Final Report

Technical Support Analysis

Problem Statement

The goal of this technical support analysis project is to develop a machine learning model that can accurately predict a result of a customer satisfaction surveys and defining main features affecting customer satisfaction. Additionally, analysis will help in justifying business decisions in regards to improving customer satisfaction

Data Wrangling

The Technical Support Dataset is a real-life dataset that reflects interactions between customers and technical support group by utilizing ticketing system.

```
Index: 1173 entries, 0 to 2322
Data columns (total 47 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   Status                               1173 non-null   object
 1   Ticket ID                            1173 non-null   int64
 2   Priority                             1173 non-null   object
 3   Source                               1173 non-null   object
 4   Topic                                1173 non-null   object
 5   Agent Group                          1173 non-null   object
 6   Agent Name                           1173 non-null   object
 7   Created time                         1173 non-null   datetime64[ns]
 8   Expected SLA to resolve              1173 non-null   datetime64[ns]
 9   Expected SLA to first response       1173 non-null   datetime64[ns]
 10  First response time                  1173 non-null   object
 11  SLA For first response               1173 non-null   object
 12  Resolution time                      1173 non-null   datetime64[ns]
 13  SLA For Resolution                   1173 non-null   object
 14  Close time                           1173 non-null   datetime64[ns]
 15  Agent interactions                   1173 non-null   float64
 16  Survey results                       1173 non-null   float64
 17  Product group                        1173 non-null   object
 18  Support Level                        1173 non-null   object
 19  Country                              1173 non-null   object
 20  Latitude                             1173 non-null   float64
 21  Longitude                            1173 non-null   float64
 22  Created time Year                    1173 non-null   int32
 23  Created time Month                   1173 non-null   int32
 24  Created time Day                     1173 non-null   int32
 25  Created time Weekday                 1173 non-null   object
 26  Created time Time                    1173 non-null   object
 27  Expected SLA to resolve Year         1173 non-null   int32
 28  Expected SLA to resolve Month        1173 non-null   int32
 29  Expected SLA to resolve Day          1173 non-null   int32
 30  Expected SLA to resolve Weekday      1173 non-null   object
 31  Expected SLA to resolve Time         1173 non-null   object
 32  Expected SLA to first response Year  1173 non-null   int32
 33  Expected SLA to first response Month 1173 non-null   int32
 34  Expected SLA to first response Day   1173 non-null   int32
 35  Expected SLA to first response Weekday  1173 non-null   object
 36  Expected SLA to first response Time  1173 non-null   object
 37  Resolution time Year                 1173 non-null   int32
 38  Resolution time Month                1173 non-null   int32
 39  Resolution time Day                  1173 non-null   int32
 40  Resolution time Weekday              1173 non-null   object
 41  Resolution time Time                 1173 non-null   object
 42  Close time Year                      1173 non-null   int32
 43  Close time Month                     1173 non-null   int32
 44  Close time Day                       1173 non-null   int32
 45  Close time Weekday                   1173 non-null   object
 46  Close time Time                      1173 non-null   object
dtypes: datetime64[ns](5), float64(4), int32(15), int64(1), object(22)
memory usage: 371.1+ KB
```

	Status	Ticket ID	Priority	Source	Topic	Agent Group	Agent Name	Created time	Expected SLA to resolve	Expected SLA to first response	First response time	SLA For first response	Resolution time	SLA For Resolution	Close time
0	Closed	1012	Low	Email	Feature request	1st line support	Kristos Westoll	2023-01-02 00:58:36	2023-01-04 00:58:36	2023-01-02 01:58:36	2023-01-02 01:03:17.432	Within SLA	2023-01-04 00:31:51.694	Within SLA	2023-01-04 04:02:59.013
3	Closed	1015	Medium	Email	Pricing and licensing	1st line support	Connor Danielovitch	2023-01-03 03:09:39	2023-01-05 03:09:39	2023-01-03 04:09:39	2023-01-03 07:09:15.835	SLA Violated	2023-01-04 14:32:34.979	Within SLA	2023-01-08 04:24:54.771
4	Closed	1016	Low	Email	Product setup	1st line support	Kristos Westoll	2023-01-03 00:03:58	2023-01-05 00:03:58	2023-01-03 01:03:58	2023-01-03 00:08:01.684	Within SLA	2023-01-04 12:03:05.986	Within SLA	2023-01-06 06:05:08.637
5	Closed	1017	Low	Email	Purchasing and invoicing	1st line support	Sheela Cutten	2023-01-03 14:25:42	2023-01-05 14:25:42	2023-01-03 15:25:42	2023-01-03 14:45:14.430	Within SLA	2023-01-04 01:55:56.533	Within SLA	2023-01-10 16:41:07.865
6	Closed	1018	Low	Phone	Product setup	1st line support	Kristos Westoll	2023-01-03 15:32:02	2023-01-05 15:32:02	2023-01-03 15:34:02	2023-01-03 15:34:00.278	Within SLA	2023-01-05 03:51:37.031	Within SLA	2023-01-07 21:52:35.202

Data Cleaning:
Handling missing values by filtering or imputing.
Converting data types, such as datetime conversions.

Exploratory Data Analysis (EDA):
Using histograms and other plots for visualizing data distributions.
Identifying and treating outliers in numerical data.

Feature Engineering:
Creating new columns like 'Time to Resolve' based on existing data.
Encoding categorical variables using techniques like one-hot encoding.

Data Analysis:
Grouping data to calculate metrics (e.g., average survey results by category).
Creating correlation heatmaps for numeric relationships.

Visualization:
Generating plots such as density and bar charts for better understanding of data.

Data Preprocessing for Modeling:
Preparing datasets for machine learning by standardizing and encoding features.
Setting up pipelines for regression and classification models.

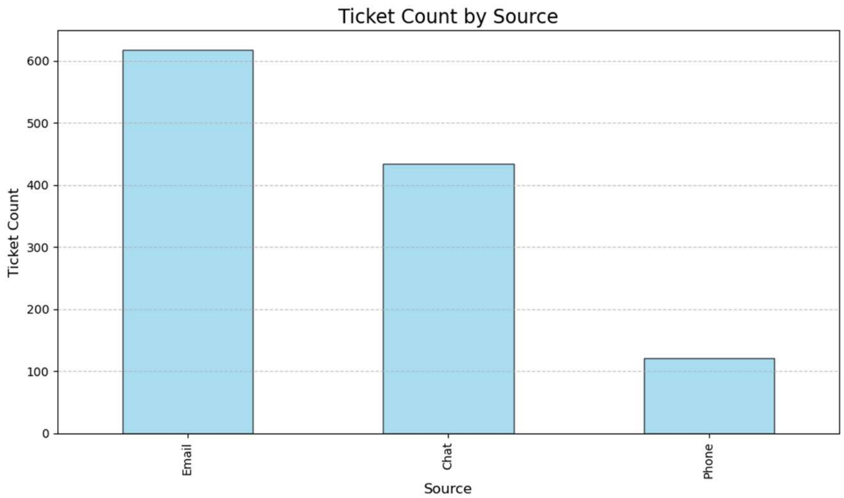
Exploratory Data Analysis

Overview of the Data:
Displaying the first few rows of the dataset using df.head() to get an initial understanding.
Checking dataset information, including column names, data types, and missing values, using df.info().

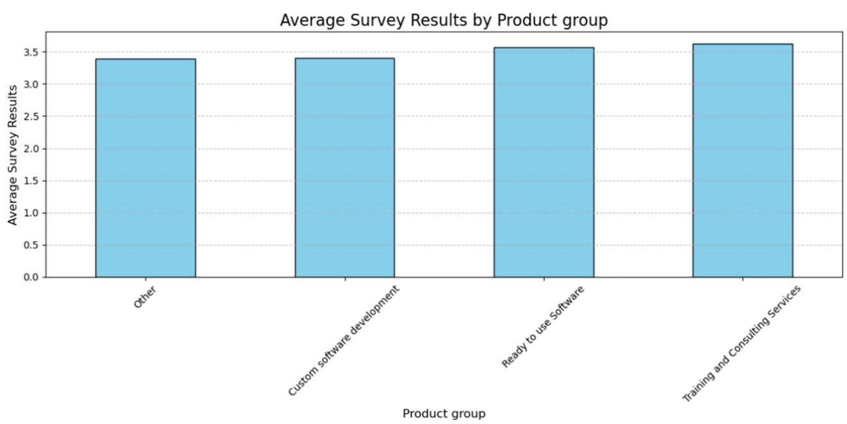
Identifying Missing Values:
Counting missing values in specific columns and filtering rows with missing data.
Visualizing the extent of missing data and deciding whether to drop or impute missing values.

Data Distributions:
Plotting histograms to visualize the distribution of numerical data.
Examples include columns like "Ticket ID," "Survey Results," and "Agent Interactions."
Identifying outliers using interquartile range (IQR) and visualizing them.

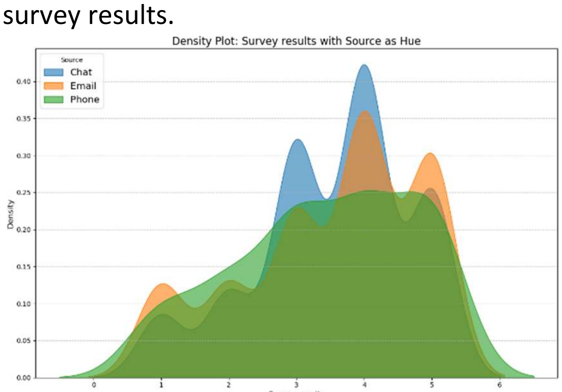
Temporal Analysis:
Extracting time-based features (Year, Month, Day, Weekday) from datetime columns like "Created Time."
Generating histograms to analyze temporal patterns in ticket creation, resolution times, and service-level agreements (SLAs).

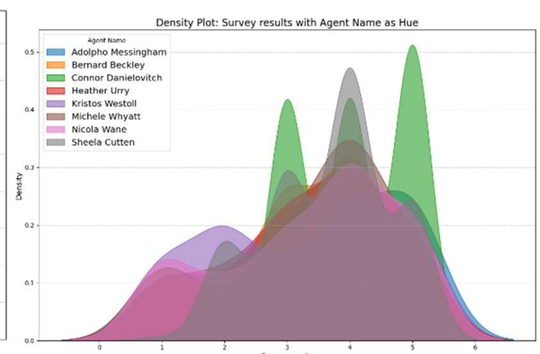


Categorical Data Analysis:
Calculating value counts for categorical columns such as "Topic" and "Agent Group."
Creating bar charts to visualize ticket counts across categories (e.g., "Priority" levels, "Sources," etc.).



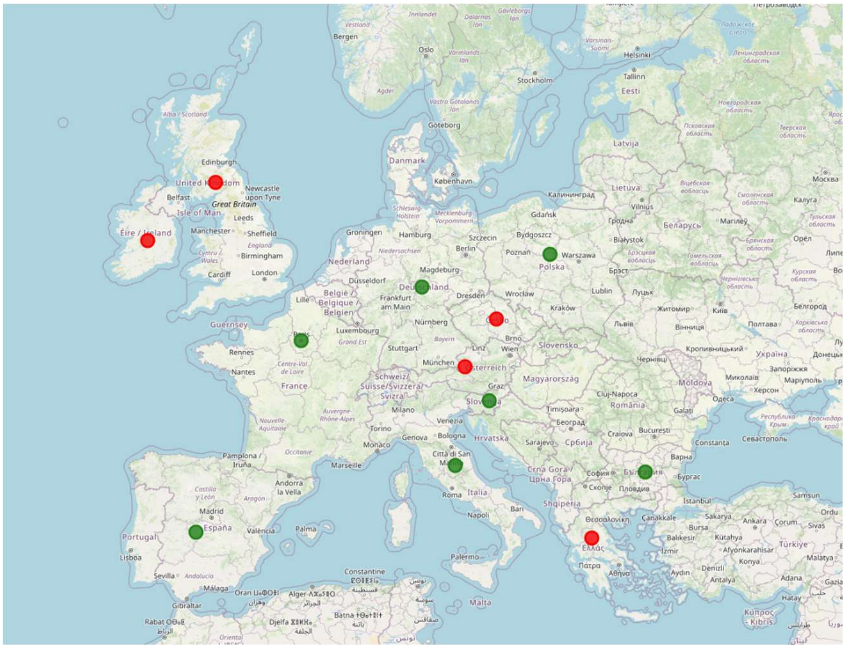
Correlations and Relationships:
Calculating and visualizing correlation heatmaps to identify relationships between numerical columns like "Survey Results," "Time to Resolve," and "Agent Interactions."
Grouping data by categories (e.g., "Country," "Topic") to compute and compare average metrics like survey results.





Geographical Analysis:

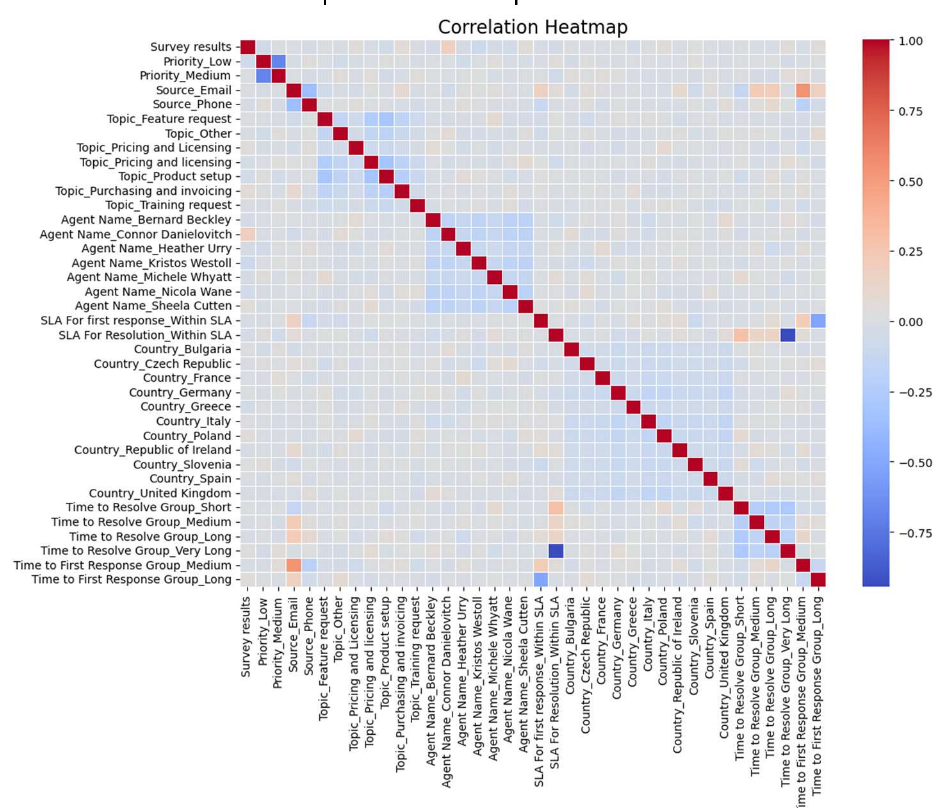
Creating maps using folium to visualize data geographically by latitude and longitude, such as survey results distribution across countries.



