

Kunskapskontroll 2 - Maskininlärning



André Lindeberg

EC Utbildning

2025-03-21

Abstract

MNIST data set is known worldwide among researchers and students within machine learning. This report investigates how an algorithm can be trained on MNIST and later applied to a Streamlit application where it will be used to predict handwritten numbers on a canvas. Three models used for classification problem, RandomForestClassifier, XGBClassifier och Support Vector Machines (SVM), were evaluated where XGBClassifier showed the best result. Through GridSearchCv, hyperparameters were tuned and the model achieved an accuracy of 98,05% on validation set and 97,77% on the test set. The web application created with Streamlit lets a user draw numbers and the model will predict/classify the number in real-time. The result would show the model would predict all numbers during testing correctly with three different users.

Innehållsförteckning

1	Inledning.....	1
2	Teori.....	2
2.1	MNIST Dataset	2
2.2	RandomForestClassifier	2
2.3	XGBClassifier	2
2.4	Support Vector Machines (SVM).....	2
2.5	Hyperparametrar och GridSearch.....	3
2.6	Utvärdering av modeller	3
3	Implementation.....	3
3.1	Streamlit.....	3
3.2	XGBClassifier	3
4	Metod	4
4.1	Datainsamling	4
4.2	Uppdelning av data	4
4.3	Modellering.....	4
4.4	Kod i Streamlit.....	5
5	Resultat och Diskussion	5
6	Slutsatser	7
7	Teoretiska frågor	8
8	Självutvärdering.....	10
	Appendix A	11
	Källförteckning.....	14

1 Inledning

Syftet med denna rapport är att undersöka hur en maskininlärningsmodell kan tränas i data setet MNIST och appliceras i en Streamlit-applikation där modellen används för att prediktera ritade siffror av en användare.

För att uppfylla syftet så kommer följande frågeställningar att besvaras:

1. *Hur kan kod utformas för att välja den bäst presterande modellen?*
2. *Hur kan kod utformas för att skapa en applikation i Streamlit med syfte att prediktera handskrivna siffror skrivna på canvas i applikationen?*
3. *Kan modellen i Streamlit-applikation prediktera alla siffror (0–9) rätt när en användare ritat siffrorna i canvasen i applikationen under test?*

2 Teori

Under detta kapitel presenteras den teorin bakom de metoder och algoritmer som valts att användas i detta projekt. MNIST Dataset som används för träning, validering och test av modellerna presenteras samt mått/metrics som används för att utvärdera modellen.

2.1 MNIST Dataset

MNIST (Modified National Institute of Standards and Technology) är ett dataset som består av 70 000 små bilder av handskrivna siffror. Dessa bilder är skapade av gymnasieelever och anställda vid US Census Bureau. Varje bild är märkt med den siffra den representerar.

MNIST betraktas ofta som "hello world" inom maskininlärning eftersom det är ett vanligt dataset för att testa och jämföra nya klassificeringsalgoritmer. Nästan alla som lär sig maskininlärning kommer i kontakt med detta dataset ("MNIST Database", 2024, s. 85).

2.2 RandomForestClassifier

Är en ensemblemodell som bygger flera slumpmässiga beslutsträd där varje träd tränas på en slumpmässig delmängd av träningsdatan. Det gör att modellen är motståndskraftig mot överanpassning. Modellen lär sig vilka features som är viktiga. Den hanterar både kategoriska och numeriska värden. Den använder sig av Bagging eller Pasting där Bagging betyder slumpmässigt urval med återläggning och Pasting slumpmässigt urval utan återläggning (Géron, 2019).

2.3 XGBClassifier

XGBClassifier är en gradient boosting-algoritm som används för klassificeringsproblem. Gradient Boosting är en ensemblemetod som bygger en modell genom att kombinera flera svaga modeller (ofta beslutsträd) i en sekventiell process. Varje nytt träd försöker korrigera felen som gjordes av de tidigare träden. XGBoost är ofta en viktig komponent i vinnande bidrag i maskininlärningstävlingar (Géron, 2019).

2.4 Support Vector Machines (SVM)

Support Vector Machines (SVM) är en kraftfull klassificeringsalgoritm som används för både linjära och icke-linjära klassificeringsproblem. SVM försöker hitta den optimala hyperplanet som separerar

klasserna i data genom att maximera marginalen mellan datapunkterna och separationslinjen. För icke-linjära problem används kärnmetoder (kernel tricks) såsom RBF (Radial Basis Function) för att mappa data till en högre dimension där det blir linjärt separerbart. SVM är särskilt effektiv för små till medelstora dataset med hög dimension (Géron, 2019).

2.5 Hyperparametrar och GridSearch

I syfte att förbättra en modells prestanda justeras dess hyperparametrar. Hyperparametrar bestämmer hur en modell lär sig till skillnad från parametrar som visar vad modellen lärt sig. GridSearchCV används ofta för att optimera en modellens hyperparametrar genom att testa olika kombinationer av hyperparametrar genom cross-validation (cv) för att hitta den bästa inställningen (Prgomet, 2025).

2.6 Utvärdering av modeller

För att jämföra och utvärdera vilken modell som presterar bäst används *accuracy* och *Confusion Matrix* som utvärderingsmått. *Accuracy* är andelen av rätt klassificeringar. *Confusion Matrix* visar hur många rätt/fel klassificeringar modellen gör (Prgomet, 2025).

3 Implementation

3.1 Streamlit

Streamlit är ett Python-ramverk med öppen källkod för Data Scientist som arbetar inom maskininlärning (ML) för att leverera applikationer inom just ML utan att behöva kunna webbutveckling (*Streamlit Docs*, u.å.). Streamlit-applikationen startas genom att högerklicka på den relevanta filen i VS-Code, vilket i detta fall är *St_app.py*. Därefter väljer användaren "Open in Integrated Terminal" och skriver *streamlit run St_app.py*. Detta öppnar applikationen i webbläsaren där användaren kan interagera med modellen.

3.2 XGBClassifier

XGBClassifier valdes eftersom den presterade bäst på valideringsdatan.

4 Metod

4.1 Datainsamling

Data laddades från förutbestämd kod in i VS Code. Data setet MNIST.

```
mnist = fetch_openml('mnist_784', version=1, cache=True, as_frame=False)

X = mnist["data"]
y = mnist["target"].astype(np.uint8)
```

4.2 Uppdelning av data

Uppdelningen av data skedde enligt följande kod:

```
X_train_val, X_test, y_train_val, y_test = train_test_split(
    X, y, test_size=10000, random_state=42)

X_train, X_val, y_train, y_val = train_test_split(
    X_train_val, y_train_val, test_size=10000, random_state=42)
```

4.3 Modellering

Modeller har valts efter deras förmåga att hantera klassificeringsproblem och vilka modeller som i allmänhet använts av andra på Github.

En pipeline skapades som först standardiserade bilderna samt skapade en placeholder för den bäst presterande modellen. Tre klassificeringsmodeller, *RandomForestClassifier*, *LinearSVC* samt *XGBClassifier* placerades i pipelinen men tränades individuellt beroende på att de tog väldigt lång tid att köra GridSearchCV för hyperparameteroptimering på varje modell.

Först tränades modellen på träningsdata varpå *XGBClassifier* presterade bäst av de tre modellerna. Efter det justerades hyperparametrarna med GridSearch. Sen utvärderades modellen på valideringsdata som fick en accuracy på 0,9805. En Confusion Matrix skrevs ut efter det för att analysera vilka siffror som modellen hade svårt med. Allt såg bra ut och en slutlig utvärdering gjordes på testdata som gav ett accuracy på 0,9777. För att förbättra modellen ytterligare tränades den om på både tränings- och valideringsdata och predikterade på testdata återigen för att slutligen sparas med joblib och användas i Streamlit-applikationen.

4.4 Kod i Streamlit

Den sparade modellen laddas in i Streamlit. Utmaningen var främst i att förstå vad som måste göras med bilden innan den skickas till modellen för prediktion. Källor till vad som gjorts innan, i liknande projekt, var olika sökningar på nätet angående MNIST prediktering, github och ChatGpt.

Först skapades en canvas där användaren kan rita sin siffra. Efter det skapades två funktioner. Den första, *def binarize_image(image_array);*, för att binarisera en bild genom att dela in den i bakgrund och förgrund. Kortfattat gör den en gråskalebild till svartvit. Andra funktionen *def preprocess_image(image_array);*, skapades för att bearbeta bilden. I bearbetningsprocessen konverteras bilden till en gråskala, sen vidare till svart och vit samt säkerställs att bakgrunden är svart och den ritade siffran vit.

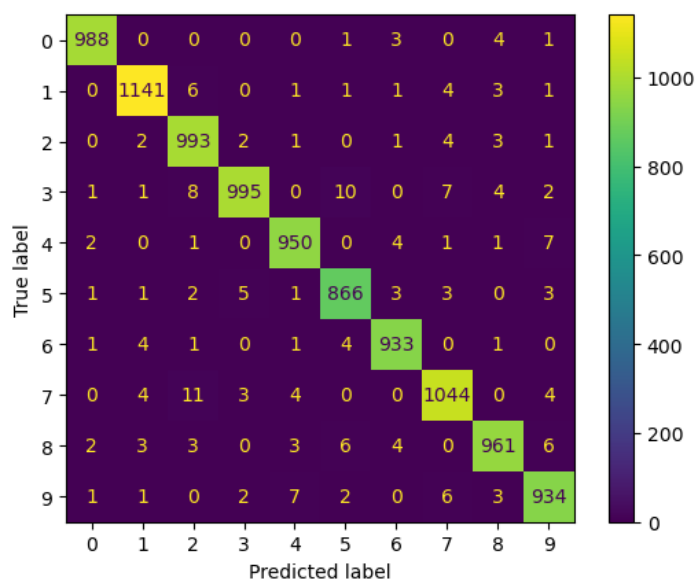
Sedan identifieras var siffran finns i bilden genom att titta på siffrans konturer, siffran skalas då om till 20x20 pixlar och centreras i en 28x28 pixlar vilket också är MNIST format. En sista centrering av siffran görs med warpAffine. Den förbehandlade bilden skrivs ut vilket har varit väldigt bra för felsökning under processen. Sist i bearbetningsprocessen konverteras bilden från en 2D-matris (28, 28) till en 1D-vektor med 784 pixlar i rad, (1, 784).

Under koden för canvas, sifferinmatningen kontrolleras att om användaren ritat något, om inte ges meddelandet, "Ingen siffra ritad! Rita en siffra innan du gör en prediktion.". Bilddata från canvas hämtas och skickas funktionen för bearbetning, *preprocess_image*. Efter det skickas den bearbetade bilden till en kod som gör en sannolikhetsbaserad prediktion och kontrollerar modellens säkerhet innan den visar resultatet. Om sannolikheten är för låg kan modellen välja att inte använda prediktionen och ge meddelandet, "Osäker prediktion! Modellen kan vara tveksam.". Om sannolikheten är över 0,5 visas resultatet som en framgångsrik prediktion.

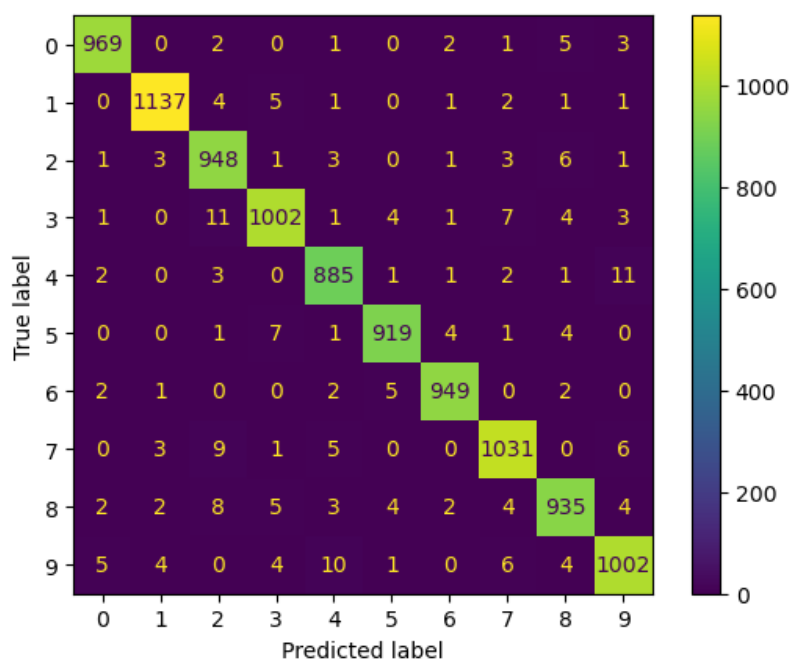
5 Resultat och Diskussion

Med anledning till att det tog lång tid att träna modellerna gjordes det var för sig via GridSearchCV. Enbart accuracy för XGBClassifier finns separat. Till nästa projekt kan det vara bättre att träna modellen utan att lägga alla i en pipeline för att kunna spara resultatet för varje modell. Resultatet/prediktionerna på valideringsdata samt testdata blev följande:

Accuracy på valideringsdata: 0,9805



Accuracy på testdata: 0,9777



Vid testning i Streamlit-applikationen lyckades modellen klassificera korrekt 10/10 siffror när tre olika användare ritade på canvassen. Vilket anses vara ett bra tecken på att modellen generaliserar bra. Dock har det i klassens forum visat sig att modellerna kan ge varierande resultat beroende på användarens handstil vilket också kan förekomma med denna modell. Ett exempel på resultatet för en användare finns presenterat i Appendix A.

6 Slutsatser

För att svara på frågeställningarna,

1. *Hur kan en kod utformas för att välja den bäst presterande modellen?* Se kod på Github, [Kunskapskontroll_ML2_André_Lindeberg](#)
2. *Hur kan en kod utformas för att skapa en applikation i Streamlit med syfte att prediktera handskrivna siffror skrivna på canvas i Streamlit?* Se kod på Github, [St_app.py](#)
3. *Kan modellen i Streamlit-applikation prediktera alla siffror (0–9) rätt när en användare ritat siffrorna i canvasen i applikationen?* I skrivande stund har tre olika användare testat att skriva alla siffror på canvas i Streamlit-applikationen som lyckats prediktera rätt på alla siffror för alla tre användare.

Länk till Streamlit-applikationen:

<https://machine-learning2025-ddztlb4gdfpsylwamf3zp.streamlit.app/>

7 Teoretiska frågor

Besvara nedanstående teoretiska frågor koncist.

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

Träning – används för att träna sin modell på datan.

Validering – används för att justera hyperparametrar och modell.

Test – används för att se hur bra modellen presterar på osedda testdata.

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 "validerings data set"?

Hon kan använda sig av GridSearch för att hitta de bästa parametrarna. GridSearch gör Cross Validation på träningsdatan. Sen får hon jämföra modellerna med funktionen `cross_val_score`.

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

Ett regressionsproblem kännetecknas när den beroende variabeln (y-variabel) är ett kontinuerligt numeriskt värde. Tillämpningsområden kan vara när vi ska prediktera huspriser eller förutsäga löner baserat på oberoende variabler (X).

4. Hur kan du tolka RMSE och vad används det till: $RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$

RMSE kan enkelt tolkas som det genomsnittliga medelavståndet från våra predikterade punkter till de sanna värdena (testdata)?

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

Ett klassificeringsproblem är när den beroende variabel (y) inte är ett kontinuerligt numeriskt värde utan antar kategorier så som "ja/nej", "man/kvinna" eller "hund/katt".

Vanliga modeller som används är Logistisk Regression, Support Vector Machines och Random Forrest.

Confusion Matrix visar hur många gånger modellen predikterat rätt eller fel. Den visar sambandet mellan predikterade värde och det sanna värdet.

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

K-means är en modell som försöker klustra data setet. Den försöker hitta varje klusters center, även kallat klustrets centroid. Den kan sen sätta en etikett för varje kluster. Algoritmen kan tillämpas inom bildsegmentering och anomalidetektering.

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 – konverterar kategoriska data till numeriska värden där kategorierna har en naturlig ordning. Exempel är betyg, där IG, G, VG kan tilldelas 1, 2, 3.

One-hot encoding – Skapar ett binärt attribut per kategori. Attributet får en 1 om det matchar annars en 0.

Dummy variable encoding – är nästan samma som *One-hot encoding*, skillnaden är att i one-hot-encoding skapas lika många binära attribut som kategorier. I Dummy variable encoding skapas ett mindre binärt attribut där en kategori enbart innehåller nollor.

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?

Julia har rätt då färger inte direkt har någon inbördes ordning. Men om vi tilldelar färgerna en ordning så blir de/datan ordinal.

9. Kolla följande video om Streamlit: <https://www.youtube.com/watch?v=ggDaRzPP7A&list=PLgzaMbMPEHEX9AIs3F3sKKXexWnyEKH45&index=12>

RzPP7A&list=PLgzaMbMPEHEX9AIs3F3sKKXexWnyEKH45&index=12

Och besvara följande fråga: - Vad är Streamlit för något och vad kan det användas till?

Streamlit är ett ramverk med öppen källkod där vi kan skapa applikationer för maskininlärning med hjälp av Python.

8 Självutvärdering

1. Utmaningar du haft under arbetet samt hur du hanterat dem.

En väldigt utmanade kurs att få grepp om. Vilka steg som ska ske när har varit utmanande.

Modelleringen har jag gjort om flera gånger och undrar vad jag skulle kunna göra för att träna min modell med alla tre modeller i pipeline. Var tvungen att avbryta efter 6 timmar då jag kände att det inte borde ta så lång tid. Men jag löste det genom att köra en modell i pipeline vilket gick mycket fortare.

Streamlit-applikation var också väldigt tuff då det inte gick att få svar från kolleger. Skulle önska att jag förstod mer om koden. Jag läste dokumentationen på Streamlit, tittade på koder från liknande projekt på github och så använde jag ChatGpt vilket jag inte tycker om men kändes nödvändigt för att klara uppgiften.


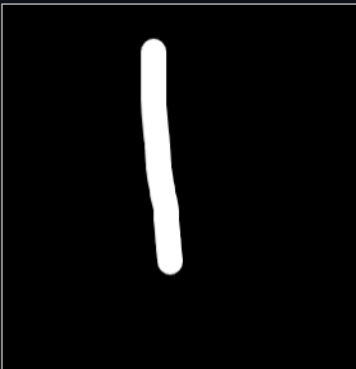






2. Vilket betyg du anser att du skall ha och varför.







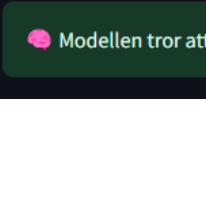
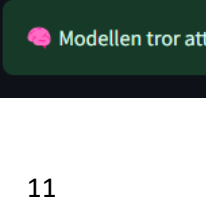
Jag anser att jag uppfyller alla kriterier för G och VG

3. Något du vill lyfta fram till Antonio?

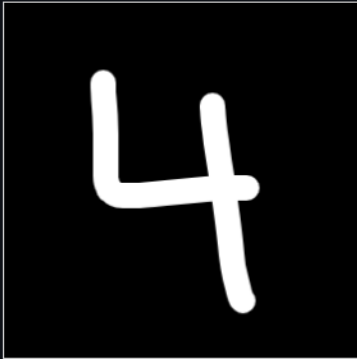
Ja, VG-delen tycker jag var viktig att få med sig och tycker den skulle ingå i undervisningen.

Appendix A

Rita en siffra i rutan nedan och låt modellen gissa!		Rita en siffra i rutan nedan och låt modellen gissa!	
			
			
Förbehandlad bild	Förbehandlad bild	Förbehandlad bild	Förbehandlad bild
🧠 Modellen tror att detta är en 0!		🧠 Modellen tror att detta är en 1!	

Rita en siffra i rutan nedan och låt modellen gissa!		Rita en siffra i rutan nedan och låt modellen gissa!	
			
			
Förbehandlad bild	Förbehandlad bild	Förbehandlad bild	Förbehandlad bild
🧠 Modellen tror att detta är en 2!		🧠 Modellen tror att detta är en 3!	

Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

Modellen tror att detta är en 4!

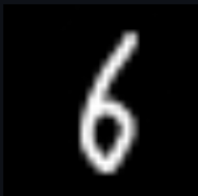
Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

Modellen tror att detta är en 5!

Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

Modellen tror att detta är en 6!

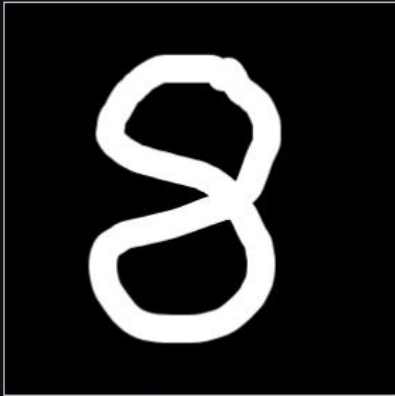
Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

Modellen tror att detta är en 7!

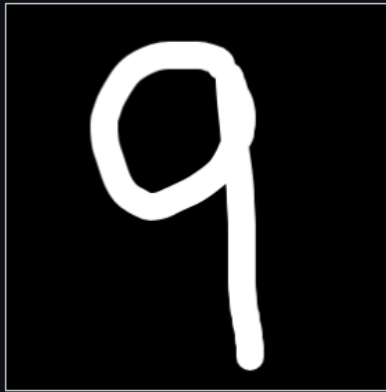
Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

🧠 Modellen tror att detta är en 8!

Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

🧠 Modellen tror att detta är en 9!

Rita en siffra i rutan nedan och låt modellen gissa!



Förbehandlad bild

Osäker prediktion! Modellen kan vara tveksam.

Källförteckning

Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems (Second edition). O'Reilly.

MNIST database. (2024). I Wikipedia.

https://en.wikipedia.org/w/index.php?title=MNIST_database&oldid=1262526473

Streamlit Docs. (u.å.). Hämtad 17 mars 2025, från <https://docs.streamlit.io/>

Prgomet, A. (2025). *Maskininlärning och modellutvärdering* [Föreläsning]. EC Utbildning, Data Scientist-programmet.