# Baruch Gaxiola Valles

1. Python: Take a short list of strings and return a new list of strings with "!" appended to

each string and using two different methods:

1. list comprehension
2. recursion

Code for this:  
def exclamation\_comprehension(string\_list):

res = [s+"!" for s in string\_list]

return res

def exclamation\_recursion(string\_list):

if not string\_list:

return []

return [string\_list[0]+"!"] + exclamation\_recursion(string\_list[1:])

def main():

string\_l = ["a","b","c","d","e","f"]

print(exclamation\_recursion(string\_l))

print(exclamation\_comprehension(string\_l))

if \_\_name\_\_ == "\_\_main\_\_":

main()

2. You have an email priority classification project (high priority or not) and one of the

candidate features that you are considering is whether the sender is in the recipient's

contact list. If the client gives you two years' worth of data to train on and every email

has in its metadata a value for is\_contact, what is important to ask the client (who

doesn’t know much about data science or machine learning) about is\_contact that

makes sense to ask in the first conversation when they give you the data (assume a

15-30 min conversation)?

The first thing I would ask about the variable would be how selective they are when they decide to add a new contact to their list of contacts. I think this is the most important aspect, because if they do it indiscriminately, the variable is very likely to be of no good use. In the other hand, this would be an incomplete assumption, cause even if they are careless with the list, it is very possible that emails that arrive from outside the list are not high priority, so a follow up question would be if, to their knowledge, most of the emails that come from not contacts are non priority, or if there is a situation that makes it usual to receive priority emails from addresses that are not in the contact list, for example new clients or new applicants. Another important aspect is their history of email management. I would need to ask if their policies about adding people to the list have changed, and if so, when did they make that change. If this has been the case, it would be worth it to look for that time where things changes to define different rules for different periods. Lastly I would ask about any hierarchy in the list, like mailing groups, that can also give information about the importance of the variable (and the list itself).

3. You need to implement a logistic regression classifier with online learning where some of

the weights might be fixed at times during learning. How would you do this? Feel free to

use existing libraries in your language of choice. You don't need to code it up, but you

should have a good idea of what steps need to be done and be prepared to answer any

questions that we might have.

Just like any classification problem, the code would be pretty much organized as: Imports, Data loading and management, Model definition, Loss function, optimizer and other hyperparameters, Training, Evaluation.

In order to make it Logistic Regression, the implementation of the module would imply having a base model (that can inherit from a Linear Regression module) and having a linear layer at the with the number of inputs and outputs (in this case we only use 1 output). Also we need to redefine the forward pass (method) so it uses a Logistic function as its activation function, that will take the outputs from the linear layer and pass them through a sigmoid. Also it is important to define the loss function as something suitable for binary classification like Binary Cross Entropy.   
 To make this an online learning method, we assume we are receiving the data one element at a time, so we pass it to the model and it will learn in every step, having the model ready for evaluation at any time. This is usually implemented as a partial fit method that comes built in in many libraries.   
 The most complex part of this implementation would be the ability to freeze some of the weights some times during learning, but it will give it the capability to examine the problem with more options. The first thing we need is to add a parameter to the model that determines if the weights will be frozen or not, but we can use the fact that when they get frozen we need to define which ones will be fixed, for example via a binary mask for the weights, and then we can just assume that if the mask is present in a training step, those weights will be frozen. The main changes we need to do are setting the fixed values for the chosen weights during the initialization, and modifying the partial\_fit method. During the initialization we can use something like “requires\_grad\_” function to tell the model that we do not want those weights to change. The partial\_fit method will have the usual prediction, loss calculation and backpropagation, but then we need to reset the gradient values for the fixed weights to zero before we apply the usual gradient descent.

And that is it, the training steps will use that partial\_fit method, and we can pass data to the model to get the prediction just for evaluation.

4. You are working at a startup hedge fund. A model receives time-series data for various

stocks and outputs automated trading decisions. A not-yet-fully-understood bug in the

code causes the algorithm to, on occasion, reverse the model's intended trade (e.g. buy

when you would sell or vice versa). It has already been determined that the bug lives in

the code executing trades and not in the model code itself.

You investigate a little bit, and discover that the probability of the bug occurring on any

given trade is ~ 0.001. Every trade either gains or loses $5,000 (all the bug does is flip

the decision the model makes).

The fund typically executes ~10,000 trades in a year. The model (independent of the

bug), chooses correctly on ~52% of the trades it selects. What do you do next and

why?

In principle, if I am in charge of any code that I know it has errors, it will be my responsibility to fix them and there after I will do what in my power to do it.

In the other hand, I would have to check the overall effect that the bug has. In this case, of the 5200 trades that are correctly predicted in a year, ~5 of them will be changed by the bug. Same wise, from the 4800 incorrectly predicted trades, ~5 of them will be changed by the bug. That means that in the overall the will be not a significant difference in the number of gains and losses in a 1 year period.

With this in mind, my opinion is that it would be more important to use my time and resources to prioritize getting a new model that is more efficient, than fixing the bug immediately. Having said that, I would have to flag and comment the problem so it is never overlooked permanently, and fixed as soon as possible.