**Project 1: Transaction Analysis**

**CS6230 AI Tools**

**Learning Outcomes**

1. Identify, access, load, and prepare (clean) data sets for a given problem.

2. Select, apply, and interpret appropriate visual and statistical methods to analyze distributions of individual variables.

3. Clean and transform data for analysis.

4. Communicate findings through generated data visualizations and reports.

**Overview**

I simulated a data set of transactions for a pet supply store. You will analyze this data set to practice using Pandas, Seaborn, and Scipy and develop your data science skills and mindset. Using the data, you are going to prepare a report that answers a series of questions that would be relevant to the fictional business. Although Jupyter notebooks can be useful for interactive analysis of data, they are not always well suited for communicating results to others. For example, your audience make not need to see the code used to analyze the data, and you may generate more plots to help your understanding than you may want to share with your audience. As such, you should do your work in Jupyter notebooks but prepare a written report that that provides answers to the questions described below.

**Instructions**

**Part I: Setting Up a Local Python Environment**

We will be using Python 3, version 3.7 or later. If you are using a Mac or Linux environment, you will likely already have Python 3 installed. You can confirm this from the terminal like so:

$ python --version

Python 3.9.10

You may see Python 2.x listed like so:

$ python --version

Python 2.7.16

In that case, your system likely has both Python 2 and Python 3 installed. The Python executable will then be named python3:

$ python3 --version

Python 3.7.3

If you are using Windows, you will need to install Python and ensure that it is available in your path. You can [download installers](https://www.python.org/downloads/windows/) from the Python web site. The installer offers the option to add Python to your path automatically. You may need to open a new terminal session for that to take effect. Similarly, you can check that Python is installed and in your path from the command-line like so:

> python.exe --version

After the base Python language is installed, we will need to install a number of libraries. Python provides a feature called [“virtual environments”](https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments/#creating-a-virtual-environment) that allow you to install libraries for a specific project in the user’s directories (rather than system wide). This feature makes it convenient to work on different code bases that need different or conflicted versions of the same libraries. To create a virtual environment, do the following:

$ python -m venv cs6230

where cs6230 is the name of the directory that will be created to hold the contents. Once created, you won’t need to create it again. If you make a mistake or when you finish the class, you can uninstall it simply by deleting the directory.

Whenever you want to use the virtual environment, you will need to activate it. You can do that like so:

$ source cs6230/bin/activate

or if on Windows:

> cs6320\Scripts\activate.bat

Your shell prompt will now change, and you confirm the location of the Python interpreter:

(cs6230) $ which python

/Users/rnowling/cs6230/bin/python

or if on Windows:

> where python

In your environment, you can now install the libraries we will need. Python provides a program called pip for downloading and installing libraries. We will start by upgrading pip and installing a package called wheel that allows your system to install pre-compiled versions of libraries.

(cs6230) $ pip install -U pip wheel

After that, install the various libraries we need:

(cs6230) $ pip install numpy scipy cython matplotlib

(cs6230) $ pip install pandas sklearn seaborn

(cs6230) $ pip install jupyter

Now that everything is installed, you can start a Jupyter notebook environment:

(cs6230) $ jupyter notebook

This will start a web server in the console that allows you to create and run Jupyter notebooks in your browser. If your browser does not automatically, open look for a line like this in the console output and enter it in your address bar:

http://localhost:8888/?token=9ab2ce97ae7e59bf48112c51911fa64addca717bbf602104

Do not close the terminal window! Keep it running in the background.

Your browser should look something like this:

Graphical user interface, application

Description automatically generated

You can create a new notebook by going to the New menu and selecting Python 3:

Graphical user interface, application

Description automatically generated

A new tab will be opened with an empty notebook. You can type some Python code and click the Run button. You will see the output below.

**Graphical user interface, application

Description automatically generated**

When done, go to File -> Save and Checkpoint to save the notebook and File -> Close and Halt to close the notebook. When all of your notebooks are closed, you can kill the notebook server from the terminal using Ctrl-C. After that, you can close the terminal window.

**Part II: Familiarize Yourself with the Data**

You are provided with three files (transactions.tsv, products.tsv, and transactionProducts.tsv). These files store tabular data in text form with columns separated by tabs. The files each have a header line that indicates the column names.

Import the Pandas and Seaborn libraries. Use the Pandas read\_csv() function to create a DataFrame from each file. (Note: read the documentation for read\_csv() to see how to handle tab-delimited files.)

Explore the data set to get a sense of what it contains. Use the info() method to see how many columns there are, their names, their data types and whether the data set contains any null values. Use the head() method to look at the first few rows of each data set. Lastly, use the describe() method to calculate summary statistics such as the min, max, average, and standard deviation of the numerical variables and the number of unique values for non-numerical variables. (Note: read the documentation for describe() to see how to force it to print output for non-numerical variables.)

Validate the data and fix any issues: Are any values outside of expected ranges? Do variables have the right types? For example, non-numerical variables with low uniqueness counts might be better represented as categorical variables. (Hint: Rename the id fields to be consistent across tables to make joins easier in later steps (e.g., id -> transaction\_id).)

**Part III: Data Analysis and Interpretation**

Analyze the data in the notebooks using Pandas queries, visualization, and/or statistics and interpret the results to reach a conclusion and answer each question below.

(Note: It can be easier to answer some of these queries by joining the data into a single table ahead of time rather than doing joins for each query.)

1. How many transactions per customer? Assuming most people need to buy food for their pets on a regular basis, do you think that customers are visiting as regularly as they could be? Should the company invest in marketing or other strategies to increase the frequency of visits?
2. How many items are sold for each animal type? To which type of animals should the store invest the most space and marketing efforts on?
3. Are the number of transactions the same month-to-month or are their increases at certain times of year? Does the company need to consider hiring temporary employees for part of the year or should they focus on their long-term employees that work year round?
4. Stores need to decide how to manage their inventories. 10 most popular products and 10 least popular products.
5. If certain products are frequently purchased together, it can be useful to place these products physically adjacent to each other in the store or create incentives with discounts to increase sales of complementary products.
6. Are all of the products sold in equal amounts all year or are some products more popular during certain months? (Note: creating a plot for each product is excessive. Consider using Scipy's chisquare() statistical test to determine whether the product sale counts are uniformly distributed by month or not. Create plots for the products, if any, that are not uniformly distributed.)

**Part IV: Written Report**

Although Jupyter notebooks are great for exploring data, they do not always provide the best medium for presenting your results for others. You will use your notebooks to explore and probably won’t end up using all of the figures. Your audience may also be confused by the code.

Write up an accompanying report in which you provide answers to each question and provide supporting evidence for your conclusions in the form of selected counts, plots, and/or statistics that you generated.

Begin your report by providing a brief overview and summary of the data. Where did it come from? What does the data set contain? What is the overall size of the data set? (e.g., How many transactions? How many products? How many types of animals? What are the product categories and subcategories? How many customers?) Were there any limitations or data issues that could impact the accuracy of the analyses?

The report does not need to be too long, but it should be professionally written. Figures should be high quality with properly labeled axes and legible font sizes. (Look up how to control the DPI and font sizes in the generated images.) When figures are included, they should be referenced in the text and there should be explanation about how to read and interpret the figures.

**Submission Instructions**

Submit your report as a PDF and notebook(s) as HTML or PDF file(s) through Canvas.