Universidade Federal do Paraná
Setor de Ciências Exatas
Departamento de Estatística
Programa de Especialização em *Data Science* e *Big Data*

Pedro Martins Moreira Neto

Car Category Recommendation: A Classification Approach

Curitiba
2019

Pedro Martins Moreira Neto

Car Category Recommendation: A Classification Approach

Monografia apresentada ao Programa de Especialização em *Data Science* e *Big Data* da Universidade Federal do Paraná como requisito parcial para a obtenção do grau de especialista.

Orientador: Prof. Daniel Weingartener

Curitiba
2019

Car Category Recommendation: A Classification Approach

Pedro Martins Moreira Neto[1]

[1]Departamento de Informática, Universidade Federal do Paraná

Resumo

Um sistema de recomendação pode ser implementado em qualquer loja online que ofereça produtos ou serviços com o objetivo de melhor a experiência do usuário e aumentar lucros. O mesmo funciona para lojas de aluguel de carros, nelas um sistema de recomendação também pode ser implementado, de modo que liste o carro que melhor se encaixa com as preferências do usuário. Este projeto tem objetivo de criar um sistema de recomendação de veículos de aluguel usando uma abordagem de classificação com um classificador MLP. Resultados parciais demonstram que esta abordagem tem potencial de sucesso, uma vez que acurácia do classificador em prever qual veículo deve aparecer na primeira posição é de 30% e de 57% de que o veiculo irá aparecer até a tercerira posição.

Palavras-chave: Sistema de Recomendação, Classificação, MLP, Aluguel de Carros

Abstract

A recommendation system can be implemented at any online shopping that offers products or services with the objective to offer a better experience to the customer and increase profits. When renting cars, we also can explore the benefits of recommending the best car to match customer wishes. This project works towards building a recommendation system by using a classification framework applying an MLP as a classifier. Partial results have demonstrated that the recommendation can be made with an accuracy of 30% that the best match car will be in the first position of search and that with of accuracy 57% the classifier can predict that the desired car category will be up to the third position.

Keywords: recommendation systems, classification, MLP, car rental

1. Introduction

Recommendation Systems are built in order to predict which product or service is good to the user, and it does it by looking at and analyzing historical data. These systems are widely used throughout online stores, and they aim to provide to the user a better shopping experience as well as increase the company profit.

The task of recommending products or services can be seen like a machine learning problem in which the customer preferences needs to be modeled by looking at his historical data or looking at similar consumption patterns. The domain of recommendations varies according to each business.

In this project, we are going to explore the car rental business. The company who provided the data to this project offers a web portal in which the client can look at many options of cars and categories and choose the one he prefers. With the purpose of offers to the client a car category that best matches his needs, we are going to work towards building a recommendation system using a classification approach.

2. Overview of Data Science Process

Before we dive into the details of this project, we start with an introduction to the basic terminology and concepts regarding the steps were taken to develop this project.

When doing Data Science we usually take a sequence of steps in order to achieve the desired result – i.e., to answer businesses questions. These steps are data acquisition, data preparation, analysis, and finally training a machine learning model in our data. And this cycle repeats whenever is needed. The process shown is by no means written in stone, thus one can add or skip any given step if he wanted. Also, each step can be split up into two or more steps, for instance, data acquisition may require transforming raw data (images, sounds, etc) into machine-interpretable data and

merge them with numerical data before to jump to the next step. The blueprint above gives us a direction on how to start work with data, thus obtaining responses to the questions.

2.1. Data Acquisition

Data acquisition refers to the process of obtaining meaningful data from various data sources. A data source can be either relational databases, CSV files, scrapped data or either raw format data like sounds, images which require minimal processing to turn them into information. Commonly the result from this step will be consolidated into a data table, CSV file or data frame which then will be analyzed further in the process.

2.2. Data Preparation

After acquiring the data it is necessary to investigate how good, complete and consistent this data is. Therefore, the data preparation step exists to either correct inconsistencies, add new pieces of information and prepare the data before passing it through analyses and modeling steps.

This step is important because most machine learning algorithms require data to be pre-processed by eliminating missing values and invalid values which can lead to misleading outcomes, by normalizing data to the same scale, and by creating new features that will add to the model performance.

2.2.1. Data Cleaning

It is common to real-world data sets come with a mixture of problems, for example, missing values, improperly formatted, and duplicated data. Such problems can be derived from combining multiple data sources, bad quality inputs, and so on. It is crucial to identify and correct these issues by replacing, removing or modifying the problematic values from the dataset because they usually lead to inaccurate analysis and therefore inaccurate decisions.

2.2.2. Feature Scaling

It is common that in the data set we find features (columns) with very different scales. This is problematic because many machine learning algorithms attempt to find trends in the data by comparing the features data points because the feature with a larger scale will complete dominating the feature with a smaller scale. Thus, the goal of feature scaling step is to apply a normalization function to every data point of the data set to force them to be in the same scale and so ensuring that each feature is equally important to the algorithm.

A very common normalization function is the min-max normalization, which is given by the formula:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

The min-max normalization maps each data point of a feature between the range [0,1]. The value 0 represents the minimum value of the feature, and 1 the maximum value of the features, all other values are mapped between this range.

Data preparation is vital to the data science process, and it is advised to perform it every time a new data is added to the dataset. Good data preparation produces a well-curated and clean data that results in more accurate model outcomes.

2.3. Data Analysis

To dig deeper into data, find correlations, and patterns, in order to understand data behavior, is the core of this step. Here we explore data through visual elements like charts and maps, we apply statistical methods focusing on identifying significant patterns and trends in our data. Data analysis and exploration will provide us useful insights that will support the creation of new features in the dataset in a process called feature engineering.

2.3.1. Feature Engineering

Feature engineering is when creativity takes place. In this step, new variables or features are built in the dataset, usually based upon previous business knowledge or recent discoveries in the data. The addition of new features are meant to gain deeper insights on the problem and can either be done by mix-and-matching pre-existing features or by adding external data sources to your data.

To illustrate, in the context of car rental where the feature date of booking is available, to build features to identify if this date is a weekend or holiday can give valuables insights about customer behavior.

2.4. Training Machine Learning Model

The next step of our workflow is choosing a machine learning model and training it in our previous prepared

date. At this point, we are going to select a target variable, which is the factor we are trying to gain a deeper understanding to answer our question. Once the target variable is determined we need to verify if the nature of our problem is classification or regression problem. A classification problem is when we want to determine to what class a certain an observation belongs, for example, filtering the incoming emails between 'inbox' and 'spam' it is a classification task.

Classification problems are not bounded to binary cases, in fact, there are many cases in which the algorithm needs to classify the observation between more than two classes, such as classifying plants into different taxonomies or classifying if a customer will prefer an economical car rather than an SUV while renting a car for the weekend.

In contrast to classification, we will do regression not to classify between classes but rather estimates a numeric value which explores the relationship between variables, i.e., estimating housing and stock prices as in financial forecasting are regression problem since its outputs are numerical and continuous values.

2.4.1. Class imbalance

Often the dataset is said to suffer for class imbalance problem, which basically the class distribution is very imbalanced [3]. This is often seen in medical data where is more common to have data about healthy people than a sick people, i.e., to predict if a person has cancer, surely there will have way more data about a person who doesn't have cancer than data of those who does have cancer. In classification learning algorithms, such structure can low the predictive accuracy of the algorithm, thus it is important to manipulate our data to mitigate this problem.

There are two strategies to solve the data imbalance problem: over-sample and under-sample. Oversampling is concerned to complement the training data with copies or synthetically generated data from the class that is least represented in the dataset. One way to perform oversampling is to use SMOTE algorithm [4], which stands for Synthetic Minority Over-sampling Technique. SMOTE works by generating synthetic examples from the minority class by operating in "feature space" rather than "data space". The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any of the k minority class nearest neighbors.

2.4.2. Feature Selection

Before feeding our machine learning algorithm with our data, we will assess if all of the features present in the dataset are essential to predict our model. The feature selection can be either done manually, by dropping columns which are known to be irrelevant to the problem, and automatically by applying some sort of algorithm, such as genetic algorithms.

Manually selecting or dropping features, are very tied with the analysis of the data. A feature that presents a constant value (like zero), will not improve or have a negative impact in the model, hence delete it will not affect the outcome of the model. Equally important is to select the features based on the business knowledge, if a feature is not relevant to business so it should be removed from the dataset. As opposed to feature engineering, feature selection doesn't involve creating new features, but instead removing e the ones that don't add value to the model.

At this point we may feel that every feature in our dataset is important to predict or estimate our target variable, still, we can apply an automatic feature selection algorithm to help us those features that actually add to model's value. In particular, the Genetic Algorithm can be applied to pick up the right number of features in order to create a predictive model.

2.4.3. Genetic Algorithms

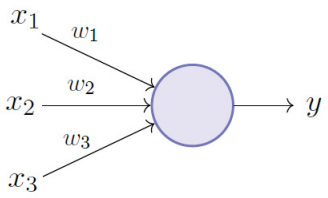
The idea behind Genetic Algorithms (GA) is inspired by the idea of natural selection, where the strongest individual of its species survive after generations. Bringing it to the machine learning world, each individual can be seen as a subset of the features of our dataset. The algorithm creates many subsets of the features combining them together, forming then a population. Each of these individuals will then be assessed against a fitness metric, such as accuracy metric in the case of a classification problem. Only those individuals who presented good fitness will survive to the next generation, thus preserving the features that have higher decision making power to the problem. Therefore, generation after generation the GA will allow the best combination of features to emerge amongst all solutions, and this subset of features that are more important to predict our target variable should be used in our machine learning model.

At this point, the data is ready to be fed into a machine learning algorithm. Many of them exist out there,

but in the context of this project, two classification algorithms were used: Random Forest and MLP.

2.4.4. Multilayer Perceptron

A Multilayer perceptron is a machine learning algorithm in which the basic unit is called a Perceptron. A perceptron is a linear classifier; that is, an algorithm that classifies any given input into to separated categories with a straight line, through the multiplication of feature vector x by the weights w and adding a bias b: $y = w * x + b$ [5].



Perceptron Model (Minsky-Papert in 1969)

Figura 1

The output of a perceptron is single real value which is obtained by performing a linear combinator of a the weights, features and bias. Then, this result is fed to a nonliner activation function denote by φ in the equation below:

$$y = \varphi\left(\sum_{i=1}^{n} w_i x_i + b\right) = \varphi\left(w^T x + b\right) \qquad (2)$$

From the combination of multiple single perceptrons classifiers comes the idea of Multilayer Perceptron (MLP). An MLP is a combination of more than two perceptrons in order to form a network. The first layer of this network is called the input layer, the last layer is called the prediction or decision layer and the layers between them are called hidden layers.

The input layer holds the feature vector, which is then computed through the hidden layers of the network. The computed results fall into the final prediction layer, which is responsible for output either a single class probability, in the case of binary-class problems, or output many probabilities when the problem is considered multi-class. In a multi-class scenario, the final layer is called softmax.

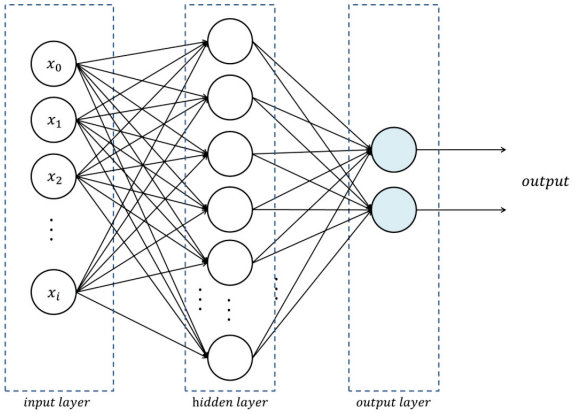


Figura 2: Multi-layer Perceptrons with one hidden layer. (Source: `https://www.cc.gatech.edu/~san37/post/dlhc-fnn/`)

3. Car Category Deep-Recommendation Systems

There are essentially three types of recommendation systems: collaborative filtering, content-based and hybrid recommender system [1]. Collaborative filtering makes recommendations by learning how user and item interact historically. The content-based recommendation is based mainly on comparisons across items' and users' auxiliary information, and hybrid model refers to the recommender system that integrates two or more types of recommendation strategies [2].

In recent years a deep learning (DL) approach to recommendation systems have emerged, and the techniques used range from simple MLP to Deep Reinforcement Learning, and Recurrent Neural Network. Deep learning offers a series of advantages to recommendation systems, such as nonlinear transformation that makes possible to capture the complex and intricate user item interaction patterns, and in the representation leaning since DL can help to reduce the efforts in hand-craft feature design [2].

