Can we predict in how many months an employee will leave?

Evan Katz

Idea!

Turnover on average costs a company 15% of an employee's annual salary!

Predicting when an employee would leave would allow for businesses to be more proactive rather than reactionary when it comes to Human Resources related problems. This saves money and productivity and a happier labor pool.

There is always the issue of inherent bias being introduced into the model. In this case, it's more sensitive since it's using employee personal data such as gender. The goal is to ensure no bias is being introduced into the decision making.

There are examples of companies using artificial intelligence in aiding recruiting (i.e. Amazon had to pull a bot it created due to gender bias in reviewing resumes).

Where's the data?

Pull human resources data from HRIS to gather data points to create a model to predict turnover.

The data used is termination data spanning back to 2008, **1,815 rows of data**.

Data is as accurate as those who inputted into the system (Workday).

The target variable is **how many months will someone work** before terminated (regardless of voluntary and involuntary). The fields used to drive the model include, gender, compa-ratio, location, job profile, job title, years of service, is a manager.

The data includes all terminated employees since the initiation of Workday.

Applying different methods - Scoring

Linear Regression

Training score: 0.8023781689762953

Testing score: -0.23789832612945383

Decision Tree

Score: 0.30417212999643517

Random Forest Regressor

Training score: 0.903877047758696

Testing score: 0.21416342234659203

Both train test split and KFold (when applicable) were used on each of the above

Model Outcome - Scoring

The fact that the model does substantially worse on the test set suggests that it is too flexible -- it is capturing random variations that happen to be present in the test set as well as the real underlying patterns that generalize beyond it.

Solutions would include to make the model more simpler, add more data points to the data set, and add additional variables.

Random Forest Regressor - Best fit?

Random Forest Regressor Attempt w/ Train Test Split (and w/ KFold) has the best performance when comparing against the other modeling techniques. The training set score was 90%, but the test set score was 21%.

This shows that the model is currently overfitting, and more engineering is required. At this point, I do not believe it's easy to make the model a better fit without additional steps.

Next Steps:

- Introduce greater data set
- Other variables that might be correlated