**Chapter 1: Data Preprocessing**

The given dataset, 'Cars93.csv,' has a shape of 94 rows and 26 columns. To prepare the data for a machine learning model, various preprocessing techniques such as data type classification, handling missing values, noise reduction, encoding categorical variables, normalization, and data splitting are implemented.

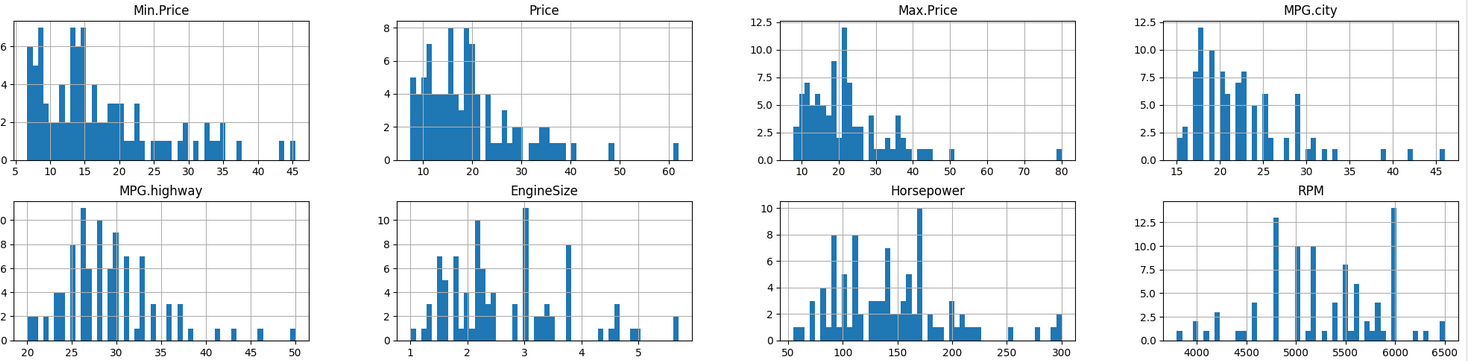
**1.1 Classification & Handling Missing values**

Classification – 1) Model: Nomimal, 2) Type : Nominal, 3) Max.Ratio : Ratio, 4) Airbags : Ordinal

A custom function, simple\_imputation, is implemented to handle missing values by replacing them with mean, median, mode, or a custom value. Mean imputation is used for numerical attributes like "Rear.seat.room" and "Luggage.room," while mode imputation is applied to categorical attributes like "AirBags."

**1.2 Noise Reduction, Encoding & Normalization**

After plotting the histogram of each feature of cars data (shown in Fig 1.A), it shows many features such as “Price”, “Max.Price”, “MPG.city”, “MPG.highway”, “EngineSize”, “Horsepower”, etc contains the outliers. Hence the IQR technique is used to detect and remove the outliers. The function “IQR\_outlier\_detection” effectively removes the datapoint which lies outside the lower and upper bound.



**Fig 1.A**

The dataset has features such as “Manufacturer”, “Model”, “Type” and “AirBags” which contains non-numerical object. The function “encoding\_data” uses OrdinalEncoder library function form scikit learn library to encode non-numerical object into unique number. OrdinalEncoder preserves the categorical order of features.

The features have varying scales hence to standardize numerical features, Z-score normalization was applied using “StandardScaler” from scikit-learn.

**1.3 Data-Splitting**

In function “split\_data”, the dataset is shuffled by using “random.permutation” function of NumPy library and split into 70:20:10 ratio using the index.

**1.4 Conclusion**

The preprocessing steps, including noise removal, normalization, and data shuffling, ensure the dataset is clean, standardized, and balanced for machine learning. Noise removal addresses data imbalance, normalization balances feature weights foraccurate predictions, and shuffling before splitting eliminates imbalance in training and test sets.

**Chapter 2: Linear & Logistic Regression**

The linear regression model is used to visualize the relationship between height and weight, implemented both using Scikit-Learn and manually. Meanwhile, the logistic regression model is a classification tool designed to predict outcomes, such as whether a purchase was made or not.

**2.1 Linear Regression**

**2.1.1 With Scikit Learn**

The “LinearRegression” model from scikit-learn was trained on the height-weight dataset. The coefficient obtained from the trained model was printed. The scatter plot of data with linear regression model line shows that model is fitted well.

**2.1.2 Without Scikit Learn**

The model was implemented using gradient descent. The cost function was minimized iteratively using learning rate (alpha) and a fixed number of iterations. The parameters (b0, b1) were updated in each iteration. If the number of steps (iterations) are lower, then the learning rate (alpha) value is ineffective, and model does not fit into the data.

**2.1.3 Conclusion**

* Both models produced similar results with comparable slopes.
* The manually trained model showed a slightly better fit compared to the scikit-learn model, though both produced similar overall performance.
* In manual training of the model, it requires more time to find optimized hyperparameters than training model with Scikit learn.

**2.2 Logistic Regression**

To train a model using logistic regression, the dataset (logistic\_regression\_dataset.csv) is loaded using pandas. The categorical feature such as Gender is encoded into numerical values. Also the “LogisticRegression” model from scikit-learn is train against the available data.

**2.2.1 Conclusion**

The model demonstrated accuracy of 80%, indicating good performance. The presence of 16 false negatives suggests that the model sometimes fails to correctly identify actual buyers, while the 8 false positives indicate false identification non-buyers as buyers. Additionally, the decision boundary reveals that most non-buyers are individuals with low estimated salaries, whereas most buyers have high estimated salaries.

**Chapter 3: Support Vector Machine (SVM)**

Support Vector Machines (SVM) are used to classify the target variable by using different kernels and regularization values. The dataset is analyzed to identify relevant features, potential skewness, and separability of data.

**3.1 Data Processing and Analysis**

The heatmap indicated a strong positive correlation between *Age* and *Experience* and a moderate correlation between *Income* and *CCAvg*. Therefore, *Age*, *Income*, and *CCAvg* were selected as the primary features. While the histogram analysis revealed that *CCAvg* and *Income* were left-skewed, indicating potential imbalance in their distributions.

The target variable distribution showed class imbalance, with **71%** samples labeled as 0 and **28%** samples labeled as 1.

The data is normalized using z-scalar normalization because of different varying scales of features which affect the accuracy of model

**3.2 Conclusion**

For LinearSVC model when trained with set of hyperparameters, the best accuracy observed was 47.4% for C = 0.01.

For very small values of *C (0.0001)*, accuracy dropped to **45.6%**, indicating underfitting.

Increasing *C* beyond 0.01 did not improve performance, indicating that linear separation is ineffective for this dataset.

For the same dataset SVM with linear kernel gives maximum accuracy of **36.8%** for C = 10 and 100 which suggests that linear separation is insufficient.

While SVM with polynomial kernel gives accuracy of **32.6%** which shows the polynomial transformation is inefficient in classification.

Also, RBF kernel give maximum accuracy of **52.4%** for C = 1 and Sigmoid kernel give maximum accuracy of **54.4%** for C = 0.1 which indicates the dataset has non-linear patterns that cannot be captured by a simple hyperplane.

The SVM model accuracy can be increased by balancing the label samples, feature engineering, hyperparameter optimization, etc.

**Chapter 4: Decision Tree & Random Forest**

The Iris dataset provided by scikit learn library contains 150 samples of iris flowers, each belonging to one of three species: *setosa*, *versicolor*, and *virginica*. Each sample includes four features: sepal length, sepal width, petal length, and petal width. The decision tree and random forest classifiers are plotted against this dataset

**4.1 Data Preprocessing and Analysis**

* The Iris dataset is small, and it does not have any null or missing data.
* The histogram plotted against the dataset shows that the distribution of "sepal length” and “sepal width” form Gaussian curve. While “petal length” and “petal width" show clear classification of one of the three features.
* The pair-plot plotted for data using sns pairplot library function shows a clear separation between *Iris setosa* and the other two species (*versicolor* and *virginica*)
* Pair plot shows that there exist strong correlations exist between petal length and petal width, particularly for *versicolor* and *virginica* species.
* Petal length and petal width are the most suitable features for species classification compared to sepal length and width.

**4.2 Decision Tree**

A Decision Tree classifier is created using DecisionTreeClassifier from scikit learn. TheTree is trained using the Gini criterion having maximum depth of 3. The model trained on iris dataset get the accuracy of 93%. The decision tree model shows that “Setosa” is classified accurately but there is a difficulty while classifying the other two species (*versicolor* and *virginica)*

The confusion matrix is plotted against the prediction made by decision tree model show indicates strong performance but some misclassification between *versicolor* and *virginica*.

**4.3 Random Forest**

A Random Forest Classifier using scikit library with 200 trees, max depth of 2, and min samples split of 2 is trained and it give accuracy of 97% which indicates that model is underfitted.

Hyperparameter tuning using GridSearchCV gives optimal parameters with which random forest gives accuracy of 93%

**4.3 Conclusion**

* Both models achieved the same accuracy of 93%, indicating that the dataset is simple and well-separated.
* The analysis shows that petal length and petal width are the most important features for classification.
* While Setosa is easily distinguishable, some overlap between Versicolor and Virginica suggests that the model may overfit if not properly controlled through hyperparameter tuning.