Danmarks Tekniske Universitet

Project 1 - 02526 Mathematical Modeling

23. February 2020

Authors:	***Study No:***
Mads Esben Hansen	s174434
Marcus Lenler Garsdal	s174440
Nicolaj Hans Nielsen	s184335

Contents

Report

1.1 Introduction

Salami play an inevitable role in food product sold and eaten in Denmark and its neighboring countries. Salami consists of cured meat and to this end it is important to ensure its quality. Companies who sell salami therefore put a lot of effort into ensuring consistent and good quality of their products. One key parameter in this process is determining the fat content of the salami. This report takes offset in this problem and tries to give efficient solutions to then. Furthermore, it will present and explain the methods used in these solutions as intuitively and concisely as possible.

1.2 Methodology

The materiel used in this report are multi-spectral images of pieces of salami after 01, 06, 13, 20, and 28 days of curing. Along with these images, an expert has annotated some parts of the salami from each to to be fat and meat respectively. Each image consists of 514*514 pixels with 19 values in each pixel which corresponds to the signal intensity for a given wavelength.

1.2.1 Simple model

The annotations denote some parts of the salami which are fat, and some parts which are meat. If we look at the histogram of the pixel values denoting these parts for e.g. band 3.

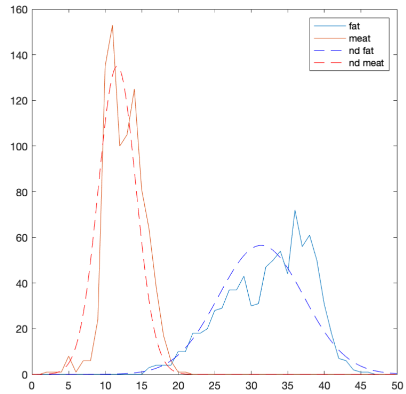

Figure 1.1: Histogram of fat and meat from day20 using band 3.

It is quite obvious that using this band it is possible to "guess" if a pixel is either fat or meat using only its pixel value. Using this idea we should be able to find some threshold value that works as a border between where we "guess" the pixel is fat and where it is meat. This threshold value, τ will then be defined as:

$$\tau = [Pdf_{fat}(x) = Pdf_{meat}(x)]_{x \in X} \quad \Leftrightarrow \quad Pdf_{fat}(\tau) = Pdf_{meat}(\tau)$$

In other words τ will be the intersection between the probability density function (PDF) of fat and meat respectively. To further simplify things, we will initially assume that the distributions of fat and meat follow the same shape, i.e. they have same variance but different means. This simplifies things even further since the threshold now can be found as the mean of the means.

$$\tau = ([\bar{fat}] + [\bar{meat}])/2$$

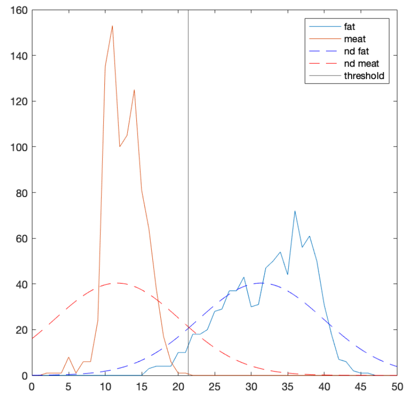

Figure 1.2: Visual representation of τ.

1.2.2 LDA

In the simple method described above, we only utilize one band and we of course want to use them all since no data should go to waste. This is in principle done the exact same way as describe in the simple method, however, we now increase the dimension of our PDF from 1 to 19 for fat and meat respectively. In other words, we are now using a multi-dimensional normal distribution. Such can be found as described in the exercise description (equation 18).

$$Pdf(x) = \frac{1}{\sqrt{2\pi}^n \sqrt{det(\hat{\Sigma})}} exp(-\tfrac{1}{2}(x - \hat{\mu})^T \hat{\Sigma}^{-1}(x - \hat{\mu}), \quad x \in X \in \mathbb{R}^n$$

Where,

$$\hat{\mu} = \tfrac{1}{m} \textstyle\sum_{i=1}^{m} x_i, \quad i = |X|, x_i \in X$$

$$\hat{\Sigma}(a, b) = \tfrac{1}{m-1} \textstyle\sum_{i=1}^{m} (x_{ai} - \hat{\mu_a})(x_{bi} - \hat{\mu_b}), \quad i = |X|, \ x_i \in X$$

For simplicity we again assume the variance of the fat and meat values are equal. This means that $\hat{\Sigma}$ will be the same for both fat and meat, thus we get:

$$\tau = [Pdf_{fat}(x) = Pdf_{meat}(x)]_{x \in X} \quad \Leftrightarrow$$

$$\tau = [(x - \hat{\mu}_{fat})^T \hat{\Sigma}^{-1}(x - \hat{\mu}_{fat}) = (x - \hat{\mu}_{meat})^T \hat{\Sigma}^{-1}(x - \hat{\mu}_{meat})]_{x \in X}$$

