

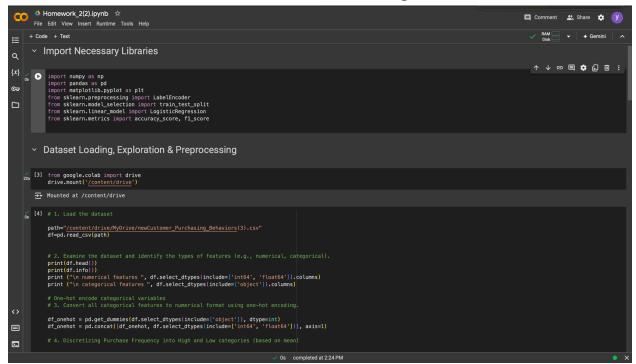
## Machine Learning HW2

Yara Alfouzan 443203087

## **DATA SET DESCRIPTION:**

Customer Purchasing Behaviors dataset. The specific features included are userr\_id, age, annual\_income, purchase\_amount, purchase\_frequency, region, and loyalty\_score. This dataset can be used to analyze and predict customer purchasing behaviors. In this hw it'll be used to predict whether a customer makes a high or low frequency of purchases.

## Clear screenshots of the code, evaluations and outputs:



```
mean_purchase_frequency = df_onehot['purchase_frequency'].mean()
df_onehot['purchase_frequency_category'] = pd.cut(df['purchase_frequency'], bins=[0, mean_purchase_frequency, float('inf')], labels=['Low', 'High'])
print(df_onehot.head())

# 5. Select purchase_frequency_category as the target variable (y) and use all other features except (purchase_frequency) as the input features (X).

y = df_onehot['purchase_frequency_category']
X = df_onehot.drop(['purchase_frequency_category'], axis=1)

# 6. Split the dataset into training (80%) and testing (20%) sets.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
Categorical features Index(['region]', dtypes'object')

region, least region, looth lo
```

```
| S | S | Calculate accuracy and f1 score | accuracy score(y_test_numeric, y_pred) | f1 = f1_score(y_test_numeric, y_pred_sklearn) | f1_sklearn = f1_score(y_test_numeric, y_p_test_numeric, y_p_test_numeric, y_p_test_numeric, y_p_test_nume
```

```
Feature Engineering and Re-evaluation

# 1. Create a new feature by squaring the purchase amount feature to capture a potential non-linear relationship.

# 2. Re-split the data to include the new feature.

# 2. Re-split the data to include the new feature.

# 3. Re-split the data to include the new feature.

# 3. Fit a new sklearn (pajstic regression model using the original features plus the new quadratic feature.

# 3. Fit a new sklearn (pajstic regression model using the original features plus the new quadratic feature.

# 3. Fit a new sklearn (padratic = Logistic Regression (max_lter=1800))

# 4. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 4. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 5. Score (Scikit-learn): ", accuracy_sklearn_quadratic, predict(X_test2))

# 6. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 7. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 7. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 8. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 9. Predict the test data and calculate Accuracy and F1 score for the new sklearn model.

# 1. Score (Scikit-learn): ", accuracy_sklearn quadratic)

# 1. Score (Scikit-learn): ", f1. Sklearn)

# Accuracy (Scikit-learn): ", f1. Sklearn)
```

# A table displaying printed results (e.g., model parameters and/or predicted values):

#### F1 score and accuracy.

Logistic regression using normal eq:

```
© 37 Accuracy: 8,5625
D F1 Score: 8,72
```

## Logistic regression using Sklearn:

```
    Accuracy (Sklearn): 0.9375
    F1 Score (Sklearn): 0.943362264158944
    //usr/loca/l/lipython3.180/dist-packages/sklearn/linear_model/_logistic.py:469; ConvergenceWarning: lbfgs failed to converge (status=1):
```

Logistic regression using Sklearn after feature engineering:

#### Part5: Observation & Comparison

### 1. Explain why one-hot encoding is necessary in the context of linear regression

Linear regression models work best with numerical data. One-hot encoding is necessary to convert categorical data into a numerical format that the model can understand; This prevents the model from misinterpreting the categorical data as ordinal, ensuring that the model treats each category as a distinct entity without any implied order or relationship between them.

## 1. Compare the performance of the model with and without the new polynomial feature

The model with the squared feature performs better than the model without it.

Accuracy: Increased from 0.9375 to 1.0

F1 Score: Increased from 0.9434 to 1.0

This indicates that the squared feature likely captures a non-linear relationship between purchase amount and purchase frequency, which the original model could not capture. As a result, the model with the squared feature is able to make perfect predictions on the test set;but observing perfect accuracy and F1 scores, might indicate overfitting.

## 2. Provide observations on which model performs better and why.

The model implemented with scikit-learn performs significantly better than the model using the normal equation.

Accuracy: 0.9375 (scikit-learn) vs. 0.5625 (normal equation)

F1 Score: 0.9434 (scikit-learn) vs. 0.72 (normal equation)

This difference in performance could be due to that Scikit-learn's LogisticRegression includes regularization by default; moreover it likely uses more robust optimization algorithms that are less susceptible to numerical issues rather than solving normal equation directly.