



# Weight of Evidence

# Weight of evidence: definition

$$WoE = \ln\left(\frac{\textit{Proportion of good events}}{\textit{Proportion of bad events}}\right)$$

Weight of Evidence (WoE) was developed primarily for the credit and financial industries to help build more predictive models to evaluate the risk of loan default.

That is, to predict how likely the money lent to a person or institution is to be lost.

# Weight of evidence: definition

$$WoE = \ln\left(\frac{\textit{Proportion of good events}}{\textit{Proportion of bad events}}\right)$$

- Proportion of good events:
  - sum of + observations per category / total positive observations
- Proportion of bad events:
  - sum of - observations per category / total negative observations

# Weight of evidence: example

	survived	non-survived
A	50	30
B	75	40
C	25	15
total	150	85
total passengers		235



% good	% bad
0.33	0.35
0.50	0.47
0.17	0.18



WoE
-0.05716
0.060625
-0.05716



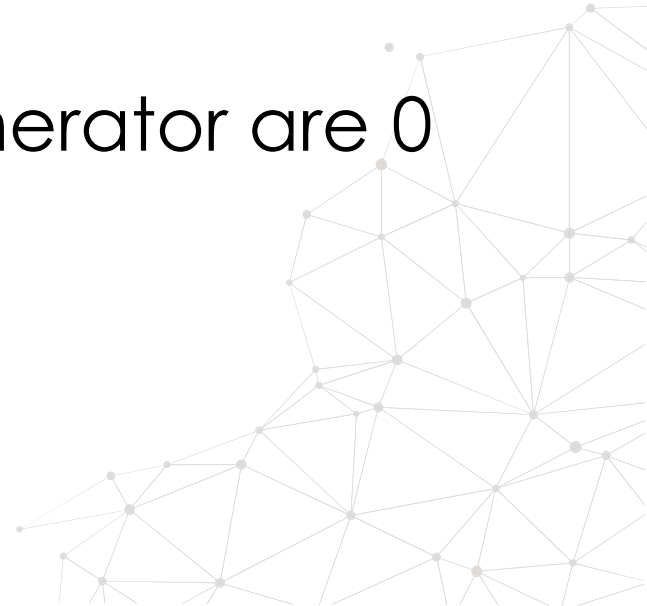
# Weight of evidence: Advantages

- Creates a monotonic relationship between the target and the variables.
- It orders the categories on a "logistic" scale which is natural for logistic regression
- The transformed variables can then be compared because they are on the same scale.
  - Therefore, it is possible to determine which one is more predictive.

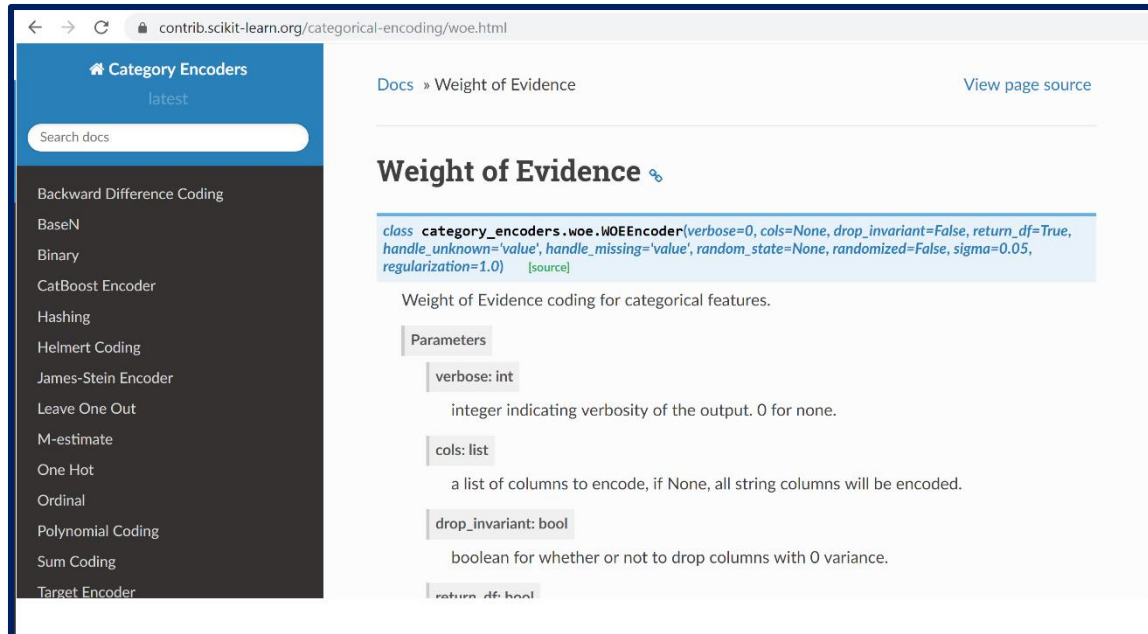


# Weight of evidence: Limitations

- May lead to over-fitting
- Not defined when the denominator or numerator are 0



# Weight of evidence: open source



The screenshot shows the Scikit-Learn website's documentation for the 'Weight of Evidence' (WOE) encoding. The page title is 'Weight of Evidence' and it is part of the 'Category Encoders' section. The page includes a search bar, a table of contents, and a description of the WOE encoding. The main content area shows the class `category_encoders.woe.WOEEncoder` with its parameters and a brief description of the encoding process.

Category Encoders  
latest

Search docs

Backward Difference Coding  
BaseN  
Binary  
CatBoost Encoder  
Hashing  
Helmert Coding  
James-Stein Encoder  
Leave One Out  
M-estimate  
One Hot  
Ordinal  
Polynomial Coding  
Sum Coding  
Target Encoder

Docs » Weight of Evidence [View page source](#)

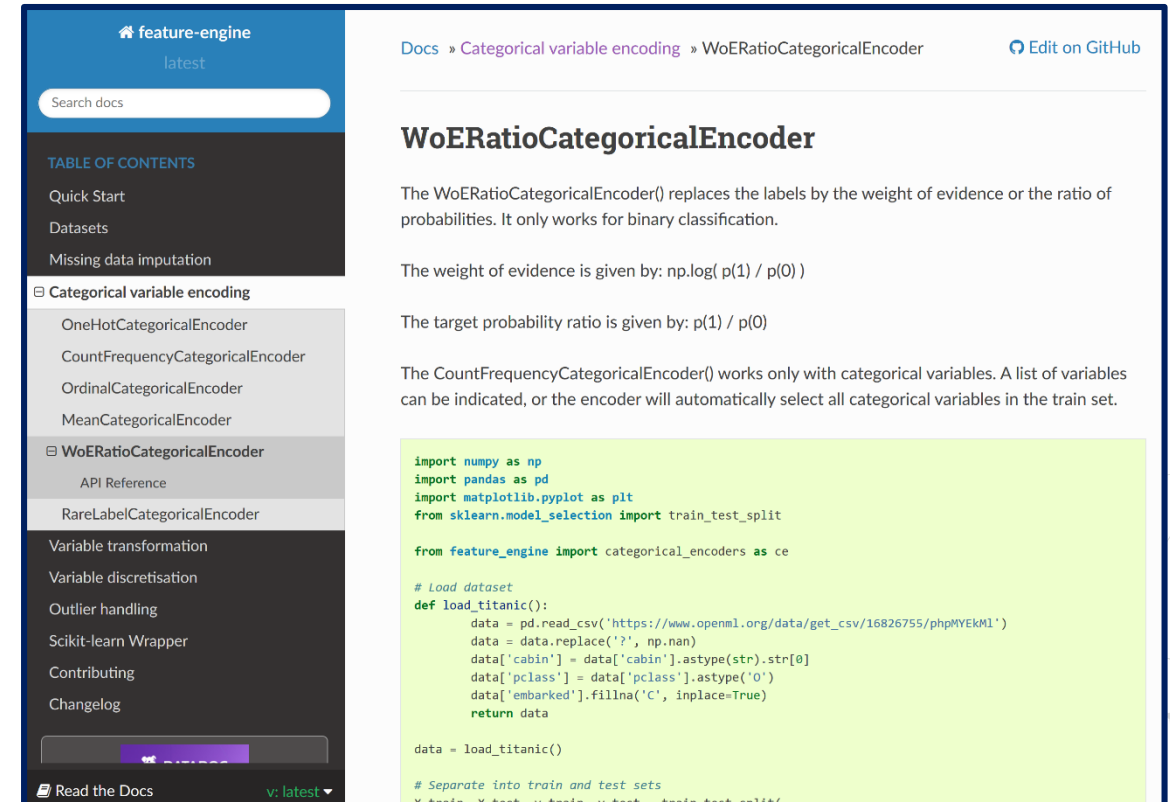
## Weight of Evidence 🔗

```
class category_encoders.woe.WOEEncoder(verbose=0, cols=None, drop_invariant=False, return_df=True,
handle_unknown='value', handle_missing='value', random_state=None, randomized=False, sigma=0.05,
regularization=1.0) [source]
```

Weight of Evidence coding for categorical features.

Parameters

- verbose:** int  
integer indicating verbosity of the output. 0 for none.
- cols:** list  
a list of columns to encode, if None, all string columns will be encoded.
- drop\_invariant:** bool  
boolean for whether or not to drop columns with 0 variance.
- return\_df:** bool



The screenshot shows the feature-engine website's documentation for the 'WoERatioCategoricalEncoder'. The page title is 'WoERatioCategoricalEncoder' and it is part of the 'Categorical variable encoding' section. The page includes a search bar, a table of contents, and a description of the encoder. The main content area shows the class `WoERatioCategoricalEncoder` with its parameters and a brief description of the encoding process. A code snippet is provided to demonstrate how to use the encoder.

feature-engine  
latest

Search docs

TABLE OF CONTENTS

- Quick Start
- Datasets
- Missing data imputation
- Categorical variable encoding
  - OneHotCategoricalEncoder
  - CountFrequencyCategoricalEncoder
  - OrdinalCategoricalEncoder
  - MeanCategoricalEncoder
  - WoERatioCategoricalEncoder**
  - RareLabelCategoricalEncoder
- Variable transformation
- Variable discretisation
- Outlier handling
- Scikit-learn Wrapper
- Contributing
- Changelog

[Read the Docs](#) [v: latest](#)

## WoERatioCategoricalEncoder

The `WoERatioCategoricalEncoder()` replaces the labels by the weight of evidence or the ratio of probabilities. It only works for binary classification.

The weight of evidence is given by:  $\text{np.log}(p(1) / p(0))$

The target probability ratio is given by:  $p(1) / p(0)$

The `CountFrequencyCategoricalEncoder()` works only with categorical variables. A list of variables can be indicated, or the encoder will automatically select all categorical variables in the train set.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

from feature_engine import categorical_encoders as ce

# Load dataset
def load_titanic():
    data = pd.read_csv('https://www.openml.org/data/get_csv/16826755/phpMYEkM1')
    data = data.replace('?', np.nan)
    data['cabin'] = data['cabin'].astype(str).str[0]
    data['pclass'] = data['pclass'].astype('O')
    data['embarked'].fillna('C', inplace=True)
    return data

data = load_titanic()

# Separate into train and test sets
```

# THANK YOU

[www.trainindata.com](http://www.trainindata.com)