Amazon Product Evaluation

Dataset used: Toys and Games

Due Date: April 28

Joris Chomarat
Halil Arici
Gyaan Antia

Feature preprocessing

	asin	###
1	weighted avg. compound text	$\in \mathbb{Q}$
2	weighted avg. compound summary	$\in \mathbb{Q}$
3	std. dev. text	$\in \mathbb{Q}$
4	std. dev. summary	$\in \mathbb{Q}$
5	verification percentage	$[0 \dots 1] \in \mathbb{Q}$
6	amount of reviews	$\in \mathbb{N}^{+}$
7	avg. star rating	$[1 \dots 5] \in \mathbb{N}$
	awesomeness	$[0,1]$

1) 2) 3) 4) on next slide

5) Verification percentage

Self-explanatory. Percentage of verified reviews out of all reviews of one product.

6) Amount of reviews

Self-explanatory. Amount of reviews a product has.

7) Average star rating

Summary data in dataset contains star rating left by reviewer, if no summary was left. The average of all ratings from these reviews was taken. If no review of a product has a star rating, the lowest possible Amazon star rating of ”1” is assigned.

Feature preprocessing (continued)

1) *Weighted average compound text*

Compound score of NLP, weighted according to whether review is verified, has image, number of votes and age. Then the average is taken over all weighted review scores of a product.

$$comp_{w,avg} = \frac{1}{n}\sum_{i=1}^{n} comp_i * w_{ver,i} * w_{img,i} * w_{vote,i} * w_{age,i}$$

2) *Weighted average compound summary*

Same procedure as with 1), but star ratings excluded. If only star ratings are available for a product, “0” is assigned.

3) *Standard deviation compound text*

Standard deviation of all unweighted compound scores of all reviews of a product.

4) *Standard deviation compound summary*

Same procedure as with 3), but star ratings excluded. If only star ratings are available for a product, “0” is assigned.

Compound score

The compound score is given by using the **Vader Sentiment Analyzer** for NLP. It is calculated by the summation of the valence scores of each word in the lexicon. Then it is normalized: -1 is the most negative and +1 is the most positive score. It can be called a normalized, weighted composite score.

Weights

Verification: 1 if not verified, 1.5 if verified

Image: 1 if no image, 1.3 if one image or more

Votes: $w_{vote} = 1 + \frac{\ln(curr+1)}{\ln(\max + 1.1)}$, *curr* being the votes on the current review, *max* being the highest number of votes on any review of this product

Age: $w_{age} = \frac{0.01}{s\ per\ year}(curr - first) + 1$, *curr* being the age (in seconds) of the current review, *first* being the age of the first review on this product

Algorithms

- Model performance was evaluated using 10-fold cross validation for each model
- Classifier hyper-parameters were optimized using a grid search algorithm maximizing the F1 score
  - Hyper-parameters not mentioned on next slides were kept at their default value
  - Where possible, the amount of CPU cores was increased to the maximum (for logistic regression and random forest classifier)
  - Parameters to optimize over were chosen based on our research which parameters should have the largest impact on performance

- Classifiers evaluated:
  - Decision Tree (*model_dt*)
  - Naïve Bayes (*model_nb*)
  - Logistic Regression (*model_lr*)
  - Random Forest Classifier (*model_rf*)

- Performance scores of algorithms may vary minimally (4^{th} digit after decimal point) after each run

Results

Decision Tree

- Parameters used in grid search & optimum

Criterion	gini, entropy, log_loss	entropy
Splitter	best, random	best
Class weight	None, balanced	None

- Performance

Precision	0.6606
Recall	0.6591
F1	0.6598

Naïve Bayes

- only Gaussian NB was evaluated, because
  1) Multinomial NB is for features > 0 only
  2) Bernoulli is for features $\in [0, 1]$ only

- Performance

Precision	0.6795
Recall	0.7575
F1	0.7162

Results (continued)

Logistic Regression

- Parameters used in grid search & optimum

Tolerance	$[0.01, \dots, 10^{-6}]$	10^{-1}
C	$[0.01, 0.1, \dots, 10^{3}]$	10
Class weight	None, balanced	None

  - Max. iterations were increased to 1000 so that every classifier converged before reaching the limit

- Performance

Precision	0.6864
Recall	0.7966
F1	0.7374

Random Forest Classifier

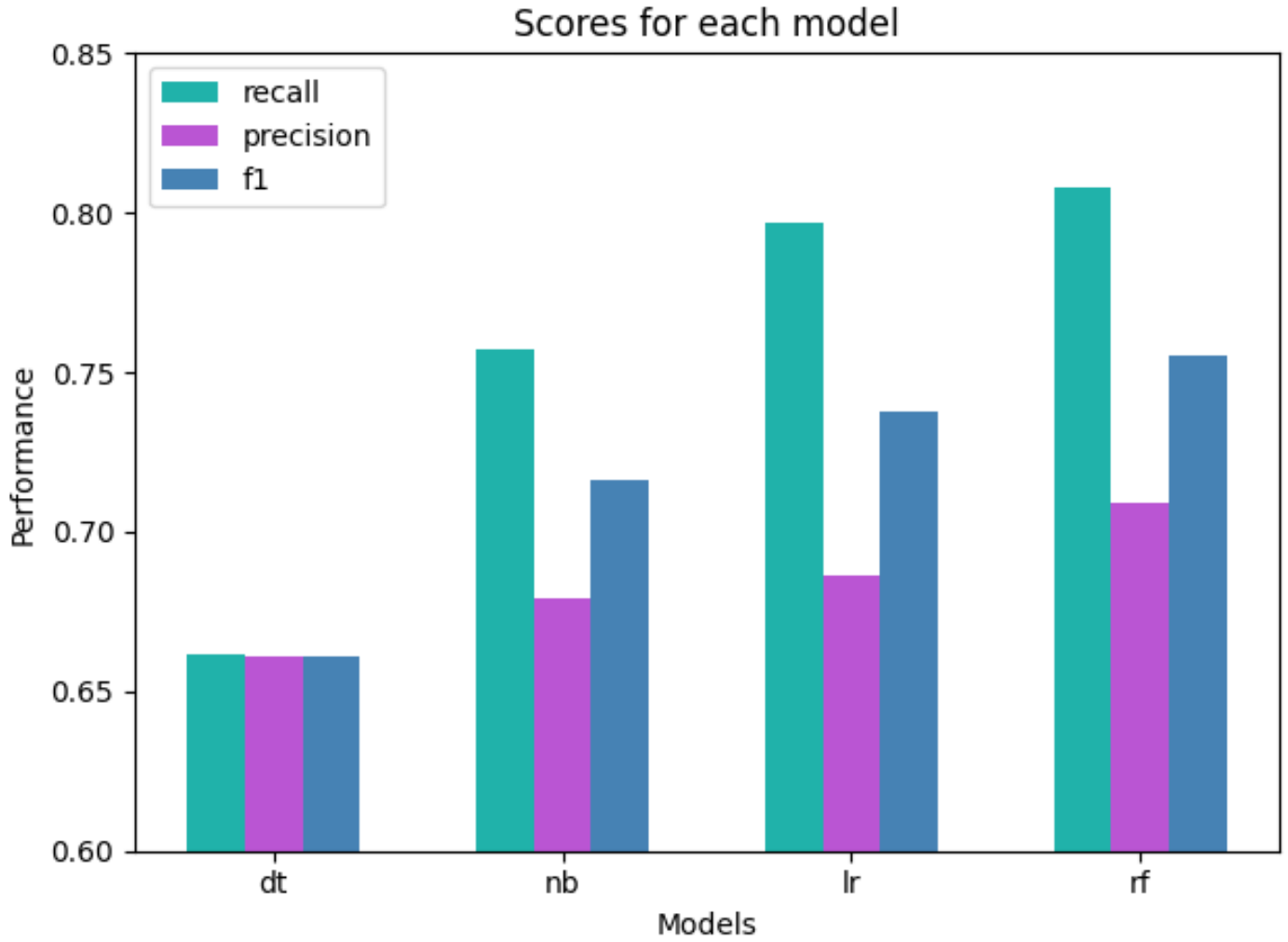
- Parameters used in grid search & optimum

N estimators	$[100, 200, 300, 500]$	100
Criterion	gini, entropy, log_loss	log_loss
Class weight	None, balanced	None
Max Depth	$[1, 2, 4, 8, 16]$	8

- Performance

Precision	0.7093
Recall	0.8079
F1	0.7554

Results (continued)



- based on performance of the optimized classifiers tried out, predictions on *test1* were done with *model_rf*

Next steps

- Classification
  - Implement late fusion to combine multiple classifiers
  - Try out further classifiers, e.g., SVM or KNN
  - Randomize hyper-parameter grid search to find even better performances with more parameters: execute a random search first, a fine grid search in the area of best performance can be performed afterwards
  - Goal: achieve F1 scores > 0.8
- Features
  - Optimize weights used for the weighted average compound scores (iteratively by hand during remaining group part, or as neural network in personal project part)
  - Look into using tentative feature presented in feature deliverable: Product age (time since first review)
  - Look into using other features: Kurtosis and new ideas