Elle winks at Amy and heads off as the guys launch into their story. It was classic. Jason hit with wood all day. And we never ran out of balls -- Amy laughs. . - Serena and Margot watch as Elle stands in front of a mirror, trying on a Herve Leger white mini-dress. Sexy and ultra- tight. ; Too demure? think you should go with red. It's the color of confidence. Well, don't want to look like

Let’s say someone handed you an eighty-word snippet from a screenplay and asked you to figure out who wrote it, where your only information is the text at hand. Not only that, but you’re not given character names or any of the screenplay information. How well do you think you could determine who wrote it? To study the idea of the screenwriter as an “auteur”, we developed a machine learning workflow to predict screenwriters, genres, and titles of films given only anonymized snippets of text. Our dataset is a group of sixteen films by four different screenwriters, chosen based on text availability and each writer’s prolificness. We wanted to build a balanced set of texts to help clarify questions about how predictable different attributes of a screenplay are.

Designing the study

When classifying data, in particular textual data, it’s important to focus on the limitations of the dataset and ways that confounders can manipulate results. For instance, if one screenwriter is overrepresented in the dataset, it is likely that a classifier will be more accurate at predictions because it will favor the overrepresented screenwriter. One possible method for setting up the experiment would have been to offer a complete screenplay to the predictive algorithm, but this would mean that there is very little training data: It’s unlikely that screenwriters have written more than a few films (in our dataset, the most prolific authors had five films represented). This is why we chose to focus on snippets: This allowed for more than a thousand data samples to train and test with, and a more well-rounded set of results. We chose eighty words for each sample as it represents about the length of short paragraph—enough to get a sense of the underlying text without being enough to end up with a small sample of data.

We also wanted to compare the results for the accuracy of our classifier on screenwriters to its accuracy on other attributes—even if we knew how accurate the classifier was at screenwriter classification, it would be hard to gauge the individuality of each writing style without relative comparisons to other classifications. To this end, we chose genre as a comparative measure. Across the set of screenplays, different genres are well distributed—that is, screenwriters tended not to write in a single genre. For instance, William Goldman’s work spans several genres, from horror and action to comedy and kids movies. Genres for movies are based on tags from Rotten Tomatoes, and are listed in order—the first listed genre is the movie’s first genre, the second listed genre is the movie’s second genre, etc. Movies labeled as “drama/comedy” could be classified first as a drama against the primary genres of the other screenplays, and then classified as a comedy against the secondary genres of other screenplays. We considered classifying against other attributes, but generally they were either poorly distributed or represented too many classes. For instance, directors tended to work on the same films as screenwriters, so the classes were too similar to derive anything meaningful from. On the other end of the spectrum, every film tended to have a different cast, so there were too many classes to derive something meaningful from predicting the top-billed actor in each film.

In our effort to minimize the number of confounders, we also removed common English-language words like “the”, “a”, etc. and removed character names from the texts. We don’t want the classifier over-weighting character names to “cheat” at figuring out the screenwriter—we wanted to gauge the way textual style generates differences in screenplays.

Once we had set up our dataset, we used several predictive algorithms to classify the data. We settled on using a support vector machine[[1]](#footnote-1) to classify the data, and tested how different parameters of the machine affected its accuracy testing on different attributes. There is one clear limitation of our dataset: We’re basing the classification wholly on vocabulary. There is no contextual awareness for the classifier—at the end of the day, it’s likely that any accurate textual classifier is working because some screenwriters use certain words more than other screenwriters.

know what's coming.

1. The general idea behind most ML algorithms is to take a sample dataset and train the parameters of a model on that sample to most accurately predict classes of data on a held-out validation set. Support Vector Machines work by plotting all the data points in space and attempt to separate them into classes with a plane. The idea is to separate the datapoints by class with the largest margin possible, i.e. points on either end of the separation boundary are as far from the boundary as possible. The classifier will predict an initial boundary, assuming all points have an equal weight. Once that boundary is placed, the classifier will re-weight the points such that misclassified points have a higher weight and accurately classified points have a lower weight. Then, it will optimize the boundary again, and repeat the process until it’s told to stop. Intuitively, the most accurate boundary will be placed after the classifier has iterated many times. The complete set of predicted boundaries can also be used in tandem, and weighted based on their accuracy, which will provide a much more accurate final result—predicting a class is easier if you can take the average guess given many different initial guesses. Because the classifier is attempting to repeatedly separate data in a high-dimensional space, it works very well for text, which has many dimensions—each individual word is treated as an individual dimension. This is also why it is a good algorithm for classification, because it is literally attempting to separate the data into different classes with a line. [↑](#footnote-ref-1)