# Python project

## Subject: Online News Popularity

Loévan LE QUERNEC, Santiago MARTIN, Bertrand NAGY

# CONTENT

# The dataset

Published by: Kelwin Fernandes Pedro Vinagre, Paulo Cortez, Pedro Sernadela on 05/30/15

Mashable

- Gathers **61 features** about **37589 articles** published by **Mashable** in a period of two years.

- **Goal** : predict the number of **shares** in social networks

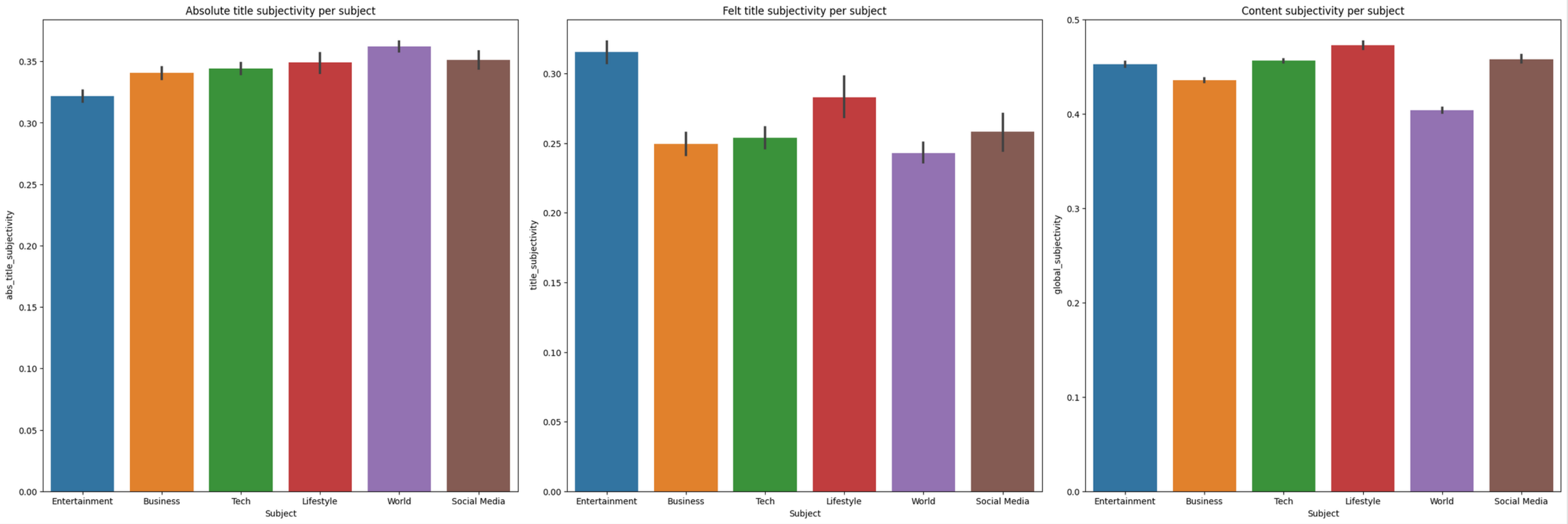
# Data cleaning

- Dataset was already fairly clean : no missing values, no empty nor duplicated rows
- Deleted the space before each column names (‘ n_tokens’ etc)
- Dropped the url and timedelta column (useless columns)
- Dropped ‘is_weekend’ as specific days columns already existed

# Data Visualization

- count plot : shares per day of the week
- countplot : number of articles  per each subject
- barplot : shares per each subject
- barplot : title subjectivity per subject
- barplot : absolute and felt title subj. per subject
- barplot : content subj. and polarity per subject
- histogram: average keywords popularity and shares
- countplot: title length
- heatmap : correlation betw. shares, absolute and felt title subj.
- heatmap : correlation betw. shares, amount of images and videos in articles
- heatmap : correlation betw. shares, global rate of positive and negative words

