IFT 6758 Projet: Étape 1

Sortie: Sep 15, 2025
Date échéance: Oct 15, 2025

L'objectif de cette étape est de vous donner une expérience avec le data wrangling et l'analyse exploratoire des données. Ces deux phases d'un projet de science de données sont souvent là où vous passerez la plupart de votre temps dans plusieurs projets. Vous allez acquérir de l'expérience avec certains des outils courants utilisés pour récupérer et manipuler des données, ainsi que pour la création d'outils et de visualisations pour vous aider à comprendre les données avant de vous lancer dans une modélisation plus avancée.

De manière générale, le schéma de cette étape est d'utiliser l'API de statistiques de la LNH pour récupérer à la fois des données agrégées (statistiques des joueurs pour une saison donnée) et des données « play-by-play » pour une période de temps spécifique. Vous commencerez par créer des visualisations simples à partir des données agrégées qui ne nécessitent pas beaucoup de prétraitement. Ensuite, vous allez créer des visualisations interactives à partir des données play-by-play. Il y aura un petit nombre de questions qualitatives simples auxquelles répondre tout au long des tâches qui seront liées aux tâches décrites. Enfin, vous présenterez votre travail sous la forme d'une simple page web statique, créée à l'aide de Jekyll.

Notez que le travail que vous faites dans cette étape sera utile pour les futures étapes, alors assurez-vous que votre code est propre et réutilisable - vous en serez reconnaissant dans les prochains mois.

Une note sur le plagiat

L'utilisation de code/modèles à partir de ressources en ligne est acceptable et courante en science des données, mais soyez clair pour citer exactement d'où vous avez pris le code si nécessaire. Un simple extrait d'une ligne qui couvre une syntaxe simple à partir d'un article StackOverflow ou d'une documentation de package ne justifie probablement pas une citation, mais la copie d'une fonction qui effectue un tas de logique qui crée votre figure le fait. Nous espérons que vous pouvez utiliser votre meilleur jugement dans ces cas, mais si vous avez des doutes, vous pouvez toujours citer quelque chose pour être sûr. Nous effectuerons une détection de plagiat sur votre code et vos livrables, et il vaut mieux prévenir que guérir en citant vos références.

L'intégrité est une attente importante de ce projet de cours et tout cas suspect sera poursuivi conformément à la politique très stricte de l'Université de Montréal. Le texte complet des règlements à l'échelle de l'université peut être trouvé ici. Il est de *la responsabilité de l'équipe* de s'assurer que cela est suivi rigoureusement et des actions peuvent être prises sur des individus ou sur l'ensemble de l'équipe selon les cas.

Données de la LNH

Le sujet de ce projet est les données de hockey, en particulier l'API des statistiques de la LNH. Ces données sont très riches; elles contiennent des informations provenant de plusieurs années. Ces données incluent:

- Les métadonnées de la saison elle-même (par exemple, le nombre de matchs joués)
- Classements de la saison
- Statistiques des joueurs par saison
- Événement précis pour chaque match joué **(play-by-play data)**.

Si vous n'êtes pas familier avec les données play-by-play, la LNH utilise ces données exactes pour générer ses visualisations play-by-play, dont un exemple est présenté ci-dessous. Pour un seul match, environ 200 à 300 événements sont suivis, généralement limités aux mises au jeu, aux tirs, aux buts, aux arrêts et aux plaquages (pas les passes ni la localisation des joueurs). Notez qu'il existe une manière logique d'attribuer un identifiant unique aux matchs, qui est décrite ici (assurez-vous de porter attention à la différence entre les matchs de saison régulière et les matchs des playoffs!).

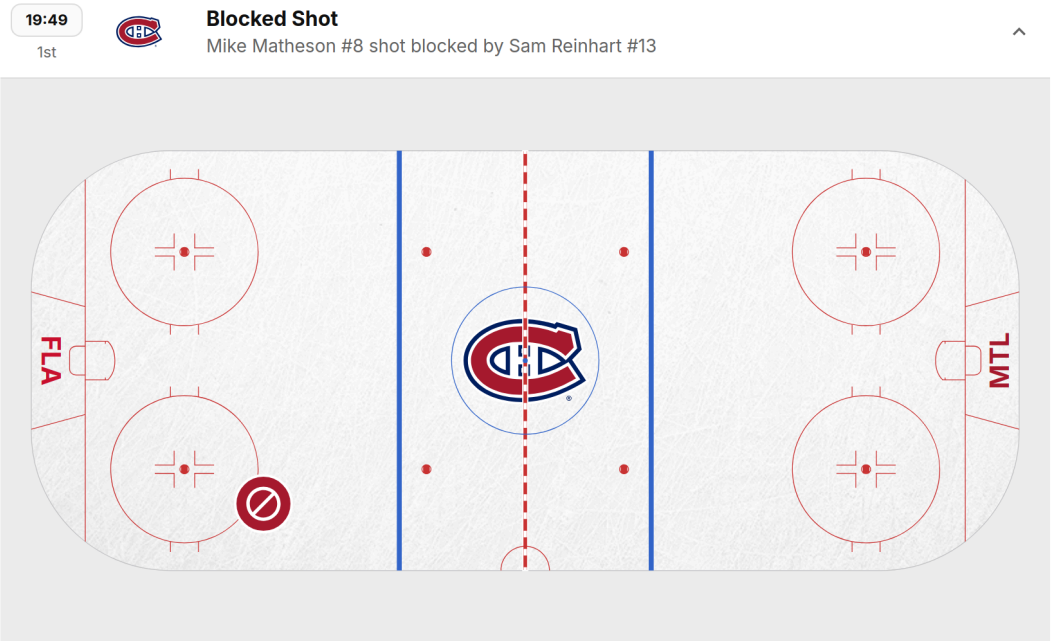


Figure 1 : Exemple de données play-by-play pour la partie **2023021190** *au début de la 1ère période. Chaque événement contient un identifiant (ex. « FACEOFF », « SHOT », etc.), une description de l'événement, les joueurs impliqués, ainsi que l'emplacement de cet événement sur la glace (dessiné sur la patinoire à droite). Les données d'événement brutes contiennent plus d'informations que ça. Vous pouvez explorer le play-by-play d'une partie ici.*

L'heure de l'événement, le type d'événement, l'emplacement, les joueurs impliqués et d'autres informations sont enregistrés pour chaque événement et les données brutes sont accessibles via l'API play-by-play. Par exemple, les données brutes pour le play-by-play ci-dessus peuvent être trouvées ici:

https://api-web.nhle.com/v1/gamecenter/{GAME_ID}/play-by-play

Par exemple. pour le jeu **2022030411** se trouve ici:

https://api-web.nhle.com/v1/gamecenter/**2022030411**/play-by-play

Un extrait des données brutes de l'événement se trouve dans la figure 2. Vous devrez explorer les données et lire les documents de l'API pour déterminer exactement ce dont vous aurez besoin.

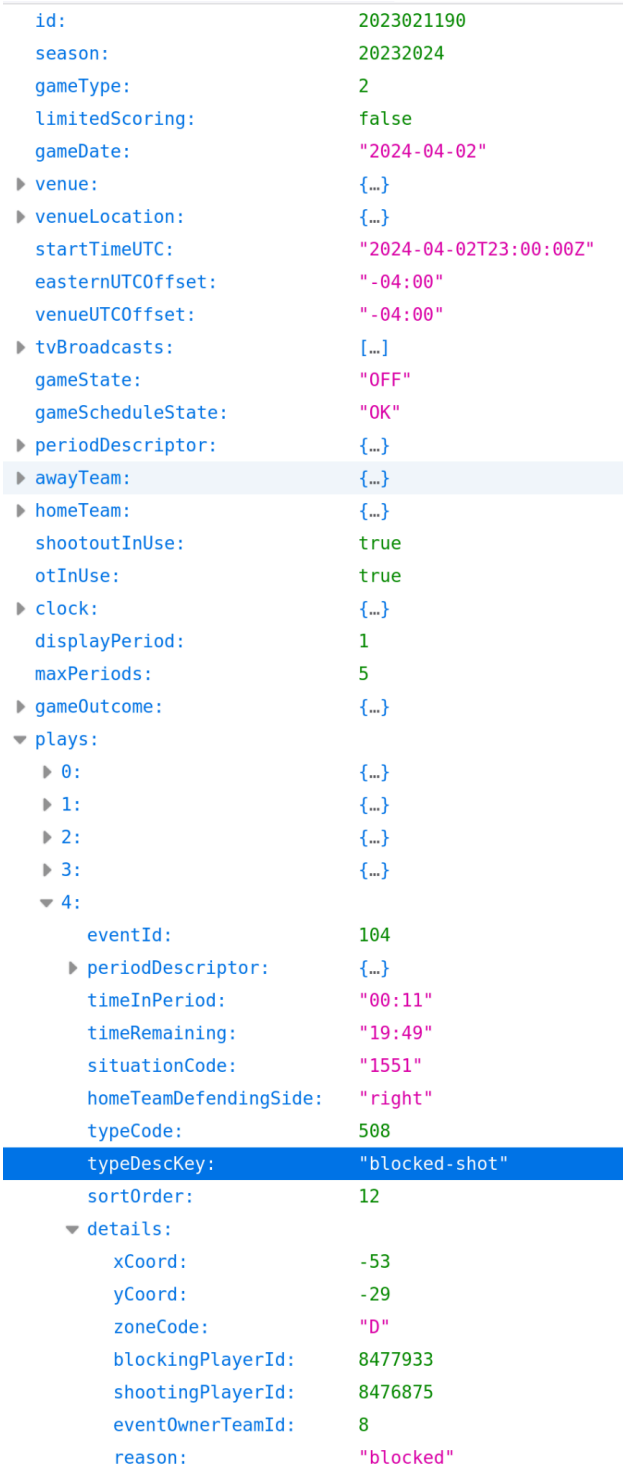


Figure 2 : Données JSON brutes obtenues à partir de l'API de statistiques de la LNH pour les mêmes événements de la figure 1. Notez qu'il existe d'autres événements autres que ceux souhaités - que vous pourrez explorer !

