

35

Build Streamlit Apps

Streamlit 搭建 Apps

用 Streamlit 搭建数学学习、数据科学、机器学习应用



没有对已有知识进行大量练习，你不太可能发现新事物；但更进一步，你应该从解决有趣的关系和有趣的问题中获得很多乐趣。

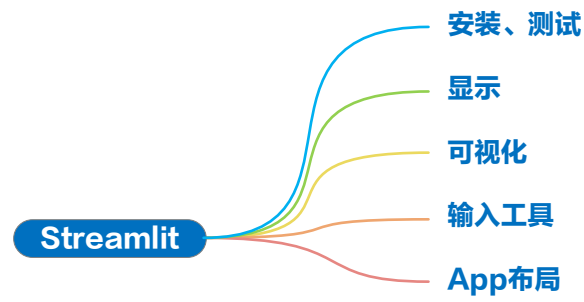
You're unlikely to discover something new without a lot of practice on old stuff, but further, you should get a heck of a lot of fun out of working out funny relations and interesting things.

—— 理查德·费曼 (Richard P. Feynman) | 美国理论物理学家 | 1918 ~ 1988



- ▶ `streamlit.area_chart()` 面积图
- ▶ `streamlit.bar_chart()` 直方图
- ▶ `streamlit.button()` 按钮，点击时会触发指定的动作
- ▶ `streamlit.checkbox()` 复选框，用户可以选择或取消选择
- ▶ `streamlit.color_picker()` 颜色选择器，用户可以选择颜色
- ▶ `streamlit.columns()` 创建多列布局
- ▶ `streamlit.container()` 是一个用于组织内容的容器
- ▶ `streamlit.date_input()` 日期输入框，用户可以选择日期
- ▶ `streamlit.expander()` 创建可展开的区域
- ▶ `streamlit.file_uploader()` 文件上传器，用户可以上传文件
- ▶ `streamlit.header()` 显示章节标题
- ▶ `streamlit.line_chart()` 线图
- ▶ `streamlit.markdown()` 显示 markdown 文本
- ▶ `streamlit.multiselect()` 多选框，用户可以从给定选项中选择多个
- ▶ `streamlit.number_input()` 数字输入框，用户可以输入数字
- ▶ `streamlit.plotly_chart()` 展示 Plotly 图像对象
- ▶ `streamlit.pyplot()` 展示 Matplotlib 图像对象
- ▶ `streamlit.radio()` 一组单选按钮，用户可以从给定选项选择一个
- ▶ `streamlit.select_slider()` 选择滑块，用户可以从给定选项选择一个值
- ▶ `streamlit.selectbox()` 下拉选择框，用户可以从给定选项选择一个
- ▶ `streamlit.sidebar()` 创建侧边栏
- ▶ `streamlit.slider()` 滑块，用户可以在指定范围内选择一个值
- ▶ `streamlit.tabs()` 创建选项卡式的布局
- ▶ `streamlit.text_area()` 多行文本输入框，用户可以输入多行文本
- ▶ `streamlit.text_input()` 文本输入框，用户可以输入文本
- ▶ `streamlit.time_input()` 时间输入框，用户可以选择时间
- ▶ `streamlit.title()` 显示标题
- ▶ `streamlit.write()` 显示字符串、数据帧、报错、函数、图像等等






35.1 什么是 Streamlit?

Streamlit 是一个用于构建数据科学和机器学习应用程序的开源 Python 库。Streamlit 能够以简单且快速的方式创建交互式应用程序，无需繁琐的前端开发。Streamlit 有如下主要功能。

- ▶ 用户交互：Streamlit 具有构建用户界面的功能，可以添加各种交互元素，例如滑块、下拉菜单和复选框，以使用户能够与应用程序进行互动，并动态地改变应用程序的行为。
- ▶ 数据可视化：Streamlit 提供了丰富的图表和可视化组件，能够直观地展示数据和模型的结果。Streamlit 还支持 Matplotlib、Seaborn、Plotly 等库创建图表，并将其集成到应用程序中。
- ▶ 模型展示：Streamlit 支持在应用程序中展示机器学习模型的结果。可以用 Streamlit 加载模型并使用它们对新数据进行预测。这对于展示模型的性能、解释结果或进行实时预测非常有用。
- ▶ 部署和共享：Streamlit 提供了一个简单的部署机制，可以轻松地将应用程序部署到 Web 上，并与其他人共享。

