STREAMLIT FOR DATA SCIENTISTS

Streamlit Overview | 2025

Joan Jaylani

INTRODUCTION TO STREAMLIT

WHAT IS STREAMLIT?

- Open-source Python library for building interactive web apps
- Designed for data science and machine learning workflows
- No web development skills required!

WHY USE STREAMLIT?

- Share results and dashboards instantly
- Real-time user interaction with Python code
- Perfect for demos, exploratory data analysis, and machine learning apps

- Jupyter: Great for prototyping, but static output
- Streamlit: Instantly interactive and web-based
- Easier to share apps with non-coders

STREAMLIT VS. JUPYTER NOTEBOOK

STREAMLIT CORE CONCEPTS

STREAMLIT BUILDING BLOCKS

- Widgets: sliders, buttons, selectboxes, etc.
- Layout: sidebar, columns, tabs
- Markdown, images, video, and more
- Caching for performance

HOW STREAMLIT SCRIPTS RUN

- Reruns top-to-bottom on any widget change
- Session state persists across tabs and steps

STREAMLIT EXAMPLES

MINIMAL APP EXAMPLE

- `import streamlit as st`
- `st.title("Hello Streamlit!")`
- `st.write("Welcome to your first data science app.")`

MATPLOTLIB PLOT EXAMPLE

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.hist(df["age"])
st.pyplot(fig)
```

WIDGETS AND INTERACTIVITY

Add widgets to collect user input:

- st.slider, st.button, st.selectbox, st.text_input

Use inputs directly in your code logic

WIDGET EXAMPLE

```
age = st.slider("Select your age:", 0, 100)
st.write(f"Selected age: {age}")
```

WORKING WITH DATA

READING AND DISPLAYING DATA

Use st.file_uploader to upload CSV/Excel/JSON

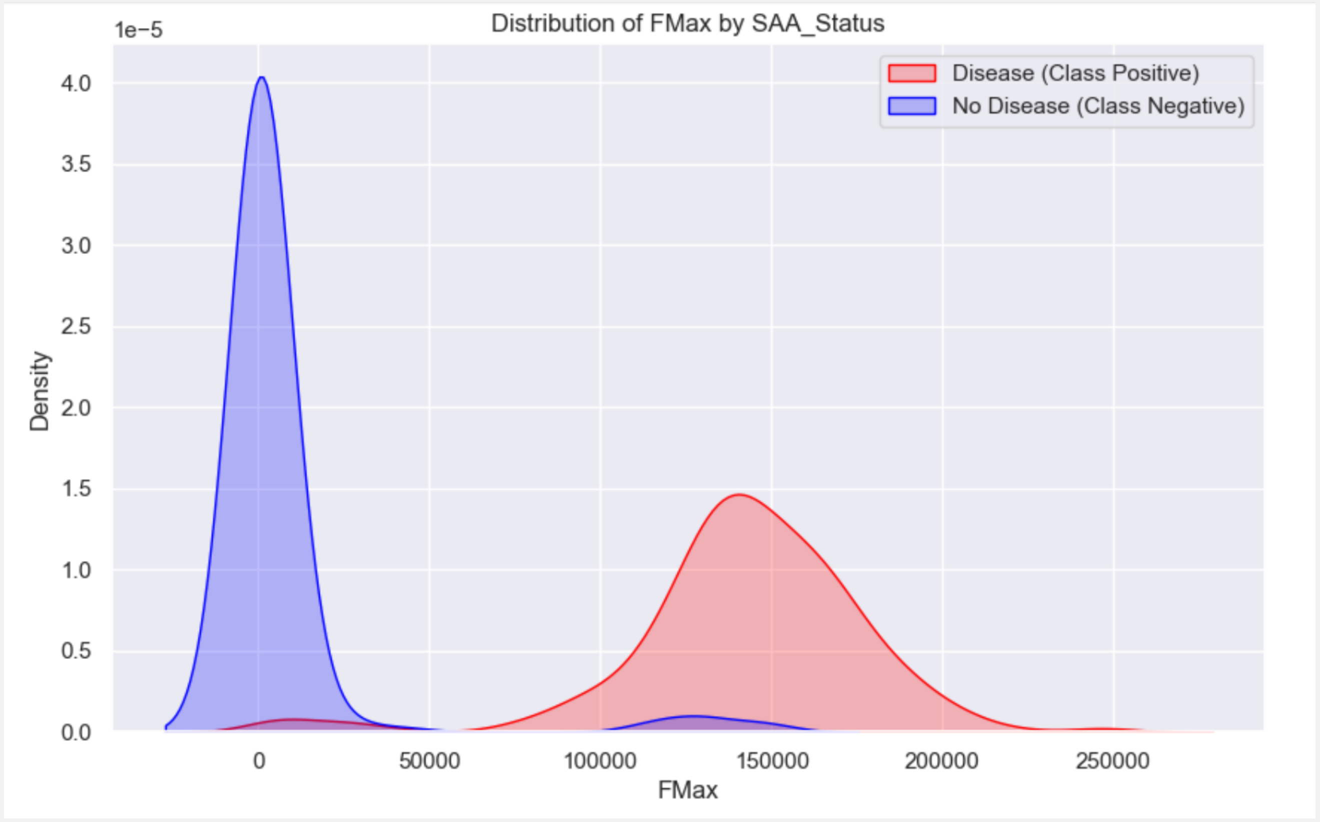
Show data with st.dataframe(df.head())

Display tables, plots, and model results

DATA UPLOAD EXAMPLE

```
uploaded_file = st.file_uploader("Upload a CSV file")
if uploaded_file:
    df = pd.read_csv(uploaded_file)
    st.dataframe(df.head())
```

CHARTS AND VISUALIZATIONS



Supports matplotlib, seaborn, plotly, and more

BUILDING MULTI-PAGE APPS

LAYOUT AND NAVIGATION

- Sidebar for navigation: st.sidebar
- Tabs: st.tabs(['EDA', 'ML', 'Results'])
- Columns for advanced layout: st.columns

TAB EXAMPLE

- `tabs = st.tabs(['Data', 'Visualization'])`
- `with tabs[0]:`
- `    st.write('Show data')`
- `with tabs[1]:`
- `    st.write('Show charts')`

FROM NOTEBOOK TO WEB APP

JUPYTER NOTEBOOK FLOW

- Data load ➔ analysis ➔ plot ➔ ML model ➔ summary
- Not easily shareable as an interactive app

HOW TO CONVERT NOTEBOOK TO STREAMLIT

1. Copy code to a new .py file
2. Replace display/output code with Streamlit widgets
3. Add user controls: sliders, file upload, checkboxes
4. Use st.write, st.dataframe, st.pyplot for output
5. Run: streamlit run my_app.py

NOTEBOOK CELL ➔ STREAMLIT WIDGET

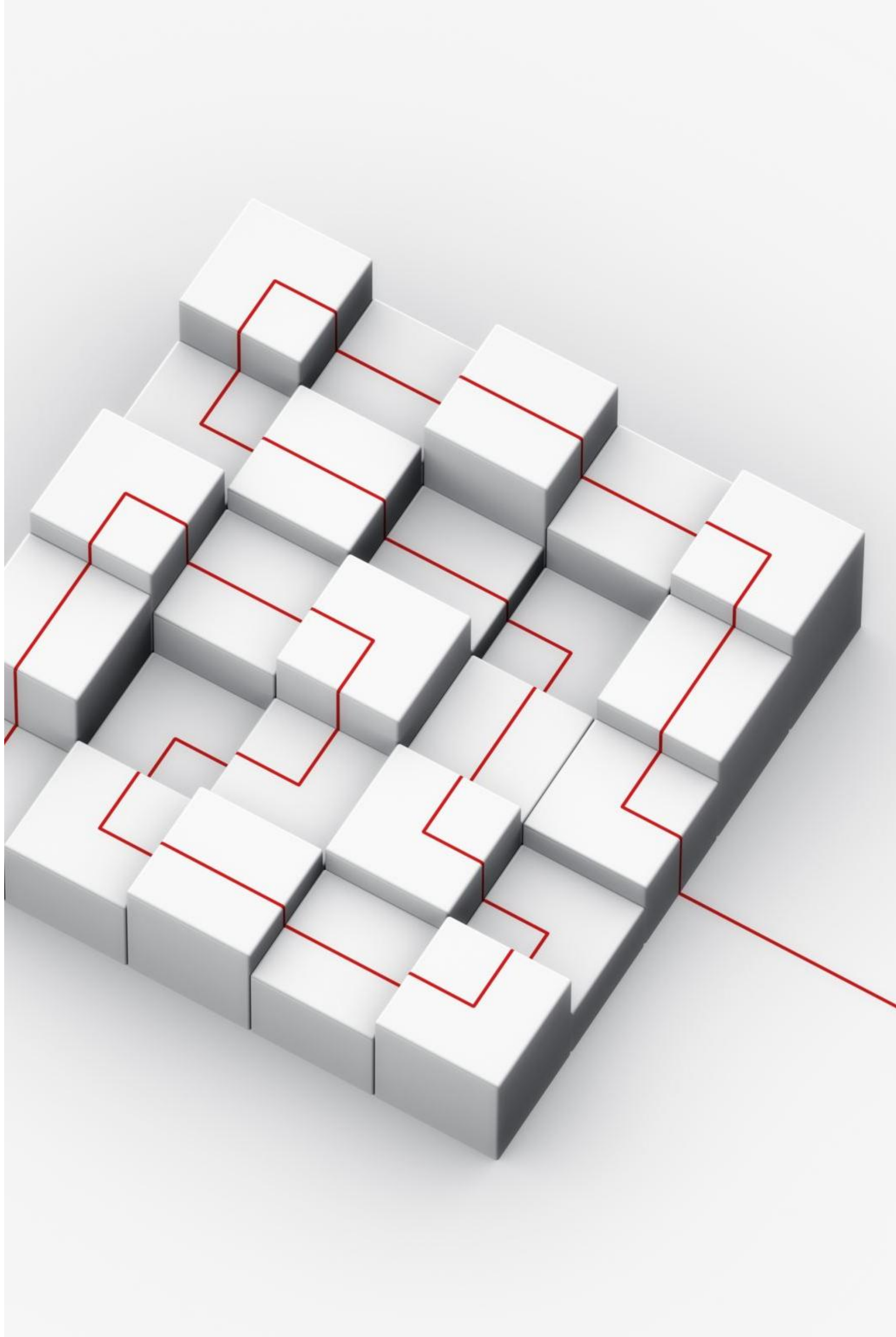
```
# Jupyter
print(df.head())
```

```
# Streamlit
st.dataframe(df.head())
```

REAL-WORLD EXAMPLE: MODULAR APP

MULTI-PAGE APP EXAMPLE

- Uses st.tabs to organize workflow
- Data upload, profiling, clustering, ML automation, AI review
- Separation of logic into modules for clarity



BEST PRACTICES & RESOURCES

BEST PRACTICES

Keep

Keep heavy computation in @st.cache_data functions

Modularize

Modularize code for readability

Use

Use session state for multi-page/tab logic

Deploy

Deploy to Streamlit Cloud for sharing

THANK YOU

Joan Jaylani

203-640-4593

Joan.Jaylani@gmail.com