CausalNLP: A Practical Toolkit for Causal Inference with Text

Arun S. Maiya
Institute for Defense Analyses (IDA)
amaiya@ida.org

Abstract

Causal inference is the process of estimating the effect or impact of a treatment on an outcome with other covariates being treated as potential confounders (or mediators or suppressors) that may need to be controlled or balanced. The vast majority of existing methods and systems for causal inference assume that all variables under consideration are categorical or numerical (*e.g.,* gender, price, blood pressure, enrollment). In this paper, we present `CausalNLP`, a toolkit for inferring causality with observational data that includes **text** in addition to traditional numerical and categorical variables. `CausalNLP` employs the use of *meta-learners* for treatment effect estimation and supports using raw text and its linguistic properties as both a treatment and a "controlled-for" variable (*e.g.,* confounder). The library is open-source and available at `https://github.com/amaiya/causalnlp`.

1 Introduction and Motivation

Many datasets and causal questions about them naturally include text data. For example:

- Surveys often include open-ended questions with responses in the form of unstructured text.
- Causal questions involving social media almost always have text associated with data points (*e.g.,* posts, tweets, profile summaries).
- News stories and blogs have associations with (and potential causal impacts on) beliefs and attitudes (*e.g.,* Romer and Jamieson (2021)).

Text and its linguistic properties (*e.g.,* topic, sentiment, emotion, readability, politeness, toxicity) can represent both a treatment variable or covariates for which adjustments are needed for unbiased causal estimates.[1] Here, we describe some examples to better illustrate this.

Example 1: Text as Treatment or Outcome. How does the readability of an email soliciting donations affect whether or not a donation is made? Or, what is the impact of politeness in an email on receiving fast responses from customer service (Pryzant et al., 2021)? Both of these questions involve text as a potential **cause** of some outcome. Text can also be treated as an **outcome** (*e.g.,* emotion of a response during a debate).

Example 2: Controlling for Text. What is the direct impact of including a theorem in a research paper on the paper's acceptance (Veitch et al., 2020)? Papers on certain topics may have more (or less) theorems and these topics may also impact a paper's acceptance. Thus, the text is a *confounding* variable that must be controlled to estimate the direct impact of inclusion of theorems (MacKinnon et al., 2000). Text can also act as a *mediator* that must be controlled in a study (MacKinnon et al., 2000). For instance, does being male increase the probability of a post being liked or upvoted? Gender is simply a categorical variable. However, aside from gender, the text of the post (*e.g.,* topic, sentiment, emotion) will obviously also have an impact on whether a post is shared. Suppose women tend to write more positive posts or post about popular topics. We need to be able to estimate the direct impact of gender on whether a post is upvoted **independent** of the text.

[1] A treatment is an independent variable that potentially *causes* some outcome, response, or effect. (We will use the terms outcome, response, and treatment effect interchangeably when referring to the dependent variable of interest in this paper.) Confounders, mediators, and suppressors are non-treatment independent variables (covariates) that may bias causal estimates unless they are "controlled for." Confounders, mediators, and suppressors are conceptually different but statistically the same (MacKinnon et al., 2000).

Despite the prevalence of natural language text and its importance and impact on larger systems as illustrated above, there is surprisingly little work (relative to other areas) on how best to incorporate text when performing causal analyses. In fact, it is only recently that this problem was more formalized as a research question (*e.g.,* Pryzant et al. (2021); Veitch et al. (2020); Roberts et al. (2020)). As a result, there are few practical tools or libraries that can be leveraged in causal impact studies with observational data including a text component.

To this end, we present `CausalNLP`, the first practical toolkit for performing causal inference with text data. `CausalNLP` is open-source, free to use under a permissive Apache license, and available on GitHub at: `https://github.com/amaiya/causalnlp`.

Contributions. In this paper, we describe and demonstrate the ways in which `CausalNLP` estimates treatment effects on data that includes text fields. We first discuss *meta-learners*, the underlying mechanism by which causal effects are estimated in `CausalNLP`. We then show how support for text can easily be incorporated into any approach based on meta-learners. Surprisingly, we find that a simple approach based on meta-learners achieves statistically the same performance of the state-of-the-art CausalBERT (Veitch et al., 2020; Pryzant et al., 2021), while simultaneously offering more flexibility and requiring only a tiny fraction of the training time using nothing more than CPU-based commodity laptop hardware.

