Book Genre Prediction Model

Final Project

Sarah Demmon, Kimani Phillips, Paola Roman, Chantal Thomas

Contents

Paola Speaking...

Purpose

Paola Speaking...

To train and test a book genre prediction model based off of thousands of book plot summaries. As an added Bonus, we will create a “Word Cloud” visualisation of the most frequently used words found in the books summary text.

Data Sources & Inspiration

Paola Speaking...

- Over **16K book titles** found in a *Carnegie Mellon University* Kaggle project dataset
  - This included Authors, Book Summaries, Genres, Publication Date, Wikipedia and Freebase IDs
  - We filtered to the most popular genres
    - Children's Literature, Crime Fiction, Fantasy, Mystery, Non-fiction, Science Fiction, Young adult literature
  - Using dropna and filtering, we were left with 10,096 entries
- Copilot Resource
- ChatGPT forums
- Shweta S. Enterprise Solutions Architect @ AWS | Deep Learning SME

Methods

- In order to utilize our machine learning frameworks, **preprocessing** and **data cleaning** were necessary in order for our frameworks to understand text sequences
  - Read the data in the text file and saved it to a dataframe
  - Deleted all the rows where the values for genre and summary were empty using .dropna
  - The values in column "Genre" were in json format, so we converted them to list format
  - The Genre field applied multiple labels to each book title. We decided to simplify the Genre field and, therefore, decreased to 1 genre per book

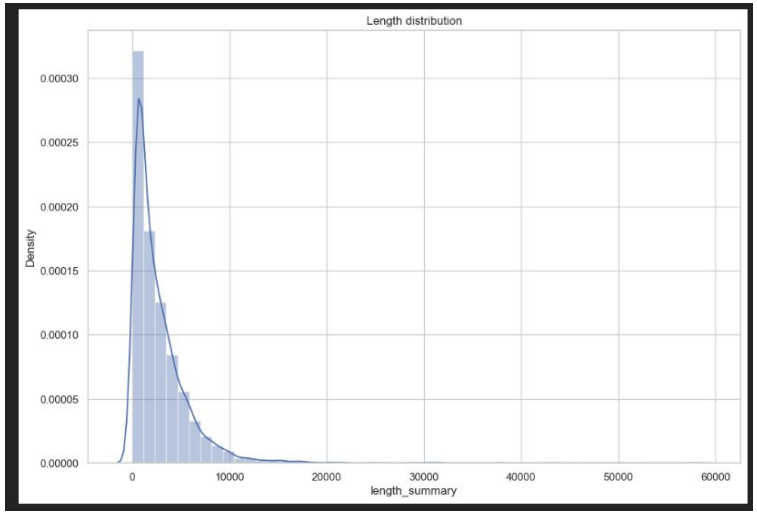
    For example, "**A Clockwork Orange**" written by Anthony Burgess was assigned *Science Fiction, Fantasy, Mystery, Crime* as genres

Methods

```
df.head()
```

	title	author	genres	summary
0	Animal Farm	George Orwell	[Roman à clef, Satire, Children's literature, ...	Old Major, the old boar on the Manor Farm, ca...
1	A Clockwork Orange	Anthony Burgess	[Science Fiction, Novella, Speculative fiction...	Alex, a teenager living in near-future Englan...
2	The Plague	Albert Camus	[Existentialism, Fiction, Absurdist fiction, N...	The text of The Plague is divided into five p...
3	An Enquiry Concerning Human Understanding	David Hume	None	The argument of the Enquiry proceeds by a ser...
4	A Fire Upon the Deep	Vernor Vinge	[Hard science fiction, Science Fiction, Specul...	The novel posits that space around the Milky ...

1. Conversion of Genre list
2. Counted the occurrence of each unique genre
3. Chose 8 main genres
4. Set features
    a. Removed “stop words”



