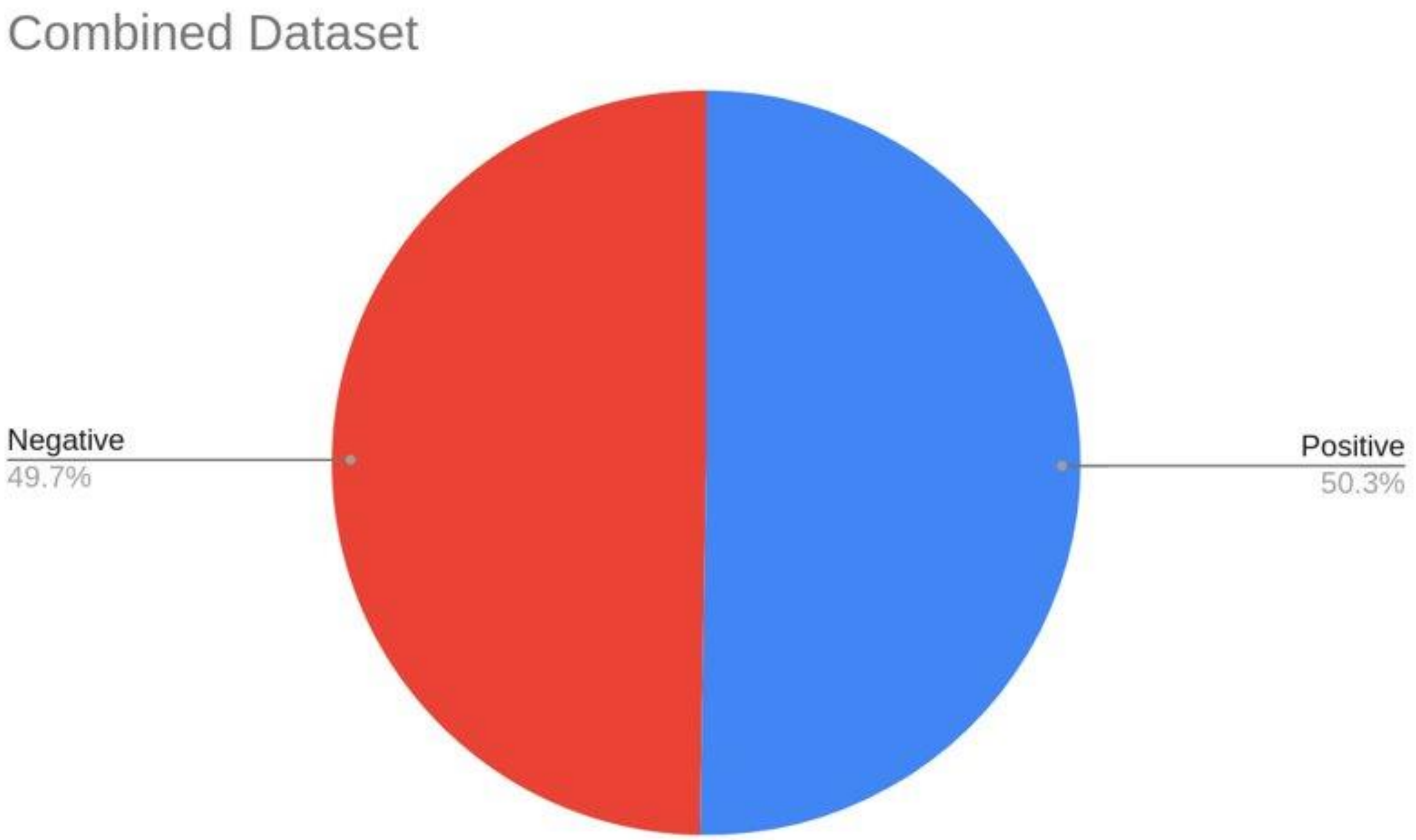
ENSF 612 Presentation 3

By Alex Leakos, Jared Kraus, and Graydon Hall

Data Collection

Combined Dataset



Did not limit ourselves to 50 / 50 split of positive negative

Original benchmark used "highly polarized reviews" vs new dataset using 1000 scraped reviews

Data Labeling

Labeling of Data

- 1 (positive review)
- -1 (negative review)

Review	JK Rating	GH Rating	AL Rating
Great job Mike Flanagan.I don't know what's going on with movies lately, I have high expectations and films suck, or in this case, my expectations were fairly low, and I was blown away.What a truly worthy follow up to The Shining this was. Almost forty years on, it captures the tone, spirit and vibe of that great film.You'd think at over two and a half hours it would be overlong, it isn't,that running time allows the complex story to be told, and for the characters to be fully developed.Young Kyleigh Curran is remarkably good, and in great company with Ewan McGregor and Rebecca Ferguson, very well acted.A great start, with that amazing music, and those glorious panoramic shots. It takes time before you arrive at that destination, the one we all waited for. The recreation is remarkable.All those involved, take a bow, this was outstanding, 9/10.	1	1	1
There's going to be a lot of different opinions about The Lighthouse, divided by the arty farty people that will like it and the people that just want easy entertainment that will not become a big fan of this movie. I'm in between, didn't think it was terrible and that's due to the good acting and some good cinematography, but certainly not impressed by the story. I think that movie should have been shot in colour, well I think that of every movie, don't see the point of black and white in our modern age. If it was not for the good acting of Robert Pattinson and Willem Dafoe I wouldn't waste your time on this one. There are better movies about madness than The Lighthouse.	-1	-1	-1

Data Preprocessing

Preprocessing steps we followed

Steps we followed continued:

1. BeautifulSoup to remove html and urls
2. Make all words lowercase
3. Strip leading/trailing whitespace
4. Remove any punctuation using Regex
5. Stem words using SnowballStemmer
6. Tokenize words
7. Remove stop words
8. Remove words < 3 characters
9. Remove non-alphabetical words (ex. Numbers).

Data Features

Bag of Words

- Initial step: create bag of words vectors using CountVectorizer
- Decided on CountVectorizer over HashingTF due to professor's recommendation
- Input = pre-processed list of words (“filtered_body” column)
- Output = term frequency vector

TF-IDF: Term Frequency-Inverse Document Frequency

- Alternative to simple bag of words model
- Instead of having simple term frequency, you have TF-IDF
- Value is reflective of words importance in the document
- Formula:

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

https://www.google.com/url?sa=i&url=https%3A%2F%2Fted-mei.medium.com%2Fdemystify-tf-idf-in-indexing-and-ranking-5c3ae88c3fa0&psig=AOvVaw3ii1Y-Od78Jx4CAcVCgKTY&ust=1636995683441000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCPj_4qOqmPQCFQAAAAAdAAAAABAS

Word2Vec Model

- Pyspark "Word2Vec" library

- We used the following parameters to create a word2vec model:
  - VectorSize = 100
  - WindowSize = 5

- Averaged word vectors to produce a review vector

Models

Model Analysis Workflow

1. Ran a base model using each feature column

2. Hyper-tuned model parameters
   - ParamGridBuilder with CrossValidator

3. Retrained best model

4. Did a model comparison of old, new and combined datasets

Random Forest Classifier

Feature Selection

- Fit a base RFC Model on each feature to determine which one performed the best
  - numTrees=10
  - maxDepth=5

Feature	**Accuracy**
Count Vectorizer	60%
TF-IDF	60%
Word2Vec	76%

Hypertuning Parameters

- Used ParamGridBuilder with CrossValidator to determine best parameters for model
- numTrees and maxDepth had the largest impact on RFC accuracy:

  - NumTrees
    - Number of trees to train
    - Parameters given: [200, 300, 400, 500]
    - CrossValidator: 400

  - MaxDepth
    - Maximum depth of the tree
    - Parameters given: [5, 10, 20, 30]
    - CrossValidator: 10

Retrain Best Model

Base Model
- RandomForestClassifier(numTrees=10, maxDepth=5)
- Accuracy = 76%

Hypertuned Model
- RandomForestClassifier(numTrees=150, maxDepth=10)
- Accuracy = 84%

Reports Model (word2vec + RFC)
- Accuracy = 84%

Dataset Comparison

Actual \ Predicted	neg	pos
neg	**TN**	**FP**
pos	**FN**	**TP**

Report Data

Actual \ Predicted	neg	pos
neg	**6080**	**1364**
pos	**1172**	**6348**

New Data

Actual \ Predicted	neg	pos
neg	**143**	**23**
pos	**67**	**23**

Combined Data

Actual \ Predicted	neg	pos
neg	**6633**	**1017**
pos	**1474**	**6143**

	Report Data	New Data	Combined
Accuracy	83.0%	61.0%	84.0%
Precision	83.0%	62.0%	84.0%
Recall	83.0%	65.0%	84.0%
F1-score	83.0%	61.0%	84.0%

Decision Tree

Feature Selection

- Fit a base DecisionTreeClassifer Model on each feature to determine which one performed the best
  - numBins = 32
  - impurity='gini'
  - maxDepth=5

Feature	Accuracy
Count Vectorizer	68%
TF-IDF	68%
Word2Vec	71%

Hypertuning Parameters

- Used ParamGridBuilder with CrossValidator to determine best parameters for model
- MaxDepth had the greatest impact on outcome
- Small average improvements using entropy over gini

- Impurity
  - Measures the quality of the data split
  - Parameters given: entropy, gini
  - CrossValidator best selected parameter: entropy

- MaxDepth
  - Number of trees to train
  - Parameters given: [5, 10, 20, 30]
  - CrossValidator best selected parameter : 20

$$GiniIndex = 1 - \sum_j p_j^2$$

Gini Formula

$$Entropy = -\sum_j p_j \cdot log_2 \cdot p_j$$

Entropy Formula

https://quantdare.com/decision-trees-gini-vs-entropy/

Dataset Comparison

		Predicted neg	Predicted pos
Actual	neg	TN	FP
	pos	FN	TP

Report Data

		Predicted neg	Predicted pos
Actual	neg	6044	1482
	pos	1248	6208

New Data

		Predicted neg	Predicted pos
Actual	neg	160	16
	pos	69	23

Combined Data

		Predicted neg	Predicted pos
Actual	neg	6429	1175
	pos	1527	6180

	Report Data	New Data	Combined
Accuracy	73.9%	54.4%	74.4%
Precision	73.9%	55.9%	74.8%
Recall	73.9%	53.5%	74.4%
F1-score	73.9%	54.4%	74.4%

Retrain Best Model

Base Model

- DecisionTreeClassifier(maxBins=32, maxDepth=5, impurity='gini')
- Accuracy = 71%

Hypertuned Model

- DecisionTreeClassifier(maxBins=32, maxDepth=20, impurity='entropy')
- Accuracy = 74%

Reports Model (word2vec + RFC)

- Accuracy = 84%

Naïve Bayes

Feature Selection

- Fit a base Naïve Bayes Model on each feature to determine which one performed the best, using default Naïve Bayes parameters
  - smoothing=1.0
  - modelType='multinomial'
- Note: Word2Vec is non-compatible with this model in MlLib library

Feature	Accuracy
Count Vectorizer	78%
TF-IDF	68%
Word2Vec	n/a

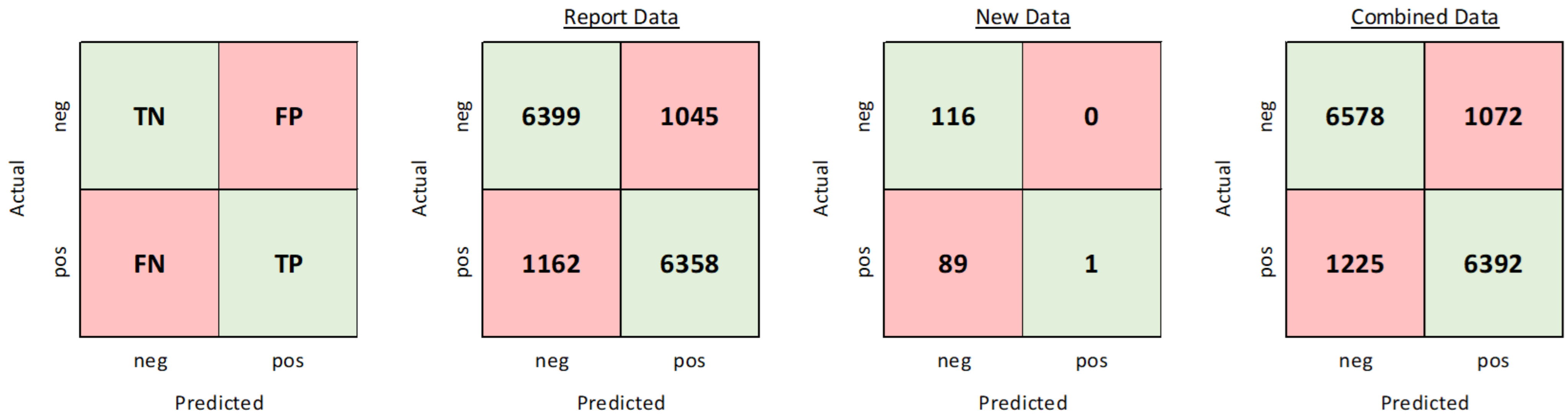
Hypertuning Parameters

- Used ParamGridBuilder with CrossValidator to determine best parameters for model
- smoothing and modelType parameters were explored:

  - Smoothing
    - *Laplace smoothing*: handles zero probability problem (for when words shows up in testing data that was not present in training data)
    - *Parameters given*: [5, 7, 9, 12, 15, 17, 19]
    - *Cross Validator Best Selected Parameter*: 12

  - Model Type
    - *Type of Naïve bayes model used:*
      - *Multinomial*: follows multinomial distribution
      - *Gaussian*: follows Gaussian normal distribution and supports continuous data
      - *Complement*: adaptation of the Multinomial NB, uses statistics from the complement of each class to compute the model's coefficients
    - *Parameters given*: ['multinomial', 'gaussian', 'complement']
    - *Cross Validator Best Selected Parameter*: 'complement'

Dataset Comparison



	Report Data	New Data	Combined
Accuracy	85.3%	51.9%	85.0%
Precision	85.3%	74.0%	85.0%
Recall	85.3%	65.2%	85.0%
F1-score	85.3%	51.9%	85.9%

Gradient Boost

Gradient Boost

Summary

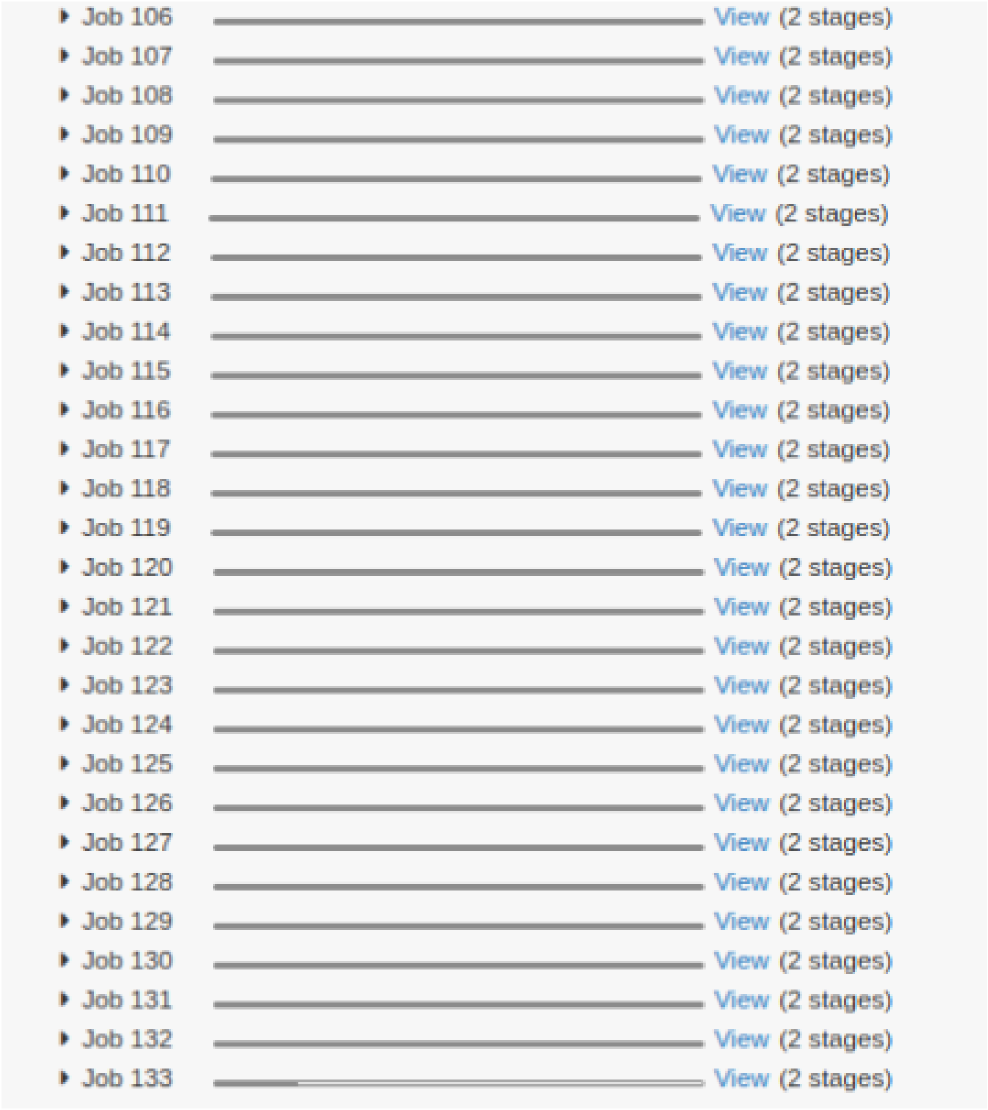
- Attempted a gradient boost using GBT Classifier
- We were not able to successfully parallelize the model to train within a reasonable time

Results:

- Successfully completed on a grid search of 1000 in 2 hours 35 minutes
- Accuracy results of 74.3% on the hypertuned parameters
- Unsuccessfully trained on full result set

Further Exploration:

- Explore parallelism parameter in ml pyspark
- Use another Gradient Boost algorithm (XGBoost)



Discussion of Results

Misclassification

Conflicting sentiment

Prediction:

Record Label: 0
Models Prediction: 1

Reason:

Words pertaining to sentiment contradicted each other

Example:

- "good" is listed 5 times
- "bad" is listed 3 times

[giant,monster,fan,see,yeti,absolut,must,especi,hear,much,thank,good,bootleg, market,abl,find,copi,pretti,easili,happili,surpris,upon,watch,flick,actual,dare,say,de centdec,actual,name,cheesi,giantmonst,flick,kick,pretti,quick,yeti,found,pretti, much,immedi,get,introduc,various,charact,consist,sleazi,one,good,one,girl,pretti, much,one,downright,strike,beauti,girl,cheesi,scifi,film,faryeti,look,like,longhair,guy ,straight,origin,woodstock,concert,realli,hes,bad,dude,especi,introduc,world,kind, funki,cagelik,thing,godzilla,despit,rude,awaken,doesnt,even,rampag,actual,rare, destroy,anyth,whole,pictur,kinda,look,puzzl,tri,figur,thing,yeti,seem,understand, english,pretti,nice,copi,dub,english,know,good,guy,bad,guy,arehowev,want,see, giant,yeti,thing,hes,pretti,much,whole,movi,typic,lowbudget,fashion,seem,chang, size,lot,depend,scene,there,even,bunch,fake,leg,shot,stand,therey,special,effect, arent,greatest,definit,good,one,scene,yeti,smash,warehous,done,well,anoth,use, window,build,ladder,step,climb,top,shatter,window,foot,often,shock,occup,insid, one,sequenc,realli,look,much,much,better,bad,movieyeti,never,stoop,low,say,ape, actual,time,even,come,close,genuin,silli,beauti,girl,caus,yeti,nippl,becom,erect,lift, eyebrow,yeah,babi,manner,even,isnt,bad,kinda,even,get,laugh,viewerth,movi, pretti,long,kind,thing,surpris,enough,doesnt,get,bore,stori,actual,good,watch, utter,gorgeous,actress,screen,make,male,viewer,happyyeti,may,upper,echelon, giant,monster,flick,definit,better,king,kong,ripoff,like,ape,queen,kong,far]

Weak sentiment

Prediction:

Record Label: 0
Models Prediction: 1

Reason:

Lack of words to give hints at sentiment

Example:

- Lack of words with strong sentiment
- Hard to say if user liked or disliked
- Review focused on describing movie

[consid,teen,film,like,breakfast,club,pretti,pink,lioniz,surpris,one,ignoredther,sex,sex,thought,includ,idea,may,matter,other,think,kid,alway,get,along,parent,neither,parent,kid,seen,alway,right,wrong,parent,seen,monstersit,deal,heroworship,one,girl,danger,thing,could,lead,real,dustier,realiz,wrongth,movi,kind,ahead,time,one,kid,ask,anoth,kid,birth,control,use,say,noth,need,birth,control,repli,wrong,oral,sex]

Sentiment Split

New vs Base Data

Differences

In all cases our new data set performed worse than the original dataset.

Accuracy Score

	Original	New Data	Difference
Random Forest	73.9%	54.4%	19.5%
Naive Bayes	85.3%	51.9%	33.4%
Decision Tree	83.0%	61.0%	22.0%

Reasons for Discrepancies:

- Original data used highly polarized results
- Excluded neutral results

Going Forward

Going Forward

Following this report, we have the following ideas for further analysis

- Hyper parameter tuning on larger portion of the dataset
  - Currently using 1000 rows due to computational limitations (Databricks community edition)
- N-grams
  - Provides improved context.
  - Ex. Words "not good" considered together
- PySpark GBTClassifier (Gradient Boosted trees)
  - Unable to complete due to computational limitations (Databricks community edition)
  - Possibly explore other Gradient Boosted classifiers as well (ex XGBoost)
- Exploring further pre-processing options
  - Leaving in punctuation and stop words
- Testing out multi-classifier
  - Currently only positive and negative labels
  - Broaden to positive, negative, netural