本章主要介绍如何使用 Streamlit 的核心功能，下两章分别介绍如何用 Streamlit 创建数据分析、机器学习相关 App 应用。

安装

安装 Anaconda 后，可以进一步安装 Streamlit。如图 1 所示，对于 Windows 用户，先打开 Anaconda Navigator，点进入 Environments，然后选择特定环境，左键点击  打开下拉菜单，选择 Open Terminal。大家也可以直接搜索打开 Anaconda Prompt，进入。进入 Prompt 之后，键入 `pip install streamlit`（注意，全小写、半角空格）安装。

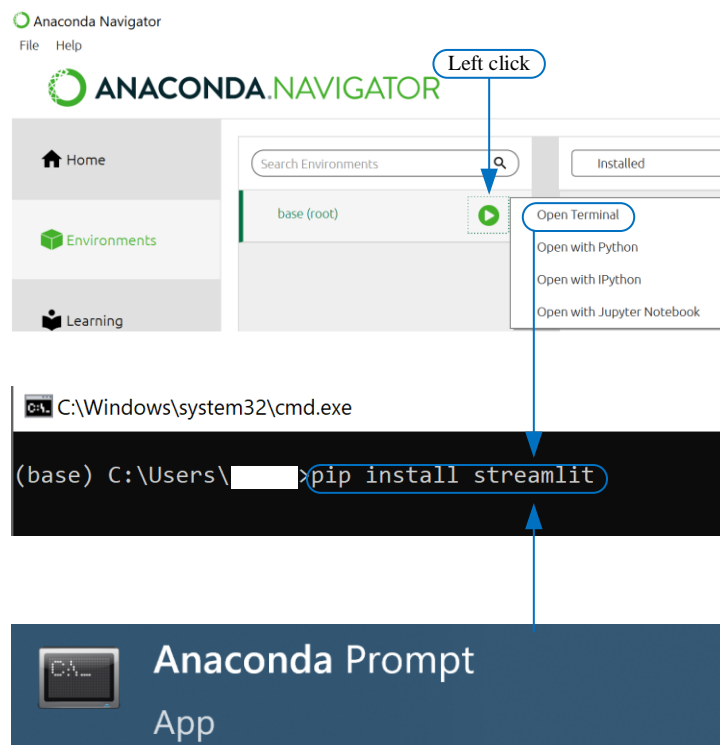


图 1. 安装 Streamlit

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

需要更新 Streamlit，请使用 `pip install streamlit --upgrade`。

对于 macOS 和 Linux 用户，请参考如下页面安装 Streamlit：

<https://docs.streamlit.io/library/get-started/installation>

测试

为了测试 Streamlit 安装成功，在 Anaconda Prompt 中大家可以键入 `streamlit hello`（注意，全小写、半角空格）。如果在默认浏览器中成功打开如图 2 下图所示网页，则成功安装 Streamlit。

如果不成功的话，请重新安装 Streamlit。如有必要可以关机重新开机再尝试安装。还是安装失败的话，可以卸载 Anaconda，再重新下载安装最新 Anaconda 后，在尝试重新安装 Streamlit。

