Machine Learning for Sentiment Analysis

Avery Peiffer and Daniel Stumpp

What is sentiment analysis?

- Humans are adept at determining the sentiment of text using context
  - Choice of vocabulary, sentence structure, punctuation, etc.
- Want machines to be able to automatically classify text based on sentiment - *sentiment analysis*
  - Analyzing customer feedback, assess well-being, etc.
- Much harder for machines to infer the context necessary for classifying sentiment

Related work – feature extraction

- First have to turn plaintext into features that are workable for an ML algorithm
  - Bag of Words, TF-IDF, Word2Vec, graph methods
- This allows an algorithm to assign a standardized score to tokens, which are used for training/prediction
- Can then use basic classification techniques to classify sentiment, such as naive Bayes, SVM, and maximum entropy classifiers

Related work (cont.) - basic classification techniques

- Neethu and Rajasree: extracted positive and negative keywords from text and used naive Bayes, SVM, and maximum entropy classifiers
    - Each achieved accuracy of ~90%

- Rathi et al.: used SVM, decision tree, and AdaBoost classifiers on Stanford Sentiment140 dataset
    - 1.6 million tweets
    - Achieved experimental accuracies of 82%, 67%, and 84%, respectively

Related work (cont.) - CNN & LSTM Architectures

- Sosa introduced combined LSTM-CNN architecture
  - Evaluated on dataset of over 1.5 million tweets labeled as positive or negative (binary classification)
  - Outperformed LSTM and CNN architecture individually, achieving accuracy of 75.2%

- Chen and Wang added encoder/decoder framework to LSTM-CNN architecture
  - Structure could allow CNN to learn features more intrinsically and effectively
  - On same datasets, achieved accuracy of 78.6%

Related work (cont.) - Transformers & BERT

- CNN/LSTM architectures have a high training cost

- Vaswani et al. introduced transformer architecture in 2017
  - Main idea: use attention mechanisms to focus on critical data and ignore the rest

- Devlin et al. introduced BERT in 2018, which uses transformers to encode additional context about a sentence
  - Bidirectional Encoder Representations from Transformers
  - Reads a sentence in both directions to gather context

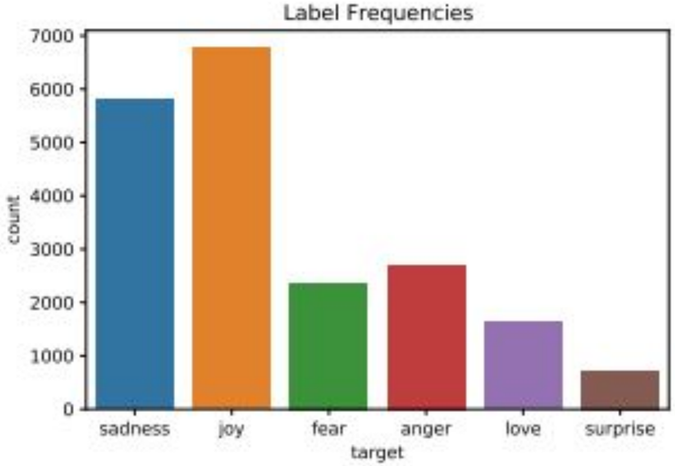
The Plan

- Analyze Carer dataset and apply various classification architectures to it
  - 20,000 tweets classified as one of six emotions: sadness, joy, fear, love, anger, surprise
  - 80/10/10 split
- Use decision tree, random forest, gradient boosting methods
- Compare to state-of-the-art models in sentiment classification
  - Implementation of BERT called RoBERTa

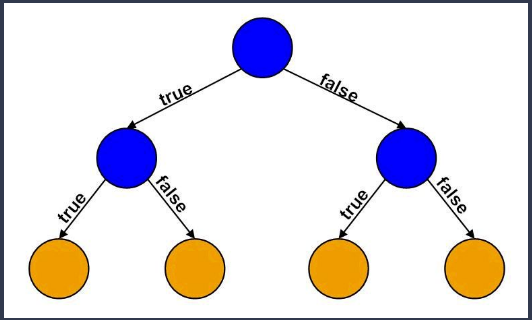
Notes on dataset

- 16,000 training samples - risk of overfitting on complex models

- Dataset is imbalanced, meaning that traditional accuracy cannot be the sole metric used to evaluate performance



Decision Tree Based Classifiers



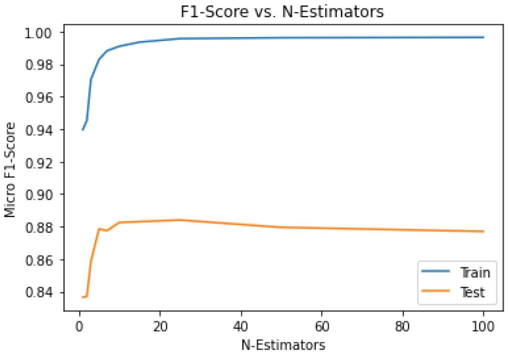
- Four decision tree based methods were considered for sentiment analysis
  - Bagging
  - Random Forest
  - AdaBoost
  - Gradient Boosting

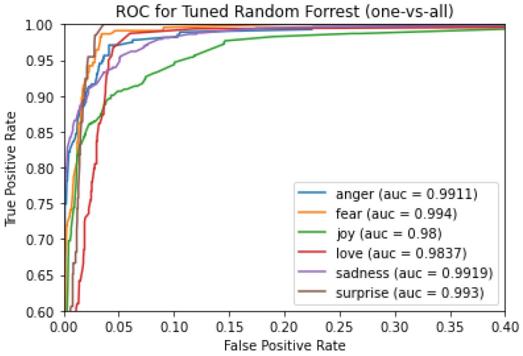
- Preliminary results collected and top two performing methods selected for further analysis
  - Random forest and gradient boosting selected

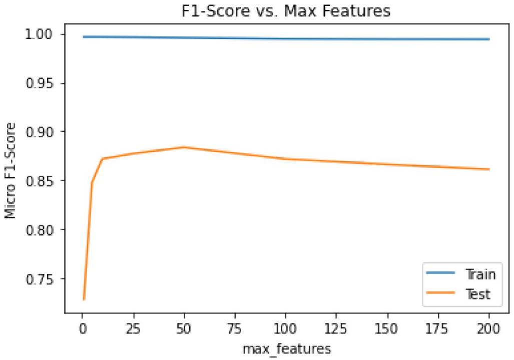
- Analysis performed using both bag-of-words and TF-IDF features
  - Equivalent results, bag-of-words selected

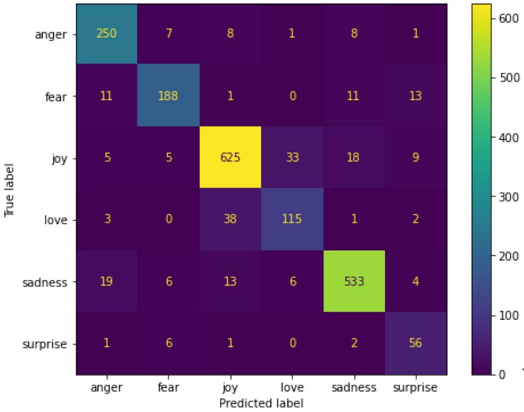
Random Forest

- Tuning requires relatively few hyperparameters
- Considered number of estimators and maximum number of features considered at each split
- Model generally insensitive to hyperparameter tuning
- Best model has 25 estimators with max features set to 50
    - Achieves F1-Score of 0.88
- ROC plot and confusion matrix show balanced performance across all classes









Gradient Boosting – Tuning

- Utilized XGBoost for training models
- Selected a subset of the many possible hyperparameters to tune
    - n_estimators, max_depth, min_child_weight, gamma, colsample_bytree, subsample, reg_lambda, and learning_rate
- Utilized early-stopping cross-validation to tune the number of trees needed
    - Multiclass log loss used for scoring metric
- Applied extensive grid-search cross-validation to find optimal model parameters
- Validation set used to avoid biasing model

parameter	description	value
objective	learning task objective	'multi:softprob'
n_class	number of classes	6
tree_method	tree construction algorithm	'hist'
eval_metric	metric for validation	'mlogloss'
nthread	parallel threads (-1 is all available)	-1

