Week 4: Deployment on Flask

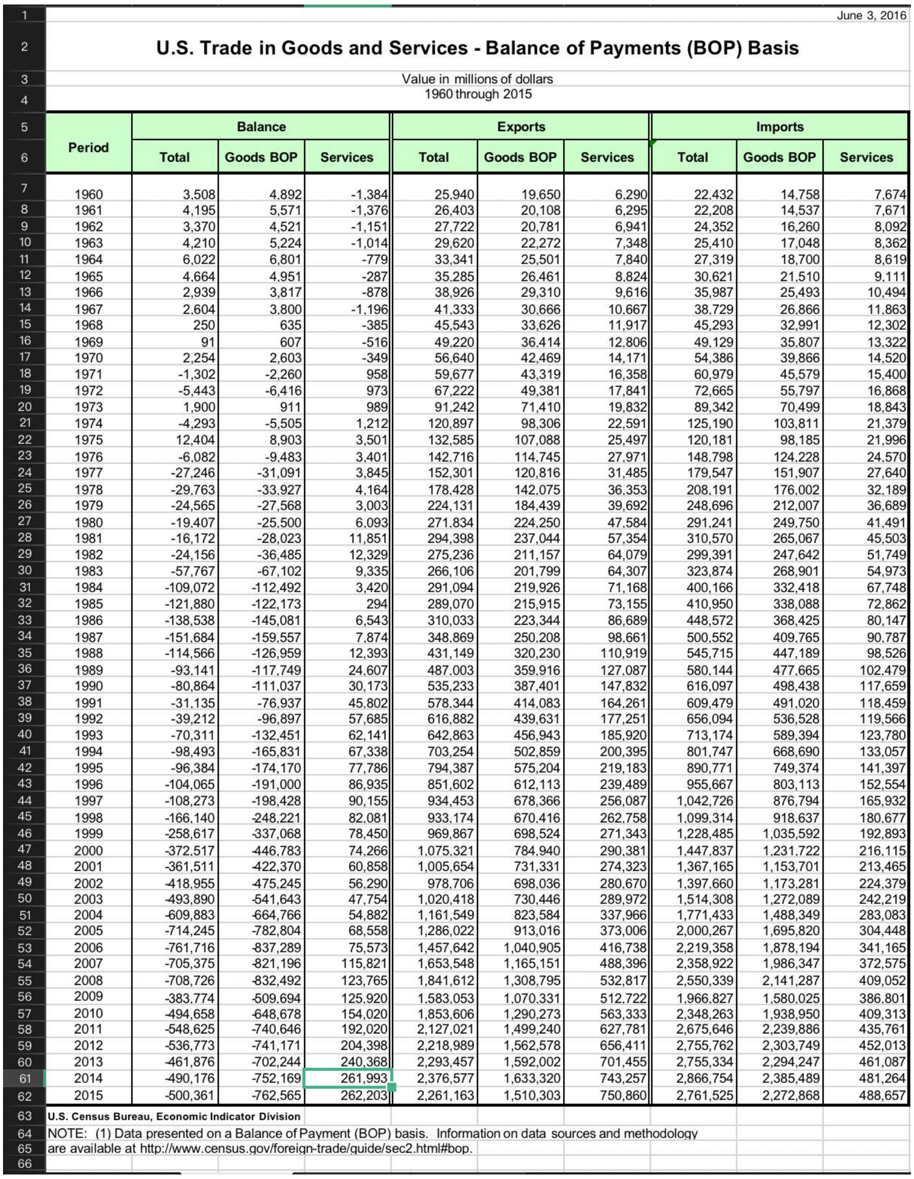
Name: Zhan Shi

Batch code: 5068942

Submission data: Aug 27, 2022

Submitted to: Data Glacier Virtual Internship Canvas Site (link provided below) https://canvas.instructure.com/courses/5068942/assignments/31350145?module_item_id=69454418

Introduction of data and model:



June 3, 2016

U.S. Trade in Goods and Services - Balance of Payments (BOP) Basis

Value in millions of dollars
1960 through 2015

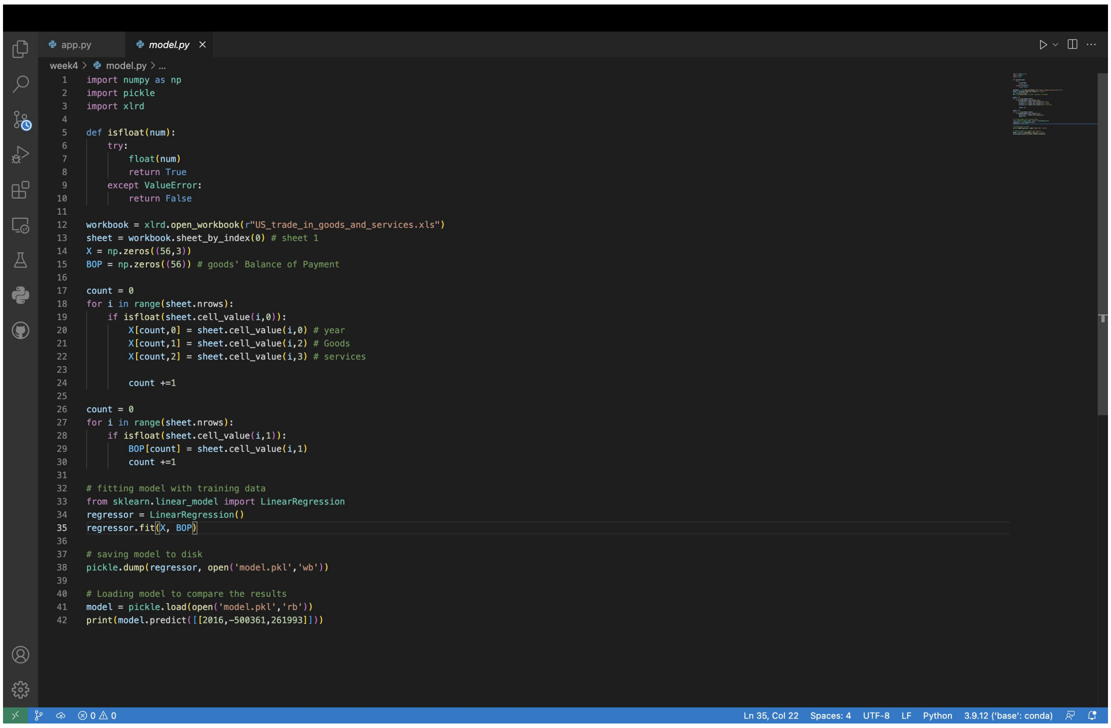
Period	Balance			Exports			Imports		
	Total	Goods BOP	Services	Total	Goods BOP	Services	Total	Goods BOP	Services
1960	3,508	4,892	-1,384	25,940	19,650	6,290	22,432	14,758	7,674
1961	4,195	5,571	-1,376	26,403	20,108	6,295	22,208	14,537	7,671
1962	3,370	4,521	-1,151	27,722	20,781	6,941	24,352	16,260	8,092
1963	4,210	5,224	-1,014	29,620	22,272	7,348	25,410	17,048	8,362
1964	6,022	6,801	-779	33,341	25,501	7,840	27,319	18,700	8,619
1965	4,664	4,951	-287	35,285	26,461	8,824	30,621	21,510	9,111
1966	2,939	3,817	-878	38,926	29,310	9,616	35,987	25,493	10,494
1967	2,604	3,800	-1,196	41,333	30,666	10,667	38,729	26,866	11,863
1968	250	635	-385	45,543	33,626	11,917	45,293	32,991	12,302
1969	91	607	-516	49,220	36,414	12,806	49,129	35,807	13,322
1970	2,254	2,603	-349	56,640	42,469	14,171	54,386	39,866	14,520
1971	-1,302	-2,260	958	59,677	43,319	16,358	60,979	45,579	15,400
1972	-5,443	-6,416	973	67,222	49,381	17,841	72,665	55,797	16,868
1973	1,900	911	989	91,242	71,410	19,832	89,342	70,499	18,843
1974	-4,293	-5,505	1,212	120,897	98,306	22,591	125,190	103,811	21,379
1975	12,404	8,903	3,501	132,585	107,088	25,497	120,181	98,185	21,996
1976	-6,082	-9,483	3,401	142,716	114,745	27,971	148,798	124,228	24,570
1977	-27,246	-31,091	3,845	152,301	120,816	31,485	179,547	151,907	27,640
1978	-29,763	-33,927	4,164	178,428	142,075	36,353	208,191	176,002	32,189
1979	-24,565	-27,568	3,003	224,131	184,439	39,692	248,696	212,007	36,689
1980	-19,407	-25,500	6,093	271,834	224,250	47,584	291,241	249,750	41,491
1981	-16,172	-28,023	11,851	294,398	237,044	57,354	310,570	265,067	45,503
1982	-24,156	-36,485	12,329	275,236	211,157	64,079	299,391	247,642	51,749
1983	-57,767	-67,102	9,335	266,106	201,799	64,307	323,874	268,901	54,973
1984	-109,072	-112,492	3,420	291,094	219,926	71,168	400,166	332,418	67,748
1985	-121,880	-122,173	294	289,070	215,915	73,155	410,950	338,088	72,862
1986	-138,538	-145,081	6,543	310,033	223,344	86,689	448,572	368,425	80,147
1987	-151,684	-159,557	7,874	348,869	250,208	98,661	500,552	409,765	90,787
1988	-114,566	-126,959	12,393	431,149	320,230	110,919	545,715	447,189	98,526
1989	-93,141	-117,749	24,607	487,003	359,916	127,087	580,144	477,665	102,479
1990	-80,864	-111,037	30,173	535,233	387,401	147,832	616,097	498,438	117,659
1991	-31,135	-76,937	45,802	578,344	414,083	164,261	609,479	491,020	118,459
1992	-39,212	-96,897	57,685	616,882	439,631	177,251	656,094	536,528	119,566
1993	-70,311	-132,451	62,141	642,863	456,943	185,920	713,174	589,394	123,780
1994	-98,493	-165,831	67,338	703,254	502,859	200,395	801,747	668,690	133,057
1995	-96,384	-174,170	77,786	794,387	575,204	219,183	890,771	749,374	141,397
1996	-104,065	-191,000	86,935	851,602	612,113	239,489	955,667	803,113	152,554
1997	-108,273	-198,428	90,155	934,453	678,366	256,087	1,042,726	876,794	165,932
1998	-166,140	-248,221	82,081	933,174	670,416	262,758	1,099,314	918,637	180,677
1999	-258,617	-337,068	78,450	969,867	698,524	271,343	1,228,485	1,035,592	192,893
2000	-372,517	-446,783	74,266	1,075,321	784,940	290,381	1,447,837	1,231,722	216,115
2001	-361,511	-422,370	60,858	1,005,654	731,331	274,323	1,367,165	1,153,701	213,465
2002	-418,955	-475,245	56,290	978,706	698,036	280,670	1,397,660	1,173,281	224,379
2003	-493,890	-541,643	47,754	1,020,418	730,446	289,972	1,514,308	1,272,089	242,219
2004	-609,883	-664,766	54,882	1,161,549	823,584	337,966	1,771,433	1,488,349	283,083
2005	-714,245	-782,804	68,558	1,286,022	913,016	373,006	2,000,267	1,695,820	304,448
2006	-761,716	-837,289	75,573	1,457,642	1,040,905	416,738	2,219,358	1,878,194	341,165
2007	-705,375	-821,196	115,821	1,653,548	1,165,151	488,396	2,358,922	1,986,347	372,575
2008	-708,726	-832,492	123,765	1,841,612	1,308,795	532,817	2,550,339	2,141,287	409,052
2009	-383,774	-509,694	125,920	1,583,053	1,070,331	512,722	1,966,827	1,580,025	386,801
2010	-494,658	-648,678	154,020	1,853,606	1,290,273	563,333	2,348,263	1,938,950	409,313
2011	-548,625	-740,646	192,020	2,127,021	1,499,240	627,781	2,675,646	2,239,886	435,761
2012	-536,773	-741,171	204,398	2,218,989	1,562,578	656,411	2,755,762	2,303,749	452,013
2013	-461,876	-702,244	240,368	2,293,457	1,592,002	701,455	2,755,334	2,294,247	461,087
2014	-490,176	-752,169	261,993	2,376,577	1,633,320	743,257	2,866,754	2,385,489	481,264
2015	-500,361	-762,565	262,203	2,261,163	1,510,303	750,860	2,761,525	2,272,868	488,657

