# Decision Support System in Smart Supply Chains for Predicting Late Deliveries

The implementation of a Python-based Bayesian Network using pgmpy and applied to the “DataCo SMART SUPPLY CHAIN FOR BIG DATA ANALYSIS” dataset. The primary goal of the DSS is to predict orders that would be delivered late. To achieve this, data-derived conditional probability distributions (CPDs) were used rather than assuming or estimating a priori knowledge of the CPDs. The performance of the DSS is compared to a multi-class classification machine learning model. The DSS can generate multiple insights and conduct what-if analyses to support decision-making.

*Keywords:* Decision Support System (DSS), Bayesian Network, Supply Chain Management (SCM), decision-making

import pandas as pd  
from io import StringIO  
  
from pgmpy.estimators import MaximumLikelihoodEstimator, BayesianEstimator  
from pgmpy.estimators import ParameterEstimator  
from pgmpy.factors.discrete import TabularCPD, DiscreteFactor  
from pgmpy.inference import VariableElimination  
from pgmpy.models import BayesianNetwork  
  
from sklearn.metrics import classification\_report, confusion\_matrix  
import matplotlib.pyplot as plt

## 1. Data Preparation & Cleaning

### 1.1 Load Source Date

df\_source = pd.read\_csv('data/raw/DataCoSupplyChainDataset.csv', encoding='unicode\_escape')  
df\_source.drop\_duplicates(inplace=True)

### 1.2 Data Description

df\_description = pd.read\_csv('data/raw/DescriptionDataCoSupplyChain.csv')  
df\_description.DESCRIPTION = df\_description.DESCRIPTION.str.replace(':', '')  
  
df\_description.style.set\_properties(\*\*{'text-align': 'left'})

<pandas.io.formats.style.Styler at 0x7f9285144370>

## 2. Feature Engineering

Select the columns that will be used as the network nodes and remove duplicate records to have a single record per order.

nodes = [  
 'Order Id',  
 'Shipping Mode',  
 'Customer Segment',  
 'Days for shipment (scheduled)',  
 'Delivery Status',  
 'Customer State',  
 'Market',  
]  
  
df\_data = df\_source[nodes] \  
 .rename(columns={  
 'Type': 'Payment Type',  
 'Customer State': 'Store State'}) \  
 .drop\_duplicates() \  
 .reset\_index(drop=True)  
  
  
print(df\_data.shape)  
with pd.option\_context('display.max\_columns', None):  
 display(df\_data.head())

(65752, 7)

Order Id Shipping Mode Customer Segment Days for shipment (scheduled) \  
0 77202 Standard Class Consumer 4   
1 75939 Standard Class Consumer 4   
2 75938 Standard Class Consumer 4   
3 75937 Standard Class Home Office 4   
4 75936 Standard Class Corporate 4   
  
 Delivery Status Store State Market   
0 Advance shipping PR Pacific Asia   
1 Late delivery PR Pacific Asia   
2 Shipping on time CA Pacific Asia   
3 Advance shipping CA Pacific Asia   
4 Advance shipping PR Pacific Asia

# find columns with missing values  
df\_data.isnull().sum()

Order Id 0  
Shipping Mode 0  
Customer Segment 0  
Days for shipment (scheduled) 0  
Delivery Status 0  
Store State 0  
Market 0  
dtype: int64

## 3. Create a Training and Test Set

# the random state is used to generate the same results for each run  
random\_state = 98421

# create the training dataset  
df\_test = df\_data.sample(frac=0.3, random\_state=random\_state)  
df\_train = df\_data.drop(df\_test.index)  
  
# reset the index of both datasets  
df\_train.reset\_index(drop=True, inplace=True)  
df\_test.reset\_index(drop=True, inplace=True)  
  
print(f'Training dataset shape : {df\_train.shape}')  
print(f'Test dataset shape : {df\_test.shape}')

Training dataset shape : (46026, 7)  
Test dataset shape : (19726, 7)

## 4. Model Training

### 4.1 Model Definition

Define the model nodes and edges.

# Create the Bayesian network model  
model = BayesianNetwork()  
  
# Add the nodes to the model  
model.add\_node('Shipping Mode')  
model.add\_node('Customer Segment')  
model.add\_node('Days for shipment (scheduled)')  
model.add\_node('Delivery Status')  
model.add\_node('Store State')  
model.add\_node('Market')  
  
# Add the edges between the nodes to the model  
model.add\_edge('Delivery Status', 'Shipping Mode')  
model.add\_edge('Delivery Status', 'Customer Segment')  
model.add\_edge('Delivery Status', 'Days for shipment (scheduled)')  
model.add\_edge('Delivery Status', 'Store State')  
model.add\_edge('Delivery Status', 'Market')

### 4.2 Compute the Probability Distribution Table (PDT)

# show a simple parameter estimation of the shipping mode feature  
pe = ParameterEstimator(model, df\_train)  
pe.state\_counts('Shipping Mode')

