Assignment 2
Machine Learning Modelling

Citations and important disclosures to students:
The following data and scenario are entirely fictitious. They have been created using synthetic data made to match real world scenarios. The data were created using a range of tools including statistical models of real-world security breaches, AI/ML analysis of real-world security breaches.

Background Context
Given recent attacks on the Healthcare sector, and some noted data breaches, FauxCura Health have engaged Quantum.LogiGuardian (Q.LG), a cyber security consultancy and analytics firm.

FauxCura believes they may have had un-detected cyber security breaches within their systems. As a caring and respectable healthcare provider, they want to examine their historic network data to determine whether undetected breaches have occurred.

The security operations centre at FauxCura run a Security Incident and Event Management (SIEM) platform called Splunk™. This platform collects vast quantities of log data from servers, desktop computers, routers and other network equipment and aggregates it in the form of reports and alerts that can be viewed by security personnel to identify incidents that require investigation.

During an initial investigation of FauxCura's data, Q.LG were able to trawl through history data to produce an initial report that provides some top-level metrics on incidents based on certain triggers. Incident specific data is also retained, but generally consists of extremely large data sets.

It is hoped that the report data contains sufficient information to be able to construct an ML model that can more accurately identify events of interest.

Data Overview
The data you are working with are records extracted from FauxCura's SIEM. The records have already been processed and reduced to a summary of individual event detections that were triggered by the SIEM.

The data have also been aggregated from multiple other sources and reports in the SIEM. This means some values may be inconsistent across systems or there may be errors in the data that need to be identified and cleaned.

Descriptions of Features:
Below is a brief explanation of the features in the data set. It is not necessary to understand these features. It is also important to note that feature naming conventions

are very subjective. Reliance on the meaning of a name may miss important data or detail.

Alert Category (Categorical):

This feature describes what type of alert was created by Splunk. It is largely subjective as the alert creators can identify their own alert levels for different types of events. The levels present in the data can be approximately summarised as:

Informational:

An event that is being logged to the system for information purposes only, it is possible these could relate to malicious activity, but this is the lowest level of alert.

Warning:

This is a higher level of alert and typically used to identify a situation that may not be typical.

Alert:

These are typically used for specific events that represent a security concern that requires action.

NetworkEventType (Categorical):

This is the type of event that the SIEM report believes has occurred. It can be used to differentiate between apparently normal network traffic, to things like policy violations and even threat detections and data exfiltration.

NormalOperation:

No specific anomalies occur in this logged event – there are many reasons this data may be logged.

PolicyViolation:

A security or business policy has been violated. This can range from attempts to run unauthorised software on the network, to using the wrong type of web-browser to access a database.

ThreatDetected:

A specific condition has been detected that has previously been identified as a security thread. These could be normal operations mis-tagged, or they may include malicious software or techniques in use.

NetworkInteractionType (Categorical):

This is another 'computer' metric that uses an unknown 3rd party plugin to identify network interactions that are not typical.

Regular:

These appear to be normal network traffic requests.

Elevated:
Requests that are attempting to access resources that require specific permissions. For example, a computer trying to log in to an administrative console or a restricted device.

Suspicious:
Generally, these are elevated network events that are unexpected, have come from an unexpected source, or have unexpected patterns of usage.

Anomalous:
Network interactions that aren't typical but may not have any relation to security events.

Critical:
A network condition that should never occur. This could be an interaction that indicates an attack condition, or a severe equipment outage or malfunction.

Unknown:
The interaction status is unknown

DataTransferVolume (out and in) (Numeric):
Quantifies the amount of data transferred over the network. Values are given whether they are into the network or out of the network.

TransactionsPerSession (Integer):
The number of transactions exchanged between devices and the service they are communicating with.

NetworkAccessFrequency (Integer):
Measures how frequently network ports are accessed, with abnormal frequencies potentially signalling unauthorized access attempts or scans.

UserActivityLevel (Numeric):
A generated metric indicating how active a user is on the system they are connected to. Higher scores generally mean more activity.

SystemAccessRate (Integer):
A generated metric that indicates how frequently the company's core systems are being accessed.

SessionIntegrityCheck (Logical):
A flag that indicates whether the session has been correctly open, communicated and closed, with all underlying network protocols and signals correctly used.

ResourceUtilizationFlag (Logical):
A flag that is raised when the resource utilisation of servers or network devices is unusually high. This could include excessive memory consumption on some devices, slow response times, or large network transfers.

SecurityRiskLevel (Numeric):
A calculated metric created by a 3rd party "AI" plugin that can identify security risks based on unknown parameters and conditions.

ResponseTime (milliseconds) (Numeric):
Measures the time taken to respond to network requests or events. This is the time between when a network resource or event occurs, and the corresponding reply packet is returned.

Classification (Categorical):
The final classification of the event. Where indicated "Normal" and "Malicious" can be assumed to have been identified with reasonable accuracy.

The raw data for the above variables are contained in the **HealthCareData_2024.csv** file.

The needle in the haystack

The data were gathered over a period of time and processed by several systems in order to associate specific events with confirmed malicious activities. However, the number of confirmed malicious events was very low, with these events accounting for approximately 4% of all logged network events.

Although the malicious events are quite uncommon, the identification of malicious events **are extremely important**.

Objectives

You are the data scientist that has been hired by Q.LG to examine the data and provide insights. Your goals will be to

- Clean the data file and prepare it for Machine Learning (ML)
- Recommend a ML algorithm that will provide the most accurate detection of malicious events.
- Create a brief report on your findings

You job
Your job is to develop the detection algorithms that will provide the most accurate incident detection. You do not need to concern yourself about the specifics of the SIEM plugin or software integration, i.e., your task is to focus on **accurate classification of malicious events** using R.

You are to test and evaluate two machine learning algorithms (each in two scenarios) to determine which supervised learning model is best for the task as described.

Task

You are to import and clean the same **HealthCareData_2024.csv**, that was used in the previous assignment. Then run, tune and evaluate two supervised ML algorithms (each with two types of training data) to identify the most accurate way of classifying malicious events.

Part 1 – General data preparation and cleaning

a) Import the **HealthCareData_2024.csv** into R Studio. This version is the same as Assignment 1.

b) Write the appropriate code in R Studio to prepare and clean the **HealthCareData_2024** dataset as follows:

   i. Clean the **whole** dataset based on the feedback received for Assignment 1.

   ii. For the feature **NetworkInteractionType**, merge the '**Regular'** and '**Unknown'** categories together to form the category '**Others'**. **Hint: use the *forcats:: fct_collapse(.)* function.**

   iii. Select only the complete cases using the *na.omit(.)* function, and name the dataset **dat.cleaned.**

   Briefly outline the preparation and cleaning process in your report and why you believe the above steps were necessary.

c) Use the code below to generated two training datasets (one unbalanced **mydata.ub.train** and one balanced **mydata.b.train**) along with the testing set (**mydata.test**). Make sure you enter your student ID into the command *set.seed(.)*.

```
# Separate samples of normal and malicious events
dat.class0 <- dat.cleaned %>% filter(Classification == "Normal") # normal
dat.class1 <- dat.cleaned %>% filter(Classification == "Malicious") # malicious

# Randomly select 9600 non-malicious and 400 malicious samples using your student
ID, then combine them to form a working data set
set.seed(Enter your Student ID)
rows.train0 <- sample(1:nrow(dat.class0), size = 9600, replace = FALSE)
rows.train1 <- sample(1:nrow(dat.class1), size = 400, replace = FALSE)

# Your 10000 'unbalanced' training samples
train.class0 <- dat.class0[rows.train0,] # Non-malicious samples
train.class1 <- dat.class1[rows.train1,] # Malicious samples
mydata.ub.train <- rbind(train.class0, train.class1)

# Your 19200 'balanced' training samples, i.e. 9600 normal and malicious samples e
ach.
set.seed(Enter your Student ID)
```

```
train.class1_2 <- train.class1[sample(1:nrow(train.class1), size = 9600,
                               replace = TRUE),]
mydata.b.train <- rbind(train.class0, train.class1_2)


# Your testing samples
test.class0 <- dat.class0[-rows.train0,]
test.class1 <- dat.class1[-rows.train1,]
mydata.test <- rbind(test.class0, test.class1)
```

Note that in the master data set, the percentage of malicious events is approximately 4%. This distribution is roughly represented by the unbalanced data. The balanced data is generated based on up-sampling of the minority class using bootstrapping. The idea here is to ensure the trained model is not biased towards the majority class, i.e. normal events.

Part 2 – Compare the performances of different ML algorithms

a) Randomly select **two** supervised learning modelling algorithms to test against one another by running the following code. Make sure you enter your student ID into the command *set.seed(.)*. Your 2 ML approaches are given by **myModels**.

```
set.seed(Enter your student ID)
models.list1 <- c("Logistic Ridge Regression",
                  "Logistic LASSO Regression",
                  "Logistic Elastic-Net Regression")
models.list2 <- c("Classification Tree",
                  "Bagging Tree",
                  "Random Forest")
myModels <- c(sample(models.list1, size = 1),
              sample(models.list2, size = 1))
myModels %>% data.frame
```

For each of your two ML modelling approaches, you will need to:

b) Run the ML algorithm in R on the two **training sets** with **Classification** as the outcome variable.

c) Perform hyperparameter tuning to optimise the model:

- Outline your hyperparameter tuning/searching strategy for each of the ML modelling approaches. Report on the search range(s) for hyperparameter tuning, which *k*-fold CV was used, and the number of repeated CVs (if applicable), and the final optimal tuning parameter values and relevant CV statistics (i.e. CV results, tables and plots), where appropriate. **If you are using repeated CVs, a minimum of 2 repeats are required.**
- If your selected tree model is **Bagging**, you must tune the **nbagg**, **cp** and **minsplit** hyperparameters, with **at least 3 values** for each.

- If your selected tree model is **Random Forest**, you must tune the **num.trees** and **mtry** hyperparameters, with **at least 3 values** for each.
- Be sure to set the randomisation seed using your **student ID**.

d) Evaluate the predictive performance of your two ML models, derived from the balanced and unbalanced training sets, on the **testing** set. Provide the confusion matrices and **report and interpret** the following measures in the context of the project:
   - Overall Accuracy
   - Precision
   - Recall
   - F1-score

   Make sure you **define** each of the above metrics in the context of the study. Hint: Use the help menu in R Studio on the *confusionMatrix(.)* function to see how one can obtain the precision, recall and F1-score metrics.

e) Provide a brief statement on your final recommended model and why you have chosen it. This includes explaining which metric(s) you have used in making this decision and why. Parsimony, and to a lesser extent, interpretability maybe taken into account if the decision is close. *You may outline your penalised model estimates in the Appendix if it helps with your argument.*

What to submit

Gather your findings into a report (maximum of 5 pages) and citing relevant sources, if necessary.

Present how and why the data was 'cleaned and prepared', how the ML models were tuned and provide the relevant CV results. Lastly, present how they performed to each other in both the unbalanced and balanced scenarios. You may use graphs, tables and images where appropriate to help your reader understand your findings. All tables and figures should be appropriately captioned, and referenced in-text.

Make a final recommendation on which ML modelling approach is the best for this task.

Your final report should look professional, include appropriate headings and subheadings, should cite facts and reference source materials in APA-7th format.

Your submission must include the following:

- Your report (5 pages or less, **excluding cover/contents/reference/appendix page**). The report must be submitted through **TURNITIN** and checked for originality.
- A copy of your R code, which is to be submitted separately from the report via another submission link.

Make sure you keep a copy of each of the two training sets and a testing set (in .csv format) in case you are asked for them later.

Note that no marks will be given if the results you have provided cannot be confirmed by your code. Furthermore, all pages exceeding the 5-page limit will not be read or examined.

Marking Criteria

Criterion	Contribution to assignment mark
Accurate implementation data cleaning and of each supervised machine learning algorithm in R. • Strictly about code (1) Does the code work from start to finish? (2) Are the results reproducible? (3) External sources of code in APA 7 referencing style (if applicable) (4) Are all the steps performed correctly? (5) Is there good documentation? **Note: At least 80% of the code (excluding those provided to you above) must aligned with unit content. Otherwise a mark of zero will be awarded for this component.**	20%
Explanation of data cleaning and preparation. • Corresponds to Part 1 b) • Briefly outline the reasons for sub-parts (i). • Provide justifications for merging of categories, i.e. sub-part(ii).	10%
An outline of the selected modelling approaches, the hyperparameter tuning and search strategy, the corresponding performance evaluation in the **training sets** (i.e. CV results, tables and plots), and the optimal tuning hyperparameter values. • Penalised logistic regression model – Outline the range of value for your lambda and alpha (if elastic-net). Plot/tabulate the CV results. Outline the optimal value(s)	20%

<table>
<tr><td>of your hyperparameter(s). Outline the coefficients if required for your arguments of model choice.
• Tree models - Outline the range of the hyperparameters (bagging and RF). Tabulate, e.g. the top combinations and the optimal OOB misclassification error, or plot the CV results (e.g. classification tree).</td><td></td></tr>
<tr><td>Presentation, interpretation and comparison of the performance measures (i.e. confusion matrices, accuracy, precision, recall and F1-score) among the selected ML algorithms. Justification of the recommended modelling approach.
• Provide the confusion matrices (frequencies, proportions) in the test set. There should be 4 in total.
<table><tr><td rowspan="2">Predicted</td><td colspan="2">Actual</td></tr><tr><td>Yes</td><td>No</td></tr><tr><td>Yes</td><td>Freq1 (Sensitivity %)</td><td>Freq2 (False positives %)</td></tr><tr><td>No</td><td>Freq3 (False negatives %)</td><td>Freq4 (Specificity %)</td></tr></table>• Outline and interpret the metrics (including accuracy, precision, recall and F1-score) in the context of the study.
• Explain which metric(s) you have used to help you decide your optimal model, and why.</td><td>30%</td></tr>
<tr><td>Report structure and presentation (including tables and figures, and where appropriate, proper citations and referencing in APA-7th style). Report should be clear and logical, well structured, mostly free from communication, spelling and grammatical errors.
• Overall structure, presentation and narrative.
• Referencing
• Table and figures are clear, and properly labelled and referenced in-text.
• No screenshots of R output, except of plots.
• Spelling and grammar.</td><td>20%</td></tr>
</table>

Academic Misconduct

Edith Cowan University regards academic misconduct of any form as unacceptable. Academic misconduct, which includes but is not limited to, plagiarism; unauthorised collaboration; cheating in examinations; theft of other student's work; collusion;

inadequate and incorrect referencing; will be dealt with in accordance with the ECU Rule 40 Academic Misconduct (including Plagiarism) Policy. Ensure that you are familiar with the Academic Misconduct Rules.

Assignment Extensions

Instructions to apply for extensions are available on the ECU Online Extension Request and Tracking System to formally lodge your assignment extension request. The link is also available on Canvas in the Assignment section.

Normal work commitments, family commitments and extra-curricular activities are not accepted as grounds for granting you an extension of time because you are expected to plan ahead for your assessment due dates.

Where the assignment is submitted not more than 7 days late, the penalty shall, for each day that it is late, be 5% of the maximum assessment available for the assignment. Where the assignment is more than 7 days late, a mark of zero shall be awarded.