Problems:

1. **(Image Compression using K-means)** You are given an RGB image. Consider each pixel value as a 3-dimensional feature. Use KMeans and represent each pixel by the centroid color.

   (a) Implement a function – *computeCentroid*, that takes n 3-dimensional features and returns their mean. **(2 pts)**
   (b) Implement a function – *mykmeans* from scratch that takes data matrix X of size $m \times 3$ where m is the number of pixels in the image and the number of clusters k. It returns the cluster centers using the k-means algorithm. **(3 pts)**
   (c) Use the centroids of k-means to represent the pixels of the image. Now, show compressed images for different values of k. **(1 pts)**
   (d) Show the results of compressed images using the k-means implementation of the sklearn library. What differences do you observe? **(2 pts)**
   (e) Spatial coherence: Incorporating spatial information helps maintain spatial coherence in the compressed image. Pixels that are nearby in the original image are more likely to be assigned to the same cluster, preserving local structures and reducing artifacts like color bleeding or noise. How do you implement spatial coherence? Write the idea, implement it, and write down your observation. **(2 pts)**

2. In this problem, you will explore the decision boundaries of Support Vector Machines (SVM) using different kernels, and also learn about tuning its hyperparameters.

   Task-1(a): Load the Iris dataset using the following code:

   ```
   from sklearn import datasets
   iris = datasets.load_iris(as_frame=True)
   ```

   For this problem, focus only on the petal length and petal width features. Create a new dataset by selecting only two classes, 'setosa' and 'versicolor' for binary classification. Normalize the dataset, and split it into train and test by choosing an appropriate split ratio. **(2 pts)**

   Task-1 (b): Train a Linear Support Vector Classifier (LinearSVC) on the training data. Plot the decision boundary of the model on the train data, and also generate another plot showing a scatterplot of the test data along with the original decision boundary. **(3 pts)**

Task-2 (a): Generate a synthetic dataset using the *make_moons*() function from scikit-learn. Take around 500 data points, and add 5% noise (misclassifications) to the dataset. **(1 pts)**

Task-2 (b): Implement SVM models with three different kernels: Linear, Polynomial, and RBF. Plot the decision boundaries for each kernel on the synthetic dataset. Analyze and comment on the differences in decision boundaries produced by these kernels. **(5 pts)**

Task-2 (c): Focus on the RBF kernel SVM model. Perform hyperparameter tuning to find the best values of gamma and C for this model. You can use techniques like grid search or random search. **(2 pts)**

Task-2 (d): Plot the decision boundary for the RBF kernel SVM with the best Hyperparameters. Explain the impact of the selected gamma and C values on the model's performance and decision boundary. Note: Ensure to complete each task thoroughly and document your findings in the lab report. **(2 pts)**

References: SVM Documentation(sklearn): Link - `https://scikit-learn.org/stable/modules/svm.html`

Interactive Demo - SVM: It is an interactive SVM visualizer. Link: `https://greitemann.dev/svm-demo`

Initially, one needs to manually add data points to the graph and then choose various values of hyperparameters (C, gamma, kernel, etc.). After that, the tool outputs the decision boundary and also highlights the support vectors.

Rubrics:
Task completion with proper documentation/comments and variable naming: 25 Points
Showing working code during lab: 5 Points
Report: 20 Points

End of Paper