# ADD COMMENTS TO ALL OF THE CODE

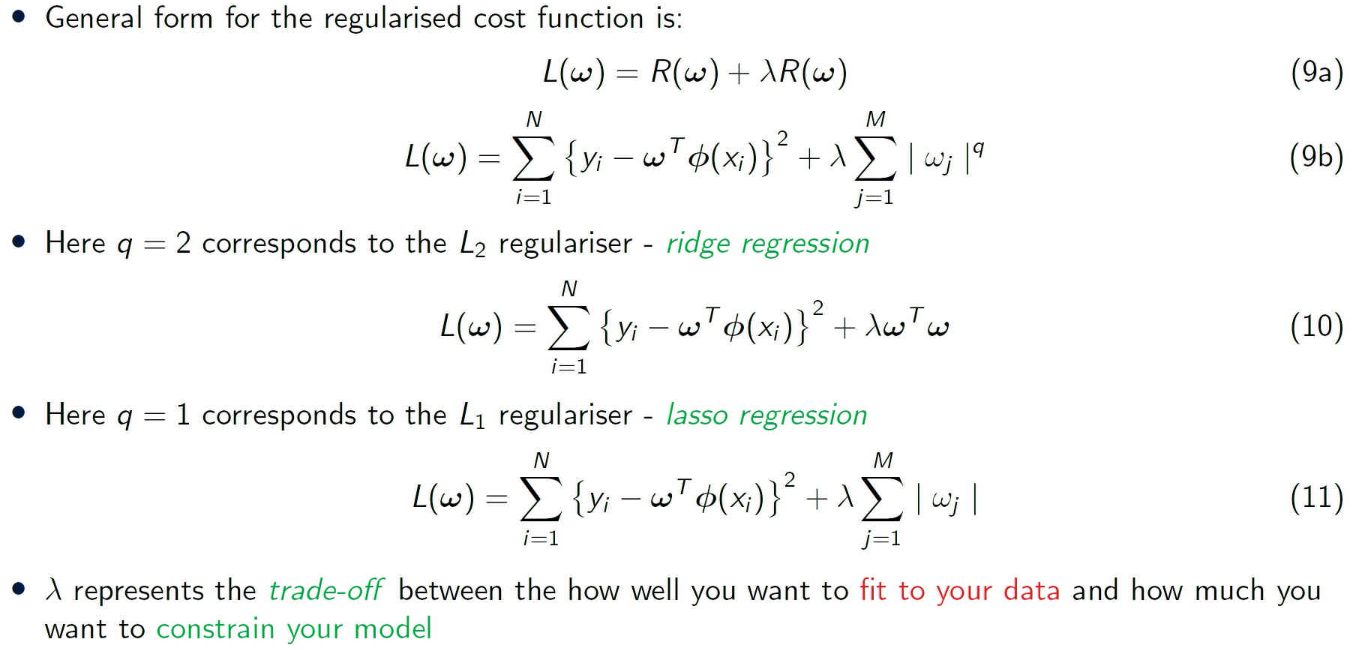
# Exercise 1

* Could calculate 1.5x IQR to show outliers for race features

# Exercise 2

* Test model when data is imputed using median vs mean vs mode(?)
* Remove target value from training set
  + Otherwise regression function will just set all weights to 0 and the weight for the target feature to 1
  + Turn the target feature column into its own 1 column dataframe
* Remove outliers from the data
* Remove non numeric values before imputing
* Scale the data to make them all go between 0 and 1
* Check shape of resulting dataframe – make sure its (x, y)
  + It ensures we are in good shape to do linear regression
* STRATIFY BASED ON ETHNICITY – create extra colum which shows ethnic majority (categorical)
* Add random attributeadder just to show I can do it AFTER finishing the CW – it is not important in this

# Exercise 3 – DOUBLE CHECK I HAVE ACTUALLY ANSWERED THE QUESTION PROPERLY

* Test 90:10 split vs 80:20
  + Split should be done at the beginning, before pipeline run
  + Remove labels from training set
  + Then run pipeline
  + Test stratified split on incidence rate, vs bach degree over 25 vs no stratified split
* Lin\_reg.fit( training set, target df)
  + Tree.fit(X, y) - I.e. it is the exact same parameters
* Ols
  + If you create the design matrix you can solve what weights you need for OLS
  + “with OLS, the cost function you want to minimise is your sum of squared residuals”
* Aim of regularisation is reducing sensitivity of model on the data
  + Use prior estimate / hypothesis - “this is what I believe the trends are” and then you verify with more data whether your initial belief was correct and if not then alter the model
  + 
* L1 regularisation (Lasso) is really good for feature selection, because it encourage sparse solutions, meaning it pushes the weights close / to zero, if the corresponding features are not important in the regression model
  + Hence, good for feature selection
* L2 regularisation (Ridge) does something similar
  + Doesn't push the weights directly to 0, but pushes weights towards 0 asymptotically
* EXPLAINING HOW I GOT THE REGULARISATION WEIGHTS USED -ChatGPT
  + The best values to use for the param\_grid dictionary in the GridSearchCV class will depend on the specific problem you are trying to solve and the characteristics of your data. In general, it is a good idea to try a range of values for the hyperparameters that you are tuning to see which values give the best performance.
  + One common approach is to start with a relatively wide range of values for the hyperparameters and then narrow down the range as you get a better sense of the performance of the model. For example, if you are tuning the alpha hyperparameter for Lasso or Ridge regression, you might start by trying a range of powers of 10, such as [0.001, 0.01, 0.1, 1, 10]. If you find that the model performs best with a smaller value of alpha, you can try a more fine-grained range of values around that point, such as [0.0001, 0.001, 0.01].
  + It is also a good idea to use cross-validation to evaluate the model's performance for each value of the hyperparameters, as this can help you to get a more accurate estimate of the model's performance on new data. The GridSearchCV class performs cross-validation by default, so you can use it to evaluate the model's performance for different values of the hyperparameters.
    - ESSENTIALLY: start off with a big scope of values for alpha, then trial and improve by slowly closing in on a more granular value

# Exercise 4

# Exercise 5

# Exercise 6

# Metrics

* Regression
  + Mean squared error and root mean squared error
    - A value of 0, especially for DTs can imply overfitting
* Test set should never be used in any training and should beset aside right until THE END of analysis
  + Do Cross fold validation to assess hyperparameters / models until the end
    - Can specify in cfv function which metric we want to use to measure performance e.g. negative MSE