Components:

* Materials:
  + Titanic dataset from Kaggle
* Functions:
  + kNearestN(dat, lab, test, K) computes the predicted labels of test based on memorized dat and lab by calling EuclidDistance to compute the distance of each neighbor to determine the predicted class label. Returns list of predicted test labels.
  + EuclidDistance(pt1, pt2) computes and returns the square root of the sum of squared difference between 2 points.
  + selfKNNAccuracy(pred, true) computes and returns the accuracy measurement between the predicted labels pred and the true labels true.
  + run\_SVC(dat, lab, D, dLab, tst, tstLab) executes a SVC classifier and computes the prediction and accuracy metric of the SVC.
  + run\_LinReg(dat, lab, D, dLab, tst, tstLab) executes a linear regression classifier and computes the prediction and accuracy metric of the linear regression.
  + run\_KNN(dat, lab, D, dLab, tst, tstLab) executes a K-Nearest Neighbors classifier and computes the prediction and accuracy metric of the K-Nearest Neighbors.
  + class TitanicData() and def import\_clean\_TitanicData(self) imports, cleans and coverts titanic dataset into numpy arrays of development set, test set and train set. Note the distribution of class labels is not consistent w/in each set as well as w/ the original set thus why there is a variability in accuracy metric reporting after each execution of classifier.

1. The LinearRegression method has default parameters fit\_intercept = True, normalize = False, copy\_X = True and n\_jobs = None, C =1 [Where C controls the width of the margins]; with the default parameters, the accuracy returns to be 69.1729323308%. Thus, with my previous assignment which used a perceptron, I was able to achieve make about 80% accuracy, in comparison, this LinearRegression classifier with default parameters is not the most effective classifier to use at this time.
2. The LinearRegressions method with C = 100 returns an accuracy: 77.44360902255639%, and when ran again returns 75.18796992481203%. In comparison, when changing the parameter C to C = 5, the accuracy is 81.9548872180451%, and when various times returns 80.45112781954887% and 84.9624060150376%. With C = 5, LinearRegression meets a max and desirable accuracy between 80% and 85%. Thus, modifying the parameter value can lead to increased accuracy within linear regression classifier with this particular dataset.
3. Using my own KKN implementation, I was able to achieve Accuracy 82.70676691729323% w/ K = 18, 81.95488721804511% with K = 50 and 68.42105263157895% with K = 3.
4. Running Logistic Regression SkLearn method and my own KNN classifier on the test data set, the results are ---note this was ran various times for comparison--- (Logistic Regression Accuracy 83.5820895522388%, self KNN Accuracy 84.32835820895522%), (Logistic Regression Accuracy 73.88059701492538%, self KNN Accuracy 75.3731343283582%), (Logistic Regression Accuracy 75.3731343283582%, self KNN Accuracy 76.11940298507462%). Therefore, my created KNN classifier has consistently better accuracy in comparison to the SkLearn classifier with the test dataset.