Implementation of machine learning algorithms for different datasets using python programming language.

Renuka sagar, Dr U Eranna

Abstract— WSNs have tremendous potential for building powerful applications, each with its own individual characteristics and requirements. Developing efficient algorithms that are suitable for many different application scenarios is a challenging task. .sensor network designers characterize machine learning as a collection of tools and algorithms that are used to create prediction models. However, machine learning experts recognize it as a rich field with very large themes and patterns. machine learning algorithms provide tremendous flexibility benefits. machine learning algorithms can be categorized by the intended structure of the model. Most machine learning algorithms fall into the categories of supervised, unsupervised and reinforcement learning. In the first category, machine learning algorithms are provided with a labeled training data set. This set is used to build the system model representing the learned relation between the input, output and system parameters. In contrast to supervised learning, unsupervised learning algorithms are not provided with labels (i.e., there is no output vector). Basically, the goal of an unsupervised learning algorithm is to classify the sample sets

Index Terms—accuracy, datasets, kmeans, logistic regression, knn algorithm, supervised, unsupervised.

INTRODUCTION:

Wireless sensor network (WSN) is composed typically of multiple autonomous, tiny, low cost and low power sensor nodes. These nodes gather data about their environment and collaborate to forward sensed data to centralized backend units called base stations or sinks for further processing. The sensor nodes could be equipped with various types of sensors , such as thermal, acoustic, chemical, pressure, weather, and optical sensors. Because of this diversity, WSNs have tremendous potential for building powerful applications, each with its own individual characteristics and requirements. Developing efficient algorithms that are suitable for many different application scenarios is a challenging task. In particular, WSN designers have to address common issues related to data aggregation, data reliability, localization, node clustering, energy aware routing, events scheduling, fault detection and security. sensor network designers characterize machine learning as a collection of tools and algorithms that are used to create prediction models. However, machine learning experts recognize it as a rich field with very large themes and patterns. machine learning algorithms provide tremendous flexibility benefits. machine learning algorithms can be categorized by the intended structure of the model. Most machine learning algorithms fall into the categories of supervised, unsupervised and reinforcement learning. In the first category, machine learning algorithms are provided with a labeled training data set. This set is used to build the system model representing the learned relation between the input, output and system parameters. In contrast to supervised learning, unsupervised learning algorithms are not provided with labels (i.e., there is no output vector). Basically, the goal of an unsupervised learning algorithm is to classify the sample sets to different groups (i.e., clusters) by investigating the similarity between the input samples. Machine learning is important in WSN applications for the following main reasons: 1) Sensor networks usually monitor dynamic environments that change rapidly over time. For example, a node's location may change due to soil erosion or sea turbulence. It is desirable to develop sensor networks that can adapt and operate efficiently in such environments. 2) WSNs may be used for collecting new knowledge about unreachable, dangerous locations (e.g., volcano eruption and waste water monitoring) in exploratory applications. Due to the unexpected behavior patterns that may arise in such scenarios, system designers may develop solutions that initially may not operate as expected. System designers would rather have robust machine learning algorithms that are able to calibrate itself to newly acquired knowledge.

LITERATURE SURVEY:

Introduction to ML

Machine learning (ML) was introduced in the late 1950's as a technique for artificial intelligence (AI) . Over time, its focus evolved and shifted more to algorithms which are computationally viable and robust. In the last decade, machine learning techniques have been used extensively for a wide range of tasks including classification, regression and density estimation in a variety of application areas such as

