Handling missing values in datasets

Presenters:

Garaev Albert
Novikova Ksenia
Khabibulloev Mukhammadsodik

25/02/2022

MACHINE LEARNING GROUP
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

INFORMATIK

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Table of Contents

What is a Missing Value?

What is a Missing Value?

Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given dataset.

The image shows the first few records of the Titanic dataset extracted and displayed using Pandas.

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
5	897	3	Svensson, Mr. Johan Cervin	male	14.0	0	0	7538	9.2250	NaN	S
6	898	3	Connolly, Miss. Kate	female	30.0	0	0	330972	7.6292	NaN	Q
7	899	2	Caldwell, Mr. Albert Francis	male	26.0	1	1	248738	29.0000	NaN	S
8	900	3	Abrahim, Mrs. Joseph (Sophie Halaut Easu)	female	18.0	0	0	2657	7.2292	NaN	C
9	901	3	Davies, Mr. John Samuel	male	21.0	2	0	A/4 48871	24.1500	NaN	S
10	902	3	Ilieff, Mr. Ylio	male	NaN	0	0	349220	7.8958	NaN	S
11	903	1	Jones, Mr. Charles Cresson	male	46.0	0	0	694	26.0000	NaN	S
12	904	1	Snyder, Mrs. John Pillsbury (Nelle Stevenson)	female	23.0	1	0	21228	82.2667	B45	S
13	905	2	Howard, Mr. Benjamin	male	63.0	1	0	24065	26.0000	NaN	S
14	906	1	Chaffee, Mrs. Herbert Fuller (Carrie Constance...	female	47.0	1	0	W.E.P. 5734	61.1750	E31	S

Not a Number

Why Is Data Missing From The Dataset?

Some of the reasons are listed below:

- Past data might get corrupted due to improper maintenance.
- Observations are not recorded for certain fields due to some reasons. There might be a failure in recording the values due to human error.
- The user has not provided the values intentionally.

Why Do We Need To Care About Handling Missing Value?

- Many machine learning algorithms fail if the dataset contains missing values. However, algorithms like K-nearest and Naive Bayes support data with missing values.

- You may end up building a biased machine learning model which will lead to incorrect results if the missing values are not handled properly.

- Missing data can lead to a lack of precision in the statistical analysis.

Delete Rows with Missing Values

Missing values can be handled by **deleting** the rows or columns having **null** values.

before

```
print(df.isnull().sum())
print(df.shape)
```

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
(418, 11)
```

handling →

after

```
df.dropna(inplace=True)
print(df.isnull().sum())
print(df.shape)
```

```
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
(87, 11)
```

Delete Rows with Missing Values

Missing values can be handled by **deleting** the rows or columns having **null** values.

before

```
print(df.isnull().sum())
print(df.shape)
```

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
(418, 11)
```

handling

after

```
df.dropna(inplace=True)
print(df.isnull().sum())
print(df.shape)
```

```
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
(87, 11)
```

Pros:

- A model trained with the removal of all missing values creates a robust model.

Cons:

- Loss of a lot of information.
- Works poorly if the percentage of missing values is excessive in comparison to the complete dataset.

Impute missing values with Mean/Median

Columns in the dataset which are having numeric continuous values can be replaced with the **mean, median, or mode** of remaining values in the column.

before

```
df['Age'][5:20]
```

```
5       NaN
6      54.0
7       2.0
8      27.0
9      14.0
10      4.0
11     58.0
12     20.0
13     39.0
14     14.0
15     55.0
16      2.0
17      NaN
18     31.0
19      NaN
Name: Age, dtype: float64
```

handling

after

```
df['Age']=df['Age'].replace(np.NaN, df['Age'].mean())
print(df['Age'][5:20])
```

```
5     29.699118
6     54.000000
7      2.000000
8     27.000000
9     14.000000
10     4.000000
11    58.000000
12    20.000000
13    39.000000
14    14.000000
15    55.000000
16     2.000000
17    29.699118
18    31.000000
19    29.699118
Name: Age, dtype: float64
```

Impute missing values with Mean/Median

Columns in the dataset which are having numeric continuous values can be replaced with the **mean, median, or mode** of remaining values in the column.

before

```
df['Age'][5:20]

5       NaN
6      54.0
7       2.0
8      27.0
9      14.0
10      4.0
11     58.0
12     20.0
13     39.0
14     14.0
15     55.0
16      2.0
17      NaN
18     31.0
19      NaN
Name: Age, dtype: float64
```

handling

after

```
df['Age']=df['Age'].replace(np.NaN, df['Age'].mean())
print(df['Age'][5:20])

5     29.699118
6     54.000000
7      2.000000
8     27.000000
9     14.000000
10     4.000000
11    58.000000
12    20.000000
13    39.000000
14    14.000000
15    55.000000
16     2.000000
17    29.699118
18    31.000000
19    29.699118
Name: Age, dtype: float64
```

Pros:

- Better than deletion of rows or columns
- Works well with a small dataset and is easy to implement.

