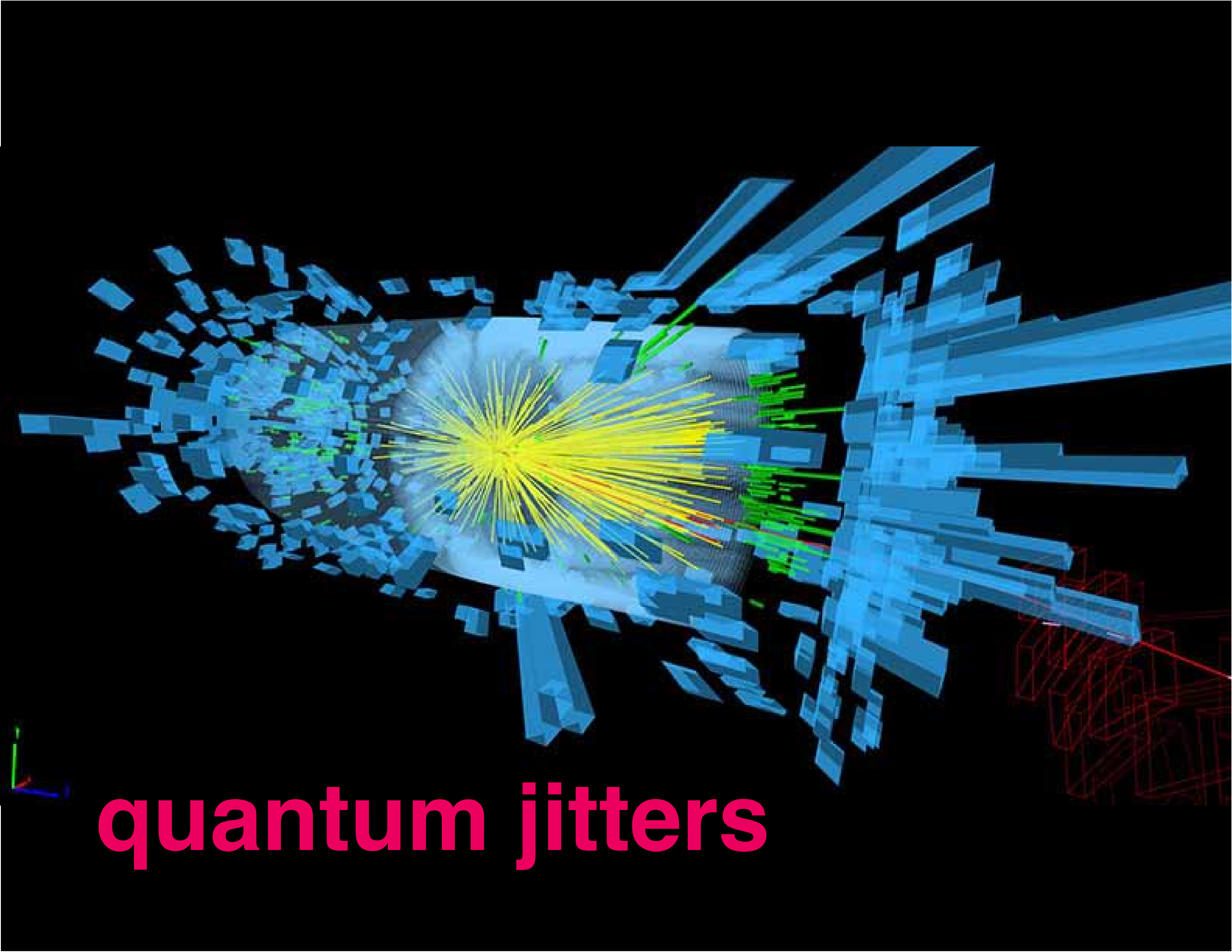

quantum jitters

Missingness & Feature Engineering

STRUCTURALLY MISSING VALUES

The missing values in the dataset (-999) resulted from two sources:

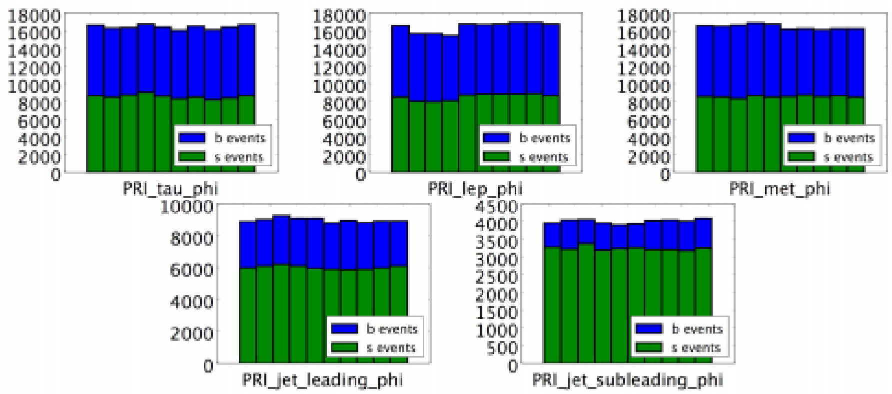
1. Bad estimates of the mass of Higgs boson; and Rotated the angle of the remaining 4 phi columns.
2. Jets: particles that can appear 0, 1, 2, or 3 times in an event.

To deal with this structural missingness, we:

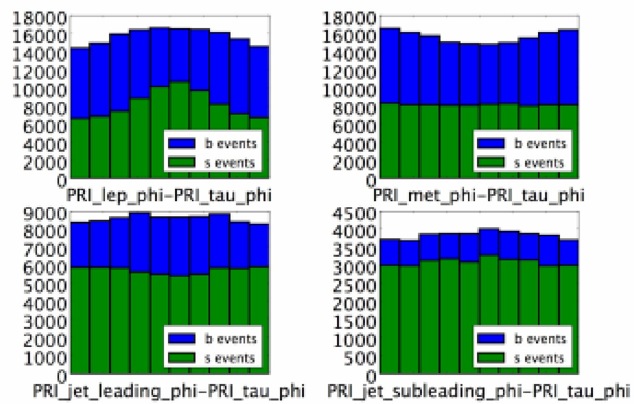
- Converted the -999s into NAs.
- Created a Boolean column to reflect missing(T)/present(F) values in the estimated mass of the Higgs boson (DER_mass_MMC).
- Created 8 binary columns reflecting each combination of presence(0)/absence(1) in the estimated mass of the Higgs boson and the number of jets (0,1,2,3).
- o J0 +M1, J0+M0, J1+M1, J1+M0…
- Imputed the column mean for all the NAs.