2 Causal Inference with Meta-learners

Meta-learners are an abundantly flexible class of techniques for causal inference on both experimental and observational data (Künzel et al., 2019; Chen et al., 2020).[2]. The basic idea behind *meta-learners* is to use underlying machine learning models (called *base learners*) to predict counterfactual outcome estimates from the covariates (*i.e.,* the auxiliary independent variables that are not the treatment). Armed with these predictions, the *meta-learner* can compute the treatment effect for each individual observation in a straightforward manner.

[2]Counterfactual outcomes are *potential* outcomes that can never be observed. For example, if you take a medication that later makes you feel better, what would have happened if you had *not* taken the medication is the counterfactual outcome.

2.1 Types of Meta-learners

Several different types of meta-learners have been proposed in the literature (Künzel et al., 2019). One of the simplest meta-learning methods is the *T-Learner*. A T-Learner uses two base learners. The first is a *control* model trained on only those observations that did not receive treatment, and the second is a *treatment* model trained on only those observations that *did* receive the treatment.

$$\mu_0(x) = \mathbb{E}[Y(0)|X = x]$$

$$\mu_1(x) = \mathbb{E}[Y(1)|X = x]$$

where each $Y_i(0)$ represents the outcome when observation i is *not* assigned the treatment, $Y_i(1)$ is the outcome when it is, and X represent the covariates. Neither the treatment model nor the control model use the treatment variable as a predictor in a T-Learner. With these two models, the meta-learner can produce two estimates for every observation: $\hat{\mu}_1(x)$ and $\hat{\mu}_0(x)$. The difference of these two estimates, $\hat{\tau}(x)$, is the T-Learner's estimate of the treatment effect: $\hat{\tau}(x) = \hat{\mu}_1(x) - \hat{\mu}_0(x)$.

Other types of meta-learners include the S-Learner, X-Learner, and R-Learner (Künzel et al., 2019). For instance, in contrast to the T-Learner, the S-Learner *does* include the treatment as a feature similar to other features in a single model:

$$\mu(x, w) = \mathbb{E}[Y^{obs}|X = x, W = w],$$

where W is the treatment and Y^{obs} are the observed outcomes. In the S-Learner, the treatment effect estimate is simply: $\hat{\tau}(x) = \hat{\mu}(x, 1) - \hat{\mu}(x, 0)$.

All aforementioned meta-learners are supported in `CausalNLP` and can handle confounding under the ignorability assumption. For a more detailed explanation of these meta-learners, please see Künzel et al. (2019).

2.2 Why Use Meta-learners?

There are three key advantages to using meta-learners when inferring causality with text data. The first is unconstrained model choice. Meta-learners can be used with virtually any underlying machine learning model as a base learner (*e.g.,* gradient boosting machines, neural networks, linear regression). `CausalNLP` currently uses the LightGBM implementation of gradient boosting decision trees as the default base learner (Ke et al., 2017), but this is configurable.

The second advantage is the fact that meta-learners naturally estimate *heterogeneous* treatment effects (Künzel et al., 2019). That is, meta-learners can estimate how causal impacts vary across observations – regardless of whether these datapoints represent individuals, text documents, or other entities. For instance, consider the aforementioned hypothetical study estimating the impact of gender on whether or not a social media post is upvoted. In addition to estimating the overall average treatment effect (ATE) of gender, *meta-learners* can identify those posts whose "upvotability" is *most* impacted by the gender of the person writing it. This, in turn, can potentially elicit hypotheses for future studies. In fact, using standard feature importance methods like permutation importance, we can easily identify the *words* or *linguistic properties* (*e.g.,* topics) associated with the magnitude of the causal effects (*i.e.,* the *change* in outcome given treatment rather than just the static outcome itself). Thus, an approach that supports both estimating heterogeneous treatment effects *and* supports incorporation of text-based covariates and treatments can greatly expand the set of interesting scientific questions that can be investigated - especially those in the social science domain.

Lastly, the third advantage is that it is very easy to use text features in combination with traditional numerical and categorical variables in any meta-learning approach to causal inference. This will be discussed in greater detail in the next section.

3 Using Text Data with Meta-learners

Like most traditional causal inference methods, meta-learners represent each observations as a fixed length vector representing the categorical or numerical attributes of each observation (*e.g.,* age, blood pressure, gender, occupation, price, ethnicity, months as customer). `CausalNLP` follows two simple approaches to leveraging text data with meta-learners: 1) *vectorization* of raw text and 2) *autocoding*. We will begin with text vectorization.

