I am a researcher, and I work with AI every day. I can argue that everyone in my position is excited like a dog that stares at an ice cream cone.🤩

This is the reason:

Open AI’s Chat GPT is awesome.

For those who don't know what I'm talking about, Chat GPT is an artificial intelligence chatbot that can do, well, pretty much everything.
It can code, it can write articles, it can help you decorate your home, it can make up a recipe (I don’t recommend that if you are Italian), and the list goes on.

Yes, we can argue that it will cause ethical (and not only ethical) problems in the future. My mother is a high school teacher, and she is **terrified** about the idea that her students will use Chat GPT to cheat on their tests, and this is just one of the many examples of how things can go wrong with this incredibly powerful technology.

But again, the problem is the **use,** not the **product.** If we strictly talk about the **technological aspect** (that is, frankly, the one I am more interested in, as I am a certified nerd), it is **freaking amazing.**

Now, a lot of developers have used and tested this chatbot to try and develop their codes and their AI ideas, and of course, the usage of this chatbot **strictly depends on your background.** For example, if you are a web developer, you would ask ChatGPT to build a website using HTML. If you are a tester, you could ask ChatGPT to help you find that bug in that specific system.

In my specific case, I am a **researcher**. In particular, what I do for a living is build some **surrogate models** using AI. Let’s say that you want to conduct research on "A," but to do "A," you need **a lot of money**, **a lot of power,** and a **lot of computational time.** The idea behind this surrogate model is to replace it with a data-driven approach using artificial intelligence.

Now let’s completely change the subject for a moment.

Let’s say I am an **entrepreneur**, and I have tons of hotels all over the USA. Given a certain review of a given hotel, I want to know if that review is a good one or a bad one for that hotel. How do I do that? I have three options:

1. **I hire a person that reads millions of reviews and classify them** every day and I probably get arrested because it is clearly an abuse of human rights.
2. **I hire a person that, among other things reads *hundreds* of reviews and classifies them** every day. After **months,** I am able to build a dataset out of this. I **train a Machine Learning model** out of that dataset.
3. ***I automatically generate good and bad reviews.*** I **instantly** build a dataset out of this. I **train a Machine Learning model** out of that dataset.

As I value the time of my reader, let me skip the first option.

The second option is what you would do **before ChatGPT**. You can't know in advance if a review is bad or good, so if you want to build a dataset out of this, you need to hire people and wait until the dataset is ready.

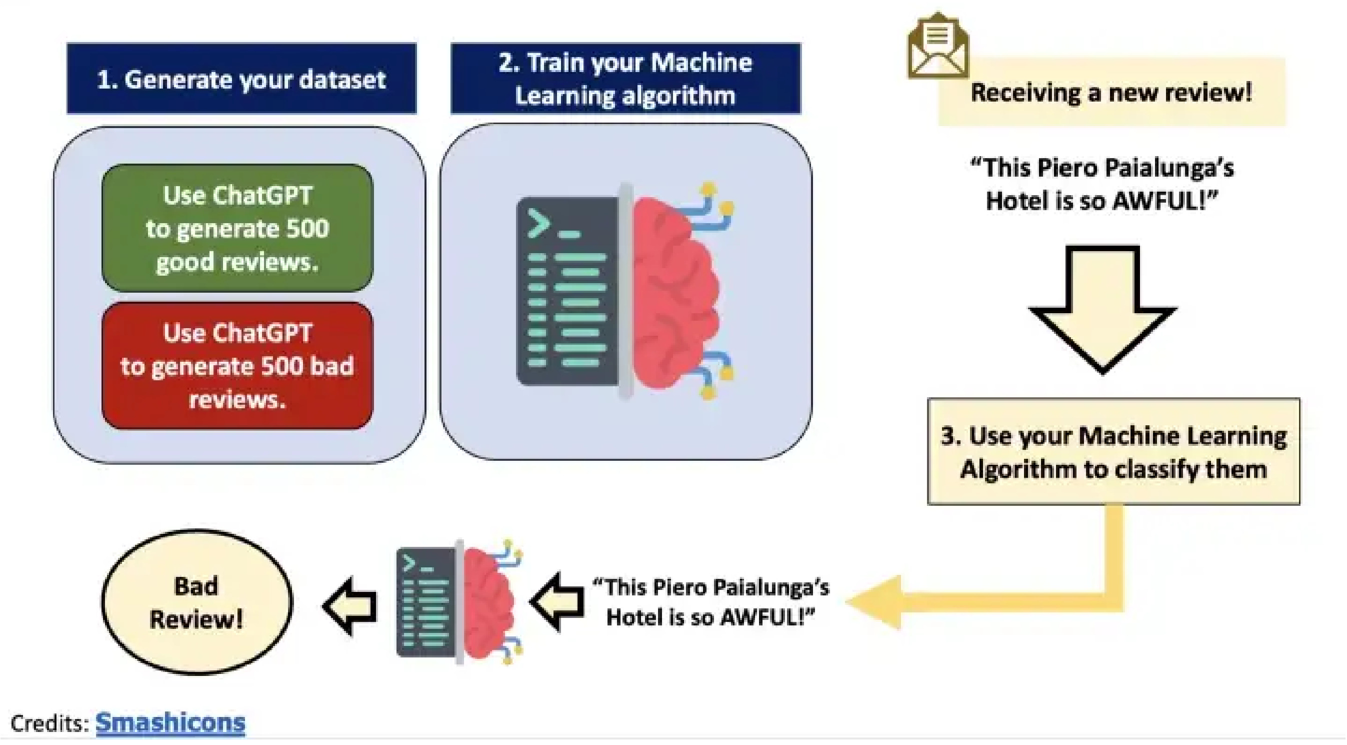
Now that we have ChatGPT, though, we can simply ask them to generate **good and bad reviews!** This would take minutes (rather than months), and it will allow us to build our machine learning algorithm to automatically classify our customer reviews!

Congratulations, this is your first **Surrogate Model.** 😊

Keep in mind that **we will not train ChatGPT or do any fine-tuning.** This model is exceptional for a task like this, and no fine-tuning is required in this instance. Now, the training of the ChatGPT model is, of course, not open source (just like the model itself). All we know is the small description that is in the Open AI's blog. They explain that the model is trained by human AI trainers and a reinforcement-learning supervised algorithm.*

> **The fact alone that OpenAI's ChatGPT is not open source raises some very tricky and interesting* ***ethical questions****. Should such a powerful model be open source, so that everyone (bad people too) can use it, or should it be not open source so that no one can really trust it?*

Let me recap:



1. Generate your dataset
Use ChatGPT to generate 500 good reviews.
Use ChatGPT to generate 500 bad reviews.
2. Train your Machine Learning algorithm
Receiving a new review!
"This Piero Paialunga's Hotel is so AWFUL!"
3. Use your Machine Learning Algorithm to classify them
"This Piero Paialunga's Hotel is so AWFUL!"
Bad Review!
Credits: Smashicons

Image by author

The little brain-shell thing that you see is the **surrogate model;** as we will see, it will be a **random forest**. But I said it was a **hands-on** article, so let's dive in! (so excited!!!)

> **I am sorry, I love spoilers.*

1. Generating our dataset

The first step is to use **Open's AI Python API to generate our simulations.**
A few things to consider:
1. **Open AI is made by geniuses for non-genius users**. For this reason, if you want to install it, you just have to do:

```
pip install --upgrade openai
```

(that is **LOVABLE**)

2. Of course, if you want to send a lot of requests, you will have to pay for a premium service. Assuming we don't want to do that, we'd just have to wait **around 30 minutes** to get our dataset of fake reviews. Again, this is **nothing** compared to the waiting time (and cost) of **months** that we'd have to wait if we did this manually. You'll also have to log in to Open AI and get your Open AI key.

3. We will automatically input whether this is a good review or a bad review by **starting** with the same sentence: “This hotel was terrible.” for a bad review and “This hotel was great.” for a good review. ChatGPT will complete our review for us. Of course, except for these first four words, which we won’t include in our reviews anyway, the rest of the review will be different.

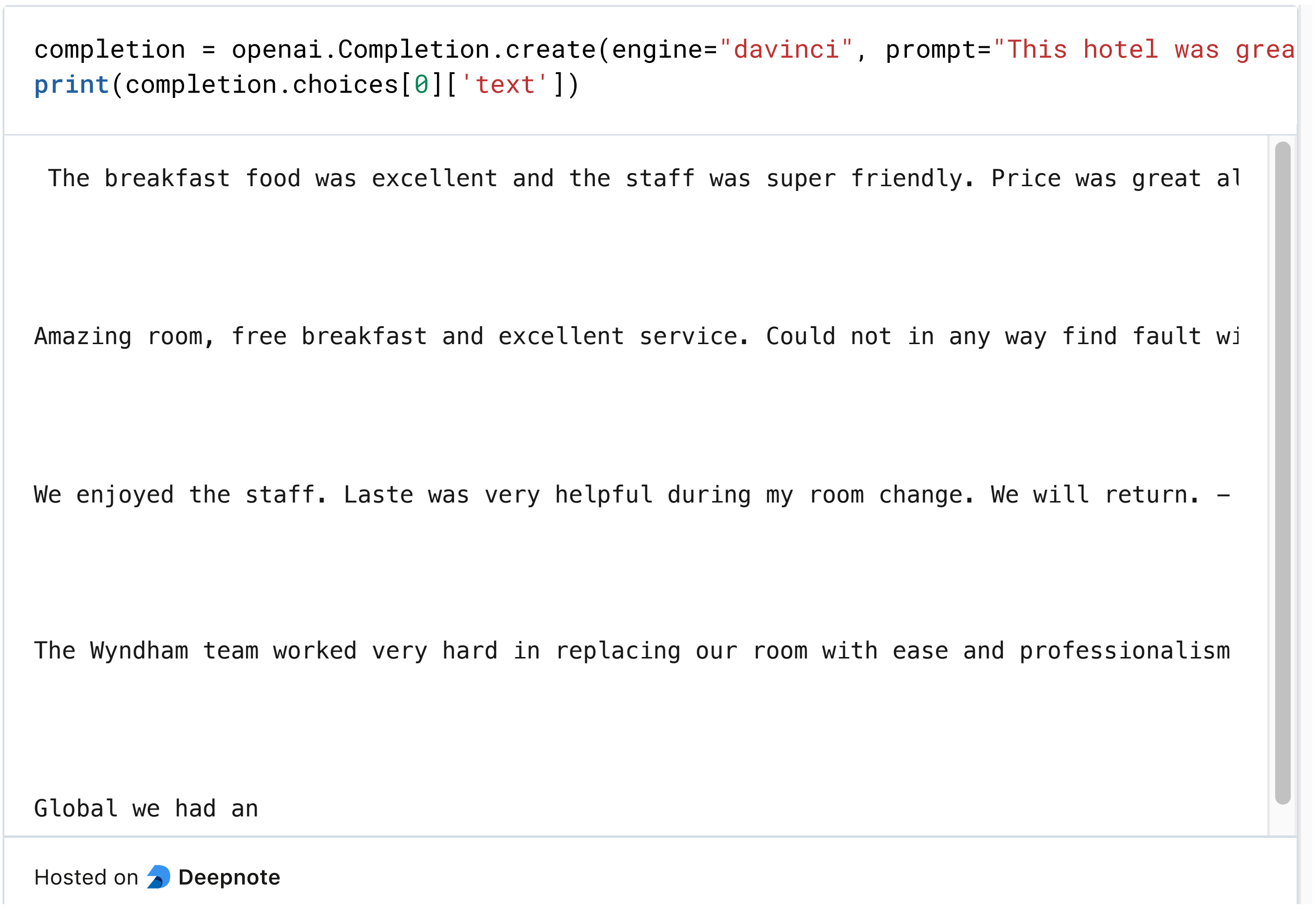
Let me give you an example of a bad review:

```python
import openai
import time
openai.api_key = "your_key"

completion = openai.Completion.create(engine="davinci", prompt="This hotel was ter
print(completion.choices[0]['text'])
```

```
 There were broken chairs everywhere and even one broken elevator. The hallways smel



The concier
```

