DEV RANGERA CS-B5 · 12M AGO · 1 VIEW · PRIVATE

0 Edit

Neural Project1

Python · Popular Video Games 1980 - 2023

Notebook Input Output Logs Comments (0) Settings

Run
3.9s

Version 2 of 2

Add Tags

Table of Contents
- EDA
- Test Set
- Model part

In [149]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/popular-video-games-1980-2023/games.csv
```

In [150]:
```
# Import training data
train = pd.read_csv('/kaggle/input/popular-video-games-1980-2023/games.csv')

# Print few random entries
train.sample(5)
```

Out[150]:

	Unnamed: 0	Title	Release Date	Team	Rating	Times Listed	Number of Reviews	Genres	Summary	Reviews	Plays	Play
1017	1017	Shin Megami Tensei III: Nocturne - HD Remaster	Oct 28, 2020	['Ghostlight Ltd.', 'Atlus']	4.0	457	457	['RPG', 'Turn Based Strategy']	200X. Everyday life suddenly came to an end. T...	['cockturne', "It's a must play if you want a ...	1.8K	387
126	126	Super Mario Bros. 3	Oct 23, 1988	['Nintendo R&D4', 'Nintendo']	4.0	1.6K	1.6K	['Platform']	Super Mario Bros. 3, the third entry in the Su...	['i remembered i beat it with warps', "I first...	14K	109
1366	1366	Mega Man X6	Nov 28, 2001	['Capcom']	2.3	308	308	['Adventure', 'Platform']	Mission Critical... Zero is Missing... Search ...	['Acho incrivel que ele é pior que o X5 mas me...	1.9K	18
21	21	Hi-Fi Rush	Jan 25, 2023	['Tango Gameworks', 'Bethesda Softworks']	4.3	926	926	['Adventure', 'Brawler', 'Music', 'Platform']	As wannabe rockstar Chai, you'll fight back ag...	["It was a great game all round ending felt a ...	3K	866
304	304	Dragon Age: Origins	Nov 03, 2009	['Electronic Arts', 'BioWare']	4.0	712	712	['RPG']	Dragon Age: Origins is a third-person role-pla...	['Console camera suits the game more. Change m...	6.1K	148

In [151]:
```
train.isnull().sum()
```

Out[151]:
```
Unnamed: 0            0
Title                 0
Release Date          0
Team                  1
Rating               13
Times Listed          0
Number of Reviews     0
Genres                0
Summary               1
Reviews               0
Plays                 0
Playing               0
Backlogs              0
Wishlist              0
dtype: int64
```

EDA

In [152]:
```
train.columns
```

Out[152]:
```
Index(['Unnamed: 0', 'Title', 'Release Date', 'Team', 'Rating', 'Times Listed',
       'Number of Reviews', 'Genres', 'Summary', 'Reviews', 'Plays', 'Playing',
       'Backlogs', 'Wishlist'],
      dtype='object')
```

In [153]:
```
train.drop(['Unnamed: 0', 'Title', 'Release Date', 'Team', 'Times Listed',
```

```
        'Number of Reviews', 'Genres', 'Summary','Plays', 'Playing',
        'Backlogs', 'Wishlist'],axis = 1,inplace=True)
```

In [154]:
```
train = train.dropna()
```

In [155]:
```
train.isnull().sum()
```

Out[155]:
```
Rating     0
Reviews    0
dtype: int64
```

In [156]:
```
# Checking the number of rows and columns
print("The DataFrame has " + str(train.shape[0]) + " samples and " + str(train.shape[1]) + " column
s")
```

```
The DataFrame has 1499 samples and 2 columns
```

In [157]:
```
train.head()
```

Out[157]:

	Rating	Reviews
0	4.5	["The first playthrough of elden ring is one o...
1	4.3	['convinced this is a roguelike for people who...
2	4.4	['This game is the game (that is not CS:GO) th...
3	4.2	['soundtrack is tied for #1 with nier automata...
4	4.4	["this games worldbuilding is incredible, with...

In [158]:
```
# Check duplicates
print("Duplicate entries in the dataset: " + str(train.duplicated().sum()))
```

