Assignment 2 Data Analytics & Communication

Micky Labreche s4021290 & Yann Amado S5091128

30-11-2022

1 Logistic regression

(1) First we load in the data and make sure that some variables are interpreted as factors. Then we put the data from subject 8 into its own dataset.

```
dat <- read.table("./decision.dat",header=T)

dat$cohFac <- as.factor(dat$cohFac)
dat$isDots <- as.factor(dat$isDots)

is.factor(dat$cohFac)
is.factor(dat$isDots)

subj8 <- dat[dat$subjNo==8,]
```

(2) Now, we start with a logistic regression model for subject 8. We relate the `ER` variable to `RT`, `blocknum`, `isDots`, `cohFac`, and `isLeft`.

```
mod <- glm(ER ~ RT + isDots + isLeft + cohFac + blocknum,data=subj8,
           family=binomial(link="logit"))

# Getting coefficients from model
summary(mod)$coefficients
```

```
##                 Estimate  Std. Error     z value     Pr(>|z|)
## (Intercept) -3.833061201 0.254504768 -15.0608620 2.929422e-51
## RT           0.108133050 0.086488232   1.2502632 2.112034e-01
## isDots1      2.418577049 0.224201894  10.7874960 3.943888e-27
## isLeft       0.594413179 0.143072206   4.1546377 3.258034e-05
## cohFac1     -1.237906247 0.154742929  -7.9997597 1.246623e-15
## blocknum     0.001504718 0.003881553   0.3876588 6.982686e-01
```

```
# Turn the regression coefficients into odds
exp(cbind(Odds_Ratio=coef(mod),confint(mod)))
```

```
## Waiting for profiling to be done...
##                Odds_Ratio      2.5 %      97.5 %
## (Intercept)  0.02164326 0.01289038  0.03503349
## RT           1.11419598 0.93727080  1.31748825
## isDots1     11.22986839 7.36197421 17.78685639
## isLeft       1.81196733 1.37155795  2.40446272
## cohFac1      0.28999075 0.21316658  0.39125536
## blocknum     1.00150585 0.99391442  1.00916537
```

```
# Performing a chi-square test on the model
anova(mod,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: ER
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                      2547     1692.4
## RT        1   88.634      2546     1603.7 < 2.2e-16 ***
## isDots    1  145.421      2545     1458.3 < 2.2e-16 ***
## isLeft    1   18.485      2544     1439.8 1.712e-05 ***
## cohFac    1   69.106      2543     1370.7 < 2.2e-16 ***
## blocknum  1    0.150      2542     1370.6    0.6982
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
prob_change <- function(OR, base) {
  prob <- OR*base/(1 + OR*base - base)
  return(prob)
}

prob_change(1.11419598, mean(subj8$ER))
```

```
## [1] 0.1136655
```

```
prob_change(11.22986839, mean(subj8$ER))
```

```
## [1] 0.5638027
```

```
prob_change(1.81196733, mean(subj8$ER))
```

```
## [1] 0.1725654
```

```
prob_change(0.28999075, mean(subj8$ER))
```

```
## [1] 0.03229942
```

We ask if we can predict whether the participant answered a trial correctly or incorrectly. We call this dependent variable, the error rate. We look at different possible variables that can influence the error rate using a logistic regression. A chi-square test was performed on the model to find if variables independently influenced the error rate.

We found that the relation between the response time and the error rate was significant (Xˆ2(1)=89, p<0.01). The odds ratio corresponding to this relation is 1.11. The baseline probability of answering a trial incorrectly is 0.103. We find that this probability increases to 0.114 with a longer response time, which is a probability increase of 0.010.

We then looked at the relation between the condition type and the error rate and found that this was significant (Xˆ2(1)=145, p<0.01). In the dots condition the subject had a higher error rate with an odds ratio of 11.22. This means that the probability in the dots condition of getting a trial incorrect is 0.564, which is a very large increase compared to other variables.

We also found that the relation between whether the dots were moving left or right had a significant effect on the error rate (Xˆ2(1)=18, p<0.01). The odds ratio for this relation is 1.81. This means the left moving dots increased the error rate from 0.103 to 0.173.

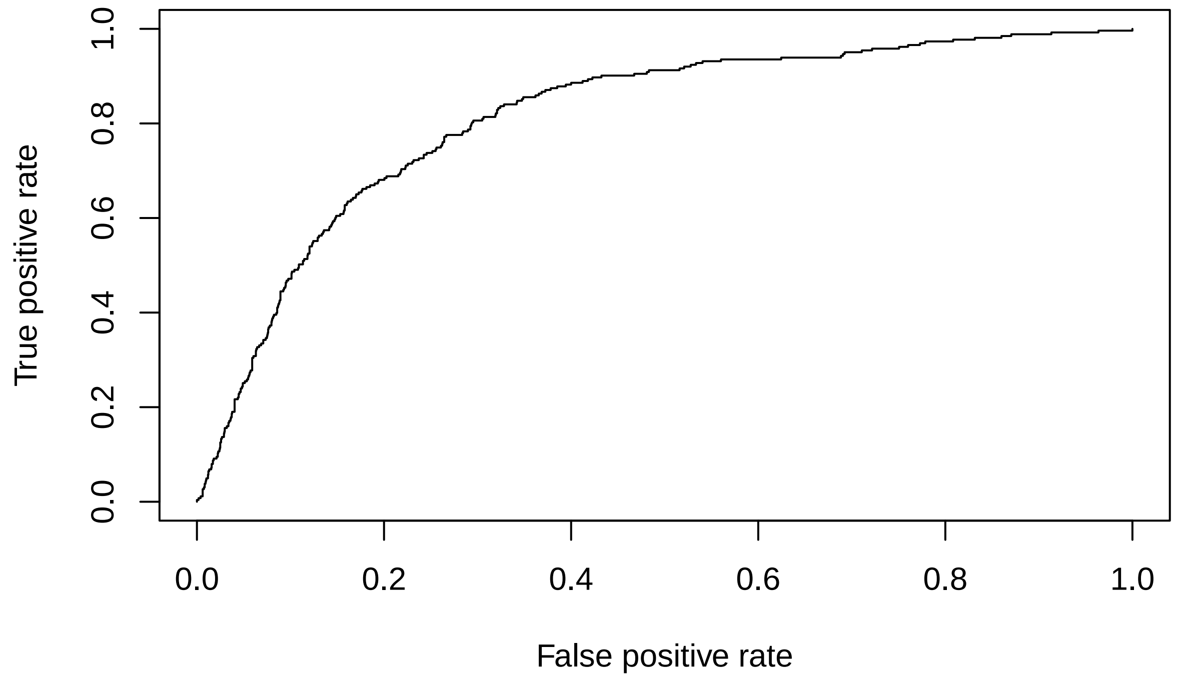
Lastly we saw that the relation between the task difficulty and the error rate was significant (Xˆ2(1)=69, p<0.01). The odds ratio corresponding to this relation is 0.29. This means that an easy task lead to a decrease in error rate. The probability of an error is now 0.032 with an easy task.

2 Model assessment

Now we try to examine the quality of the fit with an ROC using the code below.

```r
library(ROCR)

# Plot ROC graph
p <- predict(mod,type="response")
pr <- prediction(p,subj8$ER)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```



We can see from the ROC graph that the model performs well. If the graph would show up as a straight line, the model would be no better than chance. The distributions in this case are further apart.

We can look at the area under the curve to find if there is a good discrimination.

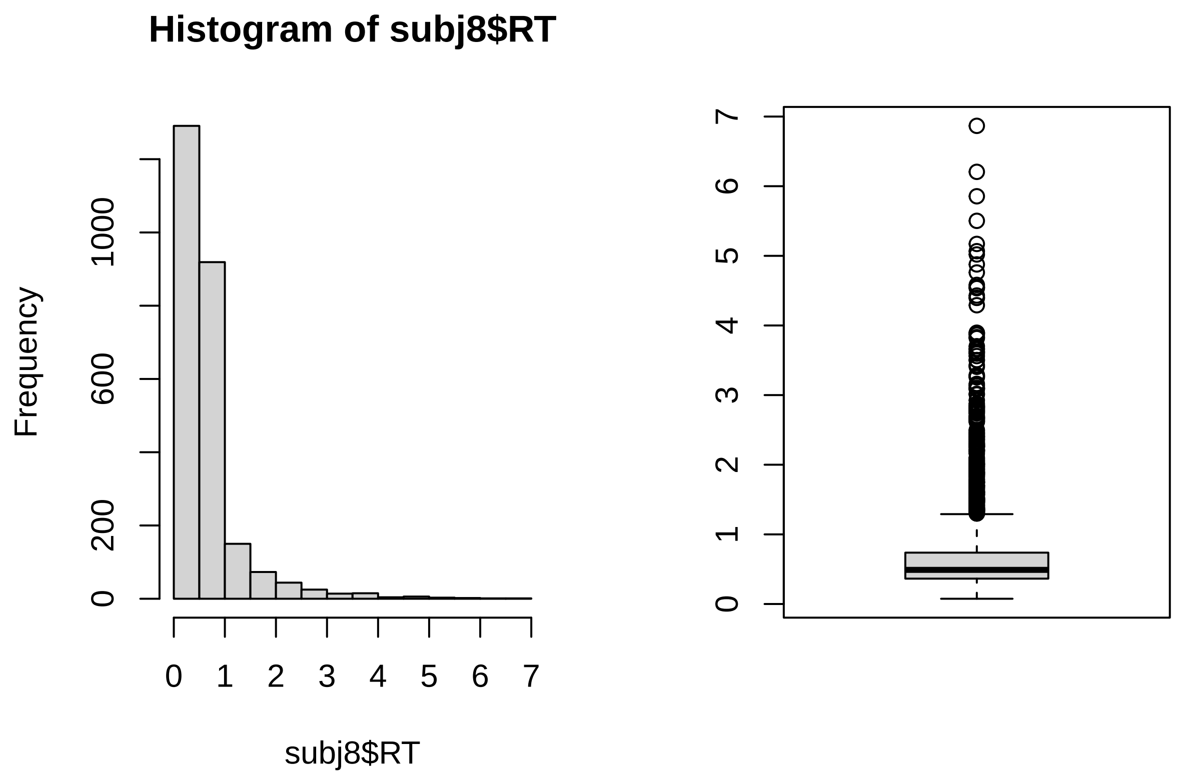
```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.8132614
```

In this case the auc is 0.81, which is a little bit higher than 0.8 which indicates a good discrimination.

(b) Now we look at the outliers in the data. The only variables that are continuous and can have outliers are the response time and the block. It seems that the response time has a lot of outliers.

```
par(mfrow=c(1,2))
hist(subj8$RT)
boxplot(subj8$RT)
```



We can observe that the response time has a lot of outliers. The histogram on the left shows that the amount of outliers is relatively small compared to the data inside the IQR. However some of the values are quite far from the IQR.

```
library("Hmisc")
```

```
## Carregando pacotes exigidos: lattice

## Carregando pacotes exigidos: survival

## Carregando pacotes exigidos: Formula

## Carregando pacotes exigidos: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
data <- subj8[, c(3,4,5,6,7)]
rcorr(as.matrix(data))
```

