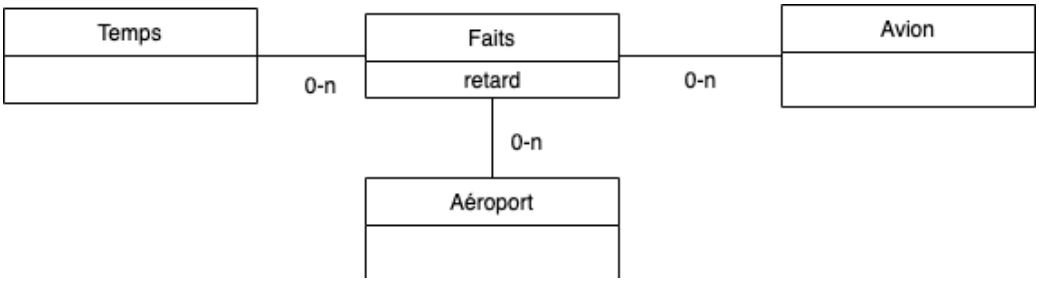
Rapport – Projet NF26

Modélisation

L'objectif de ce projet est d'être en mesure de répondre, à partir de données fournies, à des questions portant sur les retards des vol commerciaux. L'importante volumétrique des données à traiter pousse à la réflexion et à la mise en place de structures et de stratégies propres aux entrepôts de données. Trois fichiers CSV contenant chacun des données sont mis à disposition :

- {*year*}.csv : fichier contenant tous les vols caractérisés principalement par leur date et leur retard pour l'année *year* mais aussi par l'avion et le numéro de vol. (5M lignes)
- plane_data.csv : fichier contenant les informations techniques sur les avions répertoriés par leur immatriculation.
- airports.csv : fichier contenant les aéroports avec leurs différents alias et leurs locations.

Modéliser les données contenues dans ces fichiers n'est pas possible en suivant un modèle relationnel du fait de l'importante volumétrique qui rend les potentielles jointures trop coûteuses. Il est judicieux ici de choisir un schéma en étoile avec une table fait et plusieurs dimensions. Ce sont les calculs des retards qui intéressent ici, il est donc trivial que la table de fait consiste en les colonnes traitant du retard du fichier CSV principal. Les deux autres fichiers ainsi que les colonnes temporelles du premier fichier servent de dimension : data, avion, aéroport. Il faut cependant noter que dans le cadre de ce travail, puisque les questions traitées ne nécessitent pas des informations relatives aux aéroports, la dimension aéroport n'a pas été ajoutée afin de ne pas alourdir les données déjà conséquentes.



Pour construire cette modélisation en étoile, les données sont, dans un premier temps, extraites des fichiers CSV. Le fichier principal, la table de fait, étant très volumineux, les lignes lues ne sont pas stockées mais directement envoyé sous la forme d'un objet *Flight* dans un générateur. Chaque vol avant son traitement est complété par la dimension avion. Les données traitant des avions étant assez légères, il est possible de les stocker dans une structure itérable qui est ensuite parcourue afin de fournir via jointure à chaque vol l'avion correspondante. Le stockage plutôt que la lecture systématique est ici préférable étant donné le coût de lecture d'un tel fichier par rapport à sa taille. Pour la date, on se contente d'agréger les différentes composantes temporelles sous un *datetime*, il a d'ailleurs fallu faire un travail de conversion de format sur les heures de départ et d'arrivée, afin de simplifier les données.

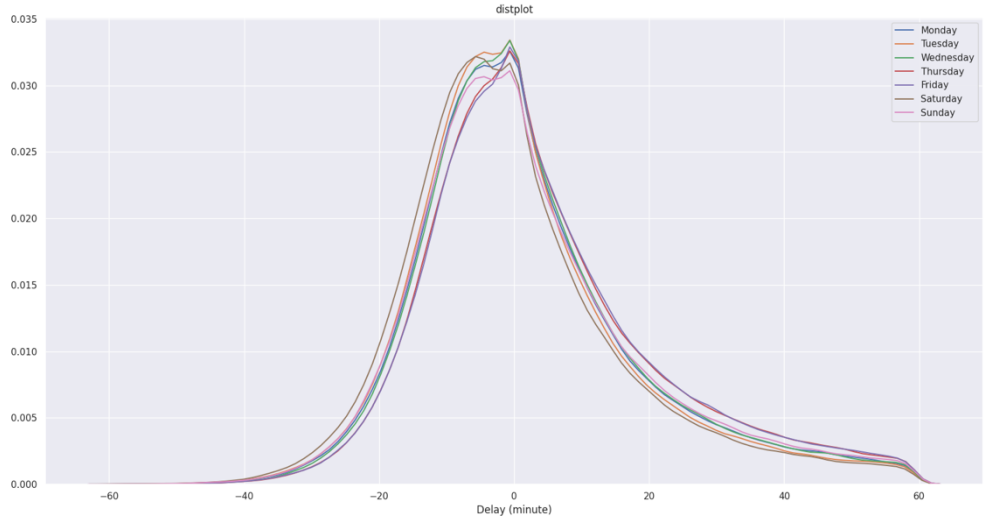
D’autres prétraitements moins importants mais réfléchis sont également effectués sur les données afin de les rendre plus propres et plus intègres sans toutefois y insérer un biais conséquent. En effet de nombreuses données possèdent des attributs non-renseignés prenant alors pour valeur ‘NA’, leur nombre important rendant impossible leur simple retrait du jeu de données. Différentes stratégies ont alors été pensés selon les cas. Dans le cas d’une composante clé et dont la valeur ne peut être déduite, la donnée est ignorée. Mais dans les autres cas, biens majoritaires, une valeur arbitraire est attribuée. Par exemple, il est logique de penser qu’un retard non-renseigné signifie qu’il n’y avait eu en réalité pas de retard (valeur de 0). De même une heure de départ ou d’arrivée réelle non-renseignée prend pour valeur l’heure de départ ou d’arrivée prévue. L’annulation ou la déviation d’un vol est supposée fausse lorsqu’elle n’est pas renseignée. Toujours dans une optique de mise au propre des données, on convertit ces composantes binaires en composantes booléennes. Le générateur ainsi construit est complet et volumineux et peut servir de bases solides pour toute question traitant du retard des avions.

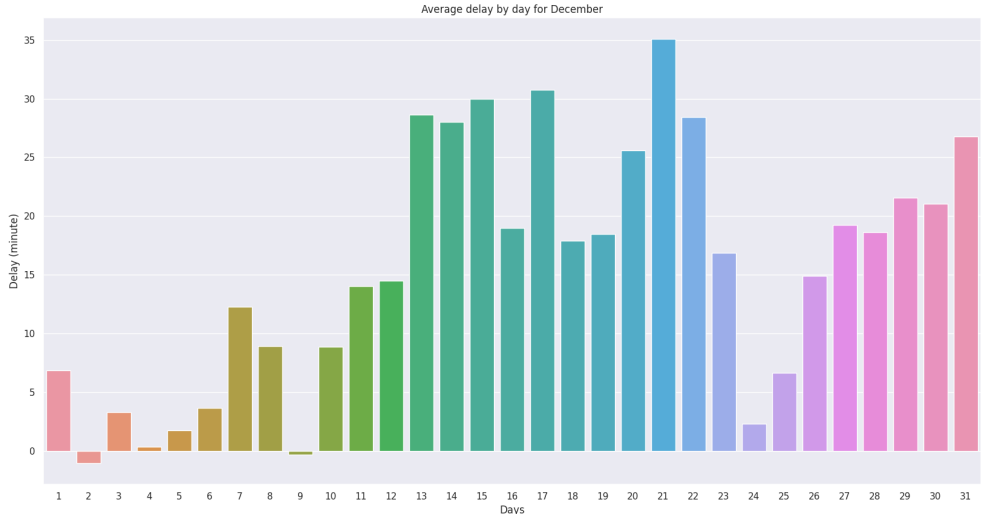
Meilleur moment/période pour minimiser le retard

Pour cette question, une implémentation du stockage des données sous forme de base orientés colonnes avec Cassandra a été choisi. En effet, Cassandra permet facilement le partitionnement des données, or au vu de la question, il peut être intéressant de partitionner temporellement les données. Il s’agit maintenant de déterminer les éléments de la hiérarchie de la dimension temporelle à utiliser pour cette clé de partitionnement composite. L’année est trivialement nécessaire dans le cas où la base de données contient les données sur plusieurs années (et donc en provenance de plusieurs fichiers CSV). De même, le mois doit également faire partie de la clé de partitionnement au vu du nombre de vols par mois (plus de 100K). La subtilité se pose alors sur le grain du partitionnement qui pourrait alors être entre autres l’heure, le jour du mois ou le jour de la semaine. Le jour du mois ne semble pas très judicieux puisque son numéro possède peu sémantique et est difficilement interprétable. Le partitionnement par heure peut lui révéler la répartition des retards sur une journée type mais peut alors ne pas être représentatif selon les jours creux ou les jours denses. C’est donc le partitionnement par jour de la semaine qui est retenu afin d’évaluer facilement le retard dans la semaine.

La clé de partitionnement déterminée, il faut maintenant définir la clé de tri. La clé de tri correcte doit permettre au sein même d’une partition de discriminer les données. En partant du principe qu’un avion ne peut décoller simultanément pour deux vols différents, la date de décollage sous forme de *timestamp* ainsi que l’immatriculation de l’avion apparaissent comme des choix justifiés pour une clé de tri composite. Pour des raisons d’optimisation de stockage, la table Cassandra répondant à la question est minimalisée et ne contient que les clés de partitionnement et de tri ainsi que la donnée à analyser : le retard. Ici, au vu des données sur le retard fournis, plusieurs manières était possibles pour définir le retard. En effet plusieurs retards différents étaient donnés : retard au décollage, retard à l’arrivée et le retard était également décomposé selon ses causes : météo, sécurité, compagnie aérienne, trafic aérien ou retard en cascade. Cependant ces derniers sont rarement renseignés et donc inutilisables. Le retard est alors ici défini comme le retard à l’arrivée, le retard au décollage étant forcément inclus dans le calcul du retard à l’arrivée.

Afin de déterminer les périodes et moment où le retard est minimal, il est intéressant de visualiser la répartition des données selon différents éléments de la hiérarchie de la dimension temporelle. Le jour de la semaine est un premier critère judicieux puisque le trafic aérien augmente ou diminue fortement selon les jours de la semaine. De plus, il est facile d'obtenir tous les trajets sur un an pour un jour de la semaine à partir de l'implémentation qui a été effectué pour la base Cassandra. Tracer successivement sur le même graphique l'histogramme des retards des différents jours de la semaine permet de comparer la distribution des retards pour chacun d'eux et de déduire le jour de la semaine où le retard est minimisé. Afin d'affiner cette recherche de période optimale, il est possible de descendre dans la hiérarchie temporelle et de travailler sur les heures des jours de la semaine en traçant un diagramme à barre des moyennes des retards par heure pour un jour de la semaine. En analysant ces graphiques, il est possible de déterminer les heures de la journée à favoriser pour un départ en avion pour chaque jour de la semaine. De manière plus générale, il est possible de s'intéresser à la moyenne des retards par mois afin de vérifier si certaines périodes de l'années sont plus ou moins sujettes aux retards. Dans le cas de retard moyen singulièrement plus faible ou plus élevé d'un mois par rapport aux autres, un approfondissement est possible en déterminant les moyennes des retards par jour du mois concerné afin de déterminer la cause de cette singularité qui peut être un pic ou une tendance sur le mois.

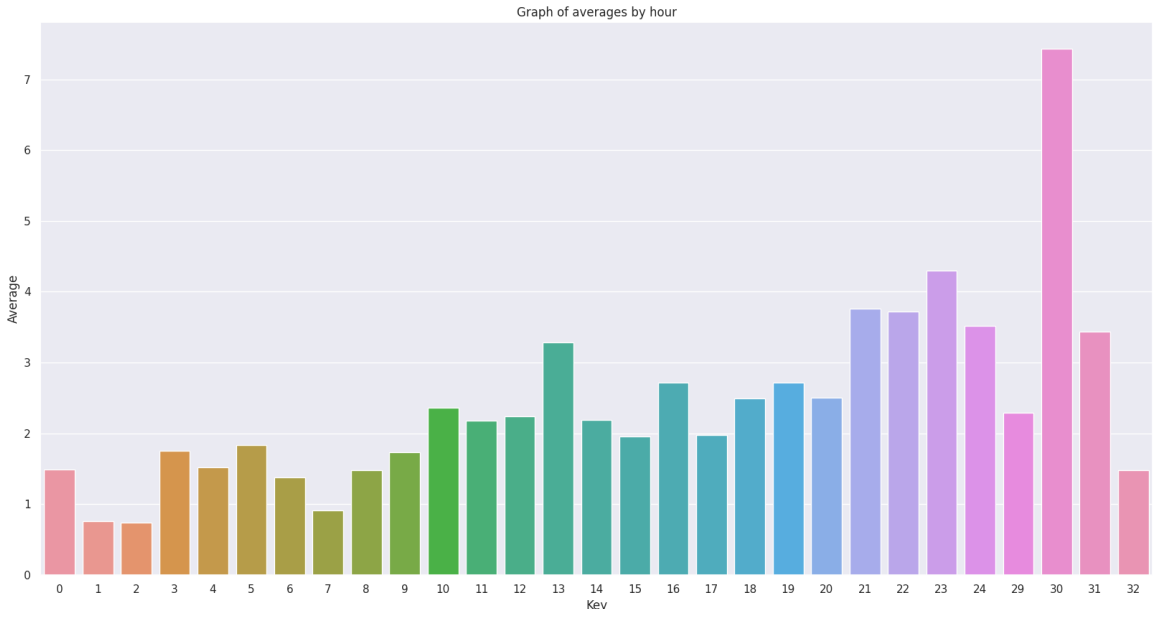




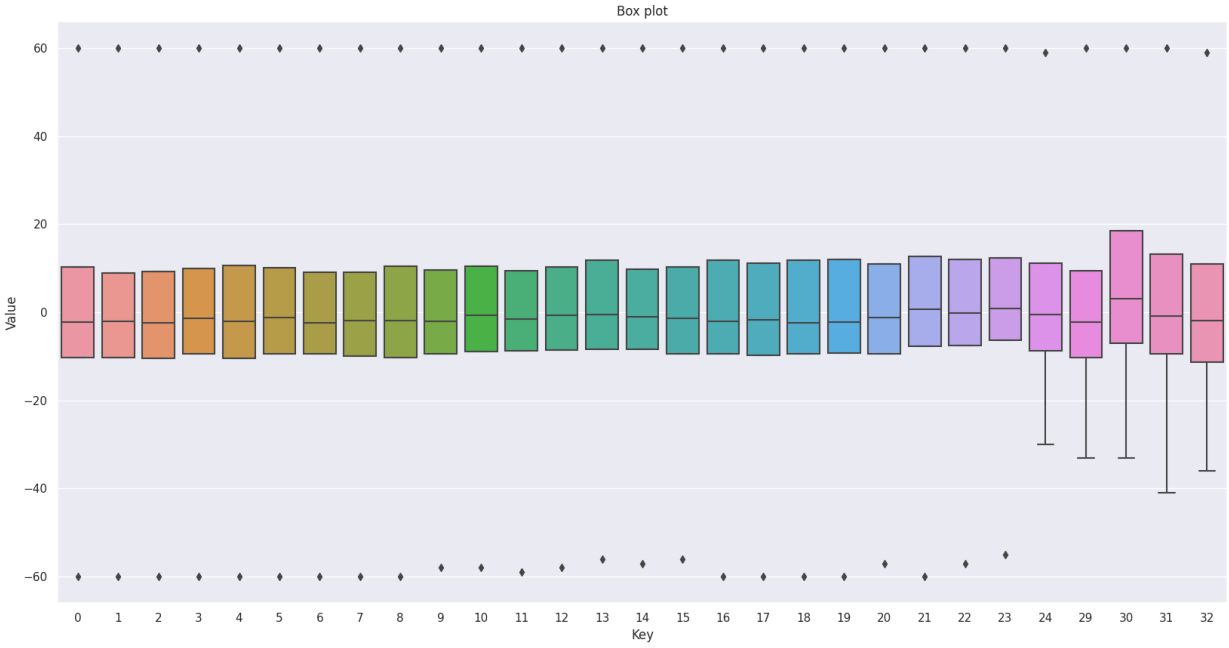
Âge de l’avion et retard

La seconde question tend à mettre en lumière la relation entre l’âge d’un avion et les retards qui lui sont propres. Il s’agit alors ici de comparer les distributions des retards en fonction de l’âge de l’avion. Dans cette optique, l’âge des avions, calculé par différence entre la date du vol et la date mise en service de l’avion, est classifié par année afin d’obtenir un nombre gérable de classes contenant une somme suffisante de données. Pour caractériser les données de chaque classe d’âge, les indicateurs statistiques courant, à savoir l’espérance, l’écart-type, l’asymétrie et le kurtosis, sont calculés. Le calcul de ces indicateurs impliquant la totalité des données classifiées justifie l’utilisation de Spark afin de paralléliser les mapping et réduction à effectuer. La première étape est la configuration de Spark afin de générer le RDD, RDD qui est fourni par un générateur intermédiaire ayant subi des opérations de filtrage et de projection des données. Ce RDD minimalisé peut entre-autre être filtré en amont sur le retard maximum autorisé, cela afin de retirer les valeurs aberrantes ou de ne sélectionner que des retards classiques. L’analyse des données via Spark est désormais possible, à noter que, volontairement, les fonctions d’analyse ont été définies très génériquement avec la nécessité de fournir des fonctions utilisateur de clé et de valeur afin de conserver la propriété de polyvalence propre aux entrepôts de données.

Le RDD prêt, il faut maintenant calculer les indicateurs statistiques par mapping et réduction pour chaque classe. Pour l’espérance et de l’écart-type, les calculs sont assez triviaux. Cependant pour l’asymétrie et le kurtosis, une étape intermédiaire de développement est nécessaire afin de calculer plusieurs petites sommes au lieu d’un unique mais complexe somme incalculable avec un seul map-reduce. Les indicateurs calculés pour chaque classe d’âge, il est alors facile de visualiser la caractérisation des données en traçant un diagramme à barre pour chacun de ces indicateurs en fonction de la classe d’âge. En effectuant un test de Pearson, le lien entre l’âge de l’avion et le retard est nié au vu de la p_value calculée : $9.30E{-}5$. De manière non-rigoureuse, il est possible de remarquer la non-normalité des données en comparant les kurtosis obtenu avec le kurtosis associé à une loi normale : 0.



Afin de caractériser les données, une seconde méthode est possible : les fractiles. Le but ici est de déterminer les fractiles de chaque classe d'âge, ce qui n'est pas si évident que cela sous contrainte de haute volumétrie. Les fractiles ne sont pas calculés mais approchés par interpolations successives. En effet, chaque interpolation propose des valeurs approximatives des fractiles recherchés. Il est alors possible de calculer la position réelle de cette valeur via la fonction de répartition élaboré avec un mapping de comparaison de la valeur proposée avec le reste des données et une réduction par somme. Le point ainsi déterminé est alors ajouté à une base de connaissances, cette base de connaissances augmentant en volume permet alors l'affinage des propositions issues des prochaines itérations. La base de connaissances est d'abord initialisée pour chaque classe avec les retards minimal et maximal et il suffit en général que de quelques itérations (moins d'une dizaine pour obtenir une approximation convenable). Deux méthodes d'interpolations sont possibles, linéaire et spline cubique monotone, et c'est cette seconde méthode qui est préférée dû au moindre nombre d'itérations nécessaires pour obtenir une approximation similaire. Il ne reste plus qu'à représenter ces fractiles au travers par exemples d'une boîte à moustache pour chaque classe d'âge. L'inexistence de lien entre l'âge de l'avion et les retards est alors confirmé au vu de la monotonie du retard en fonction de l'âge.



Note

Le code source utilisé au cours de ce projet est le fruit de mon travail personnel. Certaines fonctions, notamment pour la partie Spark, ont été reprises de mon travail au cours des TD précédents.