How to Handle Missing Data with Python

by **Jason Brownlee** on March 20, 2017 in **Data Preparation**

Last Updated on August 28, 2020

Real-world data often has **missing values**.

Data can have missing values for a number of reasons such as observations that were not recorded and data corruption.

Handling missing data is important as many machine learning algorithms do not support data with missing values.

In this tutorial, you will discover how to handle missing data for machine learning with Python.

Specifically, after completing this tutorial you will know:

- How to marking invalid or corrupt values as missing in your dataset.
- How to remove rows with missing data from your dataset.
- How to impute missing values with mean values in your dataset.

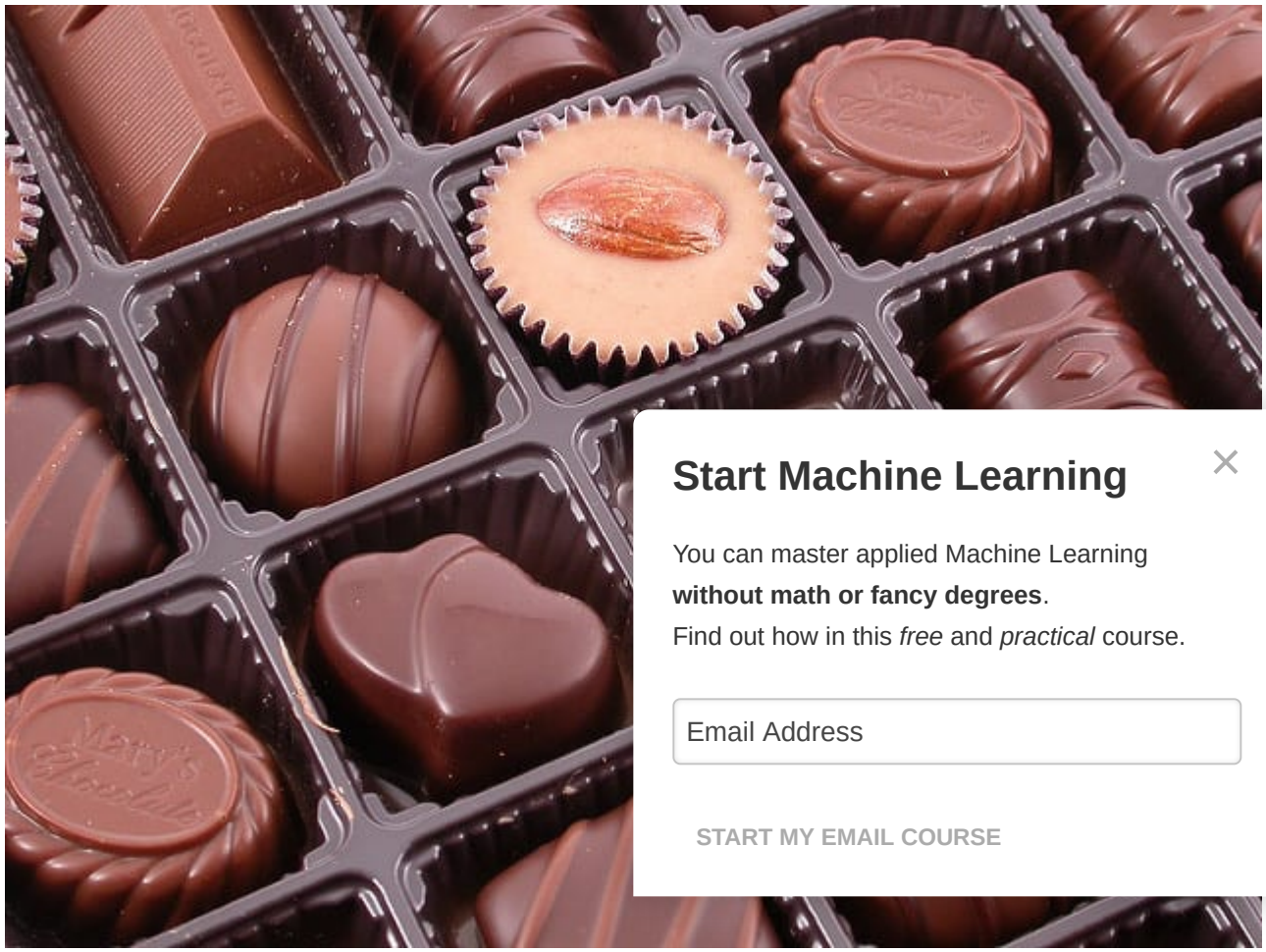
Kick-start your project with my new book Data Preparation for Machine Learning, including *step-by-step tutorials* and the *Python source code* files for all examples.

Let's get started.

Note: The examples in this post assume that you have Python 3 with Pandas, NumPy and Scikit-Learn installed, specifically scikit-learn version 0.22 or higher. If you need help setting up your environment see this tutorial.

- **Update Mar/2018**: Changed link to dataset files.
- **Update Dec/2019**: Updated link to dataset to Gith

- **Update May/2020**: Updated code examples for API changes. Added references.



How to Handle Missing Values with Python
Photo by CoCreatr, some rights reserved.

Overview

This tutorial is divided into 6 parts:

1. **Diabetes Dataset**: where we look at a dataset that has known missing values.
2. **Mark Missing Values**: where we learn how to mark missing values in a dataset.
3. **Missing Values Causes Problems**: where we see how a machine learning algorithm can fail when it contains missing values.
4. **Remove Rows With Missing Values**: where we see how to remove rows that contain missing values.
5. **Impute Missing Values**: where we replace missing values with sensible values.
6. **Algorithms that Support Missing Values**: where we learn about algorithms that support missing values.

First, let's take a look at our sample dataset with missing values.

1. Diabetes Dataset

The Diabetes Dataset involves predicting the onset of

Start Machine Learning

- Dataset File.
- Dataset Details

It is a binary (2-class) classification problem. The number of observations for each class is not balanced. There are 768 observations with 8 input variables and 1 output variable. The variable names are as follows:

- 0. Number of times pregnant.
- 1. Plasma glucose concentration a 2 hours in an oral glucose tolerance test.
- 2. Diastolic blood pressure (mm Hg).
- 3. Triceps skinfold thickness (mm).
- 4. 2-Hour serum insulin (mu U/ml).
- 5. Body mass index (weight in kg/(height in m)^2)
- 6. Diabetes pedigree function.
- 7. Age (years).
- 8. Class variable (0 or 1).

The baseline performance of predicting the most prev
approximately 65%. Top results achieve a classificatio

A sample of the first 5 rows is listed below.

```
6,148,72,35,0,33.6,0.627,50,1
1,85,66,29,0,26.6,0.351,31,0
8,183,64,0,0,23.3,0.672,32,1
1,89,66,23,94,28.1,0.167,21,0
0,137,40,35,168,43.1,2.288,33,1
...
```

This dataset is known to have missing values.

Specifically, there are missing observations for some columns that are marked as a zero value.

We can corroborate this by the definition of those columns and the domain knowledge that a zero value is invalid for those measures, e.g. a zero for body mass index or blood pressure is invalid.

Download the dataset from here and save it to your current working directory with the file name *pima-indians-diabetes.csv* .

- pima-indians-diabetes.csv

---

2. Mark Missing Values

Most data has missing values, and the likelihood of having missing values increases with the size of the dataset.

> *Missing data are not rare in real data sets. In fact, the chance that at least one data point is missing increases as the data set size increas*

— Page 187, Feature Engineering and Selection, 201

In this section, we will look at how we can identify and

We can use plots and summary statistics to help ident

We can load the dataset as a Pandas DataFrame and


```
# load and summarize the dataset
from pandas import read_csv
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv', header=None)
# summarize the dataset
print(dataset.describe())
```

Running this example produces the following output:

```
                0           1           2  ...           6           7           8
count  768.000000  768.000000  768.000000  ...  768.000000  768.000000  768.000000
mean     3.845052  120.894531   69.105469  ...    0.471876   33.240885    0.348958
std      3.369578   31.972618   19.355807  ...    0.331329   11.760232    0.476951
min      0.000000    0.000000    0.000000  ...    0.078000   21.000000    0.000000
25%      1.000000   99.000000   62.000000  ...    0.243750   24.000000    0.000000
50%      3.000000  117.000000   72.000000  ...    0.372500   29.000000    0.000000
75%      6.000000  140.250000   80.000000  ...    0.626250   41.000000    1.000000
max     17.000000  199.000000  122.000000  ...    2.420000   81.000000    1.000000

[8 rows x 9 columns]
```

This is useful.

We can see that there are columns that have a minimum value of zero (0). On some columns, a value of zero does not make sense and indicates an invalid or missing value.

> *Missing values are frequently indicated by out-of-range entries; perhaps a negative number (e.g., -1) in a numeric field that is normally only positive, or a 0 in a numeric field that can never normally be 0.*

— Page 62, Data Mining: Practical Machine Learning Tools and Techniques, 2016.

Specifically, the following columns have an invalid zero minimum value:

- 1: Plasma glucose concentration
- 2: Diastolic blood pressure
- 3: Triceps skinfold thickness
- 4: 2-Hour serum insulin
- 5: Body mass index

Let's confirm this my looking at the raw data, the example prints the first 20 rows of data.

```
# load the dataset and review rows
from pandas import read_csv
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv'
# print the first 20 rows of data
print(dataset.head(20))
```


Running the example, we can clearly see 0 values in t

```
      0    1   2   3    4     5      6   7  8
0     6  148  72  35    0  33.6  0.627  50  1
1     1   85  66  29    0  26.6  0.351  31  0
2     8  183  64   0    0  23.3  0.672  32  1
3     1   89  66  23   94  28.1  0.167  21  0
4     0  137  40  35  168  43.1  2.288  33  1
5     5  116  74   0    0  25.6  0.201  30  0
6     3   78  50  32   88  31.0  0.248  26  1
7    10  115   0   0    0  35.3  0.134  29  0
8     2  197  70  45  543  30.5  0.158  53  1
9     8  125  96   0    0   0.0  0.232  54  1
10    4  110  92   0    0  37.6  0.191  30  0
11   10  168  74   0    0  38.0  0.537  34  1
12   10  139  80   0    0  27.1  1.441  57  0
13    1  189  60  23  846  30.1  0.398  59  1
14    5  166  72  19  175  25.8  0.587  51  1
15    7  100   0   0    0  30.0  0.484  32  1
16    0  118  84  47  230  45.8  0.551  31  1
17    7  107  74   0    0  29.6  0.254  31  1
18    1  103  30  38   83  43.3  0.183  33  0
19    1  115  70  30   96  34.6  0.529  32  1
```

We can get a count of the number of missing values on each of these columns. We can do this my marking all of the values in the subset of the DataFrame we are interested in that have zero values as True. We can then count the number of true values in each column.

We can do this my marking all of the values in the subset of the DataFrame we are interested in that have zero values as True. We can then count the number of true values in each column.

```
# example of summarizing the number of missing values for each variable
from pandas import read_csv
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv'
# count the number of missing values for each
```

```
num_missing = (dataset[[1,2,3,4,5]] == 0).sum()
# report the results
print(num_missing)
```

Running the example prints the following output:

```
1  5
2  35
3  227
4  374
5  11
```

We can see that columns 1,2 and 5 have just a few zero values, whereas columns 3 and 4 show a lot more, nearly half of the rows.

This highlights that different "missing value" strategies ure that there are still a sufficient number of records left to

> *When a predictor is discrete in nature, missing if it were a naturally occurring category.*

— Page 197, Feature Engineering and Selection, 201

In Python, specifically Pandas, NumPy and Scikit-Lea

Values with a NaN value are ignored from operations

We can mark values as NaN easily with the Pandas DataFrame by using the replace() function on a subset of the columns we are interested in.

After we have marked the missing values, we can use the isnull() function to mark all of the NaN values in the dataset as True and get a count of the missing values for each column.

```
# example of marking missing values with nan values
from numpy import nan
from pandas import read_csv
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv', header=None)
# replace '0' values with 'nan'
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].replace(0, nan)
# count the number of nan values in each column
print(dataset.isnull().sum())
```

Running the example prints the number of missing values in each column. We can see that the columns 1:5 have the same number of missing values as zero values identified above. This is a sign that we have marked the identified missing values correctly.

We can see that the columns 1 to 5 have the same number of missing values as zero values identified above. This is a sign that we have marked the identified missing values correctly.

```
1  0      0
2  1      5
```

```
3  2     35
4  3    227
5  4    374
6  5     11
7  6      0
8  7      0
9  8      0
10 dtype: int64
```

This is a useful summary. I always like to look at the actual data though, to confirm that I have not fooled myself.

Below is the same example, except we print the first 20 rows of data.

```
# example of review rows from the dataset with missing values marked
from numpy import nan
from pandas import read_csv
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv'
# replace '0' values with 'nan'
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].re
# print the first 20 rows of data
print(dataset.head(20))
```


Running the example, we can clearly see NaN values
missing values in column 1, so it is not surprising we

It is clear from the raw data that marking the missing v

```
1        0      1     2     3      4     5      6   7  8
2   0    6  148.0  72.0  35.0    NaN  33.6  0.627  50  1
3   1    1   85.0  66.0  29.0    NaN  26.6  0.351  31  0
4   2    8  183.0  64.0   NaN    NaN  23.3  0.672  32  1
5   3    1   89.0  66.0  23.0   94.0  28.1  0.167  21  0
6   4    0  137.0  40.0  35.0  168.0  43.1  2.288  33  1
7   5    5  116.0  74.0   NaN    NaN  25.6  0.201  30  0
8   6    3   78.0  50.0  32.0   88.0  31.0  0.248  26  1
9   7   10  115.0   NaN   NaN    NaN  35.3  0.134  29  0
10  8    2  197.0  70.0  45.0  543.0  30.5  0.158  53  1
11  9    8  125.0  96.0   NaN    NaN   NaN  0.232  54  1
12  10   4  110.0  92.0   NaN    NaN  37.6  0.191  30  0
13  11  10  168.0  74.0   NaN    NaN  38.0  0.537  34  1
14  12  10  139.0  80.0   NaN    NaN  27.1  1.441  57  0
15  13   1  189.0  60.0  23.0  846.0  30.1  0.398  59  1
16  14   5  166.0  72.0  19.0  175.0  25.8  0.587  51  1
17  15   7  100.0   NaN   NaN    NaN  30.0  0.484  32  1
18  16   0  118.0  84.0  47.0  230.0  45.8  0.551  31  1
19  17   7  107.0  74.0   NaN    NaN  29.6  0.254  31  1
20  18   1  103.0  30.0  38.0   83.0  43.3  0.183  33  0
21  19   1  115.0  70.0  30.0   96.0  34.6  0.529  32  1
```

Before we look at handling missing values, let's first demonstrate that having missing values in a dataset can cause problems.

3. Missing Values Causes Problems

Having missing values in a dataset can cause errors with some machine learning algorithms

> *Missing values are common occurrences in data. Unfortunately, most predictive modeling techniques cannot handle any missing values. Therefore, this problem must be addressed prior to modeling.*

— Page 203, Feature Engineering and Selection, 2019.

In this section, we will try to evaluate a the Linear Discriminant Analysis (LDA) algorithm on the dataset with missing values.

This is an algorithm that does not work when there are missing values in the dataset.

The below example marks the missing values in the d
attempts to evaluate LDA using 3-fold cross validation


```
# example where missing values cause errors
from numpy import nan
from pandas import read_csv
from sklearn.discriminant_analysis import Lin
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv
# replace '0' values with 'nan'
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].r
# split dataset into inputs and outputs
values = dataset.values
X = values[:,0:8]
y = values[:,8]
# define the model
model = LinearDiscriminantAnalysis()
# define the model evaluation procedure
cv = KFold(n_splits=3, shuffle=True, random_state=1)
# evaluate the model
result = cross_val_score(model, X, y, cv=cv, scoring='accuracy')
# report the mean performance
print('Accuracy: %.3f' % result.mean())
```

Running the example results in an error, as follows:

```
ValueError: Input contains NaN, infinity or a value too large for dtype('float64').
```

This is as we expect.

We are prevented from evaluating an LDA algorithm (and other algorithms) on the dataset with missing values.

> *Many popular predictive models such as support vector machines, the glmnet, and neural networks, cannot tolerate any amount of missing values.*

— Page 195, Feature Engineering and Selection, 2019.

Now, we can look at methods to handle the missing va

4. Remove Rows With Missing Values

The simplest strategy for handling missing data is to remove records that contain a missing value.

> *The simplest approach for dealing with missing values is to remove entire predictor(s) and/or sample(s) that contain missing values.*

— Page 196, Feature Engineering and Selection, 2019.

We can do this by creating a new Pandas DataFrame with the rows containing missing values removed.

Pandas provides the dropna() function that can be use a.
We can use dropna() to remove all rows with missing

```
# example of removing rows that contain missi
from numpy import nan
from pandas import read_csv
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv
# summarize the shape of the raw data
print(dataset.shape)
# replace '0' values with 'nan'
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].r
# drop rows with missing values
dataset.dropna(inplace=True)
# summarize the shape of the data with missin
print(dataset.shape)
```


Running this example, we can see that the number of rows has been aggressively cut from 768 in the original dataset to 392 with all rows containing a NaN removed.

```
(768, 9)
(392, 9)
```

We now have a dataset that we could use to evaluate an algorithm sensitive to missing values like LDA.

```
# evaluate model on data after rows with missing data are removed
from numpy import nan
from pandas import read_csv
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv', header=None)
# replace '0' values with 'nan'
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].replace(0, nan)
# drop rows with missing values
dataset.dropna(inplace=True)
# split dataset into inputs and outputs
values = dataset.values
X = values[:,0:8]
y = values[:,8]
# define the model
model = LinearDiscriminantAnalysis()
# define the model evaluation procedure
cv = KFold(n_splits=3, shuffle=True, random_s
```

```
# evaluate the model
result = cross_val_score(model, X, y, cv=cv, scoring='accuracy')
# report the mean performance
print('Accuracy: %.3f' % result.mean())
```

Note: Your results may vary given the stochastic nature of the algorithm or evaluation procedure, or differences in numerical precision. Consider running the example a few times and compare the average outcome.

The example runs successfully and prints the accuracy of the model.

```
Accuracy: 0.781
```

Removing rows with missing values can be too limiting on some predictive modeling problems, an alternative is to impute missing values.

5. Impute Missing Values

Imputing refers to using a model to replace missing va

> *… missing data can be imputed. In this case, v predictors to, in essence, estimate the values*

— Page 42, Applied Predictive Modeling, 2013.

There are many options we could consider when replacing a missing value, for example:

- A constant value that has meaning within the domain, such as 0, distinct from all other values.
- A value from another randomly selected record.
- A mean, median or mode value for the column.
- A value estimated by another predictive model.

Any imputing performed on the training dataset will have to be performed on new data in the future when predictions are needed from the finalized model. This needs to be taken into consideration when choosing how to impute the missing values.

For example, if you choose to impute with mean column values, these mean column values will need to be stored to file for later use on new data that has missing values.

Pandas provides the fillna() function for replacing missing values with a specific value.

For example, we can use fillna() to replace missing values with the mean value for each column, as follows:

```
# manually impute missing values with numpy
from pandas import read_csv
from numpy import nan
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv
# mark zero values as missing or NaN
```

```
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].replace(0, nan)
# fill missing values with mean column values
dataset.fillna(dataset.mean(), inplace=True)
# count the number of NaN values in each column
print(dataset.isnull().sum())
```

Running the example provides a count of the number of missing values in each column, showing zero missing values.

```
0    0
1    0
2    0
3    0
4    0
5    0
6    0
7    0
8    0
dtype: int64
```

The scikit-learn library provides the SimpleImputer pre
missing values.

It is a flexible class that allows you to specify the value and
the technique used to replace it (such as mean, media
directly on the NumPy array instead of the DataFrame

The example below uses the SimpleImputer class to r mn
then prints the number of NaN values in the transform


```
# example of imputing missing values using scikit-learn
from numpy import nan
from numpy import isnan
from pandas import read_csv
from sklearn.impute import SimpleImputer
# load the dataset
dataset = read_csv('pima-indians-diabetes.csv', header=None)
# mark zero values as missing or NaN
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].replace(0, nan)
# retrieve the numpy array
values = dataset.values
# define the imputer
imputer = SimpleImputer(missing_values=nan, strategy='mean')
# transform the dataset
transformed_values = imputer.fit_transform(values)
# count the number of NaN values in each column
print('Missing: %d' % isnan(transformed_values).sum())
```

Running the example shows that all NaN values were imputed successfully.

```
Missing: 0
```

In either case, we can train algorithms sensitive to NaN values in the transformed dataset, such as LDA.

The example below shows the LDA algorithm trained in the *SimpleImputer* transformed dataset.

We use a Pipeline to define the modeling pipeline, where data is first passed through the imputer transform, then provided to the model. This ensures that the imputer and model are both fit only on the training dataset and evaluated on the test dataset within each cross-validation fold. This is important to avoid data leakage.

```
# example of evaluating a model after an imputer transform
from numpy import nan
from pandas import read_csv
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
dataset = read_csv('pima-indians-diabetes.csv', header=None)
# mark zero values as missing or NaN
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].r
# split dataset into inputs and outputs
values = dataset.values
X = values[:,0:8]
y = values[:,8]
# define the imputer
imputer = SimpleImputer(missing_values=nan, s
# define the model
lda = LinearDiscriminantAnalysis()
# define the modeling pipeline
pipeline = Pipeline(steps=[('imputer', impute
# define the cross validation procedure
kfold = KFold(n_splits=3, shuffle=True, rando
# evaluate the model
result = cross_val_score(pipeline, X, y, cv=k
# report the mean performance
print('Accuracy: %.3f' % result.mean())
```


Note: Your results may vary given the stochastic nature of the algorithm or evaluation procedure, or differences in numerical precision. Consider running the example a few times and compare the average outcome.

Running the example prints the accuracy of LDA on the transformed dataset.

```
Accuracy: 0.762
```

Try replacing the missing values with other values and see if you can lift the performance of the model.

Maybe missing values have meaning in the data.

For a more detailed example of imputing missing values with statistics see the tutorial:

- Statistical Imputation for Missing Values in Machine Learning

Next we will look at using algorithms that treat missing values as just another value when modeling.

6. Algorithms that Support Missing Values

Not all algorithms fail when there is missing data.

There are algorithms that can be made robust to missing data, such as k-Nearest Neighbors that can ignore a column from a distance measure when a value is missing. Naive Bayes can also support missing values when making a prediction.

> *One of the really nice things about Naive Bayes is that missing values are no problem at all.*

— Page 100, Data Mining: Practical Machine Learning Tools and Techniques, 2016.

There are also algorithms that can use the missing value as a unique and different value when building the predictive model, such as classification and regression trees.

> *… a few predictive models, especially tree-bas*
> *missing data.*

— Page 42, Applied Predictive Modeling, 2013.

Sadly, the scikit-learn implementations of naive bayes
robust to missing values. Although it is being consider

Nevertheless, this remains as an option if you conside
xgboost) or developing your own implementation.

Further Reading

This section provides more resources on the topic if you are looking to go deeper.

Related Tutorials

- Statistical Imputation for Missing Values in Machine Learning

Books

- Feature Engineering and Selection, 2019.
- Data Mining: Practical Machine Learning Tools and Techniques, 2016.
- Feature Engineering and Selection, 2019.
- Applied Predictive Modeling, 2013.

APIs

- Working with missing data, in Pandas
- Imputation of missing values, in scikit-learn

Summary

In this tutorial, you discovered how to handle machine learning data that contains missing values.

Specifically, you learned:

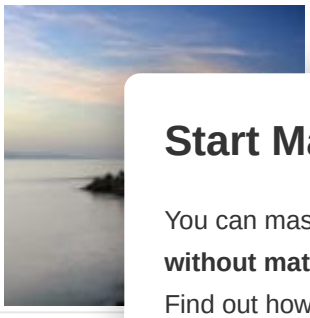
- How to mark missing values in a dataset as numpy.nan.
- How to remove rows from the dataset that contain missing values.
- How to replace missing values with sensible values.

Do you have any questions about handling missing values?
Ask your questions in the comments and I will do my best to answer.

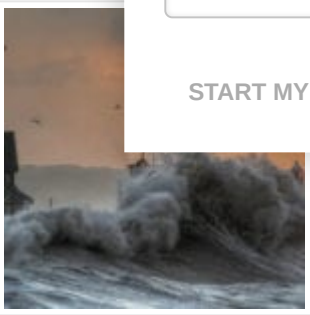
More On This Topic

Start Machine Learning

How To Handle Missing Values In Machine Learning…

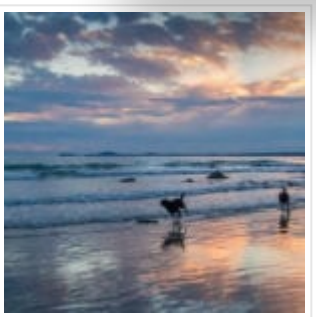


How to Handle Missing

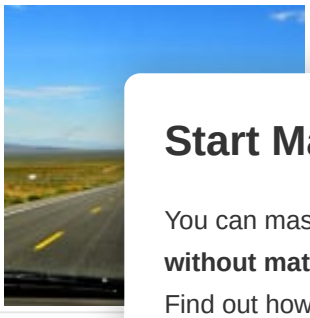


7 Ways to Handle Large Data Files for Machine Learning

Techniques to Handle Very Long Sequences with LSTMs



How to Handle Big-p, Little-n (p >> n) in Machine Learning



The Missing Roadmap to

About Jason Brownlee

Jason Brownlee, PhD is a machine learnin th modern machine learning methods via hands-on tutorials.
View all posts by Jason Brownlee →

‹ How to Train a Final Machine Learning Model

Time Series Forecasting with Python 7-Day Mini-Course ›

141 Responses to *How to Handle Missing Data with Python*

Mike March 20, 2017 at 3:16 pm #

REPLY

Fancy impute is a library i've turned too for imputation:

https://github.com/hammerlab/fancyimpute

Also missingno is great for visualizations!

https://github.com/ResidentMario/missingno

Jason Brownlee March 21, 2017 at 8:37 am #

REPLY

Thanks for the tip Mike.

ishtiaq ahmed December 10, 2019 at 4:28 am #

REPLY

Hi, friend I need that dataset ” Pima-Indians-diabetes.csv” how can I access it. it is not available on this site

Jason Brownlee December 10, 201

All datasets are here:
https://github.com/jbrownlee/Datasets

ishtiaq ahmed December 11, 2

thnx Jason

Jason Brownlee December 11, 2019 at 7:02 am #

You’re welcome.

umer January 10, 2020 at 8:06 pm #

Hi,
I have a data set with 3 lakhs row and 278 columns. I used MissForest to impute missing values. But, the system (HP Pavilion Intel i5 with 12GB RAM) runs for a long time and still didn’t complete..Can you suggest any easy way?
should I have to use any loop?

Jason Brownlee January 11, 2020 at 7:24 am #

Perhaps use less data?
Perhaps fit on a faster machine?

Revathi October 15, 2020 at 12:42 am #

Type diabetes dataset in below link
https://datasetsearch.research.google.com/

bakyalakshmi September 27, 2017 at 2:56 pm #

REPLY

please tell me about how to impute median using one dataset

Trung Nguyen Thanh July 8, 2018 at 8:42 p

please tell me, in case use Fancy impute

Jozo Kovac April 1, 2017 at 8:06 am #

Thanks for pointing on interesting problem. I v
categorical attributes in Python.

And dear reader, please never ever remove rows with missing values. It changes the distribution of your data and your analyses may become worthless. Learn from mistakes of others and don't repeat them 🙂

Jason Brownlee April 2, 2017 at 6:22 am #

REPLY

Thanks Jozo.

This post will help with categorical input data:
http://machinelearningmastery.com/data-preparation-gradient-boosting-xgboost-python/

Tommy Carstensen April 4, 2017 at 3:56 am #

REPLY

Super duper! Thanks for writing! Would it have been worth mentioning interpolate of Pandas?
http://pandas.pydata.org/pandas-docs/stable/generated/pandas.Series.interpolate.html

Jason Brownlee April 4, 2017 at 9:18 am #

REPLY

Thanks Tommy.

Aswathy April 14, 2017 at 12:10 pm #

REPLY

Hi Jason,

I was just wondering if there is a way to use a different imputation strategy for each column. Say, for a categorical feature you want to impute using the mode but for a continuous attribute, you want to impute using mean.

Jason Brownlee April 15, 2017 at 9:30 am #

REPLY

Yes, try lots of techniques, go with whatev

Salu Khadka June 11, 2017 at 12:29 am #

thanks for your tutorial sir.
I would also seek help from you for multi label classific

For example, categorizing a twitter post as related to s

Jason Brownlee June 11, 2017 at 8:26 am #

REPLY

Sure, see this post:
http://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/

Ali Gabriel Lara June 13, 2017 at 4:51 am #

REPLY

Hello Mr. Brownlee. Thank you so much for your post.

Do you know any approach to recognize the pattern of missing data? I mean, I am interested in discovering the pattern of missing data on a time series data.

The database is historical data of a chemical process. I think I should apply some pattern recognition approach columnwise because each column represents a process variable and the value coming from a transmisor.

My goal is to predict if the missing data is for a mechanical fault or a desviation in registration process or for any other causes. Then I should apply a kind of filling methods if it is required.

Have you any advice? Thanks in advance

Jason Brownlee June 13, 2017 at 8:25 am #

REPLY

I would invert the problem and model the series of missing data and mark all data you do have with a special value “0” and all missing instances as “1”.

Great problem!

Let me know how you go.

Patricia Villa October 5, 2017 at 3:45 pm #

REPLY

You helped me keep my sanity. THANK YOU!

Jason Brownlee October 5, 2017 at 5:23 pm

I’m really glad to hear that Patricia!

Sachin Raj October 6, 2017 at 7:58 pm #

How to know whether to apply mean or to replace it with mode?

Jason Brownlee October 7, 2017 at 5:54 am #

REPLY

Try both and see what results in the most skillful models.

Naga May 24, 2018 at 10:35 pm #

REPLY

Hi Sachin,
Mode is effected by outliers whereas Mean is less effected by outliers.
Please correct me if i am wrong@Jason

Jason Brownlee May 25, 2018 at 9:26 am #

REPLY

I think you meant “Median” is not affected by outliers. “Mode” is just the most common value.

Patrick October 26, 2017 at 5:06 am #

REPLY

If I have a 11×11 table and there are 20 missing values in there, is there a way for me to make a code that creates a list after identifying these values?

Let us say that the first column got names and the first row has Day 1 to 10. Some of the names does not show up all of the days and therefore there are missing gaps. I put this table into the code and rather than reading the table I get a list with:

Name, day 2, day 5, day 7
Name, Day 1, day 6

I understand that this could take some time to answer, and maybe know of good place to start on how to start

Jason Brownlee October 26, 2017 at 5:35 am

Sure, if the missing values are marked wi
values using Pandas.

Nivetha December 22, 2017 at 7:46 pm #

REPLY

can we code our own algorithms to impute the missing values????
if it is possible then how can i implement it??

Jason Brownlee December 23, 2017 at 5:15 am #

REPLY

Yes.

You can write some if-statements and fill in the n/a values in the Pandas dataframe.

I would recommend using statistics or a model as well and compare results.

Amit December 29, 2017 at 5:33 pm #

REPLY

Hi Jason,

I am trying to prepare data for the TITANIC dataset. One of the columns is CABIN which has values like ‘A22′,’B56’ and so on. This column has maximum number of missing values. First I thought to delete this column but I think this could be an important variable f

I am trying to find a strategy to fill these null values. Is there a way to fill alphanumeric blank values?

If there is no automatic way, I was thinking of fill these records based on Name, number of sibling, parent child and class columns. E.g. for a missing value, try to see if there are any relatives and use their cabin number to replace missing value.

Similar case is for AGE column which is missing. Any thoughts?

Jason Brownlee December 30, 2017 at 5:19 am #

REPLY

Sounds like a categorical variable. You could encode them as integers. You could also assign an “unknown” integer value (e.g. -999) for the miss

Perhaps you can develop a model to predict the ca .

Chidoooo February 9, 2018 at 10:15 pm #

Good day, I ran this file code
pd.read_csv(r’C:\Users\Public\Documents\SP_dow_Hi
and it gave me missing values (NAN) of return of stock

pd.read_csv(r’C:\Users\Public\Documents\SP_dow_Hi
Out[5]:
Unnamed: 0 S&P500 Dow Jones
0 Date close Close
1 1-Jan-17 2,275.12 24719.22
2 1-Jan-16 1,918.60 19762.60
3 1-Jan-15 2,028.18 17425.03
4 1-Jan-14 1,822.36 17823.07
5 1-Jan-13 1,480.40 16576.66
6 1-Jan-12 1,300.58 13104.14
7 1-Jan-11 1,282.62 12217.56
8 1-Jan-10 1,123.58 11577.51
9 1-Jan-09 865.58 10428.05
10 1-Jan-08 1,378.76 8776.39
11 1-Jan-07 1,424.16 13264.82
12 1-Jan-06 1,278.73 12463.15
13 1-Jan-05 1,181.41 10717.50
14 1-Jan-04 1,132.52 10783.01
15 1-Jan-03 895.84 10453.92
16 1-Jan-02 1,140.21 8341.63
17 1-Jan-01 1,335.63 10021.57
18 1-Jan-00 1,425.59 10787.99
19 1-Jan-99 1,248.77 11497.12
20 1-Jan-98 963.36 9181.43

21 1-Jan-97 766.22 7908.25
22 1-Jan-96 614.42 6448.27
23 1-Jan-95 465.25 5117.12
24 1-Jan-94 472.99 3834.44
25 1-Jan-93 435.23 3754.09
26 1-Jan-92 416.08 3301.11
27 1-Jan-91 325.49 3168.83
28 1-Jan-90 339.97 2633.66
29 1-Jan-89 285.4 2753.20
.. … … …
68 1-Jan-50 16.88 235.42
69 1-Jan-49 15.36 200.52
70 1-Jan-48 14.83 177.30
71 1-Jan-47 15.21 181.16
72 1-Jan-46 18.02 177.20
73 1-Jan-45 13.49 192.91
74 1-Jan-44 11.85 151.93
75 1-Jan-43 10.09 135.89
76 1-Jan-42 8.93 119.40
77 1-Jan-41 10.55 110.96
78 1-Jan-40 12.3 131.13
79 1-Jan-39 12.5 149.99
80 1-Jan-38 11.31 154.36
81 1-Jan-37 17.59 120.85
82 1-Jan-36 13.76 179.90
83 1-Jan-35 9.26 144.13
84 1-Jan-34 10.54 104.04
85 1-Jan-33 7.09 98.67
86 1-Jan-32 8.3 60.26
87 1-Jan-31 15.98 77.90
88 1-Jan-30 21.71 164.58
89 1-Jan-29 24.86 248.48
90 1-Jan-28 17.53 300.00
91 1-Jan-27 13.4 200.70
92 1-Jan-26 12.65 157.20
93 1-Jan-25 10.58 156.66
94 1-Jan-24 8.83 120.51
95 1-Jan-23 8.9 95.52
96 1-Jan-22 7.3 98.17
97 1-Jan-21 7.11 80.80

[98 rows x 3 columns]

pd.read_csv(r'C:\Users\Public\Documents\SP_dow_Hist_stock.csv',sep=',').pct_change(251)
Out[7]:
Unnamed: 0 S&P500 Dow Jones
0 NaN NaN NaN

Start Machine Learning

```
1 NaN NaN NaN
2 NaN NaN NaN
3 NaN NaN NaN
4 NaN NaN NaN
5 NaN NaN NaN
6 NaN NaN NaN
7 NaN NaN NaN
8 NaN NaN NaN
9 NaN NaN NaN
10 NaN NaN NaN
11 NaN NaN NaN
12 NaN NaN NaN
13 NaN NaN NaN
14 NaN NaN NaN
15 NaN NaN NaN
16 NaN NaN NaN
17 NaN NaN NaN
18 NaN NaN NaN
19 NaN NaN NaN
20 NaN NaN NaN
21 NaN NaN NaN
22 NaN NaN NaN
23 NaN NaN NaN
24 NaN NaN NaN
25 NaN NaN NaN
26 NaN NaN NaN
27 NaN NaN NaN
28 NaN NaN NaN
29 NaN NaN NaN
.. ... ... ...
68 NaN NaN NaN
69 NaN NaN NaN
70 NaN NaN NaN
71 NaN NaN NaN
72 NaN NaN NaN
73 NaN NaN NaN
74 NaN NaN NaN
75 NaN NaN NaN
76 NaN NaN NaN
77 NaN NaN NaN
78 NaN NaN NaN
79 NaN NaN NaN
80 NaN NaN NaN
81 NaN NaN NaN
82 NaN NaN NaN
83 NaN NaN NaN
```

```
84 NaN NaN NaN
85 NaN NaN NaN
86 NaN NaN NaN
87 NaN NaN NaN
88 NaN NaN NaN
89 NaN NaN NaN
90 NaN NaN NaN
91 NaN NaN NaN
92 NaN NaN NaN
93 NaN NaN NaN
94 NaN NaN NaN
95 NaN NaN NaN
96 NaN NaN NaN
97 NaN NaN NaN

[98 rows x 3 columns]
```


Jason Brownlee February 10, 2018 at 8:57 a

Perhaps post your code and issue to stac

Ravi March 13, 2018 at 10:59 pm #

REPLY

Hi Jason,

Thanks for your valuable writing.
I have one question :-
We can also replace NaN values with Pandas fillna() function. In my opinion this is more versatile than Imputer class because in a single statement we can take different strategies on different column.
df.fillna({'A':df['A'].mean(),'B':0,'C':df['C'].min(),'D':3})

What is your opinion? Is there any performance difference between two?

Jason Brownlee March 14, 2018 at 6:23 am #

REPLY

Great tip.

No idea, on the performance difference.

annusin0_0 March 26, 2018 at 4:31 am #

REPLY

Is there a recommended ratio on the number of NaN values to valid values , when any corrective action like imputing can be taken?
If we have a column with most of the values as null, then it would be better off to ignore that column altogether for feature?

Jason Brownlee March 26, 2018 at 10:04 am #

REPLY

No, it is problem specific. Perhaps run some experiments to see how sensitive the model is to missing values.

Ammar Hasan March 31, 2018 at 2:35 pm #

Hi Jason,

Thanks for this post, I wanted to ask, how do we imput
text labels or blanks.

Jason Brownlee April 1, 2018 at 5:44 am #

Good question, I'm not sure off hand. Perhaps start with simple masking of missing values.

rakend dubba May 27, 2018 at 6:05 pm #

REPLY

To fill the nan for a categorical column

df = df.fillna(df['column'].value_counts().index[0])
This fills the missing values in all columns with the most frequent categorical value

Gabriel April 6, 2018 at 9:44 pm #

REPLY

Thanks a lot Jason ! but I have a little question, how about if we want to replace missing values with the mean of each ROW not column ? how to do that ? if you have any clue, please tell me.. Thank you again Jason.

Jason Brownlee April 7, 2018 at 6:32 am #

REPLY

Why would you do this?

numpy.mean() allows you to specify the axis on which to calculate the mean. It will do it for you.

Adil April 13, 2018 at 8:14 am #

REPLY

Hi Jason,

I wanted to ask you how you would deal with missing timestamps (date-time values), which are one set of predictor variables in a classification problem. Would you flag and mark them as missing or impute them as the mode of the rest of the timestamps?

Jason Brownlee April 13, 2018 at 3:30 pm #

Here are some ideas:
https://machinelearningmastery.com/handle-missir

quan April 20, 2018 at 9:19 pm #

Hi Jason,

A big fan of yours.

I have a question about imputing missing numerical values. I don't really want to remove them and I want to impute them to a value that is like Nan but a numerical type? Would say coding it to -1 work? (0 is already being used).

I guess I am trying to achieve the same thing as categorising an nan category variable to unknown and creating another feature column to indicate that it is missing.

Thanks,

Jason Brownlee April 21, 2018 at 6:49 am #

REPLY

NaN is a numerical type. It is a valid float.

You could use -999 or whatever you like.

Be careful that your model can support them, or normalize values prior to modeling.

Ravi July 12, 2018 at 1:29 pm #

REPLY

Hello Jason,

You mentioned this here: “if you choose to impute with mean column values, these mean column values will need to be stored to file for later use on new data that has missing values.”, but I wanted to ask:

Would imputing the data before creating the training and test set (with the data set’s mean) cause data leakage? What would be the best approach to tackle missing data within the data pipeline for a machine learning project.

Let’s say I’m imputing by filling in with the mean. For the model tuning am I imputing values in the test set with the training set’s mean?

Jason Brownlee July 12, 2018 at 3:35 pm #

REPLY

Yes. You want to calculate the value to im

The sklearn library has an imputer you can use in
http://scikit-learn.org/stable/modules/generated/skl

Tobias August 8, 2018 at 7:36 pm #

Hi Jason,

Thanks again for that huge nice article!

is there a neat way to clean away all those rows that happen to be filled with text (i.e. strings) in a certain column, i.e. List.ImportantColumn .

This destroys my plotting with “could not convert string to float”

Thanks already in advance!

Jason Brownlee August 9, 2018 at 7:38 am #

REPLY

Yes, you can remove or replace those values with simple NumPy array indexing.

For example, if you have ‘?’ you can do:

```
1 X = X[X=='?'] = np.nan
```

Anantha Krishnan S September 15, 2018 at 5:45 pm #

REPLY

Hi Jason,

I tried using this dropna to delete the entire row that has missing values in my dataset and after which the isnull().sum() on the dataset also showed zero null valu
and i am getting an error.

Error : Input contains NaN, infinity or a value too large for dtype('float64')

This clearly shows there still exists some null values.

How do i proceed with this thanks in advance

Jason Brownlee September 16, 2018 at 5:57 am #

REPLY

Perhaps print the contents of the prepared data to confirm that the nans were indeed removed?

fatma October 27, 2018 at 2:23 am #

Hi Jason,

Thanks for this post, I'm using CNN for regression and
training samples. How can I use imputer to fill missing

Jason Brownlee October 27, 2018 at 6:03 am

Does the above tutorial not help?

fatma October 29, 2018 at 4:19 pm #

REPLY

should I apply Imputer function for both training and testing dataset?

Jason Brownlee October 30, 2018 at 5:52 am #

REPLY

Yes, but if the imputer has to learn/estimate, it should be developed from the training data and aplied to the train and test sets, in order to avoid data leakage.

fatma October 30, 2018 at 6:24 pm #

I feel that Imputer remove the Nan values and doesn't replace them. For example the vector features length in my case is 14 and there are 2 Nan values after applying Imputer function the vector length is 12. This means the 2 Nan values are removed. However I used the following setting:
imputer = Imputer(missing_values=np.

Jason Brownlee October 31, 2018 at 6:22 am #

I don't know what is happening in your case, perhaps post/search on stackoverflow?

fatma November 14, 2018 at 5:22 pm #

You mean I should fit it on training data then applied to the train and test sets as follow :

```
imputer = Imputer(strategy="mean", ax
imputer.fit(X_train)
X_train = imputer.transform(X_train)
X_test = imputer.transform(X_test)
```

Jason Brownlee November 15

Looks good.

Manik Chand October 28, 2018 at 8:58 pm #

REPLY

Thanks for this post!!!
A dataSet having more than 4000 rows and rows can be groupby their 1st columns and let there is many columns (assume 20 columns) and few columns(let 14 columns) contains NaN(missing value).
How we populate NaN with mean of their corresponding columns by iterative method(using groupby, transform and apply) .

Jason Brownlee October 29, 2018 at 5:56 am #

REPLY

Sorry, I don't understand. Perhaps you can elaborate your question?

Manik Chand October 30, 2018 at 3:19 am #

REPLY

actually i want to fill missing value in each column. Value is the mean of corresponding column. Is there any iterative method?

Jason Brownlee October 30, 2018 at 6:09 am #

REPLY

What do you mean by iterative method?

shailaja March 11, 2020 at 2:57 pm #

Is it iterative imputer? where missing value acts as dependent variable and independent variables are other features

Jason Brownlee March 12, 20

No.

Sumod December 27, 2018 at 2:47 pm #

After replacing zeroes,Can I save it as a new

Jason Brownlee December 28, 2018 at 5:50 am #

REPLY

Yes, call to_csv() on the dataframe.

CC February 5, 2019 at 7:18 am #

REPLY

what does this mean?

Jason Brownlee February 5, 2019 at 8:30 am #

REPLY

It is a function, learn more here:
https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.to_csv.html

Amit February 7, 2019 at 6:06 pm #

REPLY

```
import numpy as np
import pandas as pd

mydata = pd.read_csv('diabetes.csv',header=None)
mydata.head(20)
```

0 1 2 3 4 5 6 7 8
0 Pregnancies Glucose BloodPressure SkinThickness Insulin BMI DiabetesPedigreeFunction Age Outcome
1 6 148 72 35 0 33.6 0.627 50 1
2 1 85 66 29 0 26.6 0.351 31 0
3 8 183 64 0 0 23.3 0.672 32 1
4 1 89 66 23 94 28.1 0.167 21 0
5 0 137 40 35 168 43.1 2.288 33 1

print((mydata[0] == 0).sum()) — for any column it alwa
0 >>>>>>>…. any thing wrong here ?

whereas i have 0's in dataset

0 Pregnancies
1 6
2 1
3 8
4 1
5 0>>>>>>>>>
6 5
7 3
8 10
9 2
10 8
11 4
12 10
13 10
14 1
15 5
16 7
17 0 >>>>>>

Jason Brownlee February 8, 2019 at 7:45 am #

REPLY

Perhaps post your code and issue to stackoverflow?

Pauline May 14, 2020 at 8:49 pm #

REPLY

Hello,

Start Machine Learning

More than one year later, I have the same problem as you. When i search for 0 it does not work. However, when I look for '0' it does, which means the table is filled with strings and not number… Any idea how I can handle that?

Best Regards

REPLY

Jason Brownlee May 15, 2020 at 6:00 am #

Perhaps your data was loaded as strings?

Try converting it to numbers:

```
a = a.astype('float64')
```


Krishna March 24, 2019 at 3:59 am #

Hi sir,
For my data after executing following instructions still I
dataset= dataset.replace(0, np.NaN)
dataset.dropna(inplace=True)
dataset= dataset.replace(0, np.Inf)
dataset.dropna(inplace=True)
print(dataset.describe())
F1 F2 F3 F4
count 1200.000000 1200.000000 1200.000000 1200.000000
mean 0.653527 0.649447 1.751579 inf
std 0.196748 0.194933 0.279228 NaN
min 0.179076 0.179076 0.731698 0.499815
25% 0.507860 0.506533 1.573212 1.694007
50% 0.652066 0.630657 1.763520 1.925291
75% 0.787908 0.762665 1.934603 2.216663
max 1.339335 1.371362 2.650390 inf

How can I get out from this problem.

REPLY

Jason Brownlee March 24, 2019 at 7:07 am #

Sorry to hear that, perhaps try posting your code and question to stackoverflow?

REPLY

Raj January 9, 2020 at 11:27 pm #

df.replace(-np.Inf, 0)
df.replace(np.Inf, 0)

mubasher March 28, 2019 at 6:38 pm #

REPLY

how can we impute the categorical data in python

Jason Brownlee March 29, 2019 at 8:29 am #

REPLY

You can use an integer encoding (label er
embedding.

Rotenda Takalani April 16, 2019 at 9:42 pm #

Hi Jason

I am new to Python and I was working through the exa
piece of code to count the zeros, the code returns resu
columns.

Can you please assist?

Jason Brownlee April 17, 2019 at 7:00 am #

REPLY

Sorry to hear that, I have some suggestions here:
https://machinelearningmastery.com/faq/single-faq/why-does-the-code-in-the-tutorial-not-work-for-me

chu June 4, 2019 at 7:20 pm #

REPLY

Hi Jason,

Great post. Thanks so much.

Say I have a dataset without headers to identify the columns, how can I handle inconsistent data, for example, age having a value 2500 without knowing this column captures age, any thoughts?

Jason Brownlee June 5, 2019 at 8:36 am #

REPLY

You can use statistics to identify outliers:
https://machinelearningmastery.com/how-to-use-statistics-to-identify-outliers-in-data/

Muhammad Irfan November 26, 2019 at 7:02 am #

REPLY

Hi Jason,

Nice article. How can we add (python) another feature indicating a missing value as 1 if available and 0 if not? Is that a sensible solution?

Thank you.

Jason Brownlee November 26, 2019 at 1:28

Great question.

You could loop over all rows and mark 0 and 1 valu
original feature/rows.

Hadis December 11, 2019 at 1:44 pm #

Pima Indians Diabetes Dataset doesn't exist anymore 😕

Jason Brownlee December 11, 2019 at 2:46 pm #

REPLY

Thanks, I have updated the link.

Hussain December 12, 2019 at 9:26 pm #

REPLY

What is the current situation in AutoML field? What researchers try to bring out actually?

Jason Brownlee December 13, 2019 at 6:00 am #

REPLY

Good question, I need to learn more about that field.

Hussain December 13, 2019 at 9:21 pm #

Start Machine Learning

Let me know ,once you get to know about that someday.
Thank you for your response!!

Bill January 1, 2020 at 3:17 pm #

REPLY

[Ignore earlier misplaced post]

Jason,
Many thanks for your work in preparing these awesome tutorials!

In order to fill missing values with mean column values, I had to switch from:
from sklearn.preprocessing import Imputer
…
# fill missing values with mean column values
imputer = Imputer()

To:
from sklearn.impute import SimpleImputer
…
imputer = SimpleImputer(missing_values=numpy.NaN,

Jason Brownlee January 2, 2020 at 6:38 am #

Thanks for sharing.

German Niebles January 5, 2020 at 4:00 am #

REPLY

Jason, thanks a lot for your article,very useful.
Regards

Jason Brownlee January 5, 2020 at 7:07 am #

REPLY

You're welcome.

Raj January 30, 2020 at 6:12 pm #

REPLY

Hi Jason, great tutorial! If I were to impute values for time series data, how would I need to approach it? My dataset has data for a year and data is missing for about 3 months. Is there any way to salvage this time series for forecasting?

Jason Brownlee January 31, 2020 at 7:42 am #

REPLY

See this:
https://machinelearningmastery.com/handle-missing-timesteps-sequence-prediction-problems-python/

Raj January 31, 2020 at 7:09 pm #

REPLY

Thanks Jason!

Jason Brownlee February 1, 2020

You're welcome.

Shreya February 29, 2020 at 3:37 am #

I have near about 4 lakhs of data.
The shape of my dataset is (400000,114).
I want to first impute the data and then apply feature selection such as RFE so that I could train my model with only the important features further instead of all 114 features.
But I am unable to understand how after using SimpleImputer and MinMax scaler to normalize the data as :

```
values = dataset.values
imputer = SimpleImputer()
imputedData = imputer.fit_transform(values)
scaler = MinMaxScaler(feature_range=(0, 1))
normalizedData = scaler.fit_transform(imputedData)
```

How will we use this normalized data ??
Because on normal dataset further I am making X,Y labels as:

```
X = dataset.drop(['target'], axis=1)
y = dataset.target
```

How RFE will be used here further ?
Whether on X and y labels or before that do we have to convert all X labels to normalized data ?

Shreya February 29, 2020 at 3:41 am #

REPLY

Also training this huge amount of data with Random Forest or Logistic Regression for RFE is taking much of time ? So is a better solution available for training ?

Jason Brownlee February 29, 2020 at 7:20 am #

REPLY

Perhaps use a smaller sample of your data to start with.

Shreya February 29, 2020 at 3:07 pm #

REPLY

I have tried it with smaller set of d
But in a requirement I have to use this larg
Also RFE on RandomForest is taking a hu
And if I go with model = LogisticRegressio
dealing with warnings which I am unable t

ConvergenceWarning: The max_iter was r
"the coef_ did not converge", Convergence

How should I go further for feature selectic

Thank you very much !!

Jason Brownlee March 1, 2020 at 5:22 am #

Perhaps fit on less data, at least initially.

Jason Brownlee February 29, 2020 at 7:20 am #

REPLY

I would recommend developing a pipeline so that the imputation can be applied prior to scaling and feature selection and the prior to any modeling.

manjunath February 29, 2020 at 6:00 am #

REPLY

Hi Jason I have Time Series Data so i need to fill missing values , so which is best technique to fill time series data ?

Jason Brownlee February 29, 2020 at 7:22 a

See this tutorial:
https://machinelearningmastery.com/handle-missing-timesteps-sequence-prediction-problems-python/

Bruno Campetella April 4, 2020 at 2:05 am #

REPLY

Hello Jason
If we impute a column in our dataset the data distribution will change, and the change will depend on the imputation strategy. This in turns will affect the different ML algorithms performance. We are tuning the prediction not for our original problem but for the "new" dataset, which most probably differ from the real one.
My question is, for avoiding error predictions or overes
avoid having any NA's imputed values in our test datas
and using different imputation techniques to check per
imputed NA's).
Thanks
Bruno

Jason Brownlee April 4, 2020 at 6:25 am #

Yes. Great question!

Test a few strategies and use the approach that results in a model that has the best skill.

Deeksha Mahapatra May 11, 2020 at 3:42 am #

REPLY

Hi Jason,
First of all great job on the tutorials! This is my go to place for Machinel earning now.

I am trying to impute values in my dataset conditionally. Say I have three columns, If Column 1 is 1 then Column 2 is 0 and Column 3 is 0; If column 1 is 2 then Column 2 is Mean () and Column 3 is Mean(). I tried running an if statement with the function any() and defined the conditions separately. However the conditions are not being fulfilled based on conditions, I am either getting all mean values or all zeroes.
I have posted this on Stackoverflow and haven't gotten any response to help me with this.Please do suggest what should I apply to get this sorted.

Thanks a lot!

Jason Brownlee May 11, 2020 at 6:08 am #

REPLY

Thanks!

Perhaps try writing the conditions explicitly and enumerate the data, rather than using numpy tricks? It will be slower but perhaps easier to debug.

Deeksha Mahapatra May 11, 2020 at 4:43 pm #

REPLY

Yes, I used iloc to define the conditions separately. Worked fine. Thanks!

Jason Brownlee May 12, 2020 at 6:37 am #

REPLY

Great!

Parthiv June 14, 2020 at 8:27 am #

Thanks for the post.

Please correct me if i am wrong.

Applying these techniques for training data works for m
received with standard inverval (100 milliseconds), the
(scikit) with inherent capabilities can be used.

Your Weka post on missing values by defining threshold works great.

Jason Brownlee June 15, 2020 at 5:58 am #

REPLY

100ms is a long time for a computer, I don't see the problem with using imputation.

Parthiv June 19, 2020 at 6:58 am #

REPLY

Thanks for the reply.

Just a clarification. If one instance of data from several sensors arrive with some missing values for every 100ms, is it possible to classify based on the current instance alone. (one instance at a time).

My presumption is that we need multiple instances to calculate the statistics even for stream data.

Sorry. A bit confused on this.

Jason Brownlee June 19, 2020 at

Generally, you can frame the prediction problem any way you wish, e.g. based on the data you have and the data you need at prediction time.

Then train a model based on that framing of the problem.

Parthiv June 19, 2020 at 3:27 pm #

Thanks a lot replying with patience.

Jason Brownlee June 20, 202

I'm here to help if I can.

Anthony The Koala July 5, 2020 at 10:31 pm #

Dear Dr Jason,
Background information and question:

Background information:
Instead of playing around with the "horse colic" data wi[illegible] iris data. I had to shuffle the data to get an even spread of species 0, 1 or 2. Otherwise if I took the first 20 rows the last column would be full of species 0. Hence my shuffling of the data.

I've had great success in predicting the kind of species.

So my iris20 data looks like this – the first four columns are in the correct order of the original iris data and the last column are a variety of species. .

```
array([[7.2, 3. , 5.8, 1.6, 2. ],
       [6.3, 2.5, 5. , 1.9, 2. ],
       [5.7, 2.9, 4.2, 1.3, 1. ],
       [6.3, 2.3, 4.4, 1.3, 1. ],
       [5. , 3. , 1.6, 0.2, 0. ],
       [6.7, 3.1, 4.7, 1.5, 1. ],
       [6.5, 3.2, 5.1, 2. , 2. ],
       [5.7, 2.8, 4.5, 1.3, 1. ],
       [6.4, 3.2, 4.5, 1.5, 1. ],
       [6.3, 2.8, 5.1, 1.5, 2. ],
       [7.6, 3. , 6.6, 2.1, 2. ],
       [5.8, 2.7, 5.1, 1.9, 2. ],
       [6.3, 3.3, 6. , 2.5, 2. ],
       [5.5, 2.4, 3.7, 1. , 1. ],
       [6.7, 3. , 5. , 1.7, 1. ],
       [5. , 3.4, 1.5, 0.2, 0. ],
       [5.4, 3.4, 1.5, 0.4, 0. ],
       [5.7, 3. , 4.2, 1.2, 1. ],
       [6.3, 3.3, 4.7, 1.6, 1. ],
       [4.6, 3.4, 1.4, 0.3, 0. ]])
```

I removed 10 values 'at random' from my iris20 data, c[illegible]

```
array([[7.2, 3. , 5.8, 1.6, 2. ],
       [6.3, 2.5, 5. , nan, 2. ],
       [5.7, nan, 4.2, 1.3, 1. ],
       [nan, 2.3, 4.4, 1.3, 1. ],
       [5. , 3. , nan, 0.2, 0. ],
       [6.7, 3.1, 4.7, 1.5, 1. ],
       [6.5, 3.2, 5.1, nan, 2. ],
       [5.7, 2.8, 4.5, 1.3, 1. ],
       [6.4, 3.2, 4.5, 1.5, 1. ],
       [6.3, 2.8, 5.1, 1.5, 2. ],
       [7.6, 3. , 6.6, 2.1, 2. ],
       [5.8, 2.7, nan, 1.9, 2. ],
       [nan, 3.3, 6. , 2.5, 2. ],
       [5.5, 2.4, 3.7, 1. , 1. ],
       [6.7, 3. , 5. , 1.7, 1. ],
       [5. , 3.4, 1.5, 0.2, 0. ],
       [5.4, nan, 1.5, nan, 0. ],
       [5.7, 3. , 4.2, 1.2, 1. ],
       [6.3, 3.3, nan, 1.6, 1. ],
       [4.6, 3.4, 1.4, 0.3, 0. ]])
```

Question:
I have successfully been able to predict the kind of spe
Examples:

```
row = [7.6  ,8  ,6.6    ,2.1];  #predicts c
row = [7.6  ,NaN    ,6.6    ,2.1]; # correct
row = [7.6  ,33 ,6.6    ,2.1]; #correctly p
row = [6.3  ,2.3    ,4.4    ,1.3]; #correct
row = [6.3  ,NaN,4.4    ,1.3]; #correctly p
```

My question: In listing 8.19, 3rd last line, page 84 (101

```
yhat = pipeline.predict([row])
```

row is enclosed in brackets [row].
that is we have for example row = [[6.3 ,NaN,4.4 ,1.3]]
Why please do we double enclose the array in predict function?

When I do this

```
yhat = pipeline.predict(row); # I get errors
```

I get errors.

Thank you for your time,
Anthony of Sydney

Why enclose row as [row] since row is already enclosed by brackets.
That is why .predict([row]) and not .predict(row)

Jason Brownlee July 6, 2020 at 6:35 am #

REPLY

Nice work!

The predict() function expects a 2d matrix input, one row of data represented as a matrix is [[a,b,c]] in python.

Anthony The Koala July 6, 2020 at 9:35 am #

REPLY

First,
Thanks in advance for your reply. It is appreciated.

A question on your answer please.

Background info

```
row
[6.3, nan, 4.4, 1.3]
np.shape(row)
(4,)
np.shape([row])
(1, 4)
```

In the above example we had to structure the v
function. Here 'row' is changed from an array c

I've worked out that one can construct an n x r
matrix

To illustrate:

Recall in my above example I made a series o
with these rows:

```
row = [7.6  ,8  ,6.6    ,2.1];  #predicts correctly 2.
row = [7.6  ,NaN    ,6.6    ,2.1]; # correctly predicts 2.
row = [7.6  ,33 ,6.6    ,2.1]; #correctly predicts 2.
row = [6.3  ,2.3    ,4.4    ,1.3]; #correctly predicts 1
row = [6.3  ,NaN,4.4    ,1.3]; #correctly predicts 1
```

Now if we made an n x m matrix and feed that n x m matrix into the predict() function we should expect the same outcomes as individual predictions.

```
composite_matrix = [[7.6    ,8  ,6.6    ,2.1],[7.6  ,NaN    ,6.6    ,2.1],[7.6  ,33
yhat = pipeline.predict(composite_matrix)
yhat
array([2., 2., 2., 1., 1.])
```

Result is the same as if making individual predictions. Hence I understand the predict() function expecting a matrix and if predicting for single rows, make the single row into a 1xm matrix.

