Från bild till siffra

Klassificering av handskrivna siffror



Terese Eriksson

EC Utbildning

Kunskapskontroll 2 - Machine Learning

2025-03

# Abstract

This project aims to develop a machine learning model that predicts handwritten numbers. Several models, including logistic regression, SVM, Random Forest and XGBoost, was evaluated to find the one with the highest accuracy. In the end, XGBoost was the model that performed the best.

A Streamlit application for users to draw a number on a canvas or upload an image of a handwritten number for prediction was built to further test the model and to evaluate what kind of data pre-processing is needed for the model to perform optimally.

Innehållsförteckning

[1 Inledning 1](#_Toc193489074)

[1.1 Syfte och frågeställningar 1](#_Toc193489075)

[2 Teori 2](#_Toc193489076)

[2.1 Klassificeringsmodeller 2](#_Toc193489077)

[2.1.1 Logistisk regression 2](#_Toc193489078)

[2.1.2 SVM 2](#_Toc193489079)

[2.1.3 Beslutsträd 4](#_Toc193489080)

[2.1.4 Random Forest 4](#_Toc193489081)

[2.1.5 XGBoost 4](#_Toc193489082)

[2.2 Utvärderingsmått 4](#_Toc193489083)

[2.2.1 Confusion matrix 5](#_Toc193489084)

[2.2.2 Accuracy score 5](#_Toc193489085)

[2.3 Hyperparameteroptimering 5](#_Toc193489086)

[2.3.1 GridSearch 5](#_Toc193489087)

[2.4 Dimensionsreducering - PCA 6](#_Toc193489088)

[3 Metod 7](#_Toc193489089)

[3.1 Databeskrivning 7](#_Toc193489090)

[3.2 Modell 7](#_Toc193489091)

[3.2.1 Utforskande dataanalys 7](#_Toc193489092)

[3.2.2 Datapreprocessering 7](#_Toc193489093)

[3.2.3 Modellträning 8](#_Toc193489094)

[3.2.4 Modellval 8](#_Toc193489095)

[3.3 Streamlit-applikation 8](#_Toc193489096)

[3.3.1 Bildförbehandling 9](#_Toc193489097)

[3.3.2 Prediktion av siffra 9](#_Toc193489098)

[4 Resultat och Diskussion 10](#_Toc193489099)

[4.1 Modell 10](#_Toc193489100)

[4.2 Streamlit-applikation 11](#_Toc193489101)

[5 Slutsatser 14](#_Toc193489102)

[5.1 Frågeställningar 14](#_Toc193489103)

[6 Teoretiska frågor 15](#_Toc193489104)

[7 Självutvärdering 20](#_Toc193489105)

[Appendix A 21](#_Toc193489106)

[Källförteckning 22](#_Toc193489107)

# Inledning

Bildigenkänning är en central del inom maskininlärning. Detta möjliggör för att kunna automatisera processer där det idag krävs att en person manuellt läser av vad som finns på bilden. Möjligheten att automatiskt kunna läsa av siffror har många praktiska tillämpningar såsom posthantering och formulärigenkänning. Att utveckla modeller som klarar av detta kan alltså göra stor nytta.

## Syfte och frågeställningar

Syftet med detta arbete är att utvärdera och jämföra olika maskininlärningsmodeller för klassificering av handskrivna siffror. Den modell som presterar bäst kommer att implementeras i en Streamlit-applikation för vidare utveckling av hur klassificering av handskrivna siffror kan utföras på bästa möjliga sätt.

För att uppfylla syftet så kommer följande frågeställning(ar) att besvaras:

1. Vilken maskininlärningsmodell ger bäst noggrannhet för handskrivna siffror i MNIST-datasetet?
2. Vad krävs för att en modell tränad på MNIST anpassas för att tolka användarinmatade siffror i en Streamlit-applikation?

# Teori

I detta avsnitt presenteras teori som ligger till grund för projektet.

## Klassificeringsmodeller

Inom klassificeringsproblem är målet att prediktera ett diskret värde eller en klass för en observation baserat på dess egenskaper. Detta kan till exempel innebära att identifiera objekt i en bild. Binär klassificering är den enklaste formen av klassificering och innebär att det endast finns två möjliga värden. När det finns fler än två möjliga värden kallas det för multiklass klassificering. (GeeksforGeeks, 2025)

Klassificeringsmodeller av intresse för detta projekt presenteras nedan.

### Logistisk regression

I logistisk regression utförs prediktion genom att sannolikheten för att en observation tillhör en viss klass estimeras. Observationen tilldelas sedan den klassen med högst sannolikhet. Logistisk regression är en binär klassificerare. Om sannolikheten är över 50% kommer modellen prediktera att observationen tillhör den kategorin, och om sannolikheten är lägre än 50% kommer modellen att prediktera att observationen tillhör den andra kategorin. (Géron, 2019)

I logistisk regression är utgångspunkten att utdata modelleras som en linjär kombination av indata, plus en felterm. Precis som i linjär regression. Se Ekvation (1) nedan. (GeeksforGeeks, 2025)

( 1 )

Eftersom z i Ekvation (1) kan anta vilket numeriskt värde som helst används Sigmoid-funktionen för att begränsa utfallet till ett värde mellan 0 och 1, vilket då kan tolkas som en sannolikhet. Se Ekvation (2) nedan. (GeeksforGeeks, 2025)

( 2 )

Sannolikheten för att en observation tillhör en kategori beräknas enligt nedan. (GeeksforGeeks, 2025)

Logistisk regression kan även utvidgas för att användas för multiklass klassificering. Istället för att använda Sigmoid-funktionen, som endast fördelar sannolikheten över två klasser, används softmax-funktionen som kan fördela sannolikheterna över flera kategorier. Observationen tilldelas den kategori med högst sannolikhet. Detta kallas Softmax regression. (Géron, 2019)

Modellen för logistisk regression är linjär och fungerar bäst när förhållandet mellan utdata och indata är ungefär linjärt (GeeksforGeeks, 2025). Om så inte är fallet kan andra modeller så som SVM och Random Forest vara bättre alternativ.

### SVM

Support vector machine (SVM) fördelar observationer i olika klasser genom att hitta det hyperplan som separerar kategorierna så optimalt som möjligt, alltså en beslutsgräns. Detta illustreras i Figur 1. Två olika kategorier representeras av färgerna gul och blå. I den vänstra plotten visas tre olika alternativ för beslutsgränser. Både den lila och gröna beslutsgränsen separerar kategorierna, men är känsliga för nya observationer. Den röda streckade linjen misslyckas helt med att separera klasserna. I den högra plotten representerar den svarta heldragna linjen en beslutsgräns som tagits fram med SVM. Denna linje befinner sig så långt bort från de närmsta observationerna i respektive kategori, vilka representeras av de streckade linjerna, som möjligt. Dessa observationer kallas för support vectors. Målet är att skapa en så bred ”gata” som möjligt. (Géron, 2019)

A diagram of lines and dots

AI-generated content may be incorrect.

Figur 1 - Beslutsgränser för att separera två olika klasser.

När alla observationer strikt måste vara på rätt sida om ”gatan” kallas detta för *hard margin classification.* Om ett dataset inte är linjärt separerbart eller om det finns outliers kan det skapa problem. Detta illustreras i Figur 2. I den vänstra plotten är det inte möjligt att skapa en beslutsgräns som separerar de båda kategorierna och i den högra plotten blir ”gatan” väldigt smal, vilket förmodligen leder till dålig generalisering. För att hantera detta problem kan *soft margin classification* användas, där det tillåts att observationer befinner sig på eller på fel sida av ”gatan”. Vid modellering måste en avvägning mellan att skapa en så bred ”gata” som möjligt och att säkerställa att så få observationer som möjligt hamnar på fel sida göras. (Géron, 2019)

A comparison of a graph

AI-generated content may be incorrect.

Figur 2 - Exempel på outliers i dataset.

SVM kan användas för både linjär och icke-linjär klassificering. (Géron, 2019)

### Beslutsträd

I Figur 3 illustreras ett beslutsträd. Ett beslutsträd har en flödesdiagrams- eller trädliknande struktur som består av noder och grenar. För att göra en klassificering börjar processen vid rotnoden och fortsätter genom noderna tills att en lövnod nås. Varje lövnod representerar en kategori. (Géron, 2019)

Sant

Falskt

Falskt

Sant

Figur 3 - Schematisk bild över ett beslutsträd.

### Random Forest

Random forest är en ensemblemetod där flera beslutsträd arbetar tillsammans för att göra en prediktion. Varje träd tränas på en delmängd av det ursprungliga datasetet. Urval för dessa delmängder görs med återläggning och samma observationer kan således återfinnas i flera delmängder. Den klass som flest träd predikterar är den klass som observationen tilldelas. (GeeksforGeeks, 2025)

### XGBoost

Extreme gradient boosting (XGBoost) är precis som Random forest en ensemblemetod som utgår från beslutsträd (GeeksforGeeks, 2025). Grunden i Boosting-modeller generellt är att olika prediktorer tränas sekventiellt, där varje prediktor alltid försöker göra en bättre prediktion än den förra (Géron, 2019). För varje nytt träd som modellen tränas på lär den sig även från de felpredikteringar som gjorde av det föregående trädet (GeeksforGeeks, 2025). Den klass som flest träd predikterar är den klass som observationen tilldelas (GeeksforGeeks, 2025).

## Utvärderingsmått

För att mäta hur väl en klassificeringsmodell presterar finns flera olika utvärderingsmått. Utvärderingsmått av intresse för detta projekt presenteras nedan.

### Confusion matrix

För att göra en *confusion matrix* beräknas antalet gånger en modell klassificerade rätt och fel för varje klass. Resultatet visas i en tabell. Se Figur 4 för hur detta ser ut för en binär klassificering. (Géron, 2019)

A diagram of positive and negative

AI-generated content may be incorrect.

Figur 4 - Confusion matrix.

### Accuracy score

*Accuracy score* är ett mått på hur stor andel av alla observationer som klassificerades rätt. Enligt Ekvation (3). (GeeksforGeeks, 2025)

( 3 )

Det bör noteras att även om *accuracy score* är högt behöver inte detta betyda att modellen presterar väl. Om klasser är mycket obalanserade, låt säga att en klass består av 95% av observationerna, kan en modell som predikterar att alla observationer tillhör den klassen alltså få en mycket hög *accuracy score*, utan att nödvändigtvis alltså vara en bra modell. (Géron, 2019)

## Hyperparameteroptimering

Hyperparametrar är inställningar för olika modeller som modellen inte kan lära sig själv, utan de måste väljas av användaren. Hyperparametrar styr hur modellen lär sig och kan vara avgörande för hur modellen presterar. Hyperparameteroptimering är processen att hitta de mest optimala inställningarna för varje modell givet det problem som finns. (GeeksforGeeks, 2025)

### GridSearch

Att välja hyperparametrar manuellt är svårt och mycket tidskrävande. GridSearch är en algoritm som på ett automatiserat sätt väljer hyperparametrar. Algoritmen utgår från en uppsättning av olika värden av hyperparametrar som ges av användaren, och testar alla möjliga kombinationer med hjälp av korsvalidering för att hitta den optimala kombinationen. (Géron, 2019)

## Dimensionsreducering - PCA

Dimensionsreducering innebär att antalet dimensioner, alltså antalet variabler, i ett dataset minskas. Detta görs framför allt för att minska träningstiden, vilket ofta är på bekostnad av en sämre prediktionsförmåga. Vid tillfällen kan dock en dimensionsreducering hjälpa till att ta bort brus i data vilket då faktiskt ger en bättre prediktionsförmåga. När en dimensionsreducering genomförs skapas nya variabler som är kombinationer av de ursprungliga variablerna. Den ursprungliga datan finns alltså inte kvar i sin ursprungliga form, utan representeras av de nya variablerna. De nya variablerna har en helt ny tolkning. (Géron, 2019)

PCA är den vanligaste förekommande modellen för att genomföra dimensionsreducering. (Géron, 2019)

# Metod

I detta projekt har en modell tränats för att kunna göra prediktioner på handskrivna siffror. Modellen har implementerats för användning i en Streamlit-app. I metoden presenteras först datasetet som modellen har tränats på under databeskrivning, sedan presenteras hur processen för att välja och träna en modell genomförts, för att avslutningsvis visa på hur Streamlit-applikationen har utvecklats.

## Databeskrivning

MNIST är ett dataset bestående av 70 000 bilder på handskrivna siffror av amerikaner. Datasetet består av 60 000 bilder för träning och 10 000 bilder för test. Datasetet skapades 1994 och har sedan dess använts flitigt för att träna och testa olika maskininlärningsmodeller.

Varje bild är 28 x 28 = 784 pixlar stor. Datasetet består alltså av 784 kolumner som varje representerar pixelintensiteten för varje pixel och varje rad är en bild av en siffra. Varje siffra i datasetet hade ursprungligen en storlek på 20 x 20 pixlar. Varje siffra har sedan placerats på en svart bakgrund som är 28 x 28 pixlar stor och centrerats efter masscentrum.

Bilderna är i gråskala. Siffrorna är skrivna med vitt på en svart bakgrund. Anti-aliasing har använts för att mjuka ut kanter på siffrorna, alltså en teknik för att blanda färger mellan kanter och bakgrund. Detta leder till att det finns gråskaliga pixlar i bilderna och inte bara svart och vit. Se exempel i Figur 5.

A collage of numbers

AI-generated content may be incorrect.

Figur 5 - Exempel på siffror från MNIST-datasetet.

## Modell

Här presenteras processen för att träna, utvärdera och välja en modell för prediktion av handskrivna siffror.

### Utforskande dataanalys

Arbetet startade med en utforskande dataanalys för att skapa en större förståelse för datasetet och hur bilderna ser ut. Detta låg till grund för beslut kring vad som behövde utföras under datapreprocessering samt vilka modeller som skulle väljas för utvärdering.

### Datapreprocessering

Under datapreprocesseringen genomfördes normalisering och skalning av data samt en dimensionsreducering.

För flera av de modeller som kommer att användas krävs en normalisering av data för bästa prestanda. För de som inte kräver det har ändå normaliserad data använts för att alla modeller ska tränas på data som ser likadan ut.

En dimensionsreducering valdes att göras dels eftersom korrelationsmatrisen visade att det fanns en korrelation mellan närliggande pixlar och en dimensionsreducering således kan hjälpa till att identifiera vilka pixlar som faktiskt bär information, dels för att snabba på modellträningen, vilket möjliggjorde att modeller som annars behövts väljas bort till följd av för lång tid för träning kunde användas.

### Modellträning

För att träna och utvärdera olika modeller har k-delad korsvalidering använts. Som utvärderingsmått har *accuracy score* använts. För att få en mer robust uppskattning av modellens prestanda har medelvärdet av *accuracy score* för varje iteration använts och den modell med högst accuracy score har ansetts vara den som bästa. Även standardavvikelsen har studerats för att säkerställa att skillnaden mellan varje iteration inte varit för stor. Den utforskande dataanalysen visade att det är en jämn fördelning av antalet nummer av varje i datasetet och därför har *accuracy score* ansetts som lämplig att använda.

För att utföra hyperparameteroptimering har GridSearch använts. Antalet möjliga kombinationer och hur många hyperparametrar som optimerats har begränsats för att minska träningstiden.

För att ytterligare snabba på modellträning har endast 50% av träningsdatan använts. Den valda modellen har sedan tränats om på hela träningsdatan innan den testats på testdatan för att säkerställa att modellen tränats på så stor del av datan som den slutligen kommer att tränas på innan den ska användas i Streamlit-applikationen.

De modeller som tränats och utvärderats är logistisk regression, SVM, Random Forest och XGBoost. Modellerna representerar olika typer av modeller och ger en bred bild över vilken som är mest lämplig för just detta syfte. Både linjära och icke-linjära beslutsgränser har därmed inkluderats.

Logistisk regression är en relativt enkel modell som om inte annat ger en baslinje för modelleringen. Modellen för logistisk regression kördes både med och utan dimensionsreducering för att se om den extra informationen var nödvändig eller snarare skapade brus.

För att möjliggöra SVM som ett alternativ för modelleringen kördes denna med dimensionsreducering för att minska träningstiden som annars blev för lång för att kunna motiveras inom tidsramen.

Random Forest och XGBoost kördes båda utan dimensionsreducering.

### Modellval

Den modell med högst *accuracy score* valdes. Denna modell tränades först om på hela träningsdatasetet för att sedan utvärderas mot testsetet.

*Accuracy score* för testdatan utvärderades mot *accuracy score* för valideringsdatan för att säkerställa att modellen presterar lika väl eller bättre på osedd data. Dessutom gjordes en *confusion matrix* för testdatan för att ytterligare säkerställa att resultatet var bra och inte endast lita på *accuracy score* som enda utvärderingsmått. För att utvärdera vart svagheter i modellen kan finnas gjordes även en *error matrix*.

Eftersom modellen ska användas i en Streamlit-applikation tränades den slutligen om på hela datasetet, alltså både träning och test.

## Streamlit-applikation

En applikation för att en användare ska kunna prediktera en siffrafrån en bild genom att antingen rita på en canvas eller ladda upp en bild skapades i Streamlit. Här presenteras processen för att genomföra bildförbehandling och prediktion av siffror.

### Bildförbehandling

En bildförbehandling genomfördes för att bilderna skulle vara i ett format som modellen kan hantera och göra så bra prediktioner på som möjligt.

Bilderna konverterades till gråskala. Siffran gjordes om till en storlek om 20 x 20 pixlar och placerades sedan på en bakgrund av storleken 28 x 28 pixlar och centrerades efter masscentrum. Pixelvärdena normaliserades på samma sätt som inför modellträningen.

När en bild laddas upp gjordes även en binarisering för att få en vit siffra på en svart bakgrund.

### Prediktion av siffra

Prediktionen genomfördes med den modell som utvärderats till att prestera bäst på MNIST-datasetet och som tränats tidigare.

# Resultat och Diskussion

## Modell

I Tabell 1 visas *accuracy score* för de olika modellerna som utvärderats.

Tabell 1 - Accuracy score för samtliga modeller.

|  |  |
| --- | --- |
| ***Accuracy score* för träningsdata för olika modeller** | |
| Logistisk regression | 0,8937 ± 0,0032 |
| Logistisk regression med dimensionsreducering | 0,9110 ± 0,0029 |
| SVM | 0,9633 ± 0,0023 |
| Random Forest | 0,9614 ± 0,0013 |
| XGBoost | 0,9686 ± 0,0026 |

XGBoost är den modell med högst accuracy score och anses därför vara den modell som presterar bäst.

Det noteras även att alla modeller har en låg standardavvikelse, vilket tyder på att resultatet för varje iteration i korsvalideringen varit lika. Standardavvikelsen har därför inte varit avgörande vid modellval.

I Tabell 2 visas *accuracy score* för den valda modellen, XGBoost, efter att den har utvärderats på testdata.

Tabell 2 - Accuracy score för den bästa modellen.

|  |  |
| --- | --- |
| ***Accuracy score* för test för vald modell** | |
| XGBoost | 0,9804 |

XGBoost-modellen presterar bättre på testdatan än valideringsdatan. Detta är väntat i och med att modellen har tränats på en större andel data. Detta indikerar även att modellen inte har överanpassats.

I Figur 6 visas confusion matrix för XGBoost på testdatan. Den visar tydligt att modellen presterar mycket väl och i de flesta fall predikteras siffror rätt.

A diagram of a confused matrix

AI-generated content may be incorrect.

Figur 6 - Confusion matrix för XGBoost.

I Figur 7 visas error matrix för XGBoost på testdatan. De största felen som görs är att 4 predikteras som 9 och 7 predikteras som 2. Den siffran som oftast blir felklassificerad är 9. Felpredikteringar är alltså att vänta till viss del i Streamlit-applikationen.

A diagram of error matrix

AI-generated content may be incorrect.

Figur 7 - Error matrix för XGBoost.

XGBoost-modellen anses prestera mycket väl, men förbättringspotential kan identifieras. Eftersom tiden för projektet var begränsad gjordes avvägningen att hålla nere antalet möjliga kombinationer för hyperparametrar. Om fler iterationer gjorts och fler hyperparametrar inkluderats hade även detta förmodligen lett till att öka modellens prediktionsförmåga.

Djupinlärningsmodeller har exkluderats i detta projekt eftersom de inte ingår i kursens omfattning, men potentiellt kan neurala nätverk ge en än bättre prestanda.

## Streamlit-applikation

I applikationen är det möjligt att prediktera siffror genom att antingen rita en siffra på en canvas eller ladda upp en bild av en siffra. Se exempel från appen i Figur 8.

A screenshot of a phone

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

Figur 8 - Exempel på prediktioner av handskrivna siffror i Streamlit-applikationen som är korrekta.

Applikationen fungerar väl och predikterar ofta siffror korrekt. Det finns dock begränsningar. Speciellt om siffror ritas snett eller inte skrivs på klassiskt amerikanskt vis misslyckas modellen med att prediktera korrekt siffra. Se exempel i Figur 9.

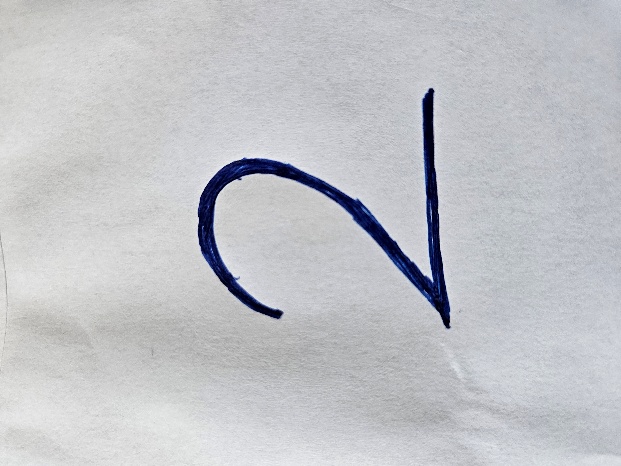
A screenshot of a phone

AI-generated content may be incorrect. A screenshot of a phone

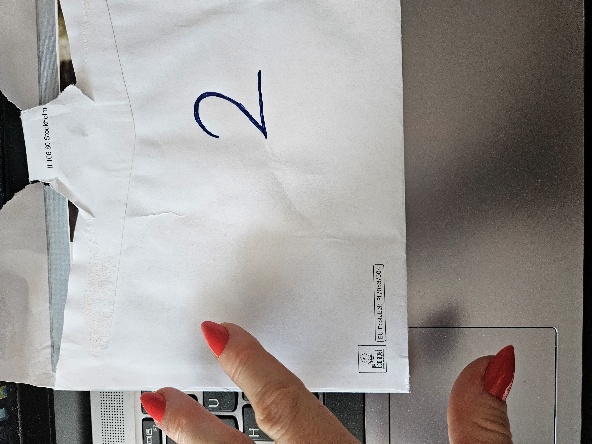
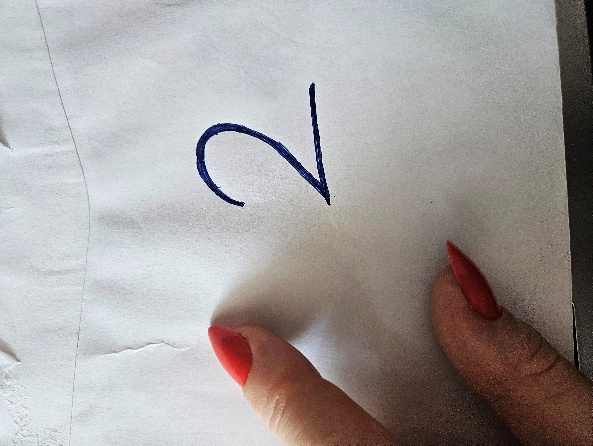
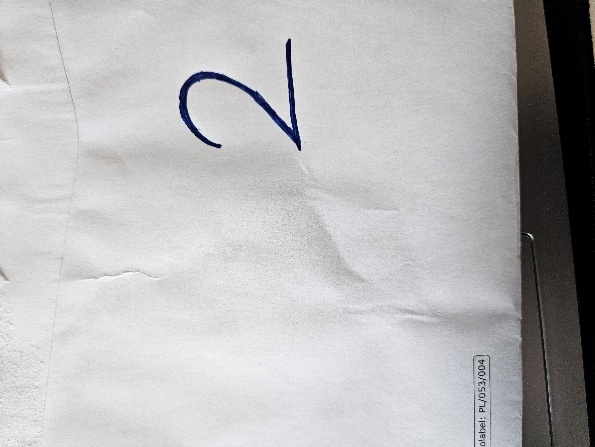
AI-generated content may be incorrect.

Figur 9 – Exempel på prediktioner av handskrivna siffror i Streamlit-applikationen som är felaktiga.

När en bild på en siffra laddas upp krävs det att siffran är tydlig och att inga andra störningar finns i bilden. Se exempel i Figur 10. När störningar finns i bilden lyckas inte modellen identifiera vad som är en siffra i bilden och misslyckas därför med att göra en prediktion. Se exempel i Figur 11. Ett försök för att få modellen att identifiera alla konturer i bilden för att sedan välja den kontur som är en siffra gjordes. Modellen lyckades väl med att identifiera konturer, men misslyckades tyvärr med att identifiera vilken av alla konturer som var en siffra. Vidare arbete för att kunna lära modellen hur siffror ska utskiljas behöver genomföras.



Figur 10 - Exempel på en bild som modellen lyckas prediktera väl.

Figur 11 - Exempel på bilder som modellen inte lyckas prediktera.

För att ytterligare förbättra modellens prediktionsförmåga identifieras dataaugmentation som en möjlighet. Dataaugmentation innebär att transformationer genomförs på befintliga bilder i datasetet, vilket leder till ett större underlag av träningsdata utan att fler bilder behöver samlas in och att fler variationer av varje siffra finns tillgänglig. Exempel på transformationer kan vara att rotera siffror, att lägga till brus och att förvränga siffror något för att simulera olika handstilar och bakgrundsvariationer. Modellen bör då kunna hantera siffror som inte skrivs på liknande sätt som i MNIST-datasetet på ett bättre sätt.

Även ytterligare förbehandling av bilder hade förmodligen kunnat öka modellens prediktionsförmåga.

# Slutsatser

I projektet har en modell som presterar mycket väl på både testdata och i Streamlit-applikationen utvecklats. Det finns dock förbättringspotential och data augmenation, bättre bildförbehandling och utvärdering av fler modeller har identifierats som möjligheter för detta.

## Frågeställningar

1. Vilken maskininlärningsmodell ger bäst noggrannhet för handskrivna siffror i MNIST-datasetet?

Efter att ha utvärderat fyra olika modeller fastställs att XGBoost är den modell som presterar bäst och fått högst *accuracy score*.

1. Vad krävs för att en modell tränad på MNIST anpassas för att tolka användarinmatade siffror i en Streamlit-applikation?

För att modellen ska kunna användas i en webapplikation krävs framför allt att en mycket god bildförbehandling för att användarens siffror ska ha ett så lika format som siffrorna i MNIST görs. Detta inkluderar bland annat omvandling till gråskala, binarisering, normalisering, storleksanpassning och centrering efter masscentrum. Ytterligare bildbehandling skulle potentiellt kunna leda till ännu bättre predikteringar än applikationen idag levererar.

# Teoretiska frågor

1. Träningsdatasetet används för att träna modellen. Det är alltså den datan som modellen tittar på för att hitta mönster och samband. Valideringsdatasetet används för att utvärdera modellens prestanda och för att finjustera hyperparametrar (OBS detta innebär inte att modellen lär sig något från valideringsdatan trots att den används för att bestämma hyperparametrar). Vi väljer den bästa modellen utifrån valideringsdatan. Testdatasetat används för att se hur väl modellen presterar på osedd data, alltså modellens generaliseringsförmåga.

Innan modellen testas med hjälp av testdatan ska den tränas om på träningsdata + valideringsdata. Innan produktionssättning tränas modellen om på all data, alltså träningsdata + valideringsdata + testdata.

Hur uppdelningen av data i tränings-, validerings- och testset görs beror dels på vilken modell som ska tränas och antalet tillgängliga observationer. En tumregel kan vara 70% träning, 15% validering, 15% test.

Modeller som presterar mycket väl på träningsdata, men sämre på valideringsdata, anses vara överanpassade. En modell som är för enkel för att fånga de underliggande mönstren i data anses vara underanpassad. Detta leder till dålig prestanda både på tränings- och valideringsdata.

1. När endast tränings- och testdata finns tillgängligt kan *k-fold cross validation* användas. Metoden går ut på att träningsdata delas upp i k lika stora delar och att modellen sedan tränas k gånger. För varje iteration används k-1 delar som träningsdata och 1 del som valideringsdata.

Metoden används med fördel när antalet observationer är litet, eftersom det kan leda till antingen för lite träningsdata och således att modeller blir dåligt tränade, eller antingen för lite valideringsdata och att det således inte finns tillräckligt med underlag för att göra ett bra modellval. Även när det finns tillräckligt med data för att dela in i tränings-, validerings- och testdata är dock *k-fold cross validation* ett bra alternativ att använda. I och med att modellen ska tränas k gånger kan det dock ta lång tid och mycket datakraft behövs, vilket blir en avvägning för om metoden bör användas eller ej. Detta blir dock ett mindre och mindre problem i takt med att datorer utvecklas.

Ofta sätts k till 5 eller 10.

1. Ett regressionsproblem är ett problem där målet är att förutsäga ett kontinuerligt/numeriskt värde baserat på ett antal variabler. Potentiella tillämpningsområden kan vara att förutspå inkomst baserat på ålder och utbildningsnivå eller förutspå pris på en bil baserat på miltal, drivmedel och märke.

Exempel på modeller:

* Linjär regression

Här beskrivs samband mellan två eller fler variabler under förutsättningen att sambandet mellan variablerna är linjärt.

* Lasso, Ridge, Elastic net

Dessa modeller bygger på/modifierar linjär regression, men har även en straffterm för att kunna antingen minska eller helt ta bort påverkan från variabler. Ridge krymper endast koefficienter, men sätter aldrig till 0, medan Lasso kan sätta helt till 0. Elastic net är en kombination av Lasso och Ridge.

* SVM

Kan hantera dels både regressions- och klassificeringsproblem samt dels både linjär och icke-linjär data. För regression: Skapar en ”gata” som bör vara så smal som möjligt, men ändå innehålla så många datapunkter som möjligt. Se figurer.

A graph of a graph with blue and red dots

AI-generated content may be incorrect.A diagram of a graph

AI-generated content may be incorrect.

* Beslutsträd

Kan hantera både regressions- och klassificeringsproblem. Ett beslutsträd har en flödesdiagramsliknande/trädliknande struktur som består av noder och grenar. Se figur. Processen börjar vid rotnoden och fortsätter genom noder tills att en lövnod nås.

Falskt

Sant

Sant

Falskt

Gren

Möjlig åtgärd

Gren

Möjlig åtgärd

* Random forest

Beslutsträd är en fundamental komponent i Random forest. Modellen bygger flera beslutsträd och varje träd tränas på en slumpmässig del av datan. Varje träd gör en prediktion och inom regression används medelvärdet för alla prediktioner till den slutliga prediktionen.

1. RMSE (Root Mean Squared Error) är ett utvärderingsmått för hur väl olika regressionsmodeller presterar. Ger medelvärdet mellan predikterade och faktiska värden i ett dataset. Ett lägre RMSE är bättre.

Residualen är i kvadrat för att ta bort negativa värden för att säkerställa så inte positiva och negativa residualer tar ut varandra i analysen.

A black square with a square and a square and a square with a square and a square with a square and a square with a square and a square with a square and a square with a square

AI-generated content may be incorrect.

1. Ett klassificeringsproblem är ett problem där målet är att förutsäga ett kategoriskt/diskret värde. Potentiella tillämpningsområden kan vara att förutspå om en person är man eller kvinna eller om en kund kommer att churna, ja eller nej.

Exempel på modeller:

* Logistisk regression

I logistisk regression estimeras sannolikheten för att en observation/datapunkt tillhör en viss klass. Om sannolikheten är högre än 50% antar modellen att den tillhör den klassen.

* SVM

Kan hantera dels både regressions- och klassificeringsproblem samt dels både linjär och icke-linjär data. För klassificering: Skapar en ”gata” som bör vara så bred som möjligt mellan olika klasser. Se figur.

A diagram of a graph

AI-generated content may be incorrect.

* Beslutsträd
* Random forest

Varje träd gör en prediktion och inom klassificering används majoriteten för alla prediktioner till den slutliga prediktionen.

