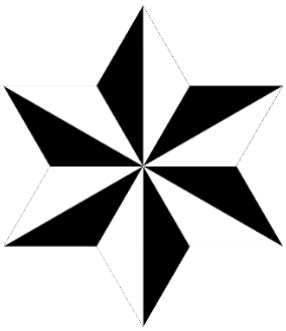
Comparison of Machine Learning algorithms for WiFi based indoor positioning



Oscar Miles

Student number: 18821327

University of Brighton

This dissertation is submitted for the degree of

Computer Science with Artificial Intelligence

Supervisor: Dr. Khuong Nguyen

Computing division 10/03/2022

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other University. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text. This dissertation contains less than 10,000 words including appendices, bibliography, footnotes, tables and equations and has less than 50 figures.

I confirm that I have a Learning Support Plan for ‘spelling and grammar and extension’ as recommended by the Disability and Dyslexia Team, and agreed by the School. I understand that the deadline for my assessment has been adjusted (as per the required School protocol) and that this, and my spelling and grammar, should be taken into consideration when my assessment is marked/ graded.

Oscar Miles

10/03/2022

Abstract

GPS signals in indoor environments are extremely poor due to the lack of line-of-sight between satellites and GPS receiver on earth. Because of this poor accuracy indoor positioning systems are implemented indoors to provide tracking and monitoring. There are many techniques that can be used to provide the location indoors, however this report focuses on the method WiFi Fingerprinting. This is where WiFi received signal strength (RSS) measurements are captured from all the surrounding Wireless Access Points (WAPs) of an area that are then stored in a database with the real-world location or building and floor number of the device. This then creates what is known as a Fingerprint of that location. This paper will look at, how Machine Learning can be applied to the datasets to determine the location of the device.

The selected regression and classification Machine Learning algorithms are compared against each other, and Machine Learning algorithms from other works in literature that were applied to the same UJIIndoorLoc database. The Machine Learning algorithms are compared using a Success Rate (Classification) and Error distance (Regression). The highest success rate and least amount of error distance found in literature were 98.32% using K-Nearest Neighbors (ryanmclark, 2021), 6.20m using K-Nearest Neighbors (Moreira et al., 2015) respectively. This paper produces a Support Vector Machine scoring a success rate of 94.50% and a K-Nearest Neighbors with an error distance of 7.21m.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

The Global Positioning System (GPS) is extremely accurate in an outdoor environment due to Line-of-Sight between the satellites in space, the receivers on earth and a mobile device. However, when a mobile device moves into an indoor environment the accuracy falls considerably. Since most people have mobile devices and perform many activities indoors, significant research has been carried out to improve the accuracy of Indoor Positioning Systems (IPS), using technologies that are found in modern smartphones. 'IPS based on WiFi received signal strength (RSS) reading are popular' (Rojo et al., 2019), this is due to the low-cost of the applications as the WiFi infrastructure has usually already been deployed.

IPS systems can use many techniques and this project focuses on one of those techniques called WiFi Fingerprinting. WiFi Fingerprinting is broken down into two phases, off-line phase (training dataset) and the on-line phase (testing dataset). After the two phases have been carried out, a database is formed to which either a regression or a classification Machine Learning algorithm can be applied. In this report, selected regression and classification Machine Learning algorithms are compared against each other, and other algorithms from works that were applied to the same UJIIndoorLoc database.

1.1 Project motivation

To find a Machine Learning algorithm within the Scikit-learn python package capable of locating a mobile device in a building, using a dataset built up of Received Signal Strength (RSS) values relating to the location of the mobile device labelled with either longitude and latitude or the building and floor number, with an accuracy of below 5m.

1.2 Project objectives

- Following the agile project process.
- Display how Machine Learning algorithms can be used for indoor positioning systems.
- Consider the performance of other Machine Learning algorithms from literature.
- Comparing the performance of various Machine Learning algorithms on a competition-graded indoor positioning dataset.

1.3 Limitations of scope

In the field of indoor positioning systems, there are an abundance of techniques for calculating the location of a user, such as, trilateration, time-of-flight, angle-of-arrival and WiFi fingerprinting. This report solely focuses on WiFi fingerprinting.

None of the Machine Learning algorithms used for this report were created from scratch as they were all imported from Scikit-Learn, however they will be compared against existing algorithms on a challenging competition dataset.

1.4 Dissertation outline

Chapter 2 will outline the indoor positioning problem, the challenges of indoor positioning and explain what WiFi fingerprinting is. Chapter 3 will focus on the works of other people in literature and the results they were able to obtain. Chapter 4 describes the development process, what tools were used and how problems were tackled, and lastly explaining the machine learning algorithms that I have used. Chapter 5 looks at the dataset I trained and tested the machine learning algorithms on and the results I was able to obtain with each of the machine learning algorithms. Chapter 6 explores the steps I took to produce this project, the testing that I carried out through the course of the project and the risk management. Chapter 7 explains the social, legal and ethical issues related to the project. Chapter 8 concludes the project, followed by a personal reflection of the project and future work.

Chapter 2 Indoor positioning problem

2.1 The challenge of indoor positioning

In outdoor environments devices are easy to track with the use of the Global Positioning System (GPS), which is a satellite-based radionavigation system where the position of a device is calculated by using two mathematical concepts, trilateration and the relationship between distance travelled, rate of travel and amount of time spent traveling. GPS-enabled smartphones are typically accurate to within 10m and with a dual-frequency receiver these systems can perform real-time positioning within a few meters, this is due to the signal not being interfered with on its journey from the satellite to the device, this is referred to as line-of-sight between the satellite and the device. However, problems with GPS positioning start to arise once the device is moved to an indoor environment, as the signals sent out are now being interfered with by the confines of the indoor space. (National Coordination Office for Space-Based Positioning, Navigation and Timing, 2019)

Indoor positioning systems were introduced as a way of locating devices in indoor environments, though these systems can be affected by attenuation. Attenuation is the reduction or loss of signal strength during transmission, which can be caused by the following processes, absorption, reflection, diffraction, scattering and interference.

Absorption is where an electromagnetic wave is absorbed by a material or object, once absorbed the energy of the wave is transferred to the particles in the material or object. Materials or objects with higher densities act as better absorbers of waves. (Sun, 2018)

Reflection is where an electromagnetic wave strikes a surface, here some of the energy of the wave is absorbed into the surface and the rest bounces off the surface. Given a smooth surface, the outgoing energy will all move in the same direction. Given a rough surface, the outgoing energy will scatter in many directions. (Sun, 2018)

Diffraction is where an electromagnetic wave encounters an object, this causes the wave to split into secondary waves. The secondary waves continue to propagate in the direction in which they were split. Since the original wave was split into multiple waves, the resulting signal will be significantly lower than expected. (Sun, 2018)

Scattering is where an electromagnetic wave encounters an uneven or rough surface, thus causing the main wave to dissipate into multiple reflected waves. In terms of WiFi signal this will cause the reflected signals to interfere with each other resulting in a weaker signal. Scattering is caused by surfaces such as, plant leaves, dust or smoke. (Sun, 2018)

Interference is where an electromagnetic wave is not able to freely pass an electronic device on its way from the access point to the device, this happens when the electronic device is operating on the same frequency as the access point. (Sun, 2018)

2.2 WiFi fingerprinting

WiFi fingerprinting is the mapping of an area by WiFi fingerprints. A WiFi fingerprint is a collection of Received Signal Strength (RSS) values recorded from several Wireless Access Points (WAPs) that relate to a specific location (longitude and latitude or building and floor) in an isolated environment. The process of collecting WiFi fingerprints can be split into two phases, an off-line, and an on-line phase.

In the off-line phase ‘a radio map of the area where the users should be detected is constructed.’ (Niu et al, 2015) In the on-line phase ‘a user obtains the signal strength of all visible access points of the WLAN that can be detected from his/her position and creates a test sample. This sample is sent to the server to be compared with the training samples of the radio map.’ (Niu et al, 2015) Essentially, the RSS values obtained by the user in the on-line phase are compared to all the entries in the database generated in the off-line phase, the entry with the most similar RSS values provides the location of the user.

After the off-line and on-line phases are complete, you are left with a training and testing set of datapoints, to which you can apply both regression and classification machine learning algorithms to determine a location using the RSS values recorded. The regression algorithms

will calculate the longitude and latitude of a fingerprint, whereas the classification algorithms will calculate the building and floor of a fingerprint.

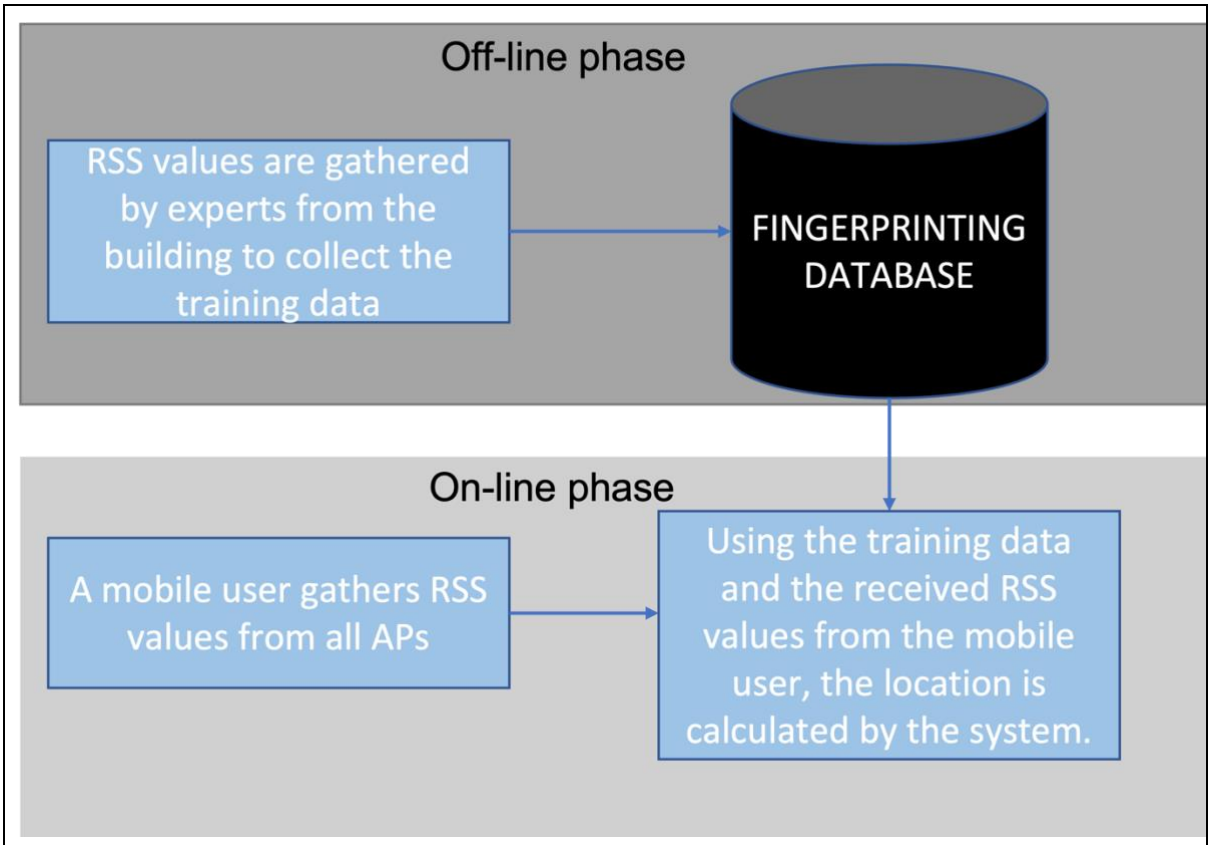


Figure 2. 1 – Off-line and On-line phases of WiFi fingerprinting

2.3 Real world applications

Indoor positioning systems can be split into two categories, tracking and navigation. The systems in the tracking category, must be able to 'detect and pinpoint the location of the person or object.' (Nguyen et al., 2021) Whereas the navigation systems 'must guide the users using the most optimal route'. (Nguyen et al., 2021) Table 2.1 below compares a few applications of indoor positioning systems in real world situations.

Table 2. 1 - The real-world applications of Indoor positioning systems

Tracking	Navigation
Security	
For the guards: • If a valuable object has been moved the guards will be notified.	For the guards: • The guards will receive the real-time location of the object and the directions to it.
For the clients: • Automatically granting access for authorised personnel.	For the clients: • Leading personnel to the nearest exit during an emergency.
Healthcare	
For the medical personnel: • Locating medical equipment.	For the medical personnel: • Finding the shortest route to the patient in an emergency.
For the patients: • Locating the nearest wheelchair. • Automatic checking-in/out upon entering/exiting the hospital.	For the patients: • Automatic wheelchair navigation in the hospital. • Route-finding to the doctor's office.
Industry	
For the administrators: • Locating products in the warehouse. • Notification when a product leaves the warehouse. • Locating warehouse machinery	For the administrators: • Automatically adjusting the speed of a conveyer belt when transporting heavy products. • Automatically sorting products based on weight and shape.
For the customers: • Tracking luggage at the airport.	For the customers: • Finding seats in large venues. • Computer guided tours.

Chapter 3 Literature research

(Ryanmclark, 2021) provides 4 Machine Learning algorithms which were also taken from the Scikit-learn python package.

(Moreira et al., 2015) provides 4 completely different Machine Learning algorithms all built from scratch, where RTLS@UM is a K-Nearest Neighbors algorithm. The rest of the algorithms presented in the paper are not described any further than the name of the team that created them during the EvAAL/IPIN 2015 competition.

(Gan et al., 2019) presents their own Machine Learning algorithm from scratch AFARLS, which is created solely for indoor localisation purposes. The algorithm combines, ELM (Extreme Learning Machine), SRC (Sparse-representation-based classification) and KNN (K-Nearest Neighbors).

(Uddin and Islam, 2015) provide 3 of their own Machine Learning algorithms built from scratch, a Support Vector Machine, K-Nearest Neighbors, and Extra-trees algorithm.

(Kim, lee and Huang, 2018) provide their own Deep Neural Network built from scratch.

(Torres-Sospedra et al., 2014) presents a K-Nearest Neighbors algorithm as a baseline for testing Machine Learning algorithms on their dataset.

(Bozkurt et al., 2015) provided their own K-Nearest Neighbors algorithm built from scratch.

(Akram, Akbar and Safiq, 2018) provide their own Machine Learning algorithm, where the algorithm is comprised of Gaussian Mixture Model-based soft clustering and Random Decision Forest ensembles.

(Wietrzykowski, Nowicki and Skrzypczyński, 2017) provide their own Machine Learning algorithm based on the structure of a FAB-MAP visual place recognition system, where it has been adapted for the WiFi fingerprinting problem.

Table 3. 1 - Other works in literatures success rate and distance error.

Source	*Algorithm*	*Success rate (%)*	*Error (m)*
Ryanmclark, 2021	K-Nearest Neighbors	98.32	8.59
Moreira et al., 2015	HFTS	98.13	8.49
Gan et al., 2019	AFARLS	97.71	6.40
Moreira et al., 2015	RTLS@UM	96.87	6.20
Ryanmclark, 2021	Random Forest	96.65	10.57
Moreira et al., 2015	MOSAIC	96.26	11.64
Uddin and Islam, 2015	Extra-Trees	95.72	10.12
Kim, lee and Huang, 2018	Deep Neural Network	95.55	9.29
Ryanmclark, 2021	Decision Tree	94.21	13.88
Moreira et al., 2015	ICSL	93.47	7.67
Uddin and Islam, 2015	Support Vector Machine	91.53	11.28
Torres-Sospedra et al., 2014	K-Nearest Neighbors	89.92	7.90
Uddin and Islam, 2015	K-Nearest Neighbors	86.23	13.43
Bozkurt et al., 2015	K-Nearest Neighbors	85.00	-
Akram, Akbar and Shafiq, 2018	HybLoc	85.00	6.46
Wietrzykowski, Nowicki and Skrzypczyński, 2017	Visual place recognition	78.00	8.21
Ryanmclark, 2021	Support Vector Machine	34.43	46.61

The success rate of the models is calculated by the percentage of testing fingerprints whose building and floor are correctly predicted, whereas the error corresponds to the real-world distance between the actual position and the predicted position. The above table lists all the classification and regression algorithms I could find that were trained and tested on the same competition graded dataset and used the same performance metrics for the results of their models.

Chapter 4 Methodology

4.1 Development tool

The development tools in this project were used to create a parameter tuning script, training and testing of the algorithms, recording the algorithms performance metrics during testing, and graphs. This was done so with the following python modules, Scikit-learn, Matplotlib, Pandas and NumPy.

```
In [28]: def cdf_plot(errors, mode):
             colours = ['blue', 'red', 'green', 'purple']
             fig, axs = plt.subplots(2, 2, constrained_layout=True, figsize=(13, 7))
             fig.suptitle(f'{mode} CDF Plot', fontsize=20)
             fig.text(0.5, -0.05, 'Distance Error (m)', ha='center', fontsize=18)
             fig.text(-0.05, 0.5, 'Probability (%)', va='center', rotation='vertical', fontsize=18)

             for graph in range(2):
                 for n in range(2):
                     if graph == 0:
                         c = n
                     else:
                         c = n + 2
                     count, bins_count = np.histogram(errors[c][0][mode], bins=100)
                     pdf = count / sum(count)
                     cdf = np.cumsum(pdf)
                     axs[graph][n].plot(bins_count[1:], cdf, color=colours[c], label=errors[c][1])
                     axs[graph][n].grid()
                     axs[graph][n].legend()
```

Figure 4. 1 – Example of the Jupyter Notebook workspace

Throughout the course of the project, I used two IDEs with the main one being Jupyter Notebook. I was able to create the graphs and tables shown in the report using Matplotlib, Pandas and NumPy. The training, testing and recording of the performance metrics was also done using Jupyter Notebook but using Scikit-learn, Pandas and NumPy. Finally, the parameter tuning script was developed on the PyCharm IDE, using Scikit-learn, Pandas and NumPy.

```python
def finding_params(model, grid, opt):
    scoring = ('accuracy' if opt == 'FLOOR' or opt == 'BUILDINGID' else mse)
    gscv = GridSearchCV(estimator=model,
                        param_grid=grid,
                        cv=5,
                        scoring=scoring,
                        verbose=3)
    gscv = gscv.fit(x_train_pca, y_train[opt])
    best_score = gscv.best_score_
    best_params = gscv.best_params_
    best_params['score'] = best_score

    return pd.DataFrame(best_params, index=[opt])


def model_details(models, location):
    t0 = t()
    floor = finding_params(models[0], parameter_grid[location]['params'], 'FLOOR')
    t1 = t()
    building = finding_params(models[0], parameter_grid[location]['params'], 'BUILDINGID')
    t2 = t()
    longitude = finding_params(models[1], parameter_grid[location]['params'], 'LONGITUDE')
    t3 = t()
    latitude = finding_params(models[1], parameter_grid[location]['params'], 'LATITUDE')
    t4 = t()
    times = [f'{(t1 - t0):.2f}', f'{t2 - t1}:.2f', f'{t3 - t2}:.2f', f'{t4 - t3}:.2f']
    details = pd.concat([floor, building, longitude, latitude])
    details['time'] = times
    return details
```

Figure 4. 2 – Example of the PyCharm IDE

Lastly, for managing my project I used Trello, which is an online project management tool where I was able track what needed to be done each week. The tracking of my progress was made easier using cards, as they gave an outline of the task that needed to be carried out.

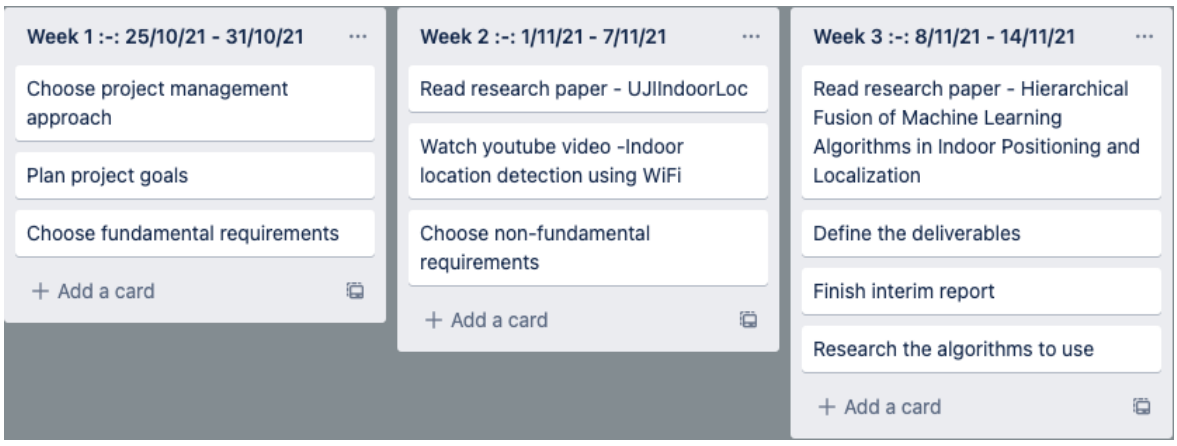


Figure 4. 3 – Example of the Trello Board

4.2 Development approach

This project uses the agile development approach, with a focus of creating features rapidly that satisfy the needs of the project. Agile development has a sprint phase, this is a short period of time where a set of features are designed, developed, tested and evaluated. For this project there was a sprint phase each week where, at the beginning of the week I would go over the work I created in the previous week, after which I would plan an idea for what feature needed to be created next. This was critical for me in the development of the project, as it gave me feedback on the features I had developed, so that I could change features that did not work or scrap features if they were not needed anymore.

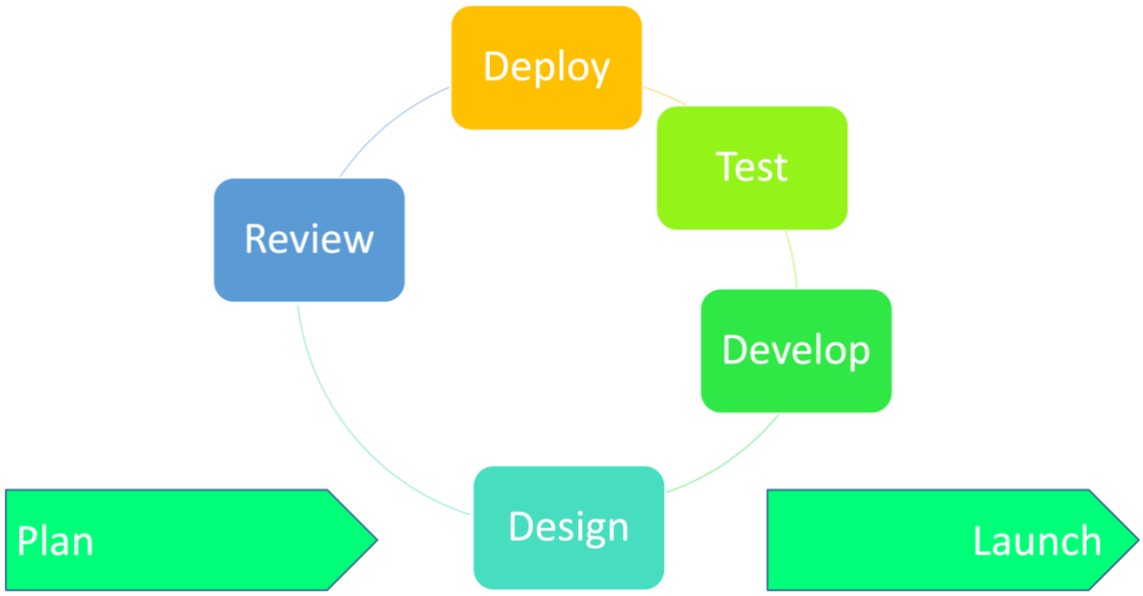


Figure 4. 4 – Agile Development Process

4.3 Machine learning algorithms

This project uses 4 supervised machine learning models. K-Nearest Neighbors, Support Vector Machines, Decision Trees, and finally Multi-Layered Perceptron. Each of these algorithms have customisable models for both regression and classification problems, because of this, this project uses the regression and classification models from each algorithm to determine the location of a mobile device.

'K-Nearest Neighbors uses feature similarity to predict the values of any new data points.' (Analytics Vidhya, 2019) With the use of a K value, we can specify how many of closest data points to incorporate into the algorithm's choice. Looking at figure 4.3, we have a green dot, that needs to be placed into either the blue or red category.

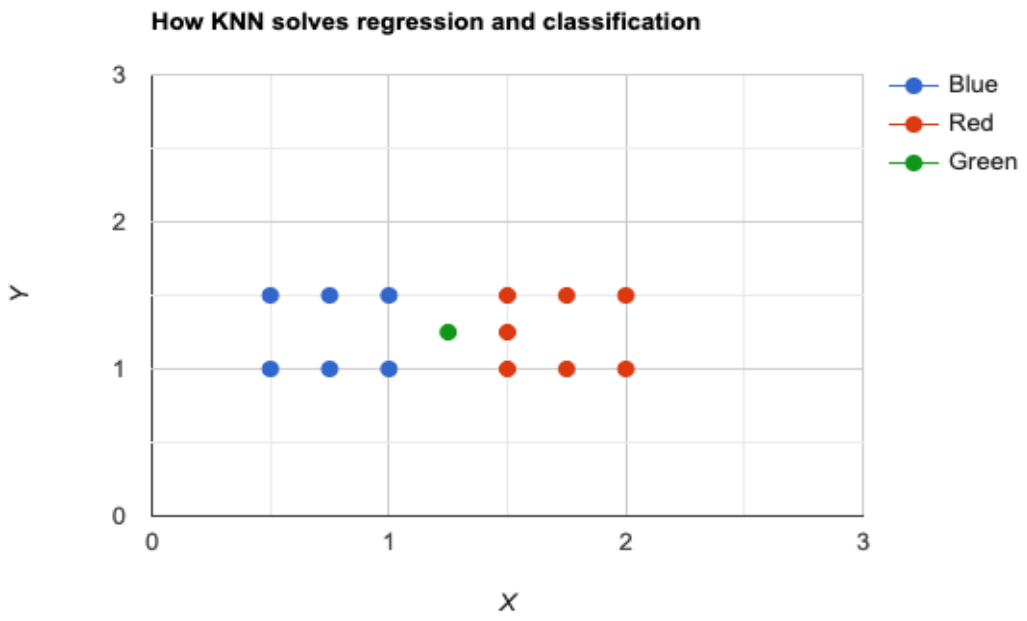


Figure 4. 5 - Given K = 5, KNN will calculate the Euclidean distance between the green point and all the other data points. Then the closest 5 data points are selected, for classification KNN provides a mode average of the 5 data points for selecting the class of the data point, here it would choose the class of red, whereas, in regression KNN provides a mean average of the 5 data points, giving you the value of the green data point.

'Support Vector Machines are based on the idea of finding a hyperplane that best divides a dataset into two classes.' (Bambrick, 2016) Where a hyperplane is a line that separates and classifies a set of data. 'Support vectors are data points nearest to the hyperplane' (Bambrick, 2016) if these data points are removed, the position of the hyperplane would be altered. This gives the support vectors critical significance.

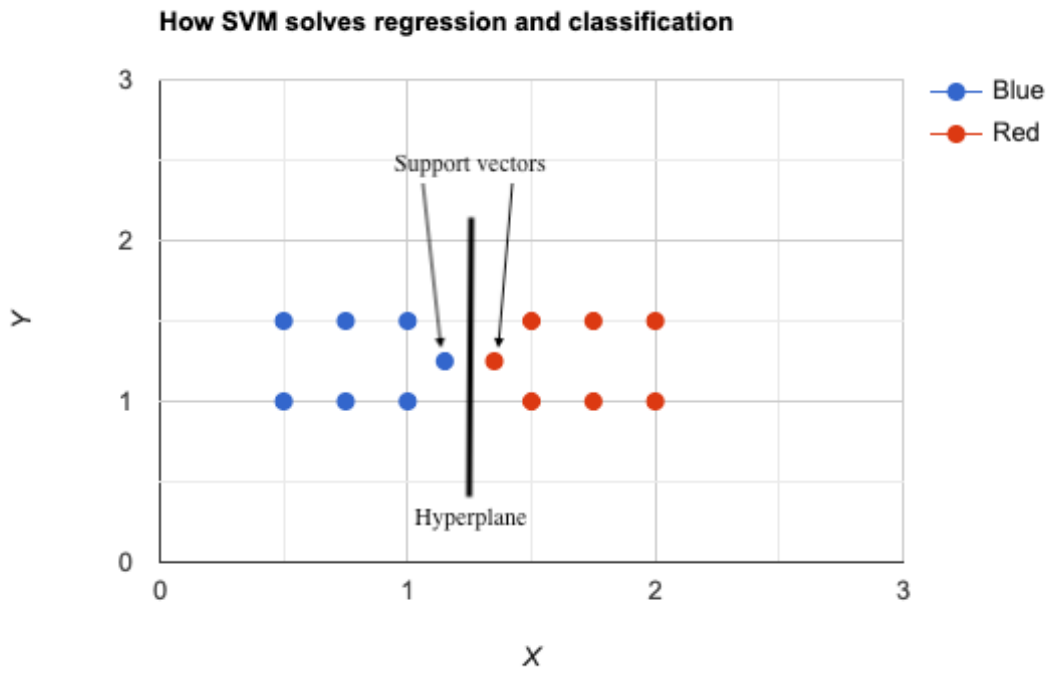


Figure 4. 6 - Shows how the SVM classifier would make its predictions, any new data point that falls to the left of the hyperplane, will be labelled as blue and any data point that falls to the right of the hyperplane, will be labelled as red.

'Decision trees are a form of predictive modelling, helping to map the different decision or solutions to a given outcome. Decision tree are made up of different nodes. The root node is the start of the decision tree, which is usually the whole dataset. Leaf nodes are the endpoint of a branch, or the final output of a series of decisions. The decision tree won't branch any further from a leaf node. … the features of the data are internal nodes, and the outcome is the leaf node.' (Seldon, 2021) When solving classification problems with decision trees 'the leaves or endpoint of the branches … are the class labels.' (Seldon, 2021) Whereas when solving regression problems with decision trees, the 'tree will create dense or spares clusters of data, to which new and unseen data points can be applied.' (Seldon, 2021)

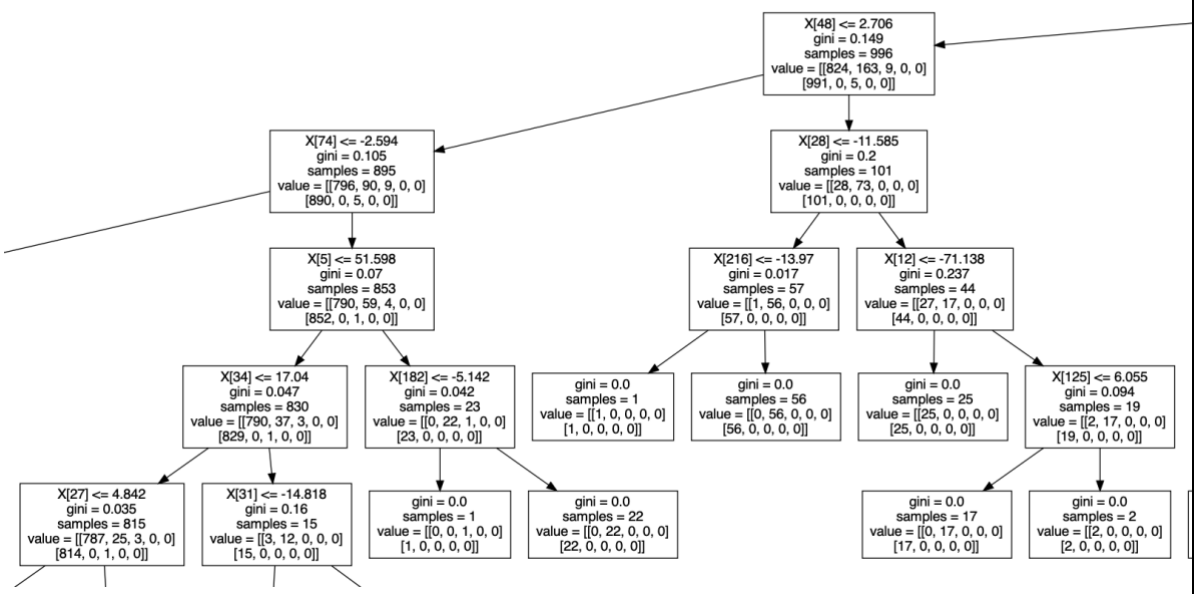
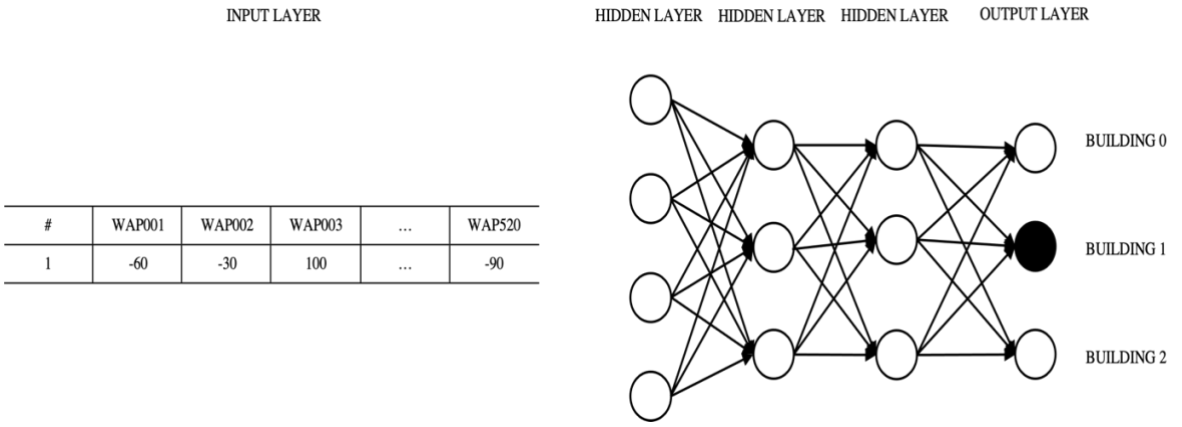


Figure 4. 7 - Shows how the Decision Tree used in this project broke down its choices

Multi-layer Perceptron (MLP) contain an input layer, multiple hidden layers and an output layer. Each node in each layer of the MLP have a set of weights and biases. The MLP works from the input layer towards the output layer, making it a feed-forward network. In the MLP 'the inputs are pushed forward through the MLP by taking the dot product of the input with the weights that exist between the input layer and the hidden layer. This dot product yields a value at the hidden layer.' (Deep AI, 2019) The dot product value is then passed through an activation function, ReLu for this project. After which the value is then pushed onto 'the next layer in the MLP by taking the dot product with the corresponding weights.' (Deep AI, 2019) This process is repeated until the value reaches the output layer. For training purposes, the final calculation will be used for backpropagation. Backpropagation is used by the MLP during the training phase to adjust the weights that are used in the dot product calculation. Finally for testing purposes, the final calculation will be used to make a prediction.



#	WAP001	WAP002	WAP003	...	WAP520
1	-60	-30	100	...	-90

Figure 4. 8 - Contains the architecture of a simple MLP where the input layer are the RSS values. There are 3 hidden layers and then in the final output layer the choice of the algorithm is provided.

Chapter 5 Experimental results

5.1 Dataset

The databased used for this project contains 21048 records, where 19937 are training records used for the training of the algorithms and the last 1111 are testing records for the testing of the algorithms.

Table 5. 1 - Dataset Description

Feature	Label	Range
1-520	WAP RSS level	-104 to 0 (+100 if not found)
521	Longitude	-7695.938754929929 to -7299.786516730872
522	Latitude	4864745.745015971 to 4865017.364684202
523	Floor	0 to 4
524	Building ID	0 to 2
525	Space ID	Categorical integer values (office, labs, etc)
526	Relative Position	0 inside the room 1 outside the room
527	User ID	0 to 18
528	Phone ID	0 to 24
529	Timestamp	UNIX time format

Each record in the database contains 529 numerical values. The first 520 are the RSS values received by the device for each of the 520 WAPs in the dataset, the RSS values range from -104 to 0, where -104dBm is a very weak signal and 0 is a very strong signal, if they have the value of 100 this means the WAP was not detected in the users reading. 521 and 522 are the longitude and latitude of the device, these values range from - 7695.938754929929 to -7299.786516730872 and from 4864745.745015971 to 4865017.364684202 respectively. 523 and 524 are the floor and building of the device, these values range from 0 to 4 and 0 to 2

respectively. 525 is the space ID of the device this describes the type of room. 526 is the relative position of the device, 0 inside the room, 1 outside the room. 527 is the user ID this value ranges from 0 to 18. 528 phone ID this value ranges from 0 to 24. 529 is the timestamp this value is in UNIX time format.

The columns, Space ID, Relative Position, User ID, Phone ID and Timestamp, were dropped from the dataset as they provided no relevance to the machine learning algorithms in determining either the longitude and latitude or the floor and building. After dropping the columns, the RSS values needed fixing since the majority of the WAPs for each record were not able to reach the device so their reading was instead replaced with an artificial value of +100dBm, this was changed to the value of -150dBm to indicate that the reading is clearly outside of the valid readings -104 to 0dBm.

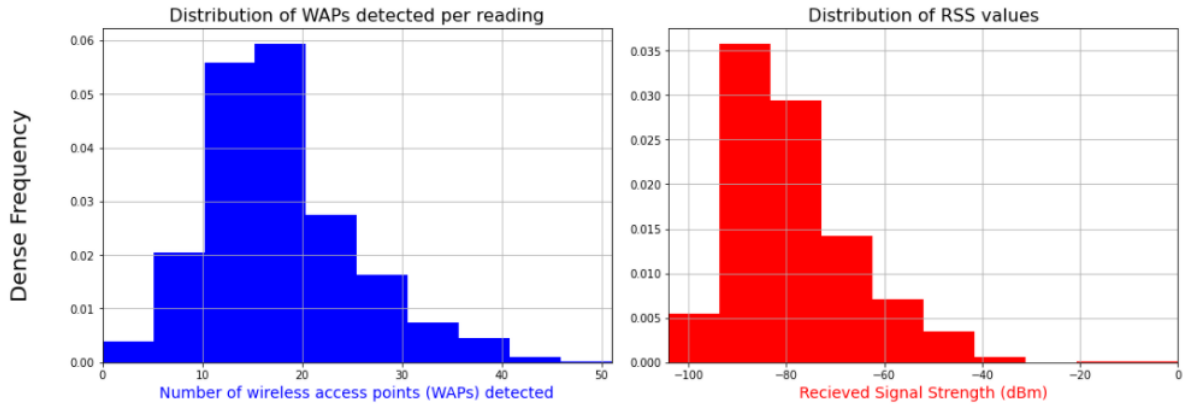


Figure 5. 1 - The distribution of WAPs and RSS values in the dataset. For the number of WAPs it is better to have a higher number rather than a lower number. For the RSS values detected it is better to have a value closer to 0 rather than -100.

From the graph on the left-hand side, Distribution of WAPs detected per reading, you can see that for most of the time there are between 15 to 20 WAPs detected by the mobile device. This tells us that over the whole dataset most records will contain between 10 to 20 valid WAPs readings. Along with this, the graph on the right-hand side, Distribution of RSS values, details the spread of the valid RSS values. As you can see most of the valid RSS values fall into the weaker signal strength bracket with the most values being recorded between -90 to -80dBm. This tells us that there is not very good WiFi coverage over the area where the dataset was created, however, these values can still be very useful for WiFi Fingerprinting as the values

that fall outside of the -104 to 0dBm range can still be used to create a fingerprint for that location.

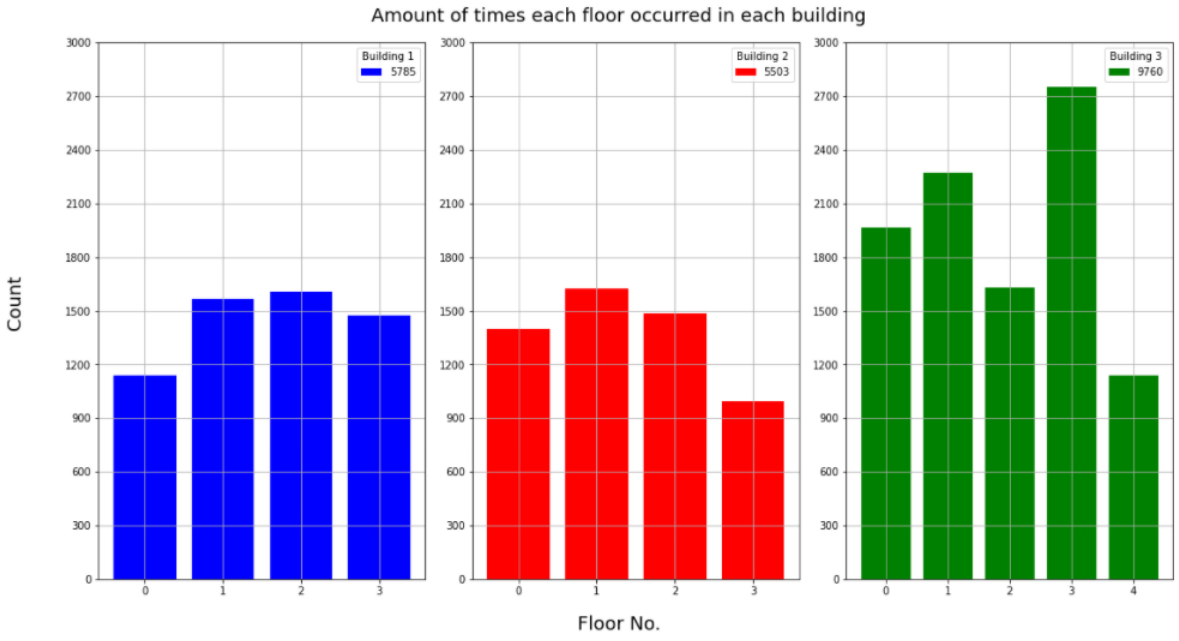


Figure 5. 2 - The number of times each floor occurred in each building

Above shows the spread of the number of occurrences of each floor in each building, from the graphic you can see most of the records for the dataset are recorded in Building 3, the significance of this is that Building 3 contains nearly double the amount of data points that Building 1 and Building 2 have when compared individually.

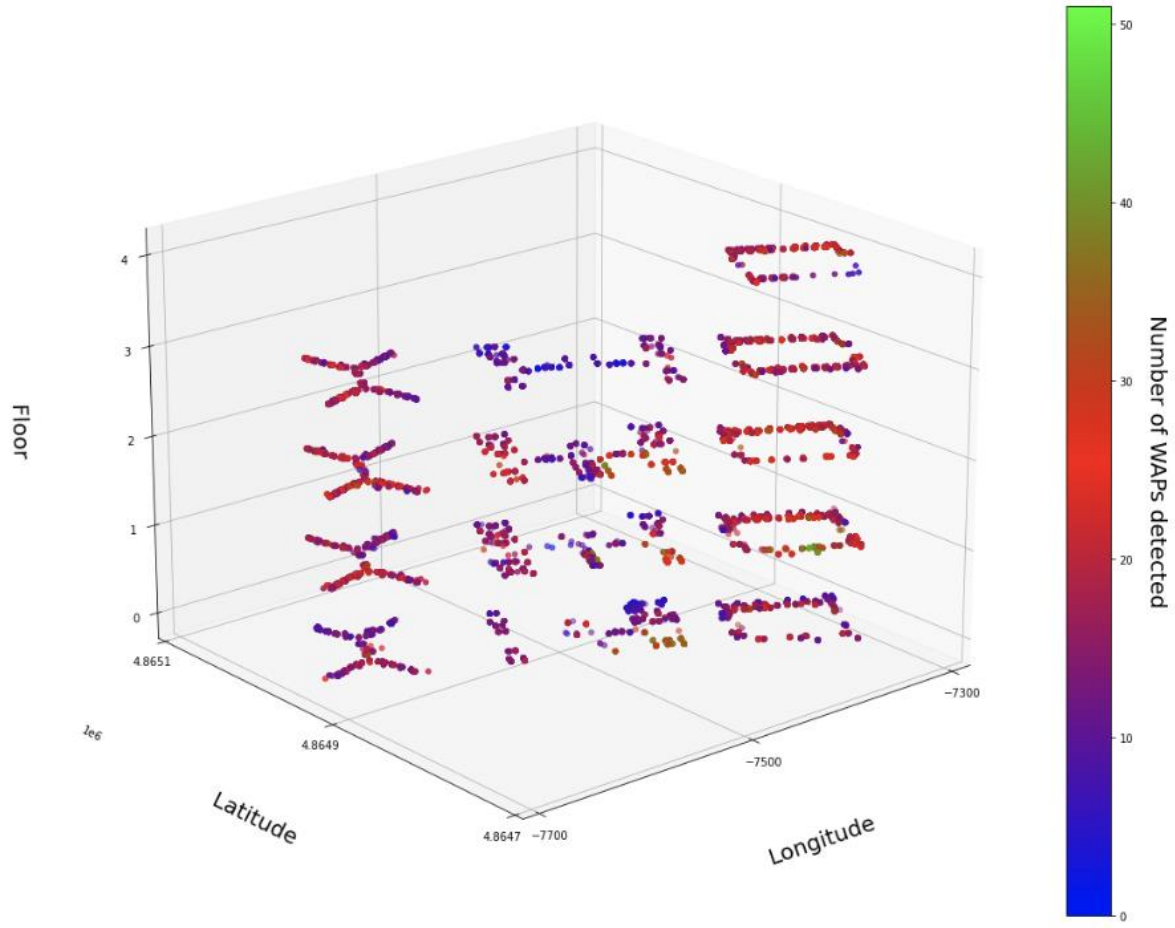


Figure 5. 3 - 3D visual displaying the number of WAPs detected over the whole area that the dataset covered.

From the 3D scatter plot you can see the exact layout of the area being mapped by the dataset. The heatmap indicates the number of WAPs detected for each location in the dataset. The scatter plot shows us that the best WiFi coverage is in building 3, the far right, as it contains the most nodes with colors between red and green indicating a high number of WAPs detected and that the lowest coverage is in building 2 with the greatest number of blue nodes. This does make sense though as building 2 has a layout that seems to have many walls which will attenuate the RSS and buildings 1 and 3 have very linear layouts with only straight corridors, thus reducing the attenuation.

The high dimensionality of the dataset is a major drawback as it greatly increases the training and testing time of the machine learning algorithms. To counter act this problem I imposed a

dimensionality reduction technique called Principal Component Analysis (PCA). PCA is used to reduce the dimensionality of large datasets by transforming a large set of variables into a smaller one that still contains most of the information in the large set. The amount of information retained is shown by a value of explained variance, with an explained variance of 1.0 (100%) you will have all the principal components.

Table 5. 2 - Explained Variance

Component	Variance (%)	Combined Variance (%)
0	14.657091	14.657091
1	5.479158	20.136249
2	4.952448	25.088697
3	4.793480	29.882177
4	3.916326	33.798503

From the above figure you can see that the highest significance (variance) is assigned to the first component 0 with a 14.7% variance. Now if we were to use the dataset with just 1 principal component we would be training and testing the machine learning algorithms with around 15% of the available data. This will drastically improve the training and testing speeds but will hinder the accuracy of the algorithms massively. In the right-hand column ‘Frequency Sum’ you can see that even after 5 components the total variance kept is only 33.8%, this is not enough to keep a good trade-off between reducing training and testing speeds whilst also not massively reducing the accuracy of the machine learning algorithms. To show when you start to reach diminishing returns for your total variance kept, you create a plot of the individual variance of each point against the cumulative variance as shown below in figure 5.4.

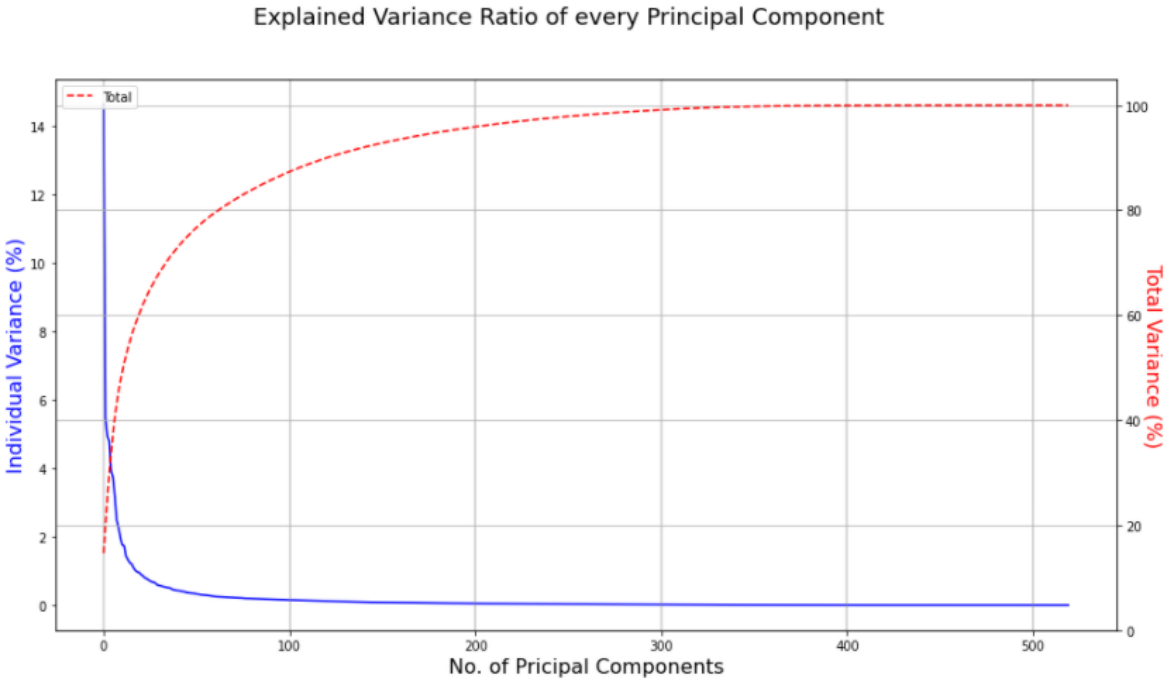


Figure 5. 4 - Explained Variance Ratio

The graph above gives you an excellent understanding of how many principal components you need before you hit the diminishing returns. For this project I kept a variance of 95% or 200 principal components, my reasoning for choosing this value was after looking at the graph I saw the apex of the curve levelling out near the 200 principal component mark. For me this represents when the diminishing returns phase has begun.

5.2 Results

This project contains 4 separate machine learning algorithms each capable of solving regression and classification problem, the performance metrics are detailed in table 5.3 and 5.5 respectively. Table 5.4 is a confusion matrix which is used to calculate the performance metrics of table 5.5

Table 5. 3 - Regression Performance Metrics

Metric	Equation
R Squared (R^2)	$R^2 = 1 - \frac{\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}{\sum_{j=1}^{n}(y_j - \underline{y})^2}$
Mean Absolute Error (MAE)	$MAE = \frac{1}{n}\sum_{j=1}^{n}\lvert y_j - \hat{y}_j \rvert$
Mean Square Error (MSE)	$MSE = \frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2$

R Squared is the percentage of dependent variable variation that a linear model explains.

$$R^2 = \frac{\text{Variance explained by the model}}{\text{Total Variance}}$$

When $R^2 = 0$, the model does not explain any of the variations in the response variable around its mean, whereas when $R^2 = 100$, the model explains all the variations in the response variable around its mean. It can be said that the larger the R^2 the better the regression model fits the observations. (Dasgupta, 2020)

Mean Absolute Error is the average of all absolute errors. Where the absolute error is, is the difference between the predicted value and the actual value. When MAE = 5, this implies that on average, the difference between the predicted value and the actual value will be 5, e.g., actual value = 100, predicted value = 95 or 105.

Mean Square Error tells you how close a regression line is to a set of points. This is done by taking the distances from the points to the regression line, where the distances are the errors, and squaring them. With the application of squaring the values it removes any negative signs and applies more weight to larger differences. A model with a low MSE has accurate predictions. (StatisticsHowTo, 2022)

Table 5. 4 - Confusion Matrix

		Predicted	
		Positives	Negatives
Actual	Positives	True Positives (TP)	False Negatives (FN)
	Negatives	False Positives (FP)	True Negatives (TN)

A confusion matrix is a table used to describe the performance of a classification model on a set of test data for which the true values are known, this is known as supervised machine learning. 'TP' are the cases where the algorithm predicted true and the actual result was true (correct prediction). 'TN' is when the algorithm predicted false and the actual result was false (correct prediction). 'FP' detail when the algorithm predicted true but the actual result was false (wrong prediction). Lastly, 'FN' is when the algorithm predicted false and the actual result was true (wrong prediction). (Rawat, 2019)

Table 5. 5 - Classification Performance Metrics

Metric	Equation
Precision	$P = \frac{TP}{TP + FP}$
Accuracy	$A = \frac{TN + TP}{TP + TN + FP + FN}$
Recall	$R = \frac{TP}{TP + FN}$
F1 Score	$F1 = \frac{P + R}{2}$

'Precision refers to how close the model's predictions are to the observed values. The more precise the model, the closer the data points are to the predictions.' (Frost, 2017) This explanation can be simplified to 'precision measures the extent of error caused by False Negatives' (LT, 2022)

$$P = \frac{True\ Positives}{Total\ number\ of positive\ predictions}$$

Accuracy can be defined as the number of classifications a model correctly predicts divided by the total number of predictions made. (Bressler, 2021)

$$A = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

'Recall is a measure of how many relevant elements were detected. Therefore, it divides true positives by the number of relevant elements.' (Huellman, 2022) Recall can also be simplified to 'recall measures the extent of error caused by False Negatives' (LT, 2022)

$$R = \frac{True\ Positives}{Total\ number\ of\ actual\ positives}$$

'F1 Score is an average of Precision and Recall, this means F1 gives an equal weight to Precision and Recall. Thus, a model will obtain a high F1 score if both Precision and Recall are high, a model will obtain a low F1 score if both Precision and Recall are low, and finally a model will obtain a medium F1 score if either Precision or Recall are low, and the other is high.' (Korstanje, 2021)

$$F1 = \frac{Precision + Recall}{2}$$

The performance of a machine learning algorithm is significantly affected by the hyperparameter chosen. Thus, a hyperparameter tuning method needs to be employed. For this project the GridSearchCV hyperparameter tuning function was used, this is provided by the Scikit-learn python module. For a grid search to work a dictionary needs to be defined containing the list of parameters you want to apply to your algorithm. 'GridSearchCV tries all the combinations of the values passed in the dictionary and evaluates the algorithm for each combination using the Cross-Validation method.' (Mujtaba, 2020) For the machine learning algorithms chosen in this project the following hyperparameters were applied, where a parameter is not mentioned here it means the original Scikit-learn parameter was used. KNN classifier and regressor have a K value of 3, SVM classifier and regressor have a C value of 50, DT classifier has a max_depth of 15 whilst the regressor has a max_depth of 35, lastly the MLP classifier has hidden_layer_sizes of (32, 32, 16) and the regressor has hidden_layer_sizes of (32, 32, 32, 100).

The results for this project come in 3 tables, 5.4 for the regression algorithms as they were able to find the longitude and latitude at the same time. 5.5 and 5.6 are the results achieved by the classification algorithms for solving the building and floor, respectively. Visual aids are provided for the regression algorithms performances in figure 5.7 and for the classification algorithms in figure 5.8

Table 5. 6 - Regression Results

Algorithm	R Squared	Mean Absolute Error (m)	Mean Squared Error (m^2)
KNN	1.0	7.21	182.75
SVM	1.0	11.15	240.56
DT	1.0	9.00	262.08
MLP	0.99	95233.92	3.298348e+10

Table 5.6 tells us that the KNN regressor, was able to achieve the least amount of error when determining the location of a mobile device. KNN was closely followed by SVM which was able to produce results with small outliers than the DT regressor was able to achieve, this is shown from the Mean Squared Error of SVM being lower than DT whilst the Mean Absolute Error is higher. Lastly it shows that the MLP regressor was a poor choice for the regression problem as it was unable to attain usable results.

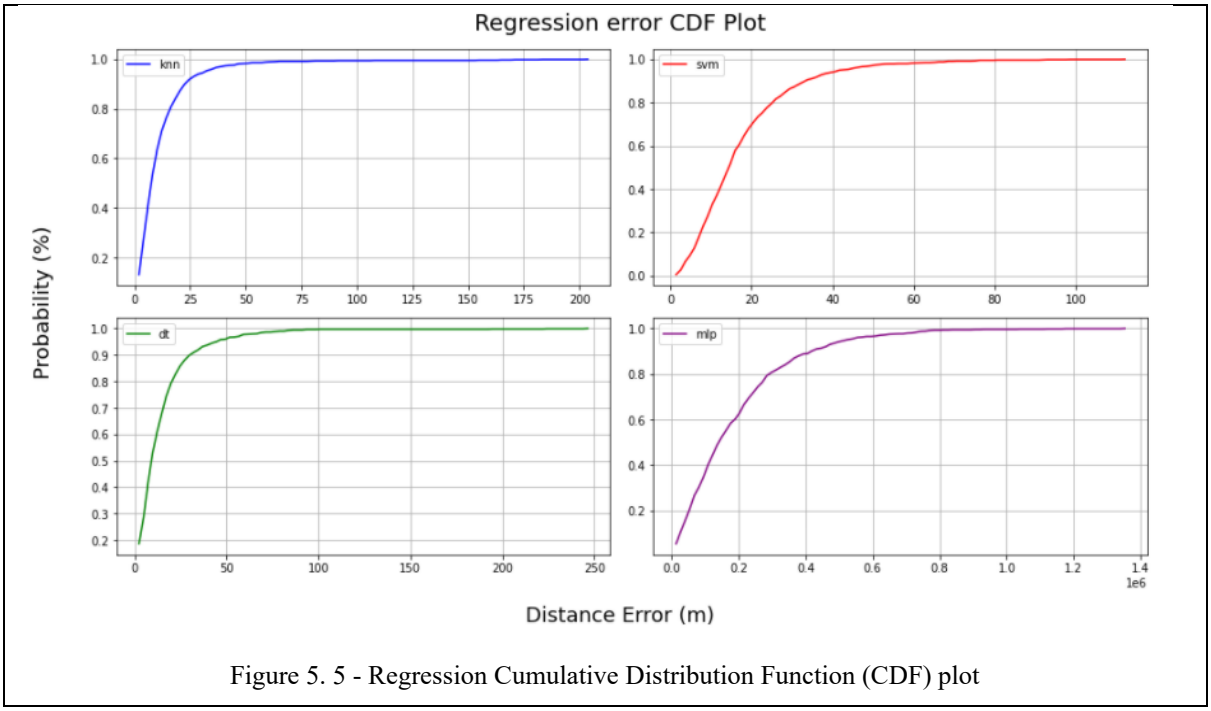


Figure 5. 5 - Regression Cumulative Distribution Function (CDF) plot

The above CDF plot shows the percentiles of the distance error produced by the regression algorithms. For example, looking at KNN you can see that there will be a 25m distance error

95% of the time. Now if you compare the values in table 5.4 with the visual representation displayed in figure 5.5, you can establish that SVM is the best performing algorithm for regression. This is shown from SVM having scored in the middle of the pack for the performance metrics and with the results shown in figure 5.5 the total error is by far the least on the SVM y scale.

Table 5. 7 - Building Results

Algorithm	Precision (%)	Accuracy (%)	Recall (%)	F1 Score (%)
KNN	99	99	99	99
SVM	100	100	100	100
DT	98	99	99	98
MLP	100	100	100	100

Table 5. 8 - Floor Results

Algorithm	Precision (%)	Accuracy (%)	Recall (%)	F1 Score (%)
KNN	85	83	83	82
SVM	88	89	88	88
DT	81	76	82	77
MLP	87	84	87	85

From table 5.7 and 5.8 you are not able to distinguish a massive difference between the 4 machine learning algorithms, however you can tell that DT is the worst performing of the 4 algorithms and SVM is the best performing algorithm.

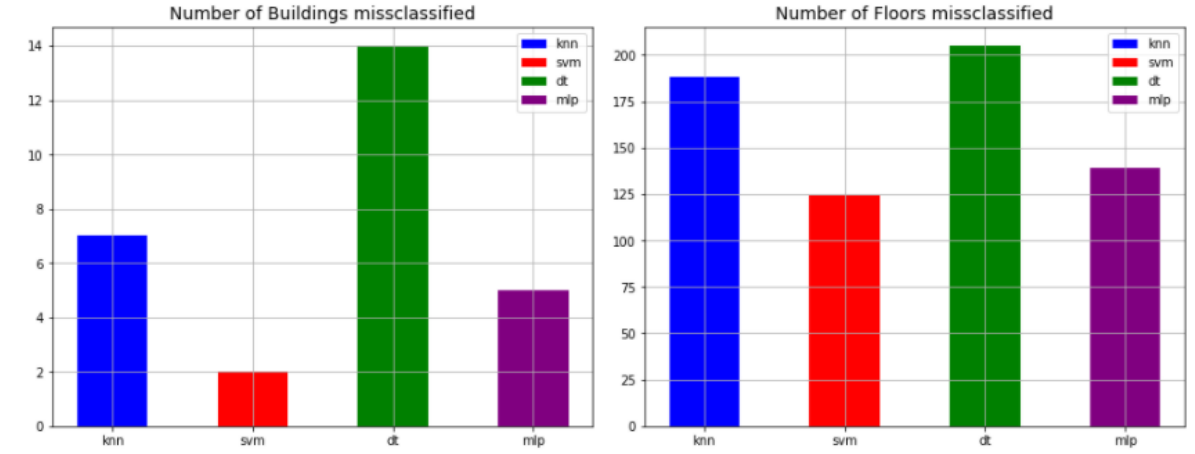


Figure 5. 6 - The number of misclassified buildings and floors. An algorithm with a tall bar has an accuracy lower than an algorithm with a small bar, e.g., DT has a much lower performance than SVM for both the number of buildings, and the number of floors.

This is further explained with figure 5.6, as SVM came in with the least number of buildings and floors misclassified when tested on the testing data. The biggest surprise comes from MLP, where it was not able to compete for performance in the regression problem, it was able to misclassify less buildings and floors than KNN.

To compare my results to the other works in literature, below is a table combing the results found in this report and other works in literature on the same competition graded dataset.

Table 5. 9 - Comparison of results based on Success Rate

Source	*Algorithm*	*Success rate (%)*	*Error (m)*
Ryanmclark, 2021	K-Nearest Neighbors	98.32	8.59
Moreira et al., 2015	HFTS	98.13	8.49
Gan et al., 2019	AFARLS	97.71	6.40
Moreira et al., 2015	RTLS@UM	96.87	6.20
Ryanmclark, 2021	Random Forest	96.65	10.57
Moreira et al., 2015	MOSAIC	96.26	11.64
Uddin and Islam, 2015	Extra-Trees	95.72	10.12
Kim, lee and Huang, 2018	Deep Neural Network	95.55	9.29
	Support Vector Machine	94.50	11.15
Ryanmclark, 2021	Decision Tree	94.21	13.88
Moreira et al., 2015	ICSL	93.47	7.67
	Multi-layered Perceptron	92.00	95233.92
Uddin and Islam, 2015	Support Vector Machine	91.53	11.28
	K-Nearest Neighbors	91.00	7.21
Torres-Sospedra et al., 2014	K-Nearest Neighbors	89.92	7.90
	Decision Tree	88.00	9.00
Uddin and Islam, 2015	K-Nearest Neighbors	86.23	13.43
Bozkurt et al., 2015	K-Nearest Neighbors	85.00	-
Akram, Akbar and Shafiq, 2018	HybLoc	85.00	6.46
Wietrzykowski, Nowicki and Skrzypczyński, 2017	Visual place recognition	78.00	8.21
Ryanmclark, 2021	Support Vector Machine	34.43	46.61

Table 5.9 is sorted in a descending order based on the success rate of the algorithm presented. The algorithms presented by me have been highlighted. The baseline algorithm set out by the creators of the UJIIndoorLoc database has been highlighted in green. The success rate describes how an algorithm performs in the classification of a building or floor.

Table 5. 10 - Comparison of results based on Error distance

Source	*Algorithm*	*Success rate (%)*	*Error (m)*
Bozkurt et al., 2015	K-Nearest Neighbors	85.00	-
Moreira et al., 2015	RTLS@UM	96.87	6.20
Gan et al., 2019	AFARLS	97.71	6.40
Akram, Akbar and Shafiq, 2018	HybLoc	85.00	6.46
	K-Nearest Neighbors	91.00	7.21
Moreira et al., 2015	ICSL	93.47	7.67
Torres-Sospedra et al., 2014	K-Nearest Neighbors	89.92	7.90
Wietrzykowski, Nowicki and Skrzypczyński, 2017	Visual place recognition	78.00	8.21
Moreira et al., 2015	HFTS	98.13	8.49
Ryanmclark, 2021	K-Nearest Neighbors	98.32	8.59
	Decision Tree	88.00	9.00
Kim, lee and Huang, 2018	Deep Neural Network	95.55	9.29
Uddin and Islam, 2015	Extra-Trees	95.72	10.12
Ryanmclark, 2021	Random Forest	96.65	10.57
	Support Vector Machine	94.50	11.15
Uddin and Islam, 2015	Support Vector Machine	91.53	11.28
Moreira et al., 2015	MOSAIC	96.26	11.64
Uddin and Islam, 2015	K-Nearest Neighbors	86.23	13.43
Ryanmclark, 2021	Decision Tree	94.21	13.88
Ryanmclark, 2021	Support Vector Machine	34.43	46.61
	Multi-layered Perceptron	92.00	95233.92

Table 5.10 is sorted in an ascending order based on the error distance of the algorithm. The algorithms presented by me have once again been highlighted to show how they compare with the other results in literature. The baseline algorithm set out by the creators of the UJIIndoorLoc database has been highlighted in green. The error distance describes the distance between the actual location stored in the dataset and the location predicted by the Machine Learning algorithm.

Chapter 6 Project management

6.1 Project life cycle

This project was managed using Trello, an online project management tool. Using Trello, the project was broken down into a week-by-week basis, where each week consist of 2 to 5 tasks. A task could be research or development based, where the research tasks consisted of reading and note taking from publicised papers, blogs, web articles, or YouTube videos and the development tasks were based around creating visual aids to be used in the report and for training and testing the machine learning algorithms. Development tasks were tested during and after creation to make sure they provided the correct results.

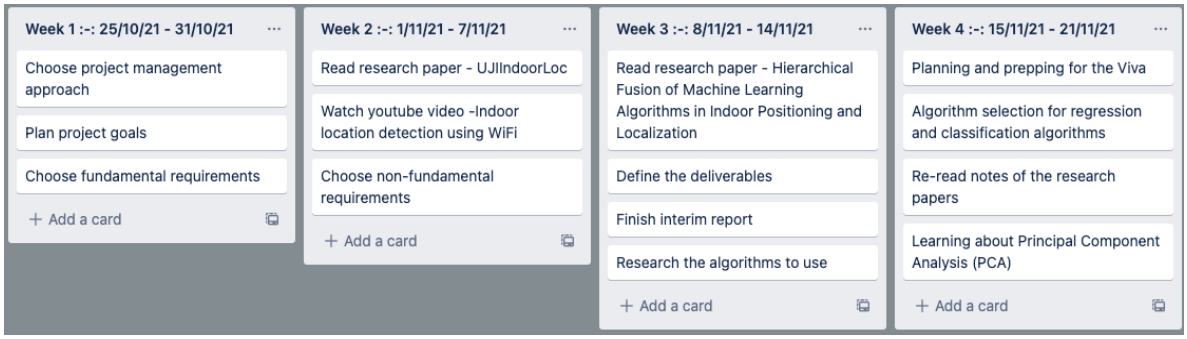


Figure 6. 1 - Trello board week 1 to 4

The first 4 weeks of the project were based around building a base knowledge of indoor positioning systems, the dataset that was going to be used for the machine learning algorithms, getting an understanding of how the machine learning algorithms used can solve classification and regression problems.

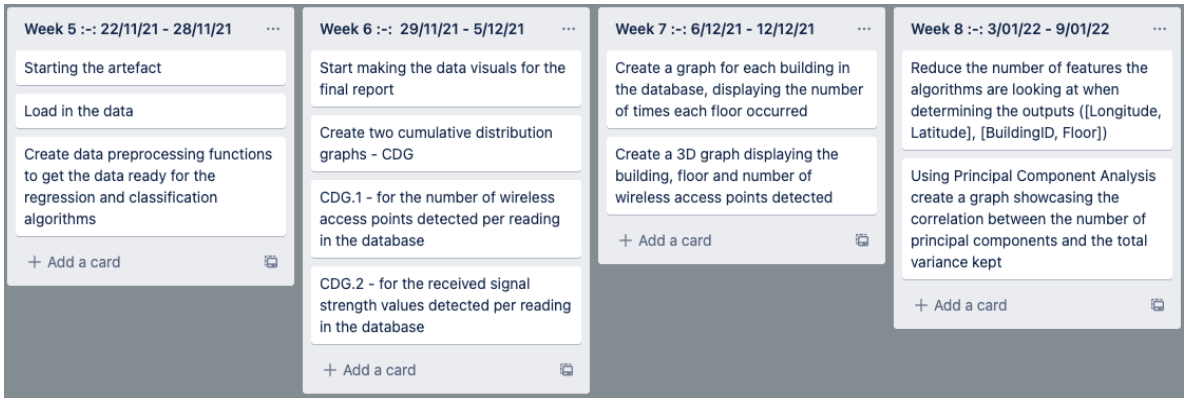


Figure 6. 2 - Trello board week 5 to 8

In the following 4 weeks (week 5 to 8), there was a focus on creating visual aids for the report. Here all the figures used for describing the dataset were created.

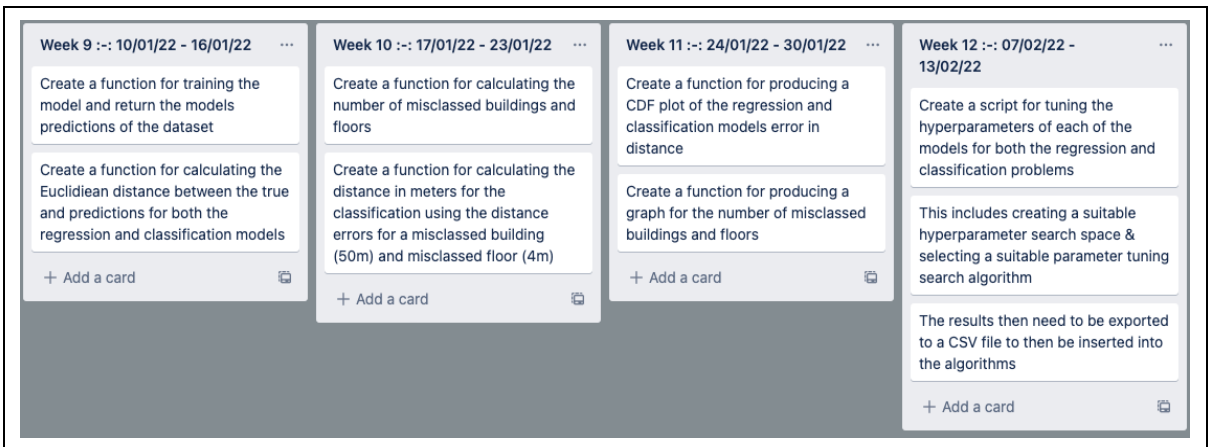


Figure 6. 3 - Trello board week 9 to 12

Weeks 9 to 12 were aimed at producing functions for the project. These functions were used for the training and testing of the models, calculating the Euclidean distance between the actual data point and the predicted data point, a function capable of producing the CDF plot for the results section above and finally the hyperparameter tuning script was created in the last week. The tuning script was given a bit more time than other tasks, as to run the script from start to finish would take around 4 to 16 hours depending on the size of the search grid provided.

Figure 6. 4 - Trello board week 13 to 16

At the beginning of this 4 week stint (weeks 13 to 16) the algorithm performance metrics were recorded, after which the focus of tasks shifted from development based to research based. In the research-based tasks the aim was to build upon the previous understanding of indoor positioning systems which were developed in the first 4 weeks of the project.

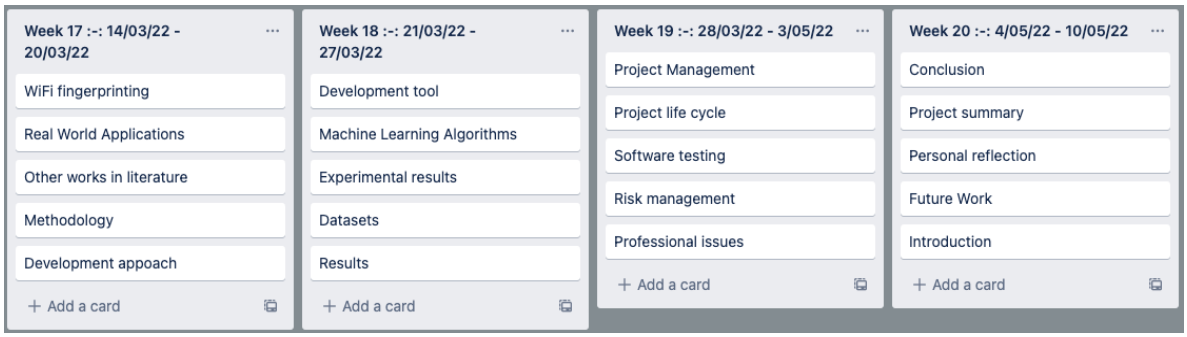
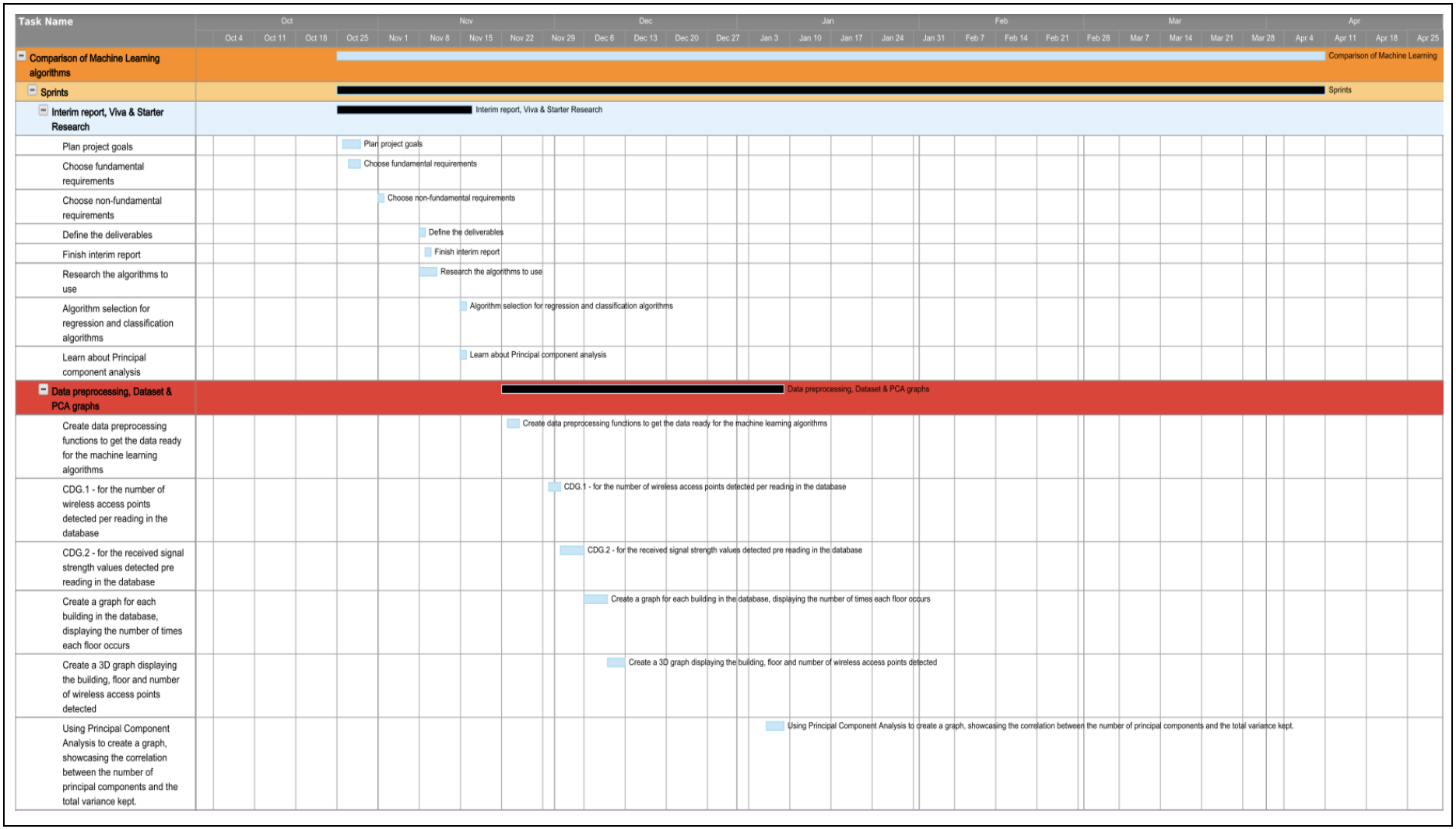


Figure 6. 5 - Trello board week 17 to 20

Finally, the last 4 weeks (week 17 to 20) of the project focused solely on the creation of the report.


Task Name
Oct
Nov
Dec
Jan
Feb
Mar
Apr
Oct 4
Oct 11
Oct 18
Oct 25
Nov 1
Nov 8
Nov 15
Nov 22
Nov 29
Dec 6
Dec 13
Dec 20
Dec 27
Jan 3
Jan 10
Jan 17
Jan 24
Jan 31
Feb 7
Feb 14
Feb 21
Feb 28
Mar 7
Mar 14
Mar 21
Mar 28
Apr 4
Apr 11
Apr 18
Apr 25
Comparison of Machine Learning algorithms
Comparison of Machine Learning
Sprints
Sprints
Interim report, Viva & Starter Research
Interim report, Viva & Starter Research
Plan project goals
Plan project goals
Choose fundamental requirements
Choose fundamental requirements
Choose non-fundamental requirements
Choose non-fundamental requirements
Define the deliverables
Define the deliverables
Finish interim report
Finish interim report
Research the algorithms to use
Research the algorithms to use
Algorithm selection for regression and classification algorithms
Algorithm selection for regression and classification algorithms
Learn about Principal component analysis
Learn about Principal component analysis
Data preprocessing, Dataset & PCA graphs
Data preprocessing, Dataset & PCA graphs
Create data preprocessing functions to get the data ready for the machine learning algorithms
Create data preprocessing functions to get the data ready for the machine learning algorithms
CDG.1 - for the number of wireless access points detected per reading in the database
CDG.1 - for the number of wireless access points detected per reading in the database
CDG.2 - for the received signal strength values detected pre reading in the database
CDG.2 - for the received signal strength values detected pre reading in the database
Create a graph for each building in the database, displaying the number of times each floor occurs
Create a graph for each building in the database, displaying the number of times each floor occurs
Create a 3D graph displaying the building, floor and number of wireless access points detected
Create a 3D graph displaying the building, floor and number of wireless access points detected
Using Principal Component Analysis to create a graph, showcasing the correlation between the number of principal components and the total variance kept.
Using Principal Component Analysis to create a graph, showcasing the correlation between the number of principal components and the total variance kept.

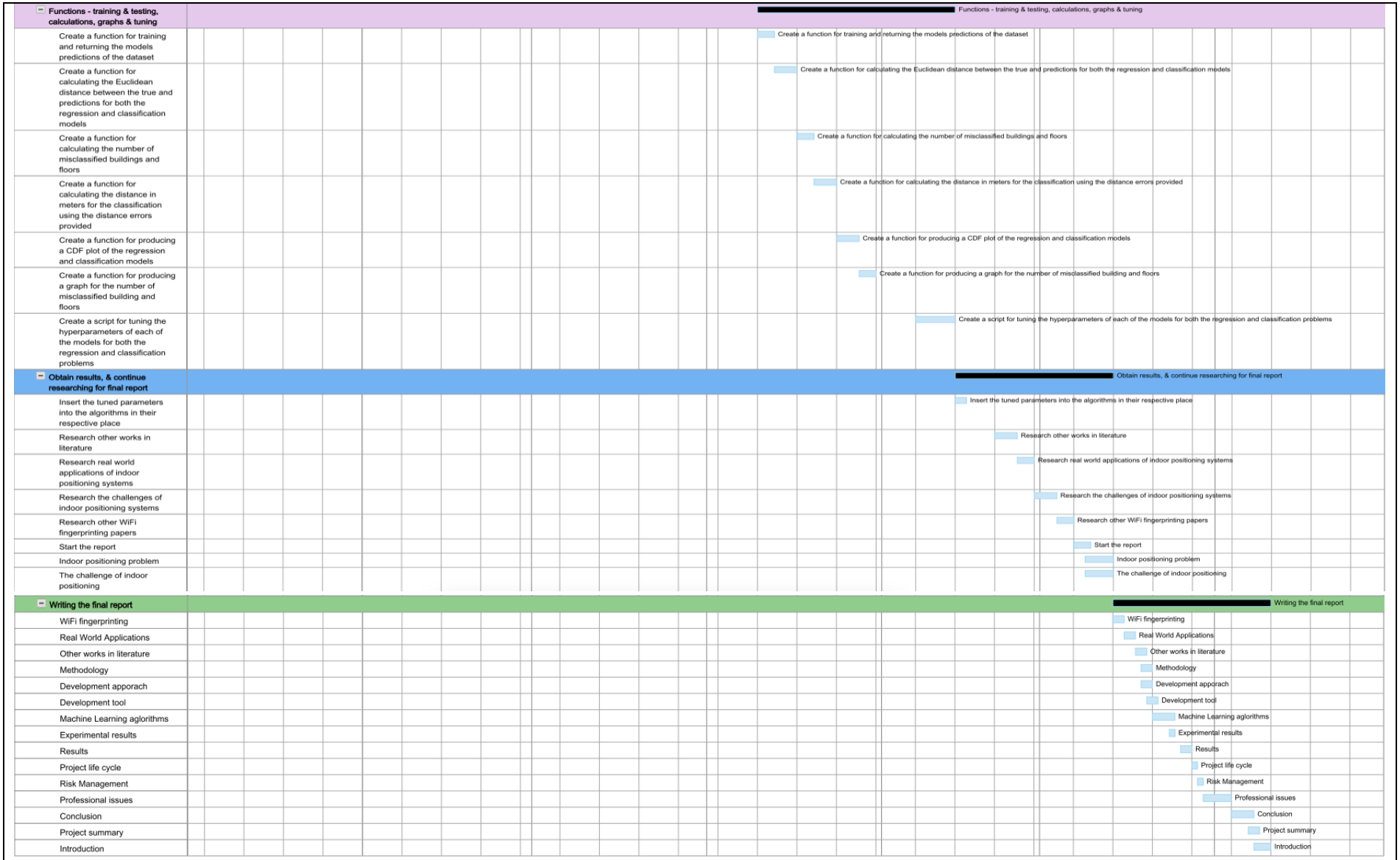


Figure 6. 6 - Gantt Chart of the project. Here there are 5 sprints in each of which there are a different category of tasks for each sprint. E.g., the second sprint – data pre-processing, dataset & PCA graphs is focused solely on the creation of graphs for the report. Whereas, in the third sprint – functions – training & testing, calculations, graphs and tuning was focused more towards obtain the results from the machine learning algorithms.

6.2 Risk management

The table below describes the problems that arose during the creation of this project and how they were addressed as well.

Table 6. 1 - Risk Assessment of Problems

Problem	Approach
The values in the dataset were affecting the performance of the model	The artificial RSS values were changed from +100 to -150 to help normalise the values before feeding them to the machine learning algorithms
If my laptop/computer got broken/stolen	All the files for the project were stored on an online google drive meaning the productivity of the project never slowed down
Noisy Data	The dataset contains 520 X values to feed to the network, this can cause the algorithms to overfit, so PCA was applied to the dataset to reduce the number of X values used during training and testing
Another lockdown	All the files for the project were stored online and the two computers used for the project both contained the required software for all parts of the project. Making it so the creation of this project could be done completely remotely.
The algorithm is overfitting	All algorithms were tuned using GridSearchCV to find the optimal parameters for each algorithm, where overfitting was reduced massively with the implementation of the parameters found here.
Falling behind on the tasks set out in the Trello board	All the tasks in the project were development quite quickly, thus making it so if any task took longer than expected, it could be move onto the next week of tasks without an issue to the flow of productivity.

Chapter 7 Professional issues

For IPS, ‘the challenge is to develop safeguards that simultaneously permit legitimate uses while prevent misuses’ (Dobson and Fisher, 2003). Even though IPS’s make life easier for businesses tracking goods in a warehouse, hospitals caring for patients or retail administrators tracking the real-time location of employees, IPS’s also bring many ethical issues with them.

In the article “Four Ethical Issues of the Information Age” (O. Mason, 1986) develops an understanding of ethics in relation to IT. These four issues are privacy, accuracy, property and accessibility. PAPA.

“The issue of privacy is important as geographical data collected by GPS devices can be misused if they fall into the wrong hands.’ (McNamee, 2005) This issue could be stopped from ever coming to fruition, if the geographical data is encrypted as it is collected from the devices.

‘GPS location data must be accurate if it is to be considered a viable data source.’ (McNamee, 2005) The same can be said for when you are using an IPS, e.g., tracking the movement of employees in your eco-friendly building, when people leave rooms the aircon, heating and lights are switched off, if the systems accuracy is not precise it could believe you to be outside the room when you are not, thus causing you to be without either aircon, heating and lights.

The issue of property can be related back to privacy when looking at IPS, as the intellectual property of your mac address as a user will be used by the operator of the IPS in the tracking of your mobile device, this could be cause for concern for some people.

Even though there is an increasing number of mobile devices being developed and deployed each year, ‘it is quite possible that a technology gap will grow between people who have access and those who do not.’ (McNamee, 2005) Once this gap becomes too big, what will happen to the people left behind?

The paper “Geoslavery” (Dobson and Fisher, 2003) is developed to ‘explore possibilities for misuse that many would consider unethical.’ (Dobson and Fisher, 2003) The term geoslavery is defined as, ‘practice in which one entity, the master, coercively or surreptitiously monitors and exerts controls over the physical location of another individual, the slave.’ (Dobson and Fisher, 2003) At the moment these types of systems can only be seen in outdoor environments such as for, ankle tag monitors, or dog shock collars. If these sorts of systems started to move into the indoor environment a massive ethical barrier could arise.

‘Is the real time monitoring and tracking or people morally right or wrong.’ (McNamee, 2005) The answer to this question is situational. Given a hospital setting, medical professionals will

need to know, for example, if a mentally ill patient is trying to leave their room unattended, and if they do leave, they will then need the precise location of said patient. However, in a shopping centre do all customers who connect to the WiFi need to have their location tracked during their visit?

The literature of LBS (Location Based Systems) tracking and monitoring does not contain any details on its social and legal implications.

Chapter 8 Conclusion

In this report, selected Machine Learning algorithms have been compared against other works in literature and themselves, in terms of success rate, a combination of the building and floor accuracy of the algorithms and error distance, the distance in metres between the actual value stored in the dataset and the value predicted by the algorithms. All the Machine Learning algorithms used for comparison were all trained and tested using the UJIIndoorLoc database. The aim of the project was to find the most appropriate classifier and the most appropriate regressor for the indoor positioning problem.

8.1 Project summary

Using table 5.9 as an aid, it is evident that the best performing algorithm for the classification problem was presented by (ryanmclark, 2021) with their K-Nearest Neighbors algorithm with a success rate of 98.32%, whereas from the algorithms presented in this paper the best performing algorithm was the Support Vector Machine with a success rate of 94.50%. However, if you compare the results obtained by me in this paper with the baseline algorithm presented by the creators of the database you can see that the Support Vector Machine, Multi-Layered Perceptron and K-Nearest Neighbors are all able to provide a better success rate.

Moving on, in table 5.10, you can see that the best performing algorithm for the regression problem was (Moreira et al., 2015) where the authors, a team of people, were able to achieve an error distance of 6.20m, whereas from the algorithms presented in this paper the best performing for the regression task was the K-Nearest Neighbors algorithm with an error distance of 7.21m. But once you compare my results to the baseline algorithm once again you can see that only the K-Nearest Neighbors algorithm is able to provide less of an error distance.

Even though the algorithms presented in this paper were not able to obtain results better than others in literature, the results presented here can be used as a reference point for people looking to solve the indoor positioning problem.

8.2 Personal reflection

Over the course of the project, I have continued to learn about different techniques to solving classification and regression problems, producing usual visual aids for understanding the problem area and a vast knowledge around WiFi and indoor positioning systems. I would have never come across this knowledge during my standard degree. I believe if I was focusing solely on the production of this project, and I did not have other modules to complete I would have been able to provide a Machine Learning algorithm from scratch capable of producing more accurate results.

8.3 Future work

Finally, soon, I estimate the following trends to emerge:

- IPS's being used by companies to reduce their carbon footprint using employee location tracking and monitoring.
- IPS's being incorporated into shopping centres and high-rise buildings for tracking people during a fire.
- IPS's being used in conjunction with GPS to produce real-time tracking both for indoor and outdoor environments.

References

Abbas, M., Elhamshary, M., Rizk, H., Torki, M. and Youssef, M. (2019). *WiDeep: WiFi-based Accurate and Robust Indoor Localization System using Deep Learning*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/abstract/document/8767421 [Accessed 10 Nov. 2021].

Akram, B.A., Akbar, A.H. and Shafiq, O. (2018). HybLoc: Hybrid Indoor Wi-Fi Localization Using Soft Clustering-Based Random Decision Forest Ensembles. *IEEE Access*, 6, pp.38251–38272 [Accessed 25 Mar. 2022].

Analytics Vidhya (2019). *A Practical Introduction to K-Nearest Neighbor for Regression*. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/ [Accessed 27 Mar. 2022].

Bahl, P. and Padmanabhan, V. (2016). *RADAR: An In-Building RF-based User Location and Tracking System*. [online] Available at: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/infocom2000.pdf [Accessed 7 Nov. 2021].

Bambrick, N. (2016). *Support Vector Machines: A Simple Explanation*. [online] Kdnuggets.com. Available at: https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html [Accessed 13 Mar. 2022].

Bozkurt, S., Elibol, G., Gunal, S. and Yayan, U. (2015). *A comparative study on machine learning algorithms for indoor positioning*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/abstract/document/7276725 [Accessed 12 Apr. 2022].

Bressler, N. (2021). *How to Check the Accuracy of Your Machine Learning Model*. [online] Deepchecks. Available at: https://deepchecks.com/how-to-check-the-accuracy-of-your-machine-learning-model/ [Accessed 14 Mar. 2022].

Dasgupta, D. (2020). *Introduction to R-Sqaure in Linear Regression*. [online] GreatLearning Blog: Free Resources what Matters to shape your Career! Available at: https://www.mygreatlearning.com/blog/r-square/ [Accessed 22 Feb. 2022].

Deep AI (2019). *Multilayer Perceptron*. [online] DeepAI. Available at: https://deepai.org/machine-learning-glossary-and-terms/multilayer-perceptron [Accessed 19 Mar. 2022].

Dobson, J. and Fisher, P. (2003). *IEEE Technology and Society Magazine, Spring 2003 47 Geoslavery*. [online] Available at: https://dusk.geo.orst.edu/virtual/2005/geoslavery.pdf [Accessed 15 Apr. 2022].

Frost, J. (2017). *Understand Precision in Predictive Analytics to Avoid Costly Mistakes - Statistics By Jim*. [online] Statistics By Jim. Available at: https://statisticsbyjim.com/regression/prediction-precision-applied-regression/ [Accessed 14 Mar. 2022].

Gan, H., Khir, M.H.B.M., Witjaksono Bin Djaswadi, G. and Ramli, N. (2019). A Hybrid Model Based on Constraint OSELM, Adaptive Weighted SRC and KNN for Large-Scale Indoor Localization. *IEEE Access*, [online] 7, pp.6971–6989. Available at: https://ieeexplore.ieee.org/abstract/document/8594559 [Accessed 10 Apr. 2022].

Huellmann, T. (2022). *What is precision vs recall in machine learning?* [online] levity.ai. Available at: https://levity.ai/blog/precision-vs-recall [Accessed 15 Mar. 2022].

Kim, K.S., Lee, S. and Huang, K. (2018). A scalable deep neural network architecture for multi-building and multi-floor indoor localization based on Wi-Fi fingerprinting. *Big Data Analytics*, 3(1) [Accessed 1 Apr. 2022].

Korstanje, J. (2021). *The F1 score*. [online] Medium. Available at: https://towardsdatascience.com/the-f1-score-bec2bbc38aa6 [Accessed 17 Mar. 2022].

LT, Z. (2022). *Essential Things You Need to Know About F1-Score*. [online] Medium. Available at: https://towardsdatascience.com/essential-things-you-need-to-know-about-f1-score-dbd973bf1a3 [Accessed 22 Mar. 2022].

Mcnamee, A. (2005). *Bachelor of Information and Communication Technology (Honours)*. [online] p.71. Available at: https://ro.uow.edu.au/cgi/viewcontent.cgi?article=1003&context=thesesinfo [Accessed 14 Apr. 2022].

Moreira, A., Nicolau, M.J., Meneses, F. and Costa, A. (2015). *Wi-Fi fingerprinting in the real world - RTLS@UM at the EvAAL competition*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/abstract/document/7346967 [Accessed 12 Apr. 2022].

Mujtaba, H. (2020). *An Introduction to Grid Search CV | What is Grid Search*. [online] GreatLearning. Available at: https://www.mygreatlearning.com/blog/gridsearchcv/ [Accessed 19 Mar. 2022].

National Coordination Office for Space-Based Positioning, Navigation, and Timing (2014). *GPS.gov: Trilateration Exercise*. [online] www.gps.gov. Available at: https://www.gps.gov/multimedia/tutorials/trilateration/ [Accessed 4 Mar. 2022].

National Coordination Office for Space-Based Positioning, Navigation, and Timing (2019). *GPS.gov: GPS Accuracy*. [online] Gps.gov. Available at: https://www.gps.gov/systems/gps/performance/accuracy/ [Accessed 6 Mar. 2022].

Nguyen, K.A. (2017). A performance guaranteed indoor positioning system using conformal prediction and the WiFi signal strength. *Journal of Information and Telecommunication*, [online] 1(1), pp.41–65. Available at: https://khuong.uk/Papers/a_performance_guaranteed_indoor_positioning_system.pdf [Accessed 6 Nov. 2021].

Nguyen, K.A., Luo, Z., Li, G. and Watkins, C. (2021). A review of smartphones-based indoor positioning: Challenges and applications. *IET Cyber-Systems and Robotics*, 3(1), pp.1–30 [Accessed 10 Nov. 2021.

Niu, J., Wang, B., Cheng, L. and Rodrigues, J.J.P.C. (2015). *WicLoc: An indoor localization system based on WiFi fingerprints and crowdsourcing*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/abstract/document/7248785 [Accessed 4 Nov. 2021].

O. Mason, R. (1986). *(PDF) Four Ethical Issues of the Information Age*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/242705009_Four_Ethical_Issues_of_the_Information_Age [Accessed 13 Apr. 2022].

Rawat, S. (2019). *Is accuracy EVERYTHING?* [online] Medium. Available at: https://towardsdatascience.com/is-accuracy-everything-96da9afd540d [Accessed 7 Mar. 2022].

Rojo, J., Mendoza-Silva, G.M., Ristow Cidral, G., Laiapea, J., Parrello, G., Simó, A., Stupin, L., Minican, D., Farrés, M., Corvalán, C., Unger, F., López, S.M., Soteras, I., Bravo, D.C. and Torres-Sospedra, J. (2019). *Machine Learning applied to Wi-Fi fingerprinting: The experiences of the Ubiqum Challenge*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/abstract/document/8911761 [Accessed 6 Apr. 2022].

ryanmclark (2021). *Localization_via_WiFi_Fingerprinting*. [online] GitHub. Available at: https://github.com/ryanmclark/Localization_via_WiFi_Fingerprinting [Accessed 7 Nov. 2021].

Seldon (2021). *Decision Trees in Machine Learning Explained*. [online] Seldon. Available at: https://www.seldon.io/decision-trees-in-machine-learning [Accessed 24 Mar. 2022].

So, J., Lee, J.-Y., Yoon, C.-H. and Park, H. (2013). *Download Limit Exceeded*. [online] citeseerx.ist.psu.edu. Available at: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.587.2737&rep=rep1&type=pdf [Accessed 10 Nov. 2021].

StatisticsHowTo (2022). *Mean Squared Error: Definition and Example*. [online] Statistics How To. Available at: https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/mean-squared-error/ [Accessed 3 Mar. 2022].

Sun, S. (2018). *Wi-Fi signals: reflection, absorption, diffraction, scattering, and interference*. [online] www.youtube.com. Available at: https://www.youtube.com/watch?v=UxDdwGhSf4o&ab_channel=SunnyClassroom [Accessed 3 Mar. 2022].

Torres-Sospedra, J., Montoliu, R., Martínez-Usó, A., Avariento, J.P., Arnau, T.J., Benedito-Bordonau, M. and Huerta, J. (2014). *UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/document/7275492 [Accessed 4 Nov. 2021].

Torres-Sospedra, J., Montoliu, R., Mendoza-Silva, G.M., Belmonte, O., Rambla, D. and Huerta, J. (2016). Providing Databases for Different Indoor Positioning Technologies: Pros and Cons of Magnetic Field and Wi-Fi Based Positioning. *Mobile Information Systems*, 2016, pp.1–22.

Uddin, Md.T. and Islam, M.M. (2015). *Extremely randomized trees for Wi-Fi fingerprint-based indoor positioning*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/abstract/document/7488051 [Accessed 14 Apr. 2022].

Wietrzykowski, J., Nowicki, M. and Skrzypczyński, P. (2017). Adopting the FAB-MAP Algorithm for Indoor Localization with WiFi Fingerprints. *Automation 2017*, pp.585–594 [Accessed 18 Apr. 2022].

Wireless LAN Professionals (2017). *Indoor Location Detection using Wifi | Marko Tisler | WLPC EU Budapest 2016*. [online] www.youtube.com. Available at: https://www.youtube.com/watch?v=vtfnlgTj_-A&t=1378s&ab_channel=WirelessLANProfessionals [Accessed 6 Nov. 2021].

Appendices

Appendix 1

Record of supervisor meetings:

Date	Guidance
5/11/21	What are you going to make? Why are you going to make it? How are you going to do it? How do you know it is achievable? What have you read to guide you?
12/11/21	What machine learning algorithms are you going to use? Produce graphs to explain the distribution of WAPs and the RSS values
19/11/21	Produce graphs of the occurrences of floors in buildings Produce a 3D graph of the dataset
16/11/21	Learn about Principal Component Analysis Produce PCA graphs
3/12/21	Current slides are too technical Structure Viva presentation to address the "why", "what" & "how"

10/12/21	Select the machine learning algorithms to be used for comparison Select performance metrics for machine learning algorithms
10/1/22	Apply the machine learning algorithms to the dataset Produce results of the performance of each algorithm
17/1/22	Produce a hyperparameter tuning script
24/1/22	Learn about CDF plots and what they explain Produce CDF plot for the regression model Showcase the number of buildings & floors misclassified
7/2/22	Provide all the errors produced by the algorithms
14/2/22	Produce all the results for the algorithms to be used in the final report
21/2/22	Start the creation of the final report
7/3/22	Provided comments on my report and how I can improve upon what was displayed to him
21/3/22	Add appendices to the report

Appendix 2

Source code is found in the following GitHub link:

https://github.com/Quints497/WiFi_Fingerprinting_Comparison