Conclusion: the predict() function expects a matrix, and we can make an n x m matrix containing the rows of what we want to predict AND get multiple results.

Thank you again in advance
Anthony of Sydney

Jason Brownlee July 6, 2020 at 2:06 pm #

REPLY

Yes.

Perhaps this will help clarify:
https://machinelearningmastery.com/make-predictions-scikit-learn/

Anthony The Koala July 6, 2020 at 7:29 pm #

Dear Dr Jason,
Thank you for the blog at https://machinelearningmastery.com/make-predictions-scikit-learn/.

Relevant to answer my question about
"Single Class Predictions" and "Multipl
1, 3 and 2).

The variable Xnew is of the structure [

In the multiple class predictions, Xnew

In both cases of single or multiple clas

```
1 ynew = model.predict(Xnew)
```

In sum predicting requires our feature
are the number of predictions and m b

Thank you,
Anthony of Sydney

Anthony The Koala July 14, 2020 at 10:02 pm #

REPLY

Dear Dr Jason,
I wish to share my two ways of deleting specific rows from a dataset as per subheading 4, "Remove Rows With Missing Values"

HOW TO DELETE SPECIFIC VALUES FROM SPECIFIC COLUMNS – TWO METHODS
Method #1 as per heading 4 = listing 7.16 on p73 (90 of 398) of your book.

```
dataset = read_csv('pima-indians-diabetes.csv', header=None)
# replace '0' values with 'nan'
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].replace(0, nan);#replace specific cols=0 with
# drop rows with missing values
dataset.dropna(inplace=True); # Delete all rows in the dataset with NaN
# split dataset into inputs and outputs
values = dataset.values
#For use in future modelling
X = values[:,0:8]
y = values[:,8]
```

Method #2 – using arrays

```
#How to delete specific values from specif
```

```
#We pretend that we don't load data in a DataFrame as in Method #1
#We wish to replace 0 with NaN in specific columns, this time 1,2,3,4,5 (1 is 2nd column)
#Then we wish to delete columns with NaN
from numpy import isnan, NaN, min, max, array

data = dataset.values; # dataset is a DataFrame containing large no of cols

specific_col = [1,2,3,4,5]
#replacing specific rows and columns whose value is 0 with NaN
for row in range(data.shape[0]):
        for col in range(len(specific_col)):
           if(data[row,specific_col[col]] == 0):
               data[row,specific_col[col]] = NaN

#Deleting columns with row,col values = NaN
new_data = [[]]
for row in range(data.shape[0]):
    temp_row = data[row]
    if not(isnan(min(temp_row)): #less time
        new_data.append(temp_row)
data = array(new_data)
#For use in modelling
X = data[:,0:8]
y = data[:,8]
```

The last method was presented in case your data set i

Thank you,
Anthony of Sydney

REPLY

Jason Brownlee July 15, 2020 at 8:17 am #

Thanks for sharing!

REPLY

Levente December 5, 2020 at 3:23 am #

Hi Jason,

I was just wondering if data imputing (e.g. replacing all missing values by the arithmetic mean of the corresponding column) in fact results in data leakage, implementing bias into the model during training? Such data imputing will, after all, fill up the dataset with information provided by instances (rows) that should be unseen by the model while training.

If that is indeed a problem, what would you recommend we do? Would it be better to add data imputing to the pipeline and thus, implement it separately for each fold of cross validation, together with other feature selection, preprocessing, and feature engineering steps?

Thanks a lot,
Levente

Jason Brownlee December 5, 2020 at 8:10 am #

REPLY

It doesn't as long as you only use the training data to calculate stats.

obby January 31, 2021 at 7:39 am #

REPLY

how can i do similar case imputation using mean for Age variable with missing values,

Jason Brownlee January 31, 2021 at 9:40 am

The mean is calculated as the sum of the

This tutorial will help you get started:
https://machinelearningmastery.com/statistical-imp

SULAIMAN KHAN February 8, 2021 at 1:05 am #

['toy stori',
'grumpier old men',
'heat',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',
'nan',

‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
‘nan’,
#################
Hi Jason , I applied embedding technique. how to handle nan values? i will improve my result.

Jason Brownlee February 8, 2021 at 7:03 am # REPLY

If you have nan values in your data you can try removing them, imputing them, masking them, etc.

If you have nan values out of your model, you’re model is broken, perhaps exploding gradients, or vanishing gradients during training.

SULAIMAN KHAN February 9, 2021 at 10:02 pm #

REPLY

precision recall f1-score support

class0(0.5) 0.00 0.00 0.00 0
class1(1) 0.00 0.00 0.00 8
class2(1.5) 0.00 0.00 0.00 2
class3(2) 0.00 0.00 0.00 10
class4(2.5) 0.02 0.22 0.03 9
class5(3) 0.00 0.00 0.00 75
class6(3.5) 0.00 0.00 0.00 16
class7(4) 0.00 0.00 0.00 74
class8(4.5) 0.00 0.00 0.00 17
class9(5) 0.00 0.00 0.00 35

accuracy 0.01 246
macro avg 0.00 0.02 0.00 246
weighted avg 0.00 0.01 0.00 246

[[0 0 0 0 0 0 0 0 0 0]
[1 0 0 0 7 0 0 0 0 0]
[1 0 0 0 0 0 1 0 0 0]
[1 2 0 0 5 0 2 0 0 0]
[5 2 0 0 2 0 0 0 0 0]
[7 21 0 0 40 0 7 0 0 0]
[2 7 0 0 7 0 0 0 0 0]
[13 32 0 0 28 0 1 0 0 0]
[1 8 0 0 7 0 1 0 0 0]
[1 21 0 0 12 0 1 0 0 0]]
okay, I removed “nan” values. above my new result. I am waiting positive response.

Jason Brownlee February 10, 2021 at 8:08 am #

REPLY

Sorry, what problem are you having exactly? Perhaps you can rephrase or elaborate your question?

SULAIMAN KHAN February 10, 2021 at 3:49 pm #

REPLY

RangeIndex: 100836 entries, 0 to 100835
Data columns (total 6 columns):
Column Non-Null Count Dtype
— —— ——— ——
0 userId 100836 non-null int64
1 movieId 100836 non-null int64

2 timestamp 100836 non-null int64
3 title 745 non-null object
4 genres 745 non-null object
5 rating 100836 non-null float64
dtypes: float64(1), int64(3), object(2)
memory usage: 4.6+ MB
##################################
Hi Jason,
I removed all missing values in “title , genra” but my total sample observations 745.why is it not improving? the column “title , genra” has text data. How to generate missing values in for text data?

Jason Brownlee February 11, 2021 at 5:48 a

Perhaps you can use the most common v
Perhaps you can use a special “no text” phrase?

Sofine Heilskov June 9, 2021 at 12:04 am #

Hi Jason
Thank you for your helpfull tutorials!

I have a “sensor saturation problem” in my dataset: censored measurements lie between zero and a lower measuring limit (A) or above an upper limit (B).

Can I apply the Scikit-learn IterativeImputer method to impute these values based on the AB? I basically want to add the extreme values (tales) to my normal distribution curve.

Extras:
Using Python 3.9.5, un-experienced user.
I have looked at the PyMC3 package (https://docs.pymc.io/notebooks/censored_data.html).
But the packages used in this example are not working well together
(https://discourse.pymc.io/t/attributeerror-module-arviz-has-no-attribute-geweke/6818)

Jason Brownlee June 9, 2021 at 5:44 am #

REPLY

Perhaps try it and see.

Murilo September 29, 2021 at 9:16 pm #

REPLY

Hello Jason,

If i have a full row of NaN, what is the commom practice to dealing with it? Should i just delete it from my dataset?

Do you have any reference i could read about this kind of problem? I have a dataset with 42k rows, and i have seen some of them are tottaly empty.

Thanks in advance,
Murilo

Adrian Tam September 30, 2021 at 1:30 am #

REPLY

If you get a row full of NaN, drop it. If you [illegible] imputation.

Leave a Reply

Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

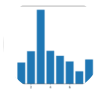
Welcome!
I'm *Jason Brownlee* PhD
and I **help developers** get results with **machine learning**.
Read more

Never miss a tutorial:

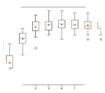
Picked for you:

How to Choose a Feature Selection Method For Machine Learning

Data Preparation for Machine Learning (7-Day Mini-Course)



How to Calculate Feature Importance With Python



Recursive Feature Elimination (RFE) for Feature Se

How to Remove Outliers for Machine Learning

Loving the Tutorials?

The Data Preparation EBook is
where you'll find the ***Really Good*** stuff.

>> SEE WHAT'S INSIDE

Start Machine Learning