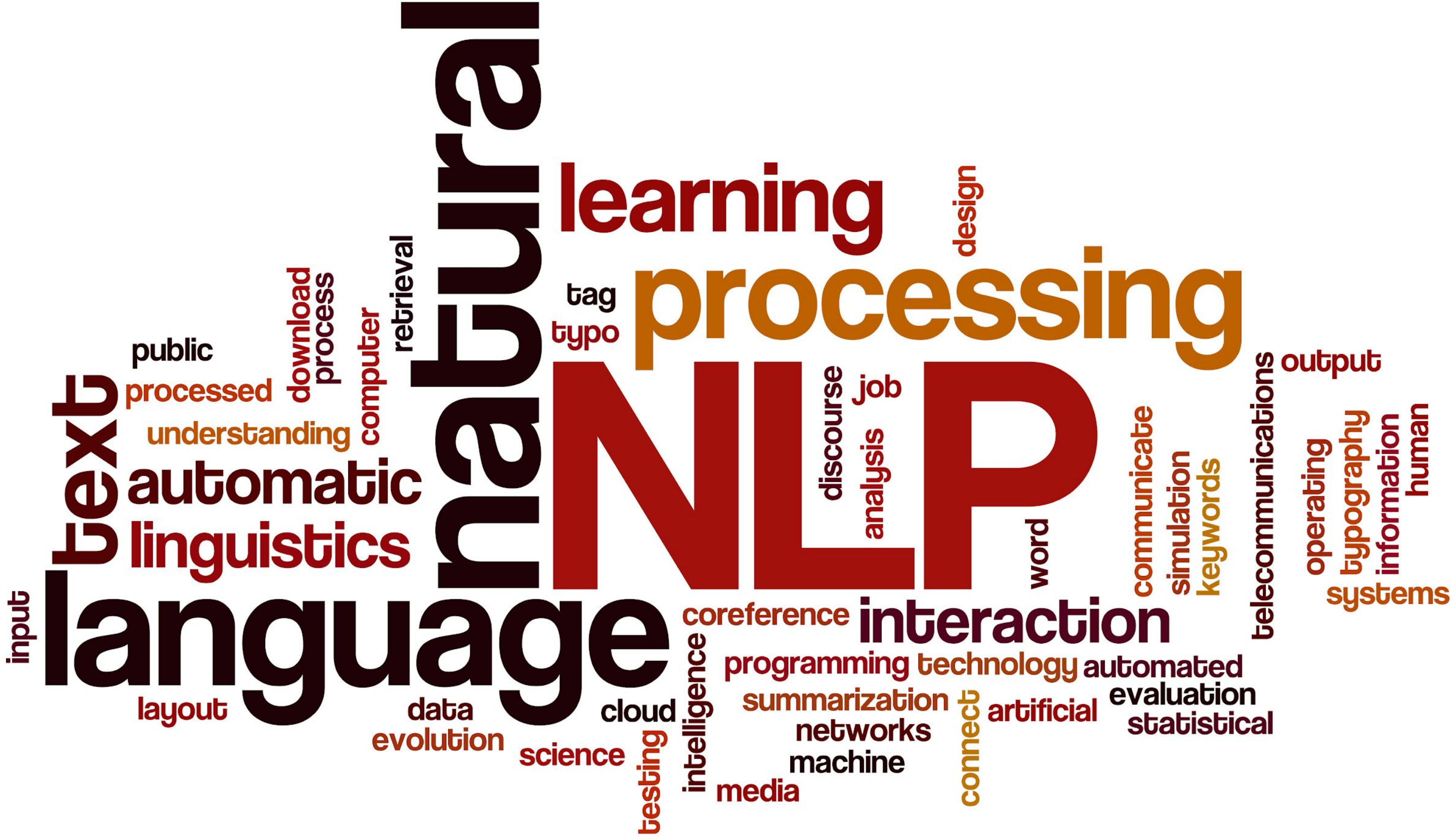


natural
learning
design
processing
retrieval
download
process
computer
tag
typo
public
NLP
text
processed
understanding
automatic
linguistics
discourse
job
analysis
communicate
simulation
keywords
telecommunications
output
operating
typography
information
human
word
systems
input
language
coreference
interaction
intelligence
programming
technology
automated
evaluation
statistical
layout
data
cloud
summarization
artificial
connect
networks
evolution
science
testing
machine
media

Using data between two subreddits on Reddit, would a Multinomial Naive Bayes model perform better than a Support Vector Machines model to predict which post belonged to the correct subreddit?

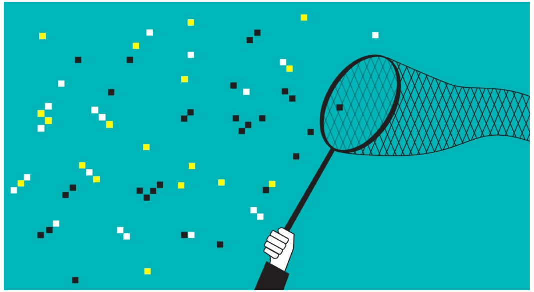
Marvel Studios and the Marvel Cinematic Universe Join

r/marvelstudios

VS

Marvel Comics Join

r/Marvel



PUSH SHIFT

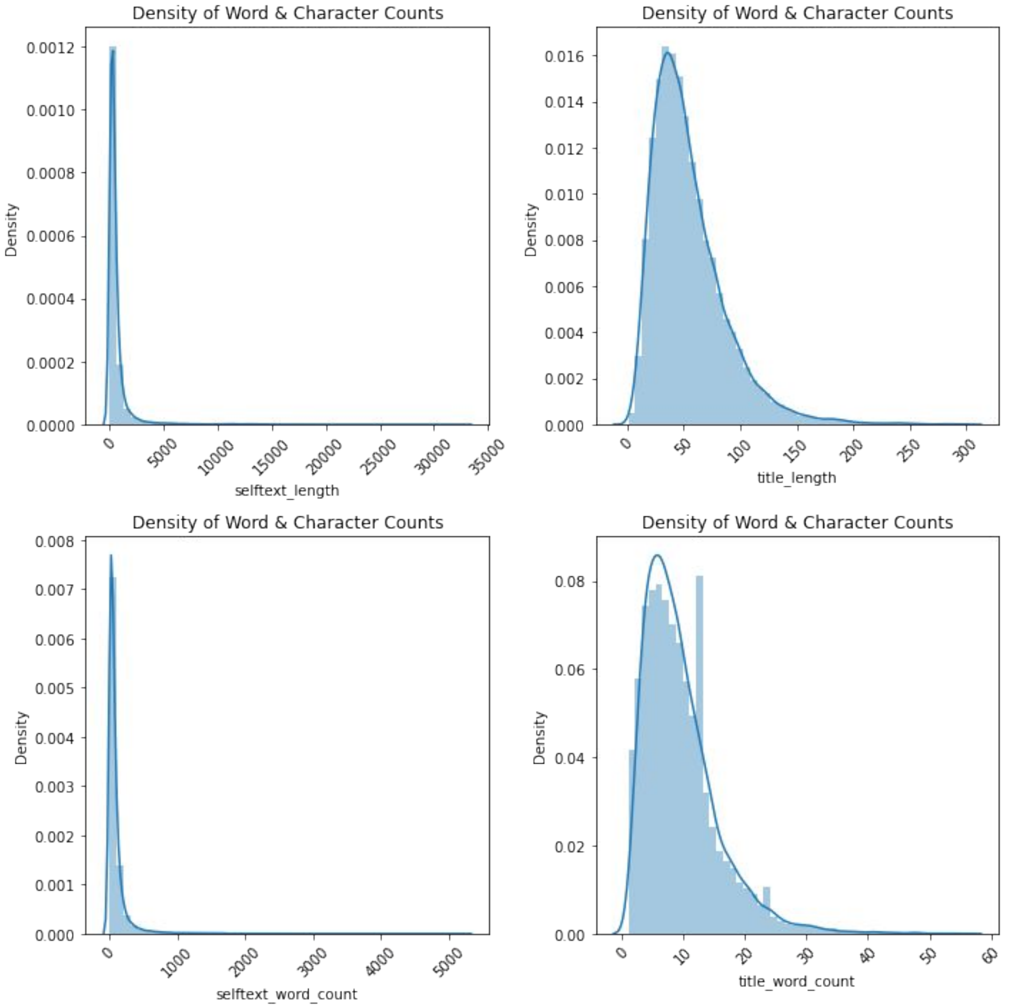
		title	selftext	subreddit	created_utc	author	num_comme	score	is_self	timestamp
2	0	Are there an	I tried to goo	Marvel	1611023005	just-another-	8	1	TRUE	1/18/21
3	1	Avengers End	First you	Marvel	1611024089	yotta_e	3	1	TRUE	1/18/21
4	2	Does thor have the odinfo		Marvel	1611025030	abseedypete	3	1	TRUE	1/18/21
5	3	How do the s	Both story,Äi	Marvel	1611028099	meggamatty	1	1	TRUE	1/18/21
6	4	Why did Doctor Strange op		Marvel	1611028193	boodiboodi	4	1	TRUE	1/18/21
7	5	Content mos	Hey so im	Marvel	1611028722	GingerGod69	7	1	TRUE	1/18/21
8	6	What TPBs c	Interested in	Marvel	1611030147	mylittledracu	4	1	TRUE	1/18/21
9	7	How could a	I hope they n	Marvel	1611032830	Imabiglose				
10	8	The Science	General	Marvel	1611033454	88y53				
11	9	Spiderman n	[removed]	Marvel	1611038275	TimeRealm				
12	10	The Height o	[removed]	Marvel	1611041548	SPN1191				
13	11	Who is More	[removed]	Marvel	1611048038	AvengerKau				

					18700	The_Champ_	11	2	TRUE	4/17/18
					21197	tetsudai	3	0	TRUE	4/17/18
					27302	Tomhur	11	1	TRUE	4/9/18
					28417	Magicknight9	4	2	TRUE	4/9/18
					29383	Isunova	11	1	TRUE	4/9/18
					30386	hazhaq	2	0	TRUE	4/9/18
					31295	hikesometrai	4	3	TRUE	4/9/18
					40332	mirois	7	3	TRUE	4/9/18
					47481	MyStupidRar	1	0	TRUE	4/10/18
4685	4683	Spider-Man I	[deleted]	Marvel	1523362635	[deleted]	0	1	TRUE	4/10/18
4686	4684	When can we expect anot		Marvel	1523368508	Maestro_Ma	4	6	TRUE	4/10/18
4687	4685	What are your theories on		Marvel	1523370727	Renmoney57	6	1	TRUE	4/10/18
4688	4686	Peter Parker is from Fores		Marvel	1523375537	hyperviolator	4	6	TRUE	4/10/18
4689	4687	Post 9/11 he	Now it's	Marvel	1523375921	cgknight1	3	2	TRUE	4/10/18
4690	4688	Let's contact	Hello,	Marvel	1523378686	longernohum	1	0	TRUE	4/10/18
4691	4689	GOTG 1&am	Hey guys,	Marvel	1523379216	lml-forwards	6	0	TRUE	4/10/18
4692	4690	All new Guai	Why do the	Marvel	1523380080	varsas	3	1	TRUE	4/10/18
4693	4691	I can't descri	I've been wa	Marvel	1523380264	DaveyRocket	7	11	TRUE	4/10/18
4694	4692	Infinity War -	[removed]	Marvel	1523381692	BigPapiChee	0	1	TRUE	4/10/18

EDA

Dropped Nulls, and all self text labeled “[removed]” because the other data in the rows was not necessary

These graphs show the distribution of words and characters in the text and titles of the posts.



Setting up data for modeling

```
[15]: X = df['selftext']
      y = df['subreddit']
```

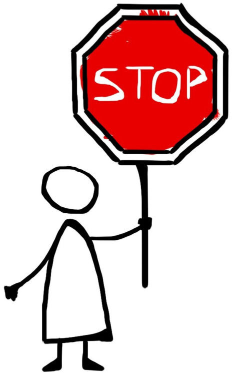
```
[16]: # Split data into the training and testing sets.
      X_train1, X_test1, y_train1, y_test1 = train_test_split(X,y,stratify=y)
```

```
[17]: #Instantiate a CountVectorizer with a default hyperparamters

      cvec = CountVectorizer(stop_words='english')
```

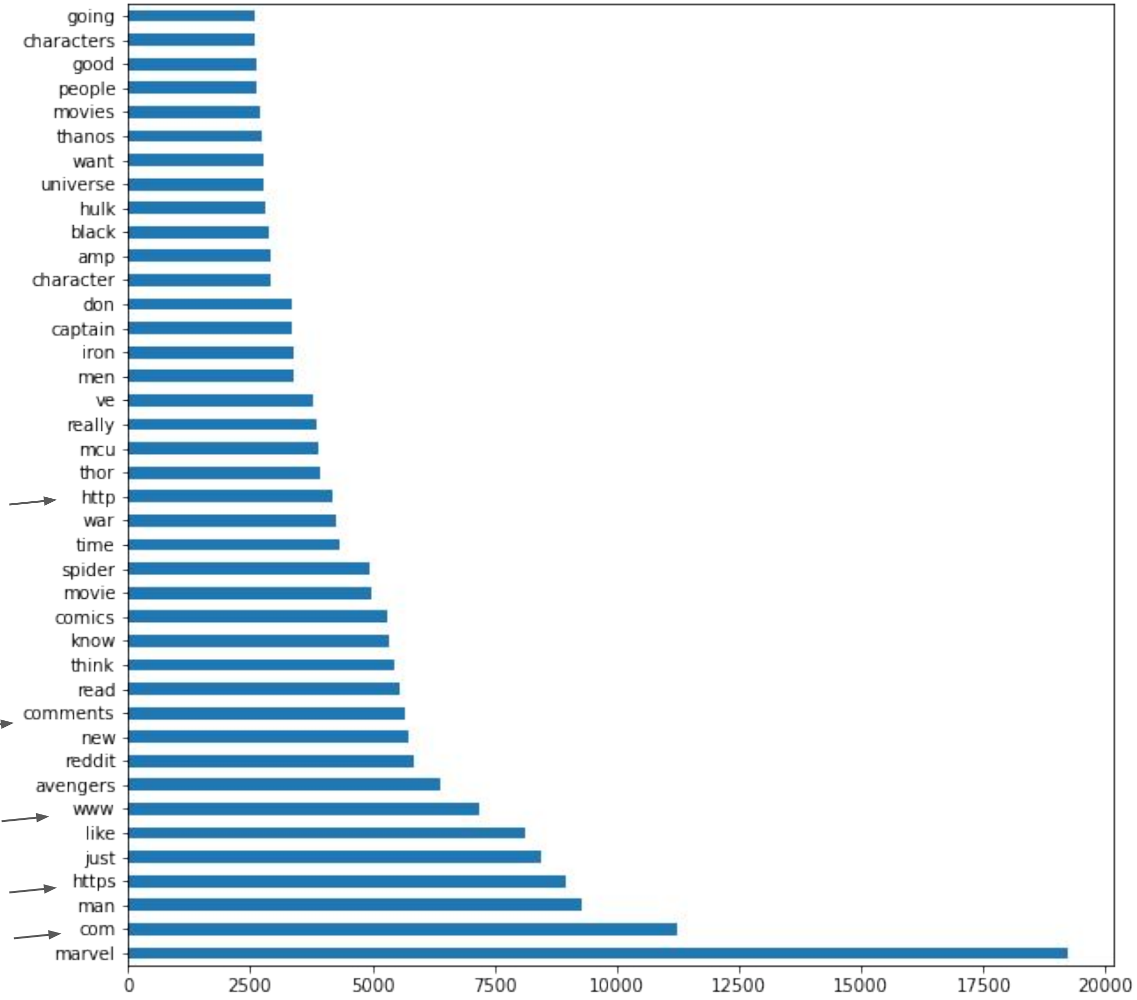
```
[18]: # Fit and transform train

      X_train1 = cvec.fit_transform(X_train1)

      # Transform the corpus.

      X_test1 = cvec.transform(X_test1)
```

```
[19]: # Plotting most used words

      X_train1_df = pd.DataFrame(X_train1.todense(),
                                 columns = cvec.get_feature_names())

      plt.figure(figsize=(10,10))
      X_train1_df.sum().sort_values(ascending=False).head(40).plot(kind='barh');
```

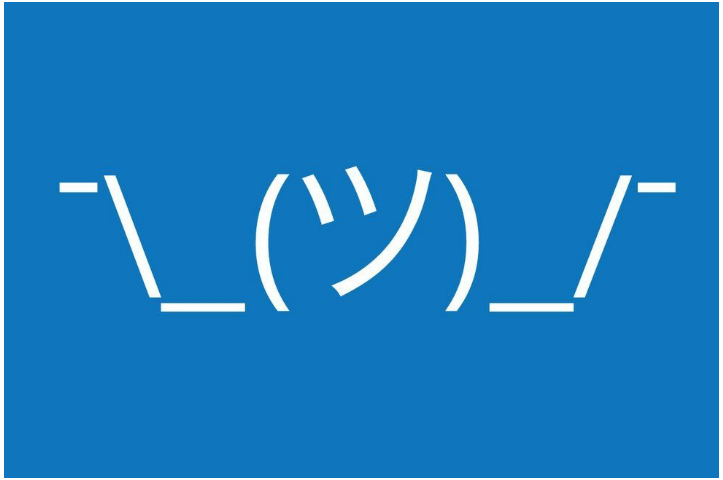
Using very basic parameters, and the basic stop words, didn’t get a very good bag of words.



This plot illustrates that the stop words need to be customized in order because words like http, https, www are hyperlinks that give no distinction between the two subreddits.



MULTINOMIAL NAIVE BAYES
MODEL with CountVectorizer

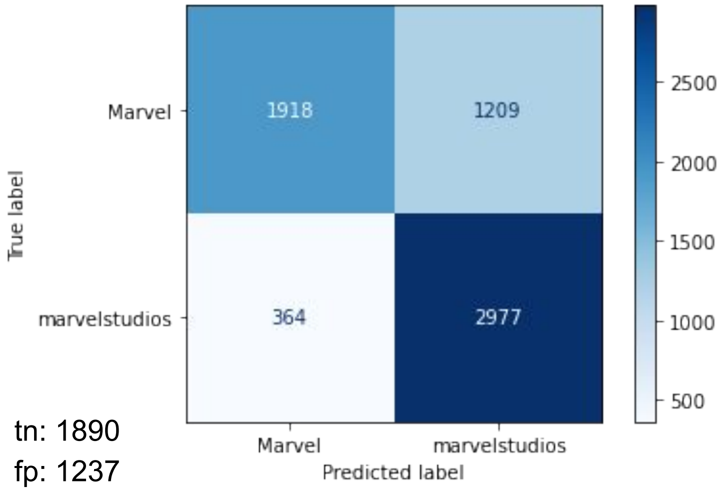


&

SUPPORT VECTOR MACHINES
MODEL with TfidfVectorizer

Multinomial Naive Bayes

The CountVectorizer ran with a Multinomial Naive Bayes model has an accuracy score of 76.3% and 76.1% which is a decently fit model and produces high number of Type I errors. The model performs semi-well with predictions on the test data.



tn: 1890
fp: 1237
fn: 311
tp: 3030

Let's set a pipeline up with two stages:

```
[27]: # 1. CountVectorizer (transformer)
      # 2. Multinomial Naive Bayes (estimator)

      pipe = Pipeline([
          ('cvect', CountVectorizer()),
          ('nb', MultinomialNB())
      ])
```