In this project, we want to explore how deep-learning, more specifically a multilayer perceptron, performs in recommending a car category to costumers while renting a car. The categories available are economic, compact, intermediate, minivan, mini, SUV, luxury, fullsize, standard, premium, special, utility, and van. The dataset used comes from a major rental car company based in Brazil, and it contains data from one-year car bookings both domestic and international.

3.1. Data Source

The data was fetched from an Oracle database combining eleven different tables containing information about booking, client, rental company, car, currency, and geographical data. One year of data was collected (2018-01-01 to 2019-01-01) resulting in a dataset containing 773.222 observations and 115 columns (variables), and it was saved into a local database.

3.2. Data Analysis and Preparation

In this section, we are going to expose some of the analysis made in this project. We are going to focus on the analyses that brought some insights to build new features.

There are three basic forms of tourism: domestic tourism, inbound tourism, and international/outbound tourism [6]. Domestic involve residents of the given country traveling only within their own country. Inbound, from the point of view of Brazil, is when the residents from another country come to Brazil. International tourism refers to the activities of a resident visitor outside of their country of residence.

Since the data is provided by a Brazilian company, we will take the point of view of the company to determine the types of tourism. Thus, a domestic booking refers to someone who is in Brazil and rent a car to use within Brazil, and a domestic booking is when a person is in any country but Brazil, and rents a car to use in any other country that is not Brazil. That made clear, we further investigate how are the behavior of the customers in each of the segments above: domestic and international.

3.2.1. Domestic Market

The domestic bookings represent 71% of total annual bookings against 29% of international bookings (figure 3). For this reason, we will concentrate the analyses on this portion of the data.

In the domestic market, the southeast accumulates most of the bookings as we can verify on figure 4, just in São Paulo holds 24% of the bookings and the southeast 43% in 2018.

It's clear on chart 5 that the Brazilian customer prefers to rent cars from economy category with more than 55% of the bookings being economic. This behavior introduces an unbalancing problem, which will be addressed in the next step.

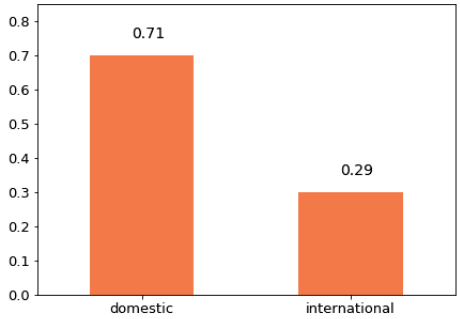


Figura 3: Distribution between domestic and international bookings

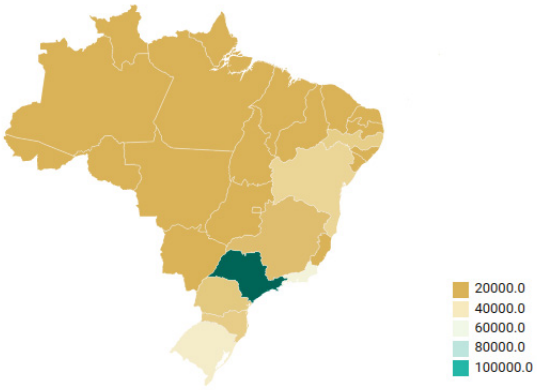


Figura 4: Booking per State

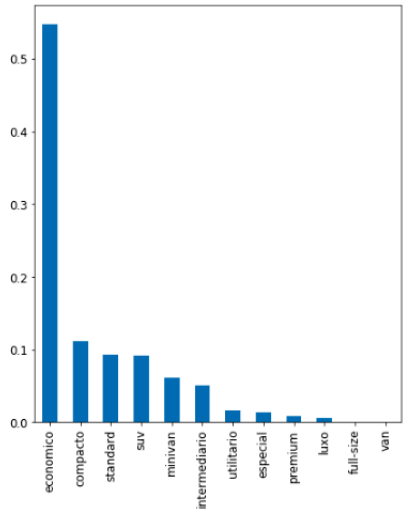


Figura 5: Volume per category

3.3. Feature Engineering

The monthly volume of the bookings can be seen in figure 6. It worth noting the spike of bookings on 27th of November, this 2018's Black Friday. In Black Friday the volume of bookings increases almost 250% in com-

parison to the rest of the year. Also, we noticed that there is some seasonality throughout the weeks, which we discover that on Saturdays the customers tend to rent less than the other days of the week. By looking at figure 6, it is reasonable to say that date features should be created to enrich the dataset. There are essentially three original date features in the data: date of the booking, pickup date, and dropoff date. In the table 1 we can see what were the features derived from the latter.

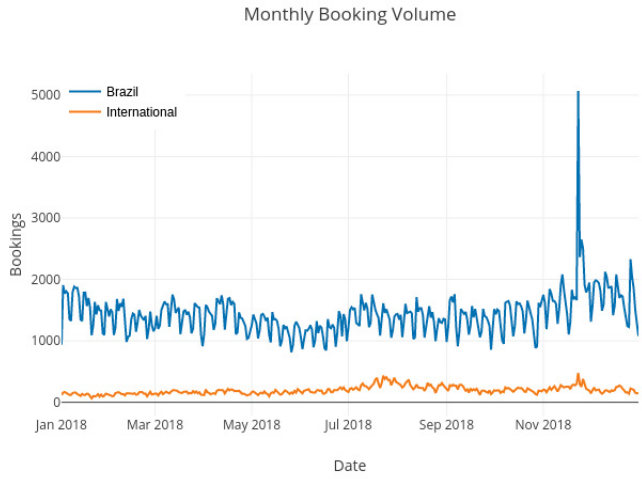


Figura 6: Monthly Booking Volume

The feature *leadtime* is given by subtracting pick-up date and the booking date, and it informs on the amount of the days the customer is planning the trip. A business trip may have less planning so an economical car can be more attractive, whereas a leisure trip may have a bigger lead time (planning) and a more expansive car can be considered. Same idea for *triplength*, with which we want to capture how many days this customer will be with the car.

By observing that some customers picks up a car on a given city and returns in another one, three other variables have been built: *cross-city*, *cross-state*, and *cross-location*. These variables are flags that are set to 1 when some of them are true. Joining the created features and the original features the final dataset has 126 features in total.

3.4. Feature Selection

Before training the algorithm, we want to select a set of features that are relevant to the business. As a first stage, a total of 36 out of 126 of features were selected manually based on the prior business knowledge. The selected features are:

Booking Date	Pick-up Date	Drop-off Date
day	day	day
week	week	week
month	month	month
dayofweek	dayofweek	dayofweek
weekend	weekend	weekend
monthstart	monthstart	monthstart
leadtime	leadtime	º
º	triplength	triplength

Tabela 1: Derived columns from date

1. id_moeda
2. id_moeda_destino
3. id_pais
4. id_estado_retirada
5. id_estado_devolucao
6. id_cidade_retirada
7. id_cidade_devolucao
8. aeroporto_devolucao
9. aeroporto_retirada
10. reserva_final_semana
11. locatario_pais_origem
12. locatario_pais_resid
13. locatario_idioma_vis
14. locatario_moeda_vis
15. reserva_mes
16. reserva_semana_ano
17. reserva_dia_semana
18. reserva_inicio_mes
19. reserva_final_semana
20. retirada_mes
21. retirada_semana_ano
22. retirada_dia_semana
23. retirada_inicio_mes
24. retirada_final_semana
25. devolucao_mes
26. devolucao_semana_ano
27. devolucao_dia_semana
28. devolucao_inicio_mes
29. devolucao_final_semana
30. lead_time
31. trip_length
32. cross_city
33. cross_state
34. cross_location
35. **veiculo_categoria**

Then, a Genetic Algorithm was applied to this set of feature in order to extract the most relevant features automatically. The GA ran through 50 iterations and optimizing the features with a pool size of length 35. The classifier used to validate the fitness solution was a random forest with F1-Score as a metric. The optimization algorithm converged fast, reaching the best solution score right in firsts iterations (figure 7).

The output of the genetic algorithm was a set of the best 23 features that maximize model performance. The automatically selected features are:

1. id_estado_devolucao
2. id_cidade_devolucao
3. aeroporto_devolucao
4. aeroporto_retirada
5. locatario_pais_origem
6. locatario_idioma_vis
7. locatario_moeda_vis
8. reserva_mes
9. reserva_inicio_mes
10. reserva_final_semana
11. retirada_mes
12. retirada_semana_ano
13. retirada_dia_semana
14. retirada_inicio_mes
15. retirada_final_semana
16. devolucao_mes
17. devolucao_semana_ano
18. devolucao_inicio_mes
19. lead_time
20. trip_length
21. cross_city
22. cross_state
23. cross_location

