DATA PREPOCESSING

Welcome to AI

Well done, now that you have mastered python, for **Machine learning**, it is important for you to start learning AI!

As we have introduced in the first chapter, our work will be divided into three major parts ,which are:

1. Data preprocessing
2. Data Visualization
3. Machine learning

Here's a glimpse of the steps that we will learn during this course

Data Preprocessing Schema

3

3

Data preprocessing, Why?

A common mistake is that people tend to pass directly from loading data to machine learning.It is not considered good practice since we usually come across lots of data which is not fit to be directly processed by Machine Learning Algorithms, we call them Raw Data.

So first we have to preprocess our data by

1. Cleaning it.
2. Transforming it.
3. Selecting the best features for our predictions.

In this chapter we will learn various techniques for preprocessing data using Python.

But first we have to understand our data structure and our variable types.

4

4

What’s Input and Output?

Usually our data is divided into Inputs columns and one output column.

For example suppose we want to predict if a passenger is going to survive or not

1. “Survived” is our output column also called target values .

2. The rest of the columns are called input .We will use them to predict the output (also called Features).

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500

Output : Target

Input: Features

5

5

What's Categorical Data?

In fact, we can distinguish two types of variables, categorical and numerical.

The variable is Categorical when our variable:

- Is represented by numbers, words or text.
- Take a limited number of values that it can be **nominal** or **ordinal**:

1. **nominal**: when the labels are unordered.
2. **ordinal**: It is nearly the same as nominal data, except that it’s ordering matters.

6

6

What's Numerical Data?

The data is numerical when our variable:

- Is represented by numbers.
- Take an infinite number of values that it can be discrete or continuous.

For example, if we consider:

- The age of an employee is **discrete**.
- The price of a house is **continuous**.

7

7

- In AI there are three major parts:

1. **Data Preprocessing**.
2. **Data Visualization**.
3. **Machine Learning prediction**.

- Our variables can be **numerical** or **categorical**.
- **Output** is our target.
- **Input**, also called feature, is the tool we use to reach our target.

8

8

Data Cleaning

Now let's start preprocessing :

First step is to clean our data by dealing with the missing values.

In fact, some of the typical reasons why data is missing is that :

1. User forgot to fill in a field
2. Data was lost while transferring manually from a legacy database (using older database technology)

But how is that ?

We start by:

- Finding the **missing** or **incorrect** values.
- Rectifying them by **modifying** or **deleting **.

9

9

Missing Values

Missing values appear as NaN in a DataFrame but what does that mean?

NaN = Not a Number

In fact, when creating a dataframe, pandas replaces missing values with NaN.

Here's how they look like in a DataFrame :

10

10

Finding missing values

A good way to get a quick feel for the data is to take a look at the first few rows.

Here's how you would do that with Pandas:

If you look closely we can notice the presence of missing values:

11

11

Finding missing values: isnull()

But that is not enough, we should search deeper using the **isnull()** function directly on a column which indicates the position of the missing values

```
print(df['Cabin'].head())
print(df['Cabin'].head().isnull())
```

```
0     NaN
1     C85
2     NaN
3    C123
4     NaN
Name: Cabin, dtype: object
0     True
1    False
2     True
3    False
4     True
Name: Cabin, dtype: bool
```

12

12

Finding missing values: isnull.sum()

We can get a summary on the number of missing values on

Each column: By using the **isnull.sum** command

We can also get the total number of missing values in the DataFrame by using the following command **print df.isnull().sum().sum()**

13

13

Dropping missing values: dropna

Now that we know how to find missing data ,we have to deal with them.

The first way is to simply drop(delete) them using the dropna() method.

So let's understand it together :

14

14

dropna() Examples

Here's some examples:

- Drop the columns where any of the elements are missing values :
- Drop the columns where all its elements are missing values :
- Keep only the rows which contain 2 missing values maximum

But sometimes dropping rows or columns isn't a really good solution since we are going to lose all the information on that row or column.

So what is the alternative solution ?

15

15

Replacing missing numerical values

One of the alternative solutions is to simply replace them.

