Understanding VADER: Using LIME to overcome the accuracy-interpretability trade-off

Arturo Soberón

June 2022

Abstract

Low bias and low variance are desirable properties that any *good* model is expected to have. Unfortunately, the quest to optimize this trade-off often leads to statistical learning methods that can no longer be interpreted due to their high levels of complexity. Depending on the context, being able to understand the logic behind a model's decision making process may be more important than maximizing its accuracy. Must increasing model complexity always be done at the expense of interpretability? In this article, I review LIME — a partial solution to this dilemma — and use it to disentangle the logic behind a popular natural language processing model.

1 Introduction

In general, low bias and low variance are desirable properties that any *good* model is expected to have. Depending on the context however, being able to understand the logic behind a model's decision making process can be more important than the aforementioned characteristics. Unfortunately, there are times when the rules of association between a set of predictors and their response cannot be accurately approximated by a simple model. When this is the case, the modeler is forced to turn to high-complexity models in order to increase prediction accuracy, but increased flexibility is often accompanied by a loss in interpretability.

The first question we should ask ourselves is why this trade-off matters. That is, if high-complexity methods are able to achieve better performance metrics than parsimonious models, why should we care about a loss in interpretability? The search for transparent systems has gained popularity in recent years due to the integration of Artificial Intelligence (AI) into our lives. When AI models are allowed to act autonomously, like they do in self-driving cars, our ability to interpret the system's decision making process is just as important as maximizing its accuracy.

In this article I give an overview of Local Interpretable Model-agnostic Explanations (LIME) and apply this procedure to a popular natural language processing (NLP) model. In short, LIME consists on making many slight modifications around a point of interest and fitting a parsimonious *surrogate* model to explain the behavior of the complex model within this neighborhood (Ribeiro et al., 2016b). The results show that LIME is a viable option that can partially overcome the trade-off between prediction accuracy and model interpretability.

1.1 Case Study

In order to test the viability of LIME, I need a complex model that I can probe indefinitely. In this article, I focus on VADER — a lexicon and rule-based natural language processing model trained to detect the tone of texts posted on social media (Hutto and Gilbert, 2014) — to analyze the tone of comments posted on YouTube.

The texts analyzed by the model are comments posted on the *Battlefield 2042* launch trailer.[1] I strategically chose this video because its comments section is filled with intensely negative communications expressed using internet slang. These texts represent the intended use for the VADER model and make for an arguably interesting analysis.

The chaotic nature of the comments analyzed in this article is due to the game's poor reception by the general public. The fact that *Battlefield 2042* made it into the top 10 list of worst-rated games (Tassi, 2021) one week after its release is testimony of the negativity surrounding this title. As a consequence

[1] *Battlefield* is a popular video game franchise owned by Electronic Arts, one of the biggest publishing companies in the video game industry.

of how the game is perceived by players, the comments posted on the video frequently express disappointment, disapproval or remorse.

Once scraped, I analyze the tone of each comment using VADER and then follow the LIME procedure to explain the factors that influence the score assigned to each text. Although LIME is not designed to make global explanations, the local models yield understandable insights that can generalize to the overall model.

2 Data

The comments analyzed by VADER were retrieved from the *Battlefield 2042* launch trailer (Battlefield, 2021) using the YouTube Data API. The research done by Kwon and Gruzd (2017) suggests that negative comments tend to detonate further chains of negative replies. Thus, I only considered first-level comments in an attempt to guarantee that the texts hereby presented relate to the content of the video rather than arguments between the viewers.[2]

I fetched a total of $5,000$ first-level comments sorted by the date they were posted (in descending order). Since VADER is not language-agnostic, I filtered out comments that were not written in English using LangID — a language identification tool that categorizes text into 97 different languages (Lui and Baldwin, 2011). In total, I considered $4,332$ comments posted within June 21, 2021 and May 12, 2022.

When a text is passed to VADER, the model returns its negative, neutral, positive and compound sentiment scores. Given the negative context of the video, I focus on comments that contain at least some degree of negativity. That is, comments whose predicted negative scores are strictly greater than zero. This accounts for a total of $2,011$ comments (46.4% of all comments written in English).[3] Table 1 shows excerpts from a few randomly chosen comments that meet these criterion.

Excerpt from comment	Score
*if it's gonna be *this* bad just dont.*	0.223
you should be sued for misleading your customers	0.148
BF2042 was the last EA game I will ever buy.	0.177
It looks like they meshed Battlefield 3 with Bad Company 2	0.332
game being so bad that almost all the servers are full of bots	0.192
I hate to see a franchise as good as this die like this	0.222

Table 1: Excerpts from randomly chosen comments

[2] Allowing for replies would not affect how LIME is implemented. This criteria was solely imposed to maintain a common theme among the comments.

[3] Once again, considering another score would not affect how LIME is implemented. This criteria was solely imposed to focus on VADER's negative score.

3 Methodology

As suggested by its full name, LIME is a model-agnostic method used to make local explanations of a complex model. The basic idea behind LIME is to make many slight modifications around a point of interest and use an interpretable model to learn how these changes affect the predictions around this point.[4] In this sense, an explanation is an interpretable model (ideally a linear model with only a few features) that is faithful to the underlying black-box model within a small neighborhood.

A black-box model is any kind of system that receives an input, returns a prediction and does not disclose the process that happens between these two stages.

Let f be a black-box model that takes a set of predictors X and uses them to make a prediction Y.

$$f : X \to Y$$

Note that no restrictions were made on X nor Y. In other words, the inputs and outputs can be numbers, strings, images, audio files, etc. Another important consideration to keep in mind is that f is not to be confused with the data-generating process. LIME is designed to explain the fitted model f and has no connection with the data-generating function that f was trained to approximate.

Although there are more types of predictive models, the most notable ones are those that return a prediction using

1. numerical or categorical data (tabular);
2. an image (image recognition); or
3. some form of text (NLP).

This articles reviews the use of LIME on NLP models. However, a formal description of the procedure and a brief review of how to apply LIME to all three types of models is presented in the following subsections.

3.1 Definition of an explanation

Following the work by Ribeiro et al. (2016b), let g be a model from a class of interpretable models G (additive models or decision trees) that uses m features to predict Y. Each of the m features used by g are dichotomous variables that indicate the presence or absence of the components used by the black-box model to make a prediction.

Let $\Omega(g)$ be a measure of the complexity of model g. For example, $\Omega(g)$ could be the $L1$ norm of the coefficients from a linear model. Likewise, let $\pi_x(z)$ be a distance metric between an instance z to x. Finally, let $\mathcal{L}$ be a loss

[4] This implies that the black-box model can be probed on demand in order to make a prediction using each modified instance in the perturbed data set.

function to be minimized (i.e. mean squared error). The optimal explanation is given by the following optimization problem:

$$\min_{g \in G} \mathcal{L}(f, g; \pi_x) + \Omega(g) \tag{1}$$

3.2 LIME for tabular data

Models that use tabular data to make predictions usually map $X \in \mathbb{R}^p$ to $Y \in \mathbb{R}^1$. This means that the black-box model will take a vector x of size $p \times 1$ (where p is the number of features) and return a single numeric value y as the prediction.

To locally explain the prediction $f(x_0) = y_0$ at x_0, LIME samples n points $\{x_0^1, x_0^2, ..., x_0^n\}$ from a multivariate normal distribution centered around x_0. An interpretable model is then fit to the generated data within a predefined distance from x_0 using a local regression with an exponential kernel (Molnar, 2022).[5] The kernel assigns a greater weight to points close to x_0 and reduces the influence of points that are far away. This process is illustrated in Figure 1.

If the neighborhood from which the perturbed instances are sampled and the weights considered by the kernel are are reasonable, the explanation g is a faithful local approximation to the underlying model f at x_0.[6]

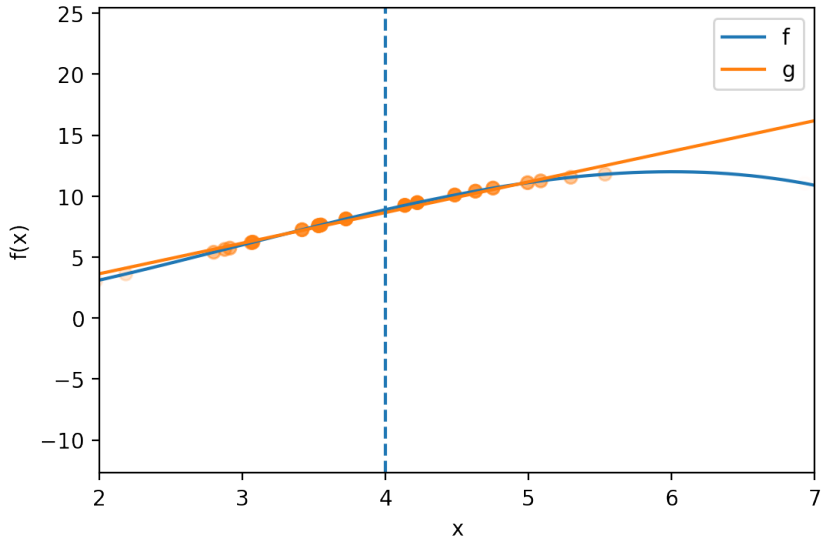

Figure 1: LIME for tabular data (univariate example)

[5]The perturbed data is standardized and the neighborhood is defined as points within $0.75 \times p$ units from 0.

[6]These are important assumptions that may undermine the credibility of LIME. However, choosing cross-validated hyperparameters can help overcome this issue.

3.3 LIME for image recognition

Assume that the black-box to be explained is a deep learning model trained to predict how likely it is for an image to contain a wolf. The leftmost image from Figure 2 represents the instance x_0 to be explained; the middle and rightmost images illustrate how LIME makes perturbations on this this type of data. The original image shows a husky lying on the snow and erroneously results in a value $f(x_0)$ close to 1.

Just as LIME for tabular data, the procedure for image recognition makes multiple modifications to the original data point in order to create a perturbed data set. LIME takes the original image and divides it into disjoint regions called *interpretable components* (also called *contiguous superpixels*). Each instance from the perturbed data set omits some of the components (represented by the grayed out regions in Figure 2) and the black-box model is used to make a prediction on each modified instance. For example, the two perturbed images in Figure 2 turn off the snow and trees respectively.

A simple model is then fit on the predictions made by f on the perturbed data set. The inputs of this surrogate model are vectors of dichotomous variables that indicate if a given contiguous superpixel is turned on or off. Each instance is weighted according to its similarity to x_0. As a consequence, the surrogate model is penalized more when it makes an incorrect prediction on instances that are very similar to the original one. The goal of this procedure is to retrieve the components that had the greatest influence on the original prediction $f(x_0)$ (Ribeiro et al., 2016a).

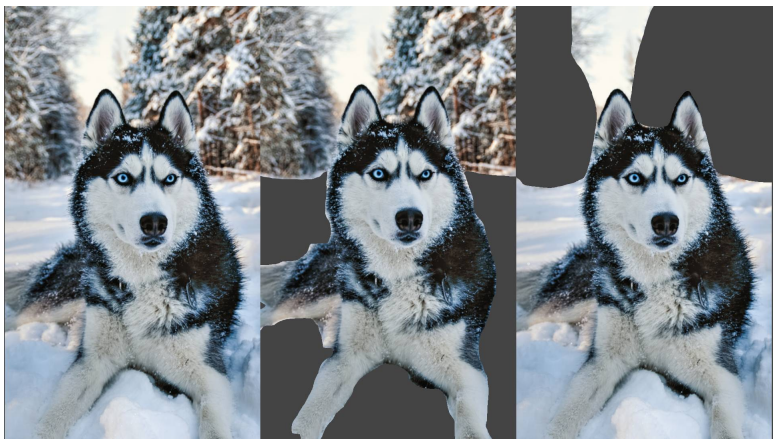

Figure 2: **Left**: Original image of a husky lying in the snow (high predicted probability). **Middle**: Perturbed image with snow removed (low predicted probability). **Right**: Perturbed image with trees removed (high predicted probability).

3.4 LIME for NLP

Much like in image recognition, LIME for NLP omits words from the original text to create a data set of perturbed instances and predicted values ($\mathcal{D}_0$).

$$\big(x_0, f(x_0)\big) \rightarrow \mathcal{D}_0 = \big\{\big(x_0^1, f(x_0^1)\big), ..., \big(x_0^n, f(x_0^n)\big)\big\}$$

For any $i \in \{1, 2, ..., n\}$, the perturbed instance x_0^i is a vector of size $1 \times m$, where m is the number of words in x_0. Each feature in x_0^i is a dichotomous variable that represents a word from the original text and shows whether or not it was removed from the i-th instance.

In its most basic form (Molnar, 2022), the weight corresponding to a perturbation in $\mathcal{D}_0$ is given by the share of words from x_0 that it includes.

$$\omega_i = \frac{1}{m} \sum_{j=1}^{m} w_j^i$$

Where $w_j^i \in x_0^i$ represents the j-th word from the original text and indicates if it is switched on or off in the i-th perturbation. A weighted surrogate model is then fit on $\mathcal{D}_0$ according to the minimization problem stated in Equation 1.

To illustrate the LIME procedure for NLP, consider the comment and the negative sentiment predicted by VADER:

$$\begin{aligned} x_0 &= \textit{How tf they managed to screw this up?} \\ f(x_0) &= 0.167 \end{aligned}$$

A perturbed data set $\mathcal{D}_0$ is generated and a surrogate model is fit to find the most influential words that drove VADER to assign a score of 0.167 to this specific comment. Five perturbed instances are shown in Table 2. Using the full version of this data set, multiple weighted Lasso models are proposed as explanations. Figure 3 shows the estimated coefficients from fitting multiple surrogate models with different penalties on $\mathcal{D}_0$.

How	*tf*	*they*	*managed*	*to*	*screw*	*this*	*up?*	ω_i	$f(x_0)$	$f(x_0^i)$
1	0	1	1	1	1	1	0	0.75	0.17	0.17
1	1	1	1	1	0	1	1	0.88	0.17	0.00
0	1	1	1	1	1	1	1	0.88	0.17	0.15
1	0	1	1	1	1	1	1	0.88	0.17	0.15
1	0	1	1	1	1	0	1	0.75	0.17	0.17

Table 2: Perturbed data set

The results from a surrogate Lasso model with the penalty parameter set to 0.11 are shown Table 3. For this arbitrary penalty (represented by the blue dotted line from Figure 3), the resulting explanation suggests that the words *How*, *tf* and *screw* are the main drivers for the score assigned to the original comment. In particular, the word *screw* is the most important factor for f around x_0. Additionally, the fact that the estimated coefficients for the words

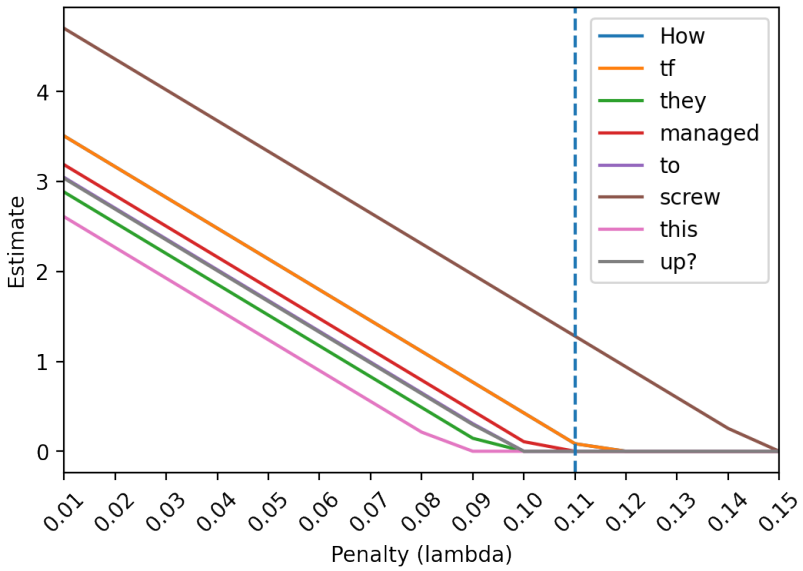

Figure 3: Surrogate models with different penalties

How and *tf* follow the same path as the penalty parameter increases suggests that VADER considers bigrams and thus the phrase *How tf* is what influences $f(x_0)$ rather than each of those words alone.

Word	Estimated Coefficient
How	0.08
tf	0.08
screw	1.28

Table 3: Surrogate model with penalty set to 0.11

4 Results

LIME is designed to make one explanation per point of interest. In order avoid presenting 2,000 individual explanations, I focus on four examples to explain why VADER made a correct or incorrect interpretation of a given comment. The cases I present in this section are:

1. a non-negative comment with a high negative score;
2. a non-negative comment with a low negative score;
3. a negative comment with a high negative score; and

4. a negative comment with a low negative score;

In order to find interesting cases, I divide the sample into quartiles according to their negative predicted scores and plot the most popular words within each group using tag clouds. These clouds are presented in Figure 4.

Figure 4: Tag clouds for each quartile

4.1 Non-negative comment with high negative score

The third and fourth quartiles contain the bigram *Bad Company*, which is the title of an older game from the *Battlefield* franchise. This is suggestive evidence of a spurious correlation that is causing VADER to assign high negative scores to comments that make a reference to this game. To test this hypothesis, I analyze the comment:

> *I still miss bad company 2*

Although the nature of the comment is not negative, it receives a score of 0.56 and is ranked in the fourth quartile in terms of predicted negativity. The explanation shown in Table 4 uses LIME to get the top-two most influential words for this comment. As hypothesized, VADER assigned a high negative score to this comment because of the word *bad*. Notwithstanding the fact that the word *miss* contributes to its predicted negativity, VADER incorrectly assigned a high score to this comment because it is unable to recognize that *bad* is used in conjunction with *company* to refer to a video game.

Word	Importance
bad	0.37
miss	0.23

Table 4: Explanation for non-negative comment with high negative score

4.2 Non-negative comment with low negative score

I analyze the comment

> *Honestly you all gotta chill. I'm going to say it. Battlefield 2042 is a good game.*

to show evidence of non-negative comments correctly interpreted by VADER. This comment received a score of 0.121 and is ranked in the lowest quartile in terms of negativity. As shown in Table 5, the LIME explanation only yields the word *battlefield*. Regardless of how low I set the penalty parameter, the explanation always returns one word because *battlefield* is the only driver for the original prediction.

Word	Importance
$battlefield$	0.13

Table 5: Explanation for non-negative comment with low negative score

This is an important consideration in the context of this article. Given that the comments come from a *Battlefield* video, many of them contain the word *battlefield* and will therefore receive high negative scores. If the modeller were to use VADER as a classifier, this concern may lead them to retrain the model in order for it to learn that the word *battlefield* does not carry a negative connotation in this context.

4.3 Negative comment with high negative score

I analyze the comment

> *This game is a disgrace*

to show evidence of negative comments correctly identified by VADER. This comment received a score of 0.444 and is ranked in the top 25% most negative comments. As shown in Table 6, the LIME explanation only yields the word *disgrace*. Just like in the previous example, this is the only word with a negative connotation.

Word	Importance
disgrace	0.47

Table 6: Explanation for negative comment with high negative score

4.4 Negative comment with low negative score

I analyze the comment

> *The game has been delayed from October to November. Thanks again, Covid. What would be of our lives without you?*

to show evidence of negative comments that were not interpreted as such by VADER. This comment received a score of 0.08 because, as most NLP models, the sarcastic tone went undetected. As shown in Table 7, the LIME explanation only yields the word *delayed*. In this example, VADER failed to contextualize the phrase *thanks again, Covid* due to the comment's sarcastic tone. The only driver for the score assigned to this observation is due to the word *delayed*.

Word	Importance
delayed	0.08

Table 7: Explanation for negative comment with low negative score

5 Conclusion

Regardless of whether a prediction is accurate or not, the ability to understand the logic behind the model's decision making process is important. The choice between accuracy and interpretability is commonly perceived as an inescapable trade-off. LIME proposes a partial solution to this dilemma by making local approximations to explain the predictions made by any kind of supervised black-box model.

One of the main benefits of using LIME is that the explanations yield insights from both accurate and inaccurate predictions. For instance, the explanations from the examples shown in the previous section suggests that VADER is unable to detect sarcasm or realize that the word *battlefield* is not negative in the context of this article. Likewise, LIME can help the modeller understand the reasons behind a seemingly accurate prediction. For example, we would not trust a model that predicts that an image contains a wolf simply because there is snow in the background.

The method reviewed in this article shows that LIME is a viable option that allows us to partially overcome the accuracy-interpretability trade-off by disentangling the logic behind a model's decision making process at a given point in its domain. Although the method is not able to derive a global interpretable

model, local explanations may point to globally-valid insights in the model's logic.

As AI continues to integrate into our lives, the matter of interpretability will continue to gain strength and methods such as LIME are an important first step to guaranteeing that even complex models remain interpretable to some extent.

References

Battlefield (2021). Battlefield 2042 Official Reveal Trailer (ft. 2WEI). *Youtube*.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

Hutto, C. and Gilbert, E. (2014). VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1):216–225.

Kwon, K. H. and Gruzd, A. (2017). Is offensive commenting contagious online? Examining public vs interpersonal swearing in response to Donald Trump 's YouTube campaign videos. *Internet Research*, 27(4):991–1010.

Lui, M. and Baldwin, T. (2011). Cross-domain Feature Selection for Language Identification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 553–561, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Molnar, C. (2022). *Interpretable Machine Learning*. Independently published, 2nd edition.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016a). Local Interpretable Model-Agnostic Explanations (LIME): An Introduction. *O'Reilly Media*.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016b). 'Why Should I Trust You?': Explaining the Predictions of Any Classifier. *22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 97–101.

Soberón, A. (2022). LIME for NLP.

Tassi, P. (2021). "Battlefield 2042" Has Entered Steam's 10 Worst Reviewed Games Of All Time List. *Forbes*.

Van Rossum, G. and Drake, F. L. (2009). *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Wes McKinney (2010). Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56–61.