U.S. Census Bureau, Economic Indicator Division

NOTE: (1) Data presented on a Balance of Payment (BOP) basis. Information on data sources and methodology are available at http://www.census.gov/foreign-trade/guide/sec2.html#bop.

Fig1. U.S. Trade in Goods and Services - Balance of Payments (BOP) Basis

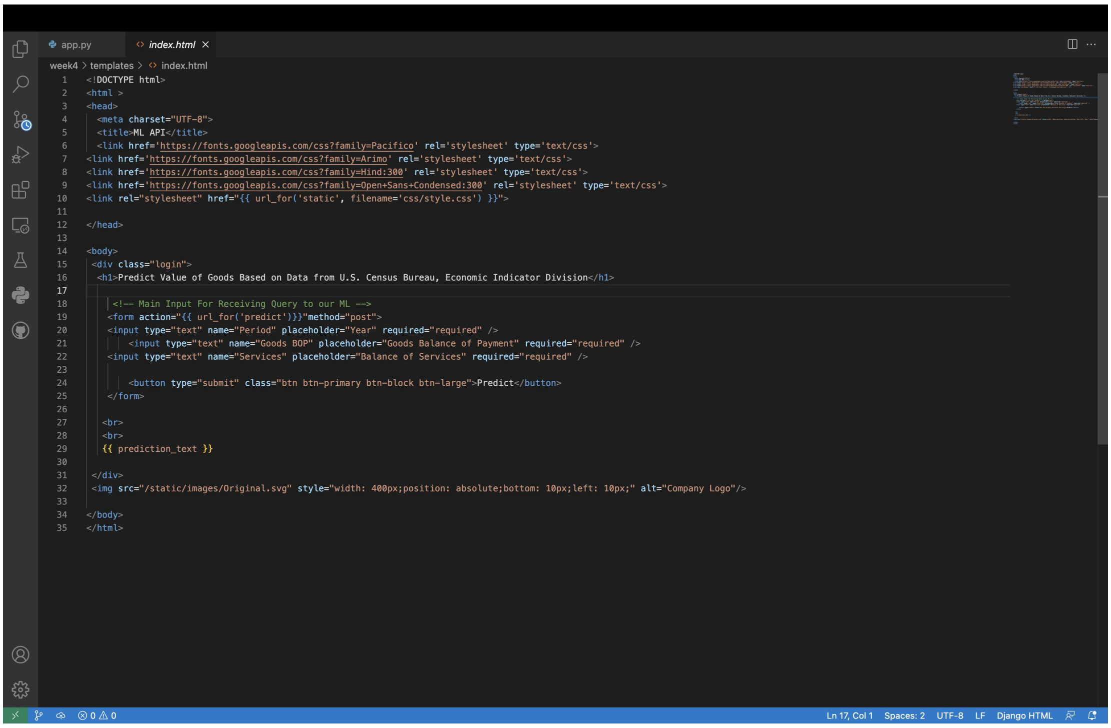
I used values in the Period and Balance features in the U.S. Trade in Goods and Services - Balance of Payments (BOP) Basis to predict the total value of balance.


app.py
model.py
week4 > model.py > ...
Ln 35, Col 22   Spaces: 4   UTF-8   LF   Python   3.9.12 ('base': conda)

```python
import numpy as np
import pickle
import xlrd

def isfloat(num):
    try:
        float(num)
        return True
    except ValueError:
        return False

workbook = xlrd.open_workbook(r"US_trade_in_goods_and_services.xls")
sheet = workbook.sheet_by_index(0) # sheet 1
X = np.zeros((56,3))
BOP = np.zeros((56)) # goods' Balance of Payment

count = 0
for i in range(sheet.nrows):
    if isfloat(sheet.cell_value(i,0)):
        X[count,0] = sheet.cell_value(i,0) # year
        X[count,1] = sheet.cell_value(i,2) # Goods
        X[count,2] = sheet.cell_value(i,3) # services

        count +=1

count = 0
for i in range(sheet.nrows):
    if isfloat(sheet.cell_value(i,1)):
        BOP[count] = sheet.cell_value(i,1)
        count +=1

# fitting model with training data
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X, BOP)

# saving model to disk
pickle.dump(regressor, open('model.pkl','wb'))

# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[2016,-500361,261993]]))
```

