Higgs Boson Machine Learning Challenge

DataUniversalis

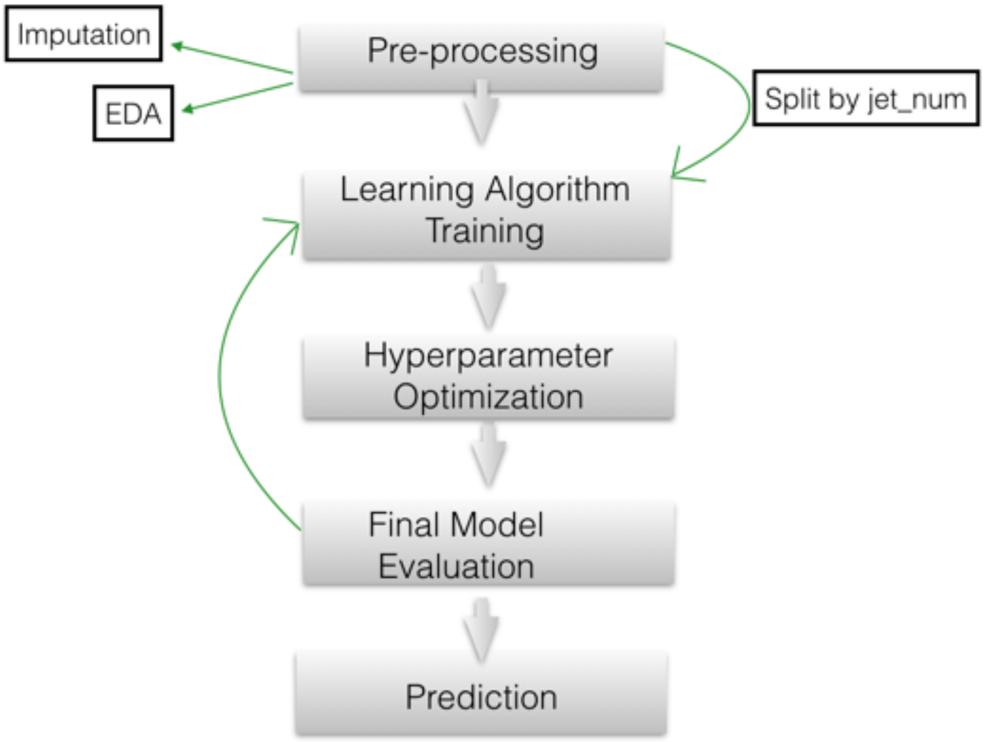
Miaozhi YU, Shuo ZHANG, Bin FANG, Chuan SUN

Aug. 30, 2016

Outline

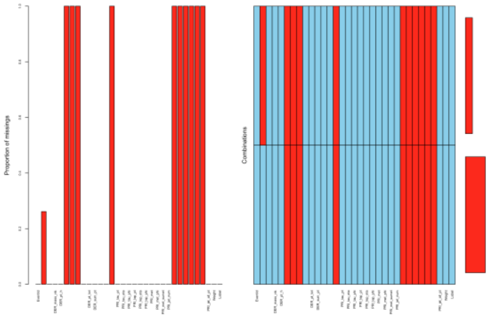
- ❑ **Workflow**
- ❑ Exploratory Data Analysis
- ❑ Models
    - ❑ Random Forests
    - ❑ Gradient Boosted Model
    - ❑ Neural Networks
    - ❑ XGBoost
- ❑ Lesson Learned

Workflow

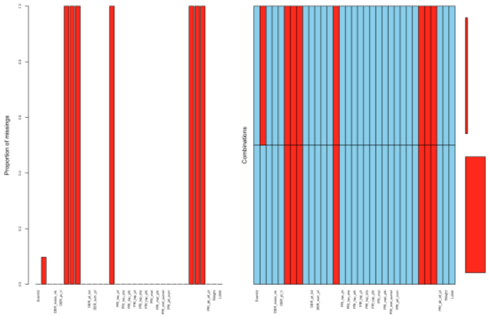

Imputation
EDA
Pre-processing
Split by jet_num
Learning Algorithm Training
Hyperparameter Optimization
Final Model Evaluation
Prediction

EDA -- Missingness Imputation by kNN

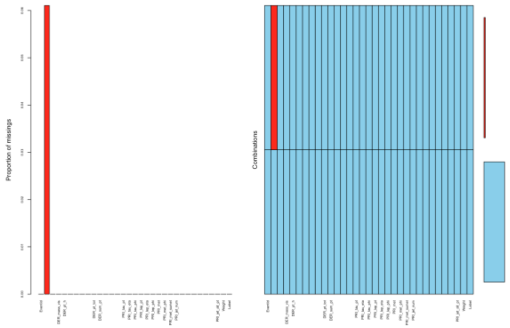
PRI_jet_num = 0



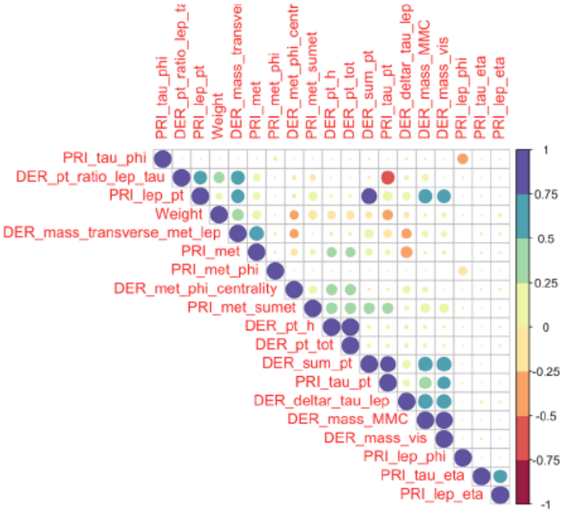
PRI_jet_num = 1



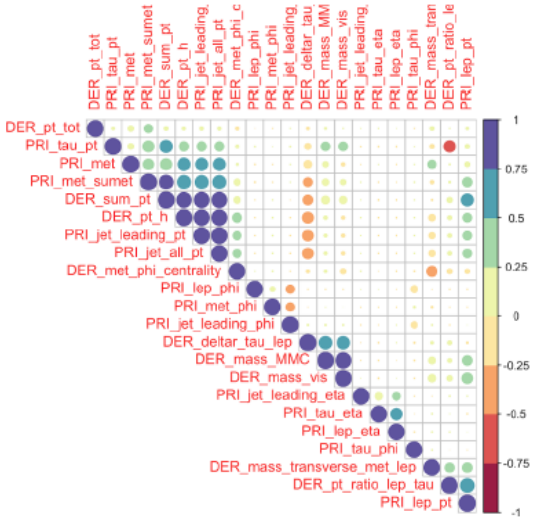
PRI_jet_num = 2 or 3



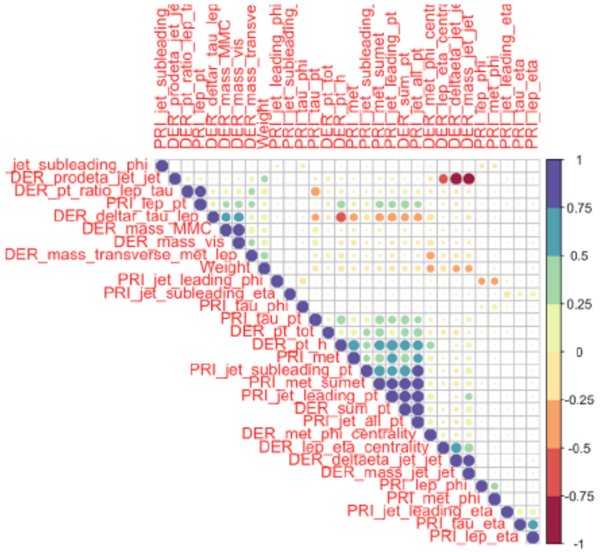
EDA -- Correlation



PRI_jet_num = 0



PRI_jet_num = 1



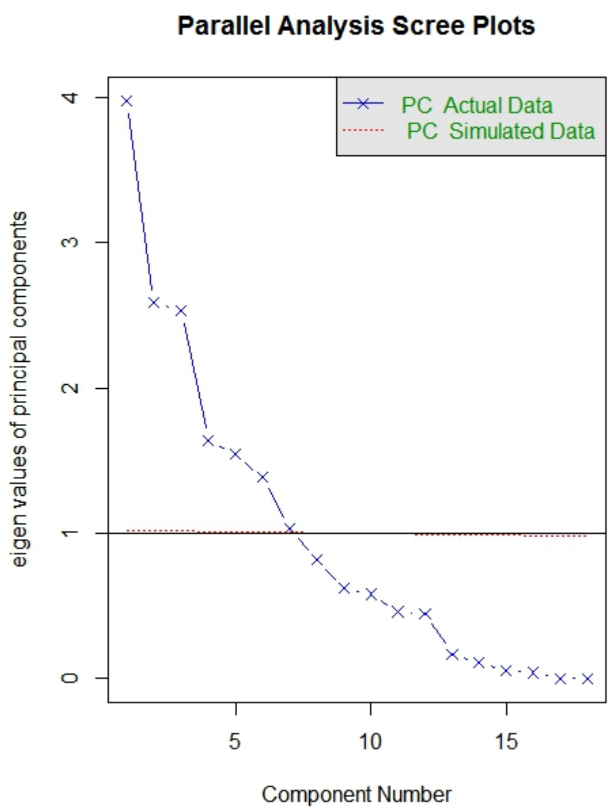
PRI_jet_num = 2 or 3

- ❑ Variables of same prefix (with some uppercase letters) have high positive or negative correlation.

EDA -- Principal Component Analysis

Principal Components Analysis
Call: principal(r = df0, nfactors = 7, rotate = "none")
Standardized loadings (pattern matrix) based upon correlation matrix

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	h2	u2	com
DER_mass_MMC	0.90	-0.18	-0.11	0.21	-0.04	0.00	-0.01	0.90	0.103	1.2
DER_mass_transverse_met_lep	0.21	0.80	-0.46	-0.05	0.04	0.00	0.00	0.90	0.095	1.8
DER_mass_vis	0.88	-0.23	-0.12	0.12	-0.03	0.00	-0.01	0.85	0.145	1.2
DER_pt_h	0.18	0.44	0.82	0.16	-0.04	-0.01	0.00	0.93	0.070	1.7
DER_deltar_tau_lep	0.51	-0.52	-0.03	0.55	-0.13	-0.01	-0.01	0.84	0.159	3.1
DER_pt_tot	0.18	0.44	0.82	0.16	-0.04	-0.01	0.00	0.93	0.070	1.7
DER_sum_pt	0.89	0.02	-0.09	-0.33	0.07	0.00	0.01	0.91	0.094	1.3
DER_pt_ratio_lep_tau	0.09	0.59	-0.44	0.59	-0.11	0.00	0.01	0.91	0.088	3.0
DER_met_phi_centrality	0.10	-0.07	0.58	0.39	-0.10	-0.01	-0.01	0.52	0.484	1.9
PRI_tau_pt	0.64	-0.32	0.19	-0.62	0.12	0.00	0.01	0.95	0.047	2.7
PRI_tau_eta	0.01	-0.03	0.02	0.20	0.86	0.02	-0.01	0.77	0.225	1.1
PRI_tau_phi	-0.01	0.01	0.00	-0.01	-0.02	0.67	-0.60	0.81	0.190	2.0
PRI_lep_pt	0.76	0.37	-0.35	0.12	-0.02	0.00	0.01	0.85	0.151	2.0
PRI_lep_eta	0.01	-0.02	0.04	0.20	0.86	0.02	-0.01	0.77	0.225	1.1
PRI_lep_phi	0.00	-0.01	0.00	-0.01	0.02	-0.85	-0.03	0.72	0.278	1.0
PRI_met	0.12	0.77	0.02	-0.36	0.10	0.00	0.00	0.75	0.254	1.5
PRI_met_phi	0.00	-0.01	0.02	0.01	0.00	0.46	0.82	0.89	0.114	1.6
PRI_met_sumet	0.47	0.14	0.47	-0.14	0.02	0.01	-0.01	0.48	0.518	2.4



- Importance: mass-related variables.
- Impute missing values: DER_mass_MMC

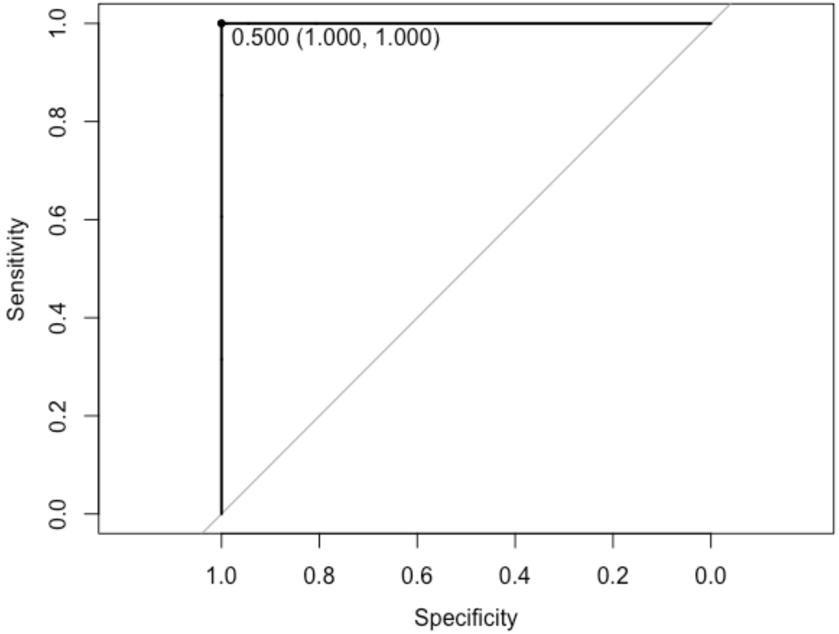
Outline

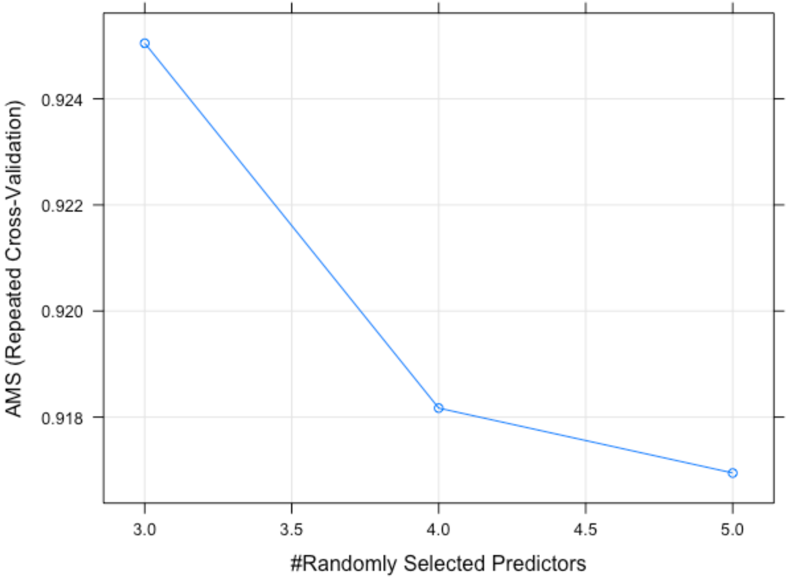
- ❑ Workflow
- ❑ Exploratory Data Analysis
- ❑ **Models**
    - ❑ **Random Forests**
    - ❑ Gradient Boosted Model
    - ❑ Neural Networks
    - ❑ XGBoost
- ❑ Lesson Learned

Random Forests

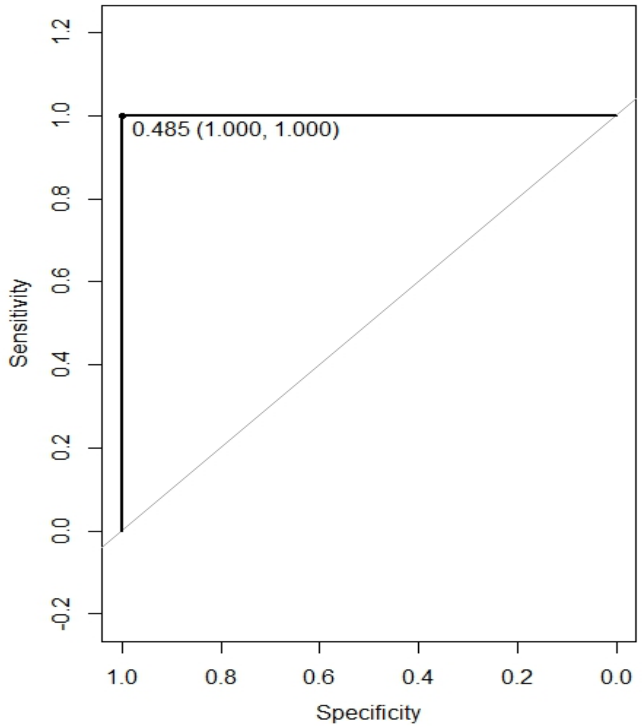
- Two parameters: ntree and mtry.
- 1st tuning: ntree = c(2000,5000,8000) mtry = c(3,4,5,6)
- However, R crushed due to large computation
- 2nd tuning: ntree = c(500,800,1000) mtry = c(3,4,5)

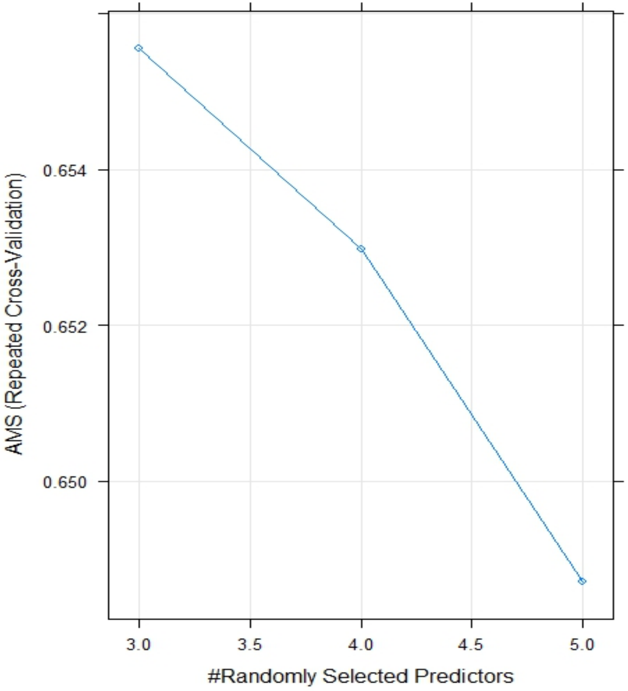
Random Forests(DF:PRI_jet_num=0, ntree = 1000)



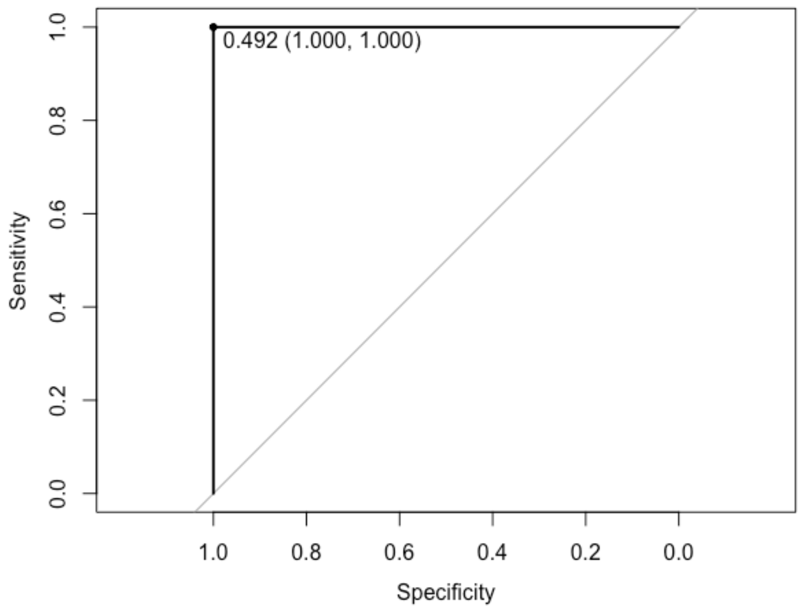


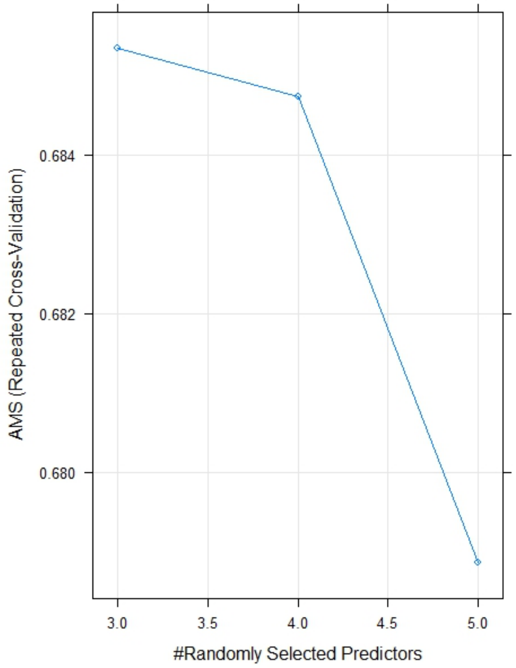
Random Forests(DF:PRI_jet_num=1, ntree = 1000)



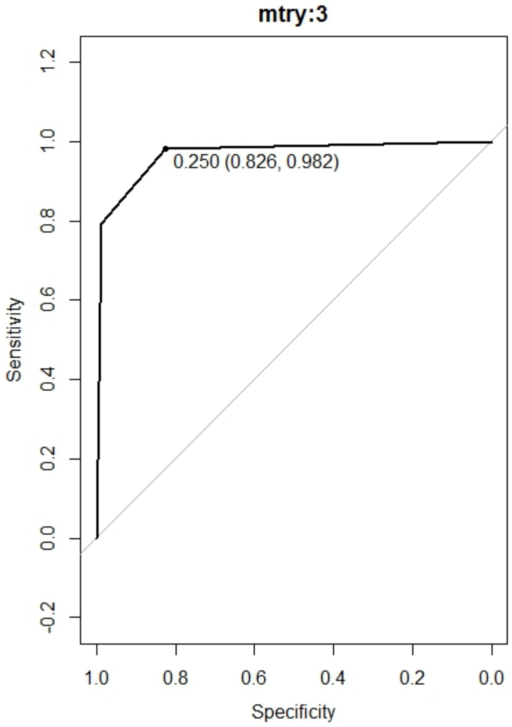


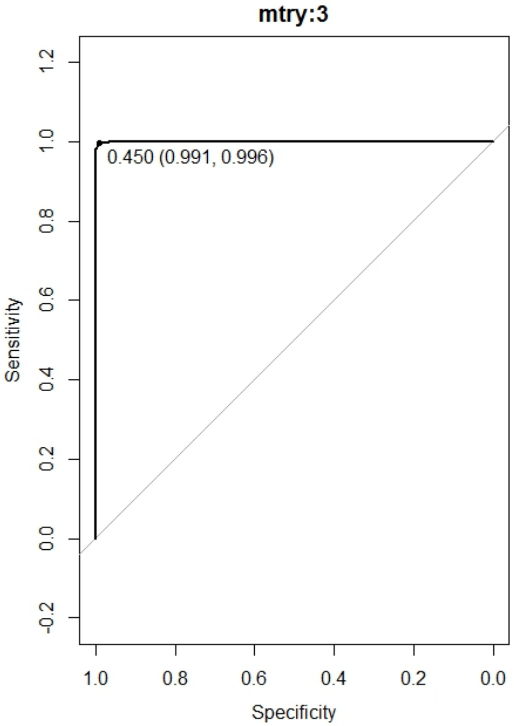
Random Forests(DF:PRI_jet_num=2&3, ntree = 1000)





Random Forests(PRI_jet_num=2&3, ntree = 2 & 10)





Random Forests---Summary

Pros:

- Theoretically, it can never overfit
- Good with very large data set
- Robust against outliers

Cons:

- Not a fast algorithm. Very time and memory space consuming
- Less accurate than boosted tree models
- Cannot handle NAs

Outline

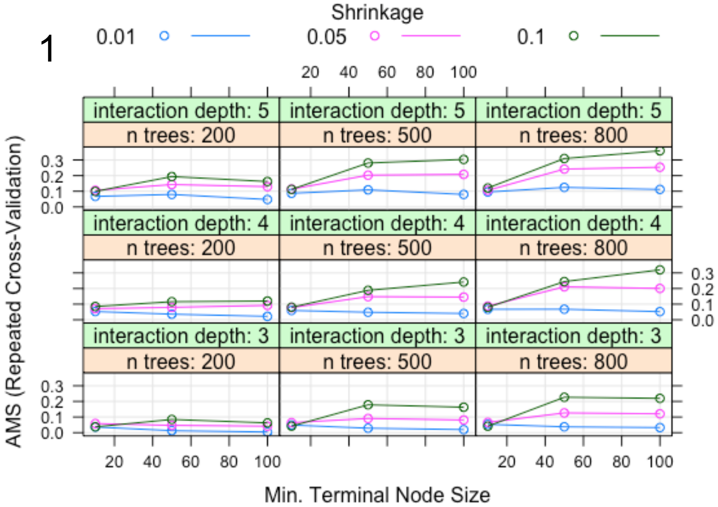
- ❑ Workflow
- ❑ Exploratory Data Analysis
- ❑ **Models**
  - ❑ Random Forests
  - ❑ **Gradient Boosted Model**
  - ❑ Neural Networks
  - ❑ XGBoost
- ❑ Lesson Learned

Gradient Boosting Model (DF:PRI_jet_num=1)

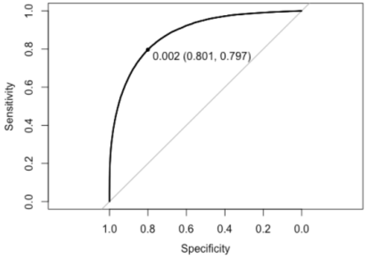
Tuning	AUC
1	**0.88**
2	**0.86**
3	**0.91**
4	**0.93**

- Tuning four parameters, 5-fold validation
- 1st tuning: interaction.depth =c(3, 4, 5), n.trees =c(200, 500, 800), shrinkage = c(0.1, 0.05, 0.01), n.minobsinnode = c(10, 50, 100)
- 2nd tuning: 800 trees and interaction.depth (5,7,10)
- 3rd tuning: 5 interaction.depth and large trees(800, 2000, 5000)
- 4th tuning: 5 interaction.depth and larger trees(5000, 7500, 10000)

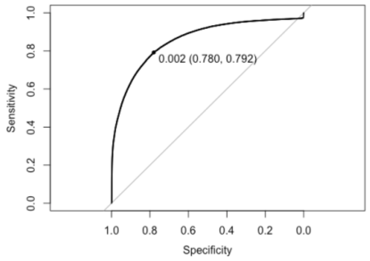
1



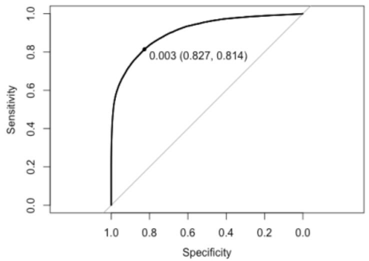
1



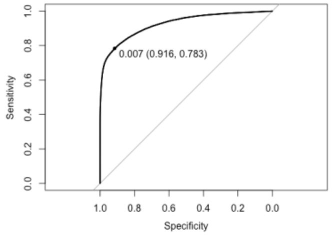
2



3



4

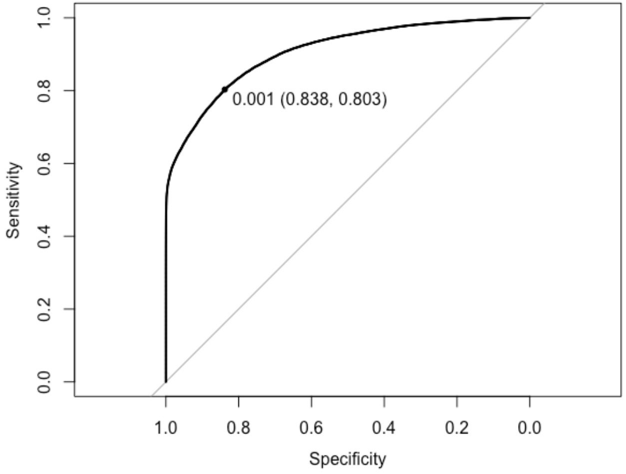


Parameter tuning for the other 2 subsets

jet_num=0

interaction.depth = 5, n.trees =10000, shrinkage = 0.1, n.minobsinnode = 100

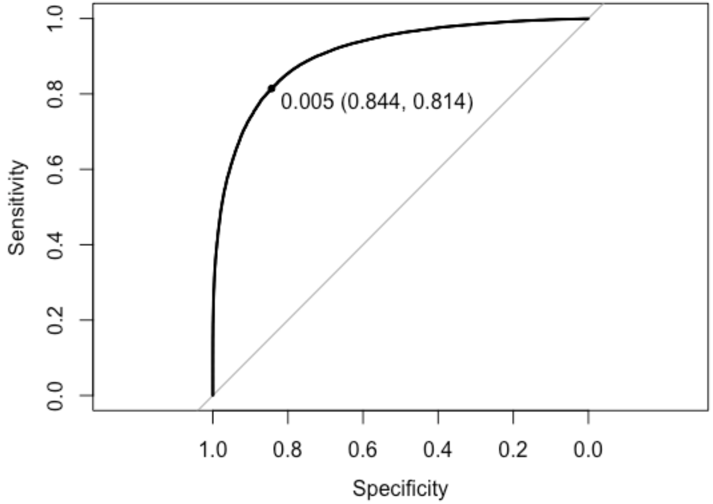
AUC=0.907



jet_num=c(2,3)

interaction.depth = 5, n.trees =800, shrinkage = 0.05, n.minobsinnode = 100

AUC=0.91



Summary of Gradient Boosting Model

Insight:

- Interaction.depth: **sqr(number of variables)**
- N.trees: **10000,** reduce overfitting by smaller learning rate and cross-validation
- Shrinkage: big trees: **0.001**; small trees: **0.1**
- N.minobsinnode: **100:** reduce variance in predictions at leaves

Pros:

- Use all features
- Higher accuracy and lower possibility of overfitting with more trees

Cons:

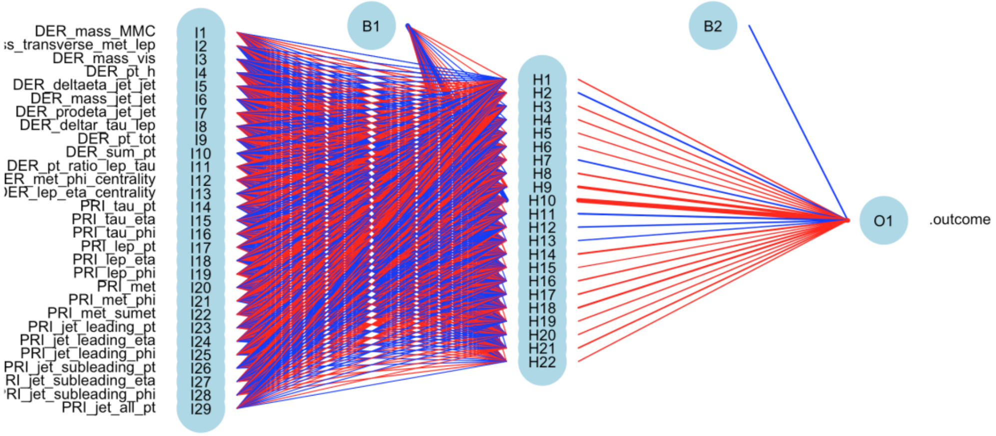
- Susceptible to outliers
- Lack of interpretability and higher complexity
- Harder to tune hyperparameters than other models
- Slow to train

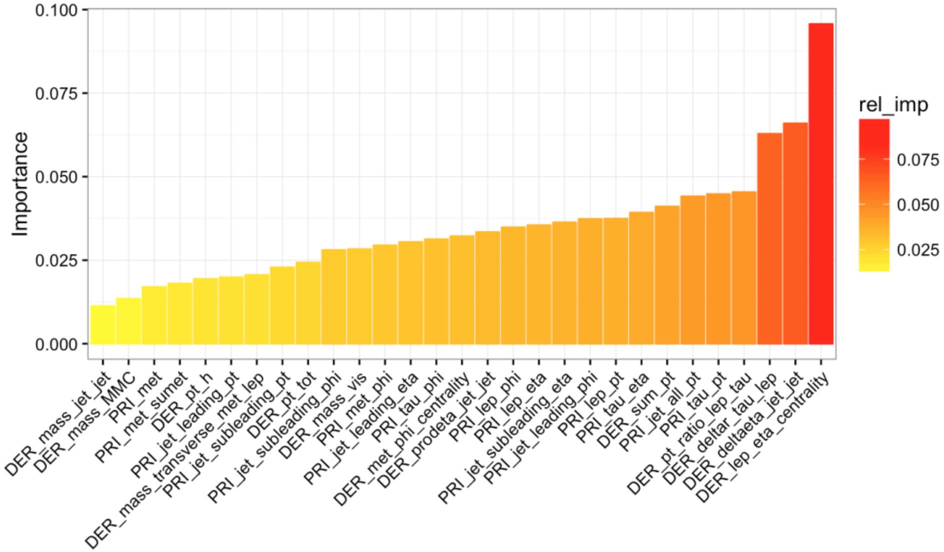
Outline

- ❑ Workflow
- ❑ Exploratory Data Analysis
- ❑ **Models**
  - ❑ Random Forests
  - ❑ Gradient Boosted Model
  - ❑ **Neural Networks**
  - ❑ XGBoost
- ❑ Lesson Learned

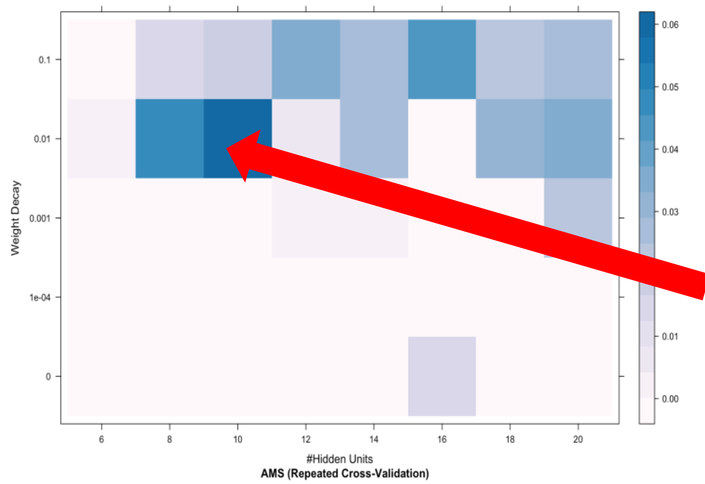
Neural Networks

- ❑Single hidden layer NN model “nnet” under “Caret” library.
- ❑Two tuning parameters: number of hidden units and weight decay.
- ❑Select initial combinations of ranges of hidden units between 6-20, and decay weights between 0-0.5.

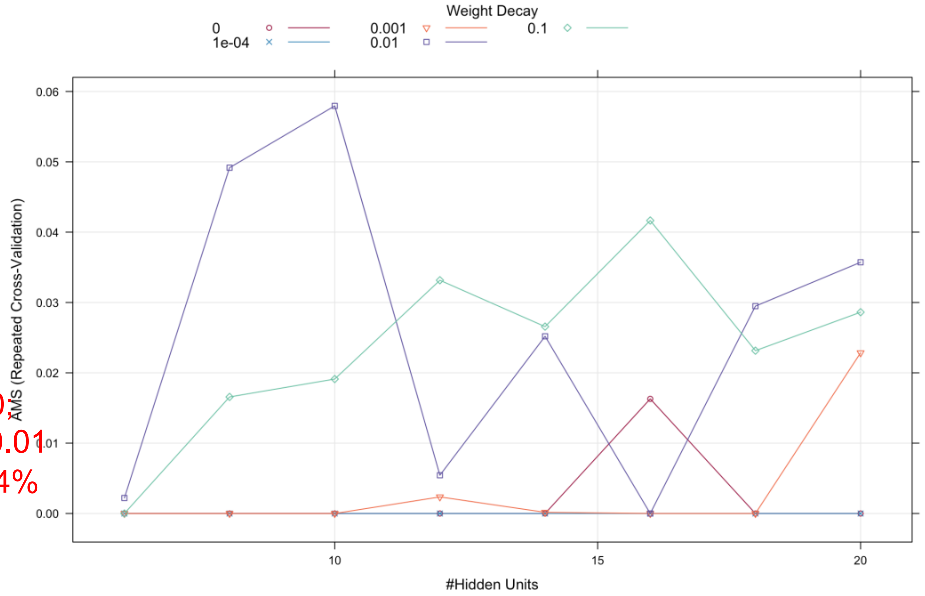


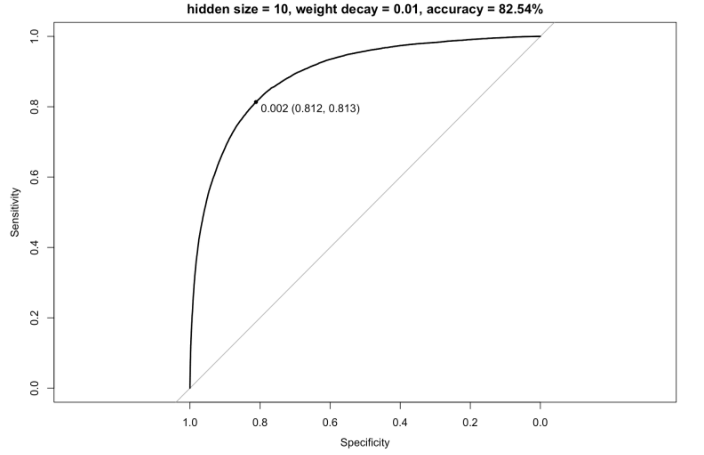


Neural Networks -- jet_number = 0



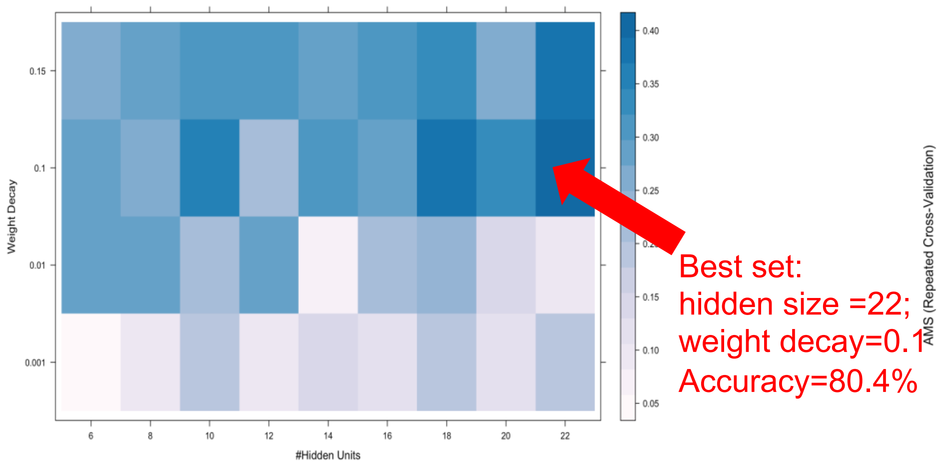
Best set:
hidden size =10;
weight decay=0.01
Accuracy=82.54%



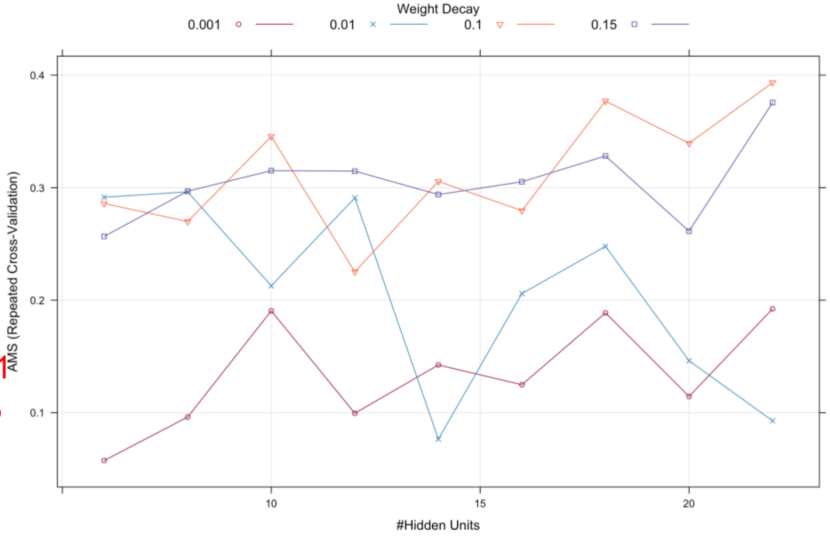


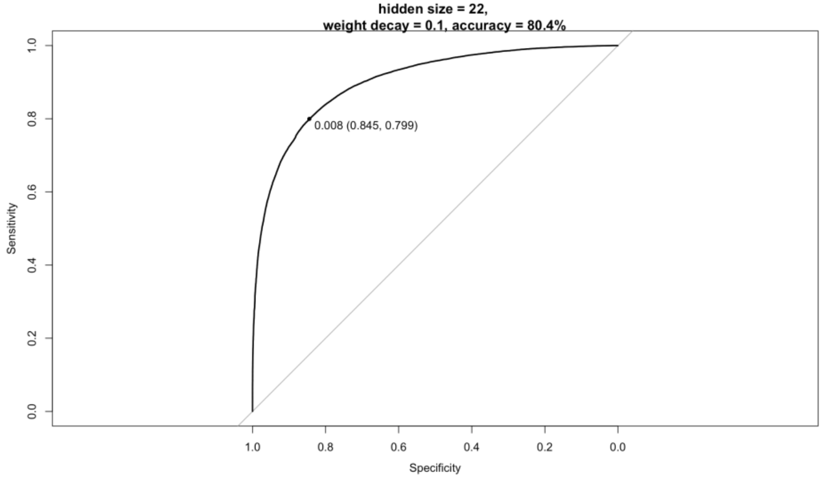
- ❑ Hidden units: seq(6,20,2); Weight decay: c(1e-04, 1e-03,0.01,0.1,0).
- ❑ Fluctuation trend of AMS is noticed as hidden unit increases.

Neural Networks -- jet_number = 2/3



AMS (Repeated Cross-Validation)





- Hidden units: seq(6,22,2); Weight decay: c(1e-03,0.01,0.1,0.15).
- AMS steadily increases as more hidden units used.

Neural Networks -- Summary

- Insight
  - Best performance when weight decay = 0.1 or 0.15.
  - More hidden nodes may be required.
- Pros & Cons
  - Can handle hundreds or thousands of input variables.
  - Once trained, predictions are fast.
  - Training is computationally expensive.
  - Need a portion of the data for validation during NN training.

Outline

- ❑ Workflow
- ❑ Exploratory Data Analysis
- ❑ **Models**
  - ❑ Random Forests
  - ❑ Gradient Boosted Model
  - ❑ Neural Networks
  - ❑ **XGBoost**
- ❑ Lesson Learned

XGboost

6 parameters to tune

nrounds

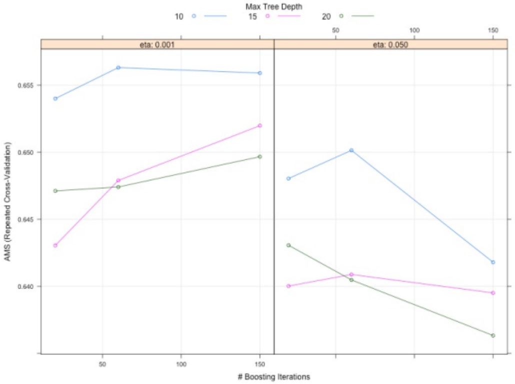
max_depth

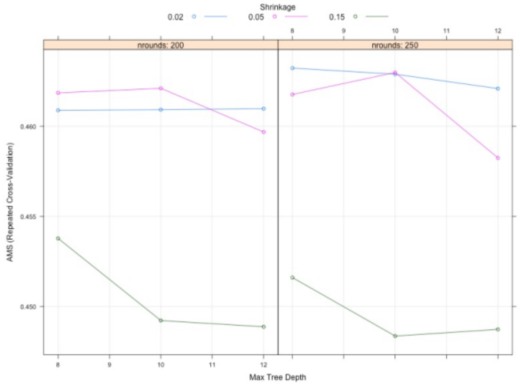
eta

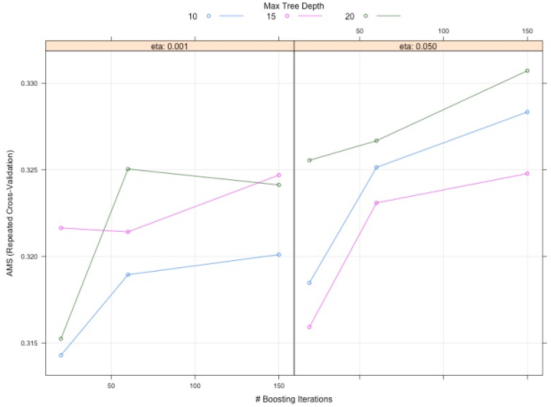
- Guidelines
  - Begin with rough grid, gradually fine tuning
  - Prefer slow learners (small eta, eg. 0.005)
  - Tune max_depth (9 or 10)
  - Fixed colsample_bytree (0.7)
  - Fixed gamma (5)
  - Fixed min_child_weight (0.1)

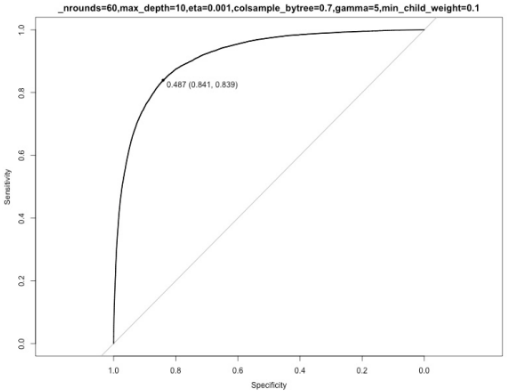
```
Parameter tuning for subset with jet number 0
    (nrounds=c(20, 60, 150), max_depth=c(10, 15, 20), eta=c(0.001, 0.05), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)
    (nrounds=c(250), max_depth=c(7, 10), eta=c(0.001), colsample_bytree=0.7, gamma=c(1, 5), min_child_weight=0.1)
    (nrounds=c(100, 150, 200), max_depth=c(8, 9, 10), eta=c(0.001, 0.005), colsample_bytree=0.7, gamma=c(5), min_child_weight=0.1)
    (nrounds=c(60, 100, 150, 200, 400), max_depth=c(10), eta=c(0.001), colsample_bytree=0.7, gamma=c(5), min_child_weight=0.1)

Parameter tuning for subset with jet number 1
    (nrounds=c(20, 60, 150), max_depth=c(10, 15, 20), eta=c(0.001, 0.05), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)
    (nrounds=c(200, 300), max_depth=c(10), eta=c(0.05), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)
    (nrounds=c(200, 250), max_depth=c(8, 10, 12), eta=c(0.02, 0.05, 0.15), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)

Parameter tuning for subset with jet number 2&3
    (nrounds=c(20, 60, 150), max_depth=c(10, 15, 20), eta=c(0.001, 0.05), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)
    (nrounds=c(200, 400), max_depth=c(20), eta=c(0.001, 0.05), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)
    (nrounds=c(200, 250), max_depth=c(10, 20), eta=c(0.001, 0.003), colsample_bytree=0.7, gamma=5, min_child_weight=0.1)
```

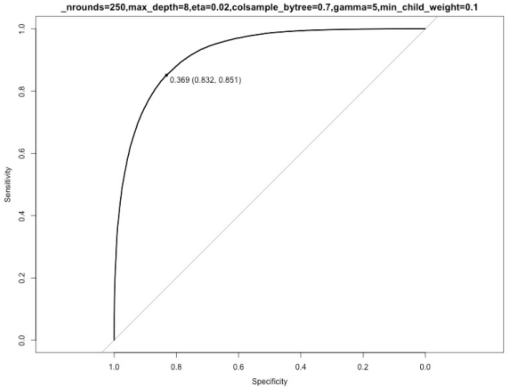
Parameter tuning for all 3 subsets

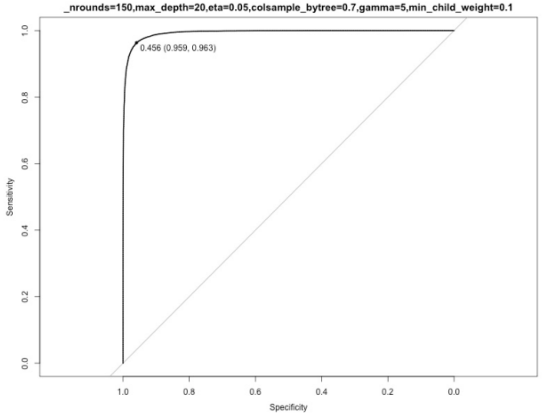




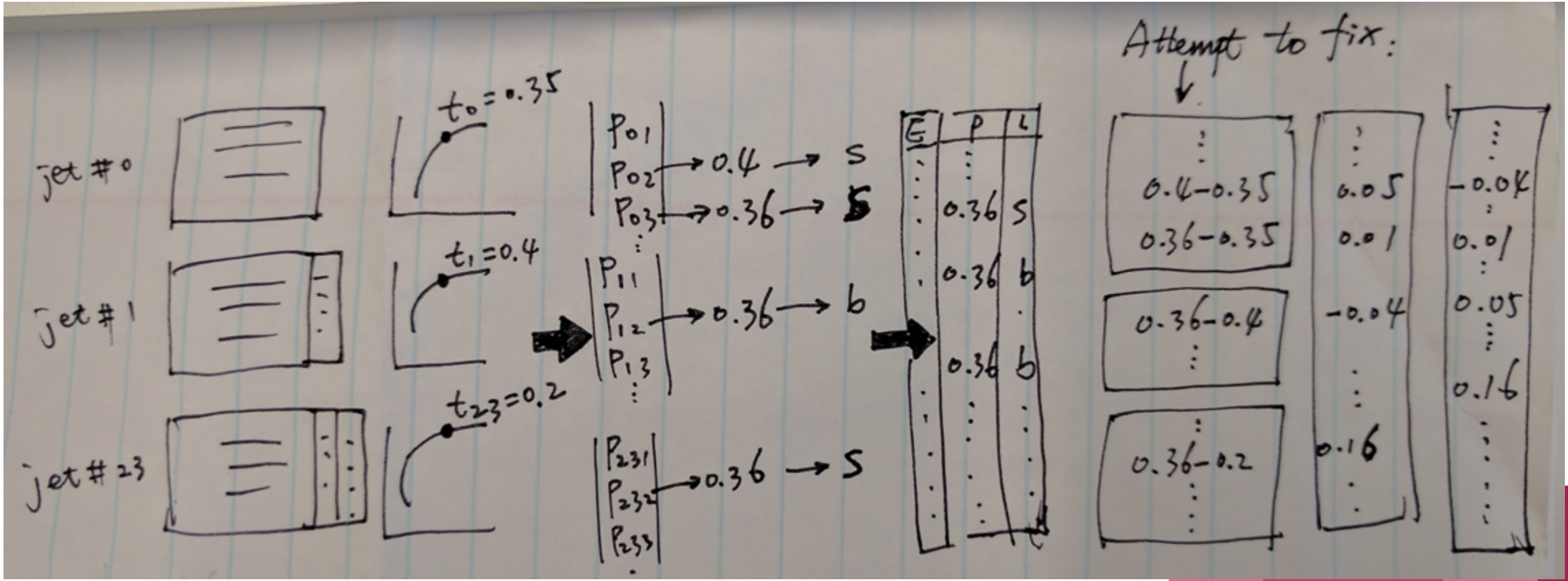








When combining all predictions ...



Outline

- ❑ Workflow
- ❑ Exploratory Data Analysis
- ❑ Models
    - ❑ Random Forests
    - ❑ Gradient Boosted Model
    - ❑ Neural Networks
    - ❑ XGBoost
- ❑ **Lesson Learned**

Lesson learned

- ❑ Prioritize team's overall strategy.
- ❑ Imputation turns out to be necessary for us (for EDA, feature importance, NN, RF).
- ❑ Use cloud computing for all team members not their own laptops (avoid crashing and uncertainties).
- ❑ Try larger grids if possible (our grids are still too sparse).
- ❑ All team members focus on one model VS each work on their own.

Lesson learned

What we did good?

- Had clear and solid plan at the beginning
- Each team member is very dedicated to the project
- Pinpoint bottlenecks early and react very fast
- Tuned large amount of parameter combinations

What we can improve?

Team shifted focus twice

-Monday - Wednesday (EDA, Imputation, Importance analysis, Brainstorming)

Thursday (Gave up imputation, tuned on complete training dataset)

Thanks!