For more on the structural nature of the missing values in the Higgs boson dataset, see http://www.jmlr.org/proceedings/papers/v42/cowa14.pdf.

Figures 2(a) and 2(b) show the histograms of the original and rotated features respectively. We can see that features that didn't look informative because they had a uniform distribution for signal and background events are now apparently important with this transformation (see figure, showing a distribution with different parameters in the case of the signal and background events.



(a) The histogram of the features related with the ϕ angle in the reference frame (see Apendix A), abscissa axis is in the $[-\pi, \pi)$ range



(b) Histogram of the new features after a rotation of every event to make PRI_tau_phi equal to zero (see section 4.1), abscissa axis is in the $[0, 2\pi)$ range

Figure 2: Change of variables to create a feature space robust to rotations around the z axis

http://www.jmlr.org/proceedings/papers/v42/diaz14.pdf

FEATURE SELECTION

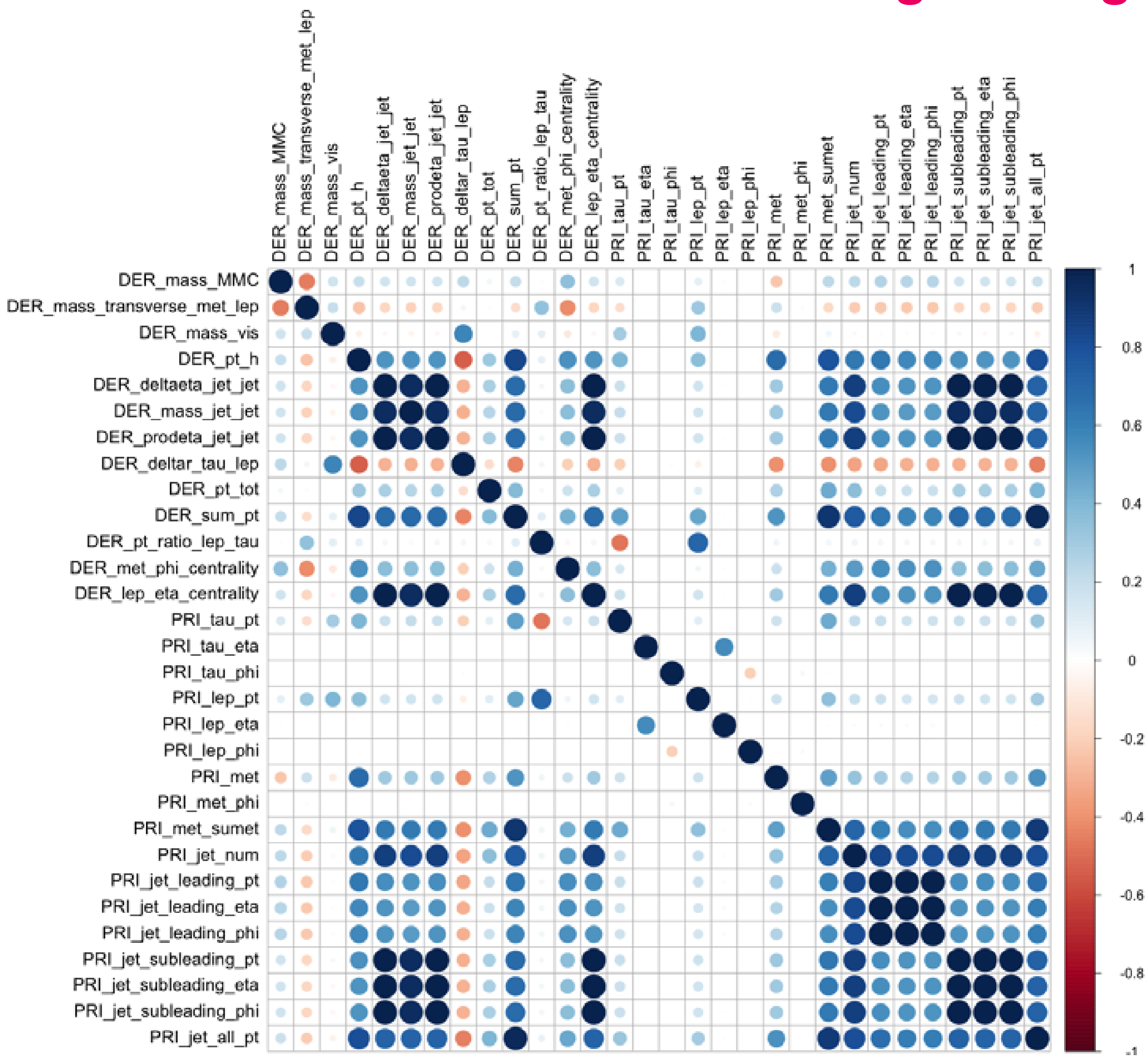
To deal with the risk of overfitting, we:

1. Subtracted the PRI_tau_phi column from the other 4 ‘phi’ columns.
2. Rotated the angle of the remaining 4 phi columns.
3. Deleted all 5 of the non-rotated phi columns, leaving us with 4 phi columns instead of 5.

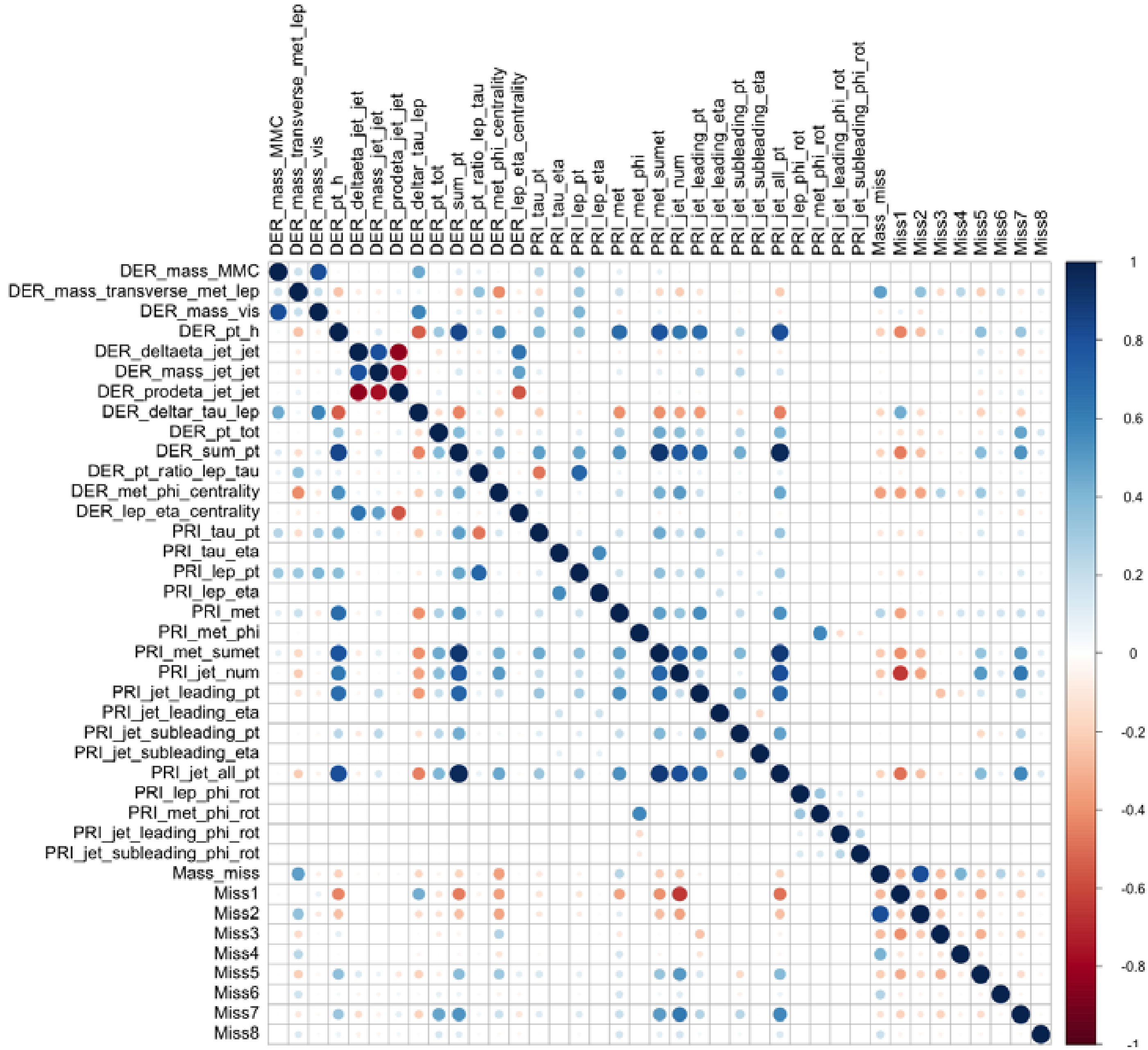
This subtraction + rotation process makes the specific pattern of the phi variables more unique, so that, once the model is trained, it is better able to discriminate between signal and background in the test set.

For more on the logic of this process, see http://www.jmlr.org/proceedings/papers/v42/diaz14.pdf

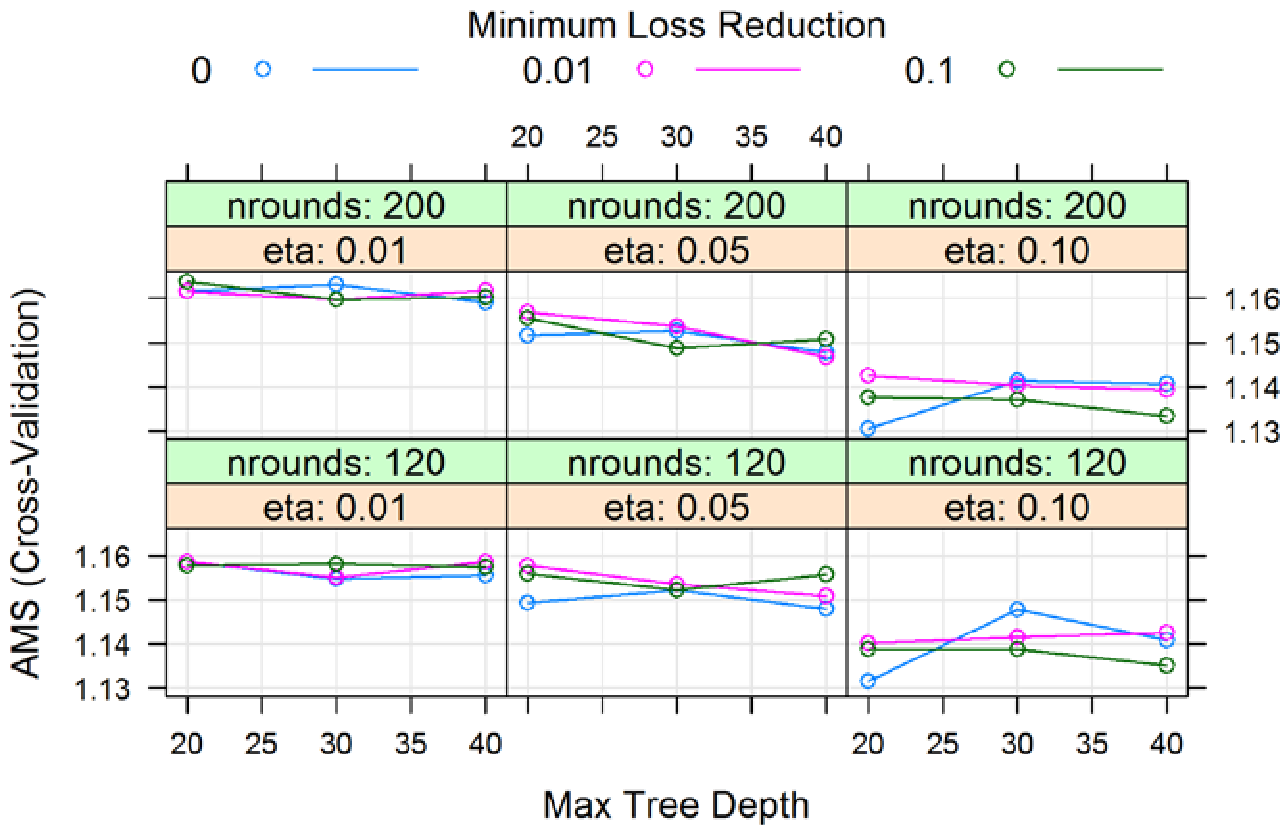
Correlation Plot before Feature Engineering



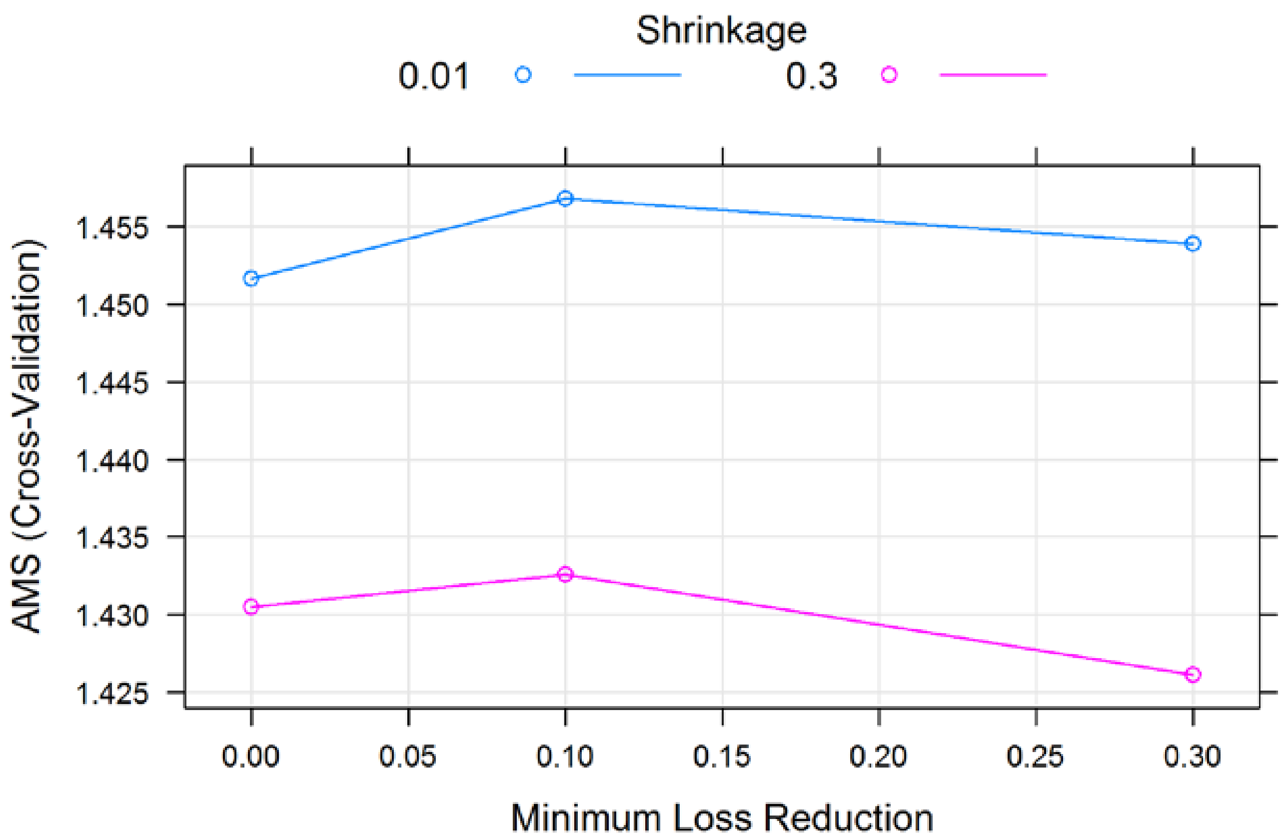
Correlation Plot after Feature Engineering



XGBoost



XGBoost: 1			
Model Parameters		Accuracy	Kaggle score
n.trees	200	0.8214	1.1637
interaction depth	20		
shrinkage	0.01		
gamma	0.1		
K-folds	5		

