The models are implemented both in Python and R
Slides 2: general information about the data set
Target: TERMINATION STATUS(A or T); this is a binary classification problem
Features provided in the data set:

```
'EMP_ID', 'ANNUAL_RATE', 'HRLY_RATE', 'JOBCODE', 'ETHNICITY', 'SEX',
'MARITAL_STATUS', 'JOB_SATISFACTION', 'AGE', 'NUMBER_OF_TEAM_CHANGED',
'REFERRAL_SOURCE', 'HIRE_MONTH', 'REHIRE', 'TERMINATION_YEAR',
'IS_FIRST_JOB', 'TRAVELLED_REQUIRED', 'PERFORMANCE_RATING',
'DISABLED_EMP', 'DISABLED_VET', 'EDUCATION_LEVEL', 'STATUS',
'JOB_GROUP', 'PREVYR_1', 'PREVYR_2', 'PREVYR_3', 'PREVYR_4', 'PREVYR_5'
```

These are potential features that could be used to predict the target variable ‘STATUS’.

First of all, 'TERMINATION_YEAR' is perfectly related to ‘STATUS’, so it is removed from feature space.

‘EMP_ID’ is the ID of the employee, we don’t expect this has useful information to predict the target variable. So ‘EMP_ID’ is also ignored.

In general, 'JOBCODE' should be related to 'JOB_GROUP'. Since ‘JOB_GROUP’ provides more information comparing to particular ‘JOBCODE’, we will use ‘JOB_GROUP’ and ignore ‘JOBCODE’.

In this data set, there are 9612 observations; 5394 are still active, and 4218 are terminated.
Two classes are relatively balanced; no need to take extra steps to deal with imbalanced classes.
We split the train and test using 8:2.

Slides 3: explorative analysis
We perform some explorative analysis to see if certain features are potentially useful.
The plot shows that different marital status does have different probability of being terminated.
This can also be seen by examining the probability of being active/terminated.

```
pd.crosstab(data.MARITAL_STATUS, data.STATUS, normalize='index')
```

STATUS	A	T
MARITAL_STATUS		
Divorced	0.558879	0.441121
Married	0.572635	0.427365
Single	0.550503	0.449497

Looks that married employee is slightly more likely to be with the company.

```
pd.crosstab(data.JOB_GROUP, data.STATUS, normalize='index')
```

STATUS	A	T
JOB_GROUP		
Accounting	0.634921	0.365079
Accounts Payable	0.793103	0.206897
Advanced Research	0.933333	0.066667
Analytical/Microbiology	0.956522	0.043478
Applied Research	0.931034	0.068966
...	...	...
Technical Packaging	0.351598	0.648402
Transportation & Warehousing	0.000000	1.000000
Treasury	1.000000	0.000000
Web	0.567568	0.432432
eCommerce	0.842857	0.157143

66 rows × 2 columns

This shows that 'JOB_GROUP' can be a useful predictor.

```
pd.crosstab(data.SEX, data.STATUS, normalize='index')
```

STATUS	A	T
SEX		
F	0.554954	0.445046
M	0.570327	0.429673

Male is slightly more likely to not terminate.

Slides 4: Model 1 – Logistic regression with L1 regularization
Before trying any model, we first coded dummy variables for all categorical variables in the data.

As we have relatively large number of features, we add L1 regularization to logistic regression.
This turns out to outperform the regular logistic regression.
The training accuracy is 71%, test accuracy is 70%. No obvious overfitting.
The heatmap shows the model performance.

Slides 5: Model 2 – random forest
We next try an ensemble model - random forest. There is no need to do feature selection for rf, since the way random forest builds the tree already does it.
To avoid overfitting, we tried different hyperparameters.
With 150 trees and the maximum depth of each tree as 12, we achieve 82% in training accuracy and 73% as test accuracy. This model is subject to overfitting. So we further decrease the numb er of trees to 120 and maximum tree depth to 11. This new model leads to training accuracy 79% with the similar test accuracy as 73%.
The heatmap provides information on the performance of the second model.

Slides 6 – Model 3 - XGBoost

We finally tried another ensemble model, which is also probably the most popular one, xgboost. Xgboost does feature selection automatically.
Unfortunately, all sets of hyperparameters we tried are subject to overfitting. The best test accuracy we have is 77%, with training accuracy 93%.

Slides 7 – Conclusion
From this series of analysis, we found that ensemble models are in general better than single model. They provide better accuracy. However, ensemble models are also easy to overfit. From our analysis, xgboost provides the best test accuracy.