Delivery Status Advance shipping Late delivery Shipping canceled \  
Shipping Mode   
First Class 0.0 6719.0 313.0   
Same Day 0.0 1110.0 119.0   
Second Class 0.0 6899.0 369.0   
Standard Class 10566.0 10479.0 1223.0   
  
Delivery Status Shipping on time   
Shipping Mode   
First Class 0.0   
Same Day 1220.0   
Second Class 1725.0   
Standard Class 5284.0

# estimate the CPDs using the Bayesian estimator  
model.fit(  
 data=df\_train,   
 estimator=BayesianEstimator,  
 prior\_type='BDeu',  
 equivalent\_sample\_size=10)

# Check if the model is valid  
model.check\_model()

True

# show the CPDs of the model  
model\_cpds = model.get\_cpds()  
model\_cpds

[<TabularCPD representing P(Shipping Mode:4 | Delivery Status:4) at 0x7f91a2adea30>,  
 <TabularCPD representing P(Customer Segment:3 | Delivery Status:4) at 0x7f91a2ade520>,  
 <TabularCPD representing P(Days for shipment (scheduled):4 | Delivery Status:4) at 0x7f91a2ade2b0>,  
 <TabularCPD representing P(Delivery Status:4) at 0x7f91a2adecd0>,  
 <TabularCPD representing P(Store State:46 | Delivery Status:4) at 0x7f91a2ade280>,  
 <TabularCPD representing P(Market:5 | Delivery Status:4) at 0x7f91a2eab700>]

# show the computed CPD values  
for cpd in model\_cpds:  
 print(f'--- {cpd.variable} ---')  
 print(cpd.values, end='\n\n')

--- Shipping Mode ---  
[[5.91380044e-05 2.66551300e-01 1.54761905e-01 7.59278382e-05]  
 [5.91380044e-05 4.40558123e-02 5.90303479e-02 1.48287068e-01]  
 [5.91380044e-05 2.73691466e-01 1.82395756e-01 2.09636761e-01]  
 [9.99822586e-01 4.15701422e-01 6.03811991e-01 6.42000243e-01]]  
  
--- Customer Segment ---  
[[0.5214395 0.52055111 0.5254544 0.52017656]  
 [0.3031493 0.29928532 0.28957974 0.29956063]  
 [0.17541121 0.18016356 0.18496587 0.18026281]]  
  
--- Days for shipment (scheduled) ---  
[[5.91380044e-05 4.40558123e-02 5.90303479e-02 1.48287068e-01]  
 [5.91380044e-05 2.66551300e-01 1.54761905e-01 7.59278382e-05]  
 [5.91380044e-05 2.73691466e-01 1.82395756e-01 2.09636761e-01]  
 [9.99822586e-01 4.15701422e-01 6.03811991e-01 6.42000243e-01]]  
  
--- Delivery Status ---  
[0.22957034 0.54760405 0.0440199 0.17880572]  
  
--- Store State ---  
[[5.14243517e-06 2.15584705e-06 2.68185670e-05 1.28086962e-04]  
 [5.14243517e-06 2.15584705e-06 2.68185670e-05 1.28086962e-04]  
 [1.94384049e-04 2.00493775e-04 2.68185670e-05 4.92540585e-04]  
 [8.56729699e-04 8.35175145e-04 5.20280200e-04 9.78478749e-04]  
 [1.86454414e-02 1.73765584e-02 1.82848990e-02 1.79863145e-02]  
 [1.58589615e-01 1.63789617e-01 1.55960695e-01 1.61702527e-01]  
 [1.13596393e-02 9.04636537e-03 9.89605123e-03 1.20335720e-02]  
 [6.81784055e-03 6.15063162e-03 5.45489653e-03 6.08082948e-03]  
 [2.84376665e-03 3.53257097e-03 3.97451163e-03 3.65113865e-03]  
 [1.14059212e-03 1.74752961e-03 2.49412673e-03 9.78478749e-04]  
 [3.17031128e-02 2.94355044e-02 3.06214398e-02 3.12281295e-02]  
 [1.37251595e-02 1.36478053e-02 1.38437443e-02 1.36128710e-02]  
 [7.66942781e-03 7.30099160e-03 9.40258960e-03 4.50153044e-03]  
 [3.83625664e-04 5.57502046e-04 5.20280200e-04 4.92540585e-04]  
 [1.23521293e-03 9.93845488e-04 2.68185670e-05 9.78478749e-04]  
 [3.86104317e-02 4.24068049e-02 3.70364411e-02 4.21617382e-02]  
 [3.78997472e-03 3.45323580e-03 2.49412673e-03 3.04371595e-03]  
 [1.99217938e-03 2.14420547e-03 1.01374183e-03 2.19332416e-03]  
 [3.22224988e-03 2.50121374e-03 1.50720347e-03 2.19332416e-03]  
 [5.30390763e-03 4.88126888e-03 8.90912797e-03 4.86598406e-03]  
 [7.85866943e-03 8.72902469e-03 1.18698978e-02 9.11794300e-03]  
 [1.14542601e-02 1.33304646e-02 1.58175908e-02 1.23980256e-02]  
 [2.00647535e-02 2.03516273e-02 1.92718223e-02 2.13878817e-02]  
 [2.27604181e-03 3.69124131e-03 6.93528143e-03 3.52965411e-03]  
 [6.15549490e-03 7.69766746e-03 5.94835817e-03 8.02458213e-03]  
 [5.72867278e-04 4.78166875e-04 1.50720347e-03 4.92540585e-04]  
 [9.65646476e-03 1.12677502e-02 1.28568210e-02 1.04542730e-02]  
 [8.56729699e-04 1.11284824e-03 2.00066510e-03 1.22144783e-03]  
 [1.95916495e-02 1.72972232e-02 2.02587455e-02 1.73788918e-02]  
 [4.07383714e-03 5.83329094e-03 4.46797327e-03 5.47340677e-03]  
 [7.85866943e-03 8.17367849e-03 9.40258960e-03 8.63200484e-03]  
 [6.86998484e-02 6.12885756e-02 5.48010599e-02 6.36645020e-02]  
 [2.25248945e-02 2.16209900e-02 2.61802851e-02 2.34531189e-02]  
 [1.42445454e-03 1.35085376e-03 1.01374183e-03 1.70738600e-03]  
 [9.56184395e-03 9.64137916e-03 9.40258960e-03 8.87497392e-03]  
 [2.04432368e-02 2.03516273e-02 2.12456688e-02 2.39390570e-02]  
 [3.88707418e-01 3.84460396e-01 3.93315740e-01 3.76365711e-01]  
 [1.23521293e-03 1.27151859e-03 5.20280200e-04 1.70738600e-03]  
 [2.46528342e-03 3.77057648e-03 4.96143490e-03 4.01559228e-03]  
 [6.81784055e-03 8.88769503e-03 9.40258960e-03 8.99645846e-03]  
 [4.84509957e-02 5.14510144e-02 4.29579807e-02 5.39457387e-02]  
 [6.06087409e-03 5.23827715e-03 3.97451163e-03 5.10895315e-03]  
 [1.20219849e-02 1.11884150e-02 1.03895129e-02 1.05757575e-02]  
 [4.64156198e-03 5.63495301e-03 4.46797327e-03 4.74449952e-03]  
 [6.43935732e-03 4.72259854e-03 3.97451163e-03 3.77262319e-03]  
 [1.99217938e-03 1.15251583e-03 1.01374183e-03 1.58590145e-03]]  
  
--- Market ---  
[[0.05937456 0.05793451 0.05748828 0.0625038 ]  
 [0.28324739 0.28515044 0.26918332 0.27412987]  
 [0.26053839 0.25789881 0.26918332 0.26319626]  
 [0.26924351 0.2689264 0.2642487 0.26769119]  
 [0.12759616 0.13008985 0.13989637 0.13247889]]

### 4.3 Model Details

model.active\_trail\_nodes('Delivery Status')

{'Delivery Status': {'Customer Segment',  
 'Days for shipment (scheduled)',  
 'Delivery Status',  
 'Market',  
 'Shipping Mode',  
 'Store State'}}

model.get\_independencies()

(Store State ⟂ Customer Segment, Days for shipment (scheduled), Market, Shipping Mode | Delivery Status)  
(Store State ⟂ Customer Segment, Days for shipment (scheduled), Market | Delivery Status, Shipping Mode)  
(Store State ⟂ Days for shipment (scheduled), Market, Shipping Mode | Delivery Status, Customer Segment)  
(Store State ⟂ Days for shipment (scheduled), Customer Segment, Shipping Mode | Delivery Status, Market)  
(Store State ⟂ Customer Segment, Market, Shipping Mode | Delivery Status, Days for shipment (scheduled))  
(Store State ⟂ Days for shipment (scheduled), Market | Delivery Status, Customer Segment, Shipping Mode)  
(Store State ⟂ Days for shipment (scheduled), Customer Segment | Delivery Status, Market, Shipping Mode)  
(Store State ⟂ Customer Segment, Market | Delivery Status, Days for shipment (scheduled), Shipping Mode)  
(Store State ⟂ Days for shipment (scheduled), Shipping Mode | Delivery Status, Market, Customer Segment)  
(Store State ⟂ Market, Shipping Mode | Delivery Status, Days for shipment (scheduled), Customer Segment)  
(Store State ⟂ Customer Segment, Shipping Mode | Delivery Status, Days for shipment (scheduled), Market)  
(Store State ⟂ Days for shipment (scheduled) | Delivery Status, Market, Customer Segment, Shipping Mode)  
(Store State ⟂ Market | Delivery Status, Days for shipment (scheduled), Customer Segment, Shipping Mode)  
(Store State ⟂ Customer Segment | Delivery Status, Days for shipment (scheduled), Market, Shipping Mode)  
(Store State ⟂ Shipping Mode | Delivery Status, Days for shipment (scheduled), Market, Customer Segment)  
(Shipping Mode ⟂ Customer Segment, Store State, Market, Days for shipment (scheduled) | Delivery Status)  
(Shipping Mode ⟂ Customer Segment, Days for shipment (scheduled), Market | Delivery Status, Store State)  
(Shipping Mode ⟂ Store State, Market, Days for shipment (scheduled) | Delivery Status, Customer Segment)  
(Shipping Mode ⟂ Store State, Customer Segment, Days for shipment (scheduled) | Delivery Status, Market)  
(Shipping Mode ⟂ Customer Segment, Store State, Market | Delivery Status, Days for shipment (scheduled))  
(Shipping Mode ⟂ Days for shipment (scheduled), Market | Delivery Status, Store State, Customer Segment)  
(Shipping Mode ⟂ Days for shipment (scheduled), Customer Segment | Delivery Status, Store State, Market)  
(Shipping Mode ⟂ Customer Segment, Market | Delivery Status, Store State, Days for shipment (scheduled))  
(Shipping Mode ⟂ Store State, Days for shipment (scheduled) | Delivery Status, Market, Customer Segment)  
(Shipping Mode ⟂ Store State, Market | Delivery Status, Days for shipment (scheduled), Customer Segment)  
(Shipping Mode ⟂ Store State, Customer Segment | Delivery Status, Days for shipment (scheduled), Market)  
(Shipping Mode ⟂ Days for shipment (scheduled) | Delivery Status, Store State, Market, Customer Segment)  
(Shipping Mode ⟂ Market | Delivery Status, Store State, Customer Segment, Days for shipment (scheduled))  
(Shipping Mode ⟂ Customer Segment | Delivery Status, Store State, Market, Days for shipment (scheduled))  
(Shipping Mode ⟂ Store State | Delivery Status, Days for shipment (scheduled), Market, Customer Segment)  
(Customer Segment ⟂ Store State, Days for shipment (scheduled), Market, Shipping Mode | Delivery Status)  
(Customer Segment ⟂ Days for shipment (scheduled), Market, Shipping Mode | Delivery Status, Store State)  
(Customer Segment ⟂ Store State, Days for shipment (scheduled), Shipping Mode | Delivery Status, Market)  
(Customer Segment ⟂ Store State, Market, Days for shipment (scheduled) | Delivery Status, Shipping Mode)  
(Customer Segment ⟂ Store State, Market, Shipping Mode | Delivery Status, Days for shipment (scheduled))  
(Customer Segment ⟂ Days for shipment (scheduled), Shipping Mode | Delivery Status, Store State, Market)  
(Customer Segment ⟂ Days for shipment (scheduled), Market | Delivery Status, Store State, Shipping Mode)  
(Customer Segment ⟂ Market, Shipping Mode | Delivery Status, Store State, Days for shipment (scheduled))  
(Customer Segment ⟂ Store State, Days for shipment (scheduled) | Delivery Status, Market, Shipping Mode)  
(Customer Segment ⟂ Store State, Shipping Mode | Delivery Status, Days for shipment (scheduled), Market)  
(Customer Segment ⟂ Store State, Market | Delivery Status, Days for shipment (scheduled), Shipping Mode)  
(Customer Segment ⟂ Days for shipment (scheduled) | Delivery Status, Store State, Market, Shipping Mode)  
(Customer Segment ⟂ Shipping Mode | Delivery Status, Store State, Market, Days for shipment (scheduled))  
(Customer Segment ⟂ Market | Delivery Status, Days for shipment (scheduled), Store State, Shipping Mode)  
(Customer Segment ⟂ Store State | Delivery Status, Days for shipment (scheduled), Market, Shipping Mode)  
(Market ⟂ Store State, Days for shipment (scheduled), Customer Segment, Shipping Mode | Delivery Status)  
(Market ⟂ Days for shipment (scheduled), Customer Segment, Shipping Mode | Delivery Status, Store State)  
(Market ⟂ Store State, Days for shipment (scheduled), Shipping Mode | Delivery Status, Customer Segment)  
(Market ⟂ Store State, Customer Segment, Days for shipment (scheduled) | Delivery Status, Shipping Mode)  
(Market ⟂ Store State, Customer Segment, Shipping Mode | Delivery Status, Days for shipment (scheduled))  
(Market ⟂ Days for shipment (scheduled), Shipping Mode | Delivery Status, Store State, Customer Segment)  
(Market ⟂ Days for shipment (scheduled), Customer Segment | Delivery Status, Store State, Shipping Mode)  
(Market ⟂ Customer Segment, Shipping Mode | Delivery Status, Store State, Days for shipment (scheduled))  
(Market ⟂ Store State, Days for shipment (scheduled) | Delivery Status, Customer Segment, Shipping Mode)  
(Market ⟂ Store State, Shipping Mode | Delivery Status, Days for shipment (scheduled), Customer Segment)  
(Market ⟂ Store State, Customer Segment | Delivery Status, Days for shipment (scheduled), Shipping Mode)  
(Market ⟂ Days for shipment (scheduled) | Delivery Status, Store State, Customer Segment, Shipping Mode)  
(Market ⟂ Shipping Mode | Delivery Status, Store State, Customer Segment, Days for shipment (scheduled))  
(Market ⟂ Customer Segment | Delivery Status, Days for shipment (scheduled), Store State, Shipping Mode)  
(Market ⟂ Store State | Delivery Status, Days for shipment (scheduled), Customer Segment, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Customer Segment, Store State, Market, Shipping Mode | Delivery Status)  
(Days for shipment (scheduled) ⟂ Customer Segment, Market, Shipping Mode | Delivery Status, Store State)  
(Days for shipment (scheduled) ⟂ Customer Segment, Store State, Market | Delivery Status, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Store State, Market, Shipping Mode | Delivery Status, Customer Segment)  
(Days for shipment (scheduled) ⟂ Store State, Customer Segment, Shipping Mode | Delivery Status, Market)  
(Days for shipment (scheduled) ⟂ Customer Segment, Market | Delivery Status, Store State, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Market, Shipping Mode | Delivery Status, Store State, Customer Segment)  
(Days for shipment (scheduled) ⟂ Customer Segment, Shipping Mode | Delivery Status, Store State, Market)  
(Days for shipment (scheduled) ⟂ Store State, Market | Delivery Status, Customer Segment, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Store State, Customer Segment | Delivery Status, Market, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Store State, Shipping Mode | Delivery Status, Market, Customer Segment)  
(Days for shipment (scheduled) ⟂ Market | Delivery Status, Store State, Customer Segment, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Customer Segment | Delivery Status, Store State, Market, Shipping Mode)  
(Days for shipment (scheduled) ⟂ Shipping Mode | Delivery Status, Store State, Market, Customer Segment)  
(Days for shipment (scheduled) ⟂ Store State | Delivery Status, Market, Customer Segment, Shipping Mode)

## 5. Model Queries

This section shows how the model can be used to answer questions about the data, make predictions, and generate insights.

infer = VariableElimination(model)

def get\_ratios(probabilities: DiscreteFactor, col\_name: str='Ratio') -> pd.DataFrame:  
 """  
 Show probabilities for a given variable.  
  
 Parameters  
 ----------  
 probabilities : DiscreteFactor  
 Probabilities for the variable.  
  
 Returns  
 -------  
 pd.DataFrame  
 Probabilities for the variable.  
 """  
 # Get the probabilities for each value  
 value\_probabilities = probabilities.values  
 variable\_name = list(probabilities.state\_names.keys())[0]  
 state\_names = probabilities.state\_names[variable\_name]  
  
 # create a dataframe with the probabilities  
 return pd.DataFrame(  
 data=value\_probabilities,  
 index=state\_names,  
 columns=[col\_name]) \  
 .sort\_values(by=col\_name, ascending=False)

### 5.1 What is the probability of a late delivery?

The probability of a late delivery is 0.55. This means that 55% of the orders in the training data were delivered late, which is a rather high percentage and indicates that the company has a problem with late deliveries.

late\_delivery = infer.query(  
 variables=['Delivery Status'],  
 joint=False)  
  
print(late\_delivery['Delivery Status'])

+------------------------------------+------------------------+  
| Delivery Status | phi(Delivery Status) |  
+====================================+========================+  
| Delivery Status(Advance shipping) | 0.2296 |  
+------------------------------------+------------------------+  
| Delivery Status(Late delivery) | 0.5476 |  
+------------------------------------+------------------------+  
| Delivery Status(Shipping canceled) | 0.0440 |  
+------------------------------------+------------------------+  
| Delivery Status(Shipping on time) | 0.1788 |  
+------------------------------------+------------------------+

#### 5.1.1 What are the indicators that an order will ship on time?

# query the joint probabilities for shipping on time  
on\_time\_factors = infer.query(  
 variables=['Shipping Mode', 'Customer Segment', 'Store State'],  
 evidence={'Delivery Status': 'Shipping on time'},  
 joint=True)  
  
# format the output  
factor\_string = on\_time\_factors.\_str(tablefmt='plain')  
factor\_stream = StringIO(factor\_string)  
  
pd.read\_fwf(factor\_stream, widths=[31, 31, 22, 54]) \  
 .rename(columns={'phi(Shipping Mode,Customer Segment,Store State)': 'phi'}) \  
 .replace('^.+\(', '', regex=True) \  
 .replace('\)$', '', regex=True) \  
 .sort\_values(by='phi', ascending=False) \  
 .reset\_index(drop=True) \  
 .head(20)

Shipping Mode Customer Segment Store State phi  
0 Standard Class Consumer PR 0.1257  
1 Standard Class Corporate PR 0.0724  
2 Standard Class Consumer CA 0.0540  
3 Standard Class Home Office PR 0.0436  
4 Second Class Consumer PR 0.0410  
5 Standard Class Corporate CA 0.0311  
6 Same Day Consumer PR 0.0290  
7 Second Class Corporate PR 0.0236  
8 Standard Class Consumer NY 0.0213  
9 Standard Class Home Office CA 0.0187  
10 Standard Class Consumer TX 0.0180  
11 Second Class Consumer CA 0.0176  
12 Same Day Corporate PR 0.0167  
13 Second Class Home Office PR 0.0142  
14 Standard Class Consumer IL 0.0141  
15 Same Day Consumer CA 0.0125  
16 Standard Class Corporate NY 0.0122  
17 Standard Class Consumer FL 0.0104  
18 Standard Class Corporate TX 0.0104  
19 Second Class Corporate CA 0.0102

### 5.2 Which states handle the most late deliveries?

We can see that this is close the the same ratio as the total number of orders per state, the slight variance in numbers is because a BayesianEstimator was used to estimate the CPDs. If we had used a MaximumLikelihoodEstimator the numbers would be the same.

# Query the ratio of a late delivery for each state  
state\_late\_delivery = infer.query(  
 variables=['Store State'],  
 evidence={'Delivery Status': 'Late delivery'})  
  
get\_ratios(state\_late\_delivery).head(5)

Ratio  
PR 0.384460  
CA 0.163790  
NY 0.061289  
TX 0.051451  
IL 0.042407

# verify the results with the source data  
df\_train['Store State'].value\_counts(normalize=True).head(5)

PR 0.384457  
CA 0.161908  
NY 0.063138  
TX 0.050841  
IL 0.041259  
Name: Store State, dtype: float64

#### 5.2.1 What is the probability of a late delivery per State?

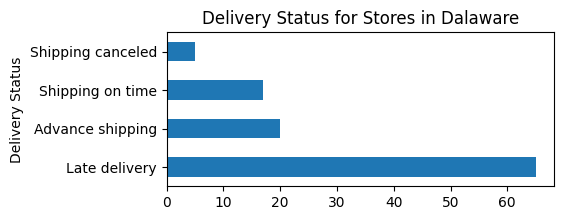
This query shows the probability of late deliveries for all the stores in a state. This query differs from the previous as the number of orders are not considered, only the probability of a late delivery.

df\_store\_late\_delivery = pd.DataFrame()  
  
for state\_name in list(state\_late\_delivery.state\_names.values())[0]:  
 # get the delivery status probabilities for the state using the bayesian network  
 df\_state = get\_ratios(  
 infer.query(  
 variables=['Delivery Status'],  
 evidence={'Store State': state\_name})  
 )  
  
 # add the state name to the dataframe  
 df\_state['Store State'] = state\_name  
  
 # append the dataframe to the main dataframe   
 df\_store\_late\_delivery = df\_store\_late\_delivery.append(df\_state \  
 .reset\_index(drop=False) \  
 .rename(columns={'index': 'Delivery Status'})  
 )  
  
# show the stores with the highest probability of late delivery  
df\_store\_late\_delivery \  
 .query('`Delivery Status` == "Late delivery"') \  
 .drop(columns=['Delivery Status']) \  
 .reindex(columns=['Store State', 'Ratio']) \  
 .sort\_values(by='Ratio', ascending=False) \  
 .rename(columns={'Ratio': 'Probability of Late delivery'}) \  
 .reset\_index(drop=True) \  
 .head(5)

Store State Probability of Late delivery  
0 DE 0.636464  
1 IA 0.605337  
2 NM 0.602145  
3 WA 0.593830  
4 MN 0.580801

We have discovered that the [state](https://www.scouting.org/resources/los/states/) Delaware (DE) has the highest probability of late delivery as reflected in the source data. This is an example that what require further investigation to improve the supply chain.

df\_data[df\_data['Store State'] == 'DE'].value\_counts('Delivery Status').plot.barh(figsize=(5, 2), title='Delivery Status for Stores in Dalaware')  
plt.show()



### 5.3 Which Market has the most late deliveries?

The European, Pacific Asia, and South American markets have the highest numbers of late deliver. This should be compared with the market profit to select the market to target first for improvement.

market\_late\_delivery = infer.query(  
 variables=['Market'],  
 evidence={'Delivery Status': 'Late delivery'})  
  
get\_ratios(market\_late\_delivery).head(5)

Ratio  
Europe 0.285150  
Pacific Asia 0.268926  
LATAM 0.257899  
USCA 0.130090  
Africa 0.057935

#### 5.3.1 What is the probability of a late delivery in Europe?

The probability of a late delivery in Europe is about 1% higher than the global probability.

get\_ratios(  
 infer.query(  
 variables=['Delivery Status'],  
 evidence={'Market': 'Europe'}),  
 col\_name='Probability')

Probability  
Late delivery 0.553643  
Advance shipping 0.230553  
Shipping on time 0.173791  
Shipping canceled 0.042013

#### 5.3.2 Which shipping method has the most late deliveries in the Pacific Asia Market?

Most of the late deliveries in the Pacific Asia market are due to the use of standard shipping.

shipping\_mode\_late\_delivery = infer.query(  
 variables=['Shipping Mode'],  
 evidence={  
 'Delivery Status': 'Late delivery',  
 'Market': 'Pacific Asia'  
 })  
  
get\_ratios(shipping\_mode\_late\_delivery).head(5)

Ratio  
Standard Class 0.415701  
Second Class 0.273691  
First Class 0.266551  
Same Day 0.044056

#### 5.3.3 What is the probability of a late delivery for orders in the Pacific Asia Market sipped using Standard Class?

When using standard shipping there is a 38% chance of a late delivery.

get\_ratios(  
 infer.query(  
 variables=['Delivery Status'],  
 evidence={  
 'Shipping Mode': 'Standard Class',  
 'Market': 'Pacific Asia'  
 }),  
 col\_name='Probability'  
).loc['Late delivery']

Probability 0.380781  
Name: Late delivery, dtype: float64

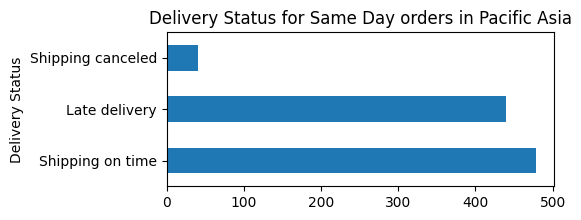
#### 5.3.4 What is the probability of a late delivery for orders in the Pacific Asia Market sipped using Same Day Shipping?

Same Day shipping is often the most expensive shipping method, so customers would find it quite disappointing that there is a 46% chance of a late delivery using this shipping method.

get\_ratios(  
 infer.query(  
 variables=['Delivery Status'],  
 evidence={  
 'Shipping Mode': 'Same Day',  
 'Market': 'Pacific Asia'  
 }),  
 col\_name='Probability'  
).loc['Late delivery']

Probability 0.454464  
Name: Late delivery, dtype: float64

# show the delivery status for Same Day orders in Pacific Asia from the source data  
df\_data \  
 .query('`Shipping Mode` == "Same Day" and `Market` == "Pacific Asia"') \  
 .value\_counts('Delivery Status') \  
 .plot.barh(figsize=(5, 2), title='Delivery Status for Same Day orders in Pacific Asia')  
  
plt.show()



## 1.7 Model Evaluation

# remove the label to predict  
df\_eval = df\_test \  
 .drop\_duplicates() \  
 .reset\_index(drop=True)  
  
df\_eval.head()

Order Id Shipping Mode Customer Segment Days for shipment (scheduled) \  
0 30009 Second Class Corporate 2   
1 71466 First Class Consumer 1   
2 54193 Standard Class Consumer 4   
3 39822 Standard Class Consumer 4   
4 23171 Standard Class Consumer 4   
  
 Delivery Status Store State Market   
0 Late delivery FL Pacific Asia   
1 Late delivery IL Europe   
2 Shipping on time MI LATAM   
3 Advance shipping FL USCA   
4 Advance shipping OR Pacific Asia

# predict the label  
df\_predict = model.predict(df\_eval.drop(columns=['Order Id', 'Delivery Status']))  
df\_predict.rename(columns={  
 'Delivery Status': 'y\_pred'}, inplace=True)  
  
# join the prediction back to the evaluation data  
df\_eval = df\_eval.join(df\_predict)  
  
df\_eval.head()

{"model\_id":"34257c3e3ae54e0c97cd976171efc0eb","version\_major":2,"version\_minor":0}

Order Id Shipping Mode Customer Segment Days for shipment (scheduled) \  
0 30009 Second Class Corporate 2   
1 71466 First Class Consumer 1   
2 54193 Standard Class Consumer 4   
3 39822 Standard Class Consumer 4   
4 23171 Standard Class Consumer 4   
  
