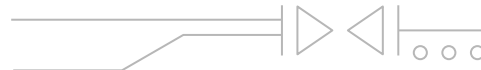




Advanced Programming For Business Analytics

Miguel Torrealba Schwarz, Julio Berrocal Alvarez, Nicolas
Gonzalez Aguirre, Abdur Rahman





Introduction

- Mission: To consistently produce top quality, premium valued bakery products with the taste of hometown goodness in every bite.
- Muffin Town is a leading bakery based in Chelsea, Massachusetts, known for its exceptional baked goods.
- Established in 1978, Muffin Town has grown from a small bakery into a renowned brand recognized for its high-quality products.
- Muffin Town offers a diverse range of baked goods, including muffins, cakes, pastries, and other delicious treats, catering to various tastes and preferences.
- With a nationwide presence, Muffin Town serves both retail consumers and the foodservice industry, providing its products to a wide range of customers across the country.





Project Background

- Visited Muffin Town's plant in Lawrence to gain insights into their operations and meet with Brian, the key contact.
- Engaged in discussions with Brian to understand his needs and objectives for the project.
- Brian provided Excel files containing data on plant performance for each SKU (Stock Keeping Unit).
- Conducted data cleaning and analysis to prepare the data for further insights.
- Established biweekly meetings with Brian to update him on project progress and gather feedback.
- Developed an Excel template covering the previously shared files and utilized the Gradio library in Python to create a platform for Brian to obtain quick insights on the performance of different plants.



Input

Muffin Town used several excel sheets to manage the information with no standards on

- Formats
- Units of measurement
- Protocol to entry
- Single source of information

This led to data that did not match with the reality of the business, neither led to valuable insights

To begin the process of improvement, several conventions were implemented in their process of managing inputs

- Created a single source of information after mapping the process and creating a schema
- Created a Master Data for all SKUs, Production Lines, and capacity
- Created tables to organize information and fed through the primary key
- Validated data through conditional formatting
- Minimized error and manual input with drop down lists



Excel Template

- The excel file standard sets the right path for the codes proper run
- Uses color coding to define data type and inputs
- Show flexibility for skus increase and other production parameters

Equipment ID	Group Name	Date	Current Jo	Start of Batch Date/Ti	End of Batch Date/Ti	Batch Cou	User1	LINE -LOCATION (Eq. Co	Optimal Cases Per H	Lower Tar	Upper Tar
20	LAWRENCE AUTOBAKE	2020-08-24	96,605	24-Aug-20	8/24/2020 19:42:41	2,445	96,605	EQP-LAWPACK1	347	312	382
20	LAWRENCE AUTOBAKE	2020-08-24	24,970	24-Aug-20	8/24/2020 20:00:05	3	24,970	EQP-LAWPACK1	364	328	400
20	LAWRENCE AUTOBAKE	2020-08-24	7,910	24-Aug-20	8/24/2020 20:19:10	49	7,910	EQP-LAWPACK1	349	314	384
20	LAWRENCE AUTOBAKE	2020-08-25	24,970	24-Aug-20	8/25/2020 06:54:17	504	24,970	EQP-LAWPACK1	364	328	400
20	LAWRENCE AUTOBAKE	2020-08-25	27,805	25-Aug-20	8/25/2020 15:17:31	2,087	27,805	EQP-LAWPACK1	260	234	286
20	LAWRENCE AUTOBAKE	2020-08-26	27,405	25-Aug-20	8/26/2020 06:56:42	2,123	27,405	EQP-LAWPACK1	260	234	286
20	LAWRENCE AUTOBAKE	2020-08-26	2,666	26-Aug-20	8/26/2020 12:59:08	1,272	2,666	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-08-26	2,661	26-Aug-20	8/26/2020 14:06:39	158	2,661	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-08-26	88,888	26-Aug-20	8/26/2020 19:38:30	0	88,888	EQP-LAWPACK1			
20	LAWRENCE AUTOBAKE	2020-08-27	2,661	26-Aug-20	8/27/2020 06:54:47	296	2,661	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-08-31	99,999	28-Aug-20	8/31/2020 06:55:54	0	99,999	EQP-LAWPACK1			
20	LAWRENCE AUTOBAKE	2020-08-31	2,991	31-Aug-20	8/31/2020 09:03:04	380	2,991	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-08-31	2,946	31-Aug-20	8/31/2020 11:05:40	406	2,946	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-08-31	23,905	31-Aug-20	8/31/2020 17:23:46	2,130	23,905	EQP-LAWPACK1	364	328	400
20	LAWRENCE AUTOBAKE	2020-08-31	24,975	31-Aug-20	8/31/2020 18:45:00	371	24,975	EQP-LAWPACK1	364	328	400
20	LAWRENCE AUTOBAKE	2020-09-01	6,670	31-Aug-20	9/1/2020 06:57:36	1,482	6,670	EQP-LAWPACK1	352	317	387
20	LAWRENCE AUTOBAKE	2020-09-01	2,666	01-Sep-20	9/1/2020 14:10:18	770	2,666	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-09-02	2,661	01-Sep-20	9/2/2020 06:57:13	1,670	2,661	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-09-02	24,961	02-Sep-20	9/2/2020 07:35:52	11	24,961	EQP-LAWPACK1	364	328	400
20	LAWRENCE AUTOBAKE	2020-09-02	6,661	02-Sep-20	9/2/2020 10:59:48	1,130	6,661	EQP-LAWPACK1	352	317	387
20	LAWRENCE AUTOBAKE	2020-09-01	2,675	01-Sep-20	9/1/2020 10:34:53	760	2,675	EQP-LAWPACK1	217	195	239
20	LAWRENCE AUTOBAKE	2020-09-02	24,961	02-Sep-20	9/2/2020 15:16:00	1,361	24,961	EQP-LAWPACK1	364	328	400

Gradio Layout

Spaces

BrianA1974/muffintown

like 0

Running

App

Files

Community

Upload Sheet

Search

Upload Template File

Drop File Here
- or -
Click to Upload

Clear

Submit


Total Observations


Compliant Observations %


Non-Compliant Observations %





Gradio Layout


 Spaces


 BrianA1974 / muffintown


 like 0


 Running

 App

 Files

 Community






Upload Sheet


Search

SKU	Group Name	Date(yyyy-mm-dd)
<input type="text"/>	<input type="text"/>	<input type="text"/>

Description

Records	Compliance %	Avg Units/Hr	Std Dev Units/Hr
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

 Units Per Hour Distribution



Submit

Gradio Layout

```
with gr.Blocks() as tab_search_by_sku:
    with gr.Row():
        sku = gr.Textbox(label="SKU")
        group_name = gr.Textbox(label="Group Name")
        date = gr.Textbox(label="Date(yyyy-mm-dd)")

    with gr.Row():
        description = gr.Textbox(label="Description")

    with gr.Row():
        records = gr.Textbox(label="Records")
        compliance = gr.Textbox(label="Compliance %")
        avgunits = gr.Textbox(label="Avg Units/Hr")
        stddev = gr.Textbox(label="Std Dev Units/Hr")

    with gr.Row():
        graph = gr.Plot(label="Units Per Hour Distribution")

    with gr.Row():
        submit_btn = gr.Button('Submit')
        submit_btn.click(
            fn=search_data,
            inputs=[sku, group_name, date],
            outputs=[description, records, compliance, avgunits, stddev, graph]
        )
```

```
tab_files = gr.Interface(
    fn = upload_file,
    inputs = [
        gr.File(label="Upload Template File", file_types=[".xlsx"], file_count="single")
    ],
    outputs = [
        gr.Textbox(label="Total Observations", interactive=False),
        gr.Textbox(label="Compliant Observations %", interactive=False),
        gr.Textbox(label="Non-Compliant Observations %", interactive=False)
    ],
    allow_flagging = "never"
)

demo = gr.TabbedInterface([tab_files, tab_search_by_sku], ["Upload Sheet", "Search"])

demo.launch(auth=("BrianA", "Brian@Muffintown1"))
```


