POLITECNICO DI TORINO

DEPARTMENT OF MANAGEMENT AND PRODUCTION ENGINEERING

DEGREE COURSE

Management Engineering

THESIS WORK

ECONOMICS

USE OF ARTIFICIAL INTELLIGENCE FOR THE ANALYSIS OF FINANCIAL MARKETS – NEURAL NETWORKS AND THE FORECASTING OF STOCK PRICES

Supervisor: Prof. Luigi Benfratello

Candidate: Andrea Lopetuso

A.Y. 2022/2023

Contents

1. DATA MINING

2.1 WHAT IS DATA MINING?

Data mining is the activity aimed at automatically recognizing and extracting information from large structured databases.
Data mining is also defined as the process of discovering knowledge from databases, which results in the sequence of the following steps:

- **data cleaning**: the data supporting the analysis must have the highest degree of correctness, therefore it is necessary to eliminate anomalous data and correct errors where possible;

- **data integration**: data can come from several different sources, which must be traced back to a common and integrated model;

- **data selection**: not all data are necessary for the analysis; in this step, only the data deemed useful are selected;

- **data transformation**: the data are processed in such a way as to make them appropriate for the mining activity, reorganizing or aggregating them;

- **data mining**: it is the real analysis process usually carried out through complex functions that scan the database looking for notable conditions;

- **pattern evaluation**: the mining functions report everything that is considered significant from the point of view of the analysis rules, but not all the conditions detected are always of real interest. In this phase, the set of conditions detected is reduced to only interesting information;

- **presentation of knowledge**: in this last phase, no less important than the previous ones, the information extracted automatically from the system is presented to the user

using graphic representations that give an idea of the actual load of knowledge brought by the extracted information at a glance.

The data mining system can be considered an evolution of the data analysis system, in fact, we note that the first phases of data mining are nothing more than phases of building a database.

2.2 ARCHITECTURE OF A DATA MINING

The architecture of data mining systems is developed in the following components:

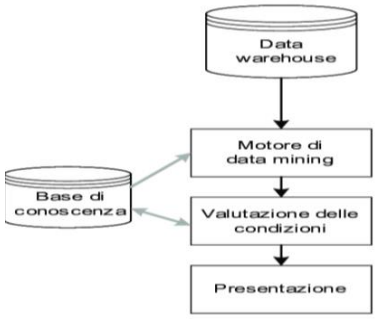


Figure 1 Architecture of a data mining system

- DATA WAREHOUSE: analysis database.

- KNOWLEDGE BASE: it contains the set of rules and knowledge of the system, which will be used both to guide the research and to filter the results by evaluating the actual interest of the patterns detected by the analyzes.

- DATA MINING ENGINE: a set of data analysis functions. The functions integrate techniques developed in disciplines such as statistics, artificial intelligence, and neural networks as well as processing optimization techniques derived from branches of information technology such as database technology.

- CONDITION EVALUATION SYSTEM: this component interacts with the mining modules to focus the research on the interesting conditions, using the information stored in the knowledge base as filter conditions.

- PRESENTATION SYSTEM: this is the interface with which the user can specify which mining activity to undertake.

2.3 MINING FUNCTIONS

The data mining activity can be carried out through different functions, which focus on finding particular conditions in the data, or patterns. In general, mining activities can be divided into two macro classes :

- Descriptive mining, through which information relating to the general properties of the data is extracted;

- Predictive mining, which by analyzing the data present, determines general rules and creates models to be used for predicting trends in the future.

Each function allows you to search for a type of information, to build a particular prediction model. In the same system, the same function can be processed through different algorithms, oriented on particular aspects of the information to be extracted.

2.4 TASKS OF DATA MINING SYSTEMS

The main tasks for data mining are:
classification: identification of classes (based on certain rules) and of the set of elements united by correspondence to the same;
clustering (or segmentation): identification of groups of homogeneous elements, which, unlike what happens in the classification, are based on hidden rules until the moment of their discovery;
association: discovery of random but recurring links that can be extrapolated from the data contained in a database, aimed for example at detecting anomalies;
regression: similar to classification, from which it differs in that the variables (ie the rules of belonging to a class), of a categorical type in the classifications, in the case of regressions can instead assume a large or infinite number of values;

time series (or historical series): these are complex regressions that incorporate time variables (dates, changes in interest rates, etc.) and therefore particularly useful for predictive purposes;
sequence discovery: takes up the concept of association but applies the sequential correlation factor, i.e. detecting when A (for example, purchase of a toy) follows B (purchase in a certain subsequent time frame of an option for that toy).

2.5 AREAS OF APPLICATION

The use of data mining in the enterprise is increasing the advantages, so we find more and more often data mining applications in the field of:

- **Marketing**, in which the main applications of data mining concern:

clustering (database marketing): identification of types of buyers sharing purchasing habits and socio-demographic characteristics ;
customer retriever: analyzing the behavior of a brand's customers it becomes predictable to identify those at risk of abandonment, and therefore adopt appropriate strategies to prevent it;
market basket analysis: which products or services are usually bought together? With the analysis of associations, it is possible to understand this.

- **Economic and financial**, in which the main data mining applications concern:

fraud detection: by analyzing, for example, the use of credit cards, it is possible to identify anomalies and finally trace fraudulent behaviors;
forecasts on stock index trends ;
analysis on the interactions between financial markets: effective for predicting the influence of the general trend of the markets on the single market.

We can find data mining applications in various other fields, from medicine, to technology and in general in all industrial fields.

2.6 FINAL CONSIDERATIONS

The answer to statistical analysis questions in any field can be contained in the databases. The problem is that it is unintelligibly so, nobody today, could handle big data in acceptable times, that is, the infinite and heterogeneous amounts of data contained in the databases**.**

This is where data mining comes into play, which manages to find associations, anomalies, and recurring patterns, therefore ultimately information, within them. But above all, thanks to the high parallelism of the computing resources used (alongside highly specialized operators) it manages to do so with an efficiency that far exceeds that of a human operator who analyzes them manually.

Data mining , in short, ensures that starting from encrypted information, disseminated without apparent order in a database, we arrive at a knowledge exploitable for various purposes **.**

2. BIG DATA

3.1 WHAT ARE BIG DATA?

Big data is defined as collections of data sets whose volume, speed or variety is so large that it is difficult to store, manage, process and analyze data using traditional databases and data processing tools. In recent years, there has been an exponential growth in both structured (data stored in databases, organized in rigid tables and patterns) and unstructured data (data stored without any schema. An example may be files containing text to narrative character) generated by information technologies, industry, healthcare, Internet of Things and other systems.

Big Data offers the ability to harness the power of data to make applications intelligent. Big Data analytics deals with the collection, storage, processing and analysis of this data on a large scale.

The term big data is an abstract concept, there is no exact and exclusive definition. First of all, big data can be negatively defined as the set of data that cannot be detected, obtained, managed and analyzed with traditional information technologies and databases. Some researchers have tried to explain the phenomenon, so there are several definitions that differ for some elements:

- In 2010 Apache Hadoop (framework that supports distributed applications with high data access under a free license): "Data set that cannot be captured, acquired and managed by general computers within a specific scope."
- In 2011 McKinsey and Company (multinational business consultancy) defined the phenomenon as the new frontier of innovation, competition and productivity. According to the company, big data is in fact that set of data that cannot be acquired and managed by classic databases. The definition shows that the volume of data is not the only criterion to be taken into consideration, two other key characteristics are the ever increasing flows of data and the management that will no longer be able to use traditional technological databases.
- Gartner Inc. (a world-leading multinational corporation in strategic consulting, research and analysis in the field of Information Technology) has identified in big data what are called the 4Vs:

1. Volume: the term refers to the huge amount of all types of data generated by different sources that cannot be managed by traditional databases, but need to be organized and analyzed.
2. Variety : that is data of different nature that can be structured or not , collected for example: via smartphones, social networks or even through commercial transactions.
3. Velocity : to be used these data require a high speed during the "data transfer" process, in this way the data can be processed almost instantaneously, guaranteeing high functionality.
4. Value: is considered the most important aspect of big data and refers to the process of identifying a high hidden value within a large number of different and rapidly growing data. It is therefore essential to evaluate the truthfulness and quality of the data, so that they can actually generate new value.

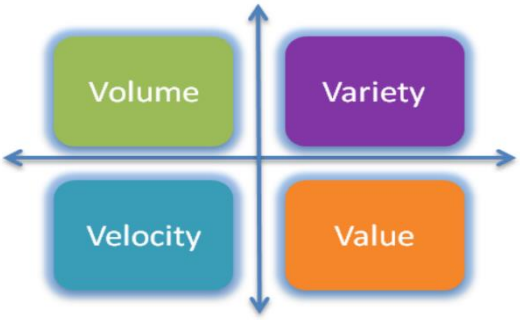


Figure 2Big data "4V"

3.2 BIG DATA MANAGEMENT

Each user generates data by interacting with various devices and through different types of platforms. The large amount of information must necessarily be collected and stored so that it can be used immediately or later. The acquisition takes place through various channels: via API (or application programming interface) used specifically to collect data when accessing a site, with special software for the collection of documents, importing data from pre-existing databases or interpreting and extrapolating the flow of data that passes through the network or through simple cookies by browsing the web. From all these operations an enormous amount of information derives, much of which

is not useful for the purposes of subsequent analysis. It follows that they must be reclaimed or cleaned of all those that do not fall within the format required for processing. At this point , Big Data can be stored and archived in systems capable of storing large datasets . We then move on to analysis and modeling through the development of targeted algorithms and subsequent interpretation so that the information can be useful for corporate performance.

3.3 BIG DATA, ALGORITHMS AND ARTIFICIAL INTELLIGENCE

In the case of Big Data, algorithms are created to allow a study of the flow of data , their analysis but above all their comparison, like a real neuronal network, in order to extract the desired result. It is for this reason that they must be increasingly parametric, multilevel and precise . Algorithms are the very essence of design in computer science and programming and engineers and mathematicians are continually aimed at creating ever more performing models capable of speeding up decision-making and analytical processes. Big Data, in fact, would have no value if it were not possible to analyze them and extrapolate fundamental information for future studies and this can only be done using the algorithms that have now entered arrogance in business decisions, not only with regard to marketing but also for production, maintenance and even for personnel selection.

Here, therefore, that in recent years the term " artificial intelligence " has also imposed itself in the field of marketing. This term means the ability to make machines perform functions that are typical of human intelligence (such as the recognition of language, sounds or images) and to learn from experiences. A part of artificial intelligence is about Machine Learning .

3.4 USES AND OPPORTUNITIES OF BIG DATA

The challenge and the opportunities that big data are varied and including any scientific and industrial field you want.

In our case we will list a series of business features that find considerable advantages from the use of big data:

- Marketing: it is the company place where the gaze is more oriented towards the outside; therefore big data give the possibility of analyzing new markets or detecting purchasing habits.
- Sales: the use of big data can allow, for example, to enrich the after- sales services provided by the company, or to calibrate commercial policies.
- Logistics: the potential offered by big data in the logistics area makes it possible to optimize routes, for example by proposing alternative routes, thus leading to greater punctuality.
- Production / Quality: the application of solutions based on big data increases the efficiency of the production departments: for example, through the analysis of data we can understand when our machinery will need to be serviced, because it is malfunctioning; thus avoiding high shortage costs due to low or no production.

3.5 FINAL CONSIDERATIONS

The data flows generated by users, processes and devices have been considered for some time as a transversal source of information also useful for purposes that transcend the original objective: for example, the tracking data of internet browsing are created with the aim of facilitating the user, allowing him to return to the pages traveled previously, and become only at a later time, and after specific processing, indicators of the person's interests with respect to the content displayed.

Big data therefore indicates sets of data to be used as raw material for new objectives, data generated automatically by devices or by processing, or even produced by users within specific applications.

Small and medium-sized enterprises are also now developing tools to use big data , realizing its importance in helping them improve product quality, expand their business opportunities and accelerate decision-making power.

The normal big data analysis procedure consists first in collecting and transforming particular data into extracted information and subsequently, those collected data are used and implemented with the use of **Machine Learning techniques** in order to predict the

3. MACHINE LEARNING

3.1 WHAT IS MACHINE LEARNING?

Machine learning is an application of artificial intelligence that provides systems with the ability to learn and improve, autonomously from experience without being explicitly programmed.

The learning process begins with observing data such as examples, direct experience or instructions, in order to make better decisions in the future based on the data we provide; so at the base of the Machine Learning concept there is this human learning process. Basically, machine learning learns from examples how to improve its performance for handling new data from the same source. Looking at machine learning from a computer perspective, instead of writing the programming code through which the machine is told step by step what to do, the program is provided with only datasets, which are processed through algorithms, capable of developing its own logic to solve the function, activity and task required.

WE FIND 2 DEFINITIONS:

ARTHUR SAMUEL (1959): "Machine learning is a field of study that gives computers the ability to learn, without being explicitly programmed".

TOM M. MITCHELL (1998): "A computer program is said to learn from experience , by respecting a series of tasks and measured performances , if the performance of a task in measured by improves with the experience ".

Therefore a Machine Learning system (automatic learning) during the training phase learns from examples. It is then able to generalize and manage new data in the same application domain.

Computer learning in Machine Learning is usually divided into three categories:

3.2 SUPERVISED LEARNING

Most machine learning problems use supervised learning. Supervised learning occurs when a working dataset is provided having examples and data composed of input variables (X) and an output variable (Y) and where we use an algorithm to learn the mapping function from input to output.

$$Y = f(X)$$

The goal is to approximate the mapping function (f) so well that when you have new input data (X) you can predict the output variables (Y) for that data.

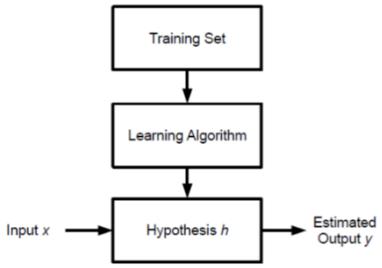


Figure 3supervised learning process

The input data are called training sets and are composed of a number n of examples or data composed of a series of features. The output value is called the target (label).

	caratt. 1	caratt. 2	caratt. 3	risultato
ESEMPI	$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$y^{(1)}$
	$x_1^{(2)}$	$x_2^{(2}$	$x_3^{(2)}$	$y^{(2)}$
	$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$y^{(3)}$
	$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$y^{(4)}$

ETICHETTE

It is called supervised learning because the process of learning an algorithm from the training dataset can be thought of as a teacher who supervises the learning process. Knowing the correct answers, the algorithm iteratively makes predictions on the training data and is corrected "by the teacher". Learning stops when the algorithm reaches an acceptable level of performance. There may be supervised learning problems further grouped into regression and classification problems.

If the output value is discrete, such as belonging or not belonging to a particular class, the problem is Classification. If, on the other hand, the output is a continuous real value in a given range then the problem is Regression.

- Classification: A classification problem is when the output variable is a category, such as red or blue.

- Regression: A regression problem is when the output variable is a real value, such as predicting a price or weight.

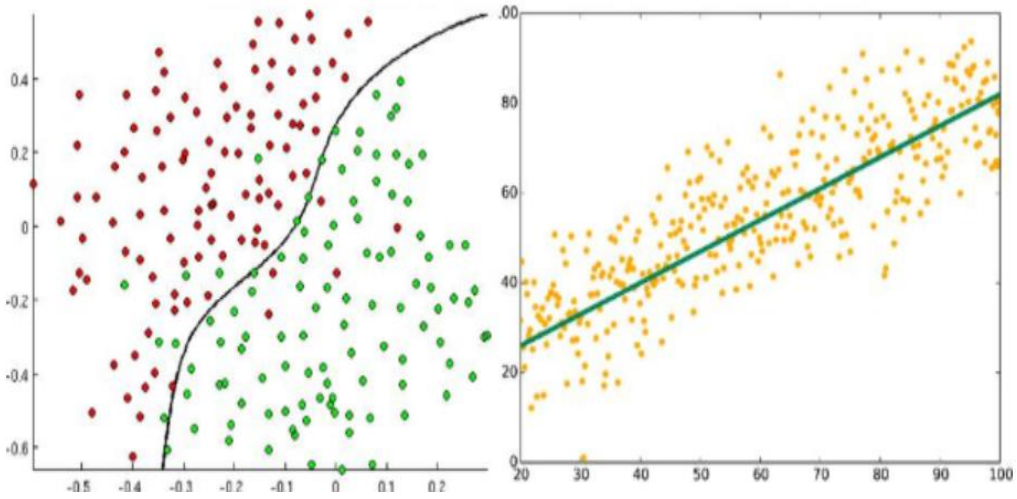


Figure 4 classificazion/regression

3.3 NON-SUPERVISED LEARNING

In unsupervised learning they come examples are provided with related features as input, but no value is provided in output, so you have unlabeled data available. The goal of this family of algorithms is to find structures within this data not labeled. If you want to identify groupings of similar elements, it is a Clustering problem. If, on the other hand, you want to identify the different sources that contributed to the creation of the data, the problem is Blind Source Separation .

So unsupervised learning identifies hidden patterns or intrinsic structures in the data.

the most common unsupervised learning technique. It is used in exploratory data analysis to detect hidden or genetic sequence patterns, market research and object recognition. For example, if a cell phone company wants to optimize the installation locations of its antennas, it can use machine learning to estimate the number of clusters of people connecting to its antennas. A phone can connect to only one antenna at a time, so the team uses clustering algorithms to design the best location for cell phone antennas to optimize signal reception for groups, or clusters, of customers.

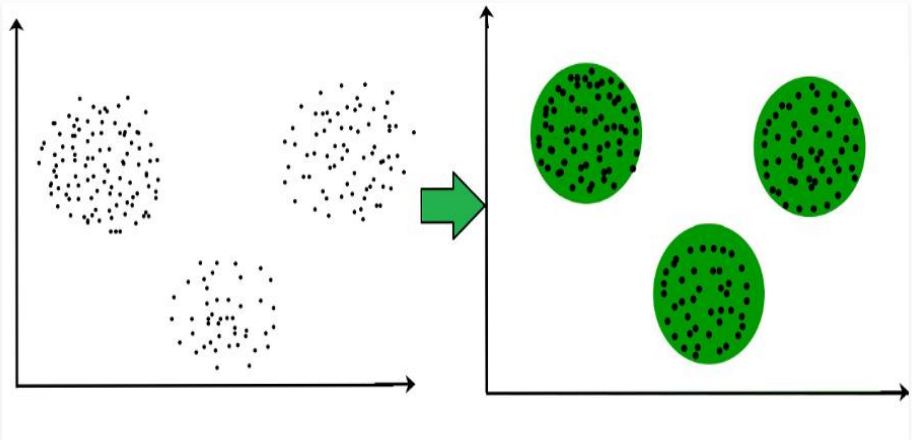


Figure 5 clustering

3.4 LEARNING BY REINFORCEMENT

Reinforcement learning is a machine learning technique that aims to create autonomous agents able to choose actions to be taken to achieve certain objectives through interaction with the environment in which they are immersed.

Reinforcement learning is one of the three main paradigms of machine learning, along with supervised and unsupervised learning. Unlike the other two, reinforcement learning differs because the system is not trained with the sample dataset. Rather, the system learns through trial and error. Therefore, a sequence of successful decisions it will involve the "strengthening" of the process because it will be better to solve the problem .

The quality of an action is given by a numerical value of "reward", inspired by the concept of reinforcement , which aims to encourage correct behavior of the agent.

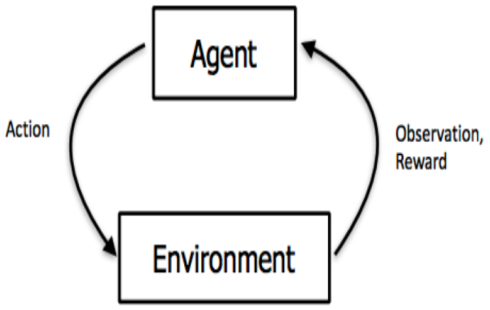


Figure 6 learning by reinforcement

Reinforcement learning is therefore a behavioral learning model.

One of the most common applications of such learning is found in robotics or play. Let's take the example of wanting to train a robot to climb stairs. The robot changes its

approach based on the result of its actions. When the robot falls, the data is recalibrated so that the steps are performed differently until the robot, which has been trained by trial and error, figured out how to climb stairs. In other words, the the robot learns on the basis of a sequence of correctly performed actions.

The learning algorithm must therefore be able to discover an association between: the goal of climbing stairs successfully without falling and the sequence of events that lead to the result.

3.5 "THE PROCESS" OF MACHINE LEARNING

The goal of machine learning is to derive meaning from data. Therefore, data is the key to unlocking machine learning. There are seven steps to machine learning, and each step revolves around data. The flow of operations to determine a machine learning model will be:

1. **Data collection**

Machine learning requires training data, in large part (labeled, which means supervised learning or untagged , which means unsupervised learning). Data collection, or datafication , is the first step in my new model.

2. **Data preparation**

Raw data alone is not very useful. The data must be prepared, normalized, deduplicated , and errors must be removed. We always aim to give the dataset a certain level of generalization, to allow a more common prediction.

3. **Choosing a model**

The third step is to select the right model. There are many models that can be used for different purposes. After selecting the model, you need to make sure that the model meets the intended objective. Also, you need to know how much preparation the model

requires, how accurate it is, and how scalable the model is. A more complex model does not always make a better model. Commonly used machine learning algorithms include linear regression, logistic regression, decision trees, Support Vector Machines (SVM), Naïve Bayes ,, Neural Networks and countless different high models.

4. **Training**

Model training is the bulk of machine learning. The goal is to use training data to incrementally improve the model's prediction performance.

5. **Evaluation**

After training, the model is evaluated. This involves testing machine learning against an unused control dataset to see how it behaves. This may be representative of how the model works in the real world, but it doesn't have to be. The greater the number of variables in the real world, the more training and test data should be.

6. **Parameter tuning**

After evaluating the model, this step refers to the tuning of the hyperparameter that is, the optimization of model parameters to improve performance.
hyperparameters can include: number of training steps, learning rate, seed and distribution values, etc.

7. **Forecast**

After completing the process of data collection, data preparation, model selection, model training and evaluation, and parameter optimization, it is time to answer the questions using predictions. These can be all kinds of predictions, ranging from image recognition to semantics to predictive analytics.

3.6 CHOICE OF THE MODEL

The choice of the model and therefore of the machine learning algorithm to be used is determined following a process of trial and error. It is also determined following a

compromise between the various specific characteristics (speed, predictive precision, interpretability, etc.) of the algorithms. For this reason it is worthwhile to dwell on the choice of the model and therefore on the type of optimal algorithm for our problem. In fact, we can have several algorithms, also divided by the type of prediction.

In the course of this text we will discuss the most important and famous machine learning algorithms in case of **supervised learning** :

3.6.1 CLASSIFICATION ALGORITHMS

Below we will do an in-depth analysis regarding the algorithms used for classification problems.

3.6.1.1 LOGISTIC REGRESSION

Suitable for a model capable of predicting the probability of a binary file with an answer therefore belonging to one class or another. Because of his

simplicity, logistic regression is used as a starting point for binary classification problems.

Best used when the data can be clearly separated by a single linear boundary.

3.6.1.2 kNN (K-NEAREST NEIGHBORS)

Figure 7 logistic regression

The instance is classified "by majority" according to the most common class among the k closest instances of the training set; it is assumed in kNN that neighboring objects are similar.

Given a new instance x to classify, the classifier looks for the training k examples that are most "similar" to x and looks at their labels. Whichever label occurs most frequently

from the nearest k labels, is chosen to assign the class to x. Distance metrics such as Euclidean ones are used to determine the "nearest neighbor".

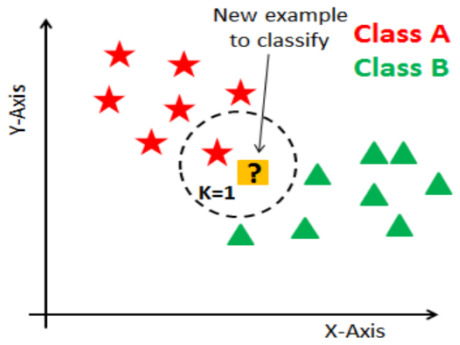


Figure 8 kNN

3.6.1.3 4SVM (SUPPORT VECTOR MACHINES)

An SVM model is a representation of the training set examples as points in a space, so that the examples of separate categories have been divided by a hyperplane which is an obvious gap that must be as large as possible. The greater the gap , the more optimized the hyperplane will be. New examples will therefore be mapped in the same space and a prediction of the category they belong to will take place based on which part of the gap they fall. If the data is not linearly separable, the loss function is used to penalize points on the wrong side of the hyperplane. SVMs sometimes use kernel transformation to transform nonlinearly separable data into higher dimensions where a hyperplane can be found.

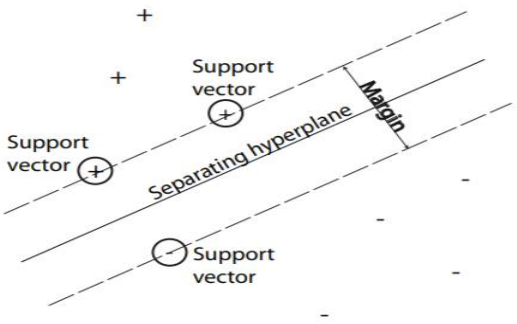


Figure 9SVM

The most important goal of an SVM is, as already mentioned, to find the hyperplane that classifies all training vectors into two classes and to maximize the margin which is

the distance between the hyperplane and the carriers (support vectors) closest to it. of both classes.

3.6.1.4 NEURAL NETWORK

They are algorithms inspired by the human brain, in fact neural networks are highly connected "networks of neurons" that relate inputs to desired outputs. The network is iteratively trained by changing the strengths of the connections so that the inputs are mapped to the correct answer. Such algorithms are best used for modeling highly nonlinear systems and when data is incrementally available and you want to constantly update the model.

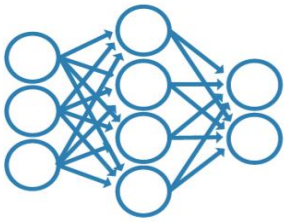
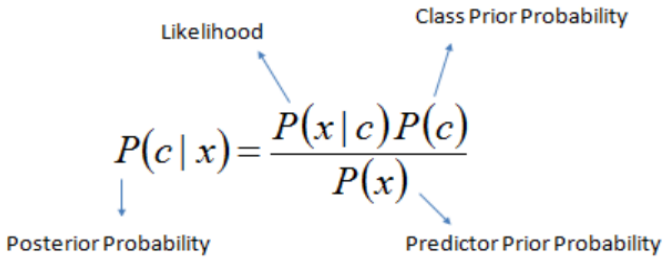


Figure 10 neural network

3.6.1.5 NAIVE BAYES

In Naive Bayes classifiers are simple probabilistic classifiers based on Bayes ' theorem. The way this algorithm works is developed in these four points:

1. Learn the probabilities that connect features to each of the possible classes.
2. Multiply all the probabilities for each resulting class.
3. Normalize the probabilities by dividing them by the total sum.
4. Takes the most likely class as an answer.



$$P(c \mid \mathbf{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

- $P(c|x)$ is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

Ultimately we could say that this algorithm classifies new data based on the highest probability value of belonging to a particular class.

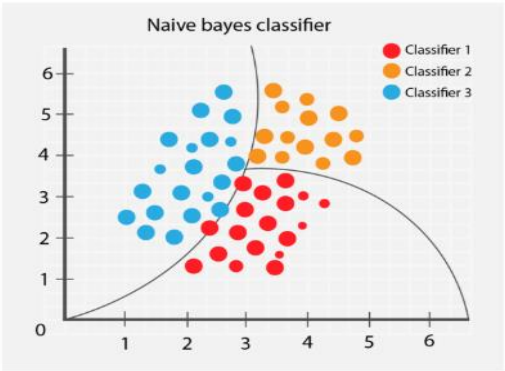


Figure 11 Naive Bayes

3.6.1.6 DISCRIMINANT ANALYSIS

Discriminant analysis classifies data by finding linear combinations of features. Discriminant analysis assumes that different classes generate data based on Gaussian distributions. The formation of a discriminant analysis model involves the search for parameters for the Gaussian distribution for each class. Distribution parameters are used to calculate boundaries, which can be linear boundaries or quadratic functions. These boundaries are used to determine and then separate classes of new data.

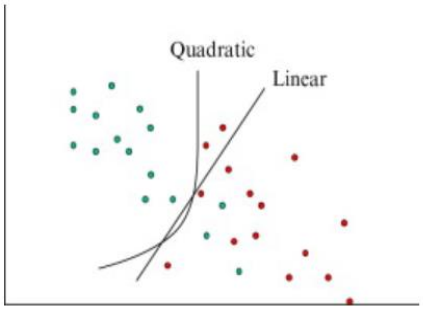


Figure 12 Discriminant Analysis

3.6.1.7 DECISION TREE

A decision tree allows you to predict the responses to the following data decisions in the tree from the root (beginning) to the leaf node.

The decision tree follows certain steps in data classification:

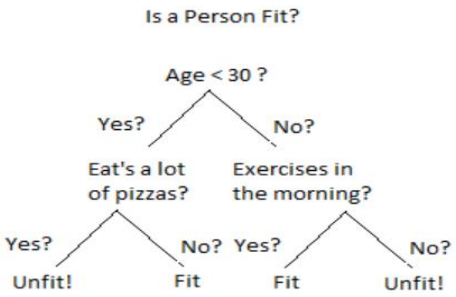
1) puts all the "training examples " in the root;

2) divide the ”training example ”based on the selected attributes;

3) select attributes using statistical measures;

4) the partition continues until, no training example remains .

The decision tree is built through a process known as binary recursive partitioning. This is an iterative process of partitioning the data and splitting it further on each of the branches. The Decision Tree can be used to map the decisions that could come out of any problem posed and every node present in the Decision Tree represents a possible decision you can make.



To try to understand the decision tree model better, we can consider the example above, where we ask ourselves the problem of predicting whether a person is fit or not. So given the person's information such as age, eating habits, physical activity (reported in the

form of decision nodes) and scaling these decision nodes, we arrive at a prediction in the leaf node.

3.6.1.8 ENSEMBLE

Ensemble models are a combination of several basic machine learning models in order to produce better predictive performance than using a single model.

Some techniques for performing ensemble models are:

1. Bagging is used when our goal is to reduce the variance of a decision tree. Here the idea is to create different subsets of data from the training sample and each data collection from the subset is used to train their trees. As a result, we end up with a set of different models.

2. Boosting is an iterative technique that adjusts the weight of an observation based on the latest classification. If an observation has been classified incorrectly, try to increase the weight of this observation. Boosting in general creates strong predictive models.

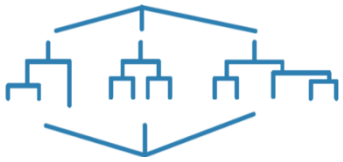


Figure 13 Ensemble

3.6.2 REGRESSION ALGORITHMS

Below we will do an in-depth analysis regarding the algorithms used for regression problems.

3.6.2.1 LINEAR REGRESSION

Linear regression is a statistical modeling technique used for describe a continuous response variable as a linear function of one or more predictive variables. Regression models a predicted (target) value based on independent variables, that is, it determines how much an independent variable x is really influential in the prediction of the

dependent variable y. Ultimately, linear regression allows you to learn a function that, given an example not previously known as input, is able to predict the output value.

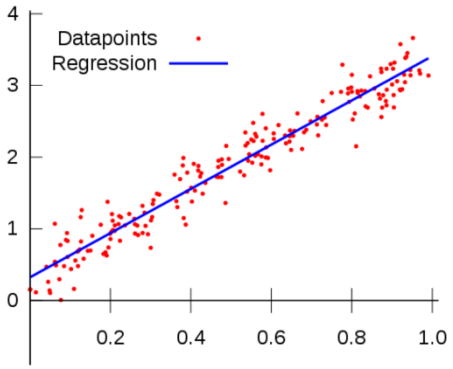


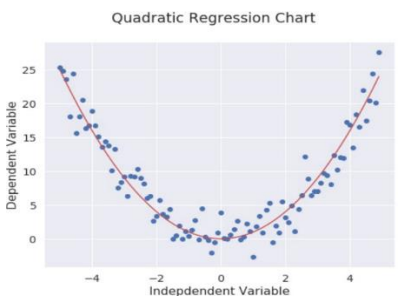
Figure 14 linear regression

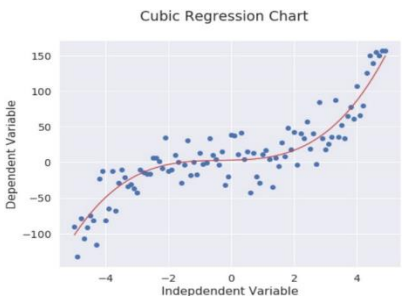
As we can see from the regression graph represented here, each red point represents an analyzed data. The closer the red points are to the regression line, the more powerful the model is; as we will identify our statistical distribution as a normal distribution.

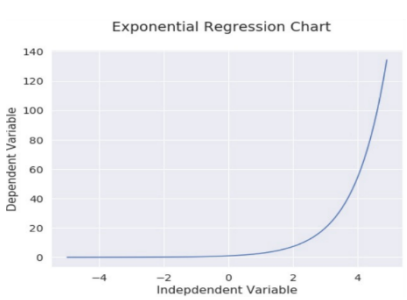
3.6.2.2 NON LINEAR REGRESSION

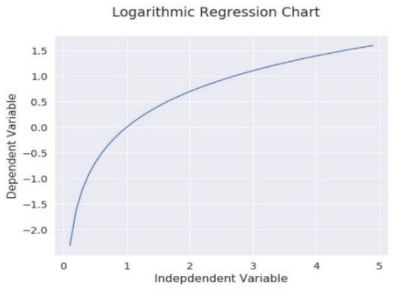
Nonlinear regression is a statistical modeling technique that helps to describe nonlinear relationships in experimental data.We can also say that it is any relationship between an independent variable x and a dependent variable y that results in data modeled on a nonlinear function and therefore not with a straight line, as in the previous case.

There are many types of nonlinear regression, each characterized by its own function; below I will report the most common:









3.6.2.3 GAUSSIAN PROCESS REGRESSION (GPR)

Gaussian process regression (GPR) models are probabilistic models used to predict the value of a continuous response variable. Training data are acquired (red points); they are subject to Gaussian observation with a certain standard deviation σ. The blue line shows the mean prediction μ (θ) of the regression of the Gaussian process o and the shaded region the corresponding uncertainty 2σ (θ). Gaussian processes make it possible to interpolate and extrapolate forecasts in regions of the parameter space where training data is absent.

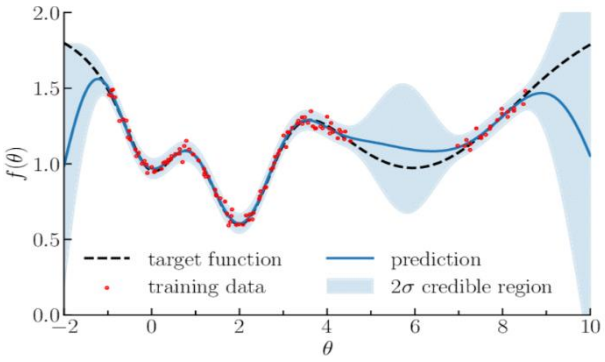


Figure 15 Gaussian Process Regression

3.6.2.4 SVM (SUPPORT VECTOR MACHINES) REGRESSION

SVM regression algorithms work the same as for the classification we talked about earlier (3.6.1.3).

They have been modified to allow for continuous responses. However , the main idea is always the same: to minimize the error, by identifying the hyperplane that maximizes the margin, but the goal when working with the SVM REGRESSION is basically to

consider the points that are inside the line therefore, our most suitable hyperplane is the one with the maximum number of points (analyzed data) within the boundary.

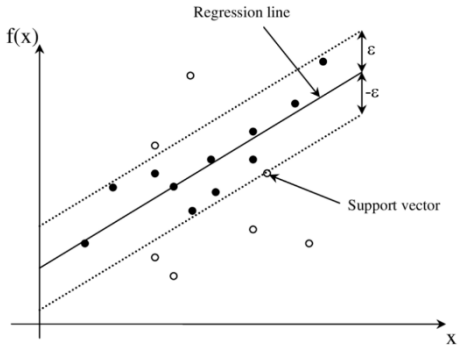


Figure 16 Support Vector Machines regression

3.6.2.5 DECISION TREE REGRESSION

decision algorithms tree for regression behave in the same way as those used for classification; obviously the prediction values must be modified which from discrete (for classification), become continuous (for regression).

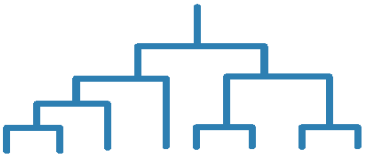


Figure 17 decision tree regression

3.7 FINAL CONSIDERATIONS

Machine learning is used everywhere today. When we interact with banks, shop online or use social media, machine learning algorithms are used to make our experience efficient, easy and safe. Machine learning helps extract meaningful information from a large set of raw data. When implemented right, machine learning can serve as a solution to a variety of business complexity problems and predict complex customer behaviors. Precisely for this reason machine learning is enjoying considerable success in the economic-financial field, since through the analysis of more and more numerous data collected (big data) it is possible to make more in-depth and accurate analyzes, this because, having a sample of so numerous data it is possible to approximate with more precision: market trends (for example predicting which product to remove from the

market as it will not be sold, and vice versa knowing a priori on which product to invest), customer behavior, predictive analysis of production, improvement of management and the ability to take risks or even machine learning used to manage pricing formulation, data analysis to calibrate the offer and generate communications and decisions etc ...; in general, however, we can find uses of machine learning a little throughout the business and industrial environment.

Figure 18 economic-financial benefits powered by ML

4. DEEP LEARNING

4.1 GENERAL INFORMATION ON DEEP LEARNING TECHNIQUES

Currently, AI is progressing rapidly. Deep Learning is one of the contributors and indicates that branch of Intelligence Artificial that does reference to algorithms inspired to the structure and function of the brain called artificial neural networks . Deep Learning (also known as deep structured learning or hierarchical learning), in fact, is part of a larger family of Machine Learning methods based on the assimilation of data representations, as opposed to algorithms for performing specific tasks. We could define Deep Learning as a system that uses a class of machine learning algorithms that:

- they use various levels of cascaded non-linear units to perform feature extraction and transformation tasks. Each subsequent level uses the level output previous one how input. The algorithms can be in form of both supervised and Not supervised and the applications include the analysis from *pattern* (unsupervised learning) and classification (learning supervised);
- they are based on unsupervised learning of multiple hierarchical levels of data characteristics (and representations). The higher level characteristics are derived from the lower level ones to create a hierarchical representation;
- they are part of the broader class of data representation learning algorithms within machine learning;
- they learn multiples levels from representation that Correspond to different levels of abstraction; these levels form a hierarchy of concepts.

By applying Deep Learning, we will therefore have a "machine" that is able to autonomously classify data and structure them hierarchically, finding the most relevant and useful ones for solving a problem (exactly as the human mind does), improving its performance with continuous learning.

