Md Toufique Hasan

Student Number: 151129267

Exercise date: 23.03.2023

Machine Learning Algorithms: exercise 2

Answer: 1

```
% Load data from file
data = load(['D:\TUNI\Courses\Period-4\DATA.ML.210 [Machine Learning Algorithms]' ...
    '\Exercises 1\Data.txt']);
data
```

```
data = 212×2
    0.8028   -1.6510
    0.9140    0.9831
    1.5186    0.9786
    1.8732   -0.1436
    1.4445    0.8740
   -0.7420    0.0806
    1.7561   -1.0257
    1.3040    0.1644
   -2.5543    0.5886
   -1.1737   -0.4639
      ⋮
```

```
size(data)
```

```
ans = 1×2
   212     2
```

```
X = data(:, 1:2);
X
```

```
X = 212×2
    0.8028   -1.6510
    0.9140    0.9831
    1.5186    0.9786
    1.8732   -0.1436
    1.4445    0.8740
   -0.7420    0.0806
    1.7561   -1.0257
    1.3040    0.1644
   -2.5543    0.5886
   -1.1737   -0.4639
      ⋮
```

```
y = data(:, 2);
y
```

```
y = 212×1
   -1.6510
```

```
    0.9831
    0.9786
   -0.1436
    0.8740
    0.0806
   -1.0257
    0.1644
    0.5886
   -0.4639
      ⋮
```

```
% Define the classifier parameters
w = 1/sqrt(2) * [1; -1];
p = [2; 2];

% Classify the points using the classifier
n = size(X, 1);
X_diff = X - repmat(p', n, 1);
y_pred = sign(X_diff * w);

% Evaluate the performance of the classifier
accuracy = sum(y_pred == y) / length(y);

TP = sum(y_pred == 1 & y == 1);
FN = sum(y_pred == -1 & y == 1);
if TP + FN == 0
    sensitivity = 0;
else
    sensitivity = TP / (TP + FN);
end

TN = sum(y_pred == -1 & y == -1);
FP = sum(y_pred == 1 & y == -1);
if TN + FP == 0
    specificity = 0;
else
    specificity = TN / (TN + FP);
end

if TP + TN == 0
    accuracy = 0;
else
    accuracy = (TP + TN) / (TP + TN + FP + FN);
end

% Display the results
fprintf('Accuracy: %.2f\n', accuracy);
```

```
Accuracy: 0.00
```

```
fprintf('Sensitivity: %.2f\n', sensitivity);
```

```
Sensitivity: 0.00
```

```
fprintf('Specificity: %.2f\n', specificity);
```

```
Specificity: 0.00
```

Answer: 2

```
if TP + TN + FP + FN == 0
    P_error = 0;
else
    P_error = (FP + FN) / (TP + TN + FP + FN);
end

% Display the results
fprintf('Accuracy: %.2f\n', accuracy);
```

```
Accuracy: 0.00
```

```
fprintf('Sensitivity: %.2f\n', sensitivity);
```

```
Sensitivity: 0.00
```

```
fprintf('Specificity: %.2f\n', specificity);
```

```
Specificity: 0.00
```

```
fprintf('Probability of error: %.2f\n', P_error);
```

```
Probability of error: 0.00
```

Answer: 3

```
% Define the class-conditional probabilities
p_x_given_w1 = @(x) double(0 <= x & x <= 1);
p_x_given_w2 = @(x) double(1/2 <= x & x <= 5/2) * 1/2;

% Define the prior probabilities
p_w1 = 1/2;
p_w2 = 1/2;

% Define the range of x values to consider
x_min = 0;
x_max = 3;
dx = 0.01;
x_range = x_min:dx:x_max;

% Compute the conditional probabilities
p_w2_given_w1 = integral(@(x) p_x_given_w2(x) * p_w1 ./ max(p_x_given_w1(x) ...
    * p_w1 + p_x_given_w2(x) * p_w2, eps), x_min, x_max);
p_w1_given_w2 = 0;
```

```
% Compute the probability of error
p_error = p_w1 * p_w2_given_w1 + p_w2 * p_w1_given_w2;

% Display the result
fprintf('Probability of error: %.2f\n', p_error);
```

```
Probability of error: 0.83
```

Answer: 5

```
% Define the class-conditional densities
p_x_given_w1 = @(x) 1 / sqrt(2 * pi) * exp(-x.^2 / 2);
p_x_given_w2 = @(x) 1 / sqrt(4 * pi) * exp(-(x - 3).^2 / 8);

% Define the prior probabilities
p_w1 = 0.3;
p_w2 = 0.7;

% Define the range of x values to consider
x_min = -1;
x_max = 6;
dx = 0.01;
x_range = x_min:dx:x_max;

% Compute the probabilities of each class
p_w1_given_x = p_x_given_w1(x_range) * p_w1 ./ (p_x_given_w1(x_range) ...
    * p_w1 + p_x_given_w2(x_range) * p_w2);
p_w2_given_x = 1 - p_w1_given_x;

% Estimate the probability of error
p_error = sum(min(p_w1_given_x, p_w2_given_x)) * dx;

% Display the result
fprintf('Probability of error: %.4f\n', p_error);
```

```
Probability of error: 0.9722
```

