Universitat Oberta de Catalunya uoc.edu

Detecting relevant events through hotel room prices

Author: Albert Navarro Pérez
Programme of study: Master's Degree in Data Science
Final work area: Data Mining and Machine Learning

Director: Jerónimo Hernández González
Teacher responsible for the subject: Jordi Casas Roma

Submission date: July 2020

FINAL MASTER THESIS DETAILS

Title:	*Detecting relevant events through hotel room prices*
Author:	*Albert Navarro Pérez*
Director:	*Jerónimo Hernández González*
Responsible teacher:	*Jordi Casas Roma*
Submission date:	*07/2020*
Programme:	*Master's degree in Data Science*
Area:	*Data Mining and Machine Learning*
Work language:	*English*
Keywords	*hotel room prices, multivariate time series, supervised learning, events detection*

Abstract:

Year after year a greater number of internet users from any part of the globe use online travel agencies, such as Expedia, TripAdvisor or Booking, for booking any kind of accommodation.
These third-party booking sites usually offer lower prices than direct booking on hotel sites, allow you to easily compare different prices and ratings, and in addition, they are quite easy to use. On their home page, you can search for a hotel just entering your chosen destination and desired dates.

However, it happens quite frequently that booking prices for these accommodations rise or fall significantly from one day to another in a certain location. It is a well-known fact that on weekends theses prices are usually higher than in the midweek, but there can be much more reasons for these unexpected fluctuations.

This Final Master Thesis aims at detecting these anomalous prices and analyse if they are justified by some kind of relevant event that takes place, for the purpose of this study, in the city of Valencia, such as congresses, fairs, concerts, sporting events, or leisure activities between others.

To carry out this project data with booking room prices will be collected making a daily scrap of the hotels on Booking.com, as well as obtaining a list of the most important social events that take place in the city by consulting different sites on the web.

Table of Contents

Table of Abbreviations

Online Travel Agency – OTA
Anomaly Detection – AD
Machine Learning – ML
Time Series Data – TSD
Multivariate Time Series Data – MTSD
K Nearest Neighbour – KNN
Support Vector Machines – SVM
Neural Networks – NN
Long Short-Term Memory Networks – LSTM

1. Introduction

1.1 Context and justification

A recent report from January 2020 [1] reveals that internet users are growing by an average of near one million new users every day. At the beginning of the current year there were 5.19 billion of unique mobile users, and 4.54 billion of internet users, from a total population of 7.75 billion. This means an increase of 2.4% of unique mobile users (124 million new users) and 7% of internet users (298 million new users) compared to January 2019.

In line with this increase, online hotels booking is also booming and growing, with exponential rate as growing the number of online hotels [2]. More number of consumers from any part of the globe use third party sites to hire a flight, a getaway, to rent a car, as well as to make reservations in different types of accommodations such as apartments, cottages and, of course, hotel rooms. These online travel agencies (OTAs) normally offer lower prices than direct booking on the hotel website, allow you to easily compare different rooms rated by stars and by location, and they are very easy to use, since everything is packaged in a neat format so you can find what you are looking for right away [3].

For all of this it seems of great interest to know why sometimes the prices of these accommodations fluctuate considerably. In particular, this project will try to evaluate how some kind of social events such as concerts, congresses, festivals, or sporting events among others, that take place in the city or near it, can influence in this alteration on the hotel room prices.

1.2 Motivation

In such context, between the most important OTAs such as Expedia, TripAdvisor or Booking, this project will focus on Booking hotel prices.

Booking.com [4], a part of Booking Holdings Inc., is the world's leading brand for booking online accommodation reservations, based on room nights booked. At December 31, 2019, it offered accommodation reservation services for approximately 2,580,000 properties in over 230 countries and territories and in over 40 languages, consisting of approximately 460,000 hotels, motels and resorts and approximately 2,120,000 homes, apartments and other unique places to stay [5].

In addition to deal with data from a huge company as Booking.com, this Final Master Thesis (TFM) has a personal motivation, since over the last few years the author has been working on an IT project focused on the revenue accounting of airlines, obtained by selling airline tickets through any sale channel, such as internet, phone, travel agencies, airport offices, etc.

It was then when the author acquired a certain degree of knowledge about the varied factors involved in the fluctuation of the final price of an airline ticket: the type of airline (conventional airline or low cost), the countries where it operates, the dates when the passenger travels or the alliances that an airline has with other companies are some of them.

This final project entails a magnificent complement to this previous knowledge acquired in the airlines scope. Both businesses treat with stays away from home, either for business or for leisure, and this work could show some interesting similarities between the pricing policy in the hospitality and in the airline industry.

1.3 Objectives

The main objective of this paper was to analyse if the rise or fall of hotel room prices at Booking.com could predict if there was or not a relevant event in the city of Valencia in the next day after a check-in date. Focused on this objective we applied some classification machine learning models on our data, in order to evaluate the accuracy of the prediction for each one.

Other secondary goals to carry out during the execution of this project were:

- To analyse if the days in advance when booking a room are relevant in relation to the price.
- To analyse if prices rise or fall in a similar way for hotels with the same number of stars.
- To analyse on which days of the week the prices of hotels are higher.
- To check which days on the time series have less informed prices and analyse the reason.

1.4 Approach and method followed

To carry out this project we followed the methodology shown in the next figure:

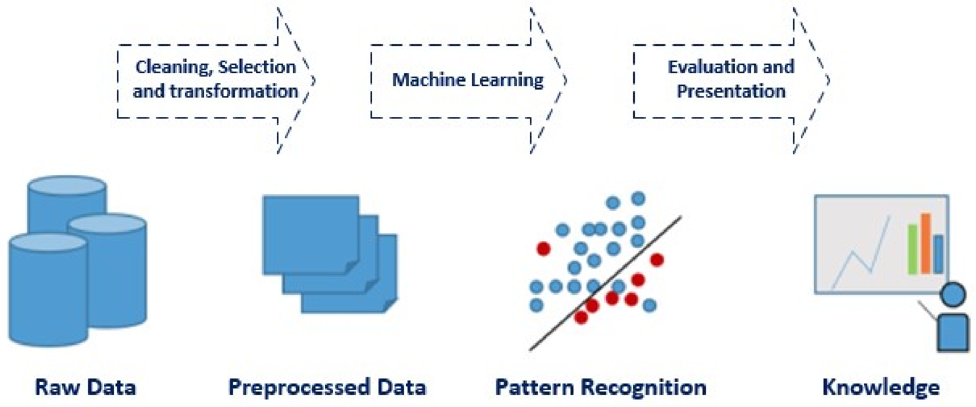


Figure 1 General method followed in the project

- Raw data: stage with unprocessed data. Our data contains all the necessary information to carry out our study.
- Preprocessed data: stage with our data filtered, cleaned, and transformed to be used by some machine learning models.
- Patter recognition: stage where machine learning models are built, trained, tested, and evaluated.
- Knowledge: stage with conclusions about the results obtained.

The resources used to carry out this project were the following:

Hardware	
Laptop	Lenovo ThinkPad T470
Operating System	Windows 10 Pro - 64 bits
Processor	Intel Core i5-7200U
CPU	2.71Ghz
RAM	8Gb

Software		*Version*
Framework	Anaconda Navigator	1.9.6
Development tool	Jupyter Notebook	6.0.3
Programming language	Python 3	3.7.7
Dataset manipulation	Numpy library	1.18.3
	Pandas library	1.0.3
Data visualization	Matplotlib library	3.1.3
	Seaborn library	0.10.0
Data processing & Classifiers & Metrics calculation	Sklearn library	0.22.1

1.5 Work planning

Following the method mentioned above, this project was planned in a Gantt diagram as follows:

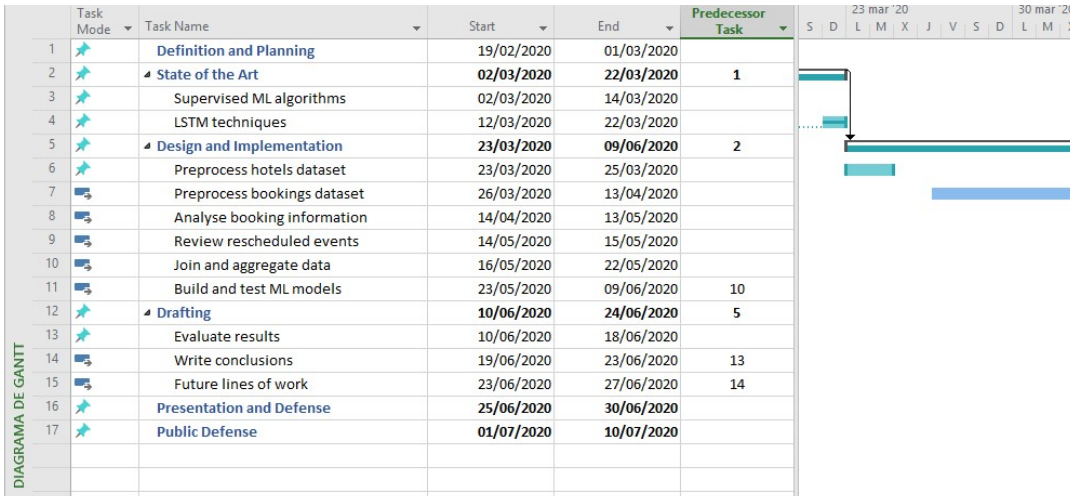


DIAGRAMA DE GANTT

	Task Mode	Task Name	Start	End	Predecessor Task
1		**Definition and Planning**	19/02/2020	01/03/2020	
2		**State of the Art**	**02/03/2020**	**22/03/2020**	**1**
3		Supervised ML algorithms	02/03/2020	14/03/2020	
4		LSTM techniques	12/03/2020	22/03/2020	
5		**Design and Implementation**	**23/03/2020**	**09/06/2020**	**2**
6		Preprocess hotels dataset	23/03/2020	25/03/2020	
7		Preprocess bookings dataset	26/03/2020	13/04/2020	
8		Analyse booking information	14/04/2020	13/05/2020	
9		Review rescheduled events	14/05/2020	15/05/2020	
10		Join and aggregate data	16/05/2020	22/05/2020	
11		Build and test ML models	23/05/2020	09/06/2020	10
12		**Drafting**	**10/06/2020**	**24/06/2020**	**5**
13		Evaluate results	10/06/2020	18/06/2020	
14		Write conclusions	19/06/2020	23/06/2020	13
15		Future lines of work	23/06/2020	27/06/2020	14
16		**Presentation and Defense**	**25/06/2020**	**30/06/2020**	
17		**Public Defense**	**01/07/2020**	**10/07/2020**	

As it can be observed, we have not enumerated any task in relation to the collection of data. The reason is because these tasks were performed during several months before begin the project, as we needed to execute a daily scraping process to get information about bookings.

2. State of the Art

The detection of anomalous events or patterns in data has always been of great interest to ML researchers and practitioners. AD has been applied in the past years in numerous application fields: intrusion detection, system health monitoring, fraud detection in credit card transactions, etc.

Although interpretations vary, an outlier is generally considered to be a data point that is far outside the norm for a variable or population. Events often are points in a time series that can be peaks, level changes, sudden changes of spectral characteristics, etc.

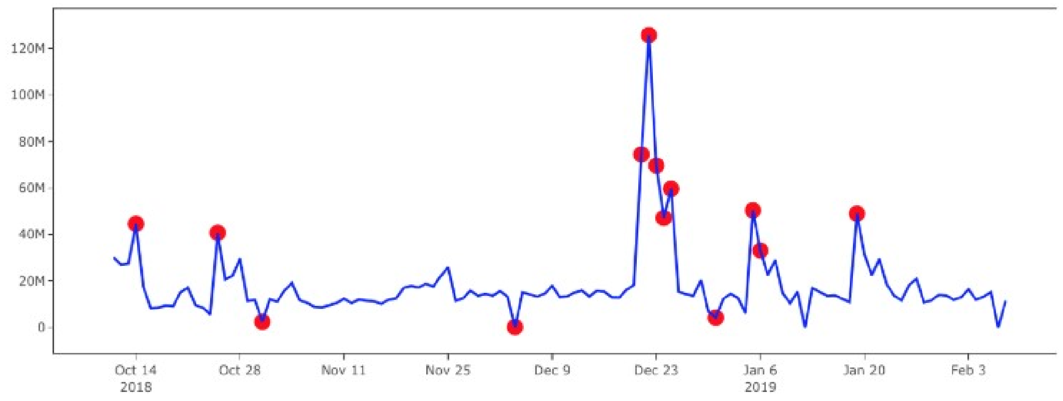


Figure 2 Example of anomaly detection. Obtained from “Anomaly Detection with Isolation Forest & Visualization”, by A.Krishnan (2019)

2.1 Anomaly detection setups

When we talk about AD, there are three main families of possible setups depending on the availability or lack of labelled data points:

- **Supervised AD** where the available data include both labelled training and test datasets. Here, ordinary ML classifiers can be deployed by first training them on the labelled train data and consequently testing them on the validation and test labels. This setting strongly reflects the typical pattern recognition classification from traditional ML problems, with the only exception that the classes are usually rather unbalanced.

- **Semi-supervised** AD where a model of the normal behaviour is learned in the training phase, fitting the AD model on a training set that consists only of normal data points, without anomalies. Outliers are then identified in the test phase as deviations from the learned normal class

- **Unsupervised** AD implies no distinction between the training and the test sets. Such techniques rely solely on the intrinsic characteristics of the data and are therefore the most flexible of all setups because they don't require any labels for training and testing and can be extremely valuable for streaming applications where anomalies are completely unknown.

With this assessment, in our study we focused on classification tasks for predicting events in supervised data, with a discriminative approach which directly maps the observed data to their corresponding label by modelling the posterior distribution.

2.2 Traditional supervised ML algorithms

There is a set of ML algorithms commonly used with supervised datasets in classification problems. Let us briefly mention some of them.

Decision Tree

A decision tree is a tree where each node represents a feature (*attribute*), each link (*branch*) represents a decision (*rule*) and each leaf represents an outcome (*categorical* or *continues* value). The most significant predictor is designated as the root node, splitting is done to form sub-nodes called decision nodes, and the nodes which do not split further are terminal or leaf nodes. The decision tree algorithms will continue running until a stop criterion such as the minimum number of observations etc. is reached.

Using a decision tree, we can visualize the decisions that make it easy to understand and thus it is a popular data mining technique. An example is shown in the next figure:

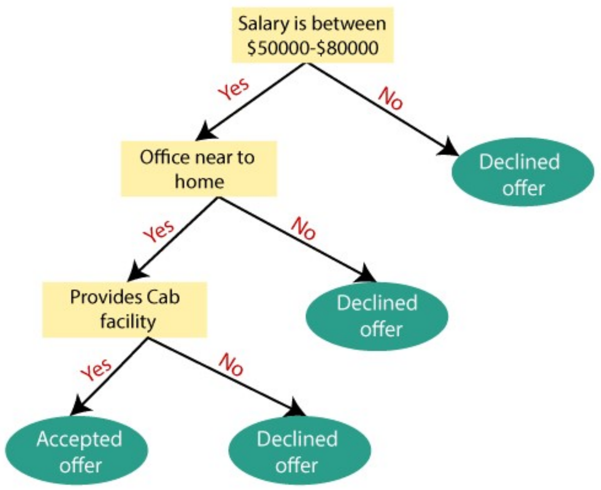


Figure 3 Example of Decision tree classifier. Obtained from "Decision Tree Algorithm Explained with Examples", at https://www.mygreatlearning.com/blog/decision-tree-algorithm/

K Nearest Neighbours

KNN is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms. KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition.
In KNN, K is the number of nearest neighbours. The number of neighbours is the core deciding factor. K is generally an odd number if the number of classes is 2. When K=1, then the algorithm is known as the nearest neighbour algorithm.

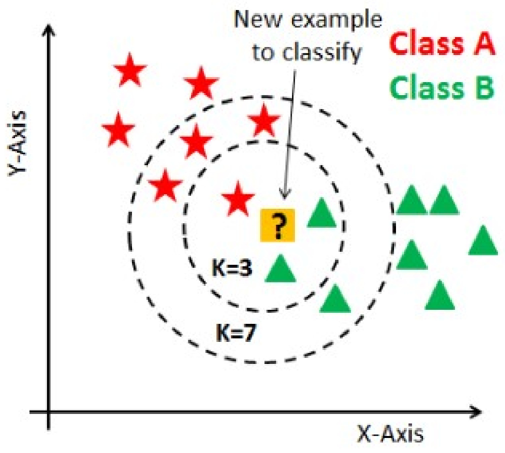


Figure 4 Example of KNN classifier. Obtained from "KNN Classification using Scikit-learn", at https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

Support Vector Machines

Generally, SVM is considered to be a classification approach, but it can be employed in both types of classification and regression problems. It can easily handle multiple continuous and categorical variables.
SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane (MMH) that best divides the dataset into classes.

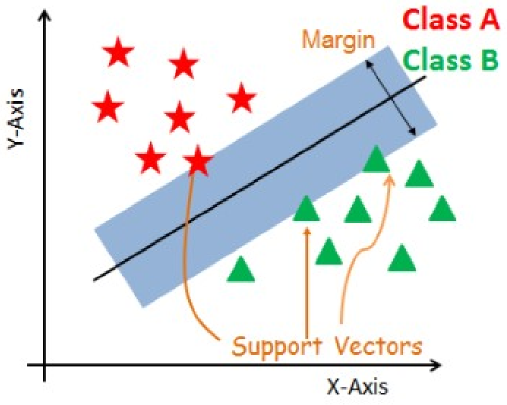


Figure 5 Example of SVM classifier. Obtained from “Support Vector Machines using Scikit-learn”, at https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python

Neural Network

Artificial neural networks are relatively crude electronic networks of neurons based on the neural structure of the brain. They process records one at a time and learn by comparing their classification of the record (i.e., largely arbitrary) with the known actual classification of the record. The errors from the initial classification of the first record is fed back into the network and used to modify the networks algorithm for further iterations.

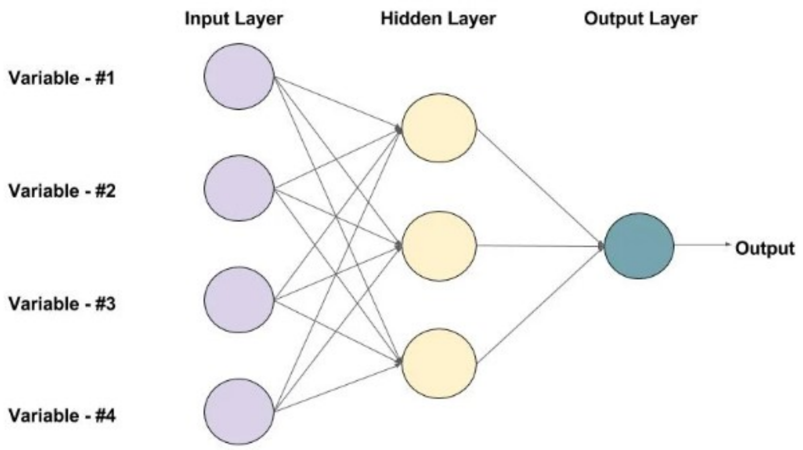


Figure 6 Example of NN with one hidden layer. Obtained from “Understanding Feedforward Neural Networks”, at https://www.learnopencv.com/understanding-feedforward-neural-networks/

Neurons are organized into layers: input, hidden and output.
As shown in the figure, the input nodes which represent the first layer, forward the observed data X by duplicating them to next layer.
The intermediate nodes, include different functions which are connected to each other as a chain structure, applied on the input data.
Since during the feedforward process, the desired labels y to each input data X are not shown, the intermediate nodes are also known as hidden layers. The learning

algorithm decides by minimizing the loss, how to use each layer to get the desired label.
The intermediate computations are designated to be the core of neural networks, since different functions for different purposes are composed together like a connected chain building different layers
The output nodes represent the predicted y denoted by ŷ.

2.3 Long Short-Term Memory Networks

Although in our study, as it will be seen later, our initial MTS were transformed and it was not analysed the time factor, we find relevant to mention the importance of LSTM when working with TS. Anomaly detection and diagnosis with MTS is challenging since it not only requires capturing the temporal dependency in each TS, but also need encode the inter-correlations between different pairs of TS.

Typically, Recurrent Neural Networks (RNN) are not suited to solve problems with learning long-term temporal dependencies because of the vanishing gradient problem. LSTM, an advanced version of RNN, use special blocks in addition to standard blocks. Each block forms a memory cell which consists of three components:

a) an input gate,
b) an output gate and
c) a forget gate.

This set of gates are used to control for how long the incoming information should be maintained in memory:

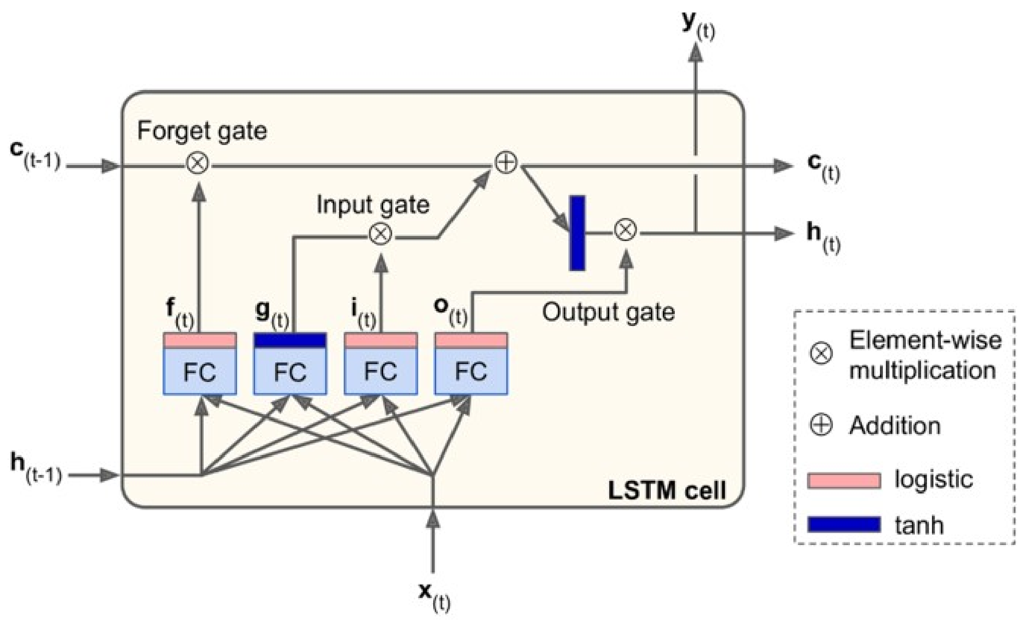


Figure 7 LSTM Cell. Obtained from "Hands-On Machine Learning with Scikit-Learn and TensorFlow" [15]

During the last years, many different LSTM architectures have been developed for working on prediction problems:

- **Vanilla LSTM**: Memory cells of a single LSTM layer are used in a simple network structure.
- **Stacked LSTM**: LSTM layers are stacked one on top of another into deep recurrent neural networks.
- **CNN LSTM**: A Convolutional Neural Network is used to learn features in spatial input and the LSTM is used to support a sequence of inputs (e.g. video of images).
- **Encoder-Decoder LSTM**: One LSTM network encodes input sequences and a separate LSTM network decodes the encoding into an output sequence.
- **Bidirectional LSTM**: Input sequences are presented and learned both forward and backward.
- **Generative LSTM**: LSTMs learn the structure relationship in input sequences so well that they can generate new plausible sequences.

More detail about these architectures can be found in the article of Jason Brownlee about LSTMs [16].

2.4 Related Works

In this section we will detail several approaches that have been carried out related with anomaly detection and anomaly diagnosis in MTS.

In 2015, P. Malhotra et al.[17] proposed an approach using **stacked LSTM** networks, due to stacking recurrent hidden layers in such networks also enables the learning of higher level temporal features, for faster learning with sparser representations. This approach was applied on four real-world datasets, and finally demonstrated that by modelling the normal behaviour of a TS via stacked LSTM networks, they could accurately detect deviations from normal TS behaviour without any pre-specified context window or pre-processing.

This method achieved better or similar results when compared with RNN-AD suggesting that LSTM based prediction models may be more robust compared to RNN based models, especially when we do not know beforehand whether the normal behaviour involves long-term dependencies or not.

In 2017, an end-to-end LSTM-based architecture that outperformed the current state of the art of event forecasting methods on Uber data was carried out by N.Laptev et al. [18]. They introduced some changes in an initial LSTM model. The model first primed the network by auto feature extraction, contrary to the standard feature extraction methods where the features are manually derived. Features vectors were then aggregated via an ensemble technique. The final vector was then concatenated with the new input and fed to LSTM forecaster for prediction.

Model architecture proposed:

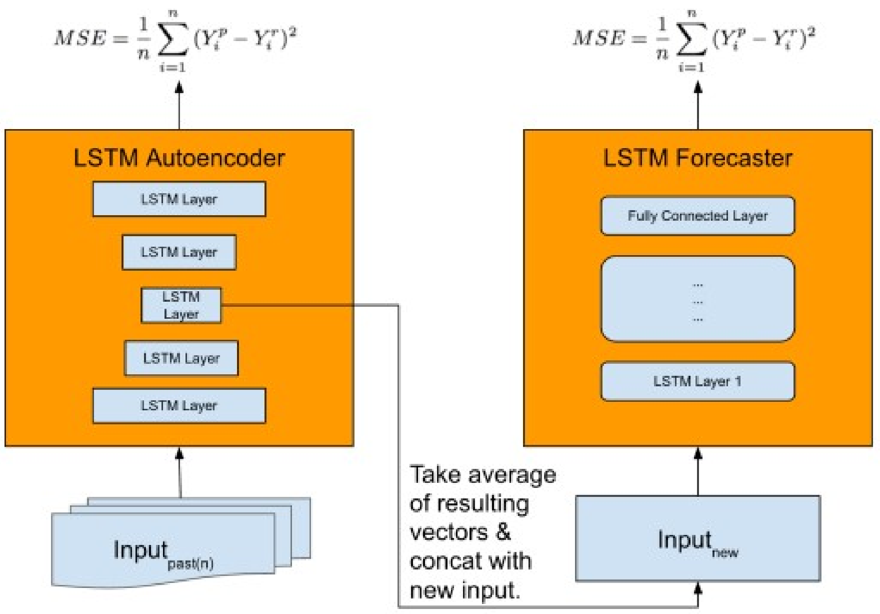

$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i^p - Y_i^r)^2$
$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i^p - Y_i^r)^2$
LSTM Autoencoder
LSTM Layer
LSTM Layer
LSTM Layer
LSTM Layer
LSTM Layer
LSTM Forecaster
Fully Connected Layer
LSTM Layer 1
Take average of resulting vectors & concat with new input.
Input past(n)
Input new

Figure 8 LSTM Architecture. Obtained from "Time-series Extreme Event Forecasting with Neural Networks at Uber", by N.Laptev et al.

Using this approach, they achieved an average 14.09% improvement over the multilayer LSTM model trained over a set of raw inputs.

In 2018, a **Multi-Scale Convolutional Recurrent Encoder-Decoder** (MSCRED), was proposed by C.Zhang et al. [19]
MSCRED was an innovative model, inspired by fully convolutional neural networks (CNN) and convolutional LSTM networks, and able to perform anomaly detection and diagnosis in multivariate time series data:

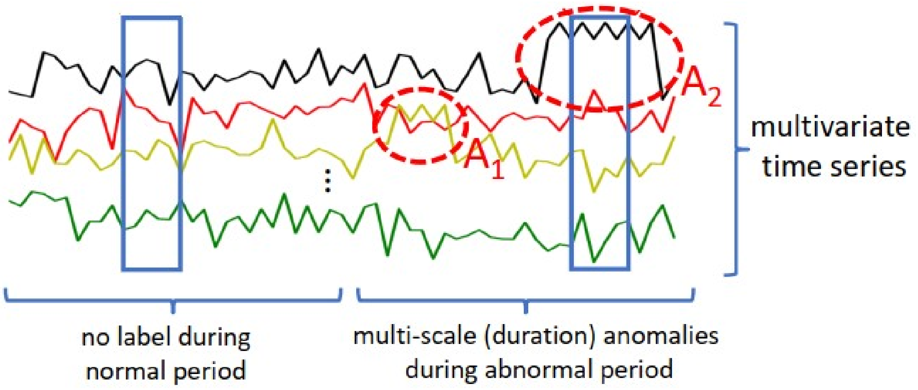


Figure 9 Unsupervised anomaly detection and diagnosis in MTS data. Obtained from "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data", by C.Zhan et al.