So we do not need to find the intersection between the PDFs per se, we only need to find the intersection between the exponents of **e** (actually even less, since the $-\frac{1}{2}$ does not matter either). Furthermore, if we know something about the data beforehand, we want to use as much of this information as possible. According to Bayes theorem cf the assignment material, we get:

$$g(x)_i = \frac{1}{\sqrt{2\pi}^n \sqrt{det(\hat{\Sigma})}} exp(-\tfrac{1}{2}(x - \hat{\mu})^T \hat{\Sigma}^{-1}(x - \hat{\mu}) \cdot p_i, \quad x \in X \in \mathbb{R}^n$$

Where p_i is the prior i.e. expected percentage. Similar to what we argued earlier, we still assume equal co-variance matrices and we can use:

$$g(x)_i = (x - \hat{\mu}_i)^T \hat{\Sigma}^{-1}(x - \hat{\mu}_i) + ln(p_i), \quad x \in X \in \mathbb{R}^n$$

as a classifier for fat and meat.

1.3 Results

1.3.1 Simple model

For our simple model the threshold values τ were calculated and shown in table 1.3.1.

Band	τ	Band	τ
1	23.7985	11	58.9076
2	17.0615	12	64.4189
3	25.0018	13	69.1780
4	27.4111	14	68.9655
5	29.7178	15	66.7830
6	30.9764	16	61.4899
7	29.9982	17	60.1673
8	32.0027	18	51.5878
9	49.8528	19	1.2867
10	55.1615		

Table 1.1: Threshold values τ calculated for all spectral bands for day 1.

The error rates ϵ_i used for assessing the accuracy of the model was calculated using equation 1.1. We calculate ϵ_i for $i \in [1 : 19]$ representing each spectral band. The relative error of pixels classified as meat f_m in the annotation images of meat A_m, and the relative error for pixels classified as fat f_f in annotation of fat A_f is averaged to determine the overall accuracy of the model.

$$\epsilon_i = \frac{1}{2} \cdot \frac{|f_m - A_m|}{A_m} + \frac{|f_f - A_f|}{A_f} \tag{1.1}$$

The best spectral band for the model is determined by the smallest error, and is determined to be band 14. Performing the characterization for all days using the optimal band trained on day 01 results in the distributions and relative errors shown in table 1.2.

-	Day 01	Day 06	Day 13	Day 20	Day 28
Fat Ratio	0.2427	0.2940	0.0919	0.1716	0.3087
Meat Ratio	0.7573	0.7060	0.9081	0.8284	0.6913
Relative Error	0.0051	0.1301	0.1532	0.2486	0.2040

Table 1.2: Distribution of meat & fat and relative error for single band model trained on day 01.

The result can also be represented graphically illustrating the categorized pixels for meat and fat for all other days as shown in picture 1.3.

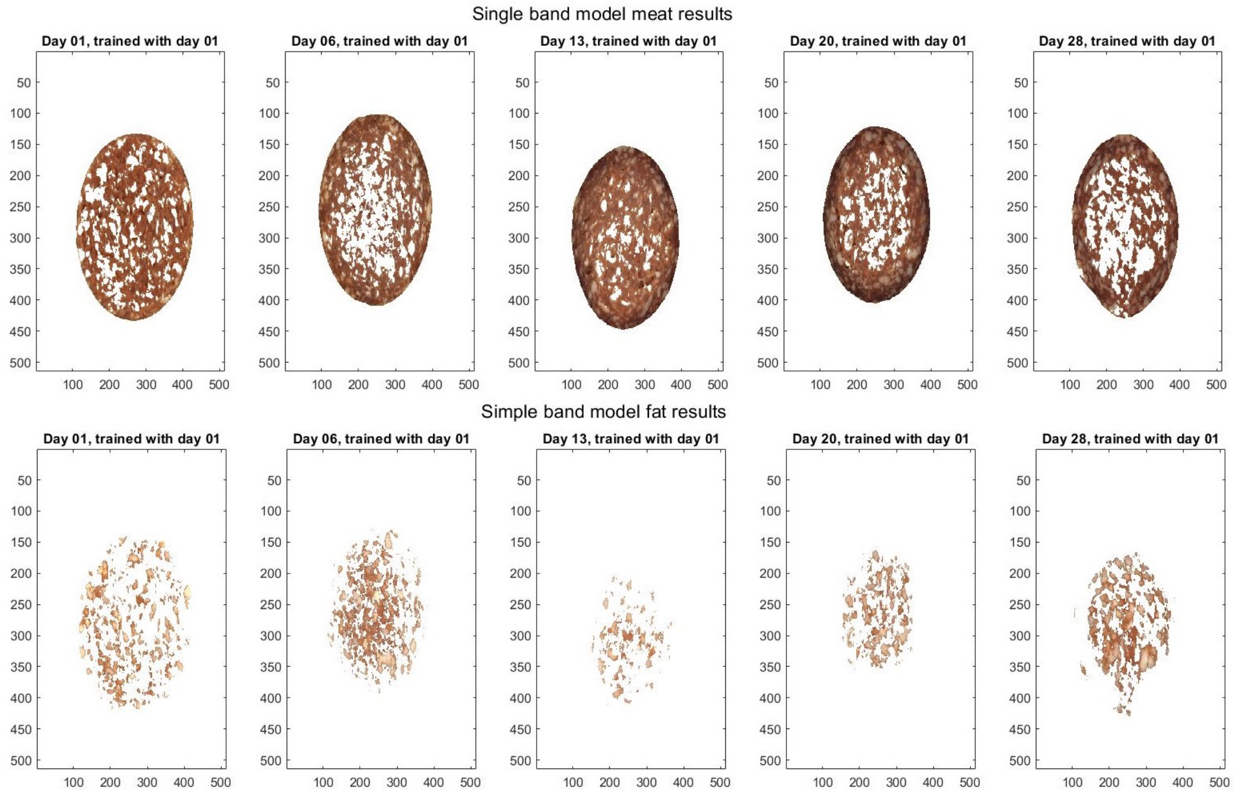

Figure 1.3: Characterization for single band model trained on day 01.

1.3.2 LDA

Before anything else, we did an evaluation of how well our model performed on data from the same day as the model was trained on. We used a set aside method where we put aside 200 pixels collected randomly from the the fat pixels and meat pixels respectively. We then generated a classifier function from the remaining data. We used this model to annotate both fat, A_{fat}, and meat, A_{meat}, for the 200 pixels we put aside and found an error rate for both of these. We then took the average between these error rates which we used as a single valued performance metric on our model.

$$ER_{day} = 1 - \frac{1}{2} \cdot \left(\frac{|A_{fat}|}{200} + \frac{|A_{meat}|}{200}\right)$$

This evaluation gave the following results:

DAY	ER
1	0.0075
6	0.9925
13	0.9975
20	0.9975
28	0.9826

Table 1.3: Error rates for the given days

Similar to the single band model the characterization for the model trained on day 01 can be illustrated as seen in fig 1.4.

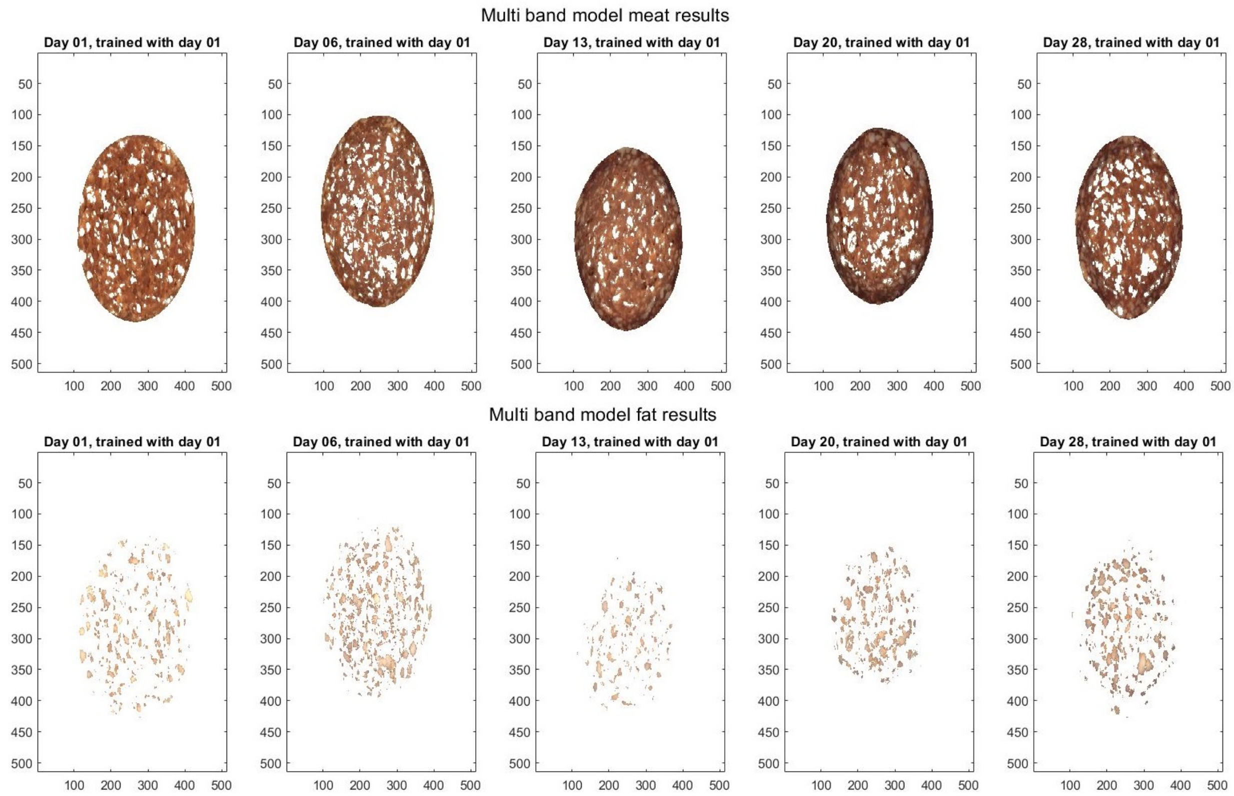

Figure 1.4: Characterization for multi band model trained on day 01.

