CAI1C01 -
Principles of Machine Learning

PROJECT PRESENTATION

01-JUN-2018

Presented By:
Lim Yuan Her (1881120H)

Agenda

- **Introduction**
  - ❑ Background
  - ❑ Problem Statement
- **Data Preparation**
  - ❑ Data Cleaning
  - ❑ Training/Validation Data
- **Methods**
  - ❑ Algorithm Evaluation
- **Feature Selection**
  - ❑ Comparison and Evaluation
- **Algorithm Tuning**
- **Summary**
- **References**

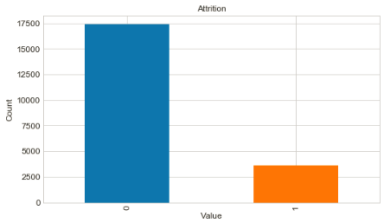
Introduction - Background

- Dataset from https://www.kaggle.com/analystanand/employee-attrition
- Consists of 2 csv (comma-separated delimiter) files
  - train.csv (25,491 records) and test.csv (4,507 records)
- Data file consists of 9 features and 1 predictor label

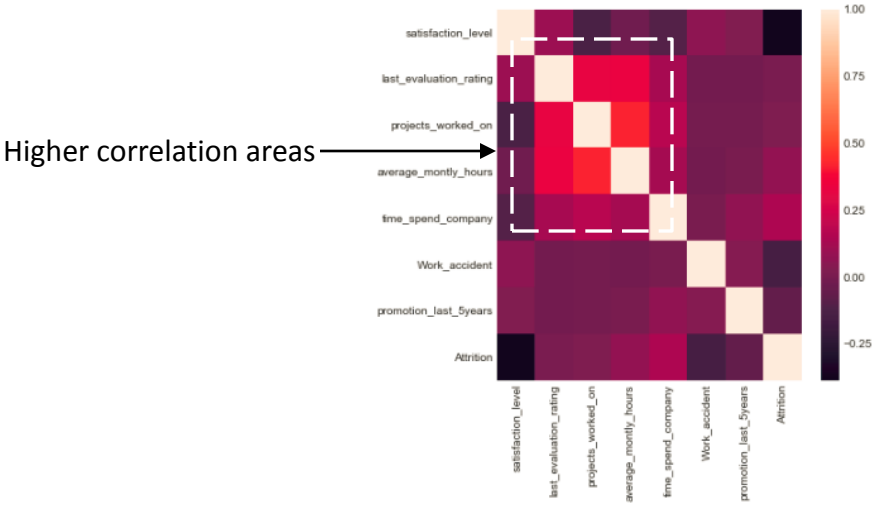
Fetaures		Predictor Label	
Name	**Data Type**	**Name**	**Data Type**
satisfaction_level	Float	Attrition	Integer
last_evaluation_rating	Float		
projects_worked_on	Integer		
average_montly_hours	Integer		
time_spend_company	Integer		
Work_accident	Integer		
promotion_last_5years	Integer		
Department	String		
salary	String		

Introduction – Problem Statement

- Objective is to predict whether employee is likely to leave the company
  - "Attrition" feature (0 – Stay, 1 – Leave)
  - "Leave" records comprise approximately 15-20% of train dataset



- Correlation Plot shows higher correlation between "projects_worked_on", "last_evaluation_rating", "average_monthly_hours" features



Data Preparation - Cleaning

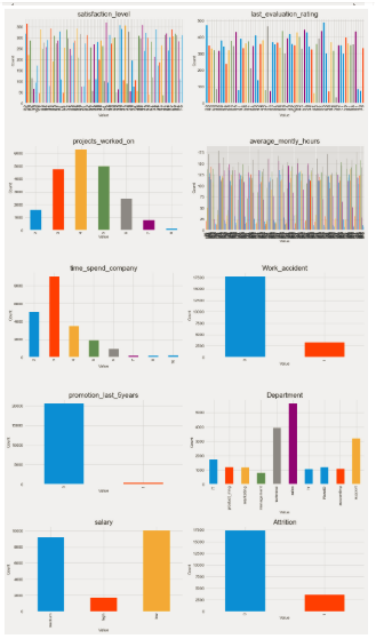
- Data Cleaning

1. Check for Missing Values
   - No records found with missing values

2. Check for duplicate records
   - 4530 duplicate records found
   - Duplicate records removed from dataset, leaving 20,961 remaining out of original 25,491 records)

3. Check for Structural Errors/ Inconsistencies
   a) Check for inconsistent values in column
      - No inconsistent values found
   b) Check no column has only 1 value
      - No inconsistent values found

4. Check for outliers
   - No outlier values found



Data Preparation – Training/Validation Data

- 7 numerical and 2 categorical features

Numerical		Categorical
satisfaction_level	time_spend_company	Department
last_evaluation_rating	Work_accident	salary
projects_worked_on	promotion_last_5years	
average_montly_hours		

- Encoding of Categorical Features
  - Use One-Hot Encoding (get_dummies() from Scikit-Learn)

- Split dataset into two, one for model development and one for validation
  - Withhold 20% of data from analysis and model development to be used to validate the final model
  - Use Train_Test_Split() from Scikit-Learn

```
# Split out dataset

# Create the X and y arrays
features_df = pd.get_dummies(df_clean.drop("Attrition", axis=1), columns=["Department", "salary"])
class_df = df_clean["Attrition"]

feature_labels = np.array(list(features_df))

X = features_df.as_matrix()
y = class_df.as_matrix()

validation_size = 0.20
seed = 7

X_train, X_validation, Y_train, Y_validation = train_test_split (X, y, test_size=validation_size,
random_state=seed)
```

Methods – Algorithm Evaluation

- Evaluate both classification algorithms (10) and ensemble method (6)

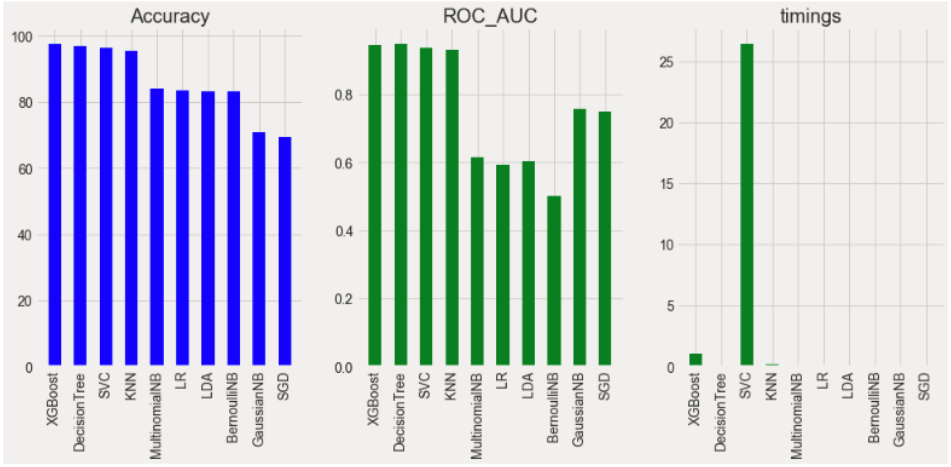
Classification		Ensemble Methods	
Logistic Regression	Bernoulli Naïve Bayes	Ada Boost	Votting
Linear Discriminant Analysis	Multinomial Naïve Bayes	Gradient Boosting	
Support Vector Classifier	Decision Tree	Random Forest	
K-Nearest Neighbors Classifier	Stochastic Gradient Descent Classifier	Extra Trees	
Gaussian Naïve Bayes	XG Boost Classifier	Bagging	

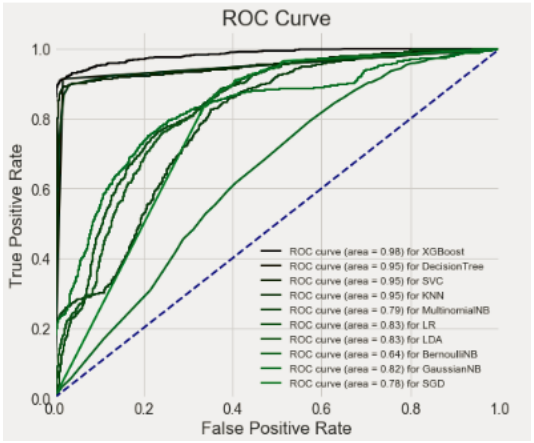
- Evaluation criteria:
  - Single
    a) Accuracy (higher is better)
    b) ROC_AUC (higher is better)
    c) # of False Positive/ False Negative (lower is better)
    d) Receiver operating characteristic (ROC) Curve
    e) Precision-Recall Curve
    f) Training Timing (lower is better, included for comparison purposes)
  - Cross-Validation
    a) Mean (higher is better)
    b) Standard Deviation (lower is better)
    c) Training Timing (lower is better, included for comparison purposes)

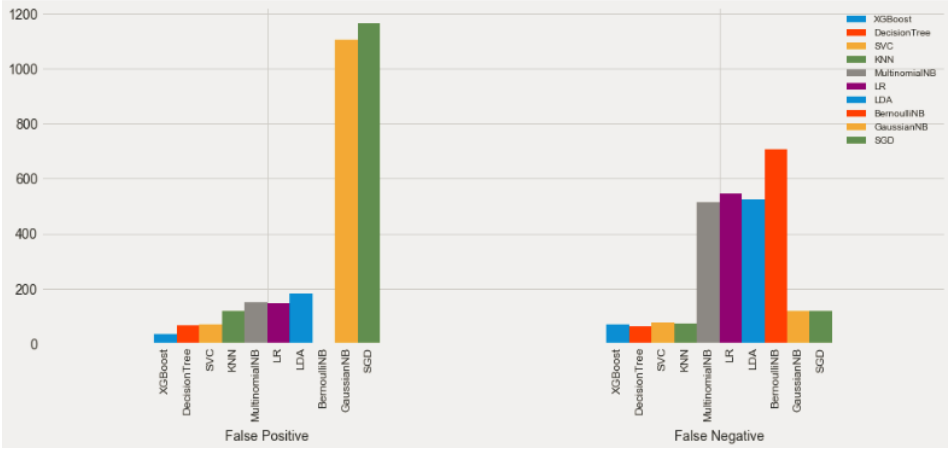
- Select 4 best classifiers based on above criteria

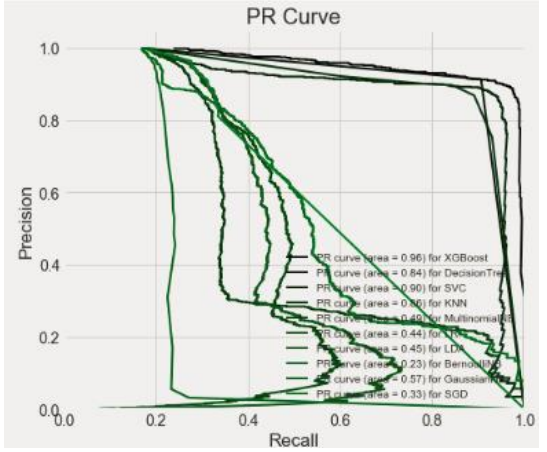
Algorithm Evaluation – Classification Algorithms

- Classification Algorithms (Best: **XGBoost**)



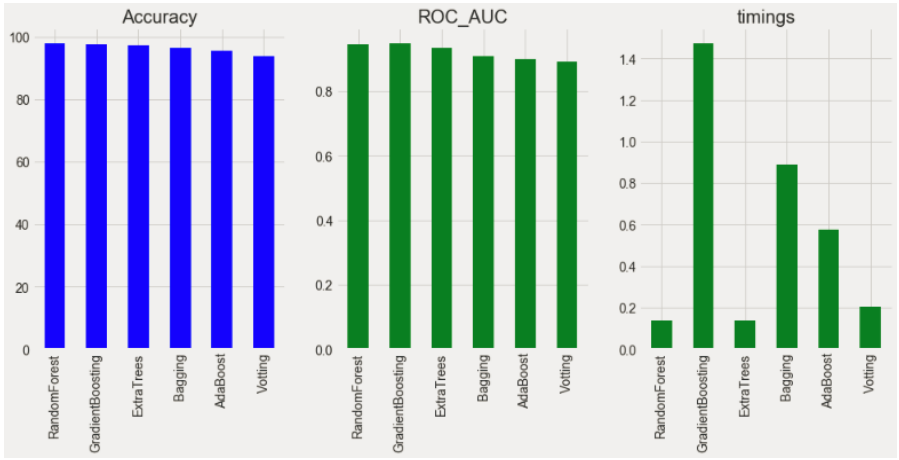


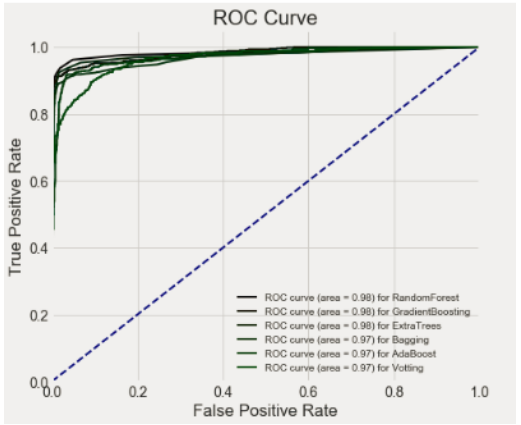


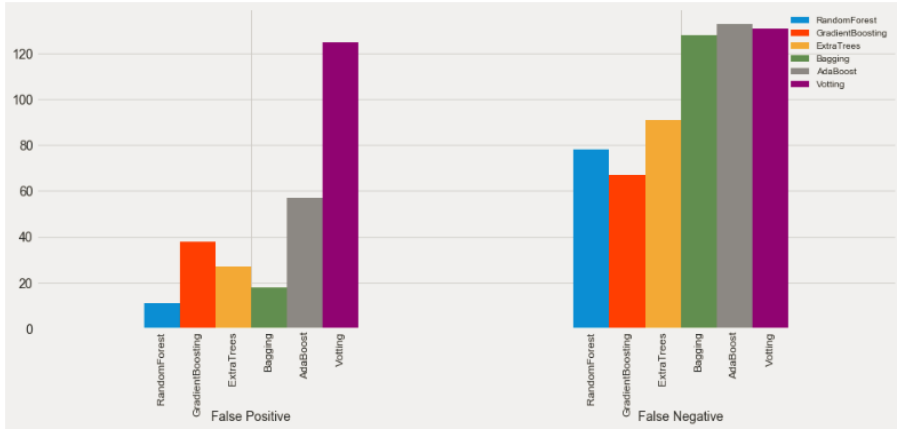


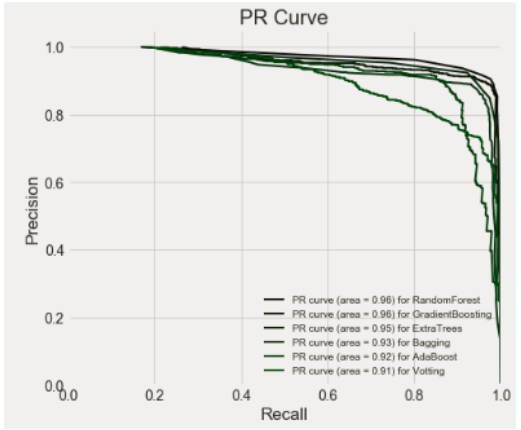
Algorithm Evaluation – Ensemble Methods

- Ensemble Methods(Best: **Random Forest**)



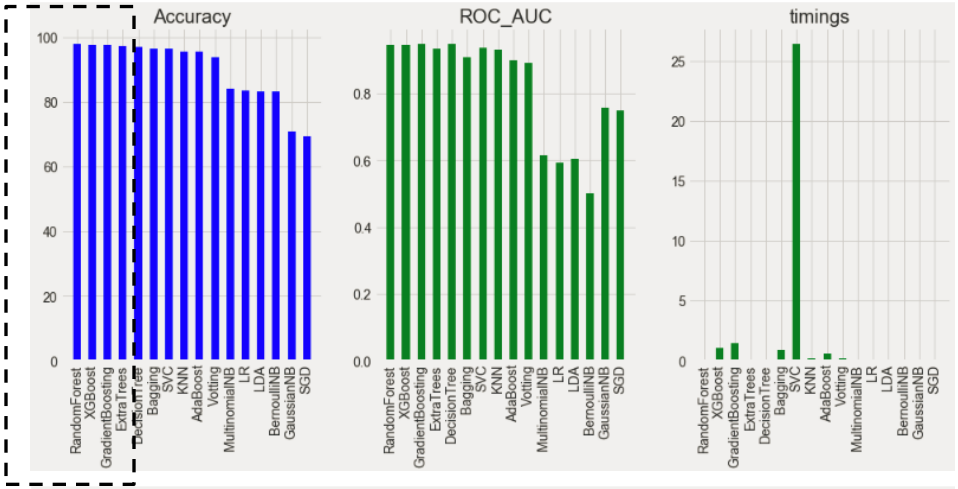


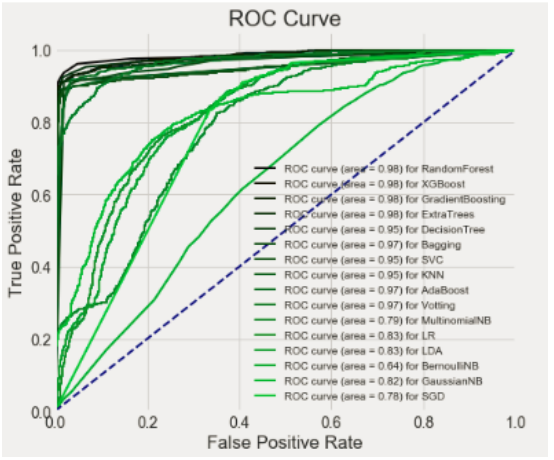


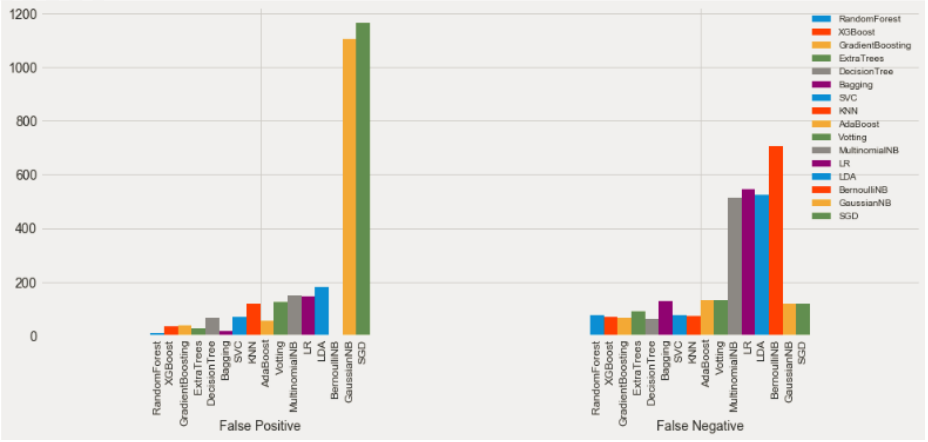


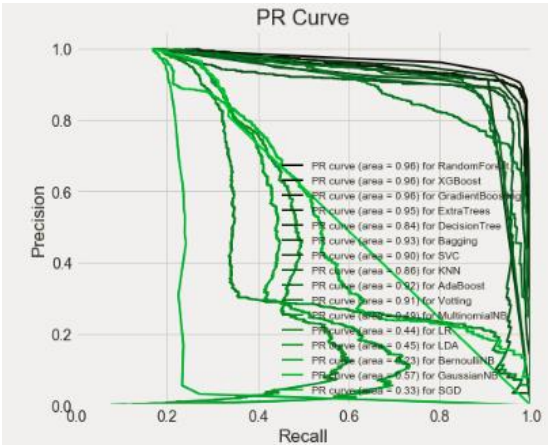
Algorithm Evaluation – All Classifiers

- Comparison of all results
  - Best 4: **Random Forest** / **XGBoost** / **Gradient Boosting** / **Extra Trees**



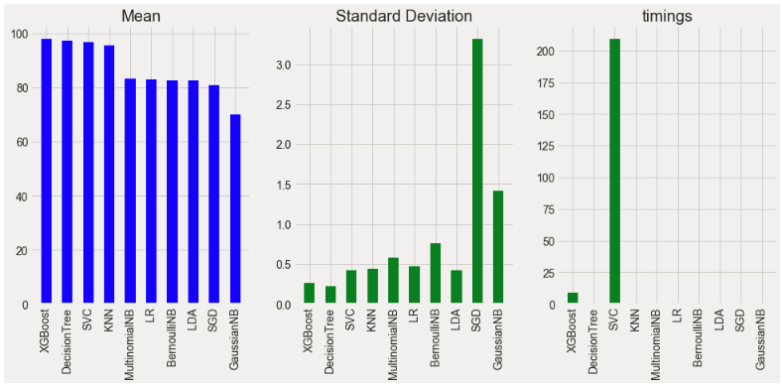




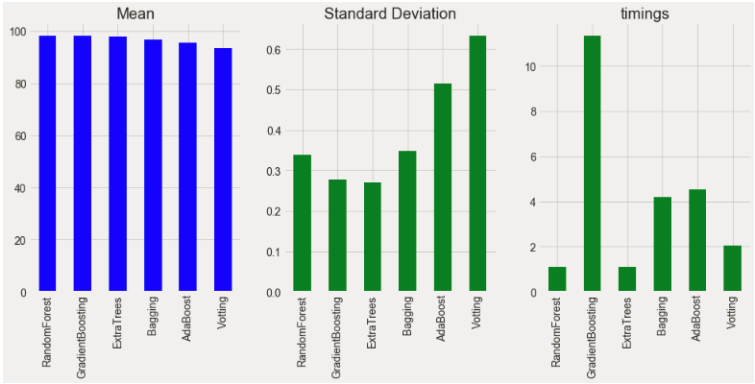


Algorithm Evaluation –Cross-Validation

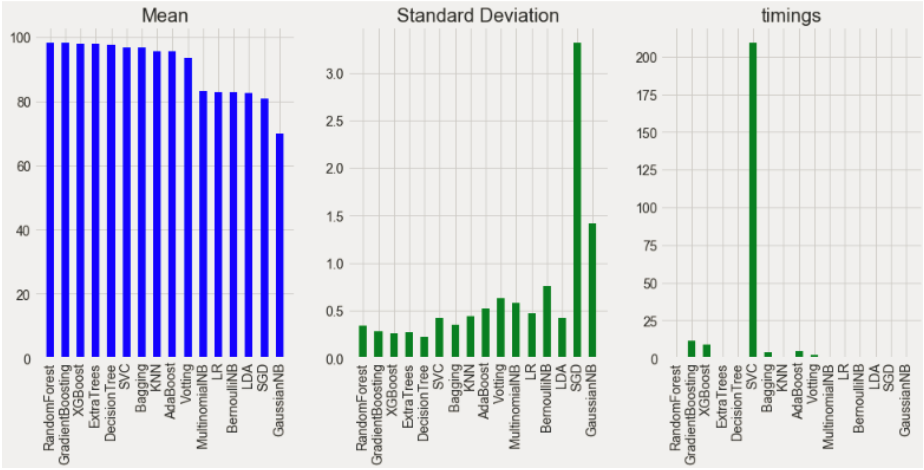
- Comparison of results from 10-fold cross validation



Best**: XGBoost** (Classification Algorithms)



Best: **Random Forest** (Ensemble Methods)



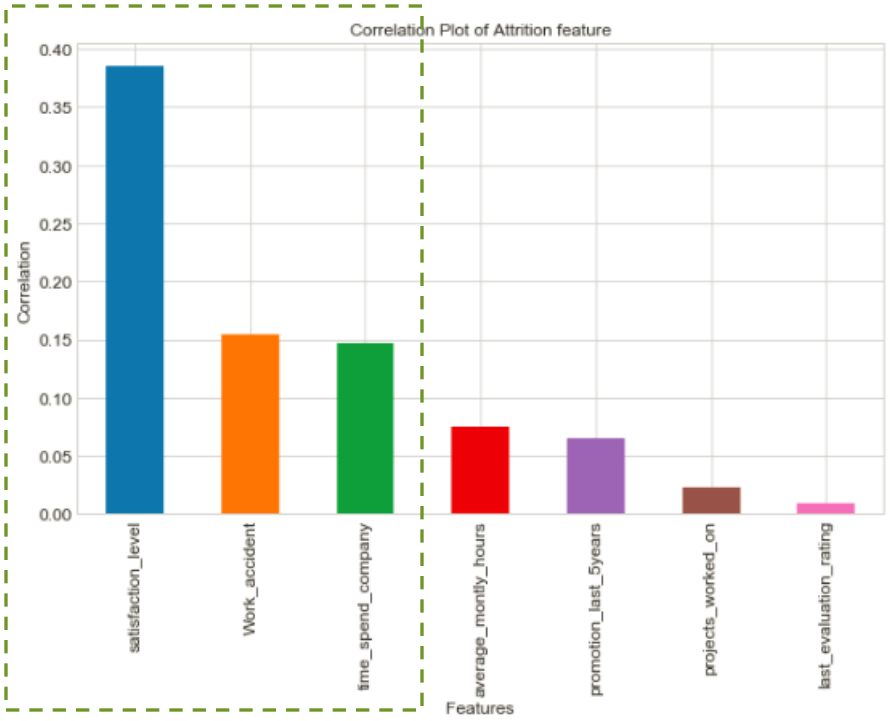
Best 4: **Random Forest** / **XGBoost** / **Gradient Boosting** / **Extra Trees**)

Feature Selection - General

- Use feature selection methods to search for feature combinations resulting in better performance for each of the 4 selected algorithms
  - Random Forest / XGBoost / Gradient Boosting / Extra Trees
- Compare performance of the 4 algorithms using their optimal feature combination to select the best for hyper-parameter tuning
- Advantages of performing feature selection include reduced overfitting, improved accuracy, less model training time
- Methods Evaluated include:
  - Manual identification of features from correlation plot
  - Univariate Selection
    - *SelectKBest class from* scikit-learn library
    - Use chi squared (chi^2) statistical test for non-negative features to select best features
  - Recursive Feature Elimination
    - Recursively removes attributes and build model on remaining attributes
    - Uses model accuracy to identify which attributes contribute most to predicting target attribute
  - Model Feature Importance
    - Uses *model.feature_importances_* for ranking features based on importance

Feature Selection – Correlation-Identified Features

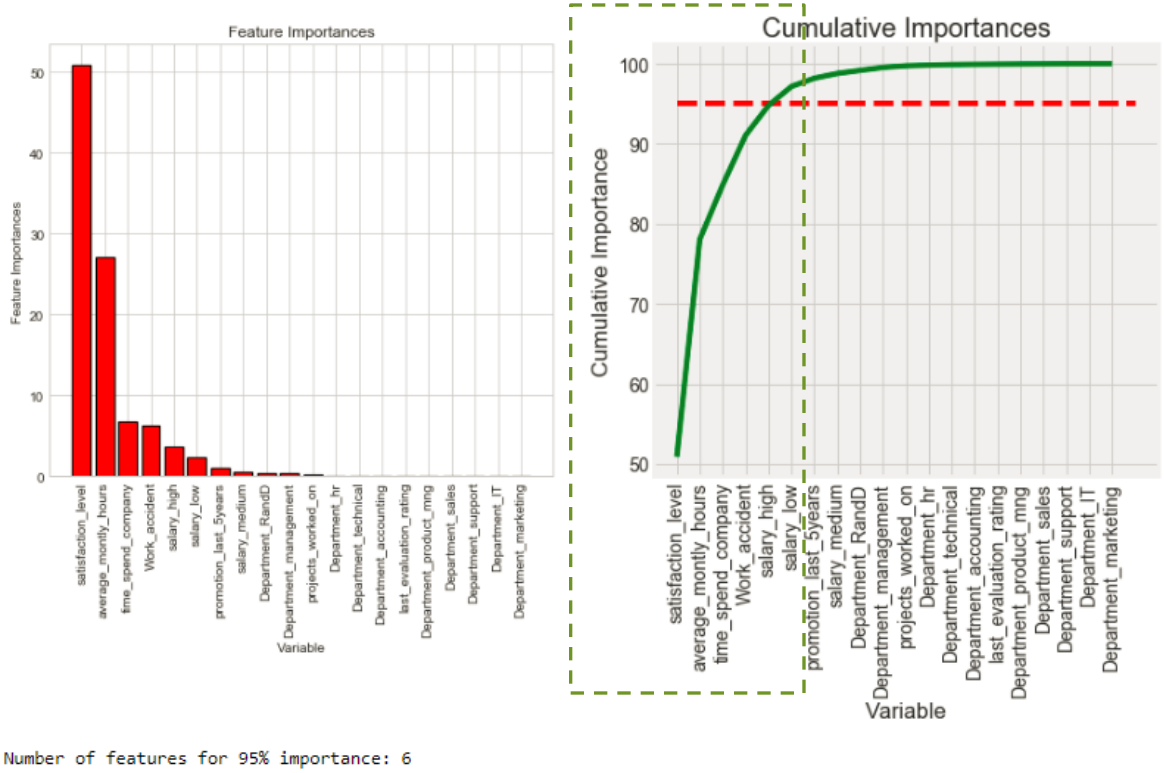
- Manual selection from "Attrition" feature correlation plot



- Top 3 features with highest correlation to "Attrition" feature selected
  - "satisfaction_level", "time_spend_company", "Work_accident"

Feature Selection – Univariate Selection

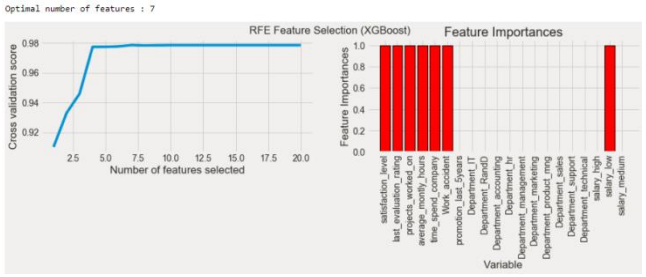
- Univariate Selection

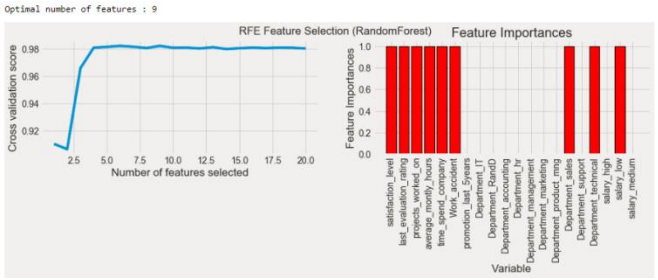


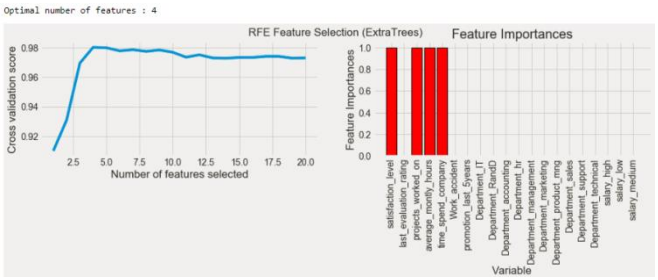
- 6 features contribute to 95% cumulative importance
  - "satisfaction_level", "average_monthly_hours", "time_spend_company", "Work_accident", "salary_high", "salary_low"

Feature Selection – RFE

- Recursive Feature Elimination



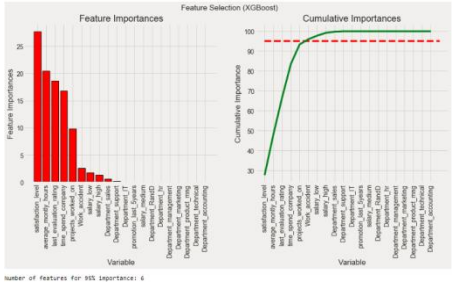


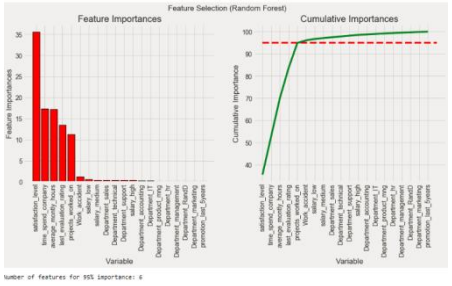


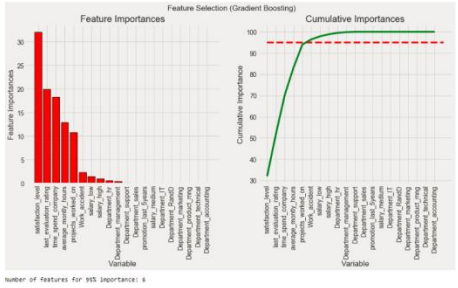
Classifier	95% Importance Threshold	Features
XG Boost	7	satisfaction_level, projects_worked_on, average_monthly_hours, time_spend_company, last_evaluation_rating, Work_accident, salary_low
Random Forest	9	satisfaction_level, projects_worked_on, average_monthly_hours, time_spend_company, last_evaluation_rating , Work_accident, salary_low, Deparment_sales, Department_technical
Gradient Boosting	13	satisfaction_level, projects_worked_on, average_monthly_hours, time_spend_company, last_evaluation_rating, Work_accident, salary_low, Department_hr, Department_management, Department_support, Department_technical, salary_high, salary_medium
Extra Trees	4	satisfaction_level, projects_worked_on, average_monthly_hours, time_spend_company

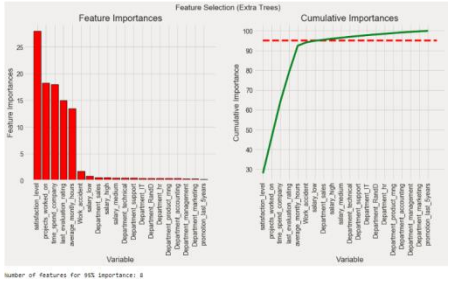
Feature Selection – Model Feature Importances

- Model Feature Importances





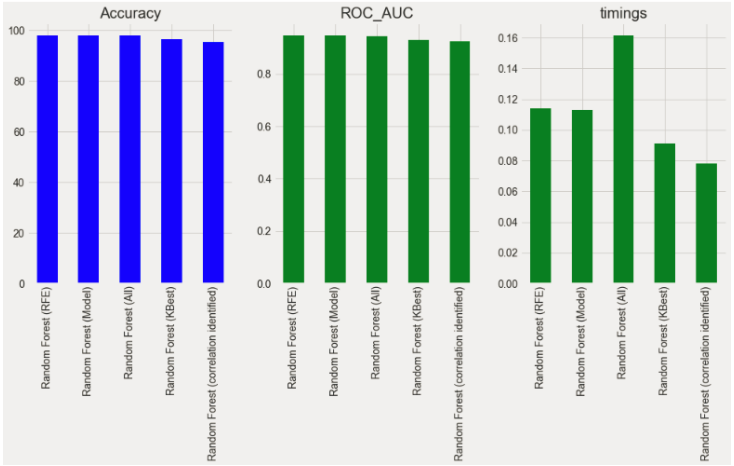


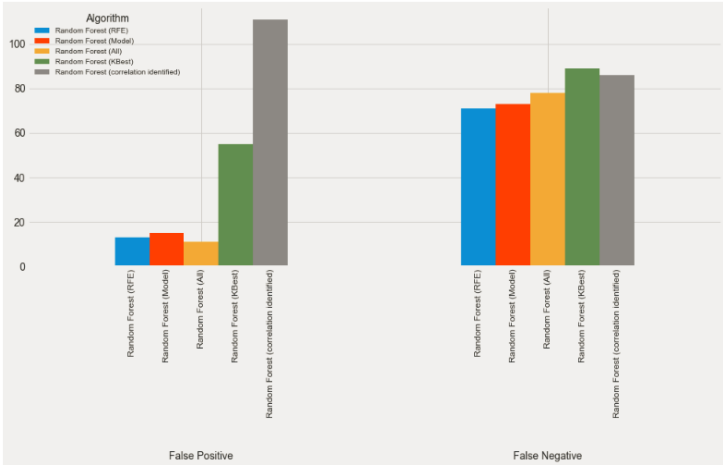


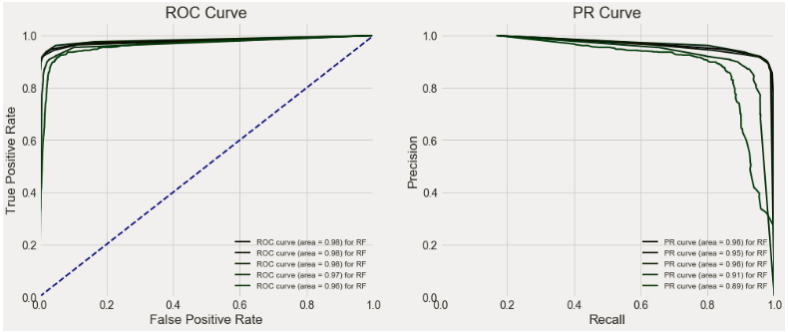
Classifier	95% Importance Threshold	Features
XG Boost	6	satisfaction_level, last_evaluation_rating, projects_worked_on, average_monthly_hours, time_spend_company, Work_accident
Random Forest		
Gradient Boosting		
Extra Trees	8	satisfaction_level, last_evaluation_rating, "projects_worked_on, average_monthly_hours, "time_spend_company", Work_accident, salary_low, department_sales

Feature Selection Evaluation – Random Forest

- Comparison of results for Random Forest classifier
  - Best: **RFE selected features**

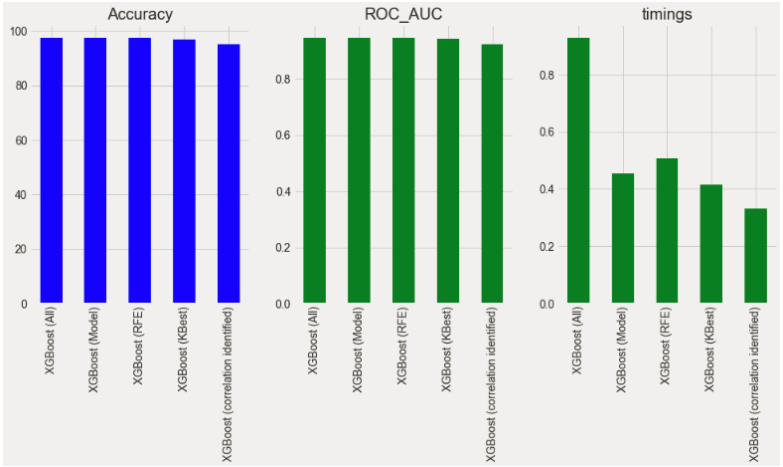


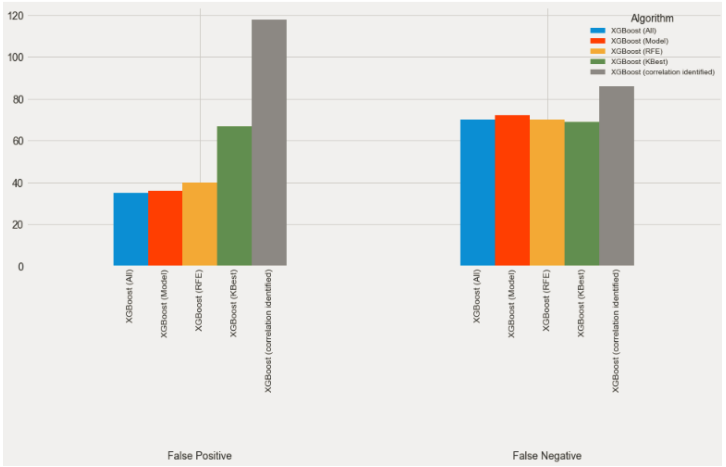


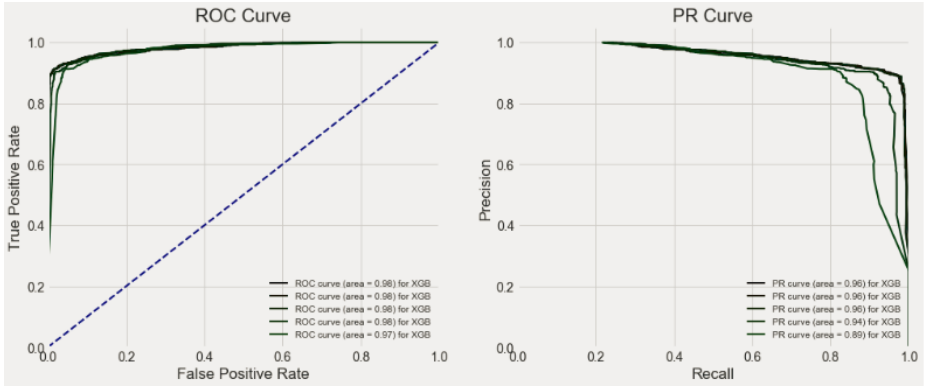


Feature Selection Evaluation – XG Boost

- Comparison of results for XG Boost classifier
  - Best: **All Features**

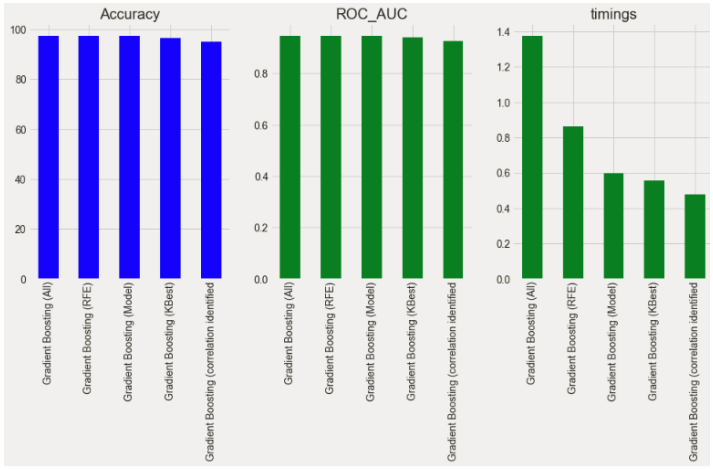


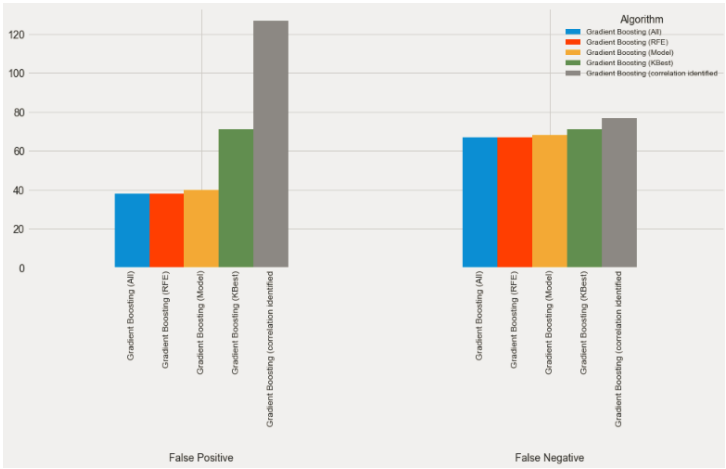


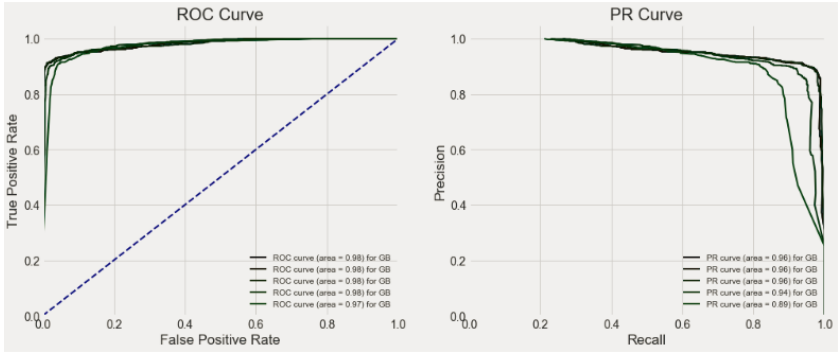


Feature Selection Evaluation – Gradient Boosting

- Comparison of results for Gradient Boosting classifier
  - Best: **<u>All Features</u>**

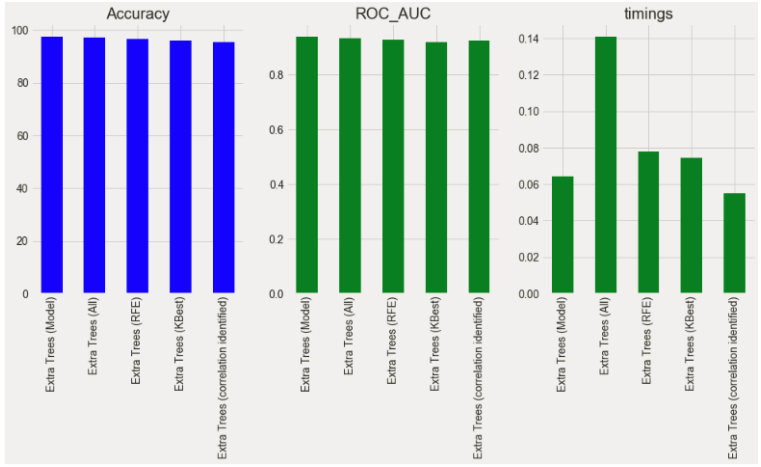


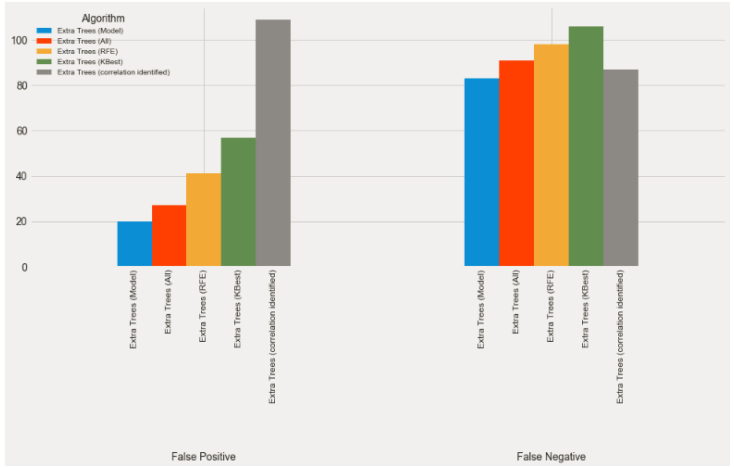


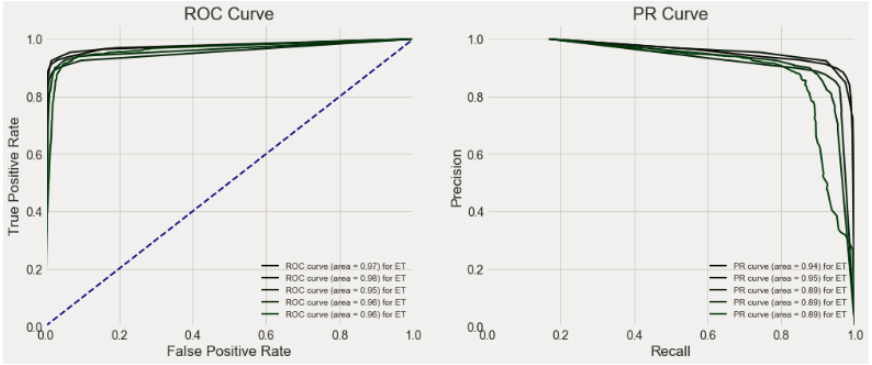


Feature Selection Evaluation – Extra Trees

- Comparison of results for Extra Trees classifier
  - Best: **Model Feature Importance** selected features

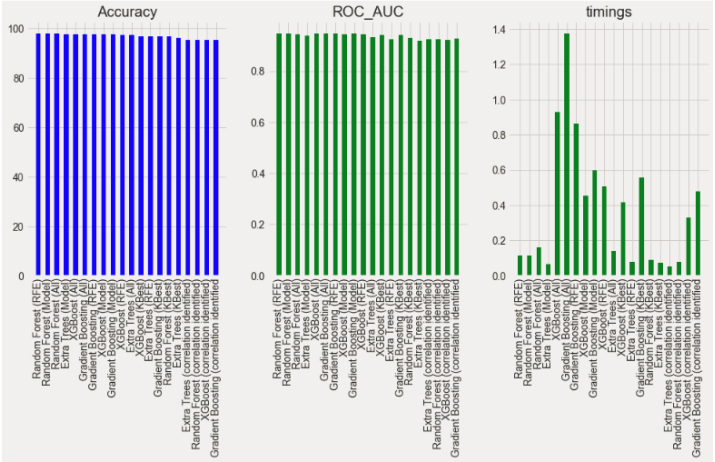


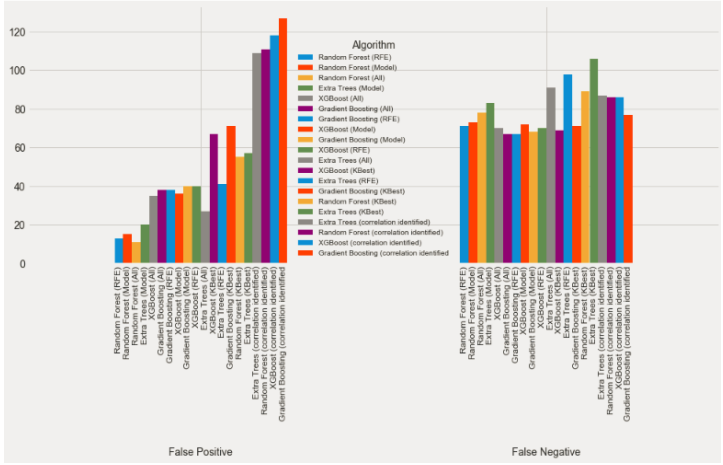


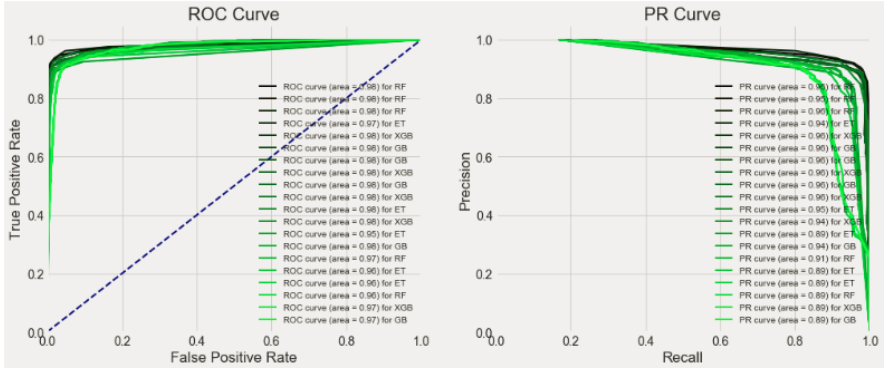


Feature Selection Evaluation – All

- Comparison of results for 4 selected algorithms
  - Random Forest / XGBoost / Gradient Boosting / Extra Trees
  - Best: **<u>Random Forest classifier</u>** using **<u>RFE</u>** selected features

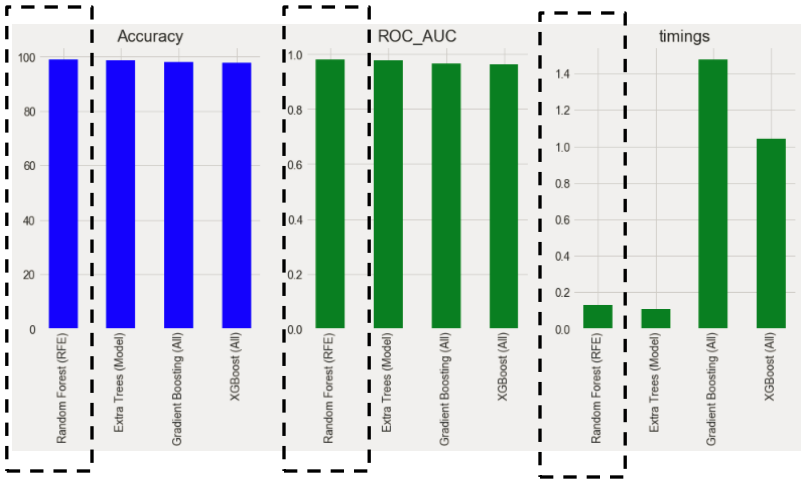


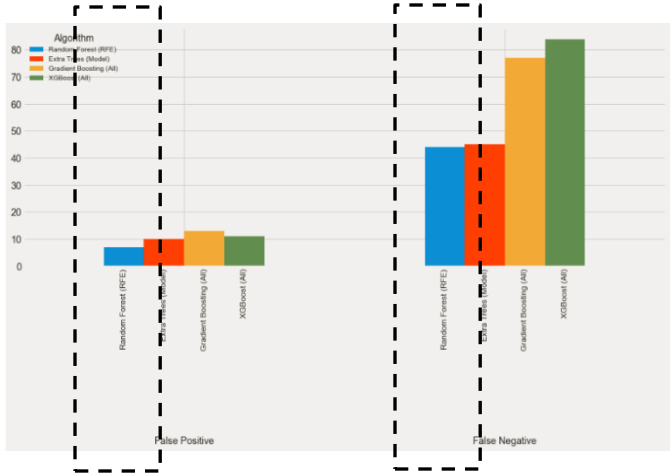


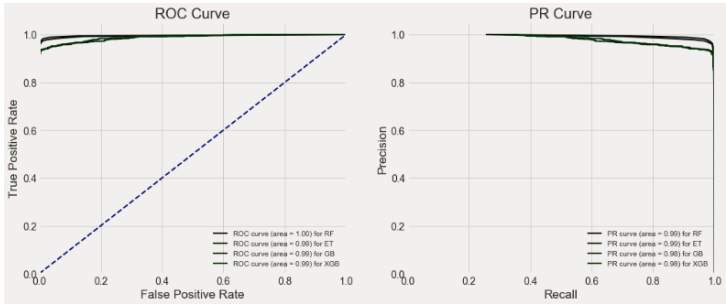


Feature Selection Evaluation – Test Dataset

- Comparison of results on test dataset (evaluate 4 best performers)
  - Best: **Random Forest classifier** using **RFE** selected features







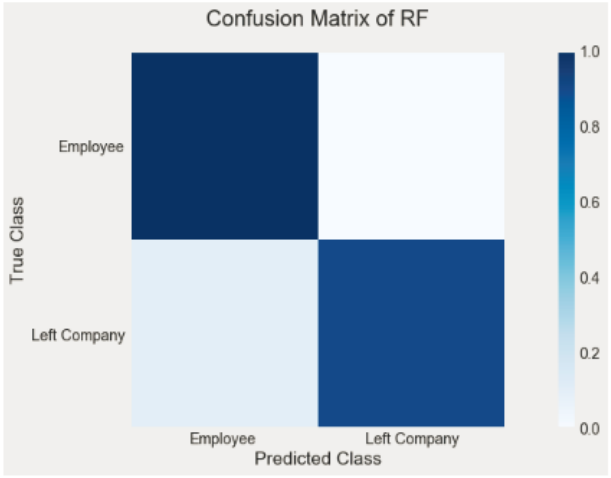
Conclusion: **Random Forest** using **RFE** selected features outperforms all other algorithms for both the train and test datasets

Feature Selection Evaluation – Train/Test Datasets Evaluation Results

- Evaluation Metrics for **Random Forest** using **RFE** selected features on both train and test datasets
  - Achieved accuracy >=98%
  - Good precision and recall characteristics

Name: Random Forest (RFE)

```
Accuracy on Validation Dataset: 98.00%
Timing on Validation Dataset: 0.11s
```



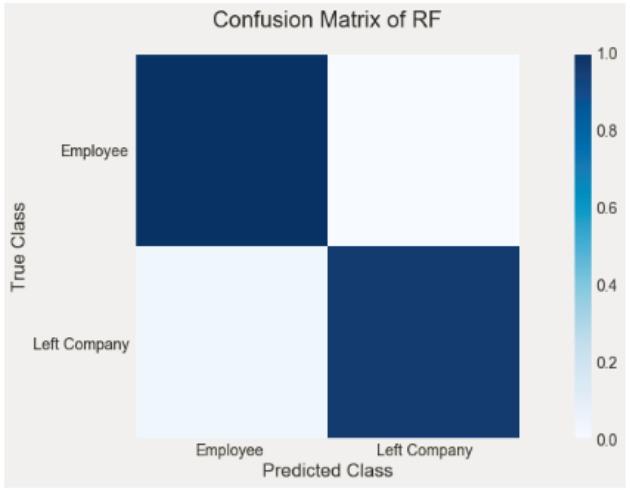
	precision	recall	f1-score	support
Employee	0.98	1.00	0.99	3485
Left	0.98	0.90	0.94	708
avg / total	0.98	0.98	0.98	4193

```
True Positives: 637
False Positives: 13
True Negatives: 3472
False Negatives: 71
```

Train Dataset

Name: Random Forest (RFE) on Test Dataset

```
Accuracy on Validation Dataset: 98.87%
Timing on Validation Dataset: 0.16s
```



	precision	recall	f1-score	support
Employee	0.99	1.00	0.99	3355
Left	0.99	0.96	0.98	1152
avg / total	0.99	0.99	0.99	4507

```
True Positives: 1108
False Positives: 7
True Negatives: 3348
False Negatives: 44
```

Test Dataset

Algorithm Tuning - General

- Use hyper-parameter tuning to improve the performance of the **Random Forest classifier** (using **RFE** selected features)

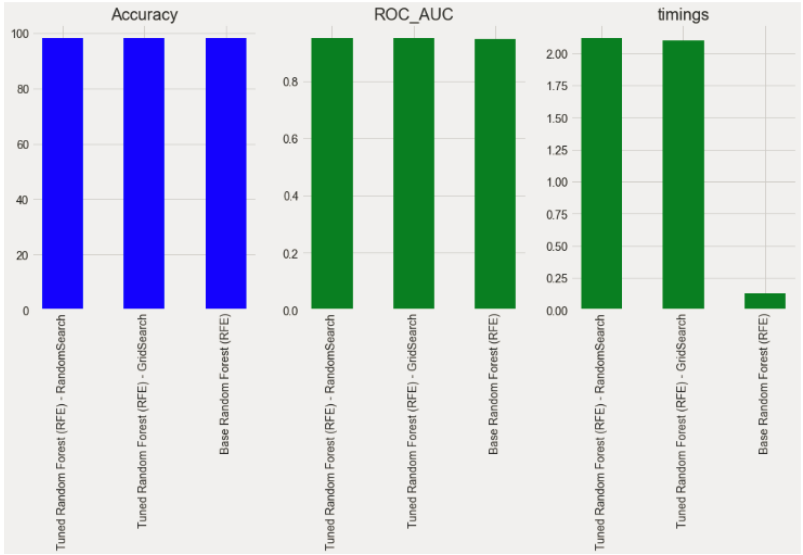
- 2 methods used:
  - Random Search Cross Validation (RandomizedSearchCV)
    - Define grid of hyper-parameter ranges and randomly sample from the grid, performing K-Fold CV with each combination of values
  - Grid Search with Cross Validation (GridSearchCV)
    - Explicitly specify every combination of settings to perform K-Fold CV with all combinations

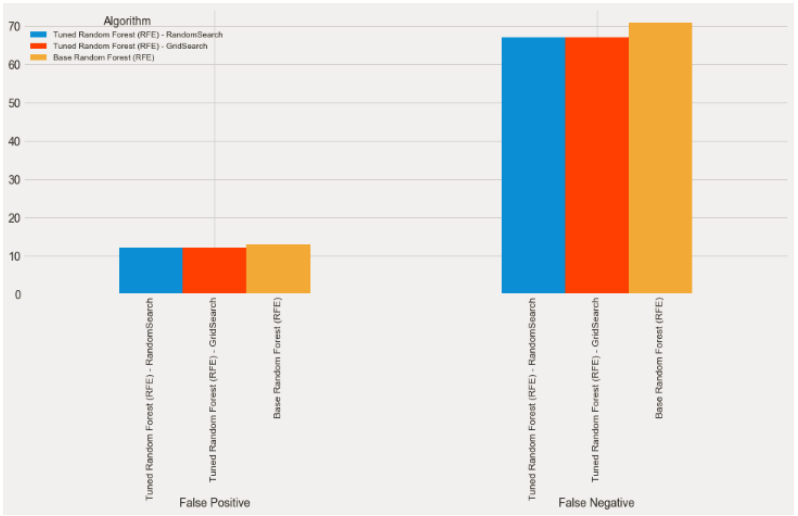
- Parameters Tuned:
  - max_features - number of features to consider when looking for the best split
  - max_depth - maximum number of levels in the tree
  - n_estimators - number of trees in the forest
  - min_samples_leaf - minimum number of samples required to be at a leaf node
  - min_samples_split - minimum number of samples required to split an internal node

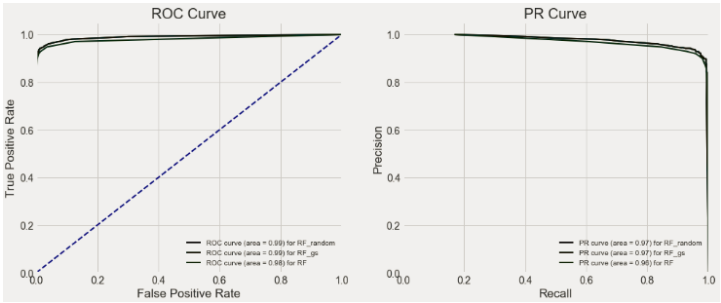
- Performance metrics of Base and tuned models (from Random and Grid Searches) are evaluated to check improvements achieved from tuning

Algorithm Tuning – Results (Train Dataset)

- Comparison of results between base and tuned models (on train dataset)
  - Best: **Tuned model using Random search derived parameters**



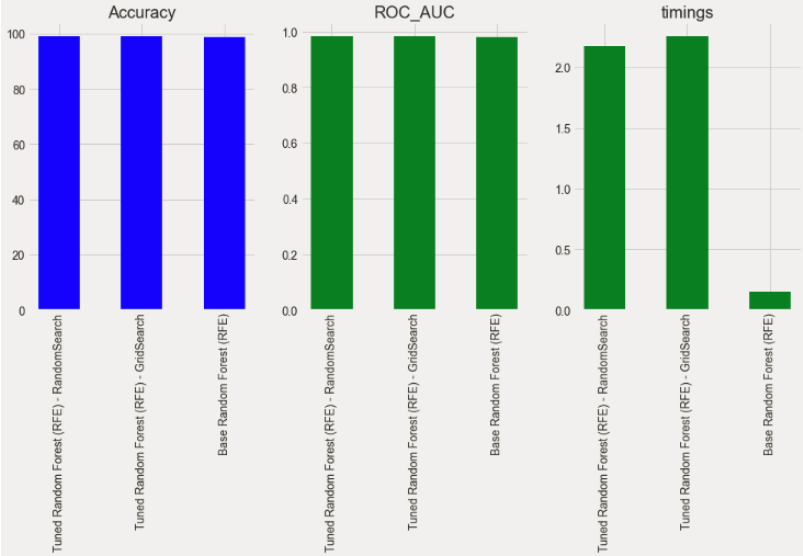


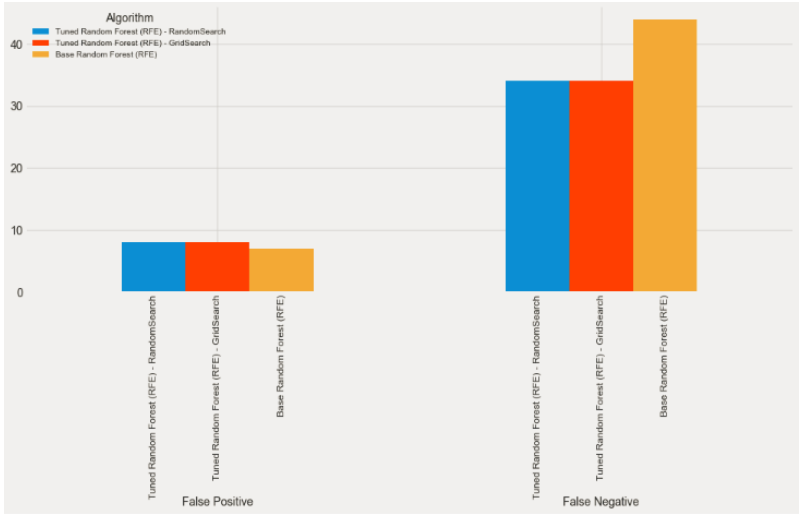


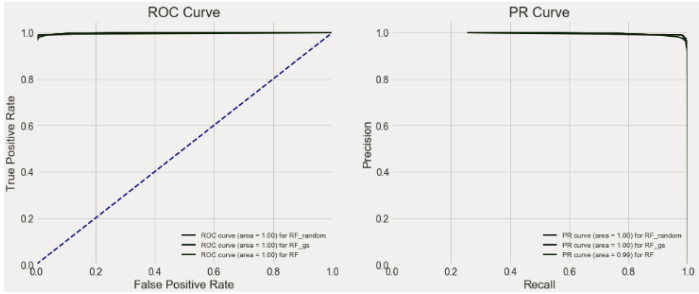
Hyper-Parameters		
Base	***Tuned***	
	Grid Search	**Random Search**
max_depth: None max_features: Auto min_samples_leaf: 1 min_samples_split: 2 n_estimators: 10	max_depth: 30 max_features: 4 min_samples_leaf: 1 min_samples_split: 2 n_estimators: 100	max_depth: 60 max_features: 4 min_samples_leaf: 1 min_samples_split: 2 n_estimators: 100

Algorithm Tuning – Results (Test Dataset)

- Comparison of results between base and tuned models (on test dataset)
  - Best: **Tuned model using Random search derived parameters**







Algorithm Tuning – Performance Analysis

- Performance Analysis on Train and Test Datasets
  - Tuned models result in slight improvement in accuracy, with increased # of false positives offset by decreased # of false negatives
  - Improvement % achieved at the expense of longer training times

	Test - Grid Search Tuned (%)	Test - Random Search Tuned (%)	Train - Grid Search Tuned (%)	Train - Random Search Tuned (%)
Accuracy	0.2020	0.2020	0.1217	0.1217
False Negatives	22.7273	22.7273	5.6338	5.6338
False Positives	-14.2857	-14.2857	7.6923	7.6923
True Negatives	-0.0299	-0.0299	0.0288	0.0288
True Positives	0.9025	0.9025	0.6279	0.6279

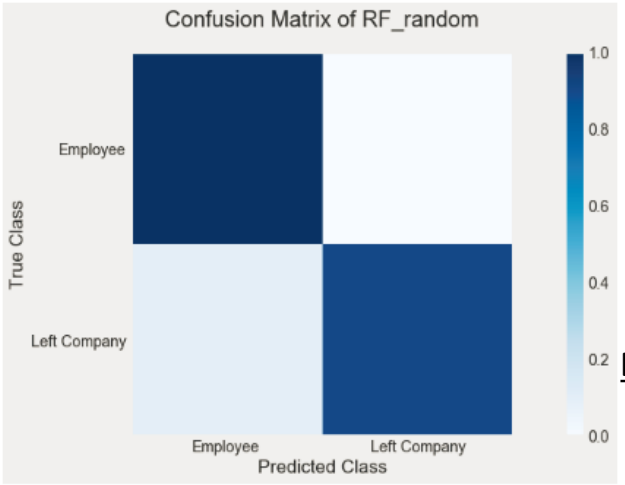
	Test - Grid Search Tuned (%)	Test - Random Search Tuned (%)	Train - Grid Search Tuned (%)	Train - Random Search Tuned (%)
Timing	-1413.4206	-1358.3870	-1493.3546	-1507.7706

Algorithm Tuning – Train/Test Datasets Evaluation Results

- Evaluation Metrics for tuned **Random Forest** (using **RFE** selected features) using **Random Search** derived parameters on both train and test datasets
  - Improved accuracy (>99% on test dataset)
  - Reduced # of False negatives but slight increase in # of false positives

Name: Tuned Random Forest (RFE) on Train Dataset (using Random Search)

```
Accuracy on Validation Dataset: 98.12%
Timing on Validation Dataset: 2.12s
```



	precision	recall	f1-score	support
Employee	0.98	1.00	0.99	3485
Left	0.98	0.91	0.94	708
avg / total	0.98	0.98	0.98	4193

```
True Positives: 641
False Positives: 12
True Negatives: 3473
False Negatives: 67
```

Train Dataset

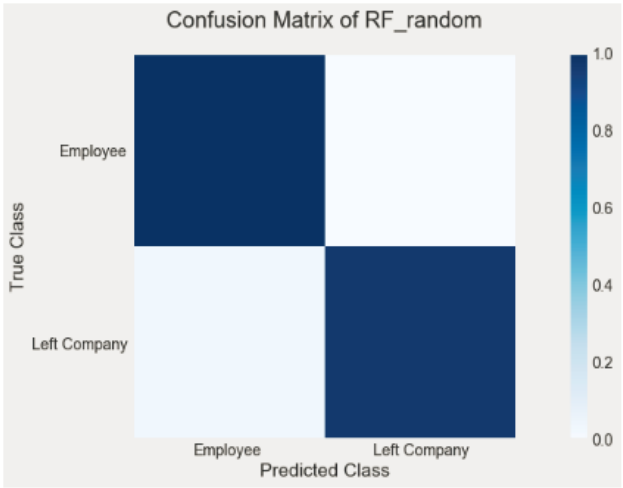
Base

```
True Positives: 637
False Positives: 13
True Negatives: 3472
False Negatives: 71
```

Base

Name: Tuned Random Forest (RFE) on Test Dataset (using Random Search)

```
Accuracy on Validation Dataset: 99.07%
Timing on Validation Dataset: 2.09s
```



	precision	recall	f1-score	support
Employee	0.99	1.00	0.99	3355
Left	0.99	0.97	0.98	1152
avg / total	0.99	0.99	0.99	4507

```
True Positives: 1118
False Positives: 8
True Negatives: 3347
False Negatives: 34
```

Test Dataset

Base

```
True Positives: 1108
False Positives: 7
True Negatives: 3348
False Negatives: 44
```

Summary

- 10 classification algorithms and 6 ensemble methods were evaluated, with **Random Forest Classifier** coupled with selected features using **Reduced Feature Elimination (RFE)** method outperforming all other algorithms/ feature selection combinations

- Hyper-parameter tuning using Random and Grid Searches yield marginal improvements in accuracy at the expense of longer model training times

References

- Scikit-Learn Supervised Learning Docs
  - http://scikit-learn.org/stable/supervised_learning.html
- Scikit-Learn Ensemble Methods Docs
  - http://scikit-learn.org/stable/modules/ensemble.html
- XGBoost Scikit-Learn API Docs
  - http://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn
- Scoring Classifier Models Using Scikit-Learn
  - http://benalexkeen.com/scoring-classifier-models-using-scikit-learn/
- Feature Selection For Machine Learning in Python
  - https://machinelearningmastery.com/feature-selection-machine-learning-python
- Markdown for Jupyter notebooks cheatsheet
  - https://medium.com/ibm-data-science-experience/markdown-for-jupyter-notebooks-cheatsheet-386c05aeebed

Q&A