Confusion matrix

Ett sätt att utvärdera hur väl en klassificeringsmodell presterar. Metoden går ut på att räkna antalet gånger ett sant värde predikteras som olika värden och representera detta i en matris. Målet är såklart att det sanna värdet och det predikterade värdet är samma, vilket representeras av diagonalen. Se figurer.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

1. K-means är en modell som används för klustring, som i sin tur är en metod inom icke-väglett lärande som syftar till att gruppera observationer i kluster baserat på mönster och strukturer i data, utan att ha förutbestämda kategorier.

Modellen fungerar så att den lokaliserar en centroid för varje kluster, varpå varje observation sedan tilldelas det kluster vars centroid är närmast. Antalet kluster anges manuellt.

Ett ofta använt tillämpningsområde är kundsegmentering. Med information om kunders ålder, lön, köpbeteende etc. är det möjligt att gruppera kunder. Troligtvis vill man hantera olika kundgrupper på olika sätt genom att anpassa produkter och marknadsföring. Unga studenter har förmodligen inte samma köpbeteende som en barnfamilj.

1. De flesta maskininlärningsmodeller kräver att datan de tränas på är numerisk (gäller för t ex linjär regression, men inte för beslutsträd och random forest). Det betyder att kategorisk data behöver omvandlas till numeriska värden för att kunna användas i modeller. Detta kan göras med Ordinal encoding, One-hot encoding och Dummy variable encoding.

Ordinal encoding används för ordinal data, alltså när kategorier har en naturlig rangordning. Exempelvis {Låg, Medium, Hög}. Varje kategori tilldelas en unik siffra, vilket gör att datorn kan tolka kategorierna och förstå rangordningen.

One-hot encoding och Dummy variable encoding används för nominal data, alltså data har specifika värden, men de saknar inbördes rangordning. Exempelvis {Man, Kvinna}, {Röd, Grön, Blå}. I One-hot encoding skapas en binär kolumn för varje kategori och värdet 1 eller 0 ges beroende på om kategorin gäller för observationen eller inte. Radsumman i varje rad blir alltid 1 eftersom endast en kategori kan vara gällande. Dummy variable encoding fungerar på liknande sätt, förutom att det skapas en kolumn mindre än antalet kategorier. När radsumman blir 0 vet vi att det är den sista kategorin som är gällande.

1. Julia har rätt. Det är alltid situationen som avgör om data är ordinal eller nominal. Data som vid en första anblick inte verkar ha en inbördes ordning kan ha det i vissa situationer, så som den som Julia beskriver.
2. Streamlit är ett ramverk för Python där Data scientists och AI/ML-engineers kan skapa appar med endast ett få antal rader kod.

# Självutvärdering

1. Utmaningar du haft under arbetet samt hur du hanterat dem.   
   Att göra ett arbete som passar inom tidsramen. Jag tycker att jag har anpassat mycket väl vad som har inkluderats och inte, samt redogjort för ytterligare möjligheter för att förbättra prediktionsförmåga som kan ses som vidare arbete som hade genomförts om mer tid fanns.

Jag önskar att jag hade lyckats med att identifiera siffran i uppladdade bilder när andra störningar fanns i bilden. Men jag är övertygad om att detta återkommer senare i utbildningen och att jag får chansen att utveckla denna förmåga.

1. Vilket betyg du anser att du skall ha och varför.

VG. Jag anser att jag implementerat modeller och metoder från maskininlärning och att jag kan redogöra för modellval, modellanpassning och modellutvärdering.

1. Något du vill lyfta fram till Antonio?

Nej, inget speciellt.

# Appendix A

Länk till Github: https://github.com/terese-eriksson/GitHub\_MachineLearning\_Terese.git

# Källförteckning

Geeks for Geeks (2025). *Getting started with Classification*. https://www.geeksforgeeks.org/getting-started-with-classification/ (Hämtad 2025-03-21).

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 2. uppl. O’Reilly Media.

Geeks for Geeks (2025). *Logistic Regression in Machine Learning*. https://www.geeksforgeeks.org/understanding-logistic-regression/ (Hämtad 2025-03-21).

Geeks for Geeks (2025). *Random Forest Algorithm in Machine Learning*. https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/ (Hämtad 2025-03-21).

Geeks for Geeks (2025). *XGBoost*. https://www.geeksforgeeks.org/xgboost/ (Hämtad 2025-03-21).

Geeks for Geeks (2025). *Understanding the Confusion Matrix in Machine Learning*. https://www.geeksforgeeks.org/confusion-matrix-machine-learning/?ref=header\_outind (Hämtad 2025-03-21).

Geeks for Geeks (2025). *Hyperparameter tuning*. https://www.geeksforgeeks.org/hyperparameter-tuning/?ref=header\_outind (Hämtad 2025-03-21).