Exploring Hackers News Posts

In this project, we'll compare two different types of posts from Hacker News, a popular site where technology related stories (or 'posts') are voted and commented upon. The two types of posts we'll explore begin with either Ask HN or Show HN.

Users submit Ask HN posts to ask the Hacker News community a specific question, such as "What is the best online course you've ever taken?" Likewise, users submit Show HN posts to show the Hacker News community a project, product, or just generally something interesting.

We'll specifically compare these two types of posts to determine the following:

```
1 . Do Ask HN or Show HN receive more comments on average?
2 . Do posts created at a certain time receive more comments on
    average?
```

It should be noted that the data set we're working with was reduced from almost 300,000 rows to approximately 20,000 rows by removing all submissions that did not receive any comments, and then randomly sampling from the remaining submissions. Introduction

First, we'll read in the data and remove the headers.

Introduction.

First, we'll read in the data and remove the headers.

In [41]:
```
from csv import reader

opened_file = open('HN_posts_year_to_Sep_26_2016.csv')
```

```
read_file= reader(opened_file)
hn = list(read_file)
hn[:5]
# Removing the headers.
headers = hn[0]
hn = hn[1:]
print(headers)
print(hn[:5])
```

```
['id', 'title', 'url', 'num_points', 'num_comments', 'author', 'created
_at']
[['12579008', 'You have two days to comment if you want stem cells to b
e classified as your own', 'http://www.regulations.gov/document?D=FDA-2
015-D-3719-0018', '1', '0', 'altstar', '9/26/2016 3:26'], ['12579005',
'SQLAR  the SQLite Archiver', 'https://www.sqlite.org/sqlar/doc/trunk/R
EADME.md', '1', '0', 'blacksqr', '9/26/2016 3:24'], ['12578997', 'What
if we just printed a flatscreen television on the side of our boxes?',
'https://medium.com/vanmoof/our-secrets-out-f21c1f03fdc8#.ietxmez43',
'1', '0', 'pavel_lishin', '9/26/2016 3:19'], ['12578989', 'algorithmic
music', 'http://cacm.acm.org/magazines/2011/7/109891-algorithmic-compos
ition/fulltext', '1', '0', 'poindontcare', '9/26/2016 3:16'], ['1257897
9', 'How the Data Vault Enables the Next-Gen Data Warehouse and Data La
ke', 'https://www.talend.com/blog/2016/05/12/talend-and-Â\x93the-data-v
aultÂ\x94', '1', '0', 'markgainor1', '9/26/2016 3:14']]
```

In [29]:

```
def explore_data(dataset, start, end, rows_and_columns=False):
    dataset_slice = dataset[start:end]
    for row in dataset_slice:
        print(row)
        print('\n') # adds a new (empty) line between rows

    if rows_and_columns:
        print('Number of rows:', len(dataset))
        print('Number of columns:', len(dataset[0]))


explore_data(hn, 0, 3, True)
```

```
['12579008', 'You have two days to comment if you want stem cells to be
```

```
classified as your own', 'http://www.regulations.gov/document?D=FDA-201
5-D-3719-0018', '1', '0', 'altstar', '9/26/2016 3:26']


['12579005', 'SQLAR  the SQLite Archiver', 'https://www.sqlite.org/sqla
r/doc/trunk/README.md', '1', '0', 'blacksqr', '9/26/2016 3:24']


['12578997', 'What if we just printed a flatscreen television on the si
de of our boxes?', 'https://medium.com/vanmoof/our-secrets-out-f21c1f03
fdc8#.ietxmez43', '1', '0', 'pavel_lishin', '9/26/2016 3:19']


Number of rows: 293119
Number of columns: 7
```

In [42]:

```python
# Identifying posts that begin with either `Ask HN` or `Show HN` and se
parate the data into different lists.
ask_posts = []
show_posts =[]
other_posts = []

for post in hn:
    title = post[1]
    if title.lower().startswith("ask hn"):
        ask_posts.append(post)
    elif title.lower().startswith("show hn"):
        show_posts.append(post)
    else:
        other_posts.append(post)

print(len(ask_posts))
print(len(show_posts))
print(len(other_posts))
```

```
9139
10158
273822
```

Calculating the average number of comments `Ask HN` posts receive.

Now that we separated ask posts and show posts into different lists, we'll calculate the average number of comments each type of post receives.

In [43]:
```
total_ask_comments = 0

for post in ask_posts:
    total_ask_comments += int(post[4])

avg_ask_comments = total_ask_comments / len(ask_posts)
print(avg_ask_comments)

print("The average number of comments in ask_posts is:",avg_ask_comment
s)
```

```
10.393478498741656
The average number of comments in ask_posts is: 10.393478498741656
```

In [44]:
```
total_show_comments = 0

for post in show_posts:
    total_show_comments += int(post[4])

avg_show_comments = total_show_comments / len(show_posts)
print(avg_show_comments)

print("The average number of comment in show_posts is:",avg_show_commen
ts)
```

```
4.886099625910612
The average number of comment in show_posts is: 4.886099625910612
```

On average, ask_posts in our sample receive roughly 10 comments, whereas show_posts receive approximately 5. Since ask _posts are more likely to receive comments, we'll focus our remaining analysis just on these posts.

Finding the Amount of Ask Posts and Comments by Hour Created

Next, we'll determine if we can maximize the amount of comments an ask_post receives by creating it at a certain time. First, should find the amount of ask_posts created during each hour of day, along with the number of comments those posts received. Then, calculating the average amount of comments ask_posts created at each hour of the day receive.

In [53]:
```
import datetime as dt

result_list = []

for post in ask_posts:
    result_list.append(
        [post[6], int(post[4])]
    )
print(result_list[:20])
```

```
[['9/26/2016 2:53', 7], ['9/26/2016 1:17', 3], ['9/25/2016 22:57', 0],
['9/25/2016 22:48', 3], ['9/25/2016 21:50', 2], ['9/25/2016 19:30', 1],
['9/25/2016 19:22', 22], ['9/25/2016 17:55', 3], ['9/25/2016 15:48',
0], ['9/25/2016 15:35', 13], ['9/25/2016 15:28', 0], ['9/25/2016 14:4
3', 0], ['9/25/2016 14:17', 3], ['9/25/2016 13:08', 2], ['9/25/2016 11:
27', 2], ['9/25/2016 10:51', 0], ['9/25/2016 10:47', 6], ['9/25/2016 9:
04', 97], ['9/25/2016 7:09', 4], ['9/25/2016 3:00', 1]]
```

In [54]:
```
comments_by_hour = {}
counts_by_hour = {}
date_format = "%m/%d/%Y %H:%M"

for each_row in result_list:
    date = each_row[0]
    comment = each_row[1]
```

```
    time = dt.datetime.strptime(date, date_format).strftime("%H")
    if time in counts_by_hour:
        comments_by_hour[time] += comment
        counts_by_hour[time] += 1

    else:
        comments_by_hour[time] = comment
        counts_by_hour[time] = 1



counts_by_hour
```

Out[54]:
```
{'02': 269,
 '01': 282,
 '22': 383,
 '21': 518,
 '19': 552,
 '17': 587,
 '15': 646,
 '14': 513,
 '13': 444,
 '11': 312,
 '10': 282,
 '09': 222,
 '07': 226,
 '03': 271,
 '23': 343,
 '20': 510,
 '16': 579,
 '08': 257,
 '00': 301,
 '18': 614,
 '12': 342,
 '04': 243,
 '06': 234,
 '05': 209}
```

Accounting the Average Number of Comments for Ask HN Posts by Hour

In [55]:
```
avg_by_hour=[]
for hour in comments_by_hour:
    avg_by_hour.append([hour, comments_by_hour[hour]/ counts_by_hour[ho
ur]])
avg_by_hour
```

Out[55]:
```
[['02', 11.137546468401487],
 ['01', 7.407801418439717],
 ['22', 8.804177545691905],
 ['21', 8.687258687258687],
 ['19', 7.163043478260869],
 ['17', 9.449744463373083],
 ['15', 28.676470588235293],
 ['14', 9.692007797270955],
 ['13', 16.31756756756757],
 ['11', 8.96474358974359],
 ['10', 10.684397163120567],
 ['09', 6.653153153153153],
 ['07', 7.013274336283186],
 ['03', 7.948339483394834],
 ['23', 6.696793002915452],
 ['20', 8.749019607843136],
 ['16', 7.713298791018998],
 ['08', 9.190661478599221],
 ['00', 7.5647840531561465],
 ['18', 7.94299674267101],
 ['12', 12.380116959064328],
 ['04', 9.7119341563786],
 ['06', 6.782051282051282],
 ['05', 8.794258373205741]]
```

Sorting and Printing Values from a List of Lists, so need to revers values in the lists.

In [56]:
```
swap_avg_by_hour = []

for row in avg_by_hour:
    swap_avg_by_hour.append([row[1], row[0]])

print(swap_avg_by_hour)

sorted_swap = sorted(swap_avg_by_hour, reverse=True)

sorted_swap
print("**********************************************************************
*********")

# Print the string "Top 5 Hours for Ask Posts Comments"
print(max(swap_avg_by_hour))
print(min(swap_avg_by_hour))

for avr, hr in swap_avg_by_hour[:5]:
    date=dt.datetime.strptime(hr,"%H").strftime("%H:%M")
    print("{}:   {:.2f}  The highest average comments per Hour".format(
date,avr))
```

```
[[11.137546468401487, '02'], [7.407801418439717, '01'], [8.804177545691
905, '22'], [8.687258687258687, '21'], [7.163043478260869, '19'], [9.44
9744463373083, '17'], [28.676470588235293, '15'], [9.692007797270955,
'14'], [16.31756756756757, '13'], [8.96474358974359, '11'], [10.6843971
63120567, '10'], [6.653153153153153, '09'], [7.013274336283186, '07'],
[7.948339483394834, '03'], [6.696793002915452, '23'], [8.74901960784313
6, '20'], [7.713298791018998, '16'], [9.190661478599221, '08'], [7.5647
840531561465, '00'], [7.94299674267101, '18'], [12.380116959064328, '1
2'], [9.7119341563786, '04'], [6.782051282051282, '06'], [8.79425837320
5741, '05']]
**********************************************************************
**
```

```
[28.676470588235293, '15']

[6.653153153153153, '09']
02:00:   11.14  The highest average comments per Hour
01:00:   7.41  The highest average comments per Hour
22:00:   8.80  The highest average comments per Hour
21:00:   8.69  The highest average comments per Hour
19:00:   7.16  The highest average comments per Hour
```

The results show that hour 02:00 recievd the most comments as acconted by 11.14 which followed by hour 01:00 as second receiving comments.

Conclusion

In this project, the analyzing of the ask_posts and show_posts to locate which type of post and time receive the most comments on average. Based on the data analysing, to maximize the amount of comments a post receives, we'd recommend the post be categorized as ask post and created between 15:00 and 16:00 (3:00 pm est - 4:00 pm est).

However, it should be noted that the data set we analyzed excluded posts without any comments. Given that, it's more accurate to say that of the posts that received comments, ask posts received more comments on average and ask posts created between 01:00 and 22:00 (1:00 am est - 10:00 pm est) received the most comments on average.

However, it should be noted that the data set we analyzed excluded posts without any comments. Given that, it's more accurate to say that of the posts that received comments, ask posts received more comments on average and ask_posts created between 01:00 and 02:00 (1:00 am est - 2:00 am est) received the most comments on average.

In [ ]: