Project: No-show appointments data analysis

Table of Contents

1. Introduction

This No-show appointments dataset collects information from 100k medical appointments in Brazil and is focused on the question of whether or not patients show up for their appointment. A number of characteristics about the patient are included in each row. including: • 'ScheduledDay' tells us on what day the patient set up their appointment. • 'Neighborhood' indicates the location of the hospital. • 'Scholarship' indicates whether or not the patient is enrolled in Brasilian welfare program Bolsa Família. (NB: the encoding of the last column: it says 'No' if the patient showed up to their appointment, and 'Yes' if they did not show up.) The questions I would like to answer with these data are: -What proportion of patients show up for their appointment? -What are the factors that are strongly correlated with patients' liklihood of whowing up at hospital? -Can we predict if a patient will show up for their scheduled appointment based on these factors?

In [182]:
```
# import statements
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
% matplotlib inline
```

2. Data Wrangling

2.1. Load data

In [161]:
```
df=pd.read_csv('noshowappointments-kagglev2-may-2016.csv')
```

2.2. Read data

In [162]:
```
df
```

Out[162]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA
1	5.589980e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA
2	4.262960e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA
3	8.679510e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI
4	8.841190e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA
5	9.598510e+13	5626772	F	2016-04-27T08:36:51Z	2016-04-29T00:00:00Z	76	REPÚBLICA
6	7.336880e+14	5630279	F	2016-04-27T15:05:12Z	2016-04-29T00:00:00Z	23	GOIABEIRAS
7	3.449830e+12	5630575	F	2016-04-27T15:39:58Z	2016-04-29T00:00:00Z	39	GOIABEIRAS
8	5.639470e+13	5638447	F	2016-04-29T08:02:16Z	2016-04-29T00:00:00Z	21	ANDORINHAS
9	7.812460e+13	5629123	F	2016-04-27T12:48:25Z	2016-04-29T00:00:00Z	19	CONQUISTA
10	7.345360e+14	5630213	F	2016-04-27T14:58:11Z	2016-04-29T00:00:00Z	30	NOVA PALESTINA
11	7.542950e+12	5620163	M	2016-04-26T08:44:12Z	2016-04-29T00:00:00Z	29	NOVA PALESTINA
12	5.666550e+14	5634718	F	2016-04-28T11:33:51Z	2016-04-29T00:00:00Z	22	NOVA PALESTINA
13	9.113950e+14	5636249	M	2016-04-28T14:52:07Z	2016-04-29T00:00:00Z	28	NOVA PALESTINA
14	9.988470e+13	5633951	F	2016-04-28T10:06:24Z	2016-04-29T00:00:00Z	54	NOVA PALESTINA
15	9.994839e+10	5620206	F	2016-04-26T08:47:27Z	2016-04-29T00:00:00Z	15	NOVA PALESTINA
16	8.457440e+13	5633121	M	2016-04-28T08:51:47Z	2016-04-29T00:00:00Z	50	NOVA PALESTINA
17	1.479500e+13	5633460	F	2016-04-28T09:28:57Z	2016-04-29T00:00:00Z	40	CONQUISTA
18	1.713540e+13	5621836	F	2016-04-26T10:54:18Z	2016-04-29T00:00:00Z	30	NOVA PALESTINA
19	7.223290e+12	5640433	F	2016-04-29T10:43:14Z	2016-04-29T00:00:00Z	46	DA PENHA

20	6.222570e+14	5626083	F	2016-04-27T07:51:14Z	2016-04-29T00:00:00Z	30	NOVA PALESTINA
21	1.215480e+13	5628338	F	2016-04-27T10:50:45Z	2016-04-29T00:00:00Z	4	CONQUISTA
22	8.632300e+14	5616091	M	2016-04-25T13:29:16Z	2016-04-29T00:00:00Z	13	CONQUISTA
23	2.137540e+14	5634142	F	2016-04-28T10:27:05Z	2016-04-29T00:00:00Z	46	CONQUISTA
24	8.734860e+12	5641780	F	2016-04-29T14:19:19Z	2016-04-29T00:00:00Z	65	TABUAZEIRO
25	5.819370e+12	5624020	M	2016-04-26T15:04:17Z	2016-04-29T00:00:00Z	46	CONQUISTA
26	2.578785e+10	5641781	F	2016-04-29T14:19:42Z	2016-04-29T00:00:00Z	45	BENTO FERREIRA
27	1.215480e+13	5628345	F	2016-04-27T10:51:45Z	2016-04-29T00:00:00Z	4	CONQUISTA
28	5.926170e+12	5642400	M	2016-04-29T15:48:02Z	2016-04-29T00:00:00Z	51	SÃO PEDRO
29	1.225780e+12	5642186	F	2016-04-29T15:16:29Z	2016-04-29T00:00:00Z	32	SANTA MARTHA
...	...	...	...	...	...	...	...
110497	7.935890e+14	5757745	M	2016-06-01T09:46:33Z	2016-06-01T00:00:00Z	76	MARIA ORTIZ
110498	9.433650e+13	5787655	F	2016-06-08T10:21:14Z	2016-06-08T00:00:00Z	59	MARIA ORTIZ
110499	8.219690e+14	5757697	F	2016-06-01T09:42:56Z	2016-06-01T00:00:00Z	66	MARIA ORTIZ
110500	4.434380e+14	5787233	F	2016-06-08T09:35:13Z	2016-06-08T00:00:00Z	59	MARIA ORTIZ
110501	4.544250e+11	5758133	M	2016-06-01T10:19:12Z	2016-06-01T00:00:00Z	44	MARIA ORTIZ
110502	7.316230e+14	5787937	F	2016-06-08T10:50:42Z	2016-06-08T00:00:00Z	22	GOIABEIRAS
110503	2.362180e+13	5759473	F	2016-06-01T13:00:36Z	2016-06-01T00:00:00Z	64	SOLON BORGES
110504	9.947980e+12	5788052	F	2016-06-08T11:06:21Z	2016-06-08T00:00:00Z	4	MARIA ORTIZ
110505	5.667340e+13	5758455	F	2016-06-01T10:45:50Z	2016-06-01T00:00:00Z	55	MARIA ORTIZ
110506	8.973880e+11	5758779	M	2016-06-01T11:09:20Z	2016-06-01T00:00:00Z	5	MARIA ORTIZ
110507	4.769460e+14	5786918	F	2016-06-08T09:04:18Z	2016-06-08T00:00:00Z	0	MARIA ORTIZ
				2016-06-	2016-06-		

110508	9.433650e+13	5757656	F	01T09:41:00Z	01T00:00:00Z	59	MARIA ORTIZ
110509	4.952970e+14	5786750	M	2016-06-08T08:50:51Z	2016-06-08T00:00:00Z	33	MARIA ORTIZ
110510	2.362180e+13	5757587	F	2016-06-01T09:35:48Z	2016-06-01T00:00:00Z	64	SOLON BORGES
110511	8.236000e+11	5786742	F	2016-06-08T08:50:20Z	2016-06-08T00:00:00Z	14	MARIA ORTIZ
110512	9.876250e+13	5786368	F	2016-06-08T08:20:01Z	2016-06-08T00:00:00Z	41	MARIA ORTIZ
110513	8.674780e+13	5785964	M	2016-06-08T07:52:55Z	2016-06-08T00:00:00Z	2	ANTÔNIO HONÓRIO
110514	2.695690e+12	5786567	F	2016-06-08T08:35:31Z	2016-06-08T00:00:00Z	58	MARIA ORTIZ
110515	6.456340e+14	5778621	M	2016-06-06T15:58:05Z	2016-06-08T00:00:00Z	33	MARIA ORTIZ
110516	6.923770e+13	5780205	F	2016-06-07T07:45:16Z	2016-06-08T00:00:00Z	37	MARIA ORTIZ
110517	5.574940e+12	5780122	F	2016-06-07T07:38:34Z	2016-06-07T00:00:00Z	19	MARIA ORTIZ
110518	7.263310e+13	5630375	F	2016-04-27T15:15:06Z	2016-06-07T00:00:00Z	50	MARIA ORTIZ
110519	6.542390e+13	5630447	F	2016-04-27T15:23:14Z	2016-06-07T00:00:00Z	22	MARIA ORTIZ
110520	9.969980e+14	5650534	F	2016-05-03T07:51:47Z	2016-06-07T00:00:00Z	42	MARIA ORTIZ
110521	3.635530e+13	5651072	F	2016-05-03T08:23:40Z	2016-06-07T00:00:00Z	53	MARIA ORTIZ
110522	2.572130e+12	5651768	F	2016-05-03T09:15:35Z	2016-06-07T00:00:00Z	56	MARIA ORTIZ
110523	3.596270e+12	5650093	F	2016-05-03T07:27:33Z	2016-06-07T00:00:00Z	51	MARIA ORTIZ
110524	1.557660e+13	5630692	F	2016-04-27T16:03:52Z	2016-06-07T00:00:00Z	21	MARIA ORTIZ
110525	9.213490e+13	5630323	F	2016-04-27T15:09:23Z	2016-06-07T00:00:00Z	38	MARIA ORTIZ
110526	3.775120e+14	5629448	F	2016-04-27T13:30:56Z	2016-06-07T00:00:00Z	54	MARIA ORTIZ

