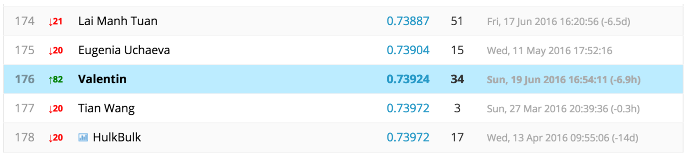
Data Mining and Knowledge Discovery (KSE525) - Kaggle Competition

Valentin Liévin - s152076 19/06/2016

1 Results

This score has been achieved using Python and two machine learning libraries: sklearn, pandas and xgb. In a few words, first extracted features from the raw data and dropped the non relevant features. I also added some extra features using knowledge discovery (popularity of the breeds). Then I selected the features using the GINI scores using random forest classifiers. Finally, I predicted the outcomes of the animals using a simple combination of a tuned xgb classifier and a tuned random forest classifier.



174	↓21	Lai Manh Tuan	0.73887	51	Fri, 17 Jun 2016 16:20:56 (-6.5d)
175	↓20	Eugenia Uchaeva	0.73904	15	Wed, 11 May 2016 17:52:16
176	**↑82**	**Valentin**	**0.73924**	**34**	**Sun, 19 Jun 2016 16:54:11 (-6.9h)**
177	↓20	Tian Wang	0.73972	3	Sun, 27 Mar 2016 20:39:36 (-0.3h)
178	↓20	HulkBulk	0.73972	17	Wed, 13 Apr 2016 09:55:06 (-14d)

Figure 1: My ranking on Kaggle

2 Preprocessing

The preprocessing step seems to be the most important step for the classification. Differences in preprocessing lead to critical changes in the classification.

The raw data contains several fields: 'AnimalID' 'Name' 'DateTime' 'OutcomeType' 'OutcomeSubtype' 'AnimalType' 'SexuponOutcome' 'AgeuponOutcome' 'Breed' 'Color'.

The data cannot be used with these raw features because some values are missing, some attributes are categorical (color and breed) and because some attributes contain more than one features. Indeed, for example, the sex of the animal says not only if the animal is male or female but also if the animal is spayed or neutered. Moreover, some information like the breed of the animal could be linked to higher level information and one attribute can produce multiple features (color, breed, sex and datetime). The data has been normalized for the xgb classifier. I have not explored more advanced techniques for data cleaning to handle the noise.

2.1 Features Extraction

AnimalID Each ID is unique, thus this attribute cannot be used for classification.

Name This attribute presents several values: the name of the animal or NaN. The name of the animal is not useful. However, the fact that the name of the animal is known or not is a useful information. Because if the animal has something on him that indicates his name, he is more likely to return to his owners.

DateTime The attributes DateTime contains a lot of useful information. As shown fig.**??**, the outcome of the animal depends a lot on the hour: owners are more likely to come to the shelter at the end of the day for example.

Therefore, from the field DateTime, I extract the Month, the Day, the Hour and even the Minute. (irrelevant features will be removed during the feature selection)

AnimalType Simply converting this categorical attribute to a binary attribute.

AgeuponOutcome The age of the animal is very important because a puppy is more likely to be adopted that an old dog. Moreover, the outcome of the animal is very different regarding to the period of the life of the animal.

Moreover, the age of the animal is given with a unit (day, week or month) which reflects quite well the different states of the life of the animal. For example, if the age is given in days, the animal is baby. If the age is given in years, it is not anymore a very young animal.

Therefore I extracted the age in days for all the animals but also the age regarding each unit of time. For example a two month old animal will be described by ('AgeInDays':0, 'AgeInWeeks':0, 'AgeInMonths':2,'AgeInYears':0). Thanks to this trick, I created an efficient way of creating categories of age using only the units of times.

SexuponOutcome The sex of the animal is quite important. However the most useful information is if the animal is neutered or spayed. Indeed, for people who want to adopt an animal, it's often more convenient to have such animal.

