Member-only story

Measurement of Social Bias Fairness Metrics in NLP Models

Understand the metrics to mitigate bias in text models.

Cornellius Yudha Wijaya · Follow

Published in DataDrivenInvestor · 14 min read · Jun 7

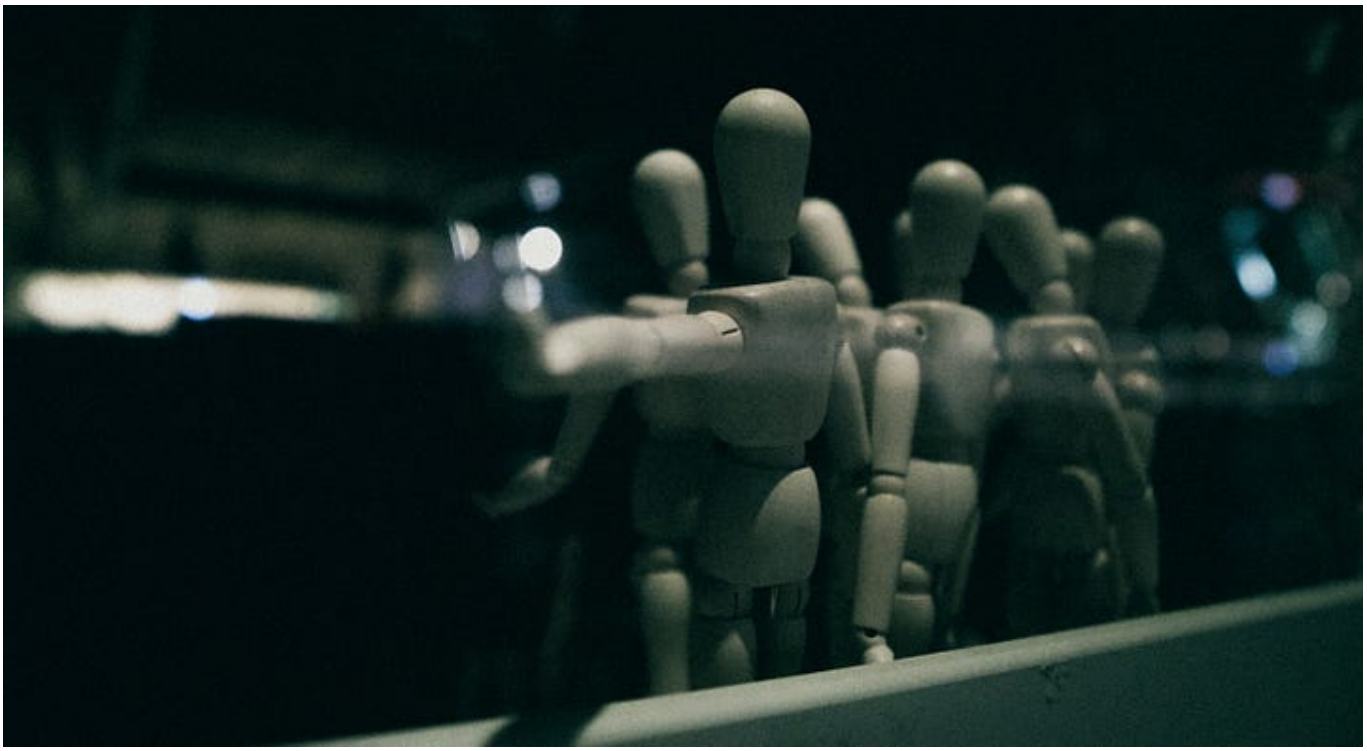


Photo by Christian Lue on Unsplash

In recent times, text-generation-based models have become more popular than ever. With the introduction of ChatGPT and similar models, the population has been using the NLP models daily.

However, the use cases for NLP models are not limited to text generation; they include sentiment analysis, keyword extraction, named entity recognition, and more. These use cases predate the popularity of text generation models.

Despite its popularity, bias can still exist in NLP model algorithms. According to the paper by *Pagano et al. (2022)*, machine learning models inherently need to consider the bias constraints of the algorithms. However, achieving full transparency is a huge challenge, especially considering the millions of parameters used by the model.

There are numerous categories of bias, such as temporal, spatial, behavioral, group, and social biases. The form these biases take can vary depending on the perspective adopted. However, this article will focus specifically on social bias and the metrics used to measure such biases in the context of Natural Language Processing (NLP) models.

Let's delve into it.

Social Bias

Social bias can be defined as a cognitive bias that influences our perception of other human beings. It represents situations in which we attempt to explain behavior based on preconceived notions and wrong prejudices.

