Automated Essay Scoring

Scoring Essays in Swedish

André Smolentzov

Institutionen för lingvistik
Examensarbete 15 hp
Kandidatprogram i datalingvistik 180 hp
Vårterminen 2012
Handledare:Mats Wirén, Robert Östling

Stockholms universitet

Automated Essay Scoring

Scoring essays in Swedish

André Smolentzov

Abstract

Good writing skills are essential in the education system at all levels. However, the evaluation of essays is labor intensive and can entail a subjective bias. Automated Essay Scoring (AES) is a tool that may be able to save teacher time and provide more objective evaluations. There are several successful AES systems for essays in English that are used in large scale tests. Supervised machine learning algorithms are the core component in developing these systems.

In this project four AES systems were developed and evaluated. The AES systems were based on standard supervised machine learning software, i.e., LDAC, SVM with RBF kernel, polynomial kernel and Extremely Randomized Trees. The training data consisted of 1500 high school essays that had been scored by the students' teachers and blind raters. To evaluate the AES systems, the agreement between blind raters' scores and AES scores was compared to agreement between blind raters' and teacher scores. On average, the agreement between blind raters and the AES systems was better than between blind raters and teachers. The AES based on LDAC software had the best agreement with a quadratic weighted kappa value of 0.475. In comparison, the teachers and blind raters had a value of 0.391. However the AES results do not meet the required minimum agreement of a quadratic weighted kappa of 0.7 as defined by the US based nonprofit organization Educational Testing Services.

Sammanfattning

Jag har utvecklat och utvärderat fyra system för automatisk betygsättning av uppsatser (AES). LDAC, SVM med RBF kernel, SVM med Polynomial kernel och ”Extremely Randomized trees” som är standard klassificerarprogramvaror har använts som grunden för att bygga respektivt AES system.

Nyckelord/Keywords

Automated Essay Scoring, Swedish Essays, Training data, Scores, Supervised Machine Learning, Linear Discriminant Analysis, Support Vector Machines, Extremely Randomized Trees, Features, Feature ranking, cross validation, cross validation error, quadratic weighted kappa, generalization error, grid search

Contents

1 Introduction

Good writing skills are essential for succeeding in a modern society where written (and spoken) communication is paramount. Essay writing is important as a classroom activity as well as in different standardized tests. The evaluation of essays is labor intensive and therefore very expensive. There is also a strong subjective element in essay scoring and therefore it is difficult to use essay scores as objective evaluation criteria in standardized tests. Automated Essay Scoring (AES) may be helpful in solving these problems.

The department of Economics at Stockholm University has access to a sample of essays written in Swedish by high school students. The purpose of this project is to develop and to evaluate AES systems based on this sample of essays. The basic functions for an AES system are to select relevant features and to perform classification tasks to assign "scores" to essays. The classification task is based on supervised machine learning principles. In summary, the aims are to identify and evaluate which features are best fitted to assign scores to essays, to evaluate different supervised machine learning algorithms and to identify relevant measurements for the results.

2 Background

2.1 About essay scoring

Essay scores may be used for very different purposes. In some situations they are used to provide feedback for writing training in the classroom. In other situations they are used as one criterion for passing/failing a course or admission to higher education. Tests that are used for training are called low-stake tests, whereas tests that have serious consequences for the students are called high-stakes tests. Examples of high-stake tests are TOEFL (Test Of English as a Foreign Language is one criterion for admission of foreign students in many US universities) and "Högskoleprovet" (one criterion for admission to higher education in Sweden).

In many countries (including Sweden, Canada, US, etc.) school authorities have standardized tests in different subjects in order to both evaluate the students individually and, on a collective level, to evaluate the quality of education in different settings. Essay writing is a component in these standard tests. A requirement is that the scores must be valid indicators for the students' knowledge and the scores must be comparable among many different schools in different areas. In Sweden the school teachers evaluate their own students' essays and determine the score. In the US (and Canada) there are most often at least two blind raters involved in the scoring process in major standardized tests. If there is an inter-rater agreement the score is defined. If they disagree, a third rater is used to resolve the disagreement.

Depending upon the purpose of the essays, they may be evaluated in the following areas: 1) mechanics, 2) structure, 3) content and 4) style and the scores must reflect these areas. The mechanics represent the grammar and spelling requirements. Correct spelling and grammar are usually basic requirements in all essays. Structure refers to the logical presentation and development of ideas. An essay may be related to a specific subject and it must fulfill some content criteria. For example, an essay may be related to some area of cell structure in biology and the scores must show that the corresponding contents are covered. Some essays may have requirements on style. For example, an essay may be required to have an expository, narrative or argumentative style (Walker, 2009).

The human evaluation of the essays may be either holistic or analytical (or trait based). The holistic approach assigns scores based on the general characteristics of the essays. That is, a score is assigned

for the whole essay based on the rater's judgment. The analytical approach identifies different characteristics (traits) of the essays that are evaluated and contribute in different ways to the final score. The holistic approach takes less time and is therefore less expensive but it may be more subjective. The analytical method is more time consuming and more expensive but it tends to have grater inter-rater reliability (Nakamura, 2004).

The scores assigned to essays by human raters are intrinsically subjective (Traub, 1994). Human raters have different characteristics like age, training, mood, prejudices, social, ethnic backgrounds, and reaction to the handwritten style that may influence the way they assign scores. That is, there are always intra-rater and inter-rater variations. For example, the same person scoring the same essay at different times may assign different scores (intra-rater variation) depending on their mood or health. Different raters scoring the same essay may assign different scores (inter-rater variation). A study about essay scores in the Swedish standard test at high school level (Hinnerich, Höglin, & Johannesson, 2011) has shown that class teachers tend to assign higher scores to their own students when compared with blind raters. The teachers may be influenced by personal knowledge of different students (positive or negative bias) and the general pressure of the schools to have higher scores (as a competition factor).

Different tests may use different scales for scoring essays. Here are some examples: TOEFL IBT (Internet Based Test) (TOEFL IBT Test scores, 2013) uses a scale with 6 scores and our sample of essays uses a 4 scores scale. Other tests may use scales with more or less scores.

2.2 Automated Essay Scoring (AES)

Automated Essay Scoring is defined as the act of assigning scores to essays using an AES system. Throughout this document I use the acronym AES by itself when discussing general aspects of automated essay scoring. I use "AES system" when discussing specific aspects related to functions and their implementations. Figure 1 presents an overview of the functions of an AES in operation. It receives an essay in digital form as an input and it outputs a score. The AES system consists of software functions. There are functions to pre-process the essays into an internal format that is accepted by the other functions, there are functions to extract the required features from the essays and there are functions to perform the classification task to decide which score should be assigned to an essay. The classifier is based on supervised machine learning where the classes are the scores and each essay is represented by a set of features. (Dikli, 2006)

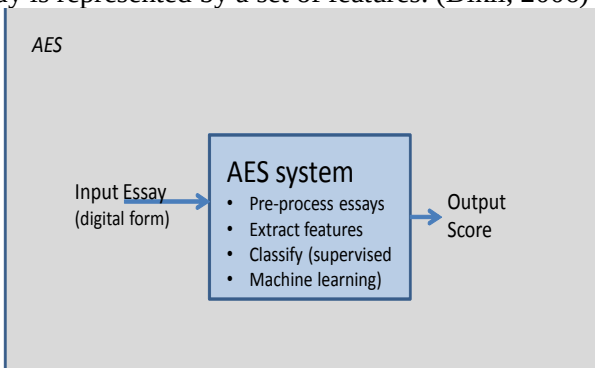

Figure 1 AES overview in operation

The life cycle of an AES system can be divided into two main phases: 1) a development phase and 2) an operational phase. During the development phase the AES system software is created and it may be a time consuming activity. An important input for this phase is the essays with the corresponding scores that are used as the training data to build a supervised machine learning classifier. When the development phase is completed a fully operational AES system is available.

During the operational phase the AES system is used to score new essays. The inputs are new essays that do not have any scores and the execution time is very short. The AES system extracts the relevant features and it assigns the corresponding scores that are output. An important aspect to take into consideration is that the operational AES should only be used to score new essays that come from

the same distribution (population) as the training data. For instance, if an AES is created using essays from high school students as the training data and the same AES is used to score new essays from university students studying literature, the AES will not have a good performance because the essays do not belong to the same population (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 15-27).

2.2.1 Summary of advantages of AES

The following are examples of advantages for the AES that are described in (Williamson , Bejar, & Hone, 'Mental Model' Comparison of Automated and Human Scoring, 1999): 1) the same essays will always get the same scores. AES has a higher degree of reproducibility than human raters. 2) The AES system will always apply the same criteria in the essay evaluation. The AES system is more consistent than human raters who can be affected by tiredness, distraction, prejudices, etc., while scoring. If the AES system is trained on a high quality training material (essays and their scores), it will keep a consistent high quality. 3) Efficiency aspects include cost reduction and faster results when evaluating essays. AES system is able to evaluate a large number of essays in an effective way (compared to humans). The larger volume of essays, the more efficient is the AES system because it has constant initial cost.

Another advantage of AES is to provide instant feedback to the students during writing exercises. Some AES systems provide Web interface where the students write the essay and receive immediate feedback in different areas. More training and more feedback will result in better writing skills.

2.2.2 Criticisms

Understanding and appreciating good writing is a very complex human activity that requires education and training. Therefore, the proposal that an AES system is able to evaluate and to score essays has generated criticism. Page and Petersen (The computer moves into essay grading:Updating the ancient test, 1995) identifies three classes of objections: 1) the humanistic, 2) the defensive and 3) the construct objections.

- The humanistic objection is related to the fact that writing can be compared to a form of art and a computer program cannot judge human creativity. In many cases, this criticism is also related to professionals that may feel threatened to become redundant by a machine (i.e. writing teachers). This objection will continue to be a long lasting philosophical discussion about computers, human intelligence and what is possible and not possible in this area. One way to handle the humanistic objection is to clarify that AES system is best used as a complement to human raters.
- The defensive objection is related to the vulnerability of AES systems to students that may try to use knowledge in the functions of the AES systems to get a better score than deserved (cheating the system). Commercial AES systems try to protect against this vulnerability by detecting abnormal structures (too long, too short, too repetitive, off-topic, etc.), but this is an issue that requires continuous attention and improvements.
- The construct objection is related to the statistical criteria used by the AES system to assign scores and the lack of direct influence by experts (writing domain). In many cases, the AES system is using a large number of features that have a good correlation to human raters' scores but do not reflect characteristics of good writing (e.g. the relative number of words used as a feature). The statistical algorithms define the weights for the different features each time a new classifier is created. The features and the statistical algorithms allow the AES system to predict scores very similar to human rater's. But the lack of human control makes acceptance of AES systems more difficult among teachers and the writing community. Another issue is that the qualities of the AES systems' scores are very much dependent upon the quality of the training material used.

The type of features used and the possibility for writing experts to influence the scoring process is an important area of discussion. There is a study that investigates three different approaches to handle the selection of features (Ben-Simon & Bennett, 2007): 1) the brute-empirical, 2) the hybrid and 3) the

substantively driven. The brute-empirical approach defines a large number of features with many different characteristics. The AES system uses purely statistical methods to select and to weight the features used during the classification. The hybrid approach has a pre-defined and limited set of features; the features are related to good writing characteristics with some theoretical foundation. The AES system uses statistical methods to define the weight of the features used during the classification process. The substantively driven approach has a pre-defined and limited set of features; the features are related to good writing characteristics with a theoretical foundation. The most important characteristic here is that the weights of the features used during classification are controlled by writing experts that are able to give different weights to the features depending upon the test requirements. That is, in this case the AES system is much more controlled by the experts. The study shows that it is possible to achieve reasonable results with the substantively driven method but the statistical selection methods (1 or 2) produce results with higher levels of inter-rater agreement.

2.2.3 Evaluation of AES

AES assigns scores to essays and these scores are characterized by their validity and reliability. Validity is defined by (Cizek, Defining and Distinguishing Validity: Interpretations of Score Meaning and Justifications of Test Use, 2012) as:" Validity is the degree to which scores on an appropriately administered instrument support inferences about variation in the characteristic that the instrument was developed to measure". That is, what are the conclusions that can be made from a test score? Can we make conclusions or inferences about the writing skills of a student who gets a certain score in an essay test? What is the empirical evidence that supports those conclusions?

A validation strategy for AES may take into consideration: 1) the different statistical relations between scores assigned by human raters and the scores assigned by the AES system (reliability aspects), 2) the agreement between independent measurements of students' writing skills and the scores assigned by an AES system and 3) the various aspects of AES, e.g. selection of features and their weighting (Yang, Buckendahl, Juskiewicz, & Bhola, 2002).

An important validity aspect is the vulnerability of the AES for manipulation that may affect the scores in an undesirable way. The students may write essays that are optimized to satisfy the features of the AES in order to get higher scores. For example, if an AES is mainly using word counts and length based features, an essay may be meaningless for a human reader but it may get a high automated score because it is optimized to satisfy the AES's feature requirements (for example, the text may contain a bunch of words with optimal length, optimal frequencies, optimal sentence lengths, etc.). That was a problem in the early versions of PEG (Dikli, 2006). In one vulnerability test with the older version of the E-rater, an essay writer got the highest score by writing a couple of "good" paragraphs that were repeated 37 times. Of course the same essay was assigned a low score by human raters (Powers, Burstein, Chodorow, Fowles, & Kukich, 2001).

2.2.3.1 Reliability evaluation

Reliability is a concept that may contain different aspects like consistency, dependability and reproducibility (Cizek & Page, The Concept of Reliability in the context of automated Essay Scoring, 2002). Reliability is an important pre-requisite for validity but not the only one. Traditionally, the reliability aspect that is measured for AES is the level of agreement between the scores assigned by human raters and the scores assigned by the AES system. There are no real golden standards regarding essay scores. In many standardized tests the scores are assigned by blind human raters that have the necessary training and experience.

There are different ways to calculate the level of agreement between scores assigned by AES system and human raters. The basic way is to count the number of scores assigned by the AES system that agree with the human raters relative to the total number of scores (in percent). This basic method is described by (Burstein, Kukich, Wolff, Lu, & Chodorow, 1998). However, there are different ways to define "inter-rater agreement" (Yang, Buckendahl, Juskiewicz, & Bhola, 2002; Cizek & Page, The Concept of Reliability in the context of automated Essay Scoring, 2002; Johnson, Penny, Fisher, & Kuhs, 2003). One way is to define that there is an agreement when there is an exact agreement between the two raters (percent of exact agreement). Another way is to define that there is an agreement even if the scores differ by a maximum of one point (percent of adjacent agreement that

also includes the exact matches). The adjacent agreement criterion has been used by GMAT and TOEFL (large scale standardized tests in the US) and it is used as agreement criteria in articles describing AES in the US (Burstein, Kukich, Wolff, Lu, & Chodorow, 1998; Chodorow & Burstein, 2004).

A Swedish study (Hinnerich, Höglin, & Johannesson, 2011)has shown an exact inter-rater agreement of 46% between teachers scoring their own students and blind raters in the standardized test. A report that describes the New York State Testing Program for English as a Second Language Achievement Test (NYSESLAT) shows exact inter-rater agreement levels in the range of 43.46% to 54.16% (Pearson, 2009). It should be noted, that the comparison of inter-rater agreement levels between different standardized tests is difficult because they may be using different scales for scoring as well as different definitions of agreement. For example, in some tests they use a scale with 5 scores and in other tests a scale with 3 scores. In any case, whenever referring to inter-rater agreement, it is important to specify its meaning.

We can also compare the level of agreement between two human raters and the level of agreement between the AES system and a human rater to show the level of consistency (Burstein, Kukich, Wolff, Lu, & Chodorow, 1998).

The use of percentage of exact/adjacent agreements provides only rough estimates of the level of agreement. According to (Yang, Buckendahl, Juskiewicz, & Bhola, 2002) it may present unreliable results depending upon the statistical characteristics of the data because it does not take into consideration the effect of chance. Kappa statistics are used to evaluate the level inter-rater agreements because it takes to consideration the chance factor (Sim & Wright, 2005).

Essay scores are typically an ordinal variable. When comparing inter-rater agreement it is important to take into consideration that disagreements are more or less serious depending upon the scoring discrepancy. For example, one rater scores an essay with a 3 and the other rater scores the essay with a 4. The scores are different but they are near each other. Another example is if a rater scores an essay with a 1 and the other rater scores the same essay with a 5. Here there is a large discrepancy between the scores and it is a more serious disagreement. The weighted Kappa statistics is able to take into consideration the magnitude of the discrepancies. A common weighted Kappa is the quadratic weighted kappa (QWK). The following QWK calculations are based on (Warrens, 2012). In order to facilitate the calculation QWK, the essay scores are structured in a confusion matrix F where the number of rows and columns correspond to the number of scores. The element f_{ij}indicates the number of essays that rater 1 assigned a score "i" and rater 2 assigned score "j". The matrix A is defined as the normalized version of the matrix F (A = F/m) where m is the number of essays. The matrix A has a generic element a_{ij} . For each row in the matrix A there is a row total defined as: $p_i = \sum a_{ij}$. For each column in the matrix A there is a column total defined as: $q_i = \sum a_{ji}$. The following are the formulas to calculate QWK using the aforementioned terms.

$$\kappa_q = 1 - \frac{O_q}{E_q}$$

O_q is the proportion of the observed values. $E_q$is the expected value. The observed value is calculated as:

$$O_q = \sum\sum (i-j)^2 \, a_{ij}$$

The expected value is calculated as:

$$E_q = \sum\sum (i-j)^2 \, p_i q_j$$

If the kappa value is zero, it means that the results are purely by chance. If the kappa value is 1, all the observations are true agreements.

When evaluating the reliability of AES, it is important to take into consideration that (most) AES systems use human rated essays as the basis (training data) to build the classifier. If the agreement level between human raters is low, it will impact the reliability of the AES system's scores in a negative way (Williamson, Xi, & Breyer, A Framework for Evaluation and use of Automated Scoring, 2012).

Educational Testing Service (ETS) (ETS home, 2012) is using statistical criteria to define if the quality of AES is acceptable for use in high-stakes standardized tests or not. The quadratic weighted

kappa and correlation coefficients are used as statistics here and the following requirements must be fulfilled (Williamson, Xi, & Breyer, A Framework for Evaluation and use of Automated Scoring, 2012):

- The level of agreement between the scores assigned by the AES system and the human raters must be at least 0.70 (both coefficients).
- The difference between the level of agreement between human raters and the level of agreement between the AES system and human raters must be less than 0.10 (both coefficients). For example: KHR is the level of agreement between human raters. KAH is the level of agreement between the AES system and the human raters. The requirement implies that KHR - KAH < 0.10. The same applies for the correlation coefficients.

2.2.4 Commercial AES

One important driving force behind the commercial development of AES systems was the US school authorities when they started deploying large scale standardized tests in the 1960s. The standardized tests in many cases included essays and it was expensive to use human raters. There has been a demand for efficient ways to score a large number of essays. Today there are several commercial AES systems that are deployed in the US in large scale commercial applications. Here are examples of some established commercial AES systems in the US:

- The first automated essay scoring system was the Project Essay Grader (PEG) that was developed in 1966. PEG is based on machine learning, using training data to find suitable correlations to the human scores. PEG has been further developed and the current version is handling text structure, some contents aspects, mechanics, etc. (Dikli, 2006).
- Intelligent Essay Assessor (IEA) is based on Latent Semantic Analysis (LSA). It is able to handle the content (semantics aspects), style, structure and mechanics. The system is capable not only of assigning scores but also providing feedback in the different areas. The classifier is built using training material in the domains to be scored (Dikli, 2006).
- E-rater version 2 is provided by ETS. ETS is a non-profit organization that provides services in the area of education (ETS home, 2012). E-rater uses a limited set of features that are related to good writing characteristics. It is able to handle the mechanics, structure and contents of the essays. Different tagged corpora are used as the reference in different areas. For example, grammar aspects (mechanics) are handled by large corpus of good English texts that are used as a reference (bigram language model). Corpus annotated with discourse elements are used to handle the structure aspects. For each prompt, the e-rater creates a corresponding word-vector that is used to evaluate the contents aspects. E-rater allows experts with competence in the (writing) domain to trim the feature weights based upon the test requirements. Of course, E-rater provides also statistical methods (multiple regression analysis) to define the optimal weights for the features (Attali & Burstein, 2006).
- Criterion is a Web based essay training tool using the E-rater modules. Criterion is used by schools as a writing training tool in the school classes. The students write their essays and they get an immediate evaluation with detailed feedback about the different issues in the essay. The feedback covers areas like the mechanics, structure and style of the essays (Attali & Burstein, 2006).
- IntelliMetric is developed by Vantage Learning. It is based on some machine learning techniques using a very large number of features and different statistics methods. The features and functions are not disclosed (commercially protected). It is able to handle the mechanics, structure and contents of the essays. Parts of speech and syntax structures are examples of features used when dealing with grammar aspects. The same system is able handle essays in different languages, English, Spanish, French, Hebrew, etc. (Dikli, 2006). IntelliMetric is handling the GMAT

Analytical Writing Assessment (AWA) tests. It is able to handle "analysis of an Issue" and "analysis of an Argument "prompts (Rudner, Garcia, & Welch, 2005).

- MY Access is a Web based essay scoring tool that is based on IntelliMetric system. It provides immediate feedback and scores to essays. It is intended to be used as a tool in writing classes (Dikli, 2006).

2.3 Supervised Machine learning

The development of AES based on supervised machine learning can be divided into two main areas: 1) feature extraction and 2) building supervised machine learning classifiers (Guyon & Elisseeff, An Introduction to Feature Extraction, 2006).

2.3.1 Feature extraction

Feature extraction is the process of identifying relevant features for scoring essays. Feature extraction consists of two main phases: 1) Identification and construction of possible features and 2) selection of relevant features (Guyon & Elisseeff, An Introduction to Feature Extraction, 2006).

Feature construction involves identifying and constructing possible features. During this phase, domain knowledge of language and writing is essential to define all the features that may be relevant to evaluate essays. Different features may be combined in different ways in order to create new features that may represent new characteristics of the essays. One example is to combine different parts of speech counters into a feature called Nominal Ratio that measures the idea density for the text (Smith & Jönsson, 2011). It should be pointed out that during this phase it is not possible to be sure that the features will be relevant for the AES .They are our initial best guesses (Guyon & Elisseeff, An Introduction to Feature Extraction, 2006).

The next step is a feature selection process where the relevant features are identified and ranked. The main purposes of feature selection are to improve the AES accuracy, to provide a better understanding of the relationship between features and the essay scoring, and to control the load on the runtime environment (Guyon & Elisseeff, An Introduction to Feature Extraction, 2006). As a general principle, the identification and removal of non-relevant features implies improved generalization with better accuracy for the classification of new data. This is because a better generalization can be achieved with less complex machine learning algorithms with fewer features (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 167-171).

The feature selection process may be based on individual features (univariate) or it may be based on multiple features (multivariate). In the individual feature case, each feature is individually ranked based on how much they contribute to the classification results. The features that do not contribute (non-relevant) are discarded. The drawback with this strategy is that it will miss all the cases of feature interaction. For example, feature "A" may be identified as a weak contributor and feature "B" may be identified as irrelevant. However, if we combine feature "A" with feature "B" (A and B interaction) they may show a very strong correlation with the results. Examples of statistical methods used for univariate feature ranking are chi-square or F-score (Dreyfus & Guyon, 2006). In the multiple feature case, the process of feature selection can be compared to a search problem with the purpose of finding the subset of features that produces the best classification results. One alternative is to run different subsets of features through the classifier and to use the results as relevance criteria. The use of the classifier as the evaluation criteria is called the wrapper method. One example of the wrapper method is to evaluate all the feature subsets in order to find the optimal solution. If there are N features, there are 2^N possible subsets of features that must be processed by the classifier. This method becomes very expensive in computer resources even with a limited number of features. There are different heuristic strategies to handle this search and the choices are dependent upon the number of features. One example is the sequential backward selection (SBS) method where you start with a set with all the features. For example, the set of features has N elements and it decreases by one in each step. For each step the algorithm removes the feature with the weakest contribution to the classification. This process continues until the optimal set is detected. Another example is sequential forward algorithm where you

start with an empty set of features and add one feature at the time that maximizes the classification results (Guyon & Elisseeff, An Introduction to Feature Extraction, 2006).

2.3.2 Supervised machine learning classifier

Building a supervised machine learning classifier can be divided into training and test phases that are described in more detail in this chapter. Figure 2 presents an overview of the components and steps involved in supervised machine learning. A prerequisite for supervised machine learning is that there is data. In this case, there are essays, their features and scores. The starting point is to describe the question to be solved as a model where there is a (unknown) function called a target function ($f: X \rightarrow Y$) that defines the relationship between essays, their features (X) and the corresponding scores (Y). This function may be impossible to be defined analytically but supervised machine learning uses the training data to define an approximation for this (unknown) target function. The training data is a set of essays that are already scored by human raters. The training data can be modeled as the set $\{(x_i, y_i)\}$, where x_i is a vector that contains the features for an essay (symbolizes an essay) and y_i is the corresponding score. The number of features defines the dimension of the X-space. For example, if the training data contains 1000 essays and each essay has 20 features, the X-space has 20 dimensions. The training set contains 1000 pairs of x vectors with the corresponding scores $(x_{1,}\ y_1), (x_2, y_2), \ldots\ldots (x_{1000}, y_{1000})$. The learning algorithm uses the training data, the hypothesis set and certain optimization criteria in order to define a final hypothesis h_f that approximates the target function. The hypothesis set includes the hypotheses that are considered during the learning activity. For example, if we decide to use a linear algorithm, the hypothesis set would include all the hyper-planes that may be used during the training process. The optimization criteria could be some classification error measurement combined with other criteria. The final hypothesis h_f is the hypothesis that best fits the training data and that fulfills all the required optimization criteria. The software system that uses h_f together with an algorithm in order to analyze new essays and to assign the corresponding scores is called a classifier (classifier = final hypothesis + algorithm). For further information about the principles for machine learning refer to (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 1-31).

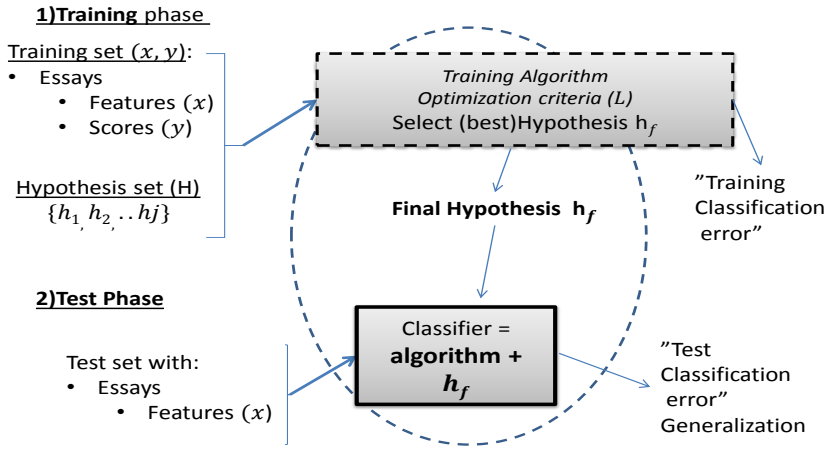

Figure 2 main components for supervised machine learning

One characteristic of the classifier is its training error that shows how well it does when classifying the training data. The fact that the classifier is able to classify all classes in the training data correctly (training error = 0%) does not assure that it will perform well when classifying new data. The key question is how well the classifier is able to generalize and to rate new essays in a consistent way. Generalization error is the error that the classifier makes when classifying new data. There are different ways to measure the generalization error. One way is to use a set of test data that was not involved in the training process. The test data contains essays and the corresponding scores assigned by human raters. It is important that that all the data used during training and test are random samples

from the same population. During the test process, the classifier receives only the set of features that represent the essays and it produces the classifier scores. The classifier scores are compared with the human raters and the number of classification errors is computed. If there is enough test data, the relative amount of classification errors during test is a good estimate for the generalization error for the classifier. It is important to avoid any situation where the test data is used in any situation related to training. Otherwise, the test results will not represent the true generalization error (test results will be too optimistic). One example of a situation where we may contaminate the test data is the following: we create a classifier using the training data as usual. Now we use the test data just once to verify the classifier and we are not satisfied with the results. We go back to the training data in order to fine tune some parameters and to make other improvements. Next time, we use the same test data again and we may get very good test results but they may not be a reliable indication for the true generalization error. The test data has indeed been used in some training activities. For more information about training and testing refer to (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 39-68).

Very often it is necessary to train and to create several classifiers using different parameters in order to select the optimal ones. This is called a grid-search procedure. Each new classifier (with a set of parameters) is tested with a new test set and the classifier that has the lowest generalization error is selected. That is, a traditional testing procedure will require a large number of test sets that may not be available (training and test data are always scarce). A solution for this limitation is to use the cross validation procedure that uses the same data set to perform both training and test. A well-established procedure is the k-fold cross-validation, where the data is divided in k-folds (disjoint sets). In each step, k-1 folds are used for training and one fold is used for test and this process is repeated until all k-folds are used for testing (and the corresponding k-1 complement sets are used for training). In the k-fold cross validation, each step will produce a cross validation error. The final cross validation error is the average of the individual cross validation errors ($Err_{cv} = \frac{1}{k}(\sum_{i=1}^{k} err_{cv-i})$). The final cross validation error is an upper bound for the generalization error for the classifier. The value of k is determined based on a compromise between maximizing the size of training sets and keeping the computation power required at a reasonable level. If k = n where n is the number of data samples (i.e. number of essays), the size of training sets in each step is maximized (n-1 data samples used for training and 1 data sample for cross validation test). This cross validation configuration is called "leave-one-out" (LOO). LOO requires the execution of n-steps; therefore it may not be practical for a large value of n due to the computation power required. Typically k = 10 (or k =20) is used as a compromise. For further information about a more theoretical background for the cross validation procedure refer to (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 145-153). In summary, the selection of optimal parameters may create many classifiers (configured with different parameters) and each classifier uses a cross validation procedure. The classifier that has the lowest cross validation error is selected as the optimal solution. Here it is important to be aware that if we use the cross validation procedure too many times, there is a risk that the cross validation error may not be representative for the true generalization error.

Other important concepts in machine learning according to (Abu-Mustafa, Magdon-Ismail, & Lin, 2012) are bias-variance trade-off, overfitting, regularization, VC dimension and Occam's razor principle.

Bias-variance is a theoretical concept and it is related to complexity of the hypothesis set. It can be shown that the generalization error contains a bias component and a variance component. The bias is related to how well the hypotheses in the hypothesis set may approximate the target function (best possible approximation). The variance is a measurement of the instability in the hypothesis set and the classifier. A large variance means that small changes in training data imply major changes in the selection of new hypothesis. Typically, if we increase the complexity of the hypothesis set, the variance increases and the bias decreases. As a consequence we get better performance in the training but worse performance in the generalization error. A simpler hypothesis set implies that the bias increases and the variance decreases. As a consequence we may get worse performance during training but better generalization performance. That is, the trade-off between bias and variance is always present when defining and improving a classifier (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 62-66).

Overfitting means that the training algorithm with a complex hypothesis set tries to fit the training data too well, learning not only the relevant information but different noisy (irrelevant) components as well. A typical situation where there is overfitting is to use a too complex hypothesis set that enables the definition of very complex boundary surfaces. The complex boundaries are very flexible and they may try to fit all patterns in the data including noisy data. The result is a classifier with very low training error (it may achieve 100% correct classification during training) but the classifier does not generalize well (larger generalization error). A less complex hypothesis set with a simpler boundary surface may fit only the major data patterns (increased bias) and it will ignore most noisy components (lower variance). The result is a classifier that is more stable and has some training errors but has better generalization performance. Regularization is a counter-measure in order to avoid overfitting that may be used with many different training algorithms. Regularization imposes some limitations in the training algorithm that cause the selection of a less complex hypothesis (fewer parameters or a number of parameters with very low values). The classifier with this regularized hypothesis (less-complex) may have an increased training error but a better generalization performance. That is, regularization becomes a new parameter in the optimization criteria used by the training algorithm when defining the best hypothesis (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 119-138).

The VC dimension (Vapnik and Chervonenkis dimension) is a measurement for the complexity of the hypothesis set. The VC dimension is an important parameter when defining the upper bound for the generalization error for any classifier based on the corresponding hypothesis set (the number of training data is another important parameter). The larger VC dimension, the more complex the hypothesis set and the higher the risk for poor generalization. In some hypothesis sets, the VC dimension is related to the X-space dimension (number of features) (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 50-58).

The golden rule for machine learning is simplicity. That is, the goal is to have simplest possible classifier (built on the simplest hypothesis set) that fulfills the requirements. The reason is that a simpler classifier has a better chance of providing good results when classifying new data (lower generalization error). This simplicity principle is sometimes called “Occam’s razor” (Witten & Frank, 2005).

Machine-learning algorithms have some mathematical foundation but the results depend very much about the assumptions that are made and how the parameters are fine tuned. The same algorithm may have an excellent performance or a very poor performance in the same data depending upon how it was tuned. In most cases there are no absolute truths. A very important factor is the practical experience with the different algorithms and the ability to fine tune the corresponding parameters in order to achieve a good result (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, p. 151).

The following chapters describe in more detail the principles for Support Vector Machine, Linear Discriminant Analysis and different Ensemble classifiers.

2.3.3 Support Vector Machine (SVM)

SVM is an algorithm for supervised machine learning classification that has a strong theoretical background and it has been used successfully in many different areas of applications including bioinformatics (Ben-Hur & Weston, 2010) and text classification (Manning , Raghavan, & Schutze, 2008, pp. 293-318). An ideal case for classification is where the data is divided between two classes (some data points belong to class 1 and some data points belong to class 2) and the classes are linearly separable. In this case, there are an infinite number of hyper-planes that define boundaries that separate the two classes. SVM in its basic configuration is used to classify binary data that is linearly separable. See figure 3 for an overview of SVM components. The optimization criterion for the SVM training algorithm is to define a hyper-plane that maximizes the distance between the margins and achieve the widest margin between the classes. When the hyper-plane is defined (parameters w and b), the classification of an unknown data x is based on the (hyper-plane) equation $h(x) = sign(w^T x + b)$. For example, x may be classified as belonging to class 1, if $h(x) = +1$ otherwise it may be classified as belonging to class 2 ($h(x) = -1$).

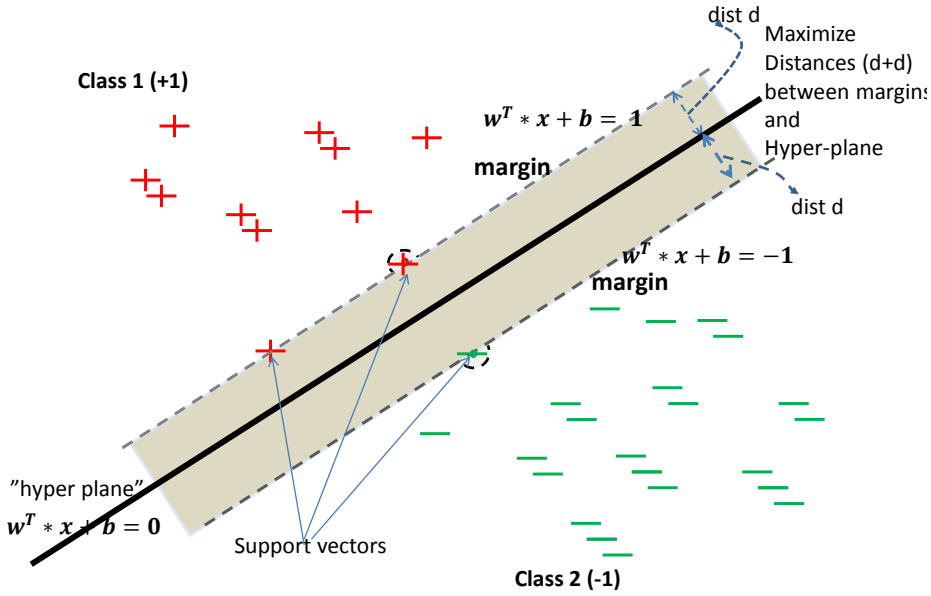

Figure 3 SVM with linearly separable data, its hyper-plane, margins and three support vectors (Manning , Raghavan, & Schutze, 2008)

The important parameters are the support vectors that are the points that belong to class 1 and to class 2 that are most near the hyper-plane. The support vectors define the margins and the hyper-plane position. Notice that we may add more data points, or move the points around in class 1 and class2 and the hyper-plane is not affected if the support vectors are not affected. The fact that SVM selects the hyper-plane with the widest margin implies a more robust hypothesis that tends to generalize well. The solution described in figure 3 is an ideal solution where the classes are nicely linearly separable and the margins are not violated (hard margin SVM). In reality there may be noisy observations and a training data may not be linearly separable. Regarding noisy observations, SVM enables the definition of a hyper-plane that keeps a wide margin at the same time that it allows that a few data points are violating the margins. Typically, this capability is controlled by the parameter C in the standard SVM packages. The parameter C defines the penalty for the violation of the margins; a small C value allows more points violating the margins and larger C fewer points. This capability is called soft margin SVM and it can be seen as a form of regularization for SVM (Manning , Raghavan, & Schutze, 2008, pp. 300-302).

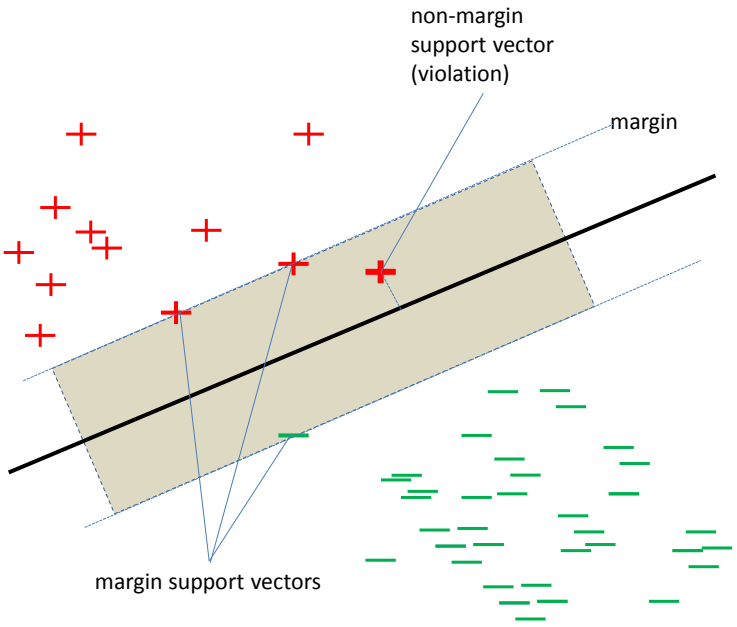

Figure 4 Support vectors that violate the SVM margins (Manning , Raghavan, & Schutze, 2008)

Now there are two kinds of support vectors, the margin support vectors that define the margins and non-margin support vectors that correspond to the points that violate the margins. Notice that all support vectors contribute to performance of the final hypothesis. The optimal value for C is typically defined with the help of cross validation.

If the data (x) is not linearly separable in the original feature space, the standard procedure is to transform the input data into a higher dimensional feature space (Z feature space) where the data is assumed to be linearly separable (Manning , Raghavan, & Schutze, 2008, pp. 303-306). The input data x is mapped into a higher dimensional Z feature space by the function $z = \Phi(x)$. Now the training is performed in the Z feature space using the corresponding data $(\Phi(x), y)$. The best hypothesis is defined and the hyper-plane in the Z feature space is the boundary for the classes in this space. Notice that the hyper-plane in the Z feature space (linear surface) may correspond to a very complex boundary surface in the original feature space. The classification of an unknown x is based on the hyper-plane equation $h_\Phi(x) = sign(\tilde{w}^T\Phi(x) + \tilde{b})$. The equation is on the Z space but it depends only on x and it generates the corresponding classification result. The mechanism used by SVM for mapping the input data into Z feature space is called kernel. The kernel provides a simple mathematical mechanism that computes the required operations in Z space (inner product) without requiring that the input data is first transformed into the Z-space. All the transformations are done in a "hidden" way by the kernel. The kernel enables the use of very high dimensional feature spaces. Examples of standard kernels supported by SVM are:

- Linear kernel creates the basic SVM with a linear hyper-plane (Hsu, Chang, & Lin, 2010).
- Gaussian Radius Basis Function (RBF) kernel creates a feature space with infinite dimension. RBF is very flexible and simple because it contains only one parameter. Therefore it is usually a good starting point when selecting a kernel and the characteristics for the data are unknown (typical situation). RBF kernel is defined by the following equation: $K(x, x') = \exp(-\gamma\left||x - x'|\right|^2)$, where $||.||$ is the Euclidean distance, $\gamma > 0$ is a constant and x, x' are two points in the input feature space (Hsu, Chang, & Lin, 2010). The γ parameter can be visualized as representing the width of the surfaces that represent the distribution of the distances between the points. It has been shown (Keerthi & Lin, 2003) that if the parameter γ gets very small values ($\gamma \to 0$), the RBF kernel(C, γ)

behaves like a linear kernel. This fact can be used to facilitate the model selection procedure where the same RBF kernel is used to cover both linear and non-linear models (one kernel versus two).

- Polynomial kernel creates a higher dimensional polynomial feature space. The dimension of the feature space is defined by the grade for the polynomial. The kernel is defined by the following equation $K(x, x') = (\gamma * (x^T * x) + b)^d$, where d is the grade of the polynomial and $\gamma > 0$ and b are constants (Hsu, Chang, & Lin, 2010). The parameter b describes the bias for the polynomial surface. For example, if it has value zero, the surface will pass through the origin otherwise it will not.

An issue that must be taken into consideration when using a higher dimension feature space is the increased complexity of the hypotheses and the risk for poor generalization. In the SVM case, the complexity of the hypotheses is defined by the number of support vectors regardless the number of dimensions in the feature space. For example, if we generate a SVM hypothesis using a very high dimensional Z space (may be 5000 dimensions or more) but the hypothesis has 4 support vectors, the complexity of the hypothesis is low because there are only 4 support vectors. The expected value for the generalization error for SVM is bound by the following equation:

$$\boldsymbol{E}\left[E_{gen}\right] < \boldsymbol{E}\left[\frac{total\ number\ of\ support\ vectors}{number\ of\ data\ points}\right]$$

The total number of support vectors includes both margin and non-margin support vectors. The generalization error is not related to the dimensionality of the data (or the number of features); it is only related to the (total) number of support vectors and number of training data. This is a very important characteristic for SVM that may have a low generalization error (few support vectors) at the same time that it is supporting an infinite number of dimensions (Vapnik, 2000). If we are using cross validation, the total generalization error is bound by the average of each cross validation error (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 145-150).

A SVM in its original version is a binary classifier where a hyper-plane is the boundary between two classes. There are different schemes in order to support multiple classes. One scheme is called one-versus-one where a binary classifier is created to compare two classes and this process is repeated until all classes are compared to each other. If there are n classes, $n(n-1)/2$ classifiers are created. Each classifier generates one vote and the class that gets most votes is selected. For example, there are three classes a, b and c in the data. One classifier is created to classify a-b and it selects a. One classifier is created to classify b-c and it selects b. One classify is created to classify a-c and it selects a. The final classification result is that the data belongs to the class a (most votes). LIBSVM uses this scheme to handle multiple classes (Chang & Lin, 2011). Another scheme is called one-versus-all where each class is compared against all the others. That is, n classifiers are created and the classifier that produces the best results (widest margins) is selected.

2.3.4 Linear Discriminant Analysis Classifier (LDAC)

Linear Discriminant Analysis classifier defines a hyper-plane ($w^T x$) that maximizes the objective function $J(w) = \frac{w^T S_B w}{w^T S_w w}$ (this criterion is called Fisher's linear discriminant). S_B is the between-class scatter matrix and S_W is within-class scatter matrix. The within-class scatter matrix for the class c_i is defined as $S_i = \Sigma_{x \in c_i}((x - \mu_i) * (x - \mu_i)^T)$ where x are the input data in the class and μ_i is the corresponding mean value. In the case that we have two classes, the corresponding within-class scatter matrix is $S_W = S_1 + S_2$. In the case there are two classes; the between-class scatter matrix is defined as $S_B = (\mu_1 - \mu_2) * (\mu_1 - \mu_2)^T$, where μ_1, μ_2 are the means for class 1 and class 2. The scatter matrices are the same as the covariance matrices multiplied by a constant. (Zhou, 2012, pp. 3-4).

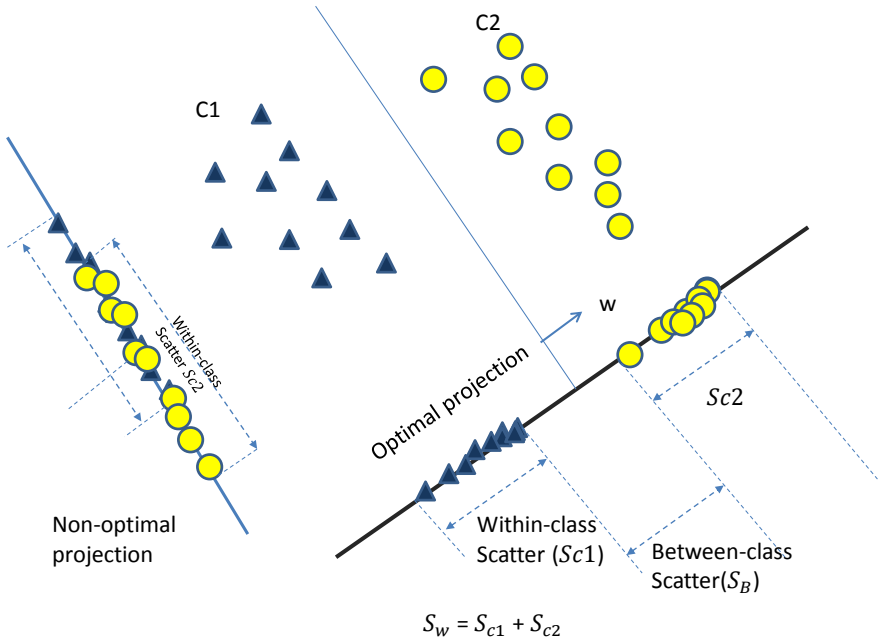

Figure 5 Geometrical interpretations of principles for linear discriminant analysis

There is an intuitive geometric interpretation of the linear discriminant classifier in the two dimensional case as shown in figure 5. Figure 5 describes two classes (C1 and C2) with the corresponding data and projections. The data points project to different lines with very different characteristics for the scatters between-classes and within-classes. There is one example of a non-optimal projection where the within-classes scatters are large and between-class scatter is very small. The linear discriminant classifier searches different axis and their projections and it finds the line where the data is projected with maximum between-class scatter and minimum within-class scatter (optimal projection) (Zhou, 2012, pp. 3-4).

If the LDAC package does not support kernel transformations, the transformation of the input features into a higher dimensional feature plane must be done manually. For example if the input is two dimensional $x \in \{(x_1, x_2)\}$ a transformation into a polynomial feature space with 6 dimensions can be made by $\Phi(x) = (1, x_1, x_2, x_1 * x_2, x_1^2, x_2^2)$. (Abu-Mustafa, Magdon-Ismail, & Lin, 2012, pp. 99-104).

2.3.5 Classifier ensemble

It is possible to create a new classifier by combining a number of classifiers where the performance of this new classifier is better than the components. This new classifier is called ensemble of classifiers. A performance improvement will happen if the components (classifiers) are performing better than random guessing (classification error < 50%) and the errors made by the classifiers are independent of each other. Refer to (Hansen & Salamon, 1990). The following are examples of ensembles based on trees.

One way to create ensemble classifiers is called “boosting” or “Adaboost” (Schapire, 1999) where weak classifiers are created in sequence with the purpose to create a boosted ensemble classifier. The principles defined here are assuming the basic scenario with binary classes. A weak classifier is any classifier with a classification error slightly better than random. That is, it fulfills the condition $P(h_j(x) \neq y_j) < 0.5 - \gamma$ where $\gamma > 0$ for every data point. There is a hypothesis set that is used to create the (weak) classifiers and the same training data is used. The training data has a certain distribution where different data points may have different weights. Typically a weak classifier will misclassify some data points. The misclassified data points get an increased weight and the other points (correct) get a decreased weight. This new distribution of the input data will influence the

selection of the next weak classifier. That is, each new (weak) classifier is created with the purpose of adding some values in relation to the previous classifiers. This algorithm is repeated recursively a number of times (T). At the end, the weak hypotheses (T) are combined in order to create an aggregated hypothesis/ensemble. The number of hypotheses in the final aggregation (T) affects its accuracy. A simple aggregation is based on majority voting $h_{agreg}(x) = sign(\sum_1^T h_i(x))$. For example, a new data x is fed into the classifier and each (weak) hypothesis h classifies the data. Some h will classify as +1, others will classify as -1. The final classification result is +1 if the majority has agreed on +1. There are more advanced aggregation mechanisms based on combining the hypothesis based on different weight.

A second way to create ensemble of classifiers is called bagging (bagging as related to "bootstrap aggregation") according to (Breiman, 1996). The bagging method aggregates a number of weak classifiers that are created independently of each other. That is, each classifier is created with the purpose to classify the data without taking into consideration the other classifiers. Each classifier is created during the training phase, based on the same hypothesis set, the same training algorithm (i.e. decision trees) but using different training data. The different training data are generated by bootstrap samples from the training data. For example, if there are N training data points, each bootstrap sample contains N data points that are randomly selected from the original training data (with replacement). The final aggregation can be based on averaging the results produced by the different components (in case of numerical results) or majority voting (in case of classification). The criterion for a successful bagging algorithm is that there is instability in the hypothesis set where small changes in the data generates different hypothesis (high variance). That is, bagging will decrease this variance and improve the results (Grandvalet, 2004). If the classifiers have a low variance before the bagging procedure, the ensemble may not produce any improvements.

A third way is a method called Extremely Randomized Trees (ERT) (Geurts, Damien, & Wehenkel, 2006) that create a large number of trees always using the same training data (all sampling data). The randomization among the trees is achieved by using the following criteria for splitting the nodes when growing the trees: 1) A few features are randomly selected among all the features, 2) the cut-point for each selected feature is randomly selected and 3) now the best feature is selected to split the node. The number of features used in the split process is an important parameter for the classifier. The final result is calculated in the same way as the other ensemble methods for classification (some majority voting or averaging algorithm).

3 Method

3.1 Data

The following corpora are used:

3.1.1 Essays

A sample of essays from the national test in Swedish B ("nationella prov") taken during the academic year 2005/2006 is used as the essay corpus. The students could choose between two themes and nine topics within the themes. That is, there is a great variation of content. The essays were collected in a project run by the Department of Economics at the Stockholm University (Hinnerich, Höglin, & Johannesson, 2011). The sample was randomly selected in order to represent the population of high school students that took the national test (Swedish B) in 2005/2006. The original sample contained 2800 individuals but only 1702 essays with all the required information (scores and other information) were finally collected. The scale contains four scores. The following symbols are used for each score: "IG" corresponds to failed, "G" corresponds to pass, "VG" corresponds to pass with distinction and "MVG" corresponds to excellent. In order to create an ordinal scale, each score is mapped with a corresponding number: IG = 0, G=1, VG = 2 and MVG = 3. The number scale used here is an internal

representation and it differs from other representations (Hinnerich, Höglin, & Johannesson, 2011). The regular class teachers scored the essays of their own students which is the usual procedure for the "nationella prov". The essays were transformed into digital form. Blind raters were hired to score the essays in digital form without any information about the students. The blind raters were teachers that had previous experience in scoring essays ("nationella prov"). Therefore, I have access to 1702 essays in digital form where each essay has two scores.

A number of tags were added to the text during the transcription and digitalization processes. There were tags indicating new page, missing words, word division at the end of a line, etc. A pre-processing procedure removed the tags in order to create consistent text. The essay corpus was tagged with token, lemma and parts of speech tags by Stagger software (Östling, 2012)

3.1.2 Newspaper article corpus

160 million words from the Swedish newspapers (Dagens Nyheter and Svenska Dagbladet) during the period from 2002 to 2010 were collected. The newspaper articles corpus is tagged with token, lemma and parts of speech tags by the Stagger software (Östling, 2012) . The newspaper corpus is used to define features. There is also a text file with the corresponding token frequency information.

3.1.3 Wordlist corpus

The wordlist SALDO (Borin, Forsbeg, & Lönngren, 2012) contains approximately 123,000 entries and over 1,800,000 word tokens. It is used to check the words in the essays and to define the corresponding feature.

3.1.4 Blog corpus

1800 million words were collected during the period November 2010 to Feb 2012 (from Twingly). The blogs were collected without any consideration about the written language structure. Some authors may have written blogs following the established structure for the Swedish written language regarding both the grammar structure and choice of words. Other authors may have written blogs using a more unconventional language structure where slangs are frequently used. Possible improvement here is to classify the blogs with some quality criteria. The blogs are used to define features. One issue here is that the blogs are covering dates that do not overlap the essays date. The blog corpus is tagged with token, lemma and parts of speech tags by the Stagger software (Östling, 2012). There is also a text file with the corresponding token frequency information.

3.1.5 Complex words set

I have defined a set that contains words that are used to create more complex sentences. The set contains the following words: vilken, vilket, vilka, vars, samt, respektive, därtill, likaså, vidare, såväl, antingen, dock, fast, fastän, ändå, annars, snarare, ty, nämligen, ju, alltså, således, därför, följaktigen, medan, emedan, ifall, såvida, såsom, tills, huruvida, därmed, härmed, sedan, då, än.

3.1.6 Complex bigrams/trigrams set

I have defined a set that contains bigrams/trigrams that are used to create more complex sentences. The set contains the following bigrams: ”även om”, ”trots att”, ”för att”, ”därför att”, ”genom att”, ”utan att”, ”i fall”, ”under det att”, ”till dess att”.

3.2 Training, cross validation and test data

The original essay data contains 1702 essays. 1500 essays are randomly selected and used for the training and cross validation tests. 202 essays are randomly selected as "reserved test data". The purpose of the "reserved test data" is to simulate a set of completely unknown essays and to provide information about the generalization error when classifying new essays. The "reserved data" are kept

locked and unseen. It will only be used when a classifier that has very good agreement with the blind raters (quadratic weighted kappa larger than 0.7) is found. Before that, I will only be using the cross validation error in order to evaluate the results.

The classifiers are trained using the scores assigned by the blind raters as target values because they are assumed to represent more fair scoring results. All training and cross validation tests are performed using the same data based on 10-fold cross validation procedures. Table 1shows the distribution of the scores in the training data:

Table 1 Frequency distribution of blind raters and teachers in training data

	Frequency of scores		
Scores	Blind raters (target values)	Teacher	Score explanation
IG = 0	237	138	failed
G = 1	751	653	passed
VG = 2	399	547	passed with distinction
MVG = 3	113	162	excellent
Total number of essays	1500	1500	
Average and Std. deviation	1.259 ± 0.811	1.489 ± 0.806	

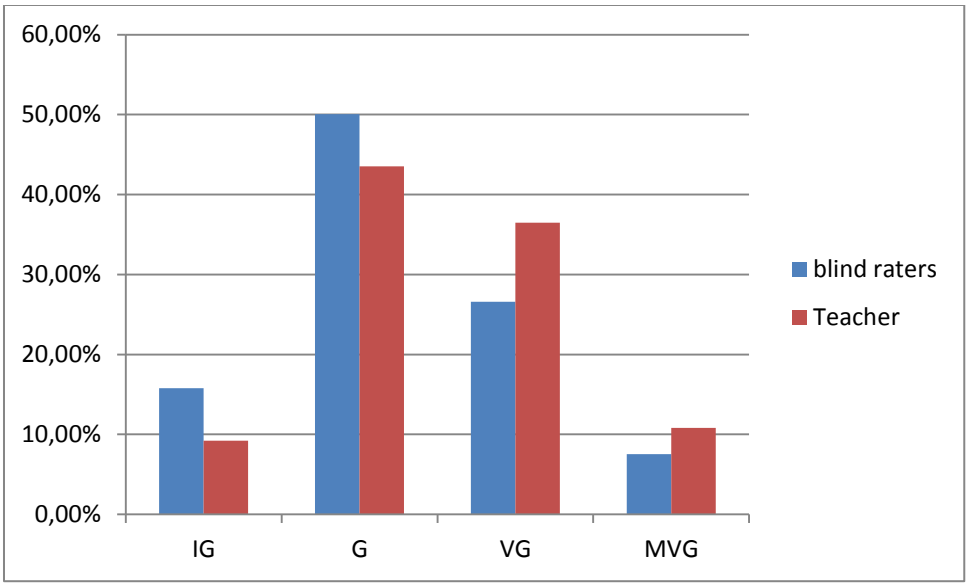

Figure 6 Relative frequency of the essay scores for blind raters and teachers

The bar chart in figure 6 shows the proportional distribution of number of essays per score related to the total number of essays in the training data. The distribution resembles a normal distribution. There are relatively few “tail scores” and relatively many “central scores”. In the blind raters’ scores, 77% of the essays are scored with “G” or VG (central scores). But there are only 7.53% of the essays that are scored with “MVG” and 15.8% that are scored with IG (tail scores).

Table 2 Confusion matrix showing absolute numbers of blind scores teachers' scores in training data

	teacher scores					Row total
Blind scores		IG	G	VG	MVG	
	IG	66	125	42	4	237
	G	59	381	260	51	751
	VG	12	122	199	66	399
	MVG	1	25	46	41	113
Column total		138	653	547	162	1500

The confusion matrix in table 2 describes the absolute values for agreements and disagreement between the teachers and the blind scores. The value in the cell (i, j) indicates the number of essays with blind score "i" that were classified by a teacher with the score "j". For example, the cell defined by the row "IG" and the column "VG" contains the value 42. It indicates that there are 42 essays that are scored with "IG" by the blind raters and with "VG" by the teacher. The values in diagonal cells (i, i) indicate the exact agreements for the ith grade. The ith row total describes the total number essays with blind score "i". The ith column total describes the total number essays with teacher's score "i".

Table 3 Normalized confusion matrix showing the proportion of agreements and disagreements

	teacher scores					Row total
Blind scores		IG	G	VG	MVG	
	IG	27.8%	52.7%	17.7%	1.7%	100%
	G	7.9%	50.7%	34.6%	6.8%	100%
	VG	3.0%	30.6%	49.9%	16.5%	100%
	MVG	0.9%	22.1%	40.7%	36.3%	100%

Table 3 shows the normalized confusion matrix where each row is normalized by the total number of essays that have the corresponding blind score. The value in the cell (i, j) indicates the proportion of essays with blind scores "i" that were assigned score "j" by the teachers. For example, the cell defined by the row "G" and the column "VG" has the value 34.6%. It indicates that 34.6% of all the essays that have the blind score "G" were assigned score VG by the teachers. A value in the diagonal (dark shaded) indicates the proportion of essays with the corresponding score where the blind raters and the teachers are in exact agreement.

The t-test for related data shows that there is a significant difference between the essay scores generated by the teachers (mean=1.489, standard deviation=0.806) and the scores generated by the blind raters (mean=1.259, standard deviation=0.811). The results show that the blind raters tend to assign lower scores than the teachers (on average). This is based on the t-statistics value of 10.11 and a corresponding p-value that is much less than 0.01 (two-tailed).

The Spearman Rank Correlation test (non-parametric) shows that there is a positive correlation between the scores assigned by teachers and the blind raters. The rho-coefficient is 0.485 and the p-value is much less than 0.01.

Table 4 shows a summary of the inter-rate agreement between teachers and blind raters in the training data:

Table 4 Summary of agreements between blind raters and teachers

	Quadratic kappa	Average Accuracy
Exact agreement	0.39121	45.8%
Adjacent agreement	-	78.7%

3.3 Feature construction

Features are measurable attributes in the essays that are used as input to the machine learning software. This chapter describes the feature construction process where possible features are defined. The basic sources of features were standard knowledge in Swedish, input from Robert Östling, the experience of AES in the US as described in the previous chapters and a thesis about the characteristics of written Swedish in the 1970ies (Einarsson, 1978). In general a very broad and ad hoc strategy was used to include as many features as possible and to let the software functions to sort out which features are important or not. Therefore I may have included redundant features and features that do not have a clear relation to good or bad writing characteristics in the initial set of features.

Number of tokens, word tokens and number of lemmas are basic concepts used when defining features. The number of "tokens" includes all words, alphanumerical characters, paired delimiters and punctuation. The number of "word tokens" is defined as the total number of occurrences of all words and other alphanumerical characters in the text but punctuation and other delimiters are excluded. Repeated words, words with different grammatical forms etc. are also included. For example if the words "bil", "bilar", "bilarnas","bil" occur in the text, they are counted as one lemma and four word tokens. In the feature definitions there are abbreviations of different tags according to SUC (Gustafson-Capkova & Hartman, 2006). There are 46 features that divided in the following categories:

Features related to basic counters of words and sentences:

- Number of word tokens.
- Number of sentences
- Number of word tokens that have more than 6 characters divided by the number of word tokens.
- Number of tokens that have less than 4 characters divided by number of word tokens.
- Number of lemmas divided by the number of word tokens.
- Number of word tokens divided by number of sentences
- Number of non-initial CAPS words divided by number of sentences
- Number of characters in the essay divided by number of sentences

Features related to nonlinear combinations of different attributes:

- Fourth root of the number of word tokens. This is a non-linear transformation of the feature that has showed a positive correlation to scores according to (Page & Petersen, 1995).
- Word Variation Index (OVIX) as defined in (Smith & Jönsson, 2011)
- Nominal Ratio (NR) as defined in (Smith & Jönsson, 2011)

Features related to grammar controls:

- Number of grammar errors in essay divided by number of word tokens. The counting of grammar errors is based on the tool "GRANSKA" (Kann, 2005).

Features related to news, blog, word-list corpora:

- Number of essay word tokens classified as "news" divided by the number of word tokens. The newspaper corpora are used as examples of words that comply with the accepted written Swedish norms. The blogs corpora are used as examples of words that are used in a more relaxed way and they may not comply with the written standards. A word token is classified as "news" if it has a higher probability of occurring in the news corpora than in the blog corpora. If the word token has a higher probability of occurring in the blog corpora, it is classified as "blog". If a word token does not occur in any corpora, it is not classified at all.
- Number of essay tokens classified as "blog" divided by number of word tokens. The number of essay tokens classified as "blog" is defined as the number of "word tokens" classified as "blog"

plus the number of essay delimiters (punctuation and other delimiters) that also occur in blogs divided by the number of word tokens.

- Perplexity of essays POS bigrams using the POS bigram language model for news corpora (SVD/DN). The perplexity formula $PP(W) = \sqrt[N]{\prod_1^N \frac{1}{P(w_i \mid w_{i-1})}}$ as described in (Jurafski & Martin, 2009, pp. 129,131) is used. N is the total number of tokens (including end of sentence marker) in the essay.
- Number of word bigrams in essay with zero match in at least one news corpus divided by the total number of essay bigrams.
- Number of essay word tokens that occur in the corpus dictionary divided by number of word tokens.

Features related to complex word and complex bigram sets:

- Number of essay word tokens in the set of complex words divided by number of word tokens.
- Number of essay bigrams in the set of complex bigrams divided by the number of bigrams

Features related to POS:

- Number of infinitive marks (IE) divided by the number of word tokens
- Number of conjunctions (KN) divided by the number of word tokens
- Number of sub-junctions (SN) divided by number of word tokens
- Number of genitive forms (GEN) divided by number of nouns.
- Number of minor delimiters (MID) divided by number of word tokens.
- Number of major delimiters (MAD) divided by number of word tokens
- Number of particles (PL) divided by number of word tokens.
- Number of relative adverbs (HA) divided by number of word tokens
- Number of determiners (DT) divided by number of word tokens
- Number of interrogative relative determiners (HD) divided by number of word tokens
- Number of participles (PC) divided by number of word tokens
- Number of personal names (PM) divided by number of word tokens
- Number of adverbs (AB) divided by number of word tokens
- Number of paired delimiters PAD divided by numbers of word tokens
- Number of passive voice sentences divided by number of sentences
- Number of active voice sentences divided by number of sentences
- Number of possessive forms (PS) divided by number of word tokens
- Number of propositions (PP) divided by number of word tokens
- Number of adjectives (JJ) divided by number of word tokens
- Number of interrogative relative pronouns (HP) divided by number of word tokens
- Number of nouns (NN) divided by number of word tokens
- Number of foreign words (UO) divided by number of word tokens
- Number of counting words (RG) divided by number of word tokens
- Number of ordinal counting words (RO) divided by number of word tokens
- Number of pronouns (PN) divided by number of word tokens
- Number of pronouns in object form (OBJ) divided by number of pronouns
- Number of verbs (VB) divided by number of word tokens

3.4 Feature Ranking

The feature ranking procedure identifies how important each feature is for the classification. The ERT algorithm provided by Scikit-learn (Pedregosa, et al., 2011) is used to generate the feature ranking (feature importance). This algorithm takes into consideration the interaction of multiple features when identifying their rank (multivariate).

A univariate feature ranking based on ANOVA (F-score) is also calculated and it is only used as a reference to find out if there are major discrepancies among the highest ranking features.

3.5 Feature Selection

The list of ranked features (multivariate) is used as the input for feature selection algorithm together with the respective classifier in a grid-search process. The basic principle is to remove the feature with the lowest ranking, to perform a classification and to compute the cross validation error. This process continues in steps where the lowest ranked feature is removed in each step and the classification is performed using a smaller set of features until some minimum number of features is reached. The optimal set of features is the one that generates the lowest cross validation error (best quadratic weighted kappa). The features "nominal ratio" and "number of essay bigrams in the set of complex bigrams divided by the number of bigrams" are always included in the set of optimal features regardless their basic ranking.

3.6 Supervised Machine Learning Algorithms

A supervised machine learning algorithm processes essays' features in order to identify the corresponding scores. It is a key component in the AES system therefore the activities to select a suitable algorithm for our type of data must be carefully planned. As I described before there are many different kinds of supervised machine learning algorithms with different advantages and drawbacks. It is difficult to know a priori how they will perform with our classification tasks. The basic approach is to make an initial selection of machine learning algorithms, try them with our data and to select the best one. Our initial selection is broad and it covers supervised machine learning algorithms that have a successful track record in many different classification tasks. They are: 1) a linear based algorithm, 2) Support Vector Machines using non-linear kernels and 3) an ensemble method based on trees. The choice of software is Python that has different machine learning packages that are relatively easy to use. One advantage of the Python packages is that they accept the same data format as input and provide the same output format. The following Python packages are used:

- LDAC (Linear Discriminant Analysis Classifier provided by MPLY) (Albanese, Visintainer, Merler, Riccadonna, Jurman, & Furlanello, 2012)
- SVM with RBF kernel provided by Scikit-learn (Pedregosa, et al., 2011)
- SVM with polynomial kernel provided by Scikit-learn (Pedregosa, et al., 2011)
- Extremely randomized trees (ERT) provided by Scikit-learn (Pedregosa, et al., 2011)

3.6.1 Cross validation test

The cross validation test using ten folds is used to evaluate performance of the classifier. The training data (1500 essays) is divided into 10 folds with 150 essays in each fold. A cross validation procedure with ten folds implies that the classifier is trained and tested in 10 steps. In each step, 9 folds (1350 essays) are used for training and one fold (150 essays) is used for testing. The fold used for testing is different in each of the ten steps. A cross validation error is generated for each step. The final cross validation error for the classifier is the average of the cross validation errors generated in each step. The quadratic weighted kappa is the main measurement for the level of agreements and it is used as the basis for optimization procedures.

3.6.2 Linear Discriminant Analysis Classifier (LDAC)

The LDAC (Linear Discriminant Analysis Classifier) package is provided by MLPY (Albanese, Visintainer, Merler, Riccadonna, Jurman, & Furlanello, 2012). It is based on the linear discriminant Analysis classifier (fisher's discriminant). It is very simple because it does not contain any parameters that must be trimmed. LDAC results can be used as a base line classifier. The following procedure is used:

- Training data is formatted in a matrix according to the standard format (rows corresponds essays and columns corresponds to features). The target values are formatted as a row matrix.
- Perform feature ranking
- Remove the feature that has the lowest ranking.
- Create a LDAC classifier using the feature set. Perform the classification using the cross validation procedure. Compute the cross validation error based on quadratic weighted kappa.
- The activities are executed until the minimum number of features is reached. Select the feature set that provides the best results based on the quadratic weighted kappa values.

3.6.3 Support Vector Machine (SVM)

The SVM package provided by Scikit-learn (Pedregosa, et al., 2011) is used. The packages are based on LIBSVM (Chang & Lin, 2011). The basic recommendations for the training process for the SVM classifier are defined in (Hsu, Chang, & Lin, 2010). Figure 7 presents an overview of the main activities required to train SVM classifiers. The different activities are described in more detail further down.

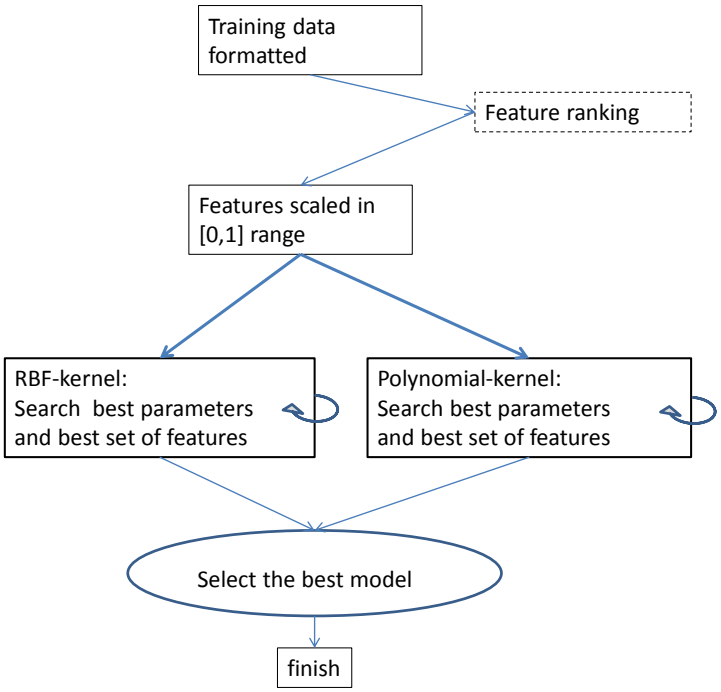

Figure 7 Overview of the grid search procedure for SVM with kernels

3.6.3.1 Data format

The training data (features, scores) are structured as a matrix where each row is an essay and each column is one feature according to the LIBSVM Format. The features here are in their basic form without any transformations. The target values (scores assigned by the blind raters) are stored in a row matrix (1xn) that matches the training data matrix.

3.6.3.2 Feature Ranking

See the aforementioned description.

3.6.3.3 Feature Scaling

SVM requires that all the features have values with a similar range. If there are different features with very different ranges, the features with the large values will dominate the results and the classification results will not be accurate. LIBSVM may also have problems with some mathematical processing if there are large discrepancies in the ranges for the features (Hsu, Chang, & Lin, 2010). Therefore SVM requires that the features are scaled. Recommended scaling ranges are [0, 1] or [-1, 1]. I will use the [0, 1] scaling. That is, each feature value is scaled using a [0, 1] scale relative to its column (each column represents a feature). The following formula is used for scaling a feature that has a value $b_{j,k}$ and belongs to certain column k:

$$b_{scal\ j,k} = \frac{b_{j,k} - min_k}{\max_k - min_k}$$

The result $b_{scal\ j,k}$ is a value between zero and one. The max and min values define the range for the values in this feature column.

3.6.3.4 RBF kernel grid search

RBF is a kernel relatively simple to tune and it is well suited for data sets where the number of data samples is larger than the number of features. RBF kernel is defined by the following formula:

$$K(x, x') = \exp(-\gamma \left| |x - x'| \right|^2)$$

There is a combination of two parameters that must be optimized. The C parameter that is a common SVM parameter must be optimized together with γ. The grid search with 10-fold cross validation is used to find the optimal combination of parameters. That is, different combinations of values for C and γ are tried and the best combination is selected (lowest validation error). The search procedures follow the recommendations in (Hsu, Chang, & Lin, 2010, p. 5) .This process may require a long computation time depending upon the number values for each parameter (c*g operations, assuming that I have c values for C and g values for γ). In order to decrease the number of operations at the same time that it provides accurate results, it is divided into two phases. In the first phase, a coarse scale for the values for the parameters is used in order to decrease the search scope. A recommendation is to have a sequence of growing values that grow in exponential steps. The following are the parameters and their ranges:

- The range of values for C is $2^{-5}, 2^{-3}, 2^{1}, \ldots . 2^{15}$.
- The range of values for γ is $2^{-15}, 2^{-13}, 2^{-11}, \ldots . 2^{3}$

There are 11 C values and 10 γ values that must be combined and checked and it will require the creation of 110 SVM classifiers using cross validation. When the best C and γ values are found using this coarse scale, a second phase is executed. During the second phase a finer scale around the "best values" is used to pinpoint the optimal values. The second phase is also based on cross validation. When the best parameters are identified, the classifier is trained with the complete training data in order to generate the final classifier. For each set of parameters a corresponding feature selection as described above is performed.

3.6.3.5 Polynomial kernel grid search

SVM with polynomial kernel supports the transformation of the features into a generic polynomial form ($x_i * x_k * x_j + x_i^2 * x_k^2 + x_i^4$ ...) where different features are moved into a higher dimension feature space and they are combined to each other in different constellations. The polynomial kernel is defined by the following formula:

$$K(x, x') = (\gamma * (x^T * x) + b)^d$$

This kernel is more complicated to tune because it contains three positive parameters (γ, b, d) that must be combined with C. That is, the number of operations increases very fast when the number of

values for each parameter increases. The number of operations is related to the product of the number of values for each parameter (a*r*d*c). Due to limitations on the computer power available, it may be necessary to perform some simplifications in the grid search.
A grid search procedure with cross validation is also used here. The process may be divided into two steps as described in the previous case. The following parameters must be optimized:

- The parameter d will have integer values in the range 3-5.
- The parameter C will have values $2^{-5}, 2^{-3}, 2^{1}, \ldots . . 2^{15}$.
- The parameter γ will have values in the range: $2^{-2},\ 2^{-1},\ 2^{0}, \ldots . . 2^{10}$.
- The parameter b may be defined in the following range: $0,\ 2^{0},\ 2^{1} \ldots . . 2^{10}$.

For each set of parameters a corresponding feature selection as described above is performed. In order to minimize the number of steps, the feature selection is limited to three sample matrices with 22, 30 and 40 features.

3.6.3.6 Select the best model

The classifier with its kernel, parameters and feature set that generates the largest quadratic weighted kappa value throughout the cross validation procedure is selected as the optimal solution.

3.6.4 Extremely Randomized Trees (ERT)

The ensemble method ERT defined in Scikit-learn (Pedregosa, et al., 2011) is used. This method creates a number of randomized tree based classifiers independently of each other. The final classification result is an average of the individual classifiers results (probabilities). The following parameters are optimized with a grid search procedure and cross validation:

- The parameter "number of trees" defines the number of trees that are created in the forest. The risk for overfitting is considered small therefore the more trees used the better results until a ceiling value is reached. After the ceiling, there is no significant improvement in the results if the number of trees is increased. Of course, computer resources are another issue when defining the maximum number of trees. Even if it the risk for overfitting is small, it is wise to try a range of values for the number of trees. The range used is 200, 300, 400 and 600 trees.
- The parameter "number of features" that defines the number of features that are randomly selected in order to decide how to split a node in the tree. The value of this parameter affects the bias/variance aspects of the classifier. The range used is (3, 4, 5, 6 and 7). The lower the value the higher randomness in the creation of trees.
- The parameter "bootstrapping" defines how the data is generated. If the parameter is set to "true" a resampling procedure with replacement (bootstrapping) is used when creating the trees. It increases the randomness and the bias. If the parameter is set to "false", all the samples are used when creating the trees. In this case, the bias is decreased.
- Other parameters are used with default values.

The following grid search procedure is followed:

- Training data is formatted in a matrix according to the standard format (rows corresponds essays and columns corresponds to features). The input is the X-matrix and the corresponding Y-matrix with the target values.
- Perform a feature ranking
- Select the parameters and create the classifier based on ERT by executing the following steps:
  - Remove the feature with lowest ranking
  - Test the results using 10-fold cross validation and calculate the quadratic weighted kappa values

- The activities are repeated until all the parameters and feature combinations are covered. The combination of parameters and features that produces the best results is selected for this classifier.

The ERT results are random therefore small variations in the results may occur when running ERT several times with the same parameters.

3.6.5 Statistics on cross validation tests

The following statistics are computed using the cross validation results for each classifier:

- Quadratic weighted kappa is the main measurement result and it is used to optimize the solutions.
- Exact score agreement accuracy is the number of essays where the classifier and blind raters had full agreement divided by the total number of essays.
- Adjacent plus exact agreement accuracy is the number of essays where the classifier's scores and the blind raters' scores have a difference of 0 (exact agreement) or $\pm$ 1 (adjacent agreements). However, the scores with value IG must always have exact agreement. The other scores may have exact and adjacent agreements. This counter is divided by the total number of essays.
- A "t-test for related data" is performed in order to compare the classifier's scores and the blind ratters' scores. The null hypothesis is that there are no differences between the two samples (both set of scores are different samples from the same population). If the p value is less than 0.01(two tailed), the null hypothesis can be rejected.
- The Spearman Rank Correlation test (non-parametric) and the corresponding p-value are computed for the scores generated by a classifier and the corresponding blind raters' scores. The null hypothesis is that the two set of scores are not related. The null hypothesis is rejected if the p-value is less than 0.01.

3.7 Summary of Software functions

Stagger (Östling, 2012) is the basic software function used for pre-processing most of the text data used by the creation of features. Stagger tokenizes Swedish text and annotates each token with parts of speech tags and lemma.

Python 2.6 is used as the basic programming language for the software function developed during this project. There are software modules that are handling the following functions:

- Initial support functions for pre-processing the essay data
- Software functions for feature components
  - Essays and news bigram for POS and words
  - Grammar check
  - Classification of words as news or blogs
  - Dictionary check
- Software functions for complete feature creation
- Software for classification based on supervised machine learning
  - LDAC
  - SVM RBF kernel
  - SVM polynomial kernel
  - ERT(Extremely Randomized Trees)
- Software for post processing
- Calculation and presentation of statistics

4 Results

The results for feature ranking, LDAC, SVM-RBF, SVM-POL and ERT classifiers are presented in the following chapters.

4.1 Feature ranking

The ERT classifier generates a statistical measurement for how important the different features are for the classification. The algorithm takes into consideration the interaction of different feature (multivariate feature ranking). The following is a short description of the principle used for feature ranking. Features are nodes in a tree and the depth of each feature defines its importance for the classification. That is, if a feature is near the root it has more impact in the classification than the features below it. An intuitive explanation is that if I disturb a feature near the root (add an error to the feature value), it will have a major impact in the tree structure and classification results (almost all the nodes below are impacted). If I disturb a feature near a leaf, the impact on the classification is smaller. The ERT provided by Scikit-learn (Pedregosa, et al., 2011) calculates the proportion of training data that is processed by a node as measurement for the depth of the node and as a measurement for feature importance (a value between 0 and 1). This value is averaged among all the trees. The larger the value the more important is the feature.

Table 5 shows the multivariate feature ranking produced by the ERT algorithm. The features are ordered in descending order of importance (feature 1 is the most important and feature 46 is less important). This table will be used as the reference for feature ranking throughout this document.

Table 5 Multivariate Feature ranking based on extremely randomized trees

Feature rank	Feature names
1	fourth root of # of word tokens (fourth root of number of word tokens)
2	# of word tokens
3	# of sentences
4	perplexity for POS bigrams in news corpora
5	# of essay word bigrams with zero match in news bigrams /# of bigrams
6	wOrd Variation Index (OVIX)
7	# of grammar errors/# of word tokens
8	# of essay word tokens classified as “news”/# of word tokens
9	# of word tokens longer than 6 chars/# of word tokens
10	# of conjunctions (KN)/# of word tokens
11	# of infinitive mark (IE)/# of word tokens
12	# of genitives/# of nouns
13	# of minor delimiters (MID)/# of word tokens
14	# lemmas/# of word tokens
15	# of particles (PL)/# of word tokens
16	# of determiners (DT)/# of word tokens
17	# of relative adverbs (HA)/# of word tokens
18	# of participles forms (PC) /# of word tokens
19	# of tokens shorter than 4 char/# of word tokens
20	# of personal names (PM)/# of word tokens
21	# of adjectives (JJ)/# of word tokens
22	# of possessives (PS)/# of word tokens
23	# of passive voice/# of sentences

24	# of prepositions (PP)/# of word tokens
25	# of interrog. relative pronouns (HP)/# of word tokens
26	# of complex bigrams in essay/# of bigrams
27	# of numeral words (RG)/# of word tokens
28	# of words in word list/# of word tokens
29	# of pronouns (PN)/# of word tokens
30	# of sub-junctions (SN)/# of word tokens
31	# of nouns (NN)/# of word tokens
32	# of essay tokens classified as “blog “/# of word tokens
33	# of pronouns in object-form(OBJ) /# of pronouns
34	# of non-initial CAPS words/# of sentences
35	# of foreign words (UO)/# of word tokens
36	nominal ratio (NR)
37	# of adverbs (AB)/# of word tokens
38	# of complex words/# of word tokens
39	# of verbs (VB)/# of word tokens
40	# of paired delimiters (PAD)/# of word tokens
41	# of active voice/# of sentences
42	# characters/# of sentences
43	# of word tokens/# of sentences
44	# of major delimiters (MAD)/# of word tokens
45	# of ordinal count words (RO)/# of word tokens
46	# of interrogative determiners (HD)/# of word tokens

As a comparison, the appendix A shows the results for a univariate feature ranking based on F-score (ANOVA) where the importance of each feature is measured without consideration of the other features.

4.2 AES results

The results for the AES systems based on LDAC, SVM-RBF, SVM-POL and ERT are presented in this chapter. The results for each AES system are based on grid-search using tenfold cross validation tests that define the optimal configuration both regarding number of features and parameters. The reserved test data was not used because none of the AES systems met the required target (quadratic weighted kappa larger than 0.7). For each AES, the following results are presented: Number of features, optimal parameters, Summary of agreements, Confusion matrices with absolute and relative agreements, and basic statistics with t-test and Spearman rank correlation.

4.2.1 AES based on LDAC

The results for the AES based on LDAC are based on the 20 most relevant features. There are no parameters optimized in LDAC. Table 6 shows an overview for the agreements between the LDAC results and the blind raters.

Table 6 Summary of agreements between blind raters and LDAC scores using 20 features

	LDAC summary	
	Quadratic kappa	Average Accuracy
Exact agreement	0.4751	57.3%
Adjacent agreement	-	83.4%

Table 7 Confusion matrix with absolute values for number of blind scores versus LDAC scores

	LDAC score predictions					Row total
Blind scores		IG	G	VG	MVG	
	IG	87	135	15	0	237
	G	47	611	79	14	751
	VG	1	236	137	25	399
	MVG	0	37	52	24	113
Column total		135	1019	283	63	1500

The confusion matrix in table 7 describes the absolute values for agreements and disagreement between the LDAC and the blind scores. The value in the cell (i, j) indicates the number of essays with blind score "i" that were classified by LDAC with the score "j". For example, the cell defined by the row "IG" and the column "VG" contains the value 15. It indicates that there are 15 essays that are scored with "IG" by the blind raters and with "VG" by the LDAC classifier. The values in diagonal cells (i, i) indicate the exact agreements for the ith grade. The ith row total describes the total number of essays with blind score "i". The ith column total describes the total number of essays with LDAC score "i".

Table 8 Normalized confusion matrix showing percent of agreements and disagreements

	LDAC score predictions					
Blind scores		IG	G	VG	MVG	
	IG	36.7%	57.0%	6.3%	0%	100%
	G	6.2%	81.4%	10.5%	1.9%	100%
	VG	0.3%	59.1%	34.3%	6.3%	100%
	MVG	0%	32.7%	46.0%	21.2%	100%

Table 8 shows the normalized confusion matrix where each row is normalized by the total number of essays that have the corresponding blind score. The value in the cell (i, j) indicates the proportion of essays with blind scores "i" that were assigned score "j" by the LDAC classifier. The cell defined by the row "G" and the column "VG" has the value 10.5%. It indicates that 10.5% of all the essays that have the blind score "G" were classified as "VG" by the LDAC classifier. A value in the diagonal (dark shaded) indicates the proportion of essays with the corresponding score where the blind raters and the LDAC classifier are in exact agreement.

The t-test for related data (LDAC scores and blind scores) shows that there is a significant difference between the essay scores generated by the AES based on LDAC (mean=1.183, standard deviation =0.642) and the scores generated by the blind raters (mean=1.259, standard deviation=0.811). The results suggest that the blind raters tend to assign higher scores than the AES using LDAC (on average). This is based on t-statistics value of -3.948 and a corresponding p-value that is much less than 0.01(two tailed). The Spearman Rank Correlation test shows that there is a positive correlation between the scores generated by AES based on LDAC and the scores generated by blind raters. This correlation is significant. The rho-coefficient is 0.61 and the p value is much less than 0.01.

4.2.2 AES based on SVM-RBF

The results for the AES based on SVM-RBF are based on the 22 most relevant features and the following optimal parameters:

- C-value = 5000
- Gamma-value = 0.756

The agreement results for the AES based on SVM-RBF are summarized in tables 9, 10 and 11.

Table 9 Summary of agreements between blind raters and SVM-RBF

	SVM-RBF	
	Quadratic kappa	Average Accuracy
Exact agreement	0.4458	57.5%
Adjacent agreement	-	84.2%

Table 10 Confusion matrix with absolute values for number of blind scores versus SVM-RBF scores

	SVM-RBF score predictions					Row total
Blind scores		IG	G	VG	MVG	
	IG	70	153	14	0	237
	G	24	632	93	2	751
	VG	4	230	150	15	399
	MVG	0	39	63	11	113
Column total		98	1054	320	28	1500

Table 11 Normalized confusion matrix showing percent of agreements and disagreements

	SVM-RBF score predictions					
Blind scores		IG	G	VG	MVG	
	IG	29.5%	64.5%	5.9%	0%	100%
	G	3.2%	84.1%	12.4%	0.3%	100%
	VG	1.0%	57.6%	37.6%	3.8%	100%
	MVG	0%	34.5%	55.8%	9.7%	100%

The t-test for related data (SVM-RBF scores and blind scores) shows that there is a significant difference between the mean essay scores generated by the AES based on SVM-RBF (mean=1.185, standard deviation=0.564) and the mean scores generated by the blind raters (mean=1.259, standard deviation=0.811). The results suggest that the blind raters tend to assign higher scores than the AES using SVM-RBF. This is based on the t-statistics value of -3.873 and a corresponding p-value that is much less than 0.01(two tailed). The Spearman Rank Correlation test shows that there is a positive correlation between the SVM-RBF scores and the blind raters that is significant. The rho-coefficient is 0.603 and the p value is much less than 0.01.

4.2.3 AES based on SVM-POL

The results for the AES based on SVM-POL use the 22 most relevant features and the following parameters:

- Degree = 5
- C-Value = 20
- Gamma-value= 1
- Constant coefficient= 0

The agreement results for the AES based on SVM-POL are described in tables 12, 13 and 14.

Table 12 Summary of agreements between blind raters and SVM-POL

	SVM-POL	
	Quadratic kappa	Average Accuracy
Exact agreement	0.4458	57.5%
Adjacent agreement	-	84.2%

Table 13 Confusion matrix with absolute values for number of blind scores versus SVM-POL scores

	SVM-POL score predictions					Row total
Blind scores		IG	G	VG	MVG	
	IG	70	149	18	0	237
	G	31	614	98	8	751
	VG	4	227	150	18	399
	MVG	3	43	51	16	113
Column total		108	1033	317	42	1500

Table 14 Normalized confusion matrix showing percent of agreements and disagreements

	SVM-POL score predictions					
Blind scores		IG	G	VG	MVG	
	IG	29.5%	62.9%	7.6%	0%	100%
	G	4.1%	81.8%	13.0%	1.0%	100%
	VG	1.0%	56.9%	37.6%	4.5%	100%
	MVG	2.7%	38.1%	45.1%	14.2%	100%

The t-test for related data (SVM-POL and blind scores) shows that there is a significant difference between the mean essay scores generated by the AES based on SVM-POL (mean=1.195, standard deviation=0.597) and the mean scores generated by the blind raters (mean=1.259, standard deviation=0.811). The results suggest that the blind raters tend to assign higher scores than the AES using SVM-POL. This is based on the t-statistics value of -3.188 and a corresponding p-value that is much less than 0.01(two tailed). The Spearman Rank Correlation test shows that there is a significant positive correlation between the SVM-POL scores and the blind raters. The rho-coefficient is 0.570 and the p value is much less than 0.01.

4.2.4 AES based on ERT

The results for the AES based on ERT are based on the 21 most relevant features and the following parameter values:

- Number of trees = 300
- Number of features to split a node = 6
- Bootstrap = False

The agreement results for the AES based on ERT are described in tables 15, 16, 17.

Table 15 Summary of agreements between blind raters and ERT scores

	ERT	
	Quadratic kappa	Average Accuracy
Exact agreement	0.4577	57.2%
Adjacent agreement	-	85.0%

Table 16 Confusion matrix with absolute values for number of blind scores versus ERT

	ERT score predictions					Row total
Blind scores		IG	G	VG	MVG	
	IG	67	158	12	0	237
	G	20	630	100	1	751
	VG	0	237	149	13	399
	MVG	0	34	67	12	113
Column total		87	1059	328	26	1500

Table 17 Normalized confusion matrix showing percent of agreements and disagreements

	ERT score predictions					
Blind scores		IG	G	VG	MVG	
	IG	28.3%	66.7%	5.1%	0%	100%
	G	2.7%	83.9%	13.3%	0.1%	100%
	VG	0%	59.4%	37.3%	3.3%	100%
	MVG	0%	30.1%	59.3%	10.6%	100%

The t-test for related data (ERT and blind scores) shows that there is a significant difference between the essay scores generated by the AES based extremely randomized trees (mean=1.195, standard deviation=0.554) and the scores generated by the blind raters (mean=1.259, standard deviation=0.811). The results suggest that the blind raters tend to assign higher scores than the AES using ERT. This is based on the t-statistics value of -3.407 and a corresponding p-value that is much less than 0.01(two tailed). The Spearman Rank Correlation test shows that there is a positive correlation between the ERT's scores and the blind raters. The rho-coefficient is 0.613 and the p value is much less than 0.01.

5 Discussion

5.1 Methods

5.1.1 Quality of the essay data

A simple cleaning procedure was executed to remove the digitalization/transcription tags for the essays. There were a number of essays that had missing words, missing sentences and other transcription/digitalization errors. However, I did not perform a more extensive analysis of the quality of the text in digital form. A more careful evaluation of the input data should be an initial step. This would include gathering transcription error statistics and determining criteria for making corrections in the essays and accepting or removing essays from the data (outliers). The decision to correct transcription errors is not trivial because it may be difficult to define when there is a transcription error or not.

An important characteristic for the training data is that there is a very low agreement between the teachers and blind raters with a quadratic weighted kappa value of 0.392 and average accuracy of 45.8%. Even though there are written guidelines for scoring essays, the human raters have a great deal of freedom when applying the scoring criteria because there is no external evaluation (Hinnerich, Höglin, & Johannesson, 2011). The lack of common scoring criteria that is really used by the raters is an important explanation for the disagreement between teachers and blind raters. This is also an issue for the training material based on scores assigned by the blind raters. The blind raters must also have had a great deal of discretion when applying the scoring guidelines. They have the same background and experiences as the teachers with the differences that they were blind raters (one of many biases removed) and there were fewer blind raters (less diversity in scoring criteria is expected). Therefore the conclusion is that the training data based on blind raters also has weak scoring criteria in it. There is no information to evaluate the quality of the blind scores and we cannot refer to it as a golden standard. This lack of common scoring criteria in the training data is a difficulty when creating and evaluating the classifiers and there is no easy solution on short term.

An ideal scenario is to have the essays in the training data sample (representative for the population) corrected by two groups of well qualified human raters. A training phase is required where the human raters study, discuss and agree upon the common scoring procedures. The essays that are scored with very different scores are reviewed by a third group in order to decide the final score. The human raters' scores should have a substantial agreement (quadratic kappa value above 0.7). In this

case we could expect a training data with more common scoring substance in it. A classifier trained on this data will provide more reliable and more accurate results and it should be fitted for scoring essays in large scale. Of course, the drawback is the cost to get this well scored training data.

One alternative to create training data with well-defined scoring criteria is to select the data where the blind raters and teachers have agreed on the scores. We have made some experiments with this training data and the results were much better as expected. However, the main draw back with this alternative is that we do not have training data that is representative for the population. We have a subset of the training data with some special characteristics that imply coincidence of scores (coincident training data). The classifier trained with the "coincident training data" may achieve very good results with the cross validation results. However, the "coincident classifier" may not generalize well. We should expect higher generalization errors when using this "coincident classifier" to score new essays (new essays and training data from different populations).

5.1.2 Features

A feature ranking was performed with the purpose to get a better understanding about how the different features are related to the classification task. The feature ranking was also used for feature selection procedures during the classification.

Important limitations in the features in the current project are: the lack of evaluation of the semantic dimension and topic control of the essays, the lack of detection of outliers and lack of protection against cheating the system. There are no features in the semantic dimension that control if the essays are covering the required topics. For example, if the topic of an essay is "cellular telephones" and the student writes a very good essay about "vegetarian food", the student may get a poor grade by the human ratters even if the actual writing may be very good. The AES system may give a good or bad score only based on the grammar and/or count features. Furthermore, the lack of features related to the semantic dimension adds noise in the training data. Two essays in the training data may have similar feature values but they may have completely different scores because the human raters take also into consideration the semantic dimension. The same set of features' values mapping to different scores is a typical example of noise data for the classifiers. All commercial AES systems have functions to analyze the semantic dimensions and topic coverage (Dikli, 2006). Some solutions are based on Latent Semantic Analysis (LSA).

There were no attempts to identify and to remove outliers for each score. Different outlier detection approaches may be used depending upon the distribution characteristics for the training data. One approach is to use a one-class classification algorithm to identify outliers for a certain score (data with one class that corresponds to score X). Outliers have a relation to the choice of features. The definition, identification and removal of outliers are not trivial. Therefore the essays identified as outliers should be carefully inspected before deciding if the essay should be removed or not. A typical outlier may be an essay with many missing words and sentences. For example an essay with MVG that contains many missing words and sentences is a valid outlier that could be removed.

There were no attempts to measure and to analyze how easy it is to cheat the system in order to get a higher score than expected. The fact that most important features are related to counters and there is no control about topics can be used to cheat the system. A completely meaningless text with many correct words, correct bigrams and sentences with very simple structures may get a good score by an AES system. One example of a meaningless text may be a sequence of simple sentences: "Jag åker till Dalarna.", "Jag köpte en bil.", "Du bakar kakor.", etc. If the counters/features have the correct values the AES system will assign a good score. A human rater would have discarded such a text with the score IG right away because it is not an essay. Possible solutions are very much related to the choice of features and how the classifiers are using the features therefore there is no general solution. A first step is to analyze and to understand the characteristics of the texts that systematically trigger the AES system to give scores that are much higher (or lower) than expected when comparing with human raters. When this vulnerability is understood, it is possible to propose solutions. Depending upon the use of the AES, this issue may be more or less important. If the AES is used for real scoring, this issue becomes very important because it affects the validity of the scores generated by the AES. If the AES is used only as a quality control tool, this issue is less important because the human raters are always the main raters.

The improvement of the features is an important area for future work. Here is a summary of areas for improvements that should be further investigated:

- There is a feature that tries to identify the use of more complex sentences' structures by defining a list of words/bigrams used in more complex sentences. Improve the quality and/quantity of this list.
- Google's N-gram corpus could be used to check essay's trigrams (bigrams) in order to define new features.
- Add features related to the semantic dimension and topic control of the essays.
- Identify and remove outliers per score.
- Add some quality control features to avoid attempts to cheat the system.
- Investigate better criteria to define essay words as "news words" or "blog words".

5.1.3 Classifiers

Four different classifiers based on different principles were used during this project. The creation of the classifiers was based on grid search in order to optimize parameters and number of features. The grid searches for tree classifiers and LDAC have few parameters and the execution time is short. The grid searches for SVM (POL, RBF) have a large number of parameters and the execution time becomes much longer. If the execution time is an issue, the grid search data can be broken down in independent segments. The grid search procedure can then run the segments in different computers at the same time. The results from the different computers are compiled for a final result.

5.2 Results

5.2.1 Feature Ranking

The bar chart in figure 8summarizes the multivariate feature ranking. For each feature there is the corresponding importance values and standard deviation produced by the extremely randomized trees algorithm. The feature with rank number 1 is the most important feature with the highest importance value. The feature number 46 is the least important feature. The feature numbers correspond to the names defined in table 5.

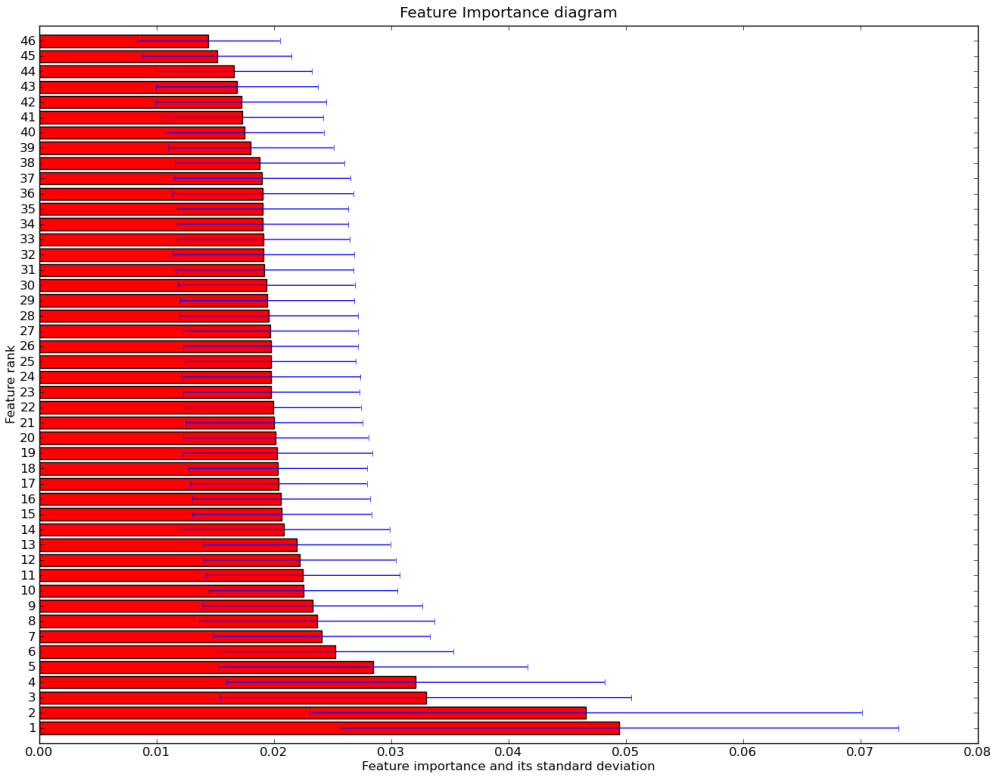

Figure 8 Summary of multivariate feature ranking

When analyzing the levels of feature importance, we can see that the two first features (fourth root of number of word tokens and number of word tokens) are clearly well ahead of all other features. Both features are absolute counters related to essay length. The third feature (number of sentences) is also related to essay length. These results are in agreement with the (early) work to score English language essays (Page & Petersen, 1995). One simple explanation for this relationship is that the American and Swedish students that have better writing skills are also able to express and to develop ideas with more words in the limited time period for the essay test.

The fourth and fifth most important features (Perplexity for POS bigrams in news corpora, number of essay word bigrams with zero match in news bigrams/number of bigrams) compare the essays with newspapers (examples of good writing). Those features have a more intuitive relationship to writing skills. OVIX and number of grammar errors are other features related to writing characteristics that have higher rank.

The features ranked from sixth to thirty eighth have more or less the same importance value for the classification. When we compare the multivariate feature ranking with the univariate feature ranking (F-score ANOVA in Appendix A), the five highest ranked features are the same in both ranking calculations. The five highest ranked features in the univariate ranking have very high F-scores which are significant with a p-value much less than 0.01.

5.2.1 Classification Results

All the classifiers generated predictions with quadratic weighted kappa results between 0.41 and 0.47. LDAC classifier hade the best results and the ERT classifier was the second best.

The feature selection procedure has a limited impact in the classification results. For example the quadratic weighted kappa values for the LDAC classification varies in between 0.45 to 0.47 depending upon the number of features selected. This is an expected behavior because the number of samples in the training data is much larger than the number of features therefore there is a small risk for overfitting.

Table 18 shows that the two best classifiers have much better agreement with the blind raters than the agreement between teachers and blind raters.

Table 18 Summary of measurements for agreements between the two best classifiers and blind raters

	Type of agreements		
	Teachers - blind raters	LDAC - blind raters	ERT - blind raters
Quadratic Kappa for exact agreement	0.391	0.475	0.457
Average accuracy for exact agreement	45.8%	57.3%	57.2%
Average accuracy for adjacent agreement	78.7%	83.4%	85.0%

Table 19 describes the relative agreements per score. When we break down the agreement figures for each score, we see a clear relationship between the number of training examples per score and the performance of the classifiers. The classifiers have an excellent performance when classifying essays with score G that correspond to the majority of the training data (classifiers have more than 80% agreement with blind raters). But the classifiers have an agreement with blind raters below 50% for all other scores. When comparing the agreement results with human raters, the classifiers do better for the essays with scores G and IG. The human raters have a much better agreement for the essays with score MVG. However, it should be noted that the human raters have a more uniform distribution of the scores but the level of agreement barely reaches 50% in any score. The results show that the classifiers are not able to handle the scores with few essays in the training data ("tail scores"). A requirement for the classifiers should be to perform better than 50% (chance) in any scores. This area must be further studied in future projects.

Table 19 Summary of average accuracy for exact agreements per score

	IG	G	VG	MVG
# of blind raters 'scores (training data)	237	751	399	113
Teacher-blind raters agreement	27.85%	50.73%	49.87%	36.28%
LDAC-blind raters agreement	36.70%	81.36%	34.30%	21.24%
ERT-blind raters agreement	28.69%	83.89%	38.10%	15.04%

The t-test for paired dependent data indicates that all AES systems are assigning scores with mean scores that are lower than the blind raters (on average). The differences in the mean scores are statistically significant (p-values much less than 0.01 two tailed).

The Spearman rank test results indicate that there is a positive correlation between the AES systems' scores and the blind raters' scores. This correlation is statistically significant.

When evaluating AES the quadratic weighted kappa value is one measurement used to describe the level of agreement between the scores assigned by the AES system and human raters and the threshold of 0.7 is defined as the basic acceptance level (Williamson, Xi, & Breyer, A Framework for Evaluation and use of Automated Scoring, 2012). If the system fulfills this criterion a further analysis taking into consideration other criteria is required. Our AES systems do not fulfill the requirement of a quadratic weighted kappa value larger than 0.7.

A reasonable assumption is that the essay data is not linearly separable but the linear classifier (LDAC) is still performing slightly better than the non-linear classifiers. An explanation is that the features are weak predictors of the target function and the selected classifiers are not able to identify relevant relationships between the features and the target scores. All classifiers have relatively high bias. However, the "simplest" classifier (LDAC) has a lower variance than the non-linear classifiers. Therefore LDAC is able to generalize better and to perform slightly better than the other classifiers

taking into consideration the features and the amount of training data used. Another factor that may have helped the LDAC performance is that the essay scores have a distribution that resembles the normal distribution.

I have used both linear and non-linear standard classifiers. I used grid search to optimize the classifiers regarding basic parameters and features. But the results do not meet the minimum requirements for an AES. Here is a summary of possible improvements:

- Based on the previous discussions, improvements are needed in the training data to reduce noise. There are some aspects of the features that can be improved as well.
- We can try to find classifiers that are better at handling both majority and minority classes. One example is to use more advanced ensemble methods with classifiers that have different niches. Some classifiers may be better for classifying essays with certain scores than other classifiers. The results from the different classifiers can be combined in an optimal way. The classifiers in the ensemble could be heterogeneous (SVM with different kernels, linear discriminant analysis, tree based, etc.). There are many different ways to handle the combination of the results in the ensemble; weighted voting, stacking and mixture of experts are some examples.
- Another consideration is to use the large amount of text available in the web (blogs, newspaper articles, books, etc.) in order to increase the data labeled with IG, MVG in the training data to improve the classification accuracy of these scores. There are techniques that combine unsupervised learning with supervised learning (semi-supervised learning) to enable the use of unlabeled and labeled data to train the classifiers (Wemmert, Forestier, & Derivaux, 2009). An important question here is to find unlabeled texts that are really relevant for to the essays.

5.3 Introduction of AES and related issues

Based on the aforementioned criticisms about AES and those of Landauer et al (Landauer, Laham, & Foltz, 2003) I would like to emphasize two important areas related to the introduction and further development of AES into schools. One area that requires attention is a strategy for introducing AES in order to avoid conflicts with teachers. It is important to define a strategy where the AES may support the existing scoring procedures. A first step could be to use the AES system as a quality assurance system. That is, the teachers still score the essays as usual; the AES system also scores the same essays. If the scores assigned by the teachers differ by more than x% from the AES system, a red flag could be raised to trigger attention. As stated before, the AES must fulfill strict performance requirements in order to be the only source of scores. However, if the AES is used to generate statistics to support quality assurance activities, the requirements on AES could be relaxed during an initial phase. This strategy would provide valuable feedback and it would support stepwise AES improvements.

Another area that must be addressed is the dialogue with experts. Linguists, teachers and experts in the field want to see a clear link between features that they accept as characteristics for good or bad essays, and they want to understand how they are used by AES. We should try to use as many features as possible from the experts if the features are measurable. We are able to predict which features are important for the classifier. However, it is very difficult to show and to explain a clear link when using AES based on supervised machine learning. The AES classifier uses the training data and their features to create a model that predicts how the "human raters" used during the training phase would have scored the new essays. Depending upon the type of classifier this model may very complicated and difficult to explain. For example, the features may be lifted to a higher dimensional space and combined in many different non-linear combinations in order to produce a classifier with good performance. The classifier may generate scores that agree with human raters but the relationship between features and scores cannot be easily explained and they may not match the expectations of the experts. One simple example is the feature "fourth root of number of word tokens" that is ranked very high in the classification process. How can we explain the use of this feature? We may talk about catching some non-linear relations between characteristics in the essays, etc. But this explanation is

not intuitive and many experts may not accept it. This lack of understanding and lack of trust will be a constant issue in the discussions about using AES. There is an intense discussion about those issues in the scientific communities dealing with AES.

5.4 Using fewer classes

The current essays contain four classes that correspond to the scores (IG, G, VG, and MVG). Can the number of classes be reduced in order to get more accurate statistics about the essays' scores? This is a relevant question if AES is only used to provide statistic support for the school administration, The following are examples of how the scores may be combined into fewer classes:

- Two classes are created by separating "IG" from the rest. The two classes are: "IG" and "G+VG+MVG" (IG against all).
- Two classes are created by separating "MVG" from the rest. The two classes are: "MVG" and "IG+G+VG".(MVG against all)
- Three classes are created by dividing the scores into three groups: One class that correspond to failed essays (IG), one class that correspond to average essays (G+VG) and one class that corresponds to the excellent essays(MVG)

It is relatively simple to group the essays into fewer classes that are input to create the classifiers. In general, it will be easier for classifiers to learn fewer classes. I also expect improved accuracy but the most important factors are still the features and the training data. What are the characteristics for the training data in those cases? In all cases IG and MVG will have very few examples in comparison with the rest (typical for this population) This situation is more accentuated in the cases of two classes When the data have few examples of some classes, current solutions do not work well. If the current classifiers (LDAC, ERT, and SVM) are trained with the training data with fewer classes, the agreement results for the majority classes would probably be excellent and the results for the minority classes would be very poor. This issue must be resolved before we may have a useful solution.

6 Conclusions

This project is a first attempt to develop and evaluate AES systems for scoring Swedish high school essays. The systems were based on standard supervised machine learning software, i.e., LDAC, SVM with RBF kernel, SVM with polynomial kernel and Extremely Randomized Trees.

The agreement between blind raters' scores and AES scores was compared to agreement between blind raters' and teacher scores. The AES systems showed better average agreement with blind raters than the agreement between teachers and blind raters. The AES based on LDAC software had the best results with a quadratic weighted kappa of 0.475. However, the AES systems do not meet the requirement that scores generated by AES and human raters should have an inter-rater agreement with a quadratic weighted kappa larger than 0.7 (as defined by ETS). A weakness of these AES systems is their poor accuracy in predicting the scores that are less common, i.e., IG and MVG. A reason for the weak performance of the AES systems is the poor quality of the training data, i.e., there is low scoring consensus between teachers and blind raters. The quadratic weighted kappa value for their inter-rater agreement is 0.391.

Regarding future development we should keep in mind that there are very good results when using AES systems for English essays in the US. Therefore I am sure that it is possible to develop AES systems that can be used for scoring essays in Swedish with good results as well. This project has identified and discussed some areas for improvement. To develop better AES systems, better quality training data are needed. Improvements in feature definition and classifier algorithms are also needed.

In regards to introducing AES into the education system, an initial first step could be to use the AES as a quality assurance tool to help school authorities evaluate scores e.g., from different teachers and schools. This introduction strategy would demand less strict performance requirements. It would also provide feedback to support the development of better AES systems.

7 References

ETS home. (2012). Retrieved April 3, 2012, from http://www.ets.org

TOEFL IBT Test scores. (2013). Retrieved January 02, 2013, from TOEFL ETS: http://www.ets.org/Media/Tests/TOEFL/pdf/Writing_Rubrics.pdf

Abu-Mustafa, Y. S., Magdon-Ismail, M., & Lin, H.-T. (2012). *Learning from Data* (1 ed.). AMLbook.

Albanese, D., Visintainer, R., Merler, S., Riccadonna, S., Jurman, G., & Furlanello, C. (2012). mlpy:machine learning Python. arXiv:1202.6548v2[cs.MS]. Retrieved from http://mlpy.sourceforge.net/

Attali , Y., & Burstein, J. C. (2006). Automated Essay Scoring wit e-rater V.2. *THe Journal of Technology, Learning and Assessment (JTLA), 4*(3).

Ben-Hur, A., & Weston, J. (2010). A user's guide to support vector machines. *Methods in Molecular Biology*, 223-239.

Ben-Simon, A., & Bennett, R. E. (2007). Toward more Substantively Meaningful Automated Essay scoring. *The Journal of Technology, Learning and Assessment (JTLA), 6*(1).

Borin, L., Forsbeg, M., & Lönngren, L. (2012). *SALDO*. Hämtat från http://spraakbanken.gu.se/resurs/saldo 2012

Breiman, L. (1996). Bagging Predictors. *Machine Learning, 24*(2), 123-140.

Burstein, J., Kukich, K., Wolff, S., Lu, C., & Chodorow, M. (1998). Computer Analysis of Essays. San Diego, CA.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST), 2*(3), 27.

Chodorow, M., & Burstein, J. C. (2004). *Beyond Essay Length: Evaluating e-rater®'s Performance on TOEFL® Essays.* Educational Testing Service.

Cizek, G. J. (2012). Defining and Distinguishing Validity: Interpretations of Score Meaning and Justifications of Test Use. *Psychologycal Methods, 17*(1), 31-43.

Cizek, G. J., & Page, B. A. (2002). The Concept of Reliability in the context of automated Essay Scoring. In M. D. Shermis, & J. Burstein, *Automated essay scoring: a cross-disciplinary perspective* (pp. 125-135). Routledge.

Dikli, S. (2006). An Overview of Automated Scoring of Essays. *The Journal of Technology, Learning, and Assessment, 5*(1).

Dreyfus, G., & Guyon, I. (2006). Assessment Methods. In I. Guyon, S. Gunn, M. Nikravesh, & Z. A. Lofti (Eds), *Feature Extraction, Foundations and Applications* (pp. 65-88). Berlin, Heidelberg: Springer-Verlag.

Einarsson, J. (1978). *Talad och Skriven svenska.* Lund: StudentLiteratur.

Geurts, P., Damien, E., & Wehenkel, L. (2006). Extremely Randomized Trees. *Machine Learning, 36*(1), 3-42.

Grandvalet, Y. (2004). Bagging Equalizes Influence. *Machine Learning, 55*(3), 251-270.

Gustafson-Capkova, S., & Hartman, B. (2006). *Manual of the Stockholm Umeå Corpus version 2.0.*

Guyon, I., & Elisseeff, A. (2006). An Introduction to Feature Extraction. In I. Guyon, S. Gunn, M. Nikravesh, & L. A. Zadeh (Eds), *Feature Extraction, Foundations and Applications* (pp. 1-23). Berlin, Heidelberg: Springer-Verlag.

Hansen, L. K., & Salamon, P. (1990). Neural Network Ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12*(10), 993-1001.

Hinnerich, B. T., Höglin, E., & Johannesson, M. (2011). Are boys discriminated in Swedish High schools? *Economics of Education Review, 30*(4), 682-90.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2010). *A Practical Guide to Support Vector Classification.* Taipei: Department of Computer Science, National Taiwan University.

Johnson, R. L., Penny, J., Fisher, S., & Kuhs, T. (2003). Score Resolution: An Investigation of the Reliability and Validity of Resolved Scores. *Applied Measurement in Education, 4*(16), 299-322.

Jurafski, D., & Martin, J. H. (2009). *Speech and Language Processing* (2nd ed.). New Jersey: Pearson Education,Inc.

Kann, V. (2005). *Svenskgrammatikgranskning*. Hämtat från http://www.csc.kth.se/tcs/projects/granska/ 2012

Keerthi, S. S., & Lin, C.-J. (2003). Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel. *Neural Computation , 15*(7), 1667 - 1689 .

Landauer, T. K., Laham, D., & Foltz, P. (2003). Automatic Essay Assessment. *Assessment in Education:Principles,Policy&Practice, 10*(3), 295-308.

Manning , C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to Information Retrieval* (1 ed.). New York: Cambridge University Press.

Nakamura, Y. (2004). A comparison of holistic and analytic scoring methods in the assessment of writing. In JALT (Ed.). Tokyo.

Page, E. B., & Petersen, N. S. (1995). The computer moves into essay grading:Updating the ancient test. *Phi Delta Kappan, 76*(7), 561-565.

Pearson. (2009). *New York State testing Program ENGLISH AS A SECOND LANGUAGE ACHIEVEMENT TEST (NYSESLAT).* New York: Pearson.

Pedregosa, F., Varoquaux, G., Gramfort , A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830.

Powers, D. E., Burstein, J. C., Chodorow, M., Fowles, M. E., & Kukich, K. (2001). *Stumping e-rater: Challenging the validity of automated essay scoring.* Princeton,NJ: Educational Testing Service (ETS).

Rudner, L. M., Garcia, V., & Welch, C. (2005). *An Evaluation of IntelliMetric Essay Scoring System Using Responses to GMAT AWA Prompts.* McLean,Virginia: GMAC.

Schapire, R. E. (1999). A brief Introduction to boosting. In T. Dean (Ed.), *IJCAI'99 Proceedings of the 16th international joint conference on Artificial. 2*, pp. 1401-1406. San Francisco: Morgan Kaufmann Publishers Inc.

Sim, J., & Wright, C. C. (2005, March). The Kappa Statistic in Reliability Studies: Use, Interpretation and Sample Size Requiremments. *physical therapy journal of the american physical therapy association, 85*(3), 257-268.

Smith, C., & Jönsson, A. (2011). Automatic summarization as means of simplifying texts, An Evaluation for Swedish. *The 18th Nordic Conference of Computational Linguistics (NODALIDA).* Riga: Association of Computational Linguistics.

Traub, R. E. (1994). *Reliability for the Social Sciences Theory and Applications* (First ed.). Thousand Oaks, CA: Sage.

Walker, K. (2009, May). *Independent Teacher (The e Journal for Independent school educators).* Retrieved April 9, 2012, from Teaching AP Language and Composition through a British Literature Framework : http://www.independentteacher.org/vol6/6.2-5-Teaching-AP-Language-and-Composition.html

Vapnik, V. N. (2000). *The nature of statistical learning theory* (2nd ed.). New York: Springer-Verlag.

Warrens, M. J. (2012). Some paradoxical results for the quadratically weighted kappa. *PSYCHOMETRIKA, 77*(2), 315-323.

Wemmert, C., Forestier, G., & Derivaux, S. (2009). Improving Supervised Learning with Multiple Clusterings. In O. Okun, & G. Valentini (Eds.), *Applications of Supervised and Unsupervised Ensemble Methods* (Vol. 245, pp. 135-149). Berlin: Springer-Verlag.

Williamson , D. M., Bejar, I. I., & Hone, A. S. (1999). 'Mental Model' Comparison of Automated and Human Scoring. *Journal of Educational Measurements, 36*(2), 158-184.

Williamson, D. M., Xi, X., & Breyer, F. J. (2012). A Framework for Evaluation and use of Automated Scoring. *Educational Measurements: Issues and Practices, 31*(1), 2-13.

Witten, I. H., & Frank, E. (2005). *Data Mining Practical Machine Learning Tools and Techniques* (2nd ed.). San Francisco: Morgan Kaufman Publishers.

Yang, Y., Buckendahl, C. W., Juskiewicz, P. J., & Bhola, D. S. (2002). A Review of Strategies for validating Computer-Automated Scoring. *Applied Measurement In Education, 15*(4), 391-412.

Zhou, Z.-H. (2012). *Ensemble Methods.* Boca Raton,Florida: CRC Press.

Östling, R. (2012). Stagger: A modern POS tagger for Swedish. *The Fourth Swedish Language Technology Conference.*

8 Appendix

8.1 Appendix A: F-scores

The univariate feature ranking uses the F-score (ANOVA) to rank how important each feature by itself is for classifying the data set. That is, only one feature at the time is analyzed without any consideration about the other features. The following are some drawbacks with this ranking principle: 1) it does not take into consideration how different features may interact to each other to contribute to the final classification (two features that are individually low ranked may become high ranked when combined). 2) It does not take into consideration if two features are redundant or not. The advantage of this ranking principle is that it is fast and it is just used as reference in this project. A large F score implies that the feature is separating the classes well. Table 20is ordered according to the F-score from the highest ranking to the lowest ranking feature.

Table 20 Univariate feature ranking based on F-Score

Name of the features	F-score	p-value
fourth root of # of word tokens	175.639986471	1.49655274558e-97
# of word tokens	150.241548478	4.18727692191e-85
# of sentences	108.506735418	1.44952852892e-63
perplexity for POS bigrams in news corpora	106.420183169	1.88825955439e-62
# of essay word bigrams with zero match in news bigrams/# of bigrams	85.8671344236	2.9062744627e-51
# of grammar errors/# of word tokens	43.8207054881	3.84785317949e-27
# of essay word tokens classified as “news”/# of word tokens	41.5838915752	8.26285113151e-26
# lemmas/# of word tokens	33.4089984775	6.71221350492e-21
# of word tokens > 6/# of word tokens	31.7594178832	6.69049656625e-20
OVIX	31.1687080711	1.52644414984e-19
# of essay tokens classified as “blog”/# of word tokens	26.59647518	9.27633267034e-17
# of tokens shorter than 4 chars/# of word tokens	23.3924121559	8.4954798256e-15
# of prep(PP)/# of word tokens	17.3028509909	4.78458938389e-11
# of pronouns(PN)/# of word tokens	14.7974218367	1.69636465669e-09
# of non-initial CAPS words/# of sentences	14.284022361	3.52672603967e-09
# of paired delimiters (PAD)/# of word tokens	10.5313096439	7.43799668586e-07
Nominal Ratio	9.82950610041	2.02168238421e-06
# of adverbs (AB)/# of word tokens	9.82549238352	2.03327254489e-06
# of participles form (PC)/# of word tokens	8.52719999871	1.28968656793e-05
# of verb(VB)/# of word tokens	8.15400645613	2.19147709184e-05
# of nouns(NN)/# of word tokens	7.53895777175	5.24471464345e-05
# of infinitive mark(IE)/# of word	6.74939222562	0.000160364416428

tokens		
# of genitives/# of nouns	6.68963167011	0.000174495027443
# of determiners (DT)/# of word tokens	6.16417181423	0.000366270267286
# of personal names (PM)/# of word tokens	5.0403442433	0.00177429194723
# of passive voice/# of sentences	4.65547695113	0.00303553605928
# of conjunctions(KN)/# of word tokens	4.58847534377	0.00333227308104
# of foreign words(UO)/# of word tokens	4.41026295951	0.00426901119003
# of words in word list/# of word tokens	3.95160620217	0.0080555760279
# of .sub-junctions(SN)/# of word tokens	3.93190416402	0.00827755975398
# of possessives (PS)/# of word tokens	3.5223361934	0.0145361652267
# of minor delimiters (MID)/# of word tokens	3.33379788382	0.0188104815917
# chars/# of sentences	3.24601325171	0.0212018025776
# of active voice/# of sentences	2.17793704818	0.0888082086961
# of .relative adverbs(HA)/# of word tokens	2.13553669786	0.0938939703041
# of particles(PL)/# of word tokens	1.5988597737	0.18779596548
# of major delimiters (MAD)/# of word tokens	1.50451102463	0.21155617367
# of adjectives (JJ)/# of word tokens	1.48780405996	0.216044141172
# of word tokens/# of sentences	1.32282774271	0.265317313712
# of complex bigram/# of bigrams	1.31310425173	0.268519308687
# of complex words/# of word tokens	1.2379727655	0.294462904015
# of interrog.rel.pronouns(HP)/# of word tokens	1.14333349012	0.330324987109
# of .ordinal count words(RO)/# of word tokens	1.07356134582	0.359177022733
# of pronouns in object-form/number of pronouns	0.986714384383	0.398113927152
# of .numeral words(RG)/# of word tokens	0.541426929063	0.653973393095
# of interrog.determiners(HD)/# of word tokens	0.154992273139	0.926497740336

8.2Appendix B: Features and software functions

Table 21 describes the different functions used to create the different features. Stagger software is the basis for the creation of all the features because it produces the files in ".conll format" that contains the information about "word", "lemma" and "POS" for each word. All essays and news corpus are available in ".conll".

Table 21Features and the corresponding software functions

Feature names	Software functions
fourth root of # of word tokens	Essay word tokens are counted and the feature is created by the feature creation software
# of word tokens	Number of essay word tokens is processed by the feature creation software
# of sentences	#of sentences is counted by the feature creation software.
perplexity for POS bigrams in news corpora	Essays and news blog's POS are processed by bigram handling software to create "perplexity" that is processed by the feature creation software.
# of essay word bigrams with zero match in news bigrams /# of bigrams	Essay word tokens and news words are processed by bigram handling software. The result is processed by the feature creation software.
Word Variation Index (OVIX)	Essays POS and other information are used by the feature creation software.
# of grammar errors/# of word tokens	Plain essay text is used as input to GRANSKA that outputs the # of grammar errors. The results are passed to the feature creation software.
# of essay word tokens classified as "news"/# of word tokens	Blog and news software compare the essay word tokens with the blog and news's word tokens. The results are passed to the feature creation software.
# of words longer than 6 chars/# of word tokens	Feature creation software uses the word token to define the corresponding length.
# of conjunctions (KN)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of infinitive mark (IE)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of genitives/# of nouns	POS is used by the feature creation software to generate the feature.
# of minor delimiters (MID)/# of word tokens	POS is used by the feature creation software to generate the feature.
# lemmas/# of word tokens	Number of lemmas in an essay is used by feature creation software to generate the feature.
# of particles (PL)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of determiners (DT)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of relative adverbs (HA)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of participles forms (PC) /# of word tokens	POS is used by the feature creation software to generate the feature.
# of tokens shorter than 4 char/# of word tokens	Feature creation software uses the essay word token to define the corresponding length.
# of personal names (PM)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of adjectives (JJ)/# of word tokens	POS is used by the feature creation software to generate the feature.

# of possessives (PS)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of passive voice/# of sentences	POS is used by the feature creation software to generate the feature.
# of prepositions (PP)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of interrog. relative pronouns (HP)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of complex bigrams in essay/# of bigrams	The software to create features uses the list of complex bigrams and essay word tokens to create this feature.
# of numeral words (RG)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of word tokens in word list/# of word tokens	The function checks if the essay word tokens is in SALDO
# of pronouns (PN)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of sub-junctions (SN)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of nouns (NN)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of essay tokens classified as “blog”/# of word tokens	Blog and news software compare the essay tokens with data in the news/blog coropra. The results are passed to the feature creation software
# of pronouns in object-form(OBJ) /number of pronouns	POS is used by the feature creation software to generate the feature.
# of non-initial CAPS words/# of sentences	POS is used by the feature creation software to generate the feature.
# of foreign words (UO)/# of word tokens	POS is used by the feature creation software to generate the feature.
Nominal Ratio (NR)	Essay POS and other information are used by the feature creation software to create this feature.
# of adverbs (AB)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of complex words/# of word tokens	The software to create features uses the list of complex words and essay word tokens to create this feature.
# of verbs (VB)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of paired delimiters (PAD)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of active voice/# of sentences	POS is used by the feature creation software to generate the feature.
# characters/# of sentences	The software to create features ……
# of word tokens/# of sentences	The count of word tokens and the number of sentences are processed by the feature creation software to create this feature.
# of major delimiters (MAD)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of ordinal count words (RO)/# of word tokens	POS is used by the feature creation software to generate the feature.
# of interrogative determiners (HD)/# of word tokens	POS is used by the feature creation software to generate the feature.

Stockholms universitet/Stockholm University
SE-106 91 Stockholm
Telefon/Phone: 08 – 16 20 00
www.su.se

Stockholms universitet