notebook

April 8, 2022

1 Can you help reduce employee turnover?

1.1 Background

You work for the human capital department of a large corporation. The Board is worried about the relatively high turnover, and your team must look into ways to reduce the number of employees leaving the company.

The team needs to understand better the situation, which employees are more likely to leave, and why. Once it is clear what variables impact employee churn, you can present your findings along with your ideas on how to attack the problem.

1.2 The data

The department has assembled data on almost 10,000 employees. The team used information from exit interviews, performance reviews, and employee records.

- "department" - the department the employee belongs to.
- "promoted" - 1 if the employee was promoted in the previous 24 months, 0 otherwise.
- "review" - the composite score the employee received in their last evaluation.
- "projects" - how many projects the employee is involved in.
- "salary" - for confidentiality reasons, salary comes in three tiers: low, medium, high.
- "tenure" - how many years the employee has been at the company.
- "satisfaction" - a measure of employee satisfaction from surveys.
- "bonus" - 1 if the employee received a bonus in the previous 24 months, 0 otherwise.
- "avg_hrs_month" - the average hours the employee worked in a month.
- "left" - "yes" if the employee ended up leaving, "no" otherwise.

[ ]:
```
import pandas as pd
df = pd.read_csv('./data/employee_churn_data.csv')
df.head()
```

[ ]:
```
    department  promoted    review  projects  salary  tenure  satisfaction  \
0  operations         0  0.577569         3     low     5.0      0.626759
1  operations         0  0.751900         3  medium     6.0      0.443679
2     support         0  0.722548         3  medium     6.0      0.446823
3   logistics         0  0.675158         4    high     8.0      0.440139
4       sales         0  0.676203         3    high     5.0      0.577607

   bonus  avg_hrs_month left
```

```
0       0      180.866070   no
1       0      182.708149   no
2       0      184.416084   no
3       0      188.707545   no
4       1      179.821083   no
```

1.3 Competition challenge

Create a report that covers the following: 1. Which department has the highest employee turnover? Which one has the lowest? 2. Investigate which variables seem to be better predictors of employee departure. 3. What recommendations would you make regarding ways to reduce employee turnover?

1.4 Judging criteria

CATEGORY	WEIGHTING	DETAILS

| **Recommendations** | 35% |

Clarity of recommendations - how clear and well presented the recommendation is.

Quality of recommendations - are appropriate analytical techniques used & are the conclusions valid?

Number of relevant insights found for the target audience.

|

| **Storytelling** | 35% |

How well the data and insights are connected to the recommendation.

How the narrative and whole report connects together.

Balancing making the report in-depth enough but also concise.

| | **Visualizations** | 20% |

Appropriateness of visualization used.

Clarity of insight from visualization.

| | **Votes** | 10% |

Up voting - most upvoted entries get the most points.

|

1.5 Checklist before publishing into the competition

- Rename your workspace to make it descriptive of your work. N.B. you should leave the notebook name as notebook.ipynb.
- Remove redundant cells like the judging criteria, so the workbook is focused on your story.

- Make sure the workbook reads well and explains how you found your insights.
- Check that all the cells run without error.

1.6 Time is ticking. Good luck!

[ ]:
```
import pandas as pd
import numpy as np
df_original = pd.read_csv('./data/employee_churn_data.csv')
df_original.head()
```

[ ]:
```
    department  promoted    review  projects  salary  tenure  satisfaction  \
0  operations         0  0.577569         3     low     5.0      0.626759
1  operations         0  0.751900         3  medium     6.0      0.443679
2     support         0  0.722548         3  medium     6.0      0.446823
3   logistics         0  0.675158         4    high     8.0      0.440139
4       sales         0  0.676203         3    high     5.0      0.577607

   bonus  avg_hrs_month left
0      0     180.866070   no
1      0     182.708149   no
2      0     184.416084   no
3      0     188.707545   no
4      1     179.821083   no
```

Because the columns 'salary' and 'left' have non numerical values we use the following conversions: - For column 'left': 'no' = 0 and 'yes' = 1 - For column 'salary': 'low' = 1, 'medium' = 2, and 'high' = 3

