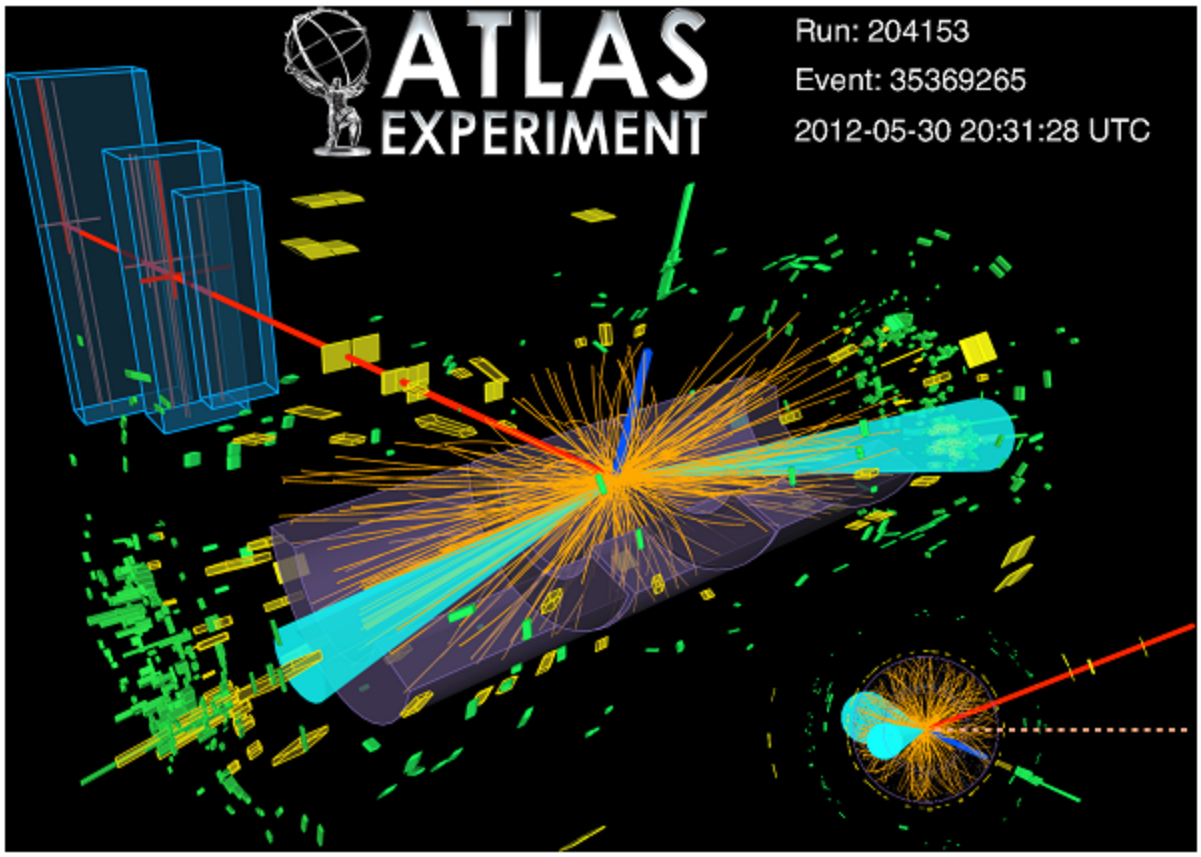
Higgs Boson Machine Learning Challenge



Team: Atom

Members:
Damith Senanayake - 100492H
Chameera Wijebandara - 100598M
Amila Perera - 100379T
Sameera Nilupul - 100597J
Diunuge Buddhika - 100606N

Table of Content

Evidence of Kaggle submission

434	↑657	kickaha	3.64655	9	Mon, 15 Sep 2014 18:39:15 (-3.8d)
435	**↑322**	**Atom**	**3.64655**	**19**	**Sat, 13 Sep 2014 10:26:18 (-29.5h)**
436	↑958	Warana@UOM	3.64655	28	Mon, 15 Sep 2014 10:58:26 (-3d)

Submission	Files	Public Score	Private Score	Selected?
Sat, 13 Sep 2014 10:26:18 90 Edit description	higgs.pred.csv	3.44393	3.49463	
Sat, 13 Sep 2014 09:46:59 XGboost +90% Edit description	higgs.pred.csv.zip	0.32095	0.33027	
Fri, 12 Sep 2014 04:57:09 Edit description	higgs.pred.csv	3.60003	3.64655	
Thu, 11 Sep 2014 13:29:27 np.delete(X_test,4,1) np.delete(X_test,5,1) np.delete(X_test,6,1) np.delete(X_test,26,1) np.delete(X_test,27,1) np.delete(X_test,28,1) imp = Imputer(missing_values=-999.0, strategy='mean', axis=0) imp.fit(X_test) X_test = imp.transform(X_test) X_test = preprocessing.normalize(X_test, norm='l2',axis=0) Edit description	Kaggle_higgs_prediction_output.csv	0.01858	0.02032	
Thu, 11 Sep 2014 13:18:26 Edit description	Kaggle_higgs_prediction_output.csv	3.30176	3.36013	
Thu, 11 Sep 2014 13:05:38 np.delete(X_test,4,1) np.delete(X_test,5,1) np.delete(X_test,6,1) np.delete(X_test,26,1) np.delete(X_test,27,1) np.delete(X_test,28,1) Edit description	Kaggle_higgs_prediction_output.csv	3.29352	3.35249	
Thu, 11 Sep 2014 12:46:13 np.delete(X_test,4,1) np.delete(X_test,5,1) np.delete(X_test,6,1) np.delete(X_test,26,1) np.delete(X_test,27,1) np.delete(X_test,28,1) Edit description	Kaggle_higgs_prediction_output.csv ▼ Submission info/warnings	0.00000	0.00000	⚠
Thu, 28 Aug 2014 14:53:19 imp = Imputer(missing_values=-999.0, strategy='mean', axis=0) imp.fit(X_test) X_test = imp.transform(X_test) X_test = preprocessing.normalize(X_test, norm='l2',axis=0) l_test = list(data_test[:,0]) Edit description	Kaggle_higgs_prediction_output.rar	0.00815	0.01050	

