FUTUREMATH ERASMUS+ PROJECT

Co-funded by the Erasmus+ Programme of the European Union

UNIVERZITET METROPOLITAN BEOGRAD

CS374 - VEŠTAČKA INTELIGENCIJA

# Classification in machine learning

Lekcija 16

PRIRUČNIK ZA STUDENTE

# CS374 - VEŠTAČKA INTELIGENCIJA

## Lekcija 16

## *CLASSIFICATION IN MACHINE LEARNING*

# Uvod

## INTRODUCTION

*In this lesson students will learn about classification basics and how to project Bayesian classifier. Two classification problems are presented and solved theoretically and practically.*

In this lecture we will first introduce:

- Basic concepts of classification
- Definition of classifier
- Choice of classifier

After that, two classification problems are given and solved.

First classification problem is binary classification of artificially generated data with Bayesian classifier. Mathematical concepts for solving of this type of classification problem are presented. Solution is implemented in programming language Python.

Second problem is classification of Iris dataset with Bayesian classifier. Also, mathematical concepts for solving of this type of classification problem are presented and solution is implemented in Python.

Previous knowledge assumed for this lecture is: Basics of calculus, basics of linear algebra, basics of statistics and basic of Python programming.
 <youtube width="460" height="258" src="https://drive.google.com/file/d/1iWl7d6lFhnE9jRSJl_FcztkEzN4h2kDa/preview" /> 

# Poglavlje 1

# Classification

## BASIC CONCEPTS OF CLASSIFICATION

*Basic concepts of classification are presented in this section.*

Classification is supervised machine learning technique which is widely used in many real world problems.

Basic question: How to separate data on classes?

Example: I have a dataset with images of cats and dogs. If a new image arrives, will my model be able to classify it? Will my model be trained well to recognize new image that wasn't in dataset that model was trained on?

How can we perform a classification? We can separate our data with function that splits data with minimal error.

There are many classifiers that we can use for solving a particular problem: Bayesian classifier, logistic regression, support vector machine, neural networks, decision trees...

Actually, we are looking for function that splits data on classes. On the image right, which function is better?

Question is how to find classification function? We need to optimize parameters of some function, or to make some estimation of the function distribution. We must have labeled data for parameter estimation.


Slika 1.1 Two classes and classification line [1]

**Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.**

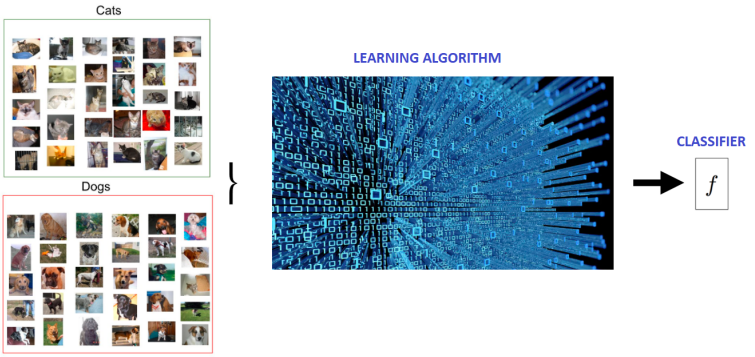
# DEFINITION OF CLASSIFIER

## *In this section classifier is defined.*

Classification is supervised learning problem because data must be labeled for parameter optimization. For each sample we must have information about class belonging.

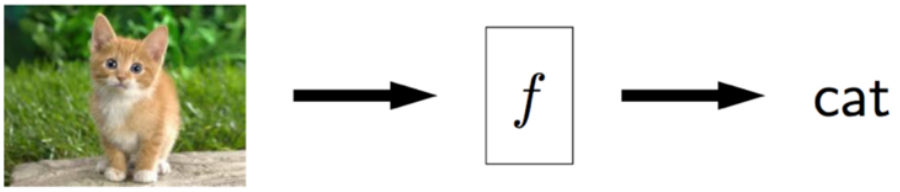
When we say classifier we actually mean on function **(f)** that maps some input vector **(X)** to categorical output **y**. We have two phases.

In phase I (learning phase) learning algorithm learns on training dataset, we have **X** and y and we want to learn **f**.




Slika 1.2 Learning phase [2], [3]

In phase II (use) we have **X** (new data that model was not trained on) and **f** and we wish to compute predicted value of **y**.




Slika 1.3 Implementation of classifier on new data [3]

Output of classification is always categorical.

Examples: Approve credit-YES or NO. Scrap product-YES or NO. Subject of paper article-SPORT, POLITICS, CULTURE. Patient is sick-YES or NO.

There are many different classifiers that uses different techniques:

- Probability (Naive Bayes, Bayes net)
- Rules (Decision trees, Rules generators)
- Functions (Regression, Support vector machine, Neural networks)

- Instances (K nearest neighbors)

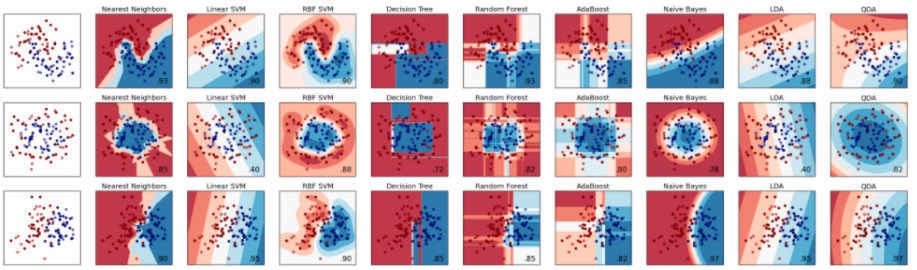
What is the difference?

All of these classifiers split space differently.

# BIAS/VARAINCE TRADEOFF

*In this section bias/varaince tradeoff is described.*

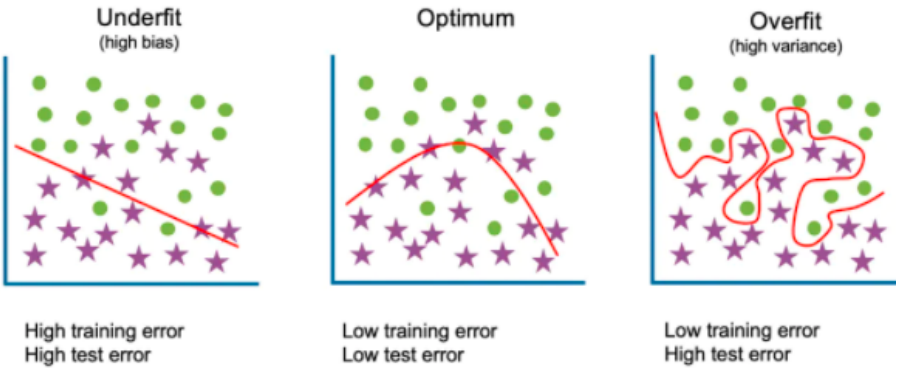
On the image below we can see that different classifiers split space differently. Question is how to choose the best classifier for our problem? We want model that explains well enough training data, but also model must be generalized.




Slika 1.4 [4]

Model must be good to explain the problem well, but also it must be efficient on new data. Underfitting and overfitting must be avoided for good results.

In literature avoiding of underfitting and overfitting is known as bias/variance tradeoff.




Slika 1.5 [5]

**Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.**

# CHOICE OF CLASSIFIER

*In this section concepts of classifier choise are described.*

Question is how to choose best classifier?
Trough iterative testing process
We always test our model on new data (data that model wasn't trained on)
How do we measure success of classifier?
With different error measurements. We measure how many samples are well classified.
Interpretation-how can we explain our model.
Implementation-how easy or difficult is to implement the classifier and maintain the classifier.
Activity that precedes the classification is choice of descriptors that describe in the best way data that we want to classify. These descriptors are called features.

Usually there are several features, and they are put in one random vector. Such vector is called pattern in pattern recognition theory [6].

Success of classifier depends on good choice of features.

**Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.**

**Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.**

# Poglavlje 2

# First classification problem

## BINARY CLASSIFICATION OF ARTIFICIALLY GENERATED DATA

*In this section first practical problem in this lesson is posted.*

First practical problem: Binary classification of artificially generated data.