Cons:

- Only with numerical continuous variables.
- Can cause data leakage
- Do not factor the covariance between features.

Imputation method for categorical columns

When missing values is from categorical columns (string or numerical) then the missing values can be replaced with the most frequent category. If the number of missing values is very large then it can be replaced with a new category.

before

```
df.isnull().sum()

PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

handling

after

```
df['Cabin']=df['Cabin'].fillna('U')
df.isnull().sum()

PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin            0
Embarked         2
dtype: int64
```

```
df[df.columns[8:11]]
```

	Ticket	Fare	Cabin
0	A/5 21171	7.2500	U
1	PC 17599	71.2833	C85
2	STON/O2. 3101282	7.9250	U
3	113803	53.1000	C123
4	373450	8.0500	U
...	...	...	...
886	211536	13.0000	U
887	112053	30.0000	B42

Other Imputation Methods

Depending on the nature of the data or data type, some other imputation methods may be more appropriate to impute missing values.

before

```
data=pd.read_csv('train.csv')

print(data.Age)

0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888     NaN
889    26.0
890    32.0
```

handling →

after

```
data["Age"] = data["Age"].fillna(method='ffill')

print(data.Age)

0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888    19.0
889    26.0
890    32.0
```

For example, for the data variable having longitudinal behavior, it might make sense to use the last valid observation to fill the missing value. This is known as the Last observation carried forward (LOCF) method.

Other Imputation Methods

before

```
data=pd.read_csv('train.csv')
```

```
print(data.Age)
```

```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888     NaN
889    26.0
890    32.0
```

handling

after

```
data["Age"] = data["Age"].interpolate(method='linear', limit_direction='forward', axis=0)
```

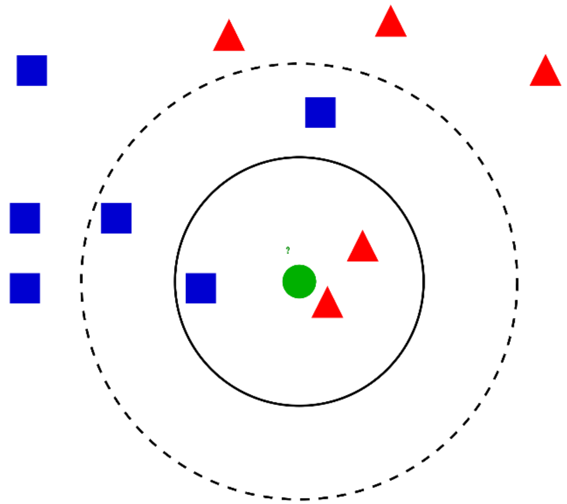
```
print(data.Age)
```

