Docker commands:

docker build -t nn\_fastapi:latest .

docker run -dit --rm --name init\_nn\_fastapi -p 8080:80 nn\_fastapi:latest

docker run --rm --name init\_nn\_fastapi -p 8080:80 nn\_fastapi:latest

http://localhost:8080

http://localhost:8080/health

http://localhost:8080/docs

[2.16] In yout browser, copy-paste http://localhost:8080/mall/customers/segmentation?genre=Female&age=65&income=38&spending=35

docker stop init\_nn\_fastapi

hello:  
"This project was designed to take a few key features from a beer styles data set and then build a model that would take those inputs and then predict the given beer style. It takes four inputs (review\_aroma, review\_appearance, review\_palate, review\_taste) and each one must be given an integer score from 1-5 and then it will predict the beer style. Here is the github repo: <https://github.com/freescania/advdsi_at2>"

Student: Angus Nicholson

Project Name: Assignment 2

Date: 14 March 2021

Deliverables:

Github repo: <https://github.com/freescania/advdsi_at2>

## Business Objective

We are a data science consultancy working for a local beer specialist store in Sydney. They have hired us to make a web app that will contain a machine learning model that will help recommend beers to people depending on their personal tastes. They have a large dataset with 104 different beer recommendation possibilities and one element they hope the project will accomplish is that it may entice consumers to look at some of the many exotic beers that they stock at the specialist beer store.

They would ideally like a web app and model that provides a relatively high degree of accuracy for people’s tastes, but also is able to show them interesting beer recommendations that they may not have thought of. They only wanted the model to use brewery\_id, review\_aroma, review\_appearance, review\_palate, review\_taste to predict beer\_style.

## Hypothesis and Objectives

We will be looking at building a neural network in pytorch and will investigate the best neural network architecture to try and deliver high quality predicted results based on consumers beer preferences.

## Data Preparation

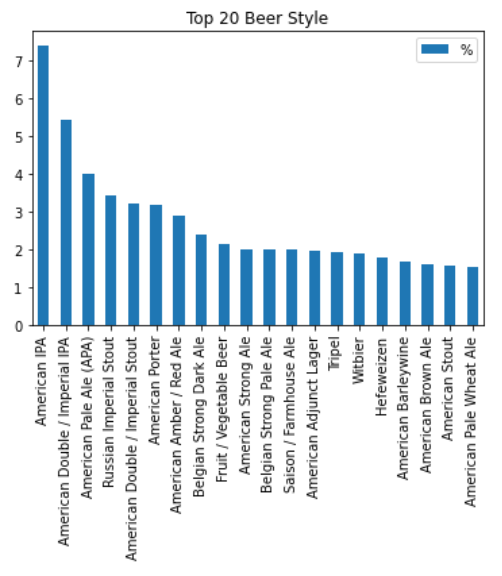
We received a CSV file named beer\_reviews that contained 1,586,614 rows and 13 columns of data. As noted above, the beer store only wanted the model to work with five features (brewery\_id, review\_aroma, review\_appearance, review\_palate, review\_taste) to predict the target variable beer\_style. And within beer\_style there were 104 different classes of beer. This immediately made it clear that we should regard this as a multiclass classification problem.

We first removed the unwanted columns, then brewery\_id we classified as a categorical variable, while for the rest we classified them as numeric variables. We tried to use one hot encoding on brewery\_id but there were 5840 unique brewery\_id which dramatically would have increased the size of the training data set in an already very large dataset causing CPU processing issues. For the remaining numeric features we used a standard scaler to normalise them to keep them in a 0-1 range as Pytorch does not process negative values.

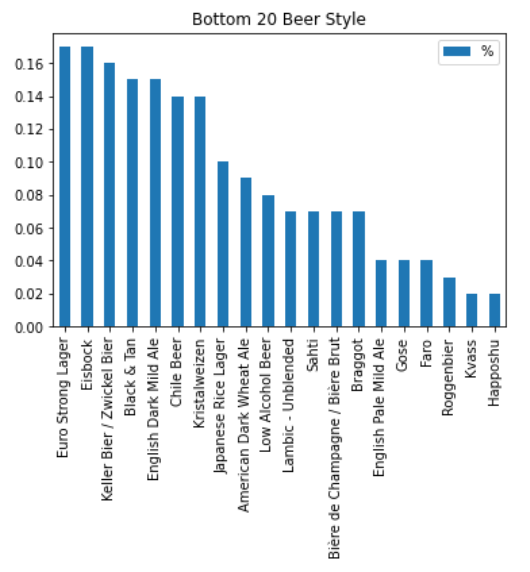
Given we wanted the target variable to stay as one column we decided to use an ordinal encoder to change the beer\_styles into numeric values so the model could process it even if the ordinal difference between them was largely irrelevant.

## Data Exploration

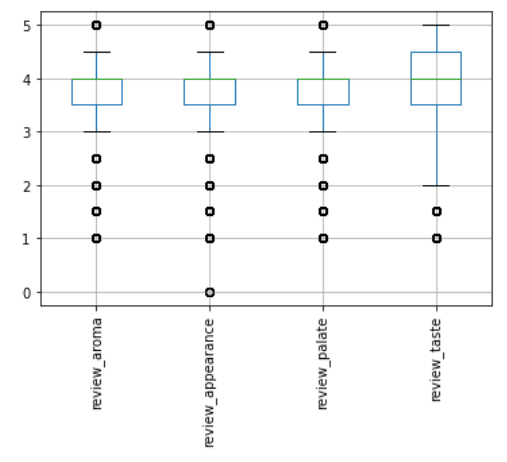
We began to look into the distribution of the 104 classes in the target variable. It was a comparatively large amount of classes with from a small number of features. The highest frequency classes are displayed below and the most frequent “American IPA” only appears just over 7% of the time, which does raise some concerns about how easy it will be to train the accuracy of the model over such a large possible set of classes in the target variable.



While for the least frequent beer styles, some such as Happoshu were appearing as infrequently as 0.02% of the time in the dataset, which doesn’t give one great confidence of building a model that would accurately predict this occurrence. The hope is that the data set is large enough that some of these low data frequency issues may be balanced out given the amount of data to train upon.



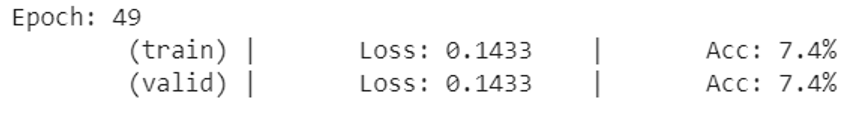
Of the four numerical features, it was clear that the were scored from 0 to 5 presumably as people reviewed the beers. One issue is that 0-5 scoring across four features may not provide enough variance in those features to help sort through the 104 classes in the target variable.



## Modelling

To initialize the model and make sure all the elements of the architecture were working we started with a relatively simple neural network. We just used two linear layers for the forward propagation with the ReLU (Rectified Linear Unit) activation function, taking in the four features, outputting 32 neurons in the first layer, and then the second layer took in the 32 neurons and output the 104 classes in the target variable before finally going through a softmax activation function to make a prediction.

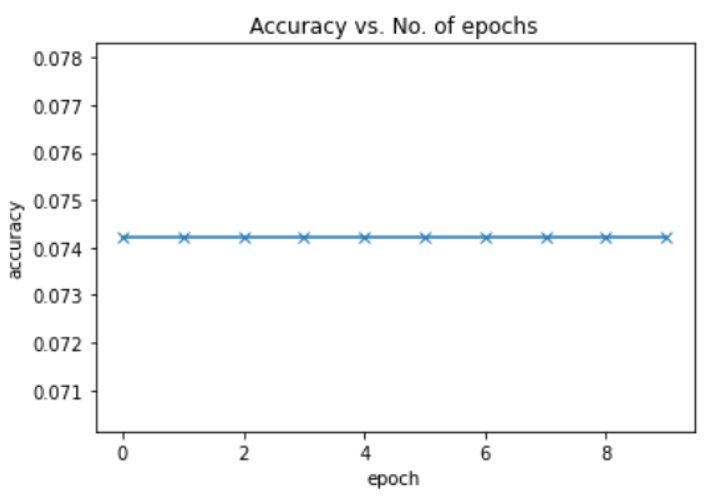
We then ran 50 epochs of 32 batches to train and evaluate the model, achieving a final loss of 0.1433 and accuracy of 7.4%. Unfortunately, that simple model showed no improvement over the 50 epochs printing the same results through the same process.



We then tried adding two more layers both with ReLU activation functions with the following framework:

* layer\_1, Linear, input=4, output=32
* layer\_2, Linear, input=32, output=104
* layer\_3, Linear, input=104, output=32
* layer\_out, Linear, input=32, output=104
* Softmax activation

And again we saw much a similar outcome of performance with loss of 0.1433 and accuracy of 7.4% and charted below.



Given the amount of different classes in the target variable it seems the model is struggling to identify most of the less frequent classes. If we look at the accuracy score of 7.4% that matches up very closely the distribution of the most frequent class – “American IPA”. It’s likely that the model is just optimizing by predicting that one class.

Potentially a model of much greater depth and layers would be able to improve on this performance and the team could look into providing further funding for buying some processing power in through a cloud provider such as Amazon Web Services or Microsoft Azure.

## API Modelling

We then took our best model and built an API using the FastAPI package to host the model as an interactive we app for customers of the beer specialty store. We built a python file that listed a series of HTTP request endpoints.

The first HTTP Get endpoint was inside the main.py file where we created a function called read\_root() that will describe the beer reviews project and how to use it. Then we added another HHTP Get endpoint “/health” for status\_code=200 that would give a healthcheck that said the model was ready to go. We then added a function that would format the inputs of users into a form that could be used in the model. Then we created another HTTP Get endpoint "/beer/type/" that would take the users inputs format them and then predict a beer type they may like. We then posted a final HTTP Get endpoint "/model/architecture/" that described the neural network architecture.

For a user to then run the app they simply need to go to the server destination and enter the values 0-5 in the follow headings in the address destination, for example:  
<https://fast-temple-40998.herokuapp.com/beer/type/params?review_aroma=4&review_appearance=4&review_palate=4&review_taste=4>

## Conclusion

We addressed the business problem from the beer specialty store as best as possible by building a neural network that would predict one of 104 different types of beer styles based on four user inputs. The issues we faced is likely the wide range of classes and relatively limited array of inputs that the client asked to use meant it was hard to build a model that good deal with the wide array of imbalances within the length of time and resources given.

It is quite possible that there may be some off the shelf pre-trained neural network models available that could address this particularly task. Or if resources were provided to pay for processing power from a cloud provider a model of much greater depth and complexity could be trained and hopefully improve on our models performance.

In any case, hopefully this could be used as interesting marketing tool to bring greater awareness to the beer specialty stores brand and the wide array of beers they have on offer even if the accuracy of the model is not particularly high.