APPLICATION OF MACHINE LEARNING IN AIR-TICKET PRICING IN CHINA

CHU QIN

PhD

The Hong Kong Polytechnic University

2022

The Hong Kong Polytechnic University
Department of Logistics and Maritime Studies

Application of Machine Learning in Air-Ticket Pricing in China

Chu Qin

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

May 2021

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

CHU Qin (Name of student)

Acknowledgements

First, I would like to take this opportunity to express my deepest gratitude to Prof. JIANG, Li, my Chief Supervisor, for his professional guidance, advice, and support in various aspects throughout my PhD study. He is always willing to give me vital encouragement and great support to my research. His constructive feedback and suggestions have significantly improved the quality of my research. His patient guidance has helped me overcome many difficulties in my life. His critical thinking has inspired me in my PhD study. I am sure that the experience would be beneficial to both my work and life in the future. I feel truly fortunate to have him as my supervisor.

I gratefully acknowledge the professors in my Faculty and Department, who have given me support and help. I would like to thank Prof. LAI, Kee-Hung who was one of the examiners of my confirmation of registration. I am thankful for his time spent on my confirmation and his comments on earlier versions of my paper. I would also like to thank Prof. LI, Chung-Lun for his supervision when I was the tutor of his course LGT3102 Management Science, from which I have learnt the necessary knowledge and skills for teaching. I would like to take this opportunity to thank Prof. Kaibo WANG and Prof. Guangwu LIU for reviewing my manuscript and providing valuable comments. I would also like to thank Prof. Hengqing YE for handling the submission and offering me the opportunity of revision.

My special thanks go to my classmates and friends, Dr. DONG, Ciwei, Dr. YANG, Yefei, Dr. HAO, Zhongyuan and many others, for their help and companionship. They have enriched my life in Hong Kong. Finally, I would like to dedicate this thesis to my family. Without their unconditional love and consistent support, this thesis would not have been possible to finish.

Table of contents

Abstract

This dissertation predicts the air ticket price to provide advice on the immediate or postponed purchase of a trip. The proposed methodology is based on the statistical learning of a price evolution model from the joint information of the trip attributes. The main originality consists in representing the price evolution with the inhomogeneous punctual process of the price variation. This representation was used to group flights from the Qunar.com database into similar behaviors and to build a common model for each of the identified behaviors. We then implemented a learning method to model the price evolution. This model provides a predictor for the occurrence of a price drop over a given period and therefore offers advice on the immediate or postponed purchase of a trip to the customer.

The research is organized in three main phases. First, we introduced a new method of representing time series of prices. This representation compared the series with each other and applied data mining algorithms. The approach is based on a preliminary statistical modeling for the dynamics of time price series and the theory of point processes. We first transformed the time series by point processes. Then we modeled these series of returns by an estimate of the intensity in gray level. In a second step, we proposed a segmentation of learning data in order to extract standard behaviors using unsupervised learning techniques. Based on our new representation of time series, we applied segmentation algorithms and extracted average behaviors called centroids. With each centroid, we simulated price curves for a behavioral prediction. In a third phase, we applied supervised learning algorithms on the attributes of the flights in each group to allocate new flights to a centroid with their attributes. The first chapter describes the data structure and explains the relevance of the decision support module. The next three chapters analyze the process of developing the representation, the data segmentation, and the learning phase with different settings for each of the steps. The experimental results are present with a comparative study of the different configurations for the algorithms and the approaches.

Introduction

The thesis presents the development of a study for predicting the evolution of finite time series and the supervised learning of typical behaviors generated by a clustering step. Airlines, followed by all tourism professionals, have generalized yield management policies in order to optimize the price of a service based on their inventory level and the reservation date. This results in opacity during the price formation process. The same ticket on the same dates can vary greatly from one supplier to another, and from one moment to another. The consumer is kept in ignorance and uncertainty. They are encouraged to book in advance or at the last minute to benefit from offers presented as discount. Prediction of time series is a widespread subject with various applications: the prediction of changes in stock prices in finance, the short-term prediction of temperature in meteorology, the prediction of the number of flight reservations and the evolution of CO_2 emission in aviation. The approach is based on the statistical analysis of previous developments with infinite series. Qunar.com is a search engine capable of searching for an airline ticket from over 250 travel agencies and airline websites. Suggestions to the purchase decision based on an estimate of the trend in the price evolution are provided.

Our time series represent the price evolution of a perishable product. They all have an expiry date beyond which they are no longer available. Our database consists of a set of finite series to extract average behaviors. We attributed the most likely behavior to the new flights to extract a prediction. We conducted firstly an unsupervised learning stage of the time series leading to standard behaviors groups, followed by a learning step of these behaviors based on the attribute vectors of the flights in each group. We observed a collection of time series of the same length, regularly sampled, on a representative panel of routes. The collection is characterized by a set of static attributes (route, timetables, airline, etc.) and the point of the curve (price at time t, past trends, etc.). We grouped these series according to their trajectory to similar behaviors. Each of these groups is associated with an identifier called label, a standard behavior and a topology of attributes corresponding to the population of the group. The objective is to predict the most likely label for each result of a search due to the flight attributes. We then extracted from the standard behavior with the label and the information necessary to predict the evolution of flight prices.

The prediction methods are essentially based on data-mining approaches, rather than exploiting the temporality of the price series or the data summary such as gross time averages. We proposed to implement machine-learning methods considering the nature and complexity of the input data. Conventional learning methods for prediction have generally been developed, as the input data is independent and identically distributed. However, the form of historical time series has an essential dependence structure with strong correlation. The envisaged approach is based on a preliminary statistical modeling of the dynamic time price series and the theory of point processes. As the observed prices evolve “by jumps”, the notion of punctual process is widely used for the queuing modeling in operational research.

We proposed a supervised learning behavior of the time series for price, basing on a new representation of data. The whole process can be divided in two stages: the first step consists in

transforming the price series into series of relative jumps. The second step groups the jumps in time and intensity window to form a gray level. A medium is to be found in the size of the windows to best represent the evolution of prices. It is therefore on these new representations that we intend to apply our algorithms to extract standard behaviors. Our new representation conducts two methods based on the modeling of time series. A first approach consists in applying the K-Means algorithm to the gray levels by measuring the distance between the series by Euclidean difference: the algorithm partitions the space of the gray level boxes to create groups of similar behavior minimizing the distance to the centroid. The other method is based on mixtures of models maximizing the likelihood of flights to the centroid of their group by an expectation-maximization algorithm.

We used several supervised learning algorithms, namely trees of classification (CART and C4.5), Adaboost and the Random Forest. We improved the EM algorithm so that it performed the segmentation step together with the classification step. The classification rules created to directly interpret and identify important attributes were observed by decision trees. Likewise, random forests rank each attribute in order of importance in the classification by observing their influence in the multitude of trees created. It is then possible to select the best attributes to build a more efficient predictor. The new representation of time series improves the aggregation stage and the probability model development. We succeeded (i) in modelling general behaviors to predict possible evolutions and (ii) in directly predicting the evolution of the price. The implementation of this research requires constant monitoring of predictions and regular updating of models.

Chapter 1 describes the data from our database to extract information about the user behaviors and the price changes. Started by describing the construction of database and the choice of our structure, different types of purchases are analyzed to choose the proper modeling parameters. Then we studied the behavior of price curves by visualizing the different types of yield management. Finally, the relevance of our project will be discussed to suggest purchasing decision. Chapter 2 analyzes our new representation of the data obtained by transforming the time series of prices into intensity estimation. The idea is to abstract price differences from flights in various distance, which are likely to follow the same types of variations at different price levels. The time series were first transformed into series of returns where only price changes are represented as a percentage change. We then modeled this sequence of points by an inhomogeneous process, the intensity of which can be estimated in the form of a pixelated image. Chapter 3 focuses on the segmentation of learning data. This segmentation extracts a finite set of typical behaviors from our flight database. The K-Means algorithm was adopted with the case-by-case gray levels as a measure of distance. We then tried to stabilize the results of the K-Means by applying the principle of bagging, which consists in multiplying the segmentation step by random subsets of the learning base. This method, called Bagged K-Means, improves convergence to the global optimal and attenuates the influence of initialization. Finally, we applied the Expectation-Maximization algorithm where the optimization criterion will not be the distance to the centroid but the overall likelihood of the model. In Chapter 4, the steps leading up to the final prediction are outlined. The first step is the supervised learning of the group identifier E_i, according to the flight's attributes of each group. Different algorithms such as classification trees, adaboost and the forest decision tree algorithm were conducted. To classify a new flight in its most probable group, the learning process associates an average behavior

with a certain number of curves simulated. By averaging the behavior of these simulations, we obtain a prediction of the future evolution of flights in corresponding groups. The allocation of a flight to a group is then improved by the addition of information on the first price changes. Finally, the experimental results are presented in the last chapter. To evaluate our results, various metrics were used, such as the percentage of satisfactory predictions, the average percentage of gain and loss per ticket and the ROC curve. It is important to know which criterion to optimize between the percentage of satisfactory predictions and the average gain saved per ticket.

Chapter 1

Introduction

Tourism has become an essential contributor to China's domestic economy. The total revenue of tourism industry in China amounted to around 7 trillion yuan in 2019, indicating a firm growth over the past decade. The emergence of an affluent middle class and an easing of restrictions were both supporting this travel boom. The sector was expected to contribute 3.3 percent to China's gross domestic product (GDP) directly by 2028. The number of domestic trips reached six billion in 2019, indicating an exponential increase compared to the number of trips made in China ten years ago. Chinese airlines have grown rapidly. From 2009 to 2019, demand grew at 14 percent yearly. China is the world’s second-largest domestic market.

Recent advances in Artificial Intelligence and Machine Learning infer rules and model variations on airfare price. They often found relationships among the features automatically. This chapter presents the structure of our data and the choice of parameters. It is essential for the learning phase construction and the validation of our approach. It involves the price optimization techniques applied by different commercial sites. Price changes in the airline industry follow rules governed by yield management or revenue management algorithms. The purpose of this practice is to increase the company's revenue per available seat. The parameters in optimizing prices are therefore the rate of aircraft occupancy and the evolution of demand.

Qunar.com is a travel search engine allowing users to compare more than 250 travel agencies and airline websites. With each user search, all the information on the results page is kept in a database representing a large source of information to be processed. Our learning base represents the majority of existing behaviors while maintains a reasonable size. The structure reconstructs the time series of prices and compares the same flights offered by different sites. The concept of single flight describes a route defined by departure and return dates, airports, flight codes and stopovers. Each unique flight therefore has a time series of prices per merchant site.

This chapter starts with the literature review, followed by the origin and the structure of our data. Then the choice of our parameters for constructing the learning base are d by the statistical analysis of the user behavior in the ticket purchase process. Finally, the selection of routes and the length of stay were discussed.

1.1 Notations

n: Number of time series in the database.
i: Flight number of the learning base $i \in 1, ..., n$
V_i: Attribute vector of flight i.
p_i (t): Flight price curve i.

1.2 Literature review

Machine learning is an important field of artificial intelligence technology. This review summarizes the machine learning methods and existing applications in the aviation industry to predict the ticket prices, forecast the future demand and maximize the revenue. Prices vary widely depending on multiple factors such as airline policies, holidays, the number of seats available and so on. Given this situation, it is a difficult task to predict airline ticket prices due to the non-linear behaviors influenced by the competitive factors and revenue maximization policies. Therefore, traditional theories have become decreasingly capable to predict airline ticket prices due to the limited capability of describing non-linear relations. In contrast, machine learning approaches, which are operated like a black box, become promising in the multi-factor based prediction of airline ticket prices, with the access to extensive data records. Machine learning algorithms are classified as artificial intelligence including many models such as decision tree, random forest, K-means, neural network and so on. All those models can be used for the prediction of airline ticket problems with different predictability. The ticket price of the same flight will change dynamically. It can change up to 7 times for the same flight on the same day (Narangajavana et al., 2014). Etzioni (2003) recorded more than 12,000 airfare observations over a 41-day period and conducted Hamlet, a multi-strategy data mining algorithm in machine learning, to generate a prediction model to suggest the best time to buy tickets.

1.2.1 Machine learning algorithm in airline industry

Domínguez-Menchero et al. (2014) found the machine learning algorithm was more effective than standard parametric techniques after analyzing the price of tickets from Madrid to London, Frankfurt, New York and Paris in two months and the sale of pre-ordered tickets for up to 30 days. They noticed that consumers had an 18-day gap before departure, and there was no significant financial penalty for buying a ticket during that time. This raises the possibility of determining the best time to buy tickets and balancing the money saved with the time constraints (Li et al., 2014). Based on empirical data, Lantseva et al. (2015) analyzed the Russian air transport market and compared the price behavior of both local and global flights. Groves and Gini (2015) proposed to use time-delay features to capture time dependencies in data and optimize the ticket purchase time. When a flight route and travel dates are determined, machine learning methods can be used to predict the lowest expected future prices for all available flights. Most of the research on the customer side has focused on the use of statistical methods to predict the optimal purchase time. As pointed out by Chen et al (2015), it is more difficult to predict the actual ticket price than the optimal purchase time due to various reasons: lack of sufficient data sets, external factors that affect ticket price, dynamic behavior of ticket pricing, competition among airlines, exclusion of airline ticket pricing policies, etc.

Etzioni et al (2003) proposed a machine learning model that suggests the user whether to buy a ticket or to wait at a particular point of time. The model generates buy or wait signals based on historical price information. The model uses various analytical techniques such as rule learning (RIPPER), reinforcement learning (Q-learning), time series methods, and combinations of these

methods to achieve different levels of accuracy. Functions used include flight number, number of hours away from departure, current price, airline and route (origin and destination city). Inspired by Etzioni et al (2003) and G et al (2013), the optimal purchase time to predict all available flights of different airlines on a given departure date and route is determined by both deterministic characteristics and aggregative characteristics. PLS regression was adopted to generate the optimal model, which saves 75.3% more compared with the previous version. Chawla et al (2017) pointed out that the ticket prices vary depending on some variables including oil prices, departure days, the number of parking slots, etc. They also described and compared two machine learning algorithms to predict price trends. Xu & Cao (2017) proposed a new type of optimal decision support service for ticket purchase (OTPS), which can continuously recommend the best purchase time before flight departure. OTPS is a dynamic ratio of potential days strategy based on low prices and the fluctuating trend of airfare over time. In order to improve the reliability of OTPS, a large number of experiments are carried out on the multi-route ticket price dataset. Tziridis (2017) generated a new dataset of 1,814 flights from Aegean Airlines to identify characteristics of a typical flight. These features were applied to eight of the most advanced machine learning (ML) models including multi-layer perceptron (MLP), generalized regression neural network, extreme learning machine (ELM), random forest regression tree, regression tree, Bagging regression tree, regression support vector machine (polynomial and linear) and linear regression (LR). The performance of each model was compared. The experimental results show that the accuracy level is close to 88% for a certain type of flight characteristics. Vu et al. (2018) proposed a random forest model without the requirements for official airline information to forecast trends.

1.2.2 Air-ticket pricing models

There are two main types of literature on airline pricing models. The first group proposed demand forecasting models (An, et al, 2016; Yuan, et al, 2014; Mumbower, et al, 2014) and the second group focused on price discrimination (Puller, et al, 2012; Mantin, et al, 2010; Alderighi, et al, 2011; Wen and Chen, 2017). An et al (2016) proposed MAP (maximizing airline profit) to help airlines forecast market share and route demand. Meanwhile, the study introduces a new integrated forecasting method called MAP-EF with two new features: derivative features and equilibrium-based pricing features. It shows that MAP-EF achieves much better Pearson Correlation Coefficients (over 0.95 vs. 0.82 for market share, 0.98 vs. 0.77 for demand), while generating much lower variance. However, compared with previous models, the proposed model has a higher time overhead due to the increased time spent on clustering and the use of more advanced regression methods. Puller, et al (2012) used unique transaction data and identified one source of airline price discrimination. It is not difficult to find that the weekend buying effect is significantly greater on routes with a mix of business and leisure travelers than on routes that disproportionately serve leisure travelers. The article shows that such pricing practices can have a significant impact on airline profits. These results have implications for other industries that could adjust prices daily based on the type of customer who purchases on a specific date.

In some cases, tickets bought in advance may cost more than tickets bought later. For example, researchers create a dynamic pricing framework, and analyze the long-term and short-term impacts using regression methods. Dynamic pricing can make better prediction based on active factors such

as demand change and price discrimination, rather than internal factors, external factors, competition among airlines and strategic customers (Malighetti et al., 2009). The hyperbolic price function is used to estimate the optimal price curve for each route and analyze the pricing policy adopted by Ryanair. It is found that the dynamic pricing is negatively correlated with route length and flight frequency. On the contrary, discounts on advance tickets will increase as the competition intensifies. The deep learning is a major trend in the future development of demand forecasting. Convolutional neural networks (CNNs) have attracted extensive attention in recent years (Krizhevsky and Hinton, 2012). How to reduce the pre-training words combined with CNN to predict the demand should be further explored. The data extracted by CNN from social media are classified into popular destinations, future events, etc., which can also be predicted for the demand forecast and price analysis.

Several methods have been proposed in the literature to identify behaviors from a set of time series. The first approach consists in not modifying the nature of the series but in using appropriate distance measurements. A major constraint is the need to have standardized and sampled series. Then the methods based on the extraction of attributes are linked to the series from the simplest extraction of important points to the spectral transformation. These methods do not correspond to the phenomenon of appearance of jumps and to the structure of our time series constant by pieces. However, each gray level box can be assimilated to an attribute of the flight that is given to the input of the K-Means algorithm. The distance between the flights will then be the sum of the differences. The centroid of each group will be the average box-to-box of all the flights in the group. The last approach considers that each time series is generated by a model or a mixture of underlying probability distributions. Xiong and Yeung assumed that their ARIMA (Auto Regressive Integrated Moving Average) time series are generated by k different ARMA models. They then used an Expectation-Maximization (EM) algorithm to learn the parameters and coefficients of these models maximizing the log-likelihood. We followed the same approach by applying the principle of the EM to estimate the parameters of a mixture of densities by automatic classification. We then assumed that the gray levels are realizations of a one-time Poisson process of intensity given by the centroids representing the average behaviors of the groups.

Within the same cabin, the airline divides its aircraft into reservation classes, or tariff classes, or even yield classes. It is a purely computerized division, invisible to the passenger, and without consequence on the positioning of travelers at the front or rear of the aircraft. This division should not be confused with the division into transport classes, which are the first class, the business class and the economic class. The booking classes are subdivisions of the aircraft within these cabins. Each flight is broken down into 10 to 20 booking classes. They are designated by letters of the alphabet. A lower class cannot encroach on a higher class, while a higher fare class can be provided for a lower class. Low-cost companies mostly apply the same principles, but in a highly simplified manner. Thus, the price of their tickets generally varies only according to two factors: the purchase in advance, and the state of filling of the plane. At any given time, there is only one price for the plane ticket, valid for everyone. This system has the advantage of being well understood by passengers, because it can be summed up by the simple formula “the earlier you buy, the less it is expensive". This principle is overridden daily by revenue management algorithms, which guarantee lower prices in various situations: ticket cancellation, increase in aircraft size, return of seats

allocated to travel agencies, etc. The main purpose of the algorithms used is to determine which booking classes will be open on a flight, with which quota of seats has been allocated to each. It is a control of the supply by adjusting the available capacities. For example, it will be necessary to open a lot of seats in the low booking classes and keep only a few for high-contribution passengers on a flight during off-peak hours, which otherwise will not be filled, whereas on a flight during peak hours it will be the opposite to obtain the maximum income.

Bid-Price is one of the methods used in the airline industry to maximize revenues. The bid-price vectors are indications of price changes per cabin sent to the GDS so that they adjust the announced prices as they are filled. Each cabin is divided into classes associated with a price. All classes with a price lower than the bid-price will then be closed for sale. The creation of this vector is conducted in several stages and requires a certain number of input parameters such as past changes in requests by cabin, the capacity per cabin (first class, business, economical), the history of seats allocated to travel agencies so on. With all these parameters from historical databases, new information such as the overbooking rate and the demand prediction will be calculated. Given all the output data, an optimal bid price vector can be constructed with optimization algorithms which will then be transmitted to the GDS. In most cases this optimization process is conducted daily for flights with a departure date. The price fluctuation will therefore behave differently over time. We evaluated the consequences of yield management on the behavior of price series in the next section.

Alexander Yates developed an algorithm called HAMLET combining several data mining algorithms. The learning base consists of flights sampled every 3 hours for 21 days. They predicted the evolution of the next point (t + 3 h) and decided to use a simple algorithm based on rules RIPPER: each point has 5 attributes which are the number of the flight, the number of hours before departure, the current price, the airline, and the route. In the learning base, they assigned a “buy” or “wait” class to each point, depending on the evolution of the next price. If the price increases the class will be “buy” and if the price is stable or decreases, the class will be “wait”. The algorithm therefore created a definite number of classification rules, easily interpretable, based on the attributes described above. Secondly, a classical reinforcement learning algorithm (Q-learning) was used by modifying the reward rule. A third algorithm based on the sliding average of the time series then intervened. Using information from the previous 7 days, following formula can be generated:

$$\frac{\sum_{i=1}^{k} \alpha(i) P_{t-k+1}}{\Sigma_{i=1}^{k} a(i)}$$

where α(i) is an increasing function of i, providing a price prediction in step p_{t+1}. The prediction rule is then the same as before: if $p_t \geq p_{t+1}$ then advice to wait, otherwise advice to buy. The binary predictions (“buy” or “wait”) of these 3 algorithms are finally added to the RIPPER algorithm as additional attributes to create the final predictor named HAMLET. Despite promising results, the increase in the number of routes and flights posed a problem of computing time and memory storage. Furthermore, their approach is inflexible because it depends on the nature of the prediction to be made. If the desired prediction suddenly changes from t+3h to t+8h or as in our case to t + 7d, the whole learning process should be performed again, but the same performance is not guaranteed.

Previous works have two main weaknesses: (a) they do not consider dynamic pricing at a travel request level, and (b) only booking data from few airlines are taken into account

Regarding the first issue, the fact is that the pricing change dynamically during the booking period because of the revenue management systems, seat availabilities, and sales strategies. It should be underlined that only few actors in the industry could have access to process exhaustive pricing information at travel request level. This could be the main reason that it has not been included in previous works. Second, most previous works had access to partial information. The bookings of a single airline may not capture some changes on the market conditions. Not observing other airlines could impact the model through airline preferences and price sensibility. The significance and magnitude of other attributes can be biased. If only the bookings of a full-service carrier are studied, the models may underestimate the price elasticity.

Therefore, our research questions are:
Is that possible to develop a price predictor for potential customers using machine learning algorithm?
How long does the prediction stay efficient?
How customers choose between itinerary alternatives when searching for flights?
How customers behave differently for various travel purposes?

1.3 Data

This section describes the data properties, and how it has influenced the database structure. Three main parts are composed to facilitate the preparatory work: the identification of a single flight, the extraction of the associated time series $p_i(t)$ and finally the creation of the corresponding attribute vectors V_i.

1.3.1 Data description

The learning and testing base were built upon a historical Qunar.com user search database. In this database, a single journey is defined by 6 attributes: the routes, the dates of departure and the time of return including the hours and the minutes and the carrier code. Prices are collected from both commercial sites such as regular airlines (Air China, China Eastern Airways, etc.) and travel agencies (Ctrip, Skyscanner, etc.). Specifically, “provider" refers to the merchant site from which the price is extracted. “Supplier” represents the airline company. Each user search results in 316 choices with parameters such as the merchant sites (provider), airlines (supplier), schedules and rates. Flights provided by different merchant sites are grouped together to compare similar offers and reduce display. Additionally, users can define alerts. The alert is a programmable tool which performs a requested search every 6 hours. After having filled in the destination and the expected dates, the traveler receives a summary of the best results by email or application alerts each day, sorted by price. These alerts long series of prices regularly sampled but no same flight is guaranteed. The data source provides consistent series to detect the price variations. To follow the nature of the database, two subsets were chosen: A set of 28 days series sampled every 6 hours and a set consisting of 90-day series sampled daily.

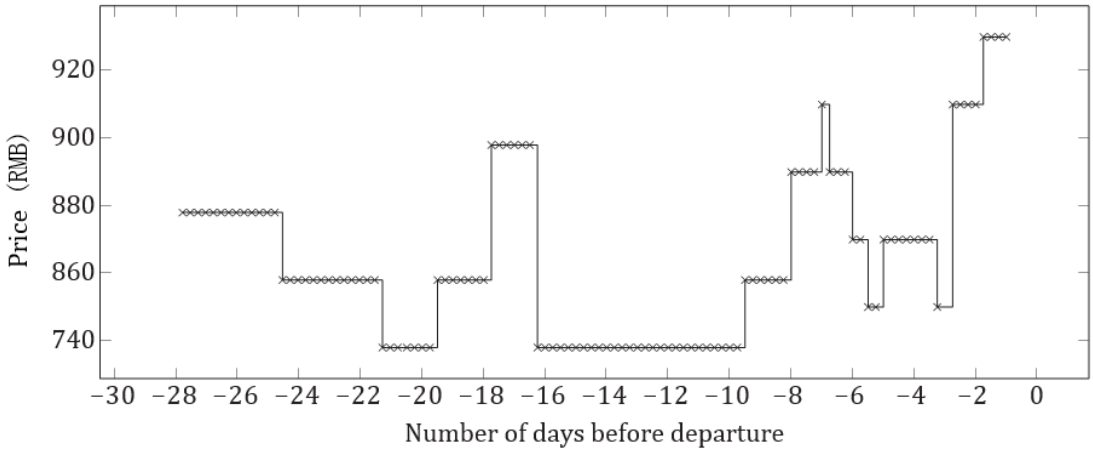

FIGURE 1.1 - Example of a price series for a Shenzhen-Chengdu from 20/11/2017 to 23/11/2018 offered by China Eastern Airways.