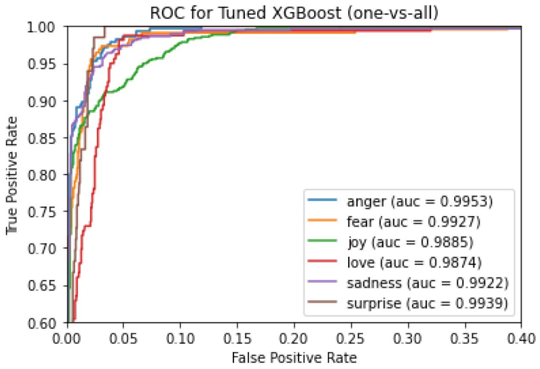
Algorithm 1 Pseudocode for XGBoost tuning.

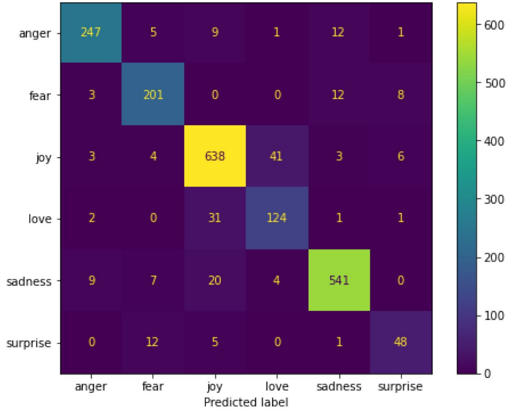
1: Load Data
2: Set initial model parameters
3: Use early-stopping cross-validation to determine initial $n\_estimators$
4: Coarse tuning of max_depth and min_child_weight using grid search cross-validation
5: Fine tuning of max_depth and min_child_weight based on coarse tuning results
6: Coarse tuning of $gamma$ using cross-validation
7: Fine tuning of $gamma$ based on coarse result
8: Use early-stopping cross-validation to re-calculate the value of $n\_estimators$
9: Tune $colsample\_bytree$ and $subsample$ using grid search cross-validation
10: Tune reg_lambda using cross-validation
11: Halve learning rate and update $n\_estimators$ using early-stopping cross-validation
12: Evaluate tuned model

Gradient Boosting – Results

- Test F1-Score improves from 0.88 to 0.89 when XGBoost model is tuned
- Test Log loss decreases from 0.272 to 0.245
    - 9.9% improvement indicates increased model confidence in classifications
- Tuned XGBoost model outperforms random forest classifier

hyperparameter	Un-tuned Value	Tuned Value
learning_rate	0.3	0.15
n_estimators	519	1791
min_child_weight	2	0
gamma	0	0.2
subsample	0.9	1
colsample_bytree	0.9	0.7





RoBERTa

- Extension of BERT that optimizes its pretraining approach
- Downloaded and fixed Marcin Zablocki's tutorial code for RoBERTa implementation
- Ran model with different hyperparameters to assess performance in detail
- Achieved consistent accuracy of 92-93% depending on hyperparameter choices

Results Comparisons

	Random Forest	XGBoost	RoBERTa
Accuracy	0.884	0.900	0.935
Macro F1-Score	0.842	0.858	0.897
Top-2 Accuracy	0.981	0.989	0.987

- RoBERTa achieves the best F1-Score, followed by gradient boosting and then random forest
- Top-2 accuracies for all models above 98%
  - Indicates class overlap in dataset accounts for majority of classification errors for all models
  - Decision tree based methods match top-2 accuracy of RoBERTa
- Decision tree based methods achieve comparable accuracy to RoBERTa
- Achieved comparable performance to related works

Conclusions

- Decision tree based classifiers can achieve comparable performance to transformer based models like RoBERTa
- Parameter tuning using cross-validation for gradient boosted decision trees is effective for increasing model performance
- Class overlap text creates challenges for quantifying performance of sentiment analysis
  - Metrics such as top-2 accuracy can help remove this issue from evaluation
- Consideration of decision tree models is warranted for applications where large BERT-like models are not feasible