The distributions of fat & meat and relative error can be seen in 1.3.2.

-	Day 01	Day 06	Day 13	Day 20	Day 28
Fat Ratio	0.0935	0.1298	0.0931	0.1097	0.1513
Meat Ratio	0.9065	0.8702	0.9069	0.8903	0.8487
Relative Error	0.0265	0.0022	0.0194	0.0436	0.0581

Table 1.4: Distribution of meat & fat and relative error for multi band model trained on day 01.

For the multi band model we also tested training the model on different days than day 01. The calculated relative errors for all combinations of training days and analyzed days are shown in figure 1.5. Note: The relative errors in the diagonal are from the set aside classification performed in the beginning of the section.

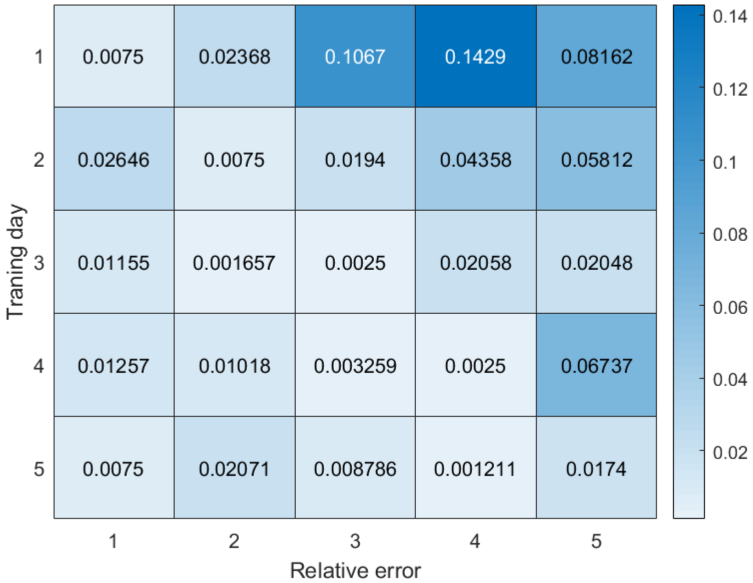

Figure 1.5: Heatmap with relative errors. Row represent training day, column is relative error on given day

Visually we can easily determinate that the relative errors are large when we train with day 1. The best results are obtained when we train on either day 13 or 28. If we sum the rows of day 13 and 28, we obtain 0.0568 and 0.0556 respectively. This suggests that day 28 is the best to train the model.
Finally we also took the priors into account, as discussed in the methodology section. Taking these into account and using the exact same methods as discussed above, we found the following error matrix:

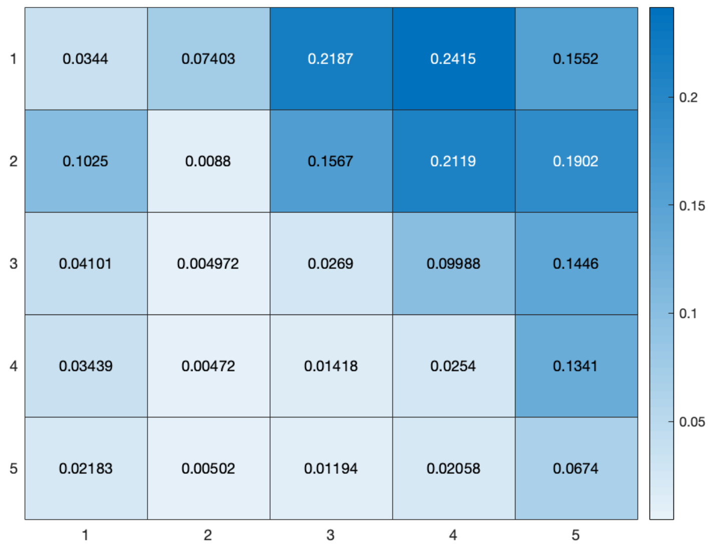

Figure 1.6: Heatmap also using priors. Row represent training day, column is relative error on given day

