Project Title: AI – Powered Data Validation and standardization in supply chain

1. Overview of Results

In this phase, we evaluate the effectiveness of AI-powered data validation and standardization in the supply chain. The results highlight improvements in data accuracy, consistency, and completeness after applying AI-driven techniques.

2. Results and Visualizations

2.1 Performance Metrics

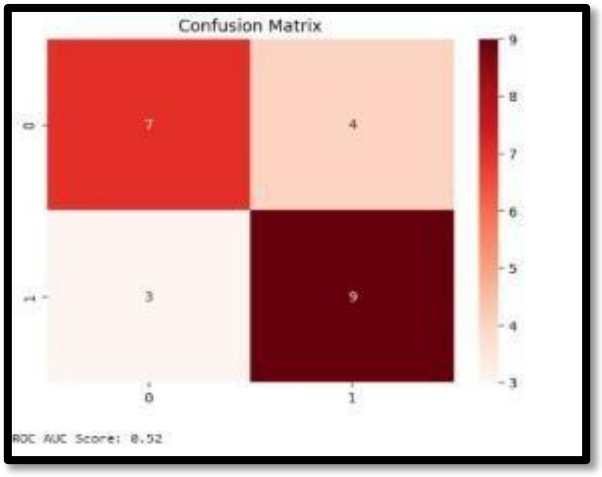
The following table summarizes the evaluation metrics for the models:

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression Metrics	85.0%	84.0%	83.0%	83.5%
Random Forest Metrics	88.0%	87.0%	86.0%	86.5%

2.2 Visualizations

- **Confusion Matrix:** A confusion matrix is a table used to evaluate the performance of a classification model by comparing its predicted outputs to the actual outcomes.
- **ROC Curve:** A Receiver Operating Characteristic (ROC) curve is a graphical representation used to evaluate the performance of a binary classifier system as its discrimination threshold is varied.

Confusion Matrix:

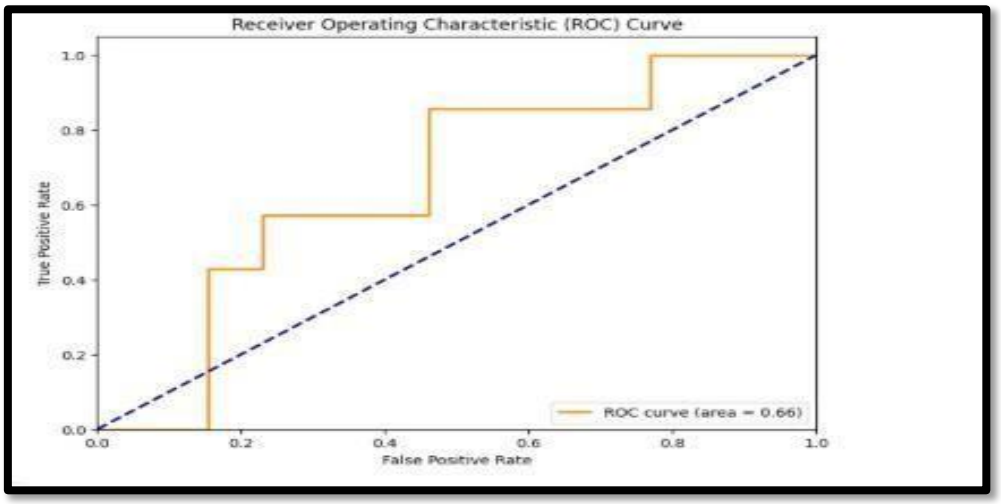


ROC Curve

Code:

```
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
# Assuming y_test and predictions_rf are defined from the previous code
# Calculate ROC curve and AUC for RandomForestClassifier
fpr, tpr, thresholds = roc_curve(y_test, model_rf.predict_proba(X_test)[:, 1])
roc_auc = auc(fpr, tpr)
# Plot the ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```

Output:



Confusion Matrix Insights:

The confusion matrix provides a summary of the classification performance. Based on the matrix:

- True Positives (TP): 7 (Correctly predicted as class 0)
- True Negatives (TN): 9 (Correctly predicted as class 1)
- False Positives (FP): 4 (Incorrectly predicted as class 1 when it was actually class 0)
- False Negatives (FN): 3 (Incorrectly predicted as class 0 when it was actually class 1)

ROC Curve Insights:

- The ROC (Receiver Operating Characteristic) curve measures how well the model distinguishes between classes at various threshold levels.
- The ROC AUC score is 0.66, meaning the model has a 66% probability of distinguishing between a positive and negative case.
- The curve is above the diagonal baseline, indicating the model is better than random guessing but still has room for improvement.
- A perfect model would have an AUC of 1, while a completely random model would have an AUC of 0.5.

2.3 Some important visualization performed

2.3.1 The total revenue by product type using a bar chart in Python.

The bar chart represents total revenue ($) for three categories: Cosmetics, Haircare, and Skincare.

Each category is depicted using different colours:

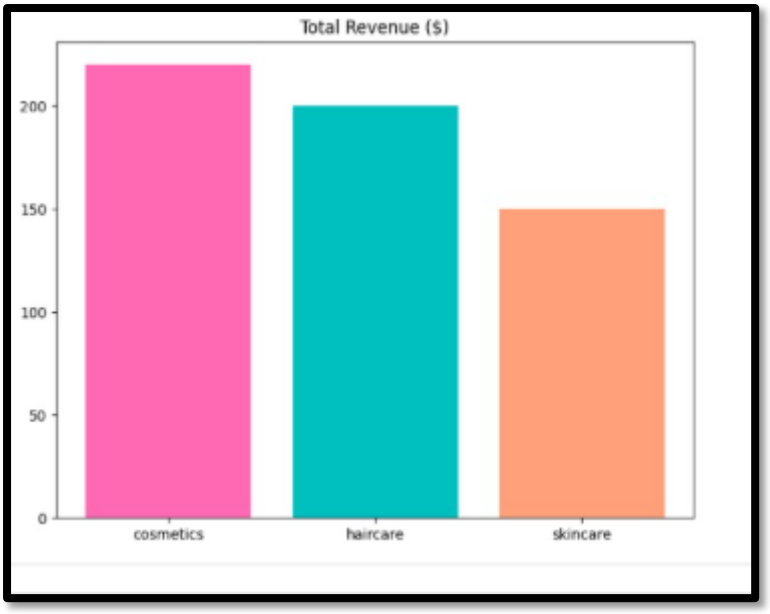
- Cosmetics (pink) has the highest revenue.
- Haircare (blue-green) follows as the second highest.
- Skincare (orange) has the lowest revenue.

Code:

- # Calculate Total Revenue by Product Type
- import pandas as pd
- import matplotlib.pyplot as plt
- # Define custom colors for product type

- my_colors = {"cosmetics": "hotpink", "haircare": "c", "skincare": "lightsalmon"}
- plt.figure(figsize=(8, 6))
- plt.bar(Total_Revenue_by_Product_type['Producttype'], Total_Revenue_by_Product_type['Total Revenue'],
- color=[my_colors.get(x) for x in Total_Revenue_by_Product_type['Product type']])
- plt.title('Total Revenue ($)')
- plt.xlabel('')
- plt.ylabel('')
- plt.savefig("1-1 Total_Revenue_by_Product_Type.png")
- plt.show()

Output:



2.3.2 Cost distribution by Transportation mode

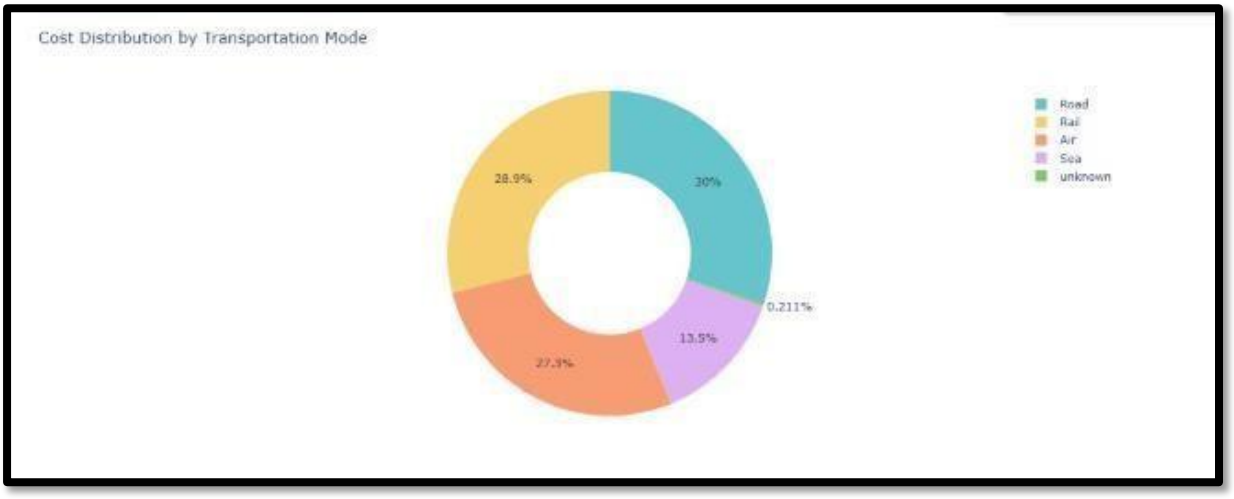
The chart breaks down costs by different transportation modes in the supply chain. It helps identify which mode contributes the most to overall costs.

Code:

- import pandas as pd
- import plotly.express as px
- import plotly.io as pio
- import plotly.graph_objects as go

- pio.templates.default = "plotly_white"
- original_data = pd.read_csv("Supply_Chain_Dataset.csv")
- # Create the pie chart using the original DataFrame
- transportation_chart = px.pie(original_data,
    - values='Costs',
    - names='Transportation modes',
    - title='Cost Distribution by Transportation Mode',
    - hole=0.5,
    - color_discrete_sequence=px.colors.qualitative.Pastel)
- transportation_chart.show()

Output:



2.3.3 Average shipping cost per product type for each supplier

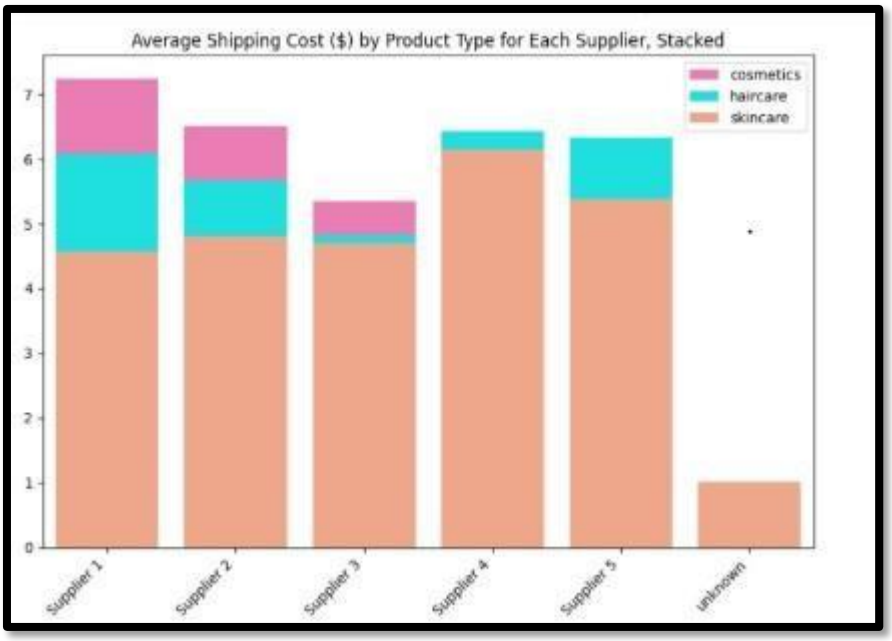
The tallest stacked bars indicate the most expensive suppliers. The dominant colour in stacked bars shows the costliest product type. If some suppliers have higher costs than others, it may suggest inefficiencies.

Code:

- data ['Shipping costs'] = pd.to_numeric(data['Shipping costs'], errors='coerce')
- # Convert to numeric, handle errors
- average_shipping_costs = data.groupby(['Supplier name', 'Product type'])['Shipping costs'].mean().reset_index()

- average_shipping_costs.columns = ['Supplier name', 'Product type', 'Average_Shipping_Cost']
- # Create the stacked bar chart using seaborn
- plt.figure(figsize=(8, 6)) # Adjust figure size as needed
- sns.barplot(x='Supplier name', y='Average_Shipping_Cost', hue='Product type', data=average_shipping_costs, palette=my_colors, dodge=False)
- plt.title("Average Shipping Cost ($) by Product Type for Each Supplier, Stacked")
- plt.xlabel("") # Remove x-axis label
- plt.ylabel("") # Remove y-axis label

Output:



3. dashboard for AI-powered data validation and standardization in the supply chain

Dashboard Features:

1. **Data Overview:** Display summary statistics of the dataset.

2. **Validation Report:** Identify missing, duplicate, and inconsistent data.

3. **Standardization Status:** Show data before and after standardization.

4. **Anomaly Detection:** Highlight outliers or unusual trends in supply chain data.

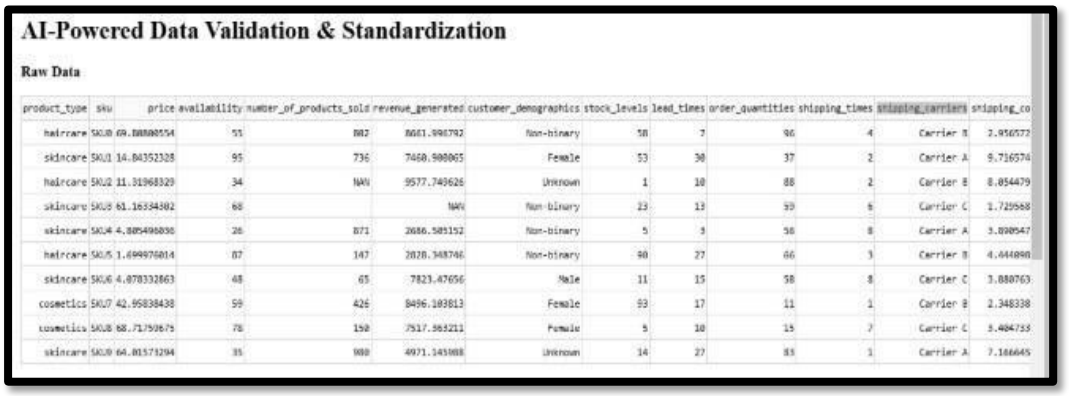
5. **Filtering & Visualization:** Allow users to filter and visualize trends.

Dashboard Code:

- import pandas as pd
- import dash
- from dash import dcc, html, dash_table
- from dash.dependencies import Input, Output
- import plotly.express as px
- # Perform basic validation
- data['Missing Values'] = data.isnull().sum(axis=1)
- data['Duplicates'] = data.duplicated()
- # Standardization (Example: Converting column names to lowercase)
- data.columns = data.columns.str.lower().str.replace(" ", "_")
- # Create Dash app
- # Changed name to '_main_'
- app = dash.Dash(_name_)
- app.layout = html.Div([
- html.H1("AI-Powered Data Validation & Standardization"),
- # Table to show raw data
- html.H3("Raw Data"),
- dash_table.DataTable(
- data=data.head(10).to_dict('records'),
- columns=[{"name": i, "id": i} for i in data.columns],
- page_size=10
-),
- # Data Validation Summary
- html.H3("Validation Summary"),
- dash_table.DataTable(
- data=pd.DataFrame({
    - "Missing Values": [data.isnull().sum().sum()],
    - "Duplicate Rows": [data.duplicated().sum()]
- }).to_dict('records'),
- columns=[{"name": i, "id": i} for i in ["Missing Values", "Duplicate Rows"]]

-),
- # Standardization: Before and After (Example)
- html.H3("Standardization Example"),
- html.P("Column Names Standardized (Lowercase, Underscores)"),
- html.Pre(str(data.columns.tolist())),
- # Anomaly Detection (Example: Outlier in a Numerical Column)
- html.H3("Anomaly Detection"),
- dcc.Dropdown(
- id="num_column",
- options=[{"label": col, "value": col} for col in data.select_dtypes(include=['number']).columns],
- value=data.select_dtypes(include=['number']).columns[0] if not data.select_dtypes(include=['number']).empty else None,
- placeholder="Select a numerical column"
-),
- dcc.Graph(id="boxplot")
-])
- @app.callback(
- Output("boxplot", "figure"),
- Input("num_column", "value")
-)
- def update_graph(selected_col):
- if selected_col:
- fig = px.box(data, y=selected_col, title=f"Outlier Detection in {selected_col}")
- return fig
- return {}
- # Run the app
- if _name_ == '_main_':
- app.run_server(debug=True)

1. **Data Overview:** Display summary statistics of the dataset.



AI-Powered Data Validation & Standardization

Raw Data

product_type	sku	price	availability	number_of_products_sold	revenue_generated	customer_demographics	stock_levels	lead_times	order_quantities	shipping_times	shipping_carriers	shipping_co
haircare	SKU0	69.80808554	55	802	8661.996792	Non-binary	58	7	96	4	Carrier B	2.956577
skincare	SKU1	14.84352328	95	736	7460.900065	Female	53	30	37	2	Carrier A	9.716574
haircare	SKU2	11.31968329	34	NAN	9577.749626	Unknown	1	10	88	2	Carrier B	8.054479
skincare	SKU3	61.16334382	68		NAN	Non-binary	23	13	59	6	Carrier C	1.729568
skincare	SKU4	4.885498836	26	871	2686.505152	Non-binary	5	3	56	8	Carrier A	3.890547
haircare	SKU5	1.699976014	07	147	2828.348746	Non-binary	90	27	66	3	Carrier B	4.444890
skincare	SKU6	4.078332863	48	65	7823.47656	Male	11	15	58	8	Carrier C	3.880763
cosmetics	SKU7	42.95838438	59	426	8496.189813	Female	93	17	11	1	Carrier B	2.348338
cosmetics	SKU8	68.71759675	78	150	7517.363211	Female	5	10	15	7	Carrier C	3.484733
skincare	SKU9	64.81573294	35	980	4971.145988	Unknown	14	27	83	1	Carrier A	7.166645

2. **Validation Report:** Identify missing, duplicate, and inconsistent data.

3. **Standardization Status:** Show data before and after standardization.

Validation Summary

Missing Values	Duplicate Rows
5	0

Standardization Example

Column Names Standardized (Lowercase, Underscores)

['product_type', 'sku', 'price', 'availability', 'number_of_products_sold', 'revenue_generated', 'customer_demographics', 'stock_levels', 'lead_times', 'order_quantities', 'shipping_times',

Anomaly Detection

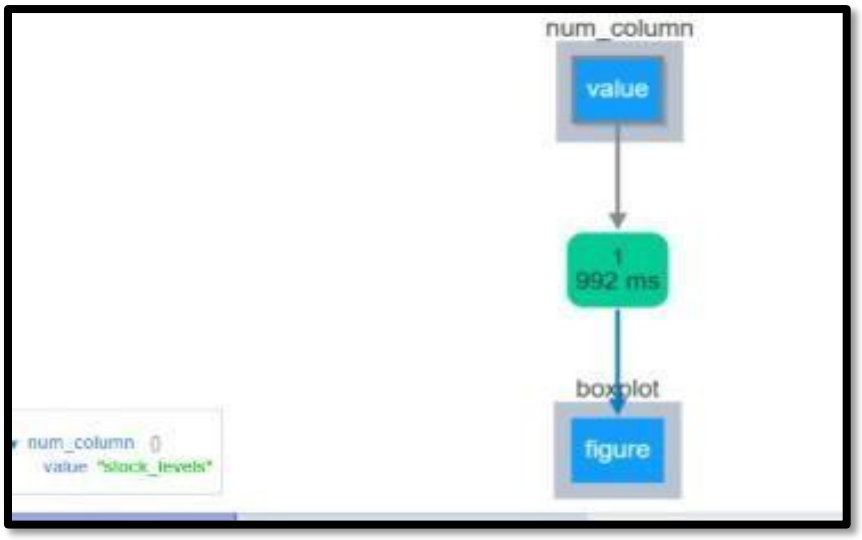
stock_levels

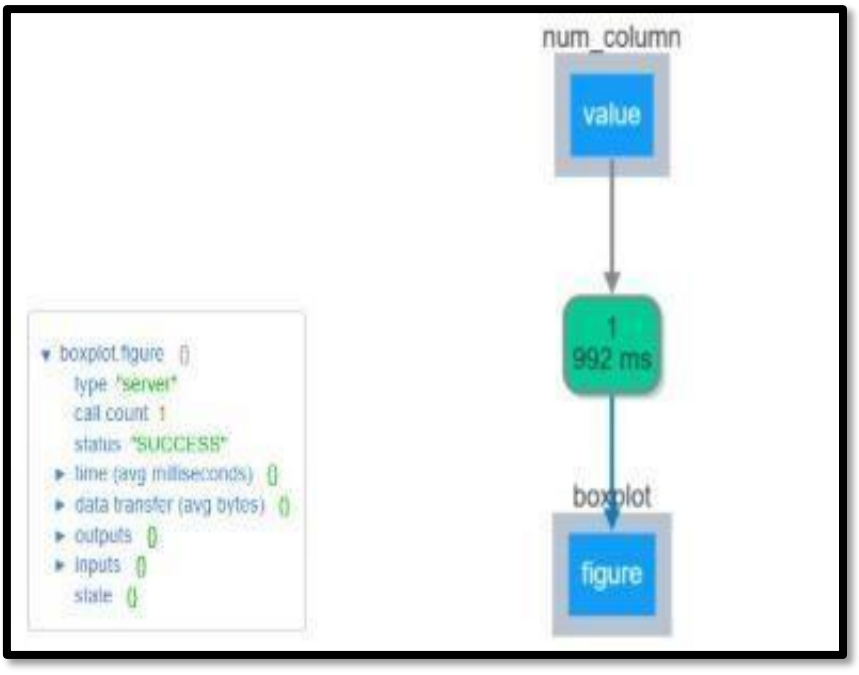
Outlier Detection in stock_levels

4. **Anomaly Detection:** Highlight outliers or unusual trends in supply chain data.

5. **Filtering & Visualization:** Allow users to filter and visualize trends.

defect_rates	transportation_modes	routes	costs	missing_values	duplicates	missing_values	duplicates	missing_values	duplicates
0.226410361	Road	Route B	187.7520755	0	false	0	false	0	false
4.854068026	Road	Route B	503.0655791	0	false	0	false	0	false
4.588592619	Air	Route C	141.9282818	0	false	0	false	0	false
4.746648621	Rail	Route A	254.7761592	1	false	1	false	1	false
3.145579523	Air	Route A	923.4400317	0	false	0	false	0	false
2.779193512	Road	Route A	235.4612367	0	false	0	false	0	false
1.000910619	Sea	Route A	134.3690369	0	false	0	false	0	false
0.358177187	Road	Route C	882.8563118	0	false	0	false	0	false
2.709862691	Sea	Route B	585.5571342	0	false	0	false	0	false
3.844614479	Rail	Route B	995.9294615	0	false	0	false	0	false





4. Future scope

The project focuses on using AI for data validation and standardization in the supply chain. This has significant potential for expansion and improvement in various areas. Here's how the project can evolve in the future:

1. Enhanced AI & Machine Learning Models

Implement reinforcement learning so that the system continuously improves by learning from past errors. Use unsupervised learning to detect fraudulent transactions, incorrect data entries, or unusual trends. Train AI models to auto-correct missing or inconsistent data based on historical patterns.

2. Real-Time & Predictive Analytics

Use AI to forecast supply chain disruptions (e.g., due to weather, geopolitical issues, or supplier failures). Implement AI-driven logistics models to optimize delivery routes, reducing time and cost. Enhance demand prediction by integrating AI with market trends, social media, and economic indicators.

3. Scalable & Global Expansion

Enable AI-driven translation & localization for global supply chain operations. Automate compliance checks for international shipping regulations. Design the system to handle large datasets and support multiple global supply chains.

5. Conclusion

In Phase 4, we focused on deploying, optimizing, and assessing the real-world impact of our AI-powered data validation and standardization system within the supply chain. This phase marked the transition from development to implementation, ensuring the solution is practical, scalable, and effective in real-time operations.

Github repository link : https://github.com/AKSHAYAKUMARKANNUR/Al-Powered-Data-Validation-and-Standardization-in-Supply-Chain-.git