Tess Anderson

As a Data Science student, I am learning how to explore data and analyze it to gather information, answer questions, or draw conclusions. The first part of this process is called **Exploratory Data Analysis.** Exploratory data analysis is all about the preliminary investigation to understand the main characteristics of the dataset, often using data visualization methods like histograms or different graphic representations of the data. This helps to discover patterns, anomalies, and many other characteristics of the data. It also helps us determine which questions we will ask about the data, which hypothesis we’ll test, and how we’ll manipulate the data set to be best able to work with it.

This is one of the most important parts of data analysis as it focuses on understanding the data deeply before attempting to draw conclusions from it. Many people may take a simple step like “understanding the dataset” for granted and underestimate its importance; this step includes extracting important variables, identifying outliers, missing values, or human error, understanding the relationship between variables, and gaining a better context and insight into the data to minimize potential error and improve ability to find strong conclusions. Overall, Exploratory Data Analysis serves two important purposes: it cleans up a dataset and it allows for a better understanding of the relationships and variables in the dataset.

There are many tools available to the data scientist. Software like R, Python, and Microsoft Excel can help us read, manipulate, and learn from datasets in a CSV form. CSV means ‘comma separated value’ and it organizes data into columns and rows, like a spreadsheet. In addition to broad software like Python, we also use tools called packages which contain code for more specific purposes with more specific usage instructions. The packages I use for Exploratory Data Analysis are NumPy, Pandas, Matplotlib, and Seaborn. Using Python for exploratory data analysis can be overwhelming at first, but when done thoughtfully, step-by-step, it can open us up to new questions and answers we wouldn’t otherwise have access to.

Any data analysis starts with a data set, usually a CSV file. Exploratory data analysis starts by understanding the variables in the dataset. The first step is to read the file into Python. To do this, we use the Pandas package. Once we have access to the file in Python, there are countless options of how to explore and analyze the data. Guided by the goals on Exploratory data analysis, we start by calling a few Python commands that will return different summaries of the data: df.shape, df.head(), df.columns, and df.describe(). df, in this case is the name of the variable that stores the dataset, and the command following the “.” tells Python what information to output or return. After prompting Python to return several summaries of the data, we have lots of important qualities to look for or consider in the data. The df.describe() command give us descriptive statistics for each column in the datatset – we can see minimum and maximum entries, mean, count, standard deviation, and more. We will check these values to ensure that they’re reasonable and make sense with our intuition about the data; if the values are not reasonable, we should check for errors in entry or missing values.

Finding and correcting errors or vacancies in the data is called “cleaning” your dataset. Cleaning is an extremely important step in Exploratory Data Analysis – human error is more common than we’d like and if the data are not accurate, the conclusions will not be accurate. Human error isn’t the only issue we address in cleaning – we try to avoid data redundancy which occurs when the same piece of data is stored in more than one place. This may mean removing redundant columns. We must also check the dataset for missing values and deal with them properly. Depending on the situation, we might either drop or fill the null values using commands in Python. In cleaning, we might also address outliers.

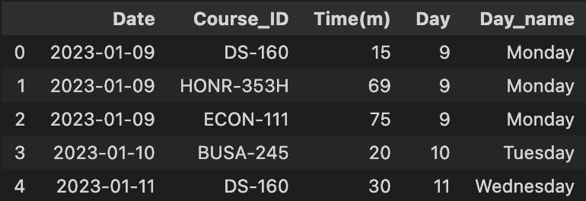
Finally, the data is clean and our last step in Exploratory Data Analysis to analyze the relationships between the variables. A correlation matrix which shows the correlation between all variables can be quickly composed using the Seaborn package in Python. Another quick way to understand relationships between data is by constructing a scatterplot which shows the values of two variables along two axes, so the relationship is easily seen. Pair plots are different graphical representations that also help visualize the relationships between variables. Histograms can be useful to show the distribution of a variable’s values – but not its relationship to another variable.

**Exploratory Data Analysis of My Study Time in Spring 2023 Semester**

Text

Description automatically generatedThe data set being analyzed is my personal study time sheet that I have been keeping track of throughout the semester. A summary of the data is shown to the left – this is the output from the command: df.info(). The columns in the original dataset are initials, date, course id, time in minutes, and summary. There are 27 entries in the data set.

My next task was to manipulate the dataset to prepare it for further analysis. First, I dropped the Initials column and the Summary column. The initials column was redundant and offered no useful information since each entry had my initials. The summary column was not useful for these purposes since the entries in this column were descriptions of the task performed and could not be easily categorized or meaningfully analyzed.



After dropping the necessary columns, I converted the Date column to a datetime object so that the day of the month and the day of the week could be easily extracted. Then, I added columns for the day of the month and day of the week using datetime commands from the Pandas package. After that, the dataframe is shown to the right.

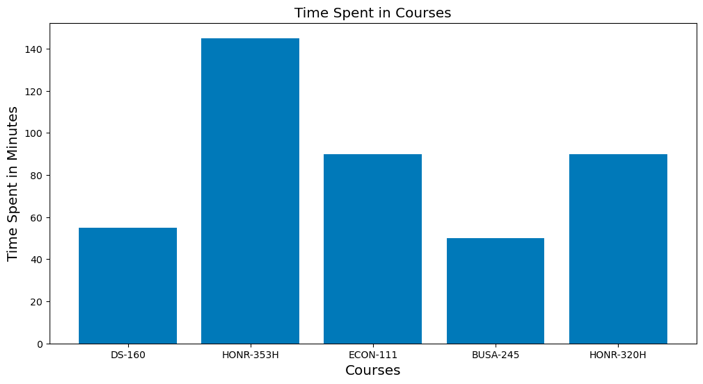
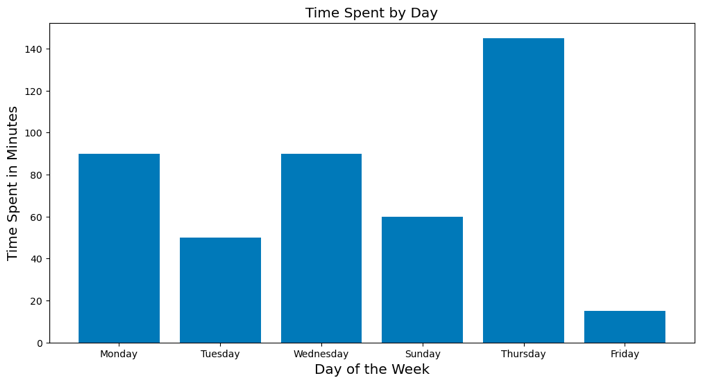
Graphical user interface, text

Description automatically generated

The point of Exploratory Data Analysis is to get familiar with the patterns and relationships in the data. I start by looking at the **average time spent studying each session**, so that I know what values are considered normal, high, or low. To find average time studying, I input the following into Python: df['Time(m)'].mean(). The output is **45.52 minutes.**

Next, I can do a query so that Python will return all of the entries where the time spent was higher than average. I input the command: df.loc[df['Time(m)'] > 46]. The results of this query are shown to the left.

Next, I examine how the time was distributed between the different factors of each variable. Below are two bar plots that show the total Time Spent broken up into relevant categories. Time Spent by Course is shown on the left – it is evident that I have spent the most time studying for HONR-353H out of all my classes this semester. Time Spent by Day of the Week is shown on the right – Thursday is much higher than the other bars. I spent more time on Thursdays than any other day, but this is influenced by a strong outlier in the dataset. I had one entry of 145 minutes that I spent on a large assignment in one sitting, which is abnormal compared to the rest of the entries. This is important to note since it may skew the data and conclusions to be misleading.

Chart, histogram

Description automatically generatedThe graphs above show the distribution of time grouped by different relevant categories. We can investigate the distribution of the variable Time independently by making a histogram of the data (shown to the right). This groups the different response values for time into bins and displays their frequency by height of bars. The histogram below also shows the mean in red. The distribution is right-skewed meaning the majority of the data are below the mean, but there are few influential large values that make the tail longer on the right than on the left of the mean. The outlier can be seen by the bar aroun 140 minutes with a count of 1. This explains the majority of the data being below the mean – the outlier brings the mean higher than the majority of the data would indicate.

Chart

Description automatically generated

I also look at a barplot that breaks the total time spent down by day, and color codes the bars according to which course was being studied. This visualization loosely highlights the courses with the longest study sessions, the most frequently occuring study sessions, and the day of the month which they occurred. The graph is shown to the left. The large outlier is clearly visible around day 9. There are several longer stretches of studying for Econ-111. I also notice the range of times logged is quite large. It appears that I worked in small chunks – completeing several tasks or sessions in one or two days, and then breaks for several days on weekends or when I fell behind on my studies.

Chart, histogram, box and whisker chart

Description automatically generated

A picture containing text

Description automatically generatedTo investigate the dispersion of the data, we look at the measures of dispersion. These include a graphic representation in the form of a boxplot (shown on the right) and descriptive representation using descriptive statistics shown in the table in black to the right. The table is the output given by input: df['Time(m)'].describe(). For the most part, the values of time range from 10 minutes to about 90 minutes. However, there is a large outlier at 145 which is shown on the plot by a small diamond.

Chart, box and whisker chart

Description automatically generated

Next, we can investigate this distribution of time more specifically by constructing a similar boxplot for the total time spent each day of the week. The graph is shown on the left.

Chart

Description automatically generated

Above are linear regression models, plotted for each course. The lines are the regression equation that best fits the points in the data set. None of them display strong, linear relationships. There are no reasonable conclusions that can be drawn from this analysis.

Chart, radar chart

Description automatically generated

Shown to the right is a pair plot. It gives lots of information at a glance about how I spent my time studying. It shows the longest time spent studying, the 145 minutes for HONR-353H. It shows the relationship between time and day for this dataset.

Chart

Description automatically generated with low confidence

Finally, I constructed a heatmap to visualize the relationship between time spent in minutes and day of the month. It shows the correlation between Day and Time spent. There is a weak relationship, as expected, between Time spent and the day of the month. This is evident by the correlation coefficient of 0.2, which is quite close to 0 indicating a weak relationship.