Predict Medical Appointments No-Shows

- Support Vector Machine
- Logistic regression
- Decision Tree & Random Forest

Supervised Binary Classification Model

Objectives:

- What and how much data are you using?
- How many total data rows (observations) do you have (in your full dataset)?
- How many columns (variables) do you have?
- How are you splitting your full dataset into training and testing datasets?
- What's your target (Y) variable? What are you trying to predict?
- What predictor (X) variables are you using to train the model? How did you select those X variables?
- What pre-processing steps are you doing (e.g., normalizing/scaling numerical variable, or encoding categorical variables)?
- What type of models did you try so far?
- What metrics are you using to evaluate the quality of your models?
- What was the simplest model that you tried? What's the performance (score) of that simplest model?
- What's your best model so far? What's the performance (score) of that best model?
- How does your best model compare with your simplest model?

Introduction to Project and data for SVM and logistic regression models

We are using data from the following Kaggle file:
https://www.kaggle.com/joniarroba/noshowappointments

Joni Hoppen put together a csv file for people who did and did not show up to their appointments in Brazil. Data was organized into 15 different columns/variables. From the data of 110,527 people, we sampled 1,000 for the SVM and logistic regression models.

In our data for these two Machine Learning Models, we had two variables. The x variable for our analysis was the patients who did not show up to appointments, while the y variable for our analysis were the patients who did.

How did we balance the data? Check out our code!

In [8]:
```
n_rows_to_sample = 1000
```

In [9]:
```
no_show_raw_df = raw_df[raw_df['No-show']=='Yes']
print(no_show_raw_df.shape)
no_show_raw_df = no_show_raw_df.sample(n = n_rows_to_sample, random_state = 0)
print(no_show_raw_df.shape)
```

```
(22319, 14)
(1000, 14)
```

In [10]:
```
show_raw_df = raw_df[raw_df['No-show']=='No']
print(show_raw_df.shape)
show_raw_df = show_raw_df.sample(n = n_rows_to_sample, random_state = 0)
print(show_raw_df.shape)
```

```
(88208, 14)
(1000, 14)
```

In [11]:
```
reduced_raw_df = no_show_raw_df.append(show_raw_df)
reduced_raw_df.shape
```

Out[11]: (2000, 14)

In [12]:
```
reduced_raw_df.head(2)
```

Out[12]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handc
99054	4.852530e+11	5733501	F	2016-05-24T13:57:02Z	2016-06-06T00:00:00Z	21	DA PENHA	0	0	0	0	
110482	5.944225e+14	5639147	F	2016-04-29T08:45:44Z	2016-06-06T00:00:00Z	43	RESISTÊNCIA	1	0	0	0	

Support Vector Machine
Classification Report

	Precision	Recall	F1-Score	Support
Yes	0.56	0.65	0.60	246
No				254
Accuracy			0.57	500
Macro Avg	0.58	0.58	0.57	500
Weighted Avg	0.58	0.57	0.57	500

The model accuracy for my SVM was 0.574.

Support Vector Machine
Code

In [25]:
```
# Support vector machine linear classifier
from sklearn.svm import SVC

model = SVC(kernel='linear')
model.fit(X_train, y_train)
```

Out[25]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

In [26]:
```
# Model Accuracy
print('Test Acc: %.3f' % model.score(X_test, y_test))
```

```
Test Acc: 0.574
```

In [27]:
```
# Calculate classification report
from sklearn.metrics import classification_report

predictions = model.predict(X_test)
print(classification_report(y_test, predictions,
                            target_names=target_names))
```

```
              precision    recall  f1-score   support

         Yes       0.56      0.65      0.60       246
          No       0.60      0.50      0.55       254

    accuracy                           0.57       500
   macro avg       0.58      0.58      0.57       500
weighted avg       0.58      0.57      0.57       500
```

Logistic Regression

	Precision	Recall	F1-Score	Support
Yes	0.62	0.70	0.66	200
No	0.66	0.56	0.61	200
Accuracy			0.64	400
Macro Avg	0.64	0.64	0.63	400
Weighted Avg	0.64	0.64	0.63	400

Decision Tree

	Precision	Recall	F1-Score	Support
Show up at the clinic	0.80	0.92	0.86	25508
Not show up at the clinic	0.30	0.14	0.19	6587
Accuracy			0.76	32095
Macro avg	0.55	0.53	0.52	32095
Weight avg	0.70	0.76	0.72	32095

Random Forest

	Precision	Recall	F1-score	Support
Show up at the clinic	0.81	0.91	0.86	25508
Not show up at the clinic	0.31	0.16	0.21	6587
Accuracy			0.76	32095
Macro avg	0.56	0.53	0.53	32095
Weighted avg	0.70	0.76	0.72	

Confusion Matrices (Decision Tree & Random Forest)

Actual

Predicted	Show-up	No-show
Show-up	**23424**	2084
No-show	5688	**899**

Decision Tree

Actual

Predicted	Show-up	No-show
Show-up	**23223**	2285
No-show	5563	**1024**

Random Forest

- True positive: these are the patients that showed up at the clinic that were correctly identified by the algorithm.
- False Negative: when patients showed up at the clinic, but the algorithm said they did not.
- False Positive: patients that did not show up at the clinic, but the algorithm says they did.
- True Negative: these are the patients that did not show up at the clinic that were correctly identified by the algorithm.