MACHINE LEARNING II

Individual Assignment 1

PREDICTING EMPLOYEE TURNOVER

Presented to Professor Jesús Salvador Renero Quintero

By Paul Jacques-Mignault – Section 01

Link to Script:

https://www.kaggle.com/pauljac1/pauljacques-mignault-mlii-assignment1?scriptVersionId=10628692

Sunday, February 17th 2019

Index

I. Data Acquisition

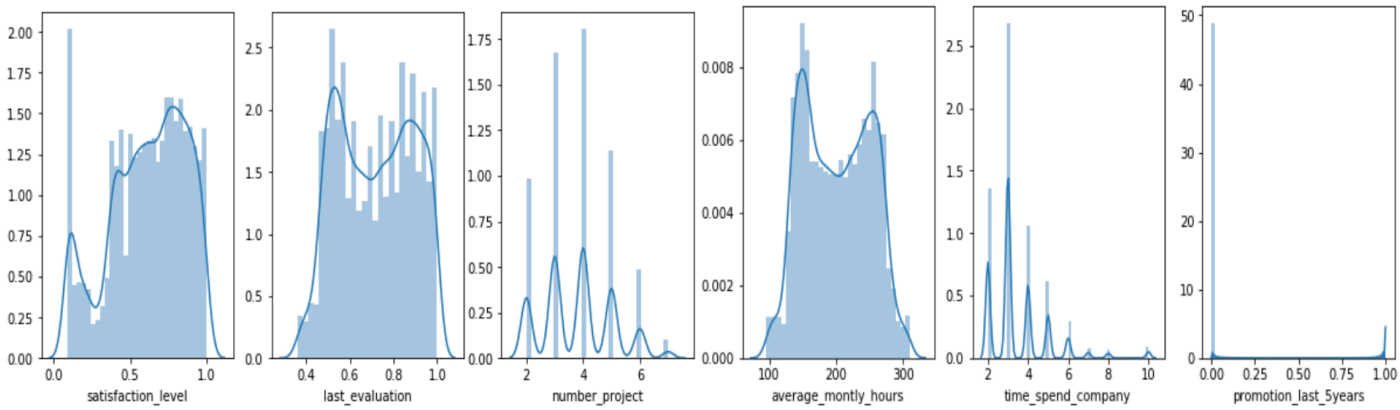
The data was acquired directly from Kaggle, in the 'Input' draft environment. In order to run the phython script, the data 'turnover.csv' must be saved in the 'Input' directory on Kaggle.

II. Data Understanding

The dataset includes 15,000 tuples and 10 columns. There are no null values. The variables are either numerical **continuous**; 'satisfaction_level', 'last_evaluation'; **integers**; 'average_montly_hours', 'number_project', 'time_spend_company'; **numerically-encoded categorical variables**: work_accident', 'left', 'promotion_last_5years'; as well as **categorical variables**; 'sales', and 'salary'. The target variable is 'left', i.e. whether or not the employee chose to leave the company. The objective of the present analysis is to establish a classification algorithm to predict whether or not a particular employee, given the information contained in the aforementioned columns, will leave the company.

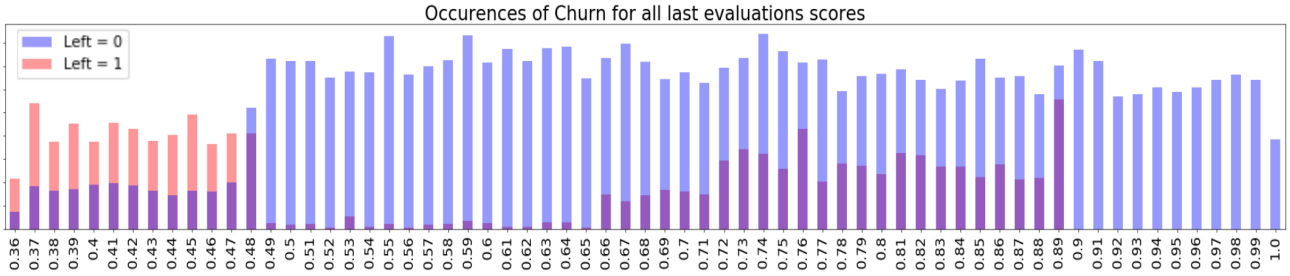
The first step to gain a better understanding of the data was to inquire on the distribution of categorical variables. This is to ensure no particular category is over/under-represented for any column; if only 1% of tuples belong to a particular category, the latter will likely be underrepresented in either the train/test split datasets. In this case, **only 2% of the 15,000 employees had received a promotion in the last 5 years**. The logistic regression model will evaluate whether excluding this variable from the analysis improves accuracy. As many as 23.8% of employees had left the company; implying balancing the dataset when splitting between train and test will not be necessary. Furthermore, the second step is to ensure **variables are not skewed**. Typically, normally distributed data fits regression models better than skewed data. In this case, as shown in Figure 1, two variables were right-skewed with **the absolute value of the skewness being greater than 0.5**: 'salary', i.e. few people were making high salaries, and 'time_spend_company', i.e. few people had been employed by the company for 4 years or more.

Figure 1: Distribution of Variables



The above figure shows the distribution of 6 of the 9 variables; none other than the aforementioned ones present significant skewness.

Figure 2:



Further to the distribution of the variables, it is important for the user to understand how variation among the dataset's variables affects our target; 'left'. For instance, **employee churn is concentrated at specific intervals of last evaluation scores**, as show in the above figure. Intuitively, the lowest-performing employees with evaluation score 0.48 and below are likely to churn. The employees with medium evaluation scores, between 0.66 and 0.89 inclusively, were also likely to churn. The very top-performers, with last evaluation scores of 0.8, however, did not leave the company. The latter pattern makes intuitive sense, since the company is likely to be keener to retain its top-performing staff. Similar patterns were observed across most variables; churners were concentrated in specific intervals. The said patterns are shown in the **appendix**. Additionally, no variable contained in the data set was significantly correlated with neither the target variable, nor with one another; the correlation heatmap can be found in the appendix.

III. Data Preparation

As the dataset was already relatively clean, only few steps were required in the data preparation stage. First of all, in order to proceed to the logistic regression, all variables need to be numerical. The two categorical variables, 'salary', and 'sales' had to be **numerically encoded**. For the former, an ordinal numerical variable was created; 0 meant low salary, 1 meant medium salary, and 2 meant high salary. For the latter, 10 new columns were created and added into the data frame; each were a numerical encoding consisting of 1 or 0, if the employee belonged to that particular column's department.

Additionally, skewness was fixed for 'salary' and 'time_spend_company', both of which were right-skewed. The methodology used was the **Box-Cox transformation**, which normalizes the normalization of both variables. It raises all values to the exponent lambda usually between -5 and 5, subtracts one and divides the result by lambda; the transformation uses the optimal value to make thee distribution as close as possible to the normal distribution.

IV. Establish Baseline Model

With the logistic regression model, a baseline was established. It included all transformed and numerically encoded variables. The accuracy score obtained was **79.33%**, as shown in the following figure. For reproducibility and consistency's sake, **the seed was set at 1,000 the random state at 101** for all models' scores in the present. Though the dataset was balanced with nearly 24% of employees who had churned, the size of the test set was 30% of tuples, to ensure the model could be trained with a sufficient number of churned employees. **A penalty was added directly in the logistic regression function to prevent overfitting as 'penalty = l1'**, implying a **lasso regression** is integrated to all models. The regression's least important coefficients are then reduced to zero, thus erasing the feature completely.

V. Feature Engineering – Feature Creation

For each of the following feature creation functions, the accuracy was measured on the updated dataset, as reported below.

The first attempt to improve upon the baseline model is to **standardize** all numerical variables. If numerical variables are measured on different scales, as is the case for instance between 'average_montly_hours', usually between 100 and 250, and 'satisfaction_level', between 0 and 1. Standardizing may slightly improve the model, whilst making the coefficients easily comparable with another. **The resulting accuracy in the test set was 79.20%;** the function was rejected as it didn't improve the model.

The second attempt included clustering departments where it made intuitive sense. A new categorical variable was created; if the employee works in either management, HR, or accounting, they were labeled as working in a 'support_function', as those presented a lower propensity to churn than other departments. **The resulting accuracy in the test set was 79.27%;** the function was rejected as it didn't improve the model.

The third attempt to engineer feature to improve the model included remove columns with underrepresented features, or with **fewer than 5% of tuple belonging to one category**. In this case, this applies to **'promotion_last_5years'**. In the data exploration phase, nearly all of the employees that had been promoted did stay with the company. Despite having little representation in the dataset, the feature may be statistically significant in determining likelihood to churn. In any case, excluding it resulted in an **accuracy in the test set of 79.22%**; the function was rejected as it didn't improve the model.

The fourth tentative included **binning each numerical variable** following specific churning patterns. For instance, in figure 2, different bins can be identified looking at employee behaviour: more employees churn if their scores are between 0 and 0.47; fewer churn if they score between 0.48 and 0.49, or 0.88 and 0.89, even fewer between 0.66 and 0.87. Churn is highly unlikely outside of these bins. This same process is repeated for the following variables: 'satisfaction_level', 'number_project', 'average_montly_hours', and "time_spend_company'. **Being overly precise in determining the extensions of the bins may result in overfitting; setting and 'l1' penalty in the logistic regression function** and feature selection in the next section may ameliorate overfit. **The resulting accuracy in the test set was 95.20%**; the function will be considered to improve the model.

The fifth attempt to improve the model was to bin the same numerical variables. However, instead of binning them following churning patterns, **the variables were binned by quartiles**. Though precise in mirroring churning behaviour, this method may be effective in improving accuracy and reducing overfitting. **The resulting accuracy in the test set was 85.89%**; the function does not perform as well as the previous binning method.

