

# Data Intake Report

Name: Ana Lilliam Recio Garcia

Report date: 10/24/2022

Internship Batch: LISUM14

Version:<1.0>

Data intake by: Ana Lilliam Recio Garcia

Data intake reviewer:

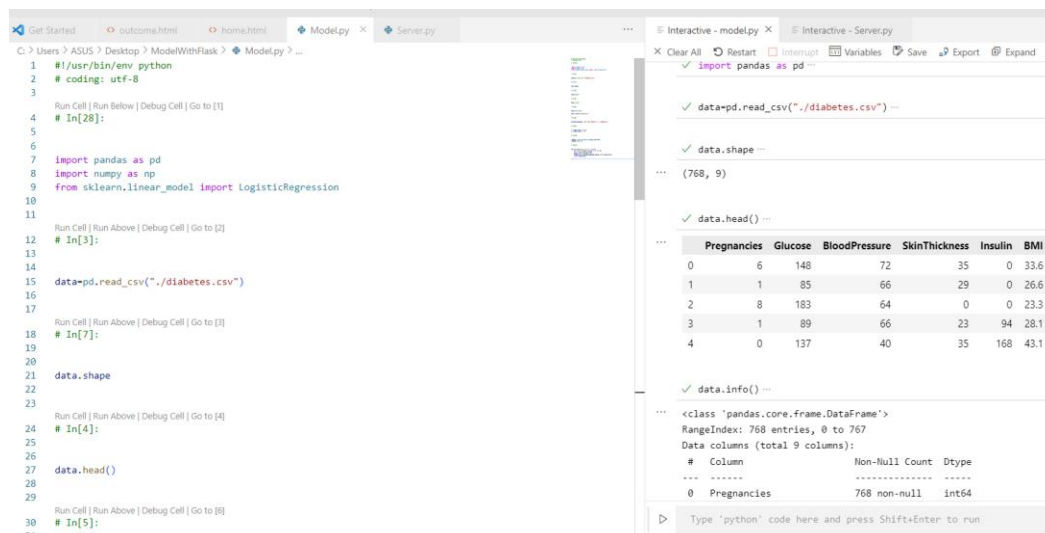
Data storage location: <https://github.com/AnaRecio/DataGlacierAna/tree/main/ModelWithFlask>

## Tabular data details:

<b>Total number of observations</b>	768
<b>Total number of files</b>	1
<b>Total number of features</b>	9
<b>Base format of the file</b>	.CSV
<b>Size of the data</b>	24KB

## Proposed Approach:

I have created from the file diabetes.csv a model using Logistic Regression, to evaluate the dependent variable which is Outcome (last column) in the dataset, from the independent variables which are the other 7 variables.



The screenshot displays a Jupyter Notebook interface with a code editor on the left and an interactive console on the right. The code in the notebook includes setting the environment, importing pandas and numpy, and loading the 'diabetes.csv' file. The interactive console shows the successful execution of these commands, displaying the data's shape as (768, 9) and a preview of the first five rows of the dataset.

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[28]:
5
6
7 import pandas as pd
8 import numpy as np
9 from sklearn.linear_model import LogisticRegression
10
11
12 Run Cell | Run Above | Debug Cell | Go to [1]
13 # In[3]:
14
15 data=pd.read_csv("./diabetes.csv")
16
17
18 Run Cell | Run Above | Debug Cell | Go to [3]
19 # In[7]:
20
21 data.shape
22
23
24 Run Cell | Run Above | Debug Cell | Go to [4]
25 # In[4]:
26
27 data.head()
28
29
30 Run Cell | Run Above | Debug Cell | Go to [6]
31 # In[5]:
```

Interactive - modelpy x Interactive - Server.py

- ✓ import pandas as pd
- ✓ data=pd.read\_csv("./diabetes.csv")
- ✓ data.shape
- ... (768, 9)
- ✓ data.head()
- ...

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI
0	6	148	72	35	0	33.6
1	1	85	66	29	0	26.6
2	8	183	64	0	0	23.3
3	1	89	66	23	94	28.1
4	0	137	40	35	168	43.1

✓ data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype
---  -
 0   Pregnancies         768 non-null    int64
```

