naga_pavan_looklikemodel

February 2, 2025

[38]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

[39]:
```
customers = pd.read_csv("C:/Users/Naga/OneDrive/Desktop/intern/Customers.csv")
products = pd.read_csv("C:/Users/Naga/OneDrive/Desktop/intern/Products.csv")
transactions = pd.read_csv("C:/Users/Naga/OneDrive/Desktop/intern/Transactions.
 ↪csv")
```

[40]:
```
customer_transactions = transactions.groupby("CustomerID")[["Quantity",␣
 ↪"TotalValue"]].sum().reset_index()
customer_data = customers.merge(customer_transactions, on="CustomerID",␣
 ↪how="left").fillna(0)
```

[42]:
```
customer_data['Region'] = LabelEncoder().fit_transform(customer_data['Region'])
```

[43]:
```
X = customer_data[['Region', 'Quantity', 'TotalValue']]
y = customer_data['CustomerID'].astype('category').cat.codes
```

[44]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
X_test_df=pd.DataFrame(X_test,columns=X_train.columns)
```

[45]:
```
logistic_model = LogisticRegression(max_iter=1000)
logistic_model.fit(X_train, y_train)
```

```
C:\Users\Naga\AppData\Local\Programs\Python\Python313\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(
```

[45]:
```
LogisticRegression(max_iter=1000)
```

[47]:
```
y_pred = logistic_model.predict(X_test_df)
accuracy = accuracy_score(y_test, y_pred)
print("Predicted Values: ",y_pred)
print("Logistic Regression Model Accuracy:", accuracy)
```

```
Predicted Values:  [167 167  24  44 191  38 155 187  83  64  86  81   7 187 167
53  64 101
  31  64 107  83  64 108 195 167 157 155 167 190 108  74  44   2  31  64
 133  53   3  31]
Logistic Regression Model Accuracy: 0.0
```

[28]:
```
lookalike_results = {}
```

[33]:
```
for i in range(20):
    cust_id = customer_data.iloc[i]['CustomerID']
    cust_features = np.array([customer_data.iloc[i][['Region', 'Quantity',␣
 ↪'TotalValue']]]).reshape(1,-1)
```

[48]:
```
 cust_features_df = pd.DataFrame(cust_features, columns=X_train.columns)
```

[49]:
```
probs = logistic_model.predict_proba(cust_features_df)
top_3_similar = np.argsort(probs[0])[-3:]
similar_customers = [(customer_data.iloc[idx]['CustomerID'], probs[0][idx]) for␣
 ↪idx in top_3_similar]
lookalike_results[cust_id] = similar_customers
```

[51]:
```
print("Lookalike Customers:", lookalike_results)
```

```
Lookalike Customers: {'C0020': [('C0075', np.float64(0.027224801660454222)),
('C0120', np.float64(0.029608276887340745)), ('C0104',
np.float64(0.030386607645596943))]}
```