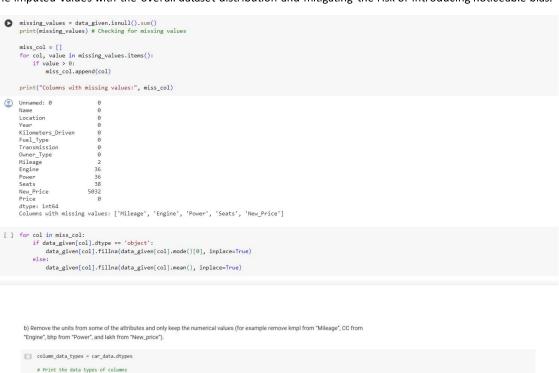
Principles of Data Science - 5530

Assignment-2

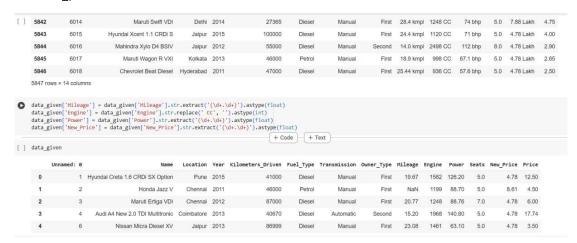
Name: Nikhil Jagadeesh Sriram

Student ID: 16352573

(A) In this context, I am addressing the identification and handling of missing values across all columns. For categorical columns, the missing values are being imputed with the mode. This approach is chosen to ensure the preservation of the existing data within that categorical field, thereby minimizing the potential for bias in the imputed values. As for numerical columns, missing values are being imputed with the mean. This method is adopted to maintain the distribution and central tendency of the available data in the numerical column, aligning the imputed values with the overall dataset distribution and mitigating the risk of introducing noticeable bias.



 (B) Strip the units from select attributes, retaining only the numerical values. For instance, eliminate "kmpl" from the "Mileage" column, "CC" from the "Engine" column, "bhp" from the "Power" column, and "lakh" from the "New_price" column.



(C) Convert the categorical variables, namely "Fuel_Type" and "Transmission," into numerical one-hot encoded values.

One-hot encoding is a technique used to convert categorical variables into a binary matrix where each category becomes a separate column, and a binary value (0 or 1) indicates the presence or absence of that category for each observation. In the context of your dataset:

For example, if you have a "Fuel_Type" column with categories 'Petrol' and 'Diesel', after one-hot encoding, you will have two new columns: 'Fuel_Type_Petrol' and 'Fuel_Type_Diesel'. For each row, the column corresponding to the actual fuel type will have a value of 1, while the other column will have a value of 0.

```
C) Change the categorical variables ("Fuel_Type" and "Transmission") into numerical one hot encoded value.

[] label_encoder = LabelEncoder()
data_given['fuel_Type_Label'] = label_encoder.fit_transform(data_given['fuel_Type'])
data_given['Transmission_Label'] = label_encoder.fit_transform(data_given['Transmission'])

onehot_encoder = OneHotEncoder(sparse=false)

fuel_type_encoded = onehot_encoder.fit_transform(data_given[['fuel_Type_Label']])
transmission_encoded = onehot_encoder.fit_transform(data_given[['Iransmission_Label']])

fuel_type_encoded_df = pd.DataFrame(fuel_type_encoded, columns=['Fuel_Type_' + str(i) for i in range(fuel_type_encoded.shape[1])])

transmission_encoded_df = pd.DataFrame(transmission_encoded, columns=['Transmission_' + str(i) for i in range(transmission_encoded.shape[1])])

data_given = pd.concat([data_given, fuel_type_encoded_df, transmission_encoded_df], axis=1)

data_given.drop(['Fuel_Type', 'Transmission', 'Fuel_Type_Label', 'Transmission_Label'], axis=1, inplace=True)
```

(D) Introduce a new feature by calculating the current age of each car and add this column to the dataset. One way to achieve this is by subtracting the "Year" value from the current year.

d) Create one more feature and add this column to the dataset (you can use mutate function in R for this). For example, you can calculate the current age of the car by subtracting "Year" value from the current year. current_year = datetime.now().year
data_given['Current_Age'] = current_year - data_given['Year'] data given 0 Location Year Kilometers_Driven Owner_Type Mileage Engine Power 4.78 12.50 Chennai 2011 NaN 1199 88.70 5.0 8.61 0.0 0.0 1.0 0.0 Chennai 2012 87000 First 20.77 1248 88.76 7.0 4.78 6.00 1.0 0.0 0.0 0.0 40670 15.20 1968 140.80 4.78 17.74 1.0

1461 63 10

5.0

4.78 3.50

(E) Perform select, filter, rename, mutate, arrange and summarize with group by operations on this dataset.

23.08

Jaipur 2013

In data manipulation, the **select** operation involves choosing specific columns, **filter** is used to subset rows based on conditions, **rename** is employed to change column names, **mutate** creates new columns, **arrange** sort rows based on specified criteria, and **summarize** with **group by** aggregates data by groups, calculating summary statistics for each group.



```
data_given = data_given.rename(columns={'New_Price': 'NewPriceCAR'})#rename
    data_given
(2)
            Unnamed:
                                    Location Year Kilometers_Driven Owner_Type Mileage Engine Power Seats N
                         Hyundai
      5847 rows × 18 columns
  [ ] data_given['Increseprice'] = data_given['NewPriceCAR']- data_given['Price']
      data_given
                         Name Location Year Kilometers_Driven Owner_Type Mileage Engine Power Seats NewPriceCAR Price Fuel_Ty
                       Hyundai
                                                                                                          4.78 12.50
        0
                                   Pune 2015
                                                       41000
                                                                    First 19.67 1582 126.20 5.0
                      CRDI SX
                        Option
      5847 rows × 19 columns
     4
     sorted car data = car data.sort values(by='Price', ascending=True)
      sorted_car_data#arrange
(2)
              Unnamed:
                                           Location Year Kilometers_Driven Owner_Type Mileage Engine Po
                       0
                             Tata Nano
                                                                              65000
       1660
                    1713
                                                Pune 2011
                                                                                                        26.00
                                                                                           Second
                                                                                                                    624
                                    Lx
     5847 rows × 19 columns
    average_price_by_fuel_type = data_given.groupby('Kilometers_Driven')['Mileage'].mean()#summarize
     average_price_by_fuel_type
 Kilometers_Driven
171 24.700000
    600
1000
              21.500000
18.493333
              22.415000
     1001
              17.180000
     480000
620000
               20.360000
              20.540000
     720000
     775000
     6500000 15.970000
Name: Mileage, Length: 3019, dtype: float64
[ ] data_given.to_csv('clean_data.csv', index=False)
```