4.2 PRINCIPLES OF DEEP LEARNING AND ITS ADVANTAGES

Deep Learning, bases the his operation on the classification and selection of the most relevant data to reach a conclusion. In the same way our biological brain behaves, which in order to formulate an answer to a question, deduce a logical hypothesis, arrive at the

resolution of a problem, sets in motion its own biological neurons and neural connections (interconnected biological neurons form our brain neural networks , those that allow each individual to reason, make calculations in parallel, recognize sounds, images, learn and Act). The Deep Learning leads in the same way by exploiting artificial neural networks, mathematical-computer computation models based on the functioning of biological neural networks, ie models made up of information interconnections. A neural network actually looks like an adaptive system in degree from change there her structure (the knots And the interconnections) relying on is on data external is on information internal that connect and they pass through there net neural during the learning and reasoning phase. Deep neural networks exploit a greater number of intermediate layers (*Hidden layer*) to build multiple levels of abstraction that can give deep neural networks a huge advantage in learning to solve complex pattern recognition problems precisely because at each intermediate level they add information and analysis useful to provide reliable output. It is easy to understand that the more intermediate levels there are in a deep neural network (and therefore the larger the neural network itself) the more effective the result (the task that is "called" to carry out) but, from versus, there scalability of the neural network is closely related to dataset, to the models mathematicians And at resources computational. Although there request from computational skills huge may to represent a limit, there scalability of the Deep Learning thanks to the increase in available data and algorithms is what differentiates it from Machine Learning: the systems from Deep Learning, indeed, improve the own performance as data increases while Machine Learning applications (or rather, the so-called superficial learning systems), once a certain level of performance has been reached, are no longer scalable even by adding examples and training data to the network neural.

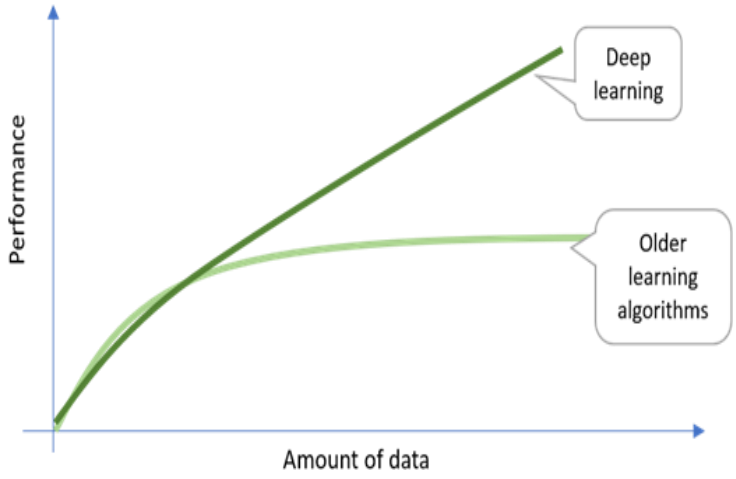


Figure 19 Cartesian diagram showing the comparison between the performance trend of Machine Learning or Deep Learning algorithms as a function of the amount of data

This happens because in Machine Learning systems the characteristics of a certain object (in the case of visual recognition systems) are extracted and selected manually and are used to create a model capable of categorizing objects (based on the classification and recognition of those features); in Deep Learning systems, on the other hand, the extraction of characteristics takes place automatically: the neural network autonomously learns how to analyze raw data and how to carry out a task (for example, classify an object by recognizing its characteristics autonomously).

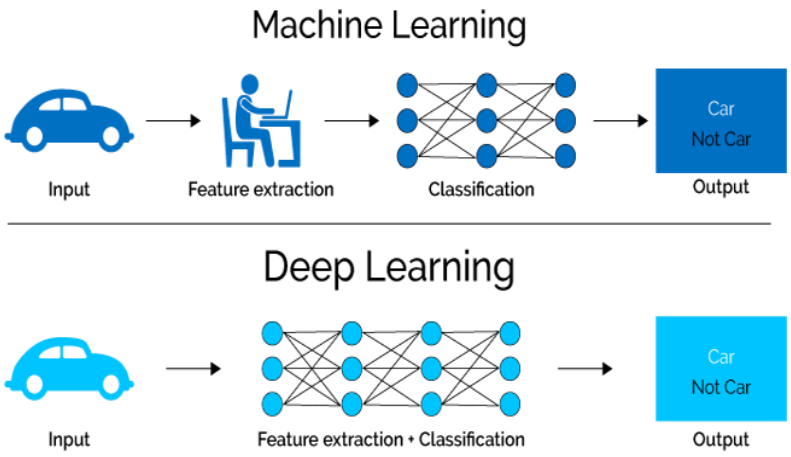


Figure 20 Difference in the performance of Feature extraction in Machine Learning and in the Deep Learning

If from the point of view of potential Deep Learning may seem more "fascinating" and useful of the Machine Learning, it goes specified that the

calculation computational required for the their operation is really important, even from an economic point of view : the CPUs and the More advanced GPUs useful for supporting *workloads* a Deep Learning system still cost thousands of dollars; the use of computational capabilities via the *Cloud* only partially mitigates the problem because the formation of a deep neural network often requires the processing of large amounts of data using high-end GPU clusters for many, many hours (it is therefore not certain that buying " As a service" the necessary computing capacity is economical).

4.3 CASES OF USE AND TYPES OF APPLICATIONS OF DEEP LEARNING

Despite the problems illustrated, Deep Learning systems have made enormous strides evolutionary and are improved very much in last years, mostly for there huge amount of data available but above all for the availability of ultra-performing infrastructures (CPU and GPU in particular). The early sectors adopting the activities have seen a strong effect on the workplace and great potential in terms of developing deep learning applications. Deep learning has enabled computers to take a step forward, specifically to solve a number of complex problems. Already today there are use cases and areas of application that we can also see as "ordinary citizens" who are not tech - savvy. *From Computer Vision* for driverless cars to drones and robots; speech and language recognition and synthesis for *chatbots* and service robots; facial recognition for surveillance. Deep learning technology has found its way into the financial services industry. An important task that the Deep Learning can carry out is e-Discovery. To example, the large society of investment how JPMorgan Chase use the analysis of the text based on Deep Learning for insider trading detection and compliance with government regulations. One of the advantages of Deep Learning over other approaches is accuracy. In many cases, the improvement approaches a 99.9% detection rate. The high

risk and associated cost to the missed detection from a threat at the safety make justified there expense related to the Deep Learning.

4.4 FEEDFORWARD *NEURAL NETWORKS* (MLP)

As already stated, one of the main algorithms on which Deep Learning is based is the neural network. It is defined as a computer system consisting of a number of elements or knots simple but highly interconnected, called "neurons", that are organized in layers that process information using dynamic state responses to external inputs. We distinguish: an Input Layer , designed to receive information from the outside in order to learn to recognize and process the same information received; hidden layer (*Hidden Layer*) that connects the input layer with the output layer and helps the neural network to learn the complex relationships analyzed by the data. Often the levels hidden are more than one. And in the end we have the level from exit (*Output Layer*), the final level showing the result, i.e. how much the algorithm has managed to learn. The simplest example of a neural network is *feedforward networks* (Or *Multi-Layer Perceptron*), described by an acyclic graph direct.

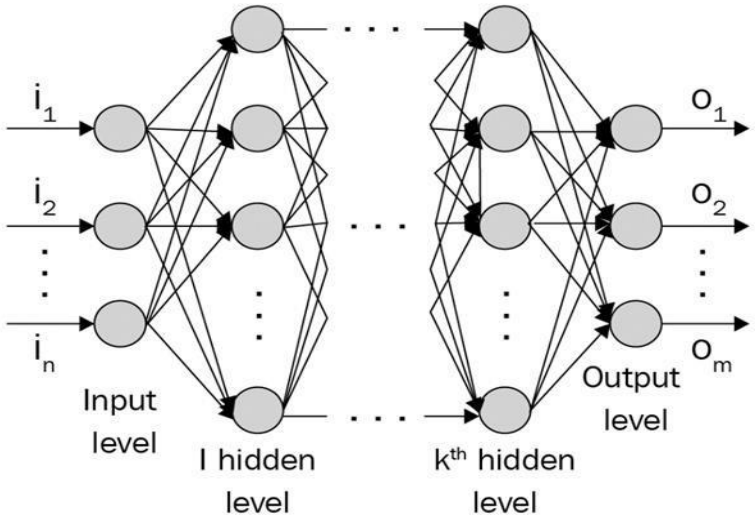


Figure 21 Basic structure of a Deep Feedforward Networks or MLP

In this network, the flow of information is one-way: when learning (through training) or when operating under normal conditions (after being trained) the schematized information is fed into the network from the input layer, directed into the hidden layers and finally they arrive at the exit level. Most neural networks are completely connected, that is every neuron belonging to the level hidden turns out connected with every neuron of the level from exit. Self were missing someone of the connections synaptic, one would speak of a partially connected neural network. Each connection between neurons is associated with a weight (commonly referred to as W *)* which determines the importance of the input value and which is multiplied by the value of the connected neuron. The starting weights are set randomly. Each neuron sums all the inputs received in this way and adds to it a value from bias (indicated with b). The bias is how the intercept addition in a linear equation. It is an additional parameter in the neural network that is used to adjust the output along with the weighted sum of the inputs to the neuron. It is essentially a constant that helps the model in a way that can better fit the data. A preset activation function is applied to this result which does nothing but mathematically transform the value before moving on to the next layer. One of its purposes is to standardize the output from the neuron if the sum is greater than a certain value threshold, he comes activated the unit to which is connected (those at the her right). There part more complex is learning. The networks neural are scheduled for learn at the same way: they use an algorithm called *Backpropagation* . In general, a Back- propagation algorithm foresees to compare the result obtained from a network with the output that is actually wanted to be obtained. Specifically using the difference between the two results plans to change the weights of the connections between the levels of the net leaving from the level output. After of which proceeding backwards involves modifying the weights of the hidden levels and finally those of the input levels. Over time, *Backpropagation* makes the

network learn, reducing the difference between the output actual And that planned to the point in which the two coincide exactly, so the network understands things exactly as it should. To quantify how far our forecast is from our real value, let's calculate an error or, in other words, define a cost function (the cost function is nothing more than the error in predicting the output correct that has there our net; in other terms, is there difference between the planned And the expected output). The goal is to minimize it through the descent of the gradient, a technique that allows us to find the global minimum of a function by modifying the weights of small increments after each iteration of the set of data.

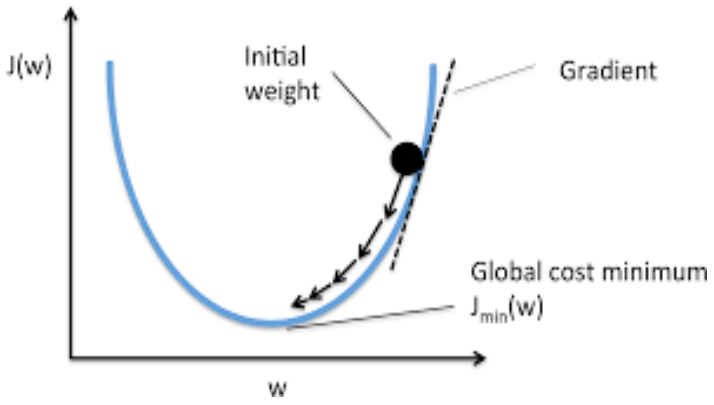


Figure 22 Example of how the gradient descent algorithm works

By calculating the derivative (or gradient) of the cost function with a given set of weights, we are able to see in which direction the global minimum of the function lies. The amount we choose to move in any direction is called the learning rate and it is what defines the speed at which we move towards the global minimum. There challenge is to find a *trade-off* , indeed a number Very small, could taking too many moves to get to this point; however, if we choose a very high number, we risk exceeding the point and never being able to reach it. So, in a nutshell, gradient descent works by repeatedly calculating the gradient ∇ C, updating the weights and distortions and trying to find the

correct values that minimize the cost of the function. And this is how the neural network learns. To minimize the cost function , it is necessary to cycle through the dataset several times through the algorithm.

4.5 RECURRING NEURAL NETWORKS (RNN)

Another widely used type of neural network architecture is the recurrent neural network. In the RNN or Recurrent Neural Networks, unlike *feedforward networks* where information can only go one way and each neuron can be interconnected with one or more neurons of the chain next one, in this kind of networks, the neurons can admit also of the loop and / or can be interconnected also to neurons from a previous one level. Recurring networks essentially provide for backward or forward links level.

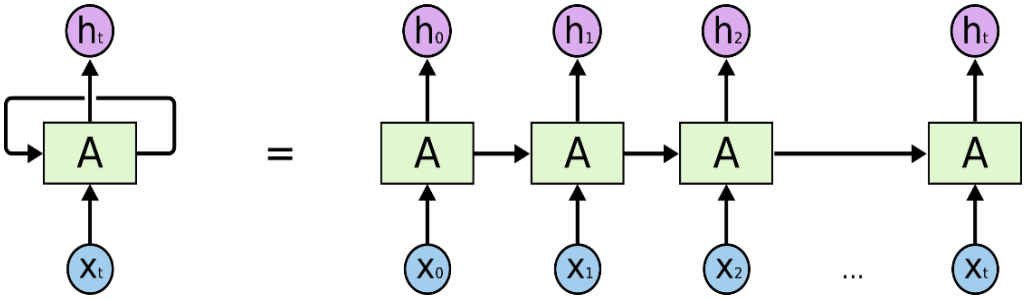


Figure 23 Structure of an unrolled recurrent neural network

This is an example of the typical structure of an unwound recurrent neural network. This is a very interesting feature, because the concept of recurrence intrinsically introduces the concept of memory of a network. In fact, in an RNN network, the output of a neuron can influence itself, in one step thunderstorm following can influence neurons of the previous chain which in turn will interfere with the behavior of the neuron on which close the loop. The RNN they work in the time, for which to difference of the networks *feedforward* classic in which the given provided was static, the kind of data that the RNN network are able to manage are a timeline or time series. In cell

RNN at each instant t, the level will receive its output S $_{(t-1)}$ in addition to the input X $_{(t)}$. The feedback of the output will allow the network to ultimately base its decisions on history. In this type of approach, it will be necessary to set the maximum number of iterations to be treated, otherwise, as can be easily understood, the network would enter a loop. The architecture of an RNN limits its long-term memory capabilities, which manage to perfectly manage long-term sequences but only remember a few sequences at a time. Consequently, the memory of RNN is only useful for shorter sequences and shorter periods of time. In fact, the addition of too many time steps increases the possibility of being faced with a *Vanishing* problem *Gradient* i.e. the loss of information during the Backpropagation.

4.5.1 LSTM

The Long-short-term memory are particular recurrent neural networks, designed to overcome the problem of *vanishing Gradient* and retain information for longer periods than traditional RNNs. They can keep a constant error, which allows them to continue learning through numerous time passages. LSTMs structurally use gate cells to *store* information outside the regular flow of the network. All recurrent neural networks take the form of a chain of repeating neural network modules. In standard RNNs, this repetitive module will have a very simple structure, with a single *tanh layer.* LSTMs also have this one-like structure chain, but the module repeated has a structure different. Instead to have a single neural level, there are four of them, which interact in a great special way.

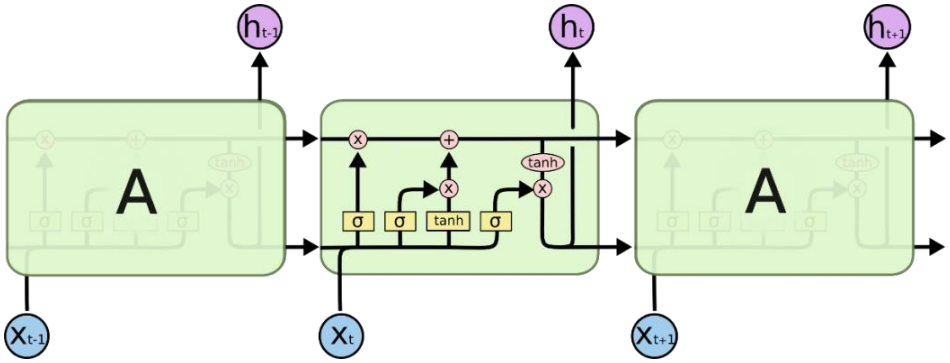


Figure 24 Structure of a repetitive module in an LSTM

The repetitive module in an LSTM contains four interacting levels.

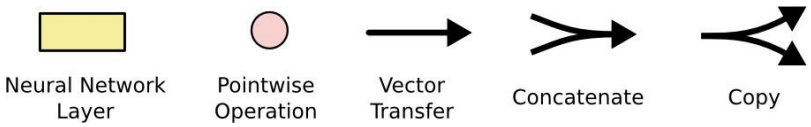


Figure 25: LSTM structure legend

As can be seen in figure, each line carries an entire vector, from the output of one node to the inputs of the others. The pink circles represent point operations, such as adding vectors, while the yellow boxes are learned neural network layers. The union of the lines denotes their concatenation, while a bifurcation indicates that his content he comes copied and copies go to different locations. The key to LSTMs is the state of the cell, the horizontal line running across the top of the diagram. The state of the cell is a bit like a conveyor belt. It runs along the entire chain, with only a few minor linear interactions. It is very easy for information to flow without changes.

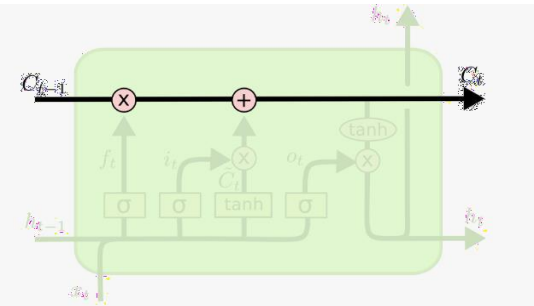


Figure 26 I know the state of the cell

LSTM has the ability to remove or add information to the cellular state, which is carefully regulated by structures called gates. Gates are a way to optionally let information pass. They are composed of a sigmoid neural network layer and a point multiplication operation.

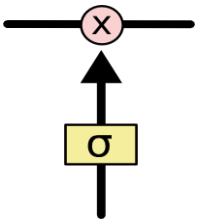
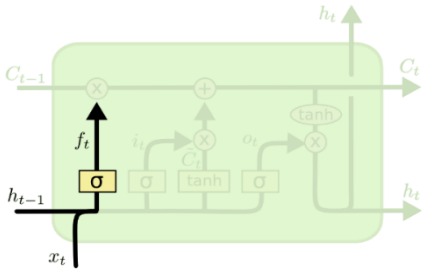


Figure 27 LSTM gates structure

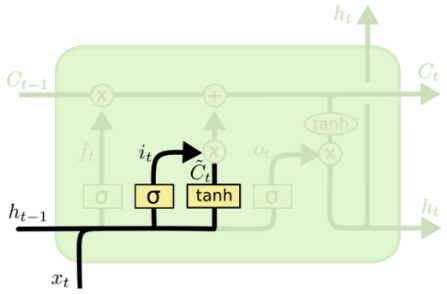
The sigmoid layer returns numbers between zero and one, describing the amount of each component that must be allowed through. The appearance of the value 0 does not let anything pass, while that of the value 1 lets everything pass. An LSTM has three of these ports, to protect and check the status of the cell. The first step is to decide which information to remove from the cell state. This decision is made by a sigmoid layer called the "*forget gate* ". Look at h_{t-1} and X_t and returns a number between 0 and 1 for each number in the state of cell C_{t-1} which will determine the pass of the information.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 28 Forget structure Gate

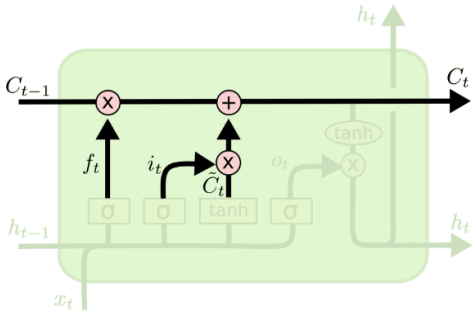
The next step is to decide what new information will be stored in the cell state. It consists of two parts. First, a sigmoid layer called the *Input Gate* decides which values will be updated. Pursued, a *tanh* layer creates a vector of new candidate values $\tilde{C}_t$, which could be added to the state. Next, these two will be combined to create an update to the state.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 29 Input Gate structure

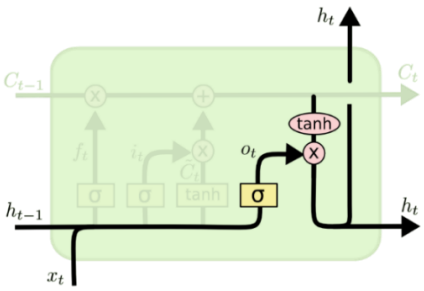
Subsequently, the old state of the cell, C_{t-1}, must be updated in the new state of the cell C_t. The previous steps have already decided what operations to perform, which they just need be fulfilled. Multiplying so the old state for f_t, forgetting the things you have decided to forget previously. And is added $i_t * \tilde{C}_t$. . These are the new ones values candidates, scaled by how much it was decided to update each status value.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 30 Cell status update

Finally, it must be decided what to produce. This output will be based on the state of our cell, but it will be a filtered version. First, a sigmoid layer is performed which decides which parts of the cellular state will be emitted. Then, we go to the state of the *tanh cell* , to insert the values between - 1 and 1 and multiply them by the sigmoid output , in order to produce only the parts that we have decided to create.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 31 Output Gate structure

4.6 CONVOLUTIONAL NEURAL NETWORKS (CNN)

In deep learning , a convolutional neural network (CNN or ConvNet , from English *Convolutional Neural Network*) is a type of artificial neural network feedforward where the connectivity pattern between neurons is inspired by the organization of the cortex visual animal, which neurons individual are willing in manner such from respond to the overlapping regions that handle *the* field of view. Convolutional networks are inspired from processes

biological and are variations of perceptrons multilayer designed for use preprocessing to a minimum . They have several applications in image and video recognition, in recommendation systems, in natural language processing, recently, in bioinformatics . A convolutional neural network is a learning algorithm deep that can to take an image from input, to assign importance (weights and learnable bias) to various aspects / objects in the image and being able to differentiate one from the other. The preprocessing required in a CNN is much lower than to other algorithms from classification. The CNN they follow so an architecture to levels, typically not cyclical.

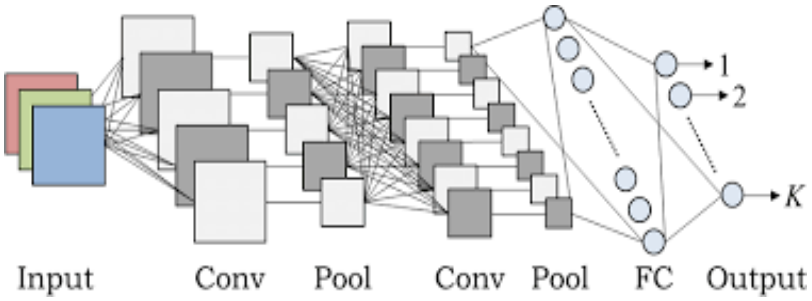


Figure 32 Structure of a CNN

The most important layers are the convolution layers , which name it.

4.6.1 CONVOLUTION

The first layer, which gives the name to architecture given its importance, is that of convolution. The element involved in performing the convolution operation in the first part of a convolutional layer is called the *Kernel* / Filter.

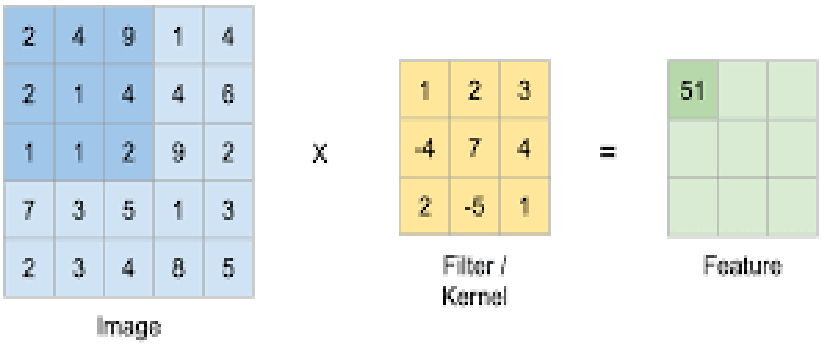


Figure 33 Graphical representation of the convolution operation

It is a weight matrix that acts like a moving average filter and extracts information details from the matrix original. Exist two types from results for the operation: one in which there characteristic convoluted is reduced in dimensionality respect to the input and the other one in which there dimensionality is increased or remains unchanged. Speak out from *Valid Padding* in the first case, or from *Same Padding* in the second. The weights they come learned in way such that there function loss is minimized in a similar way to an MLP. Therefore, we learn the weights to extract the functions from the original image that help the network in the correct prediction. The convolutional levels are the levels in which the filters they come applied to the image original or to other feature maps in deep CNN. This is where most of the user-specified parameters are found on the network.

4.6.2 ACTIVATION

At the end of the convolution operations it is applied to each output value of the *feature- map* an activation function.

4.6.3 POOLING

Similar to the convolutional level, the *pooling level* is responsible for reducing the spatial size of the convoluted feature. This is to reduce the

computational power required to process data by reducing dimensionality. Furthermore, it is useful for extracting the dominant characteristics which are invariant of rotation and position, thus maintaining the effective formation process of the template.

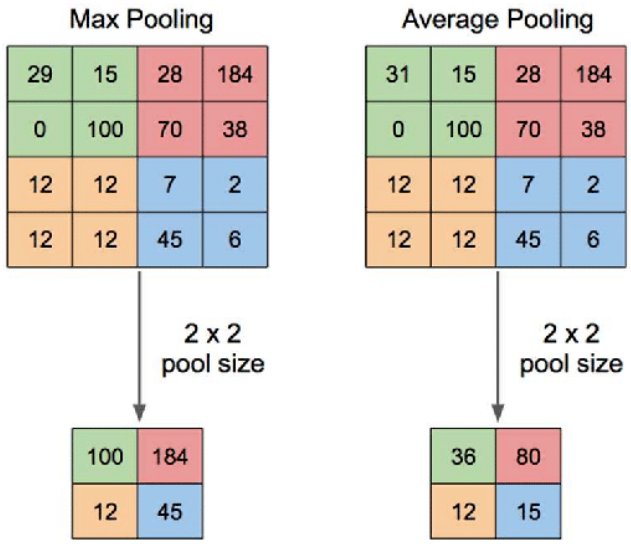


Figure 34 Graphical representation of the difference between Max Pooling and Average Pooling

There are two types of *pooling* : *Max Pooling* and *Average Pooling*. The former returns the maximum value from the part covered by the *kernel* , the latter returns the average of all values from the part image blanket from the *kernel* . The *Max Pooling* performs also there reduction of the noise along with the reduction of dimensionality . On *the* other hand, *Average Pooling* simply performs there reduction of the dimensionality how mechanism from reduction of the noise. The convolutional layer and the clustering layer together form the i-th layer of a convolutional neural network. Depending on the complexity, the number of such layers can be increased to further capture low-level details, but at the expense of more power. computational.

4.6.4 FULLY CONNECTED LAYER

At the end of the net, they find one or more layers from neurons completely connected that they are placed before the output of a CNN and are sent back

to an activation function that changes depending on the role of the layer, or of the goal of the net that either classification or regression. This is similar to the output level of a MLP.

4.6.5 CNN 1D AND THE CASE OF THE HISTORICAL SERIES

CNN's work the same way whether they are 1, 2, or 3 dimensions. The difference is in the structure of the input data and the way the filter, or convolution *kernel*, moves through the data.

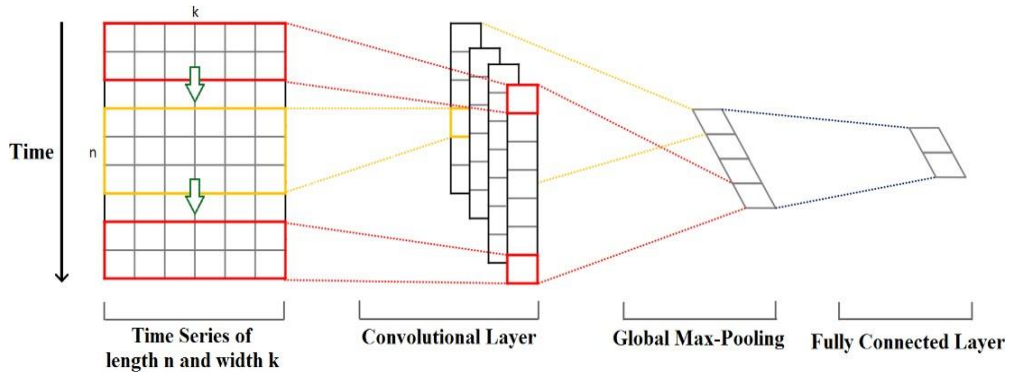


Figure 35 Structure of a CNN 1D for the analysis of a time series

In this case, we are going to consider a time series of length *n* and width *k*. The length is the number of time steps considered and the width is the number of variables in a multivariate time series. Convolution *kernels* always have the same width time series, while their length can be varied. In this way, the *kernel* moves in one direction from the beginning of a time series toward its end, performing convolution. The elements of the *kernel* are multiplied by the corresponding elements of the time series they cover at a given point. Then the results of the multiplication come added together and to the value, he comes applied a function from activation Not linear. The resulting value becomes an element of a new univariate time series "filtered", hence the *kernel* move in to come on long the series temporal to produce the following value. The number of new "filtered" time series is equal to the number of

convolution *kernels*. Go through a function from activation, he comes applied the *max pooling* global to each vectors of the series temporal filtered: the value great he comes to extract from each vector. From these values will be created a new one, containing the maximum, that will be the vector of the final function and which can be used as an input for a normal fully connected level. After the Flatten Arrays operation, the final output is sent to a fully connected layer. A 1D CNN can derive important functionality from short segments of an overall dataset when the location of each segment is not that important. The model extracts the functionality from the sequence data and maps the internal functionality of the sequence. Research has shown that the use of 1D CNNs for classifying time series has several important advantages over other methods. Are models highly resistant to noise are able to extract functions very informative and deep, time-independent, and automatically create informative representations of time series.

5. FORECAST MODEL DEVELOPMENT

5.1ABSTRACT

The market is unstable and, more than often, unpredictable. For several decades researchers have tried with time-series data to predict future values of which the most challenging and potentially lucrative application is predicting the values of stocks for a given company. However, as expected, market change depends on many parameters of which only a bunch can be quantified, such as historical stock data, the volume of trade, current prices. Of course, fundamental factors such as a company's intrinsic value, assets, quarterly performance, recent investments, and strategies all affect the traders' trust in the company and thus the price of its stock. Only a few of the latter can be incorporated effectively into a mathematical model. This makes stock price prediction using machine learning challenging and unreliable to a certain extent. Moreover, it is difficult to anticipate a piece of news that will shatter or boost the stock market in the coming weeks ie a pandemic or a war. In this second part of the thesis we develop using Python a neural network model capable of predicting the closing price of a company's shares. In our work we have considered that of Apple Inc.

5.2 DATASET & DATA CLEANING

For the dataset we relied on Yahoo Finance we used the pandas library to automatically import the observations from the platform containing the closing prices of the Apple Inc stock market from 01/01/2000 to 01/11/2022.

```
#Get the stock quote
df = pdr.get_data_ yahoo ( "AAPL" , start = "2000-01-01" , end
= "2022-11-01" )
```

The file consisted of 5745 observations illustrated by 6 features: Open, High, Low, Close, Adj Close and Volume.

Below is an example:

Date	Open	High	Low	Close	Adj Close	Volume
03/01/2000	0.936384	1.004.464	0.907924	0.999442	0.851942	535796800
04/01/2000	0.966518	0.987723	0.903460	0.915179	0.780115	512377600
05/01/2000	0.926339	0.987165	0.919643	0.928571	0.791531	778321600
06/01/2000	0.947545	0.955357	0.848214	0.848214	0.723033	767972800
07/01/2000	0.861607	0.901786	0.852679	0.888393	0.757282	460734400
...	...	...	...	...	...	...
25/10/2022	150.089.996	152.490.005	149.360.001	152.339.996	152.087.708	74732300
26/10/2022	150.960.007	151.990.005	148.039.993	149.350.006	149.102.661	88194300
27/10/2022	148.070.007	149.050.003	144.130.005	144.800.003	144.560.196	109180200
28/10/2022	148.199.997	157.500.000	147.820.007	155.740.005	155.482.086	164762400
31/10/2022	153.160.004	154.240.005	151.919.998	153.339.996	153.086.044	97943200

Below is a graphical representation of how the series of temporal observations looks:

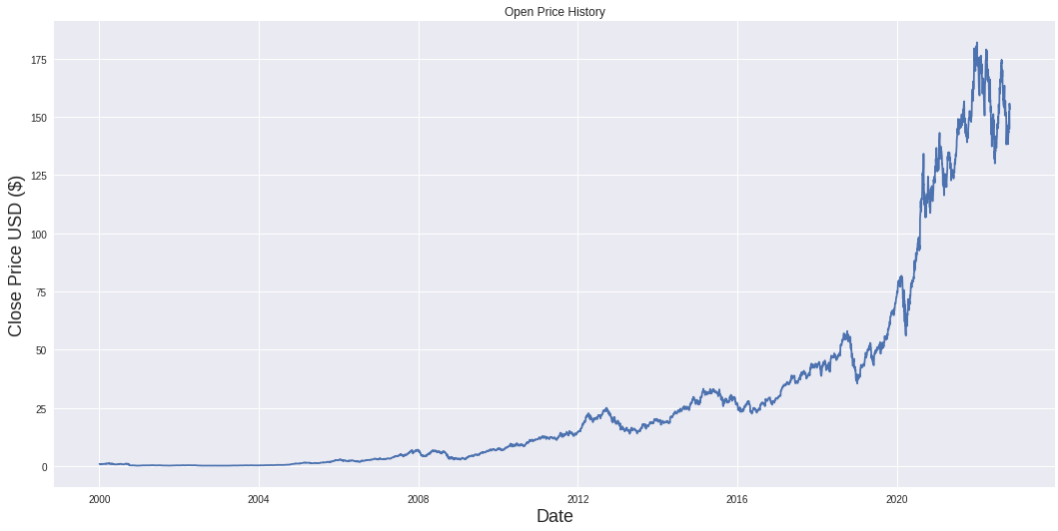


Figure 36 AAPL data visualization

We then made sure that the dataset did not contain outliers or nulls. To do this we have exploited the power of Python by automating the analysis as follows.

```
# Check missing values
df.isnull (). sum ()
Open 0
High 0
Low 0
Close 0
```

```
Adj Close 0
Volume 0
dtype : int64
```

There aren't missing value

From this analysis it appeared that the dataset did not need further modifications or cleaning processes.

5.1 DATA SPLITTING AND TRANSFORMATION

We decomposed the entire dataset by taking 80% for training, 10% for automatic algorithm validation , and the remaining 10% of the dataset we kept to test our prediction model. The validation set is used to measure the accuracy of the model during its development, so it is a test that is done autonomously on the algorithm.

In this way, therefore, we have a training dataset of 4596 observations, one of validation of 574 observations and finally that of test equal to 575 observations.

In this same part of the code we went to format the data and normalize them with a logarithmic function in order to reduce the dispersion of the data and make the model training more efficient.

5.4 MODEL CHOICE, TRAINING AND PERFORMANCE ASSESSMENT

The creation of the optimal model is a trial and error iteration. Which develops by modifying those hyperparameters that dictate the performance of the forecast model.

The Hyperparameters that we went to modify were:

- The number of **epochs** is a hyperparameter that defines the number of times the learning algorithm will work through the entire training dataset.
- **Batch size**, which is instead an hyperparameter that defines the number of samples to be analysed before updating the parameters of the internal model .

For example, having set the batch size to 200, the algorithm takes the first 200 samples (1st to 200th) from the training dataset and trains the network. Next, he takes the second 200 samples (201 to 400) and trains the net again.

In our analysis we went to consider the RNN and LSTM algorithms[26] as suitable for the prediction of time series, since CNN's algorithm are more complex one and used in other field of application as the robotics[27].

For evaluating the performance of algorithms such as loss function the root mean square error was chosen, which calculates the square of the difference between the actual value and the predicted value. One of the advantages it is used for is that the *Mean Squared Error* (or MSE) is great for ensuring that our trained model does not have anomalous predictions with huge errors, as it places more weight on these errors by means of the square part of the function. However, as a general performance index of the model we will use the R - Squared (R^2) index ie the determination coefficient. The R - squared is always smaller or equals than 1 and usually greater than 0 (is better to look for an R - squared close to 1).

5.4.1 FIRST ITERATION

To find out which algorithm was more optimized in our prediction model and for our data series, first carry out an as-is analysis so considering the model with all the features.

The results obtained are therefore:

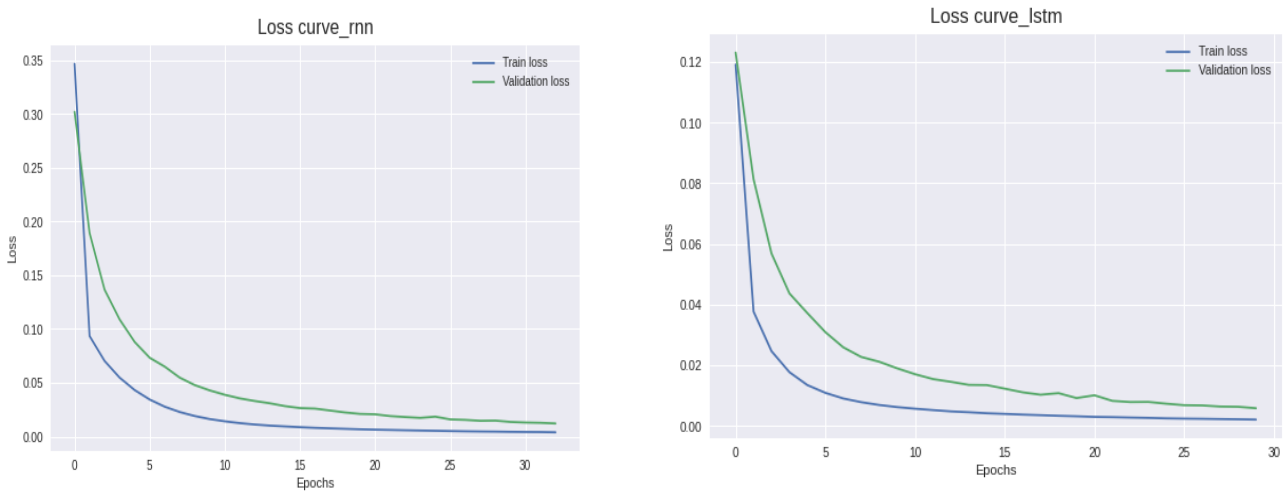


Figure 38 Loss curve comparison Rnn/LSTM

In this case we see how RNN is better as an application looking at the Loss curve indeed is evident as the RNN distances among the train loss and the validation one it start to flat over the epochs. Also looking at the performance of the forecast model we get an R^2 of 89.5723% compared to the LSTM one equal to 89.0679%.

5.4.2 SECOND ITERATION

In this second iteration we carry out a cross analysis by first subjecting our dataset to a statistical analysis to understand if all the variables were significant for purposes of prediction.

We will therefore exploit the correlation matrix. This statistical tool is of particular importance in any regression-based analysis as explained also in the paper [28], as it helps us to identify in a more detailed and conscious way the attributes to be incorporated in our prediction model.

We then use the correlation matrix to measure the degree of relationship between linearly related variables. In this case the correlation between the variables can assume a value included in the interval -1 and +1.

A correlation coefficient of 1 indicates that for every positive increase in one variable, there is a positive increase of a fixed proportion in the other variable; or if the coefficient takes value -1 there is a negative decrease of a fixed proportion in the other variable. For example, the amount of gasoline in a tank decreases in (almost) perfect correlation with speed. Value 0 of the correlation coefficient means that for each increase there is no positive or negative increase and therefore the two variables are not correlated.

	Date	Open	High	Low	Adj Close	Volume	Close
Date	1						
Open	0.899587	1					
High	0.902074	0.999268	1				
Low	0.899622	0.999042	0.998921	1			
Adj Close	0.904792	0.997926	0.999015	0.999024	1		
Volume	-0.47244	-0.48098	-0.46989	-0.49626	-0.48453	1	
Close	0.90087	0.997972	0.999028	0.999079	0.999952	-0.48388	1

As known from the table above we see that they all have a high correlation coefficient, except for the volume as it reports a negative correlation even lower than 0.5.

So let 's run the second test, therefore considering as features in the forecast model all except the volume.

For the two models the following results were obtained:

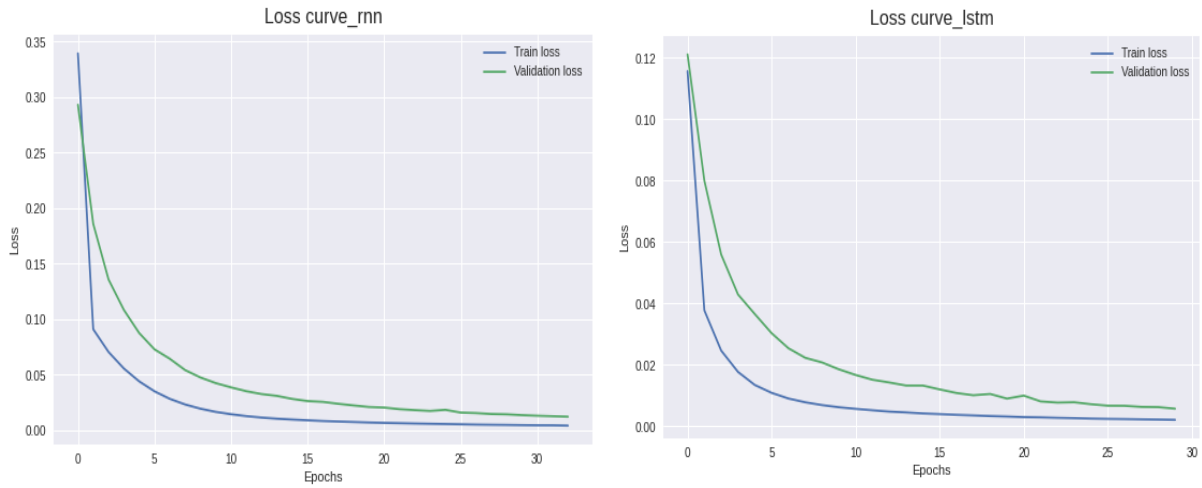


Figure 37 Loss curve comparison Rnn/LSTM

As evident from the graphs above, the algorithm performs well as we see how the MSE gradually decreases with the increase of the epochs and the validation loss flattens itself decisively on that of train loss.
We also see how the RNN is better also in this second analysis, reporting an R- squared of 92.9962% compared to the LSTM which is stated at 87.6019%.
Given that we see how this second model without considering the volume's feature in the model creation is performing better than the previous one.
That's result has confirmed what the correlation matrix highlighted so at the that feature was a potential cause outliers for the final prediction close price.

5.5 IMPROVEMENT OF THE MODEL AND RESULTS

Once we have defined which way to use in the model, ie that of the second iteration, let's now try to modify some hyperparameters in order to make our model even more performing.
Therefore, following various tests, the best hyperparameters were identified as:

- epochs = 36
- batch size = 32

This combination of parameters has in fact overturned the forecast making the model with the LSTM significantly more performing. That's not a surprise in fact given the large amount of data that we have to analyse and to capture long-term dependency on time series forecasting the LSTM algorithm has more memory to exploit.

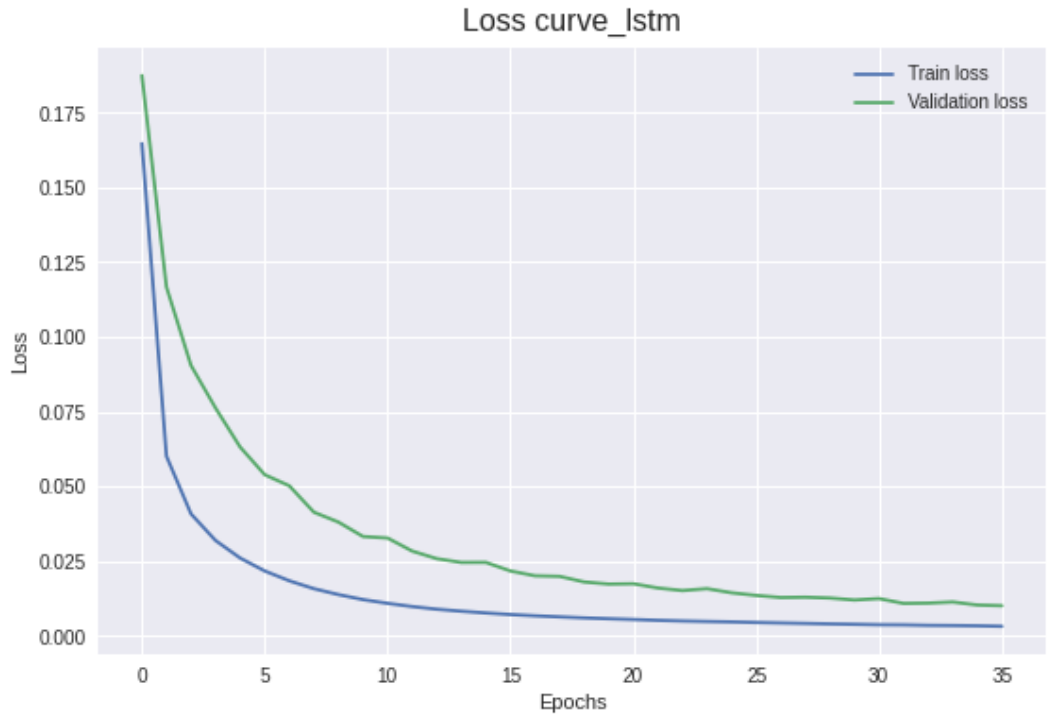


Figure 39 Loss curve final LSTM model

In fact, in this case study, the R- squared is confirmed at 95.5423% as can also be verified by the graph below, in fact the values predicted by the model almost perfectly override the real ones.

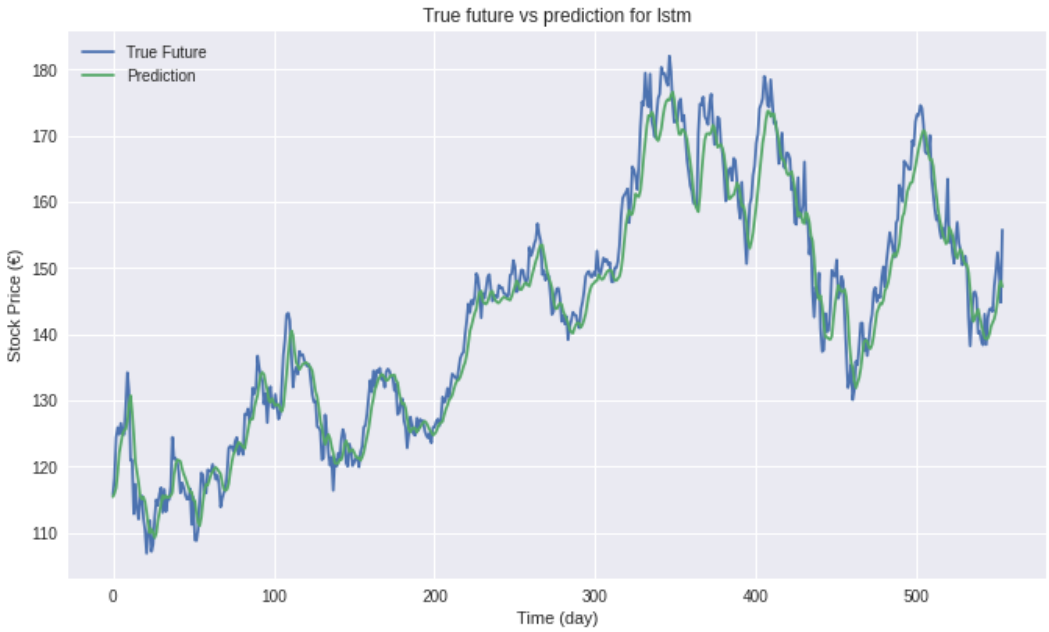


Figure 40 Result of the forecasting model

5.6 CONCLUSIONS AND FUTURE DEVELOPMENT

In this thesis work we have seen the potentialities and areas of application of artificial intelligence in the future of business. As all company functions are now adapting to a data driven vision as a support decision making.

As future developments this thesis work could see the implementation of longer-term prediction models than the one we just designed, as our neural network is capable of predicting the closing price relative to the following day only.

In addition, in order to make the prediction even more specific, is possible to develop a neural network that does not only as in our case consider the time series but also the signals that the market launches and therefore could think of designing a neural network that also analyses the information of the markets in economic newspapers and magazines in order to be able in this case to create a model that is more coherent and in line with the financial market, thus being able to predict even those cases that for our neural network would be considered outliers.

5.7CODE

```
!pip install joblib
!pip install yfinance

     #import the libreries
import numpy as np
import tensorflow as tf
import random as python_random
import sys
import seaborn as sns
import pandas as pd
from numpy import zeros, newaxis
from matplotlib import pyplot as plt
from keras.utils.vis_utils import plot_model
from tensorflow import keras
from joblib import dump
from sklearn.preprocessing import MinMaxScaler, PowerTransformer, Standa
rdScaler
from sklearn.metrics import mean_squared_error, r2_score, max_error, mea
n_absolute_error
from tensorflow.keras.optimizers import Adam, Nadam
```

```
from tensorflow.keras import Sequential, layers, callbacks
from tensorflow.keras.layers import Input, Dense, LSTM, Dropout, GRU, Bi
directional, SimpleRNN, Conv1D, MaxPooling1D, \
    Flatten, Activation
import math
import pandas_datareader as web
import numpy as np
import pandas as pd
from pandas_datareader import data as pdr
import yfinance as yf
yf.pdr_override()
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM, RNN, GRU
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')

plt.style.use('seaborn')

#Get the stock quote
df = pdr.get_data_yahoo("AAPL", start="2000-01-01", end="2022-11-01")
#show the data
df

#Visualizing the closing price history
plt.figure(figsize=(16,8))
plt.title('Close Price History')
plt.title('Open Price History')
plt.plot(df['Close'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.show()
```

Data prepocessing

Data Cleaning

Check presence of Missing values

```
# Check missing values
df.isnull().sum()
```

```
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

There aren't missing value

```
# Replace missing values by interpolation
def replace_missing(attribute):
    return attribute.interpolate(inplace=True)
```

Check and remove Outliers

```
# Detect and remove outliers with IQR
def detect_remove_outliers(df, column):
    # IQR
    Q1 = np.percentile(df[f'{column}'], 25, interpolation='midpoint')
    Q3 = np.percentile(df[f'{column}'], 75, interpolation='midpoint')
    IQR = Q3 - Q1

    # Above Upper bound
    upper = df[f'{column}'] >= (Q3 + 1.5 * IQR)
    # print("Upper bound:", upper)
    print("Upper bound outliers:", f'{column}', np.where(upper))

    # Below Lower bound
    lower = df[f'{column}'] <= (Q1 - 1.5 * IQR)
    # print("Lower bound:", lower)
    print("Lower bound:", f'{column}', np.where(lower))

    # Removing the Outliers
    # df.drop(upper, inplace = True)
    # df.drop(lower, inplace = True)

    # print("New Shape: ", df.shape)
    return

# There may be potential outliers in the Volume column, but they won't b
e considered outliers because
# a large volume of transactions is related to a change in the closing p
rice
# For the other columns it was previously verified graphically with the
boxplot that there aren't outliers.
# Also mathematically, with the IQR method, the same result is gotten.

titles = ["Open", "High", "Low", "Close", "Adj Close", "Volume"]

for i in titles:
       detect_remove_outliers(df, f'{i}')
```

- It seems that these are not outliers. They are a part of the trends of the timeseries

Data Splitting and Transformation

```
# Let's say we want to split the data in 80:10:10 for train:valid:test d
ataset it was decided to use a manual
train_size = 0.8
valid_size = 0.1

train_index = int(len(df) * train_size)
```

```
df_train = df[0:train_index]
df_rem = df[train_index:]

valid_index = int(len(df) * valid_size)

df_valid = df[train_index:train_index + valid_index]
df_test = df[train_index + valid_index:]
test_index = df_test.shape[0]

train = df_train

pts = {}

for i in df_train.columns:
    pt = PowerTransformer(method="yeo-johnson")
    s_s = pt.fit_transform(train[i].values.reshape(-1, 1))
    s_s = np.reshape(s_s, len(s_s))
    pts['pt_' + i] = pt
    train[i] = s_s

valid = df_valid

for i in df_train.columns:
    pt = pts['pt_' + i]
    s_s = pt.transform(valid[i].values.reshape(-1, 1))
    s_s = np.reshape(s_s, len(s_s))
    pts['pt_' + i] = pt
    valid[i] = s_s

test = df_test

for i in df_train.columns:
    pt = pts['pt_' + i]
    s_s = pt.transform(test[i].values.reshape(-1, 1))
    s_s = np.reshape(s_s, len(s_s))
    pts['pt_' + i] = pt
    test[i] = s_s


X_train, y_train = train.values[:, :5], train.values[:, 3]
X_valid, y_valid = valid.values[:, :5], valid.values[:, 3]
X_test, y_test = test.values[:, :5], test.values[:, 3]

print('X_train.shape:', X_train.shape, 'y_train.shape:', y_train.shape)
print('X_valid.shape:', X_valid.shape, 'y_valid.shape:', y_valid.shape)
print('X_test.shape:', X_test.shape, 'y_test.shape:', y_test.shape)
/

pts['pt_Close']

PowerTransformer()
```

*Set "**Window size**" and **step_ahead** to predict*

```
# Create a 3D input
def create_dataset(X, y, lag=1, n_ahead=1):
    Xs, ys = [], []
    for i in range(len(X) - lag - n_ahead):
        Xs.append(X[i:(i + lag)])
        ys.append(y[(i + lag):(i + lag + n_ahead)])
    return np.array(Xs), np.array(ys)


# Choose lag window
time_steps = 20
# Choose 1 for a single step prediction or 2, 3, ..., n for a multi step
prediction
step_ahead = 1

X_train, y_train = create_dataset(X_train, y_train, time_steps, step_ahe
ad)
X_test, y_test = create_dataset(X_test, y_test, time_steps, step_ahead)
X_valid, y_valid = create_dataset(X_valid, y_valid, time_steps, step_ahe
ad)


print('All shapes are: (batch, time, features)')
print('X_train.shape:', X_train.shape, 'y_train.shape:', y_train.shape)
print('X_valid.shape:', X_valid.shape, 'y_valid.shape:', y_valid.shape)
print('X_test.shape:', X_test.shape, 'y_test.shape:', y_test.shape)
```

```
All shapes are: (batch, time, features)
```

```
# Save preprocessed data and scaler
with open('Preprocessed_data_PG.npy', 'wb') as f:
    np.save(f, X_train)
    np.save(f, y_train)
    np.save(f, X_valid)
    np.save(f, y_valid)
    np.save(f, X_test)
    np.save(f, y_test)

dump(pts['pt_Close'], 'PowerTransformer_Close_PG.joblib')
```

```
['PowerTransformer_Close_PG.joblib']
```

4. Model Choice and Learning

```
X_train.shape
```

```
(4575, 20, 5)
```

```
# Create Simple RNN model
def create_rnn():
    model = Sequential()
    model.add(SimpleRNN(32, input_shape=(X_train.shape[1], X_train.shape
[2]),
```

```
                        kernel_regularizer=keras.regularizers.l2(0.01),
                        activity_regularizer=keras.regularizers.l2(0.1),
                        ))
    # model.add(Dropout(0))
    model.add(Dense(units=y_train.shape[1]))
    # Compile model
    model.compile(loss='mse', optimizer=Adam(learning_rate=0.001))  # De
fault_lr = 0.001
    model.summary()

    return model


# Create LSTM model
def create_lstm():
    model = Sequential()
    model.add(LSTM(32, input_shape=(X_train.shape[1], X_train.shape[2]),
                        kernel_regularizer=keras.regularizers.l2(0.01),
                        activity_regularizer=keras.regularizers.l2(0.1),
                        ))
    # model.add(Dropout(0))
    model.add(Dense(units=y_train.shape[1]))
    # Compile model
    model.compile(loss='mse', optimizer=Adam(learning_rate=0.001))  # De
fault_lr = 0.001
    model.summary()

    return model




def fit_rnn(model):
    early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patie
nce=1000, restore_best_weights=True)
    history = model.fit(X_train, y_train,
                        epochs=36,
                        batch_size=32,
                        validation_data=[X_valid, y_valid],
                        callbacks=[early_stop])
    return history


def fit_lstm(model):
    early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patie
nce=15, restore_best_weights=True)
    history = model.fit(X_train, y_train,
                        epochs=36,
                        batch_size=32,
                        validation_data=[X_valid, y_valid],
                        callbacks=[early_stop])
    return history
```

```
# Create Model
model_rnn = create_rnn()
model_lstm = create_lstm()

# Fit Model
history_rnn = fit_rnn(model_rnn)
history_lstm = fit_lstm(model_lstm)

def plot_model_summary(model, model_name):
    plot_model(model, to_file='model_summary_' + model_name + '.png', sh
ow_shapes=True)


plot_model_summary(model_rnn, 'rnn')

plot_model_summary(model_lstm, 'lstm')

# LOSS CURVE
# Plot train loss and validation loss

def plot_loss(history, model_name):
    plt.plot(history.history['loss'])
    plt.plot(history.history['val_loss'])
    plt.title('Loss curve_' + model_name, fontsize=16, y=1.01)
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend(['Train loss', 'Validation loss'], loc='upper right')
    plt.savefig('Loss curve_' + model_name+ '.png', dpi=1200)
    plt.show()


plot_loss(history_rnn, 'rnn')



plot_loss(history_lstm, 'lstm')

# Make prediction
def prediction(model):
    # print(X_test.shape)
    prediction = model.predict(X_test)
    return prediction


prediction_rnn = prediction(model_rnn)
prediction_lstm = prediction(model_lstm)

scaler = pts['pt_Close']
prediction_rnn[:, :] = scaler.inverse_transform(prediction_rnn[:, :])
y_test[:, :] = scaler.inverse_transform(y_test[:, :])

prediction_lstm[:, :] = scaler.inverse_transform(prediction_lstm[:, :])
```

```
def plot_future(prediction, model_name, y_test):
    plt.figure(figsize=(10, 6))
    range_future = len(prediction)
    plt.plot(np.arange(range_future), np.array(y_test), label='True Futu
re')
    plt.plot(np.arange(range_future), np.array(prediction), label='Predi
ction')
    plt.title('True future vs prediction for ' + model_name)
    plt.legend(loc='upper left')
    plt.xlabel('Time (day)')
    plt.ylabel('Stock Price (€)')
    plt.savefig('Prediction_Evaluation_plot_' + model_name + '.png', dpi
=1200)
    plt.show()


plot_future(prediction_rnn, 'rnn', y_test)



plot_future(prediction_lstm, 'lstm', y_test)
```

Model Evaluation

```
# Define a function to calculate MAE and RSME
def evaluate_prediction(predicted, actual, model_name):
    if step_ahead == 1:
        rsme = np.sqrt((mean_squared_error(predicted, actual)))
        mae = mean_absolute_error(actual, predicted)
        r2 = r2_score(actual, predicted)
        max_err = max_error(actual, predicted)
        print(model_name + ' performance:')
        print('R^2: {:.4f} %'.format(r2 * 100))
        print('Mean Absolute Error: {:.4f}'.format(mae))
        print('Root Mean Square Error: {:.4f}'.format(rsme))
        print('Max_error: {:.4f}'.format(max_err))
        print('')
        return
    else:
        titles = ["RMSE", "MAE", "R^2"]
        # calculate an RMSE score for each day
        # calculate mse
        rmse = np.sqrt(mean_squared_error(predicted, actual, multioutput
='raw_values'))
        mae = mean_absolute_error(predicted, actual, multioutput='raw_va
lues')
        r2 = r2_score(predicted, actual, multioutput='raw_values')
        df_scores = pd.DataFrame(list(zip(rmse, mae, r2)), columns=[f'{x
}' for x in titles])
        df_scores.index += 1

        colors = plt.rcParams["axes.prop_cycle"]()
        a = 1  # number of rows
        b = 3  # number of columns
```

```
        c = 1  # initialize plot counter
        fig = plt.figure(figsize=(15, 6))
        for i in titles:
            plt.subplot(a, b, c)
            plt.title(f'{i}')
            next_colour = next(colors)["color"]
            df_scores[f'{i}'].plot(marker='o', color=next_colour)
            plt.xticks((range(0, df_scores.shape[0] + 1)))
            plt.legend(loc='upper left')
            plt.xlabel('Forecast Range (Day)')
            plt.ylabel(f'{i}')
            c = c + 1

        plt.subplots_adjust(.5)
        fig.suptitle("Evaluation of performances' trend in the multi ste
p forecasted range", fontsize=16, y=1)
        plt.tight_layout()
        # plt.savefig('EvaluationMultiplePrediction_PG.png', dpi=1200)
        plt.show()

        # calculate overall RMSE
        overall_rmse = np.sqrt(mean_squared_error(predicted, actual, mul
tioutput='uniform_average'))
        overall_mae = mean_absolute_error(predicted, actual, multioutput
='uniform_average')
        overall_r2 = r2_score(predicted, actual, multioutput='uniform_av
erage')
        print(model_name + ' performance:')
        print('R^2: {:.4f} %'.format(overall_r2 * 100))
        print('Mean Absolute Error: {:.4f}'.format(overall_mae))
        print('Root Mean Square Error: {:.4f}'.format(overall_rmse))
        print('')
        return


evaluate_prediction(prediction_rnn, y_test, 'rnn')


rnn performance:
R^2: 90.6005 %
Mean Absolute Error: 4.8983
Root Mean Square Error: 5.6266
Max_error: 16.3924


evaluate_prediction(prediction_lstm, y_test, 'lstm')

lstm performance:
R^2: 95.5423 %
Mean Absolute Error: 3.0041
Root Mean Square Error: 3.8748
Max_error: 13.9313
def save_model(model, model_name):
    model.save('./' + model_name + 'model')
```

References

[1] L. Zanotti, “NETWORK DIGITAL 360,” [Online]. Available: https://www.zerounoweb.it/techtarget/searchsecurity/cybersecurity/semplici-consigli-per-rendere-piu-sicuri-i-database/.

[2] G. O. Albano, in *FONDAMENTI DI BASI DI DATI*.

[3] “NETWORK DIGITAL 360,” digital4, [Online]. Available: https://www.digital4.biz/marketing/big-data-e-analytics/sei-regole-d-oro-per-un-data-driven-marketing-di-successo/.

[4] I. ARTIFICIALE, “INTELLIGENZA ARTIFICIALE,” [Online]. Available: http://www.intelligenzaartificiale.it/data-mining/#Principali_campi_di_applicazione.

[5] M. Pighin, in *Sistemi Informativi Aziendali*.

[6] M. Bahga, BIG DATA SCIENCE & ANALYTICS.

[7] Signore, “ANALISI DEL FENOMENO DEI BIG DATA”.

[8] S. Marsland, MACHINE LEARNING, AN ALGORITHMIC PERSPECTIVE.

[9] M. v. Rijmenam, “Medium,” [Online]. Available: https://medium.com/dataseries/7-steps-to-machine-learning-how-to-prepare-for-an-automated-future-78c7918cb35d.

[10] “Towards data science,” [Online]. Available: https://towardsdatascience.com/decision-tree-classification-de64fc4d5aac.

[11] “Research Gate,” [Online]. Available: https://www.researchgate.net/figure/Illustration-of-Gaussian-process-regression-in-one-dimension-for-the-target-test_fig1_327613136.

[12] J. Brownlee, MASTER MACHINE LEARNING ALGORITHMS.

[13] “AI4BUSINESS,” [Online]. Available: https://www.ai4business.it/intelligenza-artificiale/machine-learning-come-potenzia-lanalisi-avanzata-dei-dati/.

[14] MathWorks Applying Supervised Learning.

[15] P. Molino, appunti di Machine Learning v.0.1.2.

[16] “medium.com,” [Online]. Available: https://medium.com/towards-artificial-intelligence/understanding-non-linear-regression-fbef9a396b71.

[17] B. Di Nunzio, BASI DI DATI, PROGETTAZIONE CONCETTUALE, LOGICA E SQL.

[18] [. h.-a.-i.-l.-s.-c.-1.-. introduction-to-deep-learning-d790feb974e2.

[19] Ferreira, F. , Gandomi, A. H. , & Cardoso, R. (2021). Artificial intelligence.

[20] Mittal, A. K. (2012). Application of Neural Networks in Finance and Investment.

[21] Deep Learning di Ian Goodfellow, Yoshua Bengio, Aaron Courville.

[22] "https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time- series-classification-56b1b0a07a57," [Online].

[23] "https://towardsdatascience.com/understanding-neural-networks-what-how-and-why-," [Online].

[24] "https://www.intelligenzaartificiale.it/deep-learning/," [Online].

[25] " https://towardsdatascience.com/estimating-optimal-learning-rate-for-a-deep-neural-," [Online].

[26] P. M. P. ,. P. M. D. Janki Patel, "https://www.jetir.org/papers/JETIRK006164.pdf".

[27] M. G. M. S. &. M. S. Atharva Shah, "Springer Link," [Online]. Available: https://link.springer.com/article/10.1007/s11042-022-12328-x#:~:text=Long%20short%2Dterm%20memory%20(LSTM,of%20stock%20markets%20%5B31%5D..

[28] S. wagavkar, "Medium," [Online]. Available: https://medium.com/analytics-vidhya/correlation-matrix-5e764bcee34.