Sardar Patel Institute of Technology, Mumbai
Department of Electronics and Telecommunication Engineering
T.E.-CSE (2023-2024)
PC 307 - Machine Learning

Name: Deepali Daga **Date:19-03-24**
UID: 2021600012 **Batch: A**

Experiment 4: LOGISTIC REGRESSION

AIM/OBJECTIVE: Implement Binary and Multiclass logistic regression technique using open-source software**.**

Outcomes:

- **Explore the Dataset suitable for Binary and multiclass regression problem**
- **Explore the pattern from the dataset and apply suitable algorithm**

System Requirements:

Linux OS with Python and libraries or R or windows with MATLAB

Theory:

Logistic regression is a supervised machine learning algorithm that accomplishes binary classification tasks by predicting the probability of an outcome, event, or observation. The model delivers a binary or dichotomous outcome limited to two possible outcomes: yes/no, 0/1, or true/false.
The sigmoid function is referred to as an activation function for logistic regression and is defined as:
f(x) = 1 / 1 + e^ -x
where,
e = base of natural logarithms
value = numerical value one wishes to transform
The following equation represents logistic regression Sigmoid Function:
y = e^(b0 + b1x) / 1+e(b0 + b1x)
here,
x = input value
y = predicted output
b0 = bias or intercept term
b1 = coefficient for input (x)

Part I: Binary Logistic Regression
Logistic regression is a statistical method used for binary classification tasks, where the goal is to predict the probability of a binary outcome based on one or more predictor variables. It models the probability of the dependent variable y given the independent variables x. The logistic regression model uses the logistic function, also known as the sigmoid function, to model this relationship.
The logistic function is defined as:
P = 1/ [e^(-β0 - β1*x)]

Part II: Multiclass Logistic Regression
Logistic regression can be extended to handle multiclass classification using the one-vs-rest (one-vs-all) approach. In this approach, a separate logistic regression model is trained for each class, where each model distinguishes that class from all other classes. During prediction, the model with the highest predicted probability is selected as the final class. This technique allows logistic regression, which inherently solves binary classification problems, to be applied to multiclass classification tasks effectively.

dataset link and description:
Part 1 :

The dataset consists of information about users who are potential customers for a product or service. It contains Salary - which is used to predict whether or not the user purchased the product, indicated by the output or target column 'Purchased'.

A	B	C
Salary	Purchased	
19000	0	
20000	0	
43000	0	
57000	0	
76000	0	
58000	0	
84000	0	
150000	1	
33000	0	
65000	0	
80000	0	
52000	0	
86000	0	
18000	0	
82000	0	
80000	0	
25000	1	
26000	1	
28000	1	
29000	1	
22000	1	
49000	1	
41000	1	
22000	1	
23000	1	
20000	1	

=
Part 2 :

The Iris dataset was used in R.A. Fisher's classic 1936 paper, The Use of Multiple Measurements in Taxonomic Problems, and can also be found on the UCI Machine Learning Repository.

It includes three iris species with 50 samples each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly separable from each other.

The columns in this dataset are:

Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species

	A	B	C	D	E	F
	Id	SepalLengt	SepalWidt	PetalLengt	PetalWidt	Species
	1	5.1	3.5	1.4	0.2	Iris-setosa
	2	4.9	3	1.4	0.2	Iris-setosa
	3	4.7	3.2	1.3	0.2	Iris-setosa
	4	4.6	3.1	1.5	0.2	Iris-setosa
	5	5	3.6	1.4	0.2	Iris-setosa
	6	5.4	3.9	1.7	0.4	Iris-setosa
	7	4.6	3.4	1.4	0.3	Iris-setosa
	8	5	3.4	1.5	0.2	Iris-setosa
0	9	4.4	2.9	1.4	0.2	Iris-setosa
1	10	4.9	3.1	1.5	0.1	Iris-setosa
2	11	5.4	3.7	1.5	0.2	Iris-setosa
3	12	4.8	3.4	1.6	0.2	Iris-setosa
4	13	4.8	3	1.4	0.1	Iris-setosa
5	14	4.3	3	1.1	0.1	Iris-setosa
6	15	5.8	4	1.2	0.2	Iris-setosa
7	16	5.7	4.4	1.5	0.4	Iris-setosa
8	17	5.4	3.9	1.3	0.4	Iris-setosa
9	18	5.1	3.5	1.4	0.3	Iris-setosa
0	19	5.7	3.8	1.7	0.3	Iris-setosa
1	20	5.1	3.8	1.5	0.3	Iris-setosa
2	21	5.4	3.4	1.7	0.2	Iris-setosa
3	22	5.1	3.7	1.5	0.4	Iris-setosa
4	23	4.6	3.6	1	0.2	Iris-setosa
5	24	5.1	3.3	1.7	0.5	Iris-setosa
6	25	4.8	3.4	1.9	0.2	Iris-setosa
7	26	5	3	1.6	0.2	Iris-setosa

Program: Part 1 : Binary Logistic Regression

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


df_salary_Purchased = pd.read_csv('User_Data.csv')

df_salary_Purchased.head()


df_salary_Purchased.columns = ['Salary', 'Purchased']

df_salary_Purchased.head()


plt.figure(figsize=(8, 6))
sns.countplot(x=df_salary_Purchased['Purchased'])
plt.title('Distribution of Purachase')
plt.xlabel('Purachase')
plt.ylabel('Count')
plt.show()

plt.figure(figsize=(8, 6))
sns.scatterplot(x=df_salary_Purchased['Salary'], y=df_salary_Purchased['Purchased'],
hue=df_salary_Purchased['Purchased'], s=100)
plt.title('Salary vs. Purachase')
plt.xlabel('Salary')
plt.ylabel('Purachase')
plt.legend(title='Purachase')
plt.show()


correlation_matrix = df_salary_Purchased.corr()

plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=.5)
plt.title("Correlation Matrix Heatmap")
plt.show()


mean = df_salary_Purchased.mean(axis=0)
print(mean)


size = len(df_salary_Purchased['Salary'])


mean_x = mean['Salary']
mean_y = mean['Purchased']
```

```
dev_x = df_salary_Purchased['Salary'] - mean_x
dev_y = df_salary_Purchased['Purchased'] - mean_y


prod_dev = dev_x * dev_y


sop = prod_dev.sum()


sq_dev_x = dev_x ** 2


m = sop / sq_dev_x.sum()
b = mean_y - (m * mean_x)


predicted_y = m * df_salary_Purchased['Salary'] + b


plt.plot(df_salary_Purchased['Salary'], predicted_y, color='red', label='Regression Line')


plt.xlabel('Salary')
plt.ylabel('Predicted Purachase')
plt.title('Linear Regression Example')
plt.legend()
plt.grid()

plt.scatter(df_salary_Purchased['Salary'], df_salary_Purchased['Purchased'])
plt.scatter(df_salary_Purchased['Salary'], predicted_y)


plt.show()

y_pred = [0]*size
for i in range(size):
    y_pred[i] = 1 / (1 + np.exp(-(b + m * df_salary_Purchased['Salary'][i])))


plt.figure(figsize=(8, 6))
plt.scatter(df_salary_Purchased['Salary'], y_pred, color='blue', label='Predicted Probabilities')


plt.xlabel('Salary')
plt.ylabel('Predicted Probability')
plt.title('Predicted Probabilities from Logistic Regression')
plt.legend()
plt.grid()
```

```
plt.show()
```

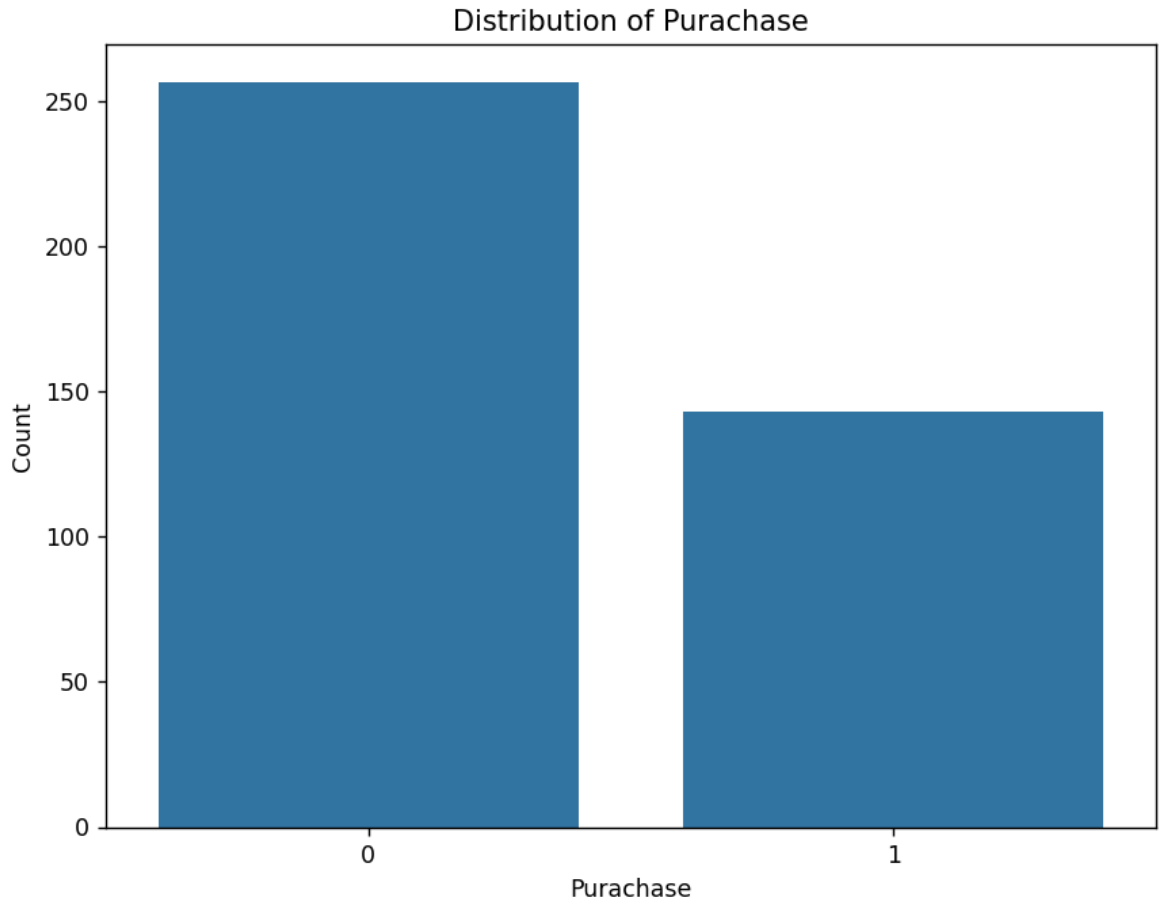
Program: Part 2 : Multiclass Logistic Regression

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder

data = pd.read_csv("Iris.csv")

label_encoder = LabelEncoder()
data['SpeciesEncoded'] = label_encoder.fit_transform(data['Species'])

X = data[['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']].values
y = data['SpeciesEncoded'].values
data.head()

X = (X - np.mean(X, axis=0)) / np.std(X, axis=0)
print(X)

X = np.hstack((np.ones((X.shape[0], 1)), X))
print(X)

def sigmoid(z):
    return 1 / (1 + np.exp(-z))

def softmax(z):
    exp_scores = np.exp(z)
    return exp_scores / np.sum(exp_scores, axis=1, keepdims=True)

def fit(X, y):
    num_features = X.shape[1]
    num_classes = len(np.unique(y))
    W = np.zeros((num_features, num_classes))

    for c in range(num_classes):
        y_one_vs_all = (y == c).astype(int)
        W[:, c] = np.linalg.pinv(X.T @ X) @ X.T @ y_one_vs_all

    return W

W = fit(X, y)

scores = np.dot(X, W)
predicted_class = np.argmax(scores, axis=1)
print(predicted_class)

accuracy = np.mean(predicted_class == y)
```

```
print(f'Accuracy: {accuracy}')

plt.figure(figsize=(12, 6))

x_values = np.linspace(np.min(X[:, 1]), np.max(X[:, 1]), 100)
for c in range(3):
    slope = -W[1, c] / W[2, c]
    intercept = -W[0, c] / W[2, c]
    plt.plot(x_values, slope * x_values + intercept, linestyle='--', label=f'Regression line for class {c}')

plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.title('Actual vs Predicted Classes with Regression Lines')
plt.legend()
plt.grid(True)
plt.show()
```

Output and Interpretation: Part I and Part II

Part I



```
Salary       69742.5000
Purchased        0.3575
```

Part II

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 2 2 1 2 1 1 2 1 1 1 2 2 2 2 2 2 1 2 1 2 1 1 1 1 2 2
 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
Accuracy: 0.8866666666666667
```

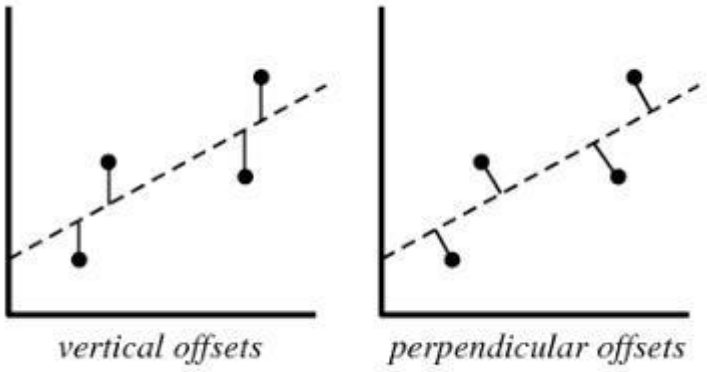
Conclusion:

In conclusion, logistic regression techniques provide efficient and interpretable methods for binary and multiclass classification tasks, allowing for probabilistic predictions and straightforward model interpretation.

Test your skill:

1. How a Multi variable regression technique different from linear regression
   Multivariable regression, also known as multiple regression, is an extension of simple linear regression that involves predicting a target variable using multiple independent variables, whereas linear regression involves predicting a target variable using only one independent variable. Model Complexity: Linear Regression: Simple linear regression models the relationship between two variables, making it less complex. Multivariable Regression: Multivariable regression models can capture more complex relationships between the dependent variable and multiple independent variables, allowing for a more nuanced analysis of the data. Dimensionality: Linear Regression: Represents a one-dimensional relationship between variables. Multivariable Regression: Represents a multidimensional relationship between variables, allowing for the consideration of multiple factors simultaneously.

2. Which of the following offsets, do we use in case of least square line fit? Suppose horizontal axis is independent variable and vertical axis is dependent variable. Justify.



A. Vertical offset
B. Perpendicular offset
C. Both but depend on situation
D. None of above

Least squares regression aims to minimize the sum of squared perpendicular distances (vertical offsets) between each observed data point and the corresponding point on the fitted line. This perpendicular distance represents the error between the observed data and the predicted values from the fitted line. Minimizing this perpendicular distance ensures that the line best fits the data by minimizing the overall error.

3. Why do we use multivariable regression models?

   We use multivariable regression models for several reasons: Handling Collinearity: Multivariable regression provides a framework for dealing with multicollinearity, which occurs when independent variables are highly correlated with each other. Techniques such as regularization can be employed to mitigate collinearity and improve model stability