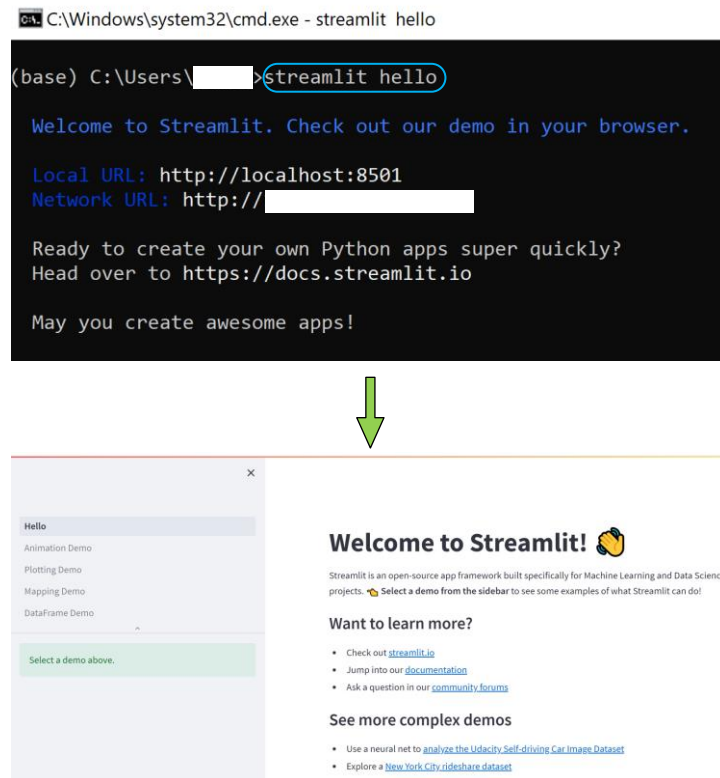


图 2. 安装 Streamlit

本章还提供了一个 Streamlit 测试代码——`streamlit_app_test.py`。

大家将配套测试代码下载保存到特定文件夹路径（比如 `C:\Users\james\Desktop\test_streamlit`），如果想要演示这个 App，大家可以在 Anaconda Prompt 键入 `streamlit_app_test.py` 所在文件夹路径，比如 `cd C:\Users\james\Desktop\test_streamlit`。其中，`cd` 表示 "Change Directory"，即切换目录的意思。这是用于在命令行中导航文件系统的命令。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

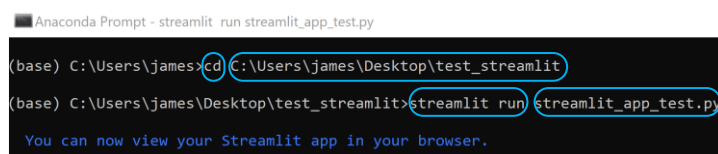
版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

然后键入 `streamlit run streamlit_app_test.py`。其中，`streamlit run` 是用于在 Anaconda Prompt 中启动和运行 Streamlit 应用程序的命令。



```

Anaconda Prompt - streamlit run streamlit_app_test.py
(base) C:\Users\james>cd C:\Users\james\Desktop\test_streamlit
(base) C:\Users\james\Desktop\test_streamlit>streamlit run streamlit_app_test.py
You can now view your Streamlit app in your browser.

```

图 3. 演示本章配套测试代码

IDE

虽然，Streamlit 社区中有用户创建了 JupyterLab 中开发 Streamlit Apps 的库，但是作者建议大家还是用 Spyder 或 PyCharm 作为开发 Streamlit Apps 的 IDE。

比如，本章配套代码的例子 `streamlit_app_test.py` 就是用 Spyder 完成。

强调一下，在各种 IDE 中运行 Python 文件并不能打开浏览器查看 Streamlit 应用程序。必须要在 Anaconda Prompt 中运行 `streamlit run _name_of_your_streamlit_app.py` (图 3) 才能查看交互应用程序。大家完全可以一边编程，一边在浏览器查看应用程序效果。如果程序运行一遍较快，可以在 App 浏览器右上角选择 Always rerun，这样一边编程，App 浏览器就跟着更新，这样方便 debug。



图 4. Streamlit 应用页面设置

API (Application Programming Interface) 直译为应用程序编程接口。简单来说，API 就是指是一些预先定义好的函数。下面我们介绍几类常用的 API 函数。

35.2 显示

代码 1 利用 Streamlit 的函数显示文字、图像，浏览器呈现的 App 效果如图 5 所示。

a 将 `streamlit` 导入，别名为 `st` (这是 Streamlit 官方通用别名，建议大家直接采用)。为了和官网技术文档保持一致，本章在介绍 Streamlit 函数时，也会直接采用 `st.function()`，而不是 `streamlit.function()`。

b 利用 `st.title()` 显示标题，这个函数的输入为 `str`。

Streamlit 最近还推出了渲染文本的语法，`:color[text to be colored]`。比如，**b** 中 `:red[Streamlit]` 用红色渲染 Streamlit。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

③ 利用 `st.header()` 显示章节标题。④ 利用 `st.markdown()` 显示 Markdown 文本。⑤ 利用 `st.write()` 显示数据帧。Streamlit 官网管 `st.write()` 叫“瑞士军刀”，根据官方技术文档，`st.write()` 几乎可以显示各种对象，比如字符串、数据帧、报错、函数、模块、图像对象（比如⑥）、sympy 符号数学表达式等等。

其他显示文本的函数还有，`st.subheader()`、`st.caption()`、`st.code()`、`st.text()`、`st.latex()`。请大家在 Spyder 中搭建 Streamlit Apps 尝试这些函数。

```

a import streamlit as st
import seaborn as sns
import plotly.express as px

# 显示标题
b st.title('Welcome to the world of :red[Streamlit]')
# 显示章节标题
c st.header('Pandas DataFrame')
# 显示 markdown 文本
d st.markdown("Load :blue[Iris Data Set]")
# 从Seaborn导入鸢尾花数据帧
df = sns.load_dataset('iris')
# 显示数据帧
e st.write(df)
# 显示章节标题
st.header('Visualize Using Heatmap')
fig = px.imshow(df.iloc[:, :-1])
# 显示热图
f st.write(fig)

```

代码 1. 用于显示的函数; Bk1_Ch35_01.py

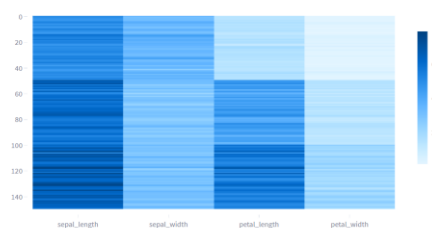
Welcome to the world of **Streamlit**

Pandas DataFrame

Load Iris Data Set

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa

Visualize Using Heatmap



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 5. 用于显示的函数，浏览器 App;  Bk1_Ch35_01.py

35.3 可视化

Streamlit 目前本身可视化方案有限，比如线图 (`st.line_chart()`)、面积图 (`st.area_chart()`)、直方图 (`st.bar_chart()`) 等。但是 Streamlit 支持其他主流 Python 可视化库，比如 Matplotlib、Plotly、Altair、Bokeh 等等。

代码 2 中 ^a 利用 `st.pyplot()` 专门绘制 Matplotlib 图像对象，大家自己打开 App 会发现这幅图为静态图像，也就是一幅图片。

而 ^b 利用 `st.plotly_chart()` 专门绘制 Plotly 图像对象，这幅图就是可交互的，大家可以在浏览器 App 中旋转、缩放这幅图。

```
import plotly.graph_objects as go
import numpy as np
import matplotlib.pyplot as plt
import streamlit as st

# 产生数据
x1_array = np.linspace(-3, 3, 301)
x2_array = np.linspace(-3, 3, 301)
xx1, xx2 = np.meshgrid(x1_array, x2_array)

# 二元函数的曲面数据
ff = xx1 * np.exp(-xx1**2 - xx2**2)

# Matplotlib图像
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(projection='3d')
ax.plot_wireframe(xx1, xx2, ff, rstride=10,
cstride=10)
a st.pyplot(fig)

# Plotly图像
fig = go.Figure(data=[go.Surface(z=ff, x=xx1, y=xx2,
                                colorscale = 'RdYlBu_r')])
b st.plotly_chart(fig)
```

代码 2. Streamlit 中的可视化示例;  Bk1_Ch35_02.py

35.4 输入工具

Streamlit 还支持各种输入工具 (input widget)，表 1 总结常用输入工具。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

请大家自行练习代码 3，并在浏览器查看输入工具效果。此外，建议大家查看每种输入工具返回值、类型，如 [a](#)、[b](#)。

表 1. Streamlit 常用输入工具

输入工具样式	说明	代码示例
	按钮，点击时会触发指定的动作	<pre>import streamlit as st st.button("Click me")</pre>
	复选框，用户可以选择或取消选择	<pre>st.checkbox("Check me")</pre>
Choose one:  Option 1  Option 2  Option 3	一组单选按钮，用户可以从给定选项中选择一个	<pre>st.radio("Choose one:", ["Option 1", "Option 2", "Option 3"])</pre>
Choose one: 	下拉选择框，用户可以从给定选项中选择一个	<pre>st.selectbox("Choose one:", ["Option 1", "Option 2", "Option 3"])</pre>
Choose many: 	多选框，用户可以从给定选项中选择多个	<pre>st.multiselect("Choose many:", ["A", "B", "C", "D"])</pre>
Select a value: 	滑块，用户可以在指定范围内选择一个值	<pre>st.slider("Select a value:", 0.0, 10.0, 5.0)</pre>
Select a value: 	滑块，用户可以从给定选项选择一个值	<pre>st.select_slider("Select a value:", options=[1, 2, 3, 4, 5])</pre>
Enter your name 	文本输入框，用户可以输入文本	<pre>st.text_input("Enter your name")</pre>
Enter a number 	数字输入框，用户可以输入数字	<pre>st.number_input("Enter a number")</pre>
Enter your message 	多行文本输入框，用户可以输入多行文本	<pre>st.text_area("Enter your message")</pre>
Select a date 	日期输入框，用户可以选择日期	<pre>st.date_input("Select a date")</pre>
Select a time 	时间输入框，用户可以选择时间	<pre>st.time_input("Select a time")</pre>

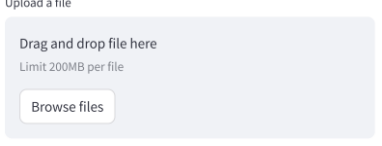

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>


欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

	文件上传器，用户可以上传文件	<code>st.file_uploader("Upload a file")</code>
	颜色选择器，用户可以选择颜色	<code>st.color_picker("Pick a color")</code>

```

import streamlit as st
a button_return = st.button("Click me")
b st.write(button_return)
st.checkbox("Check me")
st.radio("Choose one:",
        ["A", "B", "C"])
st.selectbox("Choose one:",
            ["A", "B", "C"])
st.multiselect("Choose many:",
              ["A", "B", "C", "D"])
st.slider("Select a value:",
          0.0, 10.0, 5.0)
st.select_slider("Select a value:",
                options=[1, 2, 3, 4, 5])
st.text_input("Enter your name")
st.number_input("Enter a number")
st.text_area("Enter your message")
st.date_input("Select a date")
st.time_input("Select a time")
st.file_uploader("Upload a file")
st.color_picker("Pick a color")

```

代码 3. Streamlit 的输入工具代码示例;  Bk1_Ch35_03.py

35.5 App 布局

Streamlit 提供几种 App 布局设计。

侧边栏 (sidebar) 对应的函数为 `st.sidebar()`，是 Streamlit 应用程序界面中的一个垂直边栏，可用于显示与主要内容相关的附加信息、控件和选项。侧边栏通常用于放置与应用程序设置、参数选择、数据过滤等相关的小部件。可以使用 `st.sidebar` 方法来在侧边栏中添加小部件。

如图 6 所示，这个 Streamlit 应用展示 a 、 b 、 c 三个参数对抛物线 ($f(x) = ax^2 + bx + c$) 影响。左侧边框中，用户可以通过 `st.slider()` 滑动选择 a 、 b 、 c 三个参数具体值。图 6 右侧主页面则分别打印函数，并展示函数图像。

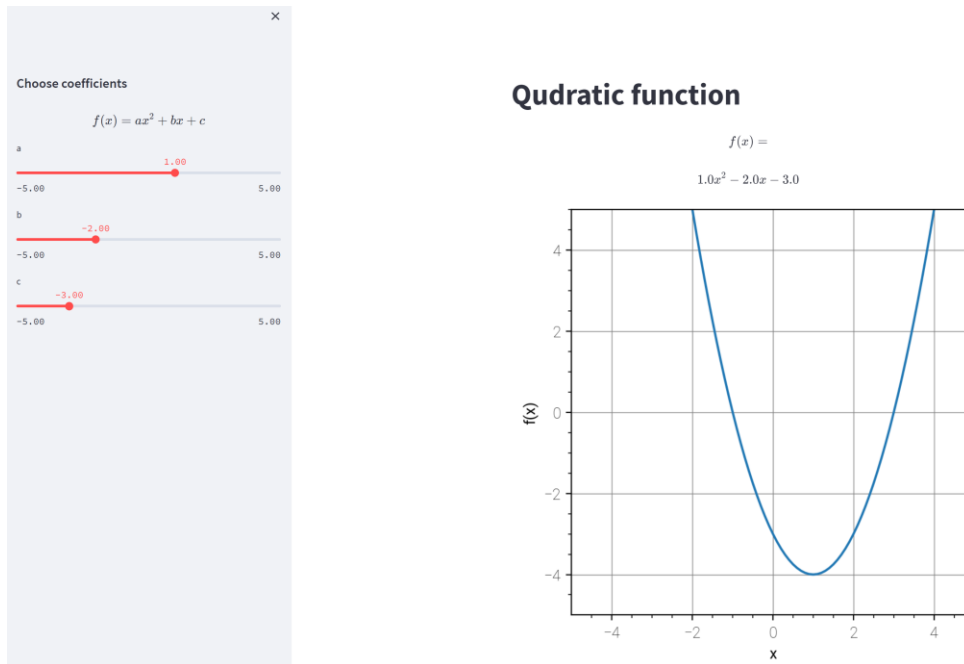


图 6. Streamlit 应用的侧边框;  Bk1_Ch35_04.py

代码 4 为创建图 6 中 Streamlit 应用。

- a 用 `with st.sidebar`: 创建了侧边框代码块。类似 `for loop`, 四个空格缩进用来表达代码块。
- b 用 `st.latex()` 打印 LaTeX 公式, 在侧边框展示 $f(x) = ax^2 + bx + c$ 。
- b 这一句还可以这样写, `st.sidebar.latex(r'f(x) = ax^2 + bx + c')`; 这种写法不需要缩进, 可以在侧边框代码块外部写。
- c 用 `st.slider()` 提供滑块输入工具, 用户可以选择输入数值, 这个数值赋值给变量 `a`。
`min_value = -5.0` 设定滑块最小值, `max_value = 5.0` 设定最大值, `step = 0.01` 设定滑块滑动步长, `value = 1.0` 设定滑块默认值。
- d 和 e 用同样输入工具给变量 `b`、`c` 赋值。
- f 创建 SymPy 符号数学表达式。
- g 利用 `sympy.lambdify()` 将符号数学表达式转化为 Python 函数。 h 计算抛物线函数值。
- i 用 `st.title()` 创建应用标题。
- j 用 `st.latex()` 将 SymPy 符号数学表达式以 LaTeX 形式打印在主页面上。
- k 用 `st.write()` 将 Matplotlib `fig` 对象显示在主页面上。

```

import streamlit as st
import numpy as np
from sympy import symbols, lambdify
import matplotlib.pyplot as plt

# 侧边框
a with st.sidebar:
    st.header('Choose coefficients')
    st.latex(r'f(x) = ax^2 + bx + c')
    a = st.slider("a", min_value = -5.0,
                  max_value = 5.0,
                  step = 0.01, value = 1.0)
    b = st.slider("b", min_value = -5.0,
                  max_value = 5.0,
                  step = 0.01, value = -2.0)
    c = st.slider("c", min_value = -5.0,
                  max_value = 5.0,
                  step = 0.01, value = -3.0)

# 抛物线
x = symbols('x')
f_x = a*x**2 + b*x + c
x_array = np.linspace(-5, 5, 101)
f_x_fcn = lambdify(x, f_x)
y_array = f_x_fcn(x_array)

# 主页面
i st.title('Quadratic function')
st.latex(r'f(x) = ')
j st.latex(f_x)

# 可视化
fig = plt.figure()
ax = fig.add_subplot(111)
ax.plot(x_array, y_array)

ax.set_xlim([-5, 5])
ax.set_ylim([-5, 5])
ax.set_aspect('equal', adjustable='box')
ax.set_xlabel('x')
ax.set_ylabel('f(x)')
k st.write(fig)

```

代码 4. Streamlit 应用的侧边框;  Bk1_Ch35_04.py

此外，函数 `st.columns()` 在 Streamlit 应用程序中创建多列布局，可以将内容水平分割成几个部分。通过这种方式，可以更好地控制内容的排列方式。

如代码 5 所示，^a 中 `st.columns(2)` 创建两列，对象分别是 `col1`、`col2`。我们还可以通过输入控制多列布局比例，比如 `col1, col2 = st.columns([3, 1])`，创建 `col1` 和 `col2` 比例为 3:1。

再比如 `col_A, col_B, col_C = st.columns([2,1,1])`，创建 `col_A, col_B, col_C` 比例为 2:1:1。




注意，目前 `st.columns()` 只能用在主页面中，不能用在侧边框。

b 在 `col1` 分栏显示文字，**c** 在 `col2` 分栏显示文字。类似侧边框，也可以用 `with col1:` 这种语法形式创建代码块。

```
import streamlit as st

# 在两列中显示不同的内容
a col1, col2 = st.columns(2)
b col1.write("This is column 1")
    col1.latex(r'f(x) = ax^2 + bx + c')

c col2.write("This is column 2")
```

代码 5. Streamlit 应用多列布局;  Bk1_Ch35_05.py

`st.tabs()` 可以用来创建选项卡式的布局，将相关的内容分组在不同的选项卡中，从而使应用程序界面更加清晰和易于导航。请大家自行学习代码 6。


`st.expander()` 创建可展开的区域，可以用来隐藏一些内容，让用户选择是否展开查看。请大家自行学习代码 7。

`st.container()` 创建组织内容的容器，可以用于控制内容的对齐方式和排列顺序。

```
import streamlit as st
# 创建两个选项卡，每个选项卡显示不同的内容
a tab_A, tab_B = st.tabs(["Tab A", "Tab B"])

b with tab_A:
    st.header("Tab A Title")
    st.write('This is Tab A.')

c with tab_B:
    st.header("Tab B Title")
    st.write('This is Tab B.')
```

代码 6. Streamlit 应用多选项卡布局;  Bk1_Ch35_06.py


```

import streamlit as st
import seaborn as sns
import plotly.express as px

# 显示标题
st.title('Iris Dataset')

# 从Seaborn导入鸢尾花数据集
df = sns.load_dataset('iris')
# 第一个可展开区域
a with st.expander("Open and view DataFrame"):
    # 显示数据集
    st.write(df)
# 第二个可展开区域
b with st.expander("Open and view Heatmap"):
    fig = px.imshow(df.iloc[:, :-1])
    # 显示热图
    st.write(fig)

```

代码 7. Streamlit 应用可展开区域;  Bk1_Ch35_07.py

本章唯一题目就是请大家在 Spyder 中复刻 Python 代码，然后分别执行打开每个 Streamlit 应用 App。

* 题目很基础，本书不给答案。



想要更全面了解 Streamlit 功能，请大家关注：

<https://docs.streamlit.io/library/api-reference>

Streamlit 社区开发者、用户开开发了很多小插件，请大家参考：

<https://extras.streamlit.app/>



请大家注意，本章仅仅介绍一些常用 Streamlit 功能；Streamlit 近期获得很大关注，用户量不断激增，开发团队不断增加新的功能、推出新版本，因此 Streamlit 语法也可能发生更新。

本书前文提过，Streamlit 特别适合快速搭建、部署机器学习 App。Streamlit 和各种 Python 第三方库兼容性极高，这也是鸢尾花书全系列采用 Streamlit 的原因。下一章，也是本书最后一章，将用 Streamlit 搭建几个机器学习 App 应用，总结本书所学！