IE School of Human Sciences & Technology

MSc in Big Data and Business Analytics

Machine Learning II

Professor: Jesus Salvador Renero Quintero

Individual Assignment: Feature Engineering

Juan Pablo García Gómez

MBD – O1

Background:

The following document presents the Feature Engineering process applied to the experimental dataset *HR Analytics.* This dataset contains information about employees of a large company regarding their satisfaction level, time in the company, performance, and salary among others. The main objective is to understand the probability of attrition (employee leaving the company) and recognize which are the most important variables to be addressed to prevent attrition to happen. A Logistic Regression Model will be applied to understand the impact of the feature engineering and a linear Model will help to understand the impact of the different variables in the attrition dependent variable. All the process will be developed in Python.

Data Understanding – Exploratory Data Analysis

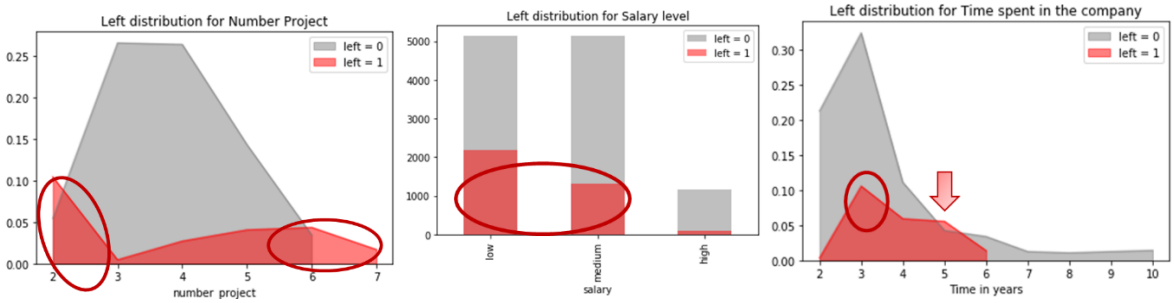
The first step prior to running the model or performing feature engineering, is loading and understanding the raw dataset. As can be seen in the annexes # 1,2 and 3, the raw data consists of 10 variables that don't present missing values, with 14,999 observations. All the variables are numeric except *Sales* and *Salary*, and our Target Variable is entitled as "*left*". We can also see that *sales* and *salary* are categorical variables with 10 and 3 different categories respectively, and that the other numerical variables shall be treated as Boolean variables except *satisfaction_level* (continuous 0 to 1), *last_evaluation* (continuous 0 to 1), *number_project* (int), *average_montly_hours* (int), and *time_spend_company* (int).

To better understand the data, in addition to the descriptive analysis found in Tables 2 and 3 of the annex, in annex # 4 we find Histograms and Box plots for the variables. Variables such as *satisfaction_level*, *last_evaluation*, *number_projects* and *time_spend_company* seem to be skewed and *time_spend_company* shows outliers in the box plot. We can also analyze that the variables show different scales, and this matter will be treated in a subsequent phase.

In the image at the right we can see the distribution of the binary variables. The first impression that we get is that *Work_accident* is heavily skewed to 0 (no accident: 98% of observations), indicating that this may not be a relevant variable for the model. Work_accident also shows a big percentage to 0. However, as we will see, this variable may have an important incident in our dependent variable. We can also see that the variable we want to predict, *left*, is balanced towards zero (nor leaving 76%).

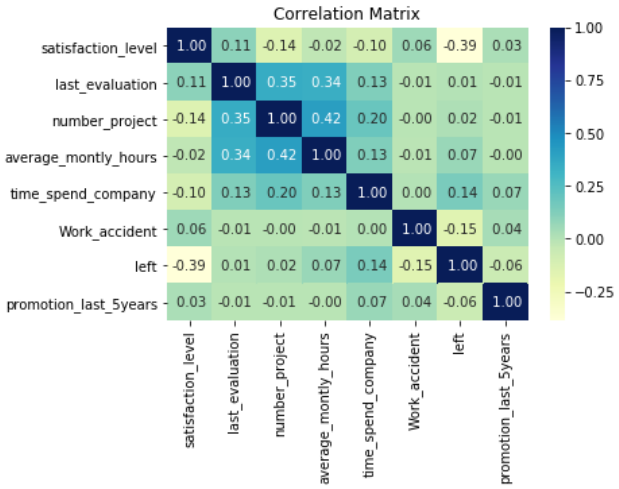
```
0    0.85539
1    0.14461
Name: Work_accident, dtype: float64
0    0.978732
1    0.021268
Name: promotion_last_5years, dtype: float64
0    0.761917
1    0.238083
Name: left, dtype: float64
```

For a better data understanding, it's important to analyze the relationship between de dependent variable and our raw features. From the following graphs we may start to figure out that the variables may explain the probability of attrition. Regarding the *number of projects* we may start to see that workers with 2 projects are more prone to leave the company (lack of involvement) and with 5 or more projects, the proportion of employments leaving vs staying is higher (overload). We may also see that there is a relationship of attrition with the *salary* level (higher probability with lower salaries), and with the *time spent in the company* that the employees with 3 years show the highest number of attrition cases (but also non-attrition). We may also see that a higher proportion of emplyees with 5 years in the company are leaving than actually staying in the company.



The following plot allows us to analyze correlation between variables. This is done after turning the *salary* variable to ordinal (low=0, medium=1, high=2). The correlation analysis is important to understand if we should drop any correlated feature and also to increase our understanding about the relationship of the variables with our dependent variable.

As we can observe, none of the correlations is high enough to drop any feature. From the correlation map, we can identify that the dependent variable is mostly correlated with: *Satisfaction level* (-0.39): The least satisfaction level, the more prone to leave from the company (quite obvious). *Work Accident* (-0.15): If the worker had a working accident, the possibility of leaving increases, *Time Spent in Company* (0.14): If the worker has spent more time working for the company, the possibility of leaving decreases.



Data Preparation

One-hot encoding: As an initial data preparation step, after we confirm that the variable types are correctly defined, the categorical variables are transformed into dummy/indicator variables for them to be considered in the logistic regression without introducing a bias by oversizing the categories. This process is done for *sales* and *salary* variables.

Under represented features: The objective of this step is rejecting those features that concentrate most of the information in the same class. In the data exploration we saw that *promotion_last_five_years* concentrates almost 98% of the data in 0. However, considering this may be a relevant variable for attrition prevention and that we are setting the deleting threshold in 99%, in this step we found out that this feature will not be eliminated.

Feature Scaling & outlier treatment: From the data description at the right, we can see that the variables *number_project*, *average_monthly_hours* and *time_spend_company* show different scales that may influence the result of the logistic regression. To run the

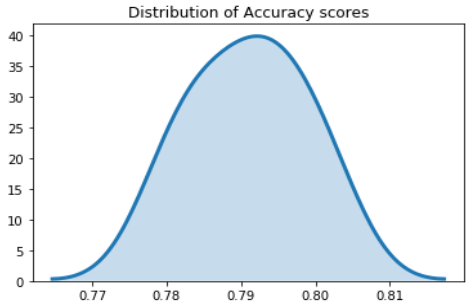
	count	mean	std	min	25%	50%	75%	max
satisfaction_level	14999.0	0.612834	0.248631	0.09	0.44	0.64	0.82	1.0
last_evaluation	14999.0	0.716102	0.171169	0.36	0.56	0.72	0.87	1.0
number_project	14999.0	3.803054	1.232592	2.00	3.00	4.00	5.00	7.0
average_montly_hours	14999.0	201.050337	49.943099	96.00	156.00	200.00	245.00	310.0
time_spend_company	14999.0	3.498233	1.460136	2.00	3.00	3.00	4.00	10.0
Work_accident	14999.0	0.144610	0.351719	0.00	0.00	0.00	0.00	1.0

baseline model, it is important to Standardize features by removing the mean and scaling to unit variance. This is performed with the *RobustScaler* function that standardizes based in percentiles, not being influenced by possible outliers.

Baseline Model

Now that the data preparation phase is done, prior to the Feature Engineering phase, is important to run the Logistic Regression model to have a benchmark in the Accuracy metric[1]. A K-fold cross validation approach is conducted to test the reliability of our baseline score with K=5:

```
Obtained 5 positive Accuracy scores
Best Validation Accuracy: 0.80
Avg. Validation Accuracy: 0.79
```

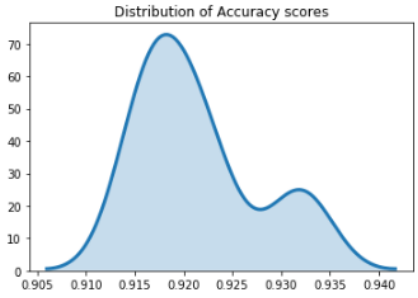


These statistics will be taken into consideration to test whether the feature engineering actually improves our model accuracy or not.

Feature Engineering

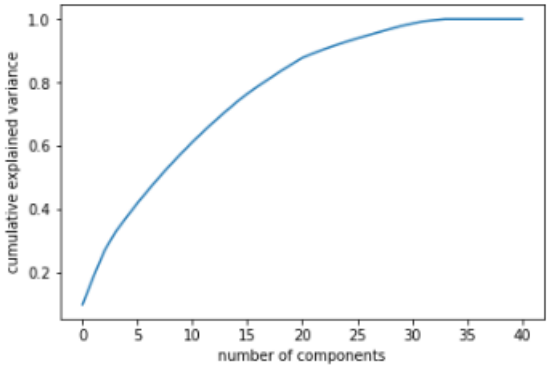
Bin Creation: As a first step in the Feature Engineering phase, the non-binary variables are binned, transformed into categorical variables, and encoded into binary variables. This binary feature creation may help maximize the accuracy level. The number of bins is created depending on the distribution of each of the variables. In annex # 6, the details of the bin creation is displayed.

This process has a positive impact in the model. In the following table, we can see that the feature engineering increased the average Accuracy score from 80 to 92, so we will keep this feature engineering process in the final model development.




```
Obtained 5 positive Accuracy scores
Best Validation Accuracy: 0.93
Avg. Validation Accuracy: 0.92
```

PCA: As a second step, now that we have a considerable amount of features, with the Principal Component Analysis I'm going to analyze if with the PCA I can reduce (aggregate) the features based in the variance they explain in the dependent variable. This process also can help in noise filtering and feature extraction. To understand the number of components the graph at the right is plotted. As we can see, around 30 components explain 90% of the variance. However, running the model after performing PCA we can see that the Accuracy score isn't improving so this feature engineering step is discarded.



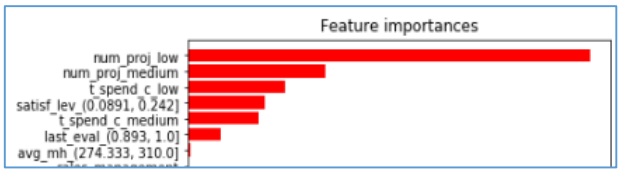

```
Obtained 5 positive Accuracy scores
Best Validation Accuracy: 0.93
Avg. Validation Accuracy: 0.92
```

[1] For this purpose, Accuracy is the metric we will use to evaluate the model. Depending of the data type, scope of project, or other components, other scoring measures may be used such as Precision, Recall, etc.

Polynomial Features: In this step, the idea is to analyze if the features may have a non-linear relationship of second degree. If the relationship is non-linear, the model will have a better fit. However, it is important to further analyze in the cross validation and the hold out dataset, that under this step we aren't incurring in overfitting of the training dataset. When we include second degree Polynomial Features in the pipeline, we can see that the Accuracy improves to 97%. The hold out dataset presents a similar score showing that we aren't presenting overfitting. To understand in the model performs correctly, a Kfolds cross validation is conducted. First, a hold-out data set is taken out (20% of original dataset) and the cross validation is conducted in the remaining dataset (k = 5). Then, the second test is conducted in the hold-out dataset to confirm there is no overfitting in the model.

```
Obtained 5 positive Accuracy scores
Best Validation Accuracy: 0.97
Avg. Validation Accuracy: 0.97
Avg. Accuracy in hold-out dataset: 0.96
```

Random Forest: As a final step, to understand the feature importance so that the HR representatives have an execution plan by understanding what variables to assess first, a Random Forest algorithm is taken into consideration. With this method, we can measure the relative importance of each feature in the prediction of the model. The following graph shows the top important features explaining almost 100% of the model. As can be seen, the most important features to consider in order of importance are:



- Empowering employees with a fair number of projects (no less than 2).
- Employees with a low time in the company (less than 4 years) are more prone to leave, so is important to pay special attention.
- Employees with a low satisfaction level.
- Employees with a poor last evaluation are more prone to leave.

The final score of the model after the feature engineering process is of 0.97.

Annexes:

Table # 1: Raw data Information

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 10 columns):
satisfaction_level       14999 non-null float64
last_evaluation          14999 non-null float64
number_project           14999 non-null int64
average_montly_hours     14999 non-null int64
time_spend_company       14999 non-null int64
Work_accident            14999 non-null int64
left                     14999 non-null int64
promotion_last_5years    14999 non-null int64
sales                    14999 non-null object
salary                   14999 non-null object
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

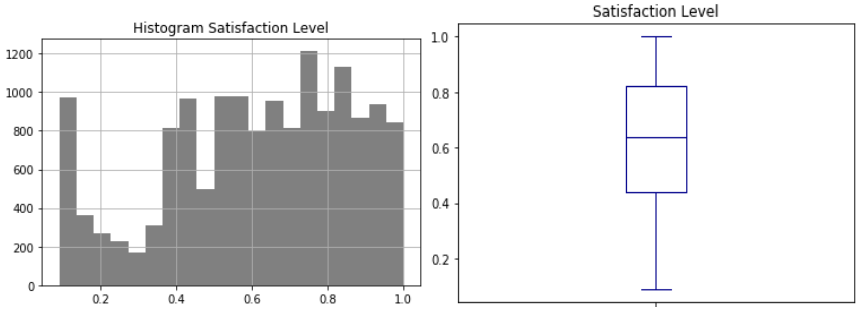
Tables # 2 & 3: Variables General Description

	count	mean	std	min	25%	50%	75%	max
satisfaction_level	14999.0	0.612834	0.248631	0.09	0.44	0.64	0.82	1.0
last_evaluation	14999.0	0.716102	0.171169	0.36	0.56	0.72	0.87	1.0
number_project	14999.0	3.803054	1.232592	2.00	3.00	4.00	5.00	7.0
average_montly_hours	14999.0	201.050337	49.943099	96.00	156.00	200.00	245.00	310.0
time_spend_company	14999.0	3.498233	1.460136	2.00	3.00	3.00	4.00	10.0
Work_accident	14999.0	0.144610	0.351719	0.00	0.00	0.00	0.00	1.0
left	14999.0	0.238083	0.425924	0.00	0.00	0.00	0.00	1.0
promotion_last_5years	14999.0	0.021268	0.144281	0.00	0.00	0.00	0.00	1.0

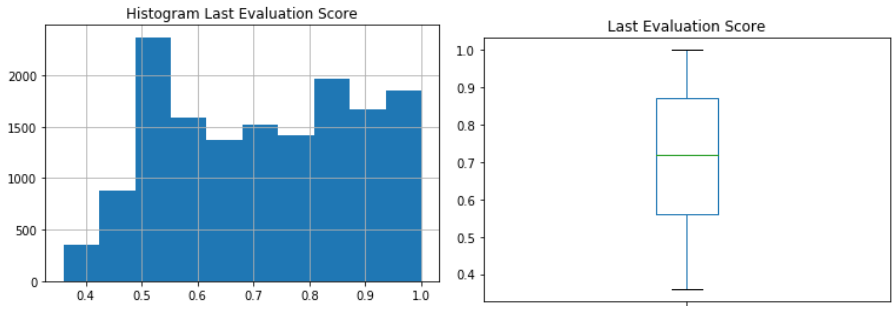
	count	unique	top	freq
sales	14999	10	sales	4140
salary	14999	3	low	7316

Annex # 4: Descriptive Analysis: Histograms & Box-plots

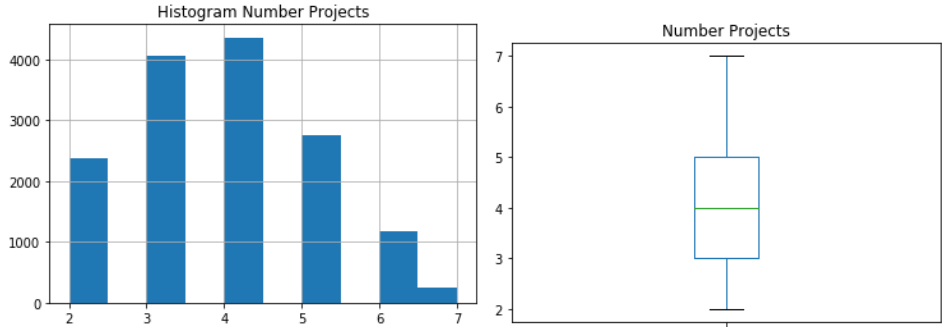
Satisfaction Level:



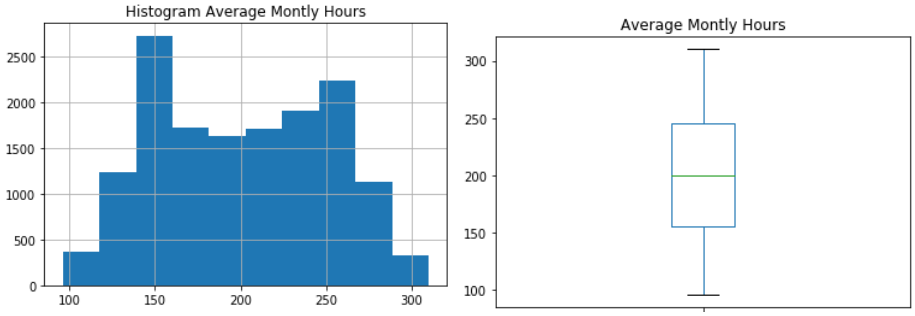
Last Evaluation Score:



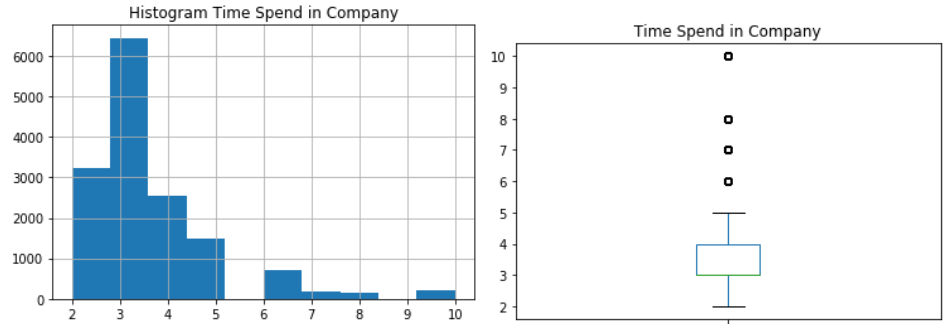
Number Projects:



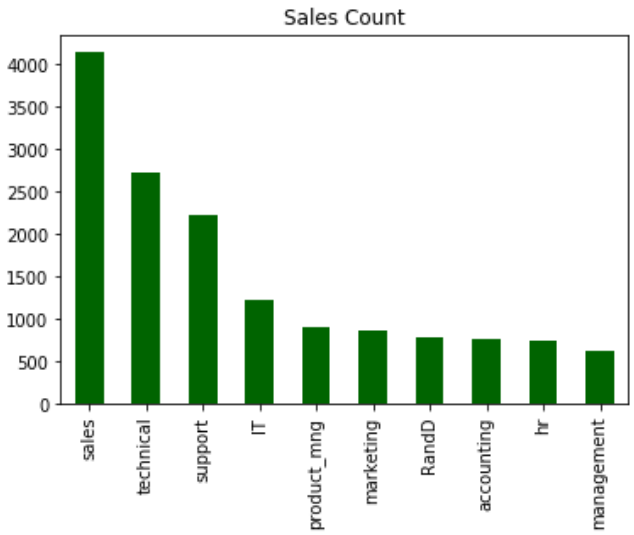
Average Monthly Hours



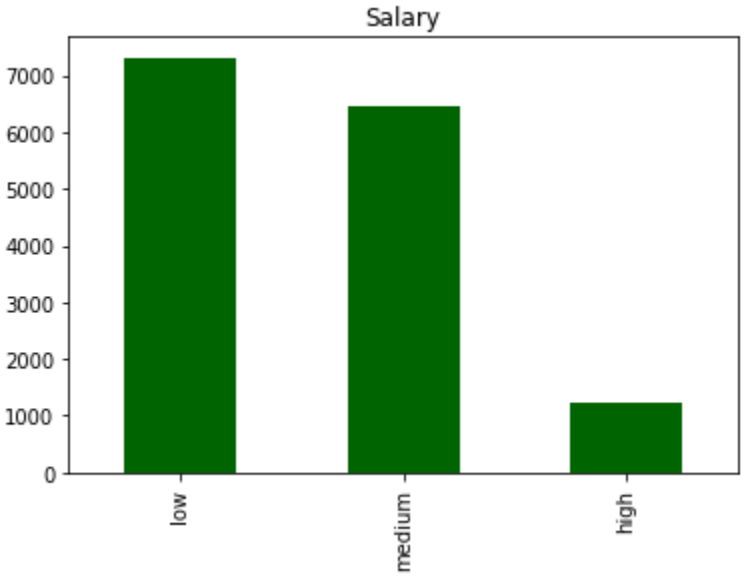
Time Spend in Company



Count of Sales (Departments)



Count of Salary Level



Annex # 5: Baseline Model Accurracy Scores – Kfold cross validation K=5

```
array([0.79533333, 0.786     , 0.80133333, 0.79166667, 0.78059353])
```

Annex # 6: Binning Details

```
#last_evaluation
raw_num_bins['last_evaluation_bin'] = pd.cut(raw_num_bins.last_evaluation ,6)
raw_num_bins = pd.concat([raw_num_bins, pd.get_dummies(raw_num_bins['last_evaluation_bin'],prefix='last_eval', prefix_sep='_')],
raw_num_bins.drop('last_evaluation_bin', inplace=True, axis=1)
raw_num_bins.drop('last_evaluation', inplace=True, axis=1)

#satisfaction_level
raw_num_bins['satisfaction_level_bin'] = pd.cut(raw_num_bins.satisfaction_level ,6)
raw_num_bins = pd.concat([raw_num_bins, pd.get_dummies(raw_num_bins['satisfaction_level_bin'],prefix='satisf_lev', prefix_sep='_
raw_num_bins.drop('satisfaction_level_bin', inplace=True, axis=1)
raw_num_bins.drop('satisfaction_level', inplace=True, axis=1)

#average_monthly_hours
raw_num_bins['average_montly_hours_bin'] = pd.cut(raw_num_bins.average_montly_hours ,6)
raw_num_bins = pd.concat([raw_num_bins, pd.get_dummies(raw_num_bins['average_montly_hours_bin'],prefix='avg_mh', prefix_sep='_')]
raw_num_bins.drop('average_montly_hours_bin', inplace=True, axis=1)
raw_num_bins.drop('average_montly_hours', inplace=True, axis=1)

#number_project
cats = { 1: "very_low", 2: "very_low", 3 : "low", 4 : "low", 5 : "medium", 6: "medium", 7: "high", 8: "high", 9:"very_high",10: "
raw_num_bins['number_project_cat'] = raw_num_bins['number_project'].map(cats)
raw_num_bins.drop('number_project', inplace=True, axis=1)
raw_num_bins = pd.concat([raw_num_bins, pd.get_dummies(raw_num_bins['number_project_cat'],prefix='num_proj', prefix_sep='_')], ax
raw_num_bins.drop('number_project_cat', inplace=True, axis=1)

#time_spend_company
cats = { 2: "very_low", 3: "low", 4: "low", 5 : "medium", 6: "medium", 7: "high", 8: "high", 9:"high",10: "high"}
raw_num_bins['time_spend_company_cat'] = raw_num_bins['time_spend_company'].map(cats)
raw_num_bins.drop('time_spend_company', inplace=True, axis=1)
raw_num_bins = pd.concat([raw_num_bins, pd.get_dummies(raw_num_bins['time_spend_company_cat'],prefix='t_spend_c', prefix_sep='_')
raw_num_bins.drop('time_spend_company_cat', inplace=True, axis=1)
```