

Ex.No: 5.a

**DIABETES CLASSIFICATION**

Date: 24-Jan-2025

**Aim:-**

To train a logistic regression model to accurately predict diabetes based on health metrics.

**Program Code:-**

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Load the dataset
data = pd.read_csv('Diabetes.csv')

# Preview the dataset
print("Preview the data")
print(data.head())

# Select features and target variable
X = data.drop('Outcome', axis=1)
y = data['Outcome']

# 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)

# Initialize and train the logistic regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

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

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

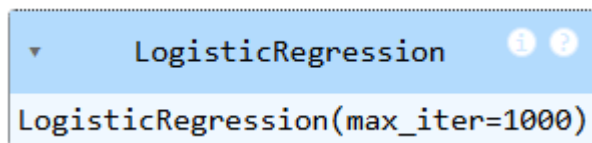
print(f'Accuracy: {accuracy:.2f}')
print('Confusion Matrix:')
print(conf_matrix)
print('Classification Report:')
print(class_report)
```

**Output:-**

Preview the data

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI \
0	6	148	72	35	0	33.6
1	1	85	66	29	0	26.6
2	8	183	64	0	0	23.3
3	1	89	66	23	94	28.1
4	0	137	40	35	168	43.1

	DiabetesPedigreeFunction	Age	Outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1



```
LogisticRegression(max_iter=1000)
```

Accuracy: 1.00

Confusion Matrix:

[[1]]

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
accuracy			1.00	1
macro avg	1.00	1.00	1.00	1
weighted avg	1.00	1.00	1.00	1

**Result:-**

Thus, the program was successfully executed.

Ex.No: 5.b

**CREDIT CARD DEFAULT PREDICTIONS**

Date: 24-Jan-2025

**Aim:-**

To train a logistic regression model to accurately predict credit card default using customer data.

**Program Code:-**

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

#Load the data
data=pd.read_csv('Creditcard.csv')

#Preview the data
print("Preview the dataset")
print(data.head())

# Select features and target variable
X = data.drop('Default', axis=1)
y = data['Default']

# 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)

# Initialize and train the logistic regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

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

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

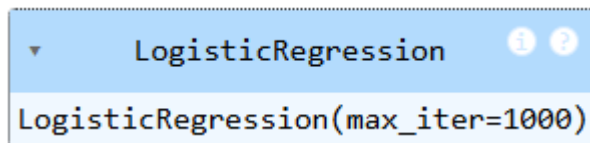
# Print results
print(f'Accuracy: {accuracy:.2f}')
print('Confusion Matrix:')
print(conf_matrix)

# Print predictions
predictions = pd.DataFrame({'CreditScore': X_test['CreditScore'], 'Actual': y_test, 'Predicted': y_pred})
print(predictions)
```

**Output:-**

Preview the dataset

	CreditScore	Age	Income	LoanAmount	Default
0	700	34	50000	20000	0
1	600	45	45000	15000	1
2	650	29	30000	12000	0
3	720	41	60000	25000	0
4	580	36	32000	10000	1



```
LogisticRegression(max_iter=1000)
```

Accuracy: 1.00

Confusion Matrix:

[[1]]

	CreditScore	Actual	Predicted
1	600	1	1

**Result:-**

Thus, the program was successfully executed.

Ex.No: 5.c

**HEART DISEASE CLASSIFICATION**

Date: 24-Jan-2025

**Aim:-**

To train a logistic regression model to accurately classify heart disease based on various health indicators.

**Program Code:-**

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

# Load the dataset
data = pd.read_csv('Heartdisease.csv')

#Preview the dataset
print(f"Preview the dataset")
print(data.head())

# Select features and target variable
X = data.drop('HeartDisease', axis=1)
y = data['HeartDisease']

# 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)

# Initialize and train the logistic regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

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

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

# Print results
print(f'Accuracy: {accuracy:.2f}')
print('Confusion Matrix:')
print(conf_matrix)

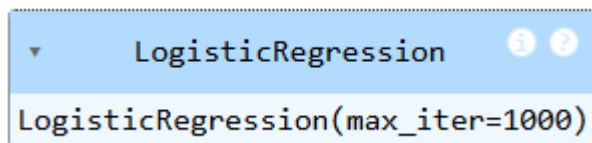
# Print predictions
predictions = pd.DataFrame({'Age': X_test['Age'], 'Actual': y_test, 'Predicted': y_pred})
print(predictions)
```

**Output:-**

Preview the dataset

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG \
0	63	1	3	145	233	1	0
1	37	1	2	130	250	0	1
2	41	0	1	130	204	0	0
3	56	1	1	120	236	0	1
4	57	0	0	120	354	0	1

	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	150	0	2.3	0	1
1	187	0	3.5	1	1
2	172	0	1.4	2	1
3	178	0	0.8	2	1
4	163	1	0.6	2	0



```
LogisticRegression(max_iter=1000)
```

Accuracy: 1.00

Confusion Matrix:

[[1]]

	Age	Actual	Predicted
1	37	1	1

**Result:-**

Thus, the program was executed successfully.