Increasing your Reddit karma

NIRAJ SARAN

FEB 23, 2022

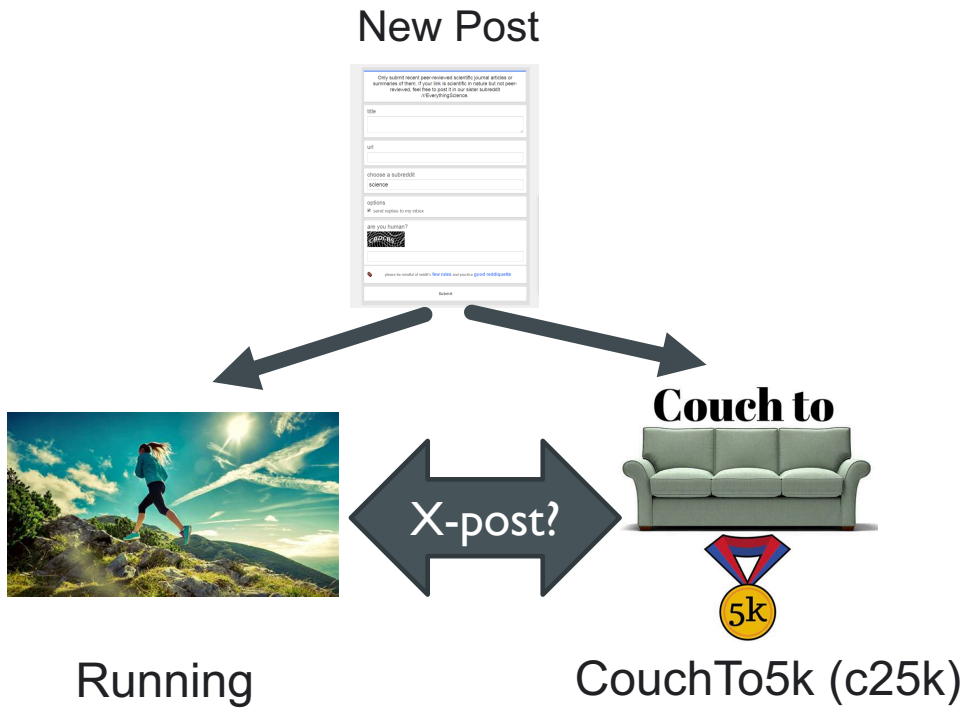
GA DSIR

Problem Statement

- Reddit: 3M subreddits, only 130k active, hundreds new every week
- Help Reddit Moderators improve karma by auto-submitting posts to the correct subreddit for more meaningful discussions and hence upvotes
- Reddiquette: Cross-post if it belongs to either or both?

Breakdown

- Many ‘specialized/subset’ communities like
  - Marketing and Social Media Marketing; Fitness and Running; Running and CouchTo5K (a group for first-time runners to get to run a distance of 5km)
- Can we reduce similarity (increase separation between groups)
- Title, Body or combination of both from reddit post?
- How do we compare models? Which metrics?
- Can we extend this from binary classification to multi-class?



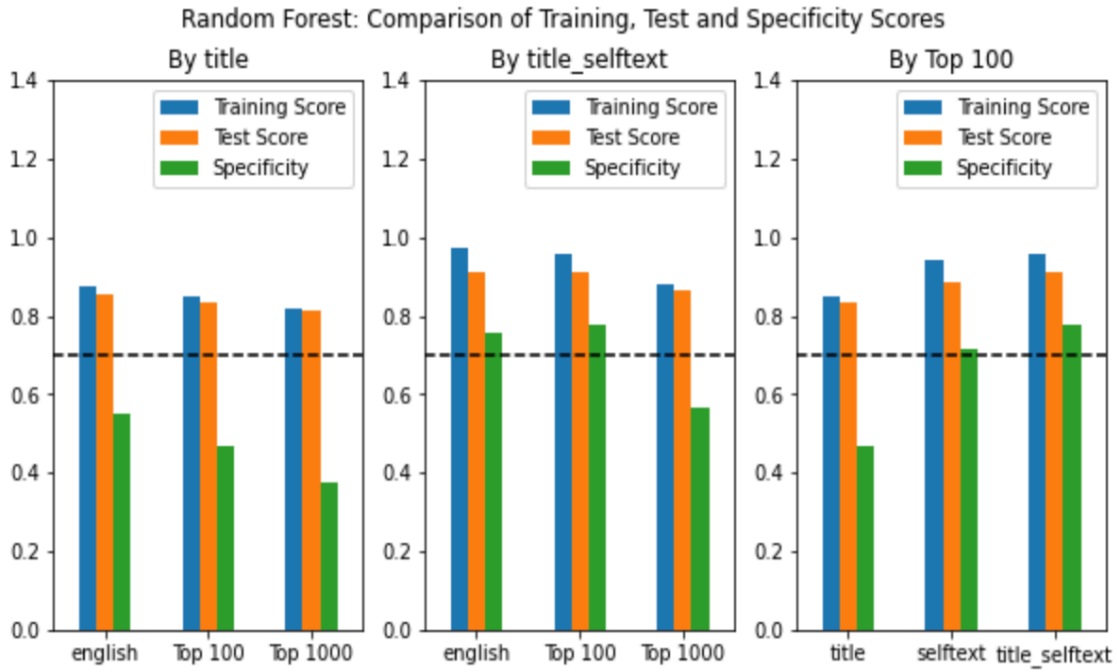
Methodology - Classification

1. NLP and Classification Model training
   - Pushshift API (10 days at a time), EDA (not much data cleanup!)
   - Pipeline of CountVectorizer, TF-IDF Transformer and estimator
   - Figure out optimal hyper-parameters for each classifier
   - Pseudo-Hyperparameters, MetaGridSearch (**really cool**!)
     - Tokenize Title, selftext (body) or concatenation of both
     - Stop words: English, Top 100, Top 1000 (pickled to file)
     - 9 combinations
2. Evaluation metrics written to file for each classifier
3. Composite file (dataframe) of results (4 classifiers * 9 iterations = 36 results) for super easy comparison and visualization
4. Deep dive into 2 models: Random Forest and SVC
5. Determination of most relevant metrics to evaluate
6. And the winner is….

75 words out of the top 100 are common across Running and C25k

Findings - NLP Findings

- 20% improvement in Accuracy by using title_selftext and Top 100 stop words
- Pre-processing of data and EDA is super important. More than spending time tuning models with GridSearch
- More data helps in training: title+selftext improved accuracy by 15% from just title (graph2 vs graph1)
- Stop Words matter (see right most graph)
  - Training/Test Score of 0.96 and 0.91, reduced variance

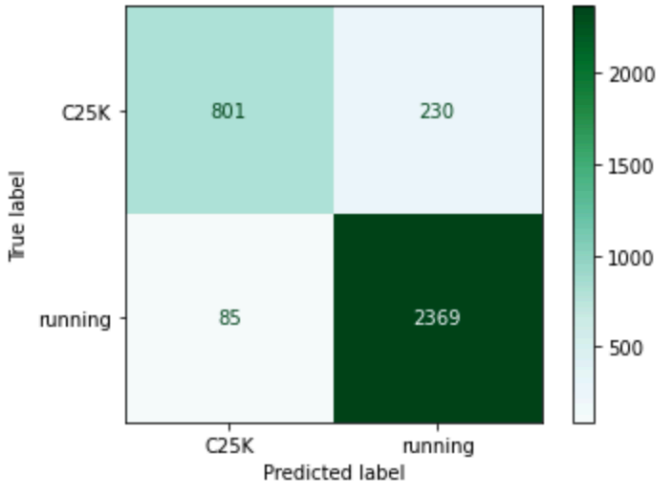


20% improvement from english & title to Top 100 & title_selftex

Baseline: 70% - dotted line

Classification Model Metrics (SVC)

Training Score	Test Score	Variance of Test vs Train Score (%)	Accuracy	Sensitivity	Specificity	Precision	ROC AUC Score
0.9219	0.9128	0.99	0.9128	0.9751	0.76	0.9078	0.9481



- Training Score and Test are above 0.9, variance is very low
- False Positives are very high. The model predicted a post belonged to running whereas it was actually in c25k (not surprising given that c25k is a more specialized form/subset of running)
- **Specificity: ranges from 0.3-0.8. All others>0.9**
- Which metrics are most relevant after Training and Test Score
  - Lower accuracy but very low variance (less than 2%), but higher False Positives (lower Specificity)
- ROC AUC Score is 0.95 - showing a good separation of classes

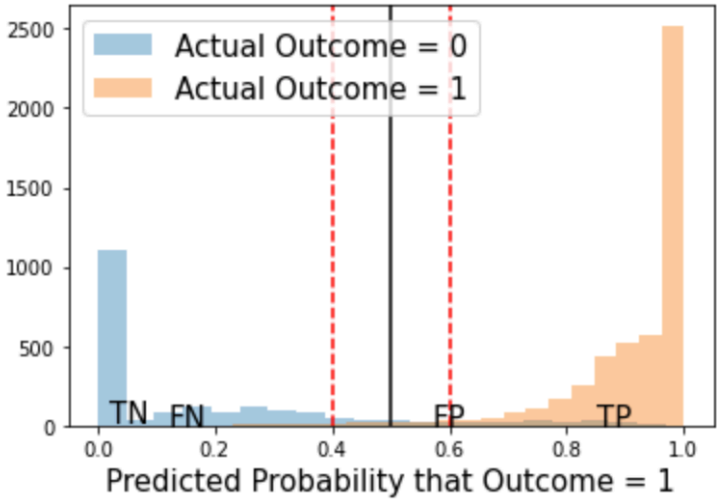
Specificity = True negative rate i.e. True Negatives/all Negatives = TN/TN+FP
As FP increases, Specificity decreases

What if model cannot reliably classify?

Cross-post those between 0.4 and 0.6 predicted probability

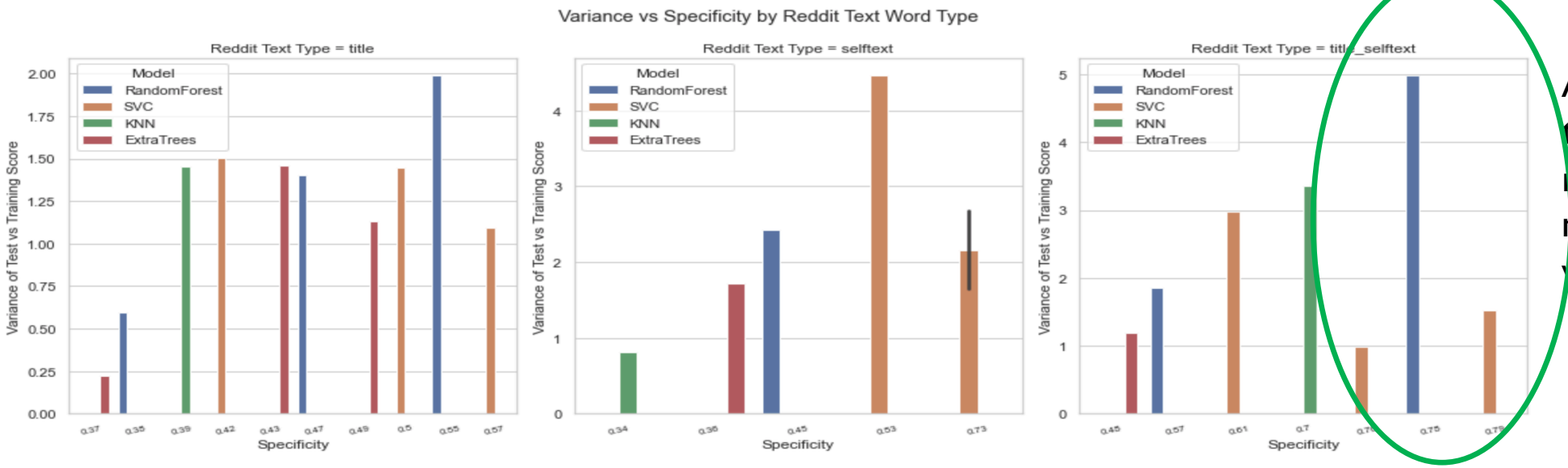
- A post could belong equally to either or both subreddits
- There are many borderline cases where the model is not very confident. Humans also may not be able correctly identify which group is more appropriate.
- Text: I've never been a runner, but yesterday I ran a mile in 10 minutes. How do I improve?
- Model Predicted proba: 0.46; Predicted: 0 (c25k); Actual: 1 (running) - False Negative
- Cross-posting increases Reddit karma by having deeper conversations in more meaningful communities, increasing upvotes
- Added advantage of reducing our False Positives and False Negatives by moving our threshold in each direction. Setting the red dotted bar to 0.4, will eliminate the above FN because it will now be posted to the c25k subreddit also

Many posts are equally at home in both groups



Comparing all 4 Classifiers

Ideal: Highest Training Score (and closest Test Score i.e. low variance, below 2%) and highest Specificity



Any of these three rightmost models will work well

SVC shows low variance of 1% with high Specificity (0.76 and 0.79) (green circled area in rightmost graph)
Random Forest and KNN also do quite well. Extra Trees not so well
In the first graph, the variance is low (all below 2% but the Specificity is very low (too many FP)

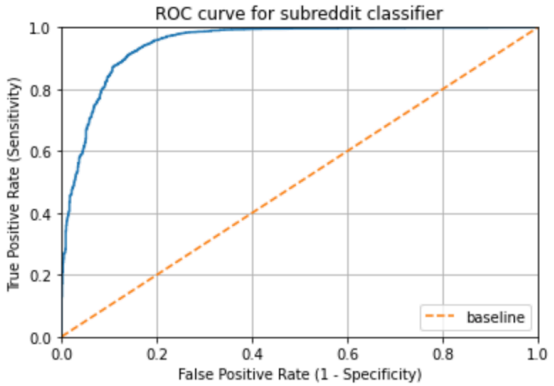
And the winner is… SVC, wait, what? or is it Random Forest?

Rank	Model	Reddit Text Type	Stop Word Type	Baseline Score	Training Score	Test Score	Variance of Test vs Train Score (%)	TN	FP	FN	TP	Accuracy	Sensitivity	Specificity	Precision	ROC AUC Score	F1 Score
1	SVC	title_selftext	Top 100	0.7042	0.9219	0.9128	0.99	788	243	61	2393	0.9128	0.975	0.76	0.9078	0.9481	0.9403
2	SVC	title_selftext	english	0.7042	0.9298	0.9156	1.52	815	216	78	2376	0.9156	0.9682	0.79	0.9167	0.9525	0.9417
3	Random Forest	title_selftext	Top 100	0.7042	0.9574	0.9096	5.00	801	230	85	2369	0.9096	0.9654	0.78	0.9115	0.9443	0.9377

Best Params

cvec__max_df	0.97
cvec__max_features	4000
cvec__min_df	2
cvec__stop_words	'Top 100'
svc__C	0.8889
svc__degree	2
svc__kernel	'sigmoid'
svc__probability	True
tfidf__use_idf	True

- 92% accuracy
- The Test Score is only 1% lower than the Training Score, implying almost no overfit!
- SVC: predict_proba doesn't match with prediction. So Random Forrest may be the winner given that cross-posting is important in our scenario



Conclusion and Learnings

- Pre-processing of data and EDA is super important. More than spending time on tuning model with hyper-parameters with GridSearch
- Accuracy by itself is not critical for all use cases. Pick your metric!
- Random Forest max_depth of the Decision Tree is 70, still test score is fine and the model is not overfit at all, due to pre-processing of data
- SVC: predict_proba doesn't match with prediction
- Evaluation metrics may not be the final decider in choosing the best model!

Significant improvement in each of our 4 classifiers by using title+selftext

Next Steps

- With a high accuracy of 92% and low variance, the model is good to go!
- Still room for improvement:
  - Use python scripts for code re-use
  - Lemmatize, stemming
  - Top 300 or Top 500 - hyper-param tuning for stop words
  - KNN more tuning
  - Extend from binary classification to multi-class
  - Can create more balanced classes: currently baseline is 70%, could maybe try oversampling minority and under-sampling the majority class (SMOTE)
  - Clean up warnings:
    - UserWarning: Your stop_words may be inconsistent with your preprocessing. Tokenizing the stop words generated tokens ['running'] not in stop_words.

References

1. Adding words to scikit-learn's CountVectorizer's stop list - Stack Overflow
2. How to change the number or rows and columns in my seaborn catplot - Stack Overflow
3. Plot ROC curve
4. Predict_proba doesn't match with predictions
5. Image credit: Reddit karma

Question & Answer