

Podstawy analizy języka naturalnego

Celem ćwiczenia jest zapoznanie się z prostymi metodami analizy języka naturalnego. W tym celu będziemy wykorzystywać metodę Bag of Words, zaimplementowaną jako klasa CountVectorizer pakietu sci-kit learn.

Będziemy analizować bazę danych zawierającą informację o filmach. Interesujące nas tabele to:

```
genres ( genre_id bigint, name text COLLATE pg_catalog."default" )
```

genres_movies (id bigint, movie_id bigint, genre_id bigint)

```
movies ( movie_id bigint, plot text COLLATE pg_catalog."default", movie_name text COLLATE pg_catalog."default" )
```

Dostęp do bazy umożliwia

```
2 import sqlalchemy
import pandas as pd
from sqlalchemy import create_engine
db_string = "postgresql://wbauer_abd:adb2020@pgsql-196447.vipserv.org:5432/wbauer_classifier_abd"
db = create_engine(db_string)
```

1. Ponieważ baza jest dość duża należy ograniczyć się do 10% danych (wybranych losowo):

```
SELECT setseed(0.5);
SELECT ...
FROM movies
JOIN genres_movies ON ...
JOIN genres ON ...
WHERE random()>0.9
```

Zapytanie uzupełnic samodzielnie tak, aby uzyskać, tytuł, opis fabuły i gatunek filmu.

```
3 select_string="SELECT setseed(0.5); " \
    "SELECT * FROM movies " \
    "JOIN genres_movies ON movies.movie_id = genres_movies.movie_id " \
    "JOIN genres ON genres.genre_id = genres_movies.genres_movies.genre_id WHERE random()>0.9"
```

```
df=pd.read_sql(select_string,con=db)
print(df.columns.values)
df

['movie_id' 'plot' 'movie_name' 'id' 'movie_id' 'genre_id' 'genre_id']
```

3		movie_id	plot	movie_name	id	movie_id	genre_id	genre_id	nar
---	--	----------	------	------------	----	----------	----------	----------	-----

	movie_id	plot	movie_name	id	movie_id	genre_id	genre_id	nar
0	11167	In the town of Flåklypa , the inventor Reodo...	Flåklypa Grand Prix	20170	11167	154	154	Stop motion
1	28269	On the morning of June 6, 1944, the beginning ...	Saving Private Ryan	5040	28269	110	110	Ensemk Film
2	29785	A group of rogue Force Recon Marines led by di...	The Rock	13978	29785	12	12	Thriller
3	30006	Clarice Starling is pulled from her training ...	The Silence of the Lambs	36231	30006	12	12	Thriller
4	30327	In a post-apocalyptic 2029, artificially inte...	The Terminator	28068	30327	12	12	Thriller
...
4246	36425670	Tomchi is a story of six notorious children wh...	Tomchi	21929	36425670	0	0	Drama
4247	36476665	The films deal with a woman named Carrie , a c...	Every 9 Seconds	21415	36476665	0	0	Drama
4248	36601633	Sam Chao , an ambitious New York attorney, is ...	Shanghai Calling	40301	36601633	0	0	Drama

	movie_id	plot	movie_name	id	movie_id	genre_id	genre_id	nar
4249	36724042	When two researchers discover a colossal shark...	Shark Attack 3: Megalodon	20500	36724042	55	55	Fantasy
4250	37478048	Anand Verma, a widower and father of a child, ...	Mr. Bechara	33221	37478048	33	33	Comed film

4251 rows × 8 columns

2. Spośród 10 najpopularniejszych gatunków filmów wybrać jeden (liczba liter w imieniu i nazwisku modulo 10). I utworzyć kolumnę przypisującą 0 gdy film jest tego gatunku i odwrotnie.

4 #gatunek 4

```
popular = df.value_counts('name')
gatunek = popular.index.tolist()[3]
print(gatunek)
my_category = pd.get_dummies(df,columns=['name'])
data = my_category[['movie_name','plot',f'name_{gatunek}']]
```

data

Crime Fiction

4

	movie_name	plot	name_Crime Fiction
0	Flåklypa Grand Prix	In the town of Flåklypa , the inventor Reodo...	0
1	Saving Private Ryan	On the morning of June 6, 1944, the beginning ...	0
2	The Rock	A group of rogue Force Recon Marines led by di...	0
3	The Silence of the Lambs	Clarice Starling is pulled from her training ...	0
4	The Terminator	In a post-apocalyptic 2029, artificially inte...	0
...
4246	Tomchi	Tomchi is a story of six notorious children wh...	0
4247	Every 9 Seconds	The films deal with a woman named Carrie , a c...	0
4248	Shanghai Calling	Sam Chao , an ambitious New York attorney, is ...	0
4249	Shark Attack 3: Megalodon	When two researchers discover a colossal shark...	0
4250	Mr. Bechara	Anand Verma, a widower and father of a child, ...	0

4251 rows × 3 columns

3. Utworzyć model bag of words, usuwając akcenty, stopwords i zmniejszając wszystkie litery na małe. Ograniczyć się do 20 najpopularniejszych słów.

```
24 from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(strip_accents='ascii',lowercase=True,max_features = 20)
mat = vectorizer.fit_transform(data['plot'])
words = vectorizer.get_feature_names_out()

print(f'Lista najczęściej występujących słów:\n{words}')
print(f'Wymiar zwrótnego dopasowania:\n{mat.shape}')

Lista najczęściej występujących słów:
as
Wymiar zwrótnego dopasowania:
(4251, 20)
```

Wnioski: max_features pozawa określić które słowa powtarzają się najczęściej w całej kolumnie plot podanej jako argument w fit_transform następnie dla każdej komórki dataframe'u zwraca 20 krotek zawierających parę liczb, indeks słowa w liście features oraz ilość wystąpień w komórce ,co łatwo dostrzec wypisując wymiar zwróconego wektora.

4. Przekształcić opisy filmów do danej postaci za pomocą metody .fit_and_transform
5. Dokonać podziału na zbiór uczący i testowy

```
16 from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(mat,data[f'name_{gatunek}'],test_size=0.33, random_state=42)
```

6. Utworzyć model regresji logistycznej i nauczyć go rozpoznawać wybrany gatunek filmu.
7. Przetestować go na zbiorze uczącym i obliczyć "confusion matrix"

```
18 from sklearn import linear_model as lin
from sklearn import metrics
from sklearn.metrics import confusion_matrix

reg = lin.LogisticRegression()
reg.fit(X_train,y_train)
y_prob = reg.predict_proba(X_test)
print(y_prob)
predicted_y = reg.predict(X_test)

print(f"Mean absolute error = {metrics.mean_absolute_error(y_test, predicted_y)}")
print(f"Mean squared error = {metrics.mean_squared_error(y_test, predicted_y)}")
print(f"Root mean squared error = {metrics.root_mean_squared_error(y_test, predicted_y, squared=False)}")
print(metrics.classification_report(y_test, predicted_y))

confusion_matrix(y_test,predicted_y,normalize='true')

[[0.95111358 0.04888642]]
```

```
[0.96670155 0.03329845]
[0.95846932 0.04153068]
...
[0.92000641 0.07999359]
[0.80012904 0.19987096]
[0.91997065 0.08002935]]
Mean absolute error = 17.811831789023522
Mean squared error = 0.0698503207412687
Root mean squared error = 0.26429211252186224
      precision    recall   f1-score   support
0         0.93     1.00     0.96     1305
1         0.00     0.00     0.00      98
accuracy                           0.93     1403
macro avg       0.47     0.50     0.48     1403
weighted avg    0.87     0.93     0.90     1403
```

```
C:\Users\Sebastian\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\metrics\_classification.py:100: UserWarning: F1 score is ill-defined on a binary base classifier
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Sebastian\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\metrics\_classification.py:100: UserWarning: F1 score is ill-defined on a binary base classifier
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Sebastian\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\metrics\_classification.py:100: UserWarning: F1 score is ill-defined on a binary base classifier
  _warn_prf(average, modifier, msg_start, len(result))
```

```
18 array([[1., 0.],
       [1., 0.]])
```

8. Używając metody `.predict_proba()` obliczyć jakie prawdopodobieństwo klasyfikator obliczył dla wyników fałszywie dodatnich i fałszywie ujemnych. Wynik zwizualizować.

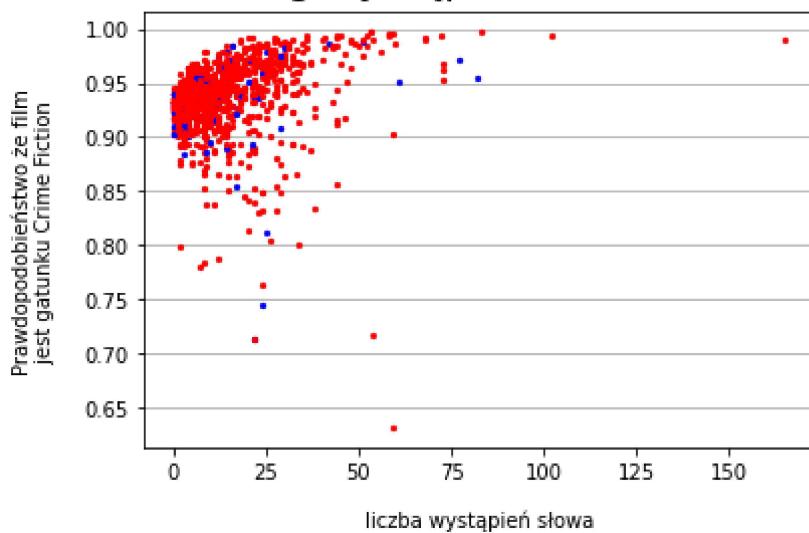
```
63 from matplotlib import patches
import matplotlib.pyplot as plt
import numpy as np

y_proba = reg.predict_proba(X_test)

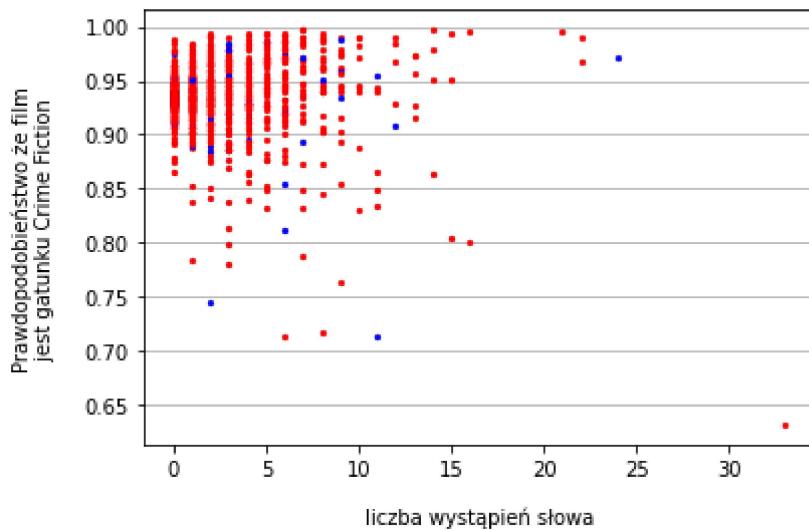
for i in range(20):
    plt.figure()
    tr = X_test.toarray()[:, i]
    trp = y_proba[:, 0]
    col=[]

    for j in range(0,len(tr)):
        if trp[j] > 0.5 and y_test.values[j]:
            col.append('b')
        else:
            col.append('r')
    for k in range(len(tr)):
        ax = plt.scatter(tr[k], trp[k], c = col[k], s = 5)
    plt.rc('axes', axisbelow=True)
    plt.title(f"Przewidywanie kategorii {gatunek} \n wedlug wystąpienia słowa '{words[i]}'", y=1.015,
    plt.xlabel("liczba wystąpień słowa", labelpad=14)
    plt.ylabel(f"Prawdopodobieństwo że film\n jest gatunku {gatunek} ", labelpad=14)
    plt.grid(axis = 'y' )
    bbox_props_decision_threshold = dict(boxstyle="round", fc="snow", ec="0.8", alpha=0.8)
    plt.show()
```

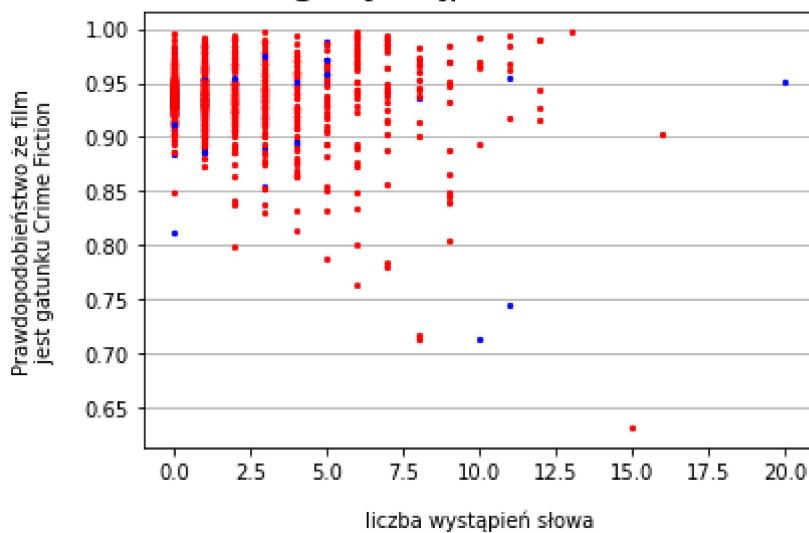
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'and'



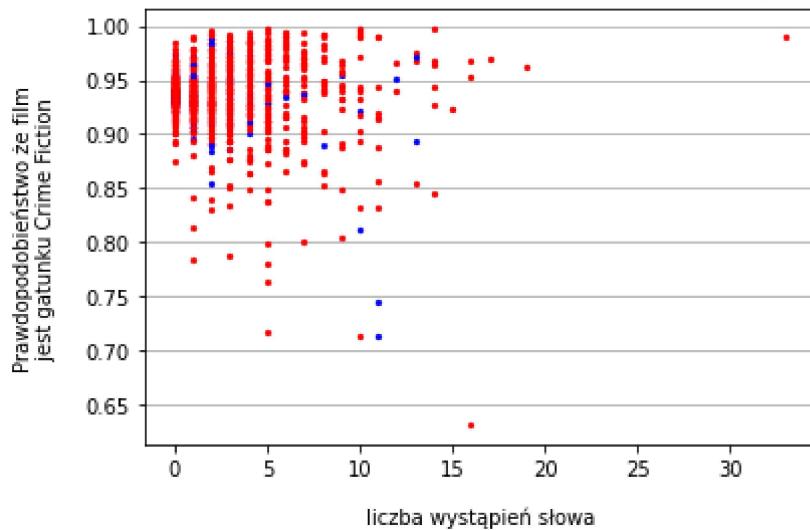
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'as'



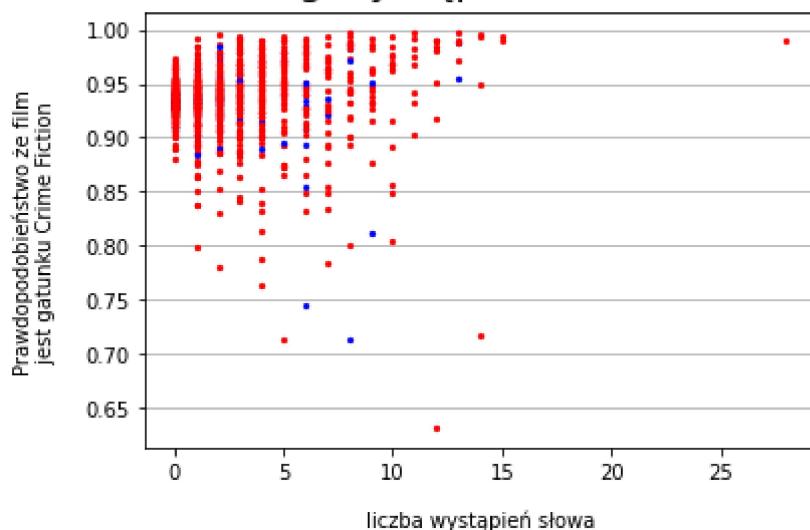
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'but'



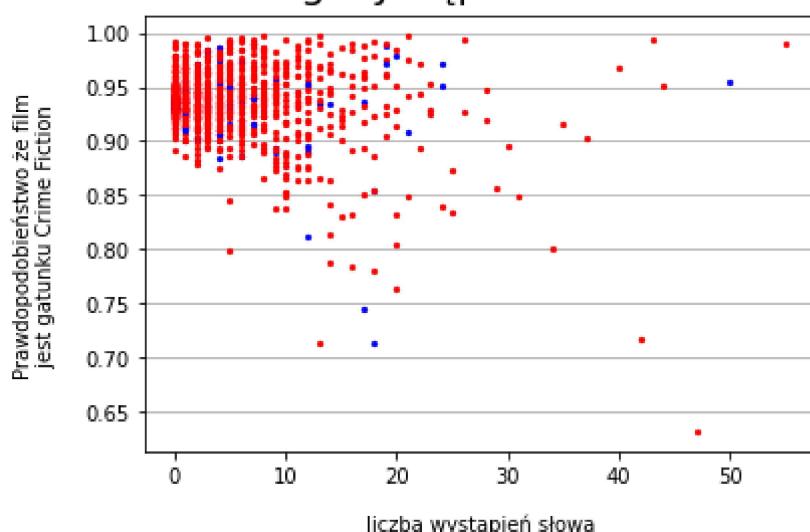
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'by'



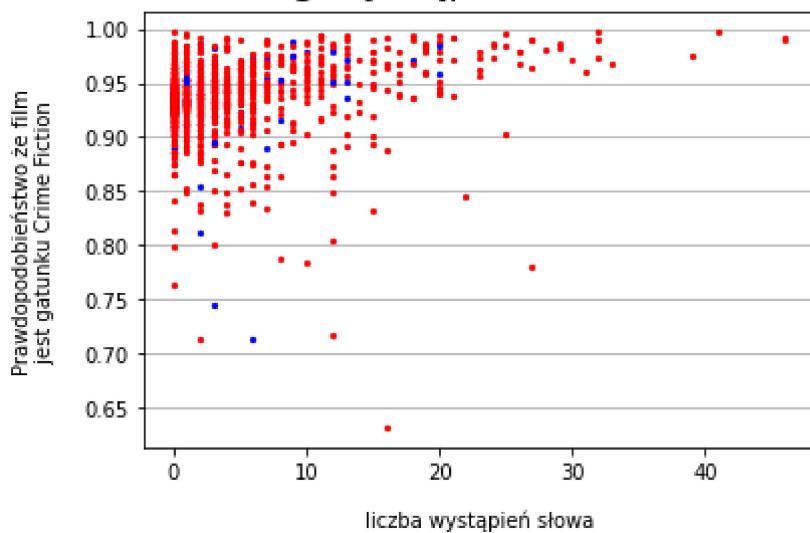
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'for'



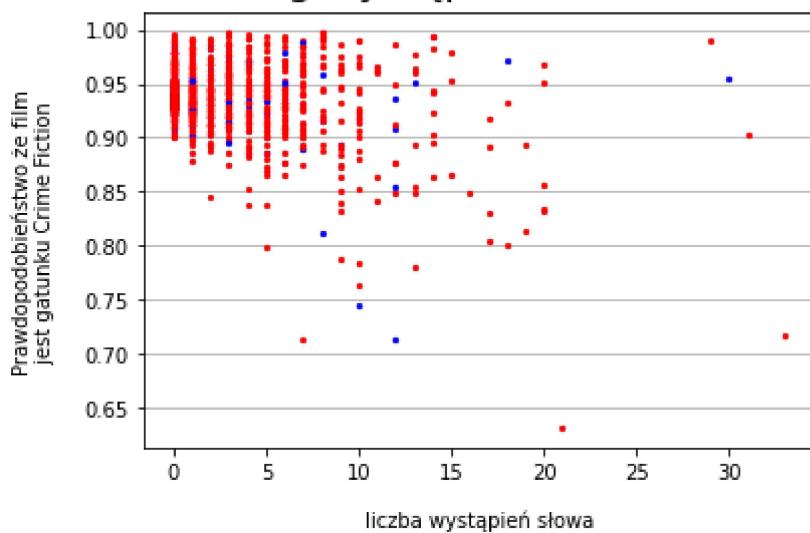
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'he'



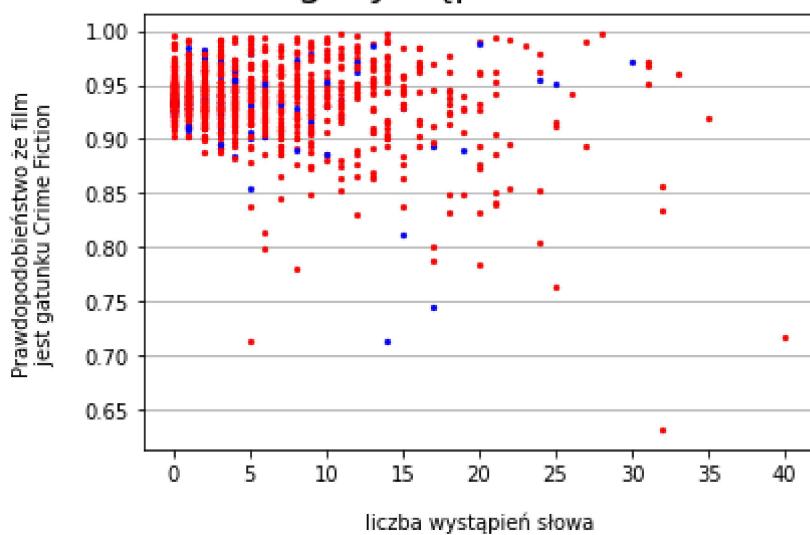
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'her'



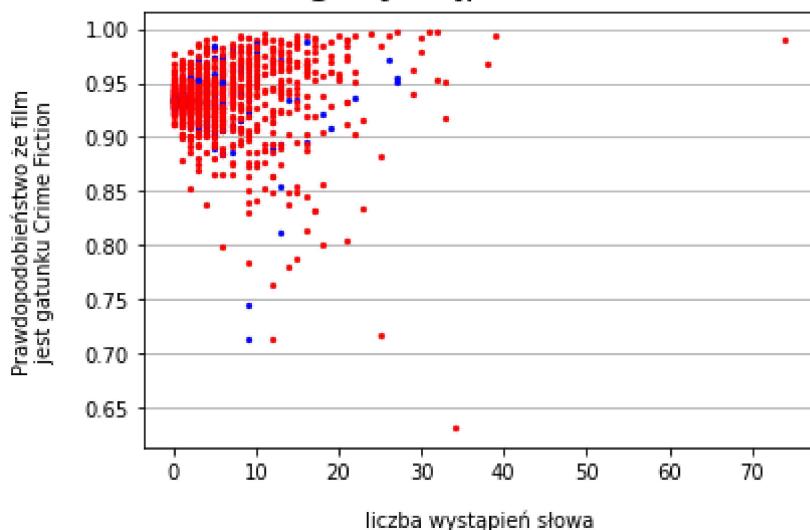
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'him'



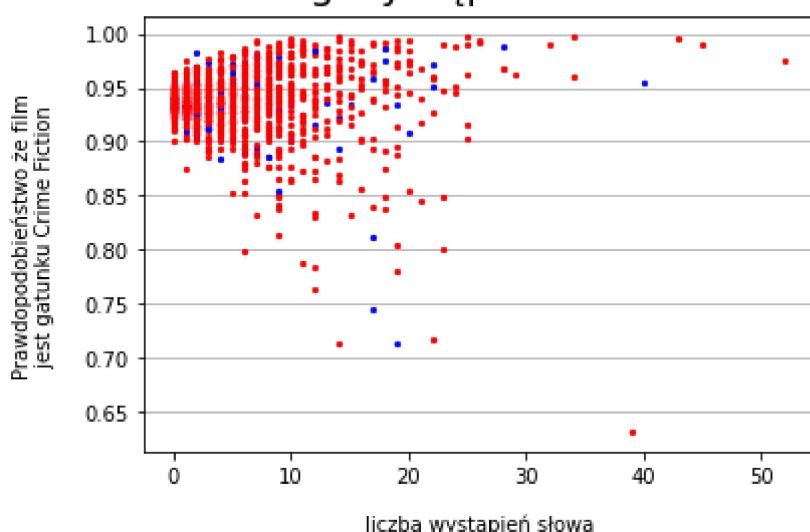
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'his'



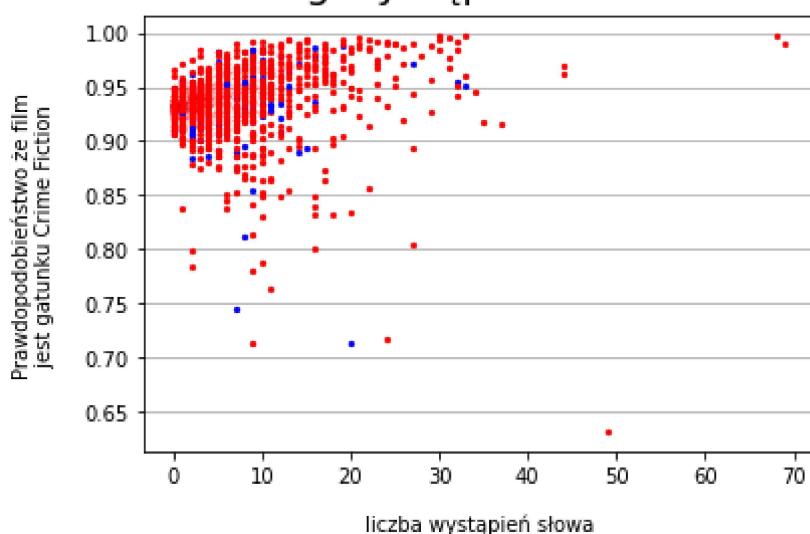
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'in'



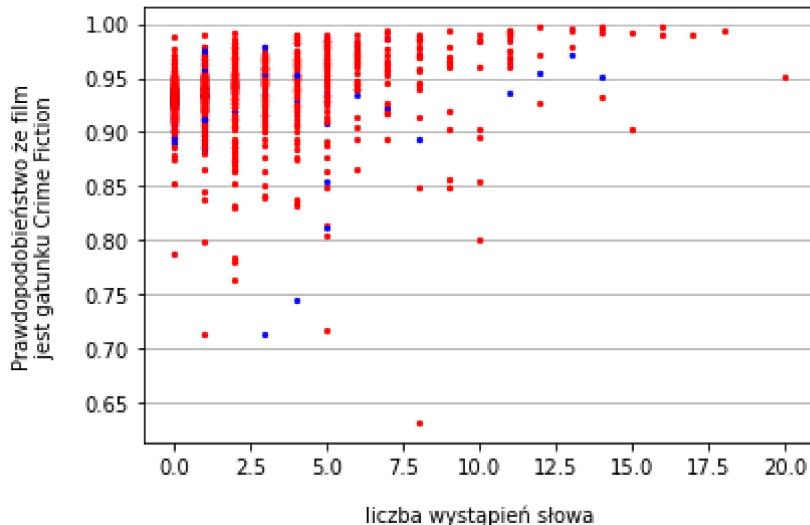
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'is'



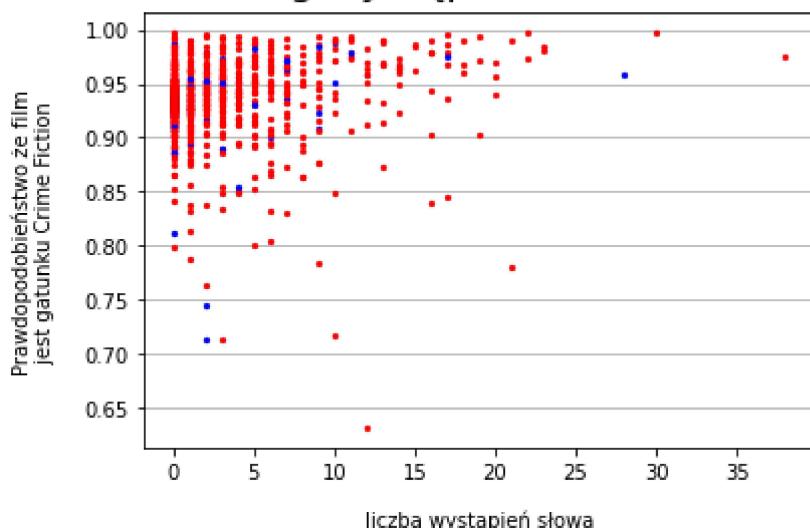
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'of'



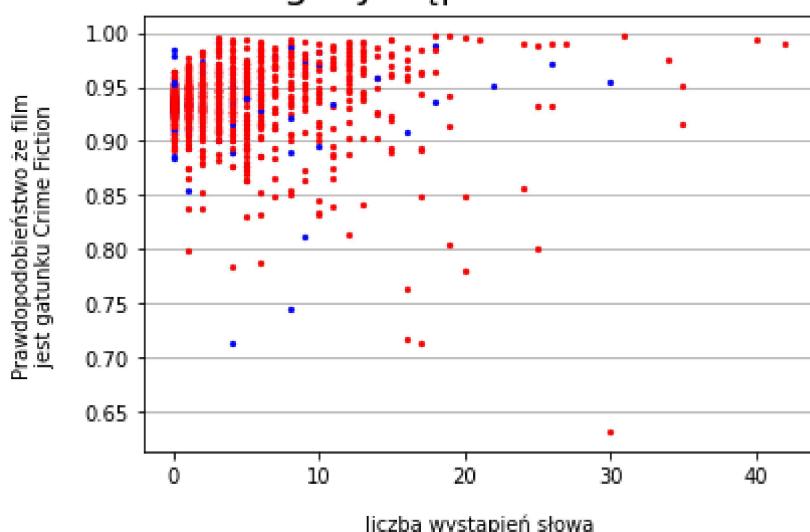
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'on'



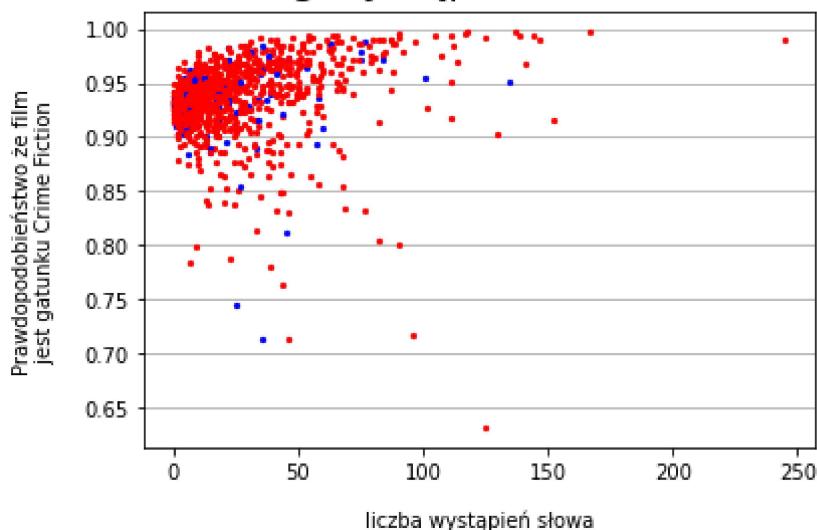
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'she'



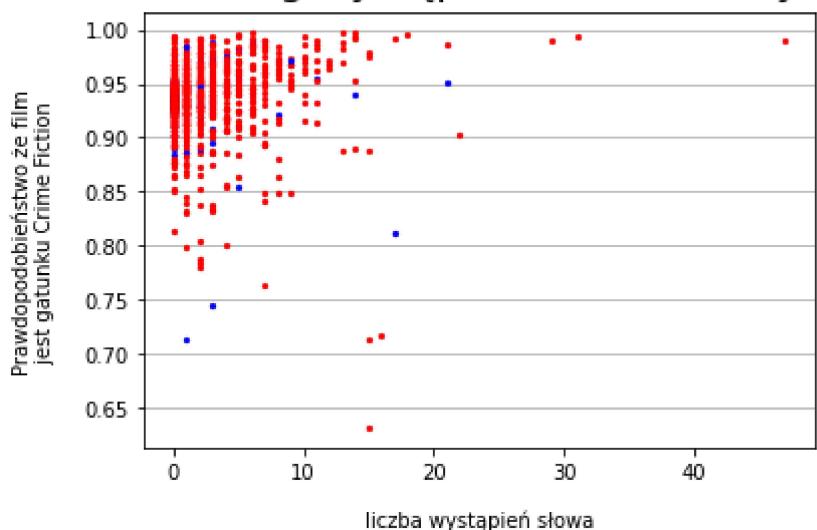
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'that'



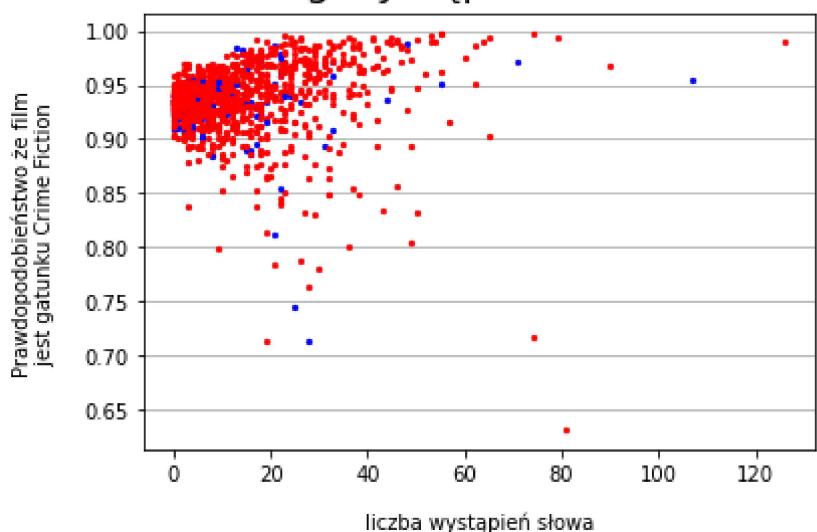
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'the'



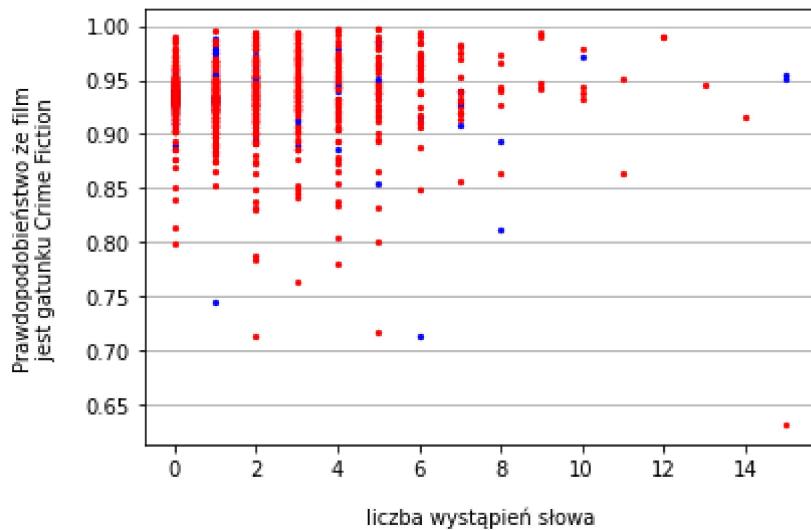
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'they'



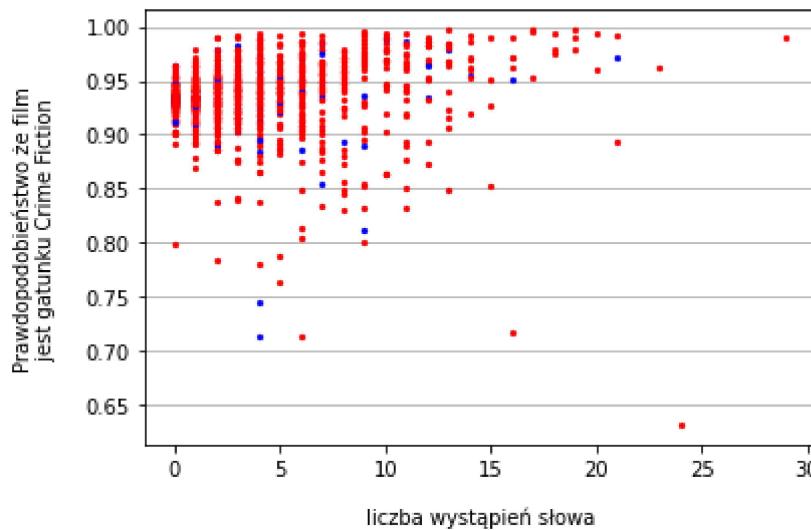
Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'to'



Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'who'



Przewidywanie kategorii Crime Fiction według wystąpienia słowa 'with'



Wnioski: Na powyższych wykresach wyniki zaznaczone kolorem czewonym dają błędne rozwiązania natomiast niebieskie są poprawne, niska skuteczność przewidywania wyników, jest najpewniej spowodowana błędym doborem bazy słów, najpopularniejsze słowa z całego zbioru danych nie są słowami kluczowymi dla żadnej kategorii, najczęściej są to spójniki lub formy osobowe, dlatego są nieodpowiednią bazą treningową.