[ ]:
```
df_original["left"].replace({"yes": 1, "no": 0}, inplace=True)
df_original["salary"].replace({"low": 1, "medium": 2, "high": 3}, inplace=True)
df_original
```

[ ]:
```
       department  promoted    review  projects  salary  tenure  satisfaction  \
0      operations         0  0.577569         3       1     5.0      0.626759
1      operations         0  0.751900         3       2     6.0      0.443679
2         support         0  0.722548         3       2     6.0      0.446823
3       logistics         0  0.675158         4       3     8.0      0.440139
4           sales         0  0.676203         3       3     5.0      0.577607
…             …       …         …       …       …        …
9535   operations         0  0.610988         4       2     8.0      0.543641
9536    logistics         0  0.746887         3       2     8.0      0.549048
9537   operations         0  0.557980         3       1     7.0      0.705425
9538           IT         0  0.584446         4       2     8.0      0.607287
9539      finance         0  0.626373         3       1     7.0      0.706455

       bonus  avg_hrs_month  left
0          0     180.866070     0
1          0     182.708149     0
```

```
2         0     184.416084     0
3         0     188.707545     0
4         1     179.821083     0
…       …              …    …
9535      0     188.155738     1
9536      0     188.176164     1
9537      0     186.531008     1
9538      1     187.641370     1
9539      1     185.920934     1

[9540 rows x 10 columns]
```

2 Department-wise statistics

```
[ ]: import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
fig, ((ax1, ax2, ax3), (ax4, ax5, ax6)) = plt.subplots(2, 3)
fig.suptitle("DEPARTMENT STATISTICS")
fig.set_figheight(10)
fig.set_figwidth(15)
for tick in ax1.get_xticklabels():
    tick.set_rotation(60)
# ax1.set_title("Number of employees in each department")
ax1.set(xlabel='', ylabel='# employees')
ax1.bar(df_original['department'].value_counts().index,␣
 ↪df_original['department'].value_counts().values)

for tick in ax2.get_xticklabels():
    tick.set_rotation(60)
df_original["left"].replace({"yes": 1, "no": 0}, inplace=True)
# ax2.set_title("Departmentwise salary")
ax2.set(xlabel='', ylabel='AVERAGE SALARY')
ax2.errorbar(df_original.groupby('department', as_index=False).
 ↪mean()['department'],
             df_original.groupby('department', as_index=False).mean()['salary'],
             yerr = df_original.groupby('department', as_index=False).
 ↪std()['salary'], fmt='o', capsize=3, ecolor = 'red')
ax2.set_ylim([0,4])

for tick in ax3.get_xticklabels():
    tick.set_rotation(60)
# ax3.set_title("Employee reviews in each department")
ax3.set(xlabel='', ylabel='AVERAGE REVIEW')
ax3.errorbar(df_original.groupby('department', as_index=False).
 ↪mean()['department'],
```

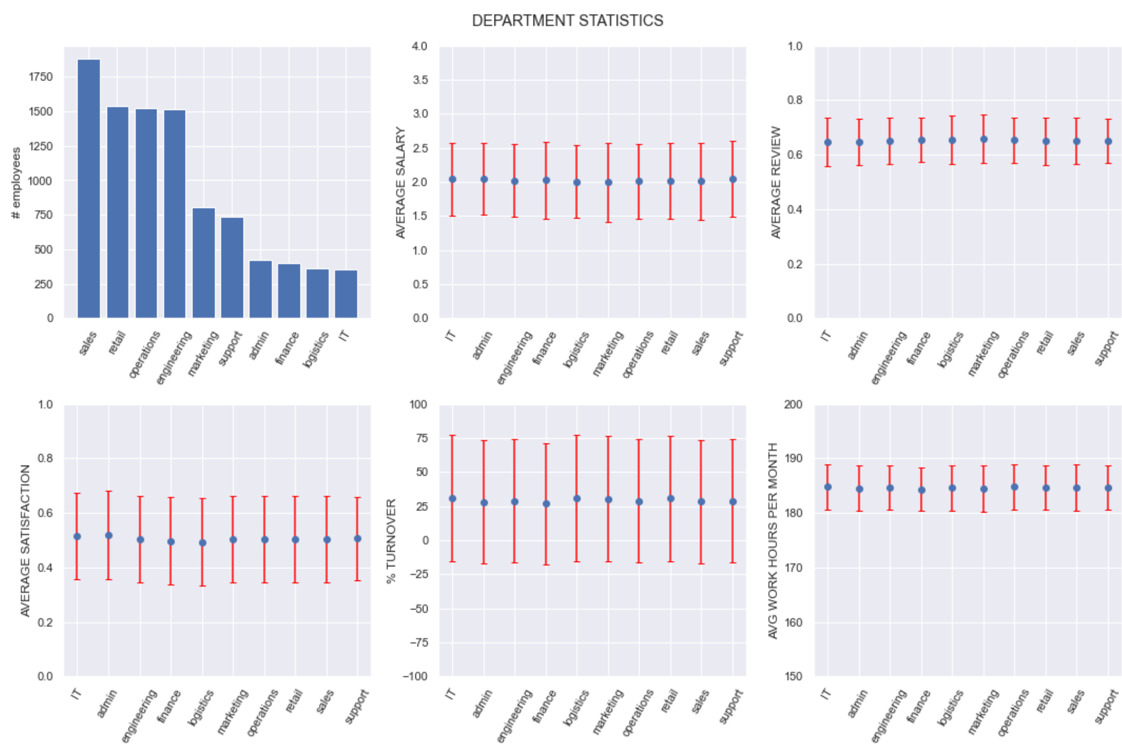
```
             df_original.groupby('department', as_index=False).mean()['review'],
             yerr = df_original.groupby('department', as_index=False).
 ↪std()['review'], fmt='o', capsize=3, ecolor = 'red')
ax3.set_ylim([0, 1])

for tick in ax4.get_xticklabels():
    tick.set_rotation(60)
# ax4.set_title("Employee satisfaction in each department")
ax4.set(xlabel='', ylabel='AVERAGE SATISFACTION')
ax4.errorbar(df_original.groupby('department', as_index=False).
 ↪mean()['department'],
             df_original.groupby('department', as_index=False).
 ↪mean()['satisfaction'],
             yerr = df_original.groupby('department', as_index=False).
 ↪std()['satisfaction'], fmt='o', capsize=3, ecolor = 'red')
ax4.set_ylim([0, 1])

df_original["left"].replace({"yes": 1, "no": 0}, inplace=True)
for tick in ax5.get_xticklabels():
    tick.set_rotation(60)
# ax5.set_title("% of employees who left")
ax5.set(xlabel='', ylabel='% TURNOVER')
ax5.errorbar(df_original.groupby('department', as_index=False).
 ↪mean()['department'],
             df_original.groupby('department', as_index=False).
 ↪mean()['left']*100,
             yerr = df_original.groupby('department', as_index=False).
 ↪std()['left']*100, fmt='o', capsize=3, ecolor = 'red')
ax5.set_ylim([-100, 100])

for tick in ax6.get_xticklabels():
    tick.set_rotation(60)
df_original["left"].replace({"yes": 1, "no": 0}, inplace=True)
# ax6.set_title("Average hours the employee worked in a month")
ax6.set(xlabel='', ylabel='AVG WORK HOURS PER MONTH')
ax6.errorbar(df_original.groupby('department', as_index=False).
 ↪mean()['department'],
             df_original.groupby('department', as_index=False).
 ↪mean()['avg_hrs_month'],
             yerr = df_original.groupby('department', as_index=False).
 ↪std()['avg_hrs_month'], fmt='o', capsize=3, ecolor = 'red')
ax6.set_ylim([150, 200])

plt.tight_layout()

print("% of employees leaving the company from each department:")
```

```
temp = df_original.groupby('department', as_index=False).mean()
temp["left (%)"] = 100 * temp["left"]
temp[['department', 'left (%)']].sort_values(by='left (%)')
```

```
% of employees leaving the company from each department:
```

