resources-vis

May 13, 2024

1 Imports

[7]:
```
import numpy as np
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import pandas as pd
```

2 Load Data and Preprocessing

[8]:
```
from loadAndPreprocess import load_and_preprocess

'''
well_info: Well, X, Y, Total Resources
sensor_data: Depth, Porosity, Hydrate Saturation, Estimated Resources
'''
well_info, sensor_data_list = load_and_preprocess()

# Remove the data point with NaN value
well_info = well_info.dropna()
for sensor_data in sensor_data_list:
    sensor_data.dropna(inplace=True)
```

3 Visualization the Distribution

3.0.1 3D Plot

[9]:
```
# Determine the global min and max values for 'Estimated Resources'
global_min = min(df['Estimated Resources'].min() for df in sensor_data_list)
global_max = max(df['Estimated Resources'].max() for df in sensor_data_list)

# Create a 3D scatter plot
fig = go.Figure()

for idx, df in enumerate(sensor_data_list):
    well_name = well_info.iloc[idx]['Well']
    x_coord = well_info.iloc[idx]['X']
```

```
    y_coord = well_info.iloc[idx]['Y']

    fig.add_trace(go.Scatter3d(
        x=[x_coord] * len(df),  # Repeat the X coordinate for each depth
        y=[y_coord] * len(df),  # Repeat the Y coordinate for each depth
        z=df['Depth'],  # Invert the depth values
        mode='markers',
        marker=dict(
            size=3,
            color=df['Estimated Resources'],  # Color points by estimated␣
 ↪resources
            colorscale='Viridis',  # Set the colorscale
            cmin=global_min,  # Set global minimum
            cmax=global_max,  # Set global maximum
            colorbar=dict(
                title='Estimated Resources',  # Title of the color bar
                titleside='right',
                titlefont=dict(size=12),
                thickness=20,
                x=0.85  # Adjust the position of the color bar (if necessary)
            ),
            opacity=0.8
        ),
        name=f'Well {well_name}'
    ))

# Update plot layout
fig.update_layout(
    title='3D Plot of Resource Distribution by Depth',
    height=600,
    width=900,
    scene=dict(
        xaxis_title='X Coordinate',
        yaxis_title='Y Coordinate',
        zaxis=dict(
            title='Depth',
            autorange='reversed'  # Automatically reverse the z-axis
        )
    ),
)

# Show only one color bar
# We ensure only the color bar of the last trace is visible
for trace in fig.data[:-1]:
    trace.marker.showscale = False

fig.show()
```

3.0.2 2D Plot in Depth

[10]:
```
from scipy.interpolate import interp1d


def align_index_step_size(df, new_depth_step=0.1):

    # Create the new depth grid starting from the base depth
    min_depth = np.around(min(df['Depth']), decimals=1)
    max_depth = np.around(max(df['Depth']), decimals=1)
    new_depths = np.around(np.arange(min_depth, max_depth, new_depth_step),␣
 ↪decimals=1)

    # Interpolate other columns
    interpolated_data = {}
    for column in df.columns:
        if column != 'Depth':
            # Create interpolation function
            f = interp1d(df['Depth'], df[column], bounds_error=False,␣
 ↪fill_value="extrapolate")
            # Interpolate data
            interpolated_data[column] = f(new_depths)

    # Create new DataFrame with interpolated data and new depth grid
    interpolated_df = pd.DataFrame(interpolated_data, index=new_depths)
    interpolated_df.reset_index(inplace=True)
    interpolated_df.rename(columns={'index': 'Depth'}, inplace=True)
    return interpolated_df
```

[11]:
```
from plotly.subplots import make_subplots

# Determine the global minimum and maximum depths across all wells for uniform␣
 ↪y-axis
min_depth = min(df['Depth'].min() for df in sensor_data_list)
max_depth = max(df['Depth'].max() for df in sensor_data_list)

# Create a uniform depth grid
uniform_depths = np.around(np.arange(min_depth, max_depth + 0.1, 0.1),␣
 ↪decimals=1)  # Adjust step size if necessary

# Update each dataset to include the uniform depth grid
uniform_datasets = []
for df in sensor_data_list:
    df = align_index_step_size(df)
    # Set Depth as index and reindex with the uniform depths, filling missing␣
 ↪values with 0
```

```
    uniform_df = df.set_index('Depth').reindex(uniform_depths, fill_value=0).
 ↪reset_index()
    uniform_datasets.append(uniform_df)

# Create subplots: one for each well
num_wells = len(uniform_datasets)
fig = make_subplots(rows=1, cols=num_wells, subplot_titles=[f"Well {well}" for␣
 ↪well in well_info['Well']])

# Add a line plot for each well
for idx, df in enumerate(uniform_datasets):
    well_name = well_info.iloc[idx]['Well']
    fig.add_trace(
        go.Scatter(
            x=df['Estimated Resources'],
            y=df['Depth'],
            mode='lines+markers',
            name=f'Well {well_name}',
            marker=dict(size=2),
            line=dict(width=1)
        ),
        row=1, col=idx+1
    )

# Update y-axis to be reversed and uniform
# Only show y-axis on the first and last subplot
for i in range(1, num_wells + 1):
    if i == 1:  # First subplot
        fig.update_yaxes(title_text="Depth (m)", autorange="reversed",␣
 ↪range=[min_depth, max_depth],
                         row=1, col=i)
    elif i == num_wells:  # Last subplot
        fig.update_yaxes(autorange="reversed", range=[min_depth, max_depth],
                         row=1, col=i, showticklabels=True,  side='right')
    else:
        fig.update_yaxes(autorange="reversed", range=[min_depth, max_depth],
                         row=1, col=i, showticklabels=False)


# Update y-axis to be reversed and uniform
fig.update_yaxes(autorange="reversed", range=[min_depth, max_depth])

# Update layout to better fit the subplots
fig.update_layout(
    height=600,
    width=1550,
    title_text="Resource Distribution in Depth for Each Sensor Tower",
```

```
    showlegend=False
)

fig.show()
```

3.0.3 2D Plot in Location

[12]:
```
import plotly.express as px

# Plotting using Plotly
fig = px.scatter(well_info, x='X', y='Y', size='Total Resources', color='Total␣
 ↪Resources',
                 hover_name='Well', size_max=40,
                 title='Resource Distribution Across Wells',
                 labels={'X': 'X Coordinate', 'Y': 'Y Coordinate', 'Total␣
 ↪Resources': 'Estimated Resources'})

fig.update_layout(
    height=600,
    width=700,
)

fig.show()
```