Classification of Injuries Sustained by Non-Motorists

Kaavya Borra, Radin Rezanezhad

10/22/2024

Table of Contents

Statement/Project Goal
The primary goal of this classification task is to predict the severity of injuries sustained by a non-motorist, such as a pedestrian or cyclist, involved in a car collision. Accurate prediction of injury severity is crucial, as it allows emergency services to assess the urgency and importance of a crash more effectively using data available at the time of dispatch, such as the type of collision, time of day, weather conditions, and location.

By predicting a high severity of injury, dispatchers can allocate additional resources—such as ambulances, paramedics, and specialized equipment—to the crash site even before the first responders arrive. This proactive approach can significantly improve response times for police departments and emergency services, ensuring that those with severe injuries receive the critical care they need more quickly. It also enables better prioritization of multiple incidents, ensuring that the most serious situations receive the attention they require.

Additionally, such predictive models can support data-driven decision-making within emergency management systems, allowing for better planning and resource management over time. For example, identifying patterns in certain types of accidents or locations prone to severe injuries could help guide preventative measures and safety campaigns. Ultimately, this classification model aims to enhance overall public safety, reduce response delays, and improve outcomes for non-motorists involved in collisions.

Description of Dataset
This dataset was downloaded from dataMontgomery. It provides information about non-motorist (pedestrians and cyclists) traffic collisions occurring on county and local roadways in Montgomery County. The data is collected through the Automated Crash Reporting System (ARCS) of the Maryland State Police and reported by various local police departments.

The dataset contains 6,104 instances, with each instance representing a non-motorist involved in a traffic collision. The primary focus of this dataset is to provide information about the circumstances and outcomes of these collisions. There are initially 28 attributes:

1. Report Number: ACRS Report Number assigned to the incident
2. Local Case Number: Case number from the local investigating agency for the incident
3. Agency Name: Name of the investigating agency.
4. ACRS Report Type: Identifies crash as property, injury, or fatal
5. Crash Date/Time: Date and Time of crash
6. Route Type: Type of roadway at crash location
7. Road Name: Name of road
8. Cross-Street Name: Name of nearest cross-street
9. Off-Road Description: Description of location for off-road collisions
10. Municipality: Jurisdiction for crash location
11. Related Non-Motorist: Type(s) of non-motorist involved
12. Collision type: Type of collision
13. Weather: Weather at collision time
14. Surface condition: Condition of roadway surface
15. Light: Lightning conditions

16. Traffic Control: Signage or traffic control devices
17. Driver Substance Abuse: Substance abuse detected for all drivers involved
18. Non-Motorist Substance Abuse: Substance abuse detected for this non-motorist
19. Person ID: Unique identifier for this non-motorist
20. Pedestrian Type: Type of non-motorist
21. Pedestrian Movement: Movement of non-motorist at time of collision
22. Pedestrian Actions: Improper actions by non-motorist
23. Pedestrian Location: Location of non-motorist
24. At Fault: Whether this non-motorist was at fault
25. Injury Severity (class): Severity of injury to this non-motorist
26. Safety Equipment: Non-Motorist - Type of safety equipment if any was used
27. Latitude: Y coordinate of crash location
28. Longitude: X coordinate of crash location
29. Location: Geolocation

In all of these attributes, there are 12,610 total blank values. There may be more hidden missing values, which would require further deep analysis into the data set to see which values count as missing values for that specific attribute (i.e. for Pedestrian Actions, “NOT VISIBLE” might count as a hidden missing value). The class we picked for this data is one bolded above, Injury Severity. Here, we analyzed that around 45% of the data was classified as a Suspected Minor Injury, 32% was classified as a possible injury, suspected serious injury and no apparent injury each make up approximately 10% of the data set, and fatal injury makes up around 2%.

Pre-Processing

Pre-processing was done on Google Sheets, and Weka was used to visualize the missing values and values for each attribute

Step 1. Remove columns with over 75% missing values, as well as, irrelevant or redundant columns (i.e. report IDs, they are just case identifiers and have nothing to do with the crash). We got rid of Road Names and Cross Street Names because they only told location, and longitude and latitude values were much better for that.

- Report Number
- Local Case Number
- ACRS Report Type (too similar with our class variable)
- Cross Street Names
- Road Names
- Municipality
- Person ID
- Pedestrian Type (redundant with Related Non- Motorist)

Step 2. Crash Date/Time: Split into days past Jan 1 (with Jan 1 being 1) and hours past midnight (with 0:00 being 0), to give us data on the times of day these crashes happen, and the times of the year.

- Split column in excel to separate date and time
- format the column as a “number” and it will give a number representing how many days past Jan 1 1900 you are (google sheets default “0” date)

- split time by the colon and use the equation (=hour + (min/60)) to get a decimal for how many hours past 0:00 you are

Step 3. Replace values like “SNOW” with “Snow” or “Montgomery County Police” with “MONTGOMERY” for uniformity within the values. Implemented in the following columns

- Agency Name
- Route Type
- Related Non-Motorist
- Weather
- Surface Conditions
- Light
- Traffic Control
- Pedestrian Type
- Injury Severity

Step 4. Removed all instances of “Unknown” and “N/A” (assumed that N/A meant data not collected. Also deleted “Other” because the value is not descriptive). Implemented in the following columns

- Related Non-Motorist
- Collision Type
- Traffic Control

Step 5. Grouped alike values (e.g. “Fog, Smog, Smoke” was grouped with “FOGGY”, “MC/BIKE HELMET” became “Helmet,” and “ALCOHOL PRESENT, None Detected,” was grouped with “None Detected”)

- Weather
- Light
- Safety Equipment
- Pedestrian Actions
- Pedestrian Location
- Non-motorist substance abuse
- Driver Substance Abuse

Step 6. Split Safety Equipment into Yes and No, because splitting into 3 different attributes: helmet, reflectors, lighting, protective pads.. etc. could confuse the model and is less effective than just stating whether safety equipment was present or not

- None = No
- Everything else = Yes

Step 7. Min/Max Normalization of both Latitude and Longitude

- formula for min/max normalization:$(x-x_{min})/(x_{max} - x_{min})$

Step 8. Download as a csv, put into WEKA, and choose Injury Severity as the class by clicking “Edit” and right clicking on Injury Severity and choosing “Attribute as class”

Step 9. Replace Missing Values with WEKA (unsupervised>attribute>ReplaceMissingValues) and save this file as crash_data.csv

At this point, after the preprocessing, there are 20 attributes and the number of instances remains at 6104. The percentages are as follows:

- No Apparent Injury: 10.1%
- Possible Injury: 32.1%
- Suspected Minor Injury: 44.5%
- Suspected Serious Injury: 11.0%
- Fatal Injury: 2.2%

Attribute Selection Algorithms

In this analysis, we employed five distinct attribute selection algorithms using WEKA to identify the most relevant features for our model. The goal was to reduce dimensionality, mitigate overfitting, and enhance model performance by selecting the most informative attributes.

A. InfoGainAttributeEval with Ranker
   a. The InfoGainAttributeEval method evaluates attributes based on information gain, which measures the reduction in entropy caused by partitioning the dataset according to the attribute [1]. We used the Ranker search method with a cutoff threshold of 0.015
   b. This approach uses the following formula to calculate the InfoGain of a particular attribute A [1]:

$$InfoGain(A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|Sv|}{|S|} * Entropy(S_v)$$

Where:

- $Entropy(S) = -\sum_{i=1}^{c} p_i * \log_2(p_i)$
- c is the number of classes
- Values(A) represents the set of all possible values of attribute A
- S_v is the subset of S where attribute A has value v

Explanation:

- Entropy(S): This measures the disorder or impurity in the dataset S before splitting on A
- Weighted Entropy of Subsets: After splitting on A, you calculate the entropy for each subset S_v, weighted by the proportion of samples that fall into that subset $\frac{|Sv|}{|S|}$
- Higher information gain means that attribute A provides a better split, reducing more uncertainty.

   c. The selected attributes were Pedestrian Location, Pedestrian Actions, Pedestrian Movement, Related Non-Motorist, Route Type, Hours Past Midnight, and Collision Type

Attribute selection output

```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 20 Injury Severity):
        Information Gain Ranking Filter

Ranked attributes:
 0.03645  15 Pedestrian Location
 0.02572  14 Pedestrian Actions
 0.02099  13 Pedestrian Movement
 0.01982   5 Related Non-Motorist
 0.01677   4 Route Type
 0.01545   3 Hours past Midnight
 0.01525   6 Collision Type
 0.01422   9 Light
 0.0135   12 Non-Motorist Substance Abuse
 0.0114   10 Traffic Control
 0.01018   1 Agency Name
 0.00772  16 At Fault
 0.00677  11 Driver Substance Abuse
 0.00648   7 Weather
 0.00332   8 Surface Condition
 0.00332  17 Safety Equipment
 0        18 Lat- Normalized
 0         2 Days after Jan 1
 0        19 Long- Normalized

Selected attributes: 15,14,13,5,4,3,6,9,12,10,1,16,11,7,8,17,18,2,19 : 19
```

B. GainRatioAttributeEval with Ranker
   a. The GainRatioAttributeEval method evaluates attributes based on the gain ratio, which is similar to InfoGainAttributeEval but accounts for the number and size of branches when choosing an attribute [2]. We used the Ranker search method with a cutoff threshold of 0.1100
   b. This approach to calculate the GainRatio of a particular attribute A is similar to InfoGain, but there an extra step as the following [2]:

$$GainRatio(A) = \frac{InfoGain(A)}{SplitInfo(A)}$$

Where:

- $SplitInfo(A) = -\sum_{v \in Values(A)} \frac{|Sv|}{|S|} * \log_2(\frac{|Sv|}{|S|})$

Explanation:

- InfoGain(A): This measures the reduction in entropy when the dataset is split based on attribute A
- SplitInfo(A): This measures how evenly the data is split based on the values of attribute A. It acts as a normalizer to avoid favoring attributes with more values, which tend to have higher InfoGain but may lead to overfitting
- The GainRatio will favor attributes that not only provide a good split (high InfoGain) but also split the dataset into meaningful subsets (low SplitInfo)

c. The selected attributes were Driver Substance Abuse, Non-Motorist Substance Abuse, Hours past Midnight, Related Non-Motorist, Pedestrian Actions, Pedestrian Location, Collision Type, and Agency Name

Attribute selection output

```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 20 Injury Severity):
        Gain Ratio feature evaluator

Ranked attributes:
 0.05316  11 Driver Substance Abuse
 0.04505  12 Non-Motorist Substance Abuse
 0.02156   3 Hours past Midnight
 0.01765   5 Related Non-Motorist
 0.01432  14 Pedestrian Actions
 0.01354  15 Pedestrian Location
 0.01233   6 Collision Type
 0.01149   1 Agency Name
 0.01087   9 Light
 0.01076   4 Route Type
 0.0101   13 Pedestrian Movement
 0.0093   16 At Fault
 0.00727  10 Traffic Control
 0.00622   7 Weather
 0.00597  17 Safety Equipment
 0.00563   8 Surface Condition
 0        18 Lat- Normalized
 0         2 Days after Jan 1
 0        19 Long- Normalized

Selected attributes: 11,12,3,5,14,15,6,1,9,4,13,16,10,7,17,8,18,2,19 : 19
```

C. OneRAttributeEval with Ranker

a. The OneRAttributeEval method evaluates attributes using the OneR classifier, which creates one rule for each attribute and selects the rule (attribute) with the smallest error rate [3]. We used the Ranker search method with a cutoff threshold of 44.5

b. The pseudocode for the 1R algorithm is as follows [3]:

```
For each attribute
    For each value of the attribute
        count frequency of each class
        find the most frequent class
        make rule: assign that class to this attribute-value
    Compute the error rate of the rules (of this attribute)
Choose the rules with the smallest error rate
```

c. The selected attributes were Pedestrian Location, Traffic Control, Surface Condition, Weather, Light, Agency Name, Safety Equipment, and At Fault

Attribute selection output

```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 20 Injury Severity):
        OneR feature evaluator.

        Using 10 fold cross validation for evaluating attributes.
        Minimum bucket size for OneR: 6

Ranked attributes:
44.954  15 Pedestrian Location
44.594  10 Traffic Control
44.577   8 Surface Condition
44.561   7 Weather
44.545   9 Light
44.545   1 Agency Name
44.545  17 Safety Equipment
44.545  16 At Fault
44.463   4 Route Type
44.463   5 Related Non-Motorist
44.397  13 Pedestrian Movement
44.397  14 Pedestrian Actions
44.397  11 Driver Substance Abuse
44.332  12 Non-Motorist Substance Abuse
44.282   6 Collision Type
40.908  18 Lat- Normalized
40.809  19 Long- Normalized
40.76    2 Days after Jan 1
40.367   3 Hours past Midnight

Selected attributes: 15,10,8,7,9,1,17,16,4,5,13,14,11,12,6,18,19,2,3 : 19
```

D. ReliefFAttributeEval with Ranker

a. The ReliefFAttributeEval method evaluates attributes by repeatedly sampling an instance and considering the value of the given attribute for the nearest instance of the same and different class. This approach is particularly effective for detecting conditional dependencies between attributes. We used the Ranker search method with a cutoff threshold of 0.009 [4]

b. The pseudocode and formulas for the ReliefF algorithm is as follows [4]:

Initialize the weight of attribute A as W(A) = 0

For each randomly selected instance X:

Find the k nearest neighbors from the same class, which is H(X)

Find the k nearest neighbors from a different class, which is $M_c(X)$

For each Attribute A, update the weight using the formula:

$$W(A) = W(A) - \frac{1}{m}\sum_{H(X)} diff(A, X, H(X)) + \frac{1}{m}\sum_{c \neq class(X)} P(c) \sum_{Mc(X)} \frac{1}{k} diff(A, X, M_c(X))$$

Normalize the weight W(A) for each attribute

Where:

- diff(A, X_1, X_2) can be defined as:
  - For continuous attributes: $\frac{|A(X1)-A(X2)|}{max(A)-min(A)}$
  - For nominal attributes: 0 if $A(X_1) = A(X_2)$ otherwise 1
- m is the number of sampled instances
- P(c) is the prior probability of class c

Explanation:

- diff(A, X_1, X_2): A function that measures the difference between the values of attribute A for two instances X_1 and X_2
- H(X): These nearest neighbors, known as "hits," are used to penalizes an attribute if its values differ too much between an instance X and its neighbors from the same class
- M_c(X): These nearest neighbors, known as "misses," are used to reward an attribute if its values differ significantly between X and its neighbors from different classes
- Higher weights indicate more important features for distinguishing between classes

c. The selected attributes were Pedestrian Location, Pedestrian Movement, Traffic Control, Route Type, Pedestrian Actions, Collision Type, and Light.

Attribute selection output

```
Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 20 Injury Severity):
        ReliefF Ranking Filter
        Instances sampled: all
        Number of nearest neighbours (k): 10
        Equal influence nearest neighbours

Ranked attributes:
 0.027354  15 Pedestrian Location
 0.018023  13 Pedestrian Movement
 0.016565  10 Traffic Control
 0.013341   4 Route Type
 0.010529  14 Pedestrian Actions
 0.009833   6 Collision Type
 0.009529   9 Light
 0.008194   5 Related Non-Motorist
 0.006928   2 Days after Jan 1
 0.006194   1 Agency Name
 0.005893   7 Weather
 0.00454   16 At Fault
 0.004455   3 Hours past Midnight
 0.003507  17 Safety Equipment
 0.002649   8 Surface Condition
 0.001727  18 Lat- Normalized
 0.000804  19 Long- Normalized
 0.000581  12 Non-Motorist Substance Abuse
 0.000338  11 Driver Substance Abuse

Selected attributes: 15,13,10,4,14,6,9,5,2,1,7,16,3,17,8,18,19,12,11 : 19
```

E. Our Pick:
   a. For our fifth attribute selection, we selected the attributes which were picked by 2 or more of the above algorithms
   b. The selected attributes: Pedestrian Location, Pedestrian Actions, Collision Type, Related Non-Motorist, Route Type, Hours Past Midnight, Agency Name, Pedestrian Movement, Traffic Control, and Light.

F. Note on BestFirst Subsets
   - It's important to note that we did not employ the BestFirst subsets method in this analysis due to compatibility issues with our specific dataset. BestFirst is typically used for its ability to navigate the attribute subset space using greedy hillclimbing

with backtracking, which can be effective for finding optimal feature subsets. Its exclusion may have limited our ability to identify certain attribute interactions or optimal subsets.

Model Classifiers Used

A. bayes.NaiveBayes

This classifier uses probability theory to classify data based on Bayes' theorem, assuming that the attributes are independent of each other. It calculates the likelihood of each class given the attributes and selects the class with the highest probability [5]:

Compute the probabilities $P(C_1|X)$, $P(C_2|X)$,..,$P(C_m|X)$ to classify X into the class that exhibits the highest probability

$$P(C_i|X) = \frac{P(X|Ci)P(Ci)}{P(X)}$$

Where:

- X is tuple characterized by n attributes, represented as $(x_1, x_2, \ldots, x_n)$, where each x_n signifies the value of attribute A_i
- m is the total number of classes, indicated by $C_1, C_2, .., C_m$
- $P(X|C_i) = \prod_{k=1}^{n} P(X_k|C_i)$

B. trees.J48

This classifier generates a decision tree by recursively splitting the dataset into subsets based on the attribute that provides the highest information gain at each step. Each internal node in the tree represents a test on an attribute, and the branches represent the outcomes of that test. The final leaf nodes indicate the predicted class [6]

C. trees.RandomForest

This classifier builds an ensemble of decision trees, where each tree is constructed by splitting data based on randomly selected attributes. Each internal node in a tree represents a test on one of these attributes, and the branches indicate the possible outcomes of the test. This process continues until a leaf node is reached, which holds the predicted class [7]

D. lazy.KStar

This classifier is an instance-based learning algorithm that classifies new data points by comparing them to existing examples in the dataset using an entropy-based distance measure. KStar calculates the probability of transforming one instance into another through a series of attribute changes, which serves as the similarity function. For each new instance, the classifier identifies the most similar neighbors and assigns a class based on these closest examples [8]

Results

The preprocessed data was split into Train/Test by writing a script on Google Colab, which is provided below.

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Step 1: Read in the CSV file
data = pd.read_csv('/content/drive/MyDrive/ml 24-25/crash_data.csv')

# Step 2: Split the data into features (X) and labels (y)
X = data.iloc[:, :-1]
y = data.iloc[:, -1]

# Step 3: Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
stratify=y, random_state=21)

train_data = pd.concat([X_train, y_train], axis=1)
test_data = pd.concat([X_test, y_test], axis=1)

train_data.to_csv('train.csv', index=False)
test_data.to_csv('test.csv', index=False)
```

Folder with Train/Test/Validation data: train/test data

InfoGainAttributeEval

A. bayes.NaiveBayes

Classifier output

```
Time taken to build model: 0.04 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.03 seconds

=== Summary ===

Correctly Classified Instances          42               40.3846 %
Incorrectly Classified Instances        62               59.6154 %
Kappa statistic                          0.0205
Mean absolute error                      0.2718
Root mean squared error                  0.378
Relative absolute error                 98.7803 %
Root relative squared error            101.0441 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.143    0.011    0.667      0.143    0.235      0.269    0.577     0.294     Suspecte
                 0.000    0.000    ?          0.000    ?          ?        0.776     0.216     Fatal In
                 0.837    0.852    0.409      0.837    0.550      -0.021   0.493     0.435     Suspecte
                 0.000    0.011    0.000      0.000    0.000      -0.034   0.568     0.121     No Appar
                 0.121    0.113    0.333      0.121    0.178      0.012    0.529     0.355     Possible
Weighted Avg.    0.404    0.391    ?          0.404    ?          ?        0.532     0.351

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  2  0 11  0  1 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  0  0 36  1  6 |  c = Suspected Minor Injury
  0  0 10  0  1 |  d = No Apparent Injury
  1  0 28  0  4 |  e = Possible Injury
```

B. trees.J48

Classifier output

```
Time taken to build model: 0.18 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances          41               39.4231 %
Incorrectly Classified Instances        63               60.5769 %
Kappa statistic                          0.0073
Mean absolute error                      0.2719
Root mean squared error                  0.3835
Relative absolute error                 98.8407 %
Root relative squared error            102.5006 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.071    0.033    0.250      0.071    0.111      0.068    0.584     0.234     Suspecte
                 0.000    0.000    ?          0.000    ?          ?        0.728     0.063     Fatal In
                 0.837    0.820    0.419      0.837    0.558      0.023    0.605     0.520     Suspecte
                 0.000    0.000    ?          0.000    ?          ?        0.561     0.130     No Appar
                 0.121    0.141    0.286      0.121    0.170      -0.027   0.514     0.329     Possible
Weighted Avg.    0.394    0.388    ?          0.394    ?          ?        0.572     0.367

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  1  0 10  0  3 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  1  0 36  0  6 |  c = Suspected Minor Injury
  0  0 10  0  1 |  d = No Apparent Injury
  2  0 27  0  4 |  e = Possible Injury
```

C. trees.RandomForest

Classifier output

```
Time taken to build model: 3.14 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.07 seconds

=== Summary ===

Correctly Classified Instances          28               26.9231 %
Incorrectly Classified Instances        76               73.0769 %
Kappa statistic                         -0.0748
Mean absolute error                      0.2741
Root mean squared error                  0.4142
Relative absolute error                 99.6446 %
Root relative squared error            110.698  %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.071    0.100    0.100      0.071    0.083      -0.033   0.698     0.224     Suspecte
                 0.000    0.000    ?          0.000    ?          ?        0.677     0.110     Fatal In
                 0.349    0.475    0.341      0.349    0.345      -0.126   0.493     0.416     Suspecte
                 0.182    0.086    0.200      0.182    0.190      0.100    0.730     0.226     No Appar
                 0.303    0.423    0.250      0.303    0.274      -0.114   0.468     0.358     Possible
Weighted Avg.    0.269    0.353    ?          0.269    ?          ?        0.543     0.343

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  1  0  7  1  5 |  a = Suspected Serious Injury
  0  0  1  2  0 |  b = Fatal Injury
  4  0 15  2 22 |  c = Suspected Minor Injury
  0  0  6  2  3 |  d = No Apparent Injury
  5  0 15  3 10 |  e = Possible Injury
```

D. lazy.KStar

Classifier output

```
Time taken to build model: 0 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.62 seconds

=== Summary ===

Correctly Classified Instances          34               32.6923 %
Incorrectly Classified Instances        70               67.3077 %
Kappa statistic                         -0.0764
Mean absolute error                      0.2716
Root mean squared error                  0.3802
Relative absolute error                 98.7094 %
Root relative squared error            101.6223 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.033    0.000      0.000    0.000      -0.068   0.638     0.201     Suspecte
                 0.000    0.000    ?          0.000    ?          ?        0.812     0.122     Fatal In
                 0.605    0.721    0.371      0.605    0.460      -0.122   0.437     0.432     Suspecte
                 0.000    0.011    0.000      0.000    0.000      -0.034   0.717     0.282     No Appar
                 0.242    0.310    0.267      0.242    0.254      -0.069   0.478     0.321     Possible
Weighted Avg.    0.327    0.402    ?          0.327    ?          ?        0.517     0.341

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 10  0  4 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  1  0 26  0 16 |  c = Suspected Minor Injury
  0  0  9  0  2 |  d = No Apparent Injury
  2  0 22  1  8 |  e = Possible Injury
```

II. GainRatioAttributeEval

A. bayes.NaiveBayes

Classifier output

```
(nominal) Non-notorist Substance Abuse                  --> 6 (nominal) Non-notorist Substance Abuse
(nominal) Pedestrian Actions                    --> 7 (nominal) Pedestrian Actions
(nominal) Pedestrian Location                   --> 8 (nominal) Pedestrian Location
(nominal) Injury Severity                       --> 9 (nominal) Injury Severity


Time taken to build model: 0.05 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.02 seconds

=== Summary ===

Correctly Classified Instances          47               41.5929 %
Incorrectly Classified Instances        66               58.4071 %
Kappa statistic                         -0.0287
Mean absolute error                      0.2663
Root mean squared error                  0.3716
Relative absolute error                 97.8846 %
Root relative squared error            100.42   %
Total Number of Instances              113

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.597     0.201     Suspected Serious Injury
                 0.333    0.009    0.500      0.333    0.400      0.395    0.809     0.216     Fatal Injury
                 0.882    0.952    0.433      0.882    0.581      -0.127   0.522     0.477     Suspected Minor Injury
                 0.000    0.010    0.000      0.000    0.000      -0.031   0.619     0.138     No Apparent Injury
                 0.030    0.063    0.167      0.030    0.051      -0.065   0.586     0.378     Possible Injury
Weighted Avg.    0.416    0.449    ?          0.416    ?          ?        0.568     0.372

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 14  0  1 |  a = Suspected Serious Injury
  0  1  2  0  0 |  b = Fatal Injury
  0  1 45  1  4 |  c = Suspected Minor Injury
  0  0 11  0  0 |  d = No Apparent Injury
  0  0 32  0  1 |  e = Possible Injury
```

B. trees.J48

Classifier output

```
(nominal) Non-notorist Substance Abuse                  --> 6 (nominal) Non-notorist Substance Abuse
(nominal) Pedestrian Actions                    --> 7 (nominal) Pedestrian Actions
(nominal) Pedestrian Location                   --> 8 (nominal) Pedestrian Location
(nominal) Injury Severity                       --> 9 (nominal) Injury Severity


Time taken to build model: 0.09 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances          48               42.4779 %
Incorrectly Classified Instances        65               57.5221 %
Kappa statistic                         -0.0261
Mean absolute error                      0.2704
Root mean squared error                  0.3758
Relative absolute error                 99.3842 %
Root relative squared error            101.5634 %
Total Number of Instances              113

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.577     0.156     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.650     0.038     Fatal Injury
                 0.922    0.968    0.439      0.922    0.595      -0.102   0.551     0.506     Suspected Minor Injury
                 0.091    0.010    0.500      0.091    0.154      0.182    0.603     0.199     No Apparent Injury
                 0.000    0.050    0.000      0.000    0.000      -0.123   0.520     0.304     Possible Injury
Weighted Avg.    0.425    0.452    ?          0.425    ?          ?        0.553     0.358

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 15  0  0 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  0  0 47  0  4 |  c = Suspected Minor Injury
  0  0 10  1  0 |  d = No Apparent Injury
  0  0 32  1  0 |  e = Possible Injury
```

C. trees.RandomForest

Classifier output

```
(nominal) Non-notorist Substance Abuse                  --> 6 (nominal) Non-notorist Substance Abuse
(nominal) Pedestrian Actions                    --> 7 (nominal) Pedestrian Actions
(nominal) Pedestrian Location                   --> 8 (nominal) Pedestrian Location
(nominal) Injury Severity                       --> 9 (nominal) Injury Severity


Time taken to build model: 6.05 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.07 seconds

=== Summary ===

Correctly Classified Instances          42               37.1681 %
Incorrectly Classified Instances        71               62.8319 %
Kappa statistic                          0.0249
Mean absolute error                      0.2639
Root mean squared error                  0.3971
Relative absolute error                 97.016  %
Root relative squared error            107.3008 %
Total Number of Instances              113

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.067    0.031    0.250      0.067    0.105      0.066    0.605     0.188     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.755     0.068     Fatal Injury
                 0.471    0.484    0.444      0.471    0.457      -0.013   0.513     0.529     Suspected Minor Injury
                 0.091    0.029    0.250      0.091    0.133      0.099    0.736     0.282     No Apparent Injury
                 0.485    0.438    0.314      0.485    0.381      0.043    0.538     0.352     Possible Injury
Weighted Avg.    0.372    0.353    ?          0.372    ?          ?        0.561     0.396

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  1  0  8  0  6 |  a = Suspected Serious Injury
  0  0  2  0  1 |  b = Fatal Injury
  3  0 24  2 22 |  c = Suspected Minor Injury
  0  0  4  1  6 |  d = No Apparent Injury
  0  0 16  1 16 |  e = Possible Injury
```

D. lazy.KStar

Classifier output

```
(nominal) Non-notorist Substance Abuse                  --> 6 (nominal) Non-notorist Substance Abuse
(nominal) Pedestrian Actions                    --> 7 (nominal) Pedestrian Actions
(nominal) Pedestrian Location                   --> 8 (nominal) Pedestrian Location
(nominal) Injury Severity                       --> 9 (nominal) Injury Severity


Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.94 seconds

=== Summary ===

Correctly Classified Instances          45               39.823  %
Incorrectly Classified Instances        68               60.177  %
Kappa statistic                         -0.0503
Mean absolute error                      0.2631
Root mean squared error                  0.3663
Relative absolute error                 96.7027 %
Root relative squared error             99.0004 %
Total Number of Instances              113

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.593     0.197     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.873     0.409     Fatal Injury
                 0.804    0.903    0.423      0.804    0.554      -0.142   0.498     0.535     Suspected Minor Injury
                 0.091    0.000    1.000      0.091    0.167      0.288    0.746     0.357     No Apparent Injury
                 0.091    0.150    0.200      0.091    0.125      -0.079   0.561     0.338     Possible Injury
Weighted Avg.    0.398    0.451    ?          0.398    ?          ?        0.563     0.412

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 15  0  0 |  a = Suspected Serious Injury
  0  0  2  0  1 |  b = Fatal Injury
  0  0 41  0 10 |  c = Suspected Minor Injury
  0  0  9  1  1 |  d = No Apparent Injury
  0  0 30  0  3 |  e = Possible Injury
```

III. OneRAttributeEval

A. bayes.NaiveBayes

Classifier output

```
(nominal) At Fault              --> 7 (nominal) At Fault
(nominal) Safety Equipment      --> 8 (nominal) Safety Equipment
(nominal) Injury Severity       --> 9 (nominal) Injury Severity


Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances          53               43.4426 %
Incorrectly Classified Instances        69               56.5574 %
Kappa statistic                         -0.0211
Mean absolute error                      0.2707
Root mean squared error                  0.3722
Relative absolute error                 99.6341 %
Root relative squared error            100.7202 %
Total Number of Instances              122

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.627     0.317     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.804     0.099     Fatal Injury
                 0.911    0.955    0.447      0.911    0.600      -0.088   0.491     0.508     Suspected Minor Injury
                 0.000    0.009    0.000      0.000    0.000      -0.029   0.602     0.133     No Apparent Injury
                 0.057    0.057    0.286      0.057    0.095      -0.001   0.550     0.354     Possible Injury
Weighted Avg.    0.434    0.455    ?          0.434    ?          ?        0.546     0.392

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 15  0  1 |  a = Suspected Serious Injury
  0  0  4  0  0 |  b = Fatal Injury
  0  0 51  1  4 |  c = Suspected Minor Injury
  0  0 11  0  0 |  d = No Apparent Injury
  0  0 33  0  2 |  e = Possible Injury
```

B. trees.J48

Classifier output

```
(nominal) At Fault              --> 7 (nominal) At Fault
(nominal) Safety Equipment      --> 8 (nominal) Safety Equipment
(nominal) Injury Severity       --> 9 (nominal) Injury Severity


Time taken to build model: 0.04 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances          56               45.9016 %
Incorrectly Classified Instances        66               54.0984 %
Kappa statistic                          0
Mean absolute error                      0.2716
Root mean squared error                  0.3695
Relative absolute error                 99.9818 %
Root relative squared error            100.0005 %
Total Number of Instances              122

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.131     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.033     Fatal Injury
                 1.000    1.000    0.459      1.000    0.629      ?        0.500     0.459     Suspected Minor Injury
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.090     No Apparent Injury
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.287     Possible Injury
Weighted Avg.    0.459    0.459    ?          0.459    ?          ?        0.500     0.319

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 16  0  0 |  a = Suspected Serious Injury
  0  0  4  0  0 |  b = Fatal Injury
  0  0 56  0  0 |  c = Suspected Minor Injury
  0  0 11  0  0 |  d = No Apparent Injury
  0  0 35  0  0 |  e = Possible Injury
```

C. trees.RandomForest

```
Classifier output
(nominal) Pedestrian Location      --> 6 (nominal) Pedestrian Location
(nominal) At Fault                 --> 7 (nominal) At Fault
(nominal) Safety Equipment         --> 8 (nominal) Safety Equipment
(nominal) Injury Severity          --> 9 (nominal) Injury Severity


Time taken to build model: 1.51 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.03 seconds

=== Summary ===

Correctly Classified Instances          56               45.9016 %
Incorrectly Classified Instances        66               54.0984 %
Kappa statistic                          0.1128
Mean absolute error                      0.2558
Root mean squared error                  0.3675
Relative absolute error                 94.1664 %
Root relative squared error             99.4487 %
Total Number of Instances              122

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.063    0.028    0.250      0.063    0.100      0.065    0.651     0.281     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.743     0.202     Fatal Injury
                 0.768    0.652    0.500      0.768    0.606      0.127    0.609     0.573     Suspected Minor Injury
                 0.182    0.054    0.250      0.182    0.211      0.148    0.701     0.288     No Apparent Injury
                 0.286    0.161    0.417      0.286    0.339      0.142    0.623     0.423     Possible Injury
Weighted Avg.    0.459    0.354    ?          0.459    ?          ?        0.631     0.454

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  1  0 11  1  3 |  a = Suspected Serious Injury
  1  0  2  0  1 |  b = Fatal Injury
  1  0 43  5  7 |  c = Suspected Minor Injury
  0  0  6  2  3 |  d = No Apparent Injury
  1  0 24  0 10 |  e = Possible Injury
```

D. lazy.KStar

```
Classifier output
(nominal) Pedestrian Location      --> 6 (nominal) Pedestrian Location
(nominal) At Fault                 --> 7 (nominal) At Fault
(nominal) Safety Equipment         --> 8 (nominal) Safety Equipment
(nominal) Injury Severity          --> 9 (nominal) Injury Severity


Time taken to build model: 0 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 1.09 seconds

=== Summary ===

Correctly Classified Instances          58               47.541  %
Incorrectly Classified Instances        64               52.459  %
Kappa statistic                          0.0624
Mean absolute error                      0.2667
Root mean squared error                  0.3657
Relative absolute error                 98.1704 %
Root relative squared error             98.964  %
Total Number of Instances              122

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.643     0.286     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.815     0.151     Fatal Injury
                 0.946    0.864    0.482      0.946    0.639      0.139    0.608     0.569     Suspected Minor Injury
                 0.091    0.000    1.000      0.091    0.167      0.289    0.711     0.266     No Apparent Injury
                 0.114    0.080    0.364      0.114    0.174      0.053    0.577     0.388     Possible Injury
Weighted Avg.    0.475    0.420    ?          0.475    ?          ?        0.620     0.439

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 14  0  2 |  a = Suspected Serious Injury
  0  0  4  0  0 |  b = Fatal Injury
  0  0 53  0  3 |  c = Suspected Minor Injury
  0  0  8  1  2 |  d = No Apparent Injury
  0  0 31  0  4 |  e = Possible Injury
```

IV. ReliefFAttributeEval

A. bayes.NaiveBayes

Classifier output

```
(nominal) Pedestrian Actions    --> 6 (nominal) Pedestrian Actions
(nominal) Pedestrian Location   --> 7 (nominal) Pedestrian Location
(nominal) Injury Severity       --> 8 (nominal) Injury Severity


Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances          42               40.3846 %
Incorrectly Classified Instances        62               59.6154 %
Kappa statistic                          0.0264
Mean absolute error                      0.2731
Root mean squared error                  0.3795
Relative absolute error                 99.2686 %
Root relative squared error            101.432  %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.143    0.011    0.667      0.143    0.235      0.269    0.581     0.316     Suspected Serious Injury
                 0.000    0.010    0.000      0.000    0.000      -0.017   0.856     0.132     Fatal Injury
                 0.814    0.852    0.402      0.814    0.538      -0.051   0.474     0.417     Suspected Minor Injury
                 0.000    0.011    0.000      0.000    0.000      -0.034   0.515     0.117     No Apparent Injury
                 0.152    0.099    0.417      0.152    0.222      0.077    0.540     0.352     Possible Injury
Weighted Avg.    0.404    0.387    0.388      0.404    0.325      0.035    0.525     0.343

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  2  0 11  0  1 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  1  1 35  1  5 |  c = Suspected Minor Injury
  0  0 10  0  1 |  d = No Apparent Injury
  0  0 28  0  5 |  e = Possible Injury
```

B. trees.J48

Classifier output

```
(nominal) Pedestrian Actions    --> 6 (nominal) Pedestrian Actions
(nominal) Pedestrian Location   --> 7 (nominal) Pedestrian Location
(nominal) Injury Severity       --> 8 (nominal) Injury Severity


Time taken to build model: 0.08 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

nning classification
ified Instances          43               41.3462 %
ssified Instances        61               58.6538 %
Kappa statistic                          0
Mean absolute error                      0.2751
Root mean squared error                  0.3741
Relative absolute error                 99.9833 %
Root relative squared error            100.0018 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.135     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.029     Fatal Injury
                 1.000    1.000    0.413      1.000    0.585      ?        0.500     0.413     Suspected Minor Injury
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.106     No Apparent Injury
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.317     Possible Injury
Weighted Avg.    0.413    0.413    ?          0.413    ?          ?        0.500     0.302

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0 14  0  0 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  0  0 43  0  0 |  c = Suspected Minor Injury
  0  0 11  0  0 |  d = No Apparent Injury
  0  0 33  0  0 |  e = Possible Injury
```

C. trees.RandomForest

Classifier output

```
(nominal) Pedestrian Actions    --> 6 (nominal) Pedestrian Actions
(nominal) Pedestrian Location   --> 7 (nominal) Pedestrian Location
(nominal) Injury Severity       --> 8 (nominal) Injury Severity


Time taken to build model: 3.46 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.02 seconds

=== Summary ===

Correctly Classified Instances          35               33.6538 %
Incorrectly Classified Instances        69               66.3462 %
Kappa statistic                         -0.0432
Mean absolute error                      0.2677
Root mean squared error                  0.3897
Relative absolute error                 97.3187 %
Root relative squared error            104.158  %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.143    0.033    0.400      0.143    0.211      0.175    0.558     0.212     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.774     0.166     Fatal Injury
                 0.628    0.721    0.380      0.628    0.474      -0.099   0.543     0.490     Suspected Minor Injury
                 0.000    0.054    0.000      0.000    0.000      -0.077   0.645     0.166     No Apparent Injury
                 0.182    0.239    0.261      0.182    0.214      -0.065   0.497     0.360     Possible Injury
Weighted Avg.    0.337    0.384    ?          0.337    ?          ?        0.548     0.368

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  2  0  9  0  3 |  a = Suspected Serious Injury
  0  0  2  0  1 |  b = Fatal Injury
  1  0 27  3 12 |  c = Suspected Minor Injury
  0  0 10  0  1 |  d = No Apparent Injury
  2  0 23  2  6 |  e = Possible Injury
```

D. lazy.KStar

Classifier output

```
(nominal) Pedestrian Actions    --> 6 (nominal) Pedestrian Actions
(nominal) Pedestrian Location   --> 7 (nominal) Pedestrian Location
(nominal) Injury Severity       --> 8 (nominal) Injury Severity


Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 1.19 seconds

=== Summary ===

Correctly Classified Instances          43               41.3462 %
Incorrectly Classified Instances        61               58.6538 %
Kappa statistic                          0.0319
Mean absolute error                      0.2687
Root mean squared error                  0.3752
Relative absolute error                 97.6873 %
Root relative squared error            100.2747 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.071    0.000    1.000      0.071    0.133      0.250    0.573     0.254     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.731     0.175     Fatal Injury
                 0.837    0.803    0.424      0.837    0.563      0.043    0.533     0.488     Suspected Minor Injury
                 0.000    0.000    ?          0.000    ?          ?        0.668     0.246     No Apparent Injury
                 0.182    0.169    0.333      0.182    0.235      0.016    0.503     0.365     Possible Injury
Weighted Avg.    0.413    0.386    ?          0.413    ?          ?        0.549     0.383

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  1  0 11  0  2 |  a = Suspected Serious Injury
  0  0  2  0  1 |  b = Fatal Injury
  0  0 36  0  7 |  c = Suspected Minor Injury
  0  0  9  0  2 |  d = No Apparent Injury
  0  0 27  0  6 |  e = Possible Injury
```

V. Our Pick

A. bayes.NaiveBayes

Classifier output

```
(nominal) Traffic Control              --> 7 (nominal) Traffic Control
(nominal) Pedestrian Movement          --> 8 (nominal) Pedestrian Movement
(nominal) Pedestrian Actions           --> 9 (nominal) Pedestrian Actions
(nominal) Pedestrian Location          --> 10 (nominal) Pedestrian Location
(nominal) Injury Severity              --> 11 (nominal) Injury Severity


Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances          41               39.4231 %
Incorrectly Classified Instances        63               60.5769 %
Kappa statistic                          0.0183
Mean absolute error                      0.2697
Root mean squared error                  0.3765
Relative absolute error                 98.0193 %
Root relative squared error            100.624  %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.143    0.000    1.000      0.143    0.250      0.355    0.594     0.294     Suspected Serious Injury
                 0.333    0.010    0.500      0.333    0.400      0.394    0.881     0.250     Fatal Injury
                 0.744    0.836    0.386      0.744    0.508      -0.113   0.489     0.450     Suspected Minor Injury
                 0.000    0.011    0.000      0.000    0.000      -0.034   0.595     0.152     No Apparent Injury
                 0.182    0.141    0.375      0.182    0.245      0.053    0.569     0.391     Possible Injury
Weighted Avg.    0.394    0.392    0.427      0.394    0.333      0.026    0.551     0.373

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  2  0 12  0  0 |  a = Suspected Serious Injury
  0  1  2  0  0 |  b = Fatal Injury
  0  1 32  1  9 |  c = Suspected Minor Injury
  0  0 10  0  1 |  d = No Apparent Injury
  0  0 27  0  6 |  e = Possible Injury
```

B. trees.J48

Classifier output

```
(nominal) Pedestrian Actions           --> 9 (nominal) Pedestrian Actions
(nominal) Pedestrian Location          --> 10 (nominal) Pedestrian Location
(nominal) Injury Severity              --> 11 (nominal) Injury Severity


Time taken to build model: 0.21 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances          41               39.4231 %
Incorrectly Classified Instances        63               60.5769 %
Kappa statistic                          0.0036
Mean absolute error                      0.2666
Root mean squared error                  0.3795
Relative absolute error                 96.8919 %
Root relative squared error            101.4329 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.071    0.011    0.500      0.071    0.125      0.150    0.567     0.219     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.757     0.067     Fatal Injury
                 0.814    0.885    0.393      0.814    0.530      -0.100   0.542     0.477     Suspected Minor Injury
                 0.091    0.011    0.500      0.091    0.154      0.179    0.637     0.182     No Apparent Injury
                 0.121    0.099    0.364      0.121    0.182      0.034    0.602     0.391     Possible Injury
Weighted Avg.    0.394    0.400    ?          0.394    ?          ?        0.581     0.372

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  1  0 12  0  1 |  a = Suspected Serious Injury
  0  0  3  0  0 |  b = Fatal Injury
  1  0 35  1  6 |  c = Suspected Minor Injury
  0  0 10  1  0 |  d = No Apparent Injury
  0  0 29  0  4 |  e = Possible Injury
```

C. trees.RandomForest

Classifier output

```
(nominal) Pedestrian Actions            --> 9 (nominal) Pedestrian Actions
(nominal) Pedestrian Location           --> 10 (nominal) Pedestrian Location
(nominal) Injury Severity               --> 11 (nominal) Injury Severity


Time taken to build model: 4.62 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.03 seconds

=== Summary ===

Correctly Classified Instances          39               37.5    %
Incorrectly Classified Instances        65               62.5    %
Kappa statistic                          0.0805
Mean absolute error                      0.2612
Root mean squared error                  0.3993
Relative absolute error                 94.9458 %
Root relative squared error            106.7205 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.286    0.067    0.400      0.286    0.333      0.254    0.621     0.277     Suspected Serious Injury
                 0.000    0.010    0.000      0.000    0.000      -0.017   0.723     0.219     Fatal Injury
                 0.488    0.393    0.467      0.488    0.477      0.094    0.571     0.462     Suspected Minor Injury
                 0.364    0.054    0.444      0.364    0.400      0.339    0.781     0.424     No Apparent Injury
                 0.303    0.408    0.256      0.303    0.278      -0.101   0.456     0.357     Possible Injury
Weighted Avg.    0.375    0.307    0.375      0.375    0.373      0.076    0.568     0.393

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  4  0  6  0  4 |  a = Suspected Serious Injury
  0  0  2  1  0 |  b = Fatal Injury
  2  1 21  0 19 |  c = Suspected Minor Injury
  0  0  1  4  6 |  d = No Apparent Injury
  4  0 15  4 10 |  e = Possible Injury
```

D. lazy.KStar

Classifier output

```
(nominal) Pedestrian Actions            --> 9 (nominal) Pedestrian Actions
(nominal) Pedestrian Location           --> 10 (nominal) Pedestrian Location
(nominal) Injury Severity               --> 11 (nominal) Injury Severity


Time taken to build model: 0 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 2.43 seconds

=== Summary ===

Correctly Classified Instances          43               41.3462 %
Incorrectly Classified Instances        61               58.6538 %
Kappa statistic                          0.0814
Mean absolute error                      0.2639
Root mean squared error                  0.3794
Relative absolute error                 95.909  %
Root relative squared error            101.4134 %
Total Number of Instances              104

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.143    0.022    0.500      0.143    0.222      0.214    0.606     0.264     Suspected Serious Injury
                 0.000    0.000    ?          0.000    ?          ?        0.713     0.391     Fatal Injury
                 0.628    0.574    0.435      0.628    0.514      0.054    0.513     0.495     Suspected Minor Injury
                 0.091    0.022    0.333      0.091    0.143      0.128    0.816     0.393     No Apparent Injury
                 0.394    0.310    0.371      0.394    0.382      0.083    0.479     0.311     Possible Injury
Weighted Avg.    0.413    0.341    ?          0.413    ?          ?        0.553     0.392

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  2  0  9  0  3 |  a = Suspected Serious Injury
  0  0  2  1  0 |  b = Fatal Injury
  1  0 27  0 15 |  c = Suspected Minor Injury
  0  0  6  1  4 |  d = No Apparent Injury
  1  0 18  1 13 |  e = Possible Injury
```

Analysis
Unfortunately, all of our models had low accuracy and none were really efficient. However, certain models did significantly worse (under 40% accuracy):

- J48 with InfoGain
- RandomForestwith InfoGain
- KStar with InfoGain J48
- RandomForest with GainRatio
- KStar with GainRatio
- RandomForest with ReliefF
- NaiveBayes with Our Pick
- J48 with Our Pick
- RandomForest with Our Pick

RandomForest classifiers performed the worst in the majority of the models, except OneR. Using the OneRAttributeEval selection did much better than the other selections on average. The best model based on accuracy was KStar classifier with OneRAttributeEval Selection with an Accuracy rate of 47.541%, but it also had a False Positive Rate of 0.420, an ROC area of 0.620, and a Recall of 0.459. While the RandomForest classifier with OneRAttributeEval Selection had an accuracy rate of 45.9%, it had a lower False Positive rate of 0.354, a higher ROC area of 0.631, and a higher Recall of 0.475, which are indicative of better performance. Therefore, the model we choose as our best is the **RandomForest with OneRAttributeEval Selection**.

The attributes of the OneRAttributeEval are as follows: Pedestrian Location, Traffic Control, Surface Condition, Weather, Light, Agency Name, Safety Equipment, At Fault

<u>Selected Model: RandomForest with OneRAttributeEval Selection</u>

Given the overall performance of our final selected model, it is evident that it is not yet suitable for real-world applications. One possible explanation for this underperformance could be the lack of interaction between attributes. The model might not be capturing complex relationships between features, which could be crucial for improvising predictive accuracy.

Additionally, similarity between class labels could have contributed to the underperformance. Some labels represent subtle differences in Injury Severity, making it difficult for models to accurately distinguish between them. For example, Possible Injury vs. Suspected Minor Injury can make it hard for the model to draw clear boundaries. These ambiguously adjacent labels could have increased the likelihood of misclassification. Moreover, Injury Severity can be subjective as two observers may label the same case differently. This may have added noise and led to inconsistencies in the data.

Conclusion
As mentioned above, the RandomForest model with OneRAttributeEval Selection had the best results of the 20 runs for this project. We successfully trained and tested a classification model, but the overall performance remains below the standard for real-world application.
For future iterations, we could use Principal Component Analysis as our method for attribute selection, using linear combination of attributes to capture the relationships between attributes.

Furthermore, we could merge ambiguous adjacent labels to simplify the classification task or explore ordinal classification methods, which recognize the inherent order among labels. This would enable the model to make more informed distinctions between injury severity, reducing confusion between adjacent labels and improving predictive accuracy.

How to Reproduce Our Model

1. open this folder with all of our data for this project: ML Q1 Project - Kaavya, Radin
2. download crash_data.csv
3. run the colab script to get train/test data, and download those files as train.csv and test.csf respectively
4. open crash_data.csv in weka and in the "Select attributes" tab, select OneRAttributeEval and use Ranker as a search method
    a. If "Injury Severity" is not already the class, then go to the Preprocess tab, click "Edit" and then right click on "Injury Severity" and select "Attribute as class." Do this for all future files as well if Injury Severity is not already selected as the class
5. Run this, and select all attributes with a cutoff value of 44.5 or higher
6. open test.csv, and remove all attributes that were not selected by the OneRAttributeEval. save this as test1R.csv
7. open train.csv, and remove all attributes that were not selected by the OneRAttributeEval. save this as train1R.csv
8. go to the "classify" tab, and choose Random Forest (under trees)
9. select "Supplied test set", choose test1R.csv, make sure "Injury Severity" is the class, and press start. If a warning comes up, press yes.

Team Members and Tasks Performed

Kaavya Borra

- Pre-Processing
- Results
- Analysis
- Conclusion/How to Reproduce Our Model

Radin Rezanezhad

- Statement/Project Goal
- Description of Dataset
- Attribute Selection Algorithms
- Model Classifiers Used

Description of Files

Main Folder: ML Q1 Project - Kaavya, Radin

- **INITIAL DATA** - initial dataset
- **preprocessed data** - the data after preprocessing, before filling missing values
- **crash_data.csv**: this is the preprocessed data WITH the missing values filled in from WEKA, and this is what is used to make the train/test data
- **train/test data**: all the training and testing data sets for each attribute selection algorithm are here, under the folders of their respective names. train.csv and test.csv represent the train/test data before the non selected attributes were removed

Appendix and Sources

[1]https://www.math.unipd.it/~aiolli/corsi/0708/IR/Lez12.pdf
[2]https://machinewithdata.com/2018/07/10/how-to-calculate-gain-ratio/
[3]https://docs.google.com/presentation/d/1EVpCV7gaNnLJH57u1d31efTZkt_e7XhjcH5hNQkz5mY/edit#slide=id.g301dfb43887_0_0
[4]https://medium.com/@yashdagli98/feature-selection-using-relief-algorithms-with-python-example-3c2006e18f83
[5]https://docs.google.com/presentation/d/1tKY3oMEc-nU6EvOo5EtUFwRsmbiCdxbi-XP59-X9qZc/edit#slide=id.g17b314ab605_0_0
[6]https://medium.com/@nilimakhanna1/j48-classification-c4-5-algorithm-in-a-nutshell-24c50d20658e
[7]https://builtin.com/data-science/random-forest-algorithm
[8]https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=88b213410192dd8ad8ba5babca8e32dd07cf1b98