ECOLE SUPERIEURE
D'INGENIEURS DE RENNES

Université de Rennes

Post Mortem

Ethan GAUTHIER - Sterenn LE HIR
Alexandre LECOMTE - Axel MARTIN
Kechiedou MEDA - Ivann VYSLANKO

31 mai 2024

Sommaire

1 Ce qui a fonctionné **2**

2 Ce qui n'a pas fonctionné **2**

3 Ce qu'on peut améliorer **3**

1 Ce qui a fonctionné

La collaboration au sein de l'équipe était bonne. Nous étions séparés en deux équipes de 2, un équipe Hardware composée d'Alexandre et Sterenn et une équipe Software composée d'Ethan et Kechiedou. Nous communiquions fréquemment de nos avancées respectives et travaillions côte à côte pour pouvoir rapidement mettre certains points au clair.

De plus pour le modèle d'IA visant à prédire l'énergie produite par votre chauffage, nous avons testé plusieurs algorithmes pour déterminer celui qui fournit les meilleurs résultats avec nos données. Au début, cela n'était pas évident de trouver le bon modèle. Parmi les modèles testés, le Random Forest a montré des performances supérieures avec un R^2 de 0.9037, indiquant une bonne capacité à prédire l'énergie produite.

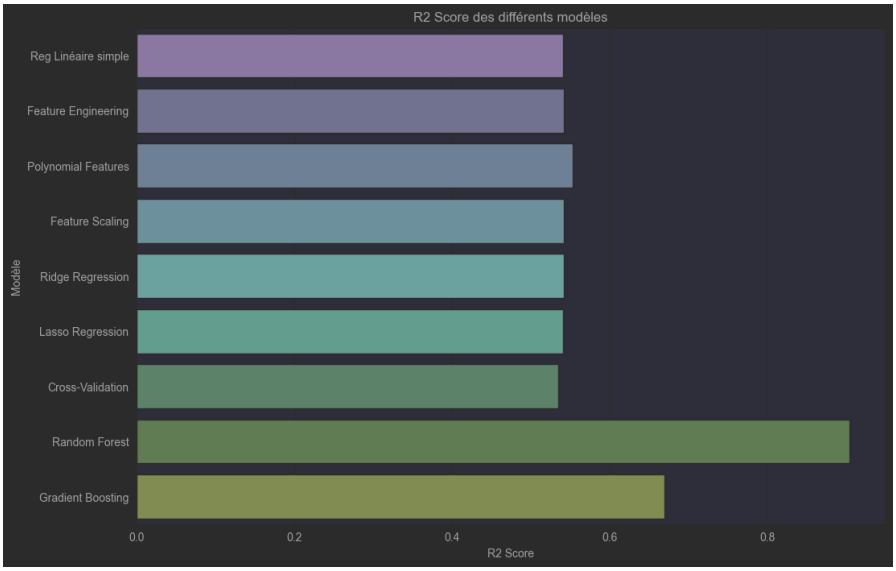

FIGURE 1 – évolution des scores des modèles de prediction

2 Ce qui n'a pas fonctionné

Notre défaut majeur tout au long de ce projet fut un manque d'anticipation, nous empêchant de finir nos tâches dans les temps. Par exemple, on a souvent pris du retard côté Hardware à essayer de faire marcher des capteurs incompatibles avec l'ESP32 (bien souvent à cause de sa tension trop basse de 3.3V) ou bien à bricoler une architecture qui ne tiens pas la route sur le plan électronique (même

alimentation pour tous les modules, manque de courant, etc).

De plus, nous avons fait le choix de ne pas intégrer notre solution dans son infrastructure finale, à savoir notre chauffage cannette car celui ci est trop grand et encombrant et travailler dessus nous aurait plus fait perdre du temps qu'autre chose. On a préféré se concentrer sur le développement d'un maximum de features, quitte à n'avoir qu'un proof of concept à échelle reduite.

Un autre pépin que nous avions plus ou moins prévu était notre incapacité à récupérer suffisamment de données du chauffage pour entrainer notre modèle de prédiction en vertu du temps qui nous était donné.

Enfin, encore une fois par manque de temps, nous n'avons pas testé le code dans l'application Flutter autrement que manuellement.

En plus des problèmes mentionnés, il n'a pas été possible de réaliser des tests complets de l'application mobile. La librairie Flutter utilisée pour les tests présentait un comportement différent de l'exécution normale du code. Par exemple, certaines fonctions affichaient des temps de chargement infinis lors de l'exécution avec la librairie, ce qui n'était pas le cas dans une exécution normale. Ce comportement erratique a empêché la validation complète de l'application mobile et la détection d'éventuels problèmes de fonctionnement.

3 Ce qu'on peut améliorer

Pour une prochaine expérience du même type on pourrait alors s'efforcer d'anticiper au maximum, surtout vis à vis du matériel requis (principal frein à notre projet) et à la faisabilité de nos idées de solution.

Aussi, nous aurions dû travailler un peu plus en environnement de production plutôt qu'en environnement de test pour faciliter l'intégration des différentes briques logicielles, c'est à dire : exclure assez rapidement les solutions qui fonctionnent en mode "local".

Un point crucial à améliorer pour les projets futurs est l'intégration des tests logiciels dès le début du processus de développement. Cela permettrait de détecter les erreurs et les problèmes de fonctionnement plus tôt, facilitant ainsi leur correction et la mise en place d'un code plus robuste et fiable.

L'implémentation des tests devrait se faire de manière itérative, en les inté-

grant au fur et à mesure de l'avancement du développement. Cela permettrait de s'assurer que chaque nouvelle fonctionnalité ou modification apportée au code ne compromet pas la stabilité et le bon fonctionnement du système.