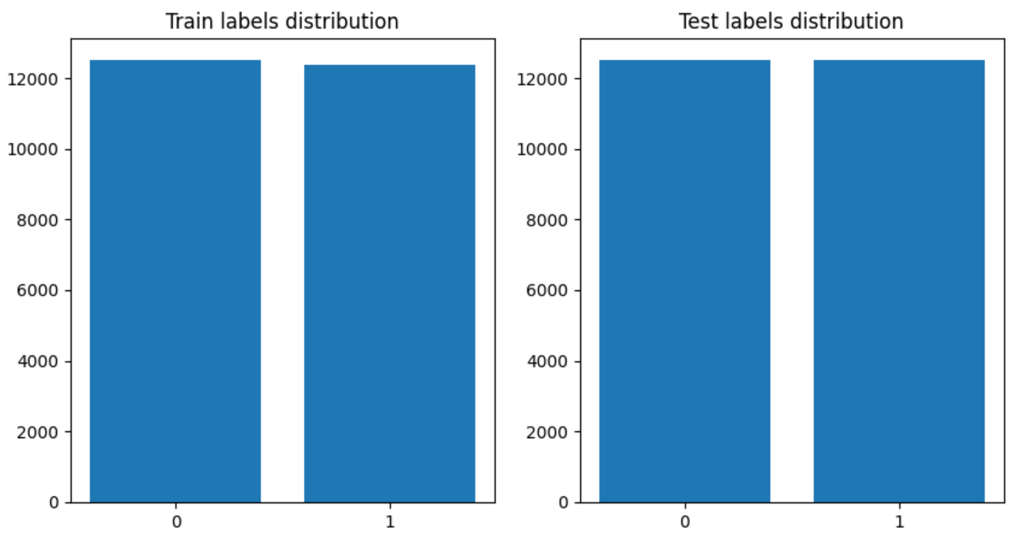
Data description

IMDB dataset having 50,000 movie reviews (positive and negative) and rating from 1 to 10. This is a dataset for binary and multiclass sentiment classification. It provides a set of 25,000 highly polar movie reviews for training and 25,000 for testing.

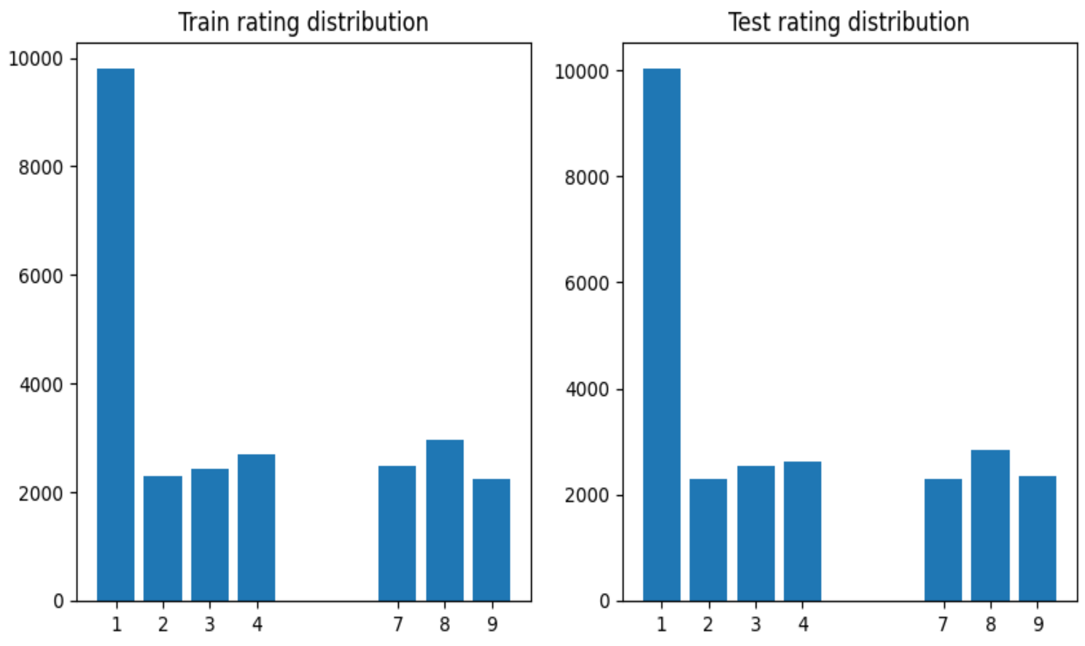
EDA

For binary classification we have balanced data

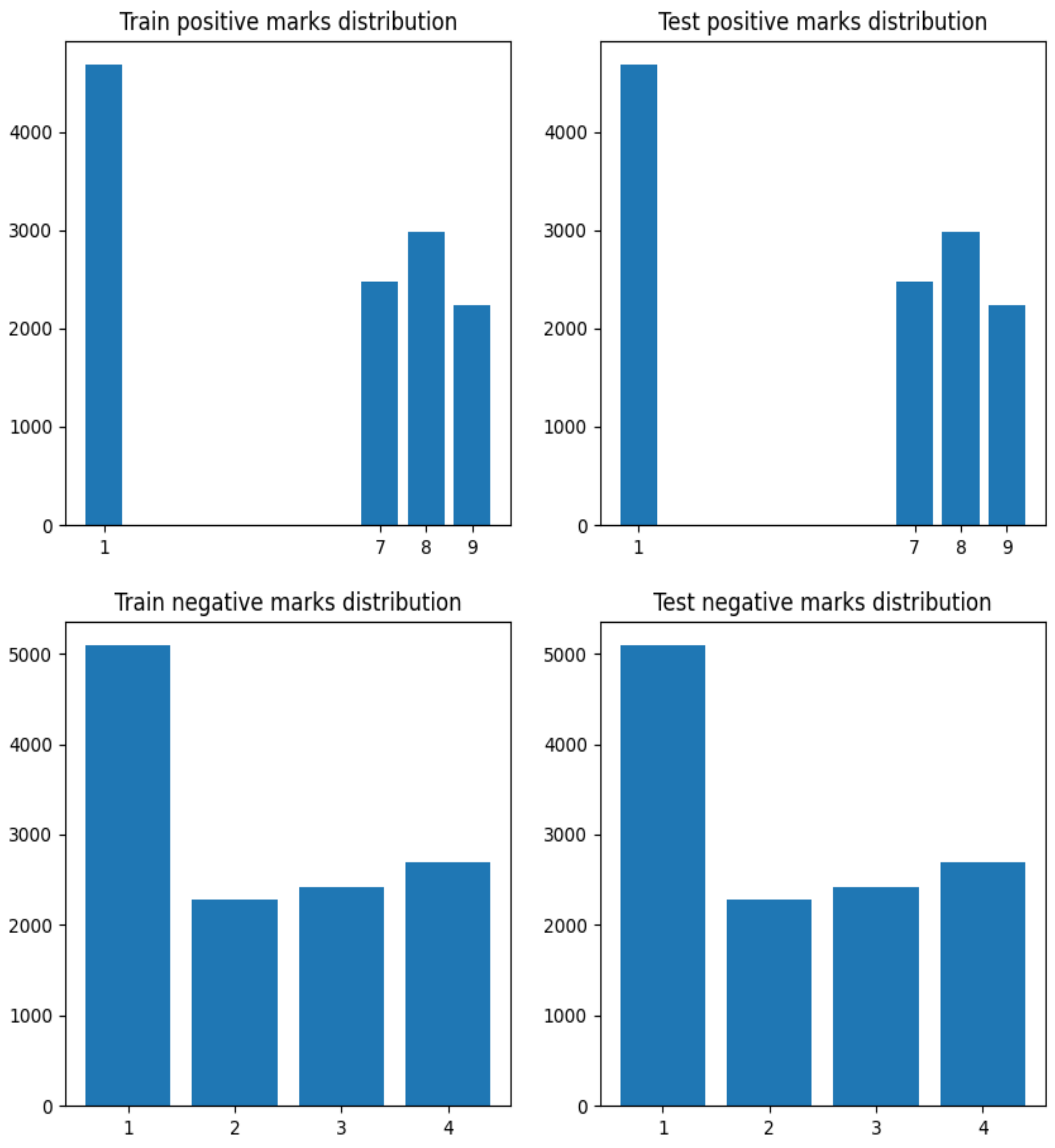


negative - 0
positive - 1

For rating classification data is imbalanced

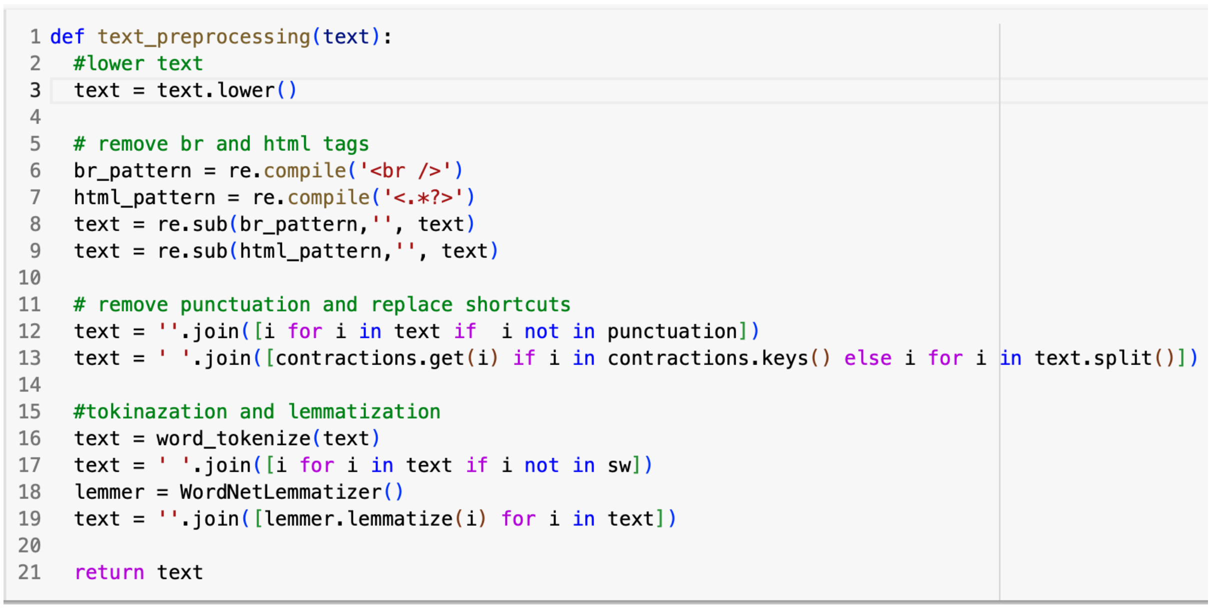


Positive and negative reviews have equal amount of rating 1 (approximately 5000), but positive reviews have a rest of rating from 7 to 9, negative - from 2 to 4.



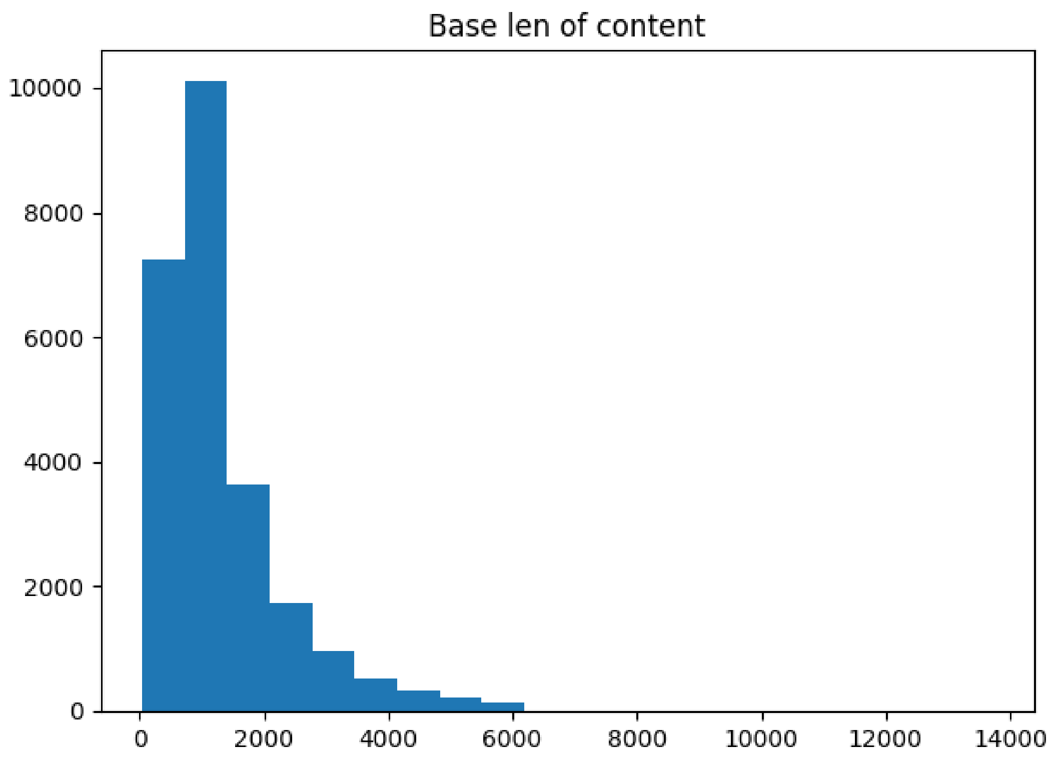
Data preprocessing

I tried to preprocess text with the custom preprocessing function below, but all my models perform much better without it.



```python
def text_preprocessing(text):
  #lower text
  text = text.lower()

  # remove br and html tags
  br_pattern = re.compile('<br />')
  html_pattern = re.compile('<.*?>')
  text = re.sub(br_pattern,'', text)
  text = re.sub(html_pattern,'', text)

  # remove punctuation and replace shortcuts
  text = ''.join([i for i in text if  i not in punctuation])
  text = ' '.join([contractions.get(i) if i in contractions.keys() else i for i in text.split()])

  #tokinazation and lemmatization
  text = word_tokenize(text)
  text = ' '.join([i for i in text if i not in sw])
  lemmer = WordNetLemmatizer()
  text = ''.join([lemmer.lemmatize(i) for i in text])

  return text
```

Distribution the len of context: the majority of reviews shorter than 2000 signs

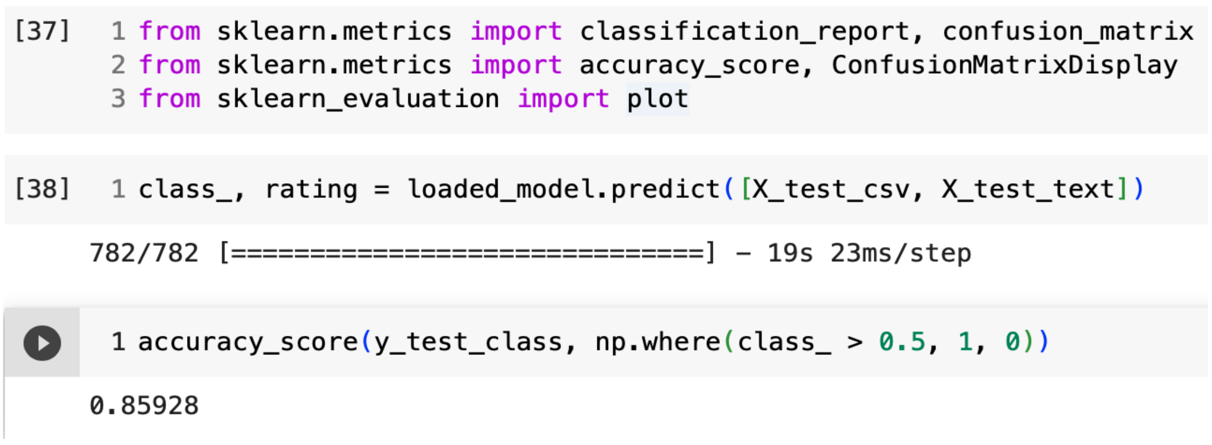


Features generation and modeling

I decided to generate extra-features from text and create multiple-input and multiple-output neural network. I created 8 features manually (like text len,punctuation amount and such) and 2 with textblob library.I put received data to csv format and write simple nn with few dense layers. For the text part I used GRU. I tried to use different options, including transfer learning with tensorflow-hub, but in the majority of situations it gives very similar results, so I decided to choose the fastests one.

Evaluation

For binary classification model gives 0.85928 accuracy

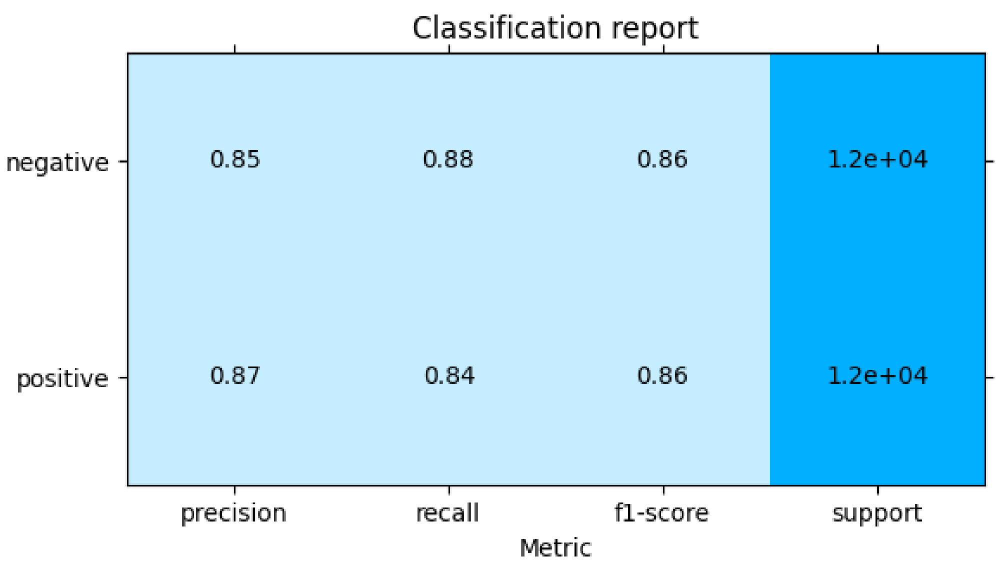


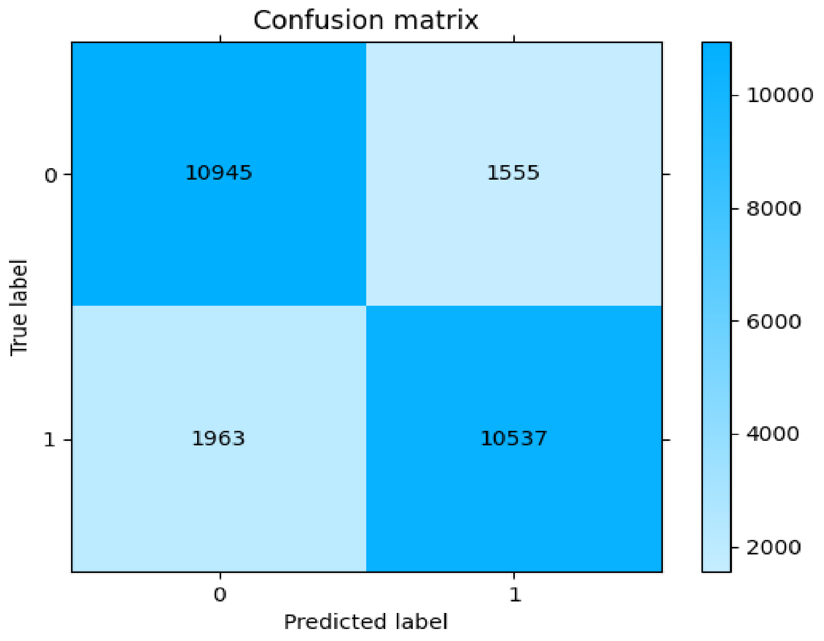
```
[37]  from sklearn.metrics import classification_report, confusion_matrix
      from sklearn.metrics import accuracy_score, ConfusionMatrixDisplay
      from sklearn_evaluation import plot
```

```
[38]  class_, rating = loaded_model.predict([X_test_csv, X_test_text])

      782/782 [==============================] - 19s 23ms/step
```

```
      accuracy_score(y_test_class, np.where(class_ > 0.5, 1, 0))

      0.85928
```

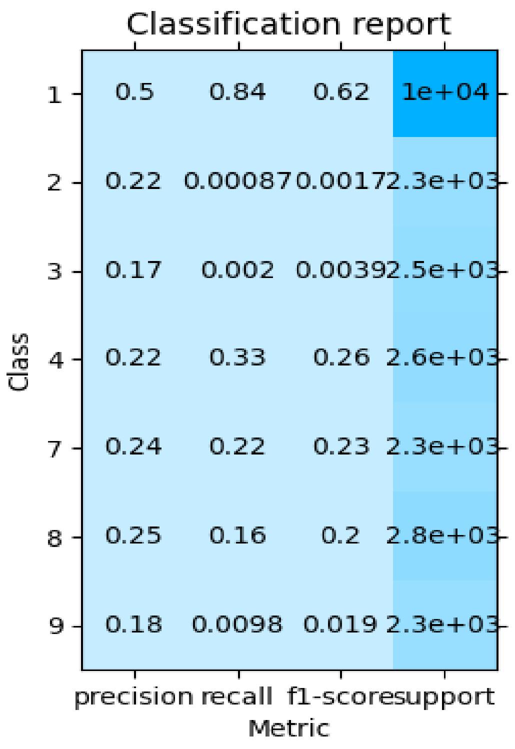
Hear we can see classification report and confusion matrix



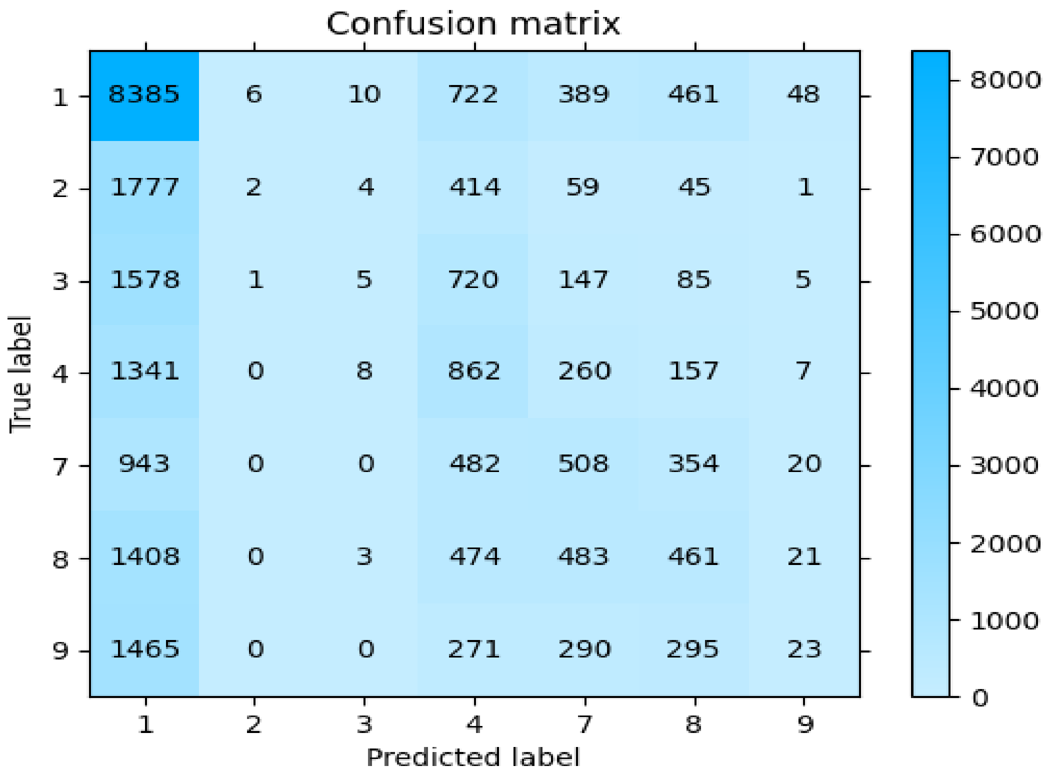


For rating classification it gives acceptable result for rating 1, but for the rest of the classes this model doesn't perform well.

Hear are classification report and confusion matrix


Classification report
Class
Metric

Class	precision	recall	f1-score	support
1	0.5	0.84	0.62	1e+04
2	0.22	0.00087	0.0017	2.3e+03
3	0.17	0.002	0.0039	2.5e+03
4	0.22	0.33	0.26	2.6e+03
7	0.24	0.22	0.23	2.3e+03
8	0.25	0.16	0.2	2.8e+03
9	0.18	0.0098	0.019	2.3e+03



	1	2	3	4	7	8	9
1	8385	6	10	722	389	461	48
2	1777	2	4	414	59	45	1
3	1578	1	5	720	147	85	5
4	1341	0	8	862	260	157	7
7	943	0	0	482	508	354	20
8	1408	0	3	474	483	461	21
9	1465	0	0	271	290	295	23

Summary

Model has a base result for reviews classification, but it is not suitable for rating classification.
I would be glad to hear how to improve my model as well.