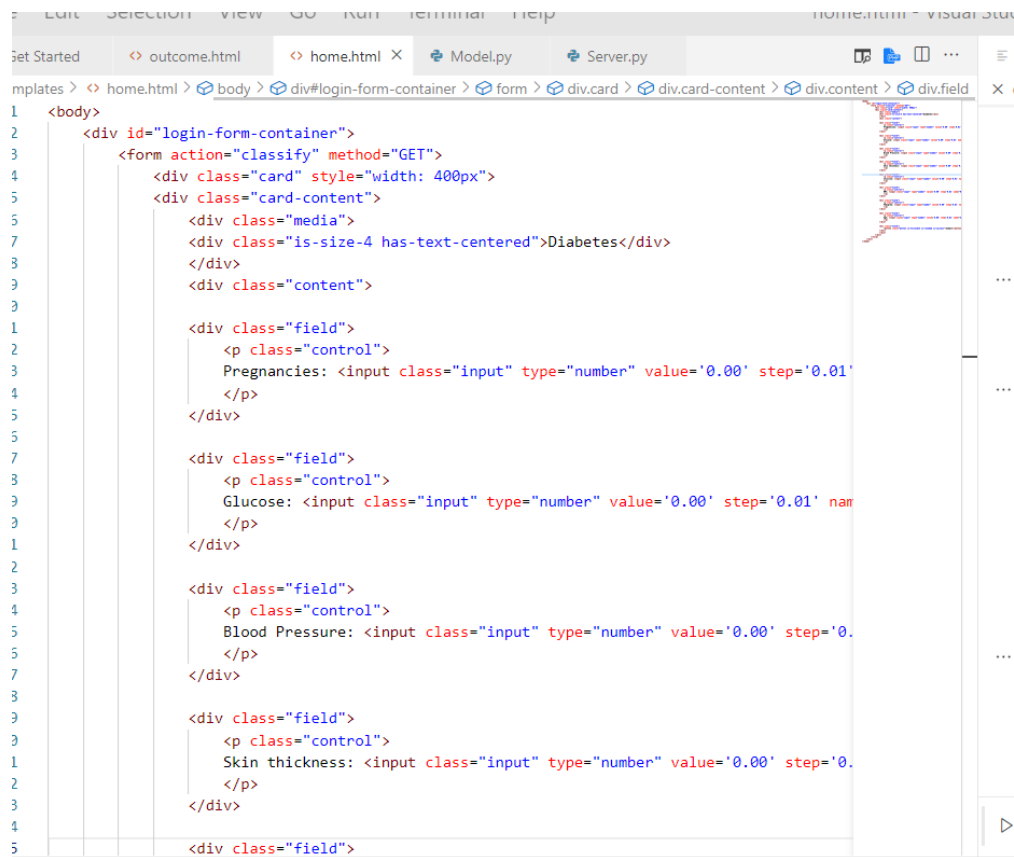
Type "python" code here and press Shift+Enter to run

```

5
6
7 Outcome_mappings = {0: "Not Diabetic", 1: "Diabetic"}
8
9
10 Run Cell | Run Above | Debug Cell | Go to [9]
11 # In[9]:
12
13
14 X = data.iloc[:, 0:-1]
15 y = data.iloc[:, -1]
16
17
18 Run Cell | Run Above | Debug Cell | Go to [10]
19 # In[10]:
20
21
22 logreg = LogisticRegression(max_iter=1000)
23 logreg.fit(X, y)
24
25
26 Run Cell | Run Above | Debug Cell | Go to [11]
27 # In[12]:
28
29
30 def outcome(a, b, c, d, e, f, g, h):
31     arr = np.array([a, b, c, d, e, f, g, h])
32     arr = arr.astype(np.float)
33     query = arr.reshape(1, -1)
34     prediction = Outcome_mappings[logreg.predict(query)[0]]
35     return prediction
36