Advanced Visualizations:
Using density plots to visualize the relationship between numerical and categorical data (e.g., survey results by agent or source).
Grouping and visualizing data to explore patterns across topics, priorities, and resolutions.

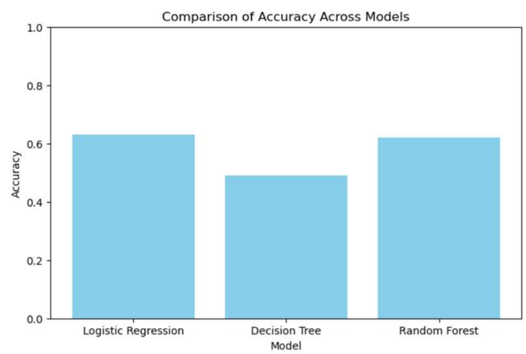
The EDA section integrates Python libraries like pandas, matplotlib, seaborn, and folium to derive insights and present the data in an intuitive and visual manner. It emphasizes understanding the dataset's structure, identifying anomalies, and uncovering patterns to inform the subsequent modeling or analysis steps.

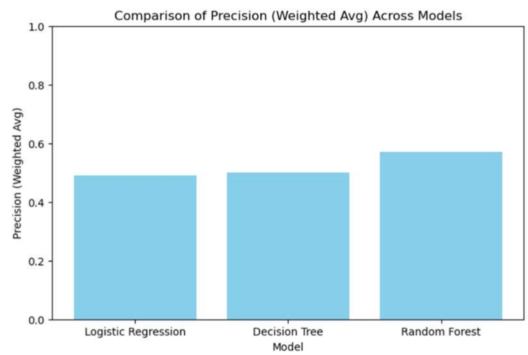
Correlation matrix heatmap to visualize dependencies between features.

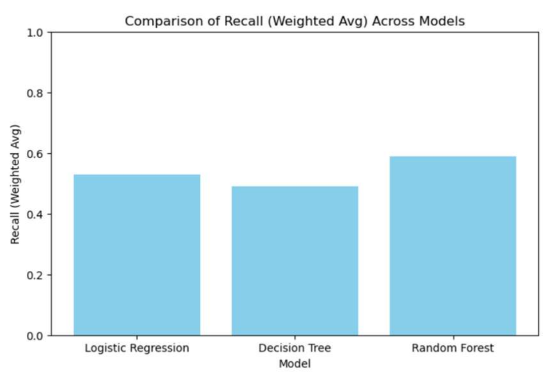


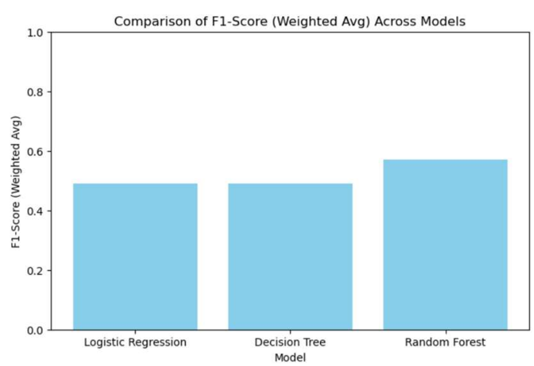
Model Selection

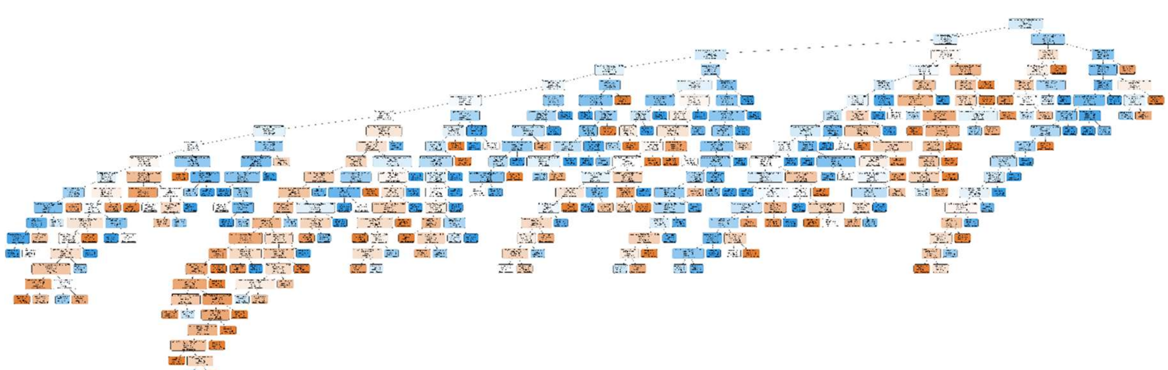
In this model selection process, cross-validation and GridSearch were employed to ensure robust evaluation and fine-tuning of each model's hyperparameters. Each model—Decision Tree, Random Forest, and Logistic Regression—was integrated into a pipeline that handled all main steps, including data preprocessing, feature scaling, and model fitting, enabling a streamlined and reproducible workflow. GridSearchCV was used to search for the best hyperparameter combinations within each model, optimizing their performance specifically for recall, the primary metric for identifying failures. The RandonForest model achieved the highest recall (0.59) on the test set with DecisionTree and Logistic Regression showing lower recall scores. This combination of pipelines, cross-validation, and GridSearch ensured a thorough and reliable model selection process, ultimately highlighting the Decision Tree as the best option for maximizing recall in predictive maintenance.











Summary of Results

Logistic Regression

- **Accuracy:** 63%
- **Precision (Weighted Avg):** 0.49
- **Recall (Weighted Avg):** 0.53
- **F1-Score (Weighted Avg):** 0.49
- **Best Parameters:** C=0.01, penalty='l2', solver='lbfgs'

Strengths**:**

- Works well with linearly separable data.
- Simpler and interpretable model.

Weaknesses**:**

- Relatively lower precision and F1-score compared to Random Forest.

Decision Tree

- **Accuracy:** 49%
- **Precision (Weighted Avg):** 0.50
- **Recall (Weighted Avg):** 0.49
- **F1-Score (Weighted Avg):** 0.49
- **Best Parameters:** criterion='entropy', max_depth=None, min_samples_split=10, min_samples_leaf=2

Strengths**:**

- Easily interpretable through decision rules.
- Handles non-linear relationships well.

Weaknesses**:**

- Prone to overfitting (mitigated here through hyperparameter tuning).
- Lower performance compared to other models.

Random Forest

- **Accuracy:** 62%
- **Precision (Weighted Avg):** 0.57
- **Recall (Weighted Avg):** 0.59
- **F1-Score (Weighted Avg):** 0.57
- **Best Parameters:** n_estimators=50, criterion='gini', min_samples_split=10, min_samples_leaf=5

Strengths**:**

- Handles non-linear relationships and high-dimensional data effectively.
- Robust to overfitting due to ensemble nature.

Weaknesses**:**

- More computationally expensive.
- Slightly complex to interpret compared to Logistic Regression.

Recommendation:

Best Model: Random Forest achieved the highest overall performance metrics, making it the most suitable choice for this classification task. In order to improve model performance, the following is recommended.

Tune Parameters Further:
Increase n_estimators to 100 or 200 to improve model stability.
Experiment with max_depth to further control overfitting.

Feature Importance Analysis:
Examine the importance of features to identify key drivers of the target variable.

Handle Class Imbalance:
Use class weighting or resampling methods to improve performance for the minority class.

Potential Improvements: Consider using more advanced ensemble methods like Gradient Boosting (e.g., XGBoost, LightGBM) or hyperparameter optimization with RandomizedSearchCV or Bayesian Optimization.