In case of numerical values: We can use the **fillna()** function to replace the missing values of the column "Age" with:

- Mean:
- Median:
- Mode:

16

16

Replacing missing categorical values

In the case of categorical values:

Replace the missing value with the **most frequent value** by using the **fillna()** function.

```
number_of_elements = len(df["Cabin"])
print("Number of elements: ",number_of_elements)
#Number of elements per category
print(df["Cabin"].value_counts())
#Replace and display the values
df["Cabin"].fillna('G6',inplace=True)
df.tail()
```

17

17

- We usually encounter some missing values in our dataset which appear as **NaN**.
- We can calculate the **sum** of all the missing value with **df.isnull().sum().sum()**.
- To **delete** all the missing values you can use **df.dropna(axis=1)**.
- To **replace** the missing values for a specific column by it's mean you can use **df['Age'].fillna(df['Age'].mean(),inplace=True)**.

18

18

Feature Transformation: Why?

Let's try to recap what we have done so far :

We did learn how to:

- Load dataset to our jupyter notebook file.
- Deal with the missing values.

Now as a final step in the preprocessing, we will learn how to;

- transform all the object type values to numerical values

Since that most of the **machine learning algorithms** and **correlation metrics** are based on **numerical values**, we need to **convert** all the data into numerical values.

19

19

Categorical to Numerical

1. If we observe the type of our features we will note the presence of type <<object>>
2. The Next step consists of determining which one of these object features is categorical.

Noting that the role can take only 6 possibilities which are

Then we can consider the role feature as a categorical feature

20

20

Label encoder function

The first method consist of converting the modalities ,the values that a variable can take, of each **categorical variable** to **a number**.

We can

- Do this by hand using a dictionary when we have 2 or 3 categories.
- Use the label encoder function in the sklearn library.

Note that we will explain in details the sklearn library in the Machine Learning chapter later on.

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
new_data['Role']=encoder.fit_transform(new_data['Role'])
new_data
```

	Age	Role	Home Adress	Full Name
0	23	0	Los Angeles	John Stott
1	20	0	California	Jack Mcbride
2	21	4	New York	Angelica Newman
3	22	5	Florida	Andrew Hines

21

21

Label Encoder Dictionary

This is how you do the same thing but with a dictionary:

We create a new dictionary then we replace categorical values to numerical values.

22

22

One-hot-encoding Principle

This method consists of transforming each modality of the categorical variable to a new feature

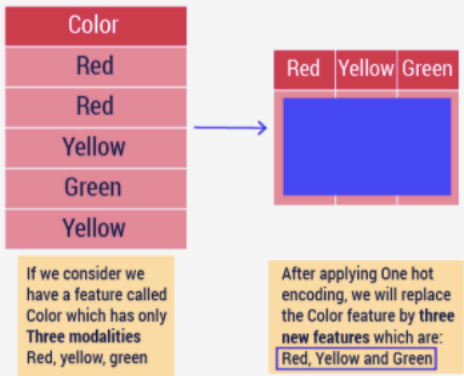
But how is that ?

23

One-hot-encoding Principle

This method consists of transforming each modality of the categorical variable to a new feature

But how is that ?



24

One hot encoding return

But how the values of the new features are filled ?

Color
Red
Red
Yellow
Green
Yellow

→

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

If we choose a specific row:

- Before the transformation, in the corresponding column color, we have for example the value Red.
- After applying one hot encoding, we will find 1 in the column Red and 0's in Yellow and Green.

25

25

One hot encoding Code

Let's try to apply this to our dataset to the categorical feature "Role"

```
# Get one hot encoding of columns Role
one_hot=pd.get_dummies(new_data['Role'])
# Drop column Role as it is now encoded
new_data = new_data.drop('Role',axis = 1)
# Join the encoded df
new_data = new_data.join(one_hot)
new_data
```

	Age	Home Adress	Full Name	AI	Datascience	Manager	WEB1	WEB2
0	23	Los Angeles	John Stott	1	0	0	0	0
1	20	California	Jack Mcbride	1	0	0	0	0
2	21	New York	Angelica Newman	0	0	0	1	0
3	22	Florida	Andrew Hines	0	0	0	0	1
4	23	Los Angeles	Isac Justice	0	0	1	0	0
5	24	Washington	Tyrone Buck	0	1	0	0	0

26

26

Feature Selection

Feature selection is the part of preprocessing where we decide which **feature** we are going to introduce as **inputs** to our machine learning algorithm.

It is generally done in two parts :

1. First, we will start the selection based on our understanding of the business and our logic.

For example

- If we are predicting the height of a plant, as biologists we know that we need to have the species.
- If we are trying to predict a salary of an employee logically we will not need his weight or height.

2. The second part will be done during the data visualization.

27

27

We have two methods to transform our data either we use:

Label encoder which will transform each categorical modality to a numerical modality.

One hot encoding which will transform each categorical modality to a new feature.

28

28