Using Target Encoding to Improve Model Predictions

Jo-fai (Joe) Chow
Data Science Evangelist, H2O.ai
joe@h2o.ai / @matlabulous

Download: bit.ly/h2o_meetups

H2O.ai Overview

Company	Founded in Silicon Valley in 2012 Funded: $75M Investors: Wells Fargo, NVIDIA, Nexus Ventures, Paxion Ventures
Products	• H2O Open Source Machine Learning (18,000 organizations) • H2O Driverless AI – Automatic Machine Learning
Team	130 AI experts (Expert data scientists, Kaggle Grandmasters, Distributed Computing, Visualization)
Global	Mountain View, NYC, London, Prague, India

H2O.ai Product Suite

H2O

In-memory, distributed machine learning algorithms with H2O Flow GUI

Spark + H2O SPARKLING WATER

H2O AI open source engine integration with Spark

H2O4GPU

Lightning fast machine learning on GPUs

Open Source

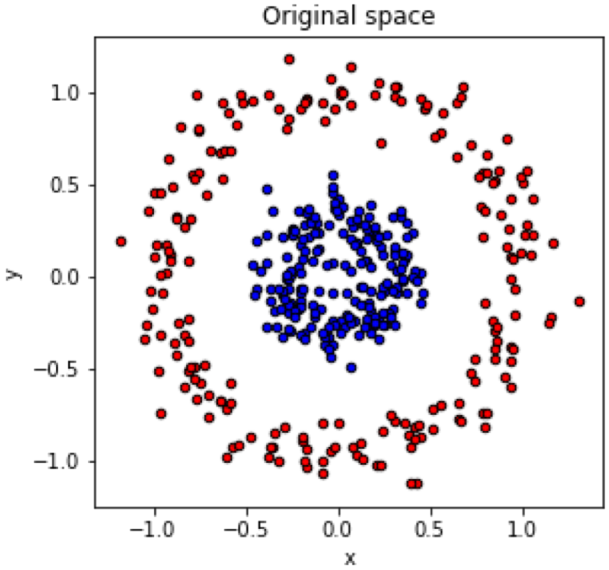
- 100% open source – Apache V2 licensed
- Built for data scientists – interface using R, Python on H2O Flow (interactive notebook interface)
- Enterprise support subscriptions

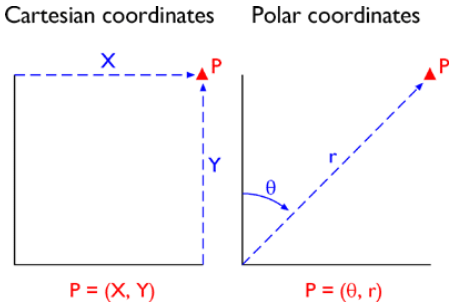
DRIVERLESSAI

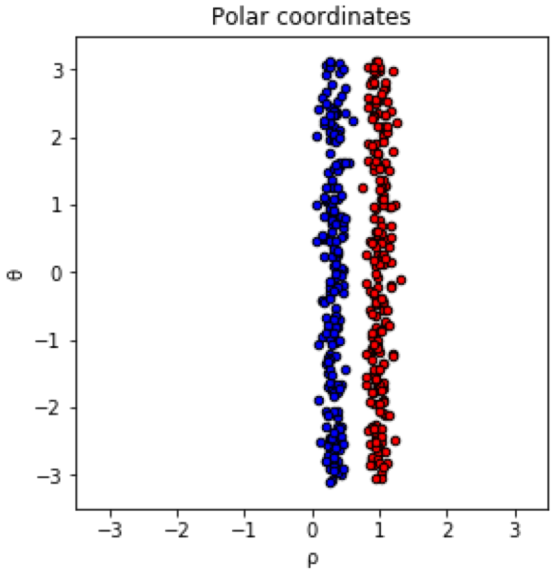
Automatic feature engineering, machine learning and interpretability

- Enterprise software
- Built for domain users, analysts and data scientists – GUI-based interface for end-to-end data science
- Fully automated machine learning from ingest to deployment
- User licenses on a per seat basis (annual subscription)

Why Feature Engineering?







$$\begin{bmatrix} \mathbf{X} \end{bmatrix} \dashrightarrow \begin{bmatrix} \mathbf{U} \end{bmatrix}$$

Feature Engineering: Target Mean Encoding

H2O.ai

What?

- Replace categorical variables with the mean of the response

Why?

- Categorical variables increase the number of features (dummy encoding) and can cause us to overfit

Caution!

- If it is applied without care, it may lead to overfitting

Target Mean Encoding

H2O.ai

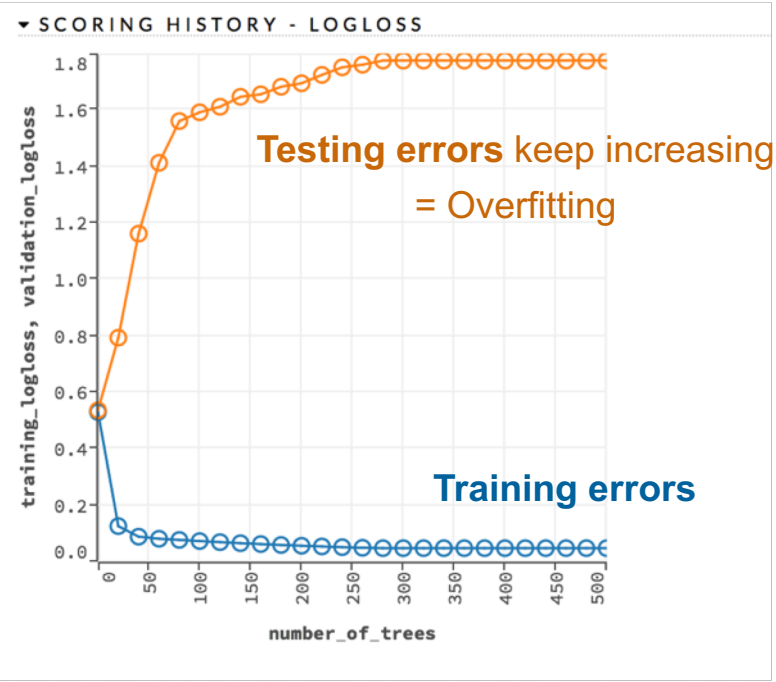
Categorical Feature	Target	Target Mean Encoding
A	1	
A	1	
A	0	
B	0	
B	1	
B	0	
C	1	

Target Mean Encoding Done Wrong

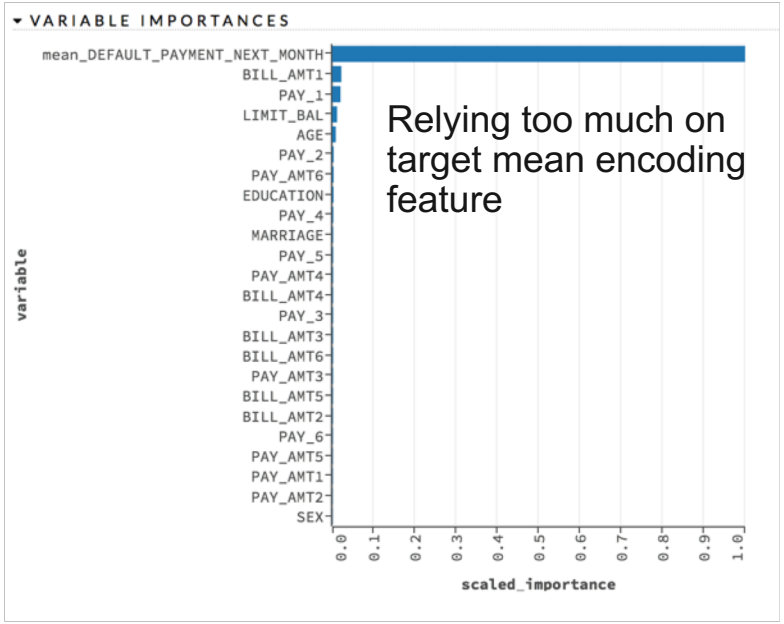
Categorical Feature	Target		Target Mean Encoding
A	1		0.66
A	1	mean	0.66
A	0		0.66
B	0		0.33
B	1	mean	0.33
B	0		0.33
C	1	same value →	1

***Worst Case Scenario**: Response Column = Mean Target Encoding*

Target Mean Encoding Done Wrong



Scoring History: Training vs Testing



Data Leakage Feature is the only important feature

Solution: Cross-Validation Target Encoding

H2O.ai

Fold	Categorical Feature	Target
1	A	1
2	A	1
3	A	0
1	B	0
2	B	1
3	B	0
1	C	1

Solution: Cross-Validation Target Encoding

Fold	Categorical Feature	Target
2	A	1
3	A	0
2	B	1
3	B	0

Fold	Categorical Feature	Target
1	A	1
1	B	0
1	C	1

Solution: Cross-Validation Target Encoding

H2O.ai

Fold	Categorical Feature	Target	Target Mean Encoding
2	A	1	0.5
3	A	0	0.5
2	B	1	0.5
3	B	0	0.5

Fold	Categorical Feature	Target
1	A	1
1	B	0
1	C	1

Solution: Cross-Validation Target Encoding

H2O.ai

Fold	Categorical Feature	Target	Target Mean Encoding
2	A	1	0.5
3	A	0	0.5
2	B	1	0.5
3	B	0	0.5

Fold	Categorical Feature	Target	CV Target Encoding
1	A	1	**0.5**
1	B	0	
1	C	1	

Solution: Cross-Validation Target Encoding

Fold	Categorical Feature	Target	Target Mean Encoding
2	A	1	0.5
3	A	0	0.5
2	B	1	0.5
3	B	0	0.5

Fold	Categorical Feature	Target	CV Target Encoding
1	A	1	0.5
1	B	0	**0.5**
1	C	1	

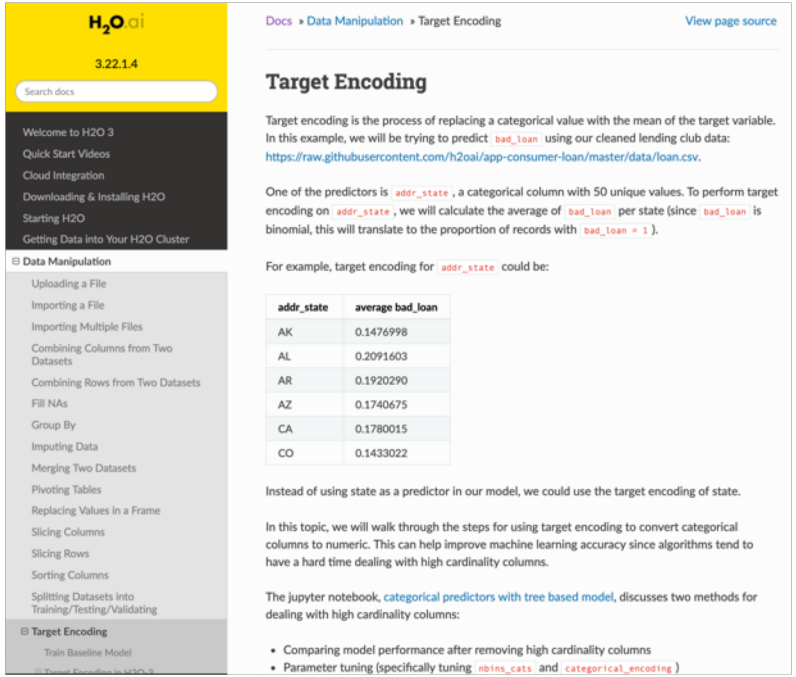
Solution: Cross-Validation Target Encoding

Fold	Categorical Feature	Target	Target Mean Encoding
2	A	1	0.5
3	A	0	0.5
2	B	1	0.5
3	B	0	0.5

No info.

Fold	Categorical Feature	Target	CV Target Encoding
1	A	1	0.5
1	B	0	0.5
1	C	1	NA

Using CV Target Encoding with H2O



H2O.ai

3.22.1.4

Search docs

- Welcome to H2O 3
- Quick Start Videos
- Cloud Integration
- Downloading & Installing H2O
- Starting H2O
- Getting Data into Your H2O Cluster
- Data Manipulation
  - Uploading a File
  - Importing a File
  - Importing Multiple Files
  - Combining Columns from Two Datasets
  - Combining Rows from Two Datasets
  - Fill NAs
  - Group By
  - Imputing Data
  - Merging Two Datasets
  - Pivoting Tables
  - Replacing Values in a Frame
  - Slicing Columns
  - Slicing Rows
  - Sorting Columns
  - Splitting Datasets into Training/Testing/Validating
  - Target Encoding
    - Train Baseline Model

Docs » Data Manipulation » Target Encoding — View page source

Target Encoding

Target encoding is the process of replacing a categorical value with the mean of the target variable. In this example, we will be trying to predict `bad_loan` using our cleaned lending club data: https://raw.githubusercontent.com/h2oai/app-consumer-loan/master/data/loan.csv.

One of the predictors is `addr_state`, a categorical column with 50 unique values. To perform target encoding on `addr_state`, we will calculate the average of `bad_loan` per state (since `bad_loan` is binomial, this will translate to the proportion of records with `bad_loan = 1`).

For example, target encoding for `addr_state` could be:

addr_state	average bad_loan
AK	0.1476998
AL	0.2091603
AR	0.1920290
AZ	0.1740675
CA	0.1780015
CO	0.1433022

Instead of using state as a predictor in our model, we could use the target encoding of state.

In this topic, we will walk through the steps for using target encoding to convert categorical columns to numeric. This can help improve machine learning accuracy since algorithms tend to have a hard time dealing with high cardinality columns.

The jupyter notebook, categorical predictors with tree based model, discusses two methods for dealing with high cardinality columns:

- Comparing model performance after removing high cardinality columns
- Parameter tuning (specifically tuning `nbins_cats` and `categorical_encoding`)

Comparison

Let's compare all three models:

Evaluation (AUC, Higher = Better):

5-Fold CV (with addr_state): `0.7045098` vs. (without addr_state): `0.7061583` vs. (with TE): `0.7072099`

Test (with addr_state): `0.7069701` vs. (without addr_state): `0.7076197` vs. (with TE) `0.708911`

Higher AUC with TE

My code example:

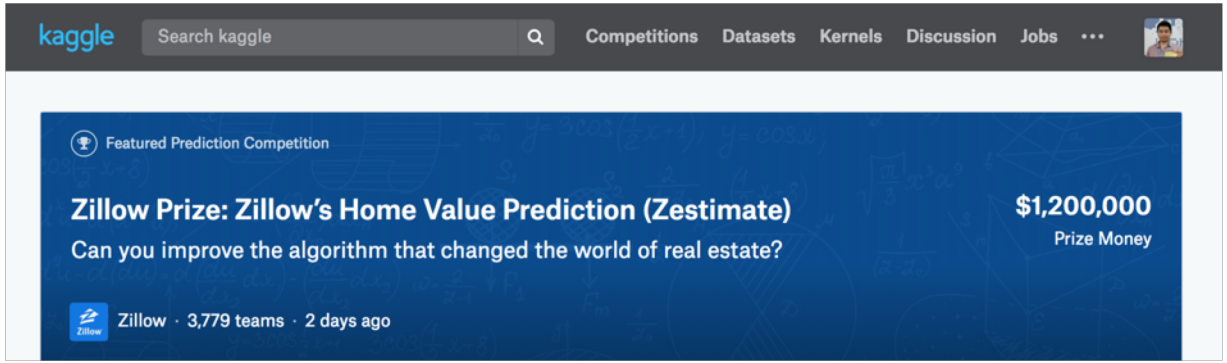
bit.ly/h2o_meetups

H2O-3 Documentation:

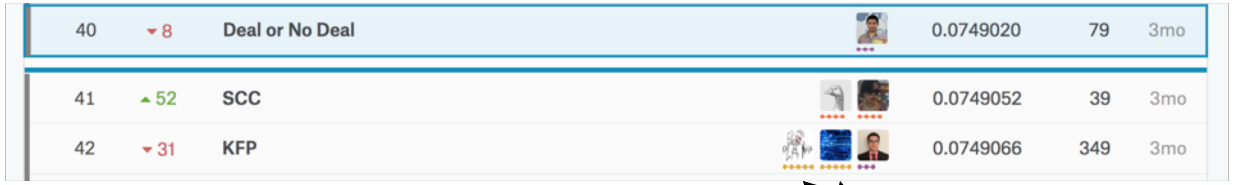
http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-munging/target-encoding.html

CV Target Encoding + Other Feature Engineering Tricks

Competition Round One (Top 100 to Next Round)

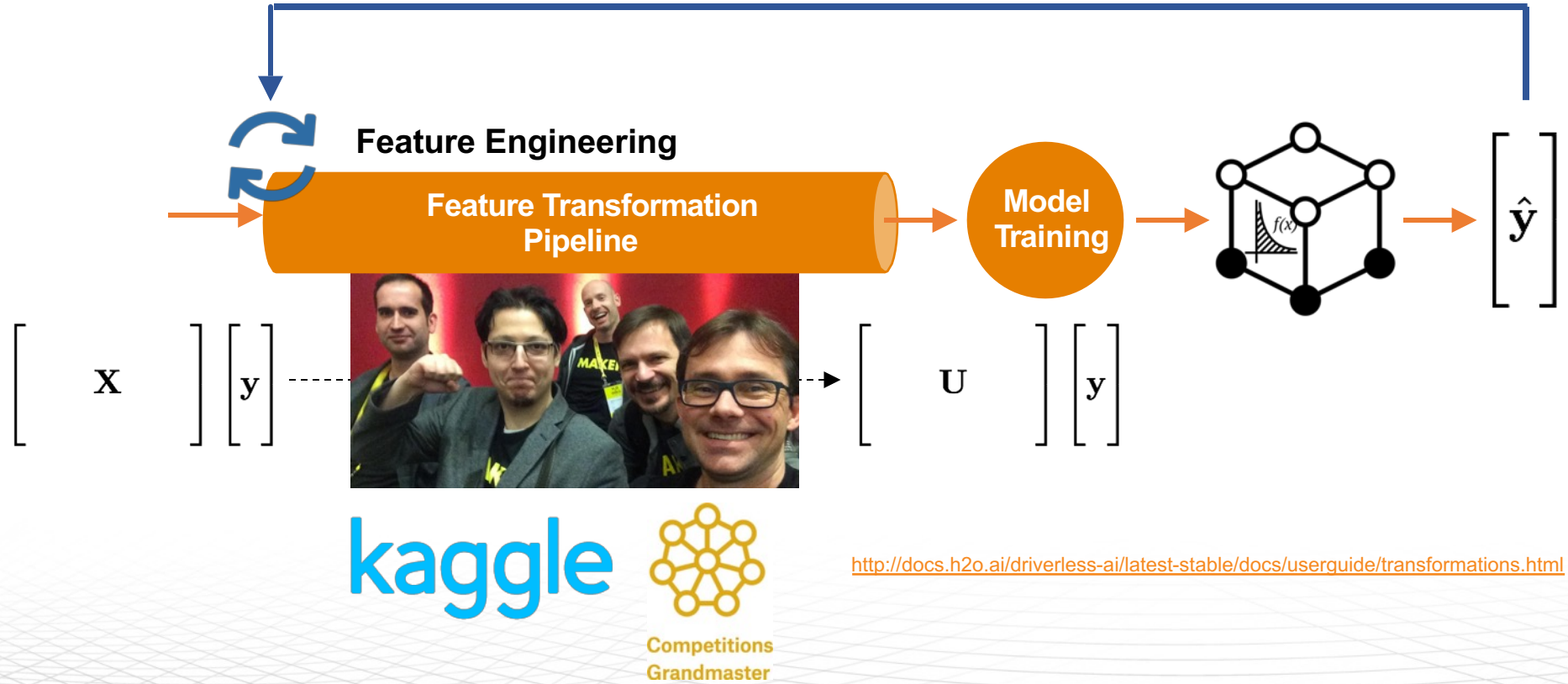


40 out of 3779 teams



Finished above my H2O Kaggle Grandmasters colleagues ☺

Kaggle Grandmaster Automatic Feature Engineering



http://docs.h2o.ai/driverless-ai/latest-stable/docs/userguide/transformations.html

Driverless AI Transformations

Transformations in Driverless AI are applied to columns in the data. The transformers create the engineered features in experiments.

Driverless AI provides a number of transformers. The downloaded experiment logs includes the transformations that were applied to your experiment. Note that you can blacklist transformations in the **config.toml** file, and that list of Blacklisted transformers will also be available in the experiment log.

Available Transformers

The following transformers are available for classification (muliclass and binary) and regression experiments.

- FilterTransformer

  The Filter Transformer counts each numeric value in the dataset.

- FrequentTransformer

  The Frequent Transformer calculates the frequency for each value in categorical column(s) and uses this as a new feature. This count can be either the raw count or the normalized count.

- BulkInteractionsTransformer

  The Bulk Interactions Transformer add, divide, multiply, and subtract two numeric columns in the data to create a new feature.

- ClusterTETransformer

  In the Cluster Target Encoding Transformer clusters selected numeric columns and calculates the mean of the response column for each cluster. The mean of the response is used as a new feature. Cross Validation is used to calculate mean response to prevent overfitting.

- TruncSVDNumTransformer

  Truncated SVD Transformer trains a Truncated SVD model on selected numeric columns and uses the components of the truncated SVD matrix as new features.

- CVTargetEncodeF

  The Cross Validation Target Encoding Transformer calculates the mean of the response column for each value in a categorical column and uses this as a new feature. Cross Validation is used to calculate mean response to prevent overfitting.

- CVCatNumEncodeF

  The Cross Validation Categorical to Numeric Encoding (Fit) Transformer converts a categorical column to a numeric column. Cross validation target encoding is done on the categorical column.

- CVCatNumEncodeDT

  The Cross Validationcal Categorical to Numeric Encoding (culates an aggregation of a numeric column for each value in a categorical column (ex: calculate the mean Temperature for each City) and uses this aggregation as a new feature.

- NumToCatTETransformer

  The Numeric to Categorical Target Encoding Transformer converts a numeric columns to categoricals by binning and then calculates the mean of the response column for each group. The mean of the response for the bin is used as a new feature. Cross Validation is used to calculate mean response to prevent overfitting.

- NumCatTETransformer

  The Numeric Categorical Target Encoding Transformer calculates the mean of the response column for several selected columns. If one of the selected columns is numeric, it is first converted to categorical by binning. The mean of the response column is used as a new feature. Cross Validation is used to calculate mean response to prevent overfitting.

- DatesTransformer

  The Dates Transformer retrieves any date values, including:

  - Year
  - Quarter
  - Month
  - Day
  - Day of year
  - Week
  - Week day
  - Hour
  - Minute
  - Second

- TextTransformer

  The Text Transformer tokenizes a text column and creates a TFIDF matrix (term frequency-inverse document frequency) or count (count of the word) matrix. This may be followed by dimensionality reduction using truncated SVD. Selected components of the TF-IDF/Count matrix are used as new features.

- ClusterDistTransformer

  The Cluster Distance Transformer clusters selected numeric columns and uses the distance to a specific cluster as a new feature.

- WeightOfEvidenceTransformer

  The Weight of Evidence Transformer calculates Weight of Evidence for each value in categorical column(s). The Weight of Evidence is used as a new feature. Weight of Evidence measures the "strength" of a grouping for separating good and bad risk and is calculated by taking the log of the ratio of distributions for a binary response column.

  $$WOE = \ln\left(\frac{\text{Distribution of Goods}}{\text{Distribution of Bads}}\right)$$

  This only works with a binary target variable. The likelihood needs to be created within a stratified kfold if a fit_transform method is used. More information can be found here: http://ucanalytics.com/blogs/information-value-and-weight-of-evidencebanking-case/.

- NumToCatWoETransformer

  The Numeric to Categorical Weight of Evidence Transformer converts a numeric column to categorical by binning and then calculates Weight of Evidence for each bin. The Weight of Evidence is used as a new feature. Weight of Evidence measures the "strength" of a grouping for separating good and bad risk and is calculated by taking the log of the ratio of distributions for a binary response column.

- LagsTransfomer

  The Lags Transformer creates target/feature lags possibly over groups. Each lag is used as a new feature.

- LagsInteractionTransfomer

  The Lags Interaction Transformer creates target/feature lags and calculates interactions between the lags (lag2 - lag1, for instance). The interaction is used as a new feature.

- LagsInteractionTransfomer

  The Lags Interaction Transformer creates target/feature lags and calculates interactions between the lags (lag2 - lag1, for instance). The interaction is used as a new feature.

- LagsAggregatesTransformer

  The Lags Aggregates Transformer calculates aggregations of target/feature lags like mean(lag7, lag14, lag21) with support for mean, min, max, median, sum, skew, kurtosis, std. The aggregation is used as a new feature.

- IsHolidayTransformer

  The Is Holiday Transformer determines if a date column is a holiday. A boolean column indicating if the date is a holiday is added as a new feature.

- NumToCatWoEMonotonicTransformer

  The Numeric to Categorical Weight of Evidence Monotonic Transformer converts a numeric column to categorical by binning and then calculates Weight of Evidence for each bin. The monotonic constraint ensures the bins of values are monotonically related to the Weight of Evidence value. The Weight of Evidence is used as a new feature. Weight of Evidence measures the "strength" of a grouping for separating good and bad risk and is calculated by taking the log of the ratio of distributions for a binary response column.

- TextLinModelTransformer

  The Text Linear Model Transformer trains a linear model on a TF-IDF matrix created from a text feature to predict the response column. The linear model prediction is used as a new feature. Cross Validation is used when training the linear model to prevent overfitting.

- TextCNNTransformer

  The Text CNN Transformer trains a CNN Tensorflow model on word embeddings created from a text feature to predict the response column. The CNN prediction is used as a new a feature. Cross Validation is used when training the CNN model to prevent overfitting.

- OHETransformer

  The One-hot Encoding transformer converts a categorical column to a series of boolean features by performing one-hot encoding. The boolean features are used as new features.

- SortedLETransformer

  The Sorted Label Encoding Transformer sorts a categorical column by the response column and uses the order index created as a new feature.

Accuracy

- Automatic feature engineering to increase accuracy - AlphaGo for AI
- Automatic Kaggle Grandmaster recipes in a box for solving wide variety of use-cases
- Automatic machine learning to find and tune the right ensemble of models