import streamlit as st

import pandas as pd

import numpy as np

import plotly.express as px

import matplotlib.pyplot as plt

st.title("Economic Indicators of countries\*")

st.sidebar.title("Economic Indicators of countries")

st.markdown(" \*This application is a Streamlit dashboard to show and analyse the economic indicators of Australia, Austria, Brazil, Germany, Russia, USA")

st.sidebar.markdown(" This application is a Streamlit dashboard to show and analyse the economic indicators of certain countries")

#DATA\_URL =("/home/rhyme/Desktop/Project/Countries\_Short.csv")

#DATA\_URL =("/Users/usr001/Desktop/1Streamlit/Countries\_Short.csv")

#@st.cache(persist=True)

#def load\_data():

 #   data = pd.read\_csv(DATA\_URL)

 #   data['Country'] = pd.to\_datetime(data['Country'])

 #   return data

#data = load\_data()

countries\_df = pd.read\_csv('Countries\_Short.csv')

st.write(countries\_df)

st.sidebar.subheader("Show random country's information")

#random\_tweeet = st.sidebar.radio('Sentiment', ('positive', 'neutral', 'negative'))

random\_result = st.button("Get the info")

st.write(random\_result)

st.sidebar.markdown(data.query('airline\_sentiment == @random\_tweet')[["text"]].sample(n=1).iat[0,0])

st.sidebar.markdown("### Number of tweets by sentiment")

select = st.sidebar.selectbox('Visualisation type', ['Histogram', 'Pie chart'], key='1')

sentiment\_count = data['airline\_sentiment'].value\_counts()

sentiment\_count = pd.DataFrame({'Sentiment':sentiment\_count.index, 'Tweets':sentiment\_count.values})

if not st.sidebar.checkbox("Hide", True):

    st.markdown("### Number of Tweets by sentiment")

    if select == "Histogram":

        fig = px.bar(sentiment\_count, x='Sentiment', y='Tweets', color='Tweets', height = 500)

        st.plotly\_chart(fig)

    else:

        fig = px.pie(sentiment\_count, values='Tweets', names='Sentiment')

        st.plotly\_chart(fig)

st.sidebar.subheader("When and where are users tweeting from?")

hour = st.sidebar.slider("Hour of day", 0, 23)

#hour = st.sidebar.number\_input("Hour of day",min\_value=1, max\_value=24)

modified\_data = data[data['tweet\_created'].dt.hour == hour]

if not st.sidebar.checkbox("Close", True, key='1'):

    st.markdown("### Tweets locations based on the time of day")

    st.markdown("%i tweets between %i:00 and %i:00" % (len(modified\_data), hour, (hour+1)%24))

    st.map(modified\_data)

    if st.sidebar.checkbox("Show raw data", False):

        st.write(modified\_data)

st.sidebar.subheader("Breakdown irline tweets by sentiment")

choice = st.sidebar.multiselect('Pick airlines', ('Us Airways', 'United', 'American', 'Southwest', 'Delta', 'Virgin America'), key='0')

if len(choice) > 0:

    choice\_data = data[data.airline.isin(choice)]

    fig\_choice = px.histogram(choice\_data, x='airline', y='airline\_sentiment', histfunc='count', color='airline\_sentiment',

    facet\_col='airline\_sentiment', labels={'airline\_sentiment':'tweets'}, height=600, width=800)

    st.plotly\_chart(fig\_choice)

# st.sidebar.header("Word Cloud")

# word\_sentiment = st.sidebar.radio('Display word cloud for what sentiment?', ('positive', 'neutral', 'negative'))

#

# if not st.sidebar.checkbox("Close", True, key='3'):

#     st.header('Word cloud for %s sentiment' % (word\_sentiment))

#     df = data[data['airline\_sentiment']==word\_sentiment]

#     words = ' '.join(df['text'])

#     processed\_words = ' '.join([word for word in words.split() if 'http' not in word and not word.startswith('@') and word != 'RT'])

#     wordcloud = WordCloud(stopwords=STOPWORDS, background\_color='white', height=640, width=800).generate(processed\_words)

#     plt.imshow(wordcloud)

#     plt.xticks([])

#     plt.yticks([])

#     st.pyplot()

import streamlit as st

import pandas as pd

import numpy as np

import plotly.express as px

import seaborn as sns

import matplotlib.pyplot as plt

st.title("Economic Indicators of countries\*")

st.sidebar.title("Economic Indicators of countries")

st.markdown(" \*This application is a Streamlit dashboard to show and analyse the economic indicators of Australia, Austria, Brazil, Germany, Russia, USA")

st.sidebar.markdown(" This application is a Streamlit dashboard to show and analyse the economic indicators of certain countries")

countries\_df = pd.read\_csv('info\_countries.csv')

st.sidebar.subheader("Show random country's information")

#random\_tweeet = st.sidebar.radio('Sentiment', ('positive', 'neutral', 'negative'))

#last\_data = st.sidebar.button("Get the last information of all countries")

if st.sidebar.button("Get the last information of all countries"):

    st.write(countries\_df)

#data = load\_data()

# countries\_df = pd.read\_csv('info\_countries.csv')

# st.write(countries\_df)

#st.sidebar.markdown(data.query('airline\_sentiment == @random\_tweet')[["text"]].sample(n=1).iat[0,0])

#Australia

australia\_df = pd.read\_csv('Australia1.csv')

#st.write(australia\_df)

#Austria

austria\_df = pd.read\_csv('Austria1.csv')

#st.write(austria\_df)

#Brazil

brazil\_df = pd.read\_csv('Brazil1.csv')

#st.write(brazil\_df)

