Random Acts of Pizza

Predicting Altruism Through Free Pizza

Random Acts of Pizza

- Thread na Redditu: Random Acts of Pizza
  - „I'll write a poem, sing a song, do a dance, play an instrument, whatever! I just want a pizza”
- Kaggle.com
  - Predicting Altruism Through Free Pizza
  - Sakupljeni podaci 2010-2013
  - How to Ask for a Favor: A Case Study on the Success of Altruistic Requests (Althoff et al.)

Supervised Classification

Pristup problemu metodom strojnog učenja s nadzorom

Nadzirano učenje

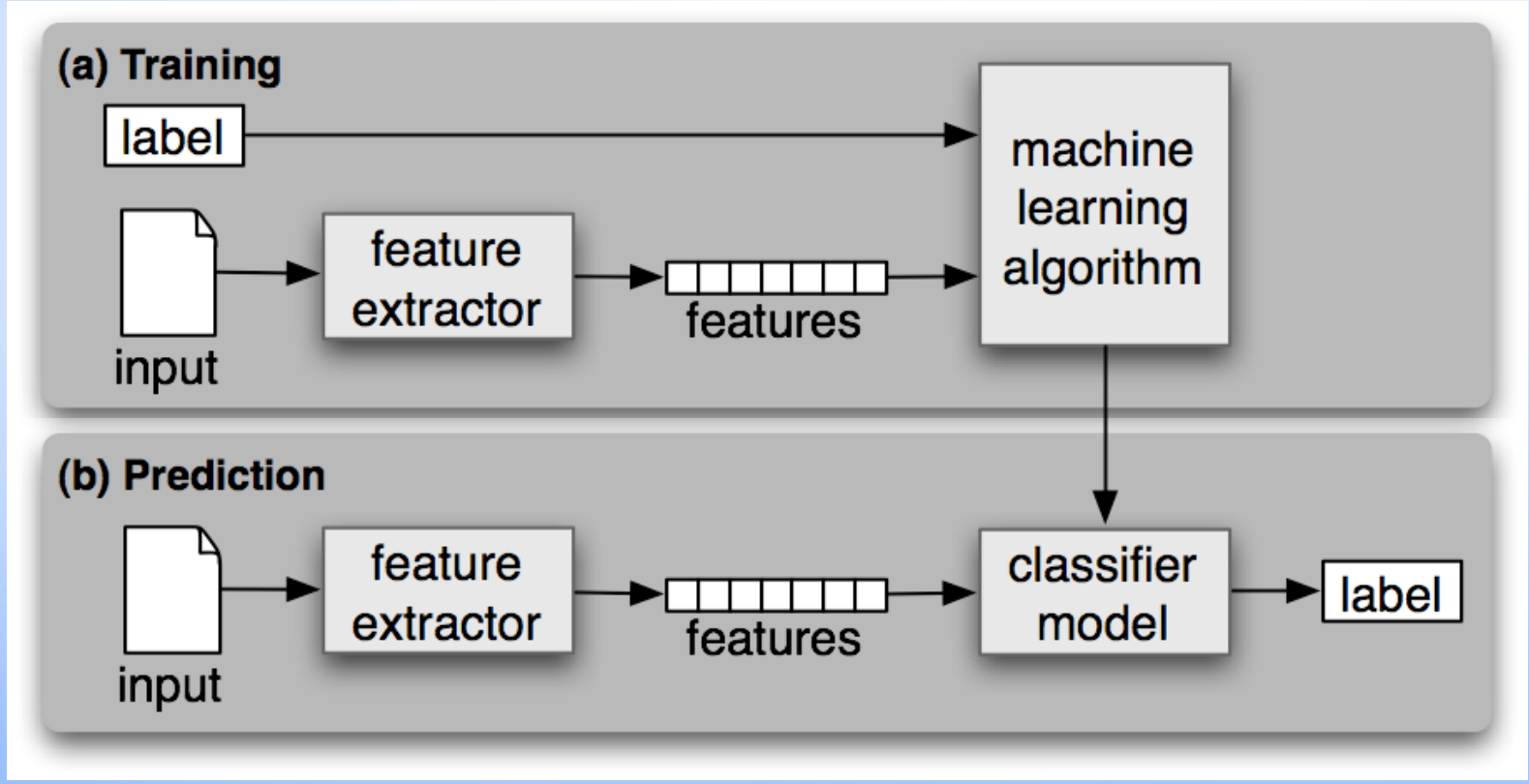
- Učenje na riješenim primjerima
- Podaci označeni predefiniranim klasama – klasifikacija teksta
- Naivni Bayesov algoritam
- 2 pristupa
  - NLTK
  - Scikit-learn

Obrada podataka

Dobiveni podaci

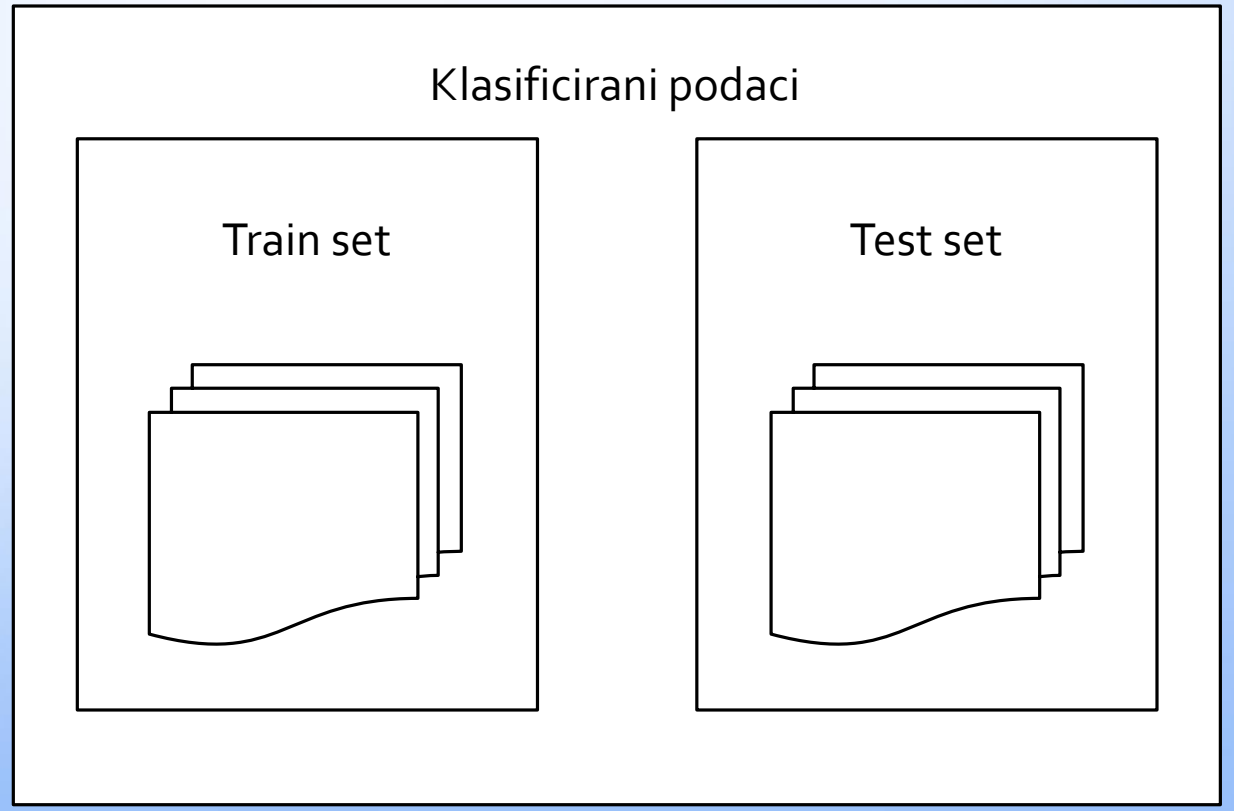
- 5671 zahtjev, 32 različita atributa
  - number_of_downvotes_of_request_at_retrieval
  - post_was_edited
  - request_number_of_comments_at_retrieval
  - request_text
  - request_title
  - …
- .json -> Python

Nadzirano učenje



Nadzirano učenje

- Učenje/treniranje – train set
- Testiranje – test set
- Reprezentativni skupovi



NLTK

- API za analizu jezika (Natural Language Toolkit)
- Programiramo feature extractor i koristimo NTLK Naive Bayes klasifikator
- 3 pristupa
  1. Uzeti u obzir sve atribute
  2. Uzeti u obzir selektirane atribute (kategorizirane vrijednosti)
  3. Obrada teksta – ključne riječi po kategorijama dobivene na temelju istraživanja u paperu

Rezultati – 1.

1. Svi atributi ~99%
   - overfitting

```
NLTK NaiveBayesAllAttributes, 1:  0.9923664122137404
Most Informative Features
requester_user_flair = 'null'           False : True   =    217.4 : 1.0
requester_upvotes_minus_downvotes_at_retrieval = '1' False : True =21.5:1.0
requester_number_of_posts_on_raop_at_retrieval = '4' True : False =
19.6:1.0
requester_upvotes_plus_downvotes_at_retrieval = '2' False : True  =17.7:1.0
requester_number_of_comments_in_raop_at_retrieval = '18'True:False=15.1:1.0
```

Rezultati – 2.

2. probrani atributi ~75%

```
NLTK NaiveBayesSomeAttributes, 2:  0.7563123899001761
Most Informative Features
requester_number_of_posts_on_raop_at_request = '"high"'True : False=8.6:1.0
request_number_of_comments_at_retrieval = '"average"'False : True=4.5 : 1.0
requester_number_of_comments_in_raop_at_request='"high"'True:False=2.9:1.0
requester_number_of_comments_in_raop_at_request='"average"'True:False =
2.4:1.0
number_of_upvotes_of_request_at_retrieval = '"low"'False : True=2.2 : 1.0
```

Rezultati – 3.

3. Obrada teksta >72%

```
NLTK NaiveBayesTextAnalysis, 3:  0.733998825601879
Most Informative Features
          contains(hire) = True              True : False  =      5.1 : 1.0
         contains(bucks) = True              True : False  =      3.2 : 1.0
         contains(drunk) = True             False : True   =      2.9 : 1.0
         contains(hired) = True              True : False  =      2.7 : 1.0
          contains(beer) = True             False : True   =      2.7 : 1.0
       contains(current) = True              True : False  =      2.7 : 1.0
           contains(mum) = True              True : False  =      2.6 : 1.0
        contains(father) = True              True : False  =      2.3 : 1.0
          contains(loan) = True              True : False  =      2.2 : 1.0
       contains(invited) = True             False : True   =      2.1 : 1.0
```

Scikit-learn

- Python modul za strojno učenje
- Bag of words metoda klasifikacije teksta
- Klase: CountVectorizer, TfidfTransformer, MultinomialNB

Bag of Words

- Dataset -> Document -> words
- Word corpus
- Matrica X[i][j] = broj pojavljivanja j-te riječi iz word corpusa u i-tom dokumentu

Rješenje

```
"""
CountVectorizer() is sklearn class from sklearn.feature_extraction.text module, used for extracting text features.
It is used to convert a collection of text documents to a matrix of token counts.
It's method fit_transform(raw_documents[,y]) learns the vocabulary dictionary and returns term-document matrix
"""
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(all_requests[:3970])
#control print of dimension of matrix of token counts
print(X_train_counts.shape)

"""
TfidfTrasnforemer() is another sklearn class from sklearn.feature_extraction.text module
It is used to transform a count matrix to a normalized tf or tf-idf representation
Tf stand for term-frequency, tf-idf stands for term-frequency times inverse document-frequency
Value of a word in count matrix is instead of occurrences changed for frequencies in tf-idf
representation and then it's weights are downscaled in tf-idf representation
We do this because words that occur in many documents in corpus (words as 'and', 'if', 'or', etc.)
are usually less informative than those occuring in smaller portions
"""
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

"""_____"""
""" Training classifier using multinomial variant of naive Bayes classfifier """
"""_____"""

classifier = MultinomialNB().fit(X_train_tfidf, all_outcomes[:3970])
```

Rezultat

- Uspjesnost: >75%

```
>>> ================================
>>>
The dataset contains 5671 samples.

(3970, 12586)
NBayes success:  0.761904761905
```

Usporedba i zaključak

NLTK

- Preciznost
  - Svi atributi: ~99%
  - Probrani atributi:~75%
  - Obrada teksta: >72%

+: jednostavnije, dobra preciznost bez analize teksta

- : overfitting

SKLEARN

- Preciznost
  - >75%

+: primjenjivo na druge probleme, popravljivo

- : smanjena preciznost

Što dalje?

- N-grami
- Grid-search – pronalazi najbolje parametre
- Druge metode i algoritmi:
  - SVM
  - RandomForests – dobar za klasifikaciju teksta, ne i za Bag of words

Hvala na pažnji!