Disciplina: Data Mining **Data:** 6/12/2016

Professor: Geraldo Zimbrão da Silva

Aluno: Leniel Macaferi **DRE:** 116005784

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO UFRJ

Mestrado em Engenharia de Dados e Conhecimento @ COPPE\UFRJ - Universidade Federal do Rio de Janeiro

Report

Classifying Kaggle's IMDB 5000 Movie Dataset

Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify loan applicants as low, medium, or high credit risks.
https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/classify.htm

1 - Problem

Classify the data contained in a spreadsheet called movie_metadata.xlsx using multiple classifiers having the Naive Bayes classifier as the baseline.

The expected result is to predict the movie **rating** for unlabelled\unclassified movie samples (spreadsheet rows) identified with a question mark ?.

2 - Data

The dataset used was the IMDB 5000 Movie Dataset with 5000+ movie data scraped from IMDB website.

The reduced data used for training\testing the model comprises 11 features (spreadsheet columns):

- duration
- director_facebook_likes
- actor_1_facebook_likes
- actor_2_facebook_likes
- actor_3_facebook_likes
- movie_facebook_likes
- cast_total_facebook_likes
- facenumber_in_poster
- budget
- gross
- Imdb_score (**rating**)

These columns were chosen because they best represent the data and lead to better predictions.

The **imbd_score** feature comprises values ranging from 0 to 10. The data was normalized to a column called **rating** which comprises values from A to E according to the following conversion

table:

rating	imdb_rating
A	>= 8
B	>= 6 and < 8
C	>= 4 and < 6
D	>= 2 and < 4
E	>= 0 and < 2

Since all the samples from the original dataset had a rating (imdb_rating) assigned, I randomly omitted the rating for almost 21% of the the movies placing a question mark [?] on the rating column. These are the ones used as the test input to the classifiers. The original ratings will be used to compare the classifiers' prediction accuracy. Original rating are stored in the sheet called Massaged in movie_metadata.xlsx.

From the 5038 samples\instances\rows in the spreadsheet, 3982 are already classified into 5 rating classes [A, B, C, D, E] and 1056 need to be classified.

The following table displays the number of instances in each class along with the number of samples to be classified. The percentage in each group is also shown.

A	B	C	D	E	?
263	2575	1026	116	2	1056
5.22%	51.11%	20.36%	2.30%	0.04%	20.96%

Table 1 - Number and percentage of samples\instances per class

3 - Development

3.1 - Tools

The following is the listing of tools used to execute this work:

Microsoft Excel

IDEs: Weka

All the files including the input and output of for this work is stored on a Github repository @ https://github.com/leniel/DataMining

3.2 - Process

Data\number crunching was done in Microsoft Excel before serving as input to the classifiers' algorithms.

Data was partitioned into 2 groups: classified and unclassified instances (spreadsheet rows).

The first group containing only instances already classified was taken out and saved in the .csv format in a file called movie_metadata_training.csv.

The second group containing only samples unclassified represented by an interrogation mark [?] was taken out and saved in the .CSV format in a file called movie_metadata_test.csv.

Later on the two CSV files were imported into Weka.

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

Figure 1 - Weka's GUI Chooser

Using Weka's Explorer application the training data was imported for preprocessing. To play with the data I converted it to Weka's own file format called arff (Attribute-Relation File Format) to make sure the data is ok. The new file is called movie_metadata_training.arff.

Training data was used to build the model from which prediction will be made using the test data.

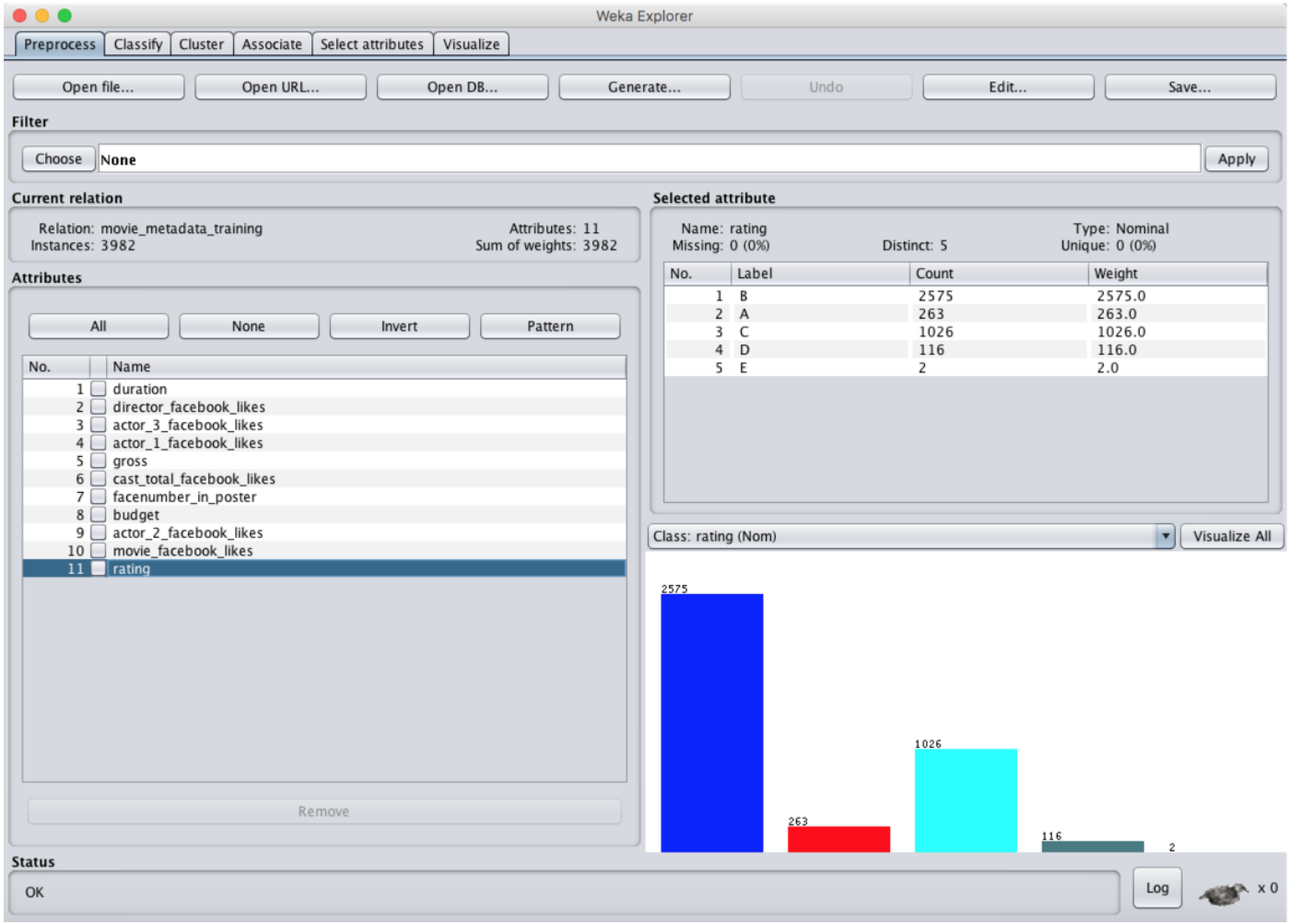

Figure 2 - Weka's Preprocess tab\window listing attributes used to train the model

3.3 - Classification

Classification was performed using the Classify window in Weka.

A total of 15 different classifiers from multiple types were tested:

- **Bayes**: Bayes Net, Naive Bayes
- **Functions**: LibSVM (SVM), Multilayer Perceptron, SMO
- **Lazy**: IBk (KNN), KStar, LWL
- **Meta**: Bagging, Classification via Regression
- **Rules**: JRip, PART
- **Trees**: j48, LMT, Random Forest

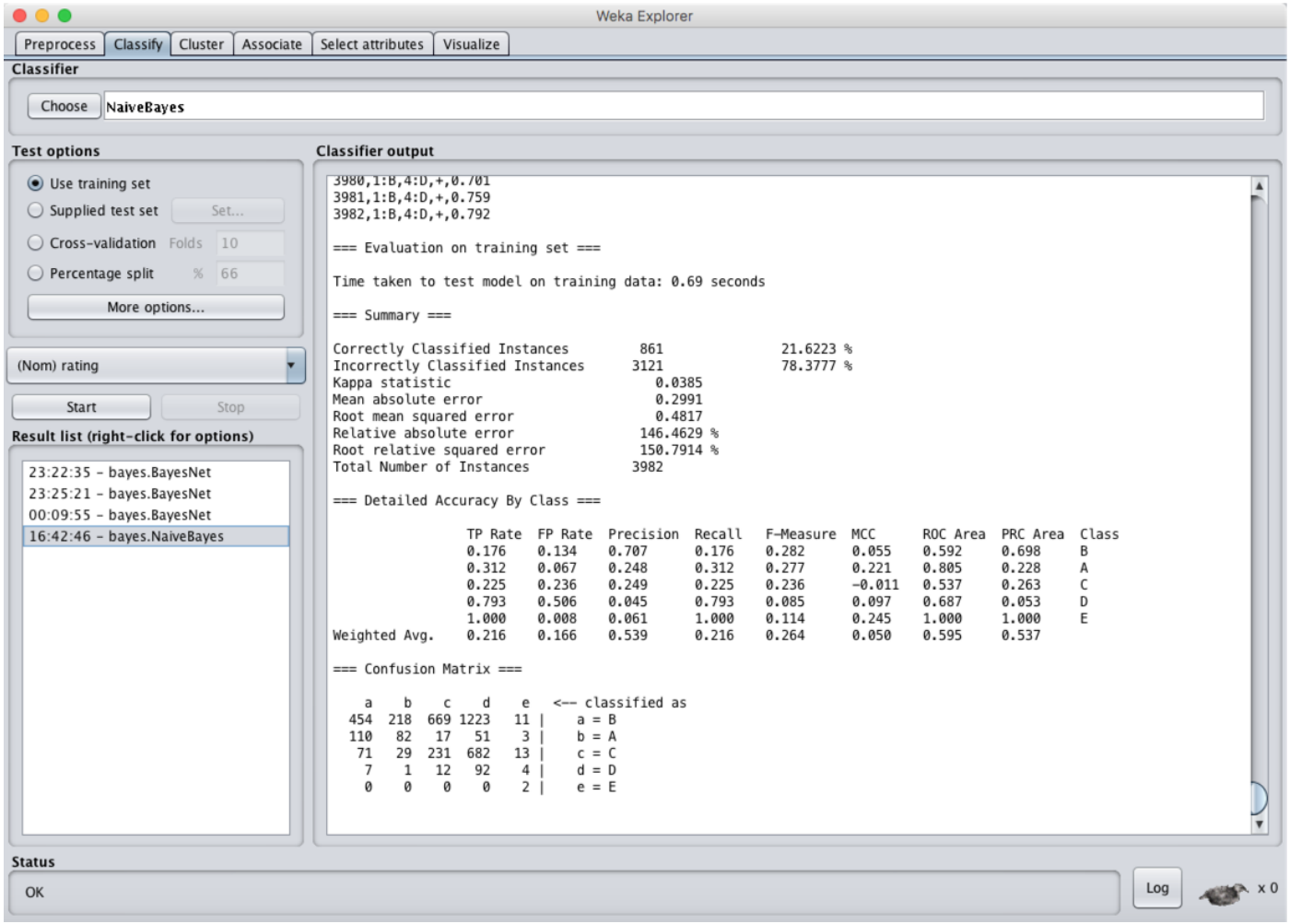

Figure 3 - Weka's Classify tab\window with Naive Bayes classifier chosen

For each classifier, the model was built using the training data (movie_metadata_training.arff) and classifier output data was taken out to a new spreadsheet called classifiers_predictions.xlsx. In this spreadsheet further analysis of the output was performed.

3.4 - Predictions

For each classifier, the test file movie_metadata_test.arff was provided in Test Options (Supplied test set) in Weka.

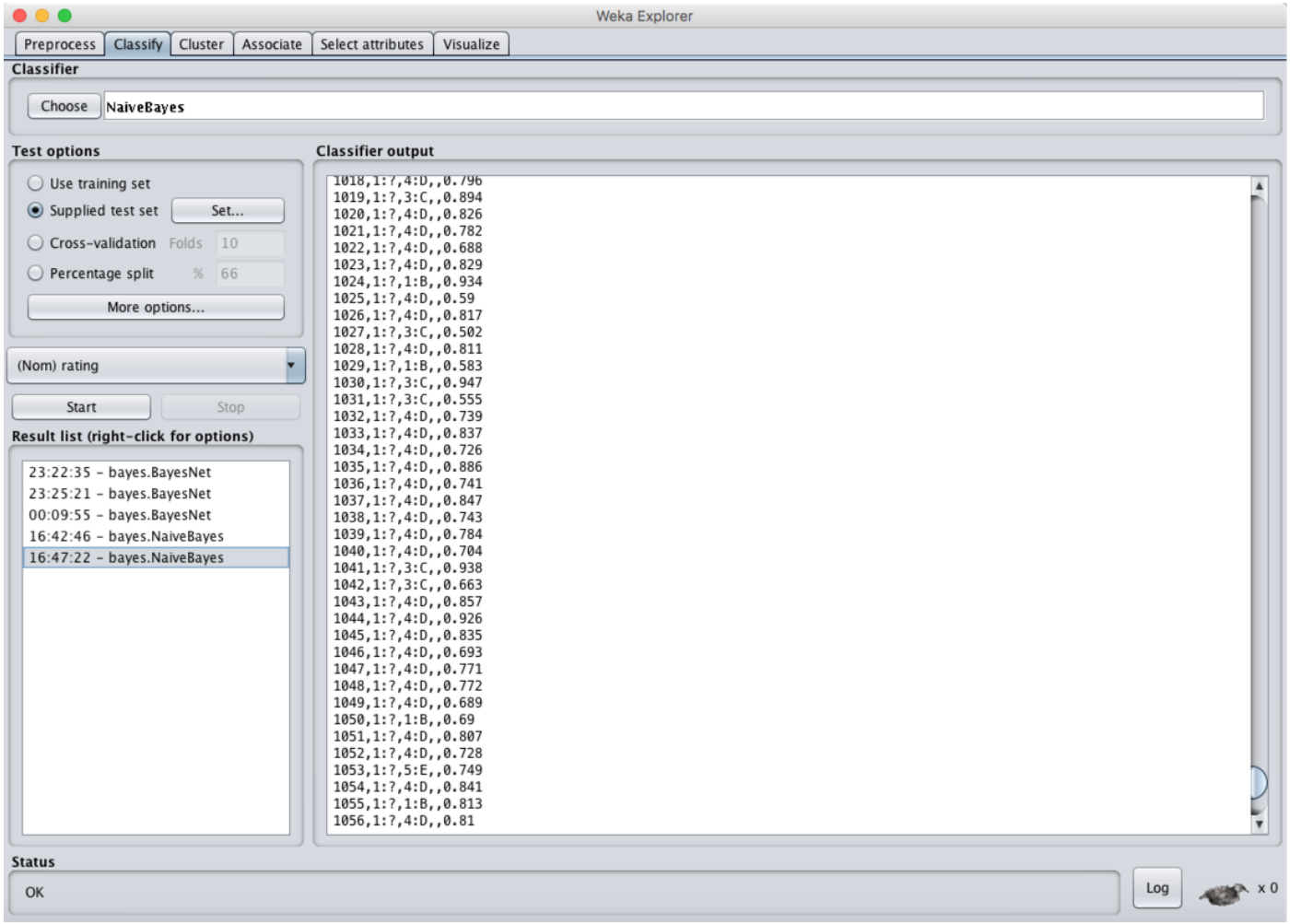

Figure 4 - Weka's Supplied test set option

While inside Weka's Classify windows, In More options... the output was set to a CSV file called predictions.csv according to Figure 5 below:

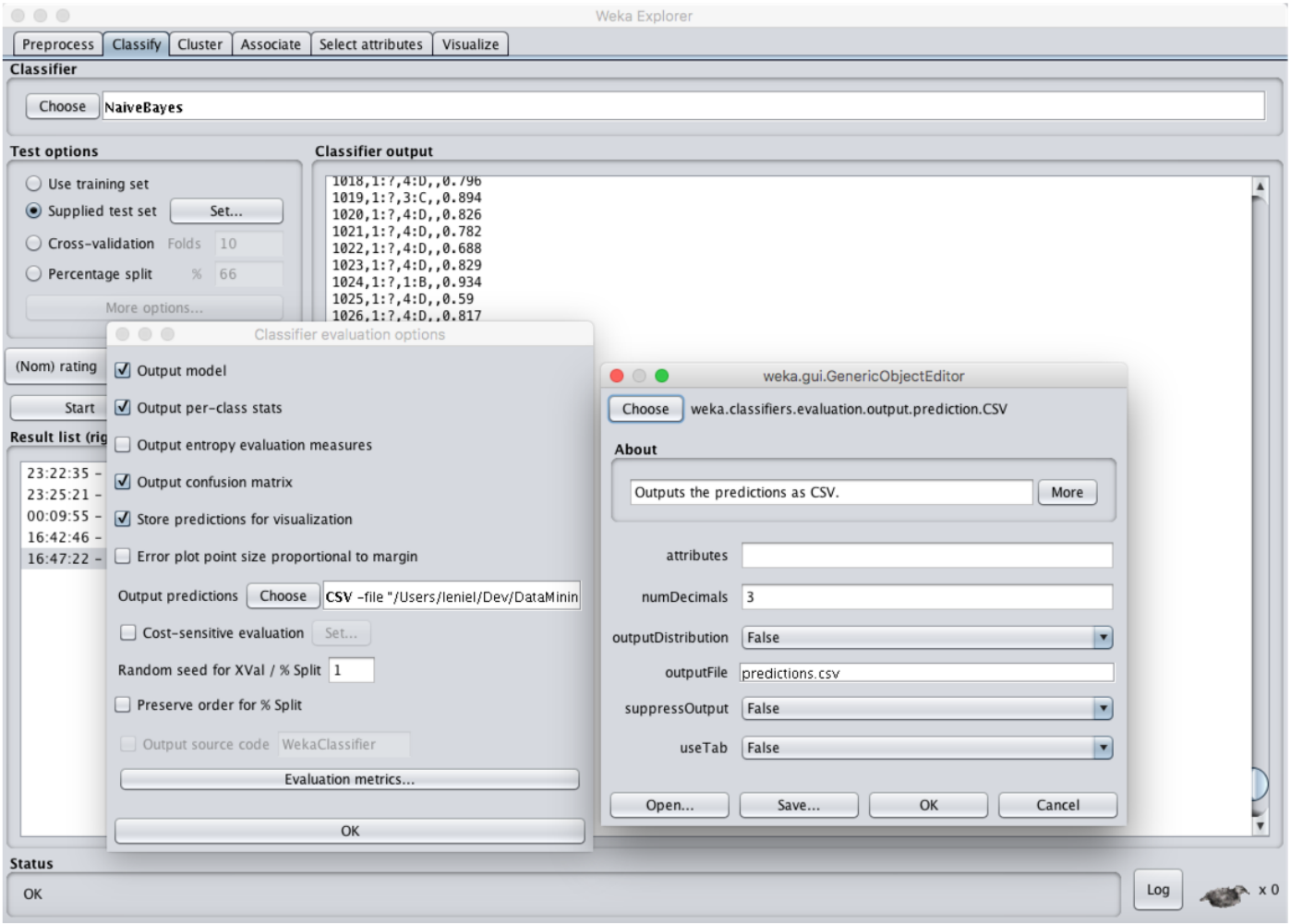

Figure 5 - Setting the predictions output

The data in the predictions.csv was then transferred to the classifiers_predictions.xlsx for further analysis.

4 - Results

The results are based on classification using both the training set versus 5-fold cross-validation for all the 15 classifiers listed in 3.3. All classifications were run with default parameters.

4.1 - Correctly classified instances (%)

KStar, LibSVN and Random Forest scored 100% when using the Training set.

JRIP (70.16%) was the best performing classifier when using 5-fold cross-validation. It repeated the same % shown when using the Training set.

The chart in Figure 6 shows the output for each classifier comparing the classification with Training Set versus 5-fold cross-validation.

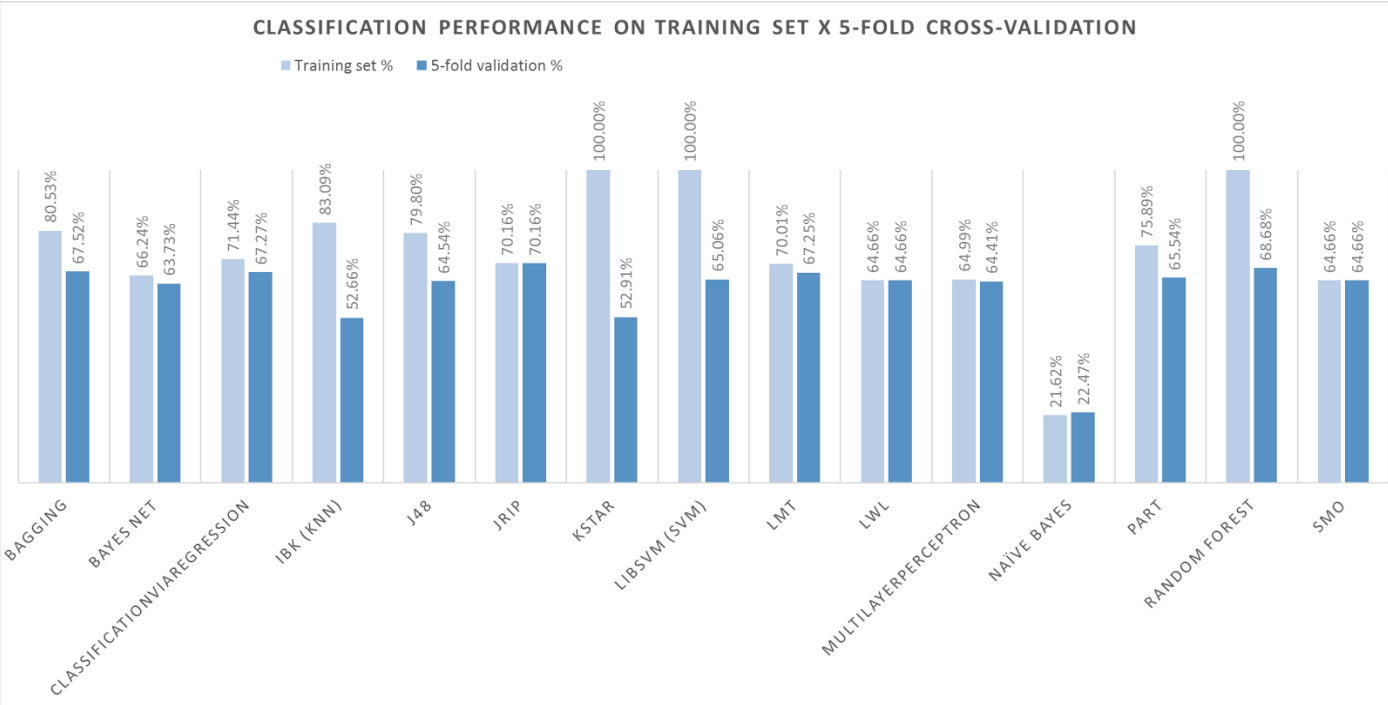

Figure 6 - Chart depicting the classifiers' accuracy

#	Classifier	% Correctly classified	# of correctly classified
1º	JRip	70.16%	2794
2º	Random Forest	68.68%	2735
3º	Bagging	67.52%	2689
4º	Classification via Regression	67.27%	2679
5º	LMT	67.25%	2678
6º	PART	65.54%	2610
7º	LibSVM (SVM)	65.06%	2591
8º	LWL	64.66%	2575
9º	SMO	64.66%	2575
10º	J48	64.54%	2570
11º	MultilayerPerceptron	64.41%	2565
12º	Bayes Net	63.73%	2538
13º	KStar	52.91%	2107
14º	IBk (KNN)	52.66%	2097
15º	Naïve Bayes	22.47%	895

Table 2 - 5-fold cross-validation best performing classifiers

4.2 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the correct values\classes are known.

Table 3 shows the confusion matrices obtained:

Training set

Classifier	A	B	C	D	E	Ratin
Bagging	91	168	4	0	0	A
Bagging	9	2511	55	0	0	B
Bagging	1	431	594	0	0	C
Bagging	0	72	33	11	0	D
Bagging	0	2	0	0	0	E
Bayes Net	71	183	7	2	0	A
Bayes Net	62	2044	445	24	0	B
Bayes Net	2	489	516	19	0	C
Bayes Net	0	39	70	7	0	D
Bayes Net	0	0	2	0	0	E
ClassificationViaRegression	56	202	5	0	0	A
ClassificationViaRegression	4	2417	154	0	0	B
ClassificationViaRegression	0	654	372	0	0	C
ClassificationViaRegression	0	70	46	0	0	D
ClassificationViaRegression	0	0	2	0	0	E
IBk (KNN)	223	40	0	0	0	A
IBk (KNN)	336	2233	6	0	0	B
IBk (KNN)	193	53	780	0	0	C
IBk (KNN)	37	7	0	72	0	D
IBk (KNN)	1	0	0	0	1	E
J48	114	140	8	1	0	A
J48	8	2430	135	2	0	B
J48	4	410	611	1	0	C
J48	1	48	44	23	0	D
J48	0	1	1	0	0	E
JRip	96	165	2	0	0	A
JRip	64	2387	124	0	0	B
JRip	2	713	311	0	0	C
JRip	0	97	19	0	0	D
JRip	0	2	0	0	0	E
KStar	263	0	0	0	0	A
KStar	0	2575	0	0	0	B
KStar	0	0	1026	0	0	C
KStar	0	0	0	116	0	D
KStar	0	0	0	0	2	E
LibSVM (SVM)	263	0	0	0	0	A
LibSVM (SVM)	0	2575	0	0	0	B
LibSVM (SVM)	0	0	1026	0	0	C
LibSVM (SVM)	0	0	0	116	0	D
LibSVM (SVM)	0	0	0	0	2	E
LMT	87	172	4	0	0	A
LMT	29	2390	155	1	0	B
LMT	1	715	310	0	0	C
LMT	0	79	37	0	0	D
LMT	0	1	0	0	1	E
LWL	0	263	0	0	0	A
LWL	0	2575	0	0	0	B
LWL	0	1026	0	0	0	C
LWL	0	116	0	0	0	D
LWL	0	2	0	0	0	E
MultilayerPerceptron	72	179	12	0	0	A
MultilayerPerceptron	9	2511	55	0	0	B
MultilayerPerceptron	1	431	594	0	0	C
MultilayerPerceptron	0	72	33	11	0	D
MultilayerPerceptron	0	2	0	0	0	E
Naïve Bayes	82	110	17	51	3	A
Naïve Bayes	218	454	669	1223	11	B
Naïve Bayes	29	71	231	682	13	C
Naïve Bayes	1	7	12	92	4	D
Naïve Bayes	0	0	0	0	2	E
PART	139	117	6	1	0	A
PART	8	2359	206	2	0	B
PART	4	512	510	0	0	C
PART	0	43	59	14	0	D
PART	0	1	1	0	0	E
Random Forest	263	0	0	0	0	A
Random Forest	0	2575	0	0	0	B
Random Forest	0	0	1026	0	0	C
Random Forest	0	0	0	116	0	D
Random Forest	0	0	0	0	2	E
SMO	0	263	0	0	0	A
SMO	0	2575	0	0	0	B
SMO	0	1026	0	0	0	C
SMO	0	116	0	0	0	D
SMO	0	2	0	0	0	E

5-folds cross-validation

Classifier	A	B	C	D	E	Rating
Bagging	72	185	6	0	0	A
Bagging	31	2291	252	1	0	B
Bagging	3	697	326	0	0	C
Bagging	0	79	37	0	0	D
Bagging	0	0	2	0	0	E
Bayes Net	48	204	11	0	0	A
Bayes Net	62	1996	501	16	0	B
Bayes Net	3	524	491	8	0	C
Bayes Net	0	38	75	3	0	D
Bayes Net	0	0	2	0	0	E
ClassificationViaRegression	42	215	6	0	0	A
ClassificationViaRegression	18	2394	161	2	0	B
ClassificationViaRegression	1	782	243	0	0	C
ClassificationViaRegression	1	85	30	0	0	D
ClassificationViaRegression	0	1	1	0	0	E
IBk (KNN)	98	153	12	0	0	A
IBk (KNN)	396	1686	452	41	0	B
IBk (KNN)	165	523	308	30	0	C
IBk (KNN)	31	51	29	5	0	D
IBk (KNN)	1	1	0	0	0	E
J48	71	178	13	1	0	A
J48	86	2131	343	15	0	B
J48	8	638	367	13	0	C
J48	1	72	42	1	0	D
J48	0	1	1	0	0	E
JRip	96	165	2	0	0	A
JRip	64	2387	124	0	0	B
JRip	2	713	311	0	0	C
JRip	0	97	19	0	0	D
JRip	0	2	0	0	0	E
KStar	53	183	23	4	0	A
KStar	238	1801	456	78	2	B
KStar	70	667	248	41	0	C
KStar	5	67	39	5	0	D
KStar	1	1	0	0	0	E
LibSVM (SVM)	2	261	0	0	0	A
LibSVM (SVM)	0	2575	0	0	0	B
LibSVM (SVM)	0	1014	12	0	0	C
LibSVM (SVM)	0	114	0	2	0	D
LibSVM (SVM)	0	2	0	0	0	E
LMT	53	202	7	0	1	A
LMT	50	2298	224	1	2	B
LMT	3	692	327	1	3	C
LMT	2	65	49	0	0	D
LMT	0	0	2	0	0	E
LWL	0	263	0	0	0	A
LWL	0	2575	0	0	0	B
LWL	0	1026	0	0	0	C
LWL	0	116	0	0	0	D
LWL	0	2	0	0	0	E
MultilayerPerceptron	26	231	6	0	0	A
MultilayerPerceptron	20	2402	152	1	0	B
MultilayerPerceptron	2	887	137	0	0	C
MultilayerPerceptron	0	97	19	0	0	D
MultilayerPerceptron	0	1	1	0	0	E
Naïve Bayes	79	108	22	53	1	A
Naïve Bayes	219	468	707	1178	3	B
Naïve Bayes	30	75	260	655	6	C
Naïve Bayes	1	8	17	88	2	D
Naïve Bayes	0	0	0	2	0	E
PART	67	189	7	0	0	A
PART	68	2154	344	9	0	B
PART	10	623	387	6	0	C
PART	0	78	36	2	0	D
PART	0	1	1	0	0	E
Random Forest	60	200	3	0	0	A
Random Forest	34	2330	211	0	0	B
Random Forest	2	681	343	0	0	C
Random Forest	0	74	40	2	0	D
Random Forest	0	1	1	0	0	E
SMO	0	263	0	0	0	A
SMO	0	2575	0	0	0	B
SMO	0	1026	0	0	0	C
SMO	0	116	0	0	0	D
SMO	0	2	0	0	0	E

Table 3 - Classifiers' confusion matrices

From the output we can see that KStar, LibSVM (SVM) and Random Forest are the best performing classifiers when using the Training set alone.

Classifier	A	B	C	D	E	Rating
JRip	96	165	2	0	0	A
JRip	64	2387	124	0	0	B
JRip	2	713	311	0	0	C
JRip	0	97	19	0	0	D
JRip	0	2	0	0	0	E

Figure 7 - JRip's confusion matrix

JRip was the best performing classifier when doing 5-fold cross-validation as shown in Figure 7 above.

Using Conditional Formatting with Color Scales in Microsoft Excel it's easier to spot that the classifiers had best performance overall while classifying **B** class samples in dark green since they represent the group with more instances in the dataset (51.11% or almost 1/2 of all samples).

On the other hand class D and E lack representation in the dataset. That way it's difficult for the classifiers to correctly classify samples of those classes.

4.3 - Predictions

Predictions were collected for all 15 classifiers listed in 3.3.

They're stored as sheets in the file classifiers_predictions.xlsx. One sheet per Classifier.

The predicted values are compared against the known\expected rating values stored in the sheet Massaged in the workbook movie_metadata.xlsx.

Figure 8 shows the predictions accuracy obtained using the file movie_metadata_test.arff as supplied Test set in Weka.

	Accuracy			
Classifier	Correct #TRUE	%	Wrong #FALSE	%
Random Forest	745	70.55%	311	29.45%
ClassificationViaRegression	744	70.45%	312	29.55%
LMT	732	69.32%	324	30.68%
Bagging	725	68.66%	331	31.34%
J48	715	67.71%	341	32.29%
JRip	712	67.42%	344	32.58%
PART	702	66.48%	354	33.52%
LibSVM (SVM)	700	66.29%	356	33.71%
Bayes Net	697	66.00%	359	34.00%
LWL	682	64.58%	374	35.42%
SMO	682	64.58%	374	35.42%
MultilayerPerceptron	666	63.07%	390	36.93%
IBk (KNN)	564	53.41%	492	46.59%
KStar	562	53.22%	494	46.78%
Naïve Bayes	314	29.73%	742	70.27%

Figure 8 - Classifiers' prediction accuracy

5 - Conclusion

As shown in this work, amongst the 15 classifiers tested on the given dataset, the best performing ones where KStar, LibSVN and Random Forest when tested using the Training Set alone.

JRip presented the best accuracy when doing 5-fold cross-validation.

Having Naive Bayes as the baseline (the worst classifier by the way) and using 5 fold validation, JRip outperformed it by tripling the amount of correctly classified instances. Naive Bayes did 895 and JRip 2794 according to Table 2.

One interesting finding is that Random Forest classifier placed 1st having the best prediction accuracy while JRip placed 6th. Random Forest placed 2nd with 5-fold cross-validation.
Naive Bayes confirmed its bad performance being again the worst classifier according to Figure 8.

Classifiers accuracy vary according to the dataset at hand and that's why trying different classifiers on the same set of data is important.

The numbers are clear: the more the % of correctly classified instances the better is the probability of the classifier being a great fit for the data at hand.

Fine tuning the classifiers is a must-do, that is, picking different sets of attributes, normalizing the data, changing the classifier's default parameters, etc can for sure have an influence in the result.

6 - References

IMDB 5000 Movie Dataset
https://www.kaggle.com/deepmatrix/imdb-5000-movie-dataset
Bagging
http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html
Bayes Net
http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/BayesNet.html
Classification via Regression
http://weka.sourceforge.net/doc.dev/weka/classifiers/meta/ClassificationViaRegression.html
IBk (KNN)
http://www.cs.tufts.edu/~ablumer/weka/doc/weka.classifiers.IBk.html
J48
http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html
JRip
http://weka.sourceforge.net/doc.stable/weka/classifiers/rules/JRip.html
KStar
http://weka.sourceforge.net/doc.dev/weka/classifiers/lazy/KStar.html
LibSVM (SVM)
https://weka.wikispaces.com/LibSVM
LMT
http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/LMT.html
LWL
http://weka.sourceforge.net/doc.dev/weka/classifiers/lazy/LWL.html
Multilayer Perceptron
http://scikit-learn.org/stable/modules/neural_networks_supervised.html
Naive Bayes
http://scikit-learn.org/stable/modules/naive_bayes.html
PART
http://weka.sourceforge.net/doc.dev/weka/classifiers/rules/PART.html
Random Forest
http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
SMO
http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/SMO.html
Model evaluation
http://scikit-learn.org/stable/modules/model_evaluation.html
Cross-validation: evaluating estimator performance:
http://scikit-learn.org/stable/modules/cross_validation.html#cross-validation
Making predictions in Weka
http://weka.wikispaces.com/Making+predictions
Confusion Matrix
https://en.wikipedia.org/wiki/Confusion_matrix