Projekt I – Linjär regressionsmodell i ett inbyggt system

Mål

- Kunna träna en regressionsmodell baserad på linjär regression i mjukvara för detektering av enkla mönster.
- Kunna sätta sig in i ett mellanstort projekt med befintliga drivrutiner och lägga till ny funktionalitet med bibehållet designmönster.
- Kombinera tidigare kunskaper om inbyggda system med nyvunnen kunskap inom maskininlärning.

Totalt 4p (G = 2p, VG = 4p)

Uppgiftsbeskrivning

- Låt oss spela ett litet rollspel, där ni är utvecklare och er lärare är er chef. Ponera att ni arbetar som embeddedutvecklare på ett företag i Göteborg. En av era kunder har tidigare köpt ett inbyggt system av ditt företag. Detta system utgörs av en ATMega328P processor samt omkringliggande hårdvara. Systemet fungerar i dagsläget mestadels bra. Den använda temperatursensorn TMP36 har dock visat sig inte vara helt konsistent beroende på omgivning, fukt med mera. Denna sensors insignaler varierar därmed över tid, varvid det inte är möjligt att beräkna temperaturen med en exakt ekvation.

- I stället för att byta hårdvara, vilket hade varit väldigt dyrt, då det rör sig om många system, har kunden bestämt sig för en mjukvarulösning. Därmed har ert företag fått i uppdrag att implementera en enkel maskininlärningsalgoritm i befintlig kodbas för att prediktera rumstemperaturen utefter indata från temperatursensorn. Vid behov av kalibrering kan då en utvecklare enkelt träna om modellen genom att manipulera träningsdatan, kompilera och köra om programmet.

- Träningsdatan ska bestå utav 10 - 15 träningsuppsättningar bestående av:
    - **Indata** i form av insignalen från temperatursensorn (behöver beräknas via den A/D-omvandlade insignalen).
    - **Utdata** bestående av aktuell rumstemperatur i grader Celsius.

- Utgå vid träning att förhållandet mellan inspänningen från temperatursensorn samt aktuell rumstemperatur överensstämmer med databladet, se bifogad bilaga A. Träningen ska ske i början av programmet innan omkringliggande hårdvara initieras. Så fort avvikelser i den uppmätta temperaturen visar sig i fält ska en utvecklare enbart behöva manipulera träningsdatan i funktionen *main*), kompilera och köra programmet. Därefter ska temperaturen predikteras med hög precision.

- Er ”chef” (er lärare) har därför bett dig samt en kollega att implementera en linjär regressionsmodell för kunden enligt beskrivningen ovan. Denna implementering ska läggas till i befintlig kodbas, se länk i bilaga B.

- Era tasks är följande:
    - Implementera en egen linjär regressionsmodell (interface + implementationsklass) i aktuell kodbas. All tillagd kod ska hålla samma stil som befintlig kod, vilket också gäller kommentarer. Algoritmen ska kunna tränas med ett givet antal epoker samt lärhastighet.
    - När modellen har tränats ska temperaturen predikteras och skrivas ut i en seriell terminal vid nedtryckning av en tryckknapp eller 60 sekunder efter senaste prediktionen. Oavsett knapptryckning eller inte ska alltså temperaturen skrivas ut var 60:e sekund.

- Som nämndes tidigare ska träningen ske direkt när systemet startar, alltså innan initiering av omgivande hårdvara. Vid träning ska därmed nedtryckning av tidigare nämnd tryckknapp inte medföra någon temperaturmätning.

Genomförande och examination

- Ni ska/får jobba i team om två på uppgiften.
- Ni får rådfråga er ”chef” (lärare) för handledning gällande konstruktionen.
- Ni är välkomna att ta inspiration från externa källor, vilket även innefattar kod från klassrepot, men ni ska göra en egen implementering. Kopiera alltså inte någon annans kod.
- Koden ska laddas upp på GitHub. Er ”chef” (eller rättare sagt lärare) kommer genomföra en review via denna plattform. Systemet ska också demonstreras live för chefen.

Utvärdering

- I samband med inlämningen, vänligen lämna in en kort diskussion i en README-fil där ni beskriver följande:
  1. Vad lärde ni er av projektet?
  2. Vad var lätt/svårt?
  3. Vad hade ni velat ha lärt er mer innan projektet?
  4. Övriga kommentarer?

Betyg

Godkänt (G) – 2p

- Leveransen uppfyller tidigare uppställda krav.

Väl godkänt (VG) – 4p

- Förutom kraven för G:
  - ✓ Man ska inte behöva ange antalet epoker samt lärhastigheten som ska användas vid träning. I stället ska er maskininlärningsalgoritm kunna justera lärhastigheten samt antal epoker som genomförs utefter hur träningen går. Om träningen går bra kan exempelvis lärhastigheten höjas 5 % tills ett visst max, börjar träningen går sämre ska lärhastigheten sänkas. Antalet epoker träning som genomförs ska inte heller behöva sättas; träning ska genomföras tills precisionen överstiger 99.9 %.
  - ✓ Ordningsföljden som era träningsuppsättningar används ska randomiseras varje epok som träning genomförs.
  - ✓ Koden ska också vara välskriven (i enlighet med resterade kod i kodbasen) och väl dokumenterad (tydliga Doxygen-kommentarer för alla klasser och publika metoder.

Bilaga A - Formler för beräkning av temperatursensor TMP36

- Sambandet mellan resultatet ADC_{value} från AD-omvandlingen samt den analoga inspänningen U_{IN} erhålls från temperatursensor TMP36 kan beräknas med följande formel:

$$U_{IN} = \frac{ADC_{value}}{1023} * 5\ V$$

- Sambandet mellan den omgivande temperaturen T samt den analoga inspänningen U_{IN} kan beräknas med följande formel:

$$T = (U_{IN} - 0{,}5) * 100,$$

vilket är ekvivalent med

$$T = 100U_{IN} - 50$$

Bilaga B – Kodbas

- Kodbasen hittar ni i kursrepot under *code/libatmega:*

https://github.com/Yrgo-24/machine-learning/tree/main/code/libatmega

OBS! *Två versioner av kodbasen finns tillgänliga, en skriven i C samt en i C++. Ni får välja vilken av varianterna ni vill använda.*

Ladda ned koden, exempelvis genom att ladda ned klassrepot. Kopiera underkatalogen *c* eller *cpp* (beroende på vilken version av kodbasen ni vill använda) till en lämplig Windows-mapp. exempelvis C-disken. Öppna projektet i Microchip Studio genom att dubbelklicka på bifogad projektfil. Before på använd kodbas har denna fil ändelsen *.cproj* eller *.cppproj.*