# Flight Cancellation

**4. Handling Airport Column Values**

To handle the issue of inconsistent airport codes in the ORIGIN\_AIRPORT and DESTINATION\_AIRPORT columns, the dataset needs to be processed to ensure a uniform data type. The chosen approach is to convert all airport codes to strings. This conversion is necessary to avoid type mismatches when performing operations such as merging datasets or encoding categorical variables.

By converting all airport codes to strings, the dataset will have consistent data types, regardless of whether the codes were originally represented as numbers or characters. This uniformity eliminates any potential conflicts that may arise during data manipulation and analysis.

To implement this approach, the airport code columns can be converted to strings using appropriate functions or methods provided by the programming language or data analysis tool being used. For example, in Python, the astype() function can be used to convert the columns to strings:

df\_flights['ORIGIN\_AIRPORT'] = df\_flights['ORIGIN\_AIRPORT'].astype(str)

df\_flights['DESTINATION\_AIRPORT'] = df\_flights['DESTINATION\_AIRPORT'].astype(str)

df\_airports['IATA\_CODE'] = df\_airports['IATA\_CODE'].astype(str)

**5. Converting Characters to Numbers or Vice Versa**

Machine learning models typically require numerical inputs to perform computations effectively. Categorical data, such as airport names or airline codes, needs to be converted into a numerical format before it can be used by these models.

The process of converting categorical variables into numerical values is known as encoding. One common encoding technique is to use the Label Encoder class from the scikit-learn (sklearn) library's preprocessing module.

In the provided notebook, the Label Encoder is used to convert the categorical variables in the dataset into numerical values. The key steps are as follows:

1. Identify the categorical variables in the dataset.

2. Create an instance of the Label Encoder class.

3. Fit the Label Encoder to the categorical variables, which assigns a unique integer value to each unique category.

4. Transform the categorical variables using the fitted Label Encoder, replacing the original categorical values with the assigned numerical values.

The end result is a dataset where the categorical variables are represented numerically, allowing the machine learning models to perform computations and generate predictions effectively.

This encoding step is crucial in preparing the data for model training and evaluation. By converting the categorical variables into a numerical format, you ensure that the machine learning algorithms can properly understand and process the input data, leading to more accurate and reliable model performance.

**6. Handling Large Number of Airports**

When faced with a multitude of distinct categorical values, such as airport names, managing high-dimensional data becomes crucial to maintain model efficiency. Employing LabelEncoder helps by translating these names into numerical labels, effectively reducing dimensionality without compromising individual uniqueness.

**7. Predicting Flight Cancellation**

The notebook details a procedure for forecasting flight cancellations, encompassing several key steps:

1. Data Preprocessing: Transforming categorical variables into numerical representations, scaling features, and organizing the data for analysis.

2. Model Training: Employing machine learning techniques to train a model using the preprocessed data.

3. Prediction: Applying the trained model to predict flight cancellations with new data.

4. Deployment of Prediction Function: Implementing a function to execute predictions based on the trained model.

**8. Handling New Airports**

This code snippet defines a function `encode\_new\_category` to handle encoding new categorical values, and another function `predict\_flight\_cancellation` to predict flight cancellations based on a trained model. Here's an explanation of the code:

1. \*\*`encode\_new\_category` function\*\*:

- This function takes a categorical value (`value`) and an encoder (`encoder`) as input.

- It checks if the value is present in the encoder's classes. If not, it returns -1.

- If the value is present, it transforms the value using the encoder and returns the encoded numerical value.

2. \*\*`predict\_flight\_cancellation` function\*\*:

- This function takes the trained model (`model`), scaler (`scaler`), label encoder (`label\_encoder`), and new data (`new\_data`) as input.

- It creates a copy of the new data (`new\_data\_encoded`).

- It encodes categorical columns ('AIRLINE', 'ORIGIN\_AIRPORT\_NAME', 'DESTINATION\_AIRPORT\_NAME') in the new data using the `encode\_new\_category` function.

- Any missing values resulting from unknown categories are filled with -1.

- The features are scaled using the provided scaler (`scaler`).

- The trained model is then used to predict flight cancellations based on the scaled features.

- The function returns the predictions.

3. \*\*Example Usage\*\*:

- An example of new data is created as a Data Frame (`new\_data`).

- The `predict\_flight\_cancellation` function is called with the trained model, scaler, label encoder, and new data as arguments.

- The predictions are printed.

This code essentially provides a way to predict flight cancellations for new data using a trained model after encoding categorical variables and scaling features appropriately.