Comparison between Maximum Entropy and Naïve Bayes classifiers: Case study; Appliance of Machine Learning Algorithms to an Odesk’s Corporation Dataset

Georgios Maroulis

Submitted in partial fulfilment of the requirements of Edinburgh Napier University for the Degree of MSc. Information Systems Development

In collaboration with oDesk Incorporation

School of Computing

January 2014
Authorship Declaration

I, Georgios Maroulis, confirm that this dissertation and the work presented in it are my own achievement.

Where I have consulted the published work of others this is always clearly attributed;

Where I have quoted from the work of others the source is always given. With the exception of such quotations this dissertation is entirely my own work;

I have acknowledged all main sources of help;

If my research follows on from previous work or is part of a larger collaborative research project I have made clear exactly what was done by others and what I have contributed myself;

I have read and understand the penalties associated with Academic Misconduct.

I also confirm that I have obtained informed consent from all people I have involved in the work in this dissertation following the School's ethical guidelines

Signed:

Date: 06/01/2014

Matriculation no: 40093896
Data Protection Declaration

Under the 1998 Data Protection Act, The University cannot disclose your grade to an unauthorised person. However, other students benefit from studying dissertations that have their grades attached.

Please sign your name below one of the options below to state your preference.

The University may make this dissertation, with indicative grade, available to others.

The University may make this dissertation available to others, but the grade may not be disclosed.

The University may not make this dissertation available to others.

Georgios Maroulis
Abstract

Natural Language Processing, Artificial Intelligence and Machine Learning are rapidly growing technologies that their appliance unlocks great opportunities and possibilities for the implementation of automated decision making systems.

The case study of this dissertation was to compare the Naïve Bayes and Maximum Entropy Machine-Learning algorithms for text classification by building an application with a portion of oDesk’s incorporation (oDesk) database as a dataset. The literature review of this dissertation introduces and explores how classification problems can be solved with machine-learning algorithms. Additionally, it is explored and presented natural language processing techniques that could be applied for data mining and text processing. Finally, this literature review ends with a technology review regarding which programming languages and libraries could be used in order to implement a text classification system. The programming language that has been used was Python (Python) with nltk (NLTK 2.0) library for the development of the experiments.

The implementation process and the results of the experiments have shown the significance of data processing and how this procedure can affect the classification results. Finally, the low performance of the implemented application does not affect the conclusion that machine-learning systems could be applied to large incorporations and organizations to improve the business processes and the customer experience.
Contents

1 INTRODUCTION ........................................................................................................... 11
  1.1 Background of Machine Learning Techniques ...................................................... 11
    1.1.1 Motivation ........................................................................................................ 12
  1.2 Case Study .............................................................................................................. 12
  1.3 Aims and Objectives ............................................................................................ 13

2 CLASSIFICATION ...................................................................................................... 15
  2.1 Definition – Text Classification ............................................................................ 15
  2.2 Text Classification Frameworks ........................................................................... 15
    2.2.1 Supervised ...................................................................................................... 16
    2.2.2 Semi-Supervised ............................................................................................ 16
    2.2.3 Unsupervised .................................................................................................. 16
  2.3 Machine Learning Algorithms .............................................................................. 17
    2.3.1 Supervised Algorithms .................................................................................. 17
    2.3.2 Unsupervised Algorithms .............................................................................. 19
  2.4 Comparison Between Naïve Bayes and MaxEnt Classifiers .................................. 22
  2.5 Applications of Text Classification techniques ..................................................... 27
  2.6 Summarization ...................................................................................................... 30

3 USEFUL TECHNIQUES .............................................................................................. 32
  3.1 Natural Language Processing (NLP) ..................................................................... 32
  3.2 Techniques for NLP Tasks ..................................................................................... 32
    3.2.1 Regular Expressions ....................................................................................... 33
    3.2.2 Stemming Words ............................................................................................ 34
    3.2.3 N-Grams ......................................................................................................... 35

4 TECHNOLOGY REVIEW .............................................................................................. 37
  4.1 Programming Languages ....................................................................................... 37
  4.2 Useful Packages, Distributions and Working Environments ................................ 37
    4.2.1 Python ............................................................................................................ 37
    4.2.2 Java ................................................................................................................ 38
  4.3 Conclusion .............................................................................................................. 39

5 THE APPLICATION ...................................................................................................... 40
  5.1 The scope of the application .................................................................................. 40
List of Tables

Table 1 - Classification Methods and Class Labels ........................................ 15
Table 2 – Algorithms and Classification Frameworks .................................... 17
Table 3 – Characteristics summarization of Unsupervised Learning Algorithms 19
Table 4 – Usage of K-Means Clustering and Fuzzy c-Means Clustering
   (Dougherty, 2013).................................................................................. 22
Table 5 – Advantages and Disadvantages of NB and MaxEnt .................... 26
Table 6 – NLP Techniques ......................................................................... 32
Table 7 – N-Grams .................................................................................... 35
Table 8 – Bigrams per class ....................................................................... 50
Table 9 – The Chosen Bigrams for the Feature List .................................... 51
Table 10 – Prediction Accuracy Metric for the Naïve Bayes Algorithm ....... 56
Table 11 – Prediction Accuracy Metric for the Maximum Entropy Algorithm ..... 57
Table 12 – Most Informative Features For Maximum Entropy and Naïve Bayes Classifiers ............................................................................ 58
List of Figures

Figure 1 - SVM ............................................................................................................... 18
Figure 2 – Decision Tree Schema .................................................................................... 19
Figure 3 – Sample Data Chart .......................................................................................... 20
Figure 4 – Results of K-Means after one iteration............................................................... 21
Figure 5 – Arrows points the cluster center .................................................................... 21
Figure 6 - Maximum Entropy with Flat/Optimised prior .................................................. 24
Figure 7 - Naïve Bayes with Flat/Optimised prior ............................................................. 24
Figure 8 - Naïve Bayes and MaxEnt with dependent features ......................................... 25
Figure 9 - Naïve Bayes and MaxEnt with independent features ...................................... 25
Figure 10 - Results of the algorithmic comparison (2002) ................................................. 26
Figure 11 – kNN-Support Vector Machines comparison results ....................................... 28
Figure 12 – Data Set analysis: Sentences counts by reviews. ........................................... 29
Figure 13 – Results of the experiment with Naïve Bayes .................................................. 30
Figure 14 – Results of the experiment with SVM ............................................................. 30
Figure 15 – Examples of the ‘title’ attribute ................................................................... 41
Figure 16 – Examples of the ‘description’ attribute .......................................................... 41
Figure 17 – Example of the ‘skills’ attribute ................................................................... 41
Figure 18 - Data Frame of the Data Set .......................................................................... 42
Figure 19 - Percentage of each category to the total amount of job posts ....................... 42
Figure 20 - Pie Chart of the Categories ......................................................................... 43
Figure 21 – Function that returns the description for a specific category ......................... 44
Figure 22 - Before............................................................................................................. 45
Figure 23 - After ............................................................................................................... 45
Figure 24 - Example of tokenized data ............................................................................. 46
Figure 25 - Example of stopwords removal ................................................................. 46
Figure 26 – Remove Numbers Function ........................................................................ 47
Figure 27 – Bigram Formation function ......................................................................... 48
Figure 28 - Frequency Distribution Function .................................................................. 48
Figure 29 – Function to sort the Frequency Distribution Data Dictionary ....................... 49
Figure 30 – Example of Bigrams Frequencies Distributions ........................................... 49
Figure 31 – The process of supervised learning (Dougherty, 2013) ................................. 53
Figure 32 – Create Feature Set ...................................................................................... 54
Figure 33 – Calculation of the description and features Set arrays. .................................. 54
Figure 34 – Data Split of the Features set ........................................................................ 55
Figure 35 – Train sets .................................................................................................... 55
Figure 36 – Test Sets ...................................................................................................... 55
List of Equations

Equation 1 - The Bayes rule for a classifier ................................................................. 22
Equation 2 – Naïve Bayes Classifier equation ............................................................. 22
Equation 3 – Maximum Entropy Distribution ............................................................... 23
Equation 4 – Normalizing Factor ................................................................................. 23
Equation 5 – Example of features with word counts ................................................... 23
Equation 6 - The non-uniform prior for NB ................................................................. 24
Equation 7 – Maximum Entropy ................................................................................. 25
Equation 8 – Example of Porter Algorithm ................................................................. 34
Equation 9 – Example of probability calculation of a sequence given the subsequence .................................................................................................................. 35
Equation 10 – Example of probability calculation with counts ................................. 35
Equation 11 – Prediction Accuracy Metric ................................................................. 53
Acknowledgements

This dissertation was an unprecedented and demanding procedure. There was no prior knowledge or experience on the field of Machine-Learning and limited programming skills. Throughout this challenge, I am truly indebted and thankful to my supervisor Dr. Emma Hart and Dr. Panagiotis Papadimitriou. Without their guidance and their help this research would not be able to be completed. Additionally, I am really grateful to my business partners Lara Findlay, Joshua Carson, Andrzej Schmidt and Abdullah Alsharif, for their patience and advices. Finally, I owe sincere and earnest thankfulness to my parents Lamprini and Ioannis, my brother Nikos, to Elena and my flat mates Ioannis and Jorge for their support.

Georgios Maroulis
1 Introduction

1.1 Background of Machine Learning Techniques

Nowadays, there is a worldwide discussion regarding ways and approaches on the manipulation of data and the extraction of useful information out of them. Machine learning techniques are involved in this discussion. Intelligent systems that will learn from data and would be able to predict or make suggestions are needed. Before we dive into more depth about this rapidly growing technology is really important to present two historical scientific figures that have contributed to the machine learning and statistical learning techniques.

The first one is Alan Turing (1950). Turing was a great contributor in the field of artificial intelligence and machine thinking of its time. His imitation game, widely known as “Turing Test” has unlocked the uncountable potentials computer science could offer. Specially, if we consider that the whole world was trying to recover from the 2nd World War, we can also consider him as a great personality who has put his own stone to the sciences and social evolution.

Turing’s game has had three participating subjects. Two subjects were humans and one was a computer. One of the humans had the role of the interrogator. Additionally, the two other participants were at separate rooms. The goal of the game for the interrogator was the distinction on who is the human and which is the computer after asking a series of questions through a teletype. The goal of the computer was to fool the interrogator and the goal of the second human was to persuade the interrogator that the other participant is a machine.

This game was one of the first experiments on machine learning and artificial intelligence. The critic on the results of Turing’s game is out of the scope of this dissertation.

The second person and a scientific personality we could not omit, is Noam Chomsky (1956). Chomsky’s research field varies from philosophical-political reviews, to finite state machines and through the years he has influenced many generations of people in all the areas he has been involved. Regarding this dissertation we distinct his work on the field of natural language processing and
language modelling. He was the first who considered finite state machines as a grammar characterization and interpreted a finite-state language as a language that have been generated by finite state grammar (2008).

A lot of progress has been done in the field of machine learning after the 1950s with many contributions from great scientists. It has been chosen the above two historical scientific figures because their work and their passion motivates until today. Nowadays the resources and access to information is extremely easy and fast. Turing and Chomsky influence because of their passion to discover and change the world. What they have achieved through their research and willingness is extraordinary regarding the knowledge of their time and their access to resources.

1.1.1 Motivation

There are different aspects for each individual person that could motivate them in order to have a certain a way of living, to produce research or to augment their creativity and build something from scratch. The above personalities gave an original motivation to this dissertation and its topic. More specifically, it has been influenced the willingness and the passion to research a rapidly growing technology and field. As an extension to this, the acquired knowledge out of this dissertation would be used to solve real life problems and find solutions that could help people’s evolution and every day life. Machine learning and Natural Language Processing (NLP) technologies are fascinating and the future on how people will process things and make decisions. Finally, the ability to explore how system like that can be implemented and how is it able to predict or make recommendations is a challenging and creative procedure.

1.2 Case Study

The case study of this dissertation is based on a data set, which is provided by ODesk Corporation (oDesk).

Odesk is a large job market place based in San Francisco, CA, U.S.A. The first steps of this company began back in 2002 by Odysseas Tsatalos and Stratis Karamanlakis. Through the years oDesk adopted the following principles for the freelancers and the clients (employers). It allows freelancers to be paid on time for all the hours they have worked, set the hourly charge rate and skills and provides to them all the necessary tools in order to start an online business. Additionally, for the clients of the platform gives the tools to audit and pay for the real hours have been worked find the right candidate for the job and manage a global team easily (oDesk).
The way this online job market works is simple. Both employers and employees have a profile. The employer posts a job with various sort of information needed for a candidate employee. Some examples of basic information provided are the type of the job, the budget, the category and description. Furthermore, the freelancer can see other interested piece of information like the total amount of money have been spent in this platform by the employer, hours that have been billed, jobs that have been posted or the total hires. Moreover, the employee has the ability to check the employee’s profile and gather information for him such as reviews of previous works and the hourly charge rate.

An interesting service oDesk offers is online tests for a variety of skills. That helps in many different ways, both for the job seeker and the employer. It provides a proof of skills that claimed in the freelancer’s profile and makes the interview-hiring process faster. Finally, the most innovative element of this online market place platform is that it can handle all the payments and provides a clever snapshot system as a proof that the hours have been invoiced have been spent actually in the project.

1.3 Aims and Objectives

This dissertation aims to answer the following research questions:

- On what extension machine learning techniques could be applied to businesses and organizations to automate and improve procedures?
- Could a Machine Learning technique be applied to classify text efficiently?

Furthermore, the objectives of this research are:

- Undertake a review of Machine Learning Techniques
- Explore and review Natural Language Processing techniques for text processing and data mining
- Review of technologies, Programming Languages and Libraries that could be applied to implement a Machine Learning system.
- Analysis of the case study data set
- Appropriately pre-process data set by applying Natural Language Processing techniques.
- Extract features from data
- Apply selected Machine Learning algorithms
- Evaluate and analyse the results

Ideally, the classifiers would take as inputs three attributes of the dataset, which are in the form of raw text and classify them into the correct categories.
2 Classification

2.1 Definition – Text Classification

To begin with, classification is a process that we apply every day in our life. For example, a post officer classifies the letters according to different areas and the mailman classifies the letters by taking into account the addresses. Additionally, nowadays we can see that a lot of movies are classified regarding the plot and language, as appropriate or inappropriate. Those are classification examples from our everyday routine. There is a strong need though to automate procedures and create intelligent systems that could learn how to classify and predict the correct class for us.

In Summary, A short definition of classification could be the process of choosing the correct class for a given input (Bird, Klein & Looper, 2009, p.221). Since we emphasize in text classification in this literature review, according to Fuchun and Huang (2006), “Text classification is the problem of assigning a document D to one of a set of $|C|$ pre-defined categories $\mathcal{C} = \{c_1, c_2, \ldots, c_{|C|}\}$. Moreover, it is usual practice to provide a learning algorithm in a supervised framework, with a set of N labeled training data in order to produce a function, which will map each document to the corresponding category.

2.2 Text Classification Frameworks

There are three frameworks of Classification: supervised, semi-supervised and unsupervised.

Table 1- Classification Methods and Class Labels

<table>
<thead>
<tr>
<th>Classification Method</th>
<th>Class Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised Learning</td>
<td>Must be known</td>
</tr>
<tr>
<td>Semi-Supervised</td>
<td>Not necessary, not all Labels are known</td>
</tr>
<tr>
<td>Learning</td>
<td>Unknown</td>
</tr>
<tr>
<td>Unsupervised Learning</td>
<td>Unknown</td>
</tr>
</tbody>
</table>
Table 1 shows the importance to know the class labels before we apply a classification framework.

2.2.1 Supervised
We can refer to a classifier as supervised when it is build referred to a training corpora where for each input, this corpora includes the correct label (2009, p.222). As a result, learning methods that contain this approach are called supervised learning methods (Table 8).

2.2.2 Semi-Supervised
As we stated above supervised data depends on labeled data. That attribute limits their applicability. Machine Learning methods, which are able to combine, labeled with unlabeled data are called semi-supervised learning methods (Sandow and Zhou, 2007). In an extension, that means that we can use supervised and unsupervised learning techniques to solve a classification problem (Table 8).

2.2.3 Unsupervised
When an analyst does not participate in the algorithm’s learning process we can define this approach as unsupervised classification technique (Table 8). There are broad ranges of methods that can be applied in unsupervised classification and we could claim as the most commons, the usage of clustering algorithms (Richards, 2013, p.319).

The Algorithms and Classification Frameworks table shows the three frameworks and a list of machine-learning algorithms that can be applied to each one of them (Table 2).
### Table 2 – Algorithms and Classification Frameworks

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Supervised</th>
<th>Semi-Supervised</th>
<th>Unsupervised</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve Bayes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Maximum Entropy</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>K-Means Clustering</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>S.V.M.</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Artificial Neural Networks</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### 2.3 Machine Learning Algorithms

#### 2.3.1 Supervised Algorithms

The most common used supervised algorithms and models are Naïve Bayes, Maximum Entropy (MaxEnt), Support Vector Machines and Decision Tree.

**2.3.1.1 Naïve Bayes**

We could apply a Naïve Bayes Classifier for learning purposes where every instance \( x \) is analyzed by a combination of features and where by targeting the function \( f(x) \) we could assign any class from a finite set of classed \( C \). Moreover, a training set is provided, a new instance is created and it is described by the features \( (f_1, f_2, f_3, ..., f_n) \). The task of the Naïve Bayes classifier is to predict the target value for this new instance (Mitchel, p.177).

**2.3.1.2 Maximum Entropy**

According to Nigam, Lafferty and McCallum (1999, p.61), maximum entropy in general, is a technique that helps us to estimate probability distribution from data. Moreover, according to them, the principle of MaxEnt is that the distribution should be as uniform as it can be, when nothing is known. We use labeled data to train the MaxEnt classifier and create a model, with a set of constrains that will characterize the class expectations for the distribution.
2.3.1.3 Support Vector Machines (SVM)

The SVMs (Figure 1) are binary or two-class classifiers. They are able to solve linear problems by “maximizing the margin, the perpendicular distance across the hyperplane to the closest instances (the support vectors) on either side of it”. In the case that the training data set is not linearly separable, there is a possibility to find a solution by introducing slack variables. Particularly, we can set the least error that incurs in the hyperplane. Another approach is to map the data set by choosing the most suitable basis functions into a higher-dimensional space, where the problem could become linear (2013, p.118).

Figure 1 - SVM

Figure 1 presents an “Illustration of optimal separating hyper plane, hyper planes and support vectors” (Khan A., Baharudin B., Lee L.H. & Khan K., 2010).

2.3.1.4 Decision Tree

The decision tree algorithm (Figure 2) is used to solve non-linear classification problems. This algorithm constructs a “top-down tree type structure” recursively. Furthermore, it models a group for all the known values of the testing property using features that have gained maximum information to classify samples by testing all features like ID3, C45, C5 (2011,p.1999). Furthermore, as Khan A., Baharudin B., Lee L.H. & Khan K. (2010) claim that the decision tree algorithm is one of the most efficient decision support tools with many advantages. The main advantage according to them is the adaptation of non-experts users because it can be easily understood and interpreted. Hence, the representation of the results as a tree figure gives a strong advantage to discuss and evaluate the founding wherever it is a customer, a research supervisor or a software engineer.
Figure 2 shows the training data and the decision tree diagram (Gorunescu F., 2011).

Table 3 – Characteristics summarization of Unsupervised Learning Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Special Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve Bayes</td>
<td>Based on assumptions.</td>
</tr>
<tr>
<td></td>
<td>Performs well with independent features.</td>
</tr>
<tr>
<td>Maximum Entropy</td>
<td>Estimates the probability distribution from data.</td>
</tr>
<tr>
<td></td>
<td>Performs well with dependent features.</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>Easy to adapt from non-expert users.</td>
</tr>
<tr>
<td>Support Vector Machines</td>
<td>Solves Linear Problems.</td>
</tr>
<tr>
<td></td>
<td>Two class or Binary Classifier.</td>
</tr>
</tbody>
</table>

2.3.2 Unsupervised Algorithms

There are various unsupervised machine-learning algorithms. In the following section we have chosen to report an unsupervised algorithm for clustering, the K-Means Clustering. We have chosen to present this algorithm since it is the most commonly used (Dougherty, 2013). Another algorithm that it could be used for
unsupervised learning is Fuzzy c-Means Clustering and we can see the usage of this algorithm in table 4.

2.3.2.1 K-Means Clustering

The most commonly used and known algorithm, in unsupervised learning for clustering, is K-Means Clustering (K-Means) algorithm.

In order to apply K-Means, we initialize the algorithm with the number of clusters we want (N). Then the algorithm is focusing on N, of what is known as cluster centroids. Then, K-Means will pick any of the clusters posts and assign it centroids to its feature vector. The next step for the algorithm is to iterate all other posts and assign to them the more near to them centroid as their present cluster. Moreover, it will transfer the centroids into the middle of all those vectors of that specific class.

This procedure changes the cluster’s assignment. As a result, a number of posts are more near to a different cluster. Consequently, the algorithm will update the assignments for all those posts that have changed. This procedure takes place, as long as the centroids have been moved at an amount that could be considered. Finally, after a number of iterations, these movements of the centroids will fall under a threshold and we could consider that clustering have finished. We can review a simple application on how K-Means works at Figures 3-5 (Richert W. & Coelho P.L., 2013, p.63).

Figure 3 – Sample Data Chart

Figure 3 represents a dataset where each vector is a document with two words (Richert W. & Coelho P.L., 2013).
Figure 4 – Results of K-Means after one iteration

Figure 4 represents the results of K-Means iteration to our example set. At this point K-Means takes any two vectors as a starting position, set labels to the remaining and finally upgrades the cluster center into the new center coordinate of all the points existing in that cluster (Richert W. & Coelho P.L., 2013).

Figure 5 – Arrows points the cluster center

At Figure 5 the arrows represents the changes of the cluster center. Depending on the tolerance thershold, after a few iterations the clusters does not move any more (Richert W. & Coelho P.L., 2013).
Finally, we can see from the above example the differences between a supervised and unsupervised approach. K-Means is a good example on how to solve an unsupervised problem. We could find the clusters on our data and form the categories that our set should be classified.

Table 4 – Usage of K-Means Clustering and Fuzzy c-Means Clustering (Dougherty, 2013)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-Means Clustering</td>
<td>The clustering is partitional. There are formed nonoverlapping clusters out of the dataset. Each object of the data belongs to only one cluster.</td>
</tr>
<tr>
<td>Fuzzy c-Means Clustering</td>
<td>Each object of the dataset can belong to one or more clusters. This is achieved by connecting each object with a cluster by calculating a set of weights.</td>
</tr>
</tbody>
</table>

2.4 Comparison Between Naïve Bayes and MaxEnt Classifiers

The Naïve Bayes classifier is based on the classical probabilistic Bayes rule where we assume that the features \((f_1, f_2, f_3, \ldots, f_n)\) are independent.

Equation 1 - The Bayes rule for a classifier

\[
P(C|f_1, f_2, f_3, \ldots, f_n) = \frac{P(C)P(f_1, f_2, f_3, \ldots, f_n|C)}{P(f_1, f_2, f_3, \ldots, f_n)}
\] (1.0)

In Equation 1 we can see that the class C is dependent on the features \((f_1, f_2, f_3, \ldots, f_n)\). The denominator is a stable that normalize the probabilities so they always add 1. We are able to omit the denominator since it is not depend on the class C. Hence, the Naïve Bayes Classifier equation could be re-written as a product of the component probabilities:

Equation 2 – Naïve Bayes Classifier equation
\[
P(C|f_1, f_2, f_3, \ldots, f_n) \propto P(C)P(f_1|C)P(f_2|C)P(f_3|C) \ldots P(f_n|C)
\]

\[
\propto P(C)\prod_{i=1}^{n} P(f_i|C) \quad (1.1)
\]

By applying the above formula (1.1) the Naïve Bayes Classifier assigns test data to a class with the maximum a posterior (MAP) probability (Dougherty, 2013, p.53).

The Maximum entropy distribution in the usual exponential form (Pietra et al., 1995):

Equation 3 – Maximum Entropy Distribution
\[
P(c|d) = \frac{1}{Z(d)} \exp(\sum \lambda_i f_i(d; c)) \quad (1.2)
\]

Equation 4 – Normalizing Factor
\[
Z(d) = \sum_c \exp(\sum \lambda_i f_i(d; c)) \quad (1.3)
\]

In equation (1.2) every \(f_i(d; c)\) is a feature for the classifier, the parameter \(\lambda_i\) is to be estimated and \(Z(d)\) (1.3) is a factor that will normalise the result to an appropriate probability (Fuchun, P., Huang, X., 2006). The maximum entropy classifier in order to learn the features can use the Generalized Iterative Scaling (GIS) and Improved Iterative Scaling (IIS) algorithms (Pietra et al., 1995).

Furthermore, we need a set of constrains to apply to the classifier. The constrains are build up of features (e.g. \(f_i(d; c)\)). Although, features are applied to many classification algorithms, especially for maximum entropy, any element that is able to separate a document should be considered as a possible feature (Fuchun, P., Huang, X., 2006).

Equation 5 – Example of features with word counts
\[
f_{w;c'}(d; c) = \begin{cases} 
0, & \text{if } c \neq c' \\
\frac{N(d; w)}{N(d)}, & \text{otherwise} \end{cases} \quad (1.4)
\]

\(N(d; w):\) The number of times a word \(w\) appers to document \(d\). 
\(N(w):\) The total number of words in the document \(d\) (Fuchun, P., Huang, X., 2006).
In order to compare the two algorithms we would review the results of (Osborne, M, 2002) and (Pang, B., Lee, L. & Vaithyanathan, S., 2002).

Osborne (2002) has used 80 conference papers as data set, exactly the same documents that Teufel (2001) has used in her experiments. He applies Naïve Bayes and maximum entropy for sentence extraction to his application. The features he has used are: Word Pairs, Sentence Length, Sentence Position and Limited Discourse features. The author also reports that for the MaxEnt he deleted the features that occur less than four times although that a frequency cutoff did not benefit the Naïve Bayes. The results of his experiments are presented at Figures 6 - 9.

Useful Notation (Osborne, 2002):

F2 score: \( r = \frac{j}{m} \), \( p = \frac{j}{k} \), \( \text{F2} = \frac{2pr}{p+r} \)

\( r = \text{Recall} \)

\( p = \text{Precision} \)

\( j = \text{number of correct sentences in summary} \)

\( k = \text{number of sentences in summary} \)

\( m = \text{number of correct sentences in the document} \)

**Figure 6 - Maximum Entropy with Flat/Optimised prior**

<table>
<thead>
<tr>
<th>Features</th>
<th>Flat prior</th>
<th>Optimised prior</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F2</td>
<td>P</td>
</tr>
<tr>
<td>Word pairs</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>and sent length</td>
<td>25</td>
<td>63</td>
</tr>
<tr>
<td>and sent position</td>
<td>28</td>
<td>62</td>
</tr>
<tr>
<td>and discourse</td>
<td>35</td>
<td>63</td>
</tr>
</tbody>
</table>

**Figure 7 - Naïve Bayes with Flat/Optimised prior**

<table>
<thead>
<tr>
<th>Features</th>
<th>Flat prior</th>
<th>Optimised prior</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F2</td>
<td>P</td>
</tr>
<tr>
<td>Word pairs</td>
<td>26</td>
<td>29</td>
</tr>
<tr>
<td>and sent length</td>
<td>31</td>
<td>33</td>
</tr>
<tr>
<td>and sent position</td>
<td>33</td>
<td>34</td>
</tr>
<tr>
<td>and discourse</td>
<td>38</td>
<td>39</td>
</tr>
</tbody>
</table>

We can see that Maxent has a better performance \( P \) than Naïve Bayes when it is applied a prior. When it is not applied, Naïve Bayes (NB) has a better performance. Also, NB has a worst performance when a prior is applied.

**Equation 6 - The non-uniform prior for NB**
\textit{label}(s) = \arg\max_{c \in \mathcal{C}} P(c) \prod_{i=1}^{n} P(g_i | c) \quad (1.5)

As we see at (1.5) \( P(c) \) is the prior. For the MaxEnt we have:

Equation 7 – Maximum Entropy

\textit{label}(s) = \arg\max_{c \in \mathcal{C}} F(c) \exp \sum_{i} \lambda_i f_i(c, s) \quad (1.6)

Where \( F(c) \) is a function similar to the prior when an unnormalised classifier is used.

Figure 8 - Naïve Bayes and MaxEnt with dependent features

<table>
<thead>
<tr>
<th>Features</th>
<th>Naive Bayes</th>
<th>Maxent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word pairs</td>
<td>F2 30 P 34 R</td>
<td>F2 32 P 33 R</td>
</tr>
<tr>
<td>and sent length</td>
<td>35 38 32</td>
<td>99 100 99</td>
</tr>
<tr>
<td>and sent position</td>
<td>40 41 39</td>
<td>100 100 100</td>
</tr>
<tr>
<td>and discourse</td>
<td>43 44 41</td>
<td>99 100 97</td>
</tr>
</tbody>
</table>

Figure 9 - Naïve Bayes and MaxEnt with independent features

<table>
<thead>
<tr>
<th>Features</th>
<th>Naive Bayes</th>
<th>Maxent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word pairs</td>
<td>84 74 97</td>
<td>25 15 91</td>
</tr>
<tr>
<td>and sent length</td>
<td>85 75 97</td>
<td>100 100 100</td>
</tr>
<tr>
<td>and sent position</td>
<td>84 73 97</td>
<td>100 100 100</td>
</tr>
<tr>
<td>and discourse</td>
<td>84 74 97</td>
<td>100 100 100</td>
</tr>
</tbody>
</table>

As we can see in Figure 8 when we have dependent features MaxEnt classifiers produces high results in performance, compared with NB. On the other hand, MaxEnt produces poor results when we have independent features where NB scores high performance (Figure 9). It is not a surprise though because as we have seen at (1.0) by definition each class is independent at NB, regarding the features.

Pang, B., Lee, L. & Vaithyanathan, S (2002) have used Internet Movie Database (IMDb) archive of the rec.arts.movies.reviews newsgroup as a data set for their application. Particularly, it has been selected movie reviews where the rating of the review is either an arithmetic value or stars. The authors consider the classification problem with an overall sentiment scope, by deciding if a document is negative or positive. We can see the features that have been used the results in the following Figure 10.
We have omitted the column with the results of SVM because it is out of the scope of our comparison. As we can see at this research we cannot say which algorithm outcomes the other. Regarding the features, the authors point out that unigrams on each own or to the features that they were participating brought the best performance.

Figure 10 - Results of the algorithmic comparison (2002)

<table>
<thead>
<tr>
<th>Features</th>
<th># of features</th>
<th>frequency or presence?</th>
<th>NB</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) unigrams</td>
<td>16165</td>
<td>freq.</td>
<td>78.7</td>
<td>N/A</td>
</tr>
<tr>
<td>(2) unigrams</td>
<td></td>
<td>pres.</td>
<td>81.0</td>
<td>80.4</td>
</tr>
<tr>
<td>(3) unigrams+bigrams</td>
<td>32330</td>
<td>pres.</td>
<td>80.6</td>
<td>80.8</td>
</tr>
<tr>
<td>(4) bigrams</td>
<td>16165</td>
<td>pres.</td>
<td>77.3</td>
<td>77.4</td>
</tr>
<tr>
<td>(5) unigrams+POS</td>
<td>16695</td>
<td>pres.</td>
<td>81.5</td>
<td>80.4</td>
</tr>
<tr>
<td>(6) adjectives</td>
<td>2633</td>
<td>pres.</td>
<td>77.0</td>
<td>77.7</td>
</tr>
<tr>
<td>(7) top 2633 unigrams</td>
<td>2633</td>
<td>pres.</td>
<td>80.3</td>
<td>81.0</td>
</tr>
<tr>
<td>(8) unigrams+position</td>
<td>22430</td>
<td>pres.</td>
<td>81.0</td>
<td>80.1</td>
</tr>
</tbody>
</table>

Table 5 – Advantages and Disadvantages of NB and MaxEnt

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
</table>
| NB        | • Independent Assumptions  
• Can be trained with small amount of data  
• Performs well with independent features | • Limits Applicability  
• We could loose performance with the assumptions |
| MaxEnt    | • Performs well with depended features  
• Uses algorithms like GIS and IIS to apply features | • Low Performance with independent features.
• The feature selection could become a complex |
Table 5 records some key advantages and disadvantages of NB and MaxEnt.

As we have seen from the first experiment in this section MaxEnt, in general, performed better than Naïve Bayes. The second experiment did not have so clear results in order that we could conclude to a safe statement. Finally, in this dissertation’s experiments, there will be used different features alone and combined together in order to achieve clearer results for algorithmic analysis and comparison, in order to solve a classification problem.

2.5 Applications of Text Classification techniques

Text classification applies to in numerous applications to our everyday digital and non-digital life. Everything in the World Wide Web is text oriented and its usage has been expanded rapidly the last decade. This fact has created a strong need to process the metadata that have been produced. Text classification techniques and algorithms lead the way in order these produced metadata to make sense.

We can find various systems and applications to the current literature. Some examples of companies and institutions, which have used classification solutions in user data and metadata, are:

- Amazon (Amazon Inc.)

Amazon the world wide online store applies classification techniques in order to make recommendations. For example it is really common when we visit Amazon to have recommendations like customers who have purchased this “A” product bought also the product “B”(Richert W. & Coelho P.L.).

- Netflix (Netflix Inc.)

Netflix is an online movie rental company. Regarding our research, Netflix started a really interesting and costly competition back in 2006. It provided a large amount of user’s movie ratings from their database and challenged the public to improve the system rating prediction system by 10% or more. The prize for this
competition was one million American dollars and it lasted three years (Richert W. & Coelho P.L.).

From all the above we can see how big organizations like Amazon and Netflix apply machine learning technologies for prediction of some values or recommendations for the customers. Additionally, we can distinct how important it is for these companies to not only improve their products but to improve the user’s-customer’s experience. For example lets take someone who uses Netflix the last three months. If his movie preferences are the adventure movies and the recommendations from Netflix are romantic comedy movies, then the customer will be disappointed.

Finally, recommendations and rating prediction are services that have started less than decade ago and now have started to mature in the online industry. There are a lot of challenges in order to build a successful prediction system and for each one of them lays a classification problem. The nature of machine learning applications includes a lot of experiments and the need to keep an eye close for further improvements.

Below is presented a review of two-machine learning applications that shows how classification techniques can be applied to solve a problem. Both of them contain results of classification algorithmic comparison and fit really well to purpose of this dissertation.

The scientific paper *A Comparative Study of Citations and Links in Document classification* (Couto, T., Cristo, M., Goncalves, M., Calado, P., Ziviani, N., Moura, E., Ribeiro-Neto, B., 2006) presents an application that attempting to compare the digital libraries citation and Web Links in the field of automatic text classification. The data set that has been used is ACM8 a sub-collection of ACM electronic library and Cade12, a web pages collection from Cade directory. The machine learning algorithms that are used are *k Nearest Neighbour* (kNN) and *Support Vector Machines*. The results are presented to Figure 11.

Figure 11 – kNN-Support Vector Machines comparison results

<table>
<thead>
<tr>
<th>Method</th>
<th>Similarity</th>
<th>micP1</th>
<th>macP1</th>
<th>Gains (%) over text classifier</th>
<th>Method</th>
<th>Similarity</th>
<th>micP1</th>
<th>macP1</th>
<th>Gains (%) over text classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNN</td>
<td>co-citation</td>
<td>61.60</td>
<td>52.50</td>
<td>-20.0 -25.5</td>
<td>kNN</td>
<td>co-citation</td>
<td>68.51</td>
<td>75.60</td>
<td>36.9 51.1</td>
</tr>
<tr>
<td></td>
<td>bib-coupling</td>
<td>83.20</td>
<td>78.20</td>
<td>8.1 10.9</td>
<td>bib-coupling</td>
<td>22.09</td>
<td>5.39</td>
<td>-55.8 -87.9</td>
<td></td>
</tr>
<tr>
<td>Amater</td>
<td>84.45</td>
<td>79.44</td>
<td>9.7 13.5</td>
<td>Amater</td>
<td>60.56</td>
<td>79.24</td>
<td>37.0 62.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cosine</td>
<td>76.38</td>
<td>76.57</td>
<td>-1.24 32.2</td>
<td>Cosine</td>
<td>50.03</td>
<td>44.50</td>
<td>-27.2 55.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>co-citation</td>
<td>59.88</td>
<td>49.53</td>
<td>-24.9 -32.2</td>
<td>SVM</td>
<td>co-citation</td>
<td>68.91</td>
<td>76.9</td>
<td>27.2 55.7</td>
</tr>
<tr>
<td></td>
<td>bib-coupling</td>
<td>89.72</td>
<td>74.99</td>
<td>2.9 12.0</td>
<td>bib-coupling</td>
<td>24.06</td>
<td>6.40</td>
<td>-86.0 -87.9</td>
<td></td>
</tr>
<tr>
<td>Amater</td>
<td>83.08</td>
<td>77.08</td>
<td>5.03 4.5</td>
<td>Amater</td>
<td>66.60</td>
<td>74.8</td>
<td>25.6 55.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TF-IDF</td>
<td>78.49</td>
<td>76.77</td>
<td>-</td>
<td>TF-IDF</td>
<td>54.18</td>
<td>49.38</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 11 presents the results of the comparison between kNN and Support Vector Machines. The left table is for ACM8 collection and the right for Cade12 collection.

The general conclusion of the authors is that the best measure it should be used for this kind of experiments can be different and it is based on the data-set characteristics. A discovery was that for web links collections, measures based on co-citation perform better. On the contrary, digital libraries that contain scientific papers, bibliographic coupling estimations have shown that are more appropriate.

Furthermore, in the paper *To Classify Opinion of Different Domain Using Machine Learning Techniques* (Daiyan, M., Khan, A. & Alam, A., 2013), we can see a presentation of applying classification techniques to an opinion-mining problem. The authors define as opinion mining as the task to discover if an opinion at an article is constructive or unconstructive. The data set that has been used is by the merchant site epic. An analysis of the dataset is shown at Figure 12. In this research the algorithms that are being compared are Naïve Bayes and SVM.

As we can see at Figures 13 - 14, SVM produces better results than Naïve Bayes. Finally, this experiment provides really interesting results regarding the usage of Unigram, Bigram and Trigram as features for each algorithm.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Canon</th>
<th>Kodak</th>
<th>Nikon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review</td>
<td>1500</td>
<td>1200</td>
<td>1400</td>
</tr>
<tr>
<td>Sentence</td>
<td>4703</td>
<td>3834</td>
<td>4347</td>
</tr>
</tbody>
</table>
In conclusion, at this section we presented some examples of machine learning applications both in real life and from the scope of the scientific observation. We have chosen two examples from two big global incorporations and two scientific reports in order to show how scientific observation, experimentation and the appliance of those two inherits with the services that businesses have to offer.

2.6 Summarization

This chapter reports and analyses the contents or frameworks that could be applied on solving a classification problem. Furthermore, it has been investigated the algorithms that could implement machine-learning systems. In addition, we have
attempted to show the business applicability of this fast-growing technology by real life examples. Automated procedures in text classification and big data processing are a market that should be further explored and has a lot of potential both for the scientific community and the enterprises.
3 Useful Techniques

3.1 Natural Language Processing (NLP)

According to Tolle and Chen (2000), “the object of Natural Language Processing is to make the computer a fluent user of ordinary (human) language”.

The rise of technology usage and the data that is gathered nowadays force us to use more clever techniques to retrieve information. Natural Language Processing can solve problems to every day tasks in human-computer interaction and change the accuracy and the ways that we manipulate data in a text formatting.

Furthermore, there are several methods and techniques that we can use and with the combination of techniques such as state machines, rule systems, logic, probabilistic models and vector space modes we can build up NLP/Machine Learning Systems. The above models are used in algorithms and we can distinguish the state space search algorithms and machine learning algorithms as the most important. (Jurafsky, D. & Martin, J., 2008).

3.2 Techniques for NLP Tasks

Table 6 – NLP Techniques

<table>
<thead>
<tr>
<th>Technique</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentence Splitting</td>
<td>How we Identifying sentence boundaries in a document.</td>
</tr>
<tr>
<td>Tokenization</td>
<td>How the documents are tokenized and tokens are re-corded or annotated, by word or phrase. This is important because many downstream components need the tokens to be clearly identified for analysis.</td>
</tr>
<tr>
<td>Part-of-Speech (pos) Tagging</td>
<td>What about the part of speech characteristics and the data annotation. How such components are assigning a pos tag to token pos information.</td>
</tr>
<tr>
<td>Stop word list</td>
<td>How stop word list will be taken, and which words are to consider as stop word in which domain.</td>
</tr>
<tr>
<td>Stemming</td>
<td>If we reduce the words to their stems, how it will affect the meaning of the documents.</td>
</tr>
</tbody>
</table>
Noisy Data
Which steps are required for the document to be clear from noisy data.

Word Sense
How we clarify the meaning of the word in the text, ambiguity problem.

Collocations
What about the compound and technical terms.

Syntax
How should make a syntactic or grammar analysis. What about data dependency, anaphoric problems.

Text Representation
Which will be more important for representation of the documents: Phrases, Word or Concept and Noun or adjective? And for this which techniques will be feasible to use.

Domain and data understanding for Ontology
How to define the area, data availability and its relation for ontology construction.

Regular Expressions
Regular Expressions (RE) is a standardized methodology to describe text sequences.

Table 6 represents NLP techniques that can be used in comparison with text classification techniques (e.g. as features) and it has been retrieved by (Khan A., Baharudin B., Lee L.H. & Khan K., 2010).

3.2.1 Regular Expressions
RE languages have been used to search text in Unix-like systems (for example see emacs system and programming editors). Additionally, we can find an extended usage of RE in web search engines. Furthermore, this notation technique owns a very important role in text and word processing field and is a powerful tool when we are referring in computer and linguistics sciences.

We need a pattern and a text corpus in order to make the RE search work. Since we make a call to a regular expression search function, it will iterate the corpus and return all the results that match the pattern we have specified (Jurafsky, D., Martin, J., 2008).

The simplest RE pattern is a coherence of characters. For instance if we want to find the word *sofa* in a text we would have to type /sofa/. The slashes are not part of the RE language but they are used to programming languages like Perl and Python (Jurafsky, D., Martin, J., 2008).

In conclusion, RE is a really powerful technique for pattern recognition. Since we are able to match any kind of pattern in a text we have the ability to remove, omit
and search an enormous text corpus. On the other hand RE have various disadvantages. An important one is the maintainability. For example if we have a really large data set with text and create a pattern to retrieve information it would be really difficult to include all the possible combinations. Furthermore, It could be really difficult to update the regular expressions in order to match modern text styles or to cover different needs regarding the information retrieval out of a text corpus. A balanced way to use RE is to combine them with various NLP techniques that will maximize the success rate to our system or experiment. Additionally, an ideal usage of RE would be for small tasks and simplified patterns.

3.2.2 Stemming Words

We can define stemming as the technique that it is used for removing affixes of a word (Perkins, 2010). Furthermore, with word stemming we can define the root of the word. A common example is the –ing ending in words. As a result the stem of the word “cooking” is “cook”. Word stemming is a powerful technique in terms of optimization in web searching engines.

The most well known algorithm for the word stemming is the Porter algorithm (Porter, 1980). In Equation 8 can been found an example on how Porter algorithm operates.

Equation 8 – Example of Porter Algorithm

\[(m>0) \cdot \text{FULNESS} \rightarrow \text{FUL} (2.0)\]

In this example (2.0) the stem should not have a non-zero measurement (m) and only if this inequality is satisfied, the suffix *FULNESS could be replaced by the suffix *FUL (Willett, 2006).

Word stemming is an efficient technique to match sequences of words with the same root. It could help to have more efficient iterations of a text corpus. This happens because we only have to iterate the lemma of the word and not the whole word. For example if we searching a text for all the words like “programming”, “programmer” or “program” we could search only for the character sequence “program”. On the other hand, sometimes we could omit using word stemming for the
purposes of our experiment or application. It has to do mostly with the purpose of our of our system’s task.

### 3.2.3 N-Grams

We could define as an N-Gram the subsequence of N elements of a known sequence. By simplifying the above, we could describe N-Gram as a frame with length N that moves over the text. The ingredients of this frame are the N-Gram (Al-Shalabi, R., Obeidat, O., 2008).

<table>
<thead>
<tr>
<th>N-Gram</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unigram</td>
<td>1</td>
</tr>
<tr>
<td>Bigram</td>
<td>2</td>
</tr>
<tr>
<td>Trigram</td>
<td>3</td>
</tr>
<tr>
<td>Quadrigram</td>
<td>4</td>
</tr>
<tr>
<td>N-Gram</td>
<td>n</td>
</tr>
</tbody>
</table>

Table 7 – N-Grams

Table 7 shows the names of the N-Grams regarding their length.

As we stated above, the goal in N-Grams is to calculate the probability of a word w (sequence) given some history h (subsequence). To present an example we assume that h is the string “i like rock music” and we want to calculate the probability that the next word would be very (2.1).

Equation 9 – Example of probability calculation of a sequence given the subsequence

\[ P(very \mid i \ like \ rock \ music) \] (2.1)

A naïve computation of this probability is to calculate the relative frequently counts of the token “i like rock music” in a text and the occurrences of the times that “very” follows this token in the text (2.2).

Equation 10 – Example of probability calculation with counts
\[ P(\text{very}|\text{i like rock music}) = \frac{C(\text{i like rock music very})}{C(\text{i like rock music})} \] (2.2)

The above computation (2.2) is not the most efficient since we would never have enough data to model the creativity of a natural language usage, even if we have an enormous data source such us the worldwide web. At table 7 we are referring also to bigrams, trigrams and so on. We can form a bigram by calculate the probability of the targeting word \( w \) by knowing the previous word \( w_{-1} \). For example by changing (2.2) into a bigram we could have the following result: \( P(\text{very}|\text{music}) \).

We can refer to the assumption that a probability of a word is depended on the previous word as a Markov assumption. By applying a Markov model we can predict the probability, given a word \( w \), of the following word \( w_{+1} \). We can generalize that model and apply it to a whole sentence by brake it down into bigrams. As a conclusion, trigram looks into the two previous words and so on (Jurafsky, D., Martin, J., 2008).

Concluding, N-Grams is a technique that it can be applied to many NLP tasks or to data sets. The main advantage is that we can form easily language models and extract features from a text corpus. That could be achieved by counting the frequency of appearances of an N-Gram to a text or word sequence. As a result, we have the ability to create a generalized model that it can be used to machine learning tasks, compute statistics and in general analyse a text corpus. On the contrary N-Grams have the disadvantage that they are based on assumptions. That means that there is the probability that the generalised model cannot be applied to our task or it could produce misleading conclusions.
4 Technology Review

The purpose of this chapter is to list programming languages, packages, tools and programming environments that help the data scientist to apply natural language processing and classification techniques.

4.1 Programming Languages

We can use many programming languages to compute statistics and solve classification problems. Here we emphasise into Python (Python Programming Language) and Java (Oracle). There are other languages that we can use to apply classification techniques like R (R). We can find an extended description for packages and interfaces that could be used with R in order to apply classification techniques at the official website of the R programming language (R).

4.2 Useful Packages, Distributions and Working Environments

4.2.1 Python

Python it is a free cross platform high-level programming language that is free for use even for commercial projects because it has an OSI-approved open source licence (Python Programming Language). In addition, supports fully object-oriented programming style.

In python the main libraries for our purpose are numpy(NumPy), matplotlib (Matplotlib), nltk(NLTK 2.0) and scikit-learn(Scikit-Learn). All the above libraries are free and open source. Regarding the version of python, it is preferable to use python version 2x and not python 3x. That is because nltk is in the alpha version for python 3x.

The best distribution for scientific computing in Python is anaconda (Continuum Analytics). Anaconda distribution is a package management system that
enables us to have different versions of python interpreters such as version 2.7.5 and 3x or IPython (IPython). IPython is a powerful interpreter with really helpful assets like high performance in scientific plots and auto completion ability.

Finally, a really efficient Intergraded Development Environment (IDE) for our purpose is spyder (Spyder). Spyder provides a lot of features with the most important:

- Loads all the necessary libraries like numpy and matplotlib for us when we first run the IDE.
- Provides an interactive editor and we are able to run our scripts to different interpreters in order to test and debug our code.

An in-detail explanation of the above scientific modules can be found in the below bibliography (Bird, S., & Klein, E., & Loper, E., 2009), (Perkins, J., 2010) and (Richert W. & Coelho P.L., 2013).

4.2.2 Java

Java is also a cross platform high-level programming language. It is free to use even commercially at its standard edition but it depends on the nature of the software so the computer scientists must pay attention to the license of use before a public release. Java now is at the version 7. Moreover, it is an efficient and powerful programming language with strong Object – Oriented support.

The most well-known and powerful IDE for java is Eclipse (Eclipse). It contains features like auto completion, packaging system, text editor and terminal.

For the purpose of this research a really good and open source complete package is weka 3 (Weka: The University of Waikato). It is released under the GNU General Public License. Weka provides java libraries that enable us to implement solutions for classification and in general machine learning tasks.

A great asset of the Weka library is that it comes with a minimal and powerful Graphical User Interface (GUI) that we can load a dataset, enter parameters and try different algorithms and plot the results of our experiments.
The best ways to explore this piece of software are the official documentation that it is provided at the weka’s website and a book that it is written by the creators of the library.

Citation and links to the above documents can be found at (Weka) and (Witten, Frank & Hall).

4.3 Conclusion

Both of the programming languages have their own advantages. In the case of our experiments we will use the python programming language. Also the anaconda distribution and Spyder IDE with IPython interpreter is going to be our scientific programming environment. We could have taken advantage the GUI version of weka but our purpose is to implement different features in order to train the Naïve Bayes and the Maximum Entropy classifier. The easy syntax of python can help us a lot to the implementation procedure in comparison with Java. Although weka library provides programming interfaces to implement faster NLP or Machine learning tasks, the fact that python can change its programming style from procedural to Object-Oriented, gives us a boost to try and implement chunks of our experiments faster.
5 The Application

5.1 The scope of the application

A problem that can be occurred for an online business like oDesk is when the job posts are not classified correctly. For example if an employer post a job for web full stack development and the necessary skills are Django (Django) and Mongo DB (Mongo DB). If the employer put the above example to the Design and Multimedia section then a web developer specialist might not see this job post. That means that oDesk looses both possible employees and employers out of their marketplace. If we take into account that in 2012 the total amount of jobs that have been posted is 1.5 million (oDesk), then we can see how a problem like this could be scaled and cause money and customer base losses. Additionally, reduces the usability for the customers, since in a possible scenario like the above the employer cannot find a freelancer to hire and a freelancer cannot find a job that it is based in his skills.

In conclusion, the scope of this application is to create a machine learning system that will predict the correct job category based on the title, description and skills of the job post. This dissertation attempts to help oDesk to evaluate the applicability of NLP and Machine Learning technology to improve the job posting section of their online business.

5.2 Data Set

The contact with oDesk has been made through Panagiotis Papadimitriou (PhD). Dr. Papadimitriou who is Principal Data Scientist/Architect at oDesk, has provided us a dataset from oDesk database that includes 10000 rows of job posts. The data schema of the data set is described below:

openings: {title, description, skills, category, subcategory, job_type, workload, duration, client_country}
This is going to be the working set to apply NLP techniques, extract features, train and test the algorithms for the machine learning processes.

Each job post (row) holds information about following:
Title: This attribute holds a title description about the job. It is raw text field and we can see some ‘title’ examples in the Figure 15:

Figure 15 – Examples of the ‘title’ attribute

'RElated project styling needs conversion to Bootstrap'
'Digital forensic software - sales manager'
'jQuery/ASP.NET MVC 4 EXPERT for WebConference SU'
'freakyfactions.com web design'

Description: This attribute holds the actual job descriptions. It is also a raw text field and some of examples of this attribute are presented in Figure 16.

Figure 16 – Examples of the ‘description’ attribute.

Out[63]: 'We are looking for a writer to perform research and write short, terse summaries of business related topics. The writing is extremely factual and of a similar nature to the pages you will find on Wikipedia. 

Out[64]: 'Copy paste picture into spreadsheet.'

Skills: This attribute contains words that represent skills relevant to the job post. The format is also raw text but it is presented most of the time as words separated with commas (Figure 17).

Figure 17 – Example of the ‘skills’ attribute

Out[66]: 'design,minecraft,management'

Moreover as we can see from the data schema we have the following attributes like subcategory, job_type, workload, duration, client_country but we are not going to use them in the experiments that will follow.

The data format that we would use is Comma Separated Values (CSV). Furthermore, we have loaded and analyzed the dataset with the Pandas (Pandas) library for python programming language (Python Programming Language). Additionally, the ipython (IPython) interpreter would be used.

The results after loading the data set into a pandas Data Frame have shown that the following table’s objects have null values (Figure 18):

Title -> 1 null value
Skills -> 1246 null values

Figure 18 - Data Frame of the Data Set

```python
Out[2]:
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10000 entries, 0 to 9999
Data columns (total 9 columns):
title            9999    non-null values
description      10000  non-null values
skills           8754    non-null values
category         10000  non-null values
subcategory      10000  non-null values
job_type         10000  non-null values
workload         10000  non-null values
duration         10000  non-null values
client_country   10000  non-null values
dtypes: object(9)
```

Moreover, the pyplot module from matplotlib (Matplotlib) library has been used to compute how much percentage has each category to the total amount of the sample (Figure 19 - 20).

Figure 19 - Percentage of each category to the total amount of job posts.
Notation for the Figure 19:

wd: ‘Web Development’
sd: ‘Software Development’
nis: ‘Networking & Information Systems’
wt: ‘Writing & Translation’
adminsup: ‘Administrative Support’
dm: ‘Design & Multimedia’
cs: ‘Customer Service’
sm: ‘Sales & Marketing’
bs: ‘Business Services’

Figure 20 - Pie Chart of the Categories

Figure 20 was generated with the machine data analytics tool Splunk (Splunk).

The dataset contain nine classes and an important point to report in this preprocessing data analysis is that the graphs in Figure 16-17 show us that in our data set we have values for all the categories. Hence, that means that we would be able to use training and test data from all the categories in our experiments.
5.3 Data Preprocessing

Data preprocessing is a challenging and important stage on solving classification problems. The results and success of the experiments are highly dependent on the work we would at this stage.

As mentioned above we have used the Pandas (Pandas) python (Python) library to load oDesk’s data. For this case study we have selected as training attributes the ‘title’, ‘description’ and ‘skills’. The “title” and “skills” attributes have null values and we take this into account.

Moreover, since we have the data loaded there is a need for functions that will return the “title”, “description” and “skills” of a job post regarding their category. Those three necessary functions have been implemented and return an array with raw data of each of three parameters based on a pre-specified category (Figure 21).

Figure 21 – Function that returns the description for a specific category

```python
def getDesPerCategory(category):
    category -> string
    returns array with strings
    textHolder = []
    for item in description:
        if odesk.category[1] == category:
            textHolder.append(item)
    return textHolder[0:]
```

5.3.1 Text / NLP Processing

At this stage it is necessary to apply text/NLP processing techniques in order to form the data in a way that enable us to extract features and create our training and testing sets.

We begin the data processing with the ‘description’ attribute. As we saw at the Figure 19, ‘description’ does not contain any null values and its length is 10.000 rows.

We have from the previous stage all the columns that are in our experiments scope as array data structure. As an example at this point in order to show how NLP
techniques are going to be applied to the dataset, we would use the “description” attribute and “Software Development” as a job category.

Furthermore, the first technique that we will apply is to make all the characters lowercase. This is really important because we avoid from the very beginning mismatching of characters and we start normalizing the raw text. It has been used the lower function (Python 2.7.6 Documentation) of python (Python Programming Language) which is a build-in one for the string sequence type.

The next necessity is the removal of the whole punctuation characters out of the raw text. This technique is really important to be applied before we go to the next stage (Figures 22 - 23). It has been used again a python’s built-in function named strip (Python 2.7.6 Documentation).

(Figures 22 - Figure 23) - Before and after remove punctuation technique has been applied

Figure 22 - Before

develop a slot machines software (but with a special “edge” which will be discussed once we contract you), for mobile phones (must work on all known platforms including blackberry new and old, android, ios, windows mobile, etc). the game is a real money game so it has to have payment processing. \nneeds to be easy to add languages to, and initially will start in english, spanish, portuguese. \nthanks,"

Figure 23 - After

develop a slot machines software but with a special edge which will be discussed once we contract you for mobile phones must work on all known platforms including blackberry new and old android for windows mobile etc the game is a real money game so it has to have payment processing needs to be easy to add languages to and initially will start in english spanish portuguese thanks"

Now we are at the stage that we need to tokenize each job post into single words. Since we have applied the above NLP techniques we should not have any implications and we could move to the next stage of our experiments, which is the feature extraction. The tokenizer that it has been used is the word_tokenize from the tokenize module in NLTK (NLTK 2.0) library. Word tokenization is a necessary technique for our experiments because in that way we can process, the description in this case, as individual words and extract features like tags, bigrams or N-Grams and apply further NLP techniques in order to achieve a better text normalization (Figure 24).
Another important technique that we would apply to the dataset is the removal of the stop words that exist to each ‘description’. We have implemented this technique with the nltk’s (NLTK 2.0) stopwords English list (Figure 25). When we refer to stopwords, we mean words like “he”, “is”, “at”, “which” or “on”. The predefined list that we have used from the nltk library includes 127 English stopwords.

Figure 24 - Example of tokenized data

'have',
'payment',
'processing',
'needs',
'to',
'be',
'easy',
'to',
'ad',
'languages',
'to',
'and',
'initially',
'will',
'start',
'in',
'english',
'spanish',
'portuguese'

Figure 25 - Example of stopwords removal

['must',
'minecraft',
'skill',
'job',
'design',
'website',
'freakyfact',
'con',
'minecraft',
'theme',
'page',
'includ',
'sign',
'with',
'user',
'name',
'email',
'databases',
'must',
'know',
'minecraft',
'game',
'server',
'manag'],
Moreover, the preprocessing procedure showed us that in the text of the descriptions there are digits. In addition to the above techniques we have implemented a function that will remove the numbers from the dataset. At a conceptual level this function seems to be easy. By examining further the dataset we have noticed that what we want is to remove the numbers and digits that appear on their own. For example, if we have the string “4 the application we need”, although here the number 4 is a stopword, it will not be removed when we apply the ‘removeStopwords’ function. On the other hand, if we have the word “wait” in the format of w8, when we apply the “removeNumbers” function the number 8 will not be removed. Actually that is really efficient for our purpose. The string “w8” would participate into a bigram or unigram but it will be used as a feature only if it has high frequency and it pairs with other informative words (Figure 26).

The implementation challenges in order to apply the above techniques to oDesk’s dataset were mainly two. The first one is that we had to deal with raw unstructured text. That means there were a strong need for organizing the order on how to apply them.

In addition, the other challenge we had to deal with is the large amount of data. The implementation approach that it has been followed is the usage of iterations throughout the data structures (in this case arrays) that were holding each time the data. It is worthwhile to report that the performance of these procedures were low. Since we have obtained the results we needed for the experiments, it has not been taken any further action for performance improvement.

Figure 26 – Remove Numbers Function

```python
def removeNumber(array):
    new_data = []
    for item in array:
        if item.isdigit()==False:
            new_data.append(item)
    return new_data
```

5.3.2 Features extraction

For the purposes of this experiment it would be used bigrams of words as the main feature.
It has been implemented a function in order to extract bigrams from our dataset. Additionally, in this function we have used the above NLP techniques. Word stemming has been also applied by using the Porter algorithm (Porter). It is assumed that by applying the stemming technique the experiment’s results will be more efficient.

Specifically, the “makeBigrams” function takes raw text as input and process the raw text with NLP techniques with the following order. Firstly, the text is tokenized into words. Then we remove all the single numbers from the tokens. Furthermore, we apply the ‘remove punctuation’ function. Additionally, we iterate the tokens set and we create a new set where we apply word stemming for every word that does not belong to the stopwords. Finally, we have used the bigrams module from the nltk (NLTK 2.0) library to from the bigrams (Figure 27).

Figure 27 – Bigram Formation function

```
def makeBigrams(text):
    stemmer = PorterStemmer()
    tokenizer = WordPunctTokenizer()
    tokens = tokenizer.tokenize(text)
    tokens = removeNumber(tokens)
    tokens = removePuncLargeSet(tokens)
    result = [stemmer.stem(x.lower()) for x in tokens if not x in stopwords.words('english') and len(x) > 1]
    bi = bigrams(result)
    return bi
```

Our target is to create a list with features for a specific job category. We are using the ‘description’ attribute out of our dataset and we already have a processed array that holds all the descriptions. Since this function takes as input parameter raw text, a function that forms the description fields into documents have been implemented and we have applied it in order to form the bigrams. In order to achieve the creation of the features list we iterate the job descriptions for each category and we apply the ‘makeBigrams’ function.

We then need to compute the frequency distribution of each bigram. We need that metric to choose the most important and informative features for each category. A function has been implemented in order to have this metric. This function takes the bigrams input, uses the “FreqDist” module from nltk (NLTK 2.0) and returns a data dictionary that the bigrams as keys and the frequencies distributions as values (Figure 28).

Figure 28 - Frequency Distribution Function
Since we have computed the frequencies Distributions for each set of bigrams for a specific job category there is a need to sort the dictionary. In order to achieve this, it has been used the built-in python’s (Python) function “sorted” (Figure 29).

Figure 29 – Function to sort the Frequency Distribution Data Dictionary

We have applied all the above to all the job categories separately. An example of what ‘sortFreq’ function returns will help us to understand more things about the dataset and make the proper decisions before we go further on to the experiment (Figure 30).

Figure 30 – Example of Bigrams Frequencies Distributions

```
(("candid", "must"), 17),
(("look", "someone"), 17),
(("need", "someone"), 17),
(("part", "time"), 18),
(("you", "must"), 19),
(("long", "term"), 20),
(("you", "need"), 20),
(("we", "need"), 23),
(("internet", "connect"), 24),
(("call", "center"), 24),
(("phone", "call"), 24),
(("full", "time"), 30),
(("isrealfak", "cuntri"), 31),
(("cuntri", "isrealfak"), 31),
(("common", "skill"), 32),
(("custom", "support"), 44),
(("we", "look"), 43),
(("custom", "servic"), 66),
(("odesk", "comronyjon"), 100),
(("comronyjon", "odesk"), 100),
(("com", "ronyjon"), 139),
(("ronyjon", "odesk"), 139),
(("odesk", "com"), 141),
(("odeskcomputeristip", "odeskcomputertip"), 357)
```

In Figure 30 we can see each bigram accompanied with its frequency distribution for the category ‘Customer service’ job category. Normally, we use for our features list the most frequent bigrams. But this is not the case. Rajaraman and Ullman had referred to this phenomenon where we can have a word or pair of words with high frequency but it would not be useful for classification. On the other hand that does not mean that every word that it has low frequency would be useful (2011).
As we can see though at the above figure, the most frequent bigrams are ('custom', 'servic'), ('odesk', 'comronyjoni'), ('comronyjoni', 'odesk'), ('com', 'ronyjon'), ('ronyjon', 'odesk'), ('odesk', 'com'), ('odeskcomtouristip', 'odeskcomtouristip') with values that varies from 86 up to 357. What we can notice is that those bigrams are not useful for our feature list because they could belong to all the categories. We have already analyzed the dataset but this is the point that we have to make decisions about what is going to fill the features list and how are we going to mine it.

It has been found two ways in order to solve the above problem. The first way is to find all those bigrams with high frequency and not importance and create a list like the one we have used for the stopwords removal. This solution seems to be the most efficient because by following this path we will be certain that we have bigrams of words with high importance for the class description. Secondly, another possible solution could be to manually decide by our experience which bigram describes best the class. The second way it is not the best solution because we might omit useful bigrams and most importantly it will be difficult to improve the feature list.

For our purpose though we will use the second route. The first way even if it is more efficient, it is time consuming and regarding the time limitation of this research it would not be applied. Additionally, another reason for rejecting the first possible solution is that in order to automate the procedure of removing the low-informative features, pattern recognition knowledge and techniques are needed. Since that knowledge has not been acquired prior to the experimentation phase, we would have also to extract manually the pairs that form a pattern.

Furthermore, we have created bigrams for all the job categories. We can see the results of this process in the following table:

<table>
<thead>
<tr>
<th>Category</th>
<th>Number of Bigrams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Development</td>
<td>51717</td>
</tr>
<tr>
<td>Web Development</td>
<td>151766</td>
</tr>
<tr>
<td>Networking &amp; Information Systems</td>
<td>12685</td>
</tr>
<tr>
<td>Writing &amp; Translation</td>
<td>64016</td>
</tr>
<tr>
<td>Administrative Support</td>
<td>98885</td>
</tr>
<tr>
<td>Design &amp; Multimedia</td>
<td>50165</td>
</tr>
<tr>
<td>Customer Service</td>
<td>16752</td>
</tr>
</tbody>
</table>
We have chosen manually ten bigrams of each job category to create our features list. The bigrams that we have chosen are represented in the following table:

Table 9 – The Chosen Bigrams for the Feature List

<table>
<thead>
<tr>
<th>Category</th>
<th>Chosen Bigrams</th>
</tr>
</thead>
</table>
These are all the bigrams that will form the feature list. The next step that has been followed was to form a dataset that it would contain the description of each row of the dataset accompanied with its job category. We processed this data set with the same NLP techniques that have been applied in order to create the features list. A similar function with the one we have seen in Figure 27 have been implemented in order to return again a new data set but this time with the each row would be formed into bigrams accompanied with the job category.

Further on, we implemented a function that will extract features out of the new data set that contains the bigrams with their corresponding label. This ‘extract_features’ function takes as input the processed dataset and the feature list. It forms the processed dataset into a set data structure and it has been used the build-in python (Python) function. Then the function iterates the features list and assign to a data dictionary every pair of words that it matches with the set. Finally it returns the features list items with a Boolean value. It assigns “True” if the bigram of the feature list has been found and “False” if it is not.

In conclusion, this is the description of the feature extraction function that has been implemented to produce our features and it is a really important state of our experiment. The wrong choice of features can cause a lot of misleading results and really low performance.

5.4 Preparation of the training and test sets – Experimentation with Classifiers

The scope of this research is to emphasize supervised learning processes in order to solve a real-life problem. This approach has been chosen because the classes are known and by definition this is a supervised classification problem. By reviewing further *Pattern Recognition and Classification* (Dougherty, 2013) we can, in a general scope claim that the supervised processes could be characterised as empirical. More specifically, a set of features (or rules) is learned by a training data set (or instances) and a supervised learning algorithm is chosen to apply the features to a new (unknown) instance. Moreover, the choice of the learning algorithm is an important and critical decision. Before adopting an algorithm for a general use in our
machine learning system we should evaluate its behavior. The most common measurement for this purpose, is the prediction accuracy (p.a.). The p.a. is measured by dividing the percentage of the positive predictions with the total number of predictions (3.0).

Equation 11 – Prediction Accuracy Metric

\[
Prediction \text{ Accuracy} = \frac{\text{Percentage of Positive Predictions}}{\text{Total Amount of Predictions}} \quad (3.0)
\]

We will report three techniques for p.a. for a chosen algorithm:

1. We can split our training set. The two-thirds will be used as training data and the rest one-third will be used for performance purposes.

2. We can apply a technique that is known as cross-validation. The procedure is the following: We split the data set into individual equal subsets.

3. Finally, in the need of an accurate classifier's error rate we can apply leave-one-out validation where all test subsets consisting of a new instance (Dougherty, 2013).

Figure 31 – The process of supervised learning (Dougherty, 2013)

Figure 31 describes the process of supervised learning. We can also refer to this process as a useful and neat methodology, in order to build a machine learning system.
The above techniques and the Figure 31 explain important metrics for our experiments in order to analyze and compare in depth the algorithms. As stated it will be used the prediction accuracy rate in the first attempt of building this machine learning application. In addition to this in the next experiments will be used the cross validation technique since we would use all the classes out of the dataset and more features. The algorithms that we have chosen for the experiments are Naïve Bayes and Maximum Entropy.

In order to split the dataset to train data and test data we need to create features set. It has been applied the “extract features” function in order to achieve this. More specifically, we took advantage of the easy and powerful syntax of the python (Python) language (Figure 32).

Figure 32 – Create Feature Set

```python
features_set = [(extract_features(item, sumlist), sense) for item, sense in bigramedata]
```

As we can see from the Figure 32 we store an array where inside we iterate the “bigramedata”, which holds the description attribute formed into bigrams and the corresponding job categories. For each item and category of the “bigramedata” we extract features and their categories.

Now that the features data set is ready we are able to create the training and test set. The train set is the one that we will train the classifier. Moreover, since we would have a trained classifier we will be able to provide the accuracy metric for both the train and test set.

Furthermore, the cross validation technique has been chosen to calculate the prediction accuracy of the classifiers. The strategy it has been done is to split the features set into 10 equal portions. As mentioned previously oDesk’s (oDesk) data set has 10.000 rows. Hence, the features set has the same length. The ‘len’ build in python (Python) function has been used to calculate the lengths of the two arrays and in order to cross check the above assumption (Figure 33).

Figure 33 – Calculation of the description and features Set arrays.

```
len(description)
10000

len(features_set)
10000
```
In order to apply the cross validation technique it has been split the features data set into 10 smaller data sets with 1000 features each (Figure 34). With these smaller sets we create 10 different training and testing data sets Figures 35-36.

**Figure 34 – Data Split of the Features set**

```python
f1 = features_set[:1000]
f2 = features_set[1000:2000]
f3 = features_set[2000:3000]
f4 = features_set[3000:4000]
f5 = features_set[4000:5000]
f6 = features_set[5000:6000]
f7 = features_set[6000:7000]
f8 = features_set[7000:8000]
f9 = features_set[8000:9000]
f10 = features_set[9000:]
```

**Figure 35 – Train sets**

```python
train_set1 = f1 + f2 + f3 + f4 + f5 + f6 + f7 + f8 + f9
train_set2 = f1 + f10 + f3 + f4 + f5 + f6 + f7 + f8 + f9
train_set3 = f1 + f2 + f3 + f4 + f5 + f6 + f7 + f8 + f9
train_set4 = f10 + f2 + f3 + f4 + f5 + f6 + f7 + f8 + f9
train_set5 = f1 + f2 + f3 + f10 + f5 + f6 + f7 + f8 + f9
train_set6 = f2 + f3 + f4 + f10 + f6 + f7 + f8 + f9
train_set7 = f5 + f2 + f3 + f4 + f5 + f10 + f7 + f8 + f9
train_set8 = f1 + f2 + f3 + f4 + f5 + f6 + f10 + f8 + f9
train_set9 = f1 + f2 + f3 + f4 + f5 + f6 + f7 + f10 + f9
train_set10 = f1 + f2 + f10 + f4 + f5 + f6 + f7 + f8 + f9
```

**Figure 36 – Test Sets**

```python
test_set1 = f10
test_set2 = f2
test_set3 = f9
test_set4 = f1
test_set5 = f4
test_set6 = f5
test_set7 = f6
test_set8 = f7
test_set9 = f8
test_set10 = f3
```

The next stage is to feed the classifiers and calculate the prediction accuracy for both the train and test sets. Our experiment will start with the Naïve Bayes algorithm. It has been applied the Naïve Bayes Classifier implementation from the nltk (NLTK 2.0) library. The experiment started by training the classifier with the ‘train_set1’. Next step was to calculate the prediction accuracy of the train and test set. This metric has been calculated by using the accuracy function of the nltk library. We feed the accuracy function with the trained classifier and the corresponding train set and the function returns the accuracy percentage. Additionally, we apply the same function
and we use the train set instead of the train one. We have run this process ten times and the results are presented in the table 10.

<table>
<thead>
<tr>
<th>Experiment Number</th>
<th>Train Set Prediction Accuracy (%)</th>
<th>Test Set Prediction Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>38.54</td>
<td>39.10</td>
</tr>
<tr>
<td>2</td>
<td>38.42</td>
<td>38.94</td>
</tr>
<tr>
<td>3</td>
<td>38.52</td>
<td>38.50</td>
</tr>
<tr>
<td>4</td>
<td>38.53</td>
<td>38.40</td>
</tr>
<tr>
<td>5</td>
<td>38.37</td>
<td>40.00</td>
</tr>
<tr>
<td>6</td>
<td>38.71</td>
<td>37.00</td>
</tr>
<tr>
<td>7</td>
<td>38.69</td>
<td>37.30</td>
</tr>
<tr>
<td>8</td>
<td>38.53</td>
<td>38.40</td>
</tr>
<tr>
<td>9</td>
<td>38.70</td>
<td>37.20</td>
</tr>
<tr>
<td>10</td>
<td>38.77</td>
<td>37.50</td>
</tr>
<tr>
<td>Average</td>
<td>38.58</td>
<td>38.23</td>
</tr>
</tbody>
</table>

The Naïve Bayes Classifier’s prediction accuracy is the average of all ten experiments. As we could notice from the Table 10 the prediction accuracy for the train set is almost 38.6%. The same metric for the test set is nearly 38.2%. Furthermore, the minimum value for the train set was found at the Experiment 5 and the maximum value was found at the Experiment 10. Additionally, The minimum prediction value for the test set was found in Experiment 6 and the maximum value in Experiment 5 (Table 10). Another observation that could be noticed is that the Experiment 5 holds the minimum prediction percentage for the training set and the highest for the test set.

Further on, the prediction accuracy of the Naïve Bayes classifier at all the above these series of experiments has a low percentage for both the train and test data. The positive result out of these experiments is that the train and test sets have accuracy percentages that are really close. The negative, which is the low overall percentages for the both sets, leads us to the assumption that the features we have used are poor on describing a class. On the other hand, the same features have
been applied to the Maximum Entropy classifier in order to report overall conclusions and make further comparison.

For the experiments with Maximum Entropy classifier we have used the implementation of the algorithm in the nltk (NLTK 2.0) library with the Improved Iterative Scaling (IIS) algorithm (Pietra et al., 1995). The results of the Maximum Entropy cross validation experiments are presented in the Table 11.

Table 11 – Prediction Accuracy Metric for the Maximum Entropy Algorithm

<table>
<thead>
<tr>
<th>Experiment Number</th>
<th>Train Set Prediction Accuracy (%)</th>
<th>Test Set Prediction Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25.25</td>
<td>27.80</td>
</tr>
<tr>
<td>2</td>
<td>25.60</td>
<td>24.94</td>
</tr>
<tr>
<td>3</td>
<td>25.49</td>
<td>25.40</td>
</tr>
<tr>
<td>4</td>
<td>25.51</td>
<td>25.20</td>
</tr>
<tr>
<td>5</td>
<td>25.34</td>
<td>26.90</td>
</tr>
<tr>
<td>6</td>
<td>25.50</td>
<td>25.30</td>
</tr>
<tr>
<td>7</td>
<td>25.67</td>
<td>23.60</td>
</tr>
<tr>
<td>8</td>
<td>25.40</td>
<td>26.30</td>
</tr>
<tr>
<td>9</td>
<td>25.50</td>
<td>25.30</td>
</tr>
<tr>
<td>10</td>
<td>25.57</td>
<td>24.60</td>
</tr>
<tr>
<td>Average</td>
<td>25.48</td>
<td>25.53</td>
</tr>
</tbody>
</table>

As we can see form the table 11 the precision accuracy for the Maximum entropy classifier is 25.49% for the training set and 25.53% for the testing set. The minimum accuracy value for the training set can be found in Experiment 1 and the maximum accuracy value can be found at the Experiment 7. Additionally, for the testing set the minimum prediction accuracy is reported at the Experiment 7 and the maximum at the Experiment 1. At this experiments we have a match between the pairs of minimum and the maximum values at the train and test sets.

The experimentation with Naïve Bayes and Maximum Entropy has shown us that Naïve Bayes performs better to this classification task. The difference in the training set accuracy is approximately 13% less for the Maximum Entropy. Additionally, for the testing set this difference is almost the same with a percentage 12.66%. On the other hand, another piece of information that we can retrieve from
the tables with the results of the experiments is that Maximum Entropy’s test set accuracy is bigger than the train set. This does not happen to the Naïve Bayes. This could be a high informative observation if the accuracy performance of the two classifiers was near with 1% or 2% difference between them. In addition, the fact that Naïve Bayes have performed better than the Maximum Entropy in the above experiments does not surprise us. At the literature review of this dissertation we have presented an application by Osborne M. (2002) where we have observed that the Naïve Bayes performs better than the Maximum Entropy when it has been chosen independent set of features. This is our case and the focus at this point is to investigate the low performance of both classifiers.

Moreover, the average accuracy percentages are low and by following the previous discussion, the root of these poor results has to do with the features selection. The importance of features selection has been mentioned to the previous section. Although many NLP techniques have been applied for data clearance and a standard process has been followed, we can see from those complete experiments the results are not satisfactory.

Additionally, it has been applied the nltk’s ‘show_most_informative_features_function’ for both of the classifiers. We have used the ‘train_set1’ for both of them and we return the ten most informative features. The results of the above approach are presented in the Table 12.

Table 12 – Most Informative Features For Maximum Entropy and Naïve Bayes Classifiers

<table>
<thead>
<tr>
<th>Naïve Bayes</th>
<th>Maximum Entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>('call', 'center')</td>
<td>('data', 'entri')</td>
</tr>
<tr>
<td>('custom', 'servic')</td>
<td>('excel', 'english')</td>
</tr>
<tr>
<td>('handl', 'custom')</td>
<td>('search', 'engin')</td>
</tr>
<tr>
<td>('data', 'entri')</td>
<td>('write', 'skill')</td>
</tr>
<tr>
<td>('window', 'server')</td>
<td>('graphic', 'design')</td>
</tr>
<tr>
<td>('servic', 'experi')</td>
<td>('facebook', 'twitter')</td>
</tr>
<tr>
<td>('logo', 'design')</td>
<td>('design', 'work')</td>
</tr>
<tr>
<td>('invest', 'fund')</td>
<td>('need', 'creat')</td>
</tr>
<tr>
<td>('system', 'administr')</td>
<td>('media', 'market')</td>
</tr>
<tr>
<td>('server', 'administr')</td>
<td>('mobil', 'app')</td>
</tr>
</tbody>
</table>
Table 12 shows us that the two classifiers have only one bigram in common the ('data', 'entri').

5.4.1 Process the title and the skills fields

Before we run we the classifiers we have processed the title and the skills attributes with the same techniques we have used to process the description attribute. The first assumption was that there would not be any further problem that we would have to deal with.

The problem that we had to deal at this stage is the null values in our dataset. The null values are stored as nan values. The approach we have taken is to use the built-in function “isnan” from the math module. This function returns “True” if it founds a “nan” value and “False” if it is not. At this stage, we notice that there similar logic with the process we have done for the digits removal. Hence, we followed the same logic for the implementation of a function that would remove all the “nan” values. In more detail, we return the tokens that exist in the skills but they are not “nan”. We get type error that a float is required Figure 25.

Figure 25 – Type Error

```python
In [11]: d = [item for item in skills if not math.isnan(item)]
Traceback (most recent call last):
  File "<ipython-input-11-234158fd0a34>", line 1, in <module>
    d = [item for item in skills if not math.isnan(item)]
TypeError: a float is required
```

At this point there is had not been found a possible solution for this error.

5.5 Summary

We have started this section by stating the scope of the application. Moreover, Odesk's dataset have been analysed. The next step that have been followed is the preprocessing and the processing of the dataset by applying all the necessary NLP techniques. Furthermore, it has been explained analytically the process that has been followed to select the features for the experimentation with the classifiers. The final step was to create the training and the testing sets in order to feed the
classifiers. Finally, it is presented an evaluation of the experimentation results and the attempt to process other attributes as inputs for the classifiers.
6 Conclusion

6.1 Summary

To begin with, this dissertation’s attempt was to research Natural Language Processing and Machine Learning techniques and apply them to solve a classification problem. Additionally, it has been done the research to the technologies that could be used in order to implement an experimental system that will be able to classify an input to N number of classes. It has been used a dataset provided by oDesk, a fast growing online job marketplace located in U.S.A.

In the case study with oDesk’s data set, it has been a challenge to build a system that could be able to classify with high accuracy a job post into the correct category. In order to be applicable and valuable such a system to a company, there is a strong need to be chosen the right technologies and techniques. This case study could be described as a competition between an intelligent system and the category choice a real person could do. Big data sets if manipulated properly could provide a lot of valuable information and could be exploited financially.

Although it has been explored a small aspect of an online business and we have seen the possibilities of these technologies. For example in our case study, if a client could be able to type a job description as raw text and the system could successfully “understand” which category this text belongs; the service is improved since the false classified job posts are eliminated. Further on, by trying to understand how oDesk operates and analyse the data that have been provided to us, we conclude that machine learning could be applied to many different sections of the enterprise. For instance, it could be implemented a recommendation service for the client. More specifically, when the customer types a job description apart from the automated classification of the post, it could have directly recommendations on who are the most suitable freelancers for the project. This is only an idea of a variety of possibilities that could be used by oDesk.

Furthermore, there are a lot online enterprises that we could note down common characteristics with oDesk. Some of the technical similarities are the use of
profiles, posts, ratings and recommendations. Additionally, there is a common need to improve these services, automate the procedures in order to be provided a better customer-user experience. Additionally, it has been reported in the literature review the importance of the Machine Learning techniques to online companies like Netflix (Netflix Inc.) and Amazon (Amazon Inc.). As stated before, it is highly important for these organizations to improve their recommendations to their users by building intelligent systems.

6.2 Achievements

This dissertation have achieved the following regarding the original aims:

1. Review of Machine Learning techniques for classification: It has been reviewed techniques and algorithms for solving classification problems. The review presents useful terminology explanation with examples. Then, dives into examples of these techniques from two big online businesses. Furthermore, it has been also presented four applications as comparison examples. At the two of them the emphasis was in Maximum Entropy and Naïve Bayes algorithms. The rest two applications present two more algorithms like Support Vector Machines and k Nearest Neighbour.

2. Review of Natural Language Processing Techniques: The third chapter introduces the most important and relevant with the topic Natural Language Processing techniques. It has been distinct as most important the Regular expressions, the Porter (Porter) algorithm for word stemming and Ngrams.

3. Technology Review: The fourth section of this dissertation derives a technology review, which reports programming languages, development packages and libraries as a mean to implement a machine learning system and apply Natural Language Processing techniques. Python (Python) has been chosen as the programming language of preference, because of its easy syntax and the larger variety of machine learning and NLP libraries that offers.

4. Machine Learning Application: We have started the application chapter by describing in depth the problem that it would be solved and the scope of the application. Moreover, it is provided step by step the methodology and the approach that have been used in order to:
   - Analyse the data set
o Pre-processing and processing the data with NLP techniques
o The features extraction process
o The preparation of the train and test datasets
o Evaluation of the results of experimentation with Naïve Bayes and Maximum Entropy algorithms

The system that it is delivered takes as an input one attribute of the dataset, the “description” one.

6.3 Non derivable

The aims and the objectives that are not delivered are:

- A machine learning system with three attributes as input. One of the original objectives of this dissertation, were to use as input the attributes ‘title’, “description” and “skills” of the oDesk’s dataset. Instead only “description” has been used.
- Experimentation with different algorithms. It has been followed the original recommendations from oDesk’s data scientist and the comparable algorithms that have been used are Maximum Entropy and Naïve Bayes. It has not been done further in depth research regarding the choice of the two algorithms.

6.4 Critical Analysis of Processes and the Results

This dissertation started by researching and introducing machine-learning techniques in order to solve a classification problem. During the research process it has been discovered several algorithms that could be applied for the implementation of a machine learning application. Furthermore, in our case study, text-data processing techniques had to be also explored because they are a necessary piece of an artificial intelligent system. After the research stage, a lot of crucial choices had to be made such as which algorithm is the best to solve the classification problem or which algorithm is easier to be implemented. It has been emphasised from the beginning to compare the Naïve Bayes and the Maximum Entropy ones. The reason for this choice was originally the suggestion that have been given by Dr. Papadimitriou (oDesk). Additionally, there has not been any prior knowledge of natural language processing and machine learning techniques. The learning curve on
these technologies was slow. A lot of time has been spent in order to understand the context of the dissertation’s topic and how those procedures could be applied.

Moreover, another crucial choice that had to be made was regarding the programming and implementation environment. It has been chosen Python (Python Programming Language) programming language. In addition, it is important to be reported that there were no prior knowledge of python and less than a year programming experience. The learning curve of the programming language syntax was fast and easy to follow. On the other hand, a lot of obstacles have been faced regarding which external libraries of the language will be applied. Nltk (NLTK 2.0) library was chosen in order to manipulate the data set with NLP techniques. This decision was really efficient because this library contains implementations of all the important NLP/text processing algorithms and processes. Further on, a difficult decision was which library will be used for classification. Both nltk and Sci-Kit Learn (Sci-Kit Learn) contains implementations of classification algorithms. After experimenting with them, we continued using the nltk package. The reason was that the features extraction and formation with nltk is in the form of a data dictionary (Python 2.7.6 Documentation), which as a data structure was more familiar and easier to manipulate. Sci-Kit Learn using vectors and numpy arrays (NumPy) where there was not solid knowledge up to this point to implement a solution. The negative side of the nltk choice was the performance of the overall processes. In the whole process, nltk was too slow to handle iterations in the data set we have been used. Spyder (Continuum Analytics) Software Development Kit (S.D.K.) is evaluated as an efficient choice. The most useful tools have been the autocompleting function and the possibility to load different python interpreters.

Moreover, the original methodology that has been followed changed in the middle of this dissertation. Firstly, the features extraction function did not contained a features set with bigrams that could be compared with the processed text and return a Boolean value whether the text contains or not the bigram. Additionally, in order to change the methodology regarding the features extraction, a lot of time has been spent. On the other hand, the progress it has been done so far with the appliance of NLP techniques helped positively on focusing mostly on the features extraction function and only few modifications have been made. In order to understand the importance of this, during the implementation cycle the most time that have been spent was about understanding, analysing and processing the dataset.
Finally, the results of the experiments are evaluated as poor. Firstly, the accuracy percentages for both classifiers are low. We conclude that these results are poor because of the features set selection. It has been stated in many sections of this research that the text processing and the features selection are the most important process prior to classification. The necessity for good understanding of the dataset before and during the pre-processing procedure has been underestimated throughout the implementation cycle. Hence, this underestimation is evaluated as the main reason for these low accuracy rates. On the other hand the positive side of the final results is that with Naïve Bayes classifier we have achieved an accuracy rate for both train and test set near 40%. Since in our case study we had nine classes, if we had a system that would be able to choose randomly a class the percentage of finding successfully the label would be nearly 11.11%.

6.5 Future Work

To begin with, a series of improvements can be made in the future regarding the experimentation and the methodology that has been used. These improvements could be achieved because further experience and knowledge have been acquired throughout the whole process of this dissertation. More specifically, there is a better understanding of Natural Language Processing and Machine Learning techniques. Additionally, the programming skills have been improved and there is a better understanding on how a machine learning system could be designed and implemented. A lot of time have been spent in order these skills to be acquired.

Moreover, if there was more available time for this dissertation the following achievements could have been made:

- Experimentation with different supervised algorithms.
- Automated features set selection.
- Usage of three attributes of the data set as inputs to the classifiers.
- Usage of dependent features in order to evaluate the behaviour of Maximum Entropy and Naïve Bayes in more depth.
- Calculation of additional metrics regarding the performance of the classifiers.
- Experiment with other technologies and libraries like Sci-Kit Learn.
- Appliance of Pattern Recognition techniques (for example Regular Expressions).
6.6 Recommendations

Machine Learning and more specifically classification problems are challenging processes. Many parameters have to be taken into account and if there is no prior knowledge of the topic it is really difficult to achieve an acceptable experimentation result. On the other hand, it is a fascinating scientific and technological field and as we have seen so far those techniques have a variety of appliance in the real world.

It is recommended that in order to understand the contents of a topic at this area the most efficient approach is to break down each single technique and spend a proper amount of time to learn in depth about it. For instance, there are plenty of NLP techniques that could be applied to a data set. Though the first thing someone should do is to analyse properly the data set. This analysis would show which NLP techniques should be applied. Towards to this, there is a strong need of programming literacy and the researchers on this field should improve their skills the same time. It is really important the theoretical and practical knowledge to be at an equal level. Finally, the most important aspects, regarding the machine-learning field, are the motivation, vision and passion to build systems that would be useful to the society and organizations. Systems that would improve people's everyday life.
7 References


MSc dissertation check list

Please insert this form, loose-leaf, into each copy of your dissertation submitted for marking.

<table>
<thead>
<tr>
<th>Milestones</th>
<th>Date of completion</th>
<th>Target deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal</td>
<td>15/6/2013</td>
<td>Week 3</td>
</tr>
<tr>
<td>Initial report</td>
<td>03/9/2013</td>
<td>Week 7</td>
</tr>
<tr>
<td>Full draft of the dissertation</td>
<td>3/12/2013</td>
<td>2 weeks before final deadline</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Learning outcome</th>
<th>The markers will assess</th>
<th>Pages</th>
<th>Hours spent</th>
</tr>
</thead>
</table>
| Learning outcome 1 | * Range of materials; list of references  
* The literature review/exposition/background information chapter | p.15-39, p. 67-70 | (for example, 200 hours) |
| Learning outcome 2 | * Evidence of project management (Gantt chart, diary, etc.)  
* Depending on the topic: chapters on design, implementation, methods, experiments, results, etc. | p.40-60 | (for example, 200 hours) |
| Learning outcome 3 | * Chapter on evaluation (assessing your outcomes against the project aims and objectives)  
* Discussion of your project’s output compared to the work of others. | p.61-66 | (for example, 200 hours) |
| Learning outcome 4 | * Is the dissertation well-written (academic writing style, grammatical), spell-checked, free of typos, neatly formatted.  
* Does the dissertation contain all relevant chapters, appendices, title and contents pages, etc.  
* Style and content of the dissertation. |       | (for example, 80 hours) |
| Learning outcome 5 | * Performance  
* Confirm authorship |       | 1 hour |

Have you uploaded your dissertation to Turnitin? Yes/No

Has your supervisor seen a full draft of the dissertation before submission? Yes/No

Has your supervisor said that you are ready to submit the dissertation? Yes/No

1 Please note the page numbers where evidence of meeting the learning outcome can be found in your dissertation.