LAB 3
Logistic REGRESSION:

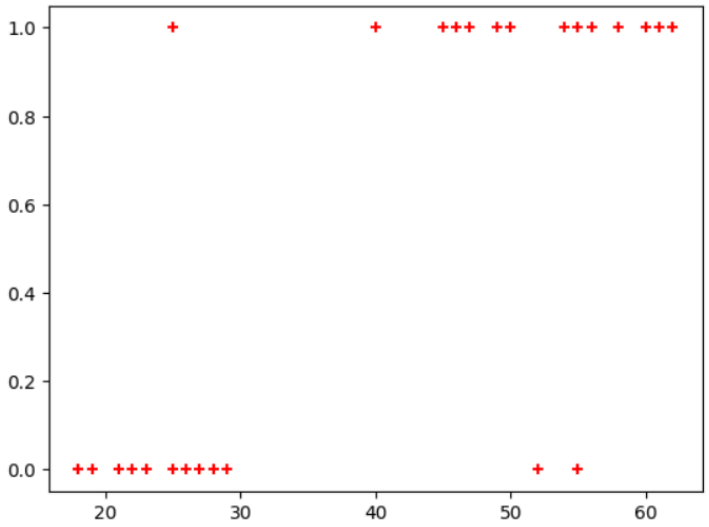
In [4]:
```
#Logistic_regression_binary
# Commented out IPython magic to ensure Python compatibility.
import pandas as pd
from matplotlib import pyplot as plt
from google.colab import files
# Upload the CSV file
uploaded = files.upload()
# Read the CSV file into a pandas DataFrame
df = pd.read_csv(next(iter(uploaded)))  # Load the first uploaded file
print(df.head())


plt.scatter(df.age,df.bought_insurance,marker='+',color='red')

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df[['age']],df.bought_insurance,train_size=0.9,random_state=10)
X_train.shape

X_test

from sklearn.linear_model import LogisticRegression
model = LogisticRegression()

model.fit(X_train, y_train)

X_test

y_test

y_predicted = model.predict(X_test)
y_predicted
```

```
model.predict_proba(X_test)

y_predicted = model.predict([[60]])
y_predicted

#model.coef_ indicates value of m in y=m*x + b equation
model.coef_

#model.intercept_ indicates value of b in y=m*x + b equation
model.intercept_

#Lets defined sigmoid function now and do the math with hand
import math
def sigmoid(x):
  return 1 / (1 + math.exp(-x))

def prediction_function(age):
   z = 0.127 * age - 4.973 # 0.12740563 ~ 0.0127 and -4.97335111 ~ -4.97
   y = sigmoid(z)
   return y

age = 35
prediction_function(age)

"""0.37 is less than 0.5 which means person with 35 will not buy the insurance"""
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving insurance_data.csv to insurance_data (1).csv
   age  bought_insurance
0   22                 0
1   25                 0
2   47                 1
3   52                 0
4   46                 1
```

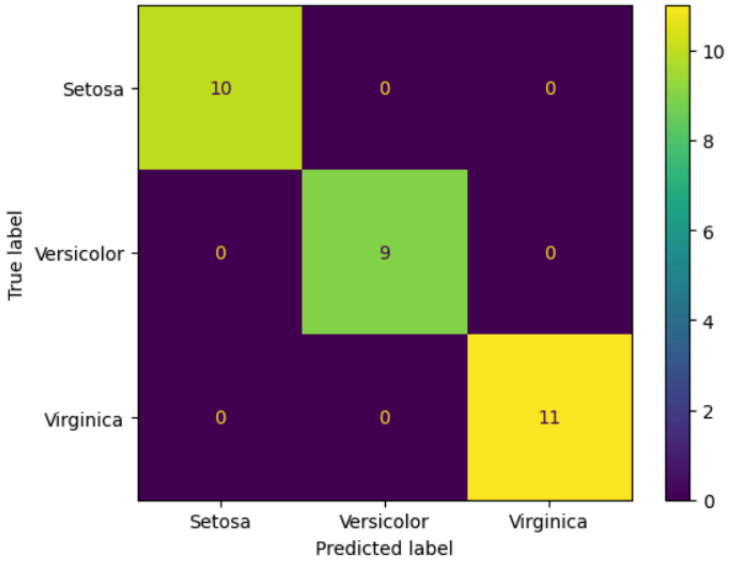
Out[4]: '0.37 is less than 0.5 which means person with 35 will not buy the insurance'



In [6]:

```
#logistic_regression_multiclass
# Import necessary libraries
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn import metrics
import matplotlib.pyplot as plt

from google.colab import files
# Upload the CSV file
uploaded = files.upload()
# Read the CSV file into a pandas DataFrame
iris = pd.read_csv(next(iter(uploaded)))  # Load the first uploaded file
print(iris.head())


X=iris.drop('species',axis='columns')# Features (sepal length, sepal width, petal length, petal width)
y = iris.species # Target labels (0: Setosa, 1: Versicolor, 2: Virginica)

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

# Initialize the Multinomial Logistic Regression model
# Use 'multinomial' for multi-class classification and 'lbfgs' solver
model = LogisticRegression(multi_class='multinomial')

# Train the model on the training data
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Calculate the accuracy of the model on the test data
accuracy = accuracy_score(y_test, y_pred)
```

```
Saving iris (1).csv to iris (1) (1).csv
   sepal_length  sepal_width  petal_length  petal_width      species
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa
Accuracy of the Multinomial Logistic Regression model on the test set: 1.00
/usr/local/lib/python3.11/dist-packages/sklearn/linear_model/_logistic.py:1247: FutureWarning: 'multi_class' was deprecated
in version 1.5 and will be removed in 1.7. From then on, it will always use 'multinomial'. Leave it to its default value to
avoid this warning.
  warnings.warn(
```



In [8]:

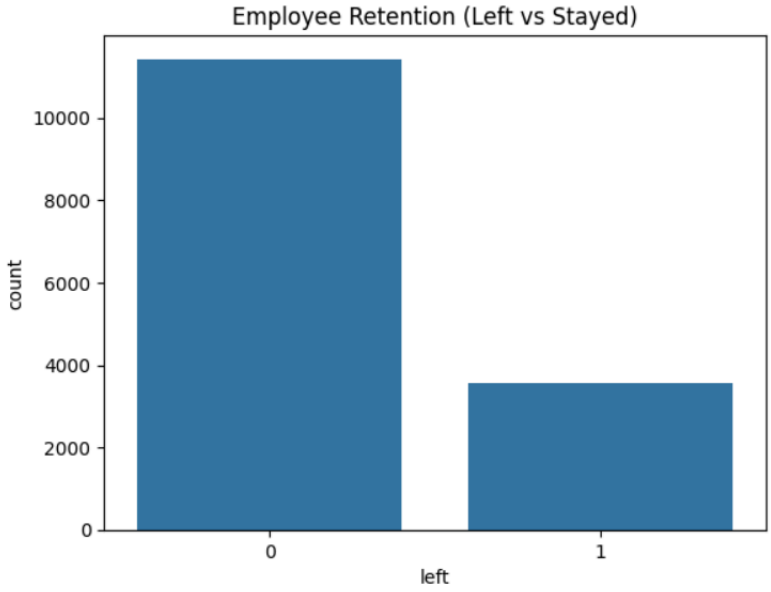
```python
#hr_comma_sep
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from google.colab import files

# Upload the CSV file
uploaded = files.upload()

# Read the CSV file into a pandas DataFrame
df = pd.read_csv(next(iter(uploaded)))  # Load the first uploaded file
print(df.head())

# Step 1: Exploratory Data Analysis (EDA)

# 1.1: Check for missing values
print("\nMissing Values:\n", df.isnull().sum())

# 1.2: Basic statistics of numerical variables
print("\nBasic Statistics:\n", df.describe())

# 1.3: Checking the distribution of the target variable 'left' (employee retention)
sns.countplot(x='left', data=df)
plt.title('Employee Retention (Left vs Stayed)')
plt.show()

# Step 2: Visualizations to explore relationships with retention

# 2.1: Impact of salary on retention
sns.countplot(x='salary', hue='left', data=df)
```

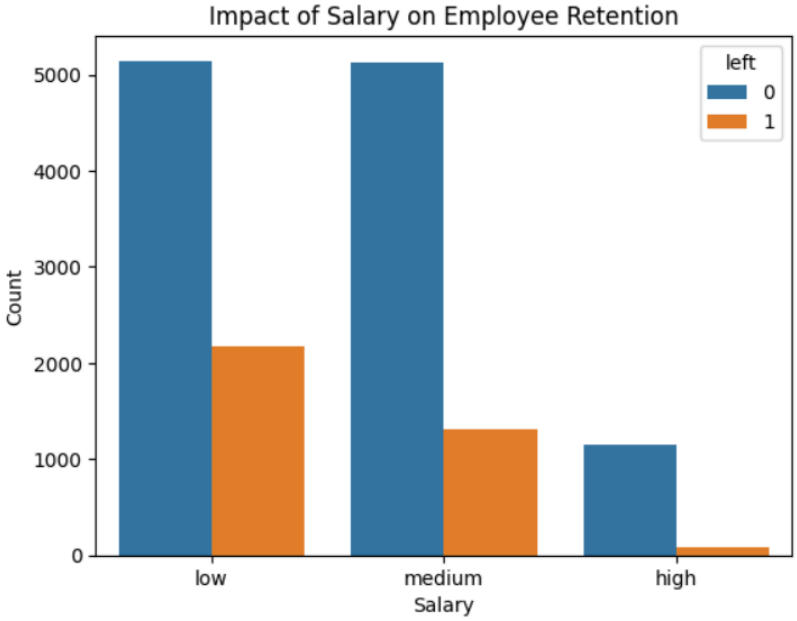
```
# 2.1: Impact of salary on retention
sns.countplot(x='salary', hue='left', data=df)
plt.title('Impact of Salary on Employee Retention')
plt.xlabel('Salary')
plt.ylabel('Count')
plt.show()

# 2.2: Impact of department on retention
sns.countplot(x='Department', hue='left', data=df)
plt.title('Impact of Department on Employee Retention')
plt.xlabel('Department')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()

# Step 3: Correlation Matrix (only for numeric columns)
numeric_df = df.select_dtypes(include=['number'])  # exclude non-numeric columns
corr = numeric_df.corr()
plt.figure(figsize=(10, 6))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()

# Step 4: Preparing the data for Logistic Regression

# 4.1: Encode categorical variables (salary, department)
df = pd.get_dummies(df, columns=['salary', 'Department'], drop_first=True)

# Features (X) and target (y)
X = df.drop(columns=['left'])
y = df['left']

# Step 5: Split 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)

# Step 6: Build the Logistic Regression model
model = LogisticRegression(max_iter=1000)
```

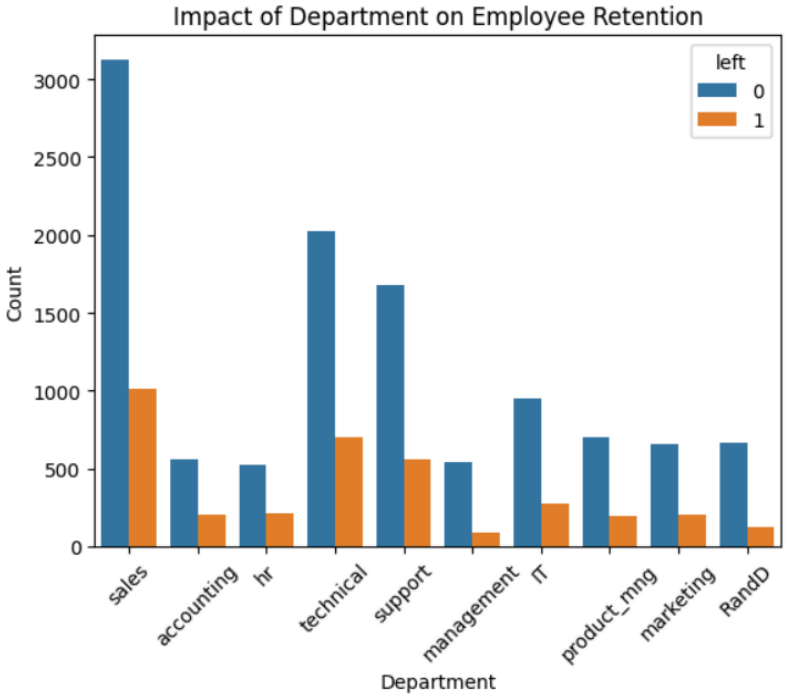
```
# Step 5: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 6: Build the Logistic Regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Step 7: Predicting and Evaluating the Model
y_pred = model.predict(X_test)

# Accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print(f'\nAccuracy of the Logistic Regression Model: {accuracy:.2f}')

# Confusion Matrix to evaluate model performance
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Stayed', 'Left'], yticklabels=['Stayed', 'Left'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

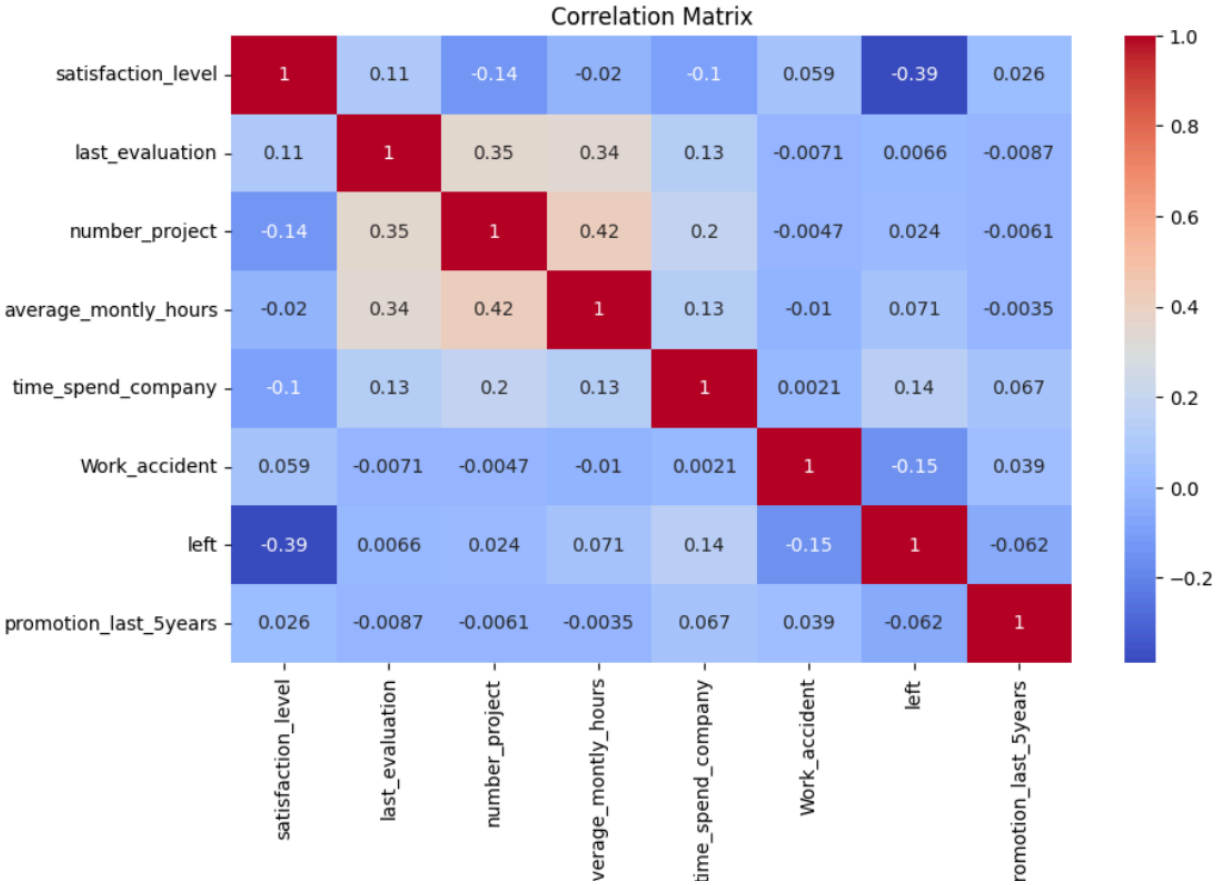
Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving HR_comma_sep.csv to HR_comma_sep (1).csv
   satisfaction_level  last_evaluation  number_project  average_montly_hours  \
0                0.38             0.53               2                   157
1                0.80             0.86               5                   262
2                0.11             0.88               7                   272
3                0.72             0.87               5                   223
4                0.37             0.52               2                   159

   time_spend_company  Work_accident  left  promotion_last_5years Department  \
0                   3              0     1                      0      sales
1                   6              0     1                      0      sales
2                   4              0     1                      0      sales
3                   5              0     1                      0      sales
4                   3              0     1                      0      sales

   salary
0     low
1  medium
2  medium
3     low
4     low

Missing Values:
 satisfaction_level       0
last_evaluation           0
number_project            0
average_montly_hours      0
time_spend_company        0
Work_accident             0
left                      0
promotion_last_5years     0
Department                0
salary                    0
dtype: int64


 Basic Statistics:
        satisfaction_level  last_evaluation  number_project  \
 count        14999.000000     14999.000000    14999.000000
 mean             0.612834         0.716102        3.803054
 std              0.248631         0.171169        1.232592
 min              0.090000         0.360000        2.000000
 25%              0.440000         0.560000        3.000000
 50%              0.640000         0.720000        4.000000
 75%              0.820000         0.870000        5.000000
 max              1.000000         1.000000        7.000000

        average_montly_hours  time_spend_company  Work_accident          left  \
 count          14999.000000        14999.000000   14999.000000  14999.000000
 mean             201.050337            3.498233       0.144610      0.238083
 std               49.943099            1.460136       0.351719      0.425924
 min               96.000000            2.000000       0.000000      0.000000
 25%              156.000000            3.000000       0.000000      0.000000
 50%              200.000000            3.000000       0.000000      0.000000
 75%              245.000000            4.000000       0.000000      0.000000
 max              310.000000           10.000000       1.000000      1.000000

        promotion_last_5years
 count           14999.000000
 mean                0.021268
 std                 0.144281
 min                 0.000000
 25%                 0.000000
 50%                 0.000000
 75%                 0.000000
 max                 1.000000
```

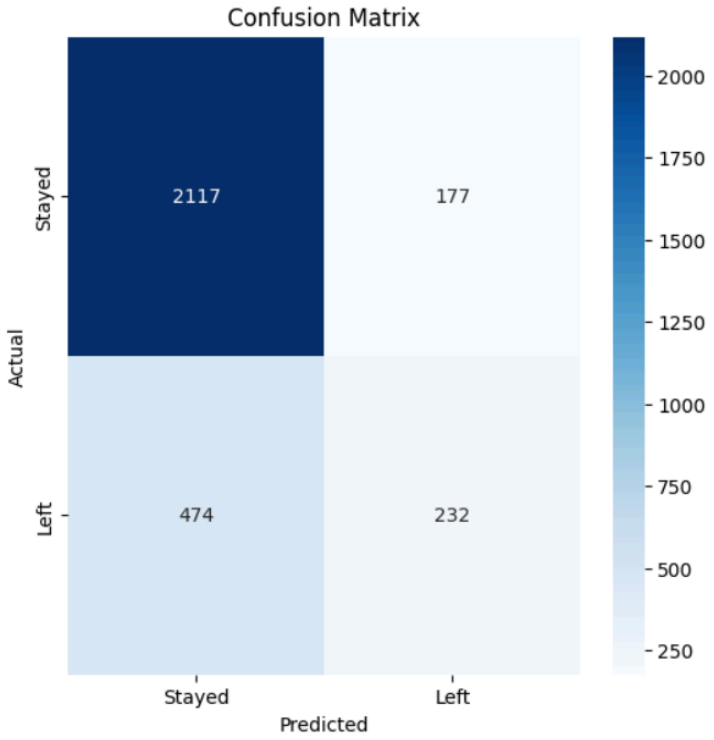

Employee Retention (Left vs Stayed)
10000
8000
6000
4000
2000
0
count
0
1
left


Impact of Salary on Employee Retention
left
0
1
5000
4000
3000
2000
1000
0
Count
low
medium
high
Salary


Impact of Department on Employee Retention
left
0
1
Count
3000
2500
2000
1500
1000
500
0
sales
accounting
hr
technical
support
management
IT
product_mng
marketing
RandD
Department


Correlation Matrix
satisfaction_level 1 0.11 -0.14 -0.02 -0.1 0.059 -0.39 0.026
last_evaluation 0.11 1 0.35 0.34 0.13 -0.0071 0.0066 -0.0087
number_project -0.14 0.35 1 0.42 0.2 -0.0047 0.024 -0.0061
average_montly_hours -0.02 0.34 0.42 1 0.13 -0.01 0.071 -0.0035
time_spend_company -0.1 0.13 0.2 0.13 1 0.0021 0.14 0.067
Work_accident 0.059 -0.0071 -0.0047 -0.01 0.0021 1 -0.15 0.039
left -0.39 0.0066 0.024 0.071 0.14 -0.15 1 -0.062
promotion_last_5years 0.026 -0.0087 -0.0061 -0.0035 0.067 0.039 -0.062 1
satisfaction_level
last_evaluation
number_project
verage_montly_hours
ime_spend_company
Work_accident
left
romotion_last_5years
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

```
Accuracy of the Logistic Regression Model: 0.78
```



In [11]:

```python
#zoo_dataset
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, confusion_matrix

from google.colab import files

# Step 1: Upload both datasets
print("Please upload zoo-data.csv")
uploaded = files.upload()
zoo_df = pd.read_csv(next(iter(uploaded)))

print("Please upload zoo-class_type.csv")
uploaded = files.upload()
class_info_df = pd.read_csv(next(iter(uploaded)))

# Step 2: Data inspection
print("\nZoo Data Sample:")
print(zoo_df.head())

print("\nClass Type Data Sample:")
print(class_info_df.head())

# Step 3: Drop 'animal_name' as it's a label, not a feature
zoo_df = zoo_df.drop(columns=['animal_name'])

# Step 4: Features and target
X = zoo_df.drop(columns=['class_type'])
y = zoo_df['class_type']
```

```
# Step 3: Drop 'animal_name' as it's a label, not a feature
zoo_df = zoo_df.drop(columns=['animal_name'])

# Step 4: Features and target
X = zoo_df.drop(columns=['class_type'])
y = zoo_df['class_type']

# Step 5: Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 6: Build and train Logistic Regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Step 7: Predictions and accuracy
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"\nModel Accuracy: {accuracy:.2f}")

# Step 8: Confusion Matrix
cm = confusion_matrix(y_test, y_pred)

# Step 8: Confusion Matrix (Fixed version)
plt.figure(figsize=(8, 6))
ax = sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
ax.set_title('Confusion Matrix')
ax.set_xlabel('Predicted Class')
ax.set_ylabel('Actual Class')
plt.show()
```

```
Please upload zoo-data.csv
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

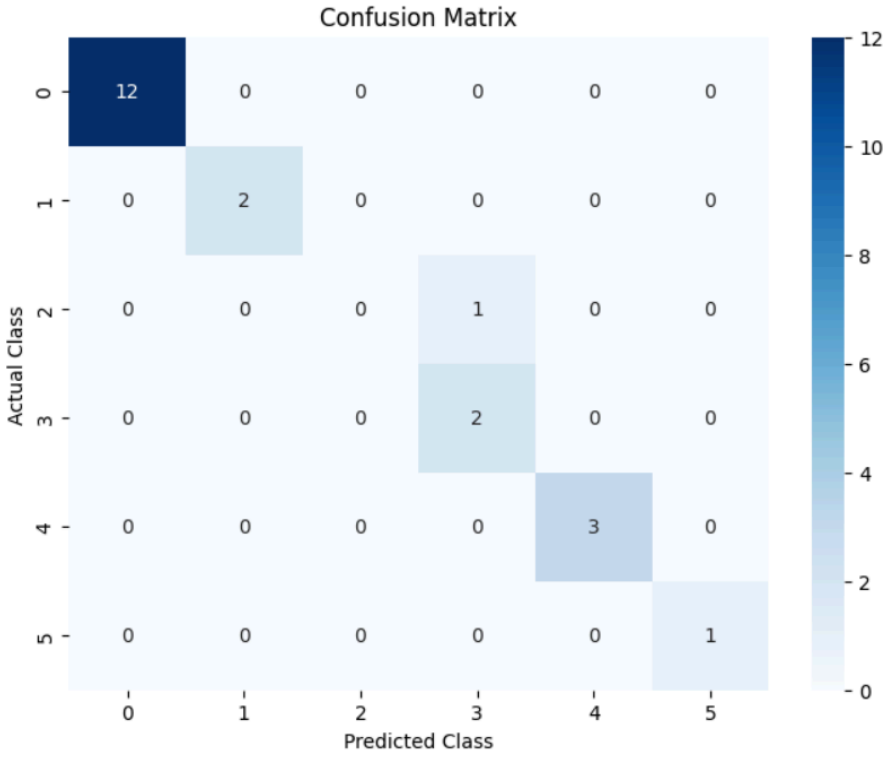
```
Saving zoo-data.csv to zoo-data (2).csv
Please upload zoo-class_type.csv
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving zoo-class-type.csv to zoo-class-type (2).csv

Zoo Data Sample:
  animal_name  hair  feathers  eggs  milk  airborne  aquatic  predator  \
0    aardvark     1         0     0     1         0        0         1
1    antelope     1         0     0     1         0        0         0
2        bass     0         0     1     0         0        1         1
3        bear     1         0     0     1         0        0         1
4        boar     1         0     0     1         0        0         1

   toothed  backbone  breathes  venomous  fins  legs  tail  domestic  catsize  \
0        1         1         1         0     0     4     0         0        1
1        1         1         1         0     0     4     1         0        1
2        1         1         0         0     1     0     1         0        0
3        1         1         1         0     0     4     0         0        1
4        1         1         1         0     0     4     1         0        1

   class_type
0           1
1           1
2           4
3           1
4           1
```

```
Class Type Data Sample:
   Class_Number  Number_Of_Animal_Species_In_Class Class_Type  \
0             1                                 41     Mammal
1             2                                 20       Bird
2             3                                  5    Reptile
3             4                                 13       Fish
4             5                                  4  Amphibian

                                     Animal_Names
0  aardvark, antelope, bear, boar, buffalo, calf,...
1  chicken, crow, dove, duck, flamingo, gull, haw...
2    pitviper, seasnake, slowworm, tortoise, tuatara
3  bass, carp, catfish, chub, dogfish, haddock, h...
4                            frog, frog, newt, toad

Model Accuracy: 0.95
```

Model Accuracy: 0.95



Monday

Date 17/3/25
Page

Lab - 3

Find linear Regression of the data of week and product sales.

x: (week)	y: (sales in thousands)
1	2
2	4
3	5
4	9

$$\beta = ((X^T X)^{-1} X^T) y$$

$$X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{bmatrix} \qquad X^T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{bmatrix}$$

$$X^T X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 4 & 10 \\ 10 & 30 \end{bmatrix}$$

$$(X^T X)^{-1} = \frac{1}{20} \begin{bmatrix} 30 & -10 \\ -10 & 4 \end{bmatrix} = \begin{bmatrix} 3/2 & -1/2 \\ -1/2 & 1/5 \end{bmatrix}$$

$$(X^T X)^{-1} X^T = \begin{bmatrix} 3/2 & -1/2 \\ -1/2 & 1/5 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1/2 & 0 & -1/2 \\ -3/10 & -1/10 & 1/10 & 3/10 \end{bmatrix}$$

$$((X^T X)^{-1} X^T) \cdot y = \begin{bmatrix} 1 & 1/2 & 0 & -1/2 \\ -3/10 & -1/10 & 1/10 & 3/10 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \\ 5 \\ 9 \end{bmatrix}$$

$$\beta = \begin{bmatrix} -1/2 \\ 11/5 \end{bmatrix}$$

$$\beta_0 = -1/2 \quad , \quad \beta_1 = 11/5$$

$$y = -1/2 + 11/5\, x$$

1) Considering the new data.csv file (canada_per-capital_income.csv, Salary.csv, hiring.csv), Did you perform any data preprocessing steps ? if yes, why ?

→ Yes,
- Missing values with the mean of respective columns to maintain data consistency
- Feature Encoding: convert categorical variables into numerical format,
- Data Splitting: train-test split with an 80/20 ratio for training & testing set
- Visualization: Scatter plots were used to identify relationships.
- ~~Feature Scaling~~ ;

2) For canada_per_capita_income.csv, did you visualize the regression line along with the data points ? what does the plot tell you about the relationship between year and per capita income ?

→ Yes,
- The Scatter plot displays individual data points representing the year vs. per capita income.
- The regression line represents the linear relationship b/w the year and per capita income.

3) For hiring.csv, what is the predicted salary for a candidate with 12 years of experience, 10 test score, and 10 interview score ?

→ $ 70,952.38

Q For 1000 companies did you use encoding? Did you scale the feature?

→ Yes, one hot encoding was used to encode state column. Scaling wasn't required as the column were all within a similar range.

Q Predict price of 20 inch pizza

Diameter	Price
8	10
10	13
12	16

17/3

→ $X = \begin{bmatrix} 1 & 8 \\ 1 & 10 \\ 1 & 12 \end{bmatrix}$, $Y = \begin{bmatrix} 10 \\ 13 \\ 16 \end{bmatrix}$, $B = \begin{bmatrix} B_0 \\ B_1 \end{bmatrix}$

$$XX^T = \begin{bmatrix} 1 & 1 & 1 \\ 8 & 10 & 12 \end{bmatrix} \begin{bmatrix} 1 & 8 \\ 1 & 10 \\ 1 & 12 \end{bmatrix} = \begin{bmatrix} 3 & 30 \\ 30 & 308 \end{bmatrix}$$

$$(X^T X)^{-1} = \frac{1}{24} \begin{bmatrix} 308 & -30 \\ -30 & 3 \end{bmatrix} = \begin{bmatrix} 77/6 & -5/4 \\ -5/4 & 1/8 \end{bmatrix}$$

$$(X^T X)^{-1} X^T = \begin{bmatrix} 77/6 & -5/4 \\ -5/4 & 1/8 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 8 & 10 & 12 \end{bmatrix} = \begin{bmatrix} 17/16 & 1/3 & -13/16 \\ -1/4 & 0 & 1/4 \end{bmatrix}$$

$$((X^T X)^{-1} X^T) Y = \begin{bmatrix} 17/6 & 1/3 & -13/16 \\ -1/4 & 0 & 1/4 \end{bmatrix} \begin{bmatrix} 10 \\ 13 \\ 16 \end{bmatrix} = \begin{bmatrix} -2 \\ 3/2 \end{bmatrix}$$

$B_0 = -2$, $B_1 = 3/2$

$y = -2 + 3/2\, x$

for 20 inch,

$$y = -2 + \frac{3}{2}(20) = ₹28$$