The feature creation method resulting in the best accuracy is **binning variables** following churn behaviour – steps will be taken in the coming sections to prevent overfit.

VI. Feature Engineering – Feature Selection

After binning all numerical variables following churn patterns, the next step is to select only the statistically relevant columns to be included in the logistic regression. The first algorithm to filter for significant variables is **the stepwise selection method, which includes both forward and backward approaches.** The forward approach starts by including the first feature in the set if it fulfils the fit criterion, i.e. p-value below 0.05 for statistical significance, then looks at the second feature, the third, etc., until it has assessed the significance of all features, only adding them in the set if it fulfills the criterion. The backward method works in the same way, though starts with all features, and removes them if not significant. In this case, **all features were statistically significant**, with the highest p-value of 0.006, observed for the 'sales_product_mng' feature.

The stepwise approach however does not reduce collinearity, and does not examine the interactions between variables and one another. For this reason, the stepwise approach needs to be complemented with other methodologies. The **information gain** method is also useful to establish which variables are the most important; a number of projects binned as 'high', i.e. having either 2 or 6 projects and a high probability of leaving, has an information gain of 0.11. Among the features with the highest information gains are bins created from the 'number_project' and 'satisfaction_level.

Additionally, the **Chi-Squared** method was also used; the latter examines whether 2 categorical variables are related in any way. In this case, the variables most closely related to the target variable 'left' are again the bins created from the 'number_project' and 'satisfaction_level, consistently with the information gain method.

Finally, another method was used to select features; the **recursive feature elimination (RFE)**. This method finds the best-performing subset of features, and excludes the worst-performing ones until the dataset is exhausted. By default, it selects half of the dataset's number of features. The dataset was then trained with all number of features between 1 and 40, as the dataset had a total of 40 variables. The results of all numbers of recursive feature elimination are detailed in the appendix. **Selecting all 40 variables delivered the best accuracy score of 95.24% on the test set.**

In order to further prevent overfitting by selecting all 40 variables, a lasso regression was performed with varying λ parameter, whereby λ represents the coefficient threshold below which the said coefficient is reduced to 0. 3 different levels of λ were tried: 0.01, 0.0001, 0.000001. For each λ, the accuracy level was measured. Each λ for all lasso regressions significantly worsened accuracy results, as detailed in the appendix. Given a lasso regression is already applied in the logistic regression function in python, all features were found to be statistically significant in the stepwise approach, and the RFE approach delivered its best accuracy scores when all features were selected, **the cross-validation will be carried out with all 40 features included in the data.**

VII. Cross-Validation and Final Metrics

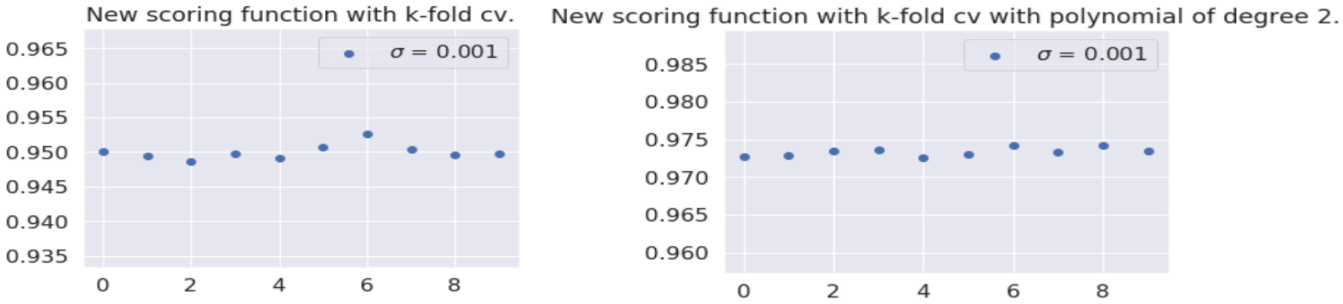
The next step is to perform **k-fold cross-validation** on the model. This operation consists in shuffling the dataset at random, and splitting it into k samples. For every k, one sample is identified as the **'hold-out' dataset**, and is used as the test set. The other groups are used to train the model, which is then assessed on the hold-out set's performance. This method ensures the model provides a lower bias towards a particular segment of the data set, and outputs more conservative results.

In this case, cross-validation was performed with **k=10**. 10 samples typically balance out high bias and excessive variance. This cross-validation resulted in an average accuracy across the 10 samples of **95.06%, with a standard deviation of 0.004709, or 0.47%.** This confirms our forecast accuracy should by around 95% with minimal variation. The maximum accuracy across the 10 samples was 95.76%, and the accuracy in the hold-out set was 95.04%.

To assess whether further improvements in accuracy are possible, **a polynomial of degree 2** was introduced. The logistic regression assumes a linear combination of variables can be used to predict the target variable. If this assumption is not met, using a polynomial generates a more extensive matrix with 1, feature1, feature2, feature1^2, feature1*feature2, and featur2^2 for a polynomial of degree 2. This may help non-linear relationships in the data. In fact, **accuracy improved to 97.53% with a polynomial of degree 2.** The same cross-validation process was repeated for the regression with polynomial of degree 2 to see if the new model presented excessive bias or variance. **Accuracy improved to 97.24%, with a standard deviation of 0.002694, or 0.27%.** Trying other degrees for polynomials would be time-consuming computationally expensive, and at best achieve marginal improvements. For this reason, **the polynomial of degree 2 is maintained.**

Finally, using the 'cross_val_score', both regressions were compared on accuracy, with or without polynomial. **This function enables the user to assess if the model's accuracy on average improve, and rapidly identify excessive variance** or bias. The operation was also repeated for k=20; those results are detailed in the appendix. For k=10, the results are displayed in the figure below.

Figure 3:



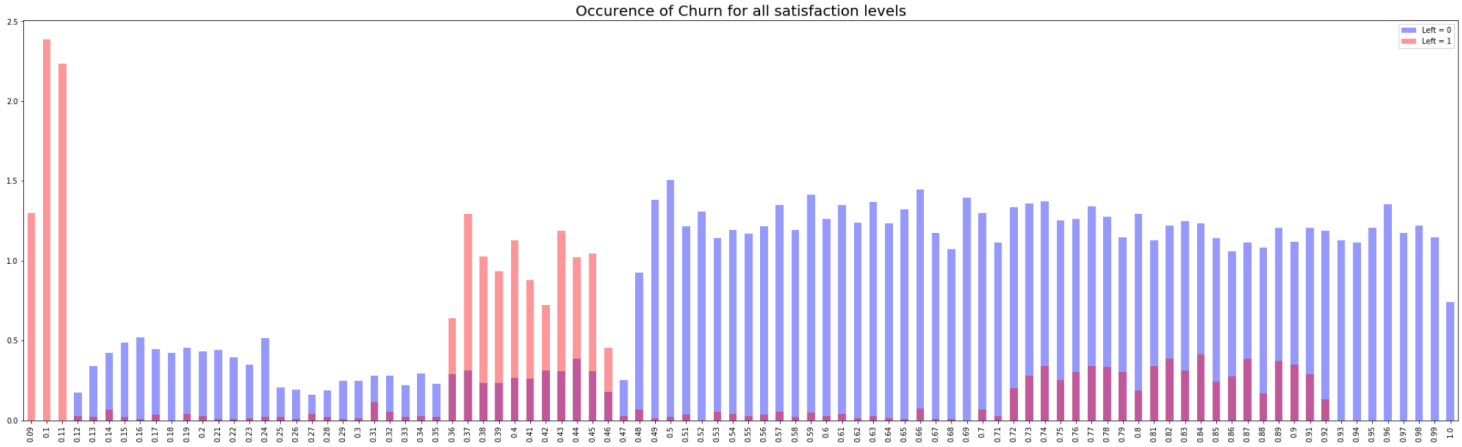
In the above graph, the model including a polynomial of degree 2 clearly outperforms the logistic regression model by roughly 2 percentage points, and consistently so across all the samples formed in the k-fold validation. The model maintains minimal standard deviation of 0.001. in both cases. The cross-validation was repeated several times to ensure resilience of the results. The final metric scores for the following:

Avg. CV accuracy: 97.24% +/- 0.002694
Accuracy from entire-dataset estimator: 97.49%
Best estimator accuracy score: 97.56%

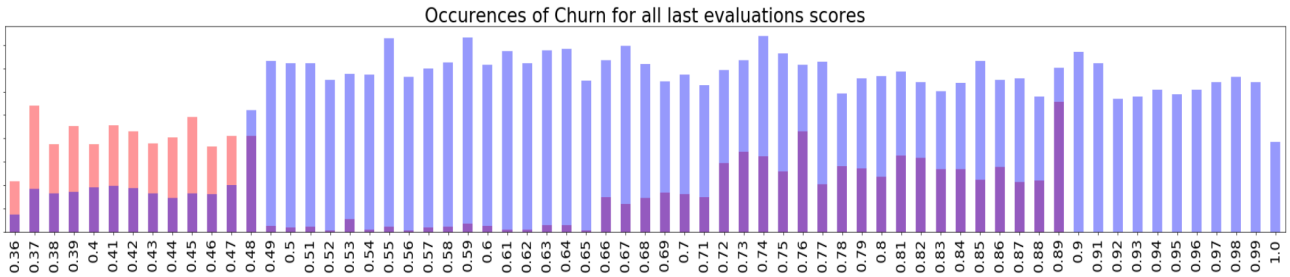
In conclusion, a high accuracy score was achieved by trying out several different models; identifying relevant bins for all features proved the most effective. After assessing several feature selection methods, all 40 were statistically significant and selected to be included in the model. **To prevent overfitting**, a lasso regression was applied. High accuracy scores were then confirmed with cross-validation, and further improved **consistently** by integrating a polynomial of degree 2. Accuracy from an independent test set would reliably and definitely confirm whether or not the model is overfitting.

APPENDIX

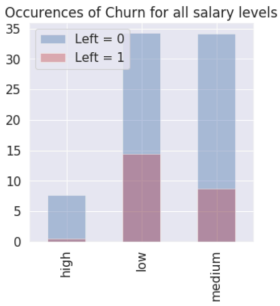
Appendix 1: Occurrence of Churn for all Satisfaction Levels



Appendix 2: Occurrence of Churn for all Last Evaluation Scores

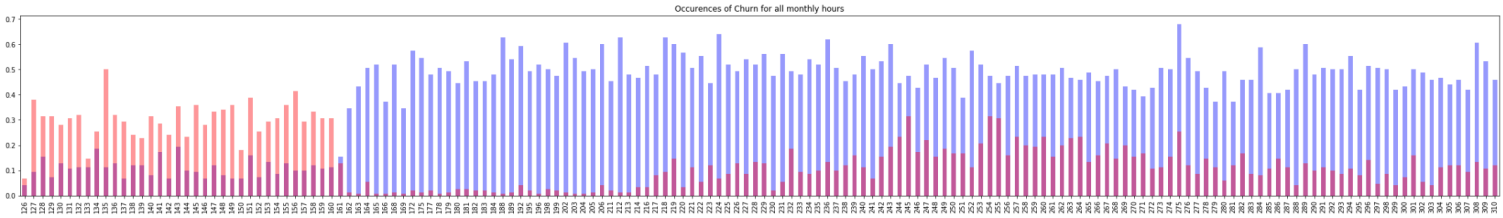


Appendix 3: Occurrence of Churn for all Salary Levels

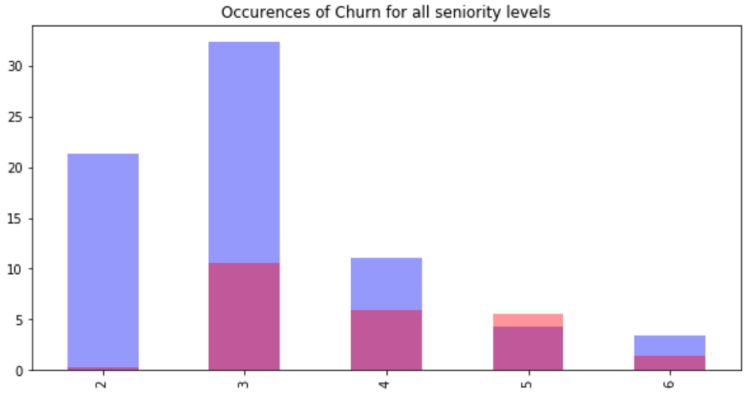


Appendix 4: Occurrence of Churn for all Number of Projects

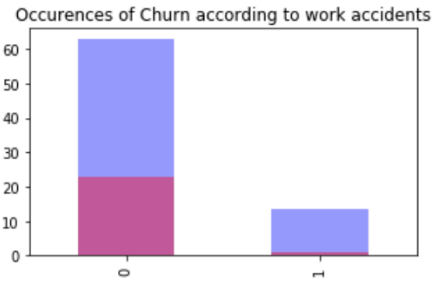
Appendix 5: Occurrence of Churn for all Number of Projects



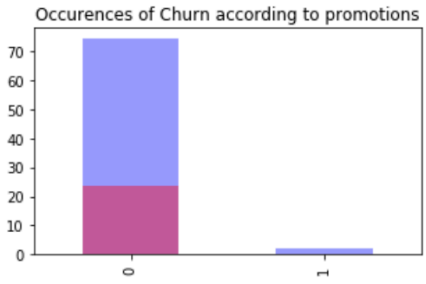
Appendix 6: Occurrence of Churn for all Number of Years Spent in the Company



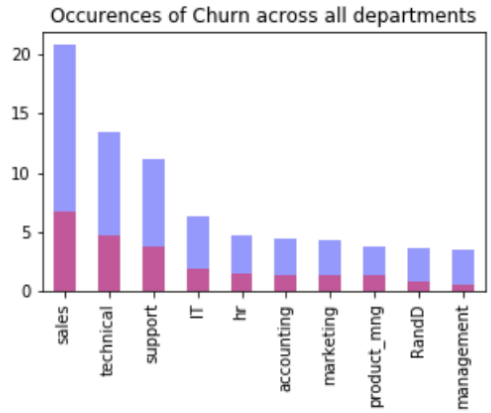
Appendix 7: Occurrence of Churn according to Work Accidents



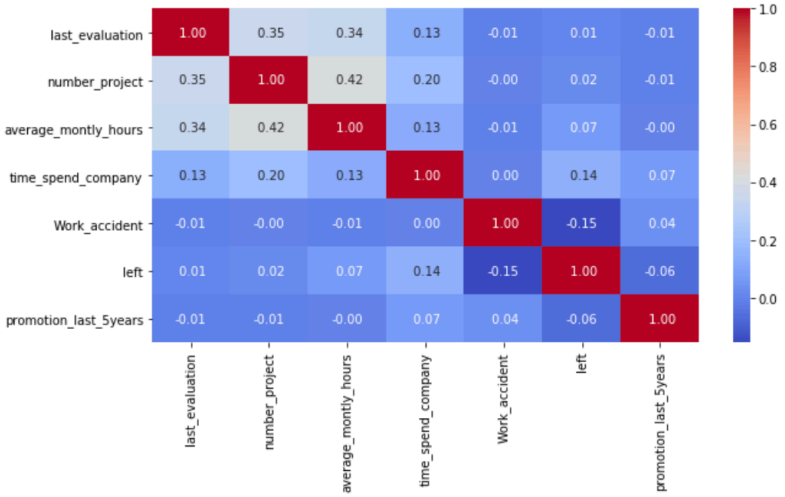
Appendix 8: Occurrence of Churn according to Promotion in the Last 5 Years



Appendix 9: Occurrence of Churn for all Number of Projects



Appendix 10: Correlation Heatmap for all of the Dataset's Features



In the above heatmap, the user can deduct that there is no significant correlation between the target variable, 'left', and all others of the dataset's variables. There is a correlation coefficient below zero with 'Work_accident' and 'promotion_last_5years', implying those are negatively correlated with the target. Aside from that, no independent variables are highly correlated with one another. Only 'number_project' and 'average_montly_hours' have a correlation coefficient above 0.4.

Appendix 11: Baseline Model's Results

[23]:
```
## Establish Baseline Model with Logistic Regression:
score_model(dataset)
```

```
[[3147  284]
 [ 646  423]]
              precision    recall  f1-score   support

           0       0.83      0.92      0.87      3431
           1       0.60      0.40      0.48      1069

   micro avg       0.79      0.79      0.79      4500
   macro avg       0.71      0.66      0.67      4500
weighted avg       0.77      0.79      0.78      4500

Accuracy: 0.7933
```

Appendix 12: Model's Results after Standardizing variables

```
[[3143  288]
 [ 648  421]]
              precision    recall  f1-score   support

           0       0.83      0.92      0.87      3431
           1       0.59      0.39      0.47      1069

    micro avg       0.79      0.79      0.79      4500
    macro avg       0.71      0.65      0.67      4500
weighted avg       0.77      0.79      0.78      4500

Accuracy: 0.7920
```

Appendix 13: Model's Results after creating 'support_function' feature

```
[[3146  285]
 [ 648  421]]
              precision    recall  f1-score   support

           0       0.83      0.92      0.87      3431
           1       0.60      0.39      0.47      1069

    micro avg       0.79      0.79      0.79      4500
    macro avg       0.71      0.66      0.67      4500
weighted avg       0.77      0.79      0.78      4500

Accuracy: 0.7927
```

Appendix 14: Model's Results after removing the 'promotion_last_5years' underrepresented features

```
[[3148  283]
 [ 652  417]]
              precision    recall  f1-score   support

           0       0.83      0.92      0.87      3431
           1       0.60      0.39      0.47      1069

    micro avg       0.79      0.79      0.79      4500
    macro avg       0.71      0.65      0.67      4500
weighted avg       0.77      0.79      0.78      4500

Accuracy: 0.7922
```

Appendix 15: Model's Results after binning variables

```
[[3323  108]
 [ 108  961]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97      3431
           1       0.90      0.90      0.90      1069

    micro avg       0.95      0.95      0.95      4500
    macro avg       0.93      0.93      0.93      4500
weighted avg       0.95      0.95      0.95      4500

Accuracy: 0.9520
```

Appendix 15: Model's Results after binning variables

```
[[3221  210]
 [ 425  644]]
              precision    recall  f1-score   support

           0       0.88      0.94      0.91      3431
           1       0.75      0.60      0.67      1069

    micro avg       0.86      0.86      0.86      4500
    macro avg       0.82      0.77      0.79      4500
weighted avg       0.85      0.86      0.85      4500

Accuracy: 0.8589
```