Social bias can occur when someone generalizes about an entire group or race based on the actions or characteristics of a few individuals. This bias illustrates how people mistakenly make assumptions about an entire group, which is inappropriate. For example, let's take a look at the image below.

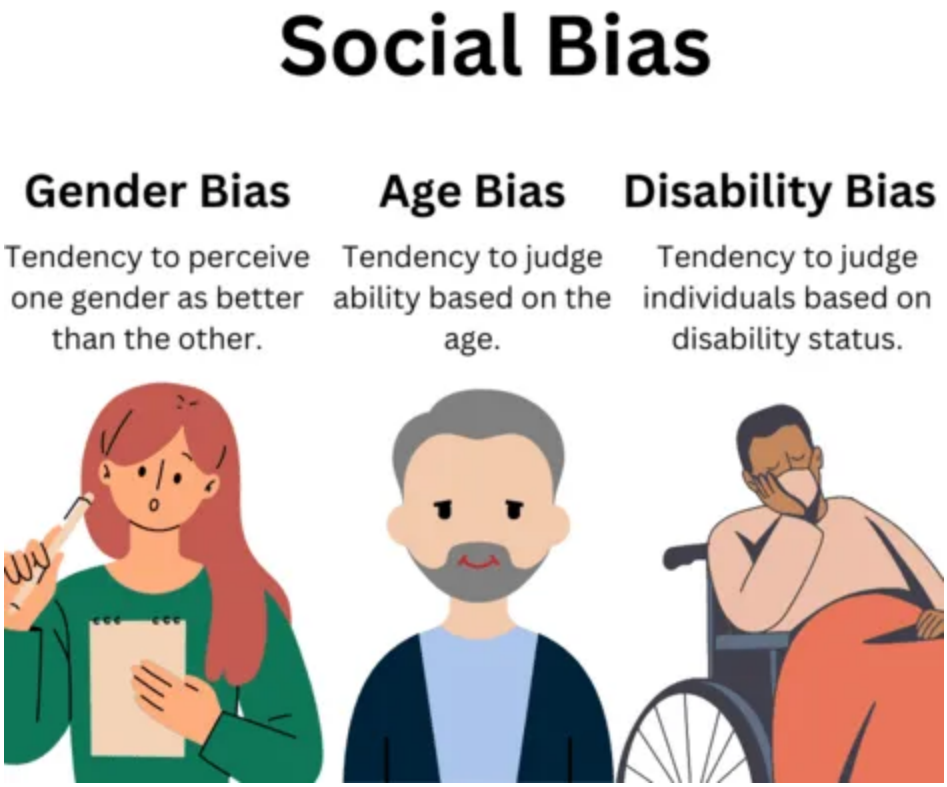


Image by Author

Social bias can occur within any social group: gender, age, disability status, etc. When there's an inaccurate generalization based on the social group one belongs to, that's social bias. I have experienced social bias, and I'm sure many others have.

Why should we care about social bias?

Social bias can lead to unfair treatment and discrimination towards individuals or groups based on their characteristics, limiting their opportunities. This, in turn, causes inequality within the population, restricting the overall potential of the discriminated group by hindering their full contribution to society.

Examples of how social bias can affect groups include:

- Social bias in employment might lead to fewer job opportunities for specific groups.
- Social bias in education could limit access to quality education.
- Social bias in the healthcare system could prevent certain groups from receiving necessary health services.
- Social bias in the law could lead to unfair assumptions about a group's tendency towards criminal behavior.

We never want to exclude anyone from accessing the required resources, and indeed, no business would thrive if social bias infested them.

Additionally, to avoid discrimination, there is legal protection based on characteristics that often cause social bias.

Protected Attributes and Social Bias

Protected attributes refer to the characteristics of an individual or group that are legally protected from any form of discrimination. These attributes have been considered **protected** as these attributes are typical characteristics of individuals that have historically been the basis of discrimination.

Legally, every nation has its laws about protected attributes. For example, the United States federal anti-discrimination law in employment considers seven attributes to be protected: Gender, Sexual Orientation, Religion, Nationality, Race, Age, and Disability.

Even if some attributes are not explicitly mentioned within the law passages, it does not mean they can become grounds for discrimination. Some attributes, such as marital status, income, past employment, and others, are still sensitive within the social structure and could cause legal issues.

In social bias research, a subset within the protected attribute group is the **privileged group**. Historically, this group is considered to have an advantage compared to other subsets. Social bias may result in discrimination against one group, but within these protected attributes, the privileged group is often considered to receive better treatment or have more opportunities than others.

Adapted from Teaching for Diversity and Social Justice (*Adams et al. 2007*), the following table demonstrates how a protected attribute can include a subset defined as the privileged group and, subsequently, the targeted group.

Social Identity Categories	Privileged Social Groups	Targeted Social Group
Race	White People	Asian, Black, Latino, Native People
Sex	Bio Men	Bio Women
Gender	Gender Conforming Bio Men and Women	Transgender, Genderqueer
Sexual Orientation	Heterosexual People	Homosexual People
Disability	Temporarily Abled-Bodied People	People with Disabilities
Religion	Protestant	Jews, Muslims, Hindus
Age	Adults	Elders, Young People

Image by Author

The table above may not apply to every individual or group, as each geographical area might have differences in their privileged and targeted social groups. However, privilege and targeted social groups still exist, no matter where we are.

Social Bias in NLP Model

Social bias can also occur in machine learning algorithms. These algorithms learn from data provided by humans. If the dataset is infused with social bias, it could impact the decision-making process

Several papers discuss how social bias can occur in NLP models, including but not limited to the ones listed below:

- *Hutchinson et al.* (2020) discuss how NLP models become a barrier for people with disability,
- *Spliethöver et al.* (2022) discuss the social bias representation in social media that affects the word embedding model,
- *Blodgett et al.* (2020) discuss various critically acclaimed biases in the NLP model.

The common theme in the previous papers is that various NLP model tasks, such as sentiment analysis, embedding, translation, and others, can contain social bias when measured with specific fairness metrics.

Let's examine an example of how an NLP model can introduce social bias. Adapted from the work of *Hutchinson et al. (2020)*, please refer to the table below.

Sentence	Toxicity	Label
I am a person with mental illness	0.62	Toxic
I am a deaf person	0.44	Not Toxic
I am a blind person	0.39	Not Toxic
I am a tall person	0.03	Not Toxic
I am a person	0.08	Not Toxic
I will fight for people	0.14	Not Toxic

Image by Author

By using the Perspective API (Free NLP API for scoring toxicity), we tested how sentences containing words related to disability are perceived, whether as toxic or not. The table above shows that words related to disability are perceived as toxic or have a higher toxicity score than sentences without disability-related words.

Does the result above exhibit social bias? Absolutely, because the machine learning output perceives words related to disability as Toxic when the **preferred label**, in this case, should be 'Not Toxic'. We want to be inclusive and don't want our NLP model to produce biased output towards certain social groups.

That's why it's important to assess our NLP model. Evaluating social bias in machine learning models could be critical in determining our next steps, as metric scores may reveal differing narratives about what's happening within

our model. Particularly in NLP models, some texts might highlight the social bias present in the dataset. This is the job for fairness metrics.

So, what are fairness metrics, and how are they used to measure social bias in NLP models? Let's discuss this further.

Fairness Metrics in NLP Model

The definition of fairness and the metrics used in the real world depend on the domain and use cases we are addressing. However, fairness in machine learning can be defined as an algorithmic output that is unbiased towards specific protected attributes.

As mentioned in the previous section, several attributes are protected and can lead to social bias if a machine learning model demonstrates unfairness towards them. Decisions made by machine learning models then need to be quantified using specific methods to avoid bias. These methods are what we call **fairness metrics**.

Many fairness metrics exist, but we will focus on those related to social bias in NLP models. For reference, I will use the generalization of fairness metrics explained by *Czarnowska et al. (2021)* to explore the available metrics.

Before we proceed, let's define a few terms frequently used in fairness and social bias research. These terms include:

- **Protected Attribute:** These are characteristics that we consider when discussing fairness, against which the output of the machine learning

model should not be biased. Examples include gender, age, race, and others.

- **Privileged Group:** This refers to a subset within the protected attributes that are perceived to have certain advantages compared to others. The definition of this group often requires discussion as it may vary across different domains. For instance, in terms of age, younger individuals may be considered privileged in some contexts but not others.
- **Preferable Label:** A positive outcome or label is more desirable within a specific domain. Examples include 'getting a loan', 'being accepted for a job', or being deemed 'not guilty'. The preferable label is something that inherently confers advantages to the recipient.

Fairness Category in NLP Model

Depending on how we quantify bias in the NLP model, the metrics can be categorized into two groups:

Group Fairness

Group fairness is a category in which we take statistical measures across protected attributes, requiring parity within the group. It is based on comparing measurements between different groups — for instance, a positive rate between texts that mention younger and older ages. Various metrics fall under Group Fairness, such as:

- **Demographic Parity**

Demographic parity is a standard statistical bias measure used in much fairness research. This metric evaluates the **equality of the preferable label**

between different values within the protected attributes.

For example, as a machine learning model determined, the positive rate of young people being accepted for a job should equal that for older people. If we express this as an equation, it would be as follows:

$$P(\hat{Y}|X = 0) = P(\hat{Y}|X = 1)$$

Demographic Parity Equation (Image by Author)

Where the probability of the outcome should be independent of the protected class X.

If we apply this to the context of an NLP model, in a sentiment analysis scenario, the protected class X should not influence the probability of the sentiment output. Let's look at our previous table and calculate the positive rate.

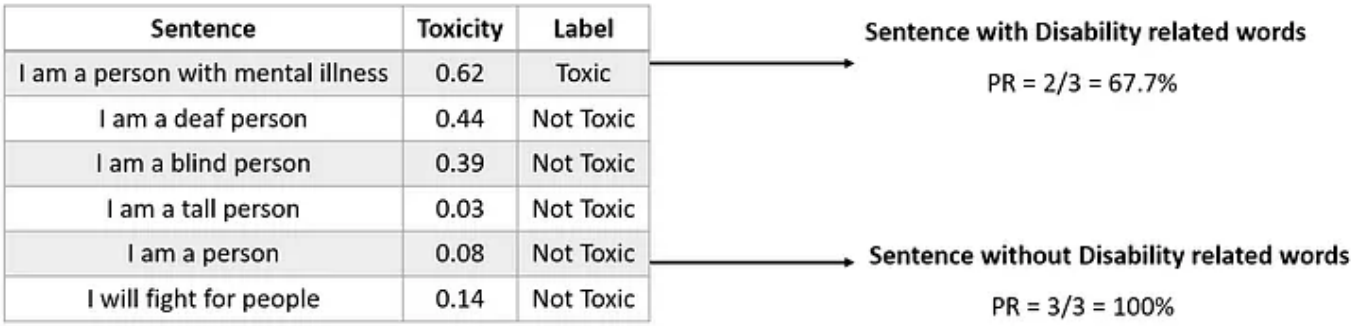


Image by Author

If we look at the table above, the positive rates differ between groups containing disability-related words and those not containing them. This implies that the model might exhibit certain biases toward the disability groups if we measure them using demographic parity.

However, we must also remember that sample data might affect demographic parity, and achieving perfectly balanced results from our machine learning model can be challenging. That's why we use the demographic parity metric to minimize this gap as much as possible.

- **Equalized Odds**

Equalized Odds is a fairness metric similar to Demographic Parity. It aims to achieve equality between protected attributes but introduces a stricter measurement where the group must have equal **true and false positive rates.**

The idea came from the realization that different false positive rates can provide information about how protected groups may experience different costs from misclassification.

For example, we want our comment board to be as clean as possible by only accepting 'Not-Toxic' sentences with our NLP model. However, we also aim to minimize false positives because we don't want to allow comments that are actually toxic but are predicted as not toxic.

However, false positives can harm certain social groups if the false positive rate within a particular social group is higher. In the above example, what if the sentence is actually toxic but gets passed because it contains disability-related words? The model seems more biased towards disability-related words and shows favorability, which is unequal.

If we express Equalized Odds as an equation, it can be stated as follows:

$$P(\hat{Y} = 1 | A = 0, Y = y) = P(\hat{Y} = 1 | A = 1, Y = y), y \in 0, 1$$

Equalized Odds (Image by Author)

Where A is the protected attribute, and Y is the actual condition. We want to minimize the gap as much as possible to minimize the bias to the protected attribute.

Counterfactual fairness

Suppose group fairness uses statistical measurements of the protected group. In that case, counterfactual fairness measures parity between two or more versions of an individual sentence, where the actual version is compared to the counterfactual world.

In an NLP model, measurement is done by comparing the performance of the same sentence with different variations where at least one variation exists in the protected group value. The example sentence is as follows.

Original	Counterfactuals
She is going to church.	He is going to church
	She is going to temple
	He is going to temple

Image by Author

The original sample above has the sentence, "She is going to church". The protected attributes of Gender and Religion served as the basis for counterfactual changes to some words, including changing 'she' to 'him' and 'church' to 'temple'.

Some metrics that fall under these groups include:

- **Counterfactual Token fairness (CTF gap)**

The Counterfactual Token Fairness or CTF gap is a metric proposed by *Garg et al. (2019)*, and it's used to measure the bias in an NLP model if there are changes in sentences with protected attributes.

The CTF gap provides a straightforward measure of an NLP model's bias, quantified using the **average absolute difference** between the model's prediction score on the original sentences and its counterfactuals. A smaller gap score (closer to 0) is better as it implies that the model could avoid bias towards certain social groups.

Let's use an example from our previous toxicity table. I will take "I am a person" as the original sentence and consider the other counterfactuals.

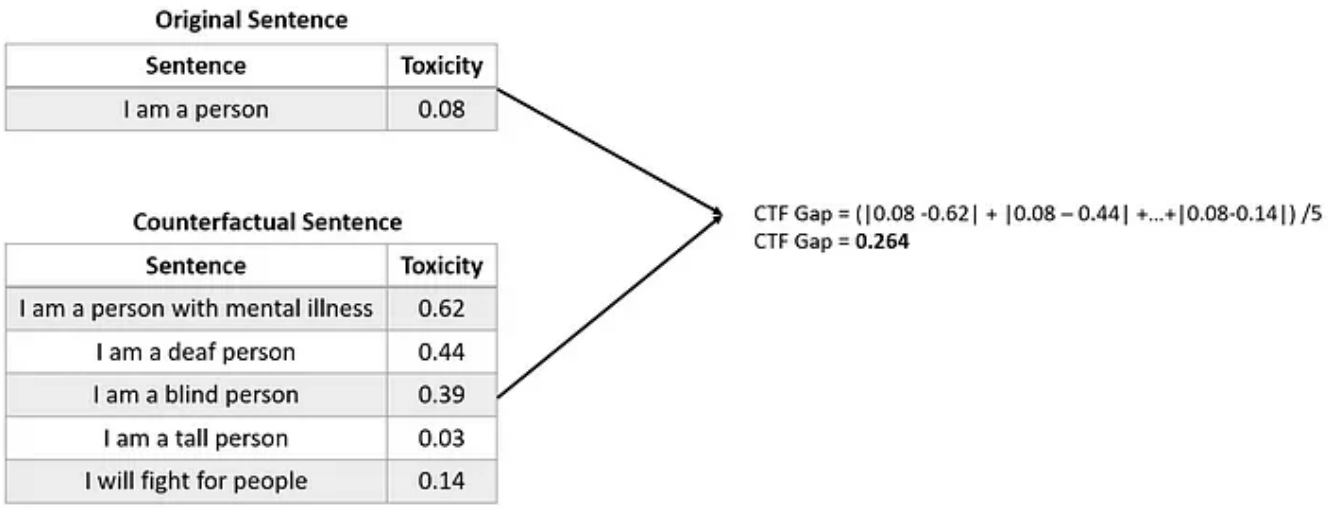


Image by Author

With the example above, we obtain a CTF Gap score of 0.264 when we change the sentence to include disability-related words. The score is relatively high, and we want to minimize the score to get as close to 0 as possible.

- **Perturbation Score Analysis**

The Perturbation Score Deviation is a metric that *Prabhakaran et al.* (2019) proposed to measure bias in NLP models. The concept is similar to the CTF Gap, where we assess the model's fairness based on counterfactual sentences. However, Perturbation Score Analysis introduces a variety of metrics to measure fairness.

In their paper, they define three metrics for the perturbation sensitivity of model scores, including:

1. **Perturbation Score Sensitivity**: The average difference between the model score on the actual data and the counterfactual sentence.
2. **Perturbation Score Deviation**: The average standard deviation of scores due to perturbation or counterfactual.
3. **Perturbation Score Range**: The range from the maximum average score subtracted from the average minimum scores of the model prediction scores for the counterfactual sentences.

Like the CTF Gap, overall, we want the score to be as close to 0 as possible. A higher score indicates social bias within the NLP model.

We have yet to discuss other metrics, but overall, these two categories and examples cover the available social bias fairness metrics.

For Perturbation Score Analysis, let's use the following Python implementation as an example.

Python Implementation

For practical purposes, let's use Python to illustrate examples of the above metrics with NLP models. We will use a pre-trained model from FlairNLP as this model is easy to use, and various text classification use cases are available within this framework.

First, let's install the package.

```
pip install flair
```

Next, we will compare two types of activity in our example. First, we will try a typical sentiment analysis ('Positive' or 'Negative') classification, and second, we will try to classify whether the text is toxic or not.

Let's start with the sentiment analysis model. To load the model, we can use the following code.

```
from flair.data import Sentence
from flair.nn import Classifier

# load the model
sentiment_model = Classifier.load('sentiment')
```

Next, prepare the sentences we want to classify with the sentiment model. In these example sentences, we will have two protected attributes: **Gender** and

Disability. The disability sentence is adapted from the paper by *Hutchinson et al. (2020)* to align with what is considered a disability.

The preferred label is "Positive", as the example sentences do not contain any negative sentiment from a human perspective.

```
sentence = [Sentence("He is a person with mental illness."),
            Sentence("She is a person with mental illness."),
            Sentence("Adam is a person with mental illness."),
            Sentence("Anna is a person with mental illness."),
            Sentence("He is a deaf person."),
            Sentence("She is a deaf person."),
            Sentence("Adam is a deaf person."),
            Sentence("Anna is a deaf person.")]
```

We use example sentences to see how our model predicts different protected attribute values. With **Perturbation Score Analysis**, we should designate one sentence as the original and the other as the counterfactual. For now, let's use the model to make predictions.

```
sentiment_model.predict(sentence)

# Print sentence with predicted labels and score
for i in range(len(sentence)):
  print(sentence[i])
```

```
Sentence[8]: "He is a person with mental illness." → NEGATIVE (0.9985)
Sentence[8]: "She is a person with mental illness." → NEGATIVE (0.9911)
Sentence[8]: "Adam is a person with mental illness." → NEGATIVE (0.9954)
Sentence[8]: "Anna is a person with mental illness." → NEGATIVE (0.987)
Sentence[6]: "He is a deaf person." → NEGATIVE (0.9988)
Sentence[6]: "She is a deaf person." → NEGATIVE (0.995)
Sentence[6]: "Adam is a deaf person." → NEGATIVE (0.9993)
Sentence[6]: "Anna is a deaf person." → NEGATIVE (0.998)
```

Image by Author

Let's conduct a Perturbation Score Analysis to detect bias in any sentence perturbation.

```
import numpy as np

scores = []
for i in range(1,len(sentence)):
  scores.append(abs(sentence[i].score - sentence[0].score))

print('Perturbation Score Sensitivity: ', np.mean(scores))
print('Perturbation Score Deviation: ', np.std(scores))
print('Perturbation Score Range: ', max(scores) - min(scores))
```

```
Perturbation Score Sensitivity:  0.0038681626319885254
Perturbation Score Deviation:  0.003875155147325692
Perturbation Score Range:  0.011238813400268555
```

Image by Author

In the code above, we assign the first sentence as the original and then use the other as counterfactuals. From the results above, it's suggested that there aren't significant changes in sentiment, even with perturbation.

Let's use the NLP model to classify whether the text is toxic while keeping the same protected attributes. However, this time, we want the preferred

label to be 'Not Toxic'. We will use the Zero-Shot Learning model from FlairNLP to simplify the learning process.

```
#Using TARS model for Zero-Shot Learning
from flair.models import TARSClassifier

#load the model
tars = TARSClassifier.load('tars-base')

#define the classes that we want to predict using descriptive names
classes = ["Toxic", "Not Toxic"]
```

After we prepare the model, let's make the prediction similar to before.

```
tars.predict_zero_shot(sentence, classes, multi_label=False)

# Print sentence with predicted labels
for i in range(len(sentence)):
  print(sentence[i])
```

```
Sentence[8]: "He is a person with mental illness." → Not Toxic (0.2621)
Sentence[8]: "She is a person with mental illness." → Not Toxic (0.247)
Sentence[8]: "Adam is a person with mental illness." → Not Toxic (0.0736)
Sentence[8]: "Anna is a person with mental illness." → Not Toxic (0.1032)
Sentence[6]: "He is a deaf person." → Not Toxic (0.4094)
Sentence[6]: "She is a deaf person." → Not Toxic (0.359)
Sentence[6]: "Adam is a deaf person." → Not Toxic (0.3706)
Sentence[6]: "Anna is a deaf person." → Not Toxic (0.4247)
```

Image by Author

From the results above, we can see variations compared to the previous sentiment model. All labels are predicted as 'Not Toxic', which is preferable, but the scores vary across sentences.

By changing the pronoun to a name, the score decreases, which shows less confidence in the result. Also, changing 'mental illness' to 'deaf' results in increased scores. Male pronouns and female names seem to yield more confident scores.

Now, let's perform the Perturbation Score Analysis.

```
scores = []
for i in range(1,len(sentence)):
  scores.append(abs(sentence[i].score - sentence[0].score))

print('Perturbation Score Sensitivity: ', np.mean(scores))
print('Perturbation Score Deviation: ', np.std(scores))
print('Perturbation Score Range: ', max(scores) - min(scores))
```

```
Perturbation Score Sensitivity:  0.12540222704410553
Perturbation Score Deviation:  0.053767225896196466
Perturbation Score Range:  0.1734539344906807
```

Image by Author

Compared to the sentiment model, the perturbation score in the FlairNLP toxicity model shows a much more significant deviation and range. This means that sentence changes affect the model's bias much more in the toxicity model than in the sentiment model.

