Programming Assignment

Name: Neel Jayeshkumar Suthar
UTA ID: 1001807983

AIM: In this programming assignment I implemented the KNN algorithm from scratch and the functions to evaluate it with a k-fold cross validation (also from scratch). We are supposed to try different distance measures and techniques to get better results from KNN obtained from Weka. Also, I focused on some basic parameters tuning for my KNN.

Dataset:

For this implementation I have used three datasets from UCI- Machine Learning Repository.

1. Hayes- Roth dataset
2. Car Evaluation dataset
3. Breast Cancer dataset

I have implemented all the dataset in the KNN scratch code and get the accuracy measures from WEKA workbench. During both implementation 10-fold cross validation was used.

KNN on Hayes-Roth dataset:

This dataset has only integers values which eases our modification requirements on it. I implemented six distance measures on it including scaling feature for Euclidean distance measure. First, I tested my KNN using Euclidean distance and I found out the accuracies were lower. I used minmax scaling on my dataset. After applying minmax scaling onto all 10 folds I again tested my KNN using Euclidean distance and I achieved **83.141%** accuracy which was a very good output. After that I tried different k values so I can improve my accuracy even more and as a result I achieved **85.790%** accuracy for **k=3**. By doing minmax scaling and parameter tuning we can get robustness measures.

```
KNN Using Euclidean Distance
Num of nbrs: 1
Scores: [84.61538461538461, 76.92307692307693, 69.23076923076923, 69.23076923076923, 84.61538461538461, 76.92307692307693, 92
Mean Accuracy: 83.077%


Num of nbrs: 3
Scores: [84.61538461538461, 76.92307692307693, 76.92307692307693, 84.61538461538461, 84.61538461538461, 76.92307692307693, 92
Mean Accuracy: 84.923%


Num of nbrs: 5
Scores: [76.92307692307693, 61.53846153846154, 76.92307692307693, 76.92307692307693, 92.3076923076923, 76.92307692307693, 92.
Mean Accuracy: 82.615%


Num of nbrs: 7
Scores: [76.92307692307693, 69.23076923076923, 69.23076923076923, 84.61538461538461, 76.92307692307693, 69.23076923076923, 92
Mean Accuracy: 78.000%
```

I also have tried different distance methods for getting neighbors. I got average accuracies with all other distance measures. I also have tried k-tuning on all different distance measures. You can see results below.

Best accuracy using Normalized Euclidean distance measure.

```
Num of nbrs: 3
Scores: [84.61538461538461, 61.53846153846154, 84.61538461538461, 76.92307692307693, 76.92307692307693, 76.92307692307693, 92.3076
Mean Accuracy: 78.872%
```

Best accuracy using Cosine similarity measur.

```
Num of nbrs: 3
Scores: [53.84615384615385, 61.53846153846154, 53.84615384615385, 61.53846153846154, 61.53846153846154, 61.53846153846154, 84.61538461538461, 61.53846
Mean Accuracy: 64.256%
```

Best accuracy using Manhattan distance measure.

```
Num of nbrs: 5
Scores: [76.92307692307693, 76.92307692307693, 76.92307692307693, 92.3076923076923, 84.61538461538461, 69.23076923076923, 92.307
Mean Accuracy: 82.615%
```

Best accuracy using Minkowski distance measure.

```
Num of nbrs: 1
Scores: [76.92307692307693, 76.92307692307693, 53.84615384615385, 76.92307692307693, 69.23076923076923, 92.3076923076923, 76.9236
Mean Accuracy: 76.256%
```

Best accuracy using Hamming distance measure.

```
Num of nbrs: 5
Scores: [84.61538461538461, 92.3076923076923, 53.84615384615385, 92.3076923076923, 84.61538461538461, 92.3076923076923, 92.3076
Mean Accuracy: 81.077%
```

KNN from Weka gives accuracy shown below.

```
Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          88               66.6667 %
Incorrectly Classified Instances        44               33.3333 %
Kappa statistic                          0.4693
Mean absolute error                      0.1835
Root mean squared error                  0.3091
Relative absolute error                 42.3182 %
Root relative squared error             66.4023 %
Total Number of Instances              132

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Cl
                 0.784     0.296     0.625       0.784   0.696      0.475    0.923     0.897     1
                 0.706     0.247     0.643       0.706   0.673      0.452    0.925     0.899     2
                 0.400     0.000     1.000       0.400   0.571      0.583    1.000     1.000     3
Weighted Avg.    0.667     0.210     0.717       0.667   0.659      0.491    0.941     0.922

=== Confusion Matrix ===

  a  b  c   <-- classified as
 40 11  0 |  a = 1
 15 36  0 |  b = 2
```

Implementing Car Evaluation dataset:

This dataset consists of string values and integer values as well. So, before evaluation of algorithm, all the string values should be converted to integer or float values. For that, **str_column_to_int** function is used.

