Interpretable Machine Learning

Data Sets

Attrition

```
attrition <- attrition %>% mutate_if(is.ordered, factor, order = F)
attrition_h2o <- as.h2o(attrition)

churn <- initial_split(attrition, prop = .7, strata = "Attrition")

churn_train <- training(churn)
churn_test <- testing(churn)

rm(churn)
```

Ames, Iowa housing data.

```
ames <- AmesHousing::make_ames()
ames_h2o <- as.h2o(ames)

set.seed(123)

ames_split <- initial_split(ames, prop =.7, strata = "Sale_Price")

ames_train <- training(ames_split)
ames_test <- testing(ames_split)

rm(ames_split)

h2o.init(max_mem_size = "10g", strict_version_check = F)
```

```
 Connection successful!

R is connected to the H2O cluster:
    H2O cluster uptime:         4 seconds 605 milliseconds
    H2O cluster timezone:       America/New_York
    H2O data parsing timezone:  UTC
    H2O cluster version:        3.28.0.4
    H2O cluster version age:    8 days
    H2O cluster name:           H2O_started_from_R_bmore_fjn064
    H2O cluster total nodes:    1
    H2O cluster total memory:   15.98 GB
    H2O cluster total cores:    16
    H2O cluster allowed cores:  16
    H2O cluster healthy:        TRUE
```

```
    H2O Connection ip:          localhost
    H2O Connection port:        54321
    H2O Connection proxy:       NA
    H2O Internal Security:      FALSE
    H2O API Extensions:         Amazon S3, Algos, AutoML, Core V3, TargetEncoder, Core V4
    R Version:                  R version 3.6.2 (2019-12-12)
```

```
train_h2o <- as.h2o(ames_train)
response <- "Sale_Price"
predictors <- setdiff(colnames(ames_train), response)
```

```
# ensure consistent categorical levels
blueprint <- recipe(Sale_Price ~., data = ames_train) %>%
  step_other(all_nominal(), threshold = 0.005)

# Create training / test h2o frames
train_h2o <- prep(blueprint, training = ames_train, retain = T) %>%
  juice() %>%
  as.h2o()

test_h2o <- prep(blueprint, training = ames_train) %>%
  bake(new_data = ames_test) %>%
  as.h2o()

Y <- "Sale_Price"
X <- setdiff(names(ames_train), Y)
```

h2o ML setup

```
# Train & cross-validate a GLM model
best_glm <- h2o.glm(
  x = X, y = Y, training_frame = train_h2o, alpha = 0.1,
  remove_collinear_columns = TRUE, nfolds = 10, fold_assignment = "Modulo",
  keep_cross_validation_predictions = TRUE, seed = 123
)

# Train & cross-validate a RF model
best_rf <- h2o.randomForest(
  x = X, y = Y, training_frame = train_h2o, ntrees = 1000, mtries = 20,
  max_depth = 30, min_rows = 1, sample_rate = 0.8, nfolds = 10,
  fold_assignment = "Modulo", keep_cross_validation_predictions = TRUE,
  seed = 123, stopping_rounds = 5, stopping_metric = "RMSE",
  stopping_tolerance = 0
)
```

```
# Train & cross-validate a GBM model
best_gbm <- h2o.gbm(
  x = X, y = Y, training_frame = train_h2o, ntrees = 5000, learn_rate = 0.01,
  max_depth = 7, min_rows = 5, sample_rate = 0.8, nfolds = 10,
  fold_assignment = "Modulo", keep_cross_validation_predictions = TRUE,
  seed = 123, stopping_rounds = 5, stopping_metric = "RMSE",
  stopping_tolerance = 0
)

# Train & cross-validate an XGBoost model

# not avaliable on windows, yet..(3/2/20)

#best_xgb <- h2o.xgboost(
#  x = X, y = Y, training_frame = train_h2o, ntrees = 5000, learn_rate = 0.05,
#  max_depth = 3, min_rows = 3, sample_rate = 0.8, categorical_encoding = "Enum",
#  nfolds = 10, fold_assignment = "Modulo",
#  keep_cross_validation_predictions = TRUE, seed = 123, stopping_rounds = 50,
#  stopping_metric = "RMSE", stopping_tolerance = 0
#)

# Train a stacked tree ensemble
ensemble_tree <- h2o.stackedEnsemble(
  x = X, y = Y, training_frame = train_h2o, model_id = "my_tree_ensemble_01",
  base_models = list(best_glm, best_rf, best_gbm),
  metalearner_algorithm = "drf"
)
```

Clean up

```
h2o.shutdown(prompt = FALSE)
```

```
[1] TRUE
```

```
# clean up
rm(list = ls())
```