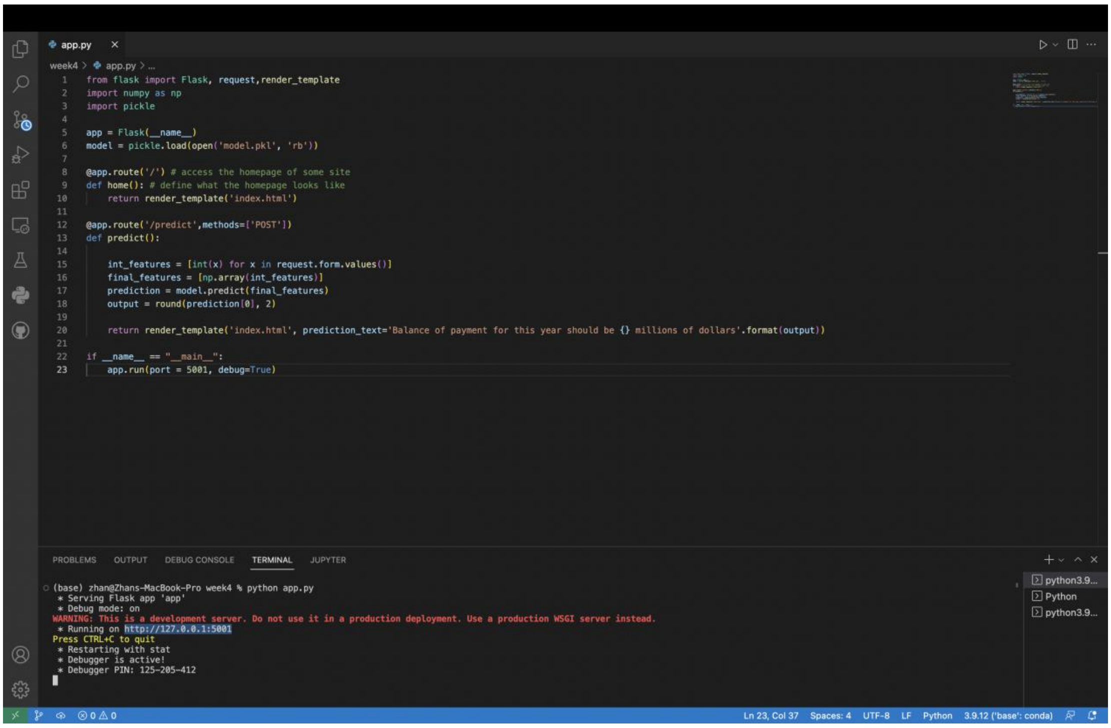
Fig2. Screenshot of model.py

First, I created the model. I used linear regression and fitted between [Year, Goods BOP, Services] and Goods. And then use the input data of three variables to predict total value of balance in the future year.


app.py
index.html
week4 > templates > index.html
Ln 17, Col 1   Spaces: 2   UTF-8   LF   Django HTML

```html
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

</head>

<body>
 <div class="login">
  <h1>Predict Value of Goods Based on Data from U.S. Census Bureau, Economic Indicator Division</h1>

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
    <input type="text" name="Period" placeholder="Year" required="required" />
        <input type="text" name="Goods BOP" placeholder="Goods Balance of Payment" required="required" />
    <input type="text" name="Services" placeholder="Balance of Services" required="required" />

        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
    </form>

   <br>
   <br>
   {{ prediction_text }}

 </div>
 <img src="/static/images/Original.svg" style="width: 400px;position: absolute;bottom: 10px;left: 10px;" alt="Company Logo"/>

</body>
</html>
```

Fig3. Screenshot of index.html



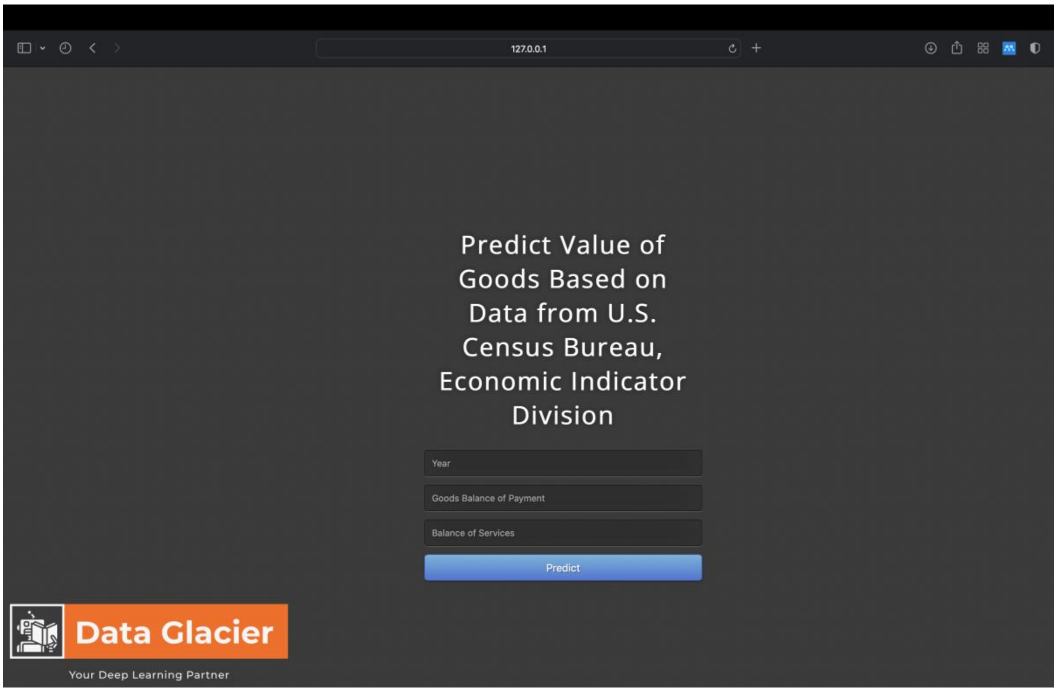
```python
from flask import Flask, request,render_template
import numpy as np
import pickle

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))

@app.route('/') # access the homepage of some site
def home(): # define what the homepage looks like
    return render_template('index.html')

@app.route('/predict',methods=['POST'])
def predict():

    int_features = [int(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)
    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text='Balance of payment for this year should be {} millions of dollars'.format(output))

if __name__ == "__main__":
    app.run(port = 5001, debug=True)
```

```
(base) zhan@Zhans-MacBook-Pro week4 % python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 125-205-412
```

Fig4. Screenshot of app.py

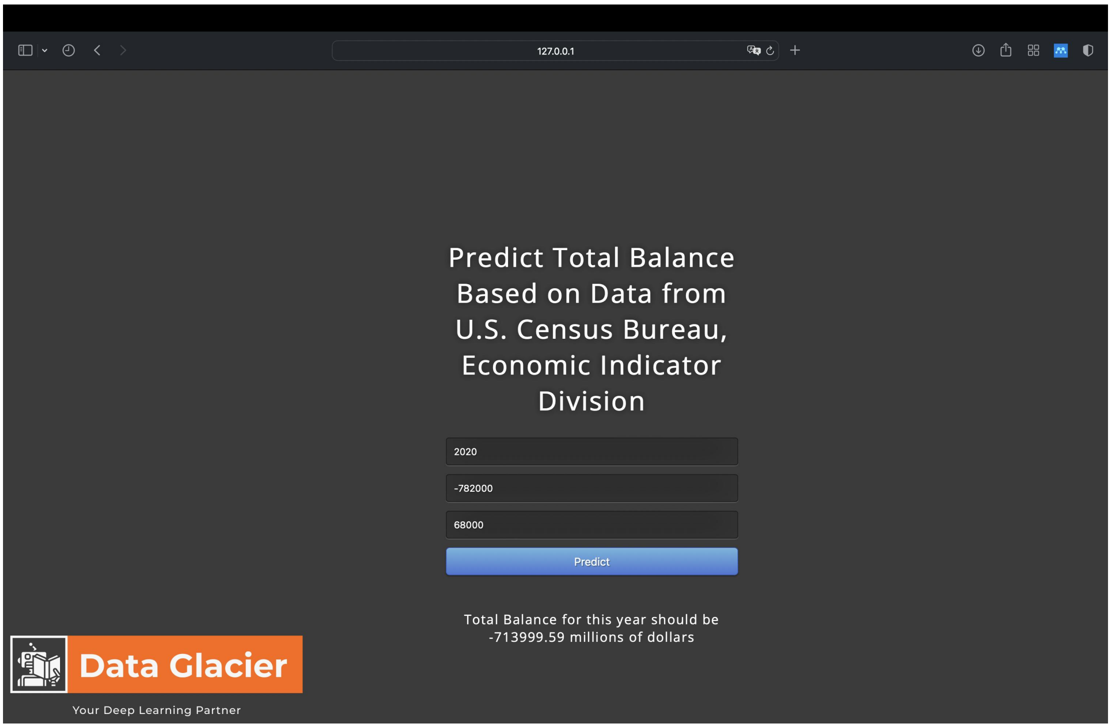
Then, i defined the structure of the website and the data we get in index.html file and input data into the model to get the predicted output value. By running app.py file, we will be able to see the website shown below in Fig 5 and use the model on site.

I tried with year 2020 with Goods BOP -782000 million dollars and Balance of Services 68000 million dollars. I got the predicted total balance as much as -713999.59 millions of dollars.



Predict Value of Goods Based on Data from U.S. Census Bureau, Economic Indicator Division

Fig5. Screenshot of the website



Predict Total Balance Based on Data from U.S. Census Bureau, Economic Indicator Division

2020

-782000

68000

Total Balance for this year should be -713999.59 millions of dollars

Fig6. The result after inputing the data