The three main tables in our database will be described in the following paragraphs.

1.3.2 Database structure

As explained above, we divided our database into three main tables to extract the information necessary for different stages of our prediction process. A first table stores the parameters of the unique paths and associate them with a unique identifier. The second table collects the prices for each single flight/provider couple. The last table contains all the parameters associated with a series of prices, such as the type of company, the city, the departure and arrival airport, etc.

Unique flights

A flight operated by an airline is defined by a departure airport (departureStation) and an arrival airport (arrivalStation) named “route”, a departure schedule (day, month, year, hour, minute), and a return schedule, a departure carrier code (transportCode) and a return carrier code (transportCodeRet). For database maintenance and volume problems, stopover information was not stored.

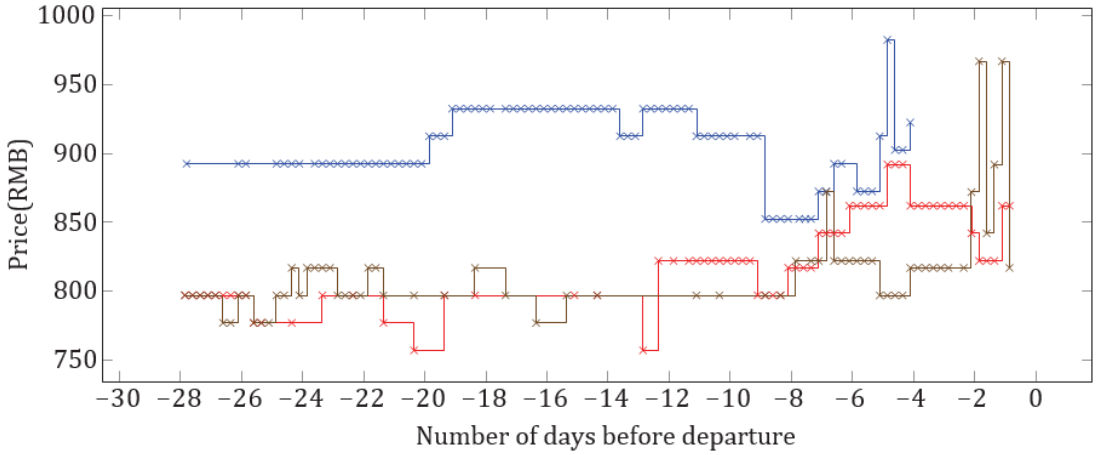

FIGURE 1.2 - Time series of the same carrier codes from Shanghai to Beijing on Air China sold by Qunar.com at different times of the year.

We then defined a single flight by the following: {departure airport, arrival airport, departure date, return date, carrier code, return carrier code} excluding the notion of merchant site. This unique flight represents the journey independently of the seller. Then a unique identifier called id_unique_flight were associated to each tuple. The creation of a unique identifier independent of the merchant site will observe competition phenomena or detect additional price changes applied by travel agencies. In certain cases, the route and the dates are identical, but the carrier codes are different with code sharing. These commercial agreements increase the visibility of the two companies and ensure better profitability.

Price series

Each flight is sold by one or more merchant sites that we called "provider". The couple {id_unique_flight, provider} (unique flight, merchant site) also has a unique identifier called id_flight. Each row of the table corresponds to the price series $p_i(t)$ proposed by a merchant at an instant t. When a user search was performed, an entry was added for each result in this table. Different identifiers were created for the associated flight tickets. For example, the same Shanghai-Beijing flight operated by Air China is sold by different travel agencies. To identify these series, it suffices to associate all the id_flight with the corresponding id_unique_flight. Given that travel agencies adopt their own yield management policy to further improve the flexibility, they optimize the price strategies up to several times a day. Different time series identified by id_flight were observed for the same id_unique_flight. In Figure 1.3, the price series of the same Shanghai-Hongkong ticket sold by the regular company (Air China) and the travel agency (Qunar) were demonstrated.

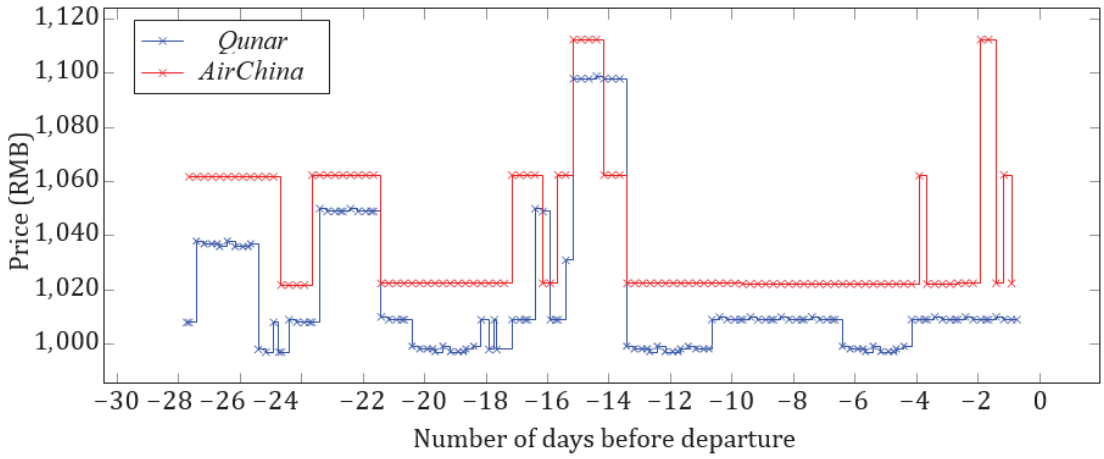

Figure 1.3- Shanghai-Hongkong flight operated by Air China, sold by Qunar and Air China

Attributes

Each id_flight is associated with an attribute vector $V_i(1), \ldots, V_i(p)$ which groups all the possible information together. These attributes are crucial for extracting statistics from the learning base. Characteristics to define a single flight were divided into four categories (Table 1.1).

TABLE 1.1 - Characteristics of the single flight.

Name	Type	Description
day	[1 – 31]	Day of departure month
month	[1 – 12]	Departure month
year	N	Departure year
departureHour	[0 – 23]	Departure time
departureMinute	[0 – 59]	Departure minute
transportCode	Code	Carrier code
departureStation	Code	Departure airport code
arrivalStation	Code	Arrival airport code

The temporal attributes (season, departure on weekends, day of the year, etc.), geographic attributes (city) and the attributes linked to the journey (number of stops, etc.) are listed below.

TABLE 1.2 - Attributes derived from single flight characteristics.

Name	Type	Description
season	[1, 2, 3, 4]	Season (1 = spring, ...)
length_of_stay	N	Length of stay
day_of_year	[1 – 365]	Day of the year
day_of_week	[1 – 7]	Day of the week (1 = Monday)
dep_on_weekend	Bin	Departure at the weekend
dep_periode	[1, 2, 3, 4]	Period of the day (1 = early morning, 2 = morning, 3 = afternoon, 4 = evening)
stops	N	Number of stopovers
departureCity	Code	Departure city code
arrivalCity	Code	Arrival city code

Table 1.3 demonstrates attributes referring to the commercial site. The contextual attributes which evolving with the time series are listed in Table 1.4. At each time t, the number of jumps observed previously and the sum of search requests are calculated. This information was stored in a separate table with the id_flight at the time t.

TABLE 1.3 Attributes linked to the merchant site

Name	Type	Description
provider	Code	Merchant site code
type	[0, 1, 2]	1 = Travel agency, 2 = Regular company, 3 = Low cost,
directSeller	Bin	Direct seller
train	Bin	Train journey

TABLE 1.4 - Contextual attributes evolving with the time series

Name	Type	Description
volatility	N	Number of jumps

volatility_increase	N	Number of increased jumps
volatility_decrease	N	Number of decreased jumps
demand	N	Number of user searches

We built our time series, accessed the attributes of a flight, and displayed the unique flight series sold by different sites simultaneously. The only difficulty was to highlight the co-branded flights, as it requires an additional unique identifier. The complexity in the database structure would decrease performance for limited utility. These behaviors influenced the parameters of the time series modeling.

1.4 Choice of parameters

The database extracts essential information including price series, attributes, and all possible derived statistics to construct the prediction model. The selection reflects the customer expectations on both prediction reliability and information accessibility. Furthermore, the differences in customer behavior between leisure and business purpose depending on the length of stay will be discussed. Therefore, the parameters of the flights (route, length of stay, merchant site) will be described first, followed by the choice of the time series length.

1.4.1 The routes

Initially, a set of representative routes was focused. Different travel purposes for leisure (7 and 14 days) and business (3 days) at different length of stays were chosen. Both medium-haul and short-haul flights were selected. The travel agencies offer a wide range of airline tickets and adjust the price more frequently. While the airline companies' official websites guarantee a more stable prices on the same flight.

TABLE 1.5 - Routes of the learning base

Low-cost carrier			
From	*To*	*Provider*	*Length of stay*
Shanghai	Chengdu	Juneyao Air	3,7
Shanghai	Hangzhou	Juneyao Air	3,7
Shanghai	Qingdao	Spring Airlines	3,7
Shanghai	Hangzhou	West Air	3,7
Travel agencies			
From	*To*	*Provider*	*Length of stay*
Shanghai	Chengdu	Qunar	3,7
Shanghai	Beijing	Qunar	3,7
Shanghai	Hangzhou	Qunar	3,7
Shanghai	Hongkong	Hong Thai Travel Services	7,14
Shenzhen	Beijing	Hong Thai Travel Services	3,7

Shanghai	Singapore	Hong Thai Travel Services	7
Regular Companies			
From	*To*	*Provider*	*Length of stay*
Shanghai	Hongkong	Cathay Gragon	7,14
Shenzhen	Beijing	Air China	3,7

The most frequently searched routes and requested lengths of stay are selected to build consistent series (Figures 1.4 and 1.5).

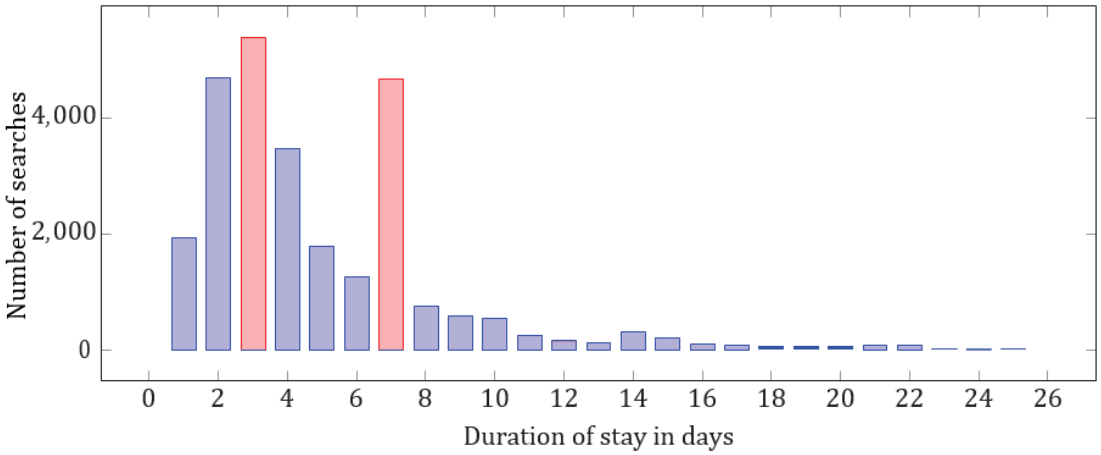

FIGURE 1.4 - Number of searches by length of stay for a Shanghai-Chengdu flight

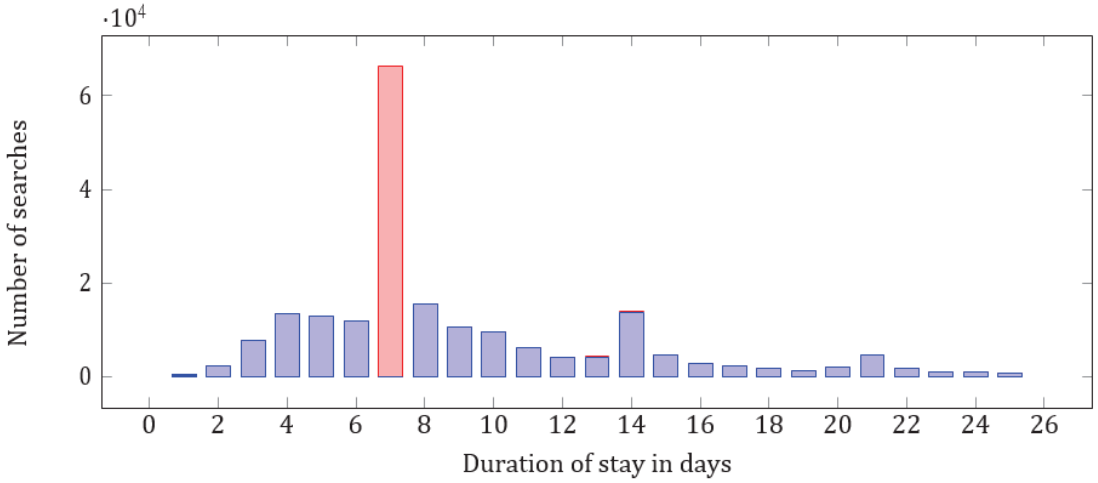

FIGURE 1.5 - Number of searches by length of stay for a Shenzhen-Beijing flight

The most popular flights were selected to guarantee price series. We initially examined the last days of our series to ensure a consistent basis. The customer behaviors were studied to find the optimal duration of the time series to choose.

1.4.2 The length of time series

In our example, time series were covered with 65% of users for medium-haul and 81% for short-haul within the last 28 days. We decided to focus on the last 28 days of the price series. In addition, the month preceding the departure date corresponds to a sharp increase in jumps as shown in Figure 1.6 and results in price volatility.

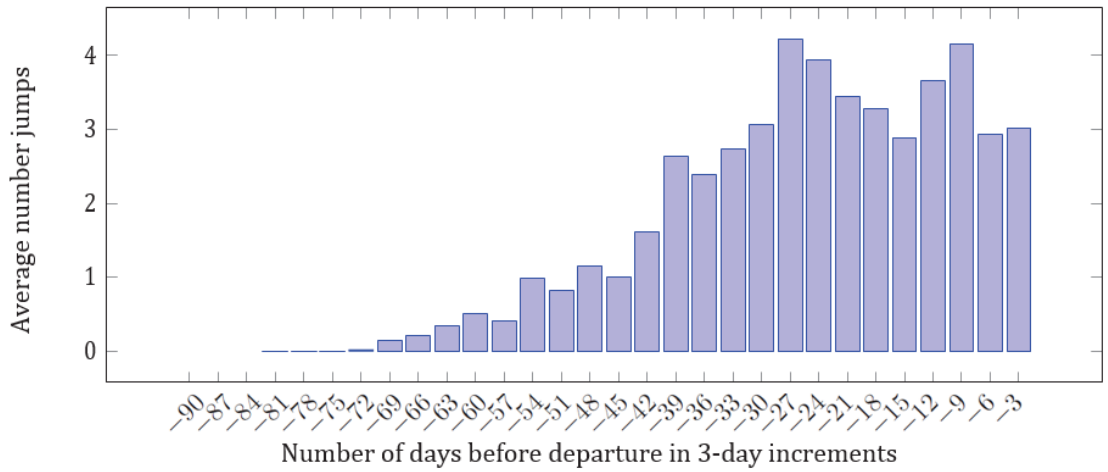

Figure 1.6 - Histogram of the number of jumps before the departure date in 3-day increments.

1.5 Relevance

For each user search, several offers provided by different companies at different schedules were presented. In these choices, the user will identify the targeted flight and the advice to buy immediately or postpone the purchase will be presented.

1.5.1 Best time to purchase

The most common behavior of a traveler is to buy air tickets in advance to ensure a reasonable price and avoid successive price increases. Studies have shown that prices are not strictly increasing and that a period of around 8 weeks before departure is optimal. There is indeed a period when prices are generally lowest, and which corresponds to approximately 50 days before the departure.

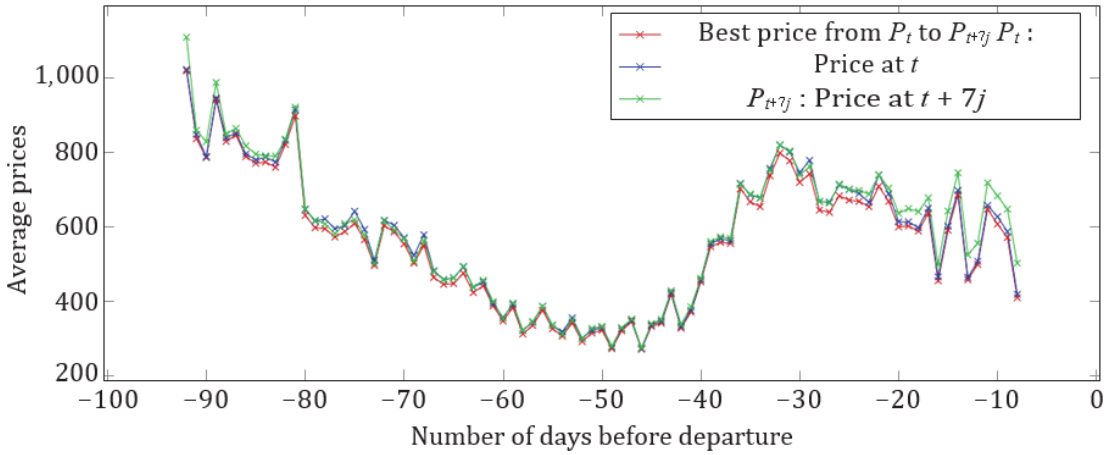

FIGURE 1.7 - Evolution of the average price depending on the time of purchase

In Figure 1.7, the optimum price at time t and 7 days later was observed. Thus, the most favorable period for the purchasing advice is between 20 to 35 days before the departure date. In this period the right purchasing advice should be provided. As far as the last days are concerned, advice is also essential. Finally, we noticed the small price differences between 60 and 40 days before departure.

1.5.2 Proportion of discounts

The research is emphasized through the proportion of frequent discounts. Regular airlines, such as Eastern Airline and Air China, are practicing scheduled discounts. While low-cost airlines are adopting more aggressive price optimizations and generally not selling refundable or exchangeable tickets with few price reductions. Certain attributes such as the name of airline are therefore more discriminating than others in price prediction process. Thus, the ticket sold by low-cost carriers may have more probability to increase within 7 days before the departure date than a regular flight.

1.5.3 Optimal gain

A gain or loss was defined as the absolute value of the difference between the initial price and the 7-day price. The optimal is the predictor who knows in any case the decision to make. The deviation was calculated as a function of the prediction date for immediate purchase, 7-day purchase, and randomly choices respectively. The red zone in Figure 1.8 represents the gain bringing to the customers through the precise advice.

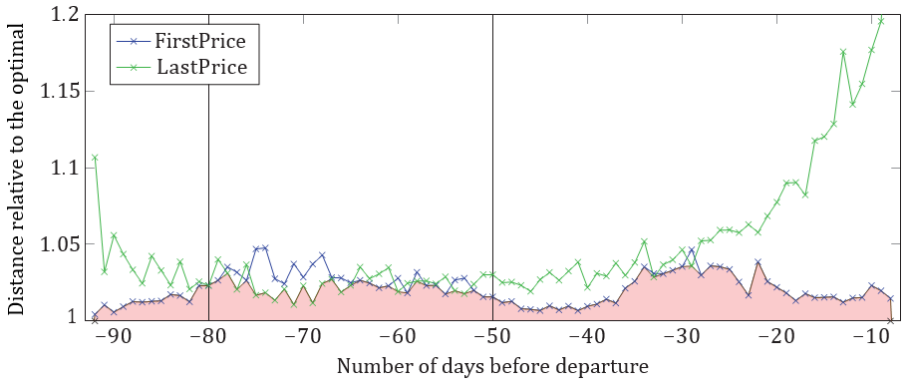

FIGURE 1.8 Distance relative to the optimal

1.6 Conclusion

This chapter describes the database structure containing user searches and application alerts. The price series were extracted to compare the same tickets sold by different merchant sites and to chose the corresponding attributes. A representative learning base with different travel purposes were implemented. Focusing on the last 28 days before departure guarantees a sufficient number of points to detect the majority of price jumps. Finally, we showed that it was necessary to provide information on price trends to the customers. The next chapter demonstrates the modeling of time series by point processes to group the same behaviors.

Chapter 2

Introduction

The step of transforming trajectories into a representation for comparing behaviors was discussed in this chapter. Similar price movements were grouped together to extract typical behaviors. Several problems regarding the similarity calculation between two price curves then raised. Understanding the nature of the price series facilitates the learning process. Statistics justified our choices in the previous chapter. The first step at this stage was to transform price curves by point processes. This step was completely bijective with price variations. The transforming process ignores irrelevant price variations and improves the computation time.

We assumed that the number of occurrences of price changes follows a Poisson distribution whose parameter $I_y(s, t)$ changes over time. An inhomogeneous Poisson process with intensity $I_y(s, t)$ was applied. The number of customers that have a travel request each day can be modelled using a Poisson distribution. This scenario meets each of the assumptions. Firstly, the number of events can be counted. The number of customers that visit the website and have a travel request each day can be counted. Secondly, the occurrence of events are independent. The arrival of one customer does not affect the arrival of another customer. Additionally, the average rate at which events occur can be calculated. Data can be easily collected on the average number of customers that have a travel request each day. The fourth assumption refers to the situation that two events cannot occur at the same instant in time. Two customers cannot technically have a travel request at exactly the same moment in time. We therefore approximated the appearance of the jumps and their performance by gray levels where the intensity of the cells represented the number of jumps in the interval. Then an imprint of the overall behavior of a flight is compared with each other. Finally, a price curve derived from the gray level was simulated.

2.1 Notations

i: Number of the flight in the learning base i 1, ..., n
V_i: Vector of p attributes of flight i among the set of attributes
$p_i(t)$: Flight price curve i
($P_i(1, k)$, $P_i(2, k)$): Raw data composed of Pi (1, k) the date of the k-th sample collected for flight i and $P_i(2, k)$ the corresponding price observed

$T_{init}^{(\tilde{l})}$: Date of first flight collection point i

$T_0^{(\tilde{l})}$: Flight departure date i

$s_i(t)$: Series of flight returns i

$(T_k^{(i)}, s_k^{(i)})$: k-th time-efficiency point of the representation in point process of the flight path i

R: All the boxes of the time-return plan
b_r: Vertical dimension (yield)

b_t: Horizontal dimension (time)
$X_i(s, t)$: Grayscale of the time-performance plan for flight i in box (s, t)

2.2 Time series

A single flight (id_unique_flight), representing a journey independently of the commercial site, is defined by a departure airport (departureStation), an arrival airport (arrivalStation), a departure date (day-month-year, hour: minute), a return date, a departure and a return transport code. Each flight has a time series per merchant site. The price series from the airline's website and travel agencies offer the same flight. The provider attribute defines the commercial site (the airline or the travel agency) and associates with the id_unique_flight.

2.2.1 Sampling problems

The average waiting time between two price jumps depends on the number of days before the departure date. We observed an average time of 2 and a half days between two jumps on all the dates and a strong increase in the frequency of price changes in the last 20 days. Flights with a search history of 28 days and 90 days were chosen. The number of user searches significantly increased in the last month before the departure date. The selection criterion offers a satisfactory number of pricing curves. The regular airlines have an average occupancy rate of around 75% and the percentage for the low-cost airlines are around 80%. According to the Deloitte group, the average occupancy rate of flights departing from Shanghai was 79.7% in 2018. We therefore believed that a large majority of flights still have seats until the last minute.

2.2.2 Behaviors of trajectories

As an illustration, consider the time series in Figure 2.1. It describes the price change of a Shanghai-Beijing flight departing on February 25, 2017 operated by Air China (the supplier) and purchased through the Qunar (the provider), for a 3-day trip. The price varies from one plateau to another, triggered by the yield management system as described in the previous chapter.

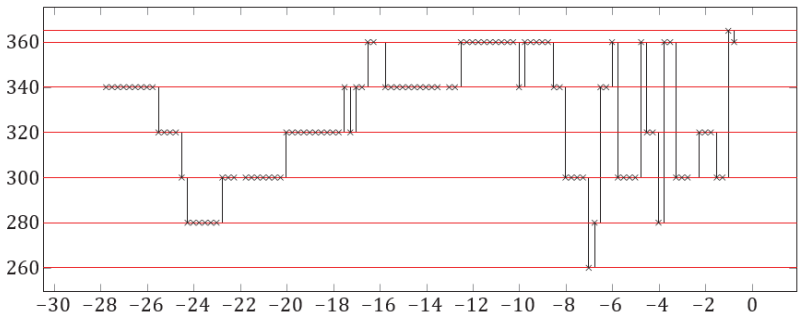

Figure 2.1 Flight Shanghai-Beijing, departure on 25/02/2017 for 7 days operated by Air China and sold by Qunar.

The commercial sites (provider) were divided into three categories: travel agencies, regular airlines, and low-cost carriers. We studied the distributions based on flights sampled every 6 hours during the last 28 days (Figure 2.3) and based on flights at 90 days (Figure 2.2). Based on 4 points per day,

the low-cost carriers have on average 18 trays in 28 days indicating a significant frequency of price changes, much more than the regular companies which have about 8 trays. The platforms often correspond to the number of classes per cabin, these are much more important for low-cost. There is a similar increase for travel agencies, but this is explained by the additional tax that some agencies apply depending on the time of ticket purchase. These multiple changes during the day then definitively increase the number of trays.

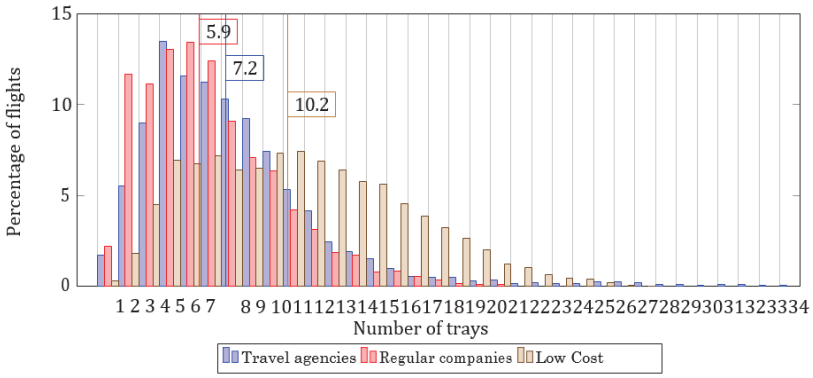

FIGURE 2.2 90-day flight base

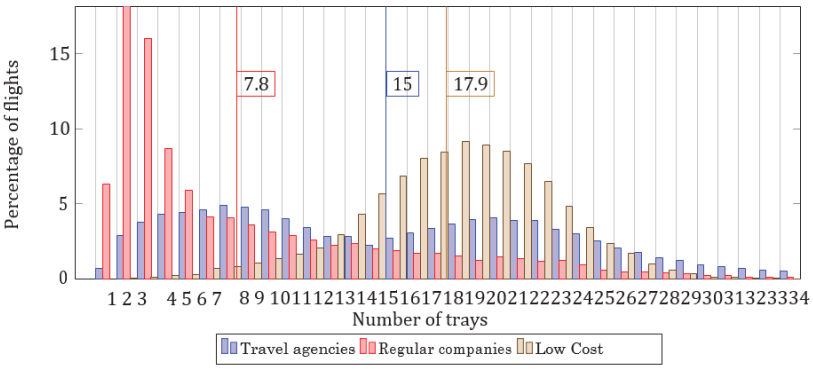

FIGURE 2.3 28-day flight base

2.2.3 Interpolation method of trajectory planning

It is practical to consider the price series until the date of departure of element i as a constant function by pieces $p_i(t)$ of continuous time $t \in [T_0^{(i)} - T_{init}, T_0^{(i)}]$, where $i \in I$ denotes the identifier id_flight of the series, $T_0^{(i)}$ is the start time (in days) of element i and T_{init} is the number of days between the start date and the first point collected. We then introduced the jump instants $T_k^{(i)}$ numbered in ascending order so that $T_0^{(i)}$ is the starting date. For each i, P_i represents the purchase price of the trip at time t. More precisely, these data will be represented in the form of a matrix P_i, $n_i \times 2$, such that $P_i(1, k)$ represents the time separating the date of departure from the date of purchase of the kth price observed and $P_i(2, k)$ the corresponding price. We assume by convention that the price is continuous on the right, hence the interpolated price curve defined for all $t < T_0^{(i)}$ by:

$$p_i(t) = \sum_{k \leq 0} p_i\left(T_{k-1}^{(i)}\right) 1_{\left[T_{k-1}^{(i)}, T_k^{(i)}\right]}(t).$$

with for all $k \leq 0$, $T_{k-1}^{(i)} = T_k^{(i)} - \delta_{-k+1}$. We denote by $p_i(t-)$ the limit of pi (s) when $s \uparrow t$.

In the example of Figure 2.4 (Shanghai-Hongkong, departure on 11/01/2019 for 14 days with Air China), there are two areas where no data has been found. Given the nature of our curves, we therefore decide to consider that the price has been constant during this period. This interpolation of the trajectories is applicable only for a small number of missing points and spaced to minimize the errors of approximation. Therefore, the proper sampling of our flights is a crucial point in the identification of trajectory behaviors.

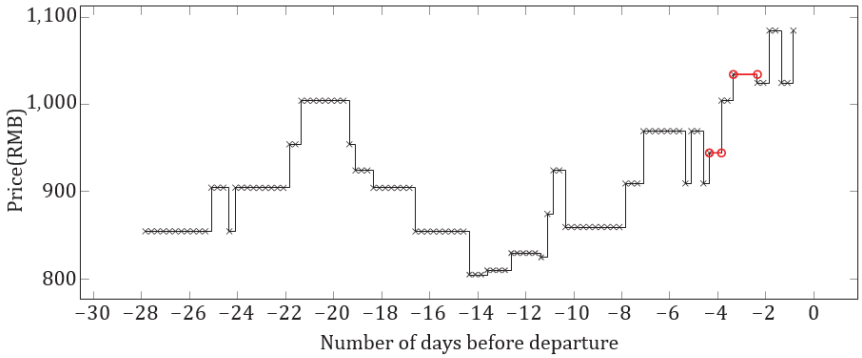

FIGURE 2.4 - Time Series: Shanghai-Hongkong departure on 11/01/2019 for 14 days by Air China

2.3 Representation by ad hoc processes

Once the time series had been constructed, we compared them to extract typical behaviors. Since the price scale is not the same for all routes, we transformed the price series into series of relative variations. We then define the following returns:

$$s_k^{(i)} = \left\{Pi\left(T_k^{(i)}\right) - Pi\left(T_k^{(i)} -\right)\right\} / Pi - \left(T_k^{(i)} -\right), \; k \leq 1$$

where p_i (t-) denotes the price just before time t. It is obvious that the price curve can be entirely reconstructed from the initial price and from the series of points $(T_k^{(i)}, s_k^{(i)})$. The calculation at t_1 of a price for t_2 is formulated as follows:

$$P_i(2, t_2) = P_i(2, t_1) \prod_{k \in [t_1, t_2]} \left(1 + s_k^{(i)}\right)$$

This sequence initially excludes absolute price values: through relative price jumps, we can therefore compare journeys with completely different price orders to aggregating series of similar behavior. During this stage, it will be possible to “correct” abnormal price fluctuations: those of small and frequent amplitudes and those of high intensity. The micro-variations vary several times a day due to additional commissions. The travel agencies apply yield management to encourage Internet users to book at “off-peak” hours and optimize their margins. It is worth mention that the agencies can negotiate tariffs with the airlines and offer cheaper tickets. These tickets undergo numerous daily variations at less than 1% value which unnecessarily disturb the price curves. The second fluctuations which alter the trajectories are large, punctual price jumps, followed by a return to the previous price in the following interval.

2.4 Modeling by ad hoc processes

After abstracting from price orders, we approximated the moment of price jumps to bring flights with similar behavior together which shifted in time. The most classic example is the travel agency which sells a ticket from an airline company and follows the variations with a certain time lag. We

therefore decided to approximate the times of the jumps and their performance of the time series.

2.4.1 Intensity estimation

We therefore model the previous representation by a point process marked inhomogeneous (fish process whose events are weighted), whose intensity X_i (s, t) can be estimated in the form of a pixelated image which takes the values:

$$\hat{X}_i(s,t)=\frac{1}{b_t b_r}\sum_{k\leq -1} 1_R\left(T_k^{(i)}, s_k\right),(s,t)\in R,$$

for a rectangular pixel R of size b_t,b_r. The time/yield plane is partitioned by a regular grid of such pixels, the intensities of which are equal to the number of hops per unit area in the time/yield plane. In Figure 2.5, we observe the transformation of the example yield series into a gray level. The greater the number of jumps in the area of the box, the darker it is.

An estimator of the intensity of the jumps and the density of the returns is written, for a bandwidth $b = (b_t, b_r) \in (0, \infty)$ and a kernel K: [0,1] → R+ such that $\int K = 1$,

$$X_i(s,t)=\frac{1}{b_t b_r}\sum_{k\leq 1} K(\{s-s_i(-k)\}/b_r)K\left(\left\{T_k^{(i)}-t\right\}b_t\right),s\in[0,1],t\leq T_0^{(i)}.$$

Note that the pixel limit is found in the lower left corner so that a yield exactly equal to a multiple of b_r will be considered to belong to the upper cell. Similarly, a yield appearing at an instant t multiple of b_t will be included in the rightmost cell. We avoid the edge effects at t ~ T (i) and s ~ 0.

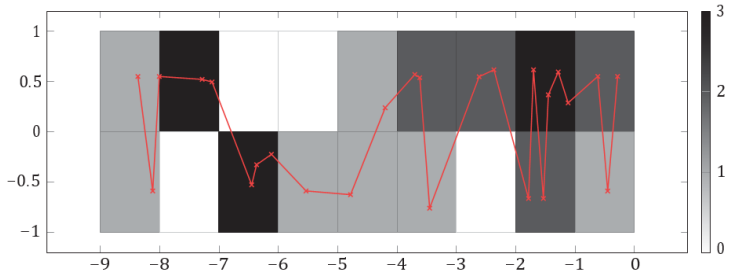

FIGURE 2.5 Grayscale: Shanghai-Hongkong departure on 01/11/2019 for 7 days by Air China

The smoothing of the small variations during the stage of transforming the time series into series of returns limits the intensity of the boxes around 0 thus avoiding confusion between a small jump and a negligible noise. As explained above, it is important to distinguish a significant price drop in price from a small variation to offer relevant decision support. Likewise, the price of a flight can suddenly double, causing a colored "pixel" to appear on a high line. This situation can arise when an economy class ticket is no longer available, and the same business ticket is offered. The gray level being a grid of the time-yield space, this generates the appearance of many empty boxes and propagates the resizing to all pixelated images. In fact, gray levels require matrices of equal dimensions implying filling with empty boxes of low-yield flights. The matrices used for clustering then become much too large and unnecessarily sparse. These events being rare, we can easily keep the jumps of low returns in the same order of magnitude and reduce the jumps with high returns by applying the natural logarithm. We then redefine the sequence of yields as follows:

$$s_k^{(i)} = ln\, Pi\left(T_k^{(i)}\right) - ln\, Pi\left(T_k^{(i)}-\right), k\leq 1$$

transforming the calculation at t_1 of a price for t_2 into:

$$P_i(2,t_2)=P_i(2,t_1)\prod_{k\in[t_1,t_2]} e^{s_k^{(i)}}$$

2.4.2 Choice of bandwidth

b_r and b_t are primordial criteria which generalize the behaviors of the series without standardizing them.
b_r represents the intensity interval in which jumps of similar intensity will be grouped.
b_r = 0.1 means that all relative jumps are grouped in 10% intervals.
b_t represents the temporal division of the gray levels. Choosing b_t=72 hours will create 3-day time windows in which all the jumps will be grouped.

Figure 2.6 provides an example of fine subdivisions of b_r and therefore price ranges. Graph 2.2 (a) is a binary division of the yield axis where only the direction of variation is considered (b_r=1). This division, when it comes to predicting only an increase or a decrease will be important because by simplifying the gray levels, we simplify the grouping into similar behaviors. On the other hand, this binary representation is very sensitive to the noises of small variations, therefore here again the step of filtering is important. Then 2.15 (b) (b_r=0.1) and 2.15 (c) (b_r=0.05), the intervals are gradually reducing the approximations of the gray levels.

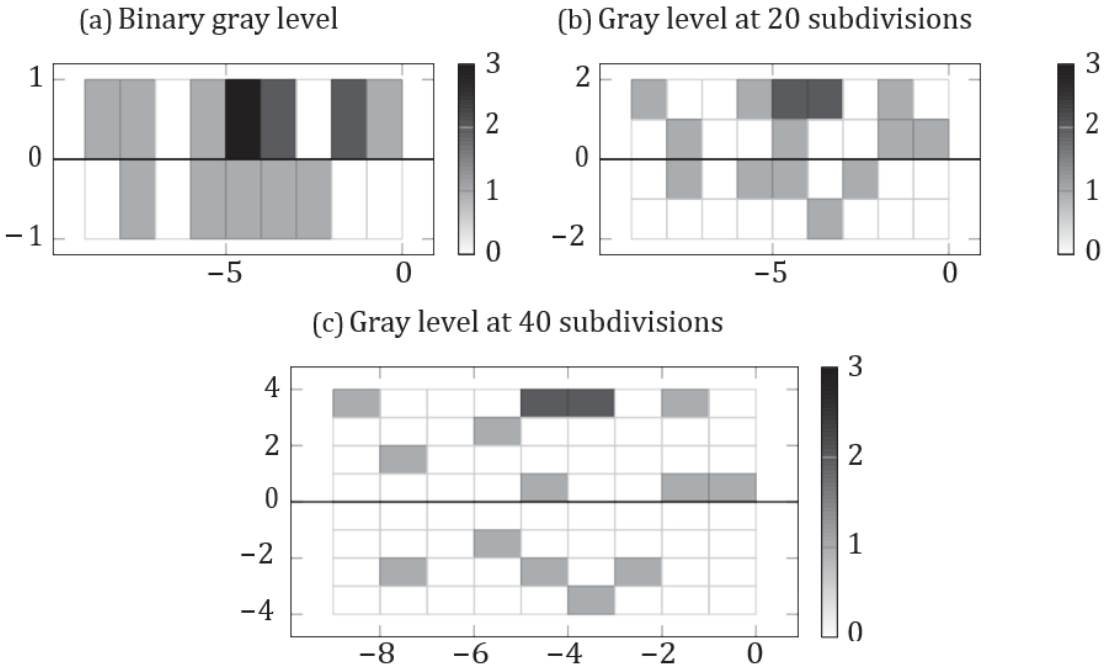

FIGURE 2.6 Evolution of gray levels with the enlargement of vertical subdivisions

2.5 Simulation

A trajectory can be reconstructed from an initial value of the price p_t at a given time t (before T_0) and knowledge of the points (T_k, s_k) such that $t \leq T_k < 0$.
We therefore simulated the point Poisson process (PPP) $N = \sum_k \delta\,(T_k, S_k)$ on a given domain D, assuming that

$$\int_D X(s,t)ds\,dt < \infty$$

The general method consists in simulating M ~ Poi ($\int_D X(s,t)ds\,dt$) then (Y_k) $_{k\leq -1}$ i.i.d. of density X (s, t)/ $\int_D X(s,t)ds\,dt$ on (s, t)∈D, and independent from M. Therefore, the process

$$N = \sum_k \delta_{Y_k}$$

is a PPP of intensity X over D. The difficulty therefore rests in the simulation of a sequence of a given density. For a general X, the rejection method facilitates the simulation process. In the simple case where X is constant by pieces on pixels of D, the following method will give a simulation at lower cost.

$$D = \bigcup_i D_i$$

with disjoint D_i and X (s, t) constant for (s, t)∈D_i. We denote by X_k the value associated with the pixel D_i. It then suffices to simulate the restriction $N^{(i)}$ from N to D_i for each i. Each $N^{(i)}$ is a homogeneous PPP on the domain D_i. We can therefore simulate it independently for each i as follows:

$$N^{(i)} = \sum_{k=1}^{M^{(1)}} \delta_{(T_k^{(i)}, s_k^{(i)})}$$

with $N^{(i)}$ ~ Poi (X_i) and $(T_k^{(i)}, s_k^{(i)})$i.i.d. on D_i and independent of $N_0^{(i)}$.

In Figure 2.7, we observe an example of transformation of a time series into a series of yields and then into a pixelated image. From this last image, a yield curve is simulated and applied the reconstruction formula of a time series to display in red. We were therefore able to create curves with statistical behavior from a gray level.

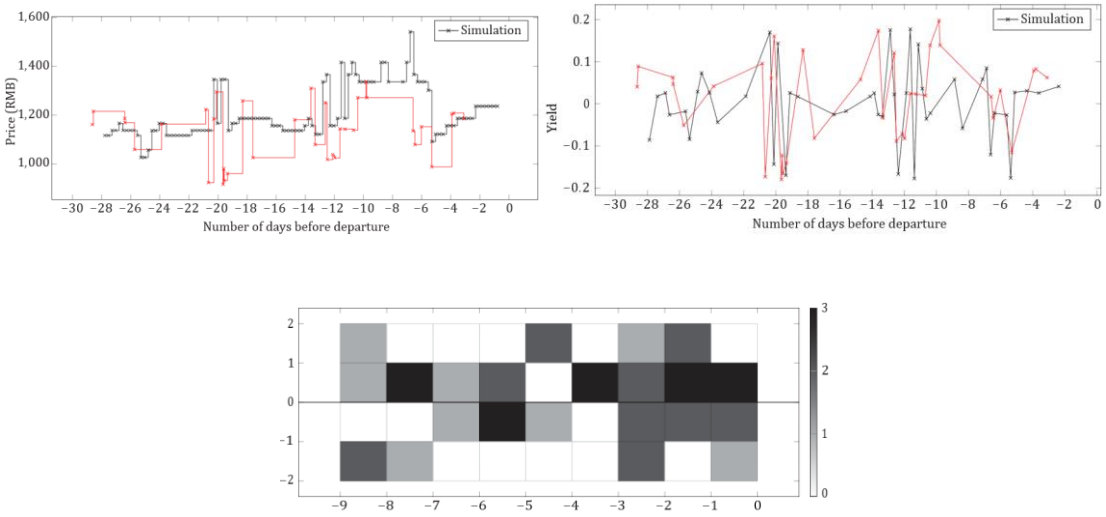

FIGURE 2.7 Transformations of the initial series and simulation by the gray level. Shanghai-Hongkong departure on 01/16/2019 for 7 days

2.6 Conclusion

This chapter describes the nature of the time series.-The average waiting time between two jumps at different number of days before the departure date were studied. It can be concluded that: (1) in the interval [-28, 0], the construction of a time series requires the collection every 6 hours and (2) in the interval [-90, -28], a daily collection guarantees the price variation detection. An innovative method to represent the time series of prices was introduced. The price series were transformed into return series to extract price orders. These series of returns by point processes were visualized through gray level pixels representing the intensity of the price jumps. The yield axis was logarithmically transformed to avoid high yields. The configuration of pixelated images was essential to generalize behaviors. Although the representation is not bijective, a similar price series can be reconstructed by performing simulations from the gray level. The different behaviors in the learning base were first identified, followed by unsupervised learning algorithms to segment the gray level into homogeneous groups. Finally, test flights were associated to establish an evolution prediction at instant t.

Chapter 3

Introduction

This chapter describes different algorithms extracting groups of similar behavior through pixelated images of intensities. The K-Means data partitioning algorithm were adopted first to create pixelated images, followed by the Expectation-Maximization (EM) algorithm. The K-Means minimizes the distance of a group to the center, while the EM maximizes the likelihood of group members using the same settings. The two main parameters of the two approaches are the number of groups and the number of initializations. The number of groups is an important variable in the segmentation step. Metrics such as GAP statistics were applied to choose the optimal number of groups. To avoid local minimum, the random departures were multiplied and the best segmentation was chosen after the optimization. An algorithm aggregates random subsets of the learning basis was conducted to reach the global optimum. This "bagging" technique applied to K-Means is called "Bagged K-Means".

With our new representation of the trajectories X_i (s, t), the behavior of all types of flights were compared. The similarity of two time series were evaluated by calculating the Euclidean distance of the gray levels. Then unsupervised learning algorithms were applied to group time series with similar behaviors. The learning data was segmented to create sets of similar behaviors and associate representative behaviors. The average gray level, named centroid and was denoted by I_y, $y \in 1, ..., C$. $p_y(t)$ curves belonging to the group y were simulated. It predicts the evolution of a test flight after assigning it its most likely average behavior.

3.1 Notations

i: Flight number of the learning base $i \in 1, ..., n$
p: Number of attributes
V_i: Vector of p attributes of flight i among the set of attributes
N_{Train}: Learning base for attribute vectors of size $n \times p$
n: Number of flights
R: All the boxes
b_r: Vertical dimension of the boxes (yield)
b_t: Horizontal dimension of the boxes (time)
$X_i(s, t)$: Grayscale of the time-performance plan for the flight i in the box (s, t)
M_{Train}: Learning base for grayscale of size $n \times (s \times t)$
$I_1, ..., I_C$: Centroids of the clusters in the form of pixelated images: $I_y(s, t)$, $(s, t) \in R$
C: Number of clusters
$\alpha_y = P(Y_i = y)$, $y = 1,2, . . . ,C.$
$\psi (y|V_i) = P(Y_i = y|V_i)$, $y = 1 , 2 , . . . , C .$

3.2 The algorithms

Three algorithms are described to partition the data in this section. Two of these algorithms, the K-Means and the Expectation-Maximization algorithm, are part of the iterative optimal segmentation following two stages: creation of models and reassignment of data. The algorithm stops after a predefined number of iterations, or when the reassignment changes no longer. Then the Bagged K-Means was applied to random subsets of the learning base.

The K-Means algorithm represents each class with a centroid calculated by averaging all the group's gray levels. These centroids are denoted I_y for $y \in [1, ..., C]$ with C the number of groups fixed in advance. The reassignment step then chooses the nearest centroid due to a predetermined distance measurement. The Expectation-Maximization algorithm models each class by a probability distribution created during the class creation step. The reassignment maximizes the likelihood of flights to the centroid. A gray level corpus $X_i(s, t)$, $i \in [1,n]$ was denoted by M_{train}.

3.2.1 K-Means

Groups with similar behaviors was created by applying the K-Means segmentation algorithm based on pixelated images of intensity $\hat{X}_i$ for i traversing the learning base. K-Means require initialization to start the iterative optimization phase. This is an important step in the algorithm which can be commonly performed in two different ways: the Forgy method and random partition method. The first method randomly chooses C points from the learning base as the centroids and then assigns the other flights to the nearest centroid. The random partition approach assigns a random cluster to each flight. As for the Expectation-Maximization algorithms and standard K-Means, the Forgy approach was recommended.

The centroids were then built by averaging the flights of each group. The algorithm followed the iterative optimization process by reassigning its nearest group to all flights and calculating the centroids again. We thus obtained C index $I_1, \ldots, I_C$ to group each learning base by similar behaviors. This number C was chosen according to certain performance criteria such as the group density or the prediction rate. To obtain the optimal number of classes, criteria such as the GAP statistic or the Calinski-Harabasz index were described.

Input data
All our grayscale X_i is the only entry point to the algorithm. To minimize their size, the pixelated images resulting from the transformation by the natural logarithm of the yield series were adopted. These gray levels were compared with equal dimensions. The parameters of the algorithm were the number of groups C, the number of random initializations *nstart* and the maximum number of iterations *iter.max.*

Algorithm description
As described above, the first step is to create a random starting point for the iterative optimization process. Forgy's approach was applied to choose C centroids randomly from the input data. The calculations were repeated several times (*nstart*) to retain the optimal solution for the chosen criterion. The most likely centroid was calculated by minimizing the Euclidean distance. The K-Means aims to minimize the variance within each group:

$$E_i^{(t)} = arg_Y min \sum_{i=1}^{n} \sum_{X_i \in Y_i^{(t)}} \left\| X_i - I_y \right\|^2$$

with $Y = \{Y_1, ..., Y_C\}$. The distance obtained between the trajectories was based on the distance L^2, the corresponding densities for given b_t and b_r:

$$d(i_1, i_2) = \left(\int_{s=0}^{1} \int_{t=T_{min}}^{0} \left[X_{i1}\left(s, T_0^{(i_1)} + t\right) - X_{i2}\left(s, T_0^{(i_2)} + t\right) \right]^2 ds dt \right)^{1/2}$$

where T_{min} is the date of the first available price at T_0. The centroids from the flights assigning to different classes were redefined:

$$I_y^{(t+1)} = \frac{1}{\left| Y_i^{(t)} \right|} \sum_{X_i \in Y_i^{(t)}} X_i$$

Flights were then reassigned according to their distance to the new centers until convergence was reached, or until the maximum number of iterations *iter.max* was reached.

Output data

A segmentation of the gray level was obtained after the K-Means process. Each flight of the training base was assigned to the group number with the closest $E_i \in 1, ..., C$ called "label". A representation of "typical" behavior was created by averaging the gray levels of all the group's flights for each group. The segmentation quality of our data was tested using several metrics, given that the quality was defined by a high out-sample similarity and a low in-sample similarity.

The flight distribution validates the homogeneity of the groups. An unbalanced distribution indicates a unsatisfactory cluster numbers or representative behaviors in the learning base.

$$n = \sum_{i=1}^{C} n_i$$

The sum of the distances to the centroids for each group, w_y, quantifies the overall distance between the flights in the group and the associated centroid. The centroid identifies the group's flights sharing the same nature. The reduction in the number of clusters creates less dense and fewer representative groups. The quantity is written:

$$w_y = \sum_{X_i \in y} \left\| X_i - I_y \right\|^2$$

$$W = \sum_{i=y}^{C} \sum_{X_i \in y} \left\| X_i - I_y \right\|^2$$

This metric selects the best performing initialization.

The distance between the different centers of the groups is defined by:

$$b_y = n_y \left\| I_y - I \right\|^2$$

$$B = \sum_{y=1}^{C} n_y \left\| I_y - I \right\|^2$$

where *I* refers to the average behavior of the database.

3.2.2 Bagged K-Means

The "Bagged K-Means" performed the bagging principle to the K-Means algorithm. Bagging applies the same algorithm multiple times to different subsets of the learning base and aggregates the results. An ascending hierarchical segmentation algorithm was conducted to aggregate the

segmentation and improve the K-Means results. This algorithm applies to a distance matrix between the centroids from the bootstrapping step.

The Bagged K-Means algorithm was proposed by Yordan:

1. Construct B subset of the M_{train} learning base, M^1_{train}, ..., M^B_{train} by random draw with (bootstrap)
2. Apply a segmentation algorithm K-Means on each subset to obtain B×C groups c_{11}, c_{12}, ..., c_{1C}, c_{21}, ..., c_{BC} where C is the desired number of clusters and c_{ij} the j -th group of the subset M^i_{train}
3. Create a new Y^B (C) base made up of all the centers $Y^B = Y^B$ (C) = c_{11}, ..., c_{BC}

Algorithm 1 Partitioning through the K-Means algorithm

For a desired number of initializations nstart **then do**

Initialization of Forgy: Random selection of C centroids among the n gray levels

If the allocation of flights changes **OR** the maximum iteration not reached iter.max **then do**

Assignment step: A group is assigned to each flight according to its proximity to the group's centroid at iteration t.

$$Y_i^{(t)} = \left\{ E_i^{(t)} : \left\| X_i - I_y^{(t)} \right\| \leq \left\| X_i - I_y^{(t)} \right\| \forall 1 \leq y \leq C \right\},$$

The flight Xi is assigned the label $E_i^{(t)}$ corresponding to the most probable group Y(t) based on the Euclidean distance.

Update step: Calculate new centroids

$$I_y^{(t+1)} = \frac{1}{\left| Y_i^{(t)} \right|} \sum_{X_i \in y_i^{(t)}} X_i$$

End If

End For

return E_i Label associated with each flight, ie. identifier of the nearest centroid

return n_y Number of flights per group

return w_y Sum of distances to the centroid by group

return I_y Centroids of groups

4. Prune Y^B by applying the partitioning algorithm M gear with the centers Y^B and removing the empty centers:

$$y^B_{prune}(C, \theta) = \{ c \in y^B(C) | \#\{x : c = c(x)\} \geq \theta \}$$

5. Apply the algorithm for clustering hierarchical described more down on Y^B
6. Let c (x) $\in Y^B$ be the center closest to x . A segmentation of the original base can be obtained by "cutting" the dendrogram at a certain level. We thus created a partition Y^B_1, ..., Y^B_m, 1 ≤ m ≤ B C of Y^B where each point x $\in M_{train}$ is associated with its cluster the closest c(x).

Hierarchical segmentation

The ascending hierarchical classification starts from all the individuals representing a class, then gathering in increasingly large classes. The qualifier produces a hierarchy H with the following properties:

1. Ω ∈ H: at the top of the hierarchy, all individuals are grouped within the same class.
2. ∀ω ∈ Ω, {ω} ∈ H: at the bottom of the hierarchy. ∀ (h, h ′) ∈ H^2, h ∩ h ′ = ∅ or h ⊂ h ′ or h ′ ⊂ h.

The different centroids segmentation created during the bootstrapping step were described in Algorithm 2

Algorithm 2 Hierarchical segmentation algorithm
Pre-conditions: M_{Train} list of individuals X_i, i ϵ1, . . . , n
Pre-conditions: C number of classes to obtain
Pre-conditions: Cur_y with y ϵ [1 , ..., nbC], nbC the number of current classes.
For i ϵ [1 , ..., n] **then do**
$Cur_i = X_i$ becomes a new class
nbC++
End For
If nbC> C **then do**
Calculation of dissimilarities between classes in an upper triangular matrix
For i ϵ [1 , ..., nbC] **then do**
For j ϵ [i + 1 , ..., nbC] **then do**
matDissim [i] [j] = dissim (Cur_i , Cur_j);
End For
End For
Finding the minimum dissimilarities
Let (i, j) such that matDissim[i] [j] = min(matDissim [k] [l]) with $1 \leq k \leq nbC$ and $k + 1 \leq l \leq nbC$
Merger of Cur_i and Cur_j
For any element in classes [j] **then do**
Cur_i.ajouter(element);
End To delete (Cur_j);
End If
return classes: initially empty list of classes, a class is seen as a list of individuals

The main idea is to stabilize the segmentation of K-Means by repeating the partition and combining the results. The K-Means are unstable because each algorithm provide different local optimal for the same data and parameters. The initialization stage has a great influence on the convergence of the algorithm. A slight modification of the training data may make the K-Means converge to a completely different optimum. Different solutions were obtained through the repeated learning on subsets. The process was independent of the initial learning base and the number of initializations.

3.2.3 Expectation Maximization

The Expectation Maximization (EM) algorithm aims to maximize the probabilistic models comprising unobserved variables. The principle of EM was applied to estimate the parameters with mixed densities by automatic classification. The flights $i \in \{1, ..., n\}$ were characterized by their gray level X_i (s, t) from C different groups. Supposing that a flight i belongs to the group $y \in \{1, ..., C\}$, $Y_i = g$, the X_i (s, t), (s, t)∈R are realizations of a Poisson point process of intensity given by the centroids I_y (s, t), (s, t)∈R. Furthermore, the proportions of the groups are given by a vector (α_1,..., α_C) where $\alpha_y = P(Y_i = y)$ ∀y∈{1, ..., C}. We noted $\alpha = (\alpha_1, ..., \alpha_C)$ the element of the simplex

$S_C = \{(\alpha_1, ..., \alpha_C) \in [0, 1]^C, \Sigma^C_{y=1}\alpha_y = 1\}$. The parameters of the model were grouped under the notation $\theta = ((\alpha_y)_{y \in \{1, ..., C\}}, (I_y(s, t))_{(s, t) \in R}) \in [0, 1]^C \times (R+)^R$

The α_k represents the grouping probability ($\alpha_y = P(X \in C_y)$) and I_y represents the law of observations, given a group label y.

Input data

The input data are the same as for the K-Means, i.e. all of our gray levels X_i (s, t), $i \in \{1 , ..., n \}$, $(s, t) \in R$.

Initialization

This process assigns an initial value to the I_k (s, t) and the αk. Two approaches were conducted. The first approach assigns values randomly to each of the parameters. The second method initializes the values by applying the K-Means algorithm. The creation I_k (s, t) randomly assigns a group number to each Xi, then averages all the flights for each group. Regarding the αk, the initialization methods were proceed as follows:

$$\alpha_y = \frac{card(X_i \in y)}{n}$$

Likelihood function

The law of the table is $X_i = (X_i(s, t))_{(s, t) \in R}$, given that the label of its associated group $Y_i = y$ is:

$$P_{I_y}(X_i(s,t) = x(s,t), (s,t) \in R | Y_i = y) = \prod_{(s,t) \in R} \frac{\left(I_y(s,t)\right)^{x(s,t)} e^{-I_y(s,t)}}{x(s,t)!},$$

where $x(s, t) \in N$, $y \in \{1, ..., C\}$ and $I_y(s, t) \in ((R)^R)^C$.

C represents the number of groups. As for a label $y \in \{1, ..., C\}$ and a given cell $(s, t) \in R$, $I_y(s, t)$ represents the centroid of the group y in cell (s, t). I_y is a pixelated intensity of the Poisson process for the flights of the group y. Consequently, the $X_i(s, t)$ are the independent Poisson variables of intensity $I_y(s, t)$. The probability of (X_i, Y_i) were calculated, given the parameters $\theta = (\alpha, I) \in S_C \times (R^R_+)^C$. We obtained:

$$P_\theta(X_i = x, Y_i = y) = P_{I_y}(X_i = x, Y_i = y) P_\alpha(Y_i = y) = \alpha_y \prod_{(s,t) \in R} \frac{I_y^{x(s,t)}(s,t) e^{-I_y(s,t)}}{x(s,t)} =: p_\theta(x, y),$$

where $x \in N^R$ and $y \in 1 , ..., p$.

Expectation Step

This step calculates the expectation of the density attached to the parameter θ' for the conditional law of the latent variables Y_i, the observed variables X_i, and the parameter θ:

$$Q(\theta', (x_i)_{i \in \{1,...,n\}} | \theta) = \sum_{i=1}^n \sum_{y_i=1}^C [log\, p_{\theta'}(x_i, y_i)] p_\theta(y_i | x_i)$$

We then have for $\theta' = (\alpha' , I_y')$:

$$log\, p_{\theta'}(x, y) = \sum_{i=1}^n log(\alpha'_{y_i}) + \sum_{i=1}^n \sum_{(s,t) \in R} \left[x(s,t)\, log\left(I'_y(s,t)\right) - I'_y(s,t) \right]$$

The conditional probability was written:

$$p_\theta(y|x) = P_\theta(Y_i = y | X_i = x) = \frac{P_\theta(X_i = x, Y_i = y)}{\sum_{k=1}^C P_\theta(X_i = x, Y_i = k)} = \frac{p_\theta(x,y)}{\sum_{k=1}^C p_\theta(x,k)}$$

Therefore, we had by noting $X = (X_i)_{i=1 , ..., n}$:

$$Q_n(\theta'|\theta) := Q(\theta', X|\theta) = \sum_{l=1}^{n} \sum_{y=1}^{p} log(\alpha'_y) p_\theta(y|X_l)$$

$$+\sum_{l=1}^{n} \sum_{(s,t)\in R} \left[X_l(s,t) \sum_{y=1}^{C} log\, I'_y(s,t) p_\theta(y|X_l) - \sum_{y=1}^{C} I'_y(s,t) p_\theta(y|X_p)\right]$$

For simplification put ∀ y∈1 ... p:

$$A_n(\theta, y) = \frac{1}{n} \sum_{i=1}^{n} p_\theta(y|X_l)$$

$$B_n(\theta, y, s, t) = \frac{1}{n} \sum_{i=1}^{n} X_l(s,t) p_\theta(y|X_l)$$

Note that B_n depends on s and t with an independent value for each box while A_n was only indexed by y∈{1, ...,C}. We then obtained:

$$Q_n(\theta'|\theta) = \sum_{y=1}^{C} log(\alpha'_y) A_n(\theta, y) + \sum_{(s,t)\in C} \sum_{y=1}^{C} log\left(I'_y(s,t)\right) B_n(\theta, y)$$
$$- \sum_{(s,t)\in R} \sum_{y=1}^{C} I'_y(s,t) A_n(\theta, y)$$

Maximization Step

Step "M" is to maximize $Q_n(\theta'|\theta)$ in θ ′ for a given θ. Optimization of α_k and I^k were applied separately.

For a fixed θ, A_n, B_n, we obtained:

$$\alpha'_y = \frac{A_n(\theta,y)}{\sum_{k=1}^{C} A_n(\theta,k)}, \forall_y \in 1 \dots C$$

$$I'_y(s,t) = \frac{B_n(\theta,y,s,t)}{A_n(\theta,y)}, \forall_y \in 1 \dots C, \forall\, (s,t) \in R$$

Output data

EM numerically calculated the parameters (α, I) = θ which locally maximize the likelihood at least., An estimator θˆ=(αˆ, Iˆ) was obtained when the algorithm stopped. A calculation of the joint density $p_\theta(x, y)$ was provided from the following two circumstances:

1. the log-likelihood $L_n(\theta) = \log \Sigma^{C}_{y=1} p_\theta(X, y)$
2. the log-likelihood to the cluster $L_n(\theta, y) = \log(p_\theta(X|y)) = \log(p_\theta(X, y)) + C$

Unlike the K-Means, the EM does not require an estimated label Y_i for each learning flight. This group were computed for the likelihood function $p_\theta(x,y)$ and the previously optimized parameters. It was applied to $X_i(s, t)$ (α_y and $I_y(s,t)$) to determine the label Y_i for each flight i. The same metrics were applied to evaluate data segmentation and choose the right parameters.

3.3 Choice of parameters

This section explains different parameters influencing the results of the segmentation and the prediction of the test flights. The corresponding modifications of these parameters were discussed in the next chapter.

3.3.1 Initialization

The algorithms described are sensitive to initialization and converge differently at each iteration. The initialization type contributes to the segmentation quality. Several clustering algorithms were compared on a large data set. Different initialization types were then tested on the algorithm. In our case, the EM were initialized by the K-Means segmentation and compared with the random zation in the next chapter.

3.3.2 Number of groups

Various techniques exist to estimate the number of groups. Milligan and Cooper (2009) studied 30 methods to estimate the optimal number of groups on simulated data. The Calinski and Harabasz index is most effective according to their research. Yan (2005) then studied these different methods and proposed the comparison with the GAP index introduced by Tibshirani et al. The results show that the GAP index can better estimate the optimal number of groups. Different methods adopted in our study are described below.

Evolution of the RAND index

Rand's index is a metric commonly used in grouping data. It measures the similarity between set partitions. The principle verifies each pair of objects and tests whether they are classified in the same group. Consider two partitions of the gray level space P_1 and P_2, we have:

a - the number of pairs present in the same group in P_1 and in P_2

b - the number of pairs in different groups in P_1 and in P_2

c - the number of pairs present in the same group in P_1 and in different groups in P_2

d - the number of pairs in different groups in P_1 and in the same group in P_2

Rand's index is then written:

$$R = \frac{a + b}{a + b + c + d}$$

By changing the number of clusters or the number of initializations, the evolution of the Rand index were observed.

The Calinski-Harabasz index

The optimal number of clusters will be determined when the index is maximized. The calculation is as follows:

$$CH^{(y)} = \frac{b_y \,/\, (C-1)}{w_y \,/\, (n-C)}$$

where b_y and w_y are the metrics described in section 3.3.1. As with the RAND index, the "bend" will be detected in the index curve according to the number of groups.

GAP statistics

Tibshirani et al. proposed to determine the optimal number of clusters based on the GAP statistics. This method can be applied to any segmentation algorithm using any distance measure. The idea is to compare the evolution of W(k) as the number of clusters k increases.

Visualization

The data partitioning can be visualized by performing a projection on the axis of two variables from

the principal component analysis (PCA): the two new variables are a linear combination of the I_y (s, t). The basic idea of the PCA is as follows: considering the cloud of n points in P dimensions, a two-dimensional plane of the cloud points is the least distorted possible. Therefore, the line δ_1 minimizes the sum of the squared distances between each point. Then, a second line δ_2 perpendicular to δ_1 has the same property. The process continues until P straight lines were obtained form an orthogonal coordinate system.

3.4 Conclusion

The step of partitioning all flights into groups of similar behaviors were described based on our new representation of time series. Two well-known clustering approaches, K-Means and the Expectation-Maximization algorithm, were conducted. The K-Means algorithm represents each class by a centroid averaging all the flights. At initialization, these centroids were chosen randomly from gray levels in the learning base. The reallocation of flights was then constructed by minimizing the Euclidean distance to the centroids. These two steps were then iterated until the groups no longer move, or the limit of iterations was reached. The distance between the pixelated images is the sum of the Euclidean difference. One method applied the K-Means to a large number of boosted learning base and then aggregated the results using a hierarchical segmentation algorithm. The Bagged K-Means approach erases the influence of initialization and stabilizes the results against the changes in parameters. The Expectation-Maximization algorithm modeled each class by a probability distribution and groups the gray levels to maximize the likelihood t the centroids. The initialization of the parameters was performed randomly for global optimal.

For K-Means and EM, initialization is an important step that should be repeated many times. Similarly, the choice of the number of clusters, which is defined prior to the launch of the algorithms is essential. The process required observing criteria such as the Gap statistic or the Rand index to choose the optimal number. As the database grows, the data segmentation updating was performed to identify additional behaviors. The stability of the behaviors grouping and the reproducibility of predictions were guaranteed. Given the gray levels of our learning base, the most likely family was assigned to its first price changes. Therefore, a supervised learning algorithm was applied to the segmented data.

Chapter 4

Introduction

The learning base was segmented into groups of similar behaviors by a set of parameters (α_k, I_k). The search results were assigned to the most likely behavior based on the attributes distribution. Each user search has a list of results corresponding to the input parameters: date of departure, date of return, city of departure and city of arrival. Each result has a list of attributes specific to its flight and the site offering the flight: departure and return times, airline, departure and arrival airport, merchant site, etc.

A supervised learning algorithm was applied on the segmented data. Each flight has a series of attributes V_i and a label E_i corresponding to the typical groups. Classification rules were created through the set of attributes V_i (1), . . ., V_i(p) to assign the test flights their most likely group k$\in$ {1, . . ., C}. Several algorithms such as CART, C4.5, Adaboost, and Random Forest were used. With CART and C4.5, the attribute segmentation and the behavior prediction rules were observed. Adaboost iteratively executes CART based on the weighted learning basis with different iterations. More weights were provided to misclassified flights to improve the prediction. Random forests multiply CART decision trees and aggregate their results by vote. The results provide a ranking of the most influential attributes in the classification. This classification rules were adopted to keep the most relevant attributes.

Therefore, each user search was assigned the most probable cluster identifier to a centroid representing its overall behavior. In Chapter 2, the price curve was simulated from a pixelated image to reconstruct a series of prices. It is therefore possible to simulate set of series potentially belonging to a group by using the cluster centroid. Averaging a predefined number of simulated curves, the price evolution assigning to the similar group was estimated. A series of returns were extracted from the simulations with the following information: the increase or decrease of the price in n days, and the variation. Secondly, the “direct” prediction was explained based on the direct learning of the price evolution at time t. This prediction can be binary (increase or decrease in price), by dial (strong increase, weak decrease etc.) or continuous (percentage change). The label E_i refers to the prediction at time t involving the model by number of days before the departure date. The typical behaviors and the simulation of curves were dispensed. Finally, the advantages and the disadvantages of this approach were discussed.

4.1 Notations

i: Number of the flight of the learning base i 1, ..., n
n_{test}: Number of test flights
p: Number of attributes
V_i: Vector of p attributes of flight i among the set of attributes V
N_{train}: Base for learning attribute vectors of size n × p
N_{test} : Base for testing attribute vectors of size n test × p

Setting the date of departure at the origin, $T_0^{(i)} = 0$, the variable $\phi_t^{(i)} \in \{0 , 1\}$ is defined to be the advice "buy" and "wait" correspondingly at time t for the flight i. $\phi_{t_1}^{(i)} = 1$ if and only if the price p_i $(-t_2)$ at 7 days is less than or equal to p_i $(-t_1)$. Once the class j is obtained by applying the classifier to the attributes of the path i, the model of class j will be used to calculate the probability $P(\phi_{t_1}^{(i)} = 1)$.

4.2 Learning process

Before the price prediction of a flight at time t, $P(\phi_t^{(i)}=1)$, the belonging group should be determined. The list of attributes $V_i = V_i(1), \ldots, V_i(p)$, and the first price changes were the only input information. We initially applied the flight attributes only to predict behaviors. Classification algorithms that segment the learning base according to the label E_i were adopted to create classification rules:

CART (Breiman, 1984) is a classification algorithm based on decision trees maximizing the information gain at each step. It chooses the most discriminating attribute at each node and separates the set of flights into two subsets. Adaboost (Freund, 1995) is a boosting method to select weak classifiers and minimize the error in classification. Random forests (Breiman, 2001) are a variant of bagging to build multiple uncorrelated trees. In most cases, Random Forests have similar performance to Adaboost with easier training and configuring. Genuer (2010), provides a complete description of the theory and practice of random forests for selecting the most relevant variables to improve the classification.

4.2.1 Decision trees: CART & C4.5

The CART algorithm can be divided into 4 steps. The first step is to build a tree recursively by separating all the flights into two subsets. Each node is associated with a predicted class according to the distribution of the flights and the cost matrix. The tree over-learns the training data after construction. Therefore, the third step of pruning the tree is to make it generalizable. Finally, a test sample was performed to minimize the rate of estimated error. The construction of the tree begins with testing all the flights grouped together. The algorithm finds the best attribute to split the flights into two subsets. All the possible values of each attribute will be checked to optimize a predefined criterion. The number of levels to be tested should be specified for the qualitative attributes to avoid long computation time.

With a node m representing a sub-partition N_m, N_{train} learning base and n_m flights, to maximize the average purity of the two sub nodes at each iteration, let us set:

$\hat{p}_{mk} = \frac{1}{n_m} \sum_{i \in N_m} \mathbb{I}_{y_i = k}$,

the proportion of class k flights at node m.
At each node m, the set of flights is classified as $k(m) = argmax_k \hat{p}_{mk}$. Different metrics of a node will be defined:

Error in classification: $\frac{1}{n_m} \sum_{i \in N_m} \mathbb{I}\left(y_i \neq k(m)\right) = 1 - \hat{p}_{mk(m)}$.

Gini index: $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{C} \hat{p}_{mk}(1 - \hat{p}_{mk})$.

Cross deviance: $-\sum_{k=1}^{C} \hat{p}_{mk} \log(\hat{p}_{mk})$.

In the case of a two-class classification, these three measures are written respectively 1-(max(p, 1-p),2p(1-p) and -p log p-(1-p) log(1-p).

The Gini index, introduced by Breiman, measures impurity in the random forest algorithm. The impurity of a population for a multiclass classification is measured by the following formula:

$$I(T) = 1 - \sum_{j=1}^{c} (\frac{n_{j.}}{n_{..}})^2$$

where n_{ji} represents the membership of class j at node i. It measures the expected classification error if the predicted class were chosen randomly following the distribution of class probabilities and the numbers of the current node T. Construction is stopped when: (a) There is only one more observation in the leaf nodes (b) The distribution of attributes is identical (c) The defined depth limit has been reached. Pruning removes unnecessary tree branches to avoid over-learning the training data. The process includes building a series of sub-trees by successive pruning and choosing the optimal one among these series. The solution obtained by a step-by-step algorithm is not necessarily globally optimal with efficiency and reliability. For a given tree A denoted by K the number of terminal nodes of A, the measure is expressed by a criterion:

$$D(A) = \sum_{k=1}^{K} D_k(A)$$

where $D_k(A)$ is the number of misclassified flights.
The construction of the sequence is based on a penalty of the tree:

$$C(A) = D(A) + \gamma K$$

The process is iterated for the construction of the sequence:

$$A_{max} = A_K \supset A_{K-1} \supset \cdots A_1$$

where A_1, the root node, is the entire sample. The optimal tree therefore corresponds to k that minimizes $D(A_k)$.

The decision trees support both quantitative and qualitative data. Its computation time on a large volume of data remains low without degrading performance. C4.5 creates several branches at each separation according to the number of levels for a qualitative variable. Regarding quantitative variable, a branch is created for each interval of the discretized version of the variable. The limitations of the algorithms of CART and C4.5 are statistical, representation and computational problems, which are described by Thomas G. (2000) and summarized by Beringer and Eyke (2006). The combination of several classifiers would make decisions more reliable, expand the possible solutions, and avoid local optima. The aim of multi-classifier systems (MCS) is to create diversity within a set of successful classifiers and to establish the best possible solution.

4.2.2 Adaboost

Adaboost is a boosting technique based on the iterative selection of weak classifiers (Yoav and

Robert, 1995). Introduced by Freund and Schapire, boosting is one of the most successful machine learning algorithms. A predictor $\hat{h}(V_n)$ can be constructed given a training sample V_n and a weak prediction method (CART for example). A first sample $V^1{}_n$ is drawn randomly (bootstrap). The basic classifier can be applied to this sample to obtain a first predictor $h(V^1{}_n)$. Then, the error of $\hat{h}(V^1{}_n)$ on the sample V_n is calculated. A second bootstrap $V^2{}_n$ sample is then drawn based on the predictions of $\hat{h}(V^1{}_n)$. A set of classifiers can be obtained after repeating the process and taking a weighted average. Boosting is therefore a sequential method with each sample being drawn according to the performance of the basic rule on the previous sample. Each model is an adaptive version of the previous one by giving more weight during the next estimation. It does not require long data pre-processing of parameters during the training procedure. In our case, each learning flight is weighted at each iteration to minimize the classification error.

