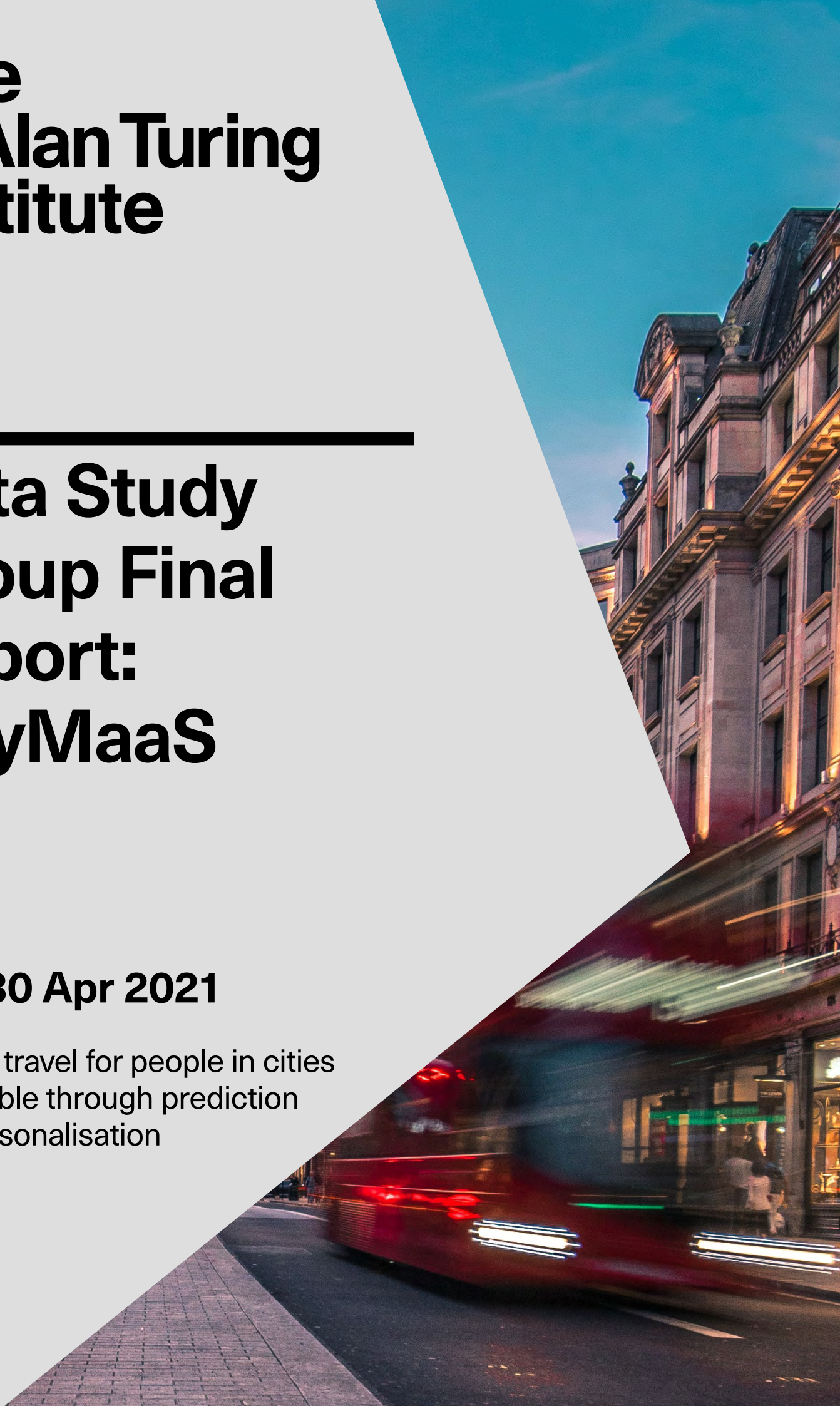
The Alan Turing Institute

Data Study Group Final Report: CityMaaS



12 – 30 Apr 2021

Making travel for people in cities accessible through prediction and personalisation

https://doi.org/10.5281/zenodo.6798694

Contents

1 Executive summary

1.1 Challenge overview

1.1.1 Motivation and summary

One of the most challenging parts of daily life for disabled people is travel. In most cities around the world, accessible travel remains inefficient. On average, completing a trip takes people with limited mobility 2.5 times longer compared to able-bodied people due to poor infrastructure such as the lack of step-free access, or real-time accessibility information. At the same time, navigation platforms fail to address the individual needs of people with disabilities, as they lack customisable routing options, on grounds of producing "mass-used" solutions. Providing accurate information about the accessibility of locations and having tailored developed routes based on personal needs and preferences, poses a real-world challenge that is yet to be fully addressed. Enabling people with disabilities to travel freely around cities will not only give each individual a greater sense of independence, but also allow for society as a whole to benefit from everything this large community has to offer.

This report presents the outputs of the 2021 Data Study Group by the Alan Turing Institute and CityMaaS, a UK based company with the vision of personalising the way the disabled community consumes accessibility data and services. The challenge presented by CityMaaS is how can we develop a platform that provides reliable information on the accessibility of destinations, and at the same time be flexible enough to address people with a varying set of needs. Thus, it can be divided into the following two sub-challenges:

- Predicting the accessibility of points of interest (POIs) in a city;
- Personalised route-planning with accessibility constraints.

CityMaaS had already a proof of concept (POC) in place for the POI accessibility prediction model. The POC for the prediction task, which is a supervised learning problem, used a random forest model to achieve an accuracy of 80% and has been utilised on their website[1]. The routing task

[1] `https://demo.citymaas.io/`

is unsupervised, and posed the further question of how to optimise routes for different “personas” who have different accessibility constraints. These accessibility constraints include, for example, the inability to travel routes involving stairs or the need to avoid routes with obstacles blocking footpaths for people using a wheelchair, or a preference for avoiding crowded locations which could lead to overstimulation for people with autism. Within the concept of personalised routing, we constructed an algorithm using open elevation data for London, for detecting short steep ascents along a route which might indicate staircases or any other sections of route too steep for a wheelchair user. We then used a score to measure the preference of avoiding steep slopes, avoiding main streets (based on street type categorisation) and avoiding obstacles per user, to develop a constrain matrix. This matrix is used to calculate the overall score of each route and recommend an ordering of routes based on the priorities of different users. In this report we will go through the development of this custom ”routing engine” and how we used it to produce different suggested routes based on a set of hypothetical users with varying disabilities.

1.2 Main objectives

Accessibility is the concept of whether a product or service can be used by everyone—however they encounter it. In the case of CityMaaS, a POI, such as a cafe or a museum, is classified as accessible if this facility/place/amenity is physically accessible by wheelchair or for people with similar mobility constraints. For example, a tube station will be classified as wheelchair accessible if it has access to a lift and/or ramp. The OpenStreetMap wiki[2] provides a comprehensive guide to tags useful for the needs of people with disabilities. Based on the above definition, the objectives of the challenge were twofold:

1. Improve the accuracy of the POI prediction above the current standard of 80% based on existing data, with the possibility of exploring derived datasets and additional public data sources.

2. Create a POC routing model to select the best route between two

[2] https://wiki.openstreetmap.org/wiki/How_to_map_for_the_needs_of_people_with_disabilities

locations given a possibly varying set of accessibility constraints.

As remarked above, the first objective was a supervised learning problem. CityMaaS provided the group with a dataset of POIs, some of which were labelled indicating their accessibility. The objective was to build a model to automatically classify unlabelled datapoints. The purpose would be to produce a tool where for each selected destination, there will be a prediction on how accessible it may be. This could then also affect the routing suggestion of the engine. Thus, the system would provide a reliable suggestion for a user with limited mobility to reach a certain destination or POI.

The second objective was an unsupervised learning problem. CityMaaS provided the team with a list of personas (typical artificial users), each having different mobility needs, that would require personalised routing suggestions. The challenge would be to build a model using indicators from available datasets to produce route suggestions based on their needs. For example, a person on a wheelchair is likely to need disabled parking lots, but additionally the route should not include steps or steep slopes. A user with a walker or rollator may be able to bypass obstacles, but may require elevators and longer time to reach to a destination, if going on foot.

1.3 Approach

1.3.1 POI accessibility prediction

For the POI accessibility prediction task, it was necessary to first to explore the importance or weight of the features included in the OpenStreetMap (OSM) POI dataset (see Figure 1).

Within the challenge, we explored OSM metadata and new features from other potentially relevant sources of data, such as, for example, the number and proportion of nearby accessible toilets. We then assessed the importance of these features in relation to accessibility.

Then, several classifier models were developed to compare their performance based on their precision metric. The precision is the most relevant scoring value in this case as it relates to a low false positive rate (avoiding POI labelled as accessible when in reality they are not). This

false positive risk was considered by the group as the most pressing need for users of the CityMaaS service. If the system would predict an accessible POI as not accessible, this could potentially be a nuisance or disruption to the user because it limits their availability of resources. If, however, a not accessible POI was predicted as being accessible, this could in some situations seriously impact the user, for example by leaving them stranded at a train station or unable to enter a building they were relying on entering. It was decided by the group that a risk averse approach should therefore be taken. Of these classifiers the XGBoost, Support Vector Machine (SVM) and Random Forest (RF) models were used on the entire dataset. In addition, the XGBoost and Extra Trees models were also applied to a small subset of the data consisting of restaurants. The purpose of this localisation to a subset was to explore whether additional sub-features (e.g. restaurant cuisine) could be used to train a more precise model.

1.3.2 POC routing model

In the second task, we developed a routing engine which integrates measures of accessibility. We used the Python routing engine OSMnx (see Secion 4.2) along with NetworkX to produce a wheelchair/walking route between two locations. The model uses Dijkstra's algorithm to compute the shortest path between the locations, with the edge weight set by default to length. Light Detection and Ranging (LIDAR) data (see Section 2.2.2) were used to detect any particularly steep sections along the route and, if these exceeded a threshold, additional alternative routes were generated. LIDAR is a technology similar to RADAR that can be used to create high-resolution digital elevation models with a coverage of around 50cm for England. The LIDAR data were also used to compute the total uphill and downhill elevation changes along a route. These metrics could be used for ranking the routes according to elevation change.

For each route, we also computed the number of nearby accessible POIs along the route using the POI data from the first task. These metrics could be used to rank the routes according to accessible POIs nearby.

