**Exercises Week 3**

1. For this exercise it is encouraged that you will install the python package [matplotlib](https://matplotlib.org/index.html). This package contains advanced plotting functions – we will not use many of those but knowing your way around in matplotlib is quite useful.
2. Load the data in ‘mixture.csv’ that we used in Week 1. Using the sklearn library, train a LinearRegression that predicts the class label 0 or 1. Load the test data ‘mixture\_test.csv’ (also from week 1) and make predictions for each point. Since we are using a regression model (as opposed to a classification model), our predictions will be below 0 and also above 1. Thus, like in the lecture, we map values class 1 and values to 0. Plot the class predictions (using matplotlib).
3. Compute the decision boundary learned by the linear regression – you will need .coef\_ and .intercept\_ to access the coefficients. Create a plot with the class predictions (as in (2)) and add your decision boundary to the plot.
4. Load the dataset ‘adni\_conversion\_train.csv’ (week 3). This dataset contains baseline information of patients diagnosed with Mild Cognitive Impairment (MCI). The outcome variable is ‘*conversion’* and it indicates whether or not the particular patient has converted to Dementia at the 2 year check-up (this is the class label we want to predict). There are a few given features (X) and they are similar to the ones used in the Week 2 exercise. Train a LogisticRegression model with the *elasticnet* penalty to predict the conversion label from the features. Set the L1-ratio to 0.5, but you will still need to find a good setting for the regularization “C”. (Hint: You can use the built-in cross-validation function LogisticRegressionCV to find good parameters for C. This will also require a scoring function. Since this problem is not balanced [i.e., fewer converters than non-converters] suggestions are ‘roc\_auc’ or ‘balanced\_accuracy’ as to the default setting, which is simply ‘accuracy’). Which *C* parameter was selected and what are the -coefficients of the trained model?
5. Make predictions for the data in the test set: ‘adni\_conversion\_test.csv’. Using the functions in [sklearn.metrics](https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics), compute a confusion matrix, the area under the ROC curve and the balanced accuracy.
6. Challenge 1: Instead of using the built-in CV function (LogisticRegressionCV), use more general grid search function (GridSearchCV) to find a good setting for C.
7. Challenge 2: Select 70 samples (at random) from the training data (hint: can be done using different ways, most popular are either random.sample() or sklearn’s train\_test\_split function). Redo steps 4 (or 6) and 5. How does the model perform compared to the one trained on the full data? How did the model parameters change? Is the model stable if you sample another 70 subjects from the training data?