Appendix 16: Output of the Stepwise Feature Selection

```
X = dataset.drop(labels=["left"],axis=1)
Y = dataset['left']
result = stepwise_selection(X, Y)

print('resulting features:')
print(result)
```

```
Add  sts_(0.35, 0.46]              with p-value 0.0
Add  tsc_(0.98, 1.04]              with p-value 0.0
Add  np_Very high                  with p-value 0.0
Add  sts_(0.0, 0.11]               with p-value 0.0
Add  sts_(0.71, 0.92]              with p-value 4.70835e-137
Add  np_high                       with p-value 4.00358e-107
Add  le_(0.89, 1.0]                with p-value 6.81623e-61
Add  np_medium                     with p-value 1.11032e-50
Add  tsc_(1.04, 1.1]               with p-value 1.08631e-45
Add  sal_num                       with p-value 3.56707e-35
Add  Work_accident                 with p-value 4.46518e-31
Add  am_(165, 178]                 with p-value 2.36235e-29
Add  am_(96, 131]                  with p-value 1.22794e-32
Add  tsc_(0.0, 0.8]                with p-value 3.73372e-24
Add  sts_(0.11, 0.35]              with p-value 8.05895e-23
Add  am_(179, 259]                 with p-value 1.85915e-11
Add  promotion_last_5years         with p-value 3.85568e-07
Add  le_(0.0, 0.47]                with p-value 3.67302e-06
Add  le_(0.65, 0.88]               with p-value 3.89255e-06
Add  am_(178, 179]                 with p-value 6.50476e-05
Add  np_low                        with p-value 1.26768e-66
Add  sales_RandD                   with p-value 0.00271259
Drop np_low                        with p-value 0.231729
Add  sts_(0.92, 1.0]               with p-value 0.00496327
Add  sts_(0.46, 0.71]              with p-value 0.0
Add  sales_IT                      with p-value 0.00855003
Drop sts_(0.0, 0.11]               with p-value 0.99237
Add  sts_(0.0, 0.11]               with p-value 0.0
Add  sales_product_mng             with p-value 0.00621258
```

Appendix 17: Output of the Information Gain Feature Selection

[29]:
```
information_gain(dataset)
```

```
[('np_high', 0.11494812401168297),
 ('sts_(0.35, 0.46]', 0.10205141531126687),
 ('sts_(0.0, 0.11]', 0.09123837587054506),
 ('sts_(0.46, 0.71]', 0.07524711567771479),
 ('np_low', 0.06751235137388635),
 ('tsc_(0.0, 0.8]', 0.052552245451108946),
 ('tsc_(0.98, 1.04]', 0.027358761452307853),
 ('sts_(0.92, 1.0]', 0.026078007591957414),
 ('np_Very high', 0.024975112248797005),
 ('np_medium', 0.023089601455358988),
 ('am_(179, 259]', 0.020550355000067744),
 ('am_(165, 178]', 0.016499668181326338),
 ('Work_accident', 0.01454223778880688),
 ('sal_num', 0.014483912915501673),
 ('le_(0.65, 0.88]', 0.011083442705596636),
 ('am_(131, 165]', 0.01086524721693774),
 ('sts_(0.11, 0.35]', 0.008728334222479366),
 ('tsc_(0.9, 0.98]', 0.006403508473172365),
 ('le_(0.89, 1.0]', 0.004372019036923136),
 ('am_(259, 287]', 0.002846816213531833),
 ('le_(0.0, 0.47]', 0.002526291024488564),
 ('promotion_last_5years', 0.0024956688052774068),
 ('le_(0.47, 0.48]', 0.0024479132252004585),
 ('sts_(0.71, 0.92]', 0.002447552810158571),
 ('sales_RandD', 0.0011902748045097208),
 ('sales_management', 0.001177892497524126),
 ('am_(178, 179]', 0.0008702614575149201),
 ('am_(96, 131]', 0.00057105443159991),
 ('sales_hr', 0.0003819234128577889),
 ('sales_technical', 0.00019890328974747629),
 ('tsc_(0.8, 0.9]', 0.00013513619993202333),
 ('le_(0.88, 0.89]', 0.00012813140904056192),
 ('sales_accounting', 0.00011278757094314307),
 ('sales_product_mng', 6.188642260637062e-05),
 ('sales_IT', 6.052358091253318e-05),
 ('sales_support', 5.676321852080002e-05),
 ('sales_sales', 4.904027937213673e-05),
 ('tsc_(1.04, 1.1]', 9.764485377920795e-06),
 ('le_(0.48, 0.65]', 5.030013605106243e-06),
 ('sales_marketing', 3.6969949563508964e-07)]
```

Appendix 18: Output of the Chi-Squared Feature Selection

```
[30]: # Perform Chi-Squared test for most significant categorical variables
      get_chi2_test(dataset)
```

```
 1.020e-01 1.992e+02 3.931e+00 1.104e+02 8.193e+02 2.922e+03 1.085e+03
 3.531e+02 1.700e+01 2.537e+02 3.014e+02 1.786e+01 3.245e+02 7.895e+01
 8.793e+02 2.317e+00 1.705e+02 8.706e+02 2.734e-01]
[[0. 1. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [1. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [0. 1. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 1. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [0. 1. 0. 0. 0. 1. 0. 0. 0. 0.]]
['sts_(0.0, 0.11]', 'sts_(0.35, 0.46]', 'sts_(0.46, 0.71]', 'sts_(0.92, 1.0]', 'np_Very high', 'np_high', 'np_low', 'np_medium', 'tsc_(0.0, 0.8]', 'tsc_(0.98, 1.04]']
```

Appendix 19: Output of the RFE – Default Mode of Selecting 20 features

```
Num Features: 20
Selected Features: [False  True  True False False False False False False False False
 False  True  True  True  True False  True False False False False False
 False  True False  True  True  True  True  True  True  True False
  True  True  True  True]
Feature Ranking: [ 3  1  1  7  6 14 11  5 18  4 17 15 13  1  1  1  1 19  1  9 12 20 21 10
  2  1  8  1  1  1  1  1  1  1  1 16  1  1  1  1]

['Work_accident',
 'promotion_last_5years',
 'sts_(0.0, 0.11]',
 'sts_(0.11, 0.35]',
 'sts_(0.35, 0.46]',
 'sts_(0.46, 0.71]',
 'sts_(0.92, 1.0]',
 'np_Very high',
 'np_low',
 'np_medium',
 'am_(96, 131]',
 'am_(131, 165]',
 'am_(165, 178]',
 'am_(178, 179]',
 'am_(179, 259]',
 'am_(259, 287]',
 'tsc_(0.8, 0.9]',
 'tsc_(0.9, 0.98]',
 'tsc_(0.98, 1.04]',
 'tsc_(1.04, 1.1]']
```

Appendix 20: Output of the RFE – For all Possible number of features

```
With 1 variables:
Accuracy: 0.7624
With 2 variables:
Accuracy: 0.7624
With 3 variables:
Accuracy: 0.8187
With 4 variables:
Accuracy: 0.8187
With 5 variables:
Accuracy: 0.8896
With 6 variables:
Accuracy: 0.8896
With 7 variables:
Accuracy: 0.8896
With 8 variables:
Accuracy: 0.8911
With 9 variables:
Accuracy: 0.8996
With 10 variables:
Accuracy: 0.9020
With 11 variables:
Accuracy: 0.9018
With 12 variables:
Accuracy: 0.9018
With 13 variables:
Accuracy: 0.8944
With 14 variables:
Accuracy: 0.8947
With 15 variables:
Accuracy: 0.9029
With 16 variables:
Accuracy: 0.9031
With 17 variables:
Accuracy: 0.8962
With 18 variables:
Accuracy: 0.8980
With 19 variables:
Accuracy: 0.9449
With 20 variables:
Accuracy: 0.9460
```

```
With 21 variables:
Accuracy: 0.9460
With 22 variables:
Accuracy: 0.9460
With 23 variables:
Accuracy: 0.9460
With 24 variables:
Accuracy: 0.9460
With 25 variables:
Accuracy: 0.9460
With 26 variables:
Accuracy: 0.9460
With 27 variables:
Accuracy: 0.9460
With 28 variables:
Accuracy: 0.9460
With 29 variables:
Accuracy: 0.9460
With 30 variables:
Accuracy: 0.9460
With 31 variables:
Accuracy: 0.9460
With 32 variables:
Accuracy: 0.9460
With 33 variables:
Accuracy: 0.9460
With 34 variables:
Accuracy: 0.9460
With 35 variables:
Accuracy: 0.9460
With 36 variables:
Accuracy: 0.9460
With 37 variables:
Accuracy: 0.9460
With 38 variables:
Accuracy: 0.9460
With 39 variables:
Accuracy: 0.9460
With 40 variables:
Accuracy: 0.9460
```

Appendix 21: Output of the Lasso Regression Function

```
training score: 0.0
test score:  -1.753892972322646e-05
number of features used:  0
training score for alpha=0.01: 0.594534450981947
test score for alpha =0.01:  0.5908529123494006
number of features used: for alpha =0.01: 13
training score for alpha=0.0001: 0.6527090181527742
test score for alpha =0.0001:  0.6503829516979018
number of features used: for alpha =0.0001: 38
training score for alpha=0.000001: 0.6528351337083305
test score for alpha =0.000001:  0.6506744774868926
number of features used: for alpha =0.000001: 39
LR training score: 0.9487570244785217
LR test score:  0.9495555555555556
```

Appendix 22: Model's Score with Cross-Validation with k=10

```
Obtained 10 positive accuracy scores
Best CV accuracy: 0.9576
Avg. CV accuracy: 0.9506 +/- 0.004709
Accuracy in hold-out dataset: 0.9504
```

```
Accuracy from entire-dataset estimator: 0.9504
Best estimator accuracy score: 0.9522
```

Appendix 22: Model's Output with Polynomial of Degree 2

```
[[3389   42]
 [  69 1000]]
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      3431
           1       0.96      0.94      0.95      1069

   micro avg       0.98      0.98      0.98      4500
   macro avg       0.97      0.96      0.97      4500
weighted avg       0.98      0.98      0.98      4500

Accuracy: 0.9753
```

Appendix 23: Model's Score with Polynomial of Degree 2 and Cross-Validation with k=10

```
Obtained 10 positive accuracy scores
Best CV accuracy: 0.9771
Avg. CV accuracy: 0.9724 +/- 0.002694
Accuracy in hold-out dataset: 0.9749
```

```
Accuracy from entire-dataset estimator: 0.9749
Best estimator accuracy score: 0.9756
```

Appendix 24: Model's Accuracy without and with Polynomial of Degree 2 and Cross-Validation with k=20 for all subsamples