Exploring_Hacker_News_Posts

February 28, 2020

1 Exploring Hacker News

- In this project, we will be exploring posts on Hacker News, one of the most poppular News website in the world where users submits posts that are voted and commented on (similar to reddit).
- Hacker News is popular in Technology and start-up circles, and the posts that make it to the top of Hacker News gets hundreds of thousands of views as a result
- What we will be doing in this project is quite simple: we will be working with a dataset which originally had 300,000 rows of data but was then reduced down to 20,000. We only wanted to focus on posts that had comments/ feedback.
- We're specifically interested in posts whose titles begin with `AskHN` or show `ShowHN`
- From there we will be making comparisons on these posts by analyzing the following:
  - Do Ask HN or Show HN receive more comments on average?
  - Do posts created at a certain time receive more comments on average?

STEP ONE: - We will start off by importing the libraries we need and reading the datasets into a list of lists below:

[1]:
```
from csv import reader
opened_file = open('hacker_news.csv')
read_file = reader(opened_file)
hn = list(read_file)

# Printing out the first 5 rows of the dataset including the header
for row in hn[0:5]:
    print(row)
```

```
['id', 'title', 'url', 'num_points', 'num_comments', 'author', 'created_at']
['12224879', 'Interactive Dynamic Video',
'http://www.interactivedynamicvideo.com/', '386', '52', 'ne0phyte', '8/4/2016
11:52']
['10975351', 'How to Use Open Source and Shut the Fuck Up at the Same Time',
'http://hueniverse.com/2016/01/26/how-to-use-open-source-and-shut-the-fuck-up-
at-the-same-time/', '39', '10', 'josep2', '1/26/2016 19:30']
['11964716', "Florida DJs May Face Felony for April Fools' Water Joke",
'http://www.thewire.com/entertainment/2013/04/florida-djs-april-fools-water-
joke/63798/', '2', '1', 'vezycash', '6/23/2016 22:20']
['11919867', 'Technology ventures: From Idea to Enterprise',
'https://www.amazon.com/Technology-Ventures-Enterprise-Thomas-
```

```
Byers/dp/0073523429', '3', '1', 'hswarna', '6/17/2016 0:01']
```

STEP TWO: - Extract all headers from the dataset and save them in a seperate variable. - Update dataset without headers. Print first 5 rows of updated dataset to check.

[2]:
```
headers = hn[0] #Extracted headers from the dataset and saved them to variable␣
 ↪headers
hn = hn[1:] #Dataset without the headers
for row in hn[0:5]: #Displaying first 5 rows of updated dataset without the␣
 ↪headers
    print(row)
```

```
['12224879', 'Interactive Dynamic Video',
'http://www.interactivedynamicvideo.com/', '386', '52', 'ne0phyte', '8/4/2016
11:52']
['10975351', 'How to Use Open Source and Shut the Fuck Up at the Same Time',
'http://hueniverse.com/2016/01/26/how-to-use-open-source-and-shut-the-fuck-up-
at-the-same-time/', '39', '10', 'josep2', '1/26/2016 19:30']
['11964716', "Florida DJs May Face Felony for April Fools' Water Joke",
'http://www.thewire.com/entertainment/2013/04/florida-djs-april-fools-water-
joke/63798/', '2', '1', 'vezycash', '6/23/2016 22:20']
['11919867', 'Technology ventures: From Idea to Enterprise',
'https://www.amazon.com/Technology-Ventures-Enterprise-Thomas-
Byers/dp/0073523429', '3', '1', 'hswarna', '6/17/2016 0:01']
['10301696', 'Note by Note: The Making of Steinway L1037 (2007)',
'http://www.nytimes.com/2007/11/07/movies/07stein.html?_r=0', '8', '2',
'walterbell', '9/30/2015 4:12']
```

STEP THREE: - Seperate ask posts and show posts rows from other posts in their own seperate lists. Display first five rows of Ask posts and Show posts to check.

[3]:
```
#Create 3 empty lists: 1 for ask posts, 1 for show posts, and 1 for other posts
ask_posts = []
show_posts = []
other_posts = []

for row in hn:
    title = row[1]
    title_lc = title.lower()
    if title_lc.startswith('ask hn'): #Adding rows from dataset w/titles␣
 ↪starting with 'ask hn' to 'ask_posts' list
        ask_posts.append(row)
    elif title_lc.startswith('show hn'): #Adding rows from dataset w/titles␣
 ↪starting with 'show hn' to 'show_posts' list
        show_posts.append(row)
    else:
        other_posts.append(row)#Adding rows from dataset w/titles that have␣
 ↪other titles to 'other_posts' list
```

```
print('# Ask Posts:',len(ask_posts))
print('# Show Posts:', len(show_posts))
print('# Other Posts:', len(other_posts))
```

```
# Ask Posts: 1744
# Show Posts: 1162
# Other Posts: 17194
```

[4]:
```
#Displaying first 5 rows the lists we are focusing on in this project
print('First 5 Ask posts rows: ')
for row in ask_posts[0:5]:
    print(row)

print('First 5 Show posts rows:')
for row in show_posts[0:5]:
    print(row)
```

```
First 5 Ask posts rows:
['12296411', 'Ask HN: How to improve my personal website?', '', '2', '6',
'ahmedbaracat', '8/16/2016 9:55']
['10610020', 'Ask HN: Am I the only one outraged by Twitter shutting down share
counts?', '', '28', '29', 'tkfx', '11/22/2015 13:43']
['11610310', 'Ask HN: Aby recent changes to CSS that broke mobile?', '', '1',
'1', 'polskibus', '5/2/2016 10:14']
['12210105', 'Ask HN: Looking for Employee #3 How do I do it?', '', '1', '3',
'sph130', '8/2/2016 14:20']
['10394168', 'Ask HN: Someone offered to buy my browser extension from me. What
now?', '', '28', '17', 'roykolak', '10/15/2015 16:38']
First 5 Show posts rows:
['10627194', 'Show HN: Wio Link  ESP8266 Based Web of Things Hardware
Development Platform', 'https://iot.seeed.cc', '26', '22', 'kfihihc',
'11/25/2015 14:03']
['10646440', 'Show HN: Something pointless I made', 'http://dn.ht/picklecat/',
'747', '102', 'dhotson', '11/29/2015 22:46']
['11590768', 'Show HN: Shanhu.io, a programming playground powered by e8vm',
'https://shanhu.io', '1', '1', 'h8liu', '4/28/2016 18:05']
['12178806', 'Show HN: Webscope  Easy way for web developers to communicate with
Clients', 'http://webscopeapp.com', '3', '3', 'fastbrick', '7/28/2016 7:11']
['10872799', 'Show HN: GeoScreenshot  Easily test Geo-IP based web pages',
'https://www.geoscreenshot.com/', '1', '9', 'kpsychwave', '1/9/2016 20:45']
```

[5]:
```
total_ask_comments = 0
total_comments = len(ask_posts) + len(show_posts)
for row in ask_posts:
    num_comments = int(row[4])
    total_ask_comments += num_comments
```

```
avg_ask_comments = total_ask_comments / total_comments
print('Average ask comments:',avg_ask_comments)

total_show_comments = 0
for row in show_posts:
    num_comments = int(row[4])
    total_show_comments += num_comments

avg_show_comments = total_show_comments / total_comments
print('Average show comments: ',avg_show_comments)
```

```
Average ask comments: 8.424982794218858
Average show comments:  4.125258086717137
```

ANALYSIS - As you can see, ask posts receive more comments on average than show posts - The reason for this is because people generally respond to when a person is asking something on a post more than a person showing something. People in ask posts expect or need feedback on their ask posts because they need help with something.

STEP FOUR: - Since we have determined that Ask posts receive more comments than show posts on average, we'll be focusing on just these posts. - Next, we'll determine if ask posts created at a certain time are more likely to attract comments. We'll use the following steps to perform this analysis: - Calculate the amount of ask posts created in each hour of the day, along with the number of comments received. - Calculate the average number of comments ask posts receive by hour created.

[6]:
```
import datetime as dt
result_list = []

for row in ask_posts:
    created_at = row[6]
    num_comments = int(row[4])
    result_list.append((created_at, num_comments))

counts_by_hour = {}
comments_by_hour = {}

for row in result_list:
    date = row[0]
    date_dt = dt.datetime.strptime(date, "%m/%d/%Y %H:%M")
    date = date_dt
    hour = date.hour
    if hour not in counts_by_hour:
        counts_by_hour[hour] = 1
        comments_by_hour[hour] = row[1]
    else:
        counts_by_hour[hour] += 1
```

```
        comments_by_hour[hour] += row[1]
```

[7]:
```
avg_by_hour = []
for hour in comments_by_hour:
    avg_by_hour.append([hour, (comments_by_hour[hour] / counts_by_hour[hour])])
print(avg_by_hour)
```

```
[[9, 5.5777777777777775], [13, 14.741176470588234], [10, 13.440677966101696],
[14, 13.233644859813085], [16, 16.796296296296298], [23, 7.985294117647059],
[12, 9.41095890410959], [17, 11.46], [15, 38.5948275862069], [21,
16.009174311926607], [20, 21.525], [2, 23.810344827586206], [18,
13.20183486238532], [3, 7.796296296296297], [5, 10.08695652173913], [19, 10.8],
[1, 11.383333333333333], [22, 6.746478873239437], [8, 10.25], [4,
7.170212765957447], [0, 8.127272727272727], [6, 9.022727272727273], [7,
7.852941176470588], [11, 11.051724137931034]]
```

[8]:
```
swap_avg_by_hr = []
for row in avg_by_hour:
    swap_avg_by_hr.append((row[1], row[0]))

print(swap_avg_by_hr)

sorted_swap = sorted(swap_avg_by_hr, reverse = True)
```

```
[(5.5777777777777775, 9), (14.741176470588234, 13), (13.440677966101696, 10),
(13.233644859813085, 14), (16.796296296296298, 16), (7.985294117647059, 23),
(9.41095890410959, 12), (11.46, 17), (38.5948275862069, 15),
(16.009174311926607, 21), (21.525, 20), (23.810344827586206, 2),
(13.20183486238532, 18), (7.796296296296297, 3), (10.08695652173913, 5), (10.8,
19), (11.383333333333333, 1), (6.746478873239437, 22), (10.25, 8),
(7.170212765957447, 4), (8.127272727272727, 0), (9.022727272727273, 6),
(7.852941176470588, 7), (11.051724137931034, 11)]
```

[41]:
```
print("Top 5 Hours of Ask Posts Comments")
print(sorted_swap[0:6])
```

```
Top 5 Hours of Ask Posts Comments
[(38.5948275862069, 15), (23.810344827586206, 2), (21.525, 20),
(16.796296296296298, 16), (16.009174311926607, 21), (14.741176470588234, 13)]
```

STEP FIVE: - Now that we have our list sorted in order of average comments per hour, it's time to format our hours into a desired format and round our average comments per hour to 2 decimal places for each row in our sorted list. - First we loop through each row in our sorted list and create a `datetime` object for each hour in our list and then extract the hours and minutes. -Then, we round our average comments per hour to 2 decimal places. -Lastly, we combine our hours and minutes with our rounded average comments per hour in the following string format: `15:00: 38.59 average comments per post`

[40]:
```
for row in sorted_swap:
    hour = row[1]
    avg_comments_by_hr = row[0]
    dt_hr = dt.datetime.strptime(str(hour), "%H")
    dt_hr = dt_hr.time()
    hour = dt_hr.hour
    minute = dt_hr.minute
    hrs_minutes_only = str(hour) + ":"+str(minute)+"0"
    rounded_avg = ": {:.2f} average comments per post".
↪format(avg_comments_by_hr)
    print(hrs_minutes_only + rounded_avg)
```

```
15:00: 38.59 average comments per post
2:00: 23.81 average comments per post
20:00: 21.52 average comments per post
16:00: 16.80 average comments per post
21:00: 16.01 average comments per post
13:00: 14.74 average comments per post
10:00: 13.44 average comments per post
14:00: 13.23 average comments per post
18:00: 13.20 average comments per post
17:00: 11.46 average comments per post
1:00: 11.38 average comments per post
11:00: 11.05 average comments per post
19:00: 10.80 average comments per post
8:00: 10.25 average comments per post
5:00: 10.09 average comments per post
12:00: 9.41 average comments per post
6:00: 9.02 average comments per post
0:00: 8.13 average comments per post
23:00: 7.99 average comments per post
7:00: 7.85 average comments per post
3:00: 7.80 average comments per post
4:00: 7.17 average comments per post
22:00: 6.75 average comments per post
9:00: 5.58 average comments per post
```

FINAL ANALYSIS - As you can see, the 5 hours that contain the most comments on average are: 3pm, 2am, 8pm, 4pm, and 9pm. - 3pm: 38.59 avg comments per post, 2am: 23.81 avg comments per post, 8pm: 21.52 avg comments per post, 4pm: 16.01 avg comments per post, and at 9pm: 16.01 average comments per post. - Between the hours of 1pm and 4pm are usually the most common study/class hours for tech students, and between the hours of 8pm and 2pm students are homework hours typically because typically that's when students are working on homework. -At 2pm, Tech students are commonly forced to do their homework assignments at those late hours because of late shifts, etc.

[ ]: