Workshop 6: Support Vector Machines

The aim of this week's workshop is to use support vector machines, compare different kernels and visualise the decision boundaries.

You will use **Wisconsin Breast Cancer data set (https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html#sklearn.datasets.load_breast_cancer)**, which is included in scikit learn.

Start by using the **template notebook supplied here (https://myuni.adelaide.edu.au/courses/71744/files/10693808?wrap=1)** ↓ (https://myuni.adelaide.edu.au/courses/71744/files/10693808/download?download_frd=1) .

1. Split the data into train, validation and test sets using a 70:15:15 ratio.
2. Build an SVM classifier, in a pipeline, with a radial basis function (rbf) kernel, default (hyper)parameters and determine the accuracy of this classifier on the validation set.
3. Now we want to establish a simple baseline to compare accuracy values, much as we did in the earlier workshop on regression. Here we will do it for each feature separately, turning the feature values into a simple “prediction probability” by using the formula: y_pred = (x-xmin)/(xmax-xmin) which gives values in the range 0 and 1 (inclusive).

   Start with the first feature and write a loop to threshold this prediction (y_pred) at a set of evenly spaced values between 0 and 1, and for each threshold calculate the accuracy. From this determine the maximum accuracy (across all your tested thresholds) for that feature.

   Now do this for each feature in turn. Which feature gives the best accuracy and how does this compare to the SVM result? What is the name of this feature?
4. The value of y_pred can also be used to plot ROC curves and calculate AUC. Do this for two features: the one that has the highest accuracy and the one that has the lowest accuracy. How do the ROC curves look, and what is the range of AUC values?
5. Choose the two features with the best accuracy scores from step 3. Use the code provided (*make_meshgrid* and *plot_contours*) to plot the decision boundaries of the SVM classifier from step 2 [note that the code provided here is a modified version of one of the **scikit learn examples (https://scikit-learn.org/stable/auto_examples/svm/plot_iris_svc.html#sphx-glr-auto-examples-svm-plot-iris-svc-py)**]. We will plot these decision boundaries using the original data, so pass in the pipeline to the plot_contours call, not just the classifier part. You will need to choose suitable ranges for *make_meshgrid*, based on the original feature values.
6. Now display a scatter plot of the training data points, in different colours for the two classes, on

top of the decision boundary. Also add a scatter plot of the validation data points using the same colours but different symbols [hint: use marker='s' to get squares].

7. Rerun the SVM classification with polynomial ('poly') and linear ('linear') kernels. Compare the results in terms of accuracy and plots of the decision boundaries.
8. Choose the best classifier and report the results on the test set. Check to see how different it is from the validation set result.