库 Clone 到你的本地计算机



2. 用 VS Code 打开项目目录，新建一个 `environment.yml` 文件，指定安装 Python 3.12 和 `jupyterlab`，然后运行 `conda env create` 命令创建 Conda 环境



```
MINGW64/c/Users/86153/rep x + v
total 25
-rw-r--r-- 1 86153 197609 109 5月 8 15:07 environment.yml
-rw-r--r-- 1 86153 197609 18805 5月 8 15:02 LICENSE
-rw-r--r-- 1 86153 197609 2239 5月 8 15:02 README.md

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$ conda env create
C:\Users\86153\anaconda3\Lib\argparse.py:2006: FutureWarning: 'remote_definition' is deprecated and will be removed in 2
5.9. Use 'conda env create --file=URL' instead.
  action(self, namespace, argument_values, option_string)
Channels:
- conda-forge
- defaults
- https://repo.anaconda.com/pkg/main
- https://repo.anaconda.com/pkg/r
- https://repo.anaconda.com/pkg/msys2
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

done
#
# To activate this environment, use
#
#   $ conda activate week07
#
# To deactivate an active environment, use
#
#   $ conda deactivate
```

3. 在项目目录下，运行 `jupyter lab` 命令，启动 ****后端**** (Backend) 服务，在浏览器里粘贴地址访问 ****前端**** (Frontend) 页面

```
MINGW64/c/Users/86153/rep x + v
(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$ conda activate week07
(week07)
86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$ ipython
Python 3.12.10 | packaged by conda-forge | (main, Apr 10 2025, 22:08:16) [MSC v.1943 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 9.2.0 -- An enhanced Interactive Python. Type '?' for help.
Tip: We can't show you all tips on Windows as sometimes Unicode characters crash the Windows console, please help us deb
ug it.

In [1]: quit
(week07)
86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$ |
```

```
MINGW64: c:/Users/86153/rep x + | v
[I 2025-05-08 15:20:25.749 ServerApp] panel.io.jupyter_server_extension | extension was successfully linked.
[I 2025-05-08 15:20:25.804 ServerApp] notebook_shim | extension was successfully loaded.
[I 2025-05-08 15:20:25.887 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2025-05-08 15:20:25.888 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2025-05-08 15:20:26.816 LabApp] JupyterLab extension loaded from C:\Users\86153\anaconda3\Lib\site-packages\jupyterlab
[I 2025-05-08 15:20:26.816 LabApp] JupyterLab application directory is C:\Users\86153\anaconda3\share\jupyterlab
[I 2025-05-08 15:20:26.817 LabApp] Extension Manager is 'pypi'.
[I 2025-05-08 15:20:26.162 ServerApp] jupyterlab | extension was successfully loaded.
[I 2025-05-08 15:20:26.170 ServerApp] notebook | extension was successfully loaded.
[I 2025-05-08 15:20:26.171 ServerApp] panel.io.jupyter_server_extension | extension was successfully loaded.
[I 2025-05-08 15:20:26.172 ServerApp] Serving notebooks from local directory: C:\Users\86153\repo\week07
[I 2025-05-08 15:20:26.172 ServerApp] Jupyter Server 2.14.1 is running at:
[I 2025-05-08 15:20:26.172 ServerApp] http://localhost:8888/tree?token=4c20ca8c138c9f9b7fd1dc5b21659191ba604bafc15a4a32
[I 2025-05-08 15:20:26.173 ServerApp] http://127.0.0.1:8888/tree?token=4c20ca8c138c9f9b7fd1dc5b21659191ba604bafc15a4a32
[I 2025-05-08 15:20:26.173 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip c
ation).
[C 2025-05-08 15:20:26.228 ServerApp]

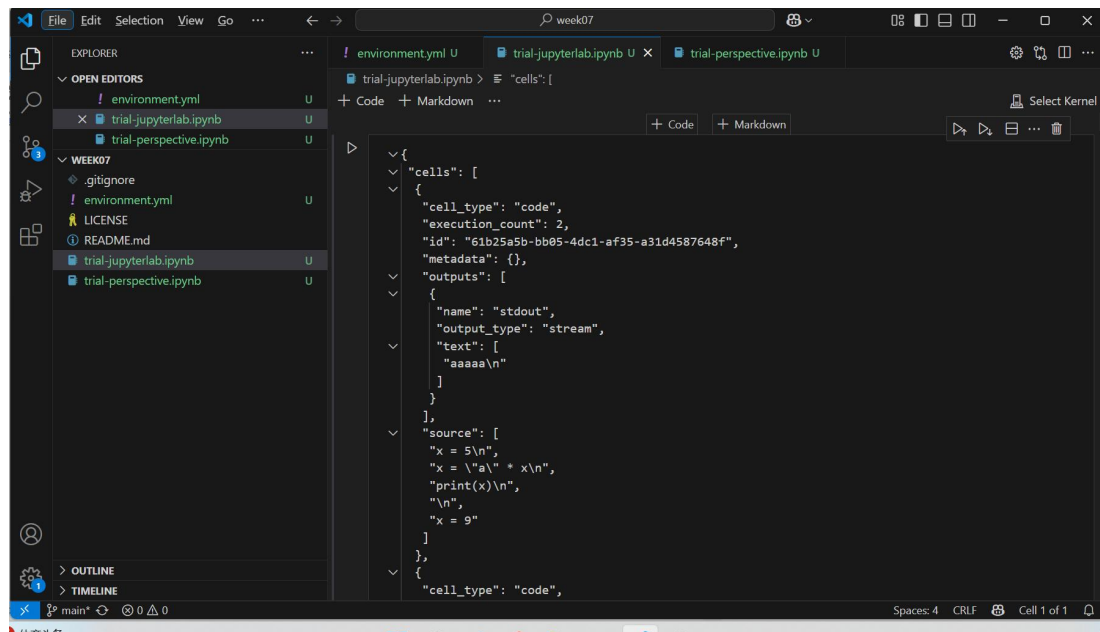
To access the server, open this file in a browser:
file:///C:/Users/86153/AppData/Roaming/jupyter/runtime/jpserver-8320-open.html
Or copy and paste one of these URLs:
http://localhost:8888/tree?token=4c20ca8c138c9f9b7fd1dc5b21659191ba604bafc15a4a32
http://127.0.0.1:8888/tree?token=4c20ca8c138c9f9b7fd1dc5b21659191ba604bafc15a4a32
[I 2025-05-08 15:20:26.383 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-server-
nodejs, javascript-typescript-langserver, jedi-language-server, julia-language-server, pyright, python-language-server,
r-languageserver, sql-language-server, texlab, typescript-language-server, unified-language-server, vscode-css-languages
erver-bin, vscode-html-languageserver-bin, vscode-json-languageserver-bin, yaml-language-server
```