```

I've created a simple form in HTML so the user can input on the variables and have an outcome



```

1 <body>
2   <div id="login-form-container">
3     <form action="classify" method="GET">
4       <div class="card" style="width: 400px">
5         <div class="card-content">
6           <div class="media">
7             <div class="is-size-4 has-text-centered">Diabetes</div>
8           </div>
9           <div class="content">
10
11             <div class="field">
12               <p class="control">
13                 Pregnancies: <input class="input" type="number" value='0.00' step='0.01'
14               </p>
15             </div>
16
17             <div class="field">
18               <p class="control">
19                 Glucose: <input class="input" type="number" value='0.00' step='0.01' nam
20               </p>
21             </div>
22
23             <div class="field">
24               <p class="control">
25                 Blood Pressure: <input class="input" type="number" value='0.00' step='0.
26               </p>
27             </div>
28
29             <div class="field">
30               <p class="control">
31                 Skin thickness: <input class="input" type="number" value='0.00' step='0.
32               </p>
33             </div>
34
35             <div class="field">

```

```
Get Started outcome.html x home.html Model.py Server.py
C: > Users > ASUS > Desktop > ModelWithFlask > templates > outcome.html > body > div#login-form-container > div.card >
1 <body>
2   <div id="login-form-container">
3     <div class="card" style="width: 400px">
4       <div class="card-content">
5         <div class="media">
6           <div class="is-size-4 has-text-centered">
7             {{ output }}
8           </div>
9         </div>
10        <form action="home">
11          <div class="field">
12            <button class="button is-fullwidth is-rounded is-success">Retry</but
13          </div>
14        </form>
15      </div>
16    </div>
17  </div>
18 </body>
```

Finally, I've imported Flask to use as server of the model and made two routes, one to home.html and the other to outcome.html

```
4 # In[18]:
5
6
7 import Model
8 from flask import Flask, render_template, request
9
10 Run Cell | Run Above | Debug Cell | Go to [2]
11 # In[19]:
12
13
14 app = Flask(__name__,template_folder="templates")
15
16 @app.route('/home')
17 def home():
18     return render_template('home.html')
19
20 Run Cell | Run Above | Debug Cell | Go to [4]
21 # In[21]:
22
23
```

```

-
}
4 @app.route('/outcome',methods=['GET'])
5 def classify_type():
6     try:
7         a = request.args.get('Pregnancies')
8         b = request.args.get('Glucose')
9         c = request.args.get('BPressure')
10        d = request.args.get('SkinThick')
11        e = request.args.get('Insuline')
12        f = request.args.get('BMI')
13        g = request.args.get('Pedigree')
14        h = request.args.get('Age')
15
16        output = Model.outcome(a, b, c, d, e, f, g, h)
17
18        return render_template('outcome.html', output=output)
19    except:
20        return 'Error'
21
22 if __name__ == '__main__':
23     app.run(debug=True)
24
;

```

I ran the server with the console and could see the port where it will run


Command Prompt - python server.py



```
Microsoft Windows [Version 10.0.22621.674]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>cd Desktop

C:\Users\ASUS\Desktop>cd ModelWithFlask

C:\Users\ASUS\Desktop\ModelWithFlask>python Server.py
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   Pregnancies           768 non-null    int64  
 1   Glucose               768 non-null    int64  
 2   BloodPressure         768 non-null    int64  
 3   SkinThickness         768 non-null    int64  
 4   Insulin               768 non-null    int64  
 5   BMI                   768 non-null    float64 
 6   DiabetesPedigreeFunction 768 non-null    float64 
 7   Age                  768 non-null    int64  
 8   Outcome               768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
* Serving Flask app 'Server'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
```

 127.0.0.1:5000/home × +

← → ↻   127.0.0.1:5000/home

## Diabetes

Pregnancies:

Glucose:

Blood Pressure:

Skin thickness:

Insuline:

BMI:

Pedigree:

Age: