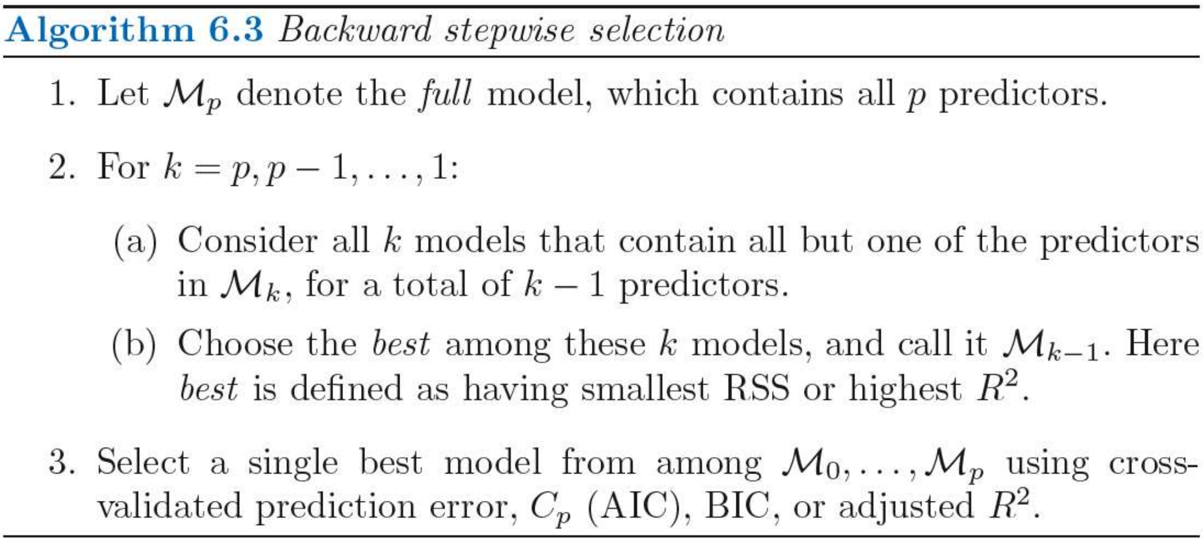
LAB 6 – Backward Stepwise Selection

In this lab, we will perform backward stepwise selection with multiple linear regression. To do this, we will need the .csv file provided within the assignment on Blackboard.

The entire algorithm is as follows:



Algorithm 6.3 *Backward stepwise selection*

```
1. Let $\mathcal{M}_p$ denote the *full* model, which contains all $p$ predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all $k$ models that contain all but one of the predictors
       in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these $k$ models, and call it $\mathcal{M}_{k-1}$. Here
       *best* is defined as having smallest RSS or highest $R^2$.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-
   validated prediction error, $C_p$ (AIC), BIC, or adjusted $R^2$.
```

Follow these instructions:

- (20 pts) Define a function for calculating the adjusted R^2 scores This should take three parameters: the original output (y), predictions for this output ($\hat{y}$), and the number of variables in your model (d). In the body of the function, calculate and return the following:

$$\text{Adjusted } R^2 = 1 - \frac{\text{RSS}/(n-d-1)}{\text{TSS}/(n-1)}$$

n is equal to the number of elements in y (or $\hat{y}$, since they have the same size).

Also implement another function for calculating regular R^2 scores, if you haven't done so in the previous labs.

- (10 pts) The input X is slightly different: Get the “age”, “experience” and “power” (x_1, x_2, x_3) columns like previous labs (don’t forget the ones column). Then, like we did in LAB 3, append a column containing random integers from -1000 to 1000 to it (x_4). The output is again the “salary" column (y).

  Since we have four variables plus the intercept, we should calculate R^2 scores of according models M_0, M_1, M_2, M_3, M_4.

  Initialize an array which should hold adjusted R^2 scores for each individual model.

  Initialize another array which should hold the indices for columns that are removed from X at each step.

- (20 pts) Calculate M_4:

  M_4 is the model which uses every variable. Therefore, using X and y, compute regression coefficients, get predictions and calculate the adjusted R^2 score *(For those who are curious: The regular R^2 score is equal to the adjusted one for this case, because we are using all variables)*. Append this score onto the corresponding array. **Note that both regression coefficients and predictions are computed using X and y, no train-test splitting are done.**

- (50 pts) Calculate the remaining models M_3, M_2, M_1, M_0:

  This should be done in a loop. At each iteration of the loop, a variable should be selected for deletion.

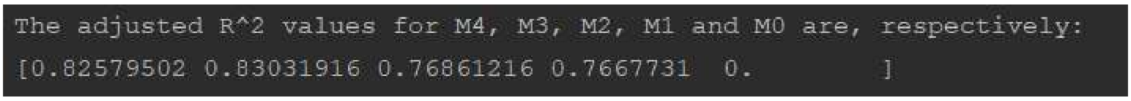
  That selection should be done in another inner loop: At each iteration of this inner loop:

  - Temporarily delete a column from the input of the previous iteration. **(except the ones column, don’t delete that one!)**
  - Perform linear regression (using this new input and y), get predictions and calculate the regular R^2 score of this model.
  - Find which deletion yields the maximum R^2.

  After the inner loop ends, you should be able to know which column is to be deleted for maximum R^2. Perform that deletion permanently. Append the index of the column which is deleted to the corresponding array.

  Then, perform linear regression using this new input and y, get predictions and calculate the adjusted R^2 score. Append this score onto the corresponding array.

When the outer loop ends, display the adjusted R^2 scores like the following:



```
The adjusted R^2 values for M4, M3, M2, M1 and M0 are, respectively:
[0.82579502 0.83031916 0.76861216 0.7667731  0.        ]
```

The first score should be different at each run of the program. The rest, however, are absolute values.

Important note: Due to the randomness of the added column, the first score is sometimes higher than the second one. But, in general, the second value should be higher. You could replicate this behavior by running the code multiple times.

Another important note: We are once again back to the “no libraries for bypassing calculations” zone. You are of course allowed to use other libraries at your will.

Continuing with the insight section:

Model selection approaches help us select variables from a large set. Some of these variables can be of little help, some can even have adverse effects on our test error, therefore, these approaches come in very handy. They also help reduce the amount of data, which in turn reduces computational requirements.

We implemented backwards selection. We started with all of our predictors and removed one predictor at a time for each model M_3, M_2, M_1, M_0. Notice that for M_0, our input is only the ones column and our adjusted R^2 score (or regular R^2 score) is zero. This should always be the case since we are using the mean of the data as our predictions (think about the calculations to understand why). Therefore, RSS should always be equal to TSS. And since our number of variables d is zero, the equation above for adjusted R^2 score equals to $1 - 1 = 0$.

You can see the effects of this algorithm: The random column is, most of the time, removed from the input to yield better results overall. This doesn't happen all the time, which should support the idea that subset selection algorithms do NOT always yield the most optimal model.

We haven't kept track of which predictors we have removed for each model, but you can extend your work to reflect that information with a bit of extra programming work.