Thu, 28 Aug 2014 14:53:19 imp = Imputer(missing_values=-999.0, strategy='mean', axis=0) imp.fit(X_test) X_test = imp.transform(X_test) X_test = preprocessing.normalize(X_test, norm='l2',axis=0) l_test = list(data_test[:,0]) Edit description	Kaggle_higgs_prediction_output.rar	0.00815	0.01050	
Thu, 28 Aug 2014 05:33:44 preprocessing.normalize(X_train, norm='l2') GradientBoostingClassifier(init=None, learning_rate=0.1, loss='deviance', max_depth=10, max_features=30, min_samples_leaf=400, min_samples_split=2, n_estimators=50, random_state=None, subsample=1.0, verbose=10) Edit description	Kaggle_higgs_prediction_output.csv	3.18046	3.26830	
Thu, 28 Aug 2014 04:33:07 GBC(n_estimators=50, max_depth=10,min_samples_leaf=400,max_features=30,verbose=10) Edit description	Kaggle_higgs_prediction_output.csv	3.42478	3.57132	
Wed, 27 Aug 2014 13:14:02 Edit description	Kaggle_higgs_prediction_output.rar	3.38713	3.43819	
Sun, 20 Jul 2014 09:55:03 Edit description	submission.csv	2.06137	2.06606	
Sun, 20 Jul 2014 09:39:45 Edit description	submission.csv	0.76312	0.76379	
Sun, 20 Jul 2014 08:49:34 Edit description	submission.csv	0.56718	0.56766	
Sun, 20 Jul 2014 08:45:06 Edit description	submission.csv ▼ Submission info/warnings	0.00000	0.00000	⚠
Sun, 20 Jul 2014 08:32:23 Edit description	submission.csv ▼ Submission info/warnings	0.00000	0.00000	⚠
Wed, 16 Jul 2014 09:21:52 Edit description	submission.rar	2.05876	2.05826	
Wed, 16 Jul 2014 07:43:59 Edit description	random_submission.rar	0.58477	0.58648	

Introduction

The Higgs Boson Machine Learning competition is a competition hosted on Kaggle and sponsored by CERN and the ATLAS experiment. The goal of this was to use machine learning algorithms to effectively predict the behaviour of Higgs boson particles. High Energy Physics (HEP) has been using Machine Learning (ML) techniques such as boosted decision trees (paper) and neural nets since the 90s. These techniques are now routinely used for difficult tasks such as the Higgs boson search. In the competition, they provided a data set containing a mixture of simulated signal and background events, built from simulated events provided by the ATLAS collaboration at CERN. We had to develop machine learning algorithm for signal/background separation.

The ATLAS experiment and the CMS experiment recently claimed the discovery of the Higgs boson [1, 2]. The discovery was acknowledged by the 2013 Nobel prize in physics given to Francois Englert and Peter Higgs. This particle was theorized almost 50 years ago to have the role of giving mass to other elementary particles. It is the final ingredient of the Standard Model of particle physics, ruling subatomic particles and forces. The experiments are running at the Large Hadron Collider (LHC) at CERN (the European Organization for Nuclear Research), Geneva, which began operating in 2009 after about 20 years of design and construction, and which will continue operating for at least the next 10 years. The Higgs boson has many different processes through which it can decay. When it decays, it produces other particles. In physics, a decay into specific particles is called a channel. The Higgs boson has been seen first in three distinct decay channels which are all boson pairs. One of the next important topics is to seek evidence on the decay into fermion pairs, namely tau-leptons or b-quarks, and to precisely measure their characteristics. The first evidence of the H to tau tau channel was recently reported by the ATLAS experiment [3], which, in the rest of this paper, will be referred to as ”the reference document”. The subject of the Challenge is to try and improve on this analysis.

The goal of the Challenge

The goal of the Challenge is to improve the procedure that produces the selection region. We are provided a training set with signal/background labels and with weights, a test set (without labels and weights), and a formal objective representing an approximation of the median significance (AMS) of the counting test. The objective is a function of the weights of selected events. We expect that significant improvements are possible by re-visiting some of the ad hoc choices in the standard procedure, or by incorporating the objective function or a surrogate into the classifier design.

Problem Description

A key property of any particle is how often it decays into other particles.ATLAS is a particle physics experiment taking place at the Large Hadron Collider at CERN that searches for new particles and processes using head-on collisions of protons of extraordinarily high energy. The ATLAS experiment has recently observed a signal of the Higgs boson decaying into two tau particles, but this decay is a small signal buried in background noise.

The data set containing a mixture of simulated signal and background events. Predictor variables are prefixed with PRI and DER. Variables prefixed with PRI (for PRImitives) are “raw” quantities about the bunch collision as measured by the detector and variables prefixed with DER (for DERived) are quantities computed from the primitive features, which were selected by the physicists of ATLAS. Data set contains 250000 events of training set and 550000 events of test set.

According to the Kaggle, the basic introduction to the problem is;

“Discovery of the long awaited Higgs boson was announced July 4, 2012 and confirmed six months later. 2013 saw a number of prestigious awards, including a Nobel prize. But for physicists, the discovery of a new particle means the beginning of a long and difficult quest to measure its characteristics and determine if it fits the current model of nature.

A key property of any particle is how often it decays into other particles.ATLAS is a particle physics experiment taking place at the Large Hadron Collider at CERN that searches for new particles and processes using head-on collisions of protons of extraordinarily high energy. The ATLAS experiment has recently observed a signal of the Higgs boson decaying into two tau particles, but this decay is a small signal buried in background noise.

The goal of the Higgs Boson Machine Learning Challenge is to explore the potential of advanced machine learning methods to improve the discovery significance of the experiment. No knowledge of particle physics is required. Using simulated data with features characterizing events detected by ATLAS, your task is to classify events into "tau tau decay of a Higgs boson" versus "background."

The winning method may eventually be applied to real data and the winners may be invited to CERN to discuss their results with high energy physicists.”

Data Set

File descriptions

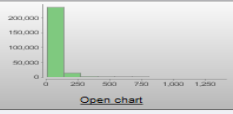
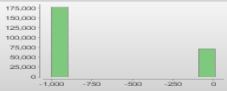
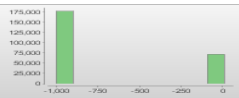
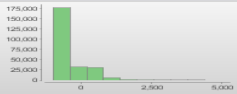
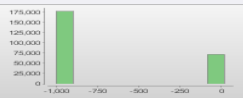
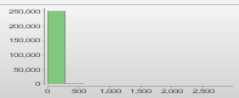
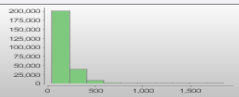
1. training.csv - Training set of 250000 events, with an ID column, 30 feature columns, a weight column and a label column.
2. test.csv - Test set of 550000 events with an ID column and 30 feature columns.
3. random_submission - Sample submission file in the correct format. File format is described on the Evaluation page.
4. HiggsBosonCompetition_AMSMetric - Python script to calculate the competition evaluation metric.

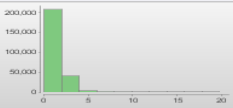
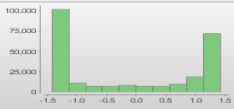
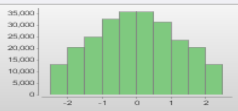
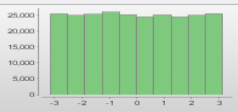
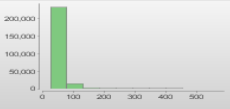
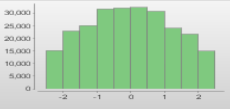
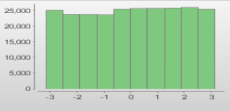
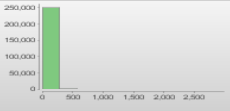
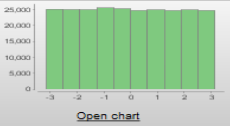
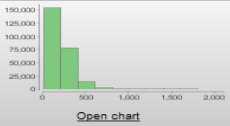
HIgh level details of the data fields

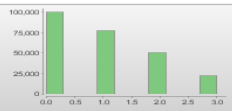
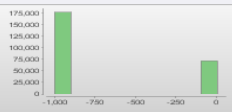
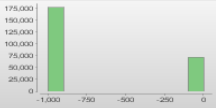
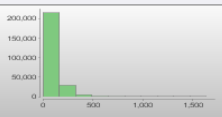
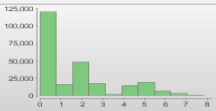
- all variables are floating point, except PRI_jet_num which is integer
- variables prefixed with PRI (for PRImitives) are “raw” quantities about the bunch collision as measured by the detector.
- variables prefixed with DER (for DERived) are quantities computed from the primitive features, which were selected by the physicists of ATLAS
- it can happen that for some entries some variables are meaningless or cannot be computed; in this case, their value is −999.0, which is outside the normal range of all variables

Pre-analysis of data

We created MySQL database for pre-processing and analysis process. We used python and Rapid minor studio for initial processing...

