In [ ]:
```
!pip -q install streamlit
!pip -q install pyngrok
```

In [2]:
```
from pyngrok import ngrok
from google.colab import drive,files
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [3]:
```
api_token = files.upload()
!mkdir ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
!pip install --upgrade --force-reinstall --no-deps kaggle

!kaggle datasets download -d olistbr/brazilian-ecommerce
!unzip '/content/brazilian-ecommerce.zip'
```

Choose Files No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving kaggle.json to kaggle.json
Collecting kaggle
  Downloading https://files.pythonhosted.org/packages/3a/e7/3bac01547d2
ed3d308ac92a0878fbdb0ed0f3d41fb1906c319ccbba1bfbc/kaggle-1.5.12.tar.gz
(58kB)
     |████████████████████████████████| 61kB 3.4MB/s
Building wheels for collected packages: kaggle
  Building wheel for kaggle (setup.py) ... done
  Created wheel for kaggle: filename=kaggle-1.5.12-cp37-none-any.whl si
ze=73053 sha256=6dfb1e2133773ca102a972d73a8cd2544fdfb2d80540ee38a6a3cfe
eb2f1f2b9
  Stored in directory: /root/.cache/pip/wheels/a1/6a/26/d30b7499ff85a4a
4593377a87ecf55f7d08af42f0de9b60303
```

```
Successfully built kaggle
Installing collected packages: kaggle
  Found existing installation: kaggle 1.5.12
    Uninstalling kaggle-1.5.12:
      Successfully uninstalled kaggle-1.5.12
Successfully installed kaggle-1.5.12
Downloading brazilian-ecommerce.zip to /content
 40% 17.0M/42.7M [00:00<00:00, 53.9MB/s]
100% 42.7M/42.7M [00:00<00:00, 116MB/s]
Archive:  /content/brazilian-ecommerce.zip
  inflating: olist_customers_dataset.csv
  inflating: olist_geolocation_dataset.csv
  inflating: olist_order_items_dataset.csv
  inflating: olist_order_payments_dataset.csv
  inflating: olist_order_reviews_dataset.csv
  inflating: olist_orders_dataset.csv
  inflating: olist_products_dataset.csv
  inflating: olist_sellers_dataset.csv
  inflating: product_category_name_translation.csv
```