In this figure red dash circles A_1 and A_2 illustrate two anomalies. The root causes are yellow and black time series, respectively.

In this study they used a synthetic dataset and a real-world power plant dataset for empirical studies.
Regarding the architecture they generated multi-scale (resolution) system signature matrices. After that they encoded the spatial information in signature matrices via a convolutional encoder and model the temporal information via an attention based ConvLSTM. Finally, they reconstructed signature matrices based upon a convolutional decoder and use a square loss to perform end-to-end learning.
The research suggested that MSCRED was much better than baseline methods as it could model both inter-sensor correlations and temporal patterns of multivariate time series effectively and, compared with AutoRegression Moving Average (ARMA) and LSTM encoder-decoder, MSCRED was more robust to the input noise checked

In another research, in 2019, P.Park et al. [20] proposed an approach that combined an **Autoencoder** to detect a rare fault event and a **LSTM network** to classify different types of faults. The autoencoder was trained with offline normal data, which was then used as the anomaly detection. The predicted faulty data, captured by autoencoder, were put into the LSTM network to identify the types of faults of severely unbalanced data for practical industrial processes. This approach basically combined the strong low-dimensional nonlinear representations of the autoencoder for the rare event detection and the strong TS learning ability of LSTM for the fault diagnosis.

The network started with a sequence input layer of the MTS samples. The autoencoder then analysed the time series data to detect rare events using the concept of anomaly detection. Once a fault was detected, the LSTM network learned the dependencies between various time steps of sequential data to identify the types of faults.

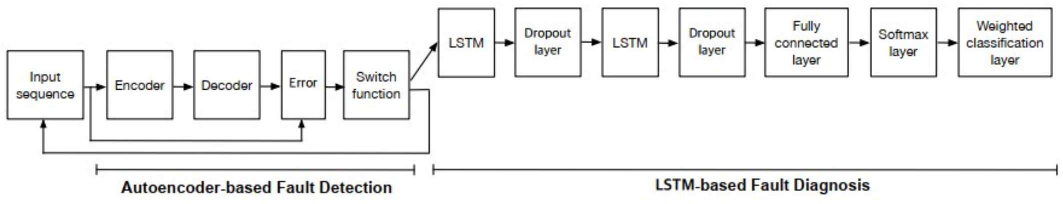


Figure 10 Structure of combined autoencoder and LSTM network for fault detection and fault diagnosis. Obtained from “Fault Detection and Diagnosis Using Combined Autoencoder and Long Short-Term Memory Network”, by P.Park et al.

The proposed approach achieved significant accuracy improvement of 16.9% over the deep convolutional neural network

3. Design and Implementation

In the first part of this chapter we will introduce the design setup for obtaining the datasets used to carry out this project and the content of these information.

As shown in next figure, we visited several web sites to obtain the main **events** that took place in the city of Valencia during the range of dates of the study.

In addition, two different Python programs were implemented to scrap data from Booking.com. The first one scraped data related with hotels (property identifier, district, local code, etc) published in Booking.com while the second one scraped data about room **bookings** available at a certain booking date in that site:

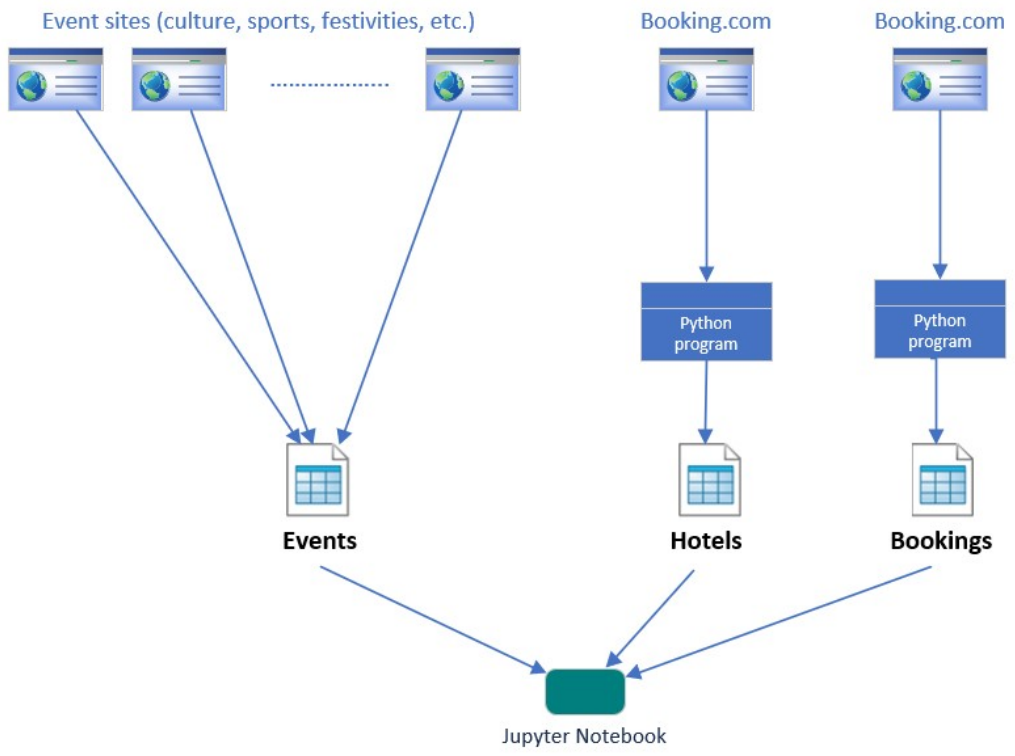


Figure 11 Dataset collection schema

3.1 Events

As mentioned before, our events dataset contained information about the most relevant events celebrated in the city of Valencia during the dates involved in the study [6] - [14].

The kind of events included in this dataset were:

- Music events, like festivals and concerts,
- Sporting events, like football, basketball, tennis and running races,
- Fairs and congresses,

- Local and national festivities,
- “Las Fallas” festival, the most important one in the city.

This dataset was obtained visiting different web sites, and periodically reviewed, as new events could be scheduled, could change its celebration date, or even could be cancelled.

Events information was saved in a unique file with the following features:

Feature	***Description***
date	Date of celebration of the event
event	Event name
place	Place where is hold the event (facility if it is indoor)
city	City where is celebrated the event
category	Category of the event: music, sports, festivity, etc.
subcategory	Subcategory of the event: concert, tennis, local festivity, etc.

Table 1 Features of Events dataset

An example about an event observation is shown in the next table:

Feature	***Value***
date	01/12/2019
event	Maratón Valencia Trinidad Alfonso EDP 2019
place	Valencia
city	Valencia
category	Sports
subcategory	Athletics

Table 2 Example of observation in the Events dataset

3.2 Hotels

The Hotels dataset contained information about all the hotels in Valencia published in the Booking web site.

Hotels information was saved in a unique file with the following features:

Feature	***Description***
propertyId	Internal identifier of the property in the Booking site system
propertyType	Type of the property: always ‘Hotel’
district	Name of the district of the property and the city where it belongs
coords	GPS coordinates inside the city expressed as latitude and longitude
zipcode	Postal code where the property is located

Table 3 Features of Hotels dataset

An example about a hotel observation is shown in the next table:

Feature	***Value***
propertyId	23332
propertyType	Hotel
district	Poblados marítimos, Valencia
coords	-0.323690325021744, 39.4642807560034
zipcode	46011

Table 4 Example of observation in the Hotels dataset

3.3 Bookings

The Bookings dataset contained information about room bookings of all kind of properties from the Booking web site, in its Spanish version, as we wanted to analyse hotel prices in euros.

To this end, a scraping process was implemented to get daily prices for these properties in the city of Valencia. As a final result we got a total of 5 different time series data (TSD):

- TSD #1: containing hotel prices for a 7-days in advance search
- TSD #2: containing hotel prices for a 14-days in advance search
- TSD #3: containing hotel prices for a 30-days in advance search
- TSD #4: containing hotel prices for a 60-days in advance search
- TSD #5: containing hotel prices for a 90-days in advance search

This information was saved in a set of files, each one holding the following features:

Feature	***Description***
propertyId	Internal identifier of the property in the Booking system.
name	Name of the property.
propertyType	Type of the property (hotel, apartment, B&B, etc).
city	City where the property is located in.
rooms	Number of booking rooms.
days	Number of days of stay.
price	Published price in euros of the property. Also referenced as netPrice.
checkIn	Date of arrival and register in the property.
checkOut	Date of leaving the property.
stars	Number of stars of the property.
score	The mean customer review score for the property on a scale out of 10.
advance	Number of days between search date and check in date.

Table 5 Features of Bookings dataset

For the scope of this study, as mentioned before, the scraping search of data were fixed as follows:

- city: only searched the city of ‘Valencia’.
- rooms: set to ‘1’ and only searched for double room.
- days: set to ‘1’ day of stay.
- advance: searched for 7, 14, 30, 60 and 90 days in advance.

With this setup the scraping process generated 5 files per each day, one per every advance search.

As an example, the 9th August 2019 there were collected 5 files:

File name	***File content***
20190809_145522_Booking_Valencia_7	Bookings with 7 days in advance search.
20190809_145534_Booking_Valencia_14	Bookings with 14 days in advance search.
20190809_145553_Booking_Valencia_30	Bookings with 30 days in advance search.
20190809_145611_Booking_Valencia_60	Bookings with 60 days in advance search.
20190809_145628_Booking_Valencia_90	Bookings with 90 days in advance search.

Table 6 Files generated on a particular search date

An example about a booking observation is shown in the next table:

Feature	***Value***
propertyId	91063
name	AC Hotel Valencia
propertyType	Hotel
city	Valencia
rooms	1
days	1
price	€ 102
checkIn	2019-11-08
checkOut	2019-11-09
stars	4
score	8,5
advance	7

Table 7 Example of observation in the Bookings dataset

4. Data Analysis and Processing

4.1 Events dataset

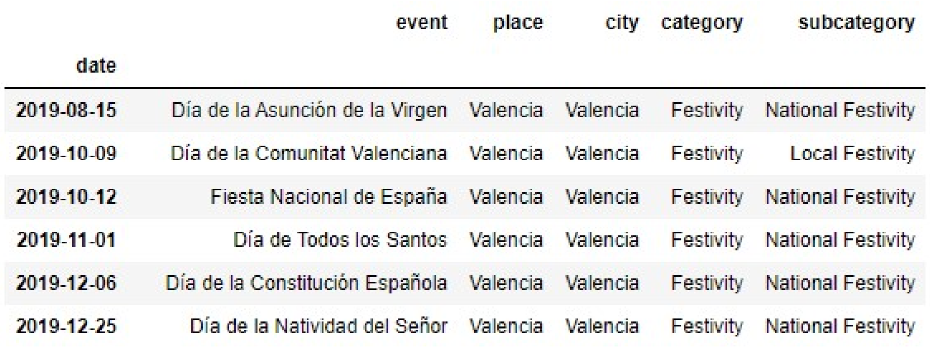
For this study, events in the city of Valencia were collected from the period delimited from **01/08/2019** to **20/03/2020**.

The total number of events collected are shown in the next table:

Year	*# of Events*
2019	131
2020	39
Total	**170**

Table 8 Events collected per year

We can look at an example of some events for the category *Festivity* in the next table:



date	event	place	city	category	subcategory
2019-08-15	Día de la Asunción de la Virgen	Valencia	Valencia	Festivity	National Festivity
2019-10-09	Día de la Comunitat Valenciana	Valencia	Valencia	Festivity	Local Festivity
2019-10-12	Fiesta Nacional de España	Valencia	Valencia	Festivity	National Festivity
2019-11-01	Día de Todos los Santos	Valencia	Valencia	Festivity	National Festivity
2019-12-06	Día de la Constitución Española	Valencia	Valencia	Festivity	National Festivity
2019-12-25	Día de la Natividad del Señor	Valencia	Valencia	Festivity	National Festivity

Table 9 Example of Events for the category *Festivity*

4.2 Hotels dataset

The scraping process imported a total of **95** hotels located in the city of Valencia.

These hotels belong to a total of 11 districts of the city:

1. Benicalap
2. Benimaclet
3. Camins al Grau
4. Campanar
5. Ciutat Vella
6. Eixample
7. El Pla del Real
8. Extramurs
9. Poblados del Oeste
10. Poblados marítimos
11. Quatre Carreres

That is, there were no published hotel rooms for all the districts in the city, such as:

- Algirós

- Jesús
- La Saïdia
- Olivereta
- Patraix
- Rascanya

This is due to the fact that not all the hotels publish its rooms in OTAs such as Booking or Tripadvisor.

4.3 Bookings dataset

There were collected a total of **750** files with all the bookings in the city of Valencia. The scraping search was executed from the period delimited between 25/07/2019 and 21/12/2019.

As a result of this executions there were obtained hotel bookings for the date range between **01/08/2019** and **20/03/2020**. Thus, a total of **233** days were involved in the present analysis.

It is important to notice that the first and the last days of that range did not had so much data available as those in central dates. That is because the days in advance of the search.
As an example, booking data for the 01/08/2019 was only regarding a 7-days in advance search, and booking data for the 20/03/2020 was only regarding a 90-days in advance search.

On the other hand, from all the scraped observations, which included all kind of properties (hotels, B&B, apartments, etc), our study focused on **50939** observations, that were those belonging exclusively to *hotel* bookings.

