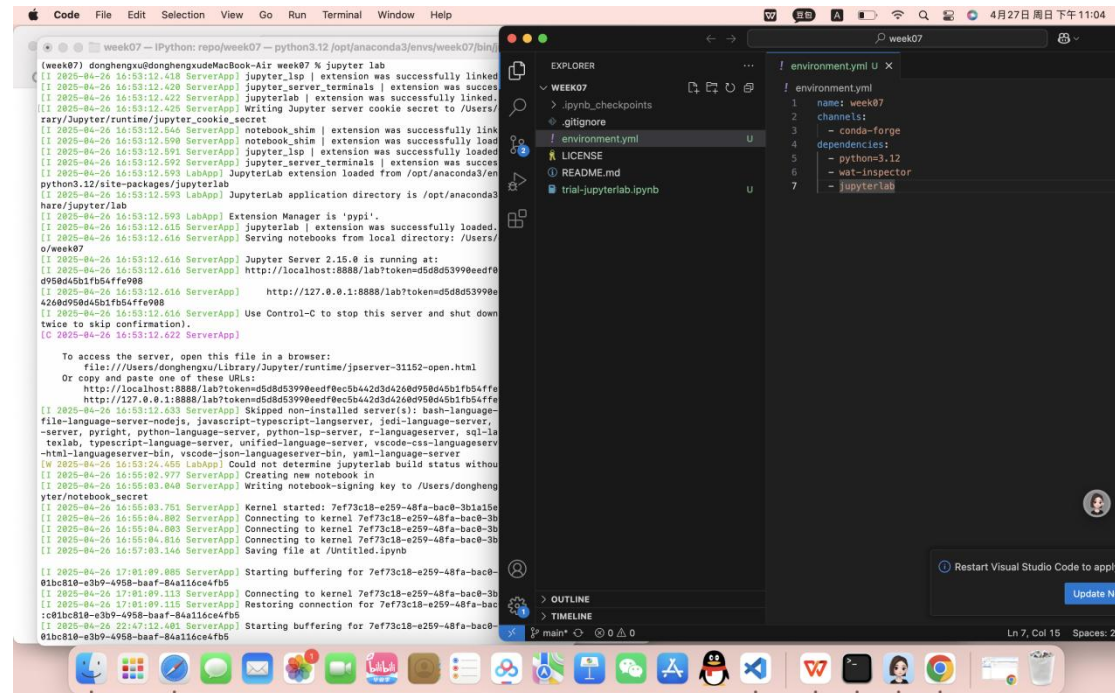


第七周报告

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

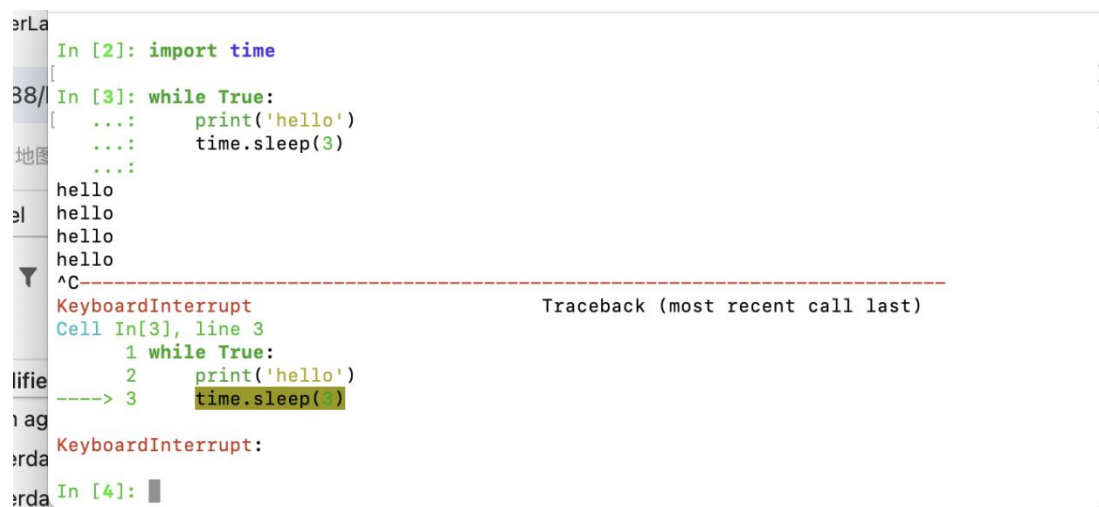
在项目目录下，运行 `jupyter lab` 命令，启动 后端 (Backend) 服务，在浏览器里粘贴地址访问 前端 (Frontend) 页面



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



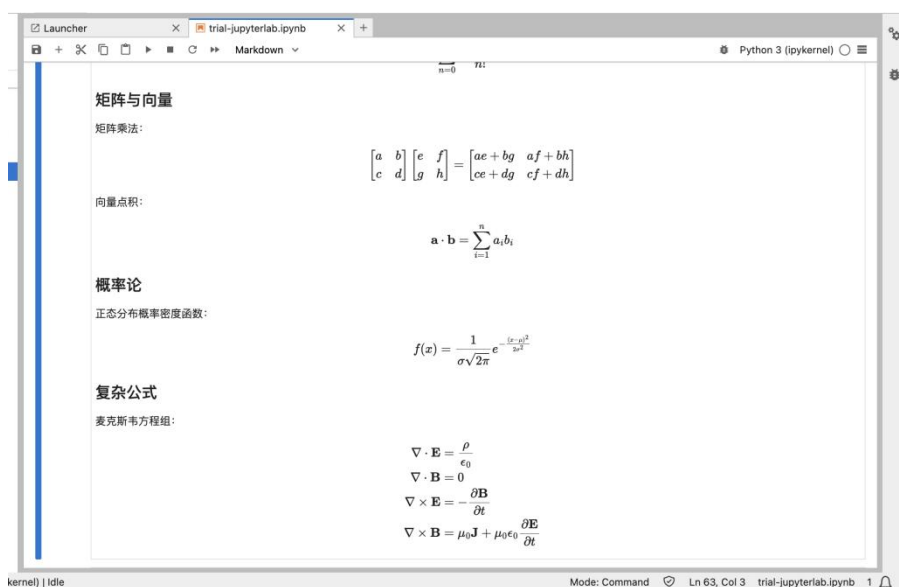
用豆包 (或 DeepSeek 等任何大模型) 生成一段示例 Markdown 代码，复制粘贴进 Markdown 单元格，运行以呈现 (Render)



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

```
week07 -- zsh -- 94x32
b.
[I 2025-05-07 09:35:39.118 ServerApp] Restoring connection for ff6735d4-0cc6-4253-bbb1-20d952e
b8fdb:1a558483-98c8-4878-8fa7-f163cd0e2440
[I 2025-05-09 14:59:17.242 ServerApp] Starting buffering for ff6735d4-0cc6-4253-bbb1-20d952eb8
fdb:1a558483-98c8-4878-8fa7-f163cd0e2440
[I 2025-05-09 14:59:17.329 ServerApp] Connecting to kernel ff6735d4-0cc6-4253-bbb1-20d952eb8fd
b.
[I 2025-05-09 14:59:17.330 ServerApp] Restoring connection for ff6735d4-0cc6-4253-bbb1-20d952e
b8fdb:1a558483-98c8-4878-8fa7-f163cd0e2440
[I 2025-05-09 16:02:50.558 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:04:50.618 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:06:50.653 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:08:50.688 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:12:50.722 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:14:50.810 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:16:50.834 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:18:50.865 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:21:05.773 ServerApp] Saving file at /trial-jupyterlab.ipynb
[I 2025-05-09 16:21:05.773 ServerApp] Starting buffering for ff6735d4-0cc6-4253-bbb1-20d952eb8
fdb:1a558483-98c8-4878-8fa7-f163cd0e2440
^C[I 2025-05-09 16:21:47.267 ServerApp] interrupted
[I 2025-05-09 16:21:47.268 ServerApp] Serving notebooks from local directory: /Users/donghengxu
u/repo/week07
1 active kernel
Jupyter Server 2.15.0 is running at:
http://localhost:8888/lab?token=0c25e21969f69dd68b5e28be84e5ac83fa859d3d64eca7c6
http://127.0.0.1:8888/lab?token=0c25e21969f69dd68b5e28be84e5ac83fa859d3d64eca7c6
Shut down this Jupyter server (y/[n])? y
[C 2025-05-09 16:21:49.696 ServerApp] Shutdown confirmed
[I 2025-05-09 16:21:49.701 ServerApp] Shutting down 4 extensions
[I 2025-05-09 16:21:49.702 ServerApp] Shutting down 1 kernel
[I 2025-05-09 16:21:49.703 ServerApp] Kernel shutdown: ff6735d4-0cc6-4253-bbb1-20d952eb8fdb
(week07) donghengxu@donghengxudeMacBook-Air week07 %
```

修改 `environment.yml` 文件，添加 `pip: tushare` (注意，`conda-forge` 没有收录 `tushare`，只能从 `PyPI` 安装，参考) 依赖项，运行 `conda env update` 更新 `Conda` 环境

```
! environment.yml
1 name: week07
2 channels:
3   - conda-forge
4 dependencies:
5   - python=3.12
6   - wat-inspector
7   - jupyterlab
8   - pip
9   - pip:
10    - tushare
```

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。

```
Last login: Sun Apr 27 23:20:34 on ttys007
(base) donghengxu@donghengxudeMacBook-Air ~ % ls
Applications          Pictures
Applications (Parallels)  Public
Desktop               Visual Studio Code.app
Documents             Visual Studio Code的副本.app
Downloads             abc.txt
Library              brew-install
Movies               repo
Music                report.txt
Parallels            tk.csv
(base) donghengxu@donghengxudeMacBook-Air ~ % cat ts.csv
cat: ts.csv: No such file or directory
(base) donghengxu@donghengxudeMacBook-Air ~ % cat tk.csv
token
my fake token
(base) donghengxu@donghengxudeMacBook-Air ~ %
```

```

[In [1]: import tushare as ts
[In [2]: pro = ts.pro_api()
[In [3]: df = pro.new_share()
[In [4]: df
Out[4]:
   ts_code sub_code name ipo_date ... pe limit_amount funds ballot
0  001390.SZ  001390  麒麟新材  20250519 ... 0.00 2.00 0.000 0.00
1  603014.SH  732014  威高血净  20250508 ... 24.82 1.10 10.902 0.03
2  301595.SZ  301595  太力科技  20250508 ... 21.55 0.65 4.615 0.02
3  688755.SH  787755  汉邦科技  20250507 ... 26.35 0.50 5.009 0.03
4  301636.SZ  301636  泽润新能  20250428 ... 17.57 0.45 5.279 0.02
...
1995 300776.SZ  300776  帝尔激光  20190507 ... 22.99 1.60 9.543 0.01
1996 300777.SZ  300777  中简科技  20190506 ... 22.98 1.10 2.425 0.04
1997 603267.SH  732267  鸿远电子  20190430 ... 16.50 1.60 8.367 0.03
1998 600989.SH  730989  宝丰能源  20190430 ... 22.07 22.00 81.550 0.25
1999 300778.SZ  300778  新城市  20190425 ... 22.99 2.00 5.466 0.02

[2000 rows x 12 columns]
[In [5]: df.to_parquet("new_share.parquet")
[In [6]: ]

```

```

[In [7]: df = pro.stock_basic()
[In [8]: df.to_parquet("stock_basic.parquet")
[In [9]: df = pro.stock_basic(fields='ts_code,symbol,name,area,industry,fullname,ennname,cnspell
,market,exchange,curr_type,list_status,list_date,delist_date,is_hs,act_name,act_ent_ty
pe')
[In [10]: df
Out[10]:
   ts_code symbol name area ... delist_date is_hs act_name act_ent_type
0  000001.SZ  000001 平安银行 深圳 ... None S 无实际控制人 无
1  000002.SZ  000002 万科A 深圳 ... None S 深圳市人民政府国有资产监督管理委员会 地方国企
2  000004.SZ  000004 *ST国华 深圳 ... None N 李映彤 民营企业
3  000006.SZ  000006 深振业A 深圳 ... None S 深圳市人民政府国有资产监督管理委员会 地方国企
4  000007.SZ  000007 全新好 深圳 ... None N 王玩虹 民营企业
...
5407 920489.BJ 920489 佳先股份 None ... None N None None
5408 920682.BJ 920682 球冠电缆 None ... None N None None
5409 920799.BJ 920799 艾融软件 None ... None N None None
5410 920819.BJ 920819 颖泰生物 None ... None N None None
5411 689009.SH 689009 九号公司-WD 北京 ... None None None None

[5412 rows x 17 columns]
[In [11]: df.to_parquet("stock_basic.parquet2")
[In [12]: ]

```

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

首先在 VScode 内在 environment.yml 文件夹中添加 perspective-python 和 polars 依赖项，运行 conda env update 更新 Conda 环境。

启

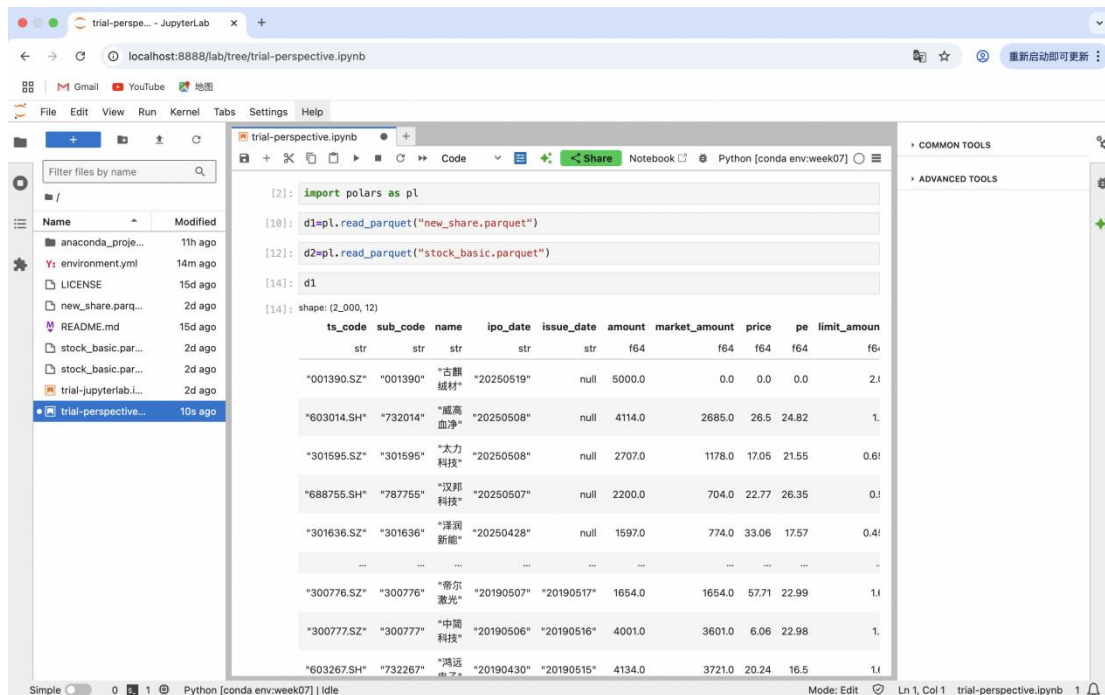
动 JupyterLab，新建一个 Notebook，改名为 trial-perspective.ipynb


```
Requirement already satisfied: charset_normalizer<4,>=2 in /opt/anaconda3/envs/week07/lib/python3.12/site-packages (from requests->tushare->r /Users/donghengxu/repo/week07/condaenv.yaup1s.requirements.txt (line 1)) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /opt/anaconda3/envs/week07/lib/python3.12/site-packages (from requests->tushare->r /Users/donghengxu/repo/week07/condaenv.yaup1s.requirements.txt (line 1)) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/anaconda3/envs/week07/lib/python3.12/site-packages (from requests->tushare->r /Users/donghengxu/repo/week07/condaenv.yaup1s.requirements.txt (line 1)) (2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in /opt/anaconda3/envs/week07/lib/python3.12/site-packages (from requests->tushare->r /Users/donghengxu/repo/week07/condaenv.yaup1s.requirements.txt (line 1)) (2025.4.26)

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

(base) donghengxu@donghengxudeMacBook-Air week07 % jupyterlab
```

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`



```
[2]: import polars as pl
[10]: d1=pl.read_parquet("new_share.parquet")
[12]: d2=pl.read_parquet("stock_basic.parquet")
[14]: d1
[14]: shape: (2,000, 12)
      ts_code  sub_code  name  ipo_date  issue_date  amount  market_amount  price  pe  limit_amount
      str      str      str      str      str      f64      f64      f64  f64  f64
-----
001390.SZ*  001390*  "古...  20250519*  null  5000.0      0.0  0.0  0.0  2.1
603014.SH*  732014*  "威...  20250508*  null  4114.0      2685.0  26.5  24.82  1.
301595.SZ*  301595*  "太...  20250508*  null  2707.0      1178.0  17.05  21.55  0.6
688755.SH*  787755*  "汉...  20250507*  null  2200.0      704.0  22.77  26.35  0.
301636.SZ*  301636*  "泽...  20250428*  null  1597.0      774.0  33.06  17.57  0.4
...      ...      ...      ...      ...      ...      ...      ...      ...
300776.SZ*  300776*  "帝...  20190507*  20190517*  1654.0      1654.0  57.71  22.99  1.
300777.SZ*  300777*  "中...  20190506*  20190516*  4001.0      3601.0  6.06  22.98  1.
603267.SH*  732267*  "鸿...  20190430*  20190515*  4134.0      3721.0  20.24  16.5  1.1
```

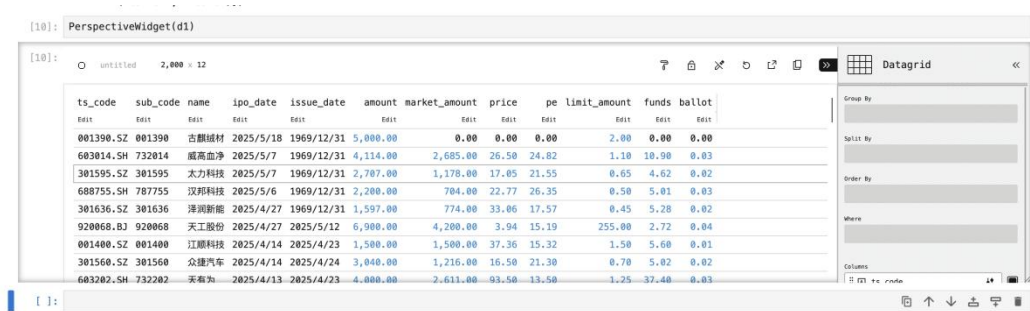
进行适当的列变换，尤其是要把实际上是日期类型的列，从 `polars.String()` 类型转换为 `polars.Date()` 类型

```
trial-perspective.ipynb
[16]: d1.with_columns(
      ipo_date=pl.col.ipo_date.str.to_date("%Y%m%d"),
      issue_date=pl.col.issue_date.str.to_date("%Y%m%d")
    )

[16]: shape: (2_000, 12)

   ts_code  sub_code  name  ipo_date  issue_date  amount  market_amount  price  pe  limit_amount  fi
   str      str      str    date      date      f64          f64      f64  f64  f64          f64
   --
   "001390.SZ" "001390" "古麒绒材" 2025-05-19 null 5000.0 0.0 0.0 0.0 2.0
   "603014.SH" "732014" "威高血净" 2025-05-08 null 4114.0 2685.0 26.5 24.82 1.1 10
   "301595.SZ" "301595" "太力科技" 2025-05-08 null 2707.0 1178.0 17.05 21.55 0.65 4
   "688755.SH" "787755" "汉邦科技" 2025-05-07 null 2200.0 704.0 22.77 26.35 0.5 5
   "301636.SZ" "301636" "泽润新能" 2025-04-28 null 1597.0 774.0 33.06 17.57 0.45 5
   ...
   "300776.SZ" "300776" "帝尔激光" 2019-05-07 2019-05-17 1654.0 1654.0 57.71 22.99 1.6 9
   "300777.SZ" "300777" "中简科技" 2019-05-06 2019-05-16 4001.0 3601.0 6.06 22.98 1.1 2
```

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)

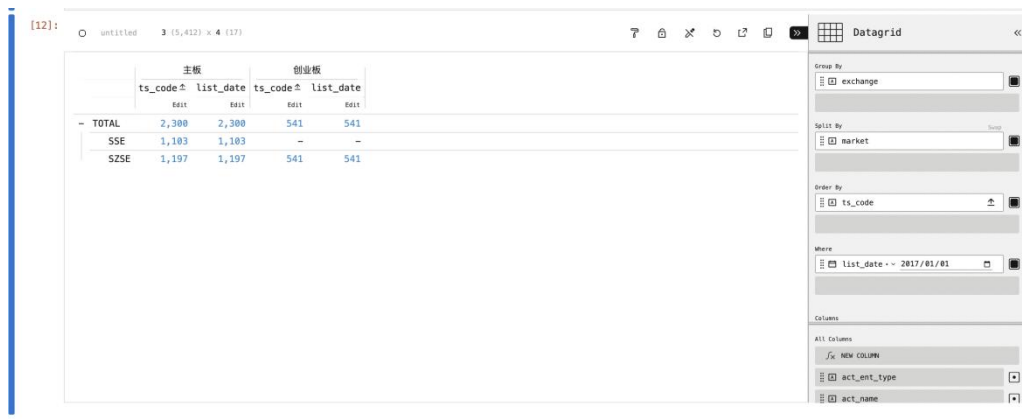
设置 Split By 选项，选择某些列作为拆分依据（横向排列）



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）

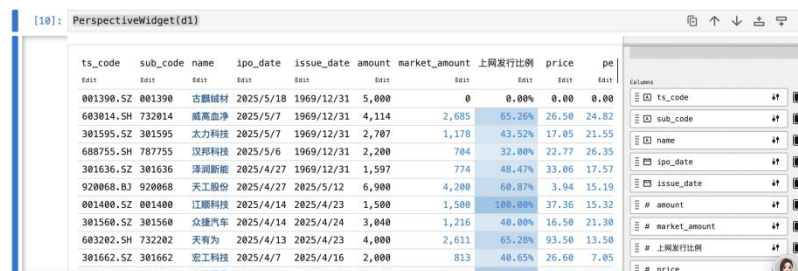


设置 Where 选项，选择某些列，进一步设置条件，进行数据行（观测）方向的过滤



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 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

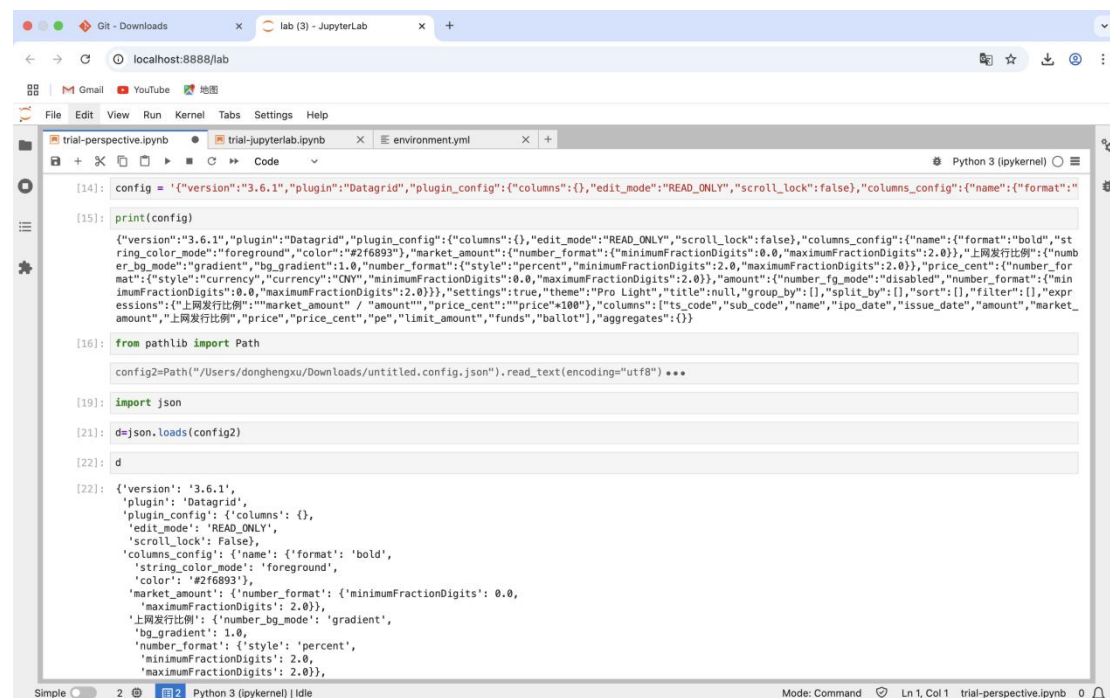
把设置 (config.json) 复制到剪贴板, 粘贴进 Notebook Cell, 保存成字符串 (str)

也可以把设置 (config.json) 导出为文件, 用 `pathlib.Path.read_text` 方法从文件读取出字符串 (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](#)) 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



```
[14]: config = '{"version": "3.6.1", "plugin": "Datagrid", "plugin_config": {"columns": {}, "edit_mode": "READ_ONLY", "scroll_lock": false}, "columns_config": {"name": {"format": "bold", "string_color_mode": "foreground", "color": "#2f6893"}, "market_amount": {"number_format": {"minimumFractionDigits": 0.0, "maximumFractionDigits": 2.0}, "上网发行比例": {"number_bg_mode": "gradient", "bg_gradient": 1.0, "number_format": {"style": "percent", "minimumFractionDigits": 2.0, "maximumFractionDigits": 2.0}, "price_cent": {"number_format": {"style": "currency", "currency": "QNY", "minimumFractionDigits": 0.0, "maximumFractionDigits": 2.0}, "amount": {"number_fg_mode": "disabled", "number_format": {"minimumFractionDigits": 0.0, "maximumFractionDigits": 2.0}}}, "settings": true, "theme": "Pro Light", "title": null, "group_by": [], "split_by": [], "sort": [], "filter": [], "expressions": [{"上网发行比例": "market_amount" / "amount", "price_cent": "price*100"}], "columns": [{"ts_code", "sub_code", "name", "ipo_date", "issue_date", "amount", "market_amount", "上网发行比例", "price", "price_cent", "pe", "Limit_amount", "funds", "ballot"}], "aggregates": {}}}
```

```
[15]: print(config)
```

```
[16]: from pathlib import Path
```

```
config2=Path("/Users/donghengxu/Downloads/untitled.config.json").read_text(encoding="utf8") ***
```

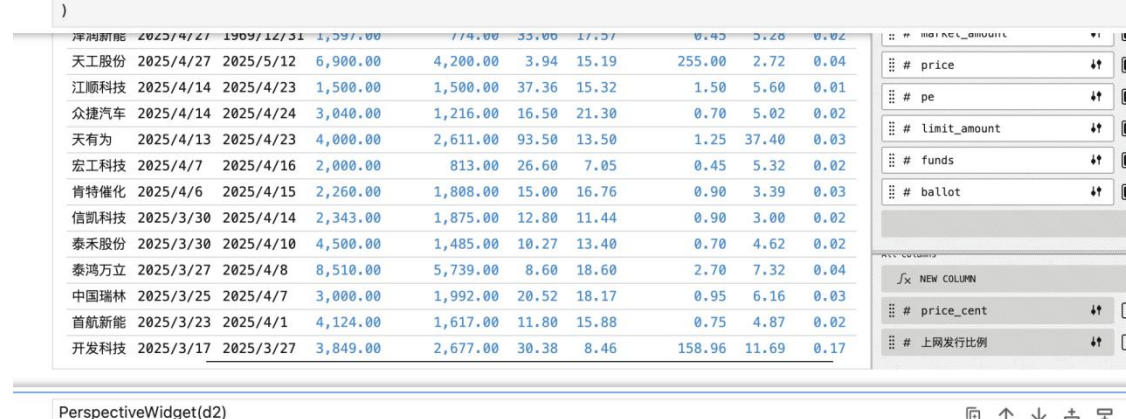
```
[19]: import json
```

```
[21]: d=json.loads(config2)
```

```
[22]: d
```

```
[22]: {'version': '3.6.1', 'plugin': 'Datagrid', 'plugin_config': {'columns': {}, 'edit_mode': 'READ_ONLY', 'scroll_lock': False}, 'columns_config': {'name': {'format': 'bold', 'string_color_mode': 'foreground', 'color': '#2f6893'}, 'market_amount': {'number_format': {'minimumFractionDigits': 0.0, 'maximumFractionDigits': 2.0}, '上网发行比例': {'number_bg_mode': 'gradient', 'bg_gradient': 1.0, 'number_format': {'style': 'percent', 'minimumFractionDigits': 2.0, 'maximumFractionDigits': 2.0}, 'price_cent': {'number_fg_mode': 'disabled', 'number_format': {'minimumFractionDigits': 0.0, 'maximumFractionDigits': 2.0}}}, 'settings': true, 'theme': 'Pro Light', 'title': null, 'group_by': [], 'split_by': [], 'sort': [], 'filter': [], 'expressions': [{"上网发行比例": "market_amount" / "amount", "price_cent": "price*100"}], 'columns': ['ts_code', 'sub_code', 'name', 'ipo_date', 'issue_date', 'amount', 'market_amount', '上网发行比例', 'price', 'price_cent', 'pe', 'Limit_amount', 'funds', 'ballot']}, 'aggregates': {}}}
```

```
251: PerspectiveWidget(  
    d1,  
    expressions={  
        '上网发行比例': "market_amount" / "amount",  
        'price_cent': "price*100"  
    }  
)
```



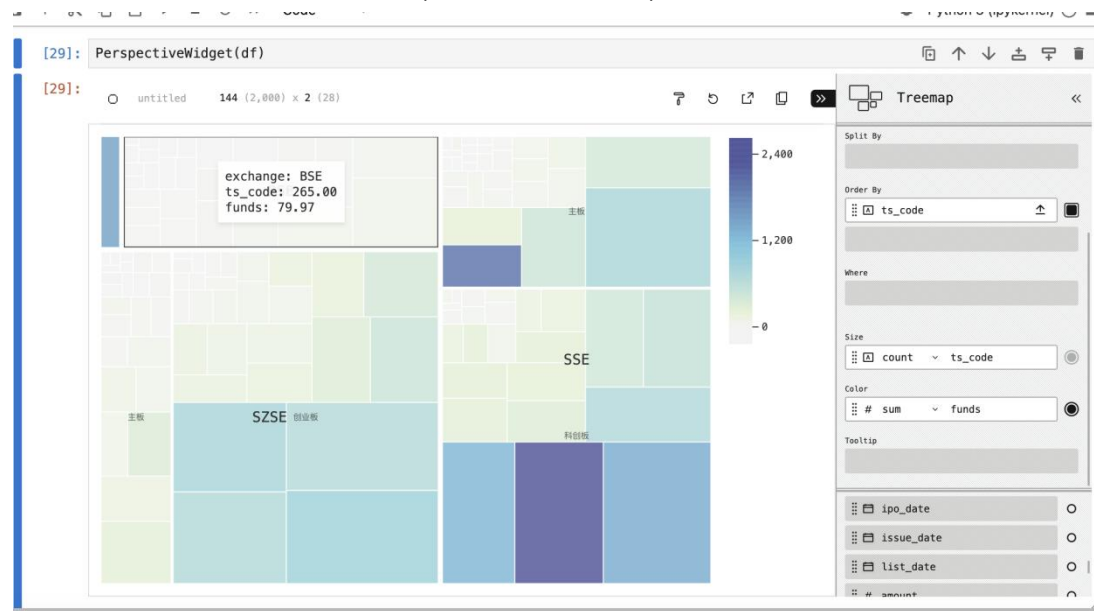
名称	日期	金额	比例	价格	市盈率	限制	资金	投票
天工股份	2025/4/27	6,900.00	4,200.00	3.94	15.19	255.00	2.72	0.04
江顺科技	2025/4/14	1,500.00	1,500.00	37.36	15.32	1.50	5.60	0.01
众捷汽车	2025/4/14	3,040.00	1,216.00	16.50	21.30	0.70	5.02	0.02
天有为	2025/4/13	4,000.00	2,611.00	93.50	13.50	1.25	37.40	0.03
宏工科技	2025/4/7	2,000.00	813.00	26.60	7.05	0.45	5.32	0.02
肯特催化	2025/4/6	2,260.00	1,808.00	15.00	16.76	0.90	3.39	0.03
信凯科技	2025/3/30	2,343.00	1,875.00	12.80	11.44	0.90	3.00	0.02
泰禾股份	2025/3/30	4,500.00	1,485.00	10.27	13.40	0.70	4.62	0.02
泰鸿万立	2025/3/27	8,510.00	5,739.00	8.60	18.60	2.70	7.32	0.04
中国瑞林	2025/3/25	3,000.00	1,992.00	20.52	18.17	0.95	6.16	0.03
首航新能	2025/3/25	4,124.00	1,617.00	11.80	15.88	0.75	4.87	0.02
开发科技	2025/3/17	3,849.00	2,677.00	30.38	8.46	158.96	11.69	0.17

NEW COLUMN

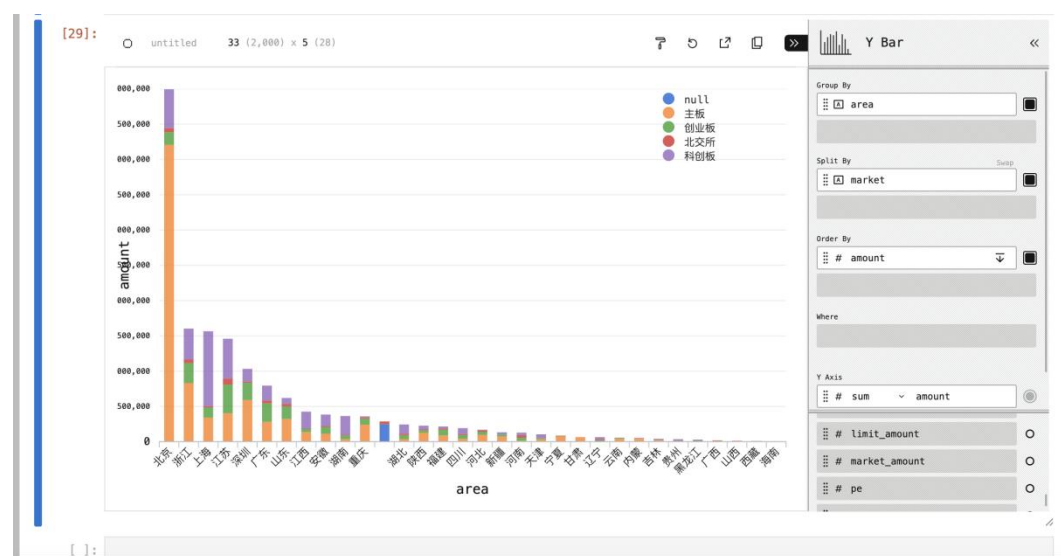
- # price
- # pe
- # limit_amount
- # funds
- # ballot
- # price_cent
- # 上网发行比例

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)

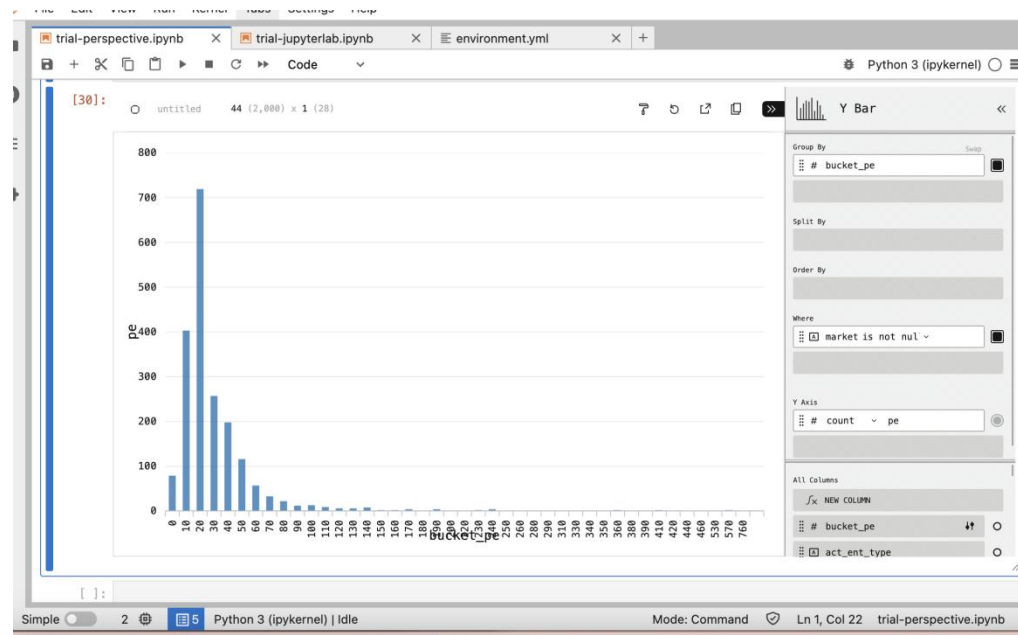


Y Bar (条形图/柱状图) 的横轴 (不同的条形) 是第一个维度, 用 `Group By` 控制, 纵轴 (条形的高度) 是第二个维度, 用 `Y Axis` 控制 (支持多变量并列显示), 还可以把每个条形进一步拆分为多个颜色, 用 `Split By` 控制