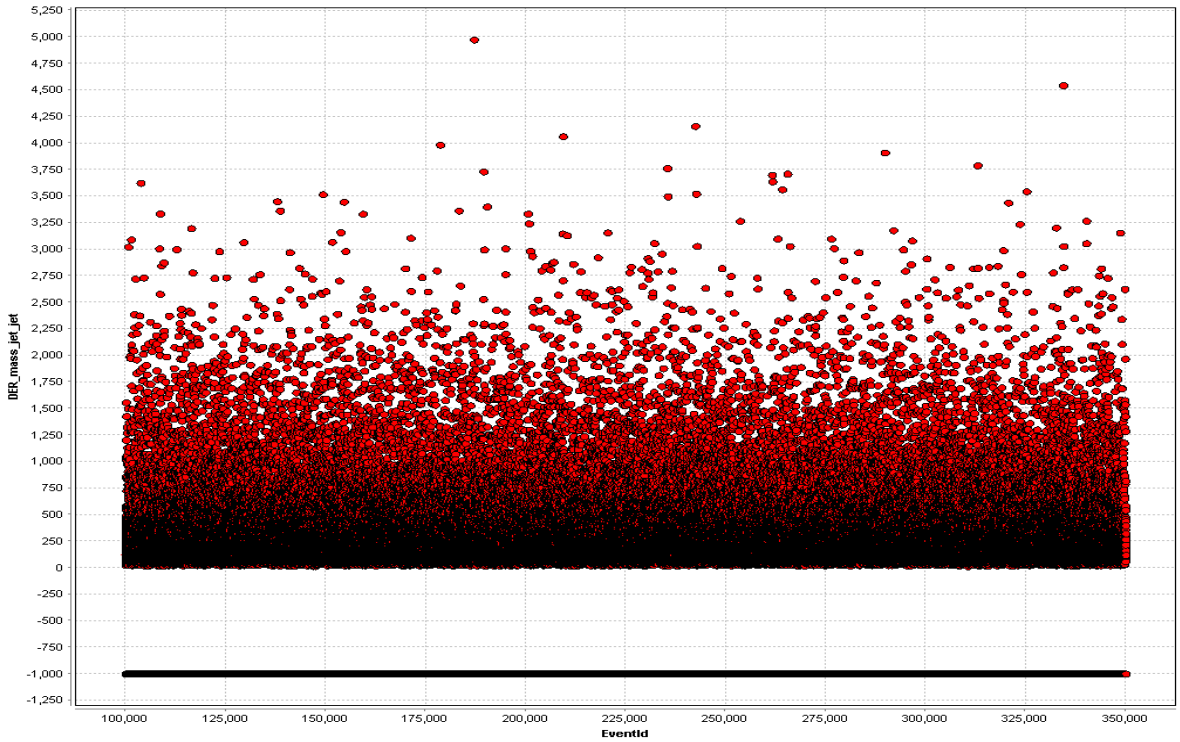
EventId	Integer	0	 Open chart	Min 100000	Max 349999	Average 224999.500	Deviation 72168.928
DER_mass_MMC	Real	0	 Open chart	Min -999	Max 1192.026	Average -49.023	Deviation 406.346
DER_mass_transverse_me...	Real	0	 Open chart	Min 0	Max 690.075	Average 49.240	Deviation 35.345
DER_mass_vis	Real	0	 Open chart	Min 6.329	Max 1349.351	Average 81.182	Deviation 40.829
DER_pt_h	Real	0	 Open chart	Min 0	Max 2834.999	Average 57.896	Deviation 63.656
DER_deltaeta_jet_jet	Real	0	 Open chart	Min -999	Max 8.503	Average -708.421	Deviation 454.481
DER_deltaeta_jet_jet	Real	0	 Open chart	Min -999	Max 8.503	Average -708.421	Deviation 454.481
DER_mass_jet_jet	Real	0	 Open chart	Min -999	Max 4974.979	Average -601.237	Deviation 657.972
DER_prodeta_jet_jet	Real	0	 Open chart	Min -999	Max 16.690	Average -709.357	Deviation 453.020
DER_deltar_tau_lep	Real	0	 Open chart	Min 0.208	Max 5.684	Average 2.373	Deviation 0.783
DER_pt_tot	Real	0	 Open chart	Min 0	Max 2834.999	Average 18.917	Deviation 22.273
DER_sum_pt	Real	0	 Open chart	Min 46.104	Max 1852.462	Average 158.432	Deviation 115.706

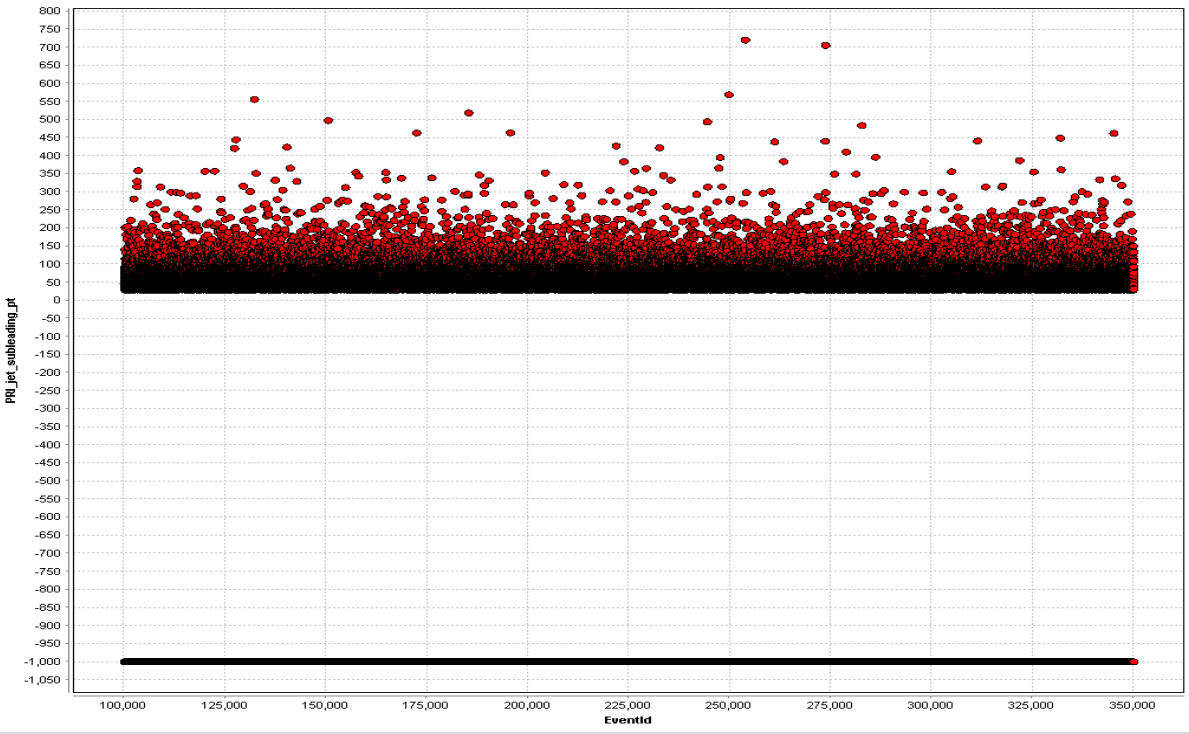
DER_pt_ratio_lep_tau	Real	0	 Open chart	Min 0.047	Max 19.773	Average 1.438	Deviation 0.845
DER_met_phi_centrality	Real	0	 Open chart	Min -1.414	Max 1.414	Average -0.128	Deviation 1.194
DER_lep_eta_centrality	Real	0	 Open chart	Min -999	Max 1	Average -708.985	Deviation 453.597
PRI_tau_pt	Real	0	 Open chart	Min 20	Max 764.408	Average 38.707	Deviation 22.412
PRI_tau_eta	Real	0	 Open chart	Min -2.499	Max 2.497	Average -0.011	Deviation 1.214
PRI_tau_phi	Real	0	 Open chart	Min -3.142	Max 3.142	Average -0.008	Deviation 1.817
PRI_lep_pt	Real	0	 Open chart	Min 26	Max 560.271	Average 46.660	Deviation 22.065
PRI_lep_eta	Real	0	 Open chart	Min -2.505	Max 2.503	Average -0.020	Deviation 1.265
PRI_lep_phi	Real	0	 Open chart	Min -3.142	Max 3.142	Average 0.044	Deviation 1.817
PRI_met	Real	0	 Open chart	Min 0.109	Max 2842.617	Average 41.717	Deviation 32.895
PRI_met_phi	Real	0	 Open chart	Min -3.142	Max 3.142	Average -0.010	Deviation 1.812
PRI_met_sumet	Real	0	 Open chart	Min 13.678	Max 2003.976	Average 209.797	Deviation 126.500

