DATE : / /

## Linear Regression :-

```
import numpy as np
import matplotlib.pyplot as plt
x = np.array([1, 2, 3, 4, 5])
y = np.array([2, 4, 5, 4, 5])

x_mean = np.mean(x)
y_mean = np.mean(y)

numerator = np.sum((x - x_mean) * (y - y_mean))
denominator = np.sum((x - x_mean) ** 2)
b1 = numerator/denominator
b0 = y_mean - (b1 * x_mean)
print(f" Linear Regression Eq-n : y = {b0:.2f} + {b1:.
y_pred = b0 + b1 * x
plt.scatter(x, y, color='blue', label = 'Data Points')
plt.plot(x, y_pred, color='red', label = 'Regression line')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Linear Regression')
plt.legend()
plt.show()
```

PAGE :
DATE : / /

## Logistic regression

```
import numpy as np
class LogisticRegression:
    def init_(self, learning_rate=0.01, num_iterations=1000):
        self.learning_rate = learning_rate
        self.num_iterations = num_iterations
        self.weights = None
        self.bias = None
    def sigmoid(self, z):
        return 1/(1 + np.exp(-z))
    def fit(self, X, Y):
        n_samples, n_features = X.shape
        self.weights = np.zeros(n_features)
        self.bias = 0
        for _ in range(self.num_iterations):
            linear_model = np.dot(x, self.weights) + self.bias
            y_predicted = self.sigmoid(linear_model) ~~dw~~=
            dw =                                    (1/(n_samples) *
                np.dot (x.T, (ypredicted - y))
                db = ~~db~~= (1/n_samples )* np.sum(
                    y_predicted-y)~~(p~~
            self.weights -= self.learning_rate * dw
            self.bias -= self.learning_rate * db
    def predict (self, x):
        linear_model = np.dot(x, self.weights) + self.bias
        y_predicted = self.sigmoid(linear_model)
        y_predicted_cls = [1 if i > 0.5 else 0 for i in
                              y_predicted]
        return np.array(y_predicted_cls)
if __name__ == "__main__":
```

```
X= np.array([[1,2], [2,3], [3,4], (4,5], [5,6]])
y = np.array([0,0,1,1,1])

model = LogisticRegression()
model.fit(X,Y)
y_pred = model.predict(x)
print("Predictions:", y_pred).
```