Name:Bhushan Malpani Roll No:33 Div:D15C

Aim: Perform Regression Analysis using Scipy and Sci-kit learn.

Problem Statement:
a) Perform Logistic regression to find out relation between variables
b) Apply regression model technique to predict the data on above dataset.

Dataset Description

The dataset contains 100,000 records with 14 attributes, focusing on dietary habits, health conditions, and lifestyle.

Key Features:

- Demographics: Age, Gender, Height (cm), Weight (kg), BMI
- Lifestyle & Health: Activity Level, Health Conditions, Dietary Preferences
- Dietary Data: Recommended Meals, Calories Intake
- Meals: Breakfast, Lunch, Snacks, Dinner

Step 1: Load the Dataset

Upload your dataset to Google Colab, then read it using pandas.

```
from google.colab import files

file_path =S
df = pd.read_csv(file_path)
print(df.info())
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 14 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   Age                 100000 non-null  int64
 1   Gender              100000 non-null  object
 2   Height_cm           100000 non-null  float64
 3   Weight_kg           100000 non-null  float64
 4   BMI                 100000 non-null  float64
 5   Activity_Level      100000 non-null  object
 6   Health_Conditions   79945 non-null   object
 7   Dietary_Preferences 100000 non-null  object
 8   Recommended_Meals   100000 non-null  object
 9   Calories_Intake     100000 non-null  int64
 10  Breakfast           100000 non-null  object
 11  Lunch               100000 non-null  object
 12  Snacks              100000 non-null  object
 13  Dinner              100000 non-null  object
dtypes: float64(3), int64(2), object(9)
memory usage: 10.7+ MB
None
```

	Age	Gender	Height_cm	Weight_kg	BMI	Activity_Level	Health_Conditions	Dietary_Preferences	Recommended_Meals	Calories_Intake	Breakfast	Lunch	Snacks	Dinner
0	56	Other	189.552819	76.388643	39.762724	Sedentary	Heart Disease	Keto	Nuts, Yogurt	2352	Grilled Paneer, Vegetables	Oats, Fruits	Nuts, Yogurt	Salad, Brown Rice
1	46	Other	144.649993	101.391668	15.588831	Active	Diabetes	Paleo	Nuts, Yogurt	2298	Salad, Brown Rice	Grilled Paneer, Vegetables	Salad, Brown Rice	Oats, Fruits
2	32	Female	158.142263	54.060753	19.146735	Active	Obesity	Vegetarian	Oats, Fruits	2021	Oats, Fruits	Oats, Fruits	Oats, Fruits	Grilled Paneer, Vegetables
3	60	Other	174.077462	97.286598	37.659330	Very Active	Heart Disease	Vegetarian	Nuts, Yogurt	2124	Oats, Fruits	Oats, Fruits	Grilled Paneer, Vegetables	Salad, Brown Rice
4	25	Female	144.947456	105.377330	16.587306	Sedentary	Heart Disease	Vegetarian	Salad, Brown Rice	2289	Nuts, Yogurt	Grilled Paneer, Vegetables	Salad, Brown Rice	Nuts, Yogurt

Step 2: Preprocess the Data

Convert categorical variables into numerical form using LabelEncoder.

```
df.info()
df.describe()
df.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 14 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   Age                  100000 non-null  int64  
 1   Gender               100000 non-null  object 
 2   Height_cm            100000 non-null  float64
 3   Weight_kg            100000 non-null  float64
 4   BMI                  100000 non-null  float64
 5   Activity_Level       100000 non-null  object 
 6   Health_Conditions    79945 non-null   object 
 7   Dietary_Preferences  100000 non-null  object 
 8   Recommended_Meals    100000 non-null  object 
 9   Calories_Intake      100000 non-null  int64  
 10  Breakfast            100000 non-null  object 
 11  Lunch                100000 non-null  object 
 12  Snacks               100000 non-null  object 
 13  Dinner               100000 non-null  object 
dtypes: float64(3), int64(2), object(9)
memory usage: 10.7+ MB
```

	0
Age	0
Gender	0
Height_cm	0
Weight_kg	0
BMI	0
Activity_Level	0
Health_Conditions	20055
Dietary_Preferences	0
Recommended_Meals	0
Calories_Intake	0
Breakfast	0
Lunch	0
Snacks	0
Dinner	0

dtype: int64

Step 3: Perform Logistic Regression

Logistic Regression helps determine the relationship between Health Conditions and other feature.

```
label_encoders = {}
categorical_cols = ['Gender', 'Activity_Level', 'Health_Conditions', 'Dietary_Preferences']

for col in categorical_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le
```

Classification of dataset

```
X_classification = df[['Age', 'Gender', 'Height_cm', 'Weight_kg', 'BMI', 'Activity_Level', 'Health_Conditions']]
y_classification = df['Dietary_Preferences']
```

Training dataset and testing accuracy

```
X_train, X_test, y_train, y_test = train_test_split(X_classification, y_classification, test_size=0.2, random_state=42)
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
log_reg = LogisticRegression(max_iter=500)
log_reg.fit(X_train, y_train)
```

```
LogisticRegression
LogisticRegression(max_iter=500)
```

```
y_pred = log_reg.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Classification Report:\n", classification_rep)
```

```
Accuracy: 0.2016
Classification Report:
              precision    recall  f1-score   support

           0       0.20      0.12      0.15      4000
           1       0.19      0.20      0.20      3953
           2       0.21      0.28      0.24      4118
           3       0.21      0.12      0.16      4008
           4       0.20      0.28      0.23      3921

    accuracy                           0.20     20000
   macro avg       0.20      0.20      0.20     20000
weighted avg       0.20      0.20      0.20     20000
```

Step 5: Perform Linear Regression

We'll predict Calories_Intake using a regression model.

Name:Bhushan Malpani Roll No:33 Div:D15C

```
X_regression = df[['Age', 'Gender', 'Height_cm', 'Weight_kg', 'BMI', 'Activity_Level', 'Health_Conditions']]
y_regression = df['Calories_Intake']
```

```
X_train, X_test, y_train, y_test = train_test_split(X_regression, y_regression, test_size=0.2, random_state=42)
```

```
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
y_pred = lin_reg.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"MAE: {mae}")
print(f"RMSE: {rmse}")
print(f"R² Score: {r2}")
```

```
MAE: 175.00294674090316
RMSE: 202.0365965061598
R² Score: -0.00024337578128275084
```

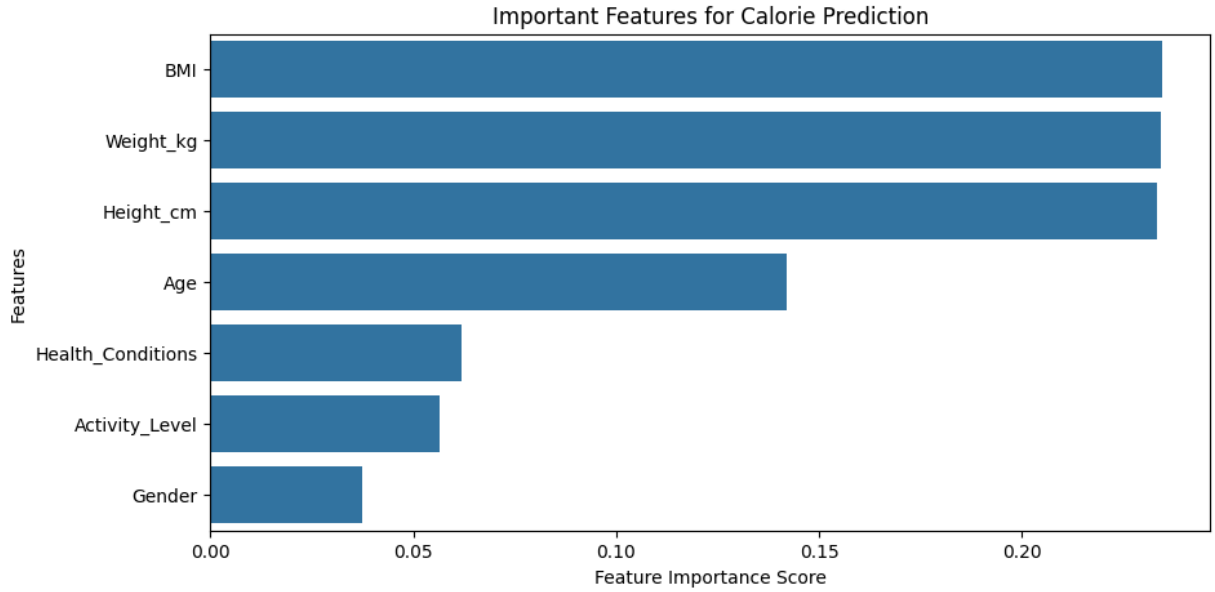
Using random Forest algo:

```
from sklearn.ensemble import RandomForestRegressor

rf_reg = RandomForestRegressor(n_estimators=100, random_state=42)
rf_reg.fit(X_train, y_train)

y_pred_rf = rf_reg.predict(X_test)

mae_rf = mean_absolute_error(y_test, y_pred_rf)
rmse_rf = np.sqrt(mean_squared_error(y_test, y_pred_rf))
r2_rf = r2_score(y_test, y_pred_rf)

print(f"Random Forest MAE: {mae_rf}")
print(f"Random Forest RMSE: {rmse_rf}")
print(f"Random Forest R² Score: {r2_rf}")
```

```
Random Forest MAE: 176.86412249999995
Random Forest RMSE: 205.31399015109758
Random Forest R² Score: -0.032958046218887205
```

Step 6: Visualize the Regression Predictions

```
feature_importance = pd.Series(rf_reg.feature_importances_, index=X_regression.columns).sort_values(ascending=False)

plt.figure(figsize=(10, 5))
sns.barplot(x=feature_importance, y=feature_importance.index)
plt.xlabel("Feature Importance Score")
plt.ylabel("Features")
plt.title("Important Features for Calorie Prediction")
plt.show()
```



Conclusion

1. We successfully performed Logistic Regression to analyze the relationship between variables and classify Health_Conditions.
2. The accuracy of Logistic Regression was low (20.2%), indicating that the dataset may require better feature selection or more advanced models.
3. We applied Linear Regression to predict Calories_Intake, but the R^2 score was nearly zero, suggesting weak predictive power.
4. The Mean Absolute Error (MAE) was 175 calories, meaning the predictions had significant deviations.
5. Feature scaling, transformation, or using non-linear models like Random Forest could improve regression performance.
6. Visualization of predicted vs actual values showed a large variance, confirming the model's limitations.
7. Future work can focus on feature engineering and advanced ML models to enhance prediction accuracy.