Employee Turnover

February 3, 2022

1 Can you help reduce employee turnover?

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.

1.1 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.
- "avg_hrs_month" - the average hours the employee worked in a month.
- "left" - "yes" if the employee ended up leaving, "no" otherwise.

[1]:

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

[1]:

```
   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
```

1.2 Report

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]:

```python
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from scipy.stats import skew
from sklearn.decomposition import PCA

df = pd.read_csv('./data/employee_churn_data.csv') # (9540, 10)

#These columns are numeric:
cols=['promoted', 'review', 'projects', 'tenure',
'satisfaction','bonus','avg_hrs_month']

y=df['left'].replace({'no': 0, 'yes': 1}).astype(int)
df=df.filter(cols).dropna()  #[9540 rows x 7 columns] #no missing values
```

2 Multicollinearity and Heatmap

- The heatmap shows a .98 correlation between avg_hrs_month and tenure.
- This was the only major Multicollinearity detected (above .80).

- Therefore, one of these variables should likely be removed.

[2]:

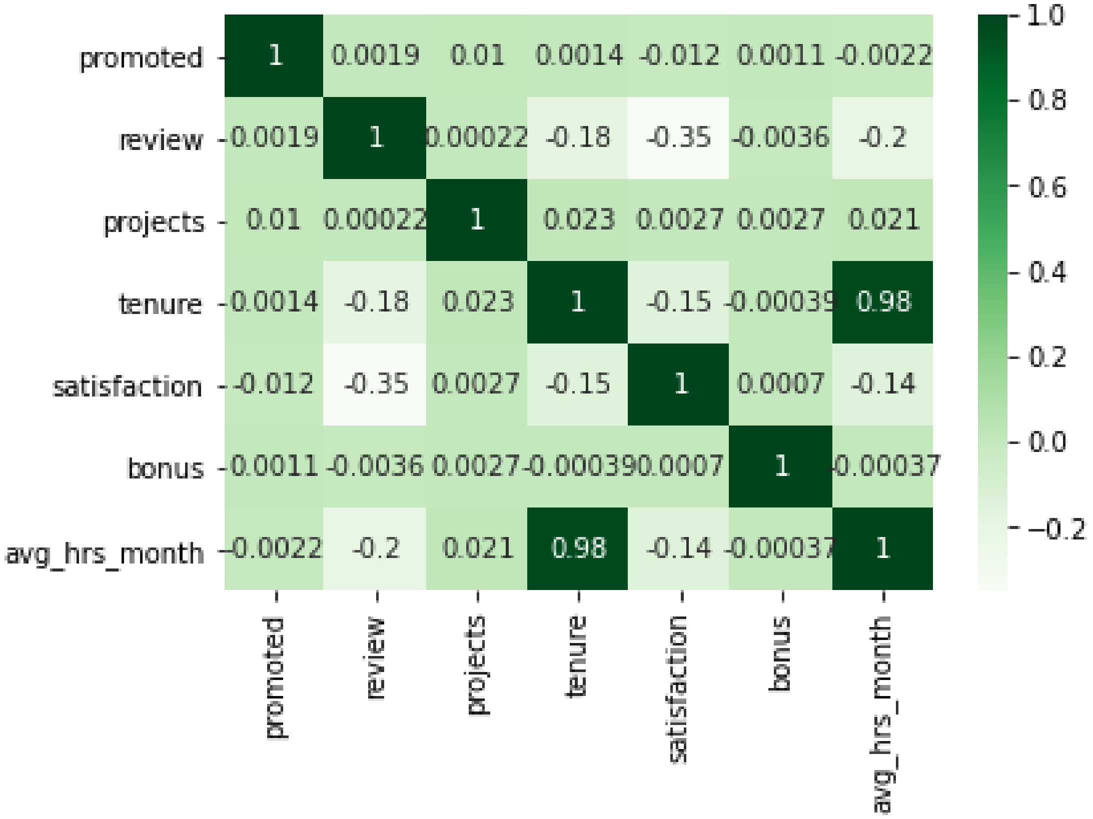
```python
def printHeat():
    Var_Corr = df.corr()
    # plot the heatmap and annotation on it
    heat=sns.heatmap(Var_Corr, xticklabels=Var_Corr.columns,␣
 ↪yticklabels=Var_Corr.columns, cmap='Greens', annot=True)
    print(heat)

#Calculating VIF
from statsmodels.stats.outliers_influence import variance_inflation_factor

def calc_vif(X):
    vif = pd.DataFrame()
    vif["variables"] = X.columns
    vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.
↪shape[1])]
    return(vif)

#calc=calc_vif(df[cols])
printHeat()
```

```
AxesSubplot(0.125,0.125;0.62x0.755)
```



3 Correlations

1. Normal correlation using .corr in Python.
2. Pearson's correlation assesses linear relationships between two continuous variables.
3. Correlationn coefficient represents the linear dependence of two variables or sets of data.

[ ]:

```
def printCor():
    dCor={} #correlation dictionary
    dPear={} #pearson dictionary
    dCov={} #covariance correlation dictionary
    for col in cols:
        x=df[col] # each individual column

        #Correlation:
        cor=y.corr(x).round(3)
        dCor[col]=cor

        #pearson correlation:
        pear=np.corrcoef(list(x), list(y))[0, 1].round(3)
        dPear[col]=pear

        #covariance correlation
        cov=np.cov(x,y)[0][1].round(3)
        dCov[col]=cov

    #Sort dictionaries by highest correlation:
    dCor=sorted(((v, k) for k, v in dCor.items()), reverse=True)
    dPear=sorted(((v, k) for k, v in dPear.items()), reverse=True)
    dCov=sorted(((v, k) for k, v in dCov.items()), reverse=True)

    print("Calculating Correlations:")
    for d in dCor:
        print(d)
    print("")

        print("Calculate Pearson Correlation:")
    for p in dPear:
        print(p)
    print("")

    print("Covariance correlation:")
    for c in dCov:
        print(c)
```

4 LOGISTIC REGRESSION

AUC is 0.7153 using logistic regression.

[ ]:

```
from sklearn.model_selection import train_test_split # splitting the data
from sklearn.linear_model import LogisticRegression # model algorithm
from sklearn import metrics

#Split the data set into x and y data:
y_data = np.asarray(y)
x_data = df[cols]
```

```
#split the dataset into training (70%) and testing (30%) sets:
from sklearn.model_selection import train_test_split
x_training_data, x_test_data, y_training_data, y_test_data =␣
 ↪train_test_split(x_data, y_data, test_size = 0.3, random_state=42)
#Logistic regression defaults to L2

#Create the model
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()

#Train the model and create predictions
model.fit(x_training_data, y_training_data)
predictions = model.predict(x_test_data)
#print(predictions)

#GENERATE PREDICTION:
results=[] #store results that will later be turned into a DF.

#Calculate performance metrics
from sklearn.metrics import classification_report
#print(classification_report(y_test_data, predictions))

#Generate a confusion matrix
from sklearn.metrics import confusion_matrix, roc_auc_score

#use model to predict probability that given y value is 1:
y_pred_proba = model.predict_proba(x_test_data)[::,1]

#calculate AUC of model
results=[]
auc = round( metrics.roc_auc_score(y_test_data, y_pred_proba), 4 )
print("AUC is: ", auc, " using logistic regression.")
results.append(auc) #going to store results in a list
```

```
AUC is:  0.7153  using logistic regression.
```

4.1 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.

|