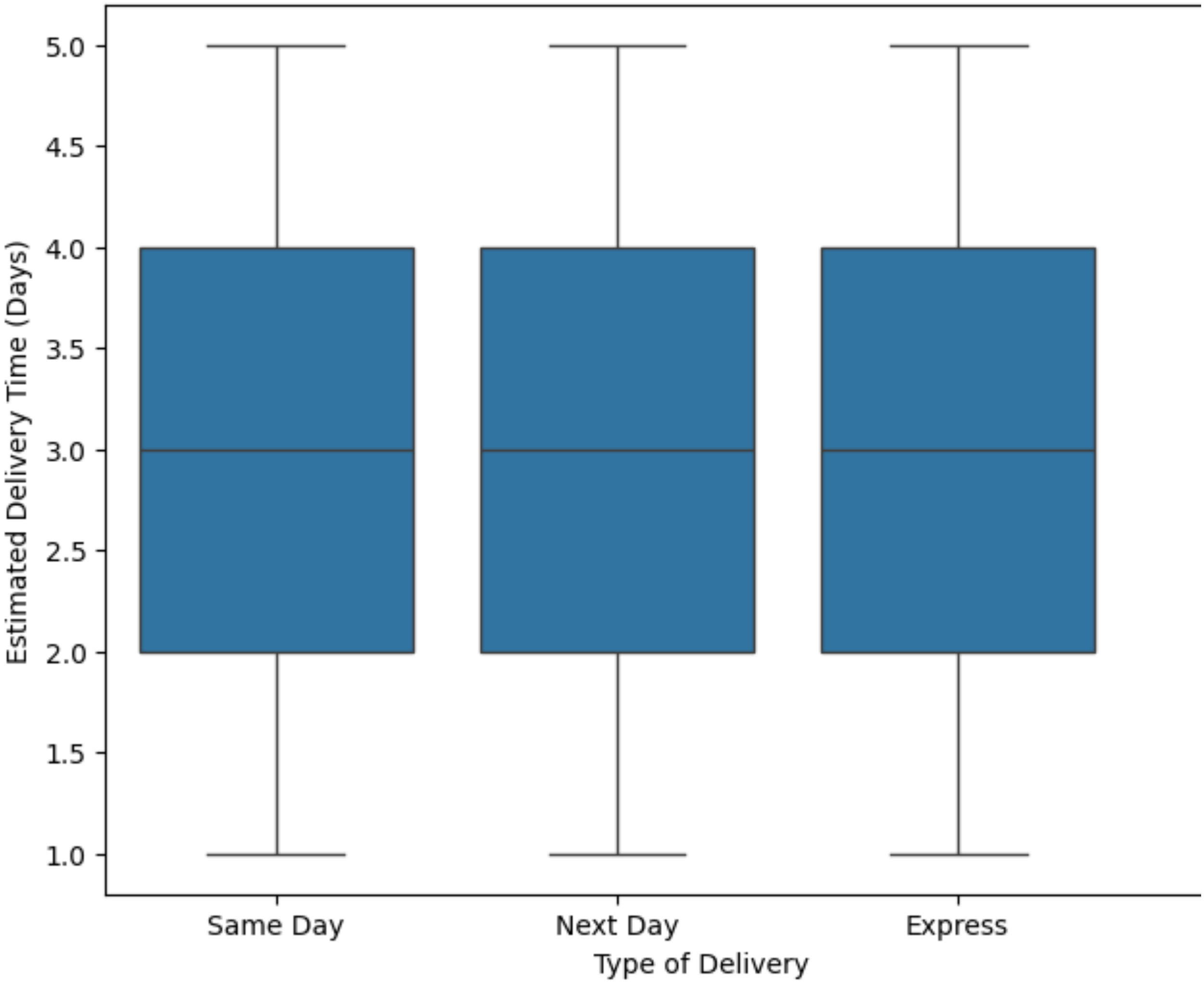
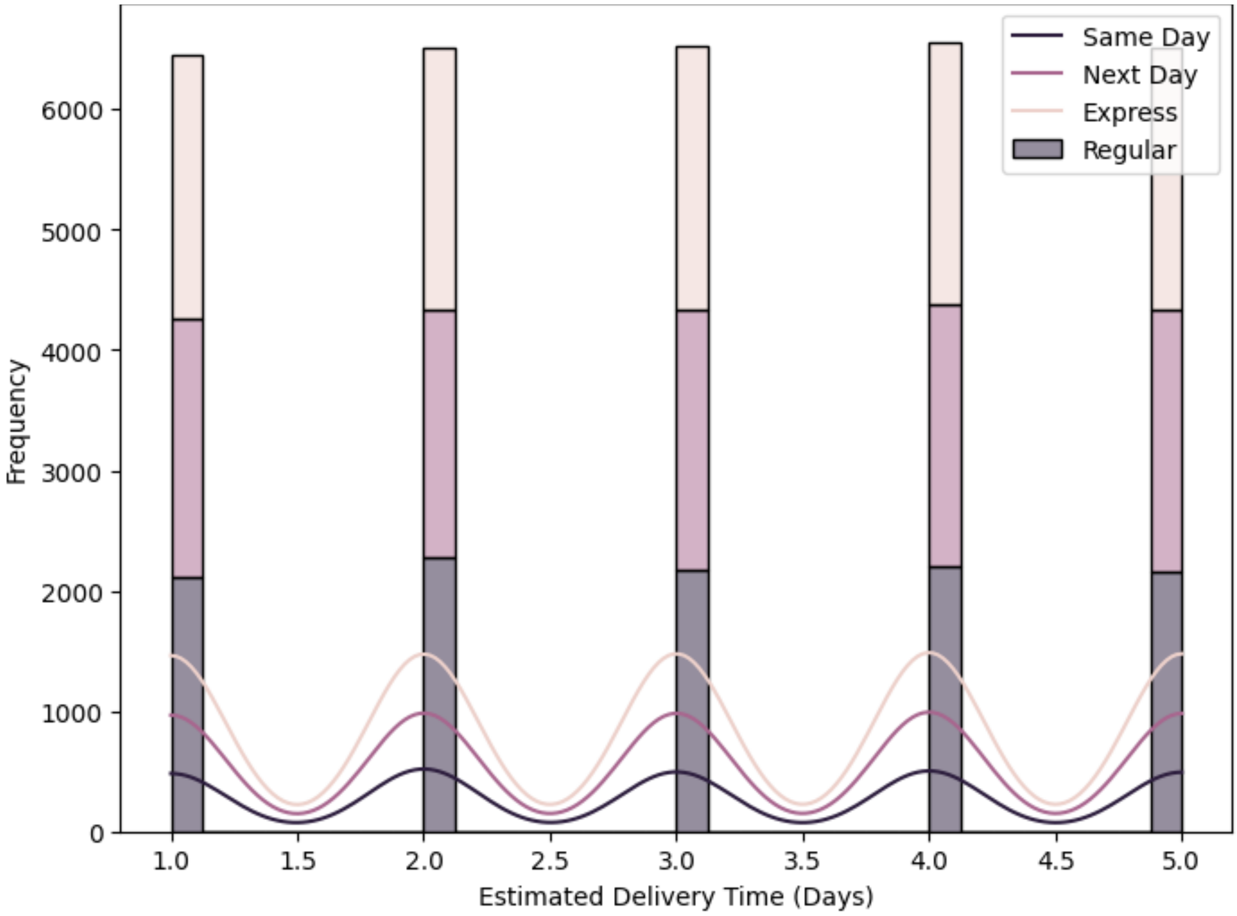
```
# Import necessary libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the dataset
data = pd.read_csv('/content/dataset_ecommerce.csv')  # Replace with your actual

# Encode 'type_of_delivery' to numeric values for the model
data['type_of_delivery'] = data['type_of_delivery'].map({
    'Same Day': 0,
    'Next Day': 1,
    'Express': 2,
    'Regular': 3
})

# Check if any rows have NaN values in the focused columns
data = data[['type_of_delivery', 'estimated_delivery_time_days']].dropna()

# Define features and target variable
# Using estimated_delivery_time_days as a simple target for this example
X = data[['type_of_delivery']]
y = data['estimated_delivery_time_days']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_

# Train a Decision Tree model
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Predict on test set and evaluate
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Model Accuracy:", accuracy)
print(classification_report(y_test, y_pred))

# Visualization 1: Type of Delivery vs. Estimated Delivery Time Days
plt.figure(figsize=(8, 6))
sns.boxplot(x='type_of_delivery', y='estimated_delivery_time_days', data=data)
plt.title('Type of Delivery vs. Estimated Delivery Time Days')
plt.xlabel('Type of Delivery')
plt.ylabel('Estimated Delivery Time (Days)')
plt.xticks(ticks=[0, 1, 2, 3], labels=['Same Day', 'Next Day', 'Express', 'Regul
plt.show()

# Visualization 2: Distribution of Estimated Delivery Time Days by Type of Deliv
plt.figure(figsize=(8, 6))
sns.histplot(data, x='estimated_delivery_time_days', hue='type_of_delivery', mul
plt.title('Distribution of Estimated Delivery Time Days by Type of Delivery')
plt.title('Distribution of Estimated Delivery Time Days by Type of Delivery')
plt.xlabel('Estimated Delivery Time (Days)')
plt.ylabel('Frequency')
plt.legend(['Same Day', 'Next Day', 'Express', 'Regular'])
plt.show()
```

```
Model Accuracy: 0.19930392056505272
              precision    recall  f1-score   support

         1.0       0.00      0.00      0.00      1907
         2.0       0.21      0.34      0.26      1969
         3.0       0.19      0.66      0.30      1925
         4.0       0.00      0.00      0.00      2005
         5.0       0.00      0.00      0.00      1963

    accuracy                           0.20      9769
   macro avg       0.08      0.20      0.11      9769
weighted avg       0.08      0.20      0.11      9769

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

Type of Delivery vs. Estimated Delivery Time Days



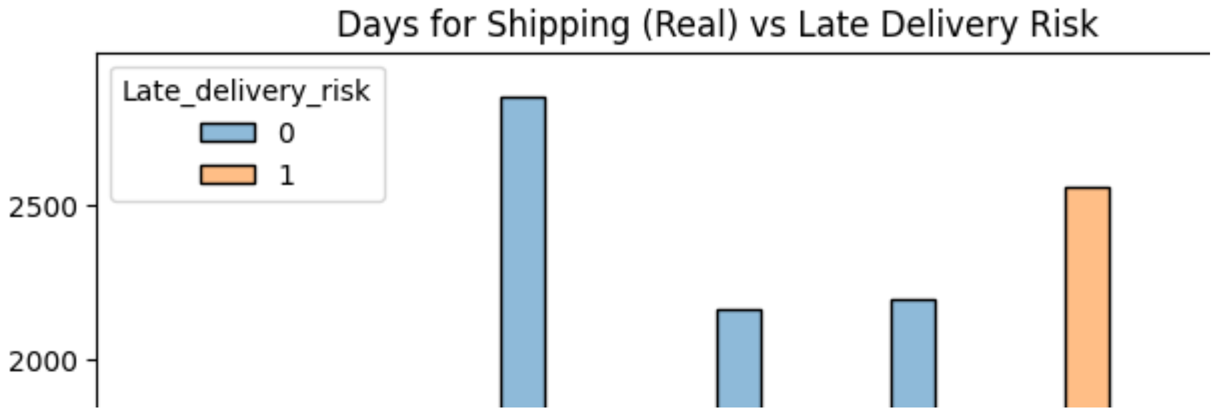
Distribution of Estimated Delivery Time Days by Type of Delivery

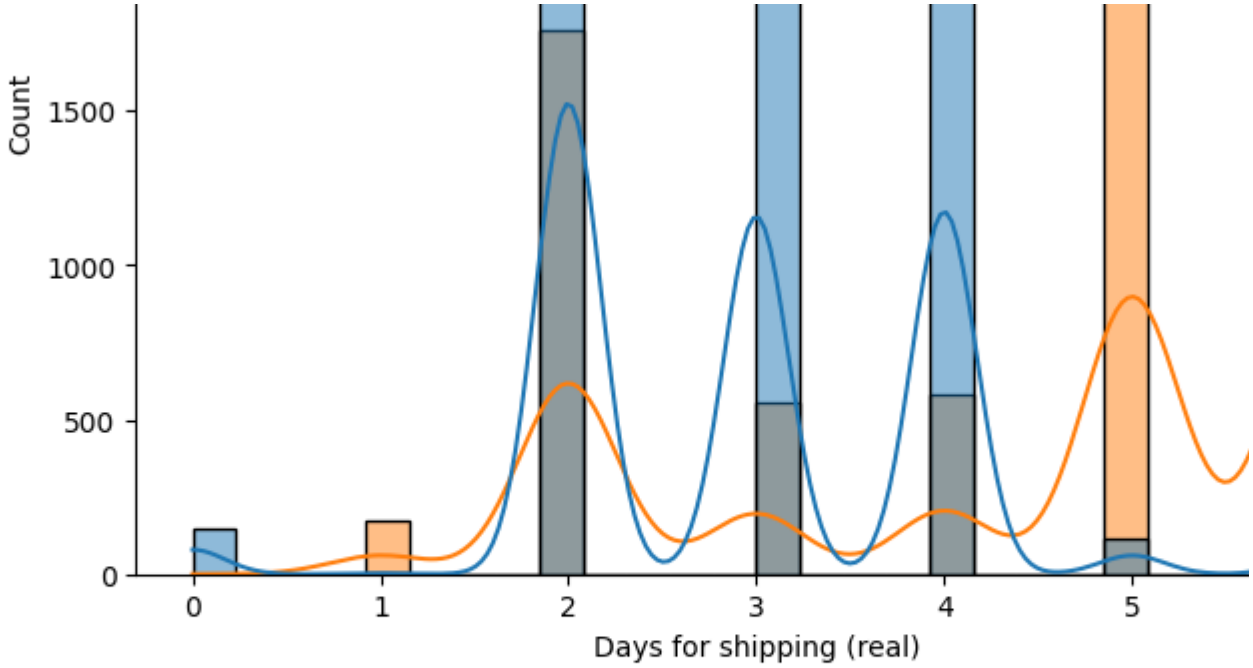


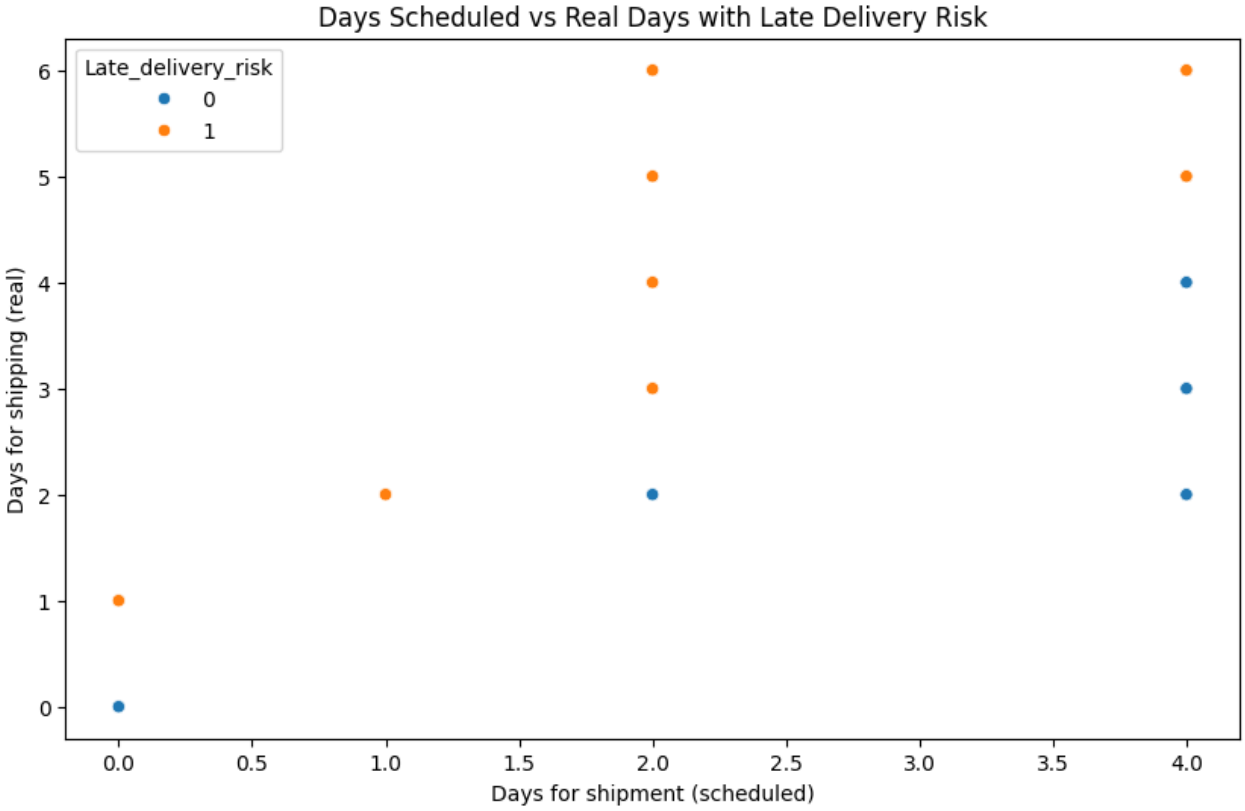
```
# Import libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load dataset
data = pd.read_csv('/content/CompanySupplyChainDataset.csv', encoding='ISO-8859

# Map 'Delivery Status' column to numeric values
data['Delivery Status'] = data['Delivery Status'].map({
    'Late delivery': 1,
    'Advance shipping': -1,
    'Shipping on time': 0,
    'Shipping canceled': -2
})

# Define features and target variable
X = data[['Days for shipping (real)', 'Days for shipment (scheduled)', 'Deliver
y = data['Late_delivery_risk']
```

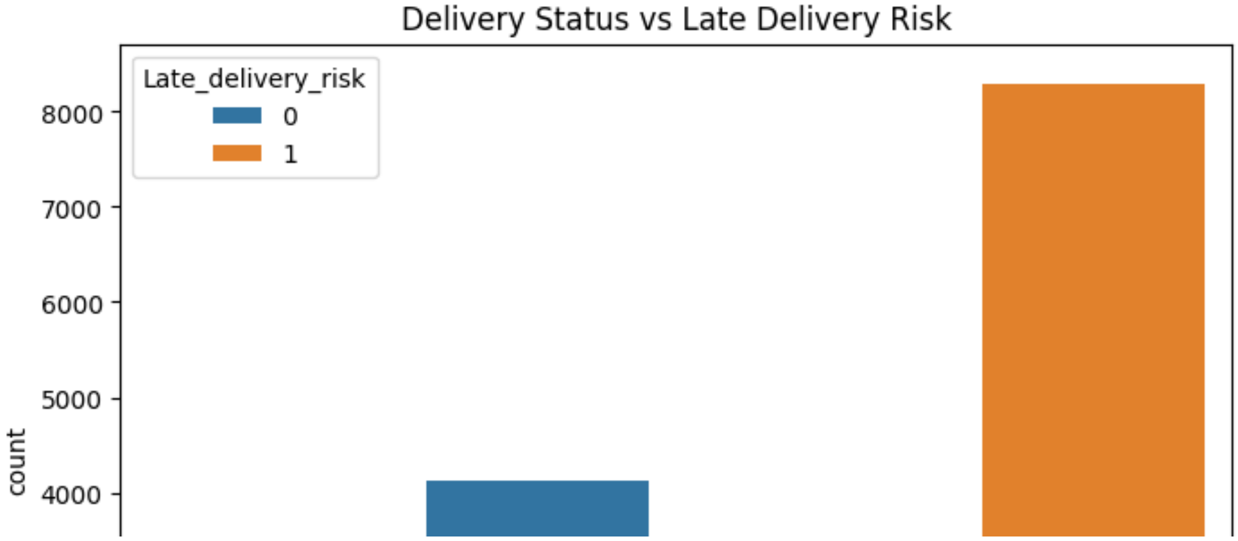
```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random

# Train a Decision Tree Classifier
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Predict on test set
y_pred = model.predict(X_test)

# Evaluate model
accuracy = accuracy_score(y_test, y_pred)
print("Model Accuracy:", accuracy)
print(classification_report(y_test, y_pred))

# Visualization 1: Distribution of Days for Shipping vs. Late Delivery Risk
plt.figure(figsize=(8, 6))
sns.histplot(data, x='Days for shipping (real)', hue='Late_delivery_risk', kde=
plt.title('Days for Shipping (Real) vs Late Delivery Risk')
plt.show()

# Visualization 2: Scatter Plot of Days Scheduled vs. Real Days with Late Deliv
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x='Days for shipment (scheduled)', y='Days for shipp
plt.title('Days Scheduled vs Real Days with Late Delivery Risk')
plt.show()

# Visualization 3: Delivery Status and Late Delivery Risk
plt.figure(figsize=(8, 6))
sns.countplot(data=data, x='Delivery Status', hue='Late_delivery_risk')
plt.title('Delivery Status vs Late Delivery Risk')
plt.show()
```

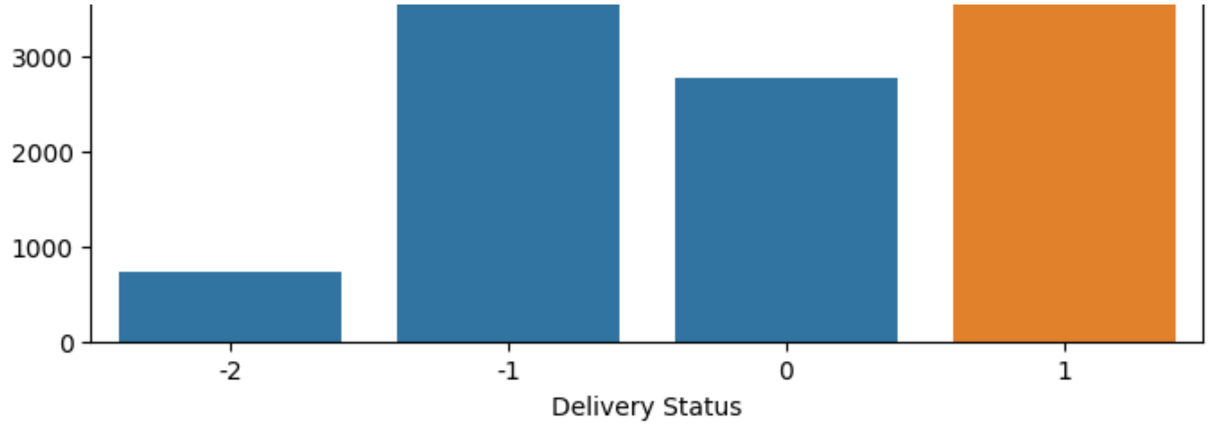
```
Model Accuracy: 1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      2285
           1       1.00      1.00      1.00      2491

    accuracy                           1.00      4776
   macro avg       1.00      1.00      1.00      4776
weighted avg       1.00      1.00      1.00      4776
```




Count
0
500
1000
1500
0
1
2
3
4
5
Days for shipping (real)


Days Scheduled vs Real Days with Late Delivery Risk
Late_delivery_risk
0
1
Days for shipping (real)
0
1
2
3
4
5
6
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Days for shipment (scheduled)


Delivery Status vs Late Delivery Risk
Late_delivery_risk
0
1
count
4000
5000
6000
7000
8000


3000
2000
1000
0
-2
-1
0
1
Delivery Status