4. 在 JupyterLab 页面里，新建一个 Notebook，改名为 `trial-jupyterlab.ipynb`，在里面实践掌握以下功能：

- 在单元格 (Cell) 里编写 Python 代码，按 `Shift+Enter` 运行 Cell 并下移
- 在单元格 (Cell) 上按 `ESC` 切换到 ****命令模式**** (command mode)，按 `Enter` 切换到 ****编写模式**** (edit mode)
- 在单元格 (Cell) 的命令模式下，按 `j` 选择下一个，按 `k` 选择上一个，按 `a` 在上方添加，按 `b` 在下方添加，按 `dd` 删除，按住 `Shift` 多选，按 `x` 剪切，按 `c` 复制，按 `v` 粘贴，按 `Shift+M` 合并，按 `z` 撤销，按 `Shift+Z` 重做，按 `Shift+L` 显示/隐藏代码行号
- 在单元格 (Cell) 的编写模式下，按 `Ctrl+Shift+-` 切分单元格
- 按按钮显示/隐藏 Minimap
- 运行单元格 (Cell) 注意序号单调递增
- 单元格最后一行如果是 ****表达式**** (expression) 且运行后返回的对象不是 `None`，则计输出 (Out)，否则只计输入 (In)，序号为 `i` 的输出，可以用 `i` 变量来引用
- 单元格 (Cell) 序号为 `*` 表示代码运行中，尚未返回，按 `ii` 可以打断 (KeyboardInterrupt) (类似于终端的 `Ctrl+C`)
- 在单元格 (Cell) 的命令模式下，按 `00` 重启后端 Python 解释器 (被 Jupyter 称为 Kernel)，重启后需要从上至下重新运行一遍代码 (`Shift+Enter`)，运行前建议先在菜单里选择 “Edit / Clear Outputs of All Cells” 清空全部页面显示的输出
- 在单元格 (Cell) 的命令模式下，按 `m` 切换至 ****Markdown 模式****，按 `y` 切换至 ****Python 模式****
- 用豆包 (或 DeepSeek 等任何大模型) 生成一段示例 Markdown 代码，复制粘贴进 Markdown 单元格，运行以呈现 (Render)
- 用豆包 (或 DeepSeek 等任何大模型) 生成一段示例 HTML 代码，复制粘贴进 Markdown 单元格，运行以呈现 (Render)；注意不支持 CSS
- 用豆包 (或 DeepSeek 等任何大模型) 生成一段示例 LaTeX 数学公式代

码，复制粘贴进 Markdown 单元格，运行以呈现 (Render)；注意要用 `` (行内模式) 或 ``` (整行模式) 包围

- 关闭前端页面，在后端按 `Ctrl+C` 打断运行中的服务，回到 Bash 提示符



5. 通过 `tushare` 软件包下载保存一些数据：

- 在 Tushare 网站上 [注册](https://tushare.pro/register?reg=gl5) 并登陆，完善修改个人资料，浏览阅读 [平台介绍](https://tushare.pro/document/1) 和 [数据接口](https://tushare.pro/document/2)

> 通过定制的推荐链接 `https://tushare.pro/register?reg=gl5` 完成注册，将可获赠 2000 平台积分 (有效期一年)。积分达到门槛才有数据接口的使用权限，否则需要 [付费购买积分](https://tushare.pro/document/1?doc_id=290) (约 200~1000 元/年) 才有权使用数据接口。本课程的量化投资实战案例，将主要通过 Tushare 平台获取数据，请确保拥有足够积分进行实践。

- 修改 `environment.yml` 文件，添加 `pip: tushare` (注意，[`conda-forge`](https://conda-forge.org/packages/) 没有收录 `tushare`，只能从 [PyPI](https://pypi.org/project/tushare/) 安装，[参考](https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#creating-an-environment-file-manually)) 依赖项，运行 `conda env update` 更新 Conda 环境

- 在终端 (Terminal) 激活 `week07` Conda 环境，运行 `ipython` 命令启动 IPython 交互界面 ([IPython](https://ipython.readthedocs.io/en/stable/) 是 Jupyter 项目的一部份，`ipython` 是 `jupyterlab` 的依赖项之一)

- 在 IPython 提示符下，运行下面的 Python 代码设置 Tushare Token

```

```python
import tushare as ts

ts.set_token("****") # 将 **** 修改成你自己的 Token 字符串
```

```

其中 `****` 要替换成你在 Tushare 平台上的 [接口TOKEN](https://tushare.pro/user/token) —— 复制粘贴即可。运行 `set_token` 函数会把 Token 字符串保存在 `~/tk.csv` 文件里, 今后每次使用 `tushare` 软件包请求数据时都会自动读取并发送 Token, 不需要反复设置。

- 按 `Ctrl+D` 结束前面的 IPython 进程, 再重新启动一个新的 IPython 进程, 运行下面的 Python 代码向 Tushare 服务器请求 [IPO 新股列表](https://tushare.pro/document/2?doc_id=123) 数据, 并保存在本地

```

```python
import tushare as ts

pro = ts.pro_api()
df = pro.new_share()
df.to_parquet("new_share.parquet")
```

```

其中请求数据函数返回的对象 `df` 是 `pandas.DataFrame` 类型, 调用其 [to_parquet](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_parquet.html) 方法能够将内存 (memory) 中的 `DataFrame` 数据按照 [Parquet](https://parquet.apache.org/) 格式 (Parquet 是大数据领域的首选格式, 已经成为业界标准) 序列化 (serialize) 为字节串 (bytes) 保存到磁盘 (disk)。

- 询问豆包 (或 DeepSeek 等任何大模型), 初步了解 Parquet 格式和 CSV 格式的特点和适用领域

- [new_share](https://tushare.pro/document/2?doc_id=123) 接口只需要 120 积分, 如果你有 2000 积分, 可以采用与上面类似的方法访问 [stock_basic](https://tushare.pro/document/2?doc_id=25) 接口, 并将数据保存为 `stock_basic.parquet` 文件 (注意, 需要指定 `fields` 参数获取全部字段)。如果积分暂时不够, 可以在终端运行下面的命令, 从我们开源的 [课程仓库](https://gitcode.com/cueb-fintech/courses/blob/main/data/stock_basic.parquet) 下载数据文件到你的本地

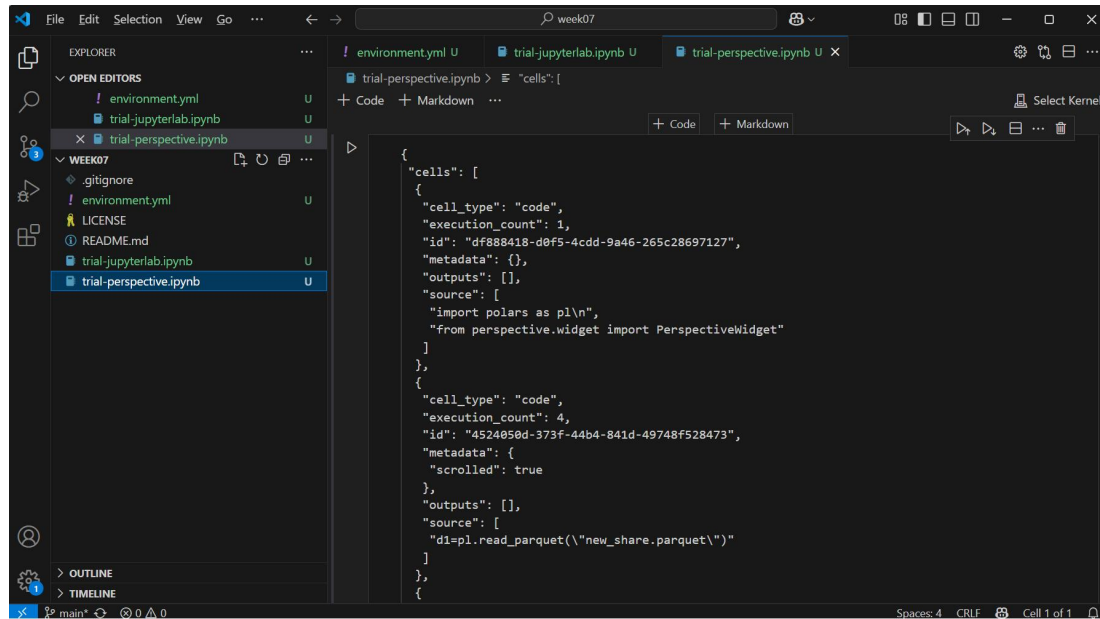
```

```bash
curl -O
https://raw.gitcode.com/cueb-fintech/courses/blobs/8fc08f7bc4dbbf17d356234472
```

```


795e59c7b9ce2f/stock_basic.parquet

...



6. 通过 `perspective-python` 软件包查看 `polars.DataFrame` 数据，实践交互式可视化：

- 修改 `environment.yml` 文件，添加 `perspective-python` 和 `polars` 依赖项，运行 `conda env update` 更新 Conda 环境
- 启动 JupyterLab，新建一个 Notebook，改名为 `trial-perspective.ipynb`
- 调用 `polars.read_parquet` 函数，分别读取磁盘 (disk) 中的 `new_share.parquet` 文件和 `stock_basic.parquet` 文件，得到内存 (memory) 中的 `polars.DataFrame` 对象，命名为 `d1` 和 `d2`
- 进行适当的列变换，尤其是要把实际上是日期类型的列，从 `polars.String()` 类型转换为 `polars.Date()` 类型
- 把 `d1` 或 `d2` 作为参数传递给 `perspective.widget.PerspectiveWidget` 类型进行初始化，返回的对象会呈现在 Notebook 的 Output 里
- 在 `PerspectiveWidget` 默认的 `Datagrid` 视图下，尝试实践：
 - 修改各种列数据类型 (文本、数值、日期) 的显示风格 (style)
 - 设置 `Group By` 选项，选择某些列作为分组依据 (纵向排列)，选择其他某些列进行汇总 (注意汇总方式有多种函数选项)
 - 设置 `Split By` 选项，选择某些列作为拆分依据 (横向排列)
 - 设置 `Order By` 选项，选择某些列作为排序依据 (注意可以切换 升序/降序)
 - 设置 `Where` 选项，选择某些列，进一步设置条件，进行数据行 (观测) 方向的过滤
 - 设置 `Columns` 选项，选择要显示的数据列 (变量)，及其显示的先后顺序
 - 在 `All Columns` 部分，是能够显示但没有显示的数据列 (变量)，可以点击 `NEW COLUMN` 添加衍生计算出的新列，需要用

[ExprTK](<https://www.partow.net/programming/exprtk/>) 语言书写表达式代码, 变量名用双引号 (") 包围, 字符串用单引号 (') 包围

- 在 `PerspectiveWidget` 图形界面依靠鼠标 (手动) 所做的设置 (configure), 可以导出代码, 根据导出的代码, 可以修改我们的代码, 使得我们运行代码直接就能得到我们所需要的视图 (自动化)

- 在 `PerspectiveWidget` 的右上方有按钮, 可以把图形界面的数据或设置 (configure) 导出 (export) 为文件, 或复制 (copy) 到剪贴板

- 把设置 (`config.json`) 复制到剪贴板, 粘贴进 Notebook Cell, 保存成字符串 (`str`)

- 也可以把设置 (`config.json`) 导出为文件, 用 `pathlib.Path.read_text` 方法从文件读取字符串 (`str`)

- 可以用 `json.loads` 函数将无结构的 (unstructured) 字符串 (`str`) 解析为有结构的 (structured) Python 字典 (`dict`), 这样就容易在 Notebook 里美化呈现, 也容易进一步通过 Python 代码访问内部的具体设置

- 也可以把复制到剪贴板的 JSON 字符串, 粘贴进某个在线的 JSON 工具网站 (比如 [链接](<https://jsonformatter.org/>)) 进行美化

- 根据导出的设置代码, 在初始化 (init) `PerspectiveWidget` 类型时, 传入适当的参数进行设置, 运行代码, 观察是否符合我们的期望

- 把 `PerspectiveWidget` 切换为 `Treemap` 视图, 尝试设置各种选项 (configure), 观察数据可视化的实际效果

- `Treemap` (树形结构图) 用不同大小的矩形来体现数据的分类占比构成情况, 还可以用矩形的颜色来体现第二个维度的数据 (文本或数值都可以)

- 点击 [链接 1](<https://datavizcatalogue.com/methods/treemap.html>) 或 [链接 2](<https://www.data-to-viz.com/graph/treemap.html>) 学习了解更多 `Treemap` 的概念、适用情形以及实现代码

- 把 `PerspectiveWidget` 切换为 `Y Bar` 视图, 尝试设置各种选项 (configure), 观察数据可视化的实际效果

- `Y Bar` (条形图/柱状图) 的横轴 (不同的条形) 是第一个维度, 用 `Group By` 控制, 纵轴 (条形的高度) 是第二个维度, 用 `Y Axis` 控制 (支持多变量并列显示), 还可以把每个条形进一步拆分为多个颜色, 用 `Split By` 控制

- 点击 [链接 1](https://datavizcatalogue.com/methods/bar_chart.html) 或 [链接 2](<https://www.data-to-viz.com/graph/barplot.html>) 学习了解更多 `Bar Chart` 的概念、适用情形以及实现代码

- `Y Bar` 视图还可以用来实现一类很重要的统计制图 —— **直方图** (histogram)。对于数据表中的某一列连续型数值变量 (比如新股发行的市盈率 `pe`), 我们经常希望观察其 **分布** (distribution)。可以用 `bucket` 函数对连续变量进行 “分桶” (比如表达式 `bucket("pe", 10)`), 生成一个新的离散变量 (比如命名为 `bucket_pe`), 然后把离散变量设置为 `Y Bar` 的横轴 (`Group By`),

把任意其他一列变量用 `count` (计数) 函数汇总, 设置为纵轴 (`Y Axis`)。这样看到的的就是直方图。“分桶” 在有的地方也叫 “分箱” (bin), 其粒度大小需要根据数据适当调节。

- 把 `PerspectiveWidget` 切换为 `Y Line` 视图, 尝试设置各种选项 (configure), 观察数据可视化的实际效果

- `Y Line` (折线图) 常用来绘制时间序列, 横轴通常是时间, 用 `Group By` 控制, 纵轴 (折线的 Y 坐标) 通常是连续型数值变量 (经过汇总), 用 `Y Axis` 控制 (支持多序列同时显示), 还可以进一步拆分为多条序列, 用 `Split By` 控制

- 点击 [链接 1](https://datavizcatalogue.com/methods/line_graph.html) 或 [链接 2](<https://www.data-to-viz.com/graph/line.html>) 学习了解更多 `Line Chart` 的概念、适用情形以及实现代码

- 使用我们的示例数据, 可以尝试观察最近几年 A 股 IPO 市场的 **融资额** (`funds`) 与 **市盈率** (`pe`) 变化情况。为了加深对数据的 *理解* 和 *验证*, 可以询问豆包 (或 DeepSeek 等任何大模型), 在某个时间段内发生了哪些影响 A 股 (或 IPO) 的重大国内外财经事件, 由此加强我们对现实背景的理解

- 也可以使用示例数据, 观察对比最近几年不同交易所 (`exchange`) 或市场 (`market`) 的平均 **中签率** (`ballot`) 情况

- 把 `PerspectiveWidget` 切换为 `X/Y Scatter` 视图, 尝试设置各种选项 (configure), 观察数据可视化的实际效果

- `X/Y Scatter` (散点图) 常用来观察两个数值型连续变量之间的相关关系 (correlation)。数据首先可以进行分组汇总, 每一个组对应一个散点, 用 `Group By` 控制。然后把两个连续型数值变量分别设置为 `X Axis` 和 `Y Axis`, 其汇总数值将作为每个散点的坐标

- 点击 [链接 1](<https://datavizcatalogue.com/methods/scatterplot.html>) 或 [链接 2](<https://www.data-to-viz.com/graph/scatter.html>) 学习了解更多 `Scatter Plot` 的概念、适用情形以及实现代码

- 散点的分布如果特别不均匀, 则意味着变量单位可能有问题, 或者需要经过变换 (比如取对数)

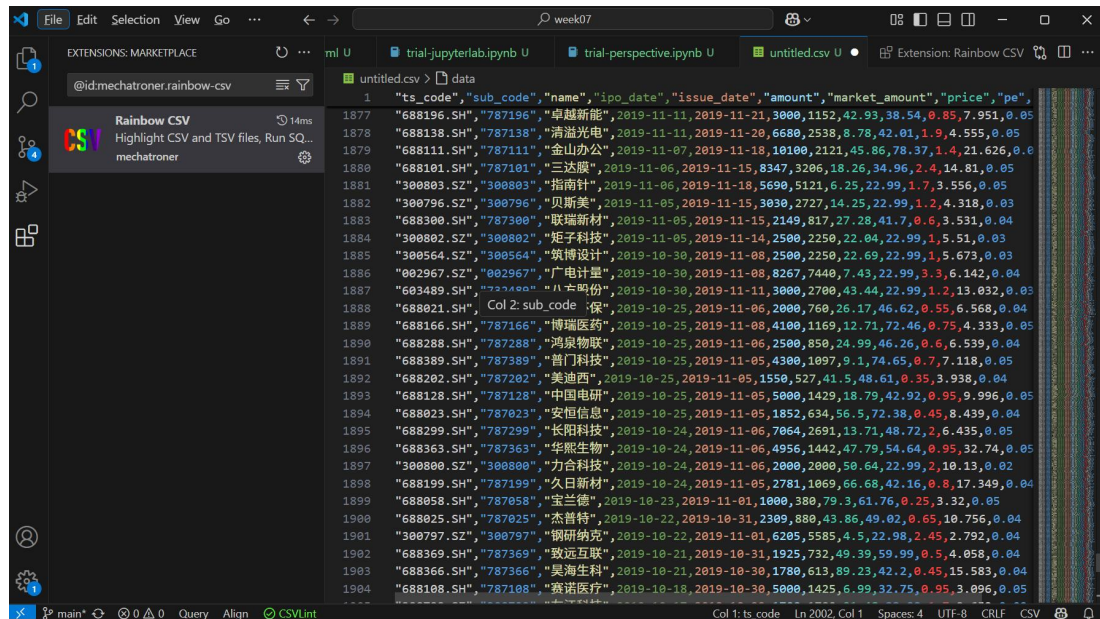
- 散点的分布如果杂乱无规律, 则意味着 X 与 Y 没有相关性

- 散点的分布如果看起来能够拟合成一条直线 (即回归线, regression), 则意味着 X 与 Y 具有正的或负的相关性, 意味着可能存在某些规律

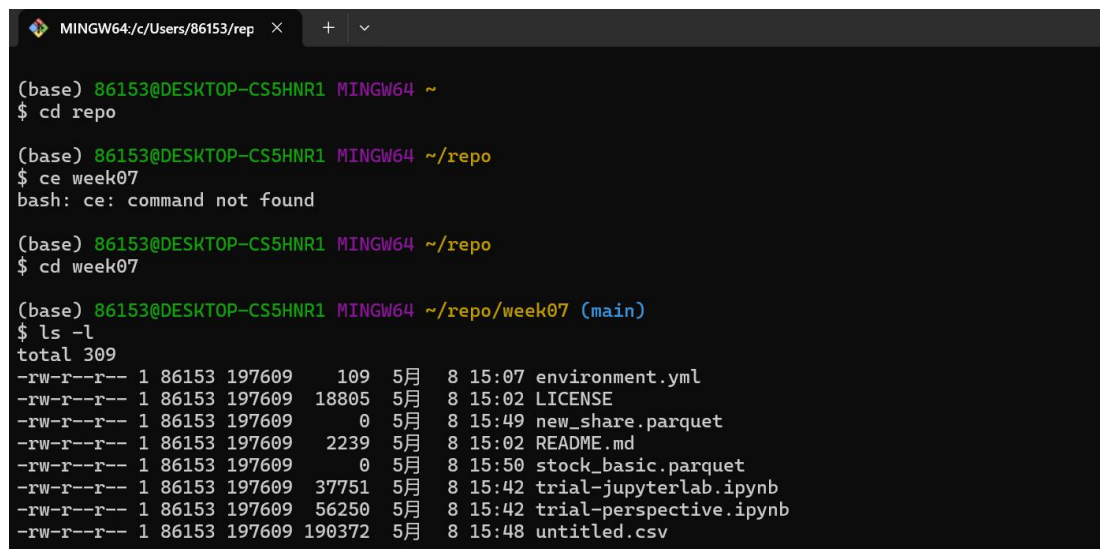
- 散点图上可以进一步体现更多的变量维度, 比如可以把更多变量映射为散点的不同颜色 (`Color`)、大小 (`Size`)、符号 (`Symbol`)、标签 (`Label`)、提示框 (`Tooltip`) 等

- 我们经常还可以把用于分类的类别变量 (类别不宜太多) 设置为 `Split By`, 从而把一个散点图拆分为多个小散点图 ([small multiple](https://en.wikipedia.org/wiki/Small_multiple))), 从而更细致地观察是否存在规律

