PREDICTING THE PRIMARY BASIS OF CLASSIFICATION OF A CAR USING LOGISTIC REGRESSION

This project is part of the assessment for the Data Science intern at JVEC. The dataset for the analysis was provided by the company in a Google Drive

(https://drive.google.com/drive/folders/1OU7wORgIde8ael4GwZhzWQSZXJk14s4-?usp=sharing)

This report provides explanation to the Exploratory data analysis performed on the car dataset.
The project is divided into 3 parts:

1. Data Cleaning
2. Data Analysis
3. Model Development

Metadata:
For lack of metadata about the dataset, this information about the data was obtained from ChatGPT and should be interpreted with caution.

The descriptions below about the columns in the dataset was obtained from ChatGPT therefore, some may not be appropriate represenations.

1. `MachineID`: A unique identifier or serial number for each car.
2. `ModelID`: An identifier for the car's specific model.
3. `fiModelDesc`: A textual description of the car's model.
4. `fiBaseModel`: The base model of the car, which may include variations or trims.
5. `fiProductClassDesc`: A description of the product class to which the car belongs (e.g., compact, SUV, sedan, etc.).
6. `ProductGroup`: A categorization of the car into a specific product group (e.g., construction equipment, trucks, etc.).
7. `ProductGroupDesc`: A description of the product group.
8. `MfgYear`: The year in which the car was manufactured.
9. `fiManufacturerID`: An identifier for the car's manufacturer.
10. `fiManufacturerDesc`: A description of the car's manufacturer.
11. `PrimarySizeBasis`: The basis for determining the car's primary size (e.g., weight, dimensions).
12. `PrimaryLower`: Lower limit or value related to the car's primary size.
13. `PrimaryUpper`: Upper limit or value related to the car's primary size.

These columns provide information about the car's characteristics, including its model, manufacturer, year of manufacture, size specifications, and product classification. They can be valuable for analyzing and categorizing cars in the dataset, as well as for conducting various types of research and analysis related to the automotive industry.

Data Cleaning:

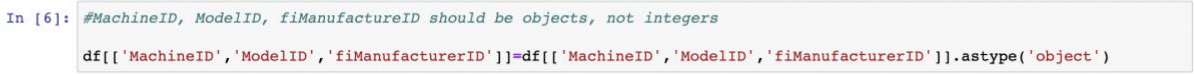
Loading the Dataset and Initial Exploration

- The dataset, 'Machine_Appendix.csv', was successfully loaded.
- Initial exploration of the dataset was performed using df.head() to display the first few rows.

- Column data types were checked using df.dtypes

Data Type Conversion

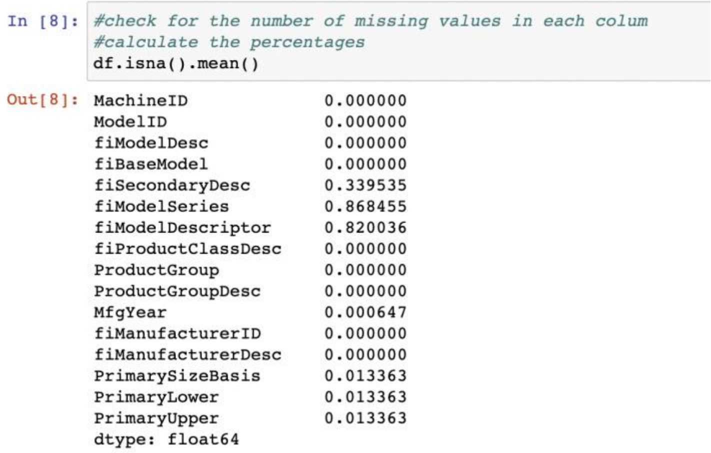
Three columns, 'MachineID', 'ModelID', and 'fiManufacturerID', were identified as needing data type conversion from integers to objects, which was done with the following code:



```
In [6]: #MachineID, ModelID, fiManufactureID should be objects, not integers

        df[['MachineID','ModelID','fiManufacturerID']]=df[['MachineID','ModelID','fiManufacturerID']].astype('object')
```

Handling Missing Values

The percentage of missing values in each column was calculated using *df.isna().mean().*



```
In [8]: #check for the number of missing values in each colum
        #calculate the percentages
        df.isna().mean()

Out[8]: MachineID             0.000000
        ModelID               0.000000
        fiModelDesc           0.000000
        fiBaseModel           0.000000
        fiSecondaryDesc       0.339535
        fiModelSeries         0.868455
        fiModelDescriptor     0.820036
        fiProductClassDesc    0.000000
        ProductGroup          0.000000
        ProductGroupDesc      0.000000
        MfgYear               0.000647
        fiManufacturerID      0.000000
        fiManufacturerDesc    0.000000
        PrimarySizeBasis      0.013363
        PrimaryLower          0.013363
        PrimaryUpper          0.013363
        dtype: float64
```

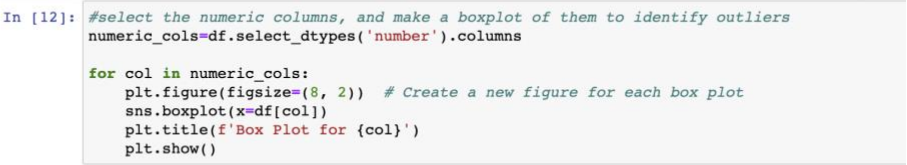
Columns with a high percentage of missing values were identified, including 'fiSecondaryDesc', 'fiModelSeries', and 'fiModelDescriptor'.

Rows with missing values in columns like 'PrimarySizeBasis', 'PrimaryLower', 'PrimaryUpper', and 'MfgYear' were dropped.

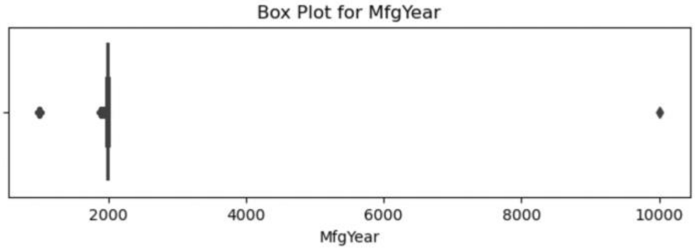
The columns 'fiSecondaryDesc', 'fiModelSeries', and 'fiModelDescriptor' were dropped from the dataset.

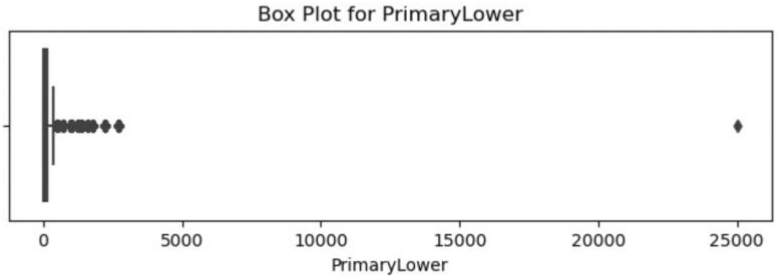
Handling Outliers

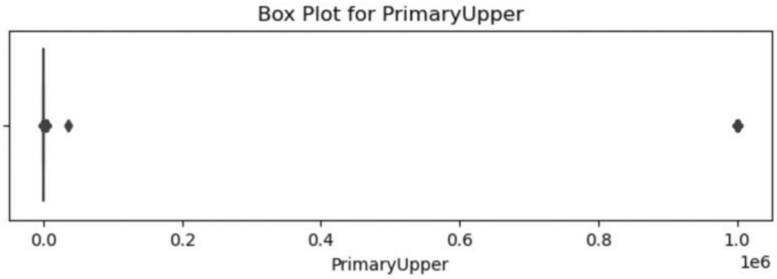
A box plot of the numeric columns was displayed to identify the outliers in the dataset.



```
In [12]: #select the numeric columns, and make a boxplot of them to identify outliers
         numeric_cols=df.select_dtypes('number').columns

         for col in numeric_cols:
             plt.figure(figsize=(8, 2))  # Create a new figure for each box plot
             sns.boxplot(x=df[col])
             plt.title(f'Box Plot for {col}')
             plt.show()
```







The 'MfgYear' column was examined to detect any invalid years, and the maximum and minimum years were identified. Rows with incorrect years were dropped.

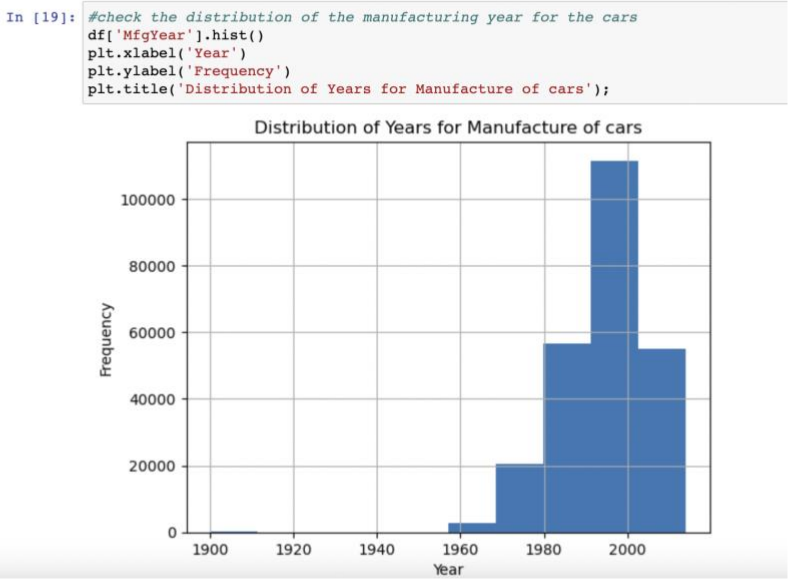
Outliers in 'PrimaryLower' and 'PrimaryUpper' were identified and removed. Specifically, rows with 'PrimaryUpper' greater than 190 and 'PrimaryLower' greater than 150 were dropped. (using the 75th percentile from the dataset).

Data Analysis:

In the 'Data Analysis' section, various tasks were performed on the cleaned dataset to gain insights and explore the relationships between variables. Here is a summary of the actions:

Distribution of Manufacturing Year

A histogram was created to visualize the distribution of the manufacturing years for cars. It was observed that most cars in the dataset were relatively recent, ranging from 1960 to 2014, with fewer cars from years before 1960.

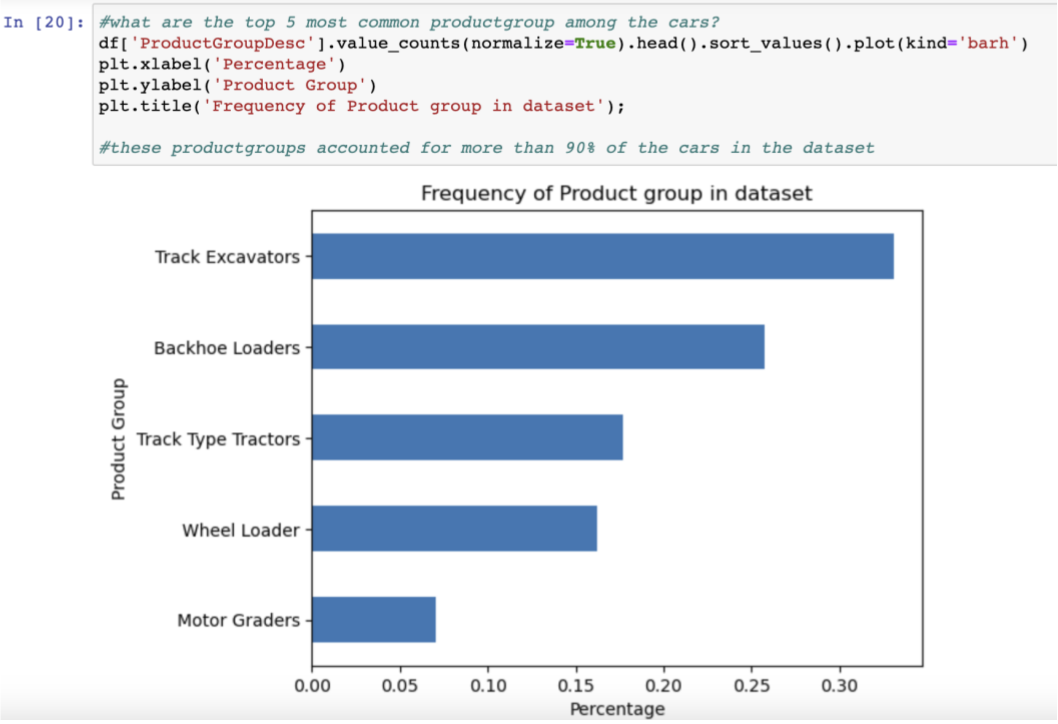


```
In [19]: #check the distribution of the manufacturing year for the cars
         df['MfgYear'].hist()
         plt.xlabel('Year')
         plt.ylabel('Frequency')
         plt.title('Distribution of Years for Manufacture of cars');
```

Frequency of Product Groups

The top 5 most common product groups among the cars were analyzed.