# Subjectivity plots:




Delta between **felt** and **abs**. title subj

‘**World**’ titles are seen as quite objective despite being subjective

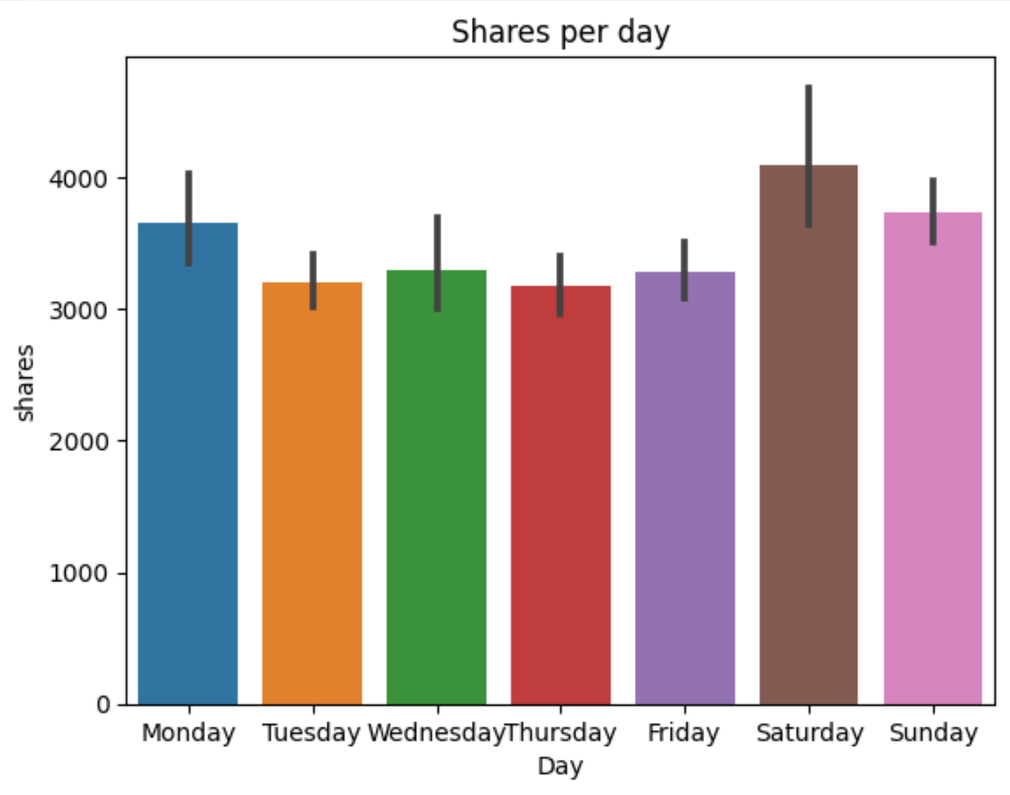
‘**Entertainment**’ titles are the **only stable values**

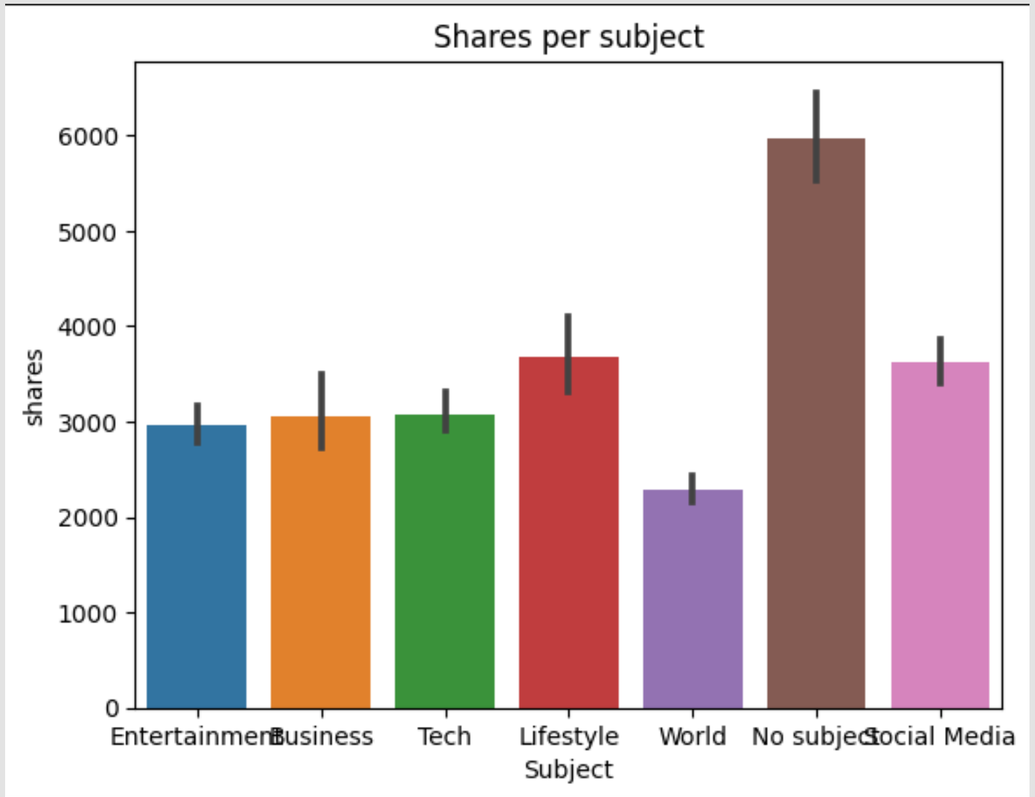
‘World’ has high title subj but **low content subj**