110527 rows × 14 columns

In [163]:
```
df.head(10)
```

Out[163]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scho

0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA
1	5.589980e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA
2	4.262960e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA
3	8.679510e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI
4	8.841190e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA
5	9.598510e+13	5626772	F	2016-04-27T08:36:51Z	2016-04-29T00:00:00Z	76	REPÚBLICA
6	7.336880e+14	5630279	F	2016-04-27T15:05:12Z	2016-04-29T00:00:00Z	23	GOIABEIRAS
7	3.449830e+12	5630575	F	2016-04-27T15:39:58Z	2016-04-29T00:00:00Z	39	GOIABEIRAS
8	5.639470e+13	5638447	F	2016-04-29T08:02:16Z	2016-04-29T00:00:00Z	21	ANDORINHAS
9	7.812460e+13	5629123	F	2016-04-27T12:48:25Z	2016-04-29T00:00:00Z	19	CONQUISTA

>Look the number of rows/records and columns/fields

In [164]:
```
df.shape
```

Out[164]:
```
(110527, 14)
```

>Describe each field with summary statistics

In [165]:
```
df.describe()
```

Out[165]:

	PatientId	AppointmentID	Age	Scholarship	Hipertension	Diabetes	
count	1.105270e+05	1.105270e+05	110527.000000	110527.000000	110527.000000	110527.000000	11
mean	1.474963e+14	5.675305e+06	37.088874	0.098266	0.197246	0.071865	
std	2.560949e+14	7.129575e+04	23.110205	0.297675	0.397921	0.258265	
min	3.920000e+04	5.030230e+06	-1.000000	0.000000	0.000000	0.000000	
25%	4.172615e+12	5.640286e+06	18.000000	0.000000	0.000000	0.000000	
50%	3.173180e+13	5.680573e+06	37.000000	0.000000	0.000000	0.000000	
75%	9.439170e+13	5.725524e+06	55.000000	0.000000	0.000000	0.000000	
max	9.999820e+14	5.790484e+06	115.000000	1.000000	1.000000	1.000000	

>Explore the data for missing values for each field

> As every field has all 110527 records, the data do not have missing values

In [166]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 14 columns):
PatientId         110527 non-null float64
AppointmentID     110527 non-null int64
Gender            110527 non-null object
ScheduledDay      110527 non-null object
AppointmentDay    110527 non-null object
Age               110527 non-null int64
Neighbourhood     110527 non-null object
Scholarship       110527 non-null int64
Hipertension      110527 non-null int64
Diabetes          110527 non-null int64
Alcoholism        110527 non-null int64
Handcap           110527 non-null int64
SMS_received      110527 non-null int64
No-show           110527 non-null object
dtypes: float64(1), int64(8), object(5)
memory usage: 11.8+ MB
```

>Explore the data types for each column/field. This help to decide the appropriatnes of variables for the analysis.

In [167]:
```
df.dtypes
```

Out[167]:
```
PatientId         float64
AppointmentID       int64
Gender             object
ScheduledDay       object
AppointmentDay     object
Age                 int64
Neighbourhood      object
Scholarship         int64
Hipertension        int64
Diabetes            int64
Alcoholism          int64
Handcap             int64
SMS_received        int64
No-show            object
dtype: object
```

2.3. Prepare data for analysis

To answer the quesions, what factors affect patients attendance at the appointment, and develop aprediction model, I need to transform string variables in to numeric type. The outcome variable "No_show" is a binary categorical variable. So, we can use a binary logistic regression to find out the significant predictors. The regression analysis is done in two steps the first one is a multivariable logistic regression with association coeficients where the adjuted immpact of each variable is assessed. The second analysis is fiting a prediction model using classification matchine learing approach.

> In the dataset the outcome variable is given as No_show and those with response "No" are patients attended their appointment. I rename the column name to "Show_up" and replaced the values of "No" by Yes and visversa.

In [172]:

```
#rename column No_show to Show_up
df.rename(columns={'No-show':'Show_up'}, inplace=True)

df.head(11)
```

Out[172]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Sch
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	
1	5.589980e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	
2	4.262960e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	
3	8.679510e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	
4	8.841190e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	
5	9.598510e+13	5626772	F	2016-04-27T08:36:51Z	2016-04-29T00:00:00Z	76	REPÚBLICA	
6	7.336880e+14	5630279	F	2016-04-27T15:05:12Z	2016-04-29T00:00:00Z	23	GOIABEIRAS	
7	3.449830e+12	5630575	F	2016-04-27T15:39:58Z	2016-04-29T00:00:00Z	39	GOIABEIRAS	
8	5.639470e+13	5638447	F	2016-04-29T08:02:16Z	2016-04-29T00:00:00Z	21	ANDORINHAS	
9	7.812460e+13	5629123	F	2016-04-27T12:48:25Z	2016-04-29T00:00:00Z	19	CONQUISTA	
10	7.345360e+14	5630213	F	2016-04-27T14:58:11Z	2016-04-29T00:00:00Z	30	NOVA PALESTINA	

In [173]:

```
df["Show_up"].replace({'No':'yes','Yes':'no'}, inplace=True)
```

```
df.head(11)
```

Out[173]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Sch
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	
1	5.589980e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	
2	4.262960e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	
3	8.679510e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	
4	8.841190e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	
5	9.598510e+13	5626772	F	2016-04-27T08:36:51Z	2016-04-29T00:00:00Z	76	REPÚBLICA	
6	7.336880e+14	5630279	F	2016-04-27T15:05:12Z	2016-04-29T00:00:00Z	23	GOIABEIRAS	
7	3.449830e+12	5630575	F	2016-04-27T15:39:58Z	2016-04-29T00:00:00Z	39	GOIABEIRAS	
8	5.639470e+13	5638447	F	2016-04-29T08:02:16Z	2016-04-29T00:00:00Z	21	ANDORINHAS	
9	7.812460e+13	5629123	F	2016-04-27T12:48:25Z	2016-04-29T00:00:00Z	19	CONQUISTA	
10	7.345360e+14	5630213	F	2016-04-27T14:58:11Z	2016-04-29T00:00:00Z	30	NOVA PALESTINA	

Convert/ encode string variables to numeric

In [175]:
```
from sklearn.preprocessing import LabelEncoder
```

Gender

In [176]:
```
df['Gender'].value_counts()
```

Out[176]:
```
F    71840
M    38687
Name: Gender, dtype: int64
```

In [177]:
```
lb_Gender = LabelEncoder()
df["Gender_code"] = lb_Gender.fit_transform(df["Gender"])
df[["Gender", "Gender_code"]].head(11)
```

Out[177]:

	Gender	Gender_code

0	F	0
1	M	1
2	F	0
3	F	0
4	F	0
5	F	0
6	F	0
7	F	0
8	F	0
9	F	0
10	F	0

Show_up

In [179]:
```
lb_Show_up = LabelEncoder()
df["show_up_code"] = lb_Show_up.fit_transform(df["Show_up"])
df[["Show_up", "show_up_code"]].head(11)
```

Out[179]:

	Show_up	show_up_code
0	yes	1
1	yes	1
2	yes	1
3	yes	1
4	yes	1
5	yes	1
6	no	0
7	no	0
8	yes	1
9	yes	1
10	yes	1

Neighbourhood

In [180]:
```
lb_Neighbourhood = LabelEncoder()
df["Neighbourhood_code"] = lb_Neighbourhood.fit_transform(df["Neighbourhoo
d"])
df[["Neighbourhood", "Neighbourhood_code"]]
```

Out[180]:

	Neighbourhood	Neighbourhood_code
0	JARDIM DA PENHA	39
1	JARDIM DA PENHA	39
2	MATA DA PRAIA	45
3	PONTAL DE CAMBURI	54
4	JARDIM DA PENHA	39
5	REPÚBLICA	58
6	GOIABEIRAS	25
7	GOIABEIRAS	25
8	ANDORINHAS	1
9	CONQUISTA	12
10	NOVA PALESTINA	50
11	NOVA PALESTINA	50
12	NOVA PALESTINA	50
13	NOVA PALESTINA	50
14	NOVA PALESTINA	50
15	NOVA PALESTINA	50
16	NOVA PALESTINA	50
17	CONQUISTA	12
18	NOVA PALESTINA	50
19	DA PENHA	15
20	NOVA PALESTINA	50
21	CONQUISTA	12
22	CONQUISTA	12
23	CONQUISTA	12
24	TABUAZEIRO	78
25	CONQUISTA	12
26	BENTO FERREIRA	6
27	CONQUISTA	12
28	SÃO PEDRO	77
29	SANTA MARTHA	66
...	...	...
110497	MARIA ORTIZ	43
110498	MARIA ORTIZ	43
110499	MARIA ORTIZ	43

110500	MARIA ORTIZ	43
110501	MARIA ORTIZ	43
110502	GOIABEIRAS	25
110503	SOLON BORGES	73
110504	MARIA ORTIZ	43
110505	MARIA ORTIZ	43
110506	MARIA ORTIZ	43
110507	MARIA ORTIZ	43
110508	MARIA ORTIZ	43
110509	MARIA ORTIZ	43
110510	SOLON BORGES	73
110511	MARIA ORTIZ	43
110512	MARIA ORTIZ	43
110513	ANTÔNIO HONÓRIO	2
110514	MARIA ORTIZ	43
110515	MARIA ORTIZ	43
110516	MARIA ORTIZ	43
110517	MARIA ORTIZ	43
110518	MARIA ORTIZ	43
110519	MARIA ORTIZ	43
110520	MARIA ORTIZ	43
110521	MARIA ORTIZ	43
110522	MARIA ORTIZ	43
110523	MARIA ORTIZ	43
110524	MARIA ORTIZ	43
110525	MARIA ORTIZ	43
110526	MARIA ORTIZ	43

110527 rows × 2 columns

Check data types

In [181]:
```
df.dtypes
```

Out[181]:
```
PatientId             float64
AppointmentID           int64
Gender                 object
ScheduledDay           object
```

```
AppointmentDay          object
Age                      int64
Neighbourhood           object
Scholarship              int64
Hipertension             int64
Diabetes                 int64
Alcoholism               int64
Handcap                  int64
SMS_received             int64
Show_up                 object
Gender_code              int64
show_up_code             int64
Neighbourhood_code       int64
dtype: object
```

Exploratory Data Analysis

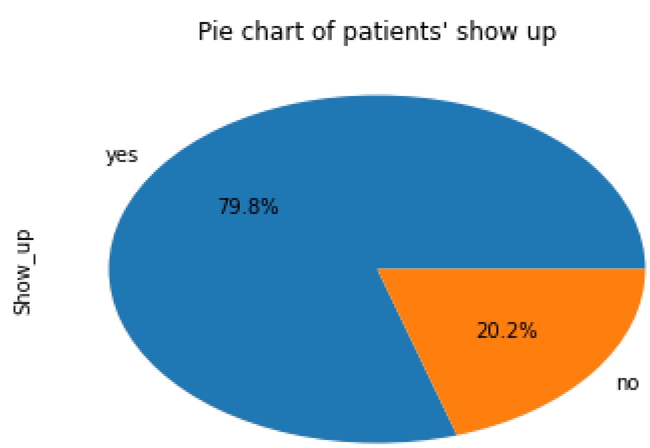
Research Question 1: What proportion show up?

In [183]:
```
df['Show_up'].value_counts()
```

Out[183]:
```
yes    88208
no     22319
Name: Show_up, dtype: int64
```

In [93]:
```
df['Show_up'].value_counts().plot(kind='pie',autopct='%.1f%%',title="Pie c
hart of patients' show up")
```

Out[93]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcf2f9b8710>
```



Finding: From the total 110,527 patients 88,208 show up for their appointment. Thus, the proportion of patients those show up to the hospitab on their appointment was 79.8%.FiG 1

Research Question 2.