4.4 Hotels without star

In our analysis we detected a total of **2092** observations in the booking dataset that had not star rating, belonging to 5 hotels.

In this case it was necessary to check in other important OTAs, such as Expedia and TripAdvisor, which star rating had these hotels, and the result was:

Name of the hotel	***Result of investigation***
MYR Hotel Plaza Mercado & Spa	It is a 3-star hotel
Boutique Creative Rooms	It is actually a Bread&Breakfast
Opera House Valencia B&B	It is a 3-star hotel
Reina Rooms	It is a 3-star hotel
Lindala	It is a 3-star hotel

Table 10 Hotels without rewarded star rating

Thus, observations of *Opera House Valencia B&B* were discarded from the present analysis.

After this approach, the total number of hotels involved in the analysis, grouped by star, was:

Hotel rating	***Number of hotels***
1-star	3
2-star	10
3-star	37
4-star	38
5-star	6
Total	**94**

Table 11 Number of hotels by star rating

4.5 Outliers and Abnormal prices

In this section we present the outliers and abnormal prices detected in our booking dataset.

To this end, and as a first step, we examined the distribution of our data through some boxplots, grouped by star rating:

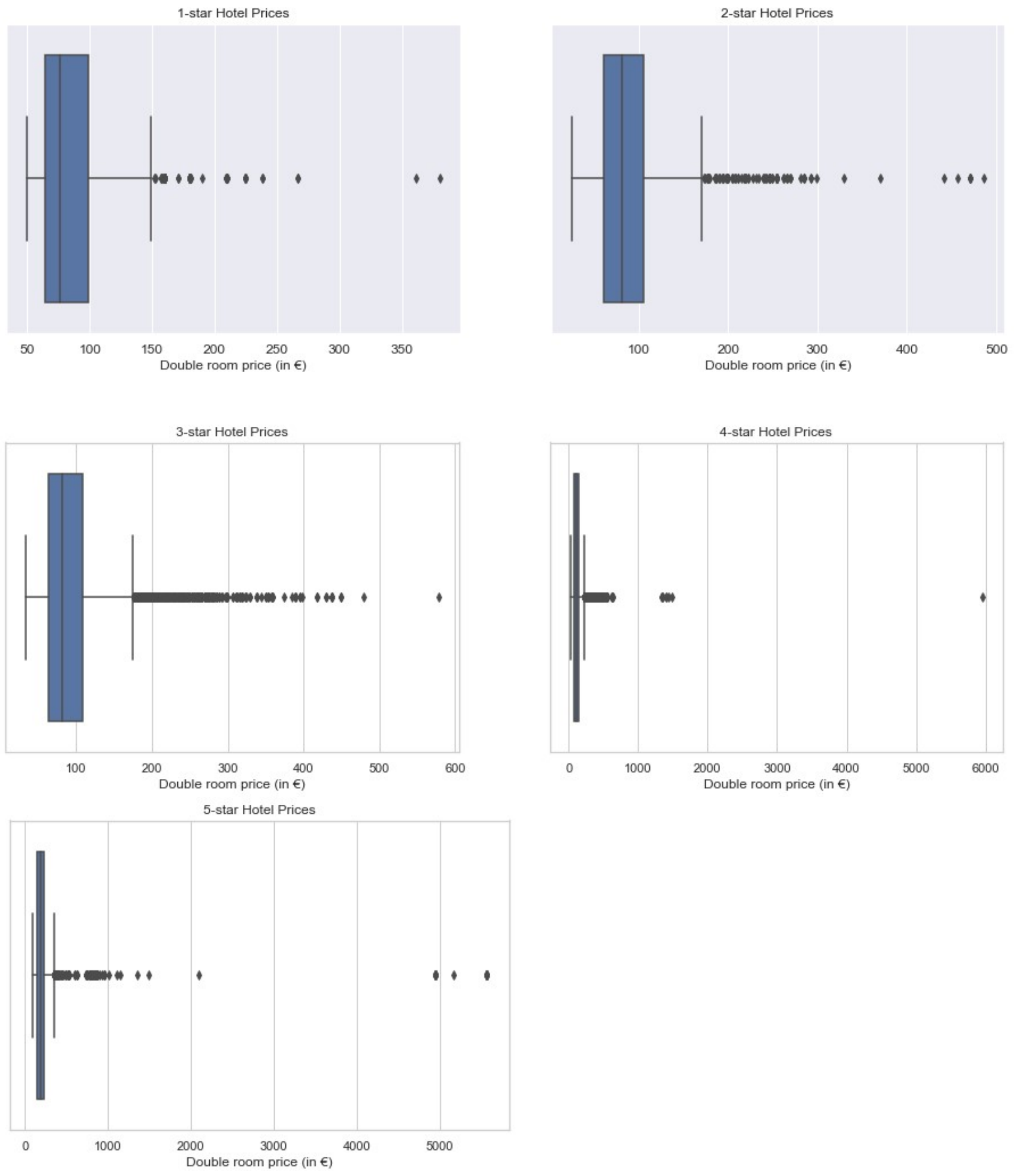


Figure 12 Boxplots with outliers by star

As it can be seen, all star-hotels presented values far from the upper quartile.

For deeper analysis we checked out which observations in each group were very far from the mean. To achieve this, we used the standard score, also called Z-score, which formula is shown in the next figure:

$$z = \frac{x - \mu}{\sigma}$$

Figure 13 Z-score formula

where:

- x was the net price of the current observation
- μ was the mean price of the grouped observations by star rating
- σ was the standard deviation of the price of the grouped observations by star rating

Applying this statistic to our data we got the observations shown hereafter with a high z-score value:

<u>1 star outliers/abnormal values:</u>

name	checkIn	stars	advance	netPrice	weekday	month	zscore
MD Modern Hotel - Jardines	30/11/2019	1-star	60 days	€ 380.00	Saturday	November	8.4866
MD Modern Hotel - Jardines	30/11/2019	1-star	90 days	€ 361.00	Saturday	November	7.9306
MD Modern Hotel - Jardines	16/11/2019	1-star	90 days	€ 266.00	Saturday	November	5.1506
MD Modern Hotel - Jardines	16/11/2019	1-star	60 days	€ 266.00	Saturday	November	5.1506

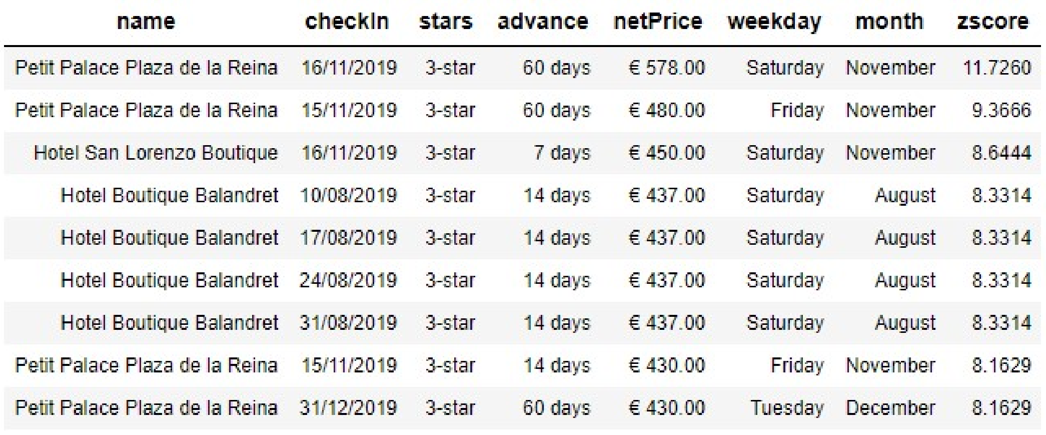
Table 12 1-star outliers/abnormal values

<u>2 star outliers/abnormal values:</u>

name	checkIn	stars	advance	netPrice	weekday	month	zscore
Casual de la Música Valencia	15/09/2019	2-star	7 days	€ 486.00	Sunday	September	10.9396
Casual de la Música Valencia	30/09/2019	2-star	30 days	€ 471.00	Monday	September	10.5272
Casual de la Música Valencia	30/09/2019	2-star	14 days	€ 471.00	Monday	September	10.5272
Casual de la Música Valencia	16/11/2019	2-star	60 days	€ 471.00	Saturday	November	10.5272
Casual de la Música Valencia	01/10/2019	2-star	7 days	€ 442.00	Tuesday	October	9.7299
MD Design Hotel - Portal del Real	18/03/2020	2-star	90 days	€ 370.00	Wednesday	March	7.7504

Table 13 2-star outliers/abnormal values

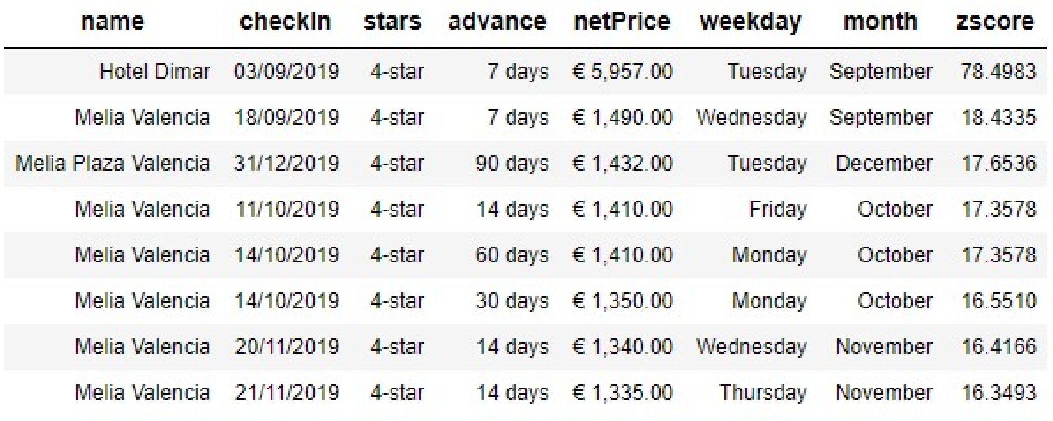
<u>3-star outliers</u>



name	checkIn	stars	advance	netPrice	weekday	month	zscore
Petit Palace Plaza de la Reina	16/11/2019	3-star	60 days	€ 578.00	Saturday	November	11.7260
Petit Palace Plaza de la Reina	15/11/2019	3-star	60 days	€ 480.00	Friday	November	9.3666
Hotel San Lorenzo Boutique	16/11/2019	3-star	7 days	€ 450.00	Saturday	November	8.6444
Hotel Boutique Balandret	10/08/2019	3-star	14 days	€ 437.00	Saturday	August	8.3314
Hotel Boutique Balandret	17/08/2019	3-star	14 days	€ 437.00	Saturday	August	8.3314
Hotel Boutique Balandret	24/08/2019	3-star	14 days	€ 437.00	Saturday	August	8.3314
Hotel Boutique Balandret	31/08/2019	3-star	14 days	€ 437.00	Saturday	August	8.3314
Petit Palace Plaza de la Reina	15/11/2019	3-star	14 days	€ 430.00	Friday	November	8.1629
Petit Palace Plaza de la Reina	31/12/2019	3-star	60 days	€ 430.00	Tuesday	December	8.1629

Table 14 3-star outliers/abnormal values

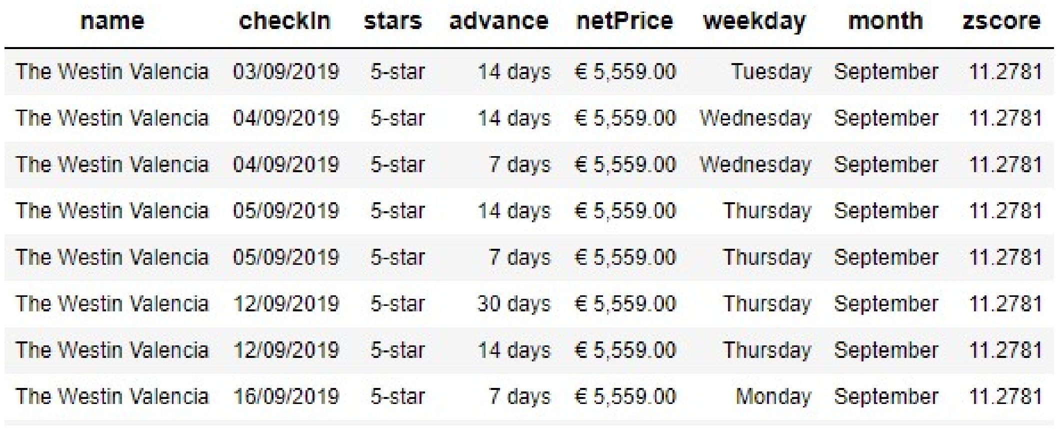
<u>4-star outliers</u>



name	checkIn	stars	advance	netPrice	weekday	month	zscore
Hotel Dimar	03/09/2019	4-star	7 days	€ 5,957.00	Tuesday	September	78.4983
Melia Valencia	18/09/2019	4-star	7 days	€ 1,490.00	Wednesday	September	18.4335
Melia Plaza Valencia	31/12/2019	4-star	90 days	€ 1,432.00	Tuesday	December	17.6536
Melia Valencia	11/10/2019	4-star	14 days	€ 1,410.00	Friday	October	17.3578
Melia Valencia	14/10/2019	4-star	60 days	€ 1,410.00	Monday	October	17.3578
Melia Valencia	14/10/2019	4-star	30 days	€ 1,350.00	Monday	October	16.5510
Melia Valencia	20/11/2019	4-star	14 days	€ 1,340.00	Wednesday	November	16.4166
Melia Valencia	21/11/2019	4-star	14 days	€ 1,335.00	Thursday	November	16.3493

Table 15 4-star outliers/abnormal values

<u>5-star outliers</u>



name	checkIn	stars	advance	netPrice	weekday	month	zscore
The Westin Valencia	03/09/2019	5-star	14 days	€ 5,559.00	Tuesday	September	11.2781
The Westin Valencia	04/09/2019	5-star	14 days	€ 5,559.00	Wednesday	September	11.2781
The Westin Valencia	04/09/2019	5-star	7 days	€ 5,559.00	Wednesday	September	11.2781
The Westin Valencia	05/09/2019	5-star	14 days	€ 5,559.00	Thursday	September	11.2781
The Westin Valencia	05/09/2019	5-star	7 days	€ 5,559.00	Thursday	September	11.2781
The Westin Valencia	12/09/2019	5-star	30 days	€ 5,559.00	Thursday	September	11.2781
The Westin Valencia	12/09/2019	5-star	14 days	€ 5,559.00	Thursday	September	11.2781
The Westin Valencia	16/09/2019	5-star	7 days	€ 5,559.00	Monday	September	11.2781

Table 16 5-star outliers/abnormal values

It was checked out that most of these days the hotels had high prices because there were relevant events in the city of Valencia on the next day after the check-in date, such as:

- 30/11/19: On the next day, 01/12/19, took place 'Maratón Valencia'
- 16/11/19: On the next day, 17/11/19, took place 'Gran Premi Comunitat Valencia' and a fair 'Dos Ruedas + VLC Bike' (Feria Valencia)

But we also found out that in the cases with a high z-score, like in Table 15, when the standard double room in a hotel was unavailable, prices downloaded from Booking belonged to superior rooms. This also happened in Table 16, where "The Westin Valencia" hotel owns several kind of rooms and the prices downloaded referred to the most expensive double room, a 80m^2 suite executive room.

The adopted decision in these cases was to replace the outliers prices by the maximum price of each hotel in all the dataset, as the lack of available rooms is usually associated to a high demand on that date. So it was important for our analysis to take into account these observations.

After this action, the distribution of observations and number of hotels per star of the dataset was the following:

Hotel rating	*Number of observations*	*% of observations*	*Number of hotels*
1-star	1257	2,50	3
2-star	4894	9,73	10
3-star	**19037**	**37,84**	**37**
4-star	**21961**	**43,65**	**38**
5-star	3161	6,28	6
Total	**50310**	**100,00**	**94**

Table 17 Observations per Hotel rating

As we can observe a little more than 80% of our observations belonged to 3 and 4-star hotels, and less than 9% for 1 and 5-star hotels.

4.6 Missing bookings

Deepening into the scraped data, we checked out that Booking did not download any information about a booking when there were no available rooms for a hotel in certain dates.

That was analysed checking the number of hotels scraped by check-in date, grouped by star, as shown in the next figures:

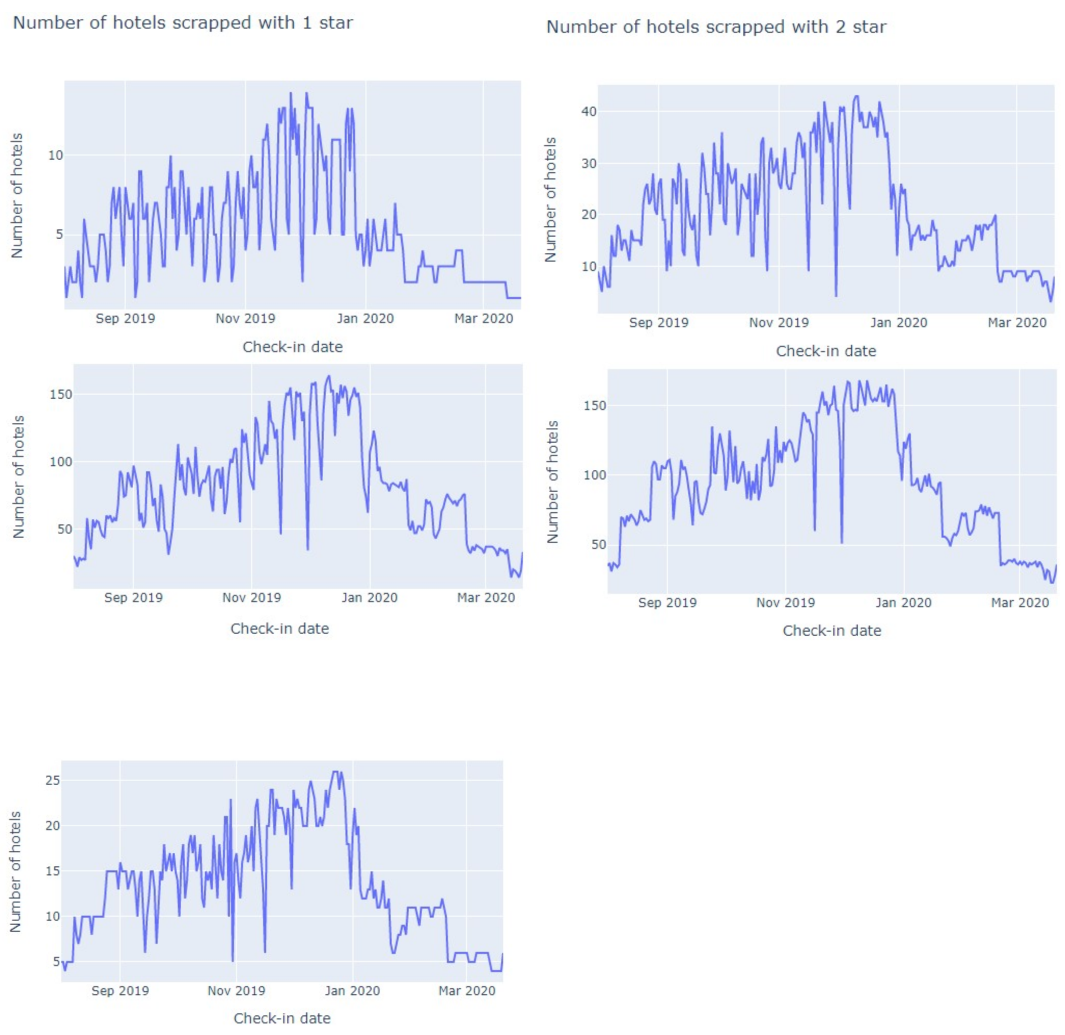


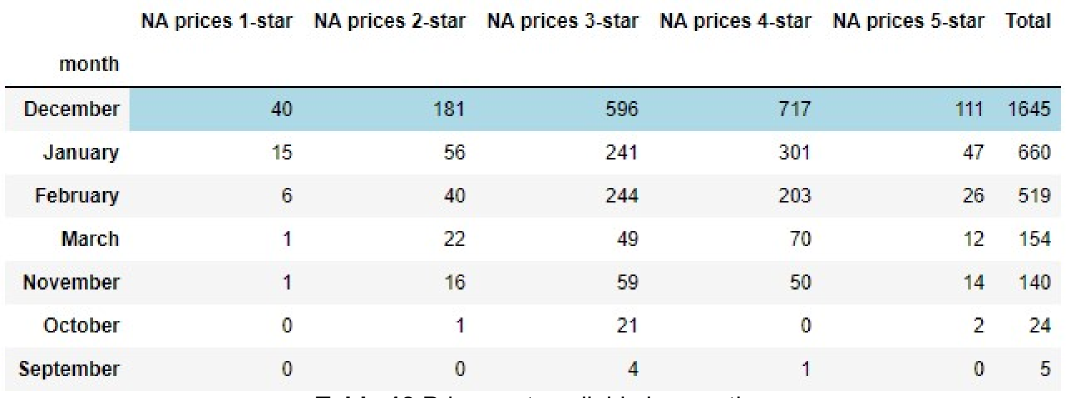
Figure 14 Scraped hotels per check-in date by star

Apart from the initial and ending check-in dates of the figures, where we had few scraped hotels because of the advance search parameter, we detected two low points on 16/11/2019 and 30/11/2019, which matched with the two relevant events mentioned in the previous section.

4.7 Prices not available

A total of **3147** observations in the booking dataset had prices not available (NA) for some hotels. That was a **6,26%** out of the total observations.

In the next table it was analysed the number of these unavailable prices grouped by the month of the check in date:



month	NA prices 1-star	NA prices 2-star	NA prices 3-star	NA prices 4-star	NA prices 5-star	Total
December	40	181	596	717	111	1645
January	15	56	241	301	47	660
February	6	40	244	203	26	519
March	1	22	49	70	12	154
November	1	16	59	50	14	140
October	0	1	21	0	2	24
September	0	0	4	1	0	5

Table 18 Prices not available by month

As it can be seen, December was by far the month with more unavailable prices, followed by January.

The distribution of unavailable prices and its percentage out of the total observations, grouped by star rating, was the following:

Hotel rating	*Number of observations*	*Number of prices NA*	*% of prices NA*
1-star	1257	63	2,00
2-star	4894	316	10,04
3-star	**19037**	**1214**	**38,58**
4-star	**21961**	**1342**	**42,64**
5-star	3161	212	6,74
Total	**50310**	**3147**	**100,00**

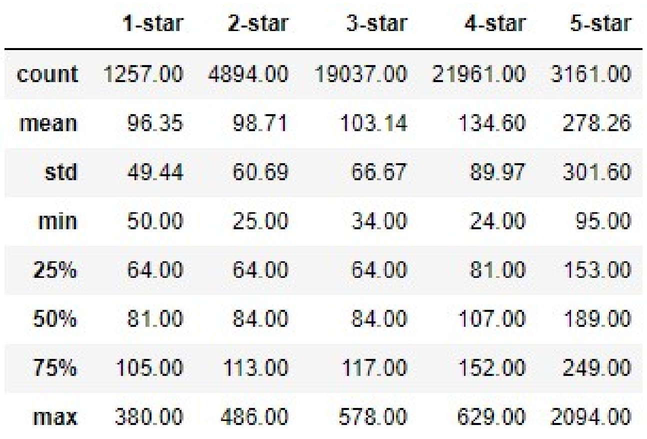
Table 19 Prices not available and its percentage grouped by star

As we can observe, the percentage of prices NA was proportional to the total number of observations per star. The 80% of the unavailable prices proceed from the 3 and 4-star hotels, the ones with more observations. While less than the 9% out of the total proceed form the 1 and 5-star hotels.

It was checked that these prices were unavailable due to the fact that Booking had not any available room for a hotel in that searching day.
At this point the decision was to replace these unavailable prices by the maximum price of each hotel in all the dataset, as aforementioned, the lack of available rooms is usually associated to a high demand on a date.

Summary statistics

After this action, we obtained the next summary statistics:



	1-star	2-star	3-star	4-star	5-star
count	1257.00	4894.00	19037.00	21961.00	3161.00
mean	96.35	98.71	103.14	134.60	278.26
std	49.44	60.69	66.67	89.97	301.60
min	50.00	25.00	34.00	24.00	95.00
25%	64.00	64.00	64.00	81.00	153.00
50%	81.00	84.00	84.00	107.00	189.00
75%	105.00	113.00	117.00	152.00	249.00
max	380.00	486.00	578.00	629.00	2094.00

Table 20 Summary of price statistics per star rating

That is, the mean price of a double room increased according to the star rating: in a 1-star hotel we found the cheapest mean prices while in a 5-star hotel mean prices were much more expensive.
The standard deviation was also lower in hotels with few stars and increased in relation to the hotel category (that is, its number of stars).

Regarding the maximum prices, they also had an increase as the number of stars was higher but, by the other hand, the minimum price was not always in relation to the hotel category. We observed the lowest minimum price in 2-star and 4-star hotels. This could be due to some particular hotels, that may have more aggressive policies of prices when there is a low demand of rooms.

4.8 Bookings over advance search

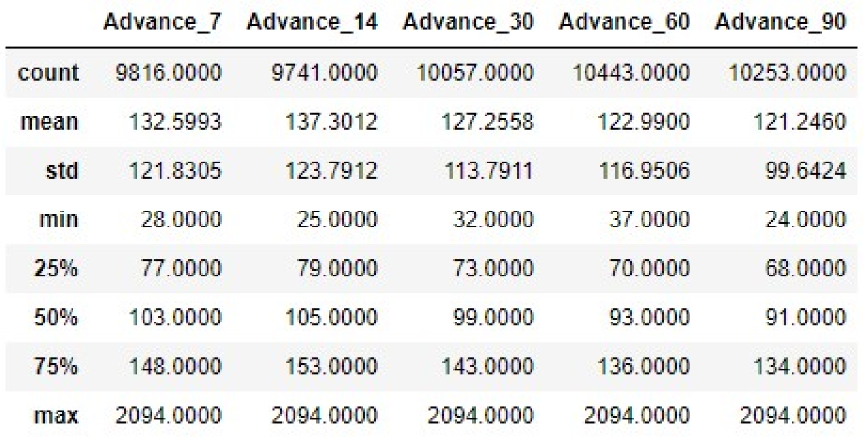
The distribution of observations in relation to the days of advance on the booking search was the following:

Days in advance	*Number of observations*	*% of observations*
7	9816	19,51
14	9741	19,36
30	10057	19,99
60	10443	20,76
90	10253	20,38
Total	**50310**	**100,00**

Table 21 Observations per Days in advance

We can look that observations were equally distributed, all of them between 19 and 21 percent. These data proved that hotels publish a very similar number of rooms for bookings between 7 and 90 days in advance.

The summary statistics about all prices in relation with this advance search is shown in the next table:

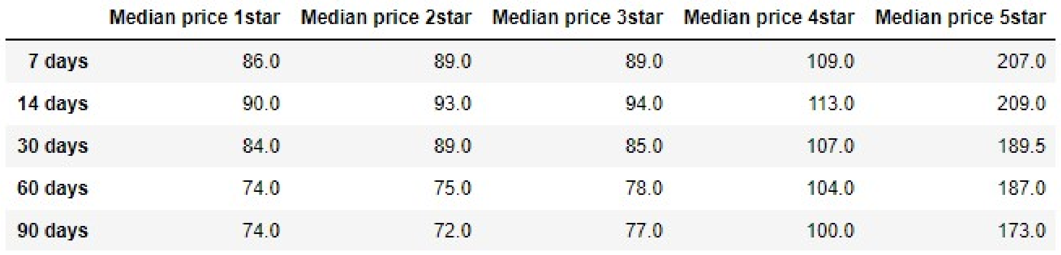


	Advance_7	Advance_14	Advance_30	Advance_60	Advance_90
count	9816.0000	9741.0000	10057.0000	10443.0000	10253.0000
mean	132.5993	137.3012	127.2558	122.9900	121.2460
std	121.8305	123.7912	113.7911	116.9506	99.6424
min	28.0000	25.0000	32.0000	37.0000	24.0000
25%	77.0000	79.0000	73.0000	70.0000	68.0000
50%	103.0000	105.0000	99.0000	93.0000	91.0000
75%	148.0000	153.0000	143.0000	136.0000	134.0000
max	2094.0000	2094.0000	2094.0000	2094.0000	2094.0000

Table 22 Summary of price statistics per advance search

We can observe that mean and median price slightly decreased as the number of advance days of search was higher except in the case of 14 in advance search.

The evolution of the daily median price of a double room per each star rating, depending on the advance search was the following:



	Median price 1star	Median price 2star	Median price 3star	Median price 4star	Median price 5star
7 days	86.0	89.0	89.0	109.0	207.0
14 days	90.0	93.0	94.0	113.0	209.0
30 days	84.0	89.0	85.0	107.0	189.5
60 days	74.0	75.0	78.0	104.0	187.0
90 days	74.0	72.0	77.0	100.0	173.0

Table 23 Median price over advance search

We can visualize it graphically in the next figure:

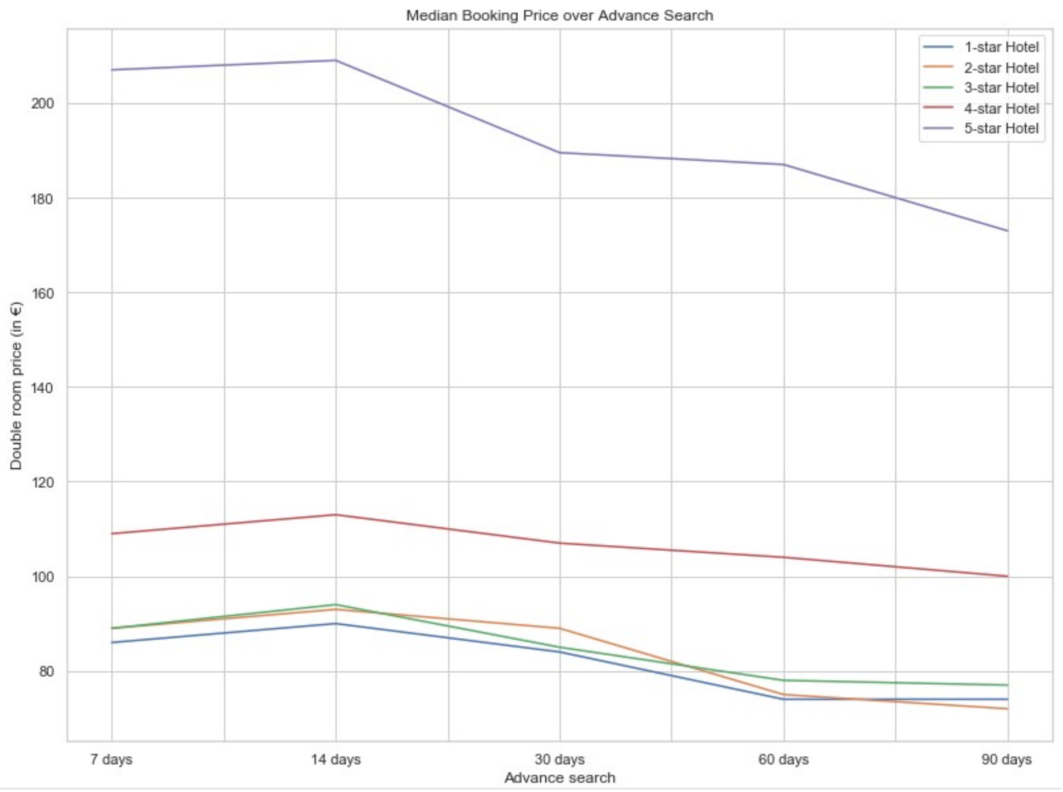


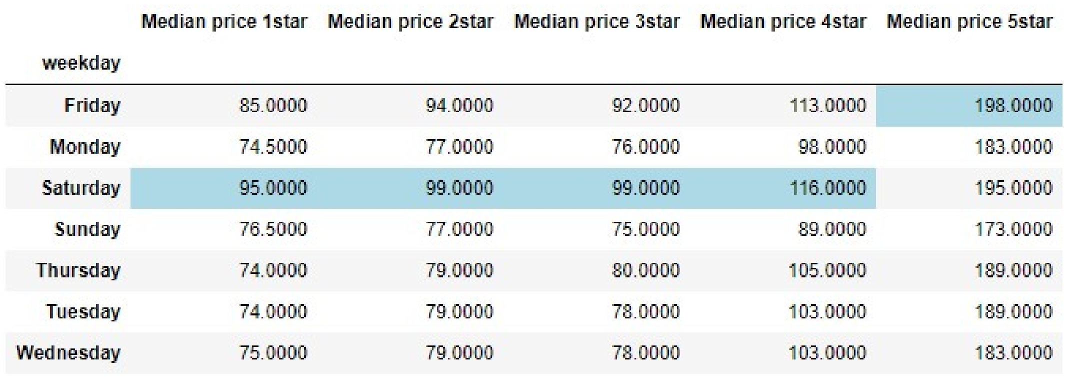
Figure 15 Median booking price over advance search grouped by star rating

As we can see, the trend for all stars were the same: the median price decreased as the customer booked with more days in advance. That is, bookings with 90 days in advance were cheaper than those made with 60 days, bookings with 60 days in advance were cheaper than those made with 30 days and so on.
It seems logical, as hotels try to complete its rooms as soon as possible in order to get capital to hold its activity. Moreover, the usual policy hotels apply is the more rooms are available the better price they offer.

As aforementioned, 14 days in advance prices were slightly higher than those made with 7 days in advance for all the stars. This could be due to the fact that hotels try to complete the rooms that have not been booked when the booking date is already very close. It is always better to have a booked room with a low price than have an unsold room.

4.9 Prices over weekday

For our study, we were also interested in seeing the behaviour of the prices depending on the day of the week, as hotels normally increase their fares at the weekend:



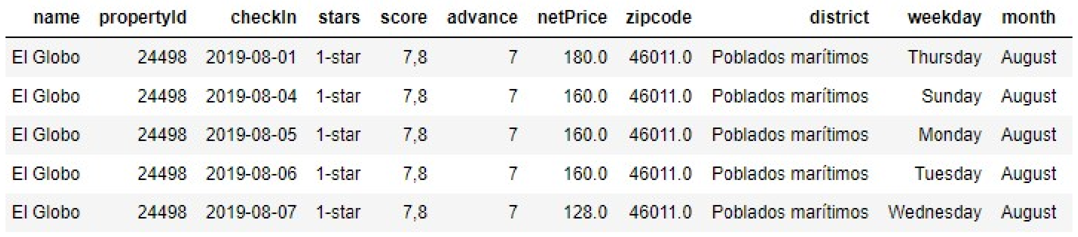
weekday	Median price 1star	Median price 2star	Median price 3star	Median price 4star	Median price 5star
Friday	85.0000	94.0000	92.0000	113.0000	198.0000
Monday	74.5000	77.0000	76.0000	98.0000	183.0000
Saturday	95.0000	99.0000	99.0000	116.0000	195.0000
Sunday	76.5000	77.0000	75.0000	89.0000	173.0000
Thursday	74.0000	79.0000	80.0000	105.0000	189.0000
Tuesday	74.0000	79.0000	78.0000	103.0000	189.0000
Wednesday	75.0000	79.0000	78.0000	103.0000	183.0000

Table 24 Median price over weekday grouped by star rating

As it can be observed, hotels with 1, 2, 3 and 4 stars had their most expensive price on Saturday. The unique exception was for 5-star hotels, where the price was slightly higher on Fridays than in Saturdays. That result confirmed that hotels use to increase their fares at weekends.

4.10 Data selection

Once we did all that previous analysis we obtained a dataset with all the observations prepared for next steps. An extract of these dataset is shown in the next table:



name	propertyId	checkIn	stars	score	advance	netPrice	zipcode	district	weekday	month
El Globo	24498	2019-08-01	1-star	7,8	7	180.0	46011.0	Poblados marítimos	Thursday	August
El Globo	24498	2019-08-04	1-star	7,8	7	160.0	46011.0	Poblados marítimos	Sunday	August
El Globo	24498	2019-08-05	1-star	7,8	7	160.0	46011.0	Poblados marítimos	Monday	August
El Globo	24498	2019-08-06	1-star	7,8	7	160.0	46011.0	Poblados marítimos	Tuesday	August
El Globo	24498	2019-08-07	1-star	7,8	7	128.0	46011.0	Poblados marítimos	Wednesday	August

Table 25 Example of observations

As it can be seen we had a total of 10 available features ready for being analysed. We discarded the **properyId** feature as did not add useful value because this is an internal identifier used by Booking.com.

At this point, and for the scope of this study we only took into consideration the next 4 features:

- **checkIn**
- **stars**
- **advance** (in days)
- **netPrice** (in euros)

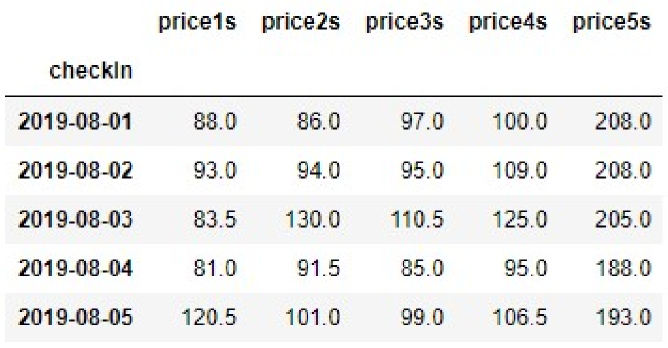
As it will be mentioned in the section Future Work, the rest of features will be available for future deeper analysis.

4.11 Data manipulation

We had to manipulate our data as we wanted a unique price per star rating in a certain check-in date, to be used later with some machine learning models.

To get this final dataset we used the median price grouped by check-in date, days of advance search and star rating.
We decided to use the median price instead of the mean price because very high or very low prices in certain hotels could led to wrong estimated prices.
Let us assume cases in which a 4-star hotel had very high prices in certain dates. As the number of observations of this category represented the 40 percent of the total observations, thus a high percentage, these prices increased considerably the mean price. Using the median price, we achieved to avoid this situation.

After this transformation, the final dataset we obtained was as the following (showing the first 5 observations):



	price1s	price2s	price3s	price4s	price5s
checkIn					
2019-08-01	88.0	86.0	97.0	100.0	208.0
2019-08-02	93.0	94.0	95.0	109.0	208.0
2019-08-03	83.5	130.0	110.5	125.0	205.0
2019-08-04	81.0	91.5	85.0	95.0	188.0
2019-08-05	120.5	101.0	99.0	106.5	193.0

Table 26 Final dataset

where:

- price1s: median price of all 1-star hotels
- price2s: median price of all 2-star hotels
- price3s: median price of all 3-star hotels
- price4s: median price of all 4-star hotels
- price5s: median price of all 5-star hotels

That is, these 5 features were the predictors applied to some machine learning models in order to predict relevant events according to these prices.

The evolution of these prices for the range of dates of our study is shown in the next figure:

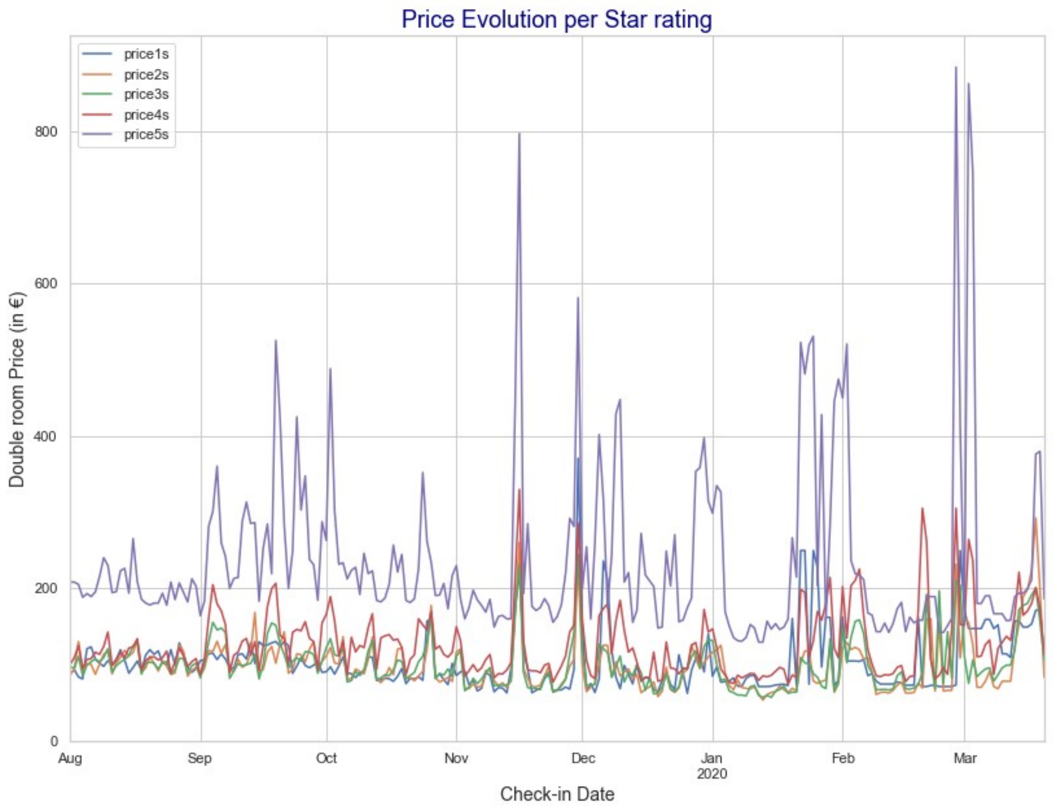
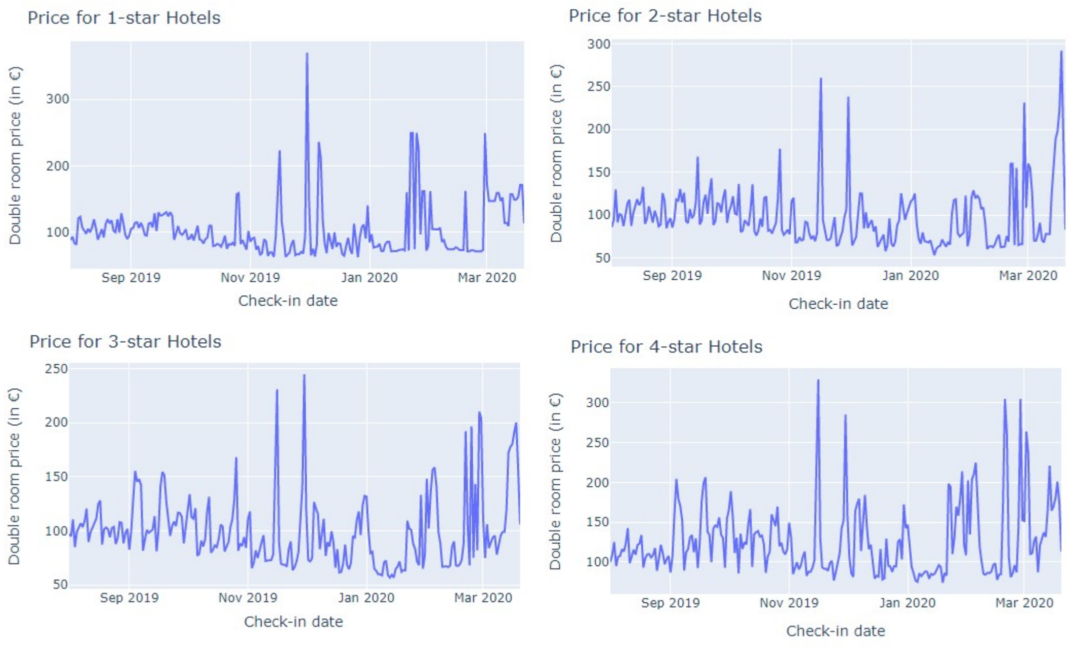


Figure 16 Price evolution for all star rating

And if we see this evolution in separated figures, one per star rating:



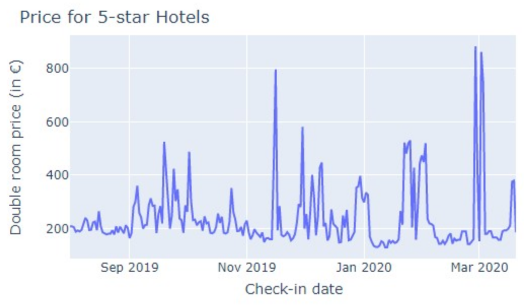


Figure 17 Price evolution per star rating

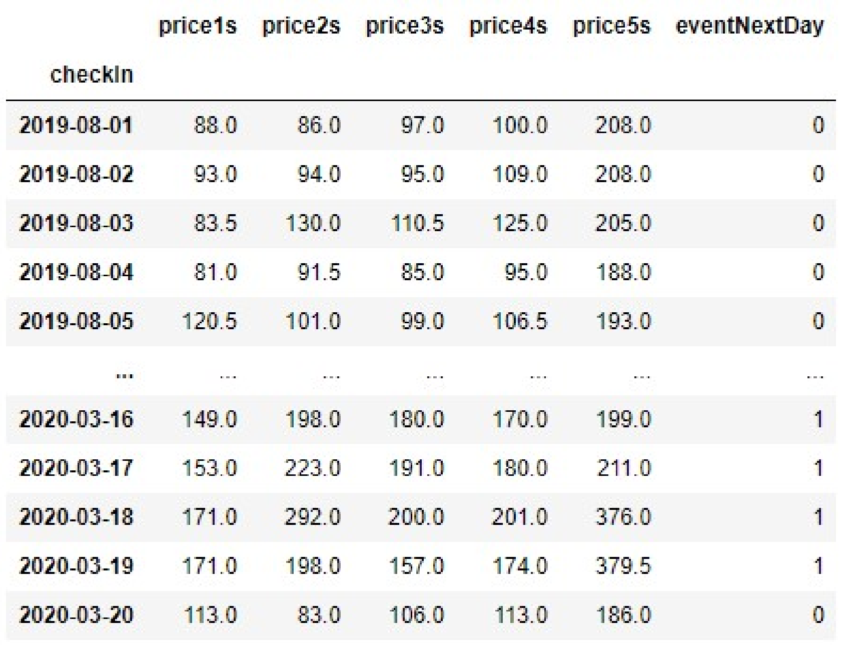
We can observe some check-in dates where the price is quite high in all the figures. The two dates mentioned before, 16/11/2019 and 30/11/2019, with important events in Valencia, and several days in March, when it took place the important festival 'Las Fallas de Valencia'.

4.12 Final data

Before applying some ML models to our dataset, the last step was to join our predictors (that is, each price per star rating and check-in date) to a class label, which we named *eventNextDay*, which had two possible values:

- 0: indicated there was not an event the day immediately after to a check-in date.
- 1: indicated there was an event the day immediately after to a check-in date.

This join gave as a result the next table:



	price1s	price2s	price3s	price4s	price5s	eventNextDay
checkIn						
2019-08-01	88.0	86.0	97.0	100.0	208.0	0
2019-08-02	93.0	94.0	95.0	109.0	208.0	0
2019-08-03	83.5	130.0	110.5	125.0	205.0	0
2019-08-04	81.0	91.5	85.0	95.0	188.0	0
2019-08-05	120.5	101.0	99.0	106.5	193.0	0
...	...	...	...	...	...	...
2020-03-16	149.0	198.0	180.0	170.0	199.0	1
2020-03-17	153.0	223.0	191.0	180.0	211.0	1
2020-03-18	171.0	292.0	200.0	201.0	376.0	1
2020-03-19	171.0	198.0	157.0	174.0	379.5	1
2020-03-20	113.0	83.0	106.0	113.0	186.0	0

Table 27 Final dataset: predictors and class label

Therefore, our class label was a binary class.

4.13 ML models

At this point we applied several binary classification models to our data, as we had a target class with only two possible values.

In the next figure we show the general process followed to train and test these ML models:

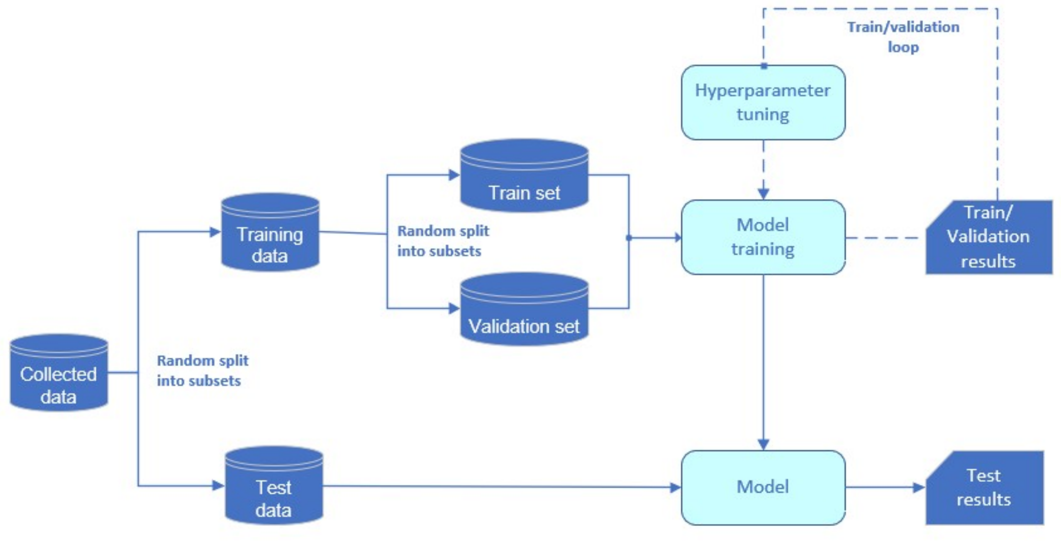


Figure 18 Process Supervised ML Models

1. First, the collected data was divided into two subsets of randomly selected training and test instances. To avoid the problem of underfitting, the number of instances in the training data was much higher than the test data.

2. In the learning phase, the training data was in turn divided randomly into the two sets of train and validation data. First the models were trained on the train set that aimed to fit the data best.

3. In the validation phase, we validated the reliability of the fitted models with its learned weights on the remaining validation set. In this phase, we used the data to tune the parameters to find a model that generalized.

4. Finally, in the testing phase, in order to confirm the performance of the fitted model, we evaluated the accuracy of each model based on the remaining test data which can be interpreted as unseen or new data.

Previously to build the classification models we checked out if our data was or not balanced, meaning a split between approximately 50/50 of true and negative observations, and this was the result:

Days analysed	*Days with event next day*	*% of days with event next day*	*Days without event next day*	*% of days without event next day*
233	110	47,21	123	52,78%

Table 28 Distribution of days with events

With these percentages our data was quite balanced.

To evaluate our data we used some of the most popular models for binary classification. These models were:

- K-Nearest Neighbors
- Support Vector Machines
- Neural Networks
- Decision Trees

When working on these models and in order to find the optimal hyperparameters for each one and avoid the *overfitting* problem (that is, a model tightly fitted to the training data), we used the *grid search* technique with *K-fold cross-validation*, as shown in the next flowchart:

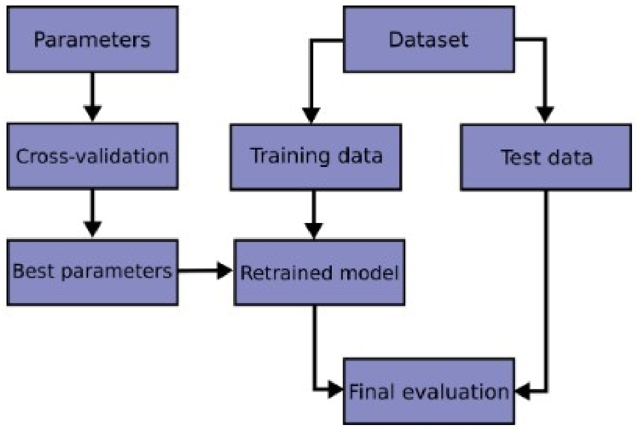


Figure 19 Typical cross validation workflow in model training.
Obtained from Scikit-Learn documentation [21]

The best hyperparameters found for each model, using a **10-fold** cross-validation, were the following:

Model	*Best hyperparameters*
K-Nearest Neighbors	n_neighbors: 3 weights: uniform metric: euclidean
Support Vector Machine	kernel: linear C: 0.1 gamma: 0.001
Neural Network	hidden layers: 1 epochs: 100
Decision Tree (CART)	criterion: entropy max_depth: 2

Table 29 Best hyperparameters for each classification model

5. Results

Classification accuracy is a popular metric used to evaluate the performance of a model based on the predicted class labels.

Accuracy responds to the question '*What percent of predictions were correct calculated*', and its formula is as follows:

Accuracy = (TP + TN) / (TP + TN + FP + FN)

where:

- TP are True Positives: instances where the model correctly predicts the positive class.
- TN are True Negatives: instances where the model correctly predicts the negative class.
- FP are False Positives: instances where the model incorrectly predicts the positive class when it is actually negative.
- FN are False Negatives: instances where the model incorrectly predicts the negative class when it is actually positive.

Therefore, applying these concepts to our study, the meaning would be:

- **TP**: instances where there was a relevant event the next day to a check-in date and the model predicted that effectively there was.
- **TN**: instances where there was not a relevant event the next day to a check-in date and the model predicted that effectively there was not.
- **FP**: instances where there was not a relevant event the next day to a check-in date and the model predicted that there was.
- **FN**: instances where there was a relevant event the next day to a check-in date and the model predicted that there was not.

With this brief introduction to this metric, the accuracy obtained on the models mentioned before can be seen in the next table:

Model	*Accuracy (%)*
K-Nearest Neighbors	70,21
Support Vector Machine	74,47
Neural Network	72,34
Decision Tree (CART)	78,72

Table 30 Accuracy of each binary classification model

Thus, all four models had an accuracy between 70 and 79 percent. *Decision Tree* model was the one with the best accuracy, a **78,72%**, so from now on we will deepen in more details about this model.

A more detailed prediction result of the class label can be summarized in a **confusion matrix**, that allow us to compare predicted values with actual values for each combination in our binary class.

Next figure shows the confusion matrix of the Decision Tree model, in percentage values:

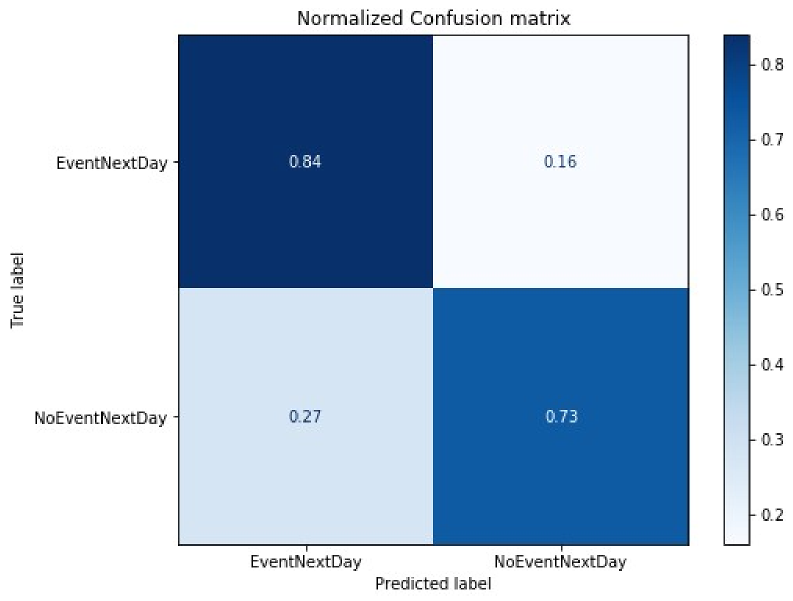


Figure 20 Confusion Matrix of Decision Tree model

where class labels indicated:

- *EventNextDay*: there was a relevant event the day immediately after to a check-in date.
- *NoEventNextDay***:** there was not a relevant event the day immediately after to a check-in date.