[ ]:

	department	left (%)
3	finance	26.865672
1	admin	28.132388
8	sales	28.518322
6	operations	28.646518
2	engineering	28.825858
9	support	28.843537
5	marketing	30.299252
7	retail	30.564568
4	logistics	30.833333
0	IT	30.898876



From the plots above (with avg +- std of different quantities) we see that there is **no significant visible variation** between different *'department'* that contributes to an employee leaving. The department with the highest and lowest turnover are (per employee in the department): **'IT'** and **'finance'**

3 Finding variables that are good predictors of employee departure

In order to find the better predictors of employee departure, we look at three popular feature selection techniques: - Pearson's correlation coefficient - Analysis of variance (ANOVA) f-test - Mutual information (MI)

We start by splitting the department into numerical values using one-hot encoding.

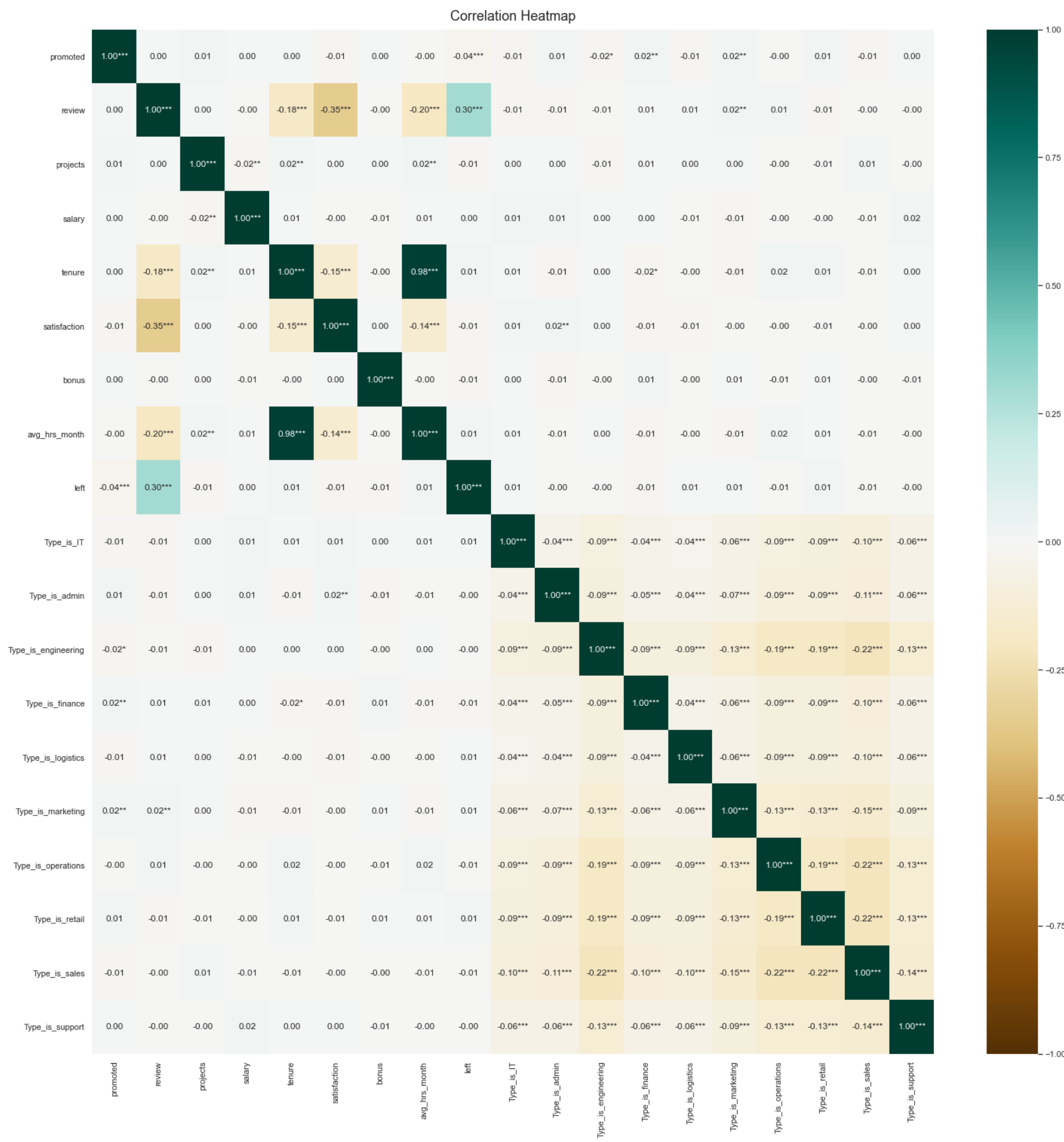
[ ]:
```
temp = pd.get_dummies(df_original, columns=["department"], prefix=["Type_is"])
print("After onehot encoding, this is the new dataframe:\n")
temp.head()
```

```
After onehot encoding, this is the new dataframe:
```

