Tsfresh

2023年4月25日 1:43

Tsfresh.extract_features(一些参数)

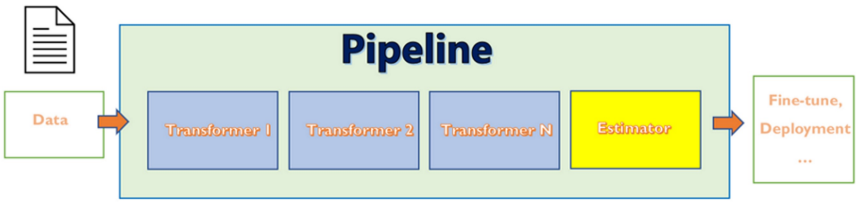
3种输入：

Column_kind, column_value = None：平面数据

Column_kind = kind, column_value = value：所有可能性

Column_kind = None, column_value = value：字典法

Scikit-learn:
Pipeline([('augmenter', RelevantFeatureAugmenter), ('classifier', RandomForestClassifier())])



pipeline将所有的预处理和分类器步骤连接在一起，然后用fit(), predict()来执行工作流。

目的：生成高可读代码，减少机器学习中工作流步骤的重复。

Feature extraction:

tsfresh.feature_extraction.extraction.extract_features: 返回带有提取特征的pandas.dataframe

tsfresh.feature_extraction.feature_calculators:

abs_energy(x): e = sigma(x_i ^2) ---- float

absolute_maximum(x): max(x_i)

absolute_sum_of_changes(x): e = sigma(abs(x_(i+1) - x_i)) ---- float

Agg_autocorrelation(x, param):

计算自相关性（举例来说 5/15天气热， 8/15天气也热， 是否存在相关性）

$$R(l) = \frac{1}{(n-l)\sigma^2}\sum_{t=1}^{n-l}(X_t-\mu)(X_{t+l}-\mu)$$

$$f_{agg}(R(1),\ldots,R(m)) \quad \text{for} \quad m = max(n, maxlag).$$

Param: {“f_agg”: x, “maxlag”, n} 此处x是f_agg在numpy中的函数名称，n为最大滞后数

agg_linear_trend(x, param):
信号被均匀采样的时候才可以使用
计算在 块上聚合 的时间序列值与从 0 到 块数减一 的序列的 **线性最小二乘回归 (不同的特征值)**
Param:“attr”: x, “chunk_len”: l, “f_agg”: f,
f = {"max", "min". "mean", "median"},
x = {"pvalue", "rvalue", "intercept", "slope", "stderr"}

approximate_entropy(x, m, r):
量化时间序列数据波动的规律性和不可预测性
m: Length of compared run of data
r: Filtering level, must be positive

ar_coefficient(x, param):

$$X_t = \varphi_0 + \sum_{i=1}^{k} \varphi_i X_{t-i} + \varepsilon_t$$

Param: {“coeff”: x, “k”: y}，k为最大值， coeff要小于k

augmented_dickey_fuller(x, param):
检查时间序列样本中是否存在单位根
返回相应测试统计量的值
Param: {“attr”: x, “autolag”: y}, x={“teststat”、 “pvalue”或“usedlag”}, y={"AIC", "BIC", "t-stats", None}
与adfuller（）有关

autocorrelation(x, lag):

$$\frac{1}{(n-l)\sigma^2} \sum_{t=1}^{n-l} (X_t - \mu)(X_{t+l} - \mu)$$

L = lag
计算指定lag的自相关

benford_correlation (x):

$$P(d) = \log_{10}\left(1 + \frac{1}{d}\right)$$

对异常检测应用程序很有用
返回第一位分布的相关性

binned_entropy (x , max_bins):

$$-\sum_{k=0}^{min(max\_bins,len(x))} p_k log(p_k) \cdot \mathbf{1}_{(p_k>0)}$$

pk是bin中样本的百分比k

c3(x, lag):

$$\frac{1}{n-2lag}\sum_{i=1}^{n-2lag} x_{i+2\cdot lag} \cdot x_{i+lag} \cdot x_i$$

测量时间序列中的非线性

change_quantiles(x, ql, qh, isabs, f_agg):
通过ql, qh划定一个区间，isabs决定是否用绝对值计算，返回f_agg函数产生的值

cid_ce(x, normalize)
返回时间序列复杂度的估计

$$\sqrt{\sum_{i=1}^{n-1}(x_i - x_{i-1})^2}$$

count_above(x, t)
返回大于t的数字的百分比

count_above_mean(x)
返回大于平均值的数字的数量

count_below(x, t)
返回小于t的数字的百分比

count_below_mean(x)
返回小于平均值的数字的数量

cwt_coefficients(x, param)
$\frac{2}{\sqrt{3a}\pi^{\frac{1}{4}}}(1-\frac{x^2}{a^2})exp(-\frac{x^2}{2a^2})$
Continuous wavelet transform for the Ricker wavelet
Param: {“widths”:x, “coeff”: y, “w”: z}, z用于索引第一个出现w值在widths中的位置

energy_ratio_by_chunks(x, param)
Param: {“num_segments”: N, “segment_focus”: i}, N为块长度， i为第几块，计算第i块的和与所有块的总和的比值。

fft_aggregated(x, param)
Param: {“aggtype”: s}, s= {“centroid”, “variance”, “skew”, “kurtosis”}
先进行绝对值傅里叶变换，然后用s里的函数计算。

fft_coefficient(x, param)
$A_k = \sum_{m=0}^{n-1} a_m \exp\left\{-2\pi i \frac{mk}{n}\right\}, \qquad k = 0, \ldots, n-1.$
Param:{“coeff”: x, “attr”: s}
计算快速傅里叶变换， s={“real”, “imag”, “abs”, “angle”}

first_location_of_maximum(x)
返回第一个最大值的位置

first_location_of_minimum(x)
返回第一个最小值的位置

fourier_entropy(x, bins)
使用welch方法， 计算熵。

friedrich_coefficients(x, param)
$\dot{x}(t) = h(\bar{x}(t)) + \mathcal{N}(0, R)$
计算_estimate_friedrich_coefficients(x, m, r)

has_duplicate(x)
判断是否有重复数值

has_duplicate_max(x)
判断最大值是否重复。

has_duplicate_min(x)
判断最小值是否重复。

index_mass_quantile(x, param)
Param:{“q”: x}, 返回q百分比后的索引i

kurtosis(x)
计算x的峰度

large_standard_deviation(x, r)
$std(x) > r * (max(X) - min(X))$
判断std()是否大于r倍的x差值

last_location_of_maximum(x)
计算最后一个最大值的索引

last_location_of_minimum(x)
计算最后一个最小值的索引

lempel_ziv_complexity(x, bins)
使用Lempel-Ziv compression algorithm来估计复杂度

length(x)

linear_trend(x, param)
在信号均匀采样的情况下
Param: {“attr”: x}, x= {“pvalue”、“rvalue”、“intercept”、“slope”、“stderr”}
返回 在给定特征attr下 linear least-squares regression

linear_trend_timewise(x, param)
使用时间序列的索引
Param: {“attr”: x}, x= {“pvalue”、“rvalue”、“intercept”、“slope”、“stderr”}

longest_strike_above_mean(x)
返回最长的连续大于平均值的子序列

longest_strike_below_mean(x)
返回最长的连续小于平均值的子序列

max_langevin_fixed_point(x, r, m)
$(x)(t) = h(x(t)) + R(N)(0, 1)$
fitted to the deterministic dynamics of Langevin model
使用_estimate_friedrich_coefficients(x, m, r)

maximum(x)

mean(x)

mean_abs_change(x)
计算前后两数的差值的绝对值的平均值。

mean_change(x)
$$\frac{1}{n-1}\sum_{i=1,\ldots,n-1} x_{i+1} - x_i = \frac{1}{n-1}(x_n - x_1)$$
计算前后两数的差值的平均值。

mean_n_absolute_max(x, number_of_maxima)
计算n个最大数的平均值。

mean_second_derivative_central(x)
$$\frac{1}{2(n-2)}\sum_{i=1,\ldots,n-1} \frac{1}{2}(x_{i+2} - 2 \cdot x_{i+1} + x_i)$$
返回二阶导数的平均值。

Median(x)

Minimum(x)

number_crossing_m(x, m)
计算前一个小于m，后一个大于m的数的数量。

number_cwt_peaks(x, n)
通过find_peaks_cwt()计算的峰数的数量

number_peaks(x, n)

计算x中 大于左右n位的数字的个数。

partial_autocorrelation(x, param)
用 OLS 拟合的 AR(k-1) 模型 （自回归模型）

$$\alpha_k = \frac{Cov(x_t, x_{t-k}|x_{t-1}, \ldots, x_{t-k+1})}{\sqrt{Var(x_t|x_{t-1}, \ldots, x_{t-k+1})Var(x_{t-k}|x_{t-1}, \ldots, x_{t-k+1})}}$$

$$x_t = f(x_{t-1}, \ldots, x_{t-k+1})$$

对过去的值进行预测

$$x_{t-k} = f(x_{t-1}, \ldots, x_{t-k+1})$$

未来值用于计算过去的值x_(t-k)
Param: {“lag”: val}
返回给定lag的偏自相关函数的值

percentage_of_reoccurring_datapoints_to_all_datapoints(x)
返回出现超过一次的数字（算多次）的百分比。

percentage_of_reoccurring_values_to_all_values(x)
返回出现超过一次的数字（只算一次）的百分比。

permutation_entropy(x, tau, dimension)
计算排列熵(时间序列随机性和动力学突变行为)

quantile(x, q):
q quantile(分位数) of x, 大于q%

query_similarity_count(x, param)：
Param: {“query”: Q, “threshold”: thr, “normalize”: norm}
返回在时间序列中找到查询的次数计数
Q: 比较查询子序列，省略的话 返回0
thr：增加查询相似度
Norm: 判断是否norm, 布尔值

range_count(x, min, max)
返回min到max中的值

ratio_beyond_r_sigma(x, r)：
大于 r 乘以 sigma 的值与 x 平均值的比值。

ratio_value_number_to_time_series_length(x)：
所有值只出现一次，返回值大于1，反之小于1

root_mean_square(x):
返回均方根

sample_entropy(x):
return sample entropy.

set_property(key, value)：
返回一个函数decorate_func

skewness(x)：
返回样本偏度（Fisher-Pearson）

spkt_welch_density(x, param)
x 在不同频率下的交叉功率谱密度
Param: {"coeff": x} , x为不同频率（存疑，doc里没指明，代码是这样的）

standard_deviation(x)：

sum_of_reoccurring_data_points(x)：

返回出现超过一次的数字（算多次）的总和。

sum_of_reoccurring_values(x)
返回出现超过一次的数字（只算一次）的总和。

sum_values(x)：
sum

symmetry_looking(x, param):
$|mean(X) - median(X)| < r * (max(X) - min(X))$
{"r": x} , x = the percentage of the range
返回布尔值

time_reversal_asymmetry_statistic(x, lag):
$$\frac{1}{n - 2lag} \sum_{i=1}^{n-2lag} x_{i+2\cdot lag}^2 \cdot x_{i+lag} - x_{i+lag} \cdot x_i^2$$
the time reversal asymmetry statistic.

value_count(x, value):
计算value在时间序列 x 中的出现次数

variance(x)
返回 x 的方差

variance_larger_than_standard_deviation(x):
方差是否高于标准差

variation_coefficient(x)
Returns the variation coefficient (standard error / mean)

Feature filtering --Fresh algorithm:

Step1: tsfresh.feature_extraction.feature_calculators

Step2: tsfresh.feature_selection.significance_tests