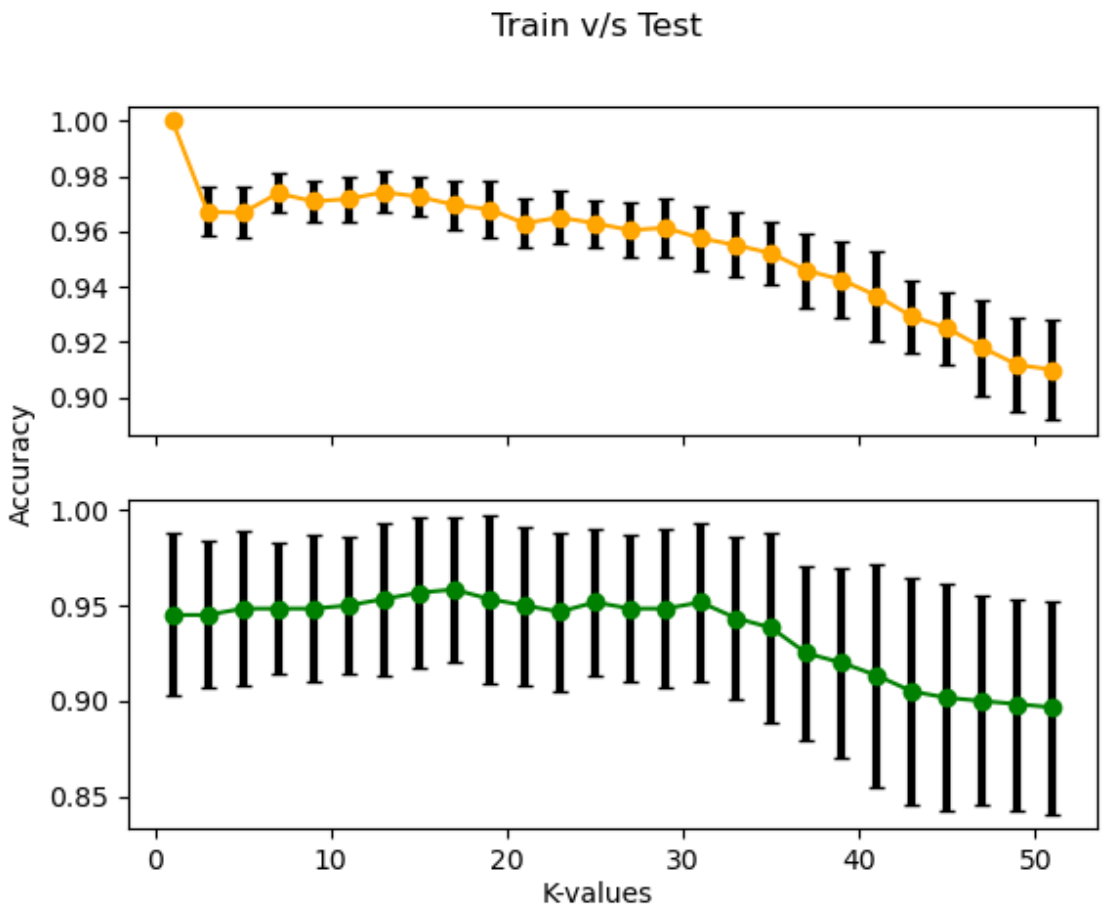
KNN –

Q1.1) **First** graph shows the KNN model trained over the **training** set.

Q1.2) **Second** graph shows the KNN model over the **testing** set.



Q1.3) In the first graph (over training set); the graph starts with 100% accuracy at K=1 because the nearest neighbor to a point is itself. This is most likely not going to yield good results on new instances. The graph first ascends up to K=7. This is because these are the optimum values of K which are used to train the data and thus yield good accuracy measurements. The graph is then steady for a while (say up to K=23) before going on a downward slope. This is because large K values are making the model underfit the data. The length of the error bar is also increasing with larger k values because there is a large variation in accuracy readings taken.

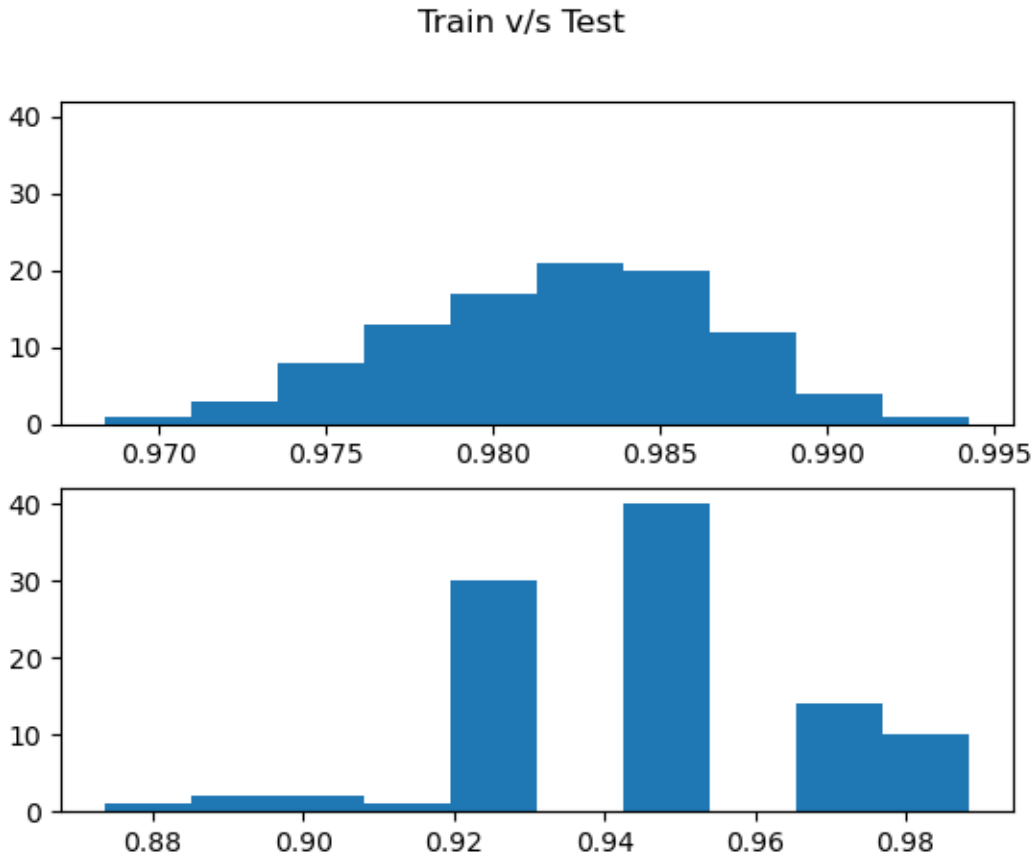
In the second graph, the graph is on an ascending slope till k=17, this is because we are finding the correct value of K for the model and the trained model is accurately predicting the classes for new instances of test data. The error bar is larger than for the training data because test data hasn't been encountered before and there are large variations in prediction. Similar to the training graph, the model produces low accuracies for higher values of k due to underfitting. The deviation bar is also increasing in length with increasing k value because of variations in the accuracy measurements.

Q1.4) I think the model does fairly well over most instances of K. I would say that for range **k = {33,…,51}** the model is **underfitting** because the accuracy of the training model is constantly reducing with increasing values of K. This also reflects on the performance graph using the testing data as we see that the prediction accuracy for test data also reduces with increasing k values from **k = {33,…,51}.**

I think the model is clearly **overfitting** for **K=1** as the training accuracy differs greatly from the testing accuracy. All other values of K, the model seems to be making reasonable predictions.

Q1.5) I have 2 values of K in mind. K=7,17. At **K=7** the testing accuracy measurements seem to have to lowest standard deviation and also a high mean accuracy. At **K=17**, the graph shows highest accuracy over the testing set. Also the accuracy of the model over the training set at **K=17** is sufficiently high and there isn't a huge difference in accuracy rate between the models' predictions over the training data versus the testing data.

Q2.1) First histogram shows accuracies of DecisionTree model over **training** set.
Q2.2) Second histogram shows accuracies of DecisionTree model over **testing** set



```
Mean Train -> 0.9820689655172414
Standard Deviation Train -> 0.0052061479759279335
Mean Test -> 0.9436781609195403
Standard Deviation Test -> 0.022199089558422955
```

Q2.3) I think the **training data histogram** is as showing as I expected. Since we are modeling our Decision Tree on the training data, there is high accuracy and there are lots of very similar high accuracy measurements across the 100 runs. Hence the variance is also very low across all the readings.

The **testing data histogram** is also very accurate but doesn't match up to the training. The variance is also slightly higher but still low. This could be because of the data distribution for some of the 100 models that we trained. For the current dataset given, one single attribute – physician_fee_freeze has very high information gain and this could affect the models predictions.

Q2.4) I think based on the **training data**, the model seems to be **overfitting**. From the training histogram, we see that the minimum accuracy for the model on the training set is 97%. Also, there is little variation and the results are concentrated around the 98% mark.

As per the model results on the testing data, although there are some accuracies recorded in the 97-97%. Fairly large amount of them are in the 94-95% mark which is significantly lower than highest occurring training accuracy recording.

Q2.5) We can see the Non-Robust property of the DecisionTree algorithm in display here.
An example is the attribute physician_fee_freeze. This attribute gives very high information gain. If we compare the values of physician_fee_freeze to the Target label, we notice that for most of the dataset if Physician_fee_freeze = 1 ; then Target = 0. However, there are a couple of rows for which the Target is – 1. Now when we shuffle the data, if one of these rows with Target = 1 for physician_fee_freeze = 1 does not end up in the training set, then we will get a different classification from when it does end up in the training set. There could be other attributes similarly. This is one of the reasons for a lower mean testing accuracy rate. I believe this is also why we see a higher variance for the testing histogram as opposed to the training histogram.