bioinformatics, speech recognition, spam detection, computer vision, fraud detection and advertising networks. The algorithms and techniques used come from many diverse fields including statistics, mathematics, neuroscience, and computer science. The following two classical definitions capture the essence of machine learning: 1) The development of computer models for learning processes that provide solutions to the problem of knowledge acquisition and enhance the performance of developed systems. 2) The adoption of computational methods for improving machine performance by detecting and describing consistencies and patterns in training data. Applying these definitions to WSNs, we see that the promise of machine learning lies in exploiting historical data to improve the performance of sensor networks on given tasks with - out the need for re-programming. 3) WSNs are usually deployed in complicated environments where researchers cannot build accurate mathematical models to describe the system behavior. Meanwhile, some tasks in WSNs can be prescribed using simple mathematical models but may still need complex algorithms to solve them (e.g., the routing problem). Under similar circumstances, machine learning provides low-complexity estimates for the system model. 4) Sensor network designers often have access to large amounts of data but may be unable to extract important correlations in them. For example, in addition to ensuring communication connectivity and energy sustainability, the WSN application often comes with minimum data coverage requirements that have to be fulfilled by limited sensor hardware resources . Machine learning methods can then be used to discover important correlations in the sensor data and propose improved sensor deployment for maximum data coverage. 5) New uses and integrations of WSNs, such as in cyberphysical systems (CPS), machine-to-machine (M2M) communications, and Internet of things (IoT) technologies, have been introduced with a motivation of supporting more intelligent decision-making and autonomous control . Here, machine learning is important to extract the different levels of abstractions needed to perform the AI tasks with limited human intervention.

Algorithms in ML

ML algorithms are classified into two supervised learning and unsupervised learning.

Supervised Learning: In supervised learning, a labeled training set (i.e., predefined inputs and known outputs) is used to build the system model. This model is used to represent the learned relation between the input, output and system parameters. Supervised learning algorithms are extensively used to solve several challenges in WSNs such as localization and objects targeting (e.g., [21], [22], [23]), event detection and query processing, media access control, security and intrusion detection, and quality of service (QoS), data integrity and fault detection . 1) K-nearest neighbor (k-NN): This supervised learning algorithm classifies a data sample (called a query point) based on the labels (i.e., the output values) of the near data samples. For example, missing readings of a sensor node can be predicted using the average measurements of neighboring sensors within specific diameter limits. There are several functions to determine the nearest set of nodes. A simple method is to use the Euclidean distance between different sensors. Knearest neighbor does not need high computational power, as the function is computed relative to local points (i.e., k-nearest points, where k is a small positive integer). This factor coupled with the correlated readings of neighboring nodes makes knearest neighbor a suitable distributed learning algorithm for WSNs.

Unsupervised Learning: Unsupervised learners are not provided with labels (i.e., there is no output vector). Basically, the goal of an unsupervised learning algorithm is to classify the sample set into different groups by investigating the similarity between them. As expected, this theme of learning algorithms is widely used in node clustering and data aggregation problems. Indeed, this wide adoption is due to data structures (i.e., no labeled data is available) and the desired outcome in such problems. 1) K-means clustering: The k-means algorithm is used to recognize data into different classes (known as clusters). This unsupervised learning algorithm is widely used in sensor node clustering problem due to its linear complexity and simple implementation. The k-means steps to resolve such node clustering problem are (a) randomly choose k nodes to be the initial centroids for different clusters; (b) label each node with the closest centroid using a distance function; (c) recompute the centroids using the current node memberships and (d) stop if the convergence condition is valid (e.g., a predefined threshold for the sum of distances between nodes and their perspective centroids), otherwise go back to step (b).

In recent year, researchers in the adversarial information retrieval community had moved towards machine learning approach to detect Web spam. Actually the Web spam problem can be viewed as a classification problem. Machine learning constructed Web spam classifiers have shown positive results due to their adaptive ability to learn the underlying patterns for classifying spam and non-spam. The WEBSPAM-UK datasets have made a leap in Web spam community for

using various machine learning models. In fact, previously there are few Web spam challenge series Web spam challenge track I 15, II 16 and III 18 which aim is to bring both machine learning and information retrieval community to solve the Web spam labelling problem. Becchetti et al. 5 study several link-based metrics which include rank propagation for links and probabilistic counting to improve the Web spam detection techniques. Moreover, the authors conducted another similar research 7 which include more link-based metrics such as degree correlation and number of neighbours, and as a result the metrics achieve 80.4% detection rate with 1.1% false positive using DT with Boosting on WEBSPAM-UK2002 dataset. Besides link-based features, some researchers 37 propose several content-based features for Web spam detection. The 436 Kwang Leng Goh and Ashutosh Kumar Singh / Procedia Computer Science 70 (2015) 434 – 441 content of Web pages can be modified in order to attract Web users, a technique known as keyword-stuffing. The authors experiment on 105 million Web pages and 86.2% spam pages detected using DT. Stacked graphical learning 32, a meta-learning scheme, has shown positive results using DT in Web spam detection 17. Some researchers 34 take advantage of stacked graphical learning by generating features by averaging known and predicted labels for similar nodes of the graph. The authors achieve improvement of 0.01% F-measure for small graph and 0.111% F-measure for large graph. Gan and Suel 27 propose 8 content features, 14 link-based features and 3 additional features which include number of hosts in the domain, ratio of pages in this host to pages in this domain and number of hosts on the same IP address. The overall features achieved more than 90% F-measure for spam and non-spam detection in Swiss dataset from DT and SVM. Castillo et al. 17 use the combination of link-based features from 7 and content-based features from 37 and experiment on WEBSPAM-UK2006 dataset and result in 88.4% of spam hosts detected with 6.3% false positive using DT. A preliminary study on using linguistic features for Web spam detection is conducted by Piskorski et al. 38 and concluded by providing several discriminating Corleone and General Inquirer attributes that are promising enough to discriminate spam and non-spam. Becchetti et al. 8 perform a detailed statistical analysis that only consider link structure of the Web for Web spam detection. Their experiments show that the performance of all combined features is comparable with that state-ofthe-art spam classifier that use content attributes. Becchetti et al. 6 later use both link and content features to classify spam and non-spam. In addition, the authors use graph clustering algorithms, propagation of predicted labels and stacked graphical learning to improve the classification accuracy using DT with bagging. As a result, their proposed methodology manages to detect up to 88% of spam pages. Linked latent Dirichlet allocation (LDA), an extension of LDA is used for Web spam classification 11. The linked LDA technique consider linkage such as topics are propagated along links in such a way that the linked document directly influences the words in the linking Document. The authors concluded that linked LDA outperforms LDA and other baseline classifier about 3% to 8% in AUC performance. Historical Web page information is important for Web spam classification. Dai et al. 20 propose 1270 temporal features to improve the performance of Web spam classifiers. The features are experimented using SVM on WEBSPAMUK2007 and have shown that their approach improves the F-measure by 30% compared to the baseline classifier which only considers current page content. Martinez-Romo and Araujo 36 presented 42 language model features to represent a Web document that calculate disagreement between two Web pages. The authors experiment using cost sensitive DT with Bagging on WEBSPAMUK2006 and WEBSPAM-UK2007 and show that the language model features improve the F-measure of the former dataset by 6% and latter dataset by 2%. Later on, the authors combined their language model features with 12 qualified link analysis features [35] along with both content and link-based features, the overall features achieve 0.86 F-measure and 0.88 AUC performance in WEBSPAM-UK2006, and 0.40 F-measure and 0.76 AUC performance in WEBSPAM-UK2007 using DT. Li et al. 33 generate 10 new features from link features based on genetic programming and show that the new features are well performed using SVM than 41 standardized link-based features and also 138 transformed link-based features. Though DT is the most used machine learning algorithm when Web Spam classification first started, SVM has become state of the art machine learning model for Web spam classification in recent years as Abernethy et al. 1 obtained the best result in Web Spam Challenge 2007 with area under receiver operating characteristic (AUC) performance of 0.963 using SVM compare to C4.5 DT with AUC performance of 0.935. Yuchun et al. 50 obtained higher AUC results with less time and space using SVM than DT in spam senders behavior analysis. Jia et al. 51 did some simulation research on machine learning models for Web spam detection and their results showed that SVM outperformed both rule-based classifier and decision tree classifier in terms of precision, recall and F1-value. Having said so, Goh et al. 28 have shown that MLP improve the AUC performance up to 14.02% over SVM in WEBSPAM-UK2006 and up to 3.53% over SVM in WEBSPAM-UK2007. Nevertheless, there are other machine learning algorithms available in literatures which will be useful to show

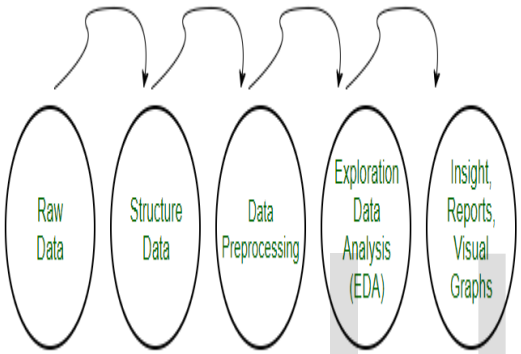
comparison with current state-of-the-art classifier for Web Spam detection.

Objective of the work

1)To implement Machine learning algorithms such as kmeans, K-nearest neighbor, Logistic Regression algorithm.

Data Preprocessing in ML

• Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.



Need of Data Preprocessing

• For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, Random Forest algorithm does not support null values, therefore to execute random forest algorithm null values have to be managed from the original raw data set.

• Another aspect is that data set should be formatted in such a way that more than one Machine

Learning and Deep Learning algorithms are executed in one data set, and best out of them is chosen.

steps involved in data preprocessing.

Rescale the data

When data is comprised of attributes with varying scales, machine learning algorithms are used to rescaling the attributes to have the same scale. This is referred to as normalization and attributes are often rescaled into the range between 0 and 1. This is useful for optimization algorithms. . It is also useful for algorithms that weight inputs like regression and neural networks and algorithms that use distance measures like K-Nearest Neighbors.

2.Standardize Data

Standardization is a useful technique to transform attributes with a Guassian Distribution and differing means and standard deviations to a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.

3. Normalize Data

Normalizing in scikit-learn refers to rescaling each observation (row) to have a length of 1 (called a unit norm in linear algebra). This preprocessing can be useful for sparse datasets (lots of zeros) with attributes of varying scales when using algorithms that weight input values such as neural networks and algorithms that use distance measures such as K-Nearest Neighbors.

4. Binarize Data (Make Binary)

Transform your data using a binary threshold. All values above the threshold are marked 1 and all equal to or below are marked as 0. This is called binarizing data or threshold your data. It is also useful when feature engineering and you want to add new features that indicate something meaningful.

1)To implement Machine learning algorithms such as kmeans, K-nearest neighbour,
Logistic Regression algorithm.

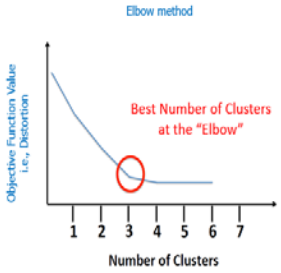
2) kmeans is one of the simplest unsupervised learning algorithm that solves the clustering problem. It groups all the objects in such a way that object in the same group(group is a cluster) are more similar to each other than in other groups.

3) Iris dataset is a table that contains several features of iris flowers of 3 species. Species are Iris-setosa, Iris-versicolor, Iris-verginica. Each flower contains 5 features. Petal length, petal width,sepal length,sepal width.

4) The Iris dataset contains the data for 50 flowers from each of 3 species- setosa,
versicolor & virginca. The data gives the measurements in centimeters of the variables
sepal length & width & petal length & width for each of the flowers.

5) The goal of kmeans algorithm is to perform exploratory analysis of tha data & build a kmeans clustering model to cluster them into groups.

6) The algorithm then iteratively moves the k-centers and selects the datapoints that are closest to that centroid in the cluster. To pick the value of K we use an elbow curve, where the x-axis is the K-value and the y axis is some objective function. A common objective function is the average distance between the datapoints and the nearest centroid. once the dataset is ready, we need to stardize the data. To standardize the data mean. Standard deviation of datasets is calculated.

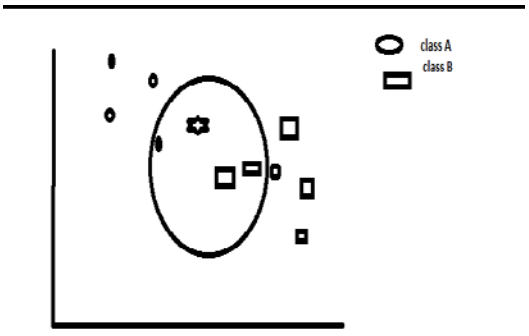


7) KNN is a simple algorithm which uses entire dataset in its training phase. Whenever prediction is required for unseen data. It searches through the entire training dataset for k most similar instances. The data with most training instance is finely returned as prediction.

8) KNN stores all the available data and classifies the new data based on similarity measure.

9) K in KNN denotes no of nearest neighbors which are voting class to the new data or training Data

10) KNN works as follows.



In the above example (o) belong to class A & ( ) belong to class B. if ( ) is a new point . Now the task is to predict whether the new point belongs to class A or class B.

11) To start the prediction the very Ist thing is to select the values of K. K in KNN refers to number of nearest neighbors.

12) In the above fig if k = 3, which means 3 points are least distances to the new point.

13) The value of K in KNN is chosen as the important parameter . cross validation technique is used to check several values of K.

14) Euclidean distance is the direct or least distance between two points A & B.

$$ED = \sqrt{(x2-x1)^2+(y2-y1)^2}$$

15) Manhattan distance is used to calculate real vector using sum of the differences. MD is the distance between A&B measured along the axis at right angles.

$$MD = mod(x2-x1)+(y2-y1)$$

16) Exploring KNN in code: The dataset is IRIS flower dataset which was first introduced in 1936 by famous statistician Ronald fisher & consists of 50observations from each of three species of Iris(Iris setosa,, Iris-virginica, Iris- versicolor).

17) Four features were measured from each sample the length and width of sepals and petals.

18) our goal is to train the algorithm to distinguish the species from one another given the measurements of the 4 features.

19)Logistic Regression is one of the most popular machine learning algorithm for binary classification.

20)LR performs well on wide range of problems using stochastic gradient descent

21)The logistic function is defined as transformed = $\frac{1}{(1+e^{-x})}$

where e is a numerical constant.

22)LR model takes real valued inputs & makes prediction as to the probability of the input belonging to the default class(class0)

23)If the prob > 0.5 output is a prediction for the default class(class0), else the prediction is for the other classs(class1)

24)For this dataset, the logistic regression has 3 coefficients output= $b0+b1*x1+b2*x2$. The job of the learning

algorithm is to discover the best values for the coefficients (b0,b1&b2) based on the training data.

25)p(class=0) = $\frac{1}{(1+e^{-output})}$. Stochastic gradient descent is a procedure used by many machine learning algorithms.

26) It works using a model to calculate a prediction for each instance in the training set & calculating the error for each prediction. We apply stochastic gradient descent for finding the coeff for LR model as follows.

27) For given each training instance calculate a prediction using the current values of the coeff

28) calculate new coeff values based on the error in the prediction. The process is repeated until the model is accurate enough or a fixed number of iterations. Update the model for training instances & correcting errors until the model is accurate.

29)calculate predictions . start by assigning 0.0 to each coefficient & calculate the prob of first training instance that belongs to class 0.

30)calculate new coeff using simple update equation

$$b = b + alpha * (y - prediction) * prediction * (1 - prediction) * x$$

where b=coeff we are updating.

31)Repeat the process, & update the model for each training instance dataset. It is common to repeat the stochastic gradient descent procedure for a fixed number of epochs.

32)At the end of the epoch, calculate the error values for the model and accuracy of the model at each iteration.

Results and Discussions :

Data preprocessing is done as follows:

1.**Rescale the data**

When data is comprised of attributes with varying scales, machine learning algorithms are used to rescaling the attributes to all have the same scale. This is referred to as normalization and attributes are often rescaled into the range between 0 and 1. This is useful for optimization algorithms. . It is also useful for algorithms that weight inputs like regression and neural networks and algorithms that use distance measures like K-Nearest Neighbors. Rescaled iris data is as follows

[0.222 0.625 0.068 0.042]
[0.167 0.417 0.068 0.042]
[0.111 0.5 0.051 0.042]
[0.083 0.458 0.085 0.042]
[0.194 0.667 0.068 0.042]]

2. Standardize Data

Standardization is a useful technique to transform attributes with a Gaussian distribution and differing means and standard deviations to a standard Gaussian distribution with a mean of 0 and a standard deviation of standardized iris data is as follows.

[[-0.901 1.032 -1.341 -1.313]
[-1.143 -0.125 -1.341 -1.313]
[-1.385 0.338 -1.398 -1.313]
[-1.507 0.106 -1.284 -1.313]
[-1.022 1.263 -1.341 -1.313]]

3. Normalize Data

Normalizing in scikit-learn refers to rescaling each observation (row) to have a length of 1 (called a unit norm in linear algebra).

This preprocessing can be useful for sparse datasets (lots of zeros) with attributes of varying scales when using algorithms that weight input values such as neural networks and algorithms that use distance measures such as K-Nearest Neighbors. Normalized iris data is as follows

[[0.804 0.552 0.221 0.032]
[0.828 0.507 0.237 0.034]
[0.805 0.548 0.223 0.034]
[0.8 0.539 0.261 0.035]
[0.791 0.569 0.221 0.032]]

4. Binarize Data (Make Binary)

You can transform your data using a binary threshold. All values above the threshold are marked 1 and all equal to or below are marked as 0.

This is called binarizing data or threshold your data. It is also useful when feature engineering and you want to add new features that indicate something meaningful. Iris binary data is as follows

[[1. 1. 1. 1.]
[1. 1. 1. 1.]
[1. 1. 1. 1.]
[1. 1. 1. 1.]
[1. 1. 1. 1.]]

For a given dataset Kmeans algorithm is applied to group the data . Based on the number of K , data is grouped into clusters. If k=3, the entire iris data is gruped into 3 clusters as 0,1,2.

Implement KNN for any dataset , For a new data to fit into the KNN model, calculate Euclidean distance to predict whether the new data fits into class A or Class B

Iris dataset has 4 features,150 observations & 5class labels(class labels decides which flower belongs to which category)

Step by step procedure to perform KNN

Handle the data: Firstly open the dataset from csv and split into test/train dataset.

Similarity: Calculate the distance between 2 data instances.

```
data1 = [2,2,2,'a']

data2 = [4,4,4,'b']
distance = euclideanDistance(data1, data2, 3)
print('Distance: ' + repr(distance))
```

Distance: 3.4641016151377544

Neighbors: select kneighbors which are at least distance from new data.

```
trainset = [[2,2,2,'a'], [4, 4, 4, 'b']]
testinstance = [5, 5, 5]
k=1
neighbors = getNeighbors(trainset, testinstance, 1)
print(neighbors)
```

[[4, 4, 4, 'b']]

Response: Generate a response from a set of data instance.

This decides whether a new point belongs to class A or class B

```
testset =[[1,1,1,'a'], [2,2,2,'a'], [3,3,3,'b']]
predictions = ['a', 'a', 'a']
accuracy = getAccuracy(testset, predictions)
print(accuracy)
```

Accuracy: summarizing the accuracy of predictions.

33.3333333333

Logistic Regression is implemented using following steps:

Collecting data: The first thing we need to do is load our data file. The data is in CSV format without a header line or any quotes. We can open the file with the open function and read the data lines using the reader function in the csv module.

Analyzing data:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1310 entries, 0 to 1309
Data columns (total 14 columns):
pclass      1309 non-null float64
survived    1309 non-null float64
name        1309 non-null object
sex         1309 non-null object
age         1046 non-null float64
sibsp       1309 non-null float64
parch       1309 non-null float64
ticket      1309 non-null object
fare        1308 non-null float64
cabin       295 non-null object
embarked    1307 non-null object
boat        486 non-null object
body        121 non-null float64
home.dest   745 non-null object
dtypes: float64(7), object(7)
memory usage: 107.5+ KB
```

Data wrangling:

```
pclass        1
survived      1
name          1
sex           1
age         264
sibsp         1
parch         1
ticket        1
fare          2
cabin      1015
embarked      3
boat        824
body        1189
home.dest    565
dtype: int64
```

Train & Test:

```
array([[14,  2],
       [16, 40]])
```

Accuracy check: 0.75

References

[1]W. Richert, L. P. Coelho, "Building Machine Learning Systems with Python", Packt Publishing Ltd., ISBN 978-1-78216-140-0

[2] M. Welling, "A First Encounter with Machine Learning"

[3] M. Bowles, "Machine Learning in Python: Essential Techniques for Predictive Analytics", John Wiley & Sons Inc., ISBN: 978-1-118- 96174-2

[4] S.B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques", Informatica 31 (2007) 249-268

[5] L. Rokach, O. Maimon, "Top – Down Induction of Decision Trees Classifiers – A Survey", IEEE Transactions on Systems,

[6] D. Lowd, P. Domingos, "Naïve Bayes Models for Probability Estimation"

[7] https://webdocs.cs.ualberta.ca/~greiner/C651/Homework2_Fall2008.html

[8] D. Meyer, "Support Vector Machines – The Interface to libsvm in package e1071", August 2015

[9] S. S. Shwartz, Y. Singer, N. Srebro, "Pegasos: Primal Estimated sub - Gradient Solver for SVM", Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, 2007

[10] http://www.simplilearn.com/what-is-machine-learning-and-why-itmatters-article

[11] P. Harrington, "Machine Learning in action", Manning Publications Co., Shelter Island, New York, 2012

[12] http://pypr.sourceforge.net/kmeans.html

[13] K. Alsabati, S. Ranaka, V. Singh, "An efficient k-means clustering algorithm", Electrical Engineering and Computer Science, 1997

[14] M. Andrecut, "Parallel GPU Implementation of Iterative PCA Algorithms", Institute of Biocomplexity and Informatics, University of Calgary, Canada, 2008

[15] X. Zhu, A. B. Goldberg, "Introduction to Semi – Supervised Learning", Synthesis Lectures on Artificial Intelligence and Machine Learning, 2009, Vol. 3, No. 1, Pages 1-130

[16] X. Zhu, "Semi-Supervised Learning Literature Survey", Computer Sciences, University of Wisconsin-Madison, No. 1530, 2005

[17] R. S. Sutton, "Introduction: The Challenge of Reinforcement Learning", Machine Learning, 8, Page 225-227, Kluwer Academic Publishers, Boston, 1992

[18] L. P. Kaelbing, M. L. Littman, A. W. Moore, "Reinforcement Learning: A Survey", Journal of Artificial Intelligence Research, 4, Page 237-285, 1996

[19] R. Caruana, "Multitask Learning", Machine Learning, 28, 41-75, Kluwer Academic Publishers, 1997

[20] D. Opitz, R. Maclin, "Popular Ensemble Methods: An Empirical Study", Journal of Artificial Intelligence Research, 11, Pages 169- 198, 1999

[21] Z. H. Zhou, "Ensemble Learning", National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

[22] https://en.wikipedia.org/wiki/Boosting_(machine_learning)

[23] https://en.wikipedia.org/wiki/Bootstrap_aggregating

[24] V. Sharma, S. Rai, A. Dev, "A Comprehensive Study of Artificial Neural Networks", International Journal of Advanced Research in Computer Science and Software Engineering, ISSN 2277128X, Volume 2, Issue 10, October 2012

[25] S. B. Hiregoudar, K. Manjunath, K. S. Patil, "A Survey: Research Summary on Neural Networks", International Journal of Research in Engineering and Technology, ISSN: 2319 1163, Volume 03, Special Issue 03, pages 385-389, May, 2014

[26] https://en.wikipedia.org/wiki/Instance-based_learning

[27] P. Harrington, "Machine Learning in Action", Manning Publications Co., Shelter Island, New York, ISBN 9781617290183, 2012

[28] J. M. Keller, M. R. Gray, J. A. Givens Jr., "A Fuzzy K-Nearest Neighbor Algorithm", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-15, No. 4, August 1985