Using LASSO for classification

Jaime Davila

4/10/2021

Introduction

The package `dslabs` has the `tissue_gene_expression` dataset. This dataset contains the gene expression for 500 random genes (out of over 20,000 measured by a microarray) for 189 samples across seven different tissues. Let's load the dataset and take a look at some of the summary statistics:

```
library(dslabs)
data(tissue_gene_expression)
str(tissue_gene_expression)
```

```
## List of 2
##  $ x: num [1:189, 1:500] 9.83 9.63 9.69 9.99 9.58 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:189] "cerebellum_1" "cerebellum_2" "cerebellum_3" "cerebellum_4" ...
##   .. ..$ : chr [1:500] "MAML1" "LHPP" "SEPT10" "B3GNT4" ...
##  $ y: Factor w/ 7 levels "cerebellum","colon",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
dim(tissue_gene_expression$x)
```

```
## [1] 189 500
```

```
table(tissue_gene_expression$y)
```

```
##
##  cerebellum       colon endometrium hippocampus      kidney       liver
##          38          34          15          31          39          26
##    placenta
##           6
```

Setting up our dataset

On a first iteration we are interested in creating a classifier function that will allow us to distinguish between `cerebellum` and `colon` based on their gene expression profile. We can do it as follows:

```
tissue.levels<- c("cerebellum","colon")
sample.ids <- tissue_gene_expression$y %in% tissue.levels

