


探究初始数据分布下少量异常值对模型精度的影响

在这里我们注意到，训练集、PB数据集、PR数据集均出自2016年1-6月出租车数据，并未做过任何数据处理。这就导致一个情况：当我们对训练集进行数据清洗时，PB、PR数据集还是保留了原始的噪声分布。那么，请作如下考虑：

对训练集进行高度数据清洗，是否会导致模型过拟合从而无法适应噪声？

本文将探究该状况，判断如何做数据清洗，做怎样的数据清洗。

在不叠加其他数据集的情况下，我们为原始数据集构造了六十一个特征，当然，并非所有特征都能使用，我们筛选了三十八个特征维度，如下所示，存在一堆One-hot编码数据，所以整个数据占用内存不多。

 image-20240820162036055

随后，我们将其中的650000个数据作为测试集，并不对这部分的测试集进行数据清洗。注意到我们构造了 `Speed` 特征，其具体含义是 $n \text{ km/h}$ ，其分布为：

```
count    799979.000000
mean      14.519106
std       17.619875
min        0.001495
25%        9.187925
50%       12.839394
75%       17.879138
max      9274.836731
Name: speed, dtype: float64
```

不难发现，最小值近乎不动，最大值达到了2576.3 m/s，显然不太正常。我们设定正常范围 A ， B ， C ，分别探讨不同清洗情况下的性能。

其中， A 表示将城际出租车的速度限制在 $[5, 60]$ ，这是比较合理的区间。 B 表示不限制最低时速，为 $[0, 60]$ ， C 表示一个更加宽松的标准 $[1, 70]$ ，分别进行独立实验。

```
A,C,B=x_train[(x_train.speed>5)&
(x_train.speed<60)],x_train[(x_train.speed>1)&
(x_train.speed<70)],x_train[x_train.speed<60]
```

```
A.shape,B.shape,C.shape
# ((764918, 39), (799601, 39), (796795, 39))
```

随后, 引入 `Light GBM` 进行实验。

```
def xgb_train(x,y):
    xgb_regressor = xgb.XGBRegressor(
        n_estimators=500,
        learning_rate=0.1,
        max_depth=25,
        subsample=0.9,
        random_state=42,
        colsample_bytree=0.8,
        eta=0.05
    )
    xgb_regressor.fit(x, y)
    print("Score in Train %.2f"%np.exp(MSE(xgb_regressor.predict(x),y)))
    print("Score in Val
%.2f"%np.exp(MSE(xgb_regressor.predict(x_val),y_val)))
```

```
%%time
print("A")
print("*"*10)
xgb_train(A,A_y)
print("*"*10)

'''
A
*****
Score in Train 0.27
Score in Val 0.41
*****
'''
```

```
%%time
print("B")
print("*"*10)
xgb_train(B,B_y)
print("*"*10)

'''
B
*****
Score in Train 0.37
Score in Val 0.38
*****
'''
```

```
%%time
print("C")
print("*"*10)
xgb_train(C,C_y)
```

```


print("*"*10)

'''
C
*****
Score in Train 0.31
Score in Val 0.39
*****
'''

```

可以看到，A 在训练集上的表现最好，说明我们范围设置的不错，但是很遗憾，其评分在验证集上表现最差。

我们去真实的LB上看一下情况：

 image-20240820170847711


B区间的性能远超A、C区间，那么引申出一个问题，如果我什么都不做呢？

```

%%time
print("Ori")
print("*"*10)
c=xgb_train(ori,y_train)
print("*"*10)

'''
Score in Train 0.37
Score in Val 0.38
'''

```

 image-20240820171934080

因此得出结论：当两个数据集都取自同一个未处理过的原始分布，什么都不做的效果比做数据清洗的好。随后，我们会验证特征数据漂移对结果的影响。