# 前期准备

## 建议流程

### 流程一

1. 仔细阅读比赛的概述和数据的描述；

2. 查找类似的Kaggle比赛。作为刚加入Kaggle社区不久的新手，我对所有的Kaggle比赛进行了搜索和分析；

3. 研究类似比赛的解决方案；

4. 阅读相关论文，确保我跟进该领域的最新进展；

5. 分析数据，并构建可靠的交叉验证；

6. 进行数据预处理、特征工程、模型训练；

7. 进行结果分析，包括预测分布、错误分析、困难样本等；

8. 根据分析，精心设计模型或设计新模型；

9. 基于数据分析和结果分析，通过设计模型增加多样性，或解决困难样本；

10. 模型集成；

11. 必要时返回前面某个步骤

### 流程二

1. 提出问题

根据上述操作，你或许有一些疑问，可以尝试在比赛论坛提出或者记录下自己的疑问。

2.想法

验证想尝试想法，验证方法次数越多，越接近正确答案。

3. 数据清洗

对原始数据集处理为最干净的形式，不同的数据集需要不同类型的预处理和转换，有时需要不同的模型。

4. 特征工程

新功能可以显着提高模型性能。不同类型的特征可能适用于不同的模型。 去野外。努力尝试，尝试尽可能多的功能和想法。

5. 错误分析

确定模型能够很好预测和失败原因，考虑结果的原因以及你可以做些什么。

1.改进可视化

增加对数据的可视化和演示，研究高票数的 EDA Notebook并学习如何构建出色的可视化。

2. 发布你的EDA Notebook

使Kaggle成为如此出色平台的原因是Kagglers和社区，你也可以做一份贡献。

3. 分享见解

在论坛发帖，或者写一些评论。你分享的越多，你学到的就越多，也会有更多的人帮助你。

1.探索模型

这是您试验和构建大量模型以找出最有效的模型的机会，阅读并理解不同模型的内部工作原理很重要。

2. 集成并组合模型

使用blend或者stacking将多个模型组合起来，并为每个基础模型搜索得到各自的特征空间和超参数。

# 数据

Tied comments 不会有相同结果的句子 df[“score”]=df[“score”].rank(method=”first”)

有5302对有2个评估，5302个有1个，当存在相斥情况时引入第三个worker评估

由于计算方式是逐对与专家评估结果进行对比，最大分数不为1.

提供了两两句子的比较结果作为验证集，提交要求在一系列句子上生成一个句子毒性的排名。

有10000条数据被评估三次 108 条被评估1次 评估1次的准确率为69

Tied comments：df[“score”]=df[“score”].rank(method=”first”)

RealToxicityPrompts dataset (2020): contains 100k sentence snippets from the web with Perspective API toxicity scores for studying the risk of neural toxic degeneration in language models."

"OLID (Offensive Language Identification Dataset; 2019): contains 14,100 English tweets, annotated according to the three-level taxonomy as described here. "

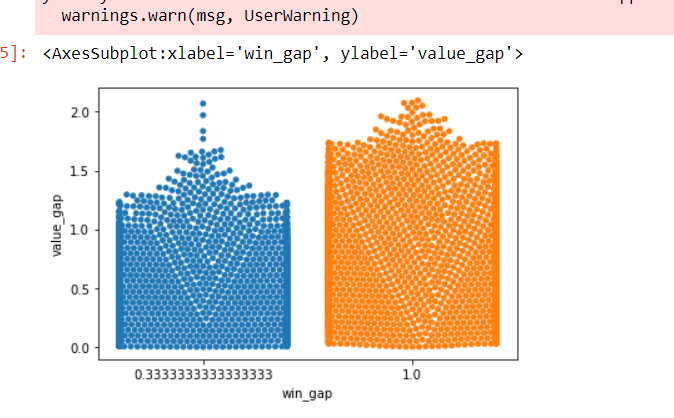
评论被标注为使另一个人脱离对话的概率。有毒被定义为任何大于0.5的概率。

## 数据增强

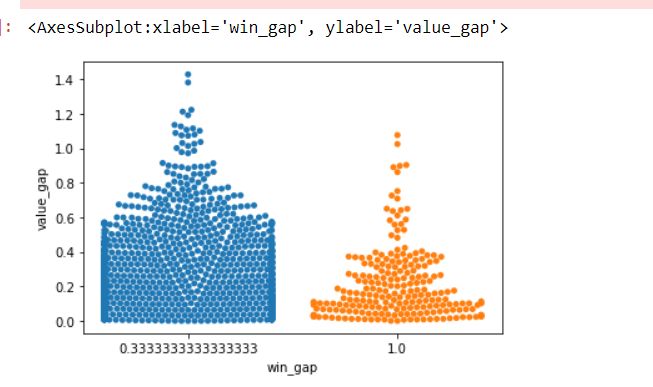
Dpa的负采样策略

## 数据分析

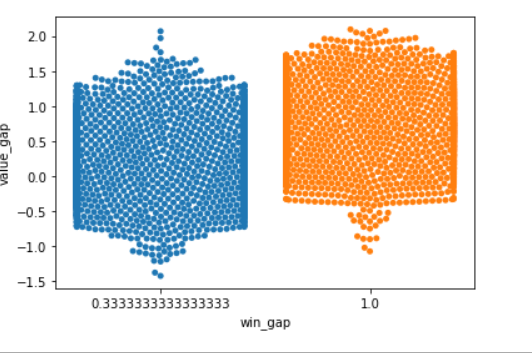
在正确评估中 较难评估的 和较容易评估的 分值差



错误评估中较难评估的和较容易评估的分值差



所有样本中



## 参考注释方法

BWS注释办法：四个组一起比较，进行a、b、c、d的打分。（v，a，d与攻击性也有关联）

# 模型处理

## 1：二分类训练

### 点到排名方法

适应性方法：带参数 bt和thurstiones’s model

非参数：duelling bandits模型

Bt模型 加一个方差 反映不同参赛队发挥稳定性差别

## 2：毒性判断

实际 训练一个能判断毒性程度的model 用两个input 进行比较 结合MarginRankingloss 调整模型

MarginRankingLoss: 

当y=1 ,认为x1>x2。若实际也是如此loss=0，否则不为0.y取1 -1（？） 损失函数与目标是否一致（？） 如果模型是训练一个毒性打分模型，两个这样的模型计算损失，是符合的。

## 稳健cv策略

Cv策略origin：每个fold 单独编号 评估计算准确性 ，

错误原因：不同fold 模型可能在两对相反结果上都猜对，cv虚高

Cv 策略1：每个fold中文本不单独评估 ，重复评估使用平均值计算准确性

Cv策略2：对单个comment 进行fold （但是会损失一部分标注）

Cv策略3： 1 额外数据 训练 val 验证 2.val 做fold训练 3.剩余10k加入训练

Cv4：将进行比较过的数据放入同一group，不同group不拆开到不同fold中

## 过去比赛经验

稳健的cv：lb上升，cv对数损失下降才认为合适。

数据增强：反复翻译 翻译增强方法 在ridge 上 和在bert上

第二次比赛：减少身份偏差带来的影响，平衡auc与减少偏差作为metric

## 待定策略

回归问题转分类问题

将过去比赛数据转换为 less和more的形式

翻译增强方法

新的消除量纲方法在 ridge 上效果分析

线性模型再优化

# 调参

## 可调参条目：

数据集占比、模型集合占比、量纲调整、fold数目调整、challenge分值的调整、

# 集成

当模型足够多样化时 考虑将其纳入 如果模型的 Pearson 相关性 > 0.99 且 Kolmogorov-Smirnov 统计量 < 0.01，则混合它们不太可能获得太多好处。一个非常宽松的经验法则：您需要 Pearson < 0.95（< 0.9 更好）和 K-S 统计 > 0.05（> 0.1 更好）的模型。

