Amazon Review sentiment prediction

SpringBoard Data Science Capstone Project, 2022

Hien Quang

Objective

- Understand modern NLP techniques
- Predict if a review is negative or positive
- Set up for potential future applications such as:
  - Discover discrepancy between review sentiment and rating
  - Extract features that impactly to review's sentiment (what buyers care about most)

Data

Source: https://nijianmo.github.io/amazon/index.html

Features:

- Reviews;
  - Ratings
  - Text
  - Helpfulness
  - Votes

- Product metadata;
  - Description
  - Category
  - Price
  - Brand
  - Image

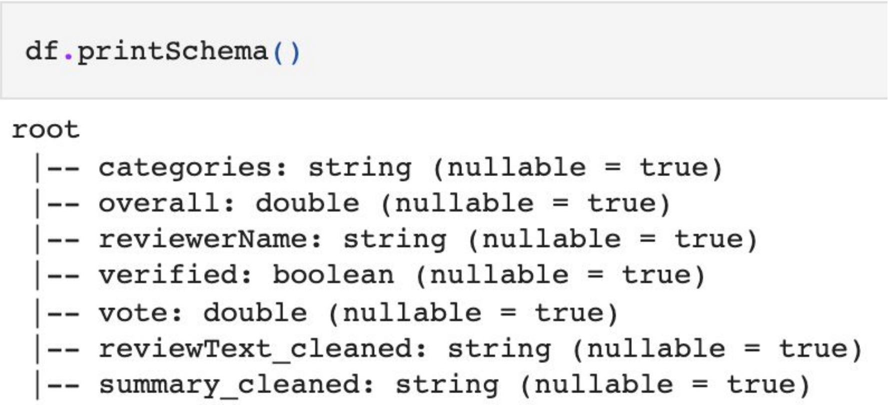
Data Cleaning

Huge dataset - even for the reduced version

- More than 10Gb total
- 30 json.gz files total, one for each categories

Approach:

- Pyspark
- Select needed features

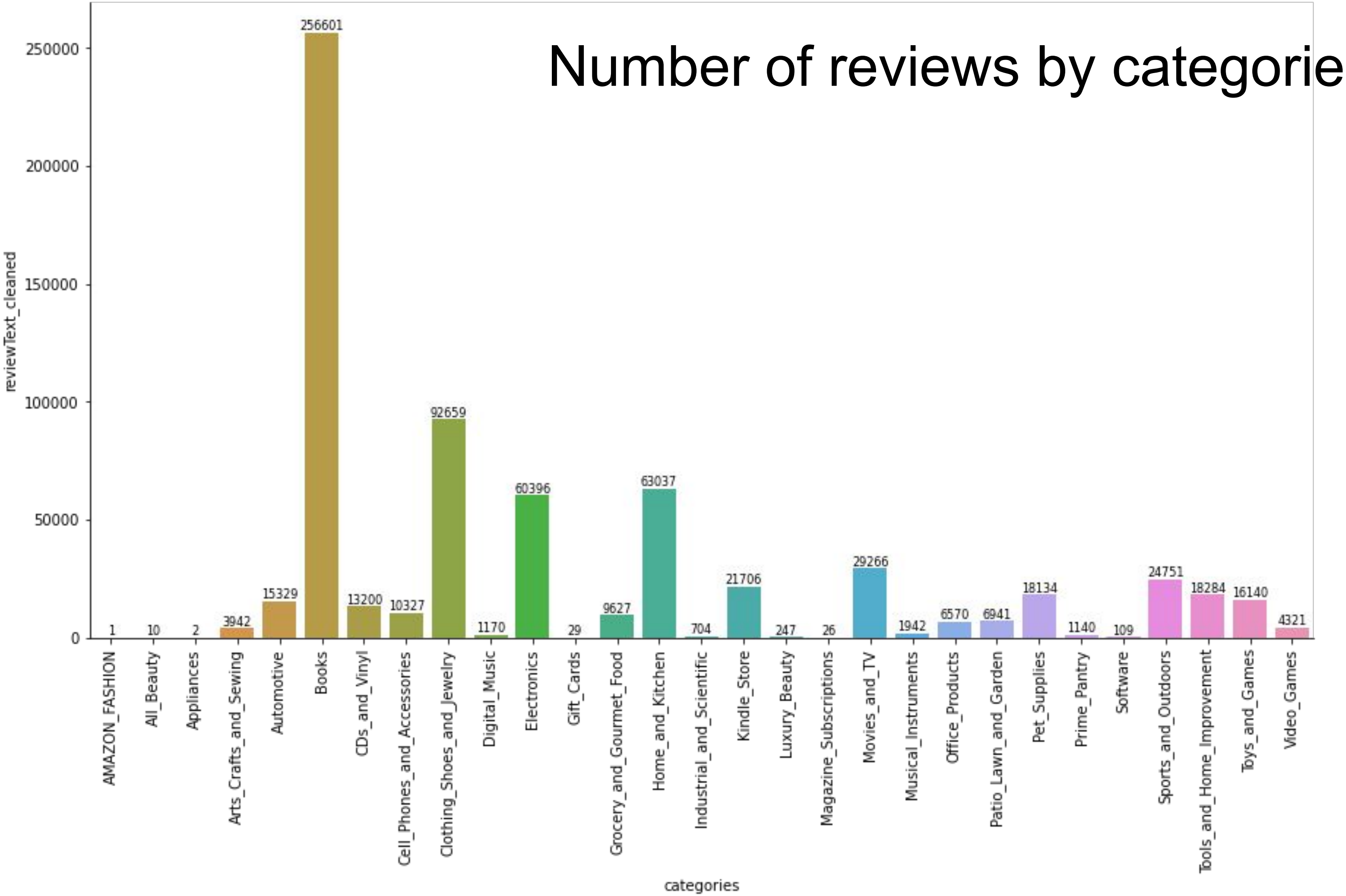


```
df.printSchema()
```

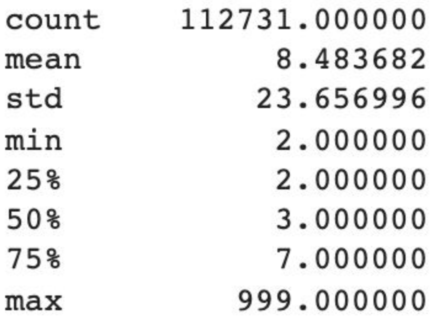
```
root
 |-- categories: string (nullable = true)
 |-- overall: double (nullable = true)
 |-- reviewerName: string (nullable = true)
 |-- verified: boolean (nullable = true)
 |-- vote: double (nullable = true)
 |-- reviewText_cleaned: string (nullable = true)
 |-- summary_cleaned: string (nullable = true)
```

Exploratory Data Analysis (EDA)

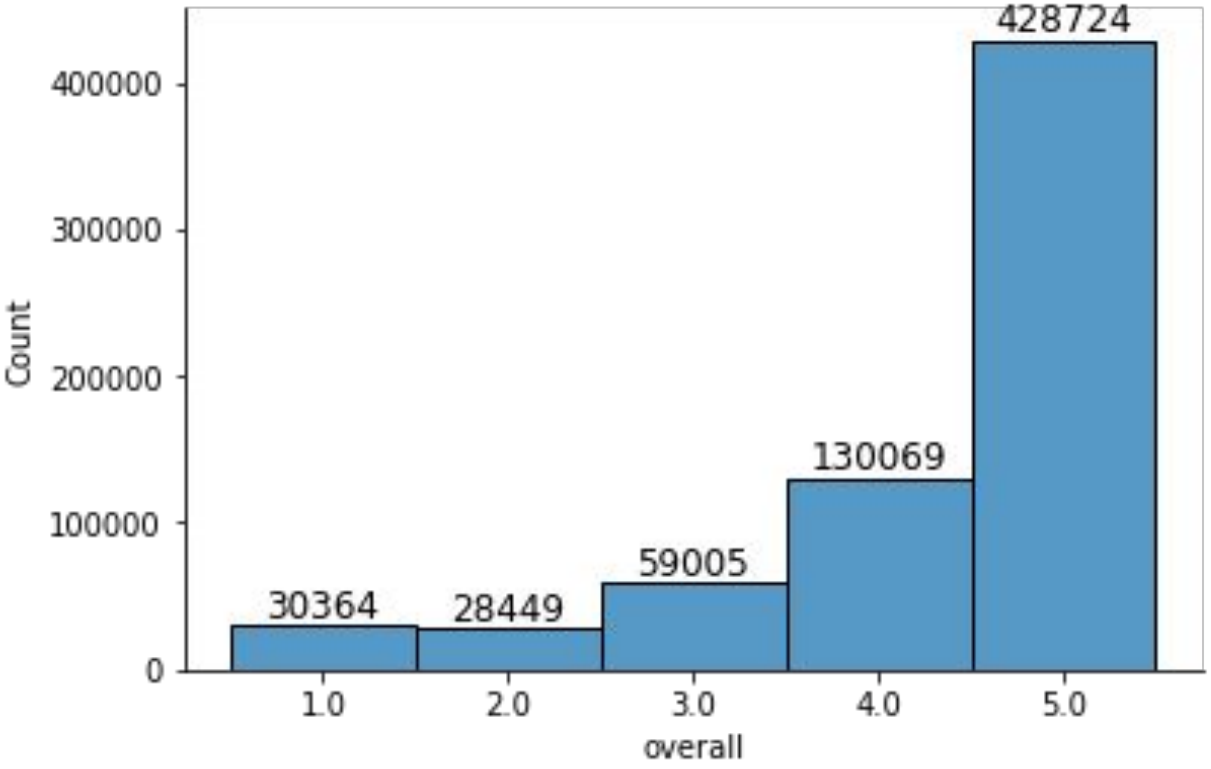
Number of reviews by categories



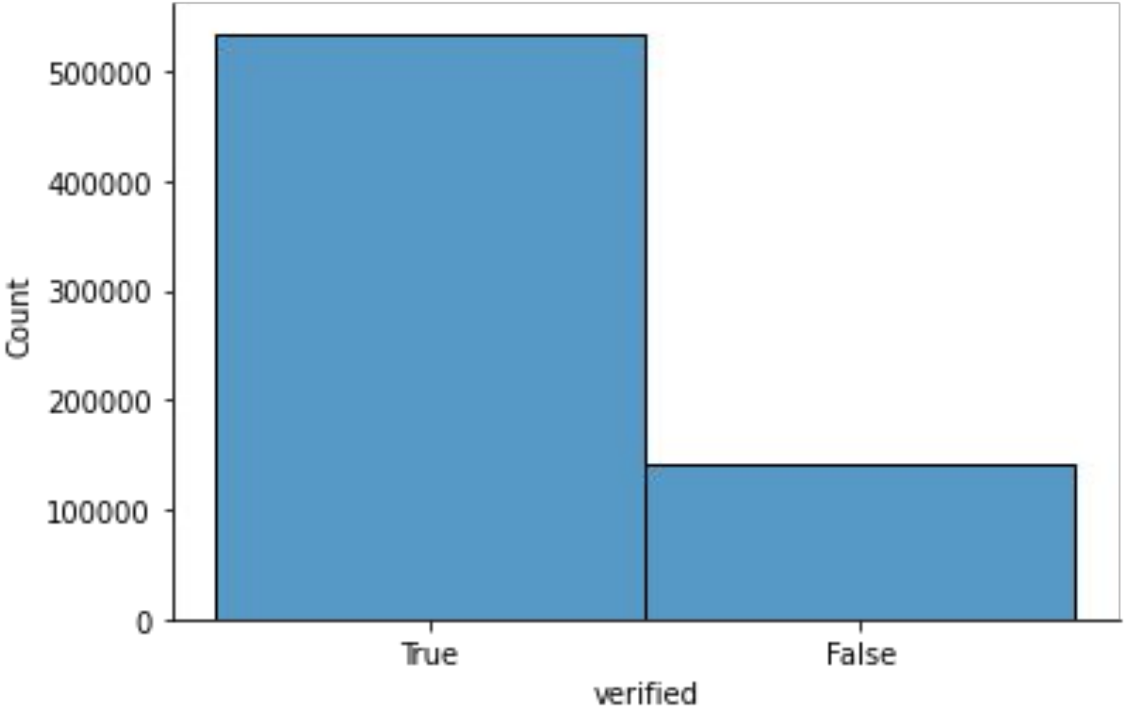
Distribution of overall score



```
count    112731.000000
mean          8.483682
std          23.656996
min           2.000000
25%           2.000000
50%           3.000000
75%           7.000000
max         999.000000
```

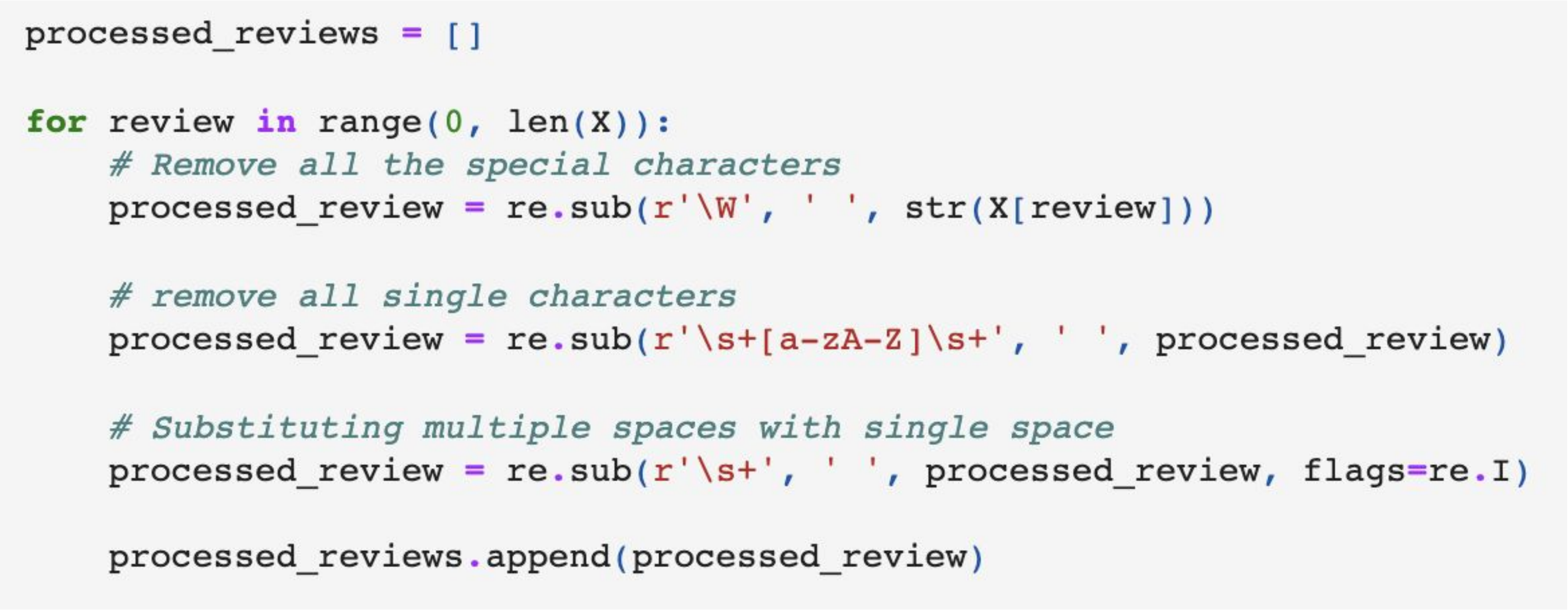


Distribution of verified reviews



Modeling

Processed text

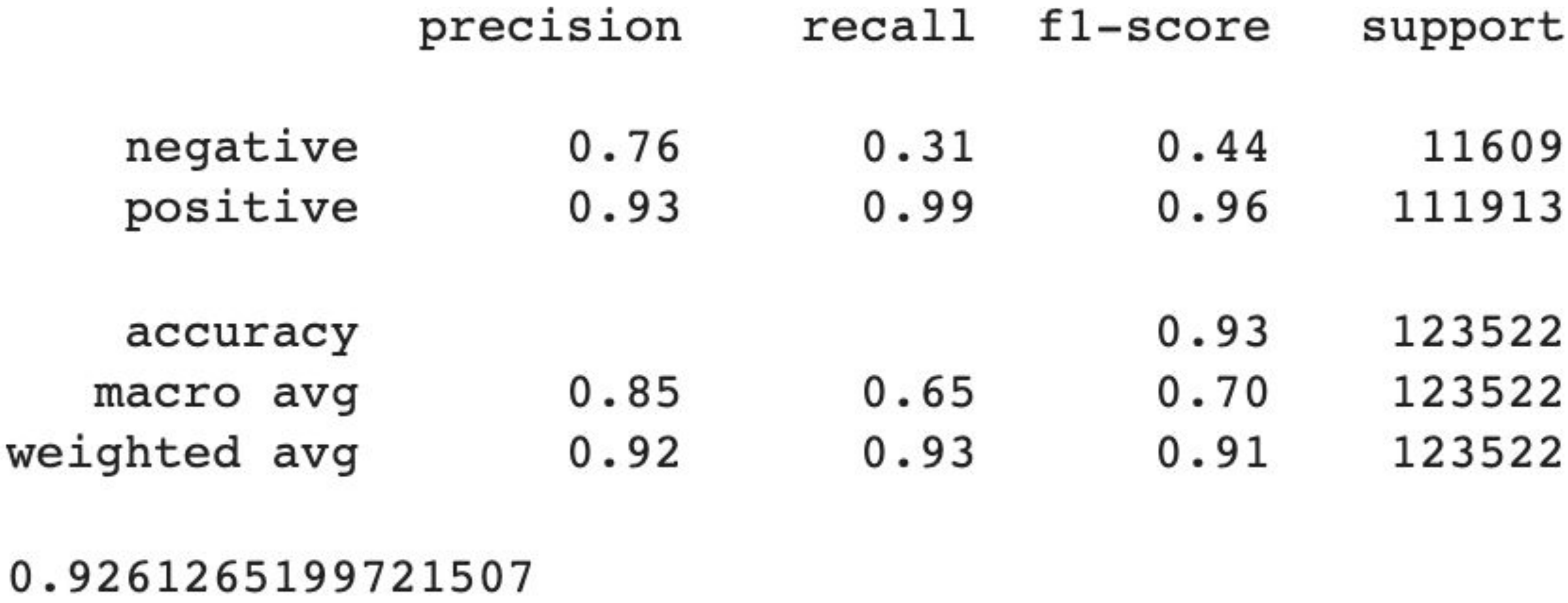


```
processed_reviews = []

for review in range(0, len(X)):
    # Remove all the special characters
    processed_review = re.sub(r'\W', ' ', str(X[review]))

    # remove all single characters
    processed_review = re.sub(r'\s+[a-zA-Z]\s+', ' ', processed_review)

    # Substituting multiple spaces with single space
    processed_review = re.sub(r'\s+', ' ', processed_review, flags=re.I)

    processed_reviews.append(processed_review)
```

TF-IDF

```
tfidfconverter = TfidfVectorizer(
        max_features=500, min_df=5, max_df=0.7,
        stop_words=stopwords.words('english'))

X = tfidfconverter.fit_transform(processed_reviews).toarray()
```

Random Forest - result



```
              precision    recall  f1-score   support

    negative       0.76      0.31      0.44     11609
    positive       0.93      0.99      0.96    111913

    accuracy                           0.93    123522
   macro avg       0.85      0.65      0.70    123522
weighted avg       0.92      0.93      0.91    123522

0.9261265199721507
```