[ ]:
```
   promoted    review  projects  salary  tenure  satisfaction  bonus  \
0         0  0.577569         3       1     5.0      0.626759      0
1         0  0.751900         3       2     6.0      0.443679      0
2         0  0.722548         3       2     6.0      0.446823      0
3         0  0.675158         4       3     8.0      0.440139      0
4         0  0.676203         3       3     5.0      0.577607      1

   avg_hrs_month  left  Type_is_IT  Type_is_admin  Type_is_engineering  \
0     180.866070     0           0              0                    0
1     182.708149     0           0              0                    0
2     184.416084     0           0              0                    0
3     188.707545     0           0              0                    0
4     179.821083     0           0              0                    0

   Type_is_finance  Type_is_logistics  Type_is_marketing  Type_is_operations  \
0                0                  0                  0                   1
1                0                  0                  0                   1
2                0                  0                  0                   0
3                0                  1                  0                   0
4                0                  0                  0                   0

   Type_is_retail  Type_is_sales  Type_is_support
0               0              0                0
1               0              0                0
2               0              0                1
3               0              0                0
4               0              1                0
```

[ ]:
```
from sklearn.feature_selection import SelectKBest, mutual_info_classif, chi2
from scipy.stats import pearsonr
import numpy as np
```

```python
X = temp.drop(columns=['left'])
Y = temp['left']
results = temp.corr().to_numpy()
pval = temp.corr(method=lambda x, y: pearsonr(x, y)[1]) - np.eye(*results.shape)
p = pval.applymap(lambda x: ''.join(['*' for t in [0.01,0.05,0.1] if x<=t]))
plt.figure(figsize=(25, 25))
strings = p.to_numpy()
labels = (np.asarray(["{1:.2f}{0}".format(string, value)
                      for string, value in zip(strings.flatten(),
                                                 results.flatten())])
         ).reshape(p.to_numpy().shape)
heatmap = sns.heatmap(temp.corr(), vmin=-1, vmax=1, annot=labels, fmt="",␣
↪cmap='BrBG')
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':18}, pad=12);
plt.show()

fs = SelectKBest(k='all')
fs.fit(X, Y);
print('\033[1m'+'Ranking features based on ANOVA F-value (based on␣
↪predictability of target variable \'left\'):'+'\033[0m')
for i in range(len(fs.scores_)):
    print('Feature \'%s\' score: %f and p-value: %f' % (temp.columns[i], fs.
↪scores_[i], fs.pvalues_[i]))

fs = mutual_info_classif(X, Y)
print('\033[1m'+"\nMutual Information between features and target variable␣
↪\'left\':"+'\033[0m')
for i in range(len(fs)):
    print('Feature \'%s\' MI: %f' % (temp.columns[i], fs[i]))
```



Ranking features based on ANOVA F-value (based on predictability of target variable 'left'):

