**Introduction:**

Purchasing a car brings many benefits such as time-saving, convenience and mobility for individuals to travel to work, run errands, and meet family and friends by saving time to commute with public transportation. That is why purchasing a car can be beneficial to many individuals.

**Problem statement:**

Consumers usually view the car model’s aesthetics, specifications, and safety level and then do a test drive before purchasing it for their use. With a wide range of car models available in the market, it can be challenging and time-consuming for potential car buyers to evaluate the vehicle effectively.

**Objective:**

To develop a machine learning model that can evaluate existing vehicles in the market accurately to streamline processes and assist consumers in making informed decisions.

Using machine learning to assess the cars’ input attributes, potential buyers can quickly filter out vehicles with unacceptable class types. This saves time for consumers who can focus on exploring better car options that align with their preferences and style.

**Dataset:**

Therefore, I found a dataset on Car Evaluation from the UC Irvine Machine Learning Repository, https://archive.ics.uci.edu/dataset/19/car+evaluation, which is used to create models to evaluate cars.

The dataset contains several attributes including:

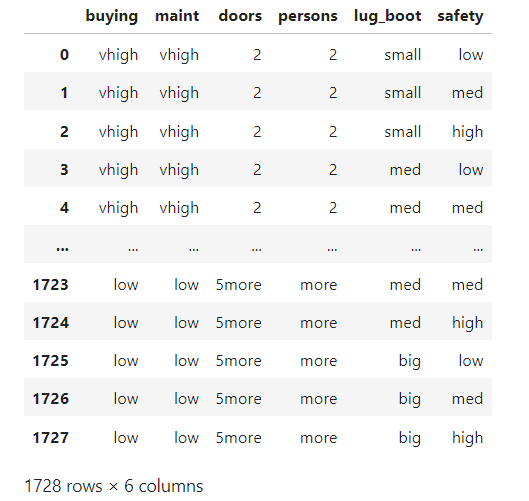
* Buying price (buying)
* Price of maintenance (maint)
* Number of doors (doors)
* Capacity in terms of persons to carry (persons)
* The size of luggage boot (lug\_boot)
* Estimated safety of the car (safety)
* Evaluation level (class)

**class** attribute is the target variable of this dataset, while the rest of the attributes are predictor variables.

**Exploratory Data Analysis:**

Here are some brief data descriptions:

1. There are 1728 data rows
2. There are no missing values in this dataset.
3. All data from each column are in object format, meaning it is a categorical data type. Furthermore, after looking closely at the dataset, they are all ordinal data types, meaning there is an order to the data.
4. Data from predictor variables (buying, maint, doors, persons, lug\_boot, safety) are evenly distributed.
5. However, data from target variable (class) are not evenly distributed, with majority of the data class belonging to unacceptable (unacc), followed by acceptable (acc), then by good (good) and lastly very good (vgood).



A screenshot of a computer

Description automatically generatedA white background with black text

Description automatically generated

A bar chart of buying

Description automatically generated

A bar chart of mains

Description automatically generated

A bar chart of doors

Description automatically generatedA bar chart with blue rectangular bars

Description automatically generatedA bar chart of lug boot

Description automatically generatedA bar chart of safety

Description automatically generated

A bar chart with blue bars

Description automatically generated

**Data Preprocessing:**

I imported scikit-learning, a machine learning library for Python Programming that only supports numeric data values. Therefore, I changed all the predicted variables that are nominal data type to numeric data type.

Before changing it to numeric data type, I first ordered the data values from each column.

A screenshot of a computer

Description automatically generated

After that, I used OrdinalEncoder() from category\_encoders library to encode categorical data columns in certain order. The categorical data is mapped to an integer (1 to 4 or 1 to 3) as shown below.

A screen shot of a computer code

Description automatically generated

OrdinalEncoder(cols=['buying', 'maint', 'doors', 'persons', 'lug\_boot', 'safety'],

mapping=[{'col': 'buying', 'data\_type': dtype('O'), 'mapping':

low 4

med 3

high 2

vhigh 1

dtype: int64},

{'col': 'maint', 'data\_type': dtype('O'), 'mapping':

low 4

med 3

high 2

vhigh 1

dtype: int64},

{'col': 'doors', 'data\_type': dtype('O'), 'mapping':

2 4

3 3

4 2

5more 1

dtype: int64},

{'col': 'persons', 'data\_type': dtype('O'), 'mapping':

2 3

4 2

more 1

dtype: int64},

{'col': 'lug\_boot', 'data\_type': dtype('O'), 'mapping':

small 3

med 2

big 1

dtype: int64},

{'col': 'safety', 'data\_type': dtype('O'), 'mapping':

low 3

med 2

high 1

dtype: int64}])

A screenshot of a computer

Description automatically generated

After that, I used train\_test\_split (X, y, test\_size = 0.2, random\_state = 42) from sklearn.model\_selection package to split the data set into 80 % training data and 20% testing data with a random state of 42.

**Machine Learning Solution:**

I implemented the Decision Tree algorithm to evaluate the car class properly. I used both the entropy criterion and gini-index criterion for the Decision Tree modelling. Here are the feature importances and decision trees plotted.

**Entropy**

A graph with blue and white bars

Description automatically generated A diagram of a computer algorithm

Description automatically generated with medium confidence

**Gini-Index**

A bar graph with blue and white bars

Description automatically generated

A diagram of a computer algorithm

Description automatically generated with medium confidence

**Solution Evaluation:**

After constructing the decision tree models, I then put the model to test by predicting X\_test data to predict its following classes. The predicted values is then compared to the actual class label (Y\_test) to see the accuracy, precision, recall, f1-score and plot the confusion matrix. Here are the results:

|  |  |  |
| --- | --- | --- |
| Item | Entropy | Gini-Index |
| Training Performance | 0.863 | 0.867 |
| Generalisation Performance | 0.879 | 0.890 |

**Entropy**

A blue squares with white text

Description automatically generated

A screenshot of a computer screen

Description automatically generated

**GINI-Index**

A diagram of a graph

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

**Evaluation & Conclusion:**

**Safety level**, **capacity of the car** and **maintenance price** are the top three features with the highest feature importance. This makes sense as safety is extremely important in purchasing a car as it protects the drivers, passengers, and pedestrians. Moreover, the capacity of the car and maintenance price of the car are also considered by people as it can be inconvenient if the car is too small or and maintenance price is too high.

Both models produce a similar performance in terms of training and generalisation (test-set accuracy). Entropy scored (0.863, and 0.879) while the GINI-index scored (0.867,0.890) for training and generalisation respectively.

The generalisation performance for both models is better than the training performance and not much difference. Thus, there the model does not seem to overfit.

Moreover, from the confusion matrix, both model's results are quite similar and their f1-score for the class (except for **good**) scores more than 0.70.

**Shortcomings:**

There are some shortcomings as well.

* Insufficient data points. The data size is small, with only 1728 data rows. This can be challenging for the model to identify which features are truly predictive.
* Uneven class label distribution. The **good** and **very good** class labels are too little. The testing dataset has too few **good** and **very good** class to test on. This may lead to a biased model as the dataset does not full represent the cars class distribution and it does not fully capture the features of good and very good cars.
* Unknown source of data inputs. As each car is different, there are no proper methodologies used to rank the safety, buying price and maintenance price as different cars would have different pricing and materials used. There needs to be a proper methodology to state how such data is evaluated otherwise the data can be biased.