Sian’s questions from class – I will attempt to answer any questions from class here that I needed to look into afterwards

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Lesson | Your question (as I understand it!) | Response including useful resources |
| 20-10-20 | Intro to Pandas | *If we are bringing two files to our data frame, with the same headers but they are in different orders, what will happen?* | My assumption on this would be that pandas/python would do the heavy lifting of matching column names for us… that has proven to be true in a couple of different variations I have tried. So as long as the column headers are the same strings, ie not lowercase /uppercase mismatched, or typos, the concat will stack the data under the appropriate columns regardless of the incoming order. If there are some new columns – eg in one file, not in another, those are added at the end of the column list. Which is quite logical too. So, no worries there  The other thing I noticed is that the order of the columns on the output is obedient to the order of the columns in the first file listed in the concat cell.  Here is a really nice tip for working with frames with lots of columns – reorder the ones that matter! https://stackoverflow.com/questions/41968732/set-order-of-columns-in-pandas-dataframe |
| 20-10-20 | Intro to Pandas | *Why do we need to say axis =0 or axis =1 … what difference does it make ?*  *Examples*  **data = pd.concat([data,file1], axis=0)**  **data = data.drop(['tcode'], axis =1)** | Ambiguity in Pandas Dataframe / Numpy Array "axis" definition - Stack  Overflow  A data frame in pandas looks like the image above. Axis 1 and 0 is pre defined. In our example concat, we want to create long thin data, like append the rows to the existing frame – hence referring to axis =0 (axis = index/rows)  In our drop example we want to search along the column headers and apply a drop, ie following axis =1. (axis = columns)  We are specifying the axis along which we compute the function.  Useful resource for where we are now and gives a hint of where this is going to come up later on :  <https://railsware.com/blog/python-for-machine-learning-pandas-axis-explained/> |
| 21.10.20 | Using seaborn and matplotlib | What are the black lines on the top of the bars in the seaborn barplot ? | I found a couple of great articles for you on this – and you may be glad to know that you aren’t the only one who asked!  <https://stackoverflow.com/questions/35193996/what-is-the-overlay-line-on-each-bar-in-a-bar-chart>  … they are vertical error bars  -“ Error bars are a graphical representation of the variability of data and are used on graphs to indicate the error, or uncertainty in a reported measurement.”  The confidence interval – 95% in our case is calculated for you based on the number of data points and diversity. Its worth understanding what it means :  The "95%" says that 95% of experiments like we just did will include the true mean, but **5% won't**.So there is a 1-in-20 chance (5%) that our Confidence Interval does NOT include the true mean.  You can modify what the error bars show – eg standard deviation, add caps to make it more obvious that’s whats showing  <https://seaborn.pydata.org/generated/seaborn.barplot.html>  You can also turn them off <https://stackoverflow.com/questions/40088585/turn-off-error-bars-in-seaborn-bar-plot> |
|  |  | How do you turn a boxplot horizontal with matplotlib? | Simple answer which we should have guessed! Just define inside the () your requirement that this is not vertical… as that’s default, we use vertical =False  **Data[[‘actual\_height’]].boxplot(vert=False)**  **Before :**    **After :** |
|  |  |  |  |