PATTERN RECOGNITION

NAIVE BAYES CLASSIFIER FOR DAY AND NIGHT IMAGES WITH MATLAB

Tasoulas Theofanis

Corfu, June of 2018

a) Synopsis

Pattern recognition is a field of computer science, that classifies a pattern in categories we called classes. This project is a basic naive Bayes classifier, which distinguishes images of day from night. By adding 40 images of day and night a plot is created, based on the mean brightness and the variance of brightness of each image. With that plot there will be a classification for each new image. In addition, it's always important to minimize the probability of error, by increasing the number of the images in database and the entry parameters. Last but not least, the risk assessment is the identification of the worst case of classification. This will help to avoid the most harmful consequences of a wrong classification.

In a contrary, the steps for a pattern recognition project are:

1) Knowledge/data base implementation

2) Determination of measurable characteristics

3) Definition of classes

4) Making decisions (including minimizing mistakes)

5) Risk assessment

For this project we use Matlab. We collect 20 photos of 20 days and 20 nights. For each of these, the RGB (Red, Green, Blue) color model was transformed into HSV (Hue, Saturation, Value). At this point, it should be noted that each image consists of 3 two-dimensional tables or we may say 3 layers. In the RGB template, the 1st layer contains the numeric values of the red for an image, the second contains the green and the third one the blue. Correspondingly to the HSV model, the three levels of the panel are representing the image's hue, intensity and brightness values (the third level is called the "value", implying the brightness values).

In order to classify an image as day or night, we produce the average brightness value of the HSV-formatted images. With these values, we plot the two-dimensional graph. This represents the distribution of the frequency (of occurrences) of an image by their mean brightness value. To put it simple, we plot the how many times this image appeared" by a certain amount of brightness. Then, we make conclusions about the separation in classes of day and night. With this way, when the algorithm will receive a new image, he will classify it to the appropriate class. Finally, we add the variance of brightness as a second parameter, in order to minimize the probability of error. Based on these two features, we also draw out a three-dimensional graph, which describes the distribution of frequency, by the mean brightness and by its variance.

b) Code analysis

Moving on to the analysis of the algorithm, it should be noted that the code consists of separated sections. The first part of the algorithm initializes the necessary vectors/tables, to assign the appropriate values from the images.

```
% Initialization
DayBr=[];              % Day Brightness Values
DayVa=[];              % Day Variance Values
NightBr=[];            % Night Brightness Values
NightVa=[];            % Night Variance Values
```

In the second part, a for loop from 1 to 20 starts, so that the 20 images are read by Matlab with the “imread” command. This coomand accepts a) the image address along with the initial letter of each image (d for day day, n for night and night), b) the number of repeat/loop and c) the file type of the images (they all are . jpg). After an image is read, the “rgb2hsv” command converts the color model from RGB to HSV. Then we take the values of the third layer of the image, which is actually the values of brightness. Then, the mean value is calculated and it gets stored to its matrix/vector. This is accomplished, by using each function into itself (nested). That is because each layer of the image is a two-dimensional array (lines and columns). Therefore, by using the function once, we derive a result for each column individually, creating a vertical array of numeric values. So with the second use, we find the desired result. Finally, the values of the respective tables are stored into their matrixes/vectors. The same goes for both of images of night and day.

```matlab
% Read the images files and take the necessary values
for i=1:20

    img_color = imread(['D:\day-night\n' num2str(i) '.jpg']);

    hsv_img = rgb2hsv(img_color);
    value = hsv_img(:,:,3);
    brightness=mean(mean(value));
    NightBr=[NightBr,brightness];
    Variance = var(var(value));
    NightVa=[NightVa,Variance];
end;
for i=1:20
    img_color = imread(['D:\day-night\d' num2str(i) '.jpg']);
    hsv_img = rgb2hsv(img_color);
    value = hsv_img(:,:,3);
    brightness=mean(mean(value));
    DayBr=[DayBr,brightness];
    Variance = var(var(value));
    DayVa=[DayVa,Variance];
end;
```

