2018-11-15 13:20:34

Running some tests to see if data transformations make things look better. I coded a within-subject normalization, and also a within-variable normalization. Let's see if it makes any difference for our best resul in each of the 2 brain modalities. Maybe try different algorithms too. We can try it with fMRI later.

```
job_name=dataTransforms_rawCV;
mydir=/data/NCR_SBRB/baseline_prediction/;
swarm_file=swarm.automl_${job_name};
rm -rf $swarm_file;
for f in struct_volume_11142018_260timeDiff12mo.RData.gz
dti_ad_voxelwise_n223_09212018.RData.gz; do
    for target in nvVSper perVSrem; do
        for pp in subjScale dataScale subjScale,dataScale None; do
            for algo in DeepLearning DRF GBM GLM; do
                for i in {1..100}; do
                    myseed=$RANDOM;
                    echo "Rscript --vanilla
~/research_code/automl/raw_multiDomain_autoValidation_oneAlgo.R
${mydir}/$f ${mydir}/long_clin_0918.csv ${target}
${mydir}/models_raw_dataTransforms/${USER} $myseed $algo $pp" >>
$swarm_file;
                    echo "Rscript --vanilla
~/research_code/automl/raw_multiDomain_autoValidation_oneAlgo.R
${mydir}/$f ${mydir}/long_clin_0918.csv ${target}
${mydir}/models_raw_dataTransforms/${USER} -$myseed $algo $pp" >>
$swarm_file;
                done;
            done;
        done;
    done;
done
sed -i -e "s/^/unset http_proxy; /g" $swarm_file;
split -l 1000 $swarm_file ${job_name}_split;
for f in `/bin/ls ${job_name}_split??`; do
    echo "ERROR" > swarm_wait_${USER}
    while grep -q ERROR swarm_wait_${USER}; do
        echo "Trying $f"
        swarm -f $f -g 40 -t 16 --time 3:00:00 --partition quick --logdir
trash_${job_name} --job-name ${job_name} -m R --gres=lscratch:10 2>
swarm_wait_${USER};
        if grep -q ERROR swarm_wait_${USER}; then
            echo -e "\tError, sleeping..."
            sleep 10m;
        fi;
    done;
done
```

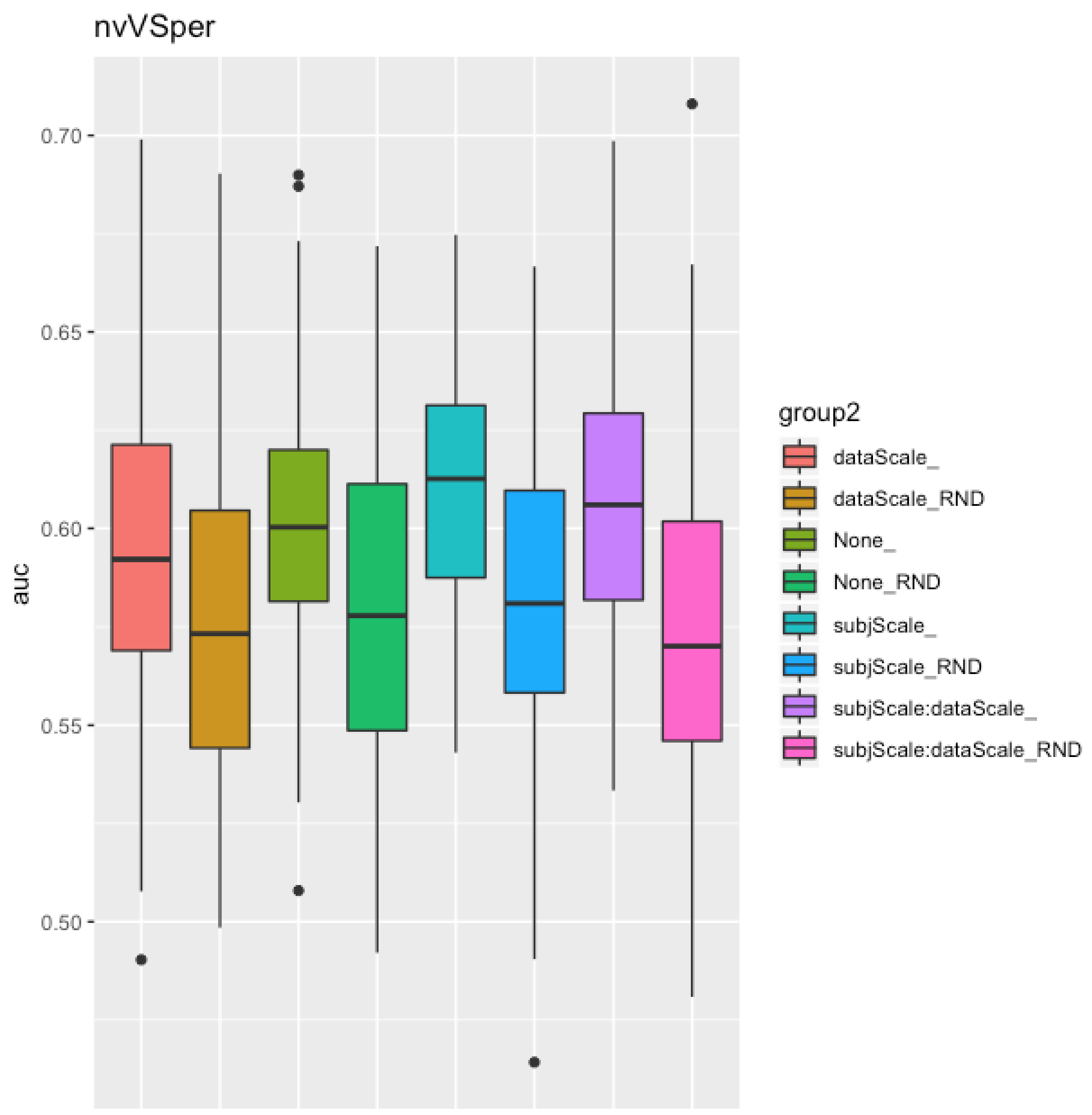
2018-11-16 14:10:03

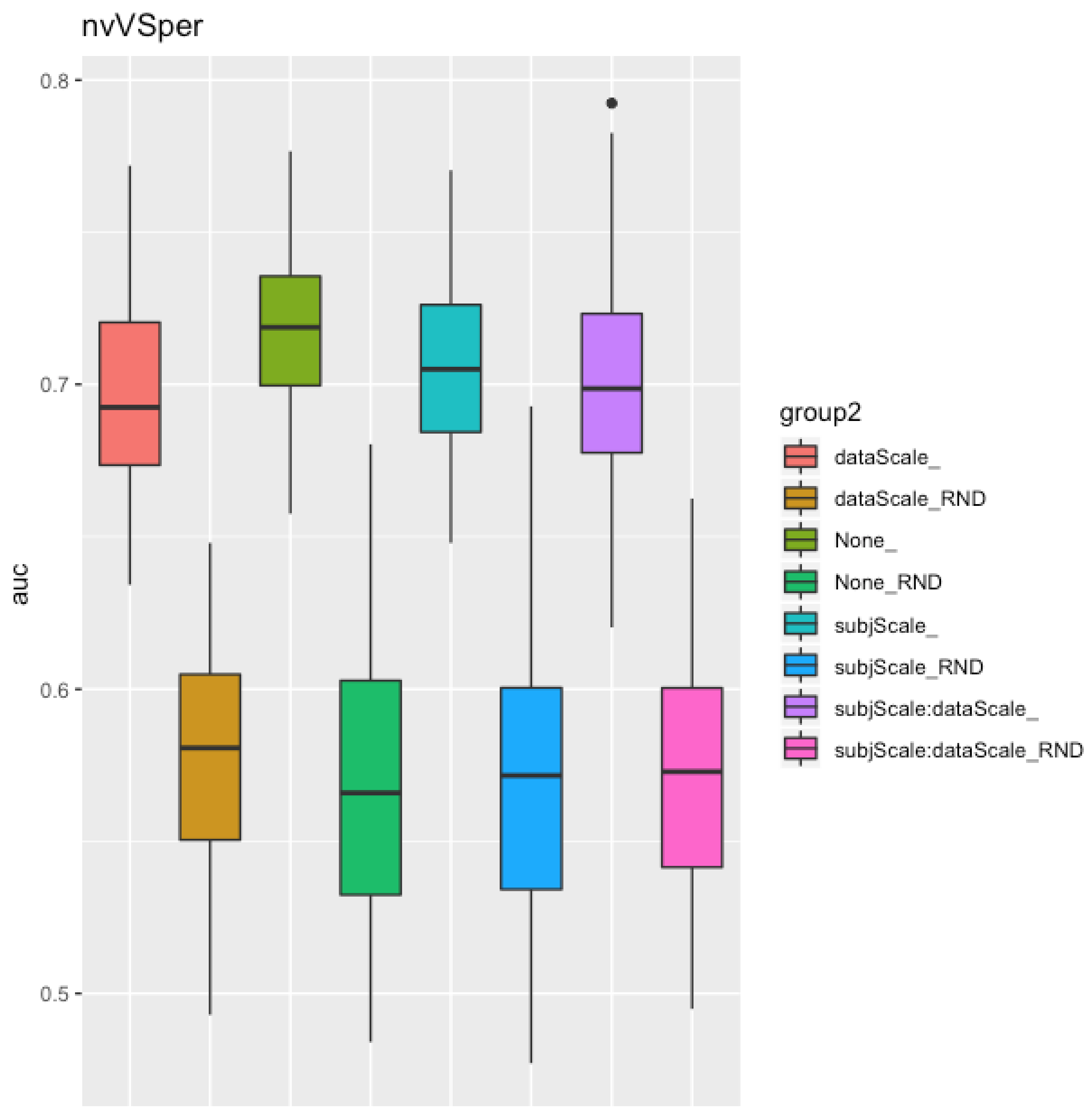
Let's compile the results to see if data transformations make any difference:

```
echo "target,pheno,var,seed,nfeat,model,auc,f1,acc,ratio" >
dataTransforms_summary.csv;
dir=dataTransforms_rawCV;
for f in `ls -1 trash_${dir}/*o`; do
    phen=`head -n 2 $f | tail -1 | awk '{FS=" "; print $7}' | cut -d"/" -f
6`;
    target=`head -n 2 $f | tail -1 | awk '{FS=" "; print $9}'`;
    seed=`head -n 2 $f | tail -1 | awk '{FS=" "; print $11}'`;
    var=`head -n 2 $f | tail -1 | awk '{FS=" "; print $13}'`;
    model=`grep -A 1 model_id $f | tail -1 | awk '{FS=" "; print $2}' |
cut -d"_" -f 1`;
    auc=`grep -A 1 model_id $f | tail -1 | awk '{FS=" "; print $3}'`;
    nfeat=`grep "Running model on" $f | awk '{FS=" "; print $5}'`;
    ratio=`grep -A 1 "Class distribution" $f | tail -1 | awk '{FS=" ";
{for (i=2; i<=NF; i++) printf $i ";"}}'`;
    f1=`grep -A 2 "Maximum Metrics:" $f | tail -1 | awk '{FS=" "; print
$5}'`;
    acc=`grep -A 5 "Maximum Metrics:" $f | tail -1 | awk '{FS=" "; print
$5}'`;
    echo $target,$phen,$var,$seed,$nfeat,$model,$auc,$f1,$acc,$ratio >>
dataTransforms_summary.csv;
done;
```

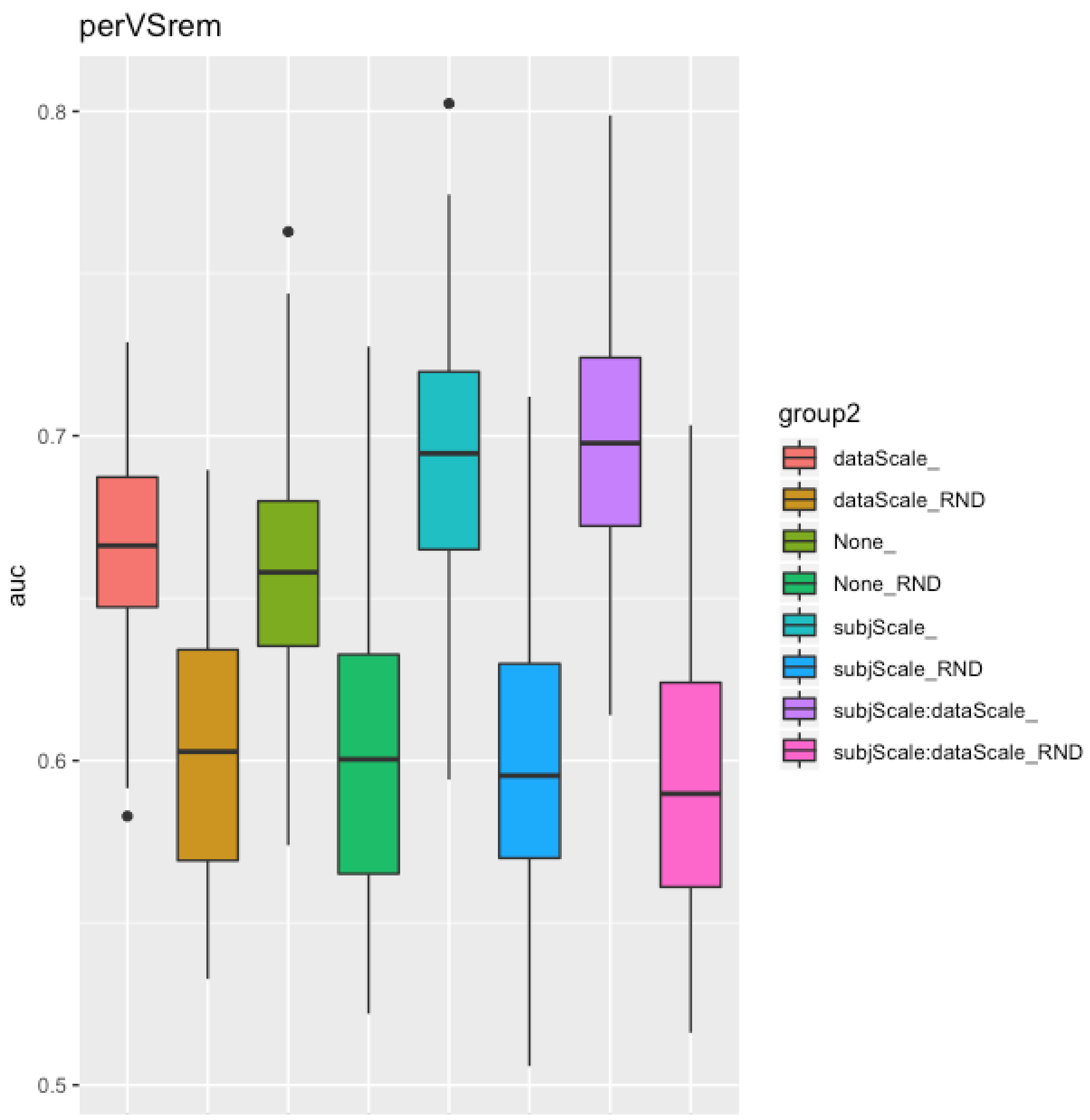
There are many variables here. First, let's stick to AD, and see if any algorithm is particularly sensitive to the 4 different data transforms:

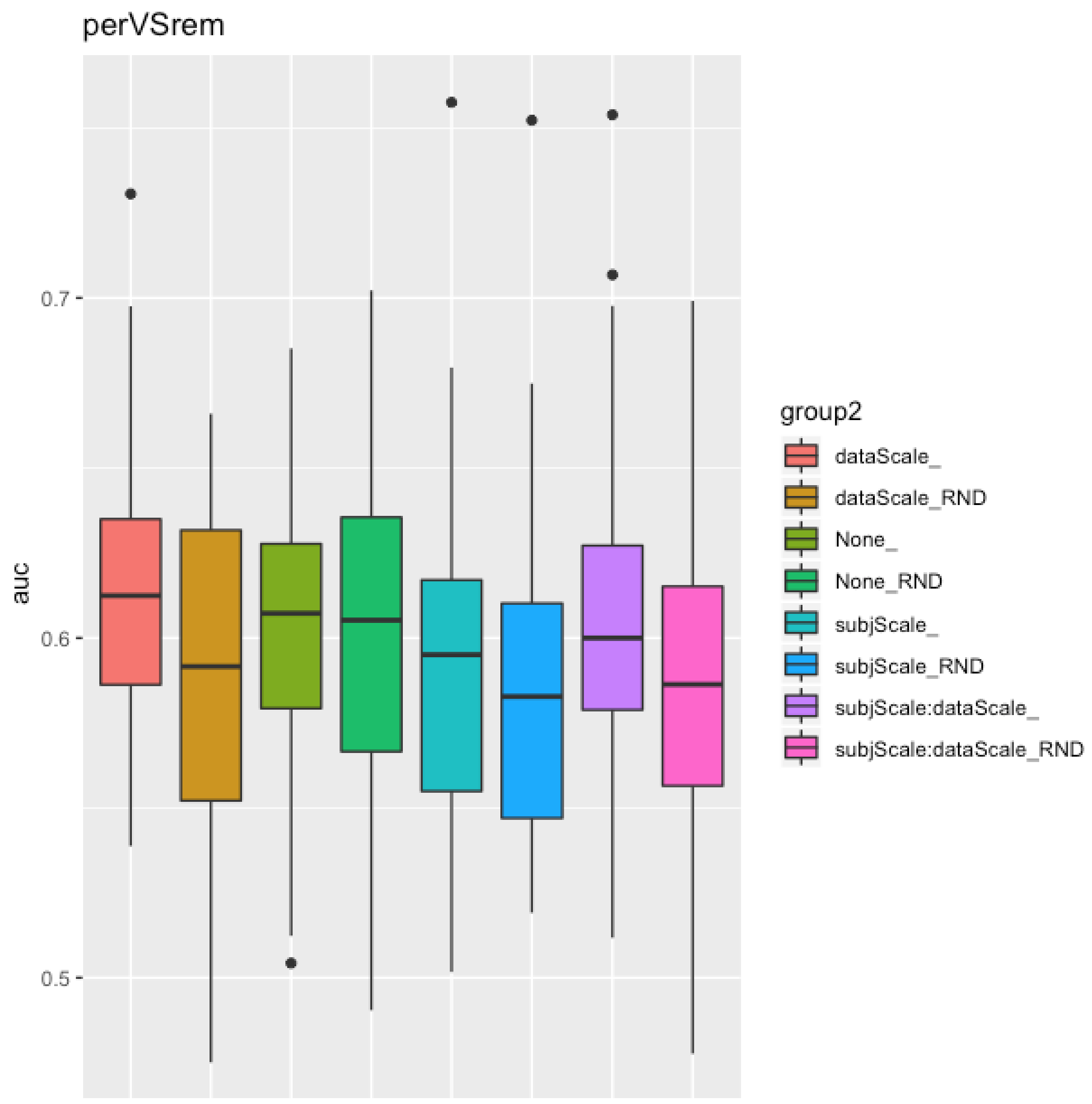
```
data = read.csv('~/tmp/dataTransforms_summary.csv')
data$group = ''
data[data$seed<0,]$group = 'RND'
data$group2 = sapply(1:nrow(data), function(x) { sprintf('%s_%s',
data$var[x], data$group[x])} )
# then, for each target
idx = data$target=='nvVSper' & data$model=='DeepLearning' &
data$pheno=='struct_volume_11142018_260timeDiff12mo.RData.gz'
p1<-ggplot(data[idx,], aes(x=group2, y=auc, fill=group2))
print(p1+geom_boxplot() + ggtitle(unique(data[idx,]$target)))
```


nvVSper
auc
0.70
0.65
0.60
0.55
0.50
group2
dataScale_
dataScale_RND
None_
None_RND
subjScale_
subjScale_RND
subjScale:dataScale_
subjScale:dataScale_RND



It looks like subjScaling has a slight effect on DeepLearning for structural, but not necessarily for DTI.


perVSrem
auc
0.8
0.7
0.6
0.5
group2
dataScale_
dataScale_RND
None_
None_RND
subjScale_
subjScale_RND
subjScale:dataScale_
subjScale:dataScale_RND



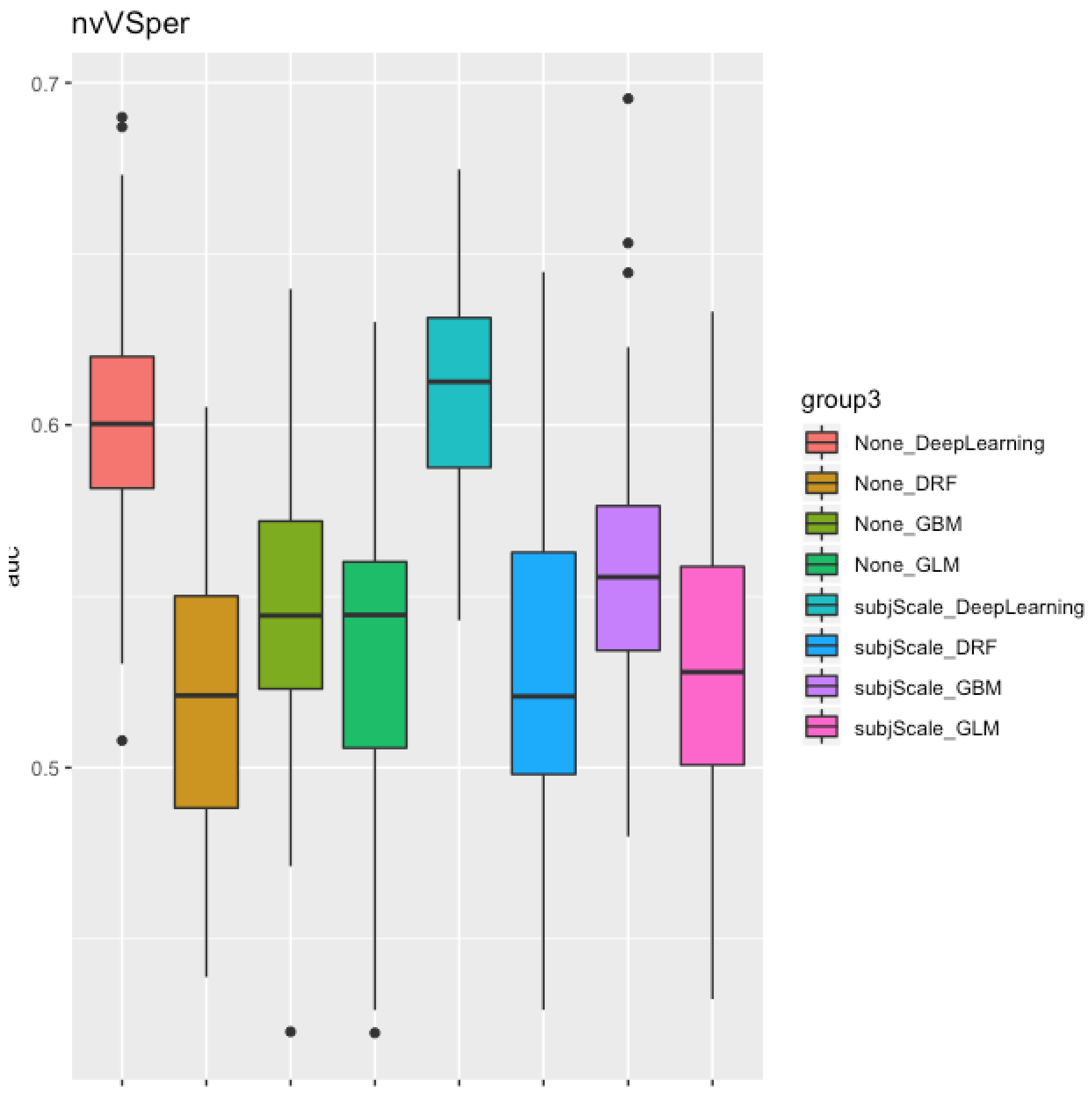
That's particularly obvious in perVSrem. Let's see if other algorithms are as affected. GBM doesn't do as well as DeepLearning, but the best results there are actually with DTI and no transformation. Not that much better than random though. Not much difference in structural.

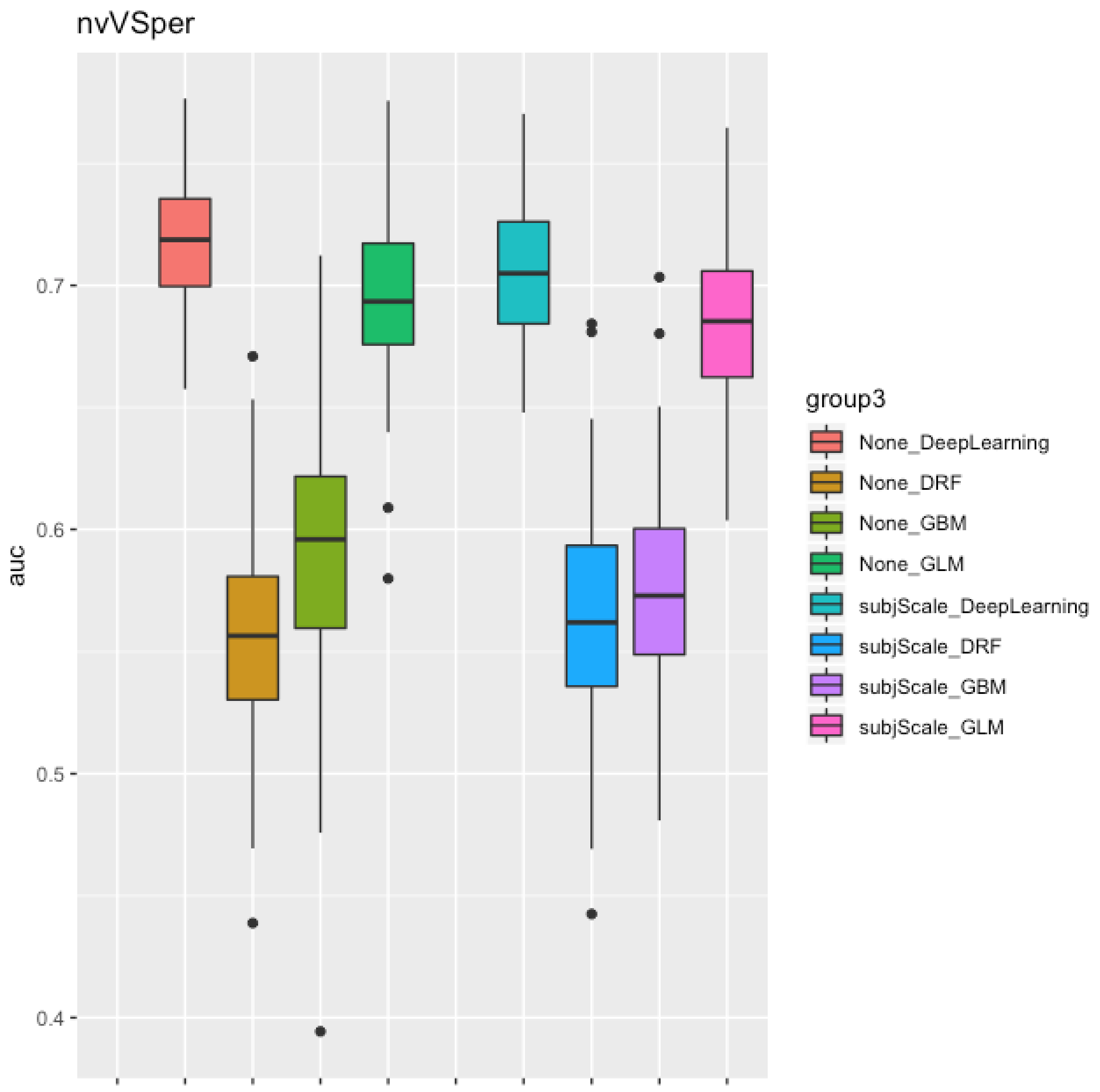
Not much improvement in DRF, which is also not better than deep learning.

The GLM results are actually much better than its random counterpart, but will need to check against other algorithms. subjScale and None are somewhat tied. But for structural, particularly in perVSrem, subjScale trumps the rest. So, it looks like we need to do some subject scaling after all. Let's see how results compare across algorithms. We can do it for None and subjScale just for kicks:

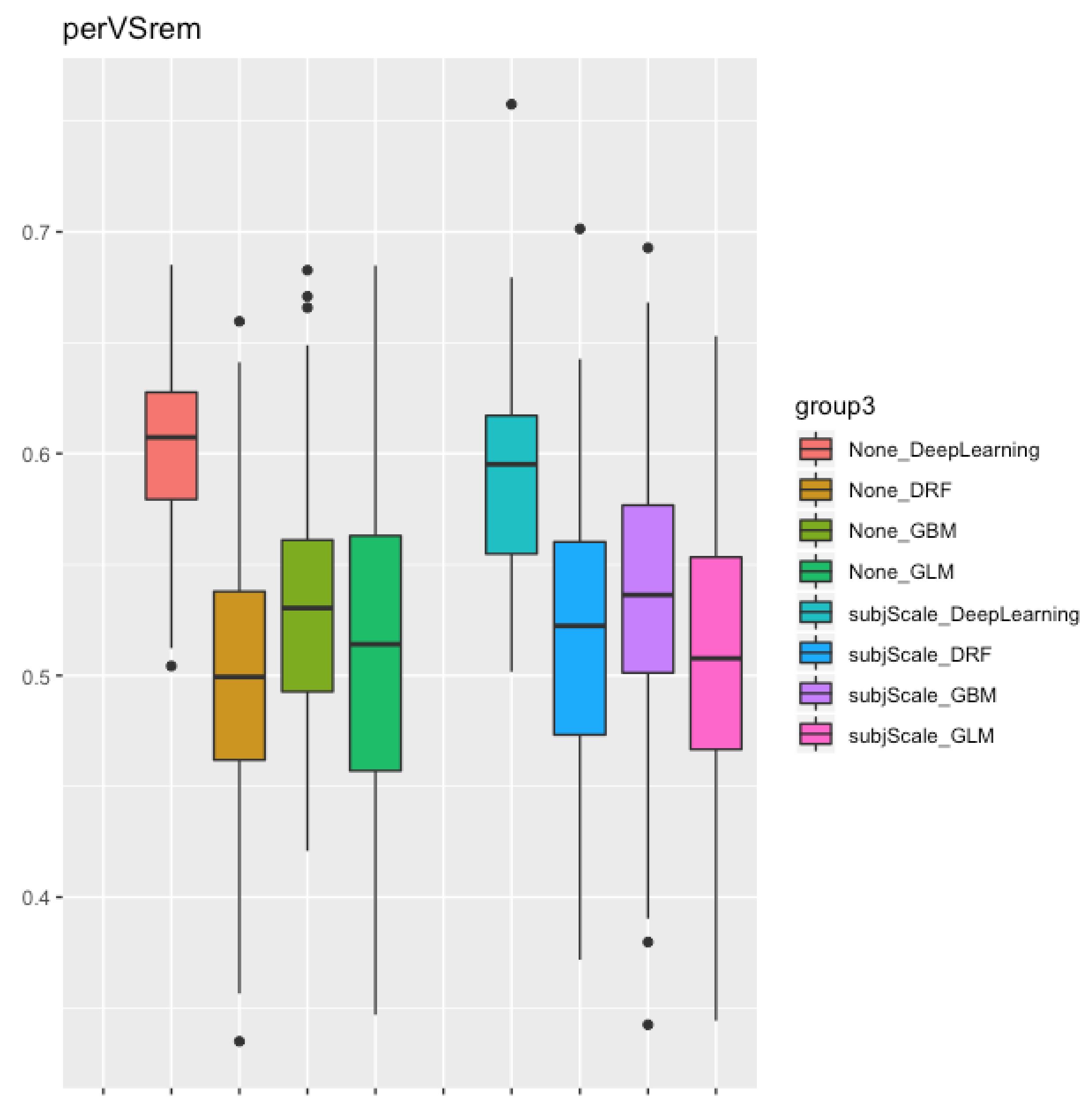
```
data[data$model=='XRT','model'] = 'DRF'
data$group3 = sapply(1:nrow(data), function(x) { sprintf('%s_%s',
data$var[x], data$model[x])} )
idx = data$target=='nvVSper' & (data$var=='None' | data$var=='subjScale')
& data$seed>0 &
data$pheno=='struct_volume_11142018_260timeDiff12mo.RData.gz'
p1<-ggplot(data[idx,], aes(x=group3, y=auc, fill=group3))
print(p1+geom_boxplot() + ggtitle(unique(data[idx,]$target)))
```

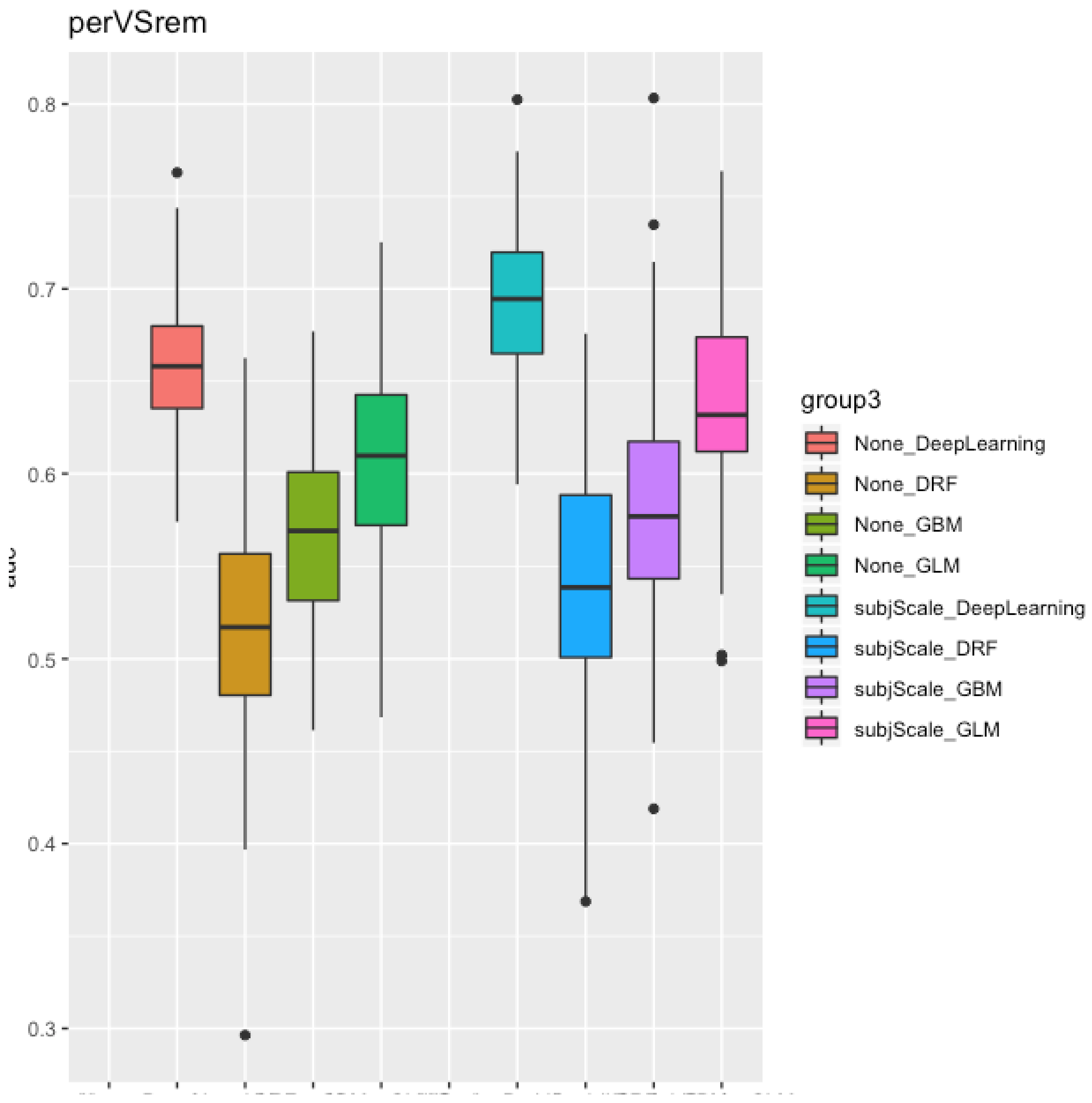
For struct nvVSper, subjScale DL, with None DL as a close second. For DTI it's actually inverted, with None slightly better. Interestingly, GLM is not too bad either.





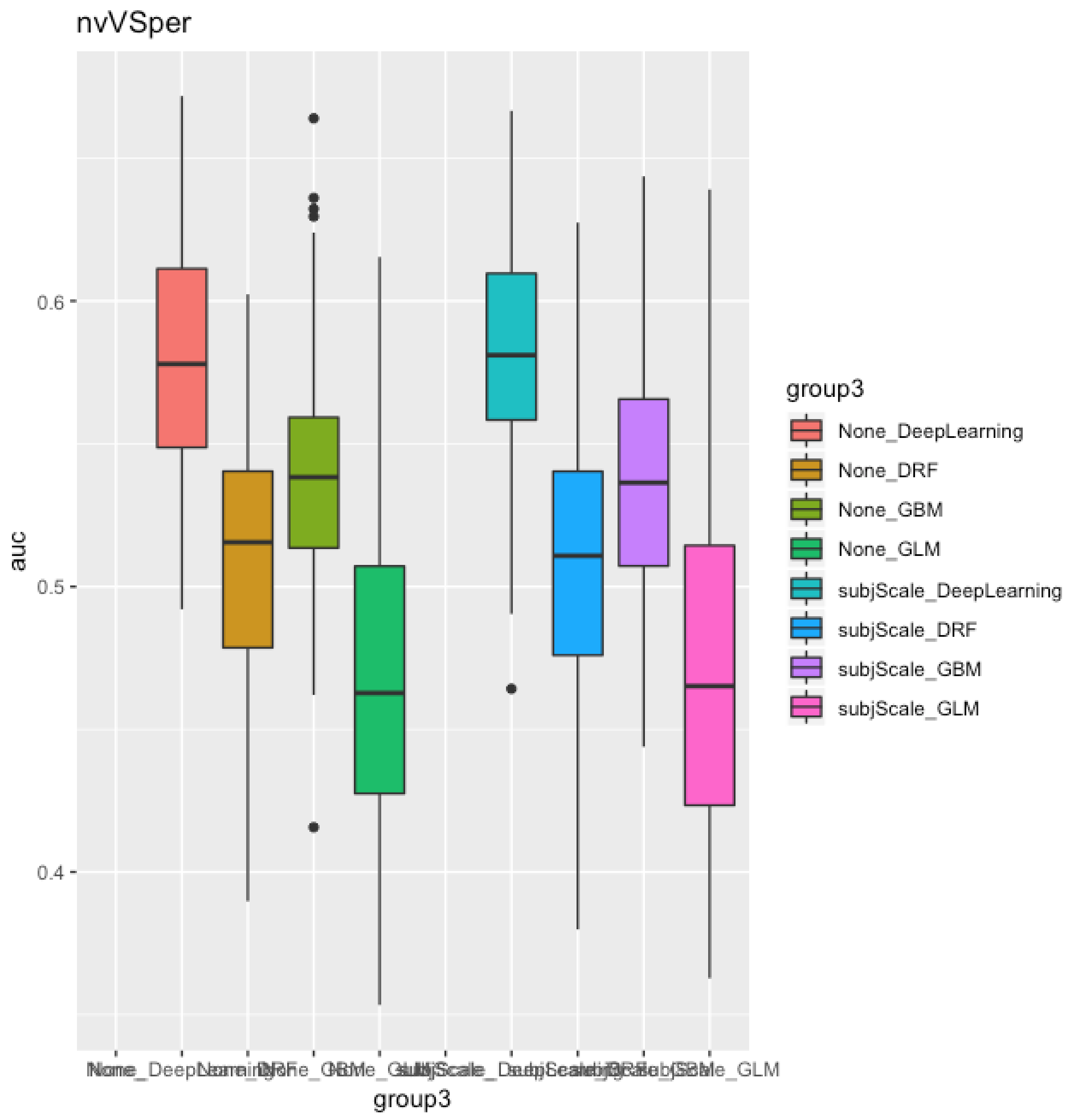
That DTI trend is also true for perVSrem, and struct still prefers subjScale.

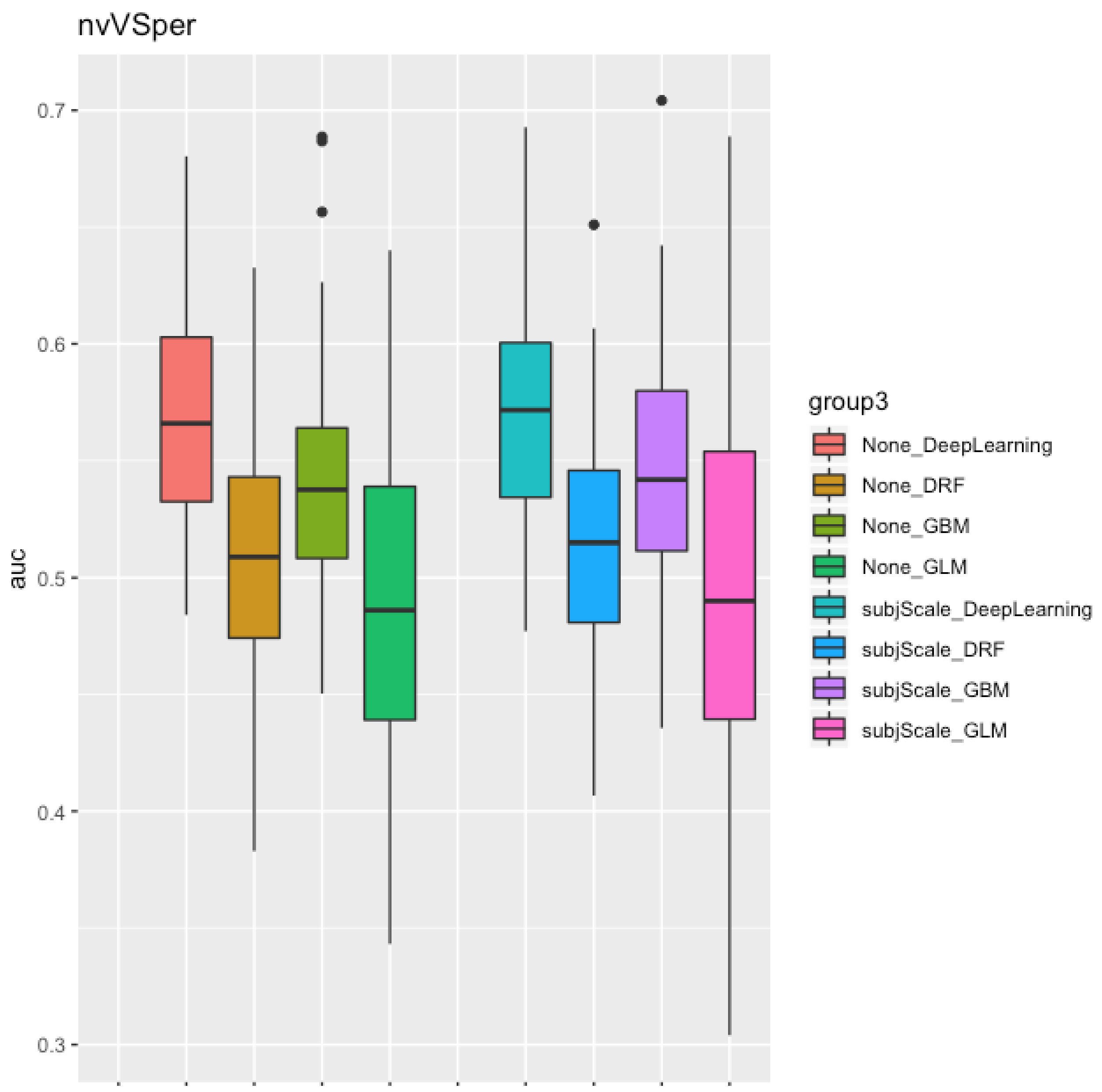

perVSrem
0.7
0.6
0.5
0.4
group3
None_DeepLearning
None_DRF
None_GBM
None_GLM
subjScale_DeepLearning
subjScale_DRF
subjScale_GBM
subjScale_GLM

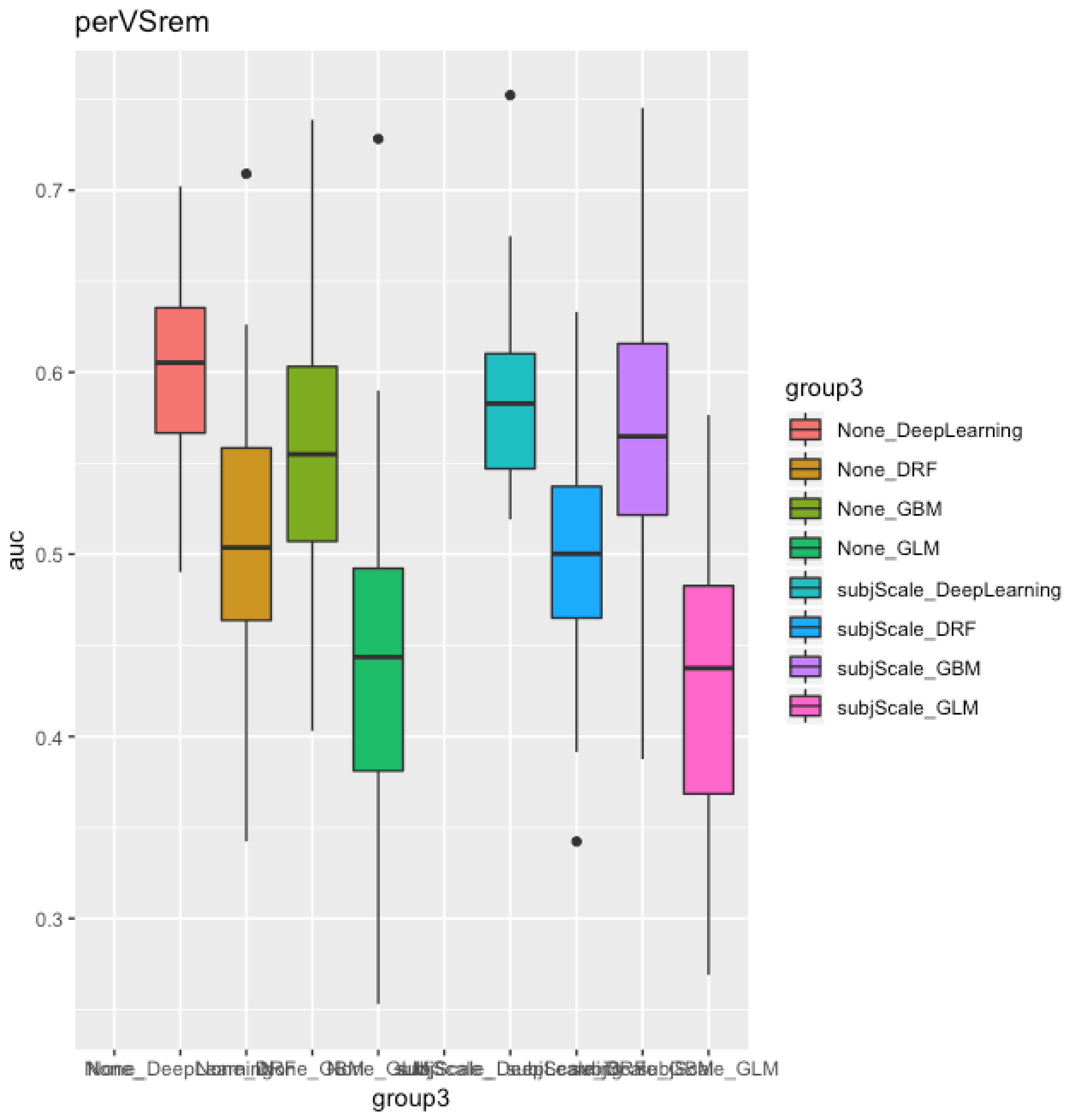


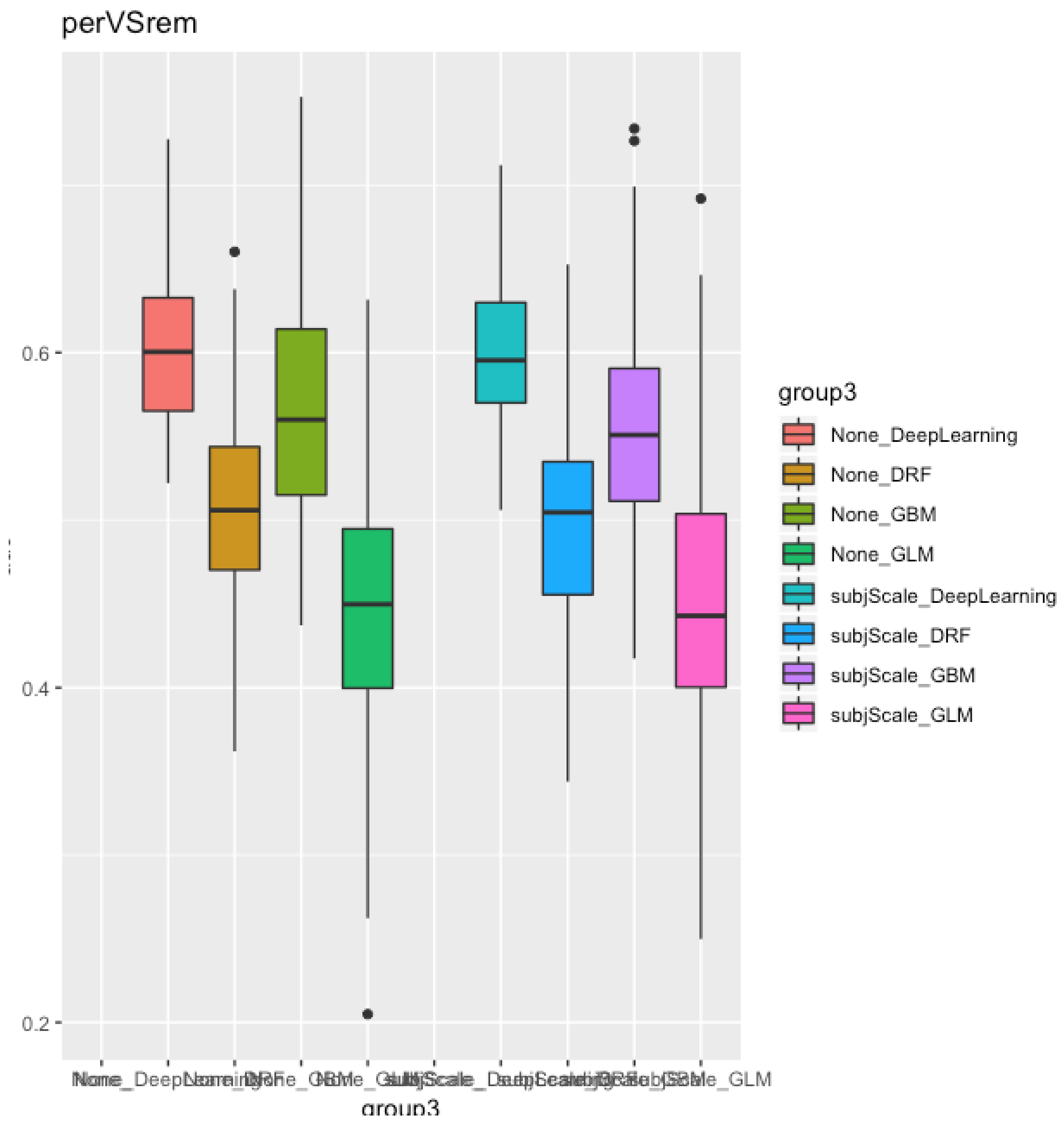
Does Deep Learning tend to overfit more?

```
idx = data$target=='nvVSper' & (data$var=='None' | data$var=='subjScale')
& data$seed<0 &
data$pheno=='struct_volume_11142018_260timeDiff12mo.RData.gz'
p1<-ggplot(data[idx,], aes(x=group3, y=auc, fill=group3))
print(p1+geom_boxplot() + ggtitle(unique(data[idx,]$target)))
```


nvVSper
auc
0.6
0.5
0.4
group3
group3
None_DeepLearning
None_DRF
None_GBM
None_GLM
subjScale_DeepLearning
subjScale_DRF
subjScale_GBM
subjScale_GLM

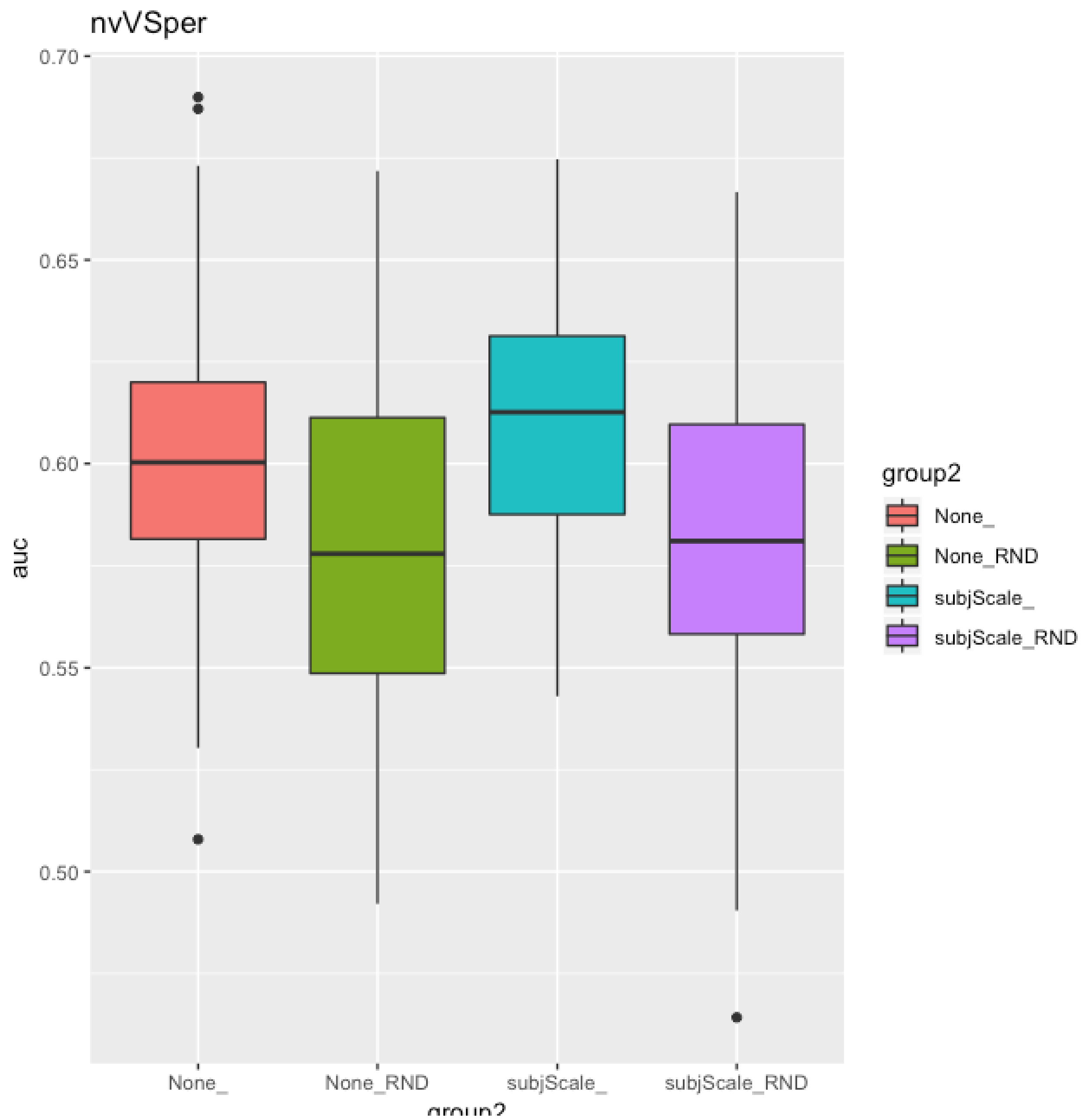

nvVSper
auc
0.7
0.6
0.5
0.4
0.3
group3
None_DeepLearning
None_DRF
None_GBM
None_GLM
subjScale_DeepLearning
subjScale_DRF
subjScale_GBM
subjScale_GLM

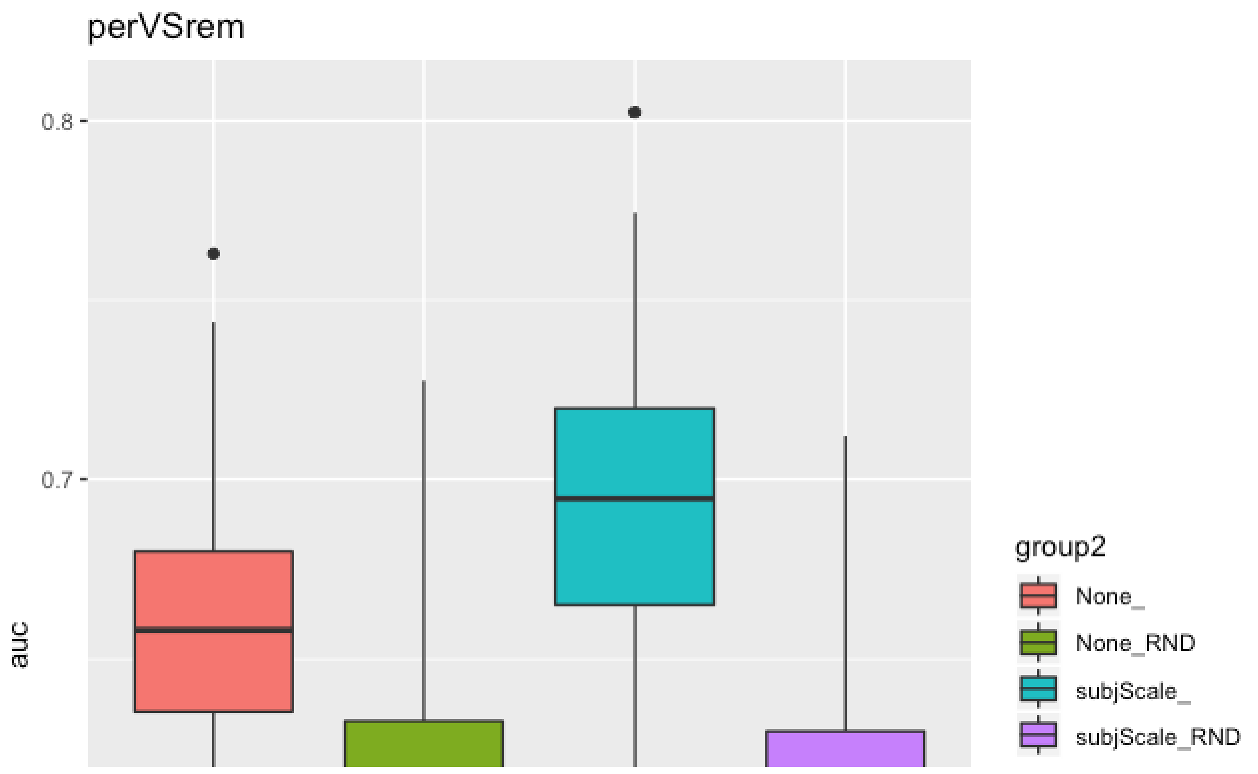

perVSrem
auc
0.7
0.6
0.5
0.4
0.3
group3
group3
None_DeepLearning
None_DRF
None_GBM
None_GLM
subjScale_DeepLearning
subjScale_DRF
subjScale_GBM
subjScale_GLM

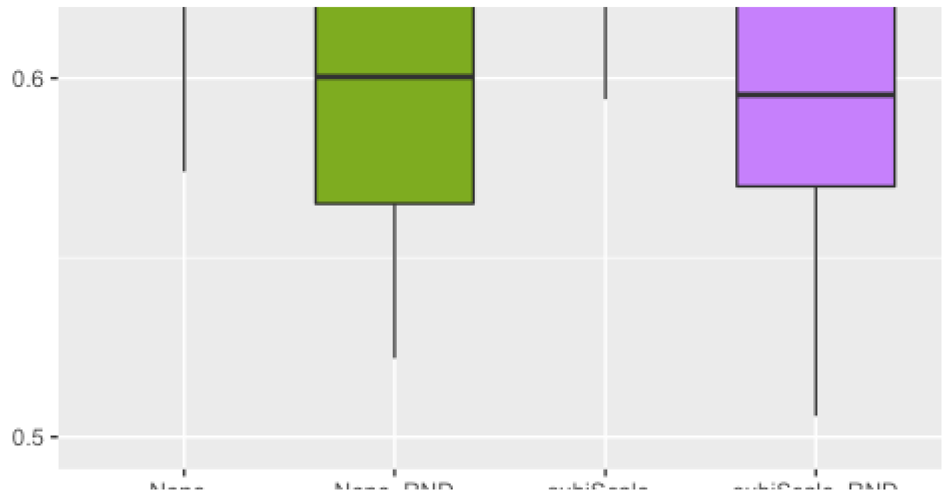


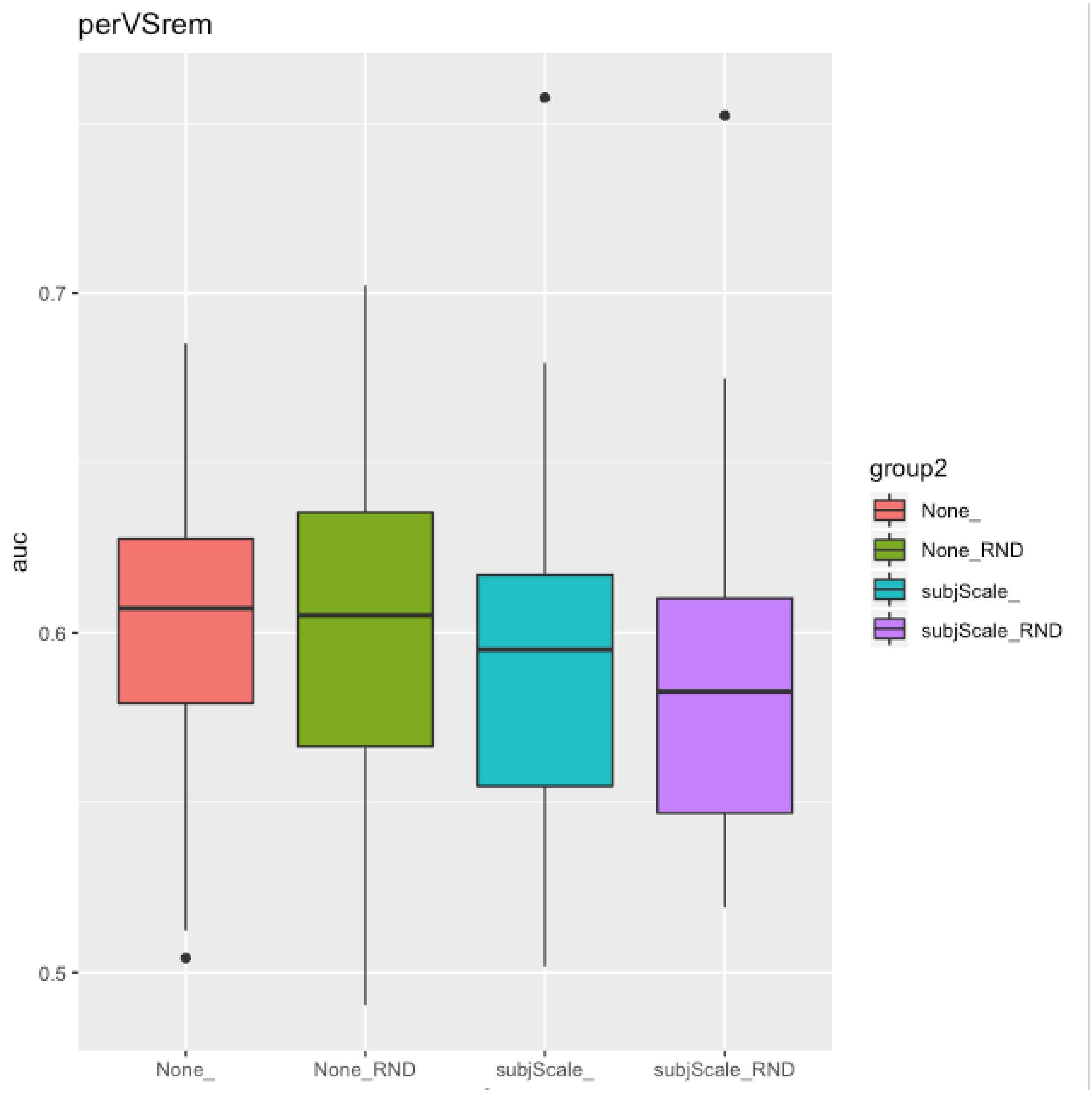
Yep, it definitely does, regardless of dataset or target. We just need to make sure whatever results we're getting are not there with random data:

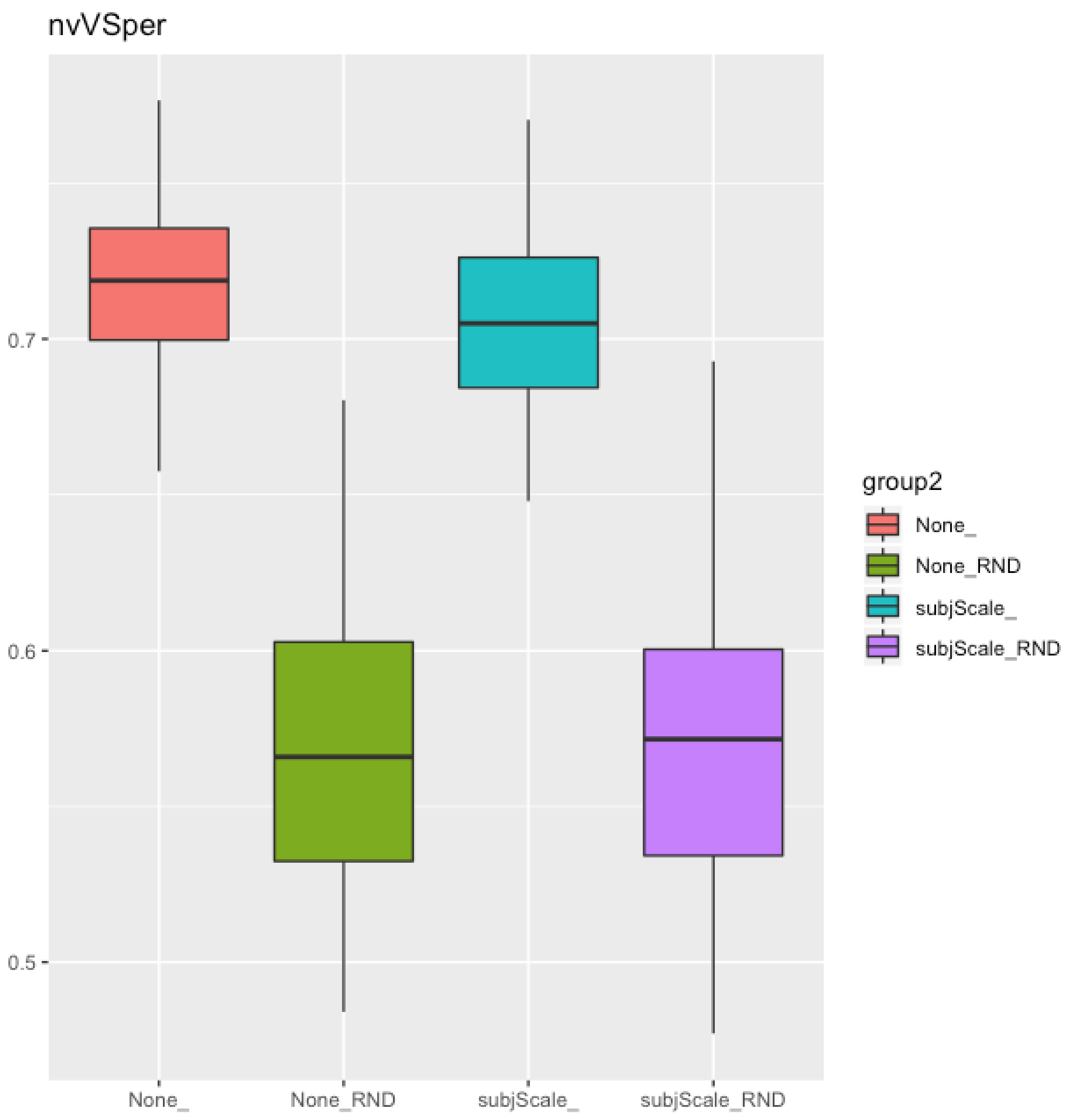
```
idx = data$target=='nvVSper' & (data$var=='None' | data$var=='subjScale')
& data$model=='DeepLearning' &
data$pheno=='struct_volume_11142018_260timeDiff12mo.RData.gz'
p1<-ggplot(data[idx,], aes(x=group2, y=auc, fill=group2))
print(p1+geom_boxplot() + ggtitle(unique(data[idx,]$target)))
```


nvVSper
0.70
0.65
0.60
0.55
0.50
auc
None_
None_RND
subjScale_
subjScale_RND
group2
group2
None_
None_RND
subjScale_
subjScale_RND


perVSrem
0.8
0.7
auc
group2
None_
None_RND
subjScale_
subjScale_RND


0.6
0.5


perVSrem
auc
0.7
0.6
0.5
None_
None_RND
subjScale_
subjScale_RND
group2
None_
None_RND
subjScale_
subjScale_RND



Maybe with nvVSper in structural, but definitely not in perVSrem, especially if we do subjScale. For DTI the story is different: not much for perVSrem, but very good for nvVSper.

So, across the board, DeepLearning does better, and it doesn't seem to care too much whether it's subjScaled or not, at least not for DTI. I wonder if this will change when we include rsFMRI, and if we combine data across domains? We could also keep subjScale for structural (and maybe fMRI) but not DTI?

```
job_name=dataTransformsDTIStruct_rawCV;
mydir=/data/NCR_SBRB/baseline_prediction/;
swarm_file=swarm.automl_${job_name};
rm -rf $swarm_file;
f1="${mydir}/dti_fa_voxelwise_n223_09212018.RData.gz,${mydir}/dti_ad_voxel
wise_n223_09212018.RData.gz,${mydir}/dti_rd_voxelwise_n223_09212018.RData.
gz";
f2="${mydir}/struct_area_11142018_260timeDiff12mo.RData.gz,${mydir}/struct
_volume_11142018_260timeDiff12mo.RData.gz,${mydir}/struct_thickness_111420
18_260timeDiff12mo.RData.gz";
f3="${mydir}/dti_fa_voxelwise_n223_09212018.RData.gz,${mydir}/dti_ad_voxel
```

```
wise_n223_09212018.RData.gz,${mydir}/dti_rd_voxelwise_n223_09212018.RData.
gz,${mydir}/struct_area_11142018_260timeDiff12mo.RData.gz,${mydir}/struct_
volume_11142018_260timeDiff12mo.RData.gz,${mydir}/struct_thickness_1114201
8_260timeDiff12mo.RData.gz";
for f in $f1 $f2 $f3; do
    for target in nvVSper perVSrem; do
        for pp in subjScale dataScale subjScale,dataScale None; do
            for algo in DeepLearning GLM; do
                for i in {1..100}; do
                    myseed=$RANDOM;
                    echo "Rscript --vanilla
~/research_code/automl/raw_multiDomain_autoValidation_oneAlgo.R $f
${mydir}/long_clin_0918.csv ${target}
${mydir}/models_raw_dataTransforms/${USER} $myseed $algo $pp" >>
$swarm_file;
                    echo "Rscript --vanilla
~/research_code/automl/raw_multiDomain_autoValidation_oneAlgo.R $f
${mydir}/long_clin_0918.csv ${target}
${mydir}/models_raw_dataTransforms/${USER} -$myseed $algo $pp" >>
$swarm_file;
                done;
            done;
        done;
    done;
done
sed -i -e "s/^/unset http_proxy; /g" $swarm_file;
split -l 1000 $swarm_file ${job_name}_split;
for f in `/bin/ls ${job_name}_split??`; do
    echo "ERROR" > swarm_wait_${USER}
    while grep -q ERROR swarm_wait_${USER}; do
        echo "Trying $f"
        swarm -f $f -g 60 -t 16 --time 3:00:00 --partition norm --logdir
trash_${job_name} --job-name ${job_name} -m R --gres=lscratch:10 2>
swarm_wait_${USER};
        if grep -q ERROR swarm_wait_${USER}; then
            echo -e "\tError, sleeping..."
            sleep 10m;
        fi;
    done;
done
```

2018-11-19 10:41:46

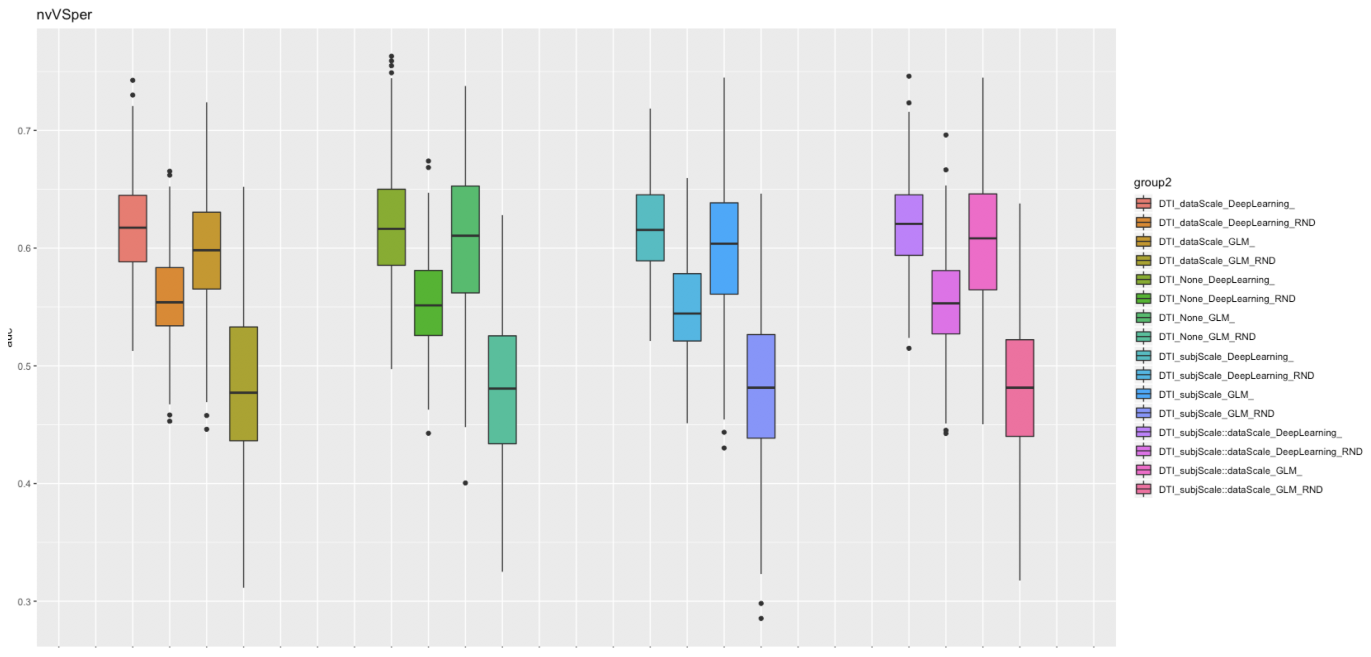
Let's see how the best fMRI run are affected by data transformation:

```
job_name=dataTransformsFMRI_rawCV;
mydir=/data/NCR_SBRB/baseline_prediction/;
swarm_file=swarm.automl_${job_name};
rm -rf $swarm_file;
for f in aparc_pcorr_kendall_trimmed_n215_11152018.RData.gz \
    aparc_pcorr_pearson_n215_11152018.RData.gz \
    aparc.a2009s_corr_kendall_n215_11152018.RData.gz; do
    for target in nvVSper perVSrem; do
        for pp in subjScale dataScale subjScale,dataScale None; do
            for algo in DeepLearning GLM; do
                for i in {1..100}; do
                    myseed=$RANDOM;
                    echo "Rscript --vanilla
~/research_code/automl/raw_multiDomain_autoValidation_oneAlgo.R
${mydir}/$f ${mydir}/long_clin_0918.csv ${target}
${mydir}/models_raw_dataTransforms/${USER} $myseed $algo $pp" >>
$swarm_file;
                    echo "Rscript --vanilla
~/research_code/automl/raw_multiDomain_autoValidation_oneAlgo.R
${mydir}/$f ${mydir}/long_clin_0918.csv ${target}
${mydir}/models_raw_dataTransforms/${USER} -$myseed $algo $pp" >>
$swarm_file;
                done;
            done;
        done;
    done;
done
sed -i -e "s/^/unset http_proxy; /g" $swarm_file;
split -l 1000 $swarm_file ${job_name}_split;
for f in `/bin/ls ${job_name}_split??`; do
    echo "ERROR" > swarm_wait_${USER}
    while grep -q ERROR swarm_wait_${USER}; do
        echo "Trying $f"
        swarm -f $f -g 60 -t 16 --time 3:00:00 --partition norm --logdir
trash_${job_name} --job-name ${job_name} -m R --gres=lscratch:10 2>
swarm_wait_${USER};
        if grep -q ERROR swarm_wait_${USER}; then
            echo -e "\tError, sleeping..."
            sleep 10m;
        fi;
    done;
done
```

And compile the results of data transformation on DTI plus struct:

```
echo "target,pheno,var,seed,nfeat,model,auc,f1,acc,ratio" >
dataTransformsDTIStruct_summary.csv;
dir=dataTransformsDTIStruct_rawCV;
for f in `ls -1 trash_${dir}/*o`; do
    phen=`head -n 2 $f | tail -1 | awk '{FS=" "; print $7}'`;
    phen2=`echo $phen | sed -e "s/,/::/g"`;
    target=`head -n 2 $f | tail -1 | awk '{FS=" "; print $9}'`;
    seed=`head -n 2 $f | tail -1 | awk '{FS=" "; print $11}'`;
    var=`head -n 2 $f | tail -1 | awk '{FS=" "; print $13}'`;
    model=`grep -A 1 model_id $f | tail -1 | awk '{FS=" "; print $2}' |
cut -d"_" -f 1`;
    auc=`grep -A 1 model_id $f | tail -1 | awk '{FS=" "; print $3}'`;
    nfeat=`grep "Running model on" $f | awk '{FS=" "; print $5}'`;
    ratio=`grep -A 1 "Class distribution" $f | tail -1 | awk '{FS=" ";
{for (i=2; i<=NF; i++) printf $i ";"}}'`;
    f1=`grep -A 2 "Maximum Metrics:" $f | tail -1 | awk '{FS=" "; print
$5}'`;
    acc=`grep -A 5 "Maximum Metrics:" $f | tail -1 | awk '{FS=" "; print
$5}'`;
    echo $target,$phen2,$var,$seed,$nfeat,$model,$auc,$f1,$acc,$ratio >>
dataTransformsDTIStruct_summary.csv;
done;
sed -i -e "s/subjScale,dataScale/subjScale::dataScale/g"
dataTransformsDTIStruct_summary.csv
```

```
data = read.csv('~/tmp/dataTransformsDTIStruct_summary.csv')
data$pheno2 = 'DTI'
idx = grepl('struct', data$phen2)
data[idx, 'pheno2'] = 'Struct'
idx = grepl('struct', data$phen2) & grepl('dti', data$phen2)
data[idx, 'pheno2'] = 'DTI+Struct'
data$group = ''
data[data$seed<0,]$group = 'RND'
data$group2 = sapply(1:nrow(data), function(x) { sprintf('%s_%s_%s_%s',
data$pheno2[x], data$var[x], data$model[x], data$group[x])} )
idx = data$target=='nvVSper' & (data$var=='None' | data$var=='subjScale')
& data$seed>0 &
data$pheno=='struct_volume_11142018_260timeDiff12mo.RData.gz'
p1<-ggplot(data[idx,], aes(x=group2, y=auc, fill=group2))
print(p1+geom_boxplot() + ggtitle(unique(data[idx,]$target)))
```



This pattern observed in nvVSper is very telling, and resembles what we had seen before. The "4 blocks" of results show the 4 different data transformations, basically showing that the best results is about the same for all of them when combining DTI. DeepLearning is about the same regardless of data transformation, but has smaller variance then GLM. DeppLEarning overfits a bit though, when compared to GLM, although the results are still better than chance even with the overfit.

Q: are thse results better than single DTI