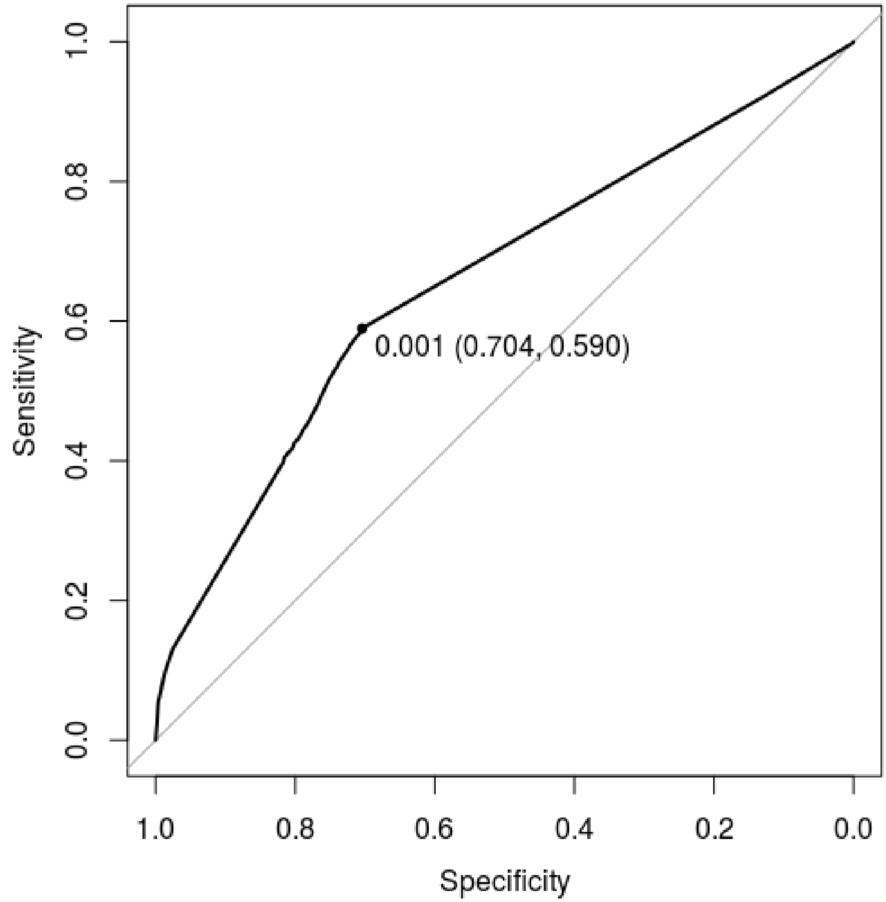


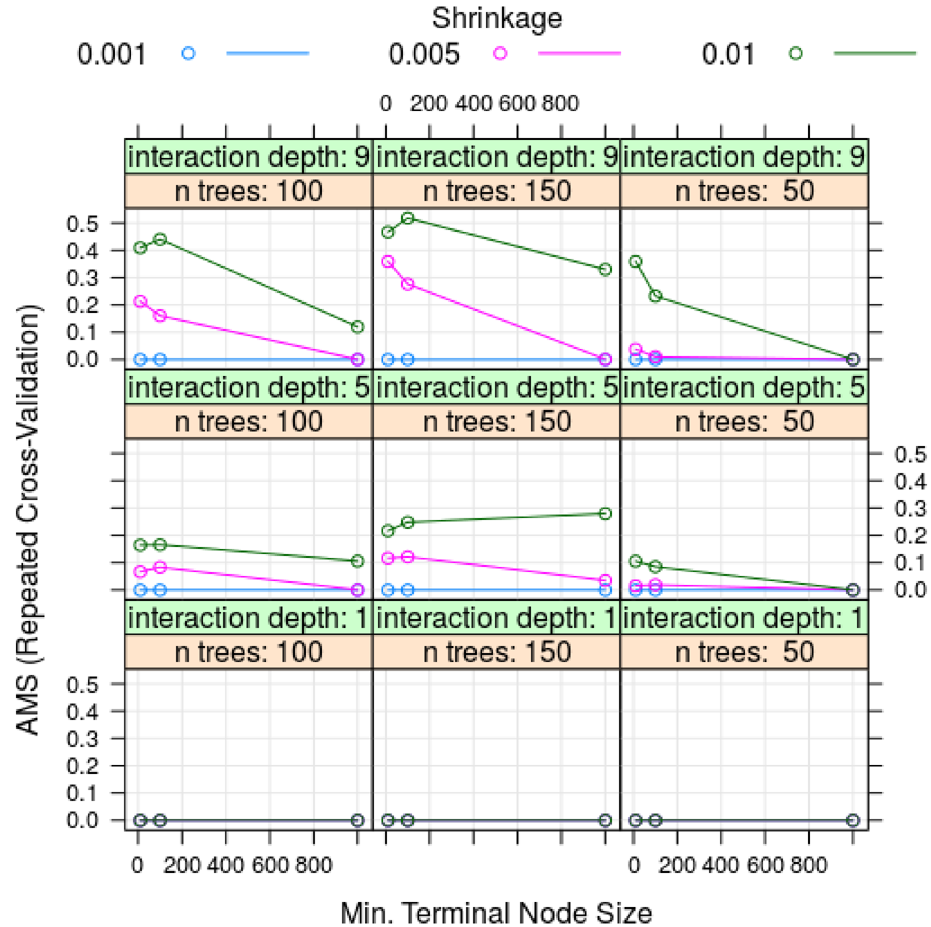
XGBoost: 2			
Model Parameters		Accuracy	Kaggle score
n.trees	300	0.8309	1.474
interaction depth	10		
shrinkage	0.1		
gamma	0.1		
K-folds	3		

- Number of cv folds: 2 - 5
- Number of rounds: 50 - 400
- Eta (shrinkage): 0.001 - 0.6
- Gamma: 0.01 - 0.6
- Max tree depth: 6 -10

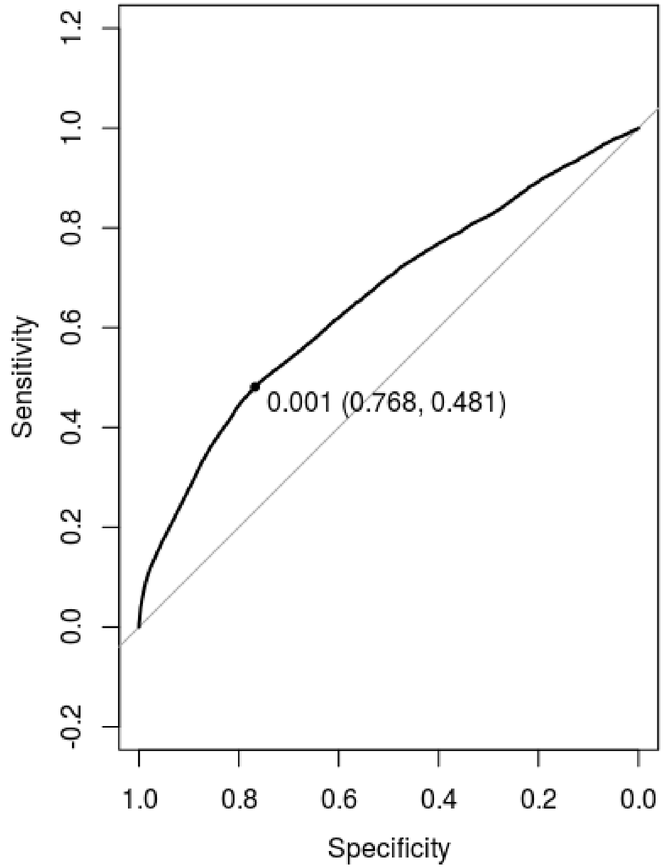
- Insights and Conclusion:
  - AMS scores on 80% train split: 1.2 - 1.8+
  - Accuracy rate of 80+ when applied to 20% test
  - Most likely due to overfitting because the AMS scores on the Kaggle full test data site were much lower, with the highest being about .5.

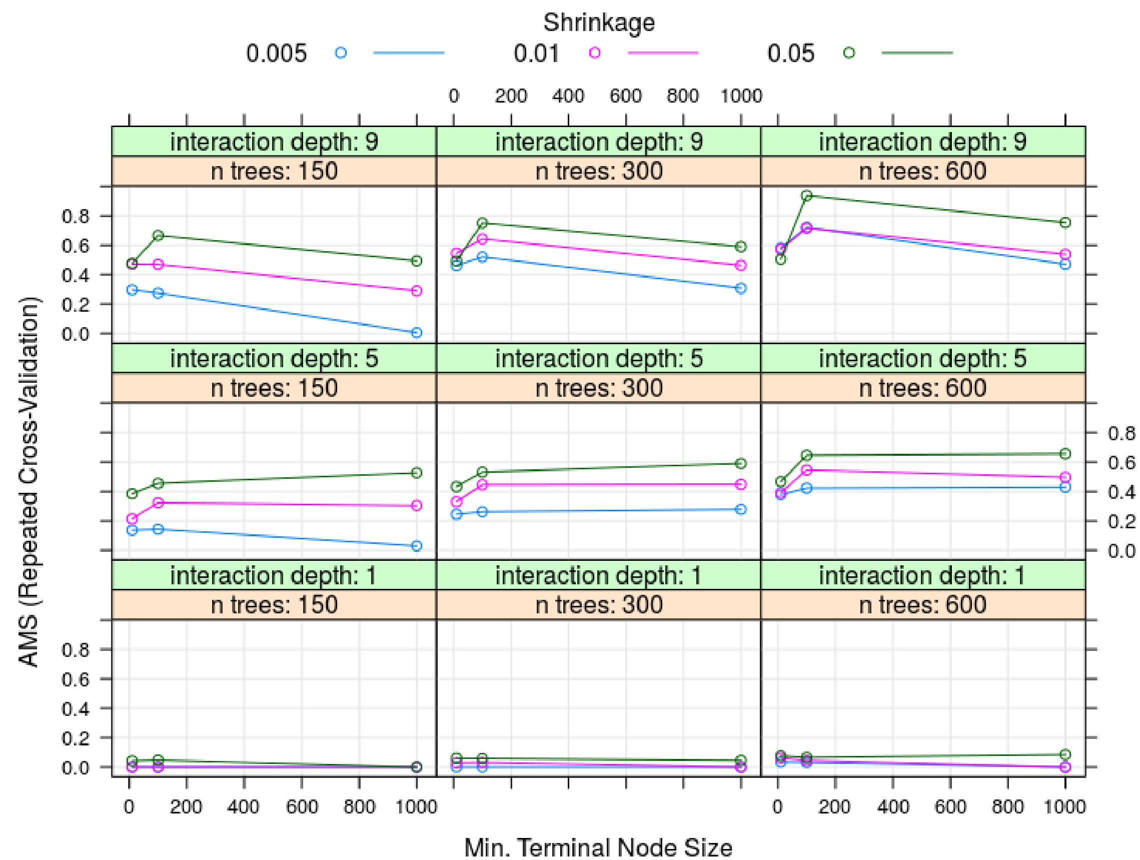
Gradient Boosting Model





GBM: 1					
Model Parameters		AUC		Accuracy	Kaggle score
n.trees	50	AUC	0.6583	0.67282	1.03128
interaction depth	9				
shrinkage	0.01	threshold	0.001		
min obs in bin	10				
K-folds	5				