The third part of the algorithm is optional and it rounds the tables that were just created. This is done by the “round” function, where the first term is the table and the second the number of digits after subdivision - in this case 1.

```matlab
% That step is optional and it limits decimal points to 1
% (for example the number 5.65128 is being converted to 5.7)
round(NightVa,1);
round(NightBr,1);
round(DayVa,1);
round(DayBr,1);
```

The next section creates a two-dimensional plot. It displays a window with the “shape” command and adds a history of campaigns with daily brightness values through the “histfit” function. Then the same is done for the images of the night. Furthermore, the “grid” is used, in order to draw some lines and make the chart more “readable”. With “xlim ([0 1])”, the x axis is being displayed limited from 0 to 1. Last but not least, the “ylabel” and “xlabel” commands are giving a name on each axis. After the plot is completed, we may delete the history bars and keep only the distribution (the red lines).

```
% Make the 2D chart of Brightness Vs Frequency of Occurrence
figure
histfit(DayBr)
grid on
hold all
histfit(NightBr)
xlim ([0 1])
ylabel 'Frequency of Occurrence'
xlabel 'Brightness'
```

The fifth part reproduces the average brightness values per variance brightness, with commands similar to the previous ones. For each grid of the new drawing, we count the red 'x' symbols of it. The number of red ‘x’ is the frequency of night images per grid. Then we write in an excel file 1) the average brightness (column 1), 2) its variance (column 2) and 3) the frequency of night images for each grid. This is the “NIGHT.csv” file. We do the same for the blue ‘x’ in each grid for the DAY.csv. This represents the day images values. Note here, that this process isn’t made on Matlab, but we do it manually.

```
% Make the 2D chart of Brightness Vs Variance
figure
plot(DayBr,DayVa,'x')
grid on
hold all
plot(NightBr,NightVa,'x')
grid on
hold all
xlabel 'Brightness mean'
ylabel 'Brightness variance'
```

In the fifth section, the new excel file is read and the column’s values are assigned to new variables/matrixes.

```
% Read the files that contains the Brightness per Variance values + the number of 'x' signs per grid
DAY = csvread('D:\day-night\DAY.csv');
NIGHT = csvread('D:\day-night\NIGHT.csv');
```

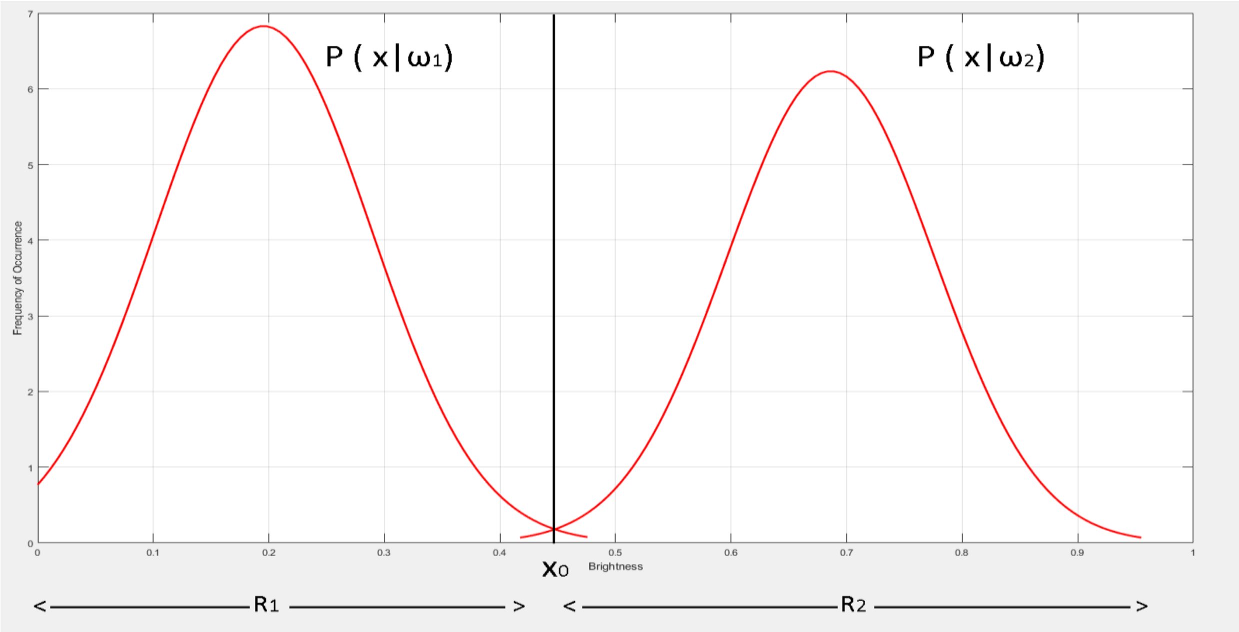
```
% Define collumns
a=NIGHT(:,1);                % Take the repetitively Brightness values
b=NIGHT(:,2);                % Take the repetitively Variance values
c=NIGHT(:,3);                % Take the number of 'x' signs
d=DAY(:,1);
e=DAY(:,2);
f=DAY(:,3);
```

