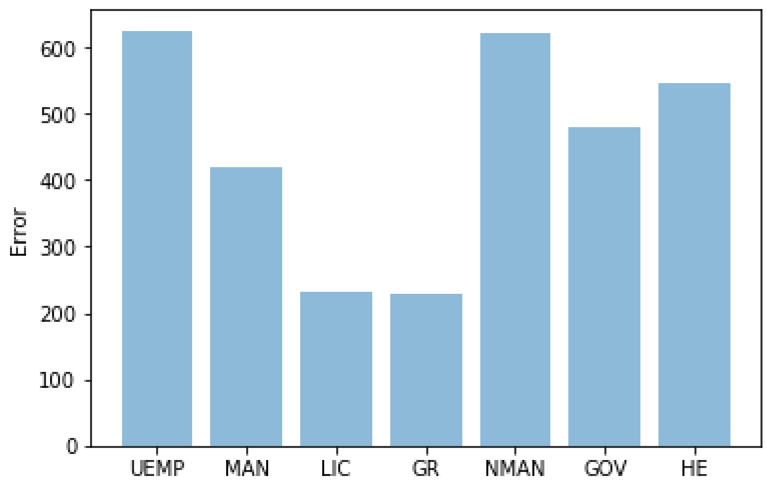
Linear Regression:

I used Least squares to solve this problem. I first arranged the data so that the variable matrix contained tuples of data, such that matrix[0] contained all tuples of FTP, WE, and a third column, with the restriction that the data was in the same row. This led to a 13 x 7 matrix. Now for each row in the main matrix (7), we get the average phi for each selected column (FTP, WE, x), as well as the average t. Now we use the equation 3.15 to find the w vector for the selected columns, and equation 3.18 to find the associated error. Because we do this for each row in the matrix, we find the w vector and the error for each tuple of columns, including FTP and WE because those are known as good predictors. We then compare the error of each tuple to find the lowest error, which would be the column that is the 3rd best predictor, in this case GR. Because lower error means that your predictions are more correct, I would think this makes GR the 3rd best predictor, after the given FTP and WE, which we factored into our regression by including them in our tuple. The calculated errors are summarized into this graph.



KNN:

For this problem I defined the functions as described in the prompt. First I read in the data with pandas.read_csv, and imputed missing values either by taking the mode of the column, in the case that it was a categorical value, or by taking the value adjusted mean, in the case of a continuous variable. Missing data in the testing set was imputed by the mode or mean of the training set. Next I normalized the values using the formula given. Specifically, I joined the columns of both training and testing sets to get the mean and the std, and then normalized both columns together. Distance was calculated as defined, if the column was categorical, distance was either 1 or 0, and if not it was the l2 distance. And then we sum all these values together to get one scalar. In the predict function for each row in the testing set, we generate a distance for each row in the training set. We then take the n-smallest rows (depending on the k passed in), and find the mode of the label, and add it to a list. Accuracy was also simply defined, checking if a column of values was equivalent to the predictions, adding 1 if it was, and then dividing by the number of values.

The total results can be summarized in this table:

For the crx data set:

K value	5	9	13
Accuracy	117/138 = .848	117/138 = .848	119/138 = .862

For the lenses data set (K had to be smaller because there were only 18 values in the training set):

K value	3	5	7
Accuracy	5/6	5/6	5/6

For these results it is important to note that even if the accuracy is the same percentage, we could have added one correct result, and removed one different previously correct result, to end in the same total percentage.