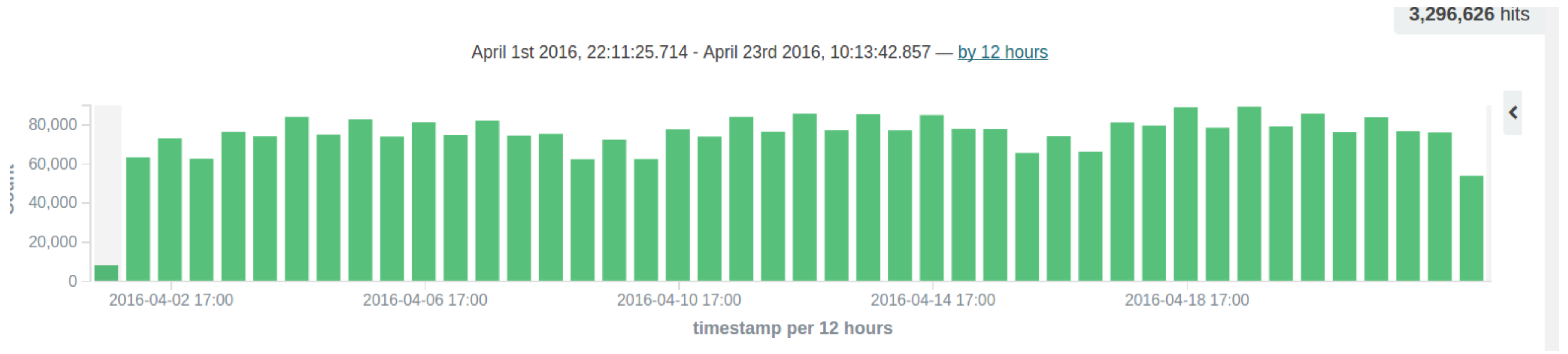
Big Data Project

Reddit Analysis

Outline

- Introduction
- Factors
  - Time of post
  - Subreddits
  - External Source
  - Controversiality of Comments
  - Comment Sentiment
- Conclusion

Time of Data Collection



Total Posts Count

3,449,206

Count

Average Score

36.041

Average score

Popular:

Score > 36

105,112

Count

Unpopular:

Score < 36

3,065,911

Count

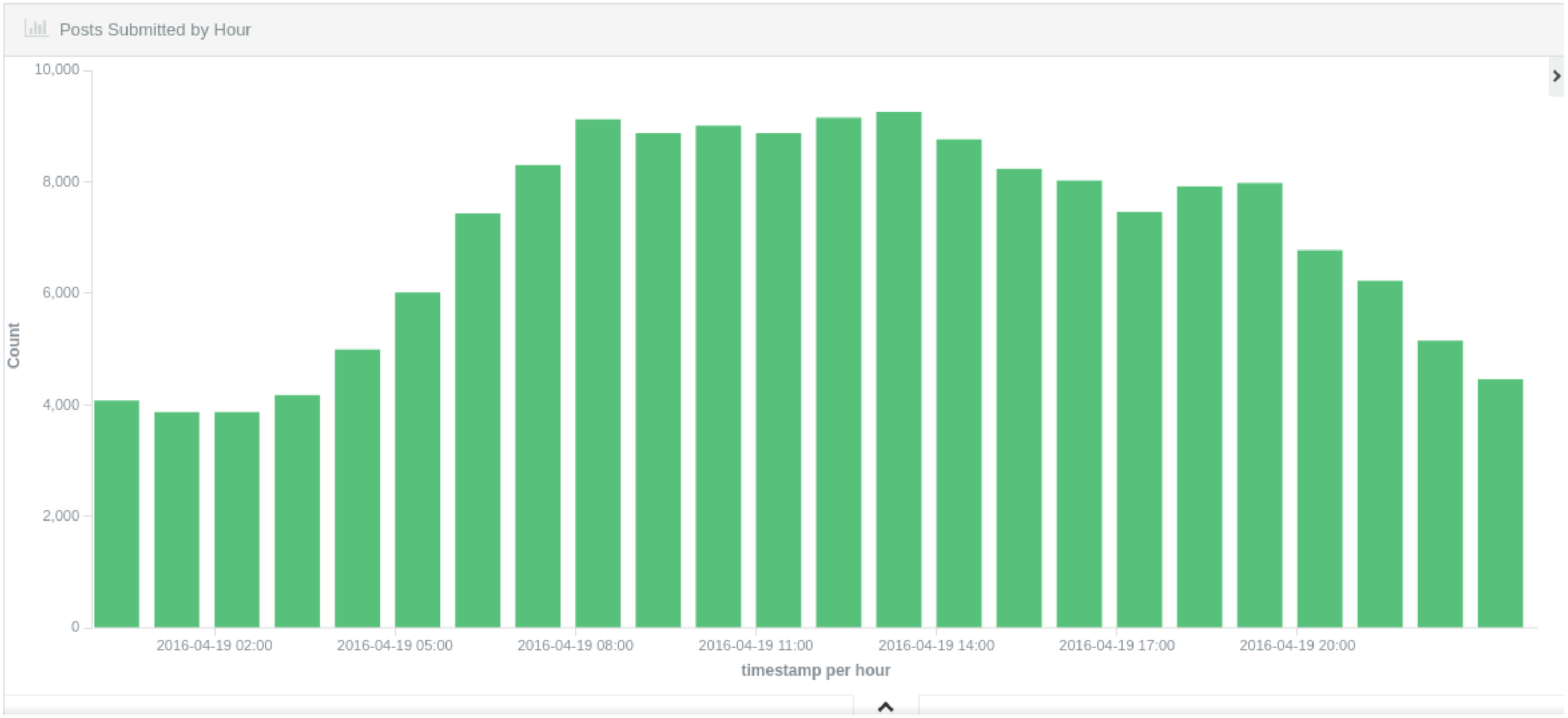
Unpopular:

Score < 36 AND comment count > 5

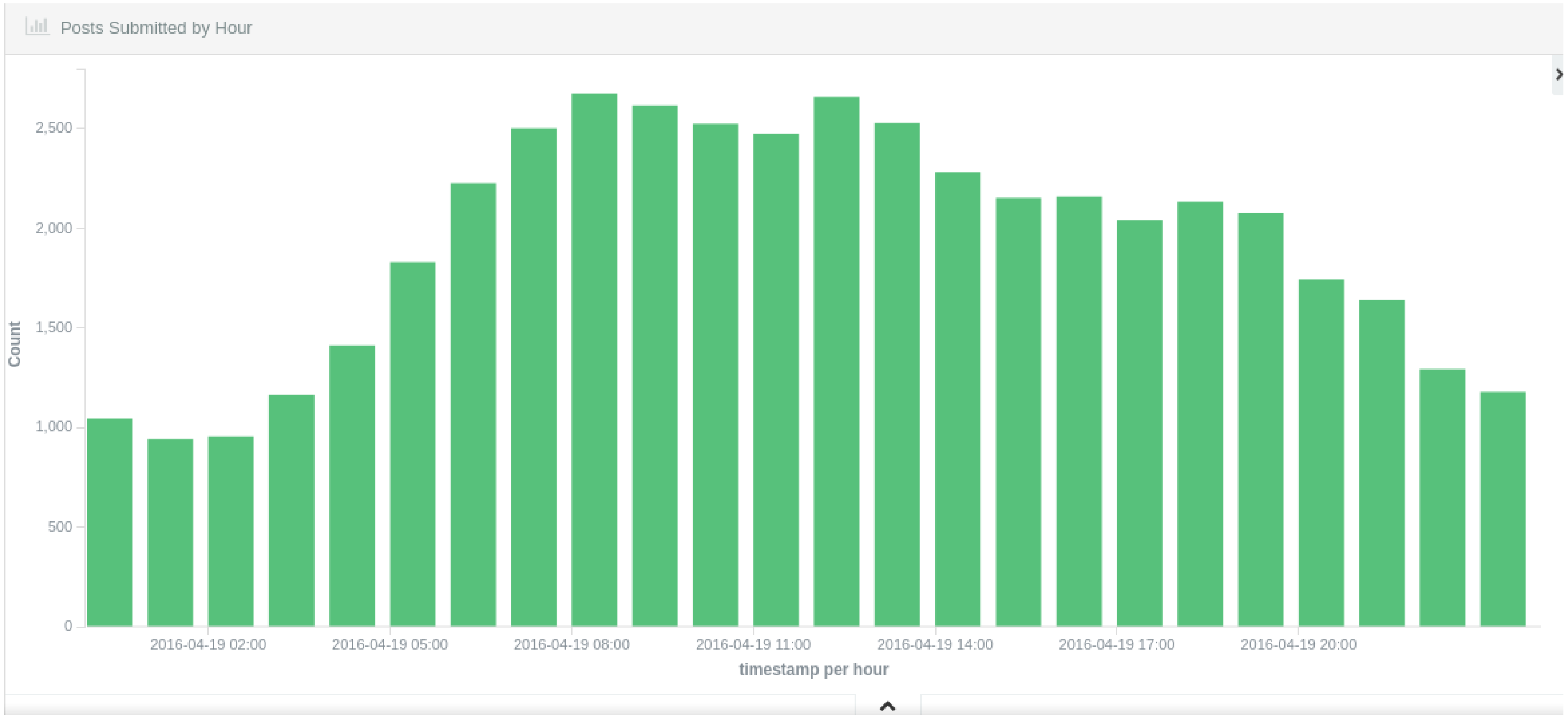
744,716

Count

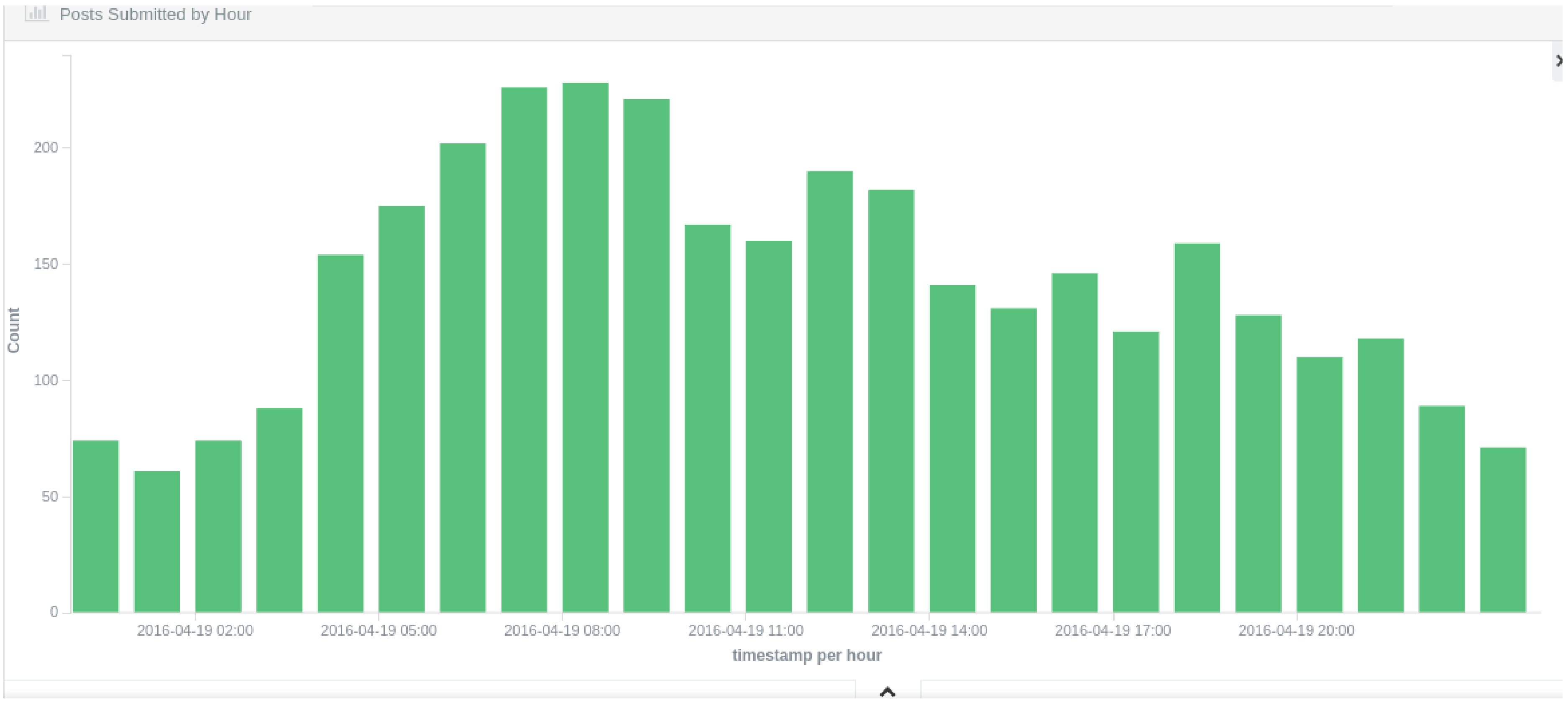
Timestamp


Posts Submitted by Hour
10,000
8,000
6,000
4,000
2,000
0
Count
2016-04-19 02:00
2016-04-19 05:00
2016-04-19 08:00
2016-04-19 11:00
2016-04-19 14:00
2016-04-19 17:00
2016-04-19 20:00
timestamp per hour

