Udacity Machine Learning Nanodegree Capstone Project

Hoang Duy Trinh

Table of Contents

Human Resource Analytics

1 Problem Definition

1.1 Domain Background

In any company, Human Resources (HR) have the task of regulating, monitoring and analyzing the complex interactions that occur between the employees and the company itself. Employee retention is of significant importance to companies. However, due to several reasons, the company may need to replace some of the employees. This would make the company to incur in high costs due to many parts, including hiring new people, and also in side effects, for example, lowering the morale of the other employees.

Employee retention is therefore a delicate matter that needs to be addressed considering a wide range of factors: studies show an immediate reduction in the productivity, and major efforts are usually required to keep pace with economic constraints (for example demand/supply management etc.).

Due to these reasons, a data-driven approach is the most significant in order to understand the complex relationships that take place in a work environment and it can help to alleviate the surplus of management costs. Similar to this, multiple efforts have been done in machine learning to cope with churn problems. In [1], the report provides a case study for customer churn linked to a well-known online payment platform. The dataset used in this project is the Human Resource Analytics from Kaggle competition [2].

1.2 Problem Statement

We want to analyze the characteristics of the employee turnover, and understand how a company could minimize the negative effects. To this end, the solution should include:

1. A detailed characterization with deep insights in the data, studying first the importance of each features, their correlations and their statistics

2. A possible classification of the employees, including clustering methods

3. An anticipatory model that could predict a possible employee leave

1.3 Methodology Overview

In order to address the presented problems, our project will include the fundamental parts of a data-driven problem solving:

- we will start from a data preprocessing step, where missing features and categorical components of the dataset will be treated; then, we will perform some EDA, where we will adopt useful visualizations to understand the features and their correlations;

- we will start our modeling and analysis: first a clustering method, can help to individuate sub-groups of employees and make us understand how to separate the employee classes;

- finally, we will study which type of model will fit best our goal: to do this, a parameters validation and search will be implemented to maximize the evaluation metrics. A comparison between different models will be given and new models will be explored: to this end, we can rely on different implementation of the gradient boosting algorithm, including *Xgboost* or *LightGBM*.

1.4 Evaluation Metrics

In this problem, *accuracy* would be a good metrics to evaluate and compare the presented models.

$$accuracy = \frac{num. of\ correct\ predictions}{total\ number\ of\ employees}$$

However, due to the unbalance class distribution it is better to consider multiple metrics, including *recall, precision* and *F1 score*. In particular, the recall of leavers, will be important to evaluate the performance of the algorithms:

$$recall = \frac{tp}{tp+fn} \quad precision = \frac{tp}{tp+fp} \quad F_1 = \frac{2RP}{R+P}$$

where *tp*, *fp* and *fn* are true positives, false positives and false negatives, respectively, referred to the class of leavers. The F1-score is defined also as the harmonic mean of the recall and the precision, and it is needed to evaluate the tradeoff of the classifier between precision and recall.

2 Exploratory Data Analysis (EDA)

2.1 Datasets and Inputs

The dataset used in this project is the Human Resource Analytics from Kaggle [2] and consists of nearly 15000 samples, each containing 9 features corresponding to one employee. The dataset is labelled under the column **left** (=1 if the employee left, =0 otherwise), the other columns are the following:

1. **satisfaction_level:** employee satisfaction level, num in [0,1]
2. **last_evaluation:** last employee evaluation, num in [0,1]
3. **number_project:** number of projects, num
4. **average_monthly_hours:** avg number of working hours, num
5. **time_spend_company:** years spent at the company, num
6. **work_accident:** whether the employee was involved in a work accident, boolean (0,1)
7. **promotion_last_5years:** whether the employee was promoted in the last 5 years, boolean (0,1)
8. **sales:** which department the employee worked in, text
9. **salary: l**abel of low, medium or high, text

The class distribution is *unbalanced*, meaning that one class is much more represented in the dataset. In particular, the employees with left = 0 are more present than the employees who left. Approximately, the distribution is (75-25%), therefore, insightful metrics need to be chosen to take care of this fact (e.g. accuracy only may not be sufficient alone and may lead to the accuracy paradox).

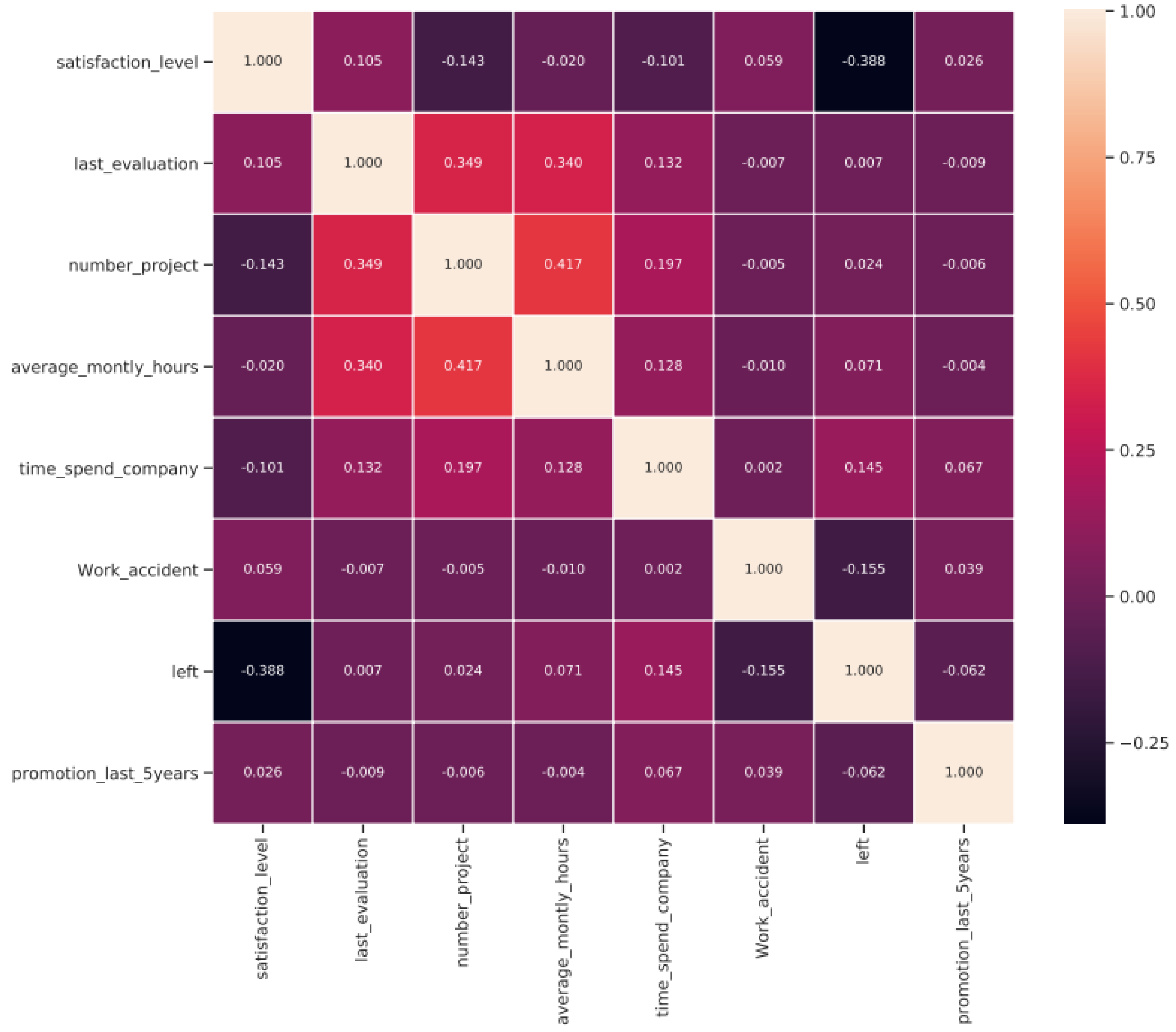
2.2 Data Visualizations

As first step of our analysis, it is fundamental to look after the hidden patterns and the cross-relationships that can exist among the features in the dataset. This is valuable in the next phases of features selection and may help to tweak the classifier performance.

2.2.1 Correlation Matrix

In general, it is useful to understand if some features are highly correlated: a high correlation (e.g. > 0.8) can imply a large linear dependence, that would allow us to eliminate some features and reduce the input dataset.

