6G6Z0048 Artificial Intelligence: 1CWK100

Delete all the reminder text in grey and all the notes in black before you submit.

Important: These notes present some specific points of guidance on how to go about meeting the requirements of the assignment specification in your written report; they address commonly asked questions (integrating answers from the FAQ on Moodle) and are hopefully useful if you’re unsure on how to get started with any aspect of the report. However, following all the suggestions *is not mandatory* and you’re free to address the assessment criteria as you see fit.

Notes:

* The word counts for each section below are guides only – hitting the overall word count mandated in the unit specification (4,000) is the important thing
* If you feel you need to use more words in one section and reduce the number of words in one or more other sections, that’s okay and up to you to judge
* You have flexibility in the range +/-10% (3,600-4,400 words), and also the potential to reduce the number of words you need by including other resources (e.g., images, figures, references, etc., see also section 3 of the assignment specification)
* Remember that captions for resources like figures, images and tables (i.e., the text immediately beneath them) don’t count towards your overall word count
* Your list of references at the end, and your appendices, also do not count towards your overall word count

# 1) Longlisting [~1,500 words]

“[…] *identifying a range of relevant algorithms, building blocks and techniques from AI – briefly explaining them, and their potential links to image classification problems*”

Notes:

* This section should be a chance to build up marks quickly
* Start off by listing in bullet points if you like (and come back and re-write into prose later on)
* “[…] algorithms, building blocks and techniques” is just a quote from the unit’s learning objectives (written by someone else) intended to cover all the constituent parts, or “components”, of any overall technical recommendation you might make (there’s a worked example below); I’ll just say “components” for short from here on in these notes
* Each item in this longlisting section should therefore be a named “component” along with a “brief explanation” of what it is, and its “potential links to image classification problems” (as in the guidance from the assignment specification, reproduced in grey above)
* In many cases writing down these three things is likely to be a challenge to do in much less than 1-2 short sentences, so you should find that you quickly grow your word count
* You should not include any components that are not potentially relevant to the problem (you are likely to end up needing all of the word count, even if it doesn’t feel like that initially)
* You could try and put together an initial version of this longlisting section today, right now, regardless of where you’re up to with lecture slides/your understanding, and you can update/maintain it gradually, as you go through more slide sets and readings during the weeks between now and the submission deadline
* The lecture slides usually name technical components inside “quotes”, so they should be relatively easy to spot (it’s up to you to think about which ones are relevant to this problem), but note that some other things that are not components are also inside quotes; e.g., important concepts or terminology (see the worked example below) so you do need to be thinking, and mentally filtering, as you review slides
* I had a look through at the first two (“recipe”) slide sets as an example, and made a little table from what I found, which you can see below; I included some components that I might want to mention in the first column (not yet filtered by whether they are relevant to the problem or not) and some concepts and terminology that might be useful to me when doing my “brief explanations” and “potential links” in the second and third columns; I did it quickly as a hopefully useful example; if you think something is missing, or is better moved into another column, I’m happy to agree with your judgement

|  |  |  |
| --- | --- | --- |
| Potentially useful components (or “algorithms, building blocks and techniques”) | Potentially useful concepts (for explanations and linking) | Potentially useful terminology (for explanations and linking) |
| Datasets, cleaning, conversion, splitting, shuffling, stratified sampling, holdout method, training/testing/validation data, repeated holdout, k-fold cross-validation, accuracy, loss, confusion matrices, MSE, TPR, FPR, ROC curves | Supervised learning, unsupervised learning, semi-supervised learning, self-supervised learning, data collection, data exploration  and preparation, model training, model evaluation, model improvement, abstraction, generalisation, regression, classification, parameters, hyperparameters, noise, over/underfitting, bias-variance tradeoff, probability/likelihood, data leakage | Observations, features, feature values, predictive features, target features, numerical/categorical features, examples, labels, values, data, fitting phase, prediction phase, class probability/likelihood |

Table 1: This is the table I ended up with after reviewing the two “recipe” slide sets for words inside “quotes” – it won’t be perfect and is intended as an illustrative example – but hopefully you can imagine producing something similar for later lectures if you wanted to; note that quite a few of the components came out of the critical analysis we did at the end of the second lecture, and hopefully you will have extracted some original components for yourself in the critical analyses you’ve done for later lecture topics during the A-labs.

* Remember, using the terminology we defined on the unit is completely optional, as long as you write clearly and consistently that’s fine; similarly, there is no requirement to use any of the notation or visual representations we’ve used on the unit, unless you want to
* Notice that I quickly ended up with lots of potentially relevant components that weren’t models…
* Many of the subsequent lectures introduced specific models, but we still met some new components as we worked (e.g., z-score standardisation in the k-NN lecture to pick a random early example[[1]](#footnote-2)), you may also have come across extra components as you did your own critical analyses in the A-labs[[2]](#footnote-3), and we also met lots of hyperparameters, many of which are relevant components in and of themselves
* Regarding individual hyperparameters, the thing to consider is whether you have an *option* about using them/whether you could use something else instead; if it’s just an inescapable property of a model component you’ve already included (e.g., the value of k in k-NN) then it’s probably not worth mentioning by itself[[3]](#footnote-4), but if it’s something you can switch on or off, or replace with another component that does a similar job, or use across different models, then it’s potentially relevant, and worth including
* As you go deeper in your thinking over time (particularly in your work on the analysis section, below), this longlisting section has the potential to become long; try to do some sensible structuring/grouping and editing as you work on it – for example, can you include closely related components together as a group (e.g., a short list of optimisers for gradient descent, or activation functions for use in artificial neurons, or handcrafted features for use with images, or methods for clustering in unsupervised learning, or image datasets to use in training, or deep networks for use in transfer learning, or models to use in regression/classification, etc., etc.), together with a single overall explanation of how they work, and their relevance to the problem
* This section should not include any critical analysis (strengths/weaknesses) – you can leave all that for the next section
* If you’re going for top marks then your aim in this section should be to include everything relevant that we’ve covered on the unit; missing a small number of components isn’t something to worry about (we’ll be flexible when marking), but if you’re aiming for the very top marks then you would typically be expected to have identified *at least some* relevant components for yourself, through your own self-directed research (e.g., in your A-lab work) – so be ambitious (rather than worried about missing a small detail in a slide somewhere)

For this section you need to name an algorithm, give a brief explanation of it and then link it back to how it works with image classification. There is some underneath, pick which ones you know and then do a bit more research on them.

name – brief explanation – link back to image classification

Supervised learning

1. K- nearest neighbour classification KNN is a simple, instance-based learning algorithm that classifies objects based on the majority class of their k-nearest neighbours in the feature space. Link to Image Classification: KNN can be applied to image classification by representing images as feature vectors and classifying them based on the classes of their nearest neighbours.
2. Feature space visualisation Space visualization allows you to represent images as points in a multi-dimensional feature space. Benefits: This representation helps in understanding the distribution and relationships between different features extracted from images, facilitating feature engineering and selection for improved classification performance.
3. Naïve baye classification – from a machine learning algorithm that extracts features from images such as colour histograms and texture features. It's efficient and simple to implement and allows for easy interpretation of the results. However, it’s not suitable for advanced data as it doesn’t understand or represent those relationships and isn’t good at capturing spatial and visual features.
4. Decision tree classification - an algorithm that splits the datasets into subsets based on features and uses the features to make decisions about which class they belong to. There are two types of nodes, one is an internal node, which represents the decisions, and a leaf node. Which represents the class. They’re easy to interpret, is easy to see because of the tree structure, they don't make assumptions, can handle non-linear relationships, as well as handling missing datasets, by using the available data to make its decisions and can easily integrate with ensemble methods. They can automatically learn features and handle numerical and categorical data. May not capture complex relationships as well as deep learning method would.
5. Ensemble classification Ensemble Methods - Description: Ensemble methods combine multiple models to improve overall performance and generalization. Link to Image Classification: Techniques like bagging (e.g., Random Forests) or boosting (e.g., AdaBoost) can be applied to combine the outputs of multiple classifiers, enhancing accuracy and robustness.
6. Regression Regression involves specifying relationship between one numeric dependent variable (value being predicted) and one or more numeric independent variables (predictors). Regression methods used for statistical hypothesis testing, which determine if premise is likely to be true of false considering observed data. Most common approach to modelling numeric data, adapted to any modelling task, estimates size and strength of relationships among features and outcome. Makes strong assumptions about data, models form needs to specify in advance, doesn’t handle missing data, only works with numeric features so categorical data requires additional preparation, requires knowledge of statistics to understand.
7. Logistic Regression classification term "regression," logistic regression is actually a classification algorithm, particularly suitable for binary classification problems. When extended for multiclass classification, it is often used in a one-vs-rest or one-vs-one strategy. Logistic regression is computationally efficient and easy to implement. It does not require extensive computational resources, making it suitable for large datasets and real-time applications. Logistic regression does not handle missing data well. Imputation methods may be needed, and the impact of missing data should be carefully considered.
8. Artificial Neural Network - also known as ANN. It's a model that has been inspired by the functioning of the human brain, and how our brain uses sensory outputs to respond to stimuli. ANNs use multiple layers to output classes. It's very commonly used in image classification, can easily capture advance relationships. requires large datasets and the hyperparameters need tuning as it’s quite intensive.
9. Support Vector machines - an algorithm associated with regression and is used to differentiate between classes using the features of the image. Support vector machines use multidimensional surfaces to define the relationship between features and outcomes. It extracts features from images and tries to find a hyperplane between the classes. Effective for tasks with a large number of features, works well with small datasets, easily adaptable with almost any type of learning task. They're very intensive, so they're harder to use in larger datasets which also makes it less scalable, they don’t provide probability estimates.
10. Feature extraction Feature Extractor: A component responsible for extracting relevant features from input images. Examples: Convolutional Neural Networks (CNNs) are widely used as feature extractors in image classification tasks due to their ability to capture hierarchical and spatial features.
11. Unsupervised learning – this is when an algorithm has a dataset with no instructions on what to do. It tries to create the relationships with no guidan ce, which is the opposite of supervised learning. Although it’s not widely used in image classification, certain aspects of it apply to image classification. There is a technique known as feature learning, where the algorithm finds features from an image, which are then used for classification. Unsuperivised learning is good at detecting anomalies in a dataset, a technique called clustering identifies similarities between images, which helps get a better understanding of the data. This is used with larger datasets. Because the algorithm has no instructions on what to look for, it may bring back results the user isn’t looking for, isn’t good at hyperparameters, can’t handle complex relationships.
12. Recurrent Neural network - also known as RNN. It's a type of architecture that is used to store images in an order. It's commonly used to handle sequential data. When it comes to image classification, it takes the images and outs them in a sequence. RNNS can stay hidden while still gaining information regarding the image sequence and can handles sequences of different lengths. Overfitting can occur with smaller datasets, harder to train compared to CNNs, as the hyperparameters need to be tuned.
13. Attention and Transformers attention mechanisms allow models to focus on specific parts of the input while processing information, giving more importance to relevant regions. Image class: allows model to concentrate on important regions of the image, enabling it to recognise patters or features critical for classification. Enhance model’s ability to capture long-range dependencies and improve performance when dealing with complex, large data. Transformers are a neural network. Cause attention mechanisms to process input data in parallel, making them highly scalable and efficient. Image class: transformers can capture complex relationships between pixels and recognise patterns in an image. Allow parallelisation, making it suitable for processing large amounts of data efficiently. Highly adaptable to different inputs, making it versatile.
14. Reinforcement Learning software agent makes observation and acts within an environment. The environment rewards them, so they know their action was good. Image class: RL is controlling where an image classification system should focus its attention. RL allows models to learn to classify images and use the information in broader decision- making contexts. Continuously adapts strategies, making them suitable for changes in data. Computationally intensive (not suitable for founder 3) and require substantial computational resources.
15. Batch Normalisation Batch Normalization - Description: Batch normalization normalizes the inputs of a neural network layer, improving training stability and convergence. Link to Image Classification: Batch normalization is commonly used in CNNs to accelerate training and improve generalization. Batch Normalization: Normalizing the inputs of a neural network layer to stabilize and speed up the training process. Importance: Batch normalization helps in mitigating internal covariate shift and improving model convergence.
16. Dataset Dataset: A collection of labelled images used for training and evaluating the image classification model. Importance: High-quality, diverse, and representative datasets are essential for training models that can generalize well to new, unseen images.
17. Neural Architecture Search Neural Architecture Search (NAS): NAS involves automating the design of neural network architectures to find optimal structures. Application: NAS can be used to search for architectures that are well-suited for specific image classification tasks, potentially outperforming manually designed architectures.
18. k-fold cross-validation used to assess the performance and generalisation ability of a model, including image classification. Data is split into K non-overlapping subsets. Model is trained K times, and each time 1 K subset is used as validation set. More reliable estimate of model’s performance, provides insight into how model’s performance varies across different subsets of data, helping to identify potential data-specific issues. Computational costs.
19. K- means clustering. Unsupervised learning. Groups data into clusters based on similarity.
20. Bagging Bagging – generates new datasets on original training data, then used to generate models using single learning algorithms. Used with relatively unstable learners (models change when input data changed slightly) which is essential for ensuring diversity: Decision trees.
21. Boosting - improves/boosts performance of weak learners to attain performance of a stronger leaner. Image class: can use boosting to improve the accuracy of classification of the image.
22. Random forests Random Forests – like bagging but adds diversity to decision tree by only allowing algorithm to choose from randomly selected subset of features each time it attempts to split.
23. Gradient boosting – gradient boosting is an ensemble technique. It predicts the classes by building decision trees, and with each tree the errors from the trees before are corrected. This is thought to improve optimisation., they are very accurate and maintain their accuracy even with more complex relationships, it highlights the importance of certain relevant features. Can be sensitive to certain hyperparameters, so the correct one must be chosen, struggles with the handling of imbalanced datasets, consume a lot of memory if there's a lot of trees.
24. Convolutional neural network (CNN) Convolutional Neural Networks (CNNs) - CNNs are specialized neural networks designed for processing grid-like data, such as images. Link to Image Classification: CNNs are the backbone of modern image classification systems. They use convolutional layers to extract hierarchical features from images, enabling them to learn spatial hierarchies of features.

# 2) Analysis [~2,000 words]

“[…] *critically analysing the suitability of (combinations of) options set out in (1) for adoption by the company, based on:*

1. *the contextual information surrounding this specific image classification problem (see* ***Appendix A****);[[4]](#footnote-5)*
2. *the theoretical characteristics of the various options set out in (1) and/or your own experimental investigations into the options set out in (1) based on a suitable dataset/(s)*”

Notes:

* This section is where you analyse the components you identified in the previous section, by presenting their relevant strengths and weaknesses
* Start off by just trying to make a couple of points for each component; e.g., one strength and one weakness (hopefully you can draw upon your A-lab work)
* Again, fine to start by listing in bullet points if you like (and come back and re-write into prose later on)
* There will typically be some things it’s possible for you to say about any one component, based on your understanding of the theory behind how it works (e.g., “X is sensitive to the random initialisation because…”, or “X will be slower the more training examples we have because…”, or “X is faster the more classes we are dealing with because…”, or “X is more challenging in terms of interpretability because…”)
* However, you should *also* be linking your analysis to the specific problem you are dealing with (e.g., “…this wouldn’t be suitable given the dataset is X”, or “…this would be good given the number of predictive features is likely to be X”, or “….this would be acceptable given the computational resources available via X”, or “…this would be a good option given stakeholder #’s/the # stakeholder group’s wish that X”)
* Linking to the details of the problem, *as well as* to your understanding of the theory is the linking required by sections (a) and (b) (as in the guidance from the assignment specification, reproduced in grey above)
* Remember that if you prefer to show that something is true experimentally, by writing some code (e.g., to generate some timings, or performance measures, or a graph or table, etc., etc.) instead of trying to explain why it’s true based on your knowledge of the theory, then that’s fine too – see also the “and/or” in (b) (above)
* As you start to develop this section further, there should be opportunities to structure it, and bring together related discussions; e.g., often you will be considering directly competing components or related and/or inter-dependent components (but it’s fine to start off relatively unstructured, in order to get going, and gradually to add structure as you are re-drafting the section over the weeks between now and the submission deadline)
* This section should not include any “weighing” of the various points you make, or any final judgements about which components should be adopted – you can leave all that for the next section
* This section also has the potential to become long (particularly as you grow a good longlisting section, see above) and you may have to be strategic about the points you can cover in your various analyses; make sure you are presenting the key points which will underpin the reasoning behind your final recommendation (in the next section)
* If you’re going for top marks then your aim in this section should be to present some technical analysis of *all* the relevant methods identified in an equally high-quality longlisting section (see the earlier guidance), and for that analysis to give at least some coverage to *all* the information that’s provided in the assignment specification

For this section you need to pick which algorithms from the first section are most suited for image classification, pick like 2-4? And then compare them and compare their advantages and disadvantages in image classification. You need to link them to the founders as well, some basic info underneath about the founders, more info in the specification.

Which longlisted model is most suited?

Habiba – CNN, SVM, ensemble

Ramisha – CNNS gradient boosting(ensemble technique), supervised learning

List good and bad that matter to the 2 founders you ae thinking about

Talk about the founders

Founder 1 – efficient(lots of users); simple (implement in-house), good open source options

Founder 2 – accuracy not critical (only good top 10)

Founder 3 – minimise computer requirement on backend (push to user devices?)

Founder 4 – consistent user experience regardless of object/ appearance (balanced classes?)

# 3) Recommendation [~500 words]

“[…] *drawing on (2) to present a conclusion that argues for a single overall approach to the problem that you believe the company should pursue, and giving your reasons why. No single solution is perfect and this will involve acknowledging weaknesses as well as highlighting strengths.*”

Notes:

* This section is where you pick up on the points you made in the analysis section (above), to make final judgements about which components to adopt
* Your need to pick out a *specific* recommendation on the way forward for the company – this is likely to combine together a number of different components, but there should be no optionality between components; e.g., if you think two different components could both work well within your recommendation, you still need to make a decision about which one to propose[[5]](#footnote-6); there are lots of different criteria to think about in the problem scenario (e.g., accuracy is only one (relatively small?) aspect…) and reaching a unique and fully justified final recommendation, with no optionality, should definitely be possible
* In terms of structure, you might start by stating your overall recommendation, relatively briefly, and then work through the justifications underpinning it, step by step
* It’s fine to write in the first person if that helps “I think that…”, and again it may be helpful to start with bullet points (and come back and re-write into prose later on)
* Reaching a final recommendation is likely to involve making some tricky “judgement calls”, where you have to decide between two or more components that have finely balanced advantages and disadvantages; doing this shouldn’t involve introducing any new analysis – you should already have set out the various considerations in the previous section and it should be a case of briefly dipping into the key points from your earlier analysis, and using them to justify final decisions (e.g., “W and X were both strong contenders in term of Y but overall I propose adopting X because of the importance of Z”, or “X would have been ideal in this regard, as per the earlier analysis of Y, but unfortunately I judge that Z precludes its use here”, or “the options are difficult to separate, but given stakeholder #’s/the # stakeholder group’s wishes regarding X, Y is the sensible choice”)…
* You don’t have to make the overall recommendation sound perfect – there isn’t a perfect solution – and it’s expected that you will need to acknowledge some weaknesses identified in the previous section, as well as highlighting strengths (as in the guidance from the assignment specification, reproduced in grey above)
* You should state any obvious implications of your technical recommendations for the company / individual stakeholders / stakeholder groups; you can “disappoint” certain stakeholders or stakeholder groups if you feel that you need to, but your reasons for doing so should be clear (e.g., something else took priority); as you read over your final recommendation and justifications, you should be able to mentally “tick off” each of the stakeholders/stakeholder groups mentioned in the assignment specification, and satisfy yourself that you’ve fully considered the context surrounding the problem as well as the problem itself
* If you’re aiming for top marks then your aim in this section should be to set out a complete and comprehensive recommendation, which fully addresses all aspects of the problem specification (both core technical challenge and surrounding context), and which is fully reasoned/justified based on an equally high-quality analysis section (and, therefore, longlisting section; see also the guidance above)

For this section you need to pick which one you would recommend to the founders, how and why you came to this recommendation.

Which options are the best?

Which one would you recommend?

And how did you decide?

Final decision should be decision trees – this is because they already have set words for the images so they can use those words to create a decision tree to classify each image.

# Reference list

This section should include any references cited in the body of your report. Remember you should be using Cite Them Right Harvard: [library support](https://www.mmu.ac.uk/library/referencing-and-study-support/referencing/cite-them-right-harvard).

Lantz, B. (2023) *Machine learning with r: Learn techniques for building and improving machine learning models, from data preparation to model tuning, evaluation, and working with Big Data*. Birmingham, UK: Packt Publishing.

In text citation: (Lantz, 2023)

Géron, A. (2023) *Hands-on machine learning with scikit-learn, Keras, and tensorflow: Concepts, tools, and techniques to build Intelligent Systems*. Sebastopol, California: O’Reilly Media.

In text citation: (Géron, 2023)

Notes:

* This section doesn’t count towards your overall word count

# Appendix 1: LLM conversations

This section should include a straight text-based copy/paste (not images/screengrabs) of any large language model (LLM) conversations you have had regarding the assignment. Formatting doesn’t matter – this section is expected to be long and relatively poorly formatted; all that matters is that the raw text (not images of the text, or similar) is included.

Notes:

* This section is covered by the marking criteria
* This section doesn’t count towards your overall word count
* Approximately 150 people will have conversations about this year’s assignment with LLMs and submit text via this appendix; LLM output is semi-stochastic, but far from unique; it’s much better to include your conversations here for markers to see how you’ve developed your thinking, and for you to explicitly ensure there is no verbatim overlap with the prose you present above
* Directly copying LLM output and omitting it from your appendix is not a good approach, and any verbatim overlap between your written prose in sections 1-3 and someone else’s appendix is likely to lead to serious problems for you (see the guidance on plagiarism, collusion and duplication in section 7 of the assignment specification)

If you spoke to a large language model for help you need to put screenshots of your conversation here since you can’t really reference it.

# Appendix 2: Source code

This section should include a straight text-based copy/paste (not images/screengrabs) of the full source code for any experimental results you present in the main body of the report. Formatting doesn’t matter – this section is expected to be long and relatively poorly formatted; all that matters is that the raw text (not images of the text, or similar) is included.

Notes:

* This section is covered by the marking criteria
* This section doesn’t count towards your overall word count
* If you are working in Colab and you would like to share links to your original notebooks so that markers have the ability to test and recreate your results, that would be welcome; but only a copy/paste of the raw code (as above) is *required*

If you did some coding with datasets you need to put the screenshots here.

1. All the examples given in these notes are chosen randomly; they’re not intended to be significant, or to indicate things that you are *expected* to include – all decisions about relevance are for you [↑](#footnote-ref-2)
2. If that hasn’t happened yet, you may find it happens as you work on the analysis section for this assignment – you can keep this section under constant review as you are redrafting, gradually adding more components you want to consider [↑](#footnote-ref-3)
3. Though hopefully you will have some component(s) relating to the process of hyperparameter tuning in general [↑](#footnote-ref-4)
4. *Note that trying to write section 2 without first reading, and thinking carefully about, the various company information and stakeholder perspectives in* ***Appendix A*** *is likely to seriously limit your mark. Spend time reading the appendix as your first step.* [↑](#footnote-ref-5)
5. Some people have worried this means you can’t include an ensemble in your recommendation… It’s absolutely fine to recommend a (specific) ensemble as your model component… What you can’t do is to recommend multiple different *alternative* model components (ensembles or otherwise) that you think are all likely to be good choices – you need to pick *one* and justify your decision [↑](#footnote-ref-6)