 Delivery Status Store State Market y\_pred   
0 Late delivery FL Pacific Asia Late delivery   
1 Late delivery IL Europe Late delivery   
2 Shipping on time MI LATAM Advance shipping   
3 Advance shipping FL USCA Advance shipping   
4 Advance shipping OR Pacific Asia Advance shipping

# show the confusion matrix  
confusion\_matrix(  
 y\_true=df\_eval['Delivery Status'],  
 y\_pred=df\_eval['y\_pred'])

array([[4561, 0, 0, 0],  
 [4516, 5787, 0, 538],  
 [ 469, 317, 0, 45],  
 [2226, 728, 0, 539]])

# show the classification report  
print(classification\_report(  
 y\_true=df\_eval['Delivery Status'],  
 y\_pred=df\_eval['y\_pred']))

/home/vscode/.local/lib/python3.8/site-packages/sklearn/metrics/\_classification.py:1334: UndefinedMetricWarning: Precision and F-score are 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, msg\_start, len(result))

precision recall f1-score support  
  
 Advance shipping 0.39 1.00 0.56 4561  
 Late delivery 0.85 0.53 0.65 10841  
Shipping canceled 0.00 0.00 0.00 831  
 Shipping on time 0.48 0.15 0.23 3493  
  
 accuracy 0.55 19726  
 macro avg 0.43 0.42 0.36 19726  
 weighted avg 0.64 0.55 0.53 19726

/home/vscode/.local/lib/python3.8/site-packages/sklearn/metrics/\_classification.py:1334: UndefinedMetricWarning: Precision and F-score are 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, msg\_start, len(result))  
/home/vscode/.local/lib/python3.8/site-packages/sklearn/metrics/\_classification.py:1334: UndefinedMetricWarning: Precision and F-score are 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, msg\_start, len(result))

### 1.7.1 Create a AutoML Model for Comparison

from pycaret.classification import \*

classifier = setup(  
 data=df\_train.drop(columns=['Order Id']),  
 target='Delivery Status',  
 train\_size=0.7,  
 session\_id=random\_state,  
 verbose=False)

# perform a model comparison  
models = compare\_models(n\_select=3)  
top\_model = models[0]

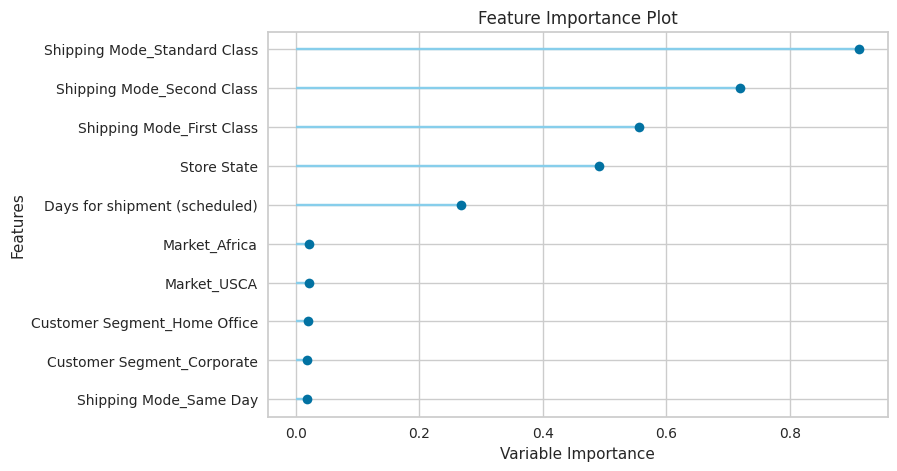
<IPython.core.display.HTML object>

<pandas.io.formats.style.Styler at 0x7f92570d70a0>

<IPython.core.display.HTML object>

# plot the feature importance  
plot\_model(top\_model, plot='feature')

<IPython.core.display.HTML object>



# perform predictions on the test data  
df\_predicted = predict\_model(estimator=top\_model, data=df\_test)

<pandas.io.formats.style.Styler at 0x7f925736bf10>

# show the confusion matrix  
confusion\_matrix(  
 y\_true=df\_predicted['Delivery Status'],  
 y\_pred=df\_predicted['prediction\_label'])

array([[4467, 94, 0, 0],  
 [4432, 5871, 0, 538],  
 [ 469, 317, 0, 45],  
 [2196, 758, 0, 539]])

# show the classification report  
print(classification\_report(  
 y\_true=df\_predicted['Delivery Status'],  
 y\_pred=df\_predicted['prediction\_label']))

precision recall f1-score support  
  
 Advance shipping 0.39 0.98 0.55 4561  
 Late delivery 0.83 0.54 0.66 10841  
Shipping canceled 0.00 0.00 0.00 831  
 Shipping on time 0.48 0.15 0.23 3493  
  
 accuracy 0.55 19726  
 macro avg 0.43 0.42 0.36 19726  
 weighted avg 0.63 0.55 0.53 19726