```
pip install dash
```

```
Collecting dash
  Downloading dash-2.18.2-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: Flask<3.1,>=1.0.4 in /usr/local/lib/python3.10/dist-packages (from dash) (3.0.3)
Collecting Werkzeug<3.1 (from dash)
  Downloading werkzeug-3.0.6-py3-none-any.whl.metadata (3.7 kB)
Requirement already satisfied: plotly>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (5.24.1)
Collecting dash-html-components==2.0.0 (from dash)
  Downloading dash_html_components-2.0.0-py3-none-any.whl.metadata (3.8 kB)
Collecting dash-core-components==2.0.0 (from dash)
  Downloading dash_core_components-2.0.0-py3-none-any.whl.metadata (2.9 kB)
Collecting dash-table==5.0.0 (from dash)
  Downloading dash_table-5.0.0-py3-none-any.whl.metadata (2.4 kB)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from dash) (8.5.0)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from dash) (4.12.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from dash) (2.32.3)
Collecting retrying (from dash)
  Downloading retrying-1.3.4-py3-none-any.whl.metadata (6.9 kB)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.10/dist-packages (from dash) (1.6.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from dash) (75.1.0)
Requirement already satisfied: Jinja2>=3.1.2 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (3.1.4)
Requirement already satisfied: itsdangerous>=2.1.2 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (2.2.0
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (1.9.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (9.0.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (24.2)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from Werkzeug<3.1->dash) (3.0.2)
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata->dash) (3.21.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (2024.8.30)
Requirement already satisfied: six>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from retrying->dash) (1.17.0)
Downloading dash-2.18.2-py3-none-any.whl (7.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.8/7.8 MB 43.4 MB/s eta 0:00:00
Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)
Downloading dash_html_components-2.0.0-py3-none-any.whl (4.1 kB)
Downloading dash_table-5.0.0-py3-none-any.whl (3.9 kB)
Downloading werkzeug-3.0.6-py3-none-any.whl (227 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 228.0/228.0 kB 14.9 MB/s eta 0:00:00
Downloading retrying-1.3.4-py3-none-any.whl (11 kB)
Installing collected packages: dash-table, dash-html-components, dash-core-components, Werkzeug, retrying, dash
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 3.1.3
    Uninstalling Werkzeug-3.1.3:
      Successfully uninstalled Werkzeug-3.1.3
Successfully installed Werkzeug-3.0.6 dash-2.18.2 dash-core-components-2.0.0 dash-html-components-2.0.0 dash-table-5.0.0 retrying-1
```

Start coding or generate with AI.

```
import dash
from dash import dcc, html
import plotly.express as px
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.ensemble import RandomForestRegressor
from dash.exceptions import PreventUpdate
```

```
app = dash.Dash(__name__)

metadata_file = "metadata.csv"
metadata_df = pd.read_csv(metadata_file)
metadata_df
```

	type	start_time	ambient_temperature	battery_id	test_id	uid	filename	Capacity	Re	
0	discharge	[2010. 7. 21. 15. 0. ...	4	B0047	0	1	00001.csv	1.6743047446975208	NaN	
1	impedance	[2010. 7. 21. 16. 53. ...	24	B0047	1	2	00002.csv	NaN	0.05605783343888099	0.200
2	charge	[2010. 7. 21. 17. 25. ...	4	B0047	2	3	00003.csv	NaN	NaN	
3	impedance	[2010 7 21 20 31 5]	24	B0047	3	4	00004.csv	NaN	0.05319185850921101	0.164
4	discharge	[2.0100e+03 7.0000e+00 2.1000e+01 2.1000e+01 2...	4	B0047	4	5	00005.csv	1.5243662105099023	NaN	
...	...	...	...	...	...	...	...	...	...	
7560	impedance	[2010. 9. 30. 7. 36. ...	24	B0055	247	7561	07561.csv	NaN	0.0968087979207628	0.154