Then, for 10-fold cross-validation, this dataset gave **96.427%** accuracy with k=7 using Manhattan distance measure

```
Num of nbrs: 1
Scores: [88.37209302325581, 82.55813953488372, 81.3953488372093, 81.3953488372093, 85.46511627906976, 83.72093023255815, 83.720930
Mean Accuracy: 84.084%


Num of nbrs: 3
Scores: [90.11627906976744, 91.27906976744185, 87.20930232558139, 88.37209302325581, 90.69767441860465, 93.02325581395348, 92.4418
Mean Accuracy: 91.196%


Num of nbrs: 5
Scores: [93.02325581395348, 96.51162790697676, 89.53488372093024, 94.76744186046511, 98.25581395348837, 95.34883720930233, 94.1860
Mean Accuracy: 94.963%


Num of nbrs: 7
Scores: [96.51162790697676, 97.09302325581395, 95.34883720930233, 95.34883720930233, 97.67441860465115, 96.51162790697676, 95.9302
Mean Accuracy: 96.472%
```

With Minkowski distance measure algorithm was time consuming and because of that I was able to get accuracies for k=1 and k=3

```
Num of nbrs: 1
Scores: [81.3953488372093, 81.97674418604652, 76.74418604651163, 82.55813953488372, 78.48837209302324, 81.3953488372093, 78.48837
Mean Accuracy: 80.043%


Num of nbrs: 3
Scores: [86.62790697674419, 84.30232558139535, 91.27906976744185, 87.20930232558139, 81.97674418604652, 85.46511627906976, 82.558
Mean Accuracy: 85.889%
```

With hamming distance there was not much difference in accuracy.

```
Num of nbrs: 1
Scores: [88.37209302325581, 82.55813953488372, 81.3953488372093, 81.3953488372093, 85.46511627906976, 83.72093023255815, 83.726
Mean Accuracy: 84.084%
```

There are various methods by which we can improve accuracy. Removing some of the data is one of those but I do not think that is a better approach but of course you can do reasoning on deciding columns for classifications and remove some of columns which are less important which will definitely help with distance measures and predictions.

KNN from Weka gives accuracy shown below.

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1632               94.4444 %
Incorrectly Classified Instances        96                5.5556 %
Kappa statistic                          0.876
Mean absolute error                      0.1122
Root mean squared error                  0.1953
Relative absolute error                 48.9977 %
Root relative squared error             57.7645 %
Total Number of Instances             1728

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Cl
                 0.964     0.061     0.819      0.964    0.885      0.854    0.988     0.958     0
                 0.188     0.001     0.867      0.188    0.310      0.395    0.994     0.859     1
                 0.994     0.023     0.990      0.994    0.992      0.974    1.000     1.000     2
                 0.708     0.000     1.000      0.708    0.829      0.836    1.000     1.000     3
Weighted Avg.    0.944     0.030     0.947      0.944    0.935      0.919    0.997     0.985

=== Confusion Matrix ===

   a    b    c    d   <-- classified as
 370    2   12    0 |    a = 0
  56   13    0    0 |    b = 1
   7    0 1203    0 |    c = 2
  19    0    0   46 |    d = 3
```

Implementing Breast-cancer dataset:

This dataset has to be preprocessed as it contains some string values along with integer values. For this dataset, KNN using Euclidean distance with k=7 performs **78.464%** accuracy which is best accuracy among all.

```
Using Euclidean Distance...

Num of nbrs: 1
Scores: [77.77777777777779, 70.37037037037037, 59.25925925925925, 66.66666666666666, 62.96296296296296, 77.77777777777779, 77.77
Mean Accuracy: 73.431%


Num of nbrs: 3
Scores: [81.48148148148148, 77.77777777777779, 66.66666666666666, 66.66666666666666, 66.66666666666666, 77.77777777777779, 85.18
Mean Accuracy: 77.135%


Num of nbrs: 5
Scores: [74.07407407407408, 74.07407407407408, 66.66666666666666, 70.37037037037037, 74.07407407407408, 77.77777777777779, 88.88
Mean Accuracy: 77.211%


Num of nbrs: 7
Scores: [74.07407407407408, 77.77777777777779, 74.07407407407408, 74.07407407407408, 66.66666666666666, 74.07407407407408, 88.88
Mean Accuracy: 78.464%
```

Best accuracy using Manhattan Distance and parameter tuning.

```
Num of nbrs: 7
Scores: [70.37037037037037, 70.37037037037037, 74.07407407407408, 70.37037037037037, 70.37037037037037, 81.48148148148148, 81.4814814
Mean Accuracy: 75.948%
```

Best accuracy using Minkowski Distance and parameter tuning.

```
Num of nbrs: 3
Scores: [81.48148148148148, 70.37037037037037, 62.96296296296296, 62.96296296296296, 77.77777777777779, 77.77777777777779, 85.1851851
Mean Accuracy: 76.906%
```

Best accuracy using Hamming Distance and parameter tuning.

```
Num of nbrs: 3
Scores: [82.14285714285714, 75.0, 67.85714285714286, 67.85714285714286, 78.57142857142857, 67.85714285714286, 75.0, 78.57142
Mean Accuracy: 75.441%
```

KNN from Weka gives accuracy shown below.

Classifier output

```
Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         211               73.7762 %
Incorrectly Classified Instances        75               26.2238 %
Kappa statistic                          0.2931
Mean absolute error                      0.3018
Root mean squared error                  0.4836
Relative absolute error                 72.1245 %
Root relative squared error            105.8067 %
Total Number of Instances              286

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cl
                 0.896    0.635    0.769      0.896    0.828      0.308    0.662     0.785     0
                 0.365    0.104    0.596      0.365    0.453      0.308    0.662     0.507     1
Weighted Avg.    0.738    0.478    0.718      0.738    0.716      0.308    0.662     0.702

=== Confusion Matrix ===

   a   b   <-- classified as
 180  21 |   a = 0
  54  31 |   b = 1
```

References for KNN and k-fold cross validation:
https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/
https://machinelearningmastery.com/k-fold-cross-validation/

With this document file I have attached following python files.

1. Hayes-Roth.ipynb
2. Car.ipynb
3. Breast-Cancer.ipynb