Final project – Python for data analysis

<u>SkillCraft1 Master Table Dataset (StarCraft II)</u>

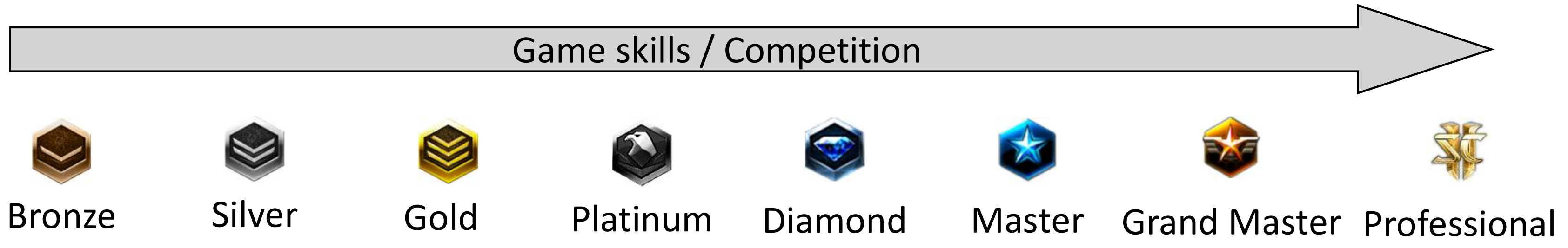
- BESSIS Hugo
- BLANOT Antoine

Table of contents

Starcraft II dataset

StarCraft II is a science fiction real-time strategy video game developed and published by Blizzard Entertainment. It is a very in-depth and complicated game. All the complexity resides in managing time, space and resources. Therefore, StarCraft II is considered as one of the hardest game to master.

Our dataset is composed of 3,395 rows (each row representing one unique player) and 21 columns representing players stats (each column represents a different statistic). Each player (or row) is assigned to its league rank (Bronze, Silver, Gold, Platinum, Diamond, Master, Grand Master and Professional).

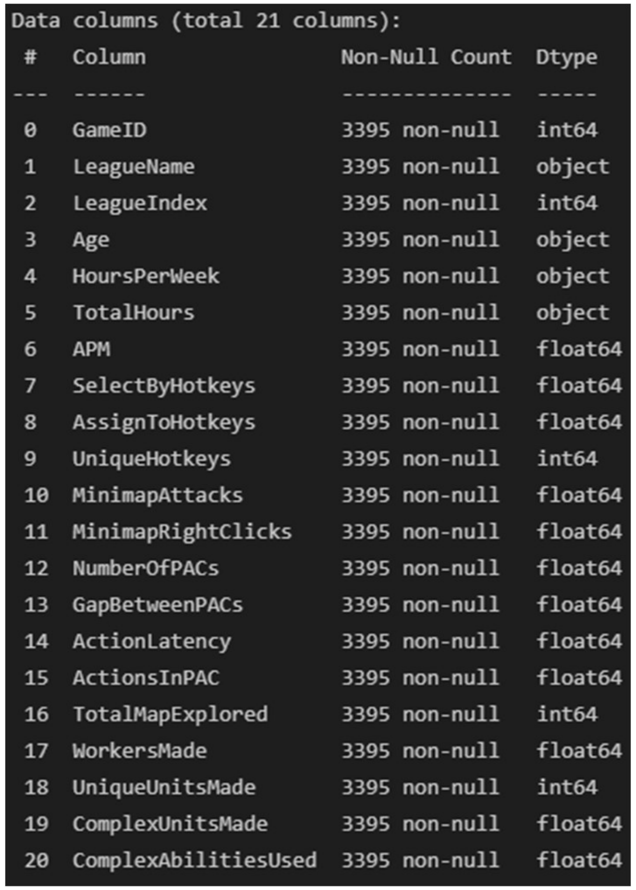


Starcraft II dataset

We have 21 columns in this dataset. These columns are composed of **in-game stats and information** about the players.

Some of them are **mechanical skills statistics** such as ActionLatency (time gap between two actions) or GapBetweenPACs (time gap between Percepion Action Cycles), and some of them are just **informative statistics** such as TotalHours (total hours played by the player), HoursPerWeek or GameID.

Almost all in-game stats are **numerical** type values (integer or float values) except of GameID (unique value representing a gamer ID) and League Index (going from 1 to 8 representing the rank of the player, the more it is high, the more the league is competitive).



```
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   GameID                3395 non-null   int64
 1   LeagueName            3395 non-null   object
 2   LeagueIndex           3395 non-null   int64
 3   Age                   3395 non-null   object
 4   HoursPerWeek          3395 non-null   object
 5   TotalHours            3395 non-null   object
 6   APM                   3395 non-null   float64
 7   SelectByHotkeys       3395 non-null   float64
 8   AssignToHotkeys       3395 non-null   float64
 9   UniqueHotkeys         3395 non-null   int64
 10  MinimapAttacks        3395 non-null   float64
 11  MinimapRightClicks    3395 non-null   float64
 12  NumberOfPACs          3395 non-null   float64
 13  GapBetweenPACs        3395 non-null   float64
 14  ActionLatency         3395 non-null   float64
 15  ActionsInPAC          3395 non-null   float64
 16  TotalMapExplored      3395 non-null   int64
 17  WorkersMade           3395 non-null   float64
 18  UniqueUnitsMade       3395 non-null   int64
 19  ComplexUnitsMade      3395 non-null   float64
 20  ComplexAbilitiesUsed  3395 non-null   float64
```

Preprocessing

Null values :

In this dataset, null values are represented by a « ? ». There were 57 players with at least one null value, and all these null values were located on the Age, TotalHours and HoursPerWeek columns. The players concerned by this null value are 55 out of 57 professional players. To have coherent data without losing the professional players information and statistics, we chose to **replace the null values by the mean of its column**.

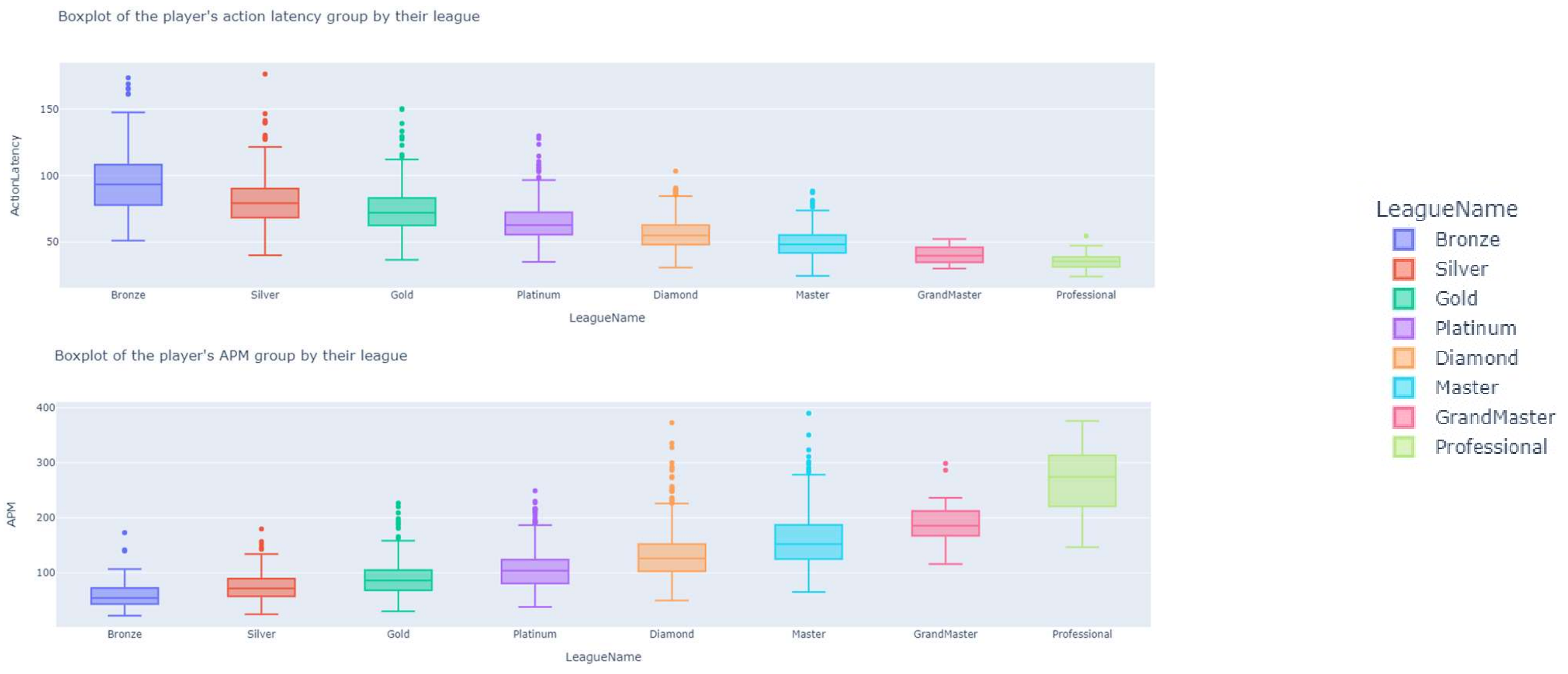
Incoherent data :

We found some incoherent data that we removed.

- Players who played for more than 168 hours a week (24 hours a day so non-stop)
- Players who played more than 100,000 hours (more than 11 years of playing non-stop but the game has been out 11 years ago…)

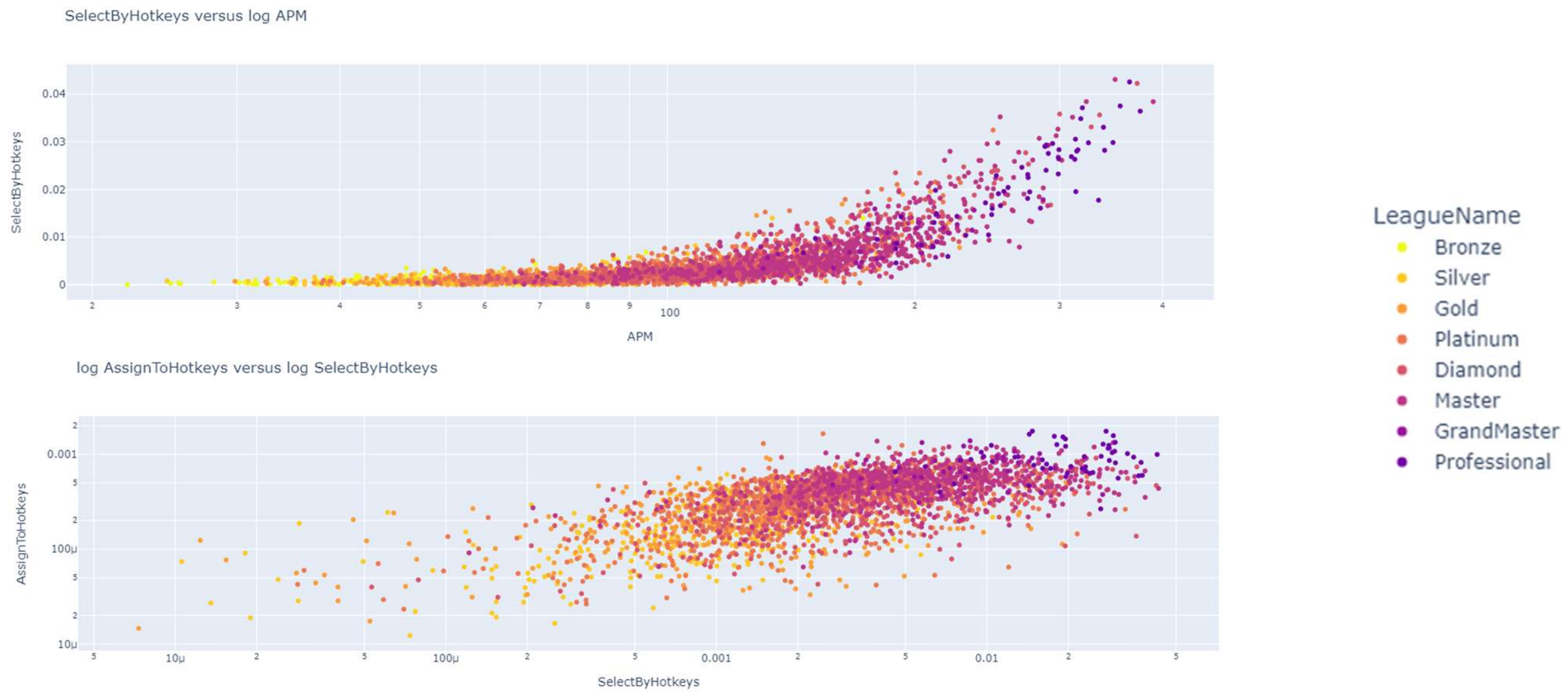
First data analysis

To explore our data, we did some visualization using plotly.express a module of plotly. We try to find some basic linear correlation between league ranks and each statistic. Here are some of the data visualization we did:



First data analysis

We also did graphs comparing two statistics for each player. We can see that some of them, like APM and SelectByHotKeys for example are representative of a league rank.

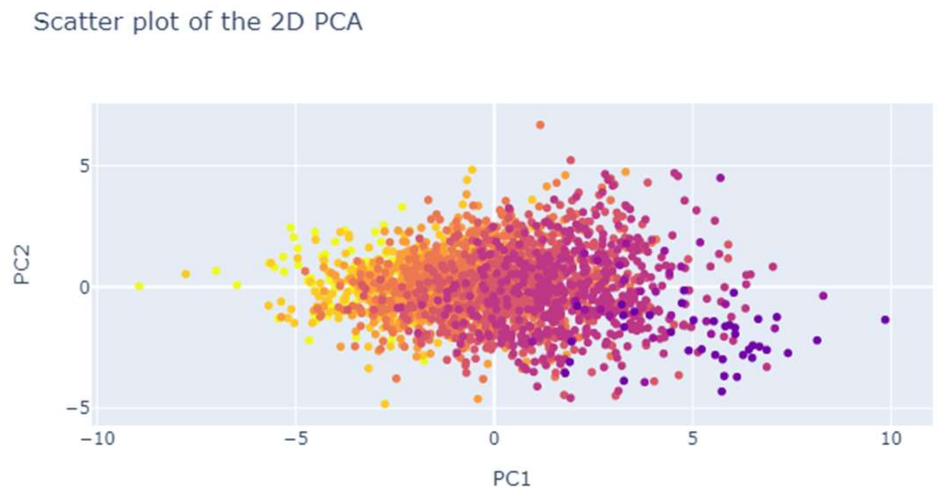


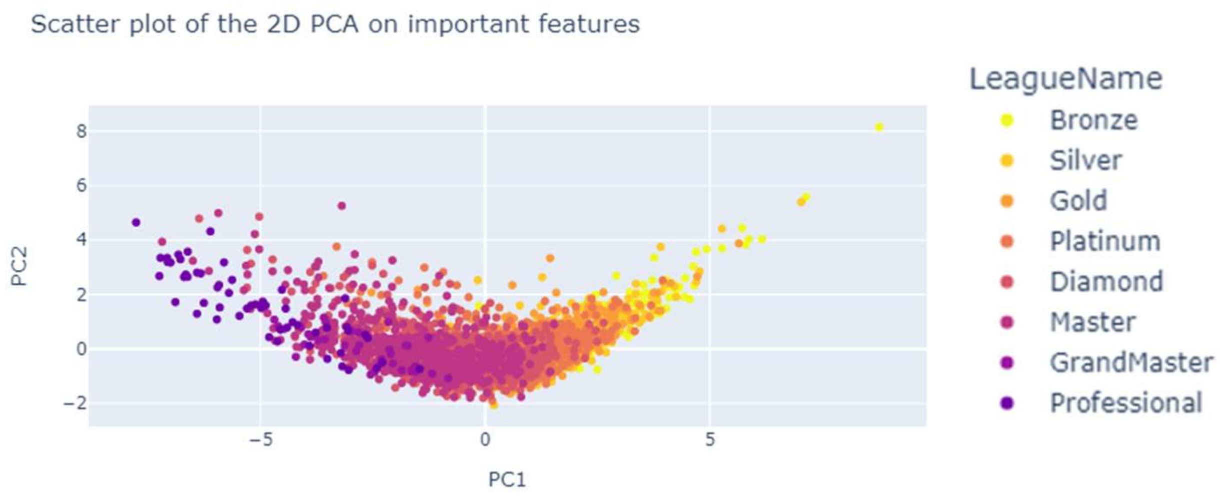
Important features and PCA

From our first data analysis, we pointed out 5 features that we think explains the most our dataset.

```
important_features = ['APM', 'SelectByHotkeys', 'AssignToHotkeys', 'ActionLatency', 'GapBetweenPACs']
```

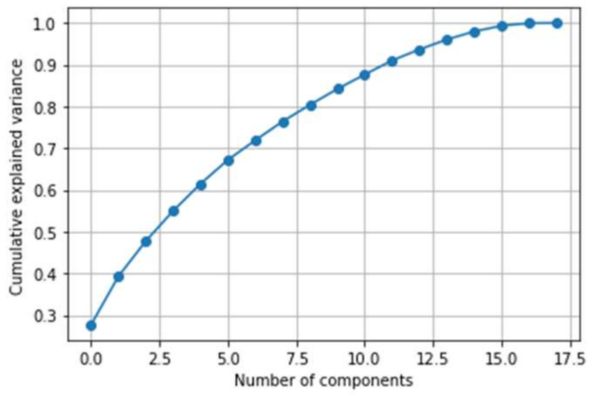
Since we have more than 2 features, we can not represent our data in a plan. To be able to graph our data, we will use PCA (Principal Component Analysis). On the graphs below (showing the PCA on all the features and the PCA on the 5 important features), we see that our data is gathered and compact. But still, we can observe a trend (bronze data is globally far from professional data). But from these graph, we can expect that classification will not be easy.





Important features and PCA

An explained variance ratio graph shows us how much information we manage to retrieve from a PCA depending on the number of components we want. For making 2D plots, we only take 2 components. Here is the explained variance ratio graph of our dataset on all the features:

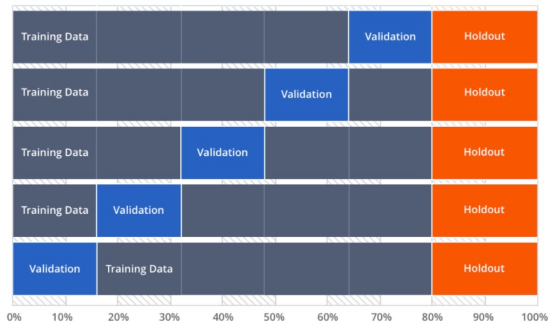


After checking both PCA 2D graph and the explained variance ratio graph, we concluded that using a PCA on this dataset is not representative of the real dataset (around 40%) and that the components are not much correlated to each other. These information are an indicator meaning that it will be hard for us to do a proper and a precise classification.

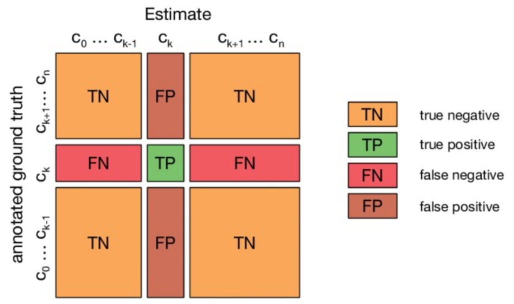
Machine Learning tools

For the next part we imported some Machine Learning tools that will help us to get the best model and prediction as possible. These tools can be found in the sklearn package and are:

- The cross-validation (k-folds method) to avoid overfitting and get a more precise score of our models
- The grid search to find the best hyperparameters for our models
- The multi-class confusion matrix to see where exactly our models make the most errors
- The accuracy score to compare our models between each other.



Cross-validation (k-folds method)

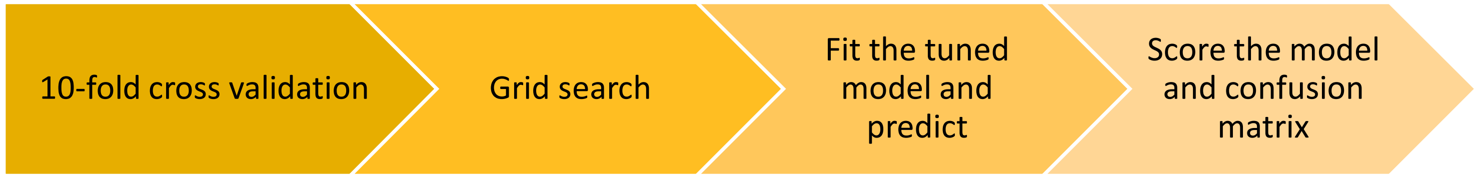


Multi-class confusion matrix

Classification models

For making our models, we try the best methods we knew for the classification. These are SVM for classification (also called SVC), the KNN classifier model, the Random Forest classifier model and the LDA model. We applied the model using tools defined on the previous slide. For each model, our methodology was:

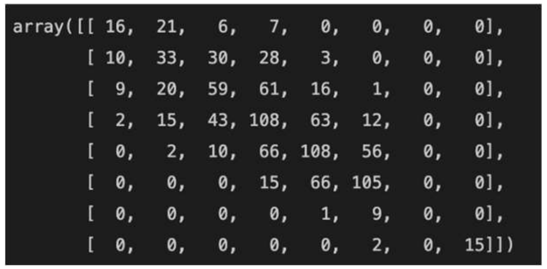
1. Apply the model using a 10-fold cross validation, to get an average score of the model without tuning
2. Apply a grid search to find the best hyperparameters
3. Fit the tuned model (with best parameters) to the train set and predict the results with the test set
4. Score the model (using accuracy) and print the confusion matrix


10-fold cross validation
Grid search
Fit the tuned model and predict
Score the model and confusion matrix

Classification models

Here are the results of the accuracy score of each model on the test set:

- KNN Classifier model : around 38%
- SVC model: around 39%
- LDA model: around 40%
- Random Forest Classifier model: around 42%



```
array([[ 16,  21,   6,   7,   0,   0,   0,   0],
       [ 10,  33,  30,  28,   3,   0,   0,   0],
       [  9,  20,  59,  61,  16,   1,   0,   0],
       [  2,  15,  43, 108,  63,  12,   0,   0],
       [  0,   2,  10,  66, 108,  56,   0,   0],
       [  0,   0,   0,  15,  66, 105,   0,   0],
       [  0,   0,   0,   0,   1,   9,   0,   0],
       [  0,   0,   0,   0,   0,   2,   0,  15]])
```

Multi-class confusion matrix on the Random Forest Classifier

As you can see, all these models have a low accuracy (around 40%) which is obviously not enough at all for a decent ranking prediction. Also, we remark from the confusion matrix that most of the wrong classifications are made between middle leagues (gold, platinum, diamond). In order to avoid these mistakes, we will use unsupervised learning to get more information on this dataset.

Supervised and unsupervised learning

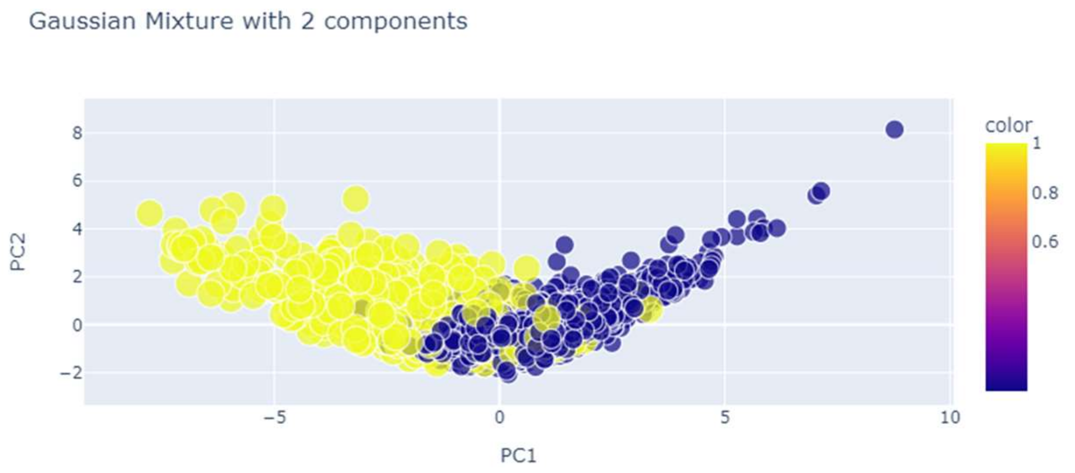
Doing classification over eight groups seems hard with the dataset we have. That is why we had the idea to reduce the number of predicted group to get a better prediction model.

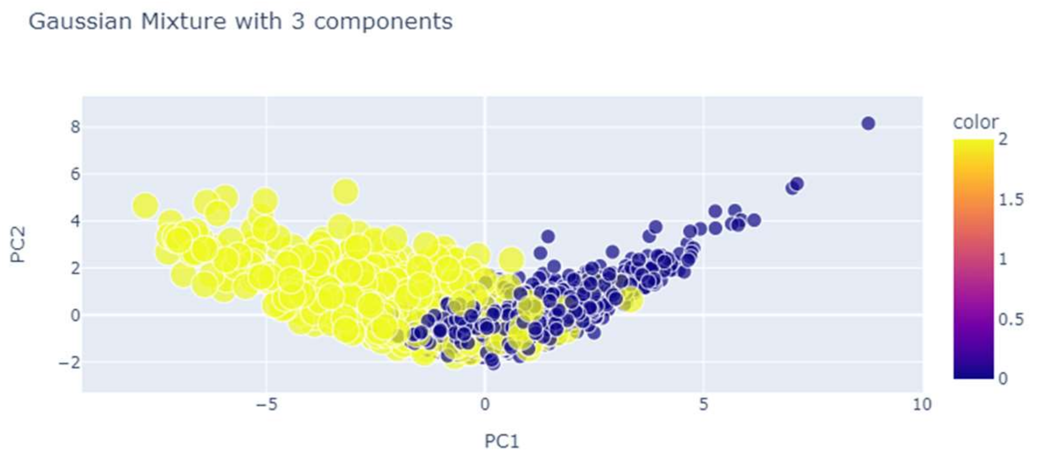
In order to do that, we will try two ways of making prediction model:

- The first one is by using **unsupervised learning** models with a number of group we want to have.
- The second way to do that is by using **supervised learning technics with predefined groups** and labels using our knowledge on the domain.

Unsupervised Learning : Gaussian mixtures

For the unsupervised learning part, we decided to use the Gaussian mixture model on the important features. A Gaussian mixture model takes as input a number which represents the number of group we want the algorithm to find. Also, to see the results of unsupervised learning techniques we can either make a graph or apply some evaluation function like standard deviation. We decided to use a PCA in order to be able to see the result of the gaussian mixture.

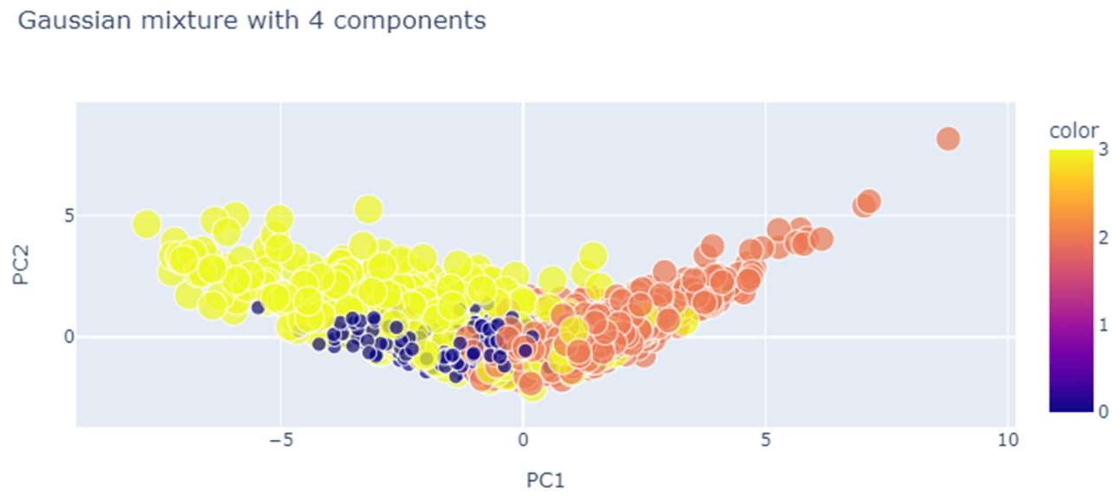




The results are not good, the algorithm itself is not able to make three distinct groups in our dataset… Let’s try with 4 components.

Unsupervised Learning : Conclusion

Here is output we got for 4 components:



The output of a Gaussian mixture with 3 and 4 components are not reliable because the clusters are too close to each other. Without zooming in, we can't distinguish more than two and three different clusters out of the three and four we have. Also, the variance within a cluster (for example cluster 3, the yellow one on the Gaussian mixture with 4 components) is very high, which again, shows us that such gaussian mixtures are not that effective.

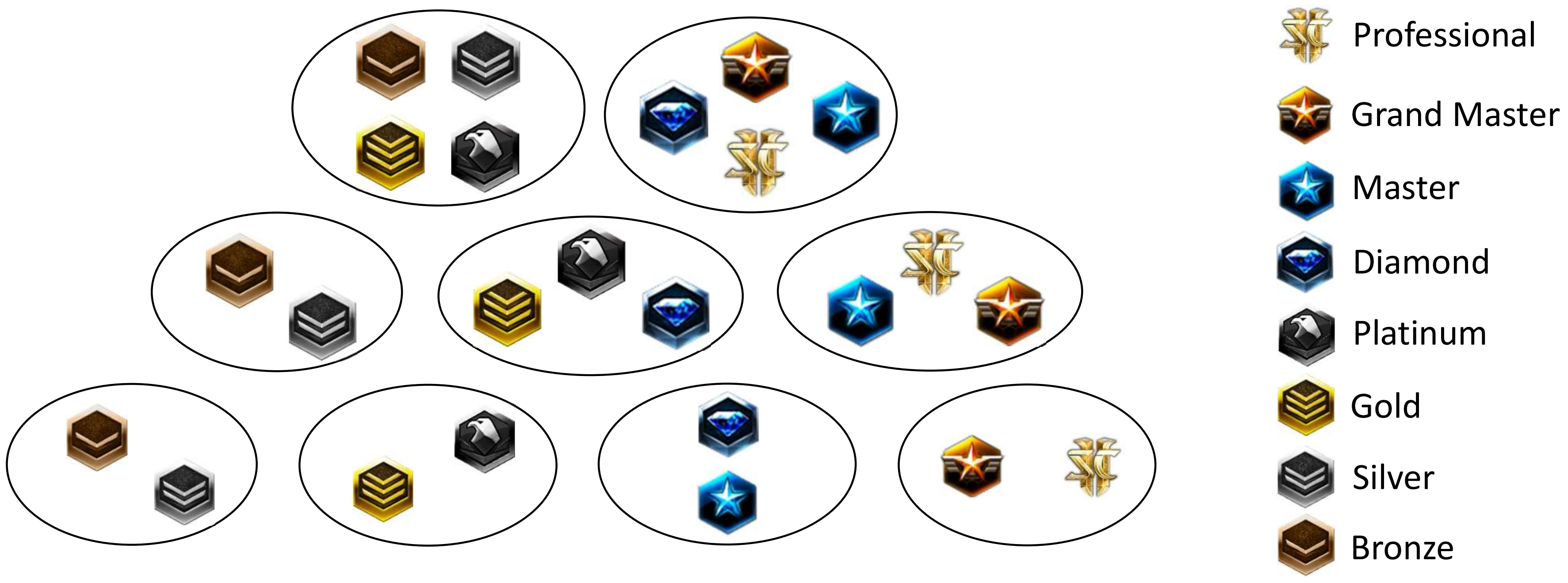
Supervised Learning : Grouping

Now we try to group the leagues into two, three and four groups in a consistent way to have similar levels of player. After making some models like this, we will be able to predict to which of these groups the player belongs to.
The different groups we made for the models are:

In 2 groups:

In 3 groups:

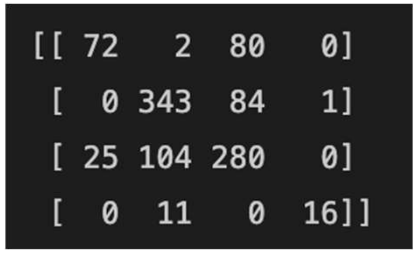
In 4 groups:



Supervised Learning : Modeling

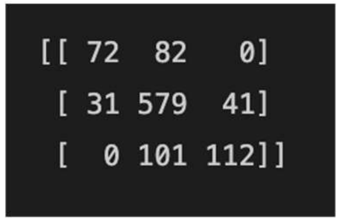
Since the Random Forest classifier was the best model when we wanted to predict the rank, we will choose it as our model for predicting groups. We will use the same methodology than before to apply our models (cross-validation, grid search, fit and results).
Our results are :

- Random Forest Classifier on **4 predefined groups**: around 69%
- Random Forest Classifier on **3 predefined groups**: around 75%
- Random Forest Classifier on **2 predefined groups**: around 80%



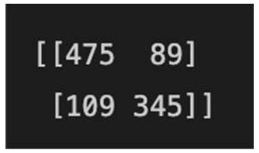
```
[[ 72   2  80   0]
 [  0 343  84   1]
 [ 25 104 280   0]
 [  0  11   0  16]]
```

Confusion matrix on 4 predefined groups



```
[[ 72  82   0]
 [ 31 579  41]
 [  0 101 112]]
```

Confusion matrix on 3 predefined groups



```
[[475  89]
 [109 345]]
```

Confusion matrix on 2 predefined groups

Conclusion : supervised learning

For grouping :

The accuracy for 3 and 4 groups are good and interesting, but the 2 grouping one is not interesting to study, even if it has the highest score. Using the 4 groups model seems to be the best since the Gold leagues is not in the same group than the Diamond league and as player, we know there is a high gap of level and expertise between these two leagues. The accuracy raise to **69%**.

For ranking:

The accuracy of all the model we tried do not exceed 50%. Based on this dataset, predicting the exact rank of the player is not consistent. The accuracy raise to **42%**.

Conclusion

With our dataset, we can not predict exactly the rank of any Starcraft II players. Maybe because we need other features that can explained more the rank of a player or maybe because it's not possible with our knowledge to predict it precisely.

However, with the data we have, we can try to get an idea of a player's level by grouping up the leagues together and forming new groups.

Knowing that, we can ask ourselves if the leagues in this game are well defined or not. Some of them could be irrelevant because of the gap of level between players in the same league. We could maybe imagine another ranking system with more or less leagues. Another model of ranking could give us a better rank prediction model.

Final project – Python for data analysis

Thank you for your attention!

BESSIS Hugo
BLANOT Antoine