COMP7703 – Homework Task 3
Joel Thomas 44793203

1.

The k-nearest means classifier works by assigning an input to the nearest cluster mean (and hence class) after convergence during training has been achieved.

Since each input $x^t \in \mathbb{R}$, for a single point, we need to calculate its distances to each of the K sample means → in total $= K \times N = KN$ distance values.

2.

Given a sample covariance matrix C, we can calculate individual correlations using $\rho_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$ $\forall\, i = 1,2,3$ where σ_{ij} refers to the covariance between the ith and jth elements and σ_i refers to the standard deviation of the ith element. Using the $corrcov()$ function in MATLAB to transform a covariance matrix into a correlation matrix, we have:

$$\rho = \begin{bmatrix} 1 & -0.1018 & 0.0192 \\ -0.1018 & 1 & -0.7559 \\ 0.0192 & -0.7559 & 1 \end{bmatrix}$$

3.

Applying quadratic discriminant analysis on the provided training dataset and using a full but equal covariance matrix for each class, we obtain the training error to be approximately 0.0317 or 3.17% and the testing/validation error to be approximately 0.0685 or 6.85%. The code $Q3.m$ used in finding these results is provided below:

```
clear all;
train_data = readtable("BreastCancerTrain.csv");
test_data = readtable("BreastCancerValidation.csv");

% Creating train x and train y
train_x = train_data{:, 1:9};
train_y = train_data{:, 10};

% Creating test x and test y
test_x = test_data{:, 1:9};
test_y = test_data{:, 10};
```

```
% Create a quadratic classifier
    % 'quadratic' = covariance matrices can vary among classes
    % 'diagquadratic' = covariance matrices are diagonal and can vary among
        % classes
model = fitcdiscr(train_x, train_y, 'DiscrimType', 'quadratic');
% model = fitcdiscr(train_x, train_y, 'DiscrimType', 'diagquadratic');

% Verify targets
model.ClassNames

% Report training classification error
train_pred = model.predict(train_x);
train_acc = 0;

for i = 1:length(train_pred)
    if train_pred(i) == train_y(i)
        train_acc = train_acc + 1;
    end
end

train_acc = train_acc/length(train_pred);
train_err = 1 - train_acc

% Report testing classification error
test_pred = model.predict(test_x);
test_acc = 0;

for i = 1:length(test_pred)
    if test_pred(i) == test_y(i)
        test_acc = test_acc + 1;
    end
end

test_acc = test_acc/length(test_pred);
test_err = 1 - test_acc
```

4.

Using the approach outlined in the textbook, we can fit a 2-D quadratic regression model:

$$f(\boldsymbol{x1}, \boldsymbol{x2}) = w_0 + w_1\boldsymbol{x_1} + w_2\boldsymbol{x_2} + w_3\boldsymbol{x_1}\boldsymbol{x_2} + w_4\boldsymbol{x_1^2} + w_5\boldsymbol{x_2^2}$$

by instead fitting a 5-D standard polynomial regression model:

$$f(\boldsymbol{z_1}, \dots, \boldsymbol{z_5}) = w_0 + w_1\boldsymbol{z_1} + w_2\boldsymbol{z_2} + w_3\boldsymbol{z_3} + w_4\boldsymbol{z_4} + w_5\boldsymbol{z_5}$$

$$\boldsymbol{z_1} = \boldsymbol{x_1}, \quad \boldsymbol{z_2} = \boldsymbol{x_2}, \quad \boldsymbol{z_3} = \boldsymbol{x_1}\boldsymbol{x_2}, \quad \boldsymbol{z_4} = \boldsymbol{x_1^2}, \quad \boldsymbol{z_5} = \boldsymbol{x_2^2}$$

The linear fit in the five-dimensional space $(\boldsymbol{z_1}, \boldsymbol{z_2}, \boldsymbol{z_3}, \boldsymbol{z_4}, \boldsymbol{z_5})$ corresponds to a quadratic fit in the two-dimensional space $(\boldsymbol{x_1}, \boldsymbol{x_2})$. We can obtain the vector of weights $\boldsymbol{w}$ by solving $(Z^TZ)^{-1}Z^T\boldsymbol{r}$ where $Z = [\boldsymbol{1}, \boldsymbol{z_1}, \boldsymbol{z_2}, \boldsymbol{z_3}, \boldsymbol{z_4}, \boldsymbol{z_5}]$ is a matrix of column vectors and $\boldsymbol{r}$ is the target vector. Performing all of these steps in MATLAB, we have the fitted model:

$$\hat{f}(\boldsymbol{x1}, \boldsymbol{x2}) = 0.0604 - 0.2570\boldsymbol{x_1} + 0.0513\boldsymbol{x_2} + 0.1381\boldsymbol{x_1}\boldsymbol{x_2} + 1.1423\boldsymbol{x_1^2} + 0.8996\boldsymbol{x_2^2}$$

$$\widehat{w}_0 = 0.0604, \quad \widehat{w}_1 = -0.2570, \quad \widehat{w}_2 = 0.0513, \quad \widehat{w}_3 = 0.1381, \quad \widehat{w}_4 = 1.1423, \quad \widehat{w}_5 = 0.8996$$

The code $Q4.m$ used in finding these results is provided below:

```
clear all;
train_data = importdata("reg2d.csv");

train_x = train_data(:, 1:2);
train_y = train_data(:, 3);

% Linear fit in 5-D space (z1, z2, z3, z4, z5) corresponds to a quadratic
% fit in 2-D space (x1, x2)
z1 = train_x(:, 1);
z2 = train_x(:, 2);
z3 = z1.*z2;
z4 = z1.^2;
z5 = z2.^2;
Z = [ones(size(z1)), z1, z2, z3, z4, z5];

r = train_y;
```

```
% Vector of weights obtained from solving (Z^T * Z)^-1 * Z^T * r
w = (Z.' * Z) \ ((Z.')*r);

% Print SSE
train_error = sum((train_y - Z*w).^2)

% Function for plotting regression model
f = @(Z) Z*w;

% Finely spaced points to use for plotting regression model
z1 = linspace(min(z1), max(z1), 1000).';
z2 = linspace(min(z2), max(z2), 1000).';
z3 = z1.*z2;
z4 = z1.^2;
z5 = z2.^2;
Z = [ones(size(z1)), z1, z2, z3, z4, z5];

% Plot 3-D scatterplot of x1, x2 and r as well 3-D plot of regression
% model
hold on
view(3);
scatter3(train_x(:, 1), train_x(:, 2), train_y)
plot3(z1, z2, f(Z))
xlabel("x1")
ylabel("x2")
zlabel("f(x1, x2)")
title("2-D Quadratic Regression Model on reg2d.csv")
set(gca, 'FontSize', 15)
hold off
```

5.

$K = 7$ classes, $d = 9$ input features, $n = 3$ mixture models

Total number of mean parameters $= 9$ per class $\times$ 7 classes $\times$ 3 mixtures $= K \times d \times n = 189$

Number of parameters in a **shared/equal diagonal** covariance matrix $= d = 9$

since $s_{ii} \neq 0, s_{ij} = 0 \; \forall \; i \neq j, i, j = 1, \dots, 9$

∴ Total number of covariance parameters $= 9$ per mixture $\times$ 3 mixtures $= K \times n = 27$

∴ Total number of model parameters to obtain $\mathbb{P}(\boldsymbol{x}) = 189 + 27 = Kn(d + 1) = 216$ disregarding priors.