#Germany

germany\_df = pd.read\_csv('Germany1.csv')

#st.write(germany\_df)

#Russia

russia\_df = pd.read\_csv('Russia1.csv')

#st.write(russia\_df)

#USA

usa\_df = pd.read\_csv('USA1.csv')

#st.write(USA\_df)

# st.write(australia\_df)

# st.write(austria\_df)

# st.write(brazil\_df)

# st.write(germany\_df)

# st.write(russia\_df)

# st.write(USA\_df)

# st.sidebar.subheader("Show random country's information")

# st.sidebar.subheader("What timeperiod are you interested in?")

# year = st.sidebar.slider("Period of time", 1999, 2021)

# year = st.sidebar.slider("Period od time",

# value=[1999,2021])

st.sidebar.subheader("What timeperiod are you interested in?")

year = st.sidebar.slider(

     'Select a range of years',

     1999, 2021, (2001, 2005))

st.sidebar.write('Chosen:', year)

st.sidebar.subheader("Select countries")

choice = st.sidebar.multiselect('Pick countries', ('Australia', 'Austria', 'Brazil', 'Germany', 'Russia', 'USA'), key='0')

if not st.sidebar.checkbox("Close", True, key='1'):

    st.markdown("### Tweets locations based on the time of day")

    st.markdown("%i tweets between %i:00 and %i:00" % (len(modified\_data), hour, (hour+1)%24))

    st.map(modified\_data)

    if st.sidebar.checkbox("Show raw data", False):

        st.write(modified\_data)

#st.sidebar.subheader("What timeperiod are you interested in?")

#year = st.sidebar.number\_input("Years",min\_value=1, max\_value=24)

# modified\_data = data[data['tweet\_created'].dt.hour == hour]

# if not st.sidebar.checkbox("Close", True, key='1'):

#     st.markdown("### Tweets locations based on the time of day")

#     st.markdown("%i tweets between %i:00 and %i:00" % (len(modified\_data), hour, (hour+1)%24))

#     st.map(modified\_data)

#     if st.sidebar.checkbox("Show raw data", False):

#         st.write(modified\_data)

# st.sidebar.markdown("### Number of tweets by sentiment")

# select = st.sidebar.selectbox('Visualisation type', ['Histogram', 'Pie chart'], key='1')

# sentiment\_count = data['airline\_sentiment'].value\_counts()

# sentiment\_count = pd.DataFrame({'Sentiment':sentiment\_count.index, 'Tweets':sentiment\_count.values})

# if not st.sidebar.checkbox("Hide", True):

#     st.markdown("### Number of Tweets by sentiment")

#     if select == "Histogram":

#         fig = px.bar(sentiment\_count, x='Sentiment', y='Tweets', color='Tweets', height = 500)

#         st.plotly\_chart(fig)

#     else:

#         fig = px.pie(sentiment\_count, values='Tweets', names='Sentiment')

#         st.plotly\_chart(fig)

# st.sidebar.subheader("When and where are users tweeting from?")

# hour = st.sidebar.slider("Hour of day", 0, 23)

# #hour = st.sidebar.number\_input("Hour of day",min\_value=1, max\_value=24)

# modified\_data = data[data['tweet\_created'].dt.hour == hour]

# if not st.sidebar.checkbox("Close", True, key='1'):

#     st.markdown("### Tweets locations based on the time of day")

#     st.markdown("%i tweets between %i:00 and %i:00" % (len(modified\_data), hour, (hour+1)%24))

#     st.map(modified\_data)

#     if st.sidebar.checkbox("Show raw data", False):

#         st.write(modified\_data)

# st.sidebar.subheader("Breakdown irline tweets by sentiment")

# choice = st.sidebar.multiselect('Pick airlines', ('Us Airways', 'United', 'American', 'Southwest', 'Delta', 'Virgin America'), key='0')

# if len(choice) > 0:

#     choice\_data = data[data.airline.isin(choice)]

#     fig\_choice = px.histogram(choice\_data, x='airline', y='airline\_sentiment', histfunc='count', color='airline\_sentiment',

#     facet\_col='airline\_sentiment', labels={'airline\_sentiment':'tweets'}, height=600, width=800)

#     st.plotly\_chart(fig\_choice)

# st.sidebar.header("Word Cloud")

# word\_sentiment = st.sidebar.radio('Display word cloud for what sentiment?', ('positive', 'neutral', 'negative'))

# if not st.sidebar.checkbox("Close", True, key='3'):

#     st.header('Word cloud for %s sentiment' % (word\_sentiment))

#     df = data[data['airline\_sentiment']==word\_sentiment]

#     words = ' '.join(df['text'])

#     processed\_words = ' '.join([word for word in words.split() if 'http' not in word and not word.startswith('@') and word != 'RT'])

#     wordcloud = WordCloud(stopwords=STOPWORDS, background\_color='white', height=640, width=800).generate(processed\_words)

#     plt.imshow(wordcloud)

#     plt.xticks([])

#     plt.yticks([])

#     st.pyplot()

<https://www.kaggle.com/code/hgultekin/eu-inflation-rates-visualization-2008-2019>

Python сегодня можно совершенно уверенно назвать лучшим языком программирования. Python в последние годы теснит другие языки программирования и завоевывает рекордную популярность во всем мире. У языка Python есть обширное комьюнити, им владеет огромное количество людей по всему миру, и поэтому при возникновении сложностей или ошибок можно обратиться за помощью.  Это почти сочетание несочетаемого: язык Python подходит для веб-разработки, анализа больших данных и машинного обучения, но при этом он интуитивно понятный, легкий и дружелюбный для новичка.