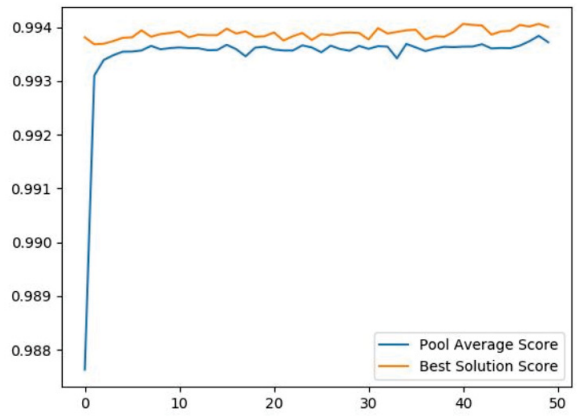


Figura 7: Feature Selection with GA: Performance

3.5. Balancing Class

As explained in section 2, when the classes are not equally represented in the dataset we have a problem with class balancing. In order to have accurate results, the classes should be either over-sampled or under-sampled, exposing the classifier to enough data samples of each class. In the problem of classifying car categories, we face the balancing problem since the customers tend to have a high preference for renting economic cars. This behavior can be seen in figure 5.

To balance the class distribution, we used the Synthetic Minority Over-sampling Technique (**SMOTE**) which creates synthetic points of the minority class by varying the attributes of the class. The result is a dataset with balanced classes (figure 8.

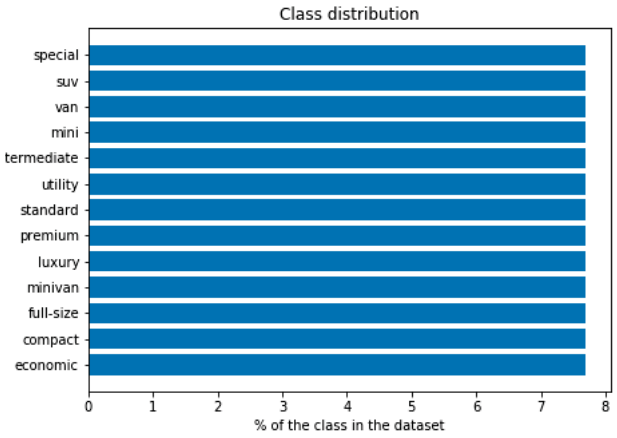


Figura 8: Class distribution after SMOTE

3.6. Training Phase

In the recent years, deep-learning models has surpassed many of the traditional approach's in various fields. In recommendation system is not different, and the concise but effective networks has demonstrated to be able to approximate any measurable function to any desired degree of accuracy[2]. Following the trend, in this project we also implemented a neural network in order to classify and recommend car category to the customer.

The architecture the network is quite simple (figure 9). It contains an input layer with size 24 which is the amount of features we have on our dataset. Followed by two hidden layers of 32 nodes, and in the output we have a softmax layer with size 14 that denotes the our classes. The layers are activated through the function **ReLu**, the optimizer used is **adadelta** and the loss function was **categorical cross-entropy**. The batch-size and epochs are 10.000 and 100 respectively.

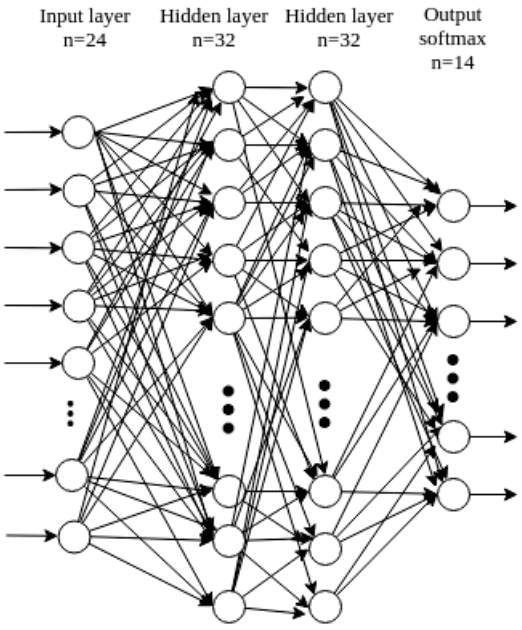


Figura 9: MLP Architecture

4. Results

Only the partial results of this project are presented. This is due to the fact that data provider terminated its participation in this project and consequently the availability of data was removed.

To measure the performance of the classifier, two metrics were used: top 1 accuracy, and top 3 accuracies. Top 1 accuracy shows how much the classifier makes

the right precision at first position, i.e., the class with maximum probability is the correct one. By the other hand, the top 3 accuracy measures the performance considering the top 3 maximum probabilities of the classes. Top 1 and Top 3 performance are presented in figure **??**. In both graphs, the blue line represents the performance metrics of training data, and the orange line is the validation data.

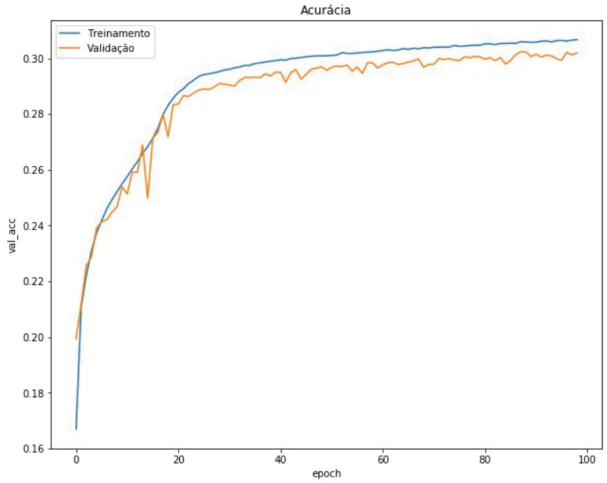


Figura 10: Top 1 Accuracy x Epochs

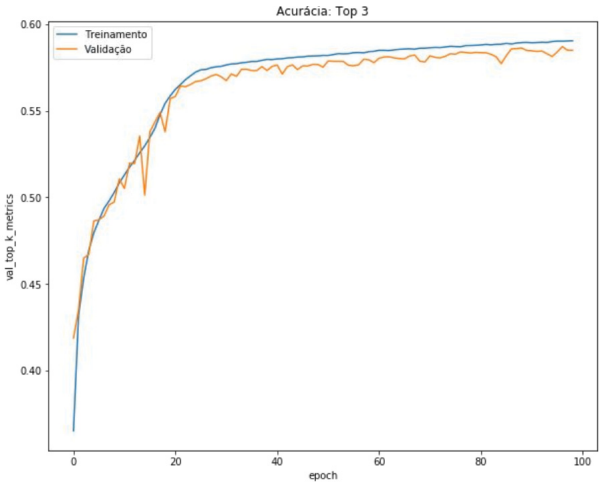


Figura 11: Top 3 Accuracy x Epochs

As can be seen in the figure 10, the accuracy on top 1 accuracy reached 30%, and in figure 11 the accuracy on top 3 reached 57%. These results shows that using the classifier approach to recommend a car category, we can predict with certain confidence that we will show to the customer the category he is looking for at first position of his search list, and with 60% confidence the we can present to him the category he wants up to third position of the list.

5. Conclusion

The goal of this project was to explore a classification approach to car category recommendations. The results have proven that is possible to build a good classifier to predict which car category will be shown in the first position of customer search. However, it is clear that to build a good classifier we need to explore more in-depth details such as to get confusion matrix, explore other forms of address data balancing, and explore simpler classifiers. But, once I stop working at the rental company, the access to data ceased and more analyses couldn't be made.

It is a real challenge to build a recommendation system. Beyond data analyses, there are many factors which should be taken into account, such as data integration and systemic support. Though, the results reached in this project were satisfactory to the scope of this project.

Referências

[1] G. Adomavicius and A. Tuzhilin. *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.* IEEE transactions on knowledge and data engineering **17**, 6 (2005).

[2] Zhang, Shuai and Yao, Lina and Sun, Aixin and Tay, Yi. *Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR)*, **52**, 1 (2019).

[3] Ling C.X., Sheng V.S. (2011) Class Imbalance Problem. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer. *SMOTE: Synthetic Minority Over-sampling Technique.*Joournal of Artificial Intelligence, (2002)

[5] Skymind. A Beginner's Guide to Multilayer Perceptrons (MLP). Accessed on 06/19/2019 **https://skymind.ai/wiki/multilayer-perceptron**

[6] Visit Britain. Introduction to tourism . Accessed on 06/19/2019 **https://www.visitbritain.org/introduction-tourism**