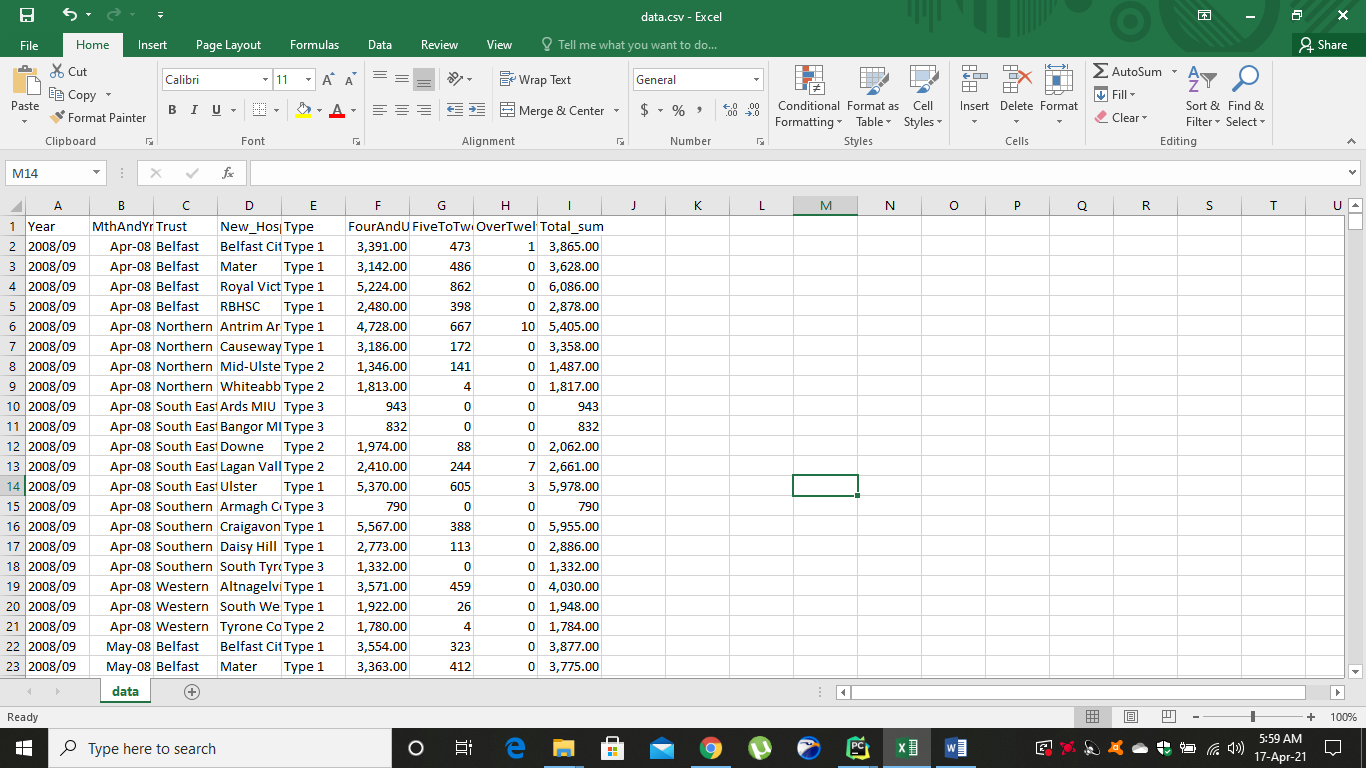
**PREDICTION OF PATIENT WAIT TIME**

**Problem Statement:**

In today’s world, it can be seen as whenever you arrived in hospitals placed at different locations, you may have to wait for your number. In this particular report, we going to identify either patient have to wait above 4 hours or not, in order to get his/her number. For this, we going to make a prediction model that will help us to identify that either the patient has to wait above 4 hours or not.

**Data-set Snippet:**

****

So in the given data-set, we have following columns

1. Year -> *Represents Year*
2. MthAndYrCode -> *Represent Month & Year*
3. Trust -> *Represents Trust*
4. New\_Hospital -> *Represents Hospital*
5. Type -> *Represents Department*
6. FourAndUnder\_sum -> *Represents number of patient waited in emergency deparment 4hrs and less*
7. FiveToTwelve\_sum -> *Represents number of patient waited in emergency deparment 5-12 hrs*
8. OverTwelve\_sum -> *Represents number of patient waited in emergency deparment 12 hrs*
9. Total\_sum -> *Represents total number of patients*

**Methodology:**

First of all, we have to understand the dataset. The current dataset does not contain target class. We have to derived our target class first. For this, let have a look at some column / features.

***“FourAndUnder\_sum”*** column tells us the number of patients that waited equal to or less than 4 hours. So, this will help us to accumulate our target. But we can’t use this column directly.

Now we will calculate chances in terms of patients that waited equal to or under 4 hours. To do this, we will be using ***“Total\_sum”*** column.

Now the formula to accumulate chances is give below:

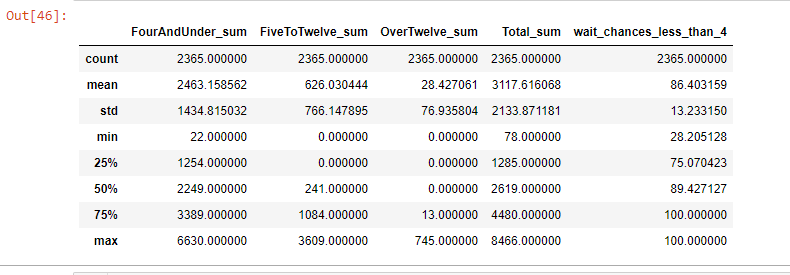
***“wait\_chances\_less\_than\_4”*** = (***“FourAndUnder\_sum” / “Total\_sum”) \* 100***

Now it’s time to derived our target class “Waited under 4 hours”. The target class holds two values. 0 or 1.

“1” means that patient had waited under 4 hours.

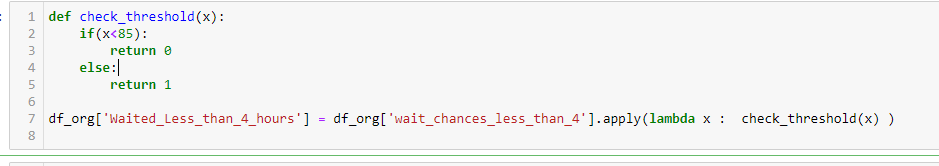
“0” means that patient had waited above 4 hours.

We need to choose a threshold to declare either patient had waited above 4 hours or less 4 hours for every single record by seeing ***“wait\_chances\_less\_than\_4”.*** To have a particular threshold, we will see the average value of ***“wait\_chances\_less\_than\_4”.***



So to be on safe side, we can select a threshold of 85%. This means that if a single record contains 85% or above chances in ***“wait\_chances\_less\_than\_4”*** then we will put “1” in our target class which represents that on average patient had waited under 4 hours. Similarly, if the chances are below 85%, we will put “0” in our target class which represents that on average patient had waited above 4 hours.

So by using a lambda function in python, we have derived our target class.

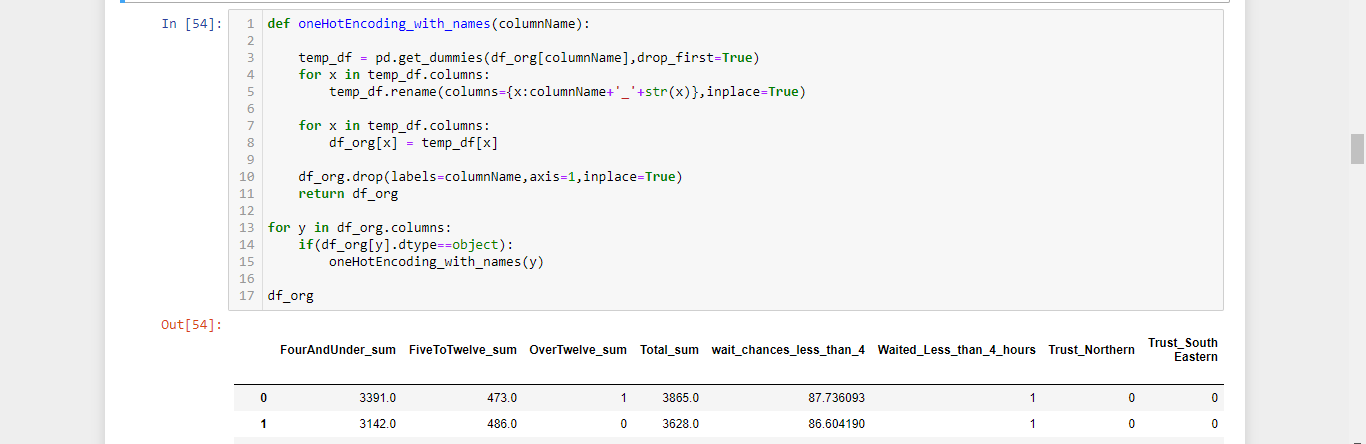


Now to avoid multi-collinearity, we will drop ***“wait\_chances\_less\_than\_4”, “FourAndUnder\_sum”, “Total\_sum”, “FiveToTwelve\_sum” and “OverTwelve\_sum”.***

Now in this way, we have made our data dependent on our target class ***“Waited\_Less\_than\_4\_hours”*** on the other features and removed the element of number of patients.

**Model & Evaluation:**

The current problem is a classification problem. First of all, we have to identify categorical columns. For example, “type”, “new hospital” etc. We will use one hot encoding to transform all categorical columns into numerical columns.



Once this is done, we will be using ***Pearson Correlation*** to check the correspondence of all features with respect to target class. By using correlation function of pandas’ data-frame, we built a correlation matrix.

By checking out different ranges for correlation, we have finally set the range from -0.2 to +0.2. Any feature correlation with target class lies in this range will be omit / neglected. After applying this methodology, we got following features:

'Waited\_Less\_than\_4\_hours',

'Trust\_South Eastern',

'New\_Hospital\_Antrim Area',

'New\_Hospital\_Mater',

'New\_Hospital\_Royal Victoria',

'New\_Hospital\_Ulster',

'Type\_Type 2',

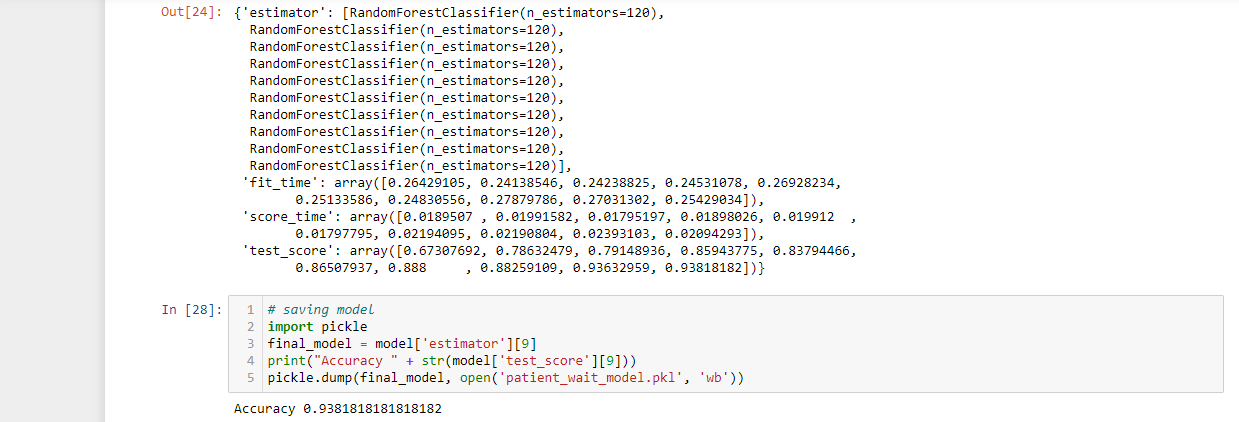
'Type\_Type 3'

Now we can easily see that the top one is target class itself as its correlation will be 1 with itself. Ignoring this, we can see other features. Now it’s time to train model.

As this is the classification problem and the features don’t show some much good correlation, we will be using random forest as random forest works best in such good conditions. But there is a way of training model in such a way that we gain high accuracy at end.

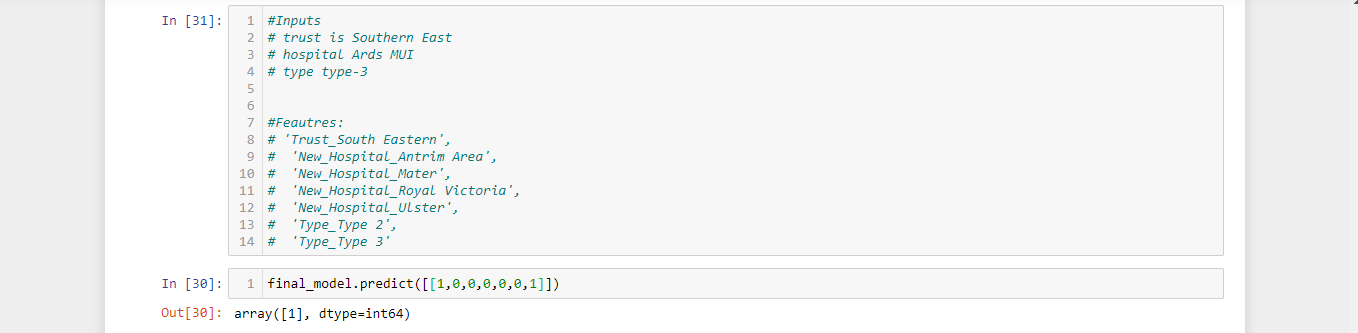
So what we will do that we will use cross validation (cv=10) and put the scoring metric “F-1”. The “F-1” scoring metric is a crucial element of confusion matrix. It helps us to determine recall and precision score simultaneously.

So after applying this methodology, we have achieved 10 different random forest models.



Out of 10, f-1 score of 7 models is above 80%. Which indicates that we have done our work in good shape with respect to data modeling and feature selection. The best model has 93.8% accuracy.

We will choose this model to make predictions in real life.



Suppose a patient walks in south eastern trust. The hospital he chooses is “Ards MIU”. And the department is type 3. So after analyzing this input, we provide this input in terms of 0 & 1 to model. The model predicts “1” which means patient will wait under four hours.

*Thankyou*