Two classes with two-dimensional shapes are given with normal probability density functions:

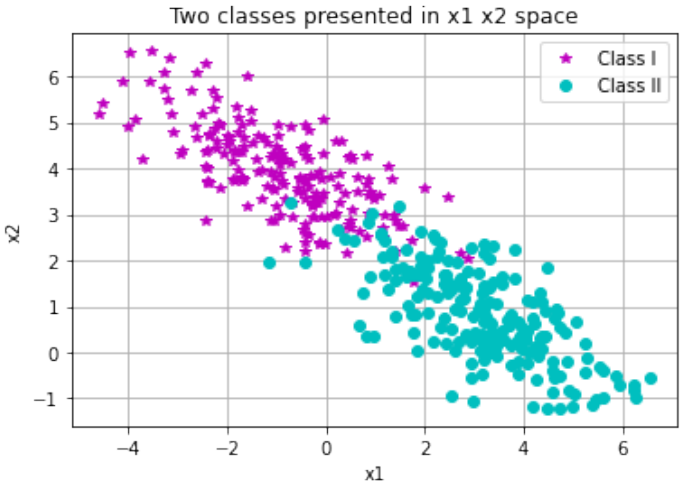
$$f_1(x) = N(M_1, \Sigma_1), M_1 = \begin{bmatrix} -1 \\ 4 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}$$

$$f_2(x) = N(M_2, \Sigma_2), M_2 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}$$

Task: Assuming that prior probabilities of classes are same, generate 200 samples for each class and project a minimal error Bayesian classifier. Visualize data and discrimination function h(x)=0. Calculate Type I error and Type II error and accuracy of classifier.

On the Figure 1, 400 samples from two classes (200 samples from each class) in 2D space with given probability density functions and generated. Software solution for generating of samples and their representation in 2D space will be given later.

Our task is to find classification line that will separate our data in two classes with minimal error. This line is called discrimination function and in our case it is Bayesian classifier.




Slika 2.1 Two classes in 2D space [Source: Author]

In this case we have binary classification, because we have classification on two classes. Our data are artificial, it means that we generated them based on given probability density functions and number of samples for each class. Also, we can see that our classes are not separable, it means that we have overlapping between classes. Thus, we can expect classification error.

Our classes have same covariance matrix (with negative covariances) and we can see on Figure 1 that our data from both classes have same orientation in space towards bottom right corner .

# RANDOM VECTORS AND THEIR PROPERTIES

## *In this section theoretical concepts of random vectors are given.*

For our binary classification problem solving it is necessary to introduce some mathematical concepts of random vectors.

In theory of statistical pattern recognition, measurements from one pattern or object are treated as random vectors. We will use term random variable in context of pattern recognition because It is clear that every time when we repeat the measurement, we will get some other value, which leads us to the idea of characterizing the measurement of physical quantity as some random variable that gives a new, different value in each realization. However, these values are not completely random, but subject to some laws. In order to formalize these laws, we will introduce two functions that will be joined to this random variable. First function is called *distribution function*, noted as $F_X(x)$ , and second is called *probability density function*, noted as $f_X(x)$.

Distribution function is defined as:

$$F_X(x) = P_r\{X \leq x\}$$

and it represents probability that random variable X will take value that is less or equal to argument x. It has three main characteristics:

$$F_X(\infty) = 1 \Leftrightarrow P_r\{X \leq \infty\} = 1$$

$$F_X(-\infty) = 0 \Leftrightarrow P_r\{X \leq -\infty\} = 0$$

$$x_1 \leq x_2 \Rightarrow F_X(x_1) \leq F_X(x_2)$$

Probability density function is defined as:

$$f_X(x) = \frac{\partial F(x)}{\partial x}$$

# PROBABILITY DENSITY FUNCTION AND DISTRIBUTION PARAMETERS

## *In this section*

Distribution function can be written as:

$$F_X(x) = \int_{-\infty}^{x} f_X(\tau) d\tau$$

Probability density function has two constraints:

$$F_X(\infty) = 1 \Rightarrow \int_{-\infty}^{\infty} f_X(x) = 1$$

$$(x_1 \leq x_2 \Rightarrow F_X(x_1) \leq F_X(x_2)) \Rightarrow (\forall x) f_X(x) \geq 0$$

Random vector X is completely described with distribution function or with probability density function. However, in many practical situations, these functions can't be determined or they are too complex. In those cases we must choose some other parameters that are less informative, but much more convenient in numerical sense. First and most important parameter is mathematical expectation or mean value of radnom vector X:

$$M_X = E\{X\} = \int_{\sigma_X} x f_X(x) dx$$

where integration goes trough whole space of random vector X. Conditional mathematical expectation of random vector X for class $w_i$ is integral

$$M_i = E\{X/w_i\} = \int x f(x/w_i) dx$$

Second very important parameter that characterize random vector X is covariance matrix:

$$\Sigma = E\left\{(X - M_X)(X - M_X)^T\right\}$$

# SAMPLE ESTIMATION TECHNIQUE

*In this section sample estimation technique is presented.*

$$\Sigma = E\left\{ \begin{bmatrix} X_1 - m_1 \\ \cdot \\ \cdot \\ \cdot \\ X_n - m_n \end{bmatrix} \begin{bmatrix} X_1 - m_1 & . & . & . & X_n - m_n \end{bmatrix} \right\} = \begin{bmatrix} c_{11} & . & . & . & c_{1n} \\ \cdot & & & & \\ \cdot & & & & \\ \cdot & & & & \\ c_{n1} & . & . & . & c_{nn} \end{bmatrix}$$

Component $c_{ij}$ of this matrix is:

$$c_{ij} = E\{(X_i - m_i)(X_j - m_j)\}; (i, j = 1, \ldots, n)$$

Thus, the diagonal elements of the covariance matrix form the variances of individual random variables in random vector, while non-diagonal elements represent covariances between random variables $X_i$ and $X_j$ . Although mathematical expectation and the covariance matrix are very important parameters which describe the distribution of a random vector, they are mostly unknown in practice and need to be estimated based on measured samples. This

procedure is called sample estimation technique. This technique is most commonly used to estimate the mean and variance of a random variable. Namely, if it is necessary to estimate the mean value of the random variable Y whose realizations $Y_i$ ; i=1,..N are known, the most simple way is to determine arithmetic mean:

$$\hat{m_Y} = \frac{1}{N}\sum_{1}^{N} Y_i$$

Similarly, estimation of variance can be written as:

$$\hat{\sigma_Y^2} = \frac{1}{N}\sum_{1}^{N} (Y_i - \hat{m_Y})^2$$

# HYPOTHESIS TESTING

## *Introduction to a priori and a posteriori probability of a class*

The main goal of pattern recognition is to decide to which category observed sample belongs. Based on observation or measurement a measurement vector is formed. This vector serves as an input to the decision rule through which this vector joins one of the analyzed classes. ***Hypothesis testing*** is a whole family of methods solving of this type of a problem. These methods are very powerful, but they assume knowing of joined probability density functions from all classes and this information is often unknown in practice.

In pattern recognition theory, we deal with random vectors gained from different classes and each of them is characterized with its distribution function and probability density function. These functions are called conditional functions, and according to that, conditional probability density function for i-th class is noted as:

$$f(x/w_i) or f_i(x); i = 1, 2, \ldots, L$$

where $w_i$ denotes class i, and L is overall number of classes. Unconditional probability density function of random vector X, which is often called mixed density function ia given as:

$$f(x) = \sum_{i=1}^{L} P_i f_i(x)$$

where $P_i$ denotes a priori probability of appareance of i-th class. A posteriori probability of class when random vector X is given is noted as $P(w_i/x)$ or $q_i(x)$ and can be calculated based on Bayesian theorem:

$$q_i(x) = \frac{P_i f_i(x)}{f(x)}$$

For our problem we will assume that measurement vector is random vector whose conditional probability density function depends from class that sample comes from. So far as these conditional probability density function are known, then pattern recognition problem becomes statistical hypothesis testing problem. We will observe case where we have two

classes $w_1$ and $w_2$ whose a priori probabilities $P_1$ and $P_2$ are known, as well as corresponding a posteriori probability density functions ($f_1(X) = f(X/w_1)$ and $f_2(X) = f(X/w_2)$).

# BAYESIAN MINIMAL ERROR DECISION RULE

*In this section Bayesian minimal error decision rule is given.*

Let assume that we have measurement vector X and our task is to determine to which of two classes this vector belongs. Simple decision rule can be based on conditional probabilities $q_1(X) = Pr(w_1/X)$ and $q_2(X) = Pr(w_2/X)$ as follows:

$$q_1(X) > q_2(X) \Rightarrow X \in w_1$$

$$q_1(X) < q_2(X) \Rightarrow X \in w_2$$

A posteriori probabilities $q_i(X)$ represent conditional probability that sample X comes from class $w_i$ if its numerical value is known, ie. realization. These probabilities can be calculated based on a priori probabilities of classes appearance $P_i$ and a posteriori probability density functions of measurement vectors $f_X = f_X(X/w_i)$ , using Bayesian theorem:

$$q_i(X) = \frac{f_i(X)P_i}{f(X)} = \frac{f_i(X)P_i}{f_1(X)P_1 + f_2(X)P_2}$$

Because mixed (a priori) probability density function is positive and joint for both a posteriori probabilities, decision rule can be written as follows:

$$P_1 f_1(X) > P_2 f_2(X) \Rightarrow X \in w_1$$

$$P_1 f_1(X) < P_2 f_2(X) \Rightarrow X \in w_2$$

or we can write above eqations as follows:

$$l(X) = \frac{f_1(X)}{f_2(X)} > \frac{P_2}{P_1} \Rightarrow X \in w_1$$

$$l(X) = \frac{f_1(X)}{f_2(X)} < \frac{P_2}{P_1} \Rightarrow X \in w_2$$

# LIKELIHOOD RATIO, TRESHOLD VALUE AND DISCRIMINATION FUNCTION

*Very important terms in pattern recognition theory are introduced- likelihood ratio, treshold value and discrimination function.*

Expression l(X) is called **likelihoodratio** , and that is very important quantity in pattern recognition theory. Ratio $P_2/P_1$ is called **tresholdvalue** in decision making. It is common practice to apply negative logarithm on likelihood ratio, and then decision rule has form:

$$h(X) = -ln(l(X)) = -ln(f_1(X)) + ln(f_2(X)) < ln(\frac{P_1}{P_2}) \Rightarrow X \in w_1$$

$$h(X) = -ln(l(X)) = -ln(f_1(X)) + ln(f_2(X)) > ln(\frac{P_1}{P_2}) \Rightarrow X \in w_2$$

A sign of inequality changed direction because of negative logarithm use. Expression h(X) is called **discrimination function** . Further on we will consider that $P_1 = P_2 = 0.5 \Rightarrow ln(P_1/P_2) = 0$ . Stated rules above are called **Bayesian rule** or **minimalerror decisiontest** . For analysis of stated rule, it is very important to determine probability of decision error. This classification rule can't ensure perefect classification (as well as other rules). When we say probability error, we consider probability of event that rule will bring wrong decision about measurement vector belonging to a class. Conditional probability of error for measurement vector, noted as r(X), is equal to smaller of probabilities $q_1(X)$ and $q_2(X)$ , ie.

$$r(X) = min[q_1(X), q_2(X)]$$

Total error which is called Bayesian error, noted as $\epsilon$ can be calculated as follows:

$$\epsilon = E\{r(X)\} = \int r(X)f(X)dX = \int min[q_1(X), q_2(X)]f(X)dX$$

$$= \int min[P_1 f_1(X), P_2 f_2(X)]dX = P_1 \int_{L_2} f_1(X)dX + P_2 \int_{L_1} f_2(X)dX$$

$$= P_1\epsilon_1 + P_2\epsilon_2$$

# TYPE I PROBABILITY ERROR AND TYPE II PROBABILITY ERROR

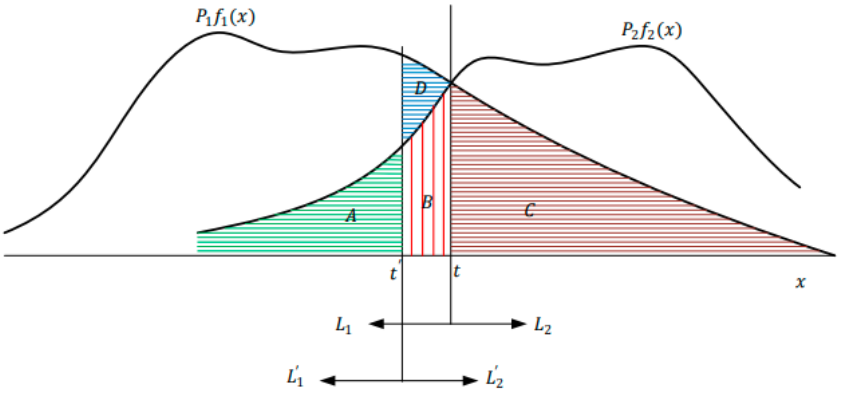
*In this section Type I probability error and Type II probability error are introduced.*

where

$$\epsilon_1 = \int_{L_2} f_1(X)dX; \epsilon_2 = \int_{L_1} f_2(X)dX$$

Probability $\epsilon_1$ is called Type I probability error and it represents probability that sample which comes from first class is wrongly classified. Similarly, probability $\epsilon_2$ is called Type II probability error and it represents probability that sample which comes from second class is wrongly classified. In a total error relation first equality represents definiton of this error, while other equality is applied Bayesian theorem. Integration area $L_1$ is the area from which the decision rule joins the measurement vector X to the class $w_1$ and analogously, integration area $L_2$ corresponds to those vectors X which the decision rule classifies into a class $w_2$ . Consequently, these areas are often called $w_1$ -area and $w_2$ -area, respectively. For measurement vectors from $L_1$ area holds the relation $P_1 f_1(X) > P_2 f_2(X)$ and according to that conditional probability error is $r(X) = P_2 f_2(X)/f(X)$ . Analogously, for vectors from $L_2$ area holds the relation $r(X) = P_1 f_1(X)/f(X)$ . Based on that, we can say that Bayesian

probability error consists of two terms. One of them refers to wrongly classified vectors from class $w_1$ , while other refers to wrongly classified vectors from the class $w_2$ .




Slika 2.2 Illustration of one dimensional random variables classification [7]

# NORMAL DISTRIBUTION OF RANDOM VECTOR

*In this section normal distribution of random vector is introduced.*

On the Figure 2 Bayesian decision rule for one dimensional measurement vectors is illustrated: Decision boundary is set to x=t, and that is the point where $P_1 f_1(X) = P_2 f_2(X)$ , and areas $x < t$ and $x > t$ are marked as $L_1$ and $L_2$ resepctively. In this way, probability errors become $P_1\epsilon_1 = B + C$ and $P_2\epsilon_2 = A$ . Total Bayesian error becomes $\epsilon = A + B + C$ , where A, B and C are marked areas, for example: $B = \int_t^{t'} P_1 f_1(X)dX$ . This decision rule generates minimal possible probability desicion error. Calculating of Bayesian probability error is very complex problem, because this probability is calculated with probability density function integration (which is function of more variables) trough very complex areas. Because of that it is much easier to consider the problem in discrimination function domain, and integration can be performed through this function. Random variable is scalar, so the integration problem can be performed in one dimensional space. We can write this as:

$$\epsilon_1 = \int_{-\infty}^{ln(P_1/P_2)} f_h(h/w_1)dh$$

$$\epsilon_2 = \int_{ln(P_1/P_2)}^{\infty} f_h(h/w_2)dh$$

where $f_h(h/w_i)$ is aposteriori probability density function of discrimination function h for samples that come from class $w_i$ .

For our problem for artificially generated data random vector X is normally distributed.

If random vector X is normally distributed, its probability density function can be written as:

$$N_X(M,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}d^2(X)} = \frac{1}{(2pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(X-M)^T\Sigma^{-1}(X-M)}$$

where $N_X(M,\Sigma)$ is shorted notation for normal distribution with mathematical expectation M and covariance matrix $\Sigma$ .Function $d^2(X)$ is called statistical distance (or $d^2$ curve) of vector X from mathematical expectation vector M.

# DISCRIMINATION FUNCTION

*In this section expression for discrimination function is given.*

For our problem of artificially generated data we have aposteriori probability density functions $f_i(X)$ , where i=1,2. These functions are normal, with mathematical expectation vector $M_i$ and covariance matrix $\Sigma_i$ . Bayesian minimal error decision rule can be written in form:

$$h(x) = -ln(l(X)) = -ln(f_1(X)) + ln(f_2(X)) < ln(\frac{P_1}{P_2}) \Rightarrow X \in w_1$$

Now we can replace $f_1(X)$ and $f_2(X)$ in upper equation and rewrite expression for h(x):

$$h(x) = -ln(\frac{1}{(2pi)^{n/2}|\Sigma_1|^{1/2}} e^{-\frac{1}{2}(X-M_1)^T\Sigma_1^{-1}(X-M_1)}) + ln(\frac{1}{(2pi)^{n/2}|\Sigma_2|^{1/2}} e^{-\frac{1}{2}(X-M_2)^T\Sigma_2^{-1}(X-M_2)})$$

We will use the logarithm propery that logarithm of quotient is difference of logarithm of numerator and logarithm of denumerator:

$$h(X) = -ln(\frac{e^{-\frac{1}{2}(X-M_1)^T\Sigma_1^{-1}(X-M_1)}}{(2\pi)^{n/2}|\Sigma_1|^{1/2}}) + ln(\frac{e^{-\frac{1}{2}(X-M_2)^T\Sigma_2^{-1}(X-M_2)}}{(2\pi)^{n/2}|\Sigma_2|^{1/2}})$$

$$h(X) = -ln(e^{-\frac{1}{2}(X-M_1)^T\Sigma_1^{-1}(X-M_1)}) + ln((2\pi)^{n/2}|\Sigma_1|^{1/2}) + ln(e^{-\frac{1}{2}(X-M_2)^T\Sigma_2^{-1}(X-M_2)}) - ln((2\pi)^{n/2}|\Sigma_2|^{1/2})$$

$$h(x) = -(-\frac{1}{2}(X-M_1)^T\Sigma_1^{-1}(X-M_1)) + ln((2\pi)^{n/2}) + ln(|\Sigma_1|^{1/2}) + (-\frac{1}{2}(X-M_2)^T\Sigma_2^{-1}(X-M_2)) - ln((2\pi)^{n/2}) - ln(|\Sigma_2|^{1/2})$$

$$h(x) = \frac{1}{2}(X-M_1)^T\Sigma_1^{-1}(X-M_1) + ln(|\Sigma_1|^{1/2}) - \frac{1}{2}(X-M_2)^T\Sigma_2^{-1}(X-M_2) - ln(|\Sigma_2|^{1/2})$$

Finally, we can write expression for discrimination function in final form:

$$h(x) = \frac{1}{2}(X-M_1)^T\Sigma_1^{-1}(X-M_1) - \frac{1}{2}(X-M_2)^T\Sigma_2^{-1}(X-M_2) + \frac{1}{2}ln(\frac{|\Sigma_1|}{|\Sigma_2|})$$

# DISCRIMINATION FUNCTION WITH SAME COVARIANCE MATRIX

*In this section expressions for discrimination function when covariance matrix are same is given.*

Last equation shows that decision boundary is quadratic function of X. If two classes have same covariance matrix, ie. if $\Sigma_1 = \Sigma_2 = \Sigma$, decision boundary becomes linear function of X and can be written in next form:

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2)$$

In our classification problem of artificially generated data classes have same covariance matrix and apriori probabilities of classes appearence are same. So, we can write expression for classification in the first class:

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) < ln(\frac{P_1}{P_2}) \Rightarrow X \in w_1$$

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) < ln(\frac{0.5}{0.5}) \Rightarrow X \in w_1$$

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) < ln(1) \Rightarrow X \in w_1$$

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) < 0 \Rightarrow X \in w_1$$

Similarly, we can write expression for classification in second class:

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) > 0 \Rightarrow X \in w_2$$

As we can see our classifier is line (we have linear function of X).

# DRAWING OF DECISION BOUNDARY

*In this section drawing of decision boundary is explained.*

I we want to draw our line, we can do that as follows. Decision of belonging to one of two classes is based on value of decision boundary (is it bigger or smaller then zero), so if we want to draw boundary itself, we must equal expression for h(x) with zero:

$$h(x) = (M_2 - M_1)^T \Sigma^{-1} X + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) = 0$$

For ease of writing, we can form matrix $M_3$ :

$$M_3 = M_2 - M_1 = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}$$

Also, inverse covariance matrix can be written as:

$$\Sigma^{-1} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

For our problem, vector X is two dimensional vector and can be written as:

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$$

Now, our expression for h(x) can be written in next form:

$$h(x) = \begin{bmatrix} m_1 & m_2 \end{bmatrix} \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2)$$

In order to draw our line, we must write expression for $X_2$ of $X_1$ , because we are in $X_1 X_2$ space. In other words, we have linear relationship and we want to express y of x, and in our case that is $X_2$ of $X_1$ . We must perform matrix multiply of first part of our expression and put $X_2$ on the left side of equal sign. We can do that as follows:

$$\begin{bmatrix} m_1 c_{11} + m_2 c_{21} & m_1 c_{12} + m_2 c_{22} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) = 0$$

$$(m_1 c_{11} + m_2 c_{21}) X_1 + (m_1 c_{12} + m_2 c_{22}) X_2 + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) = 0$$

$$(m_1 c_{11} + m_2 c_{21}) X_1 + \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2) = -(m_1 c_{12} + m_2 c_{22}) X_2$$

$$(m_1 c_{12} + m_2 c_{22}) X_2 = -(m_1 c_{11} + m_2 c_{21}) X_1 - \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2)$$

$$X_2 = \frac{-(m_1 c_{11} + m_2 c_{21}) X_1 - \frac{1}{2}(M_1^T \Sigma^{-1} M_1 - M_2^T \Sigma^{-1} M_2)}{(m_1 c_{12} + m_2 c_{22})}$$

# SHAPES OF BAYESIAN CLASSIFIERS

## *In this section shapes of Bayesian classifiers are explained.*

In our classification problem covariance matrix are same, so discrimination function is line.

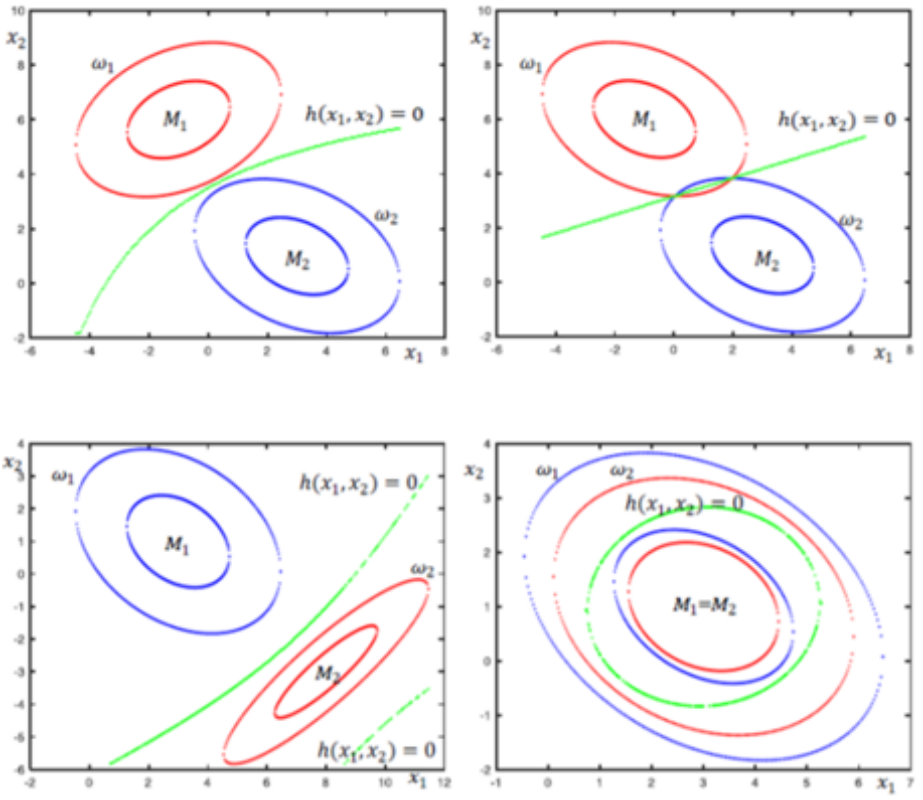
Shape of discrimination function will depend of classes distribution statistics. Depending of classes distribution statistics, discrimination function can be:

1. parable
2. line
3. hyperbola
4. ellipse

In literature we can often find for Bayesian classifier term Naive Bayesian classifier. That is because this classifier has assumption that chosen features for classification problem are

independent. Because of this assumption we can write decision rules as they are described previously.

This assumption is not really justified, because in practice selected features are almost always dependent. Although this assumption is naive, Bayesian classifier gives very good results in practice.




Slika 2.3 Characteristical shapes of Bayesian classifiers depending of classes distribution statistics [7]

# SOLUTION OF FIRST CLASSIFICATION PROBLEM IMPLEMENTED IN PYTHON

*In this section is given solution of first classification problem implemented in Python.*

```
#Generating of artificial data
import numpy as np
import matplotlib.pyplot as plt
M1 =np.array([-1, 4])
M2 =np.array([3, 1])
Sigma =np.array([[2, -1], [-1, 1]])
np.random.seed(0)
N=200
y1, y2 = np.random.multivariate_normal(M1, Sigma, N).T
z1, z2 = np.random.multivariate_normal(M2, Sigma, N).T
plt.plot(y1, y2, '*m',label='Class I')
plt.plot(z1, z2, 'oc',label='Class II')
```

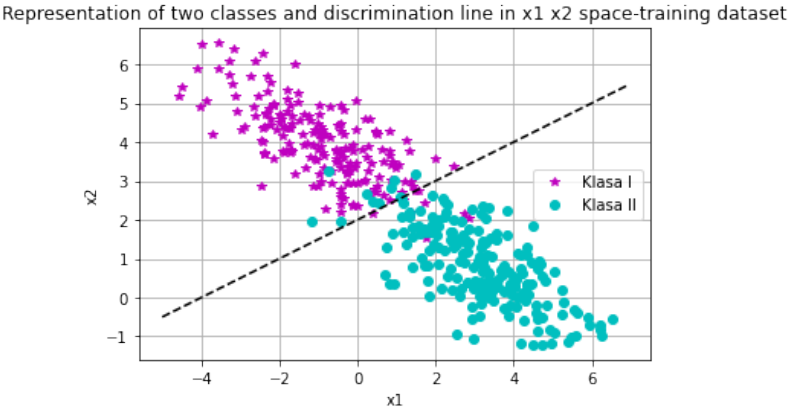
```
plt.grid(True)
plt.title('Two classes presented in x1 x2 space')
plt.xlabel('x1')
plt.ylabel('x2')
plt.legend()
plt.show()
y1.shape

#Discrimination line calculating
from scipy import linalg
invSigma=linalg.inv(Sigma)
M3=M2.T-M1.T
m1=M3[0]
m2=M3[1]
x1=np.arange(-5,7,0.1)
x2=[-(m1*invSigma[0,0]+m2*invSigma[1,0])*x1-0.5*(M1.T.dot(invSigma).dot(M1)-M2.T.dot
(invSigma).dot(M2))]/(m1*invSigma[0,1]+m2*invSigma[1,1])
plt.plot(y1, y2, '*m',label='Klasa I')
plt.plot(z1, z2, 'oc',label='Klasa II')
plt.plot(x1,x2.T,'k--')
plt.grid(True)
plt.title('Representation of two classes and discrimination line in x1 x2
space-training dataset')
plt.xlabel('x1')
plt.ylabel('x2')
plt.legend()
plt.show()


# New test data
M11 =np.array([-1.1, 4.1])
M21 =np.array([3.03, 1.02])
Sigma1 =np.array([[2.1, -1.1], [-1.1, 1.1]])
N1=50
y11, y21 = np.random.multivariate_normal(M11, Sigma1, N1).T
z11, z21 = np.random.multivariate_normal(M21, Sigma1, N1).T
plt.plot(y1, y2, '*m',label='Klasa I')
plt.plot(z1, z2, 'oc',label='Klasa II')
plt.plot(y11,y21,'*r')
plt.plot(z11,z21,'*g')
plt.plot(x1,x2.T,'k--')
plt.title('Representation of two classes and discrimination line in x1 x2
space-training and testing dataset')
plt.grid(True)
plt.xlabel('x1')
plt.ylabel('x2')
plt.legend()
plt.show()

plt.plot(y11,y21,'*r',label='Klasa I')
plt.plot(z11,z21,'*g',label='Klasa II')
plt.plot(x1,x2.T,'k--')
plt.grid(True)
```

```
plt.title('Prikaz test podataka u x1 x2 prostoru')
plt.title('Representation of two classes and discrimination line in x1 x2
space-testing dataset')
plt.xlabel('x1')
plt.ylabel('x2')
plt.legend()
plt.show()

#Calculating of error
#First class
x1p1=y11
x2p1=y21
h11=(m1*invSigma[0,0]+m2*invSigma[1,0])*x1p1+(m1*invSigma[0,1]+m2*invSigma[1,1])*x2p
1+0.5 *(M1.T.dot(invSigma).dot(M1)-M2.T.dot(invSigma).dot(M2))
error1=0
for i in range(0,N1):
      if h11[i]>0:
           error1=error1+1
#Second class
x1p2=z11
x2p2=z21
h21=(m1*invSigma[0,0]+m2*invSigma[1,0])*x1p2+(m1*invSigma[0,1]+m2*invSigma[1,1])*x2p
2+0.5 *(M1.T.dot(invSigma).dot(M1)-M2.T.dot(invSigma).dot(M2))
error2=0
for i in range(0,N1):
      if h21[i]<0:
           error2=error2+1

print(error1,error2)

#Classification error
Eps1=((error1)/N1)
Eps2=((error2)/N1)
P=0.5*Eps1+0.5*Eps2 #Total Bayesian error
T=(1-P)*100 #Accuracy
print(T)
```

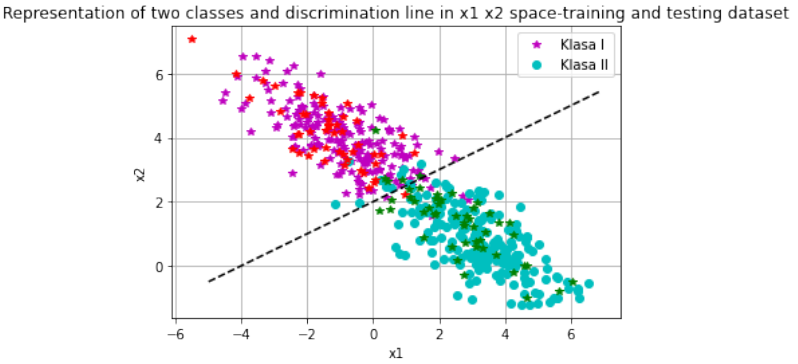
# BAYESIAN CLASSIFIER FOR FIRST CLASSIFICATION PROBLEM

*In this section training and testing dataset and discrimination line generated with Python code are shown.*

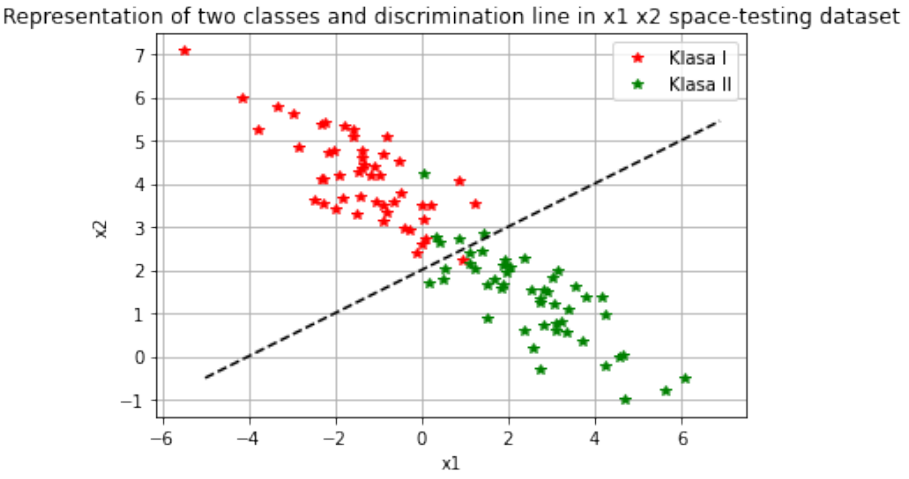
After running the code next figures are generated:




Slika 2.4 Training dataset and discrimination line. [Source: Author]




Slika 2.5 Training and testing dataset and discrimination line. [Source: Author]




Slika 2.6 Testing dataset and discrimination line. [Source: Author]

After calculating of Type I error and Type II error on testing dataset we get 1 sample that is wrongly classified from class I, and 5 samples that are wrongly classified from class II. Classifier accuracy is 94%.

One important note: Discrimination line is learned on training dataset (Figure 4) and same discrimination line is used for testing dataset (Figure 6). Our discrimination line is our

Bayesian classifier that we learned and after learning phase we used it on new data in testing phase.

# Poglavlje 3

# Second classification problem

## CLASSIFICATION OF IRIS DATASET

*In this section second classification problem is given.*

Second classification problem: Classification of Iris dataset.

Task: Choose two features for representation of data and discrimination lines in two dimensional space. Perform classification.

Description of Iris dataset: Iris dataset is a set with Iris plant species. Namely, the plant Iris can be classified into three types based on the width and length of the petals and the width and length of the sepals: Iris Versicolor, Iris Setosa and Iris Virginica. This dataset was formed back in 1936 by biologist and statistician Fisher, but it is still used today as a good initial example of machine learning. The species of the Iris plant are shown in the following figure:

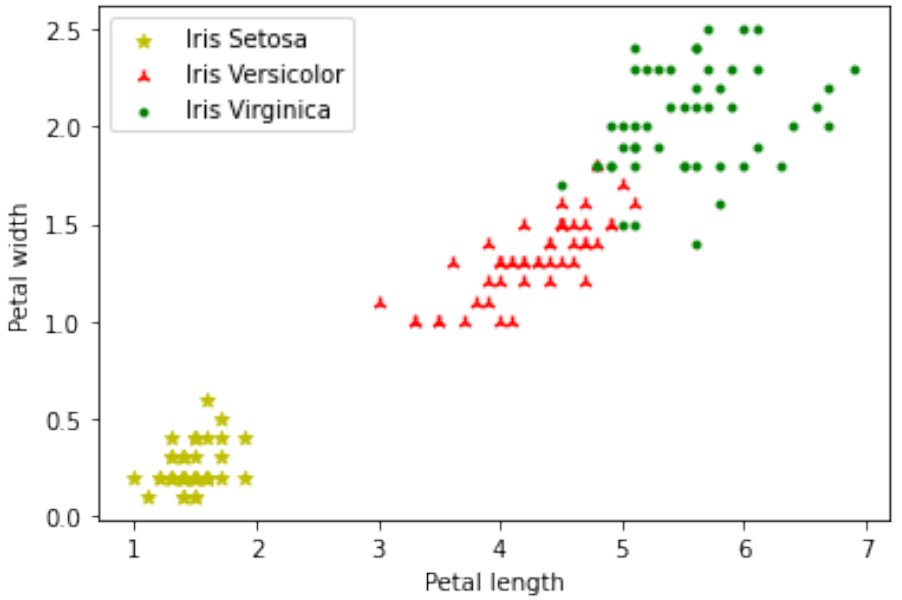

Slika 3.1 Species of the Iris plant [8]

We can see from Figure 1 that each species has a different width and length of sepals and petals, so based on these characteristics, species of the Iris plant can be determined. In fact, the lengths and widths of petals and sepals will be our features, because they describe each type of Iris plant well and distinguish between classes, i.e. Iris plant species. The iris dataset consists of five columns. The first four columns are our features, i.e. descriptors of our classes. The first column represents the length of the sepal, the second column represents the width of the sepal, the third column represents the length of the petal, while the fourth column represents the width of the petal. The fifth column represents our classes, i.e. Iris plant species. So, for a certain length and width of the petals and sepals, species of a plant Iris is known exactly, and it is marked in the fifth column. Therefore, the fifth column represents the target values for our dataset.

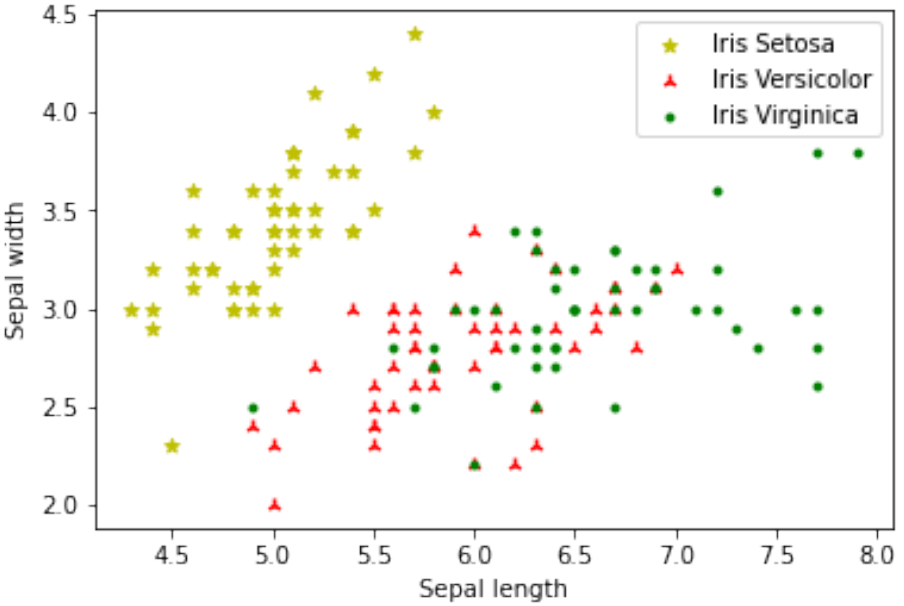
# REPRESENTATION OF IRIS DATASET WITH TWO FEATURES

*In this section representation of Iris dataset with two features is given.*

We will represent our dataset in two dimensions for ease of visualization. For that purpose, we will choose two features out of four. On the figures below are shown visualizations with two features, so we can decide which features are better for our classification problem.




Slika 3.2 Representation of Iris dataset with petal length and petal width. [Source: Author]




Slika 3.3 Representation of Iris dataset with sepal length and sepal width. [Source: Author]

We can see that with choice of features on Figure 2 we have separable classes and we can expect good classification results. With choice of features as on Figure 3, we have overlapping between classes Iris Versicolor and Iris Virginica, so these two features do not make good distinguish between classes and we will not use them in our classification problem.

# DISCRIMINATION FUNCTION WITH DIFFERENT COVARIANCE MATRIX

*In this section discrimination function with different covariance matrix is given.*

In our second classification problem we have classes with different mean vectors and different covariance matrix. Here, we will have also have binary classification problem, but now we must fine two discrimination functions- first one between classes Iris Setosa and Iris Versicolor and second between classes Iris versicolor and Iris Virginica.
When covariance matrix are not same, then discrimination function is not line, it is parable.

When we have two classes with different covariance matrix and different mean vector, it can be written as:

$$\Sigma_1^{-1} = \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix}$$

$$\Sigma_2^{-1} = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}$$

$$M_1 = \begin{bmatrix} m_{11} \\ m_{12} \end{bmatrix}$$

$$M_2 = \begin{bmatrix} m_{21} \\ m_{22} \end{bmatrix}$$

Our discrimination function can be written as follows:

$$h(x) = \frac{1}{2}(X - M_1)^T \Sigma_1^{-1}(X - M_1) - \frac{1}{2}(X - M_2)^T \Sigma_2^{-1}(X - M_2) + \frac{1}{2} ln(\frac{|\Sigma_1|}{|\Sigma_2|})$$

$$h(x) = \frac{1}{2}(\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} - \begin{bmatrix} m_{11} \\ m_{12} \end{bmatrix})^T \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix}^{-1} (\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} - \begin{bmatrix} m_{11} \\ m_{12} \end{bmatrix})$$
$$- \frac{1}{2}(\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} - \begin{bmatrix} m_{21} \\ m_{22} \end{bmatrix})^T \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}^{-1} (\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} - \begin{bmatrix} m_{21} \\ m_{22} \end{bmatrix}) + \frac{1}{2} ln(\frac{|\Sigma_1|}{|\Sigma_2|})$$

# BAYESIAN CLASSIFIER

*In this section expression for Bayesian classifier for second classification problem is given.*

$$h(x) = \frac{1}{2}[\, X_1 - m_{11} \qquad X_2 - m_{12} \,] \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix}^{-1} \begin{bmatrix} X_1 - m_{11} \\ X_2 - m_{12} \end{bmatrix}$$
$$- \frac{1}{2}[\, X_1 - m_{21} \qquad X_2 - m_{22} \,] \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}^{-1} \begin{bmatrix} X_1 - m_{21} \\ X_2 - m_{22} \end{bmatrix} + \frac{1}{2} ln(\frac{|\Sigma_1|}{|\Sigma_2|})$$

$$h(x)$$
$$= \frac{1}{2}[\, (X_1 - m_{11})z_{11} + (X_2 - m_{12})z_{21} \qquad (X_1 - m_{11})z_{12} + (X_2 - m_{12})z_{22} \,] \begin{bmatrix} X_1 - m_{11} \\ X_2 - m_{12} \end{bmatrix}$$
$$- \frac{1}{2}[\, (X_1 - m_{21})k_{11} + (X_2 - m_{22})k_{21} \qquad (X_1 - m_{21})k_{12} + (X_2 - m_{22})k_{22} \,] \begin{bmatrix} X_1 - m_{21} \\ X_2 - m_{22} \end{bmatrix}$$
$$+ \frac{1}{2} ln(\frac{|\Sigma_1|}{|\Sigma_2|})$$

We can now write our discrimination function in final form when we have different covariance matrix:

$$h(x) = \frac{1}{2}[((X_1 - m_{11})z_{11} + (X_2 - m_{12})z_{21})(X_1 - m_{11})$$
$$+ ((X_1 - m_{11})z_{12} + (X_2 - m_{12})z_{22})(X_2 - m_{12})]$$
$$- \frac{1}{2}[((X_1 - m_{21})k_{11} + (X_2 - m_{22})k_{21})(X_1 - m_{21})$$
$$+ ((X_1 - m_{21})k_{12} + (X_2 - m_{22})k_{22})(X_2 - m_{22})] + \frac{1}{2} ln(\frac{|\Sigma_1|}{|\Sigma_2|})$$

Equation above is parable. Again, we can see that shape of Bayesian classifier will depend of classes distribution statistic.

Now we can implement solution for our problem in Python.

# SOLUTION OF SECOND CLASSIFICATION PROBLEM IMPLEMENTED IN PYTHON

*In this section solution of second classification problem implemented in Python is given.*

```
import sklearn
from sklearn import datasets
dataset=datasets.load_iris()
X=dataset.data
Y=dataset.target

import matplotlib.pyplot as plt

plt.scatter(X[0:50,2],  X[0:50,3],c=u'y',marker='*',    label='Iris Setosa')
plt.scatter(X[50:100,2],    X[50:100,3],    c=u'r',marker='2',  label='Iris
Versicolor')
plt.scatter(X[100:150,2],   X[100:150,3],   c=u'g',marker='.',label =   'Iris
```

```
Virginica')
plt.xlabel('Petal length')
plt.ylabel('Petal width')
plt.legend()
plt.show()

plt.scatter(X[0:50,0],  X[0:50,1],c=u'y',marker='*',    label='Iris Setosa')
plt.scatter(X[50:100,0],    X[50:100,1],    c=u'r',marker='2',  label='Iris
Versicolor')
plt.scatter(X[100:150,0],   X[100:150,1],   c=u'g',marker='.',label =   'Iris
Virginica')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.legend()
plt.show()

import numpy as np
X1 = X[0:50, 2:4]
X2 = X[50:100, 2:4]
X3 = X[100:150, 2:4]
sigma1 = np.cov(X1.T)
sigma2 = np.cov(X2.T)
sigma3 = np.cov(X3.T)

Xaxis=np.arange(0,8,0.1)
Yaxis=np.arange(0,3,0.1)
x1,x2=np.meshgrid(Xaxis,Yaxis)
from scipy import linalg
invSigma1 = linalg.inv(sigma1)
invSigma2 = linalg.inv(sigma2)
invSigma3 = linalg.inv(sigma3)
M1 =np.mean(X1,axis=0)
M2 =np.mean(X2,axis=0)
M3 =np.mean(X3,axis=0)


y=0.5*(((x1-M1[0])*invSigma1[0][0]+(x2-M1[1])*invSigma1[1][0])*(x1-M1[0])+((x1-
M1[0])*invSigma1[0][1]+(x2-M1[1])*invSigma1[1][1])*(x2-M1[1]))-0.5*(((x1-
M2[0])*invSigma2[0][0]+(x2-M2[1])*invSigma2[1][0])*(x1-M2[0])+((x1-M2[0])*invSigma2[
0]
[1]+(x2-M2[1])*invSigma2[1][1])*(x2-M2[1]))
+0.5*np.log(np.linalg.det(sigma1)/np.linalg.det(sigma2))




b=0.5*(((x1-M2[0])*invSigma2[0][0]+(x2-M2[1])*invSigma2[1][0])*(x1-M2[0])+((x1-
M2[0])*invSigma2[0][1]+(x2-M2[1])*invSigma2[1][1])*(x2-M2[1]))-0.5*(((x1-
M3[0])*invSigma3[0][0]+(x2-M3[1])*invSigma3[1][0])*(x1-M3[0])+((x1-M3[0])*invSigma3[
0]
[1]+(x2-M3[1])*invSigma3[1][1])*(x2-M3[1]))
+0.5*np.log(np.linalg.det(sigma2)/np.linalg.det(sigma3))
```

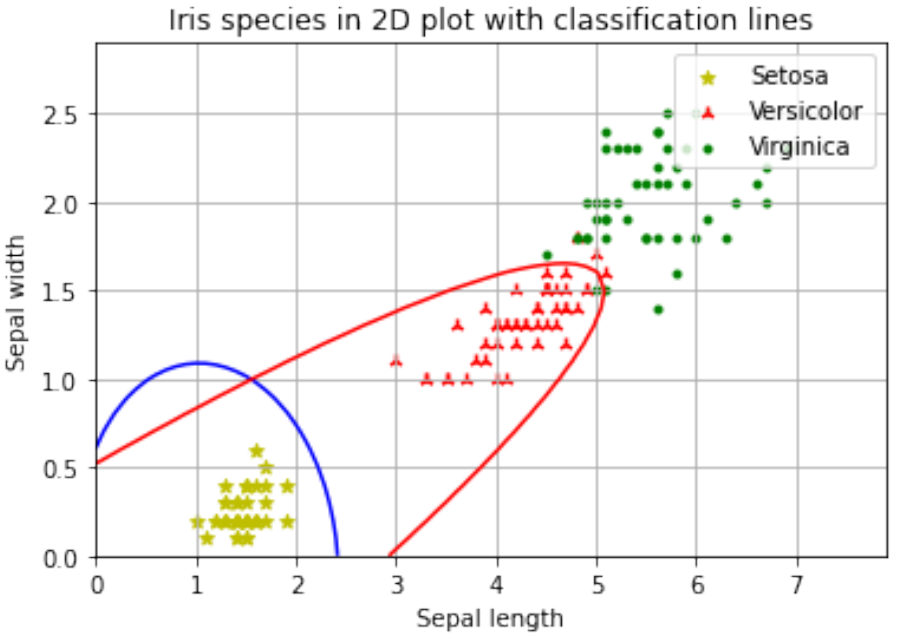
```
plt.scatter(X[0:50,2], X[0:50,3],c=u'y',marker='*', label='Setosa')
plt.scatter(X[50:100,2], X[50:100,3], c=u'r',marker='2', label='Versicolor') #
selekcija
plt.scatter(X[100:150,2], X[100:150,3], c=u'g',marker='.',label = 'Virginica') #
plt.xlabel("Sepal length")
plt.ylabel("Sepal width")
plt.legend(loc="upper right")
plt.contour(x1,x2,y,0,colors='blue')
plt.contour(x1,x2,b,0,colors='red')
plt.grid(True)
plt.title('Iris species in 2D plot with classification lines')
plt.show()




x1p1 = X[0:50, 2]
x2p1 = X[0:50, 3]
h1=0.5*(((x1p1-M1[0])*invSigma1[0][0]+(x2p1-M1[1])*invSigma1[1][0])*(x1p1-M1[0])+((x
1p1-
M1[0])*invSigma1[0][1]+(x2p1-M1[1])*invSigma1[1][1])*(x2p1-M1[1]))-0.5*(((x1p1-
M2[0])*invSigma2[0][0]+(x2p1-M2[1])*invSigma2[1][0])*(x1p1-M2[0])+((x1p1-
M2[0])*invSigma2[0][1]+(x2p1-M2[1])*invSigma2[1][1])*(x2p1-M2[1]))
+0.5*np.log(np.linalg.det(sigma1)/np.linalg.det(sigma2))
error1=0
for i in h1:
 if i>0:
  error1=error1+1
x1p2 = X[50:100, 2]
x2p2 = X[50:100, 3]
h2=0.5*(((x1p2-M1[0])*invSigma1[0][0]+(x2p2-M1[1])*invSigma1[1][0])*(x1p2-M1[0])+((x
1p2-
M1[0])*invSigma1[0][1]+(x2p2-M1[1])*invSigma1[1][1])*(x2p2-M1[1]))-0.5*(((x1p2-
M2[0])*invSigma2[0][0]+(x2p2-M2[1])*invSigma2[1][0])*(x1p2-M2[0])+((x1p2-
M2[0])*invSigma2[0][1]+(x2p2-M2[1])*invSigma2[1][1])*(x2p2-M2[1]))
+0.5*np.log(np.linalg.det(sigma1)/np.linalg.det(sigma2))
error2=0
for i in h2:
 if i<0:
   error2=error2+1
Eps1=((error1)/50.0)

Eps2=((error2)/50.0)
P=0.5*Eps1+0.5*Eps2
T=(1-P)*100
print('Classification of Setosa and Versicolor - Classification accuracy: '
+str(T)+'%')

print('Number of bad classified samples from class Setosa: '+ str(error1))
```

```
print('Number of bad classified samples from class Versicolor: '+ str(error2))

x1p1 = X[50:100, 2]
x2p1 = X[50:100, 3]
h1=0.5*(((x1p1-M2[0])*invSigma2[0][0]+(x2p1-M2[1])*invSigma2[1][0])*(x1p1-M2[0])+((x
1p1-
M2[0])*invSigma2[0][1]+(x2p1-M2[1])*invSigma2[1][1])*(x2p1-M2[1]))-0.5*(((x1p1-
M3[0])*invSigma3[0][0]+(x2p1-M3[1])*invSigma3[1][0])*(x1p1-M3[0])+((x1p1-
M3[0])*invSigma3[0][1]+(x2p1-M3[1])*invSigma3[1][1])*(x2p1-M3[1]))
+0.5*np.log(np.linalg.det(sigma2)/np.linalg.det(sigma3))
error1=0
for i in h1:
 if i>0:
    error1=error1+1
x1p2 = X[100:150, 2]
x2p2 = X[100:150, 3]
h2=0.5*(((x1p2-M2[0])*invSigma2[0][0]+(x2p2-M2[1])*invSigma2[1][0])*(x1p2-M2[0])+((x
1p2-
M2[0])*invSigma2[0][1]+(x2p2-M2[1])*invSigma2[1][1])*(x2p2-M2[1]))-0.5*(((x1p2-
M3[0])*invSigma3[0][0]+(x2p2-M3[1])*invSigma3[1][0])*(x1p2-M3[0])+((x1p2-
M3[0])*invSigma3[0][1]+(x2p2-M3[1])*invSigma3[1][1])*(x2p2-M3[1]))
+0.5*np.log(np.linalg.det(sigma2)/np.linalg.det(sigma3))
error2=0
for i in h2:
 if i<0:
   error2=error2+1
Eps1=((error1)/50.0)
Eps2=((error2)/50.0)
P=0.5*Eps1+0.5*Eps2
T=(1-P)*100
print('Classification of Versicolor and Virginica - Classification accuracy: '
+str(T)+'%')

print('Number of bad classified samples from class Versicolor: '+ str(error1))

print('Number of bad classified samples from class Virginica: '+ str(error2))
```

# BAYESIAN CLASSIFIER FOR IRIS DATASET

*In this section Bayesian classifier for Iris dataset is presented.*

After running the code classification of Iris dataset is performed:




Slika 3.4 Classification of Iris dataset [Source:Author]

```
Classification of Setosa and Versicolor - Classification accuracy: 100.0%
Number of bad classified samples from class Setosa: 0
Number of bad classified samples from class Versicolor: 0
Classification of Versicolor and Virginica - Classification accuracy: 96.0%
Number of bad classified samples from class Versicolor: 3
Number of bad classified samples from class Virginica: 1
```

Slika 3.5 Accuracy of classifier [Source:Author]

We can see on Figure 4 that Bayesian classifier for our second classification problem is parable. Also, we can notice that classes Iris Setosa and Iris Versicolor are perfectly classified with 100% accuracy, while classification accuracy is 96% for classes Iris Versicolor and Iris Virginica. These results are expected, because classes Iris Setosa and Iris Versicolor are separable, while classes Iris Versicolor and Iris Virginica are not separable, so we must expect some classification error. Bayesian classifier guarantees that this classification error will be minimal.

We saw in this classification problem how important is good choice of features. If features doesn't describe our data we can't expect good classification results.

Of course, for this problem we could use all features that we've got in our dataset, but then we couldn't visualize our data in two dimensional space, so that is the reason for choosing of two features.

**Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.**

# Poglavlje 4

# Homework

## PRACTICAL EXAMPLES FOR HOMEWORK

*In this section practical tasks for solving are given.*

***Assignment 1.***

Two classes with two-dimensional shapes are given with normal probability density functions:

$$f_1(x) = N(M_1, \Sigma_1), M_1 = \begin{bmatrix} -2 \\ 1 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

$$f_2(x) = N(M_2, \Sigma_2), M_2 = \begin{bmatrix} 2 \\ 4 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

Task: Assuming that prior probabilities of classes are same, generate 600 samples for each class and project a minimal error Bayesian classifier. Visualize data and discrimination function h(x)=0. Calculate Type I error and Type II error and accuracy of classifier.

***Assignment 2.***

Dataset Banknotes.txt is given. The set represents banknotes that are classified as real or fake banknotes based on their characteristics. The set was obtained by photographing the banknotes, and then 4 attributes that reliably describe the set were extracted from the images. The first four columns of the set are features and the fifth column contains the target values. Task: Project a minimal error Bayesian classifier. Choose two features for data visualization and discrimination function representation in 2D space. Calculate the accuracy of classifier.

Solve the assignments in programming language Python.

# Poglavlje 5

# Zaključak

## ZAKLJUČAK

*In this section conclusion in given.*

- In this lecture basic concepts of classification are explained.
- Two different classification problems vere posted and solved theoretically in detail. Solutions of problems were implemented in Python programming language.
- It is shown that shape of Bayesian classifier depends of classes distribution statistics.
- In our first problem, we performed classification of data from two classes with same covariance matrix and our Bayesian classifier was line.
- In our second problem, we performed classification of data from two classes with different covariance matrix and our Bayesian classifier was parable.
- It is shown that choice of features is crucial for good classification results.
- It is shown that when we have overlapping between classes, we must have classification error, but Bayesian classifier guarantee that this error will be minimal.

After this lecture, students will understand basic concepts of classification. They will be able to solve classification problems with Bayesian classifier
and implement their knowledge on real world datasets.

## REFERENCES

*In this sections references used for this lecture are presented.*


1. https://www.altexsoft.com/whitepapers/machine-learning-bridging-between-business-and-data-science/

2. https://www.infoworld.com/article/3394399/machine-learning-algorithms-explained.html

3. https://www.infoworld.com/article/3394399/machine-learning-algorithms-explained.html

4. https://scikit-learn.org/0.15/auto_examples/plot_classifier_comparison.html

5. https://automatika.etf.bg.ac.rs/images/FAJLOVI_srpski/predmeti/izborni_kursevi_os/obrada_signala/OS4PO/materijali/Predavanje%201.pdf

6. https://automatika.etf.bg.ac.rs/images/FAJLOVI_srpski/predmeti/izborni_kursevi_os/obrada_signala/OS4PO/materijali/Predavanje%202%202.pdf

8. http://www.lac.inpe.br/~rafael.santos/Docs/CAP394/WholeStory-Iris.html