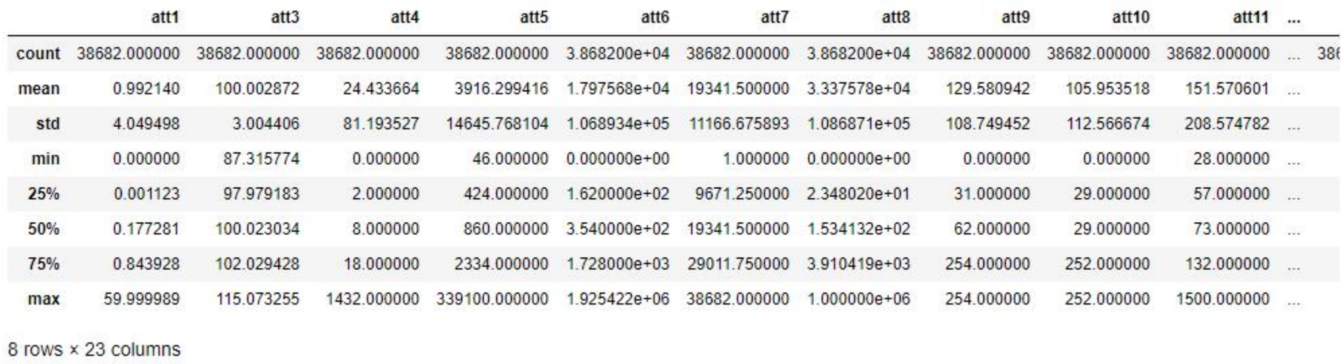
apply supervised and unsupervised techniques on a dataset:

1- Preprocessing and feature selection:

The first step was to load the data and read it through the Pandas library. After that, we used the describe() command to take an insight of the data. Then we showed the first five rows in the dataFrame with the command head.

Describe ():

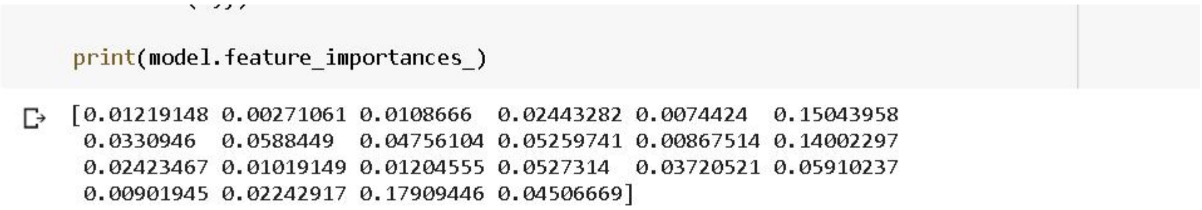


	att1	att3	att4	att5	att6	att7	att8	att9	att10	att11	...	
count	38682.000000	38682.000000	38682.000000	38682.000000	3.868200e+04	38682.000000	3.868200e+04	38682.000000	38682.000000	38682.000000	...	38
mean	0.992140	100.002872	24.433664	3916.299416	1.797568e+04	19341.500000	3.337578e+04	129.580942	105.953518	151.570601	...	
std	4.049498	3.004406	81.193527	14645.768104	1.068934e+05	11166.675893	1.086871e+05	108.749452	112.566674	208.574782	...	
min	0.000000	87.315774	0.000000	46.000000	0.000000e+00	1.000000	0.000000e+00	0.000000	0.000000	28.000000	...	
25%	0.001123	97.979183	2.000000	424.000000	1.620000e+02	9671.250000	2.348020e+01	31.000000	29.000000	57.000000	...	
50%	0.177281	100.023034	8.000000	860.000000	3.540000e+02	19341.500000	1.534132e+02	62.000000	29.000000	73.000000	...	
75%	0.843928	102.029428	18.000000	2334.000000	1.728000e+03	29011.750000	3.910419e+03	254.000000	252.000000	132.000000	...	
max	59.999989	115.073255	1432.000000	339100.000000	1.925422e+06	38682.000000	1.000000e+06	254.000000	252.000000	1500.000000	...	

8 rows × 23 columns

When looking at table we see some feature have large value of standard deviation that mean it is so far from the Mean value and it may effect in the class.