What factors are important for us to know in order to predict if a patient will show up for their scheduled appointment?

Logistic regression

> Assign the independent variables (X) and the dependent variable (y)

In [184]:
```
X=df[['Gender_code','Age','Neighbourhood_code','Scholarship','Hipertension
','Diabetes','Alcoholism','Handcap','SMS_received']]
y=df['show_up_code']
```

> Fit the data and get a summary output with coefficients

In [136]:
```
logit_model=sm.Logit(y,X)

result=logit_model.fit()

print(result.summary2())

params = result.params
conf = result.conf_int()
conf['OR'] = params
conf.columns = ['2.5%', '97.5%', 'OR']
print(np.exp(conf))
```

```
Optimization terminated successfully.
         Current function value: 0.509487
         Iterations 6
                          Results: Logit
===================================================================
Model:              Logit            Pseudo R-squared: -0.013
Dependent Variable: show_up_code     AIC:              112642.2333
Date:               2020-01-31 04:21 BIC:              112728.7504
No. Observations:   110527           Log-Likelihood:   -56312.
Df Model:           8                LL-Null:          -55603.
Df Residuals:       110518           LLR p-value:      1.0000
Converged:          1.0000           Scale:            1.0000
No. Iterations:     6.0000
-------------------------------------------------------------------
                    Coef.  Std.Err.    z     P>|z|   [0.025  0.975]
-------------------------------------------------------------------
Gender_code         0.3708   0.0151 24.5529 0.0000  0.3412  0.4004
Age                 0.0202   0.0003 61.8919 0.0000  0.0196  0.0208
```

```
Neighbourhood_code  0.0143   0.0003  55.1926 0.0000  0.0138  0.0148
Scholarship         0.1186   0.0241   4.9185 0.0000  0.0713  0.1658
Hipertension       -0.0970   0.0248  -3.9102 0.0001 -0.1456 -0.0484
Diabetes           -0.1510   0.0346  -4.3698 0.0000 -0.2187 -0.0833
Alcoholism         -0.2345   0.0453  -5.1763 0.0000 -0.3234 -0.1457
Handcap            -0.0328   0.0491  -0.6671 0.5047 -0.1291  0.0635
SMS_received       -0.4292   0.0150 -28.6123 0.0000 -0.4586 -0.3998
===================================================================

                        2.5%     97.5%        OR
Gender_code         1.406642  1.492429  1.448901
Age                 1.019744  1.021048  1.020396
Neighbourhood_code  1.013851  1.014879  1.014365
Scholarship         1.073919  1.180341  1.125873
Hipertension        0.864545  0.952794  0.907598
Diabetes            0.803546  0.920102  0.859851
Alcoholism          0.723716  0.864384  0.790929
Handcap             0.878887  1.065594  0.967749
SMS_received        0.632165  0.670452  0.651027
```

Interpretation of logit model output: The liklihood of patients showing up at the hospital at the time of their appointment higher if they are male, older age, have scholarship. And having hipertention, diabetes, alcholism and reciving SMS were found to be decreasing the probability of showing up. Although these the associations of these factors with the liklihood od showing up were statitically significant, there were no strong associations as indicated by the odds ratios (OR)

Research Question 3. Can we Predict patients show up?

Above we have seen the associations between independent variables and the outcome variable with the logit model. Now let us evaluate the pridiction capacity of our model with help of matchine learning.

> Set the train and test split

In [ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
```

> Set 75% of the data train and the test will be on 25% of the data

In [185]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_s
tate=0)
```

>import class

In [126]:
```
from sklearn.linear_model import LogisticRegression
#intiate model
```

```
logreg = LogisticRegression()
```

>fit the model with data

In [127]:
```
logreg.fit(X_train, y_train)
```

Out[127]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=Tru
e,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

>Prediction

In [137]:
```
y_pred=logreg.predict(X_test)
y_pred
```

Out[137]:
```
array([1, 1, 1, ..., 1, 1, 1])
```

>Model Evaluation using Confusion Matrix

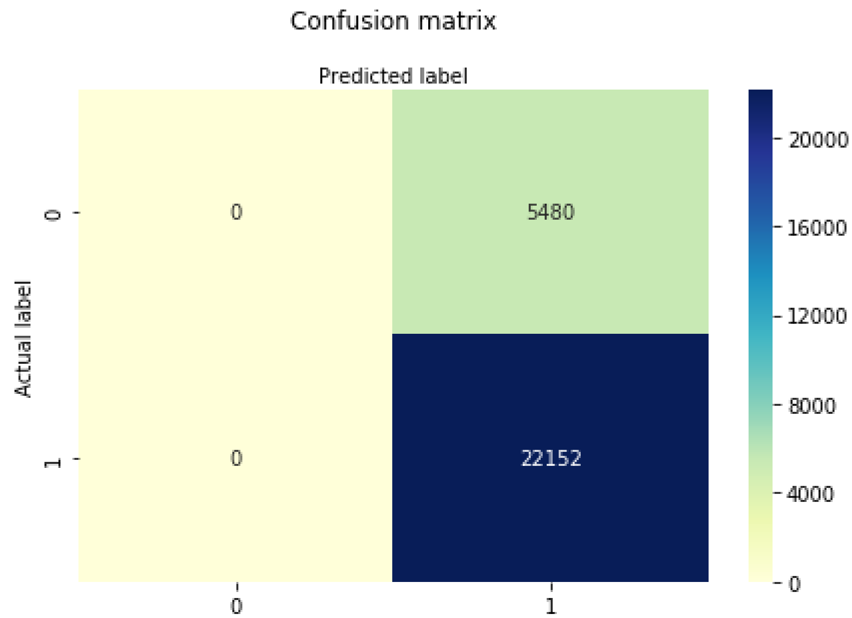
In [130]:
```
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

Out[130]:
```
array([[    0,  5480],
       [    0, 22152]])
```

>Visualizing Confusion Matrix using Heatmap

In [132]:
```
class_names=[0,1] # name  of classes
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
# create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

Out[132]:
```
Text(0.5,257.44,'Predicted label')
```



Interpretation: The model is good at prediction if the patient show up.

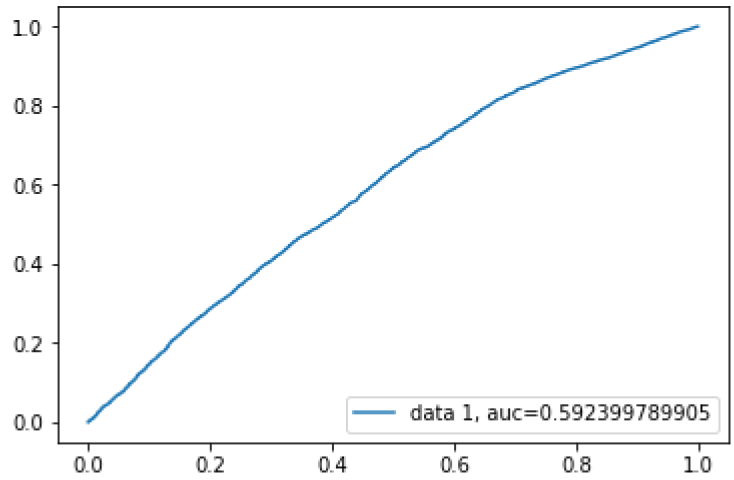
>Confusion Matrix Evaluation Metrics

In [133]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.801679212507
Precision: 0.801679212507
Recall: 1.0
```

Interpretation classification rate of 80%, considered as good accuracy. Recall: If there are patients who showed up in the test set and the model can identify it 100% of the time.

In [131]:
```
#roc curve
y_pred_proba = logreg.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()
```



Interpretation the AUC score for the case is 0.59. AUC score 1 represents perfect classifier, and 0.5 represents a worthless classifier. Thus, our model is not a good classifier.

Conclusions

In conclussion the proportion of patients showed up at the hospital based on their schedule was 79.8%. Although several factors have shown a statistically significant association, all of the associations were weak associations. Thus, from these dataset, we can not predict patients attendance by factors.

In [186]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])
```

Out[186]: 0

In [ ]: