Lecture 1 - Introduction

Neural Information Processing Systems (NIPS) is one of the top machine learning conferences in the world. It covers a broad range of topics, from deep learning and computer vision to cognitive science and reinforcement learning.

You may found an archive with all papers from NIPS 2015 at https://www.kaggle.com/benhamner/nips-2015-papers (https://www.kaggle.com/benhamner/nips-2015-papers). Apart from the papers in pdf format, the archive contains three CSV files and one SQLite database (with three tables being copies of the CVS files)

At https://github.com/benhamner/nips-2015-papers (https://github.com/benhamner/nips-2015-papers) you will find the Python code used to generate this database.

Papers.csv

Each row in this file corresponds to one of 403 papers published at the conference. It includes the following fields:

- **Id** - unique identifier for the paper
- **Title** - title of the paper
- **EventType** - type of the contribution (poster, oral, spotlight)
- **PdfName** - filename for the PDF document
- **Abstract** - text for the abstract ("scraped" from NIPS website)
- **PaperText** - raw text from the PDF document (created with the `pdftotext` tool)

Authors.csv

- **Id** - unique identifier for the author
- **Name** - author's name

PaperAuthors.csv

This file links papers to their corresponding authors:

- **Id** - unique identifier
- **PaperId** - id for the paper
- **AuthorId** - id for the author

Task 1 - find similar papers basing on their abstracts and full texts

Steps:

1. Find keywords in each paper by making use of **tf-idf**.
2. Use the **knn** model to detect similar papers.

Preparation of data:

1. Remove control codes like `\n` or `\x` from text (replace them by white spaces).
2. Convert everything to unicode (not necessary in Python 3).
3. Change text to lower case.

Digression 1: tf-idf (*tf – term frequency*, *idf – inverse document frequency*)

- a numerical statistics that is intended to reflect how important a word is to a document in a collection or corpus
- often used as a weighting factor in information retrieval, text mining, and user modeling
- the tf-idf value increases proportionally to the number of times a word appears in the document, but is often offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general
- one of the most popular term-weighting schemes
  - about 80% of text-based recommender systems in the domain of digital libraries use tf-idf
- variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query
- tf–idf can be successfully used for stop-words filtering in various subject fields including text summarization and classification
- one of the simplest ranking functions is computed by summing the tf–idf for each query term

The value of **tf-idf** is a product of two statistics, term frequency and inverse document frequency:

$$(tf-idf)_{i,j} = tf_{i,j} \times idf_i$$

The **term frequency** $tf_{i,j}$ may be determined in many ways. One of the choices is to use the raw count of a term in a document normalized by the sum of frequencies of all terms in the document:

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

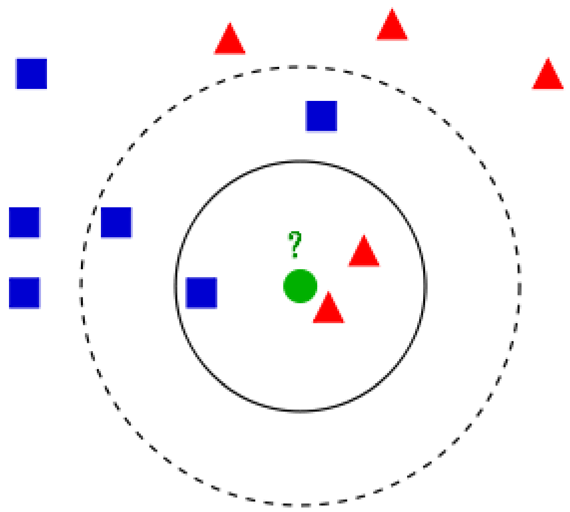
Here, $n_{i,j}$ is the number of occurences of term t_i in the document d_j.

The **inverse document frequency** is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient:

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

- $|D|$ - total number of documents
- $|\{d : t_i \in d\}|$ number of documents including the term t_i at least once.

Digression 2: knn model (*k nearest neighbours*) ¶



- a non-parametric method used for classification and regression in pattern recognition
- the input consists of the k closest training examples in the feature space
- the output depends on whether k-NN is used for classification or regression:
  - in k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor in k-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbors
- a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification
- one of the simplest machine learning algorithms.
- neighbors are taken from a set of objects for which the class (for k-NN classification) or the object property value (for k-NN regression) is known (the training set for the algorithm, though no explicit training step is required)
- sensitive to the local structure of the data

Settings:

- we are given a labelled dataset consiting of training observations (x, y). Here, x denotes a feature (attribute) and y is the target (label, class)
- given an unseen observation C with a set of attributes x we would like to predict the corresponding output y
- more formally, our goal is to learn a function $h : X \rightarrow Y$

Algorithm:

- compare the attributes of C with the attributes in the labelled dataset
- choose k most similar instances from the labelled dataset. Similarity is defined according to a distance metric between two data points (a popular choice is the Euclidian distance). k is a positive integer
- assing C to the most common class among the k neighbors

Importing of the required modules

In [1]:

```
import pandas as pd #data analysis
import sklearn      #machine learning
import numpy as np  #
import nltk         #natural language processing
import re
import codecs
import time
```

Reading the data

In [2]:

```
papers_data = pd.read_csv('data/Papers.csv')
authors_data = pd.read_csv('data/Authors.csv')
authorID_data = pd.read_csv('data/PaperAuthors.csv')
```

In [3]:

```
papers_data
```

Out[3]:

	Id	Title	EventType	PdfName	Abstract	PaperText
0	5677	Double or Nothing: Multiplicative Incentive Me...	Poster	5677-double-or-nothing-multiplicative-incentiv...	Crowdsourcing has gained immense popularity in...	Double or No Multiplicative\ M...
1	5941	Learning with Symmetric Label Noise: The Impor...	Spotlight	5941-learning-with-symmetric-label-noise-the-i...	Convex potential minimisation is the de facto ...	Learning with Symmetric La The\nImpo...
2	6019	Algorithmic Stability and Uniform Generalization	Poster	6019-algorithmic-stability-and-uniform-general...	One of the central questions in statistical le...	Algorithmic S and Uniform Generalizati...
3	6035	Adaptive Low-Complexity Sequential Inference f...	Poster	6035-adaptive-low-complexity-sequential-infere...	We develop a sequential low-complexity inferen...	Adaptive Low Complexity S Inference f...
4	5978	Covariance-Controlled Adaptive Langevin Thermo...	Poster	5978-covariance-controlled-adaptive-langevin-t...	Monte Carlo sampling for Bayesian posterior in...	Covariance-C Adaptive Langevin\nTh
5	5714	Robust Portfolio Optimization	Poster	5714-robust-portfolio-optimization.pdf	We propose a robust portfolio optimization app...	Robust Portfo Optimization\ Han\nDep...
6	5937	Logarithmic Time Online Multiclass prediction	Spotlight	5937-logarithmic-time-online-multiclass-predic...	We study the problem of multiclass classificat...	Logarithmic T Online Multic prediction\...
7	5802	Planar Ultrametrics for Image Segmentation	Poster	5802-planar-ultrametrics-for-image-segmentatio...	We study the problem of hierarchical clusterin...	Planar Ultram Image Segmentatio
8	5776	Expressing an Image Stream with a Sequence of ...	Poster	5776-expressing-an-image-stream-with-a-sequenc...	We propose an approach for generating a sequen...	Expressing a Stream with a Sequence of\
9	5814	Parallel Correlation Clustering on Big Graphs	Poster	5814-parallel-correlation-clustering-on-big-gr...	Given a similarity graph between items, correl...	Parallel Corre Clustering on Graphs\...

	Id	Title	EventType	PdfName	Abstract	PaperText
10	5638	Faster R-CNN: Towards Real-Time Object Detecti...	Poster	5638-faster-r-cnn-towards-real-time-object-det...	State-of-the-art object detection networks dep...	Faster R-CNI Towards Rea Object Detec
11	5971	Space-Time Local Embeddings	Poster	5971-space-time-local-embeddings.pdf	Space-time is a profound concept in physics. T...	Space-Time I Embeddings\ Sun1∗ Jun V
12	5830	A Convergent Gradient Descent Algorithm for Ra...	Poster	5830-a-convergent-gradient-descent-algorithm-f...	We propose a simple, scalable, and fast gradie...	A Convergen Descent Algo for\nR...
13	6002	Smooth Interactive Submodular Set Cover	Poster	6002-smooth-interactive-submodular-set-cover.pdf	Interactive submodular set cover is an interac...	Smooth Inter Submodular S Cover\nYison
14	5780	Galileo: Perceiving Physical Object Properties...	Poster	5780-galileo-perceiving-physical-object-proper...	Humans demonstrate remarkable abilities to pre...	Galileo: Perc Physical Obje Properties...
15	5766	On the Pseudo-Dimension of Nearly Optimal Auct...	Spotlight	5766-on-the-pseudo-dimension-of-nearly-optimal...	This paper develops a general approach, rooted...	The Pseudo-I of Near-Optin Auctions\...
16	5790	Unlocking neural population non-stationarities...	Poster	5790-unlocking-neural-population-non-stationar...	Neural population activity often exhibits rich...	Unlocking ne population nc stationarity\n.
17	5973	Bayesian Manifold Learning: The Locally Linear...	Poster	5973-bayesian-manifold-learning-the-locally-li...	We introduce the Locally Linear Latent Variabl...	Bayesian Ma Learning:\nTI Linea...
18	5864	Color Constancy by Learning to Predict Chromat...	Spotlight	5864-color-constancy-by-learning-to-predict-ch...	Color constancy is the recovery of true surfac...	Color Consta Learning to Predict\nChrc
19	5681	Fast and Accurate Inference of Plackett–Luce M...	Poster	5681-fast-and-accurate-inference-of-plackettlu...	We show that the maximum-likelihood (ML) estim...	Fast and Acc Inference of P Luce M...

	Id	Title	EventType	PdfName	Abstract	PaperText
20	5753	Probabilistic Line Searches for Stochastic Opt...	Oral	5753-probabilistic-line-searches-for-stochasti...	In deterministic optimization, line searches a...	Probabilistic I Searches\nfo Stochastic O
21	5857	Inferring Algorithmic Patterns with Stack-Augm...	Spotlight	5857-inferring-algorithmic-patterns-with-stack...	Despite the recent achievements in machine lea...	Inferring Algo Patterns with Aug...
22	5848	Where are they looking?	Spotlight	5848-where-are-they-looking.pdf	Humans have the remarkable ability to follow t...	Where are th looking?\n\nA Recasens∗\r
23	6032	The Pareto Regret Frontier for Bandits	Poster	6032-the-pareto-regret-frontier-for-bandits.pdf	Given a multi-armed bandit problem it may be d...	The Pareto R Frontier for Bandits\nTor
24	5719	On the Limitation of Spectral Methods: From th...	Poster	5719-on-the-limitation-of-spectral-methods-fro...	We consider the following detection problem: g...	On the Limita Spectral Methods:\nFr
25	5768	Measuring Sample Quality with Stein's Method	Spotlight	5768-measuring-sample-quality-with-steins-meth...	To improve the efficiency of Monte Carlo estim...	Measuring Sa Quality with S Method\n...
26	5778	Bidirectional Recurrent Convolutional Networks...	Poster	5778-bidirectional-recurrent-convolutional-net...	Super resolving a low-resolution video is usua...	Bidirectional I Convolutiona Networks...
27	5912	Bounding errors of Expectation-Propagation	Poster	5912-bounding-errors-of-expectation-propagatio...	Expectation Propagation is a very popular algo...	Bounding erro Expectation-Propagation\ı
28	5972	A fast, universal algorithm to learn parametri...	Poster	5972-a-fast-universal-algorithm-to-learn-param...	Nonlinear embedding algorithms such as stochas...	A Fast, Unive Algorithm\nto Parametr...
29	5633	Texture Synthesis Using Convolutional Neural N...	Poster	5633-texture-synthesis-using-convolutional-neu...	Here we introduce a new model of natural textu...	Texture Synth Using Convol Neural\n...

...	...	...	...	...	...	...

	Id	Title	EventType	PdfName	Abstract	PaperText
373	5844	Adaptive Online Learning	Spotlight	5844-adaptive-online-learning.pdf	We propose a general framework for studying ad...	Adaptive Onli Learning\nDy Foster ∗\nC.
374	5928	A Universal Catalyst for First-Order Optimization	Poster	5928-a-universal-catalyst-for-first-order-opti...	We introduce a generic scheme for accelerating...	A Universal C First-Order O
375	5810	Inference for determinantal point processes wi...	Poster	5810-inference-for-determinantal-point-process...	Determinantal point processes (DPPs) are point...	Inference for determinanta processes\nw
376	5895	Kullback-Leibler Proximal Variational Inference	Poster	5895-kullback-leibler-proximal-variational-inf...	We propose a new variational inference method ...	Kullback-Leib Proximal Vari Inferenc...
377	5825	Semi-Proximal Mirror-Prox for Nonsmooth Compos...	Poster	5825-semi-proximal-mirror-prox-for-nonsmooth-c...	We propose a new first-order optimization algo...	Semi-Proxima Prox\nfor Nor Compo...
378	5739	LASSO with Non-linear Measurements is Equivale...	Spotlight	5739-lasso-with-non-linear-measurements-is-equ...	Consider estimating an unknown, but structured...	LASSO with I Measuremen Equivale...
379	6009	From random walks to distances on unweighted g...	Poster	6009-from-random-walks-to-distances-on-unweigh...	Large unweighted directed graphs are commonly ...	From random distances on\nunweight
380	5965	Bayesian dark knowledge	Poster	5965-bayesian-dark-knowledge.pdf	We consider the problem of Bayesian parameter ...	Bayesian Dar Knowledge\n\ Korattikara, ..
381	5940	Matrix Completion with Noisy Side Information	Spotlight	5940-matrix-completion-with-noisy-side-informa...	We study matrix completion problem with side i...	Matrix Compl Noisy Side Information\..
382	5660	Dependent Multinomial Models Made Easy: Stick-...	Poster	5660-dependent-multinomial-models-made-easy-st...	Many practical modeling problems involve discr...	Dependent M Models Made Easy:\nStick.

	Id	Title	EventType	PdfName	Abstract	PaperText
383	5860	On-the-Job Learning with Bayesian Decision Theory	Spotlight	5860-on-the-job-learning-with-bayesian-decisio...	Our goal is to deploy a high-accuracy system s...	On-the-Job L with Bayesiar The...
384	5658	Calibrated Structured Prediction	Poster	5658-calibrated-structured-prediction.pdf	In user-facing applications, displaying calibr...	Calibrated St Prediction\nP Liang\...
385	5775	Learning Structured Output Representation usin...	Poster	5775-learning-structured-output-representation...	Supervised deep learning has been successfully...	Learning Stru Output Representati
386	5979	Time-Sensitive Recommendation From Recurrent U...	Poster	5979-time-sensitive-recommendation-from-recurr...	By making personalized suggestions, a recommen...	Time-Sensitiv Recommenda From\nRecur
387	5772	Learning Stationary Time Series using Gaussian...	Spotlight	5772-learning-stationary-time-series-using-gau...	We introduce the Gaussian Process Convolution ...	Learning Stat Time Series u Gaussian...
388	5995	A Market Framework for Eliciting Private Data	Poster	5995-a-market-framework-for-eliciting-private-...	We propose a mechanism for purchasing informat...	A Market Fra for Eliciting P Data\...
389	5900	Lifted Inference Rules With Constraints	Poster	5900-lifted-inference-rules-with-constraints.pdf	Lifted inference rules exploit symmetries for ...	Lifted Inferen with Constraints\n
390	5899	Gradient Estimation Using Stochastic Computati...	Poster	5899-gradient-estimation-using-stochastic-comp...	In a variety of problems originating in superv...	Gradient Esti Using\nStoch Computat...
391	5672	Model-Based Relative Entropy Stochastic Search	Poster	5672-model-based-relative-entropy-stochastic-s...	Stochastic search algorithms are general black...	Model-Based Entropy Stocl Search...
392	5947	Semi-supervised Learning with Ladder Networks	Poster	5947-semi-supervised-learning-with-ladder-netw...	We combine supervised learning with unsupervis...	Semi-Superv Learning with Networks\...

	Id	Title	EventType	PdfName	Abstract	PaperText
393	5675	Embedding Inference for Structured Multilabel ...	Poster	5675-embedding-inference-for-structured-multil...	A key bottleneck in structured output predicti...	Embedding Inference\nfo Structured Mu
394	5669	Copula variational inference	Poster	5669-copula-variational-inference.pdf	We develop a general variational inference met...	Copula variat inference\n\n Tran\nH...
395	5636	Recursive Training of 2D-3D Convolutional Netw...	Poster	5636-recursive-training-of-2d-3d-convolutional...	Efforts to automate the reconstruction of neur...	Recursive Tra 2D-3D Convo Netw...
396	5924	A Dual Augmented Block Minimization Framework ...	Poster	5924-a-dual-augmented-block-minimization-frame...	In past few years, several techniques have bee...	A Dual-Augm Block Minimiz Framework\...
397	5839	Optimal Testing for Properties of Distributions	Spotlight	5839-optimal-testing-for-properties-of-distrib...	Given samples from an unknown distribution, p...	Optimal Testi Properties of Distribution...
398	5792	Efficient Learning of Continuous-Time Hidden M...	Poster	5792-efficient-learning-of-continuous-time-hid...	The Continuous-Time Hidden Markov Model (CT-HM...	Efficient Lear Continuous-T Hidden\n...
399	5674	Expectation Particle Belief Propagation	Poster	5674-expectation-particle-belief-propagation.pdf	We propose an original particle-based implemen...	Expectation P Belief Propagation\n
400	5756	Latent Bayesian melding for integrating indivi...	Spotlight	5756-latent-bayesian-melding-for-integrating-i...	In many statistical problems, a more coarse-gr...	Latent Bayes melding for in indivi...
401	5745	Distributionally Robust Logistic Regression	Spotlight	5745-distributionally-robust-logistic-regressi...	This paper proposes a distributionally robust ...	Confidence In and Hypothes fo...
402	5666	Variational Dropout and the Local	Poster	5666-variational-dropout-and-the-local-reparam	We explore an as yet unexploited	Learning Opt Commitment Overcome In

		Reparameteri...		local reparam...	opportunity f...	Overcome ina

403 rows × 6 columns

In [5]:

```
print(authors_data)
```

	Id	Name
0	4113	Constantine Caramanis
1	4828	Richard L. Lewis
2	5506	Ryan Kiros
3	7331	Kfir Levy
4	8429	Wei Cao
5	7525	Hsiao-Yu Tung
6	7997	Kai Wei
7	4137	Mark Schmidt
8	8142	Dimitris Papailiopoulos
9	6471	Ce Zhang
10	1431	Ron Meir
11	7485	Jimmy SJ Ren
12	7982	Marin Kobilarov
13	6320	Nikhil Rao
14	8043	Rahul Kidambi
15	8249	Tianqi Chen
16	4548	Mohammad E. Khan
17	6650	Sebastien Bubeck
18	5192	Rémi Bardenet
19	7963	Dominik Rothenhäusler
20	7950	Luis Pualo Reis
21	7398	Ross Girshick
22	8247	Samuel Livingstone
23	3387	Huan Xu
24	5348	Quanquan Gu
25	5542	Pinghua Gong
26	8195	Dharmendra S. Modha
27	8160	Ian Kash
28	6798	Bernt Schiele
29	7564	koray kavukcuoglu
...	...	...
1043	3001	Lawrence Carin
1044	8250	Rahul G. Krishnan
1045	4585	Silvia Villa
1046	7502	Somdeb Sarkhel
1047	8143	Siddhartha Banerjee
1048	2966	Kilian Q. Weinberger
1049	2686	Pieter Abbeel
1050	4285	Petros Drineas
1051	8029	Yossi Arjevani
1052	8156	Christian Borgs
1053	5588	James Hensman
1054	8332	Shixiang Gu
1055	8061	Manuel Rodriguez
1056	7979	Shaona Ghosh
1057	8138	Maxim Rabinovich
1058	7633	Ariel D. Procaccia
1059	6644	Aryeh Kontorovich
1060	4022	Shakir Mohamed
1061	8023	Xiaoming Yuan
1062	7921	Todd Phillips
1063	5586	Youssef Mroueh
1064	8149	Chao Qian
1065	8092	C. Lawrence Zitnick
1066	8307	Antti Rasmus
1067	8087	Qiong Yan
1068	5558	Yuekai Sun
1069	8150	Yang Yu
1070	8065	Andrew Gelman
1071	8222	Maurizio Filippone

```
1072  4509           Raquel Urtasun

[1073 rows x 2 columns]
```

In [6]:

```
print(authorID_data)
```

	Id	PaperId	AuthorId
0	1	5677	7956
1	2	5677	2649
2	3	5941	8299
3	4	5941	8300
4	5	5941	575
5	6	6019	8419
6	7	6035	8437
7	8	6035	8437
8	9	6035	8438
9	10	5978	8366
10	11	5978	8367
11	12	5978	8368
12	13	5978	2459
13	14	5714	8012
14	15	5714	5637
15	16	5714	3555
16	17	5714	8013
17	18	5937	8068
18	19	5937	2139
19	20	5802	8130
20	21	5802	8131
21	22	5776	8090
22	23	5776	4225
23	24	5814	6551
24	25	5814	8142
25	26	5814	7243
26	27	5814	3537
27	28	5814	7986
28	29	5814	6338
29	30	5638	7875
...	...	...	...
1286	1287	5947	2813
1287	1288	5947	7190
1288	1289	5675	7954
1289	1290	5675	7658
1290	1291	5675	4524
1291	1292	5675	1904
1292	1293	5669	7944
1293	1294	5669	6452
1294	1295	5669	4085
1295	1296	5636	7872
1296	1297	5636	7873
1297	1298	5636	7874
1298	1299	5636	863
1299	1300	5924	7160
1300	1301	5924	8290
1301	1302	5924	7476
1302	1303	5839	5296
1303	1304	5839	8166
1304	1305	5839	8167
1305	1306	5792	8118
1306	1307	5792	7967
1307	1308	5792	4523
1308	1309	5792	3161
1309	1310	5792	2077
1310	1311	5674	7953
1311	1312	5674	1925
1312	1313	5674	1821
1313	1314	5756	3402
1314	1315	5756	7541

```
1315  1316     5756      7542

[1316 rows x 3 columns]
```

Helper functions

For convenience we define two helper function, which allow us to extract the id of a paper from its index and vice versa:

In [4]:

```
def given_paperID_give_index(paper_id, paper_data):
    return paper_data[paper_data['Id']==paper_id].index[0]

def given_index_give_PaperID(index, paper_data):
    return paper_data.iloc[index]['Id']
```

Raw text of a paper

In [5]:

```
Ex_paper_id = 5941
Ex_paper_index = given_paperID_give_index(Ex_paper_id, papers_data)
papers_data.iloc[Ex_paper_index]['PaperText'][0:1000]
```

Out[5]:

```
'Learning with Symmetric Label Noise: The\nImportance of Being Unhin
ged\n\nBrendan van Rooyen∗,†\n∗\n\nAditya Krishna Menon†,∗\n\nThe Au
stralian National University\n\n†\n\nRobert C. Williamson∗,†\n\nNati
onal ICT Australia\n\n{ brendan.vanrooyen, aditya.menon, bob.william
son }@nicta.com.au\n\nAbstract\nConvex potential minimisation is the
de facto approach to binary classification.\nHowever, Long and Serve
dio [2010] proved that under symmetric label noise\n(SLN), minimisat
ion of any convex potential over a linear function class can result
 in classification performance equivalent to random guessing. This o
stensibly\nshows that convex losses are not SLN-robust. In this pape
r, we propose a convex,\nclassification-calibrated loss and prove th
at it is SLN-robust. The loss avoids the\nLong and Servedio [2010] r
esult by virtue of being negatively unbounded. The\nloss is a modifi
cation of the hinge loss, where one does not clamp at zero; hence,\n
we call it the unhinged loss. We show that the optimal unhinged solu
tion is'
```

Text cleaning

In [6]:

```
def clean_text(text):
    clean_text = re.sub('[^a-zA-Z]+', ' ', text)
    return clean_text.lower()
```

In [7]:

```
papers_data['PaperText_clean'] = papers_data['PaperText'].apply(lambda x: clean_
text(x))
papers_data['Abstract_clean'] = papers_data['Abstract'].apply(lambda x: clean_te
xt(x))
```

Text after cleaning

In [8]:

```
papers_data.iloc[Ex_paper_index]['PaperText_clean'][0:1000]
```

Out[8]:

```
'learning with symmetric label noise the importance of being unhinge
d brendan van rooyen aditya krishna menon the australian national un
iversity robert c williamson national ict australia brendan vanrooye
n aditya menon bob williamson nicta com au abstract convex potential
minimisation is the de facto approach to binary classification howev
er long and servedio proved that under symmetric label noise sln min
imisation of any convex potential over a linear function class can r
esult in classification performance equivalent to random guessing th
is ostensibly shows that convex losses are not sln robust in this pa
per we propose a convex classification calibrated loss and prove tha
t it is sln robust the loss avoids the long and servedio result by v
irtue of being negatively unbounded the loss is a modification of th
e hinge loss where one does not clamp at zero hence we call it the u
nhinged loss we show that the optimal unhinged solution is equivalen
t to that of a strongly regularised svm and is t'
```

Calculating tf-idf

First of all we need a function which will reduce inflected (or sometimes derived) words to their stems (a part of the word which is common to all its inflected variants):

In [9]:

```
from nltk.stem.snowball import SnowballStemmer
stemmer = SnowballStemmer("english")
def tokenize_and_stem(text):
    # first tokenize by sentence, then by word to ensure that punctuation is cau
ght as it's own token
    tokens = [word for sent in nltk.sent_tokenize(text) for word in nltk.word_to
kenize(sent)]
    filtered_tokens = []
    # filter out any tokens not containing letters (e.g., numeric tokens, raw pu
nctuation)
    for token in tokens:
        if re.search('[a-zA-Z]', token):
            filtered_tokens.append(token)
    stems = [stemmer.stem(t) for t in filtered_tokens]
    return stems
```

Using this function, we can calculate the **tf-idf** matrix, both for the abstracts and full texts:

In [14]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
# tf_idf matrix for abstracts
tfidf_vectorizer_Abstract = TfidfVectorizer(max_df=0.95, max_features=200000,
                                 min_df=0.05, stop_words='english',
                                 use_idf=True, tokenizer=tokenize_and_stem, ngra
m_range=(1,3))
%time tfidf_matrix_Abstract = tfidf_vectorizer_Abstract.fit_transform(papers_dat
a['Abstract_clean'])

# tf_idf matrix for full papers
tfidf_vectorizer_PaperText = TfidfVectorizer(max_df=0.9, max_features=200000,
                                 min_df=0.1, stop_words='english',
                                 use_idf=True, tokenizer=tokenize_and_stem, ngra
m_range=(1,3))
%time tfidf_matrix_PaperText = tfidf_vectorizer_PaperText.fit_transform(papers_d
ata['PaperText_clean'])
```

```
---------------------------------------------------------------------
-------
LookupError                               Traceback (most recent cal
l last)
<timed exec> in <module>()

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/te
xt.py in fit_transform(self, raw_documents, y)
   1350             Tf-idf-weighted document-term matrix.
   1351         """
-> 1352         X = super(TfidfVectorizer, self).fit_transform(raw_d
ocuments)
   1353         self._tfidf.fit(X)
   1354         # X is already a transformed view of raw_documents s
o

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/te
xt.py in fit_transform(self, raw_documents, y)
    837 
    838         vocabulary, X = self._count_vocab(raw_documents,
--> 839                                           self.fixed_vocabul
ary_)
    840 
    841         if self.binary:

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/te
xt.py in _count_vocab(self, raw_documents, fixed_vocab)
    760         for doc in raw_documents:
    761             feature_counter = {}
--> 762             for feature in analyze(doc):
    763                 try:
    764                     feature_idx = vocabulary[feature]

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/te
xt.py in <lambda>(doc)
    239 
    240             return lambda doc: self._word_ngrams(
--> 241                 tokenize(preprocess(self.decode(doc))), stop
_words)
    242 
    243         else:

<ipython-input-13-e9bb4a55d55f> in tokenize_and_stem(text)
      3 def tokenize_and_stem(text):
      4     # first tokenize by sentence, then by word to ensure tha
t punctuation is caught as it's own token
----> 5     tokens = [word for sent in nltk.sent_tokenize(text) for 
word in nltk.word_tokenize(sent)]
      6     filtered_tokens = []
      7     # filter out any tokens not containing letters (e.g., nu
meric tokens, raw punctuation)

/usr/local/lib/python3.5/dist-packages/nltk/tokenize/__init__.py in 
sent_tokenize(text, language)
     92     :param language: the model name in the Punkt corpus
     93     """
---> 94     tokenizer = load('tokenizers/punkt/{0}.pickle'.format(la
nguage))
     95     return tokenizer.tokenize(text)
     96 
```

```
/usr/local/lib/python3.5/dist-packages/nltk/data.py in load(resource
_url, format, cache, verbose, logic_parser, fstruct_reader, encodin
g)
    832 
    833     # Load the resource.
--> 834     opened_resource = _open(resource_url)
    835 
    836     if format == 'raw':

/usr/local/lib/python3.5/dist-packages/nltk/data.py in _open(resourc
e_url)
    950 
    951     if protocol is None or protocol.lower() == 'nltk':
--> 952         return find(path_, path + ['']).open()
    953     elif protocol.lower() == 'file':
    954         # urllib might not use mode='rb', so handle this one
 ourselves:

/usr/local/lib/python3.5/dist-packages/nltk/data.py in find(resource
_name, paths)
    671     sep = '*' * 70
    672     resource_not_found = '\n%s\n%s\n%s\n' % (sep, msg, sep)
--> 673     raise LookupError(resource_not_found)
    674 
    675 

LookupError: 
*********************************************************************
**
  Resource punkt not found.
  Please use the NLTK Downloader to obtain the resource:

  >>> import nltk
  >>> nltk.download('punkt')

  Searched in:
    - '/home/szwabin/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/local/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/local/lib/nltk_data'
    - '/usr/nltk_data'
    - '/usr/lib/nltk_data'
    - ''
*********************************************************************
**
```

```
---------------------------------------------------------------------------
LookupError                               Traceback (most recent call last)
<timed exec> in <module>()

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/text.py in fit_transform(self, raw_documents, y)
   1350             Tf-idf-weighted document-term matrix.
   1351         """
-> 1352         X = super(TfidfVectorizer, self).fit_transform(raw_documents)
   1353         self._tfidf.fit(X)
   1354         # X is already a transformed view of raw_documents so

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/text.py in fit_transform(self, raw_documents, y)
    837 
    838         vocabulary, X = self._count_vocab(raw_documents,
--> 839                                           self.fixed_vocabulary_)
    840 
    841         if self.binary:

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/text.py in _count_vocab(self, raw_documents, fixed_vocab)
    760         for doc in raw_documents:
    761             feature_counter = {}
--> 762             for feature in analyze(doc):
    763                 try:
    764                     feature_idx = vocabulary[feature]

/usr/local/lib/python3.5/dist-packages/sklearn/feature_extraction/text.py in <lambda>(doc)
    239 
    240             return lambda doc: self._word_ngrams(
--> 241                 tokenize(preprocess(self.decode(doc))), stop_words)
    242 
    243         else:

<ipython-input-13-e9bb4a55d55f> in tokenize_and_stem(text)
      3 def tokenize_and_stem(text):
      4     # first tokenize by sentence, then by word to ensure that punctuation is caught as it's own token
----> 5     tokens = [word for sent in nltk.sent_tokenize(text) for word in nltk.word_tokenize(sent)]
      6     filtered_tokens = []
      7     # filter out any tokens not containing letters (e.g., numeric tokens, raw punctuation)

/usr/local/lib/python3.5/dist-packages/nltk/tokenize/__init__.py in sent_tokenize(text, language)
     92     :param language: the model name in the Punkt corpus
     93     """
---> 94     tokenizer = load('tokenizers/punkt/{0}.pickle'.format(language))
     95     return tokenizer.tokenize(text)
     96 
```

```
/usr/local/lib/python3.5/dist-packages/nltk/data.py in load(resource
_url, format, cache, verbose, logic_parser, fstruct_reader, encodin
g)
    832 
    833     # Load the resource.
--> 834     opened_resource = _open(resource_url)
    835 
    836     if format == 'raw':

/usr/local/lib/python3.5/dist-packages/nltk/data.py in _open(resourc
e_url)
    950 
    951     if protocol is None or protocol.lower() == 'nltk':
--> 952         return find(path_, path + ['']).open()
    953     elif protocol.lower() == 'file':
    954         # urllib might not use mode='rb', so handle this one
 ourselves:

/usr/local/lib/python3.5/dist-packages/nltk/data.py in find(resource
_name, paths)
    671     sep = '*' * 70
    672     resource_not_found = '\n%s\n%s\n%s\n' % (sep, msg, sep)
--> 673     raise LookupError(resource_not_found)
    674 
    675 

LookupError: 
*********************************************************************
**
  Resource punkt not found.
  Please use the NLTK Downloader to obtain the resource:

  >>> import nltk
  >>> nltk.download('punkt')

  Searched in:
    - '/home/szwabin/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/local/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/local/lib/nltk_data'
    - '/usr/nltk_data'
    - '/usr/lib/nltk_data'
    - ''
*********************************************************************
**
```

In [11]:

```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /home/szwabin/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

Out[11]:

```
True
```

In [12]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
# macierz tf_idf na podstawie Abstract
tfidf_vectorizer_Abstract = TfidfVectorizer(max_df=0.95, max_features=200000,
                                 min_df=0.05, stop_words='english',
                                 use_idf=True, tokenizer=tokenize_and_stem, ngra
m_range=(1,3))
%time tfidf_matrix_Abstract = tfidf_vectorizer_Abstract.fit_transform(papers_dat
a['Abstract_clean'])

# macierz tf_idf na podstawie PaperText
tfidf_vectorizer_PaperText = TfidfVectorizer(max_df=0.9, max_features=200000,
                                 min_df=0.1, stop_words='english',
                                 use_idf=True, tokenizer=tokenize_and_stem, ngra
m_range=(1,3))
%time tfidf_matrix_PaperText = tfidf_vectorizer_PaperText.fit_transform(papers_d
ata['PaperText_clean'])
```

```
CPU times: user 1.52 s, sys: 0 ns, total: 1.52 s
Wall time: 1.53 s
CPU times: user 43.5 s, sys: 212 ms, total: 43.7 s
Wall time: 43.7 s
```

In [13]:

```
terms_Abstract = tfidf_vectorizer_Abstract.get_feature_names()
terms_PaperText = tfidf_vectorizer_Abstract.get_feature_names()
```

Helper functions for presenting the results

In [18]:

```
def top_tfidf_feats(row, terms, top_n=25):
    topn_ids = np.argsort(row)[::-1][:top_n]
    top_feats = [(terms[i], row[i]) for i in topn_ids]
    df = pd.DataFrame(top_feats)
    df.columns = ['feature', 'tfidf']
    return df['feature']

def given_paperID_give_keywords(paper_data, tfidfMatrix, terms, paper_id,
top_n=20):
    row_id = given_paperID_give_index(paper_id, paper_data)
    row = np.squeeze(tfidfMatrix[row_id].toarray())
    return top_tfidf_feats(row, terms, top_n)
```

Top 10 keywords of a given paper

In [19]:

```
paper_id_example = 5941
print("Keywords (their stems) based on abstracts:")
print(given_paperID_give_keywords(papers_data, tfidf_matrix_Abstract,
                                  terms_Abstract, paper_id_example, top_n = 10))
```

```
Keywords (their stems) based on abstracts:
0            loss
1          convex
2          robust
3         classif
4          strong
5           solut
6           prove
7             ani
8    paper propos
9          result
Name: feature, dtype: object
```

knn model

In [20]:

```
from sklearn.neighbors import NearestNeighbors
# based on abstracts
num_neighbors = 4
nbrs_Abstract = NearestNeighbors(n_neighbors=num_neighbors,
                                 algorithm='auto').fit(tfidf_matrix_Abstract)
distances_Abstract, indices_Abstract = nbrs_Abstract.kneighbors(tfidf_matrix_Abs
tract)
# based on full papers
nbrs_PaperText = NearestNeighbors(n_neighbors=num_neighbors,
                                  algorithm='auto').fit(tfidf_matrix_PaperText)
distances_PaperText, indices_PaperText = nbrs_PaperText.kneighbors(tfidf_matrix_
PaperText)
```

In [22]:

```
#results for an example paper
print ("Neighbors based on abstracts: %r" % indices_Abstract[1])
print ("Neighbors based on texts: %r" % indices_PaperText[1])
```

```
Neighbors based on abstracts: array([  1,  87, 301, 112])
Neighbors based on texts: array([  1, 125, 112, 148])
```

Abstracts of similar articles

In [23]:

```
Ex_paper_id = 5941
Ex_index = given_paperID_give_index(Ex_paper_id, papers_data)
print ("Abstract of an example paper:\n")
print (papers_data.iloc[indices_Abstract[Ex_index][0]]['Abstract'])
print ("Abstracts of similar papers:\n")
for i in range(1, len(indices_Abstract[Ex_index])):
    print ("Abstract of neighbor #%r: \n" % i)
    print (papers_data.iloc[indices_Abstract[Ex_index][i]]['Abstract'])
    print ("\n")
```

```
Abstract of an example paper:

Convex potential minimisation is the de facto approach to binary cla
ssification. However, Long and Servedio [2008] proved that under sym
metric label noise (SLN), minimisation of any convex potential over
 a linear function class can result in classification performance eq
uivalent to random guessing. This ostensibly shows that convex losse
s are not SLN-robust. In this paper, we propose a convex, classifica
tion-calibrated loss and prove that it is SLN-robust. The loss avoid
s the Long and Servedio [2008] result by virtue of being negatively
 unbounded. The loss is a modification of the hinge loss, where one
 does not clamp at zero; hence, we call it the unhinged loss. We sho
w that the optimal unhinged solution is equivalent to that of a stro
ngly regularised SVM, and is the limiting solution for any convex po
tential; this implies that strong l2 regularisation makes most stand
ard learners SLN-robust. Experiments confirm the unhinged loss' SLN-
robustness.
Abstracts of similar papers:

Abstract of neighbor #1:

The framework of online learning with memory naturally captures lear
ning problems with temporal effects, and was previously studied for
 the experts setting. In this work we extend the notion of learning
 with memory to the general Online Convex Optimization (OCO) framewo
rk, and present two algorithms that attain low regret. The first alg
orithm applies to Lipschitz continuous loss functions, obtaining opt
imal regret bounds for both convex and strongly convex losses. The s
econd algorithm attains the optimal regret bounds and applies more b
roadly to convex losses without requiring Lipschitz continuity, yet
 is more complicated to implement. We complement the theoretic resul
ts with two applications: statistical arbitrage in finance, and mult
i-step ahead prediction in statistics.


Abstract of neighbor #2:

Multivariate loss functions are used to assess performance in many m
odern prediction tasks, including information retrieval and ranking
 applications. Convex approximations are typically optimized in thei
r place to avoid NP-hard empirical risk minimization problems. We pr
opose to approximate the training data instead of the loss function
 by posing multivariate prediction as an adversarial game between a
 loss-minimizing prediction player and a loss-maximizing evaluation
 player constrained to match specified properties of training data.
 This avoids the non-convexity of empirical risk minimization, but g
ame sizes are exponential in the number of predicted variables. We o
vercome this intractability using the double oracle constraint gener
ation method. We demonstrate the efficiency and predictive performan
ce of our approach on tasks evaluated using the precision at k, the
 F-score and the discounted cumulative gain.


Abstract of neighbor #3:

Modern prediction problems arising in multilabel learning and learni
ng to rank pose unique challenges to the classical theory of supervi
sed learning. These problems have large prediction and label spaces
 of a combinatorial nature and involve sophisticated loss functions.
We offer a general framework to derive mistake driven online algorit
hms and associated loss bounds.  The key ingredients in our framewor
```

```
k are a general loss function, a general vector space representation
of predictions, and a notion of margin with respect to a general nor
m. Our general algorithm, Predtron, yields the perceptron algorithm
 and its variants when instantiated on classic problems such as bina
ry classification, multiclass classification, ordinal regression, an
d multilabel classification.  For multilabel ranking and subset rank
ing, we derive novel algorithms, notions of margins, and loss bound
s. A simulation study confirms the behavior predicted by our bounds
 and demonstrates the flexibility of the design choices in our frame
work.
```

In [24]:

```
Ex_paper_id = 5941
Ex_index = given_paperID_give_index(Ex_paper_id, papers_data)
print ("Abstract of an example paper:\n")
print (papers_data.iloc[indices_PaperText[Ex_index][0]]['Abstract'])
print ("Abstracts of similar papers:\n")
for i in range(1, len(indices_PaperText[Ex_index])):
    print ("Abstract of neighbor #%r: \n" % i)
    print (papers_data.iloc[indices_PaperText[Ex_index][i]]['Abstract'])
    print ("\n")
```

```
Abstract of an example paper:

Convex potential minimisation is the de facto approach to binary cla
ssification. However, Long and Servedio [2008] proved that under sym
metric label noise (SLN), minimisation of any convex potential over
 a linear function class can result in classification performance eq
uivalent to random guessing. This ostensibly shows that convex losse
s are not SLN-robust. In this paper, we propose a convex, classifica
tion-calibrated loss and prove that it is SLN-robust. The loss avoid
s the Long and Servedio [2008] result by virtue of being negatively
 unbounded. The loss is a modification of the hinge loss, where one
 does not clamp at zero; hence, we call it the unhinged loss. We sho
w that the optimal unhinged solution is equivalent to that of a stro
ngly regularised SVM, and is the limiting solution for any convex po
tential; this implies that strong l2 regularisation makes most stand
ard learners SLN-robust. Experiments confirm the unhinged loss’ SLN-
robustness.
Abstracts of similar papers:

Abstract of neighbor #1:

In regression problems involving vector-valued outputs (or equivalen
tly, multiple responses), it is well known that the maximum likeliho
od estimator (MLE), which takes noise covariance structure into acco
unt, can be significantly more accurate than the ordinary least squa
res (OLS) estimator. However, existing  literature compares OLS and
 MLE in terms of their asymptotic, not finite sample, guarantees. Mo
re crucially, computing the MLE in general requires solving a non-co
nvex optimization problem and is not known to be efficiently solvabl
e. We provide finite sample upper and lower bounds on the estimation
error of OLS and MLE, in two popular models: a) Pooled model, b) See
mingly Unrelated Regression (SUR) model. We provide precise instance
s where the MLE is significantly more accurate than OLS. Furthermor
e, for both models, we show that the output of a computationally eff
icient alternating minimization procedure enjoys the same performanc
e guarantee as MLE, up to universal constants. Finally, we show that
for high-dimensional settings as well, the alternating minimization
 procedure leads to significantly more accurate solutions than the c
orresponding OLS solutions but with error bound that depends only lo
garithmically on the data dimensionality.


Abstract of neighbor #2:

Modern prediction problems arising in multilabel learning and learni
ng to rank pose unique challenges to the classical theory of supervi
sed learning. These problems have large prediction and label spaces
 of a combinatorial nature and involve sophisticated loss functions.
We offer a general framework to derive mistake driven online algorit
hms and associated loss bounds.  The key ingredients in our framewor
k are a general loss function, a general vector space representation
of predictions, and a notion of margin with respect to a general nor
m. Our general algorithm, Predtron, yields the perceptron algorithm
 and its variants when instantiated on classic problems such as bina
ry classification, multiclass classification, ordinal regression, an
d multilabel classification.  For multilabel ranking and subset rank
ing, we derive novel algorithms, notions of margins, and loss bound
s. A simulation study confirms the behavior predicted by our bounds
 and demonstrates the flexibility of the design choices in our frame
work.
```

```
Abstract of neighbor #3:

This paper proposes a framework for learning features that are robus
t to data variation, which is particularly important when only a lim
ited number of trainingsamples are available. The framework makes it
possible to tradeoff the discriminative value of learned features ag
ainst the generalization error of the learning algorithm. Robustness
is achieved by encouraging the transform that maps data to features
 to be a local isometry. This geometric property is shown to improve
(K, \epsilon)-robustness, thereby providing theoretical justificatio
n for reductions in generalization error observed in experiments. Th
e proposed optimization frameworkis used to train standard learning
 algorithms such as deep neural networks. Experimental results obtai
ned on benchmark datasets, such as labeled faces in the wild,demonst
rate the value of being able to balance discrimination and robustnes
s.
```

Some other helper functions

In [25]:

```
def given_paperID_give_authours_id(paper_id, author_data, author_id_data):
    id_author_list = author_id_data[author_id_data['PaperId']==paper_id]['Author
Id']
    return id_author_list

def given_authorID_give_name(author_id, author_data):
    author_name = author_data[author_data['Id'] == author_id]['Name']
    return author_name

def given_similar_paperIDs_give_their_titles(sim_papers_list_index, paper_data):
    titles = []
    for index in sim_papers_list_index:
        titles.append(paper_data.iloc[index]['Title']+'.')
    return titles
```

In [26]:

```
Ex_paper_id = 5941
Ex_index = given_paperID_give_index(Ex_paper_id, papers_data)
print ("Titles of similar papers (based on abstracts):\n\n")
for title in
given_similar_paperIDs_give_their_titles(indices_Abstract[Ex_index],
papers_data):
    print (title)
```

```
Titles of similar papers (based on abstracts):


Learning with Symmetric Label Noise: The Importance of Being Unhinge
d.
Online Learning for Adversaries with Memory: Price of Past Mistakes.
Adversarial Prediction Games for Multivariate Losses.
Predtron: A Family of Online Algorithms for General Prediction Probl
ems.
```

In [27]:

```
Ex_paper_id = 5941
Ex_index = given_paperID_give_index(Ex_paper_id, papers_data)
print ("Titles of similar papers (based on full texts):\n\n")
for title in
given_similar_paperIDs_give_their_titles(indices_PaperText[Ex_index], papers_dat
a):
    print (title)
```

```
Titles of similar papers (based on full texts):


Learning with Symmetric Label Noise: The Importance of Being Unhinge
d.
Alternating Minimization for Regression Problems with Vector-valued
 Outputs.
Predtron: A Family of Online Algorithms for General Prediction Probl
ems.
Discriminative Robust Transformation Learning.
```

Task 2 - most frequent keywords

In [28]:

```
from sklearn.feature_extraction.text import CountVectorizer

EXCLUDED_BIGRAMS = [
"et al",
"10 10",
"international conference",
"neural information",
"information processing",
"processing systems",
"advances neural",
"supplementary material"
]

cv = CountVectorizer(ngram_range=(2,2), max_features = 500,
stop_words='english')
cv.fit(papers_data.PaperText)

X = cv.transform(papers_data.PaperText)
counts = X.sum(axis=0)

df = pd.DataFrame({'Bigrams': cv.get_feature_names(), 'Count': counts.tolist()
[0]})
df = df[df.Bigrams.map(lambda x: x not in EXCLUDED_BIGRAMS)]
df.sort_values(by='Count', ascending=False, inplace=True)

print(df.head(50))
```

	Bigrams	Count
257	machine learning	1436
300	neural networks	770
254	lower bound	566
253	low rank	521
184	high dimensional	485
171	gradient descent	471
469	upper bound	418
219	large scale	418
299	neural network	415
422	stochastic gradient	413
244	log likelihood	406
245	log log	388
286	monte carlo	383
47	arxiv preprint	382
413	state art	356
343	preprint arxiv	352
477	variational inference	350
83	conference machine	347
320	objective function	335
46	artificial intelligence	328
388	sample complexity	321
327	optimization problem	311
461	training set	308
230	learning research	299
267	matrix completion	296
211	journal machine	291
89	convergence rate	291
110	data sets	284
93	convex optimization	279
368	real world	279
249	loss function	271
361	random variables	270
223	learning algorithm	267
196	ieee transactions	265
178	ground truth	261
263	markov chain	261
246	logistic regression	258
458	training data	255
32	active learning	251
420	step size	249
255	lower bounds	241
485	worst case	239
78	computer science	239
108	data points	239
375	related work	233
443	test set	230
148	figure shows	226
186	high probability	225
99	covariance matrix	222
176	graphical models	222

Good to know: https://www.kaggle.com (https://www.kaggle.com)

- data science community
- competitions on real data
- great learning platform (many working examples available)
- source of interesting data

In [ ]:

```

```