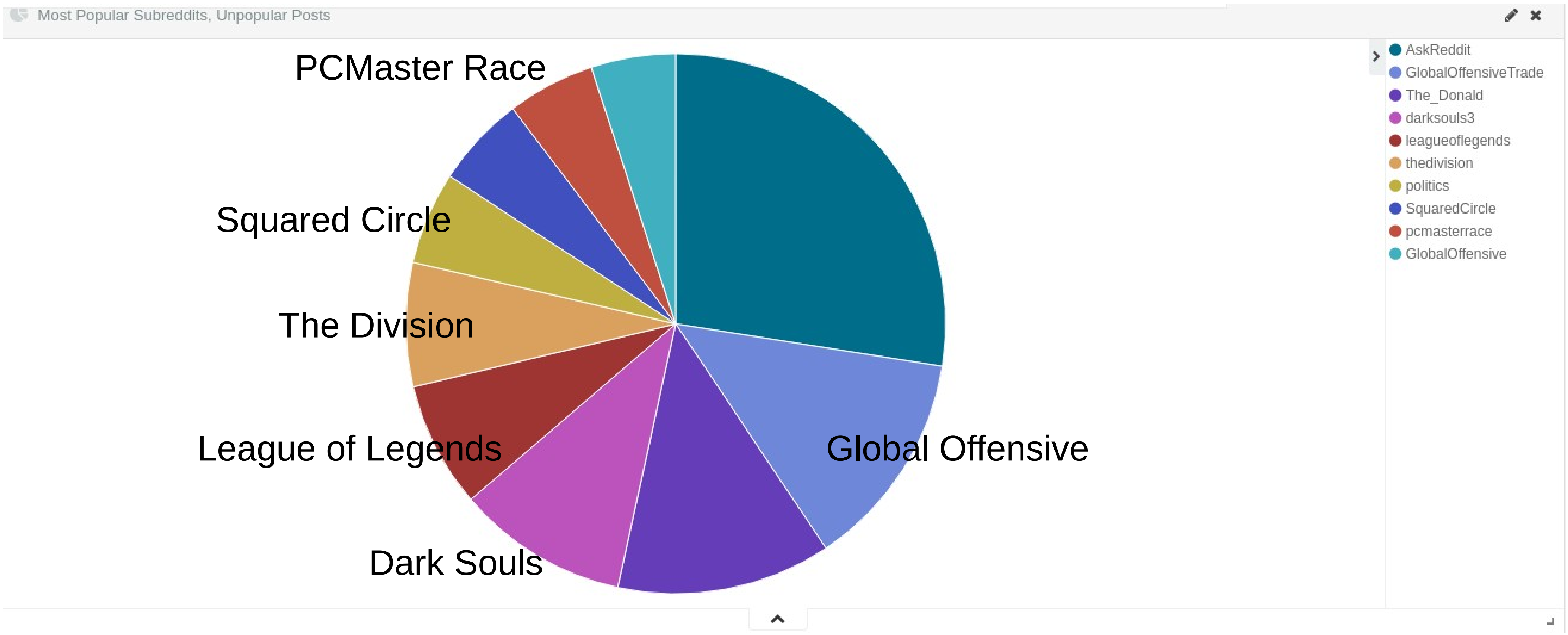
Timestamp - Unpopular



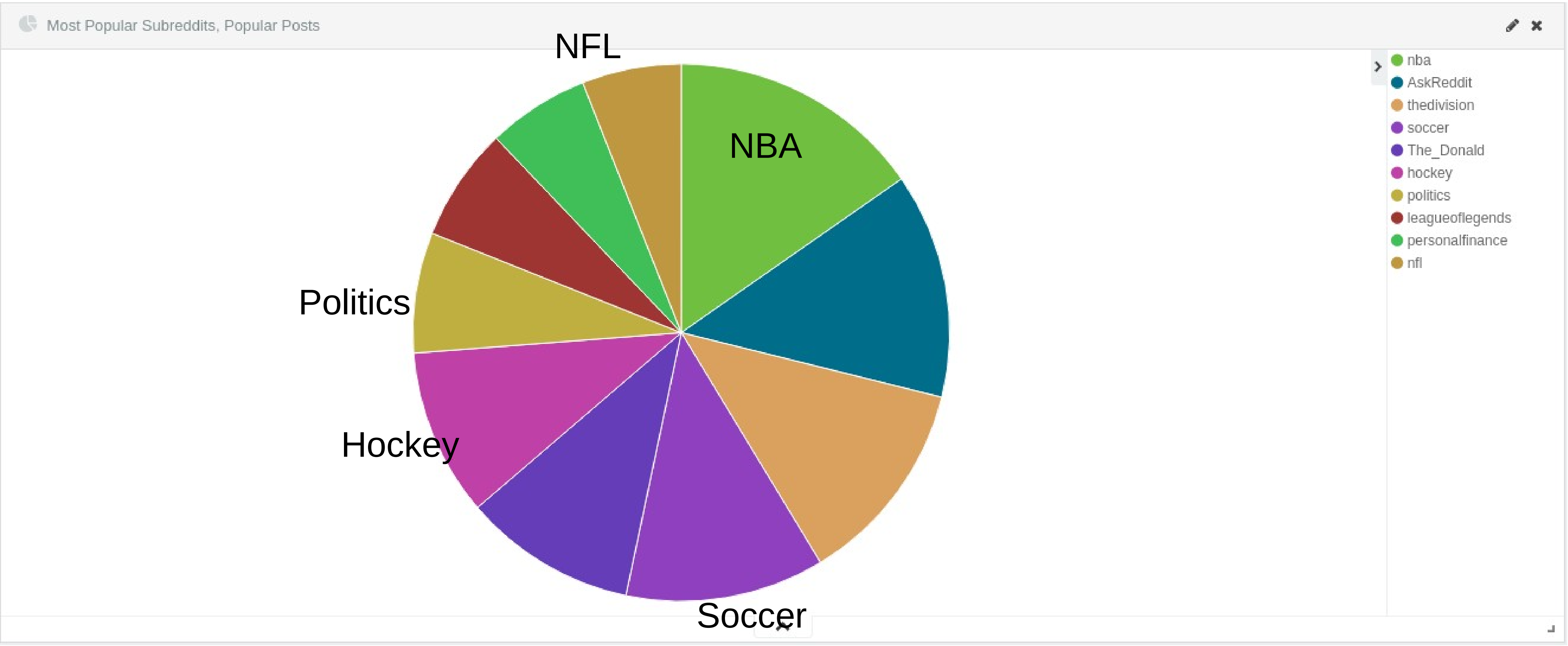
Timestamp - Popular



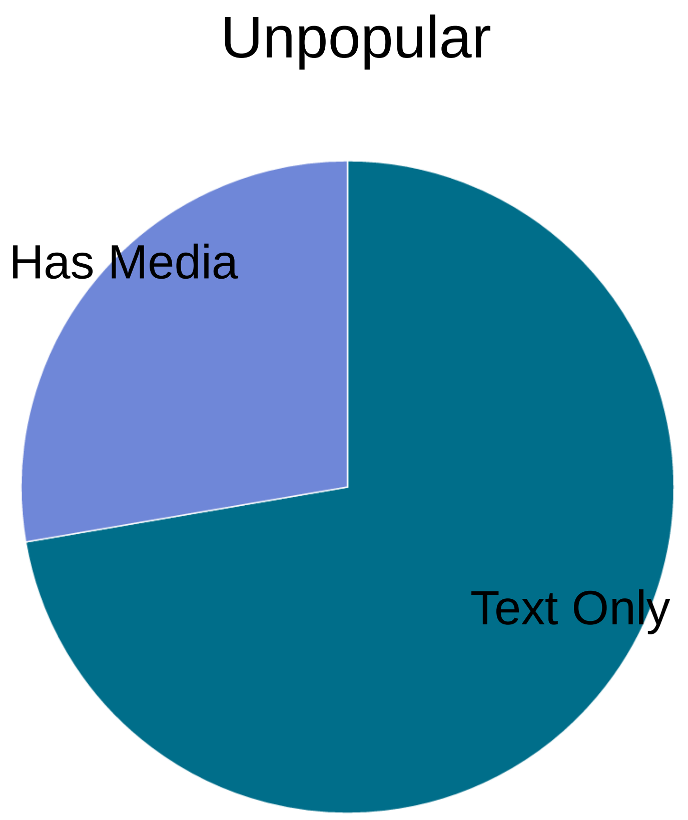
Subreddit - Unpopular

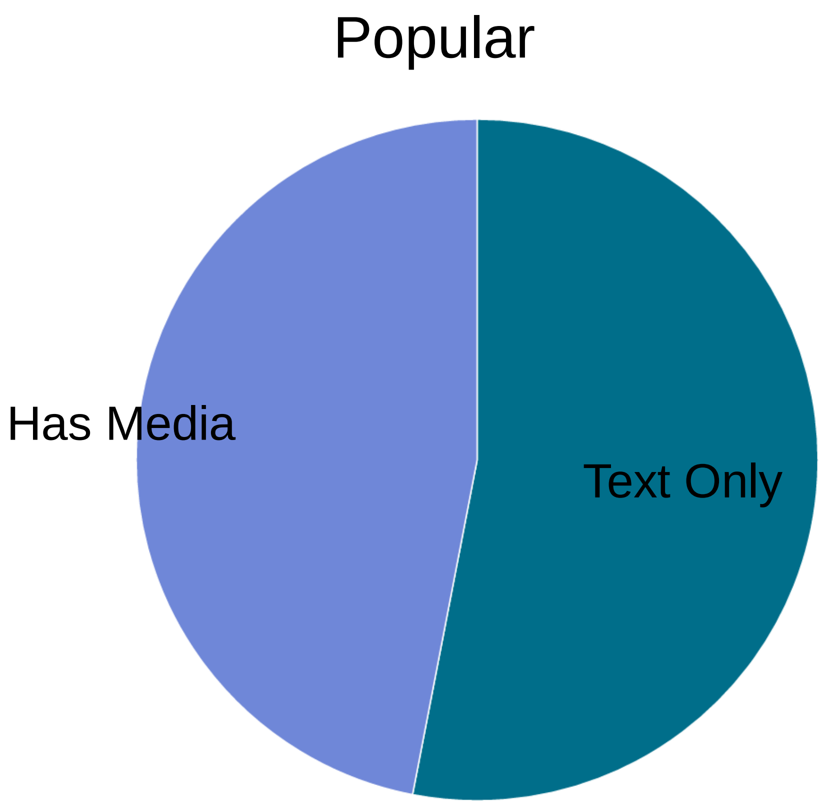


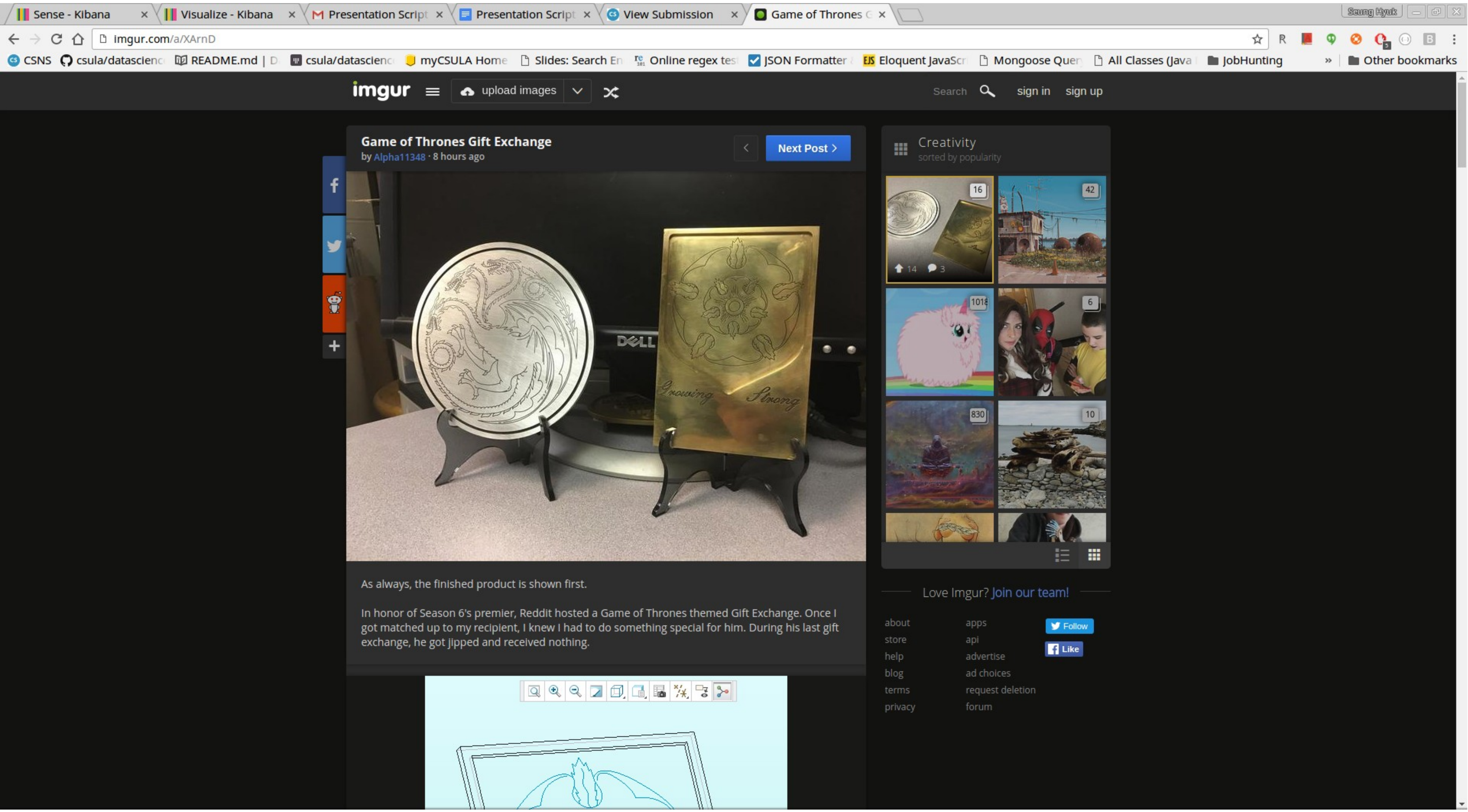
Subreddit - Popular



Has External Link





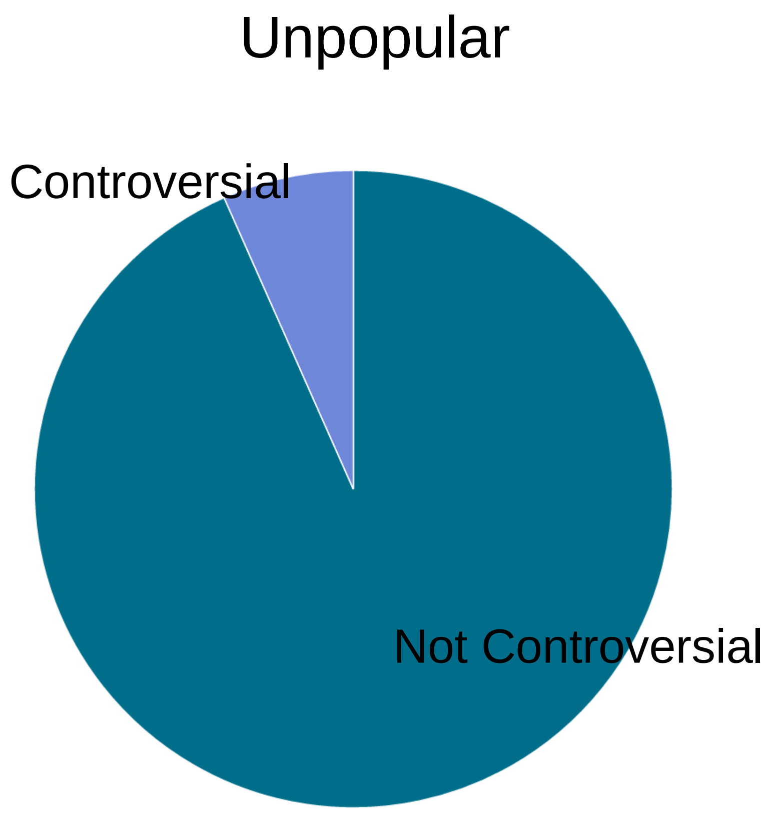


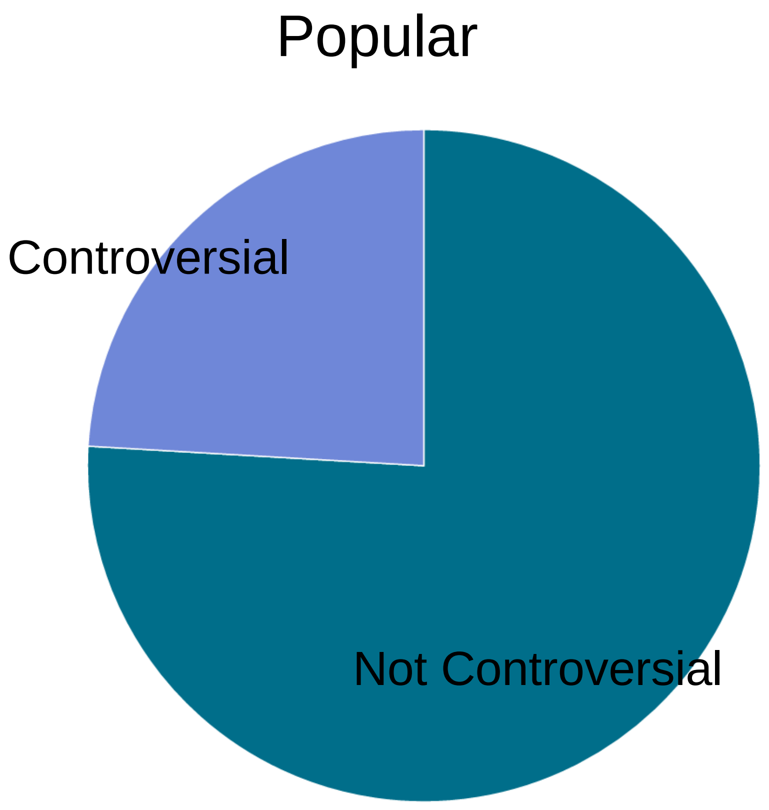
Game of Thrones Gift Exchange
by Alpha11348 · 8 hours ago

As always, the finished product is shown first.

In honor of Season 6's premier, Reddit hosted a Game of Thrones themed Gift Exchange. Once I got matched up to my recipient, I knew I had to do something special for him. During his last gift exchange, he got jipped and received nothing.

Controversial Comments

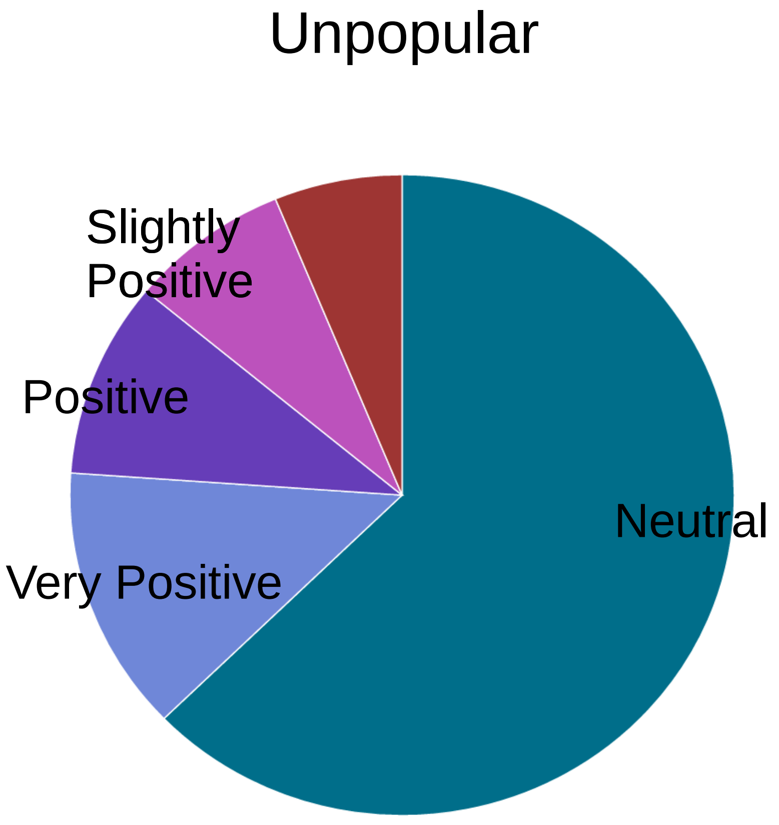


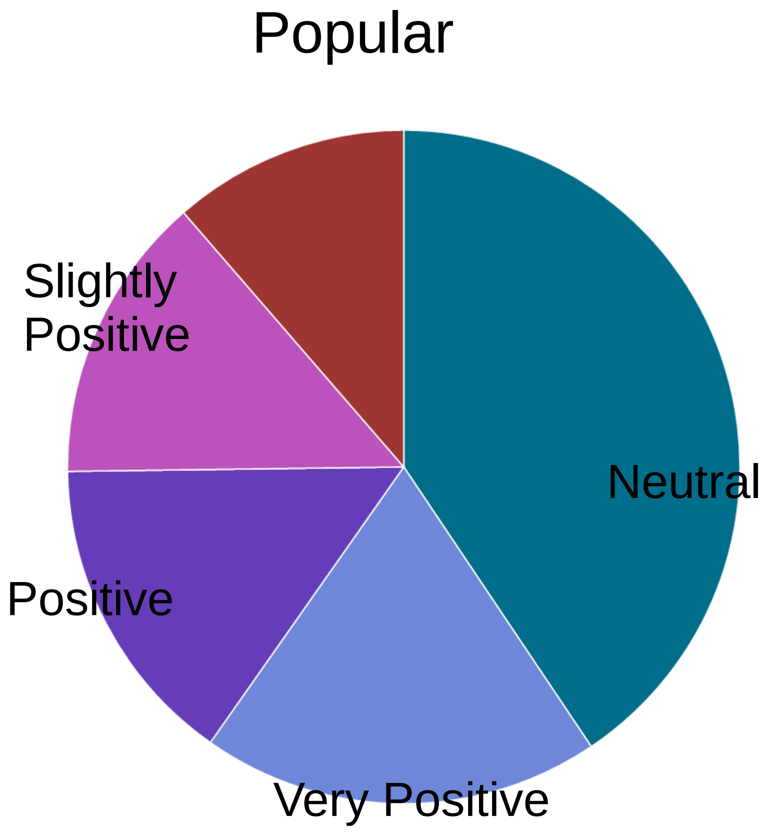


Controversial:

# upvote = # downvote

Sentiment of Comments

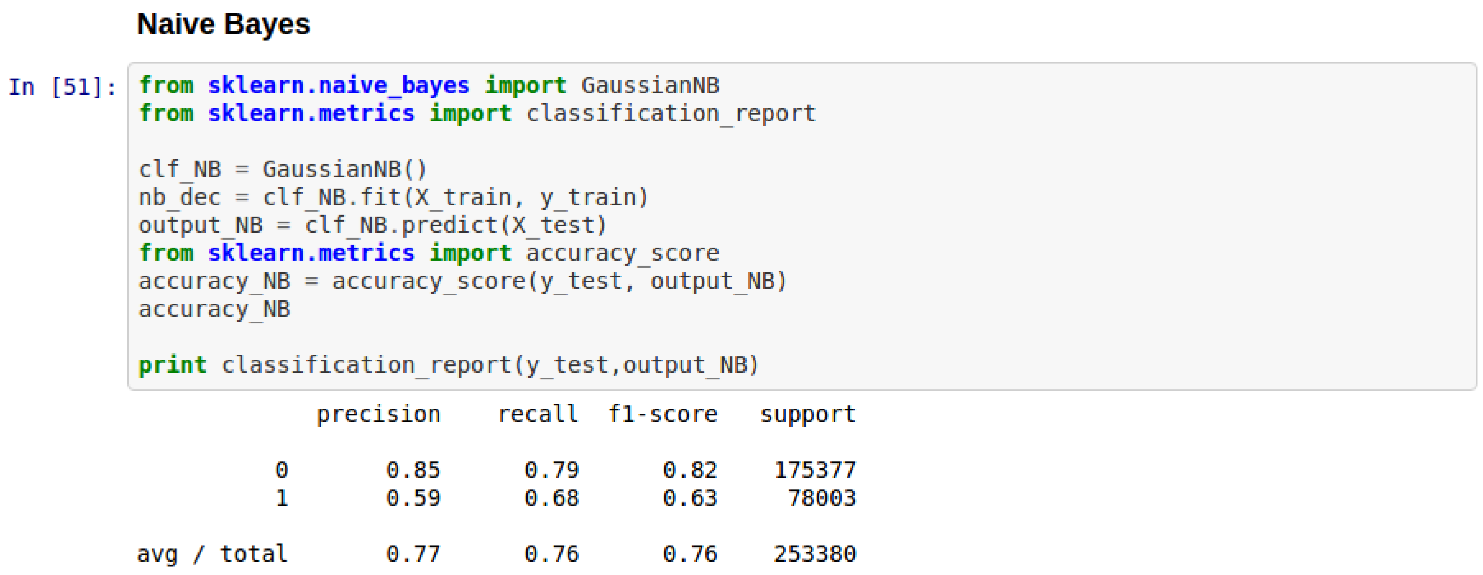




Machine Learning Prediction

report link: https://github.com/csula-students/beautiful-data-project-victorious-secret/blob/project/python/Reddit_Analysis.ipynb

- Naive Bayes
  - Precision for unpopular: 85%
  - Precision for popular: 59%
  - Precision Accuracy: 77%



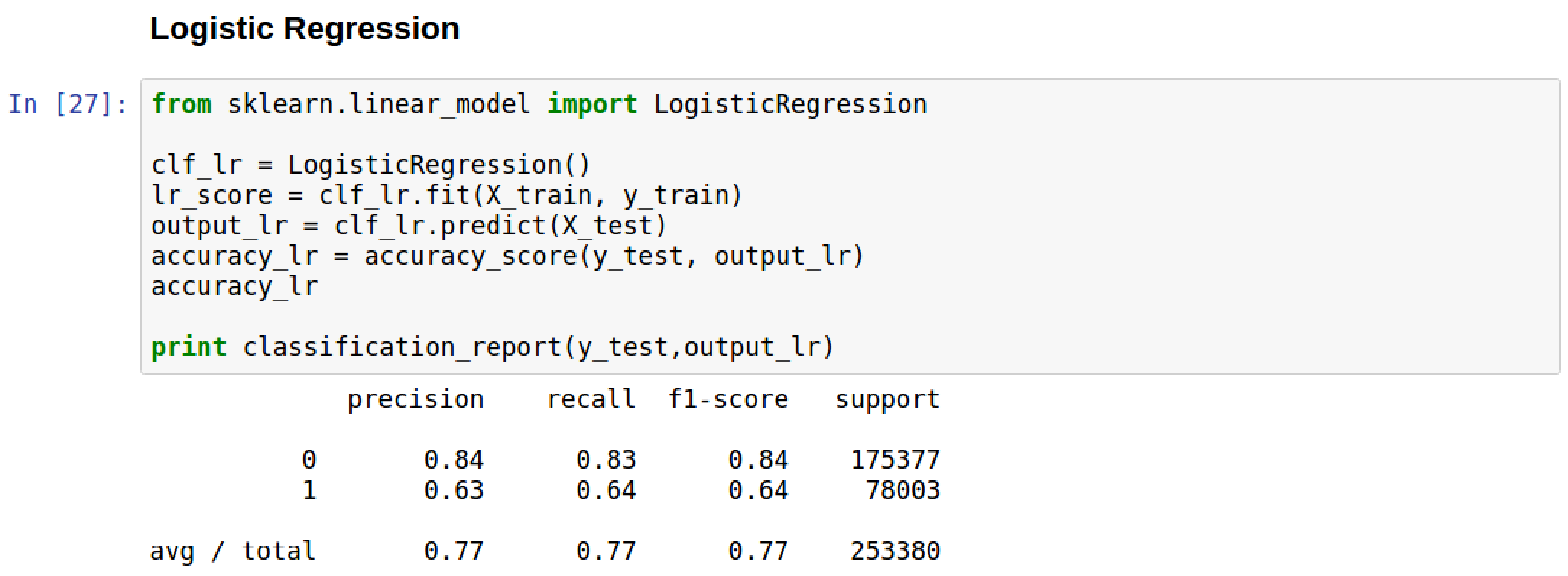
Naive Bayes

```
In [51]: from sklearn.naive_bayes import GaussianNB
         from sklearn.metrics import classification_report

         clf_NB = GaussianNB()
         nb_dec = clf_NB.fit(X_train, y_train)
         output_NB = clf_NB.predict(X_test)
         from sklearn.metrics import accuracy_score
         accuracy_NB = accuracy_score(y_test, output_NB)
         accuracy_NB

         print classification_report(y_test,output_NB)
```

```
             precision    recall  f1-score   support

          0       0.85      0.79      0.82    175377
          1       0.59      0.68      0.63     78003

avg / total       0.77      0.76      0.76    253380
```

Machine Learning Prediction

- Logistic Regression
  - Precision for unpopular: 84%
  - Precision for popular: 63%
  - Precision Accuracy: 77%



Logistic Regression

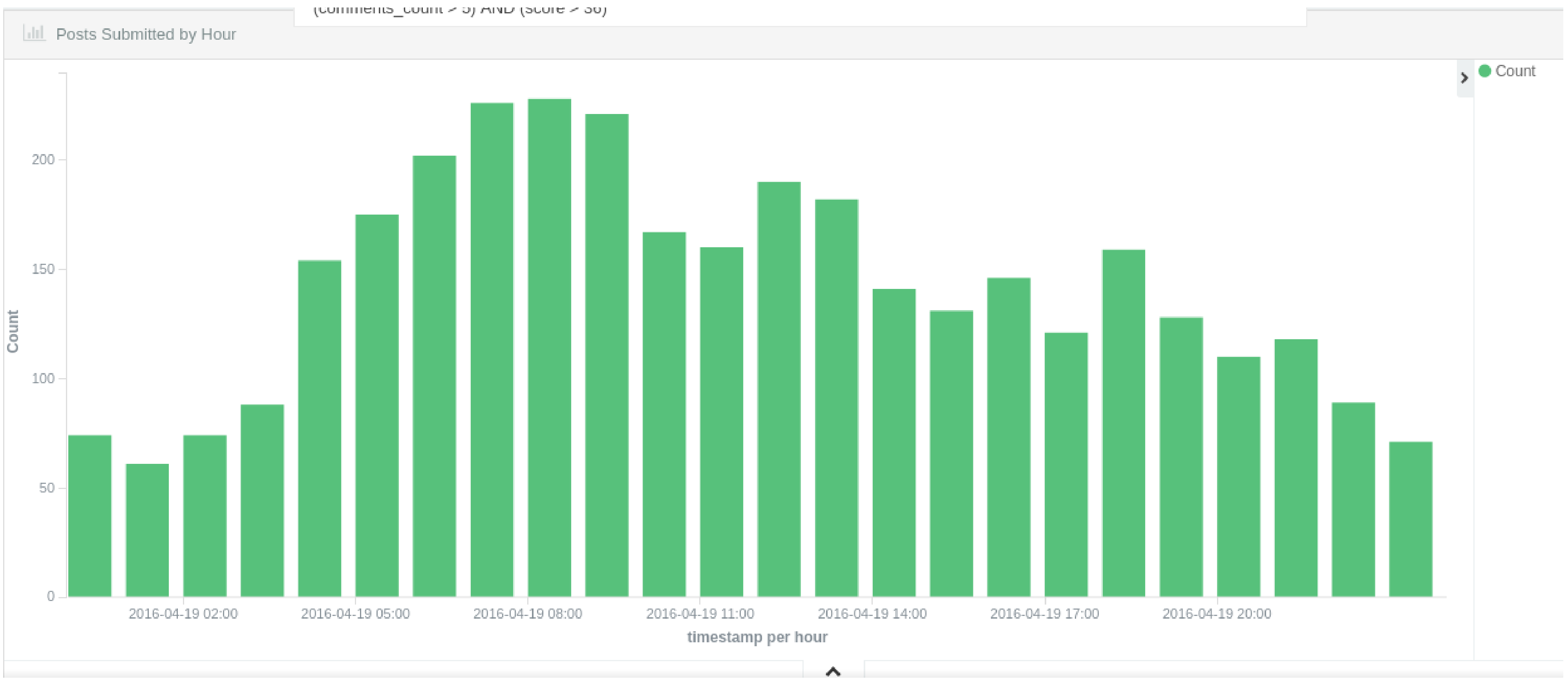
In [27]:
```
from sklearn.linear_model import LogisticRegression

clf_lr = LogisticRegression()
lr_score = clf_lr.fit(X_train, y_train)
output_lr = clf_lr.predict(X_test)
accuracy_lr = accuracy_score(y_test, output_lr)
accuracy_lr

print classification_report(y_test,output_lr)
```

```
             precision    recall  f1-score   support

          0       0.84      0.83      0.84    175377
          1       0.63      0.64      0.64     78003

avg / total       0.77      0.77      0.77    253380
```

Conclusion

3.5 million, 12 GB

Post in the Morning



Sports Related

(No Games)

External Source

(imgur, youtube)

Controversial, not-so-boring

Comments

By:

Tony Guardado
Seung Kim