Hosted on Deepnote

And of a good one:



```
completion = openai.Completion.create(engine="davinci", prompt="This hotel was grea
print(completion.choices[0]['text'])
```

```
 The breakfast food was excellent and the staff was super friendly. Price was great al



Amazing room, free breakfast and excellent service. Could not in any way find fault wi



We enjoyed the staff. Laste was very helpful during my room change. We will return. -



The Wyndham team worked very hard in replacing our room with ease and professionalism



Global we had an
```

Hosted on Deepnote

Now this is the code you will need to generate **your whole dataset.**

```
good_reviews = []
bad_reviews = []
for i in range(0,500):
  completion = openai.Completion.create(engine="davinci", prompt="This hotel was g
  good_reviews.append(completion.choices[0]['text'])
  print('Generating good review number %i'%(i))
  completion = openai.Completion.create(engine="davinci", prompt="This hotel was t
  bad_reviews.append(completion.choices[0]['text'])
  print('Generating bad review number %i'%(i))
  display = np.random.choice([0,1],p=[0.7,0.3])
  time.sleep(3)
  if display ==1:
    display_good = np.random.choice([0,1],p=[0.5,0.5])
    if display_good ==1:
      print('Printing random good review')
      print(good_reviews[-1])
    if display_good ==0:
      print('Printing random bad review')
      print(bad_reviews[-1])
```

```
Printing random bad review
 The bathrooms smelled like mold and the fans shuddered making it next to impossible
Generating good review number 491
Generating bad review number 491
Generating good review number 492
Generating bad review number 492
Generating good review number 493
Generating bad review number 493
Generating good review number 494
Generating bad review number 494
Generating good review number 495
Generating bad review number 495
Printing random bad review
 It is not new as advertised and they are renovating as you are there in the main pa
Generating good review number 496
Generating bad review number 496
Printing random bad review
 It's apparent that the hotel is trying to save money. The rooms are so small that t

Nicole on
Generating good review number 497
Generating bad review number 497
Generating good review number 498
Generating bad review number 498
Generating good review number 499
Generating bad review number 499
Printing random bad review
 I'll just say that our evening entertainment was terrible. This hotel was chosen be

Sam Iwasra Switzerland, - 17 March 2018
```

We will then store everything in a **dataframe** using **Pandas.**

1. **Import** pandas and **build** df:

```
import pandas as pd
import numpy as np
df = pd.DataFrame(np.zeros((1000,2)))
df.columns = ['Reviews','Sentiment']
df['Sentiment'].loc[0:499] = 1
```

Hosted on **Deepnote**

2. **Fill** df:

```
df['Reviews'] = good_reviews+bad_reviews
```

Hosted on **Deepnote**

3. **Export** df:

```
df.to_csv('generated_reviews.csv')
```

Hosted on **Deepnote**

2. Machine Learning time!

So now we need to build and train a machine learning algorithm.
As we are dealing with **texts,** the first thing that we need to do is use a **vectorizer.** A vectorizer is something that transforms a **text** into a **vector**.

Like:

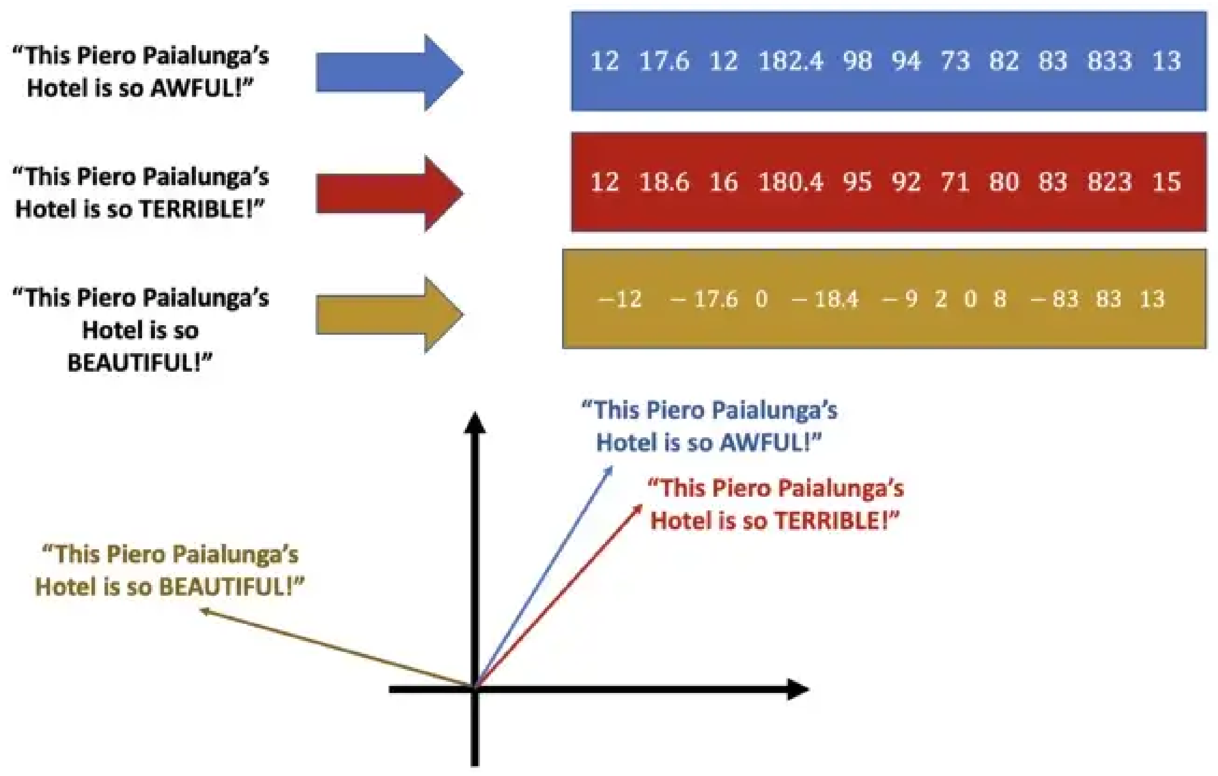

"This Piero Paialunga's Hotel is so AWFUL!"
12 17.6 12 182.4 98 94 73 82 83 833 13
"This Piero Paialunga's Hotel is so TERRIBLE!"
12 18.6 16 180.4 95 92 71 80 83 823 15
"This Piero Paialunga's Hotel is so BEAUTIFUL!"
−12 − 17.6 0 − 18.4 − 9 2 0 8 − 83 83 13
"This Piero Paialunga's Hotel is so AWFUL!"
"This Piero Paialunga's Hotel is so TERRIBLE!"
"This Piero Paialunga's Hotel is so BEAUTIFUL!"

Image by author

As you can see, **similar texts** have **similar vectors** (I know, similarity is a tricky concept, but you know what I mean). and **different texts** have **non-similar vectors.**

There are tons of ways to do the vectorization steps. Some ways are more complex than others; some ways are more **efficient than others;** some ways require machine learning, and some ways don't.

For the purpose of this project (and because I am not a NLP Machine Learning engineer), we will use a fairly simple one, which is called TfIDF vectorizer, ready to use on SkLearn.

Let's start by importing the libraries:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix,plot_confusion_matrix
from sklearn.feature_extraction.text import TfidfVectorizer
```

Hosted on Deepnote

And by importing the dataset we just generated using ChatGPT, with a little bit of preprocessing here and there…

```
labeled_data = pd.read_csv('generated_reviews.csv').drop(columns=['Unnamed: 0'])
labeled_data.Sentiment = labeled_data.Sentiment.astype(int)
labeled_data = labeled_data.dropna().reset_index()
```

Hosted on **Deepnote**

And here we go:

```
labeled_data.head()
```

Hosted on **Deepnote**

Fantastic. Now let's do this vectorization thing:

```
dataset = labeled_data
from transformers import AutoTokenizer

#tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
#tokenized_data = tokenizer(dataset["Reviews"].values.tolist(), return_tensors="np
vectorizer = TfidfVectorizer (max_features=2500, min_df=7, max_df=0.8)
tokenized_data = vectorizer.fit_transform(dataset['Reviews']).toarray()

labels = np.array(dataset["Sentiment"])  # Label is already an array of 0 and 1
```

```
/Users/pieropaialunga/miniforge3/envs/tf/lib/python3.8/site-packages/tqdm/auto.py:22:
  from .autonotebook import tqdm as notebook_tqdm
```

Hosted on **Deepnote**

As I was telling you earlier, the machine learning model we will be using is known as **Random Forest.** What is a random forest? It is a collection of **decision trees**. And what is a decision tree?

A decision tree is a Machine Learning algorithm that, given a certain information theory criterion, optimizes a tree-search of all the possible splits of the features of your dataset, until it finds a way to distinguish what is 1 and what is 0 based on that split.*

> ** I am sorry if it is super confusing, but explaining this in 4 lines is a hard task. This **article** takes its time to do so and it does that splendidly. Highly recommended.*

Now let's:

1. Define our random forest:

```
rf = RandomForestClassifier(n_estimators=100)
```

Hosted on Deepnote

2. Split our dataset into training and testing:

```
X = tokenized_data
y = labels
X_train, X_test,y_train, y_test = train_test_split(X,y,test_size=0.2)
```

Hosted on Deepnote

3. Train our model:

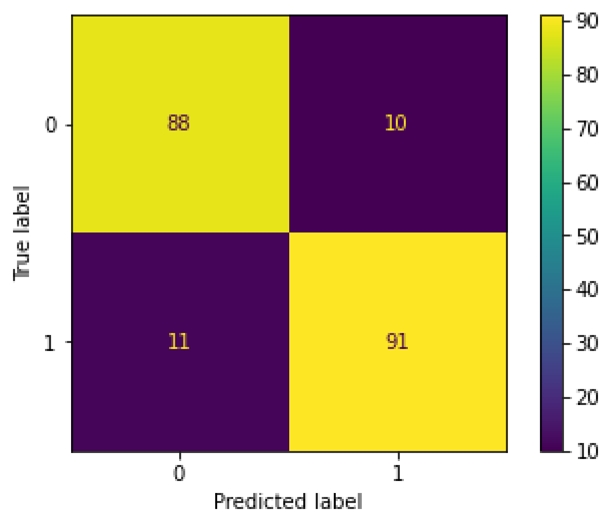
```
rf.fit(X_train,y_train)
```

Hosted on Deepnote

The results are pretty **impressive**, especially considering the lack of hyperparameter tuning.

```
plot_confusion_matrix(rf,X_test,y_test)
```

```
/Users/pieropaialunga/miniforge3/envs/tf/lib/python3.8/site-packages/sklearn/utils/depr
  warnings.warn(msg, category=FutureWarning)
```



Hosted on Deepnote

3. Sentiment Analysis

As we have our trained model, you can use that on a new, unlabeled dataset. I used a set of New York City hotel reviews I found online, but you can use your own **or** you could make up a review and see how it works.

This dataset is open source (CC0: Public Domain), very small (2MB) and can be downloaded on Kaggle.

```python
target_data = pd.read_csv('NYC_2021_airbnb_reviews_data1.csv')
target_data.drop('url',axis=1)
```

```
       listing_id  review_posted_date
0      2595        November 2019       Great location, convenient to ever
1      2595        May 2019            Place was so cute and comfy! Host
2      2595        May 2019            10 / 10 would stay again
3      2595        January 2019        The apartment met expectations to
4      2595        December 2018       Great space in a fun old building
...    ...         ...                 ...
17439  1918693     February 2022       Lovely Brownstone in Brooklyn. Cle
17440  1918693     January 2022        We had a great stay at Lorelei & A
17441  1918693     December 2021       This was a perfect spot for mine a
17442  1918693     November 2021       A lovely spot in a lovely neighbor
17443  1918693     November 2021       Overall great stay. Lorelei and Al
17444 rows × 3 columns
```

Let's preprocess the review column (or your text):

```python
dataset = target_data
vectorizer = TfidfVectorizer (max_features=2500, min_df=7, max_df=0.8)
vectorizer.fit(dataset['Reviews'])
new_data_processed = vectorizer.transform(target_data['review']).toarray()
y_pred = rf.predict(new_data_processed)
```

And let's print our predictions:

```python
J = np.random.choice(range(0,len(new_data_processed)),5)
for j in J:
    print('Review number %i: \n'%(j))
    print(target_data['review'].loc[j])
    print('Classified as %i (1=good, 0=bad)' %(y_pred[j]))
```

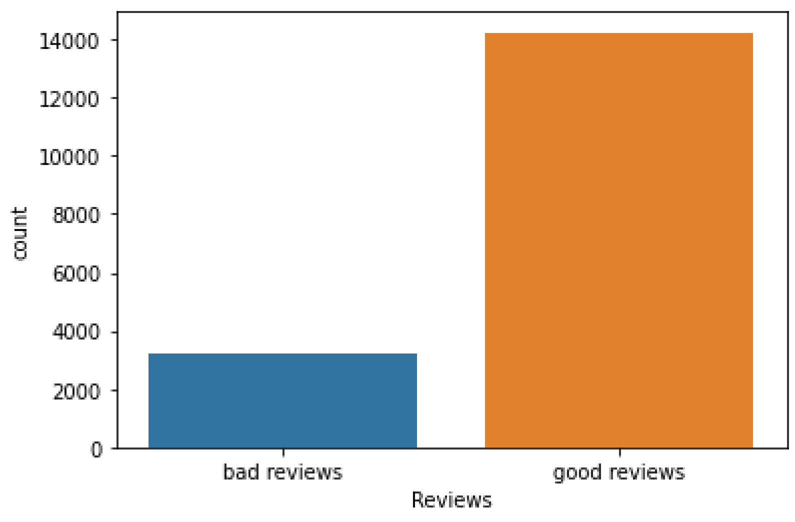
```
Review number 13052:

Great stay. Accurate to listing.
Classified as 1 (1=good, 0=bad)
Review number 8373:

The studio was a great place to stay for a couple days while on a work trip. Central
Classified as 1 (1=good, 0=bad)
Review number 9744:

Thank you
Classified as 0 (1=good, 0=bad)
Review number 10006:

Zora & Chris are the best host you could expect for. They are always available and ar

The place is even better in real life than the photos : clean, cosy and warm, it fee
If you need a place to stay in NYC, I definitely recommend it!
Classified as 1 (1=good, 0=bad)
Review number 7507:

Incredible area, a little funky that you have to walk through a common area to get t
Classified as 1 (1=good, 0=bad)
```

Hosted on Deepnote

As we can see, all these 5 random reviews that are classified as 1 are actually good!

Let’s show a count plot:

```
sns.countplot(x=pd.DataFrame(y_pred)[0])
plt.xticks([0,1],['bad reviews','good reviews'])
plt.xlabel('Reviews')
```



Hosted on Deepnote

4. Considerations

So what did we do here?

1. We acknowledge that **ChatGPT is awesome**.
2. We used ChatGPT to **build a dataset** for our surrogate model. More specifically, we used ChatGPT to make up good and bad reviews for hotels.
3. We used that **labeled dataset** that we built to train our Machine Learning model. The model that we used is a **Random Forest Classifier**.
4. We tested our trained model on a new dataset getting **promising results.**

Is there any room for improvement? A ton.

2. We can **improve our querying skills** by giving different inputs, maybe also in other languages rather than english only

3. We can **improve our Machine Learning model** by doing some hyperparameter tuning

Now, let me conclude this with a thought.

There are, and there will be, a lot of concerns about how and who is going to use Open AI ChatGPT. While I am not a lawyer (let alone an expert in ethical AI), I can imagine how this tool can be dangerous in many ways and on many different levels.

I strongly disagree with the people who are not impressed by the **performance** of ChatGPT, because I find it pretty amazing and I am so excited to see how this technology is going to evolve. I hope this toy example sparks something in my readers too. ❤️

5. Conclusions

If you liked the article and you want to know more about machine learning, or you just want to ask me something, you can:

A. Follow me on **Linkedin**, where I publish all my stories
B. Subscribe to my **newsletter**. It will keep you updated about new stories and give you the chance to text me to receive all the corrections or doubts you may have.
C. Become a **referred member**, so you won't have any "maximum number of stories for the month" and you can read whatever I (and thousands of other Machine Learning and Data Science top writers) write about the newest technology available.

Chat Open AI Machine Learning Artificial Intelligence NLP