ONE HOT ENCODING

What is Categorical Encoding?

convert categorical columns to numerical columns.  This process is called **categorical encoding**.

Categorical encoding is a process of converting categories to numbers.

## Different Approaches to Categorical Encoding

So, how should we handle categorical variables? As it turns out, there are multiple ways of handling Categorical variables. In this article, I will discuss the two most widely used techniques:

* Label Encoding
* One-Hot Encoding

### **Label Encoding**

**Label Encoding** is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

|  |
| --- |
| from sklearn import preprocessing |
|  | # label\_encoder object knows how to understand word labels. |
|  | label\_encoder = preprocessing.LabelEncoder() |
|  | # Encode labels in column 'Country'. |
|  | data['Country']= label\_encoder.fit\_transform(data[‘Country']) |
|  | print(data.head()) |

### **One-Hot Encoding**

One-Hot Encoding is another popular technique for treating categorical variables. It simply creates additional features based on the number of unique values in the categorical feature. Every unique value in the category will be added as a feature.

One-Hot Encoding is the process of creating dummy variables.

|  |
| --- |
| # importing one hot encoder |
|  | from sklearn from sklearn.preprocessing import OneHotEncoder |
|  | # creating one hot encoder object |
|  | onehotencoder = OneHotEncoder() |
|  | #reshape the 1-D country array to 2-D as fit\_transform expects 2-D and finally fit the object |
|  | X = onehotencoder.fit\_transform(data.Country.values.reshape(-1,1)).toarray() |
|  | #To add this back into the original dataframe |
|  | dfOneHot = pd.DataFrame(X, columns = ["Country\_"+str(int(i)) for i in range(data.shape[1])]) |
|  | df = pd.concat([data, dfOneHot], axis=1) |
|  | #droping the country column |
|  | df= df.drop(['Country'], axis=1) |
|  | #printing to verify |
|  | print(df.head()) |