Dear Smart data Solutions,

Thank you for this opportunity. Here are my exam results.

The Java programming exercise was completed using the Visual Studio Code platform for Java development. To run it, open up the "Java\_CSV\_Merge" folder with Visual Studio Code.

Upon inspection of the provided CSV files, they are already in sorted order, so I have assumed that the algorithm I should write receives these in sorted order. If that is not the case, then I would simply sort them prior to running the algorithm I created. The algorithm works by stepping through both files simultaneously, incrementing one or the other based upon if the key values match, or if one is greater than the other. When they match, we merge the corresponding lines to the output file.

My algorithm has a worst case run time of order N1 + N2, where N1 is the number of lines for of the first CSV file, and N2 is the length of the second. My algorithm uses a small, constant amount of data since it only needs to know the current key values of each of the CSV files. Therefore this algorithm is optimal and correct. Exceptions will be thrown if the assumptions I made about the input CSV files are incorrect.

For the Machine Learning exercise, I build a Neural network Classifier based only upon the image data. The model can be run by opening the Visual Studio solution as long as the appropriate Python extensions are included. I divided the image data into approximately 75% devoted to supervised learning, and 25% devoted to testing the model. The classifier separates the images into 5 different classifications corresponding to the subdirectories 0, 2, 4, 6, and 9 from the provided data set.

I like using TensorFlow with Python. I used a similar model to build a system for distinguishing between real and fake pictures of faces a while ago, and so I decided to use the TensorFlow library again. However, TensorFlow does not handle TIF files, so I needed to convert the TIF image files into JPG image files, which was handled through a third party system.

The model required about 40 minutes of run time to execute, and ultimately achieved an accuracy of approximately 70%. The output from running the program is included as a text file in the Test Output folder. If I were able to devote more time to this, I would have liked to incorporate the text information from the OCR files.

Sincerely,

-Carsten Quinlan