Lab Homework 2 – Encoding the Data

Name: Alfred Joseph Pereira  
Student ID: 02181583  
Course: CIS 550-7101 – Advanced Machine Learning (Summer 2025)

**Step 1: Setup and Import**

We began by importing the necessary libraries for the task. These included:

* pandas: for handling and analyzing the dataset.
* warnings: to suppress future warnings.
* pd.set\_option: to customize the visibility of rows and columns for better readability in the notebook.

A screenshot of a computer program

AI-generated content may be incorrect.

**Code block showing library imports and display settings**

**Step 2: Load and Clean Data**

The dataset was loaded from the UCI Machine Learning Repository using the provided link to imports-85.data.  
We assigned appropriate column names to match the dataset structure and used na\_values="?" to handle missing values.

A screenshot of a computer program

AI-generated content may be incorrect.

**Code that loads and cleans the data**

**Step 3: Select 4 Columns for Encoding**

We selected the following 4 categorical columns for encoding:

* aspiration
* num-of-doors
* drive-wheels
* num-of-cylinders

This subset helped focus on relevant features while simplifying the encoding task. We then used .head() to preview the data.

A screenshot of a computer program

AI-generated content may be incorrect.

**Output of df\_car.head() with selected 4 columns**

**Step 4: Encode Ordinal Features**

Ordinal columns have an inherent order. We manually mapped values as follows:

* num-of-doors: "two" → 2, "four" → 4
* num-of-cylinders: "two" → 2, "three" → 3, ..., "twelve" → 12

These were encoded into new columns called doors and cylinders.

A screenshot of a computer program

AI-generated content may be incorrect.

**A screenshot of a computer

AI-generated content may be incorrect.**

**DataFrame showing doors and cylinders columns after mapping**

**Step 5: Encode Non-Ordinal Features**

For non-ordinal features like aspiration and drive-wheels, we used one-hot encoding (pd.get\_dummies()).

This created new binary columns like:

* aspiration\_turbo
* drive-wheels\_fwd, drive-wheels\_rwd, drive-wheels\_4wd

A screenshot of a computer program

AI-generated content may be incorrect.

**DataFrame output showing encoded columns**

**Step 6: Challenge Task – Add & Encode 2 More Columns**

To meet the challenge task, we added:

* fuel-type
* engine-location

These were encoded using one-hot encoding as well, resulting in:

* fuel-type\_gas
* engine-location\_rear

A screenshot of a computer program

AI-generated content may be incorrect.

**DataFrame showing fuel-type\_gas and engine-location\_rear columns**

**Findings**

* We successfully handled a real-world dataset with mixed categorical features.
* Ordinal features were converted with custom mappings, ensuring their semantic order was retained.
* Nominal features were handled using one-hot encoding, avoiding any artificial ordering.
* We used .head() and .columns to verify each transformation step.
* Two new features were added and encoded in Step 6, proving our understanding of extensible preprocessing.
* Final dataset is fully numeric and suitable for use in ML algorithms.
* All missing values were handled gracefully by using na\_values='?'.

This task helped solidify understanding of encoding, which is a critical preprocessing step in any data science pipeline.

**Conclusion**

This lab provided comprehensive hands-on experience in preparing categorical data for machine learning workflows. I began by loading the automobile dataset from the UCI repository, assigning appropriate headers, and handling missing values. By narrowing the focus to four key categorical columns, I practiced two critical techniques: ordinal encoding for features with natural order and one-hot encoding for nominal features.

Transforming num-of-doors and num-of-cylinders gave insight into how domain knowledge affects data encoding choices. One-hot encoding for aspiration and drive-wheels avoided introducing false ordinal assumptions and showed how to convert text features into machine-readable formats.

The challenge task—adding and encoding fuel-type and engine-location—helped me apply the learned concepts to new attributes, reinforcing flexibility in preprocessing. At every stage, I verified the changes using .head() and column listings.

Overall, I gained a deeper understanding of the impact of encoding on machine learning model performance. I also became more proficient in using pandas for data manipulation, particularly with mapping dictionaries, handling null values, and generating dummy variables. This lab solidified my confidence in applying encoding techniques and in designing reusable, scalable preprocessing pipelines.

**References**

* UCI Machine Learning Repository – [Automobile Dataset](https://archive.ics.uci.edu/ml/datasets/automobile)
* Pandas Documentation – <https://pandas.pydata.org/docs/>
* LabManual\_3\_Encoding the Data (provided by course instructor)