```
Duplicate entries in the dataset: 384
```

In [159]:
```
train = train.drop_duplicates()
```

In [160]:
```
# Check duplicates
print("Duplicate entries in the dataset: " + str(train.duplicated().sum()))
```

```
Duplicate entries in the dataset: 0
```

In [161]:
```
# Check null values and data type of each column
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1115 entries, 0 to 1511
Data columns (total 2 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Rating   1115 non-null   float64
 1   Reviews  1115 non-null   object
dtypes: float64(1), object(1)
memory usage: 26.1+ KB
```

In [162]:
```
#Some Libraries
import warnings
import tensorflow as tf
from tensorflow import keras
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
```

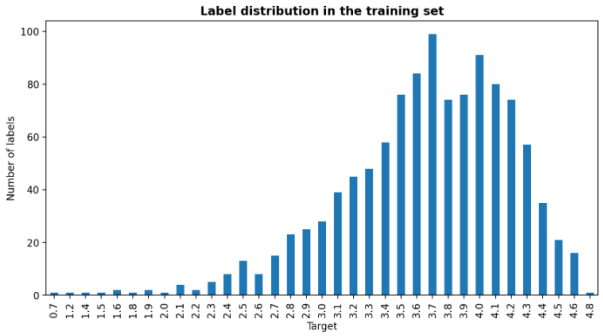
In [163]:
```
import plotly.express as px
train['length_of_text'] = [len(i.split(' ')) for i in train['Reviews']]

fig = px.histogram(train['length_of_text'], marginal='box',
                   labels={"value": "Length of the Text"})

fig.update_traces(marker=dict(line=dict(color='#000000', width=2)))
fig.update_layout(title_text='Distribution of the Length of the Texts',
                  title_x=0.5, title_font=dict(size=22))
fig.show()
```

In [164]:
```
FreqOfWords = train['Reviews'].str.split(expand=True).stack().value_counts()
FreqOfWords_top200 = FreqOfWords[:200]
```

```
fig = px.treemap(FreqOfWords_top200, path=[FreqOfWords_top200.index], values=0)
fig.update_layout(title_text='Frequency of the Words in the Train Dataset',
                  title_x=0.5, title_font=dict(size=22)
                  )
fig.update_traces(textinfo="label+value")
fig.show()
```

In [165]:

```
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
# Check target balance
train['Rating'].value_counts().sort_index().plot.bar(figsize=(10,5))
plt.title('Label distribution in the training set', fontweight="bold")
plt.xlabel('Target')
plt.ylabel('Number of labels')
plt.show()
```



In [166]:

```
import re
from nltk.corpus import stopwords

# Create a stopwords set adding some personal 'words'
stopwords_english = set(stopwords.words('english'))
my_stopwords = set(["http", "'s", "n't", "'m", "'re", "'ve"])
stopwords_english.update(my_stopwords)

def preprocess_review(text):
    # Convert to lower case
    text = text.lower()

    # Remove numbers
    text = re.sub(r'\d+', '', text)

    # Remove punctuation
    text = re.sub(r'[^\w\s]', '', text)

    # Split text into tokens
    tokens = text.split()

    # Filter tokens
    clean_tokens = [tok for tok in tokens if tok not in stopwords_english and len(tok) > 1]

    # Join tokens into a string
    clean_text = ' '.join(clean_tokens)

    return clean_text
```

In [167]:

```
# Get one review as sample
sample = train['Reviews'][20]

print('ORIGINAL REVIEW:   ' + sample + '\n')
print('------------------------------------------------------------------------------- \n')
print('WITH PROCESSING:    ' + preprocess_review(sample))
```

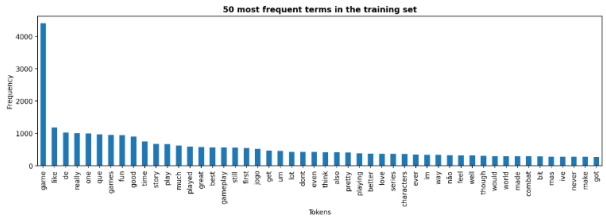
```
ORIGINAL REVIEW:   ['The more mature story (by recent Pokémon standards) , lore ramifications fo
r sinnoh , and the amazing real time catch mechanics make this game one of the best Pokémon game
s ever easily ;though a good Pokémon is still far from being a good video game generally speakin
g . The textures are muddy , the levels especially alabaster Icelands and cobalt coastlands feel
janky and overly linear despite the game encouraging more mobile movement and freedom , and of c
ourse the lack of meaningful presentation through stiff character animations and lack of voice a
cting hurt this game alot. I still feel like Pokémon gameplay has peaked here but they just do
n't let these games cook long enough and only get away with it because Pokémon fans only play Po
kémon and don't branch out to know what's actually good', 'Really refreshing take on the series,
just a little empty overall.', "Great game! but it would be better if there was voice acting and
no sudden cutscenes every 2 seconds. And it gets pretty bad as you play it. But if you're findin
g a good first Pokemon game, I would recommend it tho!", 'Aside from obvious performance issues,
this was easily the most fun ive had with the Pokemon series since the release of Black/White. i
hope we get a follow up game in the future that focuses on the unova region and kyurem', "Put me
in feudal japan with an Iphone and I probably wouldn't get as far as I did here", 'PLA: the best
```

```
pokémon game I've ever played and why I'll (probably) never play another Pokémon game in my lif
e']

--------------------------------------------------------------------------------

WITH PROCESSING:    mature story recent pokémon standards lore ramifications sinnoh amazing real
time catch mechanics make game one best pokémon games ever easily though good pokémon still far
good video game generally speaking textures muddy levels especially alabaster icelands cobalt co
astlands feel janky overly linear despite game encouraging mobile movement freedom course lack m
eaningful presentation stiff character animations lack voice acting hurt game alot still feel li
ke pokémon gameplay peaked dont let games cook long enough get away pokémon fans play pokémon do
nt branch know whats actually good really refreshing take series little empty overall great game
would better voice acting sudden cutscenes every seconds gets pretty bad play youre finding good
first pokemon game would recommend tho aside obvious performance issues easily fun ive pokemon s
eries since release blackwhite hope get follow game future focuses unova region kyurem put feuda
l japan iphone probably wouldnt get far pla best pokémon game ive ever played ill probably never
play another pokémon game life
```

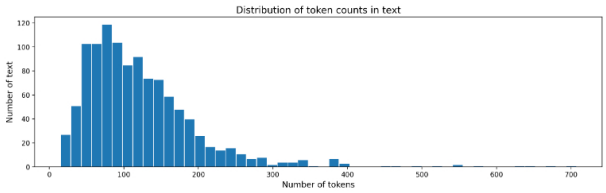
In [168]:
```
%%time

# Preprocess training data adding new column
train["clean_text"] = train["Reviews"].apply(preprocess_review)
```

```
CPU times: user 160 ms, sys: 0 ns, total: 160 ms
Wall time: 160 ms
```

In [169]:
```
# Get 50 most frequent words in the training set
frequent_words = pd.Series(' '.join(train.clean_text).split()).value_counts()[:50]
frequent_words.plot.bar(figsize=(15,4))
plt.title('50 most frequent terms in the training set', fontweight="bold")
plt.xlabel('Tokens')
plt.ylabel('Frequency')
plt.show()
```



In [170]:
```
# Plot the number of tokens in cleaned reviews
plt.figure(figsize=(15,4))
plt.hist(train['clean_text'].apply(lambda x:len(x.split())), bins=50, edgecolor='white')
plt.xlabel('Number of tokens', fontsize=12)
plt.ylabel('Number of text',fontsize=12)
plt.title('Distribution of token counts in text', fontsize=14)
plt.show()
```



In [171]:
```
train.head()
```

Out[171]:

	Rating	Reviews	length_of_text	clean_text
0	4.5	["The first playthrough of elden ring is one o...	255	first playthrough elden ring one best eperienc...
1	4.3	['convinced this is a roguelike for people who...	343	convinced roguelike people like genre art tech...
2	4.4	['This game is the game (that is not CS:GO) th...	327	game game csgo played ever played game hours r...
3	4.2	['soundtrack is tied for #1 with nier automata...	137	soundtrack tied nier automata super charming s...
4	4.4	["this games worldbuilding is incredible, with...	643	games worldbuilding incredible amazing soundtr...

In [175]:
```
train.head()
```

Out[175]:

	Rating	Reviews	clean_text
0	4.5	["The first playthrough of elden ring is one o...	first playthrough elden ring one best eperienc...
1	4.3	['convinced this is a roguelike for people who...	convinced roguelike people like genre art tech...
2	4.4	['This game is the game (that is not CS:GO) th...	game game csgo played ever played game hours r...
3	4.2	['soundtrack is tied for #1 with nier automata...	soundtrack tied nier automata super charming s...
4	4.4	["this games worldbuilding is incredible, with...	games worldbuilding incredible amazing soundtr...

Test Set

In [176]:
```
from sklearn.model_selection import train_test_split

# Let's say we want to split the data in 80:10:10 for train:valid:test dataset
train_size=0.8

X = train["clean_text"].values
y = train["Rating"].values

# In the first step we will split the data in training and remaining dataset
X_train, X_rem, y_train, y_rem = train_test_split(X,y, train_size=0.8,random_state=0)

# Now since we want the valid and test size to be equal (10% each of overall data).
# we have to define valid_size=0.5 (that is 50% of remaining data)
```

```
test_size = 0.5
X_valid, X_test, y_valid, y_test = train_test_split(X_rem,y_rem, test_size=0.5,random_state=0)

print(X_train.shape), print(y_train.shape)
print(X_valid.shape), print(y_valid.shape)
print(X_test.shape), print(y_test.shape)
```

```
(892,)
(892,)
(111,)
(111,)
(112,)
(112,)
```

Out[176]:

```
(None, None)
```

In [177]:

```
X_train.shape
```

Out[177]:

```
(892,)
```

Model part

In [178]:

```
import warnings
warnings.filterwarnings("ignore")

import tensorflow as tf

try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
    tf.config.experimental_connect_to_cluster(tpu)
    tf.tpu.experimental.initialize_tpu_system(tpu)

    strategy = tf.distribute.experimental.TPUStrategy
except ValueError:
    strategy = tf.distribute.get_strategy()
    print('Number of replicas:', strategy.num_replicas_in_sync)
try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolver() # TPU detection
except ValueError:
    tpu = None
    gpus = tf.config.experimental.list_logical_devices("GPU")
if tpu:
    tf.tpu.experimental.initialize_tpu_system(tpu)
    strategy = tf.distribute.experimental.TPUStrategy(tpu,)
    print('Running on TPU ', tpu.cluster_spec().as_dict()['worker'])
elif len(gpus) > 1:
    strategy = tf.distribute.MirroredStrategy([gpu.name for gpu in gpus])
    print('Running on multiple GPUs ', [gpu.name for gpu in gpus])
elif len(gpus) == 1:
    strategy = tf.distribute.get_strategy()
    print('Running on single GPU ', gpus[0].name)
else:
    strategy = tf.distribute.get_strategy()
    print('Running on CPU')
print("Number of accelerators: ", strategy.num_replicas_in_sync)
```

```
Number of replicas: 1
Running on CPU
Number of accelerators:  1
```

In [179]:

```
%%time

from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

# Tokenize and pad the sequences
tokenizer = Tokenizer(num_words=20000)
tokenizer.fit_on_texts(X_train)

max_seq_length = 250

X_tr_seq = tokenizer.texts_to_sequences(X_train)
X_tr_seq = pad_sequences(X_tr_seq, maxlen=max_seq_length)

X_va_seq = tokenizer.texts_to_sequences(X_valid)
X_va_seq = pad_sequences(X_va_seq, maxlen=max_seq_length)

X_te_seq = tokenizer.texts_to_sequences(X_test)
X_te_seq = pad_sequences(X_te_seq, maxlen=max_seq_length)
```

```
CPU times: user 305 ms, sys: 959 µs, total: 306 ms
Wall time: 306 ms
```

In [180]:

```
X_tr_seq.size
```

Out[180]:

```
223000
```

In [187]:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, Conv1D, MaxPooling1D, LSTM, Bidirectional, Dense, Bat
chNormalization, Dropout
from tensorflow.keras.optimizers import Adam

with strategy.scope():
    model = Sequential()
    model.add(Embedding(input_dim=20000, output_dim=128, input_length=250))

    model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
    model.add(MaxPooling1D(pool_size=2))

    model.add(Bidirectional(LSTM(units=64, return_sequences=True)))
    model.add(BatchNormalization())
    model.add(Dropout(0.2))

    model.add(Bidirectional(LSTM(units=128)))
    model.add(BatchNormalization())
```

```
    model.add(Dropout(0.2))

    model.add(Dense(units=32, activation='relu'))
    model.add(BatchNormalization())
    model.add(Dropout(0.2))

    model.add(Dense(units=6, activation='softmax'))

    model.compile(optimizer=Adam(learning_rate = 1e-4),
                  loss = 'sparse_categorical_crossentropy',
                  metrics=['acc']
                 )

model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_6 (Embedding)     (None, 250, 128)          2560000

 conv1d_6 (Conv1D)           (None, 248, 64)           24640

 max_pooling1d_6 (MaxPooling  (None, 124, 64)          0
 1D)

 bidirectional_12 (Bidirecti  (None, 124, 128)         66048
 onal)

 batch_normalization_18 (Bat  (None, 124, 128)         512
 chNormalization)

 dropout_18 (Dropout)        (None, 124, 128)          0

 bidirectional_13 (Bidirecti  (None, 256)              263168
 onal)

 batch_normalization_19 (Bat  (None, 256)              1024
 chNormalization)

 dropout_19 (Dropout)        (None, 256)               0

 dense_12 (Dense)            (None, 32)                8224

 batch_normalization_20 (Bat  (None, 32)               128
 chNormalization)

 dropout_20 (Dropout)        (None, 32)                0

 dense_13 (Dense)            (None, 6)                 198

=================================================================
Total params: 2,923,942
Trainable params: 2,923,110
Non-trainable params: 832
_________________________________________________________________
```

In [182]:
```
X_tr_seq.size
```

Out[182]:
```
223000
```

In [183]:
```
y_train.size
```

Out[183]:
```
892
```

In [184]:
```
from tensorflow.keras.callbacks import EarlyStopping

# Creates 'EarlyStopping' callback
earlystopping_cb = EarlyStopping(patience=4, restore_best_weights=True)
```

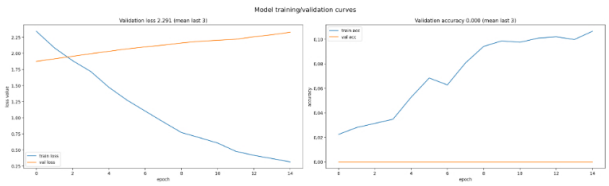
In [192]:
```
%%time

history = model.fit(X_tr_seq,
                    y_train,
                    validation_data=(X_va_seq, y_valid),
                    batch_size=64,
                    epochs=15,
                    verbose=1,
                   )
```

```
Epoch 1/15
14/14 [==============================] - 7s 486ms/step - loss: 2.3429 - acc: 0.0224 - val_loss:
1.8758 - val_acc: 0.0000e+00
Epoch 2/15
14/14 [==============================] - 6s 446ms/step - loss: 2.0847 - acc: 0.0280 - val_loss:
1.9148 - val_acc: 0.0000e+00
Epoch 3/15
14/14 [==============================] - 6s 442ms/step - loss: 1.8864 - acc: 0.0314 - val_loss:
1.9547 - val_acc: 0.0000e+00
Epoch 4/15
14/14 [==============================] - 6s 447ms/step - loss: 1.7182 - acc: 0.0348 - val_loss:
1.9948 - val_acc: 0.0000e+00
Epoch 5/15
14/14 [==============================] - 6s 442ms/step - loss: 1.4730 - acc: 0.0527 - val_loss:
2.0315 - val_acc: 0.0000e+00
Epoch 6/15
14/14 [==============================] - 6s 439ms/step - loss: 1.2736 - acc: 0.0684 - val_loss:
2.0697 - val_acc: 0.0000e+00
Epoch 7/15
14/14 [==============================] - 6s 432ms/step - loss: 1.1038 - acc: 0.0628 - val_loss:
2.1002 - val_acc: 0.0000e+00
Epoch 8/15
14/14 [==============================] - 6s 437ms/step - loss: 0.9349 - acc: 0.0807 - val_loss:
2.1282 - val_acc: 0.0000e+00
Epoch 9/15
14/14 [==============================] - 6s 443ms/step - loss: 0.7707 - acc: 0.0942 - val_loss:
2.1607 - val_acc: 0.0000e+00
Epoch 10/15
14/14 [==============================] - 6s 459ms/step - loss: 0.6907 - acc: 0.0987 - val_loss:
2.1877 - val_acc: 0.0000e+00
```

```
Epoch 11/15
14/14 [==============================] - 6s 443ms/step - loss: 0.6068 - acc: 0.0975 - val_loss:
2.2034 - val_acc: 0.0000e+00
Epoch 12/15
14/14 [==============================] - 6s 437ms/step - loss: 0.4787 - acc: 0.1009 - val_loss:
2.2192 - val_acc: 0.0000e+00
Epoch 13/15
14/14 [==============================] - 6s 439ms/step - loss: 0.4161 - acc: 0.1020 - val_loss:
2.2574 - val_acc: 0.0000e+00
Epoch 14/15
14/14 [==============================] - 6s 441ms/step - loss: 0.3659 - acc: 0.0998 - val_loss:
2.2889 - val_acc: 0.0000e+00
Epoch 15/15
14/14 [==============================] - 6s 462ms/step - loss: 0.3146 - acc: 0.1065 - val_loss:
2.3282 - val_acc: 0.0000e+00
CPU times: user 5min 4s, sys: 16 s, total: 5min 20s
Wall time: 2min 22s
```

In [193]:

```
# Create two plots: one for the loss value, one for the accuracy
fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))

plt.suptitle('Model training/validation curves', size=15)

# Plot loss values
ax1.plot(history.history["loss"], label="train loss")
ax1.plot(history.history["val_loss"], label="val loss")
ax1.set_title(
    "Validation loss {:.3f} (mean last 3)".format(
        np.mean(history.history["val_loss"][-3:])  # last three values
    )
)
ax1.set_xlabel("epoch")
ax1.set_ylabel("loss value")
ax1.legend()

# Plot accuracy values
ax2.plot(history.history["acc"], label="train acc")
ax2.plot(history.history["val_acc"], label="val acc")
ax2.set_title(
    "Validation accuracy {:.3f} (mean last 3)".format(
        np.mean(history.history["val_acc"][-3:])  # last three values
    )
)
ax2.set_xlabel("epoch")
ax2.set_ylabel("accuracy")
ax2.legend()
plt.tight_layout()
plt.show()
```



In [194]:

```
test_loss, test_acc = model.evaluate(X_va_seq,y_valid)

print('Validation loss:', test_loss)
print('Validation accuracy:', test_acc)
```

```
4/4 [==============================] - 0s 79ms/step - loss: 2.3282 - acc: 0.0000e+00
Validation loss: 2.328200340270996
Validation accuracy: 0.0
```

In [195]:

```
# Compute validation set predictions
pred = [np.argmax(i) for i in model.predict(X_va_seq)]
```

```
4/4 [==============================] - 2s 107ms/step
```

In [ ]:

Continue exploring

Input
1 file →

Output
0 files →

Logs
3.9 second run - successful →

Comments
0 comments →