3.1 Using Raw Text Directly as Covariates

One approach to incorporating text as additional covariates in a meta-learner is to simply transform the raw text into a static fixed-length vector representation of the words (*i.e.,* a feature vector). Although `CausalNLP` supports several text vectorization schemes, the default represents text fields as simple TF-IDF vectors. Other encoding schemes include those based on topic modeling (Blei et al., 2003) and transformer models fine-tuned on a natural language inference task (Wolf et al., 2020).

There is a very important benefit to this simple approach: modeling flexibility. Such feature vector representations of text can be easily used in conjunction with any base machine learning model and any number of traditional numerical or categorical covariates representing supplemental attributes of observations (*e.g.,* age, occupation, publication source, etc.). These combined vectors can be used directly to train the base learners in any metalearner model. In practice, this can be valuable during construction of a causal model when trying to ensure bias is minimized by controlling for confounders.

A second benefit, as mentioned previously, is model interpretability. For instance, if certain words of a TF-IDF vector are predictive of larger treatment effects, these can be discovered through any feature importance scheme like permutation importance. `CausalNLP` supports interpretability of causal models in this way.

3.2 Deriving Linguistic Properties

In the previous section, we adjusted for text in a general sense without a specific linguistic property in mind. That is, we leave it to the base learners to discover which linguistic properties are of relevance to the problem through a sparse (TF-IDF) or dense (embedding) vector representation of the text. In some cases, analysts may be interested in *specific* linguistic properties like sentiment, emotion, or topic. For instance, the sentiment of a document may clearly be a potential confounder in a particular problem. In other cases, analysts may need to *create* a binary treatment variable from raw text to test a causal hypothesis. To test the causal effect of sentiment on sales, for example, analysts must derive a variable to identify which texts are positive and which are not. Such a variable might also be used as an outcome in a different study.

3.2.1 Coding Text Data

Historically, when a survey contains text fields (*i.e.,* open-ended questions that are answered with unstructured, free-form text), the typical approach to transforming the raw text into a form that can be used in causal or key driver analysis is to *code* the text fields (Vaughn and Turner, 2016). The term *coding* in survey analysis refers to the process of thematically grouping similar responses together, so that they the text can be analyzed as a conven-

Analyzer	Columns Created	Example Usage
Sentiment	positive, negative	`df = ac.code_sentiment(texts, df)`
Emotion	joy, anger, fear, sadness	`df = ac.code_emotion(texts, df)`
Topic	(user-specified topics)	`df = ac.code_custom_topics(texts, df, labels)`
User-Defined	(any linguistic property)	`df = ac.code_callable(texts, df, fn)`

Table 1: **Autocoder.** Examples of built-in text analyzers available in Autocoder. In the **Example Usage** column, `ac` is an Autocoder instance, `texts` is a list of raw texts as Unicode strings, and `df` is a *pandas* DataFrame containing the dataset under analysis. For the `code_callable` method, `fn` is any Python callable that takes text as input and returns a dictionary with desired column names as keys and numerical values (*e.g.,* probabilities, scores) as values. After invocation, `df` will contain newly-created columns containing the variables of interest.

tional categorical variable. Since (by today's standards) surveys typically contain a smaller number of observations, coding is often performed manually (Vaughn and Turner, 2016). In cases where it is done in a more automatic manner, the approaches are typically simple (*e.g.,* does the text contain a specific word pattern?).

For causal inference studies where text plays a more prominent role, these existing approaches may not be adequate. For instance, observed data may be collected from more voluminous sources like online reviews or news and social media instead of smaller surveys. Due to the proliferation of text data in many domains of interest, manual coding or simplistic approaches may no longer be feasible. Moreover, in text-based studies, a linguistic property of text may itself be the *treatment* or *outcome* under study and must be derived in an accurate way (*e.g.,* sentiment, emotion, topic).

3.2.2 The Autocoder in `CausalNLP`

For the reasons described above, `CausalNLP` includes what we refer to as the *Autocoder*. The Autocoder is a built-in suite of state-of-the-art text analyzers that transform raw text into either a binary treatment variable or additional numerical covariates for any causal inference study involving text. Table 1 shows examples of out-of-the-box text analyzers included in `CausalNLP`. Each are implemented using state-of-the-art transformer models (Wolf et al., 2020). For instance, the `code_custom_topics` method is implemented using **zero-shot text classification** where a transformer model fine-tuned on a natural language inference task is used to detect user-specified topics *without* training examples (Yin et al., 2019; Maiya, 2020). Here, we show a self-contained example for illustration purposes:

```
# Zero-Shot Topic Classification with Autocoder

# sample data
comments = ["What is your favorite sitcom?",
            "Can't wait to vote!"]
import pandas as df
df = pd.DataFrame({
    'OVER_18': ['yes', 'no'],
     'COMMENT' : comments,
      })

# autocode text in DataFrame
from causalnlp.autocoder import Autocoder
ac = Autocoder()
labels=['TV', 'POLITICS']
df = ac.code_custom_topics(df['COMMENT'].values,
                           df, labels)
```

After executing the Python code above, the *pandas* DataFrame, `df`, will contain two new columns (`television` and `politics`) containing predicted probabilities representing the degree to which the text pertains to each of these topics:

```
OVER_18   TV     POLITICS   COMMENT
yes       0.98   0.00       Favorite sitcom?
no        0.00   0.95       Can't wait to vote!
```

These new columns can either be used as "controlled-for" covariates in a causal analysis or binarized for use as a treatment (or outcome).

4 Experiments

In this section, we evaluate the ability of `CausalNLP` to recover causal effects from data that includes text. Evaluating such causal effect estimations is challenging because ground-truth treatment effects are typically unavailable. Existing works in this area address this problem by generating semi-synthetic datasets where the text is real and the outcomes are simulated (Veitch et al., 2020; Pryzant et al., 2021).

Datasets. We use a semi-simulated dataset of Amazon reviews similar to the one used by Pryzant et al. (2021). The task at hand involves estimating the causal effect of a positive review on whether or not a product is clicked. The true sentiment of the review is the treatment, as

Method	ATE Estimate	Δ from Oracle	Training Time
Oracle (ground truth)	15.88	0.0	-
Naive (no confounding adjustments)	7.72	8.16	-
S-Learner w/ LogisticRegression (ours)	14.26	**1.62**	0.86 sec. (CPU)
T-Learner w/ LogisticRegression (ours)	10.97	4.91	1 sec. (CPU)
T-Learner w/ LGBM, [num_leaves=100] (ours)	12.90	**2.98**	2 sec. (CPU)
T-Learner w/ LGBM, [num_leaves=31] (ours)	11.88	4.00	1 sec. (CPU)
X-Learner w/ LGBM, [num_leaves=500] (ours)	12.58	3.30	13 sec. (CPU)
R-Learner w/ LGBM, [num_leaves=31] (ours)	11.89	3.99	10 sec. (CPU)
CausalBERT (Veitch et al., 2020)	18.51	**2.63**	16 min. (GPU)
TEXTCAUSE (Pryzant et al., 2021)	10.03	5.85	24 min. (GPU)

Table 2: **Results.** Meta-learners in `CausalNLP` achieved statistically the same performance as CausalBERT, as measured by **Δ from Oracle** (lower is better), while training faster on only a CPU and offering more versatility. **Oracle** represents the ground truth ATE (average treatment effect). All values are means across random trials. Differences between CausalBERT and top meta-learner scores in **bold** (under **Δ from Oracle** column) were not statistically significant at $p = 0.05$, as determined by a one-way ANOVA test and Post Hoc Tukey HSD.

determined by the rating. Confounders include a categorical variable indicating the product type. We assume the rating is hidden and sentiment is inferred from text. Thus, text plays a role as both a treatment and a "controlled-for" variable. Both the dataset and the code to simulate the outcomes for the dataset are available here.[3] Aside from varying the random seeds for each trial, default settings were used when simulating outcomes.

Baselines. We compare the S-Learner, T-Learner, X-Learner, and R-Learner in `CausalNLP` with CausalBERT (Veitch et al., 2020) and TEXTCAUSE (Pryzant et al., 2021). (Note that TEXTCAUSE uses CausalBERT in adjusting for text.) The code for CausalBERT was obtained from here.[4] TEXTCAUSE is available here.[5] The code for `CausalNLP` is available on GitHub.[6] To create the binary treatment from the raw text (*i.e.,* positive sentiment of review), we use our aforementioned `CausalNLP` *Autocoder* for all methods except TEXTCAUSE, which uses its own algorithm. To adjust for the raw text in `CausalNLP`, we used TF-IDF vectorization.

Results. As shown in Table 2, meta-learners in `CausalNLP` performed statistically the same as CausalBERT while offering more versatility and requiring only a tiny fraction of the training time on a 2.6 GHz i7 CPU. This is despite giving CausalBERT and TEXTCAUSE the added advantage of a Titan V GPU. Differences among top scores in bold under the **Δ from Oracle** column were not statistically significant at $p = 0.05$, as determined by a one-way ANOVA and Post Hoc Tukey HSD. `CausalNLP` also offers more flexibility in practice by allowing any number of additional categorical or numerical variables as covariates. This better helps analysts to reduce sources of biased estimates by controlling for them.

Reproducibility. `CausalNLP` is a *low-code* library that enables causal analyses with minimal effort. As such, we include the full code to reproduce `CausalNLP` results from Table 2 here:[7]

```
# Low-Code Causal Inference with CausalNLP

# load semi-simulated dataset
import pandas as pd
df = pd.read_csv('data.tsv', sep='\t',
                 error_bad_lines=False)

# use Autocoder to create treatment from text
from causalnlp.autocoder import Autocoder
ac = Autocoder()
df = ac.code_sentiment(df['text'].values,
                       df,
                       batch_size=16)
df['T_ac'] = (df['positive'] > 0.5).astype('int')

# train S-Learner for causal-inference
from causalnlp.causalinference import ↩
    CausalInferenceModel
from sklearn.linear_model import ↩
    LogisticRegression
base_learner = LogisticRegression(
                        solver='liblinear',
                        penalty='l1',
                        fit_intercept=False)
cm = CausalInferenceModel(df,
                    metalearner_type='s-learner',
                    treatment_col='T_ac',
                    outcome_col='Y_sim',
                    text_col='text',
                    include_cols=['C_true'],
                    learner=base_learner)
cm.fit()
cm.estimate_ate() # calculate ATE
```

[3] `https://github.com/rpryzant/causal-text`

[4] `https://github.com/rpryzant/causal-bert-pytorch`

[5] `https://github.com/rpryzant/causal-text`

[6] Code for `CausalNLP` is available at: `https://github.com/amaiya/causalnlp`

[7] The T-Learner used the following as base learner: `LGBMClassifier(num_leaves= 100, min_child_weight= 100.0, colsample_bytree = 0.59, min_child_samples = 59, reg_alpha= 10, reg_lambda= 100, sub_sample=0.64)`

In the code above, `data.tsv` is a semi-simulated dataset created using this script.[8] When simulating outcomes, default settings of the simulation script were used (*e.g.,* higher levels of confounding).

5 Conditional Average Treatment Effects

As mentioned, one of the key advantages of meta-learners is their natural ability to estimate how causal impacts vary across observations (*i.e.,* heterogeneous treatment effects). This can be easily calculated and inspected in `CausalNLP`. For instance, in the Amazon review dataset considered in our experiments, we can calculate the treatment effect for only those reviews that contain the word "toddler" (the conditional average treatment effect or CATE) as follows:

```
# CATE for reviews containing the word "toddler"
series = df['text']
cm.estimate_ate(series.str.contains('toddler'))
```

Finally, despite the "NLP" in `CausalNLP`, the library can be used to easily perform causal inference on any dataset regardless of whether text is included. As a general-purpose API for causal analysis, `CausalNLP` serves to democratize causal inference by making it much easier to apply and use. To demonstrate this, we include a brief example of using `CausalNLP` to estimate the CATE of a foreclosure or short sale on the sale price of houses greater than 2000 square feet using a publicly available Kaggle dataset.[9]:

```
# Example: Traditional Causal Inference

# prepare data
import pandas as pd
df = pd.read_csv('housing.csv')
fn = lambda x: 1 if x == 'Abnorml' else 0
df['treatment'] = df['SaleCondition'].apply(fn)

# estimate effect of foreclosures and short sales
# on sale price of homes > 2000 sq.
from causalnlp.causalinference import ↩
    CausalInferenceModel
cm = CausalInferenceModel(df,
                    metalearner_type='t-learner',
                    treatment_col='treatment',
                    outcome_col='SalePrice',
                    ignore_cols=['Id', '↩
                        SaleCondition'])
cm.fit()
cm.estimate_ate(df['GrLivArea']>2000])
```

Numerical and categorical fields are automatically preprocessed appropriately. For instance, categorical fields are auto-detected and one-hot-encoded, and missing values are imputed automatically.

6 Related Work

Causal inference with text is a relatively nascent research area, but there has been a recent surge of work in the area. A number of works focus on deriving or discovering **treatments** from text (Pryzant et al., 2021; Wood-Doughty et al., 2018; Fong and Grimmer, 2016; Igaba et al., 2020). For instance, Wood-Doughty et al. (2018) examined different errors associated with predicting treatment labels with classifiers. Fong and Grimmer (2016) proposed ways to discover and derive treatments from text corpora. Pryzant et al. (2021) proved bounds on biases arising from estimated treatments in addition to proposing a method to handle text-based treatments. There are also a few works that examine text as an **outcome** (*e.g.,* Sridhar and Getoor (2019)). Finally, other works focus more on adjusting for text as **confounders** or **mediators** (Veitch et al., 2020; Mozer et al., 2020; Roberts et al., 2020). For instance, Mozer et al. (2020) and Roberts et al. (2020) explored the use of text matching to control for text in a causal analysis. More recently, Veitch et al. (2020) have proposed CausalBERT to adjust for text in causal inference. For a comprehensive survey of approaches to control for text, the reader may refer to Keith et al. (2020). `CausalNLP` is the first library to unify these tasks into a practical and versatile toolkit.

7 Conclusion

In this paper, we presented `CausalNLP`, a Python library for causal inference with text from observational data. Based on meta-learners, `CausalNLP` is a versatile toolkit supporting inclusion of text and its linguistic properties as treatments, outcomes, confounders, or mediators in a causal inference study. The intersection of causality and NLP is a newer research area with a rapidly changing landscape ripe with opportunities for future work. Examples include better explainability of causal inferences and characterizing and better understanding meta-learner performance with respect to text. For instance, while the S-Learner performed well, we did observe that other meta-learners (*e.g.,* the T-Learner) were better able to identify heterogeneous treatment effects in terms of how causal impacts varied across observations.

[8] `https://github.com/rpryzant/causal-text/blob/main/src/simulation.py`

[9] This housing dataset is available on Kaggle here: `https://www.kaggle.com/c/house-prices-advanced-regression-techniques/`.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Huigang Chen, Totte Harinen, Jeong-Yoon Lee, Mike Yung, and Zhenyu Zhao. 2020. Causalml: Python package for causal machine learning. *arXiv preprint arXiv:2002.11631*.

Christian Fong and Justin Grimmer. 2016. Discovery of treatments from text corpora. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1600–1609, Berlin, Germany. Association for Computational Linguistics.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Katherine A. Keith, David Jensen, and Brendan O'Connor. 2020. Text and causal inference: A review of using text to remove confounding from causal estimates. *arXiv preprint arXiv:2005.00649*.

Sören R. Künzel, Jasjeet S. Sekhon, Peter J. Bickel, and Bin Yu. 2019. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the National Academy of Sciences*, 116(10):4156–4165.

Andres lgaba, David Ardia, Kevin Bluteau, Samuel Borms, and Kris Boudt. 2020. Econometrics meets sentiment: An overview of methodology and applications. *Journal of Economic Surveys*, 34.

David P. MacKinnon, Jennifer L. Krull, and Chondra M. Lockwood. 2000. Equivalence of the mediation, confounding and suppression effect. *Prevention science*.

Arun S. Maiya. 2020. ktrain: A low-code library for augmented machine learning. *arXiv preprint arXiv:2004.10703*.

Reagan Mozer, Luke Miratrix, Aaron Russell Kaufman, and L Jason Anastasopoulos. 2020. Matching with text data: An experimental evaluation of methods for matching documents and of measuring match quality. *Political Analysis*.

Reid Pryzant, Dallas Card, Dan Jurafsky, Victor Veitch, and Dhanya Sridhar. 2021. Causal effects of linguistic properties. *arXiv preprint arXiv:2010.12919*.

Margaret E Roberts, Brandon M Stewart, and Richard A Nielsen. 2020. Adjusting for confounding with text matching. *American Journal of Political Science (forthcoming)*.

Daniel Romer and Kathleen Hall Jamieson. 2021. Patterns of media use, strength of belief in covid-19 conspiracy theories, and the prevention of covid-19 from march to july 2020 in the united states: Survey study. *J Med Internet Res*, 23(4):e25215.

Dhanya Sridhar and Lise Getoor. 2019. Estimating causal effects of tone in online debates. *arXiv preprint arXiv:1906.04177*.

Porcia Vaughn Vaughn and Charlie Turner. 2016. Decoding via coding: Analyzing qualitative text data through thematic coding and survey methodologies. *Journal of Library Administration*, 56(1):41–51.

Victor Veitch, Dhanya Sridhar, and David M. Blei. 2020. Adapting text embeddings for causal inference. *arXiv preprint arXiv:1905.12741*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zach Wood-Doughty, Ilya Shpitser, and Mark Dredze. 2018. Challenges of using text classifiers for causal inference. *arXiv preprint arXiv:1810.00956*.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. *arXiv preprint arXiv:1909.00161*.