- 在我们的示例数据中，**融资额**(`funds`)、**市盈率**(`pe`)、**中签率**(`ballot`) 是数值型连续变量，适合用散点图观察他们的规律，散点可以以个股为单位 (不汇总)，也可以按 **行业**(`industry`) 汇总，或者按 **上市时间**(`ipo_date`) 汇总 (每月分桶)，都可以大胆尝试探索



```
1 "ts_code","sub_code","name","ipo_date","issue_date","amount","market_amount","price","pe",
1877 "688196.SH","787196","卓越新能",2019-11-11,2019-11-21,3000,1152,42.93,38.54,0.85,7.951,0.05
1878 "688138.SH","787138","清溢光电",2019-11-11,2019-11-20,6680,2538,8.78,42.01,1.9,4.555,0.05
1879 "688111.SH","787111","金山办公",2019-11-07,2019-11-18,10100,2121,45.86,78.37,1.4,21.626,0.0
1880 "688101.SH","787101","三达膜",2019-11-06,2019-11-15,8347,3206,18.26,34.96,2.4,14.81,0.05
1881 "300803.SZ","300803","指南针",2019-11-06,2019-11-18,5690,5121,6.25,22.99,1.7,3.556,0.05
1882 "300796.SZ","300796","贝斯美",2019-11-05,2019-11-15,3030,2727,14.25,22.99,1.2,4.318,0.03
1883 "688300.SH","787300","联瑞新材",2019-11-05,2019-11-15,2149,817,27.28,41.7,0.6,3.531,0.04
1884 "300802.SZ","300802","矩子科技",2019-11-05,2019-11-14,2500,2250,22.04,22.99,1.5,5.51,0.03
1885 "300564.SZ","300564","筑博设计",2019-10-30,2019-11-08,2500,2250,22.69,22.99,1.5,6.673,0.03
1886 "002967.SZ","002967","广电计量",2019-10-30,2019-11-08,8267,7440,7.43,22.99,3.3,6.142,0.04
1887 "603489.SH","787489","八大股份",2019-10-30,2019-11-11,3000,2700,43.44,22.99,1.2,13.032,0.03
1888 "688021.SH","787021","保",2019-10-25,2019-11-06,2000,760,26.17,46.62,0.55,6.568,0.04
1889 "688166.SH","787166","博瑞医药",2019-10-25,2019-11-08,4100,1169,12.71,72.46,0.75,4.333,0.05
1890 "688288.SH","787288","鸿泉物联",2019-10-25,2019-11-06,2500,850,24.99,46.26,0.6,6.539,0.04
1891 "688389.SH","787389","普门科技",2019-10-25,2019-11-05,4300,1097,9.1,74.65,0.7,7.118,0.05
1892 "688202.SH","787202","美迪西",2019-10-25,2019-11-05,1550,527,41.5,48.61,0.35,3.938,0.04
1893 "688128.SH","787128","中国电研",2019-10-25,2019-11-05,5000,1429,18.79,42.92,0.95,9.996,0.05
1894 "688023.SH","787023","安恒信息",2019-10-25,2019-11-05,1852,634,56.5,72.38,0.45,8.439,0.04
1895 "688299.SH","787299","长阳科技",2019-10-24,2019-11-06,7064,2691,13.71,48.72,2.6,4.435,0.05
1896 "688363.SH","787363","华熙生物",2019-10-24,2019-11-06,4956,1442,47.79,54.64,0.95,32.74,0.05
1897 "300800.SZ","300800","力合科技",2019-10-24,2019-11-06,2000,2000,50.64,22.99,2.10,13.0,0.02
1898 "688199.SH","787199","久日新材",2019-10-24,2019-11-05,2781,1069,66.68,42.16,0.8,17.349,0.04
1899 "688058.SH","787058","宝兰德",2019-10-23,2019-11-01,1000,380,79.3,61.76,0.25,3.32,0.05
1900 "688025.SH","787025","杰普特",2019-10-22,2019-10-31,2309,880,43.86,49.02,0.65,10.756,0.04
1901 "300797.SZ","300797","钢研纳克",2019-10-22,2019-11-01,6205,5585,4.5,22.98,2.45,2.792,0.04
1902 "688369.SH","787369","致远互联",2019-10-21,2019-10-31,1925,732,49.39,59.99,0.5,4.058,0.04
1903 "688366.SH","787366","昊海生科",2019-10-21,2019-10-30,1700,613,89.23,42.2,0.45,15.583,0.04
1904 "688108.SH","787108","赛诺医疗",2019-10-18,2019-10-30,5000,1425,6.99,32.75,0.95,3.096,0.05
```



```
(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~
$ cd repo

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo
$ ce week07
bash: ce: command not found

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo
$ cd week07

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week07 (main)
$ ls -l
total 309
-rw-r--r-- 1 86153 197609 109 5月 8 15:07 environment.yml
-rw-r--r-- 1 86153 197609 18805 5月 8 15:02 LICENSE
-rw-r--r-- 1 86153 197609 0 5月 8 15:49 new_share.parquet
-rw-r--r-- 1 86153 197609 2239 5月 8 15:02 README.md
-rw-r--r-- 1 86153 197609 0 5月 8 15:50 stock_basic.parquet
-rw-r--r-- 1 86153 197609 37751 5月 8 15:42 trial-jupyterlab.ipynb
-rw-r--r-- 1 86153 197609 56250 5月 8 15:42 trial-perspective.ipynb
-rw-r--r-- 1 86153 197609 190372 5月 8 15:48 untitled.csv
```