Machine Learning

Feature extraction

To implement any kind of machine learning algorithm we need some features to train on. This dataset contains mostly categorical features: country, designation, province, region, taster name, taster twitter handle, title won, variety and winery. It also contains two numerical features, specifically points (rating) and price. There is another feature – description- which is text (would require NLP to extract information).

One idea to encode all these categorical features into numerical data which can be used by machine learning is to use a One-Hot Encoding. This means that to encode, for example, country, we would need a feature vector of length equal to the total number of countries in the dataset (43, as we will discover later). Each column of this vector would have a value of either 0 or 1 depending on whether the wine is from the country representing that column or not.

Doing some simple data analysis we discover the following:

unique countries: 43

unique provinces: 425

unique designations: 37955

unique regions: 1230

unique varieties: 701

unique wineries: 16745

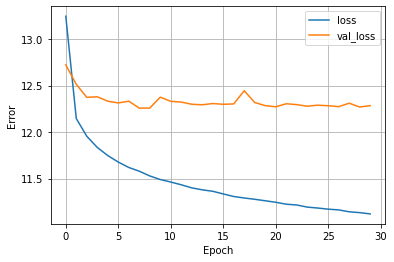
This means that if we were to use everything, our data points would have 57099 features each. This is too much. We narrowed it down to a few of these features. By looking at the data we can already tell country is redundant in the sense that “province” already represents only one country. We can also eliminate “designation” since there are way too many unique values - same situation for “wineries”. The “region” column had a lot of null values so it was omitted as well. This leaves: province and variety as our categorical features. We can also add points and price as numerical features. I also decided to OHE the points labels as they only ranged from 80 to 100. This totals to 1147 feature vector for a single sample.

The final dataset that was used for machine learning has the following columns: province, variety, points and price with no null values except in price. This leaves a total of 120915 samples without null values at all which will be used for training and validation and 8992 samples which are missing the price tag which we will use for testing and completing the dataset.

Model

For the model we chose to use a Neural Network. It has one hidden layer of size 1024 with ReLu activation. The optimizer used was Adam. Since the self-imposed task is determining the price of a wine based on the selected features (regression), we use MAE (mean absolute error).

Training



The final validation loss was around 12.25. This means that the algorithm can predict the correct price of a wine +-12 dollars based on the province it’s from, its variety and a score given to it (80 to 100).