SC1015
A133 Team 9

Boslyn Pang U2221298A
Tan Yue Hui U2221209K
Shen Jia Cheng U2220979E

Contents

01

Motivation & Problem Definition

Our Motivation

Increasing Trend

Of fake news being ever present in the virtual world

Range of Banes

Including altering our perception and beliefs

Undesirable

In our increasingly tech-dependant society

Our Problem Statement

How might we determine if news on online platforms is real or fake?

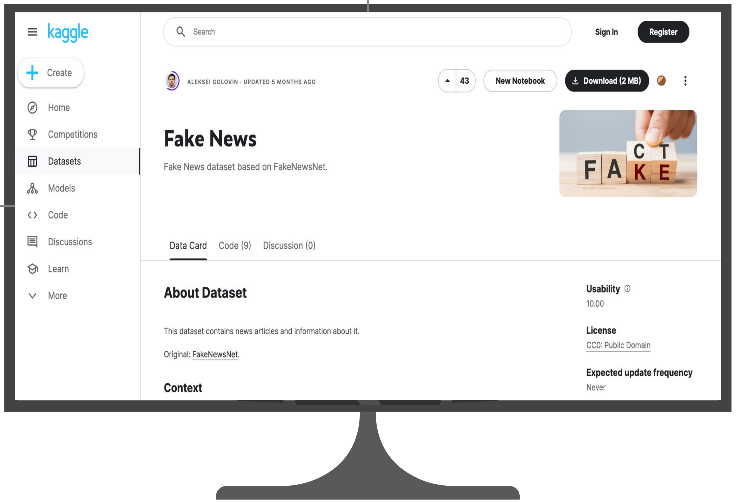
Automated Detection and Classification by using Machine Learning Techniques

Different Prediction Models to determine the validity of the news

Dataset Used

Raw data is assessed from:
Kaggle.com

5 Variables



title *of article*

news_url:
URL of article

source_domain:
web domain of article

tweet_num:
number of retweets

Real:
Real = 1
Fake = 0

02

Data Preparation And Exploration

Data Preparation

1

Calculating total number of words in title

Storing the result in new column **“title count”**

```
#Creating new column for number of words in title
def word_count(title):
    count = 1
    for x in title:
        if x == " ":
            count += 1
    return count

df["title count"] = word_count(df['title'])
for y in range(0,23196):
    df["title count"][y] = word_count(df['title'][y])
```

	title	news_url	source_domain	tweet_num	real	title count
0	Kandi Burruss Explodes Over Rape Accusation on...	http://toofab.com/2017/05/08/real-housewives-a...	toofab.com	42	True	13
1	People's Choice Awards 2018: The best red carp...	https://www.today.com/style/see-people-s-choic...	www.today.com	0	True	9
2	Sophia Bush Sends Sweet Birthday Message to 'O...	https://www.etonline.com/news/220806_sophia_bu...	www.etonline.com	63	True	15
3	Colombian singer Maluma sparks rumours of inap...	https://www.dailymail.co.uk/news/article-33655...	www.dailymail.co.uk	20	True	10
4	Gossip Girl 10 Years Later: How Upper East Sid...	https://www.zerchoo.com/entertainment/gossip-g...	www.zerchoo.com	38	True	17

Data Preparation

2

Removing Stop Words and Tokenization

Removing of stop words and splitting of **title** into individual words

```
#Cleaning of title by removing stop words and lemmatizing of words
def clean_data(text):
    text = text.lower()
    text = re.sub('[^a-zA-Z]' , ' ' , text)
    token = text.split()
    token = [lemmatizer.lemmatize(word) for word in token if not word in stop_words]
    clean_news = ' '.join(token)

    return clean_news
df['title'] = df['title'].apply(lambda x : clean_data(x))
```

Data Preparation

3

Removing Null Values

Removing missing values

```
[7] df.isnull().sum()

title              0
news_url         330
source_domain    330
tweet_num          0
real               0
dtype: int64
```

Data Preparation

3

Removing Null Values

Removing missing values

```
df.dropna(inplace=True)
```

```
[11] df.isnull().sum()
```

```
title            0
news_url         0
source_domain    0
tweet_num        0
real             0
dtype: int64
```