‘Lifestyle’ is the most **subjective all aroud**

‘Tech’ is surprisingly subjective

# Shares plots:

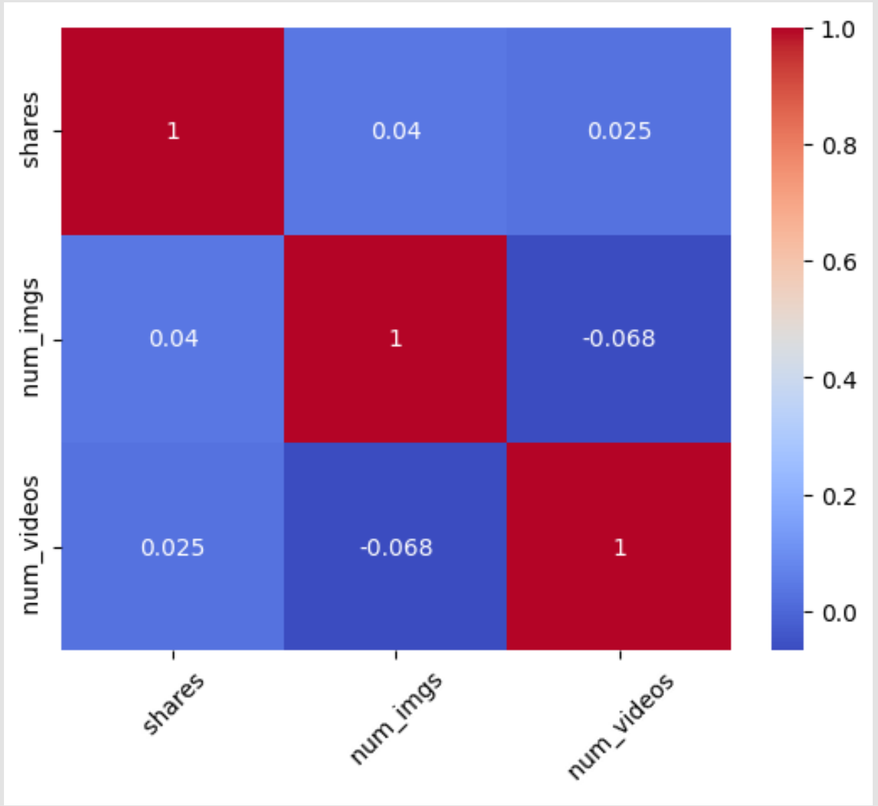



- More shares the **weekend**
- The highest value is ‘Saturday’

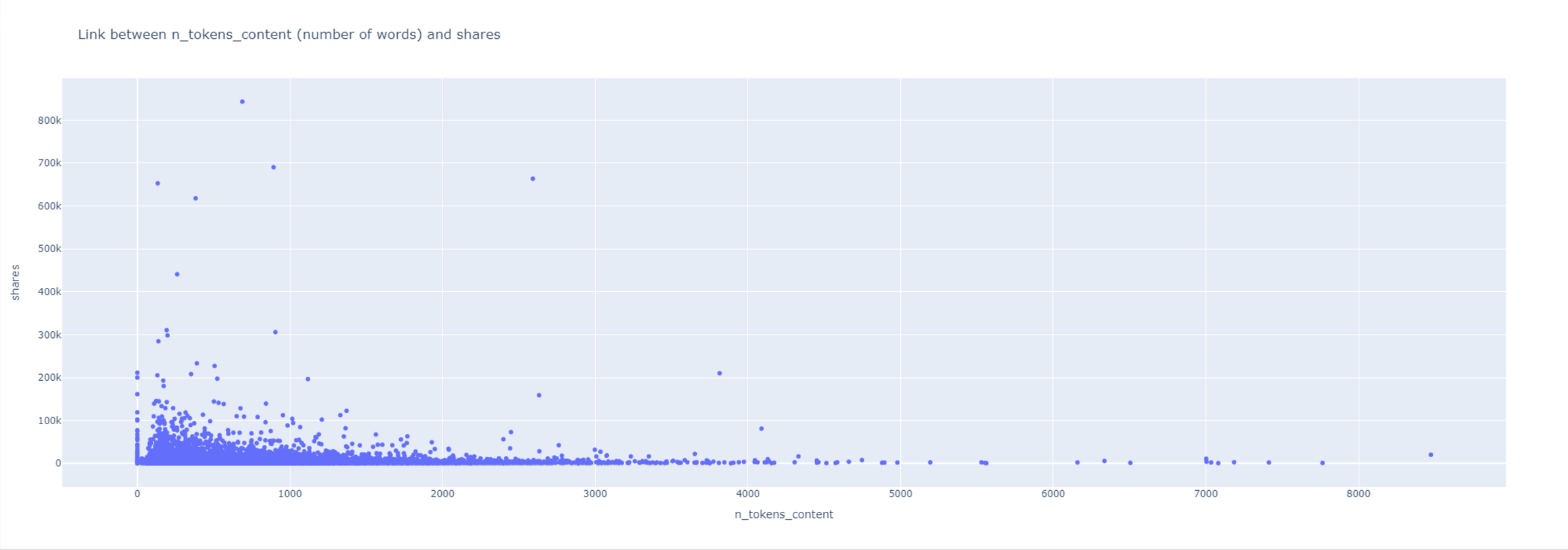
- A lot of articles with **‘No subject‘**
- Predominant subject are ‘Lifestyle’ and ‘Social Media’

# Correlation of images, videos and shares:




**Low** correlation : proves that **written** articles still have their place in the era of social media

# Influence of the words in an article:




An article with **many words** is not necessarily shared more than an article with **few words**

# Machine Learning

Here are the steps to prepare the machine learning part:

- Split the data into two variables:
  - the featured variables
  - the target variable (with a decision threshold of 1400, equal either to 0 or 1)
- Split into two sets:
  - training set
  - test set
- Revome some correlated columns with a decision boundarie of 0.85 (example: n_non_stop_words/unique_tokens')
- Scale the data
- Evaluate each model

The preview of the heatmap:

# Machine Learning

## LinearRegression and Lasso:

LinearRegression:

```
Accuracy:  0.6258692628650904
Mean Squared Error: 0.3741307371349096
R-squared: -0.50203238491699899
```

Good accuracy

MSE really high

$R^2$ negative

Lasso:

```
Accuracy:  0.5302819572638766
Mean Squared Error: 0.4697180427361234
R-squared: -0.8857892226990933
```

Bad accuracy (compared to LinearRegression)

MSE really high

$R^2$ negative

# Machine Learning

SVC:

SVC:

```
Accuracy:  0.639903906941459
Precision:  0.6449181739879414
Confusion matrix:
 [[2066 1649]
 [1199 2995]]
```

Good accuracy (0.63)

SVC with GridSearch:

```
Accuracy:  0.6392717157668479
Precision:  0.6479152878888154
Confusion matrix:
 [[2119 1596]
 [1257 2937]]
```

Good accuracy (0.63)

GridSearch not improved our model

# Machine Learning

## DecisionTreeClassifier and RandomForestClassifier:

DecisionTreeClassifier:

```
Accuracy:  0.5737767100771273
Precision:  0.5976738666033705
Confusion matrix:
 [[2020 1695]
 [1676 2518]]
```

Bad accuracy (0.57)

RandomForestClassifier:

```
Accuracy:  0.6371222657731698
Precision:  0.6433521004763967
Confusion matrix:
 [[2068 1647]
 [1223 2971]]
```

Good accuracy (0.63)

RandomForestClassifier with GridSearch:

```
Accuracy:  0.6473637628018712
Precision:  0.6501389185723445
Confusion matrix:
 [[2078 1637]
 [1152 3042]]
```

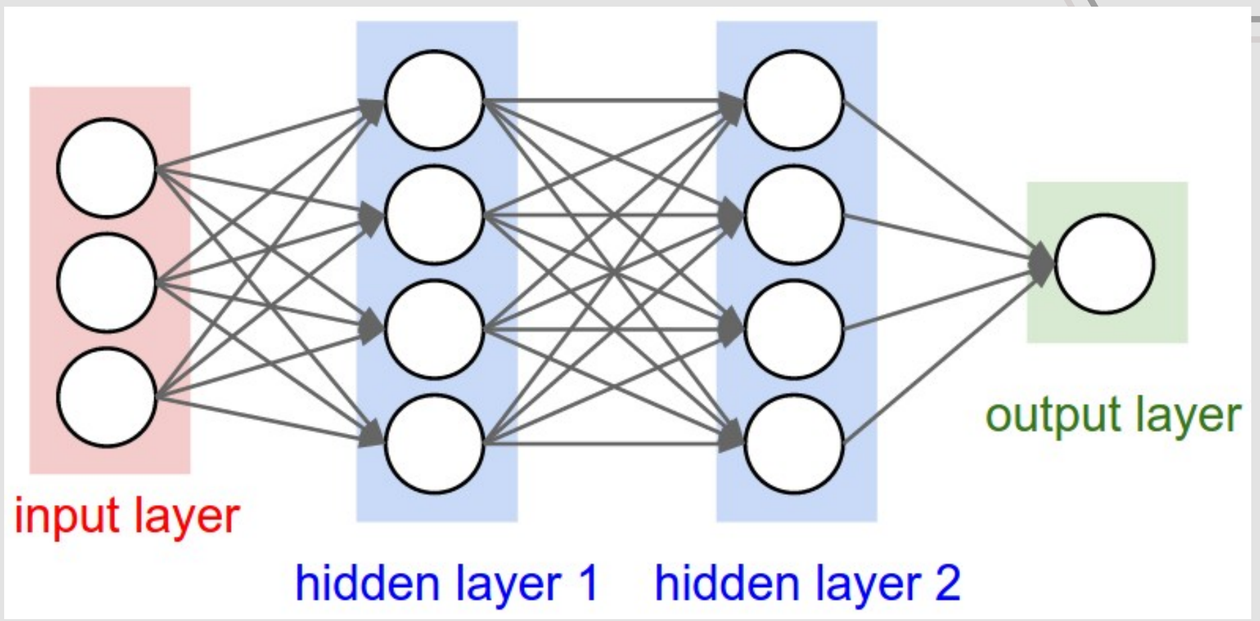
Best accuracy (0.64)

GridSearch improved the model

# Deep Learning

## Binary classification with Neural Networks

- Predict the number of shares of a news piece based on a decision boundary of 1400
  - y = 1 if shares > 1400 else 0
- Model inspired by human brain's neural system for learning patterns and relationships in data
- Complex architecture: input layer, hidden layers (ReLu activation), output layer (sigmoid activation)



# Hyperparameter tuning for model optimization

## Random Search Cross-Validation

- Vast array of parameters to test: units, dropout rate, learning rate, epochs, batch size
  - Random Search >> Grid Search
- 10 iterations with 3 folds: 30 total fits
- Best iteration provides 66% mean accuracy
- Used RandomizedSearchCV, GridSearchCV, and KerasClassifier from sklearn

**Values to test:**

- **units: [64, 32], [128, 64, 32]**
- **dropout rate: 0.3, 0.5, 0.7**
- **learning rate: 0.1, 0.01, 0.001**
- **epochs: 10, 20, 30**
- **batch size: 16, 32, 64**

# Final Model

**Format**

- Neuron count and dropout rate taken from best parameters of Random Search
- Dense layers (ReLu activation) followed by Dropout layers
- Final Dense layer with single neuron and sigmoid activation

**Compilation and fitting parameters**

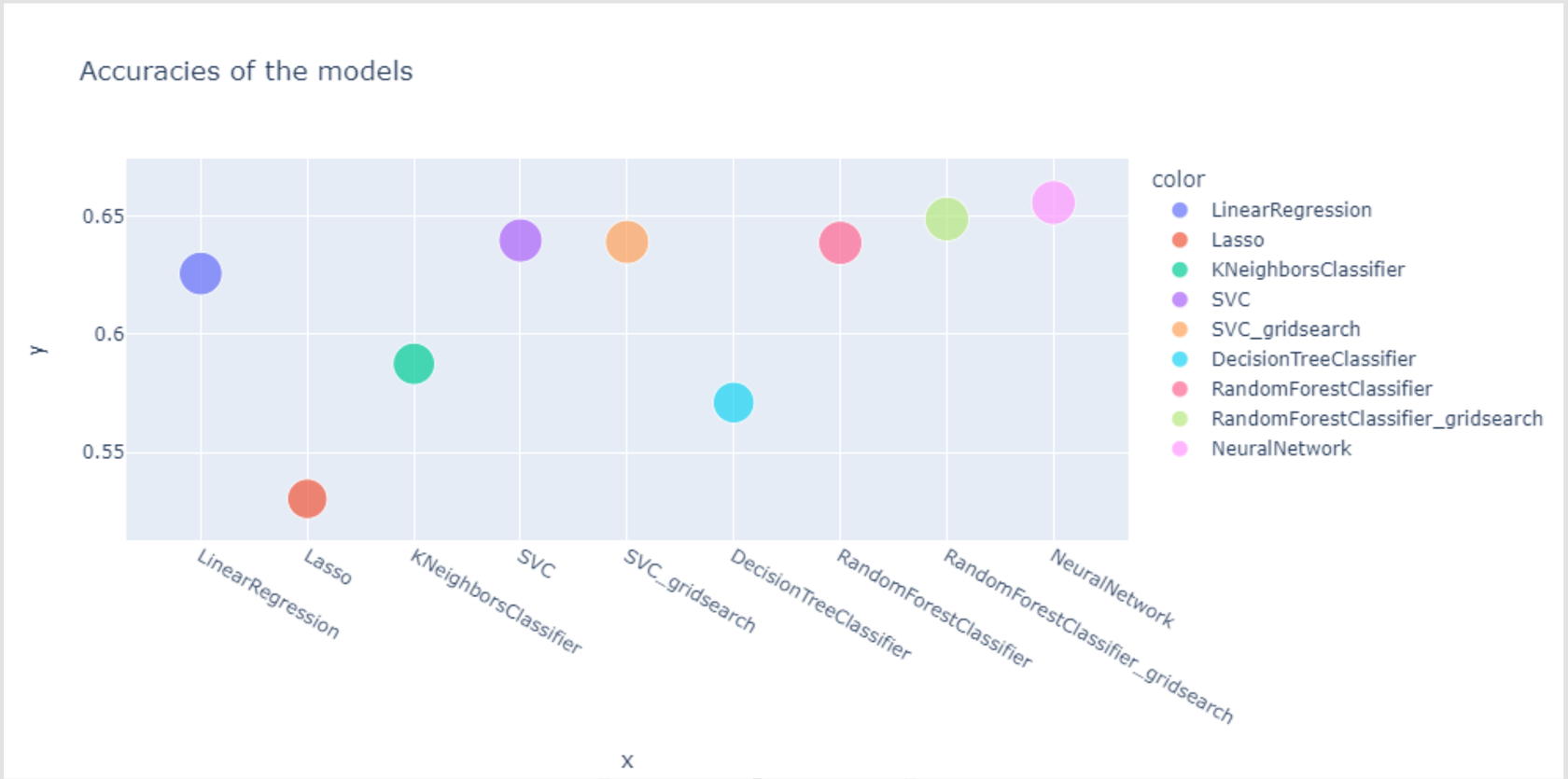
- optimizer: Adam with learnig rate of 0.001
- loss function: Binary Crossentropy
- metrics: Accuracy
- Epochs and batch size taken from best parameters of Random Search

**Performance**

```
Epoch 17/20
481/481 [==============================] - 1s 1ms/step - loss: 0.6074 - accuracy: 0.6677 - val_loss: 0.6177 - val_accuracy: 0.6596
Epoch 18/20
481/481 [==============================] - 1s 1ms/step - loss: 0.6058 - accuracy: 0.6673 - val_loss: 0.6154 - val_accuracy: 0.6598
Epoch 19/20
481/481 [==============================] - 1s 1ms/step - loss: 0.6063 - accuracy: 0.6673 - val_loss: 0.6161 - val_accuracy: 0.6580
Epoch 20/20
481/481 [==============================] - 1s 1ms/step - loss: 0.6039 - accuracy: 0.6675 - val_loss: 0.6153 - val_accuracy: 0.6585
```

```
248/248 [==============================] - 0s 588us/step - loss: 0.6134 - accuracy: 0.6625
Model loss: 0.6133824586868286 and accuracy: 0.66253632307052
```

# What’s the best model?




Our best model: the **NeuralNetwork** one

# THANKS FOR LISTENING TO US