Automated Student Assessment Prize: Short Answer Scoring

Team JJJ

Technical Methods Paper

James Jesensky
james.jesensky@gmail.com

1 Introduction

The Automated Student Assessment Prize: Short Answer Scoring[1] was a contest to develop a scoring algorithm for student-written short-answer responses.

The contestants were supplied with 10 data sets and some domain documentation. Each data set contained student answers in ASCII format to a single question prompt. Additionally, those examples intended for training also included two integer scores assigned by a human grader. One set of scores was arbitrary selected as the gold standard.

The goal of the contest was to build an algorithm that could score new, unlabeled examples. Algorithms were judged on how closely their generated scores compared to a human grader's.

Team JJJ was awarded the 4th place prize for their final submission's score of 0.73392[2]. This document describes the technical methods used to create that final submission.

2 Method Overview

Methods implemented by the author were done using the Java programming language. Two open source Java libraries were found very helpful:

- Weka[3], a data mining library (GPL).
- JGAP[4], a genetic algorithm library (GLPL).

To create the submission used in the contest, four distinct steps were executed:

[1] http://www.kaggle.com/c/asap-sas
[2] Weighted mean quadratic weighted kappa.
[3] http://www.cs.waikato.ac.nz/ml/weka/index.html
[4] http://jgap.sourceforge.net/

1. *Text Preprocessing.* The raw text from the data sets was altered by applying several transformations.

2. *Feature Extraction.* Features were extracted from the transformed text. Features were conveniently organized into several new data sets.

3. *Predictive Modeling.* Predictive models were built using a variety of techniques. Models were trained using different subsets of features extracted in step 2. Approximately 40 to 45 models were built for each essay prompt.

4. *Ensemble Learning.* The outputs of multiple weak learners were combined to create a strong learner. This strong leaner was used to make the final predictions.

The four steps were executed separately for each essay prompt, i.e. they were repeated 10 times. The final submission was simply the union of the 10 disjoint sets of predictions.

3 Text Preprocessing

The following transformations were applied to the data prior to feature extraction.

3.1 Removal of Suspect Training Examples

It was posted[5] in the contest forum that some training examples were possibly truncated during the data capture process. In particular, the captured students' responses were only a few words--almost certainly deserving a score of zero--but the captured human graders' scores were all greater than zero.

These examples were removed form the training set and were not included in the predictive modeling or ensemble learning steps. See Appendix A for a complete list of the examples removed.

3.2 Text Substitution

Several global text substitutions were applied to the essay text. The substitutions were chosen based on a manual inspection of the data sets.

- Leading and trailing quotation marks (") were removed.
- A repeated sequence of quotation marks was replaced with a single quotation mark.
- Two adjacent apostrophes (') were replaced with a quotation mark.
- Ampersands (&) were replaced with the word "and".
- Plus signs (+) were replaced with the word "and".

[5] http://www.kaggle.com/c/asap-sas/forums/t/2126/are-the-scores-of-essay-set3-correct

- Slashes (/) were replaced with the word “or”.
- The sequences “^p”, “^P”, and “P^” were replaced with “^”.
- The sequence “(DEG)” was replaced with the word “degrees”.

3.3 Additional Preprocessing for Essay Set #10

Essay Set #10 was unique in two ways:

1. Students were required to provide an answer to both a multiple choice prompt and an essay prompt.
2. The responses contained extraneous spaces introduced (presumably) during the transcription process.

The student’s answer to the multiple-choice prompt was embedded into the essay response. This was removed during the preprocessing.

The exact pattern of extraneous spaces could not be determined so a programmatic solution to reverse the errors was not possible. Instead, a list of keywords was manually constructed. If removing a space between two adjacent words formed one of the keywords, then it was assumed that the space was extraneous, and the space was removed from the text. See Appendix B for a complete list of the keywords used.

3.4 Correction of Misspelled Words

To (attempt to) correct misspelled words, the program JaSpell[6] was run on the essay text in an offline mode. In an offline mode, JaSpell automatically chooses the most likely correction for a detected misspelling.

Two custom dictionary files were created to supplement the spell check process.

A dictionary of commonly misspelled words and their preferred correction was created by manually inspecting the data sets. When JaSpell finds a misspelled word in this list, it always chooses the correction specified in the list (over the correction it would normally choose). See Appendix C for the complete list.

A dictionary of jargon words was created. JaSpell ignores words found in the jargon list. The jargon dictionary was created from two sources:

1. A manual list of terms was created by inspecting the data sets. See Appendix D.

[6] JaSpell was written by Bruno Martins and is covered by the BSD license. See http://jaspell.sourceforge.net for more information.

2. The contest data included an archive Data_Set_Descriptions.zip, which contained several documents containing domain information about the prompts. All words found in these documents were added to the jargon dictionary.

4 Feature Extraction

4.1 Bags of Words

Most features were extracted using the Weka class StringToWordVector[7]. The StringToWordVector class takes text as input and outputs a set of variables containing word occurrence information. More commonly, this set of variables is referred to as a *bag of words*.

The following table lists the data sets created in this manner:

Name	Essays	Tokenizer (a)	Stop Words (b)	Stem (c)	Min TF (d)	Max Words (d)	Output (e)
alphabetic	1-10	Contiguous alphabetic sequences	None	No	2	65536	{0,1}
ngrams-1	1-10	Words	Custom	Yes	2	65536	{0,1}
ngrams-1-2	1-10	Words and word bigrams	Custom	Yes	2	65536	{0,1}
ngrams-2	1-10	Word bigrams	Custom	Yes	2	65536	{0,1}
ngrams-1-3	1-10	Words, word bigrams, and word trigrams.	Custom	Yes	4	1000	{0,1}
ngrams-2-3	1-10	Word bigrams and word trigrams.	Custom	Yes	4	1000	{0,1}
pls-2	1,4,10	Words	Custom	Yes	2	1000	{0,1}
pls-4	2-3, 6-8	Words	Custom	Yes	4	1000	{0,1}
pls-8	5	Words	Custom	Yes	8	1000	{0,1}
pls-32	9	Words	Custom	Yes	32	1000	{0,1}
reg-120	5-6	Words	Custom	Yes	1	120	{0,1}
reg-160	1-4, 7-10	Words	Custom	Yes	1	160	{0,1}
cos-1 (f)	1	Words	Custom	Yes	1	100	{0,1}
cos-2 (f)	2	Words	Custom	Yes	1	100	tf*idf
cos-3 (f)	3	Words	None	Yes	1	100	{0,1}
cos-4 (f)	4	Words	None	Yes	1	700	{0,1}
cos-5 (f)	5	Words	None	Yes	1	400	log(tf+1)
cos-6 (f)	6	Words	None	Yes	1	200	{0,1}
cos-7 (f)	7	Words	Custom	Yes	1	500	tf*idf
cos-8 (f)	8	Words	Weka	Yes	1	300	{0,1}
cos-9 (f)	9	Words	Custom	Yes	1	200	{0,1}
cos-10 (f)	10	Words	Weka	Yes	1	100	log(tf+1)
q-grams	1-10	All subsequences of length 3 (g)	None	No	64	2000	tf

[7] http://weka.sourceforge.net/doc.dev/weka/filters/unsupervised/attribute/StringToWordVector.html

Table Notes:

(a) The method used to convert the raw text into tokens. Words are considered the text between whitespace, periods, commas, semi-colons, colons, apostrophes, quotation marks, parentheses, question marks, and exclamation points.
(b) The list of stop words used, if any.
    i. Custom – The list in Appendix E was used.
    ii. Weka – The default list included in the Weka library was used.
    iii. None – No stop word list was used.
(c) Indicates if stemming was applied to the words. If yes, Weka's implementation of the Lovins stemmer was used.
(d) Minimum term frequency and maximum words to keep for each class. The lesser of these two quantities sets an approximate upper bound on the number variables in the bag for each class.
(e) The quantity output for each term in a document.
    i. {0,1} – Output 1 if the term is in the document, else 0.
    ii. tf – Output the number of times the term occurs in the document.
    iii. tf*idf – Output the tf*idf transformation[8].
    iv. log(tf+1) – Output the log of the term frequency plus 1.
(f) Minimum term frequency and maximum words to keep were calculated globally for all training examples (as opposed to separately per class).
(g) For example, the text "hello world" is tokenized into the terms "hel", "ell", "llo", "lo$", "o$w", "$wo", "wor", "orl", and "rld", where $ represents a space character in the original text.

4.2 Other Features

Besides term frequency information, a few additional features were extracted as listed in the following table:

Data set name	Essays	Variable Name	Description
extra	1-10	sent	The number of sentences. Determined by running the essay text through the Stanford Lexicalized Parser[9].
extra	1-10	para	The number of carrot (^) characters in the text. These characters are inserted by the transcriber to indicate a paragraph break.
extra	1-10	length	The character length of the essay text.
extra	1-10	words	A rough count of the number of words in the essay text.
extra	1-10	unique_words	A rough count of the number of unique words in the essay text.
cat-color	10	color	A category variable with categories {x,black,dark gray,light gray,white}. Represents the student's answer to the multiple-choice question. The category x is used if the student did not answer the question.

[8] http://en.wikipedia.org/wiki/Tf*idf
[9] http://nlp.stanford.edu/software/lex-parser.shtml

Data set name	Essays	Variable Name	Description
bin-color	10	color_x	A binary variable that is 1 if the student did not answer the multiple-choice question, otherwise 0.
bin-color	10	color_black	A binary variable that is 1 if the student answered "black" to the multiple-choice question, otherwise 0.
bin-color	10	color_dark_gray	A binary variable that is 1 if the student answered "dark gray" to the multiple-choice question, otherwise 0.
bin-color	10	color_light_gray	A binary variable that is 1 if the student answered "light gray" to the multiple-choice question, otherwise 0.
bin-color	10	color_white	A binary variable that is 1 if the student answered "white" to the multiple-choice question, otherwise 0.

5 Predictive Modeling

5.1 Training Methods

Models were built using a variety of algorithms on different subsets of the extracted features. However, the same training method was used for all models.

All models are trained twice, once using stratified 10-fold cross validation, and once on the full training set.

The training examples include two dependant variables, score1 and score2, which represent the scores awarded by two separate human graders. Since most machines learning algorithms are designed to work with a single dependant variable, each training example was inserted twice into any generated training set[10]. One instance used score1 as the dependant variable, and the other instances used score2.

Example:

Given the training observation:

(x_1,x_2,…,x_k,score1,score2)

The follow two observations were inserted into the training set used by machine learning algorithms:

(x_1,x_2,…,x_k,score1)
(x_1,x_2,…,x_k,score2)

[10] In the case of cross validation, the held out fold is not considered part of the training data and will only contain one observation for the gold standard.

5.2 Prediction Methods

Two sets of predictions were generated for each model. The first set of predictions was generated during the cross validation process. This was simply the union of the disjoint predictions made on the held out fold each round. The second set of predictions was generated for the unlabeled observations. Such predictions were made by a classifier trained on the full training set.

The method used to predict class labels depends on the type of machine learning algorithm. For regression algorithms, the prediction is a single continuous value, so it is rounded to the nearest integer class label. For classification algorithms, class support values[11] are calculated, and the class receiving the most support is chosen. If there is a tie, the class with the greatest representation in the training set is chosen.

5.3 The Models

This section enumerates the model types that were built for the contest. The essays and features used with each model type, as well as cross validated estimates of the scoring metric, are included in Appendix F through Appendix O.

Canonical Name	Reg (a)	Weka (b)	Feature Selection (c)	Parameters (d)
Bayes Network	No	BayesNet	Information gain > 0.0001	Estimator: BMAEstimator
Naïve Bayes	No	NaiveBayes	Information gain > 0.0001	
Radial Basis Function Network	No	RBFNetwork	Information gain > 0.0001	
SVM (Linear Kernel)	No	SMO	Information gain > 0.0001	
Naïve Bayes Tree	No	NBTree	Information gain > 0.0001	
Logistic Regression	No	Logistic	Information gain > 0.0001	
Bagged Trees	No	Bagging and REPTree	Information gain > 0.0001	Iterations: 100
Bagged SVM (Linear Kernel)	No	Bagging and SMO	Information gain > 0.0001	Iterations: 100
Cost-sensitive Naïve Bayes (e)	No	CostSensitiveClassifier, Bagging, and NaiveBayes	Information gain > 0.0001	Iterations: 100, Minimize Expected Cost: true

[11] Class support values are sometimes called class probability estimates.

Canonical Name	Reg (a)	Weka (b)	Feature Selection (c)	Parameters (d)
K-nearest Neighbors	No	IBk	None	Distance Function: cosine distance, Distance Weighting: 1/d, K: 13 (essay 3), K: 21 (essays 2,6,10), K: 34 (essays 1,4,8,9), K: 55 (essays 5,7)
SVM (RBF Kernel)	No	SMO	Information gain > 0.01	Filter type: none, Build Logistic Models: true, Kernel: RBFKernel, Gamma: 0.05
Partial Least Squares (f)	Yes	PLSClassifier	None	For q-grams data sets: Default parameters For ngrams-1-3 data sets: Preprocessing: Center Components: 5 For other data sets: Preprocessing: None Components: 5
Linear Regression	Yes	LinearRegression	None	
Random Subspaces	Yes	RandomSubSpace and LinearRegression	None	Attribute Selection Method: none, Iterations: 30
Boosted Stumps	Yes	AdditiveRegression and DecisionStump	None	Iterations: 1000
Boosted Trees	Yes	AdditiveRegression and REPTree	None	Iterations: 100

Table Notes:

(a) Is this a regression model (Yes) or a classification model (No). Regression models represent the class variable as a continuous variable and classification models represent the class variable as a category variable.

(b) The unqualified name(s) of the Weka classifier(s) used. For more details about a particular classifier's methods, consult the Weka documentation at:
http://weka.sourceforge.net/doc.dev/
http://weka.sourceforge.net/doc.packages/RBFNetwork/
http://weka.sourceforge.net/doc.packages/naiveBayesTree/
http://weka.sourceforge.net/doc.packages/partialLeastSquares/.

(c) Additional feature selection used, if any. This involves calculating the information gain[12] (with respect to the class) for each feature, and discarding features below a minimum gain. In the case of cross validation, the information gain is re-calculated each training iteration.

(d) Additional parameters passed to the Weka classifier beyond the default parameter values. More information about the parameters and their default values can be found in the links provided in table note (b).

[12] http://en.wikipedia.org/wiki/Information_gain_in_decision_trees#General_definition

(e) Takes into account the cost of misclassification when predicting class labels[13]. Bagging is used to generate more accurate class probability estimates. Uses the following cost matrix for essays with three classes:

$$C = \begin{bmatrix} 0.0 & 0.25 & 1.0 \\ 0.25 & 0.0 & 0.25 \\ 1.0 & 0.25 & 0.0 \end{bmatrix}$$

And the following cost matrix for essays with four classes:

$$C = \begin{bmatrix} 0.0 & 0.1 & 0.4 & 1.0 \\ 0.1 & 0.0 & 0.1 & 0.4 \\ 0.4 & 0.1 & 0.0 & 0.1 \\ 1.0 & 0.4 & 0.1 & 0.0 \end{bmatrix}$$

(f) Only most common parameters used shown. Exceptions are as follows: for non q-gram data sets, centering is always used with essay set #10, and standardization is always used with essay set #7. Also, for non q-gram data sets, 10 components are always used with essay set #7.

6 Ensemble Learning

The predictive models built in section 5 were considered weak learners. In the ensemble learning step, their outputs were combined to create a strong learner. In this case, 60 strong learners were actually built forming a committee of strong learners. The outputs of the committee members were again combined to make the final predictions.

The method used to build the committee was as follows:

(1) Let L be the set of weak learners; let C be an empty committee.
(2) Initialize a random number generator with a unique seed.
(3) Initialize a genetic algorithm[14] with the random number generator from (2).
(4) Evolve the genetic algorithm outputting $L^* \subseteq L$.
(5) Train[15] a naïve Bayes classifier, S, on the class labels predicted[16] by L*.
(6) Add (S, L*) to C.
(7) Repeat steps (2) through (6) until C has 60 members.

[13] http://weka.wikispaces.com/CostSensitiveClassifier
[14] http://en.wikipedia.org/wiki/Genetic_algorithm
[15] Similar to section 5.1, two observations are used for every one training example: one using score1 as the dependant variable and the other using score2.
[16] That is, the class labels predicted on the held out folds during the stratified 10-fold cross validation discussed in Section 5.2. These predictions were saved to be later used in the ensemble process.

The genetic algorithm performs a heuristic search over the subsets of weak learners—outputting the subset found with the highest fitness score. The fitness score of a subset was determined by performing stratified 5-fold cross validation on the class labels predicted for the held-out training examples in Section 5.2. (Since these predictions were themselves a product of cross validation, this technique is sometimes called *embedded cross validation*.) A naïve Bayes classifier was trained[17] on each of the 5 training splits, and then used to generated predictions on the corresponding held-out fold. The fitness score was finally taken as the quadratic weighted kappa metric calculated using the predicted labels vs. the actual gold standard.

Genetic algorithms represent a point in the search space as a collection of genes. In this case a gene is a Boolean variable, and there is one such variable for each candidate weak learner. If the variable's value is true, the corresponding weak learner's predictions were included in the fitness calculation, otherwise they were excluded. The other parameters to the generic algorithm were:

Parameter	**Value**
Population Size	10
Generations	30
Preserve Fittest Individual	True
Keep Population Size Constant	True
Generic Operators:	Standard crossover and random mutation
Crossover Rate:	35% of population size
Mutation Rate:	1 in 12 genes (average)

To predict the class label for an unlabeled example using the committee, the following method was used:

(1) Let $\mathbf{x}$ be an unlabeled example.
(2) Let Y be an empty set.
(3) Let $f(S,\mathbf{x})$ be the class label predicted by trained classifier S on input $\mathbf{x}$.
(4) For each committee member (S, L*), perform (5) through (7).
(5) Let $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ where $m_i = f(\ell_i, \mathbf{x})$ for all $\ell_i \in L^*$.
(6) Let $y = f(S, \mathbf{m})$.
(7) Add y to Y.
(8) Output the mode of Y.

In other words, the final prediction was the simple majority of the votes of all the committee members. (In cases where there was a tie, the class label with the greatest representation in the training examples was used). Hence, the final submission was exactly all unlabeled examples as predicted by the committee.

[17] Again, like in section 5.1, two observations are used for every one training example: one using score1 as the dependant variable and the other using score2.

7 Discussion

7.1 What Wasn't Used

This section briefly describes some methods that were considered, but not used in the final solution.

- *The Burrows-Wheeler transform*[18]. An alternate set of q-gram models were built, but with BWT first applied to the essay text. Unfortunately, models built in this manner did not perform well.

- *String kernels*. Some prototype models were built using Weka's StringKernel[19] class. Unfortunately, even these simple string kernel models required a day of processing time, so string kernels were abandoned simply because the processing requirement made them impractical.

- *Statistical parser*. All responses were run through the Stanford Lexicalized Parser[20] generating a variety of data that could be engineered into additional features. Unfortunately, there wasn't enough time to incorporate this data into the final solution.

- *Stacked generalization.* Initially StackingC[21] was used for the ensemble model combining, but was later abandoned. Poorly calibrated class probability estimates may have accounted for the algorithm's less than expected performance.

7.2 Performance across Different Essay Prompts

Appendices F through O are sorted by the scoring metric to more clearly show how much the performance of an individual classifier can vary across essay prompts. However, no attempt was made to try to explain these differences. Instead, most effort was focused on holistic improvements to the overproduce-and-choose process.

---

[18] http://en.wikipedia.org/wiki/Burrows%E2%80%93Wheeler_transform
[19] http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/supportVector/StringKernel.html
[20] http://nlp.stanford.edu/software/lex-parser.shtml
[21] A.K. Seewald: How to Make Stacking Better and Faster While Also Taking Care of an Unknown Weakness. In: Nineteenth International Conference on Machine Learning, 554-561, 2002.

Appendix A

Excluded training examples that appeared to have been truncated during the data capture process:

Id	EssaySet	Score1	Score2	EssayText
14	1	2	2	In order the replicate the experiment you need
646	1	2	2	The group will need
1051	1	1	1	I think that these braid made to for the
1194	1	3	3	I would need to know how much vine
1304	1	1	1	The group in this experiment can do
1452	1	1	1	After adding I determined that
3002	2	3	3	In this experiment students tested to see
3070	2	2	2	WSR
3148	2	3	3	A) In conclusion plastic type
3196	2	3	3	NSR
3393	2	2	2	In conclusion trial any
3403	2	1	1	The conclusion
3489	2	3	3	Plastic
3709	2	2	2	Based
3806	2	1	1	A-
3850	2	3	3	Nor.
4009	2	2	2	Well as we see on

Appendix B

A list of keywords manually constructed for essay set #10 to help identify transcription errors:

absorb
absorbs
affect
air
average
because
black
cold
color
colors
cool
cooler
dark
darker
data
degress
doghouse
effect
energy
experiment
gray
heat
hot
inside
keep
light
not
paint
summer
sun
temperature
warm
warmer
white
winter

Appendix C

A manually constructed list of commonly misspelled words found in the responses and their preferred correction:

Misspelling	Correction
alot	a lot
artical	article
auther	author
backround	background
becuase	because
celcius	celsius
couldnt	couldn't
didnt	didn't
diffrent	different
doesnt	doesn't
dont	don't
eachother	each other
expirement	experiment
grey	gray
judgement	judgment
membrain	membrane
protien	protein
protiens	proteins
recieves	receives
satelites	satellites
seperate	separate
streched	stretched
suprised	surprised
te	the
tempature	temperature

Misspelling	Correction
tha	that
throught	through
trys	tries
vinagar	vinegar
vinager	vinegar
vinaggar	vinegar
vineagar	vinegar
vineager	vinegar
vinegear	vinegar
vineger	vinegar
vinergar	vinegar
vingar	vinegar
vingear	vinegar
vinger	vinegar
vingeror	vinegar
vingnr	vinegar
vinigar	vinegar
vinigare	vinegar
vinigars	vinegar
viniger	vinegar
viniguer	vinegar
vinnegar	vinegar
vinrgar	vinegar
wich	which
wouldnt	wouldn't

Appendix D

A manually constructed list of jargon terms:

adp
anaphase
anticodon
anticodons
atp
codon
codons
endocytosis
endoplasmic
exocytosis
golgi
grna
grna
interphase
macinnes
metaphase
mitochondria
mrna
prophase
rna
rrna
telophase
towards
trna

Appendix E

The list of stop words removed from many of the data sets:

a
able
about
across
after
all
almost
also
am
among
an
and
any
are
as
at
be
because
been
but
by
can
cannot
could
dear
did
do
does
either
else
ever
every
for
from
get
got
had
has
have
he
her
hers
him
his
how
however
i
if
in
into
is
it
its
just
least
let
like
likely
may
me
might
most
must
my
neither
no
nor
not
of
off
often
on
only
or
other
our
own
rather
said
say
says
she
should
since
so
some
than
that
the
their
them
then
there
these
they
this
tis
to
too
twas
us
wants
was
we
were
what
when
where
which
while
who
whom
why
will
with
would
yet
you
your

Source: http://www.textfixer.com/resources/common-english-words.php
Used with permission.

Appendix F

Candidate models built for essay set #1.

#	Data set(s)	Model	Quadratic Weighted Kappa
1	ngrams-1-2	SVM (RBF Kernel)	0.779929706
1	ngrams-1-3	Partial Least Squares	0.778922509
1	ngrams-2-3	Partial Least Squares	0.76972849
1	alphabetic	SVM (RBF Kernel)	0.767386615
1	ngrams-2	SVM (RBF Kernel)	0.764279819
1	ngrams-1	SVM (RBF Kernel)	0.752857712
1	pls -2	Partial Least Squares	0.749383346
1	q-grams	Partial Least Squares	0.74805245
1	q-grams	Random Subspaces	0.747835101
1	ngrams-1-2	Bayes Network	0.746180163
1	ngrams-1-2	Naive Bayes	0.745333169
1	alphabetic	SVM (Linear Kernel)	0.745091723
1	ngrams-1-2	SVM (Linear Kernel)	0.744685234
1	alphabetic	Bagged SVM (Linear Kernel)	0.743877377
1	q-grams	Linear Regression	0.743195996
1	ngrams-1	Bagged SVM (Linear Kernel)	0.74046416
1	reg-160, extra	Linear Regression	0.74030169
1	ngrams-1	SVM (Linear Kernel)	0.73928431
1	alphabetic	Bayes Network	0.737776965
1	reg-160, extra	Boosted Stumps	0.737736346
1	ngrams-2	Naive Bayes	0.736089067
1	q-grams	Boosted Stumps	0.734096876
1	alphabetic	Naive Bayes	0.730848432
1	alphabetic	Bagged Trees	0.728758029
1	alphabetic	Cost-sensitive Naive Bayes	0.728360716
1	ngrams-2	Bayes Network	0.727252175
1	ngrams-2	SVM (Linear Kernel)	0.72291366
1	ngrams-1	Bagged Trees	0.720495247
1	ngrams-1	Naive Bayes Tree	0.715700225
1	ngrams-1	Cost-sensitive Naive Bayes	0.714994076
1	ngrams-1	Bayes Network	0.711815069
1	ngrams-1	Naive Bayes	0.709234563
1	alphabetic	Naive Bayes Tree	0.695046667
1	reg-160, extra	Random Subspaces	0.692917959
1	ngrams-1	Logistic Regression	0.67400644
1	reg-160, extra	Boosted Trees	0.665268685
1	q-grams	Boosted Trees	0.664167599
1	alphabetic	Logistic Regression	0.662303121
1	cos-1	K-nearest Neighbors	0.623394882
1	ngrams-2	Radial Basis Function Network	0.537589627
1	ngrams-1-2	Radial Basis Function Network	0.53529324
1	alphabetic	Radial Basis Function Network	0.519991618
1	ngrams-1	Radial Basis Function Network	0.467906261

Appendix G

Candidate models built for essay set #2.

#	Data set(s)	Model	Quadratic Weighted Kappa
2	ngrams-1-3	Partial Least Squares	0.720127154
2	ngrams-1-2	SVM (RBF Kernel)	0.713853306
2	ngrams-2-3	Partial Least Squares	0.691773528
2	ngrams-1-2	Naive Bayes	0.690652099
2	ngrams-1-2	Bayes Network	0.677804208
2	ngrams-1	SVM (RBF Kernel)	0.656573521
2	ngrams-1-2	SVM (Linear Kernel)	0.65224078
2	pls -4	Partial Least Squares	0.651668289
2	ngrams-2	SVM (RBF Kernel)	0.651454906
2	ngrams-2	Naive Bayes	0.651247087
2	alphabetic	SVM (RBF Kernel)	0.63982195
2	reg-160, extra	Boosted Stumps	0.638513982
2	alphabetic	Bagged SVM (Linear Kernel)	0.630863108
2	alphabetic	SVM (Linear Kernel)	0.625728336
2	alphabetic	Bayes Network	0.625343055
2	reg-160, extra	Linear Regression	0.623052304
2	alphabetic	Naive Bayes Tree	0.620040789
2	ngrams-1	Bagged SVM (Linear Kernel)	0.619713106
2	alphabetic	Cost-sensitive Naive Bayes	0.619096022
2	ngrams-1	Naive Bayes	0.619007335
2	ngrams-2	Bayes Network	0.618117425
2	alphabetic	Naive Bayes	0.618008023
2	ngrams-1	Cost-sensitive Naive Bayes	0.617842505
2	q-grams	Partial Least Squares	0.615158794
2	q-grams	Boosted Stumps	0.613580038
2	ngrams-1	Bayes Network	0.613329133
2	ngrams-1	SVM (Linear Kernel)	0.610895176
2	ngrams-2	SVM (Linear Kernel)	0.606374775
2	q-grams	Random Subspaces	0.606323189
2	ngrams-1	Bagged Trees	0.604226946
2	ngrams-1	Naive Bayes Tree	0.594349309
2	alphabetic	Bagged Trees	0.589889192
2	q-grams	Linear Regression	0.588218967
2	alphabetic	Logistic Regression	0.584512965
2	ngrams-1	Logistic Regression	0.579733754
2	reg-160, extra	Random Subspaces	0.576049331
2	reg-160, extra	Boosted Trees	0.498844166
2	q-grams	Boosted Trees	0.492334171
2	ngrams-1	Radial Basis Function Network	0.443717561
2	alphabetic	Radial Basis Function Network	0.406843333
2	cos-2	K-nearest Neighbors	0.394853848
2	ngrams-2	Radial Basis Function Network	0.387093339
2	ngrams-1-2	Radial Basis Function Network	0.365574728

Appendix H

Candidate models built for essay set #3.

#	Data set(s)	Model	Quadratic Weighted Kappa
3	pls -4	Partial Least Squares	0.654060998
3	ngrams-1-2	Bayes Network	0.652765584
3	ngrams-1	Bagged SVM (Linear Kernel)	0.647975078
3	ngrams-1	SVM (Linear Kernel)	0.643727348
3	reg-160, extra	Linear Regression	0.643641476
3	alphabetic	SVM (Linear Kernel)	0.643561179
3	alphabetic	Bayes Network	0.64343564
3	ngrams-1-2	SVM (RBF Kernel)	0.642374057
3	alphabetic	Bagged SVM (Linear Kernel)	0.642217811
3	q-grams	Random Subspaces	0.642006835
3	ngrams-1-3	Partial Least Squares	0.638519346
3	ngrams-1	SVM (RBF Kernel)	0.634987379
3	ngrams-1	Naive Bayes Tree	0.634217287
3	alphabetic	SVM (RBF Kernel)	0.631166391
3	ngrams-1	Bayes Network	0.629226396
3	reg-160, extra	Boosted Stumps	0.627443007
3	q-grams	Partial Least Squares	0.624568697
3	reg-160, extra	Random Subspaces	0.623195543
3	ngrams-1	Cost-sensitive Naive Bayes	0.620442574
3	ngrams-1	Naive Bayes	0.617535613
3	ngrams-1-2	SVM (Linear Kernel)	0.617335543
3	ngrams-1-2	Naive Bayes	0.612156897
3	alphabetic	Logistic Regression	0.611763943
3	alphabetic	Naive Bayes Tree	0.611662043
3	q-grams	Linear Regression	0.609424047
3	alphabetic	Cost-sensitive Naive Bayes	0.601948244
3	alphabetic	Naive Bayes	0.601812372
3	ngrams-1	Logistic Regression	0.598431095
3	ngrams-2	Bayes Network	0.593340721
3	ngrams-2-3	Partial Least Squares	0.582998644
3	q-grams	Boosted Stumps	0.579947994
3	ngrams-2	Naive Bayes	0.576082879
3	ngrams-2	SVM (Linear Kernel)	0.568885703
3	ngrams-1	Bagged Trees	0.565992703
3	ngrams-2	SVM (RBF Kernel)	0.550887051
3	reg-160, extra	Boosted Trees	0.521640064
3	alphabetic	Bagged Trees	0.52146127
3	ngrams-2	Radial Basis Function Network	0.520127352
3	cos-3	K-nearest Neighbors	0.512602777
3	ngrams-1-2	Radial Basis Function Network	0.510051003
3	q-grams	Boosted Trees	0.498043363
3	alphabetic	Radial Basis Function Network	0.468711051
3	ngrams-1	Radial Basis Function Network	0.433854744

Appendix I

Candidate models built for essay set #4.

#	Data set(s)	Model	Quadratic Weighted Kappa
4	alphabetic	Bagged SVM (Linear Kernel)	0.654060998
4	alphabetic	SVM (Linear Kernel)	0.657286399
4	reg-160, extra	Linear Regression	0.649794023
4	q-grams	Random Subspaces	0.649623079
4	pls -2	Partial Least Squares	0.649308919
4	alphabetic	Bayes Network	0.645007665
4	alphabetic	SVM (RBF Kernel)	0.643363499
4	alphabetic	Cost-sensitive Naive Bayes	0.640462444
4	alphabetic	Naive Bayes Tree	0.638950037
4	reg-160, extra	Random Subspaces	0.638733715
4	alphabetic	Naive Bayes	0.635808484
4	reg-160, extra	Boosted Stumps	0.63239169
4	ngrams-1	Bagged SVM (Linear Kernel)	0.628586114
4	ngrams-1-3	Partial Least Squares	0.622750174
4	ngrams-1	SVM (Linear Kernel)	0.621842309
4	ngrams-1	Bayes Network	0.621572692
4	ngrams-1	Cost-sensitive Naive Bayes	0.620375785
4	ngrams-1-2	SVM (RBF Kernel)	0.619395887
4	ngrams-1	SVM (RBF Kernel)	0.617844651
4	ngrams-1	Naive Bayes	0.615805996
4	ngrams-1	Naive Bayes Tree	0.610465084
4	q-grams	Partial Least Squares	0.604795399
4	ngrams-1-2	Naive Bayes	0.602917039
4	q-grams	Boosted Stumps	0.592620147
4	ngrams-1-2	Bayes Network	0.586213524
4	q-grams	Linear Regression	0.580154231
4	ngrams-1	Logistic Regression	0.568152202
4	alphabetic	Logistic Regression	0.567531399
4	alphabetic	Bagged Trees	0.564871197
4	reg-160, extra	Boosted Trees	0.564811736
4	ngrams-1-2	SVM (Linear Kernel)	0.558493659
4	ngrams-1	Bagged Trees	0.548416413
4	ngrams-2-3	Partial Least Squares	0.542279134
4	ngrams-2	SVM (Linear Kernel)	0.518466968
4	ngrams-2	Naive Bayes	0.508136041
4	cos-4	K-nearest Neighbors	0.503399338
4	q-grams	Boosted Trees	0.495173352
4	ngrams-2	Bayes Network	0.484901445
4	ngrams-1	Radial Basis Function Network	0.465503933
4	alphabetic	Radial Basis Function Network	0.459078311
4	ngrams-1-2	Radial Basis Function Network	0.454967711
4	ngrams-2	Radial Basis Function Network	0.448214755
4	ngrams-2	SVM (RBF Kernel)	0.406093851

Appendix J

Candidate models built for essay set #5.

#	Data set(s)	Model	Quadratic Weighted Kappa
5	ngrams-1-2	SVM (RBF Kernel)	0.758390713
5	ngrams-1-3	Partial Least Squares	0.753698462
5	ngrams-2-3	Partial Least Squares	0.752900786
5	ngrams-2	SVM (RBF Kernel)	0.75235842
5	alphabetic	SVM (RBF Kernel)	0.749067337
5	reg-120, extra	Linear Regression	0.748142232
5	pls -8	Partial Least Squares	0.747394708
5	ngrams-1	SVM (RBF Kernel)	0.741412893
5	ngrams-2	Bayes Network	0.737462643
5	q-grams	Random Subspaces	0.734517168
5	reg-120, extra	Random Subspaces	0.726552225
5	ngrams-1	Bagged Trees	0.718879868
5	alphabetic	Bagged SVM (Linear Kernel)	0.718648006
5	ngrams-1-2	Bayes Network	0.718455128
5	alphabetic	SVM (Linear Kernel)	0.716305235
5	reg-120, extra	Boosted Stumps	0.715540226
5	alphabetic	Bagged Trees	0.71538724
5	ngrams-1	Bagged SVM (Linear Kernel)	0.706379902
5	ngrams-1	Bayes Network	0.705023599
5	q-grams	Partial Least Squares	0.704264036
5	q-grams	Linear Regression	0.702265429
5	ngrams-1-2	SVM (Linear Kernel)	0.690914893
5	cos-5	K-nearest Neighbors	0.686068718
5	ngrams-1	SVM (Linear Kernel)	0.683098008
5	alphabetic	Bayes Network	0.670966456
5	ngrams-2	SVM (Linear Kernel)	0.666353785
5	q-grams	Boosted Stumps	0.66551353
5	reg-120, extra	Boosted Trees	0.65013967
5	ngrams-1	Cost-sensitive Naive Bayes	0.648057256
5	ngrams-1	Naive Bayes	0.646867597
5	ngrams-2	Naive Bayes	0.64606823
5	ngrams-1-2	Naive Bayes	0.642444573
5	q-grams	Boosted Trees	0.636729347
5	alphabetic	Cost-sensitive Naive Bayes	0.6317879
5	alphabetic	Naive Bayes	0.630549531
5	ngrams-1	Logistic Regression	0.604893759
5	ngrams-1	Naive Bayes Tree	0.599842318
5	alphabetic	Logistic Regression	0.597531581
5	ngrams-1-2	Radial Basis Function Network	0.53913713
5	alphabetic	Naive Bayes Tree	0.52602908
5	ngrams-2	Radial Basis Function Network	0.52509636
5	alphabetic	Radial Basis Function Network	0.505545657
5	ngrams-1	Radial Basis Function Network	0.460586734

Appendix K

Candidate models built for essay set #6.

#	Data set(s)	Model	Quadratic Weighted Kappa
6	ngrams-1	SVM (RBF Kernel)	0.809243468
6	alphabetic	SVM (RBF Kernel)	0.808576499
6	ngrams-1-2	SVM (RBF Kernel)	0.805532263
6	q-grams	Random Subspaces	0.801649061
6	ngrams-1-3	Partial Least Squares	0.800310828
6	ngrams-1-2	SVM (Linear Kernel)	0.800306866
6	pls -4	Partial Least Squares	0.799413204
6	ngrams-1	Bagged Trees	0.790705948
6	ngrams-1-2	Bayes Network	0.789520712
6	alphabetic	Bagged Trees	0.789072168
6	alphabetic	Bagged SVM (Linear Kernel)	0.787296471
6	alphabetic	SVM (Linear Kernel)	0.78712017
6	ngrams-2	SVM (RBF Kernel)	0.783976475
6	ngrams-1	Bagged SVM (Linear Kernel)	0.78298978
6	q-grams	Boosted Trees	0.772917916
6	ngrams-2	Bayes Network	0.772164448
6	ngrams-1	Bayes Network	0.769624768
6	reg-120, extra	Linear Regression	0.768214536
6	reg-120, extra	Random Subspaces	0.767943976
6	cos-6	K-nearest Neighbors	0.765403528
6	q-grams	Partial Least Squares	0.764405888
6	ngrams-2	SVM (Linear Kernel)	0.76144846
6	ngrams-2-3	Partial Least Squares	0.757116093
6	alphabetic	Bayes Network	0.75251375
6	reg-120, extra	Boosted Stumps	0.752265997
6	q-grams	Linear Regression	0.748513429
6	ngrams-1	SVM (Linear Kernel)	0.745082149
6	reg-120, extra	Boosted Trees	0.744052211
6	ngrams-1	Cost-sensitive Naive Bayes	0.741238498
6	ngrams-1	Naive Bayes	0.739782803
6	q-grams	Boosted Stumps	0.729146076
6	ngrams-1-2	Naive Bayes	0.702344283
6	alphabetic	Naive Bayes Tree	0.689026237
6	alphabetic	Cost-sensitive Naive Bayes	0.6863128
6	ngrams-1	Naive Bayes Tree	0.686259519
6	alphabetic	Naive Bayes	0.676493879
6	ngrams-2	Naive Bayes	0.6714972
6	ngrams-2	Radial Basis Function Network	0.60158577
6	ngrams-1	Logistic Regression	0.5801314
6	ngrams-1-2	Radial Basis Function Network	0.576700399
6	alphabetic	Logistic Regression	0.436885762
6	ngrams-1	Radial Basis Function Network	0.313730302
6	alphabetic	Radial Basis Function Network	0.293912066

Appendix L

Candidate models built for essay set #7.

#	Data set(s)	Model	Quadratic Weighted Kappa
7	Alphabetic	Bagged SVM (Linear Kernel)	0.609802013
7	pls -4	Partial Least Squares	0.60614047
7	reg-160, extra	Linear Regression	0.588010569
7	ngrams-1-3	Partial Least Squares	0.576579102
7	Alphabetic	Bagged Trees	0.576142028
7	ngrams-1	Bagged SVM (Linear Kernel)	0.57391419
7	q-grams	Random Subspaces	0.571613017
7	reg-160, extra	Boosted Stumps	0.571473605
7	Alphabetic	SVM (Linear Kernel)	0.570259804
7	q-grams	Boosted Stumps	0.560444593
7	ngrams-1-2	SVM (RBF Kernel)	0.558275152
7	Alphabetic	SVM (RBF Kernel)	0.556876234
7	ngrams-1-2	SVM (Linear Kernel)	0.550373281
7	ngrams-1	SVM (Linear Kernel)	0.549104304
7	ngrams-1-2	Bayes Network	0.545760715
7	ngrams-2-3	Partial Least Squares	0.542911954
7	q-grams	Linear Regression	0.542591719
7	q-grams	Partial Least Squares	0.541922776
7	ngrams-1	Bagged Trees	0.534415088
7	Alphabetic	Bayes Network	0.519105222
7	ngrams-1	Logistic Regression	0.517438602
7	Alphabetic	Logistic Regression	0.51649297
7	ngrams-1	SVM (RBF Kernel)	0.514866431
7	cos-7	K-nearest Neighbors	0.514862501
7	reg-160, extra	Random Subspaces	0.514301537
7	ngrams-2	Bayes Network	0.512400172
7	Alphabetic	Naive Bayes Tree	0.507899079
7	ngrams-1	Cost-sensitive Naive Bayes	0.505614008
7	ngrams-1	Naive Bayes	0.505083578
7	ngrams-2	SVM (Linear Kernel)	0.50334772
7	ngrams-2	SVM (RBF Kernel)	0.500461111
7	ngrams-1	Bayes Network	0.500077388
7	ngrams-1-2	Naive Bayes	0.481776407
7	alphabetic	Cost-sensitive Naive Bayes	0.480833484
7	alphabetic	Naive Bayes	0.479431855
7	ngrams-1	Naive Bayes Tree	0.478480102
7	reg-160, extra	Boosted Trees	0.476745652
7	ngrams-1-2	Radial Basis Function Network	0.448067262
7	q-grams	Boosted Trees	0.445278254
7	ngrams-2	Naive Bayes	0.444566473
7	ngrams-2	Radial Basis Function Network	0.436984911
7	alphabetic	Radial Basis Function Network	0.373836846
7	ngrams-1	Radial Basis Function Network	0.156719651

Appendix M

Candidate models built for essay set #8.

#	Data set(s)	Model	Quadratic Weighted Kappa
8	ngrams-1-2	SVM (RBF Kernel)	0.586837221
8	ngrams-1-2	Bayes Network	0.574584491
8	ngrams-1-3	Partial Least Squares	0.570425374
8	ngrams-1	SVM (RBF Kernel)	0.569856628
8	ngrams-1-2	SVM (Linear Kernel)	0.559576399
8	ngrams-1	SVM (Linear Kernel)	0.556331365
8	pls-4	Partial Least Squares	0.55596663
8	ngrams-1	Naive Bayes Tree	0.554628539
8	ngrams-1	Bagged SVM (Linear Kernel)	0.554427988
8	ngrams-2	Bayes Network	0.549243895
8	ngrams-1	Bagged Trees	0.54487999
8	alphabetic	Bagged SVM (Linear Kernel)	0.544529617
8	ngrams-1	Cost-sensitive Naive Bayes	0.539374961
8	ngrams-1	Naive Bayes	0.538767524
8	reg-160, extra	Linear Regression	0.536604913
8	alphabetic	SVM (RBF Kernel)	0.535125321
8	alphabetic	Bayes Network	0.532473595
8	alphabetic	Naive Bayes Tree	0.53147557
8	reg-160, extra	Boosted Stumps	0.529212076
8	ngrams-1	Bayes Network	0.521240712
8	alphabetic	Naive Bayes	0.521115198
8	alphabetic	SVM (Linear Kernel)	0.520167191
8	ngrams-1-2	Naive Bayes	0.518311927
8	alphabetic	Cost-sensitive Naive Bayes	0.518023463
8	ngrams-2-3	Partial Least Squares	0.515807693
8	ngrams-2	SVM (Linear Kernel)	0.515222945
8	q-grams	Random Subspaces	0.514656481
8	ngrams-2	SVM (RBF Kernel)	0.513181867
8	ngrams-1	Logistic Regression	0.510336174
8	alphabetic	Bagged Trees	0.508998181
8	q-grams	Partial Least Squares	0.493480154
8	q-grams	Boosted Stumps	0.493465938
8	ngrams-2	Naive Bayes	0.490339221
8	reg-160, extra	Random Subspaces	0.481502453
8	q-grams	Linear Regression	0.472028723
8	q-grams	Boosted Trees	0.471423842
8	alphabetic	Logistic Regression	0.451965393
8	cos-8	K-nearest Neighbors	0.424225419
8	ngrams-2	Radial Basis Function Network	0.397693403
8	reg-160, extra	Boosted Trees	0.391960745
8	ngrams-1-2	Radial Basis Function Network	0.387565148
8	alphabetic	Radial Basis Function Network	0.357974348
8	ngrams-1	Radial Basis Function Network	0.336262569

Appendix N

Candidate models built for essay set #9.

#	Data set(s)	Model	Quadratic Weighted Kappa
9	ngrams-1-2	SVM (RBF Kernel)	0.744690045
9	ngrams-1	SVM (RBF Kernel)	0.735886766
9	alphabetic	SVM (RBF Kernel)	0.734740044
9	alphabetic	Bagged Trees	0.72723741
9	ngrams-1	Bagged SVM (Linear Kernel)	0.72254638
9	ngrams-1	SVM (Linear Kernel)	0.721748401
9	ngrams-1	Bagged Trees	0.717481879
9	ngrams-1	Bayes Network	0.707853538
9	alphabetic	SVM (Linear Kernel)	0.705548167
9	alphabetic	Bagged SVM (Linear Kernel)	0.704008952
9	ngrams-1-2	Bayes Network	0.701089347
9	ngrams-1-2	SVM (Linear Kernel)	0.699790798
9	alphabetic	Bayes Network	0.689423847
9	ngrams-1-3	Partial Least Squares	0.684899106
9	ngrams-1-2	Naive Bayes	0.677466874
9	ngrams-1	Naive Bayes Tree	0.677327374
9	ngrams-2	Bayes Network	0.676056723
9	ngrams-1	Naive Bayes	0.674036935
9	ngrams-1	Cost-sensitive Naive Bayes	0.672964436
9	alphabetic	Naive Bayes	0.668319834
9	alphabetic	Naive Bayes Tree	0.666306189
9	reg-160, extra	Boosted Stumps	0.666304564
9	alphabetic	Cost-sensitive Naive Bayes	0.666286048
9	pls-32	Partial Least Squares	0.666232496
9	q-grams	Boosted Trees	0.664472535
9	reg-160, extra	Linear Regression	0.664163002
9	reg-160, extra	Boosted Trees	0.662779596
9	cos-9	K-nearest Neighbors	0.657695021
9	q-grams	Boosted Stumps	0.657113668
9	ngrams-2	Naive Bayes	0.645307988
9	ngrams-2	SVM (RBF Kernel)	0.640432136
9	ngrams-2	SVM (Linear Kernel)	0.637873374
9	q-grams	Random Subspaces	0.629982477
9	ngrams-2-3	Partial Least Squares	0.627947548
9	q-grams	Partial Least Squares	0.610322004
9	reg-160, extra	Random Subspaces	0.607266574
9	ngrams-1	Logistic Regression	0.598874324
9	q-grams	Linear Regression	0.568039359
9	alphabetic	Logistic Regression	0.511281643
9	ngrams-2	Radial Basis Function Network	0.491649802
9	ngrams-1-2	Radial Basis Function Network	0.473799594
9	ngrams-1	Radial Basis Function Network	0.464856187
9	alphabetic	Radial Basis Function Network	0.40044514

Appendix O

Candidate models built for essay set #10.

#	Data set(s)	Model	Quadratic Weighted Kappa
10	ngrams-1, cat-color	Bayes Network	0.704549239
10	ngrams-1-2, cat-color	SVM (RBF Kernel)	0.703975745
10	ngrams-1, cat-color	SVM (RBF Kernel)	0.69995709
10	ngrams-1-2, cat-color	Naive Bayes	0.696455038
10	ngrams-1-2, cat-color	Bayes Network	0.695154457
10	ngrams-1, cat-color	Bagged Trees	0.69376916
10	alphabetic, cat-color	SVM (RBF Kernel)	0.693208044
10	ngrams-1, cat-color	Cost-sensitive Naive Bayes	0.690653973
10	ngrams-1, cat-color	Naive Bayes	0.689791474
10	alphabetic, cat-color	Bayes Network	0.687180002
10	alphabetic, cat-color	Bagged Trees	0.683720634
10	ngrams-1, cat-color	Naive Bayes Tree	0.679854404
10	ngrams-1, cat-color	Bagged SVM (Linear Kernel)	0.678262461
10	ngrams-1, cat-color	SVM (Linear Kernel)	0.672233837
10	alphabetic, cat-color	Cost-sensitive Naive Bayes	0.668813324
10	alphabetic, cat-color	Naive Bayes	0.668071232
10	ngrams-1-2, cat-color	SVM (Linear Kernel)	0.659638054
10	ngrams-2, cat-color	Bayes Network	0.658596216
10	alphabetic, cat-color	Bagged SVM (Linear Kernel)	0.657001247
10	alphabetic, cat-color	SVM (Linear Kernel)	0.655965927
10	ngrams-2, cat-color	Naive Bayes	0.65242671
10	ngrams-1-3, bin-color	Partial Least Squares	0.632023234
10	alphabetic, cat-color	Naive Bayes Tree	0.627249357
10	pls-2, bin -color	Partial Least Squares	0.627246253
10	q-grams, bin -color	Boosted Stumps	0.623831526
10	q-grams, bin -color	Boosted Trees	0.618778663
10	ngrams-2, cat-color	SVM (Linear Kernel)	0.616825399
10	cos-10, cat-color	K-nearest Neighbors	0.614374074
10	reg-160, extra, bin-color, cat-color	Boosted Stumps	0.609298089
10	reg-160, extra, bin-color, cat-color	Boosted Trees	0.6059193
10	ngrams-2, cat-color	SVM (RBF Kernel)	0.602881952
10	ngrams-1, cat-color	Logistic Regression	0.588642332
10	ngrams-2-3, bin -color	Partial Least Squares	0.569882578
10	q-grams, bin -color	Partial Least Squares	0.564079496
10	q-grams, bin -color	Linear Regression	0.547562509
10	ngrams-1-2, cat-color	Radial Basis Function Network	0.537864328
10	alphabetic, cat-color	Logistic Regression	0.531885526
10	ngrams-2, cat-color	Radial Basis Function Network	0.526622453
10	q-grams, bin-color	Random Subspaces	0.524671521
10	alphabetic, cat-color	Radial Basis Function Network	0.435442571
10	ngrams-1, cat-color	Radial Basis Function Network	0.317232943