Function prepare_df

```
def prepare_df(file_path):  
  
    df = pd.read_excel(file_path)  
  
    df['Current Job'] = df['Current Job'].astype(str)  
  
    df['Start of Batch Date/Time'] = pd.to_datetime(df['Start of Batch Date/Time'])  
    df['End of Batch Date/Time'] = pd.to_datetime(df['End of Batch Date/Time'])  
  
    df['Batch Length (Hours)'] = (df['End of Batch Date/Time'] - df['Start of Batch Date/Time']).dt.total_seconds() / 3600  
  
    df['Units Per Hour'] = df['Batch Count'] / df['Batch Length (Hours)']  
  
    df.dropna(subset=['Optimal Cases Per Hour'], inplace=True)  
  
    return df
```

Function initial_analysis

```
def initial_analysis(df):  
  
    compliant_df = df[(df['Units Per Hour'] >= df['Lower Target']) & (df['Units Per Hour'] <= df['Upper Target'])]  
    non_compliant_df = df[(df['Units Per Hour'] < df['Lower Target']) | (df['Units Per Hour'] > df['Upper Target'])]  
  
    total_observations = len(df)  
    pct_compliant = 100 * round(len(compliant_df)/total_observations,2)  
    pct_non_compliant = 100 * round(len(non_compliant_df)/total_observations,2)  
  
    return total_observations, pct_compliant, pct_non_compliant
```

Function search_data

```
def search_data(sku, group_name, date):

    if(sku == '' and group_name == '' and date != ''):
        filtered_df = initial_df[initial_df['Date'] == date]

    elif(sku == '' and group_name != '' and date == ''):
        filtered_df = initial_df[initial_df['Group Name'] == group_name]

    elif(sku != '' and group_name == '' and date == ''):
        filtered_df = initial_df[initial_df['Current Job'] == sku]

    elif(sku == '' and group_name != '' and date != ''):
        filtered_df = initial_df[(initial_df['Group Name'] == group_name) & (initial_df['Date'] == date)]

    elif(sku != '' and group_name == '' and date != ''):
        filtered_df = initial_df[(initial_df['Current Job'] == sku) & (initial_df['Date'] == date)]

    elif(sku != '' and group_name != '' and date == ''):
        filtered_df = initial_df[(initial_df['Current Job'] == sku) & (initial_df['Group Name'] == group_name)]

    else:
        filtered_df = initial_df[(initial_df['Current Job'] == sku) & (initial_df['Group Name'] == group_name) & (initial_df['Date'] == date)]

    records = len(filtered_df)

    compliant_records = len(filtered_df[(filtered_df['Units Per Hour'] >= filtered_df['Lower Target']) & (filtered_df['Units Per Hour'] <= filtered_df['Upper Target'])])

    compliance_percentage = round((compliant_records / records),2) * 100 if records != 0 else 0

    avg_units_per_hour = round(filtered_df['Units Per Hour'].mean(),2)

    stdev_units_per_hour = round(filtered_df['Units Per Hour'].std(),2)

    most_common_value = filtered_df['Description'].value_counts().idxmax()

    fig1 = plt.figure()
    plt.hist(filtered_df['Units Per Hour'], bins=20, color='skyblue', edgecolor='black')
    plt.xlabel('Units Per Hour')
    plt.ylabel('Frequency')
    plt.grid(True)

    return most_common_value, records, compliance_percentage, avg_units_per_hour, stdev_units_per_hour, fig1
```

Application Demo



<https://huggingface.co/spaces/BrianA1974/muffintown>

Thanks!

Do you have any questions?

CREDITS: This presentation template was created by [Slidesgo](#), and includes icons by [Flaticon](#), and infographics & images by [Freepik](#)