Algorithm 3 Adaboost

Input: $S = (V_i, E_i)_{i = 1, ..., n}$, with E_i the class obtained by applying the classifier
For i = 1 ... n do
$p_1 (V_i) \leftarrow 1/n$
End For
For j = 1 ... nb Classifier **do**
Draw a learning sample S_j in according to the probabilities p_j
Train a classifier C_j on this subsample.
Let q_j be the apparent error of C_j on S
Calculate $\alpha_j = 1/2\ln(1-\epsilon_t)/\epsilon_t$
For i = 1 ... n **do**
if $h_j (V_i) = y_i$ (well classified by h_t) then
$p_{j+1} (V_i) \leftarrow p_j (V_i)/Z_t \, e^{-\alpha t}$
if not
$p_{j+1} (V_i) \leftarrow p_j (V_i)/Z_t \, e^{+\alpha t}$
End if
With Z_t normalized such that $\Sigma_1{}^m p_j (V_i) = 1$
End For
End For
return The set of trees $\{T_b\}_1{}^B$

Boosting methods are frequently cited for comparison with random forests. Boosting is based on a deterministic principle for the creation of diversity in sets, whereas random forests adopted principles of randomization.

4.2.3 Random Forest

Random forest methods are based on the combination of elementary classifiers of decision tree types. A small change in the learning base causes a significant change in the structure of the tree and results in its generalization performance. The specificity of trees used in random forests is disturbed by a random factor to generate diversity. In 1996, Breiman proposed a method to construct multiple

independent classifiers on random subsets of the learning base. Each classifier is a CART decision tree previously described but which has not been pruned. The predictions of each tree are grouped together and the final prediction is chosen by a voting system. This method is based on a principle called bagging. In the case of bagging, the tree construction algorithm divides the nodes by choosing the best variable among all possible variables. The selection of the best variable is performed on a random subset of the variables. Breiman demonstrates the importance of random forests convergence. His research shows that the error rate in generalization of a random forest converges towards a limit value, when the number of trees increase. The main strength of bagging is therefore to reduce instability and improve performance in generalization. Random selection does not require cross validation, or even an independent test basis to estimate test error. This error is calculated as follows: For each tree built on the subset Z^*, a prediction is made on all the flights of the set. By aggregating the predictions made by all the trees, one flight will be predicted once in three on average in our case. Algorithm 4 describes the different stages of the construction.

Algorithm 4 Decision trees for classification
Input: $S = (V_i, E_i)_{i = 1, ..., n,}$ with Ei the class obtained by applying the classifier
Entry: *ntree* the number of trees in the forest
Entry: *mtry* the number of characteristics to be randomly selected at each node
For j = 1 ... *ntree* **do**
  $S_j \leftarrow$ boostrap set, the data of which are randomly drawn (with replacement) from S
  tree ← an empty tree composed of its root only
  tree.root ← *RndTree (tree.root, S_j, mtry*)
  forest ← forest ∪ tree
End For
return forest

Algorithm 5 RndTree

Entry: n the current node
Entry: S all the data associated with node n
Entry: mtry the number of characteristics to be randomly selected at each node
if it is not a leaf then
  C ← mtry characteristics chosen randomly
  for any A ∈ C do
    CART procedure for the creation and evaluation (Gini criterion) of the partitioning produced by A, according to
  End For
  partition ← partition which optimizes the Gini criterion
  n.addChild (partition)
  for any child ∈ n.node
    RndTree (child, child.data, mtry)
  End For
End if

return

Two methods were calculated to rank the variables in order of importance in learning. To measure the importance of the j-th attribute after training, the values of the j-th attribute are swapped in the training database and the OOB error is computed again on this "disturbed" data set. The importance of the j-th attribute is then calculated by averaging the difference between the OOB error before and after the disturbance on all trees. The score is normalized by the standard deviation of these differences. The more the precision of the forest classification decreases by adding an additional variable, the more important it becomes. The second method measures the influence of a variable in the homogeneity of nodes. As soon as a variable ed to divide a node, the Gini coefficient is calculated and compared to the initial node. The difference of the coefficients is summed for each variable and normalized at the end of the process: the greater the drop, the greater the influence of the variable on the purity of the nodes.
Strobl et al. propose to use another implementation of random forests where the weak classifiers are unbiased classification trees based on a conditional approach (Torsten, 2006). In addition, Strobl et al. recommend the use of a bootstrap without replacement to obtain an unbiased measure of the importance of the variables. One of the disadvantages of random forests is the multiplication of levels disturbing the classification both qualitatively and computationally. Another solution is to group the poorly represented levels together and keep the most frequent ones. We therefore decided to adopt the levels appearing more than 2% of the time within a limit of 30 levels for the qualitative variables. A new level "-1" was then created to group together all the other values.

4.3 Predicting behavior

This section describes the transition of predicting a group from a vector attributes to forecast price changes. A group number E_i can be associated with a flight from the vector of attributes V_i following classification rules. Once this group is assigned to the flight, the associated centroid simulations was applied to predict the future behavior of the price series at time t of the demand prediction.

Learning a behavior
Each test flight has a vector of attributes V_i in the learning base. Each qualitative attribute was created an additional level called "-1". In the process of constructing the V_i of N_{test}, the levels of qualitative attributes were checked in the learning base and in the opposite case the value "-1" was assigned. The selection of the most present values was applied to group the rarest in the same level "-1". The classification models generated previously for each test flight were represented by its attribute vector V_i and an output vector corresponding to the probability of belonging clusters: $P(V_i \in I_k)$, $\forall k = 1, ..., C$. The label E_i corresponding to the number of the most probable cluster was then associated with the test flights.

Predicting an Evolution
A model assigning a cluster number to a new flight through its attributes was generated. This cluster number associates a centroid with the test flight and an overall flight behavior represented by the group's centroid. In the case of centroids, reconstructing flights from any gray level means that synthetic time series can be created from this cluster. Before simulating the price curves, the nature of the prediction should be defined. This prediction was performed at an instant t. The price series

ϕ_t is binary with the research questions: will the price increase in 7 days? will the price drop at least 24 hours in the next 10 days? N curves from the cluster assigned to the test flight were simulated and calculated for each of them $\phi_t \in [0, 1]$. The final prediction of the flight P ($\phi_t = 1$) was obtained by averaging the predictions. However, the evolution of prices according to dials such as "Strong Rise", "Weak Rise", "Stable", "Low Decline" and "Strong Decline" should be predicted. Accordingly, the procedure remains unchanged except that ϕ_t takes 5 values. For the simulations, each dial was incremented when the price change matched its criterion. The results of all the dials were then normalized.

4.4 Direct prediction

Another method of predicting ϕ_t is to apply the learning phase to the function of ϕ_t. In this way, the segmentation step is no longer necessary, as it suffices to calculate ϕ_t for all the flights of the learning base. The label is therefore no longer the group number but directly ϕ_t. With this method, there is no information loss due to grayscale transformation or centroid simulations. Moreover, if the learning is performed on a binary value of ϕ_t, the error chance naturally decreases as the problem complexity decreases. The main drawback of this approach is the obligation to have a model per number of days before the departure date. Unlike the classic approach where only the simulation updates are required, the direct prediction has to recalculate the models at each new definition of ϕ.

4.5 Sequential approach

The yield management techniques were discussed using the demand evolution as the main parameter in the price modification. The information of first price change is subject to research. Popular destinations can benefit from this additional information, as the search volume is large enough for the combinations of dates, routes and length of stay. The first price changes can be added in various ways. In direct prediction, the learning part ϕ_t by random forests can be conducted using the contextual attributes described in Chapter 1. These attributes depend on the number of days before the departure date. They represent the volatility observed in prices and the number of increases or decreases. Using the first points as the sole prediction parameter creates a new type of classifier. The most probable cluster was assigned to each of the test flights from Expectation-Maximization algorithm. Finally, the information on the first points was applied in the behavior learning phase by adding a parameter to the Expectation-Maximization algorithm. The learning is then coupled with the segmentation step and makes it possible to calculate the likelihood of belonging to a cluster thanks to the Vi and the partial gray levels of the test flights.

4.5.1 Classification only by the first price change

Supposing that the curve of flight i is observed up to a date T, we reconstructed the partial gray level of the flight $X_i(s, t)$ for $(s, t) \in \tilde{R}$ where $\tilde{R}$ is the set in the time-yield plane corresponding to the prices observed. We then denoted by $\tilde{X}_i$ the set of $X_i(s, t)$ for $(s, t) \in \tilde{R}$. Given the $\tilde{X}_i$, the I_y centroids and the α_y, the likelihood of belonging cluster for the test flights was calculated as follows:

$$\mathbb{p}_{\alpha,I}\left(Y_i = y \middle| \tilde{X}_i\right) = \frac{p_\alpha I_y(\tilde{X}_i, Y_i = y)}{\sum_{j=1}^{C} p_\alpha I_j(\tilde{X}_i, Y_i = j)} \propto \alpha_y \prod_{(s,t)\in\tilde{R}} \frac{\left(I_y(s,t)\right)^{X_i(s,t)} e^{-I_y(s,t)}}{X_i(s,t)!},$$

for y∈1,. . . ,C.

The classification chooses the y to maximize the likelihood. The calculation of $\tilde{X}_i$ (s, t) is higher than the sampling frequency.

4.5.2 logit EM

The logit EM is a classification method accompanied by a segmentation step similar to the EM with inseparable classification and segmentation. The input data is the set of gray levels X_i and the information of the attributes V_i. The main assumption is the gray levels X_i of flight i depend on attributes through a particular cluster only. In probabilistic terms, the conditional distribution of X_i does not depend on attributes. On the other hand, the law of Y_i depends on the attributes through a function ψ (.,.): $P(Y_i = y) = \psi(y|V_i)$.

Since Y_i is not observed, the law of X_i depends on the attributes in the form:

$$\mathbb{p}(X_i = x) = \sum_{y=1}^{C} \mathbb{p}((X_i = x|Y_i = y)\psi(y|V_i).$$

As in section 3.3.3, the probability $P(X_i = x|Y_i = y)$ depending on the parameter $I_y \in (R^{*}_{+})^R$ is given by Poisson's law. Therefore, a Poisson model was adopted to calculate the probabilities of belonging clusters using the parameters (α_y) which were replaced by the functions (ψ (y|v)) in logit parameterization.

Algorithm description

The algorithm is similar to the Expectation-Maximization algorithm in Section 3.3.3, except that it introduces the information of the V_i attributes into the segmentation step.

Input data

The input data are those of the EM described in section 3.3.3 and the attributes V_i of all the flights i∈1, …, n.

Initialization

The initialization is almost identical to the EM of section 3.3.3, however a third optimized parameter is added, given by the function ψ defined above. A first segmentation was therefore applied by the K-Means, initializing the centroids I_y and the pairs (X_i, $\hat{Y}_i$). The pairs (V_i, $\hat{Y}_i$) was used to optimize the function ψ.

Likelihood function

The log density of the observed variables X_i, i = 1, . . ., n, given the observed attributes V_i, i = 1, . . ., n, is written:

$$\log p_\theta(x_1, \ldots, x_n|v_1, \ldots, v_n) = \sum_{i=1}^{n} \log p_\theta(x_i|V_i)$$

$$= \sum_{i=1}^{n} \log \sum_{y_i=1}^{C} p_\theta(x_i|V_i, y_i) p_\theta(y_i|V_i)$$

$$= \sum_{i=1}^{n} \log \sum_{y_i=1}^{C} p_\theta(x_i|y_i) p_\theta(y_i|V_i),$$

where, for θ = (I, ψ),

$$p_\theta(y_i|V_i) = \psi(y|V_i)$$

(4.1)

and

$p_\theta(x_i|y_i) = \prod_{r\in R} \frac{e^{x_i(r)(\log I(r|y_i)) - I(r|y_i))}}{x_i(r)!}$.

By factoring all the terms in the log-likelihood function, we obtain:

$$\log p_\theta(x_1, \dots, x_n|v_1, \dots, v_n) = (\dots) + \sum_{i=1}^{n} \log \sum_{y_i=1}^{C} \exp(\ell(y_i|V_i, I))\, \psi(y|V_i),$$

(4.2)

where the constant (...) does not depend on θ.

$$\ell(x|y, I) = \sum_{r\in R} x_i(r)(\log(I(r|y_i)) - I(r|y_i)).$$

(4.3)

Expectation Step

Knowing the attributes $v_1, \dots, v_n$ and the two parameters θ=(I,ψ) and θ′ = (I′,ψ′), the conditional expectation of the log-density joined to the parameter θ′ was calculated.

$$Q(\theta', x_i, \dots, x_n|\theta, v_1, \dots, v_n) = \sum_{i=1}^{n} \sum_{y_i=1}^{C} [\log p_{\theta'}(x_i, y_i|V_i)]\, p_\theta(y_i|x_i, V_i).$$

For convenience, we omitted $x_1, \dots, x_n$ and $v_1, \dots, v_n$ in the notation of Q and simply wrote $Q_n(\theta'|\theta)$ We then obtain:

$$Q_n(\theta'|\theta) = (\dots) + \sum_{y=1}^{C} (\sum_{r\epsilon R} B_n(y, r|\theta) \log I'(r|y) - A_n(y|\theta) \sum_{r\epsilon R} I,(r|y))$$

$$+ \sum_{y=1}^{C} \sum_{i=1}^{n} (\log \psi'(y|V_i)) p_\theta(y|x_i, V_i),$$

where (...) is a constant independent of θ', and we have:

$$p_\theta(y|x, v) = \frac{p_\theta(x, y|v)}{\sum_{y'} p_\theta(x, y'|v)}, \text{and } p_\theta(x, y|v) = \psi(y|v) \exp \ell(x|y, I)$$

with ℓ defined in (4.3) and

$$A_n(y|\theta) = \sum_{i=1}^{n} p_\theta\,(y|x_i, V_i)\ and\ \ B_n(y, r|\theta) = \sum_{i=1}^{n} x_i(r) p_\theta(y|x_i, V_i).$$

(4.4)

Maximization Step

This step maximizes Q (θ′|θ) in θ′ for a given θ.

As for the EM, the optimization at I' with the weight ψ can be treated separately. We obtain:

$$I'(r|y) = \frac{B_n(y, r|\theta)}{A_n(y|\theta)}, \quad r\epsilon R, y = 1, \dots, C.$$

(4.5)

The optimization in ψ depends on the function used in the set of functions. We obtain:

$$\psi' = arg \max_{\phi \neq \Psi} \sum_{i=1}^{n} \sum_{y=1}^{C} (\log \phi(y|V_i))\, p_\theta(y|x_i, V_i).$$

(4.6)

The class ψ represents a set of probable functions for ψ. In the particular case where ψ is constant in V, we obtain

$$\psi'(y) = \frac{A_n(y|\theta)}{\sum_{y'}^{C} 1^{A_n(y'|\theta)}}.$$

To involve the attributes, the function below in logit type is chosen:

$$\psi(y, v) = \log it_\varphi(y|V) = \frac{e^{\varphi_y^T V}}{\sum_{y=1}^{C} e^{\varphi_y^T V}}$$

(4.7)

The function ψ is fully described by the parameter $\phi \in R^p$. The ψ′ is calculated therefore to generate the new parameter ϕ′.

We then obtain:

$$\varphi' = arg \max_{\varphi}(\sum_{i=1}^{n} \sum_{y=1}^{C} (\varphi_y^T V_i) p_\theta(y|x_i, V_i) - \sum_{i=1}^{n} \sum_{y=1}^{C} \log(\sum_{j=1}^{C} e^{\varphi_y^T V}) p_\theta(y|x_i, V_i))$$

$$= arg \max_{\varphi}(\sum_{i=1}^{n} \sum_{y=1}^{C} (\varphi_y^T V_i) p_\theta(y|x_i, V_i) - \sum_{i=1}^{n} \log \sum_{y=1}^{C} e^{\varphi_y^T V_i})$$

By setting $\phi_C = 0$,

$$\varphi' = arg \max_{\varphi}(\sum_{i=1}^{n} \sum_{y=1}^{C-1} \varphi_y^T V_i p_\theta(y|x_i, V_i) - \sum_{i=1}^{n} \log(1 + \sum_{y=1}^{C-1} e^{\varphi_y^T V_i})$$

After differentiating, ϕ' is the solution of the equation:

$$0 = \sum_{i=1}^{n} V_i p_\theta(y|x_i, V_i) - \sum_{i=1}^{n} \frac{V_i e^{\varphi_y^T V_i}}{(1 + \sum_{j=1}^{C-1} e^{\varphi_y^T V_i})} \quad , \varphi_1, \dots, \varphi_{C-1} \in \mathbb{R}$$

Output data

When the algorithm is stopped, an estimator $\hat{\theta}= (\hat{I},\ \hat{\psi})$ or $\hat{\theta}= (\hat{I},\ \hat{\varphi})$ in the logit parameterization will be generated. The logit EM calculates the most probable group for each flight i using the likelihood function $p_\theta(x, y|v)$. It applies to $X_i(s,t)$ to determine the label $\hat{Y}_i$ for each flight i.

$$\hat{Y}_i = arg \max_y \mathbb{p}_\theta(Y_i = y|V_i) = arg \max_y \psi(y|V_i)$$

We then define $X_i = (\tilde{X}_i,\ \hat{X}_i)$ with $\tilde{X}_i$ the observed points, and $\hat{X}_i$ the future evolutions. We would therefore have:

$$\mathbb{p}_\theta(\hat{\varphi}|V_i, \tilde{X}_i) = \sum_{y=1}^{C} \mathbb{p}_\theta(\hat{\varphi}|V_i, \tilde{X}_i, Y_i = y)\mathbb{p}_\theta(Y_i = y|V_i, \tilde{X}_i)$$

$$= \sum_{y=1}^{C} \mathbb{p}_\theta(\hat{\varphi}|Y_i = y)\mathbb{p}_\theta(Y_i = y|V_i, \tilde{X}_i)$$

for any event $\hat{\varphi}$ dependent on $\hat{X}_i$, with

$$\mathbb{p}_\theta(Y_i = y|V_i, \tilde{X}_i) = \frac{p_\theta(\tilde{X}_i|y)\psi(y|V_i)}{\sum_{j=1}^{C} \mathbb{p}_\theta(\tilde{X}_i|j)\psi(j|V_i)} \propto p_\theta(\tilde{X}|y)\psi(y|V_i).$$

In this case, EM segmentation evaluation metrics can be used along with prediction performance. To limit computation time and resource, the number of attributes should be reduced. Therefore, variable selection techniques described above were adopted and 10 most discriminating variables were chosen.

4.6 Conclusion

Different steps were discussed to provide price prediction for all the results of a user search. The first step associates each result with its most probable class. After segmenting the flight behaviors from the gray level basis, the supervised learning algorithm was applied through the attributes. The flight characteristics V_i and the associated label E_i were assigned to the corresponding class assigned during the segmentation step. The supervised learning algorithm provides classification rules to classify the test flights into their most likely group based solely on their attributes. Therefore, algorithms such as CART and C4.5, decision trees were used to extract rules. Adaboost, Random Forests and Bagging were also conducted on training multiple decision trees with subsets of data drawn randomly. An extension of the EM algorithm was introduced to perform the segmentation and classification step simultaneously.

Therefore, a label E_i the centroid of the cluster associated was obtained for each test flight, followed by the calculation of the probabilities P (ϕ (i) = 1), $\forall t \in$ [Tinit, T_{0-7}] for a 7-day prediction. The

direct prediction were introduced to directly model the series without the segmentation step. This method involves a model per days and imposes the renewal of learning at each change of ϕ_t. Finally, the different possible approaches were defined in the refinement of the prediction. The context attributes described in Chapter 1 were used as attributes in direct prediction. The supervised learning required information on demand at time t, the volatility of the first price changes, and the present value of the price. To create a partial gray level and calculate the partial likelihood, all these methods required one model per day before the departure date.

Chapter 5

Introduction

This chapter studies the empirical analysis of the different approaches' performance. First, different algorithms with their influence on the prediction were compared. The results in terms of behavioral predictions were presented first as the prediction of an increase or a decrease binary ($\phi_t[0, 1]$) at 7 days. The notions of True Positives, True Negatives, False Positives and False Negatives were used. These four criteria were compared using the confusion matrices. The number of false positives was minimized to avoid financial loss. The rate of true positives was visualized as a function of the rate of false positives when the decision threshold of ϕ_t varies. Then the ROC curve was used to compare the performances of the different approaches at time t. The ROC curve also determined the optimal threshold at each instant t. Finally, at a fixed threshold, the satisfactory prediction rate was visualized according to the number of days before the departure date. The optimal threshold was determined by performing a prediction on the learning basis.

At the same time, the financial gains for the user and the percentage of satisfactory predictions were visualized. The methods to calculate the gains and losses was discussed. At each step (modeling, segmentation, classification), different parameters were compared to choose the optimal configuration. At the clustering stage, the number of clusters, the number of initializations for all the previously mentioned approaches such as K-Means, Bagged K-Means and EM were compared. At the learning stage, the different algorithms (CART, Adaboost, Decision Tree and EM logit) were compared. The influence of the attributes used for classification was discussed. We then checked the feature selection step to optimize the results by eliminating correlated or unnecessary attributes.

It is important to know if it is necessary to use the entire database or 10 percent is enough to predict correctly. The evolution of the satisfactory prediction rate was discussed to identify the renewal frequency of the models. The effects of adding the first jumps information in the prediction were compared to the direct prediction. Finally, the results of our approach were described on the extended basis of 90-day flights.

5.1 Measures of Performance

5.1.1 Confusion matrix

The confusion matrix is a metric that measures the quality of a classification system. Usually used in binary prediction, it can be extended to any number of classes. In the case of a binary prediction, 4 values are defined: True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). In our case, $\phi = 1$ corresponds to the advice "wait" and $\phi = 0$ refers to the suggestion "buy immediately". The number of true positives is then the number of times the predictor detected a price drop correctly. The false positive will be a false predictive value when in reality the price increased. One of the advantages of the confusion matrix is that it shows quickly whether the system classify correctly. In our case, it is important to minimize the number of false negatives as it implies

greater frustration.

5.1.2 Evolution over time

With confusion matrices, the overall percentage of satisfactory predictions evolving over time was visualized. We then defined the overall rate of satisfactory predictions:

$$f_{global}(t) = \frac{TP + TN}{TP + TN + FP + FN}$$

The specificity, or the probability of detecting an increase correctly:

$$f_{spec}(t) = \frac{TN}{TN + FP}$$

The sensitivity, the proportion of price drops detected:

$$f_{sens}(t) = \frac{TP}{TP + FN}$$

The evolution of these criteria according to the number of days before the departure date was visualized. The challenge is to maintain a satisfactory rate of global predictions and balance the detection of increases and decreases correctly. Therefore, the notion of decision threshold was applied. These curves depend on the thresholds [0, 1] where P ($\phi = 1$) is considered as a drop. After a cluster prediction, a flight was associated with a probability of evolution at an instant t. If s = 0.5 with $P(\phi = 1) \geq 0.5$, a decrease will be predicted, and vice versa. For all values of $s < 0.5$, we favored the negative predictive values. Conversely, for all $s > 0.5$, the predictions will be positive. Each curve therefore corresponds to the evolution of satisfactory predictions as a function of the date before departure.

5.1.3 ROC curve

The ROC (Receiver Operating Characteristic) curve measures the performance evolution of a binary classifier as a function of a discrimination threshold. A high threshold corresponds to conservative behavior. Conversely, when the threshold is low, the negative prediction is favored. The ROC curve can be formulated as follows:

$$ROC = \frac{\mathbb{P}(x|Positive)}{\mathbb{P}(x|Negative)}$$

Different selection criteria can be applied depending on the problem. For example, area under the curve (AUC) is the best compromise between sensitivity and specificity. In our case, the number of FPs was minimized.

The Youden Index

The Youden Index assesses the ability of the classifier to avoid prediction error. The index gives equal weight to false positive and false negative values, therefore all tests with the same value of the index provide the same proportion of total misclassified results. It should be noted that in our case, we performed the N simulations only once per cluster to minimize the computation time. As a result, two test flights assigned to the same cluster will have the same P ($\phi_t = 1$). A confusion matrix of gains and losses was created. For each of the four classic confusion matrix, we defined a gain or loss as follows: d = firstPrice-lastPrice, with firstPrice indicating the price of the ticket at the time

of prediction and lastPrice referring to the actual price at the end of the prediction. A new confusion matrix was defined:

TABLE 5.1 Confusion matrix of gains and losses

		Reality	
		0	1
Predict	0	G_{TN}=-d	P_{FN}=d
	1	G_{FN}=-d	G_{TP}=d

We then found 3 curves: the average gain or loss per flight $g_{global}= (\Sigma_n G_{TP} - G_{TN} + P_{FP} – P_{FN})/n$, the average gain per flight $g_{gain} = (\Sigma G_{TP} - G_{TN})/n$ and the average loss per flight $g_{loss} = (\Sigma -P_{FP} + P_{FN})/n$. Two comparative curves were introduced which are the gain made by purchasing on the day of the prediction systematically, i.e. $\forall$ t, $P(\varphi= 1) = 0$, $g_0=(\Sigma_n - G_{TP} - G_{TN} - P_{FP} - P_{FN})/n$. An optimum curve describes the gain for someone who knows the future, and makes the right choice systematically, $g_{opt} =(\Sigma_n G_{TP} - G_{TN} + P_{FP} - P_{FN})/n$. The aim of the research is to find a balance between optimizing the rate of satisfactory predictions and increasing the average gain per ticket.

5.2 Results

This section studies the influence of parameters and algorithms on the prediction quality. At each step, the specific metrics and the performance measurement were described. Started by studying the parameters of the data segmentation such as number of groups and initializations, the classification step was analyzed. Then, the influence of the input parameters (base size, pixel size, ...) was identified. We focused on the base of flights sampled every 6 hours over a period of 28 days representing 444,503 flights from June 2017 to January 2019. The test base includes 19,848 flights between January 1, 2019 and February 28, 2019. Finally, the base extended to 90 days was studied. The algorithms were applied on a completely independent basis.

5.2.1 Segmentation

K-Means

In previous chapter, the different parameters of the K-Means algorithm were described. It is important to differentiate the parameters on the segmentation quality and the final prediction. Therefore, two performance criteria for each result are displayed with gains and losses depending on the number of days before the departure date. In fact, not only the rate of satisfactory predictions but also the overall gain was concerned. The criterion of the optimized K-Means is as follows:

$$w_y=\sum_{X_i \in y} \left\| X_i - I_y \right\|^2$$

$$W=\sum_{i=y}^{C} \sum_{X_i \in y} \left\| X_i - I_y \right\|^2$$

It can be applied to choose the K-Means parameters.

Evolution of performance as a function of the number of starts

The number of random starts is essential for the smooth running of the algorithm. The variance of

W over several iterations as a function of the number of initial starts was observed. The intra-centroid variance is represented by the horizontal lines. The probability of converging towards the global optimal therefore increases with the number of random starts. However, the computation time and the resources used by the K-Means are an obstacle to the choice of multiple departures. We noted that the computation time is almost linearly correlated to the number of departures, implying a compromise in the choice of the value of *nstart*.

Regarding the final prediction, Figure 5.1 demonstrates the influence of the *nstart* parameter on the evolution of the percentage of satisfactory predictions. It is a function of the date before departure and the ROC curves associated with t=-21. A minimal difference can be found between 10 starts and 100 starts. No difference is observed between 100 and 1000. The number of starts was decided to be 100 for the K-Means.

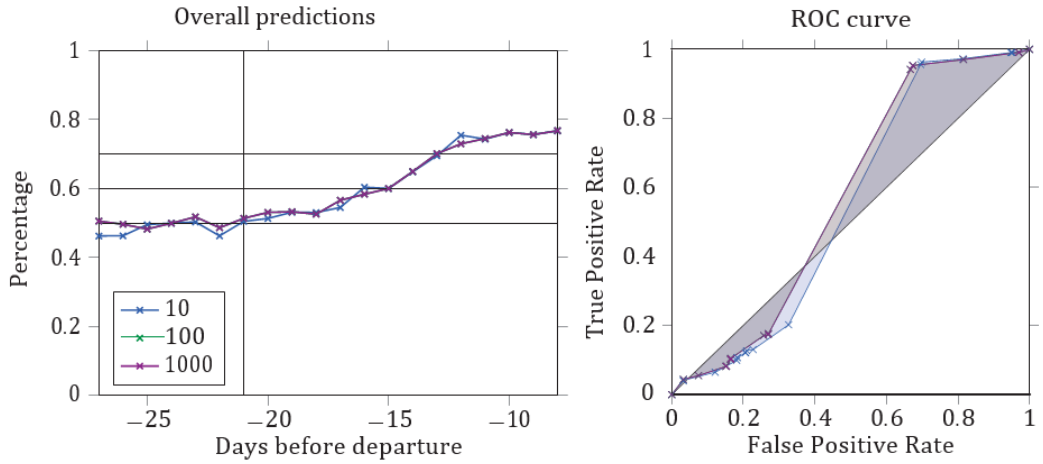

FIGURE 5.1 - Evolution of the percentage of satisfactory predictions as a function of the date before departure and ROC curve at -21 days for different *nstart*

Evolution of performance as a function of the number of groups

The sum of the intra-centroid distances and the Calinski-Harabasz index were observed. These two segmentation quality indicators are strictly decreasing. Regarding predictions, the K-Means maintain a stable rate of satisfactory predictions from 10 groups with similar ROC curves at -21 days. The differences observed on the evolution of the satisfactory predictions rate and on the gains and losses are caused by the choice of the threshold.

The GAP algorithm is time consuming, therefore the last 5 percent of the learning base was selected for groups ranging from 1 to 50. Different metrics were applied to choose the best number of clusters:

1. The maximum value of GAP_k at 50 groups
2. The first occurrence of a local maximum at 27 groups
3. The smallest k for which $f(k) \geq f(k+1) - s_{k+1}$ at 27 groups

The K-Means applied to the last 10 percent of the learning base. The cluster of 27 groups provided satisfactory results for the ROC curve. This choice was applied to the “EM” and “Bagged K-Means” algorithms in Figure 5.2. The first three dials respectively represent the overall prediction rate, true

negative predictions and true positive values according to the number of days before the departure date. The gain for the customer were compared for each algorithm. The K-Means algorithm and the Bagged K-Means have similar performances with an almost equal overall prediction rate (59% and 58% respectively). The EM performance reached 60% of the overall satisfactory predictions despite a much smaller area under the ROC curve at -21 days. Additionally, GAP method provided the magnitude order of the group numbers minimizing W.

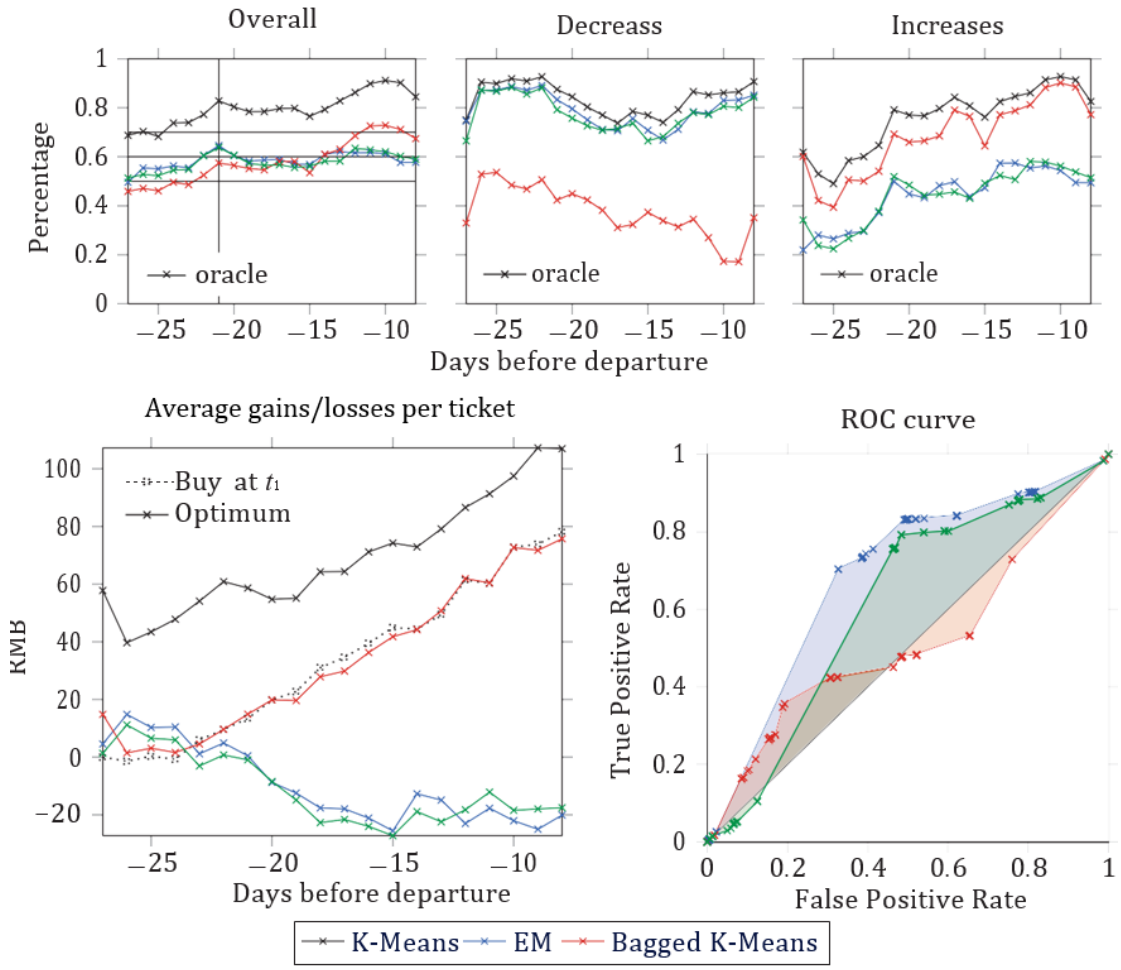

FIGURE 5.2 K-Means, EM and Bagged K-Means with 27 groups over the last 10 percent of the base.

The Bagged K-Means

The Bagged K-Means algorithm attempts to reduce the influence of the initialization of the K-Means and to promote convergence towards a global optimum by applying the K-Means algorithm to subsets drawn at random and by aggregating the results of each of these segmentations. It is possible to modify the number of initializations of the K-Means. By assuming that the Bagged K-Means suppress the influence of the parameter, the number of initializations does not have an influence on performance. The algorithm provides satisfactory results and tends to better predict declines. We therefore did not retain this algorithm and chose the K-Means for performance and calculation speed.

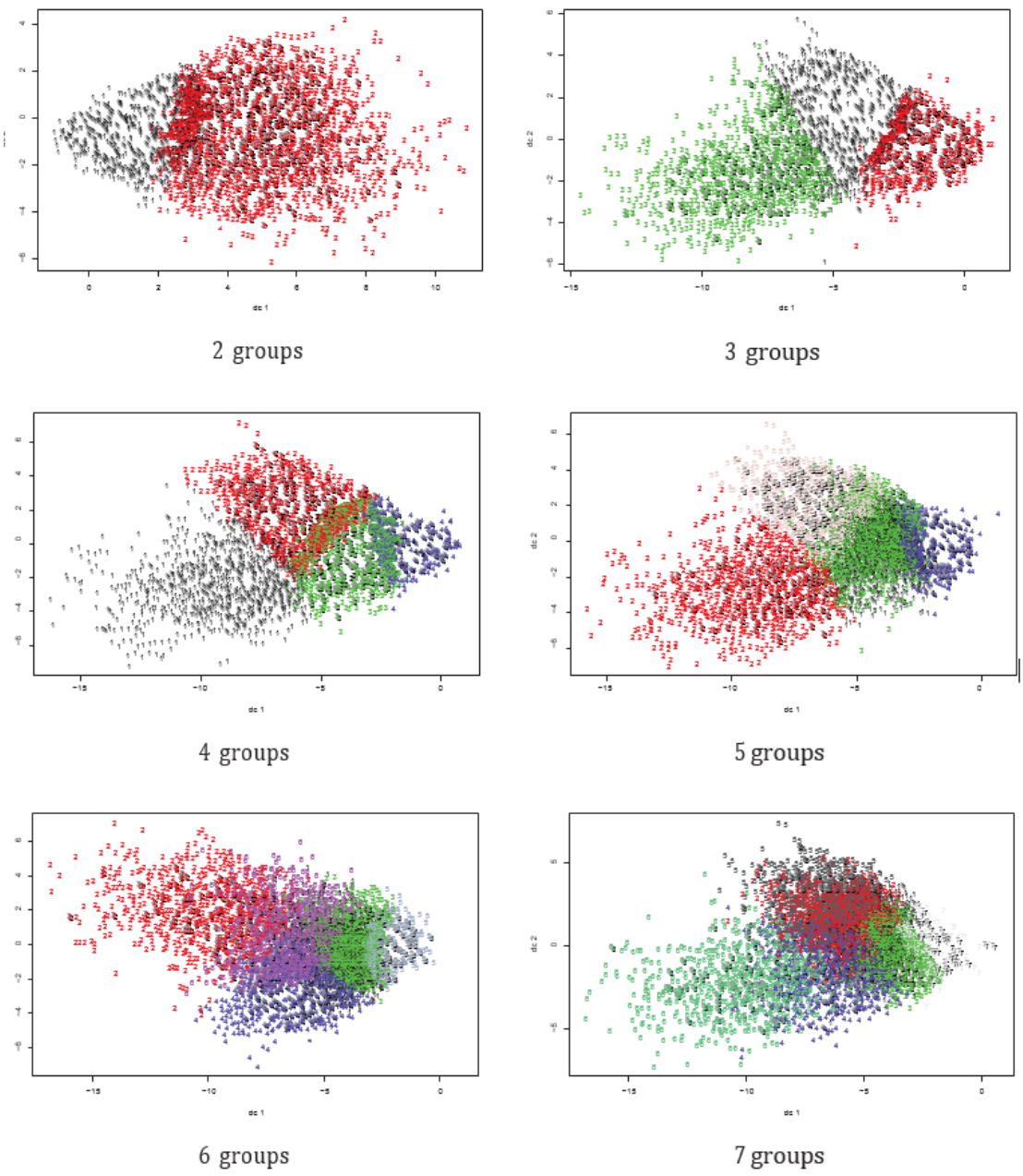

FIGURE 5.3 Function of the groups number for K-Means

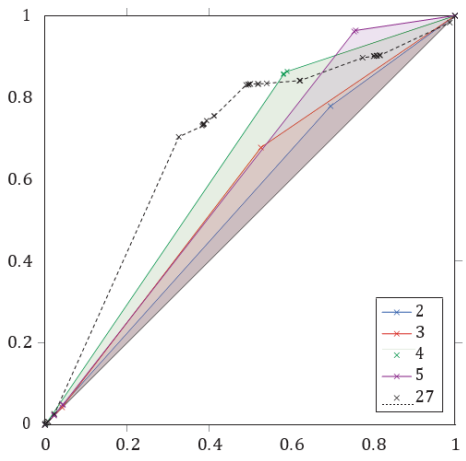

FIGURE 5.4 - ROC curve for a K-Means with 2, 3, 4 and 5 clusters compared to K-Means with 27 groups

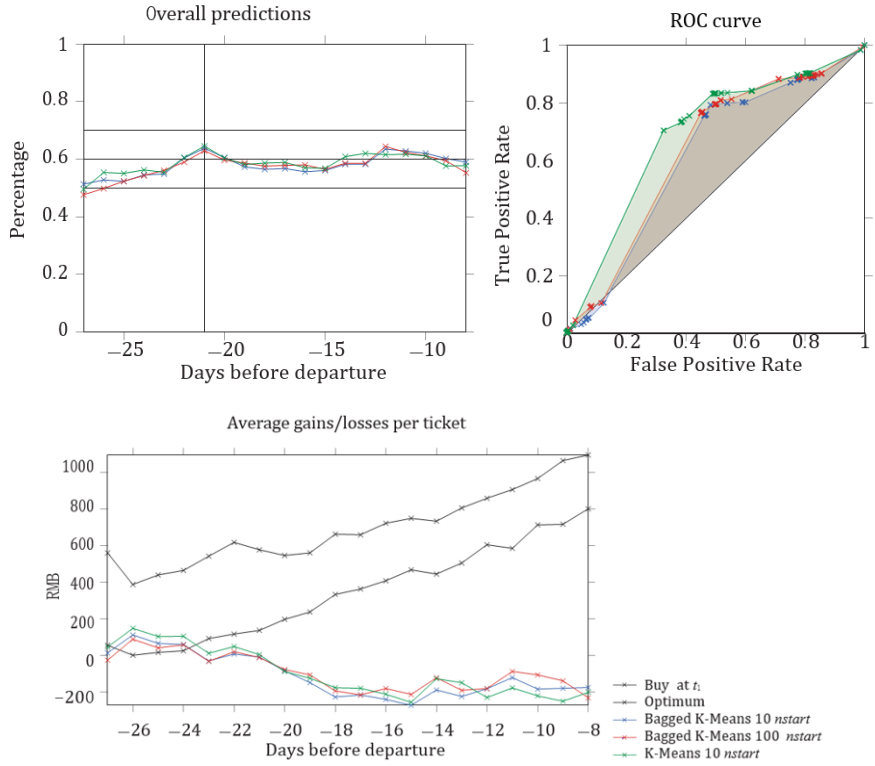

FIGURE 5.5 - Comparison of Bagged K-Means with a K-Means

Expectation-Maximization

As for the Expectation-Maximization algorithm, the issues are the same as for the K-Means: local convergence, number of starts, number of clusters, etc. Therefore, the same metrics are used to evaluate the configuration. However, the criterion to be optimized is not the same for the two algorithms (W for the K-Means and the global likelihood for the EM), we mainly compared their prediction quality. As for the K-Means, the evolution of the criterion to be optimized improves with the increase in the number of groups, and the evolution of W is strictly decreasing. Additionally, unlike the K-Means, the Expectation-Maximization algorithm is sensitive to variations in its parameters. When the right number of groups is chosen, the EM performances in satisfactory predictions are better against the K-Means. However, the results are radically reduced if different clusters are tested. It is therefore necessary to test different choices and select the number of groups which gives the best results. In our case, 5 and 30 groups seem to stand out.

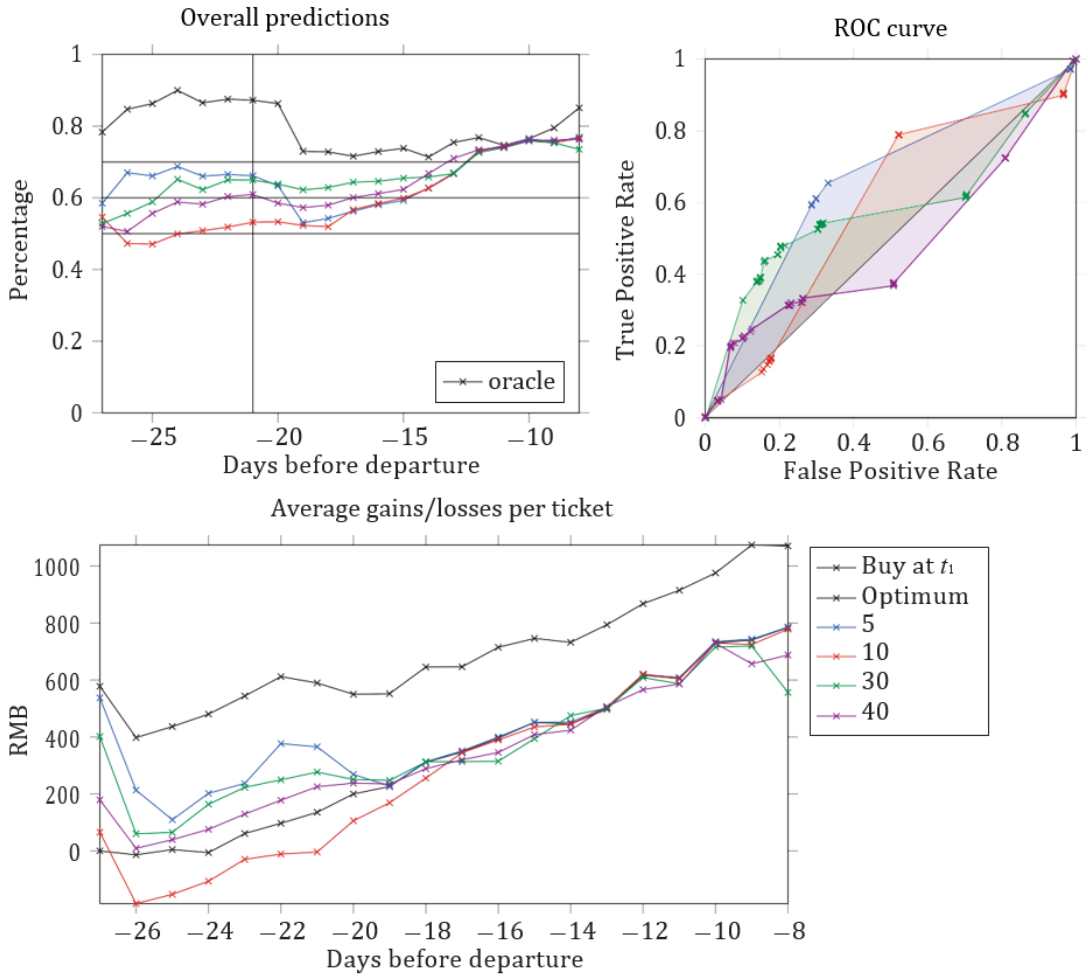

FIGURE 5.6 Changes in the performance of the EM algorithm as a function of the number of clusters

Different ways to initialize the Expectation-Maximization algorithm are described in previous chapter. Figure 5.7 shows that there are no major differences between the two methods including random initialization of the parameters and segmentation by the K-Means. We adopted the second method due to the slight advantage of initialization by K-Means.

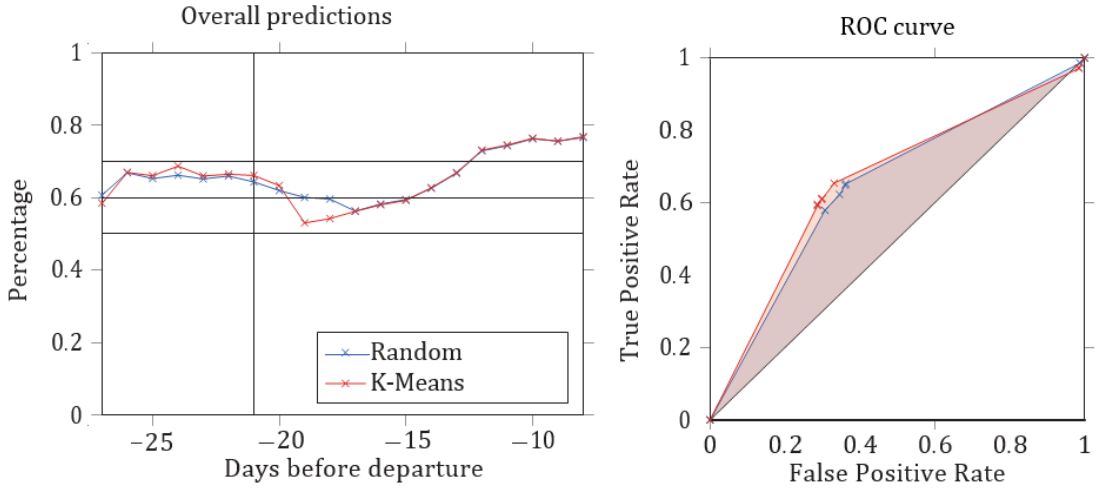

FIGURE 5.7 Comparison of the performance of the EM algorithm according to its type of initialization

Comparison

Among the three segmentation algorithms, the EM has the best rate of satisfactory predictions. Table 5.3 provides a summary of the final performances of the segmentation configurations. We chose the most stable one so as not to bias the results. Systematic purchasing at the time of prediction t_1 saves on average 390RMB per ticket while buying at the best time saves on average 560RMB per ticket. The maximum savings compared to immediate purchases are therefore 700RMB. We have represented in the fourth column of table 5.2 the difference in gain on immediate purchase.

TABLE 5.2 Comparison

Method	Groups	BP Rate	Economy/t_1
K-Means	5	62%	-100
EM	5	66%	400
K-Means	10	64%	100
EM	10	62%	-300
K-Means	30	62%	-700
EM	30	64%	200
K-Means	40	66%	-180
EM	40	64%	100

The first observation is the similar rates of satisfactory predictions, but the gains for the user vary from a slight increase to a large decrease. We therefore retained two configurations which are the EM with 5 clusters and with 30 clusters. Supervised learning algorithms were applied in next section to the two best segmentation configurations.

5.2.2 Classification

CART

In Figure 5.8, we found that the deeper the tree, the better the results. In fact, when the tree is pruned, most of the groups are no longer represented, thus limiting the distribution of test flights over two or three clusters only. In our case, it is better to represent as much behavior as possible, therefore the whole tree created by the CART algorithm was kept at the end of the construction step. The performances are rather weak, as the rate of satisfactory predictions rarely exceeds 60%. We then used another decision tree algorithm: C5.0.

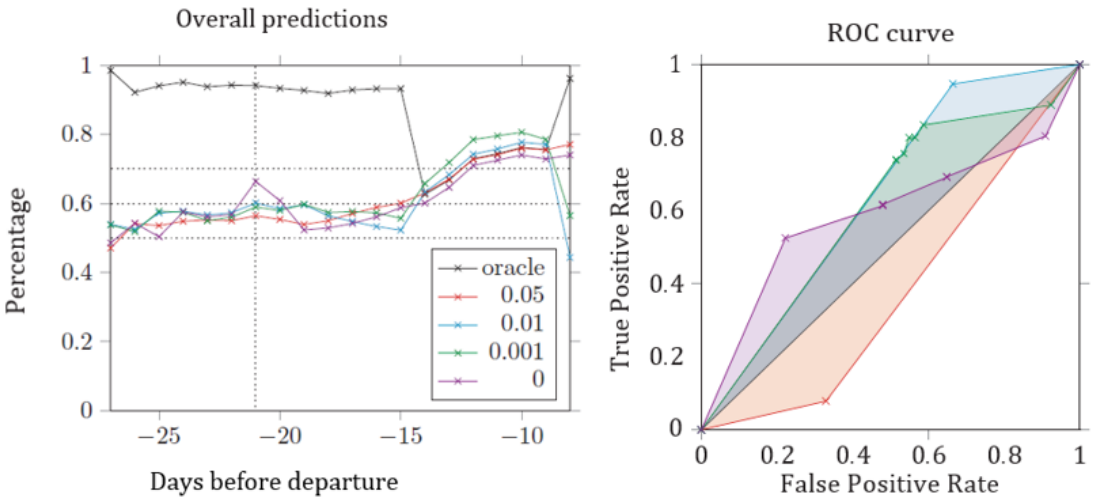

FIGURE 5.8 Evolution of the percentage of satisfactory predictions as a function of the date before departure and ROC curve at -21 days

C 5.0

The C5.0 algorithm has a parameter regulating the pruning of the tree created. In Figure 5.9, different values of the confidence factor between 0 and 1 are studied. We then observed that the default value of 0.25 provides the best results.

