NLP

Oscar Teeninga
296699

1. Wyniki treningu modeli na dostarczonym zbiorze danych

Uzyskana dokładność w przypadku dołączonych parametrów była równa 90.5%

```
[27] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:00<00:00, 319.09it/s]

100% 32/32 [00:01<00:00, 25.61it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8164913139221923, 'tp': 116, 'tn': 116, 'fp': 18, 'fn': 6, 'acc': 0.90625, 'eval_loss': 0.278163752052933}
```

Natomiast użycie hiper-parametrów pozwoliło uzyskać 91.4%

```
[15] import sklearn

     result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:04<00:00, 53.58it/s]

100% 32/32 [03:41<00:00, 6.91s/it]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8277465133349645, 'tp': 111, 'tn': 123, 'fp': 11, 'fn': 11, 'acc': 0.9140625, 'eval_loss': 0.30612796312198043}
```

1.1. Modyfikacja rozmiar batch

Dla *batch_size: 1* osiągnąłem dokładność 52.5%, co jest znacznie gorszym wynikiem.

```
[31] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:00<00:00, 389.88it/s]

100% 32/32 [00:01<00:00, 24.83it/s]
/usr/local/lib/python3.6/dist-packages/sklearn/metrics/_classification.py:900: RuntimeWarning: invalid value encountered in double_scalars
  mcc = cov_ytyp / np.sqrt(cov_ytyt * cov_ypyp)
INFO:simpletransformers.classification.classification_model:{'mcc': 0.0, 'tp': 0, 'tn': 134, 'fp': 0, 'fn': 122, 'acc': 0.5234375, 'eval_loss': 0.9228930957615376}
```

Dla *batch_size: 16* osiągnąłem dokładność 90.6%, a więc spodziewam się, że optymalny rozmiar będzie gdzieś pomiędzy.

```
[46] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:03<00:00, 78.70it/s]

100% 16/16 [00:01<00:00, 13.37it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8214111636628237, 'tp': 119, 'tn': 113, 'fp': 21, 'fn': 3, 'acc': 0.90625, 'eval_loss': 0.27530812472105026}
```

Dla *batch_size: 12* osiągnąłem najlepszy wynik 92.6%.

```
[50] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:03<00:00, 80.79it/s]

100% 22/22 [00:01<00:00, 16.36it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8549852197240615, 'tp': 118, 'tn': 119, 'fp': 15, 'fn': 4, 'acc': 0.92578125, 'eval_loss': 0.23207193917848848}
```

1.2 Modyfikacja rozmiaru epochs

Dostarczony zestaw parametrów definiował ten parametr jako 1 powyższe testy były przeprowadzone dla takiej wartości epochs i *batch_size: 12.* Ze względu na problemy z czasowym działaniem obliczeń ograniczyłem się tutaj do mniejszej ilości testów.

```
[50] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:03<00:00, 80.79it/s]

100% 22/22 [00:01<00:00, 16.36it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8549852197240615, 'tp': 118, 'tn': 119, 'fp': 15, 'fn': 4, 'acc': 0.92578125, 'eval_loss': 0.23207193917848848}
```

Dla *epochs: 2* osiągnąłem nieco gorszy wynik 92.2%

```
[58] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:02<00:00, 85.79it/s]

100% 22/22 [00:01<00:00, 15.19it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8440483135045428, 'tp': 114, 'tn': 122, 'fp': 12, 'fn': 8, 'acc': 0.921875, 'eval_loss': 0.2476541942239485}
```

Dla *epochs: 3* osiągnąłem 92.6%, a więc wynik pośredni.

```
[54] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:03<00:00, 67.05it/s]

100% 22/22 [00:01<00:00, 16.19it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.851207116872585, 'tp': 112, 'tn': 125, 'fp': 9, 'fn': 10, 'acc': 0.92578125, 'eval_loss': 0.26436068079519}
```

Dla *epochs: 8* i *batch_size: 64*, ponieważ dla mniejszego size trwałoby to zbyt długo. Osiągnięty wynik to 92.2%, więc całkiem nieźle, ale nie udało się poprawić najlepszego wyniku.

```
[67] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100% 256/256 [00:00<00:00, 287.12it/s]

100% 4/4 [00:01<00:00, 3.73it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.8440483135045428, 'tp': 114, 'tn': 122, 'fp': 12, 'fn': 8, 'acc': 0.921875, 'eval_loss': 0.34635376930236816}
```

1.3 Wnioski

Udało się nieznacznie poprawić o nieco ponad 1 pkt. procentowy. Jest to zadawalająca wartość, jednak większość testów dawało mniejszą dokładność, a więc można sądzić, że jesteśmy blisko doskonałej parametryzacji dla naszego zbioru treningowego.

2. Własny zbiór treningowy

Mój zbiór składał się z opisów dwóch rodzai filmów/seriali - anime oraz filmy dokumentalne. Dosyć duża rozbieżność tematyk powinna dać dobre wyniki klasyfikacji. Opis filmów/seriali anime pobierane były ze strony www.shinden.pl, natomiast dokumentalnych z http://www.repozytorium.fn.org.pl. Dane były identyfikowane po id, więc do adresu wystarczyło dokleić liczbę.

```
Index(['type', 'description'], dtype='object')
Film dokumentalny    1100
Anime                1100
Name: type, dtype: int64
Index(['type', 'description'], dtype='object')
1    882
0    878
Name: type, dtype: int64
0    222
1    218
Name: type, dtype: int64
```

Pobrałem 1100 rekordów z obu kategorii. Niestety dane musiały być pobierane szeregowo, ponieważ w innym przypadku liczba zapytań do serwera była na tyle częsta, że dostawałem blokadę, więc pobranie całego zbioru trwało kilka godzin (większość id było puste).
LINK: https://drive.google.com/file/d/1pD3_-2et4HQwCIvwThG1zfPC32f-8eV8/view?usp=sharing

2.1 Klasyfikacji przy pomocy prostych metod

Przy pomocy wagowania TF*IDF otrzymałem bardzo przyzwoite wyniki:

```
Applying best classifier on test data:
              precision    recall  f1-score   support

           0       1.00      0.96      0.98       222
           1       0.96      1.00      0.98       218

    accuracy                           0.98       440
   macro avg       0.98      0.98      0.98       440
weighted avg       0.98      0.98      0.98       440

[Parallel(n_jobs=3)]: Done  36 out of  36 | elapsed:    9.2s finished
```

2.2 Klasyfikacja przy pomocy Modelu Roberta

Przy pomocy modelu Roberta osiągnąłem zadowalającą dokładność klasyfikacją równą 99.6%:

```
[26] result, model_outputs, wrong_predictions = cls_model_2.eval_model(test_df, acc=sklearn.metrics.accuracy_score)

INFO:simpletransformers.classification.classification_model: Converting to features started. Cache is not used.
100%  440/440 [00:01<00:00, 316.60it/s]

100%  55/55 [00:01<00:00, 41.22it/s]
INFO:simpletransformers.classification.classification_model:{'mcc': 0.9909023230424111, 'tp': 235, 'tn': 203, 'fp': 2, 'fn': 0, 'acc': 0.9954545454545455, 'eval_loss': 0.037341278926892714}
```

2.3 Wnioski

Dokładność klasyfikacji przerosła moje oczekiwania. Te dwa zbiory różnią się znacząco od siebie, nie bez przyczyny jest również używanie słowa “anime” w opisach anime i “film” w filmach dokumentalnych, jednak nie zmienia to faktu, że udało się uzyskać bardzo dobre wyniki.