Machine Learning Lab - Exercise Answers

1. Read top five values

```
df.head()
```

2. Print dataframe info and data types of each column

```
df.info()
```

3. Print number of rows and columns

```
df.shape
```

4. Drop duplicate rows if any

```
df = df.drop_duplicates()
```

5. Print number of rows and columns after dropping duplicates

```
df.shape
```

6. Print summary statistics for numerical variables

```
df.describe()
```

7. Print number of missing values in each column

```
df.isnull().sum()
```

8. Drop the column with most missing values

```
df.drop(columns=[df.isnull().sum().idxmax()], inplace=True)
```

9. Drop the rows with categorical missing values

```
df.dropna(subset=df.select_dtypes(include=['object']).columns, inplace=True)
```

10. Impute (fill) missing numerical values

```
df.fillna(df.mean(), inplace=True)
```

11. Sort the data w.r.t price and find the details of the most and least expensive cars

```
df_sorted = df.sort_values(by="Price in thousands")
df_sorted.iloc[0]  # Least expensive car
df_sorted.iloc[-1] # Most expensive car
```

12. Function to find min and max values of any column

```
def find_min_max(df, column_name):
    return df[column_name].min(), df[column_name].max()
```

13. Min and Max values for Horsepower, Length, and Fuel Efficiency

```
find_min_max(df, "Horsepower")
find_min_max(df, "Length")
find_min_max(df, "Fuel efficiency")
```

14. Plot histogram of continuous numerical variables

```
df[["Price in thousands", "Sales in thousands", "Horsepower", "Fuel efficiency"]].hist()
```

15. Probability density distribution of Length

```
sns.kdeplot(df["Length"], fill=True)
```

16. Count by category - group by manufacturer

```
df.groupby("Manufacturer").size()
```

17. Select all numerical variables

```
df.select_dtypes(include=['number']).columns
```

18. Correlation coefficient between Price and Sales

```
df["Price in thousands"].corr(df["Sales in thousands"])
```

19. Scatterplot of Price vs Sales

```
sns.scatterplot(x=df["Price in thousands"], y=df["Sales in thousands"])
```

20. Pair plot of numerical variables

```
sns.pairplot(df)
```

21. Boxplot of Sales of different manufacturers

```
sns.boxplot(x="Manufacturer", y="Sales in thousands", data=df)
```

22. Boxplot of other numerical variables w.r.t Manufacturer

```
for col in df.select_dtypes(include=['number']).columns:
    sns.boxplot(x="Manufacturer", y=col, data=df)
```

23. Divide the data into input and output

```
X = df.drop(columns=["Sales in thousands"])
y = df["Sales in thousands"]
```

24. Encode categorical variables using LabelEncoder

```
le = LabelEncoder()
for col in X.select_dtypes(include=['object']).columns:
    X[col] = le.fit_transform(X[col])
```

25. Encode categorical variable 'Vehicle type' using One-Hot Encoding

```
X = pd.get_dummies(X, columns=['Vehicle type'], drop_first=True)
```

26. Split the dataset into train and test sets (70% train, 10% test)

```
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.30, random_state=42)
X_test, _, y_test, _ = train_test_split(X_temp, y_temp, test_size=2/3, random_state=42)
```

27. Apply feature scaling on numerical variables

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```