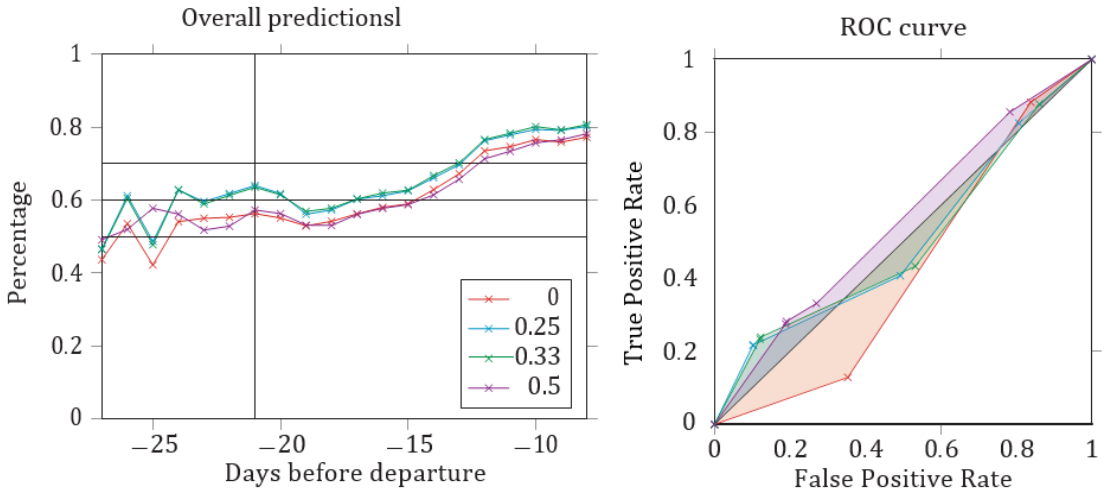

FIGURE 5.9 Evolution of the percentage of satisfactory predictions as a function of the date before departure and ROC curve at -21 days

However, the results are not satisfactory, as the ROC curves are close to the diagonal. Therefore, more complex algorithms such as Adaboost and random forests were studied.

Random Forests

As explained in the previous chapter, random forests are an aggregation of multiple CART decision trees applied to random subsets of the learning base. The two main parameters are the number of attributes chosen randomly at each node and the number of trees constructed. We therefore studied the influence of these two parameters on the quality of the classification and the prediction.

Number of attributes selected randomly at each node: mtry

The parameter K determined the number of features selected randomly at each node during the procedure of inducing a tree. Its value is therefore chosen in the interval [1 . . . M], where M represents the dimension of the description space. The number controls the amount of randomness introduced into the feature selection process. The smaller the value of K, the more randomness will be introduced. In the case where K = 1 for example, the characteristic of each tree is chosen entirely randomly from the available characteristics. Conversely, when K = M, the randomness does not intervene in the selection of the partitioning rule. Concerning the value of K, the performances were not sensitive since the average deviations of the error rates obtained are not exceed 1%. We showed with experiments that the influence of this parameter on forest performance is not negligible. Learning several forests for K values ranging from 1 to 20, the error evolution with the number K is calculated in Figure 5.18. The value of mtry is decided to be 12 in our case.

Number of trees: ntree

Breiman states that the generalization error rate of a random forest necessarily converges towards a limit value, when the number of trees that compose it increases. One interpretation of this result is that when building a random forest, it is not necessary to add decision trees to get better performance. Beyond a certain number of random trees, no significant performance gain will be achieved by adding more. To stable the classification, the assignment of a group to learning flights is required to be the same. The random share of random forests therefore forces us to increase the number of trees and we observe that the percentage of difference between two classification by identical random forests evolves in the same way as the OOB error. It is therefore necessary to know what is the limit number of trees beyond which it is no longer useful to add more. A limit comes naturally during experiments due to a problem of memory and computing time. The memory space taken up by the creation of the model, although linearly increasing.

In Figure 5.10, the evolutionary performance of random forests is visualized as a function of the number of trees built. It can be noticed that the variations become minimal starting from 120 trees. Therefore, a set of 100 trees was chosen to maintain a satisfactory prediction rate.

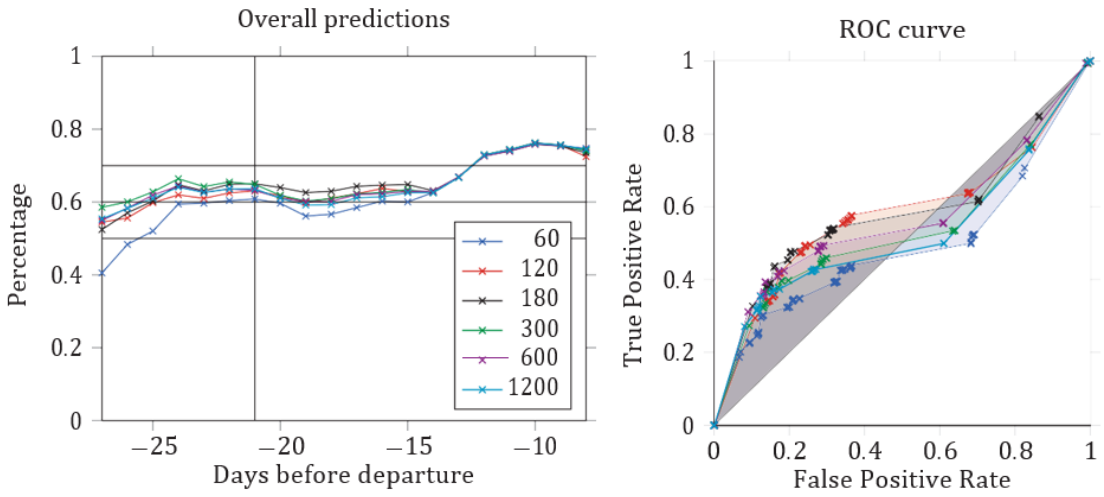

FIGURE 5.10 The percentage of satisfactory predictions as a function of the date before departure

and ROC curve at -21 days for different ntree

Adaboost

The algorithm of random forests is often compared to Adaboost. Boosting is based on a deterministic principle for the creation of diversity while random forests is performed by the principle of randomization., These two algorithms show similar results in some articles, but in our case, the Adaboost seems to be less efficient. In Figure 5.11, we competed random forests against Adaboost for the same segmentation by EM with 5 clusters. As the number of attributes for classification was reduced by selecting the best ten variables, the results of the two algorithms were improved.

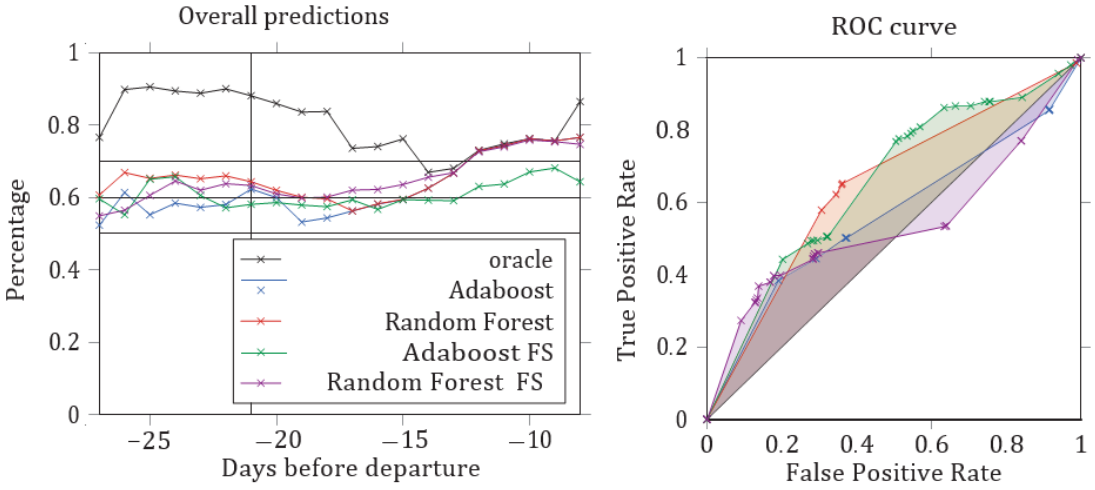

FIGURE 5.11 - Adaboost and RandomForest comparison

Based on the CART algorithm, the optimal configuration in the pruning step is chosen. The implementation of the algorithm requires an enormous amount of execution time. Therefore, random forests rather than Adaboost was selected for its prediction efficiency.

Logit EM

Logit EM is the simultaneous application of segmentation and classification. We first chose the most relevant variables according to a classification by random forests. Then the most frequent values were selected to transfer the qualitative attributes with many levels into binary attributes. The others were finally grouped into a single attribute. The attributes retained are the city of arrival (arrivalStation), the commercial site (provider), the type of flight, ticket sold by the airline company (directSeller), the season, the month of the departure (month) and the month of the return (monthRet).

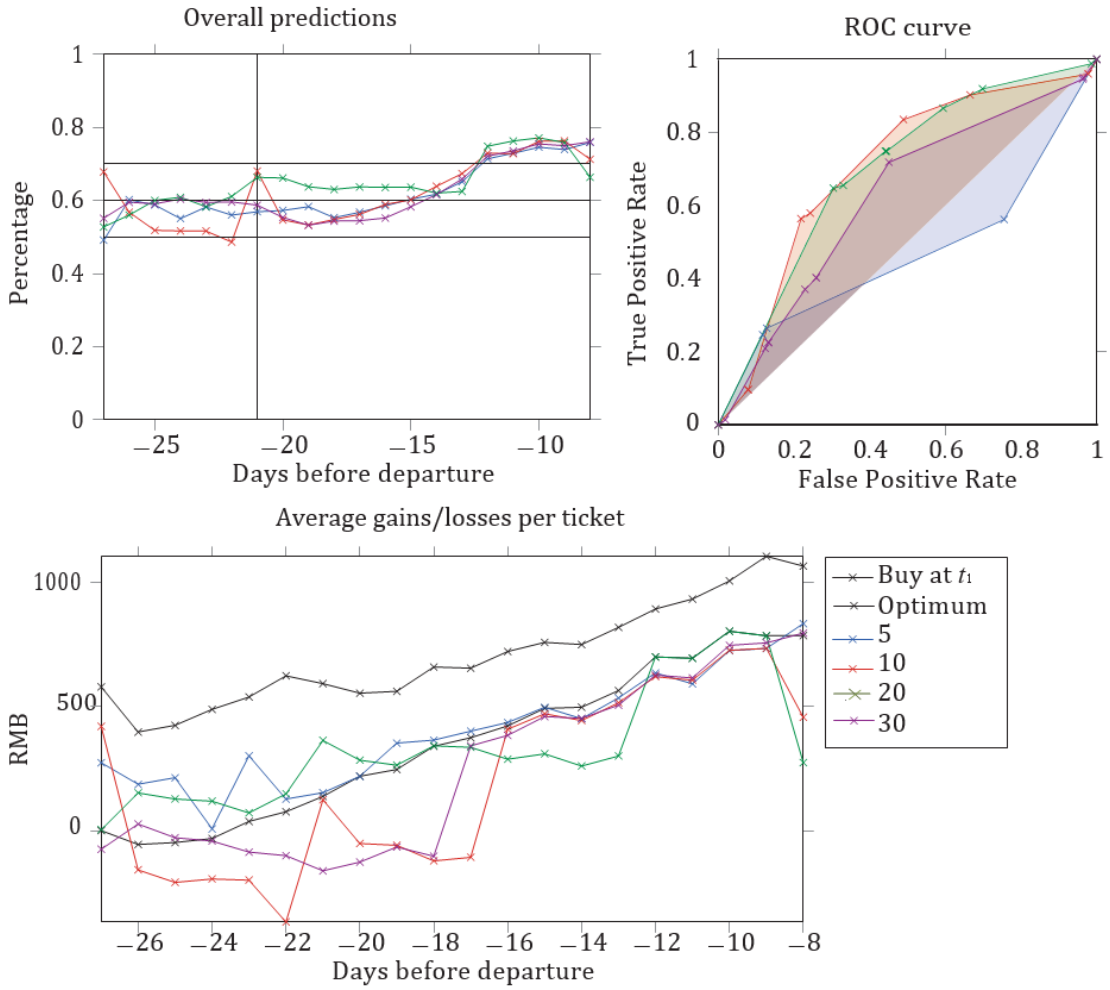

FIGURE 5.12 Logit EM performance according to the number of groups

In Figure 5.12, the performance of the logit EM is compared according to different number of clusters. The ROC curve reflects prediction performance at -21 days. As for the rate of satisfactory predictions, the performance of the segmentation into 10 groups is punctual, however, the results of the segmentation into 5 and 20 groups are promising. These two groups were used for future comparisons.

Comparison

Table 5.3 summarizes the results previously discussed. All the experiments were carried out with the same EM segmentation step except the EMlogit. We observed a similarity in the rates of satisfactory predictions but not in the final gain compared to the systematic immediate purchase. As stated in the table, the performance rate of EMlogit, the Random Forest and the C5.0 algorithm arrive at 66%. However, only the random forests method balances the rate of satisfactory predictions and the economic gain by saving 240RMB per ticket on average. Then different algorithms in selection were conducted to improve the performance rate.

TABLE 5.3 - Comparison

Method	Clustering NO.	Rate	Economic gain /t_1
Logit EM	5	63%	400

Logit EM	20	66%	-400
RF	5	66%	240
RF	30	64%	100
Adaboost		63%	0
CART	5	62%	0
CART	30	62%	-300
C5.0	5	66%	-200
C5.0	30	61%	-700

After selection of variables

After an initial classification by random forests, the attributes should be classified by their order of importance. Applying the first 10 attributes only, the performance of the classification algorithms were improved with reduced computation time and resources. We found six attributes including the price at -28 days (initialPrice), the day of the year (day_of_year), the arrival city (arrivalCity), the airline companies (directSeller), the merchant site (provider), and the return month (monthRet) in the two classifications. The results were almost the same with a slight advantage in the rForest version, suggesting that the 6 variables previously mentioned were the most relevant.

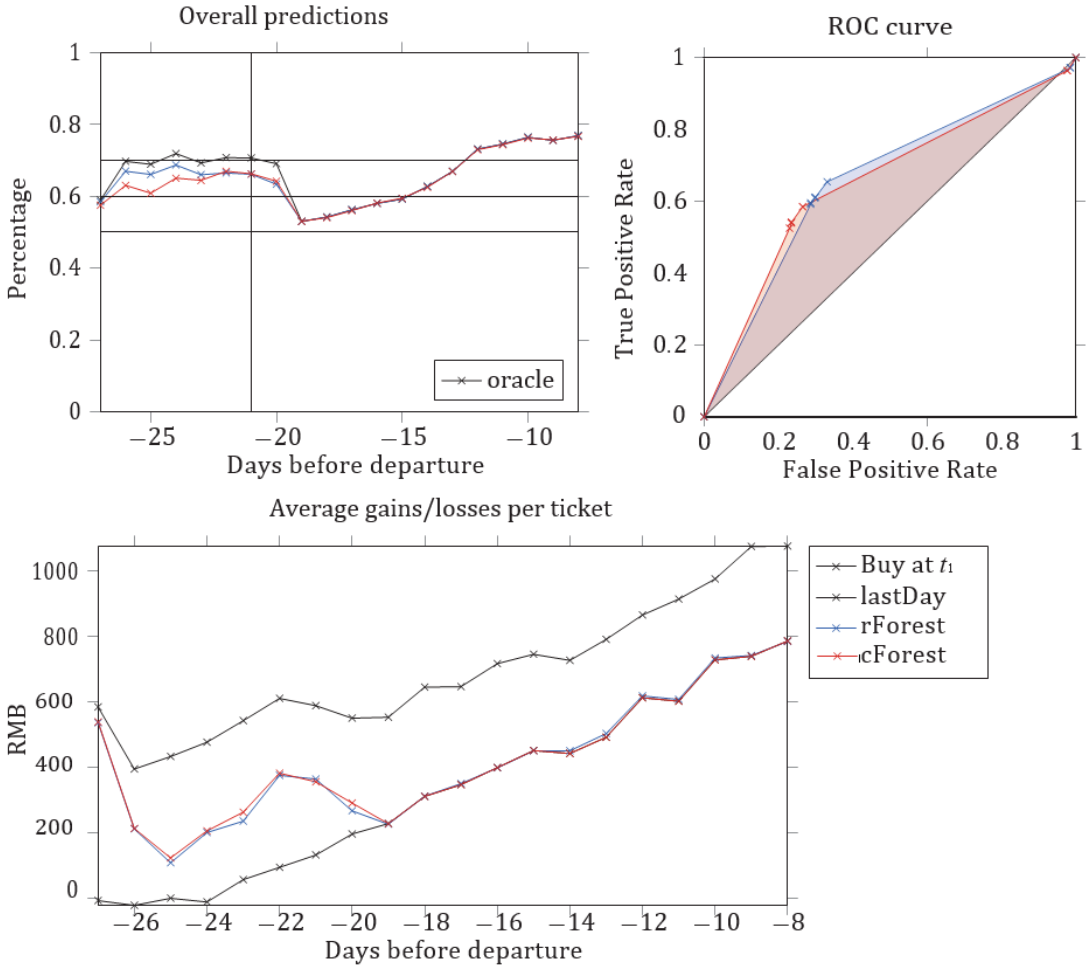

FIGURE 5.13 Difference of performance between the variable selection by rForest and cForest

Therefore, the rForest method was adopted to the classification algorithms previously selected. We

obtain:

TABLE 5.4 - Comparison

Method	Groups NO.	Rate	Economic gain /t_1
RF	5	66 → 66%	240 → 400
RF	30	64 → 64%	100 → 100
Adaboost	5	63 → 65%	0 → 100
CART	5	62 → 63%	0 → 100
CART	30	62 → 64%	-300 → -200
C5.0	5	66 → 71%	-200 → 0
C5.0	30	61 → 67%	-700 → 500

The improvement is incredible on almost all algorithms. The performances of each algorithm are visualized in Figure 5.14. The algorithm C5.0 provided positive results after selecting the variables. To avoid over-learn from the decision trees, the same algorithms were applied on an independent basis of flights departing in January 2019. The curves tend to approach the point (1, 1). We observed that the logit EM succeeded in generalizing the predictions using an identical ROC curve. All the approaches seem to have difficulties in predicting correctly at 18 days before the departure date. Whatever the threshold applied, the performances are either equal or lower than the immediate purchase. It can be explained by the demand-dependent daily adjustment during the days approaching to the departure date.

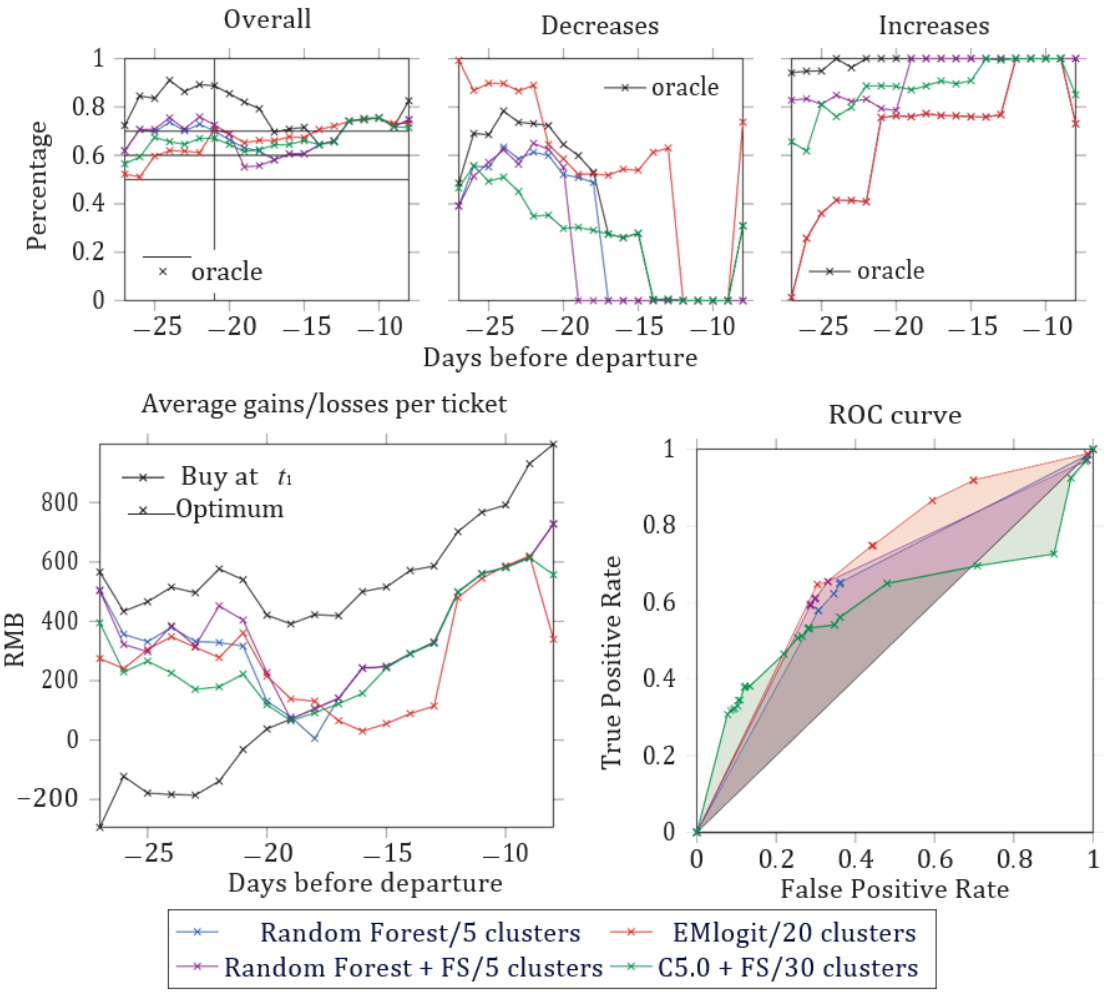

FIGURE 5.14 Comparisons of the cross validation of the 4 selected algorithms

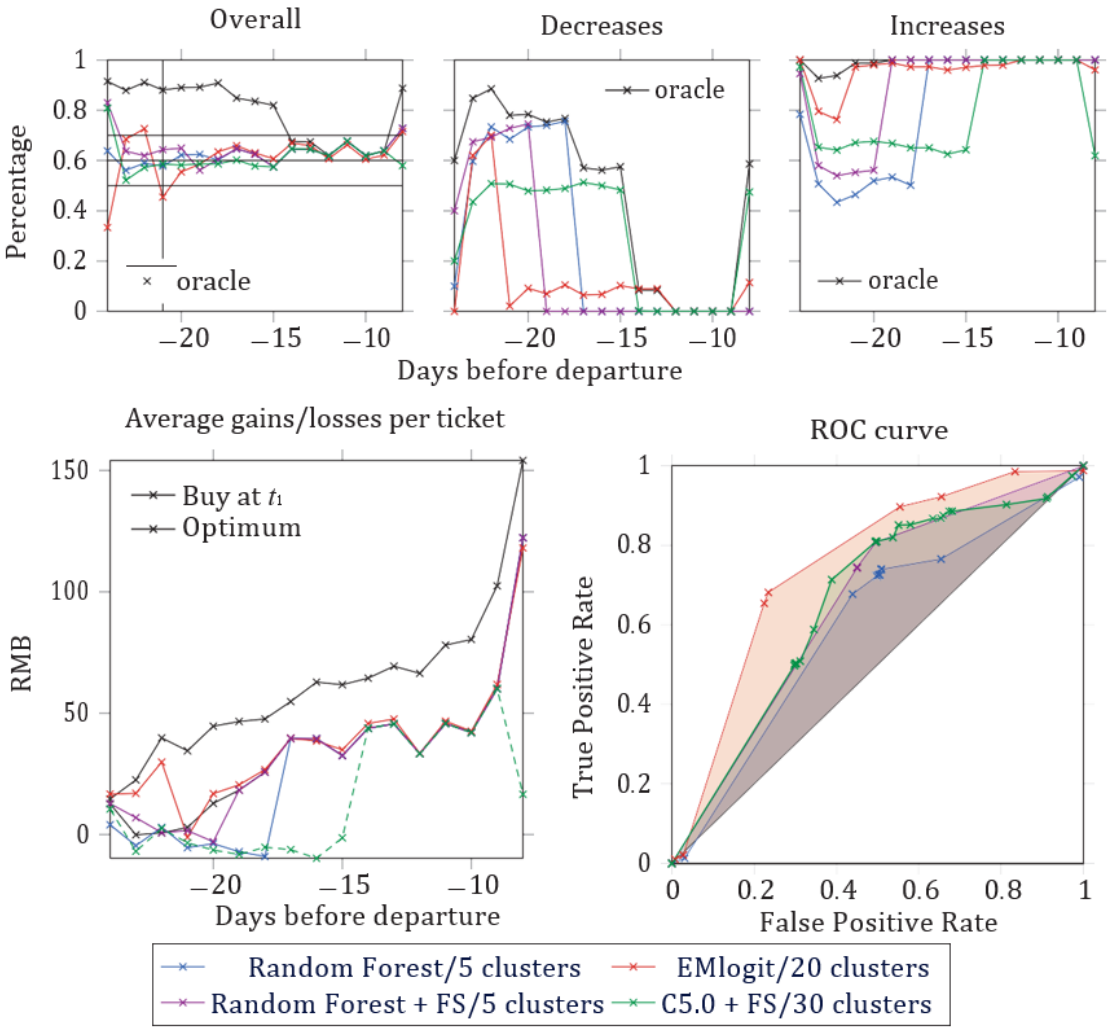

FIGURE 5.15 Comparisons of the 4 algorithms selected on an independent basis

5.2.3 Influence of external parameters

In the segmentation phase, the behavioral topology of our learning base has a great influence on the final predictions. All possible trajectories were reflected to refine the attribution of behavior to a test flight. As the sample of the two-year's flights is sufficient to produce a quality segmentation, additional historical flights may even be counterproductive. Therefore, in Figure 5.16, dissatisfactory performance is observed with the first 10 percent of the database. The performance was improving when 30 to 50 percent of the learning base was tested. The rate of satisfactory predictions was decreased when another 20 percent was added. Therefore, half of the database was chosen.

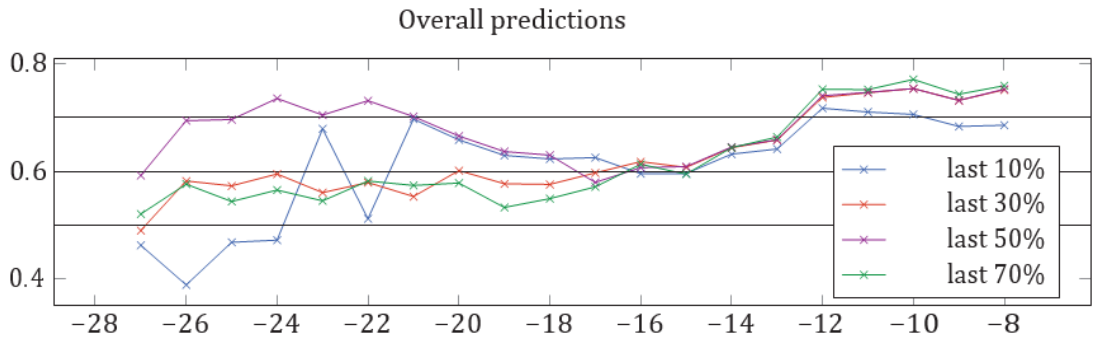

FIGURE 5.16 - Evolution of the rate of satisfactory predictions with the increase in the learning base, for a couple EM / 5 clusters and Random Forest.

5.3 Extensions

5.3.1 Direct prediction

The direct prediction adopts the learning algorithm to directly predict the variation at time t. Figure 5.17 demonstrates the performance of EM logit and random forests in direct prediction. The results are equivalent or even worse than a model prediction.

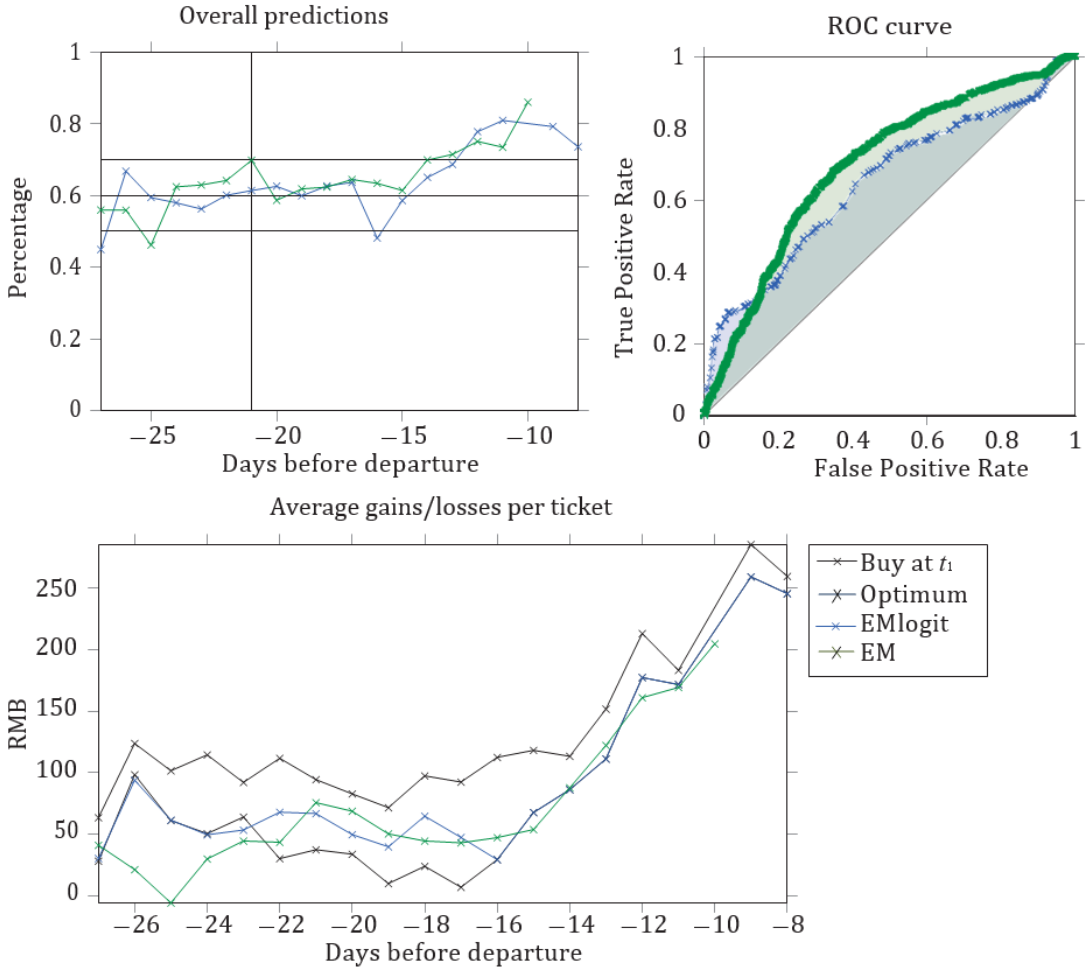

FIGURE 5.17 Evolution of performance in direct prediction by EM and EM logit

5.3.2 Evolution of performance over time

The previous experiments were carried out on a test basis between January 2019 and February 2019, two months after the last price collected from the learning base. The model limits in terms of longevity were discussed. How long does the prediction stay efficient? This information provides schedule to the systematic renewal of our model. Figure 5.18 shows that the performances of the first two months are above 63%, while the rates of satisfactory predictions and the ROC curves from the 3rd month were reduced. A daily performance monitoring system was required to renew our model.

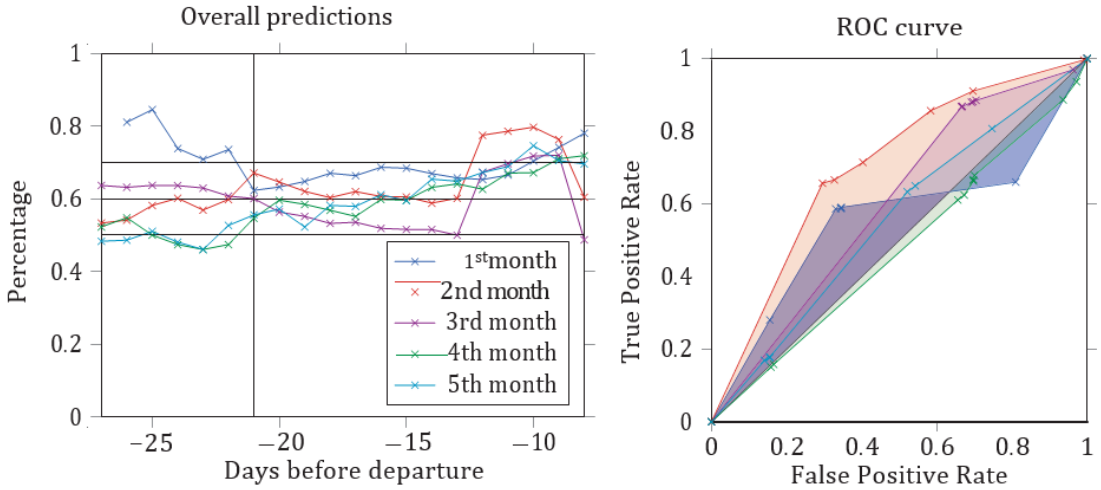

FIGURE 5.18 - Evolution of EMlogit performance for 20 clusters over the first 5 months

5.3.3 Base extended to 90 days

Finally, the sample base extended to a 90-day flight was tested. All user searches with at least 60 separate collection days over a set of 90 days were performed. Algorithms were applied to observe the adaptability to new data. In Figure 5.19, the results after a segmentation by EM with 5 clusters and a classification by random forests were visualized. Other combinations of algorithms provides similar results. Our approach was not adapted to a large base of flights of different nature as the results shown. Possible modifications including reconstructing the gray levels by increasing the size of the pixels. However, the period was too long for matching the similar flights. It was suggested to divide into three thirty-day periods. The whole process was performed on each of these parts to reduce differences between the behaviors. The prediction performance on the furthest two parts from the departure date provides better results, as the rates of satisfactory predictions tend to decrease towards 20 days before departure. It indicates that behaviors are more predictable and stable until one month before departure.

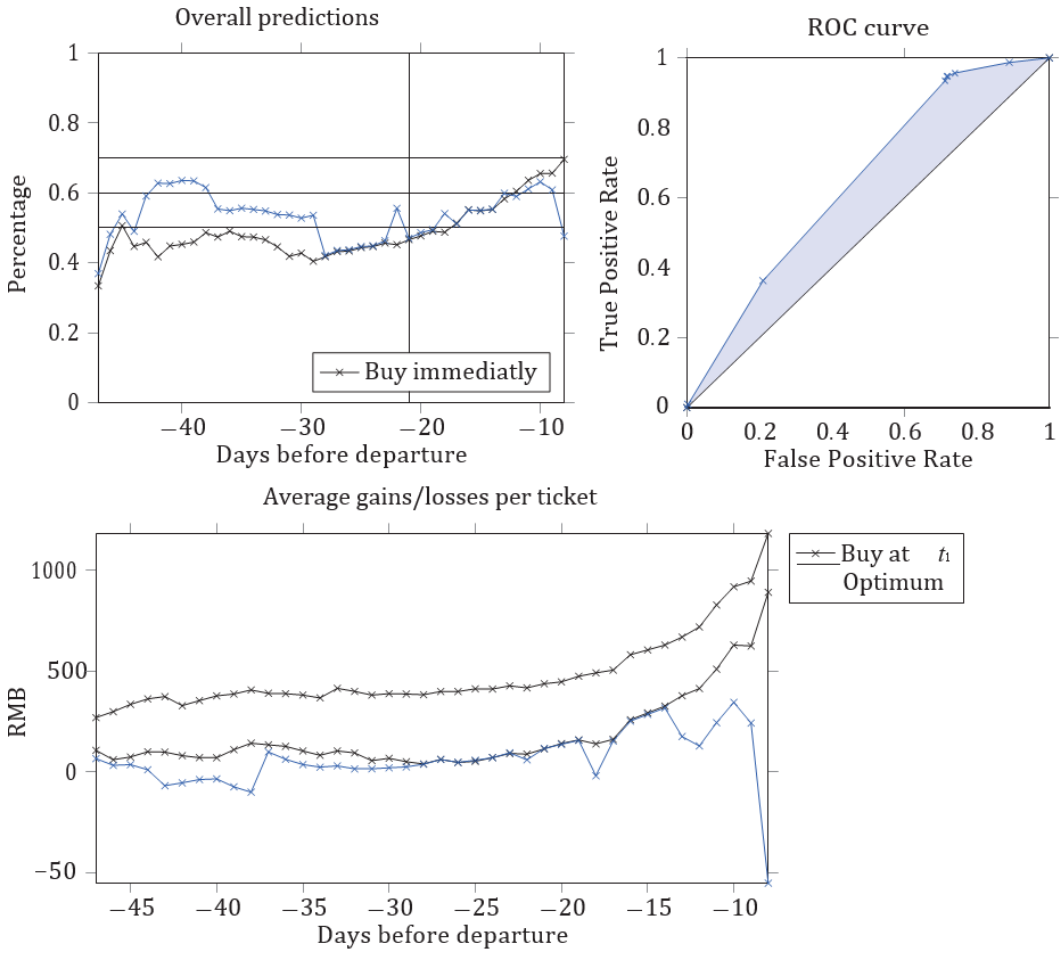

FIGURE 5.19 - Performance changes based on 90-day flights. EM 5 clusters

5.4 Conclusion

The qualities of the different algorithms were highlighted. Four configurations were tested on an independent basis. The values of ϕ in the learning base was balanced. The calculation of 3-day or 15-day prediction was modified. The problem was complex with multiple optimization criteria such as user gains, prediction rate, and minimization of false negatives. The optimal choice of the classifier for each prediction provides satisfactory results. Two complementary approaches were studied: first price change in prediction and direct prediction. The prediction performance was expected to improve with a simplified problem and reduced errors. The inflection points in the ROC curves spread, but the performances remain identical. This finding validates our approach to predicting behaviors.

5.4.1 Contributions and limitations

The analysis was driven by the industrial implication of this model. Different players belonging to the air industry have diverse needs. All will be benefiting if they could predict itinerary price and demand accurately. Among these players, airlines could improve the product customization such as prices and schedules to maximize expected revenues. Travel agencies could maximize conversion

through sorting or filtering travel alternatives after a search request. Finally, potential customers will benefit from the customized schedule recommendation with a lower price, through flash sale and pop-up advertisement offered by mobile applications.

The main contributions of this paper can be summarized as follows:
(a) a dynamic pricing predictor was proposed to maximize expected contribution for potential customers at a travel request level. (b) a machine learning approach was developed to improve the overall prediction accuracy. (c) the dataset is more comprehensive. Round-trip alternatives instead of just one-way were studied. Multiple markets, different travelers profiles, and different sales channels were analyzed in a single data set of travel request sessions.

The quality of the database is limited by the following factors. The search logs were extracted from a limited source, which is mainly a leisure-travel website. The model may overestimate price sensitivity. The bookings generated from Qunar data only represents GDS bookings. Other sources of bookings, such as airline websites, were not included. This limitation could also cause unpredictable bias as it depends on the markets. Furthermore, different performance metrics should be applied.

5.4.2 Further research directions

The future research directions can be classified in the following three aspects: studying alternative prediction models, adopting multiple data, and exploring the relationship between pricing strategies and revenue management strategies. This field has been dominated by traditional models. Recent progress on machine learning methods brings new opportunities to predict prices. These new algorithms have been proven to show their efficiency when the data are unstructured. Multiple data is also worthy to explore. The main limitation of this study is related to the fact that the data are generating from Qunar, a single sales channel. The passengers booking directly in airline websites are not represented in our dataset. However, obtaining such an exhaustive dataset composed of all bookings on all channels for all competitors is very challenging. Airlines may have difficulties to observe the real-time prices and booking information of competitors in the market. Both aspects are key to predict accurate prices. Furthermore, insights about the learned groups should be analyzed. What's the typical feature of each group? Finally, one of directions is to explore the relationship between pricing prediction and revenue management strategies. The key issues of pricing prediction are the access to real-time data. Thus, pricing strategies should be a great complementary module to revenue management strategies. Modern revenue management system also employs customer expectation that are overridden by the pricing optimizer. How could both revenue management and pricing strategies be adapted to maximize long-term airline revenues?

References

Abdella, J. A., Zaki, N., & Shuaib, K. (2018, November). Automatic Detection of Airline Ticket Price and Demand: A review. In 2018 International Conference on Innovations in Information Technology (IIT) (pp. 169-174). IEEE.

Alderighi, M., Cento, A., & Piga, C. A. (2011). A case study of pricing strategies in European airline markets: The London–Amsterdam route. Journal of Air Transport Management, 17(6), 369-373.

An, B., Chen, H., Park, N., & Subrahmanian, V. S. (2016, August). MAP: Frequency-based maximization of airline profits based on an ensemble forecasting approach. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 421-430).

An, B., Chen, H., Park, N., & Subrahmanian, V. S. (2017). Data-driven frequency-based airline profit maximization. ACM Transactions on Intelligent Systems and Technology (TIST), 8(4), 1-28.

Annaka Kalton, Pat Langley, Kiri Wagstaff and Jungsoon Yoo. Generalized clustering, supervised learning, and data assignment. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 299–304. ACM, 2001.

B. Efron and R. J. Tibshirani. An Introduction to the Bootstrap. Chapman & Hall, Singapore, 1993.

B. Smith, J. Leimkuhler, R. Darrow, and Samuels. Yield management at American Airlines. Interfaces, 22 (1): 8–31, 1992.

Barry C Smith, John F Leimkuhler, and Ross M Darrow. Yield management at American airlines. Interfaces, 22(1) :8–31, 1992.

Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. BMC Bioinformatics, 8(25), 2007.

Chen, Y., Cao, J., Feng, S., & Tan, Y. (2015, October). An ensemble learning based approach for building airfare forecast service. In 2015 IEEE International Conference on Big Data (Big Data) (pp. 964-969). IEEE.

CT Shaw and GP King. Using cluster analysis to classify time series. Physica D: Nonlinear Phenomena, 58 (1): 288–298, 1992.

D. J. Daley and D. Vere-Jones. An introduction to the theory of point processes. Flight. I. Probability and its Applications (Singapore). Springer-Verlag, Singapore, second edition, 2003. Elementary theory and methods.

Dell Zhang and Yisheng Dong. Semantic, hierarchical, online clustering of web search results. In Advanced Web Technologies and Applications, pages 69–78. Springer, 2004.

Domínguez-Menchero, J. S., Rivera, J., & Torres-Manzanera, E. (2014). Optimal purchase timing in the airline market. Journal of Air Transport Management, 40, 137-143.

E. Poutier and P. Legohérel. Revenue Management - Optimization of sales in the

Elman, J. L. (1990). Finding structure in time. Cognitive science, 14(2), 179-211.

Escobari, D. (2014). Estimating dynamic demand for airlines. Economics Letters, 124(1), 26-29.

Etzioni, O., Tuchinda, R., Knoblock, C. A., & Yates, A. (2003, August). To buy or not to buy: mining airfare data to minimize ticket purchase price. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 119-128).

Glenn W Milligan and Martha C Cooper. An examination of procedures for determining the number of clusters in a data set. Psychometrical, 50 (2): 159–179, 1985.

Groves, W., & Gini, M. (2013, May). An agent for optimizing airline ticket purchasing. In Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems (pp. 1341-1342).
Groves, W., & Gini, M. (2015). On optimizing airline ticket purchase timing. ACM Transactions on Intelligent Systems and Technology (TIST), 7(1), 1-28.
In Data Mining, 2002. ICDM2003. Proceedings. 2002 IEEE International Conference on, pages 717–720. IEEE, 2002.
J. Ross Quinlan. Induction of decision trees. Machine learning, 1 (1): 81–106, 1986.
Jacob Kogan. Introduction to Clustering Large and High-Dimensional Data. Cambridge University Press, Singapore, NY, USA, 2007.
John Ross Quinlan. C4. 5: programs for machine learning, volume 1. Morgan Kaufmann, 1993.
Jürgen Beringer and Eyke Hüllermeier. Online clustering of parallel data streams. Data & Knowledge Engineering, 58 (2): 180–204, 2006.
Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105.
L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA, 1984.
Lantseva, A., Mukhina, K., Nikishova, A., Ivanov, S., & Knyazkov, K. (2015). Data-driven modeling of airlines pricing. Procedia Computer Science, 66, 267-276.
Laura Tolosi and Thomas Lengauer. Classification with correlated features: unreliability of feature ranking and solutions. Bioinformatics, 27(14) :1986–1994, 2011.
Lawrence R Weatherford and Samuel E Bodily. A taxonomy and research overview of perishable-asset revenue management: yield management, overbooking, and pricing. Operations Research, 40(5) :831–844, 1992.
Leo Breiman. Random Forests. Machine Learning, 45 (1) : 5–32, October 2001.
Leonard Kaufman and Peter J Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, 2005.
Li, J., Granados, N., & Netessine, S. (2014). Are consumers strategic? Structural estimation from the air-travel industry. Management Science, 60(9), 2114-2137.
Liu, J., Liu, B., Liu, Y., Chen, H., Feng, L., Xiong, H., & Huang, Y. (2017). Personalized air travel prediction: A multi-factor perspective. ACM Transactions on Intelligent Systems and Technology (TIST), 9(3), 1-26.
Malighetti, P., Paleari, S., & Redondi, R. (2009). Pricing strategies of low-cost airlines: The Ryanair case study. Journal of Air Transport Management, 15(4), 195-203.
Mantin, B., & Koo, B. (2010). Weekend effect in airfare pricing. Journal of Air Transport Management, 16(1), 48-50.
Marc Möller and Makoto Watanabe. Advance purchase discounts versus clearance sales *. The Economic Journal, 120 (547): 1125–1148, 2010.
Michail Vlachos, Jessica Lin, Eamonn Keogh, and Dimitrios Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series. In in Proc.Workshop on Clustering High Dimensionality Data and Its Applications. Citeseer, 2003.
Mingjin Yan. Methods of Determining the Number of Clusters in a Data Set and a New Clustering Criterion. PhD thesis, Virginia Polytechnic Institute and State University, 2005.
Mumbower, S., Garrow, L. A., & Higgins, M. J. (2014). Estimating flight-level price elasticities

using online airline data: A first step toward integrating pricing, demand, and revenue optimization. Transportation Research Part A: Policy and Practice, 66, 196-212.
Narangajavana, Y., Garrigos-Simon, F. J., García, J. S., & Forgas-Coll, S. (2014). Prices, prices and prices: A study in the airline sector. Tourism Management, 41, 28-42.
Peter P Belobaba. Survey paper — airline yield management an overview of seat inventory control. Transportation Science, 21 (2): 63–73, 1987.
Puller, S. L., & Taylor, L. M. (2012). Price discrimination by day-of-week of purchase: Evidence from the US airline industry. Journal of Economic Behavior & Organization, 84(3), 801-812.
Robert Tibshirani, GuentherWalther, and Trevor Hastie. Estimating the number of clusters in a dataset via the gap statistic. 63 :411–423, 2000.
S. Daudel and G. Vialle. Yield management: applications to air transport and other service industries. Shanghai: Presses of the Institute of Air Transport, 1994.
Services: Optimization of sales in Services. Marketing - Communication. Dunod, 2017.
Sidney Resnick. Adventures in stochastic processes. Birkhäuser Boston Inc., Boston, MA, 1992.
T. Hastie, R. Tibshirani, and J. H. Friedman. The Elements of Statistical Learning. Springer, corrected edition, July 2003.
Tak-chung Fu, Fu-lai Chung, Vincent Ng, and Robert Luk. Pattern discovery from stock time series using self-organizing maps. In Workshop Notes of KDD2001 Workshop on Temporal Data Mining, pages 26–29. Citeseer, 2001.
TK Moon. The expectation-maximization algorithm. IEEE Signal Processing Magazine, 13 (6): 47–60, November 1996.
Tziridis, K., Kalampokas, T., Papakostas, G. A., & Diamantaras, K. I. (2017). Airfare prices prediction using machine learning techniques. In 2017 25th European Signal Processing Conference (EUSIPCO) (pp. 1036-1039). IEEE.
Vu, V. H., Minh, Q. T., & Phung, P. H. (2018, January). An airfare prediction model for developing markets. In 2018 International Conference on Information Networking (ICOIN) (pp. 765-770). IEEE.
W. J. Youden. Index for rating diagnostic tests. Cancer, 3(1) :32–35, 1950.
Wen, C. H., & Chen, P. H. (2017). Passenger booking timing for low-cost airlines: A continuous logit approach. Journal of Air Transport Management, 64, 91-99.
Wen, Chieh-Hua, and Po-Hung Chen, "Passenger booking timing for low-cost airlines: A continuous logit approach," Journal of Air Transport Management, vol. 64, pp. 91-99, 2017.
Wesam Barbakh and Colin Fyfe. Online clustering algorithms. International Journal of Neural Systems, 18 (03): 185–194, 2008.
William M Rand. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association, 66 (336): 846–850, 1971.
Xu, Y., & Cao, J. (2017, December). OTPS: A decision support service for optimal airfare Ticket Purchase. In 2017 IEEE International Conference on Big Data (Big Data) (pp. 1363-1368). IEEE.
Yimin Xiong and Dit-Yan Yeung. Mixtures of amra models for model-based time series clustering.
Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In Computational learning theory, pages 23–37. Springer, 1995.
Yuan, H., Xu, W., & Yang, C. (2014, June). A user behavior-based ticket sales prediction using data mining tools: An empirical study in an OTA company. In 2014 11th International Conference on Service Systems and Service Management (ICSSSM) (pp. 1-6). IEEE.