Using machine learning methods to predict the outcome of UVB psoriasis treatment.

By Ben Hutchings (190080886)

May 2022

G405 MComp Computer Science

Supervisor - Dr Paolo Zuliani

Word count – 15,000

Abstract

In the past few decades, there has been an exponential growth in the capabilities of computers, which has enabled a similar growth in the field of machine learning. AI now drives cars, searches the web, detects spam, and predicts airline travel. The next sector predicted to be revolutionised by machine learning is Healthcare. Healthcare has always been slow to embrace technology so there are many parts which could greatly benefit from technology (Handelman *et al.*, 2018).

With the large time commitment and a variety of other options, phototherapy treatment would greatly benefit from machine learning input. Better predicting the outcome from a few early sessions would save weeks of intense treatment and better inform the patient of their choices. This dissertation explores many prediction methods for psoriasis using a variety of machine learning and data science techniques.

Declaration

I declare that this dissertation represents my own work except where otherwise stated

Acknowledgements

I would like to thank my project supervisor Paolo Zuliani for introducing me to the topic, gathering the dataset and guiding me throughout.

Project Structure

Chapter 1 – Introduction

The purpose of this section is to provide an overview of the dissertation, including the context of the problem, my motivation for choosing this topic and key points about the project.

Chapter 2 – Background

This section explores the research I conducted and the mechanisms of the techniques I will be using. This information will then be applied in later development stages.

Chapter 3 – Analysis Development

This section will discuss the background mechanisms of the analytic techniques and how I applied them to my dataset to achieve the results which I will discuss in chapter 5.

Chapter 4 – Analysis Results

In this section, I will be reviewing the results of the data analytics stage of the development and how the results will inform my decisions when creating models.

Chapter 5 – Model Development

In this section I will discuss the development process of the models and justify the decisions made.

Chapter 6 – Model Evaluation

This section reviews the evaluation of the models created in the model development and compares different configurations for each model.

Chapter 7 – Conclusion

This section gives a review of the whole project including the combined analysis of all models and future work.

Table of Contents

Table Of Figures

Chapter 1– Introduction

1.1 Motivation, Problem and Rationale

Psoriasis is an incurable chronic skin disease which affects around 2/3% of the population (Primary Care Dermatology Society, 2022), but is most common in ages 18-35 (NHS, 2022). The disease usually presents itself as red, flaky, scaly skin but severe forms of the condition can lead to other complications, such as cardiovascular disease and arthritis. Because of the visible nature of the disease, people with severe cases can feel socially stigmatized and have higher rates of depression, anxiety, and suicide (Koo *et al.*, 2017). A 1998 questionnaire of Americans with psoriasis (Krueger *et al.*, 2001) found that 10.7% of participants had issues interacting with their family/spouse and 5.3% contemplated suicide. Research shows that psoriasis patients also have elevated levels of proinflammatory cytokines, which have been linked to increased rates of depression (Koo *et al.*, 2017), suggesting there could be a physiological link between the conditions. Due to the repetitive nature of the disease and expensive treatments, there are also large economic ramifications. The total annual cost of psoriasis in the US is estimated to be $1.6 - $4.3 billion (Bhosle *et al.*, 2006). A study put the average cost of psoriasis of one regimen of phototherapy treatment in the US at $5,713 (Brezinski, Dhillon and Armstrong, 2015). As this is not a cure, this cost is repetitive. A study of remission times found that over half of participants who received phototherapy, without any maintenance therapy, had some form of remission within 137 days of treatment (Koo and Lebwohl, 1999). To live without psoriasis would require three regimens of treatment per year, which would cost around ~$17,000 and up to 198 hours of treatment time a year.

The cause of psoriasis is not fully understood, but research suggests that it is a problem with the immune system. Skin cells are usually replaced on a 3-to-4-week cycle, but in psoriasis patients this process takes around 3 to 7 days. This rapid change means the new cells are not mature enough to be on the surface of the skin and become the flaky scaly skin associated with psoriasis (NHS, 2022). While there is no cure, there are three lines of treatments for the symptoms. The first line is the topical therapies which include salves and creams to help with mild psoriasis. The second line is phototherapy which involves three sessions a week for around 11 weeks (depending on the effectiveness). Finally, the third line of treatment is systemic therapies, which involve immune suppressants and other drugs (NICE, 2022). This last line is very invasive and can open the patient up to many more complications and illnesses.

For this project, I will be trying to make predictions about UVB phototherapy treatments. This treatment involves the patient attending three phototherapy sessions a week for up to 12 weeks, depending on the effectiveness. The therapy involves fully exposing the patient to artificial light for a length of time. There is currently very little research into determining how many sessions the patient will need or how well-processed well the treatment will work. Knowing this information will help a patient better evaluate their treatment options and reduce prescription of ineffective treatment. To achieve this, I am going to use PASI values after each week of treatment and other information about the patient (e.g., age, sex, smoking…). These PASI values are calculated by observing the skin condition and ranking severity. The calculation process is seen in Figure 1,2,3 An example of the PASI trajectory can be seen in Figure 4. To make these predictions I will be using a variety of machine learning techniques to perform analysis on the PASI data in the hope to be able to predict the outcome.

Area of region affected	Area score
None	0
< 10%	1
10 - 29%	2
30 - 49%	3
50 - 69%	4
70 - 89%	5
90 - 100%	6

Figure 1 – Area Score Table

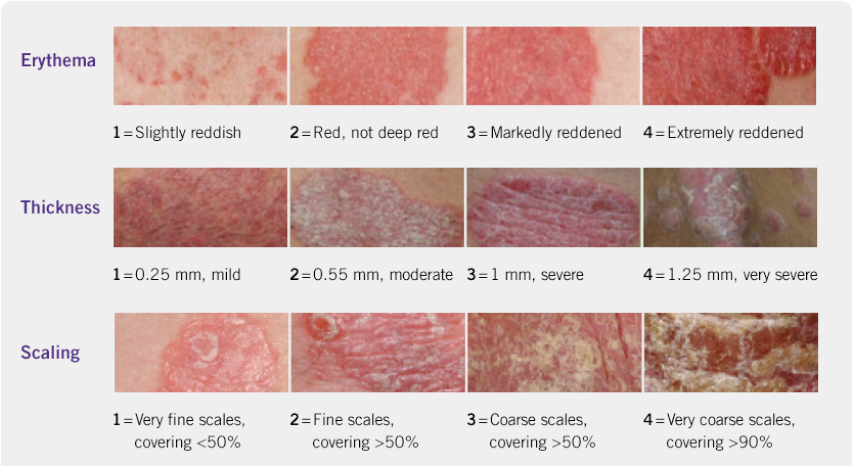


Figure 2 - Psoriasis Symptoms and their scores (Oakley, 2014)

$$C_{body\ area} = (Erythema + Thickness + Scaling) * Area\_score$$
$$PASI = 0.1 * C_{head} + 0.2 * C_{upper\ limbs} + 0.3 * C_{trunk} + 0.4 * C_{lower\ limbs}$$

Figure 3 - PASI Equations

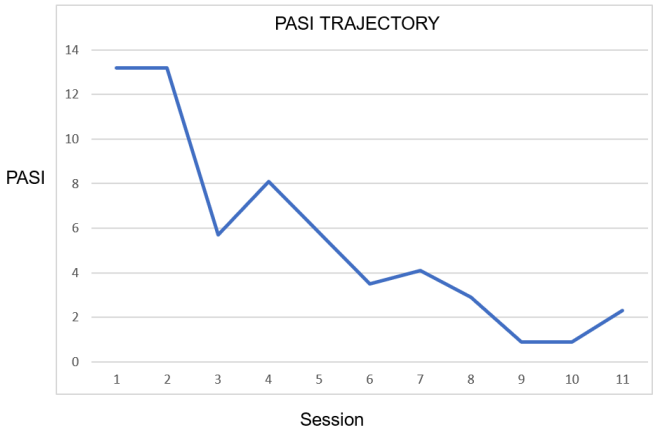


Figure 4 - Example PASI Trajectory

1.2 Aims and Objectives

The aim of my project is:

"Use machine learning methods to predict the outcome of UVB psoriasis treatment."

I have broken this aim down into objectives to guide the progress of the project and explained why they are needed.

1. Investigate and evaluate current research into predicting UVB treatment.

Explanation: Understanding the current research into the area is very important. Going into the project blind could mean I spend time working on problems that have already been solved, instead of making new advancements. As I am also new to time series analysis and factors affecting psoriasis, research may help me find good information and aid my education in the area.

2. Understand the best features for predicting treatment outcome through data analysis.

Explanation: The best machine learning model in the world cannot be better than the data that is fed into it. A key aspect of data processing is finding redundant data through analysis. If redundant data is not removed, then bad connections will be made in the data, leading to bad predictions.

3. Create a working prototype which can predict treatment outcomes.

Explanation: This is a very important step in the process. By being able to create a prototype, I can prove it is possible to predict the treatment outcome. For a prototype to be successful, the success rate must be greater than a guessing chance, across many samples.

4. Experiment with at least 3 different machine learning techniques to improve prediction.

Explanation: Once the prototype is successful the next logical step is to make improvements to it. I will know I have achieved this objective when I can make a model with a better success rate than the prototype. To be thorough I am going to try at least 3 different techniques to get the best sample.

5. Using a variety of metrics, compare the different techniques and find the best predictors.

Explanation: There are a wide variety of metrics which can be used to assess a model's performance, some of which are more useful in my circumstances than others. Knowing how to properly assess my predictors is a very important step in determining the success of the project.

1.3 Risks

I have foreseen a few risks that I may encounter in the project:

1. Lack of experience in the time series models

I am hoping to mitigate this risk through extensive research and preliminary work, but it still will not make me an expert. I will more than likely encounter issues which will require research and slow down development.

2. Small dataset

The next risk is with the data. Machine learning generally requires lots of data to be effective so a small dataset can be a very limiting factor. I have identified a few possible solutions to this in my preliminary research, but I will not know if they are effective until later.

3. Data has gaps

Another issue with the data, is the gaps. These have come from patients missing sessions or opting out of the study. With incomplete data it is very hard to build a complete picture of the treatment which will affect the outcome of the prediction. There are methods to combat this, but they come with drawbacks and payoffs.

Chapter 2 – Background

2.0 Introduction

The first idea of a modern-day artificial intelligence was written by Czech playwright Karl Capek in 1921 whose sentient mechanical men, designed for working on a factory line, revolted against their human masters. The idea of robots was popularised by famous sci-fi writer Isaac Asimov who wrote his famous 3 laws which made it impossible for a robot to be nothing but obedient. A common thread throughout most robot sci-fi is that these artificial beings are destined to only do us harm (Stanford University, 2022). As science has caught up to these ideas, most often the mechanical mind can be put to good. I hope to be able to apply this to the problem of psoriasis treatment.

2.1 Research

Data science is a very deep and complicated field. Jumping straight into development without research would mean I am facing concepts which I don't have the foundational knowledge to start to understand. The aim of this research section is to understand the techniques so I can properly implement them and understand more complex techniques to get the best results possible.

2.2 Project Plan

Having a plan is very important for an ongoing project like this. An effective plan will minimise the backtracking of work and make the most effective use of my time. I am going to break the project down into major milestone sections and break each section down further when necessary.

In my research (AlexSoft, 2018; Jordan, 2018), machine learning projects tend to follow the same general structure: analysis, training, and evaluation. My project milestones will be an enhanced version of that structure as they are currently too vague to properly gauge progress. My project milestones will be:

1. Background Research
2. Analysis Development
3. Analysis Results
4. Model Development
5. Model Evaluation
6. Overall Conclusion

These milestones will be a good way to track my progress. My background research will be structured in a way to resemble the milestones. This is to keep my background research on track and make it easy to refer to further in the project.

2.3 Background research

2.3.1 Analysis

"Everything in data science starts with data. Your machine learning model is just as good as the data fed into it" - (Yıldırım, 2021)

Data analysis is a cornerstone of data science. Without the insights from analysis, machine learning would have terrible performance. Having the knowledge to perform in-depth analysis will enable me to create effective prediction models later. This process of graphing and reviewing data is called **exploratory data analysis (EDA)**.

In machine learning, data is broken up into variables. These variables are information about each data point in my dataset. For example, age, BMI, and ethnicity would all be variables of a person. My dataset is broken down into 36 variables which describe the UVB treatment. A variable can be categorised as either a continuous variable or categorical variable. From my research (Patil, 2018; IBM, 2020), EDA is broken down into two aspects: univariate and bivariate/multivariate analysis. This will be explored further.

2.3.1.1 Univariate

Univariate analysis is where the scope of the analysis is limited to one variable. This is the simplest kind of analysis and involves finding the summary statistics. The summary statistics involves investigating the distribution of a variable and depends on the type of variable: continuous or categorical.

Continuous Summary Statistics

A continuous variable is one which can be measures on an infinite scale with an unlimited set of possible values. An example of a continuous variable is distance, which can be endlessly broken down. There are many ways to viewing the distribution of a continuous variable. The variables I believe give the best summary are mean, median, range, standard deviation, coefficient of variation, and IQR.

Categorical Summary Statistics

A categorical variable is one which can be broken into distinct groups or categories. For example, gender is categorical because there are finite number of groups. There is a subset of categorical variables called ordinal variables. Ordinal variables are categorical variables which have an order, e.g., TV ratings on a scale from 1-5. We implicitly know a 5 is either better or worse than a 1. The distribution of categorical variables can be best shown by simply using frequency distribution graphs, such as bar graph and pie charts.

2.3.1.2 Bi-variate analysis

Bi-variate analysis is the analysis of relationships between two variables. There are two types of bivariate analysis: continuous-continuous and categorical-categorical

Continuous-to- Continuous

Linear correlation

Linear correlation is defined as when the ratio of two given variables are same/constant, also called a linear relationship, which is best discovered using the Pearson Correlation Coefficient. This summarises the correlation between variables into a value between $-1 \leq x \leq 1$, with -1 being perfect negative correlation and 1 being perfect positive correlation.

A common thread in my research is the handling of these correlated values. When two variables are highly correlated, one of the values should be removed. Having 2+ correlated variables is not critical to the success of the model, but it does have a nice improvement to the data storage size and training speed. A paper found that with a large amount of inter variable correlation, there came a ~3% reduction in accuracy across a selection of models, except for decision trees (Lannge, 2021).

Distance Correlation

While Pearson correlation is very adept as discovering linear correlation, many variables have non-linear correlation. A nonlinear correlation is defined as one where the ratio of two given values is not constant. A comparison of linear/non-linear correlation is shown below in Figure 5.

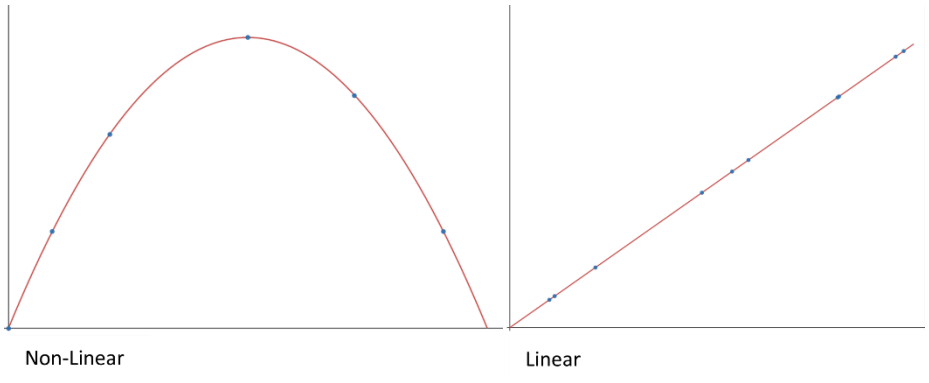


Figure 5 - Comparison of Linear and Non-Linear Relationship

The main methods for finding non-linear correlation are Spearman's rank correlation and distance correlation. I think distance correlation is superior for my situation as it produces a continuous value which is comparable to Pearson Correlation whereas the Spearman method ranks the variables by their correlation.

If two variables are neither distance nor Pearson correlated with the prediction variable, then they should be removed as they provide no information to help prediction and will only slow down model training.

Principle Component Analysis (PCA)

Principle Component Analysis is an adaptive data transformation technique which aims to help reduce the dimensionality of datasets while minimising information loss. This technique helps the issues which occur from having intervariable correlation and for finding new axes to apply clustering algorithms. This technique is widely used in machine learning.

Categorical-to-Categorical

Chi Square Test of Independence

The Chi-Square test is used to find the dependency of two categorial variables by using a statistical hypothesis test. This test compares the expected frequency distribution of the variables with the real frequency and calculates the chance the variables are dependent on each other. The dependency works similarly to the correlation of continuous variables. Therefore, if the test determines two variables are dependent past a critical value (α) then one should be removed.

2.3.2 Machine Learning Models

In my objectives I specified that at least three machine learning techniques/models should be used. All machine learning models can be classified as either: supervised, semi-supervised, unsupervised or reinforcement. In this situation, semi-supervised and reinforcement are not applicable, which leaves supervised and unsupervised models. Supervised can be further split into regression and classification models. I will choose my models from the classification, regression, and unsupervised types. When choosing models, my aim is to get a variety of techniques to compare which ones are most effective.

Classification

Classification models are ones which predict an outcome from a finite set (a categorical variable). There are a wide variety of classification techniques to choose from. When looking through medical research, clustering algorithms appear very regularly (Veloso *et al.*, 2014; Huang *et al.*, 2019; Ramachandran *et al.*, 2021) and seem effective. The most popular clustering approach to classification learning is the K-Nearest Neighbours model, which I have previous experience with. My dataset also contains a wide variety of variables which enables the Random Forest model to be effective. Therefore, I will use these two models.

Regression

Regression models are ones which predict a continuous variable. Regression learning is mainly dominated by deep learning models. Because of this I am going to use the Artificial Neural Network (ANN) and Recurrent Neural Network (RNN) deep learning models. RNN models were created for prediction of temporally structured data, such as the time series data in my dataset. I expect these to be effective predictors.

RNN models come in a few forms: one-to-one, one-to-many, many-to-one, and many-to-many. This describes the mapping of inputs to outputs in the model. Once I have analysed my data I will have to then decide on the relevant mapping.

Unsupervised

Unsupervised is when there are no labels to my data, the model tries to find patterns in the data. As mentioned earlier, clustering is a common technique used in predictive medicine. K-Means is a very popular unsupervised model which uses clustering to group the feature variables into *k* clusters.

2.3.3 Model evaluation

The final project objective is to evaluate the models I have created. Choosing performance metrics which give me the best evaluation of a model is very important. Each type of model (classification, regression or unsupervised) has its own metrics which behave very differently.

Classification

Classification model evaluation is the easiest form of model evaluation. The output of a classification model falls into a discrete set of possible outputs. This means if the output differs from the correct label of the data, we can firmly say it is the wrong answer. The simplest and most effective metric for this is accuracy, which is the percentage of times the answer was correct

On the surface accuracy is a one-size-fits-all metric, but it does have some flaws. These come about when there is an imbalance with the frequencies of each output classes in the set. This tends to happen when trying to predict something which very rarely happens. For example, predicting an asteroid hitting the earth. Since 99% of the time this will not happen there are many examples of the asteroid not hitting earth and very few when it does. So, if the model just predicted that the asteroid will never hit earth, the accuracy would be 99%, it would just fail in the small number of times an asteroid does hit, making the model useless. To fix this issue more metrics are required. The most common metrics which accompany accuracy are precision, recall and F1-score. These metrics fix the situation where accuracy falls short, but in most instances accuracy is enough.

Regression Evaluation

When dealing with a regression model, we get an output and label of a continuous number. Unlike classification, regression is dealing with an infinite output set and cannot rely on an answer being right or wrong. Regression deals with the error between labels and predicted values. This error can be interpreted in many ways. Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are the most common measurements. Both of these take the absolute error between a prediction and label. RMSE takes the mean of the squared error then the square root of that, whereas MAE just takes the average absolute error. The difference is subtle, but RMSE will get worse if the error is very large as it is squared, so it has high weight towards large errors. MAE just gives the average absolute error. The last metric I am going to use is the Mean Absolute Percentage Error (MAPE), which is MAE in percentage error form because it is easier to understand.

Unsupervised

Unsupervised learning is inherently a much hard type of model to evaluate as there is no “correct” value, therefore no comparisons can be made between the model output and a target value. The only way we can understand the performance of an unsupervised model is by quantifying the quality of the clusters. A good quality cluster is one which is very dense with points, which is best measured using the Within Cluster Set of Squares (WCSS) metric. This takes the average Euclidean distance between every point in the same cluster. If this value is low, then every cluster is dense and probably fitted well. This leaves the distance between clusters to be measured. This can be done with a silhouette score, which takes the inter-cluster distance and creates a value between $-1 \leq x \leq 1$. These metrics should adequately quantify the model performance.

(Vlachos, 2011; Pathak, 2020)

Chapter 3 – Analysis Development

3.0 Introduction

The purpose of this section is to document the development process of the data analytics. This section will not discuss the results of the analysis.

3.1 Development Strategy

Machine learning tends to work very well with the quick implementation/review system of agile development strategy. While data analysis is a very big aspect of the machine learning process it does not perform as well under this style of development. This is because it is much more difficult to evaluate the performance of data analytics. Machine learning models have very specific metrics which

state how well a model has performed whereas the success of data analytics comes from the insights which are generated by the analysis. Because of this I have chosen to use the waterfall model for development in this section (Figure 6).

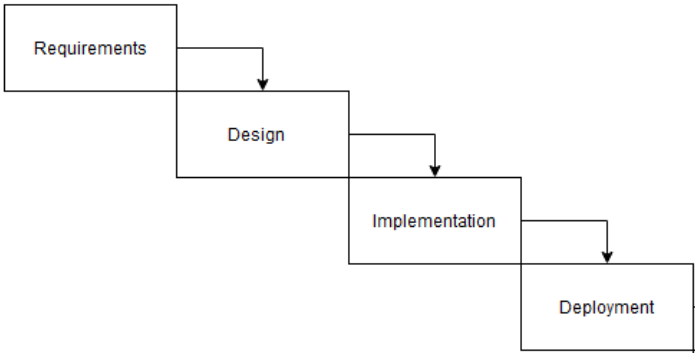


Figure 6 - Analysis Development Model

This each step in the model is justified and explained below.

3.1.1 Requirements and Design

The biggest factor in determining the performance of machine learning is the quality of the data going in and coming out. From my research I found data analytics is usually broken into univariate analysis and multivariate analysis. According to this, to properly perform the analysis, I must first find the summary statistics of each variable and then the correlation between each other, which will include comparing features variables to prediction variables.

3.1.2 Implementation

Language

To implement my statistical analysis, I had to choose between using Python (van Rossum and Drake, 2009) and R (R Core Team, 2021). Both languages are interpreted languages with strong libraries and built-in functions for doing statistical analysis and displaying graphics, so would both be suitable. R was developed in 1992 with a focus on in-depth statistical analysis while Python was originally developed to be a general-purpose software but has recently developed a strong statistical analysis foundation.

In the end I chose Python for my analysis development tool. This is because:

- **Experience**

I have quite a few years of experience in Python and have an in-depth knowledge of the libraries commonly used with statistical analysis. R is also reported to have a high learning curve and with the time scale of my dissertation, I did not have the time to learn the new language

- **Following**

R is a relatively new language and while it is very popular, it is nowhere near Python's popularity. This means there is less online support and documentation for R which will slow down my development.

- **Support Tools**

Python has an array of environments to develop in and present the information. Google Colabratory (Alphabet Inc, 2022), the tool I am going to use to develop, is specialised in sharing information and developing in a very intuitive way.

- **Speed**

Python is reportedly 5.8x faster than R (Korstanje, 2020). Machine learning uses huge amounts of data to find patterns and train, so data analysis requires fast computation speeds.

Structures

Dealing with data in analytics can be a complicated process which is made much simpler with the use of structures. For my analytics I used a mixture of the Pandas DataFrame (McKinney, 2010) and NumPy's Arrays (Harris *et al.*, 2020). Dataframes are specialised for handling multidimensional data in an easy-to-use format which can be presented in a clear way. They are structured like tables with named columns and indexed rows so can easily be accessed. They are also well supported by other libraries which can handle data in the DataFrame format.

NumPy arrays are the basis of all scientific research in Python because of their speed. They are very simple arrays which can easily support multidimensionality. They are the standard structure and therefore have the most support. Their only downfall is they are harder to understand and present. A combination of NumPy array and Pandas DataFrame covers all the needs for this project.

3.1.3 Deployment

Deployment in this case refers to how the data is presented. If improperly presented it can be much harder to understand the complex interactions and patterns might be missed. I will aim to use graphical display as much as possible, as it is much easier to understand than raw data.

3.2 Data descriptions

This is a description of all the variable in my dataset, what they describe and the possible outputs.

Variable Name	Description	Value set
PASI.END.WEEK.0	PASI value at beginning of treatment	0≤ PASI ≤72
PASI.END.WEEK.1	PASI value after 1 week of treatment	0≤ PASI ≤72
PASI.END.WEEK.2	PASI value after 2 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.3	PASI value after 3 weeks of treatment	0 ≤PASI ≤72
PASI.END.WEEK.4	PASI value after 4 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.5	PASI value after 5 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.6	PASI value after 6 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.7	PASI value after 7 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.8	PASI value after 8 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.9	PASI value after 9 weeks of treatment	0≤ PASI ≤72
PASI.END.WEEK.10	PASI value after 10 weeks of treatment	0≤ PASI ≤72
CLASS	Results from previous unsupervised learning experiment ran on the dataset	{1,2,3}
ENTRY_AGE	Age at the start of the study	18 ≤ age ≤ 84
MALIGNANT_SKIN_CANCER	If patient has malignant skin cancer	{0=No, 1=Yes}
FH_OF_PSORIASIS	If there is a family history of psoriasis	{0=No, 1=Yes}
SEASONAL_FLARE	Patients can be subject to quick flares of psoriasis in certain seasons	{N, Spring, Summer, Autumn, Winter}
SEASONAL_FLARE_2	Patients can be subject to quick flares of psoriasis in certain seasons	{N, Spring, Summer, Autumn, Winter}
SKIN_TYPE	Dermatologically different skin types	{1,2,3,4}
SMOKING	How much the patient smokes	{0=Never, 1=Currently, 2=Ex-smoker, 3=Occasionally}
BMI	Body Mass Index of patient	18 ≤ BMI ≤ 42
ALCOHOL	How much alcohol the patient consumes	{0=Never, 1=Occasionally, 2=Moderately, 3=Excessively}
SEX	Male/Female	{0,1}
CURRENT_TOPICALS	Other topical medicines being taken during study	{EMOLLIENTS, TAR, TOPICAL TREATMENT, N, TOPICAL STEROIDS, TAR, DITHRANOL}
JOINTS	Has the psoriasis affected patients' joints	{0=No, 1=Yes}
NAILS	Has the psoriasis affected patients' nails	{0=No, 1=Yes}
SCALP	Has the psoriasis affected patients' scalp	{0=No, 1=Yes}
ONSET_SPEED	How quickly the psoriasis developed	{0=No, 1=Yes}
BASELINE_DLQI	DLQI* at start of study	0 ≤ DLQI ≤ 30
BASELINE_CRP	C-reactive protein levels in blood at start of study	4 ≤ CRP ≤ 17
BASELINE_ZINC	Zinc levels in blood at start of study	13.1 ≤ ZINC ≤ 26.4
BASELINE_CALCIUM	Calcium levels in blood at start of study	2.2 < CALCUIM < 2.65
BASELINE_ADJ_CALCIUM	ADJ calcium levels in blood at start of study	2.22 ≤ ADJ CALCIUM ≤ 2.55
BASELINE_MAGNESIUM	Magnesium levels in blood at start of study	0.55 ≤ MAGNESIUM ≤ 0.95

BASELINE_ALBUMIN	Albumin levels in blood at start of study	39 ≤ ALBUMIN ≤ 52
BASELINE_SELENIUM	Selenium levels in blood at start of study	0.52 ≤ SELENIUM ≤ 1.41
BASELINE_VIT_D	Vitamin-D levels in blood at start of study	12 ≤ VIT D ≤ 152

* DLQI – Dermatology Life Quality Index. Self-administered questionnaire which asks ten questions about how affected the patients' life has been by their skin condition.

3.3 Missing Value Handling

3.3.1 Variables

Performing analysis over data with missing values is very difficult. When dealing with a small data set like mine, having missing values can be a big problem. Because of this I decided it would be better to remove any variable with a significant quantity of missing values.

Variable	Missing Values
BMI	5
BASELINE_DLQI	1
BASELINE_CRP	3
BASELINE_ZINC	**22**
BASELINE_CALCIUM	**18**
BASELINE_ADJ_CALCIUM	**21**
BASELINE_MAGNESIUM	**16**
BASELINE_ALBUMIN	**15**
BASELINE_SELENIUM	**16**
BASELINE_VIT_D	5

Figure 7 - Missing Values in Dataset Variables

The missing values for each variable can be seen above in Figure 7. To fill the missing values, I decide to use a statistical technique called imputation. This involves taking the mean of each column and replacing the missing values with that mean. The issue with doing this on a larger scale is it can disrupt the distribution of values in the variable and affect the prediction. To limit this, I decided to only impute on variables which have less than 10 missing values. If a variable has more than 10 missing values, it is removed.

Removed variables: BASELINE_ZINC, BASELINE_CALCIUM, BASELINE_ADJ_CALCIUM, BASELINE_MAGNESIUM, BASELINE_ALBUMIN, BASELINE_SELENIUM

3.3.2 PASI Values

Time series is the name given to data which has a regular temporal structure between the data points. This is very useful for discovering the underlying causes of trends and pattern across different time periods. This type of data is key to how I am going to predict treatment outcomes. The issue with this data is it requires a regular time difference between the points which is broken when values are missing. With the PASI trajectories there are two types of missing values: when patients have dropped out/finished treatment or when they have missed a week of sessions. If the patient has missed a week of sessions, there will be a value missing in the trajectory. To deal with this I chose to linearly interpolate to fill the missing values. Linear interpolation works well when the all the data follows a general direction and does not massively fluctuate. All the trajectories in my dataset tend towards 0 over time, so linearly interpolating will follow this direction. Just like data imputation, if too many values are artificially generated it can throw off the distribution of the variable. In my data there is only 7 trajectories with missing values, so interpolating them is not an issue. All time series' with missing values are shown below in Figures 8 and 9, before and after interpolation.

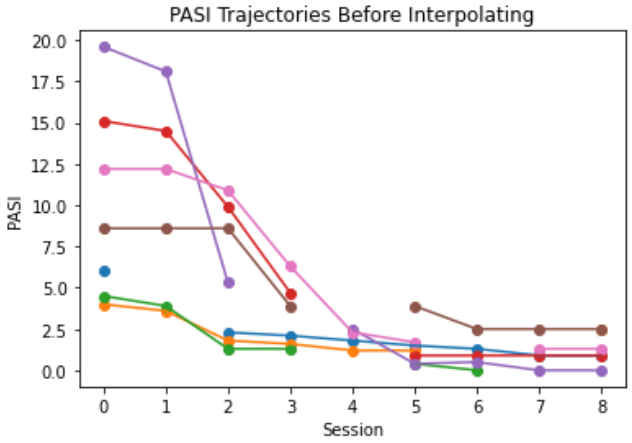


Figure 8 - Graph of PASI Value Before Interpolation

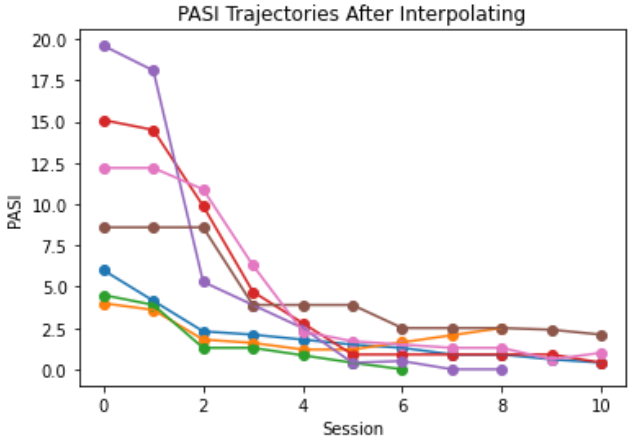


Figure 9 - Graph of PASI Values After Interpolation

3.4 Data Collection

Before proceeding with analysis, I found attributes about each PASI trajectory to use as prediction labels. The goal of my project is to discover how well a patient will respond to the treatment. Since people begin at different PASI levels I standardised the reduction of each patient by finding the percentage PASI reduction from start to end value. I am dealing with how effective the response is so absolute values are not as important.

$$Percentage\ PASI\ Difference = \frac{|StartPASI - EndPASI|}{StartPASI} * 100$$

Figure 10 - Equation for Percentage PASI Difference

This was a good summary variable but didn't take how quickly the patient responded to the treatment into account. To address this, I found the average percentage gradient of the trajectory by diving the percentage difference by the number of sessions in the study. This rate of change variable is much a much more descriptive label.

$$Average\ PASI\ Gradient = \frac{Percentage\ PASI\ difference}{n_{Sessions}}$$

Figure 11 - Equation for Average PASI Gradient

3.5 Univariate analysis

Univariate analysis is where the scope of the analysis is limited to one variable. This is the simplest kind of analysis and starts with finding the summary statistics. The summary statistics depends on the type of variable: continuous or categorical. A categorical variable is data whose set of values is finite. Continuous data is data that can be measured on an infinite scale and can be any value on that scale.

3.5.1 Continuous Summary Statistics

Summary statistics for continuous variables involves looking at the distribution of datapoints in the dataset and looking at the spread and skew. The metrics I am using for continuous analysis were established in my research.

Notation

ε = Full data set

x_v = All values of a variable

n_v = Total frequency of variable

Y_i^v = Index of element in sorted variable set

Name	Equation
Range	$Range_v = \left(x_v\right) - min(x_v)$
Mean	$\bar{x}_v = \sum(x_v)/n_v$
Median	If n_v is even: $Q_2 = \tilde{x}_v = \frac{1}{2}(Y^v_{\frac{n_v}{2}} + Y^v_{1+\frac{n_v}{2}})$ If n_v is odd: $Q_2 = \tilde{x}_v = Y^v_{\frac{n_v+1}{2}}$ (Weisstein, no date)
Standard Deviation	$\sigma_v = \sqrt{\frac{1}{n_v}\sum_{i=1}^{n_v}(x_i^v - \bar{x}_v)^2}$ (Weisstein, 2022)
Coefficient Of Variation	$CV = \frac{\sigma_v}{\bar{x}_v}$
Quartiles	1st Quartile: $Q_{v1} = Y^v_{\frac{1}{4}(n_v+1)}$ 3rd Quartile: $Qv_3 = Y^v_{\frac{3}{4}(n_v+1)}$
Inter-Quartile Range	$IQR_v = Q_{v3} - Q_{v1}$

Because of the structures I am using, finding these summary statistics is as simple as calling the relevant function. The harder part was graphically displaying the statistics. Only coefficient of variance can be directly compared since the scale between variables has been standardised. I found that a bar plot was the best way to easily compare the size of coefficient of variance. For the other variables, I found that a box + whisker plot was the best way to make the comparison.

3.5.2 Categorical Summary Statistics

Binning

Binning is the process of converting a continuous variable into a categorical variable. Binning is required for classification models, which require a categorical variable to predict. It also adds more information to analyse for correlations. The aim of binning is to get the biggest average difference between bins while keeping the frequency in each bin as uniform as possible. There are three main ways of doing this: uniform, quantile, and K-Means.

Uniform – Keep bin widths a uniform size

Quantile –Keep the frequencies across all bins the same

K-Means – Use the K-Means algorithm to group the data into clusters and form bins around them.

To decide on the best method, I plotted the average frequency difference against mean value difference for each bin (Figure 12).

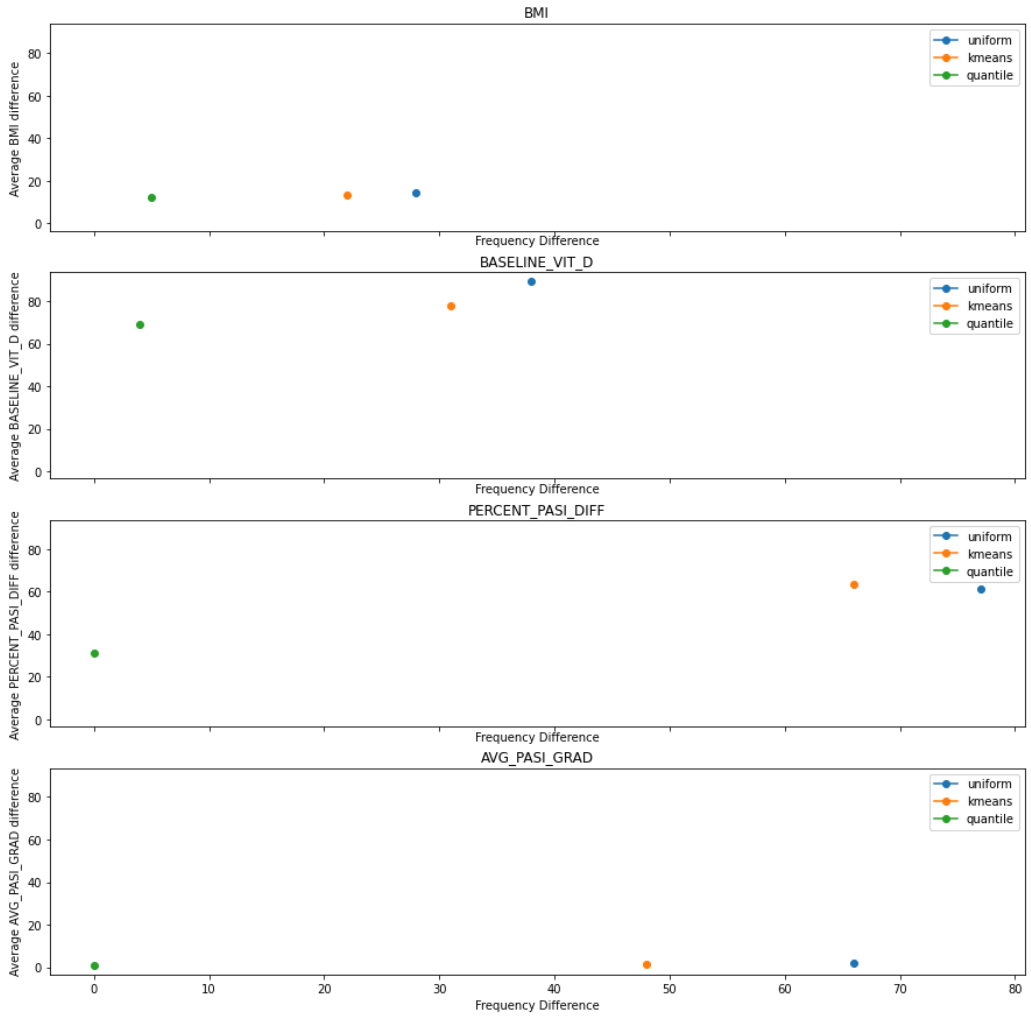


Figure 12 - Graphs of Frequency Differences vs Mean Bin Value differences in Many Categorical Variables For Different Binning Techniques

The aim for binning is a high average difference with a low frequency difference, so the perfect combination would be in the top left of the graph and worst is bottom right of the graph. While K-Means and uniform did give very good mean differences between the categories, they consistently had very high frequency differences between the categories. When dealing with a small dataset I think having very low samples in some of the categories will affect the model prediction more than a difference in the mean. Since, by definition, quantile has the same frequency distribution this won't be

a problem and by looking at the graph there is a good enough mean difference between the categories to continue with it.

Summary Statistics

The summary statistics for categorical variables is the frequency distribution of the categories. The simplest way of plotting this is the bar chart because it is very easy to compare the relative heights of the bars to show the frequency differences.

3.6 Bi-Variate analysis

The bi-variate analysis can be broken down into two categories: categorical-to-categorical and continuous-to-continuous. With continuous-to-continuous variables, each variable is compared to every other variable and the correlation is found between them. To find this correlation it must be done in two ways: Pearson's and distance correlation, to test for linear and nonlinear correlation. These values are best presented in a heatmap, which shows larger values as brighter colours. For the colour scheme I find that blue colour gradients show differences better.

As I found in my research, if two feature variables have a high correlation (>0.7) one of the variables should be removed. This is because the information from one of the variables is enough to find the patterns while the other is essentially redundant. Another conclusion we can draw is if the correlation between a feature and predicting variable is very low (<0.1) it should be removed as there is no pattern to find. This should only be done if the correlation is low in both distance and Pearson's correlation.

For categorical-to-categorical, chi squared test of independence is used. This works similarly to the continuous correlation test. In this case, since we are searching for two-way independence, we are performing a two-tailed statistical test. This means our critical value for dependency is $\alpha=0.025$. If the p value from the test is below this value, then the variables are dependent on each other and, similarly to continuous correlation, one should be removed.

Chapter 4 - Analysis Results

4.0 Introduction

In this section I will be showcasing the results of my analysis and describing the influence the results had on my methodology. The aim of the analytics is to find the characteristics and intercorrelation features of the data. Then use this information to prepare the data and construct suitable machine learning models. The aim of the analysis is to generate a **curated dataset** which will be optimised for prediction.

4.1 Univariate analysis

4.1.1 Continuous Variables

In this analysis, I will be looking at the continuous feature columns. PASI variables are not included as they have a multivariable structure which must be considered

Column	IQR	Range	Standard Deviation	Coefficient of Variance	Mean	Median
ENTRY_AGE	24.5	66.0	16.739	0.373	44.917	43.5
BMI	7.0	24.0	5.418	0.195	27.742	27.742
BASELINE_DLQI	9.0	27.0	6.492	0.579	11.211	10.0
BASELINE_CRP	1.0	13.0	2.497	0.487	5.129	4.0
BASELINE_VIT_D	47.0	140.0	31.195	0.509	61.242	59.0

Percentage PASI difference	16.419	85.714	16.53	0.198	83.424	87.426
Average PASI Gradient	0.599	2.987	0.512	0.603	0.849	0.738

Figure 13 - Univariate Continuous Variable Analysis Results

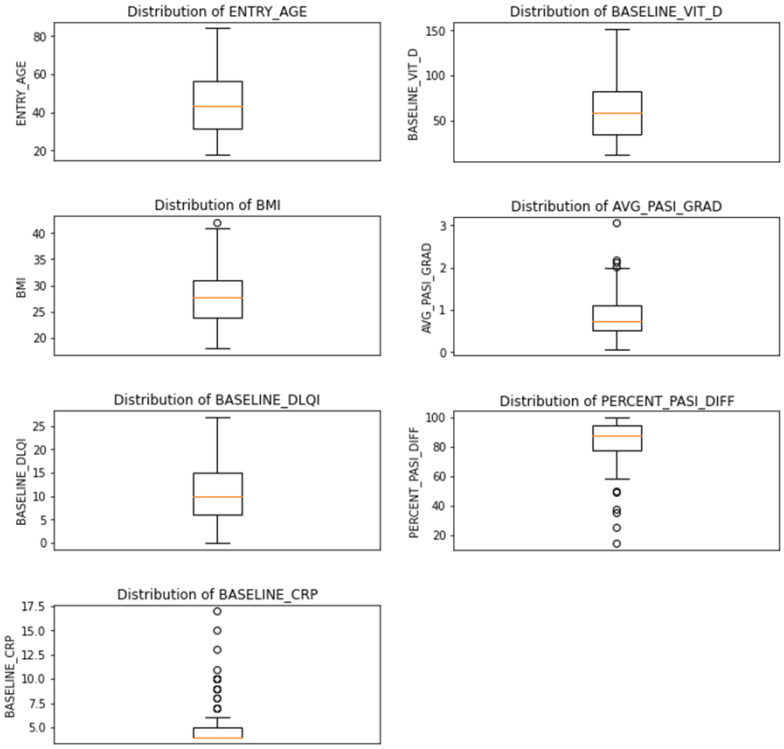


Figure 14 - Box + Whisker Plots of Univariate Continuous Variable Analysis Results

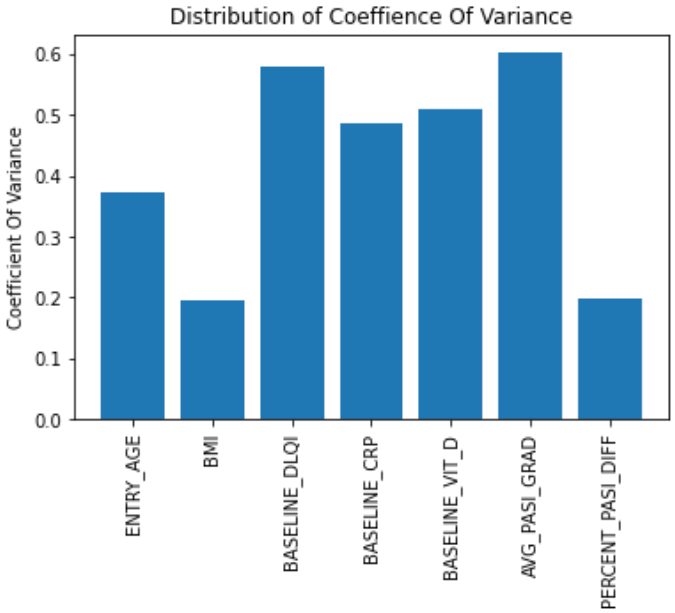


Figure 15 - Bar Chart Comparison of Coefficient of Variance

Coefficient of Variance (CoV) is a good metric for comparing the distribution of values with different scales. The distribution of CoV is shown in Figure 15. A larger CoV is especially important with

predictor variables (Percentage PASI difference, Average PASI gradient). Average PASI gradient has a larger variance which makes it a better predictor than Percentage PASI difference, which is much lower. Therefore, I am going to remove Percentage PASI difference from my curated dataset.

Another notable statistic is with mean and median, BMI has the same mean and median. This means that the BMI data has no skew. This is not representative of the UK population, where the dataset is taken from. Research has shown that BMI in the UK has a right skew, which does not show up in the data (Tsang *et al.*, 2018). The skew is shown in Figure 16.

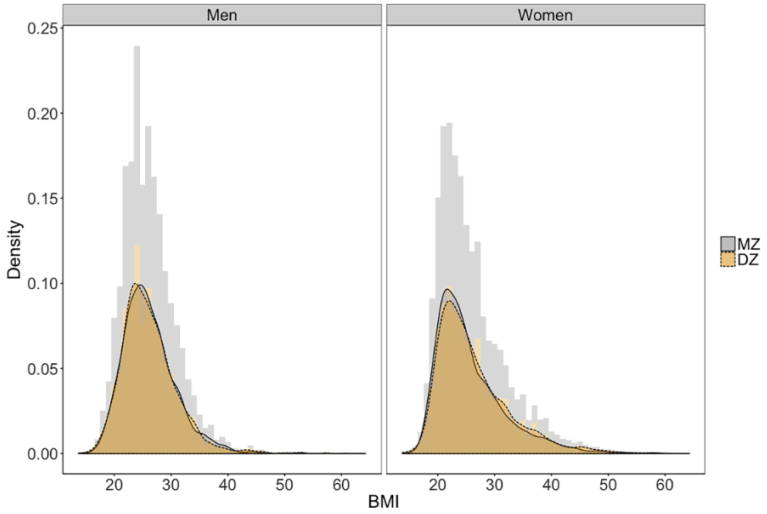


Figure 16 - Graph of BMI Distribution (Tsang et al., 2018)

The comparison of range and IQR can tell us lots about how the variable is distributed (shown in Figure 13). IQR shows us the central 50% of the data range. By extrapolating out IQR to 100% (x2) and then comparing it with the range we can see how affected the distribution is by outliers. This is mostly a problem with BASELINE_CRP as there is a 146% difference between the extrapolated IQR and range which shows that CRP is highly affected by the outliers. As Figure 14 shows this is true. Outliers can mislead a model which worsens prediction. Outliers account for 16 data points in the BASELINE_CRP variable. There is not enough data to say if this is representative of the true BASELINE_CRP distribution. With outliers removed the CoV is 0.10455. These factors make me unsure about the validity of the BASELINE_CRP variable so I am going to remove it for my curated dataset.

4.1.2 Categorical Variables

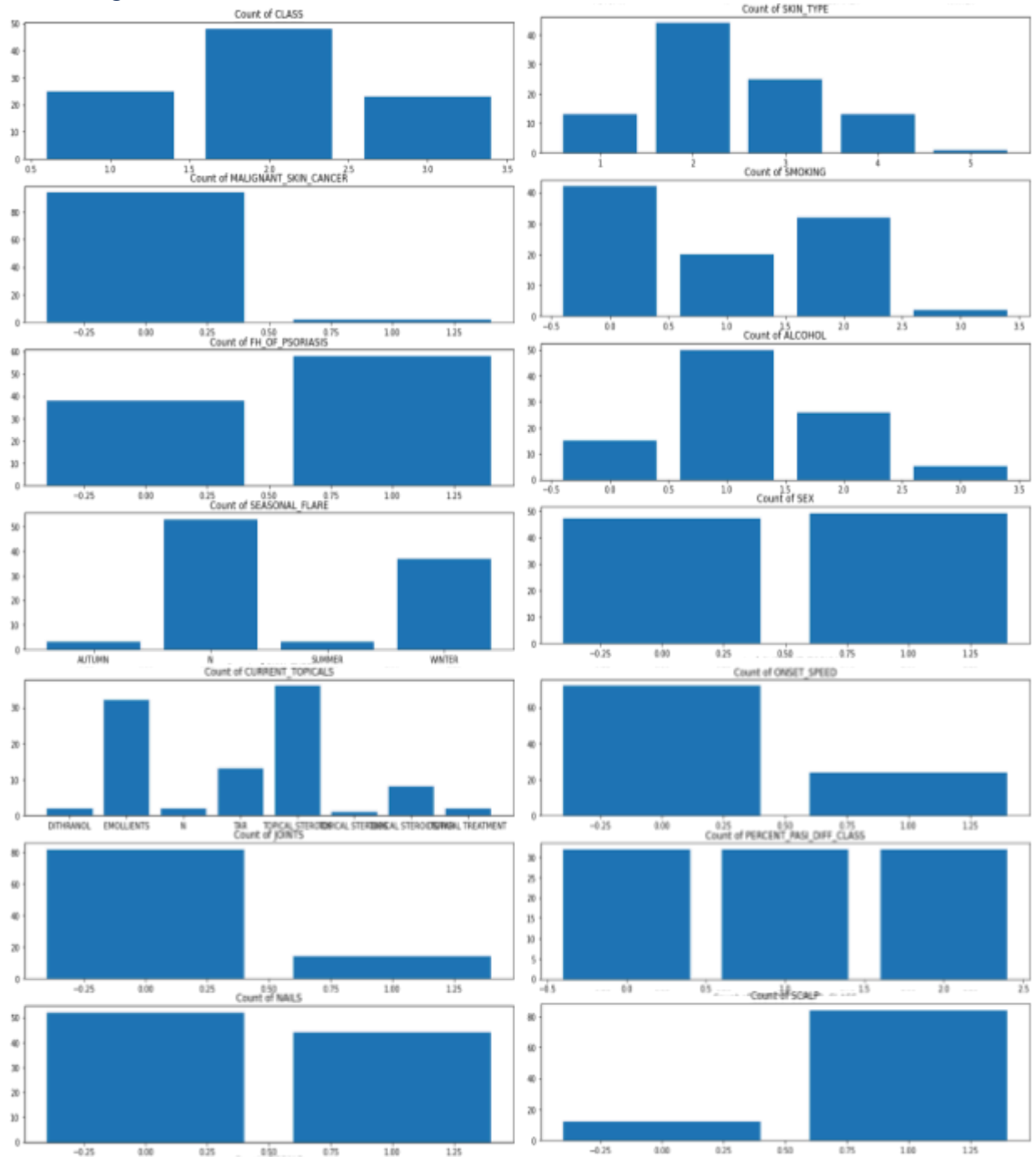


Figure 17 - Bar Graphs of Frequency Distribution for all Categorical Variables

One of the standout variables is malignant skin cancer patients where there is a significant difference in distributions. Because of the lack of positive subjects this is a candidate for removal. Another significant variable is the “current topicals”. Because of the wide spread of the different topicals there is a few categories which have very few samples. One method to reduce the effect of this is to keep the most frequent categories (Emollients, Topical Steroids) and then collect the other categories into an “Other” category.

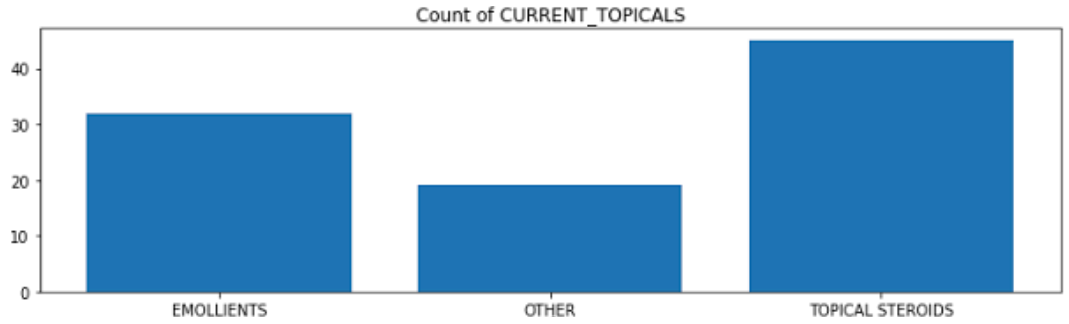


Figure 18 - Bar Graph of Modified CURRENT_TOPICALS Distribution

Figure 18 shows the results. This has a much more balanced distribution.

4.2 Bi-Variate Analysis

4.2.1 Continuous-to-Continuous

Pearson's Correlation

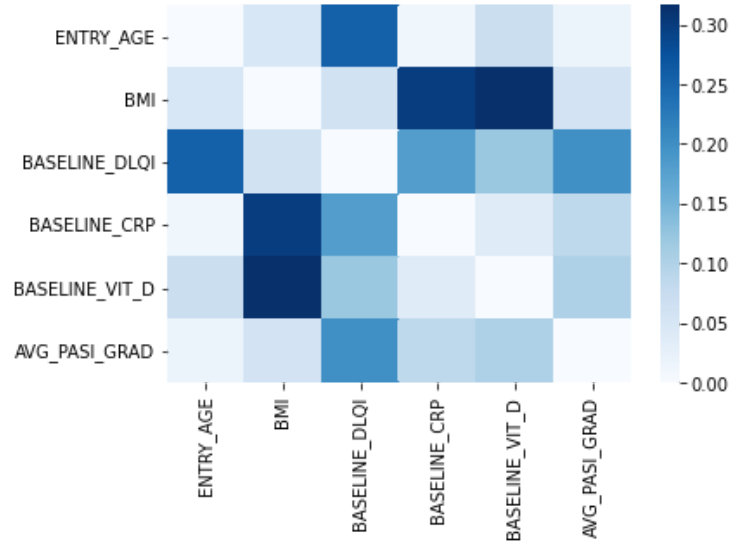


Figure 19 - Heatmap of Pearson's Correlation

Variable	**Pearson's Correlation To Average PASI Gradient**
BASELINE_DLQI	0.199218
BASELINE_VIT_D	0.101590
BASELINE_CRP	0.086951
BMI	0.059075
ENTRY_AGE	0.018351

Figure 20 - Table of Pearson's Correlation to Average PASI Gradient

Variables	**Pearson's Inter-correlation**
BASELINE_VIT_D, BMI	0.31622399053094363
BASELINE_CRP, BMI	0.29998909520694267
BASELINE_DLQI, ENTRY_AGE	0.2560594376301109
AVG_PASI_GRAD, BASELINE_DLQI	0.19921778490218908
BASELINE_CRP, BASELINE_DLQI	0.1818350491092097

Figure 21 - Table of Top 5 Pearson's Inter-Correlation

The heatmap in Figure 19 shows the Pearson correlation between continuous values. Correlations between the same variables are always 1 so they have been set to 0. This is because the 1 value was overshadowing the other values making the heatmap much harder to read.

As mentioned in the research section, having highly correlated variables (>0.7) is detrimental to machine learning in some cases. Also having very low correlation between a variable and Average PASI gradient is detrimental (<0.10). This metric makes it easy to show which metrics contribute no information towards the prediction and therefore can be removed.

From Figure 20 we can see the correlation with average PASI gradient. The variables with very low correlation (<0.1) are BASELINE_CRP, BMI, ENTRY_AGE. Pearson's correlation does have one big flaw, it only searches for linear correlation. This means we cannot remove variables just based on the Pearson correlation as they could have nonlinear correlations.

Figure 21 shows the top values of intercorrelation between variables. There are no variables with a significantly high intercorrelation to warrant their removal.

Distance Correlation

Distance correlation fixes the linear-only problem with Pearson's correlation by searching for nonlinear correlation. If a variable has no significant distance or Pearson correlation with the Average PASI Gradient or if a variable has a very significant distance correlation with another variable, it can be removed. I also have taken -1 away from each distance correlation value to move the domain from [0 to 2] to [-1 to 1] so it is in line with Pearson's correlation.

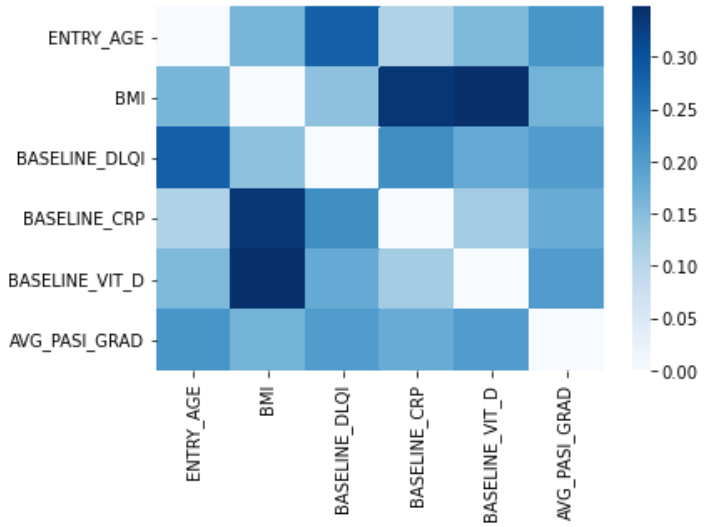


Figure 22 - Heatmap of Distance Correlation

Variable	**Distance Correlation to Average PASI Gradient**
ENTRY_AGE	0.211438
BASELINE_DLQI	0.202059
BASELINE_VIT_D	0.201413
BASELINE_CRP	0.177260
BMI	0.167089

Figure 23 - Table of Distance Correlation to Average PASI Gradient

Variables	**Distance Intercorrelation**
BASELINE_VIT_D, BMI	0.34792193062684507
BASELINE_CRP, BMI	0.3407589583808258
BASELINE_DLQI, ENTRY_AGE	0.28274696633620244
BASELINE_CRP, BASELINE_DLQI	0.22031292036828415

AVG_PASI_GRAD, ENTRY_AGE	0.2114383532832685

Figure 24 - Table of Distance Inter-Correlation

The results from distance correlation analysis are shown in Figure 22, 23, 24. From this information we can see there is much more nonlinear correlation occurring in the data. The values with low linear correlation (BMI, ENTRY_AGE, BASELINE_CRP) all have higher nonlinear correlation with Average PASI Gradient, so should not be removed. Especially with ENTRY_AGE as this had ~0.01 linear correlation but a 0.211 nonlinear correlation.

Figure 24 shows the top values of distance intercorrelation which shows there is not a significantly high intercorrelation value, so nothing is removed off the base of that either.

Notable Statistics

One of the useful correlations I found in the dataset was the relationship between the average PASI gradient and the starting PASI value. As Figure 25 shows there is almost a perfect positive correlation between these values. This relationship even extrapolates to the outlier value. This clearly shows that phototherapy is more effective the higher the starting PASI value. Interestingly, this correlation does not appear in the percentage PASI difference points.

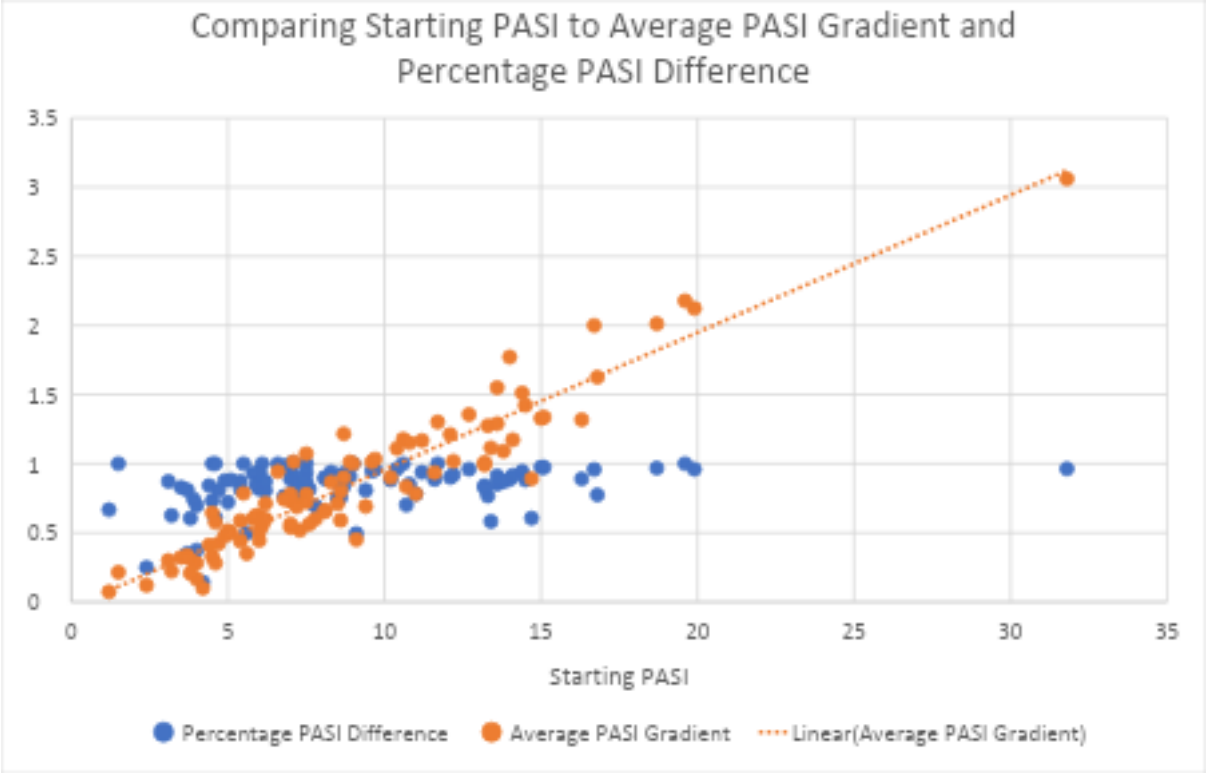


Figure 25 - Graph of Average PASI Gradient and Percentage PASI Difference vs Starting PASI

4.2.2 Categorical-to-Categorical

Chi-Square Test of Independence

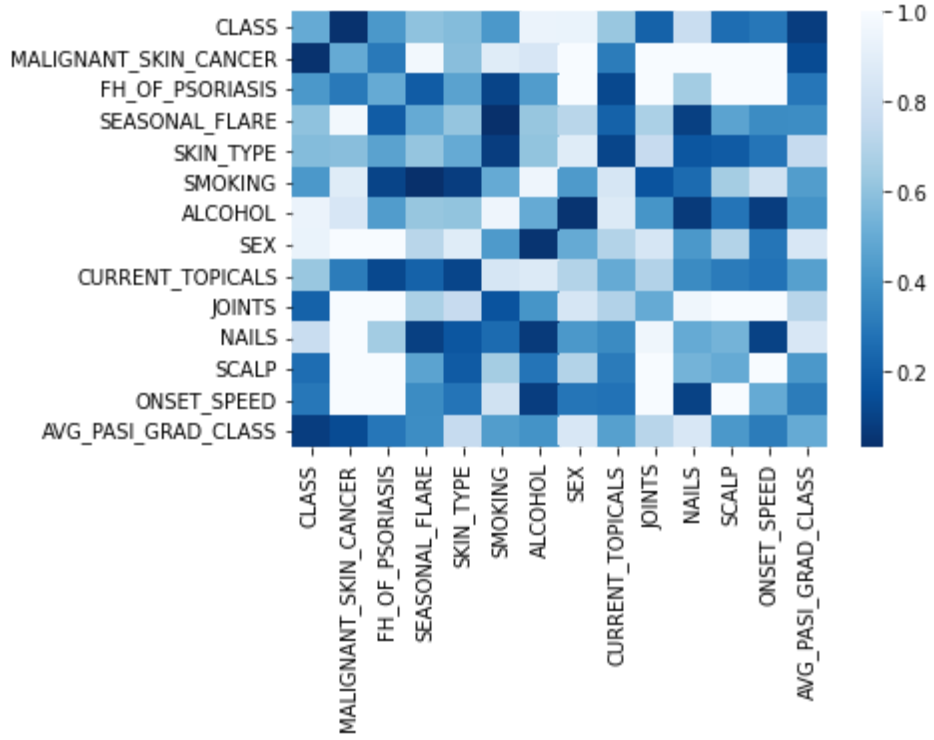


Figure 26 - Heatmap of Chi Square Test of Independence P-Values

The Chi-Square test of independence tests whether two categorical variables are related by giving us a p-value, which tells us the chance that the variables are not related. If this p-value is below a critical value (α), the variables are related. Since this is a two tailed test the critical value is $\alpha<0.025$. The heatmap in Figure 26 shows the comparison of all p-values between variables. The p-value between the same variable is always 0 so that has been changed to 0.5 to not overshadow the other data. The darkest colours signify the higher chance the variables are dependent on each other.

Variables	**P-Values**
SMOKING, SEASONAL_FLARE	0.031052
MALIGNANT_SKIN_CANCER, CLASS	0.039107
SEX, ALCOHOL	0.046764

Figure 27 - Table of Variables with Chi Square Test of Independence P-Values < 0.05

Figure 27 shows the lowest p-values. There are no values which pass our critical value, so no variables are sufficiently dependant on each other.

4.3 Evaluation

From the analysis I have discovered a lot about the variable distribution and how the variables correlate. Using this information, I have made my curated dataset which should improve the model performance. I understand that my analysis could be incorrect, so I will also be testing the models using the **full set of data** and comparing the results.

Changes to dataset in curated list:

- Removed BASELINE_CRP, MALIGNANT_SKIN_CANCER
- Modified CURRENT_TOPICALS

Chapter 5 – Model Development

5.0 Introduction

5.1 Development Strategy

Machine learning projects tend to perform very well using the agile development strategy. This is because once a model has been prototyped, it needs repeated testing, research, and implementation to improve it. Agile methodology also helps prevent one of the other inherent issues with machine learning, reproducibility. Since machine learning models are partially based on chance, they can sometimes have reliability problems. Because agile methodology involves very rapid minor changes, there is lots of testing, so reproducibility problems can be spotted early. This is opposed to other models, like Waterfall, which only tests when a model is complete, so it is unlikely to spot these issues. The proposed model is shown in Figure 28.

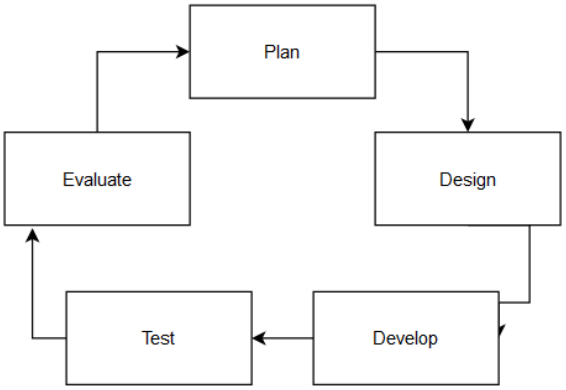


Figure 28 - Model Development Plan

5.1.1 Plan

To thoroughly test a variety of approaches to the problem I am going to implement models using classification, regression, and unsupervised learning techniques. I will choose models with very different underlying techniques, if possible, to get a wide range of principles.

5.1.2 Design

For every model I will be training a "naïve" model which will be an untrained, randomised model. This is to get a baseline for the performance of the trained model to show the improvements from a "guessing" model. I will also be comparing the results from training with the curated dataset vs the full dataset. Therefore, for each model a curated, full, and naïve model will be made.

Classification

The classification models I am going to use are the Random Forest and KNN models. These are very different techniques which can be very effective at classification.

Regression

The regression models I have chosen are ANN "multistep" model, RNN "multistep" model and ANN "session focused" model. While two of the models are using the ANN approach, they are using it in very different ways, and I think ANN models have lots of potential so they should be explored fully. I am also using an RNN model which will behave very differently from the ANN.

Unsupervised

The only unsupervised model I found which could be applied to this forecasting style of prediction was the K-Means model. I am also going to apply the analytics/dimensionality reduction technique of PCA to compare the improvements it gives.

5.1.3 Evaluation

Evaluation will dictate the metrics I am going to use to measure the performance for each model type.

Classification

The metrics I am going to use are Accuracy, Precision, Recall and F1-score. These metrics complement each other and provide a wide range of insights.

Regression

The metrics for regression come from the error of the predicted vs label values. The different metrics are forms of this error, so choosing metrics which are easy to interpret is more important in this case. I chose Root Mean Square Error, Mean Absolute Error, and Mean Absolute Percentage Error. These metrics are easy to understand and compare.

Unsupervised

By its nature, unsupervised learning is hard to evaluate since there are no labels to compare results to. Instead of measuring the labels vs predicted values we measure the quality of the clusters. If the model works well then it should create well-spaced, tight clusters. For this, I have chosen Within Cluster Set of Squares and the Silhouette Score, as discussed in the research.

5.2 Pre-Processing Principles

5.2.1 Feature Scaling

When dealing with some models, especially ANN's or any model involving Euclidean distance, it is very important to have a standardised range in all variables. This is to avoid some variables having more effect than another. For example, if a model was dealing with age and salary as inputs, with ranges of 10<age<50 and 10,000<salary<50,000. In this example, each value of age is 10x less valuable than salary. If the data was scaled to 0<x<1, then each variable is of equivalent value to all other variables.

Another problem created by non-standardised ranges is exploding values. Variables are frequently multiplied together, so with variables in -1<x<1 they will tend towards 0, but with values x<-1 or x>1 the values tend towards infinity, which leads to much more computation and storage.

Example

$$(1.01)^{1000} = 20959.16\ (x\infty)$$

$$(0.99)^{1000} = 0.000043\ (x0)$$

In some models, like Random Forest, this is not important as the variables are partitioned not directly compared to each other. Therefore, relative scales are not important.

The feature scaling method I am using is min-max scaling. I chose this because of its simplicity and doesn't require a normal distribution, like standardisation. The equation for min-max scaling is in Figure 29.

$$minmax(x) = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

Figure 29 - Min-Max Normalisation Equation

5.2.2 One Hot Encoding

The minmax normalisation technique works very well for continuous data but falls short with categorical data. As mentioned before, model inputs cannot be greater than 1 and simply don't work for string inputs. This is where one hot encoding can be used.

The first step of one hot encoding is mapping all unique values to an integer value.

e.g. {blue, red, blue, green, red} □ {blue=1, red=2, green=3}

Then converting this integer mapping into a binary mapping in a sparse matrix. These are also called dummy variables.

e.g. {1, 2, 1, 3, 2}

Integer	**1**	**2**	**3**
1	1	0	0
2	0	1	0
1	1	0	0
3	0	0	1
2	0	1	0

Figure 30 - Example Sparse Matrix for One Hot Encoding

Figure 30 shows that an input of 2 would convert into: 010

Since there are now no values greater than 1, this can be understood by a model. In some models, like random forest, only the integer mapping is necessary since values greater than 1 do not affect the performance.

5.2.3 Training, Testing, Validation Splitting

When training a model, we do not just put all our data in at once, otherwise we would have nothing to use to check the model's performance. We could test the model on some of the data it has already seen but this would bias the results compared to unseen data. To solve this, we split the data into training data, testing data and validation data. Validation data is data we use while the model is being trained but the model does not train on this. It is a little sample of testing data which we can use when training to gauge how under/overfit the data is. When splitting the data there is a balance. We want as much training data as possible, enough testing data to get a good sample and accurate results, and we want enough validation data to get a picture of the performance. Since this is a relatively small dataset, I am going to use 80% training, 15% testing and 5% validation, which I believe achieves the best balance.

5.2.4 K-Fold Cross Validation

This extends the training/test/validation splitting by using every bit of data for training and testing. We split the groups into k groups. We then take one of the groups and reserve it for testing. The other groups are used for training/validation. Once trained and tested, we save the evaluation metrics and discard the model. Then the next group is used for testing… This is done for all groups and the average of all the metrics is used. By doing this we use all the data in the set for training/testing and get a picture of the model's performance on all the data, instead of having to reserve some for testing. This is especially useful in a smaller dataset, where all the data is needed. When breaking up into groups, the ratio of sizes of training/testing groups is: $\frac{k-1}{k}$: $\frac{1}{k}$ (respectively) where k is the number of k-fold groups. Since we are aiming for ~15% testing data, the k value which achieves this split is 6. The calculation for this is shown in Figure 31.

Rounded down:

Figure 31 - Calculation of k Value

5.2.5 Hyperparameter Tuning

In machine learning we have two types of parameters. The first type is the variables which are passed into a model and used for prediction. The other type is the parameters we use when we setup a model, called hyperparameters. An example of these hyperparameters is the number of hidden layers to use in an ANN, or how many k-neighbours to search in KNN. By tuning these hyperparameters, we can

adjust the performance of the model. One of the methods I will be using, especially with a small set of hyperparameters, is the Grid Search Cross Validation. This exhaustively tests every combination of the hyperparameters and finds the best combination of them. This method does not work for ANN's where the number of layers/nodes is virtually unlimited, so searching is impossible. For these I will be starting with a small simple network then adding complexity until the performance caps, in a trial-and-error method.

5.3 Models

5.3.1 Classification

K-Nearest Neighbours

The general process for creating a model is very similar for all machine learning models:

1. Format the input/output data
2. Create model
3. Train model
4. Evaluate

Data

The input data will be the first 3 weeks of PASI values and, depending on the curated or full dataset, other feature variables. I believe the first 3 weeks to be enough to make predictions. Adding more weeks would mean the patient has to go through more weeks of treatment before they get a prediction. Any less would make an uncertain prediction.

K-Nearest Neighbours makes predictions by plotting the new point and finding the mode class of the k nearest points. The nearest points are found using the Euclidean distance between them, which means scale will influence the quality of clustering. Normalisation of continuous variables will fix this. Since the data is being scaled, we will also need to one-hot-encode the categorical variables.

For the data type I will be converting from the Pandas DataFrame into the NumPy array which is optimised for the learning that will be happening in the machine learning models. This will be the same for all models I will be using.

Model

I will be using the library SciKit Learn (Pedregosa *et al.*, 2011). This library has a KNN function which is very easy to use, simply enter the training data, labels, and the k number of neighbours.

Training

Unlike most models, KNN models are deterministic. This means if a model is given the same data twice it will have the same output, there is no randomness. Because of this we do not need to perform the training multiple times, with the same data, to mitigate any outlier results from the randomness.

I will also be performing K-Fold while training the model. By doing this I will be getting many random samples of the dataset. With each sample of the dataset, I will train a model over k many values to find the best value. I will also be doing many iterations of the K-Fold training to get an even bigger sample size. Many iterations are required because random samples of the dataset are taken so the data is changing. The training is shown as pseudocode in Figure 32.

```
n_iterations;
k_size
n_neighbours;

# Multiple Iterations of KFOLD
FOR iteration IN n_iterations:
    # KFOLD iteration
    FOR k IN k_size:
        # Neighbour size iteration
            FOR n IN n_neighbours
                model = KNN(n) # KNN with n neighbours
                model.train(k.train_data) # Train with that kfold iteration of data
                evaluate(model, k.test_data) # classification evaluation
            ENDFOR
        ENDFOR
    ENDFOR
```

Figure 32 - Pseudocode for KNN Training

Once an optimal *k* value is established, I will use it to make a best performance model and naïve model. The naïve will use randomised input/output data to generate a random model. I will then run a full classification evaluation on both models for comparison. This process will be repeated for both the curated and full datasets, as specified in the design.

Evaluation

During training I believe the best metrics to use are accuracy and F1-Score, because this measures most important aspects of classification learning. These will be collected every time a model is trained, so I will have many samples and will be resistant to random values. At the end of training, I will have an average accuracy and F1-Score for each *k* neighbour over many iterations.

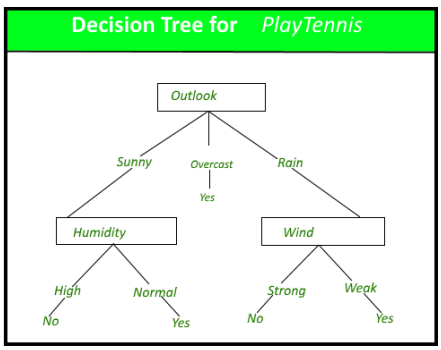
Using this information an optimal *k* neighbours can be found by using the best value for accuracy and F1-Score. This optimal *k* neighbours can then be used to train a full performance model and a full classification report can be made.

Random Forest

Data

The input data will be the first 3 weeks of PASI values and, depending on the curated or full dataset, other feature variables.

The random forest behaves very differently to other models regarding the input data. A random forest is a cluster of decision trees, which instead of comparing data variables, breaks up a variable into decisions. This means the relative scales between variables is not important and therefore no normalisation needs to take place. An example decision tree is shown in Figure 33.



33 - Example Decision Tree (GeeksForGeeks, 2021)

One hot encoding has an interesting effect on random forests. When data is one hot encoded it creates a sparse matrix, which is mostly zeros. Decision trees create their decisions by breaking a variable up and finding which combination results in the best information gain. When a sparse variable from the sparse matrix is broken up, since it is mostly zeros, there is much less information gain than if it was the full variable. Random forests also randomly sample the variables, so spreading out the information across many variables means the variables are sampled less, which lowers learning rate and negates performance. Therefore, data will not be one hot encoded.

Model

I am using the random forest model from SciKit Learn, which simply takes the input/output data and constructs a random forest. The construction of the forest can be controlled by a couple of hyperparameters which I will utilise in the training section.

Training

Hyperparameters have a substantial effect on the performance on the random forest model, so I will be using grid search cross validation to test a few combinations of hyperparameters. While grid search is very effective, the number of tests grows exponentially with the number of hyperparameters tested. Therefore, I will try and keep the number of tests low, otherwise it will be very time inefficient. The parameters I will be testing are:

- n_estimators: Number of trees in forest
- max_depth: Max depth of nodes in each tree

I believe these parameters will have the biggest effect and should yield the best results. Once I have got the best hyperparameters, I will proceed with them.

Compared to other models, random forest is small and can be very influenced on one iteration by randomness. Therefore, to get an accurate result I will need to run many iterations of training. To do this I will use a similar method to KNN training, by doing many iterations of K-Fold. The pseudocode of the training is found below in Figure 34.

```
1. n_iterations;
2. k_size
3.
4. # Multiple Iterations of KFOLD
5. FOR iteration IN n_iterations:
6.     # KFOLD iteration
7.     FOR k IN k_size:
8.         model = RandomForest(Best_Parameters)
9.         model.train(k.train_data) # Train with that kfold iteration of data
10.              evaluate(model, k.test_data) # classification evaluation
11.          ENDFOR
12.      ENDFOR
```

Figure 34 - Pseudocode for Random Forest Training

To train the naïve model I will generate random inputs/output and train the model. I will repeat this many times, to again, reduce randomness.

Evaluation

In each training cycle I will collect the classification metrics (Accuracy, Precision, Recall and F1-score). Since my goal is to find the average of each metric over many models, the best way to present this is a moving average. In each training iteration I will recalculate the mean with the new value, then plot this value. This will plot the moving mean over all iterations, showing it tend towards the true average for the model. The last value is the best prediction of average but using the graph I

can see if there has recently been a big change in the average. This allows me to see how stable/reliable the prediction is. I will repeat this for the naïve, full, and curated model.

5.3.2 Regression

Session Focused Model

Data

The input data will be the first 3 weeks of PASI values and, depending on the curated or full dataset, other feature variables. It will also receive a session number and the corresponding PASI value for that session. This is illustrated in Figure 35.

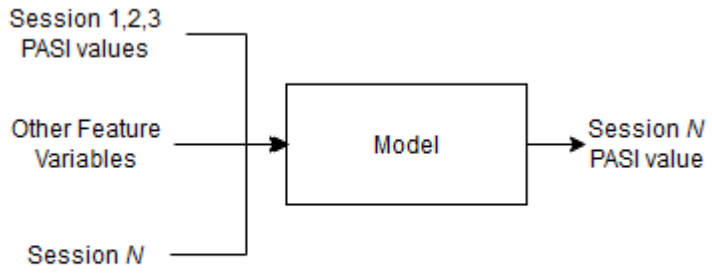


Figure 35 - Graphic Showing Inputs/Outputs of Session Focused Model

An example of this being deployed can be seen below in Figure 36

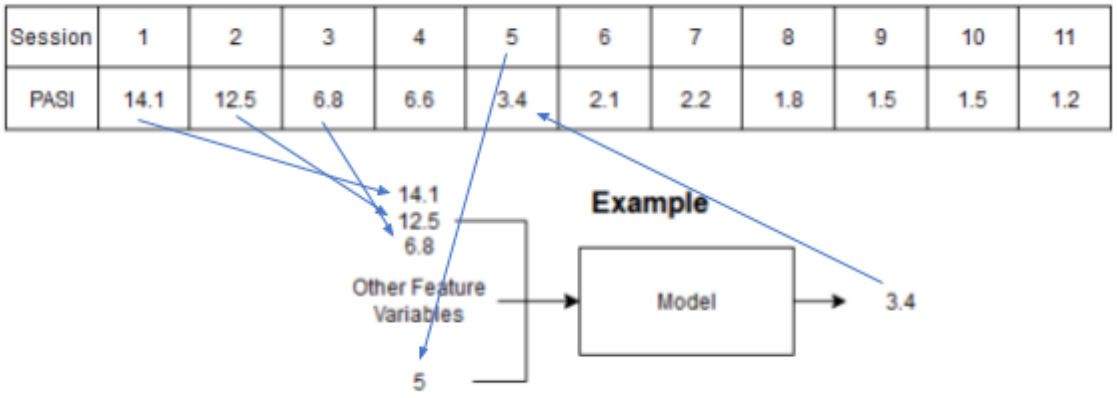


Figure 36 - Example Inputs/Output of Session Focused Model

I have chosen ANN architecture for the model, which requires values in the range $0 \leq x \leq 1$, so the values will be normalised and one hot encoded. The session value was an interesting value to deal with, it could be a continuous or categorial value. If dealing with it as a categorical value, it would have to be one hot encoded. The issue is that is with more unique values in a variable, the size of the one hot encoding increases rapidly. With the 10 session variables there would be nearly 1000 extra values, most of which are zeros. I decide this would to too much data and would slow the model learning down, so I chose to normalise instead.

Model

As specified in hyperparameter tuning it is impossible to use Grid Search to find the appropriate size of the hidden layers in my ANN model. I am going to use 2 hidden layers and experiment with the first hidden layer to achieve the best performance using trial-and-error.

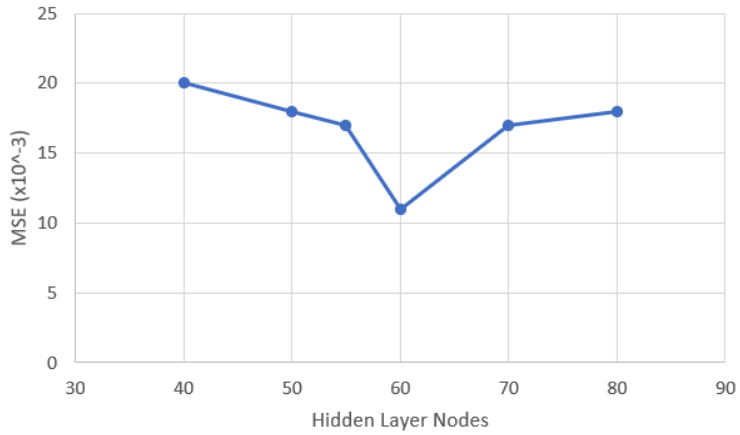


Figure 37 - Graph of MSE vs Hidden Nodes

The results of the experiment (Figure 37) show a significant performance gain at 60 nodes. Therefore, I will use 60 nodes as my first hidden layer.

Training

When training a model, the amount of training depends on the epochs and batch size. Since I am dealing with relatively small amounts of data, batch size can be 1. Epochs tell the model how many times to train off the data, changing how much the model is fitted to the data. I also take 5% of the training data for validation data.

I found that the optimal epoch level for this data set was around 150 epochs. Figure 38 shows the RMSE over 300 epochs vs Figure 39 which is over 150 epochs. The 300-epoch graph shows at 150 epochs the learning rate slows and stops learning. This is the point where overfitting begins and has marginal gains.

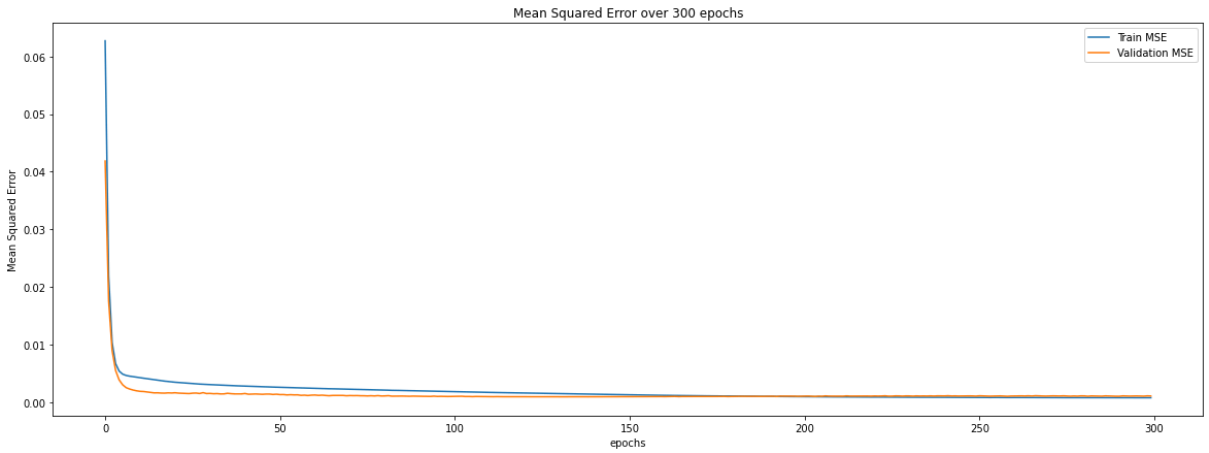


Figure 38 - Graph of Training for Session Focused Model Over 300 Epochs

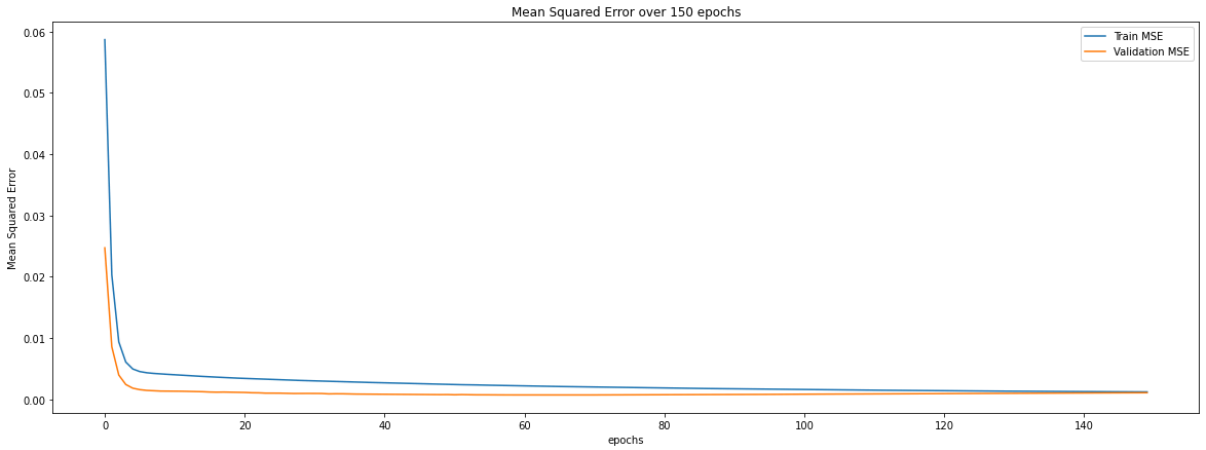


Figure 39- Graph of Training for Session Focused Model Over 150 Epochs

The performance metrics of both models:

	300 epochs:	**150 Epochs**
Mean Average Percentage Error	0.49036689799107	0.5353760041867648
Mean Absolute Error	0.04191066232140421	0.0432579213488685
Root Mean Square Error	0.235788	0.251426

Figure 40 - Comparison Table of Evaluation Metrics for 150 epochs vs 300 epochs

From the data in Figure 40, we can see the gains are very minimal and not worth double the training time, the differences could be down to a random sample. For these reasons the model will be trained with 150 epochs.

When training I will be using K-Fold technique to optimise the data usage, although I will only be doing one iteration of K-Fold learning as it would be infeasible to do many iterations.

I will also be creating a naïve model. Since when the model is defined it has randomised weights, it comes pre-randomised, so no random data needs generating.

Evaluation

Once a model has been fitted it can be evaluated. As previously discussed, the metrics for regression are Root Mean Squared Error, Mean Absolute Error, Mean Absolute Percentage Error

ANN and RNN multistep model

Introduction to principle

A multistep model is one which takes in all the previous sessions/weeks and predicts the next one. For example, the first model will take in sessions 1-3 and predict session 4. The second model use session 1-4 and predict session 5… An illustrated example is shown below in Figure 41.

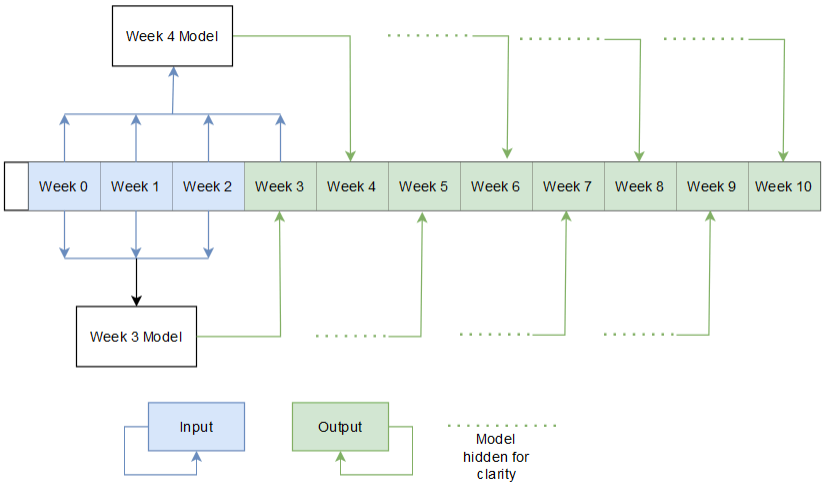


Figure 41- Graphic Showing the Process of Multi-Step Model Training

I will be using two different model types, ANN and RNN, and compare the evaluation results.

Data

The input data will be the first 3 weeks of PASI values and, depending on the curated or full dataset, other feature variables. Since both models require values in $-1 \leq x \leq 1$ range, the input values will be normalised and one hot encoded.

One issue I can foresee is the number of samples for each week, and the average value for each of those weeks. Below (Figure 42), shows the frequency distribution over all the sessions/weeks. As it shows there is a significant frequency drop off in the end sessions as people finish treatment.

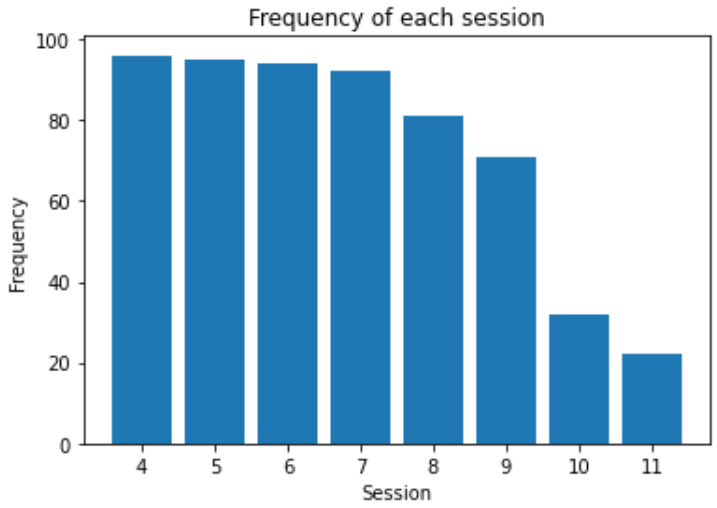


Figure 42 - Bar Chart Showing Frequency Distribution Across Sessions

This influences the average PASI gradient in the later sessions, shown in Figure 43. I theorise this is because people that require 9+ sessions of treatment tend to have a higher starting PASI, backed up by Figure 44. This means by the time most people have removed enough of the psoriasis, around 8 sessions, they still need treatment. I predict this will cause the model to predict a PASI increase at the 9th week.

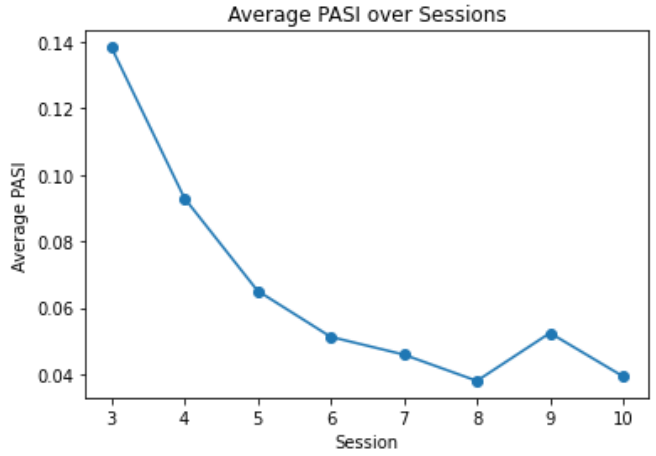


Figure 43- Graph of Average PASI Across the Sessions

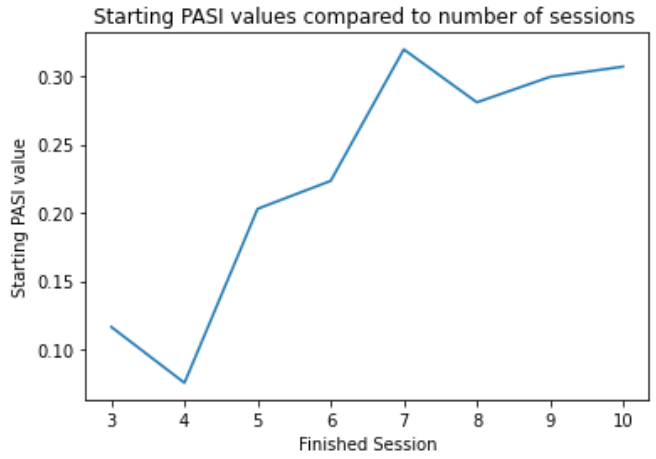


Figure 44 - Graph of Starting PASI values vs Number of Treatment Sessions

My solution is to replace the missing values, once a patient is finished, with zeros. For the few patients who dropped out early this isn't the case, but that is a small amount and should not affect the distribution greatly. This brings the average PASI graph in line with what is expected, shown in Figure 45.

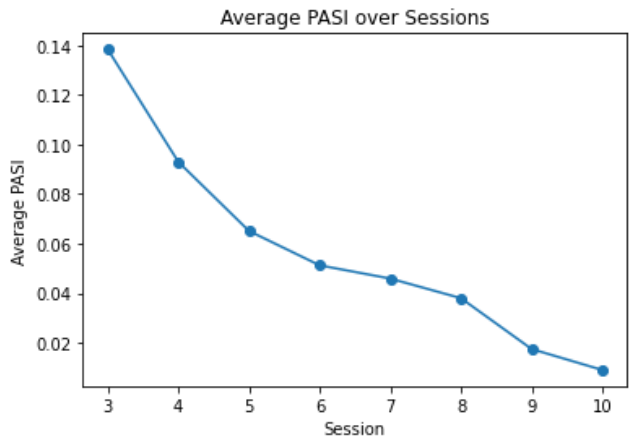


Figure 45 - Graph of Modified PASI Values Across Sessions

Models

There are 11 weeks in the treatment time, since the first three weeks are given, 8 models are required to predict a full series. There will be two sets of these 8 models, one with ANN models and one with RNN models.

For the ANN model I am using the optimal model found in the session focused model, as this data is very similar and should have similar performance.

As specified in my research, RNN models come in a few types (one-to-many, many-to-one, and many-to-many) which describes the mapping of the inputs to outputs. Since I am putting **many** variables into the models and getting **one** output, I will be using the many-to-one mapping. An RNN also takes *n* units which affects the learning. I have chosen to use one layer with *n* RNN units. I have graphed my trial-and-error experiment of unit sizes' effect on RMSE, shown in Figure 46.

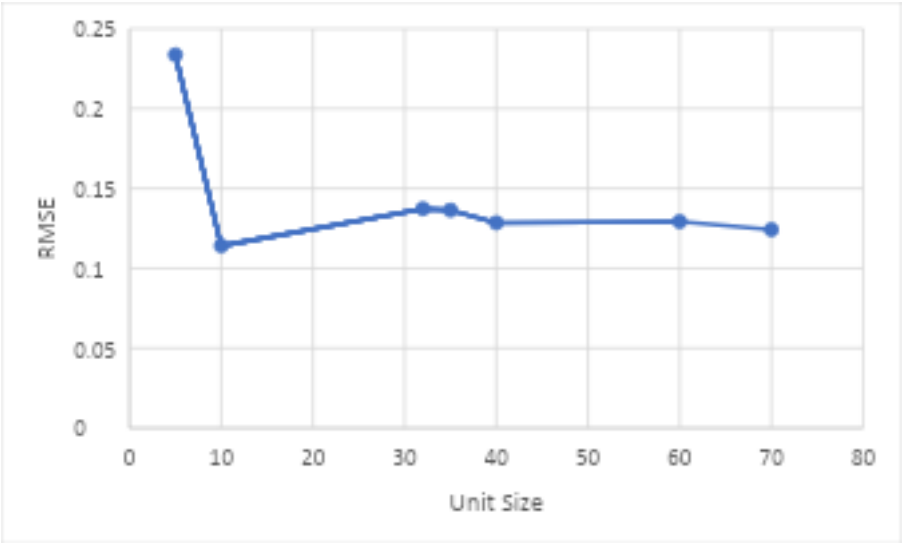


Figure 46 - Graph of RMSE vs RNN Units Size

While the lowest point is 10, I think this is too small and the improvement was probably down to the random data sample. I am going to go with 40 units as this should work well for all models.

Training

Training will be done simply in this with the data being fed into the models. I have once again gone with 150 epochs, as this should be a good level without overtraining.

Evaluation

The purpose of this is to test the effectiveness of ANN vs RNN on this data. I theorise that the RNN will perform well as this structured data is what it is built for.

Each of the 7 models in both model sets will produce the regression metrics: Root Mean Square Error, Mean Absolute Error, Mean Absolute Percentage Error. I can then directly compare the values as they use the same units.

5.3.3 Unsupervised

K-Means

Data

The training techniques I am using for K-Means will shadow the techniques used for the KNN training. Since both techniques use clustering, they will behave similarly. In accordance with this I am going to normalise, and one hot encode the data.

One difference in the data is the usage of PCA. For K-Means I am going to be using PCA as a dimensionality reduction and clustering tool. I will be comparing the model with and without PCA. SciKit Learn provides a very useful method for implementing PCA. Firstly, the cumulative explained variance needs to be found. This is the cumulative sum of explained variance from highest variance to lowest. Shown in Figure 47 is the cumulative variance.

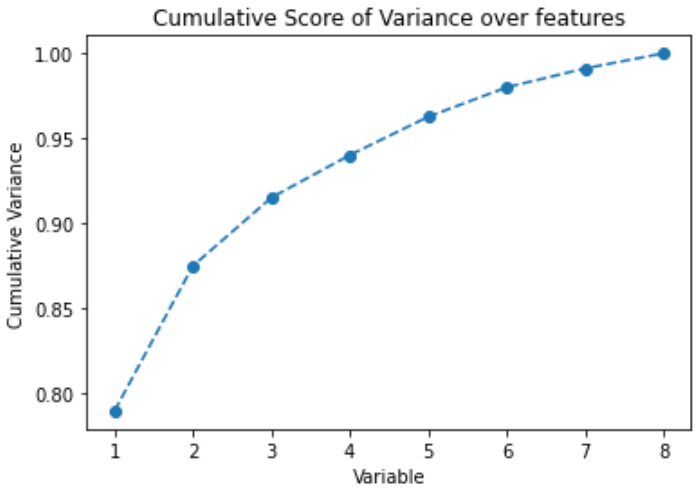


Figure 47 - Graph of Cumulative Explained Variance

For PCA we aim to keep around 85% of the variance. To keep this 85% variance, we take the first 2 variables. Using these variables, we apply PCA and transform the data.

Since there is no labelling to the data, we do not need to split data into test/train or create label variables.

Model

The model I will be using is the SciKit Learn K-Means library. This takes *n_clusters* as a parameter which constricts the number of clusters. I will also be comparing the performance with and without PCA. These methods will be applied to the curated, full, and naïve models, for comparison.

Training

I will be training 20 models with different *n_cluster* values to test which cluster size gives the best values. I will also be using K-Fold to maximise data usage.

Evaluation

To evaluate the K-Means models, I am using Within Cluster Set of Squares (WCSS) and Silhouette Score. These metrics will be taken for each cluster size to show how well the model performs over a

range of cluster sizes. Then the metrics are visualised by a line graph to show the changes in an easy-to-understand form.

Once I have the metrics for a range of clusters sizes for each model, I have to choose a *k* value which performs best. Each model will have to use the same *k* value to allow them to be compared. To find this cluster size we cannot just use the best metrics, because the model will always improve with the more clusters added. 1000 clusters will always perform better than 1 cluster, but it is very impractical to use 1000 clusters, so we use the elbow technique. An elbow is a point of inflection on the WCSS graph. An example of this is in Figure 48 which shows an obvious graph elbow at x=2.

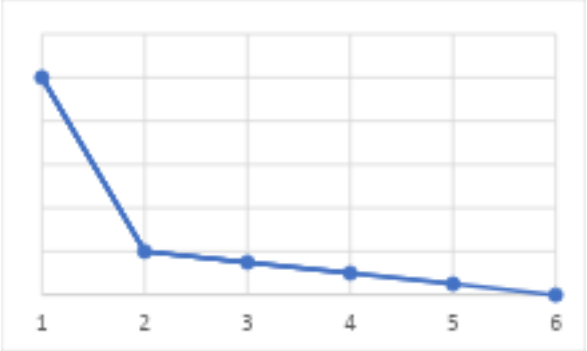


Figure 48 - Example Graph with Elbow

Chapter 6 – Model Evaluation

6.1 Classification

6.1.1 KNN

Data Set	K	Accuracy	Precision	Recall	F1-Score
Full	8	0.3125	0.3073	0.3125	0.3013
Curated	7	0.4375	0.53393	0.4375	0.4611
Naïve	7	0.25	0.2375	0.25	0.23077

Figure 49- Table of Results for KNN Models

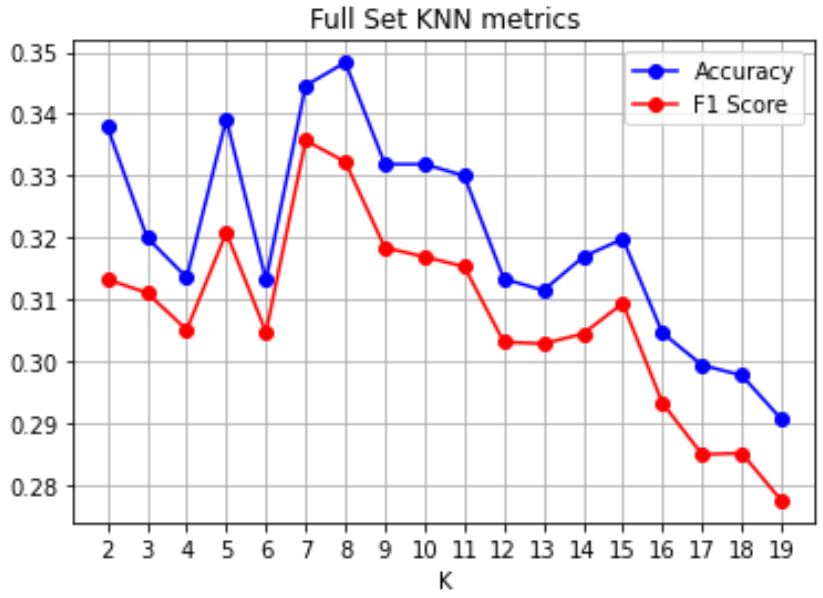


Figure 50 - Graph of Accuracy and F1-Score Across K Neighbours using Full Dataset

Figure 50 shows the accuracy and F1-Score of the model's performance on the test set over a range of K values. We can see there is a peak of accuracy and F1-Score of at K=8

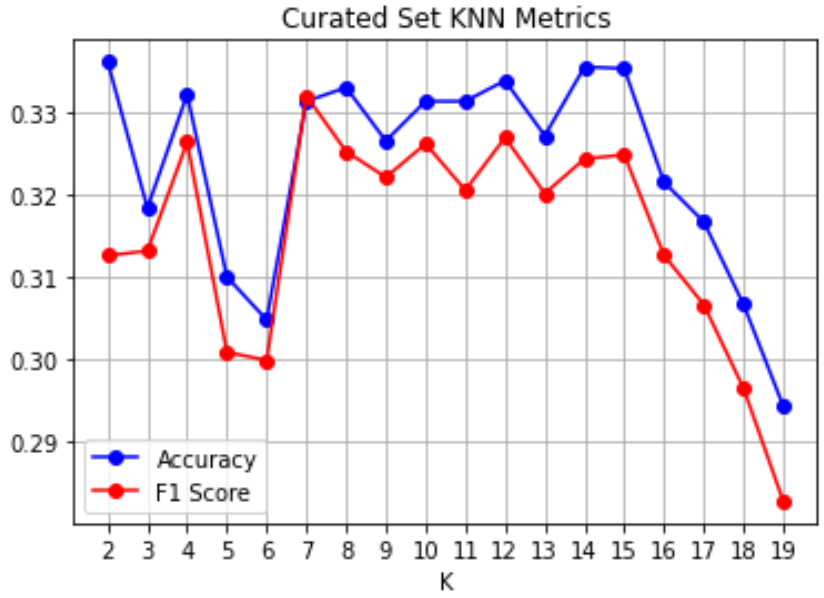


Figure 51 - Graph of Accuracy and F1-Score Across K Neighbours using Curated Datasets

Figure 51 shows the same graph for the curated dataset. This is less obvious of which value should be chosen as the lines seem to plateau. For this I would choose 7 neighbours as there is a peak of both accuracy and F1-Score. This is a good value as having too few neighbours is few sample points and that can be very affected by randomness. Having too many can cause the result to be very affected by outliers.

There is a large amount of uncertainty in these values. Running the model training again gives a very different set of results. I increased the iterations to get a higher sample size, but this had very little effect.

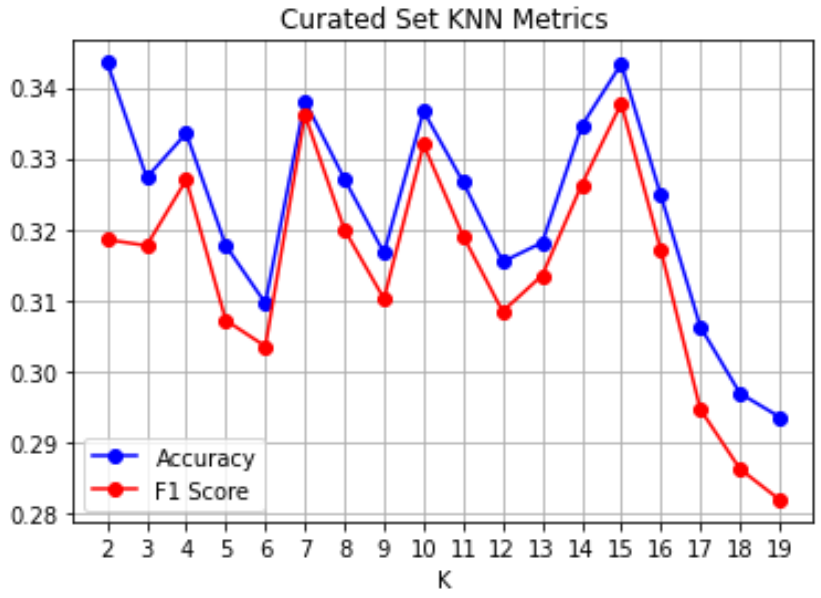


Figure 52- Re-Run Graph of Accuracy and F1-Score Across K Neighbours using Curated Dataset

Figure 52 shows another round of learning which has got very different results. I believe this stems from a small dataset and this problem not being suited for clustering algorithms. This belief is backup by the low accuracy of the trained models compared to the naïve models in Figure 49.

When comparing the models in Figure 49 we can see, as predicted, that the curated dataset performs better than the full dataset. While both models are superior to the naïve model, the difference is not great. The trained models are only marginally better than naïve model and would not be enough to reliably inform a patient.

6.1.2 Random Forest

To find the best hyperparameters I used grid search cross validation to test *n_estimators={10,100,1000}* and *max_depth={10,100,1000}*. My assumption would be that the larger the estimators and depth, the more accurate the results. This turned out to be false as the best combination was *max_depth=10* and *n_estimators=100*. The larger values must have been overfitting to the data and causing the performance to drop. This was true for both datasets.

Data Set	**Accuracy**	**Precision**	**Recall**	**F1-Score**
Full	0.6489583	0.69556	0.64896	0.64757
Curated	0.6557292	0.68642	0.65573	0.65214
Naïve	0.2875	0.33061	0.2875	0.26344

Figure 53 - Table of Results for Random Forest Models

Figure 53 shows another surprising result in the difference between the full and curated dataset. All other models have a big difference between the datasets. This is especially apparent in the KNN model where there is a 30% difference in accuracy. My theory is that since the features are not directly compared to each other, there is no influence on the useful data by the less useful data. I do think there will always be a slight difference in performance as the full dataset, when the random forest is randomly choosing features, has a larger set of features to choose from and will choose less useful features more often.

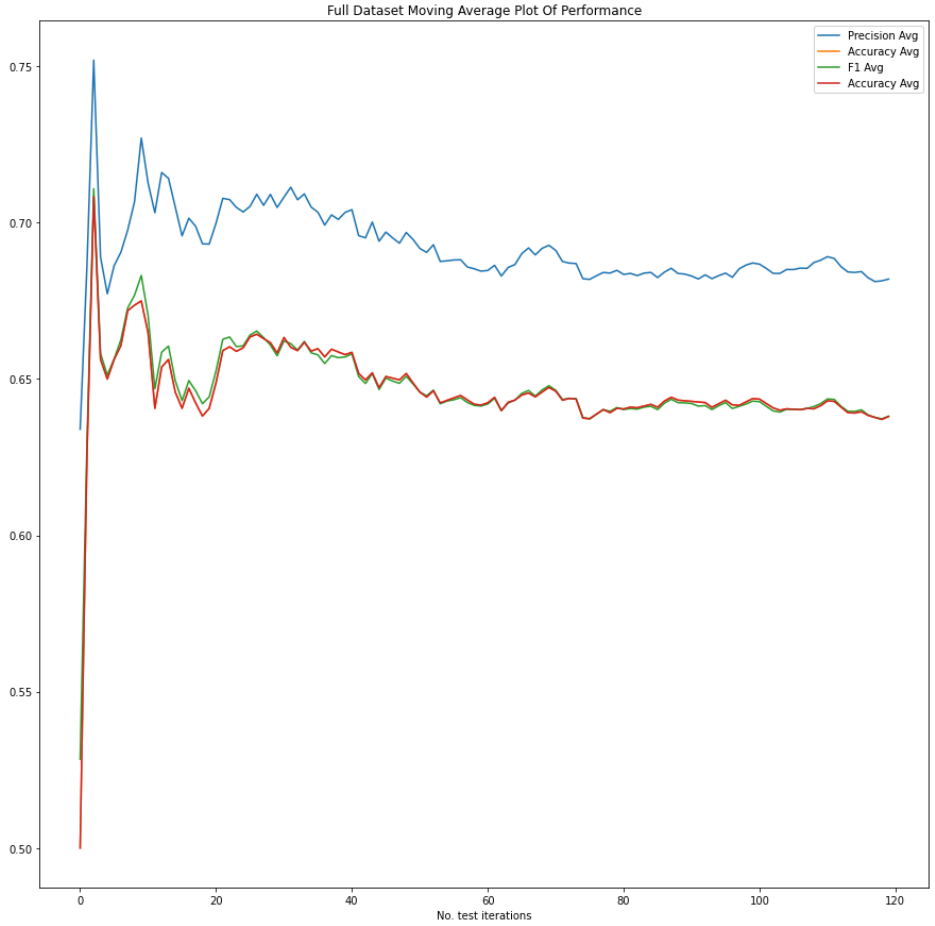


Figure 54 - Graph of the Moving Averages During Training Using Full Dataset

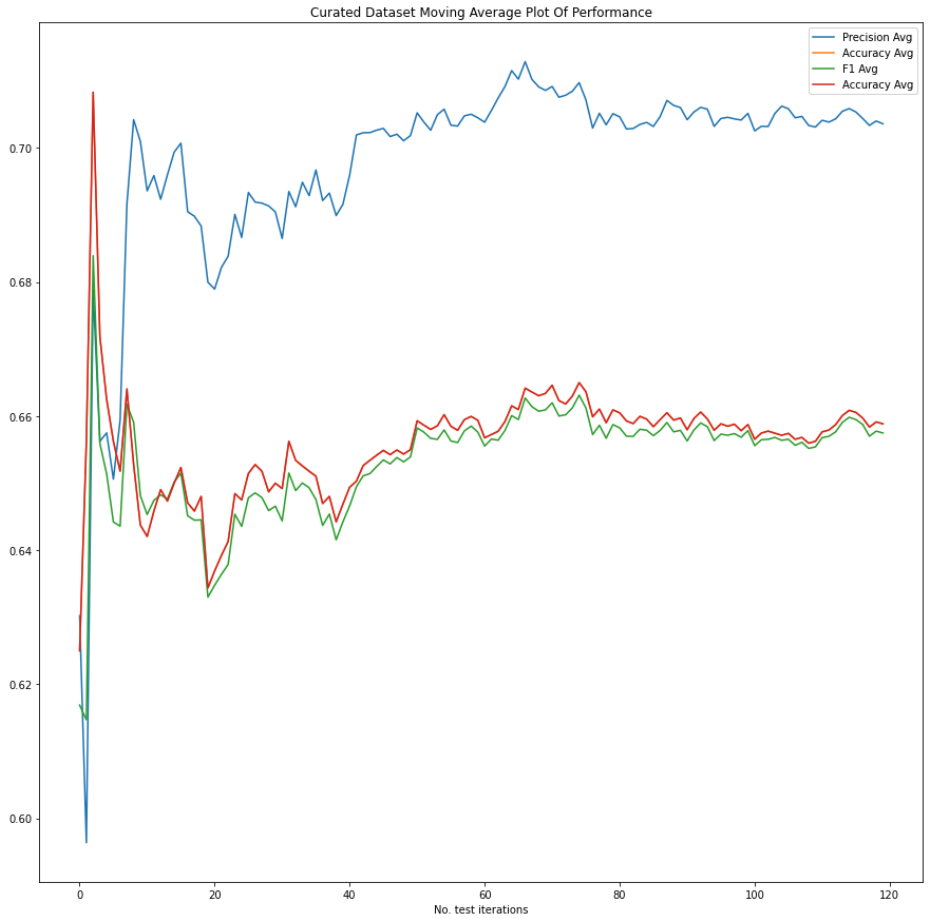


Figure 55 - Graph of the Moving Averages During Training Using Curated Dataset

The graphs above (Figure 54, Figure 55) show the moving averages of the performance metrics. As we can see the full dataset lines are smoother, compared to the curated dataset, which shows us the performance of the curated set was much less consistent. I have no explanation for this, but it would give a reason to choose the full dataset in this instance.

This model did perform well on this dataset. For the naïve model we should have expected an accuracy of 33%, so the naïve model slightly underperformed in this case. Even with this accounted for, the trained models performed twice as well in every category making this a good predictor.

6.2 Regression

6.2.1 Session Focused Model

Dataset	**RMSE**	**MAE**	**MAPE**
Full	0.08136	0.066383	0.771005
Curated	0.061748	0.043654	0.584162
Naïve	0.52077	0.511124	0.873988

Figure 56 - Table of Results for Session Focused Model

As expected, Figure 56 shows that the curated dataset has performed the best and the naïve has performed the worst. These results show the performance of the model over every week, but this is not very useful for the context. The most valuable results for the patient are the last 2 weeks (weeks 10 and 11).

Dataset	**RMSE**	**MAE**	**MAPE**
Full	0.03268	0.02602	0.72064
Curated	0.03287	0.02689	0.53558

Naïve	0.60022	0.59231	0.89101

Figure 57 - Table of Results for Weeks 10 and 11

Figure 57 shows the performance of the model on weeks 10, 11. As we can see the model performs better than average on these last weeks. The difference between the datasets is very minimal in this instance.

6.2.2 ANN and RNN multistep Model

My prediction for this approach was that the RNN model would outperform the ANN model as it could understand the week structure more than the ANN would.

ANN model

	RMSE		MAE		MAPE	
Week	**Full**	**Curated**	**Full**	**Curated**	**Full**	**Curated**
4	0.09316	0.09933	0.07852	0.08082	0.58016	0.61737
5	0.06970	0.08502	0.05151	0.06907	0.55767	0.76342
6	0.05052	0.07356	0.02752	0.04495	1.01342	2.03853
7	0.04654	0.05140	0.03326	0.03149	1.51663	1.48625
8	0.04202	0.04441	0.03133	0.02447	3.55954	6.85505
9	0.03515	0.02849	0.02603	0.02169	30.8130	2.00707
10	0.03655	0.02461	0.01872	0.01558	77.2645	5.86040
11	0.01403	0.02143	0.00748	0.01125	4.85296	5.19514
AVERAGE	0.048459	0.052281	0.034296	0.037415	15.01974	3.102904
Naïve Model	0.458895	0.454482	0.455626	0.451300	0.891287	0.895565

Figure 58 - Table of Results for ANN Multistep Model

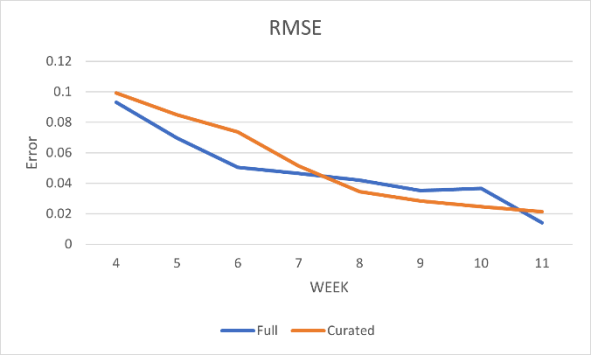


Figure 59 - Graph of RMSE Across All ANN Multistep Models

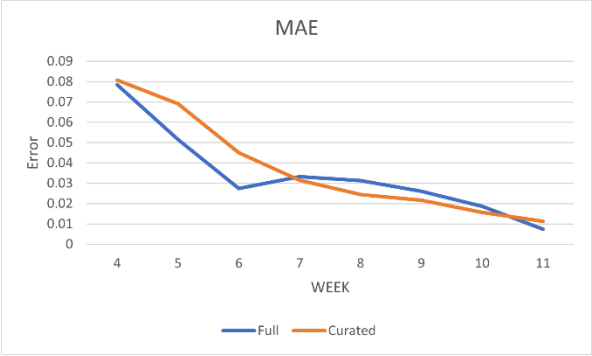


Figure 60 - Graph of MAE Across All ANN Multistep Models

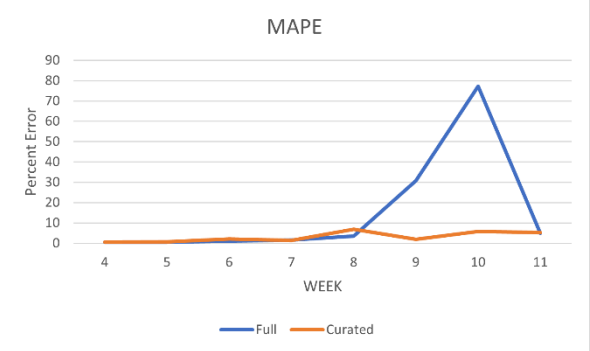


Figure 61 - Graph of MAPE Across All ANN Multistep Models

From the data (in Figures 58, 59, 60), we can see there is very little difference between the datasets. The difference is mainly down to random sample chance. The most important prediction models are the last few weeks (week 10/11), as that is the most relevant to the context of the project because that is the final outcome. Interestingly I expected the prediction to get worse as the weeks continued because the number of samples decreases towards week 11. This is actually the opposite of what happens.

In this instance I think MAPE is an ineffective metric. I think there must have been some very large percentage errors which skewed the data heavily, shown in Figure 61. I came to this conclusion since the changes in MAPE do not line up with RMSE and MAE, which are consistent with each other. In this instance I am going to ignore it.

This model has been successful as there is a 10-fold increase in average performance in RMSE and MAE compared to the naïve model.

RNN model

	RMSE		MAE		MAPE	
Week	**Full**	**Curated**	**Full**	**Curated**	**Full**	**Curated**
4	0.10797	0.09878	0.08962	0.08277	0.69853	0.55050
5	0.43274	0.41285	0.43041	0.40815	0.85714	0.81684
6	0.38924	0.45933	0.38622	0.45831	0.85618	0.90088
7	0.51971	0.41025	0.51902	0.40669	0.92184	0.87767
8	0.44252	0.45713	0.44138	0.45563	0.92224	0.91493
9	0.46018	0.49030	0.45937	0.48952	0.95759	0.95258
10	0.51722	0.51173	0.51692	0.51063	0.98046	0.96739
11	0.51921	0.51069	0.51885	0.51004	0.97783	0.97741
AVERAGE	0.423599	0.418883	0.420224	0.415218	0.896476	0.869775
Naïve Model	0.442817	0.45529	0.439930	0.452028	0.890399	0.882065

Figure 62 - Table of Results for RNN Multistep Model

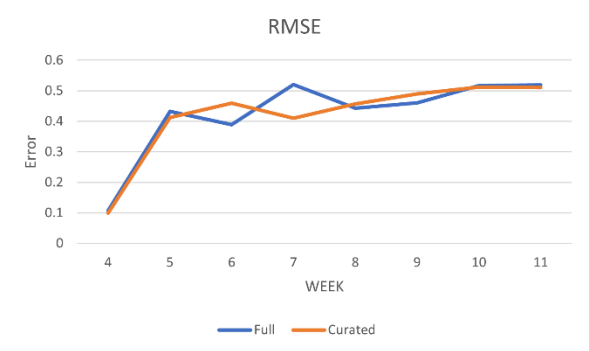


Figure 63 - Graph of RMSE Across All RNN Multistep Models

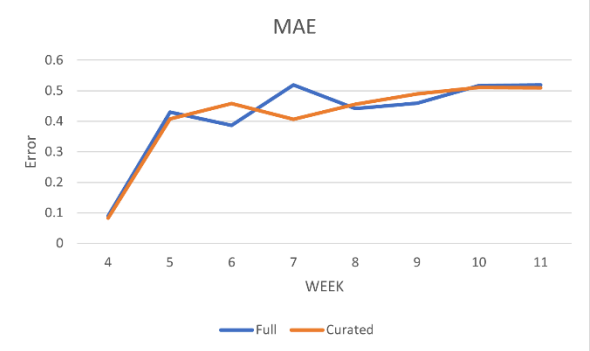


Figure 64 - Graph of MAE Across All RNN Multistep Models

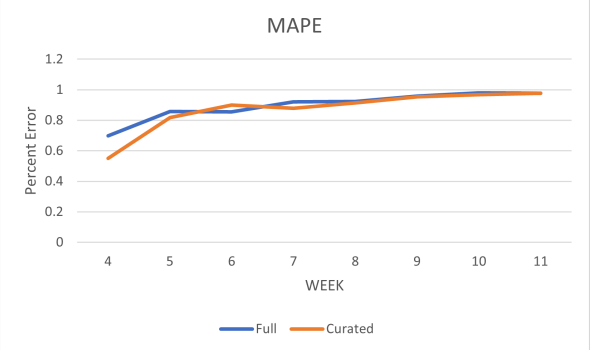


Figure 65 - Graph of MAPE Across All RNN Multistep Models

Figures 62-65 show that again, there is no significant difference between the full and curated dataset. Given the full dataset has more data, it will have taken longer to train compared to the curated dataset, but the results are very similar. This shows the data I removed had no effect on the performance.

Interestingly the error distribution is the opposite to the ANN. The error increases through the weeks (Figures 63 – 65) as opposed to the ANN which decreased. This is more in line with what I expected because the sample size decreases as the weeks increase. If this did occur, I would have expected the RNN model error to decrease not the ANN. This is because the RNN should perform better with more structured data, not the ANN.

Comparing this model to the naïve model shows how disastrously this model has performed. It performs very marginally better than the naïve model. This is very surprising as I thought it would be competitive with the ANN model. This could be down to the data the RNN had access too. Since RNN models can only take in temporal data, it didn't have access to other variables like BMI or AGE, whereas the ANN did.

Comparison

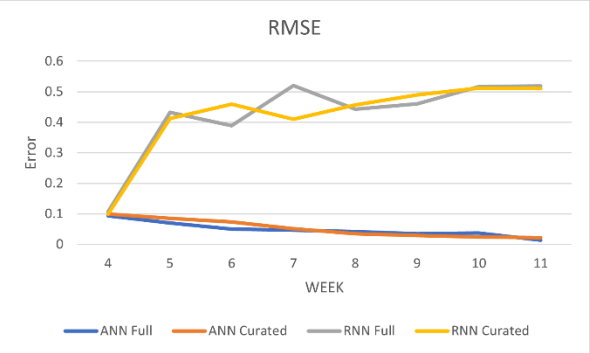


Figure 66 - Graph Comparing RMSE Between RNN and ANN Multistep Models

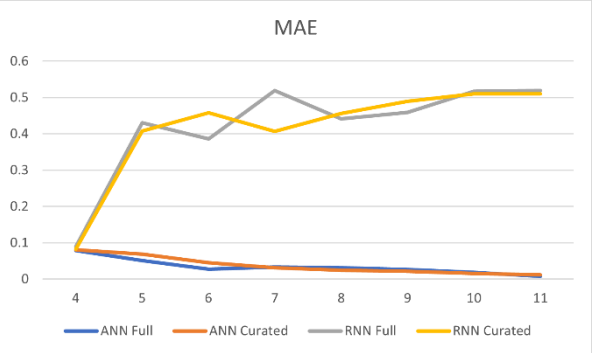


Figure 67 - Graph Comparing MAE Between RNN and ANN models

Figures 66 and 67, shows the comparison of RNN to ANN data. The ANN is the very obvious winner. In every single metric the ANN vastly outperforms the RNN. This is interesting as it is the opposite of my expectations. My theory is that the time series are not long enough to properly utilise the RNNs potential, and it didn't have access to all the information.

6.3 Unsupervised

6.3 K-Means

My previous prediction was that this problem was not suited for clustered learning, I will explore that hypothesis using the results.

Without PCA

The results in this section will be from a model run without PCA being used on the input data.

Full

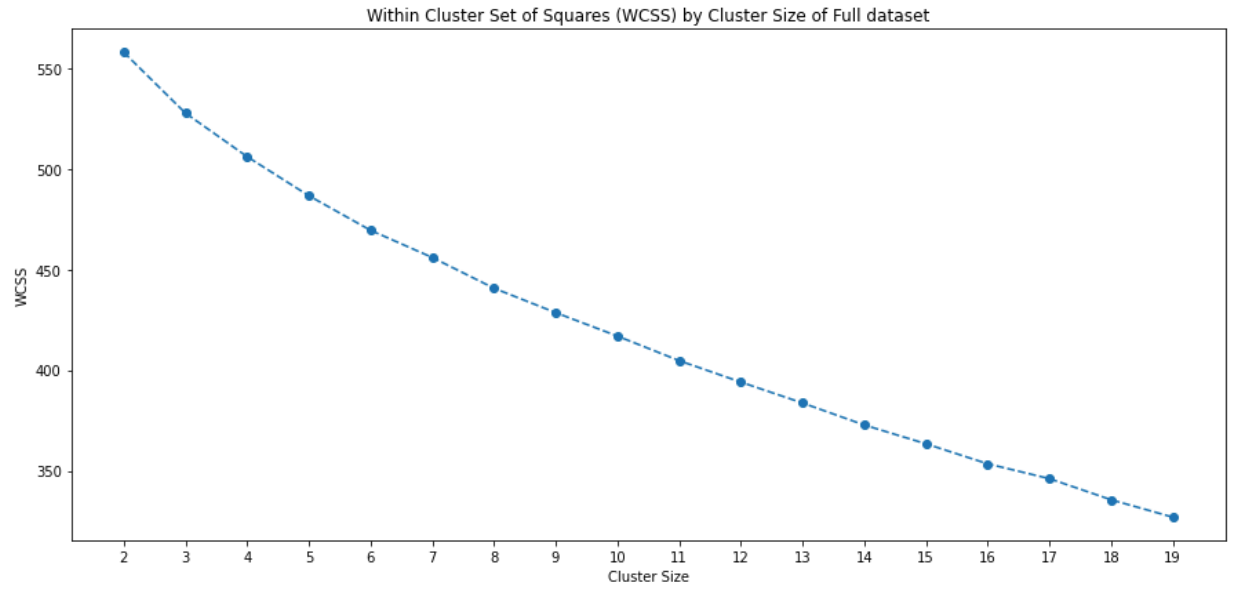


Figure 68 - Graph of Average WCSS Across K Values in K-Means Model (Without PCA) Using Full Dataset

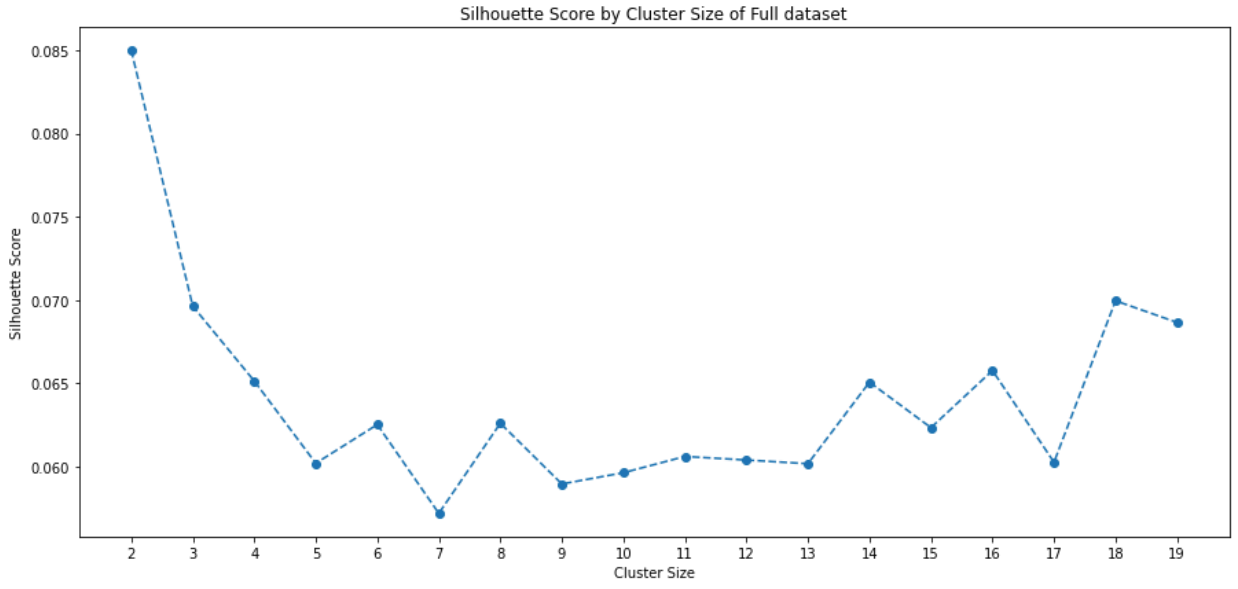


Figure 69- Graph of Average Silhouette Score Across K Values in K-Means Model (Without PCA) Using Full Dataset

Curated

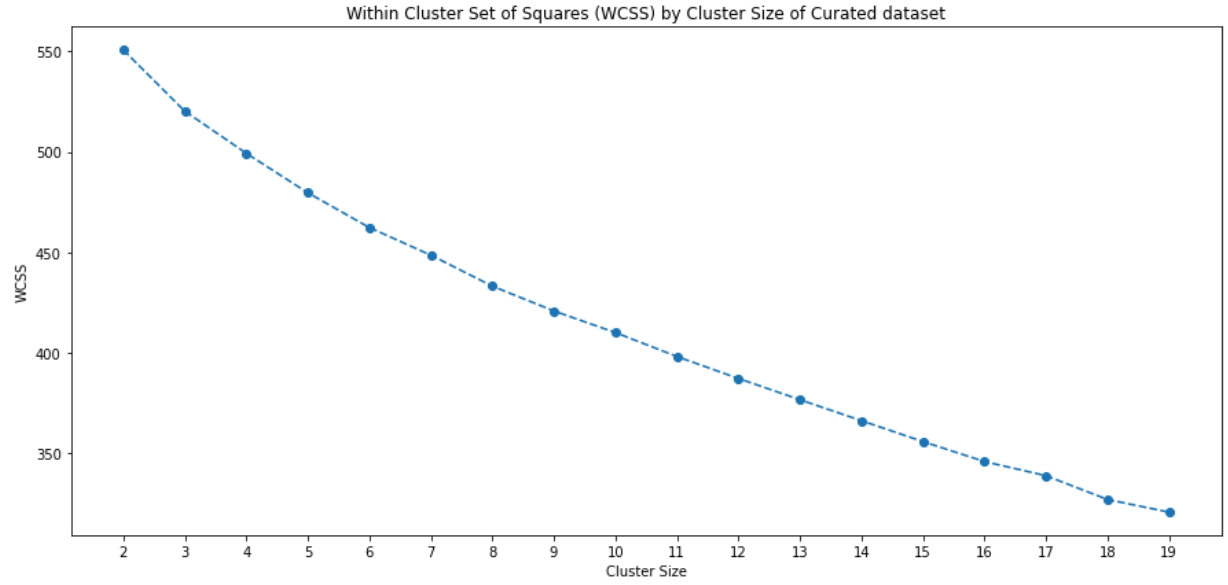


Figure 70 - Graph of Average WCSS Across K Values in K-Means Model (Without PCA) Using Curated Dataset

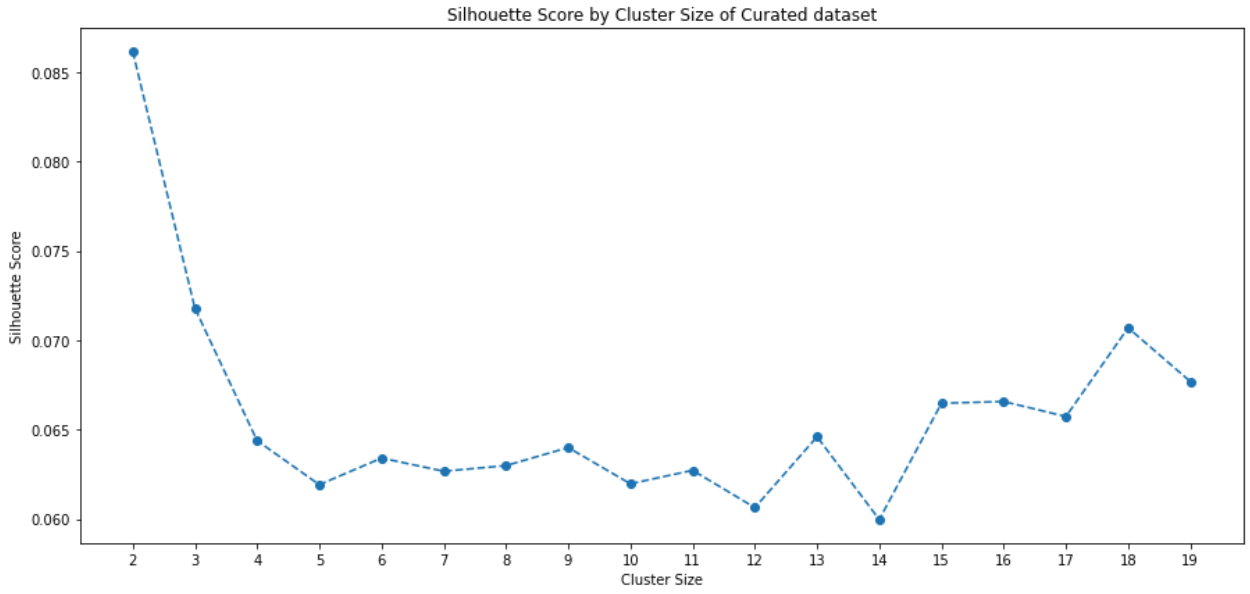


Figure 71 - Graph of Average Silhouette Score Across K Values in K-Means Model (Without PCA) Using Curated Dataset

From Figures 68 – 71, an optimal K value needs to be found. The first step is to find the graph "elbow" discussed in the development section. Unfortunately, on the graph there is no obvious graph elbow, which is a sign K-Means is not a good algorithm for this data. I will be optimising for Silhouette score instead. In both cases we can see an improved Silhouette score at 18 cluster size, therefore I will be using that.

Comparison

Dataset	**WCSS**	**Silhouette Score**
Full	339.7265637	0.069263
Curated	330.0562276	0.072165
Naïve	225.4766695	0.020352

Figure 72 - Table of Results From K-Means Model (without PCA)

The *k* size was chosen to maximise silhouette score. In terms of the silhouette score, everything is as expected. The curated dataset outperformed the other models, and the naïve model has performed worst. While this is as expected, the difference in values is very small. The silhouette score can range from $-1 \leq x \leq 1$ and these values are very close to 0. This supports the theory that clustering models are not suited to this problem. Also supporting this theory is the WCSS. While we were not optimising for it, it does show that the naïve model has significantly outperformed the others.

With PCA

Full

The first step of PCA is finding the number of variables to use. This is found using the graph of cumulative explained variance (figure 73).

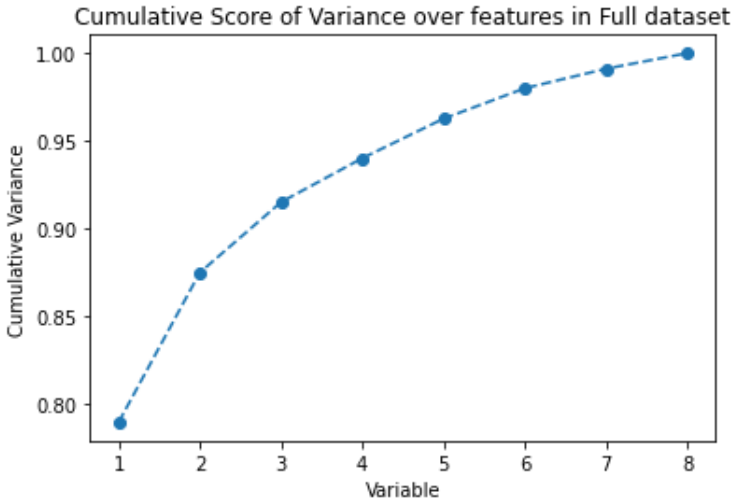


Figure 73 - Graph of Cumulative Score of Variance in Full Dataset

To achieve the 85% of cumulative variance, which is best for PCA, I chose to use 2 variables. Using this, Figures 74 and 75 were generated.

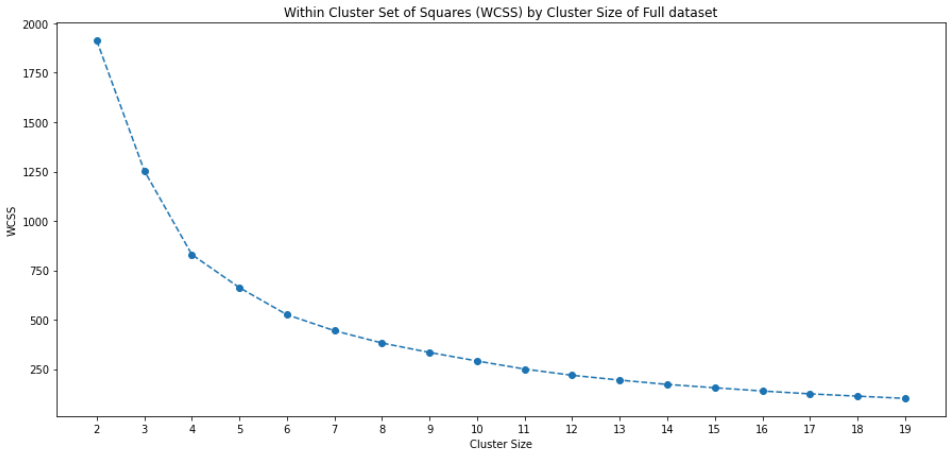


Figure 74 - Graph of Average WCSS Across K Values in K-Means Model (With PCA) Using Full Dataset

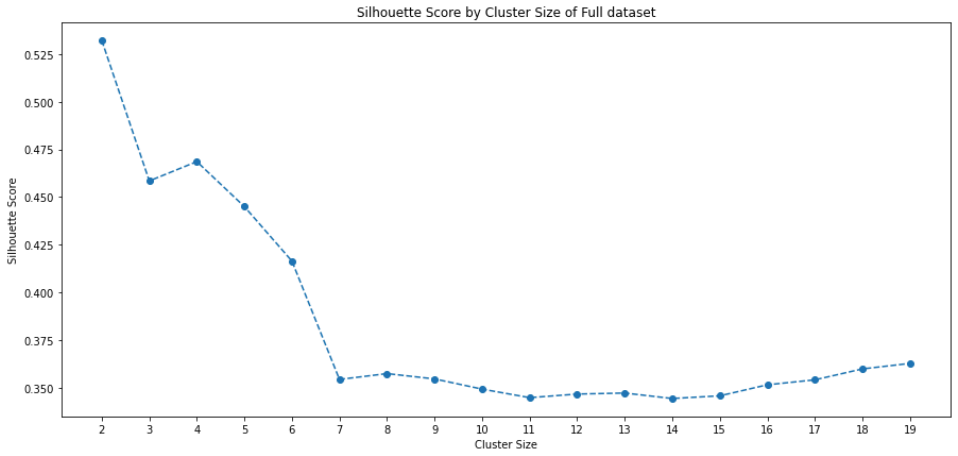


Figure 75 - Graph of Average Silhouette Score Across K Values in K-Means Model (With PCA) Using Full Dataset

Curated

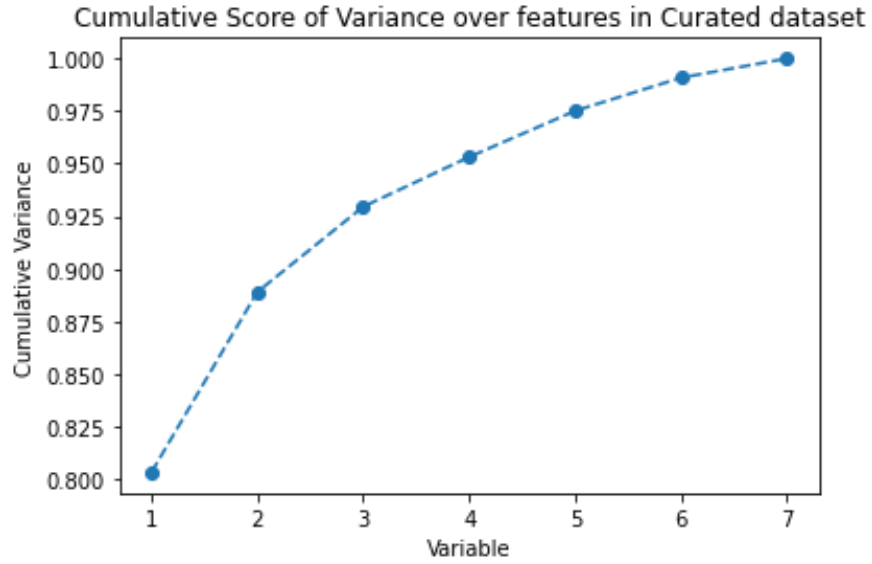


Figure 76 - Graph of Cumulative Score of Variance in Curated Dataset

PCA variable size needs to be found for the curated dataset, from Figure 76. Once again, I decided the optimal number of variables is 2. Using this, Figures 77 and 78 were generated.

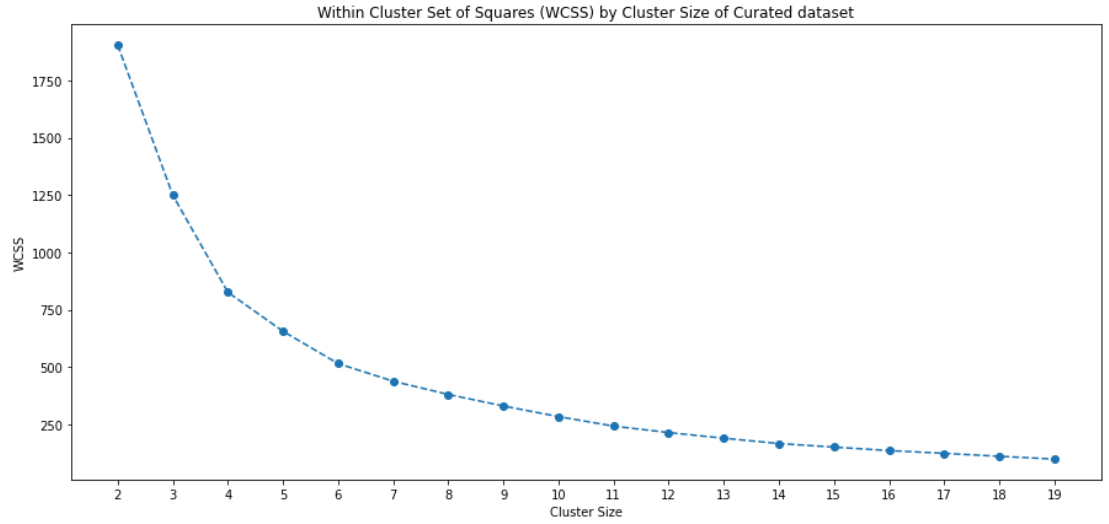


Figure 77 - Graph of Average WCSS Across K Values in K-Means Model (With PCA) Using Curated Dataset

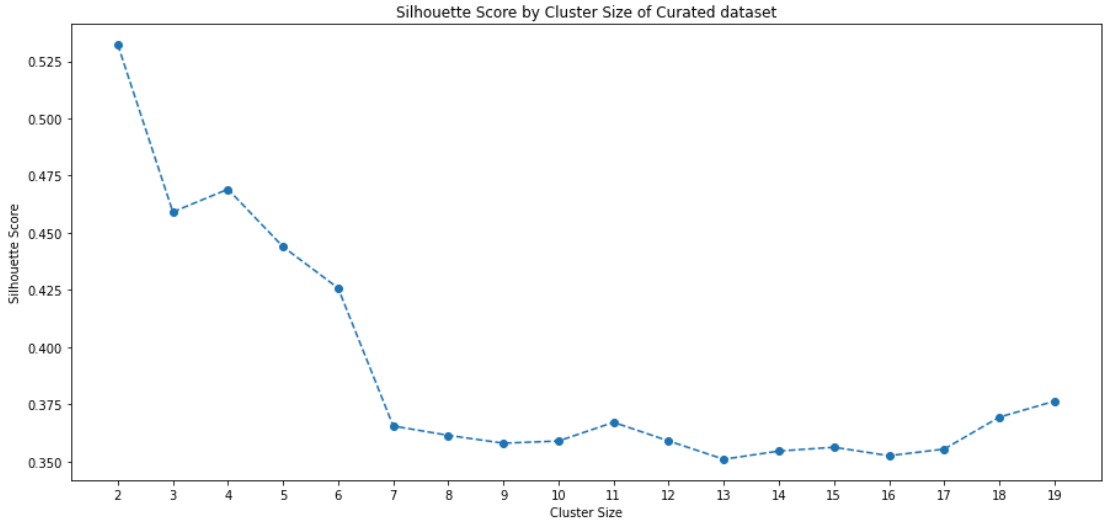


Figure 78 - Graph of Average Silhouette Score Across K Values in K-Means Model (With PCA) Using Curated Dataset

To make a comparison with the non-PCA models, I will have to use the same number of clusters. I think 18 clusters does still perform well enough on this model, so it should be a fair comparison.

Comparison

Dataset	**WCSS**	**Silhouette Score**
Full	234.71394	0.35934
Curated	232.27845	0.36938
Naïve	0.4871311	0.44410

Figure 79 - Table of K-Means Model (with PCA)

The data from both models is collected in Figure 79. The curated data has, as expected, performed better than the full dataset showing the improvements made by the data analytics. Compared to the non-PCA data, the silhouette scores and WCSS are also much better, showing the data transformation has improved performance significantly. The most significant changes were made to the naïve model. Performing PCA cause a 21x improvement in the silhouette score and a 450x improvement to the WCSS, showing PCA technique can work under the right circumstances.

Chapter 7 – Conclusion

In this section I will conclude the success of this project by looking at the component parts and how they affected the overall outcome.

7.1 Analysis Conclusion

From my analysis I generated a curated dataset to optimise the performance of the models. In each stage I have compared the full dataset with my curated dataset. I believe the curated dataset has helped improve the performance. In most cases the performance has either not changed or increased with the improvements of the curated dataset. Therefore, I think the analysis was successful and would proceed with the variables in the curated dataset.

The standout discovery made was the correlation between starting PASI value and average PASI gradient (Figure 25). This information can help a patient before any treatment is necessary.

7.2 Model Performance

7.2.1 Classification

KNN

Data Set	K	Accuracy	Precision	Recall	F1-Score
Full	8	0.3125	0.3073	0.3125	0.3013
Curated	7	0.4375	0.53393	0.4375	0.4611
Naïve	7	0.25	0.2375	0.25	0.23077

Figure 80 - Table of Results for KNN Models (Same as Figure 45)

Random Forest

Data Set	Accuracy	Precision	Recall	F1-Score
Full	0.6489583	0.69556	0.64896	0.64757
Curated	0.6557292	0.68642	0.65573	0.65214
Naïve	0.2875	0.33061	0.2875	0.26344

Figure 81 - Table of Results for Random Forest Models (Same as Figure 49)

When comparing which classification model was more successful, using Figures 80 and 81, I think the Random Forest model was obviously the more successful model. It was much better in every measurable way. This also leads me to believe that this problem is not suited to clustering models, as the trained models' accuracy was not significantly better than the naïve model for KNN.

7.2.2 Regression

To make the results easier to understand in the context of the problem, I have de-normalised the values in MAE, which gives the uncertainty as a PASI value. To de-normalise I rearranged the min-max normalisation equation. I only de-normalised MAE as it gives the absolute error, in terms of PASI, whereas RMSE alters the value.

Min-Max normalisation: $y = \frac{x-min}{max-min}$

Rearranges to: $x = y(max - min) + min$

For the PASI values:

Max = 31.8

Min = 0

Session Predictor

Dataset	RMSE	MAE (De-normalised)	MAPE

Full	0.08136	2.110979	0.771005
Curated	0.061748	1.388197	0.584162

Figure 82 - Table of Results for Session Focused Model (Same as Figure 52)

Multistep

	AVG RMSE		AVG MAE (De-normalised)		AVG MAPE	
Model Type	**Full**	**Curated**	**Full**	**Curated**	**Full**	**Curated**
RNN	0.423599	0.418883	13.36312	13.20393	0.896476	0.869775
ANN	0.048459	0.052281	1.090613	1.189797	15.01974	3.102904

Figure 83 –Table of Average Results of RNN and ANN Multistep Models (Taken from Figure 54 and Figure 58)

From Figures 82 and 83 we can compare the results of session predictor and multistep. The best performance models were Multistep Full-dataset ANN and Session Focused Curated models. While they both perform well the multistep model is best. While these results are important, the most important ones are the final weeks as these are most useful to the context of the problem.

Multistep Full Dataset ANN Weeks 10,11

Week	**RMSE**	**MAE (de-normalised)**	**MAPE**
10	0.03655	0.595296	77.2645
11	0.01403	0.237864	4.85296
Average	**0.02529**	**0.41658**	**41.0587**

Figure 84 - Table of Results for Multistep Full-Dataset ANN on Weeks 10 and 11

Session Focused Curated Dataset Weeks 10, 11

Dataset	**Avg RMSE**	**Avg MAE (de-normalised)**	**Avg MAPE**
Curated	0.03287	0.855102	0.53558

Figure 85 - Table of Results for Session Focused Curated-Dataset on Weeks 10 and 11

As can be seen on Figure 84, the multistep can predict the PASI values for weeks 10 and 11 to an accuracy of 0.41658. Figure 85 shows the session focused model can predict it with 0.855 uncertainty. Based off this the multistep model is superior.

7.2.3 Unsupervised

Dataset	**WCSS**	**Silhouette Score**
Curated (with PCA)	232.27845	0.36938
Naïve (With PCA)	0.4871311	0.44410

Figure 86 - Best K-Means Performances

Figure 86 shows the best performances for the K-Means model. As can be seen the naïve model was much more successful, which emphasises the theory that K-Means is not suited to this problem and should not be used.

7.2.4 Overall Comparison

Random Forest vs Multistep.

These models perform the best in their respective categories, so which one is better? This is a hard question to answer as there is no way to directly compare a classification model to a regression model, as each has their advantages and disadvantages. Personally, I think the multistep model is more informative as it can predict within less than 1 PASI the final outcome. Despite this, some people might find the classes which random forest predicts to be more informative.

7.3 Risks

At the start of the project, I came up with some risks I believed I would face in the project. I will now assess if the risks if they came up and if they affected the project.

1. Lack of experience in the time series models

At the start of the project this did slow me down. When researching I came across lots of information which was beyond my knowledge and required further reading to understand. As the project developed, I became more accustomed with the techniques and terminology, so the risk was mitigated.

2. Small dataset

I think this was a fundamental problem with the project that I could not get around. I did mitigate the issue by using techniques such as K-Fold and PCA and I did manage to get a good result in the end. I think this project would benefit greatly from a bigger dataset.

3. Data has gaps

In the end this risk was not a problem. A combination of removing variables with lots of missing values and using data imputation to fix the rest, gave me a good solid dataset which I could use.

7.4 Future Work

As repeatedly mentioned, the dataset I have been using is small. To continue I would start by collecting more data, preferably from more places. This dataset has been taken from a Newcastle hospital which means the patients are from the surrounding area. There may be an underlying environmental factor which is affecting the treatment.

I would also like some examples of treatment which was unsuccessful, as pretty much all the patients in this dataset had a good reaction to treatment and most had at least 70% reduction in PASI. Having some negative reactions would allow me to compare the variables with successful treatment to see if there is a connecting factor/s. It would also allow me to see if there are any variables which are an indicator of psoriasis.

Machine learning is a huge area of research, with many more types of models than I could explore here. To continue with this project, I would start by experimenting with more models. Some models I wanted to try were the Naïve Bayes Classifier, Bayesian Linear Regression and many more. Another classification technique which could be successful would be to use ensemble models, which is when multiple model types are polled for their prediction and the most common answer is taken from them. This utilises the strengths of many models and mitigates each individual weakness. In my research I learnt about RNN mappings. I would like to try another mapping technique where I give the model **many** variables and predict the whole time series at once. This would be the many-to-many mapping.

References

AlexSoft (2018) *Machine Learning Project Structure: Stages, Roles, and Tools | AltexSoft*, *AlexSoft*. Available at: https://www.altexsoft.com/blog/datascience/machine-learning-project-structure-stages-roles-and-tools/ (Accessed: May 12, 2022).

Alphabet Inc (2022) *Google Colaboratory*.

Bhosle, M.J. *et al.* (2006) "Quality of life in patients with psoriasis," *Health and Quality of Life Outcomes*, 4(1), p. 35. doi:10.1186/1477-7525-4-35.

Brezinski, E.A., Dhillon, J.S. and Armstrong, A.W. (2015) "Economic Burden of Psoriasis in the United States," *JAMA Dermatology*, 151(6), p. 651. doi:10.1001/jamadermatol.2014.3593.

GeeksForGeeks (2021) *Decision Tree , GeeksForGeeks*. Available at: https://www.geeksforgeeks.org/decision-tree/ (Accessed: May 12, 2022).

Handelman, G.S. *et al.* (2018) "eDoctor: machine learning and the future of medicine," *Journal of Internal Medicine*, 284(6), pp. 0–10. doi:10.1111/joim.12822.

Harris, C.R. *et al.* (2020) "Array programming with NumPy," *Nature*, 585(7825), pp. 357–362. doi:10.1038/s41586-020-2649-2.

Huang, L. *et al.* (2019) "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records," *Journal of Biomedical Informatics*, 99, p. 103291. doi:10.1016/J.JBI.2019.103291.

IBM (2020) *What is Exploratory Data Analysis? - United Kingdom | IBM*, *IBM Cloud Education*. Available at: https://www.ibm.com/uk-en/cloud/learn/exploratory-data-analysis (Accessed: May 12, 2022).

Jordan, J. (2018) *Organizing machine learning projects: project management guidelines.*, *jeremyjordan*. Available at: https://www.jeremyjordan.me/ml-projects-guide/ (Accessed: May 12, 2022).

Koo, J. *et al.* (2017) "Depression and suicidality in psoriasis: review of the literature including the cytokine theory of depression," *Journal of the European Academy of Dermatology and Venereology*, 31(12), pp. 1999–2009. doi:10.1111/jdv.14460.

Koo, J. and Lebwohl, M. (1999) "Duration of remission of psoriasis therapies," *Journal of the American Academy of Dermatology*, 41(1), pp. 51–59. doi:10.1016/S0190-9622(99)70406-8.

Korstanje, J. (2020) *Is Python faster than R?. R vs Python Speed Benchmark on a simple… | by Joos Korstanje | Towards Data Science*. Available at: https://towardsdatascience.com/is-python-faster-than-r-db06c5be5ce8 (Accessed: April 27, 2022).

Krueger, G. *et al.* (2001) "The impact of psoriasis on quality of life: results of a 1998 National Psoriasis Foundation patient-membership survey.," *Archives of dermatology*, 137(3), pp. 280–4.

Lannge, E.J. (2021) "Does removal of correlated variables affect the classification accuracy of machine learning algorithms?," *Uppsala Universitet* [Preprint].

McKinney, W. (2010) "Data Structures for Statistical Computing in Python," in, pp. 56–61. doi:10.25080/Majora-92bf1922-00a.

NHS (2022) *Psoriasis*, *https://www.nhs.uk/conditions/Psoriasis*.

NICE (2022) *Psoriasis: assessment and management*, *https://www.nice.org.uk/guidance/cg153/chapter/1-Recommendations*.

Oakley, A. (2014) *Psoriasis: Symptoms, Treatment, Images and More - DermNet*, *DermNet NZ*. Available at: https://dermnetnz.org/topics/psoriasis (Accessed: May 12, 2022).

Pathak, M. (2020) *Evaluation Metrics For Machine Learning For Data Scientists*. Available at: https://www.analyticsvidhya.com/blog/2020/10/quick-guide-to-evaluation-metrics-for-supervised-and-unsupervised-machine-learning/ (Accessed: April 21, 2022).

Patil, P. (2018) *What is Exploratory Data Analysis?* , *TowardsDataScience*. Available at: https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15 (Accessed: May 12, 2022).

Pedregosa, F. *et al.* (2011) "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, 12, pp. 2825–2830.

Primary Care Dermatology Society (2022) *Psoriasis: an overview and chronic plaque psoriasis*, *https://www.pcds.org.uk/clinical-guidance/Psoriasis-an-overview*.

R Core Team (2021) "R: A Language and Environment for Statistical Computing." Vienna, Austria. Available at: https://www.R-project.org/.

Ramachandran, R. *et al.* (2021) “Assessing the Value of Unsupervised Clustering in Predicting Persistent High Health Care Utilizers: Retrospective Analysis of Insurance Claims Data,” *JMIR Medical Informatics*, 9(11), p. e31442. doi:10.2196/31442.

van Rossum, G. and Drake, F.L. (2009) *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Stanford University (2022) *Robotics: A Brief History*, *https://cs.stanford.edu/people/eroberts/courses/soco/projects/1998-99/robotics/history.html*.

Tsang, S. *et al.* (2018) “Differential models of twin correlations in skew for body-mass index (BMI),” *PLOS ONE*, 13(3), p. e0194968. doi:10.1371/journal.pone.0194968.

Veloso, R. *et al.* (2014) “A Clustering Approach for Predicting Readmissions in Intensive Medicine,” *Procedia Technology*, 16, pp. 1307–1316. doi:10.1016/J.PROTCY.2014.10.147.

Vlachos, A. (2011) “Evaluating unsupervised learning for natural language processing tasks,” in *Proceedings of the First workshop on Unsupervised Learning in NLP*, pp. 35–42.

Weisstein, E.W. (2022) *Standard Deviation -- from Wolfram MathWorld*. Available at: https://mathworld.wolfram.com/StandardDeviation.html (Accessed: April 12, 2022).

Weisstein, E.W. (no date) *Statistical Median -- from Wolfram MathWorld*. Available at: https://mathworld.wolfram.com/StatisticalMedian.html (Accessed: April 12, 2022).

Yıldırım, S. (2021) “Data scientists without data engineering skills will face the harsh truth,” *Medium* [Preprint]. Towards Data Science. Available at: https://towardsdatascience.com/data-scientists-without-data-engineering-skills-will-face-the-harsh-truth-ff482a223ddc.