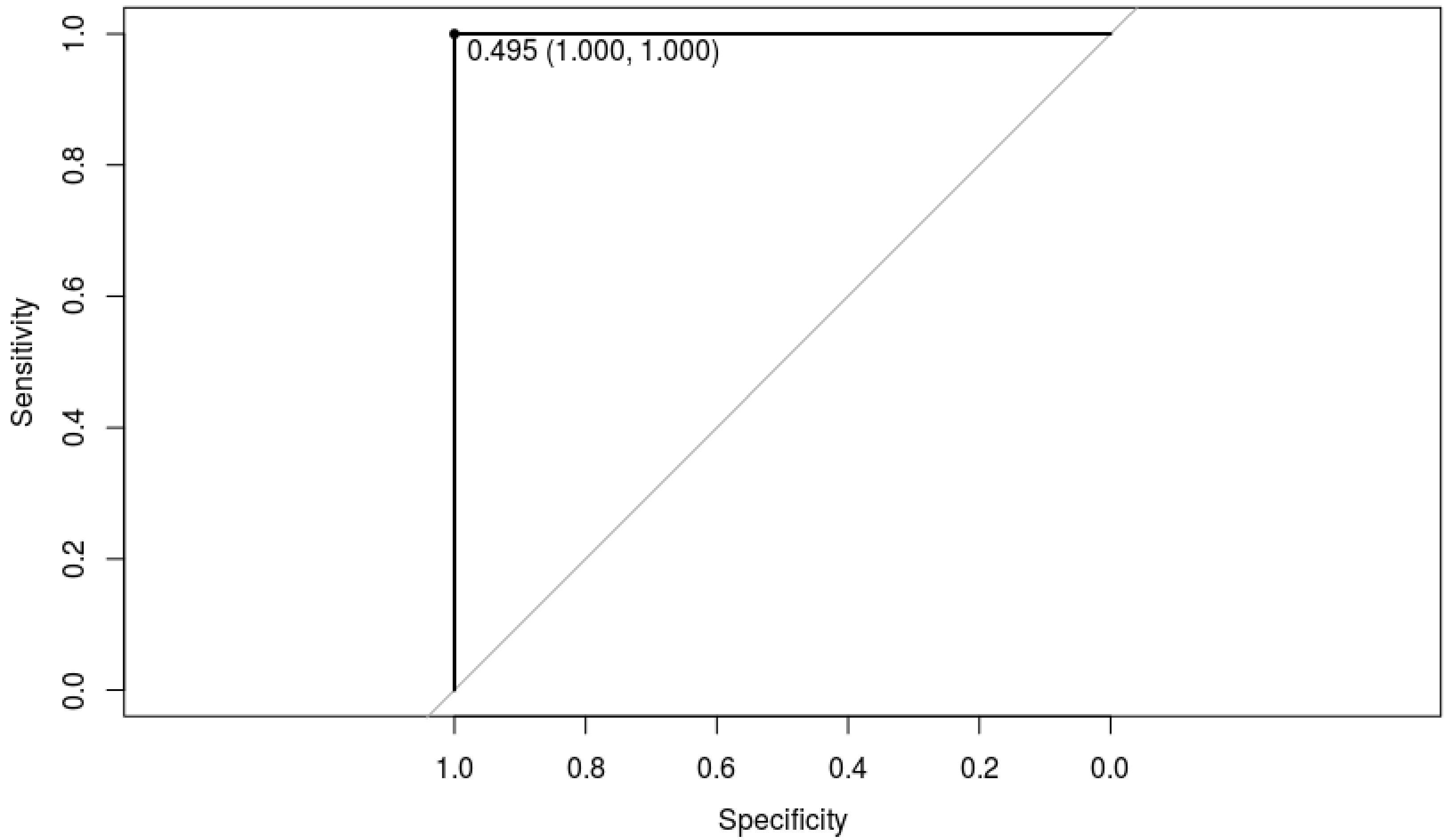


GBM: 2					
Model Parameters		AUC		Accuracy	Kaggle score
n.trees	600	AUC	0.656	0.644	1.06846
interaction depth	9				
shrinkage	0.05	threshold	0.001		
min obs in bin	100				
K-folds	3				

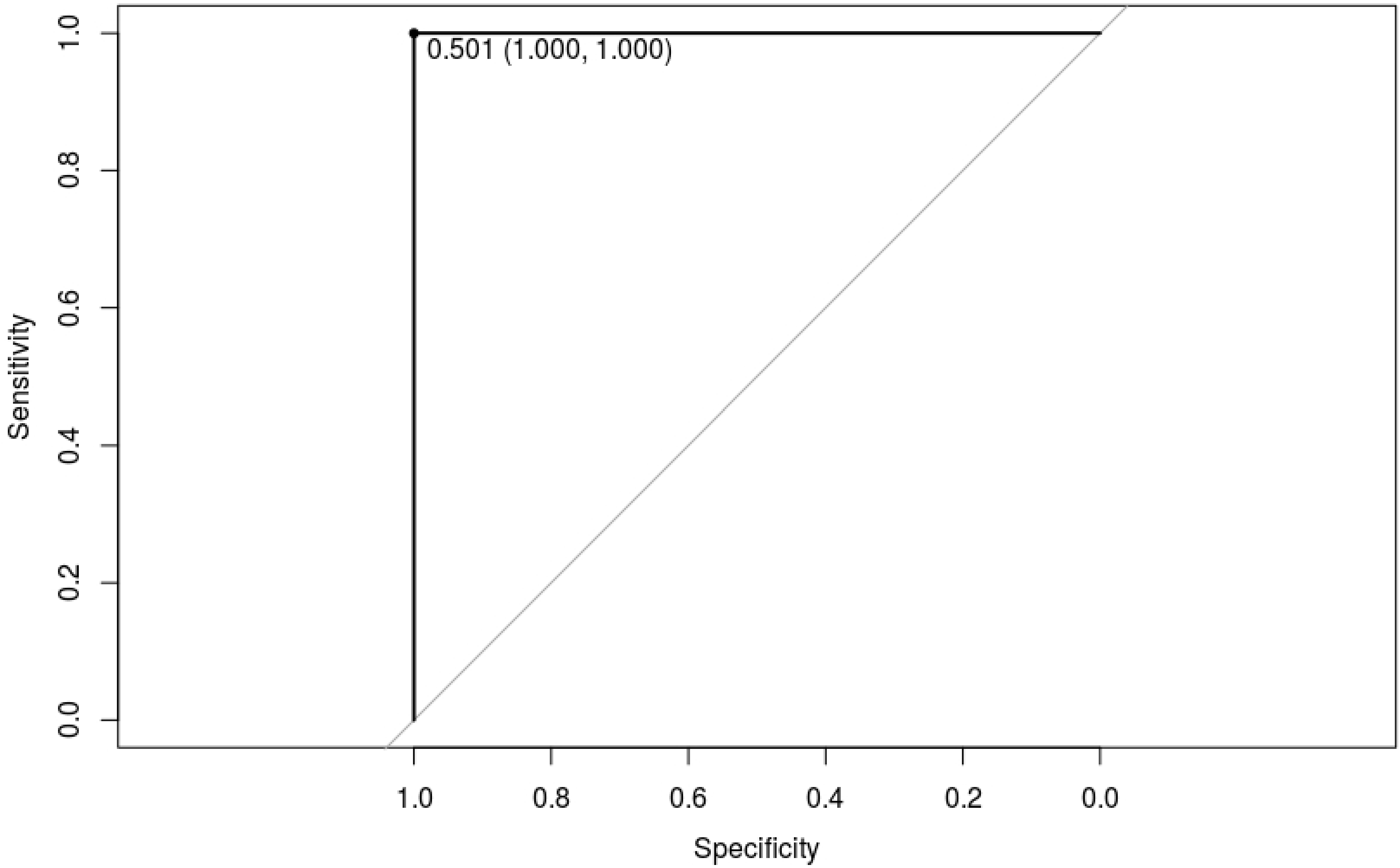
Insights and conclusions:

- Should have increased the number of trees and held the shrinkage parameters constant to compare.
- In the future, I would grow the model slower by increasing the number of trees and starting with a much smaller shinkage parameter.
- For future Kaggle competitions, look at benchmark documents first to get a sense of optimal number of trees.
- In general our boosted models underperformed because of the small number of trees.