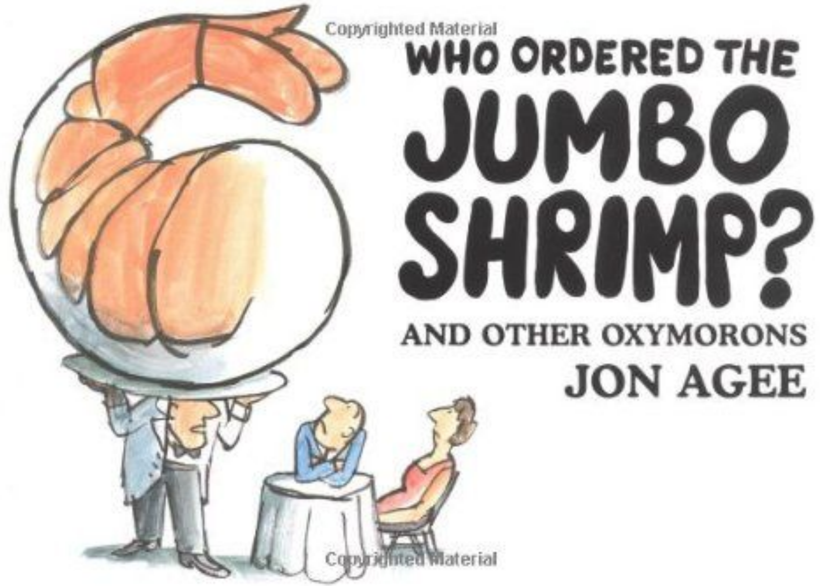
```
[*]: pipe_params = {
         'cvect__max_features': [2000, 3000, 4000, 5000], # Capping features
         'cvect__min_df': [2, 3], # Word has to show up in more than 2-3 documents
         'cvect__max_df': [.9, .95], # Word can't show up in 90% and 95%
         'cvect__ngram_range': [(1,1), (1,2)], # 1-gram, 2-gram
         'cvect__stop_words': [None, 'english',stop_words], # Using stop_words
     }
```

```
[29]: # Instantiate GridSearchCV.

      gs = GridSearchCV(pipe, # what object are we optimizing?
                        param_grid=pipe_params, # what parameters values are we searching?
                        cv=5, verbose = 1,
                       n_jobs = -2) # 5-fold cross-validation.
```

```
[30]: # Fit GridSearch to training data.

      gs.fit(X_train, y_train)

      Fitting 5 folds for each of 32 candidates, totalling 160 fits
      [Parallel(n_jobs=-2)]: Using backend LokyBackend with 15 concurrent workers.
      [Parallel(n_jobs=-2)]: Done  20 tasks      | elapsed:   13.0s
      [Parallel(n_jobs=-2)]: Done 160 out of 160 | elapsed:  1.4min finished

[30]: GridSearchCV(cv=5,
                   estimator=Pipeline(steps=[('cvect', CountVectorizer()),
                                             ('nb', MultinomialNB())]),
                   n_jobs=-2,
                   param_grid={'cvect__max_df': [0.9, 0.95],
                               'cvect__max_features': [2000, 3000, 4000, 5000],
                               'cvect__min_df': [2, 3],
                               'cvect__ngram_range': [(1, 1), (1, 2)]},
                   verbose=1)
```

Pros:

1. Very fast modeling
2. Excellent Classifier, lesser complication.

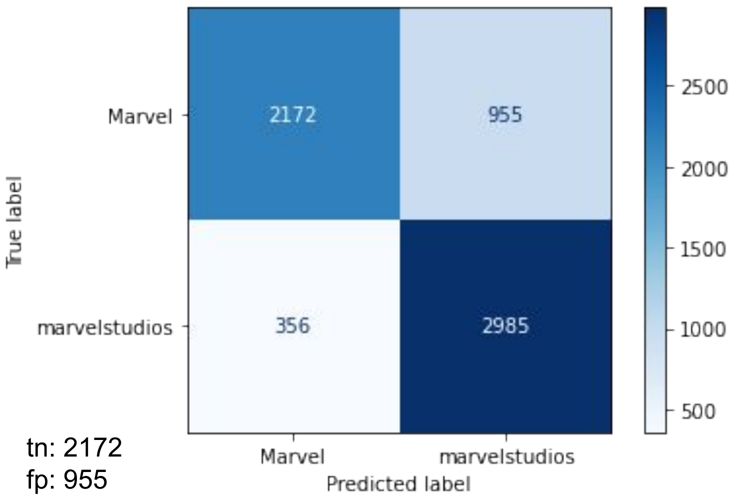


Cons:

1. Naive Bayes assumes each feature is independent from one another. Text data is never independent. Jumbo Shrimp.

Support Vector Machines

Running a TfidfVectorizer transformer on a Support Vector Machines estimator spat out a decently overfit model. 91.9% accuracy score on training data and 79.7& on testing data with significant Type I errors but better than the Naive Bayes Model.



tn: 2172
fp: 955
fn: 356
tp: 2985

Modeling with TfidfVect and Support Vector Machines

```
[61]: # 1. TfidfVectorizer (transformer)
      # 2. Support Vector Machines (estimator)


      pipey = Pipeline([
          ('tvec', TfidfVectorizer()),
          ('svc', SVC())
      ])
```

```
[65]: pipe_tvec_params = {
          'tvec__max_features': [500, 1_000, 2_000, 3_000, 4_000, 5_000],
          'tvec__stop_words': [None, 'english',stop_words],
          'tvec__ngram_range': [(1,1), (1,2)],
          'svc__C': np.linspace(0.00001,2,10),
          'svc__kernel': ['poly', 'rbf'],
          'svc__gamma': ['scale', 'auto']
      }
```

```
[66]: # Instantiate GridSearchCV.

      gsv = GridSearchCV(pipey, # what object are we optimizing?
                         param_grid=pipe_tvec_params, # what parameters values are we searching?
                         cv=5, verbose = 1,
                        n_jobs = -3) # 5-fold cross-validation.
```

```
[*]: # Fit and wait

     gsv.fit(X_train,y_train)

     Fitting 5 folds for each of 1440 candidates, totalling 7200 fits
     [Parallel(n_jobs=-3)]: Using backend LokyBackend with 14 concurrent workers.
     [Parallel(n_jobs=-3)]: Done  22 tasks      | elapsed:  3.7min
     [Parallel(n_jobs=-3)]: Done 172 tasks      | elapsed: 28.9min
     [Parallel(n_jobs=-3)]: Done 422 tasks      | elapsed: 70.0min
     [Parallel(n_jobs=-3)]: Done 772 tasks      | elapsed: 129.0min
     [Parallel(n_jobs=-3)]: Done 1222 tasks      | elapsed: 200.0min
```

Pros:

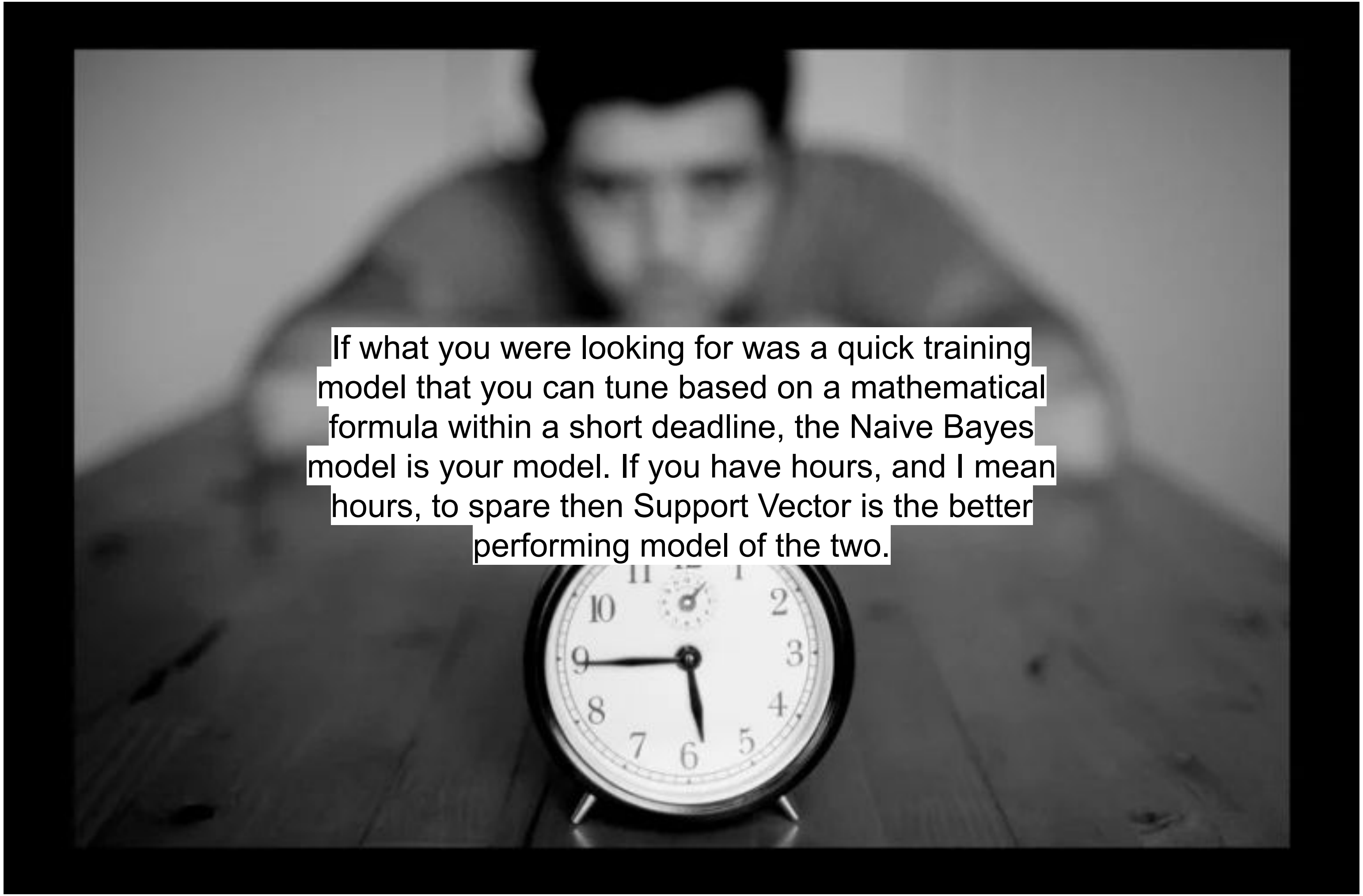
1. The model performs really well.
2. Effective in high-dimensional data
3. Can work with non-linear boundaries
4. Fast to compute with most datasets(kernel trick)

```
Fitting 5 folds for each of 1440 candidates, totalling 7200 fits
[Parallel(n_jobs=-3)]: Using backend LokyBackend with 14 concurrent workers.
[Parallel(n_jobs=-3)]: Done  22 tasks      | elapsed:  3.7min
[Parallel(n_jobs=-3)]: Done 172 tasks      | elapsed: 28.9min
[Parallel(n_jobs=-3)]: Done 422 tasks      | elapsed: 70.0min
[Parallel(n_jobs=-3)]: Done 772 tasks      | elapsed: 129.0min
[Parallel(n_jobs=-3)]: Done 1222 tasks      | elapsed: 200.0min
```

Cons:

1. Black box model. There isn't an explicable rhyme or reason, it works.
2. Very slow to train. Very slow.

(I started my fit before I left for work and it was done in 4 hours. I missed a lot of hyper parameters and ran it again at 1AM. It's almost 6AM.)



If what you were looking for was a quick training model that you can tune based on a mathematical formula within a short deadline, the Naive Bayes model is your model. If you have hours, and I mean hours, to spare then Support Vector is the better performing model of the two.