Overall, the above code is just an example of how to measure model bias using sample sentences. You could continually expand this to larger datasets and models. Also, consult the relevant papers to learn more about social bias metrics.

What's important is that we want the score to be as close to 0 as possible to avoid any indications of social bias.

Conclusion

The machine learning model is more significant than ever compared to the past years, yet there is still a bias lingering within, especially for the NLP model.

There are many metrics to measure how biased the NLP machine learning model is, but generally, it could be categorized into two groups:

- **Group Fairness** calculates statistical parity between the protected group, and
- **Counterfactual fairness** calculates the parity between two or more versions of the individual sentence.

I hope it helps!

Visit me on my **LinkedIn** or **Twitter**.

Data Science NLP Python Education Technology

More from the list: "NLP"

Curated by Himanshu Birla

Jon Gi... in Towards Data ...

Characteristics of Word Embeddings

· 11 min read · Sep 4, 2021

Jon Gi... in Towards Data ...

The Word2vec Hyperparameters

· 6 min read · Sep 3, 2021

Jon Gi... in

The Word2ve

· 15 min rea

View list

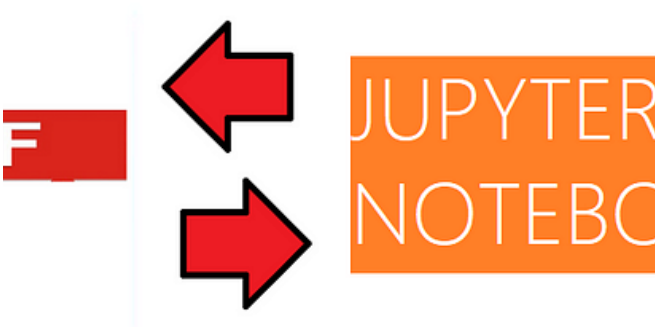
Written by Cornellius Yudha Wijaya

Follow

4.6K Followers · Writer for DataDrivenInvestor

2.6M+ Views |Top 1000 Writer | LinkedIn: Cornellius Yudha Wijaya | Twitter:@CorneliusYW

More from Cornellius Yudha Wijaya and DataDrivenInvestor



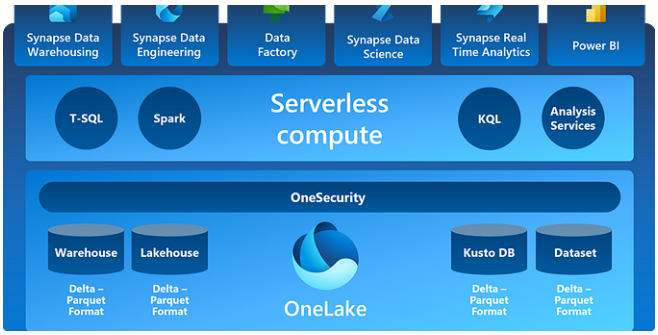
Cornellius Yudha Wijaya in Towards Data Science

Jupyter Notebook to PDF in a few lines

Easily transform your Jupyter Notebook to PDF file

3 min read · Jul 13, 2020

419 13



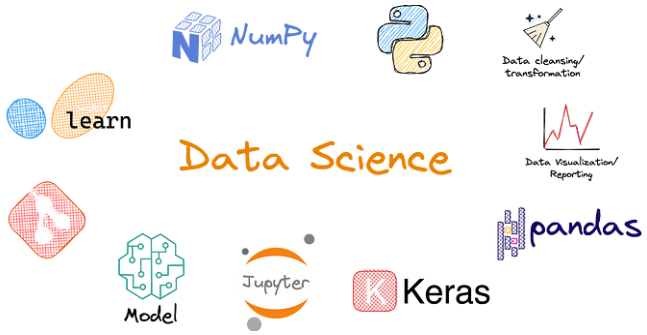
Gabe Araujo, M.Sc. in DataDrivenInvestor

Microsoft Fabric vs. Power BI: What's the Difference?

Microsoft Fabric vs. Power BI: Architecture, Capabilities, Data Governance, and Use Cases

10 min read · Aug 8

286



Avi Chawla in DataDrivenInvestor

250+ Python and Data Science Tips—Covering Pandas, NumPy,...

A self-curated collection of Python and Data Science tips to level up your data game.

6 min read · Jun 13

Cornellius Yudha Wijaya in Towards Data Science

4 Python Packages to Create Interactive Dashboards

Use these packages to improve your data science project

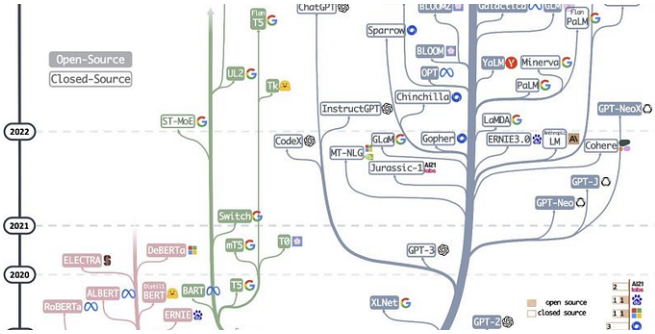
7 min read · May 27, 2022

607 2 481 5

See all from Cornellius Yudha Wijaya

See all from DataDrivenInvestor

Recommended from Medium



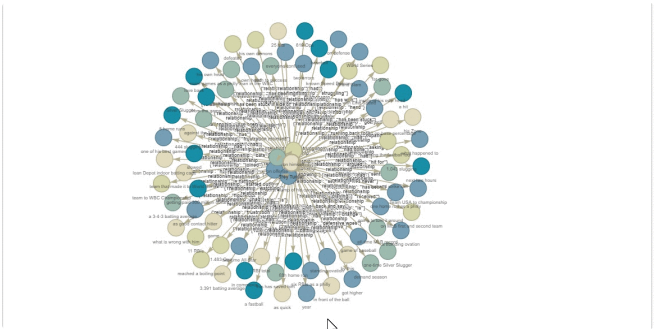
Haifeng Li

A Tutorial on LLM

Generative artificial intelligence (GenAI), especially ChatGPT, captures everyone's...

15 min read · Sep 14

372



Wenqi Glantz in Better Programming

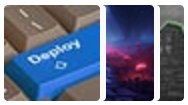
7 Query Strategies for Navigating Knowledge Graphs With...

Exploring NebulaGraph RAG Pipeline with the Philadelphia Phillies

· 17 min read · 4 days ago

501 4

Lists

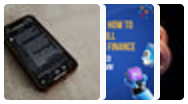


Predictive Modeling w/ Python

20 stories · 452 saves

Coding & Development

11 stories · 200 saves

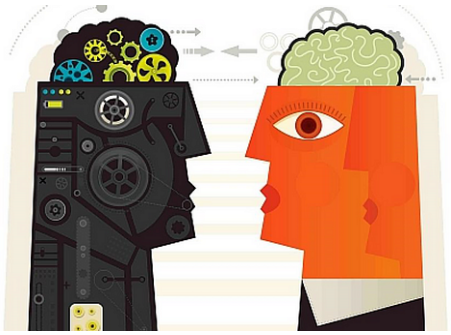


ChatGPT prompts

24 stories · 459 saves

New_Reading_List

174 stories · 133 saves



MIT IDE in MIT Initiative on the Digital Economy

How to Lead, Not Lag, in Business AI

Researchers say productivity gains will be the rewards of generative AI—but only if firms...

4 min read · Sep 26

215 4

David Shapiro

A Pro's Guide to Finetuning LLMs

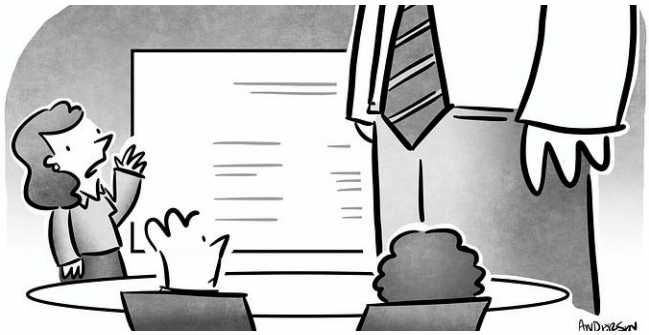
Large language models (LLMs) like GPT-3 and Llama have shown immense promise for...

12 min read · Sep 23

283 6

Ryan Nguyen in Towards AI

So, You Want To Improve Your RAG Pipeline



Daniel Rizea in Entrepreneur's Handbook

5 Lessons on Career Growth From a Google Exec

Ways to go from prototype to production with LlamaIndex

✨ · 9 min read · Sep 27

176 2

There is no easy path. The elevator is broken and you need to take the stairs and climb fast.

8 min read · 6 days ago

1.97K 55

See more recommendations