Random Forest



Random Forest: 1				
Model Parameters		AUC		Kaggle score
m.try	38	AUC	1	2.8554
		threshold	0.495	

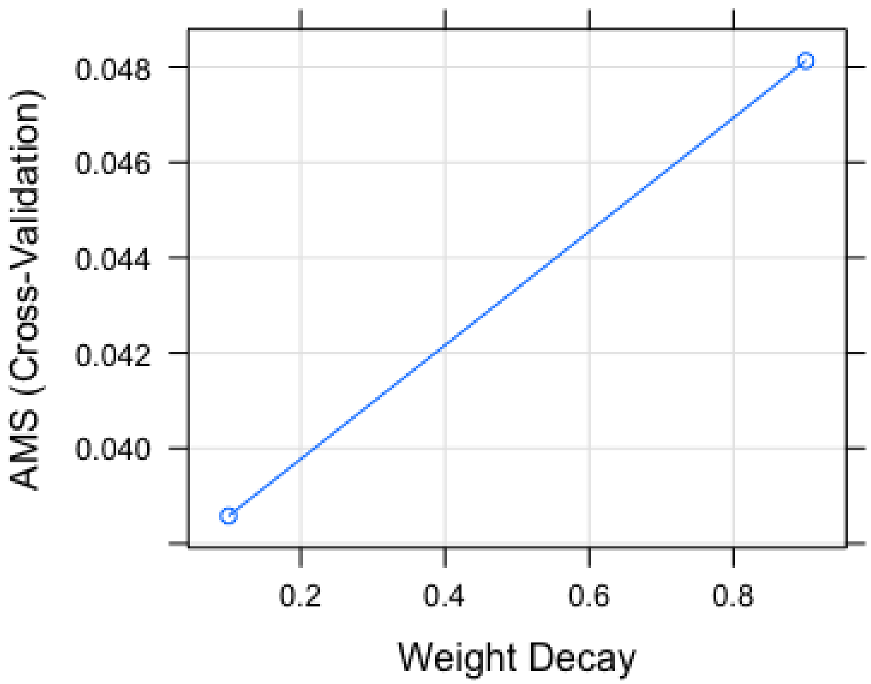


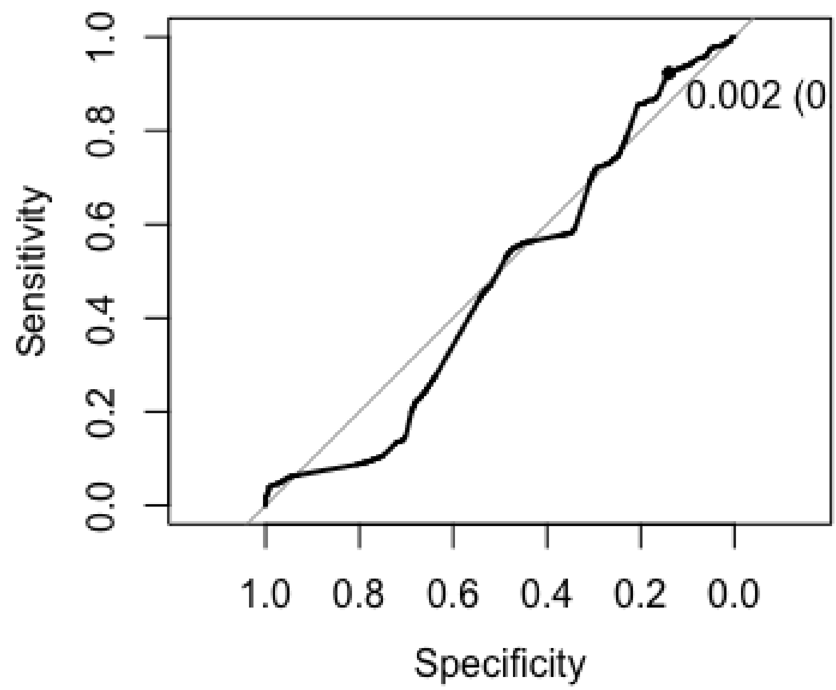
Random Forest: 2				
Model Parameters		AUC		Kaggle score
m.try	7,12,38	AUC	1	2.8554
		threshold	0.501	

Insights & Conclusions

- Because I was using Random Forests, I trained my model on the entire training dataset, using the out of bag error.
- There was no difference between bagging (mtry=38) and a tuneGrid containing 7, 12, and 38.
- Using parallel processing and a server enabled me to process my models more quickly than on my laptop.
- I should have selected the number of trees myself, as practice for comparing a random forest and boosted trees model trained on the same dataset, which would help enable determining the point at which a boosted model outperforms the random forest model on that dataset.

Neural Networks





Neural Net				
Model Parameters		AUC		Accuracy
K-Folds	10	AUC	0.4727	0.39484
hidden layers	20,20,20	threshold	0.002	

Conclusion:

- Adding more hidden layers does not necessarily mean better threshold

- Implement the dropout technique for future analysis

Conclusions & next steps

What we would like to do next:

- Look into ensembling models
- Exploration of other packages other than Caret.
- With more time, we could have written a function that:
    - Identifies values of PRI_tau_eta < 0
    - Converts the signs of all eta values in the same row (- to +, + to -)
    - Imputes the column mean for missing values and randomly assigns a sign (+,-)
    - http://www.jmlr.org/proceedings/papers/v42/diaz14.pdf

	PRI_tau_eta	PRI_lep_eta	PRI_jet_leading_eta	PRI_jet_subleading_eta
1	1.017	2.273	2.150	1.240
2	2.039	0.501	0.725	-999.000
3	-0.705	-0.953	2.053	-999.000
4	-1.655	-0.522	-999.000	-999.000
5	-2.197	0.798	-999.000	-999.000
6	0.371	-0.884	-2.412	0.224
7	1.113	0.675	0.864	0.131
8	0.654	0.506	-0.715	-999.000
9	2.433	0.210	-999.000	-999.000
10	-1.533	-0.317	-2.767	-999.000
11	-0.866	0.126	-999.000	-999.000
12	-0.669	-0.165	-0.790	1.773
13	-0.766	0.722	-0.970	-999.000
14	-0.654	-1.665	-999.000	-999.000
15	1.389	1.856	-999.000	-999.000
16	-1.107	-1.944	-999.000	-999.000
17	0.484	-0.215	-0.766	-999.000

Showing 1 to 18 of 250,000 entries