Data Preparation

3

Converting variable type

Variable: *real*
Converting var type of int64 (binary) to boolean

```
# Converting int64 variable (real) to boolean (True/False)
df['real'] = df['real'].astype('bool')
print(df.dtypes)
```

```
title            object
news_url         object
source_domain    object
tweet_num         int64
real               bool
dtype: object
```

03

Exploratory Data Analysis (Univariate and Bivariate)

Categorical Plot
(Univariate Plot)

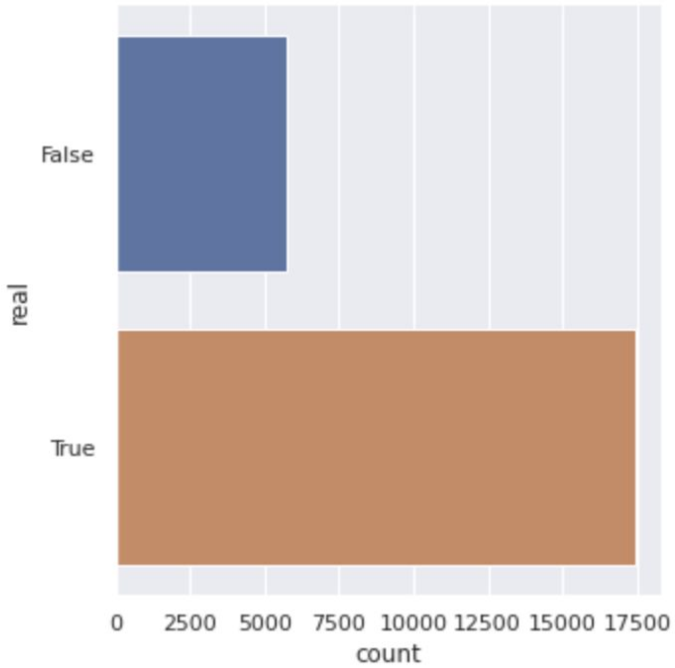
True vs False news articles

Data type: Categorical

Analysis:
Number of real articles are significantly higher than false articles
- Data set is skewed towards TRUE ARTICLES

```
# plotting the cat plot of True vs False (univariate)
print(df["real"].value_counts())
sb.catplot(data=df, y='real',kind='count')
```

```
True     17441
False     5755
Name: real, dtype: int64
<seaborn.axisgrid.FacetGrid at 0x7f5fe6dd7e50>
```



Source Domain

Data Type: Categorical
Number of different sources: 2441

~73% of articles of the dataset came from the same source: People.com

```python
print('Number of sources: ', len(df['source_domain'].unique()))
print(df['source_domain'].value_counts())
```

```
Number of sources:  2441
people.com                 1786
www.dailymail.co.uk         964
en.wikipedia.org            741
www.usmagazine.com          709
www.etonline.com            666
                           ...
bioguide.congress.gov         1
dailyheadlines.net            1
www.duggarfamily.com          1
www.naturallycurly.com        1
flashnewscorner.com           1
Name: source_domain, Length: 2441, dtype: int64
```

Number of Retweets

Variable name: tweet_num

Data Type: Numerical
Average number of retweets: ~89 / article
Median: 37 / article

Data is very skewed as:

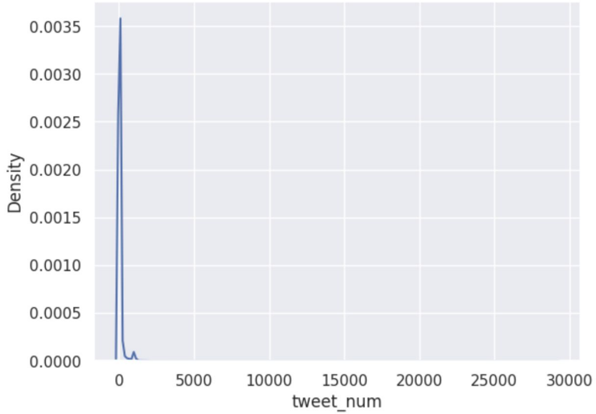
- Maximum is a lot higher than the 75th percentile
- Std = ~489
- Mean > median

```
# describe data for number of retweets
df['tweet_num'].describe()
```

```
count    22866.000000
mean        88.398802
std        488.712092
min          0.000000
25%         11.000000
50%         38.000000
75%         65.000000
max      29060.000000
Name: tweet_num, dtype: float64
```

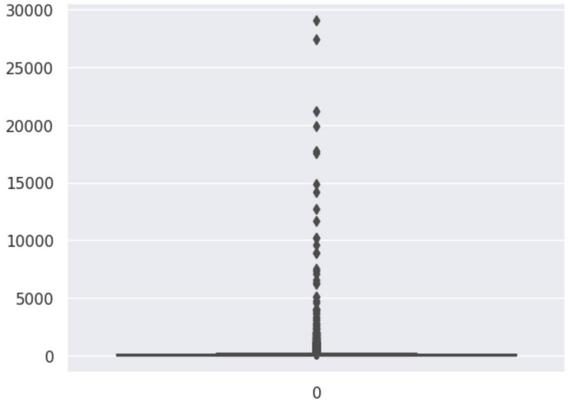
```
#Plot distribution for number of retweets
sb.kdeplot(df['tweet_num'])
```

<Axes: xlabel='tweet_num', ylabel='Density'>




```
# Plot boxplot for number of retweets
sb.boxplot(df['tweet_num'])
```

<Axes: >



Title count

title count: Number of words in title
Data Type: Numerical

```
# Statistical summary of number of words in title
df['title count'].describe()
```

```
count    22866.000000
mean        11.044564
std          4.125675
min          1.000000
25%          9.000000
50%         11.000000
75%         14.000000
max         53.000000
Name: title count, dtype: float64
```

Title count

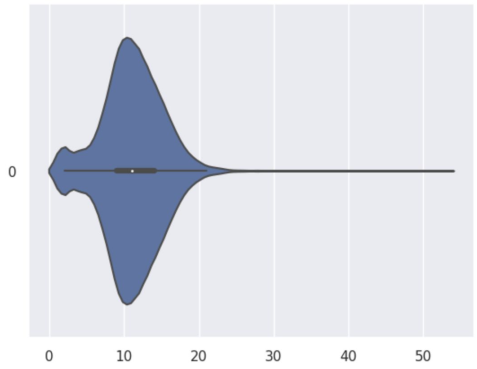
title count: Number of words in title
Data Type: Numerical

Observations:

- Not a lot of outliers, therefore do not have to clean data based on this variable
- Distribution is similar to a normal distribution as mean=median=11, and mode is ~10.

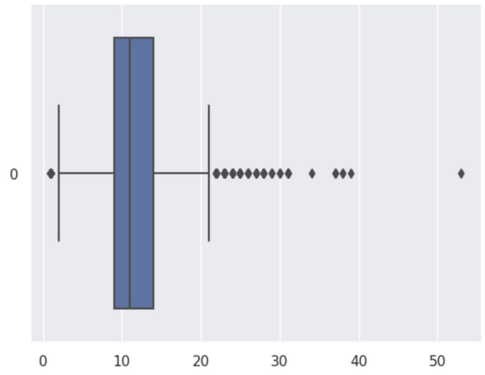
```
sb.violinplot(df['title count'], orient='h')
```

<Axes: >



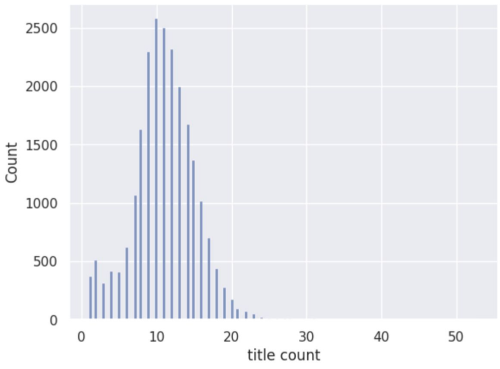

```
sb.boxplot(df['title count'],orient='h')
```

<Axes: >



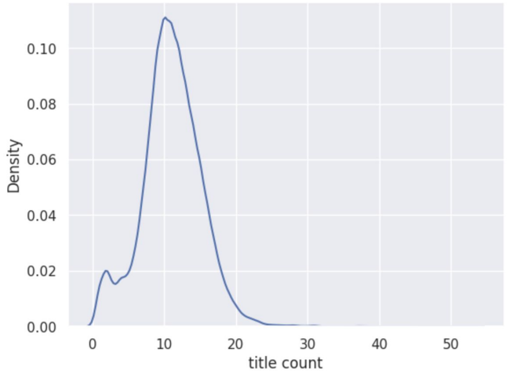

```
sb.histplot(df['title count'])
```

<Axes: xlabel='title count', ylabel='Count'>




```
sb.kdeplot(df['title count'])
```

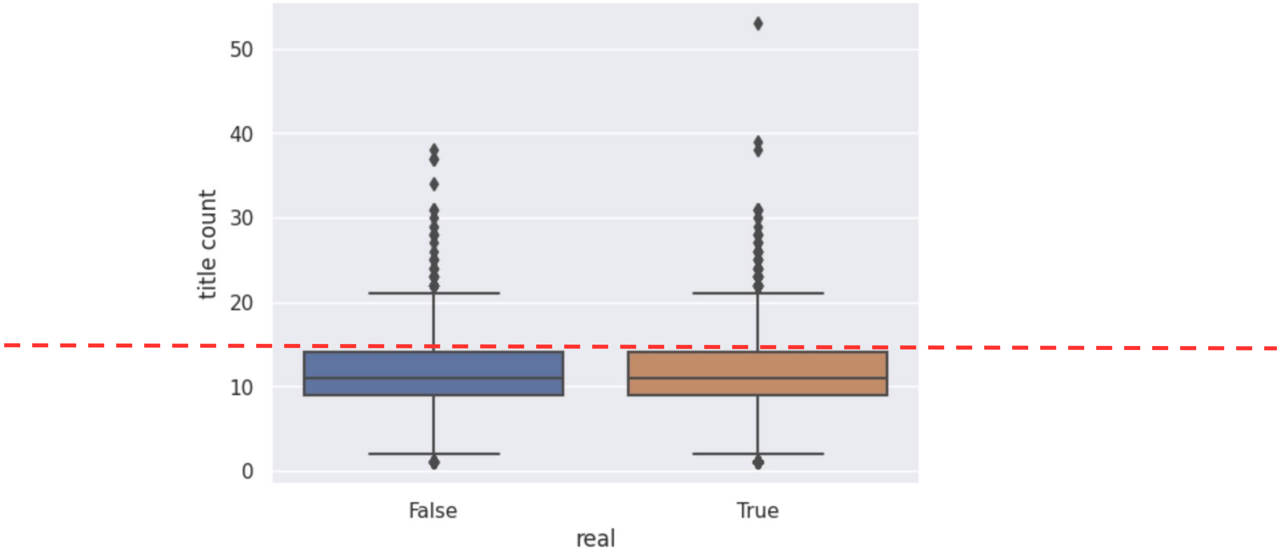
<Axes: xlabel='title count', ylabel='Density'>



Bivariate Statistics

```
sb.boxplot(x='real', y='title count', data=df)
```

<Axes: xlabel='real', ylabel='title count'>



Little difference between the 2 boxplots

Insignificant relationship between *real* and *title count*

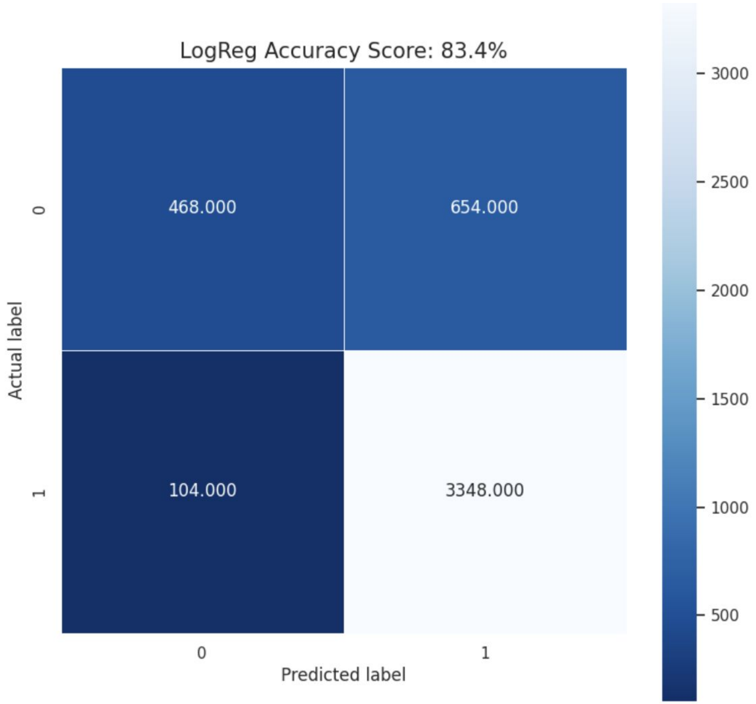
04

Machine Learning

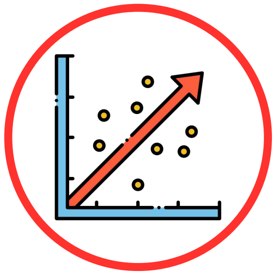
T=True	F=False
P=Positive	N=Negative

1. Logistic Regression Model

Used when the **response variable** is **categorical**



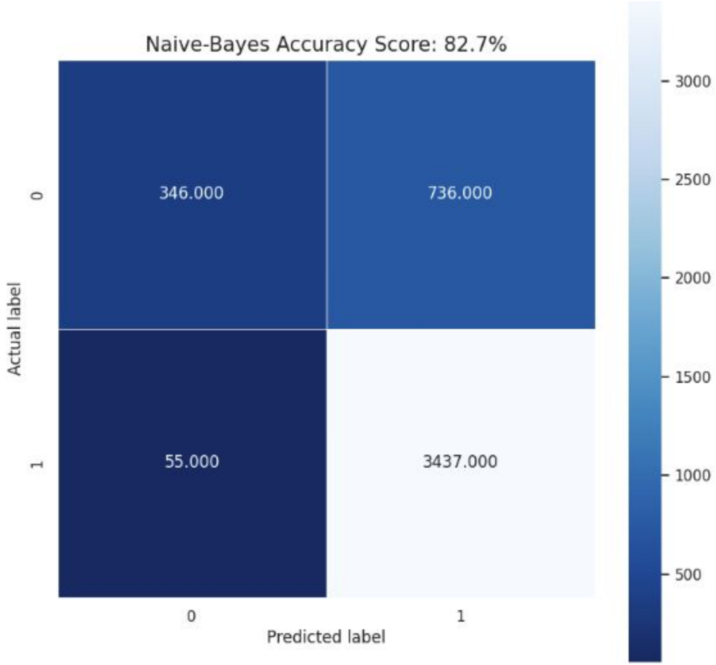
- **LogReg** Accuracy Score: **83.4%**
- TPs > FPs
- TNs>FNs
- FN rate = 0.0301 (3 s.f.)
- FP rate = 0.583 (3 s.f.)
- **HIGH** FPR, **LOW** FNR



T=True	F=False
P=Positive	N=Negative

2. Naive-Bayes Prediction Model

Handles both **continuous** and **discrete** data



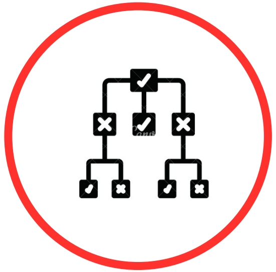
- **Naive-Bayes** Accuracy Score: **82.7%**
- TPs > FPs
- TNs>FNs
- FN rate = 0.0158 (3 s.f.)
- FP rate = 0.680 (3 s.f.)
- HIGH FPR, LOW FNR

T=True	F=False
P=Positive	N=Negative

3. Decision Tree Classifier Model

- ❖ Lay out the problem and **all** possible outcomes
- ❖ **Gini Index**

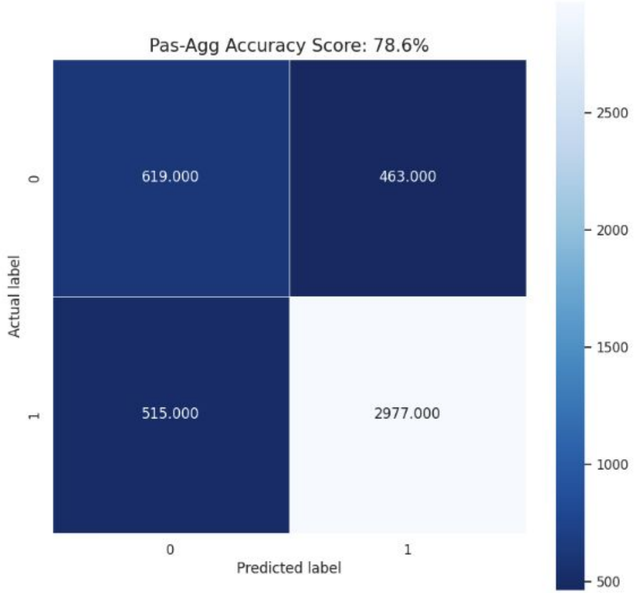
- **Dec Tree** Accuracy Score: **80.0%**
- TPs > FPs
- TNs>FNs
- FN rate = 0.122 (3 s.f.)
- FP rate = 0.453 (3 s.f.)



T=True	F=False
P=Positive	N=Negative

4. Passive-Aggressive Classifier Model

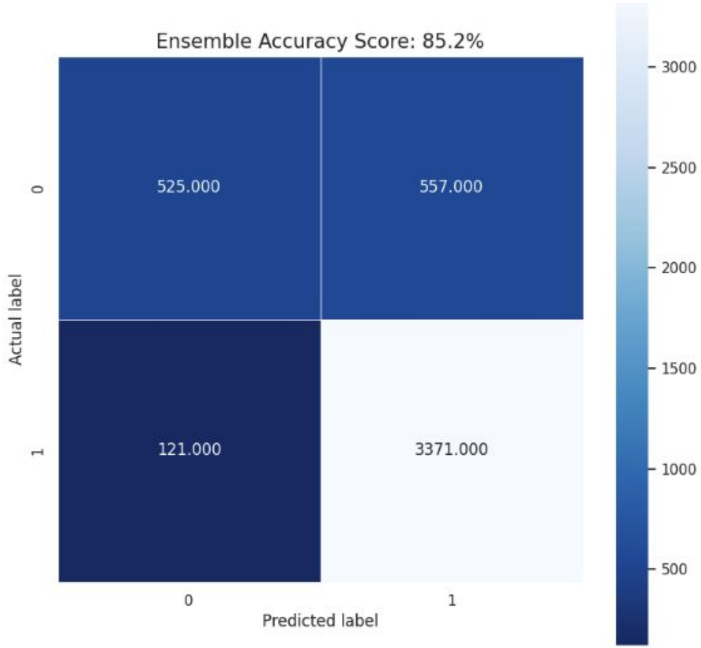
❖ **Updates** its model based on each new instance it encounters



- LogReg Accuracy Score: **78.6%**
- TPs > FPs
- TNs>FNs
- FN rate = 0.147 (3 s.f.)
- FP rate = 0.428 (3 s.f.)

T=True	F=False
P=Positive	N=Negative

5. Ensemble



- LogReg Accuracy Score: **85.2%**
- TPs > FPs
- TNs>FNs
- FN rate = 0.0346 (3 s.f.)
- FP rate = 0.515 (3 s.f.)
- HIGH FPR, LOW FNR

SUMMARY:

	Model	Accuracy	True Pos	False Pos	True Neg	False Neg
0	Log Reg	84.6	85.1	14.9	81.1	18.9
1	Naive-Bayes	82.7	82.4	17.6	86.3	13.7
2	Decision Tree	80.0	86.2	13.8	58.2	41.8
3	Pas-Agg	78.6	86.5	13.5	54.6	45.4
4	SVM	84.8	86.6	13.4	75.9	24.1
5	Ensemble	85.2	85.8	14.2	81.3	18.7

Highest Accuracy

Lowest Accuracy

05

Insights & Evaluation

Model Outcomes

Accuracy of Model

Out of the 5 different trained and tested models, our ensemble model performed the best

Accuracy of Model: 85.2%

Mostly able to detect strong emotional words in the titles of the fake news

Solved our original problem of detecting and warning users of fake news

Model Evaluation

Errors

Despite the high accuracy of our model, it would be best coupled with human vetting

A

Political Data

Our dataset is heavily skewed towards political content, may not be accurate in predicting other types of fake news

B

Future Recommendations

Other Datasets

1

Can look at datasets of non-political news as well

Other Variables

2

Eg. No. of words in article, pictures, etc.

Other Models

3

Other ensemble models such as stacking

Thank You!