Finally, the three-dimensional diagram, which contains the average value of brightness per variance of brightness per the frequency of the images, is formed. This is implemented by the “load franke” command, where it loads the Franke bivariate function. After that, “fit” locates the appropriate data transparency in a linear fashion (accepts as the third parameter the “linear” font) and the plot draws the graph.

```
% Make the 3D chart
figure
load franke                          % Franke bivariate test function
sf = fit([a, b],c,'linear')          % Fit linear surface to data
plot(sf,[a,b],c)                     % Creates a 3D plot of a set of data points
hold on
sf = fit([d, e],f,'linear')
plot(sf,[d,e],f)
hold off                             % Sets the hold state to off so that new plots added to the axes
                                     % clear existing plots and reset all axes properties
```

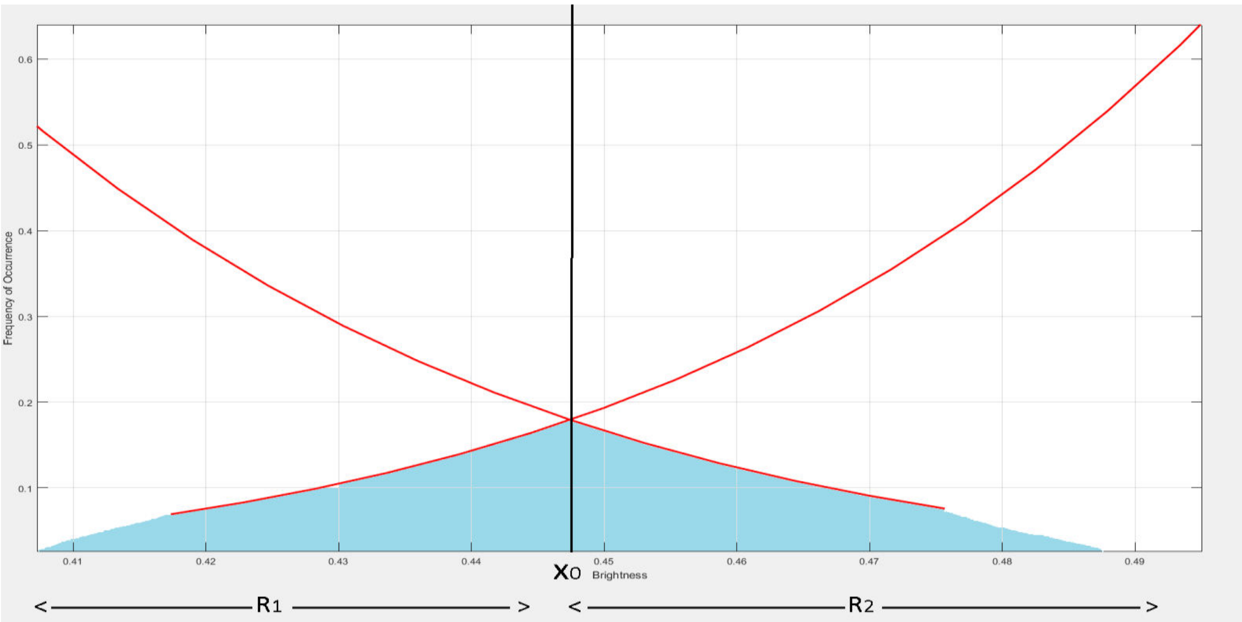
c) Explanation of the plots

The result of the first graph is two curves. The right-hand curve expresses the frequency of the average brightness for the day images and the left one for the night images. This is reasonable since the day images are obviously brighter than the night images. Usually in real problems there is no ideal outcome. Similarly to the present one there is no equivalence in the high of the curves. The fact that the right curve is a bit lower (in high) than the right, indicates the most common low brightness appearance in a picture. In other words, the average picture (both of day and night) is darker. If we change the images, there might be the opposite result. Moreover, the result depends on the number of images. By adding more images (as long as the images are “good”), the results improves and obviously (but not surely) the curves decrease their deflection.



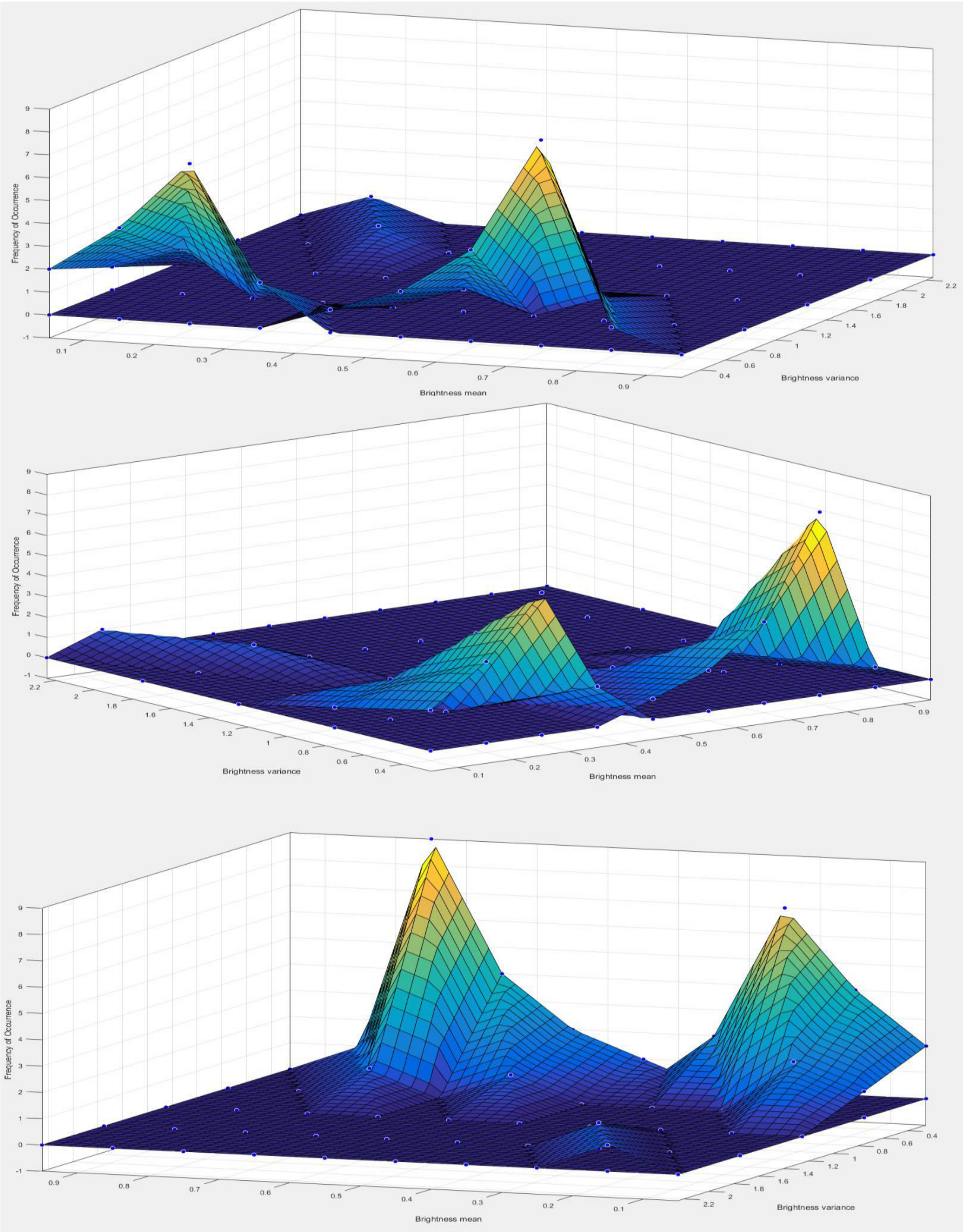
The vector x represents the average brightness. The black vertical line at x0 is a threshold, separating the chart in two regions R1 and R2, with their center at x0. As a rule on the Bayes classifier, when the x of an image belongs to R1, it will be classed as ω1. In our case, it will be considered as a night image. If it belongs to R2, it will be classified as a day image in ω2 class.

However, if it belongs to the region where the two curves collide, then the day or night class is undefined. This is the probability of an error. The following graph is an enlargement of the above, with additional coloring in the error area.

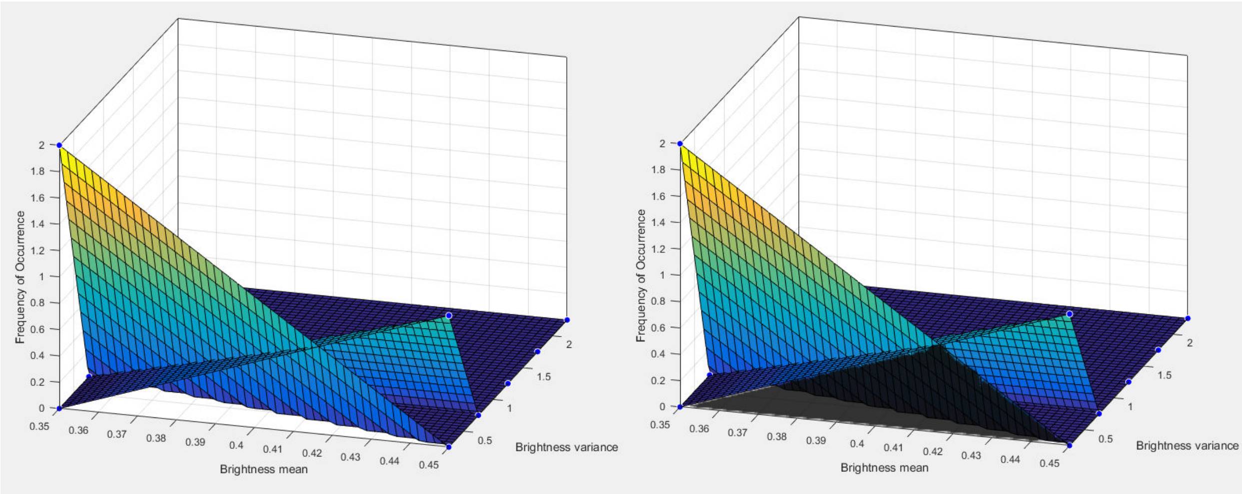


Note that the R, P (X | ω), X0 and threshold were not extracted from Matlab
Instead they were designed in Photoshop

Subsequently, the three-dimensional layout is displayed on three different angles in the images below. Keep in mind that the x axis is the average brightness, y is its variance and z is the frequency of images. Therefore, the surface formed represents the probability of an image, belonging to the day or night class, correlating with the values of the x,y,z axis.