Revisions to Process

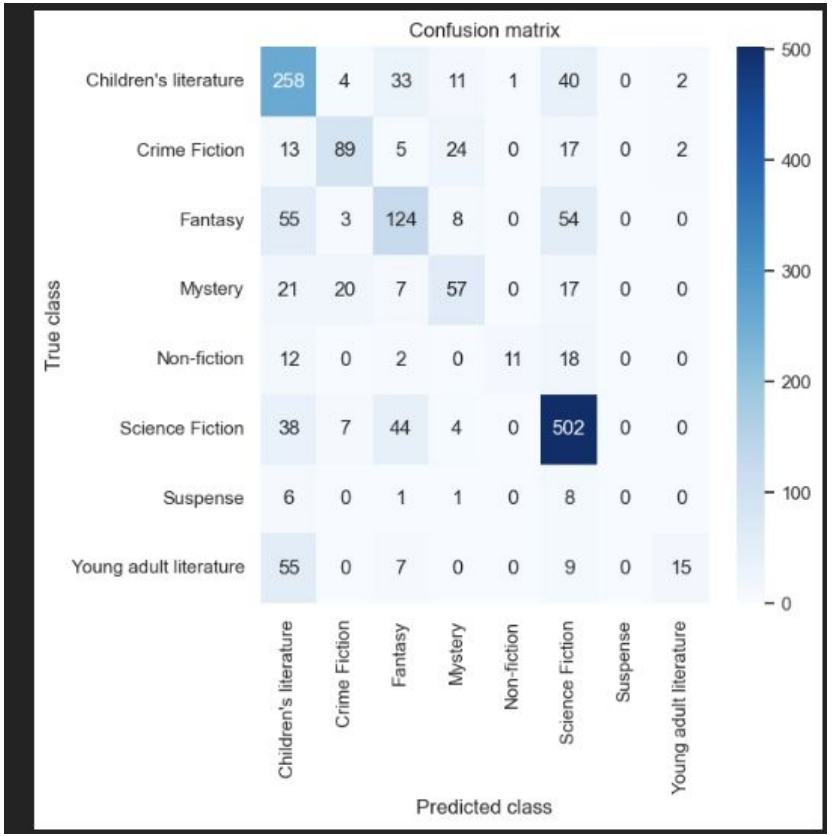
**Logistic Regression model example*

- Removed suspense
- Added feature
  - Max words = 1000

Result:

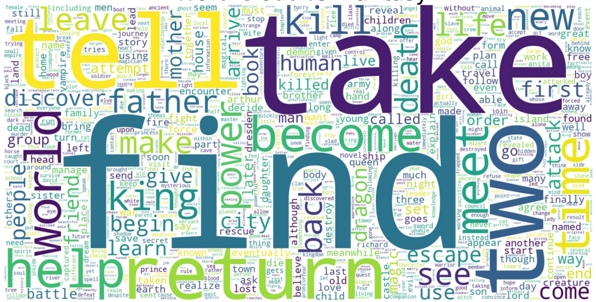
- Train Accuracy: 82.5%
- Test Accuracy: 70.3%

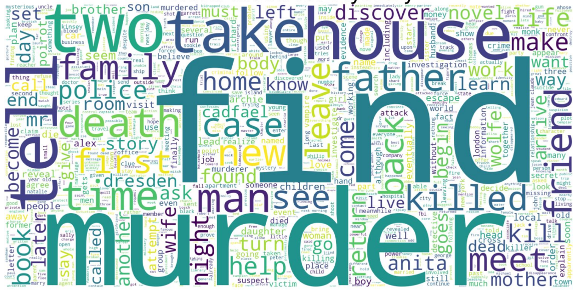
	precision	recall	f1-score	support
Children's literature	0.56	0.74	0.64	349
Crime Fiction	0.72	0.59	0.65	150
Fantasy	0.56	0.51	0.53	244
Mystery	0.54	0.47	0.50	122
Non-fiction	0.92	0.26	0.40	43
Science Fiction	0.75	0.84	0.80	595
Suspense	0.00	0.00	0.00	16
Young adult literature	0.79	0.17	0.29	86
accuracy			0.66	1605
macro avg	0.61	0.45	0.48	1605
weighted avg	0.66	0.66	0.64	1605



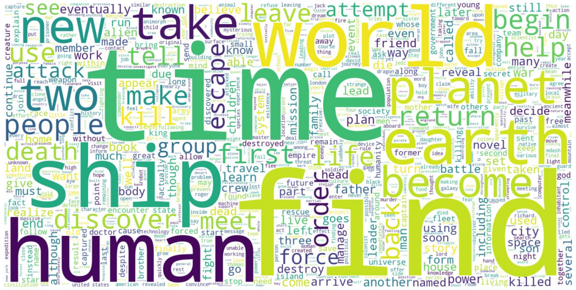
Text as Data Visualization


Word Cloud for Crime Fiction


Word Cloud for Fantasy


Word Cloud for Mystery


Word Cloud for Non-fiction


Word Cloud for Science Fiction


Word Cloud for Suspense

Building the Model(s)

Classification Models

*Require **pre-processing** and transformation of text data into numerical features before training the models*

- **SVC** (Support Vector Classifier): supervised learning model used for classification tasks
- **Logistic Regression**: Used for binary classification and predicts the probability of each class
- **Naive Bayes:** Probabilistic classification algorithm based on Bayes' theorem with an assumption of independence between features
- **XGBoost** (Extreme Gradient Boosting): builds a series of decision trees sequentially, where each tree corrects the errors made by the previous ones.

Neural Networks

***Built-in mechanisms** to process and learn from raw text data*

- **LSTM** (Long Short-Term Memory): learn patterns in sequential data
  - Great for sentiment analysis
- **BERT** (Bidirectional Encoder Representations from Transformers): uses a transformer architecture

Libraries

Classification Models

- **scikit-learn:**
  - LogisticRegression
  - Pre-processing tools like CountVectorizer or TfidfVectorizer ↠ used for text data transformation
- **TfidfVectorizer:** It's a feature extraction method that converts text documents into numerical vectors.
- **OneVsRestClassifier:** It's a wrapper that allows using binary classifiers for multiclass classification tasks
- **Pipeline**: integration of feature extraction and classification steps, making it easier to train and evaluate models.
- **CountVectorizer**: convert a collection of text documents into a matrix of token counts
- **xgboost**: This library provides an efficient and scalable implementation of gradient boosting algorithms

Neural Networks

- **transformers** (formerly known as pytorch-transformers): provides pre-trained BERT models and tokenizers for both PyTorch and TensorFlow.
- **_tensorflow or torch:** Both TensorFlow and PyTorch provide implementations of Bert and LSTM cells and layers for building recurrent neural networks.
- **keras**: If using TensorFlow, you can also use the Keras API, which provides a high-level interface for building neural networks, including LSTM models

Data Model Optimization

Naïve Bayes - Tuned to Highest Accuracy

```
#### Model Evaluation on Test Data: ####
Train Accuracy : 0.586
Test Accuracy : 0.457
```

[13]:

```
benchmarks = {'NB' : [0.0, 0.0, 0.0],
              'NB_tuned':  [0.0, 0.0, 0.0],
                }
```

[14]:

```
t0 = time()
parameters = {
    'tfidf__use_idf': (True, False),
    #'tfidf__lowercase': (True, False),
    'tfidf__norm': ('l1', 'l2'),
    'clf__estimator__alpha': (1, 0.1, 0.01, 0.001, 0.0001)
     }
NB_grid = GridSearchCV(NB_pipeline, param_grid=parameters, n_jobs=-1, verbose=5)
NB_grid.fit(train_x, train_y)
#print("Training took: {:.2f} ".format(time() - t0))
benchmarks['NB_tuned'][0] = (time() - t0)/60
```

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits
```

[15]:

```
print("####After tuning:####")
print('Train Accuracy : %.3f'%NB_grid.best_estimator_.score(train_x, train_y))
print('Test Accuracy : %.3f'%NB_grid.best_estimator_.score(test_x, test_y))
```

```
####After tuning:####
Train Accuracy : 0.980
Test Accuracy : 0.720
```

[ ]:

```
```

Process:

- Used NB_Pipeline
- Set parameters
- GridSearch: performs an exhaustive search over a specified parameter grid

```
####After tuning:####
Train Accuracy : 0.980
Test Accuracy : 0.720
```

Data Model Optimization

Naïve Bayes - Tuned to Highest Accuracy

```
pred_nb = NB_grid.best_estimator_.predict(test_x)
pred_nb_df = save_print_results(pred=pred_nb, labels=test_y, titles=test_titles, save_file='.
pred_nb_df.head(30)
```

```
#########################################
#       Test accuracy is 71.9626%       #
#########################################
```

	titles	genres	prediction	result
0	What I Was	Young adult literature	Children's literature	Wrong
1	A Great and Terrible Beauty	Fantasy	Children's literature	Wrong
2	Gilded Latten Bones	Fantasy	Fantasy	Correct
3	Five Go Off In A Caravan	Mystery	Children's literature	Wrong
4	100 Cupboards	Children's literature	Children's literature	Correct
5	Anastasia, Ask Your Analyst	Young adult literature	Children's literature	Wrong
6	In Spite of Thunder	Mystery	Crime Fiction	Wrong
7	The Infinity Doctors	Science Fiction	Science Fiction	Correct
8	The Child of the Cavern	Science Fiction	Science Fiction	Correct
9	These Our Actors	Science Fiction	Children's literature	Wrong
10	Christy	Children's literature	Science Fiction	Wrong

Prediction Results

Additional Model Results

XGBoost

```
Training took: 0.000[seconds] to complete and has been saved as ./XGB_model.sav
####Before tuning:####
Train Accuracy : 0.986
Test Accuracy : 0.634
```

Bert Model

```
# Train the model
model.fit(train_dataset.shuffle(1000).batch(4), epochs=3, batch_size=4)
```

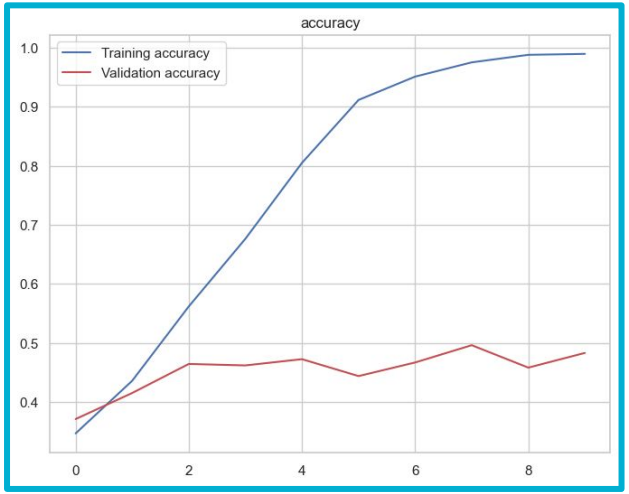
```
Epoch 1/3
WARNING:tensorflow:AutoGraph could not transform <function infer_framework at 0x1493cd760> and will run it as-is.
Cause: for/else statement not yet supported
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <function infer_framework at 0x1493cd760> and will run it as-is.
Cause: for/else statement not yet supported
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert


2024-04-22 13:52:10.757195: W tensorflow/core/framework/local_rendezvous.cc:404] Local rendezvous is aborting with status: I
NVALID_ARGUMENT: indices[0,13733] = 13733 is not in [0, 512)
         [[{{node tf_distil_bert_for_sequence_classification/distilbert/embeddings/Gather_1}}]]


---------------------------------------------------------------------------
InvalidArgumentError                      Traceback (most recent call last)
Cell In[17], line 2
      1 # Train the model
----> 2 model.fit(train_dataset.shuffle(1000).batch(4), epochs=3, batch_size=4)

File ~/anaconda3/lib/python3.11/site-packages/transformers/modeling_tf_utils.py:1229, in TFPreTrainedModel.fit(self, *args,
**kwargs)
    1226 @functools.wraps(keras.Model.fit)
    1227 def fit(self, *args, **kwargs):
```

LSTM



Logistic Regression

```
Training took: 0.000[seconds] to complete and has been saved as ./LogReg_model.sav
####Before tuning:####
Train Accuracy : 0.825
Test Accuracy : 0.703
```

Data Model Optimization

SVC - Tuned to Highest Accuracy

```
Accuracy: 0.44554455445544555
                        precision    recall  f1-score   support

 Children's literature       0.39      0.50      0.44       187
               Fantasy       0.40      0.40      0.40       219
               Mystery       0.39      0.35      0.37       141
           Non-fiction       0.92      0.43      0.59        28
       Science Fiction       0.61      0.66      0.63       296
              Suspense       0.02      0.02      0.02        65
Young adult literature       0.38      0.19      0.25        74

              accuracy                           0.45      1010
             macro avg       0.44      0.36      0.38      1010
          weighted avg       0.45      0.45      0.44      1010

Training Accuracy: 0.7473035439137135
Testing Accuracy: 0.44554455445544555
```

```
# Splitting the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=25)

# Extracting features
vectorizer = TfidfVectorizer(max_features=1000)

# Selecting SVC model
model = Pipeline([
    ('tfidf', TfidfVectorizer(stop_words=stop_words, ngram_range=(1,2))),
    ('clf', OneVsRestClassifier(LinearSVC(), n_jobs=1)),
])
```

```
Training Accuracy: 0.7473035439137135
Testing Accuracy: 0.44554455445544555
```

Documentation

Project-4 Public

Watch 1 | Fork 0 | Star 0

main | 1 Branch | 0 Tags | Go to file | Add file | Code

sdemmon bert, xgb, lstm, lr, NB — e178449 · 1 minute ago — 35 Commits

.ipynb_checkpoints	bert, xgb, lstm, lr, NB	1 minute ago
images	latest readme	4 hours ago
.DS_Store	bert, xgb, lstm, lr, NB	1 minute ago
BERT_2.ipynb	bert, xgb, lstm, lr, NB	1 minute ago
LSTM.ipynb	bert, xgb, lstm, lr, NB	1 minute ago
Logistic Regression Accuracy.ipynb	bert, xgb, lstm, lr, NB	1 minute ago
Naive Bayes.ipynb	bert, xgb, lstm, lr, NB	1 minute ago
README.md	Update README.md	2 hours ago
XGB.ipynb	bert, xgb, lstm, lr, NB	1 minute ago
booksummaries.txt	rough draft	last week
starter_code.ipynb	latest readme	4 hours ago
svc_model.ipynb	paola final	4 days ago
word_cloud.ipynb	paola final	4 days ago

README

Project-4

Book Genre Prediction Model

Final Project

About

Book Genre Prediction Model

Readme
Activity
0 stars
1 watching
0 forks
Report repository

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 4

- ChantalThom ChantalThom
- sdemmon
- paolaromanvaldes
- KimaniPhillips12

Languages

- Jupyter Notebook 100.0%

README

Introduction

Our objective for this project was to train and evaluate various machine learning models to determine which deep learning model could effectively learn and predict a book's genre based on its plot summary. Using several supervised machine learning models (below), we effectively tested and trained a database of 16K book titles to predict book genres.

Data Sources

- Kaggle Dataset of Carnegie Mellon University book summaries - https://www.cs.cmu.edu/~dbamman/booksummaries.html

CMU Book Summary Dataset

The CMU Book Summary Dataset supports ongoing work described in:

David Bamman and Noah Smith (2013), "New Alignment Methods for Discriminative Book Summarization." [ArXiv]

booksummaries.tar.gz [17M]

This dataset contains plot summaries for 16,559 books extracted from Wikipedia, along with aligned metadata from Freebase, including book author, title, and genre.

All data is released under a Creative Commons Attribution-ShareAlike License. For questions or comments, please contact David Bamman (dbamman@cs.cmu.edu).

Example

The following example illustrates the data and metadata available for Don DeLillo's *White Noise*.

Book metadata

Wikipedia ID	1166383
Freebase ID	/m/04cvx9
Book title	White Noise
Book author	Don DeLillo
Publication date	1985-01-21
Genres	Novel, Postmodernism, Speculative fiction, Fiction

Plot summary

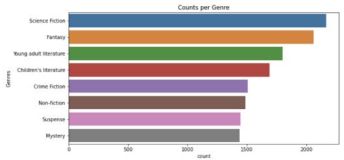
Set at a bucolic Midwestern college known only as The-College-on-the-Hill, White Noise follows a year in the life of Jack Gladney, a professor who has made his name by pioneering the field of Hitler Studies (though he hasn't taken German language lessons until this year). He has been married five times to four women and has a brood of children and stepchildren (Heinrich, Denise, Steffie, Wilder) with his current wife, Babette. Jack and Babette are both extremely afraid of death; they frequently wonder which of them will be the first to die. The first part of White Noise, called "Waves and Radiation," is a chronicle of contemporary family life combined with academic satire.

- CoPilot Resources
- ChatGPT forums
- Project Inspiration from Shweta S. Enterprise Solutions Architect

Data Cleaning

The first step is to understand the data and make it adequate for our neural network.

- Create Dataframe: Read in from the text file which contains all the book summaries, genre, author, book title, etc. and visualize the data and save it to a dataframe.
- Delete all the rows where the values for genre and summary are empty because these rows won't be any use for us.
- Genres: The genres had multiple labels for each book. We will take into account only one label. To do so, a study was done to see which genres are the more frequent ones and discard those that aren't.



- We also analyzed the text summaries. To make the text more suitable for our models, we have to remove the "stopwords" (e.g. i,me,my,myself,we,our,ours,ourselves,you,your,yours,a,an,the,and,but,if,or,because,as)

```
#Download stopwords from nltk
nltk.download('stopwords')
stop_words = list(stopwords.words('english'))

#Perform count vectorization
cv = CountVectorizer(analyzer='word', max_features=None, strip_accents='ascii', stop_words=st

X_cnt = cv.fit_transform(book_df['summary'])

[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/sarahdemmon/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Discussion

Questions?

Last presentation of the night

Congratulations Everyone!