Bachelor Informatica

University of Amsterdam

Informatica — Universiteit van Amsterdam

Early risk prediction of Anorexia from social media writings using supervised machine learning

Roy Broertjes

June 8, 2018

Supervisor(s): Maarten Marx, Maarten de Rijke

Abstract

Anorexia is a potentially life-threatening eating disorder which is estimated to occur in 2.9 million people globally. In this paper an attempt is made to develop an early risk prediction system which tries to identify risk cases of anorexia from writings of users on the social media platform reddit using supervised machine learning algorithms and its applicability to the real world is evaluated. Three machine learning algorithms for classification are deployed, which are Logistic Regression (LR), Support Vector Classifier (SVC) and Random Forest (RF), and performance of these different algorithms is compared. The machine learning algorithm that is found to perform the best is RF. Results from this paper show that early risk prediction systems have great potential to help with the task of identifying risk cases of anorexia via social media, however the final system proposed here is merely developed to explore the potentials of such automatic systems and more advanced systems are required to be effectively used in the real world.

Contents

1 Introduction **7**
- 1.1 Problem 7
- 1.2 Background 8
- 1.3 Structure of paper 8

2 eRisk **9**
- 2.1 Data 9
- 2.2 Evaluation Metric 10

3 Method **11**
- 3.1 Features 11
  - 3.1.1 First person pronouns 11
  - 3.1.2 Male and female third person pronouns 11
  - 3.1.3 Absolutist words 12
  - 3.1.4 Negative emotion words 12
  - 3.1.5 Anorexia-risk words 12
  - 3.1.6 Anorexia-risk words and singular first person writings 12
  - 3.1.7 Feature vectors per subject 14
- 3.2 Classification 15
  - 3.2.1 Logistic regression 15
  - 3.2.2 Support Vector Classifier 15
  - 3.2.3 Random forest 16
  - 3.2.4 Parameters 16
- 3.3 Early Risk Prediction system 17
  - 3.3.1 Specifcations 17
  - 3.3.2 Parameters 17
  - 3.3.3 Implementation 18
  - 3.3.4 Optimization and evaluation of the system 20

4 Results **23**
- 4.1 Experiments 23
- 4.2 Results 24

5 Conclusion **29**
- 5.1 Discussion 29
- 5.2 Conclusion 30

6 Appendix **33**

CHAPTER 1

Introduction

Anorexia nervosa, often referred to as anorexia or AN, is a psychological and potentially life-threatening eating disorder and is usually developed during teen years or young adulthood [9]. Anorexia, and eating disorders in general, is more common among females compared to males [22]. Individuals who suffer from anorexia often think of themselves as being overweight, when in fact they often are (extremely) underweight [15]. It is estimated that the number of people that suffer from anorexia is about 2.9 million people globally [8]. In this paper an attempt is made to develop an automatic system which chronologically processes chunks of writings, from individuals on social media, and detects early traces of anorexia as soon as possible by using natural language processing and machine learning techniques.

1.1 Problem

According to [10], in the Netherlands approximately 40% of the individuals that have anorexia have been diagnosed and about 80% of the individuals that have been diagnosed are treated. When anorexia is not treated properly, complications such as skin-, cardiovascular- and gastrointestinal diseases can arise, where some complications could eventually lead to death [11]. Additionally, studies show that suicide attempts are not uncommon by patients with anorexia [1, 2]. The low diagnosis rate found by [10] shows that there is room for improvement regarding the current way in which individuals are diagnosed.

Since anorexia could potentially be life-threatening when not treated properly and there seems room for improvement regarding the diagnosis rate, with this paper an alternative method, which uses an automatic system that should identify risk cases at an early stage using data from social media, is explored and the usefulness of this system when it would be put into practice is evaluated. Although this system should ideally detect traces of anorexia at the earliest stage possible, the focus of this paper is more towards correctly identifying risk-cases of anorexia than it is on detecting risk cases as early as possible. The system will automatically and chronologically process and analyse writings from users of the social news platform reddit with natural language processing techniques to convert the textual content of these writings to feature vectors. These vectors consist of measurable numerical properties extracted from the textual content, known as features. These feature vectors are then used to train and built a supervised machine learning model. After the model has been trained it is able to predict risk cases of anorexia for unseen data, which are also feature vectors, based on the data used to train the model. The correctness of the predictions depends on the quality of the features and the machine learning algorithm used. In this paper multiple machine learning algorithms are deployed and their performance in terms of precision, recall, f1-score and $ERDE$-score is compared and the system which yields the best results is discussed and used as the final system. Here, $ERDE$ is the Early Risk Prediction Error, which takes into account the time taken to identify risk cases and is described in Section 2.

Perhaps, when the system's ability to identify risk cases of anorexia seems promising, the system could be used by healthcare facilities to monitor social media platforms and help with identifying risk cases of anorexia for example. When social media platforms are monitored to detect risk cases of anorexia this could help improve the diagnosis rate of anorexia and let the number of individuals that seek proper treatment increase. When the number of people that seek treatment increases, the number of anorexia related deaths might decline.

1.2 Background

Early risk prediction, where signs of risk-cases are detected in an early stage, is a relatively new field of research. Losada et. al (2016) try to a shine light on this subject by organizing competitions in which participants are challenged to develop systems that perform early risk prediction tasks [7]. Losada et. al (2016) hope to explore the usefulness of these systems when used for medical, social and safety problems such as the early risk prediction of depression, anorexia and terrorist attacks with these competitions. These systems make use of techniques from fields on which extensive research has been done over the years already, such as natural language processing and machine learning techniques. Natural language processing is a field of study in computer science and artificial intelligence which is concerned with processing and analyzing large amounts of natural language data. Machine learning is concerned with the ability to let algorithms learn, often using statistical techniques, by feeding data into the algorithm [18]. After the algorithm has learned from the data, it tries to make predictions about data that has not been seen yet.

The system developed by this paper is programmed in `Python 3` [19]. For natural language processing tasks, the natural language processing toolkit `NLTK` is used [12]. For machine learning tasks, `scikit-learn` is used [17].

1.3 Structure of paper

In Section 2 the context in which this study is performed is discussed. eRisk, the competition which provided the data, is described as well as the data provided by eRisk and the way in which the early risk prediction system will be evaluated by eRisk. Section 3 is important, where Section 3.1 describes the way in which features are extracted from the writings, Section 3.2 describes the different machine learning algorithms used by the system and Section 3.3 provides the implementation for the early risk prediction system itself and discuss several choices made regarding the system's implementation. Section 4 describes the experiments performed on all different systems and presents the results obtained from these expirements. In Section 5 these results are discussed and the final early risk prediction system is chosen and compared with systems of fellow students. Lastly, Section 6 discusses and concludes about the usefulness of the final early risk prediction system when put into practice and suggests future work that might improve its performance and usefulness when put into practice.

CHAPTER 2

eRisk

eRisk is a competition, which is organized by D.E. Losada, F. Crestani and J. Parapar and described in *eRISK 2017: CLEF Lab on Early Risk Prediction on the Internet: Experimental Foundations*. The goal of eRisk was to explore issues processes related to early risk detection, a relatively new interdisciplinary research field [7]. Participants of eRisk are challenged to develop a system which detects anorexia as soon as possible and as accurate as possible under users of Reddit, with the help of natural langauge processing.

2.1 Data

The dataset is extensively described in an other paper from the organisators, *A Test Collection for Research on Depression and Language Use* [6]. As the reader might has noted, this paper describes a test collection for depression, but eRisk has stated that the test collection for anorexia has the same format as described in this paper.

The data consists of reddit users' posts and comments. Posts are placed on a sub-reddit, which can be seen as a topic. There exist subreddits relevant to anorexia, such as EatingDisorders and MyProAna. The reddit users are divided in two groups, an anorexia group and a control group. Users belong to the anorexia group if they have explicitely stated, on reddit, that they have been diagnosed with anorexia. For the control group random users are used and to make the collection more realistic a number of users which were active on subreddits relevant to anorexia, but did not suffer from anorexia are included.

The dataset consists of a total of 152 users, also referred to as subjects, from which 20 are labeled as risk cases and 132 are labeled as non-risk cases of anorexia. In this paper, risk cases are labeled as "1" and non-risk cases are labeled as "2". Each subject is stored as a XML-file.

	number of subjects
risk cases	20
non-risk cases	132
total cases	152

Table 2.1: Class distribution in the dataset

Each XML-file consists of posts and comments of a subject, with a maximum of 1000 posts and 1000 comments and thus a maximum of 2000 submissions. Each submission consists of the title of the submission (if available), the date of the submission and the textual content of the submission. Since the goal of the competition is to detect a risk of anorexia at as possible, each subject's XML-file is split into 10 chunks, where the first chunk contains the subject's oldest

10% submissions and the last chunk the newest 10% submissions The submissions of a subject $s$should be processed in a chronological way, in which the system should decide after every chunk whether s is a risk case, s is a non-risk case or that more information is required before the system can make a decision for s. [6].

2.2 Evaluation Metric

In the paper *A Test Collection for Research on Depression and Language Use* [6] an evaluation metric called Early Risk Detection Error ($ERDE$) is proposed. This evaluation metric differs from standard classification measures in the fact that it not only takes the correctness of the system's decision into account, but also includes the delay taken by the system to decide. The ERDE for a subject is defined as:

$$ERDE_o(d,k) = \begin{cases} c_{fp} & \text{if d = 1 and t = 2} \\ c_{fn} & \text{if d = 2 and t = 1} \\ lc_o(k) * c_{tp} & \text{if d = 1 and t = 1} \\ 0 & \text{if d = 2 and t = 2} \end{cases}$$

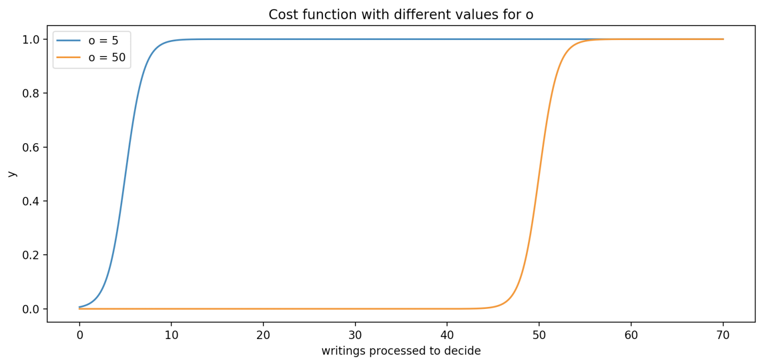
In this equation d is the decision made by the system for a subject s, where $d = 1$ means that the system decides s is a risk case of anorexia and $d = 2$ means that the system decides s is a non-risk case of anorexia. t denotes the true value for a subject s, where $t = 1$ means that s is a risk case of anorexia and $t = 2$ means that s is not a risk case of anorexia. k describes the delay the system has taken to make decision d, where its value is the number of writings the system has processed to make decision d. c_{fp} is the weighted error for false positives and c_{fn} is the weighted error for false negatives. These errors are weighted since our dataset is unbalanced; that is our control group contains relatively many subjects compared to our group of subjects who have been diagnosed with anorexia.

Here $c_{fn} = 1$, $c_{fp} = \dfrac{\text{risk cases}}{\text{total cases}}$ and $c_{tp} = 1$.

$lc_o(k)$ is a cost function which will grow with delay k and is defined as:

$$lc_o(k) = 1 - \frac{1}{1 + e^{k-o}}$$

The value of parameter o affects on which k the cost function will be the steepest, visualized in the figure below. $ERDE_5$ for example, where $o = 5$, could be used for evaluating systems where extremely early risk detection is required. $ERDE_{50}$ could be used when the earliness of the risk detections required is of less interest. Please note that in the ERDE function the delay, k, does not affect the error of a true negative. The overall $ERDE_o$ for a particular value of o of a system is defined as the mean of the $ERDE_o$ for all subjects.



CHAPTER 3

Method

3.1 Features

In this section the features chosen to be used by the system are discussed. Features are measurable characteristics of an observation which are used by machine learning models to train the model and let the model classify new data. In this paper, features should be extracted from writings submitted by users on reddit. Since writings are textual and features are often represented by numeric values, a method of converting textual data to numerical features should be searched for.

There are many ways to represent textual content in terms of features, such as the traditional Bag of Words (*BoW*) approach in combination with a weighting factor such as term frequency-inverse document frequency (*TF-IDF*). However, in this paper we try to choose features based on outcomes of various studies concerned with language use and the psychological state someone is in. Individuals that suffer from anorexia might have certain habits in terms of language use that could distinguish them from the general population. Since there is a lack of studies that examine the language use of anorexic individuals, the choice of including a feature is often supported by studies concerned with depressive people. Also, eatings disorders in general are found to often co-occur with affective disorders, such as depression, [23] and thus findings of these studies focussed on depression might also be applicable to individuals with anorexia.

3.1.1 First person pronouns

The average occurrence of first person pronouns (”I”, ”me”, ”myself”) per sentence is included in the feature vector. According to a study from 2004, which examined the language use of depressed and non-depressed college students, depressed students used the first person pronoun ”I” more often than students which never have been depressed [20]. A probable cause might be that depressed people are more self-focussed, according to this study.

3.1.2 Male and female third person pronouns

When examining the data something interesting was observed; cases from the anorexia group often used female third person pronouns (”she”, ”her”, ”herself”) more than male third person pronouns (”he”, ”him”, ”himself”), while cases from the control group generally often used male third person pronouns more often than female first person pronouns. A possible explanation for this may be that there are relatively more females suffering from anorexia compared to males [22], and that people tend to use pronouns of their own gender more often, however the latter is not backed up by any researches. Also, the data could simply be biased and there is no relation between an individual’s gender and the usage of gender-specific pronouns. Still, the ratio of male third person pronouns and female third person pronouns is included in the feature vector. The ratio is defined as:

$$r = \frac{pc_{male} - pc_{female}}{pc_{male} + pc_{female}}$$

where pc_{male} is the male pronoun count and pc_{female} is the female pronoun count. If there are no male- nor female pronouns used, $r = 0$. $r = -1$ implies only female pronouns are used and $r = 1$ implies only male pronouns are used.

3.1.3 Absolutist words

Absolutist thinking, sometimes referred to as absolute thinking, is the habit of describing feelings and circumstances in concrete, absolute terms. According to [14], absolutist thinking has a strong empirical link to several mental health groups such as suicidal ideation and eating disorders. This study found that people who suffer from anxiety, depression and eating disorders used absolutist words more often compared to the general population. These people would also have a habit of replacing these absolutist words with swear words, such as "fucking", which are used to create an even stronger denotion of totality. Due to these interesting findings, the average occurrence of absolutist words per sentences are included in the feature vector as well. All absolutist words used in this paper are obtained from [14] and listed in Appedix A.1.

3.1.4 Negative emotion words

According to several studies, writings from individuals suffering from affective disorders, such as anxiety disorders and depression, tend to contain relatively more words that denote negative emotions compared to the writings of the control group. Additionally, this is also the case for individuals suffering from anorexia according to [3], which found that college students who suffered from anorexia used words denoting negative emotions more often compared to students of the control group who have not suffered from anorexia. Based on these studies, the average occurrence of negative emotion words per sentence is also included in the feature vector. All negative emotion words used can be found in Appendix A.2.

3.1.5 Anorexia-risk words

Research where poets of suicidal and non-suicidal individuals were analysed, found that the suicidal individuals used more words associated with death [21]. Hence, occurrences of words associated with anorexia are counted and included as a feature. Anorexia-risk words consist of: words used to refer to anorexia, words associated with the obsessions anorexic people often have and words that describe the symptoms and side-effects of anorexia. Examples are "anorexia", "ed", "disorder", "food", "weight", "calories", "depression" and "skinny". For the complete list, please refer to Appendix A.3.

3.1.6 Anorexia-risk words and singular first person writings

Some of the anorexia-risk words described above can be used in combination with singular first person pronouns to refer to one's self. The subset of anorexia-risk words used in this feature is presented in Appendix A.4. Example of such sentences are: "My ed plays a major role in my daily life", "I have suffered from anorexia for years" and "Anorexia is killing me". The usage of these words in combination with first person pronouns could indicate a strong risk that this particular individual suffers from anorexia. However, the sentence "I think my friend suffers from anorexia" does not directly indicate a risk of anorexia for the person that wrote this sentence, although this sentence contains a first person pronoun in combination with an anorexia-risk word (this sentence indicates that the writer's friend might be a risk case). This shows that simply looking for a sentence in which a first person pronoun and an anorexia-risk word is present is not sufficient to conclude that the anorexia-risk word relates directly to the writer himself. To know for sure that this is the case, the sentence should be analysed grammatically. More specifically, the subject of the sentence and the part of the sentence that directly relates to the subject should be obtained and analysed to find any present anorexia-risk words that directly relate to the writer.

Stanford Parser

To be able to grammatically analyse sentences the Stanford Parser is used, which is a natural language parser program that works out the grammatical structure of sentences [13]. This parser is a probabilistic parser, which means the parser uses knowledge of language gained from hand-parsed sentences to try to produce the most likely analysis of new sentences. The parser will convert a sentence to a tree-structured object from which grammatical properties of the sentence can be extracted. For example when the sentence "I have suffered from anorexia for years" is parsed, the tree shown below is obtained:

```
(ROOT
  (S
    (NP (PRP I))
    (VP (VBP have)
      (VP (VBN suffered)
        (PP (IN from)
          (NP
            (NP (NN anorexia))
            (PP (IN for)
              (NP (NNS years))))))))))
```

Note that the goal is to obtain the subject and the part of the sentence which directly relates to the subject. For a tree like above, the subject is defined as the node labeled NP (noun phrase) that is a child of a node labeled S (sentence) and the sibling of a node labeled VP (verb phrase) [4]. The VP that is the sibling of the subject contains the words that relate to the subject directly. In the example above, the subject is "I" and the relatable sentence part is "have suffered from anorexia for years". However sometimes a sentence consists of multiple subjects, which is the case for "I think my friend suffers from anorexia" for example. When parsing this sentence, the tree below is obtained:

```
(ROOT
  (S
    (NP (PRP I))
    (VP (VBP think)
      (SBAR
        (S
          (NP (PRP$ my) (NN friend))
          (VP (VBZ suffers)
            (PP (IN from)
              (NP (NNP anorexia))))))))))
```

The subjects of the sentence can be found the same way as with one subject per sentence, but the part of the sentence that relates to each subject is harder to find. For the subjects "I" and "my friend" are found. However if all words from the sibling VP of each subject are considered directly relatable to this particular subject, like done previously, "think my friend suffers from anorexia" and "suffers from anorexia" come up as the relatable parts for "I" and "my friend" respectively. This is not desirable, since only "think" directly relates to "I". To overcome this problem, the directly relatable part of a subject is found by walking through the VP, starting at the top, and collecting all words found and stop when a different subject is found. This will result in the relatable parts being "think" and "suffers from anorexia" for the subjects "I" and "my friend" respectively.

Finding the valid sentences

Now that the subject and its directly relatable part of sentence can be found, it is time to find valid sentence parts, which is the concatenation of the subject and its directly relatable part, that could indicate a risk of anorexia for the writer. Note that there exist three forms of first person and depending on the form, different definitions of valid sentences are used:

The subjective first person form where "I" is used as the pronoun. An example of a sentence written in this form is "I have suffered from anorexia for years". Here the subject is "I" and the directly relatable part is "have suffered from anorexia for years". In the case of subjective first person writings, the sentence is defined to be valid if the subject contains the pronoun "I" and the directly relatable part contains an anorexia-risk word. Note that according to this definition, the sentence "I have suffered from anorexia for years" *does* indicate a risk of anorexia for the writer and "I think my friend suffers from anorexia" *does not* indicate a risk for the writer.

The possessive first person form, where "my" or "mine" is used as a first person pronoun. An example of a sentence written in this form is "My ed plays a major role in my daily life". Here the subject is "My ed" and its directly relatable part is "plays a major role in my daily life". In the case of possessive first person writings, the sentence is defined to be valid if the subject contains *both* "my/mine" and an anorexia-risk word. It thus is sufficient to only analyse the subject for this form. According to this definition, "My ed plays a major role in my daily life" *is* valid and "My friend plays a major role in my life" *is not* valid, since "friend" is not an anorexia-risk word.

The objective form, in which "me" is used as a first person pronoun. Examples of sentences written in this form are "Anorexia will kill me" and "His anorexia will kill me". For the former, the subject is "Anorexia" and the relatable part is "will kill me" and for the latter the subject is "His anorexia" and the relatable part is "will kill me". In the case of objective first person writings, the sentence is defined to be valid if the subject contains no pronouns and does contain an anorexia-risk word *and* the relatable part contains the first person pronoun "me". According to this definiton, "Anorexia will kill me" *is* valid and "His anorexia will kill me" and "Anorexia will kill him" *are not* valid.

All valid sentences are counted and combined for each first person form of writing, and the average occurrence per writing is included in the feature vector.

3.1.7 Feature vectors per subject

The process of extracting features for all subjects is quite time expensive, especially when the Stanford Parser is used for the last feature in the vector. There is a need to store these feature vectors properly, so that the features have to be generated only once and not every time an experiment is performed. Since the risk prediction system should process writings per chunk and decide after each chunk, it seems logical to generate feature vectors per chunk for each subject. This way, for each subject ten feature vectors are created where the first vector corresponds to features generated from the first chunk of writings, the second vector corresponds to features generated from the first two chunks of writings and so on. The last feature vector thus corresponds to features extracted from the writings of all chunks. These feature vectors for all subjects are stored in a `pickle` file, which makes it possible to re-use these features when the program is ran again without having to extract all features again.

3.2 Classification

The early risk prediction system requires a machine learning algorithm to be able to perform classification based on the feature vectors discussed in the previous section. There exist various machine learning algorithms, which are distinguished by two types; linear- and non-linear algorithms. Linear algorithms use a linear function as its prediction function and can solve problems that are linearly seperable, where non-linear algorithms use non-linear functions as its prediction function and tend to be more powerful but harder to train. In this paper, one linear algorithm and two non-linear algorithm are deployed. The linear algorithms deployed is Logistic Regression and the non-linear algorithms used are the Support Vector Classifier and the Random Forest classifier. These algorithms are available in the `scikit-learn` package, which is a machine learning package for Python [17]. The goal is to find the best suitable learning algorithm for this dataset, which is determined by comparing the system's performance for each algorithm deployed. The results are presented in Section 4.2 and discussed in Section 5.

3.2.1 Logistic regression

In logistic regression the log-odds of the probability of an event is a linear combination of prediction variables, which are estimated in the training phase. In this paper, there are two events for which the probability is calculated; an individual does have anorexia and an individual does not have anorexia. The former event is labeled as "1" and the latter as "2" in this paper. Logistic regression falls under the linear algorithms since the probability of those events is calculated by a linear combination of the estimated prediction variables. This algorithm itself is not a classifier, but can be used as classifier by defining the probability treshold required to classify inputs as "1". Logistic regression is used in several fields, such as in the medical field to predict the risk of developing a particular disease for example.

In this paper, the classifier is initialized by the following snippet:

```
model = LogisticRegression(C=C, random_state=42, class_weight='balanced')
```

where `LogisticRegression()` is implemented by `scikit-learn`.

3.2.2 Support Vector Classifier

The SVC implemented by `scikit-learn` uses a Support Vector Machine (SVM) at its core. A SVM is a non-probabilistic binary classifier which assigns new observations to one category or the other. In this paper, there are two categories; an individual does have anorexia and an individual does not have anorexia. A SVM represents the observations used to train the model as points in space, mapped in such way that observations from one category are divided by a gap, that is as wide as possible, from the other category. A new observation that should be classified is mapped into the same space as the training examples. Then a prediction of this new observation is made, which is based on the side of the gap this observation is mapped to. However, in this paper probabilities are required for each prediction that is made and SVMs do not supply probabilities with their predictions. For this reason SVC is used, which uses a SVM and does provide probabilities with each prediction made. `scikit-learn` states that these probabilities are calculated using cross-validation, however the exact manner in which this is done is not clear. SVC is considered a nonlinear algorithm since in this case it uses the radial basis function (RBF) as its kernel, which is a nonlinear kernel.

In this paper, the classifier is initialized by the following snippet:

```
model = SVC(C=C, random_state=42, class_weight='balanced', probability=True)
```

where `SVC()` is implemented by `scikit-learn`.

3.2.3 Random forest

Random forest is an ensemble learning algorithm used for classification. Ensemble learning algorithms use multiple learning algorithms to often obtain a better predictive performance than that could be obtained from any of those learning algorithms alone [16]. This algorithm constructs a number of decision trees in the training phase and predicts a new observation by finding the mode of the classes ("1" and "2") from all these decision trees. A decision tree uses a tree-like graph of decisions and their possible consequences used for decision making. Decision trees used for classification often suffer from overfitting, but random forest tries to overcome this problem by creating many decision trees. The algorithm is random by the fact that each decision tree uses a random subset of all features and the tree has access to only a random subset of the training data. Random forests are sometimes used as a feature-selection algorithm since these forests can rank the importance of the features by which the prediction is done, that is the influence each feature had on the prediction made.

In this paper, the classifier is initialized by the following snippet:

```
model = RandomForestClassifier(n_estimators=50,
                               random_state=42,
                               class_weight='balanced_subsample')
```

where `RandomForestClassifier()` is implemented by `scikit-learn`.

3.2.4 Parameters

- C, used by logistic regression and SVC, is a parameter which denotes the inverse regularization strength where $C > 0$. C can be used to improve the classifier's generalization performance, which is its performance on unseen data. Smaller values for C specify stronger regularization, which would help against overfitting our system. However, when C is too small the system could underfit.

- *probability*, used by SVC, is a parameter that needs to be set to true to let the SVC estimate probabilities with its predictions.

- *n_estimators*, used by random forest, denotes the number of decision trees that should be generated by the classifier. A higher value will generally tend to result in better predictions at the cost of the time needed to execute the algorithm. In this paper, *n_estimators* is set to 50.

- *class_weight*, used by all classifiers, denotes the weights used for each class. A weight assigned to a class affects the penalization used when predictions are incorrect for this class and is by default 1. A classifier is trained so that it minimizes the total penalty of the trained dataset. In an unbalanced dataset it is recommended to set this to *balanced*, to prevent the classifier from predicting the class which is most prevalent too often. It is set to *balanced_subsample* for random forest to compute the class weights per decision tree, where *balanced* would compute the class weights at the start using all training data.

- *random_state*, used by all classifiers, is the seed used by the random generator. It is set to a constant value so that results can be compared more easily, since results would be consistent each time the algorithm is performed with the same input. The value used is 42, which seems to be generally regarded as the standard for random seeds in computer science.

3.3 Early Risk Prediction system

The ultimate goal of this paper is to create a system which detects traces of anorexia for individuals on social media and correctly classify these individuals as risk-cases as soon as possible. The specifications of how the system should work are described by [7] and are summed up below.

3.3.1 Specifcations

The system should process all individuals (also referred to as subjects) per $chunk_i$, where $i = 1, ..., 10$. The system should start at processing writings of $chunk_1$ for each subject, which contains the oldest 10% of a subject's writings. After all writings of $chunk_1$ have been processed for each subject, the system should decide 1, 2 or 0 for each subject. Here, 1 means the system thinks the subject is a risk-case of anorexia, 2 means the system thinks the subject is not a risk-case. 0 means the system is not sure whether the subject is a risk-case or not yet, and will wait for new information obtained from the next chunk.

When the system has decided for all subjects on $chunk_1$, all writings of $chunk_2$ should be processed for all subjects for which the system has not made a decision of 1 or 2 yet, and the system should decide again. It is important to note that all writings in $chunk_1$ should also be used when deciding for $chunk_2$. More generally, when the system is deciding for $chunk_j$, it should use all writings found in $chunk_i$ where $i \leq j$. Also note that when the system has decided 1 or 2 for a subject s, this is the final decision for s.

After processing $chunk_{10}$, the last chunk, the system should make a decision for all subjects that have not been classified as 1 or 2 yet. The system has to decide 1 or 2, so that there remain no subjects classified as 0.

3.3.2 Parameters

There are various parameters that can be thought of which may affect the performance of the system in different ways when these parameters take on different values. In this paper, two parameters are included as parameters of the early risk prediction system itself, which are *treshold* and *treshold_on_final*. Additionaly, when using Logistic Regression or SVC as the classifier of choice in the system, C is included as a hyperparameter for the classifier and is described in 3.2.4.

treshold or t is the minimum probability p required with which the classifier used in the system thinks a subject should be classified as 1, to let the system actually decide the subject should be classified as 1. When $t = 0.50$, the system will decide 1 for a subject s whenever the classifier used thinks with $p > 0.50$ that s belongs to 1. Generally, when increasing t the $precision$ will increase and the $recall$ will decrease, where decreasing t will lead to a decreasing $precision$ and increasing $recall$.

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

treshold_on_final is a boolean that affects the way in which the system classifies subjects that still need to be classified on $chunk_{10}$, the final chunk. When true, the threshold t is used as well on the final chunk. On the other hand, when *treshold_on_final* is set to false, the treshold used on the final chunk is set to 0.50. It is not immediately clear which value for this parameter should be used to obtain the best performance, so both values will be used and the effects of this parameter's value on the system's performance will be presented in Section 4.2 and its value is chosen in Section 5.

3.3.3 Implementation

```
def early_risk_prediction(train_set, test_set, features, erp_parameters, model):
    # Initialise variables
    treshold            = erp_parameters['treshold']
    treshold_on_final   = erp_parameters['treshold_on_final']
    system_decisions    = init_decision_dictionary(test['x'])

    for chunk_i, chunk_key in enumerate(chunk_keys):

        # get features for all chunks for trainset and
        # get features of current chunk for testset
        train_set['features'] = ft.split_features(train_set['x'], features, 'chunk_10')
        test_set['features']  = ft.split_features(test_set['x'],  features, chunk_key)

        classifier = cc.classify(train_set, test_set, model)

        for subject_i, subject in enumerate(classifier['x']):
            # if a system already decided on a subject (1 or 2),
            # use this decision on all later chunks aswell
            if system_decisions[subject.id]['decided'] > 0:
                system_decisions[subject.id]['decisions'][chunk_i] =
                    system_decisions[subject.id]['decided']
                continue

            r_probability = classifier['probabilities'][subject_i][0]

            # if r_probability > treshold -> decide "1"
            if r_probability > treshold:
                system_decisions[subject.id]['decisions'][chunk_i] = 1
                system_decisions[subject.id]['decided'] = 1
                system_decisions[subject.id]['decided_on'] = chunk_i

                continue
            else:
                system_decisions[subject.id]['decisions'][chunk_i] = 0

            # if on last chunk, decide for every subject that has not been decided yet
            # with treshold = 0.5 or given treshold value (depending on treshold_on_final)
            if chunk_key == 'chunk_10':
                if treshold_on_final:
                    if r_probability > treshold
                        prediction = 1
                    else:
                        prediction = 2
                elif r_probability > 0.50:
                    prediction = 1
                else:
                    prediction = 2

                system_decisions[subject.id]['decisions'][chunk_i] = prediction
                system_decisions[subject.id]['decided'] = prediction
                system_decisions[subject.id]['decided_on'] = chunk_i

    return system_decisions
```

`early_risk_prediction()` takes the five arguments `train_set` `test_set`, `features`, `erp_parameters` and `model` as input. Here `train_set` and `test_set` contain the subjects and labels for training and testing the system respectively. `features` are all feature vectors for all subjects, where each subject has 10 different feature vectors. Here, the i-th feature vector of a subject, with $1 \leq i \leq 10$, is created from processing all j chunks where $1 \leq j \leq i$. This way, the features per chunk can be obtained efficiently and running `early_risk_prediction()` many times is less expensive, compared to when the feature vectors would be generated per chunk run-time in `early_risk_prediction()`. `erp_parameters` contain the parameters of the system defined earlier, *treshold* and *treshold_on_final*. Lastly, `model` contains the classifier that should be used and if applicable its hyperparameter C.

`early_risk_prediction()` consists primarily of three actions:

1. Use the classifier to make a prediction for all subjects in `test_set` for chunk i.
2. Let the system make a decision for all subjects in `test_set` for chunk i, according to these predictions made in the first step and any previous decisions made by the system
3. Increment i by one, where $1 \leq i \leq 10$ and starts at $i = 1$.

The code below is responsible for the first action. The features for the train- and testset are obtained in line 11 and 12 respectively and line 14 lets the classifier make predictions for the subjects in the testset. Note that the features that are obtained for the trainingset always use all chunks, due to the last argument of `ft.split_features()` always being `chunk_10` (line 11). It's not explicitely stated by [7] that this should be the case, however when the system would be used for real world applications it seems logical to use all data that is available for training our model. Please note that the features of the testset are obtained per chunk, so that the concept of early risk prediction is retained (line 12).

```
9     # get features for all chunks for trainset and
10    # get features of current chunk for testset
11    train_set['features'] = ft.split_features(train_set['x'], features, 'chunk_10')
12    test_set['features']  = ft.split_features(test_set['x'],  features, chunk_key)
13
14    classifier = cc.classify(train_set, test_set, model)
```

Below, the code responsible for action 2 is provided. Lines 19-22 check if the system made a decision other than 0 on previous chunks for subject s and whenever this is the case, the system makes the same decision for the current chunk and goes to the next subject. The reason for this is because in the specifications it's explicitely stated that once a decision other than 0 is made for s, this is the final decision for s. Lines 27-35 check if the probability with which the classifier thinks subject s is a risk case is greater than the threshold t. When this is the case the system will decide 1 for s and goes to the next subject. If not the system will decide 0 for s. Lines 39-52 let the system decide 1 or 2 at the final chunk for all subjects for which the system has not made a decision other than 0 yet. Please note that the ERDE is defined as:

$$ERDE_o(d,k) = \begin{cases} c_{fp} & \text{if d=FP} \\ c_{fn} & \text{if d=FN} \\ lc_o(k) * c_{tp} & \text{if d=TP} \\ 0 & \text{if d=TN} \end{cases}$$

Here, d is the correctness of the decision made by the system for a subject and k is the delay taken to make the decision. Because k is only used when d is a true positive, k does not affect the score when d is a true negative. For this reason, the system only decides 2 at the final chunk where it has the most information and thus has a higher probability of making a correct decision.

```
    # if a system already decided on a subject (1 or 2),
    # use this decision on all later chunks aswell
    if system_decisions[subject.id]['decided'] > 0:
        system_decisions[subject.id]['decisions'][chunk_i] =
            system_decisions[subject.id]['decided']
        continue

    r_probability = classifier['probabilities'][subject_i][0]

    # if r_probability > treshold -> decide "1"
    if r_probability > treshold:
        system_decisions[subject.id]['decisions'][chunk_i] = 1
        system_decisions[subject.id]['decided'] = 1
        system_decisions[subject.id]['decided_on'] = chunk_i

        continue
    else:
        system_decisions[subject.id]['decisions'][chunk_i] = 0


    # if on last chunk, decide for every subject that has not been decided yet
    # with treshold = 0.5 or given treshold value (depending on treshold_on_final)
    if chunk_key == 'chunk_10':
        if treshold_on_final:
            if r_probability > treshold
                prediction = 1
            else:
                prediction = 2
        elif r_probability > 0.50:
            prediction = 1
        else:
            prediction = 2

        system_decisions[subject.id]['decisions'][chunk_i] = prediction
        system_decisions[subject.id]['decided'] = prediction
        system_decisions[subject.id]['decided_on'] = chunk_i
```

3.3.4 Optimization and evaluation of the system

Optimization

Since the early risk prediction system discussed above takes the parameters *treshold*, *treshold_on_final* and depending on the classifier used sometimes C and particular value combinations for these parameters tend to affect the performance of our system differently, optimization of the system is required. This is the process of finding the right combination of values for each parameter at which the system's performance is maximized, where the measure of performance may differ. For example, the system could be optimized on its recall, which would mean searching for the optimal parameter values at which the system's recall is maximized. However, other measures of performance could be helpful as well, depending on the situation in which the system will be used. In this paper a system is optimized on several metrics, each of them being discussed in Section 4.1. The parameter space on which a grid search is performed is:

```
treshold = [0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95]
C        = [0.5, 1.0, 10, 20, 40, 60]
```

Evaluation

Once the system is optimized, evaluation of this system is required. With evaluation, the optimized system's performance on unseen data can be tested and compared with other systems. (Losada et. al) has delivered an evaluation script in which a system's precision, recall, f1-score and $ERDE$-score will be calculated automatically. However, this script requires all decisions made by the system to be written to files which are then read and the performance is calculated. As this paper performs heaps of early risk prediction experiments for various reasons, it seemed more logical to integrate this evaluation script in the program in such way that decisions made by the system do not need be stored in files and the score can be calculated more easily.

Cross-validation

For both optimization and evaluation, cross-validation is often used. The goal of cross-validation is to test the systems ability to predict new data that were not used in estimating it, in order to give an insight on how the system will perform to unseen data. With cross-validation, the dataset is partitioned in to two complementing subsets where one is used to train the system and the other to test the trained model. There are two types of cross-validation; exhaustive- and non-exhaustive cross-validation. The former trains and tests all possible ways a dataset can be partitioned in, which is quite expensive. Conversely, the latter will train and test some ways in which the dataset can be partitioned, but not all, which is less expensive. In this paper, cross-validation will be used to optimize the system and to evaluate the optimized system. However, performing cross-validation to optimize and evaluate a system seperately will result in evaluating the system with data that has also been used to optimize the system, since only one dataset is available. When this is the case, the measured performance obtained from evaluating the system will be optimistically biased [5]. This should be avoided because the measured performance will not give a correct reflection of the system's performance regarding unseen data, since it is optimistically biased.

To resolve this optimistic bias, nested cross-validation should be used, which divides the dataset in three sets; one for training the system, one for validation and one for testing. With nested cross-validation first an inner cross-validation is performed to optimize the system and obtain its optimal parameters and secondly, an outer cross-validation is performed with the optimal parameters found in the inner cross-validation (with data that has not been used in the optimization) to evaluate our system. The fact that the system is optimized with data that is not used to evaluate the optimized system on prevents the presence of an optimistic bias in the evaluation results. Unfortunately nested cross-validation is more expensive and for this reason non-exhaustive cross-validation is used, *k-fold* cross-validation in specific. With *k-fold* cross-validation, the dataset is partitioned in k subsets of equal size, where $k-1$ subsets are used for training and 1 subset is used for testing the system. This is done k times such that every subset has been used exactly once as testing set. However, traditional $k-fold$ cross-validation does not take into account the way the data is distributed (the number of positive and negative cases in the dataset) when partitioning the dataset. Since the dataset used is highly unbalanced, that is the number of negative cases in our dataset is significantly higher as the number of positive cases, traditional $k-fold$ cross-validation may generate partitions in which the fraction of positive and negative cases differs greatly between these partitions. To overcome this problem, *stratified k-fold* cross-validation is used, where the proportion between both cases is roughly the same for all partitions. This paper uses $k=10$ for both the inner- and the outer cross-validation performed.

CHAPTER 4

Results

4.1 Experiments

In this section the experiments performed to be able to determine the best performing system are described. That is, the classifier and parameter combinations which will result in the system's performance being maximized. However note that the performance of such system can be described in several metrics, such as precision and recall for example. This paper measures the system's performance in terms of its precision, recall, f1-score, $erde_5$-score and $erde_{50}$-score.

Systems can differ in the classifier being used and the parameters passed with it. The possible classifers that can be used are: Logistic Regression, SVC and Random Forest. The parameter values that are used are dependant on the metric the system is optimized on. This paper optimizes the system on four different metrics, which are: recall, precision, f1-score and Youden's J statistic. In contrast to the other parameters, `treshold_on_final` is not obtained from optimization but is used for each system twice, once as `False` and once as `True`. So ultimately, 12 different systems are presented for each value of `treshold_on_final`. The optimal parameter values for each system are also presented. Furthermore, the precision-recall curve is plotted for all three classifiers. To evaluate the features in the feature-vector, the importance of each feature in the final system is also presented.

4.2 Results

Table 4.1: *Performance of optimized systems for different metrics optimalisation is performed on. For each metric, three systems are optimized which differ in the type of classifier used (Logistic Regression, Support Vector Classifier and Random Forest).* `treshold_on_final = False` *for all systems.*

Optimized on precision					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.42$ $\sigma^2 = 0.06$	$\mu = 0.65$ $\sigma^2 = 0.05$	$\mu = 0.48$ $\sigma^2 = 0.04$	$\mu = 14.89$ $\sigma^2 = 2.59$	$\mu = 11.28$ $\sigma^2 = 21.87$
SVC	$\mu = 0.52$ $\sigma^2 = 0.09$	$\mu = 0.60$ $\sigma^2 = 0.09$	$\mu = 0.52$ $\sigma^2 = 0.05$	$\mu = 14.02$ $\sigma^2 = 2.43$	$\mu = 9.72$ $\sigma^2 = 16.41$
RF	$\mu = 0.83$ $\sigma^2 = 0.04$	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.73$ $\sigma^2 = 0.03$	$\mu = 13.21$ $\sigma^2 = 0.75$	$\mu = 8.26$ $\sigma^2 = 14.16$
Optimized on recall					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.30$ $\sigma^2 = 0.01$	**$\mu = 0.85$ $\sigma^2 = 0.05$**	$\mu = 0.44$ $\sigma^2 = 0.02$	$\mu = 15.32$ $\sigma^2 = 2.39$	$\mu = 8.94$ $\sigma^2 = 11.75$
SVC	$\mu = 0.52$ $\sigma^2 = 0.08$	$\mu = 0.65$ $\sigma^2 = 0.05$	$\mu = 0.53$ $\sigma^2 = 0.04$	$\mu = 14.55$ $\sigma^2 = 3.28$	$\mu = 8.32$ $\sigma^2 = 12.06$
RF	**$\mu = 0.85$ $\sigma^2 = 0.04$**	$\mu = 0.80$ $\sigma^2 = 0.06$	**$\mu = 0.79$ $\sigma^2 = 0.03$**	**$\mu = 13.21$ $\sigma^2 = 0.75$**	**$\mu = 7.59$ $\sigma^2 = 20.29$**
Optimized on f1-score					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.40$ $\sigma^2 = 0.05$	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.47$ $\sigma^2 = 0.03$	$\mu = 15.24$ $\sigma^2 = 2.63$	$\mu = 11.63$ $\sigma^2 = 19.17$
SVC	$\mu = 0.51$ $\sigma^2 = 0.10$	$\mu = 0.60$ $\sigma^2 = 0.09$	$\mu = 0.51$ $\sigma^2 = 0.06$	$\mu = 14.17$ $\sigma^2 = 2.67$	$\mu = 9.87$ $\sigma^2 = 16.03$
RF	$\mu = 0.83$ $\sigma^2 = 0.04$	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.73$ $\sigma^2 = 0.03$	$\mu = 13.21$ $\sigma^2 = 0.75$	$\mu = 8.26$ $\sigma^2 = 14.16$
Optimized on Youden's J statistic					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.39$ $\sigma^2 = 0.02$	$\mu = 0.80$ $\sigma^2 = 0.06$	$\mu = 0.52$ $\sigma^2 = 0.03$	$\mu = 14.61$ $\sigma^2 = 1.02$	$\mu = 8.16$ $\sigma^2 = 14.04$
SVC	$\mu = 0.61$ $\sigma^2 = 0.09$	$\mu = 0.65$ $\sigma^2 = 0.05$	$\mu = 0.57$ $\sigma^2 = 0.03$	$\mu = 14.17$ $\sigma^2 = 2.67$	$\mu = 10.86$ $\sigma^2 = 14.23$
RF	$\mu = 0.83$ $\sigma^2 = 0.04$	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.73$ $\sigma^2 = 0.03$	$\mu = 13.21$ $\sigma^2 = 0.75$	$\mu = 8.26$ $\sigma^2 = 14.16$

Table 4.2: *Identical to Table 4.1, except* `treshold_on_final = True` *for all systems.*

Optimized on precision					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.58$ $\sigma^2 = 0.12$	$\mu = 0.50$ $\sigma^2 = 0.10$	$\mu = 0.51$ $\sigma^2 = 0.08$	$\mu = 13.42$ $\sigma^2 = 0.66$	$\mu = 8.52$ $\sigma^2 = 17.01$
SVC	$\mu = 0.55$ $\sigma^2 = 0.04$	$\mu = 0.45$ $\sigma^2 = 0.12$	$\mu = 0.47$ $\sigma^2 = 0.12$	$\mu = 13.21$ $\sigma^2 = 0.34$	$\mu = 8.91$ $\sigma^2 = 16.11$
RF	**$\mu = 0.87$ $\sigma^2 = 0.04$**	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.75$ $\sigma^2 = 0.03$	$\mu = 13.21$ $\sigma^2 = 0.68$	$\mu = 8.17$ $\sigma^2 = 14.36$
Optimized on recall					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.30$ $\sigma^2 = 0.01$	**$\mu = 0.85$ $\sigma^2 = 0.05$**	$\mu = 0.44$ $\sigma^2 = 0.02$	$\mu = 15.32$ $\sigma^2 = 2.39$	$\mu = 8.94$ $\sigma^2 = 11.75$
SVC	$\mu = 0.55$ $\sigma^2 = 0.13$	$\mu = 0.50$ $\sigma^2 = 0.10$	$\mu = 0.49$ $\sigma^2 = 0.08$	$\mu = 13.58$ $\sigma^2 = 0.66$	$\mu = 8.69$ $\sigma^2 = 15.55$
RF	$\mu = 0.85$ $\sigma^2 = 0.04$	$\mu = 0.80$ $\sigma^2 = 0.06$	**$\mu = 0.79$ $\sigma^2 = 0.03$**	$\mu = 13.21$ $\sigma^2 = 0.75$	**$\mu = 7.59$ $\sigma^2 = 20.29$**
Optimized on f1-score					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.34$ $\sigma^2 = 0.04$	$\mu = 0.55$ $\sigma^2 = 0.12$	$\mu = 0.41$ $\sigma^2 = 0.06$	$\mu = 14.23$ $\sigma^2 = 1.31$	$\mu = 9.49$ $\sigma^2 = 17.04$
SVC	$\mu = 0.58$ $\sigma^2 = 0.15$	$\mu = 0.50$ $\sigma^2 = 0.10$	$\mu = 0.51$ $\sigma^2 = 0.09$	$\mu = 13.32$ $\sigma^2 = 0.48$	$\mu = 8.42$ $\sigma^2 = 13.93$
RF	**$\mu = 0.87$ $\sigma^2 = 0.04$**	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.75$ $\sigma^2 = 0.03$	**$\mu = 13.12$ $\sigma^2 = 0.68$**	$\mu = 8.17$ $\sigma^2 = 14.36$
Optimized on Youden's J statistic					
classifier	precision	recall	f1-score	$erde_5$	$erde_{50}$
LR	$\mu = 0.39$ $\sigma^2 = 0.01$	$\mu = 0.75$ $\sigma^2 = 0.06$	$\mu = 0.51$ $\sigma^2 = 0.03$	$\mu = 14.52$ $\sigma^2 = 0.85$	$\mu = 8.65$ $\sigma^2 = 18.05$
SVC	$\mu = 0.58$ $\sigma^2 = 0.15$	$\mu = 0.50$ $\sigma^2 = 0.10$	$\mu = 0.51$ $\sigma^2 = 0.09$	$\mu = 13.41$ $\sigma^2 = 0.56$	$\mu = 8.51$ $\sigma^2 = 14.87$
RF	**$\mu = 0.87$ $\sigma^2 = 0.04$**	$\mu = 0.70$ $\sigma^2 = 0.06$	$\mu = 0.75$ $\sigma^2 = 0.03$	**$\mu = 13.12$ $\sigma^2 = 0.68$**	$\mu = 8.17$ $\sigma^2 = 14.36$

Optimized on precision		
classifier	treshold	C
LR	$\mu = 0.94$	$\mu = 30.2$
SVC	$\mu = 0.71$	$\mu = 42.2$
RF	$\mu = 0.60$	N/A
Optimized on recall		
classifier	treshold	C
LR	$\mu = 0.50$	$\mu = 13.0$
SVC	$\mu = 0.50$	$\mu = 9.4$
RF	$\mu = 0.50$	N/A
Optimized on f1-score		
classifier	treshold	C
LR	$\mu = 0.86$	$\mu = 30.3$
SVC	$\mu = 0.74$	$\mu = 36.25$
RF	$\mu = 0.60$	N/A
Optimized on Youden's J statistic		
classifier	treshold	C
LR	$\mu = 0.66$	$\mu = 54.0$
SVC	$\mu = 0.66$	$\mu = 29.25$
RF	$\mu = 0.59$	N/A

Optimized on precision		
classifier	treshold	C
LR	$\mu = 0.94$	$\mu = 30.0$
SVC	$\mu = 0.59$	$\mu = 49.1$
RF	$\mu = 0.64$	N/A
Optimized on recall		
classifier	treshold	C
LR	$\mu = 0.50$	$\mu = 13.0$
SVC	$\mu = 0.50$	$\mu = 18.4$
RF	$\mu = 0.50$	N/A
Optimized on f1-score		
classifier	treshold	C
LR	$\mu = 0.78$	$\mu = 52.1$
SVC	$\mu = 0.55$	$\mu = 45.1$
RF	$\mu = 0.59$	N/A
Optimized on Youden's J statistic		
classifier	treshold	C
LR	$\mu = 0.67$	$\mu = 56.0$
SVC	$\mu = 0.51$	$\mu = 35.1$
RF	$\mu = 0.59$	N/A

Table 4.3: *Parameter values of optimized systems for different metrics optimalisation is performed on. For each metric, three systems are optimized which differ in the type of classifier used (Logistic Regression, Support Vector Classifier and Random Forest).* `treshold_on_final = False` *for the left table and* `treshold_on_final = True` *for the right table*

From Table 4.1 and Table 4.2 the system's performance with different values for `treshold_on_final` can be compared, where results presented in 4.1 are obtained with `treshold_on_final = False` and 4.2 with `treshold_on_final = True`. Note that the values displayed in bold in Table 4.1 and Table 4.2 are the highest values for each metric (column-wise) per table. It can be seen that there seems to be no real significant advantage for using one value over the other, however one thing to observe is that the optimal performances (displayed in bold) for each metric are all obtained when the system is optimized on recall in Table 4.1, where as in Table 4.2 the optimal performances are more spreaded out over different optimizations. Since eventually the system can be optimized on one metric only, it is decided to use `treshold_on_final = False`, where the optimal statistics all lie in optimization for recall.

To find the best system, one of the systems from Table 4.1 has to be chosen. It seems that the system that performs as best overall is optimized on recall and uses the random forest classifier (RF). This system performs best regarding precision, f1-score and $ERDE$-scores of all systems and its recall is just slightly lower (0.05) than the best recall found, which is 0.85 for optimization on recall with the logistic regression classifier (LR).

Also, for all classifiers the precision-recall curve (PRC) is plotted in Figure 4.1 below. A precision-recall curve is plotted by calculating the precision and recall when varying the treshold, where the precision is on the y-axis and the recall on the x-axis. This curve shows the trade-off between recall and precision for different tresholds. From this plot can be observed that random forest (RF) is the best performing classifier, since its precision is higher than LR and SVC for every value of recall.

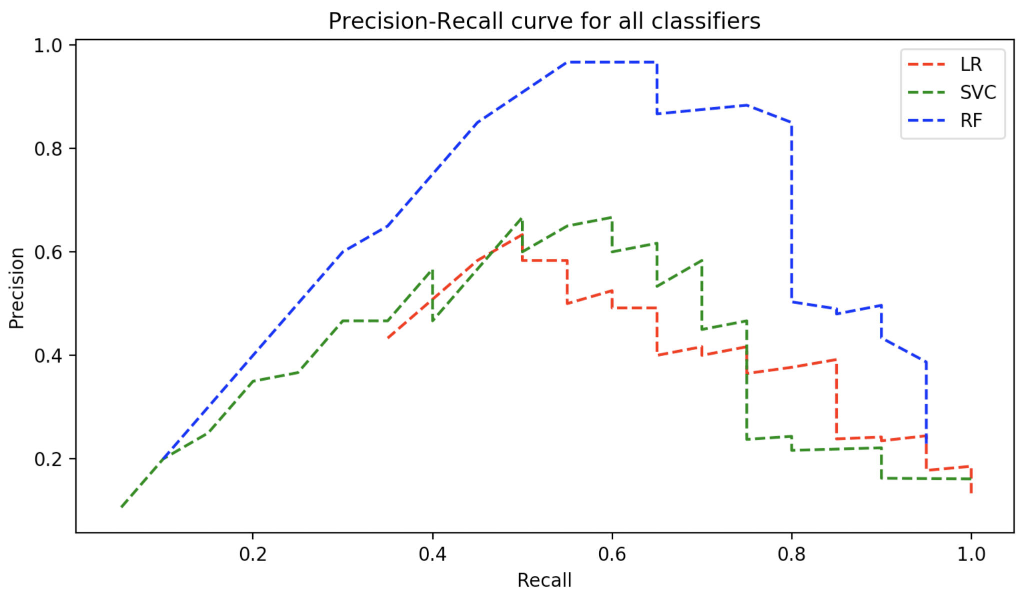


Figure 4.1: Precision plotted against recall by varying treshold in space [0.05 - 0.99] for all three classifiers; logistic regression (LR), Support Vector Classifier (SVC) and random forest (RF). For LR and SVC hyperparameter C = 40. Please note that `treshold_on_final = True`.

To get an impression of the earliness of the decisions made by the final system, the number of subjects for which "1" has been decided is plotted per chunk in Figure 4.2 below.

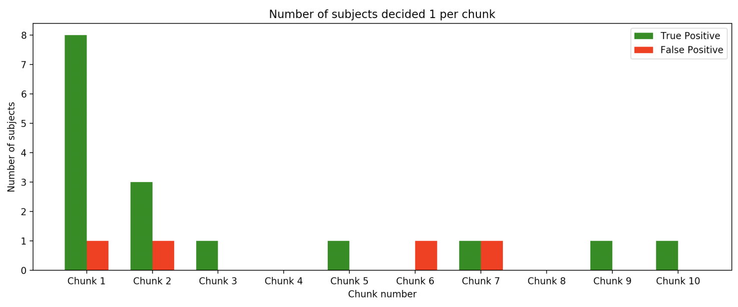


Figure 4.2: Number of times the final system decided "1" plotted per chunk, where the green bars are correct decisions and the red bars are incorrect decisions.

As can be seen, the majority of decisions are performed on the first two chunks which is good. Also, only just two of these decisions in the first two chunks are incorrect. The reason that the majority of the decisions is made in the first few chunks is probably because optimization is done on recall, which causes the treshold to be 0.50.

To get an impression of which features in the feature-vector are most informative and distinguish risk cases from non-risk cases the best in our system, the feature importance for the final system is shown below.

feature	**feature importance**
first person pronouns	0.17
male/female pronoun ratio	0.11
absolutist words	0.05
negative emotion words	0.05
anorexia-risk words	0.28
anorexia-risk words in first-person writings	0.34

Table 4.4: Feature importances for final system (RF), where higher importance indicates a more informative feature

From this table it can be observed that the best features are the only two features that use anorexia specific words, namely the average occurrence of anorexia-risk words and the average occurrence of anorexia-risk words in combination with first person writings. Two features of decent quality are the average occurrence of first person pronouns and the male/female pronoun ratio.

CHAPTER 5

Conclusion

5.1 Discussion

Although the results of the final system seem promising, it is important to note that with a relatively small dataset like in this case, there is a high risk of overfitting the system. Various actions are taken to lower this risk, such as the use of nested k-fold cross-validation. Also, the final system uses RF as its classifier, which tends to reduce overfitting as well by making use of ensembling techniques. Also, the results presented in Table 4.1 and Table 4.2 show that there is a relatively small variance in the results obtained from the different folds of the outerloop in the nested cross-validation. A low variance implies that the predictions made by the system are not highly dependant on the data used to train the system. However, the only way to get a clear picture of how well the system performs on new data, is by gathering extra data unfortunately.

To gain an impression on how well our system performs compared to fellow students their system, the table below is shown:

	Student's system			
Metric	S0	S1	S2	S3
Precision	**0.85**	0.55	N/A	N/A
Recall	**0.80**	0.75	N/A	N/A
F-1	**0.79**	0.63	N/A	N/A
$ERDE_5$	13.21	**12.90**	N/A	N/A
$ERDE_{50}$	**7.59**	8.21	N/A	N/A

Table 5.1: Performance of the final system (S0) and the system's performance for other students. For precision, recall and f-1 a higher score is better. For ERDE a lower score is better.

Note that it is not exactly known in which way felllow students evaluated their system and thus no useful comparison can be made from the presented results in the table above. For S1 it is for example known that he has left out random non-risk cases of anorexia in such way that the dataset is balanced, which may yield results that are very different compared to when *k*-fold cross-validation would be used. For S2 and S3 no results are available unfortunately.

5.2 Conclusion

From the three classifiers, Logistic Regression (LR), Support Vector Machine (SVC) and Random Forest (RF), RF is found to have the best overall performance of these three. Recall is found to be the best metric to optimize the system on. The final system thus uses RF as its classifier, optimized on recall. The final system's parameters and performance is shown below:

	parameters		performance				
classifier	**treshold**	**treshold_on_final**	**precision**	**recall**	**f-1**	$\boldsymbol{erde_5}$	$\boldsymbol{erde_{50}}$
Random Forest (RF)	0.50	False	0.85	0.80	0.79	13.21	7.59

Table 5.2: The final system's parameter values and performance. A higher precision, recall and f-1 is better and a lower ERDE-score is better.

Features that distinguished risk cases from non-risk cases the best, were features that are very specific to anorexia. In these features the use of words associated with anorexia is analyzed. These words consist of words used to refer to anorexia, words associated with the obsessions anorexic individuals often have and words that describe the symptoms and side-effects of anorexia. Features that were much less informative were the use of absolutist words and words denoting negative emotions.

The usefullness of the final system when used in real world applications, such as monitoring social media sites to identify risk cases, seems promising. The features extracted and machine learning algorithms deployed in this paper are not very complex, however the performance of the final system is still relatively good. One thing to keep in mind though is that the dataset dealt with is very small and there may be a risk that the system is overfitting to this data. This could cause the system to perform significantly worse on new, un-seen data.

This paper concludes that early risk prediction systems have great potential to help with the task of identifying risk cases of anorexia via social media, however the final system proposed here is merely developed to explore the potentials of such automatic systems and more advanced systems are required to be used for real-world problems.

Future work might include:

- deployment of more advanced and powerful classifiers, such as neural networks
- more advanced ways to extract features, for example via improved grammatical analysis of sentences, since the feature that was extracted by grammatical analyzing sentences was the most informative feature in the final system
- gathering more data to get a better insight on how well the system works on unseen data

Bibliography

[1] M. Bulik et. al. “Suicide Attempts in Anorexia Nervosa”. In: *Psychosomatic Medicine* 70-3 (2008), pp. 378–383.

[2] M. Pompili et. al. “Suicide in anorexia nervosa: A metafffdfffdfffdanalysis”. In: *International Journal of Eating Disorders* 36-1 (2004), pp. 99–103.

[3] Wolf et. al. “Linguistic analyses of natural written language: Unobtrusive assessment of cognitive style in eating disorders”. In: *Internation Journal of Eating Disorders* 40-8 (2007), pp. 711–717.

[4] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python.* O'Reilly Media, 2009.

[5] Cawley and Talbot. “On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation”. In: *Journal of Machine Learning Research* 11 (2010), pp. 2079–2107.

[6] F. Crestani D.E. Losada. *A Test Collection for Research on Depression and Language Use.* 2016.

[7] J. Parapar D.E. Losada F. Crestani. *CLEF 2017 eRisk Overview: Early Risk Prediction on the Internet: Experimental Foundations.* 2016.

[8] GBD 2015 Disease, Injury Incidence, and Collaborators Prevalence. “Global, regional, and national incidence, prevalence, and years lived with disability for 310 diseases and injuries, 1990-2015: a systematic analysis for the Global Burden of Disease Study 2015”. In: *Lancet* 388 (2016), pp. 1545–1602.

[9] H.W. Hoek F.R.E. Smink D. van Hoeken. “Epidemiology of Eating Disorders: Incidence, Prevalence and Mortality Rates”. In: *Current Psychiatry Reports* 14 (August 2012), pp. 406–414.

[10] H.W. Hoek and D. van Hoeken. “Review of the Prevalence and Incidence of Eating Disorder”. In: *International Journal of Eating Disorders* 34-4 (2003), pp. 383–396.

[11] Scott Crow James E. Mitchell. “Medical complications of anorexia nervosa and bulimia nervosa”. In: *Current Opinion in Psychiatry* 19 (July 2006), pp. 438–443.

[12] Edward Loper and Steven Bird. “NLTK: The Natural Language Toolkit”. In: *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1.* ETMTNLP '02. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2002, pp. 63–70. DOI: 10.3115/1118108.1118117. URL: https://doi.org/10.3115/1118108.1118117.

[13] Christopher D. Manning et al. “The Stanford CoreNLP Natural Language Processing Toolkit”. In: *Association for Computational Linguistics (ACL) System Demonstrations.* 2014, pp. 55–60. URL: http://www.aclweb.org/anthology/P/P14/P14-5010.

[14] Mohammed Al-Mosaiwi and Tom Johnstone. “In an Absolute State: Elevated Use of Absolutist Words Is a Marker Specific to Anxiety, Depression, and Suicidal Ideation”. In: *Clinical Psychological Science* 1-14 ().

[15] NIMH. *What are Eating Disorders?* `https://web.archive.org/web/20180510125044/https://www.nimh.nih.gov/health/topics/eating-disorders/index.shtml`. Accessed: 2018-05-10.

[16] D. Opitz and R. Maclin. "Popular Ensemble Methods: An Emperical Study". In: *Journal of Artificial Intelligence Research* 11 (1999), pp. 169–198.

[17] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[18] F. Provost R. Kohavi. "Glossary of terms". In: *Machine Learning* 30 (1998), pp. 271–274.

[19] Guido Rossum. *Python Reference Manual.* Tech. rep. Amsterdam, The Netherlands, The Netherlands, 1995.

[20] Eva-Maria Gortner James Pennebaker Stephanie Rude. "Language use of depressed and depression-vulnerable college students". In: *Cognition Emotion* 18:8 (2004), pp. 1121–1133.

[21] Pennebaker Stirman and James. "Word Use in the Poetry of Suicidal and Nonsuicidal Poets". In: *Psychosomatic Medicine* 63-4 (2001), pp. 517–522.

[22] Striegel-Moore. "Risk Factors for Eating Disorders". In: *Annals of the New York Academy of Sciences* 817-1 (1997), pp. 98–109.

[23] Blair Uniacke and Allegra Broft. "The Interplay of Mood Disorders and Eating Disorders". In: *Psychiatric Times* 33-5 (2016).

CHAPTER 6

Appendix

A.1.

Absolutist words
absolutely
always
all
complete
completely
constant
definitely
entire
ever
every
everyone
everything
full
must
never
nothing
totally
whole

Table 6.1: List of absolutist words used, obtained from [14]

A.2.

Negative emotion words
sad
angry
upset
lonely
disgusted
disappointed
nervous
bored
frustrated
miserable
discouraged
exhausted
terrible
worthless

Table 6.2: List of words used to denote negative emotions

A.3.

Anorexia-risk words
anorexia
ed
disorder
disorders
depression
depressed
skinny
food
weight
underweight
overweight
calories
calorie
diet
exercise
meal

Table 6.3: Words indicating a risk for anorexia

A.4.

Anorexia-risk words
anorexia
ed
disorder
depression
depressed
weight
underweight
overweight

Table 6.4: Anorexia-risk words that could be used in combination with first person writings