CA4010 Data Mining Report

Group 40

Shaun Carey - 16450454 – shaun.carey26@mail.dcu.ie

Nigel Guven – 14493422 - nigel.guven2@mail.dcu.ie

Date of Completion: 9th December 2019

Table of Contents

1. Introduction:

This report is written and designed to present our discoveries from the dataset that we chose to base our prediction on. We are hoping to accurately predict the probability of a driver failing to finish a Formula 1 race based on a wide number of conditions.

The motivation behind this project came from our love of racing, specifically Formula 1, but also because F1 is the most data-driven sport in the world, so it seemed like a no-brainer. Every little detail that happens during the race can be logged and used to the team's benefit. Not to mention the numerous telemetry information screens that the race engineers need to be checking thoroughly throughout each practice session and race. We felt that this would be an incredibly useful prediction tool, the likes of which the teams probably use themselves.

Our hypothesis is that we should look for primary indicators such as weather conditions and the circuits. Secondary indicators may include, drivers, teams and starting positions on the track. While primary indicators are a dead give away of probable cause, we want to examine secondary indicators so that we may produce interesting results by mining more interesting information and developing clearly outlined facts with which we can hold against our hypothesis.

1.1 *Materials*

The dataset that we worked from in this project was taken from the following URL:

https://www.kaggle.com/cjgdev/formula-1-race-data-19502017/

although we had to modify it accordingly. The dataset contains all race results, race information, circuit information, driver information, team information, amongst other tables, from 1950 until 2017. We manually created our own test set by adding in race results and any new entity information from 2018 onward and examined the results from these seasons.
Analysis of the data was carried out using Microsoft Excel and RStudio.

1.2 *Dataset*

The data was downloaded in the form of CSV files, with each concurrent table having its own CSV file. There were a huge number of attributes contained within the 'Race Results' table, and we dug out the ones which would have the most bearing on whether a driver finished a race or not. These attributes include weather, grid position of the driver, team of the driver, the circuit, the driver himself, and the end result; i.e. whether the driver finished or not. We made sure that each attribute removed would have zero effect on our prediction – more on this later.

As such, 'end result' turns out to be the class in that table that we are actually predicting. Initially, there were hundreds of end results, ranging from 'Finished', to 'Spun out of the race', to 'Clutch failure', etc. We decided to boil all of these down to only 3 separate classes: 'Finished', 'Car Failure', or 'Crashed', which was done manually. Whilst having a more specific result might be helpful, there were many rare end results which only appeared one or two times and thus wouldn't be very helpful to us. On top of that, we were more-so only concerned with whether the driver actually finished or not, but decided to separate the cases where a driver didn't finish into the two main causes: a crash,

which is normally the fault of the driver, and a mechanical failure, usually brought on by a fault from the team's end of things.

2. Data Preparation

2.1 *Preprocessing*

Our dataset was fantastic, but almost contained *too* much information! Our primary goal was to combine all relevant information into a single CSV table, but first we had to remove completely irrelevant attributes and tables. Firstly, we decided to only take the entries from 2003 onwards. This is a result of the Formula 1 cars becoming incredibly more reliable over the years, thus we felt that including results from decades ago would only give us inaccurate information based on the modern-day cars.

Before eliminating any attributes or tables, we made sure that anything being removed is deemed completely useless regarding finding our end prediction of DNF probability. The tables which we deleted included:

- constructorStandings (the teams' standings throughout the years)
- driverStandings
- pitStops (pit stop info for each race, including time spent stopped etc.)
- lapTimes (the time of each drivers' lap in every race they've driven in)
- qualifying (drivers' qualifying info)

This still left 8 tables of relevant information for us to play around with. However, the table that we were most concerned with was 'Race Results', which contained all information that you would expect from the name: driver ID, team ID, circuit ID, weather ID, what position they started in, what position they finished in (if they did finish), etc.

Although, there were also multiple attributes which we felt were useless for our prediction. These included values such as the driver's fastest lap in that particular race, or their pitstop times, which would have zero bearing on our end result: simply whether they finished the race, crashed out of the race, or had a mechanical problem.

I did mention a lot of identifier attributes there however, and that is where the problem of making a single table comes in. Most of the other tables in the dataset were tables which just simply linked up each numeric identifier with a whole other plethora of information. For example, the 'Drivers' table had a 'driverID' for every driver, which linked that ID to their personal information such as their first name, last name, and date of birth.

We preferred to list our data as categorical data to make it easier for us to mine for information whilst also still being machine-readable. Since our 'Race Results' table was jam packed with identifiers and was extremely human-unreadable, we took it upon ourselves to forge our own, much more readable table using Power Query. See Figure 2.1 and 2.2 for a small Before & After snippet!

Figure 2.1 Before Integration

resultId	raceId	driverId	constructo	circuitId	grid	weatherIc	statusId
7554	1	18	23	1	1	1	1
7555	1	22	23	1	2	1	1
7556	1	15	7	1	20	1	1
7557	1	10	7	1	19	1	1
7558	1	4	4	1	10	1	1
7559	1	3	3	1	5	1	1
7560	1	67	5	1	13	1	1
7561	1	7	5	1	17	1	1
7562	1	16	10	1	16	1	1
7563	1	2	2	1	9	1	1
7564	1	21	10	1	15	1	1
7565	1	17	9	1	8	1	1
7566	1	20	9	1	3	1	3
7567	1	9	2	1	4	1	3
7568	1	8	6	1	7	1	2
7569	1	13	6	1	6	1	2
7570	1	12	4	1	14	1	3
7571	1	6	3	1	11	1	3

Figure 2.2 After Integration

Driver	Team	Start_Pos	Circuit	Weather	Result
Michael Schumacher	Ferrari	1	A1-Ring	Dry	Finished
Kimi Raikkonen	McLaren	2	A1-Ring	Dry	Finished
Juan Pablo Montoya	Williams	3	A1-Ring	Dry	Car Failure/Retired
Nick Heidfeld	Sauber	4	A1-Ring	Dry	Car Failure/Retired
Rubens Barrichello	Ferrari	5	A1-Ring	Dry	Finished
Jarno Trulli	Renault	6	A1-Ring	Dry	Finished
Jenson Button	BAR	7	A1-Ring	Dry	Finished
Antonio Pizzonia	Jaguar	8	A1-Ring	Dry	Finished
Giancarlo Fisichella	Jordan	9	A1-Ring	Dry	Car Failure/Retired
Ralf Schumacher	Williams	10	A1-Ring	Dry	Finished
Olivier Panis	Toyota	11	A1-Ring	Dry	Car Failure/Retired
Jacques Villeneuve	BAR	12	A1-Ring	Dry	Finished
Cristiano da Matta	Toyota	13	A1-Ring	Dry	Finished
David Coulthard	McLaren	14	A1-Ring	Dry	Finished
Heinz-Harald Frentzen	Sauber	15	A1-Ring	Dry	Car Failure/Retired

2.2 *Attributes and Descriptions*

The following table describes the full set of attributes that may be associated with a given entity in the finalised pre-processed data set. This set of attributes were selected from the entire set because under our assumptions, these specific attributes were the relevant items in mining results according to our goal of the prediction of a driver receiving a DNF in their associated race.

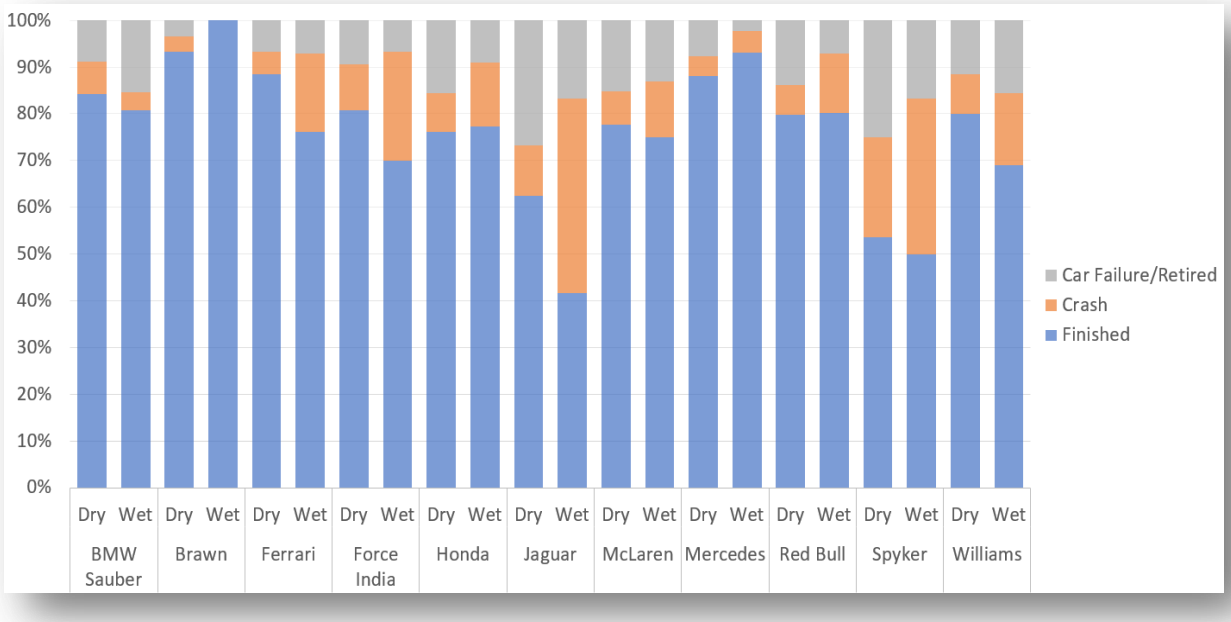
2.3 *Workshop Insights*

Our selected workshop was Workshop 5 – Integration and Transformation. This workshop provided

Attribute Identifier	Type of Variable	Description of Variable
Driver	[**First Name && Last Name**]	96 Formula 1 drivers between 2003 and 2018
Team	[**Team Name**]	29 Teams between 2003 and 2018
Circuit	[**Name of Circuit**]	From a total of 68 locations
Position	[**1-20**]	Starting position on a track
Weather	[**Dry**] [**Wet**]	Weather conditions during the race
Status	[**Finished**] [**Car Failure/Retired**] [**Crash**]	Result being either Finished or DNF

us with valuable retrospectivity with regards to data preprocessing. The knowledge gained from the processes of Integration and Transformation helped to reshape our data into a unified, feasibly operable dataset. This refurbished dataset provided us with the ability to perform small, localised tests on our trained, white-box data. This white box data contains racing statistics ranging from 2003 to 2018.

Figure 2.3 Team Subset Results Percentage



This interesting information was gathered from a subset of teams over the trained dataset. Some interesting points are listed below from the statistical set of results:

Result Type	Dry	Wet	Total
Car Failure/Retired	786	109	895
Crash	429	137	566
Finished	4267	658	4925
Total	**5482**	**904**	**6386**

- Most teams perform better in dry weather, a given.

- The chance of a crash is much higher in wet weather which is 15.1% against 7.8% in dry weather, double the amount, understandable given the driving conditions

- An interesting point to make is that car failure due to mechanical etc. is roughly homogeneous across both wet and dry weather at 12.7% for the former and 14.3% for the latter. Why there are more failures in dry weather lies in the fact that there are far more data available for dry weather races which is bound to cause a skew.

- If one looks at the table, one can view that the team named 'Brawn' have a 100% record in wet weather. This is most likely that this team only ever participated in one race under wet conditions.

Our next step from this workshop would be to develop an algorithm based on Bayesian classification. We could run our processed dataset and the newly added black-box training set through this algorithm which could provide us with our goal state and to visualise some interesting comparisons. The black box training set is taken from the most recent Formula 1 season, 2019.

3. Algorithm

3.1 *Description*

Naïve Bayesian Classification

Naïve Bayesian classifiers are probabilistic classifiers which can classify data using conditional probabilities against Bayes theorem. They assume that the effect of a given attribute value on a class is independent of other attribute values. In order to use this classifier with our dataset, we must assume that no pair of entity values are dependent on one another. Each attribute is essentially given the same importance and are all contributing equally to a specific outcome.

Why did we choose this algorithm?

We chose this classification algorithm for the following reasons:

- With regards to classification, naïve Bayes are comparable to the more popular decision tree classifiers in terms of performance and efficiency.
- During the lecture itself on classification, we pointed out how naïve Bayes would fit our training and testing model quite well. This was later proved to be true, as with later research we found out:
    - Naïve Bayes works very well with text-based classifiers, of which our dataset was mainly composed.
    - There were plenty of libraries for naïve Bayesian classification in the R language which we could utilise and some functions only requiring a few lines of code.
- Bayesian classifiers have a minimum error rate similar to other algorithms and their computation is not too expensive as we have seen when we tried to perform Bayesian Belief networks which did not work according to our goals.

The actual implementation was written in the R language, specifically on RStudio, a perfect platform for us to see visualisations of our dataset and R being one of the most popular languages for data mining projects. See the next section for the full implementation of the Naïve Bayesian classification algorithm.

3.2 *Implementation*

The following pictures and descriptions are code snippets of the whole file. Each snippet contains a section of the algorithm with a description beneath which explains what each instruction is doing.

Fig 3.1 Libraries

```
#install(naivebayes)
#install(dplyr)
#install(ggplot2)
#install(psych)

library(naivebayes)
library(dplyr)
library(ggplot2)
library(psych)
options(warn=-1)
```

Fig 3.1 Description

We had to call in various helpful libraries which would allow us to perform classification and visualisation techniques. These libraries provide helpful functions like ‘pairs.panels’ and ggplot which allow visualisation using histograms and density charts. We also hid R warning messages which were annoying as these messages were telling us that some entities had zero-valued attributes which of course we knew about but the system believed that this data was erroneous.

Fig 3.2 Data I/O

```
# Data

trainset <- read.csv("trainingset.csv")
testset <- read.csv("testingset.csv")

str(trainset)
xtabs(~position+status, data = trainset)

str(testset)
xtabs(~position+status, data = testset)
```

Fig 3.2 Description

We read in our two files, the training set and the testing set to examine how they would correlate. Two separate cross tabulations are shown in Figure 3.2.1 which present the result of an entity given their starting position on the track from 0 to 24. The right chart is the training data and the left chart is test data.

Fig 3.2.1 Cross Tabulation of Results

```
> xtabs(~position+status, data = trainset)
        status
position Car Failure/Retired Crash Finished
       1                  20    11      268
       2                  20    16      262
       3                  26    20      253
       4                  30    15      254
       5                  39    18      243
       6                  37    20      242
       7                  36    26      238
       8                  33    30      237
       9                  36    28      235
       10                 36    24      239
       11                 46    28      224
       12                 42    41      214
       13                 45    33      222
       14                 43    29      227
       15                 54    32      212
       16                 54    34      210
       17                 59    27      216
       18                 49    39      210
       19                 60    29      205
       20                 56    35      206
       21                 33    14      105
       22                 27     9      115
       23                  5     4       47
       24                  9     4       41
```

Trained set

```
> xtabs(~position+status, data = testset)
        status
position Car Failure/Retired Crash Finished
       1                   0     0       21
       2                   1     1       19
       3                   1     1       19
       4                   1     1       19
       5                   0     2       18
       6                   1     1       19
       7                   2     0       19
       8                   2     2       16
       9                   1     1       19
       10                  1     4       16
       11                  2     2       16
       12                  3     2       17
       13                  1     1       19
       14                  2     2       17
       15                  2     2       17
       16                  2     3       18
       17                  2     1       18
       18                  2     2       16
       19                  2     0       18
       20                  0     0       23
```

Test set

We can see some interesting patterns emerging from these tables. The further the starting vehicle on the track from the first position, the more likely they are to retire from the race. This can be seen in the trained set with the number of car failures incrementing as the position goes further back. It appears it is more likely that a driver will finish a race without fault if they at the head of the pack.

Note that most modern Formula 1 races have 20 vehicles as opposed to 24 in previous races. Therefore, there are no results for 21 – 24 from the test data and this might possibly skew our data.

Fig 3.3 Visualisation

```
#Visualisation

pairs.panels(trainset[-1])
trainset %>%
          ggplot(aes(x=weather,y=position,fill = status)) +
          geom_boxplot() +
          ggtitle("BOXPLOT")

trainset %>%  ggplot(aes(x=position, fill = status)) +
          geom_density(alpha=0.8,color = 'blue') +
          ggtitle("DENSITY")

pairs.panels(testset[-1])
testset %>%
          ggplot(aes(x=weather,y=position,fill = status)) +
          geom_boxplot() +
          ggtitle("BOXPLOT")

testset %>%  ggplot(aes(x=position, fill = status)) +
          geom_density(alpha=0.8,color = 'blue') +
          ggtitle("DENSITY")
```

Fig 3.3 Description

In order to view some actual representations of our data in graphs and charts, we must convert the data from table formats into visualised data. Four functions, two for each create Boxplots and Density plots.

The two items Figures 3.3.1 and 3.3.2 show close correlation between the possibility of an incident occurring at the later positions of the track judging by two peaks seen in the graphs. There is also an interesting pattern with regarding a driver finishing a race given that they are near the front of the pack. This can be seen with the amount of high 'blue' data from the 1st to the 6th position. There is some overlapping as can be seen in both plots which may indicate more interesting patterns.

Fig 3.3.1 Trained Density Plot

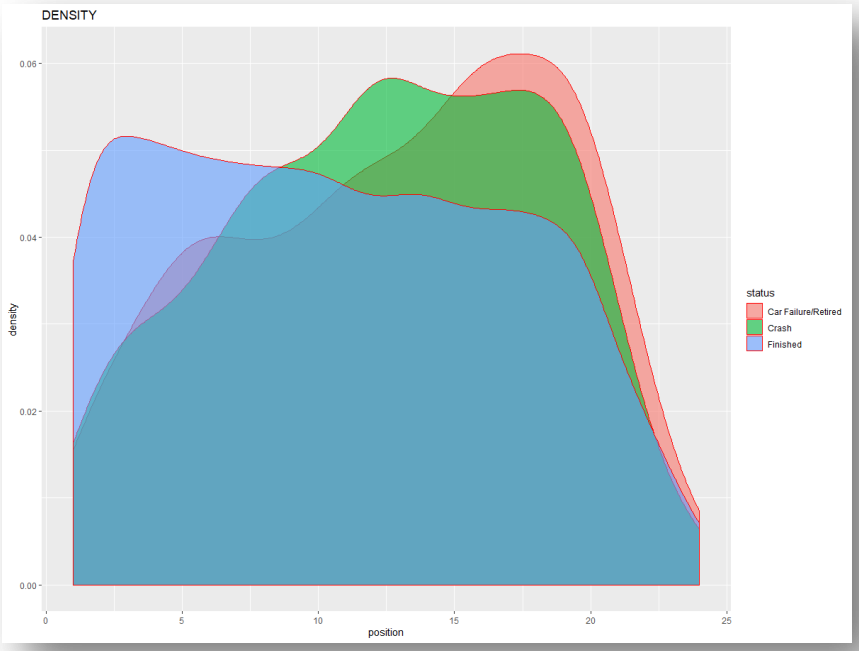


Fig 3.3.2 Test set Density Plot

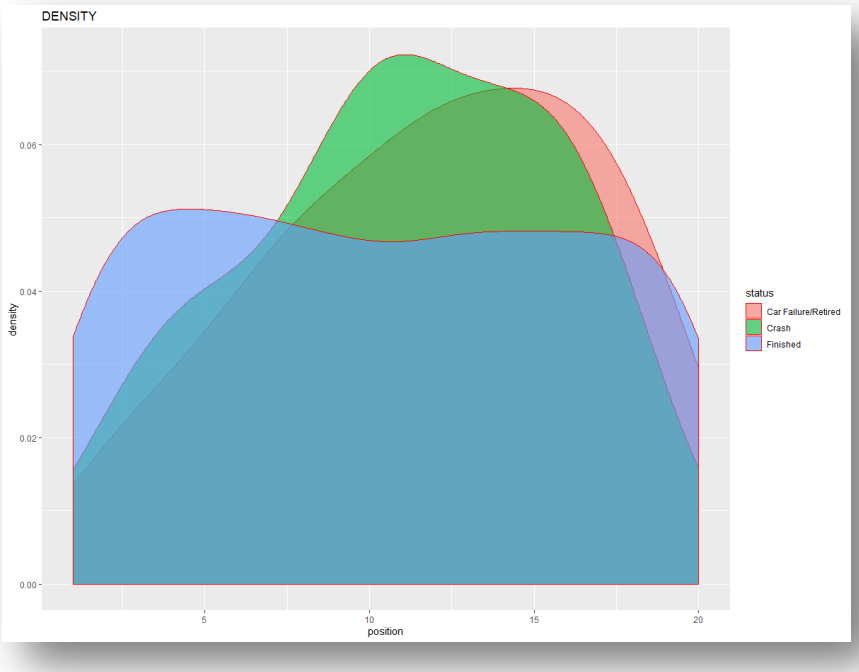


Fig 3.4 Data Partitioning

```
53  # Data Partition
54
55  set.seed(6386)
56  index <- sample(2, nrow(trainset),replace = TRUE, div = c(0.8,0.2))
57
58  train <- trainset[index== 1,]
59  test <- trainset[index==2,]
```

Fig 3.4 Description

In order to develop a strong classification and prediction routine, we partitioned the data into separate groups. The partitions were made randomly across the entire set with replacement. The sample taken was divided into two different sets, one of which was would be our training partition and hold 80% of the sample and the other being the testing data and hold the remaining 20%.

Note that the seed was changed at various stages to improve accuracy. See below.

Fig 3.5 Naïve Bayesian Classification and Prediction

```
61  # Naive Bayes Classification
62
63  model <- naive_bayes(status~., data = train, laplace = 1, usekernel = TRUE)
64
65  train %>%
66    filter(status=="Car Failure/Retired",position=="14")
67
68  plot(model)
69
70  # Prediction
71
72  predictor1 <- predict(model,train,type='div')
73  head(cbind(predictor1,train))
```

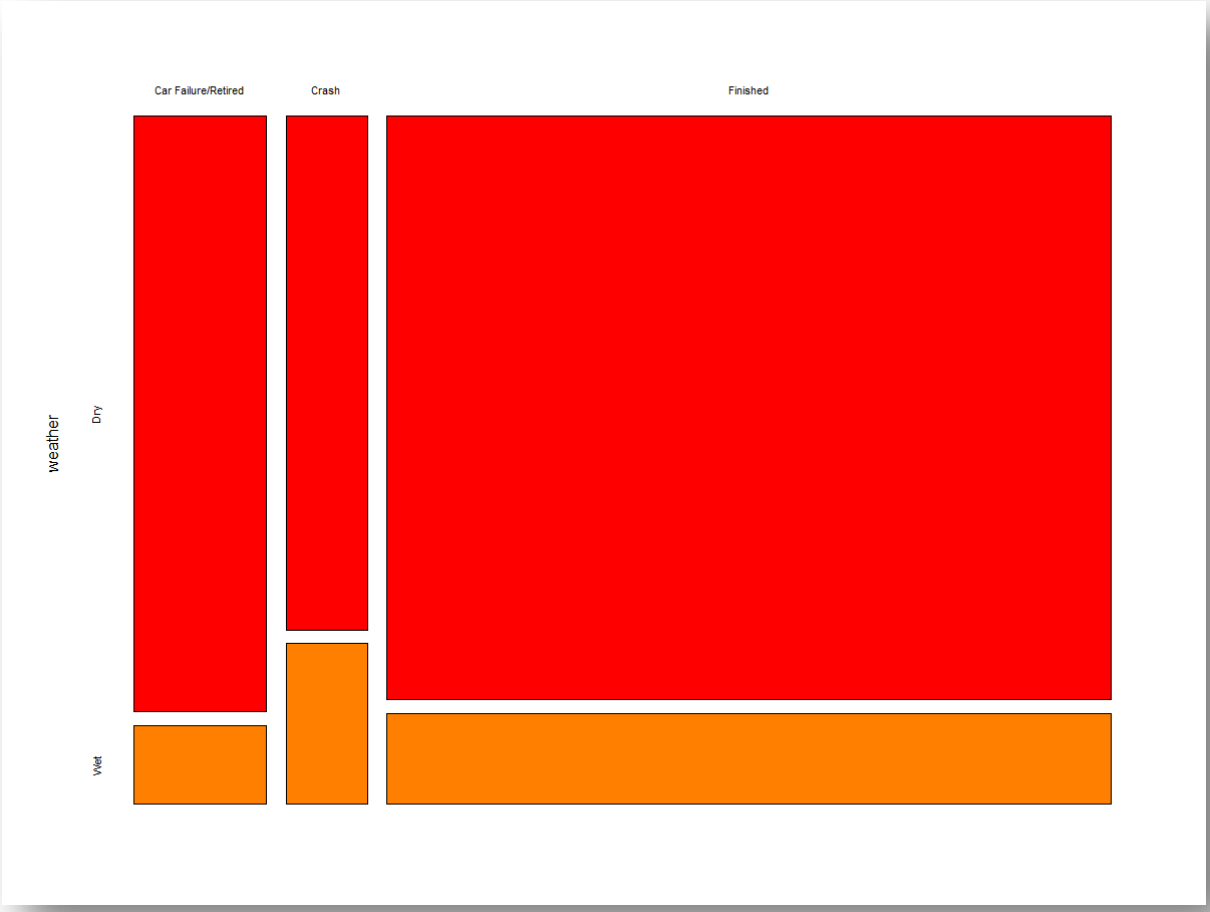
Fig 3.5 Description

We used a naïve Bayes function to create conditional probabilities from the data with respect to each single value based on their relation to the status being either ‘Crash’, ‘Finished’ or ‘Car Failure’. We used Laplace smoothing which was quite necessary as we did have some values that would end up as zeroed out probabilities. The conditional probabilities can be used to test an example class.

For example, where driver=’Lewis Hamilton’ and weather=’wet’ what is the possibility of him crashing? The answer is 0.0004%, a very small insignificant chance of this result occurring.

An interesting plot can be seen in Fig 3.5.1 which shows the spread of resulting statuses. An interesting development here shows that crashes are higher in wet weather judging by the orange region comparing wet weather to crash results.

Fig 3.5.1 Class Plot Chart



Following the generation of conditional probabilities, the next step is to create a prediction on the training and testing models. Using the NB classifier model and the training set, the probabilities are listed alongside respective attributes and how likely they are to occur given these parameters. This can be seen in Figure 3.5.2.

Fig 3.5.2 Head of Predicted Attributes

	Car Failure/Retired	Crash	Finished	driver	team
1	0.07202254	0.02435151	0.9036260	Michael Schumacher	Ferrari
2	0.15618645	0.01876310	0.8250504	Kimi Raikkonen	McLaren
3	0.24473374	0.03092167	0.7243446	Juan Pablo Montoya	Williams
4	0.32710036	0.02625358	0.6466461	Nick Heidfeld	Sauber
5	0.10347645	0.01840899	0.8781146	Rubens Barrichello	Ferrari
6	0.34614746	0.02445185	0.6294007	Jarno Trulli	Renault

We can now deduce that given that Michael Schumacher is the driver and the team is Ferrari, the chance of a crash occurring is 0.07%. This can be done for all attribute values.

Fig 3.6 Confusion Matrices

```
75   # Confusion Matrix - Trained Data
76
77   predictor2 <- predict(model,train)
78   table1 <- table(predictor2,train$status)
79   1 - sum(diag(table1)) / sum(table1)
80
81   # Confusion Matrix - Test Data
82
83   predictor3 <- predict(model,test)
84   table2 <- table(predictor3,test$status)
85   1 - sum(diag(table2)) / sum(table2)
```

Fig 3.6 Description

In order to test the correctness of our prediction function, we must utilise confusion matrices to see if our predictions are correct. Confusion matrices are performance analysers for classification algorithms to measure their accuracy. We generated two confusion matrices and they can be seen below.

Fig 3.6.1 Training Accuracy – Seed = (6386)

```
predictor2            Car Failure/Retired Crash Finished
  Car Failure/Retired                  86    29      139
  Crash                                10    26       35
  Finished                            624   399     3715

[1] 0.244124
```

Fig 3.6.2 Testing Accuracy – Seed = (1000)

```
predictor3            Car Failure/Retired Crash Finished
  Car Failure/Retired                  18     7       37
  Crash                                 4     2       10
  Finished                            155   106      890

[1] 0.7404394
```

A confusion matrix is made up of four values:

- True Positives: Values that the classifier predicted as a correct status that was that specific status.
- True Negatives: Values that the classifier predicted as an incorrect status that was that specific incorrect status
- False Positives: The cases that the classifier inaccurately predicted were true for that entity
- False Negatives: The cases that the classifier inaccurately predicted were not true for that entity.

We calculated the accuracy of the classification algorithm using an accuracy formula which gave us 0.244% for the training set and 0.236% for the test set. Because they are so close, we presume that

the trained data is similar to our test data. However, the accuracy of this classifier is not efficient as the accuracy formula states that the percentage must be closer to 1 to be optimal. We changed the seed to improve the accuracy and sure enough, it changed with value becoming 0.74%.

4. Conclusion

- In evaluation of this data set, we have gained valuable experience into the world of data mining and how it is useful in finding hidden patterns in large blocks of data. The Formula One dataset has many statistics and analysis to work with that were of benefit to us in finding our goal of a driver not finishing a race given a certain set of circumstances.

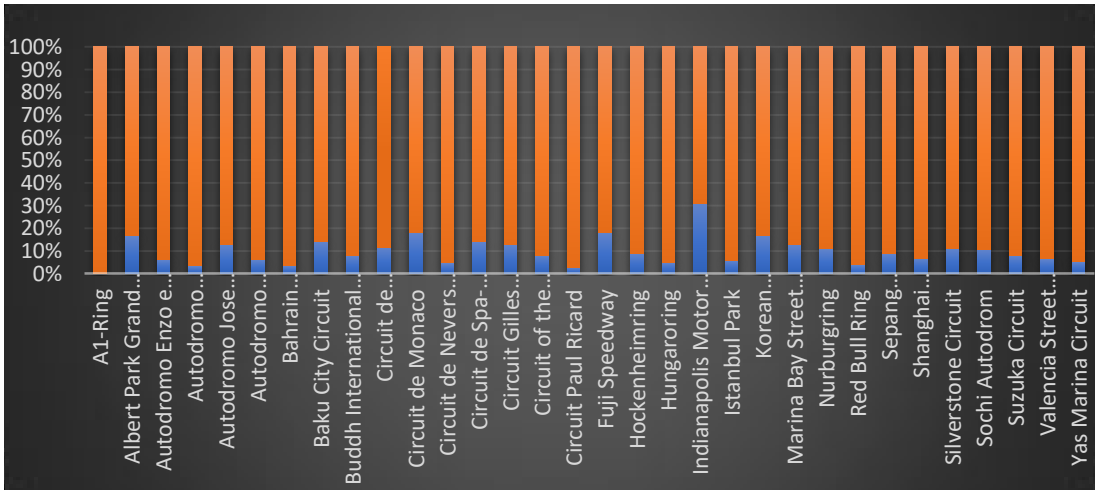
- We received low accuracy for our initial algorithm which made us believe we should change from naïve Bayesian classification to the more popular decision tree classification. Instead of doing this we played around with some of the variables in the algorithm but most of these did not change the accuracy of our confusion matrices.

- When we changed the seed number generator, we did notice that the algorithm improved by quite a lot. Our assumption here is that we were taking in a large chunk of the data as a sample set which was causing these inaccuracies. We therefore believe that the smaller the sample set, the more accurate, the algorithm could be.

- We also worked mainly on comparing the position and weather to the status type as can be seen from most of our graphs. This was due to the fact that they were presented better visually than the driver and circuit attributes. This does not mean that we ignored the latter attributes as they still formed a large part of our research into finding the goal state.

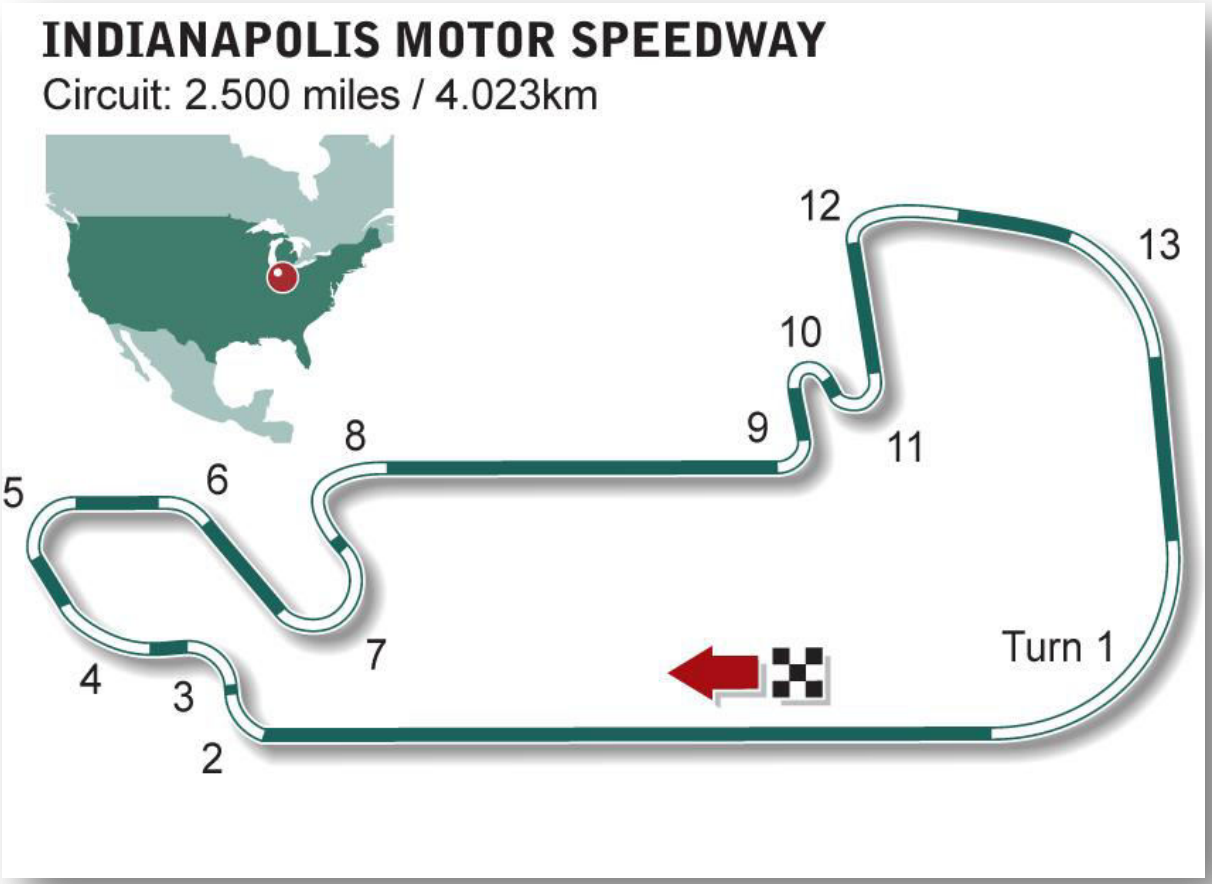
- Considering our discoveries, we can deduce that Formula One races are becoming safer with the rate of crashes from our latest data providing us with less crashes and more finishes. The vehicles themselves are becoming a lot safer and are the designs are improving so that mechanical failures might become less common also.

- We realise that most crashes occur on more difficult circuits like those with sharp turns and narrow tracks. While we can find these tracks in a picture and view that they are quite dangerous, there is no measure for the level of danger on a track and the only way we can visualise it is by creating a graph that shows the percentage between finishes and crashes on



a track

- One course that stands out here is the Indianapolis Speedway which has almost double the crashes of the other races. Here might be the why:



- This picture has given us an idea that we could measure the level of risk on a track by adding a new attribute for circuits which is the number of dangerous bends on a track. Although this is still subjective and will require further analysis outside the time scope of our project. We would have included this as an extra parameter if we were to restart this project.

5. Appendices

Data Mining: Concepts and Techniques, 2nd Edition, Han and Kamber

https://www.youtube.com/watch?v=RLjSQdcg8AM – Naïve Bayes in R with Dr. Bharatendra Rai

http://f1.imgci.com/PICTURES/CMS/4500/4557.jpg - Indianapolis Motor Speedway

https://www.kaggle.com/cjgdev/formula-1-race-data-19502017/ - Kaggle Formula 1 Dataset

-- End of report --

DCU School of Computing
Assignment Submission

Student Name(s):	**Nigel Guven Shaun Carey**
Student Number(s):	**14493422**
Programme:	**BSc in Computer Applications**
Project Title:	**Data Mining Report**
Module code:	**CA4010**
Lecturer:	**Mark Roantree**
Project Due Date:	**18th December 2019**

Declaration

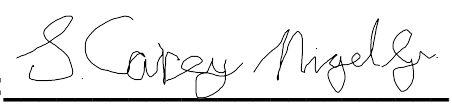
I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying is a grave and serious offence in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion, or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references.

I have not copied or paraphrased an extract of any length from any source without identifying the source and using quotation marks as appropriate. Any images, audio recordings, video or other materials have likewise been originated and produced by me or are fully acknowledged and identified.

This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. I have read and understood the referencing guidelines found at http://www.library.dcu.ie/citing&refguide08.pdf and/or recommended in the assignment guidelines.

I understand that I may be required to discuss with the module lecturer/s the contents of this submission.

I/me/my incorporates we/us/our in the case of group work, which is signed by all of us.

Signed: 

Print this page