In [108]:

```
%%writefile app.py

import pandas as pd
import numpy as np
import pickle
import re
import streamlit as st
from sklearn.preprocessing import OneHotEncoder, OrdinalEncoder, Standa
rdScaler
from sklearn.decomposition import TruncatedSVD
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from xgboost import XGBClassifier

all_stopwords = pickle.load(open('/content/drive/MyDrive/Olist/final_mo
dels/stop_words.pkl','rb'))

def process_texts(texts):
    processed_text = []
    dates = '^([0]?[1-9]|[1|2][0-9]|[3][0|1])[./-]([0]?[1-9]|[1][0-2])
[./-]([0-9]{4}|[0-9]{2})$'
    for text in texts:
        text = re.sub(r'\r\n|\r|\n', ' ', text)
        text = re.sub(r'^https?:\/\/.*[\r\n]*', ' ', text)
        text = re.sub(dates, ' ', text)
        text = re.sub('[ \t]+$', '', text)
        text = re.sub('\W', ' ', text)
        text = re.sub('[0-9]+', ' ', text)
        text = re.sub('\s+', ' ', text)
        text = ' '.join(e for e in text.split() if e.lower() not in all
_stopwords)
        processed_text.append(text.lower().strip())
    return processed_text

def test_response(test,dict_frame,dict_f1,dict_f2):
  t_state_0, t_state_1 = [],[]
  for i in range(len(test)):
    if dict_frame.get(test[i]):
      t_state_0.append(dict_f1.get(test[i],0)/dict_frame.get(test[i]))
      t_state_1.append(dict_f2.get(test[i],0)/dict_frame.get(test[i]))
    else:
      t_state_0.append(0.5)
      t_state_1.append(0.5)
  df4 = pd.DataFrame({'State_0':t_state_0, 'State_1':t_state_1})
  return df4.to_numpy()

uploaded_file = st.file_uploader("Choose a file")
if uploaded_file is not None:

  data_point = pd.read_csv(uploaded_file, index_col='Unnamed: 0')
  emb_text = pickle.load(open('/content/drive/MyDrive/Olist/final_model
s/test_embedded_first_idx.pkl','rb'))
  X_train = pickle.load(open('/content/drive/MyDrive/Olist/final_model
s/X_train.pkl','rb'))
  y_train = pickle.load(open('/content/drive/MyDrive/Olist/final_model
s/y_train.pkl','rb'))
  xgb_model = pickle.load(open('/content/drive/MyDrive/Olist/final_mode
ls/xgb_original.pkl','rb'))
```

```
  prod_cat_dict_frame = pickle.load(open('/content/drive/MyDrive/Olist/
final_models/prod_cat_dict_frame.pkl','rb'))
  prod_cat_dict_f1 = pickle.load(open('/content/drive/MyDrive/Olist/fin
al_models/prod_cat_dict_f1.pkl','rb'))
  prod_cat_dict_f2 = pickle.load(open('/content/drive/MyDrive/Olist/fin
al_models/prod_cat_dict_f2.pkl','rb'))

  pay_seq_dict_frame = pickle.load(open('/content/drive/MyDrive/Olist/f
inal_models/pay_seq_dict_frame.pkl','rb'))
  pay_seq_dict_f1 = pickle.load(open('/content/drive/MyDrive/Olist/fina
l_models/pay_seq_dict_f1.pkl','rb'))
  pay_seq_dict_f2 = pickle.load(open('/content/drive/MyDrive/Olist/fina
l_models/pay_seq_dict_f2.pkl','rb'))

  strn = StandardScaler()
  strn.fit(X_train[['price','freight_value','product_photos_qty','produ
ct_weight_g', 'product_length_cm',
        'product_height_cm', 'product_width_cm', 'payment_value','purch
ase-delivery difference','estimated-actual delivery difference','purcha
se_delivery_diff_per_price']])
  X_test_strn = strn.transform(data_point.loc[['price','freight_value',
'product_photos_qty','product_weight_g', 'product_length_cm',
        'product_height_cm', 'product_width_cm', 'payment_value','purch
ase-delivery difference','estimated-actual delivery difference','purcha
se_delivery_diff_per_price']].T)

  X_test_resp_prod_cat = test_response(data_point.loc['product_category
_name'].values,prod_cat_dict_frame,prod_cat_dict_f1,prod_cat_dict_f2)

  ohe_order_item = OneHotEncoder()
  ohe_order_item.fit(X_train['order_item_id'].values.reshape(-1,1))
  X_test_order_item = ohe_order_item.transform(data_point.loc['order_it
em_id'].values.astype(int).reshape(-1,1)).toarray()

  X_test_resp_payment_seq = test_response(data_point.loc['payment_seque
ntial'].values,pay_seq_dict_frame,pay_seq_dict_f1,pay_seq_dict_f2)

  ohe_payment_type = OneHotEncoder()
  ohe_payment_type.fit(X_train['payment_type'].values.reshape(-1,1))
  X_test_payment_type = ohe_payment_type.transform(data_point.loc['paym
ent_type'].values.reshape(-1,1)).toarray()

  enc_price = OrdinalEncoder()
  enc_price.fit(X_train['price_category'].values.reshape(-1,1))
  enc_price.categories_ = [np.array([ 'cheap', 'affordable', 'expensiv
e'], dtype=object)]
  X_test_cat_price = enc_price.transform(data_point.loc['price_categor
y'].values.reshape(-1,1))

  X_train_comment_preprocess = process_texts(X_train['review_comment_me
ssage'])
  X_test_comment_preprocess = process_texts(data_point.loc['review_comm
ent_message'])
  data_point.loc['embedded_review_comment_message'] = [pickle.load(open
('/content/drive/MyDrive/Olist/final_models/X_test_embedded_review_comm
ent_message.pkl','rb')).loc[int(data_point.loc['Unnamed: 0.1'].values[0
])]]

  tok = Tokenizer()
  tok.fit_on_texts(X_train_comment_preprocess)
  X_test_text_input = pad_sequences(tok.texts_to_sequences(X_test_comme
nt_preprocess), padding='post')

  data_point.loc['review_availability'] = 1 if data_point.loc['review_c
omment_message'].values[0] != 'indisponível' else 0
  X_test_final = np.concatenate((X_test_strn,X_test_resp_prod_cat, X_te
st_order_item,
        X_test_resp_payment_seq,X_test_payment_type,X_test_cat_price,da
ta_point.loc['review_availability'].values.reshape(-1,1),
        np.vstack(data_point.loc['embedded_review_comment_message'].val
ues)), axis=1)
  sentiment = xgb_model.predict(X_test_final)[0]
  if sentiment == 1:
    st.write('The review is positive!')
  else:
    st.write('The review is negative!')
```

```
Overwriting app.py
```

In [106]:
```
!ngrok authtoken 1s9EWYEkYGU9J6chwq2Z0Hucw5R_2iBbA53YyhmegQY3b67nm
```

```
Authtoken saved to configuration file: /root/.ngrok2/ngrok.yml
```

In [109]:
```
public_url = ngrok.connect(port='80')
print (public_url)
!streamlit run --server.port 80 app.py >/dev/null
```

```
NgrokTunnel: "http://11989861b782.ngrok.io" -> "http://localhost:80"
2021-05-08 16:53:37.147 An update to the [server] config option section
was detected. To have these changes be reflected, please restart stream
lit.
2021-05-08 16:53:37.960888: I tensorflow/stream_executor/platform/defau
lt/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.1
1.0
```

In [ ]: