数据挖掘第二次作业 q2 报告

> 1651718 方沛
>
> 产生的数据集`time_sery.csv`已经放在文件夹中

题目要求

特征工程: 对每个商品编号，设计如下特征信息：

1. 商品编号、品牌编号、4 级品类结构、日期（标记为 d）、是否工作日、当日销量、前 1 日 (即 d-1 日)至前 7 日(即 d-7 日)当天销量，计 16 个特征；
2. 该商品对应品牌前 1 日至前 7 日当日销量，计 7 个特征；
3. 该商品对应 4 级品类前 1 日至前 7 日当日销量，计 28 个特征
4. 该商品前第 2 周（即 d-8 日至 d-14 日）、前第 3 周（即 d-15 日至 d-21 日）、前第 4 周（即 d-22 日至 d-28 日）中该周的每日销量平均值、该周的某日最大值、该周的某日最小值， 计 *33=9 个特征;*
5. *该品牌前第 2 周（即 d-8 日至 d-14 日）、前第 3 周（即 d-15 日至 d-21 日）、前第 4 周（即 d-22 日至 d-28 日）中该周的销量每日平均值、该周的某日最大值、该周的某日最小值， 计 33=9* 个特征；
6. 该 4 级品牌前第 2 周（即 d-8 日至 d-14 日）、前第 3 周（即 d-15 日至 d-21 日）、前第 4 周（即 d-22 日至 d-28 日）中该周的销量每日平均值、该周的某日最大值、该周的某日最 小值，计 *334=36* 个特征；

代码设计

预处理数据

读入数据集,设置以pluno为索引

```
dataset = pd.read_csv("trade_new.csv").fillna(0)
dataset['pluno'] = dataset['pluno'].astype('str')
dataset['sldatime']=pd.to_datetime(dataset['sldatime'])
dataset['sldatime']=[datetime.datetime.strftime(x,'%Y-%m-%d') for x in dataset['sldatime']]
dataset = dataset.set_index('pluno',drop = False)
```

插入初始的没有时间序列信息的数据

```
for pluno in dataset.index.unique():
    if str(pluno)[:2] not in ['22','23','25','27']:
        continue
    # 一条数据
    pluno_data=dataset.loc[[pluno]]
    for index, good in pluno_data.iterrows():
        row = dict()
        row['pluno'] = good['pluno']
        row['bndno'] = good['bndno']
        row['pl_1'] = int(good['pluno'][:2])
        row['pl_2'] = int(good['pluno'][:3])
```

```
        row['pl_3'] = int(good['pluno'][:4])
        row['pl_4'] = int(good['pluno'][:5])
        row['purchase_date'] = good['sldatime']
        row['is_workday'] = is_workday(good['sldatime'])
        row['qty'] = float(good['qty'])
        row['prev_d1'] = 0
        row['prev_d1'] = 0
        row['prev_d2'] = 0
        row['prev_d3'] = 0
        row['prev_d4'] = 0
        row['prev_d5'] = 0
        row['prev_d6'] = 0
        row['prev_d7'] = 0
        ...
        data_values.append(list(row.values()))

```

时间序列数据聚合

对每个pluno及其所对应的日期进行聚合

```
indexs=['pluno','purchase_date']
'''
@descrption: 为数据集属性指定聚合函数
@params:
    - properties: 一个dataframe的columns
@output: 聚合函数列表
'''
def get_func(properties):
    funcs = dict()
    for prop in properties:
        if prop == 'pluno' or prop == 'purchase_date':
            continue
        elif prop == 'qty':
            funcs[prop] = 'sum'
        else:
            funcs[prop] = 'mean'
    return funcs
# 转换格式才能聚合
for col in df.columns:
    if col == 'pluno' or col == 'purchase_date':
        continue
    elif col in ['pl_1','pl_2','pl_3','pl_4']:
        df[col]=df[col].astype("int")
    else:
        df[col]=df[col].astype("float")

funcs = get_func(properties)
df=df.groupby(['pluno','purchase_date']).agg(funcs)
```

聚合结果

```
In [12]: df=df.groupby(['pluno','purchase_date']).agg(funcs)

In [13]: df=df.reset_index()

In [14]: df.head(5)
```

Out[14]:

	pluno	purchase_date	bndno	pl_1	pl_2	pl_3	pl_4	is_workday	qty	prev_d1	...	pl3_4week_min	pl4_2week_avg	pl4_2week_max	pl4_2week_...
0	22000005	2016-07-31	0.0	22	220	2200	22000	0.0	0.704	0.0	...	0.0	0.0	0.0	
1	22000008	2016-02-18	0.0	22	220	2200	22000	1.0	0.704	0.0	...	0.0	0.0	0.0	
2	22000009	2016-07-25	0.0	22	220	2200	22000	1.0	0.666	0.0	...	0.0	0.0	0.0	
3	22000009	2016-07-27	0.0	22	220	2200	22000	1.0	1.120	0.0	...	0.0	0.0	0.0	
4	22000010	2016-03-16	0.0	22	220	2200	22000	1.0	1.914	0.0	...	0.0	0.0	0.0	

5 rows × 105 columns

数据工程流程

为了方便调试,将题目中要求的特征生成各封装成了函数

以下函数与题目序号一一对应:

```
'''
@descrption: 产生每个pluno过去7天的销量
@params:
    - df:目标数据集
'''
def set_prev_d(df):
    count =0
    pluno_indexs = df.index.levels[0]
    date_indexs= df.index.levels[1]
    for p in pluno_indexs:
        count = count+1
        print("p:{}{}".format(p,count))
        subset = df.loc[p]
        if subset.shape[0] == 1:
            continue
        else:
            subdates = list(reversed(subset.index))
            for i in range(len(subdates)):
                now_date = pd.to_datetime(subdates[i])
                for j in range(i+1,len(subdates)):
                    prev_date = pd.to_datetime(subdates[j])
                    #print(prev_date)
                    interval_days = int((now_date - prev_date).days)
                    if 0<interval_days<=7:
                        col = 'prev_d'+ str(interval_days)
                        df.at[(p,subdates[i]),col]=df.at[(p,subdates[i]),col]+
df.at[(p,subdates[j]),'qty']
                        if df.at[(p,subdates[j]),'qty']!=0:
                            print("index:",p,subdates[j],df.at[(p,subdates[j]),'qty'])
                            print(df.at[(p,subdates[i]),col])
                            print('-'*30)
                    else:
```

```
                        break
'''
@descrption: 产生每个品类过去7天的销量
@params:
    - df:目标数据集
    - lv:具体哪一级品类
'''
def set_pl_d(df,lv):
    all_properties = np.unique([pl[:lv+1] for pl in df['pluno']])
    for prop in all_properties:
        print("prop:",prop)
        subset = df.loc[df['pluno'].str.startswith(prop),:]
        sorted_subset = subset.sort_values(by=['purchase_date'],ascending=[False])
        plength=len(sorted_subset.index)
        for i in range(plength):
            now_date = pd.to_datetime(sorted_subset.iloc[i]['purchase_date'])
            now = sorted_subset.index[i]
            for j in range(i+1,plength):
                prev_date = pd.to_datetime(sorted_subset.iloc[j]['purchase_date'])
                interval_days = (now_date - prev_date).days
                if 0<interval_days<=7:
                    col = 'pl{}_d{}'.format(lv,interval_days)
                    prev = sorted_subset.index[j]
                    df.at[now,col] =df.at[now,col]+df.at[prev,'qty']
                    if df.at[prev,'qty']!=0:
                        print(now,col,df.at[now,col])
                else:
                    break
'''
@descrption: 产生每个品类过去几周的最大最小和均值
@params:
    - df:目标数据集
    - wk:具体哪一周
'''
def set_pluno_week(df,wk):
    min_day = 7*(wk-1)
    max_day = min_day+7
    all_plunos = np.unique(df['pluno'])
    colmax = 'pl_{}week_max'.format(wk)
    colavg = 'pl_{}week_avg'.format(wk)
    colmin = 'pl_{}week_min'.format(wk)
#     tt=0
    for pluno in all_plunos:
        print("pluno:",pluno)
        subset = df.loc[df['pluno']==pluno,:]
        sorted_subset = subset.sort_values(by=['purchase_date'],ascending=[False])

        plength=len(sorted_subset.index)
        for i in range(plength):
            now_date = pd.to_datetime(sorted_subset.iloc[i]['purchase_date'])
            #print("now_date:",now_date)
            now = sorted_subset.index[i]
            count = 0
```

```
            for j in range(i+min_day,plength):
                prev_date = pd.to_datetime(sorted_subset.iloc[j]['purchase_date'])
                #print("prev_date:",prev_date)
                interval_days = int((now_date - prev_date).days)
                #print("intrval_days:",min_day,interval_days,max_day)
                if interval_days>max_day:
                    break
                elif min_day<interval_days<=max_day:
                    count = count+1
                    prev = sorted_subset.index[j]
                    if df.at[prev,'qty']>df.at[now,colmax]:
                        df.at[now,colmax] = df.at[prev,'qty']
                        #print("max:",df.at[now,colmax])
                    elif df.at[prev,'qty']<df.at[now,colmin]:
                        df.at[now,colmin] =df.at[prev,'qty']
                        #print("min:",df.at[now,colmax])
                    df.at[now,colavg]=df.at[now,colavg]+df.at[prev,'qty']
                    #print("avg:",df.at[now,colmax])

            if not count == 0:
                df.at[now,colavg] = df.a
                t[now,colavg]/count
            #print(df.at[now,colavg])


'''
@descrption: 产生每个品牌过去几周的最大最小和均值,由于所选的商品都没有品牌,直接复制品类
@params:
    - df:目标数据集
'''
def set_bndno_week(df):
    df['bnd_2week_avg'] = df['pl_2week_avg']
    df['bnd_2week_max'] = df['pl_2week_max']
    df['bnd_2week_min'] = df['pl_2week_min']
    df['bnd_3week_avg'] = df['pl_3week_avg']
    df['bnd_3week_max'] = df['pl_3week_max']
    df['bnd_3week_min'] = df['pl_3week_min']
    df['bnd_4week_avg'] = df['pl_4week_avg']
    df['bnd_4week_max'] = df['pl_4week_max']
    df['bnd_4week_min'] = df['pl_4week_min']

'''
@descrption: 产生每个品类过去几周的最大最小和均值
@params:
    - df:目标数据集
    - wk:哪一周
    - lv:哪一级品类
'''
def set_smaller_pluno_week(df,wk,lv):
    min_day = 7*(wk-1)
    max_day = min_day+7
    all_plunos = np.unique([pl[:lv+1] for pl in df['pluno']])
    colmax = 'pl{}_{}week_max'.format(lv,wk)
```

```
    colavg = 'pl{}_{}week_avg'.format(lv,wk)
    colmin = 'pl{}_{}week_min'.format(lv,wk)
    for pluno in all_plunos:
        print(pluno)
        subset = df.loc[df['pluno'].str.startswith(pluno),:]
        sorted_subset = subset.sort_values(by=['purchase_date'],ascending=[False])

        for date in reversed(np.unique(sorted_subset['purchase_date'])):
            #print("date:",date)
            indexs = sorted_subset[sorted_subset["purchase_date"]==date].index
            start_date = pd.to_datetime(date)
            before_dates =[start_date-dateutil.relativedelta.relativedelta(days=x) for x in range(min_day,max_day)]
            before_dates = [datetime.datetime.strftime(x,'%Y-%m-%d') for x in before_dates]
            calculated_data = sorted_subset[sorted_subset["purchase_date"].isin(before_dates)]
            df.loc[indexs,colmax]=np.max(calculated_data['qty'])
            df.loc[indexs,colavg]=np.mean(calculated_data['qty'])
            df.loc[indexs,colmin]=np.min(calculated_data['qty'])
```

在完成这些函数后,我们将它们封装入 `my_feature_engineering_pipeline` 中方便调用

```
'''
@descrption: 整个特征工程流程
@params:
    - df:目标数据集
'''
def my_feature_engineering_pipeline(df):
    print('step 1')
    df = df.set_index(['pluno','purchase_date'])
    set_prev_d(df)
    df =df.reset_index()
    set_bnd_d(df)

    print('step 2')

    set_pl_d(df,1)
    set_pl_d(df,2)
    set_pl_d(df,3)
    set_pl_d(df,4)

    print('step 3')

    set_pluno_week(df,2)
    set_pluno_week(df,3)
    set_pluno_week(df,4)

    print('step 4')
    set_bndno_week(df)
    set_smaller_pluno_week(df,2,1)
    set_smaller_pluno_week(df,3,1)
    set_smaller_pluno_week(df,4,1)
```

```
    set_smaller_pluno_week(df,2,2)
    set_smaller_pluno_week(df,3,2)
    set_smaller_pluno_week(df,4,2)
    set_smaller_pluno_week(df,2,3)
    set_smaller_pluno_week(df,3,3)
    set_smaller_pluno_week(df,4,3)
    set_smaller_pluno_week(df,2,4)
    set_smaller_pluno_week(df,3,4)
    set_smaller_pluno_week(df,4,4)
```