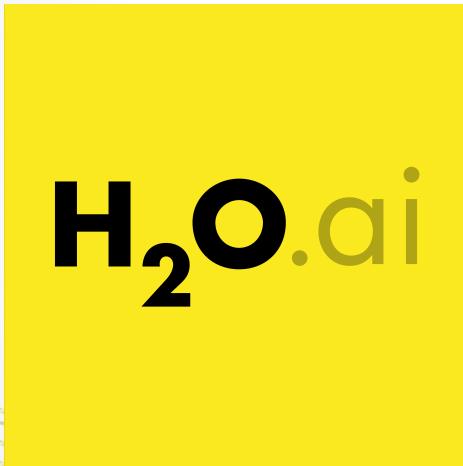


# Improving Model Predictions via Stacking and Hyper-Parameters Tuning



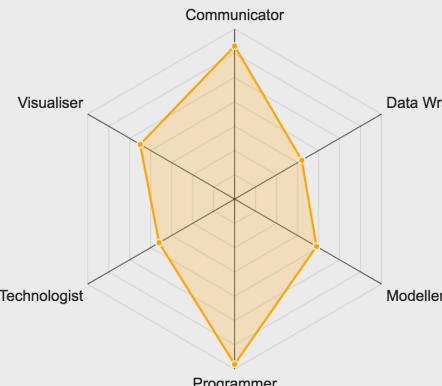
Jo-fai (Joe) Chow  
Data Scientist  
[joe@h2o.ai](mailto:joe@h2o.ai)  
@matlabulus

# About Me

- 2005 - 2015
- Water Engineer
  - Consultant for Utilities
  - EngD Research
- 2015 - Present
- Data Scientist
  - Virgin Media
  - Domino Data Lab
  - H2O.ai

# Mango Data Science Radar

## You are a Programmer



When you are at the forefront of innovation, often the tools needed to solve a problem simply do not exist.

You may already be a master of multiple technical languages and enjoy adding languages to your skillset.

You combine carefully constructed analysis workflow, with robust pipelines to automate as much of the process as possible, but enjoy building applications from scratch.

You understand the value of planning, and know that thinking through an analysis is more efficient and less error prone than starting an analysis and seeing what sticks.

You always ensure that your customers are not misled by your work by providing thorough reporting, including lists of assumptions and descriptions of algorithms along with your findings.

You use software development approaches such as unit testing and version control to ensure that costly mistakes are not caused by your work.

Your cool head and rational approach serve as a great counterpoint to team members whose focus on the big picture can lead to key details being missed.

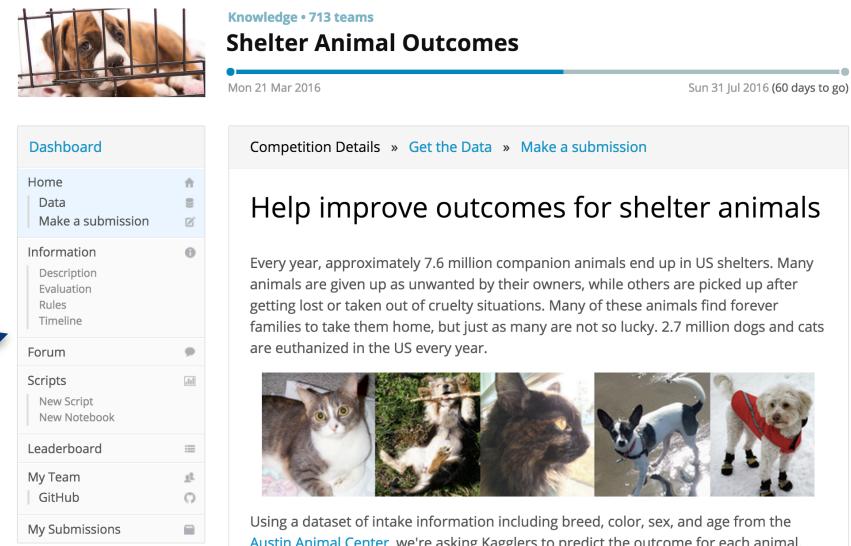
# About This Talk

- Predictive modelling
  - Kaggle as an example
- Improve predictions with simple tricks
- Use data science for social good 



# About Kaggle

- World's biggest predictive modelling competition platform
- 560k members
- Competition types:
  - Featured (prize)
  - Recruitment
  - Playground
  - 101



The screenshot shows a competition page for 'Shelter Animal Outcomes'. At the top right, it says 'Knowledge • 713 teams' and 'Mon 21 Mar 2016 Sun 31 Jul 2016 (60 days to go)'. Below that is a large image of a brown and white dog looking through a cage. To the left is a sidebar with links: Home, Data, Make a submission; Information (Description, Evaluation, Rules, Timeline); Forum; Scripts (New Script, New Notebook); Leaderboard; My Team (GitHub); and My Submissions. The main content area has a header 'Shelter Animal Outcomes' and a sub-header 'Help improve outcomes for shelter animals'. It explains that every year, approximately 7.6 million companion animals end up in US shelters, with many being given up or picked up after getting lost or being abandoned. It notes that while many find forever homes, others are euthanized. Below this is a row of five small images of different shelter animals: a grey and white cat, a calico cat, a black cat, a small black and white dog, and a small white dog wearing a red vest.

Knowledge • 713 teams

Shelter Animal Outcomes

Mon 21 Mar 2016 Sun 31 Jul 2016 (60 days to go)

Dashboard

Home Data Make a submission

Information

Description Evaluation Rules Timeline

Forum

Scripts

New Script New Notebook

Leaderboard

My Team GitHub

My Submissions

Competition Details » Get the Data » Make a submission

Help improve outcomes for shelter animals

Every year, approximately 7.6 million companion animals end up in US shelters. Many animals are given up as unwanted by their owners, while others are picked up after getting lost or taken out of cruel situations. Many of these animals find forever families to take them home, but just as many are not so lucky. 2.7 million dogs and cats are euthanized in the US every year.

Using a dataset of intake information including breed, color, sex, and age from the [Austin Animal Center](#), we're asking Kagglers to predict the outcome for each animal.

# Predicting Shelter Animal Outcomes

- X: Predictors
  - Name
  - Gender
  - Type ( or )
  - Date & Time
  - Age
  - Breed
  - Colour
- Y: Outcomes (5 types)
  - Adoption
  - Died
  - Euthanasia
  - Return to Owner
  - Transfer
- Data
  - Training (27k samples)
  - Test (11k)

# Basic Feature Engineering

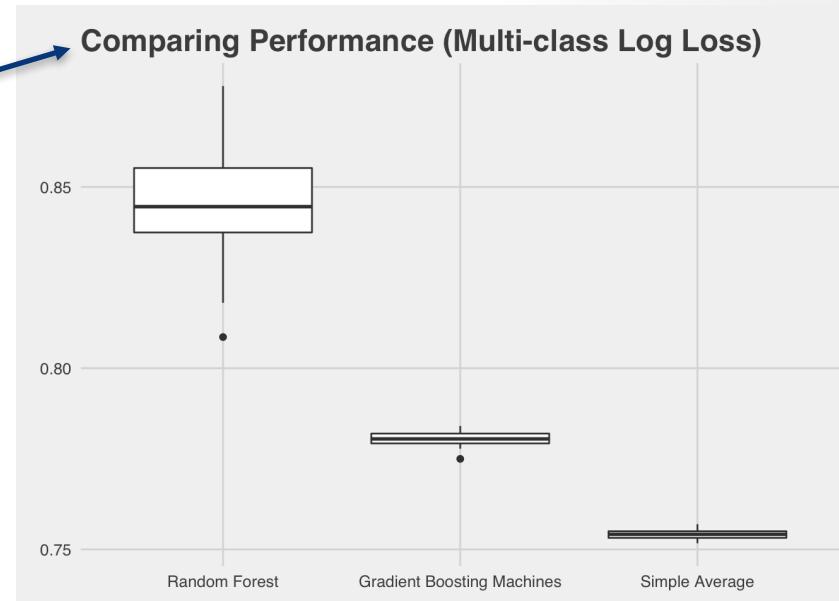
X	Raw (Before)	Reformatted (After)
Name	Elsa, Steve, Lassie	[name_len]: 4, 5, 6
Date & Time	2014-02-12 18:22:00	[year]: 2014 [month]: 2 [weekday]: 4 [hour]: 18
Age	1 year, 3 weeks, 2 days	[age_day]: 365, 21, 2
Breed	German Shepherd, Pit Bull Mix	[is_mix]: 0, 1
Colour	Brown Brindle/White	[simple_colour]: brown