Start coding or generate with AI.

```
print(metadata_df.info())
print(metadata_df.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7565 entries, 0 to 7564
Data columns (total 10 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   type                 7565 non-null   object
 1   start_time           7565 non-null   object
 2   ambient_temperature  7565 non-null   int64 
 3   battery_id           7565 non-null   object
 4   test_id              7565 non-null   int64 
 5   uid                  7565 non-null   int64 
 6   filename             7565 non-null   object
 7   Capacity             2794 non-null   object
 8   Re                   1956 non-null   object
 9   Rct                  1956 non-null   object
dtypes: int64(3), object(7)
memory usage: 591.1+ KB
None
type                      0
start_time                0
ambient_temperature       0
battery_id                0
test_id                   0
uid                       0
filename                  0
Capacity               4771
Re                     5609
Rct                    5609
dtype: int64
```

```
metadata_df['Capacity'] = pd.to_numeric(metadata_df['Capacity'], errors='coerce')
capacity_mean = metadata_df['Capacity'].mean()
metadata_df['Capacity'] = metadata_df['Capacity'].fillna(capacity_mean)

metadata_df['Re'] = pd.to_numeric(metadata_df['Re'], errors='coerce')
metadata_df['Rct'] = pd.to_numeric(metadata_df['Rct'], errors='coerce')

re_mean = metadata_df['Re'].mean()
rct_mean = metadata_df['Rct'].mean()
metadata_df['Re'] = metadata_df['Re'].fillna(re_mean)
metadata_df['Rct'] = metadata_df['Rct'].fillna(rct_mean)
```

```
print(metadata_df.info())
print(metadata_df.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7565 entries, 0 to 7564
Data columns (total 10 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   type                 7565 non-null   object
 1   start_time           7565 non-null   object
 2   ambient_temperature  7565 non-null   int64
 3   battery_id           7565 non-null   object
 4   test_id              7565 non-null   int64
 5   uid                  7565 non-null   int64
 6   filename             7565 non-null   object
 7   Capacity             7565 non-null   float64
 8   Re                   7565 non-null   float64
 9   Rct                  7565 non-null   float64
dtypes: float64(3), int64(3), object(4)
memory usage: 591.1+ KB
None
type                   0
start_time             0
ambient_temperature    0
battery_id             0
test_id                0
uid                    0
filename               0
Capacity               0
Re                     0
Rct                    0
dtype: int64
```

```
print(metadata_df.describe())
```

```
       ambient_temperature      test_id          uid     Capacity  \
count          7565.000000  7565.000000  7565.000000  7565.000000
mean             20.017713   176.012558  3783.000000     1.326543
std              11.082914   152.174147  2183.971726     0.285841
min               4.000000     0.000000     1.000000     0.000000
25%               4.000000    55.000000  1892.000000     1.326543
50%              24.000000   129.000000  3783.000000     1.326543
75%              24.000000   255.000000  5674.000000     1.326543
max              44.000000   615.000000  7565.000000     2.640149

                 Re           Rct
count  7.565000e+03  7.565000e+03
mean  -4.976500e+11  1.055903e+12
std    1.113788e+13  2.363212e+13
min   -9.689245e+14 -2.091081e+02
25%   -4.976500e+11  2.405557e-01
50%   -4.976500e+11  1.055903e+12
75%    4.033353e-02  1.055903e+12
max    4.482291e+02  2.055843e+15
```

```
from sklearn.preprocessing import StandardScaler

# Select features for modeling
features = metadata_df[['ambient_temperature', 'Re', 'Rct']]

scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

metadata_df[['ambient_temperature_scaled', 'Re_scaled', 'Rct_scaled']] = scaled_features
```

Start coding or generate with AI.

```
metadata_df
```

	type	start_time	ambient_temperature	battery_id	test_id	uid	filename	Capacity	Re	Rct	ambient
0	discharge	[2010. 7. 21. 15. 0. ...	4	B0047	0	1	00001.csv	1.674305	-4.976500e+11	1.055903e+12	
1	impedance	[2010. 7. 21. 16. 53. ...	24	B0047	1	2	00002.csv	1.326543	5.605783e-02	2.009702e-01	
2	charge	[2010. 7. 21. 17. 25. ...	4	B0047	2	3	00003.csv	1.326543	-4.976500e+11	1.055903e+12	
3	impedance	[2010 7 21 20 31 5]	24	B0047	3	4	00004.csv	1.326543	5.319186e-02	1.647340e-01	
4	discharge	[2.0100e+03 7.0000e+00 2.1000e+01 2.1000e+01 2...	4	B0047	4	5	00005.csv	1.524366	-4.976500e+11	1.055903e+12	
...	...	...	...	...	...	...	...	...	...	...	
7560	impedance	[2010. 9. 30. 7. 36. ...	24	B0055	247	7561	07561.csv	1.326543	9.680880e-02	1.548974e-01	
7561	discharge	[2010. 9. 30. 8. 8. ...	4	B0055	248	7562	07562.csv	1.020138	-4.976500e+11	1.055903e+12	
7562	charge	[2010. 9. 30. 8. 48. 54.25]	4	B0055	249	7563	07563.csv	1.326543	-4.976500e+11	1.055903e+12	
7563	discharge	[2010. 9. 30. 11. 50. ...	4	B0055	250	7564	07564.csv	0.990759	-4.976500e+11	1.055903e+12	
7564	charge	[2010. 9. 30. 12. 31. ...	4	B0055	251	7565	07565.csv	1.326543	-4.976500e+11	1.055903e+12	

7565 rows × 13 columns

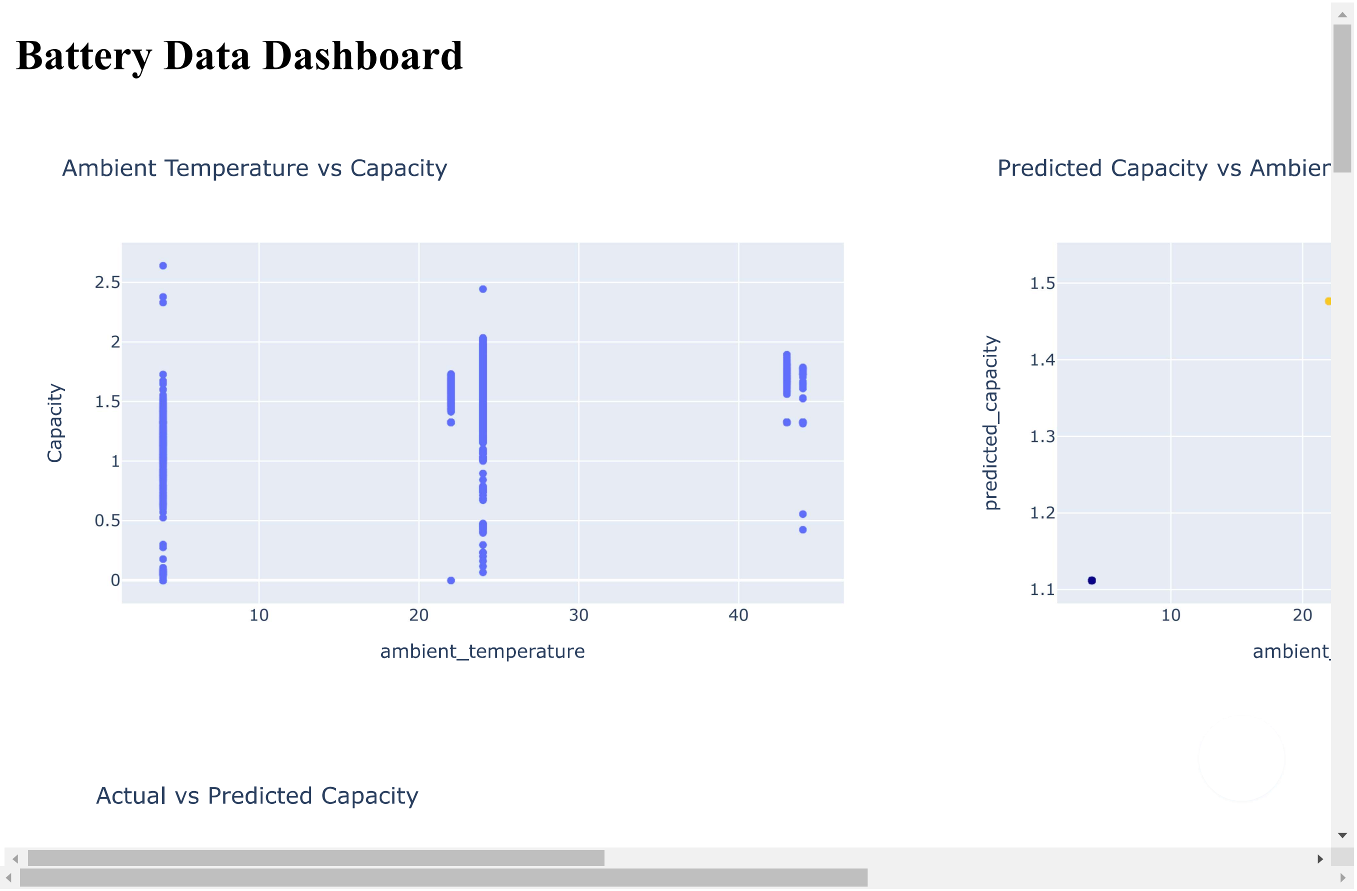
```
import dash
from dash import dcc, html
import plotly.express as px

# Create a Dash application
app = dash.Dash(__name__)

# Sample Plotly plot (scatter plot)
fig = px.scatter(metadata_df, x='ambient_temperature', y='Capacity', title="Ambient Temperature vs Capacity")

# Define the layout of the dashboard
app.layout = html.Div([
    html.H1("Battery Data Dashboard"),
    dcc.Graph(id='scatter-plot', figure=fig),
])

# Run the app
if __name__ == '__main__':
    app.run_server(debug=True)
```



