**程序报告**

学号：2111454 姓名：李潇逸

1. **问题重述**

kmeans实现异常点检测。

1. **设计思想**

利用机器学习进行模型构建和训练

1. **代码内容**

import os

import sklearn

import numpy as np

import pandas as pd

from copy import deepcopy

from sklearn.cluster import KMeans

from sklearn.externals import joblib

from scipy.spatial.distance import cdist

class KMeans():

"""

Parameters ----------

"""

def \_\_init\_\_(

self,

n\_clusters,

n\_init,

max\_iter

):

self.n\_clusters = n\_clusters

self.max\_iter = max\_iter

self.n\_init = n\_init

self.centroids=[]

def fit(self, x):

"""

用fit方法对数据进行聚类

:param x: 输入数据

:best\_centers: 簇中心点坐标 数据类型: ndarray

:best\_labels: 聚类标签 数据类型: ndarray

:return: self

"""

data = x.values

centroids=np.array(self.centroids,dtype=np.float)

if(centroids.shape==(0,)):

centroids=data[np.random.randint(0,data.shape[0],self.n\_clusters),:]

for i in range(self.max\_iter):

distances=cdist(data,centroids)

labels=np.argmin(distances,axis=1)

for i in range(self.n\_clusters):

if i in labels:

centroids[i]=np.mean(data[labels==i],axis=0)

best\_centers = centroids

best\_labels = labels

self.cluster\_centers\_ = best\_centers

self.labels\_ = best\_labels

return self

def preprocess\_data(df):

"""

数据处理及特征工程等

:param df: 读取原始 csv 数据，有 timestamp、cpc、cpm 共 3 列特征

:return: 处理后的数据, 返回 pca 降维后的特征

"""

# 请使用joblib函数加载自己训练的 scaler、pca 模型，方便在测试时系统对数据进行相同的变换

# ====================数据预处理、构造特征等========================

# 例如

df['timestamp'] = pd.to\_datetime(df['timestamp'])

df['cpc X cpm'] = df['cpm'] \* df['cpc']

df['cpc / cpm'] = df['cpc'] / df['cpm']

df['hours'] = df['timestamp'].dt.hour

df['daylight'] = ((df['hours'] >= 7) & (df['hours'] <= 22)).astype(int)

# ======================== 模型加载 ===========================

# 请确认需要用到的列名，e.g.:columns = ['cpc','cpm']

columns = None

columns = ['cpc', 'cpm', 'cpc X cpm', 'cpc / cpm']

data = df[columns]

scaler = None

pca = None

# 例如

# scaler = joblib.load('./results/scaler.pkl')

# pca = joblib.load('./results/pca.pkl')

# data = scaler.transform(data)

scaler = joblib.load('./results/scaler.pkl')

pca = joblib.load('./results/pca.pkl')

data = scaler.transform(data)

data = scaler.fit\_transform(data)

data = pd.DataFrame(data, columns=columns)

#通过 n\_components 指定需要降低到的维度

n\_components = 3

data = pca.fit\_transform(data)

data = pd.DataFrame(data,columns=['Dimension' + str(i+1) for i in range(n\_components)])

return data

def get\_distance(data, kmeans, n\_features):

"""

计算样本点与聚类中心的距离

:param data: preprocess\_data 函数返回值，即 pca 降维后的数据

:param kmeans: 通过 joblib 加载的模型对象，或者训练好的 kmeans 模型

:param n\_features: 计算距离需要的特征的数量

:return:每个点距离自己簇中心的距离，Series 类型

"""

# ====================计算样本点与聚类中心的距离========================

distance = []

for i in range(0,len(data)):

point = np.array(data.iloc[i,:n\_features])

center = kmeans.cluster\_centers\_[kmeans.labels\_[i],:n\_features]

distance.append(np.linalg.norm(point - center))

distance = pd.Series(distance)

return distance

def get\_anomaly(data, kmean, ratio):

# ====================检验出样本中的异常点========================

#num\_anomaly = int(len(data) \* ratio)

num\_anomaly = int(len(data) \* ratio)

new\_data = deepcopy(data)

#data = new\_data

new\_data['distance'] = get\_distance(new\_data,kmeans,n\_features=len(new\_data.columns))

threshould = new\_data['distance'].sort\_values(ascending=False).reset\_index(drop=True) [num\_anomaly]

new\_data['is\_anomaly'] = new\_data['distance'].apply(lambda x: x > threshould)

#data['is\_anomaly'] = None

return new\_data

def predict(preprocess\_data):

"""

该函数将被用于测试，请不要修改函数的输入输出，并按照自己的模型返回相关的数据。

在函数内部加载 kmeans 模型并使用 get\_anomaly 得到每个样本点异常值的判断

:param preprocess\_data: preprocess\_data函数的返回值，一般是 DataFrame 类型

:return:is\_anomaly:get\_anomaly函数的返回值，各个属性应该为

（Dimesion1,Dimension2,......数量取决于具体的pca），distance,is\_anomaly，请确保这些列存

在

preprocess\_data: 即直接返回输入的数据

kmeans: 通过joblib加载的对象

ratio: 异常点的比例，ratio <= 0.03 返回非异常点得分将受到惩罚！

"""

# 异常值所占比率

ratio = 0.015

# 加载模型

#kmeans = joblib.load('./results/model.pkl')

kmeans = joblib.load('./results/model.pkl')

# 获取异常点数据信息

# is\_anomaly = get\_anomaly(preprocess\_data, kmeans, ratio)

is\_anomaly = get\_anomaly(preprocess\_data, kmeans, ratio)

return is\_anomaly, preprocess\_data, kmeans, ratio

1. **实验结果**



1. **总结**

达到了与其设计目的。

改进方向有如下几点：

* 选择更好的模型，使准确率提升。
* 考虑使用深度学习。
* 尝试了更改训练模型为自己所写，然而收效甚微
* 尝试更改了超参数，结果差别不大