DOUGLAS COLLEGE – Winter 2025

CSIS 3290 – 002 – Lab 02

- **Create a folder and rename it according to the folder structure and naming convention stated below**
- **All the files you are required to submit for the assignment should be placed inside this folder.**
- **You will lose points if you just cut and paste materials from close exercises (e.g., If I see the same comments, variable names, etc. from class exercises being using in your code).**
- **If cheating is determined (i.e., you shared your work with another student in the class), your work will a ZERO mark and you will face further consequences.**
- **Make sure to include all the necessary files to make sure that the code can run properly without producing any error**

In this lab, we will practice how to analyze and explore a clean dataset. You need study the demo code and do your own research to make sure that you can perform all the tasks describe below.

1. Create a python notebook named as **Lab2_ABcXXX** with A signifies the first letter of your **first name**, Bc signifies the first two letters of your **last name** and XXX denotes the last three digits of your **student ID**.

2. Create a markdown cell at the top of the Jupyter notebook to state the lab, **your name and student ID** with the correct heading.

3. For each of the following section, you need to create a **markdown heading cell** followed by a few code cells to complete the tasks. Please also put some comments in each code cell.
   a. **Load the python library**. Please load all the required python libraries in this section
   b. **Read the data**. Please load the csv you have ***prepared from Lab1*** and have a peek at the data by using the head() function. Then display the column information. Also display the summary of datatypes of your dataset.
   c. Find out the **summary statistics** of the dataset using describe(). However, pass the following parameter to see a more detail statistics, percentiles=[0.01, 0.25, 0.5, 0.75, 0.99]. Notice the distribution of the features. Look at the maximum and minimum values of all features. Do they make sense to you?
   d. Analyze and display some interesting information
      - Display the record(s) where the **price** is maximum. Display the record(s) where the **mileage** is maximum.
      - Display the record(s) where the **mpg** is maximum. What car make(s) and model(s) do these records belong too? If they belong to the same car **make** and **model**, look at the fuel_type of these records; do these records have similar fuel_type value? Are these records valid? Note: we will leave the records as it is.
      - Display the record(s) where the **mileage** is greater than its 99% percentile. What is the average price of these cars?
      - Display the record(s) where the **mileage** is the minimum. Are these data valid? Noticing the vast differences between the minimum, maximum and other percentile values of mileage, we can see that the mileage distribution is skewed and that it probably has outliers.
      - Find the records where the **engine_size** is the minimum. Considering that cars should have engine_size greater than zero. These records definitely not valid. Drop any rows where the engine_size is zero.
      - Use groupby() to find the average **price** of different car **make**.

   e. Display the correlation between features, focusing on the **price** that will be our target for prediction.

f. Univariate and Multivariate

- Display the distribution plot of the **price**. Notice that the distribution is a bit skewed. Create a new column **log_price** using np.log1p() and plot its distribution. Notice that the log_price has a better distribution, albeit with some outliers. Drop the original **price** column.
- Find the records of where the **log_price** is smaller than its 1% percentile or greater than 99% percentile. Drop these records. Plot the distribution of log_price. You should see a much better distribution.
- Display the distribution plot of the **mileage**. Notice that the distribution is right skewed with long tail on the right. Create a new column **sqrt_mileage** using np.sqrt() and plot its distribution. Notice that the **sqrt_mileage** has a better distribution, with some outliers. Drop the original **mileage** column.
- Find the records of where the **sqrt_mileage** is smaller than its 1% percentile or greater than 99% percentile. Drop these records. Plot the distribution of **sqrt_mileage**. You should see a much better distribution. Note: there may still be some outliers, but we will ignore it for now.
- Plot the distribution of column **year**. Let's decide to consider only cars that were made from 2010 onwards. Drop any records whose year is less than 2010. Plot the distribution of column year again.
- Use the countplot to display the distribution of car **make**.

g. Multivariate analysis

- Display a multivariate analysis plot for **log_price** against **mileage**. Can you see any relation at all? Write your observations in a markdown text cell.
- Display a multivariate analysis plot for **log_price** against another feature that you think important, and record your observations as markdown text

h. Reducing the number of values in categorical columns

- Use value_counts() to see the values of the categorical columns: **transmission**, **fuel_type**, **make**, and **model**.
- The **model** column has too many different values. Please drop column model. However, we could reduce the number of categories in the make column.
- There are too many car **make** categories in our dataset. However, you can see that the make's value count has a rather nice distribution. Thus we will use the average value of make's value_counts() and use it as the cut-off such that any record whose make's value counts is less than the cut-off will be assigned a new car manufacturer ***other***. Note, you can use the following code:

```
make_mean = df.make.value_counts().mean()
filter = df.make.value_counts() < m_mean
make_other = filter[filter==True].index.to_list()
```

  You can then use **make_other** to change the make value in those rows into **other**. When you use value_counts() on make feature again, you should see something like below:

```
ford          14037
volkswagen    11734
vauxhall      10442
merc           9926
Audi           8088
BMW            8027
other          4215
```

- There are only two cars whose **fuel_type** is **electric**. This can be considered an outlier. For now, we will leave it as it is. Similarly, there are only six cars whose **transmission** is **other**. This can be considered an outlier. For now, we will leave it as it is.

i. **Create dummy_features** for the categorical column.
   - Generate dummy values for the **fuel_type** column with the following parameters: **`drop_first=True`**, **`prefix='fuel'`** and **`dtype='int'`**. Join the dummy values to the dataframe either by using df.join() or pd.concat(). For example, assuming that the dummy values were saved as **`fl_dummy`**, you can use **`df = df.join(fl_dummy).`** After that, drop the **fuel_type** column from the dataframe
   - Similarly perform the above step for the **transmission** column. Use **`prefix='tr'`**.
   - For the **make** column, recall that we created category ***other***. We will NOT use drop_first. Instead, generate dummy values for the **make** column with the following parameters: **`prefix='mk'`** and **`dtype='int'`** (notice that we do not use drop_first parameter). Join the dummy values to the dataframe. After that, drop the **mk_other** and **make** columns from the dataframe
   - You should now have 19 columns and 66469 rows of data

j. Display the correlation to the log_price column. You can use abs() and sort_values() to see that some new columns have a relatively better correlation to the log_price column

k. Check the dataset's information again. Make sure all columns are numerical, and you do not have any null.

4. **Reset the index** (make sure to drop the new index) **and save the csv file** as Lab2_prepared_ABcXXX.csv (remember to use index=False)

Note on submission:
- Create a folder named as Lab2_ABcXXX following the naming convention.
- Put your Jupyter notebook and the original and cleaned dataset in this folder.
- Zip the file and submit it through the blackboard

LAB/ASSIGNMENT PRE-SUBMISSION CHECKLIST
- Did you follow the naming convention for your files?!
- Did you follow the naming convention for your folder?!
- Does your submission work on another computer?!
- Double check **before** submitting