Looking at the percentage values of the figure we observed that:

- **TP**: in 84% of the instances there was a relevant event the next day to a check-in date and the model predicted that effectively there was.
- **TN**: in 73% of the instances there was not a relevant event the next day to a check-in date and the model predicted that effectively there was not.
- **FP**: in 27% of the instances there was not a relevant event the next day to a check-in date and the model predicted that there was.
- **FN**: in 16% of the instances there was a relevant event the next day to a check-in date and the model predicted that there was not.

Therefore our model had a higher percentage of FP instances than FN instances, in particular, a 11 percent more. We can interpret this as our model predicted worse when there were not relevant events in the city than when there were.

Although hotels usually increase their prices when there is a near relevant event, they could also increase these prices for other reasons. Let us suppose the case of hotels with a high star rating, such as 4 and 5-star, which can eventually host special events in their facilities. As these hotels charge highest prices this could impact significantly to the aggregated final price for that check-in date and, consequently, lead our model to make a wrong prediction.

We can find other reasons related with the own hotel management. For instance, it could be a public holiday on a Thursday and, assuming that a lot of people will take a day off on Friday to have a total of 4 day holidays between Thursday and Sunday, hotels could expect an important number of bookings and, therefore, increase once again their prices due to the high demand.

In addition to accuracy, other alternative performance metrics can be used, as reporting the classification accuracy may be misleading. This is because with accuracy we only evaluated the model respect the correctly predicted observations.

For our study some questions arise, such as what is more significant, when our model predicts a relevant event and this is not true, or when our model predicts that there is not a relevant event and actually there is?
It seems that is worse to predict an event when there is actually not, because clients would pay a higher price for a room without reasonable cause.

Trying to look into these questions we used some other metrics such as *Precision*, *Recall* and the *F1 score*.

Precision responds to the question '*What percent of positive predictions were correct?'*. Thus Precision is the ability of our classifier not to label an instance positive that is actually negative. Its formula is as follows:

$$\text{Precision} = TP / (TP + FP)$$

As shown in the formula, Precision is tied to False Positives.

Recall or Sensitivity responds to the question '*What percent of positive cases our model captured?*'. Thus Recall is the ability of our classifier to find all positive instances. Its formula is as follows:

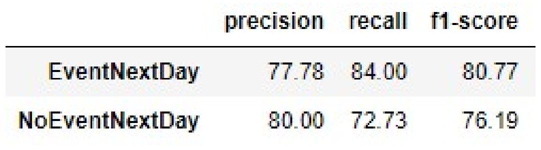
$$\text{Recall} = TP / (TP + FN)$$

As shown in the formula, Recall is tied to False Negatives.

Finally, **F1 score** is the harmonic mean of *Precision* and *Recall*, and its formula is as follows:

$$\text{F1 score} = 2 / (1/\text{Precision} + 1/\text{Recall})$$

The result of these three metrics over our Decision Tree classifier are shown through the following classification report:



	precision	recall	f1-score
EventNextDay	77.78	84.00	80.77
NoEventNextDay	80.00	72.73	76.19

Table 31 Decision Tree model. Metrics per class label

As we can see, for the Precision metric:

- Our model correctly predicted near 78% of instances when there was a relevant event.
- Our model correctly predicted 80% of instances when there was not a relevant event.

As we can see, both of them had a very similar percentage.

And for the Recall metric:

- Our model caught 84% of the instances when there was a relevant event.
- Our model caught near 73% of the instances when there was not a relevant event.

Therefore, at Recall we got a difference of about 11% between both of them. Our model caught more instances with a relevant event the next day to a check-in date.

As mentioned before, we considered worse to predict a relevant event when there is actually not, that is the False Positives cases. So we got an acceptable Precision between 78 and 80 percent of right predictions.

In terms of F1-score we got better percentage in the EventNextDay class label than in the NoEventNextDay class, 80,77% percent against a 76,19%, but this is mainly due to the high percentage of the Recall metric in the EventNextDat class compared with the low percentage in the NoEventNextDay class obtained in the same metric.

6. Difficulties

During the execution of the project there have been numerous difficulties. Let us numerate some of them.

In reference to the events dataset, it was quite difficult to collect all the relevant events in our city, as there was not a unique site where they could be found. Furthermore, it was necessary to check periodically this dataset due to possible rescheduling or cancellations.

Another issue, but related with the scraping process of bookings, was that some hotels that began to be published at the site did not have initially star rating information. So it was necessary to complete this with their final star rating, informed some days or weeks later.

Concerning to hotels information, Booking.com placed some hotels in a called 'Valencia' district. This district does not exist, so it was necessary to associate them to the correct location.
And also about hotels, although we did not use the hotel name for any analysis, some hotels changed its name during the scraping process. As an example, 'Tryp Valencia Feria' was later named 'Port Feria Valencia', or 'Hotel Sercotel Acteón Valencia' became 'Eurostars Acteón'. This situation was detected through the property identification used by Booking.com to identify each property in its system.

Finally, and because the author of this project had to travel during the phase of collecting data, it was tried to continue with the scrapping process abroad, but we found the problem that prices from Booking.com downloaded in the currency where it was launched the process. We decided not to use this information, as we do not know the Booking.com pricing policy in other countries. So finally these files had to be discarded.

7. Conclusion

In the present study, we tested and evaluated several ML models in order to predict relevant events according to a rise in hotel room prices at certain dates in the city of Valencia.

We analysed the accuracy of each one and checked out that the best accuracy was in a Decision Tree model, implemented in its CART version, with a 78,72 percent of right predictions.
Moreover, evaluating the Precision metric, our model correctly predicted near 78% of instances when there was a relevant event, and a 80% of instances when there was not a relevant event.

We consider that with a larger bookings dataset we could improve a little bit this percentages, although it would be difficult to improve much more due to the fact that prices can rise not only the day before to a relevant event, as it has been mentioned in the results section. Hotel policies take into account much more factors than relevant events in a city to establish its prices.

7.1 Research Limitations

Having presented the results of our evaluation and discussed the findings' meaning in relation to the data analysed, it is now important to highlight the limitations of this study.

The first and most important one in our opinion was the limited booking dataset. Our scraping process obtained a total of 233 days of booking data. It would have been good to increase this number to at least one year, to have a photo with all the annual events in the city.

In relation to the models analysed, this particular study restricted its scope to four binary classification models, selected to illustrate some of the most representative use cases of AD applications today.
Beside this, also the variety of the hyperparameters chosen was limited to the core settings of the binary classification models

In general, the more algorithms and the more datasets one can use for benchmarking, the better and more generalizable the results of the evaluation will be.

8. Future work

Although the provided analysis and methodology is quite good there are a set of potential developments and improvements that can still be made.

8.1 About bookings data

We focused all our analysis on four main features: check-in date, stars, days of advance search and room price.

But during the scraping process we obtained some other features that could be considered for deeper analysis. Let us mention some of them:

- Type of property: we carried out our study focused on hotels, but it would be interesting to see the prices behaviour on other kind of properties, such as apartments or hostels.
- District and postal code: an analysis of these features could show if there is a similar behaviour on room prices in hotels that are close to each other.

8.2 About events data

Some other enhancements to work in are related to the events dataset. We know that not all events have the same impact over the room prices. It will not be the same an international plants and flowers trade fair than a motorcycle racing World Championship, specially here in Spain where there is a huge number of fans of this sport. Thus, it would be interesting to give more weight to this kind of events.

Also regarding to this matter, in events dataset we had some dates with more than one event in the city. One question that arises is whether several events celebrated on a particular date have the same impact on prices as an important unique event. In our study we labelled our final class with value ‘1’ if there was as minimum of one relevant event the day immediately after to a check-in date. Weighing up our events the study could lead to a multi-class problem where the final class with value equal or higher than 1 could indicate the relevance of the event on that date.

8.3 About ML models

In our analysis we trained, tested, and evaluated our classification models with a final dataset composed of 5 predictors: one price per star rating and check-in date.
It could be interesting to evaluate these models against a 25 predictors dataset, obtained by: one price per star rating, days of advance search and check-in date.

Another point to mention is that we selected some of the best-known classification models, but there are some others that could have a good accuracy. Some of these models that could be tested are, between others, Naïve Bayes or Random Forest.

In relation to the Decision Tree used during our analysis, this was an implementation of the CART algorithm, but could also be tested the C5.0 one to check if achieves a better performance.

8.4 About a new approach

Before beginning to work with some of the ML models used in the present study, we transformed our initial TS data into five final predictors. Therefore, another exciting analysis would be to deepen in these TS and try to apply a predictive model such as LSTM to treat this valuable information, although it would be good to work with a larger booking dataset as deep neural networks have a lot of parameters to learn, and with small data, running a large number of iteration can result in overfitting.

9. Glossary

Term	Description
Online Travel Agent (OTA)	Third party booking websites, such as Expedia and Priceline, which offer travellers an easy-to-search database of travel providers.
Time Series (TS)	In general, a time series is a series of data points indexed in time order. In most cases, a time series is a sequence of observations measured at successive points in time, equally spaced from each other, hence creating a so-called discrete time series.
Multivariate Time Series (MTS)	A multivariate time series presents more than one time-dependent variable. Each variable depends not only on its past values but also shows some dependency on other variables (with varying degrees of correlation).
Anomaly Detection (AD)	Anomaly detection is the identification of data points, items, observations, or events that do not conform to the expected pattern of a given group. Anomaly detection is also known as outlier detection.
Recurrent Neural Network (RNN)	A Recurrent Neural Network is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. RNNs can use their internal state (memory) to process sequences of inputs.
Long Short-Term Memory (LSTM)	Long Short-Term Memory networks are a special kind of RNN, capable of learning long-term dependencies.
Cross-validation (CV)	Cross validation is a method of model validation which splits the data in creative ways in order to obtain the better estimates of “real world” model performance and minimize validation error.
Grid search	Grid-searching is the process of scanning the data to configure optimal parameters for a given model. Grid-Search will build a model on each parameter combination possible. It iterates through every parameter combination and stores a model for each combination
Classification report	A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False
Naive Bayes	Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features given the value of the class variable
Random Forest	Random Forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction

10. Bibliography

[1] Kemp S., (2020). “Digital 2020: Global Digital Overview”, online at https://wearesocial.com/digital-2020

[2] “Trends in the hotel & Travel industry booking and consumer behavior”, online at http://www.orisysinfotech.com/Article_trends-in-the-hotel.php

[3] Morris L., (2017). “Morris House Hotel”, online at https://morrishousehotel.com/third-party-booking-sites

[4] “Booking.com”, online at https://www.booking.com/index.es.html

[5] Booking Holdings, (2020) “Form 10-K Annual Report”, online at https://ir.bookingholdings.com/node/24796/html

[6] Feria Valencia, online at https://www.feriavalencia.com/visitar/eventos. Last accessed Mar 01,2020.

[7] Ticketmaster, online at https://www.ticketmaster.es/city/valencia/165063. Last accessed Mar 01,2020.

[8] Ocio Levante-EMV, online at https://ocio.levante-emv.com/agenda/valencia. Last accessed Mar 01, 2020.

[9] Love Valencia, online at https://www.lovevalencia.com. Last accessed Mar 01, 2020.

[10] Visit Valencia, online at https://www.visitvalencia.com. Last accessed Mar 01, 2020.

[11] Fundació Deportiva Municipal Valencia, online at http://www.fdmvalencia.es/es/agenda/. Last accessed Mar 01, 2020.

[12] Mostra de Valencia, online at https://lamostradevalencia.com Last accessed Mar 01, 2020.

[13] Fallas de Valencia, online at https://www.visitvalencia.com/en/events-valencia/festivities/the-fallas Last accessed Mar 01, 2020.

[14] Calendarios Laborales en Valencia, online at https://calendarios.ideal.es/laboral/comunidad-valenciana/valencia/valencia/2019 https://calendarios.ideal.es/laboral/comunidad-valenciana/valencia/valencia/2020 Last accessed Mar 01, 2020.

[15] Géron A., (2017). “Hands-On Machine Learning with Scikit-Learn and TensorFlow”, available at O’Reilly Media https://learning.oreilly.com/library/view/Hands-On+Machine+Learning+with+Scikit-Learn+and+TensorFlow/9781491962282/ch14.html#lstm_cell_diagram

[16] Brownlee J. “Develop Deep Learning Models for your Sequence Prediction Problems”, available at https://machinelearningmastery.com/lstms-with-python/

[17] Malhotra P., Vig L., Shroff G., Agarwal P., (2015). “Long Short Term Memory Networks for Anomaly Detection in Time Series”, In: ESANN 2015 proceedings, Bruges, i6doc.com publ., ISBN 978-287587014-8

[18] Laptev N., Yosinski J., Erran L., Smyl S., (2017), “Time-series Extreme Event Forecasting with Neural Networks at Uber”, In: ICML 2017 Time Series Workshop, Sydney

[19] Zhang C., Song D., Chen Y., Feng X., Lumezanu C., Cheng W., Ni J., Zong B., Chen H., Chawla N.V., (2018). "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data"

[20] Park P., Di Marco P., Shin H., Bang J., (2019). "Fault Detection and Diagnosis Using Combined Autoencoder and Long Short-Term Memory Network", In: MDPI – Sensors. 19. 4612. 10.3390/s19214612

[21] Cross-validation: evaluating estimator performance, online at https://scikit-learn.org/stable/modules/cross_validation.html
Last accessed Apr 11, 2020