计算结果脚本在这

https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/discussion/50827

# 结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 Roberta | Cv1 | lb | loss | Loss |
| 3epoch 4层输出 | 0.698385 | 0.797 | 0.3394 |  |
| 3epoch origin | 0.6949 | 0.789 | 0.338 |  |
| 3epoch 50eva | 0.698 | 0.784 | 0.341 |  |
| 3.6e | 0.6978 | 0.786 | 0.3414 |  |
| No dropout | 0.6982 | 0.807 | 0.3417 |  |
| Warm up | 0.6985 |  | 0.3437 |  |
| 3.6e | 0.6978 | 0.786 | 0.3414 |  |
| No dropout | 0.6982 | 0.807 | 0.3417 |  |
| Warm up | 0.6985 | 0.788 | 0.3437 |  |
| stolastcls | 0.6982 | 0.782 | 0.3421 |  |
| maxpooling | 0.6885 | 0.795 | 0.3488 |  |
| 5epoch 7fold | 0.6985 |  | 0.3434 |  |
| 4epoch 7fold | 0.6985 | 0.787 | 0.342 |  |
| 8e-6 7fold | 0.6975 | 0 | 0.3432 |  |
| 8e-6 7fold 无分层 | 0.6961 |  | 0.3409 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | Cv | Lb | Cv new |
| Tf\_idf+lgboost 0.5 0.5 |  | 0.795 |  |
| Tf\_idf+lgboost 0.85+0.15 | 0.690912 | 0.817 |  |
| Ensenmble of 3 | 0.69138 | 0.83 |  |
| Ensenmble 3 +最佳搜索 | 0.6932 | 0.812 |  |

数据集选择

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | Cv f2 | Lb f2 |  |
| Jc | 68.48 | 0.837 |  |
| Jc\_fr | 65.99 |  |  |
| Jc\_de | 65.88 |  |  |
| Jc\_es | 66.5 |  |  |
|  |  |  |  |
| Juc | 68.56 | 0.753 |  |
| Ruddit | 65.31 | 0.798 |  |
| jcc | 68.5 | 0.836 |  |
| Jc+translate | 68.91 | 0.823 |  |
|  |  |  |  |
| jc+juc+r+jcc g | 66.66 | 0.695 |  |
| jc+juc+r+jcc g三百迭代 | 67.78 | 0.727 |  |
| jc+juc+r+jcc （r+g） | 68.57 | 0.806 |  |
|  |  |  |  |
| Jc r jcc | 68.87 | 0.860 |  |
| Jc\_ 分值调整 | 66.25 | 0.836 |  |
| Jc\_ r jcc 分值调整 | 67.22 | 0.855 |  |
|  |  |  |  |
| Jc2 f 1fold 分值调整 不要测试集 | 67.05 | 0.868 |  |
| Jc\_2 rud jc\_ 1fold 分值调整 不要测试集 | 68.1048 | 0.873 |  |
| jc jc1 jc2 fr es de | 67.6465 | 0.836 |  |
|  |  |  |  |
| Jc\_ rud jc1\_ jc2 1:1 | 67.8391 | 0.870 | 68.2694 |
| Jc\_ rud jc1\_ jc2 1:1.5 | 68.2277 | 0.870 | 68.3773 |
| Jc1 jc2 jc3 rud 1:1.5 | 68.241 | 0.870 | 68.2694 |

结论：clean有效果，但是不能替代非clean。Juc 和gbm的分数都有点虚高。 Jc也存在虚高

虚高的可能原因：juc的数据偏旧 ？

Ruddit\_bert

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | Cv | Lb |  |
| Roberta 4epoch | 0.68619 | 0.775 |  |
| Roberta 4epoch 翻译 | 0.6855 | 0.761 |  |
| Roberta 3epoch | 0.6863 |  |  |
| Roberta 4epoch 10eva | 0.6818 | 0.768 |  |
| Hatebert 4epoch | 0.67012 | 0.724 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

集成模型

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | Cv f2 | Lb f2 |  |
| 0.85 0.15（0.86 0.831） |  | 0.867 |  |
| 0.5 0.5 （0.86 0.831） |  | 0.846 |  |
| 0.7 0.3 （0.86 0.831） |  | 0.861 |  |
| 0.8 0.2 （0.86 0.831） |  | 0.864 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 结果分析

使用未间隔的