Del 1

1. **Hur är AI, Maskininlärning och Deep Learning relaterat?**
   Artificiell intelligens (AI) är som ett större begrepp som ska då härma mänskligt beteende som att fatta beslut, identifiera saker och förstå tal samt objekt. För att få datorer att förstå en viss datamängd så behövs maskininlärning för att träna modeller och anpassa det för olika syften baserat på data, utan att programmera för vardera del av datan. Hur Deep Learning också är relaterat är när man använder mer kraftfulla och mera avancerade former av maskininlärning som neurala nätverk med många lager. Det djupa nätverk som neurala nätverk har en massa lager som gör det möjligt att upptäcka komplexa mönster i stora mängder data. Samma neurala nätverk kan användas för till exempel bildigenkänning eller språkförståelse.

2. **Hur är Tensorflow och Keras relaterat?**
   Tensorflow är ett bibliotek i Python som möjliggör byggande av mer avancerade maskininlärningsmodeller och Keras är ett högre abstraktionslager på Tensorflow som förenklar skapandet av modeller. Keras möjliggör också snabbt byggande och testande av neurala nätverk med mindre kod och enklare läsbarhet. Men Tensorflow erbjuder stor flexibilitet och kraft till att skapa grafer och optimeringsstrategier som i sin tur gör så att Keras kan komma igång enklare med modellens struktur.

3. **Vad är en parameter? Vad är en hyperparameter?**
   När man tränar sin modell i neurala nätverk så har parametrar och hyperparametrar viktiga roller. Under träningsdelen av modellen så använder man sig av parametrar för att sätta vikter i ett neuralt nätverk som i sin tur avgör hur mycket av indata påverkar utfallet. Men när det gäller hyperparametrar så bestäms de innan en modell ska tränas för att styra hur modellen ska tränas. Ett exempel på hur hyperparametrar gör är hur snabbt modellen ska hantera inlärningshastigheten. Därför sätts hyperparametrar manuellt för att sedan kunna exprimentera med dem och validera dem.

4. **När man skall göra modellval och modellutvärdering kan man använda tränings-, validerings- och testdataset. Förklara hur de olika delarna kan användas.**
   Vid utveckling samt utvärdering av modeller i maskininlärning så delar man upp datan i tre delar som då heter träning, validering och test. I träningsdatan lär man modellen att känna igen mönster och i valideringsdatan så ser man hur modellen presterar med ny osedd data. Det finns något som kallas för overfitting (övertränad) och det kan komma upp i valideringen vilket kan betyda att man måste gå in och justera hyperparametrarna igen så att den inte blir övertränad. Till sist har man testdatan som är en utvärdering av modellens prestation. I testdatan ser man hur modellen har i praktiken presterat.

5. Förklara vad nedanstående kod gör:

```
#Här hämtar vi antalet kolumner (features) i träningsdatan x_train.
# Det är alltså längden på varje datapunkt (t.ex. hur många indata varje observation innehåller).
# Detta används som indataform till nätverket.
n_cols = x_train.shape[1]

#Vi skapar ett nytt sekventiellt neuralt nätverk med Keras.
#"Sekventiell" betyder att lagren läggs till i en linjär följd – varje lager följer efter det förra, utan grenar eller parallella vägar.
nn_model = Sequential()

#Här lägger vi till det första dolda lagret i nätverket.
# Det består av 100 noder (neuroner), och varje nod använder aktiveringsfunktionen ReLU (Rectified Linear Unit) – en vanlig funktion som hjälper modellen lära sig icke-linjära mönster.
#input_shape=(n_cols,) talar om för modellen hur många värden varje indataexempel innehåller, alltså antalet features.
nn_model.add(Dense(100, activation='relu', input_shape=(n_cols, )))

#Detta är ett dropout-lager, vilket är en form av regularisering.
# Vid varje träningssteg stängs 20 % (0.2) av noderna av i föregående lager – slumpmässigt.
# Detta gör att modellen inte förlitar sig för mycket på vissa noder och lär sig mer robusta mönster, vilket hjälper till att förhindra överträning (overfitting).
nn_model.add(Dropout(rate=0.2))

#Här lägger vi till ytterligare ett dolt lager med 50 noder och ReLU-aktivering.
# Detta lager lär sig ännu mer abstrakta representationer av indata, baserat på vad det första lagret har lärt sig.
nn_model.add(Dense(50, activation='relu'))

#Detta är utgångslagret. Det består av en enda nod eftersom vi gör en binär klassificering – till exempel "ja/nej", "positiv/negativ", "1/0".
#Vi använder sigmoid-aktivering, vilket gör att nodens output blir ett värde mellan 0 och 1 – alltså en sannolikhet.
nn_model.add(Dense(1, activation='sigmoid'))


#Här kompileras modellen – det innebär att vi bestämmer hur den ska tränas:
#optimizer='adam' betyder att vi använder optimeringsalgoritmen Adam, en kraftfull metod som anpassar inlärningshastigheten automatiskt.
#loss='binary_crossentropy' används som förlustfunktion. Den mäter hur långt ifrån rätt svar modellen är – perfekt för binära klassificeringsproblem.
#metrics=['accuracy'] gör att modellen under träning visar hur ofta den hade rätt (exakt match mellan förutsägelse och verkligt värde).
nn_model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy']
)

#Vi skapar ett early stopping-objekt, vilket är ett verktyg för att automatiskt stoppa träningen när modellen slutar förbättras.
#patience=5 innebär att träningen får fortsätta max 5 epoch utan förbättring i valideringsförlusten – därefter stoppas den.
early_stopping_monitor = EarlyStopping(patience=5)
#Här tränas modellen. Vi ger den både indata (x_train) och korrekta svar (y_train).
#validation_split=0.2 innebär att 20 % av träningsdatan används för validering, så att modellen kan utvärderas under träningen.
#epochs=100 anger att modellen får tränas i max 100 varv (epoch) över hela datan.
#callbacks=[early_stopping_monitor] innebär att om modellen inte förbättras under 5 epoch i rad, så avbryts träningen för att undvika överträning och spara tid.
nn_model.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=100,
    callbacks=[early_stopping_monitor]
)
```

6. **Vad är syftet med att regularisera en modell?**
   Syftet med att regularisera en modell är för att förhindra att modellen blir övertränad (overfitting) i träningsdatan. Alltså när modellen lär sig för mycket detaljer från träningsdatan och börjar prestera dåligt på ny data. Det finns olika regularisering tekniker som L1/L2-regularisering, Dropout och Early stopping.

7. **”Dropout” är en regulariseringsteknik, vad är det för något?**
   När man använder sig av dropout som en regulariseringsteknik så använder man det för att stänga av neuroner i nätverket vilket tvingar nätverket att inte bli beroende av specifika noder utan att lära sig mer spridda representationer.

8. **”Early stopping” är en regulariseringsteknik, vad är det för något?**
   Till exempel i en av de föregående frågorna hade vi early stopping som är en regulariseringteknik för att stoppa i träningen när prestandan uppnår sitt max och att valideringsdatan slutar förbättras. Den skyddar alltså modellen från att bli övertränad som i sin tur gör att den slutar prestera på ny data.

9. **Din kollega frågar dig vilken typ av neuralt nätverk som är populärt för bildanalys, vad svarar du?**
   Jag hade svarat att Convolutional Neural Networks (CNN) är det bästa valet för bildanalys. Eftersom CNN är specialbyggt för att hantera bilddata genom att fokusera på mönster i bilder som kanter, former och strukturer.

10. **Förklara översiktligt hur ett ”Convolutional Neural Network” fungerar.**
    Convolutional Neural Network (CNN) arbetar genom att först använda konvolutionslager som applicerar filter över bilden för att identifiera mönster. Därefter så använder den ett pooling-lager som minskar storleken på bilden men behåller viktiga egenskaper. Slutligen används dense layers (anslutna lager) för att fatta beslut baserat på de extraherade mönstren.

11. **Vad gör nedanstående kod?**

```
#model.save("model_file.keras"): Sparar hela modellen (struktur, vikter, konfiguration).
model.save("model_file.keras")
#load_model("model_file.keras"): Läser in den sparade modellen så du kan fortsätta använda den utan att träna om.
my_model = load_model("model_file.keras")
```

12. **Deep Learning modeller kan ta lång tid att träna, då kan GPU via t.ex. Google Colab skynda på träningen avsevärt. Skriv mycket kortfattat vad CPU och GPU är.**
    CPU (Central Processing Unit) är det som kallas för hjärnan i datorn som då utför allmänna beräkningar. CPU jobbar snabbt men begränsad och kan jobba men en sak i taget. En GPU (Graphics Processing Unit) är specialiserad för att utföra många liknande uppgifter samtidigt vilket gör att den kan vara effektiv för att träna Deep Learning modeller snabbt.

Del 2

En fördjupad och kritisk diskussion kring “hur din modell hade kunnat användas i verkligheten och vilka potentiella utmaningar och möjligheter (affärsmässiga, etiska och andra perspektiv du finner relevanta) som finns”.

Jag tänkte börja med att berätta vad E3 Control är och varför jag valde just dem. Innan jag påbörjade min utbildning hos EC som Data Scientist så gjorde jag min praktik som elektriker hos E3 och sedan fick jobba där. Så jag gick in på deras webbplats och såg att de inte har en chatbot samt att gamla kollegor som jag har kontakt med säger att de får göra uppgifter som inte tillhör deras arbetsposition. De uppgifter är till exempel att prata med kunder som ringer in för FAQ frågor som redan finns på hemsidan.

Så jag valde att göra chatbot till just deras FAQ frågor och har själv lagt in några små ändringar som fraktberäkning och kontaktuppgifter som då inte tillhör FAQ. Med min chatbot var idéen att minska onödiga samtal från kunder till medarbetarna där så att de kan fokusera på sitt arbetsområde.

Som potentiella utmaningar så vill jag påstå att chatbotten är långt ifrån klar och har förbättringsområden som till exempel användning av LLM modeller som OpenAI API. Men jag valde att göra chatbotten utan då jag vill inte betala för någon API-nyckel utan tänkte göra en gratis demo vilket fungerade ändå okej som demo. Sen har vi ju att om företaget lägger till nya FAQ, måste indexet samt datan uppdateras.

Avslutningsvis vill jag också påpeka debugpanelen jag använde mig av för att testa semantiska sökningar. Jag har valt att lämna den kvar för att kunna förbättra chatbottens svars förmåga i framtiden. Genom distansvärde (dist) så ser man hur lik texten är frågan och du får också ut ett utdrag av texten med upp till 200 tecken från matchade dokument som visas som exempel.

Självutvärdering

1. Vad har varit roligast i kunskapskontrollen?
   Jag personligen tycker att allt med kunskapskontrollen har varit kul att göra. Speciellt med bra genomgångar med Linus har hjälpt. Tycker dock att få använda sin kreativitet är nog det roligaste. Att få bestämma vad för chatbot du ska göra, vilket jag valde för mitt gamla företag.

2. Vilket betyg anser du att du ska ha och varför?
   Jag vill helst låta läraren bestämma det då jag anser ingenting utan jag gör mitt bästa och så får jag ta feedback.

3. Vad har varit mest utmanande i arbetet och hur har du hanterat det?
   Eftersom att jag gjorde en demo med hjälp av en “gratis” LLM modell som heter Ollama eller LLama3 i början så fick jag inte till det. Jag ville inte heller betala för OpenAI som GPT-4 vilket gjorde det lite mer utmanande de senaste veckan. Så jag skulle nog säga att den sista veckan har varit intensiv med min nuvarande chatbot.