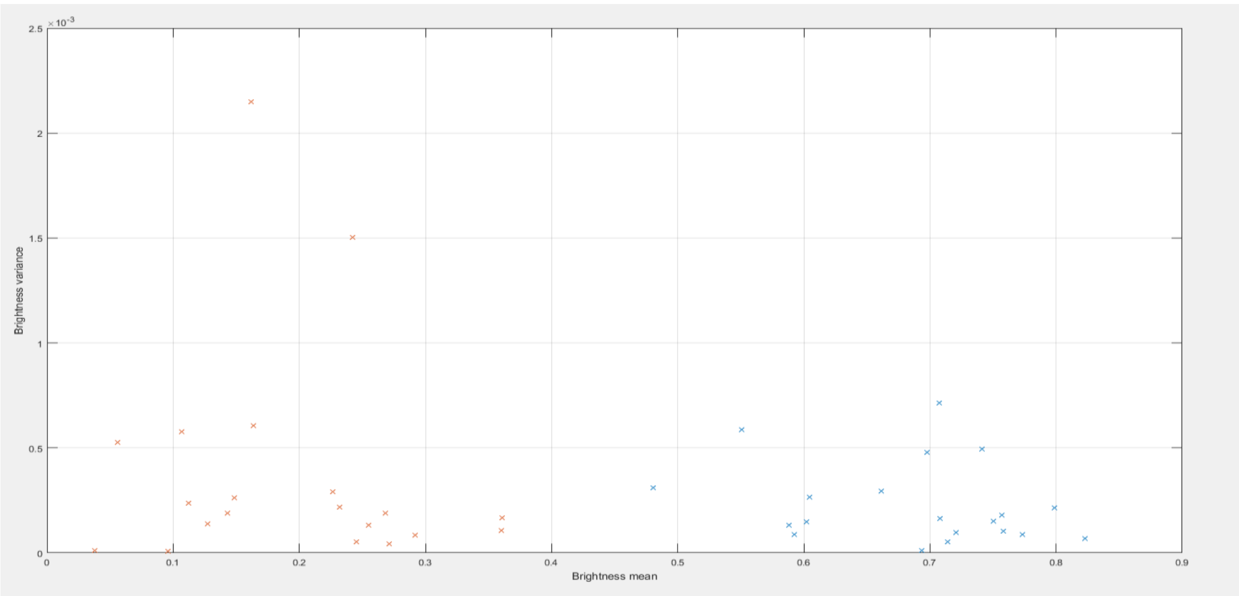


Remarkable is the blank area created at average brightness values from 0.35 to 0.45 (the right image shows this area shadier). This represents the new probability of error.



By comparing the two error probabilities, it is concluded that if the probability of error was reduced by adding the new parameter (the brightness variance), then the new parameter is a good example for detecting day or night in a photo. However, the opposite would mean that this feature is not a good example and that it shouldn't be used on classifying images of day and night.

Now the previous 3D charts came by the plot below. This just displays the number of day images (blue x) and the number of night images (red x) per grid. The grid separates the area 50 times, because we counted 10 mean brightness values and 5 of its variance. There are not many significant conclusions from this plot, it is just displayed because it was a part of the project.



d) Risk assessment

In the last part of this project, we evaluate the risk. More specifically we incorporate the probability of error into a class, in order to avoid the maximum possible damage. For instance, if a new image's variance = 0 and mean brightness = 0.45, that probably means that there is no clear classification (probability of error). In this case, if we consider that the Night class is the most harmful class, then the image will be ranked in the night class. However, if the day class is considered more damaging, the opposite will happen. Certainly, in the classification of day or night images, the point of the evaluation of risk is unclear. So let's move on to another example. In a tumor classifier, which distinguishes whether a patient has a benign or malignant tumor, every decision is critical. If the patient has a benign tumor (in reality) and the classifier categorizes the patient in the malignant class, there will be some financial charges for medicines and treatments and examinations. But there will, certainly, not be any irreversible consequences. On the contrary, if there is actually a malignant tumor, but the algorithm classifies the tumor as benign, it may even lead to death, because there will be no proper care. Consequently, in order to avoid the second and most damaging case, the probability of error must match to the "malignant tumor" class.

Network sources – Bibliography

Αναγνώριση Προτύπων, S. Theodoridis, K. Koutroumbas, 2011, ΕΚΔΟΣΕΙΣ ΠΑΣΧΑΛΙΔΗΣ

https://opencourses.ionio.gr/courses/DDI133/

https://www.mathworks.com/help/matlab/