Kmit

SONET

Data Science
(Level-I)

Ajeet K. Jain, M. Narsimlu

(ML TEAM)- SONET, KMIT, Hyderabad

Session – 28

This session deals with

Normalization

Sklearn

Categorical Data

Encoding Categorical Data

Data Science Project Life cycle

Introduction to Case Study

Normalization

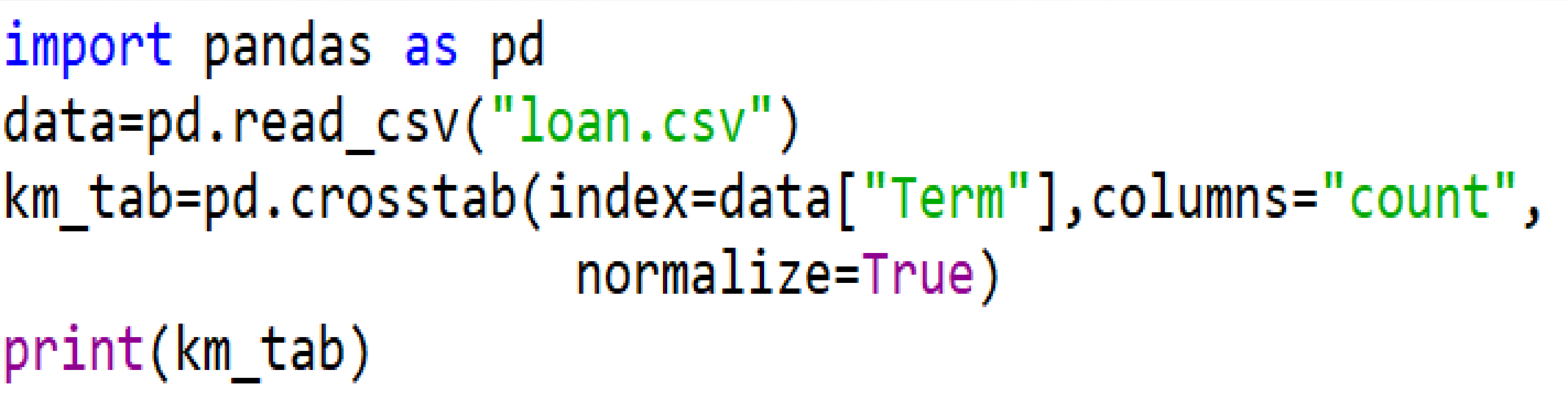
It is the process of reorganizing data in a dataset so that it meets two basic requirements:
(1)There is no redundancy of data (all data is stored in only one place), and
(2)data dependencies are logical (all related data items are stored together)

Normalizing in scikit-learn refers to rescaling each observation (row) to have a length of 1 (called a unit norm in linear algebra).
This preprocessing can be useful for sparse datasets (lots of zeros) with attributes of varying scales when using algorithms that weight input values such as neural networks and algorithms that use distance measures such as K-Nearest Neighbors.

Normalization

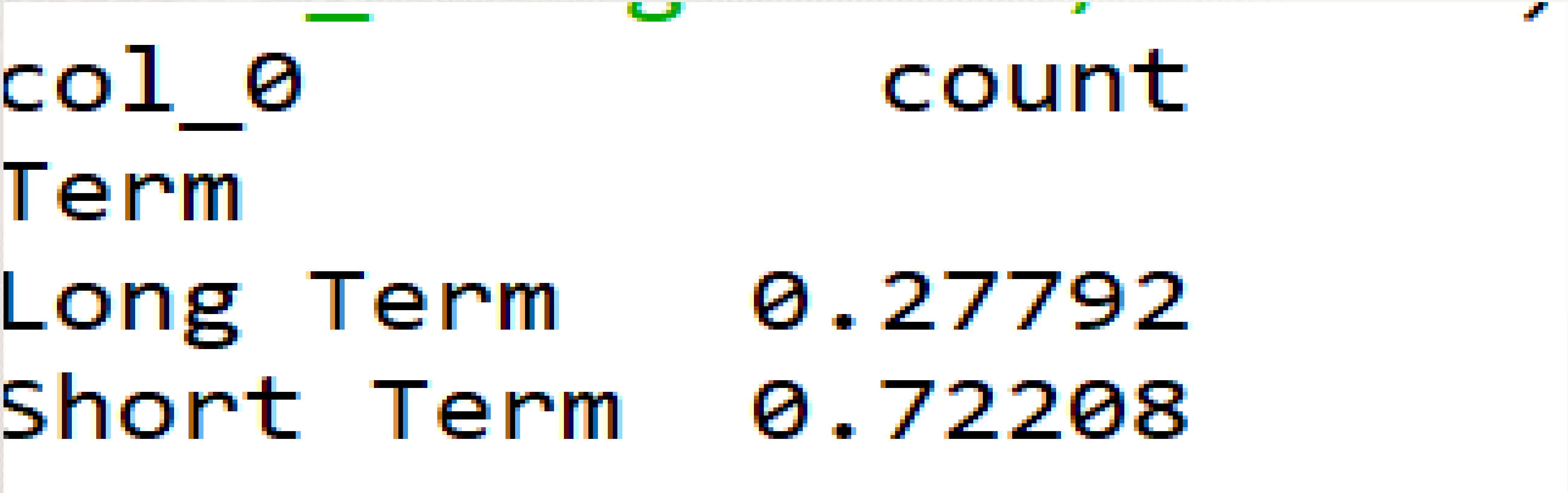
Normalizing in scikit-learn refers to rescaling each observation (row) to have a length of 0 to1

Read the Loan dataset and normalize the “Term” categorical variable



```
import pandas as pd
data=pd.read_csv("loan.csv")
km_tab=pd.crosstab(index=data["Term"],columns="count",
                          normalize=True)
print(km_tab)
```

Output



```
col_0              count
Term
Long Term        0.27792
Short Term       0.72208
```

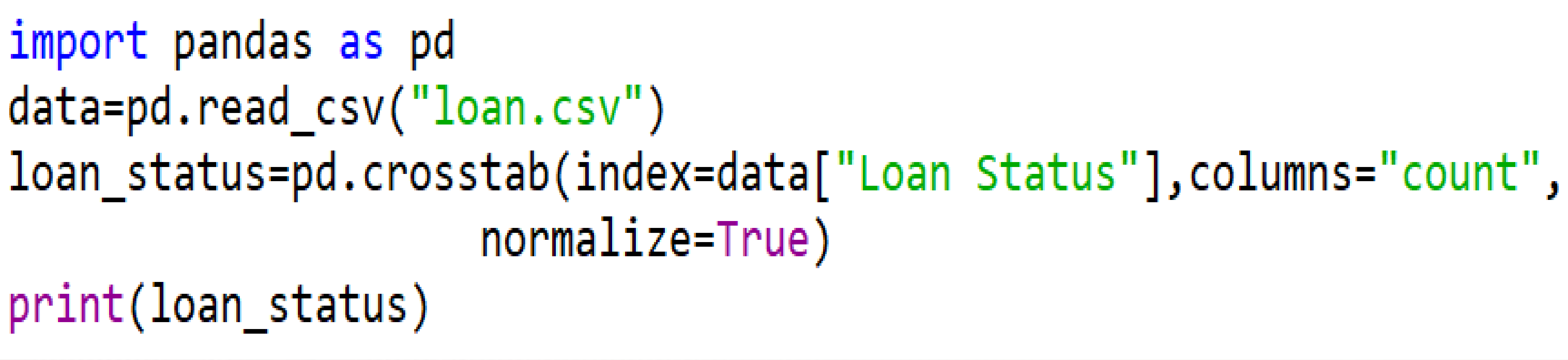
Exercise-1

Read the Loan dataset and perform the following tasks:

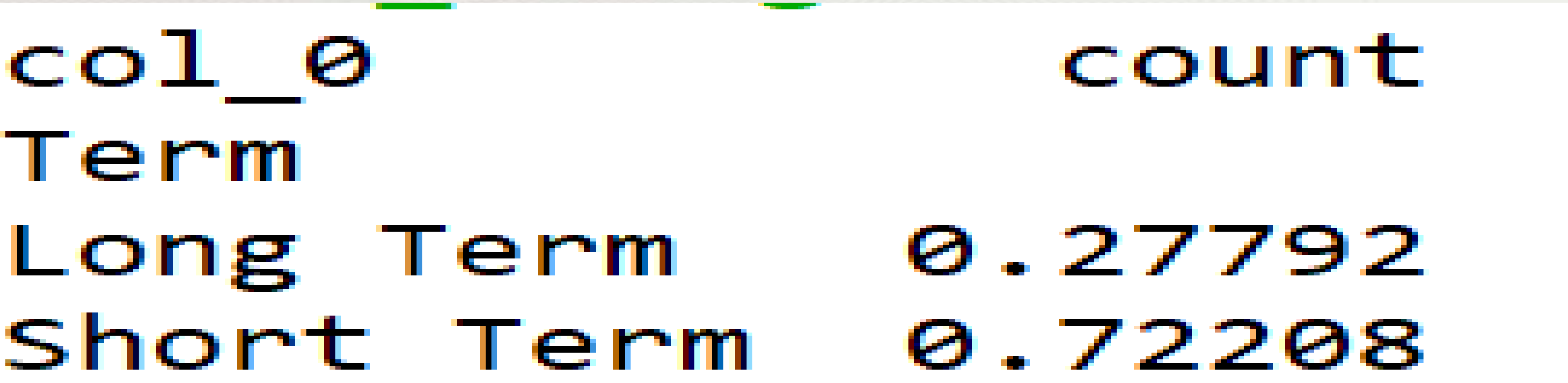
1.Create a one way table on “Loan Status” categorical variable and normalize it

2. Create a one way table on “Purpose” categorical variable and normalize it

1.Solution

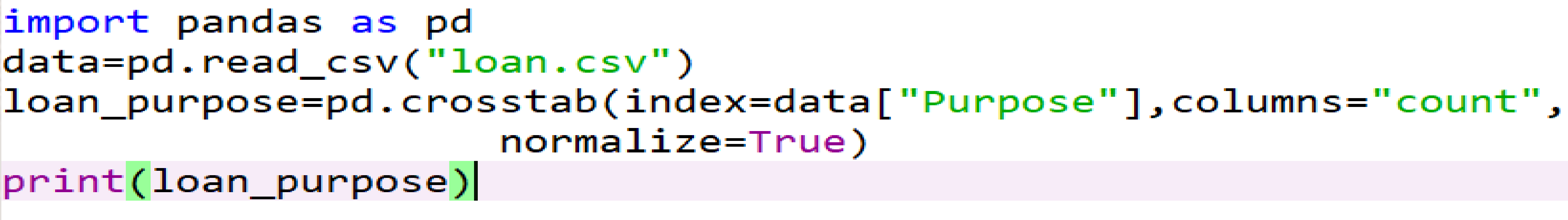


```
import pandas as pd
data=pd.read_csv("loan.csv")
loan_status=pd.crosstab(index=data["Loan Status"],columns="count",
                    normalize=True)
print(loan_status)
```



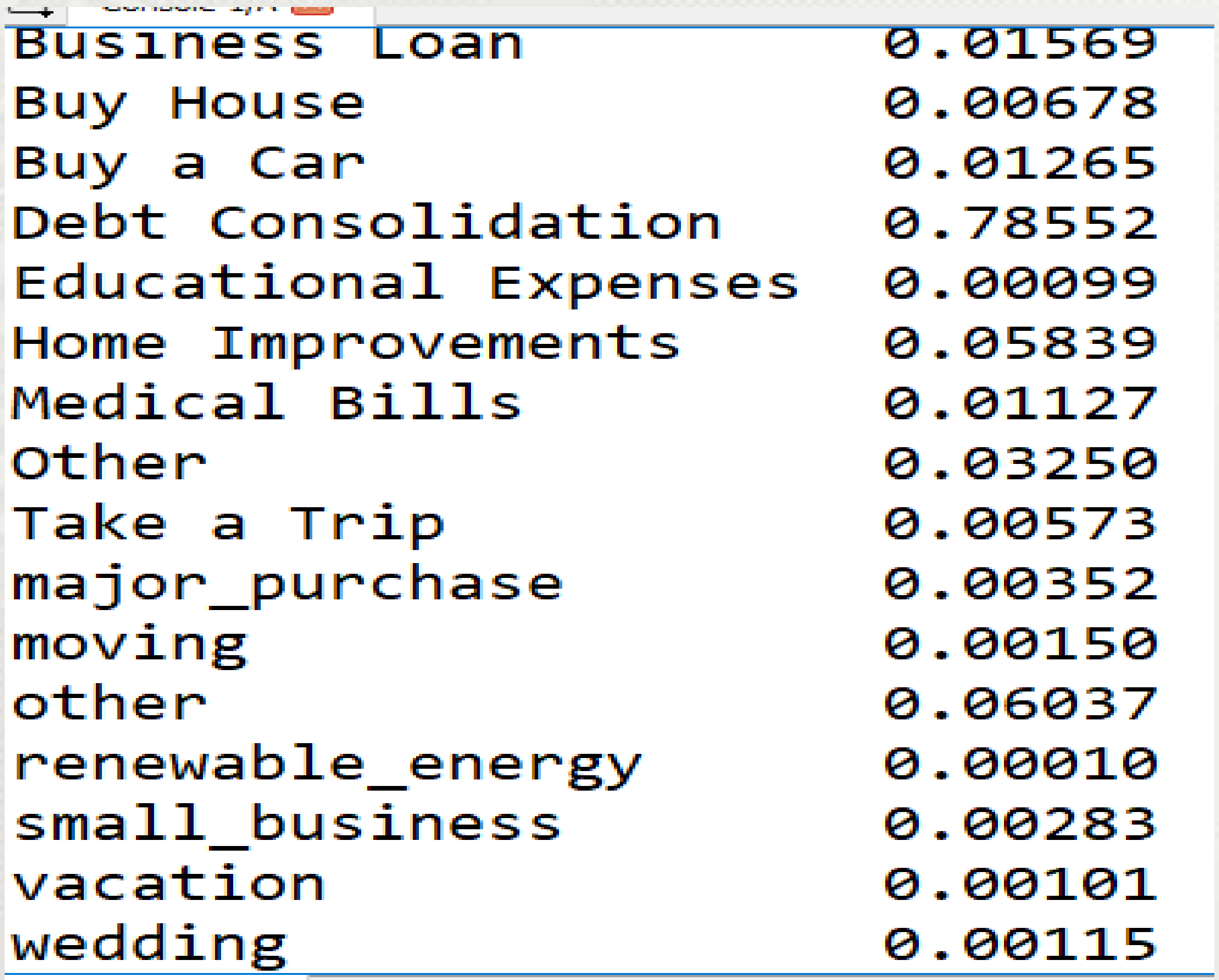
```
col_0           count
Term
Long Term     0.27792
Short Term    0.72208
```

2.Solution



```
import pandas as pd
data=pd.read_csv("loan.csv")
loan_purpose=pd.crosstab(index=data["Purpose"],columns="count",
                    normalize=True)
print(loan_purpose)
```

Output



```
Business Loan                0.01569
Buy House                    0.00678
Buy a Car                    0.01265
Debt Consolidation           0.78552
Educational Expenses         0.00099
Home Improvements            0.05839
Medical Bills                0.01127
Other                        0.03250
Take a Trip                  0.00573
major_purchase               0.00352
moving                       0.00150
other                        0.06037
renewable_energy             0.00010
small_business               0.00283
vacation                     0.00101
wedding                      0.00115
```

Sklearn-Introduction

Machine Learning library

Designed to inter-operate with Numpy and SciPy

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

The library is built upon the SciPy (Scientific Python)

Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

What are the features

Clustering: for grouping unlabeled data such as KMeans.

Cross Validation: for estimating the performance of supervised models on unsee

Datasets: for test datasets and for generating datasets with specific properties for investigating model behavior.

Dimensionality Reduction: for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis

What are the features

Ensemble methods: for combining the predictions of multiple supervised models.

Feature extraction: for defining attributes in image and text data.

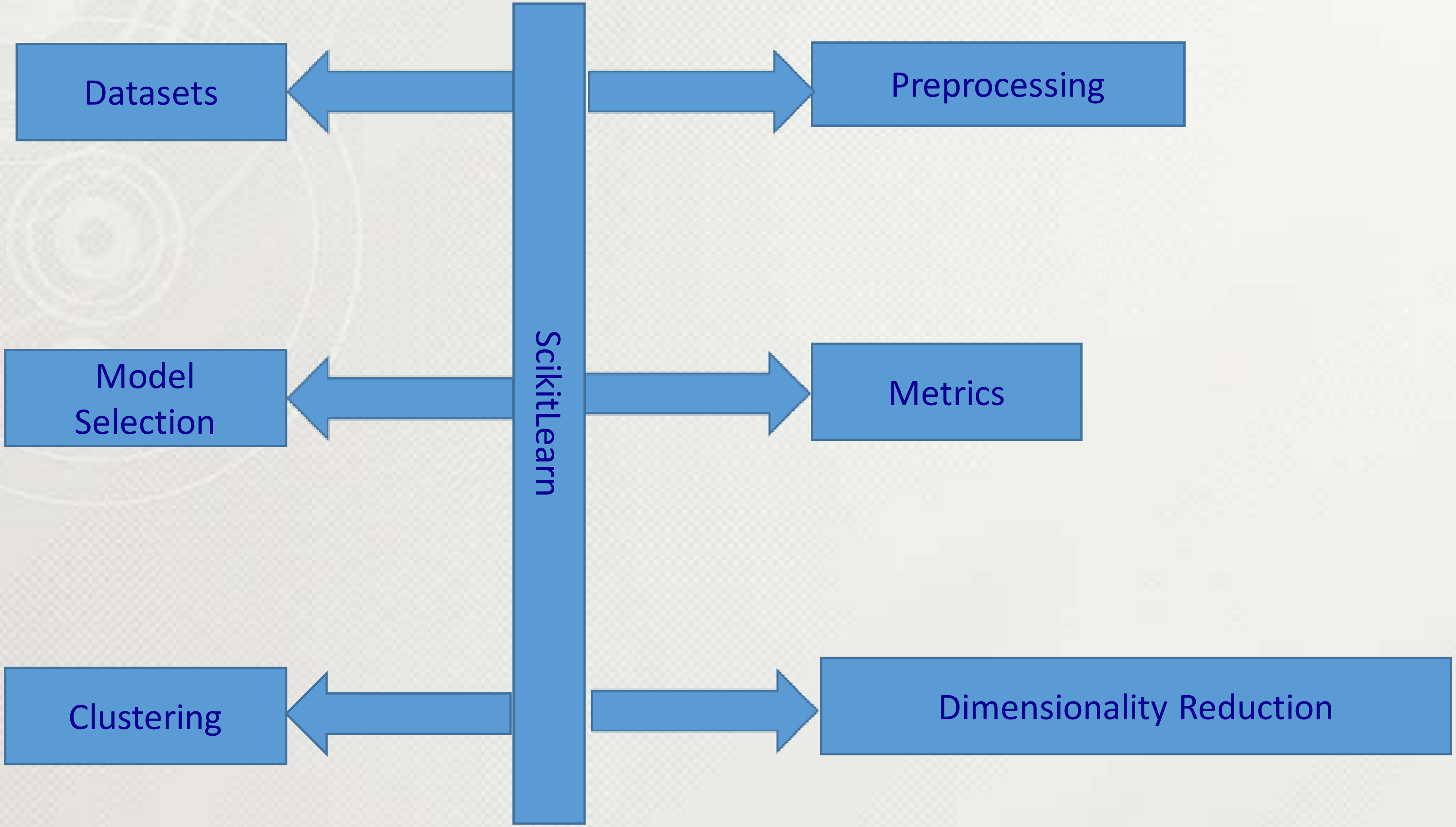
Feature selection: for identifying meaningful attributes from which to create supervised models.

ter Tuning: for getting the most out of supervised models.

Manifold Learning: For summarizing and depicting complex multi-dimensional data.

Supervised Models: a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

ScikitLearn



ScikitLearn

Datasets

Model Selection

Clustering

Preprocessing

Metrics

Dimensionality Reduction

ScikitLearn

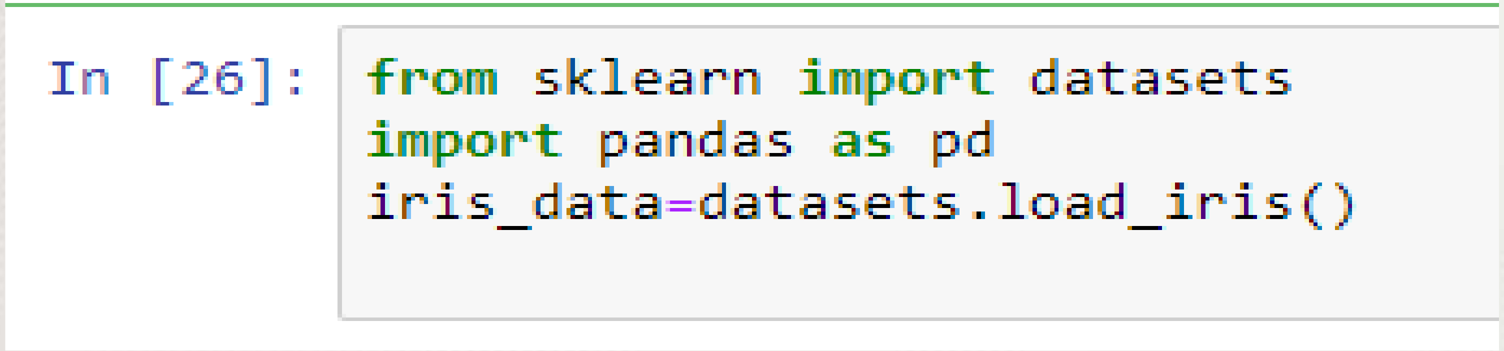
Pre-Processing

Function	Description
sklearn.preprocessing.StandardScaler	Standardize features by removing the mean and scaling to unit variance
sklearn.preprocessing.Imputer	Imputation transformer for completing missing values
sklearn.preprocessing.LabelBinarizer	Binarize labels in a one-vs-all fashion
sklearn.preprocessing.OneHotEncoder	Encode categorical integer features using a one-hot a.k.a one-of-K scheme
sklearn.preprocessing.PolynomialFeatures	Generate polynomial and interaction features

ScikitLearn Libraries

Loading Datasets:

scikit-learn comes with a few standard datasets, for instance the **iris and digits datasets** for classification and the **Boston house prices** dataset for regression.



```
In [26]: from sklearn import datasets
         import pandas as pd
         iris_data=datasets.load_iris()
```

ScikitLearn Libraries

Available Datasets:

scikit-learn comes with a few small standard datasets that do not require to download any file from some external website.

load_boston([return_X_y]) Load and return the boston house-prices dataset (regression)
load_iris([return_X_y]) Load and return the iris dataset (classification).
load_diabetes([return_X_y]) Load and return the diabetes dataset (regression).
load_digits([n_class, return_X_y]) Load and return the digits dataset (classification).
load_linnerud([return_X_y]) Load and return the linnerud dataset (multivariate regression)
load_wine([return_X_y]) Load and return the wine dataset (classification).
load_breast_cancer([return_X_y]) Load and return the breast cancer wisconsin dataset (classification).

ScikitLearn Libraries

Regression	
Function	**Description**
sklearn.tree.DecisionTreeRegressor	A decision tree regressor
sklearn.svm.SVR	Epsilon-Support Vector Regression
sklearn.linear_model.LinearRegression	Ordinary least squares Linear Regression
sklearn.linear_model.Lasso	Linear Model trained with L1 prior as regularized (a.k.a the lasso)
sklearn.linear_model.SGDRegressor	Linear model fitted by minimizing a regularized empirical loss with SGD
sklearn.linear_model.ElasticNet	Linear regression with combined L1 and L2 priors as regularizor
sklearn.ensemble.RandomForestRegressor	A random forest regressor
sklearn.ensemble.GradientBoostingRegressor	Gradient Boosting for regression
sklearn.neural_network.MLPRegressor	Multi-layer Perceptron regressor

ScikitLearn Libraries

classification	
Function	**Description**
sklearn.neural_network.MLPClassifier	Multi-layer Perceptron classifier
sklearn.tree.DecisionTreeClassifier	A decision tree classifier
sklearn.svm.SVC	C-Support Vector Classification
sklearn.linear_model.LogisticRegression	Logistic Regression (a.k.a logit, Max Ent) classifier
sklearn.linear_model.SGDClassifier	Linear classifiers (SVM, logistic regression, a.o.) with SGD training
sklearn.naive_bayes.GaussianNB	Gaussain Naïve Bayes
sklearn.neighbors.KNeighborsClassifier	Classifier implementing the k-nearest neighbors vote
sklearn.ensemble.RandomForestClassifier	A random forest classifier
sklearn.ensemble.GradientBoostingClassifier	Gradient Boosting for classification

ScikitLearn Libraries

Clustering	
Function	**Description**
sklearn.cluster.Kmeans	K-Means clustering
sklearn.cluster.DBSCAN	perform DBSCAN clustering from vector array or distance matrix
sklearn.cluster.AgglomerativeClustering	Agglomerative clustering
sklearn.cluster.SpectralBiclustering	Spectral bi-clustering

ScikitLearn Libraries

Model Selection	
Function	**Description**
sklearn.model_selection.Kfold	K-Folds cross-validator
sklearn.model_selection.StratifiedKFold	Stratified K-Flods cross-validator
sklearn.model_selection.TimeSeriesSplit	Time Series cross-validator
sklearn.model_selection.train_test_split	Split arrays or matrices into random train and test subsets
sklearn.model_selection.GridSearchCV	Exhaustive search over specified parameter value for an estimator
sklearn.model_selection.cross_val_score	Evaluate a score by cross-validation

Kmit SONET

DATA SCIENCE

ScikitLearn Libraries

Dimensionality Reduction	
Function	**Description**
sklearn.decomposition.PCA	Principal component analysis (PCA)
sklearn.decomposition.LatentDirichletAllocation	Latent Dirichlet Allocation with online variational Bayes algorithm
sklearn.decomposition.SparseCoder	Sparse coding
sklearn.decomposition.DictionaryLearning	Dictionary learning

ScikitLearn Libraries

Metric	
Function	**Description**
sklearn.metrics.accuracy_score	Classification Metric: Accuracy classification score
sklearn.metrics.log_loss	Classification Metric: Log loss, a.k.a logistic loss or cross-entropy loss
sklearn.metrics.roc_auc_score	Classification Metric: Compute Receiver operating characteristics ROC
sklearn.metrics.mean_absolute_error	Regression Metric: Mean absolute error regression loss
sklearn.metrics.r2_score	Regression Metric: R^2 (coefficient of determination) regression score
sklearn.metrics.label_ranking_loss	Ranking Metric: Compute Ranking loss measure
sklearn.metrics.mutual_info_score	Clustering Metric: Mutual Information between two clustering.

Categorical data

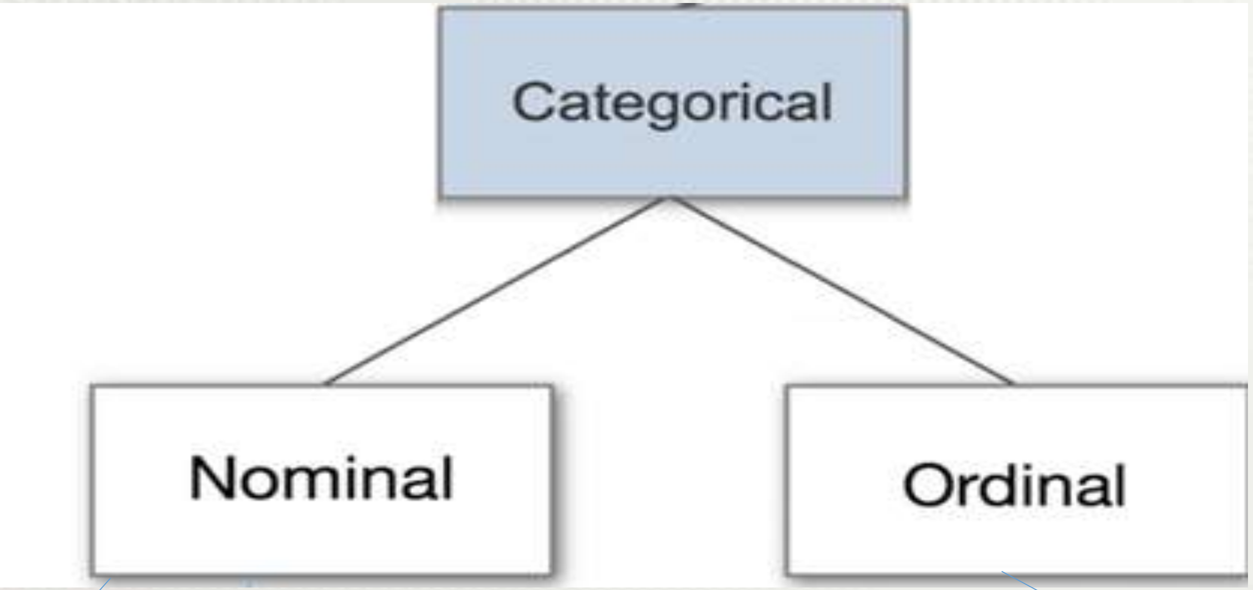
Categorical Attributes –

- When the number unique values in a categorical column are too high, check the value counts of each of those values. Replace rarely occurring values together into a single value like ‘Other’ before encoding.
- When number of unique values is huge and even the values are equally distributed, try to find some related values and see if the multiple categorical values can be clubbed into single (grouping), thereby reducing the count of categorical values.

Related Attributes –

- If there multiple attributes with same information with different granularity, like city and state, it’s better to keep columns like state and delete city column. Additionally, keeping both columns and assessing feature importance might help in eliminating one column.

Types of Categorical Data



Gender : Male, Female
Car Color: Brown, red,blue,orange,white

Railway Reservation Tickets : 1 class,2 class,3 class
Feedback on machine learning: average, good,very good,excellent
Education : Kindergarden, School, Undergraduate, bachelor, master,doctoral

Types of Data

Categorical data:

- It represents characteristics.
- Therefore it can represent things like a person’s gender, language etc.

1. **Nominal Data**

Nominal values represent discrete units and are used to label variables, that have no quantitative value.

➢ nominal data that has no order.

➢ Used to “name,” or label a series of values.

EX: what’s your favourite movie

- ○ Spider man
- ○ Ant man
- ○ Iron man

Types of Data

2.Ordinal Data

Ordinal values represent discrete and ordered units. It is therefore nearly the same as nominal data, “except that it’s ordering matters”.

➡ Ordinal scales provide good information about the order of choices.

Ex1:

What’s your rating for Avengers Infinity War?

○ *

○ **

○ ***

○ ****

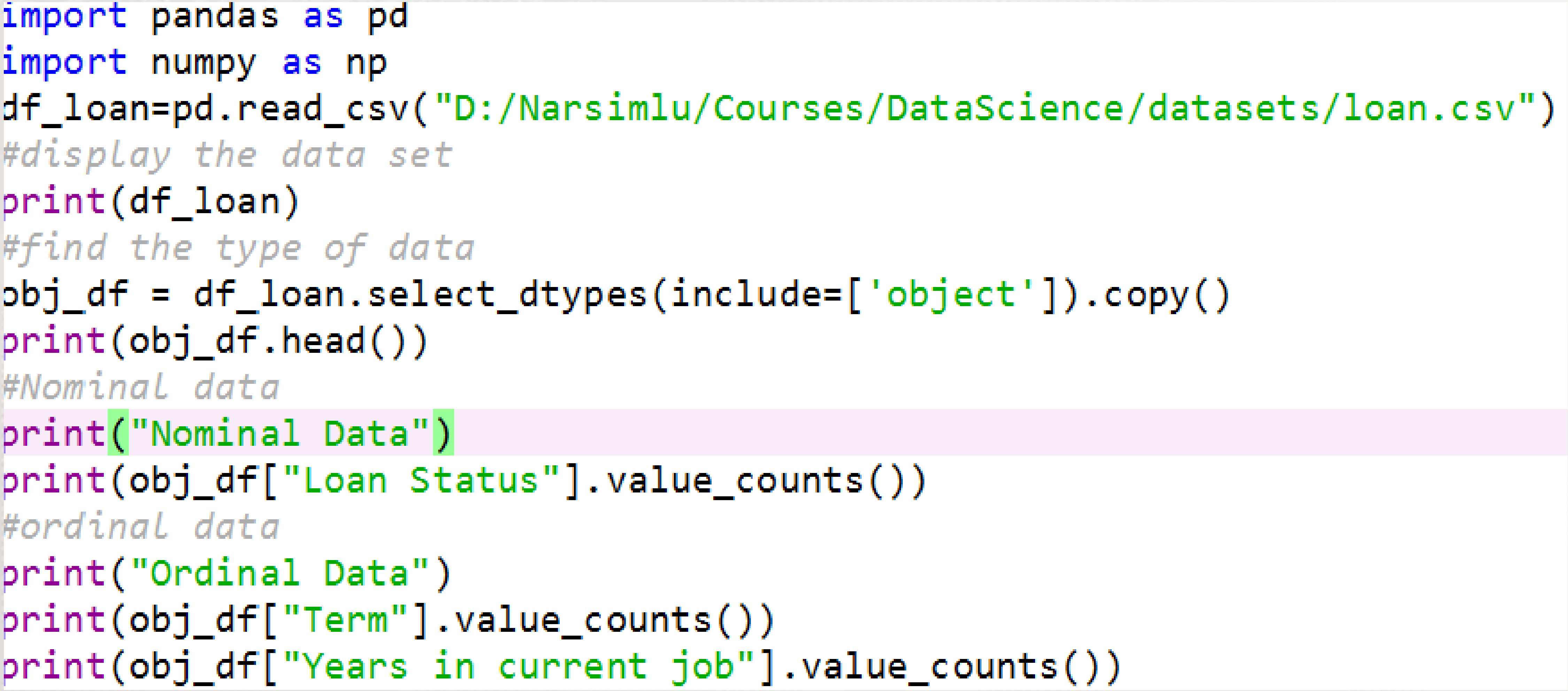
○ *****

- Perform the following steps to identify categorical data

- Load the data
- Describe the data set columns
- Identify the categorical data

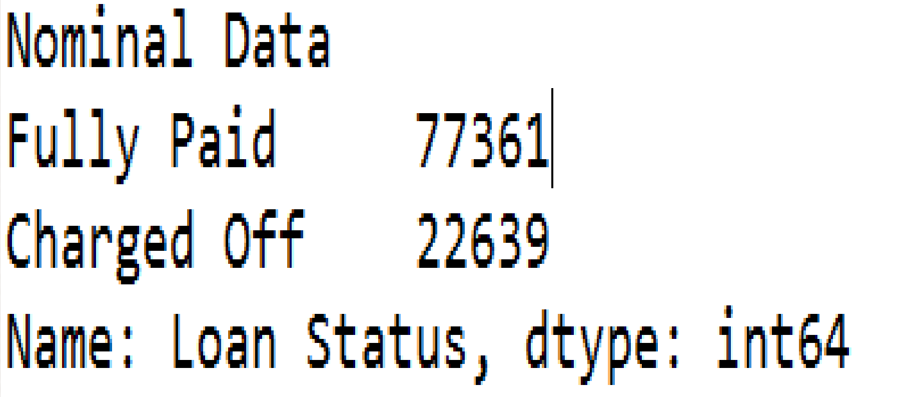
Exercise-1

- Read the loan Data set and identify the categorical data

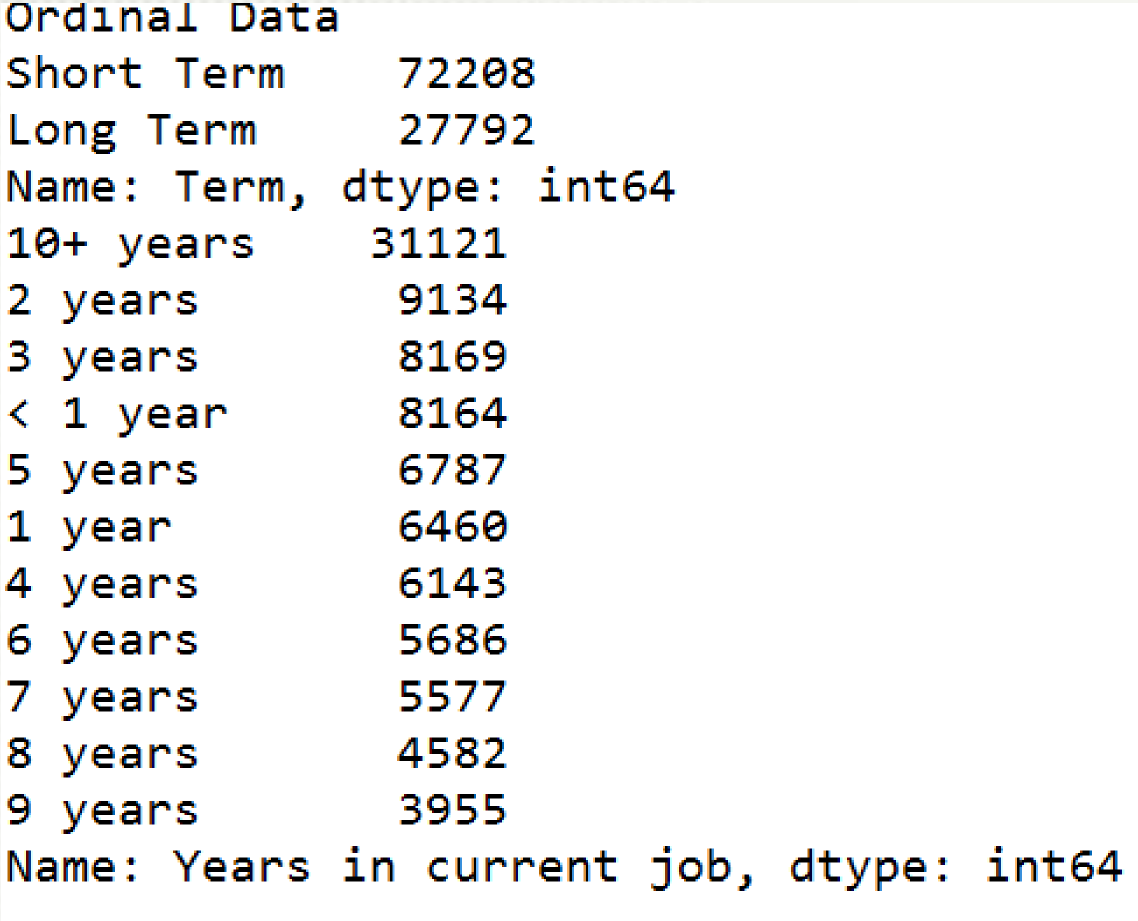


```
import pandas as pd
import numpy as np
df_loan=pd.read_csv("D:/Narsimlu/Courses/DataScience/datasets/loan.csv")
#display the data set
print(df_loan)
#find the type of data
obj_df = df_loan.select_dtypes(include=['object']).copy()
print(obj_df.head())
#Nominal data
print("Nominal Data")
print(obj_df["Loan Status"].value_counts())
#ordinal data
print("Ordinal Data")
print(obj_df["Term"].value_counts())
print(obj_df["Years in current job"].value_counts())
```

Output



```
Nominal Data
Fully Paid      77361
Charged Off     22639
Name: Loan Status, dtype: int64
```



```
Ordinal Data
Short Term     72208
Long Term      27792
Name: Term, dtype: int64
10+ years    31121
2 years       9134
3 years       8169
< 1 year      8164
5 years       6787
1 year        6460
4 years       6143
6 years       5686
7 years       5577
8 years       4582
9 years       3955
Name: Years in current job, dtype: int64
```

Data Encoding

- Many machine learning algorithms cannot operate on label data directly. They require all input variables and output variables to be numeric.

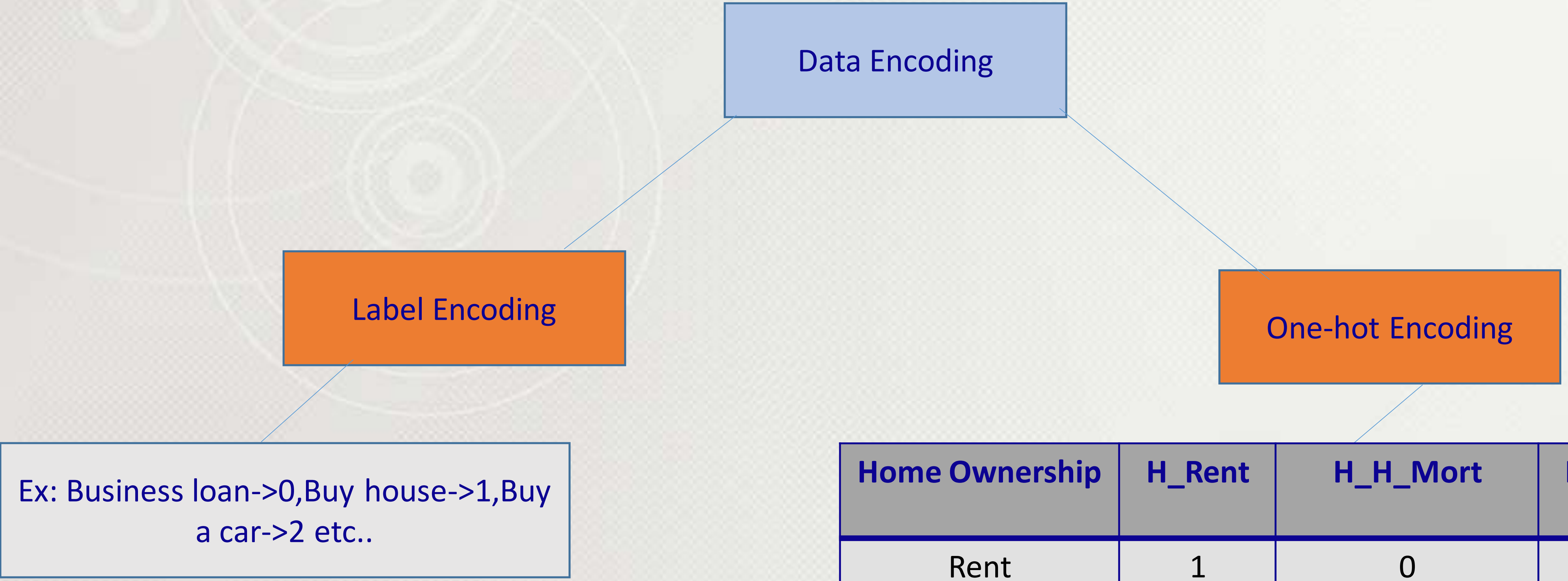
- In general, this is mostly a constraint of the efficient implementation of machine learning algorithms rather than hard limitations on the algorithms themselves.

- This means that categorical data must be converted to a numerical form.

Label Encoding

- Numerical variable will to assign a unique number to each possible outcome of the variable and replace the variables values with its corresponding number.
- Ex:

Purpose	loan_purpose_cat
Business loan	0
Buy house	1
Buy a car	2
Debt Consolidation	3
Educational Expenses	4



Data Encoding

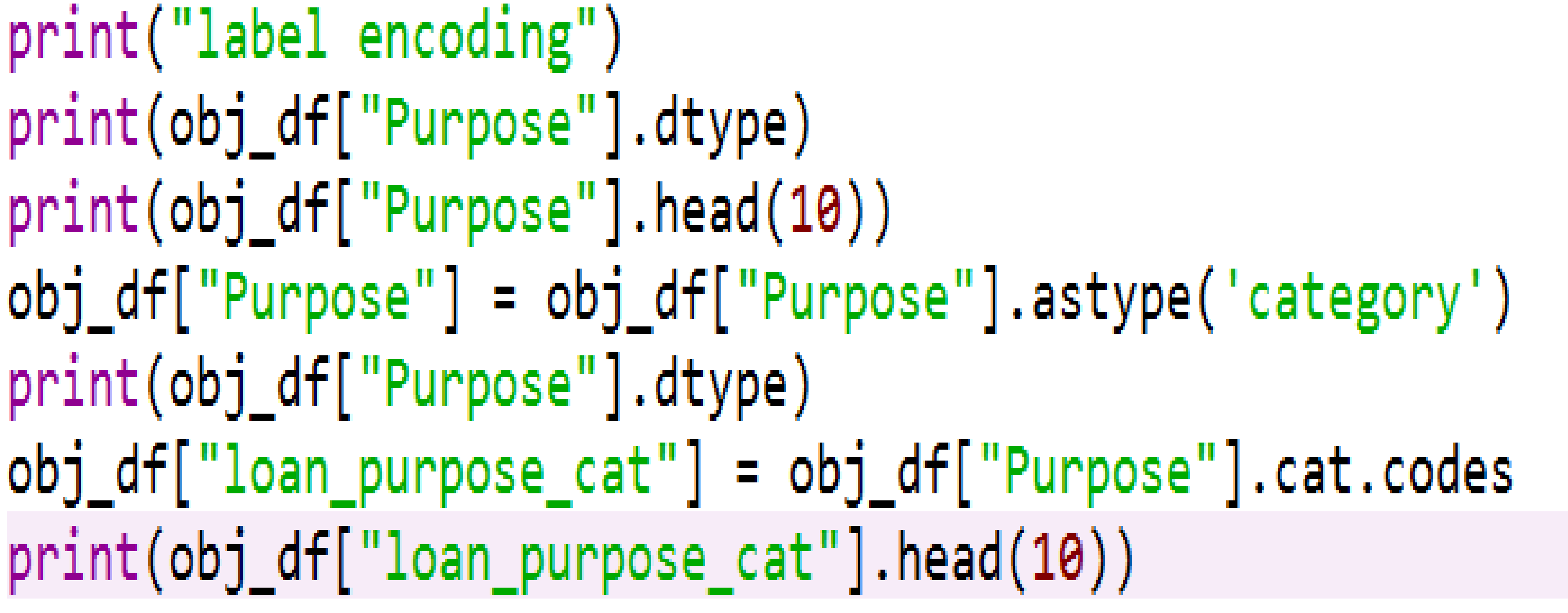
Label Encoding

One-hot Encoding

Ex: Business loan->0,Buy house->1,Buy a car->2 etc..

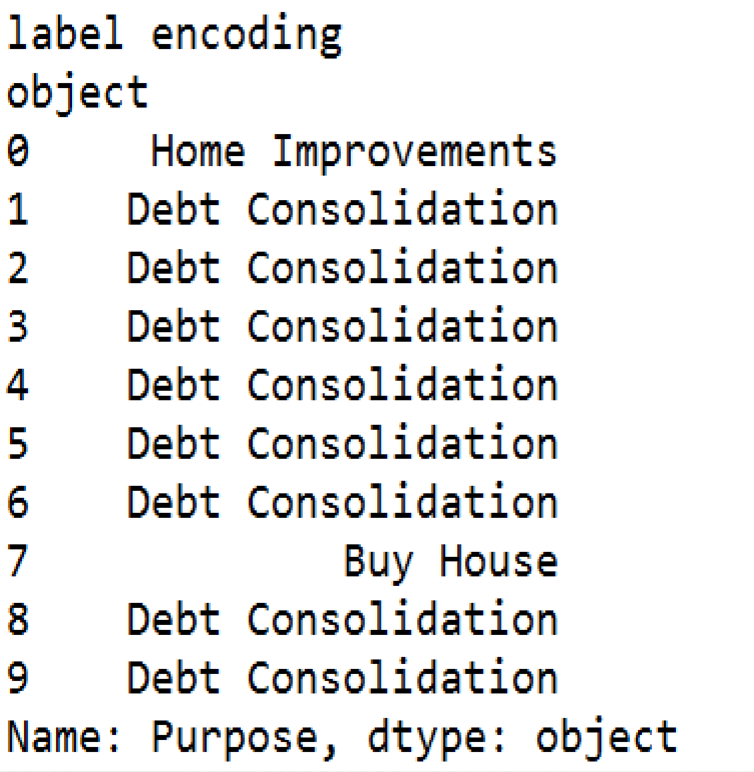
Home Ownership	H_Rent	H_H_Mort	H_Own Home
Rent	1	0	0
Rent	0	1	0
Own Home	1	0	0
Home Mortgage	0	0	1
Home Mortgage	0	0	1

Label Encoding Example

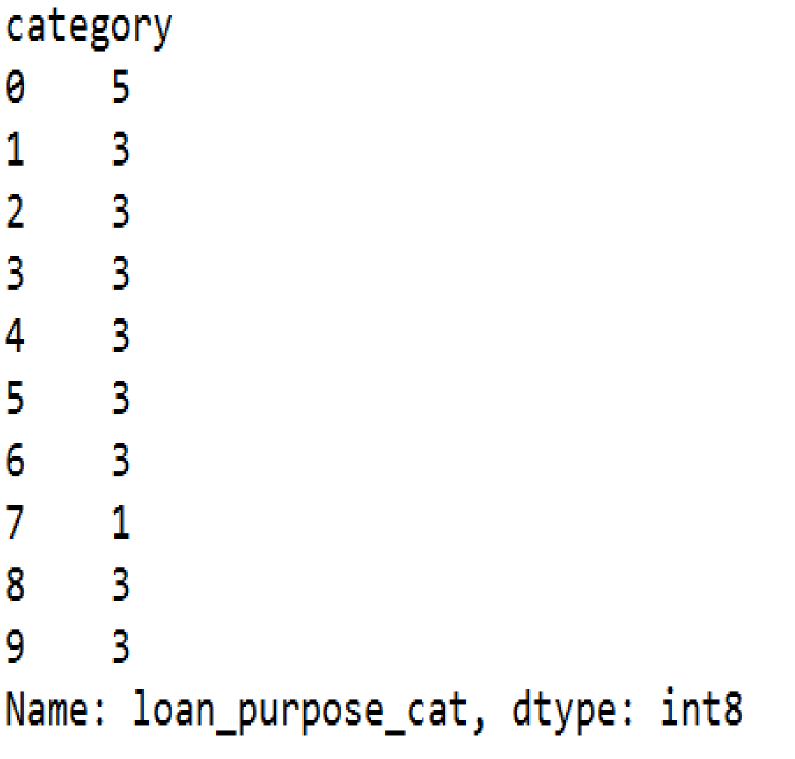


```
print("label encoding")
print(obj_df["Purpose"].dtype)
print(obj_df["Purpose"].head(10))
obj_df["Purpose"] = obj_df["Purpose"].astype('category')
print(obj_df["Purpose"].dtype)
obj_df["loan_purpose_cat"] = obj_df["Purpose"].cat.codes
print(obj_df["loan_purpose_cat"].head(10))
```

Output

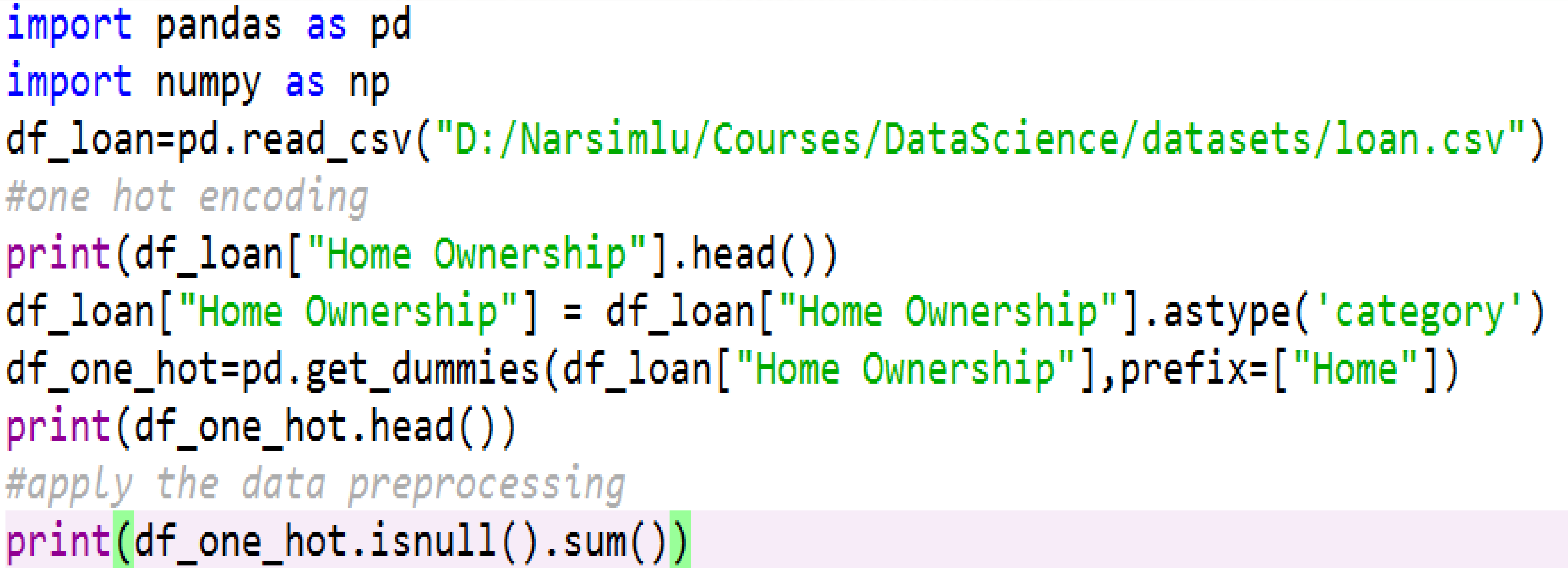


```
label encoding
object
0     Home Improvements
1    Debt Consolidation
2    Debt Consolidation
3    Debt Consolidation
4    Debt Consolidation
5    Debt Consolidation
6    Debt Consolidation
7             Buy House
8    Debt Consolidation
9    Debt Consolidation
Name: Purpose, dtype: object
```



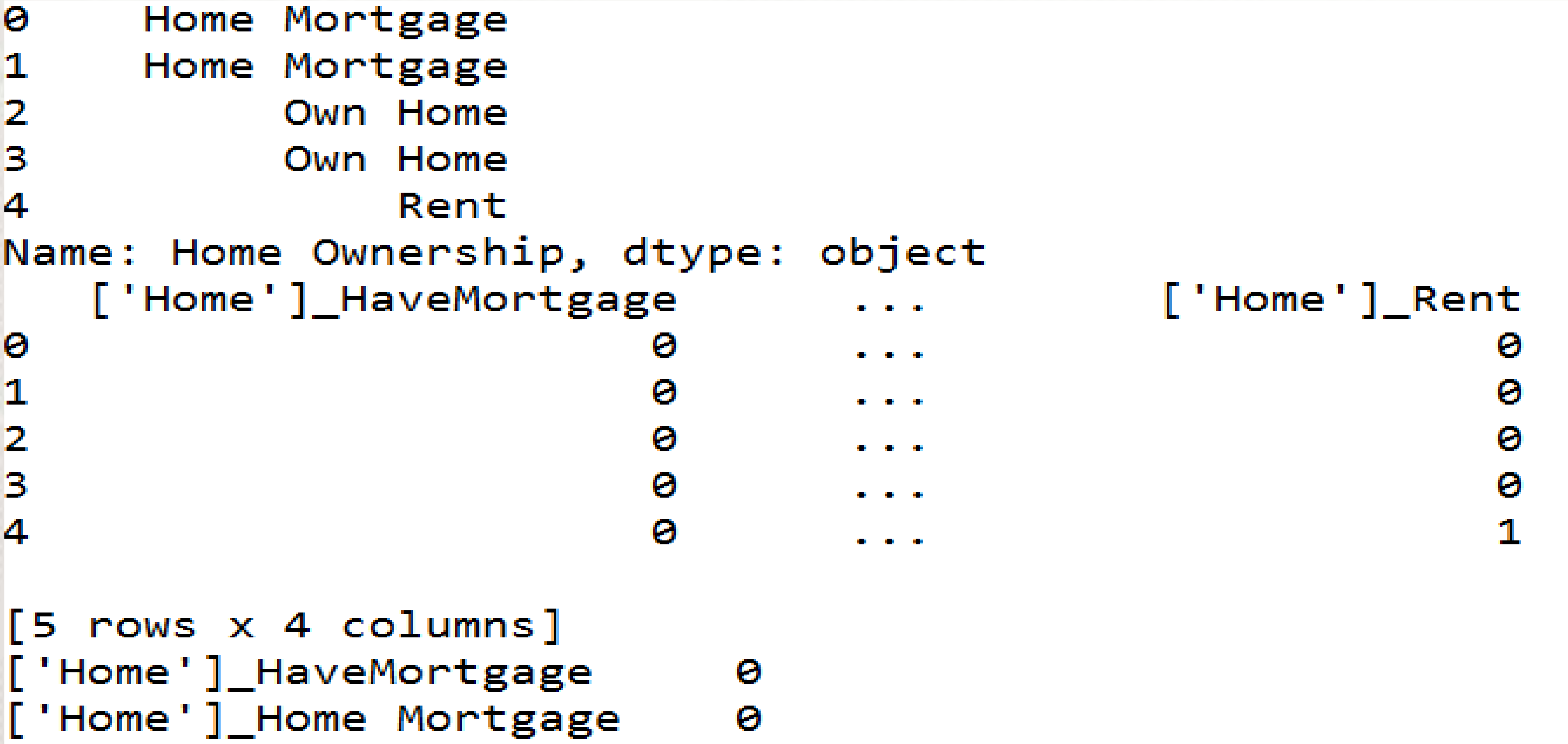
```
category
0    5
1    3
2    3
3    3
4    3
5    3
6    3
7    1
8    3
9    3
Name: loan_purpose_cat, dtype: int8
```

One hot Encoding Example



```python
import pandas as pd
import numpy as np
df_loan=pd.read_csv("D:/Narsimlu/Courses/DataScience/datasets/loan.csv")
#one hot encoding
print(df_loan["Home Ownership"].head())
df_loan["Home Ownership"] = df_loan["Home Ownership"].astype('category')
df_one_hot=pd.get_dummies(df_loan["Home Ownership"],prefix=["Home"])
print(df_one_hot.head())
#apply the data preprocessing
print(df_one_hot.isnull().sum())
```

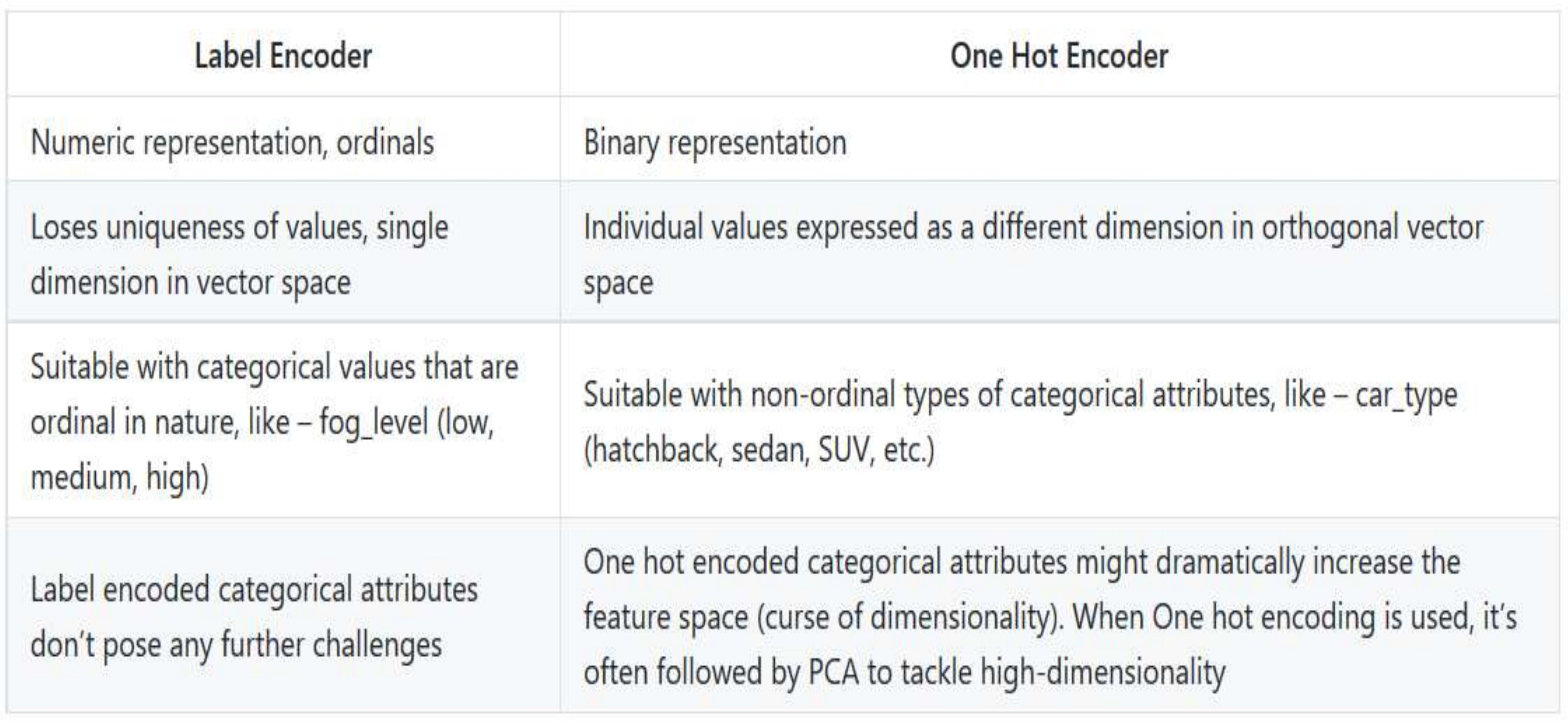
Output



```
0    Home Mortgage
1    Home Mortgage
2         Own Home
3         Own Home
4             Rent
Name: Home Ownership, dtype: object
   ['Home']_HaveMortgage     ...          ['Home']_Rent
0                      0     ...                      0
1                      0     ...                      0
2                      0     ...                      0
3                      0     ...                      0
4                      0     ...                      1

[5 rows x 4 columns]
['Home']_HaveMortgage     0
['Home']_Home Mortgage    0
```

Categorical data



Label Encoder	One Hot Encoder
Numeric representation, ordinals	Binary representation
Loses uniqueness of values, single dimension in vector space	Individual values expressed as a different dimension in orthogonal vector space
Suitable with categorical values that are ordinal in nature, like – fog_level (low, medium, high)	Suitable with non-ordinal types of categorical attributes, like – car_type (hatchback, sedan, SUV, etc.)
Label encoded categorical attributes don't pose any further challenges	One hot encoded categorical attributes might dramatically increase the feature space (curse of dimensionality). When One hot encoding is used, it's often followed by PCA to tackle high-dimensionality

Conclusion

You are aware of

Data Preprocessing

We will proceed with

Case Study

THANK
YOU