Quelques notes/avertissements sur l'API :

- L'API a été modifiée au cours de la saison 2023-24 ; par conséquent, document API non officiel détaillé est malheureusement en grande partie obsolète et ne peut pas être utilisé. De même, de nombreuses documentations communautaires (par exemple sur Reddit, les articles de blog) sont également obsolètes et ne fonctionneront pas.
- De nouveaux documents API communautaires sont disponibles, mais ils sont moins détaillés/complets que la documentation non officielle de l'ancienne API. Ils peuvent être trouvés ici :
  - Référence API NHL de Zmalski
  - Documentation API NHL de dword4
- Sur une note plus positive, l'interface semble être plus claire et plus détaillée (c'est-à-dire plus de notes sur les informations de jeu, plus de types d'événements tels que les tirs manqués/bloqués avec les coordonnées, etc.). Bien que comprendre l'API puisse être un peu plus difficile, l'utilisation des données devrait être beaucoup plus facile.

Motivation

Bien que nous comprenions que certaines personnes ne sont pas fans de sport, nous pensons qu'il s'agit d'un ensemble de données très riche et intéressant :

- Il s'agit d'un ensemble de données du monde réel utilisé par des data scientists professionnels, certains travaillant pour les équipes de la NHL elles-mêmes, tandis que d'autres gèrent leur propre entreprise d'analytique.
- Pendant la saison de hockey, les données sont mises à jour en temps réel lorsque les matchs sont en cours! Cela vous donne l'opportunité d'interagir fréquemment avec de nouvelles données, vous offrant un aperçu de l'importance d'un "pipeline" et de l'écriture d'un code propre et réutilisable dans un workflow réussi en science des données.
- Il contient énormément de données.
- L'API fonctionne généralement bien.
- Il est "désordonné" dans le sens où toutes les données brutes sont en JSON et ne sont pas immédiatement adaptées pour une utilisation dans un workflow de science des données. Vous devrez "ordonner" les données dans un format utilisable, ce qui constitue une part significative de nombreux projets en science des données. Étant donné que les données sont déjà fournies dans un format cohérent, nous pensons que c'est un bon équilibre entre vous donner du travail pour nettoyer les données, sans demander une quantité excessive de travail.
- Le hockey est souvent un excellent facilitateur de conversation ici au Canada (et surtout à Montréal). Si vous êtes nouveau au Canada, c'est une excellente façon d'en apprendre un peu sur la culture locale :)
- Même si vous n'êtes pas un fan de hockey, nous espérons que vous trouverez cette expérience de projet intéressante et éducative. Nous pensons que travailler avec des données du monde réel est plus gratifiant et beaucoup plus représentatif du workflow en

science des données que de travailler avec des ensembles de données préparés, tels que ceux disponibles sur Kaggle. Si vous êtes particulièrement fier de votre projet, certains des livrables vous apprendront à héberger votre contenu de manière publiquement accessible (via les pages Github), ce qui peut vous aider dans vos futures recherches de stage ou d'emploi!

Objectifs d'apprentissage

- **Acquisition et nettoyage des données**
  - Comprendre ce qu'est une API REST
  - Téléchargement programmatique de données depuis Internet en utilisant Python
  - Mettre en forme les données brutes en tables de données utiles
  - Se familiariser avec l'idée de "pipelining" de votre travail ; c'est-à-dire, créer des composants logiquement séparés tels que:
    - Télécharger et sauvegarder les données
    - Charger les données brutes
    - Traiter les données brutes dans un certain format
- **Exploration des données**
  - Explorer les données brutes et comprendre à quoi elles ressemblent
  - Construire des outils interactifs simples pour vous aider à travailler plus efficacement avec les données
- **Visualisation et science de données exploratoire**
  - Acquérir une certaine intuition et répondre à des questions simples sur les données en examinant des visualisations
  - Utiliser Matplotlib et Seaborn pour créer des figures agréables
  - Créer des figures interactives pour communiquer vos résultats de manière plus efficace

LIVRABLES

Dépôts de modèles sur lesquels travailler:

- Blog post template
- Project template

Vous devez soumettre:

1. Rapport en style d'article de blog
2. La base de code **reproductible** de votre équipe ; c'est-à-dire que toutes les figures peuvent être facilement régénérées.

Au lieu d'un rapport traditionnel rédigé en LaTeX, il vous sera demandé de soumettre un article de blog qui contiendra des points de discussion et des figures (interactives!). Nous fournirons un modèle et des instructions donc ne vous inquiétez pas de devoir tout comprendre par vous-même. À un niveau élevé, vous utiliserez Jekyll pour créer une page web statique à partir de Markdown. C'est une manière très simple de créer des pages au design agréable, et cela

pourrait vous être très utile à l'avenir si vous êtes intéressé par la rédaction de blogs, ou si vous souhaitez enrichir votre CV lors de recherches d'emploi. Bien que nous ne déploierons pas ces pages publiquement, il est très simple d'utiliser les pages Github pour publier votre contenu. Vous êtes plus que bienvenus pour le faire à la fin du cours!

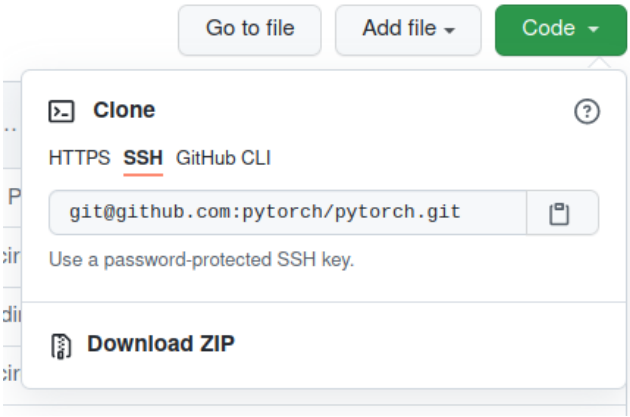
Détails de Présentation

Pour soumettre votre projet, vous devez:

- ☐ Publier votre soumission étape finale à la branche **`master`** ou **`main`**
  (Vous devez le faire avant de télécharger les ZIPs!)
- ☐ Soumettez un fichier ZIP de votre blog à gradescope
- ☐ Soumettez un fichier ZIP de votre base de code à gradescope
- ☐ Ajoutez le compte IFT6758 TA Github (**`@ift-6758-a25`**) à votre repo en tant que *viewer*

Note: Une seule personne par équipe doit faire la soumission sur gradescope!

Pour soumettre une archive ZIP de votre dépôt, vous pouvez la télécharger via l'interface utilisateur de Github.



N'oubliez pas que cette méthode ne télécharge pas l'intégralité de git, mais uniquement la branche principale ou principale. Assurez-vous que tout votre code se trouve dans la branche master avant de télécharger le ZIP.

Tâches et questions

Les tâches nécessaires pour la première étape sont décrites ici. Une description générale de ce qui est attendu est fournie au début de chaque tâche. La section "Questions" de chaque tâche détaillera le contenu requis pour l'article de blog correspond. On vous demandera soit des questions d'interprétation, soit vous devrez produire des figures ou des images à inclure dans l'article de blog. Nous essayons d'écrire en gras la plupart des éléments qui sont requis dans le rapport, mais assurez-vous de répondre à tout ce qui vous est demandé dans chaque question. Nous ne nous attendons pas à de longues réponses pour les questions; dans la plupart des cas, quelques phrases suffiront.

0. Article de blog

Créez un article de blog à l'aide du modèle fourni, contenant toutes les figures, réponses et discussions requises mentionnées dans la section précédente. Vous n'aurez pas besoin de déployer l'article de blog pour qu'il soit accessible au public. Cependant, des instructions seront fournies si vous souhaitez mettre en avant votre projet impressionnant sur votre CV une fois le cours terminé!

Vous DEVEZ le soumettre dans un format d'article de blog !

Nous vous suggérons de commencer à créer un environnement de travail pour les articles de blog dès le début, car il sera beaucoup plus facile de répondre aux questions et d'ajouter des graphiques pendant que vous travaillez sur le projet, plutôt que d'essayer de tout configurer juste avant la date limite!

1. Acquisition de données (25 %)

Créez une fonction ou une classe pour télécharger les données play-by-play de la LNH pour la saison régulière et les séries éliminatoires. Le point d'accès principal qui nous intéresse est:

`https://statsapi.web.nhl.com/api/v1/game/[GAME_ID]/feed/live/`

Vous devrez lire le document de l'ANCIEN API non officiel pour comprendre comment le GAME_ID est formé. Vous pourriez ouvrir ce point d'accès dans votre navigateur pour examiner le JSON brut afin de l'explorer un peu (Firefox dispose d'un agréable visualiseur JSON intégré). **Notez que ce document est généralement obsolète car la LNH est passée à une nouvelle API au cours de la saison 2023-24, mais l'ID de jeu est toujours formé de la même manière !**

Utilisez votre outil pour télécharger les données de la saison 2016-17 jusqu'à la saison 2023-24. Vous pouvez implémenter cela comme vous le souhaitez, mais si vous avez besoin de conseils, voici quelques astuces :

1. Il s'agit d'une API publique, et à ce titre, vous devez être conscient que quelqu'un d'autre paie pour les requêtes. Vous devriez télécharger les données brutes et les sauvegarder localement, puis utiliser cette copie locale pour en extraire des ensembles de données propres/utilisables.
2. **Ne commettez pas les données (ou de gros blobs binaires) dans votre dépôt GitHub**. C'est une mauvaise pratique, git est pour le code, pas pour le stockage de fichiers. Notez que la suppression et le commit d'un fichier ne suppriment pas réellement le fichier; il vous faudra réécrire l'historique de git (qui peut être assez difficile à faire). Une bonne façon d'éviter accidentellement de commettre des fichiers est d'utiliser un fichier **.gitignore** et d'ajouter le suffixe de fichier que vous souhaitez (comme *.npy ou *.pkl).

3. Une façon de procéder pourrait être de définir une fonction qui accepte l'année cible et un chemin de fichier en argument, puis vérifie si un fichier correspondant à l'ensemble de données que vous allez télécharger existe à l'emplacement spécifié. Si c'est le cas, elle pourrait immédiatement ouvrir le fichier et renvoyer le contenu sauvegardé. Sinon, elle pourrait télécharger le contenu depuis l'API REST et le sauvegarder dans le fichier avant de renvoyer les données. Cela signifie que la première fois que vous exécutez cette fonction, elle téléchargera et mettra en cache les données localement, et la prochaine fois, elle chargera les données locales. Considérez l'utilisation de variables d'environnement pour permettre à chaque membre de l'équipe de spécifier des emplacements différents.

4. Si vous voulez être encore plus sophistiqué, vous pourriez envisager d'intégrer cette logique dans une classe. Ceci se prête bien à la façon dont les données sont séparées par saisons de hockey, et vous permettrait d'ajouter une logique qui se généraliserait à toute autre saison que vous pourriez souhaiter analyser de manière propre et évolutive. Pour être encore plus sophistiqué, vous pourriez envisager de surcharger l'opérateur "(`__add__`)" sur cette classe pour vous permettre d'ajouter les données entre les saisons à une structure de données commune, vous permettant d'agréger les données sur plusieurs saisons. Ce n'est absolument pas obligatoire, ce ne sont que quelques idées pour vous inspirer! Vous êtes encouragés à être créatifs et à appliquer vos anciennes connaissances en structures de données/OOP à la science des données - cela peut vous faciliter la vie!

5. Écrire des docstrings pour vos fonctions est une bonne habitude à prendre.

Questions

1. **Rédigez un bref tutoriel** sur comment votre équipe a téléchargé l'ensemble de données. Imaginez que vous cherchiez un guide sur comment télécharger les données play-by-play; votre guide devrait vous faire dire "Parfait - c'est exactement ce que je cherchais!". **Incluez votre fonction/classe** et **fournissez un exemple de son utilisation**. Assurez-vous de ne pas simplement démontrer que votre fonctionnalité fonctionne. Il s'agit également d'un exercice de documentation et de communication de votre implémentation. Il n'est pas nécessaire que ce soit extrêmement compliqué, mais on attend quelque chose d'un peu plus cohérent et digeste que de simples captures d'écran de vos fonctions/code.

2. Outil de débogage interactif (10%)

Lorsqu'on travaille avec de nouvelles données, il est souvent utile de créer des outils interactifs simples pour vous aider à parcourir les données et à prototyper des implémentations. Un outil utile est ipywidgets, qui vous permet de créer très rapidement et facilement des widgets HTML dans une cellule de notebook Jupyter. Un cas d'utilisation courant pour ces widgets est de les appliquer en tant que décorateurs et de les utiliser pour spécifier les arguments de fonction. Par exemple, si vous souhaitez récupérer des informations qui résident dans un élément d'un

tableau, vous pouvez utiliser un *IntSlider* pour contrôler l'indice qui est passé dans cette fonction. Vous pouvez ensuite définir une logique dans cette fonction pour afficher votre image; si votre liste est une liste de chemins d'image, vous pouvez charger l'image et l'afficher via matplotlib. Ces widgets peuvent également être imbriqués, vous offrant un haut degré de flexibilité avec très peu d'effort.

Questions

1. **Implémentez un ipywidget** (ou un outil interactif de votre choix) qui vous permet de parcourir tous les événements, pour chaque match d'une saison donnée, avec la possibilité de changer entre la saison régulière et les séries éliminatoires. **Dessinez les coordonnées de l'événement sur l'image de la patinoire fournie**, similaire à l'exemple ci-dessous (vous pouvez simplement imprimer les données de l'événement lorsqu'il n'y a pas de coordonnées). Vous pouvez également imprimer toutes les informations que vous jugez utiles, telles que les métadonnées du jeu/boxscores et les résumés des événements (mais ce n'est pas obligatoire). **Prenez une capture d'écran de l'outil et ajoutez-la à l'article de blog**, accompagnée du **code de l'outil** et d'une **brève description (1-2 phrases)** de ce que fait votre outil. Vous n'avez pas à vous soucier de l'intégration de l'outil dans le blogpost.

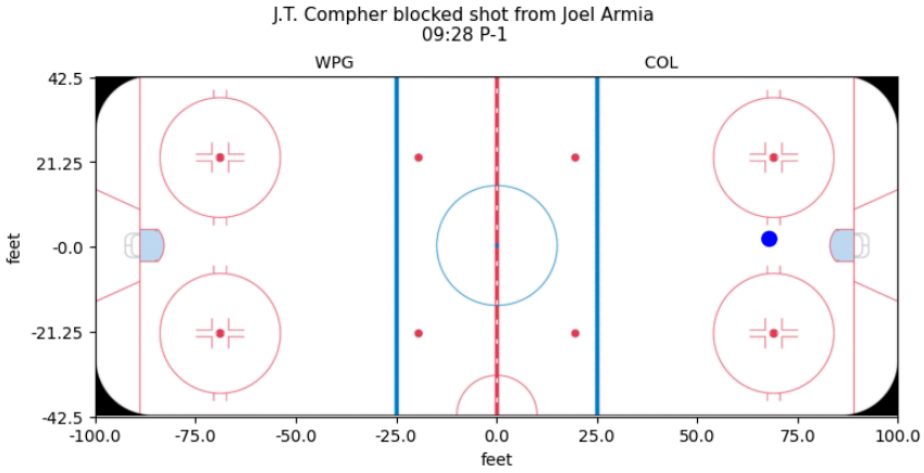
Remarque : *Un bon test de cohérence consiste à vérifier un jeu spécifique avec les données disponibles sur le site Web de la LNH, dont un exemple peut être trouvé ici. Vous remarquerez que les coordonnées de l'événement sont également dessinées, ce qui vous permet de confirmer si vos chiffres sont valides. Pour vous inspirer, une capture d'écran de celle que j'ai rapidement créée se trouve ci-dessous. À part la figure, vous n'avez pas besoin de copier cette mise en page; n'hésitez pas à ajouter toutes les informations que vous jugez utiles !*

game_id 379

```
2017-11-30T02:30:00Z
Game ID: 379; COL (home) vs WPG (away)

       OT
                    Home    Away
       Teams:       COL     WPG
       Goals:       3       2
       SoG:         26      33
       SO Goals:    None    None
       SO Attempts: None    None
```

i 46




```
{'about': {'dateTime': '2017-11-30T02:52:22Z',
           'eventId': 77,
           'eventIdx': 46,
           'goals': {'away': 1, 'home': 0},
           'ordinalNum': '1st',
           'period': 1,
           'periodTime': '09:28',
           'periodTimeRemaining': '10:32',
           'periodType': 'REGULAR'},
 'coordinates': {'x': 68.0, 'y': 2.0},
 'players': [{'player': {'fullName': 'J.T. Compher',
                         'id': 8477456,
                         'link': '/api/v1/people/8477456'},
              'playerType': 'Blocker'},
             {'player': {'fullName': 'Joel Armia',
                         'id': 8476469,
                         'link': '/api/v1/people/8476469'},
              'playerType': 'Shooter'}],
 'result': {'description': 'J.T. Compher blocked shot from Joel Armia',
            'event': 'Blocked Shot',
            'eventCode': 'COL77',
            'eventTypeId': 'BLOCKED_SHOT'},
 'team': {'id': 21,
          'link': '/api/v1/teams/21',
          'name': 'Colorado Avalanche',
          'triCode': 'COL'}}
```


Un exemple de widget interactif que vous pouvez créer pour explorer les données. Celui-ci a été créé à l'aide de simples ipywidgets et de matplotlib. Optionnellement, pour obtenir le nom des joueurs, vous pouvez vous référer à https://github.com/Zmalski/NHL-API-Reference?tab=readme-ov-file#get-specific-player-info

3. Nettoyer les données (10%)

Maintenant que vous avez obtenu et exploré un peu les données, nous devons formater les données de manière à faciliter l'analyse (c'est-à-dire c'est le temps de ranger les données)! Nous souhaitons généralement travailler avec de beaux dataframes Pandas plutôt que des données

brutes, vous êtes donc ici chargé de traiter les données d'événement brutes de chaque jeu en dataframes qui seront utilisables pour les tâches suivantes.
Créez une fonction pour convertir tous les événements de chaque jeu en dataframe Pandas. Pour ce milestone, vous devrez inclure des événements de type **«tirs»** et **«buts»**. Vous pouvez ignorer les **tirs manqués** ou les **tirs bloqués** pour le moment. Pour chaque événement, vous devrez inclure comme information (au minimum):

- l'heure/la période de jeu
- l'identifiant du jeu
- les informations sur l'équipe (quelle équipe a tiré)
- s'il s'agit d'un tir ou d'un but
- les coordonnées sur la glace
- le nom du tireur et du gardien de but (ne vous inquiétez pas des assists pour l'instant)
- le type de tir
- si c'était sur un filet vide
- si un but était à force égale en désavantage numérique ou en avantage numérique.

Questions

1. Dans votre article de blog, **incluez un petit extrait** de votre dataframe final (par exemple, en utilisant `head(10)`). Vous pouvez simplement inclure une capture d'écran plutôt que de vous battre pour que les tableaux soient soigneusement formatés en HTML/markdown.

2. Imaginez que le champ de « force » (c.-à-d. égal, avantage numérique ou en désavantage numérique) n'existait que pour les buts, pas pour les tirs. **Discutez de la façon dont vous pourriez ajouter les informations sur la force réelle** (c'est-à-dire 5 contre 4, etc.) aux *tirs et aux buts*, compte tenu des autres types d'événements (autre que ces derniers) et des autres données disponibles. **Vous n'avez pas besoin d'implémenter cette fonctionnalité pour ce milestone.**

3. En quelques phrases, **discutez d'au moins 3** caractéristiques supplémentaires que vous pourriez envisager de créer à partir des données disponibles dans cet ensemble de données. Nous ne cherchons pas de réponses particulières, mais si vous avez besoin d'inspiration, un tir ou un but pourrait-il être classé comme un rebond/tir en contre-attaque (expliquez comment identifier ceux-ci)?

4. Visualisations simples (25 %)

Utilisons maintenant les données nettoyées pour créer des graphiques simples sur les données agrégées. Pour chacune des questions ci-dessous, vous devez faire un choix approprié de graphique pour montrer le lien demandé. Il existe généralement plusieurs façons correctes de le faire - si votre graphique démontre bien le lien entre les données, vous obtiendrez tous les points.

Questions

1. **Produisez un graphique comparant** les types de tirs de toutes les équipes dans une saison de votre choix (i.e. agrégez juste sur tous les tirs). Superposez le nombre de buts sur le nombre de tirs. Quel semble être le type de tir le plus dangereux? Le type de tir le plus courant? Pourquoi est-ce que vous avez choisi ce type de graphique? Ajoutez ce graphique et cette discussion à votre article de blog.

    a. Quelle est la relation entre la distance à laquelle un tir a été effectué et la chance qu'il s'agisse d'un but? **Produisez un graphique** pour chaque saison entre 2018-19 et 2020-21 pour répondre à cette question, et ajoutez-le à votre article de blog avec quelques phrases décrivant le graphique. Y a-t-il eu beaucoup de changements au cours des trois dernières saisons? Pourquoi est-ce que vous avez choisi ce type de graphique?
    b. Trouver la distance du tir nécessite la combinaison de plusieurs des données, c'est à vous de trouver une méthode qui fonctionne pour la majorité des parties (certaines ont des informations manquantes).
    c. Si vous notez quelques données aberrantes, ne vous inquiétez pas, on en reparlera dans le milestone 2.

2. Combinez les informations des sections précédentes pour **produire un graphique** qui montre le pourcentage de buts (# buts / # tirs) en fonction à la fois de la distance par rapport au filet et de la catégorie de types de tirs (vous pouvez choisir une seule saison de votre choix). **Discutez brièvement de vos conclusions.** Par exemple, quels sont les types de tirs les plus dangereux?

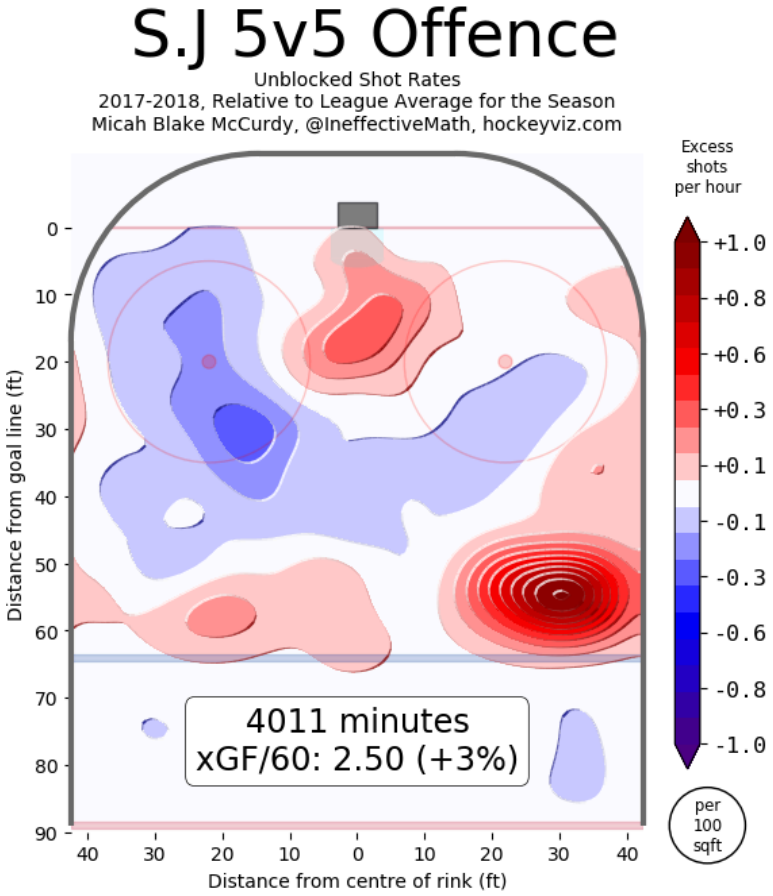
5. Visualisations avancées:(30 %)

L'ensemble final de visualisations que vous créerez sont des plans de tir pour une équipe de la LNH donnée, pour une année et une saison donnée. Un excellent exemple de ces graphiques, avec une description détaillée de comment les interpréter, se trouve sur le site Web de hockeyviz (qui est une excellente ressource pour ce qui est de l'application de la science des données au hockey). Notez que vous devrez créer ces graphiques à partir de zéro; pour ce milestone, vous ne pouvez pas utiliser de bibliothèque spécifique au hockey. Vous recevrez un exemple d'image de patinoire qui a le bon ratio.

Pour créer ces figures, vous devez :

1. Vous assurer que vous pouvez travailler correctement avec les coordonnées de l'événement. Cela inclut de s'assurer que les tirs sont du bon côté de la patinoire (en raison de changements de période ou de commencer de différents côtés pendant un match), ainsi que de pouvoir mapper des coordonnées physiques aux coordonnées pixels du graphique.
    a. Vous devriez être capable de déduire ces informations pour la majorité des matchs (néanmoins pour certains il y a des informations manquantes).
    b. Si vous notez quelques données aberrantes, ne vous inquiétez pas, on en reparlera dans le milestone 2.

2. Calculez les statistiques agrégées des emplacements de tir dans l'ensemble de la ligue pour calculer le *taux de tir moyen par heure de la ligue* par emplacement. Vous pouvez faire quelques hypothèses simplificatrices :
    a. Vous pouvez ignorer le fait d'essayer de déterminer si un événement s'est produit lors d'un avantage numérique ou d'un désavantage numérique.
    b. Vous pouvez supposer que chaque match dure 60 minutes.
3. Regroupez les tirs par équipe et utilisez les *moyennes par ligue du taux de tir moyen par heure* calculées ci-dessus pour calculer *la différence du taux de tir par heure* pour chaque équipe avec la moyenne. Vous pouvez choisir de représenter cela soit comme une différence brute de buts entre les équipes, soit comme un pourcentage.
4. Faites des choix appropriés pour regrouper vos données lors de leur affichage. Vous pouvez également envisager d'utiliser des techniques de lissage pour rendre vos plans de prise de vue plus lisibles. Une stratégie courante consiste à utiliser l'estimation de densité de noyau avec un noyau gaussien.
5. Rendez le graphique interactif, avec des options pour sélectionner l'**équipe**. La façon la plus simple de le faire est d'utiliser quelque chose comme *plotly* ou *bokeh*. Un bel exemple de ce que vous pouvez faire avec plotly peut être trouvé ici.
6. Produisez un graphique interactif pour chaque saison de 2016-17 à 2020-2021. (compris). Il vous suffit de créer les figures de la zone offensive ; vous n'avez rien à faire pour la zone défensive. Comme mentionné ci-dessus, vous pouvez également ignorer le fait d'essayer de déterminer si un événement s'est produit lors d'un avantage numérique ou d'un désavantage numérique.



Source : www.hockeyviz.com. Exemple de carte de tir offensif pour les Sharks de San Jose, sur la période 2017-2018 Vous n'avez pas besoin de calculer les xGF pour/60, ou le nombre de minutes dans la zone offensive.

Questions

1. Exportez les 4 graphiques de zone offensive au format HTML et **intégrez-les dans votre article de blog**. Votre graphique doit permettre aux utilisateurs de sélectionner n'importe quelle équipe pour la saison sélectionnée.
   ***Note**: Parce que vous pouvez trouver ces graphiques sur l'internet, répondre à ces questions sans produire ces graphiques* ne *vous rapportera pas de points !*

2. **Discutez** (en quelques phrases) de ce que vous pouvez interpréter à partir de ces graphiques.

3. Considérez l'Avalanche du Colorado; jetez un œil à leur carte de tir au cours de la saison 2016-17. **Discutez de** ce que vous pourriez dire sur l'équipe au cours de cette saison. Regardez maintenant la carte de tirs de l'Avalanche du Colorado pour la saison 2020-21 et **discutez de** ce que vous pouvez conclure de ces différences. Est-ce que ça a du sens? *Astuce : regardez le classement*.

4. Considérez les Sabres de Buffalo, une équipe qui a connu des difficultés ces dernières années, et comparez-les au Lightning de Tampa Bay, une équipe qui a remporté la coupe Stanley pour deux années consécutives. Regardez les plans de tir de ces deux équipes des saisons 2018-19, 2019-20 et 2020-21. **Discutez des** observations que vous pouvez faire. Y a-t-il quelque chose qui pourrait expliquer le succès du Lightning, ou les problèmes des Sabres ? Est-ce que ces images sont suffisantes pour tout comprendre les succès ou problèmes d'une équipe?

***Remarque**: le but de cet exercice est de vous familiariser avec l'utilisation des bibliothèques Python standard pour créer des visualisations. Vous ne pouvez pas utiliser d'outil qui crée pour vous des visualisations spécifiques à un domaine (par exemple, le hockey). Vous êtes libre d'utiliser des bibliothèques de base (matplotlib, seaborn, plotly, bokeh, etc.) pour générer ces graphiques.*

Références utiles

Contenu du cours

- IFT6758 Hockey Primer
- IFT6758 Blog post template
- IFT6758 Project template - ancienne version - sera bientôt officiellement publiée

Documentation API

- Zmalski's NHL API Reference - documentation incomplète sur la **nouvelle API**
- dword4's NHL API Doc - une autre référence incomplète pour la **nouvelle API**
- Unofficial (old) NHL API Doc - **ancienne API -** cela ne fonctionne plus, cependant certains formats d'éléments tels que l'ID de jeu sont toujours valides

Divers

- Cookiecutter Data Science - Un outil utile pour vous aider à modéliser vos dépôts git