Answer: 6

```
% Read in the data
data2 = load(['D:\TUNI\Courses\Period-4\DATA.ML.210 [Machine Learning Algorithms]' ...
    '\Exercises 2\data2.txt']);

% Extract the x and y data
x = data2(:,1);
y = data2(:,2);

% Fit polynomial of order 1
fit1 = fitlm(x, y);
fit1
```

```
fit1 =
Linear regression model:
    y ~ 1 + x1

Estimated Coefficients:
                   Estimate       SE        tStat       pValue
                   ________    _________   _______    __________

    (Intercept)     0.95134     0.035392     26.88    1.8495e-106
    x1             -0.30298    0.0097573   -31.051    6.5516e-129


Number of observations: 629, Error degrees of freedom: 627
Root Mean Squared Error: 0.444
R-squared: 0.606,  Adjusted R-Squared: 0.605
F-statistic vs. constant model: 964, p-value = 6.55e-129
```

```
% Fit polynomial of order 2
fit2 = fitlm(x, y, 'poly2');
fit2
```

```
fit2 =
Linear regression model:
    y ~ 1 + x1 + x1^2

Estimated Coefficients:
                    Estimate        SE         tStat       pValue
                   __________    _________    ________    __________

    (Intercept)       0.94893     0.053025      17.896    3.7356e-58
    x1               -0.30067     0.039002      -7.709    4.9995e-14
    x1^2           -0.0003672    0.0060128    -0.06107       0.95132


Number of observations: 629, Error degrees of freedom: 626
Root Mean Squared Error: 0.445
R-squared: 0.606,  Adjusted R-Squared: 0.605
F-statistic vs. constant model: 481, p-value = 2.55e-127
```

```
% Fit polynomial of order 3
fit3 = fitlm(x, y, 'poly3');
fit3
```

```
fit3 =
Linear regression model:
    y ~ 1 + x1 + x1^2 + x1^3

Estimated Coefficients:
                   Estimate       SE        tStat      pValue
                   ________    _________    ______    _________

    (Intercept)     -0.2005     0.010586    -18.94    1.443e-63
    x1               1.9045      0.01461    130.35            0
    x1^2           -0.87891    0.0054085    -162.5            0
    x1^3           0.093264    0.0005661    164.75            0


Number of observations: 629, Error degrees of freedom: 625
Root Mean Squared Error: 0.0668
R-squared: 0.991,  Adjusted R-Squared: 0.991
F-statistic vs. constant model: 2.33e+04, p-value = 0
```

```
% Fit polynomial of order 4
fit4 = fitlm(x, y, 'poly4');
fit4
```

```
fit4 =
Linear regression model:
    y ~ 1 + x1 + x1^2 + x1^3 + x1^4

Estimated Coefficients:
                   Estimate        SE         tStat       pValue
                   _________    __________    _______    __________

    (Intercept)     -0.19959      0.013196    -15.125     2.9319e-44
    x1                1.9015      0.029165     65.199    9.6805e-281
    x1^2            -0.87681      0.018908    -46.373    2.4247e-204
    x1^3            0.092741     0.0045254     20.494     9.2351e-72
    x1^4           4.162e-05    0.00035747    0.11643        0.90735


Number of observations: 629, Error degrees of freedom: 624
Root Mean Squared Error: 0.0668
R-squared: 0.991,  Adjusted R-Squared: 0.991
F-statistic vs. constant model: 1.74e+04, p-value = 0
```

```
% Fit polynomial of order 5
fit5 = fitlm(x, y, 'poly5');
fit5
```

```
fit5 =
Linear regression model:
    y ~ 1 + x1 + x1^2 + x1^3 + x1^4 + x1^5

Estimated Coefficients:
                     Estimate         SE         tStat       pValue
                   __________    __________    _______    __________

    (Intercept)      0.015498     0.0010264       15.1    3.9625e-44
    x1                 0.8625     0.0033118     260.44             0
    x1^2              0.28506     0.0032744     87.058             0
    x1^3             -0.40121     0.0013232    -303.22             0
    x1^4             0.088563    0.00023234     381.17             0
    x1^5           -0.0056383    1.4724e-05    -382.92             0


Number of observations: 629, Error degrees of freedom: 623
Root Mean Squared Error: 0.00435
R-squared: 1,  Adjusted R-Squared: 1
F-statistic vs. constant model: 3.32e+06, p-value = 0
```

Answer: 4

```
function prob = normprob(mu, sigma2, a, b)
% Calculate the probability for an event that falls on the interval [a,b]
% in a one-dimensional normal distribution with mean mu and variance sigma2

% Define the standard deviation
sigma = sqrt(sigma2);

% Define the limits of integration
```

```
x = linspace(a, b, 1000);

% Calculate the probability density function
pdf = 1/(sigma*sqrt(2*pi)) * exp(-(x-mu).^2/(2*sigma^2));

% Approximate the integral using cumulative sum of the pdf
prob = sum(pdf)*(b-a)/length(x);
end
```