1.4 Discussion

When analyzing the images using our models we have made the assumption that we can distinguish between single pixels in our given image. Due to the high resolution of our data, 264.196 samples for a full image and 70.000 average samples for our subject, this has been determined to be a valid assumption.
In 1.5 we first took out the data in the diagonal because it doesn't seem fair to compare relative errors on data we also used to train the model. Later we made the set aside classification which is a better measurement to compare with and therefore we fitted into the diagonal. We acknowledge that the error from the set aside classification will deviate slightly each time because the training data is taken out randomly. However, it's still a better measurement.
Whether we include the set aside results or not, day 28 is still the best day to train on judge on the sum of relative errors in annotated areas.
We can now compare the simple threshold model and the LDA model. We use day 1 as training day to compare hence the first row in figure 1.5 and the the relative errors in table 1.2 with the sum 0.3624 and 0.7410 respectively. This suggest that the LDA is overall the better. The discrepancies in the relative errors are greatest on last days and this is also reflected into the images in figure 1.4 and figure 1.3. It's very evident if we look at the fat images on the last days where the simple model obviously classifies much meat as fat. The LDA model also performs worse the last days but not to the same extend.
On the images we also see another trend on the last days. It's as though the models performs worse on the periphery of the salami and if we look at the pictures, the salami is bend and darker in these areas. To optimize the model we should perhaps exclude this part or inspect it separately.
We are informed that 30% of the salami is fat. From this information, we where able to tweak our model using priors as described in the assignment material and in the methodology section of this report. Our result were a bit surprising at first, because it seemed the model performed worse after taking this into account. However, there are two parts to this story. Firstly, whenever a model gets more complex, there is always some risk of over-complicating or over-fitting your data; secondly, we tested the performance of our model on annotated data where the probability is more like 50%/50% than 30%/70%.
If we want to further improve our model, we would suggest making a move from LDA to QDA, and hence not assuming equal co-variance matrices; or perhaps even combining the two in a more exotic hybrid model.
An error source in our model could be that annotations are manually found for only areas which can be specifically distinguished to be either meat or fat. This introduces an inherent bias in the model, which further introduces uncertainty for the found results. It is also further assumed that the salami has a uniform distribution of fat and meat through its depth, this might not be the case, but can not be determined completely accurately based on our 2D cross section analysis.

1.5 Conclusion

Our goal was to investigate the possibilities of determining fat content in a salami based on a single 19-band image of a piece of salami. We modelled the fat content using both a simple threshold model and a model based on linear discriminant analysis. We found that if we train the model on day 01, it turns out the LDA model is more precise. The overall optimal model is obtained when we train the LDA model on day 28 where we obtain a relative error of 0.0556. Based on qualitative assessment of pictures of classified fat, the model seems to classify most fat successfully. We found that adding priors did not decrease the error rate on the annotated data, we expect the reason for this is because the annotated data is not a 30/70 percent slit, as it would be in the salami itself.

Appendices

APPENDIX 1

A.0.1 Appendix 0.1 Matlab - discriminant function

```
function S = S(multiIm, SigmaInv, mu,p)
    % Generate matrix of right dimensions
    Xim = double(reshape(multiIm, size(multiIm,1)*size(multiIm,2), size(multiIm,3)))';
    % Do matrix multiplications
    S = Xim'*SigmaInv*mu - 1/2*mu'*SigmaInv*mu+log(p);
    %Reshape matrix to origianl dimensions
    S = reshape(S,size(multiIm,1),size(multiIm,2),1);
end
```

