Universidad de Cantabria

Facultad
de
Ciencias

Aprendizaje Automático Supervisado para la Predicción de Calidad en Procesos de Moldeo Industrial

(Supervised Machine Learning for Quality Prediction in Industrial Moulding Processes)

Trabajo de Fin de Grado
para acceder al

GRADO EN MATEMÁTICAS

Autor: Andrea Sainz Bear

Director: Juan Antonio Cuesta Albertos

Febrero - 2024

Resumen

Este trabajo se centra en el mantenimiento predictivo. Más concretamente, se pretende predecir si las piezas producidas por determinada maquinaria van a ser satisfactorias o defectuosas. Para ello se dispone de una muestra de piezas fabricadas por dicha maquinaria. Por lo tanto, estamos ante un problema de aprendizaje supervisado para cuya solución se ha optado por el uso de las técnicas de k Vecinos Más Próximos y el "Support Vector Machine". En el trabajo se estudian las propiedades matemáticas de ambas técnicas prestando especial atención a su consistencia universal. También se incluye un análisis del comportamiento práctico de las técnicas seleccionadas en el problema que nos ocupa.

Palabras clave: Aprendizaje supervisado, k Vecinos Más Cercanos,"Support Vector Machine", Consistencia universal.

Abstract

This work focuses on predictive maintenance. More specifically, the aim is to predict whether the pieces produced by certain machines are going to be satisfactory or defective. For this purpose, a sample of pieces produced by this machine is available. Therefore, we are dealing with a supervised learning problem for whose solution we have decided to use the techniques of k-Nearest Neighbours and the Support Vector Machine. The paper studies the mathematical properties of both techniques, paying special attention to their universal consistency. An analysis of the practical behaviour of the selected techniques on the problem at hand is also included.

Keywords: Supervised learning, k Nearest Neighbours, Support Vector Machine, Universal consistency.

Acknowledgement

After five hard years of university, I can look back and feel proud of all that I have achieved and fought for when the time came to hand in this final project.

I cannot say that I have achieved this on my own because I would be lying. I have many people to thank for having accompanied and helped me along the way, but I will try to be brief.

First of all, I must thank my classmates in the double degree who from the first day we were together helping and learning from each other.
Secondly, to each and every one of the teachers who have passed through my university life. Thank you for having taught me everything you knew about your specialities and for having always been willing to help me and to resolve any doubts that arose.

I would like to make a special mention to Juan Antonio. Thank you for having taught me so much in this short period of time, for having been there offering me your help so quickly, and above all, thank you for the enthusiasm and excitement you put into teaching, you infect all your students with it and make the classes a thousand times more entertaining.

Thank you to my friends for being a fundamental pillar in my life, for being so generous, for being so supportive, for rejoicing in my achievements even more than I do, and of course, for being there for me all my life.

Thanks to my family, especially my parents, my brother and my partner, for understanding me when I was in a bad mood because of my exam nerves, for always supporting and helping me and for celebrating with me every little achievement. You are a fundamental part of me.

Thanks also to Sonia for taking care of me like her daughter and for always being there in good and bad times.

Finally, thanks to my colleagues for their help and support during this time.

Contents

1 Introduction **6**
1.1 Motivation . . . 6
1.2 Objectives . . . 7
1.3 Organisation and structure of the document . . . 8

2 Introduction to supervised classification **9**
2.1 Basic ideas . . . 9
2.2 Classifier consistency . . . 11
2.3 The Bayes classifier . . . 12

3 Theoretical basis of mathematical models **15**
3.1 k-Nearest Neighbors . . . 15
3.2 Support Vector Machine . . . 25

4 Results and analysis **31**
4.1 Statistical Parameters . . . 31
4.2 Cross Validation . . . 33
4.3 Data analysis . . . 34
4.3.1 k-Nearest Neighbors . . . 36
4.3.2 Support Vector Machine . . . 38
4.4 The results . . . 42

5 Conclusions **45**

References **47**

Appendixes **49**
Appendix 1 . . . 49
Appendix 2 . . . 51

List of Figures

1 Confusion Matrix [4] 32
2 K-fold Cross Validation process [2]. 34
3 Variables available in the data set ordered by their relevance 36
4 Statistical parameters for the kNN algorithm 38
5 Statistical parameters for SVM with linear kernel 39
6 Statistical parameters for SVM with polynomial kernel 40
7 Statistical parameters for SVM with radial kernel 40
8 Statistical parameters for SVM with sigmoid kernel 41
9 Confusion matrix for the kNN algorithm with 17 neighbours 42
10 Confusion matrix for the SVM algorithm with polynomial kernel and a regularisation parameter c =1 43

List of Tables

1 Comparison of statistical parameters between the different kernels of the SVM algorithm 41
2 Comparison of statistical parameters between the two algorithms 44

1 Introduction

1.1 Motivation

Industrialisation is a significant period in history that has had a profound impact on production and the economy. It is characterised by the transition from handmade methods to mechanical and automated processes in the production of goods. During the 18^{th} century Industrial Revolution, the introduction of machinery, automation of processes, and energy usage drove a significant increase in efficiency and production, transforming economic activities.

The impact of industrialisation has been deep, with advanced technologies boosting production capacity, lowering costs, and improving product quality. Despite the benefits, industrialisation has also brought challenges such as labour automation, changes in the workforce, and environmental concerns. The relationship between industrialisation and industry is dynamic, marked by constant technological advances that require continuous adaptation to maintain competitiveness and address emerging issues.

The field of statistical learning, better known as Machine Learning, is a branch of artificial intelligence that focuses on developing algorithms and models that allow machines to learn patterns and perform tasks without being explicitly programmed. Instead of following specific instructions, Machine Learning models can improve their performance as they are given more data and feedback. The central idea is that machines can learn from experience, identify patterns and make intelligent decisions without direct human intervention.

The concept of Machine Learning has its roots in the 19^{th} century, and pioneers such as Alan Turing and Marvin Minsky contributed significantly to its development. Arthur Samuel first coined the term "Machine Learning" and defined it as the field of study that gives computers the ability to learn without being explicitly programmed. Since then, Machine Learning has experienced rapid progress, especially with the increase in computational power and the availability of large data sets.

There are different approaches in Machine Learning, mainly classified into four types: supervised, unsupervised, reinforced and semi-supervised learning. In supervised learning, labelled data are used to train the model, while in unsupervised learning, unlabelled data are used to discover patterns. Reinforcement learning involves learning through trial and error, with the model receiving feedback in the form of reinforcement signals. Finally, semi-supervised learning is a combination of the supervised and unsupervised approaches, leveraging both labelled and unlabelled datasets to train models more efficiently.

Machine learning has had a significant impact on industry, enabling process optimization, product customization, and real-time data analysis. It is important to consider these challenges alongside the benefits of this technological breakthrough. While it has opened new doors for innovation and reduced costs, there are also ethical and security challenges that require careful attention. The convergence of industrialisation and machine learning defines this new era of production, where human dexterity and computational potential come together in a unique way.

Predictive maintenance is a direct application of statistical learning in industry. It involves studying variables measured by sensors, such as temperature, pressure, and vibration, to determine when a machine is likely to fail. This work aims to predict whether the pieces produced by a plastic moulding machine will be defective or not based on the selected values by the operator. If the pieces are predicted to be defective, the operator can adjust the parameters to find a combination that yields a satisfactory result.

1.2 Objectives

The objective of this work is the in-depth study of the mathematical bases and the universal consistency of two supervised algorithms that are the k-nearest neighbors (kNN) and the support vector machine (SVM). Its actual application in a plastic molding machine is done to improve the process, the pieces quality and savings in both raw material and cost.

More specifically, the main objectives of this work are:

- To introduce the kNN including the analysis of its main characteristics and the obtention of its universal consistency.
- To introduce the SVM including the analysis of its main characteristics and the obtention of its universal consistency.
- The application of both procedures to a real data set and the analysis of the obtained results.
- Propose some possibles improvements on both procedures.

1.3 Organisation and structure of the document

This report consists of an introduction to the supervised classification problem along three chapters that will examine the kNN and support vector machine algorithms both theoretically and practically.

The distribution will be as follows:

- **Chapter 1**: a mathematical introduction to the problem and to the supervised classification will be described.
- **Chapter 2**: the mathematical development and universal consistency of both models will be analysed.
- **Chapter 3**: the study of the results obtained with kNN and SVM when applied to a real dataset from a molding machine.
- **Chapter 4**: the final conclusions of the project will be discussed and suggestions for improvement will be made.

2 Introduction to supervised classification

This chapter introduces basic and essential concepts and some results on the study of universal consistency.

2.1 Basic ideas

The material in this subsection is taken from [3], pages 1 to 4.

In that follows, given $M \in \mathbb{N}$, we will assume that we have a sample $(X_1, Y_1), ..., (X_n, Y_n)$, where $X_i \in \mathbb{R}^d$ and $Y_i \in \{1, ..., M\}$, for $i = 1, ..., n$, and an *observation* (X, Y) from which we know X and want to guess Y, what we call a *class*.
A *classifier* is any map $g(X) : \mathbb{R}^d \to \{1, ..., M\}$; where the value $g(x)$ represents the guess of Y given $X = x$.
The most common scenario is when $M = 2$. For instance, this occurs when determining if a person is healthy or infected, or if a piece is defective or correct.

Usually, the observation does not fully describe the underlying process (Y is not a deterministic function of X), then it is possible for the same X to produce two different Y at different times. Thus all classifiers are imperfect, so we cannot discard a classifier because it misclassifies a particular X. Therefore we introduce a probabilistic setting and let (X, Y) be a $\mathbb{R}^d \times \{1, ..., M\}$-valued random pair. The distribution of (X, Y) describes the frequency with which certain pairs occur in practice. It is implied that the distribution of X given $Y = i$ is different from that obtained if $Y = j$ for some $j \neq i$.

The classifier errs if $g(X) \neq Y$, where the probability of error for a classifier g is

$$L(g) = P\{g(X) \neq Y\}.$$

There exits a best possible classifier, g^*, which is defined by

$$g^* = \arg\min P\{g(X) \neq Y\},$$

where g varies on the set of measurable maps from $\mathbb{R}^d$ to $\{1, ..., M\}$. Obviously, g^* depends on the distribution of (X, Y). The problem of finding g^* is the *Bayes' problem*, and the classifier g^* is called *Bayes rule*. The minimal probability of error is called the *Bayes error* and is denoted by $L^*(L^* = L(g^*))$. In most cases, the distribution of (X, Y) is unknown, so g^* is also unknown.

Unless there is a guarantee that the set of $\{(X_i, Y_i),\ i = 1, ..., n\}$ is representative of the distribution (which is mostly unknown), finding a classifier g with a small probability of error is impossible.

From this point on, it will be assumed that the set of observations (X_1, Y_1), ..., (X_n, Y_n) is a sequence of independent identically distributed (i.i.d.) random pairs with the same distribution as (X, Y).

We denote by g_n to a classifier constructed from the observations $(X_1, Y_1), ..., (X_n, Y_n)$ and therefore, Y is guessed by $g_n(X)$. We call a sequence of classifiers (functions) a *classification rule*, i.e., $\{g_n, n \geq 1\}$ is a classification rule.

The process of constructing g_n is called *supervised learning*. The performance of g_n is measured by the conditional probability of error

$$L_n = L(g_n) = P\{g_n(X) \neq Y | (X_1, Y_1), ..., (X_n, Y_n)\}$$

This is a random variable because it depends on the data. Thus, L_n averages over the distribution of (X, Y), when the data remain fixed.

On the other hand, the number $E\{L_n\}$ is a more useful indicator since it measures the quality on an average sequence of data, not on the sequence of available observations.
A discriminatory rule is considered good if it is consistent, meaning it satisfies the following

$$\lim_{n \to \infty} E\{L_n\} = L^*$$

or alternatively, if $L_n \to L^*$ in probability when $n \to \infty$.

The consistency of the rule implies that we can reconstruct the unknown distribution of (X, Y) by taking more samples, as L_n can be as close to L^* as we want. Without this guarantee, taking more samples would not be of interest. A rule that is consistent for all distributions of (X, Y) is referred to as *universally consistent*.

Note 2.1.1. *After this general introduction to the world of classifiers, we will focus on binary classifiers, where the class Y only takes values on $\{0, 1\}$. We make this simplification in order to avoid some technicalities that are not very relevant, since it is also relatively easy to obtain the solution for any $M \geq 2$ from this case. Finally, the problem that arises in this work of determining whether the parts are defective or fit for sale belongs to this type.*

2.2 Classifier consistency

This subsection is devoted to the in deep study of the notion of consistency in classifiers which were introduced in the previous section.

Given a sequence of training data, $D_n = ((X_1, Y_1), ..., (X_n, Y_n)),\ n \in \mathbb{N}$, the best we can hope from a classification function is to achieve the Bayes error probability, L^*. In general, it is not possible to achieve the *Bayesian error probability* exactly, but it is possible to create a sequence of classification functions, called a classification rule, such as $\{g_n\}$, that can make the error probability $L(g_n) = P\{g_n(X, D_n) \neq Y | D_n\}$ approach L^* with high probability.

Definition 2.2.1 (Weak and strong consistency)**.** *(See [3], pages 91-92) A classification rule is weakly consistent (or asymptotically Bayes-risk efficient) for a certain distribution of* (X, Y) *if*

$$E\{L_n\} = P\{g_n(X, D_n) \neq Y\} \to L^* \text{ as } n \to \infty$$

and strongly consistent if

$$\lim_{n\to\infty} L_n = L^* \text{ with probability 1.}$$

A decision rule can be consistent for certain distributions of (X, Y), but may not be consistent for others. It is desirable to have a rule that is consistent for a large class of distributions. Since there are many situations in which there is no prior information about the distribution, it is important to have a rule that works well for all the distributions.

Definition 2.2.2 (Universal consistency)**.** *(See [3], page 92) A sequence of decision rules is called universally (strongly) consistent if it is (strongly) consistent for every distribution of the pair* (X, Y)*.*

Proving universal consistency for a classification rule $\{g_n\}$ does not guarantee that g_n works well for a particular classification task and a fixed n. In fact, for every rule and every decreasing null sequence $\{a_n\} \subset \left(0, \frac{1}{16}\right]$ there exists a distribution P with $L^* = 0$ and

$$E\{L(g_n)\} \geq a_n \quad \text{for all } n \geq 1$$

It can be deduced that there is no classifier for which a positive, increasing, and unbounded sequence $\{\alpha_n\}$ and a real number $p > 0$ exist such that

$$P\{|L(g_n) - L^*| \geq \epsilon\} \leq e^{-c\epsilon^p \alpha_n}$$

holds for all distributions P on $\mathbb{R}^d \times \{0, 1\}$ and all $n \geq 1$ even if $c > 0$ depends on P, i.e., although there is universal consistency, this consistency is not uniform in P.

Therefore, any study on the rate of convergence of a specific classifier must limit the class of distributions considered. [11]

2.3 The Bayes classifier

In this subsection we characterize the Bayes classifier and its plug in situations. These results will be used later in the study of the universal consistency of the kNN rule.

Let (X, Y) be a pair of random variables with values in $\mathbb{R}^d \times \{0, 1\}$.
The random pair (X, Y) is determined by the pair (μ, η), where μ represents the probability distribution of X, and η stands for the regression of Y on X. More precisely, for a Borel-measurable set $A \subseteq \mathbb{R}^d$,

$$\mu(A) = P\{X \in A\}$$

and for any $x \in \mathbb{R}^d$,

$$\eta(x) = P\{Y = 1 | X = x\} = E\{Y | X = x\}.$$

If $C \subseteq \mathbb{R}^d \times \{0, 1\}$ is a Borel set, we have that

$$C = (C_0 \times \{0\}) \cup (C_1 \times \{1\})$$

and so

$$\begin{aligned} P\{(X, Y) \in C\} &= P\{X \in C_0, Y = 0\} + P\{X \in C_1, Y = 1\} \\ &= \int_{C_0} (1 - \eta(x))\mu(dx) + \int_{C_1} \eta(x)\mu(dx). \end{aligned}$$

A classifier has been defined as a function $g : \mathbb{R}^d \to \{0, 1\}$. We will see that the Bayes classifier (or decision function) is:

$$g^*(x) = \begin{cases} 1 & \text{if} \quad \eta(x) > 1/2 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

which is the function that minimizes the probability of error.

Theorem 2.3.1. *(See [3], pages 9-11) For any decision function $g : \mathbb{R}^d \to \{0, 1\}$, the classifier g^* as defined in (1) satisfies*

$$P\{g^*(X) \neq Y\} \leq P\{g(X) \neq Y\}$$

that is, $g^*(x)$ is the optimal decision.

Proof. Given $X = x$, the conditional probability of error for any decision g is

$$\begin{aligned} P\{g(X) \neq Y|X = x\} &= P\{Y = 1, g(X) = 0|X = x\} + P\{Y = 0, g(X) = 1|X = x\} \\ &= 1 - P\{g(X) = Y|X = x\} \\ &= 1 - (P\{Y = 1, g(X) = 1|X = x\} + P\{Y = 0, g(X) = 0|X = x\}) \\ &= 1 - (I_{\{g(x)=1\}}P\{Y = 1|X = x\} + I_{\{g(x)=0\}}P\{Y = 0|X = x\}) \\ &= 1 - (I_{\{g(x)=1\}}\eta(x) + I_{\{g(x)=0\}}(1 - \eta(x))) \end{aligned}$$

where I_A denotes the indicator of the set A. Consequently, for every $x \in \mathbb{R}^d$,

$$\begin{aligned} &P\{g(X) \neq Y|X = x\} - P\{g^*(X) \neq Y|X = x\} \\ &\quad = \eta(x)(I_{\{g^*(x)=1\}} - I_{\{g(x)=1\}}) + (1 - \eta(x))(I_{\{g^*(x)=0\}} - I_{\{g(x)=0\}}) \\ &\quad = (2\eta(x) - 1)(I_{\{g^*(x)=1\}} - I_{\{g(x)=1\}}) \geq 0, \end{aligned} \tag{2}$$

by the definition of g^*. Integrating on both sides with respect $\mu(dx)$ gives the desired result. □

The function η is usually unknown. Suppose then that a function $\tilde{\eta} \in [0, 1]$ approximating η can be calculated. In this case, it would be logical to use the plug-in decision function to approximate the Bayes decision. The plug-in function is defined as

$$\tilde{g}(x) = \begin{cases} 0 & \text{if} \quad \tilde{\eta}(x) \leq 1/2 \\ 1 & \text{otherwise} \end{cases}$$

If $\tilde{\eta}(x)$ is close to $\eta(X)$ in the L1-sense, then the error probability of the decision $\tilde{g}$ is near the optimal decision g^* (see [3], pages 15-16):

Theorem 2.3.2. *For the error probability of the plug-in decision $\tilde{g}$ defined above, we have*

$$P\{\tilde{g}(X) \neq Y\} - L^* = 2\int_{\mathbb{R}^d} |\eta(x) - 1/2| I_{\{g^*(x)\neq\tilde{g}(x)\}}\mu(dx)$$

and

$$P\{\tilde{g}(X) \neq Y\} - L^* \leq 2\int_{\mathbb{R}^d} |\eta(x) - \tilde{\eta}(x)|\mu(dx) = 2E\{|\eta(X) - \tilde{\eta}(X)|\}.$$

Proof. Notice that if $x \in \mathbb{R}^d$ satisfies that $\tilde{g}(x) \neq g^*(x)$, then $(I_{\{g^*(x)=1\}} - I_{\{\tilde{g}(x)=1\}}) \in \{-1, 1\}$. Thus, (2) gives

$$P\{\tilde{g}(X) \neq Y|X = x\} - P\{g^*(X) \neq Y|X = x\} = |2\eta(x) - 1| I_{\{\tilde{g}(x)\neq g^*(x)\}}$$

Since $\tilde{g}(x) \neq g^*(x)$ implies $|\eta(x) - 1/2| \leq |\eta(x) - \tilde{\eta}(x)|$ we finally have that

$$P\{\tilde{g}(X) \neq Y\} - P\{g^*(X) \neq Y\} = \int_{\mathbb{R}^d} 2|\eta(x) - 1/2| I_{\{\tilde{g}(x)\neq g^*(x)\}}\mu(dx) \leq \int_{\mathbb{R}^d} 2|\eta(x) - \tilde{\eta}(x)|\mu(dx).$$

□

The following corollary is a direct result of Theorem 2.3.2 (see [3], page 93).

Corollary 2.3.2.1. *The error probability of the classifier*

$$g_n(x) = \begin{cases} 0 & \text{if} \quad \eta_n(x) \leq 1/2 \\ 1 & \text{otherwise,} \end{cases}$$

satisfies the inequality

$$L(g_n) - L^* \leq 2 \int_{\mathbb{R}^d} |\eta(x) - \eta_n(x)| \mu(dx) = 2E\{|\eta(X) - \eta_n(x)||D_n\}.$$

Applying the Cauchy-Schwarz inequality we obtain the following result.

Corollary 2.3.2.2. *The error probability of g_n satisfies*

$$P\{g_n(X) \neq Y|D_n\} - L^* \leq 2\sqrt{\int_{\mathbb{R}^d} |\eta(x) - \eta_n(x)|^2 \mu(dx)}.$$

3 Theoretical basis of mathematical models

This chapter provides the definitions of the kNN and SVM classifiers and an in-depth analysis of their universal consistency.

3.1 k-Nearest Neighbors

The kNN is a non parametric classification algorithm developed by Evelyn Fix and Joseph Hodges in 1951 [5]. It is a supervised learning algorithm which is known for its effectiveness, in addition to its simplicity of execution and low computation time [15].

Note 3.1.1. *Despite the asymptotic properties remain valid to a wide variety of metrics, the neighbors are defined in terms of the euclidean distance.*

Be $x \in \mathbb{R}^d$ the point whose label is to be predicted. The data $(X_1, Y_1), ..., (X_n, Y_n)$ are to be sorted with respect to the increasing values of $||x - X_i||$, where $|| \cdot ||$ denotes the euclidean distance. The reordered data sequences is denoted by

$$(X_{(1)}(x), Y_{(1)}(x)), ..., (X_{(n)}(x), Y_{(n)}(x))$$

where $X_{(k)}(x)$ is the k-th nearest neighbor of x (see [3], page 63).

Definition 3.1.2 (k-Nearest-Neighbors)**.** *(See [3], page 170) The k-Nearest-Neighbors rule is defined by the classifier*

$$g_n(x) = \begin{cases} 0 & \text{if} \quad \sum_{i=1}^{k} I_{\{Y_{(i)}(x)=1\}} \leq \sum_{i=1}^{k} I_{\{Y_{(i)}(x)=0\}} \\ 1 & \text{otherwise} \end{cases} .$$

In other words, $g_n(x)$ is a majority vote among the labels of the k nearest neighbors of x.

To study the universal consistency of this method, we must first examine some auxiliary results that will aid in understanding and proving the main theorem. Let us begin by defining a cone (see [3], page 66).

Definition 3.1.3. *(See [3], page 66) For $\theta \in (0, \pi/2)$ and $x \in \mathbb{R}^d$, the cone $C(x, \theta)$ is defined as the collection of all the $y \in \mathbb{R}^d$ for which $angle(x, y) \leq \theta$, or equivalently :*

$$C(x, \theta) = \left\{ y \in \mathbb{R}^d : \frac{x^T y}{||x||||y||} \geq \cos(\theta) \right\}.$$

In addition, the set $s + C(x, \theta)$ is the translation of $C(x, \theta)$ by s, so

$$s + C(x, \theta) = \left\{ y \in \mathbb{R}^d : y = s \text{ or } angle(y - s, x) \leq \theta \right\}.$$

Lemma 3.1.4. *If $y, y' \in s + C(x, \pi/6)$, and $||y - s|| < ||y' - s||$, then $||y - y'|| < ||y' - s||$*

Proof. Let $y, y' \in s + C(x, \pi/6)$ with $||y - s|| < ||y' - s||$ and θ be the angle formed by $(y - s)$ and $(y' - s)$.

$$\begin{aligned} ||y - y'||^2 &= ||(y - s) - (y' - s)||^2 \\ &= ||y - s||^2 + ||y' - s||^2 - 2\,||y - s||\,||y' - s||\cos(\theta) \\ &\leq ||y - s||^2 + ||y' - s||^2 - 2\,||y - s||\,||y' - s||\cos(\pi/3) \\ &= ||y' - s||^2 \left(1 + \frac{||y - s||^2}{||y' - s||^2} - \frac{||y - s||}{||y' - s||}\right) \\ &< ||y' - s||^2. \end{aligned}$$

□

Lemma 3.1.5. *(See [3], pages 67-68) Let $\theta \in (0, \pi/2)$ be fixed. Then there exists a set $\{x_1, ..., x_{\gamma_d}\} \subset \mathbb{R}^d$ such that*

$$\mathbb{R}^d = \bigcup_{i=1}^{\gamma_d} C(x_i, \theta).$$

Furthermore, it is always possible to take

$$\gamma_d \leq \left(1 + \frac{1}{\sin(\theta/2)}\right)^d - 1.$$

Proof. Let $\theta \in (0, \pi/2)$ and let S_1^d be the unit d-dimensional sphere.
Without loss of generality, let S_i^d be the sphere with centre $x_i \in S_1^d$ and radius $r = \sin(\theta/2)$. We have that

$$S_1^d \cap S_i^d = S_1^d \cap C(x_i, \theta)$$

Given $\gamma_d \in \mathbb{N}$ we take the set $\{x_1, ..., x_{\gamma_d}\} \subset S_1^d$ with the property $||x_i - x_j|| \geq r$ for all $j \neq i$. We see that it is true that $\bigcup C(x_i, \theta) = \mathbb{R}^d$ if and only if $S_1^d \subset \cup S_i^d$. We have then that the spheres $S_i^{d'}$ with radius $r/2$ and centres x_i are disjoint and furthermore $\bigcup S_i^{d'} \subseteq S_{1+\frac{r}{2}}^d \backslash S_{\frac{r}{2}}^d$.

Call v_d to the volume of S_1^d. Then

$$\gamma_d v_d \left(\frac{r}{2}\right)^d \leq v_d \left(1+\frac{r}{2}\right)^d - v_d \left(\frac{r}{2}\right)^d$$

and, then,

$$\gamma_d \leq \left(1+\frac{2}{r}\right)^d - 1 = \left(1+\frac{1}{\sin(\theta/2)}\right)^d - 1.$$

□

Note 3.1.6. *In particular, if* $\theta = \pi/6$ *then* $\gamma_d \leq \left(1+\frac{2}{\sqrt{2-\sqrt{3}}}\right)^d - 1$.

Lemma 3.1.7 (Stone 1977)**.** *(See [3], pages 65-69) If* $X_1, ..., X_n$ *and* X *are i.i.d random vectors , then for every integrable function* f, $n \in \mathbb{N}$ *and* $k \leq n$

$$\sum_{i=1}^{k} E\{|f(X_{(i)}(X))|\} \leq k\gamma_d E\{|f(X)|\}$$

where $\gamma_d \leq \left(1+\frac{2}{\sqrt{2-\sqrt{3}}}\right)^d - 1$ *depends upon the dimension only.*

Proof. Let $\theta = \pi/6$, by the Lemma 3.1.5 we can cover $\mathbb{R}^d$ with γ_d cones of the form $X + C(x_j, \pi/6)$, $1 \leq j \leq \gamma_d$.
For each j, we can mark the k nearest neighbours of X in the set $\{X_1, ..., X_n\} \cap C(x_j, \pi/6)$, and if there are less than k points in a cone, we mark all of them. If $X_i \in X + C(x_j, \pi/6)$ is not marked, then X cannot be among the k nearest neighbours of X_i in $\{X_1, ..., X_{i-1}, X, X_{i+1}, ..., Xn\}$ due to the property appearing in the Lemma 3.1.4. So if f is a non-negative function,

$$\sum_{i=1}^{k} E\{|f(X_{(i)}(X))|\} = E\left\{\sum_{i=1}^{n} I_{\{X_i \text{ is among the k nearest neighbors of } X \text{ in } \{X_1,...,Xn\}\}} f(X_i)\right\}$$

$$= E\left\{f(X)\sum_{i=1}^{n} I_{\{X \text{ is among the k nearest neighbors of } X_i \text{ in } \{X_1,...,X_{i-1},X,X_{i+1},...,Xn\}\}}\right\}$$

$$\leq E\left\{f(X)\sum_{i=1}^{n} I_{\{X_i \, is \, marked\}}\right\} \leq k\gamma_d E\{f(X)\}$$

where the second equality holds because $X_1, ..., X_n$ and X are i.i.d random vectors and consequently, we can permute X and X_i in each summand. The last inequality comes from the fact that at most k X_i have been marked in every cone $C(x_j, \pi/6)$.

□

A general theorem by Stone allows us to deduce universal consistency for several classification rules. Consider a rule based on an estimate of the regression function η of the form

$$\eta_n(x) = \sum_{i=1}^{n} I_{\{Y_i=1\}} W_{ni}(x) = \sum_{i=1}^{n} Y_i W_{ni}(x)$$

where the weights $W_{ni}(x) = W_{ni}(x, X_1, ..., X_n)$ are non-negative and add to one.

Thus, η_n might be viewed as a local average estimator, and g_n a local (weighted) majority vote.

Notice that η_n is a weighted average estimator of η. It is intuitively clear that the pairs (X_i, Y_i) such that X_i is close to x should provide more information about $\eta(x)$ than those far form x. Thus, the weights should be much larger in the neighborhood of x, so η_n is roughly a (weighted) relative frequency of the X_i's that have label 1 among points in the neighborhood of x (see [3], pages 97-98).

The classification rule is defined as

$$g_n(x) = \begin{cases} 0 & \text{if} \quad \eta_n(x) \leq 1/2 \\ 1 & \text{otherwise} \end{cases} \quad . \tag{3}$$

Theorem 3.1.8 (Stone (1977)). *(See [3], pages 98-100) Assume that for any distribution of X, the weights satisfy the following three conditions:*

1. *There exists a constant c such that, for every non-negative measurable function f satisfying $E\{f(X)\} < \infty$,*

$$E\left\{\sum_{i=1}^{n} W_{ni}(X) f(X_i)\right\} \leq cE\{f(X)\}.$$

2. *For every $a > 0$*

$$\lim_{n\to\infty} E\left\{\sum_{i=1}^{n} W_{ni}(X) I_{\{\|X_i - X\| > a\}}\right\} = 0.$$

3.

$$\lim_{n\to\infty} E\left\{\max_{1\leq i\leq n} W_{ni}(X)\right\} = 0.$$

Then the classifier g_n, as defined in (3), is universally consistent.

Proof. By Corollary 2.3.2.2 it suffices to show that for every distribution of (X, Y)

$$\lim_{n\to\infty} E\{(\eta(X) - \eta_n(X))^2\} = 0.$$

Introduce the notation

$$\hat{\eta}_n(x) = \sum_{i=1}^{n} \eta(X_i) W_{ni}(x).$$

Then by the simple inequality $(a+b)^2 \le 2(a^2+b^2)$ we have

$$\begin{aligned} E\{(\eta(X) - \eta_n(X))^2\} &= E\{((\eta(X) - \hat{\eta}_n(X)) + (\hat{\eta}_n(X) - \eta_n(X)))^2\} \\ &\le 2(E\{(\eta(X) - \hat{\eta}_n(X))^2\} + E\{(\hat{\eta}_n(X) - \eta_n(X))^2\}). \end{aligned} \tag{4}$$

It is therefore sufficient to show that both terms on the right-hand side go to zero. Since the W_{ni}'s are non-negative and add one, by Jensen's inequality, the first term is

$$E\{(\eta(X) - \hat{\eta}_n(X))^2\} \le E\left\{\sum_{i=1}^{n} W_{ni}(X)(\eta(X) - \eta(X_i))^2\right\}.$$

Let C_B^1 denote the set of the continuous, $[0,1]$-valued, with bounded support functions. If $\eta^* \in C_B^1$ it is also uniformly continuous and given $\epsilon > 0$, there exists $a > 0$ such that for $\|x_1 - x\| < a$, $|\eta^*(x_1) - \eta^*(x)|^2 < \epsilon$ (recall that $\|x\|$ denotes the Euclidean norm of $x \in \mathbb{R}^d$). Thus, since $|\eta^*(x_1) - \eta^*(x)| \le 1$,

$$\begin{aligned} &E\left\{\sum_{i=1}^{n} W_{ni}(X)(\eta^*(X) - \eta^*(X_i))^2\right\} \\ &\quad\le E\left\{\sum_{i=1}^{n} W_{ni}(X) I_{\{\|X - X_i\| \ge a\}}\right\} + E\left\{\sum_{i=1}^{n} W_{ni}(X)\epsilon\right\} \to \epsilon, \end{aligned}$$

by Assumption 2. Since the set of continuous functions with bounded support is dense in $L_2(\mu)$, there exists $\eta^* \in C_B^1$ such that $E\{(\eta(X) - \eta^*(X))^2\} < \epsilon$; and from here:

$$\begin{aligned} &E\{(\eta(X) - \hat{\eta}_n(X))^2\} \\ &\quad\le E\left\{\sum_{i=1}^{n} W_{ni}(X)(\eta(X) - \eta(X_i))^2\right\} \\ &\quad\le 3E\left\{\sum_{i=1}^{n} W_{ni}(X)((\eta(X) - \eta^*(X))^2 + (\eta^*(X) - \eta^*(X_i))^2 + (\eta^*(X_i) - \eta(X_i))^2\right\} \\ &\quad\le 3E\{(\eta(X) - \eta^*(X))^2\} + 3E\left\{\sum_{i=1}^{n} W_{ni}(X)(\eta^*(X) - \eta^*(X_i))^2\right\} + 3cE\{(\eta(X) - \eta^*(X))^2\} \end{aligned}$$

where we have used the Cauchy-Schwarz inequality, the fact $(a+b+c)^2 \leq 3(a^2+b^2+c^2)$, and Assumption 1. Therefore,

$$\limsup_{n\to\infty} E\{(\eta(X) - \hat{\eta}_n(X))^2\} \leq 3\epsilon(1+1+c).$$

To handle the second term on the right-hand side of (4), observe that

$$E\{(Y_i - \eta(X_i))(Y_j - \eta(X_j))\} = 0 \text{ for all } i \neq j$$

by independence. Therefore,

$$\begin{aligned}
E\{(\hat{\eta}_n(X) - \eta_n(X))^2\} &= E\left\{\left(\sum_{i=1}^n W_{ni}(X)(\eta(X_i) - Y_i\right)^2\right\} \\
&= \sum_{i,j=1}^n E\{W_{ni}(X)(\eta(X_i) - Y_i)W_{nj}(X)(\eta(X_j) - Y_j)\} \\
&= \sum_{i=1}^n E\{W_{ni}^2(X)(\eta(X_i) - Y_i)^2\} \\
&\leq E\left\{\sum_{i=1}^n W_{ni}^2(X)\right\} \leq E\left\{\max_{1\leq i\leq n} W_{ni}(X) \sum_{j=1}^n W_{nj}(X)\right\} \\
&= E\left\{\max_{1\leq i\leq n} W_{ni}(X)\right\}
\end{aligned}$$

which converges to zero by Assumption 3 □

Now we are going to show that if $k_n \to \infty$ with $k_n/n \to 0$, the kNN classification rule is weakly universally consistent. The proof is a very simple application of Stone's theorem. This result, appearing in Stone's paper (1977) [12], was the first universal consistency result for any rule (see [3], pages 100-101).

In order to simplify the notation we will omit the dependence of k on n.

Theorem 3.1.9 (Stone (1977))**.** *(See [3], page 101) If $k \to \infty$ with $k/n \to 0$, then for every distribution of (X, Y) we have that $E\{L_n\} \to L^*$.*

Proof. First, we check the conditions of Stone's weak convergence Theorem 3.1.8. Let us define the weight W_{ni} as

$$W_{ni} = \begin{cases} 1/k & \text{if } X_i \text{ is among the } k \text{ nearest neighbors of } X \\ 0 & \text{otherwise} \end{cases}$$

Let us check the assumptions in Theorem 3.1.8. Assumption 3 is obvious since $k \to \infty$. Observe that Assumption 2 holds when

$$P\left\{||X_{(k)}(X) - X|| > \epsilon\right\} \to 0$$

But in Lemma .0.2 from the Appendix 1 we show that this is true for all $\epsilon > 0$ whenever $k/n \to 0$.
Finally, Assumption 3 was shown in Lemma 3.1.7 with $c = \gamma_d$. □

To prove the desired theorem, we require a generalisation of Lemma 3.1.7. Let us examine the following outcome.

Lemma 3.1.10 (Devroye and Györfi (1985)). *(See [3], page 171) If $B_a(x') = \{x : \mu(S^d_{x,||x-x'||})) \leq a\}$, where $S^d_{x,||x-x'||}$ is the d-dimensional sphere with centre x and radius $||x - x'||$, then for all $x' \in \mathbb{R}^d$*

$$\mu(B_a(x')) \leq \gamma_d a.$$

Proof. For $x \in \mathbb{R}^d$ let us consider the cone $x + C(s, \pi/6) \subset \mathbb{R}^d$.

Due to Lemma 3.1.4 we have that if $y, y' \in x + C(s, \pi/6)$, and $||x - y|| < ||x - y'||$, then $||y - y'|| < ||x - y'||$.

By Lemma 3.1.5 there exists $C_1, ..., C_{\gamma_d}$ a collection of cones centered at x' with different central directions covering $\mathbb{R}^d$. Then

$$\mu(B_a(x')) \leq \sum_{i=1}^{\gamma_d} \mu(C_i \cap B_a(x')).$$

Let $x^* \in C_i \cap B_a(x')$. Then by the property of the cones mentioned above we have

$$\mu(C_i \cap S^d_{x',||x'-x^*||} \cap B_a(x')) \leq \mu(S^d_{x^*,||x'-x^*||}) \leq a$$

where we use the fact that $x^* \in B_a(x')$. Since x^* is arbitrary.

$$\mu(C_i \cap B_a(x')) \leq a$$

which completes the proof of the lemma.

□

Lemma 3.1.10 implies that the number of points among $X_1, ..., X_n$ for which X is one of their k-nearest neighbors is at most a constant multiple of k.

Corollary 3.1.10.1. *(See [3], page 171) It happens that*

$$\sum_{i=1}^{n} I_{\{X \text{ is among the } kNN\text{'s of } X_i \{X_1,...,X_n,X\}-\{X_i\}\}} \leq k\gamma_d$$

Proof. Apply Lemma 3.1.10 with $a = k/n$ and let μ be the empirical measure μ_n of $X_1, ..., X_n$, that is, for each Borel set $A \subseteq \mathbb{R}^d$, $\mu_n(A) = (1/n)\sum_{i=1}^{n} I_{\{X_i \in A\}}$. □

Next we introduce without proof the well known McDiarmid inequality for further reference. The interested reader can find the proof of this inequality in [3], pages 136 and 137.

Theorem 3.1.11 (McDiarmid (1989))**.** *Let $X_1, ..., X_n$ be independent random variables taking values in a set A, and assume that $f : A^n \to \mathbb{R}$ satisfies*

$$\sup_{\substack{x_1,...,x_n \\ x_i' \in A}} |f(x_1, ..., x_n) - f(x_1, ..., x_{i-1}, x_i', x_{i+1}, ..., x_n)| \leq c_i, \quad 1 \leq i \leq n, \tag{5}$$

for any $x_i, X_i' \in A$, $i = 1, ..., n$.
Then for all $\epsilon > 0$

$$P\{|f(X_1, ..., X_n) - E\{f(X_1, ..., X_n)\}| \geq \epsilon\} \leq e^{-2\epsilon^2/\sum_{i=1}^{n} c_i^2}$$

Finally, the strong universal consistency for the kNN rule is shown in the following result.

Theorem 3.1.12 (Devroye and Györfi (1985), Zhao (1987))**.** *(See [3], pages 170-174) Assume that μ has a density. If $k \to \infty$ and $k/n \to 0$ then for every $\epsilon > 0$, there exists $n_0 \in \mathbb{N}$ such that for $n > n_0$*

$$P\{L_n - L^* > \epsilon\} \leq 2e^{-n\epsilon^2/(72\gamma_d^2)},$$

where γ_d is the minimal number of cones centered at the origin of angle $\pi/6$ that cover $\mathbb{R}^d$. Thus, the kNN rule is strongly consistent.

Proof. Let us remember that the decision rule is written as

$$g_n(x) = \begin{cases} 0 & \text{if} \quad \eta_n(x) \leq 1/2 \\ 1 & \text{otherwise,} \end{cases}$$

where η_n is the corresponding regression function estimate $\eta_n = \frac{1}{k}\sum_{i=1}^{k} Y_{(i)}(x)$.

The result follows from Theorem 2.3.2 if we show that for sufficiently large n

$$P\left\{\int |\eta(x) - \eta_n(x)|\mu(dx) > \frac{\epsilon}{2}\right\} \leq 2e^{-n\epsilon^2/(72\gamma_d^2)}.$$

Define $\rho_n(x)$ as any solution of the equation

$$\frac{k}{n} = \mu(S^d_{x,\rho_n(x)}).$$

Note that the absolute continuity of μ implies that a solution always exists. Also define

$$\eta_n^*(x) = \frac{1}{k}\sum_{j=1}^{n} Y_j I_{\{||X_j - x|| < \rho_n(x)\}}.$$

The basis of the proof is the following decomposition:

$$|\eta(x) - \eta_n(x)| \leq |\eta_n(x) - \eta_n^*(x)| + |\eta_n^*(x) - \eta(x)|. \tag{6}$$

For the first term on the right-hand side, observe that denoting $R_n(x) = ||X_{(k)}(x) - x||$,

$$\begin{aligned} |\eta_n(x) - \eta_n^*(x)| &= \frac{1}{k}\left|\sum_{j=1}^{n} Y_j I^d_{\{X_j \in S_{x,\rho_n(x)}\}} - \sum_{j=1}^{n} Y_j I^d_{\{X_j \in S_{x,R_n(x)}\}}\right| \\ &\leq \frac{1}{k}\sum_{j=1}^{n}\left|I^d_{\{X_j \in S_{x,\rho_n(x)}\}} - I^d_{\{X_j \in S_{x,R_n(x)}\}}\right| \qquad (7) \\ &= \left|\frac{1}{k}\sum_{j=1}^{n} I^d_{\{X_j \in S_{x,\rho_n(x)}\}} - 1\right| = |\hat{\eta}_n^*(x) - \hat{\eta}(x)|, \end{aligned}$$

where $\hat{\eta}_n^*(x)$ is defined as η_n^* with Y replaced by the constant random variable $\hat{Y} = 1$, and $\hat{\eta} = 1$ is the corresponding regression function.

Let us examine the first summand of (6). From (7) and using the Cauchy-Schwarz inequality, we have

$$\begin{aligned} E\left\{\int |\eta_n^*(x) - \eta_n(x)|\mu(dx)\right\} &\leq \left\{\int E|\hat{\eta}_n^*(x) - \hat{\eta}(x)|\mu(dx)\right\} \\ &\leq \int \sqrt{E\{|\hat{\eta}_n^*(x) - \hat{\eta}(x)|^2\}}\mu(dx) \\ &= \int \sqrt{\frac{1}{k^2} n \mathrm{Var}\{I_{\{X \in S^d_{x,\rho_n(x)}\}}\}}\mu(dx) \\ &\leq \int \sqrt{\frac{1}{k^2} n \mu(S^d_{x,\rho_n(x)})}\mu(dx) \\ &= \int \sqrt{\frac{n}{k^2}\frac{k}{n}}\mu(dx) = \frac{1}{\sqrt{k}} \end{aligned}$$

which converges to zero.

For the expected value of the second term on the right-hand side of (6), note that in the proof of Theorems 3.1.8 and 3.1.9 we already showed that

$$\lim_{n\to\infty} E\left\{\int |\eta(x) - \eta_n(x)|\mu(dx)\right\} = 0;$$

and, consequently

$$E\left\{\int |\eta_n^*(x)-\eta(x)|\mu(dx)\right\}\to 0.$$

Assume now that n is so large that

$$E\left\{\int |\hat{\eta}_n^*(x)-\hat{\eta}(x)|\mu(dx)\right\}+E\left\{\int |\eta_n^*(x)-\eta(x)|\mu(dx)\right\}<\frac{\epsilon}{6}.$$

Then, by (6) and (7), we have

$$\begin{aligned}
&P\left\{\int |\eta(x)-\eta_n(x)|\mu(dx)>\frac{\epsilon}{2}\right\}\\
\leq &P\left\{\int |\eta_n^*(x)-\eta(x)|\mu(dx)-E\left\{\int |\eta_n^*(x)-\eta(x)|\mu(dx)\right\}>\frac{\epsilon}{6}\right\} \qquad (8)\\
&+P\left\{\int |\hat{\eta}_n^*(x)-\hat{\eta}(x)|\mu(dx)-E\left\{\int |\hat{\eta}_n^*(x)-\hat{\eta}(x)|\mu(dx)\right\}>\frac{\epsilon}{6}\right\}.
\end{aligned}$$

Next we get an exponential bound for the first probability on the right-hand side of (8) by using McDiarmid's inequality.
Fix an arbitrary realization of the data $D_n=(x_1,y_1),...,(x_n,y_n)$, and replace (x_i,y_i) by $(\hat{x}_i,\hat{y}_i)$, changing the value of $\eta_n^*(x)$ to $\eta_{ni}^*(x)$. Then

$$\left|\int |\eta_n^*(x)-\eta(x)|\mu(dx)-\int |\eta_{ni}^*(x)-\eta(x)|\mu(dx)\right|\leq\int |\eta_n^*(x)-\eta_{ni}^*(x)|\mu(dx).$$

But $|\eta_n^*(x)-\eta_{ni}^*(x)|$ is bounded by $2/k$ and can differ from zero only if $\|x-x_i\|<\rho_n(x)$ or $\|x-\hat{x}_i\|<\rho_n(x)$ if and only if $\mu(S^d_{x,\|x-x_i\|})<k/n$. But the measure of such x's is bounded by $\gamma_d k/n$ by Lemma 3.1.10. Therefore,

$$\sup_{x_1,y_1,...,x_n,y_n,\hat{x}_i,\hat{y}_i}\int |\eta_n^*(x)-\eta_{ni}^*(x)|\mu(dx)\leq\frac{2}{k}\frac{\gamma_d k}{n}=\frac{2\gamma_d}{n}$$

and by McDiarmid's inequality

$$P\left\{\left|\int |\eta(x)-\eta_n^*(x)|\mu(dx)-E\left\{\int |\eta(x)-\eta_n^*(x)|\mu(dx)\right\}\right|>\frac{\epsilon}{6}\right\}\leq e^{-n\epsilon^2/(72\gamma_d^2)}.$$

Finally, we need a bound for the second term on the right-hand side of (8). This probability may be bound by McDiarmid's inequality exactly the same way as for the first term, obtaining

$$P\left\{\left|\int |\hat{\eta}_n^*(x)-\hat{\eta}(x)|\mu(dx)-E\left\{\int |\hat{\eta}_n^*(x)-\hat{\eta}(x)|\mu(dx)\right\}\right|>\frac{\epsilon}{6}\right\}\leq e^{-n\epsilon^2/(72\gamma_d^2)}.$$

□

3.2 Support Vector Machine

The SVM is a supervised learning algorithm used in many classification and regression problems, such as signal and natural language processing or image and speech recognition [14]. The SVM algorithm has its origins in statistical learning theory and was introduced in the 90s by Vapnik and his collaborators Boser et al. (1992) [13].

The material in this section is taken from [11] and [10].

The SVM is a mathematical algorithm that aims to find a hyperplane with a d-dimensional band around it as wide as possible, with no points in it, and all points of one category on one side of the band and those of the other category on the other one. The points that determine the direction of the hyperplane and the width of the band are called support vectors.

In classification problems, finding a linear separator between the two classes involved is usually impossible. Instead, a transformation is constructed, denoted as h:

$$h : \mathbb{R}^d \to H, \tag{9}$$

where H is a high-dimensional or even infinite-dimensional Hilbert space.

This transformation allows complicated curves in $\mathbb{R}^d$ to become hyperplanes in H. However, when making this change, it is expected that both H and h would be complex objects. The inverse by h of the corresponding half-spaces must be calculated to determine what is being dealt with in the original space [7].

For the description of the algorithm we assume that we have the training set $D_n = (x_1, y_1), ...,$ $(x_n, y_n) \in \mathbb{R}^d \times \{1, 1\}$ and we are going to suppose that the two classes for Y are 1 and -1.

We recover the probability of error for a classifier of f_n from the task of thinking about consistency as

$$L(f_n) = P\{f_n(X) \neq Y\}$$

and we reconstruct the definition of the Bayes classifier for this algorithm as

$$f^*(x) = \begin{cases} 1 & \text{if} \quad x \in \eta(x) \geq 1/2 \\ -1 & \text{if} \quad x \in \eta(x) < 1/2 \end{cases} .$$

So, now, the Bayes' error is

$$L^* = L(f^*) = \inf_{f:\mathbb{R}^d \to \{-1,1\}} L(f).$$

Suppose that we have a linearly separable data D_n, i.e, assume that there exist elements $w \in S_1^d$ and a real number $b \in \mathbb{R}$ with

$$\langle w, x_i \rangle + b > 0 \qquad \text{for all i with } y_i = 1$$

$$\langle w, x_i \rangle + b < 0 \qquad \text{for all i with } y_i = -1$$

In other words, D_n can be correctly separated by the affine linear hyperplane determined by w and b.

In this case the generalized portrait algorithm constructs the correctly separating hyperplane (w_n, b_n) that has maximal distance to the training points. The resulting decision function is defined by

$$f_n(x) := \text{sign}(\langle w_n, x \rangle + b_n) \quad \text{for all } x \in \mathbb{R}^d.$$

Until normalisation, (w_n, b_n) is the unique solution of the optimisation problem

$$\begin{cases} \min\limits_{b,w} & ||w|| \\ \text{subject to} & y_i(\langle w, x_i \rangle + b) \geq 1 \quad i = 1, ..., n \end{cases}$$

We can clearly see that this approach has two shortcomings: in the first term, a linear decision function may be inappropriate to discriminate between the two classes because it may happen that they are not linearly separable and so (w_n, b_n) may not exist.

Second, even if we have linearly separable data, in the presence of noise it may happen that any good decision function will misclassify some cases.

To try to fix the first problem, the SVM maps the input data $x_1, ..., x_n$ into a (possibly infinite dimensional) Hilbert space, H, called the *feature space*, using a non-linear function $\phi : X \to H$. Now, the approach of the generalised portrait algorithm is implemented in H instead of X, i.e. we simply replace x and the x_i's in the decision function and optimisation problem by $\phi(x)$ and the $\phi(x_i)$'s, and the vector w in the optimisation problem is chosen in H. The corresponding algorithm is called *Maximal Margin Classifier* and was the first classifier of the SVM type.

To avoid the second problem, the linear constraints in the optimisation problem are relaxed to $y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i$ with $\xi_i \geq 0$. Then, to avoid trivial solutions, the objective function must also take into account the slack variables ξ_i. Combining both modifications can lead to the following quadratic optimisation problem.

$$\begin{cases} \min\limits_{b,w} & \langle w, w\rangle + c\sum_{i=1}^n \xi_i \\ \text{subject to} & y_i(\langle w, \phi(x_i)\rangle + b) \geq 1 - \xi_i \quad i = 1, ..., n \\ & \xi_i \geq 0 \quad i = 1, ..., n \end{cases}$$

where $c > 0$ is a free parameter that is usually set heuristically and is called regularisation parameter. Hereafter we denote a solution of this quadratic optimisation problem by $(w_n^{\phi,c}, b_n^{\phi,c}) \in H \times \mathbb{R}$.

Since the objective function of the quadratic optimisation problem remains convex, we can consider the Wolf Dual

$$\begin{cases} \max\limits_{c} & \sum_{i=1}^n \alpha_i - \frac{1}{4}\sum_{j,i=1}^n y_i y_j \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j)\rangle \quad i = 1, ..., n \\ \text{subject to} & \sum_{i=1}^n y_i \alpha_i = 0 \quad i = 1, ..., n \\ & 0 \leq \alpha_i \leq c \end{cases}$$

If $(\alpha_1^*, ..., \alpha_n^*)$ denotes a solution of the previous problem, then the solution vector $w_n^{\phi,c}$ is

$$w_n^{\phi,c} = \frac{1}{2}\sum_{i=1}^n y_i \alpha_i^* \phi(x_i)$$

and the corresponding bias $b_T^{\phi,c}$ by

$$b_n^{\phi,c} = y_j - \frac{1}{2}\sum_{i=1}^n y_i \alpha_i^* \phi(x_i)\langle \phi(x_i), \phi(x_j)\rangle$$

for every α_j^* with $0 < \alpha_j^* < c$.

An algorithm that provides the decision function

$$f_n(x) := \text{sign}(\langle w_n^{\phi,c}, \phi(x)\rangle + b_n^{\phi,c})$$

for every data D_n is called 1-*norm soft margin classifier* (1-SMC) with feature map ϕ and parameter c.

Note that only inner products of ϕ with itself occur, both in the optimisation problem and in the evaluation of the resulting decision function. Therefore, it is sufficient to know the function $\langle \phi(\cdot), \phi(\cdot)\rangle : \tilde{X} \times \tilde{X} \to \mathbb{R}$ where $\tilde{X} \subset \mathbb{R}$, and, it happens that it is not needed to compute the feature map. Even more, it is neither needed to know H.

Having proved that the kNN classifier is universally consistent, it remains an open question whether the SVM is universally consistent for a particular choice of parameters. The universal consistency of the 1-SMC is achieved by selecting the regularisation parameter in a distinctive way and using a kernel (see Definition 3.2.1) from a specific class, known as universal kernels.

Definition 3.2.1. *Let $(\tilde{X}, d)$ be a metric space. A function $k : \tilde{X} \times \tilde{X} \to \mathbb{R}$ is said to be a kernel on $\tilde{X}$ if there exists a Hilbert space H and a map $\phi : \tilde{X} \to H$ with*

$$k(x, y) = \langle \phi(x), \phi(y) \rangle$$

for all $x, y \in \tilde{X}$. We call ϕ a feature map and H a feature space of k.

Notably, both H and ϕ are not at all unique, but for a given kernel there is a canonical feature space (with associated feature map), which is, in fact, a so-called reproducing kernel Hilbert space (RKHS). However, the use of these spaces will be unnecessary in this problem.

Definition 3.2.2. *Let k be a kernel on $\tilde{X}$ and $\phi : \tilde{X} \to H$ be a feature map of k. A function $f : \tilde{X} \to \mathbb{R}$ is induced by k if there exists an element $w \in H$ such that $f(\cdot) = \langle w, \phi(\cdot) \rangle$.*

The next lemma shows that this definition is independent of ϕ and H.

Lemma 3.2.3. *Let $k : \tilde{X} \times \tilde{X} \to \mathbb{R}$ be a kernel and $\phi_1 : \tilde{X} \to H_1$, $\phi_2 : \tilde{X} \to H_2$ be two feature maps of k. Then for all $w_1 \in H_1$ there exist $w_2 \in H_2$ with $||w_2|| \leq ||w_1||$ and*

$$\langle w_1, \phi_1(x) \rangle \;=\; \langle w_2, \phi_2(x) \rangle, \qquad \textit{for all } x \in \tilde{X}.$$

Proof. Let $H_1^* := \overline{\left\langle \phi_1\left(\tilde{X}\right) \right\rangle}$ and $\tilde{H}_1$ its orthogonal complement in H_1. Then $w_1 \in H_1$ can be written as $w_1 = w_1^* + \tilde{w}_1$ with $w_1^* \in H_1^*$ and $\tilde{w}_1 \in \tilde{H}_1$. Given an $x \in \tilde{X}$ we have $\langle \tilde{w}_1, \phi_1(x) \rangle = 0$ and therefore we obtain $\langle w_1^*, \phi_1(x) \rangle = \langle w_1, \phi_1(x) \rangle$ for all $x \in \tilde{X}$. Now by the definition of H_1^* there exists a sequence $\left\{ w_n^{(1)} \right\} \subset \langle \phi_1(\tilde{X}) \rangle$ with $w_n^{(1)} = \sum_{m=1}^{m_n} \lambda_m^{(n)} \phi_1\left(x_m^{(n)}\right)$ and $w_1^* = \sum_{n=1}^{\infty} w_n^{(1)}$. Then for $w_n^{(2)} = \sum_{m=1}^{m_n} \lambda_m^{(n)} \phi_2\left(x_m^{(n)}\right)$ and $l_2 \geq l_1 \geq 1$, from the kernel definition we obtain

$$\begin{aligned}
\left|\left| \sum_{n=l_1}^{l_2} w_n^{(1)} \right|\right|^2 &= \sum_{n=l_1}^{l_2} \sum_{m=1}^{m_n} \sum_{i=l_1}^{l_2} \sum_{j=1}^{m_i} \lambda_m^{(n)} \lambda_j^{(i)} \left\langle \phi_1\left(x_m^{(n)}\right), \phi_1\left(x_j^{(i)}\right) \right\rangle \\
&= \sum_{n=l_1}^{l_2} \sum_{m=1}^{m_n} \sum_{i=l_1}^{l_2} \sum_{j=1}^{m_i} \lambda_m^{(n)} \lambda_j^{(i)} \left\langle \phi_2\left(x_m^{(n)}\right), \phi_2\left(x_j^{(i)}\right) \right\rangle \\
&= \left|\left| \sum_{n=l_1}^{l_2} w_n^{(2)} \right|\right|^2 .
\end{aligned}$$

Therefore, $\left\{\sum_{n=1}^{m} w_n^{(2)}\right\}_{m\geq 1}$ is a Cauchy sequence and therefore converges to $w_2 = \sum_{n=1}^{\infty} w_n^{(2)} \in H_2$. Obviously we then have $||w_2|| = ||w_1^*|| \leq ||w_1||$. Also, a simple calculation similar to the one above shows that $\langle w_1, \phi_1(x)\rangle = \langle w_2, \phi_2(x)\rangle$ for all $x \in \tilde{X}$. □

We introduce the space $C(\tilde{X})$ of all continuous functions $f : \tilde{X} \to \mathbb{R}$ on the compact metric space $(\tilde{X}, d)$ endowed with the usual supremum norm:

$$||f||_\infty := \sup_{x\in\tilde{X}} |f(x)|$$

which is a classic example of an algebra.

Definition 3.2.4. *A continuous kernel $k : \tilde{X} \times \tilde{X} \to \mathbb{R}$ on a compact metric space $(\tilde{X}, d)$ is called universal if the space of all functions induced by k is dense in $C(\tilde{X})$, i.e., for all $g \in C(\tilde{X})$ and all $\epsilon > 0$ there exits a function f induced by k with $||f - g||_\infty \leq \epsilon$.*

Examples of universal kernels are :

1. The polynomial kernel, $k(x, y) := (\langle x, y\rangle + 1)^p$ for all $p > 1$.

2. The sigmoid kernel, $k(x, y) := \tanh(\alpha\langle x, y\rangle + \beta)$ for all $\alpha, \beta \in \mathbb{R}$.

3. The Gaussian RBF kernel, $k(x, y) := \exp(-\sigma^2||x-y||_2^2)$ for all $\sigma > 0$ and all compact $\tilde{X} \subset \mathbb{R}^d$.

4. The kernel $k(x, y) := \exp(\langle x, y\rangle)$ for all compact subsets $\tilde{X} \subset \mathbb{R}^d$.

5. Vovk's real infinite polynomial, $k(x, y) := (1-\langle x, y\rangle)^{-\alpha}$ for all $\alpha > 0$ and all compact subsets $\tilde{X} \subset \{x \in \mathbb{R}^d : ||x||_2 < 1\}$.

6. The stronger regularized Fourier kernel, $k(x, y) := \prod_{i=1}^{d} \frac{1-q^2}{2(1-2q\cos(x_i-y_i)+q^2)}$ for all $0 < q < 1$ and all compact $\tilde{X} \subset [0, 2\pi)^d$.

7. The weaker regularized Fourier kernel $k(x, y) := \prod_{i=1}^{d} \frac{\pi}{2q\sinh(\pi/q)} \cosh(\frac{\pi-|x_i-y_i|}{q})$ for all $0 < q < \infty$ and all compact $\tilde{X} \subset [0, 2\pi)^d$.

The result where the universal consistency of the 1-SMC classifier is proven requires the use of covering numbers and outer probability.

Let us define the first concept.

Definition 3.2.5. *Assume we have a metric space $(\tilde{X}, d)$. For $\epsilon > 0$, we define the covering numbers of this metric space as*

$$N((\tilde{X}, d), \epsilon) := \inf \left\{ n \in \mathbb{N} : \text{ there exists } x_1, ..., x_n \text{ with } \tilde{X} \subset \bigcup_{i=1}^{n} B_d(x_i, \epsilon) \right\}$$

We have that the space $(\tilde{X}, d)$ is precompact if and only if $N((\tilde{X}, d), \epsilon)$ for all $\epsilon > 0$.

We also need the following result.

Lemma 3.2.6. *Let $k : \tilde{X} \times \tilde{X} \to \mathbb{R}$ be a universal kernel on a compact subset $\tilde{X}$ of $\mathbb{R}^d$ and $\phi : \tilde{X} \to H$ be a feature map of k. Then ϕ is continuous and the map*

$$d_k(x, y) := ||\phi(x) - \phi(y)||_k$$

defines a metric on $\tilde{X}$ such that $id : (\tilde{X}, d) \to (\tilde{X}, d_k)$ is continuous.

Proof. By definition of d_k and k

$$d_k(x, y) = \sqrt{k(x, x) - 2k(x, y) + k(y, y)}$$

we observe that $d_k(x, .) : (\tilde{X}, d) \to \mathbb{R}$ is continuous for every $x \in \tilde{X}$. In particular, $\{y \in \tilde{X} : d_k(x, y) < \epsilon\}$ is open with respect to d and consequently, $id : (\tilde{X}, d) \to (\tilde{X}, d_k)$ is continuous. □

Next, we introduce the concept of outer probability.

Definition 3.2.7. *For a probability space $(\Omega, \mathcal{A}, P)$, the outer probability of an arbitrary $B \subset \Omega$, denoted $P^*(B)$, is the infimum over all $P(A)$ such that $B \subset A$ and $A \in \mathcal{A}$ [16].*

Outer probabilities can be used for statistical inference instead of probability measures to provide flexibility in model specification [8].

Finally, the following theorem shows the strong universal consistency of the 1-norm soft margin classifier. The proof of this theorem is beyond the contents of this report but can be found at [11].

Theorem 3.2.8. *Let $\tilde{X} \subset \mathbb{R}^d$ be compact and k be a universal kernel on $\tilde{X}$ with $N((\tilde{X}, d), \epsilon) \in O(\epsilon^{-\alpha})$ for some $\alpha > 0$. Suppose that we have a positive sequence $\{c_n\}$ with $nc_n \to \infty$ and $c_n \in O(n^{\beta-1})$ for some $0 < \beta < \frac{1}{\alpha}$. Then for all Borel probability measures P on $\mathbb{R}^d \times \{-1, 1\}$ and all $\epsilon > 0$ we have*

$$\lim_{n\to\infty} P^* \left(\left\{ D_n \in (\mathbb{R}^d \times \{-1, 1\}) : L(f_n^{k,c_n}) \leq L^* + \epsilon \right\} \right) = 1$$

where P^* is the outer probability of P.

4 Results and analysis

In this section, the kNN and SVM classifiers are studied when applied to the particular case of the plastic moulding machine. Both algorithms are going to be compared using different statistical values.

The Python language was chosen for this study due to its ease of use, clear and concise syntax, and wide ecosystem of tools and libraries for data analysis and visualization. Additionally, it integrates easily with other technologies [17].

The kNN and SVM algorithms are included in the Scikit-learn library, which offers a wide range of machine learning algorithms. Let us explain the way in which we handle both algorithms.

Among the possibilities that Scikit-learn offers, we have chosen the Euclidean metric to compute distances between points in the kNN. We will employ the linear, polynomial (grade three by default), radial (rbf) and sigmoid kernels with the SVM algorithm (they are introduced theoretically in page 29).

In the binary case of SVM algorithm, the classifier gives a score for each sample and then probabilities are calibrated using logistic regression on the SVM's scores. This is then fit by a cross validation (see Subsection 4.2) on the training data. Details on this process can be seen in [9].

4.1 Statistical Parameters

In both cases, to study the quality of the classifier we compute the confusion matrix and three statistical parameters which are the accuracy, the precision and the recall.

We start by explaining the confusion matrix, since the parameters are derived from it.

The basic idea of the confusion matrix is to count the number of times a record entry of class A is classified as class A or B, where A and B can both be 0 or 1 (or in our case bad part and good part) [6].

Furthermore, there are four possibilities which are described below and that are represented in Figure 1:

- True Negatives (TN): is the number of instances that the algorithm correctly classified as negative. In this specific case, these are the damaged pieces class entries that have been classified as damaged.
- False Negatives (FN): is the number of instances that the algorithm incorrectly classified as negative. In this case, these are the entries of the good pieces class that have been classified as damaged.
- False Positives (FP): is the number of instances that the algorithm incorrectly classified as positive. In this specific case, these are the damaged pieces class entries that have been classified as good.
- True Positives (TP): is the number of instances that the algorithm correctly classified as positive. In this case these are the good pieces class entries that have been classified as good.

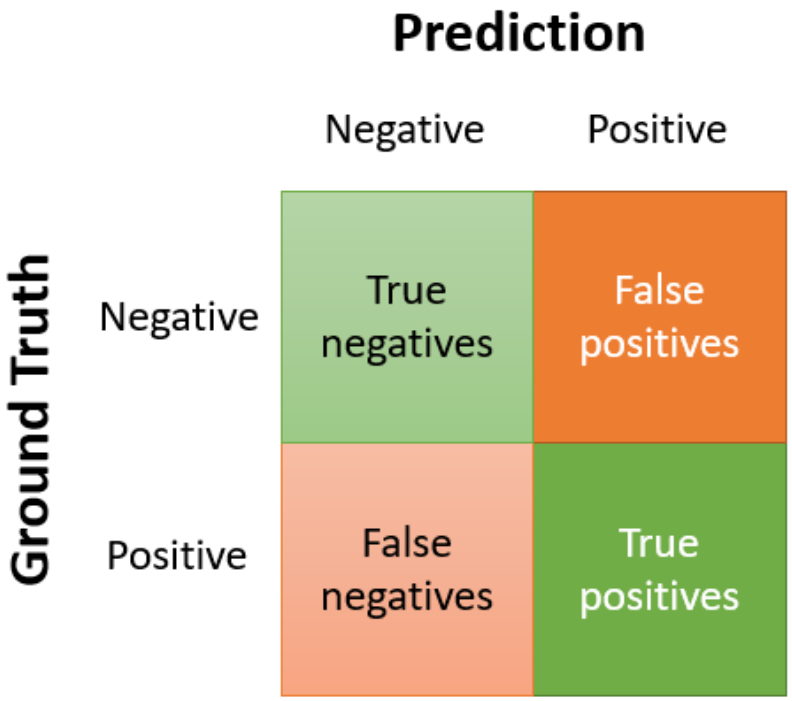

Figure 1: Confusion Matrix [4]

From here we consider the following statistical parameters [6]:

1. **Accuracy**: calculates the proportion of correct predictions.

$$accuracy := \frac{TN + TP}{TN + TP + FN + FP}$$

2. **Precision**: refers to the number of true positives divided by the total number of positive predictions.

$$precision := \frac{TP}{TP + FP}$$

3. **Recall**: is the proportion of positive inputs that are correctly classified by the model.

$$recall := \frac{TP}{TP + FN}$$

As in this specific problem we are interesting in choosing the right parameters to be sure the piece we are due to produce would be good, the most important statistical parameter is the precision.

Finally, we need to measure the quality of every possible combination of classifiers and parameters. In order to avoid overfitting problems we will use the Cross Validation procedure, which is explained in Subsection 4.2.

4.2 Cross Validation

Usually, the hyperparameters for classifiers, such as the c value for an SVM, are fitted measuring the performance on the training set. Quite often this produce some overfitting on the training set. This is because the parameters can be adjusted until the classifier performs optimally on this set, which can cause the knowledge of the training set to filter into the model and the evaluation metrics to lose their generalisation performance.

To address this issue, a validation set can be created by holding out a portion of the training set. The model is trained on the training set and evaluated on the validation set. If the results are satisfactory, the final evaluation is performed on a third subset, the test set.

However, partitioning the available data into three sets drastically reduces the number of data available for model learning, and the results can depend on a particular random choice for the pair of train and validation sets. To deal with this problem, a procedure called cross-validation (CV) is used.

There are different ways of CVs, in this work a variation of the k-fold CV has been used. The k-fold CV approach involves splitting the dataset into training and test sets. Then it makes another partition to the training set into k smaller sets, called folds, with equal sizes.

For each of the k-folds, a model is trained using the other $k-1$ of the folds as training data and the selected fold as validation (often also referred to as a test). Each of these iterations are called splits. The resulting model is then evaluated on the remaining part of the data, using it as a test set to compute a performance measure such as accuracy [2]. To better understand the k-fold process, see Figure 2.

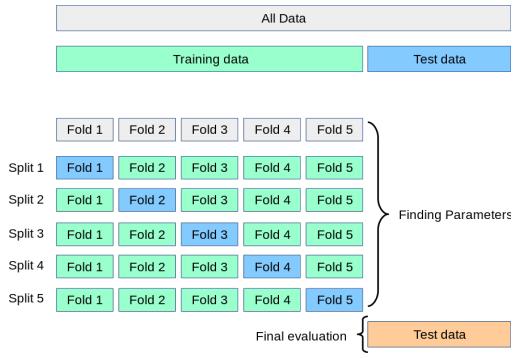

Figure 2: K-fold Cross Validation process [2].

In this work, as the parameters are not adjusted within the CV, this process is used for a better evaluation of the performance of the chosen parameters.

Instead of initially dividing the dataset into test and train, we apply the k-folds directly on the complete set, i.e. we divide the dataset into k folds. For each split, the three statistical parameters are measured on the fold that corresponds to the training data. The average of the values computed in the loop is the performance measure reported by k-fold CV for the selected parameters.

For the case of the confusion matrix of the k-fold process it is the sum of each confusion matrix obtained in each of the splits.

4.3 Data analysis

We have a dataset with 4889 entries and 27 variables, where 4600 entries are of good pieces and 289 are of defective pieces. The *Good_Bad_Piece* variable indicates whether the piece was good or defective. The dataset contains numerical and categorical values, which can be in text form. Excluding the type of material, all variables are numerical, including temperature, heating time and humidity. All the variables available in the dataset are explained in Appendix 2.

To enable machine learning algorithms to process the categorical variable, the material type must be converted from text to numerical values. We do this using dummy variables. Dummy variables involve creating a binary column for each unique value in the categorical variable being coded. The column corresponding to the value present in each record is marked with a 1, while the other columns are marked with a 0.

In this case, the variables *category_abs* and *category_ps* have been created, which are the two types of material for which data is available. Additionally, we have normalised the variables between 0 and 1, as the procedures may give more weight to certain variables, such as temperatures, due to their larger dispersion.

The studies of both methods are totally different. While the kNN algorithm works worse when the number of variables is large due to the curse of dimensionality (when the dimension is increased, under rather general conditions the probability of finding a point nearby to the point to be classified tends to zero) the SVM works nicely with high dimensions and does not need to reduce the number of variables used.

Furthermore, we are working with unbalanced data, because the amount of data in one class is much higher than in the other one. In this case, the percentage of bad pieces is around 6% of the data; this means that if we classify all the pieces as good, we would achieve an accuracy of 94%. Obviously, the classifiers' efficiency could been affected by this unbalancing.

For this reason, we propose two ways to solve this problem. First of all, we are going to try to raise the threshold for the decision of classify a data as good, making easiest to say a piece is defective.

These algorithms work calculating the probability for the SVM or the proportion of instances of class 0 among the k nearest neighbors, good piece, or to the class 1, bad piece. The SVM (respectively the kNN) decides that the piece is good when the probability of be good is larger than 0.5 (respectively the proportion of one class is equal or higher than 0.5). The main idea is to replace 0.5 by a higher value in order to reduce dominance.

If we observe that the imbalance is still large in the results of the algorithm, we can apply oversampling or under sampling techniques. These techniques multiply the instances of the minority class or remove some instances of the majority class respectively.

4.3.1 k-Nearest Neighbors

With the kNN algorithm the largest the dimension (more variables are used) the worst the results, so we are going to study which are the variables that have more relevance in this problem and we are going to select the better ones to apply the algorithm.

A scikit learn function called *mutual_info_classif* has been used to measure the dependency between each variable and the target variable (*Good_Bad_Piece*).

The mutual information between two random variables is a non-negative value that measures the information between two variables. It is equal to zero only if the variables are independent and higher values indicate higher dependence. The values can be between zero and infinity.

The function is based on non-parametric methods that estimate entropy from distances to the kNN (note that this is different from the classification made above with kNN). See [1] for more information.

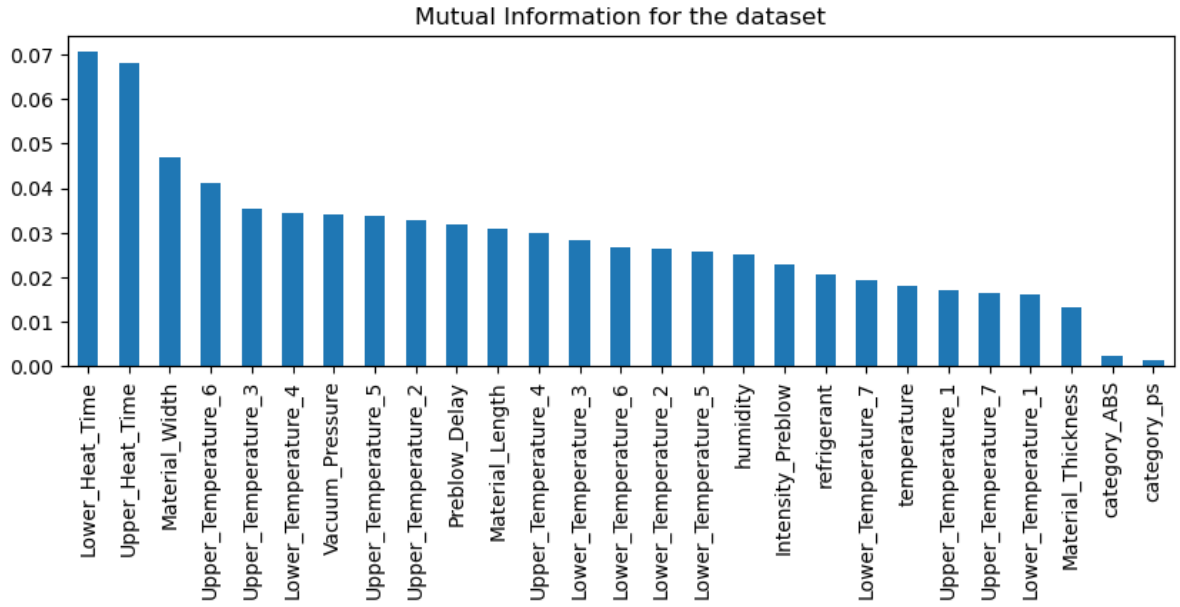

Figure 3: Variables available in the data set ordered by their relevance

We can see that the characteristics that have more information for this case are undoubtedly the lower and upper heating times. As we must find a balance between the amount of information we store and the amount of variables we use, taking into account that the information between the target and the two most important variables is around 0.07, we will take the characteristics that have 0.023 or more (1/3 of the information that the first two characteristics have). With this criterion we find that the number of characteristics to use is 18, since the nineteenth characteristic has 0.02 of the information.

For the issue of sample imbalance the two ways considered above become necessary, raising the threshold and using oversampling techniques.

There are two reasons for increasing the threshold. Firstly, if the threshold remains at 0.5, the algorithm will classify practically all the instances as good pieces due to the high proportion of good pieces in our data. Secondly, to prevent the production of defective pieces, the algorithm should only classify a piece as good if it is highly confident in its classification.

The threshold will be set at 0.95, because we obtained the same results using 0.95 and higher rates, but the algorithm still remains biased due to the low proportion of defective pieces (the precision is around 0.94). To address this issue, the oversample function is utilized to balance the number of entries for each class within the dataset, obtaining the same number of defective pieces as the number of good pieces in the training set.

After these initial adjustments, we continue with the study of the optimal number of neighbours to obtain the best classifier with this algorithm.

With this aim in mind, in Figure 4 we represent the three statistical parameters versus the number of neighbours and make a decision based first of all in the precision and if some values of k have equal or very similar precision we select the best by using the other parameters.

It is important to clarify that we are using odd number of neighbours due to the fact that we are choosing between two groups and we want to avoid ties (although after increasing the threshold for decisions this fact should not be decisive).

Figure 4 shows the precision, accuracy and recall of the CV procedure versus the number of neighbours used to create the algorithm.

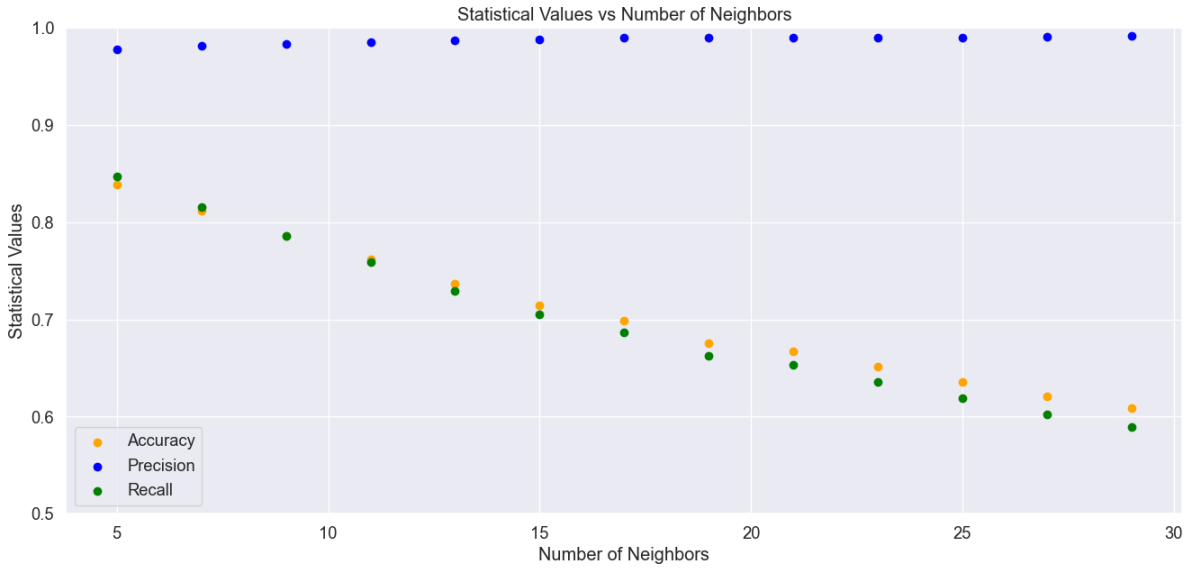

Figure 4: Statistical parameters for the kNN algorithm

Looking at Figure 4 we can see that for a number of neighbours of 17 to 29 the precision is almost identical and close to 100%. The other two parameters decrease as the number of neighbours increases. So we choose $k = 17$ as the number of neighbours with the best parameters.

4.3.2 Support Vector Machine

As mentioned above the SVM algorithm still works well with high dimensions so we will use all the variables available.

For the issue of sample imbalance we will only raise the threshold to 0.95, because, as well as before, we obtained the same results using 0.95 than higher rates.

For this method we have the possibility of using four type of kernels, already discussed in page 29: the linear, the polynomial, the radial and the sigmoid kernels. In order to determine the best SVM algorithm for this specific case we are studying eight possibilities for the regulation parameter for the four kernels.

Figures 5, 6, 7 and 8 show the precision, accuracy and recall of the CV procedure versus the regularisation parameter for each of the selected kernels. The regularisation parameter is the power of ten of the minus number on the x-axis, i.e. where the x-axis marks a one we have a regularisation parameter of $c =$ 1e-1.

In Figure 5 we can see how the precision increases slowly until the regularisation parameter with a value of 1e-3. Moreover, for this same value we can see a increase of the other two statistical parameters. This is why we choose 1e-3 as the regularisation parameter.

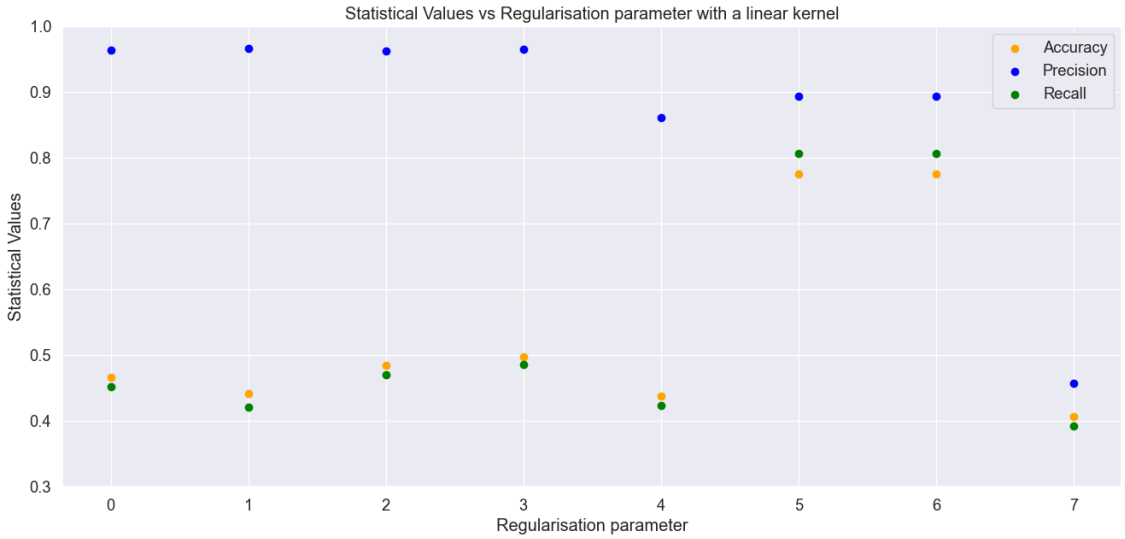

Figure 5: Statistical parameters for SVM with linear kernel

In Figure 6 we observe a trend opposite to the one in Figure 5. It is noticeable that the precision tends to decrease as we decrease the regularisation parameter. Among the first three values, 1, 1e-1 and 1e-2, we can see that in addition to the precision, the accuracy and recall are higher for the first one. For this reason, we choose the value 1 as the regularisation parameter with the best statistical parameters for the polynomial kernel (remember that by default, the degree of the polynomial of the kernel is 3).

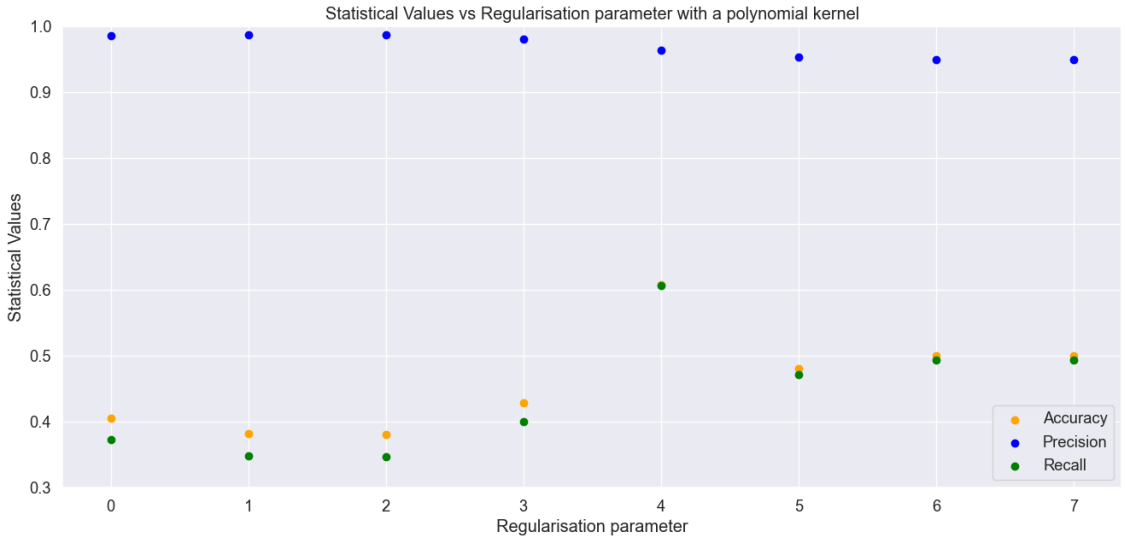

Figure 6: Statistical parameters for SVM with polynomial kernel

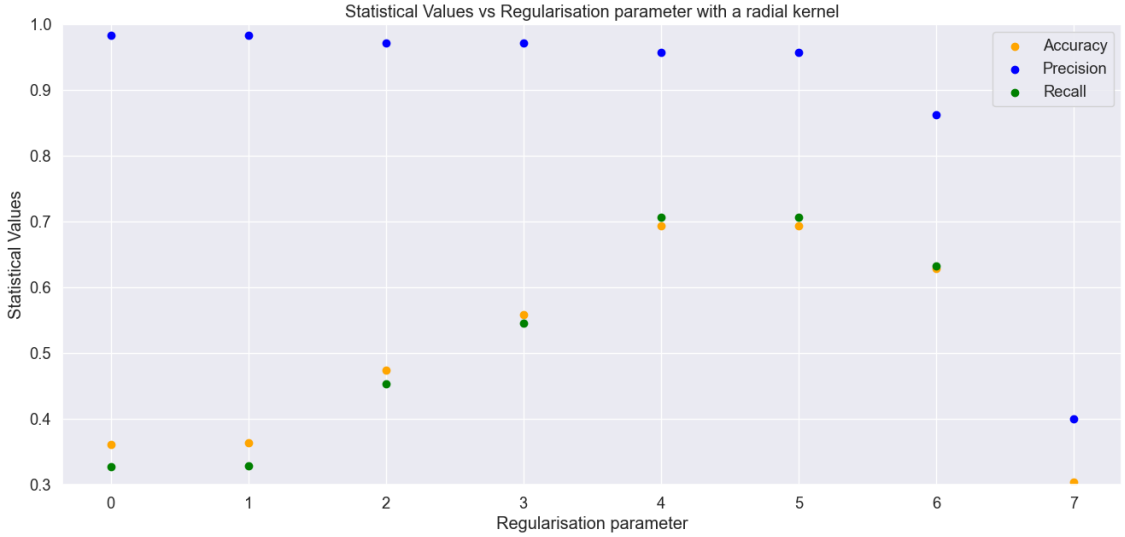

Figure 7: Statistical parameters for SVM with radial kernel

In Figure 7 we can observe that we have obtained almost the same precision for the two first regulation parameters 1 and 1e-1. Also, looking the rest of the parameters we realise that they are basically the same in both.

As the main objective of the SVM algorithm is to maximize the margin between classes, due to the fact that the two regulation parameters have obtained the same parameters, we choose the one that has a regulation parameter of 1e-1 because is the one with higher margin.

Finally, in Figure 8 we observe a clear trend of increasing precision as we decrease the regularisation parameter. However, for the value 1e-4 we observe a noticeable decrease in the precision although the accuracy and the recall have slightly grown. For the same reason as in the previous graph, we will choose the regularisation parameter equal to 1e-3 as the parameter with the best statistical values.

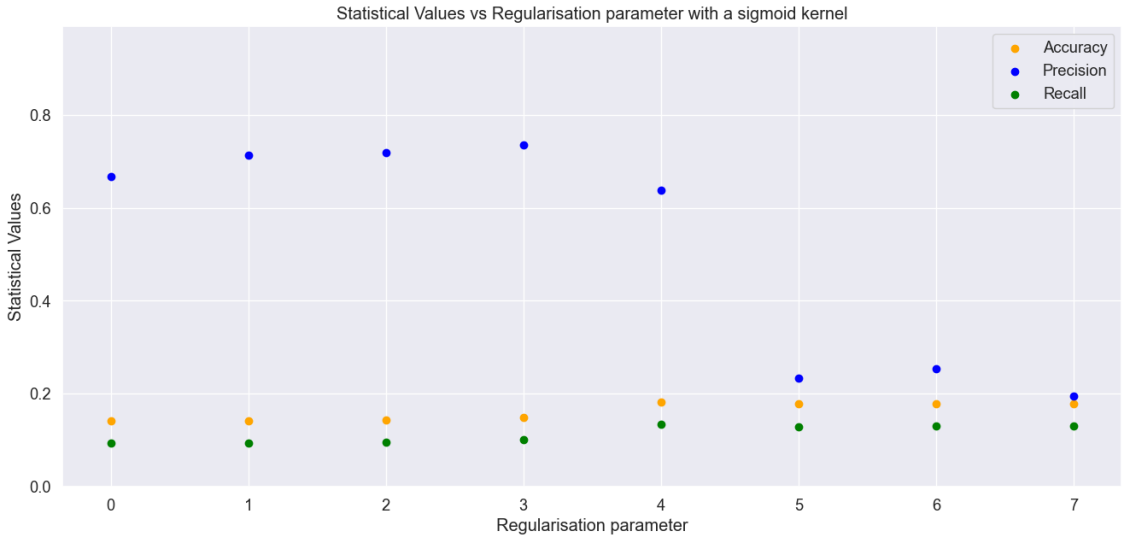

Figure 8: Statistical parameters for SVM with sigmoid kernel

Now we are going to compare the statistical parameters for the kernels of them selecting the best regulation parameter for each one.

Kernel	***C***	***Accuracy***	***Precision***	***Recall***
lineal	1e-3	**0.4971**	0.9656	**0.4857**
poly.	1	0.4052	**0.9865**	0.3728
radial	1e-1	0.3637	0.9835	0.3294
sigmoid	1e-3	0.1487	0.7352	0.1012

Table 1: Comparison of statistical parameters between the different kernels of the SVM algorithm

Focusing on the precision value, we observe that the radial and polynomial kernel obtain similar and higher values. But, as the accuracy and the recall are higher for the polynomial kernel we conclude that the polynomial kernel is the best option for the SVM algorithm on the dataset at hand.

4.4 The results

Once both methods have been studied separately and the best intrinsic parameters for each have been selected, it remains to compare both methods to determine which one most accurately predicts our data.

To compare the two methods we will first use the confusion matrix obtained for each one. Then, Table 2 shows with the three statistical criteria used in the selection of the parameters to consolidate the observations.

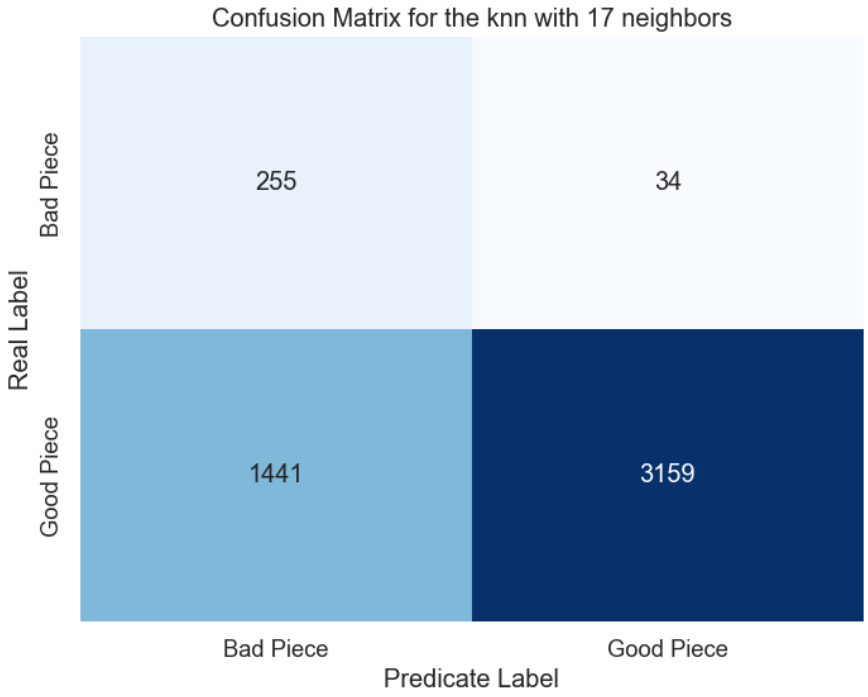

Figure 9: Confusion matrix for the kNN algorithm with 17 neighbours

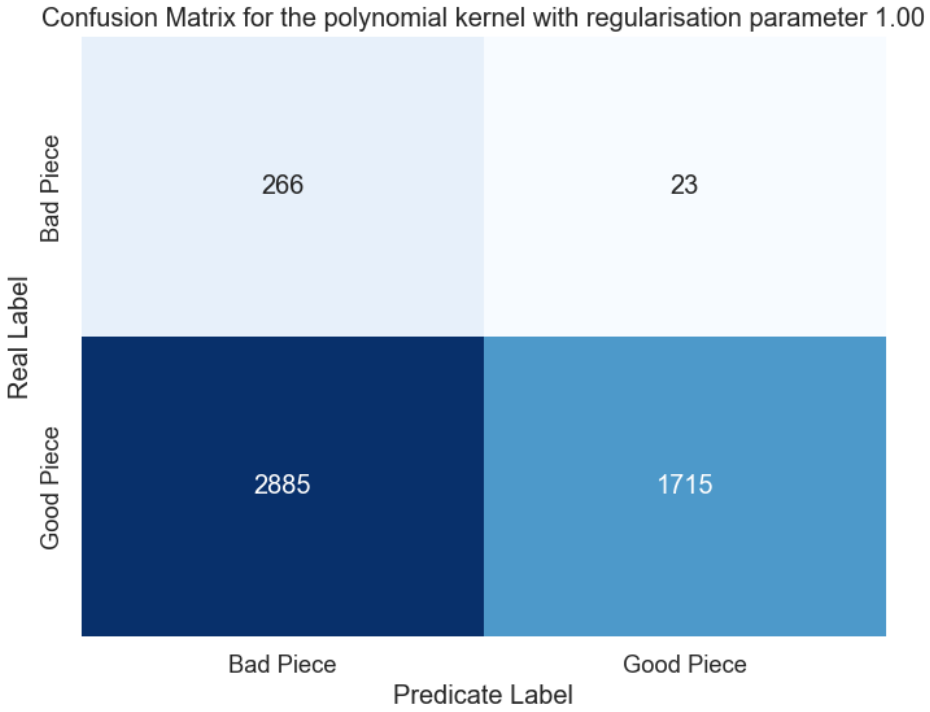

Figure 10: Confusion matrix for the SVM algorithm with polynomial kernel and a regularisation parameter $c = 1$

The first thing we notice in these figures is that the total number of classified data is the same. Although it has been mentioned that for the implementation of the kNN algorithm it has been necessary to oversample the dataset, this has been done in the training set and not in the test set, where the statistical parameters have been calculated.

Leaving this aside, another notable aspect of these graphs is the rate of falsely classified good pieces, i.e. the number of pieces that were classified as good but were actually bad (top right box in both graphs). It can be observed that this number is 34 for the kNN algorithm and only 23 for the SVM. However, the kNN has almost classify 2 times more good pieces correctly than the SVM.

After the initial impression, we will compile the three statistical parameters used in this work for the kNN and the SVM with polynomial kernel in the following table.

Algorithm	***Accuracy***	***Precision***	***Recall***
kNN	**0.6983**	**0.9996**	**0.6870**
SVM poly.	0.4052	0.9865	0.3728

Table 2: Comparison of statistical parameters between the two algorithms

Table 2 confirms that kNN has a higher percentage of true positives compared to SVM. However, upon examining the remaining parameters, it becomes apparent that they are considerably higher for the kNN algorithm. As a result, we have selected this algorithm as the optimal classifier for this specific problem with these training data.

This algorithm can be implemented to reduce the number of defective pieces produced by the moulding machines. When the operator enters the parameter values: the different temperatures, the heating times or the properties of the plastic used (the humidity, for example, is measured directly by the machine from the ambient) this algorithm makes a quick classification and informs the operator if with those characteristics the piece will be good or defective. If the piece is defective, the operator must adjust the parameters again to increase the chance of obtaining good piece.

5 Conclusions

We have shown that the kNN algorithm has universal consistency for continuous distributions, which means that if we increase the number of observations the algorithm will improve the classification rates approaching the optimal Bayes error. This is very important as it ensures that as more pieces are made and the database records more entries (which will be mostly good pieces as we are trying not to make damaged pieces) the model is giving us more accurate information on whether the values chosen for the parameters are good or need to be adjusted.

In addition, we have seen that the SVM algorithm is not universally consistent for all possible distributions and kernels. However, we have shown that using universal kernels on distributions with compact support the algorithm is universally consistent.

In the practical implementation, the Scikit-learn library only implements the lineal, polynomial, radial and sigmoid kernels. These four kernels are universal kernels, so it would still be true that the more time passes and the more observations we have, the more correct classifications the SVM will make.

On the other hand, due to the high computational cost of increasing the number of inputs for the SVM we would have to make sure that it classifies well and fast for a limited number of observations.

By applying both models to a data set with variables measured by sensors on a real moulding machine we have obtained interesting results.

Following the objective of minimising raw material waste and reducing costs and energy, it is obvious that what we are trying to achieve is a reliable algorithm that tells us that the piece is going to be ready for sale.

In the case of the kNN algorithm, its precision is 0.9996 and for the SVM it is 0.9865. Both precisions are very high but the proportions of false negative is almost two times higher for the SVM than for the kNN . Additionally, the other statistical parameters were much higher for the kNN.

With respect to improvements or future work on the practical side, there are several ways that can be explored to try to improve the results.

In addition to the classification algorithms used, kNN and SVM, there are many others such as boosting, random forest or trees whose use could be interesting in certain parts of the process followed in this project.

First, instead of using the *mutual_info_classif* function, which measures the entropy dependence between the features and the dependent variable, we can use the powerful *XGBoost* algorithm (Extreme Gradient Boost). This supervised algorithm is based on decision trees. As the model is trained, XGBoost evaluates the importance of each feature to the classification problem. You can then extract the importance of the features and use that information to select the most relevant features. This algorithm is useful when you are working with complex models and large datasets.

Another approach can be to combine the two methods, for example you could use XGBoost to obtain an initial list of important features and then apply mutual information to refine the selection.

Secondly, you could implement the function *GridSearchCV*. This is a CV technique that when run searches through the different parameters you enter in the parameter grid and extracts the best values and combinations of parameters for the problem you present to it. In this case it could be used to calculate the best number of neighbours in the kNN (among a larger number of possibilities), to find the regularisation parameter and the kernel in the SVM or even to determine the number of relevant features that make the model perform better.

On the other hand, to keep as much information as possible by decreasing the dimension for the kNN algorithm, the Principal Components Analysis can be used, which creates new variables by making linear combinations of the original ones with minimal loss of information.

Finally, it could be very interesting to create an algorithm that, based on the environmental data and the characteristics of the plastic to be used, would indicate the best values for the machine parameters and thus increase the proportion of pieces suitable for sale.

References

[1] *sklearn.feature_selection.mutual_info_classif*. Jan. 2024. URL: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html.

[2] *Cross-validation: evaluating estimator performance.Visited on 01/28/2024*. Jan. 2024. URL: https://scikit-learn.org/stable/modules/cross_validation.html.

[3] Luc Devroye, László Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Vol. 31. Springer Science & Business Media, Jan. 1996. ISBN: 978-1-4612-6877-2. DOI: 10.1007/978-1-4612-0711-5.

[4] *Disposición de la matriz de confusión*. Dec. 2023. URL: https://interactivechaos.com/es/manual/tutorial-de-machine-learning/disposicion-de-la-matriz-de-confusion.

[5] Evelyn Fix and Joseph Hodges. *Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties*. Vol. 57. 3. Wiley, International Statistical Institute (ISI), 1989, pp. 238–247. URL: http://www.jstor.org/stable/1403797.

[6] Aurélien Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and Tensorflow, 2nd edition*. O'Reilly, 2019. ISBN: 9781492032649.

[7] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. "Support Vector Machines and Flexible Discriminants". In: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer New York, 2009, pp. 417–458. ISBN: 978-0-387-84858-7. DOI: 10.1007/978-0-387-84858-7_12. URL: https://doi.org/10.1007/978-0-387-84858-7_12.

[8] Jeremie Houssineau, Neil K. Chada, and Emmanuel Delande. *Elements of asymptotic theory with outer probability measures*. 2020. arXiv: 1908.04331 [math.ST]. URL: https://doi.org/10.48550/arXiv.1908.04331.

[9] *sklearn.svm.SVC. Visited on 01/28/2024*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

[10] Ingo Steinwart. "On the influence of the kernel on the consistency of support vector machines". In: *Journal of machine learning research* 2.Nov (Jan. 2001), pp. 67–93. DOI: 10.1162/153244302760185252.

[11] Ingo Steinwart. "Support Vector Machines are Universally Consistent". In: *Journal of Complexity* 18.3 (2002), pp. 768–791. ISSN: 0885-064X. DOI: https://doi.org/10.1006/jcom.2002.0642. URL: https://www.sciencedirect.com/science/article/pii/S0885064X02906424.

[12] Charles J. Stone. "Consistent Nonparametric Regression". In: *The Annals of Statistics* 5.4 (1977), pp. 595–620. ISSN: 00905364. URL: http://www.jstor.org/stable/2958783 (visited on 02/13/2024).

[13] Enrique J. Carmona Suárez. "Tutorial sobre máquinas de vectores soporte (sVM)". In: *Tutorial sobre Máquinas de Vectores Soporte (SVM)* 1 (Nov. 2016), pp. 1–12.

[14] *Support Vector Machine (SVM)*. Jan. 2024. URL: https://es.mathworks.com/discovery/support-vector-machine.html.

[15] Kashvi Taunk et al. *A Brief Review of Nearest Neighbor Algorithm for Learning and Classification*. 2019, pp. 1255–1260. DOI: 10.1109/ICCS45141.2019.9065747.

[16] Aad W. van der Vaart and Jon A. Wellner. "Outer Integrals and Measurable Majorants". In: *Weak Convergence and Empirical Processes: With Applications to Statistics*. Springer New York, 1996, pp. 6–15. ISBN: 978-1-4757-2545-2. DOI: 10.1007/978-1-4757-2545-2_2. URL: https://doi.org/10.1007/978-1-4757-2545-2_2.

[17] *Why Python for Machine Learning?* Jan. 2024. URL: https://pythonbasics.org/why-python-for-machine-learning/.

Appendixes

Appendix 1

In this appendix we employ the condition presented in Section 3.1.

Lemma .0.1 (Cover and Hart)**.** *(See [3], page 579)*
Let μ be a probability measure on $\mathbb{R}^d$, and define its support set by

$$A = Supp(\mu) = \{x : \text{ for all } r > 0, \mu(S^d_{x,r}) > 0\}$$

Then $\mu(A) = 1$.

Proof. By the definition of A,

$$A^c = \{x : \mu(S^d_{x,r_x}) = 0 \text{ for some } r_x > 0\}.$$

Let Q denote the set of vectors in $\mathbb{R}^d$ with rational components. Then for each $x \in A^c$, there exists $y_x \in Q$ with $||x - y_x|| \leq r_x/3$. This implies $S^d_{y_x,r_x/2} \subset S^d_{x,r_x}$. Therefore, $\mu(S^d_{x,r_x/2}) = 0$, and

$$A^c \subset \bigcup_{x \in A^c} S^d_{y_x,r_x/2}$$

The right hand side of this expression is a union of countably many sets of zero measure, and therefore
$\mu(A^c) = 1$.

□

Lemma .0.2. *(See [3], pages 63-64)*
Let us assume that $\lim_{n\to\infty} \frac{k}{n} = 0$. If $x \in supp(\mu)$, then $||X_{(k)}(x) - x|| \to 0$ with probability one. If X is independent of the data and has probability measure μ, then $||X_{(k)}(X) - X|| \to 0$ whenever $k/n \to 0$.

Proof. Take $\epsilon > 0$. Let $x \in supp(\mu)$. Then $\mu(S^d_{x,\epsilon}) > 0$. Observe that $||X_{(k)}(x) - x|| > \epsilon$ if and only if

$$\frac{1}{n}\sum_{i=1}^{n} I_{\{X_i \in S^d_{x,\epsilon}\}} < \frac{k}{n}.$$

By the strong law of large numbers, the left-handed side converges to $\mu(Sx, \epsilon) > 0$ with probability one, while, by assumption, the right-hand side tends to zero. Therefore, $||X_{(k)}(x) - x|| \to 0$ with probability one.

The second statement follows from the previous argument as well. First note that by Lemma .0.1, $P\{X \in supp(\mu)\} = 1$, therefore for every $\epsilon > 0$,

$$P\{||X_{(k)}(X) - X|| > \epsilon\} = E\left[I_{\{X \in supp(\mu)\}} P\{||X_{(k)}(X) - X|| > \epsilon | X\}\right],$$

which converges to zero by the dominated convergence theorem, proving convergence in probability. If k does not change with n, then $||X_{(k)}(X) - X||$ is monotone decreasing for $n \geq k$; therefore, it converges with probability one as well. If $k = k_n$ is allowed to grow with n such that $k/n \to 0$, then using the notation $X_{(k_n,n)}(X) = X_{(k)}$, we see by a similar argument that the sequence of monotone decreasing random variables

$$\sup_{m \geq n} ||X_{(k_m,m)}(X) - X|| \geq ||X_{(k_n,n)}(X) - X||$$

converges to zero in probability, and therefore, with probability one as well.

□

Appendix 2

This appendix explains the information contained in the dataset variables used for both models, the kNN and the SVM.

- *Preblow_Delay*: measures the time from the start of the cycle until pre-blowing is applied.
- *Lower_Heat_Time*: measures the time of heat application in the lower area of the material.
- *Upper_Heat_Time*: measures the time of heat application in the upper area of the material.
- *Material_Width*: measures the width of the plastic sheeting used.
- *Material_Thickness*: measures the thickness of the plastic sheeting used.
- *Material_Length*: measures the length of the plastic sheeting used.
- *Intensity_Preblow*: measures the intensity of the pre-blow air. Proper pre-blowing can help to preform the material before the main blowing.
- *Lower_Temperature_*1 a *Lower_Temperature_*7: these variables measure the temperatures at different sections or points on the bottom of the mould during the moulding process.
- *Upper_Temperature_*1 a *Upper_Temperature_*7: these variables measure the temperatures at different sections or points on the top of the mould during the moulding process.
- *Vacuum_Pressure*: measures the vacuum pressure applied during the blow moulding process. Vacuum helps to remove air bubbles and improve the quality of the moulded part.
- *Good_Bad_Piece*: indicates whether the moulded piece is acceptable (good) or defective (bad). It is based on predefined quality criteria.
- *humidity*, *refrigerant* and *temperature*: these variables measure the environmental conditions of the work area, such as the relative humidity, the temperature of the refrigerant used in the cooling system and the ambient temperature.
- *category_ABS* and *category_ps*: indicate the type of plastic used in the moulding process, such as ABS or polystyrene. If in the *category_ABS* variable there is a 1 in an entry then it means that ABS is being used in the production of the part.