-Provide the number of rows and columns in this dataset.

-Which columns had no missing values? Provide a set of column names that have no missing values.

-Which columns have the most missing values? Provide a set of column name that have more than 75% if their values missing.

-Provide a pandas series of the different **Professional** status values in the dataset. Store this pandas series in **status\_vals**. If you are correct, you should see a bar chart of the proportion of individuals in each status.

-Provide a pandas series of the different **FormalEducation** status values in the dataset. Store this pandas series in **ed\_vals**. If you are correct, you should see a bar chart of the proportion of individuals in each status.

-Provide a pandas series of the different **Country** values in the dataset. Store this pandas series in **count\_vals**. If you are correct, you should see a bar chart of the proportion of individuals in each country.

-Now you have had a closer look at the data, and you saw how I approached looking at how the survey respondents think you should break into the field. Let's recreate those results, as well as take a look at another question.

-In order to understand how to break into the field, we will look at the **CousinEducation** field. Use the **schema** dataset to answer this question. Write a function called **get\_description** that takes the **schema dataframe** and the **column** as a string, and returns a string of the description for that column.

-The question we have been focused on has been around how to break into the field. Use your **get\_description** function below to take a closer look at the **CousinEducation** column.

-Provide a pandas series of the different **CousinEducation** status values in the dataset. Store this pandas series in **cous\_ed\_vals**. If you are correct, you should see a bar chart of the proportion of individuals in each status. If it looks terrible, and you get no information from it, then you followed directions. However, we should clean this up!

- I wonder if some of the individuals might have bias towards their own degrees. Complete the function below that will apply to the elements of the **FormalEducation** column in **df**.

-Now we would like to find out if the proportion of individuals who completed one of these three programs feel differently than those that did not. Store a dataframe of only the individual's who had **HigherEd** equal to 1 in **ed\_1**. Similarly, store a dataframe of only the **HigherEd** equal to 0 values in **ed\_0**.

Notice, you have already created the **HigherEd** column using the check code portion above, so here you only need to subset the dataframe using this newly created column.

-What can you conclude from the above plot? Change the dictionary to mark **True** for the keys of any statements you can conclude, and **False** for any of the statements you cannot conclude.

-The proportion of missing values in the Job Satisfaction column'

'According to EmploymentStatus, which group has the highest average job satisfaction?'

'In general, do smaller companies appear to have employees with higher job satisfaction?'

-Do individuals who program outside of work appear to have higher JobSatisfaction?':

'Does flexibility to work outside of the office appear to have an influence on JobSatisfaction?'

'A friend says a Doctoral degree increases the chance of having job you like, does this seem true?

-For the last two questions regarding what are related to relationships of variables with salary and job satisfaction - Each of these questions will involve not only building some sort of predictive model, but also finding and interpretting the influential components of whatever model we build.

-Now take a look at the summary statistics associated with the quantitative variables in your dataset.

- column just listing an index for each row'

'The maximum Satisfaction on the scales for the survey'

'The column with the most missing values'

'The variable with the highest spread of values'

-A picture can often tell us more than numbers.

-Often a useful plot is a correlation matrix - this can tell you which variables are related to one another. (heat map)

- Use the scatterplot matrix above to match each variable (**a\**, \**b**, **c\**, \**d**, **e\**, \**f**, or **g\**) as the appropriate key that describes the value in the \**scatter\_sol** dictionary.

-'The column with the strongest correlation with Salary'

'The data suggests more hours worked relates to higher salary'

'Data in the \_\_\_\_\_\_ column meant missing data in three other columns'

'The strongest negative relationship had what correlation?'

-Here we move our quantitative variables to an X matrix, which we will use to predict our response. We also create our response. We then split our data into training and testing data. Then when starting our four step process, our fit step breaks.

-'What is the reason that the fit method broke?'

'What does the random\_state parameter do for the train\_test\_split function?'

'What is the purpose of creating a train test split?'

You have seen:

1. sklearn break when introducing missing values
2. reasons for dropping missing values

It is time to make sure you are comfortable with the methods for dropping missing values in pandas. You can drop values by row or by column, and you can drop based on whether **any** value is missing in a particular row or column or **all** are values in a row or column are missing.

A useful set of many resources in pandas is available [here](https://chrisalbon.com/). Specifically, Chris takes a close look at missing values here

<https://chrisalbon.com/>

<https://chrisalbon.com/python/data_wrangling/pandas_dropping_column_and_rows/>

<https://stackoverflow.com/questions/13413590/how-to-drop-rows-of-pandas-dataframe-whose-value-in-a-certain-column-is-nan>

-Drop any row with a missing value.

- Drop only the row with all missing values.

-Drop only the rows with missing values in column 3.

-Drop only the rows with missing values in column 3 or column 1.

-So, you now have seen how we can fit a model by dropping rows with missing values. This is great in that sklearn doesn't break! However, this means future observations will not obtain a prediction if they have missing values in any of the columns.

-What proportion of individuals in the dataset reported a salary?

-Remove the rows associated with nan values in Salary (only Salary) from the dataframe **num\_vars**. Store the dataframe with these rows removed in **sal\_rem**.

-Using **sal\_rm**, create **X\*\* be a dataframe (matrix) of all of the numeric feature variables. Then, let \*\*y** be the response vector you would like to predict (Salary). Run the cell below once you have split the data, and use the result of the code to assign the correct letter to **question3\_solution**.

-Remove the rows associated with nan values in any column from **num\_vars** (this was the removal process used in the screencast). Store the dataframe with these rows removed in **all\_rem**.

-Using **all\_rm**, create **X\_2** be a dataframe (matrix) of all of the numeric feature variables. Then, let **y\_2** be the response vector you would like to predict (Salary). Run the cell below once you have split the data, and use the result of the code to assign the correct letter to **question5\_solution**.

-Now, use **lm\_2\_model** to predict the **y\_2\_test** response values, and obtain an r-squared value for how well the predicted values compare to the actual test values.

-The number of reported salaries in the original dataset'

'The number of test salaries predicted using our model'

'If an individual does not rate stackoverflow, but has a salary'

'If an individual does not have a a job satisfaction, but has a salary'

'Our model predicts salaries for the two individuals described above.'