Karina Balandina
Data Processing S1

Can the late deliveries be accurately predicted based on shipment characteristics?

Project Proposal

The goal of the project is to **create a model that can accurately predict whether the shipment is going to be delayed or delivered on time**. Real world logistics data is going to be used. The chosen dataset will contain data such as order date, ship mode, and delivery status, which also can help to analyze which factors contribute most to the delays.

Since the previous projects of this course were mostly focused on regression problems (such as Linear Regression, etc.), it was decided that **classification techniques** will be used (e.g. Logistic Regression). In addition, **Seaborn** is going to be used, which is a new library to learn (it is going to show, e.g countplot/ histogram of delays by ship mode). Meanwhile, **Scikit-learn** library (also a new library) will be used for data processing.

Overall, the project can be described by this research question: “Can the late deliveries be accurately predicted based on shipment characteristics?”. The outcome will be a well-documented analysis that highlights key insights into delivery performance.

Dataset

https://www.kaggle.com/datasets/prachi13/customer-analytics

Train.csv (440.46 kB)

Detail Compact Column

About this file

Contains Cleaned 10999 observations of 12 variables.

ID ID Number of Customers.	Warehouse_block The Company have big Warehouse which is divided in to block such as A,B,C,D,E.	Mode_of_Shipment The Company Ships the products in multiple way such as Ship, Flight and Road.	Customer_care_c... The number of calls made from enquiry for enquiry of the shipment.	Customer_rating The company has rated from every customer. 1 is the lowest (Worst), 5 is the highest (Best)	Cost_of_the_Prod... Cost of the Product in US Dollars.	Prior_purchases The Number of Prior Purchase.	Product_importa... The company has categorized the product in the various parameter such as low, medium, high.	Gender Male and Female.	Discount_offered Discount offered on that specific product.	Weight_in_gms It is the weight in grams.	Reached.on.Time... It is the target variable, where 1 Indicates that the product has NOT reached on time and 0 indicates it has reached
1 11.0k	F 33% D 17% Other (5499) 50%	Ship 68% Flight 16% Other (1760) 16%	2 7	1 5	96 310	2 10	low 48% medium 43% Other (948) 9%	F 50% M 50%	1 65	1001 7846	0 1
1	D	Flight	4	2	177	3	low	F	44	1233	1
2	F	Flight	4	5	216	2	low	M	59	3088	1
3	A	Flight	2	2	183	4	low	M	48	3374	1
4	B	Flight	3	3	176	4	medium	M	10	1177	1
5	C	Flight	2	2	184	3	medium	F	46	2484	1
6	F	Flight	3	1	162	3	medium	F	12	1417	1

The dataset contains shipment records where each row represents one delivery.
It includes information such as shipment mode, weight, warehouse location, product importance, and customer data.
The target variable is **Reached.on.Time_Y.N**, where **1 means on time** and **0 means late**.

- data
  - Train.csv
- final_submission
  - final_presentation.pdf
  - final_product.ipynb
  - process_book.md
  - project_description.md
- src
  - eda.py
  - main.py
  - train.py
- .gitignore
- README.md
- requirements.txt

Project Structure

EDA (1)

```python
# Basic infobout the dataset
print("BASIC DATASET INFO")
print("Number of rows and columns:", df.shape)
print("\nColumn names:")
print(df.columns.tolist())
```

```
BASIC DATASET INFO
Number of rows and columns: (10999, 12)

Column names:
['ID', 'Warehouse_block', 'Mode_of_Shipment', 'Customer_care_calls', 'Customer_rating', 'Cost_of_the_Product', 'Pr
ior_purchases', 'Product_importance', 'Gender', 'Discount_offered', 'Weight_in_gms', 'Reached.on.Time_Y.N']
```

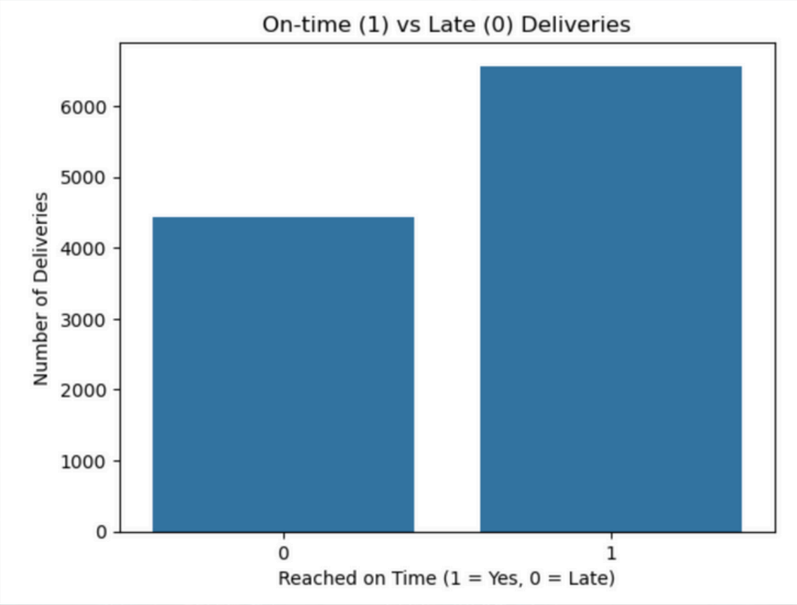
```python
# Check missing values
print("\n MISSING VALUES PER COLUMN")
print(df.isna().sum())
```

```
 MISSING VALUES PER COLUMN
ID                     0
Warehouse_block        0
Mode_of_Shipment       0
Customer_care_calls    0
Customer_rating        0
Cost_of_the_Product    0
Prior_purchases        0
Product_importance     0
Gender                 0
Discount_offered       0
Weight_in_gms          0
Reached.on.Time_Y.N    0
dtype: int64
```

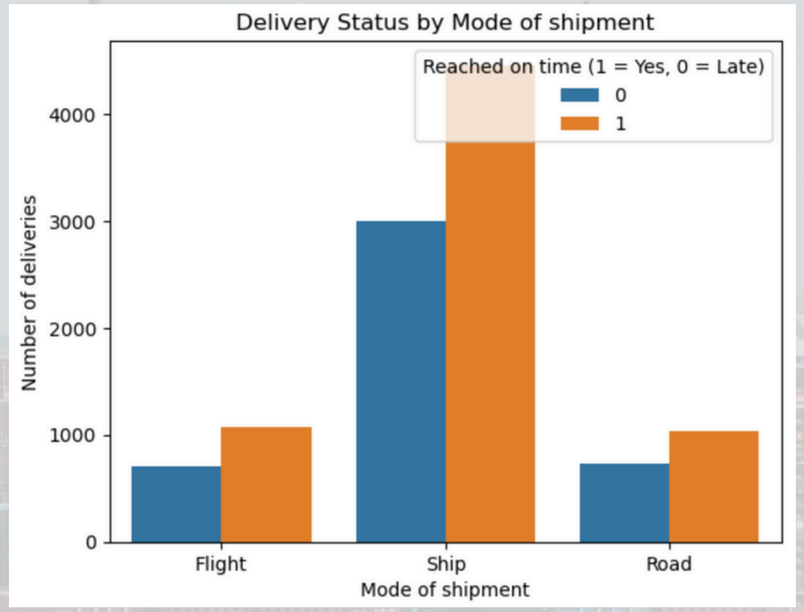
```python
# On-time vs late deliveries
# Reached.on.Time_Y.N: 1 = on time; 0 = late
print("\n ON-TIME VS LATE DELIVERIES")
print(df["Reached.on.Time_Y.N"].value_counts())
```

```
 ON-TIME VS LATE DELIVERIES
Reached.on.Time_Y.N
1    6563
0    4436
Name: count, dtype: int64
```

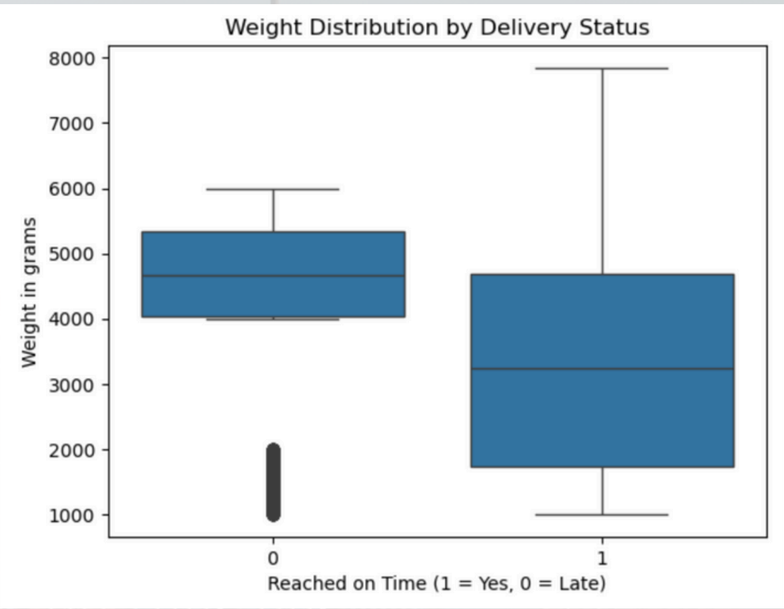
EDA (2)




```
# Plot: On-time vs late deliveries
sns.countplot(data=df, x="Reached.on.Time_Y.N")
plt.title("On-time (1) vs Late (0) Deliveries")
plt.xlabel("Reached on Time (1 = Yes, 0 = Late)")
plt.ylabel("Number of Deliveries")
plt.show()
```




```
# Plot: Delivery status by Mode of shipment
sns.countplot(
    data=df,
    x="Mode_of_Shipment",
    hue="Reached.on.Time_Y.N"
)
plt.title("Delivery Status by Mode of shipment")
plt.xlabel("Mode of shipment")
plt.ylabel("Number of deliveries")
plt.legend(title="Reached on time (1 = Yes, 0 = Late)")
plt.show()
```




```
# Plot: Weight distribution by delivery status
sns.boxplot(
    data=df,
    x="Reached.on.Time_Y.N",
    y="Weight_in_gms"
)
plt.title("Weight Distribution by Delivery Status")
plt.xlabel("Reached on Time (1 = Yes, 0 = Late)")
plt.ylabel("Weight in grams")
plt.show()
```

- Box = where most shipment weights fall
- Line = median (typical) weight
- Late deliveries have heavier shipments

MODELING (1)

(LOGISTIC REGRESSION) + PRINTING STRONGEST FEATURES

```python
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```python
TARGET_COL = "Reached.on.Time_Y.N"

X = df.drop(columns=[TARGET_COL])
y = df[TARGET_COL]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2, # 80% training, 20% testing
    random_state=42, # same split each run
    stratify=y # keep similar 0/1 proportions in train/test
)
```

```python
categorical_cols = X.select_dtypes(include=["object"]).columns.tolist()
numeric_cols = X.select_dtypes(exclude=["object"]).columns.tolist()

preprocessor = ColumnTransformer(
    transformers=[
        ("cat", OneHotEncoder(handle_unknown="ignore"), categorical_cols),
        ("num", "passthrough", numeric_cols),
    ]
)

model = LogisticRegression(max_iter=5000, solver="liblinear")

clf = Pipeline(steps=[
    ("preprocessor", preprocessor),
    ("model", model)
])
```

```python
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print("Accuracy:", round(accuracy_score(y_test, y_pred), 4))
print("\nConfusion matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.6582

Confusion matrix:
 [[563 324]
 [428 885]]

Classification report:
              precision    recall  f1-score   support

           0       0.57      0.63      0.60       887
           1       0.73      0.67      0.70      1313

    accuracy                           0.66      2200
   macro avg       0.65      0.65      0.65      2200
weighted avg       0.67      0.66      0.66      2200
```

- Accuracy: 65.8% (1,448 / 2,200 correct)
- Correct: 885 on-time, 563 late
- Errors: 324 late -> on-time, 428 on-time ->late
- Precision: on-time 0.73, late 0.57
- Recall: on-time 0.67, late 0.63
- Better at predicting on-time deliveries

MODELING (2)

```
preprocessor = clf.named_steps["preprocessor"]
feature_names = preprocessor.get_feature_names_out()

coefs = clf.named_steps["model"].coef_[0]

coef_df = pd.DataFrame({"feature": feature_names, "coefficient": coefs})
coef_df = coef_df.sort_values(by="coefficient", ascending=False)

top_n = 10

print("TOP FEATURES PUSHING TOWARD ON-TIME (class 1)")
display(coef_df.head(top_n))

print("\nTOP FEATURES PUSHING TOWARD LATE (class 0)")
display(coef_df.tail(top_n).sort_values(by="coefficient"))
```

- Coefficients = feature impact on delivery timing
- Positive - more on-time, negative - more late
- High product importance (+0.426) - most on-time
- Male (+0.413) & female (+0.387) customers - more on-time
- Transport: flight (+0.303), ship (+0.267), road (+0.230)
- Warehouses: block D (+0.230), block B (+0.168)
- Many customer care calls (−0.077) - more late
- Many prior purchases (−0.043)- more late
- Cost & weight - very small effect

TOP FEATURES PUSHING TOWARD ON-TIME (class 1)

	feature	coefficient
8	cat__Product_importance_high	0.426138
12	cat__Gender_M	0.412527
11	cat__Gender_F	0.386908
5	cat__Mode_of_Shipment_Flight	0.302564
7	cat__Mode_of_Shipment_Ship	0.266839
3	cat__Warehouse_block_D	0.230287
6	cat__Mode_of_Shipment_Road	0.230032
9	cat__Product_importance_low	0.205040
10	cat__Product_importance_medium	0.168257
1	cat__Warehouse_block_B	0.167662

TOP FEATURES PUSHING TOWARD LATE (class 0)

	feature	coefficient
14	num__Customer_care_calls	-0.077095
17	num__Prior_purchases	-0.042767
16	num__Cost_of_the_Product	-0.000680
19	num__Weight_in_gms	-0.000195
13	num__ID	-0.000162
15	num__Customer_rating	0.022286
18	num__Discount_offered	0.092898
2	cat__Warehouse_block_C	0.114653
0	cat__Warehouse_block_A	0.123120

Conclusion

- This project studied whether shipment characteristics can be used to predict if a delivery is on time or late.
- A Logistic Regression model was trained using shipment and customer-related features.
- The model achieved an **accuracy of about 66%**, meaning it correctly classified around two thirds of the deliveries.
- The results show that the model predicts on-time deliveries better than late deliveries.
- Feature analysis shows that **product importance, shipment mode, and warehouse location** have the strongest influence on delivery performance, while **customer care calls and prior purchases** are more related to late deliveries.
- The results indicate that shipment characteristics **can be used to predict delivery status to a moderate extent**, but more advanced models or additional data could further improve prediction accuracy.