POLYTECH SORBONNE MAIN5

Projet Apprentissage Statistique : Forest Cover Type Prediction

Réalisé par:

Romaric KANYAMIBWA

Boussad MERHANE

Elèves ingénieurs en mathématiques appliquées et informatiques Polytech-Sorbonne

Janvier 2019

Table of content

1 Introduction

Le projet dont nous allons discuter dans la suite de ce rapport a pour objectif la classification de différents types de forêts à partir d'observations faites sur le terrain. Il s'agit d'un problème de classification multi-classe avec 7 classes. C'est un problème qui avec l'avènement du Machine Learning et le Deep Learning a été beaucoup étudier depuis les années 2000.

L'intérêt de ce projet repose dans son application potentielle.En effet la gestion des ressources étant une problématique actuelle, ce projet avec différents modèles predictif pourrait potentiellement aider les gestionnaires des ressources naturelles d'élaborer des stratégies des gestions des différents écosystèmes sous leur responsabilité.

Dans ce projet, nous allons faire une études sur un jeu de donnée et appliquer différents modèle de classifications afin de prédire au mieux les 7 classes. Pour réaliser cela, nous allons d'abords réaliser une études descriptive de notre jeu de données, discuter les résultats puis comparer les résultats des différents modèles. Pour réaliser ce travail, nous avons travailler avec le langage **Python** et utilisant principalement les bibliothèques sklearn,matplotlib et pandas,

2 Données

L'ensemble des données est composé de 581 012 instances, où chaque observation (instance) correspond à une cellule de 30 x 30 mètres. Pour chaque observation, 12 mesures (attributs) sont données. Ainsi, l'ensemble de données final est composé de 10 variables quantitatives, de 4 **wilderness areas** binaires et de 40 variables binaires de **soil type** , pour un total de 54 colonnes de données. Ci-dessous, quelques descriptions à propos de ces est donné :

Variable	**Description**
Elevation	Altitude en mètres
Aspect	Orientation
Slope	Pente en degrés
Horizontal_Distance_To_Hydrology	Distance horizontale par rapport aux éléments aquatiques de surface les plus proches
Vertical_Distance_To_Hydrology	Distance verticale par rapport aux éléments aquatiques de surface les plus proches.
Horizontal_Distance_To_Roadways	Distance horizontale de la route la plus proche
Hillshade_9am	Indice d'ombrage à 9h, solstice d'été (0 à 255)
Hillshade_Noon	Indice d'ombrage à midi, soltice d'été (0 à 255)
Hillshade_3pm	Indice d'ombrage à 15h, solstice d'été (0 à 255)
Horizontal_Distance_To_Fire_Points	Distance horizontale par rapport aux points d'inflammation d'incendie de forêt les plus proches
Hillshade_Noon	Indice d'ombrage à midi, soltice d'été (0 à 255)
Hillshade_3pm	Indice d'ombrage à 15h, solstice d'été (0 à 255)
Wilderness_Area	Désignation d'une réserve intégrale. 4 colonnes binaires, 0 (absence) ou 1 (présence)
Soil_Type	Désignation du type de sol. 40 colonnes binaires, 0 (absence) ou 1 (présence)
Cover_Type	Désignation du type de couverture forestière.C'est l'attribut à prédire (1 à 7)

Table 1: *Récapitulatif des variables)*

Afin de lire nos données, nous avons utilisé la librairie pandas afin de manipuler une `DataFrame`. Dans un premier temps, nous avons afficher les données pour voir directement leur représentation et avoir une première idée. Nous avons remarqué que les noms des variables ne figurait nul part, donc nous avons créé une liste avec le nom des variable et nous l'avons renommé le nom des colonnes pour avoir une meilleur visibilité et une manipulation plus simple des données.

Nous observons que environ **85%** de nos données appartiennent aux classes 1 ou 2 et que **0.5%** de données font partie de la classe 4,ces caractéristiques particulières de notre jeu données vont nous poser des problèmes quand nous allons commencer à implémenter des modèles de prédictions.

3 Analyse des données

L'analyse des données est une étape primordiale dans toute modélisation statistique.Elle nous permet de décrire et explorer les données avant d'en tirer de quelconques lois ou modèles prédicatifs de plus elle nous permet d'extraire tous les informations pertinentes contenu dans nos données.

En outre une analyse de données exhaustif nous permet de tirer énormément d'information, avoir une première vision sur notre jeu de données et nous donner des indication les modèles prédicatifs susceptible d'adapter nos caractéristiques. Il faut savoir aussi qu'une bonne analyse de données peut améliorer considérablement nos résultats.

Par conséquence pour faire face aux problèmes d'équilibrage vu précédemment nous allons utiliser différentes techniques d'analyse de données.

3.1 Explorations de données

Dans un premier temps, nous nous somme intéressé à la répartition des données entre les sept classes afin de voir si le jeu de données et bien équilibré, ou bien au contraire une ou plusieurs classes en une présence plus importante.

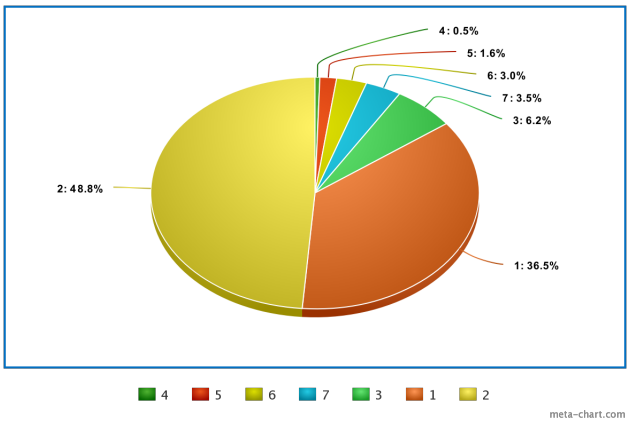

Figure 1: *Proportion des données pour chaque classe*

Sur le diagramme circulaire, on remarque que le jeu de données n'est pas équilibré. En effet, les deux première classe représente environ 85% des données, tant dis que la taille de la classe 4 est insignifiante par rapport à la taille du jeu de données. Cela nous pouce à réfléchir sur que qu'on cherche à prédire, et à nous poser des questions sur le faite de vouloir peu être essayer de prédire uniquement les 2 premières classe, ou peu être encore, enlever des données et équilibré le pourcentage de présence de chaque classe afin d'essayer de prédire toute les classes.

Dans un second temps, nous avons voulu visualiser toute notre jeu données, nous avons donc pensé à une représentation en 2D, nous avons alors décidé de réaliser une `analyse en composante principale` afin de trouver 2 axes de projection qui garde une représentation plus fidèle de nos données. Nous avons bien-sûr

pensé dès le départ que les 2 premières classe seront plus représenté vu qu'elles constituent 85% de nos données, mais la possibilité de voir l'apparition d'une géométrie particulière reste possible.

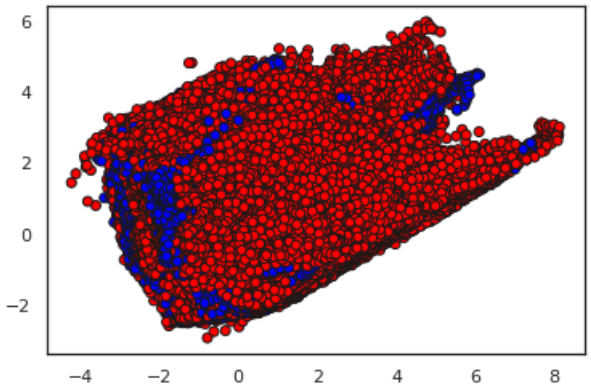

Figure 2: *Analyse en composante principale*

Comme nous l'avant pensé dès le départ, la projection en 2D, nous a pas apporté une information en particulier, nous n'arrivons pas à distinguer un une classe d'une autre. La combinaison linéaire des variables ne sépare pas les classes.

Comme énoncé précédemment, notre jeu de donnée se compose de 54 variables, 10 variables quantitatives et le reste c'est des variables qualitatives. Nous nous somme intéressé au début aux variables qualitatives, nous avons décider de visualiser les boxplots de chaque variable pour chaque classe pour voir afin de voir si on peut repérer un comportement moyen spécifique à une classe.

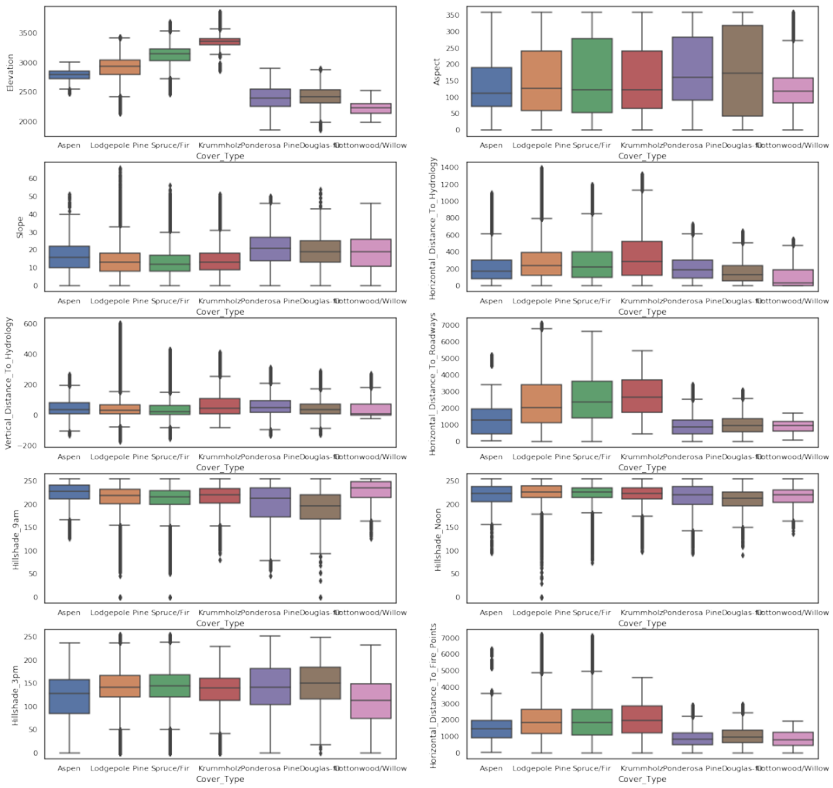

Figure 3: *Boxplot des différents features*

On remarque sur les boxplots que la valeurs moyens de chaque variable pour toute les classe sont en générale très proche sauf pour la variable **Elevation** où on remarque un comportement différent. Cela nous donne une indication sur le faite ce variable puisse être très significative dans le classification. Après ce résultat, il serait intéressant de comparer la densité des variables pour chaque classe.

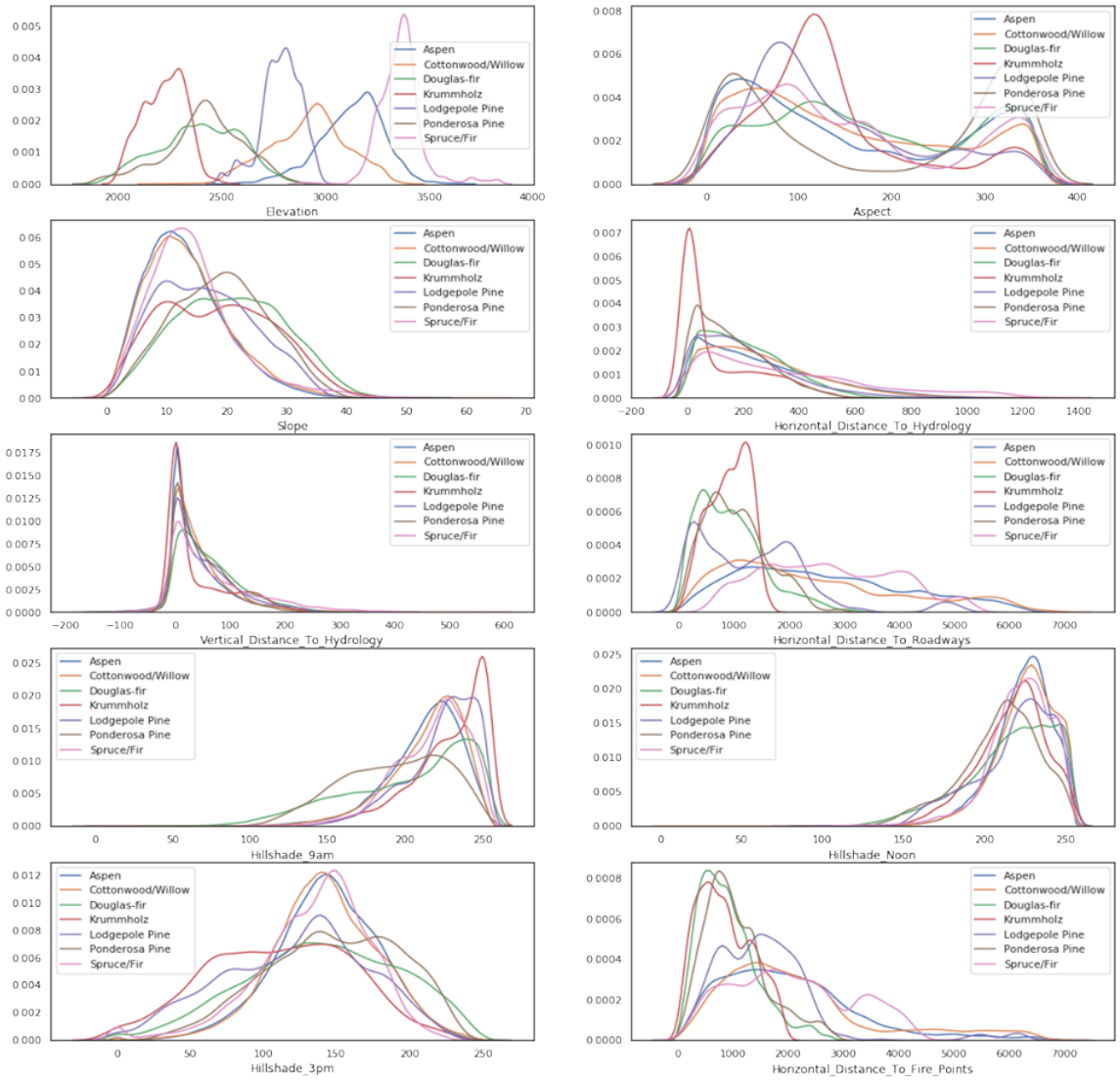

Figure 4: *Graphes de densité pour les différents features qualitatives*

On remarque sur les graphiques des distributions très similaire pour les classe pour la majorité des variables. Les densité pour les variables **Elevation** sont très différentes, cela montre plus clairement que cette variable peu avoir un rôle très important dans la classification.

Dans un la même démarche, sur les variables quantitatives, pour essayer de repérer un variable plus importante que les autres, nous visualisons la proportion d'apparition et d'absence d'un caractère pour chaque classe. On remarque que seul la variable **Wilderness_Area_3** à l'aire d'avoir des proportions équilibré pour les deux caractères dans chaque classe.

Figure 5: *Graphes pour l'apparition des différents features qualitatives*

Une matrice de corrélation est un tableau montrant les coefficients de corrélation entre les variables. Chaque cellule du tableau montre la corrélation entre deux variables. Une matrice de corrélation est utilisée comme moyen de résumer les données, comme entrée dans une analyse plus avancée et comme diagnostic pour les analyses avancées par exemple elle nous permettra plus tard de faire une sélection de variable plus pertinentes.

Dans la matrice une corrélation positive indique dans quelle mesure ces variables augmentent ou diminuent en parallèle ; une corrélation négative indique dans quelle mesure une variable augmente à mesure que l'autre diminue.Donc lors la sélection de variable nous allons opter de garder des variables peu corréler entre eux pour avoir les feautures le plus représentatives.

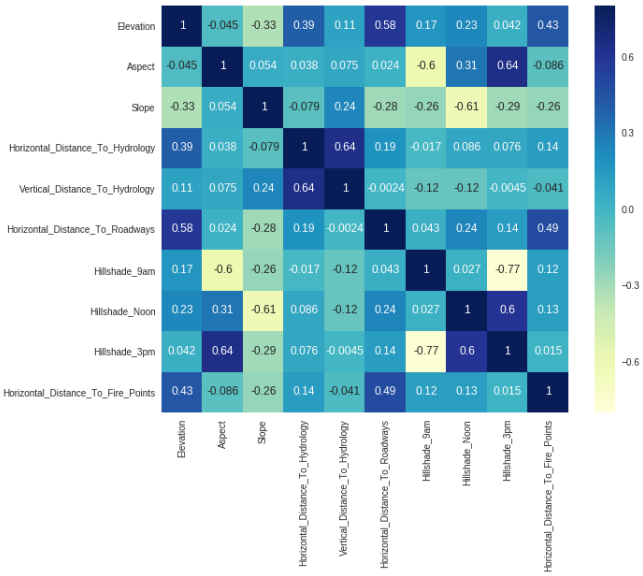

Figure 6: *Matrice de correlation (heatmap)*

3.2 Sélection des Variables

La sélection des Variables est l'un des concepts de base de l'apprentissage statistique qui a un impact considérable sur les performances de notre modèle. Les caractéristiques de données que nous utilisons pour former nos modèles d'apprentissage machine ont une influence énorme sur les performances que nous pouvons atteindre.Les caractéristiques non pertinentes ou partiellement pertinentes peuvent avoir un impact négatif sur les performances du modèle.

Dans cette partie, nous avons décidé d'appliquer deux méthode de sélection de variables, afin de de choisir les variables les plus importante susceptible de classifier au mieux notre jeu de données. Notre démarche était la suivante, nous avons décidé de choisir les dix variables les plus importantes avec deux modèles de sélection de variables, puis de comparaison les résultats des deux sélections, nous avons trouvé sept variables en commun, nous avons décider de les garder et travailler sur l'apprentissage des modèles avec ces dernières. Lors de l'analyse descriptive des données, nous avons repérer certaine variables susceptible d'être intéressante pour la classification notamment la variable **Elevation**, dans les deux modèles de sélection, cette variable est très significative. Pour réaliser cette sélection, nous avons utiliser la librairie sklearn, et exactement les modèles SelectKBest avec un test chi2 et le modèle ExtraTreesClassifier. Nous avons garder les variables suivante :

- `Elevation , Aspect, Slope`
- `Horizontal_Distance_To_Hydrology ,Vertical_Distance_To_Hydrology', Hillshade_3pm,`
- `Wilderness_Area_3`

Le faite de faire une sélection de variables, était pour nous manière de mettre en évidence les caractéristique les plus importante et se passer des informations sans valeurs, donc avoir moins de données à utiliser pour l'entraînement et gagner du temps.

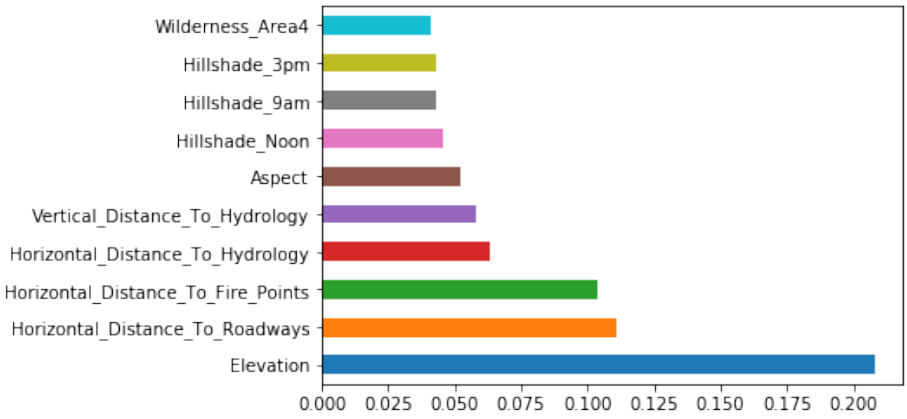

Figure 7: *10 Meilleures features selon ExtraTreesClassifier*

3.3 Équilibrage de données

Comme vu précédemment dans la figure Figure 1 nos donnée sont très déséquilibré, en fait nous pouvons entraîner un classifier que à reconnaître les données de la classe 1 et 2 et nous aurons un taux de réussite d'environ 80%[1].

Autrement dit nous avons besoin d'entraîner nos modèles de prédictions sur de données plus équilibrés pour qu'ils puissent marcher sur tous types de données.Donc pour faire ceci, dans un premier temps nous détectons la classe avec le moins d'éléments (classe 4 avec 0.5%) et ensuite nous construisons nos bases de train à partir de la taille de cette classe.Nous prenons 80% des éléments de la classe 4 et nous le mettons dans notre train. Ensuite nous récupérons un échantillon de tailles comparables des autres classes et nous les mettons aussi dans le train. Si on considère que la classe 4 avait 100 éléments nous prenons 80 de ces éléments pour le train et des par les autres classes nous prenons entre 150 et 250 éléments. Au final nous avons un jeu de données qui est pas terrible déséquilibré ,comme avant, mais qui il a toujours la classe 4 comme sa plus petite classe.

Avec ce train équilibré nous entraînons le modèle et ensuite nous testons la précision en utilisant comme test les données restantes.Nous avons donc dans le meilleure de cas un train qui représente 6% de la données globales[2].

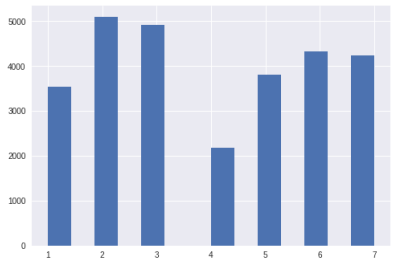

Figure 8: *Proportion des données de test équilibrés*

[1] Dans le fichier model.ipynb nous avons effectuer ce test et on moyenne nous avions 77% de précision

[2] Toute cette procédure d'équilibrage et classification est implémenté dans le fichier ForestCoverTypePrediction_Classification.ipynb

4 Classification

Dans cette partie, nous avons appliqué différentes méthodes de classification, pour cela nous avons utilisé la bibliothèque sklearn. Nous avons chercher à faire plusieurs prédictions en utilisant toutes les variables où uniquement les variables sélectionnées dans la section précédente. A chaque expérience, division notre jeu de données en des données d'entraînement et de test avec 10% ou 30% pour les données de test. Pour vérifier le résultat de la classification, nous calculons le **score**. Pour cette classification[3], nous avons utilisé les modèles suivant:

- Nearest Neighbors initialiser avec 7 points
- SVM
- Radial Basis Function (RBF) kernel SVM
- Logit
- Decision Tree
- Random Forest avec n_estimators=10,100,500
- Extra Tree Classifier n_estimators=100,500
- Neural Net alpha= 0.0001,max_iter=450
- AdaBoost
- Naive Bayes
- QDA

	Type of Classifier	Number of Features	Training Time	Accuracy
0	RBF SVM	7	389.025991	75.4%
1	Nearest Neighbors	7	0.058918	73.8%
2	Neural Net	7	80.309711	73.3%
3	Random Forest 100	7	7.678117	69.7%
4	Random Forest 500	7	37.874980	69.7%
5	Random Forest 7	7	0.722159	69.7%
6	Decision Tree	7	0.153864	69.2%
7	Decision Tree	7	0.152850	69.2%
8	Logit	7	14.510261	68.9%
9	Linear SVM	7	49.637981	68.7%
10	ExtraTreesClassifier 100	7	2.159880	68.2%
11	ExtraTreesClassifier 500	7	10.484713	67.9%
12	Naive Bayes	7	0.020671	60.5%
13	AdaBoost	7	2.903359	46.4%
14	QDA	7	0.031310	0.5%

Figure 9: *Table de Précision pour un train de 10%*

[3]Les résultats de cette partie se trouvent dans model.ipynb et ForestCoverTypePrediction_Classification_non_balanced.ipynb

Dans ce premier tableau(Figure 9) nous implémentons nos différents modelés sur un train de 10% avec que les 7 meilleures features ,identifier pendant la phase de sélection de variable.Tous les tableaux donnés sont triés en ordre décroissant de précision. Cette structure nous permet de voir que globalement ,indépendamment du nombre de feautures, le modèle le plus performant est la SVM RBF. Mais en même temps elle demande un temps considérable d'entraînement pour finalement pas de grand choses par rapport aux autres modèles. Contrairement le KNN qui est la 2_{me} méthode la plus précise elle demande un temps d'entraînement très faible pour un résultat presque équivalent à celui de RBF.En troisième place nous avons un MLP ,un réseau de neurones classique que en revanche demande aussi un temps considérable d'entraînement.Après nous avons les arbres,le logit et la SVM classique qui nous donnent des précisions autour 60%-70% Enfin nous avons QDA,AdaBoost et Naive Bayes qui ne sont pas du tout adapté au problème car ils donnent des précisions souvent très faibles.En gardant les mêmes nombres de caracteristiques et en augmentant à 30% le training set nous pouvons augmenter encore plus les accuracy des différents classifiers.

	Type of Classifier	Number of Features	Training Time	Accuracy
0	RBF SVM FeatImpo	20	3131.476452	84.76%
1	RBF SVM Chi2Sel	20	2773.267598	84.02%
2	Nearest Neighbors FeatImpo	20	0.221105	82.86%
3	Nearest Neighbors Chi2Sel	20	0.106862	82.12%
4	Neural Net FeatImpo	20	140.358320	81.11%
5	Neural Net Chi2Sel	20	381.993305	80.21%
6	Logit FeatImpo	20	49.083575	71.85%
7	Linear SVM FeatImpo	20	174.403965	71.78%
8	Logit Chi2Sel	20	26.734608	71.43%
9	Linear SVM Chi2Sel	20	93.219044	71.27%
10	Random Forest 7 FeatImpo	20	1.125619	71.17%
11	Random Forest 100 FeatImpo	20	10.698176	71.06%
12	Random Forest 500 FeatImpo	20	52.333740	71.01%
13	Random Forest 500 Chi2Sel	20	104.479617	70.83%
14	Random Forest 100 Chi2Sel	20	21.475590	70.74%
15	Random Forest 7 Chi2Sel	20	2.490886	70.74%
16	Decision Tree Chi2Sel	20	0.981607	70.06%
17	Decision Tree FeatImpo	20	0.296024	69.89%
18	ExtraTreesClassifier 500 FeatImpo	20	15.913786	69.36%
19	ExtraTreesClassifier 100 Chi2Sel	20	3.922214	69.19%
20	ExtraTreesClassifier 500 Chi2Sel	20	20.109502	68.89%
21	ExtraTreesClassifier 100 FeatImpo	20	3.179654	68.6%
22	AdaBoost Chi2Sel	20	5.545707	63.42%
23	AdaBoost FeatImpo	20	4.094217	59.13%
24	QDA Chi2Sel	20	0.161014	36.46%
25	Naive Bayes FeatImpo	20	0.035007	15.77%
26	Naive Bayes Chi2Sel	20	0.060927	11.08%
27	QDA FeatImpo	20	0.081880	6.16%

Figure 10: *Table de precision pour un train de 10%*

Maintenant que nous avons classifiés avec 7 features parmi 53 possibles on peut en ajouter un peu plus pour voir comment la précision va évoluer.Pour le choix de feautures nous avons utilisé les 2 méthodes vu précédemment(Chi2 et feauture_importance_ de ExtraTreesClassifier).En passant de 7 à 20 features le temps de calcule se multiplient par 10 et on gagne que quelques points de précisions.Dans le meilleure de cas notre modèle gagne 9% en précision chose qui n'est pas un échange acceptable surtout pour le coût en temps

et mémoire payé.Finalement nous pouvons conclure que 7 à 10 features c'est assez acceptable pour faire une classification correcte quite à augmenter un peu la base de train.

4.1 Classification avec de données équilibrés

Dans la classification avec une base de train équilibré, comme nous avons que 6% de données globales 7 ou 10 ou même 15 features ne suffisent pas pour dépasser le seuil de 55% de précision. C'est pour cela que nous prenons 20 et 25 features. Mais même avec cela nous arrivons à des précisions beaucoup très faible qui pourtant sont calculés en moins de temps que avant.Nous ne continuions pas à ajouter des features parce que cela pourrait nous amener à du overfitting (sur-apprentissage).Pour combler à ce problème de précisions il faut jouer sur le différents paramètres de chaque modèles et faire un Feature engineering beaucoup plus exhaustifs.

	Type of Classifier	Number of Features	Training Time	Accuracy
0	RBF SVM Chi2Sel	25	176.540705	59.8%
1	RBF SVM FeatImpo	25	215.868554	59.0%
2	RBF SVM Chi2Sel	20	147.593845	58.0%
3	RBF SVM FeatImpo	20	177.857041	57.8%
4	Neural Net FeatImpo	25	204.504870	57.0%

Figure 11: *5 meilleures modèles sur les données équilibre*

	Type of Classifier	Number of Features	Training Time	Accuracy
0	RBF SVM Chi2Sel	20	147.593845	58.0%
1	RBF SVM FeatImpo	20	177.857041	57.8%
2	Neural Net FeatImpo	20	201.246059	56.5%
3	Neural Net Chi2Sel	20	205.082673	54.0%
4	Nearest Neighbors FeatImpo	20	0.057774	53.4%
5	Nearest Neighbors Chi2Sel	20	0.058662	53.2%
6	Random Forest 100 FeatImpo	20	6.007339	44.0%
7	ExtraTreesClassifier 500 Chi2Sel	20	8.693676	43.9%
8	Random Forest 7 Chi2Sel	20	0.623534	43.8%
9	ExtraTreesClassifier 100 Chi2Sel	20	1.785344	43.5%
10	Random Forest 100 Chi2Sel	20	5.505603	43.4%
11	Random Forest 500 Chi2Sel	20	27.763447	42.7%
12	ExtraTreesClassifier 100 FeatImpo	20	1.882931	42.2%
13	ExtraTreesClassifier 500 FeatImpo	20	8.963648	41.8%
14	Random Forest 500 FeatImpo	20	29.968629	41.7%
15	Random Forest 7 FeatImpo	20	0.719914	40.7%
16	Decision Tree Chi2Sel	20	0.158112	40.4%
17	Linear SVM Chi2Sel	20	24.549706	39.9%
18	Logit FeatImpo	20	15.911756	39.9%
19	Decision Tree FeatImpo	20	0.171524	39.8%
20	Linear SVM FeatImpo	20	24.614858	39.2%
21	Logit Chi2Sel	20	18.782337	38.0%
22	QDA Chi2Sel	20	0.108034	37.7%
23	AdaBoost FeatImpo	20	2.598887	29.9%
24	AdaBoost Chi2Sel	20	2.561570	20.0%
25	Naive Bayes FeatImpo	20	0.023196	16.1%
26	Naive Bayes Chi2Sel	20	0.024368	6.8%
27	QDA FeatImpo	20	0.071297	5.6%

Figure 12: *Table de précision pour les données équilibré*

	Type of Classifier	Number of Features	Training Time	Accuracy
0	RBF SVM Chi2Sel	25	176.540705	59.8%
1	RBF SVM FeatImpo	25	215.868554	59.0%
2	Neural Net FeatImpo	25	204.504870	57.0%
3	Nearest Neighbors FeatImpo	25	0.073480	56.4%
4	Nearest Neighbors Chi2Sel	25	0.078339	55.8%
5	Neural Net Chi2Sel	25	200.036752	55.8%
6	Random Forest 500 Chi2Sel	25	23.778029	45.5%
7	Random Forest 100 Chi2Sel	25	4.802805	45.4%
8	Random Forest 100 FeatImpo	25	5.208535	44.9%
9	Random Forest 500 FeatImpo	25	25.756289	44.8%
10	ExtraTreesClassifier 100 FeatImpo	25	1.903720	43.8%
11	ExtraTreesClassifier 100 Chi2Sel	25	1.884171	43.5%
12	ExtraTreesClassifier 500 Chi2Sel	25	8.987914	43.3%
13	ExtraTreesClassifier 500 FeatImpo	25	8.745472	42.8%
14	Random Forest 7 Chi2Sel	25	0.623760	41.5%
15	Linear SVM Chi2Sel	25	59.215138	41.0%
16	Logit Chi2Sel	25	19.878175	40.9%
17	Random Forest 7 FeatImpo	25	0.620116	40.8%
18	Logit FeatImpo	25	19.961309	40.6%
19	Decision Tree Chi2Sel	25	0.178446	40.3%
20	Linear SVM FeatImpo	25	28.580697	40.3%
21	Decision Tree FeatImpo	25	0.191273	39.6%
22	AdaBoost FeatImpo	25	2.683703	29.9%
23	AdaBoost Chi2Sel	25	2.691562	20.0%
24	Naive Bayes Chi2Sel	25	0.039472	17.4%
25	Naive Bayes FeatImpo	25	0.028365	17.3%
26	QDA Chi2Sel	25	0.070471	5.6%
27	QDA FeatImpo	25	0.094743	5.6%

Figure 13: *Caption*

5 Conclusion

Pour le Forest Cover Type les modèles basés sur les SVM RBF,KNN et MLP avaient une bonne précision, atteignant 80% sur des prédictions de la base de test.Mais la performance de RBF est toujours un problème à revoir si possible sinon KKN est la solution efficace. Pour ce qui concerne les classifiers type arbre Il est possible que, comme les données sont descriptives et présentent une certaine relation entre les attributs, les modèles arborescents aient donné de bons résultats. Les KNN et les Arbes sont efficaces en calcul et en utilisation de la mémoire pendant l'entraînement et la prédiction. Les modèles arborescents fonctionnent environ 3 à 10 fois plus vite que le réseau neuronal avec plusieurs couches cachées. Pour les travaux futurs, plus de teste tests devront être effectuer pour trouver des paramètres bien adaptés aux modèles predictifs qui s'entraîne sur de données équilibrés.