Investigating The No-show Appointments Dataset

July 24, 2022

1 Introduction

The No-show 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.

'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. Be careful about 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.

1.0.1 Overview of the dataset

A detailed description of the columns is tabulated below.

Attribute	Description
PatientId	is a unique identifier for each patient.
AppointmentID	patients recive a unique appointment identifier
Gender	The patient's gender
ScheduledDay	The patient's scheduled day
AppointmentDay	The patient's appointment day
Age	The patient's age

Attribute	Description
Neighbourhood	The patient's neighbourhood
Scholarship	The patient's scholarship status with two attributes, 0 - No Scholarship and 1 - has Scholarship
Hipertension	The patient's Hipertension status with two attributes, 0 - Has no Hipertension and 1 - Has Hipertension
Diabetes	The patient's Diabetes status with two attributes, 0 - No Diabetes and 1 - Diabetic
Alcoholism	The patient's Alcoholism status with two attributes, 0 - Not alcoholic and 1 - Alcoholic
Handcap	The patient's Handcap status with two attributes, 0 - Not Handcapped and 1 - Handcapped

Attribute	Description
SMS_received	Describes whether the patient received a text before the appontment, 0 - No and 1 - Yes
No-show	this attribute is the Target Variable which describes whether the patient showed for the appointment

1.0.2 Importing Libraries

Importing libraries to be used for mathematical computation and data visualization is the first step. Numpy and pandas will assist in computation while matplotlib will be used to visualize the data.

```
In [1]: # Use this cell to set up import statements for all of the packages that you
        #   plan to use.
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        import tensorflow as tf
        # Remember to include a 'magic word' so that your visualizations are plotted
        #   inline with the notebook. See this page for more:
        #   http://ipython.readthedocs.io/en/stable/interactive/magics.html
```

```
/opt/conda/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:458: FutureWarning:
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/opt/conda/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:459: FutureWarning:
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/opt/conda/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:460: FutureWarning:
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/opt/conda/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:461: FutureWarning:
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/opt/conda/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:462: FutureWarning:
```

```
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/opt/conda/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:465: FutureWarning:
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

1.0.3 Update libraries

Update the pandas library

```
In [2]: # Upgrade pandas to use dataframe.explode() function.
        !pip install --upgrade pandas==0.25.0
```

```
Requirement already up-to-date: pandas==0.25.0 in /opt/conda/lib/python3.6/site-packages (0.25.0
Requirement already satisfied, skipping upgrade: python-dateutil>=2.6.1 in /opt/conda/lib/python
Requirement already satisfied, skipping upgrade: numpy>=1.13.3 in /opt/conda/lib/python3.6/site-
Requirement already satisfied, skipping upgrade: pytz>=2017.2 in /opt/conda/lib/python3.6/site-p
Requirement already satisfied, skipping upgrade: six>=1.5 in /opt/conda/lib/python3.6/site-packa
```

View the first five rows of your data

```
In [3]: # Load your data and print out a few lines. Perform operations to inspect data
        #   types and look for instances of missing or possibly errant data.
        pd.options.display.max_rows = 9999
        df = pd.read_csv('/home/workspace/Database_No_show_appointments/noshowappointments-kaggl
        df.head()
```

Out[3]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1	0	0	0	0	No
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	0	0	0	0	0	No
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0	0	0	0	0	0	No
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0	0	0	0	0	0	No
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	1	1	0	0	0	No

The dataset has 110527 rows and 14 columns.

```
In [4]: df.shape

Out[4]: (110527, 14)
```

The info module produce a detailed description of range index, column number, value count of each attribute and data type of each attribute

```
In [5]: 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
```

To inspect where the missing values are, the isnull() or isna() modules are used.

```
In [6]: df.isnull().sum()

Out[6]: PatientId         0
        AppointmentID     0
        Gender            0
        ScheduledDay      0
        AppointmentDay    0
        Age               0
        Neighbourhood     0
        Scholarship       0
        Hipertension      0
        Diabetes          0
        Alcoholism        0
        Handcap           0
        SMS_received      0
        No-show           0
        dtype: int64
```

1.1 Data Wrangling

The dataset was gathered in one spreadsheet,assessed and cleaned

1.1.1 Data Cleaning

The PatientId and Scheduleday were dropped as they did not help explain the No-show attribute

Convert 'Yes' to 1 and 'No' to 0 from the No-show column

```
In [7]: from sklearn.preprocessing import LabelEncoder
        labelencoder_Y = LabelEncoder()
        df.iloc[:,-1] = labelencoder_Y.fit_transform(df_clean.iloc[:,-1].values)


        ---------------------------------------------------------------------------

        NameError                                 Traceback (most recent call last)

        <ipython-input-7-7f22eb1e1880> in <module>()
          1 from sklearn.preprocessing import LabelEncoder
          2 labelencoder_Y = LabelEncoder()
    ----> 3 df.iloc[:,-1] = labelencoder_Y.fit_transform(df_clean.iloc[:,-1].values)


        NameError: name 'df_clean' is not defined


In [ ]: # After discussing the structure of the data and any problems that need to be
        #   cleaned, perform those cleaning steps in the second part of this section.
        #remove= ['id','imdb_id','budget','revenue','original_title','cast','homepage','director
        df_clean = df.drop(['PatientId','AppointmentDay','AppointmentID'],axis=1)
        df_clean.head()
```

Exploratory Data Analysis

1.1.2 How does each independent variable help explain the dependent variable (No-show)?

First let's get a summary of our cleaned dataset

```
In [ ]: # Use this, and more code cells, to explore your data. Don't forget to add
        #   Markdown cells to document your observations and findings.
        df_clean.describe()
```

```
In [ ]: df_clean['Gender'].value_counts(normalize=True)
```

There is imbalance in the gender of the patients as the ratio of female to male is 65:35.

