MVPA TOOLBOX
A BRIEF INTRODUCTION

Version: alpha-1-0-7

This MVPA toolbox allows you to perform binary classification using linear and non-linear SVM from the libsvm toolbox (http://www.csie.ntu.edu.tw/~cjlin/libsvm/). The other major feature is that it can apply the classifier to a 3rd (or more) populations. It has some functionality to do other MVPA analyses such as hierarchical clustering or self-organizing map though not from the GUI and are not user-friendly yet. It is matlab-based though future versions may be in Python (or integrated with MVPA toolboxes out there). The toolbox changes almost on a weekly basis, so please check in with F Hoeft for updates.

SOME KEY FEATURES

- Data preprocessing from images to matrices, adjusting matrices for nuisance variables and various forms of normalization and mean-centering
- Feature / dimensionality reduction options such as pre-cross-validation (CV) mask, pre-CV / within-CV PCA, pre-CV / within-CV univariate ttest, within-CV standard, nested recursive feature elimination (RFE)
- Several options to combine data from different brain regions or modalities (e.g. fMRI, sMRI, Bx, genetics, environmental) such as simple combination, PCA output, SVM distance measure.
- Grid-search for penalty constant and gamma, linear and non-linear SVM, weighted-SVM, L1/L2 SVM.

NEW FEATURES SINCE 5/18/2010 Version alpha-1-0-4

- Option to do non-linear SVM (radial basis function), and optimization of variable gamma 'γ' (in addition to the penalty constant 'C').
- Weighted SVM (using square-root of sj N in addition to the standard N).
- Expanded options to optimize penalty constants (for linear and non-linear SVM, default C=1) and gamma values (for nonlinear SVM, detault γ=1/num_features)
- Started including version info in log file.
- Cleaned output file structure to reduce clutter.
- Leave-two-out and matching N when sample-size is unequal.
- Calculate threshold after doing permutation analysis (both compared to chance level and compared to another classifier).
- When running permutation and you know you will be comparing a classifier with another classifier you have already run, you can choose an option where the randomization of class labels are matched for the 2nd classifier you are permuting.
- FIXES: Fixed some bugs (e.g. nested ttest mask options), restructured applyClassifier, mat2img and permutation options to accommodate newer options. Also fixed 'additional features' options. (concatenate matrices option still has bugs in that it gets stuck calculating TP (true positive) values etc at the end).

FEATURES NOT FULLY TESTED

SVM – additional feature opt not fully tested. L1 SVM not fully tested (but many/most functions should work.), concatenating matrices within SVM not tested (but you can concatenate before hand and run analyses).

PLANS FOR FUTURE RELEASES

Statistical comparisons between classifiers (and not just whether significantly better than chance), Boosting, Reliability filter, k-fold CV, Multi-class SVM, SVM Regression, New GUI

ACKNOWLEDGMENT

A Etkin for lots of brain-storming, C Chang for the L1 function and brain-storming, E Walter for initial GUI implementation, S Bahl for libSVM initial implementation, libsvm group in Taiwan (esp. Chen-Tse Tai, and Chih-Jen Lin), and many others at Stanford for suggestions and advice.

PUBLICATIONS USING THIS SOFTWARE

Hoeft et al. Arch Gen Psychiatr 2008, Etkin et al. Am J Psychiatr 2010, and several under review (Hoeft*, Walter* et al., Hoeft, McCandliss et al., Marzelli, Hoeft et al.)

1. INSTALLATION AND STARTING THE MVPA TOOLBOX

Drag the whole SVM_gui_libsvm folder to your matlab directory (or where you put other things like SPM8). Start matlab and go to File then Set Path and Add with Subfolders and choose the SVM_gui_libsvm dir.

2. INFORMATION YOU NEED TO HAVE BEFORE YOU START

1. ***INPUT IMAGES OF ALL SUBJECTS (note file format by doing SPM Display of 1 image, e.g. int16)***
   e.g. - smoothed and normalized VBM images
   - fMRI con images
2. ***MASK FILE (note file format by doing SPM Display, e.g. int16)***
   This will restrict regions that you want to examine. The mask does not have to be binary, but the regions you are interested in need to have a value greater than 0. Basically you want to restrict your voxels so that you won't get 'noise'. But you also don't want to bias your data so that you are choosing regions that show the effect you will be looking for (e.g. patients vs. controls).
   Ideally, you also want to do everything in your cross-validation loop (i.e. keep training and test data as separate as possible). So if you are going to be a purist, then choose a really standard mask such as option 3rd of 4th below, and then do everything within the SVM loop to further restrict your voxels (features) – known as nested feature elimination.
   In reality however, people do still get away with using any of these types of masks even if they are generated by data including the test data.
   e.g. - grey matter mask you used for your VBM univariate stats
   - mask.nii that was produced from SPM stats
   - some SPM template
   - multiple AAL ROIs like mPFC, amygdala and hippocampus (made separately or combined as one image)
   - fMRI stats image thresholded at p=0.05 uncorrected for task vs. baseline
3. ***CLASS (GROUP) INFORMATION***
   Information regarding which image belongs to which group (e.g. patient, control)
4. ***REGRESSORS (optional)***
   Information regarding regressors perhaps organized in a mat file, text file or excel.
   e.g. - total grey matter volume if using VBM images
   - age, task performance, etc…

3. DOWNSAMPLE DATA [optional]

<Downsample (opt): resample.m>

This uses the imcalc routine in SPM8 and down-samples data to 2mm or 4mm voxel images.
PRO: Saves you computational time at later steps.
CON: If you downsample to 4mm, the performance won't change too much (in my experience) as you want to use smoothed data anyway, but the reconstructed images could look coarse.
The other time you want to use this is just to put all images in the same bounding-box. You will need to do this if you want to use Img2Mat to construct a data matrix (but you can use REX+CONCAT as well).

1. ***Choose images you want to resample. Hit done.***
2. ***Select 2 or 4mm, the size you want to down-sample to.***

4. CONSTRUCTING DATA MATRICES (mat4SVM_xxx)

This step creates the main mat4SVM matrix that you will feed into SVM.

Option 1: Img2Mat

<Img 2 Mat: image2matrix.m>

1. ***Type in file name to save (e.g. mat4SVM_FxsCon_gm)***
2. ***Choose individual subjects' images.***
3. ***Type in file-type (it is int16 (default) if you resampled)***
4. ***Choose mask image.***
5. ***Type in file-type (it is int16 (default) if you resampled)***

Option 2:

<Extract from Mask(s) (REX): rex.m>

This is a program written by Sue Whitfield-Gabrieli from MIT.
PROS: This one has the advantage of being able to extract voxel-by-voxel data from multiple ROIs all at once and also not having to put into the same bounding box prior to this step.
CON: Takes longer.

1. ***Sources – choose images***
2. ***ROIs – choose mask(s)***
3. ***Choose Voxel-level***
4. ***Save REX project & output files (only the .txt format needs to be selected).***
5. ***Hit Done.***
6. ***Hit Extract.***

<Concatenate Matrices: concatMatFles.m>

1. ***Choose the .txt output file from REX. Hit Done.***
2. ***Choose output file name (e.g. mat4SVM_ConFxs_gm_rex)***

Note: Concatenating Matrices

You have a number of options to do this.
- You can do it manually in matlab (e.g. mat4SVM = [a b]).
- You can use Concatenate Matrices if you want to put them together before you do anything (e.g. combining ROIs).
- If you choose 'Additional Feature' option with SVM, then it will add as an additional matrix but will just add as it (and won't normalize PCA or do other things to that matrix).

5. ASSIGNING CLASS LABELS (class_xxx)

<Create Class File: createClass.m>

This creates a class file that labels one group with a label of +1 (group of interest) and the other with -1.

1. ***Type in file name to save (e.g. class_Fxs+1_Con-1)***
2. ***[(-1)*ones(1,22) ones(1,28) (-1)*ones(1,28) ones(1,24)]***
   (or you can type in '1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 …..' without the … and '' if you want)

6. TAKING OUT THE EFFECT OF NUISANCE VARIABLES (regressor_xxx, mat4SVM_res_xxx) [optional]

<Create Regressor File (opt): createRegressors.m>

This step creates a matrix with regressors. For example you might want to take out the effect of total GMV from your VBM matrix or age and task performance from your fMRI matrix.

1. ***Type in file name to save (e.g. regressors_FxsCon_gm)***
2. ***Type in how many regressors you have (e.g. 3 for age, scanner and TGMV)***
3. ***Enter each regressor as a vector (e.g. '3.3 2.2 1.7 … 4.0' for age, then '1 1 1 1 1 2 2 2 2 2…2' for scan site, then '800 780 900 … 700' for TGMV). (copy and paste from excel is easy)***

<Get Residuals (opt): getResiduals.m>

This step residualized the effect of the regressors from your matrix. It could take a long time if you have many voxels.

1. ***Type in file name to save (e.g. mat4SVM_res_FxsCon_gm)***
2. ***Select the mat4SVM file you saved earlier (e.g. mat4SVM_FxsCon_gm)***
3. ***Select the regressor file you saved earlier (e.g. regressors_FxsCon_gm)***

7. NORMALIZING THE MATRIX (mat4SVM_normOpt4_res_xxx) [optional]

<Normalize matrix (opt): normalization.m>

This step normalizes your data either across subjects, across features (e.g. voxels) or in both directions such that mean=0 and SD=1.
Since version alpha-1-0-4, there is now an option of mean centering only for either across subjects, across features, and in both directions.
You definitely want to do this prior to SVM but probably not after PCA. There is an option within SVM where you can do it within the CV (cross-validation) loop.
Typically one would choose this option (or PCA) when they want to do lots of testing to see what works best then they would do the final analyses within the SVM CV loop.

1. ***Type in file name (e.g. mat4SVM_normOpt4_res_FxsCon_gm).***
2. ***Concatenate matrices (typically, choose no).***
3. ***Select mat4SVM file (e.g. mat4SVM_res_FxsCon_gm).***
4. ***Select norm option (typically choose 4th option, within then across subjects).***

8. PCA (xxx_PCs, xxx_EOFs) [optional]

<PCA (opt): pcaCheck.m>

This step performs PCA on all data and tells you the optimal number of components you should choose (see matlab command window after this runs). You can choose how much variance your eigenvectors to explain (a value between 0.7-0.9 are commonly used. You will see that neuroimaging yields many more PCs than PCA of Bx data for example).
It also saves the PCs and loadings (eofs) so one can reconstruct back into the original (well close to) matrix later for example to create brain maps.

1. File name to save output (e.g. mat4SVM_normOpt4_res_FxsCon_gm (it will add _PCs and _eofs at the end of each output file so naming it the same as above is ok.).
2. Select mat4SVM file (e.g. mat4SVM_normOpt4_res_FxsCon_gm).
3. Select subset of subjects to examine (typically choose the default value, e.g. in this case 1:102).
4. Choose none for Norm Option if you have used normalized data. (also note that PCA will center the data for you, but also note that centering is different from normalizing).
5. Choose PCs you would like to save. If you choose 'optimal' it will ask for the optimal eigenvalue you want to use (choose somewhere between 0.7 and 0.9). e.g.
   - You might want to choose '1' or '2' if you want to combine later lots of PCs from many different types of data such as many ROIs (116 AAL labels), Bx, genetics, fMRI, VBM, etc… so that each modality will be 1 or 2 features in the SVM).
   - You might want to choose 'optimal' if you want to put this matrix in SVM and use all the output PCs in SVM without feature reduction.

- You might want to choose 'all' if you want to put this matrix in SVM and perform RFE (recursive feature elimination) to see which PCs are important for SVM. This is much much faster than doing it in the loop. Once you figure things out, then you can go back and do it 'properly', PCA within the loop later.

9. SVM

<SVM (Support Vector Machine): svm_gui.m, svm_loop.m>

This is the main part of the toolbox. It has many options and there are no 'best ways' although there are 'better ways' to do things. So I will go through one sample and may add other scenarios later. Although I don't have any documentation, you can type 'help svm_gui' in the command window to figure out what options you can choose from and some minimal explanation (sorry need to work on documentation).

Before you start, I like reorganizing my files that were created during preprocessing. As you do more analyses files could easily get confused. I like having folders like Class_Labels, Mat4SVM_NormOpt4, Mat4SVM_Res, Mat4SVM_Raw, Intermediate, for example.

SAMPLE 1: (Good for exploratory analyses)

1. ***PREP***
   Create a results directory (e.g. RES_sample1) and cd (change directory) into that directory. Start SVM and click on SVM button.
2. ***EXISTING LOG FILE (choose NO)***
   Choose No for 'Use saved log file?' unless you have run the same process before and it crashed mid-way or you modified an existing log file to be run for a new analysis.
3. ***TYPE PREFIX***
   I usually make the directory name more intuitive and just type in 'a' or 'test' for this. Easier that way.
4. ***SELECT MAT4SVM FILE (e.g. choose mat4SVM_normOpt4_res_FxsCon_gm)***
   You can choose your PC data file (e.g. **mat4SVM_normOpt4_res_FxsCon_gm_PCs**) if you want to do a quick test (though results could be fairly different from the whole sample vs. PC data.). If you plan to compare with a 3rd group later (i.e., use applyClassifier option), then you might want to consider making a large matrix comprising of all 3 (or more) groups) and feeding it in here (for the analysis, it will only take the 2 groups you are interested in).
5. ***SELECT CLASS FILE (choose class_Fxs+1_Con-1)***
6. ***SVM OPTION (choose regular L2)***
   You can try any of them though L1 may have some bugs. It has not given me great results so far but let me know if it does.
   You can also try weighted SVM if you have unequal sample size, but with linear SVM and small sample-size, it seems like it may not make a big difference. Since version alpha-1-0-4, we also added an option where you can choose the square-root of N (sample-size) but this typically does not change things much either.

LibSVM has another way to do weighted SVM and we may implement that sometime in the near future.

7. ***RECURSIVE FEATURE ELIMINATION OPTION (choose NO RFE)***
   Typically you want to reduce features somehow even though SVM is pretty good with high-dimensional data but for this example we will choose this. You can also try this as first pass as a starting point to see the performance. Often if this works, you are in good shape!
   If you have PCs as input data, then you can certainly choose NO RFE as the dimensions are reduced already and are reasonable.
   Regular RFE does it within CV but does not do nested (CV of CV essentially) meaning that the features are selected without separating training and test data, so it is likely to bias you toward worse results than nested RFE where you would separate training and test data for both defining features that give good performance (high weight) and testing ultimate performance. Fixed target RFE is essentially the same as nested RFE but saves more stuff along the way (e.g. performances each time it reduces features so that you can see the optimal performance displayed as a graph later on).
   Regular RFE is a good option to start and could work great if you have PCs as input but often it selects too few features and does not yield great performance.
8. ***MATCH SAMPLE-SIZE (FOR VERY UNEQUAL N)? (New since version alpha-1-0-5)***
   I wrote multiple versions of this so let me know this is not exactly what you are looking for…
   Sometimes when the sample-size is very unequal, you can certainly match them before you start SVM and preselect which subjects you want to include, but if you want to utilize all your subjects, this might be the option for you. Often by running this, you will eliminate the problem of having very skewed results (e.g. very high sensitivity and chance or below chance level specificity).
9. ***ADDITIONAL FEATURES (choose NO)***
   If you have say Bx data you want to add and wanted to treat that matrix separately and have it preprocessed already then you would choose this option. Recently it has not been tested with all new options so I am almost certain it won't work. Let me know if you desperately need this to work.
10. ***PENALTY (COST) CONSTANT AND GAMMA (choose default NO)***
    Many people still use the default C=1 so you can choose NO here. You can also choose to find the optimal C val, or if you know your optimal C val already, you can select 'choose your own C val' and type in a value (typically between 2^-5 to 2^15).
    Since version alpha-1-0-4, there is now an option to optimize gamma value as well. You would choose this if you wanted to optimize parameters for nonlinear SVM (for linear SVM, you only need to optimize C values).
11. ***UNIVARIATE TTEST MASK OPTION (choose default NO as we have masked our sample matrix already with a GM mask)***
    2sample ttest (assumes unequal variance) is sort of cheating as that is what you are interested in, although it is ok as long as you choose the 'nested (meaning it does it within CV)' option. 2sample is ok for VBM and fMRI data.
    1sample ttest would be mostly for fMRI as VBM images always have only positive values so all voxels will be selected anyway unless all subjects have a value of 0 in that particular voxel. For nested you have the option of choosing the

union of all voxels or intersection only. Seems like these voxels that are selected for an unknown reason could choose fairly different voxels than the SPM output. If you wanted to play around with this, I would also start with 'noNested 1sample union' option (takes the union of all voxels that are significant examined separately for each class label (if you ad a 3rd group, for the 3rd group as well).

12. ***NORMALIZATION OPTION (choose NO since we normalized data already).***
    To do it properly (though most people don't document in their manuscript how it was done) it is most proper to do it within CV hence do normalization here at this stage. I often do a bunch of testing on normalized data and once I finalize the parameters, choose this option to get final performance.
    There is no one way to do normalization and all seem ok to do though I often choose BOTH these days.
13. ***PCA OPTION (choose NO for now)***
    Again if you want to test out how PCA works, you might want to use the PCA button to create a matrix of PCs and test that out first (super fast). If it works then use this option where it would do PCA from all voxel data and within CV to get final classifier and performance.
14. ***Then it will save a log file and should start running!***
15. ***OUTPUT***
    xxx_log.mat: Contains all parameters you chose.
    xxx_output.mat: Contains all necessary output information for further processes.
    xxx_svmResults.txt: Results (accuracy, sensitivity, specificity, positive predictive value, negative predictive value).
    xxx_distance.txt: Distance of each subject from hyperplane.
    The last two outputs are redundant with xxx_output.mat but have done so for each access.

10. Applying Classifier to New Data (one or more populations)

<Apply Classifier to New Data (opt): svm_apply_gui.m, svm_apply.m, svm_loop.m>

You would use this option if you had for example a 3rd group (e.g. 2nd patient group) and wanted to know if the 1st patient group vs. control classifier will classify the 2nd patient group as the 1st patient group or the control group. Here you can have 2 (or more) groups mixed as well.

Before you start, you need to prepare a couple of files. First you will need a new class file for these subjects. For example if you hypothesize that the 2nd patient group will be categorized as controls and if controls were labeled as '-1' in SVM, then you would label the 2nd group as '-1's. If you have a 2nd patient group plus a new group of controls and you hypothesize that the 2nd patient group are more like the 1st patient group (labeled as '+1's) and the new control group is more like the 1st control group (labeled as '-1's) then you would give a class label of '+1's to the 2nd patient group and '-1's to the new control group.

You will also need to prep a mat4SVM for the this data-set. You can either prepare them similarly to how you did for the primary SVM analysis, or you can prepare the data you use for the primary SVM and the data you want to use here in one large matrix. The advantage of doing it the latter way is that if you are doing creating residuals, doing normalization, PCA etc… to the data, then you can apply similar parameters to the this new data-set. If you choose to do it the former, I would suggest you apply the normalization parameters, PCA weights etc from the primary SVM data-set to this new data-set. This method is more conservative as you are not 'peeking' at the new data-set when you preprocess that data (though most studies I believe don't do this and treat all these preprocessing steps as preprocessing and include all data together).

SAMPLE 1: (Good for exploratory analyses)

1. ***PREP***
   Cd (change directory) into the directory where your primary SVM results resides. Start SVM (type 'SVM' without the '' in the terminal) and click on 'Apply Classifier to New Data'.
2. ***EXISTING LOG FILE (choose NO)***
   Choose No for 'Use saved applyClassifier log file?' unless you have run the same process before and it crashed mid-way or you modified an existing log file to be run for a new analysis.
3. ***PRIMARY LOG FILE***
   Select the log file created when doing the primary SVM analysis.
4. ***APPLY CLASSIFIER TO?***
   Select 'New Data' if you want to choose a mat4SVM that consists only the new data and this new data were not included in the mat4SVM used for primary SVM. Select '3rd group from SVM' if the mat4SVM from the primary SVM already included the 3rd group (Note that the 3rd group refers to all new data where you want to apply the classifier from the primary SVM to and can include more than one population).
   If you choose 'New Data', then you will be prompted to select the new mat4SVM file.
5. ***NEW CLASS FILE***
   Select new class file.
6. ***OUTPUT***
   xxx_applyClassifierLog.mat: The parameters used in this analysis is saved here.
   xxx_applyClassifierOutput.mat: Output is saved here. Classification accuracy of new data will be printed in the command window also.
   APPLCLASS.acc: accuracy
   APPLCLASS.dist: distance measure

11. PERMUTATION ANALYSES AND DETERMINING THRESHOLDS

<Permutation: svm_permut.m, svm_loop.m, svm_apply.m>

You would almost always want to do this step and the next to determine two kinds of thresholds (one may not be relevant for you if you did not use brain data or if you used RFE for your SVM process). The first type of threshold you will get tells you whether the primary SVM (and applyClassifier SVM) is significantly better than chance (and to compare with other classifiers). The other threshold is for your brain maps if your input data were brain maps (Note that some don't consider this legitimate as in theory if you used the whole brain to classify the groups, then it is kind of odd to threshold the brain maps as the whole-brain was necessary to obtain that classification accuracy. In practice, however, people do this just for visualization purposes to show which ones had the greatest weights (either positive or negative)).

1. ***PREP***
   Cd (change directory) into the directory where your primary SVM results resides. Start SVM (type 'SVM' without the '' in the terminal) and click on 'Apply Classifier to New Data'. You do not need to have done the ApplyClassifier step to run this… just the SVM step is necessary.
2. ***EXISTING LOG FILE (choose NO)***
   Choose No for 'Use saved permutation log file?' unless you have run the same process before and it crashed mid-way or you modified an existing log file to be run for a new analysis.
3. ***PRIMARY LOG FILE***
   Select the log file created when doing the primary SVM analysis.
4. ***HOW MANY PERMUTATIONS?***
   Type 100 or so if you just want to check out approx the range. If you want to do formal analyses, type 2000 or so.
5. ***P-VALUES FOR DISLAYING BRAIN-MAPS***
   In most cases you can keep the default values which are p=0.05, p=0.01, p=0.001. Here you specify the thresholds this process outputs, which will be used to threshold the brain for example when you display the brain maps later using MRIcroN. If you did not use brain maps as input, it does not matter what you choose here as you won't need this info.
5. ***USE PERMUTED CLASS LABELS FROM DIFFERENT PERMUTATION?***
   If you plan on comparing performance of this classifier with a different classifier, which you have run permutation analysis on already, then choose 'Yes' and choose the xxx_randClassLabel.txt file that was generated from the permutation of the classifier you want to compare with. Otherwise say 'No'.
6. ***DID YOU USE PCS AS ORIGINAL MAT4SVM FILE?***
   If your input mat4SVM was principle components rather than voxels, then say 'Yes' here. It will look for the 'eofs.mat' file so please keep that in the original location and together with the 'PCs.mat' file.
7. ***PERMUTATION ON APPLYCLASSIFIER COMPONENTS TOO?***
   If you did ApplyClassifier and want to calculate thresholds for that, select 'Yes'. If not select 'No'.

8. ***OUTPUT***

xxx_permutLog.mat: Parameters chosen for this process is saved here.
xxx_permutCoeffThres.txt: Saves thresholds to be used when displaying brain maps (default will be lower then upper bound for p=0.05, lower then upper bound for p=0.01, and lower then upper bound for p=0.001).
xxx_permutAccuracy.txt: Saves accuracy (plus sensitivity, specificity, positive predictive value, negative predictive value) for each permutation when random class labels are assigned in each permutation.
xxx_permutAccuracyApply.txt: If you chose 'Yes' for process 8 above, then it will save this additional file. Where it will tell you if your applyClassifier process was significant.
Xxx_randClassLabel.txt: The list of all randomized class labels will be listed for all permutations.
Note: **If your computer crashes mid-way**, you can reselect the parameters (say for example if it crashed 100 permutations away from completion, you can type in 100 permutations this time and leave all other options the same…), or you can upload the log file created from this step and just let it run a certain amount of permutations and kill the process.

<Calculate Threshold: svm_thresCalc.m> New since alpha-1-0-7.

1. ***PERMUTATION LOG FILE***

Select the log file created from the permutation analysis if you want to test whether the classifier is significantly better than chance. Select two log files if you want to compare performance of two classifiers. When doing the latter, choose between Kolmagorov-Smirnov 2-sample ttest and delta performance test (for this, you should have matched the randomized class labels during permutation analysis).

2. ***OUTPUT***

xxx_permutSumOutput.mat: All of the following parameters will be saved. Also it will be printed out in the terminal when this process completes.

PERMUTSUM .coeffThres: average values of xxx_permutCoeffThres.txt (from step 5 above).
PERMUTSUM.pvalPrim: p value of your primary SVM. If you want to compare between classifiers and not whether it is significantly better than chance, then you currently have to calculate this yourself (though I should be able to code it in several minutes when I get a chance).
PERMUTSUM.pvalAppl: (if you performed the Apply Classifier analysis) p value of your applyClassifier performance.

XXX_permutSumOutputCompare.mat: This file gets created when you compare between different models. All of the following parameters will be saved. Also most of these will be printed out in the terminal when this process completes.

PERMUTSUM.classifier_name_a(b): 1st classifier permutation log file ('_b' stands for the 2nd classifier)
PERMUTSUM.pvalPrim_a(b): P values indicating whether the 1st classifier is significantly better than chance.
PERMUTSUM.primAccBestIterNum_a(b): (if you ran Fixed Target RFE, then) It will tell you which iteration of RFE gave you the best performance.

PERMUTSUM.primAccBestPerform_a(b): (if you ran Fixed Target RFE, then) It will tell you the best performance accuracy.
PERMUTSUM.primAccBestPval_a(b): (if you ran Fixed Target RFE, then) it will give you the p value corresponding to the best performance accuracy.
PERMUTSUM.compClassifiersOpt: (if you compared 2 classifiers) it will tell you which option (KS or delta) you used to calculate significance.
PERMUTSUM.pvalPrimCompClassifiers: Statistical significance of the comparsison between the two classifiers.

12. RECONSTRUCTING BRAIN MAPS

<Matrix 2 Image: mat2img.m>

You would almost always want to do this if you used brain maps for classification so you can see pretty (or not so pretty) pictures at the end.

1. ***PREP***
   Cd (change directory) into the directory where your SVM results resides. Start SVM (type 'SVM' without the '' in the terminal) and click on 'Matrix 2 Image'.
2. ***SELECT OUTPUT.MAT FILE***
   Select output.mat file.
3. ***SELECT TEMPLATE FILE***
   Select the template file you used to initially create the matrix.
4. ***FORMAT OF TEMPLATE/MASK?***
   If you used the Downsample option, then this will be the default 'int16'.
5. ***DID YOU USE PCS AS ORIGINAL MAT4SVM FILE FOR PRIMARY SVM?***
   Choose 'Yes' if you used PCs.mat file. If 'Yes', it will use the original eofs.mat file to reconstruct the PCs back into original space. If not, choose 'No.
7. ***OUTPUT***
   xxx_avg.hdr / .img: Average of weights from the cross-validation will be reflected in each voxel value.
   xxx_effectSize.hdr / .img: Mean/Std from the cross-validation will be reflected in each voxel value.
   xxx_effectSizeAbs.hdr / .img: Absolute values of Mean/Std from the cross-validation will be reflected in each voxel value.
   xxx_freq.hdr / .img: Frequency (how often that feature [voxel] was selected during the cross-validation procedure) will be reflected in each voxel value..
   These can be displayed using for example MRIcroN.

13. VISUALIZATION USING SELF-ORGANIZING MAPS (c.f. Formisano et al. Science '08)

<SOM for Visualization: somVis.m>

This is very preliminary. It also does not give you any really new information but is simply a visual representation of the results. Tends to work better when you use RFE data. Script could be buggy, let me know if it is. Also SOM toolbox that

Formisano uses may be better and I might replace this in the future (current version uses the one in matlab).

1. ***PREP***
   If you have more than 2 groups, i.e., if you did applyClassifier, then ideally you would have done SVM where all data were concatenated. If not, we have to tweak the code a bit.
   You will also need to make a new class file where for example, 1s for the 1st group, 2s for the 2nd group, 3s for the 3rd group, etc….
   Cd (change directory) into the directory where your SVM results resides. Start SVM (type 'SVM' without the '' in the terminal) and click on 'SOM for Visualization'.
2. ***SELECT PRIMARY SVM LOG FILE***
   Select log file created during primary SVM analysis file you used in primary SVM analysis.
3. ***SELECTED VOXELS FROM RFE OR L1?***
   If you did RFE or L1, then it looks for normalw to see which features had weights of 0 and will ignore those features from SOM analysis.
4. ***HOW MANY CLASSES?***
   Type in how many classes (different populations) you have.
5. ***SELECT NEW CLASS FILE***
   Select the new class file you just created.
6. ***PERFORM PCA?***
   If you have lots of features (e.g. did not use PCA as the input matrix) then you probably want to choose the PCA option. It will display up to 3 PCs in 3d. If you used PCs.mat file as the mat4SVM file, then you can choose the 3rd option where it used the 1st 2 vectors (PCs) to display the results. The matlab version of SOM is not great and may be best to where PCA is done after SOM is performed.
   Note: Keep in mind that running the results each time could results in different results each time, especially if the solution is unstable.

14. OTHER NOTES [will add more as I go]

1. HOW TO MODIFY LOG FILES.
   In the matlab command window:
   Load an old log file (e.g. Type 'load results_log').
   See what variables you have (e.g. Type 'whos').
   Replace variables (e.g. Type 'inputDataFile_name='/Users/fumiko/Desktop/Psyc250/mat4SVM.mat'').
   Save log (e.g. Type 'save results_log').
   Next time you run SVM, then say that you have log file already and choose this newly saved log file.
   If you want to know the options, to type 'help svm_gui' without the '' or open this file svm_gui.m.

2. RFE OPTIONS (regular, nested, fixed target RFE)
   Nested and fixed target RFE is not terribly different anymore. Fixed target RFE just saves more stuff so if you can afford space, then this option is always recommended. Basically it does nested RFE (LOO to determine features based on the performance of the left-out subject, then LOO to test final performance once the features are determined) rather than regular RFE where LOO is done only once (features are selected just by taking cutting off the bottom X% of features with lowest weights and not based on performance). Obviously regular RFE is much faster.