A.0.2 Appendix 0.2 Matlab - main script

```
%% Simple model thresholds
clear all
% Folder where your data files are placed
%dirIn = 'C:\Users\Garsdal\Desktop\02526 Mathematical Modelling\Salami\';
%dirIn = '/Users/mads/Google Drev/Skole/Uni/6. Semester/Mathematical Modelling/Projekt 1 - ...
    salamis/Data/';
dirIn ='C:\Users\Nicolaj\OneDrive - Danmarks Tekniske Universitet\DTU mapper\4. ...
    semester\Matematiskmod\Exercises\Salami\';

% Load multiIm day 1 and
[multiIm, annotationIm] = loadMulti('multispectral_day01.mat' , 'annotation_day01.png', dirIn);

% We define fat and meat pixels
[fatPix, fatR, fatC] = getPix(multiIm, annotationIm(:,:,2));
[meatPix, meatR, meatC] = getPix(multiIm, annotationIm(:,:,3));

% Simple model ¬ assume normal distribution mu_f / mu_m with same variance
% Threshold is found as intersection of normal distributions:
threshold = mean(fatPix)/2 + mean(meatPix)/2

%% Plotted thresholds and distributions

h = showHistograms(multiIm, annotationIm(:,:,2:3), 12, 1);
axis square
hold on
xline(threshold(12))
title("Spectral image 12")

j = showHistograms(multiIm, annotationIm(:,:,2:3), 3, 1);
axis square
hold on
xline(threshold(3))
title("Spectral image 3")

%% Error rate calculation for each band
error = zeros(1,19);
error_f = zeros(1,19);
error_m = zeros(1,19);
error_tot = zeros(1,19);
for i = 1:19
    % We apply the threshold
    multiImMeat = multiIm(:,:,i) < threshold(i);
    multiImFat = multiIm(:,:,i) ≥ threshold(i);
```

```

    % We remove all the background that is not in annotationIm for meat/fat
    multiImMeat(annotationIm(:,:,3)==0) = 0;
    multiImFat(annotationIm(:,:,2)==0) = 0;

    % We count all pixels classified as meat / fat
    cnt_meat = sum(sum(multiImMeat));
    cnt_fat = sum(sum(multiImFat));

    % We determine the error from the correct pixels in annotationIm
    % abslute deviation in meat/fat divided by total meat/fat pixels

    %error_fat = abs(cnt_fat-sum(annotationIm(:,:,2),'all'))/sum(annotationIm(:,:,2:3),'all')
    %error_meat = abs(cnt_meat-sum(annotationIm(:,:,3),'all'))/sum(annotationIm(:,:,2:3),'all')
    %error(1,i) = abs(cnt_meat-sum(annotationIm(:,:,3),'all'))/sum(annotationIm(:,:,2:3),'all');

    % different error calculation
    error_m(1,i) = abs(cnt_meat - sum(annotationIm(:,:,3),'all')) / ...
        sum(annotationIm(:,:,3),'all');
    error_f(1,i) = abs(cnt_fat - sum(annotationIm(:,:,2),'all')) / ...
        sum(annotationIm(:,:,2),'all');
    error_tot(1,i) = (error_m(1,i)+error_f(1,i))/2;
end

%optimal_band = find(error == min(error))
optimal_band = find(error_tot == min(error_tot))

%% Characterization using optimal band

% We apply the threshold
multiImCharM = multiIm(:,:,optimal_band) < threshold(optimal_band);
multiImCharF = multiIm(:,:,optimal_band) ≥ threshold(optimal_band);

% We remove all background
multiImCharM(annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3)==0) = 0;
multiImCharF(annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3)==0) = 0;

% We count all pixels classified as meat / fat
cnt_meat = sum(multiImCharM, 'all');
cnt_fat = sum(multiImCharF, 'all');
cnt_total = sum(annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3), 'all');

% We determine the ratio of meat / fat pixels
ratio_meat = cnt_meat/cnt_total
ratio_fat = cnt_fat/cnt_total

%% Import all data in cell

multiImC = cell(5,1);
annotationC = cell(5,1);
imageC = cell(5,1);
dataNo = ["01", "06", "13", "20", "28"];
str1 = 'multispectral_day01.mat';
str2 = 'annotation_day01.png';
str3 = 'color_day01.png';
for i = 1:5
    str1(18:19) = dataNo(i);
    str2(15:16) = dataNo(i);
    str3(10:11) = dataNo(i);
    [multiImC{i},annotationC{i}] = loadMulti(str1 , str2, dirIn);
    imageC{i} = imread([dirIn str3]);
end

% all data is stored in cells. Indexing using with {}.
%% Characterization of remaining days

% Use the above characterization and put in a for loop
% Use the trained model for day1 on remaining days

relErrSimple = zeros(5,1);
```

```
for i=1:5
    % We apply the threshold
    multiImCharM = multiImC{i}(:,:,optimal_band) < threshold(optimal_band);
    multiImCharF = multiImC{i}(:,:,optimal_band) ≥ threshold(optimal_band);

    % We remove all background
    multiImCharM(annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3)==0) = 0;
    multiImCharF(annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3)==0) = 0;

    % sums classified fat within annotated fat
    classified_fat = sum(multiImCharF(annotationC{i}(:,:,2)==1),'all');
    classified_meat = sum(multiImCharM(annotationC{i}(:,:,3)==1),'all');
    % sums annotation
    sumAnnotation_fat = sum(annotationC{i}(:,:,2),'all');
    sumAnnotation_meat = sum(annotationC{i}(:,:,3),'all');

    % calculates the relative error
    relErr_fat = (sumAnnotation_fat-classified_fat)/sumAnnotation_fat;
    relErr_meat = (sumAnnotation_meat-classified_meat)/sumAnnotation_meat;
    relErrSimple(i) = (relErr_fat+relErr_meat)/2;
end



%% Multivariate linear discriminant

% We define fat and meat pixels
[fatPix, fatR, fatC] = getPix(multiIm, annotationIm(:,:,2));
[meatPix, meatR, meatC] = getPix(multiIm, annotationIm(:,:,3));

% Define parameters
X = [fatPix' meatPix'];
Sigma = cov(X');
SigmaInv = inv(Sigma);
mu_fat = mean(fatPix)';
mu_meat = mean(meatPix)';
prior = 1;

% Generate Si matricies
S_fat = S(multiIm,SigmaInv,mu_fat,prior);
S_meat = S(multiIm,SigmaInv,mu_meat,prior);

% Generate annotations
mask = (annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3))==1;
dif = (S_fat-S_meat).*mask;
annotation_fat = (dif≥0).*mask;
annotation_meat = (dif<0).*mask;


% Fat ratio
fat_ratio = sum(annotation_fat,'all')/(sum(annotation_fat,'all')+sum(annotation_meat,'all'));

%% QDA
clear all
% Folder where your data files are placed
dirIn = '/Users/mads/Google Drev/Skole/Uni/6. Semester/Mathematical Modelling/Projekt 1 - ...
    salamis/Data/';

% Load multiIm day 1 and
[multiIm, annotationIm] = loadMulti('multispectral_day20.mat' , 'annotation_day20.png', dirIn);

% We define fat and meat pixels
[fatPix, fatR, fatC] = getPix(multiIm, annotationIm(:,:,2));
[meatPix, meatR, meatC] = getPix(multiIm, annotationIm(:,:,3));

% Define parameters
Sigma_fat = cov(fatPix);
Sigma_fat_inv = inv(Sigma_fat);
```

```
Sigma_meat = cov(meatPix);
Sigma_meat_inv = inv(Sigma_meat);
mu_fat = mean(fatPix)';
mu_meat = mean(meatPix)';
p_fat = 1;
p_meat = 1;
prior = 1;


% Lets try another image
[multiIm, annotationIm] = loadMulti('multispectral_day13.mat' , 'annotation_day13.png', dirIn);

%Iterative approach
f_fat = zeros(size(multiIm,1),size(multiIm,2));
f_meat = zeros(size(multiIm,1),size(multiIm,2));
aux_fat = 1/(sqrt(2*pi)^(19)*sqrt(det(Sigma_fat)));
aux_meat = 1/(sqrt(2*pi)^(19)*sqrt(det(Sigma_meat)));
for j=(1:size(multiIm,1))
    for i=(1:size(multiIm,2))
        x = double(reshape(multiIm(j,i,:),19,1,1));
        f_fat(j,i) = aux_fat*exp(-1/2*(x-mu_fat)'*Sigma_fat_inv*(x-mu_fat))*p_fat;
        f_meat(j,i) = aux_meat*exp(-1/2*(x-mu_meat)'*Sigma_meat_inv*(x-mu_meat))*p_meat;
    end
end

% Generate annotations
mask = (annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3))==1;
dif = (f_fat-f_meat);
annotation_fat = (dif≥0).*mask;
annotation_meat = (dif<0).*mask;

% Fat ration
fat_ratio = sum(annotation_fat,'all')/(sum(annotation_fat,'all')+sum(annotation_meat,'all'));

%% Show RGB
% Load RGB image
imRGB = imread([dirIn 'color_day13.png']);

% Pixel coordinates for the fat annotation
[fatPix, fatR, fatC] = getPix(multiIm, annotation_fat);

% Concatenate the pixel coordinates to a matrix
pixId = [fatR, fatC];

% Make the new images
rgbOut = setImagePix(imRGB, pixId);
figure
imagesc(rgbOut)

%% Train on day 1 make image visulization and relative error

annotation_fat = cell(5,1);
annotation_meat = cell(5,1);
relErr = zeros(1,4);
for i = 1:5
    S_fat = S(multiImC{i},SigmaInv,mu_fat,prior);
    S_meat = S(multiImC{i},SigmaInv,mu_meat,prior);
    mask = (annotationC{i}(:,:,1)+annotationC{i}(:,:,2)+annotationC{i}(:,:,3))==1;
    dif = (S_fat-S_meat).*mask;
    annotation_fat{i} = (dif≥0).*mask;
    annotation_meat{i} = (dif<0).*mask;

    % sums classified fat within annotated fat
    classified_fat = sum(annotation_fat{i}(annotationC{i}(:,:,2)==1),'all');
    classified_meat = sum(annotation_meat{i}(annotationC{i}(:,:,3)==1),'all');

    % sums annotation
    sumAnnotation_fat = sum(annotationC{i}(:,:,2),'all');
    sumAnnotation_meat = sum(annotationC{i}(:,:,3),'all');