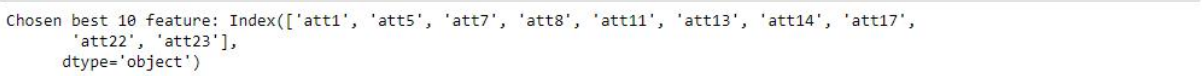
We can use built in function to select better Features to use and here I used two method using sklearn, the First one is Decision trees, and the ensemble methods for this we use (feature_importances_) Function that work by dividing the data into subsets that belong to a single category. The tree will continue building various subsets until it understands and represents the relationship of the variables to the target. (Scikit-learnn.d)



```
print(model.feature_importances_)

[0.01219148 0.00271061 0.0108666  0.02443282 0.0074424  0.15043958
 0.0330946  0.0588449  0.04756104 0.05259741 0.00867514 0.14002297
 0.02423467 0.01019149 0.01204555 0.0527314  0.03720521 0.05910237
 0.00901945 0.02242917 0.17909446 0.04506669]
```

The second one is Recursive Feature Elimination or RFE this method is not very different from the previous one, as the data is divided into subsets and each time removing features until the desired number remains.



```
Chosen best 10 feature: Index(['att1', 'att5', 'att7', 'att8', 'att11', 'att13', 'att14', 'att17',
       'att22', 'att23'],
      dtype='object')
```

Afte that I chose the target and feature from data and indicated them as x,y when I prepare x I drop two feature first one is class cause that our target the second is att2 cause it contain catogrical value, after that I split the data to train and test to ensure that will not gain miss-classify when calculate the accuracy .

A- Supervised :

2- choose the model and calculate accuracy

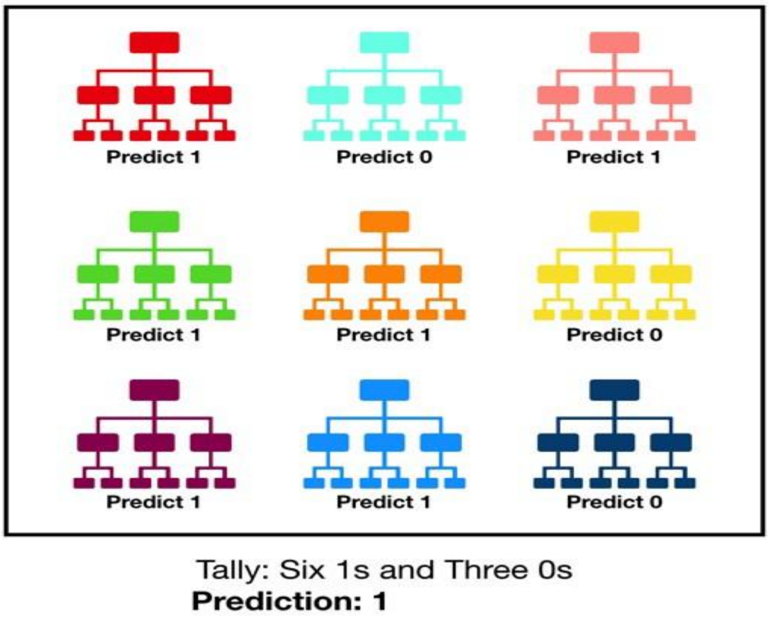
I used about 2 type of classification I will compare between them in table :

For calss 0	precision	recall	f1-score	accuracy	TP	TN
LogisticRegression	0.96	1.0	0.98	0.96	7393	17
RandomForestClassifier	1.0	1.0	1.0	1.0	7410	327
For calss 1	precision	recall	f1-score	accuracy		
LogisticRegression	0.29	0.02	0.04	0.96	7393	17
RandomForestClassifier	1.0	1.0	1.0	1.0	7403	334

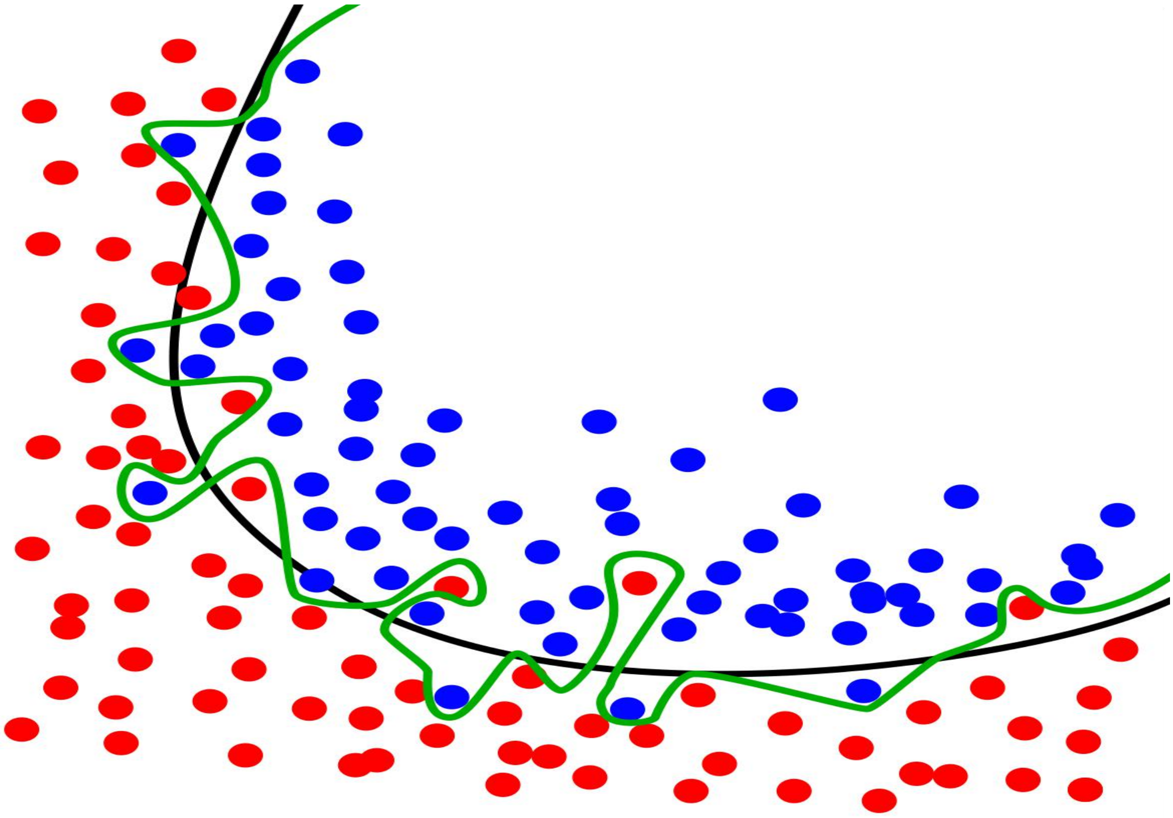
I think this is a very misleading result for many reasons may because we use RandomForestClassifier when selecting Feature cause that they give outstanding accuracy and when use LogisticRegression makes some sense but it also misleads the result, maybe the reason unbalanced data.

The (TP and TN) refer to trure positive and trure Negative How many predictions were correct and his prediction in the correct class and how many number of predictions were in error and his prediction in the wrong class, which is another measure of the accuracy of the model.

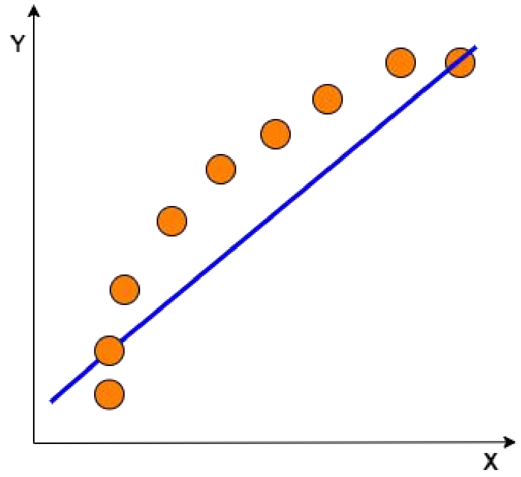
The behavor of our model (will focus on RandomForest): Random forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction .



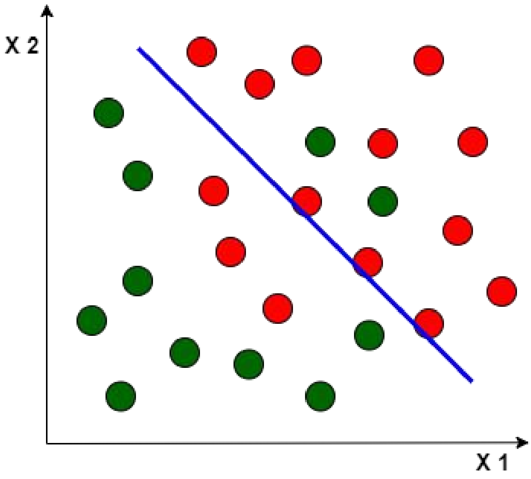
In our example this a number of trees are made and in each tree the specific expectation is taken, for example if we had 10 trees and eight trees expected the class 0 and the two trees expected the class 1, this means that the first prediction will be taken (class 0), our model works with this Way , but as you show there a problem that appear clearly when we use RandomForest the problem called overFitting (happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize.(JasonBrownlee 2016)



We see the overfitting happen when the model trace a feature without understanding the pattern, in the other hand when our model have lack of feature it make underfitting .

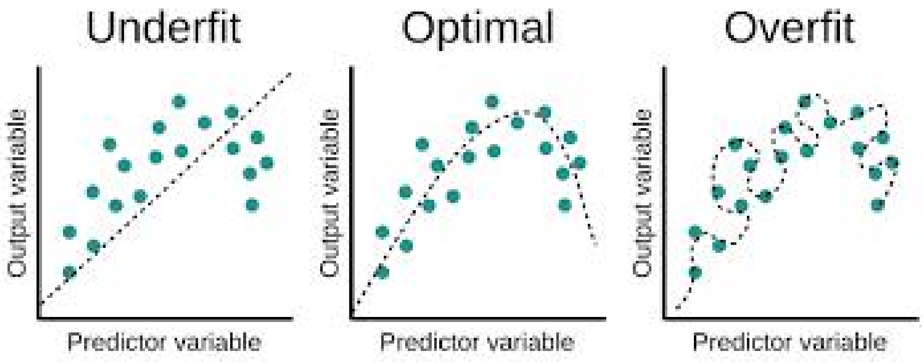


Linear Regression



Logistic Regression

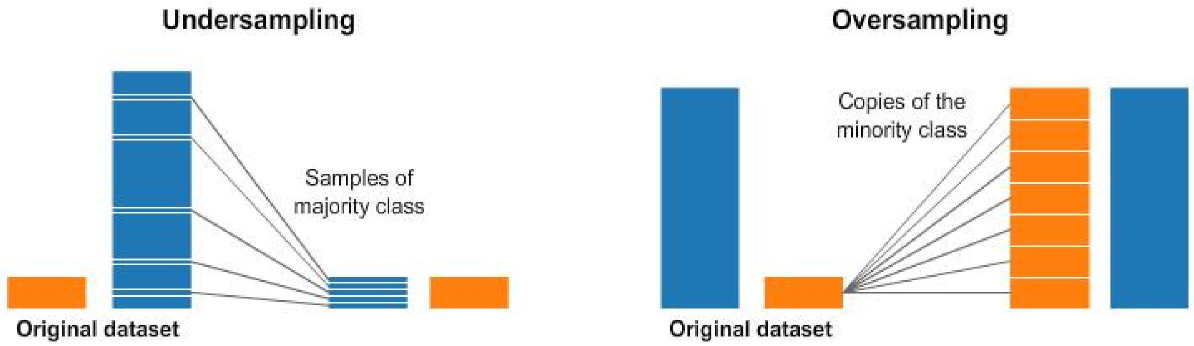
We can prevent over and under fitting by remove some features or we can add more data all that based on the result we have.



3 – use under- sampling and over-sampling to resample the data to handel with unbalanced data :

The term undersapmling refer to removing samples from the majority class.

And oversampling refer to adding more examples from the minority class .(Rafael Alencar 2019)



When I resampling data using two method the model now make sense and I feel it results was close to true value.

B- Un-Supervised:

Conclusion:

When I tested several models before to deal with the problem of unbalanced data, it was clear that the random forest is the most efficient model, and this is evident in the Confusion Matrix, and it is not make sense for the model to be 100% accurate and this high accuracy due to the imbalance of the data and therefore resorted to resampling Data using under and over sampling, I used the same model, which is the random forest, and I found that it gives excellent results, but theoretically correct, and to ensure the misleading, I divided the data also to train and test.

Reference:

Jason Brownlee 2016, *over and under fitting*, viewed 18 Apr 2021,

> https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/<

Rafael Alencar 2018, *Resampling*, viewed 18 Apr 2021,

> https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets/<

scikit-learn S n.d, *Machine learning tutorial* , viewed 18 Apr 2021,

>https://scikit-learn.org/stable/index.html/<