PORTFOLIO

Donnelly Baart

STUDENTNUMMER: 15047628

Inhoudsopgave

Samenvatting en hyperlinks

In dit hoofdstuk beschrijf ik kort wat ik allemaal gedaan heb tijdens deze minor. Om te beginnen heb ik alle datacamp opdrachten afgerond.

Ik heb ook veel van de code geschreven. Ik heb de eerste opzetten geschreven voor het trainen, het berekenen van de kosten, het genetische algoritme en om resultaten te evalueren. Daarna heb ik de rest geholpen met het toevoegen van hun aanpassingen en met het schrijven van efficiënte numpy code. In het hoofdstuk Code ga ik hier dieper op in. In het hoofdstuk Genetisch algoritme beschrijf ik iets specifieker hoe ik het genetische algoritme in elkaar heb gezet en hoe ik aan de hyperparameters gekomen ben.

Ik heb een paar presentaties voorbereid en gegeven. Een introducerende open presentatie en een educatieve presentatie over genetische algoritmen. Verder heb ik samen met de rest van de groep gepresenteerd bij de intreerede van Sander Mertens. Voor de educatieve presentatie heb ik ook een Jupyter Notebook gemaakt. Daarnaast heb ik ook geholpen bij de voorbereidingen van de meeste andere presentaties. Ik heb met Michiel gepresenteerd tijdens de techathon. Deze is op YouTube te zien. Hierover vertel ik meer in het hoofdstuk Presentaties en paper.

Aan het einde van dit document is een evaluatie van wat ik dit project gedaan en geleerd heb.

Het onderzoek

Ons is door het lectoraat gevraagd of we naar hun probleem kunnen kijken met een data science blik. Danone wilt een groene energiecentrale voor één van hun fabrieken. Deze energiecentrale moet een constante hoeveelheid stroom leveren. Dit is lastig omdat wind- en zonne-energie eigenlijk nooit constant zijn. De fluctuaties moeten opgevangen worden met opslag.

Nu wilt het lectoraat graag weten hoe je aan de beste opstelling van zonnepanelen en windturbines komt. Dit is niet simpel te berekenen en alle combinaties uitproberen duurt natuurlijk te lang en is niet schaalbaar als in de toekomst meer van dit soort energiecentrales gebouwd worden.

Hoofdvraag en deelvragen

Onze hoofdvraag was “Wat is de optimale opstelling van een hybride solar/wind energy system voor een fabriek waarbij de kosten het laagst zijn?”. Hierbij hebben we de volgende deelvragen bedacht:

- Wat is er nodig om de aangeleverde simulatie in MATLAB aan te sturen via Python?
- Wat is het beste algoritme om de optimale hybride solar/wind energy configuratie te vinden?
- Welke hyperparameters zijn het beste voor het algoritme?
- Welke kosten zijn relevant voor het vinden van de optimale hybride solar/wind energy configuratie?
- Hoe werkt een hybride solar/wind energy systeem?

Ik heb me vooral bezig gehouden met de tweede en de derde deelvraag. De eerste deelvraag was best makkelijk. Er kwamen eigenlijk maar twee opties in me op: Een genetisch algoritme en reinforcement learning. Reinforcement learning is een ai techniek waarbij je een neuraal netwerk traint om in een bepaalde staat de toekomstige rewards voor elke mogelijke actie te voorspellen. Bijvoorbeeld: “Als je nog een windmolen neerzet gaan de kosten zoveel omlaag” of “Extra zonnepanelen zouden juist slecht zijn”. In het hoofdstuk Genetisch algoritme wordt uitgelegd wat een genetisch algoritme precies is.

Reinforcement learning werkt dus met discrete keuzes. Dit is goed om te bepalen of je nog een windmolen of juist nog een paar zonnepanelen erbij wilt zetten. Het werkt helaas minder goed voor vragen zoals: “Wat als we een deel van de zonnepanelen iets meer naar het oosten zetten” of “Wat als we ietsje minder zonnepanelen gebruiken?”.

