Submission Instructions: Please submit all the materials, including codes, in a single file (a single tar file is preferred). In the codes, please add all the necessary comments so that the grader can follow what you are doing. Otherwise, if the grader does not understand your codes, you may unnecessarily lose points.

The goal of this project is two-fold: First, you will have an opportunity design several well-known binary classifiers – (a) decision tree, (b) k-nearest neighbors (kNN), (c) linear discriminant analysis (LDA), and (d) support vector machine (SVM) – using real world data collected from loan applications and compare their performance. Second, you will study how the dimensionality reduction achieved via the principal component analysis (PCA) affects the performance of some of these classifiers as you change the number of principal components you use to create new features (as linear combinations of original features).

The data for the project is provided in two separate files; the samples in `'TrainingData.csv'` will be used for training your classifiers, and the samples in `'TestingData.csv'` will be used to evaluate their performance. Each file contains the values of 27 features (Age, Annual Income, etc. in columns 1 through 27), and the label (LoanApproved in column 28 - 0: approved, 1: denied).

The training data in the first file contains 450 applications that were approved and 450 applications that were denied with a total of $n = 900$ samples. Similarly, the testing data includes 200 approved applications and 200 denied applications.

1 Binary Classifiers: Part 1 - Original Features

In the first part of the project, you will use the samples in `'TrainingData.csv'` to train the following four (4) classifiers. You will evaluate the performance of each classifier by considering two different types of errors for the testing data: a type 1 error happens when your classifier chooses 'Denied' for an application that was 'Approved' in the testing data. A type 2 error occurs when your classifier selects 'Approved' for a 'Denied' application in the testing data. For each classifier you design, using the testing data, compute the type 1 error rate (which is the number of type 1 errors divided by 200) and the type 2 error rate (which is the number of type 2 errors divided by 200).

Some basic instructions are provided below. However, you are allowed to choose the algorithms used for these classifiers. For example, for a decision tree, you can use any of the following algorithms – ID3, C4.5, CART (classification and regression tree). Please make sure to set them up correctly when using built-in algorithms/functions and explain which algorithms are used in your code (this can be embedded in your codes as comments).

1. Linear discriminant analysis - For this binary classifier, you first need to find a suitable direction $\mathbf{w} \in \mathbb{R}^{27}$ so that the projections of the samples can be used for classification. In other words, if $\mathbf{x}^k$ is the k-th sample in the testing data and y^k is the coefficient for its projection obtained using the vector $\mathbf{w}$ you find from the training data, your new features are $y^k = \mathbf{w}^T\mathbf{x}^k$, $k = 1, \ldots, 400$, and the classification of the k-th sample is performed using y^k (instead of $\mathbf{x}^k$) by selecting a suitable threshold on the new features. Plot both type 1 error rate and type 2 error rate as you vary the threshold.

2. Decision tree - For designing a decision tree, you can either use a built-in function or (if you want to challenge yourself) identify the attribute or question at each decision node using Gini impurity or

information gain (e.g., `Is Annual Income > $40k?`). If you take the latter approach, make sure to provide the set of questions you use for your decision tree at the decision nodes.

3. k-nearest neighbors - Evaluate the performance of kNN algorithm for $k = 1, 3, 5, 10$.

4. Support vector machine - The provided training data is not (linearly) separable, i.e., there is no separating hyperplane (short of using other techniques). Thus, you will need to use soft margin. If you use a built-in function, provide the formulation/optimization problem used for the classifier (which can be found in the accompanying documentation). If you design your own SVM, provide the objective function used for selecting the hyperplane.

2 Binary classifiers: Part 2 - Use the PCA to Design New Features

In the first part of the project, you were asked to use the original 27 features in the training data for training the classifiers (and the same features for testing). In the second part of the project, you are asked to design both the kNN classifier and the SVM based on the new features you obtain after applying the PCA. Let K be the number of principal components you use to approximate the training samples (hence the number of new features). For $K \in \{5, 10, 15\}$, design both kNN and SVM classifiers using K new features (which are the coefficients in the approximations). Compare the performance of the new kNN and SVM classifiers to that of the classifiers in the first part.