# Common Machine Learning Techniques

- Ensembles
  - Bagging/boosting of decision trees
  - Reduces variance and increase accuracy
  - Popular R Packages (used in next example)
    - “randomForest”
    - “xgboost”
- There are a lot more machine learning packages in R:
  - “caret”, “caretEnsemble”
  - “h2o”, “h2oEnsemble”
  - “mlr”

# Simple Trick – Model Averaging

- Stratified sampling
  - 80% for training
  - 20% for validation
- Evaluation metric
  - Multi-class Log Loss
  - Lower the better
  - 0 = Perfect
- 50 runs
  - different random seed



# More Advanced Methods

- Model Stacking
  - Uses a second-level metalearner to learn the optimal combination of base learners
  - R Packages:
    - “SuperLearner”
    - “subsemble”
    - “h2oEnsemble”
    - “caretEnsemble”
- Hyper-parameters Tuning
  - Improves the performance of individual machine learning algorithms
  - Grid search
    - Full / Random
  - R Packages:
    - “caret”
    - “h2o”

For more info, see

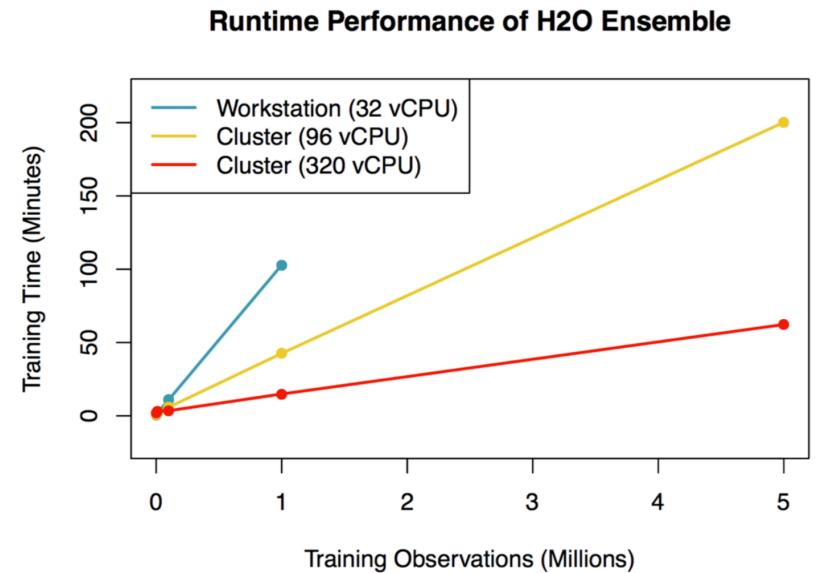
[https://github.com/h2oai/h2o-meetups/tree/master/2016\\_05\\_20\\_MLconf\\_Seattle\\_Scalable\\_Ensembles](https://github.com/h2oai/h2o-meetups/tree/master/2016_05_20_MLconf_Seattle_Scalable_Ensembles)  
<https://github.com/h2oai/h2o-3/blob/master/h2o-docs/src/product/tutorials/gbm/gbmTuning.Rmd>

# Trade-Off of Advanced Methods

- Strength
  - Model tuning + stacking won nearly all Kaggle competitions.
  - Multi-algorithm ensemble may better approximate the true predictive function than any single algorithm.
- Weakness
  - Increased training and prediction times.
  - Increased model complexity.
  - Requires large machines or clusters for big data.

# R + H2O = Scalable Machine Learning

- H2O is an open-source, distributed machine learning library written in Java with APIs in R, Python and more.
- “h2oEnsemble” is the scalable implementation of the Super Learner algorithm for H2O.



# H2O Random Grid Search Example

Define search range and criteria

```
# Define a list of hyper-parameters
h_param_gbm <- list(
  max_depth      = seq(5, 10, 1),
  sample_rate    = seq(0.5, 0.8, 0.1),
  col_sample_rate = seq(0.5, 0.8, 0.1)
)

# Search Criteria
search_criteria = list(
  strategy        = "RandomDiscrete", # Use random grid search
  max_runtime_secs = 600               # limit the runtime to 10 minutes
)

# Grid Search
grid <- h2o.grid(
  hyper_params    = h_param_gbm,
  search_criteria = search_criteria,
  algorithm       = "gbm",           # which H2O algorithm to run

  # Standard model parameters
  training_frame   = hex_train,
  validation_frame = hex_valid,
  x                = features,
  y                = "OutcomeType",
```

H2O Grid Details

=====

Grid ID: gbm\_grid

Used hyper parameters:

- sample\_rate
- max\_depth
- col\_sample\_rate

Number of models: 21

Number of failed models: 0

Best performance models

Hyper-Parameter Search Summary: ordered by increasing logloss

	sample_rate	max_depth	col_sample_rate	model_ids	logloss
1	0.8	7	0.6	gbm_grid_model_2	0.752684243384921
2	0.8	6	0.5	gbm_grid_model_10	0.75318954895972
3	0.8	8	0.6	gbm_grid_model_4	0.753499220269728
4	0.7	6	0.6	gbm_grid_model_8	0.755403405990038
5	0.7	5	0.5	gbm_grid_model_14	0.756978014232093

	sample_rate	max_depth	col_sample_rate	model_ids	logloss
16	0.8	10	0.8	gbm_grid_model_12	0.765548067940223
17	0.6	9	0.7	gbm_grid_model_5	0.765760036358475
18	0.6	10	0.7	gbm_grid_model_16	0.769422559292015
19	0.5	10	0.8	gbm_grid_model_0	0.770202667369843
20	0.7	10	0.8	gbm_grid_model_17	0.773242423940347
21	0.6	10	0.5	gbm_grid_model_18	0.776840032616653

# H2O Model Stacking Example

Using `h2o.stack(...)` to combine multiple models

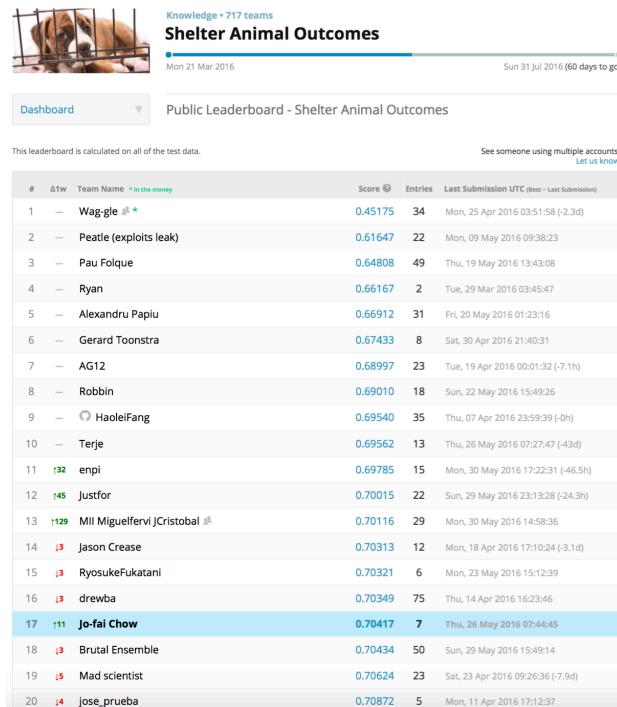
```
# H2O GBM
model_gbm <- h2o.gbm(training_frame = hex_train,
                      x = features, y = target,
                      nfolds = 5,
                      fold_assignment = "Modulo",
                      keep_cross_validation_predictions = TRUE)

# H2O Random Forest
model_drf <- h2o.randomForest(training_frame = hex_train,
                               x = features, y = target,
                               nfolds = 5,
                               fold_assignment = "Modulo",
                               keep_cross_validation_predictions = TRUE)

# Define list of models and metalearner
models <- list(model_gbm, model_drf)
metalearner <- "h2o.glm.wrapper"

# Run h2o.stack for metalearning
stack <- h2o.stack(models = models,
                   response_frame = hex_train[, target],
                   metalearner = metalearner,
                   keep_levelone_data = TRUE)
```

Getting reasonable results



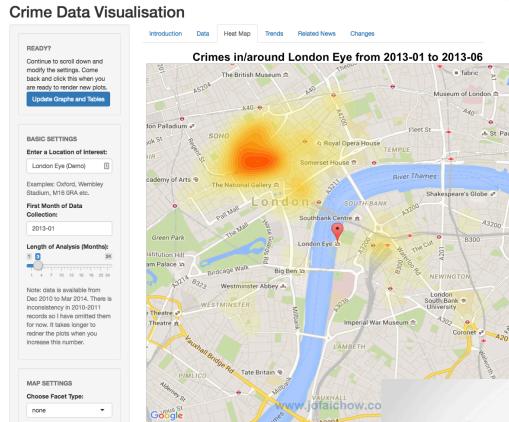
17 out of 717 teams (≈ top 2%)

# Conclusions

- Many R packages for predictive modelling.
- Use hyper-parameters tuning to improve individual models.
- Use model averaging / stacking to improve predictions.
- Trade-off between model performance and computational costs.
- Use R + H2O for scalable machine learning.
- H2O random grid search and stacking.
- Use data science for social good 

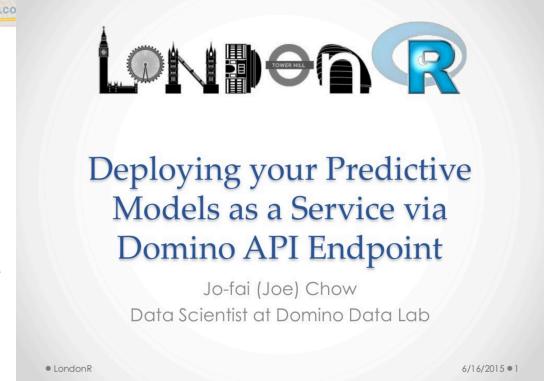
# Big Thank You!

- Mango Solutions
- RStudio
- Domino Data Lab
- H2O
  - Erin LeDell
  - Raymond Peck
  - Arno Candel



2<sup>nd</sup> LondonR Talk  
Domino API Endpoint

1<sup>st</sup> LondonR Talk  
Crime Map Shiny App



# Any Questions?

- Contact
  - joe@h2o.ai
  - @matlabulous
  - [github.com/woobe](https://github.com/woobe)
- Slides & Code
  - [github.com/h2oai/h2o-meetups](https://github.com/h2oai/h2o-meetups)
- H2O in London
  - Meetups / Office (soon)
  - [www.h2o.ai/careers](https://www.h2o.ai/careers)
- More H2O at Strata tomorrow
  - Innards of H2O (11:15)
  - Intro to Generalised Low-Rank Models (14:05)

# Extra Slide (Stratified Sampling)

## Raw Target (y)

```
smry_target_raw <- data.frame(table(y_train$OutcomeType))
smry_target_raw$Percentage <- smry_target_raw$Freq / length(y_train$OutcomeType)
smry_target_raw$Percentage <- round(smry_target_raw$Percentage, 4)
print(smry_target_raw)
```

```
##           Var1  Freq Percentage
## 1      Adoption 10769    0.4029
## 2          Died   197    0.0074
## 3  Euthanasia  1555    0.0582
## 4 Return_to_owner  4785    0.1790
## 5      Transfer  9422    0.3525
```

## Target by Fold

```
for (n_fold in 1:5) {
  cat("Summary of fold", n_fold, "\n")
  smry_target_fold <- data.frame(table(y_train[which(y_train$fold == n_fold), ]$OutcomeType))
  smry_target_fold$Percentage <- smry_target_fold$Freq / length(y_train[which(y_train$fold == n_fold), ]$OutcomeType)
  smry_target_fold$Percentage <- round(smry_target_fold$Percentage, 4)
  print(smry_target_fold)
  cat("\n")
}
```

```
## Summary of fold 1
##           Var1  Freq Percentage
## 1      Adoption 2154    0.4028
## 2          Died   40    0.0075
## 3  Euthanasia  311    0.0582
## 4 Return_to_owner  957    0.1790
## 5      Transfer 1885    0.3525
```