**PRODUCT SALES ANALYSIS**

**Phase 2: Incorporating machine learning algorithms to predict future sales trend or customer behaviors**

**1. Data Collection and Preprocessing:**

* Gather relevant data: Collect historical sales data, customer data, and any other relevant information. Ensure data quality and consistency.
* Data preprocessing: Clean, format, and transform the data into a suitable format for machine learning. Handle missing values and outliers.

**2. Data Splitting:**

* Split your dataset into training, validation, and testing sets. The training set is used to train the model, the validation set is used to tune hyperparameters, and the testing set is used to evaluate the model's performance.

**3. Model Selection:**

* Choose appropriate machine learning algorithms for your prediction task. For predicting sales trends or customer behaviors, common choices include linear regression, decision trees, random forests, gradient boosting, neural networks, and time series models like ARIMA or Prophet.

**4. Model Training:**

* Train your selected model(s) on the training dataset. Ensure you optimize hyperparameters and use appropriate evaluation metrics like Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE) for regression tasks, or accuracy and F1-score for classification tasks.

**5. Model Evaluation:**

* Assess the model's performance on the validation dataset. Fine-tune the model if necessary by adjusting hyperparameters or trying different algorithms.

**6. Feature Importance Analysis:**

* which features have the most impact on your predictions. This can help in feature selection or further refining the model.

## 7.Data Column

* OrderID : A specific ID given to each product OrderPriority : Priority of the product
* OrderQuantity: No of product items sold Sales ShipMode: Divided in two categories - Express Air and Regular Air
* Profit: Profit earned from the sale
* CustomerName: Name of the customer purchasing the products Region: Region to which the customer belongs - - -
* CustomerSegment: Divided as per the size of business
* ProductCategory: Divided according to the usage of the product ProductSub-Category: Divided according to the usage of the product
* ProductName: Name of the product ProductContainer: Type of container in which the product is shipped

**PROGRAM :**

Import statement:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from pandas\_profiling import ProfileReport

Include dataset:

df = pd.read\_csv('Salesstore.csv')

# Show percentage on countplot/barplot

def barPerc\_without\_hue\_v(ax, feature):

total = sum(feature)

for p in ax.patches:

percentage = '{:.1f}%'.format(100 \* p.get\_width()/total)

x = p.get\_x() + p.get\_width()

y = p.get\_y() + p.get\_height()

ax.annotate(percentage, (x, y), size = 12)

def barPerc\_without\_hue\_h(ax, feature):

total = sum(feature)

for p in ax.patches:

percentage = '{:.1f}%'.format(100 \* p.get\_height()/total)

x = p.get\_x() + p.get\_width() - 0.5

y = p.get\_y() + p.get\_height()

ax.annotate(percentage, (x, y), size = 12)

def barPerc\_without\_hue2\_v(ax, feature, dec=True):

total = len(feature)

for p in ax.patches:

if dec == True:

percentage = '{:.2f}'.format(p.get\_width())

else:

percentage = int(p.get\_width())

x = p.get\_x() + p.get\_width()

y = p.get\_y() + p.get\_height()

ax.annotate(percentage, (x, y), size = 12)

def barPerc\_without\_hue2\_h(ax, feature, dec=True):

total = len(feature)

for p in ax.patches:

if dec == True:

percentage = '{:.2f}'.format(p.get\_height())

else:

percentage = int(p.get\_height())

x = p.get\_x() + p.get\_width() - 0.6

y = p.get\_y() + p.get\_height()

ax.annotate(percentage, (x, y), size = 12)

# **Higher The Order Priority Level, The Higher The Income**

* 1. There isn't a significant difference in Order Quantity, so Priority has nothing to do with quantity.
  2. For Sales, it looks Critical Priority has the highest Min and Max Sale, this is normal because usually the highest the priority there will be an additional cost. But if we see at mean sales, Critical Priority is the lowest.
  3. Low Priority showed up as the Priority that gives the most ProfitThe conclusion is Order Priority is not very influential to Profit, because at first, we have a prejudice that the highest priority will give the higher profit, but after looking at this plot we can say that our prejudice is wrong. But we are still curious why Critical priority has the lowest profit, and we find out that Critical priority has the highest number of minus number on profit, and maybe we will know why Critical has the highest of minus in the next analysis.

**PROGRAM :**

groupby1 = df.groupby('Order\_Priority').agg({'Order\_Quantity': [np.sum, np.mean, np.min, np.max], 'Sales':[np.sum, np.mean, np.min, np.max], 'Profit': [np.sum, np.mean, np.min, np.max]}).reset\_index()

#display(groupby1)

# Order Quantity

fig, axs = plt.subplots(1,2, figsize=(10,3))

plt.suptitle('Order Quantity')

sum1 = groupby1['Order\_Quantity']['sum'].values

ax = sns.barplot(data=groupby1, x='Order\_Priority', y=sum1, ax=axs[0])

axs[0].set\_title('Sum')

barPerc\_without\_hue2\_h(ax, sum1, False)

mean1 = groupby1['Order\_Quantity']['mean'].values

ax2 = sns.barplot(data=groupby1, x='Order\_Priority', y=mean1, ax=axs[1])

axs[1].set\_title('Mean')

barPerc\_without\_hue2\_h(ax2, mean1)

plt.tight\_layout()

plt.show()

# Sales

fig, axs = plt.subplots(1,4, figsize=(15,4))

plt.suptitle('Sales')

sum2 = groupby1['Sales']['sum'].values

ax = sns.barplot(data=groupby1, x='Order\_Priority', y=sum2, ax=axs[0])

axs[0].set\_title('Sum')

barPerc\_without\_hue2\_h(ax, sum2, False)

mean2 = groupby1['Sales']['mean'].values

ax2 = sns.barplot(data=groupby1, x='Order\_Priority', y=mean2, ax=axs[1])

axs[1].set\_title('Mean')

barPerc\_without\_hue2\_h(ax2, mean2)

amin2 = groupby1['Sales']['amin'].values

ax3 = sns.barplot(data=groupby1, x='Order\_Priority', y=amin2, ax=axs[2])

axs[2].set\_title('Min')

barPerc\_without\_hue2\_h(ax3, amin2)

amax2 = groupby1['Sales']['amax'].values

ax4 = sns.barplot(data=groupby1, x='Order\_Priority', y=amax2, ax=axs[3])

axs[3].set\_title('Max')

barPerc\_without\_hue2\_h(ax4, amax2)

plt.tight\_layout()

plt.show()

# Profit

fig, axs = plt.subplots(1,4, figsize=(15,4))

plt.suptitle('Profit')

sum2 = groupby1['Profit']['sum'].values

ax = sns.barplot(data=groupby1, x='Order\_Priority', y=sum2, ax=axs[0])

axs[0].set\_title('Sum')

barPerc\_without\_hue2\_h(ax, sum2, False)

mean2 = groupby1['Profit']['mean'].values

ax2 = sns.barplot(data=groupby1, x='Order\_Priority', y=mean2, ax=axs[1])

axs[1].set\_title('Mean')

barPerc\_without\_hue2\_h(ax2, mean2)

amin2 = groupby1['Profit']['amin'].values

ax3 = sns.barplot(data=groupby1, x='Order\_Priority', y=amin2, ax=axs[2])

axs[2].set\_title('Min')

barPerc\_without\_hue2\_h(ax3, amin2)

amax2 = groupby1['Profit']['amax'].values

ax4 = sns.barplot(data=groupby1, x='Order\_Priority', y=amax2, ax=axs[3])

axs[3].set\_title('Max')

barPerc\_without\_hue2\_h(ax4, amax2)

plt.tight\_layout()

plt.show()

**output:**

A screenshot of a computer

Description automatically generated

**PROGRAM :**

z = df[df['Profit']<0]

z.groupby('Order\_Priority')['Profit'].count().reset\_index(name='count\_minus').sort\_values('count\_minus', ascending=False)

**output:**

A screenshot of a computer

Description automatically generated

# **Category That Give The Highest Profit**

* **Ship Mode**: If we see from the sum, regular air is given more profit than express, but this happens because many customers use regular than express air. If we compare with mean, it seems that express air gives more profit than regular. That's normal because we all know that express air will give you more cost than regular
* **Region**: This part is quite surprising, why? Because Northwest Territories dominate both in sum and mean profit, whereas Northwest Territories is at 3rd for count of the region but it can surpass West which is 1st.
* **Customer Segment**: Home Office has the higher mean profit than Corporate that has the highest sum profit, we do some price comparison for Home Office and Corporate with the same product, same ship mode, same region and the same product container and the result is the price different for Home Office and Corporate even though with exact product
* **Product Category**: This part for us is normal, because Technology always have high price
* **Product Container**: For this part, we want to focus on Medium Box, Medium Box has the lowest count but has the highest mean profit, after we do some research, Medium Box is often used for an expensive product.  
  The conclusion from this section is not all of the categories with the highest count have the highest profit and vice versa, and then this store has a different price for different Customer Segment even though with the exact product

**PROGRAM :**

catcol = ['Ship\_Mode', 'Region', 'Customer\_Segment', 'Product\_Category', 'Product\_Container', 'Product\_Sub-Category']

for i in catcol:

groupbyx = df.groupby(i).agg({'Profit': [np.sum, np.mean]}).reset\_index()

if i == 'Product\_Sub-Category':

fig, axs = plt.subplots(1,2, figsize=(15,5))

plt.suptitle(i)

sum1 = groupbyx['Profit']['sum'].values

ax = sns.barplot(data=groupbyx, x=sum1, y='Product\_Sub-Category', ax=axs[0])

axs[0].set\_title('Sum')

barPerc\_without\_hue2\_v(ax, sum1)

mean1 = groupbyx['Profit']['mean'].values

ax = sns.barplot(data=groupbyx, x=mean1, y='Product\_Sub-Category', ax=axs[1])

axs[1].set\_title('Mean')

barPerc\_without\_hue2\_v(ax, mean1)

else:

fig, axs = plt.subplots(1,2, figsize=(10,3))

plt.suptitle(i)

sum1 = groupbyx['Profit']['sum'].values

ax = sns.barplot(data=groupbyx, x=i, y=sum1, ax=axs[0])

axs[0].set\_title('Sum')

barPerc\_without\_hue\_h(ax, sum1)

mean1 = groupbyx['Profit']['mean'].values

ax2 = sns.barplot(data=groupbyx, x=i, y=mean1, ax=axs[1])

axs[1].set\_title('Mean')

if i == 'Region':

label = ['Atlantic', 'Northwest Territories', 'Nunavut', 'Ontario', 'Prarie', 'West']

axs[0].set\_xticklabels(label, rotation=20)

axs[1].set\_xticklabels(label, rotation=20)

barPerc\_without\_hue2\_h(ax2, mean1)

plt.tight\_layout()

plt.show()

**output:**

A screenshot of a computer

Description automatically generated

# **The Most Purchased Sub-Category In Each Region:**

* From the visualization we get the Products that always on TOP 5 in every region, the Products are Paper, Binder and Telephone Communication

**PROGRAM :**

test = df.groupby(['Region','Product\_Sub-Category'])['Region'].count().reset\_index(name='count')

fig, axs = plt.subplots(3,2, figsize=(15,8))

for i, col in enumerate(test['Region'].unique()):

plt.suptitle('The Most Purchased Product Sub-Category In Each Region', fontsize=15)

test1 = test[test['Region']==col].sort\_values('count', ascending=False).head(5)

x = [0,0,1,1,2,2]

y = [0,1,0,1,0,1]

sns.barplot(data= test1, x='count', y='Product\_Sub-Category', ax=axs[x[i]][y[i]])

axs[x[i]][y[i]].set\_title(f"Region {col}")

plt.tight\_layout()

plt.show()

**output:**

A screenshot of a computer

Description automatically generated

# **Customer In This Store :**

* Region West, Atlantic and Northwest Territories dominate Customer that give the most profit for the store and loyal customer. And For Customer who give the most loss and new customer dominated by Nunavut region

**PROGRAM :**

custprofit = df.groupby('Customer\_Name')['Profit'].sum().sort\_values(ascending=False).reset\_index(name='sumprofit')

custquantity = df.groupby('Customer\_Name')['Order\_Quantity'].sum().sort\_values(ascending=False).reset\_index(name='sumquantity')

cust = custprofit.merge(custquantity,on='Customer\_Name', how='left')

fig, axs = plt.subplots(2,2, figsize=(15,5))

plt.suptitle('All About Customer In This Store', fontsize=15)

ax = sns.barplot(data=cust.head(5), y='Customer\_Name', x='sumprofit', ax=axs[0][0])

axs[0][0].set\_title('Customer That Give The Most Profit')

barPerc\_without\_hue2\_v(ax, cust['sumprofit'])

ax = sns.barplot(data=cust.tail(5), y='Customer\_Name', x='sumprofit', ax=axs[0][1])

axs[0][1].set\_title('Customer That Give The Most Loss')

barPerc\_without\_hue2\_v(ax, cust['sumprofit'])

cust1 = cust.sort\_values('sumquantity', ascending=False)

ax = sns.barplot(data=cust1.head(5), y='Customer\_Name', x='sumquantity', ax=axs[1][0])

axs[1][0].set\_title('Loyal Customer')

barPerc\_without\_hue2\_v(ax, cust1['sumquantity'])

ax = sns.barplot(data=cust1.tail(5), y='Customer\_Name', x='sumquantity', ax=axs[1][1])

axs[1][1].set\_title('New Customer')

barPerc\_without\_hue2\_v(ax, cust1['sumquantity'])

plt.tight\_layout()

plt.show()

**output:**

A screenshot of a computer

Description automatically generated

# **Customer Segment Distributed In Each Region :**

* West region dominate in every segment except for Small Business, Northwest Territories has more segments in Small Business than other region.

**PROGRAM :**

plt.figure(figsize=(10,5))

sns.countplot(data=df, x='Customer\_Segment', hue='Region')

plt.title('Customer Segmentation In Each Region')

plt.show()

**output:**

A screenshot of a graph

Description automatically generated

# **Region Has The Most Order Quantity :**

* West become a region with the most ordered quantity by sum, this normal because West is the most count of Region. For mean order quantity, Atlantic and Prarie show up as the region with mean of order quantity more than 26.

**PROGRAM :**

reg1 = df.groupby('Region')['Order\_Quantity'].sum().reset\_index(name='sum')

reg2 = df.groupby('Region')['Order\_Quantity'].mean().reset\_index(name='mean')

fig, axs = plt.subplots(1,2, figsize=(15,5))

plt.suptitle('Order Quantity Per Region', fontsize=15)

ax = sns.barplot(data=reg1, y='Region', x='sum', ax=axs[0])

axs[0].set\_title('SUM')

barPerc\_without\_hue2\_v(ax, reg1['sum'])

ax = sns.barplot(data=reg2, y='Region', x='mean', ax=axs[1])

axs[1].set\_title('MEAN')

barPerc\_without\_hue2\_v(ax, reg2['mean'])

plt.tight\_layout()

**output:**

A screenshot of a graph

Description automatically generated

# **Order Priority Distributed In Each Region :**

**PROGRAM :**

gb1 = df.groupby(['Region', 'Order\_Priority'])['Order\_Priority'].count()

gb2 = gb1 / gb1.groupby(level=0).sum()

fig, axs = plt.subplots(2,3, figsize=(15,7))

plt.suptitle('Order Priority Distribution In Each Region', fontsize=15)

gb2['Atlantic'].plot(kind='bar', ax=axs[0][0], colormap='Dark2', title='Atlantic')

axs[0][0].axes.get\_xaxis().set\_visible(False)

gb2['Northwest Territories'].plot(kind='bar', ax=axs[0][1], colormap='Paired', title='Northwest Territories')

axs[0][1].axes.get\_xaxis().set\_visible(False)

gb2['Ontario'].plot(kind='bar', ax=axs[0][2], colormap='Accent', title='Ontario')

axs[0][2].axes.get\_xaxis().set\_visible(False)

gb2['Prarie'].plot(kind='bar', ax=axs[1][0], colormap='bone', title='Prarie')

gb2['Nunavut'].plot(kind='bar', ax=axs[1][1], colormap='hsv', title='Nunavut')

gb2['West'].plot(kind='bar', ax=axs[1][2], colormap='brg', title='West')

plt.tight\_layout()

**output:**

A screenshot of a computer

Description automatically generated

# **Names Of The Product That Yield The Highest Profit And Loss To The Store :**

**PROGRAM :**

df['Marginal Profit'] = df['Profit'] / df['Order\_Quantity']

df['Marginal Profit'] = df['Marginal Profit'].round(2)

max\_marginal = df.sort\_values('Marginal Profit',ascending = False).head(10)

min\_marginal = df.sort\_values('Marginal Profit',ascending = True).head(10)

fig, axs = plt.subplots(1,2, figsize=(20,10))

#plt.suptitle('Top 10 Single Product That Give the Most Loss and Profit', fontsize=15)

ax = sns.barplot(data=max\_marginal, y='Product\_Name', x='Marginal Profit', ax=axs[0])

axs[0].set\_title('Top 10 Hihghest Profit Products')

barPerc\_without\_hue2\_v(ax, max\_marginal['Marginal Profit'])

ax = sns.barplot(data=min\_marginal, y='Product\_Name', x='Marginal Profit', ax=axs[1])

axs[1].set\_title('Top 10 Loss Products ')

barPerc\_without\_hue2\_v(ax, min\_marginal['Marginal Profit'])

plt.tight\_layout()

plt.show()

**output:**

A screenshot of a computer

Description automatically generated

# **Sales, Profit and Profit Margin Analysis**

* In this section, we will introduce profit margin into the analysis. The percentage % of profit margin indicates **how many cents of profit the business has generated for each dollar of sale**. For instance, if a product achieved a **50% profit margin** during the last quarter, it means the product had a **net positive profit of $0.50 for each dollar of sales generated**.
* Conversely, a **negative percentage % of profit margin** indicates **how many cents of loss the business has generated for each dollar of sale**. For instance, if a product achieved a **-50% profit margin**, it means the business loss $0.50 cent for each sale of the product.
* The percentage of profit margin can be calculated by = (Profit / Sales) \* 100% **OR** [(Sales - Cost)/(Sales)] \* 100%

**PROGRAM :**

customers\_profit = df.groupby('Customer\_Name')['Profit'].sum().sort\_values(ascending=False).reset\_index(name='Profit')

customers\_sales = df.groupby('Customer\_Name')['Sales'].sum().sort\_values(ascending=False).reset\_index(name='Sales')

sales\_sum = customers\_sales['Sales'].sum()

profit\_sum= customers\_profit['Profit'].sum()

customers\_sales['Sales Contribution %'] = (customers\_sales['Sales'] / sales\_sum) \* 100

customers\_profit['Profit Contribution %'] = (customers\_profit['Profit'] / profit\_sum) \* 100

inner = pd.merge(customers\_sales, customers\_profit, how="inner", on="Customer\_Name")

inner['Profit Margin %'] = (inner['Profit'] / inner['Sales']) \* 100

display(inner)

**output:**

A screenshot of a computer

Description automatically generated

# **Catergory and Subcategory is the Most Profitable**

category\_profit = df.groupby('Product\_Category')['Profit'].sum().sort\_values(ascending=False).reset\_index(name='Profit')

category\_sales = df.groupby('Product\_Category')['Sales'].sum().sort\_values(ascending=False).reset\_index(name='Sales')

sales\_sum = category\_sales['Sales'].sum()

profit\_sum= category\_profit['Profit'].sum()

category\_sales['Sales Contribution %'] = (category\_sales['Sales'] / sales\_sum) \* 100

category\_profit['Profit Contribution %'] = (category\_profit['Profit'] / profit\_sum) \* 100

inner = pd.merge(category\_sales, category\_profit, how="inner", on="Product\_Category")

inner['Profit Margin %'] = (inner['Profit'] / inner['Sales']) \* 100

display(inner)

print("-----------------------------------" + "\n")

fig, axs = plt.subplots(1,2, figsize=(15,5))

ax = sns.barplot(data= inner, y='Product\_Category', x='Profit', ax=axs[0])

ax.set(ylabel=None)

axs[0].set\_title('Profit For Each Product Category')

barPerc\_without\_hue\_v(ax, inner['Profit'])

ax = sns.barplot(data=inner, y='Product\_Category', x='Profit Margin %', ax=axs[1])

axs[1].set\_title('Profit Margin % For Product Category')

ax.set(ylabel=None)

barPerc\_without\_hue2\_v(ax, inner['Profit Margin %'])

plt.tight\_layout()

**output:**

A screenshot of a computer

Description automatically generated

# 

**Conclusion :**

In conclusion, integrating machine learning algorithms for predicting future sales trends or customer behaviors offers businesses a powerful tool to gain valuable insights and make informed decisions. As businesses navigate this intersection of technology and human insight, they position themselves to thrive in an increasingly dynamic and competitive landscape.