Product groups, such as Track Excavators (TEX), Backhoe Loaders (BL), Track Type Tractors (TTT), Wheel Loaders (WL), and Motor Graders (MG), collectively accounted for more than 90% of the cars in the dataset.

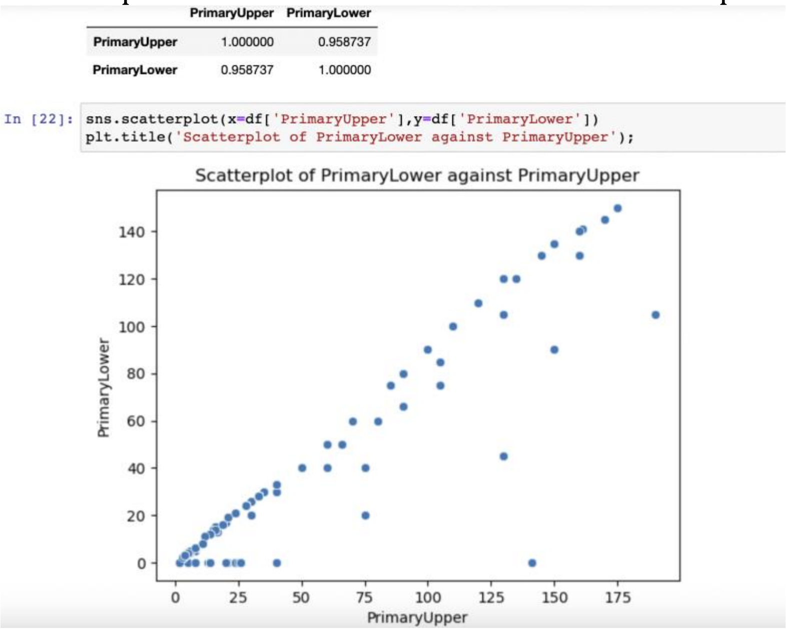


```
In [20]: #what are the top 5 most common productgroup among the cars?
         df['ProductGroupDesc'].value_counts(normalize=True).head().sort_values().plot(kind='barh')
         plt.xlabel('Percentage')
         plt.ylabel('Product Group')
         plt.title('Frequency of Product group in dataset');

         #these productgroups accounted for more than 90% of the cars in the dataset
```

Correlation Analysis

The correlation between 'PrimaryUpper' and 'PrimaryLower' was calculated, showing a high correlation of 95%.

A scatterplot was created to visualize this relationship.



	PrimaryUpper	PrimaryLower
PrimaryUpper	1.000000	0.958737
PrimaryLower	0.958737	1.000000

```
In [22]: sns.scatterplot(x=df['PrimaryUpper'],y=df['PrimaryLower'])
         plt.title('Scatterplot of PrimaryLower against PrimaryUpper');
```

Range of Primary Measurements

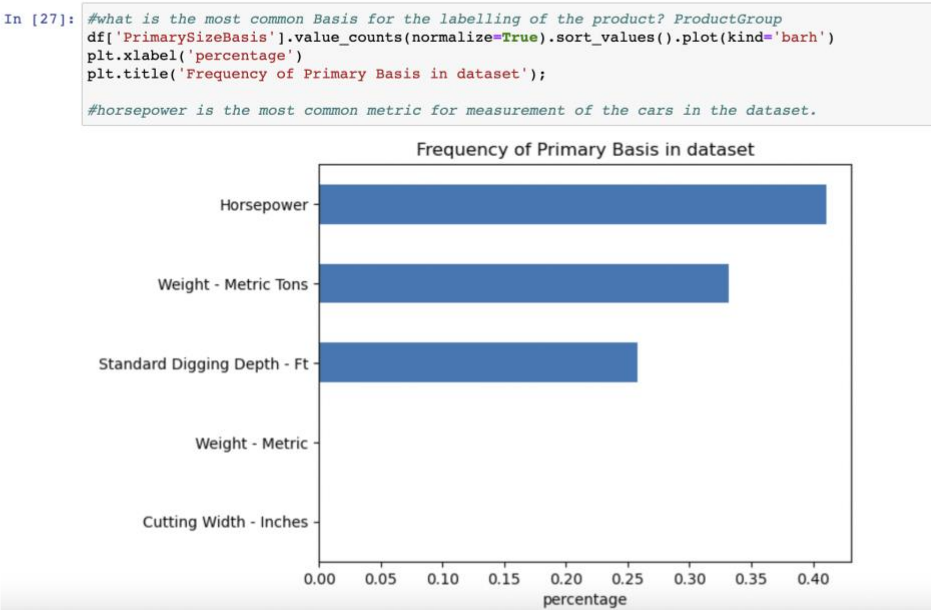
A new column, 'Range_PrUpper_Lower,' was created to represent the difference between 'PrimaryLower' and 'PrimaryUpper.'

The correlation between 'MfgYear' and 'Range_PrUpper_Lower' was calculated and visualized with a scatterplot, helping to understand how the range of primary measurements changes over the years.

Frequency of Primary Size Basis

The most common basis for labeling the product's primary size was explored.

It was found that horsepower was the most common metric for measuring cars in the dataset.

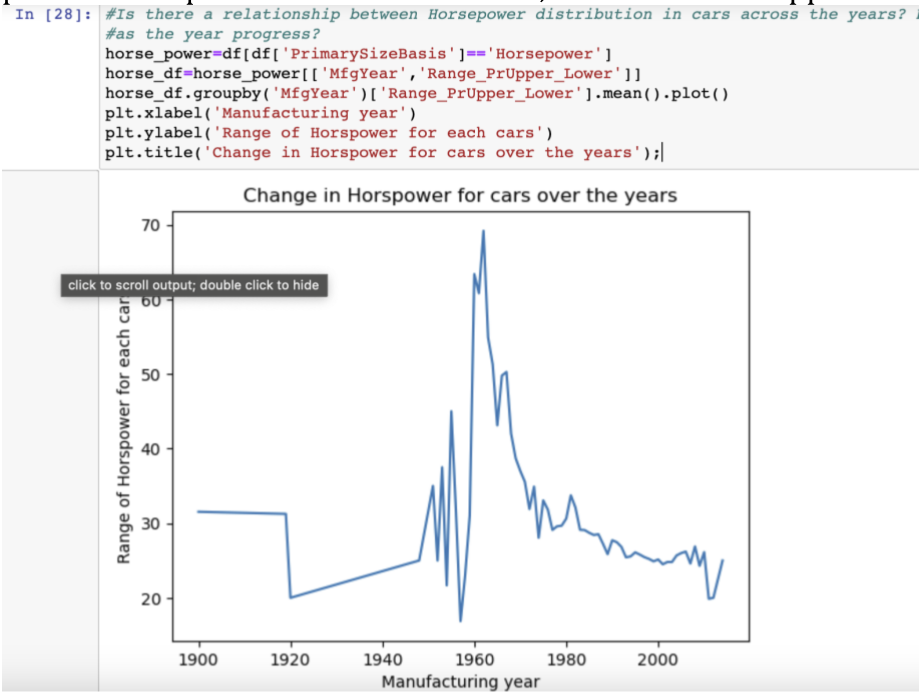


```
In [27]: #what is the most common Basis for the labelling of the product? ProductGroup
         df['PrimarySizeBasis'].value_counts(normalize=True).sort_values().plot(kind='barh')
         plt.xlabel('percentage')
         plt.title('Frequency of Primary Basis in dataset');

         #horsepower is the most common metric for measurement of the cars in the dataset.
```

Horsepower Analysis Over Time

The relationship between the distribution of horsepower in cars across different years was analyzed.

The analysis showed changes in the horsepower range as the years progressed, with a peak in horsepower around 1960-1970, after which it dropped.



```
In [28]: #Is there a relationship between Horsepower distribution in cars across the years?
         #as the year progress?
         horse_power=df[df['PrimarySizeBasis']=='Horsepower']
         horse_df=horse_power[['MfgYear','Range_PrUpper_Lower']]
         horse_df.groupby('MfgYear')['Range_PrUpper_Lower'].mean().plot()
         plt.xlabel('Manufacturing year')
         plt.ylabel('Range of Horspower for each cars')
         plt.title('Change in Horspower for cars over the years');
```

Change in Horspower for cars over the years
click to scroll output; double click to hide
Range of Horspower for each car
70
60
50
40
30
20
1900
1920
1940
1960
1980
2000
Manufacturing year

Weight-Tons Measurement Over Time

A similar analysis was conducted to explore changes in the weight-tons measurement of cars over the years.

A peak was observed around 1960-1970, indicating that cars manufactured during that period had higher weight-tons measurements.

Standard Digging Depth Over Time

The change over the years in the standard digging depth for cars, focusing on the 'Standard Digging Depth - Ft' basis, was examined.

The analysis showed that the increase in standard digging depth for cars only started to climb after 1980.

These analyses provided valuable insights into the dataset and helped in understanding the distribution and relationships between variables.

Model Development

In the 'Model Development' section, a basic Logistic regression model was built to predict the target variable 'PrimarySizeBasis.' The following steps were taken:

Data Splitting

The dataset was split into training and testing sets. Some of the categories in the target variable were greatly undersampled, they were therefore dropped.

Data Encoding

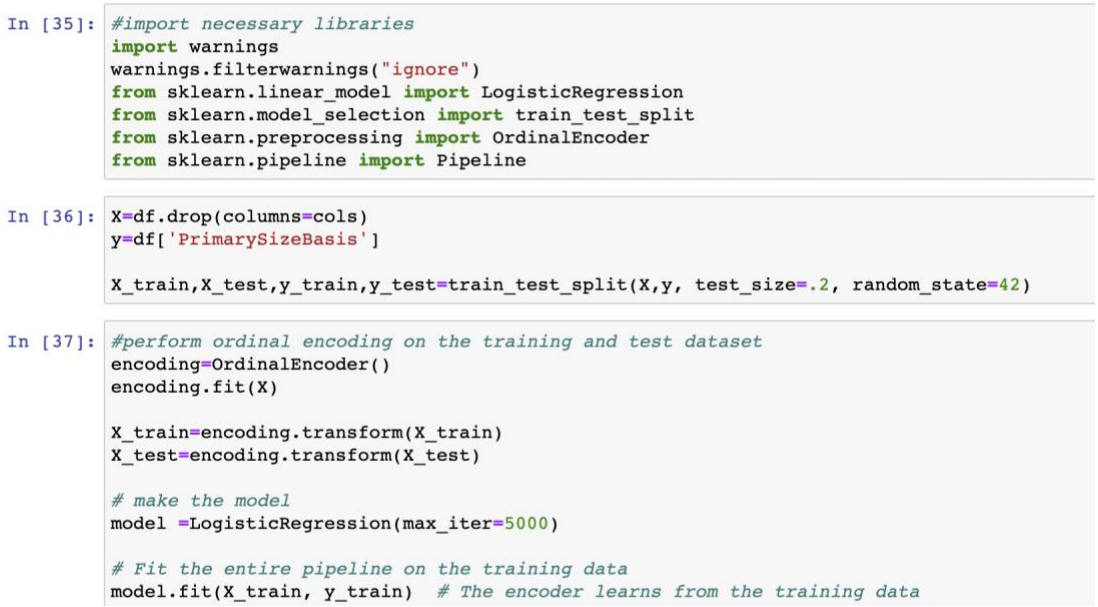
Ordinal encoding was applied to the training and test datasets to prepare them for modeling.

Model Selection

A logistic regression model was chosen for this classification task.

Model Training

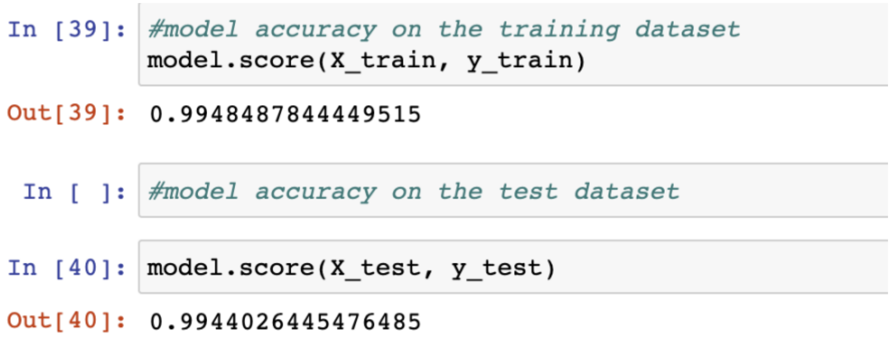
The logistic regression model was trained on the training data.



```
In [35]: #import necessary libraries
         import warnings
         warnings.filterwarnings("ignore")
         from sklearn.linear_model import LogisticRegression
         from sklearn.model_selection import train_test_split
         from sklearn.preprocessing import OrdinalEncoder
         from sklearn.pipeline import Pipeline

In [36]: X=df.drop(columns=cols)
         y=df['PrimarySizeBasis']

         X_train,X_test,y_train,y_test=train_test_split(X,y, test_size=.2, random_state=42)

In [37]: #perform ordinal encoding on the training and test dataset
         encoding=OrdinalEncoder()
         encoding.fit(X)

         X_train=encoding.transform(X_train)
         X_test=encoding.transform(X_test)

         # make the model
         model =LogisticRegression(max_iter=5000)

         # Fit the entire pipeline on the training data
         model.fit(X_train, y_train)  # The encoder learns from the training data
```

Model Evaluation

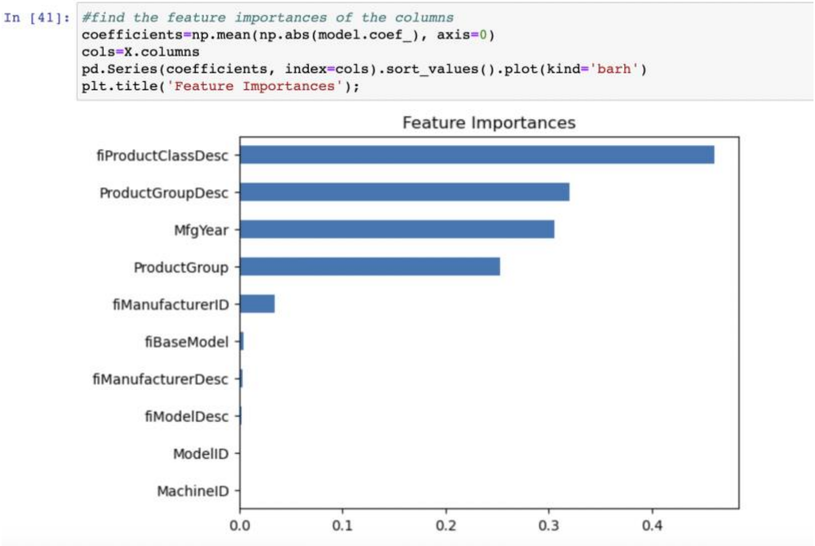
The model's performance on the testing data was evaluated, and it was found that it achieved high accuracy (99%) on both the training and test datasets.



```
In [39]: #model accuracy on the training dataset
         model.score(X_train, y_train)

Out[39]: 0.9948487844449515

In [ ]: #model accuracy on the test dataset

In [40]: model.score(X_test, y_test)

Out[40]: 0.9944026445476485
```

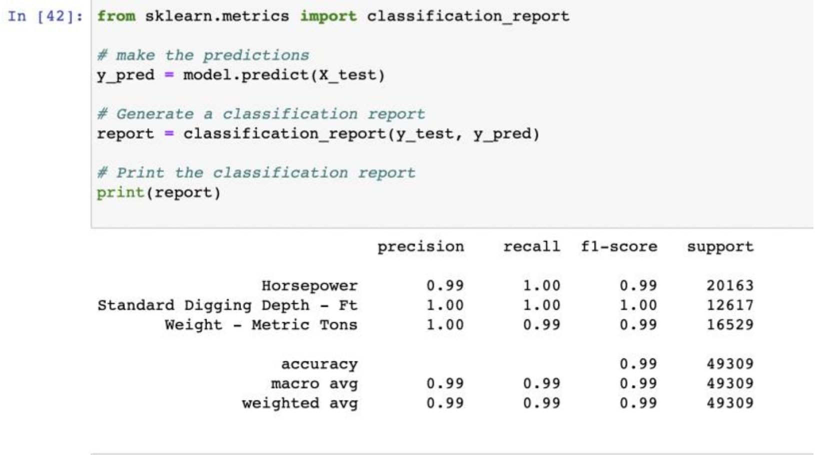
Feature Importance

The feature importances were determined, and it was identified that 'Product class Description' and 'Manufacture Year' had the most impact on the model's predictions.



```
In [41]: #find the feature importances of the columns
         coefficients=np.mean(np.abs(model.coef_), axis=0)
         cols=X.columns
         pd.Series(coefficients, index=cols).sort_values().plot(kind='barh')
         plt.title('Feature Importances');
```

Classification Report



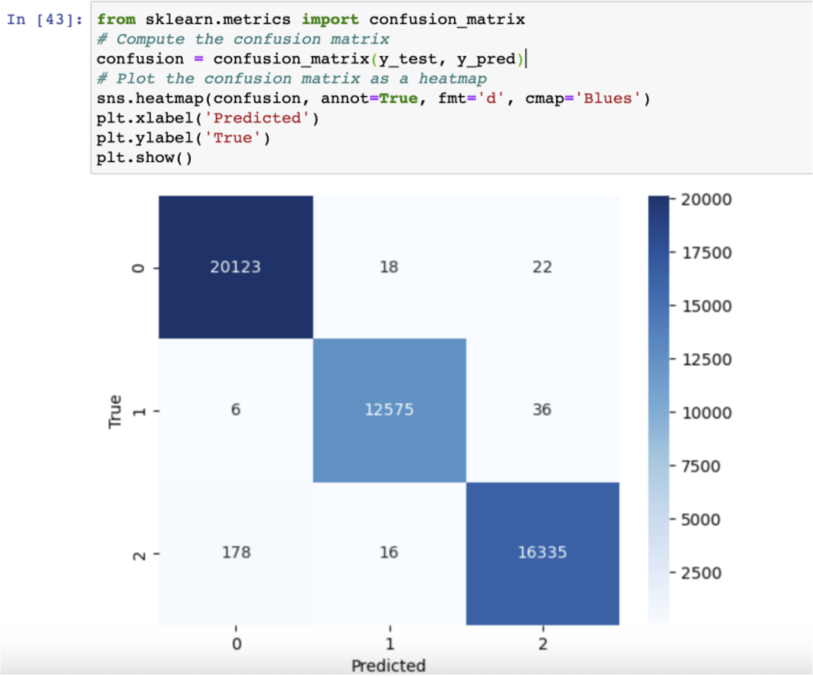
```
In [42]: from sklearn.metrics import classification_report

         # make the predictions
         y_pred = model.predict(X_test)

         # Generate a classification report
         report = classification_report(y_test, y_pred)

         # Print the classification report
         print(report)

                          precision    recall  f1-score   support

              Horsepower       0.99      1.00      0.99     20163
Standard Digging Depth - Ft       1.00      1.00      1.00     12617
      Weight - Metric Tons       1.00      0.99      0.99     16529

                accuracy                           0.99     49309
               macro avg       0.99      0.99      0.99     49309
            weighted avg       0.99      0.99      0.99     49309
```

A classification report for the model was generated, providing details on precision, recall, F1-score, and support for each class.

Confusion Matrix

A confusion matrix was created and visualized to understand the model's predictions.

In conclusion, data cleaning was successfully performed, insightful data analysis was conducted, and a machine learning model was built to predict the primary size basis for cars in the dataset. The model demonstrated high accuracy, and the data analysis provided valuable insights into the dataset.



```
In [43]: from sklearn.metrics import confusion_matrix
         # Compute the confusion matrix
         confusion = confusion_matrix(y_test, y_pred)
         # Plot the confusion matrix as a heatmap
         sns.heatmap(confusion, annot=True, fmt='d', cmap='Blues')
         plt.xlabel('Predicted')
         plt.ylabel('True')
         plt.show()
```

Summary and Conclusion

From this analysis, we can observe that the year a car was manufactured and the description of the product class influences the model's performance largely in predicting which Primary Basis will be used to obtain the lower and higher boundaries.

The distribution of the manufacturing year for the cars in the dataset shows that most cars were manufactured between 2000 and 2010.

Limitations:

The metadata was not provided and hence a proper interpretation of the variables was not possible.

Further Improvements (Given much time)

The model's accuracy can be finetuned to nearly 100% by hyperparameter tuning. Further exploratory data analysis could also reveal why there was a spike in the 1960s for the cars whose primaryBasis was Weight - Metric Tons and Horsepower