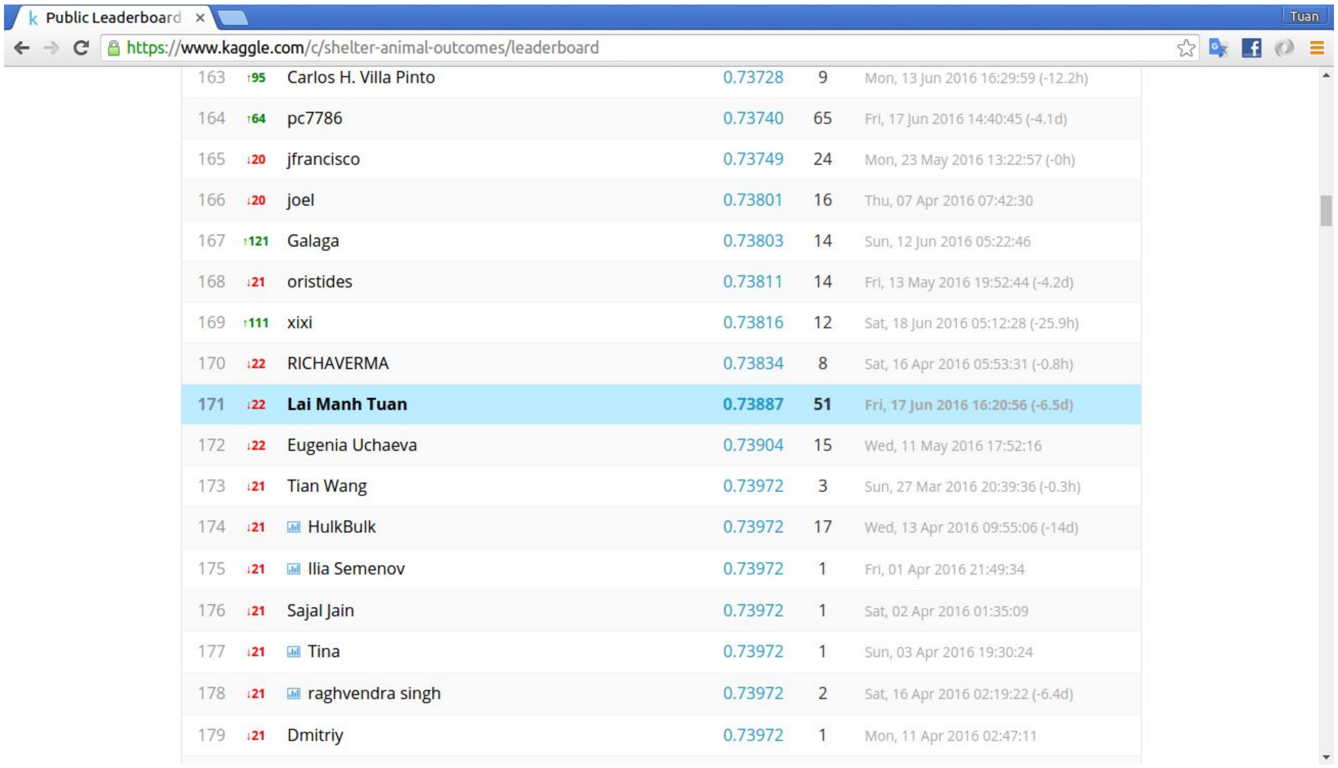
Tuan Lai Manh
20130729

Research Project: Shelter Animal Outcomes

1. Screenshot

By the time of writing this report, my rank was 171 and my best score was 0.73887.


Public Leaderboard
https://www.kaggle.com/c/shelter-animal-outcomes/leaderboard
Tuan

163	↑95	Carlos H. Villa Pinto	0.73728	9	Mon, 13 Jun 2016 16:29:59 (-12.2h)
164	↑64	pc7786	0.73740	65	Fri, 17 Jun 2016 14:40:45 (-4.1d)
165	↓20	jfrancisco	0.73749	24	Mon, 23 May 2016 13:22:57 (-0h)
166	↓20	joel	0.73801	16	Thu, 07 Apr 2016 07:42:30
167	↑121	Galaga	0.73803	14	Sun, 12 Jun 2016 05:22:46
168	↓21	oristides	0.73811	14	Fri, 13 May 2016 19:52:44 (-4.2d)
169	↑111	xixi	0.73816	12	Sat, 18 Jun 2016 05:12:28 (-25.9h)
170	↓22	RICHAVERMA	0.73834	8	Sat, 16 Apr 2016 05:53:31 (-0.8h)
171	↓22	**Lai Manh Tuan**	**0.73887**	**51**	**Fri, 17 Jun 2016 16:20:56 (-6.5d)**
172	↓22	Eugenia Uchaeva	0.73904	15	Wed, 11 May 2016 17:52:16
173	↓21	Tian Wang	0.73972	3	Sun, 27 Mar 2016 20:39:36 (-0.3h)
174	↓21	HulkBulk	0.73972	17	Wed, 13 Apr 2016 09:55:06 (-14d)
175	↓21	Ilia Semenov	0.73972	1	Fri, 01 Apr 2016 21:49:34
176	↓21	Sajal Jain	0.73972	1	Sat, 02 Apr 2016 01:35:09
177	↓21	Tina	0.73972	1	Sun, 03 Apr 2016 19:30:24
178	↓21	raghvendra singh	0.73972	2	Sat, 16 Apr 2016 02:19:22 (-6.4d)
179	↓21	Dmitriy	0.73972	1	Mon, 11 Apr 2016 02:47:11

2. Feature Selection and Data Cleaning

The code for preprocessing the raw dataset is stored in the file *data_preprocessing.R*. The code has many comments. Basically, there were three main issues I had to deal with.

First, in the original dataset, the Breed attribute has more than 1500 different values. Intuitively, information about breed should be important in this Kaggle problem. However, using a categorical attribute with too many levels could lead to overfitting. For example, suppose we want to build a model to predict the academic performance of a student will be good or bad. By using only the ID attribute (a categorical attribute with many levels), we can easily build a decision tree that can classify all training examples correctly; however, obviously, the decision tree will not be able to generalize well. My approach for tackling this issue was to try to create a small number of new binary features that can neatly summarize all the information about breed. For example, I defined a new feature called *is_mixed*. If an animal has a mixed breed (e.g., Pit Bull Mix, Pharaoh Hound Mix), the feature value will be true. Otherwise, if the animal doesn't have a mixed breed, the value will be false.

Second, the Color attribute also has many different levels. What I did was to convert all the original colors into a small number of different groups. Here are a few examples:

“Black Brindle” ⇒ ”Black” group
"Black Smoke" ⇒ ”Black” group
"Black Tabby" ⇒ ”Black” group
"Blue Merle" ⇒ ”Blue” group

Finally, deciding which features to use for building predictive models can be difficult. There are many systematic ways for doing feature selection such as forward search, backward search, or floating search. However, I decided to use a ‘trial-and-error’ approach in doing feature selection. Initially, I started with a set of all features that I thought to be important in predicting the outcome of shelter animals. Then, I built predictive models using the chosen features. Later, whenever I came up with a new feature, I tried to add that feature into the set and check if the feature could improve the accuracies of my models. If yes, then I incorporated the new feature into the set of existing features. In addition, whenever I have some doubt on one of the existing features, I tried to remove it from the set and also check if the accuracies would improve. If yes, then I removed that feature from the set.

3. Random Forest
I built a random forest model by using the randomForest package in R (please refer to the file *rf.R*). My random forest consists of 800 decision trees. Usually, the more trees, the better. However, my computer was slow so I decided to build a random forest that consists of just 800 trees.

4. Xgboost
I built an Xgboost model by using the xgboost package in R (please refer to the file *xgboost.R*). Note that an Xgboost model has many more parameters than a random forest model. Normally, in order to build a good Xgboost model, hyper-parameter tuning is needed. Again, I used a ‘trial-and-error’ approach in doing parameter tuning. However, there were some basic rules that I used in doing parameter tuning. For example:

- max_depth [default=6] This parameter is the maximum depth of a tree. Increasing this value will make model more complex / likely to be overfitting. Therefore, if there is the overfitting problem (high training accuracy but low test accuracy), we can try reducing this value.
- eta [default=0.3] Roughly speaking, this parameter is like the learning rate.
- subsample [default=1] This denotes the fraction of observations to be randomly sampled for each tree. Lower values make the algorithm more conservative and can prevent overfitting but too small values might lead to under-fitting.
- ...

Tuan Lai Manh
20130729

5. Combining Random Forest model and Xgboost model

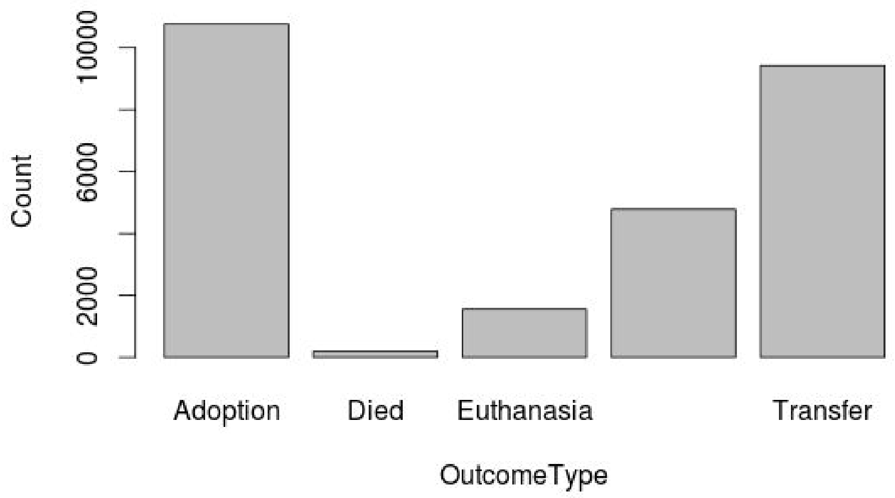
I tried to build new models on top of the Random Forest model and the Xgboost model that I built.
I tried two different approaches for combining the predictions of the two models.

- ❏ In the first approach, I used simple voting for combining the predictions of the two models. For example, given the info of some particular animal, if my Random Forest model predicts that the probability of the animal being adopted is 80%, while my Xgboost model predicts the probability to be 90%, then my new model will output 85% for the final probability. Please refer to the code in the file *simple_voting.R*.
- ❏ My second approach is more complex than the first approach. I built a logistic regression classifier on top of the Random Forest model and the Xgboost model. Please refer to the code in the file *logreg_combiner_stacking.py*.

6. Results and Discussion

My Xgboost model was the best among all the models that I built. Its best score was 0.73887.
The best score of my Random Forest model was 0.83321.
The two models built on top of the Xgboost model and the Random Forest model did not perform very well. They were even worse than each of the individual models.

In the future, I will try to build a neural network and a SVM model. In addition, I figured out that the training dataset was imbalanced as illustrated by the figure below. I will also try to mitigate this issue.


Count
Adoption
Died
Euthanasia
Transfer
OutcomeType