```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888    22.5
889    26.0
890    32.0
```

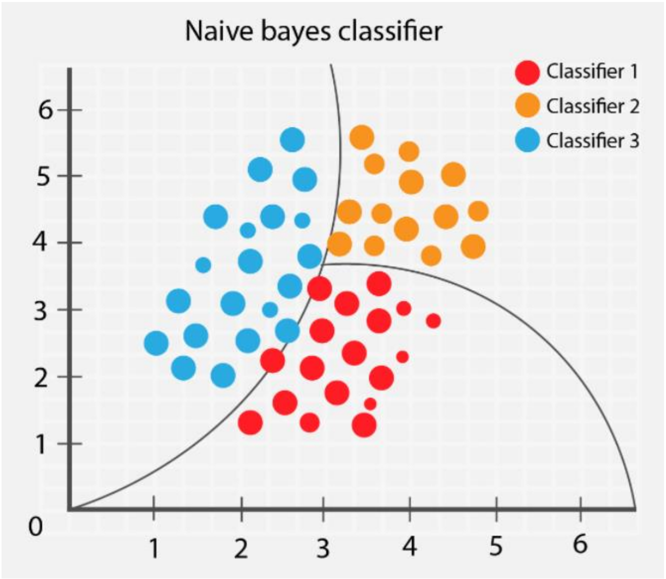
For the time-series dataset variable, it makes sense to use the interpolation of the variable before and after a timestamp for a missing value.

Using Algorithms that support missing values

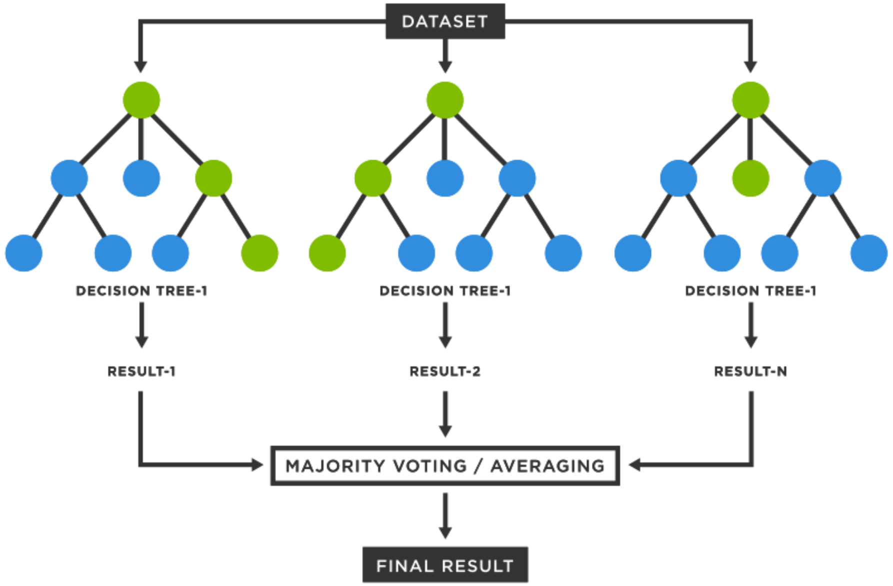
All the machine learning algorithms don't support missing values but some ML algorithms are robust to missing values in the dataset.



The k-NN algorithm can ignore a column from a distance measure when a value is missing. While the algorithm is applied, KNN considers the missing values by taking the majority of the K nearest values.



Naive Bayes can also support missing values when making a prediction.



Random Forest works well on non-linear and categorical data. It adapts to the data structure taking into consideration the high variance or the bias, producing better results on large datasets.

Using Algorithms that support missing values

Pros:

- No need to handle missing values in each column as ML algorithms will handle them efficiently.

- Correlation of the data is neglected

Cons:

- No implementation of these ML algorithms in the scikit-learn library.
- Is a very time consuming process and it can be critical in data mining where large databases are being extracted
- Choice of distance functions can be Euclidean, Manhattan etc. which is do not yield a robust result

Prediction of missing values

Missing values can be predicted using the other features that have non-null values.

The *classification* or *regression* model can be used for the prediction of missing values depending on the nature (*categorical* or *continuous*) of the feature having missing value.

Pros:

- Gives a better result than earlier methods
- Takes into account the covariance between the missing value column and other columns

Cons:

- Considered only as a proxy for the true values

Prediction of missing values

- Categorical (numerical, object)

  For prediction a *classification* *machine learning algorithm* is required such as **Logistic Regression, SVM, Naive Bayes**, etc.

- Continuous Variable (numerical)

  For prediction a *regression* *machine learning algorithm* is required such as **Linear Regression, SVR**, etc.

Imputation using Deep Learning Library

This method works very well with such features as:

- categorical
- continuous
- non-numerical

Datawig is a library that learns Machine Learning models using Deep Neural Networks to impute missing values in the datagram.

Pros:

- Quite accurate compared to other methods.
- It supports CPUs and GPUs.

Cons:

- Can be quite slow with large datasets.

Conclusion

Each dataset *has missing values* that need to be handled intelligently to create a robust model.

For handling missing values we need:

- explore the data
- find out what variables have missing data
- what is the percentage
- what category does it belong to

There is *no rule* for handling missing values in a specific way.

Various methods are used for different functions depending on the data type.

Thank you for your attention!

5. References

[1] Fake News Detection.

[2] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "FakeNewsNet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media," *Big Data*, vol. 8, no. 3, pp. 171188, Jun. 2020.

[3] F. K. A. Salem, R. Al Feel, S. Elbassuoni, M. Jaber, and M. Farah, "FA-KES: A fake news dataset around the Syrian war," in *Proc. Int. AAAI Conf. Web Social Media*, vol. 13, 2019, pp. 573582.

[4] H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in *Proc. Int. Conf. Intell., Secure, Dependable Syst. Distrib. Cloud Environ.* Vancouver, BC, Canada: Springer, 2017, pp. 127138.

[5] J. Bergstra, D. Yamins, and D. Cox, "HyperOpt: Distributed asynchronous hyper-parameter optimization," *Retrieved May*, vol. 21, p. 2020, 2012.

[6] A. Sikandar, W. Anwar, U. I. Bajwa, X. Wang, M. Sikandar, L. Yao, Z. L. Jiang, and Z. Chunkai, "Decision tree based approaches for detecting protein complex in protein interaction network (PPI) via link and sequence analysis," *IEEE Access*, to vol. 6, pp. 22108_22120, 2018.