Kunskapskontroll 2

Machine Learning

Lisa Påhlsson

EC Utbildning

Kurs: Machine Learning

2025-03

Abstract

This study investigates and compares the performance of three machine learning models (Logistic Regression, Random Forest, and MLPClassifier) on the MNIST dataset, a benchmark for handwritten digit recognition. The aim is to identify the model that exhibits the best performance in terms of training time and evaluation metrics such as accuracy, precision, and recall. Results demonstrate that MLPClassifier outperforms the other models, achieving the highest accuracy and precision while requiring the shortest training time. Logistic Regression, on the other hand, shows the lowest performance, indicating limitations in capturing complex patterns within the dataset. This research highlights the importance of model selection and the impact of model complexity on both training efficiency and classification accuracy in image recognition tasks.

Skapas automatiskt i Word genom att gå till Referenser > Innehållsförteckning.

Innehåll

1 Inledning

I en digitaliserad värld genereras enorma mängder data varje dag. Förmågan att extrahera meningsfull information och bygga prediktiva modeller har blivit både en utmaning och en möjlighet inom många områden. Maskininlärning spelar en central roll i denna utveckling genom att låta datorer lära sig från data istället för att enbart följa fördefinierade regler. Det möjliggör system som kan identifiera mönster, göra prediktioner och fatta beslut med hög precision

Ett brett användningsområde för maskininlärning är bildigenkänning, där en vanlig uppgift är bildklassificering vilket innebär att automatiskt tilldela en bild en korrekt etikett baserat på dess innehåll. För att främja forskning och utveckling inom detta område har flera standardiserade dataset utvecklats. Ett av de är MNIST databas.

MNIST (Modified National Institute of Standards and Technology) är en datamängd som består av tusentals handskrivna siffror från 0 till 9. På grund av sin relativt enkla struktur men ändå utmanande natur har MNIST blivit ett standardriktmärke för att testa och jämföra olika maskininlärningsalgoritmer, särskilt inom området för djupinlärning och neurala nätverk. Att framgångsrikt kunna klassificera siffror i MNIST-datamängden är ett viktigt steg i förståelsen och utvecklingen av mer komplexa bildigenkänningssystem (Wikipedia, 2024).

Denna rapport undersöker hur olika maskininlärningsmodeller presterar på MNIST-datamängden. Målet är att identifiera vilken av de tränade modellerna som uppvisar bäst resultat baserat på utvärderingsmått såsom noggrannhet och klassificeringsprestanda. För att uppnå detta syfte undersöker rapporten följande frågeställningar:

- Hur skiljer sig prestandan mellan logistisk regression, Random Forest och MLPClassifier på MNIST-datamängden med avseende på träningstid och utvärderingsmått såsom accuracy, precision och recall?
- Vilken av de utvärderade modellerna uppvisar bäst prestanda på MNIST-datamängden baserat på träningstid och utvärderingsmått såsom accuracy, precision och recall?

2 Teori

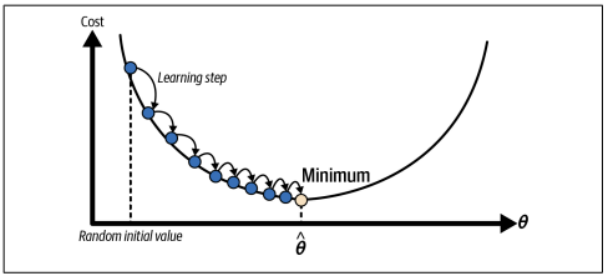
2.1 Klassificering

Inom maskininlärning innebär ett klassificeringsproblem att man med hjälp av algoritmer försöker förutsäga vilken kategori eller klass en observation tillhör baserat på insamlad data. Den beroende variabeln är kategorisk och kan anta två klasser (binär klassificering) eller fler (multiklassklassificering). Klassificeringsmodeller används inom en rad olika områden, såsom bildigenkänning, spamfiltrering och medicinsk diagnos. Exempel på vanliga klassificeringsmodeller är logistisk regression, beslutsträd, stödvektormaskiner (SVM) och neurala nätverk.

2.1.1 Logistisk regression

Logistisk regression är en algoritm som används för klassificering. Trots sitt namn är det en klassificeringsmodell. Det innebär att den uppskattar sannolikheten att en instans tillhör en viss klass eller ej, klass 0 eller klass 1.

Modellen tränas genom att minimera loggförlusten (log loss) med hjälp av optimeringsalgoritmer som exempelvis Gradient Descent, se figur 1.



Figur 1: Gradient Descent *Illustrationen visar hur optimeringsalgoritmen Gradient Descent fungerar för att minimera kostnadsfunktionen. De blå punkterna representerar stegen som modellen tar för att justera parametrarna (ϑ), medan pilarnas riktning visar hur algoritmen rör sig mot det lägsta värdet av kostnaden (minimum). Genom att iterera på detta sätt kan modellen lära sig optimal parametrar för att minimera loggförlusten (log loss) (Géron, 2023, p.139)*

När modellen har uppskattat sannolikheten (p) för att en instans tillhör den positiva klassen, görs prediktionen :

- Om $p < 0.5$, predikteras klass 0.
- Om $p \geq 0.5$, predikteras klass 1.

2.1.1.1 Praktiska tillämpningar

Logistiks regression är användbart inom många områden där klassificeringsproblem uppstår så som exempelvis medicinsks diagnostik, marknadsföring, finanssektorn och bildbehandling.

2.1.1.2 För- och nackdelar

En av fördelarna med logistisk regression är dess enkelhet och snabbhet att träna, även på stora dataset. Den är också lätt att tolka, vilket gör den användbar i situationer där det är viktigt att förstå hur variabler påverkar resultatet. Dessutom fungerar den relativt bra på linjära problem och erbjuder effektiva sannolikhetsförutsägelser.

Nackdelen är dock att logistisk regression är en linjär modell vilket innebär att den har begränsad förmåga att fånga komplexa eller icke-linjära samband i data. Den kan därför prestera sämre jämfört med mer avancerade modeller som neurala nätverk eller Random Forest när data är mycket komplex eller innehåller många interaktioner mellan variabler.

2.1.1.3 Regularisering

Regularisering inom logistisk regression är en metod för att minska risken för överanpassning, vilket inträffar när modellen anpassar sig alltför väl till träningsdata och presterar dåligt på ny data. Genom att införa begränsningar på modellens komplexitet kan regularisering hjälpa modellen att generalisera bättre och undvika hög varians, som är ett kännetecken för överanpassning (Géron, 2023, p.32)

2.1.2 Random Forrest

Random Forest är en så kallad ensemble-inlärningsmetod som bygger på att kombinera flera olika beslutsträd. Ensemble-inlärning innebär att kombinera flera svaga modeller (som enskilda beslutsträd) för att skapa en starkare och mer robust modell.

Det är en kraftfull och robust inlärningsalgoritm som bygger på ensembler av slumpmässigt genererade beslutsträd för att uppnå bättre prediktionsförmåga och stabilitet. Den fungerar både vid regressions- och klassificeringsproblem. Vid klassificeringsproblem röstar de enskilda träden och den slutliga prediktionen avgörs genom ett majoritetsbeslut.

2.1.2.1 Praktiska tillämpningar

Random Forest är även den användbar inom många olika områden där klassificeringsproblem uppstår så som till exempel att förutspå sjukdomar baserat på patientdata eller marknadsföring för att avgöra vilka kunder som kommer att köpa en viss produkt.

2.1.2.2 För- och nackdelar

En stor fördel med Random Forest är att den är robust mot överanpassning tack vara att den använder boostrap sampling och feature randomness för att skapa variation mellan träden.

Random Forest har en bra förmåga att hantera både numeriska och kategoriska data samt att fånga icke-linjära relationer, den ger också ett mått på feature importance vilket innebär att den ger mått på vilka features som är viktiga för inlärningen.

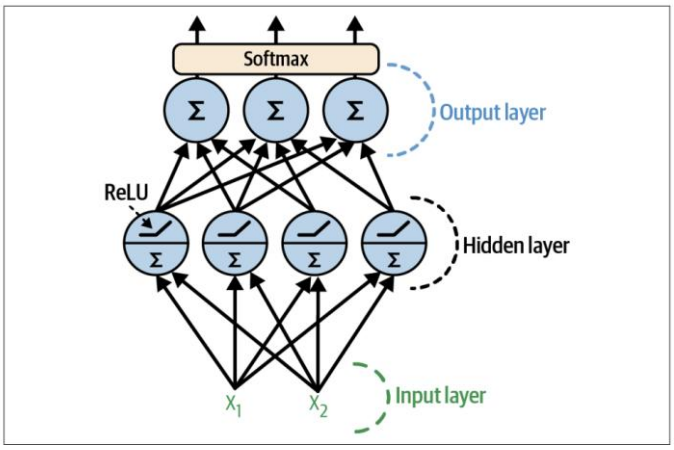
En nackdel kan vara att den kan vara svårare att tolka än ett enskilt beslutsträd eller logistisk regression. Random Forest kan även bli långsam när det är många träd och/eller mycket data.

2.1.2.3 Hyperparametrar

Viktiga Hyperparametrar att justera är exempelvis som antalet träd (n_estimators) och maximalt djup på träden (max_depth) vilket ger en större robusthet till modellen. Även "min_samples_split" (minsta antal observationer för att dela en nod) eller "min_samples_leaf" (minsta antal observationer i ett löv).

2.1.3 Multi-layer Perceptron Classifier (MLPClassifier)

MLPClassifier är en typ av artificiellt neuralt nätverk som används för klassificeringsproblem. Det är en modell inom djupinlärning som består av flera lager av artificiella neuroner: ett ingångslager, ett eller flera dolda lager och ett utgångslager. Varje neuron i ett lager är kopplad till neuroner i nästa lager och använder en aktiveringsfunktion för att lära sig komplexa samband i data. MLP är särskilt användbart när data innehåller icke-linjära relationer som enklare modeller inte kan fånga.



Figur 2: MLP-uppbyggnad *Skiss över hur en MLP-träning fungerar (Géron, 2023, p.316)*

2.1.3.1 Praktiska tillämpningar

MLP används inom många olika områden, särskilt när uppgifterna är komplexa och kräver avancerad mönsterigenkänning. Exempel kan vara bild- och textanalys.

2.1.3.2 För- och nackdelar

En stor fördel med MLP är dess förmåga att modellera mycket komplexa icke-linjära relationer i data, vilket ofta leder till hög noggrannhet på uppgifter som kräver avancerad mönsterigenkänning, såsom bildklassificering och naturlig språkbehandling. Modellen är även flexibel och kan användas för både strukturerade och ostrukturerade data.

En nackdel är att träningen kan vara beräkningsmässigt krävande, särskilt för stora dataset eller nätverk med många lager och neuroner. Det finns också en risk för överanpassning men detta kan hanteras med tekniker som regularisering och dropout. Dessutom kan modellen vara svår att tolka, eftersom den fungerar som en "svart låda".

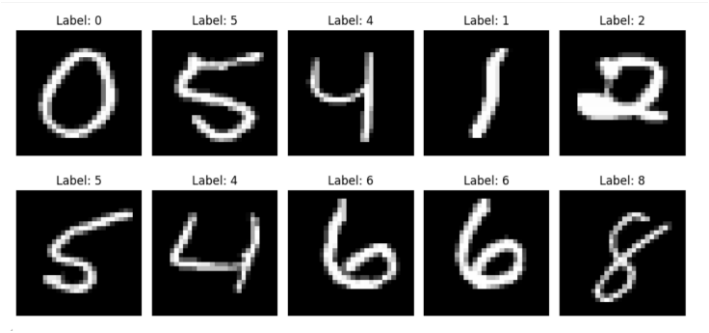
2.1.3.2.1 Hyperparametrar

Viktiga hyperparametrar att hålla koll på är bland annat antal lager, antal neuroner per lager, aktiveringsfunktioner, inlärningshastighet (learning_rate).

3 Metod

Data för detta arbete hämtades från MNIST-databasen. MNIST (Modified National Institute of Standards and Technology) är en väletablerad datamängd bestående av tusentals gråskalebilder av handskrivna siffror (0 till 9), där varje bild har en upplösning på 28x28 pixlar.

Experimenten genomfördes i Python med hjälp av Scikit-learn, ett omfattande bibliotek för maskininlärning som tillhandahåller verktyg för bland annat databearbetning, modellimplementering och utvärdering. Initialt delades datamängden upp i en träningsmängd och en testmängd. För att möjliggöra hyperparameteroptimering och val av modell delades träningsmängden ytterligare upp i en separat tränings- och valideringsmängd med hjälp av funktionen train_test_split i Scikit-learn



Figur 3: Visualisering av hur datasetet ser ut,

Figur 4: Visualisering av fördelningen av klasserna för att se att det finns en jämn fördelning eller om det behövs justeras något.

Innan modellträningen påbörjades genomfördes en liten dataanalys för att säkerställa datamängdens kvalitet och balans mellan de olika klasserna. Därefter tränades de tre valda maskininlärningsmetoderna (Logistisk Regression, Random Forest och MLPClassifier) på träningsdatan.

Pixelvärdena i MNIST-datasetet, som ursprungligen sträcker sig från 0 till 255, normaliserades genom att dividera varje värde med 255. Denna process skapar en skala där alla värden ligger mellan 0 och 1, vilket underlättar modellens träningsprocess.

En betydande del av arbetet ägnades åt att undersöka effekten av olika hyperparameterinställningar för att optimera modellernas prestanda. Bland annat utvärderades olika parameterkombinationer systematiskt med hjälp av GridSearchCV. Trots noggrann hyperparameteroptimering resulterade justeringarna endast i små prestandaförbättringar i modellernas prestanda. Även en Support Vector Machine (SVM) undersöktes initialt men på grund av beräkningsmässiga utmaningar och tidsbegränsningar valdes det att fokusera på de tre tidigare nämnda metoderna.

Med intresset att även undersöka potentialen hos neurala nätverk inkluderades en enklare variant i form av en Multi-layer Perceptron (MLP) i studien.

Då de ytterligare justeringarna av hyperparametrar inte resulterade i signifikanta prestandaförbättringar beslutades att använda relativt enkla konfigurationer för de slutgiltiga modellerna.

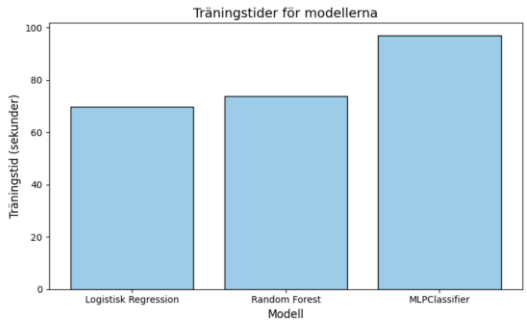
4 Resultat och Diskussion

Under träningsprocessen stötte jag på specifika utmaningar som krävde justeringar av modellinställningarna. För Logistisk Regression observerades konvergensproblem med optimeringsalgoritmen. Trots försök att öka antalet iterationer uppnådde modellen inte konvergens. Som en alternativ lösning implementerades solver="liblinear", även om denna solver är mer anpassad för mindre datamängder. Detta resulterade i förbättrad konvergens. Bytet av solver påverkade dock inte de slutliga utvärderingsresultaten signifikant vilket tyder på att de grundläggande begränsningarna för Logistisk Regression kvarstod.

För Random Forest-modellen fastställdes att n_estimators=200 gav den bästa prestandan efter initiala experiment. Detta värde användes därför konsekvent under träningen.

Gällande MLPClassifier valdes max_iter=300 för att säkerställa att träningen fortsatte tillräckligt länge för att uppnå konvergens. Värt att notera är att MLPClassifier har en inbyggd funktion för tidig avstängning (early stopping) som avbryter träningen när modellen inte längre förbättras på valideringsdatan, vilket säkerställer att träningen inte fortsätter i onödan..

Dessa observationer belyser vikten av att experimentera med olika optimeringsalgoritmer och hyperparametrar för att hantera specifika utmaningar, såsom konvergensproblem.



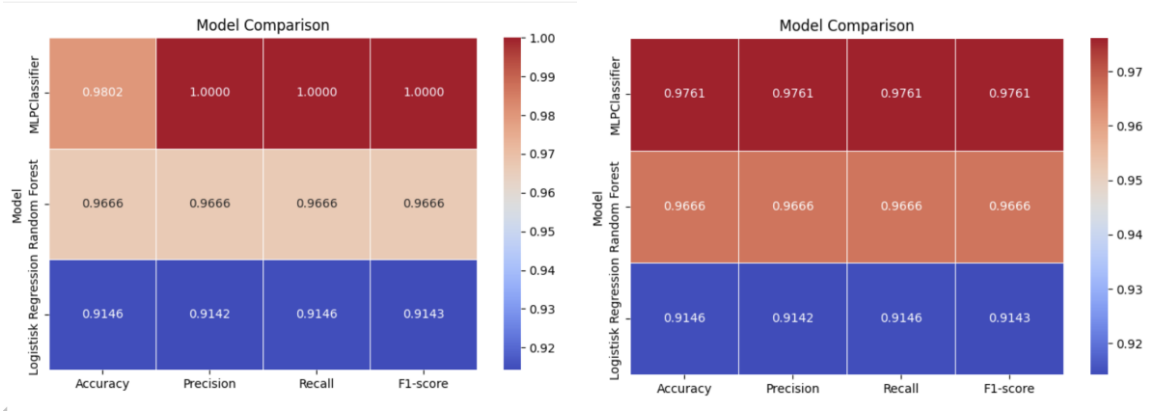
Figur 5: Figuren ger en översikt över hur skillnad hur lång tid varje modell tar att tränas

Diagrammet visar träningstider (i sekunder) för tre olika modeller: Logistisk Regression, Random Forest och MLPClassifier.

- Logistisk Regression har en träningstid på cirka 70 sekunder (69.8 sekunder).
- Random Forest har en något längre träningstid på cirka 74 sekunder (73.7 sekunder).
- MLPClassifier har den längsta träningstiden av de tre modellerna, med cirka 97 sekunder (96.9 sekunder).

Till skillnad från tidigare observationer i början av testningen visar nu resultaten att MLPClassifier har den längsta träningstiden. Detta kan bero på modellens komplexitet och antalet parametrar som behöver optimeras. Träningstiden för Logistisk Regression är nu kortare än både Random Forest och MLPClassifier.

Sammanfattningsvis visar resultatet att träningshastighet inte alltid korrelerar direkt med modellens upplevda komplexitet. Faktorer som optimeringsalgoritmer, hyperparametrar och modellarkitektur kan ha en betydande inverkan på träningstiden.



Figur 6: Skillnad i heatmap beroende på inställning, visar resultaten mellan de olika modellernas prestanda.

Figur 6 visar en jämförelse av prestandan hos de tre utvärderade modellerna (MLPClassifier, Random Forest och Logistisk Regression) på MNIST-datamängden. De utvärderingsmått som presenteras är accuracy, precision, recall. och F1-poäng. Varmare färger i diagrammet indikerar högre prestanda.

Överlag presterade MLPClassifier bäst på samtliga utvärderingsmått med en noggrannhet på 0.9802 och perfekta poäng (1.00) för precision, återkallelse och F1-score. 1.00 kan indikera perfektion i modellen men även potentiell överanpassning. Random Forest visade också god prestanda med en noggrannhet på 0.9666 och liknande värden för de andra måtten. Logistisk Regression hade den lägsta prestandan av de tre modellerna, med en noggrannhet på 0.9146.

För säkerhets skull ändrade jag MLPClassifier efter detta resultat genom att ändra learning_rate_init=0.1 för att styra hur stora steg modellen tar vid varje uppdatering av vikterna under träningen. En högre inlärningsfaktor kan göra att modellen lär sig snabbare men riskerar att missa den optimala lösningen, medan en lägre gör inlärningen stabilare men långsammare.

Resultaten visar att den mer komplexa modellen, MLPClassifier var bäst lämpad för att extrahera de komplexa mönstren i MNIST-datamängden för att korrekt klassificera handskrivna siffror. Random Forestpresterade också betydligt bättre än den linjära modellen Logistisk Regression.

5 Slutsatser

Denna studie syftade till att utvärdera och jämföra prestandan hos tre olika maskininlärningsmodeller, Logistisk Regression, Random Forest och MLPClassifier, på MNIST-datamängden. Resultaten visade skillnader i både träningstid och utvärderingsmått mellan modellerna.

5.1 Prestandaskillnader

- **Träningstid:** Som förväntat hade MLPClassifier den längsta träningstiden, vilket bekräftar antagandet att mer komplexa modeller kräver mer beräkningsresurser och tid för träning. Logistisk Regression visade sig ha den kortaste träningstiden.

- **Utvärderingsmått**: MLPClassifier uppvisade den högsta prestandan i samtliga utvärderingsmått (inklusive accuracy, precision, recall och F1-score). Random Forest presterade också väl, men något sämre än MLPClassifier. Logistisk Regression hade den lägsta prestandan, vilket tyder på att linjära modeller har svårare att fånga komplexa mönster i MNIST-datamängden.

5.2 Bästa prestanda

Baserat på både träningstid och utvärderingsmått uppvisade MLPClassifier den bästa övergripande prestandan på MNIST-datamängden. Detta tyder på att för komplexa uppgifter som bildigenkänning kan komplexare modeller som MLPClassifier vara mer lämpliga.

5.3 Resultatets betydelse

Resultaten från denna studie ger värdefulla insikter i valet av maskininlärningsmodell för bildigenkänningsuppgifter. De visar att komplexiteten i modellen har en direkt inverkan på både träningstid och prestanda. För uppgifter som kräver hög noggrannhet kan det vara värt att investera i mer komplexa modeller, även om de kräver mer beräkningsresurser.

6 Teoretiska frågor

1. *Kalle delar upp sin data i ”Träning”, ”Validering” och ”Test”, vad används respektive del för?*

   Träning – På denna data tränar man modellen
   Validering – Valideringsdatan används för att validera det man har tränat och kan hjälpa till att finjustera. Innan man kör modellen på testdatan slår man ihop träning och validering och tränar om modellen.
   Test – Det slutliga testet, om modellen håller eller inte.

2. *Julia delar upp sin data i träning och test. På träningsdatan så tränar hon tre modeller; ”Linjär Regression”, ”Lasso regression” och en ”Random Forest modell”. Hur skall hon välja vilken av de tre modellerna hon skall fortsätta använda när hon inte skapat ett explicit ”valideringsdataset”?*

   Hon kan använda sig av Cross-validation för att testa sin data.

3. *Vad är ”regressionsproblem? Kan du ge några exempel på modeller som används och potentiella tillämpningsområden?*

   Ett regressionproblem är när den beroende variabeln (y) har kontinuerliga värden. Några modeller som kan använda är linjär regression, beslutsträd, random forrest regression och SVM (Support Vector Machines). Kan användas för att förutspå priset på ett hus eller analysera vilka aktier som kommer gå bra.

4. *Hur kan du tolka RMSE och vad används det till:*

$$RMSE = \sqrt{\sum_i (y_i - \hat{y}_i)^2}$$

   RMSE är ett mått som används för att utvärdera prestandan hos en modell, främst i regressionsmodellerna. Den mäter den genomsnittligs skillnaden mellan de prediktion från en modell och de faktiska värdena i datasetet

5. *Vad är ”klassificieringsproblem? Kan du ge några exempel på modeller som används och potentiella tillämpningsområden? Vad är en ”Confusion Matrix”?*

   Ett klassificeringsproblem är när den beroende variabeln kan anta två klasser. Några klassificeringsmodeller att använda är logistisk regression, beslutsträd, Random Forest Classification och SVM (Support Vector Machines). Det kan exempelvis användas till Spamfilter för att avgöra om mailet är spam eller inte, men även bildigenkänning. En Confusion Matrix är ett verktyg som används för att utvärdera prestandan hos en klassificeringsmodell.

6. *Vad är K-means modellen för något? Ge ett exempel på vad det kan tillämpas på.*

   K-means är en form av unsupervised learning algoritm som man kan använda för att klustra data i ett antal förutbestämda grupper.

7. *Förklara (gärna med ett exempel): Ordinal encoding, one-hot encoding, dummy variable encoding. Se mappen "l8" på GitHub om du behöver repetition.*

   Ordinal encoding omvandlar kategoriska värden till heltal. Röd, Gul, Grön -> 1,2,3
   One-Hot encoring skapar en binär kolumn för varje kategori

Färg	*Röd*	*Grön*	*Blå*
Röd	*1*	*0*	*0*
Grön	*0*	*1*	*0*
Blå	*0*	*0*	*1*

   Dummy variable encoding, lite samma som ovan men lite mer effektiv då de tar bort en kolumn.

Färg	*D1*	*D2*
Röd	*1*	*0*
Grön	*0*	*1*
Blå	*0*	*0*

8. *Göran påstår att datan antingen är "ordinal" eller "nominal". Julia säger att detta måste tolkas. Hon ger ett exempel med att färger såsom {röd, grön, blå} generellt sett inte har någon inbördes ordning (nominal) men om du har en röd skjorta så är du vackrast på festen (ordinal) – vem har rätt?*

   Jag tänker att båda har rätt på olika sätt. Data i sig är nominal men beroende på kontext kan den ordinal om den används för att tolka något som ger den en rangordning.

9. *Vad är Streamlit för något och vad kan det användas till?*

   Streamlit är en open source källa för att kunna göra dataapplikationer för att visa, analysera eller visualisera data.

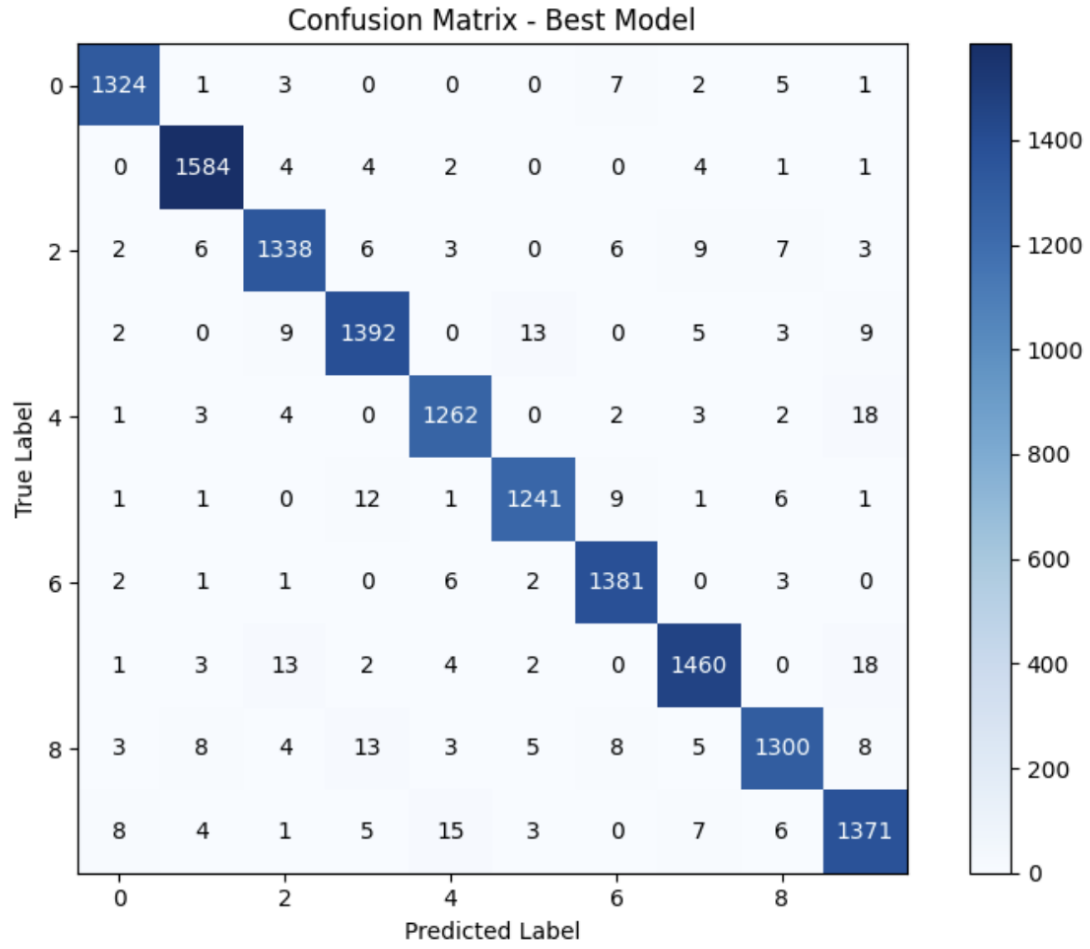
7 Självutvärdering

1. **Utmaningar du haft under arbetet samt hur du hanterat dem.**
   Tiden! Behöver lägga mycket av min tid på dottern vilket gör att jag inte hinner sitta i den takt jag tycker jag behöver för inlärandet. Samt lite motivation men det är nog kopplat till tiden, att inte hinna lägga den tid och energi som man vill göra. Vilket jag inte hanterat bra då jag uppenbarligen inte har anpassat tiden till utförandet med att lägga mer tid än jag kanske borde.

2. **Vilket betyg du anser att du skall ha och varför.**
   Inte min starkaste gren detta att anse. Tänker att jag gav mig på att göra en streamlitapp, men det tog väldigt mycket tid vilket gjorde att jag fick lägga mindre tid på rapporten . Delen ”Redogöra för och kritiskt diskutera modellval, modellanpassning och modellutvärdering” är väl där jag faller då kanske då jag inte hann lägga tillräckligt mycket krut på det, inser också att jag missade att diskutera VG-delen och hur man förbehandlat bilder mm men ett G bör jag ha om jag inte missuppfattat nått någonstans.

3. **Något du vill lyfta fram till Antonio?**
   Uppskattar verkligen lektionerna och arbetet under dessa dagar, det har dalat lite på senaste så var trevligt att det fick att uppsving i denna kurs.

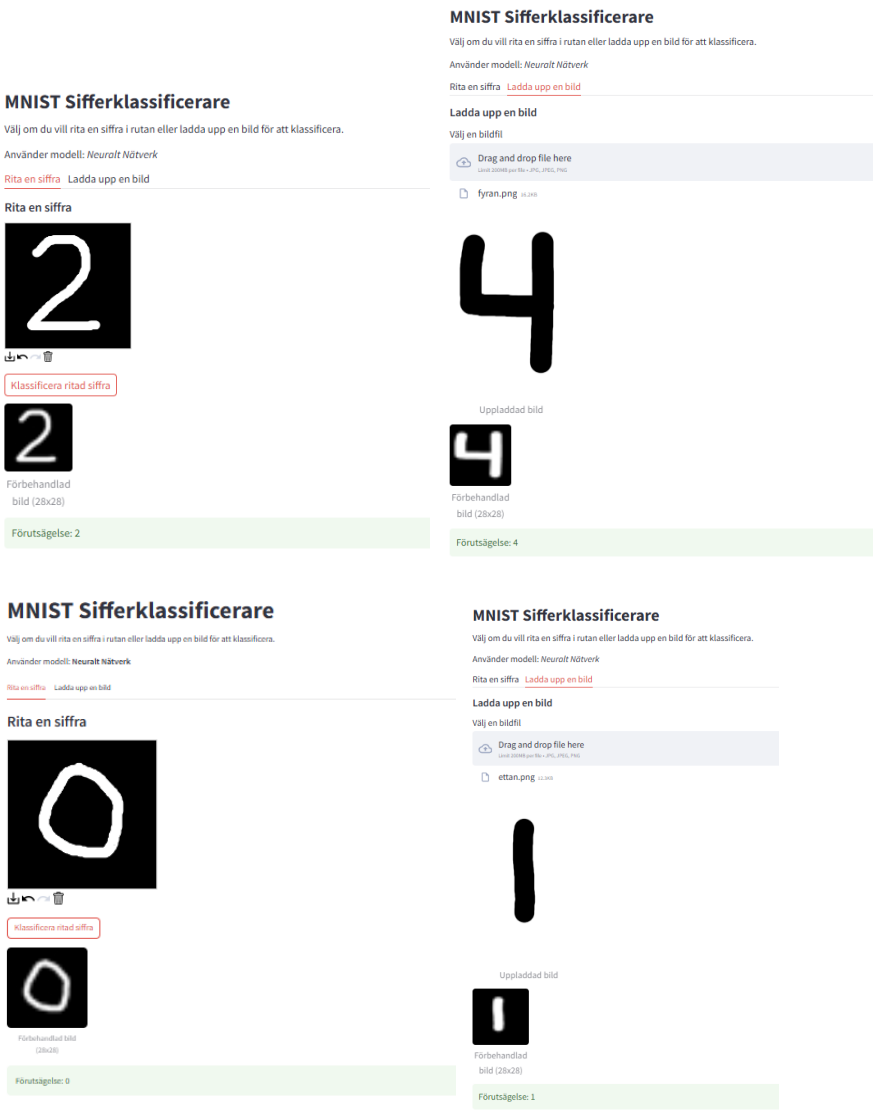
Appendix A



Klassifikationsrapport (Bästa modell efter omträning):

	precision	recall	f1-score	support
0	0.99	0.99	0.99	1343
1	0.98	0.99	0.99	1600
2	0.97	0.97	0.97	1380
3	0.97	0.97	0.97	1433
4	0.97	0.97	0.97	1295
5	0.98	0.97	0.98	1273
6	0.98	0.99	0.98	1396
7	0.98	0.97	0.97	1503
8	0.98	0.96	0.97	1357
9	0.96	0.97	0.96	1420
accuracy			0.98	14000
macro avg	0.98	0.98	0.98	14000
weighted avg	0.98	0.98	0.98	14000

Figur7: Resultat över bästa modellen



Figur 8: Skärmdumpar från streamlitappen

Källförteckning

Wikipedia (11 december 2024). *MNIST database*
https://en.wikipedia.org/wiki/MNIST_database

Géron, A. (2023). *Hands-On Machine Learning with Scikit-Learn & TensorFlow: Concepts, Tools, and Techniques*. O'Reilly Media.