

Cardiff School of Computer Science and Informatics

Coursework Assessment Pro-forma

Module Code: CMT309
Module Title: Computational Data Science
Lecturer: Dr. Matthias Treder, Dr. Luis Espinosa-Anke
Assessment Title: CMT309 Data Science Portfolio
Assessment Number: 2
Date set: 26-02-2021
Submission date and time: 21-05-2021 at 9:30am
Return date: before 25-06-2021

This assignment is worth 70% of the total marks available for this module. If coursework is submitted late (and where there are no extenuating circumstances):

- 1 - If the assessment is submitted no later than 24 hours after the deadline, the mark for the assessment will be capped at the minimum pass mark;
- 2 - If the assessment is submitted more than 24 hours after the deadline, a mark of 0 will be given for the assessment.

Your submission must include the official Coursework Submission Cover sheet, which can be found here:

<https://docs.cs.cf.ac.uk/downloads/coursework/Coversheet.pdf>

Submission Instructions

Your coursework should be submitted via **Learning Central** by the above deadline. You have to upload the following files:

Description		Type	Name
Cover sheet	Compulsory	One PDF (.pdf) file	Student_number.pdf
Your solution to Part 1 and Part 2	Compulsory	One Jupyter Notebook (.ipynb) file	Part1_2.ipynb
HTML version of part1_2.ipynb	Compulsory	One HTML (.html) file	Part1_2.html
Your solution to Part 3	Compulsory	One Jupyter Notebook (.ipynb) file	Part3.ipynb
HTML version of part3.ipynb	Compulsory	One HTML (.html) file	Part3.html

For the filename of the Cover Sheet, replace ‘Student_number’ by your student number, e.g. “C1234567890.pdf”. Make sure to include your student number as a comment in all of the Python files! Any deviation from the submission instructions (including the number and types of files submitted) may result in a reduction of marks for the assessment or question part.

You can submit multiple times on Learning Central. ONLY files contained in the last attempt will be marked, so make sure that you upload all files in the last attempt.

Staff reserve the right to invite students to a meeting to discuss the Coursework submissions.

Assignment

Start by downloading the following files from the Assessment tab in Learning Central:

- Part1_2.ipynb (Jupyter Notebook file for Part 1 and Part 2)
- Part3.ipynb (Jupyter Notebook file for Part 3)
- listings.csv
- reviews.csv

Then answer the following questions. You can use any Python expression or package that was used in the lectures and practical sessions. Additional packages are not allowed unless instructed in the question. You answer the questions by filling in the appropriate sections in the Jupyter Notebook. Export your final Jupyter Notebooks as HTML to produce the corresponding HTML files. **Before submitting your Jupyter Notebooks and HTML files, make sure to restart the kernel and execute each cell such that all outputs and figures are visible.**

Scenario

In this assignment you slip into the role of a Data Scientist who has been hired by Airbnb. Airbnb is an online market place for vacation and short-term rentals of rooms or flats which operates in many countries in the world (see <https://en.wikipedia.org/wiki/Airbnb>). Airbnb collects data on the listing and users interacting with the platform. Let us define these terms first:

- *User*: A user is someone using the Airbnb platform (guest or host).
- *Guest*: A guest is a user who uses the Airbnb platform to book a room, flat or house.
- *Host*: A host is a user who offers a room, flat, or house for rent.
- *Listing*: A listing is a room, flat, or house offered for rent. In the dataset, each row corresponds to a listing.

Since we do not have access to Airbnb's internal database, we will instead use data published by the website *Inside Airbnb*. The data has been acquired by web-scraping publicly available data from Airbnb. It is available online (<http://insideairbnb.com/about.html>) but you **do not** need to download any data from this website since we have downloaded and renamed all datasets you need and made them available on Learning Central.

In our scenario, you are responsible for the Amsterdam branch of Airbnb operations. Your main task is to provide insights on the data collected in Amsterdam as well as write algorithms to improve the experience of Airbnb users. The assignment is split into three parts. In the first two parts, you will focus on the numerical parts of the data. In the last part, you will focus on the text data.

- **Part 1: Pre-processing and exploratory analysis**
You start by reading the csv file into a Pandas DataFrame, cleaning the data and removing unwanted columns, performing conversions and calculating new columns. Then, you will perform exploratory analysis to look at the properties and distribution of the data, and answer a couple of questions your manager put forward to you.
- **Part 2: Statistical analysis and recommender systems**
Starting from the pre-processed DataFrame, you will perform statistical analysis using t-tests and linear regression to identify variables that significantly affect the price of the rent. Then, you will design a series of recommender systems that have been requested by users: a function that helps in setting the price for someone offering a new property, and a function that helps in selecting a city to visit given a particular budget.
- **Part 3: Text analysis and ethics**
You will mostly work with unstructured text data in a Pandas Dataframe representing user reviews.

To get you started with the dataset, you can have a look at a [corresponding Jupyter Notebook](#) (note: following and working through this link is not required for this assignment, but maybe you find some inspiration or useful information there). You are allowed to re-use code from the Jupyter Notebook provided that you properly reference it.

Part 1 – Pre-processing and exploratory analysis [25 marks]

You answer this question by filling in the first part of the *Part1_2.ipynb* Jupyter Notebook.

Question 1a – Drop columns (Total 4 marks)

When reading in the dataframe using the `load_csv` function, one can see that it contains a lot of textual data which will not be relevant for the numerical analyses in Part 1 and Part 2. Therefore, implement two functions `drop_cols` and `drop_cols_na` which remove some of the columns. Detailed instructions: [2 marks each]

- `drop_cols(df)` : takes the dataframe as an input. It returns the reduced dataframe after dropping the following columns:

```
'scrape_id', 'last_scraped', 'description', 'listing_url', 'neighbourhood', 'calendar_last_scraped', 'amenities', 'neighborhood_overview', 'picture_url', 'host_url', 'host_about', 'host_location', 'host_total_listings_count', 'host_thumbnail_url', 'host_picture_url', 'host_verifications', 'bathrooms_text', 'has_availability', 'minimum_minimum_nights', 'maximum_minimum_nights', 'minimum_maximum_nights', 'maximum_maximum_nights', 'minimum_nights_avg_ntm', 'maximum_nights_avg_ntm', 'number_of_reviews_l30d', 'calculated_host_listings_count', 'calculated_host_listings_count_entire_homes', 'calculated_host_listings_count_private_rooms', 'calculated_host_listings_count_shared_rooms'
```
- `drop_cols_na(df, threshold)` : drop columns according to the amount of NaN values they contain. `threshold` is a fraction between 0 and 1. If the fraction of NaNs in a column is equal or larger than the threshold, the respective columns is dropped. For

instance, if threshold is 0.5, all columns that have at 50% or more NaNs are dropped. The default value for threshold is 0.5.

To solve the question, complete the functions in Question 1a in the notebook.

Question 1b – Recode and add columns (Total 6 marks)

We continue pre-processing by recoding some of the columns and adding columns with additional information. To this end, implement the following functions:

`binary_encoding`, `add_host_days`, `convert_price`.

Detailed instructions:

[2 marks each]

- `binary_encoding(df)`: if we have a close look at some of the columns, we can see that some of them are binary variables representing 1 (True) and 0 (False) values. However, the values are encoded as the strings 't' (for True) and 'f' (for False). Recode these columns by turning them into the integer numbers 0 and 1.
- `add_host_days(df)`: it would be useful to have a column that represents the number of days (with respect to the current date) that the host has been registered. To this end, create a new column called 'host_days' (hint: look into Pandas' `to_datetime` method).
- `convert_price(df)`: the 'price' column represents the nightly price in USD. However the prices are encoded as strings. This function should convert the prices into floating point numbers. For instance, the string '\$40' should be converted to the floating point number 40.0.

To solve the question, complete the functions in Question 1b in the notebook.

Question 1c – Answering questions (Total 6 marks)

Your manager has a couple of questions about the dataset, and provided you with a list of questions they want answered. The questions are as follows:

[1 mark each]

- How many hosts offer two or more properties for rent?
- What is the highest price for a listing?
- What is the ID of the listing that has the largest number of bedrooms?
- What is the ID of the listing with the largest advertised price?
- There are different room types. How many listings are there for the most common room type?
- How many hosts are there that have been registered for more than 3000 days?

To solve the questions, provide corresponding Pandas code in Question 1c in the notebook.

Question 1d – Exploratory analyses (Total 9 marks)

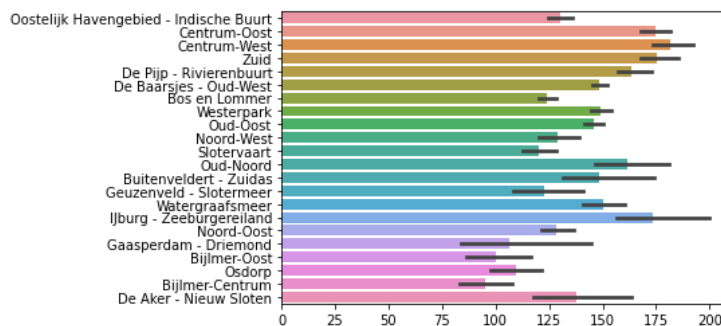
To further explore the dataset, you produce a number of exploratory plots. In particular, you set out to produce the following three plots:

[3 marks each]

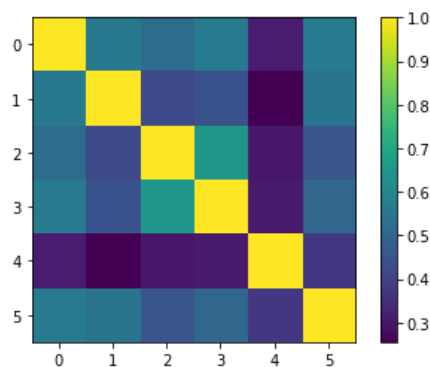
- Plot 1: A barplot with horizontal bars representing average nightly prices. Produce a bar for each neighbourhood (use the `neighbourhood_cleansed` column).
- Plot 2: The review ratings have 5 additional subitems: cleanliness, checkin, communication, location, value. You are interested in the correlations between the subitems. To this end, produce a correlation matrix that depicts all pair-wise Pearson correlations between these 5 variables.
- Plot 3: Your manager is interested in the geographical distribution of nightly prices for the more expensive listings. To this end, produce a scatterplot using latitude/longitude as coordinates. The price should be encoded both by color and size of the circles. Make sure to include only those listings with a price larger than \$150.

The following figures are indications of how such plots might look like. Note that title, labels etc are missing. It is just intended for orientation. Your solution can have a different style, colors, etc.

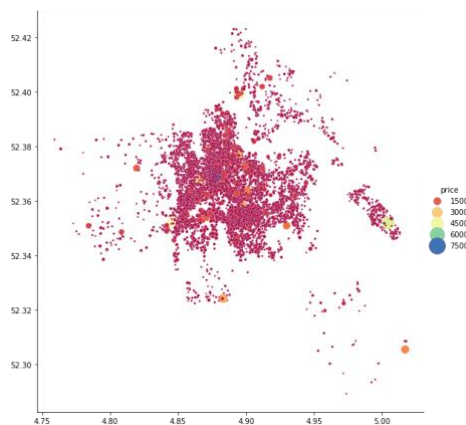
Plot 1:



Plot 2:



Plot 3:



Part 2 – Statistical analysis and recommender systems [35 marks]

You answer this question by filling in the second part of the *Part1_2.ipynb* Jupyter Notebook.

Question 2a – Linear regression and t-tests (Total 8 marks)

In this question you will perform a linear regression and a number of t-tests. Detailed instructions: [4 marks each]

- **Linear regression:** Review scores consist of an overall rating as well as additional scores on subitems (accuracy, cleanliness, checkin, communication, location, value). Can the overall rating be predicted from the scores on the subitems? To investigate this, perform a linear regression on rating using statsmodels. After fitting the model, print the summary. Make sure that the variable names shown in the summary are short and readable (accuracy, cleanliness, checkin, communication, location, value).
- **T-tests:** You want to investigate whether the difference in prices for different room types is significant. To investigate this, perform independent samples t-tests for the prices of each combination of room types using $\alpha = 0.01$. Which room types are significantly different in terms of room type? Do your conclusions change if you perform Bonferroni correction to the alpha level (see https://en.wikipedia.org/wiki/Bonferroni_correction). Finally, create a Dataframe with 4 rows and 4 columns that holds the p-values for all pairwise combinations. The column/row names should be the different room types, and each entry in the dataframe corresponds to the p-value for the respective combination of room types.

To solve this question, provide corresponding code in Question 2a in the notebook and provide short answers in the space designated as YOUR ANSWER.

Question 2b – Linear regression with variable selection (Total 15 marks)

In this question you investigate how well nightly price can be predicted from the other variables in the dataframe. You need to decide yourself which variables to choose, but make sure you have at least 10 variables. The only requirement is that you use `room_type` as a predictor. Because `room_type` is a categorical variable, you first have to use dummy coding to turn it into a number of binary variables (hint: `pd.get_dummies()`). In the notebook, provide a short explanation for your choice of variables.

Starting from the variables you have chosen, our goal is to derive a sparse model with fewer variables. This process is called *variable selection*. In variable selection ('variable' means the same as 'predictor'), variables get iteratively added or removed from the regression model. Once finished, the model typically contains only a subset of the original variables. It makes it easier to interpret the model, and in some cases it makes it generalise better to new data. To perform variable selection, implement a function

```
variable_selection(df, predictors, target, alpha)
```

where `df` is the listings dataframe, `predictors` is a list with your initial selection of at least 10 variables (e.g. ['bedrooms', 'beds', ...]), `target` is the target variable for the regression (e.g. 'price'), and `alpha` is the significance level for selecting significant predictors (e.g. 0.05). The function returns `pred`, the selected subset of the original `predictors`.

To calculate regression fits and p-values you can use `statsmodels`. Your approach operates in two stages: In stage 1, you build a model by adding variables one after the other. You keep adding variables that increase the [adjusted R² coefficient](#). You do not need to calculate it by hand, it is provided by `statsmodels` package. In stage 2, starting from these variables, if any of them are not significant, you keep removing variables until all variables in the model are significant. The output of the second stage is your final set of variables. Let us look at the two stages in detail:

Stage 1 (add variables) [8 marks]

- Start with an empty set of variables
- Fit multiple one-variable regression models. In each iteration, use one of the variables provided in `predictors`. The variable that leads to the largest increase in adjusted R² is added to the model.
- Now proceed by adding a second variable into the model. Starting from the remaining variables, again choose the variable that leads to the largest increase in adjusted R².
- Continue in the same way for the third, fourth, ... variable.
- You are finished when there is no variable left that increases adjusted R².

Stage 2 (remove non-significant variables) [7 marks]

It is possible that some of the variables from the previous stage are not significant. We call a variable "significant" if the p-value of its coefficient is *smaller or equal to* the given threshold alpha.

- Start by fitting a model using the variables that have been added to the model in Stage 1.
- If there is a variable that is not significant, remove the variable with the largest p-value and fit the model again with the reduced set of variables.
- Keep removing variables and re-fitting the model until all remaining variables are significant.
- The remaining significant variables are the output of your function.

To solve this question, provide corresponding code in Question 2b in the notebook and provide a short answer in the space following YOUR ANSWER. To test your function, add a function call with your selection of predictors and alpha level.

Question 2c – Recommender systems (Total 12 marks)

There have been requests from customers to provide automated recommendations. First, guests requested a recommender system that helps them identifying neighbourhoods in a city that fit their budget. Second, hosts who want to offer new listings would like a recommender

system that suggests a nightly price. As a response to this request, you and your team have worked out specifications for two recommender systems.

Recommender system 1: Recommend a neighbourhood given a budget [6 marks]

Guests who are traveling on a budget have been requesting a tool that allows them to quickly see which neighbourhood in a city offers most accommodation opportunities within their budget bracket. You want to implement a Python function that delivers this functionality. The plan is to integrate the Python function into the Airbnb website. After some deliberation, you work out that the function should meet these specifications:

- The user should be able to specify their budget bracket, that is, a minimum and maximum budget. For instance, a user might look for properties priced in the \$10-\$50 range. Another user looking for luxury accommodation may opt for a \$100-\$500 range.
- Your function identifies which neighbourhood has the largest number of properties within this range. It returns a string representing the name of the neighbourhood. Use the `neighbourhood_cleansed` variable for the names of the neighbourhoods.
- Some neighbourhoods have more properties than others, so by considering only *absolute numbers* neighbourhoods with more properties could always be preferred by your algorithm. An alternative is to consider *relative numbers*, that is, the proportion of listings within the budget bracket relative to the total number of listings within a given neighbourhood. The user should be able to select whether they want absolute or relative numbers.

From these specifications, you arrive at the following function signature:

```
recommend_neighbourhood(df, budget_min, budget_max, relative)
```

with `df` being your listings dataframe, the variables `budget_min` and `budget_max` being floating point numbers representing the budget bracket. The numbers are inclusive, i.e., a nightly price exactly equal to `budget_min` or `budget_max` is considered, too. The variable `relative` is a Boolean specifying whether relative numbers (fractions) should be considered in the recommendation. If False, absolute numbers are considered instead.

Recommender system 2: Price recommender for hosts [6 marks]

If a new host wants to offer their room / flat / house on Airbnb, they need to decide on what the nightly price will be. There is no official guidance but hosts have been requesting for Airbnb to provide an algorithm. After some deliberation, you work out that the function should meet these specifications:

- The user has to provide the geolocation (latitude and longitude) of their property.
- Your algorithm searches for the geographically closest properties (simply measured by Euclidean distance in terms of latitude/longitude) that are already listed on Airbnb. Your price recommendation will be the mean of the nightly prices of these closeby properties.
- The user should be able to set the number of closeby properties (called neighbours) that are considered. A larger number of neighbours indicates a larger geographical area.
- You can ignore the fact that some neighbours could be in a different neighbourhood.

- Nightly prices are quite different for different room types. The user should be able to set the desired room type. If the room type is defined, only properties of the respective room type are taken into consideration.

From these specifications, you arrive at the following function signature:

```
recommend_price(df, latitude, longitude, n_neighbours, room_type)
```

with the variables being `latitude` and `longitude` representing geolocation of the property, `n_neighbours` the number of neighbouring properties the user wants to take into account. `room_type`, if specified, restricts the neighbours search to properties of the given room type; it should default to `None` which means that any property type is considered.

To test your two recommendation system, provide function calls for each of the two functions. You can freely select the parameters of the function call.

To solve this question, provide corresponding code in Question 2c in the notebook.

Part 3 – Text analysis and ethics

[40 marks]

You answer this question by filling in the Part3.ipynb Jupyter Notebook. In this part, you will be working with the `reviews.csv` file providing reviews for the listings, and more specifically, the ‘comments’ column.

Question 3a – Pointwise Mutual Information (Total 30 marks)

In this question, you implement and apply the **pointwise mutual information (PMI)** metric, a word association metric introduced in 1992, to the Airbnb reviews. The purpose of PMI is to extract, from free text, pairs of words than tend to co-occur together more often than expected by chance. For example, $\text{PMI}(\text{'new'}, \text{'york'})$ would give a higher score than $\text{PMI}(\text{'new'}, \text{'car'})$ because the chance of finding ‘new’ and ‘york’ together in text is higher than ‘new’ and ‘car’, *despite ‘new’ being a more frequent word than ‘york’*. By extracting word pairs with high PMI score in our reviews, we will be able to understand better how people feel and talk about certain items of interest (e.g., ‘windows’ or ‘location’).

The formula for PMI (where x and y are two words) is:

$$\text{PMI}(x, y) = \log \frac{p(x, y)}{p(x)p(y)}$$

Watch [this video](#) to understand how to estimate these probabilities.

Your solution will involve the following steps:

1. (4 marks) Processing the raw reviews, applying the following steps in this specific order:
 - a. **Tokenize** all reviews. Use `nltk’s word_tokenize method`.
 - b. **Part-of-speech (PoS) tagging**: to be able to differentiate nouns from adjectives or verbs. Use `nltk’s pos_tag method`.

- c. **Lower case:** to reduce the size of the vocabulary.

What to implement: A function `process_reviews(df)` that will take as **input the original dataframe** and will **return it with three additional columns**: `tokenized`, `tagged` and `lower_tagged`, which correspond to steps **a**, **b** and **c** described above.

2. (5 marks) Starting from the output of step **1.c** (`tokenized`, PoS-tagged and lower cased reviews), create a vocabulary of ‘center’ (the **x** in the PMI equation) and ‘context’ (the **y** in the PMI equation) words. Your vocabulary of center words will be the 1,000 most frequent NOUNS (words with a PoS tag *starting with ‘N’*), and the context words will be the 1,000 most frequent words tagged as either VERB or ADJECTIVE (words with any PoS tag *starting with either ‘J’ or ‘V’*).

What to implement: A function `get_vocab(df)` which takes as **input the DataFrame generated in step 1.c**, and **returns two lists**, one for the 1,000 most frequent center words (nouns) and one for the 1,000 most frequent context words (either verbs or adjectives).

3. (8 marks) With these two 1,000-word vocabularies, create a co-occurrence matrix where, for each center word, you keep track of how many of the context words co-occur with it. Consider this short review with only one sentence as an example, where we want to get co-occurrences for **verbs** and **adjectives** for the center word **restaurant**:

- a. ‘A **big** restaurant **served** **delicious** food in **big** dishes’
 `>>> {'restaurant': {'big': 2, 'served': 1, 'delicious': 1}}`

What to implement: A function `get_coocs(df, center_vocab, context_vocab)` which takes as **input the DataFrame generated in step 1**, and **the lists generated in step 2** and **returns a dictionary of dictionaries**, of the form in the example above. It is up to you how you define *context* (full review? per sentence? a sliding window of fixed size?), and how to deal with exceptional cases (center words occurring more than once, center and context words being part of your vocabulary because they are frequent *both as a noun and as a verb*, etc). Use comments in your code to justify your approach.

4. (3 marks) After you have computed co-occurrences from all the reviews, you should convert the co-occurrence dictionary as a pandas DataFrame. The DataFrame should have 1,000 rows and 1,000 columns.

What to implement: A function called `cooc_dict2df(cooc_dict)`, which **takes as input the dictionary of dictionaries generated in step 3** and **returns a DataFrame** where each row corresponds to one center word, and each column corresponds to one context word, and cells are their corresponding co-occurrence value. Some (x,y) pairs will never co-occur, you should have a 0 value for those cases.

5. (5 marks) Then, convert the co-occurrence values to PMI scores.

What to implement: A function `cooc2pmi(df)` that **takes as input the DataFrame generated in step 4**, and **returns a new DataFrame** with the same rows and columns, but with PMI scores instead of raw co-occurrence counts.

6. (5 marks) Finally, implement a method to retrieve context words with highest PMI score for a given center word.

What to implement: A function `topk(df, center_word, N=10)` that **takes as input: (1) the DataFrame generated in step 5, (2) a center word** (a string like 'towels'), and **(3) an optional named argument called N with default value of 10**; and **returns a list of N strings, in descending order of their PMI score with the center_word**. You do not need to handle cases for which the word `center_word` is not found in `df`.

Question 3b – Ethical considerations (10 marks)

Local authorities in touristic hotspots like Amsterdam, NYC or Barcelona regulate the price of recreational apartments for rent to, among others, ensure that fair rent prices are kept for year-long residents. Consider your price recommender for hosts in Question 2c. Imagine that Airbnb recommends a new host to put the price of the flat at a price which is above the official regulations established by the local government. Upon inspection, you realize that the inflated price you have been recommended comes from many apartments in the area only being offered during an annual event which brings many tourists, and which causes prices to rise.

In this context, critically reflect on the compliance of this recommender system with **one of the five actions** outlined in the **UK's Data Ethics Framework**. You should prioritize the action that, in your opinion, is the weakest. Then, justify your choice by critically analyzing the three **key principles** outlined in the Framework, namely *transparency*, *accountability* and *fairness*. Finally, you should propose and critically justify a solution that would improve the recommender system in at least one of these principles. You are strongly encouraged to follow a scholarly approach, e.g., with peer-reviewed references as support.

Your report should be between 500 and 750 words long.
Write your answer in Part3.ipynb (under Q3b).

Learning Outcomes Assessed

- Carry out data analysis and statistical testing using code
- Critically analyse and discuss methods of data collection, management and storage
- Extract textual and numeric data from a range of sources, including online
- Reflect upon the legal, ethical and social issues relating to data science and its applications

Criteria for assessment

Credit will be awarded against the following criteria. Different criteria are applied to Pandas code (using Pandas outside of a function), function code, and figures obtained with matplotlib or seaborn. Pandas code (e.g. in Question 1c) is exclusively judged by its functionality. Functions are judged by their functionality and additionally their quality will be assessed. Figures are judged by their quality and completeness. The ethics question is judged by its quality. The below table explains the criteria.

Pandas code	Mark	Functionality (100%)
	Distinction (70-100%)	Fully working code that demonstrates an excellent understanding of the assignment problem using a relevant python approach
	Merit (60-69%)	All required functionality is met, and the code is working probably with some minor errors
	Pass (50-59%)	Some of the functionality developed with incorrect output and major errors
	Fail (0-50%)	Faulty code with wrong implementation and wrong output

Functions	Mark	Functionality (80%)	Quality (20%)
	Distinction (70-100%)	Fully working function that demonstrates an excellent understanding of the assignment problem using relevant python approach	Excellent documentation with usage of <code>__docstring__</code> and comments, overall clean code and notebook
	Merit (60-69%)	All required functionality is met, and the function is working probably with some minor errors	Good documentation with minor missing of comments
	Pass (50-59%)	Some of the functionality developed with incorrect output and major errors	Fair documentation
	Fail (0-50%)	Faulty function with wrong implementation and wrong output	No comments or documentation at all

Figures	Mark	Quality and completeness (100%)
	Distinction (70-100%)	Excellent figure depicting the quantity of interest, with complete and informative data and formatting, labels, titles, and legends if appropriate
	Merit (60-69%)	Good figure with good formatting, labels, titles, and legends
	Pass (50-59%)	Acceptable figure with missing information, bad formatting, labels, titles, or legends
	Fail (0-50%)	Faulty or missing figure

Ethics	Mark	Quality (100%)
	Distinction (70-100%)	One action addressed and critically justified and all principles addressed, in both cases with extensive use of supporting peer-reviewed references. Solution is justified extensively with references and empirical data.

	Merit (60-69%)	One action addressed and critically justified. All principles addressed. Some supporting peer-reviewed references. Justified solution based on limited literature and/or empirical data.
	Pass (50-59%)	One action addressed without proper justification. Some principles addressed. No or trivial solution provided.
	Fail (0-50%)	Fails to address the actions and principles in the UK Data Ethics Framework. No or trivial solution provided.

Feedback and suggestion for future learning

Feedback on your coursework will address the above criteria. Feedback and marks will be returned within 4 weeks of your submission date via Learning Central. In case you require further details, you are welcome to schedule a one-to-one meeting.