

# Preprocessing Data for Machine Learning

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



**Sarah Guido**  
Senior Data Scientist

# What is data preprocessing?

- Beyond cleaning and exploratory data analysis
- Prepping data for modeling
- Modeling in Python requires numerical input <sup>Python</sup>가

# Refresher on Pandas basics

```
import pandas as pd
hiking = pd.read_json("datasets/hiking.json")
print(hiking.head())
```

	Accessible	Difficulty	Length	Limited_Access
0	Y	None	0.8 miles	N
1	N	Easy	1.0 mile	N
2	N	Easy	0.75 miles	N
3	N	Easy	0.5 miles	N
4	N	Easy	0.5 miles	N

# Refresher on Pandas basics

```
print(hiking.columns)
```

```
Index(['Accessible', 'Difficulty',  
      'Length', 'Limited_Access',  
      'Location', 'Name',  
      'Other_Details', 'Park_Name',  
      'Prop_ID', 'lat', 'lon'],  
      dtype='object')
```

```
print(hiking.dtypes)
```

```
Accessible      object  
Difficulty      object  
Length          object  
Limited_Access  object  
Location        object  
Name            object  
Other_Details   object  
Park_Name       object  
Prop_ID         object  
lat             float64  
lon             float64  
dtype: object
```

# Refresher on Pandas basics

```
print(wine.describe())
```

	Type	Alcohol	...	Alcalinity of ash
count	178.000000	178.000000	...	178.000000
mean	1.938202	13.000618	...	19.494944
std	0.775035	0.811827	...	3.339564
min	1.000000	11.030000	...	10.600000
25%	1.000000	12.362500	...	17.200000
50%	2.000000	13.050000	...	19.500000
75%	3.000000	13.677500	...	21.500000
max	3.000000	14.830000	...	30.000000

# Removing missing data

```
print(df)
```

	A	B	C
0	1.0	NaN	2.0
1	4.0	7.0	3.0
2	7.0	NaN	NaN
3	NaN	7.0	NaN
4	5.0	9.0	7.0

```
print(df.dropna())
```

	A	B	C
1	4.0	7.0	3.0
4	5.0	9.0	7.0

# Removing missing data

```
print(df)
```

	A	B	C
0	1.0	NaN	2.0
1	4.0	7.0	3.0
2	7.0	NaN	NaN
3	NaN	7.0	NaN
4	5.0	9.0	7.0

```
print(df.drop([1, 2, 3]))
```

	A	B	C
0	1.0	NaN	2.0
4	5.0	9.0	7.0

# Removing missing data

```
print(df)
```

	A	B	C
0	1.0	NaN	2.0
1	4.0	7.0	3.0
2	7.0	NaN	NaN
3	NaN	7.0	NaN
4	5.0	9.0	7.0

```
print(df.drop("A", axis=1))
```

axis=1

가

	B	C
0	NaN	2.0
1	7.0	3.0
2	NaN	NaN
3	7.0	NaN
4	9.0	7.0



# Removing missing data

```
print(df)
```

	A	B	C
0	1.0	NaN	2.0
1	4.0	7.0	3.0
2	7.0	NaN	NaN
3	NaN	7.0	NaN
4	5.0	9.0	7.0

```
print(df[df["B"] == 7])
```

	A	B	C
1	4.0	7.0	3.0
3	NaN	7.0	NaN

B가 7

dataframe  
B가 7 dataframe

가

# Removing missing data

```
print(df)
```

	A	B	C
0	1.0	NaN	2.0
1	4.0	7.0	3.0
2	7.0	NaN	NaN
3	NaN	7.0	NaN
4	5.0	9.0	7.0

```
print(df[df["B"].notnull()])
```

	A	B	C
1	4.0	7.0	3.0
3	NaN	7.0	NaN
4	5.0	9.0	7.0

```
print(df["B"].isnull().sum())
```

```
2
```

null

null

sum

, notnull

B null

dataframe

# Let's practice!

PREPROCESSING FOR MACHINE LEARNING IN PYTHON

# Working With Data Types

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



**Sarah Guido**  
Senior Data Scientist

# Why are types important?

```
print(volunteer.dtypes)
```

```
opportunity_id    int64
content_id        int64
vol_requests      int64
...              ...
summary          object
is_priority       object
category_id       float64
```

- object: string/mixed types
- int64: integer
- float64: float
- datetime64 (or timedelta):  
datetime

가 object, int64 float64  
Panda가  
int64 Python  
64 가

# Converting column types

```
print(df)
```

	A	B	C
0	1	string	1.0
1	2	string2	2.0
2	3	string3	3.0

C      object

```
print(df.dtypes)
```

A	int64
B	object
C	object
dtype: object	

type      .

# Converting column types

```
print(df)
```

	A	B	C
0	1	string	1.0
1	2	string2	2.0
2	3	string3	3.0

```
df["C"] = df["C"].astype("float")  
print(df.dtypes)
```

astype

```
A      int64  
B      object  
C      float64  
dtype: object
```

# Let's practice!

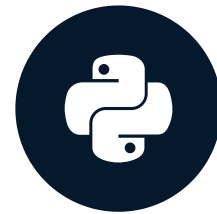
PREPROCESSING FOR MACHINE LEARNING IN PYTHON



가

# Training and Test Sets

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



**Sarah Guido**  
Senior Data Scientist

# Splitting up your dataset

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

```
X_train y_train
0      1.0      n
1      4.0      n
...
5      5.0      n
6      6.0      n
```

```
X_test y_test
0      9.0      y
1      1.0      n
2      4.0      n
```

가  
가  
EX)  
train  
n  
test  
train  
test dataset

# Stratified sampling

- 100 samples, 80 class 1 and 20 class 2
- Training set: 75 samples, 60 class 1 and 15 class 2
- Test set: 25 samples, 20 class 1 and 5 class 2

dataset가 100 sample가 80 class 1, 20 class 2가 .  
train set test set sample 80%가 class 1 20%가 class2 ,  
가 60 class 1 sampl 15 class 2 sample 75 sample train  
25 , 20 1 5 2 가

# Stratified sampling

```
# Total "labels" counts  
y["labels"].value_counts()
```

```
class1    80  
class2    20  
Name: labels, dtype: int64
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y)
```

# Stratified sampling

```
y_train["labels"].value_counts()
```

```
class1    60  
class2    15  
Name: labels, dtype: int64
```

train    test label

```
y_test["labels"].value_counts()
```

```
class1    20  
class2     5  
Name: labels, dtype: int64
```

y

# Let's practice!

PREPROCESSING FOR MACHINE LEARNING IN PYTHON