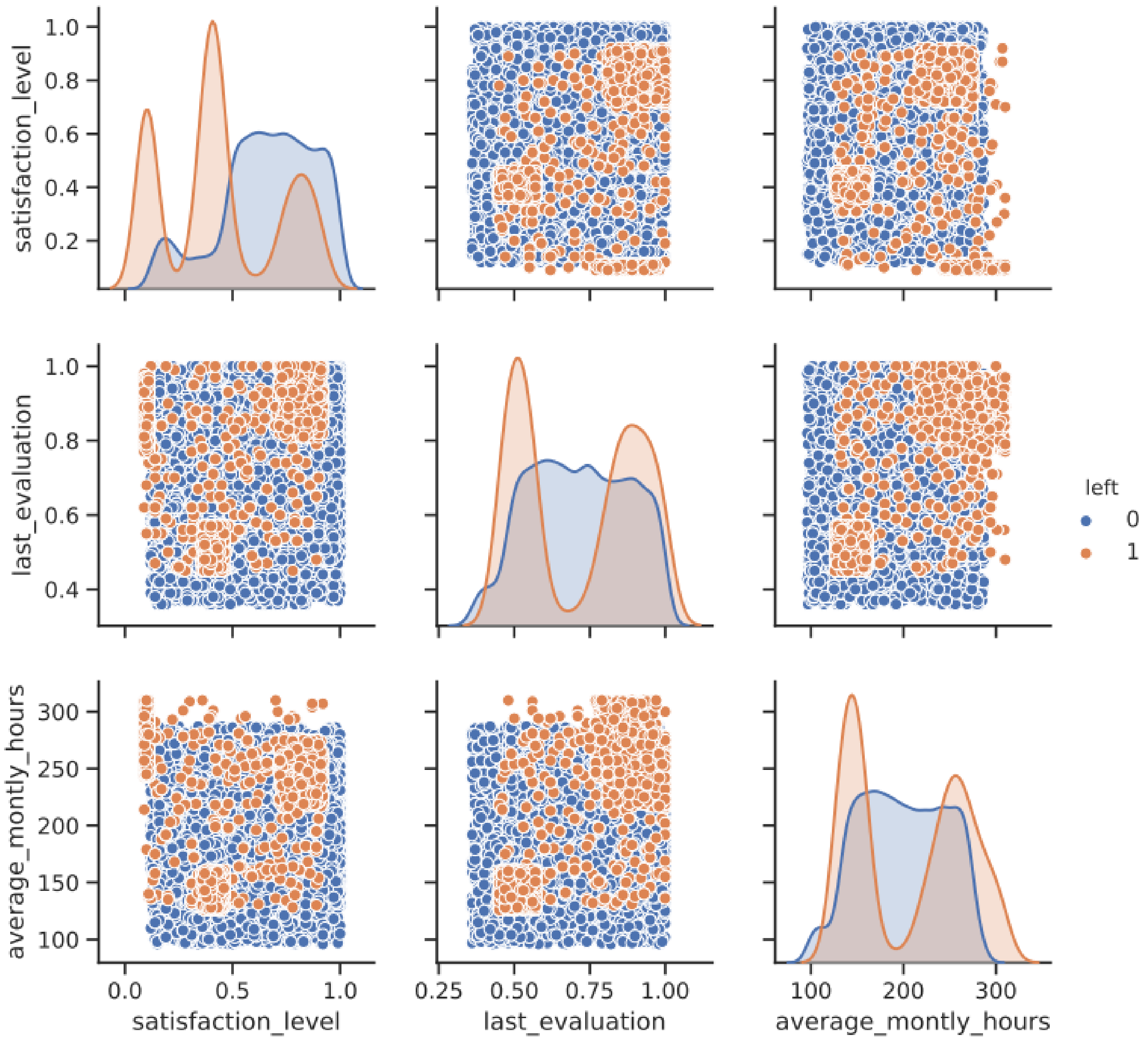
However, in our case, the correlation matrix does not show high correlations: the highest values are between the *average_monthly_hours* and *number_project* features (0.42), the *last_evaluation* and *number_project* (0.35) and the *average_monthly_hours* and *last_evaluation* (0.34).



2.2.2 Continuous Features PairPlot

Additionally, in order to find pairwise features relationships, we plot the scatter plots of the continuous features from the dataset. In the plot below, we can see that only 3 variables are continuous (the others are binary or categorical).

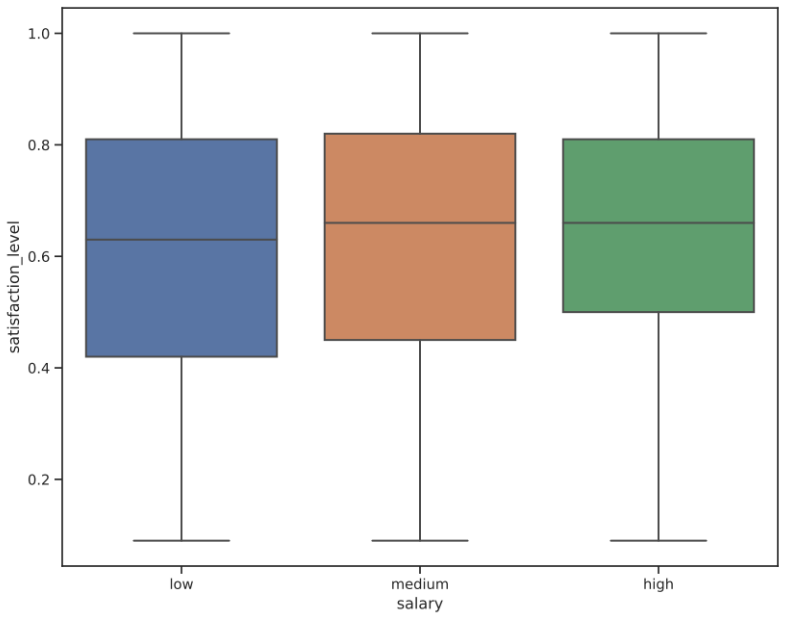
One thing that should be noticed, is that separating the two classes of employees (left = 0/1), we can different distributions of the continuous variables (shown on the diagonal). However, the "peaks" in the distributions for left = 1 are also due to class unbalance, with much fewer samples for the leaver class.

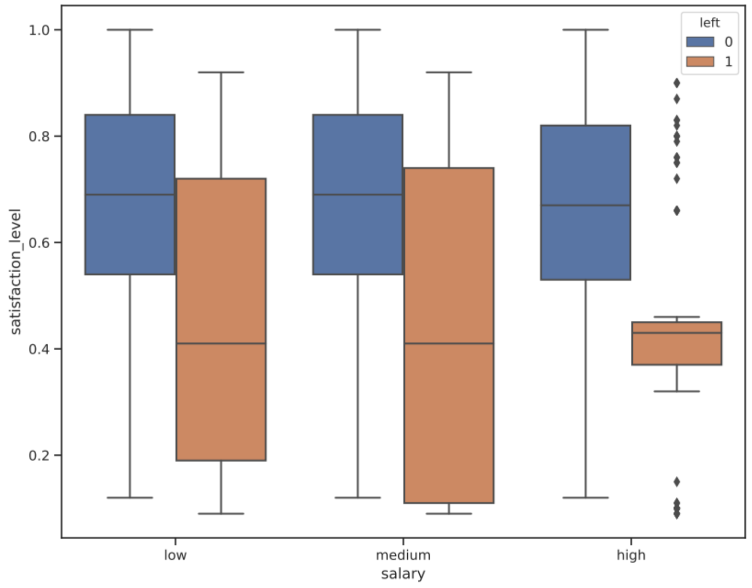


2.2.3 Salary vs. Satisfaction Level

We can try to find other patterns, using the box plot between the pair salary – satisfaction_level features. The interval of the box plots varies from 0 to 1 in any salary case (low, medium, high). However, as expected, the box dimensions are smaller and centered on higher satisfaction_level values as the salary increases.

A more interesting plot can be obtained if we breakdown the class of leavers from the employees who remained: here, the difference is more observable, showing that who left also had a lower satisfaction level.

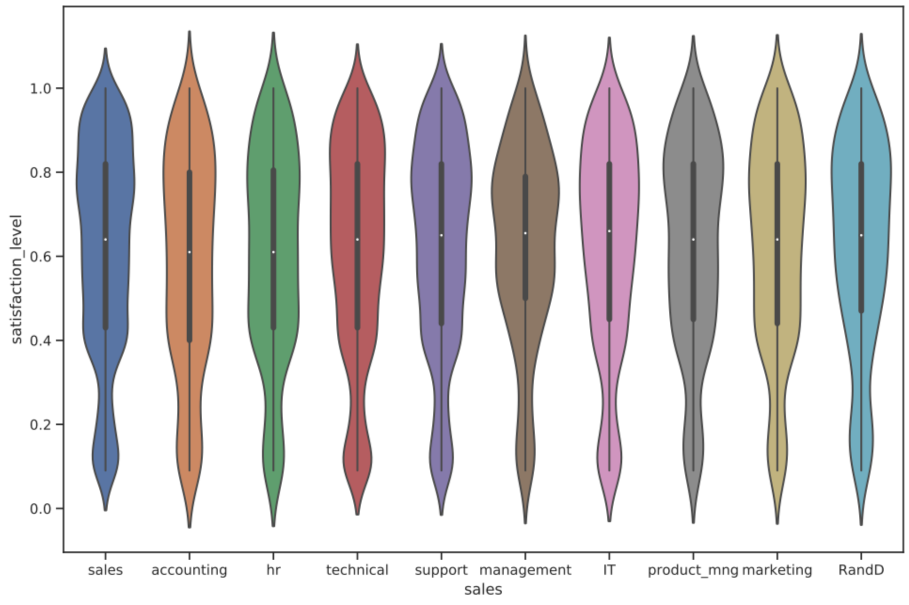


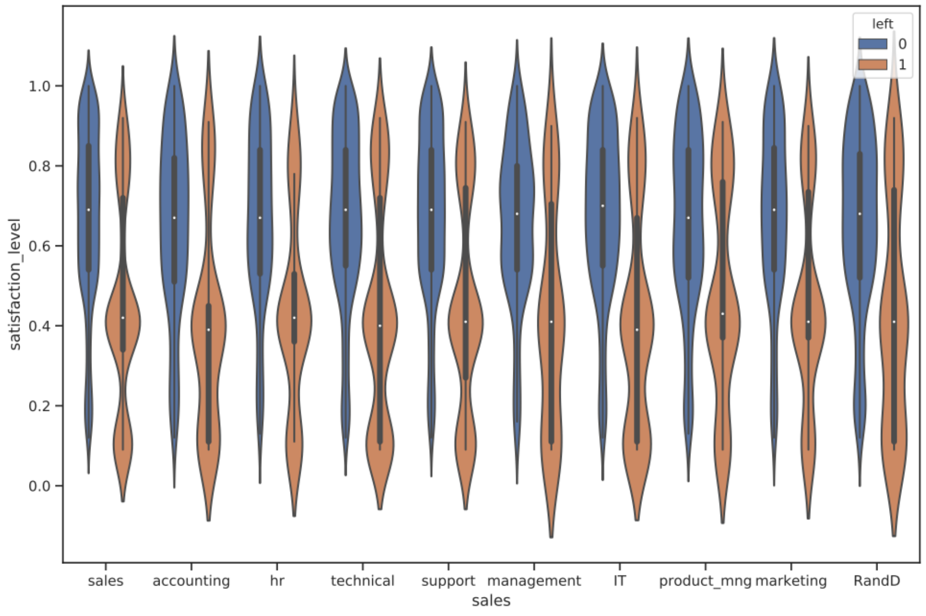


2.2.4 Department vs Satisfaction Level

Similar to the previous plots, we can attempt to find further insights in the satisfaction_level of the employees. This time we breakdown the **sales** (or department) feature, using violin plots, to observe if there is any particular behaviour or some relationships between the satisfaction level in the different company department.

However, even distinguishing the two employee classes we cannot observe noticeable differences among the department.



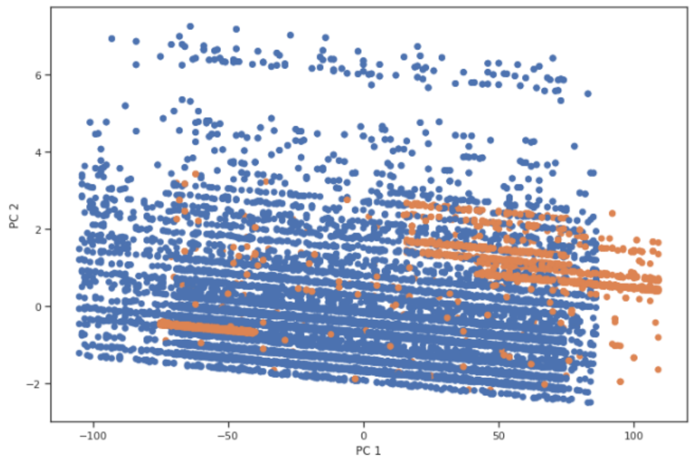


3 Principal Components and Clustering Analysis

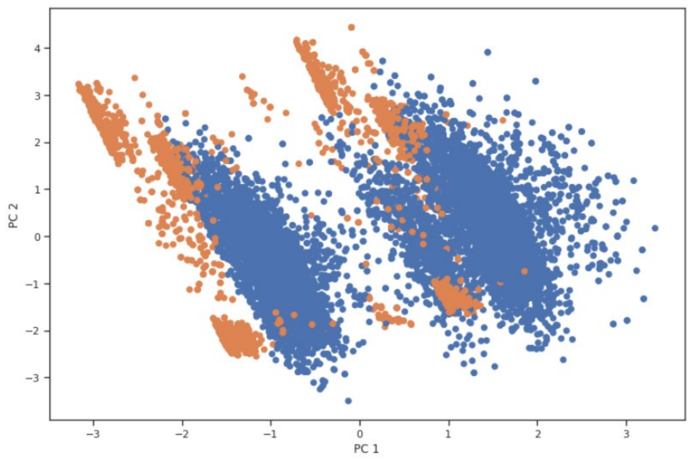
3.1 Principal Component Analysis

PCA is a linear transformation that can be used to explain the variance of the features in the dataset. It can be used as features selection/reduction method, and also to obtain 2D-visualizations of high dimensional dataset.

To perform PCA it is important that data is normalized: we use a Standard Scaler from sklearn, and we obtain a scatter plot using the 2 first Principal Components (the two that contain most of the variance). Below, we show the importance of the normalization step. In the second figure, we can notice 4 or 5 more separated groups of samples, however, using different colors for each class of employees, we can see that PCA is not sufficient alone to separate the leavers from those who remained.



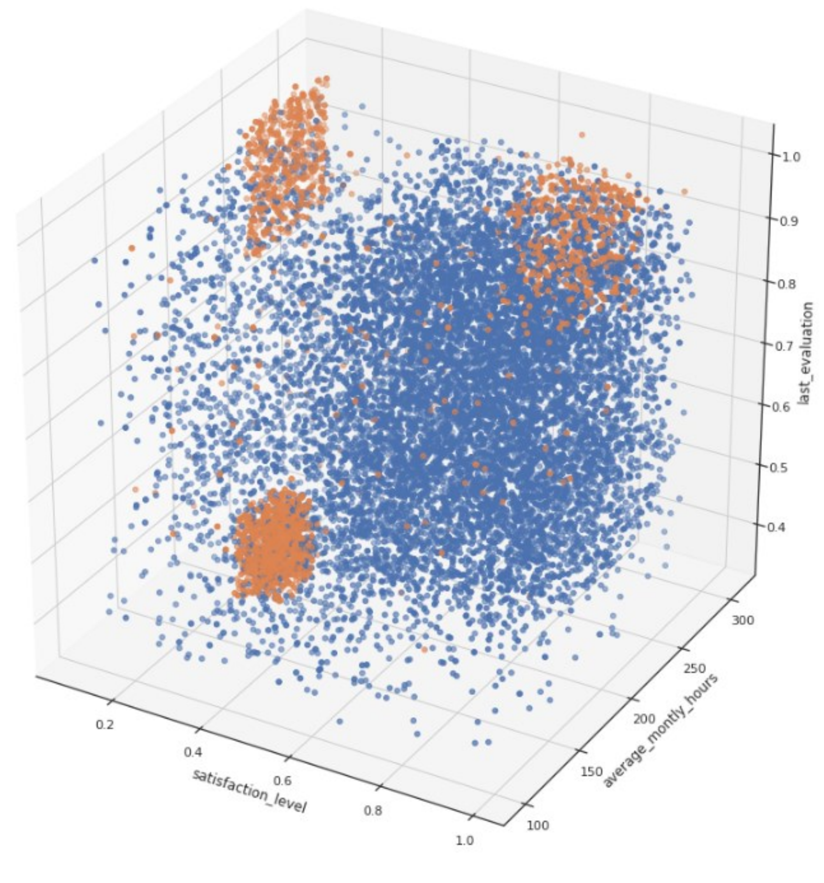
PCA without normalization



PCA with normalization

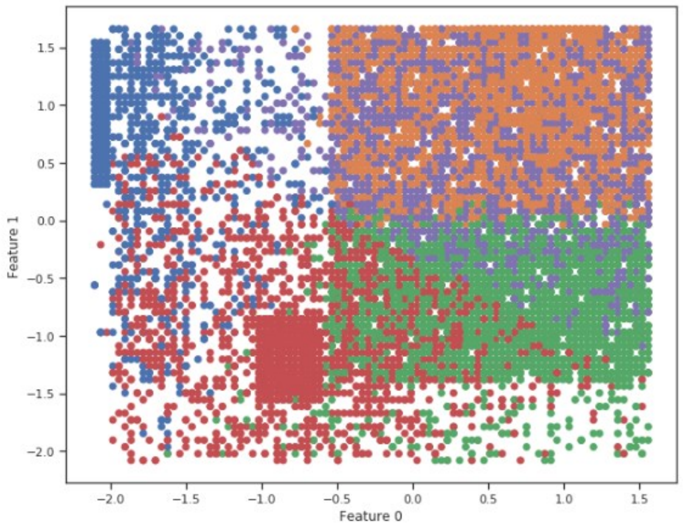
3.2 Clustering

As first modeling attempt, we can try to separate the two employee classes *unsupervisedly:* first, we plot the 3D-scatter plot using the three continuous features. We can observe already that the leavers samples are mostly concentrated in 3 different areas of the 3D plots. This motivates us to continue the clustering analysis.

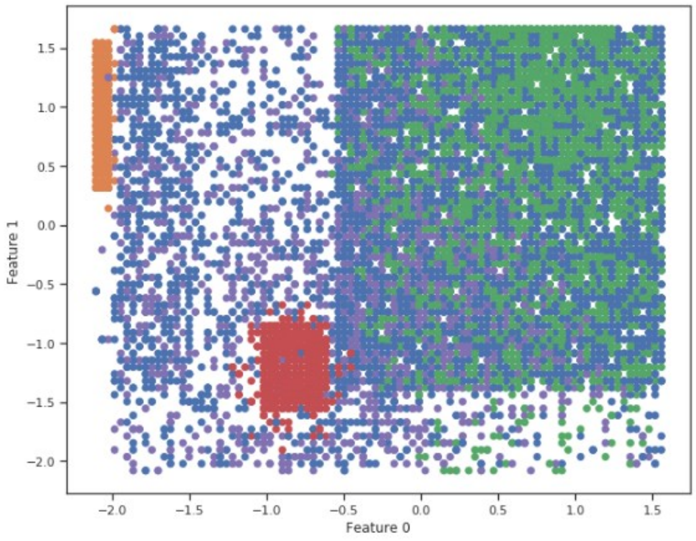


We use two different clustering methods on our dataset: **k-means** and **Gaussian Mixture**. The two plots below show the results of the clustering methods. However, they are not very informative. What it is important is the composition of each cluster, in percentage of leavers in each cluster, which is reported in table below.

We can observe first that k-means is too simple, and second, that in 2 clusters, the Gaussian Mixture is able to isolate the leavers with high probability (93% and 98%). This is a good result considering that we apply this method unsupervisedly, and it may help later to improve the model for the employee leaving prediction.



k-means clustering



Gaussian Mixture clustering

k-means

cluster	total size	% leavers
1	3160	2.40
2	3551	44.47
3	1611	58.41
4	3449	26.44
5	3228	1.98

Gaussian Mixture

cluster	total size	% leavers
1	1639	92.62
2	4332	21.93
3	5199	2.21
4	2935	3.54
5	894	98.88

4 Modeling and Prediction

4.1 Data Preprocessing

The dataset is complete, meaning that we don't have to deal with NaN, missing values, etc. However, there are Boolean features (*work_accident, promotion_last_5years*) and Categorical Features (*sales, salary*). In both cases, we applied the one-hot encoding method.

In the clustering analysis, we saw that using Gaussian Mixture, we are able to form some clusters that contains almost only leaver samples. One pre-processing step could be to append this information to the dataset, before being input to the predictive model. Next, the dataset is split into training and test set with a split-ratio of 0.3.

4.2 Benchmark Model

A comprehensive analysis can be found from the Kaggle kernel in [3], which includes investigation of the variables, their correlations. Also good insights and visualizations are given to determine which are the main factors that correlate the satisfaction level and the employee retention.

A set of models is considered in a logistic regression problem, to predict the employee retention. By using accuracy as one of evaluation metrics, we are able to directly compare the proposed predictive model against this one, In the provided kernel the maximum achieved accuracy is 95%: our objective is to implement a predictive model that at least achieves this score.

4.3 Algorithms

In our analysis, we choose a set of algorithms that serve as benchmark for the performance comparison. In particular, we choose three algorithms, namely **Logistic Regression**, **Decision Tree** and **Random Forest**. The implementation of these algorithms can be found in the popular *sklearn* library.

Also, we investigate new models: we focus on the **Gradient Boost Algorithm**, which is gradient-descent based algorithm that recently gained a lot of popularity among Kagglers and ML enthusiasts, due to its efficient implementation. In this project, we will use the implementation from the ***xgboost*** library [4].

5 Results

5.1 Numerical Results

First, we train and we test the chosen algorithms, without taking care of any parameter or optimization step. We use the *sklearn* implementations and the default parameters to understand approximately which type of model fits best in this classification problem. We don't perform any ulterior preprocessing on the data (except the one-hot encoding of categorical features).

Below we report the results: we can see that *SVM* and *Logistic Regression* perform bad compared to the other algorithms. Both are linear classifiers (precisely Logistic Regression is log-linear), and both seems to fail (or underperform) in the classification task.

The best results are achieved using Tree-based classifier with accuracies higher than 97%,

	Accuracy	*Recall*	*Precision*	*F1-score*
Logistic Regression	0.84	0.69	0.63	0.66
SVM	0.96	0.91	0.90	0.91
Decision Tree	0.97	0.93	0.96	0.95
Random Forest	0.98	0.98	0.94	0.96
XGB	0.99	0.99	0.96	0.98

5.2 Model Refinement

We choose the gradient boosting algorithm, which performed the best in the classification, and we try to optimize its performance. As shown in [5], the XGBClassifier is characterized by a set of input parameters including the three following:

- **max_depth** (*int*) – Maximum tree depth for base learners.
- **learning_rate** (*float*) – Boosting learning rate (xgb's "eta")
- **n_estimators** (*int*) – Number of trees to fit.

We perform an exhaustive cross-validation search (using sklearn **GridSearchCV**), to obtain the parameters that best predict the employee retention. To evaluate the different model configuration, we choose the F-score metric of the leavers since accuracy can lead to undesired results due to the strong class unbalance

We report the results, highlighting that we have a perfect F-score of 1 on the training dataset and a F-score of 0.98 in the test set. The accuracy in both cases is close to 99%, which is higher than the benchmark model.

We report also some statistics of the model refinements procedure, including the mean fit time and its standard deviation over the 5 cross-validation folds that we performed to find the best parameters. We report the results as it is output by the provided code.

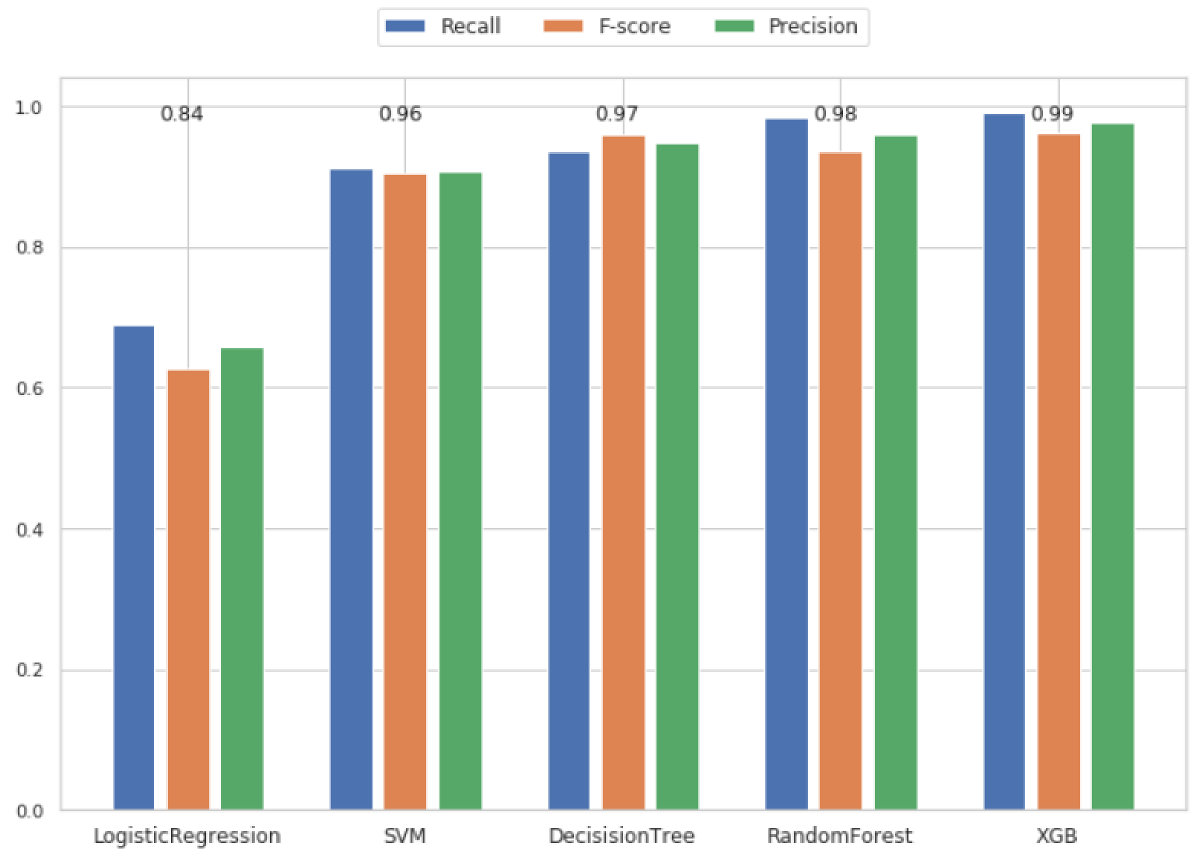
```
Best model results on training set
Accuracy: 99.98%
Recall of leavers: 1.00
Precision of leavers: 1.00
F1-score of leavers: 1.00

Best model results on test set
Accuracy: 98.84%
Recall of leavers: 0.96
Precision of leavers: 0.99
F1-score of leavers: 0.98

Full results
mean_fit_time                                                3.70931
std_fit_time                                                0.206786
mean_score_time                                            0.0453256
std_score_time                                             0.0140487
param_learning_rate                                              0.1
param_max_depth                                                   10
param_n_estimators                                               200
split0_test_score                                          0.976507
split1_test_score                                             0.9766
split2_test_score                                            0.97472
split3_test_score                                             0.9716
split4_test_score                                            0.97472
mean_test_score                                              0.97485
std_test_score                                             0.0017996
split0_train_score                                            0.9997
split1_train_score                                            0.9997
split2_train_score                                             0.999
split3_train_score                                             0.999
split4_train_score                                            0.9997
mean_train_score                                              0.9996
std_train_score                                           0.00012256
```

5.3 Performance Comparison

For the sake of completeness, we report a visual representation of the performance of the prediction models utilized in this project. The plot below contains all the considered metrics: recall, precision, F1 score, and accuracy (written above the bars in black):



6 Conclusions and Discussion

In this project, we analyzed a dataset containing information of 15000 employees and we study the possibility of employee leaving (or retention). We start trying to find possible *cross-correlation* between the features and we attempted to reduce the initial dataset looking at different pairwise plots.

Next, we apply a linear transformation, the *Principal Component Analysis*: here the results are not satisfying, however we can see that we can differentiate at least 4 clusters in the dataset. We use this information and we apply 2 different unsupervised clustering methods*: k-means* doesn't perform well. However, using *Gaussian Mixture*, we are able to obtain clusters that nearly contains samples only from the leaver class. Using this information, we can improve the prediction: in fact, we append the clustering results to the dataset before being input to the predictive model.

We test a set of different prediction methods: the best results are obtained using tree-based algorithms. Precisely, with respect to other solutions including our benchmark model, we use the *gradient-boosting* algorithm implemented with the *xgboost* library. Optimizing this model on the F1 score through a cross-validation *parameter search*, we achieved an accuracy of *100%* on the training set and *99%* on the test set, which is above the *95%* accuracy of the benchmark model.

Given the initial dataset, we understand that in this problem linear models are not able to achieve high results, and *tree-based* algorithms are the best option. We can acknowledge this also from the 3D plot: the fact that the employees who left are in separable and distinguishable areas, can make us intuitively understand that tree-based algorithms can perform better. Also, we acknowledge one more time that *xgboost* is a really reliable implementation of the gradient boosting algorithm and this justifies its great success among the ML area.

References

[1] https://www.h2o.ai/wp-content/uploads/2019/02/Case-Studies_PayPal.pdf

[2] https://www.kaggle.com/jacksonchou/hr-data-for-analytics

[3] https://www.kaggle.com/jacksonchou/hr-analytics

[4] https://xgboost.readthedocs.io/en/latest/python/python_api.html