wrangle_report

2018 年 12 月 25 日

0.1 收集

- 1)Udacity 的主页下载 WeRateDogs 推特档案 *twitter-archive-enhanced.csv*, 从 *twitter-archive-enhanced.csv* 导入为 archive_df。
- 2) 调用 request 包的 `get()` 函数，输入给定的 url 参数, 获得 response 对象。利用 `with open () as file` 句式创建 *image-predictions.tsv*, 并将 response 对象的内容写入 *image-predictions.tsv*，从 *image-predictions.tsv* 导入为 image_df。
- 3)Udacity 的主页下载 *tweet_json.txt*, 利用 `with open () as file` 句式打开 *tweet_json.txt*，并逐行添加到 `status` 列表中，从 `status` 列表导入为 tweet_df。

0.2 评估

0.2.1 目测评估

- 1) 目测评估 archive_df
- 2) 目测评估 image_df
- 3) 目测评估 tweet_df

0.2.2 编程评估

- 1) 检查 archive_df 的数据缺失、数据类型、行列个数
- 2) 检查 archive_df 的数据重复
- 3) 检查 image_df 的数据缺失、数据类型、行列个数
- 4) 检查 image_df 的数据重复
- 5) 检查 tweet_df 的数据缺失、数据类型、行列个数
- 6) 检查 tweet_df 的 id 列和 id_str 列的数据重复
- 7)# 检查 archive_df 的 rating_numerator 列、rating_denominator 列数据值分布

0.2.3 评估结果

质量

`archive_df` 表格

- tweet_id 是整型，而不是字符串型
- in_reply_to_status_id 列、in_reply_to_user_id 列缺失数据
- timestamp 是字符串型，而不是 datetime 型
- 依据项目动机要求，删除 timestamp 列数据值在 2017 年 8 月 1 日之后的行
- 依据项目动机要求，删除 retweeted_status_id 列或 retweeted_status_user_id 列或 retweeted_status_timestamp 列为非 nan 的数据行
- archive_df 的 *rating_numerator* 列值、*rating_denominator* 列值分别主要集中于 <14 和 <11。

`image_df` 表格

- tweet_id 是整型，而不是字符串型

`tweet_df` 表格

- id_str 的列名与 `archive_df` 表格的 tweet_id 列名、`image_df` 表格的 tweet_id 列名不一致
- created_at 列是字符串型，而不是 datetime 型
- 依据项目动机要求，删除 created_at 列数据值在 2017 年 8 月 1 日之后的行
- 依据项目动机要求，只保留 tweet_id 列、favorite_count 列、retweet_count 列

清洁度

- `archive_df` 表格四列表示一个个变量 (doggo、floofer、pupper、puppo)
- 依据项目要求，依据相同的 tweet_id，将 `tweet_df` 表格、`image_df` 表格、`archive_df` 表格合并为同一表格

0.2.4 数据清洗

- 1) 清洗数据前，保存副本
- 2) 使用 pandas 的 `.astype()` 函数，把 archive_df 的 *tweet_id* 一列数据类型转化为 `str`。
- 3) 使用 pandas 的 `.drop()` 函数，删除 archive_df 的 *in_reply_to_status_id* 列、*in_reply_to_user_id* 列。
- 4) 使用 pandas 的 `to_datetime()` 函数，将 archive_df 的 *timestamp* 列转换为 `datetime` 类型。
- 5) 使用 pandas 的 `query()` 函数，删除 archive_df 的 *timestamp* 列数据值在 2017 年 8 月 1 日之后的行。

- 6) 使用 series 的 `isnull()` 函数，保留 archive_df 的 *retweeted_status_id* 列、*retweeted_status_user_id* 列、*retweeted_status_timestamp* 列均为 nan 的行。然后运用 pandas 的 `drop()` 函数删除 *retweeted_status_id* 列、*retweeted_status_user_id* 列、*retweeted_status_timestamp* 列。

- 7)pandas 的赋值函数保留 rating_numerator 列值 <14 且 rating_denominator 列值 <11 的行。

- 8) 使用 pandas 的 `.astype()` 函数，把 image_df 的 *tweet_id* 一列数据类型转化为 `str`。

- 9) 使用 pandas 的 `.rename()` 函数，把 tweet_df 的 *id_str* 的列名改为为 `tweet_id`。

- 10) 使用 pandas 的 to_datetime() 函数，将 tweet_df 的 created_at 列转换为 datetime 类型

- 11) 使用 pandas 的 `query()` 函数，删除 tweet_df 的 *created_at* 列数据值在 2017 年 8 月 1 日之后的行。

- 12) 选择 tweet_df 的 tweet_id 列、favorite_count 列、retweet_count 列并重新赋值给 tweet_df。

- 13) 利用`.replace()` 函数将 archive_df 的 doggo、floofer、pupper、puppo 各列的 `None` 替代为 ‘，检查替换后结果。利用 `+` 运算符合并各列为 `1` 列，并命名为 stage，利用 str.extract() 函数利用正则表达式捕获任意非数字的字符，将上述替换后的 `stage` 列值修改为便于理解与可视化并检查替代结果。利用.fillna() 函数将 Nan 替换 None‘, 删除 doggo、floofer、pupper、puppo 各列。

- 14) 先利用`.merge()` 函数，依据 archive_df 的 `tweet_id` 将 *archive_cl*、*tweet_cl* 合并为 `dog_cl`, 之后利用`.merge()` 函数，依据 `tweet_id` 将 *dog_cl*、*image_cl* 合并。

- 15) 将清理后的数据集存储到 CSV 文件中，命名为 twitter_archive_master.csv

0.3 探索数据

0.3.1 问题

- 1 喜欢数和转发数和最高的前 5 种狗的品种?
- 2 评分数和最高的前 5 种狗的品种?
- 3 狗的地位中喜欢数和转发数和的排序?

0.3.2 数据整理

为 dog_cl 创建 count 列为 favorite_count 列和 retweet_count 列之和；为 dog_cl 创建 rating 列为 rating_numerator 列和 rating_denominator 列之比；保留 “rating’’ 列,“stage’’ 列,“count’’ 列,“p1’’ 列,“p1_conf’’ 列,“p1_dog’’ 列，并创建 dog_val。

0.3.3 数据可视化

- 1）导入可视化工具包 *matplot.pyplot*

问题 1 1) 喜欢数和转发数和最高的前 5 种狗的品种?

- 1) 打印统计结果
- 2) 可视化结果
- 3) 设置可视化图标签及标题

结论 1： 喜欢数和转发数和最高的前 5 种狗的品种从 到低依次为 golden_retriever、Labrador_retriever、Pembroke、Chihuahua、Samoyed。

问题 2 2) 评分数和最高的前 5 种狗的品种?

- 1) 打印统计结果
- 2) 可视化结果
- 3) 设置可视化图标签及标题

结论 2： 评分数和最高的前 5 种狗的品种从 到低依次为 bow_tie、golden_retriever、Labrador_retriever、Pembroke、Chihuahua。

问题 3 3) 狗的地位中喜欢数和转发数和的排序?

- 1) 打印统计结果
- 2) 可视化结果
- 3) 设置可视化图标签及标题

结论 3： 除去未标明狗的地位，标明狗地位中喜欢数和转发数和从 到低依次为 pupper、doggo、puppo、doggo|pupper、floofer、doggo|puppo、doggo|floofer。