Deel 1: Productieproces

Overzicht

Deze oefening bestaat uit het analyseren van productiegegevens van verschillende fabrieken in meerdere steden. Deze fabrieken produceren een chemische vloeistof die gemeten wordt in hectoliters, waarbij alle waarden afgerond worden naar het dichtstbijzijnde gehele getal.

Het doel is om een model te ontwikkelen dat de verdeling van toekomstige productiehoeveelheden kan voorspellen. In eerste instantie willen we de dagelijkse productiehoeveelheid modelleren. Vervolgens zullen simulaties worden gebruikt om de verdeling van de productie over meerdere dagen te bepalen. Als bonus vergelijken we onze simulaties met wat de centrale limietstelling ons vertelt.

Voor het huidige toepassingsgebied wordt aangenomen dat de productie op specifieke dagen onafhankelijk en identiek verdeeld is (i.i.d.). Om de totale productie over een periode van 5 dagen te simuleren, kan men dus 5 waarden p_1, p_2, p_3, p_4, p_5 simuleren, en de som $p = \sum_{i=1}^5 p_i$ vertegenwoordigt een realisatie van de simulatie van de productie over een periode van 5 dagen. Merk op dat dagen met 0 productie ook voorkomen en deze afzonderlijk beschouwd moeten worden.

Gegevens

In de map **data_productie** staan twee soorten bestanden. Het eerste, *master_data.json*, bevat informatie over de Maximale Duurzame Snelheid (MSR, Maximal Sustainable Rate) in de productie-installatie. Het tweede type bevindt zich in *daily_production/city/date.json*. Deze bestanden bevatten details over de productie voor elke datum, waaronder de volgende waarden:

- *DoW*: Day of the Week.
- *hour*: Hour of registration.
- *minute*: Minute of registration.
- *date*: Date of registration.
- *maintenance*: Yes/No, indicating scheduled maintenance on this date.
- *prod_loss*: Unexpected production loss on this date.
- *prod_loss_perc*: Percentage of MSR lost.
- *production*: Total production on this date.

Gedetailleerde opdracht

In deze opdracht volg je een reeks stappen om een projectstructuur op te zetten, versiebeheer met Git op te zetten, een virtuele omgeving te creëren met Anaconda (andere methodes zijn ook toegestaan) en een Python-programma te ontwikkelen om een productieproces te simuleren.

Stap 1: Gegevens downloaden en organiseren

Begin met het downloaden van alle relevante gegevens en het opzetten van een zinvolle codebase.

Stap 2: Git Repository initialiseren

Maak van de projectmap een Git repository op je persoonlijke GitHub account om wijzigingen bij te houden en effectief samen te werken. Je kunt de gegevens reeds toevoegen aan deze Git Repository.

Stap 3: Creëer een virtuele omgeving met Anaconda

Gebruik Anaconda om een virtuele omgeving voor je project te maken. Zorg ervoor dat je alleen de benodigde pakketten installeert die nodig zijn voor het project.

Stap 4:Lezen van gegevens in Python

Schrijf code om alle gegevens uit stap 1 in te lezen in Python.

Stap 5: Bepaal een geschikte verdeling voor het dagelijkse productieproces

Onderzoek en bepaal een geschikte kansverdeling die het beste past bij de kenmerken van het productieproces op een dag dat er normaalgesproken gewone productie zou moeten zijn op basis van de gegevens. Je mag hierbij uitgaan van het volgende:

- De dagen waarop we weten dat er geen productie is, hoeven niet mee in de kansverdeling te worden gestoken.
- Maak een onderscheid tussen de dagen met 0 productie en de dagen met reguliere productie dient wel gemaakt te worden, dus in je kansverdeling heb je gewoon een bepaalde kans op 0 productie.
- Voor de dagen met productie mag je veronderstellen dat de productie normaal verdeeld is, je kan dus `scipy.stats.norm.fit(data)` gebruiken om de parameters van de beste normale verdeling te vinden.

Stap 6: Simulatieprogramma ontwikkelen

Maak een Python-programma dat het productieproces kan simuleren over een door de gebruiker bepaalde periode, `n` dagen. Het programma moet de gebruiker flexibiliteit bieden om de duur van de simulatie te kiezen. Het simulatieprogramma mag veronderstellen dat er op alle `n` dagen normaalgesproken productie zou moeten zijn, maar elke dag heeft wel een kans om 0 productie te hebben.

Stap 7: Gebruik je simulatie

Gebruik je simulatieprogramma om de productiekansen over een periode van `n` dagen te simuleren. Maak ook een grafiek waarin je de vorm van de empirische cdf: $$F(x) = P{X_n \leq x},$$ met X_n de willekeurige variabele die gelijk is aan de productie over n (i.i.d.) dagen. Voor het bepalen van de ecdf kan je gebruik maken van de functie `scipy.stats.ecdf`.

Pas je functie ook eens toe op een periode van 7 dagen. Je code zou er ongeveer zo uit moeten zien:

```
import matplotlib.pyplot als plt
import numpy als np
xx = np.linspace(0, 10 ** 4)
data = simulation(7)
Fxx = cumulatieve_distribution_function(xx)
plt.plot(xx, Fxx)
```

Bonus: Vergelijken met de centrale limietstelling

Bekijk de verdeling die je bepaalde in de vorige stap. Deze geeft je de verdeling van de productie over een aantal dagen. De centrale limietstelling zegt dat als je een groot aantal onafhankelijke waarnemingen samen normaal verdeeld zijn. Kan je deze observatie visueel voorstellen door de empirische cdf te vergelijken met de theoretische cdf van de normale verdeling, dit voor een stijgend aantal dagen `n` die je accumuleert?

Deel 2: Autoproductie

Overzicht

We kijken naar de dataset `cars.csv`, deze bevat informatie over de verkoop van 2^e hands Volvo's. We hebben hier een aantal vragen over, beantwoord deze vragen door een beetje code te schrijven en de antwoorden op te schrijven in een Jupyter Notebook. Hiertoe hebben we ook de notebook `autoproductie.ipynb` voorzien.

Vraag 1:

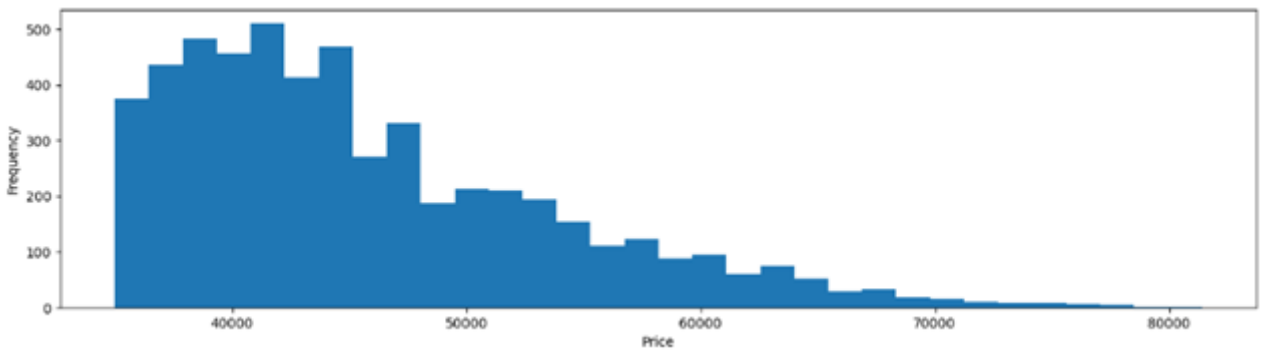
Wat voor soort data zijn `year`, `fuel_type` en `seller_rating`? Motiveer je antwoord.

Vraag 2:

Ik vraag me af hoe een gewoonlijke (mbt prijs) 2^e hands Volvo eruit ziet. Kan je deze zoeken en tonen?

Vraag 3:

We kunnen de verdeling van de prijs van 2^e hands Volvo's voorstellen met een histogram:



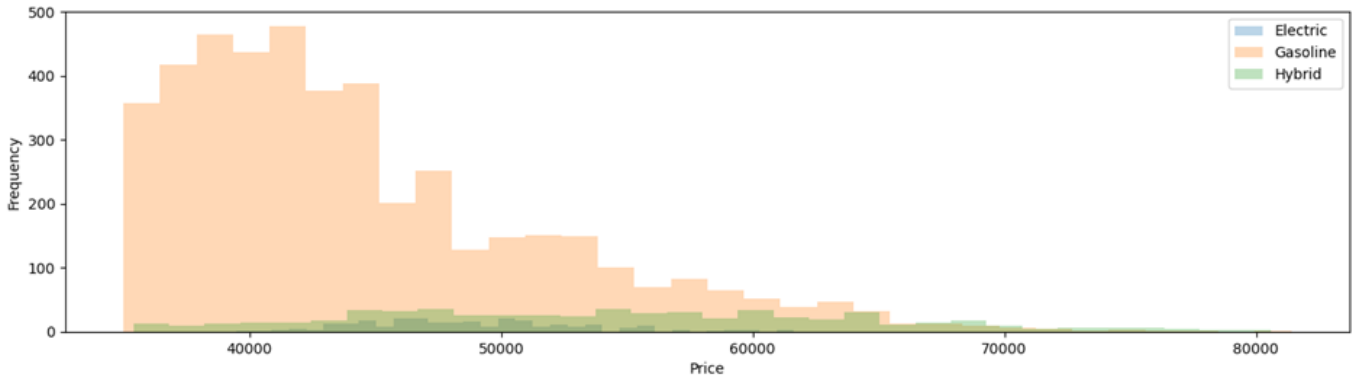
- Waar zou je verwachten dat het gemiddelde +/- ligt voor deze verdeling?
- Waar zou je verwachten dat de mediaan +/- ligt?

Teken dit histogram met `.hist` en gebruik `.axvline` om de mediaan en het gemiddelde toe te voegen aan de plot. Komt dit overeen met je verwachting?

- Kan je de emprisiche cummulatieve distributie functie tekenen voor de verdeling die overeenkomt met dit histogram (zie ook `scipy.stats.ecdf`).

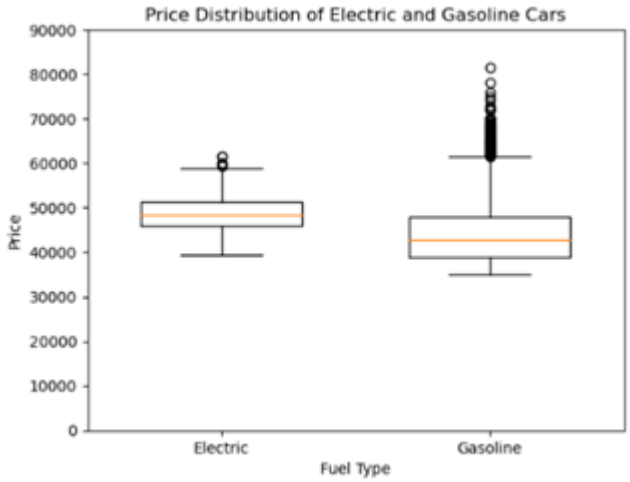
Vraag 4:

Ik wil nu de verschillende fuel types vergelijken hoeveel deze kosten. Hiervoor maak ik volgende afbeelding met meerdere histogrammen hoe kunnen we deze beter maken? Welke alternatieven zijn er nog?



Vraag 5:

Gegeven volgende boxplots van de prijs van elektrische & benzine wagens, zijn volgende uitspraken waar/onwaar/niet te besluiten & leg uit:



- De spreiding op de prijs van elektrische wagens is groter dan die van benzinewagens.
- Een wagen die meer dan 50 000 euro kost heeft een grotere kans om elektrisch te zijn dan benzine.
- Een elektrische wagen heeft een grotere kans om meer dan 50 000 euro te kosten dan een benzine wagen.

Vraag 6:

Maak een visualizatie om het verband tussen mileage & price te tonen?

Vraag 7:

Volgens deze data is de gemiddelde prijs van 2^e hands elektrische Volvo's 48 000 euro; als we veronderstellen dat onze dataset representatief is voor alle 2de hands volvo's, hoe kan je de accuraatheid van deze schatting nagaan?

Deel 3: Je oplossing delen

Voeg alle oplossingen van Deel 1 en Deel 2 samen op een zinvolle manier. Maak vevolgens een *README.md* bestand om gebruik van je oplossing te documenteren en nieuwe gebruikers te begeleiden door je project.

Geef zeker ook een beknopte uitleg over de projectstructuur en voeg een *YAML* file toe die ik kan gebruiken om een virtuele omgeving op te zetten. Voeg ook een link toe naar de github pagina waarop deze code beschikbaar is. Eens dit gebeurd is, doe het volgende:

- Steek je volledige codebase (inclusief data) in een zip bestand. Gebruik als naam van je `.zip` bestand het formaat `<naam1>_<naam2>_<naam3>.zip` waarbij je `<naami>` vervangt door de naam van het `i`'de lid an je groepje.
- Stuur dit `.zip` bestand in op de leeromgeving.