```

```
    % calculates the relative error
    relErr_fat = (sumAnnotation_fat-classified_fat)/sumAnnotation_fat;
    relErr_meat = (sumAnnotation_meat-classified_meat)/sumAnnotation_meat;
    relErr(i) = (relErr_fat+relErr_meat)/2;
end

str4 = 'Day 00, trained with day 01';
figure
for i = 1:5
    str4(5:6) = dataNo(i);
    %subset fat
    [fatPix, fatR, fatC] = getPix(multiImC{i}, annotation_fat{i});
    % Concatenate the pixel coordinates to a matrix
    pixId = [fatR, fatC];
    % Make the new images
    rgbOut = setImagePix(imageC{i}, pixId);
    subplot(1,5,i), imagesc(rgbOut)
    title(str4);
end

figure
for i = 1:5
    str4(5:6) = dataNo(i);
    %subset fat
    [meatPix, meatR, meatC] = getPix(multiImC{i}, annotation_meat{i});
    % Concatenate the pixel coordinates to a matrix
    pixId = [meatR, meatC];
    % Make the new images
    rgbOut = setImagePix(imageC{i}, pixId);
    subplot(1,5,i), imagesc(rgbOut)
    title(str4);
end


%% Train on one day evaluate the rest
str4 = 'Day 00, trained with day 01';
relErr = zeros(5,5);
ratio_meat = zeros(5,5);
ratio_fat = zeros(5,5);
annotation_fat = cell(5,1);
annotation_meat = cell(5,1);


for i=1:5
    % We define fat and meat pixels
    [fatPix, fatR, fatC] = getPix(multiImC{i}, annotationC{i}(:,:,2));
    [meatPix, meatR, meatC] = getPix(multiImC{i}, annotationC{i}(:,:,3));

    % Define parameters
    X = [fatPix' meatPix'];
    Sigma = cov(X');
    SigmaInv = inv(Sigma);
    mu_fat = mean(fatPix)';
    mu_meat = mean(meatPix)';

    for j = 1:5
        S_fat = S(multiImC{j},SigmaInv,mu_fat,1);
        S_meat = S(multiImC{j},SigmaInv,mu_meat,1);
        mask = (annotationC{j}(:,:,1)+annotationC{j}(:,:,2)+annotationC{j}(:,:,3))==1;

        dif = (S_fat-S_meat).*mask;
        annotation_fat{j} = (dif≥0).*mask;
        annotation_meat{j} = (dif<0).*mask;

        % measure the fat and meat ratio:

        cnt_meat = sum(annotation_meat{j}, 'all');
        cnt_fat = sum(annotation_fat{j}, 'all');
        cnt_total = sum(annotationC{j}(:,:,1)+annotationC{j}(:,:,2)+annotationC{j}(:,:,3), ...
            'all');
```

```

        ratio_meat(i,j) = cnt_meat/cnt_total;
        ratio_fat(i,j) = cnt_fat/cnt_total;

        % sums classified fat within annotated fat
        classified_fat = sum(annotation_fat{j}(annotationC{j}(:,:,2)==1),'all');
        classified_meat = sum(annotation_meat{j}(annotationC{j}(:,:,3)==1),'all');

        % sums annotation
        sumAnnotation_fat = sum(annotationC{j}(:,:,2),'all');
        sumAnnotation_meat = sum(annotationC{j}(:,:,3),'all');

        % calculates the relative error
        relErr_fat = (sumAnnotation_fat-classified_fat)/sumAnnotation_fat;
        relErr_meat = (sumAnnotation_meat-classified_meat)/sumAnnotation_meat;
        relErr(i,j) = (relErr_fat+relErr_meat)/2;
    end


    % computes plot with classified fat:

    str4((end-1):end) = dataNo(i);
    %{
    figure
    for j = 1:5
        str4(5:6) = dataNo(j);
        %subset fat
        [fatPix, fatR, fatC] = getPix(multiImC{j}, annotation_fat{j});
        % Concatenate the pixel coordinates to a matrix
        pixId = [fatR, fatC];
        % Make the new images
        rgbOut = setImagePix(imageC{j}, pixId);
        subplot(1,5,j), imagesc(rgbOut)
        title(str4);
    end
    %}

    % computes plot with classified meat:

    figure
    for j = 1:5
        str4(5:6) = dataNo(j);
        %subset fat
        [meatPix, meatR, meatC] = getPix(multiImC{j}, annotation_meat{j});
        % Concatenate the pixel coordinates to a matrix
        pixId = [meatR, meatC];
        % Make the new images
        rgbOut = setImagePix(imageC{j}, pixId);
        subplot(1,5,j), imagesc(rgbOut)
        title(str4);
    end

end

% takes out points where data is compared with itself
errRate = relErr.*(¬eye(5)) + diag([0.0075 0.0075 0.0025 0.0025 0.0174]);
figure
heatmap(errRate);
xlabel("Relative error");
ylabel("Traning day")
sumRow = sum(errRate, 2);
```

A.0.3 Appendix 0.3 Matlab - Set Aside training

```
clear all
% Folder where your data files are placed
```

```
dirIn = '/Users/mads/Google Drev/Skole/Uni/6. Semester/Mathematical Modelling/Projekt 1 - ...
    salamis/Data/';
% Load multiIm
MI = 'multispectral_day28.mat';
AD = 'annotation_day28.png';
[multiIm, annotationIm] = loadMulti(MI , AD, dirIn);

% We define fat and meat pixels
[fatPix, fatR, fatC] = getPix(multiIm, annotationIm(:,:,2));
[meatPix, meatR, meatC] = getPix(multiIm, annotationIm(:,:,3));
mask = (annotationIm(:,:,1)+annotationIm(:,:,2)+annotationIm(:,:,3))==1;

% Shuffel the data
fatPix = fatPix(randperm(size(fatPix, 1)), :);
meatPix = meatPix(randperm(size(meatPix, 1)), :);

fatTrain = fatPix;
fatTest = fatPix;

meatTrain = meatPix;
meatTest = meatPix;

%fatTrain = fatPix(1:(size(fatPix,1)-200),:);
%fatTest = fatPix((size(fatPix,1)-200):end,:);

%meatTrain = meatPix(1:(size(meatPix,1)-200),:);
%meatTest = meatPix((size(meatPix,1)-200):end,:);


% Define parameters
X = [fatTrain' meatTrain'];
Sigma = cov(X');
SigmaInv = inv(Sigma);
mu_fat = mean(fatTrain)';
mu_meat = mean(meatTrain)';


%    ##### ER FOR FAT #####
S_fat = fatTest*SigmaInv*mu_fat - 1/2*mu_fat'*SigmaInv*mu_fat;
S_meat = fatTest*SigmaInv*mu_meat - 1/2*mu_meat'*SigmaInv*mu_meat;
dif = (S_fat-S_meat);
annotation_fat = (dif≥0);

fatER = 1 - sum(annotation_fat,'all')/size(fatTest,1);

%    ##### ER FOR MEAT #####
S_fat = meatTest*SigmaInv*mu_fat - 1/2*mu_fat'*SigmaInv*mu_fat;
S_meat = meatTest*SigmaInv*mu_meat - 1/2*mu_meat'*SigmaInv*mu_meat;
dif = (S_meat-S_fat);
annotation_meat = (dif>0);

meatER = 1 - sum(annotation_meat,'all')/size(meatTest,1);

ER = (fatER+meatER)/2
```