Activity_Course 2 TikTok project lab

August 31, 2023

1 TikTok Project

Course 2 - Get Started with Python

Welcome to the TikTok Project!

You have just started as a data professional at TikTok.

The team is still in the early stages of the project. You have received notice that TikTok's leadership team has approved the project proposal. To gain clear insights to prepare for a claims classification model, TikTok's provided data must be examined to begin the process of exploratory data analysis (EDA).

A notebook was structured and prepared to help you in this project. Please complete the following questions.

2 Course 2 End-of-course project: Inspect and analyze data

In this activity, you will examine data provided and prepare it for analysis.

The purpose of this project is to investigate and understand the data provided. This activity will:

1. Acquaint you with the data
2. Compile summary information about the data
3. Begin the process of EDA and reveal insights contained in the data
4. Prepare you for more in-depth EDA, hypothesis testing, and statistical analysis

The goal is to construct a dataframe in Python, perform a cursory inspection of the provided dataset, and inform TikTok data team members of your findings. *This activity has three parts:*

Part 1: Understand the situation * How can you best prepare to understand and organize the provided TikTok information?

Part 2: Understand the data

- Create a pandas dataframe for data learning and future exploratory data analysis (EDA) and statistical activities
- Compile summary information about the data to inform next steps

Part 3: Understand the variables

- Use insights from your examination of the summary data to guide deeper investigation into variables

To complete the activity, follow the instructions and answer the questions below. Then, you will us your responses to these questions and the questions included in the Course 2 PACE Strategy Document to create an executive summary.

Be sure to complete this activity before moving on to Course 3. You can assess your work by comparing the results to a completed exemplar after completing the end-of-course project.

3 Identify data types and compile summary information

Throughout these project notebooks, you'll see references to the problem-solving framework PACE. The following notebook components are labeled with the respective PACE stage: Plan, Analyze, Construct, and Execute.

4 PACE stages

- `[Plan](#scrollTo=psz51YkZVwtN&line=3&uniqifier=1)`
- `[Analyze](#scrollTo=mA7Mz_SnI8km&line=4&uniqifier=1)`
- `[Construct](#scrollTo=Lca9c8XON8lc&line=2&uniqifier=1)`
- `[Execute](#scrollTo=401PgchTPr4E&line=2&uniqifier=1)`

4.1 PACE: Plan

Consider the questions in your PACE Strategy Document and those below to craft your response:

4.1.1 Task 1. Understand the situation

- How can you best prepare to understand and organize the provided information?

Begin by exploring your dataset and consider reviewing the Data Dictionary.

==> ENTER YOUR RESPONSE HERE

4.2 PACE: Analyze

Consider the questions in your PACE Strategy Document to reflect on the Analyze stage.

4.2.1 Task 2a. Imports and data loading

Start by importing the packages that you will need to load and explore the dataset. Make sure to use the following import statements: * `import pandas as pd`

- `import numpy as np`

[2]:
```python
# Import packages
import pandas as pd
import numpy as np
```

Then, load the dataset into a dataframe. Creating a dataframe will help you conduct data manipulation, exploratory data analysis (EDA), and statistical activities.

Note: As shown in this cell, the dataset has been automatically loaded in for you. You do not need to download the .csv file, or provide more code, in order to access the dataset and proceed with this lab. Please continue with this activity by completing the following instructions.

[3]:
```python
# Load dataset into dataframe
data = pd.read_csv("tiktok_dataset.csv")
```

4.2.2 Task 2b. Understand the data - Inspect the data

View and inspect summary information about the dataframe by **coding the following:**

1. `data.head(10)`
2. `data.info()`
3. `data.describe()`

Consider the following questions:

Question 1: When reviewing the first few rows of the dataframe, what do you observe about the data? What does each row represent?

Question 2: When reviewing the `data.info()` output, what do you notice about the different variables? Are there any null values? Are all of the variables numeric? Does anything else stand out?

Question 3: When reviewing the `data.describe()` output, what do you notice about the distributions of each variable? Are there any questionable values? Does it seem that there are outlier values?

[4]:
```python
# Display and examine the first ten rows of the dataframe
data.head(10)
```

[4]:
```
   # claim_status    video_id  video_duration_sec  \
0  1        claim  7017666017                  59
1  2        claim  4014381136                  32
2  3        claim  9859838091                  31
3  4        claim  1866847991                  25
4  5        claim  7105231098                  19
```

```
5   6        claim  8972200955                  35
6   7        claim  4958886992                  16
7   8        claim  2270982263                  41
8   9        claim  5235769692                  50
9  10        claim  4660861094                  45

                              video_transcription_text verified_status  \
0  someone shared with me that drone deliveries a…    not verified
1  someone shared with me that there are more mic…    not verified
2  someone shared with me that american industria…    not verified
3  someone shared with me that the metro of st. p…    not verified
4  someone shared with me that the number of busi…    not verified
5  someone shared with me that gross domestic pro…    not verified
6  someone shared with me that elvis presley has …    not verified
7  someone shared with me that the best selling s…    not verified
8  someone shared with me that about half of the …    not verified
9  someone shared with me that it would take a 50…        verified

  author_ban_status  video_view_count  video_like_count  video_share_count  \
0      under review          343296.0           19425.0              241.0
1            active          140877.0           77355.0            19034.0
2            active          902185.0           97690.0             2858.0
3            active          437506.0          239954.0            34812.0
4            active           56167.0           34987.0             4110.0
5      under review          336647.0          175546.0            62303.0
6            active          750345.0          486192.0           193911.0
7            active          547532.0            1072.0               50.0
8            active           24819.0           10160.0             1050.0
9            active          931587.0          171051.0            67739.0

   video_download_count  video_comment_count
0                   1.0                  0.0
1                1161.0                684.0
2                 833.0                329.0
3                1234.0                584.0
4                 547.0                152.0
5                4293.0               1857.0
6                8616.0               5446.0
7                  22.0                 11.0
8                  53.0                 27.0
9                4104.0               2540.0
```

[5]:

```
# Get summary info
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19382 entries, 0 to 19381
```

```
Data columns (total 12 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   #                       19382 non-null  int64
 1   claim_status            19084 non-null  object
 2   video_id                19382 non-null  int64
 3   video_duration_sec      19382 non-null  int64
 4   video_transcription_text 19084 non-null  object
 5   verified_status         19382 non-null  object
 6   author_ban_status       19382 non-null  object
 7   video_view_count        19084 non-null  float64
 8   video_like_count        19084 non-null  float64
 9   video_share_count       19084 non-null  float64
 10  video_download_count    19084 non-null  float64
 11  video_comment_count     19084 non-null  float64
dtypes: float64(5), int64(3), object(4)
memory usage: 1.8+ MB
```

[6]:

```
# Get summary statistics
data.describe()
```

[6]:

```
                  #      video_id  video_duration_sec  video_view_count  \
count  19382.000000  1.938200e+04        19382.000000      19084.000000
mean    9691.500000  5.627454e+09           32.421732     254708.558688
std     5595.245794  2.536440e+09           16.229967     322893.280814
min        1.000000  1.234959e+09            5.000000         20.000000
25%     4846.250000  3.430417e+09           18.000000       4942.500000
50%     9691.500000  5.618664e+09           32.000000       9954.500000
75%    14536.750000  7.843960e+09           47.000000     504327.000000
max    19382.000000  9.999873e+09           60.000000     999817.000000

       video_like_count  video_share_count  video_download_count  \
count      19084.000000       19084.000000          19084.000000
mean       84304.636030       16735.248323           1049.429627
std       133420.546814       32036.174350           2004.299894
min            0.000000           0.000000              0.000000
25%          810.750000         115.000000              7.000000
50%         3403.500000         717.000000             46.000000
75%       125020.000000       18222.000000           1156.250000
max       657830.000000      256130.000000          14994.000000

       video_comment_count
count         19084.000000
mean            349.312146
std             799.638865
min               0.000000
25%               1.000000
```

```
50%              9.000000
75%            292.000000
max           9599.000000
```

===> ENTER YOUR RESPONSE TO QUESTIONS 1-3 HERE

1. Each row contains data on a single TikTok video. video ID, duration, interaction counts of various types, the text transcription, and information about the creator/author.
2. From the summary info, we can see that there are a few null values, but they are somewhat consistent. There are 19,382 total entries in the table. Some columns have this many entries, but others have only 19,084. This likely means that there was a specific error in compiling the data, or there are some kinds of content that do not have certain types of information. Sorting by view count or status will likely help us get to the bottom of this.
3. It is worth noting that many of the stdev values are higher than the means. This makes sense for a platform that is based on sharing and viral properties of videos. The "highest performing" content will skew these distributions. Despite some of the values being large, I do not believe that they are necessarily excludable outliers. The describe() function also shows us that there are 298 videos that are missing all the numeric data, except that they exist.

4.2.3 Task 2c. Understand the data - Investigate the variables

In this phase, you will begin to investigate the variables more closely to better understand them.

You know from the project proposal that the ultimate objective is to use machine learning to classify videos as either claims or opinions. A good first step towards understanding the data might therefore be examining the `claim_status` variable. Begin by determining how many videos there are for each different claim status.

[7]:
```python
# What are the different values for claim status and how many of each are in the data?
print(data['claim_status'].value_counts().sum())
data['claim_status'].value_counts()
```

```
19084
```

[7]:
```
claim      9608
opinion    9476
Name: claim_status, dtype: int64
```

Question: What do you notice about the values shown? There are both claims and opinions, and the two types total 19,084, which is the number we expected. We understand the source of the discrepancy. Not all the videos in the data set are marked as having a claim or opinion.

Next, examine the engagement trends associated with each different claim status.

Start by using Boolean masking to filter the data according to claim status, then calculate the mean and median view counts for each claim status.

[8]:
```python
# What is the average view count of videos with "claim" status?
claim_mask = data['claim_status'] == 'claim'
print('Mean views for "claim" status:', data[claim_mask].
 ↪mean()['video_view_count'])
print('Median views for "claim" status:', data[claim_mask].
 ↪median()['video_view_count'])
```

```
Mean views for "claim" status: 501029.4527477102
Median views for "claim" status: 501555.0
```

[9]:
```python
# What is the average view count of videos with "opinion" status?
opinion_mask = data['claim_status'] == 'opinion'
print('Mean views for "opinion" status:', data[opinion_mask].
 ↪mean()['video_view_count'])
print('Median views for "opinion" status:', data[opinion_mask].
 ↪median()['video_view_count'])
```

```
Mean views for "opinion" status: 4956.43224989447
Median views for "opinion" status: 4953.0
```

Question: What do you notice about the mean and media within each claim category? In both of these statuses, the mean and median are very close together, which is reassuring from the standpoint about making assumptions of normal(ish) distribution. It is worth noting that the mean and median views for the claim status are two orders of magnitude larger than for opinions.

Now, examine trends associated with the ban status of the author.

Use `groupby()` to calculate how many videos there are for each combination of categories of claim status and author ban status.

[10]:
```python
# Get counts for each group combination of claim status and author ban status
data.groupby(['claim_status', 'author_ban_status']).sum()[['#']]
# I decided to limit this objec to just number for readability.
```

[10]:

claim_status	author_ban_status	#
claim	active	31692180
	banned	6769281
	under review	7700175
opinion	active	126440294
	banned	2858518
	under review	6648622

Question: What do you notice about the number of claims videos with banned authors? Why might this relationship occur? The number of claim videos with banned authors is the smallest of the claim categories. This makes sense, since an author can be banned for making false claims.

Continue investigating engagement levels, now focusing on `author_ban_status`.

Calculate the median video share count of each author ban status.

[11]:
```python
data.groupby(['claim_status', 'author_ban_status'])['video_share_count'].
↪median()
```

[11]:
```
claim_status  author_ban_status
claim         active               17774.5
              banned               19018.0
              under review         18084.0
opinion       active                 121.0
              banned                 108.5
              under review           124.0
Name: video_share_count, dtype: float64
```

[12]:
```python
# What's the median video share count of each author ban status?
data.groupby(['claim_status', 'author_ban_status'])['video_share_count'].
↪median()
```

[12]:
```
claim_status  author_ban_status
claim         active               17774.5
              banned               19018.0
              under review         18084.0
opinion       active                 121.0
              banned                 108.5
              under review           124.0
Name: video_share_count, dtype: float64
```

Question: What do you notice about the share count of banned authors, compared to that of active authors? Explore this in more depth. Banned authors appear to have a larger median video share count, which would indicate a propensity for the spread of disinformation.

Use `groupby()` to group the data by `author_ban_status`, then use `agg()` to get the count, mean, and median of each of the following columns: * `video_view_count` * `video_like_count` * `video_share_count`

Remember, the argument for the `agg()` function is a dictionary whose keys are columns. The values for each column are a list of the calculations you want to perform.

[28]:
```python
data.
↪groupby('author_ban_status')['video_view_count','video_like_count','video_share_count'].
↪agg(['count','mean','median'])
```

[28]:
```
                  video_view_count                           video_like_count  \
                             count           mean    median             count
author_ban_status
active                       15383  215927.039524    8616.0             15383
banned                        1635  445845.439144  448201.0              1635
under review                  2066  392204.836399  365245.5              2066

                                           video_share_count                \
```

```
                         mean     median             count          mean
author_ban_status
active              71036.533836    2222.0           15383  14111.466164
banned             153017.236697  105573.0            1635  29998.942508
under review       128718.050339   71204.5            2066  25774.696999


                    median
author_ban_status
active               437.0
banned             14468.0
under review        9444.0
```

Question: What do you notice about the number of views, likes, and shares for banned authors compared to active authors? There is way more interaction(views, likes, and shares) for each video from a banned author by a LARGE margin.

Now, create three new columns to help better understand engagement rates: * `likes_per_view`: represents the number of likes divided by the number of views for each video * `comments_per_view`: represents the number of comments divided by the number of views for each video * `shares_per_view`: represents the number of shares divided by the number of views for each video

[40]:
```python
# Create a likes_per_view column
data['likes_per_view'] = data['video_like_count'] / data['video_view_count']

# Create a comments_per_view column
data['comments_per_view'] = data['video_comment_count'] /␣
↪data['video_view_count']

# Create a shares_per_view column
data['shares_per_view'] = data['video_share_count'] / data['video_view_count']
```

Use `groupby()` to compile the information in each of the three newly created columns for each combination of categories of claim status and author ban status, then use `agg()` to calculate the count, the mean, and the median of each group.

[43]:
```python
### YOUR CODE HERE ###
data.groupby(['author_ban_status','claim_status'])['likes_per_view',␣
↪'comments_per_view', 'shares_per_view'].agg(['count','mean','median'])
```

[43]:
```
                                 likes_per_view                      \
                                          count      mean    median
author_ban_status claim_status
active            claim                    6566  0.329542  0.326538
                  opinion                  8817  0.219744  0.218330
banned            claim                    1439  0.345071  0.358909
                  opinion                   196  0.206868  0.198483
under review      claim                    1603  0.327997  0.320867
```

```
                  opinion                 463  0.226394  0.228051

                                 comments_per_view                      \
                                             count      mean    median
author_ban_status claim_status
active            claim                       6566  0.001393  0.000776
                  opinion                     8817  0.000517  0.000252
banned            claim                       1439  0.001377  0.000746
                  opinion                      196  0.000434  0.000193
under review      claim                       1603  0.001367  0.000789
                  opinion                      463  0.000536  0.000293

                                 shares_per_view
                                           count      mean    median
author_ban_status claim_status
active            claim                     6566  0.065456  0.049279
                  opinion                   8817  0.043729  0.032405
banned            claim                     1439  0.067893  0.051606
                  opinion                    196  0.040531  0.030728
under review      claim                     1603  0.065733  0.049967
                  opinion                    463  0.044472  0.035027
```

Question:

How does the data for claim videos and opinion videos compare or differ? Consider views, comments, likes, and shares.

It appears that banned authors enjoy more likes, comments, and shares per view, possibly by approximately 30%.

4.3 PACE: Construct

Note: The Construct stage does not apply to this workflow. The PACE framework can be adapted to fit the specific requirements of any project.

4.4 PACE: Execute

Consider the questions in your PACE Strategy Document and those below to craft your response.

4.4.1 Given your efforts, what can you summarize for Rosie Mae Bradshaw and the TikTok data team?

Note for Learners: Your answer should address TikTok's request for a summary that covers the following points:

- What percentage of the data is comprised of claims and what percentage is comprised of opinions?

- What factors correlate with a video's claim status?
- What factors correlate with a video's engagement level?

==> ENTER YOUR RESPONSE HERE

1. We can see that claims and opinions have approximately equal prevalence. While claims are slightly more common, the difference is not statistically significant(P=0.17).

2. There are many more 'claim' videos in banned and under review authors. This dos not inherently give us mroe information, since people are not banned for opinions, just false claims.

3. We see much higher engagement with claim videos, espeically from banned authors. This demonstrates the need for a strong system to combat misinformation.

Congratulations! You've completed this lab. However, you may not notice a green check mark next to this item on Coursera's platform. Please continue your progress regardless of the check mark. Just click on the "save" icon at the top of this notebook to ensure your work has been logged.