```
In [ ]: df_clean['No-show'].value_counts(normalize=True)
```

Their is another imbalance between those who attended their scheduled appointments versus those who did not attend at a ratio of 20:80.

```
In [ ]: df_clean.groupby('No-show').mean()
```

The average age of those who did not show up for their scheduled appoint is higher than those who attended their scheduled appointment,

Those who received scholarships were likely to attend their scheduled appointment compared to those who did not receive scholarships,

Those who never showed up for their scheduled appointment had a higher Hipertension, Diabetes and Handcap averages compared to those who showed up,

The alcoholism average is approximately the same for bothe groups, and

Those who received SMS reminding them of their appointments were more likely to show up compared to those who never received the SMS.

Categorical mean from all the other attributes are:

```
In [ ]: df_clean.groupby('Scholarship').mean()
```

```
In [ ]: df_clean.groupby('Hipertension').mean()
```

```
In [ ]: df_clean.groupby('Diabetes').mean()
```

```
In [ ]: df_clean.groupby('Alcoholism').mean()
```

```
In [ ]: df_clean.groupby('Handcap').mean()
```

```
In [ ]: df_clean.groupby('SMS_received').mean()
```

Age has a negative correlation with Scholarship and No-show,

Scholarship has a negative relationship with Age, Hipertension,Diabetes and Handcap,

Hipertension, Diabetes and Handicap all have a negative correlation with Scholarship, SMS_received and No-show; and

Alcoholism has a negative correlation with SMS_received and No-show

```
In [ ]: %matplotlib inline
        %config inlinebackend.figure_format = 'retina'
```

1.1.3 a)univariate anaysis

```
In [ ]: sns.countplot(df_clean['Gender'],label='count');
        plt.title('Distribution of Gender');
```

```
In [ ]: sns.countplot(df_clean['No-show'],label='count');
        plt.title('Distribution of No-show');
```

1.1.4 b)Bivariate Analysis

```
In [ ]: pd.crosstab(df_clean['SMS_received'],df_clean['No-show']).plot(kind='bar')
        plt.title('Appointment Distribution for SMS_received')
        plt.xlabel('SMS_received')
        plt.ylabel('No-show')
        plt.savefig('no-show-SMS_received')
```

The likelihood of attending the appointment depends a great deal on whether SMS was received or not. Thus, the SMS-received can be a good predictor of the target variable.

```
In [ ]: pd.crosstab(df_clean['Scholarship'],df_clean['No-show']).plot(kind='bar')
        plt.title('Appointment Distribution for Scholarship')
        plt.xlabel('Scholarship')
        plt.ylabel('No-show')
        plt.savefig('no-show-Scholarship')
```

The likelihood of attending the appointment depends a great deal on whether the individual received a scholarship or not. Thus, the Scholarship can be a good predictor of the target variable.

```
In [ ]: pd.crosstab(df_clean['Hipertension'],df_clean['No-show']).plot(kind='bar')
        plt.title('Appointment Distribution for Hipertension')
        plt.xlabel('Hipertension')
        plt.ylabel('No-show')
        plt.savefig('no-show-Hipertension')
```

Hipertension may be a good predictor of the outcome

```
In [ ]: df_clean['Age'].hist()
        plt.title('Histogram of Age')
        plt.xlabel('Age')
        plt.ylabel('Frequency')
        plt.savefig('hist_age')
```

Most of the patients in this dataset are in the age range of 0-10.

```
In [ ]: pd.crosstab(df_clean['Age'],df_clean['No-show']).plot(kind='hist')
        plt.title('Appointment Distribution for Age')
        plt.xlabel('Age')
        plt.ylabel('No-show')
        plt.savefig('no-show-age')
```

Age may be a good predictor of the outcome

```
In [ ]: pd.crosstab(df_clean['Diabetes'],df_clean['No-show']).plot(kind='bar')
        plt.title('Appointment Distribution for Diabetes')
        plt.xlabel('DIabetes')
        plt.ylabel('No-show')
        plt.savefig('no-show-diabetes')
```

Diabetes may be a good predictor of the outcome

```
In [ ]: pd.crosstab(df_clean['Handcap'],df_clean['No-show']).plot(kind='bar')
        plt.title('Appointment Distribution for Handcap')
        plt.xlabel('Handcap')
        plt.ylabel('No-show')
        plt.savefig('no-show-handcap')
```

Handcap may not a good predictor of the outcome

```
In [ ]: pd.crosstab(df_clean['Alcoholism'],df_clean['No-show']).plot(kind='bar')
        plt.title('Appointment Distribution for Alcoholism')
        plt.xlabel('Alcoholism')
        plt.ylabel('No-show')
        plt.savefig('no-show-alcoholism')
```

Alcoholism may not be a good predictor

```
In [ ]: sns.pairplot(df_clean,hue='No-show');
```

1.1.5 c)Multivariate Analysis

```
In [ ]: df_clean.corr()
```

```
In [ ]: fig, ax = plt.subplots(figsize=(10,8))
        sns.heatmap(df_clean.corr(),annot=True,cmap='coolwarm');
```

Conclusions

Independent variables that help explain the No-show variable are: age,scholarship,hipertension,diabetes and SMS- received.

Majority of the patents are aged between 0 and 10 which explains why they are likely to miss their appointment.

Also children under the age of 10 are not all expected to own a cell phone, Which might explain why some never received an SMS about their appointment.

The correlation between No-show and SMS-received is 0.13 which is pretty strong compared to other independent variables. Which means it's influence on the No-show outcome is pretty strong.

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