Feature 'promoted' score: 12.918187 and p-value: 0.000327
Feature 'review' score: 973.292239 and p-value: 0.000000
Feature 'projects' score: 1.468616 and p-value: 0.225594
Feature 'salary' score: 0.008485 and p-value: 0.926609
Feature 'tenure' score: 1.055947 and p-value: 0.304167
Feature 'satisfaction' score: 0.901350 and p-value: 0.342444
Feature 'bonus' score: 1.258221 and p-value: 0.262016
Feature 'avg_hrs_month' score: 0.773990 and p-value: 0.379008
Feature 'left' score: 0.527130 and p-value: 0.467834

```
Feature 'Type_is_IT' score: 0.236088 and p-value: 0.627057
Feature 'Type_is_admin' score: 0.110843 and p-value: 0.739194
Feature 'Type_is_engineering' score: 1.089849 and p-value: 0.296531
Feature 'Type_is_finance' score: 0.493335 and p-value: 0.482461
Feature 'Type_is_logistics' score: 0.528422 and p-value: 0.467289
Feature 'Type_is_marketing' score: 0.251581 and p-value: 0.615976
Feature 'Type_is_operations' score: 1.698900 and p-value: 0.192463
Feature 'Type_is_retail' score: 0.500536 and p-value: 0.479282
Feature 'Type_is_sales' score: 0.044236 and p-value: 0.833419


Mutual Information between features and target variable 'left':
Feature 'promoted' MI: 0.007114
Feature 'review' MI: 0.062089
Feature 'projects' MI: 0.002376
Feature 'salary' MI: 0.000000
Feature 'tenure' MI: 0.037707
Feature 'satisfaction' MI: 0.000830
Feature 'bonus' MI: 0.001912
Feature 'avg_hrs_month' MI: 0.059985
Feature 'left' MI: 0.000000
Feature 'Type_is_IT' MI: 0.000000
Feature 'Type_is_admin' MI: 0.003746
Feature 'Type_is_engineering' MI: 0.000000
Feature 'Type_is_finance' MI: 0.000000
Feature 'Type_is_logistics' MI: 0.000000
Feature 'Type_is_marketing' MI: 0.000000
Feature 'Type_is_operations' MI: 0.003914
Feature 'Type_is_retail' MI: 0.000000
Feature 'Type_is_sales' MI: 0.000000
```

The first plot shows the correlations (with the '*' denoting p-value<0.1, '**' denoting p-value<0.05 and finally '***' denoting p-value<0.01 repsectively) between the different features (*'department', 'promoted', 'review', 'projects', 'salary', 'tenure', 'satisfaction', 'bonus', 'avg_hrs_month'*) and also to the target variable (*'left'*). The values are between -1 and 1. Positive values indicate a positive correlation and vice-versa. The higher the value (unsigned) the more relevant the feature is. From this plot, we find **the two features *'review'* and *'promoted'* are most relevant to the target variable *'left'***.

The next statistical measure we looked at is the **ANOVA F-value** (the higher the score the better). We find that again **the features *'review'* and *'promoted'* to be the most relevant** with statistical significance (p-value < 0.05).

Finally, we calculate the **Mutual Information** between the features and the target variable. This measures the dependence between variables. The higher the value the higher the dependency. This also **corroborates the conclusion from previous tests that the most relevant feature to be *'review'*** (even though the overall ranking differs).

From the correlation heatmap we observe that the variables '*review*' is negatively correlated to variables '*satisfaction*', '*tenure*', and '*avg_hrs_month*' and '*review*' is postively correlated to the target variable '*left*'. This seems to suggest, albeit counter intuitively, that the **higher the employee's composite score received in their last evaluation, the more likely they are to leave**. In order to understand this better, we look more closely at the *'review'* feature.

Finally this statistical analysis confirms our observation that the variable *'department'* is not a good predictor of an employee leaving.

4 Using Random Forests to identify the most important predictors of an employee leaving

We first split the dataset into training set (80%) and testing set (20%)

[ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.20)
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(max_depth = 13, class_weight='balanced')
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
Y1_pred = classifier.predict(X_train)
print("Printing the confusion matrix ((True Positive, False Positive),(False␣
 ↪Negative, True Negative)) for the training set:\n",␣
 ↪confusion_matrix(Y_train, Y1_pred))
print("Printing the statistics corresponding to the class('left' = 1) and␣
 ↪class('left' = 0) for training set:\n", classification_report(Y_train,␣
 ↪Y1_pred))
print("Printing the confusion matrix ((True Positive, False Positive),(False␣
 ↪Negative, True Negative)) for the testing set:\n", confusion_matrix(Y_test,␣
 ↪Y_pred))
print("Printing the statistics corresponding to the class('left' = 1) and␣
 ↪class('left' = 0) for testing set:\n", classification_report(Y_test, Y_pred))
importance = classifier.feature_importances_
# summarize feature importance
for i,v in enumerate(importance):
        print('Feature: %s, Importance score: %.5f' % (X.columns[i],v))
```

```
Printing the confusion matrix ((True Positive, False Positive),(False Negative,
True Negative)) for the training set:
 [[5079  321]
 [ 148 2084]]
Printing the statistics corresponding to the class('left' = 1) and class('left'
= 0) for training set:
               precision    recall  f1-score   support

           0       0.97      0.94      0.96      5400
           1       0.87      0.93      0.90      2232
```

```
    accuracy                           0.94      7632
   macro avg       0.92      0.94      0.93      7632
weighted avg       0.94      0.94      0.94      7632

Printing the confusion matrix ((True Positive, False Positive),(False Negative,
True Negative)) for the testing set:
 [[1214  142]
 [ 122  430]]
Printing the statistics corresponding to the class('left' = 1) and class('left'
= 0) for testing set:
              precision    recall  f1-score   support

           0       0.91      0.90      0.90      1356
           1       0.75      0.78      0.77       552

    accuracy                           0.86      1908
   macro avg       0.83      0.84      0.83      1908
weighted avg       0.86      0.86      0.86      1908

Feature: promoted, Importance score: 0.00384
Feature: review, Importance score: 0.25926
Feature: projects, Importance score: 0.01590
Feature: salary, Importance score: 0.01402
Feature: tenure, Importance score: 0.11519
Feature: satisfaction, Importance score: 0.25729
Feature: bonus, Importance score: 0.00838
Feature: avg_hrs_month, Importance score: 0.28705
Feature: Type_is_IT, Importance score: 0.00291
Feature: Type_is_admin, Importance score: 0.00307
Feature: Type_is_engineering, Importance score: 0.00476
Feature: Type_is_finance, Importance score: 0.00336
Feature: Type_is_logistics, Importance score: 0.00301
Feature: Type_is_marketing, Importance score: 0.00393
Feature: Type_is_operations, Importance score: 0.00476
Feature: Type_is_retail, Importance score: 0.00477
Feature: Type_is_sales, Importance score: 0.00487
Feature: Type_is_support, Importance score: 0.00362
```

Above we trained a Random Forest with class weighting and rank the features that best predict an employee leaving. We again find that the best features that predict an employee leaving to be ‘*review*’, ‘*statisfaction*’ and ‘*avg_hrs_month*’.

5 Conclusion and recommendations

- The department with the highest and lowest turnover are (per employee in the department): **‘IT’** and **‘finance’**
- The variable ‘**review**’ was highly correlated with the target ‘**left**’. This suggests people who score high on their last evaluation, also ended up leaving. Hence trying to identify what

makes employees dissatisfied even though they scored high on their reviews could help reduce emplyee turnover.

- We also found negative correlations between '**review**' with '**satisfaction**', '**avg_hrs_month**', and '**tenure**'. This suggests people who scored high on their last evaluation – also are more likely to be dissatisfied, and worked fewer hrs on average every month and tend to have been relatively newer employees. Since we did not find any significant correlations between ('**review**', '**avg_hrs_month**', and '**tenure**') with variables such as '**promotion**', '**bonus**' or '**salary**' (although since we were given only salary ranges – low, medium and high it is possible that a salary increase was offered but still ended up being in the same bracket hence mudling this correlation result), a possibility could be that those employees scored higher on their evaluations (possibly due to being relatively newer employees and in hopes of getting more incentives) ended up feeling they were not rightly compensated hence leading them to work less and be more dissatisfied. So trying to address this might improve their morale.
- We also found large positive correlations between '*avg_hrs_month*' and '*tenure*' suggesting that people who have been working with the company for longer tend to work for longer hrs on average. But interestingly due to the negative correlation between '*satisfaction*' with '**tenure**' and '**avg_hrs_month**' suggests they tend to be more dissatisfied. This corroborates the previous point.