6.867 Machine Learning (Fall 2018)
Prof.: Suvrit Sra
TA: Alex Turner

Youssef Berrada (ID: 9118441544)
Georgy Guryev (ID: 924667514)
Sheng Yao (ID: 916855968)

Project Milestone 4: Initial Results

In this milestone, we will get an initial version of your system running end-to-end and produce initial results. Basically, after cleaning the data and pre-processing it as describe below, we tried different algorithm such as a neural network, and some decision tree models (CART, Random forest, boosting) with arbitrary hyper-parameters in order to have the full workflow.

Data Description and Feature Engineering

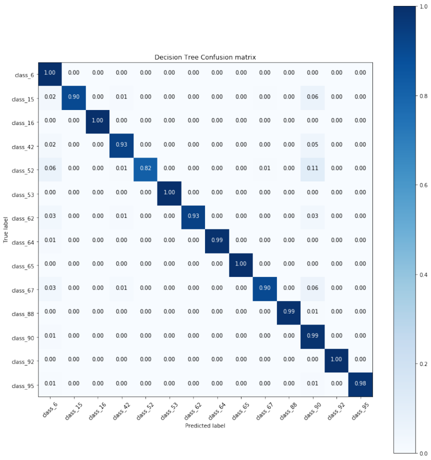
The datasets that we used to predict astronomical bodies are from two types. The first source includes meta-data about the objects such as its location in the sky, red shift, etc. The other source of data includes 6 different light pass-band measurements observed across 2 years (each object is observed during different periods within two years). Our main challenge with feature engineering is to incorporate the time-series data from different frequency band measurements to classification. As a first-pass, we simply take the average of each passband's time series, for each object, thereby reducing the time dimension of the data. Our next steps for future milestones are to apply different methods to preserve the time-series information and incorporate them into classification.

Initial Results

Tree-based Models

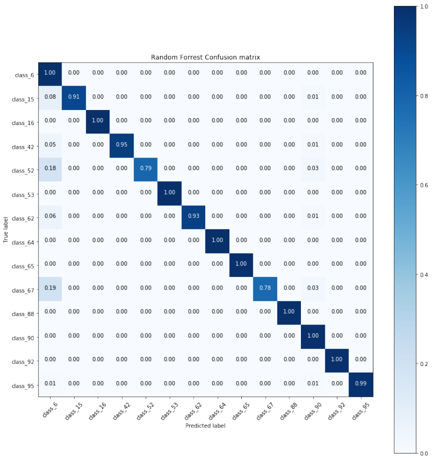
CART

The CART Model overfits the training set perfectly (100% precision) with max-depth of 20. It is undesirable but we include it as a first try of the method.



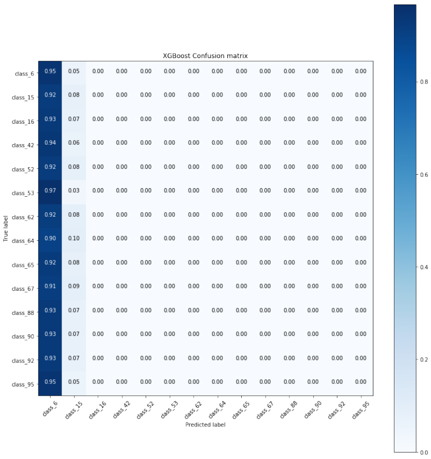
Random Forest

The Random Forest model is less prone to overfit than CART Model theoretically, but with same max-depth of 20. We can see that it also tends towards overfitting.



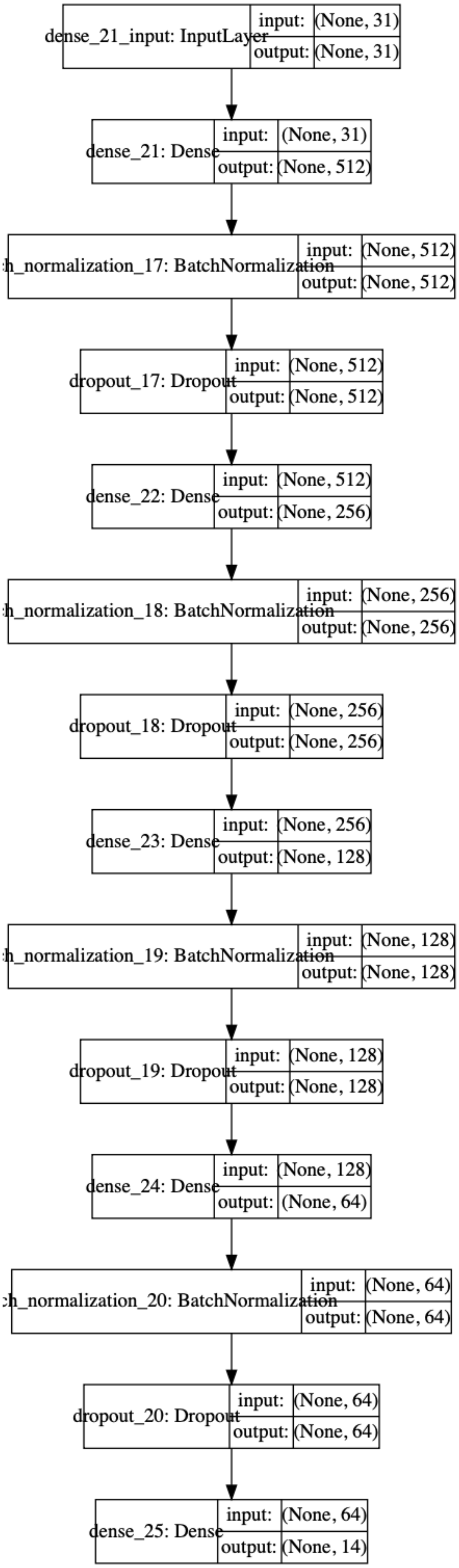
Boosting Tree

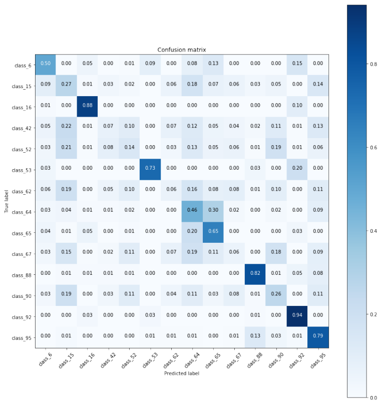
Based on the previous two results, we would expect the Boosting method to exhibit similar behavior. However, the method underfits the training set and produces only 1%precision



Neural Network

We implemented a feed-forward Neural Network with 4 hidden layers and a softmax output layer. To prevent overfitting, we add dropout and batch normalization at each layer. The activation function at each hidden layer is ReLu and we train the model with 5 epochs and batch size of 100.


dense_21_input: InputLayer
input: (None, 31)
output: (None, 31)
dense_21: Dense
input: (None, 31)
output: (None, 512)
h_normalization_17: BatchNormalization
input: (None, 512)
output: (None, 512)
dropout_17: Dropout
input: (None, 512)
output: (None, 512)
dense_22: Dense
input: (None, 512)
output: (None, 256)
h_normalization_18: BatchNormalization
input: (None, 256)
output: (None, 256)
dropout_18: Dropout
input: (None, 256)
output: (None, 256)
dense_23: Dense
input: (None, 256)
output: (None, 128)
h_normalization_19: BatchNormalization
input: (None, 128)
output: (None, 128)
dropout_19: Dropout
input: (None, 128)
output: (None, 128)
dense_24: Dense
input: (None, 128)
output: (None, 64)
h_normalization_20: BatchNormalization
input: (None, 64)
output: (None, 64)
dropout_20: Dropout
input: (None, 64)
output: (None, 64)
dense_25: Dense
input: (None, 64)
output: (None, 14)



Confusion matrix

True label	class_6	class_15	class_16	class_42	class_52	class_53	class_62	class_64	class_65	class_67	class_88	class_90	class_92	class_95
class_6	0.50	0.00	0.05	0.00	0.01	0.09	0.00	0.08	0.13	0.00	0.00	0.00	0.15	0.00
class_15	0.09	0.27	0.01	0.03	0.02	0.00	0.06	0.18	0.07	0.06	0.03	0.05	0.00	0.14
class_16	0.01	0.00	0.88	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00
class_42	0.05	0.22	0.01	0.07	0.10	0.00	0.07	0.12	0.05	0.04	0.02	0.11	0.01	0.13
class_52	0.03	0.21	0.01	0.08	0.14	0.00	0.03	0.13	0.05	0.06	0.01	0.19	0.01	0.06
class_53	0.03	0.00	0.00	0.00	0.00	0.73	0.00	0.00	0.00	0.00	0.03	0.00	0.20	0.00
class_62	0.06	0.19	0.00	0.05	0.10	0.00	0.06	0.16	0.08	0.08	0.01	0.10	0.00	0.11
class_64	0.03	0.04	0.01	0.01	0.02	0.00	0.00	0.46	0.30	0.02	0.00	0.02	0.00	0.09
class_65	0.04	0.01	0.05	0.00	0.01	0.00	0.00	0.20	0.65	0.00	0.00	0.00	0.03	0.00
class_67	0.03	0.15	0.00	0.02	0.11	0.00	0.07	0.19	0.11	0.06	0.00	0.18	0.00	0.09
class_88	0.00	0.01	0.01	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.82	0.01	0.05	0.08
class_90	0.03	0.19	0.00	0.03	0.11	0.00	0.04	0.11	0.03	0.08	0.01	0.26	0.00	0.11
class_92	0.00	0.00	0.03	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.01	0.00	0.94	0.00
class_95	0.00	0.01	0.00	0.00	0.00	0.01	0.01	0.01	0.00	0.01	0.13	0.03	0.01	0.79

Predicted label

Figure 1: The results of astronomical object classification with the feed-forward Neural Net

Conclusion and Next Steps

Running the above neural network led us to be ranked 230 in Kaggle, which is a strong improvement from the previous iteration with a loss going from 6.6 to 1.70 where the leader is around 0.8. This first approach forces us to have a complete work flow from the data pre-processing to the data visualization. This helped us get to know the data and produce some initial (bad) results... For the next step, we would like to:

- Refine the approach with aggregated data to get finer results across different methodology.
- Once we are confident we our models with this approach, we would like to switch to a time series analysis and use the RNN framework to see if the results will be better. The time-series approach offers a wide possibility of features engineering, as we can identify pattern using technical indicators (Bollinger bands...)

Division of Labor

All three team members contributed significantly to the milestone. We actually all brainstormed at the same time to understand each algorithm and discussed what methods should we explore and package we should use. We then all went over all algorithms and libraries we plan to use.