tissue.gene.tbl <- tissue_gene_expression$x[sample.ids,] %>%
  as_tibble() %>%
  mutate(tissue = factor(tissue_gene_expression$y[sample.ids], levels=tissue.levels))
```

And let's divide dataset into training/testing datasets:

```
set.seed(123456)
tissue.split <- initial_split(tissue.gene.tbl, prop=0.5)
tissue.train.tbl <- training(tissue.split)
tissue.test.tbl <- testing(tissue.split)
```

Finally, let's check the dimension of the training dataset:

```
dim(tissue.train.tbl)
```

```
## [1]  36 501
```

Notice that we only have 36 observations and we have 500 variables!

1. Talk to the people in your group and try to explain why logistic regression would not work using this training table.

Using LASSO for binary classification

We would like to build a model that will allow us to predict the tissue type based on the gene expression. Furthermore we would like to identify a small number of features (variables) to use in this model. Given LASSO's ability to identify a small subset of variables, seems this method is particularly well-suited for the problem.

Notice that we have used LASSO only for prediction so far, however `tidymodels` and `glmnet` allows us to use LASSO for classification by using the following syntax

```
tissue.model <-
  logistic_reg(mixture = 1, penalty=tune()) %>%
  set_mode("classification") %>%
  set_engine("glmnet")
```

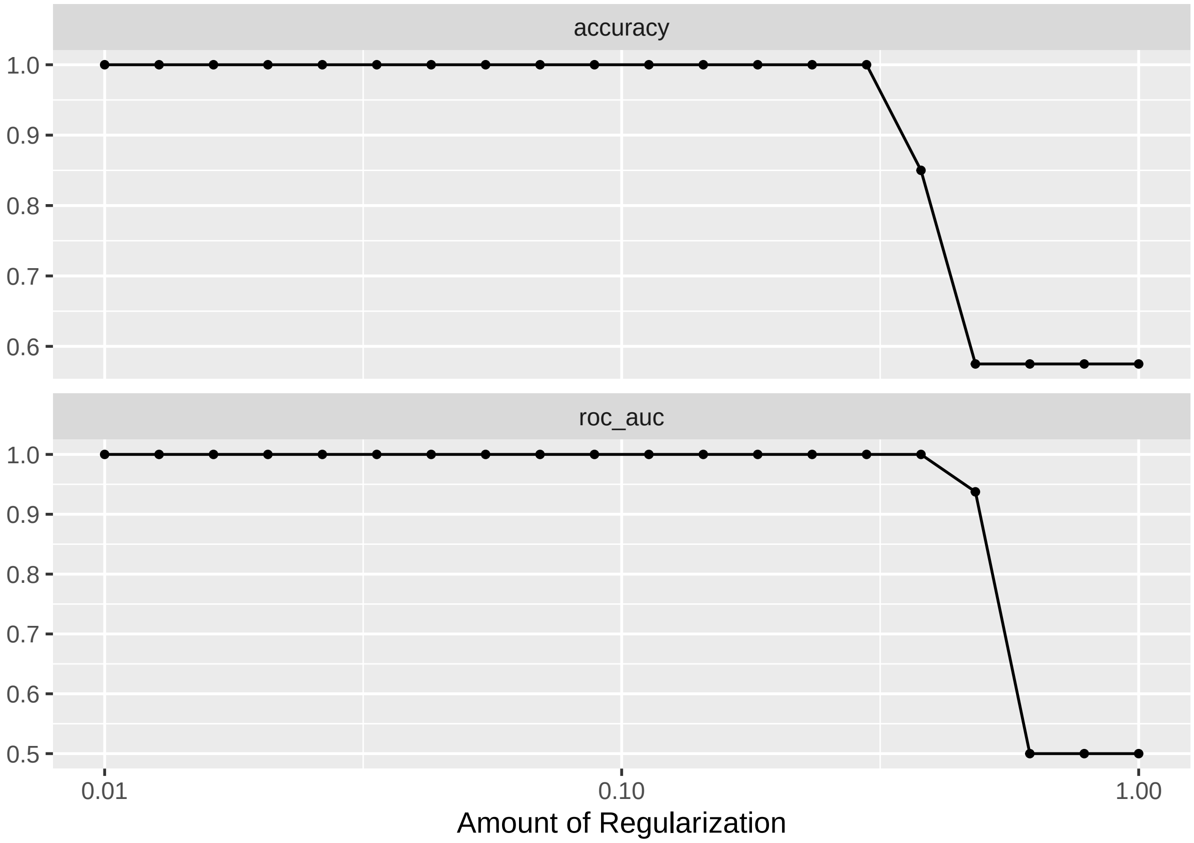
2. Create a LASSO model and optimize the parameter λ by following these steps

a. Create a recipe `tissue.recipe` that would predict tissue type. Justify whether or not you need to use `step_dummy()` as a step in your recipe? How about `step_normalize()`? Create a `tissue.wf` workflow by combining the recipe and the model.

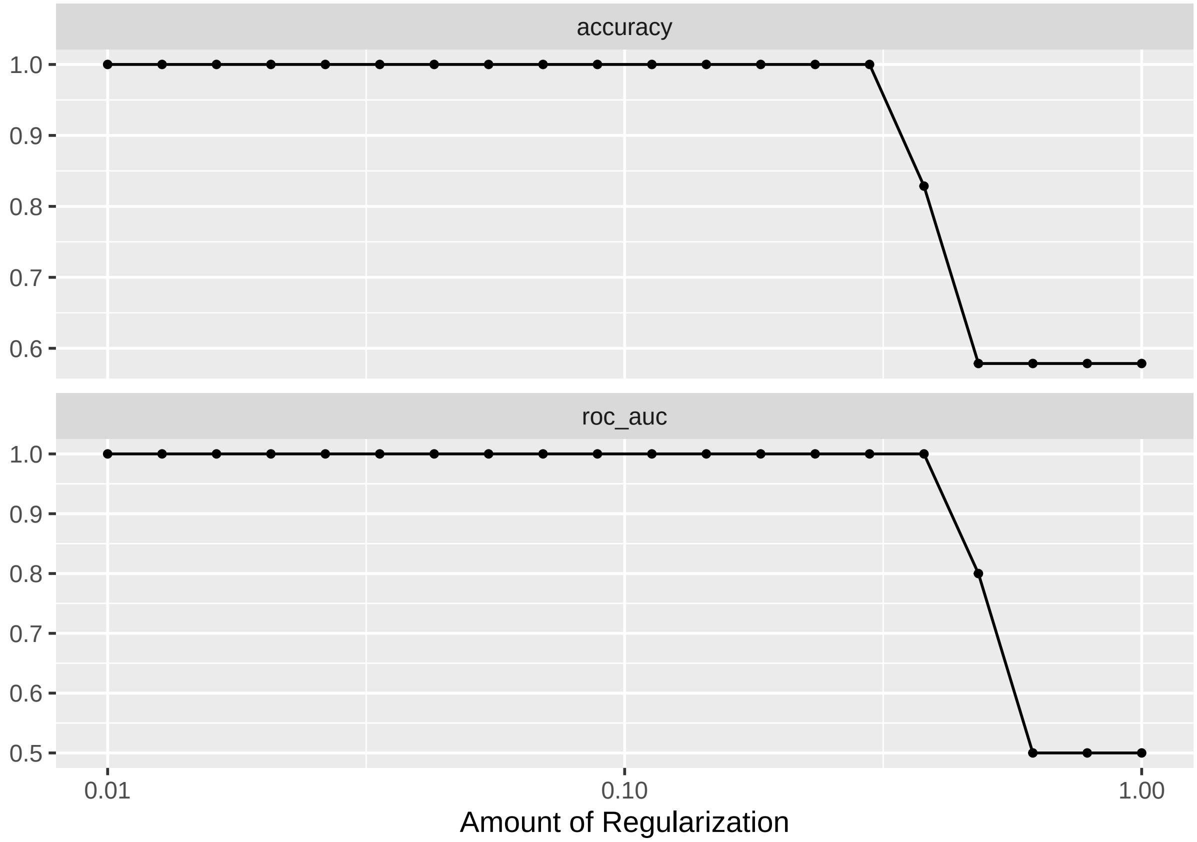
```
tissue.recipe <-
  recipe(formula = tissue ~ ., data = tissue.train.tbl) %>%
  step_normalize(all_predictors())

tissue.wf <- workflow() %>%
  add_recipe(tissue.recipe) %>%
  add_model(tissue.model)
```

```
b. Create a 5 fold and a 10-fold cross validation dataset ‘tissue.fold‘. Create a grid ‘penalty.grid‘ be
```

```
set.seed(1234)
tissue.fold.10 <- vfold_cv(tissue.train.tbl, v = 10)
tissue.fold.5 <- vfold_cv(tissue.train.tbl, v = 5)

penalty.grid <-
  grid_regular(penalty(range = c(-2, 0)), levels = 20)

tune.res <- tune_grid(
  tissue.wf,
  resamples = tissue.fold.10,
  grid = penalty.grid
)
autoplot(tune.res)
```



```
tune.res <- tune_grid(
  tissue.wf,
  resamples = tissue.fold.5,
  grid = penalty.grid
)
autoplot(tune.res)
```



```
c. Select the best penalty by using ‘select_by_one_std_err()‘ using accuracy as your metric and sorting
```

```
show_best(tune.res, metric = "accuracy")
```

```
## # A tibble: 5 x 7
##   penalty .metric  .estimator  mean     n std_err .config
##     <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <fct>
## 1  0.01   accuracy binary         1     5       0 Preprocessor1_Model01
## 2  0.0127 accuracy binary         1     5       0 Preprocessor1_Model02
## 3  0.0162 accuracy binary         1     5       0 Preprocessor1_Model03
## 4  0.0207 accuracy binary         1     5       0 Preprocessor1_Model04
## 5  0.0264 accuracy binary         1     5       0 Preprocessor1_Model05
```

```
(best.penalty <- select_by_one_std_err(tune.res,
                                       metric = "accuracy",
                                       desc(penalty)))
```

```
## # A tibble: 1 x 9
##   penalty .metric  .estimator  mean     n std_err .config           .best .bound
##     <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <fct>             <dbl>  <dbl>
## 1   0.298 accuracy binary         1     5       0 Preprocessor1_Mo~     1      1
```

```
tissue.final.wf <- finalize_workflow(tissue.wf, best.penalty)
tissue.final.fit <- fit(tissue.final.wf, data = tissue.train.tbl)

augment(tissue.final.fit, new_data = tissue.test.tbl) %>%
  conf_mat(truth = tissue, estimate = .pred_class)
```

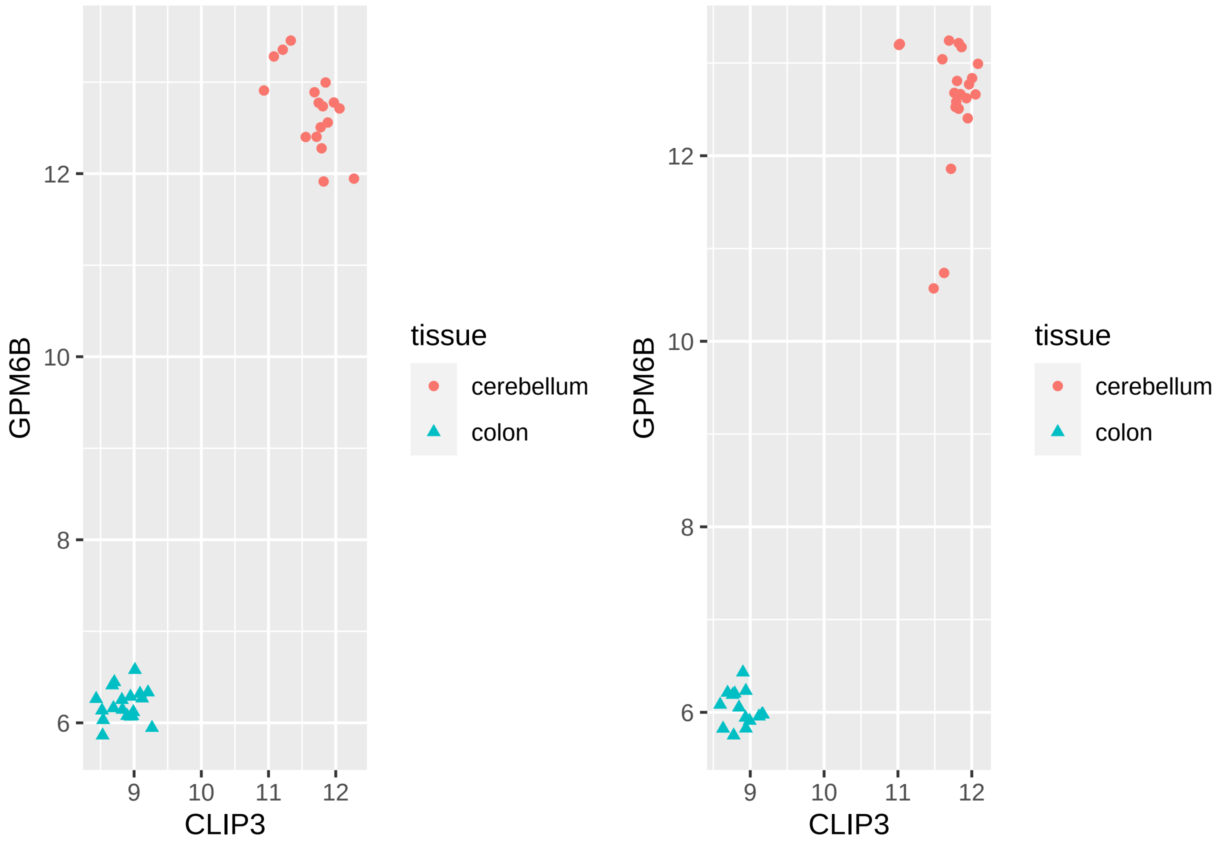
```
##               Truth
## Prediction   cerebellum colon
##   cerebellum         17     0
##   colon               0    19
```

d. Determine which coefficients in your LASSO model are non-zero. Do a quick google search for “GPM6B genecards”. Does it make sense that this gene distinguishes between cerebellum and colon? Plot the values of these two genes in your training and testing dataset? Are the values of these two genes very different across the two tissue types?

```
tidy(tissue.final.fit) %>% filter(estimate!=0)
```

```
## # A tibble: 3 x 3
##   term        estimate penalty
##   <chr>          <dbl>   <dbl>
## 1 (Intercept)   -0.381   0.298
## 2 CLIP3         -0.437   0.298
## 3 GPM6B         -0.394   0.298
```

```
# library(gridExtra)
gg1 <- ggplot (tissue.test.tbl, aes(x=CLIP3, y=GPM6B, color=tissue, shape=tissue))+
    geom_point()
gg2 <- ggplot (tissue.train.tbl, aes(x=CLIP3, y=GPM6B, color=tissue, shape=tissue))+
    geom_point()

grid.arrange(gg1,gg2,ncol=2)
```



Multiple tissue classification

Now that we gained some confidence in distinguishing between two tissues, we would like to create a more complex classifier. First, let's take a look at the number of tissues in our dataset

```
table(tissue_gene_expression$y)
```

```
##
##  cerebellum       colon endometrium hippocampus      kidney       liver
##          38          34          15          31          39          26
##    placenta
##           6
```

It seems we don't have enough placenta tissues in our dataset, so we will exclude them from our dataset.

```
placenta.idx <-which(tissue_gene_expression$y=="placenta")
tissue_gene_expression$x <- tissue_gene_expression$x[-placenta.idx,]
tissue_gene_expression$y <- droplevels(tissue_gene_expression$y[-placenta.idx])

multiple.tissue.gene.tbl <- tissue_gene_expression$x[-placenta.idx,] %>%
  as_tibble() %>%
  mutate(tissue = droplevels(tissue_gene_expression$y[-placenta.idx]))

table(multiple.tissue.gene.tbl$tissue)
```

```
##
##  cerebellum       colon endometrium hippocampus      kidney       liver
##          38          34          15          31          39          26
```

And create a testing/training dataset

```
set.seed(6543)
tissue.split <- initial_split(multiple.tissue.gene.tbl, prop=0.5)
multiple.tissue.train.tbl <- training(tissue.split)
table(multiple.tissue.train.tbl$tissue)
```

```
##
##  cerebellum       colon endometrium hippocampus      kidney       liver
##          20          17          11          14          18          11
```

```
multiple.tissue.test.tbl <- testing(tissue.split)
table(multiple.tissue.test.tbl$tissue)
```

```
##
##  cerebellum       colon endometrium hippocampus      kidney       liver
##          18          17           4          17          21          15
```

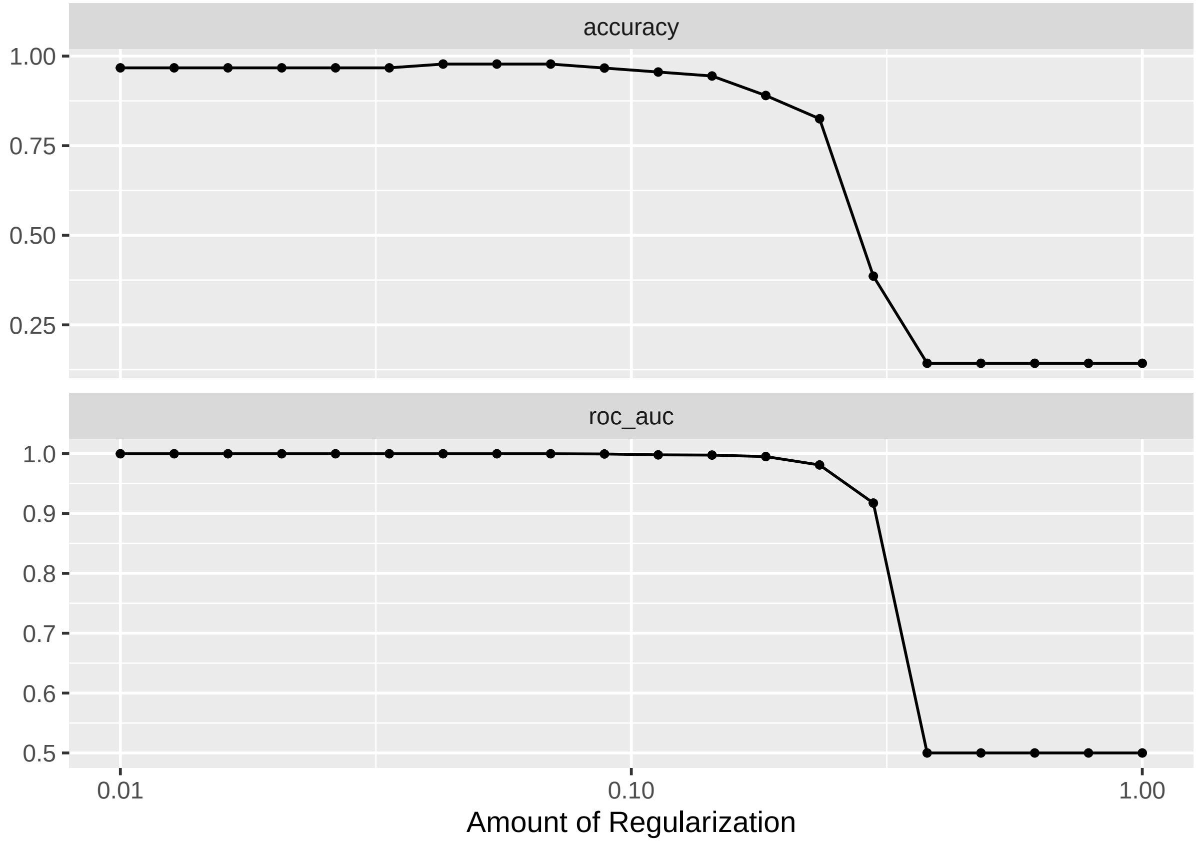
Finally we can set up our model by making use of `multinom_reg()`

```
tissue.model <-
  multinom_reg(mixture = 1, penalty=tune()) %>%
  set_mode("classification") %>%
  set_engine("glmnet")
```

3. Create a LASSO model and use 5-fold cross validation and `select_by_one_std_err()` to determine the optimal penalty. What is the accuracy of LASSO on the testing dataset? How does the confusion matrix look on the testing dataset?

```r
multiple.tissue.recipe <-
  recipe(formula = tissue ~ ., data = multiple.tissue.train.tbl) %>%
  step_normalize(all_predictors())

multiple.tissue.wf <- workflow() %>%
  add_recipe(multiple.tissue.recipe) %>%
  add_model(tissue.model)
```

```r
set.seed(1234)
multiple.tissue.fold <- vfold_cv(multiple.tissue.train.tbl, v = 5)

tune.res <- tune_grid(
  multiple.tissue.wf,
  resamples = multiple.tissue.fold,
  grid = penalty.grid
)
autoplot(tune.res)
```



```r
(best.penalty <- select_by_one_std_err(tune.res, metric = "accuracy", desc(penalty)))
```

```
## # A tibble: 1 x 9
##   penalty .metric  .estimator  mean     n std_err .config            .best .bound
##     <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <fct>              <dbl>  <dbl>
## 1  0.0886 accuracy multiclass 0.967     5  0.0222 Preprocessor1_Mo~  0.978  0.964
```

```r
multiple.tissue.final.wf <- finalize_workflow(multiple.tissue.wf, best.penalty)
tissue.final.fit <- fit(multiple.tissue.final.wf, data = multiple.tissue.train.tbl)
```

```
augment(tissue.final.fit, new_data = multiple.tissue.test.tbl) %>%
  accuracy(truth = tissue, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.989
```

```
augment(tissue.final.fit, new_data = multiple.tissue.test.tbl) %>%
  conf_mat(truth = tissue, estimate = .pred_class)
```

```
##              Truth
## Prediction    cerebellum colon endometrium hippocampus kidney liver
##   cerebellum          17     0           0           0      0     0
##   colon                0    17           0           0      0     0
##   endometrium          0     0           4           0      0     0
##   hippocampus          0     0           0          17      0     0
##   kidney               1     0           0           0     21     0
##   liver                0     0           0           0      0    15
```

4. a. Using the command `tidy()` determine the non-zero coefficients of your model (please remove the constant terms). You should have about 21 non-zero coefficients. How do you interpret these terms? How many non-zero coefficients correspond to each tissue?

```
tidy(tissue.final.fit) %>%
  filter(estimate!=0 & term!="(Intercept)")
```

```
## # A tibble: 24 x 4
##    class       term     estimate penalty
##    <chr>       <chr>       <dbl>   <dbl>
##  1 cerebellum  LRRN3      0.0183  0.0886
##  2 cerebellum  KCTD2      0.395   0.0886
##  3 cerebellum  KCNJ12     0.680   0.0886
##  4 cerebellum  ASTN2      0.295   0.0886
##  5 colon       H2AFY      0.0472  0.0886
##  6 colon       GPA33      0.576   0.0886
##  7 colon       GTF2IRD1   0.206   0.0886
##  8 colon       CEP55      0.418   0.0886
##  9 endometrium FAP        0.267   0.0886
## 10 endometrium FBN1       0.769   0.0886
## # ... with 14 more rows
```

```
tidy(tissue.final.fit) %>%
  filter(estimate!=0 & term!="(Intercept)")  %>%
  group_by(class) %>%
  summarize(n=n()) %>%
  arrange(-n)
```

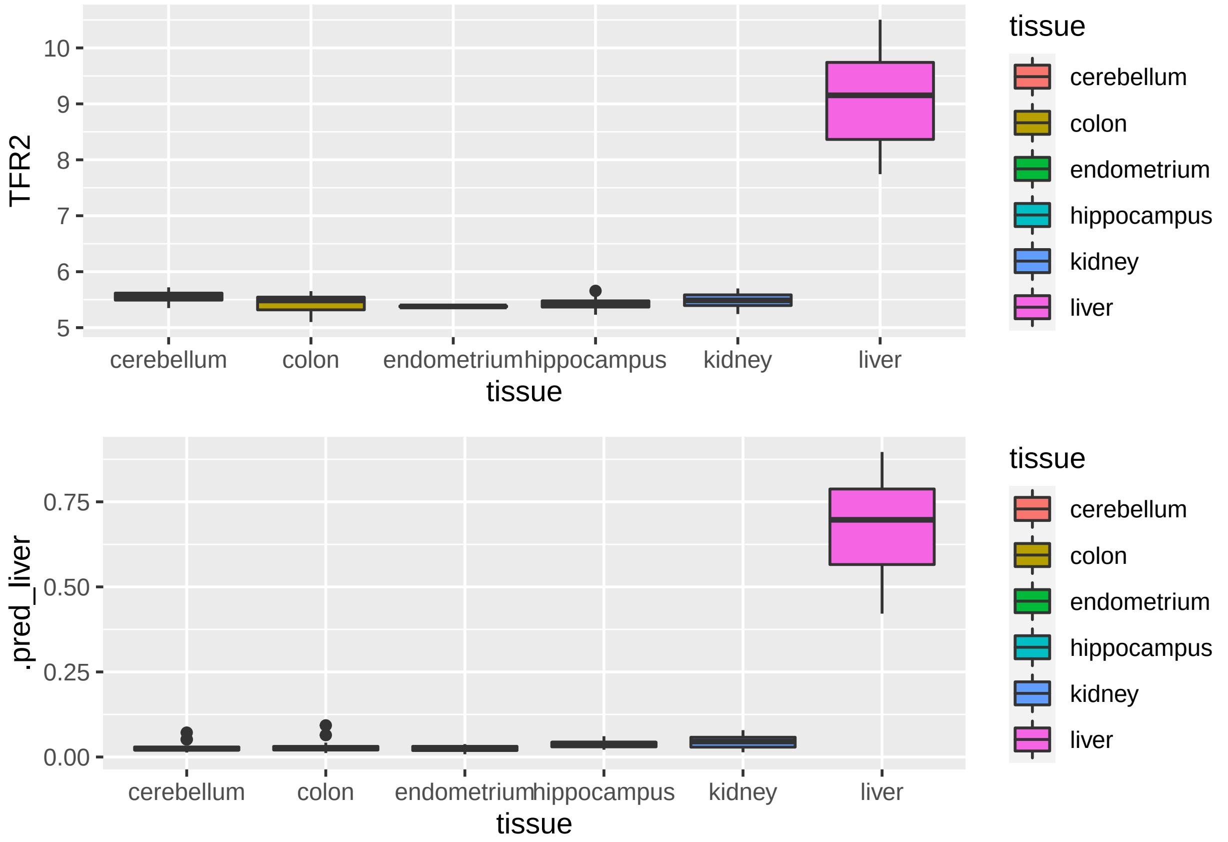
```
## # A tibble: 6 x 2
##   class           n
##   <chr>       <int>
## 1 kidney          8
## 2 hippocampus     5
## 3 cerebellum      4
## 4 colon           4
## 5 endometrium     2
## 6 liver           1
```

b. What is the gene that allows you to distinguish a liver? Do a google search of the gene plus genecards and see if it makes sense. Do a boxplot of the value of the gene across the tissues from your testing dataset and interpret it. Do a boxplot of the predicted probability of being a liver across all the tissues from your testing dataset and interpret it.

```
tidy(tissue.final.fit) %>%
  filter(estimate!=0 & term!="(Intercept)")  %>%
  filter(class=="liver")
```

```
## # A tibble: 1 x 4
##   class term  estimate penalty
##   <chr> <chr>    <dbl>   <dbl>
## 1 liver TFR2      1.05  0.0886
```

```
test.pred.tbl <-
  augment(tissue.final.fit, new_data = multiple.tissue.test.tbl)

gg1 <- ggplot(test.pred.tbl, aes(x=tissue, y=TFR2, fill=tissue))+
  geom_boxplot()

gg2 <- ggplot(test.pred.tbl, aes(x=tissue, y=.pred_liver, fill=tissue))+
  geom_boxplot()

grid.arrange(gg1,gg2,nrow=2)
```



5. a. In your testing dataset there was a cerebellum that was missclassified as a kidney. Identify this

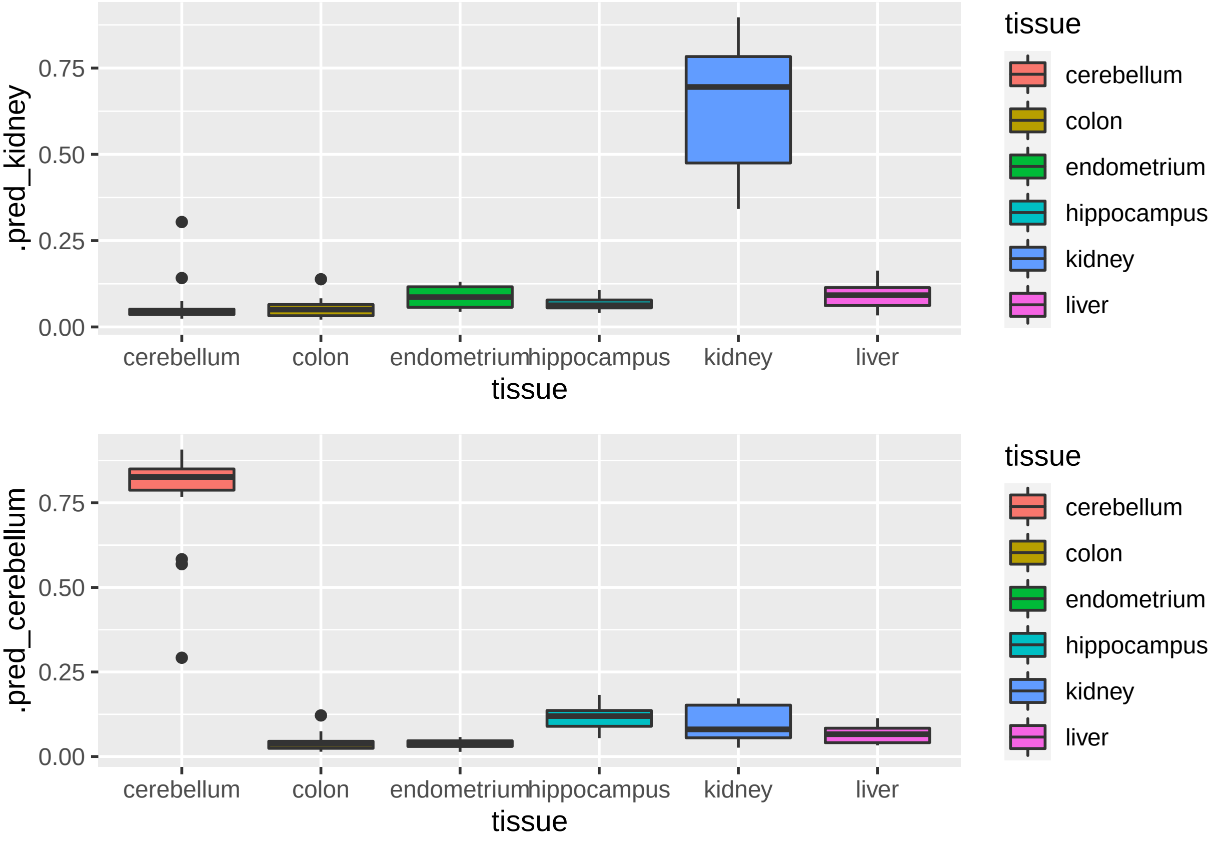
observation and determine its predicted probability of being each distinct tissue.

```
test.pred.tbl %>%
  filter(tissue=="cerebellum" & .pred_class=="kidney") %>%
  select(.pred_cerebellum, .pred_colon, .pred_endometrium,
         .pred_hippocampus, .pred_kidney, .pred_liver)
```

```
## # A tibble: 1 x 6
##   .pred_cerebellum .pred_colon .pred_endometrium .pred_hippocampus .pred_kidney
##              <dbl>       <dbl>             <dbl>             <dbl>        <dbl>
## 1            0.292       0.101            0.0890             0.142        0.304
## # ... with 1 more variable: .pred_liver <dbl>
```

b. Using your testing dataset do a boxplot of the predicted probability of being a cerebellum and the predicted probability of being a kidney. Can you identify the misclassified sample in the two boxplots?

```
gg1 <- ggplot(test.pred.tbl, aes(x=tissue, y=.pred_kidney, fill=tissue))+
  geom_boxplot()

gg2 <- ggplot(test.pred.tbl, aes(x=tissue, y=.pred_cerebellum, fill=tissue))+
  geom_boxplot()

grid.arrange(gg1,gg2,nrow=2)
```


.pred_kidney
cerebellum
colon
endometrium
hippocampus
kidney
liver
tissue
tissue
cerebellum
colon
endometrium
hippocampus
kidney
liver
.pred_cerebellum
cerebellum
colon
endometrium
hippocampus
kidney
liver
tissue
tissue
cerebellum
colon
endometrium
hippocampus
kidney
liver