Battery Data Dashboard

```
app = dash.Dash(__name__)

X = metadata_df[['ambient_temperature_scaled', 'Re_scaled', 'Rct_scaled']]
y = metadata_df['Capacity']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

knn_model = KNeighborsRegressor(n_neighbors=10, weights='uniform', metric='euclidean')
knn_model.fit(X_train_scaled, y_train)

y_pred = knn_model.predict(X_test_scaled)

metadata_df['predicted_capacity'] = knn_model.predict(scaler.transform(X))

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("Final Model MSE:", mse)
print("Final Model R²:", r2)
```

```
Final Model MSE: 0.06140122044996822
Final Model R²: 0.22388127927006296
```

```
fig1 = px.scatter(metadata_df, x='ambient_temperature', y='Capacity', title="Ambient Temperature vs Capacity")
fig2 = px.scatter(metadata_df, x='ambient_temperature', y='predicted_capacity', title="Predicted Capacity vs Ambient Temperature", col
fig3 = px.line(metadata_df, x='ambient_temperature', y=['Capacity', 'predicted_capacity'], title="Actual vs Predicted Capacity")
fig4 = px.bar(metadata_df, x='ambient_temperature', y=['Capacity', 'predicted_capacity'], title="Capacity Comparison", barmode='group'
fig5 = px.histogram(metadata_df, x='ambient_temperature', nbins=30, title="Distribution of Ambient Temperature")
fig6 = px.histogram(metadata_df, x='Capacity', nbins=30, title="Distribution of Capacity")
fig7 = px.imshow(metadata_df[['ambient_temperature_scaled', 'Re_scaled', 'Rct_scaled', 'Capacity']].corr(), title="Correlation Heatmap
fig8 = px.box(metadata_df, y='Capacity', title="Box Plot of Capacity")
fig9 = px.violin(metadata_df, y='Capacity', box=True, points="all", title="Violin Plot of Capacity")
fig10 = px.scatter_3d(metadata_df, x='ambient_temperature', y='Re_scaled', z='Rct_scaled', color='Capacity', title="3D Scatter Plot of

# Handling layout with multiple plots
app.layout = html.Div([
    html.H1("Battery Data Dashboard"),

    # Scatter and Line Plots
    html.Div([
        dcc.Graph(id='scatter-plot', figure=fig1),
        dcc.Graph(id='scatter-predicted', figure=fig2)
    ], style={'display': 'flex', 'flex-direction': 'row', 'justify-content': 'space-between'}),

    # Actual vs Predicted Capacity Line Plot
    html.Div([
        dcc.Graph(id='line-plot', figure=fig3),
    ], style={'margin-top': '20px'}),

    # Bar Plot and Histograms
    html.Div([
        dcc.Graph(id='bar-plot', figure=fig4),
    ], style={'margin-top': '20px'}),

    html.Div([
        dcc.Graph(id='histogram-temp', figure=fig5),
        dcc.Graph(id='histogram-capacity', figure=fig6)
    ], style={'display': 'flex', 'flex-direction': 'row', 'justify-content': 'space-between', 'margin-top': '20px'}),

    # Heatmap and Distribution Plots
    html.Div([
        dcc.Graph(id='heatmap', figure=fig7),
    ], style={'margin-top': '20px'}),

    html.Div([
        dcc.Graph(id='box-plot', figure=fig8),
        dcc.Graph(id='violin-plot', figure=fig9)
    ], style={'display': 'flex', 'flex-direction': 'row', 'justify-content': 'space-between', 'margin-top': '20px'}),

    # 3D Scatter Plot and Pairwise Scatter Plot
    html.Div([
        dcc.Graph(id='3d-scatter', figure=fig10),
    ], style={'margin-top': '20px'}),



    # Display Model Performance
    html.Div([
        html.H3(f"Mean Squared Error: {mse:.4f}"),
        html.H3(f"R² Score: {r2:.4f}")
    ])
])

# Run the app
if __name__ == '__main__':
    try:
```