Multinomial Logistic Regression

Machine Learning for Artificial Intelligence

Assessment 4

MAHDSE221F-030 R. H. I. Himanshi

MAHDSE221F-037 B. J. Manawaduge

NATIONAL INSTITUTE OF BUSINESS MANAGEMENT

DESCRIBE THE SOURCE CODE

- **import pandas as pd**

  from sklearn.linear_model import LogisticRegression

  from sklearn.model_selection import train_test_split

  from sklearn.preprocessing import LabelEncoder

  Importing Libraries: The code starts by importing the necessary libraries: pandas for data manipulation, LogisticRegression from sklearn.linear_model for building the model, train_test_split from sklearn.model_selection for splitting the data into training and testing sets, and LabelEncoder from sklearn.preprocessing for encoding categorical variables.

- **#Read the dataset**
  data = pd.read_csv("New 1000 Sales Records.csv")

  Loading the Data: The code reads the sales data from a CSV file named "New 1000 Sales Records.csv" using the pd.read_csv function. The data is stored in a DataFrame called data.

- **#Split the dataset into features(X) and target variable(y)**
  X = data[['Region','Total Revenue','Total Profit']]
  y = data['Item Type']

  Feature and Target Setup: The features (independent variables) for the model are selected. In this case, the features are 'Region', 'Total Revenue', and 'Total Profit', which are extracted from the DataFrame data. The target variable (dependent variable) is 'Item Type', which represents the preferred food type.

- **#Encode categorical variables into numerical representation**

  label_encoder = LabelEncoder()

X['Region'] = label_encoder.fit_transform(X['Region'])

Label Encoding: Categorical variables like 'Region' need to be converted into a numerical format for the model to work. The LabelEncoder is used to transform the categorical values in the 'Region' column into numerical labels. The updated 'Region' column is stored in the DataFrame X.

- **#Spit the data into training and testing sets**

  X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=42)

  Train-Test Split: The data is split into training and testing sets using the train_test_split function. The features are split into X_train and X_test, and the target variable 'Item Type' is split into y_train and y_test. The parameter test_size=0.2 indicates that 20% of the data will be used for testing, and random_state=42 ensures reproducibility.

- **#create and train the multinomial logistic regression model**

  model = LogisticRegression(multi_class = 'multinomial', solver = 'lbfgs')

  model.fit(X_train, y_train)

  Model Creation and Training: A multinomial logistic regression model is created using the LogisticRegression class. The parameter multi_class='multinomial' specifies that the model will handle multiple classes, and solver='lbfgs' is the optimization algorithm used. The model is then trained on the training data using the fit method.

- **#Predict the preffered food type for test data**

  y_pred = model.predict(X_test)

  print("X_test:", X_test)

  print("y_pred:", y_pred)

Prediction: The model is used to predict the preferred food types for the test data (X_test) using the predict method. The predicted food types are stored in the array y_pred.

Printing Predictions: The code prints the test data (X_test) and the corresponding predicted food types (y_pred).

- **#Evaluation the model's accuracy**

  accuracy = (y_pred == y_test).mean()

  print("Accuracy:", accuracy)

  Model Evaluation: The accuracy of the model is evaluated by comparing the predicted food types (y_pred) with the actual food types from the test set (y_test). The accuracy is calculated as the ratio of correct predictions to the total number of predictions.

SOURCE CODE

```
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

#Read the dataset
data = pd.read_csv("New 1000 Sales Records.csv")

#Split the dataset into features(X) and target variable(y)
X = data[['Region','Total Revenue','Total Profit']]
y = data['Item Type']

#Encode categorical variables into numerical representation
```

label_encoder = LabelEncoder()

X['Region'] = label_encoder.fit_transform(X['Region'])

#Spit the data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=42)

#create and train the multinomial logistic regression model

model = LogisticRegression(multi_class = 'multinomial', solver = 'lbfgs')

model.fit(X_train, y_train)

#Predict the preffered food type for test data

y_pred = model.predict(X_test)

print("X_test:", X_test)

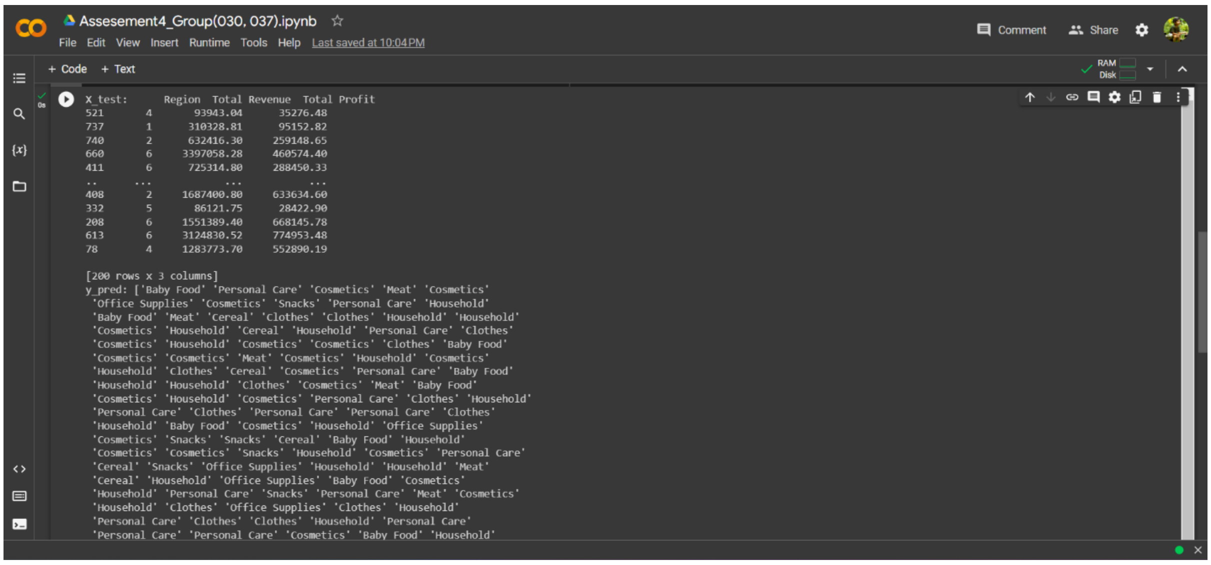
print("y_pred:", y_pred)

#Evaluation the model's accuracy

accuracy = (y_pred == y_test).mean()

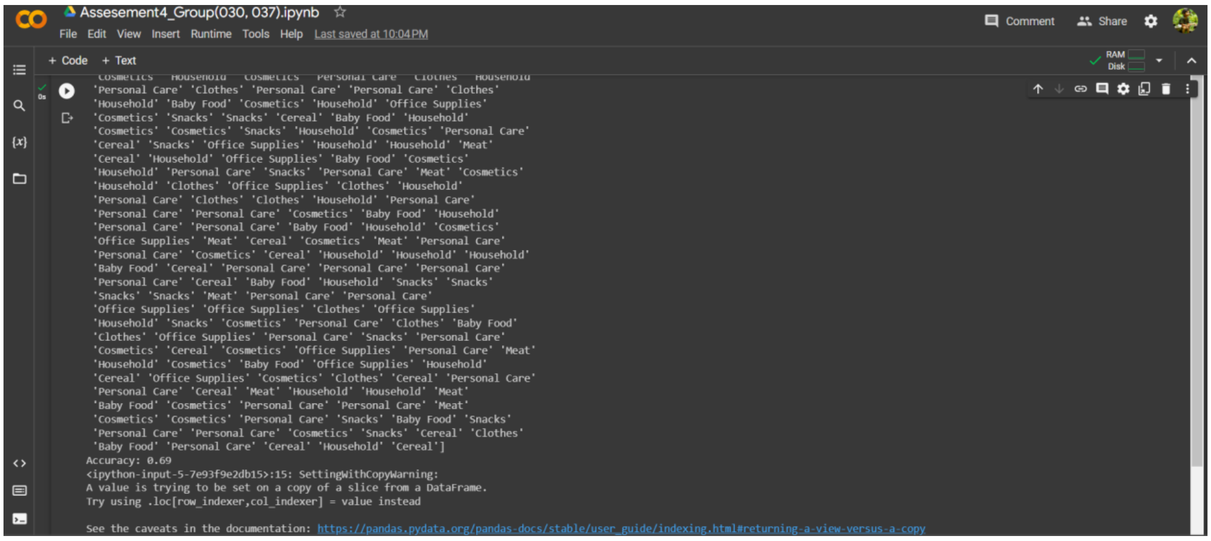
print("Accuracy:", accuracy)

OUTPUTS


Assesement4_Group(030, 037).ipynb
File Edit View Insert Runtime Tools Help Last saved at 10:04 PM
Comment Share
+ Code + Text
RAM Disk

```
X_test:      Region  Total Revenue  Total Profit
521       4       93943.04      35276.48
737       1      310328.81      95152.82
740       2      632416.30     259148.65
660       6     3397058.28     460574.40
411       6      725314.80     288450.33
..      ...            ...           ...
408       2     1687400.80     633634.60
332       5       86121.75      28422.90
208       6     1551389.40     668145.78
613       6     3124830.52     774953.48
78        4     1283773.70     552890.19

[200 rows x 3 columns]
y_pred: ['Baby Food' 'Personal Care' 'Cosmetics' 'Meat' 'Cosmetics'
 'Office Supplies' 'Cosmetics' 'Snacks' 'Personal Care' 'Household'
 'Baby Food' 'Meat' 'Cereal' 'Clothes' 'Clothes' 'Household' 'Household'
 'Cosmetics' 'Household' 'Cereal' 'Household' 'Personal Care' 'Clothes'
 'Cosmetics' 'Household' 'Cosmetics' 'Cosmetics' 'Clothes' 'Baby Food'
 'Cosmetics' 'Cosmetics' 'Meat' 'Cosmetics' 'Household' 'Cosmetics'
 'Household' 'Clothes' 'Cereal' 'Cosmetics' 'Personal Care' 'Baby Food'
 'Household' 'Household' 'Clothes' 'Cosmetics' 'Meat' 'Baby Food'
 'Cosmetics' 'Household' 'Cosmetics' 'Personal Care' 'Clothes' 'Household'
 'Personal Care' 'Clothes' 'Personal Care' 'Personal Care' 'Clothes'
 'Household' 'Baby Food' 'Cosmetics' 'Household' 'Office Supplies'
 'Cosmetics' 'Snacks' 'Snacks' 'Cereal' 'Baby Food' 'Household'
 'Cosmetics' 'Cosmetics' 'Snacks' 'Household' 'Cosmetics' 'Personal Care'
 'Cereal' 'Snacks' 'Office Supplies' 'Household' 'Household' 'Meat'
 'Cereal' 'Household' 'Office Supplies' 'Baby Food' 'Cosmetics'
 'Household' 'Personal Care' 'Snacks' 'Personal Care' 'Meat' 'Cosmetics'
 'Household' 'Clothes' 'Office Supplies' 'Clothes' 'Household'
 'Personal Care' 'Clothes' 'Clothes' 'Household' 'Personal Care'
 'Personal Care' 'Personal Care' 'Cosmetics' 'Baby Food' 'Household'
```


Assesement4_Group(030, 037).ipynb
File Edit View Insert Runtime Tools Help Last saved at 10:04 PM
Comment
Share
+ Code + Text
RAM
Disk

```
 'Personal Care' 'Clothes' 'Personal Care' 'Personal Care' 'Clothes'
 'Household' 'Baby Food' 'Cosmetics' 'Household' 'Office Supplies'
 'Cosmetics' 'Snacks' 'Snacks' 'Cereal' 'Baby Food' 'Household'
 'Cosmetics' 'Cosmetics' 'Snacks' 'Household' 'Cosmetics' 'Personal Care'
 'Cereal' 'Snacks' 'Office Supplies' 'Household' 'Household' 'Meat'
 'Cereal' 'Household' 'Office Supplies' 'Baby Food' 'Cosmetics'
 'Household' 'Personal Care' 'Snacks' 'Personal Care' 'Meat' 'Cosmetics'
 'Household' 'Clothes' 'Office Supplies' 'Clothes' 'Household'
 'Personal Care' 'Clothes' 'Clothes' 'Household' 'Personal Care'
 'Personal Care' 'Personal Care' 'Cosmetics' 'Baby Food' 'Household'
 'Personal Care' 'Personal Care' 'Baby Food' 'Household' 'Cosmetics'
 'Office Supplies' 'Meat' 'Cereal' 'Cosmetics' 'Meat' 'Personal Care'
 'Personal Care' 'Cosmetics' 'Cereal' 'Household' 'Household' 'Household'
 'Baby Food' 'Cereal' 'Personal Care' 'Personal Care' 'Personal Care'
 'Personal Care' 'Cereal' 'Baby Food' 'Household' 'Snacks' 'Snacks'
 'Snacks' 'Snacks' 'Meat' 'Personal Care' 'Personal Care'
 'Office Supplies' 'Office Supplies' 'Clothes' 'Office Supplies'
 'Household' 'Snacks' 'Cosmetics' 'Personal Care' 'Clothes' 'Baby Food'
 'Clothes' 'Office Supplies' 'Personal Care' 'Snacks' 'Personal Care'
 'Cosmetics' 'Cereal' 'Cosmetics' 'Office Supplies' 'Personal Care' 'Meat'
 'Household' 'Cosmetics' 'Baby Food' 'Office Supplies' 'Household'
 'Cereal' 'Office Supplies' 'Cosmetics' 'Clothes' 'Cereal' 'Personal Care'
 'Personal Care' 'Cereal' 'Meat' 'Household' 'Household' 'Meat'
 'Baby Food' 'Cosmetics' 'Personal Care' 'Personal Care' 'Meat'
 'Cosmetics' 'Cosmetics' 'Personal Care' 'Snacks' 'Baby Food' 'Snacks'
 'Personal Care' 'Personal Care' 'Cosmetics' 'Snacks' 'Cereal' 'Clothes'
 'Baby Food' 'Personal Care' 'Cereal' 'Household' 'Cereal']
Accuracy: 0.69
<ipython-input-5-7e93f9e2db15>:15: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Accuracy: 0.69

Accuracy: The accuracy of the model's predictions is calculated by comparing the predicted preferred food types (y_pred) with the actual preferred food types from the test set (y_test). The accuracy is approximately 69%, indicating that the model correctly predicted the preferred food type for about 69% of the data points in the test set.