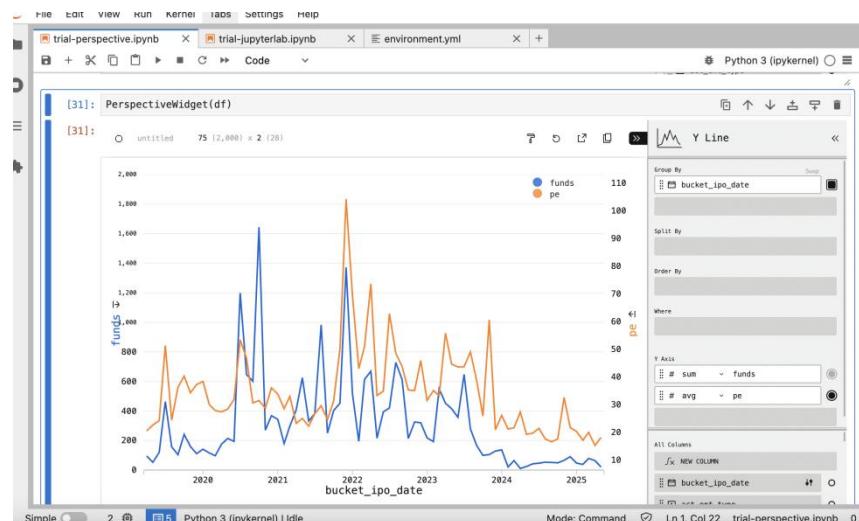
Y Bar 视图还可以用来实现一类很重要的统计制图 —— 直方图 (histogram)。对于数据表中的某一系列连续型数值变量 (比如新股发行的市盈率 `pe`), 我们经常希望观察其分布 (`distribution`)。可以用 `bucket` 函数对连续变量进行“分桶” (比如表达式 `bucket("pe", 10)`), 生成一个新的离散变量 (比如命名为 `bucket_pe`), 然后把离散变量设置为 **Y Bar** 的横轴 (`Group By`), 把任意其他一系列变量用 `count` (计数) 函数汇总, 设置为纵轴 (`Y Axis`)。这样看到的就是直方图。“分桶”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

视觉的实际效果

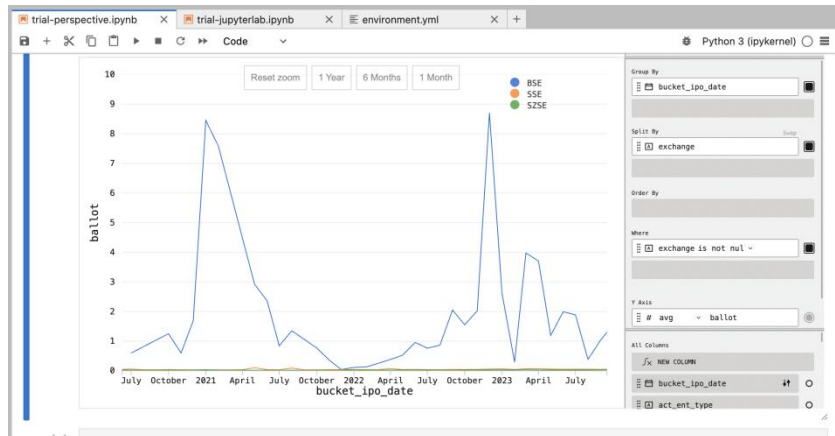


Y Line (折线图) 常用来绘制时间序列，横轴通常是时间，用 **Group By** 控制，纵轴 (折线的 Y 坐标) 通常是连续型数值变量 (经过汇总)，用 **Y Axis** 控制 (支持多序列同时显示)，还可以进一步拆分为多条序列，用 **Split By** 控制

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 (cinfigure)，观察数据可视化的实际效果

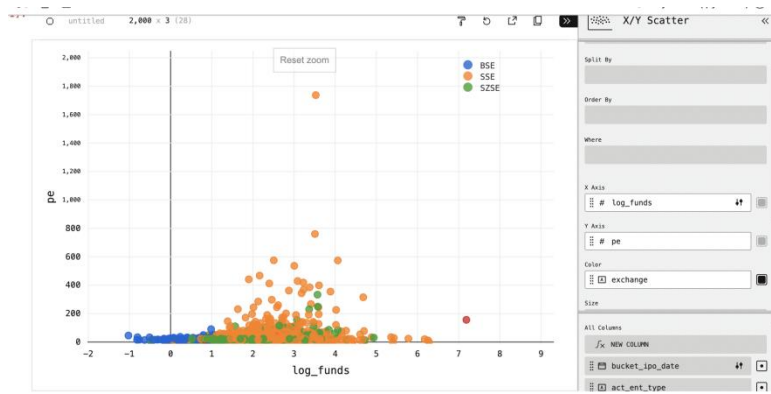
X/Y Scatter (散点图) 常用来观察两个数值型连续变量之间的相关关系 (correlation)。数据首先可以进行分组汇总，每一个组对应一个散点，用 **Group By** 控制。然后把两个连续型数值变量分别设置为 **X Axis** 和 **Y Axis**，其汇总数值将作为每个散点的坐标
散点的分布如果特别不均匀，则意味着变量单位可能有问题，或者需要经过变换 (比如取对数)

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**)，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

