Pierre Kachi 232732

Rami Chalouhi 221631

Said Chebbo 220700

We began by using pandas to read the dataset, which contains information about shipped products, and then used seaborn and matplotlib, to visualize a heatmap of the correlation of the data with each available parameter (height, length and width), as well as individual plots, all in correlation to the price, which we will train our model to guess. We then used sklearn train\_test\_split to separate the training data from the testing data, before building our model, using 3 separate dense layers. The first one consists of 32 neurons, uses relu as an activation function, and we gave it an input shape parameter equal to the amount of keys found within the x (the data used to try and guess y). The second layer this time consists of 64 neurons, and uses relu, but doesn’t take in a shape parameter as it will be working on the output of the first layer. The final layer only consists of one neuron, and it will be responsible for outputting the generated results. We used the Adam optimizer and gave it a learning rate of 0.001. We then used the mode.compile, and the mean squared error loss function, and gave it mean average and the mean squared error as metrics to calculate during training to finalize it before using return model finish it. We then used the model to evaluate the loss(mse), the mae, the root mse, and our r2 score which represents how well our ai manages to guess, which turned out to be abysmally bad, as it returned negative. We were offput at first by such a score, and considered changing datasets, but we thought it would be interesting to highlight the relationship between datasets and the chosen models. The main potential issue we might have found, is poor relation between the price and the size of the shipped objects. As can be seen by the initial plots as well, the data is very noisy, we therefore ended up with a very bad r2 score, and predictions far from accurate.