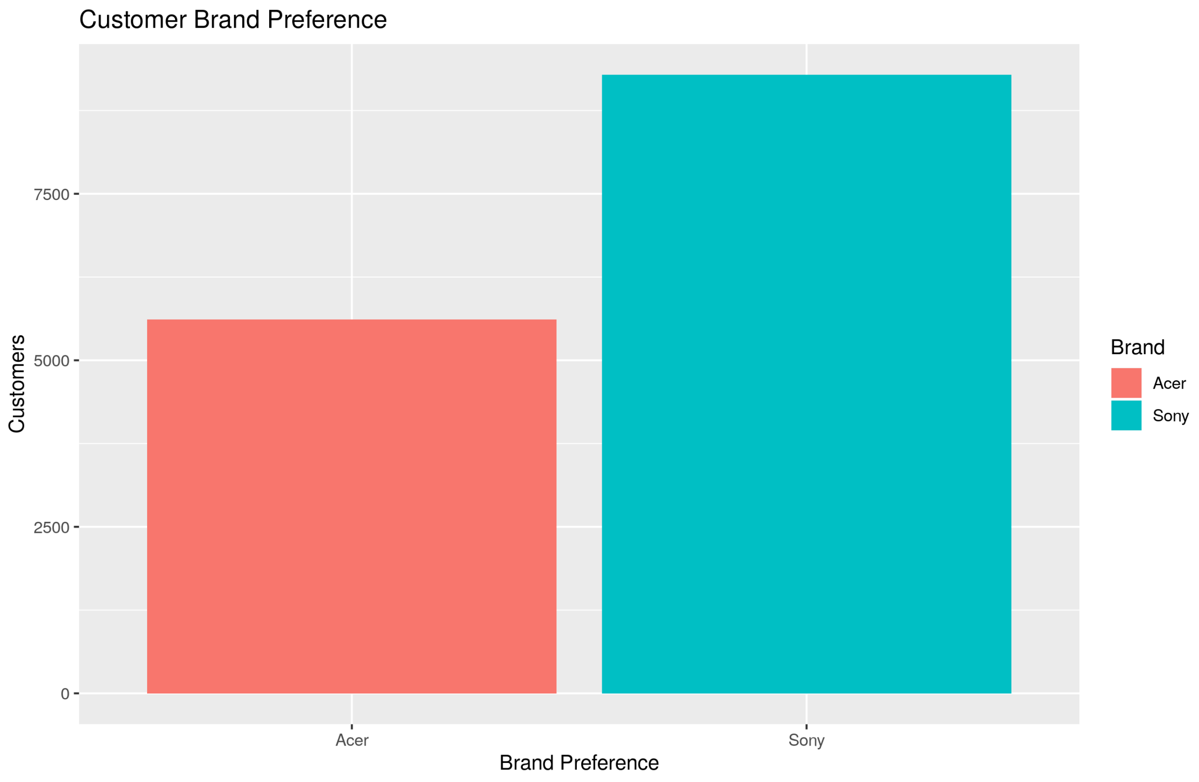
Data Analytics 2 - Predicting Brand Preference

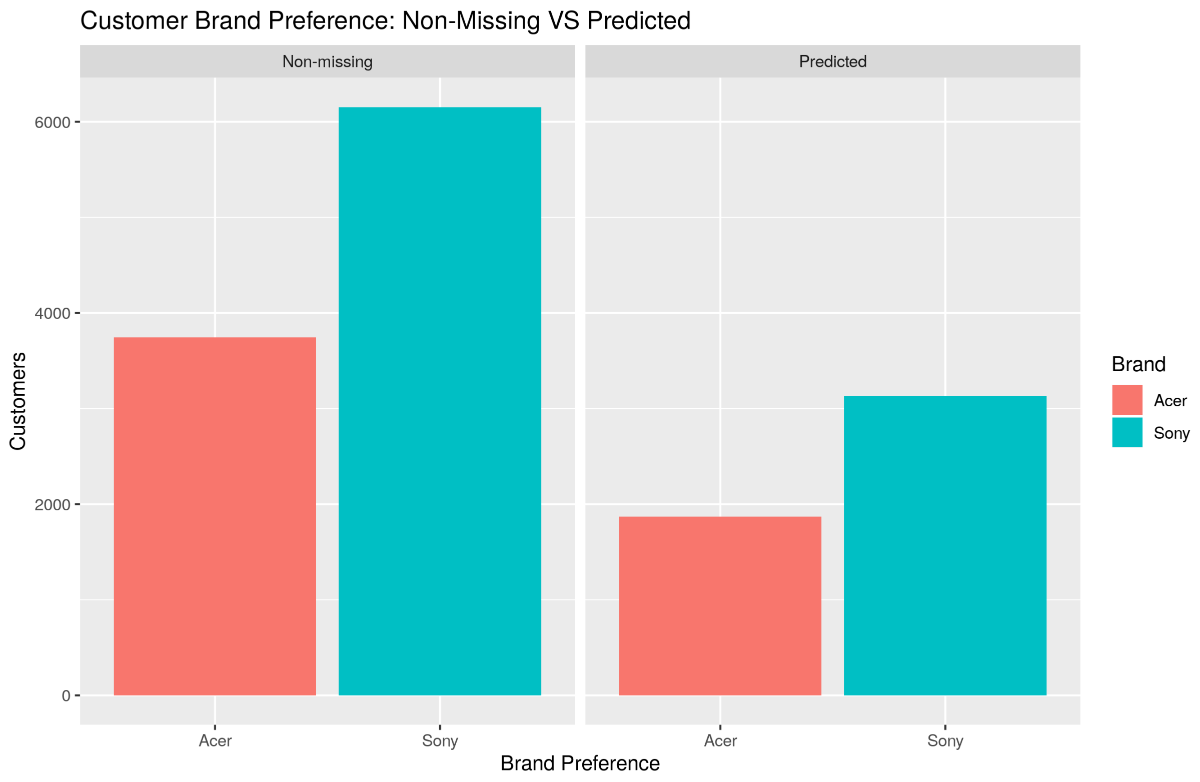
Tuomo Kareoja

Version Number	Changes	
0.1	Created basic outline without content	01.0
0.2	Added visualizations, printouts and text outlines	02.0
0.9	Better visualizations and more explanations. Abandoned ensemble models	05.0

Brand Preference with Predicted Missing Values



These are results of the brand preference question after we added in our predictions. We can wee that Sony is clearly the more preferred brand, but that the difference is not huge.



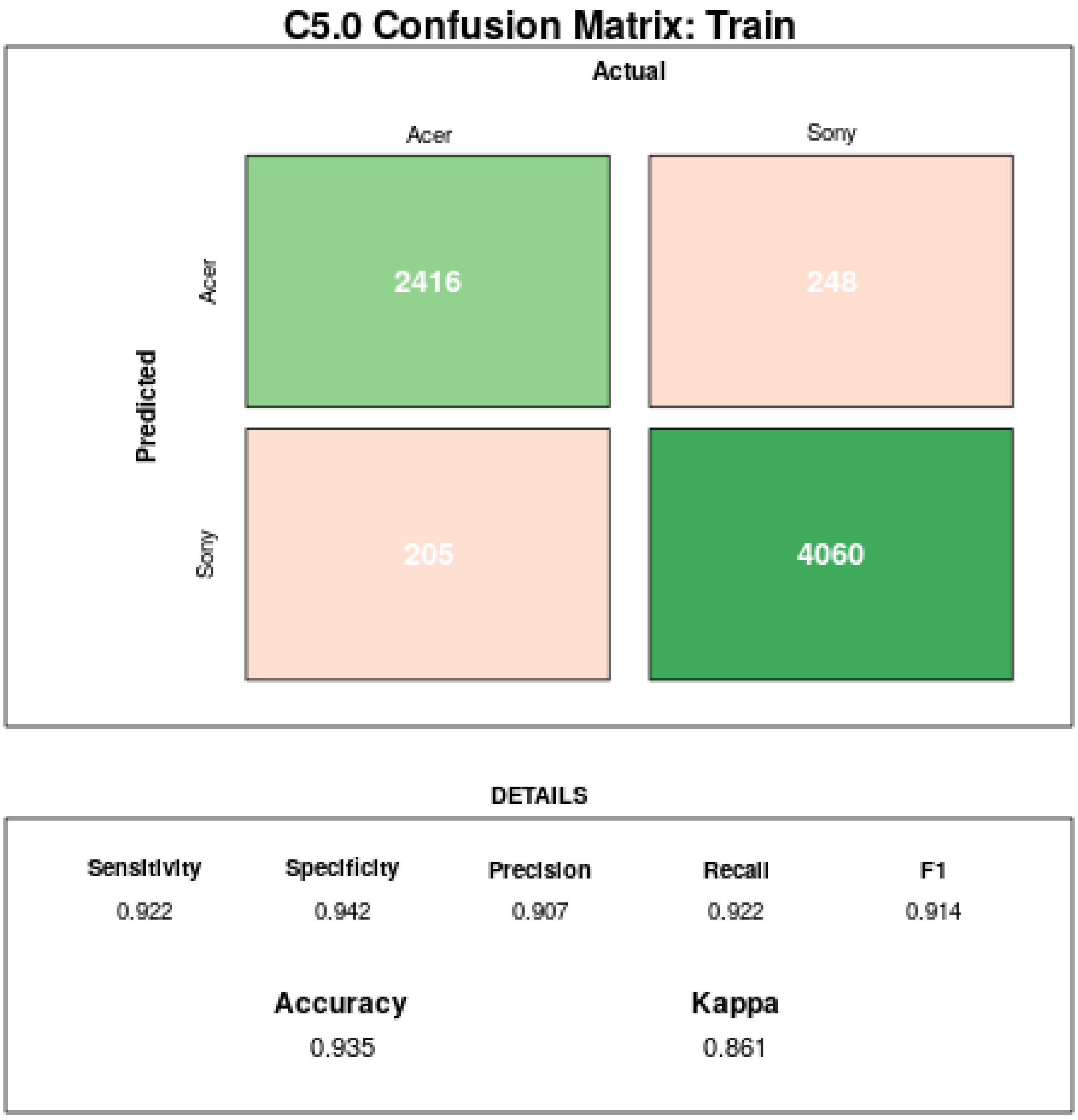
Does look this is actually missing at random in relation to the brand preference.

TODO check percentages

Here we can see that if we would have just used the data without missing values our estimate of the difference between the two brand would have been much bigger. In the predicted values there much more customers who are predicted to prefer Acer more than Sony. Assuming that our model is correct, this means that the data about brand preference is not missing at random and that using just the non-missing values would lead to quite biased conclusions.

Chosen Model and Its' Performance

The model used for the predictions is a C5.0 classification tree. From the the graphs below we can see that on all the important metrics this model performs really well with the Test set accuracy score being. If we compare the test set and train set accuracy metric we can see that the model also seems to have almost no overfit.



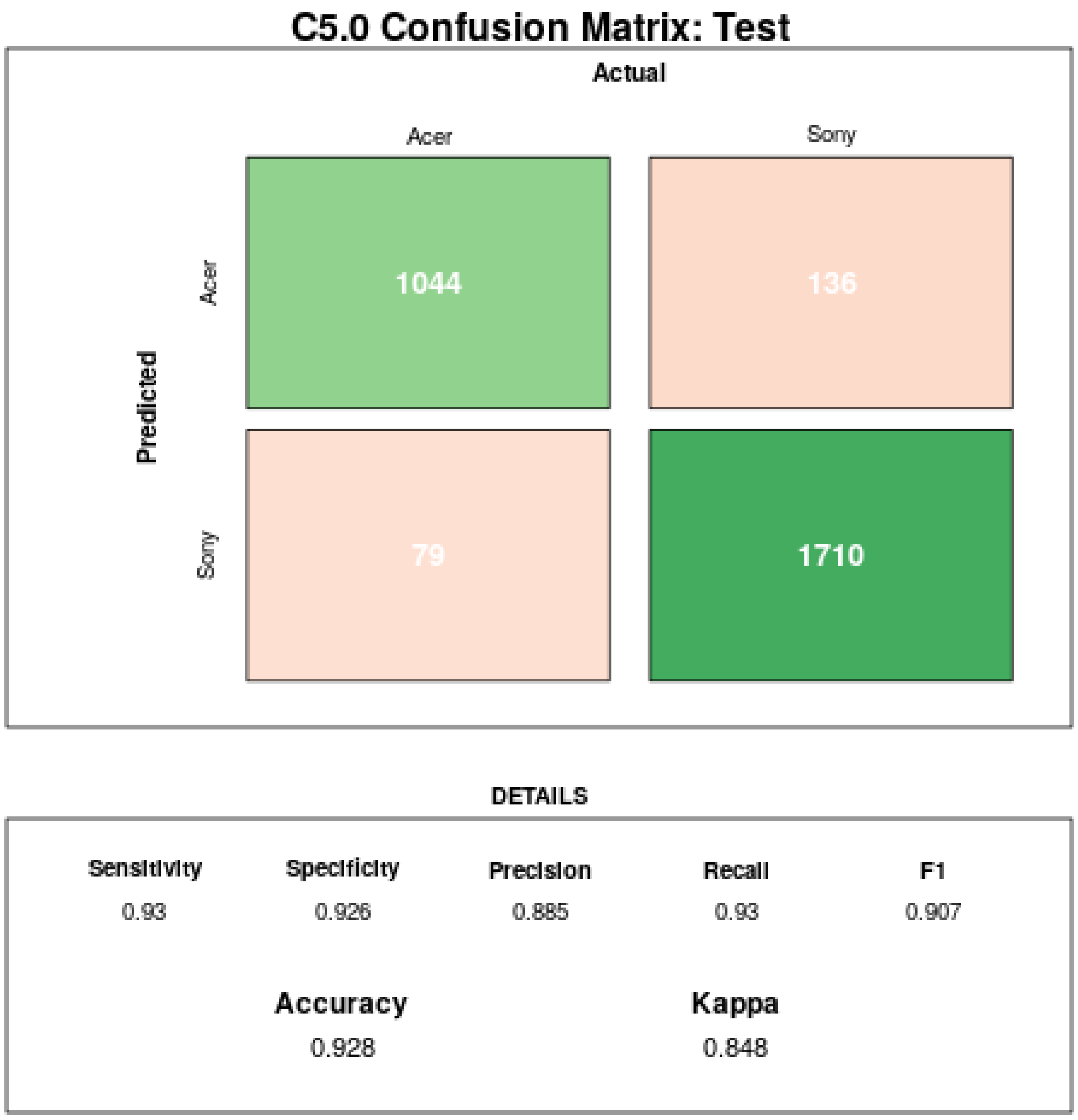
C5.0 Confusion Matrix: Train

Predicted \ Actual	Acer	Sony
Acer	2416	248
Sony	205	4060

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.922	0.942	0.907	0.922	0.914

Accuracy	Kappa
0.935	0.861



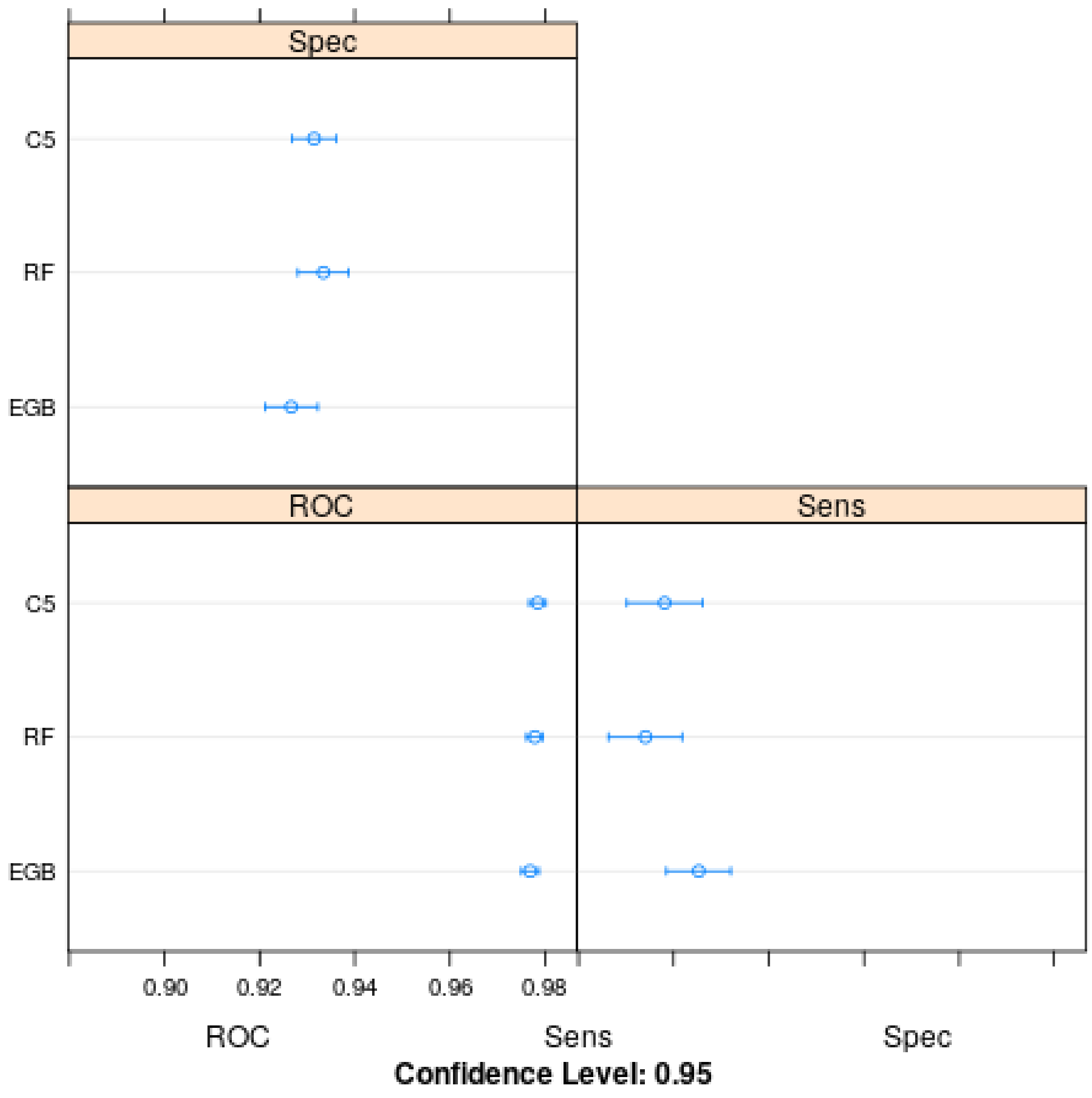
As the C5.0 decision tree is too complex to understandably visualize. We can instead look at the relative model feature importances:

We can see that by far the most import variable is the salary of the customer. If we plot the distribution of salary with the brand preference, we can see that there is clearly differences but that these are quite complex and nonlinear and would have not been easily captured with a linear model.

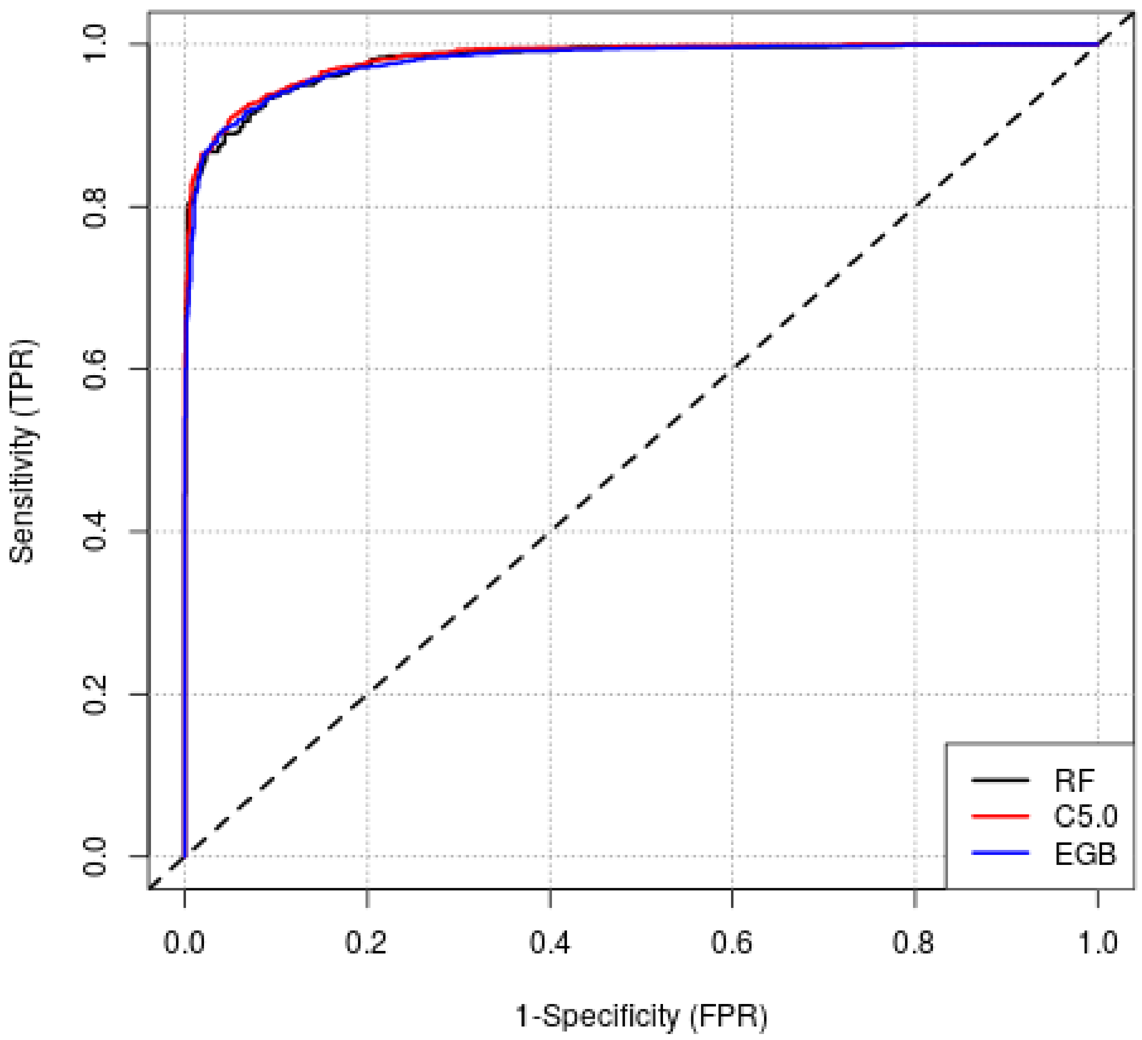
Model Comparison and Performance

In total three different model types were tried: C5.0, Random Forest (RF) and Extreme Gradient Boosting (EGB). All of these model are tree based models and all performed excellently. This made is quite problematic to choose the best model.

From the cross validation boxplots we can see that all models perform very well in all metrics and it is hard to decide which is the best:

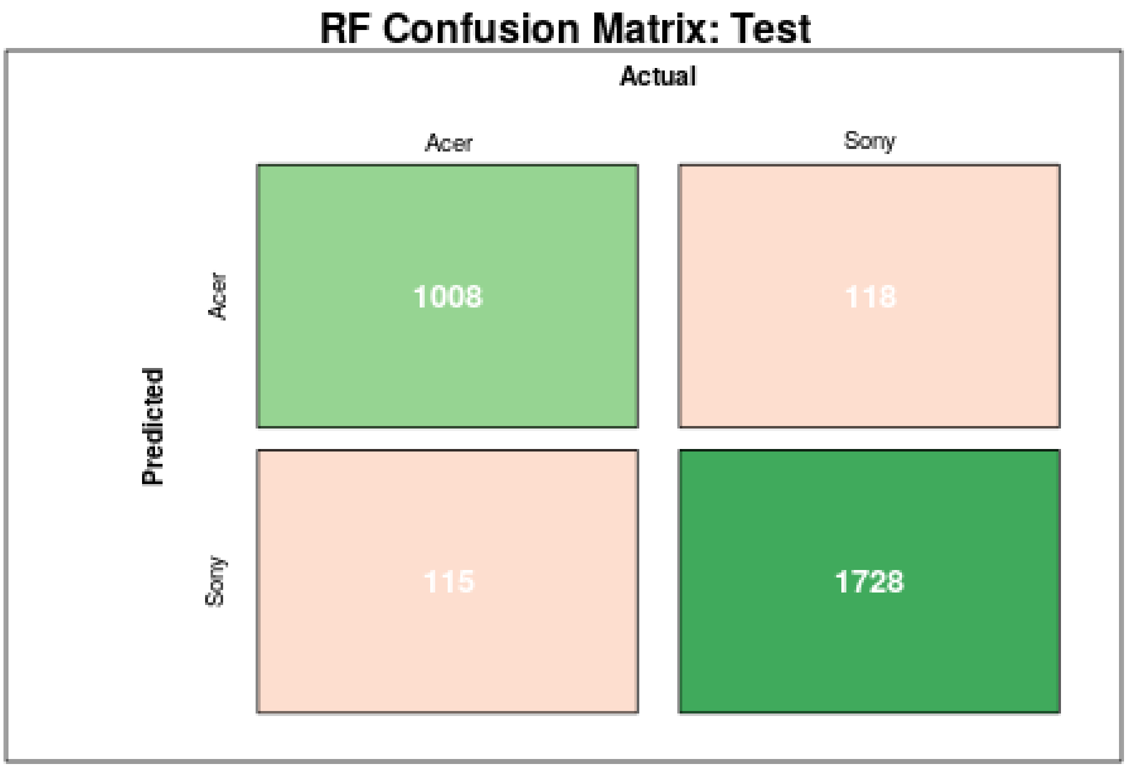


The small differences between the models can best be seen in the ROC-curve:



The models are almost indistinguishable, with our chosen C5.0 model maybe somewhat getting a head around the optimal (Youden Index) point.

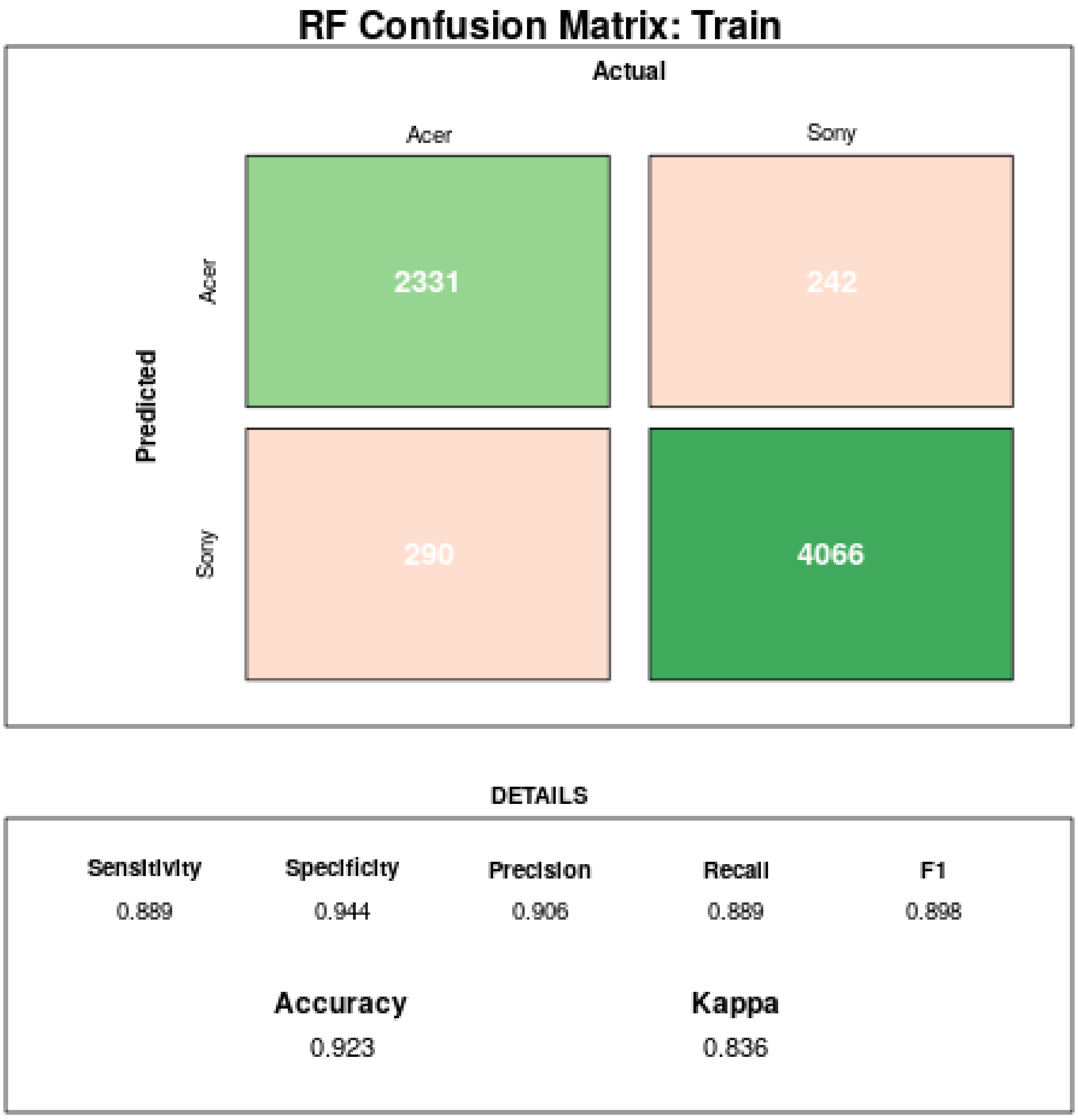
Random Forests confusion matrix test and train



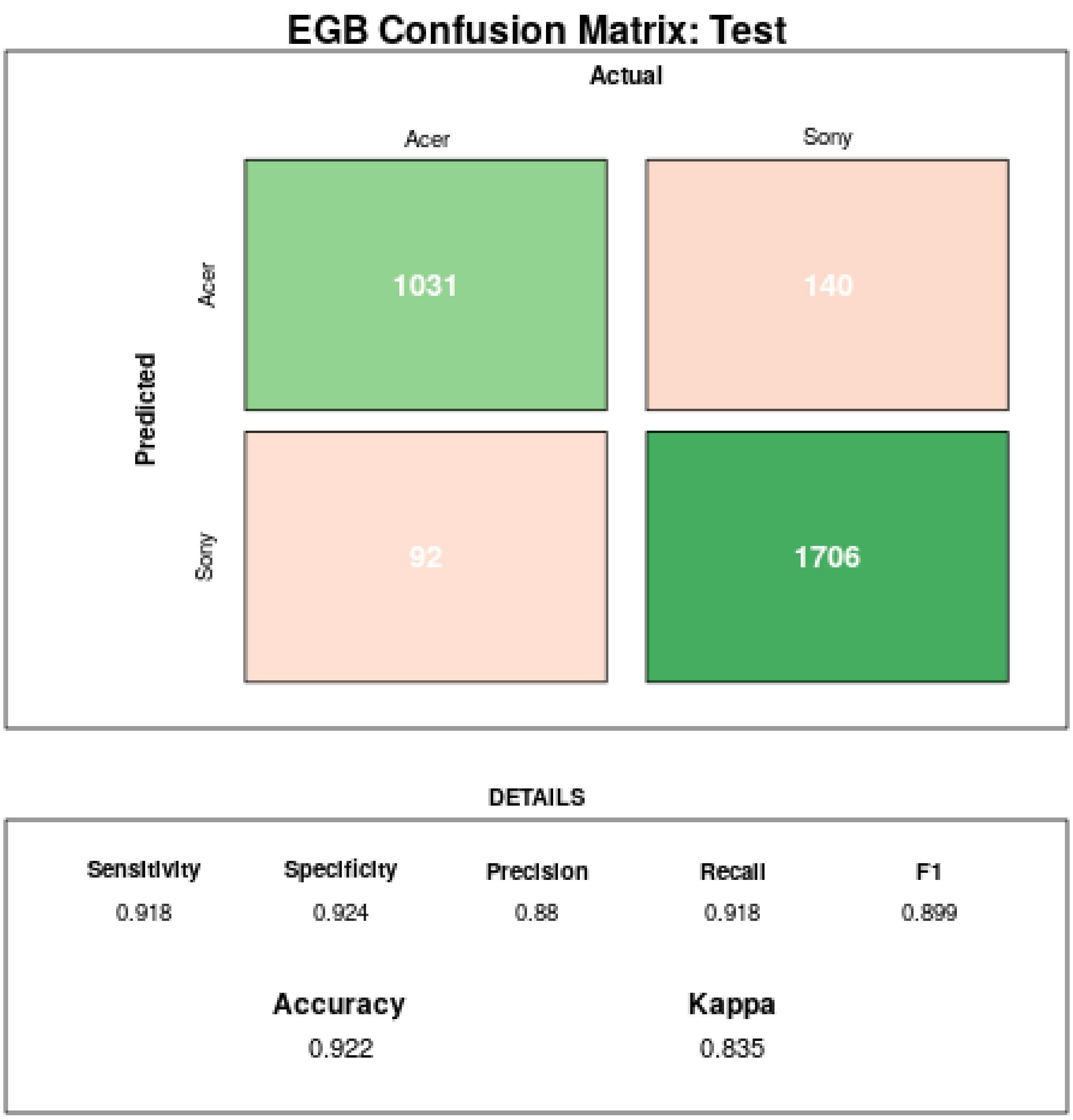
DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.898	0.936	0.895	0.898	0.896

Accuracy	Kappa
0.922	0.833



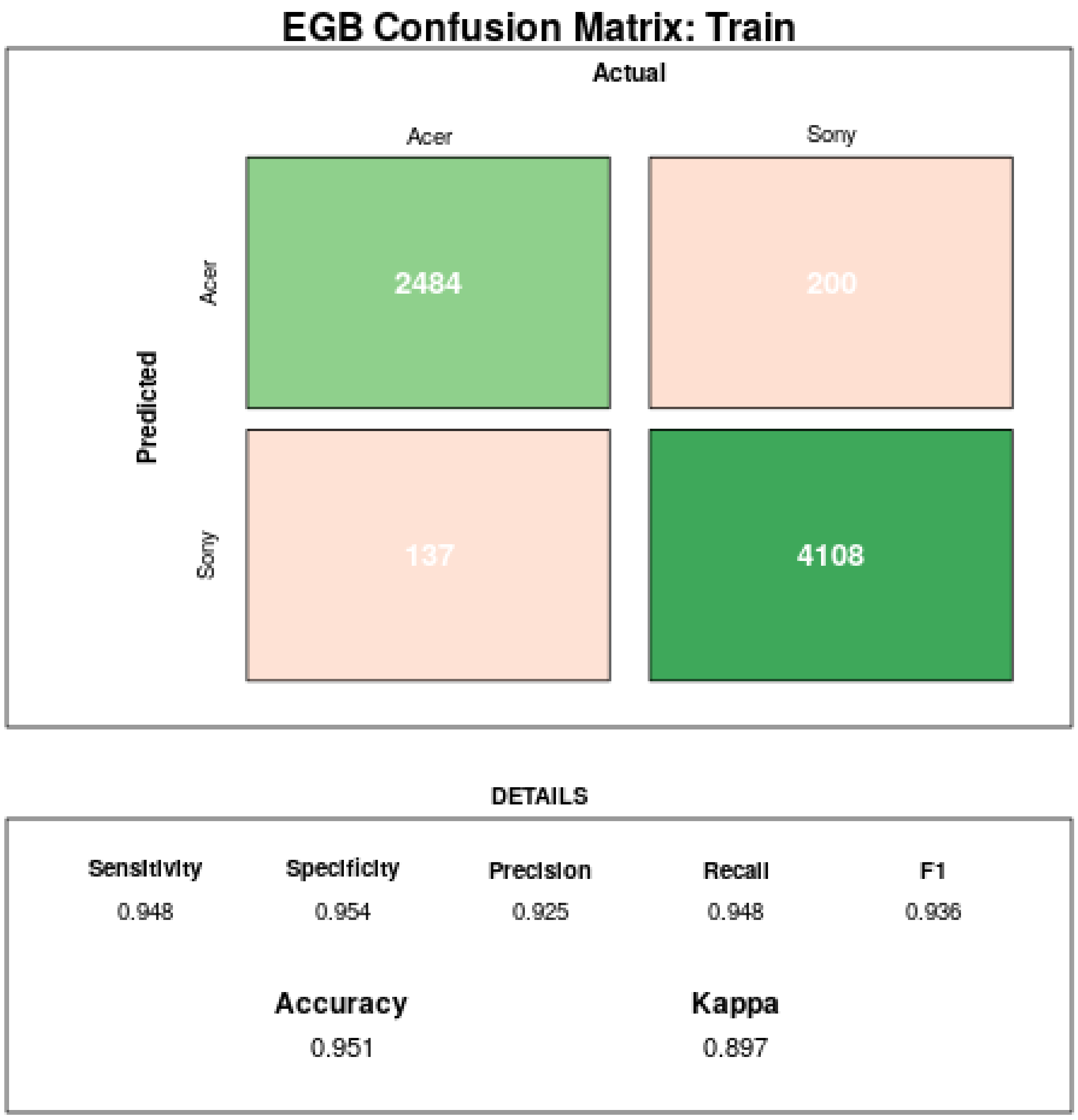
Extreme Gradient Boosting confusion matrix test and train



Predicted \ Actual	Acer	Sony
Acer	1031	140
Sony	92	1706

Sensitivity	Specificity	Precision	Recall	F1
0.918	0.924	0.88	0.918	0.899

Accuracy	Kappa
0.922	0.835



Model prediction correlation is quite low (cross validated scores don't line up neatly on the diagonal).

This opened up the possibility of trying out an ensemble model that combines all the models, so that they could compensate for each others weaknesses. This was tried out with the combinations of two models and all three models. Accuracy of the model improved slightly, but at the same time difference between test accuracy and train accuracy grew. Because the gains where small, but the change of overfitting much bigger, this venture was abandoned and a more simple model chosen. In the end C5.0 won, because of the extremely small difference between test and train accuracy. This model seemed to offer accurate predictions combined with the lowest change of overfitting.

Here are the command line summaries of the models in cross validation state:

Listing 1: C5.0

```
> summary(model_c5)

Call:
(function (x, y, trials = 1, rules = FALSE, weights = NULL, control = C5.0C
funcCall
 118811.6, 97162.92, 41801.08, 83385.8, 108972.8, 50444.96, 82882.6, 49168.
 129376.1, 101090.2, 129777.1, 45991.51, 63077.13, 36877.89, 37634.


C5.0 [Release 2.07 GPL Edition]        Mon Aug  5 11:51:56 2019
-------------------------------

Class specified by attribute ‘outcome’

Read 6929 cases (38 attributes) from undefined.data

15 attributes winnowed
Estimated importance of remaining attributes:

   307%  salary
   190%  age
     1%  elevel.Some.College
     1%  elevel.High.School.Degree
     1%  elevel.Master.s..Doctoral.or.Professional.Degree
    <1%  elevel.4.Year.College.Degree
    <1%  car.Chrysler
    <1%  car.None.of.the.above
    <1%  elevel.Less.than.High.Scool.Degree
    <1%  car.BMW
    <1%  car.Dodge
    <1%  car.Ford
    <1%  car.Honda
    <1%  car.Hyundai
    <1%  car.Jeep
    <1%  car.Lincoln
    <1%  car.Mazda
    <1%  car.Mitsubishi
    <1%  zipcode.East.South.Central
    <1%  zipcode.Mountain
    <1%  zipcode.Pacific
    <1%  credit

-----  Trial 0:  -----

Decision tree:
```

Listing 2: Extreme Gradient Boosting

```
> model_egb
eXtreme Gradient Boosting

6929 samples
  37 predictor
   2 classes: 'Acer', 'Sony'

No pre-processing
Resampling: Bootstrapped (30 reps)
Summary of sample sizes: 6237, 6236, 6236, 6236, 6237, 6236, ...
Resampling results across tuning parameters:

  eta  max_depth  colsample_bytree  subsample  nrounds
ROC        Sens       Spec
  0.3  1          0.6               0.50        50
0.7863588  0.6605622  0.7683386
  0.3  1          0.6               0.50       100
0.7840224  0.6479779  0.7746832
  0.3  1          0.6               0.50       150
0.7836610  0.6413626  0.7787061
  0.3  1          0.6               0.75        50
0.7866807  0.6548462  0.7700387
  0.3  1          0.6               0.75       100
0.7843845  0.6441602  0.7756084
  0.3  1          0.6               0.75       150
0.7828528  0.6391983  0.7796325
  0.3  1          0.6               1.00        50
0.7878786  0.6511523  0.7747605
  0.3  1          0.6               1.00       100
0.7863602  0.6413588  0.7793246
  0.3  1          0.6               1.00       150
0.7858035  0.6385617  0.7796340
  0.3  1          0.8               0.50        50
0.7853186  0.6561107  0.7701171
  0.3  1          0.8               0.50       100
0.7844160  0.6402142  0.7772348
  0.3  1          0.8               0.50       150
0.7826027  0.6367776  0.7787827
  0.3  1          0.8               0.75        50
0.7888752  0.6515378  0.7728249
  0.3  1          0.8               0.75       100
0.7839652  0.6413588  0.7766158
  0.3  1          0.8               0.75       150
0.7827488  0.6364013  0.7796331
  0.3  1          0.8               1.00        50
0.7902034  0.6502603  0.7775442
```

Listing 3: Random Forest

```
> summary(model_rf)

Call:
(function (x, y, trials = 1, rules = FALSE, weights = NULL, control = C5.0C
funcCall
 118811.6, 97162.92, 41801.08, 83385.8, 108972.8, 50444.96, 82882.6, 49168.
 129376.1, 101090.2, 129777.1, 45991.51, 63077.13, 36877.89, 37634.


C5.0 [Release 2.07 GPL Edition]         Fri Aug  2 14:05:09 2019
-------------------------------

Class specified by attribute `outcome'

Read 6929 cases (38 attributes) from undefined.data

-----  Trial 0:  -----

Decision tree:

salary > 123755.3: Sony (1448/37)
salary <= 123755.3:
:...salary <= 47688.71:
    :...age <= 59: Sony (1006/26)
    :   age > 59:
    :   :...salary > 30273.7: Acer (292/8)
    :       salary <= 30273.7:
    :       :...salary <= 20241.24: Sony (27/1)
    :           salary > 20241.24: Acer (173/81)
    salary > 47688.71:
    :...age > 60:
        :...salary <= 76556.47: Acer (445/42)
        :   salary > 76556.47: Sony (841/28)
        age <= 60:
        :...salary <= 68615.58:
            :...age <= 40:
            :   :...salary <= 51545.83: Sony (64/29)
            :   :   salary > 51545.83: Acer (316/25)
            :   age > 40:
            :   :...age <= 57: Sony (318/6)
            :       age > 57: Acer (51/21)
            salary > 68615.58:
            :...salary > 98981.91:
                :...age <= 38: Sony (380/36)
                :   age > 38: Acer (473/63)
                salary <= 98981.91:
```