1. INTRODUCTION :

K-nearest neighbors algorithm is a classification algorithm that does not involve learning weights. It can be used for both classification and regression. The basic idea is to find k nearest data points (training points) for every test point, get the posterior probability (Probability of the cluster belonging to a class) and use this probability to predict the class.

2. TRAINING AND TESTING :

Since we have different training and test set, for every test sample proximity to all other training samples is calculated. Using these proximities, k nearest neighbors are determined and class is predicted for every test sample using posterior probability. Posterior probability P(+|x)/P(-|x) is calculated as a proportion of k nearest examples that are in class +/-. Equal weights for k nearest neighbors is assumed for simplicity. But, it makes more sense to assign different weights to k samples as they are at different proximities to the test example in question.

Alternative : Combined the training and test data, and used cross validation on the entire data set. Eight fold cross validation is used. Although this does not give you the accuracy for the test set alone, it is useful to see the potential of your classifier.

3. PERFORMANCE :

3.1 Proximity measures used and performance : Jaccardian for capital_loss and capital_gain, Manhattan for age,fnlwgt and hour_per_week. Simple comparison and ordinal for the rest.

Performance is measured for different classifiers (different values for k from 1 to 50). For each classifier, confusion matrix (TP, FP, TN and FN), classification accuracy, misclassification error (MSE) is calculated. The maximum accuracy observed is 77.77% for k = 7. Accuracy increases from k = 1 to k = 7, decreases after k = 7 and remains

constant after k = 18. Fig 3.1 shows the plot of classification accuracy and k. Positive class throughout this report is <=50K.

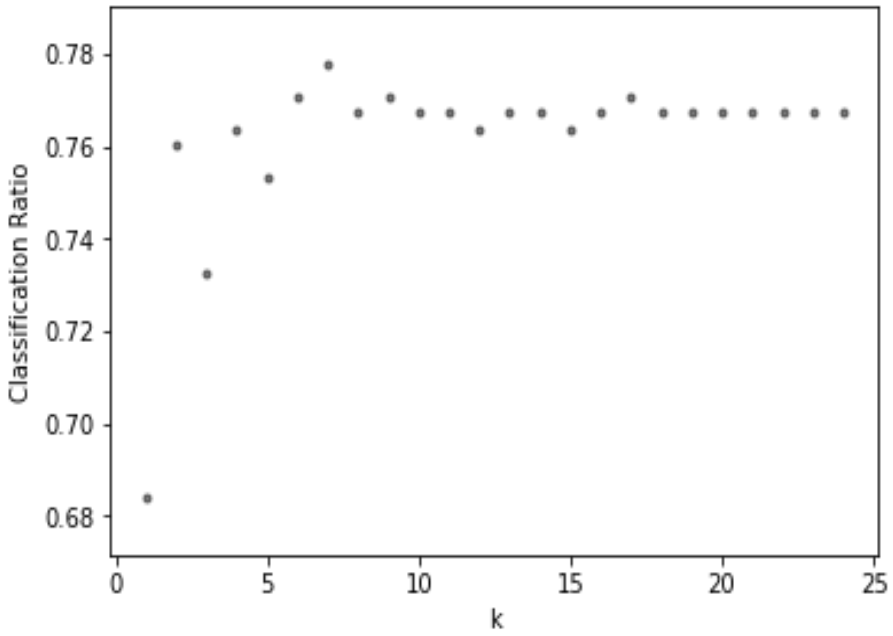


Fig 3.1 : Classification Ratio for different k's without weighted posterior Probability

Confusion Matrix for k = 5 :

	<=50K	>50K
<=50K	211	10
>50K	61	6

Accuracy : 75.34 % Precision : 0.77 Recall : 0.95
TP rate = 0.954 FP rate = 0.91
F-measure : 0.855

Confusion Matrix for k = 7 :

	<=50K	>50K
<=50K	220	1
>50K	63	4

Accuracy = 77.77 % Precision : 0.77 Recall : 0.99
TP rate = 0.995 FP rate = 0.94
F-measure : 0.87

Confusion Matrix for k = 9 :

	<=50K	>50K
<=50K	220	1
>50K	65	2

Accuracy = 77.083 Precision : 0.77 Recall : 0.99
TP rate = 0.995 FP rate = 0.97
F-measure = 0.86

True Positive rate and False Positive rate are calculated as TP/(TP+FN) and FP/(FP+TN) and is used to plot ROC for different k's . Fig 3.2 shows the ROC. High False Positive rate is observed for every k. This is because there are 420 examples in the training set that belong to one class and 100 in other class. So, in each cluster majority of the examples fall in the dominant class. This can be observed in the above confusion matrix.

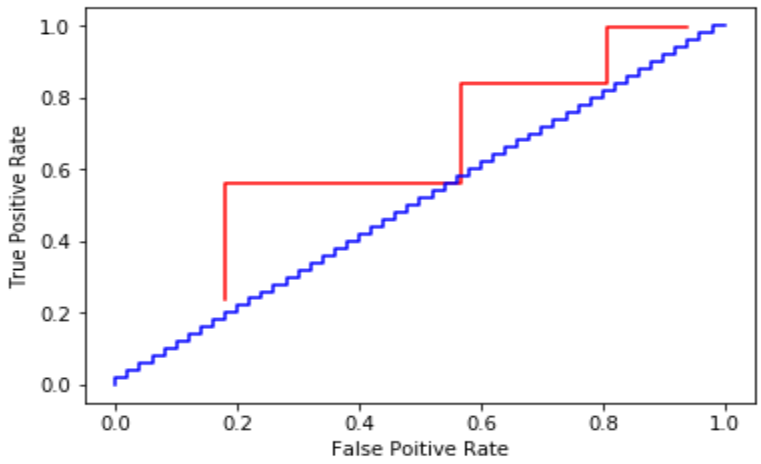


Fig 3.2 : ROC curve for knn without weighted posterior probability and k = 7

Precision and Recall :

There is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other. Precision and recall are not used separately to measure the performance. They are often combined. For Ex: F-measure.

F-measure :

It is the weighted harmonic mean of precision and recall. F - measure has best value of 1 and worst value of 0.

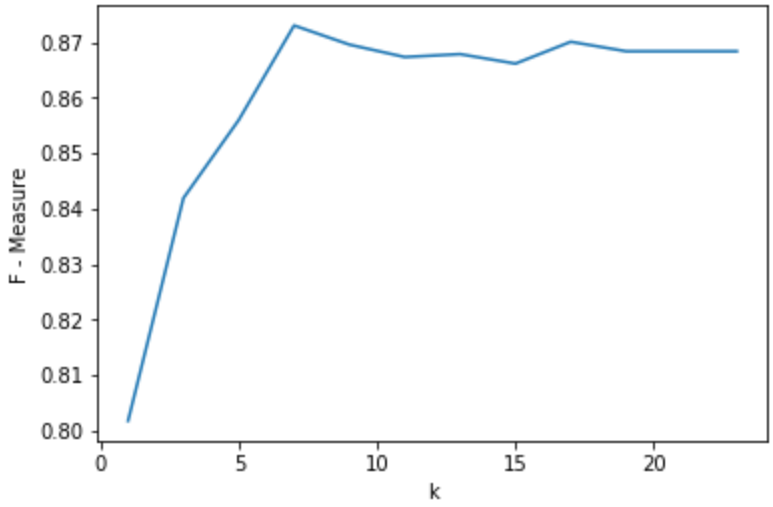


Fig 3.3 : F - measure

From figure 3.3 F - measure reaches its peak at k = 7.

True Positive and False Positive :

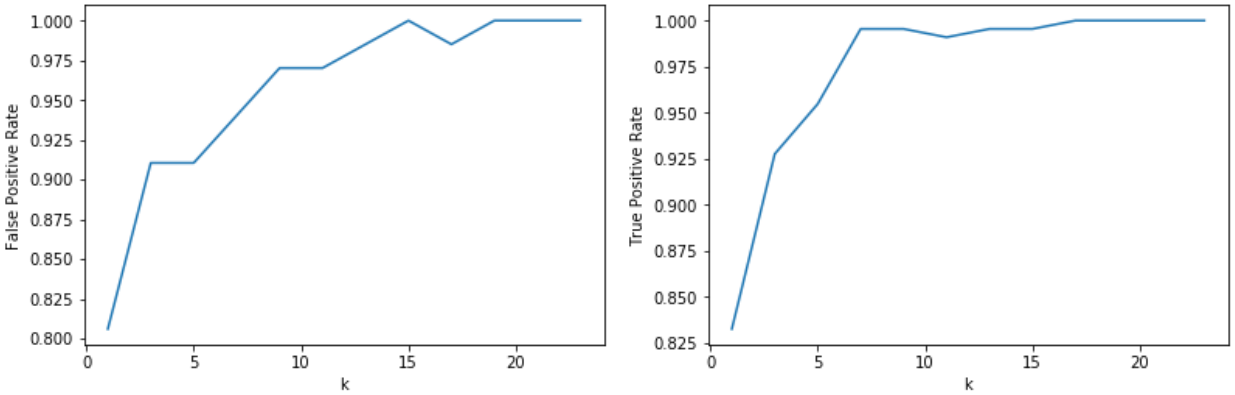


Fig 3.4 : False Positive and true positive vs k

True positive is peak at k = 7, but false positive is pretty high. Although F - measure is high at k = 7, the fact that FPR is pretty high makes knn a bad classifier for the proximity measures used.

Another way of looking at the above measures would be to take the weighted average of these measures for both class. Let's not consider this for simplicity.

Recommended value of k for this dataset is 7 because of the high accuracy and good F - measure value.

3.2 Changing the proximity measure : The proximity measure for capital_gain and capital_loss is now changed to simple comparison similarity. And the results observed are :

Confusion Matrix for k = 7 :

	<=50K	>50K
<=50K	219	2
>50K	64	3

Accuracy = 77.08 Precision : 0.773 Recall : 0.99
TP rate = 0.995 FP rate = 0.97

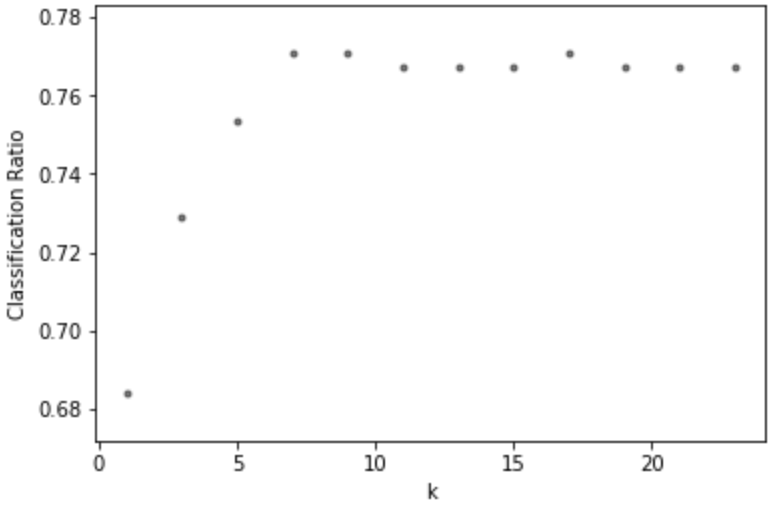


Fig 3.5 : Classification ratio for different proximity measure

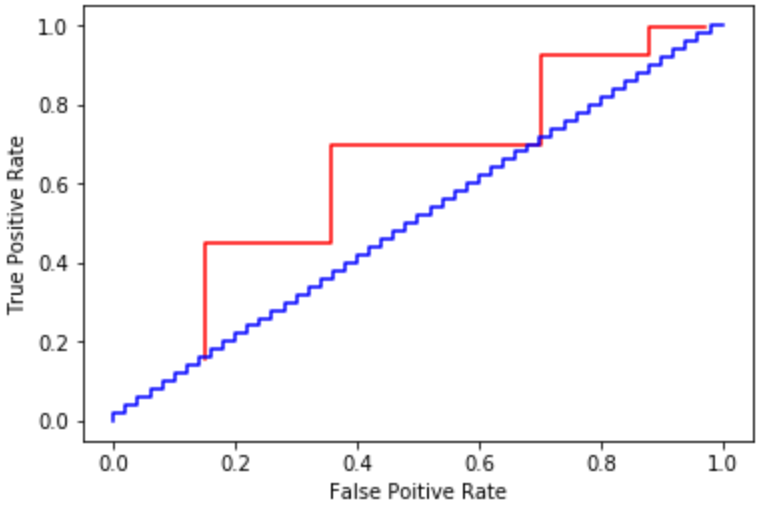


Fig 3.6 : ROC for different proximity measure and k = 7

This is very similar to previous case. **Changing the proximity measure did not change the performance significantly.**

3.3 Alternative :

Cross Validation :

Confusion Matrix for k = 9 :

	<=50K	>50K
<=50K	398	22
>50K	58	42

Accuracy : 84.6 %

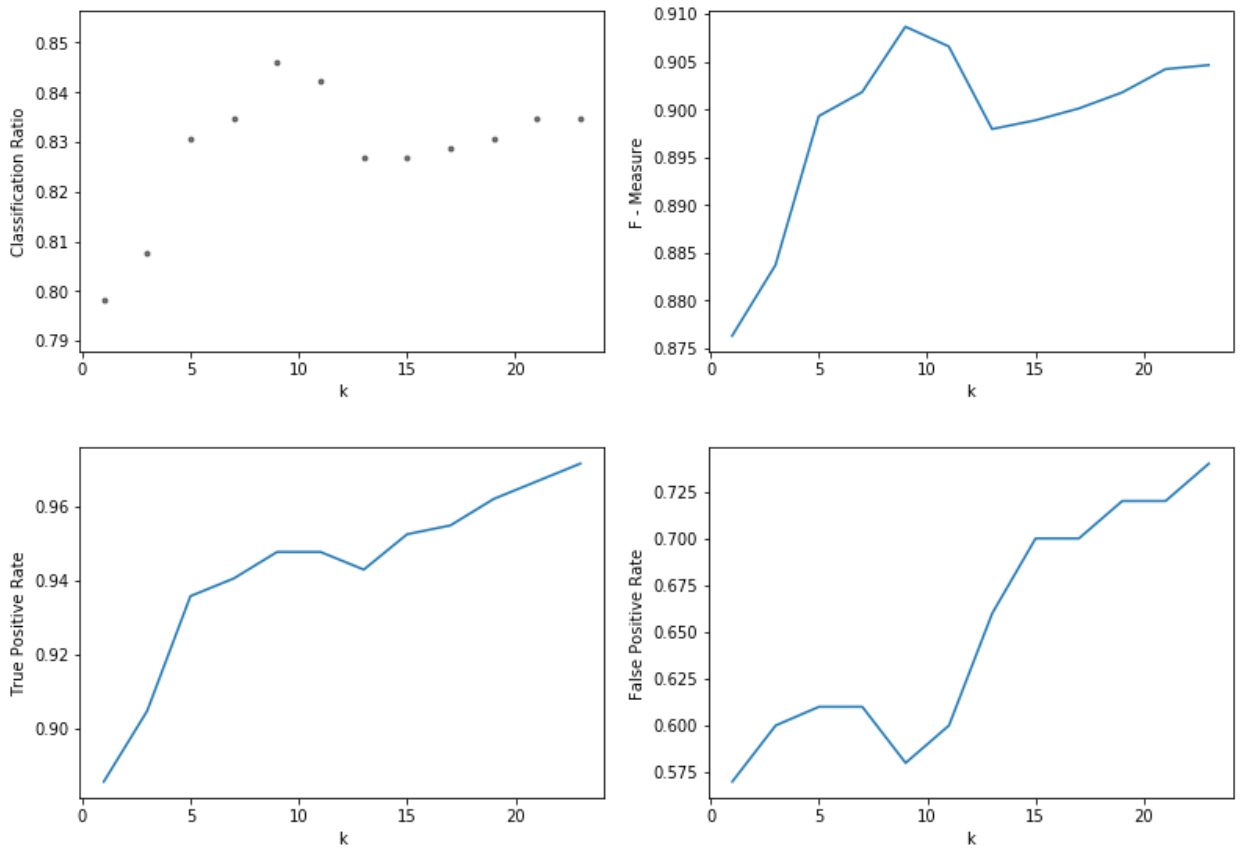


Fig 3.7 : False Positive and true positive vs k for cross validation

F - measure has peak value at k = 9 and at k = 9, the false positive rate is pretty low. So, k = 9 is the recommended value in this case.

4. OFF THE SHELF KNN :

KNN classifier in Weka is used with k varying from 1 to 20. Results observed are :

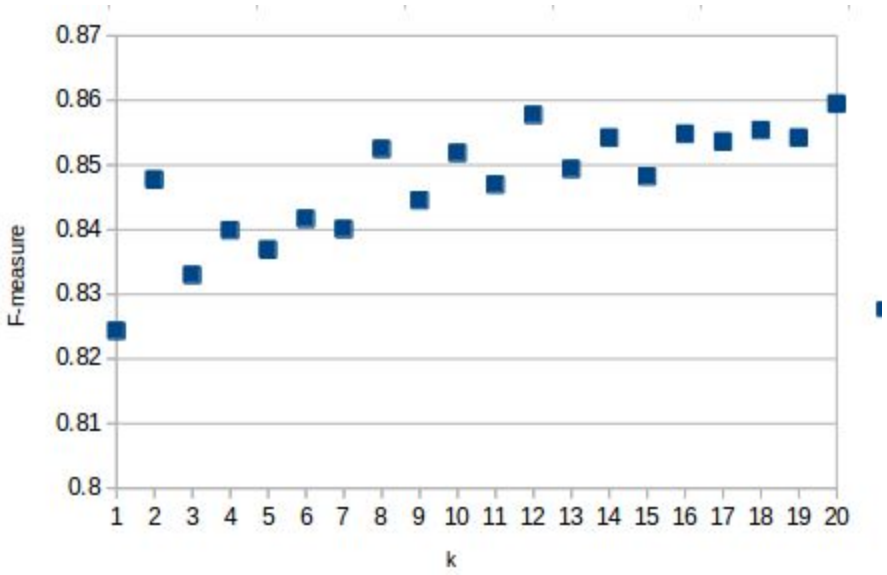
For k = 12 :

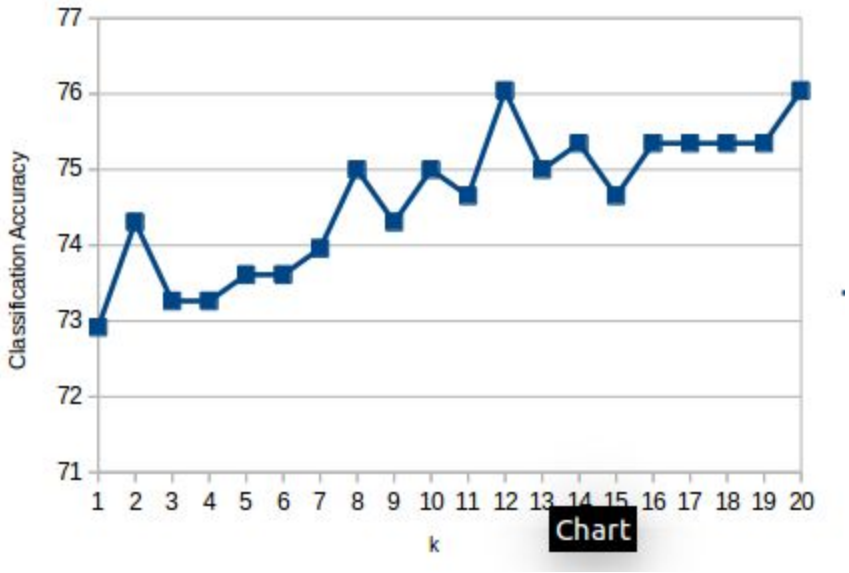
Correctly Classified Instances	223	77.4306 %
Incorrectly Classified Instances	65	22.5694 %

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.914	0.687	0.815	0.914	0.861	0.278	0.782	0.917	<=50K
	0.313	0.086	0.525	0.313	0.393	0.278	0.782	0.451	>50K
Weighted Avg.	0.774	0.547	0.747	0.774	0.752	0.278	0.782	0.809	

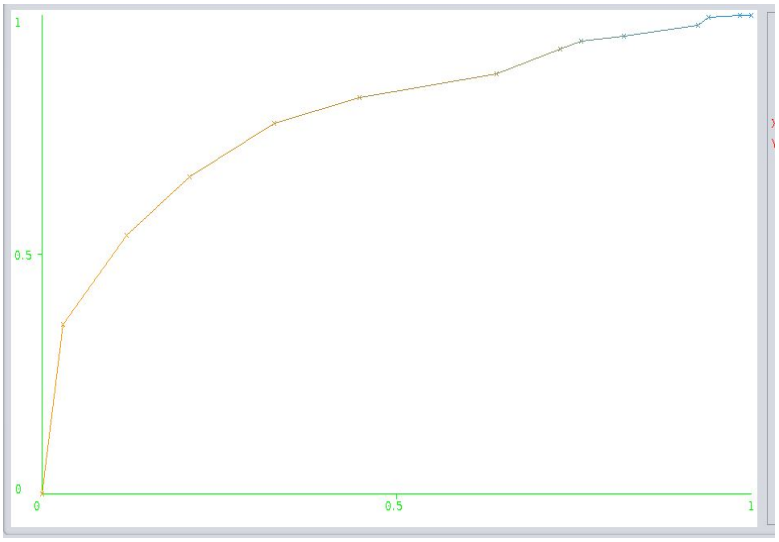
=== Confusion Matrix ===

a	b	<-- classified as
202	19	a = <=50K
46	21	b = >50K





ROC for k = 12 :



Classification accuracy is about the same for implemented KNN and off the shelf KNN.