Logistic Regression model 1

January 12, 2026

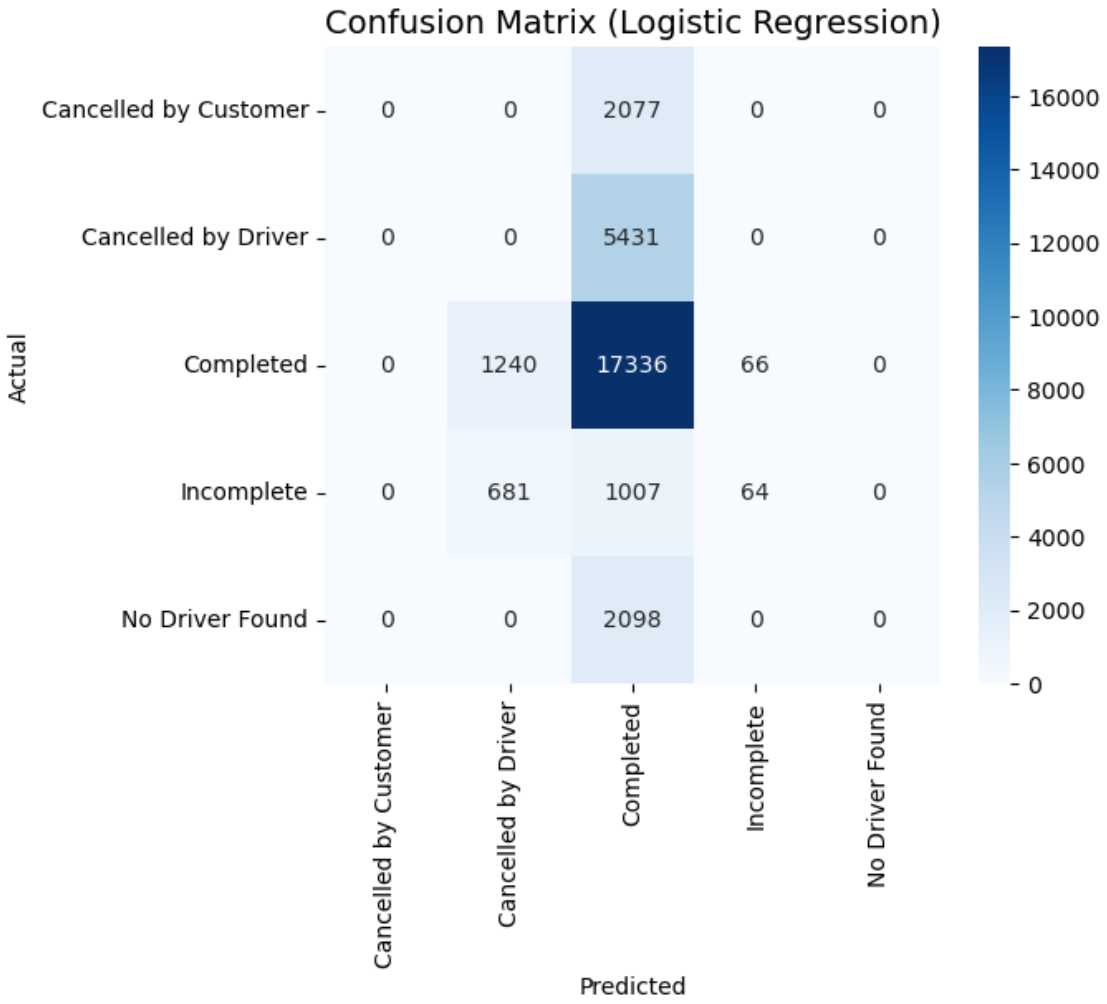
[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder, label_binarize
from sklearn.model_selection import train_test_split
from sklearn.metrics import (
    accuracy_score, f1_score, classification_report, confusion_matrix,
    roc_curve, auc
)
from sklearn.linear_model import LogisticRegression
from itertools import cycle

# Load dataset
df = pd.read_csv("ncr_ride_bookings.csv")

# Convert Date and Time
df['Date'] = pd.to_datetime(df['Date'])
df['year'] = df['Date'].dt.year
df['month'] = df['Date'].dt.month
df['day'] = df['Date'].dt.day

df['Time'] = pd.to_timedelta(df['Time'])
df['seconds'] = df['Time'].dt.total_seconds().astype(int)

# Select relevant columns
df = df[['year', 'month', 'day', 'seconds', 'Booking Status', 'Vehicle Type',
         'Pickup Location', 'Drop Location', 'Avg VTAT', 'Avg CTAT',
         'Booking Value', 'Ride Distance', 'Driver Ratings', 'Customer Rating',
         'Payment Method']]

# Handle missing values
df.fillna(df.median(numeric_only=True), inplace=True)

# Encode categorical variables
x_categorical = df.select_dtypes(include=['object']).apply(lambda col:␣
 ↪LabelEncoder().fit_transform(col))
```

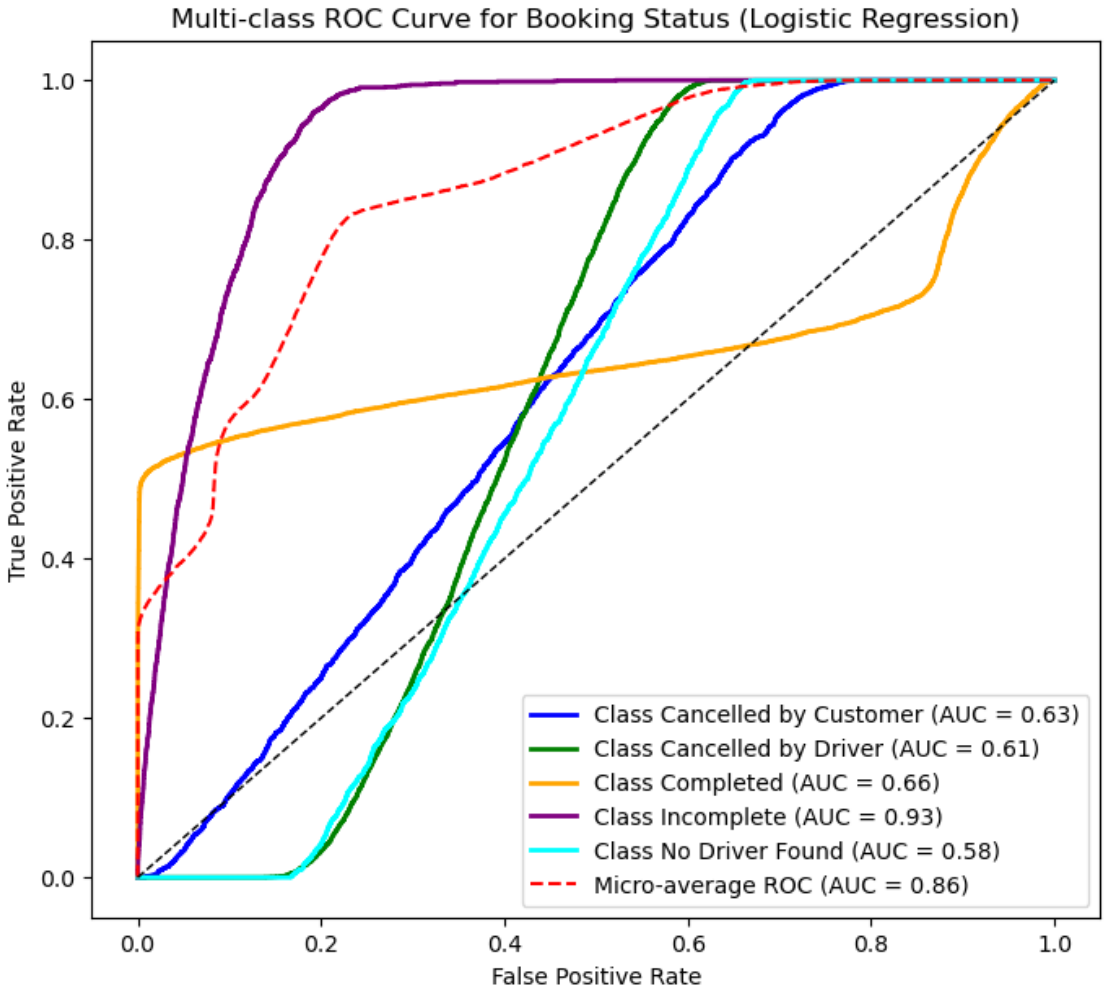
```
x_numerical = df.select_dtypes(exclude=['object'])
X = pd.concat([x_numerical, x_categorical], axis=1).drop(columns=['Booking␣
 ↪Status'])

# Encode target
le = LabelEncoder()
y = le.fit_transform(df['Booking Status'])

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Logistic Regression (multi-class = multinomial, solver 'lbfgs')
classifier = LogisticRegression(max_iter=500, multi_class='multinomial',␣
 ↪solver='lbfgs')
classifier.fit(X_train, y_train)

# Predictions
y_pred = classifier.predict(X_test)
y_prob = classifier.predict_proba(X_test)

# Evaluation
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')

print(f"Accuracy: {accuracy:.4f}")
print(f"F1 Score (weighted): {f1:.4f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred,␣
 ↪target_names=le.classes_))

# ========================
# Confusion Matrix
# ========================
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=le.classes_, yticklabels=le.classes_)
plt.title("Confusion Matrix (Logistic Regression)", fontsize=14)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# ========================
# ROC Curve (binary or multi-class)
# ========================
n_classes = len(le.classes_)
```

```
if n_classes == 2:
    # Binary ROC
    fpr, tpr, _ = roc_curve(y_test, y_prob[:, 1])
    roc_auc = auc(fpr, tpr)

    plt.figure(figsize=(7,6))
    plt.plot(fpr, tpr, color='blue', lw=2, label=f'ROC Curve (AUC = {roc_auc:.
 ↪3f})')
    plt.plot([0, 1], [0, 1], color='red', linestyle='--')
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title("ROC Curve for Booking Status (Logistic Regression)")
    plt.legend(loc="lower right")
    plt.show()

else:
    # Multi-class ROC
    y_test_bin = label_binarize(y_test, classes=np.arange(n_classes))

    fpr = dict()
    tpr = dict()
    roc_auc = dict()

    for i in range(n_classes):
        fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_prob[:, i])
        roc_auc[i] = auc(fpr[i], tpr[i])

    fpr["micro"], tpr["micro"], _ = roc_curve(y_test_bin.ravel(), y_prob.
 ↪ravel())
    roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

    plt.figure(figsize=(8,7))
    colors = cycle(['blue', 'green', 'orange', 'purple', 'cyan', 'brown'])
    for i, color in zip(range(n_classes), colors):
        plt.plot(fpr[i], tpr[i], color=color, lw=2,
                 label=f'Class {le.classes_[i]} (AUC = {roc_auc[i]:.2f})')

    # Fixed the key access - removed the extra quotes around "micro"
    plt.plot(fpr["micro"], tpr["micro"], color='red', linestyle='--',
             label=f'Micro-average ROC (AUC = {roc_auc["micro"]:.2f})')

    plt.plot([0, 1], [0, 1], 'k--', lw=1)
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title("Multi-class ROC Curve for Booking Status (Logistic Regression)")
    plt.legend(loc="lower right")
    plt.show()
```

```
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\linear_model\_logistic.py:1247: FutureWarning: 'multi_class'
was deprecated in version 1.5 and will be removed in 1.7. From then on, it will
always use 'multinomial'. Leave it to its default value to avoid this warning.
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:465:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1565: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1565: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1565: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
Accuracy: 0.5800
F1 Score (weighted): 0.4664

Classification Report:
                       precision    recall  f1-score   support

Cancelled by Customer       0.00      0.00      0.00      2077
  Cancelled by Driver       0.00      0.00      0.00      5431
            Completed       0.62      0.93      0.74     18642
           Incomplete       0.49      0.04      0.07      1752
      No Driver Found       0.00      0.00      0.00      2098

             accuracy                           0.58     30000
            macro avg       0.22      0.19      0.16     30000
         weighted avg       0.41      0.58      0.47     30000
```


Confusion Matrix (Logistic Regression)
Cancelled by Customer
Cancelled by Driver
Completed
Incomplete
No Driver Found
Actual
0 0 2077 0 0
0 0 5431 0 0
0 1240 17336 66 0
0 681 1007 64 0
0 0 2098 0 0
Cancelled by Customer
Cancelled by Driver
Completed
Incomplete
No Driver Found
Predicted
16000
14000
12000
10000
8000
6000
4000
2000
0


Multi-class ROC Curve for Booking Status (Logistic Regression)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Class Cancelled by Customer (AUC = 0.63)
Class Cancelled by Driver (AUC = 0.61)
Class Completed (AUC = 0.66)
Class Incomplete (AUC = 0.93)
Class No Driver Found (AUC = 0.58)
Micro-average ROC (AUC = 0.86)

[ ]: