COMP 309 — Machine Learning Tools and Techniques
Assignment 3: Kaggle Competition

Student ID: 300417869
Name: Karu Skipper

Core: Exploring and understanding the Kaggle process [50 marks]

• (25 marks) Highlight the findings of your dataset exploration. You should identify each pattern clearly using examples, and discuss the potential consequence this may have on your results.

Business Objective:

Develop the best possible machine learning system to predict the completion times of given tramping tracks in NZ.

Dataset Exporation:

Let's analyse our raw CoreDataSet-Train.csv dataset to understand what we will be investigating in this question.

Id	difficulty	Shape__Length	X	Y	Class
1	Easy	6067.516674	168.6052622	-43.98688424	0
2	Easy	1190.301175	174.9241203	-41.34988879	0
3	Easiest	516.548047	169.5614119	-46.5176704	0
4	Advanced	11422.56164	174.802487	-38.47091454	1
5	Advanced	3330.867075	168.772812	-45.137258	1
6	Easiest	718.0697865	170.1131962	-46.00161732	0
7	Advanced	4557.861652	172.5518639	-40.63409605	0
8	Advanced	2992.429545	177.8007948	-38.98481859	1
9	Advanced	3284.209167	171.22407	-43.63237	1
10	Easiest,Easy	975.8485552	171.3275657	-42.11516818	0
11	Easiest	5921.0808	174.1083856	-39.25751982	0
12		3741.219226	174.4652588	-36.84250676	1
13	Intermediate	71926.56678	172.3028297	-40.88644937	2
14	Advanced,Expert	11579.62589	167.4592865	-45.83573596	1
15	Advanced	32189.20895	171.0091276	-43.10568786	2
16	Advanced	125082.131	172.4395031	-41.31655782	2
17	Advanced	18593.83517	168.9647944	-45.15107353	1
18	Easy	329.6915735	167.1564114	-45.46365128	0
19	Advanced	1195.007303	168.8168876	-45.05986931	1
20	Advanced	35089.00364	174.8585441	-39.34062725	2
21	Easy	1706.756362	172.9402163	-40.83876811	1
22	Easy	1590.882815	172.8618034	-40.88149487	0
23	Easiest	565.4305424	169.4859153	-46.5040545	0
24	Advanced	2946.382344	177.3881146	-38.27966379	1
25	Easy	17562.7012	172.7835552	-34.45340293	1
26	Advanced	4047.873045	176.3006879	-39.62729289	1
27	Intermediate	4833.243419	168.1267263	-44.81812323	1
28	Advanced	4694.175046	170.4064419	-45.81564491	1
29	Easy	719.6814706	173.6275539	-35.23992101	0
30	Easy	3965.725559	176.2644738	-39.72037491	1
31	Expert	16096.67815	172.643854	-42.37819934	1
32	Easy	9261.72863	171.8812823	-41.609507	1

The six attributes of a given instance:
This section was important to investigate as it provides insight before we apply algorithms to the dataset.

1: Id)
Looking at the figure above, the first attribute is the *'id'* of the instance, which lists each instance (tramping track) in order. {1, 2, 3 ... }.

2: Difficulty)
The second column identifies the *'difficulty'* of the track, which classifies each instance into different categories including; { 'Easiest,' 'Easy,' 'Intermediate,' 'Advanced,' or 'Expert.'} *There are also combinations of these categories to help further increase accuracy of the difficulty of the track eg. {'Intermediate, Advanced.'}*

3: Shape Length)
The third column indentifies the *'shape length'* of the instance. This feature will be important to investigate and keep in our dataset as it provides a length of each track. *Which could affect the order a person may finish a given track, changing the time class it may be associated with.*

4 & 5: X and Y)
The next two attributes are X and Y values of each instance. I am unsure how relevant this attribute may be, however it could affect the difficulty of the course.

6: Time Class)
Finally, the attribute in the last column is the 'time class' the instance falls into. This is one of the more important attributes in our investigation. This class ranges from {0 – 2}, based off the time a person may take to complete a given track.

Highlights of the dataset exploration:

- The last attribute '*class*' is of numerical type, this has been changed to nomial so we can classify our data. *Not changing this would mean classification methods would not apply to our dataset, which means we would not be able to make predictions for the time class.*

- I've had to change the combination of difficulty attributes eg. "Advanced, Expert" as converting this to a .csv format confuses what attribute this difficulty is under (because of the comma). *This would mean an extra 'empty' attribute would be added to the end of of the instance. This would incorrectly classify this instance, either ignoring it or adding incorrect values to our classification. Resulting in data loss or even accuracy loss, which is not ideal for our predictions.*

- The X and Y ranges are very different for each instance, regardless of what difficulty the track is. *This means that our model may have a harder time classifying our data. For eg. Instance 1's X and Y values are very similar to Instance 2's X and Y, even though they are the same difficulty. Feeding this into an MLP for example, may change what time class the predicted instance may fall into.*

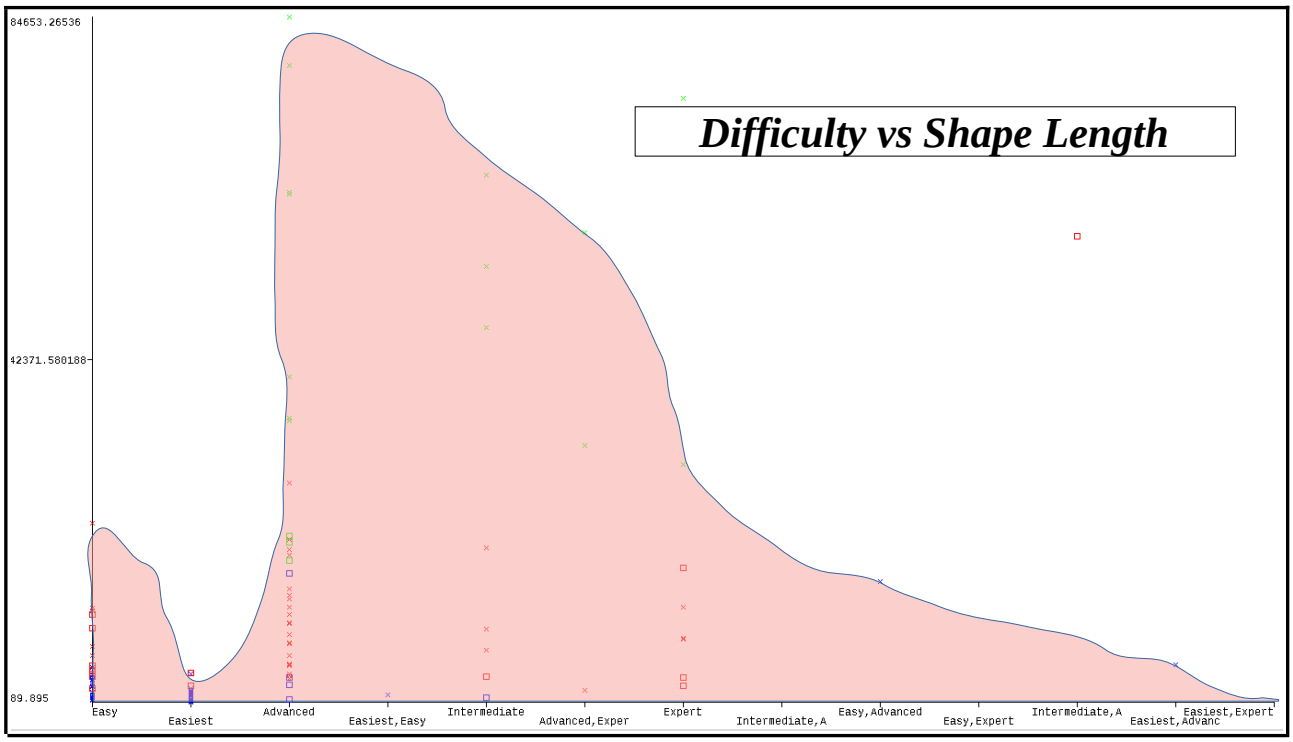
- *I have removed ID from the dataset as it isn't relevant to our investigation of helping us classify the instances.*

Patterns between attributes of an instance (Using WEKA and scikit):
Let's look at patterns between the attributes difficulty and shape_length that help classify our instances.

From quick investigation, I noticed a large amount of '*easier tracks*' seem to fall into the lower-range of the *'shape_length'* attribute and vice versa for '*more advanced*' tracks. (This is with the exception of some outliers).

If we investigate this further, we can visually see the difference between the first '0' time class and the second '1.', based off *difficulty* and *shape_length*. However, it is harder to see the difference between the next time class '2' as the *shape_length* ranges are very similar. This is where other attributes would contribute to helping classify the data.

From the graph below we can see a significant amount of '*advanced*' to '*expert*' instances are greater in '*shape_length*' than easy to easiest.



This graph indicates that harder tracks are usually in the upper-range of *shape_length* and the easy tracks are usually in the *lower-range*.

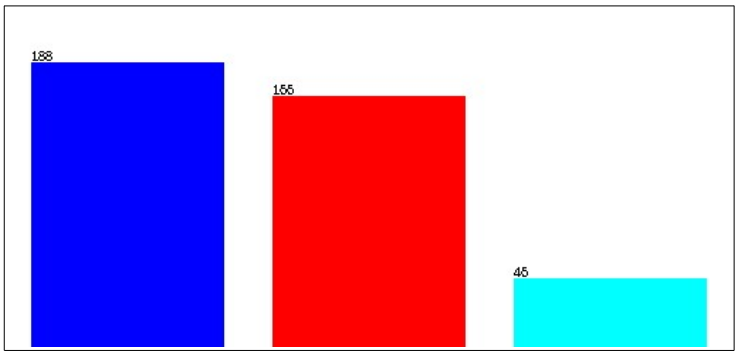
Let's account for other complicated patterns that could impact our investigation.

For this section I am using scikit to quickly analyse unusual patterns that may be occuring.

First, I ran a null scan and found there is only one missing value for difficulty in the dataset. This has no pattern in relation to other attributes in the dataset and will be removed in pre-processing.

Another pattern to note is the class imbalance that is in our training set. This affects the accuracy of our results and can be highly unreliable when testing.

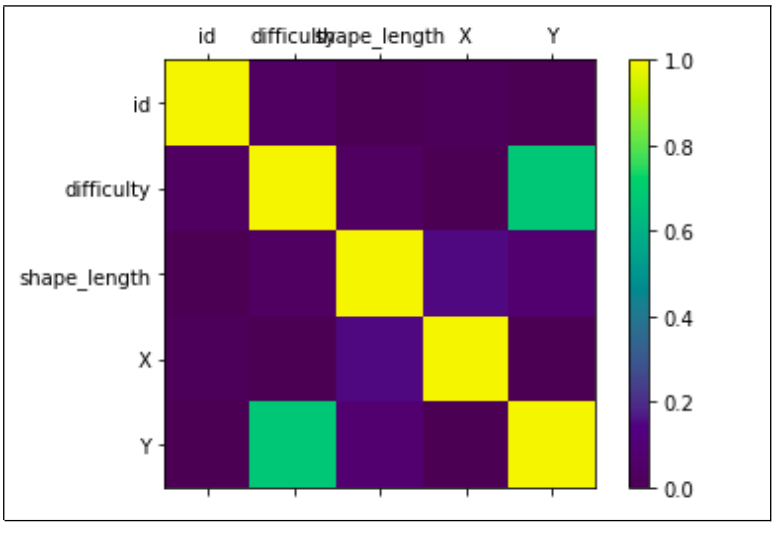
```
In [5]: data.isnull().sum()
Out[5]: Id              0
        difficulty      1
        Shape__Length   0
        X               0
        Y               0
        Class           0
        dtype: int64
```



• (25 marks) Visualisation is an important aspect of this task. Please illustrate at least one important finding of your work.

One important finding of my work;
Below is a Correlation Matrix Plot of every attribute in dataset (I am using the scikit-learn tool to display the graph). This graph will tells us if there are any important correlations between attributes.

Visually, there is a positive correlation of about 0.7 (strength) between the *'difficulty'* of a course and the *'Y value'* of a course. (The higher the y-value, the higher the difficulty). Other than this, there are no visible correlations I can identify from the graph.



To conclude, from this investigation I found '*advanced*' to '*expert*' instances are greater in '*shape_length*' than '*easy*' to '*easiest*'. But also, the instance count is significantly lower for more 'advanced' tracks, which could affect the reliability of these results. I have also found a correlation between the 'difficulty' of a course and the 'Y-value' of a course. This evidence helps our algorithm make predictions of each time class we are classifying the instance into.

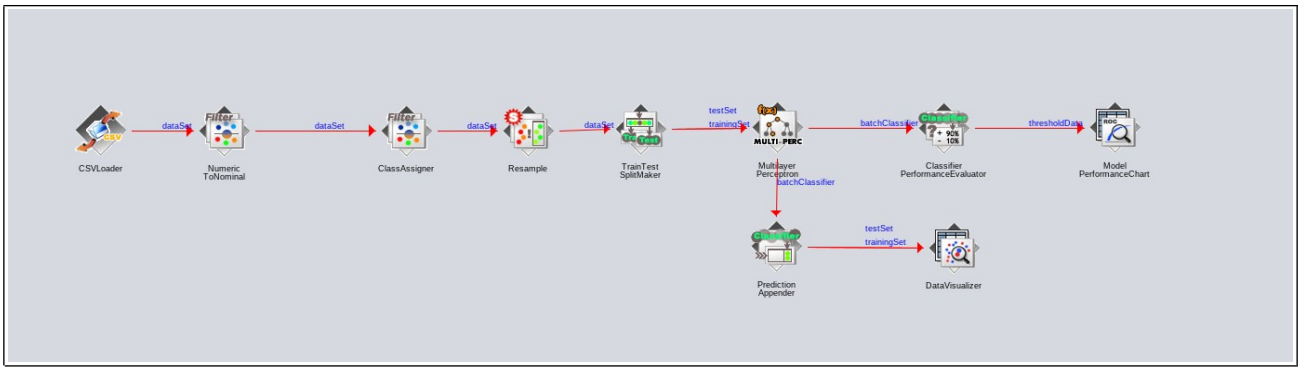
2.3 Completion: Developing and testing your machine learning system [40 marks]

Requirements You should refine your machine learning system a number of times (at least 3, including the initial system) based on the performance you achieve on the public leaderboard. Your submitted report should contain up to 4 pages regarding the challenge component:

• (10 marks) Discuss the initial design of your system, i.e. before you have submitted any predictions to the Kaggle competition. Justify each decision you made in its design, e.g. reference insight you gained in the Core part.

The First System

Initial design in WEKA (Pipeline saved in '/core/Pipelines and Models/')



Note: This is the original system I used for the core part of the assignment, from this I will be altering and adapting it to suit the completion section.

How the core system works:

The dataset is imported into the *CSV-Loader* where we make apply the *NumericToNomial* converter for classifying our data into the specified time class. We then prepare the data using a *ClassAssigner* (Specifically used for identifying the class attribute with in a dataset). Then we *Resample* the data to balance the given instances to make the test more fair, as previously there was a large class imbalance.

The next step is to split the training data using the *TrainTestSplitMaker* so we can have both a training set and a test set. This is where we apply our algorithm, I have chosen MLP for this first system. (This was changed frequently to find the best

possible model for our dataset in later systems). After we apply an algorithm, we use a *Prediction Appender* so we can see our predicted results and use them for our final test set.

Output of .model files for the algorithms I used are saved in the '*/completion/Pipelines and Models/*' directory. The accuracy of th system is show below:

```
Correctly Classified Instances         107               78.6765 %
Incorrectly Classified Instances        29               21.3235 %
Kappa statistic                          0.6373
Mean absolute error                      0.1788
Root mean squared error                  0.3289
Relative absolute error                 44.3349 %
Root relative squared error             73.9345 %
Total Number of Instances              136
```

How we can improve this for the completion section?:
The MLP model I am using outputs an accuracy of about 78.67% with the training set. This model performs well against the online Kaggle Competition (core), with a result of around 84%. However, it is in need of improvements if we are to increase the accuracy of this model. We are also working with two training sets instead of one in the completion section, both containing similar and also different attributes.

I have chosen this initial design because of the simplicity of the data it was loading. This data was easy to identify, small and robust. Meaning it required a simple system to create the desired output model.

Because of this criteria, I could not apply this to the new completion dataset. As described earlier, the new data consisted of multiple attributes with two training sets. This of course does not fit our current systems requirements, thus I made few iterations to the system to ensure we have the best results for this section.

• (20 marks) Discuss the design of one or more of your intermediary systems. Justify the changes you made to the previous design based on its performance on the leaderboard, and from any other additional investigation you performed.

The Second System

The second system included exploring different algorithms that we could apply to our new training datasets. This also included meant we have to apply more pre-processing techniques to our data.

Second System dataset1: Challange-AB-Train1.csv
Second System dataset2: Challange-AB-Train2.csv

Pre-Processing of the Second System:
Removal of unnecessary data:
Before we do this, we must consider the removal of unnecessary attributes and data that could affect our model. Now that we are working with two training sets which contain a large amount of attributes and instances, we need to remove anything that could be taking up space or affecting our results. This was one for the core issues with the dataset provided.
(This included adding an AttributeRemover to the second system).
List of removed attributes from train1.csv

- Name
- Introduction
- hasAlerts
- introductionThumbnail
- walkingAndTrampingWebpage
- dateLoadedToGIS

Refining data into similar completion times:
This was the biggest issue with the provided training datasets. I identified this attribute as 'messy' due to the variation in time completions that contained string extensions, making our data hard to apply an algorithm to.

completionTime
30min
20min
2 hr
3 hr circuit
4 - 5 days
6 - 8 hr
1 hr 30 min
1 hr 30 min
30 min to 1 hr
1 hr return
3 hr
40 min return
20 min - 1 hr

In this second system, I explored familiar techniques like the '*Linear Regression*' algorithm, which is part of the Regression category. I chose this because our data's completion time wasn't categorical, it was numerical based. This should create a better accuracy model.

In doing this, I accounted for walking 'days' as 8 hours, which was then converted into minutes. I repeated this for other values of this attribute, looking for keywords such as 'hr,' 'min,' returning a value in minutes. Here is the function I used for idenfiying certain strings within a cell.

```
=ISNUMBER(SEARCH(substring,text))
```

After this, if this returned true for substrings such as ‘hrs,’ it would be times by 60 to convert it into a seperate time class.

Comparison to the First System:

• (10 marks). Use your judgement to choose the best system you have developed — this may not necessarily be the most accurate system on the leaderboard. Make sure you select this submission as your final one on the competition page before the deadline. Explain why you chose this system, and note any particularly novel/interesting parts of it. You should submit the source and executable code required to run your chosen submission so that the tutors can verify its authenticity. We will select a subset of students to demonstrate in person in ECS computer laboratory, so please make sure your work does not just run on a private desktop machine.