Uses and applications of Data Science in a car company

***Abstract*— During these pages, you will be guided through a path that relates our experience and learning outcomes during the final data engineering project. For this project we decided to make an analysis and exposure of machine learning and data science techniques applied to the automotive world and the car buying and selling sector, we hope you enjoy the journey.**

I. INTRODUCTION

For this project we have worked with a database on cars grouped into different brands and models. The data is distributed in such a way that each row corresponds to a different car and each column to a car's characteristic. These datum has been chosen because there are many observations (cars) and it has both qualitative and quantitative variables of high quality and usefulness.

The qualitative variables are: name of the manufacturer, model, transmission, color, engine fuel, whether the car is gas or not, type of engine, body type, whether it has a warranty, and the state of the car (owned or not). On the other hand, the quantitative variables are: odometer value, year produced, engine capacity, price (in USD), number of photos, the days it has been in use and the up counter.

Before starting to work on the database, some questions and objectives have been established, with their possible and corresponding machine learning techniques. These are the main methods we will use and the questions we will try to answer:

1. PCA: Can we reduce the dimensionality of our data?
2. KNN: What mileage should I expect in the car I am buying??
3. Recommender System: What could be the price of my next car?
4. K-Means: Is it possible group the data without labels?

Before working on the database, those values that are not useful or easy to work with have been cleaned. For instance all the variables that did not contribute anything to any of the objectives planned for the project have been deleted from the original dataframe. Once we realized that there were some missing values (NaN) in the updated data and checked that there were an insignificant amount of them; we filled those using the mean of the corresponding variables. That´s how we left our data clean and ready to work on it.

II. PCA: CAN WE REDUCE THE DIMENSIONALITY OF OUR DATA?

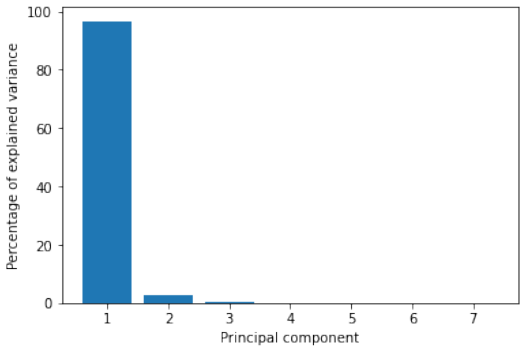
Short answer: Yes. As you imagine this is great. If we are capable of projecting the same conclusions with a simplified version of the data, this will make our young data scientist lives much easier. This is exactly what the principal component analysis does, it simplifies the complexity in high-dimensional data while retaining trends and bringing out strong patterns in a dataset.

How did we manage to this, you will ask. Ok, so we started our adventure through dimensions selecting the variables that we would use, in this case only the numerical variables, so we had 7 features or dimensions. We continued by normalizing and scaling our data, in order to better manage it. Maybe this might look as a non-very-fashionable way to start, but is really important when it comes to statistical methods.

Then we created a PCA object with sklearn, those can be trained in one dataset and applied to other, and that´s exactly what we would need just a in a moment, because we would apply this pca transformation to the scaled data.

After plotting a barplot that showed the percentage of explained variance that each principal component represented. In this case almost all the variation is along the first principal component but using PC1 and PC2, it should do a even nicer representation of the original data.

Finally we checked if everything was working right and there it was.



The 99.55 percent of our data could be explained just by two principal components instead of the seven dimensions that we started with. Amazing, don't you think?

III. KNN: What mileage should I expect in the car I am buying?

This question is answered by the prediction of the amount of kilometers that a car may have traveled, which we thought that might be calculated in a satisfactory way by the use of the K-nearest neighbors algorithm.

In order to develop this algorithm, we needed several features, which would help in the prediction. In fact, the potential client would let us know, for instance; the year of fabrication of the car, the price, the engine capacity and the colour, among many others, in this case we used 17.

We took from our dataset a sample of cars that have all these features, odometer value (mileage) included, in this case we chose the cars for our sample randomly from our database, due to the bias avoidance that this method offer.

Once the set of cars prepared, its features and our objective clear, we segmented the mileage in 3 sections: low, medium and high. Here we had two options:

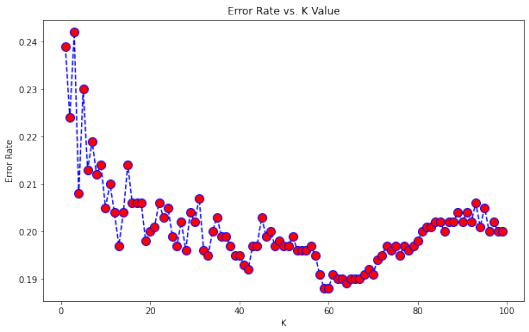
1. divide by quantiles, so in each section we´ll have the same amount of cars, sections are: (0 - 190000.0], (190000.0 - 300000.0], (300000.0 - 1000000.0] (Once seen the results we realized that for the KNN using quantiles to correctly perform we needed a much bigger amount of cars of our dataset, actually 25000).
2. use the linspace function, which divides the data by the amount, in this case, there would be much more cars in low section than in high sections, due to the fact that there are a few cars with for instance 1M kilometers but thousands with less than 200k. sections are: (0 - 333333.333], (333333.333 - 666666.667], (666666.667 - 1000000] (in this case, using linspace, 500 cars were more than enough to get a good accuracy)

We decided to use both in a parallel way and check the results later.

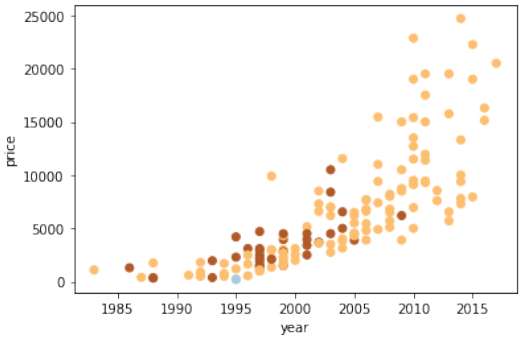
KNN works by finding the distances between a datapoint and all the other examples in the data, for then selecting the K closest examples to the objective datapoint, assuming that the value that appears the most will also fit our objective datapoint. In this case, as we have 17 dimensions, the size of distances might be more accurate, as it will find the real similar datapoints, this will normally make our results better.

After entering the car characteristics in our KNN algorithm, it should return a range of mileage in which the car will lay, with a high accuracy.

If we go step by step; we splitted up our data into train and test sets and standarized it. Subsequently we choose our K via an error rate vs k value table for then creating our KNeighbors classifier with the train data and the chosen "k".



Finally we made a prediction with the test data and visualized the classification report, hopefully with a nice result.



When it came the time to compare the results of each KNN, we realized that the one that used linspace function to create the mileage ranges performed much better and actually was computationally cheaper. This is due to the fact that it only needed 500 random cars of our dataset to give its better results (around 85 percent of accuracy). Meanwhile, the KNN using the quantiles method to create the ranges needed tens of thousands of cars, which is computationally devastating, and worked much worse (around 75 percent of accuracy). So finally we would stay with the "linspace KNN". Nevertheless it has been great to do both in order to compare and value how amazingly can a algorithm perform against another just with little changes.

For you to understand what this results mean, in the case that the car buyer wanted a standard car in all its features, the "linspace knn"will correctly predict the range of the mileage 85 percent of the times while "quantile knn"would only do it correctly 75 times in every 100.

IV. Recommender System: What could be the price of my next car?

How can we predict the price of one of a car taking into account a series of characteristics? That was the main question that appeared in our minds when approaching the idea of developing a recommender system algorithm.

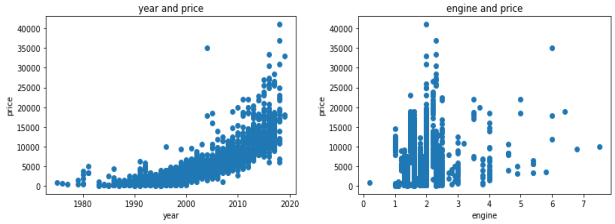
We have worked on this objective with pandas and numpy. The idea is that given a row with a 0 or a missing value (NaN)

in the price variable, thanks to the correlation with the other cars, an approximate price can be given taking into account count the features it has.

In order to make the recommender system, different points have to be taken into account. The first of all is to have all the data clean and without any type of error or missing value, then you have to make sure that all the rows that will be studied have the same number and the same variables as the car you want to recommend a price. And last but not least, the data of the same variable, or in this case of the entire dataframe, must be of the same type (int, float, object...).

In this part of the project, we have chosen only a part of the dataframe ,only Audi cars, although it has been studied with other brands to compare the accuracy of the recommender system, but but we observed that this is the most accurate. Regarding the variables that have been used, these have all been quantitative and are the year of production, the odometer value, the engine capacity, and the duration of the car.

To ensure that the variables were appropriate, we have done a series of plots to see the correlation that exists between that column, and the variable of the price. For this, the matplotlib.pyplot library has been used and we have computed the correlations with different variables. Once this is done, in the following graphs you can see the difference between the correlation of the year with its price, and the correlation of the engine capacity with also its price.



In the first graphic a high correlation is seen because the more recently that car has been produced, the higher price is. However, in the second graph we can state that for a same engine capacity value, a large number of different prices coexist and this range of prices is similar for other cars with a different engine capacity.

Once the data is ready, we can start working on it. First, it has been decided to follow a user-based recommender system since, as our data is very different from each other, the item-based recommender system would not work well (this type of recommender is used more in situations where the values between the variables are similar, for instance, scores).

To achieve our goal, we have used the similarity between the rows with the car whose price we want to recommend and this has been carried out with the corrwith() function. Then, we have selected the 3 cars with the highest correlation and compute the average.

As mentioned before, this same algorithm has been used for more than one car brand, such as Ford and Opel, but in the original database there were not as many rows of these cars and when executing the program it was clear that the recommender system was less accurate and has therefore been discarded. On the other hand, we wanted to check if the variables that at first seem not to have much correlation with the price, which is at the end our objective, really affected the result of the recommended price. After adding two extra variables to the code (number of photos and up counter), we have noticed that the results have been different than we expected since once these variables were added, the price was even more adjusted to reality.

V. K-Means: Is it possible group the data without labels?

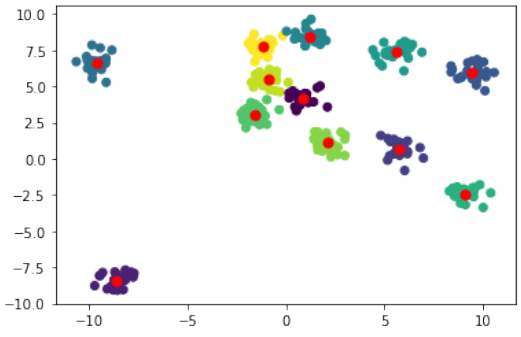
For this last data science method the main objective resided in checking if we were able to develop an algorithm that would unsupervisedly cluster our data in the 12 different body type of cars but without giving to the algorithm the specific label, that is: Make 12 group of cars that are the most similar to the 12 groups that the 12 different body types create without any given label.

In order to perform this task we decided that the best option was to use the well-known K-Means algorithm.

Actually, K-Means is an algorithm that tries to split the dataset into K pre-defined different non-overlapping clusters where each data point belongs to only one group. It tries to make the data points in one same cluster as similar as possible while also keeping the different clusters as far as possible from each other. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid is the minimum possible. The centroids are the arithmetic means of all the data points that belong to a specific cluster). The less variation we have within clusters, the more similar the data points are within the same cluster.

Our K-Means approach was based in the following set of steps: First of all I set all the features as quantitative variables, then we set the number of clusters of the K-Means to twelve due to the fact that my aim was to compare it with the car body types ground truthful clusters.

After that we fitted the data to the model and got the centroids of the clusters. Following this step we plotted the prediction and there it was, an almost perfectly separated set of 12 clusters that once checked and compared to the ground truth clusters turned out to perform with a 87 percent of accuracy compared to the real clusters of the car´s body types.



VI. CONCLUSIONS

As a summary of results of the goals and objectives, we can proudly state that we successfully managed to realize with a good ending all the required techniques and methods.

1. For the first objetive we derive that the PCA is so useful when it comes to huge datasets because it simplifies the exceeding dimensions that those datasets may have.

In our case we were able to derive almost the same amount of information just with 2 principal components instead of the seven numerical features that we had.

2. Looking towards the second objective, we performed a KNN that bases itself in predicting a value by the closeness to other k similar points.

We specifically wanted to predict the mileage so we developed the algorithm and understood how it worked, as a result we got that the algorithm predicted the amount of kilometers a car had traveled with a sufficient accuracy.

3. For the third objective, we can conclude that previously understanding the data is essential to achieve a good algorithm. In addition, we have been able to verify that we cannot keep the first recommender system that we obtain, since we may be overlooking a lot of information that makes our program much more precise, as it was the case of the photos variable.

4. Finally for the fourth and last objective, a K-Means algorithm was required. When it comes to analyze the results, we conclude that comparing to the main ground truth clusters that we had, based on the car body types, a nice accuracy was obtained. This is due to the fact that the predicted clusters were equivalent in 87 percent of the situations.

THOUGHTS ON THE EXPERIENCE

After finalizing this project we can conclude that we have developed a series of vital skills of the field, while we have had a great time learning and understanding all its processes.

We can also affirm that as a practical example was used, it was projected in us a much deeper interest, which we think that has been actually well reflected in the final result.

The methods used were so useful, as well as interesting. Getting informed, searching and reading took a long time, but even though this could sound boring, it was not boring at all. Even better, it led us to apply all the required algorithms through the project with a profound understanding of the processes. Also provided us with knowledge of some of the leading cutting edge methods used today in the state of the art of data science, machine learning and even, why not, car companies.

This work changed a lot our vision towards data. We found a gold mine where we could only see rocks previously; data can look tough sometimes but if you dig into it, it reveals reality as anything else can.

ACKNOWLEDGEMENTS

We are so grateful to all those who accompanied us in the development of this adventure. Actually we can state that this project was great, sometimes frustrating but always amazing, thanks, in part, to every person that helped us, every single person that dedicated a few minutes (or hours) to our project. Working as a team, with cool colleagues around is an unbelievable experience that we undoubtedly want to keep repeating.

As an honour mention, we would like to thank our guide and sensei through all the process, Javier Vazquez, our data engineering teacher. He was always there when we needed him, there are no words to thank his effort, so at least, we would love to dedicate this whole project to him. Sincerely thanks for being a great teacher.

REFERENCES

[1] datacamp.com/tutorial/principal-component-analysis-in-python
[2] towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
[3] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[4] towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a
[5] realpython.com/build-recommendation-engine-collaborative-filtering/
[6] adamsmith.haus/python/answers/how-to-convert-a-pandas-dataframe-column-from-object-to-int-in-python
[7] https://towardsdatascience.com/linear-regression-in-6-lines-of-python-5e1d0cd05b8d
[8] https://datascientest.com/es/que-es-el-algoritmo-knn
[9] https://www.adamsmith.haus/python/answers/how-to-convert-a-pandas-dataframe-column-from-object-to-int-in-python
[10] Cintia Ganesha Putri, D., Leu, J. S., Seda, P. (2020). Design of an unsupervised machine learning-based movie recommender system. Symmetry, 12(2), 185.
[11] Ramzan, B., Bajwa, I. S., Jamil, N., Amin, R. U., Ramzan, S., Mirza, F., Sarwar, N. (2019). An intelligent data analysis for recommendation systems using machine learning. Scientific Programming, 2019.