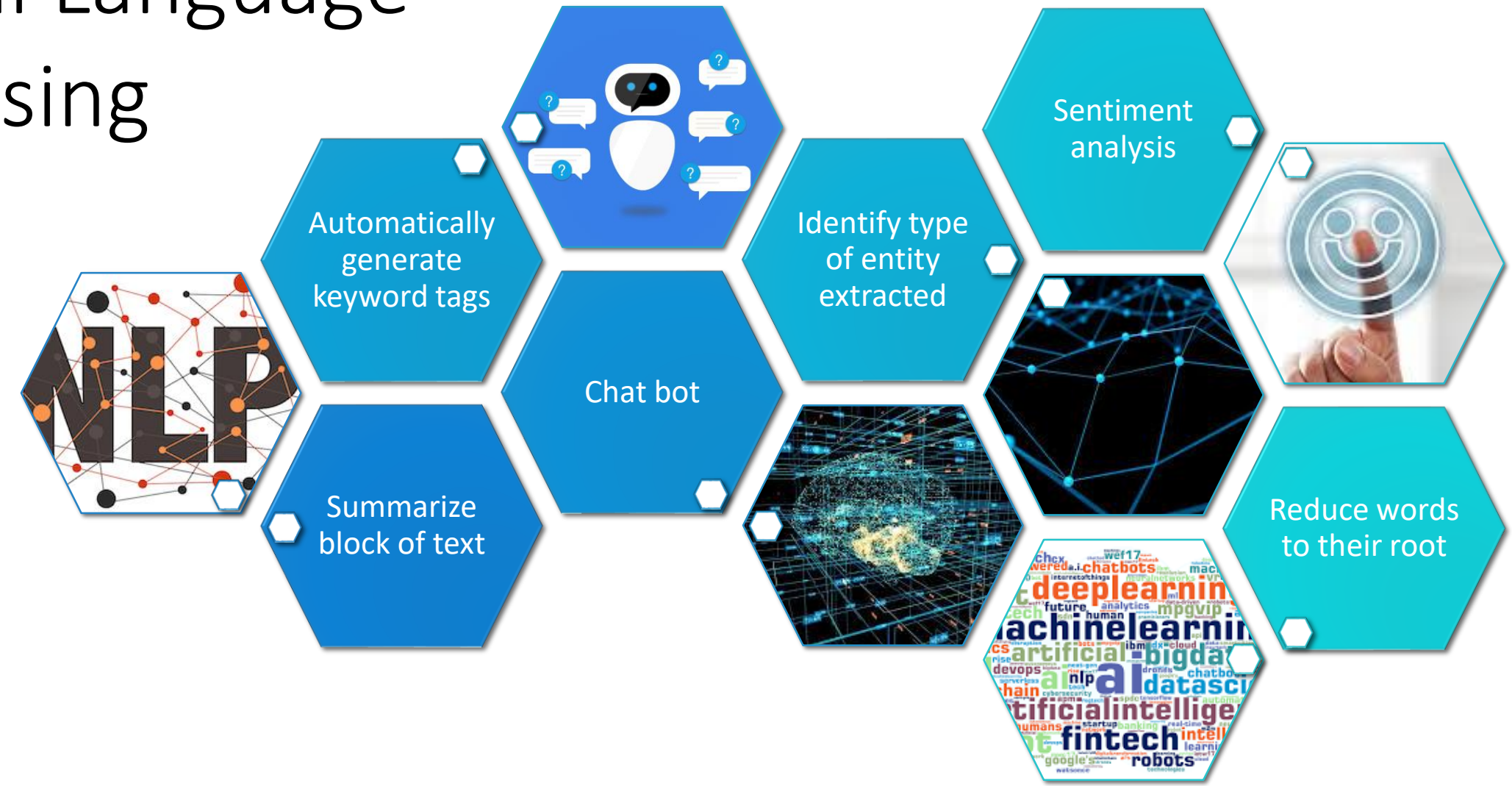
Sentiment analysis

TATIANA

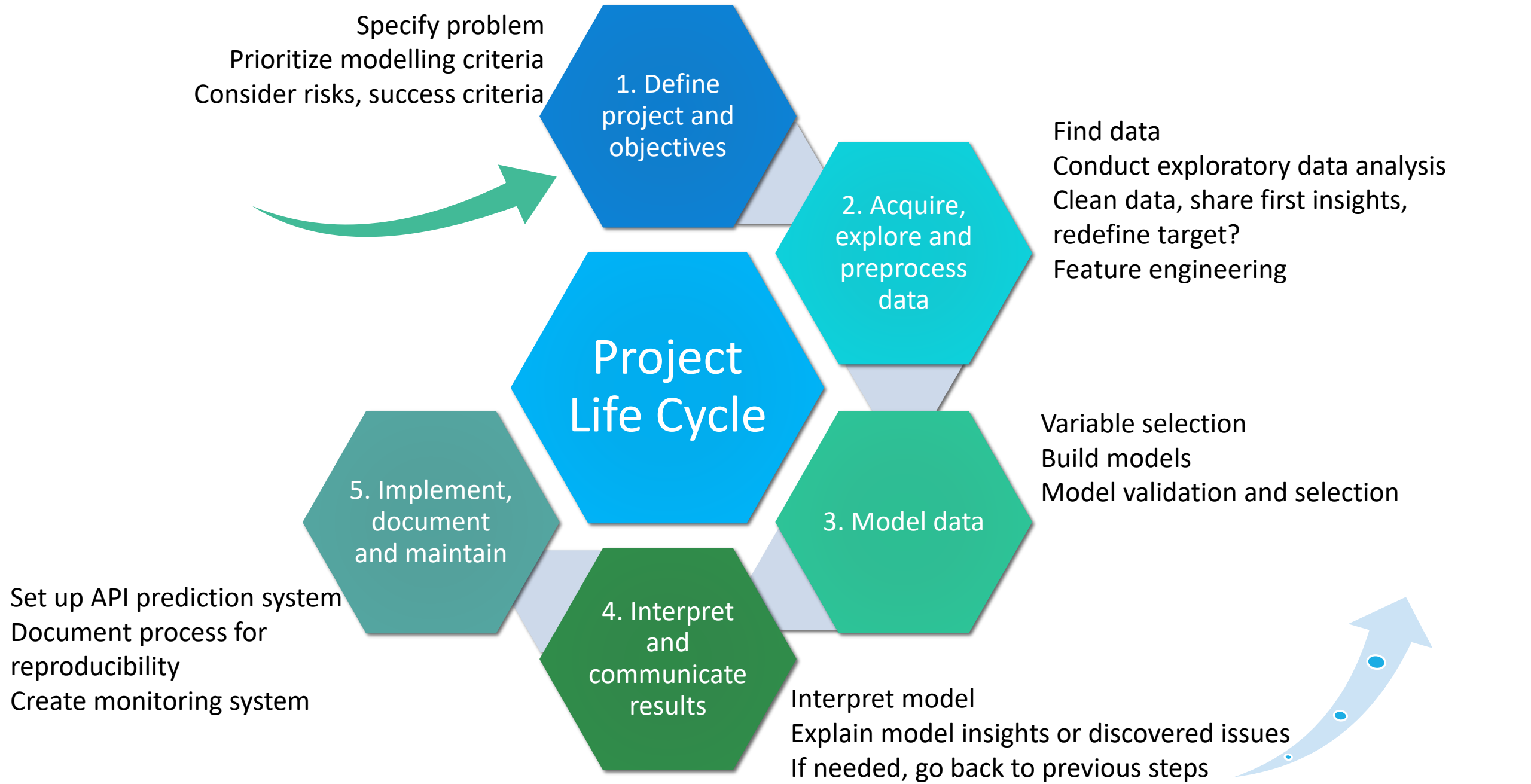
Natural Language Processing



Sentiment analysis

- Unbelievably disappointing
- Full of zany characters and richly applied satire, and some great plot twists
- This is the greatest screwball comedy ever filmed
- It was pathetic. The worst part about it was the boxing scenes.

Picture from https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/


Specify problem
Prioritize modelling criteria
Consider risks, success criteria
1. Define project and objectives
2. Acquire, explore and preprocess data
Find data
Conduct exploratory data analysis
Clean data, share first insights, redefine target?
Feature engineering
Project Life Cycle
3. Model data
Variable selection
Build models
Model validation and selection
4. Interpret and communicate results
Interpret model
Explain model insights or discovered issues
If needed, go back to previous steps
5. Implement, document and maintain
Set up API prediction system
Document process for reproducibility
Create monitoring system

Project
Movie review prediction

Build a movie review sentiment classifier

15,423 movie reviews with a score from 1 to 10 from IMDB

Data preprocessing

Remove punctuation, duplicated letters, numbers

Normalization = to *normalize* the words in the corpus by trying to convert all of the different forms of a given word into one

Stemming and *Lemmatization*

Stemming: basic rules to chop off the ends of words.

Lemmatization: identify the part-of-speech of a given word and then apply more complex rules to transform the word into its true root

Feature extraction

Features:

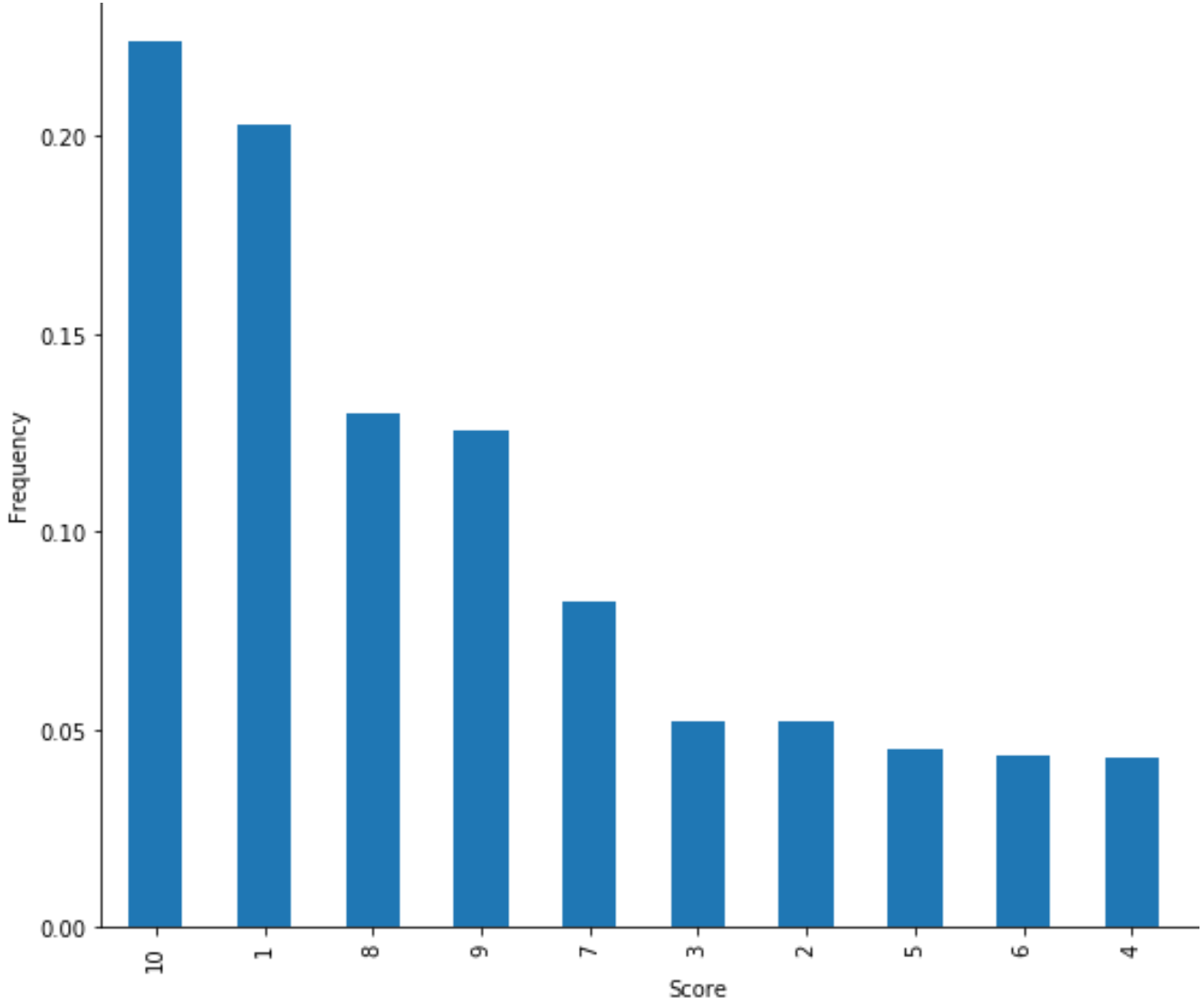
- words (bag-of-words)
- **n-grams**
- parts-of-speech (e.g. Adjectives and adjective-adverb combinations)
- opinion words (lexicon-based: dictionary or corpus)
- valence intensifiers and shifters (for negation); modal verbs; syntactic dependency

Feature selection based on

- frequency
- information gain
- odds ratio (for binary-class models)
- mutual information

Feature weighting

- term presence or term frequency
- frequency-inverse document frequency (TF-IDF)
- term position : e.g. title, first and last sentence(s)



Target value

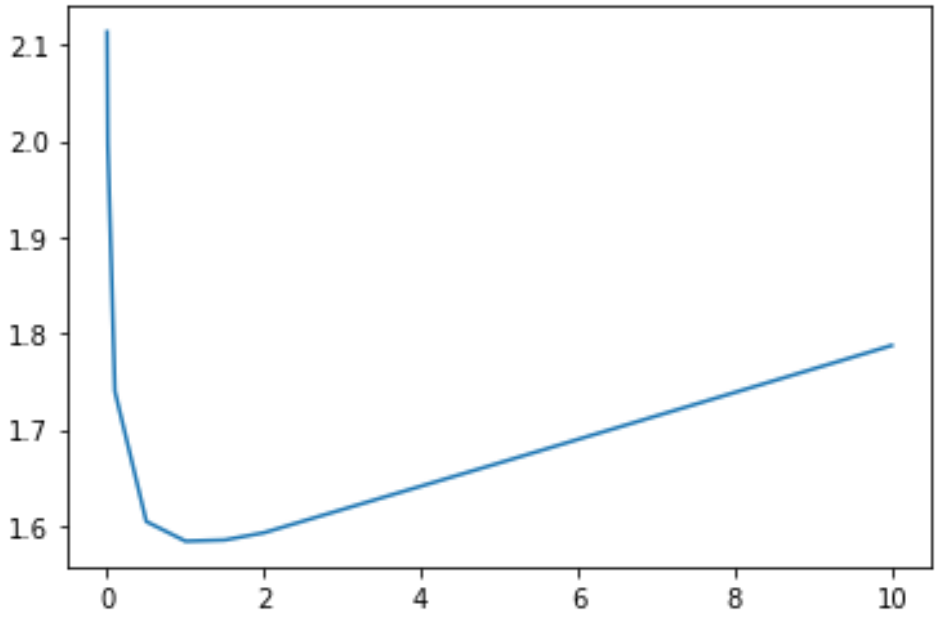
Unbalanced classes

More prone towards positive scores

After preprocessing, **68774** words in the dictionary

Classification / Regression

Baseline algorithms based on cross validation with 5 folds	
Ridge	1.911



Selection of optimal parameters **Alpha** in Ridge regression

Optimal parameters

with en- and bi-grams and set of English stop words

SVM

- C = 6
- class_weight='balanced’
- loss='squared_hinge’
- multi_class='ovr’

1.3381

GradientBoosting

- Subsample = 0.8
- Need to adjust all parameters

1.6504

Ridge

- alpha = 0.45

1.5486

SGD

- Max_iter = 600
- Tol = 1e-4

1.6904

Predictive words in SVC

('worst', 5.40054035348061)

('wast', 4.163555236735492)

('aw', 3.885021962099259 4)

('terribl', 3.603815130330364 7)

('garbag', 3.162175429687793)

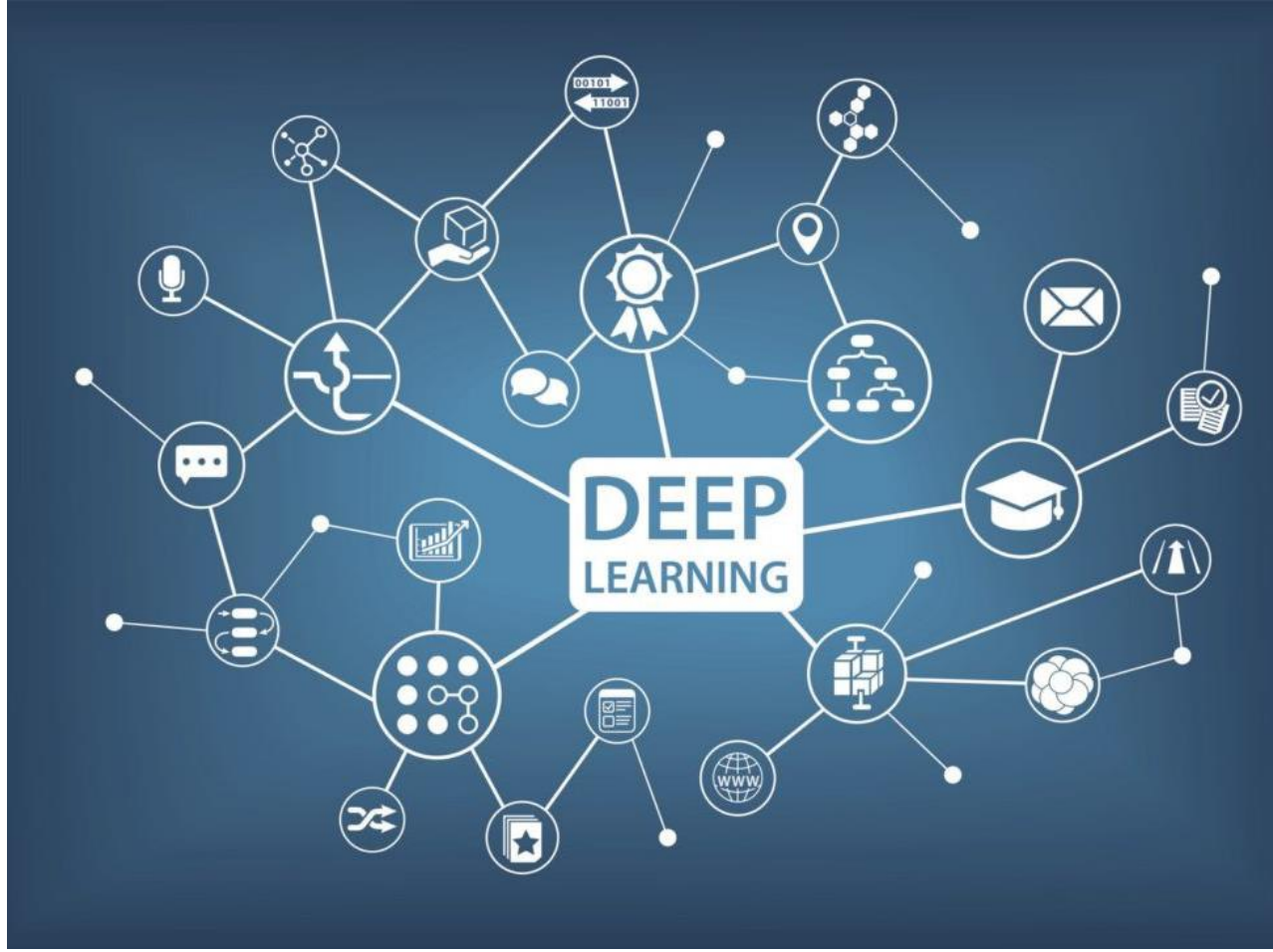
('great', -2.174787957631803)

('pretti', -1.969316873197518 1)

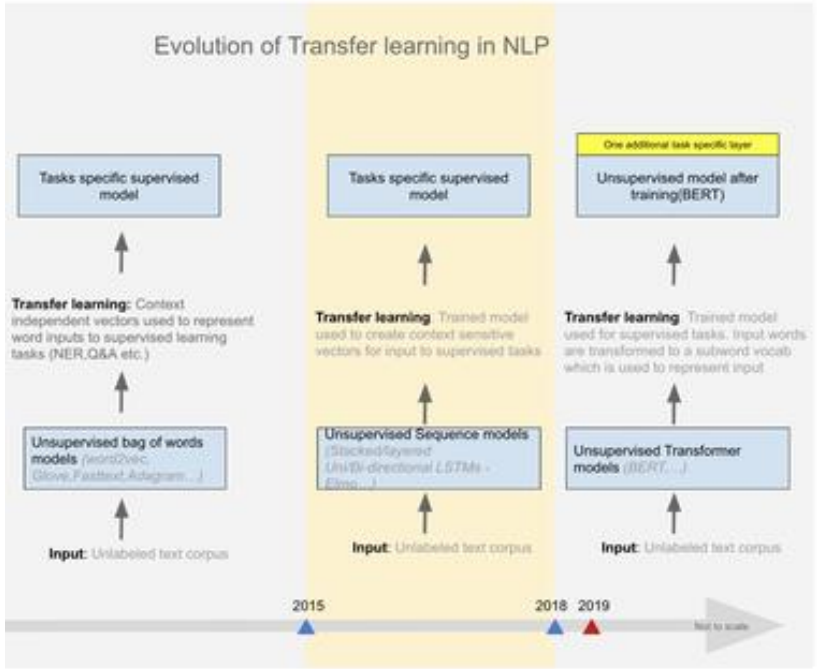
('veri', -1.872858223683766)

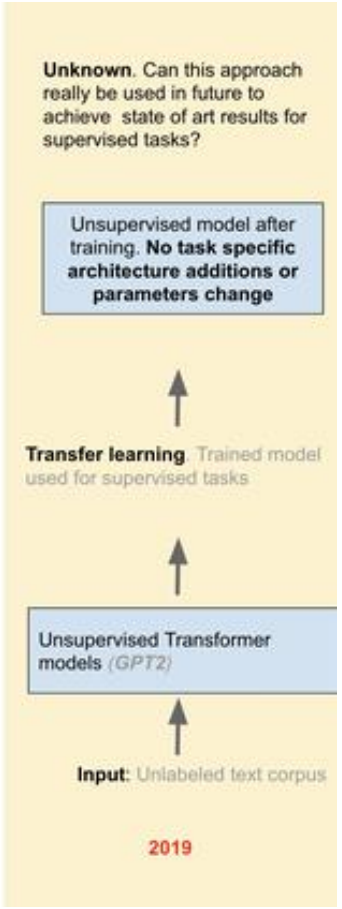
('best', -1.858891244423087 8)

('enjoy', -1.838569010618240 9)



Deep learning in NLP





Top NLP models

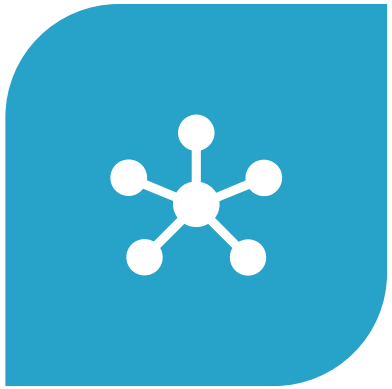
Bert, Bidirectional Encoder Representations from Transformers

GPT-2, based on generative pre-trained model (GPT)

https://medium.com/@ajitrajasekharan/gpt-2-a-promising-but-nascent-transfer-learning-method-that-could-reduce-or-even-eliminate-in-some-48ea3370cc21

Deep Learning

KTRAIN - BERT



KTRAIN – NBSVM

KERAS -LSTM

LSTM

Layer (type) Output Shape Param #
=== embedding_11
(Embedding) (None, 48, 300) 4119900
____ lstm_21 (LSTM)
(None, 48, 128) 219648
____ lstm_22 (LSTM)
(None, 64) 49408
____ dense_21
(Dense) (None, 100) 6500
____ dropout_11
(Dropout) (None, 100) 0
____ dense_22
(Dense) (None, 5) 505
=== Total params:
4,395,961 Trainable params: 4,395,961 Non-trainable params: 0

https://www.visitnorway.com/places-to-go/northern-norway/helgeland/

1

Create app.py with a selected classifier

2

Call app.py, train a model and open http://127.0.0.1:5000 in your browser

3

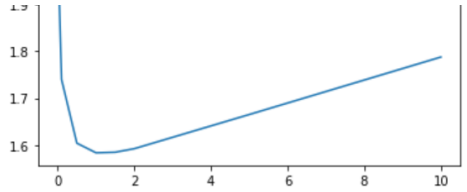
Input English text and check a predicted score

Deployed application

tprojects.pythonanywhere.com

Summary

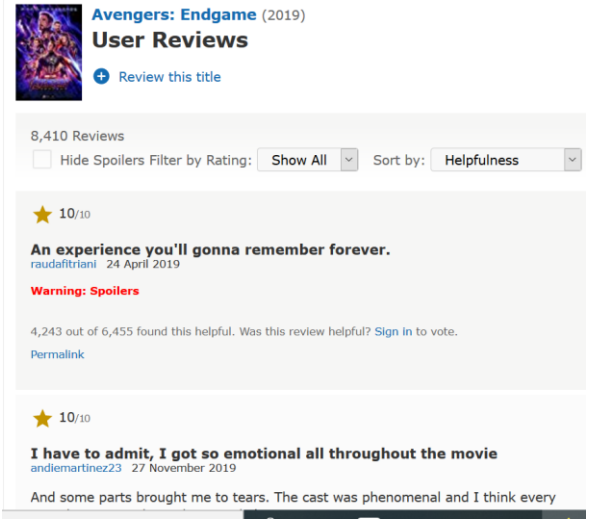
- SVM gives fast and relatively accurate classifier for sentiment analysis task
- Deployed app at **tprojects.pythonanywhere.com**
- BERT, LSTM and Gradient Boosting requite long training time and bigger memory, but these methods could give better results after adjustment
- Many libraries and labelled datasets for binary classification
- Explainable AI

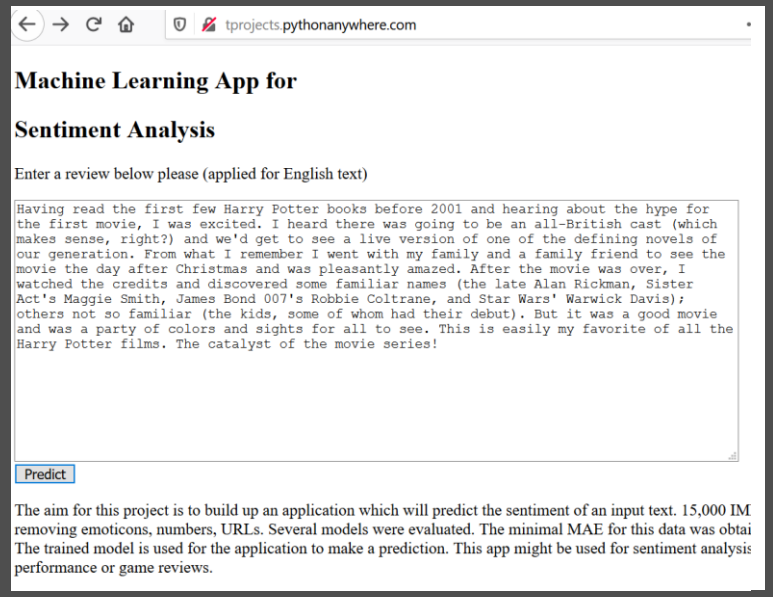


Generate pipeline and set of parameters for grid search.

In [10]:
```
def make_pipeline(vectorizer, transformer, classifier):
    return Pipeline([
            ('vectorizer', vectorizer),
            ('transformer', transformer),
            ('classifier', classifier)
        ])
```

In [11]:
```
params_grid_vectorizer = {
    'vectorizer__max_df' : [0.85, 0.95, 1],
    'vectorizer__ngram_range' : [(1, 1), (1, 2), (1, 3)
    'vectorizer__stop_words' : ['english', None]
}
```






```
C:\Users\aleff\Documents\PythonScripts\Untitled Folder\myapp>python app3.py
C:\Users\aleff\Anaconda3\lib\site-packages\sklearn\externals\joblib\__init__.py:15: Deprec
externals.joblib is deprecated in 0.21 and will be removed in 0.23. Please import this fun
om joblib, which can be installed with: pip install joblib. If this warning is raised when
s, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=DeprecationWarning)
 * Serving Flask app "app3" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
C:\Users\aleff\Anaconda3\lib\site-packages\sklearn\externals\joblib\__init__.py:15: Deprec
externals.joblib is deprecated in 0.21 and will be removed in 0.23. Please import this fun
om joblib, which can be installed with: pip install joblib. If this warning is raised when
s, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=DeprecationWarning)
 * Debugger is active!
 * Debugger PIN: 299-287-796
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```