PRI_jet_num	Integer	0	 Open chart	Min 0	Max 3	Average 0.979	Deviation 0.977
PRI_jet_leading_pt	Real	0	 Open chart	Min -999	Max 1120.573	Average -348.330	Deviation 532.963
PRI_jet_leading_eta	Real	0	 Open chart	Min -999	Max 4.499	Average -399.254	Deviation 489.338
PRI_jet_leading_phi	Real	0	 Open chart	Min -999	Max 3.141	Average -399.260	Deviation 489.334
PRI_jet_subleading_pt	Real	0	 Open chart	Min -999	Max 721.456	Average -692.381	Deviation 479.875
PRI_jet_subleading_eta	Real	0	 Open chart	Min -999	Max 4.500	Average -709.122	Deviation 453.385
PRI_jet_subleading_phi	Real	0	 Open chart	Min -999	Max 3.142	Average -709.119	Deviation 453.389
PRI_jet_all_pt	Real	0	 Open chart	Min 0	Max 1633.433	Average 73.065	Deviation 98.016
Weight	Real	0	 Open chart	Min 0.002	Max 7.823	Average 1.647	Deviation 1.875
Label	Polynominal	0	 Open chart	Least s (85667)	Most b (164333)	Values b (164333), s (85667) Details...	

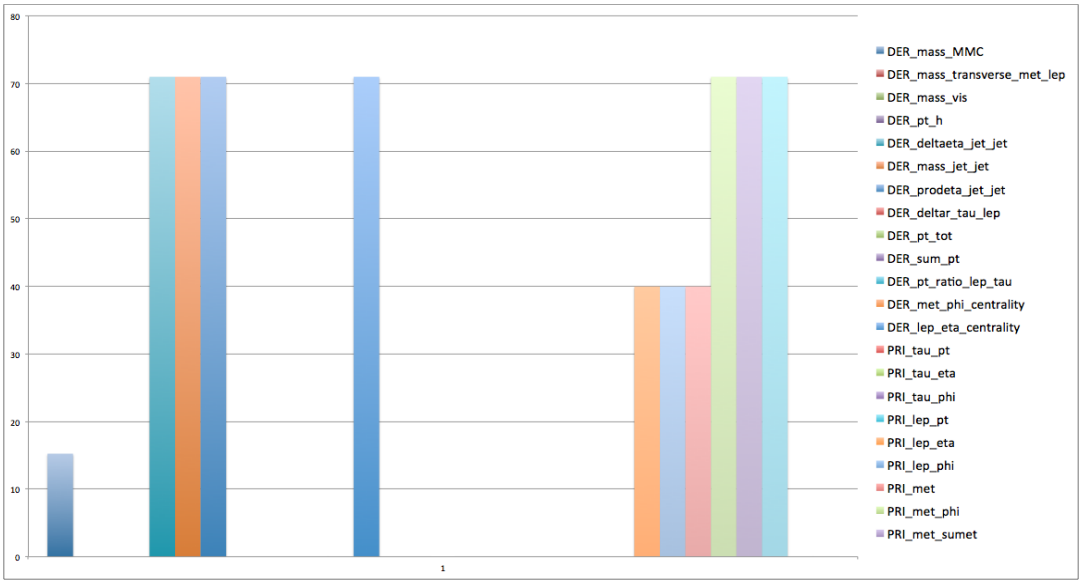
Missing values

We analyzed the input data for missing values. Some of missing values data is given below.



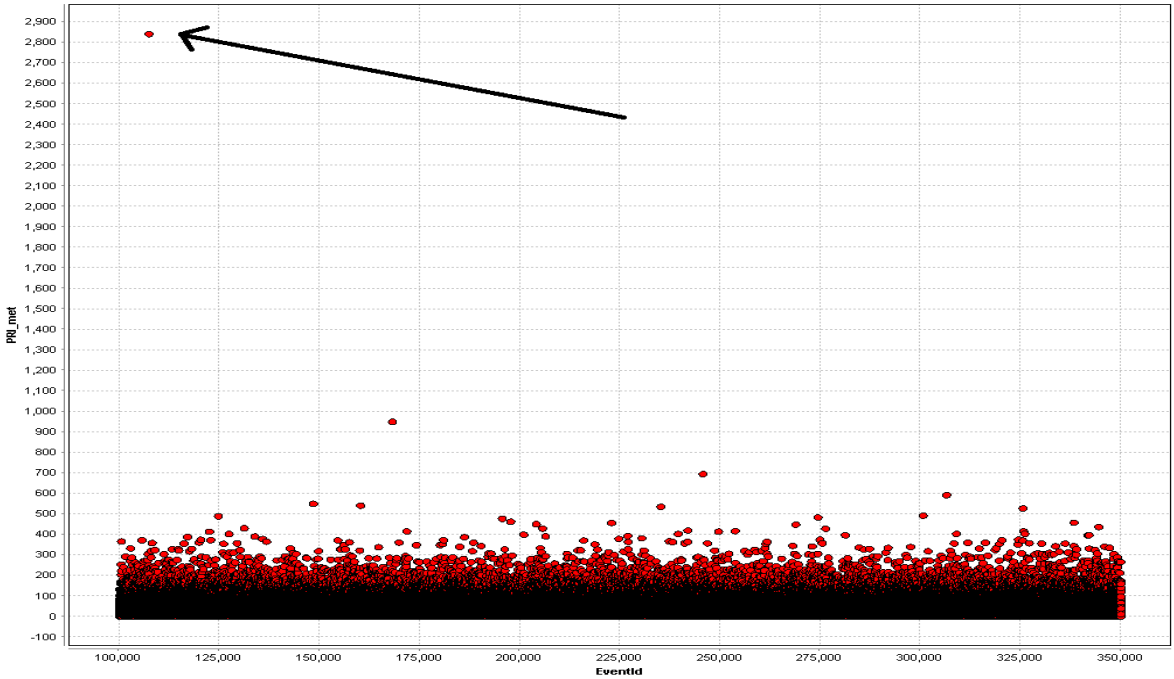


Seven data fields has missing values up to 70% and three data fields have missing values up to 40%. One data field(DER_mass_MMC) has 15.24% missing values. All the other data fields contains no missing values. Missing values percentages for predictor variables are given below.



Outliers

Outliers are detected in some of attributes in the data set. We used several methods for remove that outliers depend on the data type. Possible recognized outlier analysis chart is given below.



Special Observations

One of the most notable observations was that as is exemplified by above diagrams, most of the variables have a very low variance. And dropping low variance attributes from the training model resulted in a drop of performance. And finding a combination of low variance was also a difficult task given the number of attributes.

Pre-Processing of Data

Filling missing values

It was concluded that missing values did not matter that much in this particular contest because we have tried replacing these with min, max, median, mean, random-forest predictions, glm predictions and so forth. It did not vastly change the leaderboard scores (or AUC) compared to doing nothing and leaving everything at -999.0. It was also noted that a lot of these benchmarks change these to NAs and just run a training model which handled NA values. It did not change a score of 3.6 into a 3.7 regardless of the method used to handle missing values.

The academic literature might say outliers matter. Maybe in some theoretical toy example of OLS versus LAD with huge leverage but most algorithms (gbm, rf) seem fairly robust especially if we use ensemble.

```
imp = Imputer(missing_values=-999.0, strategy='mean', axis=0)
imp.fit(X_test)

X_test = imp.transform(X_test)
X_test = preprocessing.normalize(X_test, norm='l2',axis=0)
l_test = list(data_test[:,0])
```

Remove outliers

Some numeric feature variables contained outliers data. We filled them with average values. Outliers was defined as points which had a distance from the median that exceeds 1.5 times the interquartile range of the given feature

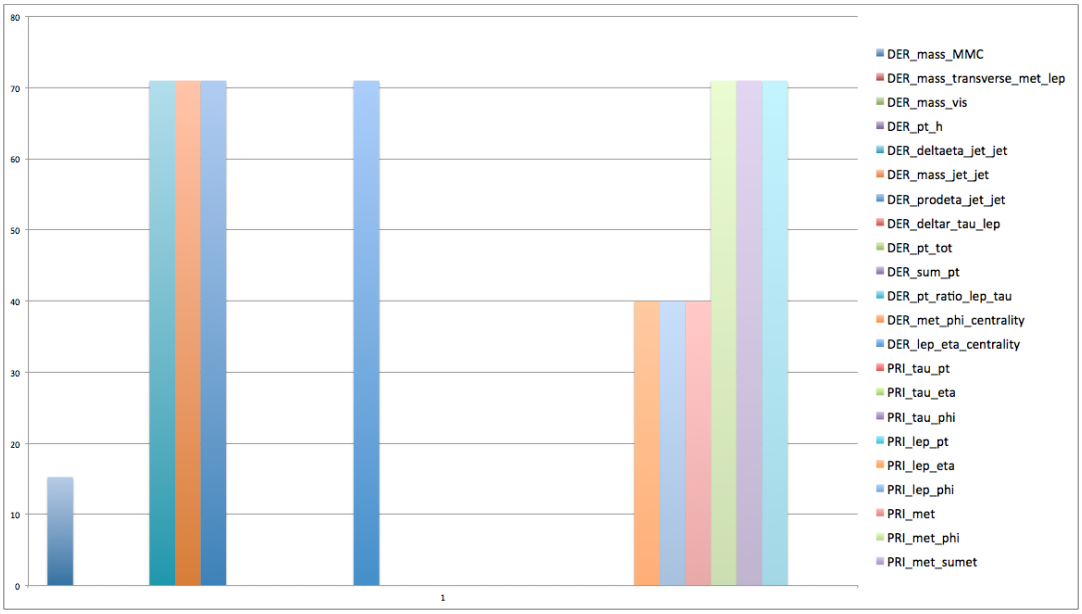
Data Selection and Transformations

Attribute reduction

Before applying any methods to the dataset, we first did a Principal Component Analysis on the attribute to identify priority of attributes but it was not that successful.

```
pca = PCA(n_components=2)
pca.fit(X)
PCA(copy=True, n_components=2, whiten=False)
```

We removed some features with a high percentage of missing values. Missing values percentages for predictor variables are given below.




```
np.delete(X_test,4,1)
np.delete(X_test,5,1)
np.delete(X_test,6,1)
np.delete(X_test,26,1)
np.delete(X_test,27,1)
np.delete(X_test,28,1)
```

Although, when features are removed with high percentage of missing values accuracy of the results seems to reduced.

Normalization

When normalizing the numeric predictor variables, accuracy of the model reduced unexpectedly.

```
preprocessing.normalize(X_train, norm='l2')
```

For final algorithm we avoid the normalizing for predictor variables.

Data Mining and Prediction

Model Selection

Naive Bayes Classifier

The Naive Bayes Classifier was the first benchmark that we broke with the Scikit Learn packages. It was also the first submission that we got a higher score than the random submission benchmark.

Pybrain Neural Network

We tried to solve this problem by training a neural network to classify each events into “Higgs Boson” or “Background”. We used pybrain machine learning library and we used a neural network with following specifications.

- Input layer : 30 Sigmoid Neurons
- Hidden Layer : 45 Sigmoid Neurons
- Output Layer : 1 Sigmoid Neuron

This is a fully connected neural network and we used back propagation as the training algorithm. Since we didn’t get any good results from this method and training neural network takes a long time, we decided to change our approach.

```
fnn = FeedForwardNetwork()
inLayer = SigmoidLayer(cd.indim)
hiddenLayer = SigmoidLayer(int(cd.indim*1.5))
outLayer = SigmoidLayer(cd.outdim)

fnn.addInputModule(inLayer)
fnn.addModule(hiddenLayer)
fnn.addOutputModule(outLayer)

in_to_hidden = FullConnection(inLayer, hiddenLayer)
hidden_to_out = FullConnection(hiddenLayer, outLayer)

fnn.addConnection(in_to_hidden)
fnn.addConnection(hidden_to_out)

fnn.sortModules()

trainer = BackpropTrainer( fnn, dataset=cd, momentum=0.1, verbose=True,
weightdecay=0.01)
```

Multiboost Classifier

The multiboost classifier is a benchmark classification provided by the competition hosts. The strategy of multiboost is to iteratively run an AdaBoost classification 3000 times to get better results. This method was quickly disregarded, however, due to the fact that very little flexibility and transparency about the algorithm itself was provided.

SVM

Support Vector Machines employ the technique of mapping a featureset into a higher dimension and finding a hyperplane that best separates two sets of mapped data points on the new vector space. However, using this in this particular case did not result in a good score.

```
clf = svm.SVC()
clf.fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=3,
gamma=0.0, kernel='rbf', max_iter=-1, probability=False, random_state=None,
shrinking=True, tol=0.001, verbose=False)
```

Gradient Boosting Classifier

The gradient boost algorithm is an ensemble method where weaker learners (such as decision trees) combine to make a stronger learner. The algorithm, in simple terms, is one that uses a differentiable loss function to calculate the adjustments needed to be made to a consecutive successor learner in an iterative learning sequence. The adjustment is done through proxy by taking the steepest descent of the loss function with respect to the hypothetical pseudo-residuals. It is regarded as one of the first boosting algorithms to render arbitrarily better results as opposed to the primitive AdaBoost (Adaptive Boosting).

Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$, number of iterations M.

Algorithm:

1. Initialize model with a constant value:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$$

2.
3. For *m* = 1 to *M*:
    1. Compute so-called *pseudo-residuals*:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n.$$

    2. Fit a base learner $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
    3. Compute multiplier γ_m by solving the following one-dimensional optimization problem:

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right).$$

    4. Update the model:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

4. Output $F_M(x)$.

An ASM score of about 3.571 was obtained by simply using gradient boosting on the data set.

```
# Train the GradientBoostingClassifier using our good features

print 'Training classifier'
gbc = GBC(n_estimators=20, max_depth=10,min_samples_leaf=400,max_features=20,
verbose=10,learning_rate=0.1)
gbc.fit(X_train,Y_train)
```

XGBoost Classifier

This is the main method used for final results. The XGboost classifier contains a modified version of the gradient boosting algorithm. The eXtreme Gradient Boost classifier is a modified version of this, developed by Tianqi Chen, including generalized linear model and gradient boosted regression tree.. The library is parallelized using OpenMP. Using the package of the implementation that he has provided on the data set, we were able to achieve an accuracy of 3.64655.

When constructing the xgboost.DMatrix from numpy array of training data, we treated -999.0 as missing value for the algorithm. The algorithm run parallelly in 8 threads with average training time of around 1 min (Executed environment: Core i5 1.8GHz, 4GB Ram, Ubuntu 14.04 LTS, Python 2.7) and 1 min prediction time.

```
xgmat = xgb.DMatrix( data, label=label, missing = -999.0, weight=weight )
bst = xgb.train( plst, xgmat, num_round, watchlist );


xgmat = xgb.DMatrix( data, missing = -999.0 )
bst = xgb.Booster({'nthread':8}, model_file = modelfile)
ypred = bst.predict( xgmat )
```

Cross Validation

We used 10% data from training set for the cross validation to estimate the accuracy while prevent overfitting using the AMS Metric.

```
# Random seed for reproducible results.
np.random.seed(42)
# Random number for training/validation splitting
r =np.random.rand(data_train.shape[0])

# Put Y(truth), X(data), W(weight), and I(index) into their own arrays
print 'Assigning data to numpy arrays.'
# First 90% are training
Y_train = data_train[:,32][r<0.9]
X_train = data_train[:,1:31][r<0.9]
W_train = data_train[:,31][r<0.9]

# Lirst 10% are validation
Y_valid = data_train[:,32][r>=0.9]
X_valid = data_train[:,1:31][r>=0.9]
W_valid = data_train[:,31][r>=0.9]
```

Evolution of the Model

AMS Metric

The evaluation metric is the approximate median significance (AMS):

$$\mathrm{AMS} = \sqrt{2\left((s+b+b_r)\log\left(1+\frac{s}{b+b_r}\right)-s\right)}$$

where

- s,b: unnormalized true positive and false positive rates, respectively,
- b_r=10 is the constant regularization term,
- log is the natural log.

More precisely, let (y1,…,yn)∈{b,s}n be the vector of true test labels, let (y^1,…,y^n)∈{b,s}n be the vector of predicted (submitted) test labels, and let (w1,…,wn)∈R+n be the vector of weights. Then

$$s = \sum_{i=1}^{n} w_i 1\{y_i = \mathrm{s}\} 1\{\hat{y}_i = \mathrm{s}\}$$

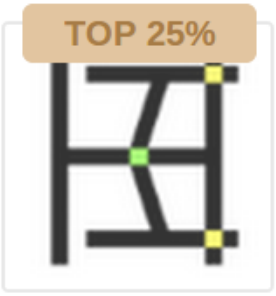
and

$$b = \sum_{i=1}^{n} w_i 1\{y_i = \mathrm{b}\} 1\{\hat{y}_i = \mathrm{s}\},$$

Results

We were able to score 3.64655 score with this method. And we are able to secured a position in top 25%.

434	↑657	kickaha	3.64655	9	Mon, 15 Sep 2014 18:39:15 (-3.8d)
435	**↑322**	**Atom**	**3.64655**	**19**	**Sat, 13 Sep 2014 10:26:18 (-29.5h)**
436	↑958	Warana@UOM	3.64655	28	Mon, 15 Sep 2014 10:58:26 (-3d)



435th/1785

Conclusion

The Higgs Boson Challenge is a relatively difficult machine learning task. Even the winning submissions score about 3.8, which is a relatively low accuracy. The novelty of the usage of algorithms and handling different features have less of an effect than having a slow and steady learning rate and running the same algorithm iteratively. A good result can be achieved, given enough computational power, by using established algorithms (especially ensemble techniques such as boosting) and running through the calculation with several iterations.

Limitations

We had limited processing power and limited memory amount. With that resources were not able to do much testing and analysis on data. It took about hour to complete the some processing iteration. Processing all the data at once been a problem with limited resources. So, we had to divide the process into parts.

The lack of domain knowledge required to understand and evaluate the features given reduced the ability of selecting correct preprocessing methods to clean and select data to train the model. While going through the forums it was clear that most contestants used their domain knowledge to create aggregated features using the given data which improved their model score.

Further Improvements

We used one model at a time to compute the predictions. But using several models and methods and using ensemble methods and using a constrained optimization to come within a good consensus result, we can improve the prediction accuracy. Also there are remaining given data which isn't considered for the prediction. Using those data we may be able to improve the results.

Since the correlation of individual features or feature sets with a small number of features was not significant, it would be prudent to assume that a feed-forward neural network with a depth of around four hidden neural layers would have provided with a significantly better result. However, the resource requirement (computational and temporal) will be immense.

Acknowledgement

We used python scikit-learn library for Gradient Boosting and Support Vector Machines Also we used various tools to initial analytics and pre-processing such as Weka 3, Rapid Miner 5.3, MySQL with phpmyadmin.

Also we used several python libraries such as XgBoost, pandas, numpy, pickle, sklearn, matplotlib and pybrain are some of them.

References

[1] G. Aad et al., “Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC”, Phys.Lett., vol. B716, pp. 1–29, 2012.

[2] S. Chatrchyan et al., “Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC”, Phys.Lett., vol. B716, pp. 30–61, 2012.

[3] The ATLAS Collaboration, “Evidence for higgs boson decays to tau+tau- final state with the atlas detector”, Tech. Rep. ATLAS-CONF-2013-108, November 2013, http://cds.cern.ch/record/1632191.

[4] V. M. Abazov et al., “Observation of single top-quark production”, Physical Review Letters, vol. 103, no. 9, 2009.

[5] Aaltonen, T. et. al, “Observation of electroweak single top-quark production”, Phys. Rev. Lett., vol. 103, pp. 092002, Aug 2009.

[6] G. Cowan, K. Cranmer, E. Gross, and O. Vitells, “Asymptotic formulae for likelihood-based tests of new physics”, The European Physical Journal C, vol. 71, pp. 1554–1573, 2011.

[7] G. Aad et al., “A Particle Consistent with the Higgs Boson Observed with the ATLAS Detector at the Large Hadron Collider”, Science, vol. 338, pp. 1576–1582, 2012.

[8] S. S. Wilks, “The large-sample distribution of the likelihood ratio for testing composite hypotheses”, Annals of Mathematical Statistics, vol. 9, pp. 60–62, 1938.

[9] L. Devroye, L. Gy ¨ orfi, and G. Lugosi, A Probabilistic Theory of Pattern Recognition, Springer, New York, 1996.

[10] Introduction to Information Retrieval, Cambridge UP, 2009.
[11] D. Sculley , Results from a Semi-Supervised Feature Learning Competition:, Google Pittsburgh.

[12] A. Herschtal and B. Raskutti. Optimising area under the ROC curve using gradient descent. In Proceedings of the International Conference on Machine Learning (ICML), Ban?, Canada, 2004.

[13] J. H. Friedman. Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29:1189–1232, 2001.

[14] George Athanasopoulos and Rob J Hyndman, “The value of feedback in forecasting competitions” 10 March 2011.

[15] Scikit Learn Documentation, Online Available, at http://scikit-learn.org/stable/documentation.html

[16] Kaggle.com, Online Available, at https://www.kaggle.com/competitions

[17] Benjamin M. Marlin, Missing Data Problems in Machine Learning, 2008, Online Available, at http://www.cs.toronto.edu/~marlin/research/phd_thesis/marlin-phd-thesis.pdf