etection ☆

1. Load the data and load all the libraries
2. Data Preparation and Data transformation
    1. Convert all text into LowerCase
    2. Remove all special characters
    3. Remove stop words
    4. Lemmatization and Stemming
3. Vectorization
    1. TFIDF Vectorizer
4. Machine Learning and also Deep Learning

[31] 0s

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

[32] 0s

```
df = pd.read_csv('spam.csv', encoding = 'latin-1')
```

ratory × Gradio × +

.research.google.com/drive/1bH1jUZM-OOUFrTP8Wjmazvv0GzLM8-Cp#scrollTo=A2yLBJAEpcmZ

etection ☆

ew Insert Runtime Tools Help All changes saved

Comme

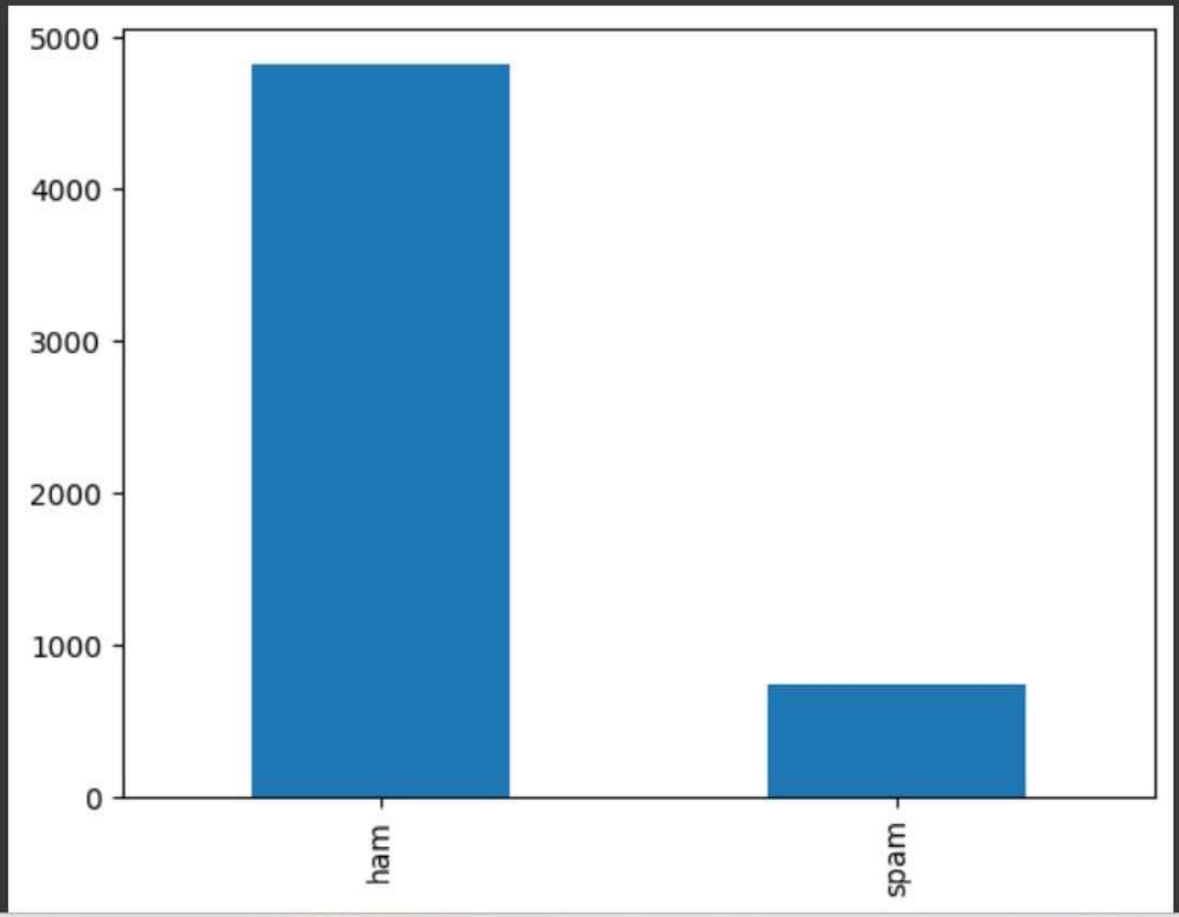
+ Code + Text

```
memory usage: 87.2+ KB
```

0s

```
[36] df1['v1'].value_counts().plot(kind='bar')
```

```
<Axes: >
```



82.83 GB available

Search

etection ☆

0s
```
nltk.download('stopwords')
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

0s
```
[39] nltk.download('stopwords')
     lemmatizer = WordNetLemmatizer()
     stemmer = PorterStemmer()
     nltk.download('wordnet')
     def preprocess(sentence):
         sentence=str(sentence)
         sentence = sentence.lower()
         sentence=sentence.replace('{html}',"")
         cleanr = re.compile('<.*?>')
         cleantext = re.sub(cleanr, '', sentence)
         rem_url=re.sub(r'http\S+', '',cleantext)
         rem_num = re.sub('[0-9]+', '', rem_url)
         tokenizer = RegexpTokenizer(r'\w+')
         tokens = tokenizer.tokenize(rem_num)
         filtered_words = [w for w in tokens if len(w) > 2 if not w in stopwords.words('english')]
         stem_words=[stemmer.stem(w) for w in filtered_words]
         lemma_words=[lemmatizer.lemmatize(w) for w in stem_words]
         return " ".join(filtered_words)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
```

```
[nltk_data]   Package wordnet is already up-to-date!
```

11s
```
[40] df1['v2'] = df1['v2'].map(lambda s:preprocess(s))
```

0s
```
[41] df1['v2']
```

```
0       jurong point crazy available bugis great world...
1                                   lar joking wif oni
2       free entry wkly comp win cup final tkts may te...
3                           dun say early hor already say
4                  nah think goes usf lives around though
                              ...
5567    time tried contact pound prize claim easy call...
5568                                going esplanade home
5569                               pity mood suggestions
5570    guy bitching acted like interested buying some...
5571                                       rofl true name
Name: v2, Length: 5572, dtype: object
```

0s
```
[42] df['v2']
```

```
0       Go until jurong point, crazy.. Available only ...
1                           Ok lar... Joking wif u oni...
2       Free entry in 2 a wkly comp to win FA Cup fina...
3       U dun say so early hor... U c already then say...
4       Nah I don't think he goes to usf, he lives aro...
                              ...
5567    This is the 2nd time we have tried 2 contact u...
```

```
5568                Will I_ b going to esplanade fr home?
5569    Pity, * was in mood for that. So...any other s...
5570    The guy did some bitching but I acted like i'd...
5571                          Rofl. Its true to its name
Name: v2, Length: 5572, dtype: object
```

Vectorize the text

[43]
```
from sklearn.feature_extraction.text import TfidfVectorizer
vect = TfidfVectorizer()
```

[44]
```
x = vect.fit_transform(df1['v2'])
```

[45]
```
x.shape
```

```
(5572, 7386)
```

[46]
```
type(x)
```

```
scipy.sparse._csr.csr_matrix
```

[47]
```
le = LabelEncoder()
```

[48]
```
y = le.fit_transform(df1['v1'])
```

etection ☆

ew Insert Runtime Tools Help All changes saved

Comme

+ Code + Text

0s

```
[50] print(x_train.shape)
     print(x_test.shape)
     print(y_train.shape)
     print(y_test.shape)
```

```
(4457, 7386)
(1115, 7386)
(4457,)
(1115,)
```

```
[51] from sklearn.model_selection import RandomizedSearchCV
     from sklearn.ensemble import RandomForestClassifier
```

```
[52] random_grid = {'criterion': ['gini', 'entropy', 'log_loss'],
                    'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110],
                    'min_samples_leaf': [1, 2, 4],
                    'min_samples_split': [2, 5, 10],
                    'n_estimators': [130, 180, 230]}
```

```
[53] rf = RandomForestClassifier()
```

```
[54] clf = RandomizedSearchCV(estimator = rf, param_distributions = random_grid, verbose = 2, random_stat
```

```
[55] search = clf.fit(x_train, y_train)
```

82.83 GB available

```
[CV] END criterion=log_loss, max_depth=50, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=log_loss, max_depth=50, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=log_loss, max_depth=50, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=log_loss, max_depth=50, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=log_loss, max_depth=50, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=entropy, max_depth=110, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=entropy, max_depth=110, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=entropy, max_depth=110, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=entropy, max_depth=110, min_samples_leaf=1, min_samples_split=2, n_estimators=130
[CV] END criterion=entropy, max_depth=110, min_samples_leaf=1, min_samples_split=2, n_estimators=130
```

3m

```
[56] search.best_params_
```

0s

```
{'n_estimators': 230,
 'min_samples_split': 10,
 'min_samples_leaf': 1,
 'max_depth': 70,
 'criterion': 'gini'}
```

```
[57] search.best_score_
```

0s

```
0.9762148138082368
```

```
[59] from sklearn.ensemble import RandomForestClassifier
     rf = RandomForestClassifier(n_estimators = 230,max_depth = 3,min_samples_split= 10,
      min_samples_leaf= 1,criterion = 'gini',verbose = 2)
     rf.fit(x_train,y_train)
```

1s

```
building tree 214 of 230
[59] building tree 215 of 230
building tree 216 of 230
building tree 217 of 230
building tree 218 of 230
building tree 219 of 230
building tree 220 of 230
building tree 221 of 230
building tree 222 of 230
building tree 223 of 230
building tree 224 of 230
building tree 225 of 230
building tree 226 of 230
building tree 227 of 230
building tree 228 of 230
building tree 229 of 230
building tree 230 of 230
[Parallel(n_jobs=1)]: Done  40 tasks      | elapsed:    0.0s
[Parallel(n_jobs=1)]: Done 161 tasks      | elapsed:    0.1s
[Parallel(n_jobs=1)]: Done  40 tasks      | elapsed:    0.0s
[Parallel(n_jobs=1)]: Done 161 tasks      | elapsed:    0.0s
```

```
[60] print('Train accuracy score of the model is: ', round(accuracy_score(y_train,preds_rf_train),2))
     print('Test accuracy score of the model is: ', round(accuracy_score(y_test, preds_rf_test),2))
```

```
Train accuracy score of the model is:  0.86
Test accuracy score of the model is:  0.88
```

Application of NB Classifier

etection ☆

Application of NB Classifier

0s
```
[61] from sklearn.naive_bayes import MultinomialNB
```

0s
```
nb = MultinomialNB()
```

2s
```
[63] nb.fit(x_train.toarray(), y_train)
     nb_preds_train = nb.predict(x_train.toarray())
     nb_preds_test = nb.predict(x_test.toarray())
     print('Train accuracy score of the model is: ', round(accuracy_score(y_train, nb_preds_train),2))
     print('Test accuracy score of the model is: ', round(accuracy_score(y_test, nb_preds_test),2))
```

```
Train accuracy score of the model is:  0.98
Test accuracy score of the model is:  0.97
```

31s
```
[64] !pip install transformers
```

```
Collecting transformers
  Downloading transformers-4.31.0-py3-none-any.whl (7.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.4/7.4 MB 12.5 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformer
Collecting huggingface-hub<1.0,>=0.14.1 (from transformers)
  Downloading huggingface_hub-0.16.4-py3-none-any.whl (268 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 268.8/268.8 kB 12.6 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transfor
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from tran
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transfor
```

ratory × Gradio × +

.research.google.com/drive/1bH1jUZM-OOUFrTP8Wjmazvv0GzLM8-Cp#scrollTo=U7nk-EQ-dpq0

etection ☆

ew Insert Runtime Tools Help All changes saved

Comme

+ Code + Text

31s [64]
```
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from request
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from r
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from r
Installing collected packages: tokenizers, safetensors, huggingface-hub, transformers
Successfully installed huggingface-hub-0.16.4 safetensors-0.3.2 tokenizers-0.13.3 transformers-4.31.
```

16s [65]
```
from transformers import pipeline
```

9s [66]
```
pipe = pipeline("text-classification")
```

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english and revision af0
Using a pipeline without specifying a model name and revision in production is not recommended.
```

Downloading (…)lve/main/config.json: 100% 629/629 [00:00<00:00, 22.6kB/s]

Downloading model.safetensors: 100% 268M/268M [00:01<00:00, 173MB/s]

Downloading (…)okenizer_config.json: 100% 48.0/48.0 [00:00<00:00, 2.59kB/s]

Downloading (…)solve/main/vocab.txt: 100% 232k/232k [00:00<00:00, 674kB/s]

```
No CUDA runtime is found, using CUDA_HOME='/usr/local/cuda'
Xformers is not installed correctly. If you want to use memory_efficient_attention to accelerate tra
pip install xformers.
```

0s [67]
```
df1['v2'][0]
```

82.83 GB available

Search

etection ☆

ew Insert Runtime Tools Help All changes saved

Comme

+ Code + Text

```
'jurong point crazy available bugis great world buffet cine got amore wat'
```

[68] `preds = pipe(str(df1['v2'][0]))` 1s

[69] `print(preds)` 0s

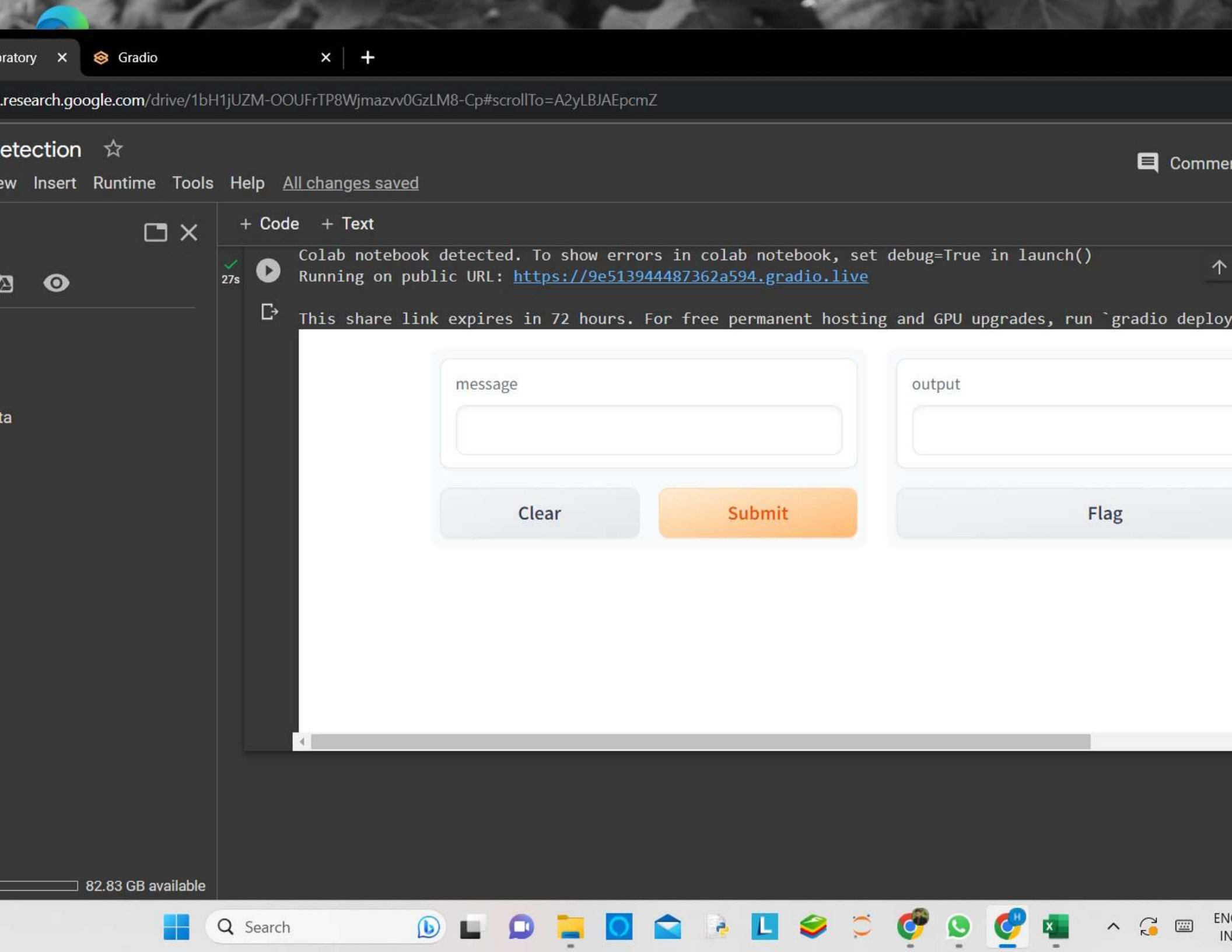
```
[{'label': 'NEGATIVE', 'score': 0.983607292175293}]
```

[74] 27s

```
!pip install gradio
import gradio as gr
from transformers import pipeline

classifier = pipeline("text-classification")

def classify_message(message):
    result = classifier(message)
    label = result[0]['label']
    score = result[0]['score']
    return f"Predicted Label: {label}, Confidence Score: {score:.2f}"

iface = gr.Interface(fn=classify_message, inputs="text", outputs="text")
iface.launch(share=True)
```

```
Requirement already satisfied: gradio in /usr/local/lib/python3.10/dist-packages (3.39.0)
Requirement already satisfied: aiofiles<24.0,>=22.0 in /usr/local/lib/python3.10/dist-packages (from
```

82.83 GB available


Gradio
.research.google.com/drive/1bH1jUZM-OOUFrTP8Wjmazvv0GzLM8-Cp#scrollTo=A2yLBJAEpcmZ
etection
ew Insert Runtime Tools Help All changes saved
+ Code + Text
27s
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
Running on public URL: https://9e513944487362a594.gradio.live
This share link expires in 72 hours. For free permanent hosting and GPU upgrades, run `gradio deploy
message
output
Clear
Submit
Flag
82.83 GB available
Search