We also considered different *personas* and assigned them a score on four different "accessibility dimensions" – namely avoiding steep slopes,

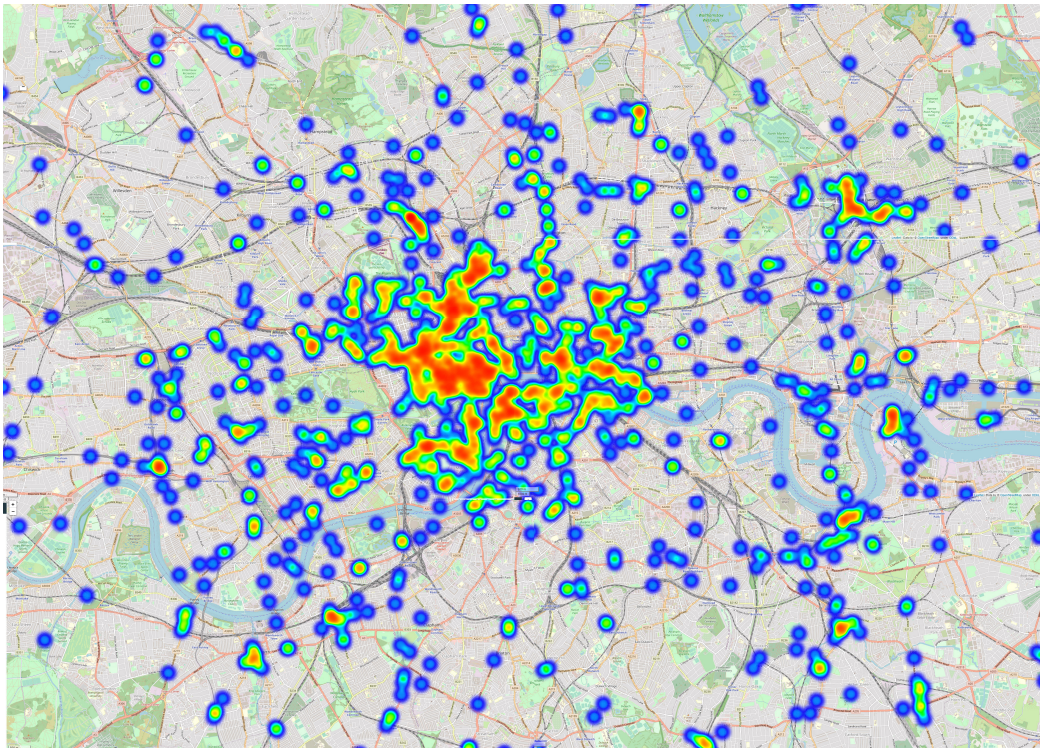


Figure 1: Heatmap of wheelchair accessible POIs in London, data by OSM, as of April 2021. In the wider area of London there is a lack of data on accessibility. A prediction algorithm can help fill in the gaps for a more reliable route recommendation system.

avoiding obstacles, avoiding busy streets and prioritising the shortest distance. By comparing them with the metrics computed for each route we can recommend routes to the user depending on their specific needs. This approach allows for the personalisation of routes depending on individual needs and accessibility constraints.

1.4 Main conclusions

1.4.1 POI prediction

On the prediction task, we showed that it was possible to improve the POI accessibility classifier past the existing RF model by using an XGBoost

model with several additional derived features as well as features from external datasets. We found that adding derived features greatly improved the precision and recall of our predictive models. With these additions, we found that our chosen XGBoost model achieved 91% accuracy when predicting accessible POIs. In particular, the improvement was likely the result of the new features which encompassed the number of POIs within certain distances of each POI. This, coupled with the strong dependence on latitude and longitude, suggests that the strongest correlation with wheelchair accessibility is geographical, particularly linked to density. We found that of our tested classifiers, the XGBoost model performed best across all metrics and furthermore required fewer trees than its closest competitor, the RF, to achieve a slightly higher precision score.

1.4.2 Routing

For routing, we created a POC for a routing algorithm ("routing engine") that attempts to avoid obstacles in the form of steep route sections, busy areas, and that can be set to attempt to minimise overall distance or up/down elevation change, and maximise number of accessible POIs nearby to the route. For this purpose, we incorporated the open-source routing engine OSMnx with LIDAR data (point elevation data) for detecting elevation changes as well as the provided OSM POI data which contain information on obstacles. We developed an algorithm [3] for creating several different wheelchair/walking routes between any two locations and assigning scores on 4 different metrics to these. End-users could then choose from the routes according to their own needs by comparing the values along the different metrics.

Due to the short duration of the challenge, the project focused on wheelchair/walking routes. There are still additional accessibility challenges once a user is within a public transport system (e.g. changing trains), but we decided that the initial challenge was routing for an individual walking or using a wheelchair.

[3] `https://github.com/alan-turing-institute/DSGApril2021CityMaas`

1.5 Limitations

There were several limitations which affected the outcome of the project and have to be considered when interpreting our results.

1.5.1 POI prediction

- Data was imbalanced: approximately 73% of labelled points were accessible. A possible explanation for this could be that people are more likely to label a place as accessible than non-accessible. For future work it would be prudent to repeat experiments with a balanced dataset which may improve precision on 'non-accessible' predictions. The open question here is whether this distribution of accessibility is representative of the true distribution.
- High Cardinality: Owing to the fact that the POI data is taken from OSM, where users are allowed to freely input tags, there are many possible values for each feature. This leads to a high cardinality data set with poor predictive power in these features. We address this partially in the challenge by manually homogenising the data, however, a more robustly labelled data set would also likely improve performance.
- Data Noise. It is worth mentioning the general noise/robustness issues of relying on a crowdsourced set such as OSM. Data have limited labelling and they are subject to the user preferences. For this reason, CityMaaS are looking into integrating further official sources into their platform, as well as their own platform's user inputs.
- Finally, as this was a case study applied in the area of the UK, and in the case of the routing algorithm in Greater London, this may arguably produce geographical bias. For example, due to the prevalence of institutional architecture with a legal accessibility requirement.

1.5.2 Routing

- The approach was only tested on a limited amount of data located in the centre of London. Further testing would be necessary to establish whether the method performs equally well in less urban regions which might also carry additional challenges not yet included in the model, such as missing dropped curbs.
- The way the method currently avoids obstacles is through recalculating the route to take another street where potentially another obstacle could be encountered. The iterative process is not very computationally efficient which could prove challenging especially for complex and long routes.
- The system currently only supports routes by foot, it does not take into consideration public transport or car routing.
- The employed LIDAR data has an accuracy of around 50cm, meaning that there is room for error when calculating slopes.

1.6 Recommendations and future work

One potential research challenge that could derive from this work is to integrate the POI prediction model within the routing engine. For example, certain POIs predicted to be accessible that have an essential function en-route (such as a bus stop) could then be passed to the routing engine as a good location to route through. Conversely, POIs predicted as non-accessible could be avoided.

The following sections contain recommendations for the separate projects within the challenge:

1.6.1 POI prediction

- As discussed in Section 1.5.1, the labelled POIs in the OSM data for Great Britain were imbalanced, with approximately 73% of labelled POIs being accessible. This meant that model precision was higher when predicting “accessible” vs predicting “non-accessible”. Therefore, we recommend to investigate the imbalance of POI data and whether this corresponds to the true data distribution or

whether it is an artefact of using OSM. For example, one hypothesis is that people are more likely to enter POI accessibility information if it is accessible than if it is non-accessible. Exploring additional datasets would also be strongly suggested.

1.6.2 Routing

- The open-source LIDAR data we used from the Department for Environment, Food & Rural Affairs (DEFRA) had a resolution of 1 metre. It would be interesting to investigate higher-resolution LIDAR data and whether this improves the elevation component of the routing model – for example, predicting highly steep sections.
- Not all "obstacles" referenced in Section 4.4.1 are necessarily non-accessible and their accessibility can be user-dependent. Therefore, it would be helpful to filter these more precisely so that the model only avoids POIs that are truly non-accessible.
- Integrate the list of OSM obstacles with the routing model fully. (See Section 4 for a list of all the obstacles integrated)

2 Data overview

Data provided by CityMaaS included manually enhanced point data from Open Street Map (OSM), however, during this challenge the participants creatively explored additional data obtained from open sources.

2.1 POI prediction

2.1.1 Dataset description

The provided OSM data contained geolocated information on POIs around the UK. These could be, for example, the location of a restaurant, its name and address, the type of amenity and service area, and whether it is wheelchair accessible. The datasets were provided in three categories (accessible, non-accessible and unlabelled) and separate `.json` files were provided for the accessible POIs in each country.

Directly from OSM	Processed from OSM tags	Augmented and derived
Latitude (num)	Access (cat)	Index of Deprivation (ord)
Longitude (num)	Barrier (cat)	Toilets total (num)
Wheelchair acc. (bool.)	Bicycle (cat)	Number acc. toilets (num)
	Car (cat)	Acc./total toilets ratio (num)
	Opening hours (text/cat)	Number nearby acc. POI (num)
	Type (cat)	Ratio nearby acc. POI (num)

Table 1: Overview of available features which were taken directly from the provided dataset, extracted and pre-processed, or augmented from external sources. Datatypes: numeric (num), categorical (cat), ordinal (ord), boolean, text. Toilet number and nearby POI numbers were calculated at four radii (100m, 250m, 500m, 1km).

OSM Json exports typically report data using a `key=value` format for each POI, in which the value can also be left empty. The following fields were included:

- An ID;
- A dictionary of OSM tags, including among others category (e.g. “restaurant”), address, wheelchair accessibility and name;
- Longitude and latitude.

In the initial dataset for Great Britain, 29,282 POIs were present, with nine columns of features. After homogenisation and the addition of new features, this extended to 30 columns of available data. Additionally, after deleting rows with missing data, we are left with 21,129 POIs. A summary is provided in Table 1. In the following sections we describe the process of homogenisation and how additional metadata was used to augment the dataset.

2.1.2 Data quality issues

Concerning the data, there were two main objectives: at first, there was a data cleaning process, in which we homogenised tags and values of OSM POIs. As data is input by the users, slight differences in the naming may be observed, though different values may refer to the same property of a POI. Secondly, we also worked on trying to enrich the dataset by

assessing which additional data we could extract from the OSM dataset, and importing external data.

Another issue we observed in the wheelchair accessibility data was a class imbalance. Approximately 73% of the POIs examined were wheelchair accessible, leading to better prediction for these points than those that were not wheelchair accessible. An important nuance is that whether or not this is actually a problem depends on whether this distribution is representative of the true distribution of POIs. If the true distribution of POIs is also skewed in this way, then our classification algorithms would be unbiased. However, we have no way of determining this from the OSM data alone. Accordingly, we recommend that CityMaaS compare the OSM data against more robust data sources such as Ordnance Survey in future investigations.

2.1.3 Data homogenisation

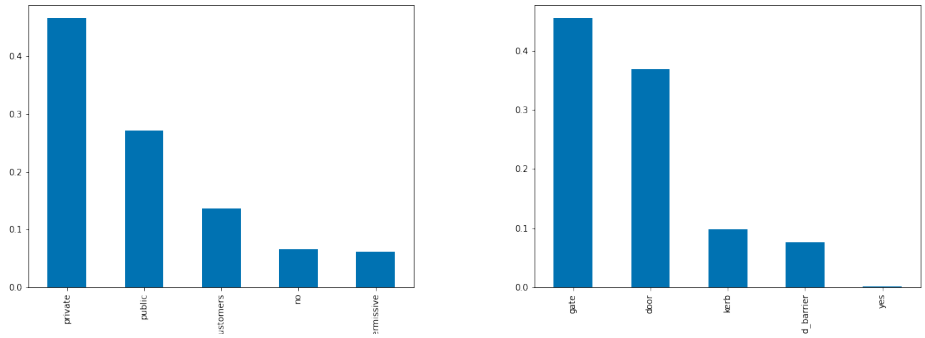
The main objective of the data homogenisation sub task is to avoid having naming errors in the dataset, and blend the data into common categories and values.

The data show a distribution on which there are few points which divert from the most popular tag values, which may confuse prediction models and adds noise to the dataset. Hence, we have worked towards homogenising this data, by reading different values in the dataset and changing them to one of the values we selected to keep. The result of this can be seen in Figure 2, which shows that the number of values for each tag has been reduced, and the dataset is now more clean and ready to be processed.

2.1.4 Additional data: POI data from OpenStreetMap

A second objective for the pre-processing data task was to explore additional sources that can potentially enrich the dataset, to improve the prediction of POIs accessibility.

A first possibility is to use metadata already contained in the OpenStreetMap dataset. OpenStreetMap data is a crowdsourced platform, in which users have the freedom to add additional tags to any



(a) Access tag distribution.

(b) Barrier tag distribution.

Figure 2: Distribution of homogenised tags in the available dataset with frequency on the y-axis and label on the x-axis.

POIs. Therefore, we began by plotting wordclouds of the tags to see if any useful tags may emerge in order to be added into the dataset.

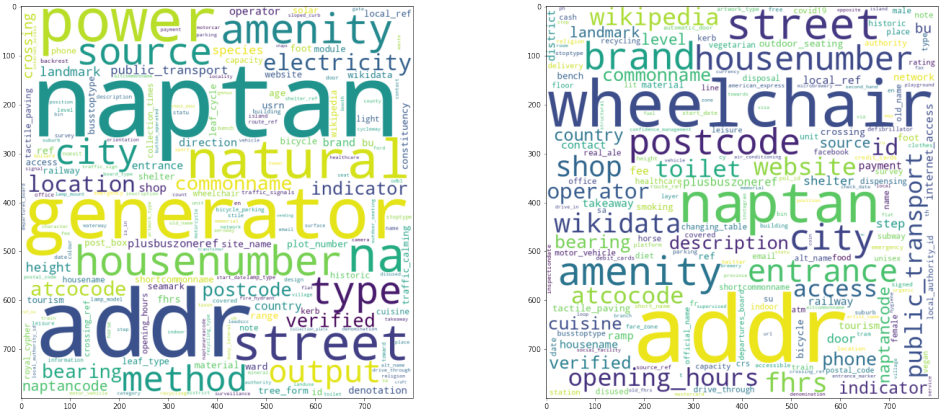
We have performed two different types of analysis, one for all the POIs (Figure 3a), and one using POIs which contain accessibility information (Figure 3b).

We observed that the word distribution shows some of the tags standing out in terms of frequency compared to others. These are the `addr` tag, the `naptan` tag, and the `generator` tag, as all the others appear less frequently. For example, address seems to be available for all the POIs while wikidata seems to have a lot of empty values. The analysis showed that the most common tags contained little information relevant to the accessibility prediction. Thus, we decided against proceeding in this direction.

2.1.5 Additional data: Toilets

An additional dataset which we considered adding to the CityMaaS challenge is the ”public toilets in the UK” dataset. The source for these geolocated data is the Great British Toilet Map website[4], which also provides an API with easy access to a fully complete dataset. Using this

[4] `https://www.toiletmap.org.uk/`



(a) All POIs (b) POIs tagged with Accessibility

Figure 3: Wordclouds for POIs with and without accessibility information. The larger a word appears in a wordcloud, the more frequent it is in the dataset. The first wordcloud was formed using all available datapoints including those for which no accessibility information was available, while the second contained only POIs with accessibility information. The analysis presents merged specialisations and removed all missing values.

dataset, we computed 12 additional features on the principal dataset, which correspond to the number of total toilets in a given radius from a POI, the number of accessible toilets in the same radius, and the ratio between accessible toilets and all the toilets in a given radius. We then computed these features for four different radii, namely at 100m, 250m, 500m and 1km. The data relating to the total number of toilets and the ratio have then been added to the main dataset for the CityMaaS challenge.

2.1.6 Additional data: Deprivation Index data

Another important factor beyond the addition of features to improve POI prediction was to look into how census data may relate to accessibility. For example, we considered whether more deprived areas in London may have fewer accessible spaces. For this we used the Index of Multiple Deprivation (IMD, 2019) for England, available from the UK government[5]. The IMD is a measure of overall deprivation of an area based on 39 various indicators across 7 themes (e.g. Income, Employment, Health Deprivation and disability, etc.). Neighbourhoods of England are ranked based on the IMD score. The IMD decile of 1 represents the most deprived areas and 10 the least deprived ones. We used the IMD available at Lower Super Output Area (LSOA) level and used the LSOA boundaries (December 2011) for England available from the Office for National Statistics[6] to integrate the location of POI to the decile of IMD it belongs to.

2.1.7 OSM categories of amenities: Restaurants

Going back to the augmented OSM dataset, it is noticeable that some grouped categories of amenities may contain special features that others do not. For example, restaurant POIs have features such as cuisine type, level of restaurant and restaurant name etc. which could be informative features to infer the accessibility. In addition, in the case of pub or supermarket POIs, the brand of pubs and supermarkets could also reflect accessibility information as large chains may be more accessible to

[5] https://www.gov.uk/government/statistics/english-indices-of-deprivation-2019
[6] https://geoportal.statistics.gov.uk/datasets/

different customer needs. Therefore, it is worth considering splitting the dataset into different categories and training them with different models.

Data homogenisation and cleaning
For the purposes of this sub-challenge we extracted restaurants by filtering the OSM ”amenities” field. Most of the restaurant POIs share the following two unique features: cuisine types and name of the restaurant. The location of the city is also considered a feature.

- **Cuisine types**: Initially, there were more than 200 types of cuisines. This is due to inconsistent names given by different information providers. Merging the cuisine types into 8 main categories was carried out to create a cleaner and simpler dataset for further processing.
- **Name of restaurant**: The name of the restaurant also had more than 1000 keys, but for chain restaurants the names given were relatively consistent. A potential way to further classify the name of the restaurant is to mark them as a chain restaurant or non-chain restaurant. But for the purpose of this analysis the restaurant names were kept as they were used for the vector representation.
- **City of location**: City of location was a good variable as the name used was consistent throughout the dataset. The dataset covers restaurants in more than 1000 cities in the UK.

It should be mentioned that although these three features are largely shared among the restaurant POIs, a small proportion of POIs may have missing information. However, this did not affect the analysis due to their relatively small number.

Vector representation
One of the key challenges for training the word dataset was how to represent different keys in features effectively in the model. Features can only be treated as vectors in machine learning models and the way we represent them plays a key role in the model performance. For a binary feature, i.e. a true/false Boolean feature, this can be easily done by replacing them with 0 and 1 (e.g. chain-restaurant/ non-chain restaurant). However, for features with more than two keys, such as cuisine types, or

features only containing two keys without a binary relationship, some other techniques were required.

A simple way to do this is using sequential representation, i.e. represent different keys in features by sequential numbers $\{0, 1, 2, 3, \dots\}$. This can be a straightforward approach but could also be problematic because of the following two reasons:

1. Different keys in features are not created equally.
2. Different keys do not have a sequential relationship between them.

Another approach to producing text representations is using word embeddings, a natural language processing (NLP) concept that converts a word to a vector by considering its relationship with the rest of the words in the whole vocabulary, as well as the sample sequence. The leading approaches to producing these embeddings relies on extensive pre-training on a large language-specific corpus, during which the co-occurrence of words is used to produce a vector which captures some of their semantic information. This is usually combined with fine-tuning, during which the pre-trained vectors are re-trained with a task-specific corpus, producing more specialised outputs for a particular task domain. Without this re-training process however, word embeddings may be a good representation of the dictionary meaning of a word while not representing well a key in our feature list. In addition, the internal relationship structure of our keys is not dependent on their general semantic meaning and would be ignored.

Our objective was to identify an effective way for converting keys into vectors that considers the relationship between different keys, minimises the information lost and can be easily replicated for new keys. Based on this consideration, using the frequency of keys in features could be an effective approach. The normalised frequency of a key (i.e. the number of occurrences of the key/total number of the dataset) contains information about the scale of that feature. For example, in the name of restaurant feature, Pizza Hut may have a frequency of 0.2 while a local restaurant may only have a frequency of 0.001. This representation assumes that there is a relationship between the scale of the restaurant brand and

wheelchair accessibility. Similarly, this could be used for city and cuisine types. Intuitively, this makes sense to a degree, as the larger the city is the more accessible the restaurants in that city might be and similarly, the bigger a brand is, the more likely accessible facilities might be provided.
For those POIs that do not have information for a specific feature, it is fair to set them as zero as they may be a unique name/cuisine/city that never again appears in the dataset. With this approach, the three features we were interested in could be converted to their normalised frequency and intuition behind them are shown below:

1. The normalised frequency of restaurant name: the scale of the restaurant brand
2. The normalised frequency of cuisine types: the popularity of the cuisine type
3. The normalised frequency of the city of location: the relative size of the city

2.2 Routing task

2.2.1 Dataset description

Information on end-user personas was provided by CityMaaS. These personas were artificial people modelled after potential users of the CityMaaS platform. In order to offer personalised routes for each customer, our focus was on different end-user personas, as a way of grouping users together based on needs and personal preferences. Rather than deciding on a one-size-fits-all route, the idea was to use the persona concept to personalise the best route for an individual persona.

Each persona had the following fields:

- Place of residence (district within a city)
- Civil status
- Profession
- Monthly income

- Hobbies
- Segment (indicating travel usage)
- Ticket (indicating usual way of paying for travel)
- Information source (indicating method of obtaining travel data)
- Conditions (indicating specific accessibility constraints)
- Need
- Enjoys (specific preferred accessibility adjustments)
- Goals (general travel-related goals)

From the 12 end-user personas provided by CityMaaS, we selected six that aligned with the POC being created. These personas were chosen by the group participants for representing an array of different, both physical and mental needs, which the group assumed to impact their travel behaviour. We further focused our attention on those personas with disabilities as opposed to those needing additional support due to circumstances, such as travelling with luggage or prams. While this group presents another type of CityMaaS user, we considered it beyond the scope of this project and decided to focus on users with disabilities only. The following Table 2 describes the end-user personas selected:

Description	Explanation	Persona
Motor impairment	Wheelchair User	Claudia Perth
Motor impairment	Walker rollator	Oscar Davies
Blind	Braille alphabet	Leonie Johnson
Visual impaired/ Partially sighted	Cloudy and faded world	Steven Wright
Learning disorder	Motor skills	Allison Clarke
Autism	Social phobia	John Lobster

Table 2: End-user personas selected. Shown are the six personas that were selected for the POC of the routing engine. The first column describes their impairment, the second column shows their individual needs which affect their routing needs, and the third column personifies the persona further with a name.

2.2.2 Additional data: Using spatial data from OpenStreetMap

Identifying additional data was also necessary for the routing recommendation task. As no starting point or routing data was available, in order to generate a POC the main data for this task was external.

Spatial data from OSM was therefore employed as the base information on which to build our POC. OpenStreetMap allows users to download spatial data to model, project, visualise, and analyse real-world street networks. Users can download and model walkable, driveable, or bikeable urban networks and then easily analyse and visualise them. Moreover, users can easily download and work with other infrastructure types, amenities/points of interest, building footprints, elevation data, street bearings/orientations, and speed/travel time.

2.2.3 Additional data: LIDAR data

Another dataset chosen for the routing task was LIDAR data, which was used to check the elevation change along a route. We used the Digital Terrain Model (DTM) at 1m created from LIDAR data and downloaded from DEFRA[7].

2.3 Data summary

The data employed for this challenge can be summarised as the data provided by CityMaaS together with additional data from several sources. An overview is provided in Figures 4 and 5, which show example rows for the provided data and augmented data of accessible toilets.

For the routing task, the above mentioned OSM data was used as well as LIDAR data in the form of elevation level per geospatial location given with latitude and longitude.

[7] `https://environment.data.gov.uk/DefraDataDownload/`

id	access	barrier	bicycle	motor_vehicle	opening_hours	wheelchair	amenity	lon	lat
99878	permissive	gate	no	no	dawn-dusk	yes	None	-0.152985	51.524358
104734	NONE	None	None	None	None	yes	None	-1.785876	51.565653
106213	NONE	None	None	None	None	yes	None	-0.142942	51.525660
108042	NONE	None	None	None	Mo-We 16:00-23:30; Th-Fr 16:00-01:00; Sa 16:00...	limited	pub	-0.135513	51.523561

Figure 4: Four example rows of available POI data with different variables.

all_loos_250	all_loos_500	all_loos_1000	ratio_100	ratio_250	ratio_500	ratio_1000	amenity	Index of Multiple Deprivation (IMD) Decile
1	7	12	0.0	0.0	0.428571	0.25	None	8.0
0	1	3	0.0	0.0	0.000000	0.00	None	4.0
0	4	12	0.0	0.0	0.500000	0.50	None	6.0
2	4	20	0.0	1.0	0.500000	0.35	pub	6.0

Figure 5: Second part of the table in Figure 4. Four example rows of accessible toilet and Deprivation Index data which are linked to the data in Figure 4 via unique ID. The displayed rows are not necessarily the same as in the previous Figure. For each ID the number of accessible and non-accessible toilets is calculated for different radii, as described in Section 2.1.5. For each ID and associated location the Deprivation Index is derived as described in Section 2.1.6. Not shown is the number of nearby accessible POIs which were added to the dataset in the same way as the toilet data.

3 Predicting the Accessibility of POIs

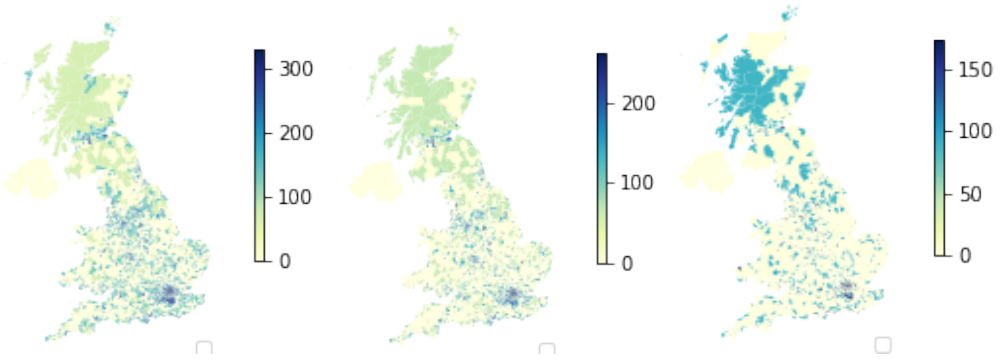
Here we present different experiments we conducted to improve the prediction of accessible POIs. We tested different algorithms to evaluate the performance, to explore whether a particular selection of algorithms may play a role in predicting accessibility. As a first step in the task, we visualised the spatial distribution of the data to better comprehend it and aid in the research question. Afterwards, the first broad research question we sought to answer as part of this challenge was whether we could predict the accessibility of a given POI based on some information given about it. For the scope of this challenge, we focused on one type of accessibility in particular: wheelchair accessibility. This presented a classification problem – based on some input features to be explored, would a POI be classed as wheelchair accessible or not. Accordingly, we explored the utility of three types of supervised learning classification models for the task: Support Vector Machines (SVM), Random Forests (RF), and XGBoost. Deep Learning approaches were briefly discussed as another avenue, however due to time constraints and an interest of the group to move onto the second research question of a routing engine, they were eventually dismissed as beyond the scope of the project.

3.1 Spatial distribution of POIs

We explored how the OSM POI data varies spatially across our study area in the UK. For this we used the ward level boundaries of UK provided by the Office for National Statistics[8], as of May 2020. Figure 6b shows the density of overall POI data distribution for the UK. It illustrates POIs are quite unevenly distributed across the area, with most POIs being concentrated around urban centres. This indicates a lack of completeness in terms of spatial coverage in the OSM POI dataset used. The spatial bias could be a result of imbalance in our dataset on the accessibility information across the POIs under consideration.

From the available POIs, we extracted the ones showing to be wheelchair accessible (tagged as wheelchair=yes). The density is shown in Figure 6b. The density distribution of the accessible POIs corresponds to the

[8] https://geoportal.statistics.gov.uk/datasets/



(a) All POIs (b) Accessible POIs. (c) Non-accessible POIs.

Figure 6: Density distribution of POIs in Great Britain. The density of POIs is plotted with darker colours indicating a higher point density and lighter colours a more sparse one. Only POIs in Great Britain (England, Scotland, Wales) are considered for the analysis.

overall distribution of all POIs. Some spatial bias is also noticable with the majority of accessible POIs being located in major cities/towns. We also explored the density distribution of the inaccessible POIs, in terms of the POIs categorised to have no accessibility for wheelchairs (Figure 6c). This also concentrates on major city centres, as visible for London. In this figure the very low inaccessibility is not a result of accessibility information but rather the lack of information on accessibility in it.

We have also particularly looked at POIs in Greater London to create a heatmap (Figure 7). The figure shows a kind of spatial bias, with the central part of London to be largely covered by accessible POIs compared to its outskirt.

3.2 Support Vector Machines

SVMs are a supervised-learning algorithm that can be used for both classification and regression. In the case of classification, SVMs work by finding a hyperplane that divides up the data into classes such that a margin between the hyperplane and points on all sides is maximised.

For this investigation, we set up our SVM using the `scikit-learn`

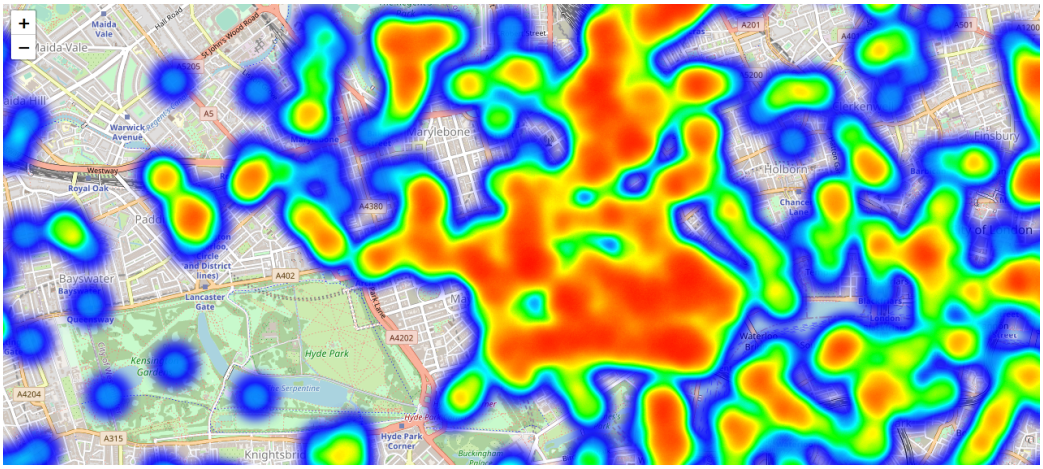


Figure 7: Heatmap of accessible POIs in the centre of London. The density of accessible POIs is shown visually with red and orange regions indicating higher density and blue regions indicating lower density. No colour indicates no available data. Online interacting version of the heatmap can be found on the project's GitHub account [9].

package. We used a cross-validated grid search in order to perform hyperparameter optimisation. Our final choice of parameters was as follows: a linear kernel with a kernel coefficient of 0.12, and a regularisation parameter of 0.1.

3.3 Random Forest

Ensemble learning methods are frequently used to achieve better performance by combining several weak learners into a combined, strong one. The RF is one such type of ensemble learning model. Such models consist of ensembles of Decision Trees, where the predictions of individual trees are averaged over in order to improve the accuracy and prevent overfitting.

In this work, we created our RFs using the `scikit-learn` library. The RFs were instantiated with 250 trees, and each tree was set to use a Gini impurity score as its decision making mechanism. This quantity is a representation of how often a predicted class for a particular input would be incorrect, if the class labels were predicted randomly.

3.4 XGBoost

An alternative ensemble learning method we studied makes use of Gradient Boosted Decision Trees (GBDT). To implement these, we used the XGBoost package [2]. Boosting is an ensemble learning technique whereby the various learners in the ensemble are trained sequentially, rather than simultaneously. A specific form of this is gradient boosting, in which subsequent learners are trained on the prediction error residuals of the previous learner in the sequence. Within this investigation, we used a GBDT model with 100 rounds of boosting. As with the RF case, the Gini score was used to make decisions within the individual trees.

3.5 Results and Discussion

3.5.1 Feature Analysis

In order to identify which of the many input features were most informative, we determined feature importance scores using both the RF and XGBoost models. This analysis will allow CityMaaS to select a limited number of features going forward; reducing both the volume of data that must be collected for the prediction algorithm, as well as the complexity of any models constructed.

Figures 8 and 9 show the feature importance scores obtained from the RF and XGBoost models respectively, before data homogenisation. Additionally, in this case, we examined only the features already supplied with the data. In the RF case, the importance score is the Gini importance which describes the normalised total number of divisions which include a given feature. On the other hand, the XGBoost model computes feature scores using the F-score. This metric was chosen as it is a similar metric to the Gini importance, with the difference that the F-score in contrast to Gini is unnormalised. The F-score is the sum of the number of times a given feature is split on, over all trees. It therefore tells us about the discriminate power of a specific feature in separating classes, in contrast to an alternative metric such as gain which describes the contribution of a feature to the model.

In both cases, we observe similar trends. Overwhelmingly, the dominant features appear to be latitude and longitude. This could indicate a

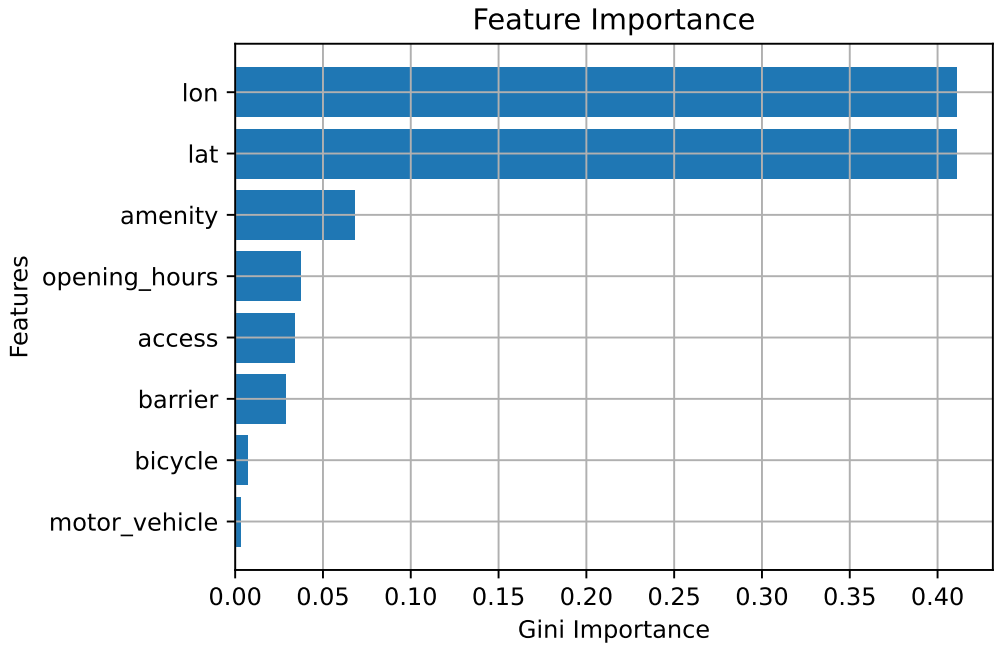


Figure 8: Gini feature importance score for the unprocessed, initially provided input features, obtained from the Random Forest model. The y-axis shows the different features with their Gini score on the x-axis.

particularly strong geographical trend in wheelchair accessibility; naively, one could intuit that more isolated areas are also less accessible. However, we recall that as discussed in Section 2.1.3, each of the features can take a large range of possible values, due to having been manually input by OSM users. This high cardinality within the data somewhat compromises the reliability of these scores.

Accordingly, we also recomputed the feature importance, now using only the XGBoost model, after the homogenisation procedure described in Section 2.1.3. Furthermore, we now also included the derived features and the accessible toilets data described in Section 2.1.4. This is shown in Figure 10. From this, we see once again that latitude and longitude are the most important features. However, now the new derived features encapsulating nearby accessible POIs are also relatively important. In particular, the number of POIs within 100m and the number of accessible

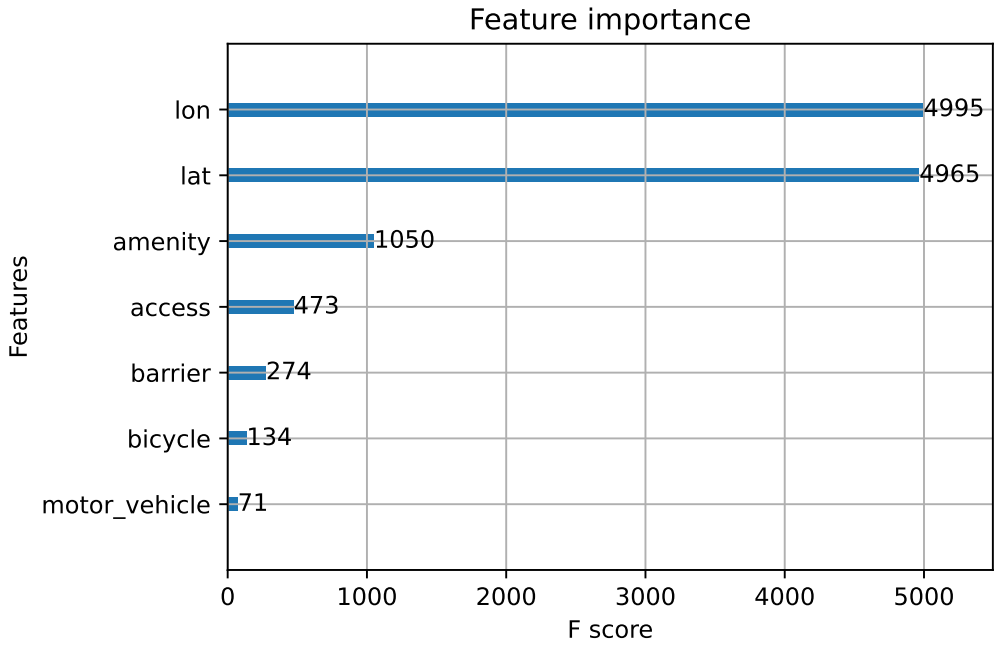


Figure 9: F-score importance metric for the unprocessed, initially provided input features, obtained from the XGBoost model. Shown are on the y-axes the input features and on the x-axis their respective F-score.

POIs within 100m achieve relatively high F-scores. Given that both these new features, and the latitude and longitude represent geographical correlations, this lends credence to the idea that geographical location tends to be the best predictor of wheelchair accessibility. Interestingly, the IMD is a relatively weak predictor of accessibility, suggesting that the POI density of a given region is more important in determining accessibility than how deprived that area is.

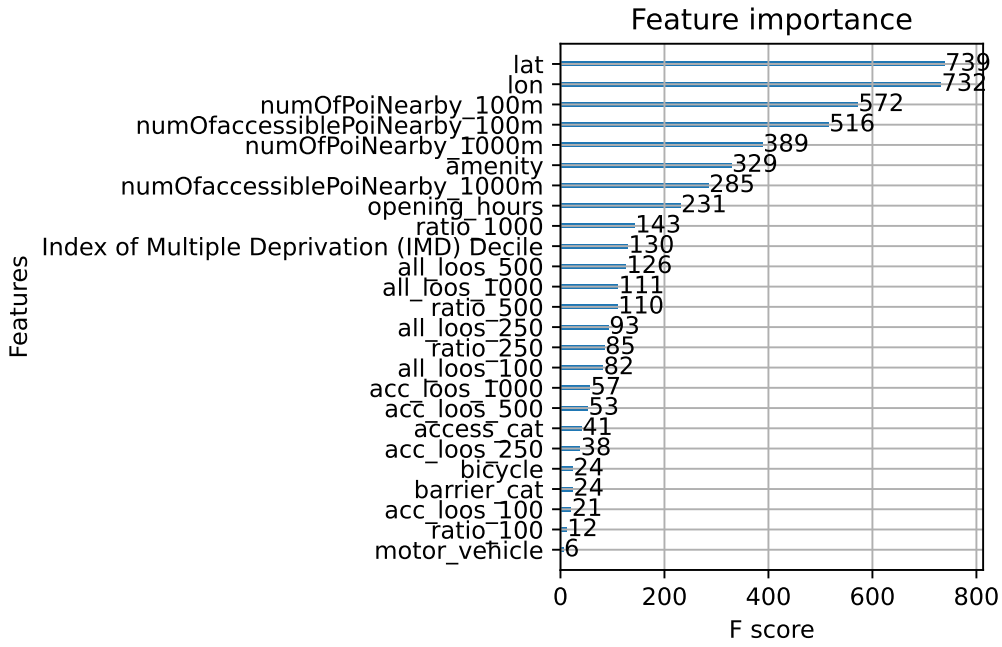


Figure 10: F-score importance metric for the homogenised and derived features, obtained from the XGBoost model. Shown are on the y-axes the input features and on the x-axis their respective F-score.

Lastly, we also used variance inflation factors, in order to test for multicollinearity within the features. This analysis revealed the following features as being independent: `opening_hours`, `lon`, `access_cat`, `numOfPoiNearby_1000m`, `acc_loos_100`, `amenity`. However, in order to carry out this analysis, we had to drop data points with missing data for any of the features, reducing the amount of data available, and meaning that this analysis may not necessarily be reflective of the true multicollinearity present.

These results were obtained with the central London dataset and could be due to the large amount of institutions in the sample data. A useful suggestion was to compare the differences with a PCA excluding these, which would be interesting to be considered in future work.

3.5.2 Predictive Performance

Here, we compared the efficacy of the three studied models – SVMs, RFs, and XGBoost – at predicting the wheelchair accessibility of a given POI. We compared on the basis of three metrics in order to quantify this: the precision, the recall, and the F1 score of the models. We considered the precision of the models to be of the highest importance among the three metrics due to the given application. Therefore, when comparing the models we consider that with the highest precision to be superior. By doing this we are minimising the false positive rate. This metric was chosen to be the most important because we want to minimise the chance of a wheelchair user arriving at a POI recommended by the CityMaaS service, only to discover the location is not in fact accessible.

We first trained the Random Forest model on exclusively the initial data and features provided, in order to establish a baseline of prediction accuracy. The scores for this model can be found in Table 3. Then, input features were homogenised, and the additional derived and external features were added. The performance of three models, an SVM, the Random Forest and an XGBoost model, are stated in Tables 4 and 5. We also note that any POIs with `NaN` values for any of the individual features were dropped for the purposes of this comparison. Accordingly, the training data consisted of 15846 POIs, while the training set contained 5283 points – a 75-25 test-train split. The data was also stratified in a representative manner when split, making both the training and test sets representative of the overall wheelchair accessibility distribution.

Model	Precision	Recall	F1-score
Baseline RF ‘yes’	0.78	0.86	0.82
Baseline RF ‘no’	0.60	0.53	0.56

Table 3: Comparison of precision, recall, and F1 score obtained for the ‘yes’ and ‘no’ wheelchair accessibility classes. Result of baseline RF model using non-homogenised data and with no derived features added.

Model	Precision	Recall	F1-score
SVM	0.84	0.96	0.89
Random Forest	0.90	0.94	0.93
XGBoost	0.91	0.94	0.93
XGBoost (undersampled)	0.96	0.84	0.90

Table 4: Comparison of precision, recall, and F1 score obtained for the 'yes' wheelchair accessibility class from three types of classifier: an SVM, a Random Forest, and XGBoost. The homogenised data set is used and additional data and derived features are included. Given that the dataset is imbalanced to favour wheelchair accessible POIs, a comparison of the predictive power of XGBoost when this majority class is undersampled is also provided.

Additionally, given the imbalance in data discussed in Section 2.1.2, we also tested an instance of the XGBoost model where the majority class, the 'yes' class for wheelchair accessibility, was undersampled so that the training set consisted of 3779 instances of both classes, each. The results of this model are also shown in Tables 4 and 5.

Comparing Table 3 on the unaltered dataset with Tables 4 and 5, we see that the inclusion of the derived and additional features significantly improved all three metrics of performance. Consideration of the Random Forest model on unaltered data and the Random Forest model on augmented data shows the improvement of the same model's performance when including features. Of particular note is the significant improvement in the recall of the 'no' class. Additionally, we see that the RF and XGBoost models have a stronger performance than the SVM model, particularly when predicting the 'no' class, and are competitive with each other across all metrics. We also note that undersampling the majority class does not significantly affect performance. Counter-intuitively, this increases the precision of the 'yes' class, while reducing the precision of the 'no' class. Whether this is to be considered an improvement depends on the precise ways these labels would be fed into the overall CityMaaS service.

Model	Precision	Recall	F1-score
SVM	0.76	0.40	0.52
Random Forest	0.80	0.69	0.74
XGBoost	0.80	0.71	0.75
XGBoost (undersampled)	0.64	0.88	0.90

Table 5: Comparison of precision, recall, and F1 score obtained for the 'no' wheelchair accessibility class from three types of classifier: an SVM, a Random Forest, and XGBoost. The homogenised data set is used and additional data and derived features are included. Given that the dataset is imbalanced to favour wheelchair accessible POIs, a comparison of the predictive power of XGBoost when this majority class is undersampled is also provided.

Going forward, we recommend the use of the XGBoost model for accessibility prediction, as it is able to achieve the best results, and with only 100 trees in comparison to the 250 required by the RF. An additional advantage of the XGBoost model is its ability to include POIs with no data in certain features, whereas other models would require the omission of those points all together.

4 Routing Engine with user-based accessibility metrics

Here we describe how to modify an open source routing engine to include obstacles and score routes based on an individuals abilities. This way, we create a system where recommendations can be made based on levels of accessibility as well as shortest paths. There are a number of obstacles that can be included in the routing engine, however, those we managed to explore during the short time of the DSG are staircases and steep ascents, slope elevation profile and busyness on the route. We then developed an accessibility matrix, which can be scored per user needs. This section focuses on the development of a proof-of-concept algorithm to compute and rank routes against several accessibility metrics.

4.1 Approach

The goal of this task was to create a routing algorithm to find the best route – or a *list* of best routes – between two locations, given a set of accessibility constraints. Our approach was to "outsource" the routing problem to a routing engine of choice to output a selection of possible routes. At this stage, it is also possible to specify a list of points to avoid when recommending a route (e.g. locations that are known/predicted to be non-accessible). These routes can then be assigned *weights* according to several metrics. The end-user could then choose a route from the options, or add their constraints to an app which then selects the best route for their particular needs.

For example, one measure of the weight of a route is the total elevation change along a route. If a user has particular accessibility needs that make it difficult to travel on routes with large elevation changes, the routes could be ranked according to total elevation change.

4.2 Choice of routing engine

The team originally decided to use Open Source Routing Machine (OSRM) for the routing task. OSRM can be given two locations (or a list of locations) and outputs a list of routes from one to another (passing through any specified midpoints). It is available as a remote service or as a Docker image that can be downloaded and run from the associated GitHub repository[10].

For security reasons we were not able to run the Docker image on the DSG Safe Haven and could therefore only access the remote version of OSRM, which suffers from several key limitations:

- The number of alternative routes is capped at $n = 3$;
- The only allowed route method is *driving*.

For these reason we decided to instead use the Python package OSMnx [1]. OSMnx uses geospatial data from OSM in street networks and other geospatial geometries. It has options for walkable, driveable and bikeable urban networks and integrates OSM POIs as nodes in the network.

[10] `https://github.com/Project-OSRM`

The routing engine has been built keeping also future extensions into account. We have condensed all the contributions into a single python function, which upon being called creates a route between two points, using the `osmnx` library, adds relevant meta-data information to it, and adds it to a route catalogue. This enables the user to create virtually any kind of route, and keep all of them together so they can be analysed, compared, and ranked. Moreover, the function that builds the route was developed to accommodate any future additional metadata enrichment of it. In other words, any future work can be added as an additional piece of metadata to be included in the route definition.

4.3 Deciding whether a route is accessible or not

A particular route may not be accessible for two rather different general reasons: in the first case, there may be highly localised "obstacles" or "barriers" on the route that make a specific point of it non-accessible – for example, a stile that must be climbed over or a short steep elevations unsafe for wheelchair users. In the second case, a route may be non-accessible due to more de-localised factors, such as large overall elevation changes or high noise levels along the route.

4.4 Obstacles en-route

Localised "obstacles" make a point or section of a route completely impassible. Therefore, we decided to directly incorporate obstacle avoidance into our routing algorithm.

4.4.1 Localised obstacles from OSM data

First, we identified a list of non-accessible "obstacles" from OSM tags that could serve as an initial list of POIs to avoid along a route: `tank_trap`, `rope`, `log`, `kerb`, `jersey_barrier`, `chain`, `bollard`, `wicket_gate`, `turnstile`, `toll_booth`, `swing_gate`, `sump_buster`, `stile`, `spikes`, `sally_port`, `motorcycle_barrier`, `lift_gate`, `kissing_gate`, `kent_carriage_gap`, `horse_stile`, `height_restrictor`, `hampshire_gate`, `sliding_gate`, `gate`, `full-height_turnstile`, `entrance`, `debris`, `cycle_barrier`, `cattle_grid`, `bus_trap`,

`bump_gate`, `border_control`, `block`, `wall`, `retaining_wall`, `hedge`, `handrail`, `guard_rail`, `fence`, `ditch`, `city_wall`, `cable_barrier`.

The initial idea was to pass the list of all POIs to be avoided to the routing engine at the time of route creation so that all routes created specifically avoided these POIs.

Limitations Any POI on OSM with one of the above tags corresponds to a *node* in the OSMnx walking network. However, on OSMnx it is only possible to pass a list of *edges* to be avoided when computing a route. Therefore, one problem that was not fully resolved was how to best identify a list of edges to be avoided given a set of non-accessible nodes. Therefore, this aspect of obstacle avoidance was not integrated as part of the POC for the time being.

4.5 Detecting short steep ascents

In addition, we attempted to construct an algorithm for detecting short steep ascents along a route which might indicate staircases or any other sections of route too steep for a wheelchair user.

Each route is given as a list of OSMnx nodes and edges between them which generally represent sections of a road or path. To begin, we used the route data to generate a list of metre-spaced coordinates which trace out the given route. This can be done by extracting the geometry of each section of road using the osmnx.geometries module and interpolating along each straight subsection of road. Using LIDAR data it was then easy to detect short steep ascents which might be unsafe for wheelchair use. As a consequence, it seems that this method has the capacity to detect most staircases, which would of course also be unsuitable for wheelchair users.

4.6 Iterative procedure for avoiding localised obstacles

The way that the algorithm avoids obstacles such as very steep slopes by using an alternative street for a section means that this part of the

calculation had to be done iteratively for the case that another such obstacle was encountered on the new street. In theory, this algorithm could continue to return routes with absolute slopes exceeding a given threshold, so we capped the number of iterations to 5.

The algorithm including the slope detection and avoidance works as follows:

1. Set $i = 1$ and `routes=[]`
2. Use OSMnx to find a route between the two locations using the available network.
3. Compute POIs close to the route at 50m, 100m and 250m.
4. For each two waypoints along the route, compute the slope of the segment connecting them using the integrated LIDAR data (see §2.2.3).
5. If the absolute slope exceeds a threshold of 0.083, increase the weight of that edge to an extreme threshold (effectively removing the edge from the network). Else, go to Step 6.
6. Increment $i \mapsto i + 1$.
7. If $i < 5$, add the route to `routes` and return to Step 2.
8. Return `routes`.

4.7 Other accessibility considerations

In order to incorporate more delocalised factors such as total elevation change or high noise levels along a route, we then rank the suitable routes given by our algorithm with respect to the particular needs of each user.

4.7.1 Elevation profile

As mentioned before, the elevation profile of a route was an important factor in our routing engine, as steep elevation can be problematic for certain users. For example, for wheelchair users the recommend

Figure 11: Simulated route between the Tower of London and Waterloo station

maximum slope is no steeper than 1:12 for independent use[11]. For this reason we simulated a couple of routes from location A to B, an example of which is shown in Figure 11, and plotted the corresponding elevations in Figures 12 and 13. The x-axes of the plots shows the distance (km) between the start point and end point of the route, while the y-axes display the elevation in meters. As we can see in Figure 12 there is around a 7m increase over around a 150m. This just a slight uphill on the road compared to the second route in Figure 13, where a similar elevation is much steeper.

4.7.2 Busyness of the route

Avoiding main streets may be a need for some users, and , therefore the project also investigated how to achieve this in the route recommendation engine. Minimising the busyness of the route could be done through an iterative process. Due to time constraints in duration of the project, this function has not been implemented in the code base but the high-level concept is shown here:

1. The fastest route is calculated based on the given starting point and destination.
2. The busyness of all edges on the fastest route is assessed based on speed parameters for each edge. Any edges with excessive busyness (greater than a threshold) will be avoided in next iteration.

[11] `https://www.wheelchair-ramps.co.uk/special/maximum-slope-for-wheelchair-ramps/`

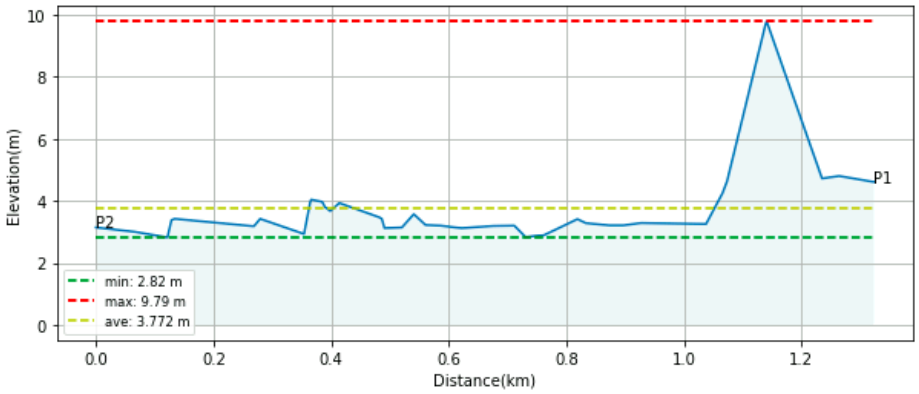


Figure 12: The elevation profile of a short simulated route. Shown are the distance in km on the x-axis and the elevation in metres on the y-axis. The red and green dotted lines indicate the maximum and minimum elevation levels reached throughout the route.

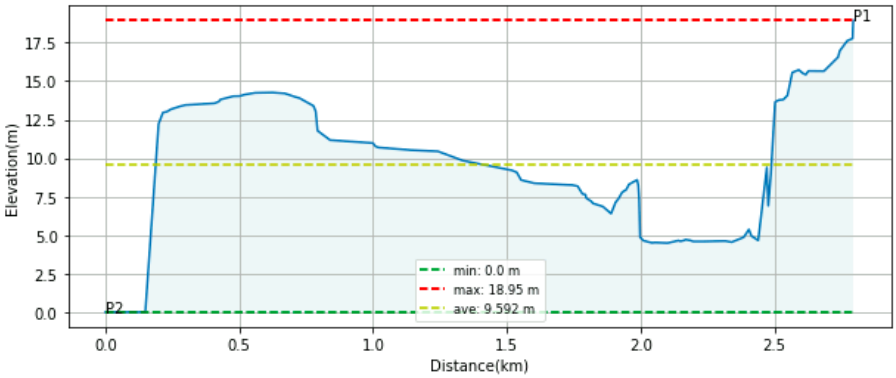


Figure 13: The elevation profile of another simulated route. Similarly to the figure above, the x-axis shows the distance in km and on the y-axis the elevation level in metres is shown.

3. Repeat step 1 and 2 to generate five routes and recommend all the five routes to the user with a busyness indicator for each of the routes.

It should be noted that assessing the busyness of a road based on the average speed on it might not be the best way to do that as this ignores

speed limits. An alternative approach would use a higher delta between speed limit and average speed as an indicator for a busy road (e.g. more slower-moving traffic even though speed limit would allow for faster travel).

4.8 Route personalisation

In order to offer personalised routes for each customer, the focus is on different end-user personas already identified by CityMaaS. Rather than deciding on a one-size-fits-all route, the idea is to use the persona concept to personalise the best route for an individual persona as different users will have different priorities and needs.

4.8.1 Features

As a POC, the routing engine has the following capabilities:

- **It computes the shortest route between any 2 points.** Currently we take into account the length of the path, but in the future this may be expanded to accommodate additional features, or to compute multi-variable parameters through which the route optimisation engine will find the best route.
- **It computes the elevation changes through the route.** Thanks to the use of LIDAR data, we can retrieve the elevation of any point in the area of interest. Therefore, we are able to compute the raw elevation hence the slope of the path, so that steep slopes can be avoided.
- **It finds nearby accessible POIs close to the route.** Leveraging the work done on the POIs prediction, we find for each route the number of accessible POIs along the route at various distances. This allows us to understand, besides the route itself, the overall accessibility of the area.
- **It optimises any route trying to flatten the slope of it.** For any found route, the routing engine looks for steep slopes along it, and if any is found, it removes them by looking for alternative routes. All

the routes are added to the catalogue, and are shown to the user so that they can find the one which suits their needs the most.

Each persona considered has therefore a constraints matrix which describes their accessibility requirements for the route recommendation in terms of the features mentioned above.

4.8.2 End-user personas

For each of the personas selected, the group members decided which features would be included with which importance, with a score out of 5 describing increasing importance depending on their accessibility requirements. It has to be noted that this scoring system carried a degree of subjectivity and potential for assumption bias. It should therefore be seen as a proof of concept with potential for improvement, for example by at a later stage including user ratings or their history in choosing a specific route out of a list of recommended ones, in order to derive the scores instead of assuming them. CityMaaS also provided us with further information on each persona to illustrate their individual needs:

- **Claudia Perth**: 44-years old wheelchair user. Her goals are to move independently and to do excursions with both her family and on her own.

Feature	Score
Avoid Steep slope	4/5
Avoid Obstacles	5/5
Avoid Main Street	2/5
Shortest Distance	2/5

Table 6: Claudia Perth constraints matrix. A high score indicates a high importance of this feature for the specific persona's needs.

- **Oscar Davies**: 78-years old pensioner who uses a walker rollator. His goals are to disrupt his daily routine, to plan his journeys thoughtfully and increase his independence and fastness.

Feature	Score
Avoid Steep slope	4/5
Avoid Obstacles	0/5
Avoid Main Street	1/5
Shortest Distance	3/5

Table 7: Oscar Davies constraints matrix.

- **Leonie Johnson**: 40-years old blind woman. Her goals are to travel independently and save. She also wants to visit more of the world and use more applications.

Feature	Score
Avoid Steep slope	0/5
Avoid Obstacles	4/5
Avoid Main Street	3/5
Shortest Distance	3/5

Table 8: Leonie Johnson constraints matrix.

- **Steven Wright**: 28-years old visually impaired or partially sighted mean. His main goals are efficient travel, easy information and to be able to use the time during travel.

Feature	Score
Avoid Steep slope	0/5
Avoid Obstacles	4/5
Avoid Main Street	3/5
Shortest Distance	3/5

Table 9: Steven Wright constraints matrix.

- **Allison Clarke**: 29-years old with motor skills learning disorder. Her main goals are short distances and to spend as much time as possible with her family.

Feature	Score
Avoid Steep slope	0/5
Avoid Obstacles	2/5
Avoid Main Street	2/5
Shortest Distance	5/5

Table 10: Allison Clarke constraints matrix.

- **John Lobster**: 15-years old student with autism. His main goals are to avoid too many people and to travel easily without a lot of planning.

Feature	Score
Avoid Steep slope	0/5
Avoid Obstacles	0/5
Avoid Main Street	5/5
Shortest Distance	0/5

Table 11: John Lobster constraints matrix.

4.8.3 Personalisation for personas

Personalisation of the route recommendation is achieved by considering the priorities of different users. This information can be obtained by user inputs or user identities. Currently, the route engine can only choose to consider or not consider certain constraints (i.e. shortest distance, slope condition, obstacle locations and busyness of streets), as shown in Table 12, and recommends the first five iterations to the user. The user is then given the independence of making an informed choice of their preferred route. A potential way to achieve further personalisation is to develop indices for each constraint and combine them together after weighting them with the constraint matrix to form an overall score for a specific route. Given a starting point and destination, 20 routes can be generated by the engine. We could then calculate the overall score of each route and recommend the top five routes to users.

Figure 14 shows the route generation process for Claudia from Fulham to the Big Ben. In this demonstration, avoiding high slope angles and

User	Steep slope	Obstacles	Main Street	Shortest Distance
Claudia Perth	Yes	Yes	No	Default
Oscar Davies	Yes	No	Yes	Default
Leonie Johnson	No	Yes	No	Default
Steven Wright	No	Yes	No	Default
Allison Clarke	No	Yes	No	Default
John Lobster	No	No	Yes	Default

Table 12: Constraint condition for personas

obstacles are both considered. The blue line is the original shortest path and in red the optimised path after 5 iterations is shown. Figure 15 shows another route demonstration for persona Oscar from Trafalgar Square to the City of London while avoiding obstacles. The orange route is the first shortest path and the red route is the final path after five iterations.

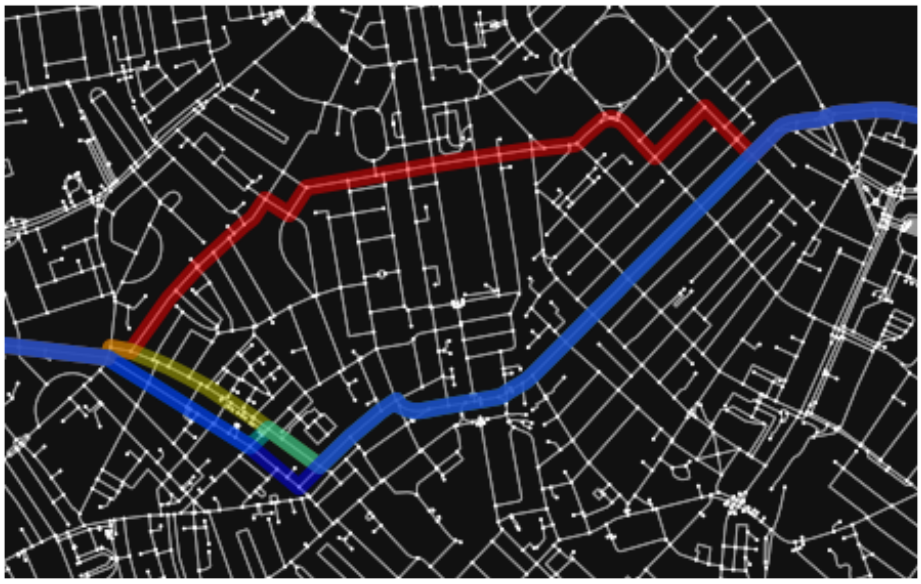


Figure 14: Route recommendation for Claudia Perth from Fulham to the Big Ben. Shown is the area where the route engine offers different options to the user, shown in different colours. The blue route is the shortest path and the red route shows a deviation from it due to Claudia's preference for avoiding steep slopes.

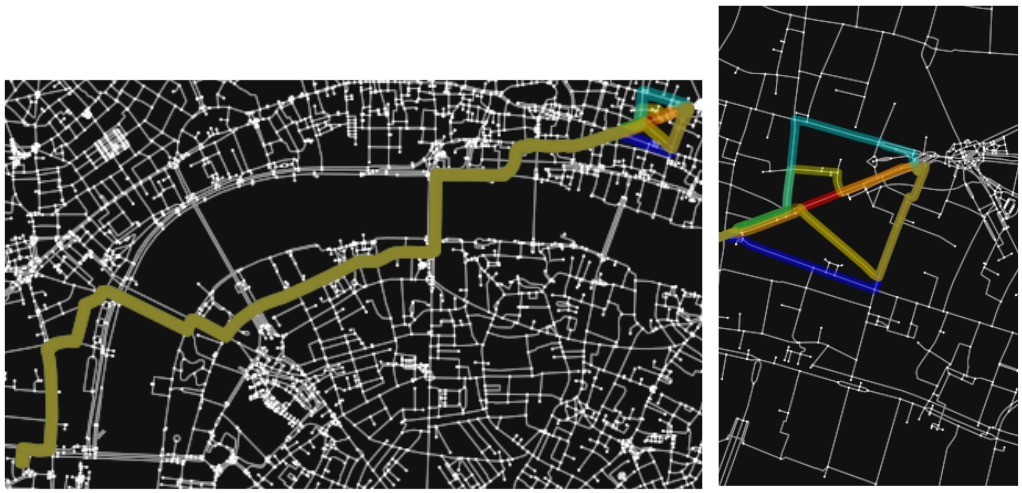


Figure 15: Route recommendation for Oscar from Trafalgar square to the City of London. Shown are the full route on the left and a zoomed in version on the right which shows different options for the user based on their individual needs.

5 Future work and research avenues

One immediate direction for future work could be integrating the POI accessibility prediction with the routing engine. One way in which this could be implemented is by assigning an “attractive” force to certain POIs either known or predicted to be accessible, so that routes prioritise passing through these points. Conversely, a “repelling” force could be assigned to certain POIs known or predicted to be non-accessible so that these nodes are avoided when routing. Here we stress that not all POIs will be relevant to this task: for example, a non-accessible restaurant that happens to occur along a particular route may not have any actual accessibility impact on that route. Therefore, a carefully selected subset of OSM tags that are deemed “relevant” to routing must be agreed upon as part of this task.

5.1 POI prediction

- Get more balanced data on POIs, or otherwise investigate whether the OSM data really reflects the true data distribution of accessible POIs. A possible question to investigate in this context is whether people are more likely to report an accessible POI on OSM than report an inaccessible one.
- Employ NLP techniques to label data, improving the homogenisation of data in the pre-processing stage.
- Use web scraping and NLP techniques to get information on reviews from online websites, such as Tripadvisor. This could be especially relevant for restaurants and similar venues but precaution should be taken to avoid data duplication.
- Use regulation on building types as a source for information for predicting accessibility, considering that some types of buildings have building code regulations and therefore must be accessible.
- Use information on how old the building is and if it has planning permission as a similar source of information.
- Link OSM with other geodata, for example from POI's websites itself to augment missing data.
- Use image data for car driving options to identify accessible information, for example disabled parking or use data stemming from external projects such as the Blue Badge Parking scheme[12].
- Use additional data on factors impacting other types of accessibility in addition to wheelchair accessibility. For example, sound level data from Sound Print[13], which builds a crowdsourced database of noisy/quite restaurants. Other noise and busyness related data includes peak time data, traffic data or pollution data.

[12] `https://bluebadgeparking.com/`
[13] `https://www.soundprint.co/`

5.2 Routing

- Recursively adjust route recommendations based on user history. This would allow to take into consideration the choices users made in the past from their list of recommended routes and also enable the algorithm to learn over time.
- Compare the results to Google's wheelchair routing algorithm to investigate whether the developed model performs better regarding one of the features implemented (slope/obstacle/busyness avoidance and/or personalisation) [14].
- Use image data for car driving options to identify accessible information, for example disabled parking. This would open up the routing engine to other modes of travel in addition to walking/wheeling.
- Include additional accessibility-relevant information for users with needs in addition to or instead of wheelchair access. For example, sound level data from Sound Print or other noise/buysness data as mentioned above.
- Routes with accessible toilets nearby. This could be especially relevant for all users of the engine for longer routes as well as users with conditions such as Crohn's disease.
- Minimise factors such as the number of different streets, number of obstacles, elevation change/steepness. Optimise time and distance improvement of suggested routes compared to original routes.
- Implement Fault Tolerant Algorithms for live updates/suggestions of the routes.
- Investigate genetic algorithms as a base for the routing task.

6 Conclusions

Accessibility while travelling poses a complex and highly personalised challenge. At the same time, addressing it is crucial for supporting and

[14] https://www.blog.google/products/maps/introducing-wheelchair-accessible-routes-transit-navig

ensuring the independence of millions in their daily lives. With our project ”Making travel for people in cities accessible through prediction and personalisation” we set ourselves two research questions to tackle in this context: How can we improve the accuracy of predicting the accessibility of POIs and how can we make routing through cities accessible and personalised for a user's individual needs.

For the first question, our proposed approach of using an XGBoost model trained on a pre-processed and augmented dataset of POIs was able to predict wheelchair accessibility of points with a precision of 0.91. It outperformed other models (SVM and RF) on the same dataset with precision scores of 0.84 and 0.90 respectively. To provide further context, a separate analysis of an RF model on a non-augmented dataset achieved a precision of 0.80. Besides making use of the OSM data, we have included further sources of information through external datasets, such as accessible toilets, which added to the predictive power of the model.

The second question was addressed through a POC algorithm which takes into account features such as the slope of a route when making recommendations. The ”routing engine” used LIDAR data to detect short steep ascents which might be unsuitable for wheelchair use, or users with reduced mobility. As a consequence, it seems that this method has the capacity to detect most staircases, which would be excluded in a hypothetical suggested route for wheelchair users. We used the steep slopes in combination with the OSM obstacles and the categorisation of road networks to develop a constraint matrix, which is used to calculate the overall score of each route and recommend an ordering of them based on the priorities of different users. This method proved to be effective in providing a personalised recommendation routing system, and is also scaleable, as the constraint matrix can be further expanded with other types of obstacles and urban infrastructure.

Furthermore, the suggestion of five possible routes to choose from and the ability of the algorithm to include additional features in the future makes our approach suitable for continued learning from user history and new data, while enabling users to make an informed decision. By using

personas of users the routing engine shows its flexibility to be adapted to different situations and individual sets of needs.

Multiple challenges and limitations were encountered during the project, which were mostly related to data. The POI dataset was skewed towards accessible POIs, which means that the model is potentially not trained on a representative dataset. Being user-provided the data also experienced high variability in the description of features, which made it necessary to pre-process them to make them more usable. The routing engine currently only takes into account walking routes, but has the potential to be extended for other modes of transport. Further research into this area is highly encouraged and potential avenues for this are clearly visible in both parts of the challenge.

7 Team members

Marta Aragones has acted as a facilitator in the CityMaaS challenge. She is a last-year PhD at the University of Cambridge working on the computational study of porous materials. Prior to her PhD, she obtained an MSc at Imperial College London and a BEng at IQS, Barcelona. Marta also has extensive experience as a teacher and mentor. During her PhD and outside of the academic environment, she has collaborated with several charities and is passionate about using data science for social good. Moving on, she is hoping to make a meaningful social impact through the application of data science and machine learning to social issues such as education and health.

Luca Bedogni has acted as a participant in the CityMaaS challenge. Luca is an Assistant Professor at the University of Modena and Reggio Emilia (UNIMORE), Italy, working on ubiquitous computing, Internet of Things and privacy issues in this domain. Prior to joining UNIMORE, Luca was an Assistant Professor at the University of Bologna, Italy. He was also a visiting researcher at the RWTH Aachen University (in 2013), at the Queen Mary University London (in 2015), and the University of California, Irvine (in 2017 and 2019).

Anurag Deshpande has acted as a participant in the CityMaaS challenge. He is a PhD researcher at University College London, and his research is focused on utilising Bayesian statistics and Machine Learning techniques to improve the accuracy of modern Cosmology. Anurag also has a keen interest in applying data science and AI solutions to inter-disciplinary problems, and has recently completed an Applied Science internship at Microsoft, developing AI-driven recommendation systems.

Antonia Gieschen has acted as a principal investigator (PI) in the CityMaaS challenge. Antonia is a doctoral researcher at the University of Edinburgh, where she is researching the development and application of cluster analysis for spatial and spatio-temporal data. She is interested in how these methods can be used by organisations and businesses from different sectors, including public health and SME financing. Her current research focus is the recovery of the Scottish tourism industry after the COVID-19 pandemic.

Zhening Huang has acted as a participant in the CityMaaS challenge. He is a PhD student at the Engineering Department, University of Cambridge. He is currently working on using 3D computer vision and deep learning to automate the process of geometric digital twin creation for the road network. His research interests include point cloud segmentation, scene graph generation, video understanding, etc.

Finlay McIntyre has acted as a participant in the CityMaaS challenge. Finlay completed a PhD in Pure Analysis at the University of Edinburgh in July 2021. His thesis focused on the study of multilinear geometric inequalities arising in Harmonic Analysis and certain underlying connections to Convex Geometry. Over the past few years he has taken a keen interest in coding using Python and C++, and hopes to explore the field of Algorithm Design and Optimisation. Currently, he is working as a tutor at AIMS Rwanda while exploring potential research topics in the use of High Performance Computing in training AI.

Shahreen Muntaha Nawfee has acted as a participant in the CityMaaS challenge. She is currently a PhD student in the School of Geography, Geology and the Environment in the University of Leicester. Her PhD project focuses on, 'diverse geographies of user-generated content.' Her research interest includes spatial data science, focusing on spatial data analysis of urban environment and also on crowd-sourced spatial data.

Flora Roumpani has acted as a principal investigator (PI) in the CityMaaS challenge. Flora is a Research Associate in REG in the Alan Turing Institute, working on urban analytics. She is also a Visiting Lecturer at the Royal College of Art in the Environmental Architecture MSc. She holds a PhD and an MRes from the Centre for Advanced Spatial Analysis in Bartlett UCL and has a diploma on Architecture Engineering from the Department of Architecture in the University of Patras.

Alex Saad has acted as a facilitator in the CityMaaS challenge. Alex is a data scientist with a particular interest in using data science to improve quality of life in urban environments. He recently completed a DPhil in number theory at the University of Oxford. Since then he has shifted careers and is working as a Senior Data Scientist at Wayfair, based in Berlin.

Reka Vonnak has acted as a participant in the CityMaaS challenge. Reka holds an MA in Social and Public Policy with Quantitative Methods and an MSc in Urban Analytics from the University of Glasgow. She currently works as a Data Scientist at Datapolis, based in Budapest. Her research interests focus on urban data science and spatial analysis.

References

[1] Geoff Boeing. "OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks". In: *Computers, Environment and Urban Systems* 65 (2017), pp. 126–139. ISSN: 0198-9715. DOI: https://doi.org/10.1016/j.compenvurbsys.2017.05.004. URL: https://www.sciencedirect.com/science/article/pii/S0198971516303970.

[2] Tianqi Chen and Carlos Guestrin. "XGBoost". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016). DOI: 10.1145/2939672.2939785.

turing.ac.uk
@turinginst