Een genetisch algoritme bleef over als duidelijke keuze omdat dit algoritme ideaal is voor kleine aanpassingen zoals iets minder zonnepanelen en omdat het goed om kan gaan met elke kostenfunctie zonder dat het algoritme iets hoeft te weten over de werking hiervan. Dit laatste was belangrijk omdat de simulatie in matlab aangeleverd was en wij in python werken. Hierdoor werd een groot deel van de kostenfunctie een soort “black box”.

De hyperparameters voor een genetisch algoritme zijn niet zo belangrijk als bij neurale netwerken. Je hebt best wat speling en zonder de optimale waardes kom je er vaak ook. Ik heb er tuurlijk wel wat tijd aan besteed. Hierover is meer te lezen in het hoofdstuk Genetisch algoritme.

Planning

We hebben scrum gebruikt tijdens het project. De meesten hadden dit al eerder gebruikt en Michiel was er erg enthousiast over. Hij was dan ook onze scrum master. We gebruikten sprints van twee weken. Aan het begin van een sprint maakten we met zijn allen taken aan. Tijdens de sprint pakten we dan een taak en als we hem af hadden pakten we een nieuwe. We hadden zelfs een fysiek

scrumbord. Ik zeg hadden omdat dit in stukken is gescheurd en daarna is weggegooid. Dit bord hebben we dus niet meer.

Resultaat

Het resultaat ligt aan de gekozen kosten voor de opslag. Met lage opslagkosten kiest het algoritme een opstelling die in een jaar net genoeg produceert om rond te komen. Op momenten dat er een overschot is wordt dan bijna alles opgeslagen voor later. Als de opslagkosten te hoog worden zorgt het algoritme ervoor dat er bijna altijd genoeg geproduceerd wordt met minimale opslag. Helaas ligt de tweede situatie dichter bij de realiteit. De kosten voor de opstelling zijn op het moment van schrijven te hoog en de energiecentrale zou niet rendabel zijn.

Er lijkt geen situatie tussen de andere twee in te zitten. Er is een keerpunt waar meer zonnepanelen en windturbines neerzetten gewoon goedkoper is dan opslag. Ditzelfde lijkt tot op zekere hoogte te gelden voor de verhouding tussen windmolens en zonnepanelen. Het algoritme kiest in ons klimaat altijd voor meer windturbines met een minimale hoeveelheid zonnepanelen. In een ander klimaat zou dit juist andersom kunnen zijn.

Suggesties voor de toekomst

Ik noemde eerder al dat het algoritme in ons klimaat eerder voor wind- dan voor zonne-energie kiest. We hebben het genetische algoritme wel getraind op meerdere locaties in Nederland, maar niet op locaties in andere landen. Hier hadden we de weerdata niet voor. Ik denk wel dat het erg interessant zou zijn om te kijken wat eruit komt als je traint met weerdata uit andere landen. In Egypte zal het netwerk waarschijnlijk eerder voor zonnepanelen kiezen. Door de hogere irradiatie zal het genetische algoritme ook sneller kiezen voor meer zonnepanelen dan meer opslag. Bij een hogere irradiatie zijn de kosten per kilowattuur namelijk minder omdat er met hetzelfde paneel meer geproduceerd wordt.

Nu hebben we altijd op één jaar getraind. Dit komt doordat de simulatie in Matlab hiervoor gemaakt was. Ik denk dat het resultaat betrouwbaarder zou zijn als je op meerdere jaren traint. Dan ben je er zekerder van dat de opstelling ook met bijzonder slechte jaren om kan gaan.

Genetisch algoritme

Een genetisch algoritme werkt door heel random verzamelingen van waarden te maken. Deze verzamelingen worden ook wel individuen of chromosomen genoemd. Al deze verzamelingen bij elkaar vormen de populatie. Hieruit kiest hij de besten om weer nieuwe individuen te maken. Daarna verandert (muteert) hij de waarden van de nieuwe individuen een beetje door ze random te verhogen of te verlagen. Dit doet het algoritme een aantal keer achter elkaar (het aantal generaties). Uiteindelijk kies je het beste individu uit en dat is je resultaat. Hopelijk komt dit dan ook in de buurt van de meest efficiënte verzameling van waarden mogelijk.

De code van het genetische algoritme kan intimiderend overkomen, maar aan de andere kant is het ook wel weer elegant. Je hebt namelijk alleen Numpy nodig om de code te schrijven. Met de Jupyter Notebook die ik voor bij de presentatie gemaakt heb, is de werking iets beter te begrijpen.

Ik heb de code zelf geschreven omdat ik precies wist wat ik wilde en het meer tijd zou kosten om uit te zoeken hoe bestaande libraries werken en welke ik dan het beste zou kunnen kiezen. Er is niet één library die bijna iedereen gebruikt. Ik heb bij een korte zoektocht minstens vijf verschillende frameworks gevonden die allemaal hun eigen features en eigenaardigheden hebben. De meesten waren helaas niet erg goed gedocumenteerd.

Ons genetische algoritme is ietsje anders dan een standaard algoritme. Michiel had ergens gevonden dat je de kans op het vastzitten in lokale minima kunt verkleinen door niet alleen een aantal van de beste, maar ook een aantal van de meest verschillende individuen te selecteren om de nieuwe generatie mee te maken. Daarna bedacht ik me nog dat een aantal random individuen selecteren ook nog zou kunnen helpen.

Het algoritme heeft een aantal hyperparameters: de mutatie kans, de mutatie sterkte, het aantal beste individuen voor de volgende generatie, het aantal meest verschillende individuen voor de volgende generatie, het aantal random individuen voor de volgende generatie en of je het beste individu onaangepast aan de nieuwe generatie toe wilt voegen zodat je in ieder geval nooit een stap achteruit gaat. Ik heb natuurlijk niet alle combinaties uit kunnen proberen, maar ik heb enigszins intuïtief de parameters aangepast en gekeken of het hierdoor beter ging of niet. Ik heb een erg simpel script geschreven om de gekozen configuratie te bekijken om te checken of de output van het algoritme redelijk is. Later hebben de andere groepsleden nog meer hyperparameters uitgeprobeerd en de conclusie was dat het niet erg veel uitmaakt.

Code

Naaste het genetische algoritme heb ik ook code geschreven om de kosten te berekenen, te trainen en om de voortgang op te slaan. Vooral de code om de kosten te berekenen heb ik vell tijd aan besteed. Het was niet simpel om zo veel mogelijk van de for loops uit de code te halen en te vervangen met numpy functies. Het resultaat is helaas niet meer goed leesbaar, maar draait wel snel. Ik heb de oude (langzamere) voor de zekerheid erin laten staan.

Daarna heb ik anderen geholpen met het toevoegen van hun eigen ideeën en creaties aan de code. Ik heb geprobeerd ze uit te leggen wat de relevante stukken code doen en waarom ze hun code beter op een bepaalde manier kunnen schrijven.

Ik heb ook veel geholpen met het schrijven van efficiënte numpy code. Het is erg makkelijk om langzame python code te schrijven. Voor het trainen is het natuurlijk belangrijk dat alles zo snel mogelijk draait. Omdat ik al best wat ervaring met numpy heb, weet ik goed hoe je dingen zoals for loops het beste kan vermijden en hoe je bepaalde functies, boolean indexing en array indexing creatief kan toepassen. Hierdoor zijn belangrijke stukken code erg snel geworden.

Ik heb als indicatie van wat ik bedoel met numpy code een stukje code toegevoegd dat ik voor het project geschreven heb om een functie in matlab te vervangen. In dit geval vermenigvuldigd het de in_values keer twee omdat de waarden in array b twee keer zo groot zijn dan die in array a. In de praktijk is a een array van windsnelheden en b een array van energieproductie van een windmolen bij een bepaalde snelheid. De code kijkt tussen welke twee waarden in array a een input waarde ligt en geeft dan een proportionele output door de twee waardes uit array b te pakken. Het is van belang dat dit gebeurt zonder loops of if statements.

Presentaties en paper

Ik heb een aantal presentaties gegeven. Ik heb natuurlijk ook geholpen bij de voorbereiding van veel andere presentaties. De eerste presentatie was een open presentatie waar ik uitlegde hoe we ons project tot dan toe hadden aangepakt en wat onze resultaten waren. Ik had het idee dat deze presentatie erg goed ging omdat er maar één vraag was, maar dat zou natuurlijk kunnen betekenen dat het juist erg slecht ging.

De tweede presentatie was een presentatie over hoe een genetisch algoritme werkt en wat je er allemaal mee kan. Hier heb ik ook een Jupyter Notebook voor gemaakt om uit te leggen hoe zo'n genetisch algoritme precies werkt.

We hebben ook met de hele groep gepresenteerd aan de gasten bij de intreerede van Sander Mertens en ik heb met Michiel gepresenteerd tijdens de techathon. Deze is op YouTube te zien.

Paper

Ik heb een eerste opzet voor de paper in latex gemaakt. Ik heb er alvast een grafiekje en een (nep) referentie in gezet zodat de rest wist hoe het werkt. Nadat de rest hun delen hadden ingevuld hebben we met zijn alle het document verbeterd.

Evaluatie

Ik heb aan het begin van het project best een aardig stuk van de code geschreven. Ik was de enige met veel python ervaring, en ook de enige met veel machine learning ervaring. Hierdoor ben ik aan de code gaan zitten terwijl anderen andere taken op zich namen. Dit leek ons toen een handige verdeling. Ik had in de eerste week eigenlijk al bijna alle code voor het project geschreven.

Voor mij was het namelijk hele simpele code om te schrijven, maar voor de rest was dit misschien een leuke uitdaging geweest. Ik vraag me daarom af of het wel een goed idee was om in mijn eentje aan de code te beginnen. Er waren natuurlijk ook voordelen. Doordat ik met de meeste bestanden begonnen ben heb ik de toon gezet voor de programmeerstijl en de architectuur. Ik denk dat dit de leesbaarheid voor mensen die er in de toekomst mee werken heeft verhoogd.

De rest wilde ook een kans om wat code te schrijven, dus heb ik daarna bijna geen code meer geschreven. Ik heb hooguit wat korte stukje numpy code geschreven. Ik heb me meer gefocust op het helpen van anderen met hun code. Om te beginnen heb ik uitgelegd hoe mijn code werkt en waar zij hun code toe konden voegen. Ik heb ze ook individueel geholpen met hun code en het maken van keuzes over hoe iets het beste geprogrammeerd kan worden. Ze hadden namelijk niet allemaal evenveel ervaring met het schrijven van code. Ik denk dat ze daar veel aan gehad hebben en dat ze er veel van geleerd hebben.

Daarnaast heb ik ook laten zien hoe je efficiënte numpy code schrijft. Dit is namelijk niet erg intuïtief als je er geen ervaring mee hebt. Als je die ervaring wel hebt kan je veel leuke dingen doen met dingen zoals boolean indexing, array indexing en de vele numpy functies. Hier heb ik in het verleden zelf veel aan gehad en dit wilde ik anderen meegeven. Ik denk dat dit wel gelukt is, omdat groepsleden elkaar er halverwege het project op begonnen te wijzen dat bepaalde stukken efficiënter geschreven kunnen. Bijvoorbeeld dat een for loop onnodig is en dat het ook kan met een combinatie van functies en indexing.

Ik heb natuurlijk ook een aantal presentaties gegeven. De eerste open presentatie vond ik erg goed gaan. Ik wilde aan de aanwezigen vertellen wat ons project inhield, hoe wij het aan hebben gepakt en wat het resultaat was. Er was maar één vraag dus ik neem aan dat de presentatie als helder en duidelijk ervaren is.

Ik heb het eerste opzetje voor de paper in Latex gemaakt. Ik heb er wat voorbeeldjes van verwijzingen en afbeeldingen bij gezet. Ik denk dat dit goed heeft uitgepakt. Onze paper ziet er professioneel uit en door de voorbeelden was het voor de anderen niet erg moeilijk om dingen toe te voegen.

Mijn leren

Ik was een beetje bang dat ik niet veel zou leren. Ik had al best veel kennis over python, data science en deep learning. Ik wilde in ieder geval leren hoe andere vormen van machine learning werken. Ik wilde ook beter bekend worden met Pandas. Hier had ik op mijn werk niet veel mee gedaan, terwijl het wel een veelgebruikte python library is. Tot slot wilde ik graag een keer een rapport maken met Latex. Hier had ik goede dingen over gehoord en ik wilde het graag een keer gebruikt hebben om te weten hoe het werkt.

Ik had in eerste instantie aangenomen dat de Datacamp opdrachten zinloos zouden zijn. Ik kon immers al programmeren in python. De eerste course was inderdaad nutteloos, maar daarna begon ik wel het een en ander te leren. Ik leerde rustig op mijn eigen tempo over Pandas. Hier werd ik wel blij van, omdat ik daar juist meer over wilde weten.

Ik leerde ook veel over het maken van grafieken. Dit deed ik op mijn werk eigenlijk nooit. Daardoor had ik er ook niet aan gedacht toen ik aan de minor begon. Dit was een aangename verassing. Het was leuk om te leren hoe ik allerlei plotjes kan maken. Hoewel ik er op mijn werk dus niet veel mee doe, denk ik wel dat ik er in de toekomst wat aan heb. De meeste mensen vinden het fijn als data gevisualiseerd wordt. Het is ook handig om te weten hoe je grafieken kan maken voor bijvoorbeeld presentaties en rapporten.

Eén van mijn doelen was om te leren over andere vormen van machine learning. Dit is eigenlijk vanzelf gegaan omdat veel van de lessen daarover gingen. Ik denk dat ik nu goed begrijp welke algoritmen in welke situaties gebruikt kunnen worden. Hier ben ik erg blij mee, omdat ik denk dat ik dit in de toekomst op mijn werk toe kan passen. Ik vind het alleen jammer dat ik ze niet in mijn project heb kunnen uitproberen.

Omdat ik graag meer wilde leren over Latex heb ik ervoor gekozen om zelf de eerste versie van de paper in Latex te maken. Hierdoor heb ik best wat geleerd over hoe het werkt. Hier ben ik blij mee omdat ik deze kennis later goed kan gebruiken. Dit was dus een goede keuze.

Verder heb ik door de presentaties en evenementen nog meer kunnen oefenen met presenteren. Ik had niet het idee dat ik dit nodig had, maar het kan natuurlijk ook geen kwaad. Ik heb vooral veel gehad aan de evenementen waar we als groep aan hebben deelgenomen. Daardoor ben ik toch weer wat ervaringen rijker.

Het project

Ik vind dat het project erg goed gegaan is. Ik heb al eerder aangegeven dat ik me er zorgen over maakte dat ik in de eerste week erg veel van de code geschreven had. Uiteindelijk zorgde dit er wel voor dat we constant voorliepen op schema. Dit creëerde een ontspannen werksfeer waarin iedereen tijd kon nemen voor dingen die ze wilden leren en dingen die niet absoluut noodzakelijk waren voor het project.

Eén van de dingen die niet absoluut noodzakelijk waren voor het project was het herschrijven van de simulatie in python. Dit is Jer uiteindelijk gelukt en dit heeft veel opgeleverd. Hierdoor konden we in minuten in plaats van in uren of zelfs dagen trainen. Hierdoor konden we meer verschillende runs draaien en konden we dus meer dingen uitproberen.

Dit zal in de praktijk voor de volgende gebruikers ook erg waardevol zijn. Als een bedrijf langs komt en wilt weten wat de beste opstelling voor een groene energiecentrale zou zijn, is het natuurlijk een stuk leuker als je ze na een paar minuten het antwoord al kan geven, dan als je moet zeggen dat ze het de volgende dag pas horen.

Een ander ding dat niet echt noodzakelijk was voor het project was een grafische userinterface. Ook dit verhoogt de waarde voor de eindgebruiker. Niet iedereen werkt graag met een command line interface. Het is door de grafische interface ook leuker om het product aan bedrijven en andere bezoekers te laten zien. Zo een grafische interface geeft iets dat je laat zien toch wat extra pit. Mensen zien het dan eerder als een echte applicatie.

Ik vind dat we het project goed hebben laten zien bij de intreerede van Sander Mertens. Hier hebben we met een standje gestaan en aan de bezoekers uitgelegd wat we aan het doen waren. Er waren veel geïnteresseerden en we kregen ook veel goede vragen. Hierdoor heb ik het gevoel dat we veel interesse hebben opgewekt onder de bezoekers en dat mensen graag een vervolg zien van ons project.