Inlämningsuppgift: Optimering av Förtjänst för Lindas Lustfyllda Bud & Åkeri

Kurs: Applicerad AI
Sebastian Öhman

25 november 2024

Introduktion

Lindas Lustfyllda Bud & Åkeri är i stort behov av hjälp med att utveckla ett program som optimerar förtjänsten varje dag för deras 10 budbilar (`bil_1`, `bil_2`, ...), som utför leveranser varje dag.

Varje kväll/natt mellan kl 23:00 och 04:00 uppdateras en fil, `lagerstatus.csv`, som innehåller data om nuvarande lager med paket som ska levereras. Filen ser ut på följande sätt:

```
Paket_id,Vikt,Förtjänst,Deadline
2234938218,3.4,2,3
2234938219,2.1,3,1
2234938220,4.0,4,0
2234941232,1.2,8,6
2234939231,8.2,5,3
2234939232,5.7,6,-1
2234939444,10.8,10,2
...
```

Beskrivning av data

- **Paket_id**: Paket-ID för det unika paketet med tillhörande information.
- **Vikt**: Vikt på paketet i kg.
- **Förtjänst**: Kategori mellan 1 och 10 som symboliserar förtjänsten för Lindas.
- **Deadline**: Antal dagar kvar tills deadline för leverans till kund. Vid överskriden deadline tillämpas en straffavgift på $-(x^2)$ där x är antal dagar efter deadline. Här kan alltså förtjänstvärdet landa utanför de satta värdena ovan.

Kapaciteten för en budbil är 800 kg och får ej överskridas.

Uppgift

Er uppgift är att skriva ett program som berättar vilka paket som ska till vilken budbil. Paketering av budbilarna börjar kl 05:00.

Vi vill också veta:

- Den uppmätta förtjänsten för dagens leveranser.
- Den totala straffavgiften för paket som är kvar i lager.
- Antal paket kvar i lager.
- Den totala förtjänsten kvar i lager (exklusive straffavgiften).
- Statistik på fördelningen av vikt och förtjänst för paketen, både i bilarna och de som är kvar i lager:
  - Histogram.
  - Medelvärde.
  - Varians.
  - Standardavvikelse.

Tips

- Ett paket kan inte vara i två budbilar samtidigt.
- Alla paket i lager behöver inte alltid paketeras i budbilarna.
- Tänk på vilken policy vi vill ha när det gäller försenade paket!
- Tänk på att `lagerstatus.csv` kan innehålla **många** paket.
- Använd tekniker för inklusion/exklusion vid val av paket.
- Använd `NumPy` och `Matplotlib` för beräkningar och visualiseringar.
- För er egen del, bygg in en *optimize-tracker* som kollar hur algoritmen förbättrar resultatet för varje generation för att hitta ett optimalt stoppkriterium.

Genomförande

1. **Datahantering**:
   - Läs in `lagerstatus.csv` och bearbeta data korrekt.
   - Hantera stora datamängder effektivt.
2. **Algoritmutveckling**:
   - Utveckla en algoritm som optimerar förtjänsten under givna begränsningar.
   - Ta hänsyn till deadlines och straffavgifter.

- Se till att totalvikten per budbil inte överstiger 800 kg.

3. **Resultat och Analys**:
   - Presentera resultaten på ett tydligt sätt.
   - Generera nödvändig statistik och visualiseringar.
   - Diskutera eventuella trade-offs och beslutsfattande i er algoritm.
4. **Optimering**:
   - Implementera en mekanism för att spåra och förbättra algoritmens prestanda över tid.
   - Definiera ett stoppkriterium för när optimeringen ska avslutas.

Inlämning

- Källkoden till programmet.
- En kort rapport (1-2 sidor) som beskriver er lösning och hur den uppfyller kraven.
- Visualiseringar och statistik enligt ovan.

Bedömning

Bedömningen kommer att baseras på:

- Korrekthet och effektivitet i algoritmen.
- Hur väl ni uppfyller de givna kraven och hanterar begränsningarna.
- Kvaliteten på er kod (struktur, läsbarhet, kommentarer).
- Tydlighet och insikt i rapporten.
- Kvaliteten på visualiseringar och statistisk analys.

Frågor

För eventuella frågor, kontakta kursledaren via e-post: `[e-postadress]`.