Breed An animal can be from several breeds and be mixed or not. I created one feature for each unique breed, for example an animal is a Shetland Sheepdog or not. A feature 'isMix' has also been added. Lastly, I also added a feature for the number of breeds because pure breed animals are often more valuable.

Color As for the breeds, I converted all the unique colors into new features. I also added a feature for the number of colors.

2.2 Filling Missing values

Some values are missing in the raw data. These values need to be filled for the classification. Simple techniques have been used but more advanced techniques like the use of a Bayes classifier would have been more efficient.

Name The name attribute is often missing. However, the feature extracted from this attribute already handle this.

Age When the age is missing, the age of the animal is set to the average of the population.

2.3 Feature Engineering - Knowledge discovery

The breed attribute contain a lot of information because the breed is a very discriminative attribute. For example, almost every Labradors will be adopted because the breed is very popular. However, dogs from unpopular breeds are less likely to be adopted.

I used the rankings of the breeds for dogs and cat according to the popularity (found on the Internet) in order to give a popularity score for each breed. Thus, it is easier for the classifier to generalise the model given that two popular breeds will have similar values.

The popularity score has enabled me to make my classification 1% more accurate.

I've also tried to build group of breeds using knowledge retrieved from the web (categories of breeds) but it was not successful. Then, I tried to build clusters of breeds based on the outcome of the animals but it was not successful either.

2.3.1 Feature selection

The first steps of the preprocessing process gives 315 features which makes the classification task quite difficult because it requires more computing power and because the model will be more sensible to the noise. Therefore, we select the more relevant features.

Correlation I used the correlation coefficient in order to determine if the features 'isMix' and the count of breeds are correlated (if they are correlated, we can remove one of them). However, with a correlation coefficient of 0.7, I deducted that the two variables are not correlated.

GINI index In order to select the most important features, I've first trained a random forest classifier on all the features. Then, I sorted the features by importance (GINI index) and selected the number of features that optimise the accuracy on a classification with a random forest classifier. Only 55 features are selected on 315 initial features.

3 Classification

Several classifiers have been tested including kNN (however all variables are not continuous), random forest classifier, AdaBoost, SVM (not efficient for a large problem), multi-layer perceptron with TensorFlow, linear regression and xgb. However, I have kept only the random forest classifiers and xgb combined together because they scored the best.

3.1 Tuning

The two classifiers have been tuned on the training set using cross validation and choosing the best parameters regarding to the average accuracy of the models.

3.2 Random Forest Classifier

Surprisingly, one of the simplest classifier is also one of the best. The Random Forest classifier provides an accuracy of 67% to 68% on the training set and uses few computing power. This is a really good candidate for this classification task.

3.3 Softmax classifier: Extreme Gradient Boosting

xgb is an optimized algorithm for Gradient Boosting that implements softmax classification. The optimization is quite impressive and it is really fast to train the model compared to other implementations.

As we face a multiclass classification problem, I use a softmax model for probabilities output (objective = multi:softprob).

The algorithm tries to fit a model using a gradient descent algorithm to find the minimum of a cost function based on the softmax function. To sum up, xgb optimizes the parameters of the model to predict the best probabilities thanks to the training set.

3.4 Multiple classifiers

Models have advantages and disadvantages. One model can models well specific distribution but can achieve poor performances on other specific tasks. My results have shown that random forest and xgb perform quite well. However, it was impossible to improve my score with only one model. Therefore I combine the two models for the classification task.

3.4.1 Averaging

In order to take profit of the two classifiers, I use the average of the probabilities given by the two classifiers. It has enabled me to perfom my best score.

3.4.2 Voting

I've tried to use voting classifiers from the sklearn library with the random forest classifiers and xgb. The results are quite good (and more stable to the random states) but my best results are still obtained with the simple averaging classifier. For other problems,

3.5 Code

I developped my model using IPython notebooks. All the source code is available in IPython notebooks and PDF.