```
##              RT isLeft cohVec cohFac blocknum
## RT         1.00   0.04  -0.18  -0.18     0.07
## isLeft     0.04   1.00   0.00   0.00     0.00
## cohVec    -0.18   0.00   1.00   1.00     0.02
## cohFac    -0.18   0.00   1.00   1.00     0.02
## blocknum  0.07   0.00   0.02   0.02     1.00
##
## n= 2548
##
##
## P
##          RT     isLeft cohVec cohFac blocknum
## RT              0.0766 0.0000 0.0000 0.0005
## isLeft   0.0766        0.8083 0.8083 0.8556
## cohVec   0.0000 0.8083        0.0000 0.3261
## cohFac   0.0000 0.8083 0.0000        0.3261
## blocknum 0.0005 0.8556 0.3261 0.3261
```

3 Model comparison

It is important that the model fits the data well. A model that is too simple will cause underfitting and a model that is too complex leads to overfitting. In this exercise we start with a full model that incorporates all the variables. By selecting a subset of variables that explains the data best, we try to reduce any bias and variance in the data.

(a) Use stepAIC to investigate what variables in this regression model are crucial and which ones may be left out. Report on the results. You can use code like the following:

```r
library("MASS")
mod.step<-stepAIC(mod,trace=3)
```

```
## Start:  AIC=1382.57
## ER ~ RT + isDots + isLeft + cohFac + blocknum

## trying - RT

## trying - isDots

## trying - isLeft

## trying - cohFac

## trying - blocknum

##            Df Deviance    AIC
## - blocknum  1   1370.7 1380.7
## - RT        1   1372.1 1382.1
## <none>          1370.6 1382.6
## - isLeft    1   1388.3 1398.3
## - cohFac    1   1439.8 1449.8
## - isDots    1   1538.9 1548.9
##
## Step:  AIC=1380.72
## ER ~ RT + isDots + isLeft + cohFac

## trying - RT

## trying - isDots

## trying - isLeft

## trying - cohFac

##          Df Deviance    AIC
## - RT      1   1372.5 1380.5
## <none>        1370.7 1380.7
## - isLeft  1   1388.4 1396.4
## - cohFac  1   1439.8 1447.8
## - isDots  1   1539.1 1547.1
##
## Step:  AIC=1380.46
## ER ~ isDots + isLeft + cohFac

## trying - isDots

## trying - isLeft

## trying - cohFac

##          Df Deviance    AIC
## <none>        1372.5 1380.5
## - isLeft  1   1390.2 1396.2
## - cohFac  1   1454.4 1460.4
## - isDots  1   1600.5 1606.5
```

```
mod.step$anova
```

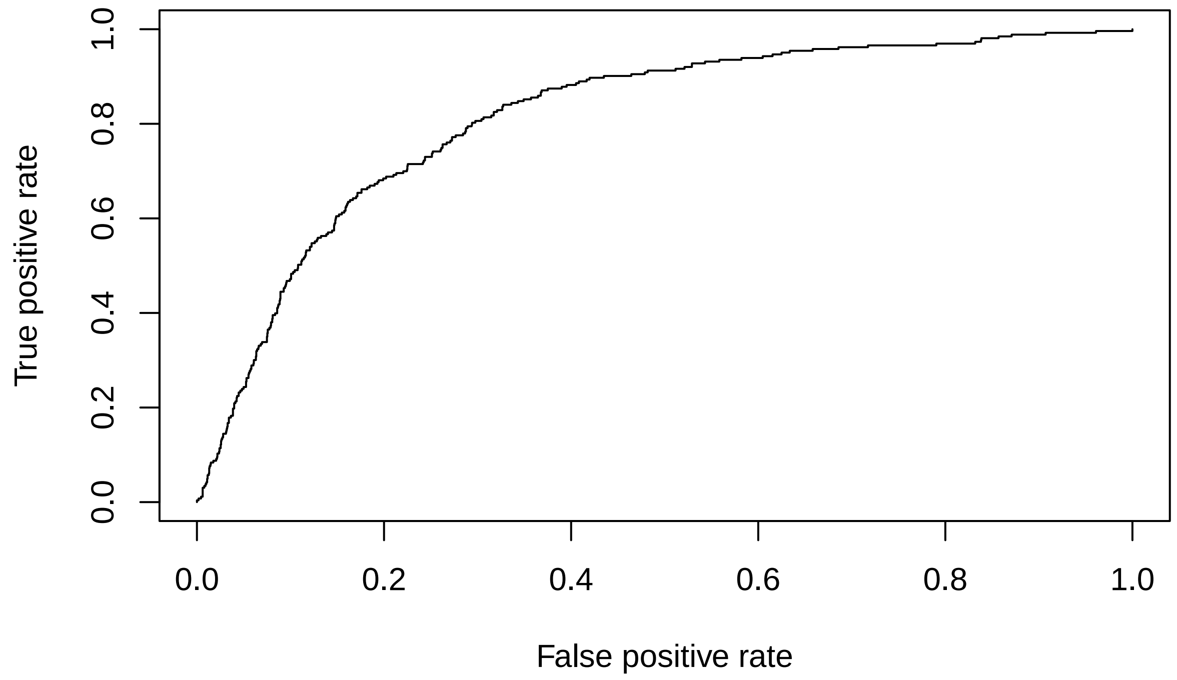
```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## ER ~ RT + isDots + isLeft + cohFac + blocknum
##
## Final Model:
## ER ~ isDots + isLeft + cohFac
##
##
##         Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                              2542   1370.570 1382.570
## 2 - blocknum  1 0.1503048      2543   1370.721 1380.721
## 3       - RT  1 1.7356969      2544   1372.456 1380.456
```

According to the results, the variables blocknum and RT can be left out, with the remaining variables isDots, isLeft and cohFac being crucial to the model. It's possible to verify this by analyzing the steps of AIC, initially it's AIC score was 1382.570, diminishing to 1380.721 after removing blocknum and to 1380.456 after removing RT; in other words, according to AIC, the model is able to achieve similar results even if those variables are removed.

(b) Make a more complicated model by adding an interaction between coherence level and task type (i.e., whether it is dots or control task). Fit this more complicated model and describe your results. Describe in your results what this interaction actually means.

Fitting the model and measuring the AUC:

```
mod <- glm(ER ~ RT + isDots + cohFac + isDots * cohFac + isLeft + blocknum,data=subj8, family=binomial(l
library(ROCR)
p <- predict(mod,type="response")
pr <- prediction(p,subj8$ER)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```



```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
```

At first glance, it doesn't look like making this additional interaction makes that much of a difference. Upon further inspection of the AUC values, it's possible to observe that this additional interaction yielded an AUC value of 0.814, increased from 0.813. Although there can be said that there's a slight improvement, it's not possible to say if it's worth the additional complexity of the model. This additional interaction describes a relationship between the variables isDots and cohFac, trying to relate both variables between each other. On a high level, we hope that this interaction will help the model to identify interactions between the nivel of difficulty with the type of trial the participant is partaking; on a low level, this interaction is a linear combination of the features that was related in order to further improve the accuracy of the model, a process known as feature extraction.

(c) Run stepAIC again on the extended model and describe your findings. Was the interaction a good addition to the model? Explain your answer.

```r
mod.step<-stepAIC(mod,trace=3)
```

```
## Start:  AIC=1378
## ER ~ RT + isDots + cohFac + isDots * cohFac + isLeft + blocknum

## trying - RT

## trying - isLeft

## trying - blocknum

## trying - isDots:cohFac

##                  Df Deviance    AIC
## - blocknum        1   1364.2 1376.2
## - RT              1   1365.0 1377.0
## <none>                1364.0 1378.0
## - isDots:cohFac   1   1370.6 1382.6
## - isLeft          1   1381.3 1393.3
##
## Step:  AIC=1376.19
## ER ~ RT + isDots + cohFac + isLeft + isDots:cohFac

## trying - RT

## trying - isLeft

## trying - isDots:cohFac

##                  Df Deviance    AIC
## - RT              1   1365.3 1375.3
## <none>                1364.2 1376.2
## - isDots:cohFac   1   1370.7 1380.7
## - isLeft          1   1381.5 1391.5
##
## Step:  AIC=1375.33
## ER ~ isDots + cohFac + isLeft + isDots:cohFac

## trying - isLeft
## trying - isDots:cohFac
##                 Df Deviance    AIC
## <none>               1365.3 1375.3
## - isDots:cohFac  1   1372.5 1380.5
## - isLeft         1   1382.7 1390.7
```

```
mod.step$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## ER ~ RT + isDots + cohFac + isDots * cohFac + isLeft + blocknum
##
## Final Model:
## ER ~ isDots + cohFac + isLeft + isDots:cohFac
##
##
##         Step Df Deviance Resid. Df Resid. Dev      AIC
## 1                             2541   1364.002 1378.002
## 2 - blocknum  1 0.192641      2542   1364.195 1376.195
## 3       - RT  1 1.136214      2543   1365.331 1375.331
```

The interaction resulted in a lower AIC score of roughly 4 points at every step, this is a slight improvement from the original model without this interaction. On the AIC model, the same variables blocknum and RT where deemed not significant enough and where discarded much like the more simple model. This slight change can be explained by the fact that this new feature is only a linear combination of two other features present in the data that reveals only a small bit of extra information to the model,

Although this addition resulted in an improved result of both AIC and AUC scores, it's debatable whether or not this interaction is a good addition. From one hand, this extra interaction resulted in better scores, as well as making it possible to reduce the dimensionality of the model by removing both isDots and cohFac from the model; however, on the other hand, the results on this particular example don't seem to be significant enough to warrant the extra layer of complexity for this particular model.

In the R implementation of AIC, if both isDots and cohFac are removed and changed for the new feature, the AIC algorithm will actually return that the most optimal model will also include the variables isDots and cohFac individually, as seen below. For this reason, we believe that it's not a good enough addition to actually implement in the model

```
mod <- glm(ER ~ RT + isDots * cohFac + isLeft + blocknum,data=subj8, family=binomial(link="logit"))
mod.step<-stepAIC(mod,trace=3)
```

```
## Start:  AIC=1378
## ER ~ RT + isDots * cohFac + isLeft + blocknum


## trying - RT


## trying - isLeft


## trying - blocknum


## trying - isDots:cohFac
##                 Df Deviance    AIC
## - blocknum       1   1364.2 1376.2
## - RT             1   1365.0 1377.0
## <none>               1364.0 1378.0
## - isDots:cohFac  1   1370.6 1382.6
## - isLeft         1   1381.3 1393.3
##
## Step:  AIC=1376.19
## ER ~ RT + isDots + cohFac + isLeft + isDots:cohFac

## trying - RT

## trying - isLeft

## trying - isDots:cohFac

##                 Df Deviance    AIC
## - RT             1   1365.3 1375.3
## <none>               1364.2 1376.2
## - isDots:cohFac  1   1370.7 1380.7
## - isLeft         1   1381.5 1391.5
##
## Step:  AIC=1375.33
## ER ~ isDots + cohFac + isLeft + isDots:cohFac

## trying - isLeft
## trying - isDots:cohFac

##                 Df Deviance    AIC
## <none>               1365.3 1375.3
## - isDots:cohFac  1   1372.5 1380.5
## - isLeft         1   1382.7 1390.7
```

```
mod.step$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## ER ~ RT + isDots * cohFac + isLeft + blocknum
##
## Final Model:
## ER ~ isDots + cohFac + isLeft + isDots:cohFac
##
##
##         Step Df Deviance Resid. Df Resid. Dev      AIC
## 1                             2541   1364.002 1378.002
## 2 - blocknum  1 0.192641      2542   1364.195 1376.195
## 3       - RT  1 1.136214      2543   1365.331 1375.331
```

4 Extending this to algorithms more generally

(a) Say, you are working on creating a classifier to detect humans in satellite images. Can you use AIC in this case? If so, why? If not, why not? What other methods could you use to compare different versions of your classifier? What model comparison methods are good for what kinds of models?

Yes, it's possible to use. In order to use AIC 3 assumptions must be met:

- The same data is used between models.
- The models must measure the same outcome variable.
- The dataset must have a sufficiently large sample size.

Since our focus is to compare different versions of the same classifier, AIC will prove itself useful by comparing the models between themselves while taking into consideration the complexity of the model. Other methods that could be used would be goodness of fit, BIC ,ROC curve and PSP.

- Goodness of fit: Verify how much the model fits the data correctly, can be used in the same dataset accross diferent models to verify which one most closely fits the data. It's good to measure similarly complex models, which in this case it'll measure which model has the highest accuracy in the dataset.
- BIC (Bayesian Information Criterion): Closely related to AIC, can be used in the same instances as AIC, utilizes a Bayesian approach to tackle the problem. Takes into consideration the complexity of the models, making it possible to compare models that have different complexities.
- ROC Curve: It's a graphical plot that illustrates the performance of a model on a given dataset when several classification threshold are utilized. It's good to compare models that tradeoff specifity and sensitivity, specially when one is favored over the other.
- PSP (Parameter-Space-Partitioning): Compares models based on the parameter space they cover, taking into consideration how much space each parameter takes, taking into consideration the flexibility of the model. It's good for for comparing the decision processes of the model, making it possible to choose the one most adequate to the task taking into account the complexity of the model.

In these kinds of large computational projects, you always have to worry about overfitting. One of the readings for this week talks about the danger of overfitting. As we learnt in the reading for this week, overfitting is deceptively easy, even when you use cross-validation.

(b) How can overfitting still occur, even when you use cross-validation?

If cross validation is repeatedly used in the same dataset, it's possible that information leaks into the model. For example, when using cross-validation to tune hyperparameters, if the same dataset is repeatedly used, cross validation will yield results in relation to that dataset, and not in relation to new unforeseen data like it should theoretically do. Meaning that, the model will have the best hyperparameters to that particular dataset, not generalizing well into other datasets, causing overfitting.

(c) How is the lockbox method (also known as train-test/validation) different from cross-validation, and why is it effective to prevent overfitting?

The lockbox method involves "locking away" one part of the dataset as a test set, which will only be used at the vary last moment to simulate the behaviour of that model on new unforeseen data, giving an estimate of how well it'll generalize without any bias involved. The lockbox method by itself does not prevent overfitting but it gives a metric of how much overfitting has occurred.

(d) What other methods could you use to prevent overfitting? For each method you mention, briefly explain how it helps to prevent overfitting.

- Pre-registration: In pre-registration, the team pre registers it's plan to an easily accessable journal, encouraging the researchers to think thoroughly about their method and hyperparameters tests. It helps prevent overfitting by not letting the original study deviate as much from the original plan, meaning that even if new information about the dataset is obtained through the researchers interaction, the pre registered plan won't allow those insights to be used, preventing overfitting.
- Nested cross-validation: In nested cross-validation, inner and outer rounds of cross-validation are applied, each evaluating different sets of hyperparameters and making it possible to learn different parameters without information leakage. It prevents overfitting by essentially applying the lock box method several times, each for one configuration of hyperparameters, allowing the researcher to search for different hyperparameters without information leaking to the model, preventing overfitting.
- Blind analysis: In blind analysis, the model learn it's parameters or hyperparameters by analysing data that is orthogonal to it's main objective or by removing the labels of the data, meaning that it won't learn any meaningful insights about the actual data it'll be tested on. It can be seen as a regularization technique, such as dropout for neural networks or L2 for machine learning in general, and it prevents overfitting by increasing the bias, and thus, reducing variability.

5 Contributions

Questions 1 to 2 -> Micky

Questions 3 to 4 -> Yann