



UNIVERSIDADE DE SÃO PAULO

Escola de Artes, Ciências e Humanidades

ACH2016 - Inteligência Artificial

Karina

Ivo de Andrade de Deus

8075238

Relatório do Exercício Programa 2 Classificação

São Paulo
2018

Índice

1. Introdução	3
1.1. Pré-processamento de dados	3
1.2. Conjuntos de Treinamento e Teste	3
2. Linguagem e ferramentas utilizadas	4
3. Algoritmo k-NN	4
3.1 Comparação entre métodos de medição de distâncias	4
3.2 Comparação entre diferentes valores de k	5
3.3 Padronização de dados	6
4. Algoritmo Naive Bayes	7
5. Árvores de Decisão	7
6. Robustez quanto à ruídos	7
7. Conclusões	8

1. Introdução

Para este exercício, foi escolhido a base de dados TMD (*Transportation Mode Detection with Unconstrained Smartphones Sensors*), a qual contém informações de sensores de smartphones captadas durante uma multitude de trajetos feitos em 4 modos de transporte (classificados pelos atributos de classe Bus, Car, Train e Walking), mais a condição de objeto parado (classificado como Still). Esta base de dados se encontra disponível no Kaggle, e foi fornecido pelo enunciado do problema.

A base de dados TMD contém 5893, com cerca de 1180 exemplos para cada tipo de classe (contabilizados pelo algoritmo em `csv_conjuntos.csv`), e possui 13 atributos de valores contínuos sobre o tempo do trajeto, assim como os valores máximos, mínimos e desvios padrões médios do acelerômetro, giroscópio e volume sonoro captados pelo smartphone.

1.1. Pré-processamento de dados

Inicialmente, foram feitas correções quanto aos valores dos desvios padrões presentes na base de dados, uma vez que o desvio padrão, às vezes, apresentava valores não nulos quando seus valores máximos e mínimos eram idênticos. Para tal correção, foi-se utilizado o código encontrado em `csv_correcao.py`, que recebeu os dados em arquivo csv e corrigiu esses valores para salvar-los em um novo arquivo.

1.2. Conjuntos de Treinamento e Teste

Neste exercício programa, definido um conjunto de treinamento com 70% dos exemplos e um conjunto de testes com os 30% dos exemplo remanescentes, onde em ambos a proporção de exemplos de uma dada classe foi mantida. Esta divisão foi feita por meio do algoritmo encontrado em `csv_conjuntos.csv`, onde foram separados os elementos por classes e, em seguida, designados para os conjuntos de treinamento e teste de forma aleatória.

2. Linguagem e ferramentas utilizadas

As ferramentas utilizadas para este exercício programa foram o Python IDLE 3.7, onde inicialmente foram realizados os testes com o algoritmo k-NN redigido em linguagem Python, e o simulador WEKA (*Waikato Environment for Knowledge Analysis*), para os testes relativos aos algoritmos de Naive Bayes e árvores de decisão.

Este exercício programa foi realizado em um sistema operacional Windows 10, com processador Intel Core i3-2120 de 3.3GHz, e memória RAM de 6Gb.

3. Algoritmo k-NN

O algoritmo k-NN (*k-Nearest Neighbors*) visa estudar, através de um conjunto de treinamento, as distâncias de dado conjunto de teste aos conjunto de dados conhecido e, por meio dos exemplos 'vizinhos' mais próximos a um dado caso de teste, definir suas classe.

Para os seguintes testes, utilizamos o algoritmo `csv_knn.py`, o qual calcula a implementação de k-NN em um conjunto de treinamento e testes, onde é possível manipular o intervalo de k a ser trabalhado (estudando o k-NN com múltiplos valores de k em sequência) e se será utilizado a distância Euclidiana ou de Manhattan para a definição dos vizinhos. Como critério de desempate (caso se encontre dois ou mais grupos de mesma classe próximos ao dado exemplo para um dado valor k), optou-se pela redução o valor de k por um até que se encontre um grupo de vizinhos de mesma classe a qual sua frequência é predominante sobre as outras.

3.1 Comparação entre métodos de medição de distâncias

Primeiro, com a implementação de k-NN com valor de k igual a 3, comparamos com o algoritmo interage com a base de dados ao utilizar diferentes métodos de verificação de vizinhos mais próximos, a distância Euclidiana e a distância de Manhattan. Enquanto a distância Euclidiana soma o quadrado das diferenças entre as coordenadas de dois pontos e tira a raiz quadrada do valor obtido, a distância de Manhattan soma o módulo das diferenças entre as coordenadas.

$$\sqrt{\sum_{i=1}^n (p_i - q_i)^2} \quad \sum_{i=1}^n |p_i - q_i|$$

Figura 1. Distância Euclidiana e Distância de Manhattan

Após a implementação, obteve-se os seguintes resultados:

Método de distância utilizado em k-NN	Número de acertos	Porcentagem de casos corretos
Euclidiana	1061	60,0793 %
Manhattan	1167	66,0815 %

Tabela 1. Comparação entre distância Euclidiana e distância de Manhattan

Nessa implementação, a porcentagem de acertos foi de certo modo maior com a distância de Manhattan do que pela distância Euclidiana apesar de, teoricamente, ser uma diferença dispensável.

3.2 Comparação entre diferentes valores de k

Em seguida, houve a implementação do algoritmo k-NN com valores de k entre 1 até 10, utilizando-se da distância Euclidiana para a medição dos exemplos mais próximos, e com isso foram obtidos os seguintes resultados:

Valor de k	Número de acertos	Porcentagem de acertos	Valor de k	Número de acertos	Porcentagem de acertos
1	1078	61,0419 %	6	1025	58,0408 %
2	1078	61,0419 %	7	1008	57,0781 %
3	1061	60,0793 %	8	1012	57,3046 %
4	1028	58,2106 %	9	994	56,2854 %
5	1025	58,0408 %	10	971	54,9830 %

Tabela 2. Comparação entre diferentes valores de k

Acurácia entre valores de k

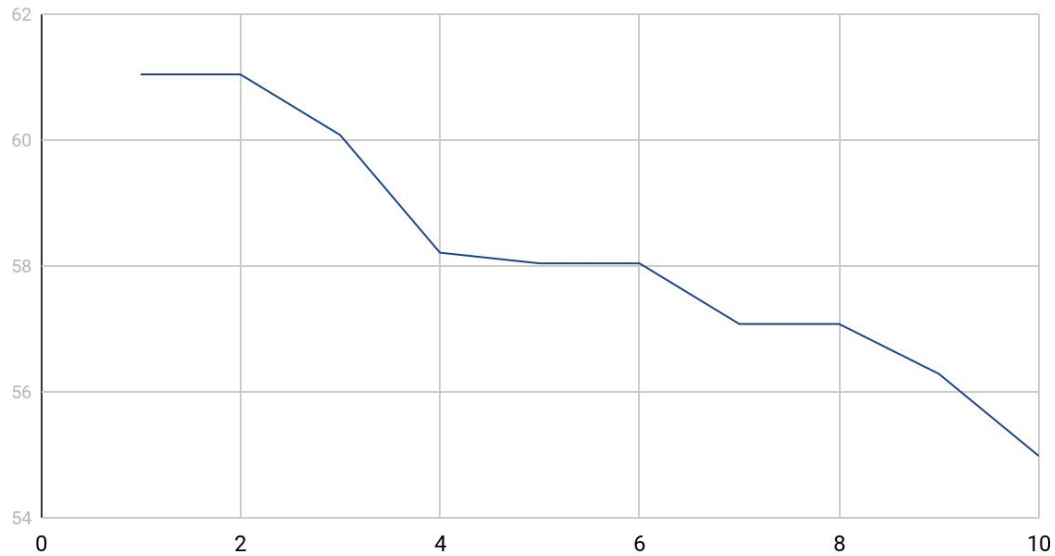


Figura 2. Relação entre valor de k e acurácia de k-NN

Percebe-se que há uma redução na eficácia do algoritmo conforme o valor de k é aumentado, como se é esperado do algoritmo k-NN. Porém, para os valores de 1 e 2 para k, estes são apresentados com melhor eficiência do que k igual a 3, o que vai contra a teoria de que estes dois valores nem sempre serem confiáveis quanto a classificação de um elemento por si sós.

3.3 Padronização de dados

Por fim, houve a padronização dos atributos em ambos os conjuntos de treinamento e de teste (com média 0 e variância 1), com o intuito de calcular novamente de aplicar o algoritmo de k-NN com o melhor valor de k encontrado no item anterior. Uma vez que, em teoria, o valor de k igual é o mais apropriado mas os resultados apontaram para uma melhor eficácia com os valores de k iguais a 1 ou 2, decidiu-se realizar o teste novamente com os valores de k entre 1 a 10, com os seguintes resultados:

Valor de k	Número de acertos	Porcentagem de acertos	Valor de k	Número de acertos	Porcentagem de acertos
1	1291	73,1031 %	6	1297	73,4428 %
2	1291	73,1031 %	7	1292	73,1597 %
3	1309	74,1223 %	8	1287	72,8766 %
4	1300	73,6127 %	9	1282	72,5934 %
5	1300	73,6127 %	10	1279	72,4236 %

Tabela 3. Comparação entre diferentes valores de k com dados padronizados

Acurácia entre valores de k com dados personalizados

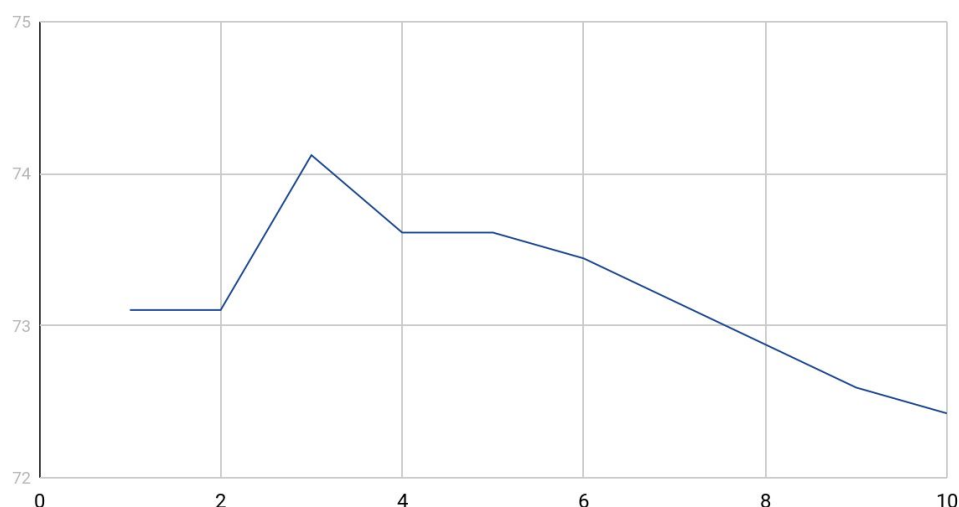


Figura 2. Relação entre valor de k e acurácia de k-NN

Neste caso, o valor de k igual a 3 foi o melhor com os dados padronizados, como se era esperado no item anterior, se demonstrando mais confiável quanto a classificação de um elemento do meio de k-NN.

4. Algoritmo Naive Bayes

O algoritmo Naive Bayes busca prever a classe de um dado exemplo por meio do estudo probabilístico dos casos de treino, utilizando os atributos do banco de dados para calcular as probabilidades deste exemplo ser de cada uma das classes classificatórias. Para os testes a seguir, foi utilizado o simulador WEKA (*Waikato Environment for Knowledge Analysis*) com o algoritmo de Naive Bayes presente neste.

Antes dos testes, porém, foi efetuado a discretização dos dados dos conjuntos de treinamento e de teste onde, para cada atributo, foi identificado o intervalo entre os valores máximos e mínimos e, com sua divisão em 6 partes de tamanho igual, categorizou-se os dados em 6 diferentes classes (A, B, C, D, E e F). Este processo foi realizado em Python, por meio do algoritmo `csv_discretizacao.py`.

Para os testes, então, foram realizados dois métodos de validação de dados para ambos os dados padronizados e não-padronizados: a validação a partir da verificação do modelo com o conjunto de testes; e a validação cruzada, por meio de 10 folds sobre o próprio conjunto de treinamento (o conjunto de treinamento é dividido em dez partes e testado dez vezes, com uma fração décima reservada como conjunto de testes em cada iteração. Com isso obteve-se:

Conjunto de dados	Método de validação	
	Conjunto de testes	Validação Cruzada
Não-Padronizado	52,265 % (Figura 3)	52,411% (Figura 4)
Padronizado	57,5472 % (Figura 5)	51,7083 % (Figura 6)

Tabela 4. Acurácia do algoritmo Naive Bayes

Em seguida, encontram-se as configurações do WEKA para os testes realizados acima, assim como a matriz de confusão resultante dos testes de validação.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier: Choose NaiveBayes

Test options:

- ☐ Use training set
- ☒ Supplied test set Set...
- ☐ Cross-validation Folds 10
- ☐ Percentage split % 66
- More options...

(Nom) target

Start Stop

Result list (right-click for options)

19:40:27 - bayes.NaiveBayes

Classifier output:

```

1764 3:Train 3:Train 0.978
1765 4:Bus 3:Train + 0.899
1766 3:Train 3:Train 0.968

=== Summary ===
Correctly Classified Instances      923      52.265 %
Incorrectly Classified Instances    843      47.735 %
Kappa statistic                    0.4033
Mean absolute error                0.1891
Root mean squared error            0.3961
Total Number of Instances          1766

=== Detailed Accuracy By Class ===

```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,311	0,079	0,495	0,311	0,382	0,279	0,778	0,473	Car
	0,813	0,028	0,878	0,813	0,844	0,808	0,925	0,854	Walking
	0,799	0,351	0,362	0,799	0,499	0,361	0,828	0,623	Train
	0,275	0,048	0,588	0,275	0,375	0,311	0,821	0,495	Bus
	0,416	0,090	0,536	0,416	0,469	0,361	0,861	0,590	Still
Weighted Avg.	0,523	0,119	0,572	0,523	0,514	0,424	0,843	0,607	

```

=== Confusion Matrix ===
 a b c d e <-- classified as
110 8 153 42 41 | a = Car
20 287 25 14 7 | b = Walking
19 7 282 6 39 | c = Train
64 13 139 97 40 | d = Bus
9 12 179 6 147 | e = Still

```

Status: OK Log

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier: Choose NaiveBayes

Test options:

- ☐ Use training set
- ☐ Supplied test set Set...
- ☒ Cross-validation Folds 10
- ☐ Percentage split % 66
- More options...

(Nom) target

Start Stop

Result list (right-click for options)

19:51:38 - bayes.NaiveBayes

Classifier output:

```

=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances      2163      52.411 %
Incorrectly Classified Instances    1964      47.589 %
Kappa statistic                    0.4051
Mean absolute error                0.19
Root mean squared error            0.3989
Relative absolute error             59.3669 %
Root relative squared error         99.7188 %
Total Number of Instances          4127

=== Detailed Accuracy By Class ===

```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,320	0,077	0,511	0,320	0,393	0,294	0,761	0,473	Car
	0,796	0,031	0,865	0,796	0,829	0,790	0,930	0,858	Walking
	0,828	0,370	0,359	0,828	0,501	0,368	0,832	0,584	Train
	0,276	0,039	0,639	0,276	0,386	0,338	0,814	0,477	Bus
	0,401	0,078	0,561	0,401	0,468	0,368	0,882	0,629	Still
Weighted Avg.	0,524	0,119	0,587	0,524	0,515	0,431	0,844	0,604	

```

=== Confusion Matrix ===
 a b c d e <-- classified as
264 29 382 61 90 | a = Car
65 656 52 38 13 | b = Walking
36 12 684 14 80 | c = Train
138 42 341 228 76 | d = Bus
14 19 446 16 331 | e = Still

```

Status: OK Log

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose NaiveBayes

Test options

☐ Use training set

☒ Supplied test set

☐ Cross-validation Folds 10

☐ Percentage split % 66

(Nom) target

Result list (right-click for options)

19:53:35 - misc.InputMappedClassifier

Classifier output

Time taken to test model on supplied test set: 0.09 seconds

=== Summary ===

Correctly Classified Instances	61	57.5472 %
Incorrectly Classified Instances	45	42.4528 %
Kappa statistic	0.4604	
Mean absolute error	0.186	
Root mean squared error	0.329	
Relative absolute error	58.1403 %	
Root relative squared error	82.2564 %	
Total Number of Instances	106	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,045	0,012	0,500	0,045	0,083	0,100	0,853	0,587	Car
	0,938	0,044	0,789	0,938	0,857	0,934	0,989	0,946	Walking
	0,962	0,363	0,463	0,962	0,625	0,516	0,827	0,529	Train
	0,273	0,071	0,500	0,273	0,353	0,258	0,779	0,461	Bus
	0,700	0,058	0,737	0,700	0,718	0,655	0,891	0,726	Still
Weighted Avg.	0,575	0,124	0,579	0,575	0,509	0,450	0,859	0,627	


=== Confusion Matrix ===

```

a b c d e <-- classified as
1 0 14 5 2 | a = Car
0 15 0 1 0 | b = Walking
0 0 25 0 1 | c = Train
1 3 10 6 2 | d = Bus
0 1 5 0 14 | e = Still

```

Status

OK  x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose NaiveBayes

Test options

☐ Use training set

☐ Supplied test set

☒ Cross-validation Folds 10

☐ Percentage split % 66

(Nom) target

Result list (right-click for options)

19:53:35 - misc.InputMappedClassifier

19:54:29 - bayes.NaiveBayes

Classifier output

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	2134	51.7083 %
Incorrectly Classified Instances	1993	48.2917 %
Kappa statistic	0.3963	
Mean absolute error	0.2106	
Root mean squared error	0.3505	
Relative absolute error	65.8044 %	
Root relative squared error	87.6255 %	
Total Number of Instances	4127	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,103	0,016	0,620	0,103	0,177	0,195	0,685	0,352	Car
	0,761	0,029	0,867	0,761	0,811	0,770	0,896	0,851	Walking
	0,857	0,389	0,355	0,857	0,502	0,375	0,767	0,384	Train
	0,259	0,067	0,492	0,259	0,340	0,251	0,787	0,433	Bus
	0,605	0,103	0,596	0,605	0,601	0,500	0,856	0,628	Still
Weighted Avg.	0,517	0,121	0,586	0,517	0,486	0,418	0,798	0,529	


=== Confusion Matrix ===

```

a b c d e <-- classified as
85 38 443 134 126 | a = Car
16 627 79 54 48 | b = Walking
11 3 708 28 76 | c = Train
15 47 460 214 89 | d = Bus
10 8 303 5 500 | e = Still

```

Status

OK  x 0

5. Árvores de Decisão

Árvore de decisão	Conjunto de dados	
	Não-Padronizado	Padronizado
Sem Poda	76,4439 % (Figura 7)	53,8505% (Figura 8)
Com poda	77,6897 % (Figura 9)	53,8505 % (Figura 10)

Tabela 5. Acurácia do algoritmo de árvore de decisão

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -O -U -M 2

Test options

☐ Use training set

☒ Supplied test set Set...

☐ Cross-validation Folds 10

☐ Percentage split % 66

More options...

(Nom) target

Start Stop

Result list (right-click for options)

20:23:00 - misc.InputMappedClassifier

Classifier output

```

=== Summary ===

Correctly Classified Instances      1350           76.4439 %
Incorrectly Classified Instances    416           23.5561 %
Kappa statistic                    0.7055
Mean absolute error                0.0975
Root mean squared error            0.2857
Relative absolute error            30.4679 %
Root relative squared error        71.4166 %
Total Number of Instances          1766

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
0,675    0,086    0,664    0,675    0,669    0,586    0,838    0,611    Car
0,884    0,034    0,867    0,884    0,875    0,844    0,927    0,794    Walking
0,720    0,070    0,720    0,720    0,720    0,649    0,837    0,588    Train
0,748    0,079    0,704    0,748    0,725    0,655    0,852    0,603    Bus
0,796    0,026    0,884    0,796    0,838    0,801    0,913    0,803    Still
Weighted Avg.   0,764    0,059    0,767    0,764    0,765    0,707    0,874    0,680

=== Confusion Matrix ===

  a  b  c  d  e  <-- classified as
239  6  45  58  6  |  a = Car
  9 312  4  16 12  |  b = Walking
 49  9 254  25 16  |  c = Train
 44 17  25 264  3  |  d = Bus
 19 16  25  12 281 |  e = Still

```

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -O -U -M 2

Test options

☐ Use training set

☒ Supplied test set Set...

☐ Cross-validation Folds 10

☐ Percentage split % 66

More options...

(Nom) target

Start Stop

Result list (right-click for options)

20:23:00 - misc.InputMappedClassifier

20:24:35 - misc.InputMappedClassifier

Classifier output

```

=== Summary ===

Correctly Classified Instances      951           53.8505 %
Incorrectly Classified Instances    815           46.1495 %
Kappa statistic                    0.4231
Mean absolute error                0.1827
Root mean squared error            0.4045
Relative absolute error            57.1079 %
Root relative squared error        101.1346 %
Total Number of Instances          1766

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
0,534    0,180    0,427    0,534    0,474    0,327    0,715    0,393    Car
0,822    0,068    0,751    0,822    0,785    0,729    0,868    0,629    Walking
0,416    0,115    0,474    0,416    0,443    0,316    0,652    0,304    Train
0,569    0,137    0,509    0,569    0,537    0,415    0,690    0,361    Bus
0,351    0,076    0,534    0,351    0,424    0,325    0,800    0,472    Still
Weighted Avg.   0,539    0,115    0,539    0,539    0,533    0,422    0,745    0,432

=== Confusion Matrix ===

  a  b  c  d  e  <-- classified as
189 19  59  67 20  |  a = Car
 14 290 10  29 10  |  b = Walking
 71 13 147  79 43  |  c = Train
 42 38  37 201 35  |  d = Bus
127 26  57  19 124 |  e = Still

```

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -C 0.25 -M 2

Test options

☐ Use training set

☒ Supplied test set Set...

☐ Cross-validation Folds 10

☐ Percentage split % 66

More options...

(Nom) target

Start Stop

Result list (right-click for options)

20:23:00 - misc.InputMappedClassifier

20:24:35 - misc.InputMappedClassifier

20:25:54 - misc.InputMappedClassifier

20:26:30 - misc.InputMappedClassifier

Classifier output

```
=== Summary ===

Correctly Classified Instances      1372           77.6897 %
Incorrectly Classified Instances    394           22.3103 %
Kappa statistic                    0.7211
Mean absolute error                0.0989
Root mean squared error            0.281
Relative absolute error             30.9053 %
Root relative squared error        70.2589 %
Total Number of Instances         1766

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
0.678    0.078    0.686    0.678    0.682    0.603    0.852    0.628    Car
0.887    0.028    0.887    0.887    0.887    0.858    0.933    0.837    Walking
0.717    0.057    0.757    0.717    0.737    0.673    0.845    0.640    Train
0.771    0.085    0.694    0.771    0.730    0.660    0.859    0.620    Bus
0.833    0.030    0.872    0.833    0.852    0.817    0.911    0.793    Still
Weighted Avg.   0.777    0.056    0.779    0.777    0.777    0.722    0.880    0.703

=== Confusion Matrix ===

  a  b  c  d  e  <-- classified as
240  5  38  61  10 | a = Car
  8 313  4  16  12 | b = Walking
  47  8 253  29  16 | c = Train
  39 17  20 272   5 | d = Bus
  16 10  19  14 294 | e = Still
```

Status

OK

Log

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -C 0.25 -M 2

Test options

☐ Use training set

☒ Supplied test set Set...

☐ Cross-validation Folds 10

☐ Percentage split % 66

More options...

(Nom) target

Start Stop

Result list (right-click for options)

20:23:00 - misc.InputMappedClassifier

20:24:35 - misc.InputMappedClassifier

20:25:54 - misc.InputMappedClassifier

20:26:30 - misc.InputMappedClassifier

20:30:50 - misc.InputMappedClassifier

20:31:59 - misc.InputMappedClassifier

Classifier output

```
=== Summary ===

Correctly Classified Instances      951           53.8505 %
Incorrectly Classified Instances    815           46.1495 %
Kappa statistic                    0.4231
Mean absolute error                0.1842
Root mean squared error            0.4039
Relative absolute error            100.9641 %
Root relative squared error        100.9641 %
Total Number of Instances         1766

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
0.542    0.173    0.440    0.542    0.486    0.343    0.736    0.474    Car
0.827    0.070    0.747    0.827    0.785    0.729    0.885    0.717    Walking
0.380    0.098    0.493    0.380    0.429    0.312    0.673    0.342    Train
0.567    0.153    0.481    0.567    0.520    0.390    0.706    0.372    Bus
0.377    0.094    0.530    0.377    0.440    0.336    0.822    0.499    Still
Weighted Avg.   0.539    0.115    0.538    0.539    0.532    0.422    0.764    0.481

=== Confusion Matrix ===

  a  b  c  d  e  <-- classified as
192 21  41  79  21 | a = Car
 13 292   9  29  10 | b = Walking
  68 14 134  90  47 | c = Train
  38 42  33 200  40 | d = Bus
 125 22  55  18 133 | e = Still
```

Status

OK

Log