Using Machine Learning Tools

University of Adelaide

2022

Revision Session

5 Key Elements of UMLT

Summary of Assessed Quiz 2

Summary of Assignment 2

Pointers for Assignment 3

5 Key Elements of UMLT

1. Generalisation and Unbiased Estimates
2. Working with Datasets
3. Performance Measures
4. Learning Curves
5. Optimisation of Hyperparameters / Model Selection

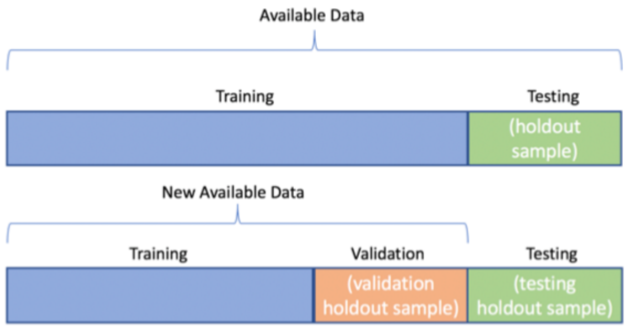
1 - Generalisation and Unbiased Estimates

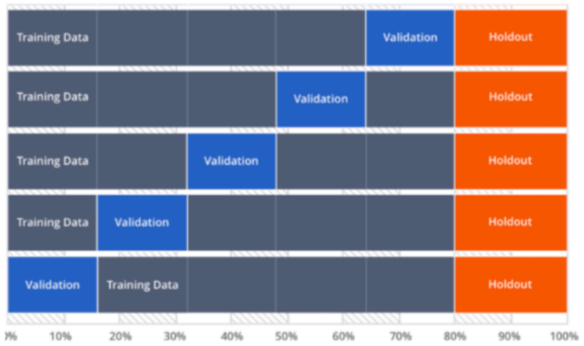
- Most real world project failures are caused by not getting proper unbiased estimates of the ability to generalise to new data

- This is what the **test set** results should provide
  - but it is very important to ***never*** **do a *comparison* using the test set**
  - it is comparisons (selections, choices) between alternatives that induces bias
  - **leave the test set until the end**
    - **if** you need to redo something, then go back to using the validation set for selection and forget/delete all previous test set results — only then can you reuse the test set for the selected model to get an unbiased result

- Selection of validation sets — fixed or cross validation (CV) — is about this too

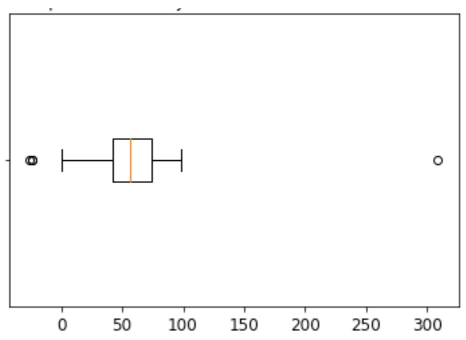
- Using pipelines is also about avoiding bias in pre-processing

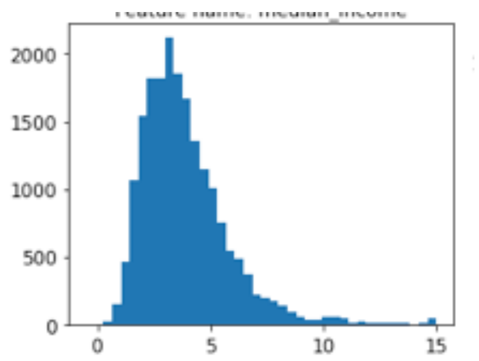


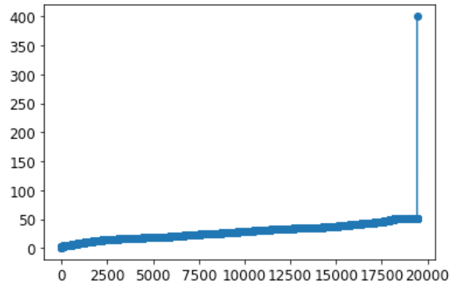


2 - Working with Datasets

- Data is often not perfect so we must:
  - Check it
    - graphical and text-based summaries
    - look at some examples (good and bad)
  - Fix it
    - find errors and fix them (often set to NaN for scalar data)
    - **imputation**
  - Use pre-processing
    - StandardScaler is OK, but RobustScaler is often better
    - MinMaxScaler is prone to problems unless fixed range (when it is best)
    - Use a pipeline except if it is fixed scaling (e.g. divide by 255 in our DL)







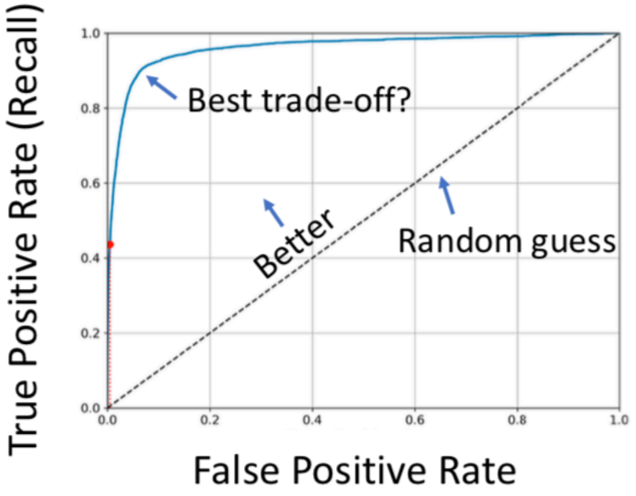
- Splitting into separate datasets
  - Use **stratify** if you have labels (it is good for all circumstances)
  - Either fixed or CV (not both) is fine for validation set
  - Test set should be fixed in all cases

- The size, accuracy and richness of the dataset you work with is hugely important
  - Biases in performance can be due to biases in the dataset too

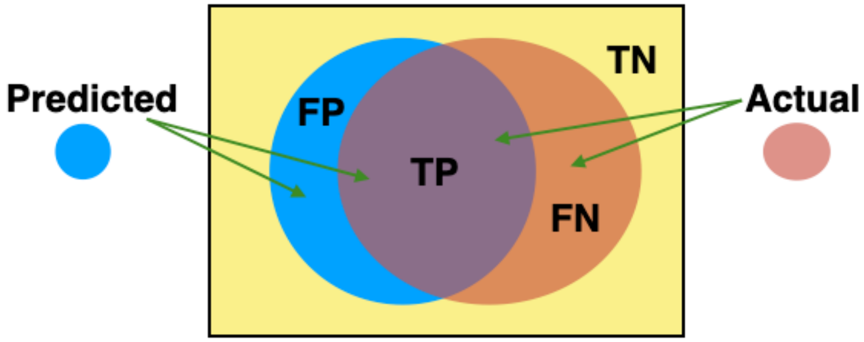
3 - Performance Measures

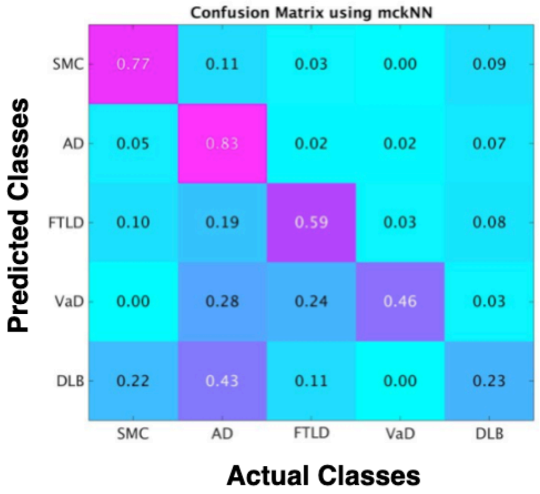
Main ones are …

- Regression:
  - (root) mean squared error (RMSE, MSE)
  - mean absolute error/deviation (MAE, MAD)
  - combination of these (elastic net)



- Classification
  - accuracy
  - AUC
  - F$_\beta$ scores
  - cross-entropy
  - confusion matrices





Confusion Matrix using mckNN

Predicted Classes \ Actual Classes	SMC	AD	FTLD	VaD	DLB
SMC	0.77	0.11	0.03	0.00	0.09
AD	0.05	0.83	0.02	0.02	0.07
FTLD	0.10	0.19	0.59	0.03	0.08
VaD	0.00	0.28	0.24	0.46	0.03
DLB	0.22	0.43	0.11	0.00	0.23

- Some are best for driving internal optimisations (e.g. MSE, cross-entropy)
  - Look at what is typical in other implementations (e.g. documentation, blogs, papers with code) or just try it and see
- Some are best for evaluating/monitoring performance (accuracy, AUC, conf mat)
  - These should be aligned with what you can interpret and what is important for the client/end-user

4 - Learning Curves

- Under-fitting:
  - “poor” performance on training dataset (and hence all others)
    - try to fix with a different / more complex model
- Over-fitting:
  - “large” gap between training and validation performance
    - try to fix by regularisation, alternative model (or augmentation, dropout, …)
- Both of these require subjective judgement

- The learning curves allow you to check:
  - learning rate
  - number of epochs (or early stopping behaviour)
  - stability

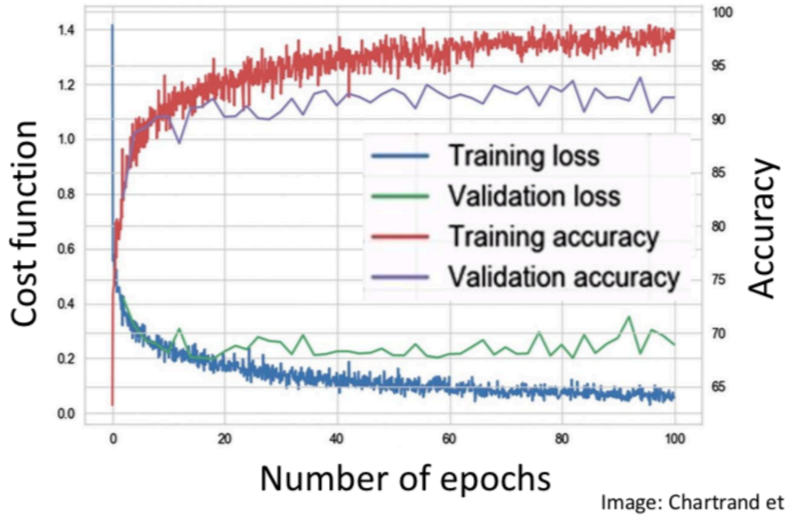
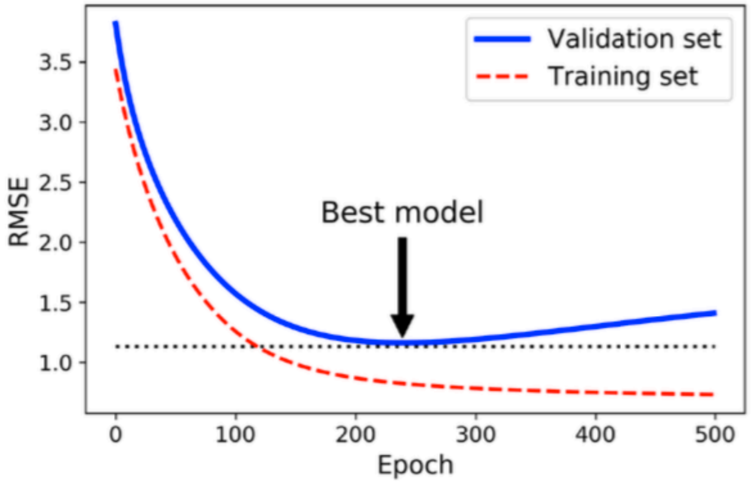
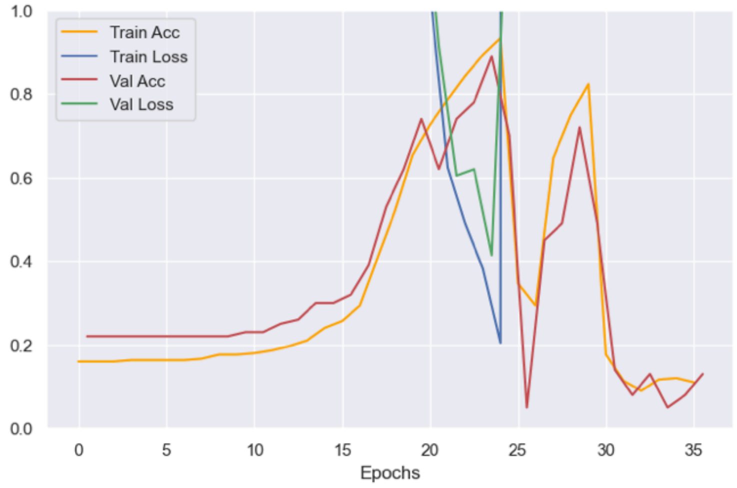


Image: Chartrand et al. RadioGraphics 2017

- Without curves you can still use final values (training, validation) to do most of the same things (main exception = stability)





5 - Optimisation of Hyperparameters or Model Selection

- A parameter is optimised **within** a method
- A hyperparameter is optimised outside
- Any difference in any setting / hyperparameter / architecture should be considered a separate model

- Options include:
  - brute force (grid search)
  - random search
  - single parameter searches (similar in concept to ablation studies)
- In each case evaluate on the **validation set**
  - best mean validation result is usually selected
  - never use the test set for choosing the best model

- Learning curves only tell you about internal parameter optimisation

- If the final result is not good enough, then you can repeat with different models/ choices (e.g. based on over/under-fitting observations) but should **not** try to optimise the test set results, as then it is biased!

Summary of ML Workflow

1. Load data, check data, fix data, split data, setup pre-processing pipeline
2. Choose a performance metric (possibly one internal and one for monitoring)
3. Establish a baseline (either very simple or a first-guess method)
4. Perform hyperparameter optimisation (evaluate with the validation set only)
5. Does it look like it will be good enough? If not, repeat step 4 with more models.
6. Measure unbiased performance with the test set

7. If you have to revisit this (e.g. new conditions, features, criteria) then repeat steps 2 onwards, and delete all records of your original result from step 6 (pretend it never happened)

Summary of Assessed Quiz 2

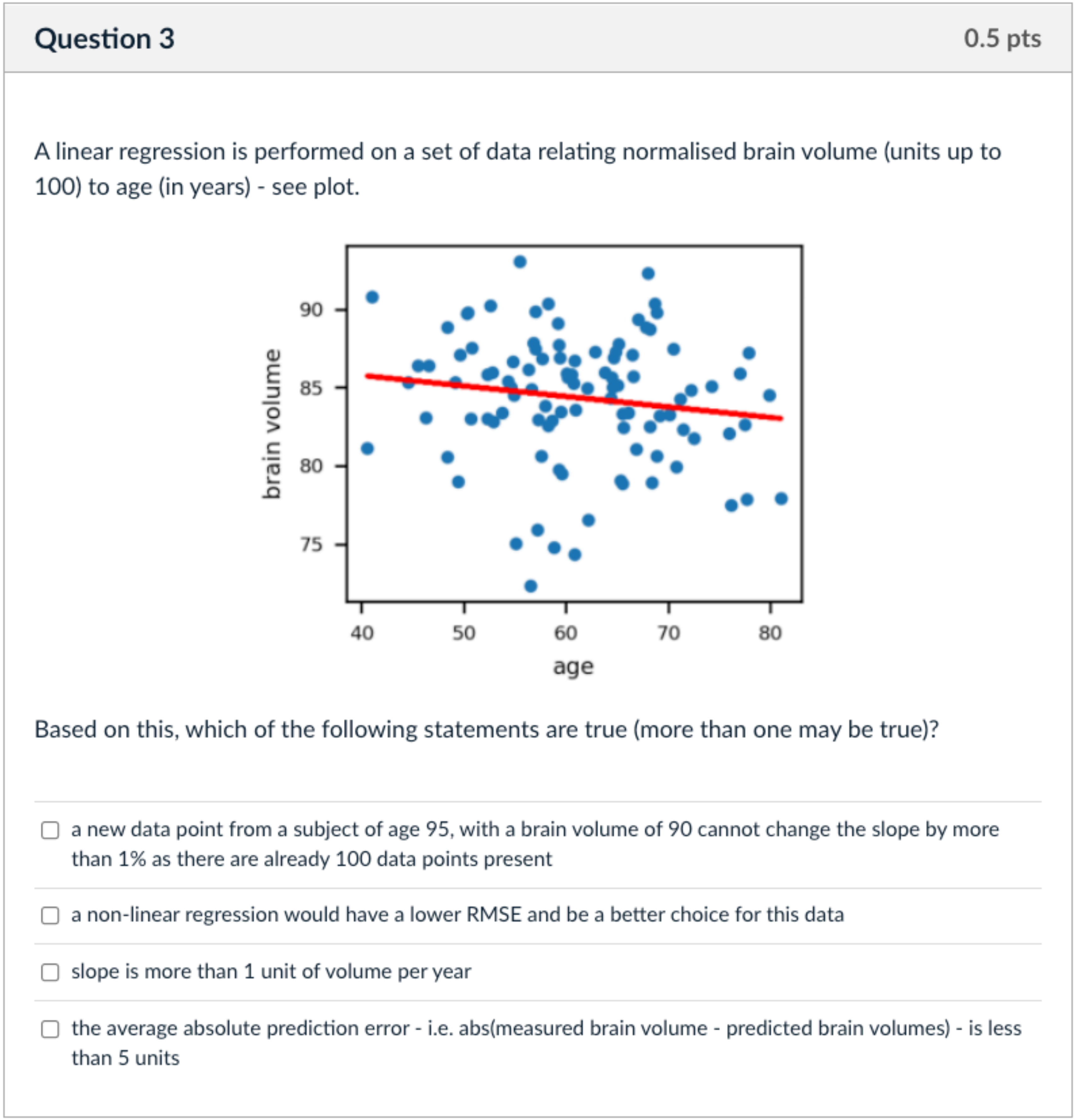
- This was a challenging quiz, with many multiple answer questions

  - Q1 (checking data) = good
  - Q2 (scaling & pipelines) = very good
  - Q3 (linear regression)
  - Q4 (cross validation) = very good
  - Q5 (GridSearchCV output)
  - Q6 (confusion matrix)
  - Q7 (PCA) = very good
  - Q8 (repeat analysis)
  - Q9 (unbiased performance) = very good
  - Q10 (DL parameters) = very good

Summary of Assessed Quiz 2

- This was a

  - Q1 (c
  - Q2 (s
  - Q3 (I
  - Q4 (c
  - Q5 (C
  - Q6 (c
  - Q7 (F
  - Q8 (r
  - Q9 (u
  - Q10



Question 3 0.5 pts

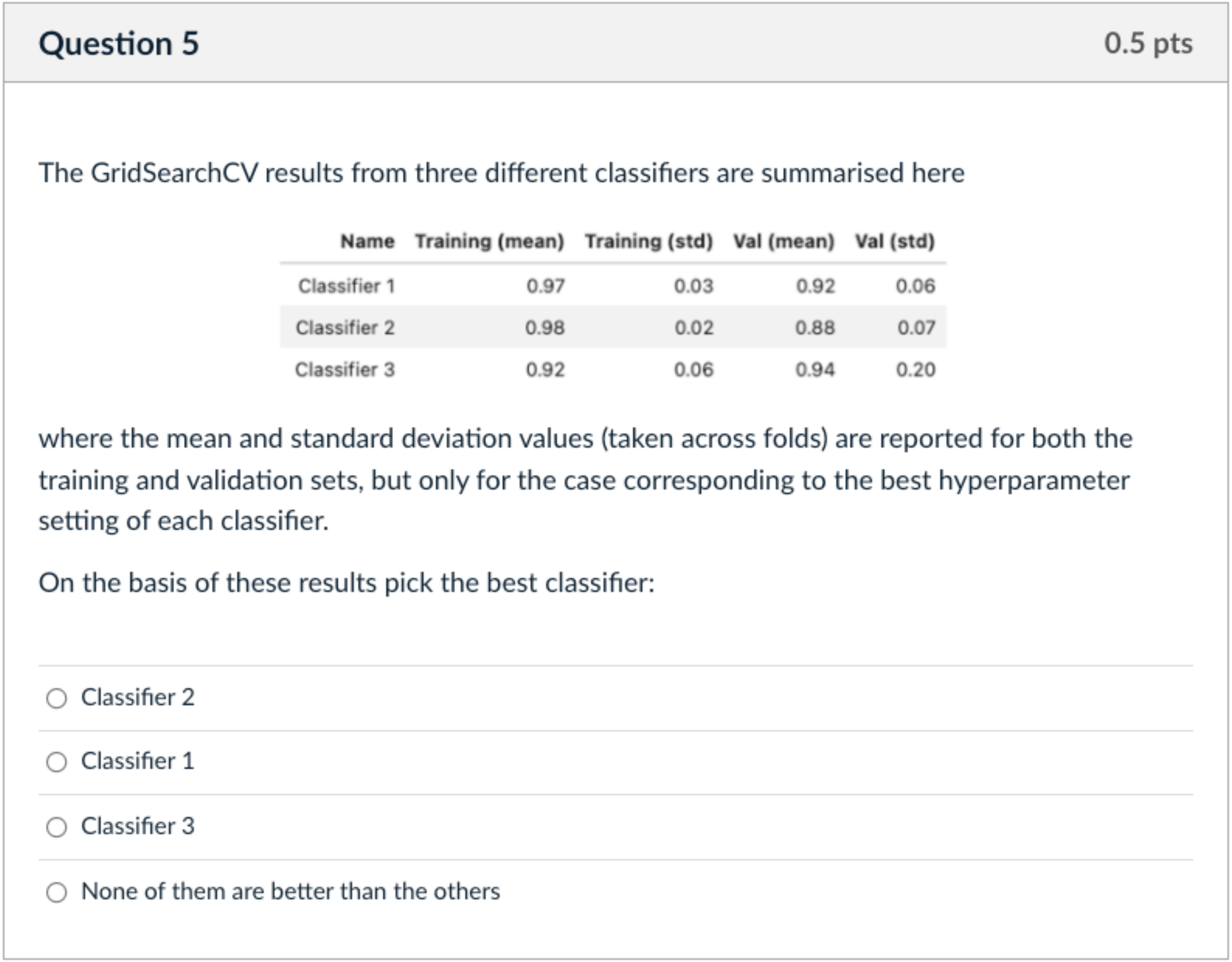
A linear regression is performed on a set of data relating normalised brain volume (units up to 100) to age (in years) - see plot.

Based on this, which of the following statements are true (more than one may be true)?

- ☐ a new data point from a subject of age 95, with a brain volume of 90 cannot change the slope by more than 1% as there are already 100 data points present
- ☐ a non-linear regression would have a lower RMSE and be a better choice for this data
- ☐ slope is more than 1 unit of volume per year
- ☐ the average absolute prediction error - i.e. abs(measured brain volume - predicted brain volumes) - is less than 5 units

Summary of Assessed Quiz 2

- Thi



Question 5 0.5 pts

The GridSearchCV results from three different classifiers are summarised here

Name	Training (mean)	Training (std)	Val (mean)	Val (std)
Classifier 1	0.97	0.03	0.92	0.06
Classifier 2	0.98	0.02	0.88	0.07
Classifier 3	0.92	0.06	0.94	0.20

where the mean and standard deviation values (taken across folds) are reported for both the training and validation sets, but only for the case corresponding to the best hyperparameter setting of each classifier.

On the basis of these results pick the best classifier:

○ Classifier 2

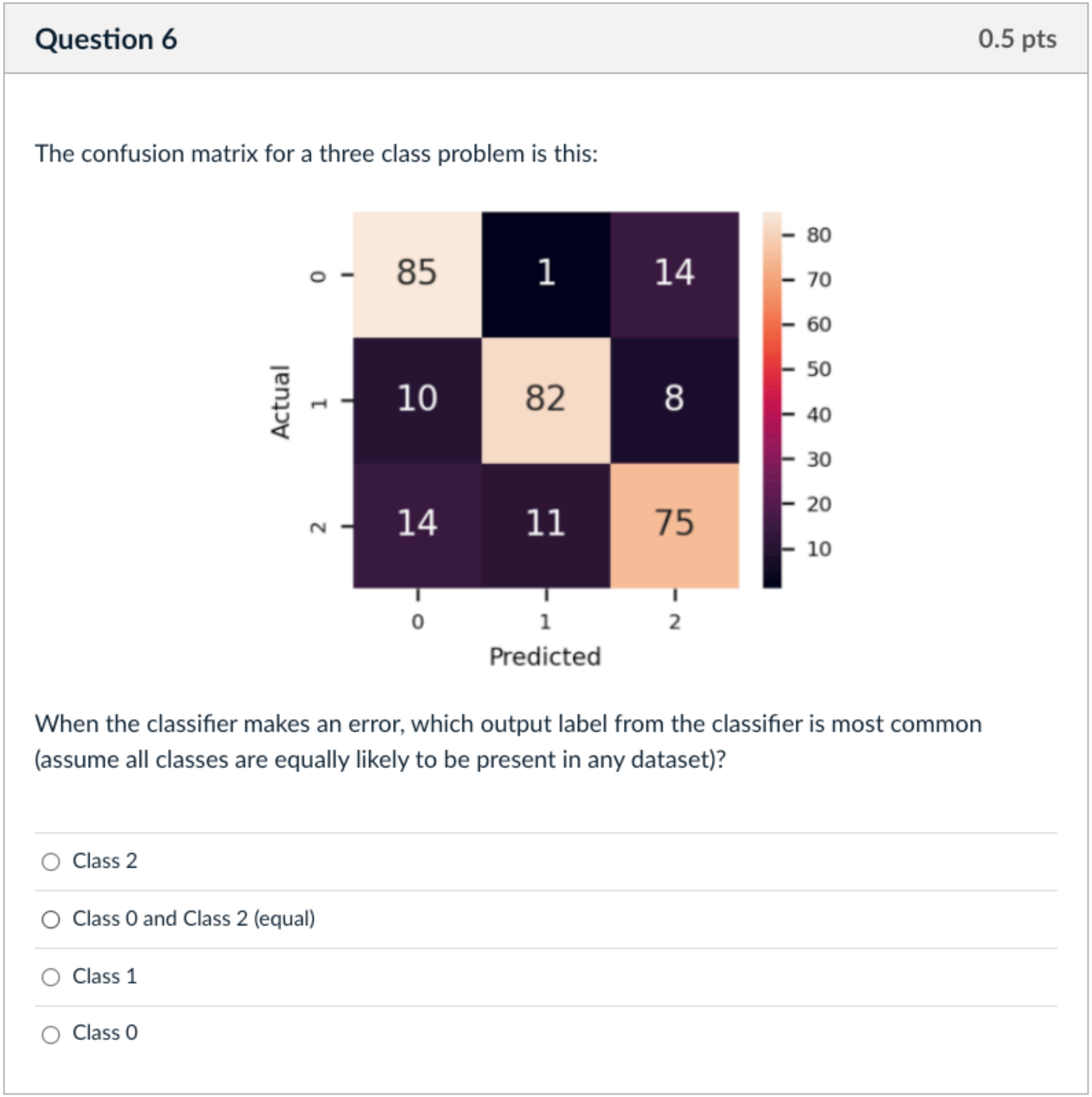
○ Classifier 1

○ Classifier 3

○ None of them are better than the others

Summary of Assessed Quiz 2

- This was
  - Q1
  - Q2
  - Q3
  - Q4
  - Q5
  - Q6
  - Q7
  - Q8
  - Q9
  - Q10



Question 6 0.5 pts

The confusion matrix for a three class problem is this:

Actual \ Predicted	0	1	2
0	85	1	14
1	10	82	8
2	14	11	75

When the classifier makes an error, which output label from the classifier is most common (assume all classes are equally likely to be present in any dataset)?

○ Class 2

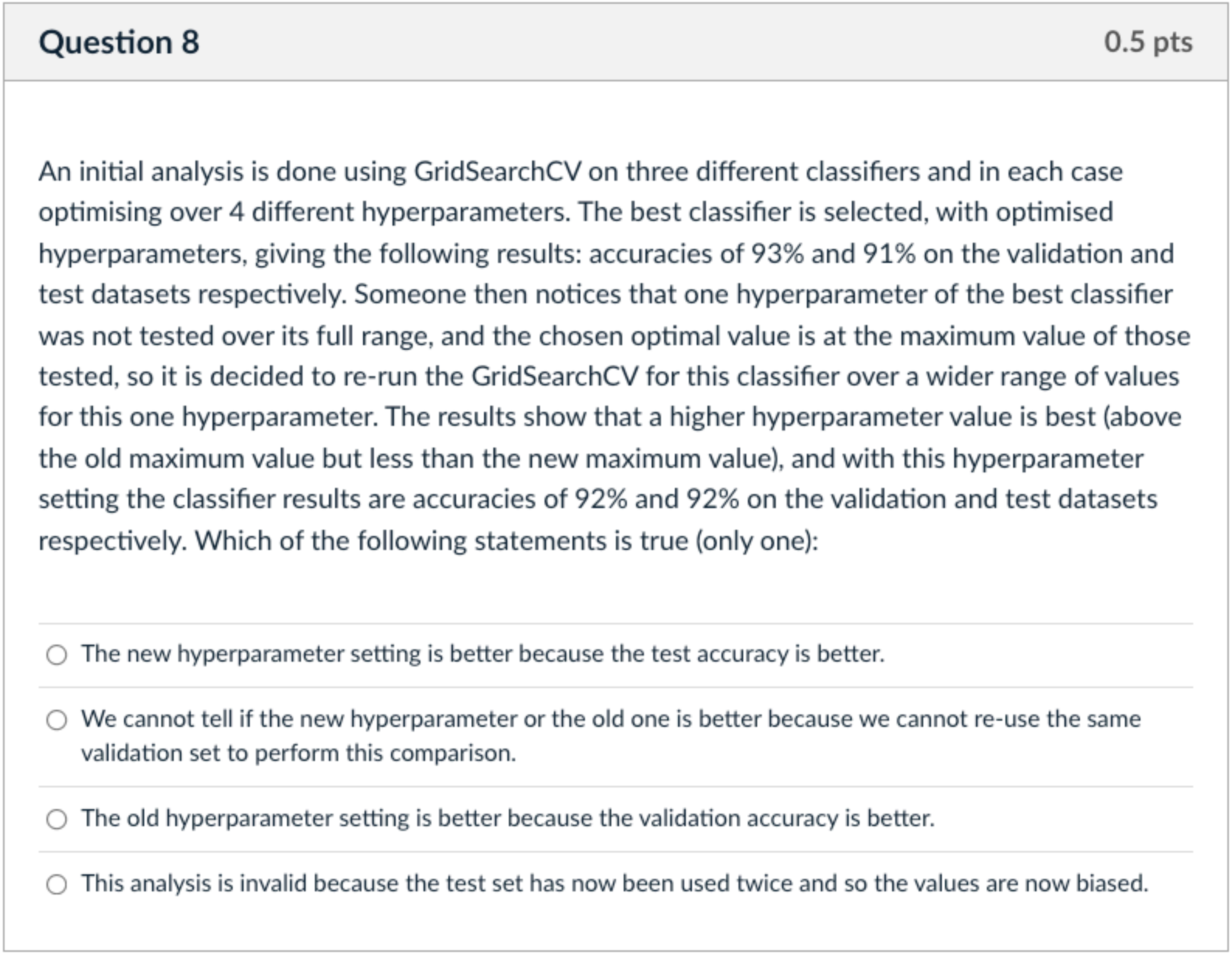
○ Class 0 and Class 2 (equal)

○ Class 1

○ Class 0

Summary of Assessed Quiz 2

- Thi



Question 8 0.5 pts

An initial analysis is done using GridSearchCV on three different classifiers and in each case optimising over 4 different hyperparameters. The best classifier is selected, with optimised hyperparameters, giving the following results: accuracies of 93% and 91% on the validation and test datasets respectively. Someone then notices that one hyperparameter of the best classifier was not tested over its full range, and the chosen optimal value is at the maximum value of those tested, so it is decided to re-run the GridSearchCV for this classifier over a wider range of values for this one hyperparameter. The results show that a higher hyperparameter value is best (above the old maximum value but less than the new maximum value), and with this hyperparameter setting the classifier results are accuracies of 92% and 92% on the validation and test datasets respectively. Which of the following statements is true (only one):

○ The new hyperparameter setting is better because the test accuracy is better.

○ We cannot tell if the new hyperparameter or the old one is better because we cannot re-use the same validation set to perform this comparison.

○ The old hyperparameter setting is better because the validation accuracy is better.

○ This analysis is invalid because the test set has now been used twice and so the values are now biased.

Summary of Assignment 2

- In general people did well, but common mistakes were:
  - using the **test set too early**
  - creating a *fixed validation* set and also using *CV* (you don’t do both)
  - not using pipelines
  - not performing hyperparameter optimisation (e.g. *GridSearchCV or alternative*)
  - not using stratify to create datasets
  - not interpreting confusion matrix correctly
    - probability answer = bottom left value of row-normalised matrix
  - misinterpreting the selection of a performance metric as the selection of a baseline case (which is not needed)
  - not spotting that recall is a bad metric as it hits maximum for a poor baseline
  - not reading the question carefully or not answering a question
  - not trying question 4
    - this should have been easy - you just repeat what you did for the last half of question 2 but with a different feature set (so same code, different inputs)
  - not trying question 3 (out of time?)

Pointers for Assignment 3

- Limit is on the number of times you call model.fit, not the number of epochs
  - Each training run can have as many epochs as you like, just limit the number of training runs (across all the networks you try) to a maximum of 50
- Follow the standard workflow and make sure you document what you are doing by showing outputs of code and verifying that things are working or look OK
  - load, check and fix data
  - run a baseline model
  - perform hyperparameter optimisation across different models/hyperparams
  - report the final unbiased results
- As in the workshops, fixed pre-processing is fine and *no pipelines are needed*
- You can test any number of models (from 2 up; at least one CNN and one dense)
  - Complex models (as in recent papers or other courses) are ***not required***
  - It is not how recent the model is that counts - it is your ***process*** that counts
- GridSearchCV is very costly for DL models, so try an alternative (see workshops)
- Read the instructions carefully!
- Try to make your code clean, concise and clearly commented
- Explain why you make certain decisions (not long paragraphs, but just a short sentence or two is normally all that is needed)
- Label your outputs so it is clear what they are