Python for data science

MAGIC GAMMA TELESCOPE

Gasso Marc-Eddy

Année scolaire 2018-2019

sommaire

- 1/ La base de donnée Magic Gamma Télescope
- 2/ Mes objectifs
- 3/ Les diferentes features disponibles
- 4 Hyper paramètres
- 5/ Les gains de performance des differents modèles
- 6/ Conclusion

1/La base de donnée Magic Gamma Télescope

- **les donnée disponible sur :** https://archive.ics.uci.edu/ml/machine-learning-databases/magic/magic04.names

- Les données sont générées pour simuler l'enregistrement de particules gamma de haute énergie dans un télescope gamma « Cherenkov » atmosphérique basé au sol en utilisant la technique d'imagerie.

- Le télescope gamma « Cherenkov » observe les rayons gamma de haute énergie, tirant parti des radiations émises par les particules chargées. Elles sont produites dans les gerbes électromagnétiques initiées par les gammas et se développant dans l'atmosphère. .

2/ Mes objectifs

- Ma tâche consiste à évaluer et à interpréter l'erreur de classification sur une classification de particules à l'aide de l'ensemble de données du télescope MAGIC Gamma dans les dépôts UCI.

- Le jeu de données comprend 10 entités continues pour 19020 observations et une étiquette soit 'g' pour un événement gamma ou un 'h' pour un événement hadron

3/ Les différentes features disponibles

classe:	g, h gamma (signal), hadron (fond)
fLongueur:	axe continu majeur de l'ellipse [mm]
fWidth:	axe continu mineur de l'ellipse [mm]
fSize:	10 de la somme du contenu de tous les pixels [dans #phot]
fConc:	ratio continu de la somme des deux pixels les plus élevés sur fSize [ratio]
fConc1:	rapport continu du pixel le plus élevé sur fSize [ratio]
fAsym:	distance continue du pixel le plus élevé au centre, projetée sur le grand axe [mm]
fM3Long:	continu 3ème racine du troisième moment le long du grand axe [mm]
fM3Trans:	3ème racine continue du troisième moment le long d'un axe mineur [mm]
fAlpha:	angle continu de l'axe principal avec le vecteur à l'origine [deg]
fDist:	distance continue de l'origine au centre de l'ellipse [mm]

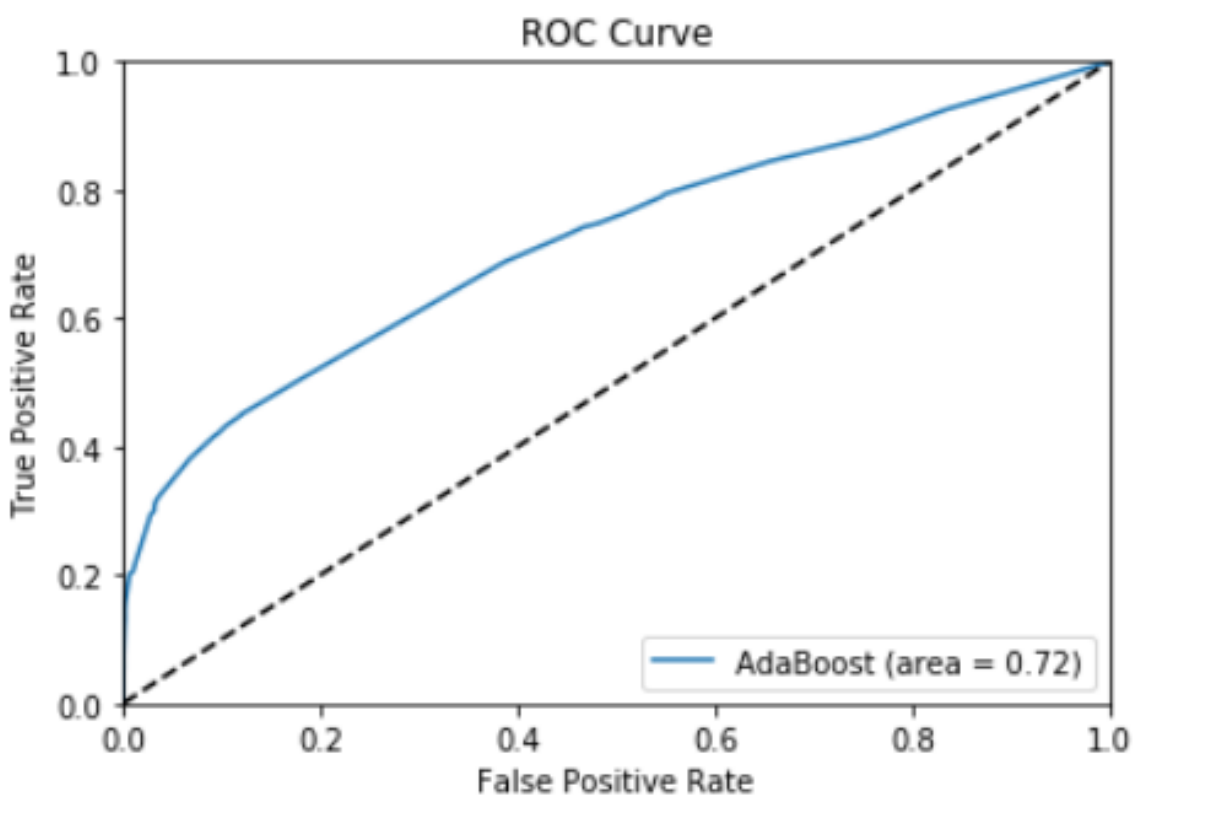
4/ Hyper paramètres

Optimisation des hyperparamètres

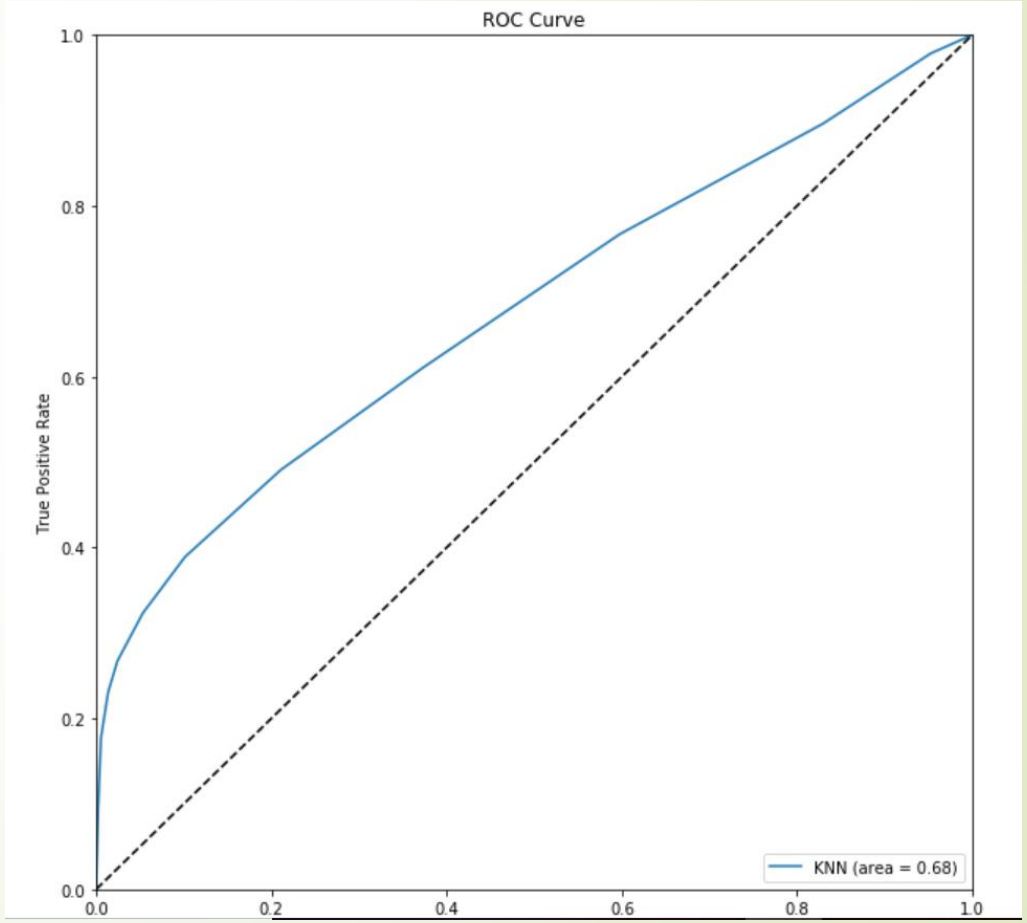
```
from sklearn import grid_search
#Utilisation du grid search avec les paramettre utiliser en cours
parameters = {  'gamma' : [0.01, 0.1, 0.5]           ,
                "probability" : [True]}
grid       = grid_search.GridSearchCV(algorithme, parameters, n_jobs=-1)
grid.fit(X_train, y_train)
print grid.best_score_, grid.best_estimator_.score(X_test, y_test)
```

```
0.85276119403 0.877696969697
```

5/ Les gains de performance des différents modèles

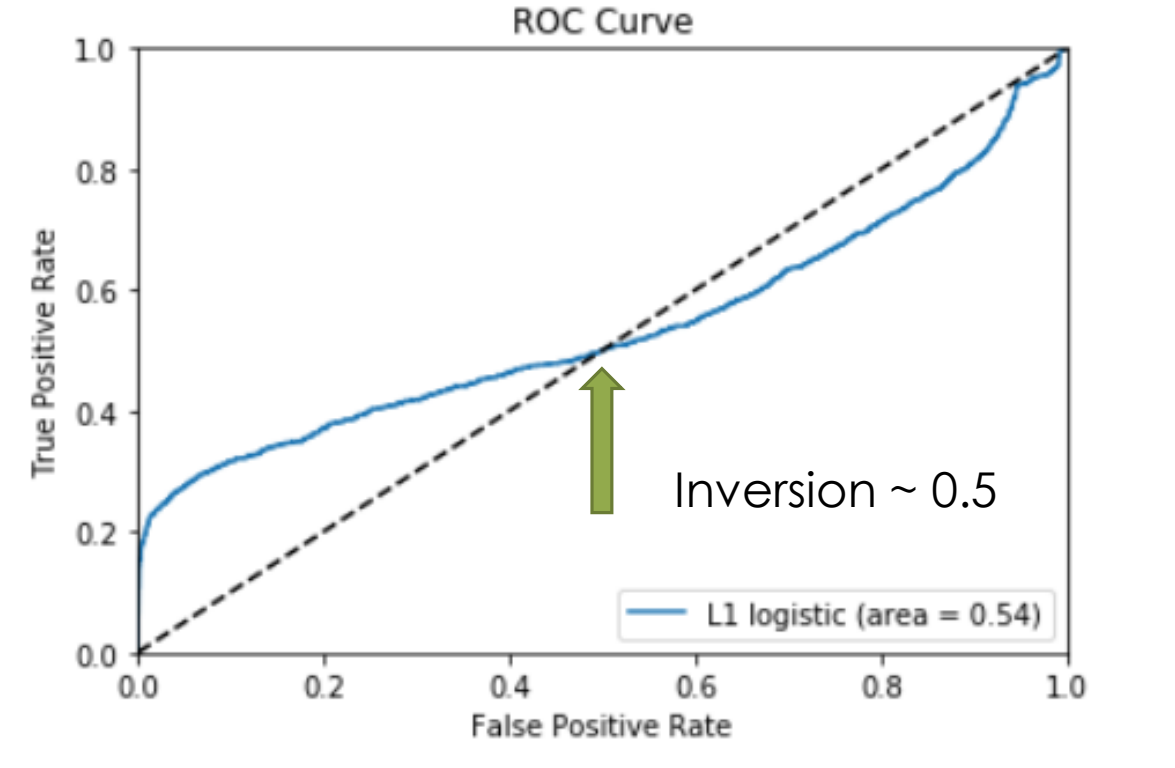


AdaBoost accuracy = 0.7396424815983176



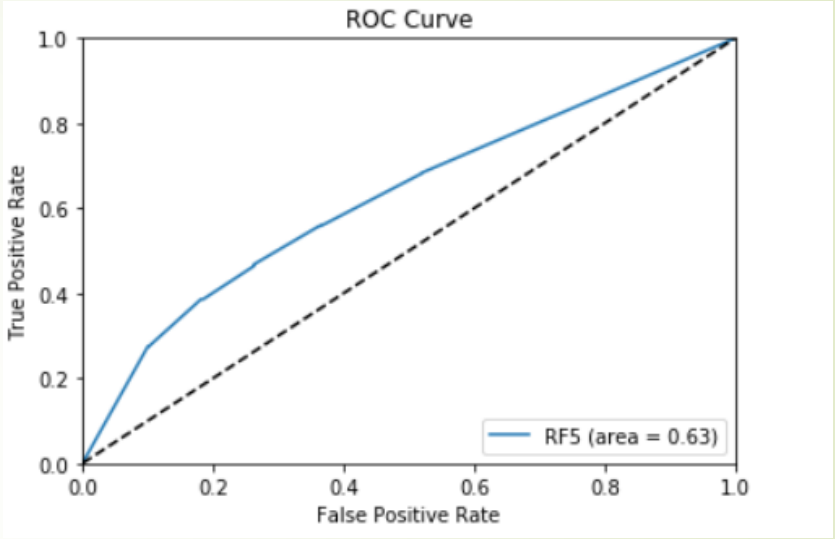
KNN accuracy = 0.720925341745531

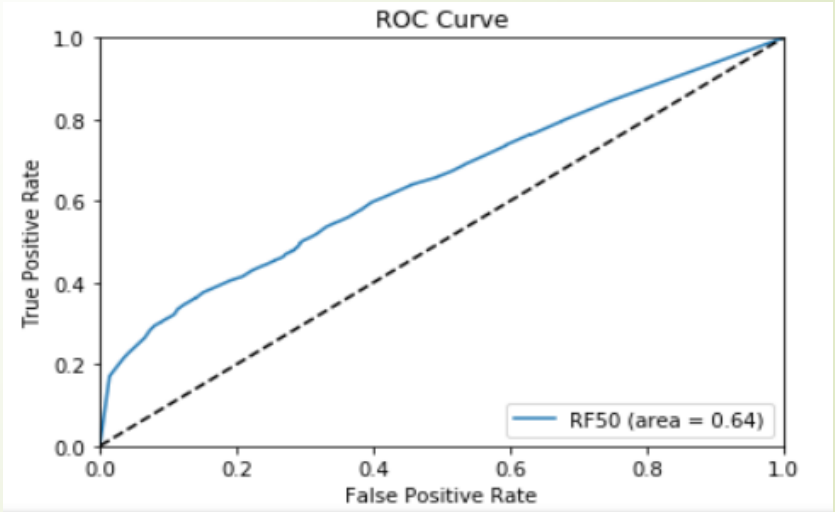
5/ Les gains de performance des différents modèles



L1 logistic accuracy = 0.7202944269190326

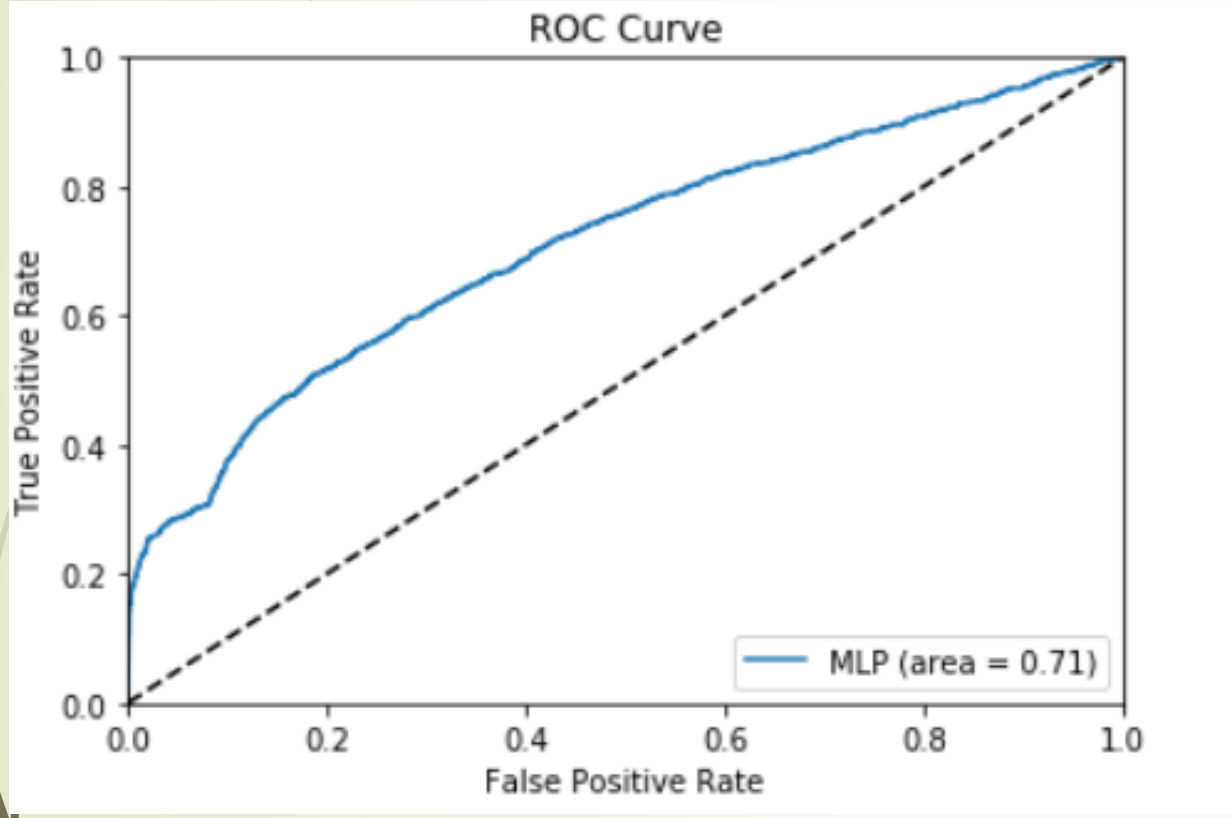
RF5 accuracy = 0.6426919032597266





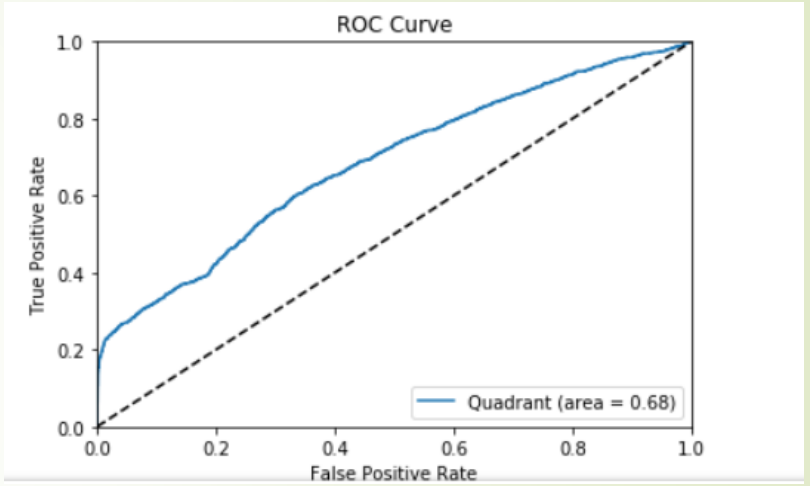
RF50 accuracy = 0.6361724500525763

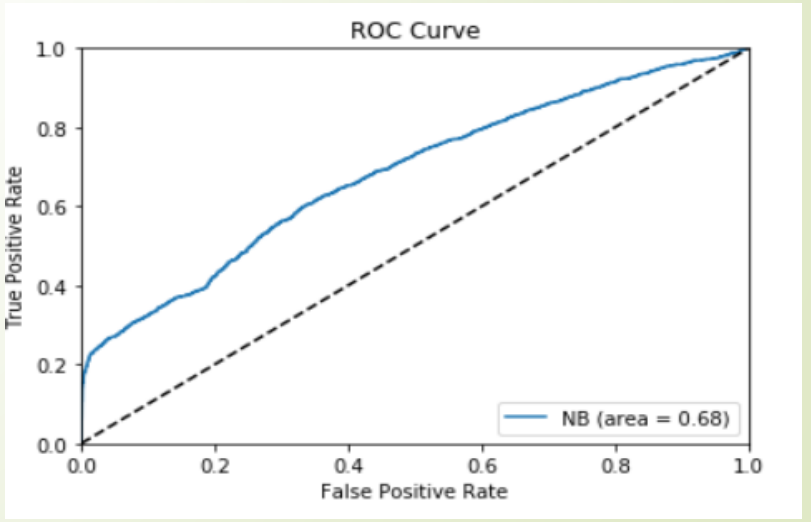
5/ Les gains de performance des différents modèles



MLP accuracy = 0.7211356466876971

NB accuracy = 0.7184016824395373





Quadrant accuracy = 0.7184016824395373

6/ Conclusion

- Pour les faibles valeurs de C ,**C <1** , forte régularisation, la précision diminue.
  → Indique une insuffisance.

- Pour des valeurs élevées de C régularisation faible, la précision reste essentiellement la même.
- Une légère diminution **pour C> 30**.
- Nous pouvons donc en conclure qu'il existe suffisamment de données pour éviter les sur-ajustements.

- Ce projet à été très enrichissent et ma permis de découvrir le télescope « Cherenkov »