Programming Assignment

Submission guidelines:

- Download the supplied files from Moodle. Details on every file will be given in the exercises. You need to update the code only in the skeleton files, i.e., the files that have a prefix "skeleton". Written solutions, plots and any other non-code parts should be included in the written solution submission.
- Your code should be written in Python 3.
- Make sure to comment out or remove any code which halts code execution, such as matplotlib popup windows.
- Your code submission should include the file `svm.py`.

1. **(25 points) SVM.** In this exercise, we will explore different kernels for SVM and the relation between the parameters of the optimization problem. We will use an existing implementation of SVM: the SVC class from sklearn.svm. This class solves the soft-margin SVM problem. You can assume that the data we will use is separable by a linear separator (i.e. that we could formalize this problem as a hard-margin SVM). In the file skeleton `svm.py` you will find two implemented methods:

   - `get_points` - returns training and validation sets of points in 2D, and their labels.
   - `create_plot` - receives a set of points, their labels and a trained SVM model, and creates a plot of the points and the separating line of the model. The plot is created in the background using matplotlib. To show it simply run `plt.show()` after calling this method.

   In the following questions, you will be asked to implement the other methods in this file, while using `get_points` and `create_plot` that were implemented for you.

   (a) **(5 points)** Implement the method `train_three_kernels` that uses the training data to train 3 kernel SVM models - linear, quadratic and RBF. For all models, set the penalty constant C to 1000.
   - How are the 3 separating lines different from each other? You may support your answer with plots of the decision boundary.
   - How many support vectors are there in each case?

   (b) **(10 points)** Implement the method `linear_accuracy_per_C` that trains a linear SVM model on the training set, while using cross-validation on the validation set to select the best penalty constant from $C = 10^{-5}, 10^{-4}, \ldots, 10^{5}$. Plot the accuracy of the resulting model on the training set and on the validation set, as a function of C.
   - What is the best C you found?
   - Explain the behavior of error as a function of C. Support your answer with plots of the decision boundary.

   (c) **(10 points)** Implement the method `rbf_accuracy_per_gamma` that trains an RBF SVM model on the training set, while using cross-validation on the validation set to select the best coefficient $\gamma = 1/\sigma^2$. Start your search on the log scale, e.g., perform a grid search $\gamma = 10^{-5}, 10^{-4}, \ldots, 10^{5}$, and increase the resolution until you are satisfied. Use $C = 10$

as the penalty constant for this section. Plot the accuracy of the resulting seperating line on the training set and on the validation set, as a function of γ.

- What is the best γ you found?
- How does γ affect the decision rule? Support your answer with plots of the decision boundary.