# 강의록 코드 구현 불가능 정리

1. R의 predict 함수의 method 부분을 구현 못 함.
    a. method = "predictive"
    b. method = "debiased"

R

```
predplot(cush.lda, "LDA")
predplot(cush.qda, "QDA")
predplot(cush.qda, "QDA (predictive)", method = "predictive")
predplot(cush.qda, "QDA (debiased)", method = "debiased")
```

R

python

```
predplot(cush_lda,1, "LDA")
predplot(cush_qda,3, "QDA")

#R의 predict 함수의 method 부분을 구현 못 함. 이 부분 다시 확인.
#predplot(cush_qda,2, "QDA (predictive)", method = "pre
#predplot(cush_qda,4, "QDA (debiased)", method = "debia
```

python

---

2. LDA 함수의 method="t"를 구현 못 함.

R

```
fgl.ld <- predict(lda(type ~ ., fgl), dimen = 2)$x

fgl.rld <- predict(lda(type ~ ., fgl, method = "t"), dimen = 2)$x
```

R

python

```
std_fgl = StandardScaler().fit_transform(fgl.drop(["typ

#$x LDA 함수의 method="t"를 구현해서 한 번 더 그림.
fgl_ld=LDA().fit(std_fgl,fgl_target).transform(std_fgl)
fgl_ld[:,0]=-fgl_ld[:,0]
fgl_ld=pd.DataFrame(data=fgl_ld,columns=lda_columns)
```

python

---

3. 여기 값이 다르게 나옴. 아마 hidden layer size 때문인 것 같음. 재확인 필요.

R

```
Z <- matrix(0, nrow(cushT), ncol(tpi))
#print( matrix(0, nrow(cushT), ncol(tpi)))
for(iter in 1:20) {
   set.seed(iter)
   cush.nn <- nnet(cush, tpi, skip = TRUE, softmax = TRUE, size = 8,
       decay = 0.01, maxit = 1000, trace = FALSE)
   print(cush.nn$value)
   Z <- Z + predict(cush.nn, cushT)
   ?nnet
   cat("final value", format(round(cush.nn$value,3)), "\n")
   b1(predict(cush.nn, cushT), col = 2, lwd = 0.5)
}
pltnn("Averaged")
b1(Z, lwd = 3)
```

R

python

```
Z = np.zeros( (cushT.shape[0], tpi.shape[1]))


for iter in range(0,20) :
#    np.random.seed(iter)
#    cush_nn = nnet(cush, tpi, skip = TRUE, softmax = T
#       decay = 0.01, maxit = 1000, trace = FALSE)
    cush_nn= MLPClassifier(hidden_layer_sizes=(27),
                           activation='logistic',
                           solver='sgd',
                           max_iter=1000,learning_rat
    Z = Z + cush_nn.fit(cush,tpi).predict_proba( cushT)
#    print("final value", f"{cush_nn.coefs_}", "\n")
    b1(cush_nn.predict(cushT), col = 'k')
```

python

4. tr$cptable 구현을 못 함.

R

```
library(rpart)
res.rpart <- CVtest(
  function(x, ...) {
    tr <- rpart(type ~ ., fgl[x,], ...)
    cp <- tr$cptable
    r <- cp[, 4] + cp[, 5]
    rmin <- min(seq(along = r)[cp[, 4] < min(r)])
    cp0 <- cp[rmin, 1]
    cat("size chosen was", cp[rmin, 2] + 1, "\n")
    prune(tr, cp = 1.01*cp0)
  },
  function(obj, x)
    predict(obj, fgl[x, ], type = "class"),
  cp = 0.001
)

cptable: a matrix of information on the optimal prunings based on a complexity parameter
```

R

python

```
## !!!!!
def func3(x, *args):

    tr=DecisionTreeClassifier(criterion='entropy',
                              max_depth=3,
                              random_state=0).fit(fgl.i
                                                  fgl.t
    cp = tr.cptable
    r = cp[:, 4] + cp[:, 5]
    rmin = np.min(seq(along = r)[cp.iloc[:, 4] < min(r)
    cp0 = cp[rmin, 1]

    print("size chosen was", cp.iloc[rmin, 2] + 1, "\n"
    prune(tr, cp = 1.01*cp0)

from sklearn.tree import DecisionTreeClassifier_model

res_rpart= CVtest(func3,func2)
```

python

5. olvq1 함수가 파이썬에는 없음. LVQ 함수 오류.

R

```
CV.lvq <- function()
{
    res <- fgl$type
    for(i in sort(unique(rand))) {
        cat("doing fold", i, "\n")
        cd0 <- lvqinit(fgl0[rand != i,], fgl$type[rand != i],
                       prior = rep(1, 6)/6, k = 3)
        cd1 <- olvq1(fgl0[rand != i,], fgl$type[rand != i], cd0)
        cd1 <- lvq3(fgl0[rand != i,], fgl$type[rand != i],
                    cd1, niter = 10000)
        res[rand == i] <- lvqtest(cd1, fgl0[rand == i, ])
    }
    res
}
```

R

python

```
cd0= LVQBaseClass().fit(fgl0, fgl.type)


con(fgl.type, lvqtest(cd0, fgl0))
```

python