stats_in_julia_solutions

April 4, 2017

1 Question 1

Using the data in chronic_kidney_disease.csv, determine whether or not the oldest patient in the sample has chronic kidney disease. Note that the `class` variable indicates a patient's CKD status.

As a hint, you will probably want to use the `maximum()` function and the `find()` function.

And you'll need to consider how to handle `NA` values; there is a function `dropna()` that will be useful.

```
In [37]: using DataFrames

         ckd = readtable("../data/chronic_kidney_disease.csv");

In [10]: maxage = maximum(dropna(ckd[:age]))

         oldest_idx = find(ckd[:age] .== maxage)

         ckd[oldest_idx, :class]

Out[10]: 1-element DataArrays.DataArray{String,1}:
          "ckd"
```

2 Question 2

Using the `chronic_kidney_disease.csv` dataset from above, use an appropriate statistical test to determine if patients with chronic kidney disease (CKD) have significantly higher blood urea than patients without CKD.

As a hint, you will likely want to use the HypothesisTests package.

```
In [10]: using DataFrames
         using HypothesisTests

         ckd = readtable("../data/chronic_kidney_disease.csv")

         # get row indices
         ckd_idcs = find(ckd[:class] .== "ckd")
         nonckd_idcs = find(ckd[:class] .== "notckd")
```

```
# get `blood_urea` scores
ckd_grp = ckd[ckd_idcs, :blood_urea]
nonckd_grp = ckd[nonckd_idcs, :blood_urea]

UnequalVarianceTTest(dropna(ckd_grp), dropna(nonckd_grp))
```

```
Out[10]: Two sample t-test (unequal variance)
         ------------------------------------
         Population details:
             parameter of interest:   Mean difference
             value under h_0:         0
             point estimate:          39.590418424753864
             95% confidence interval: (31.86529602463314,47.31554082487459)

         Test summary:
             outcome with 95% confidence: reject h_0
             two-sided p-value:           1.7990230669795845e-20 (extremely signifi

         Details:
             number of observations:   [237,144]
             t-statistic:              10.090709649973618
             degrees of freedom:       264.8784441340254
             empirical standard error: 3.9234523435977935
```

---

3 Question 3

Using the chronic_kidney_disease.csv data, determine which of the following predictors (if any) are related chronic kidney disease (CKD):
 blood_urea,
 hemoglobin,
 red_blood_cell_count,
 white_blood_cell_count.
 Hints: - There are a few days this could be done, let's use a single regression model of some kind - Our outcome variable is `class` in the CKD data, this needs to be re-coded as 0/1

```
In [6]: using GLM
        using DataFrames

        ckd = readtable("../data/chronic_kidney_disease.csv", makefactors = true)

        ckd[:has_ckd] = [x == "ckd" ? 1 : 0 for x in ckd[:class]]

        mod3 = @formula(has_ckd ~ 1 + blood_urea + hemoglobin + red_blood_cell_cour

        fm3 = glm(mod3, ckd, Binomial())
```

```
Out[6]: DataFrames.DataFrameRegressionModel{GLM.GeneralizedLinearModel{GLM.GlmResp{

        Formula: has_ckd ~ 1 + blood_urea + hemoglobin + red_blood_cell_count + whi

        Coefficients:
                                    Estimate   Std.Error  z value Pr(>|z|)
        (Intercept)                  20.5118     3.82243  5.36617    <1e-7
        blood_urea                 0.0135445   0.0152221 0.889787   0.3736
        hemoglobin                   -1.4337    0.273702 -5.23817    <1e-6
        red_blood_cell_count       -0.685826    0.399729 -1.71573   0.0862
        white_blood_cell_count   0.000152042 0.000141559  1.07406   0.2828
```

4 Question 4

Using the `stagec` data from above, fit several models experimenting with different numbers of trees and different numbers of variable subsets for candidate splitting (i.e., `m_try`, the third argument to the `build_forest()` function).

In order to evaluate the quality of the models on training data, write a function that calculates the mean-squared error of the fitted model. The function should take 3 arguments: (1) the fitted model, (2) the vector with the outcome variable and (3) the matrix of predictors.

What was the mean-squared error of your best-fitting model?

```
In [11]: # This is a quick function to obtain the mean-squared
         # error of a fitted random forest (or bagged tree) model.

         function mse(fitted, y, X)
             yhat = apply_forest(fitted, X)
             sqerr = (y .- yhat).^2
             out = mean(sqerr)
             return out
         end

Out[11]: mse (generic function with 1 method)
```

```
In [14]: # Load the data
         using DecisionTree
         using RDatasets
         stagec = dataset("rpart", "stagec")

         stagec_comp = stagec[completecases(stagec), :];

WARNING: using DecisionTree.fit! in module Main conflicts with an existing identifi
WARNING: using DecisionTree.R2 in module Main conflicts with an existing identifier
WARNING: using DecisionTree.predict in module Main conflicts with an existing ident
```

```
In [15]: # Clean up data and fit model
```

```
          is_tetraploid = stagec_comp[:Ploidy] .== "tetraploid"

          stagec_comp[:tetra] = is_tetraploid

          # must convert to Array
          y = convert(Array{Float64,1}, stagec_comp[:G2])
          X = convert(Array{Float64,2}, stagec_comp[[:Age, :Grade, :Gleason, :EET, :

          fm4a = build_forest(y, X, 5, 100)
          fm4b = build_forest(y, X, 3, 100)
          fm4c = build_forest(y, X, 5, 500)

Out[15]: Ensemble of Decision Trees
         Trees:      500
         Avg Leaves: 31.272
         Avg Depth:  9.914

In [16]: @show mse(fm4a, y, X)
         @show mse(fm4b, y, X)
         @show mse(fm4c, y, X)

mse(fm4a,y,X) = 18.4084866706618
mse(fm4b,y,X) = 22.123257454931814
mse(fm4c,y,X) = 18.49337910973606


Out[16]: 18.49337910973606
```

5 Question 5

Using the `aldh2` dataset from the `gap` package in R, try fitting a few random forest (or bagged tree) models using Julia to predict whether a given patient is an alcoholic using their genetic information.

What is the prediction accuracy of your best model? What were the meta-paremeters of your best-fitting model?

The data can be loaded using the code below.

```
In [30]: using RDatasets
         using DecisionTree

         aldh2 = dataset("gap", "aldh2");

In [33]: # must convert to Array
         y = convert(Array{Float64,1}, aldh2[:Y])
         X = convert(Array{Float64,2}, aldh2[:, 3:end])

         # fit model
         fm10 = build_forest(y, X, 10, 500)
```

```
Out[33]: Ensemble of Decision Trees
         Trees:      500
         Avg Leaves: 64.718
         Avg Depth:  14.248
```

```
In [34]: # get prediction accuracy
         yhat_bool = apply_forest(fm10, X) .> 0.5
         ybool = y .== 1
         mean(ybool .== yhat_bool)
```

```
Out[34]: 0.9809885931558935
```

6 Question 6

Use R and the randomForest package via the RCall.jl package in Julia to fit a random forest model on the `chronic_kidney_disease.csv` data set. In particular, fit a model with 5000 trees to predict whether or not patients have chronic kidney disease.

After fitting the model, extract the variable importance estimates (mean Gini decrease) from the fitted model. Pass a dataframe back to Julia that has two columns (1) name of the predictor, and (2) mean Gini decrease for that predictor.

Sort the returned data frame such that larger values are at the top.

The following steps should serve as a general to complete this:

1. Read the data in to Julia
2. Ensure RCall.jl is loaded
3. Pass the dataframe from Julia to R
4. Fit the model in R using `randomForest()` function with argument `ntrees = 5000`
5. Use the `importance()` function to extract the estimates of variable importance from the fitted model
6. Pass the estimates back to Julia

Some Hints:

- The `importance()` function in R returns a data frame whose row names are the variable names, and the last column is mean Gini decrease
- The documentation for the randomForest package in R can be found here: https://cran.r-project.org/web/packages/randomForest/randomForest.pdf
- The `sortperm()` function in Julia will be useful for sorting in the last step

```
In [35]: using DataFrames
         using RCall

         ckd = readtable("../data/chronic_kidney_disease.csv", makefactors = true)
         @rput ckd

         R"
         library(randomForest)

         ckd2 <- ckd[complete.cases(ckd), ]

         fm6 <- randomForest(class ~ ., ckd2, importance = TRUE, ntree = 5000)

         imp_df <- importance(fm6)

         gini_decrease <- imp_df[, ncol(imp_df)]

         preds <- row.names(imp_df)

         var_imp_df <- data.frame(preds, gini_decrease)
         "
         @rget var_imp_df;
```

```
In [36]: indcs = sortperm(var_imp_df[:, 2], rev = true)
         res = var_imp_df[indcs, :]
```

```
Out[36]: 24Œ2 DataFrames.DataFrame
          Row  preds                       gini_decrease

          1    "albumin"                   10.5288
          2    "serum_creatinine"          8.49594
          3    "packed_cell_volume"        8.46403
          4    "hemoglobin"                8.42965
          5    "red_blood_cell_count"      6.2716
          6    "specific_gravity"          5.00724
          7    "blood_urea"                3.82916
          8    "hypertension"              3.46274
          9    "pus_cell"                  1.66581
          10   "diabetes_mellitus"         1.44764
          11   "blood_glucose_random"      1.12767
          12   "sodium"                    0.8763
          13   "white_blood_cell_count"    0.624537
          14   "sugar"                     0.390784
          15   "pedal_edema"               0.339301
          16   "red_blood_cells"           0.298891
          17   "blood_pressure"            0.290002
          18   "appetite"                  0.238108
          19   "anemia"                    0.148841
          20   "pus_cell_clumps"           0.0765711
          21   "bacteria"                  0.0746202
          22   "potassium"                 0.0720374
          23   "age"                       0.0613
          24   "coronary_artery_disease"   0.0250922
```

```
In [ ]:
```