Machine Learning_23:

Beginner MNIST Data analysis

Quang Tri Tran

EC Utbildning

Kunskapskontroll 2

2024-03

Abstract

In this workshop, we will explore MNIST database and find out what nature this database is. To help us in our digital journey we will include python libraries such as Matplotlib and NumPy and additionally Machine learning algorithms to help us understand MNIST better. From Machine learning algorithms there will be regression models and classification models to test the category tasks of this MNIST dataset. Results indicates MNIST as a classification category database since during exploration the dataset shows images of numbers ranging between 0 to 9 in 28x28 pixels as its data. This would mean classifications tasks are primary and regressions tasks are not.

Innehåll

1 Inledning

Maskininlärning (ML) är en bland många andra grenar som är involverad i skapande av AI. Det är genom maskininlärning som datorsystem använder för att utveckla algoritmer och statistiska modeller som används för att utföra komplexa uppgifter utan explicita instruktioner från användaren. I detta arbete kommer vi utforska vetenskapen bakom maskininlärningen, hur ett arbetsflöde ser ut och med samlad kunskap experimentera med MNIST databasen. Detta arbete är menad för att skapa en god grund och förståelse om maskininlärning.

Syftet med denna rapport är att undersöka MNIST databas och bestämma vilken modell som passar bäst, för att uppfylla syftet så kommer följande frågeställning(ar) att besvaras:

1. Vad är MNIST?
2. Är det en regressionproblem?
3. Är det ett klassificeringsproblem?

2 Teori

Källor och referenser i denna rapport är ursprungligen från Aurelien Géron (2019). Modellerna finns mer att läsa om i hemsidan [https://scikit-learn.org/stable/]. För att göra det enkelt för läsaren och rapporten kommer endast parameter som används i modellerna i detta arbete att nämnas.

2.1 Regressionsmodeller

Regression är en teknik som används för prediktiv modellering i maskininlärning där en algoritm används för att förutsäga kontinuerliga resultat. En modells regression kan bedömas genom RMSE som står för *Root Mean Squared Error*, som är en mätmetod för att utvärdera regressionsproblem och mäter den genomsnittliga skillnaden (medelavståndet) mellan en statistisk modells förutsagda värden och de faktiska värdena. Formeln för RMSE visas nedan i figur 1.

$$RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n}(\mathrm{y_i} - \hat{y}_i^2)}$$

Figur 1. *Formula for Root mean squared error. (Géron, 2019)*

2.1.1 Linjär regression

Linjär regression är en fundamental statistisk modell som beräkna och predicera den linjära sambanden mellan beroende variabler och oberoende variabler. Det är en enkel modell utan att behövas justera sina parameter.

2.1.2 Lasso regression

Lasso regression eller Least Absolute Shrinkage and Selection Operator Regression är en regulariserad variant av linjär regression och använder sig av straffterm L1, för att förhindra överanpassning.

2.1.2.1 Regularisering i Lasso

Regulariseringen i Lasso modellen hjälper till med att undvika överanpassning genom att straffa vissa koefficienter i modellen. Regulariseringsstyrkan är beroende av parametern ”alpha” som styr L1 strafftermer.

2.1.2.2 Hyperparameter

Alpha: styr regulariseringsstyrkan, är alpha = 0 så fungera modellen som en enkel linjär regression utan strafftermer.

2.1.3 Ridge regression

Modellen är en regulariserad version av linjär regression, tillskillnad från Lasso regression som följer L1 straffterm så har Ridge regression L2 straffterm.

2.1.3.1 Hyperparameter

Alpha: styr regulariseringsstyrkan, är alpha = 0 så fungera modellen som en enkel linjär regression utan strafftermer.

2.1.4 Elastic Net

Modellen är en kombination mellan Lasso och Ridge då modellen använder både L1 och L2 regularisering.

2.1.4.1 Regularisering i Elastic Net

Regulariseringstermen är en mix av Lasso och Ridge och går att styra genom att justera på förhållande uttrycken, r. Om r=0 så fungera den ekvivalent som Ridge regressionen och om r=1 är den likt med Lasso regressionen.

2.1.4.2 Hyperparameter

Alpha: Styr regulariseringsstyrkan, är alpha = 0 så fungera modellen som en enkel linjär regression utan strafftermer.

l1_ratio: Blandningsparameter, styr kombinationen mellan L1-straff och L2-straff.

2.2 Klassifikations modeller

2.2.1 Logistik regression

Det finns vissa regressionsalgoritmer som kan användas till klassifikationer, logistik regression är en av dem. Modellen är vanligt använd för att uppskatta sannolikheten om en instans tillhör en viss klass eller inte. Alltså om den uppskattade sannolikheten är större än 50% då förutsäger modellen att instansen tillhör den klassen och vice versa.

2.2.1.1 Välja hyperparameter

C: denna parameter kontrollera regulariseringsstyrkan. Högre värde ge reducerad styrka och vice versa.

max_iter: kontrollera hur många iterationer som ska göras.

2.2.2 MLPClassifier

MLPclassifier är en neural nätverksalgoritm som används för klassificeringsuppgifter. Det står för Multi-layer Perceptron classifier. Modellen förlita sig på ett underliggande neuralt nätverk för att utföra sin klassificering vilket andra klassificeringsalgoritmer inte har såsom Support Vectors.

2.2.2.1 Hyperparameter

Hidden_layer_sizes: parametern specificera antalet lager och antalet noder som skapas i modellens neurala nätverk.

Activation: Aktiverings funktionen för hidden_layer.

Alpha: kontrollera styrkan i L2 regulariseringen.

Max_iter: kontrollera antalet iterationer som kommer utföras.

3 Metod

Hur har du genomfört ditt arbete? Exempelvis, hur har datan erhållits?

3.1 Studera data

Databasen som används är MNIST_784 som laddas in med hjälp av python kodning i samband med Scikit-learn biblioteket. MNIST är en känd dataset som används inom maskininlärning. För att förstå mer om MNIST databasen utförs en dataanalys för att undersöka databasens struktur, vad det är för variabler och vilken data typ som kommer arbetas med.

3.2 Förbehandla data – Split & Preprocessing

MNIST dataset splittras i tränings set, validerings set och test set. Indelningen är 80% tränings data, 10% validerings data och 10% test data. Därefter förbehandlas data med StandardScaler som standardisera data så att den korrekt passa in i modellerna samt förbättra tolkningsbarheten.

3.3 Välja modell – Regressionsmodeller

Regressionsmodellerna importeras in och körs med standard parameter för att observera resultaten. Parametrarna kommer justeras för att välja den mest optimala inställningar för bästa prestationsförmåga med metoden GridSearchCV. Modellerna körs om med optimala parametrar och resultatet observeras och bedöms. Bedömningen för att välja ut bästa regressionsmodellen beror på omdömet från RMSE utvärderingen.

3.4 Välja modell – klassifikationsmodeller

Logistisk regression och MLPClassifier är dem enda klassifikationsmodellerna i detta arbete. De importeras och körs först med standard parameter. GridSearchCV används därefter för att justera om parametrarna till optimal inställning. Bedömningen ske i form av träffsäkerhet i prediktivförmåga med hjälp av accuracy_score metoden. En Confusion matrix kommer utföras för att visa upp sammanfattad prestation hos modellen, detta görs för att få en bildtolkning på modellens prediktiva beteende.

All programmering är gjord i Jupyter Notebook i språket Python. Bibliotek som används är Numpy, Matplotlib och Scikit-learn.

4 Resultat

4.1.1 Data analys

En översikt på data analysen över MNIST 784 databas, se Appendix A.

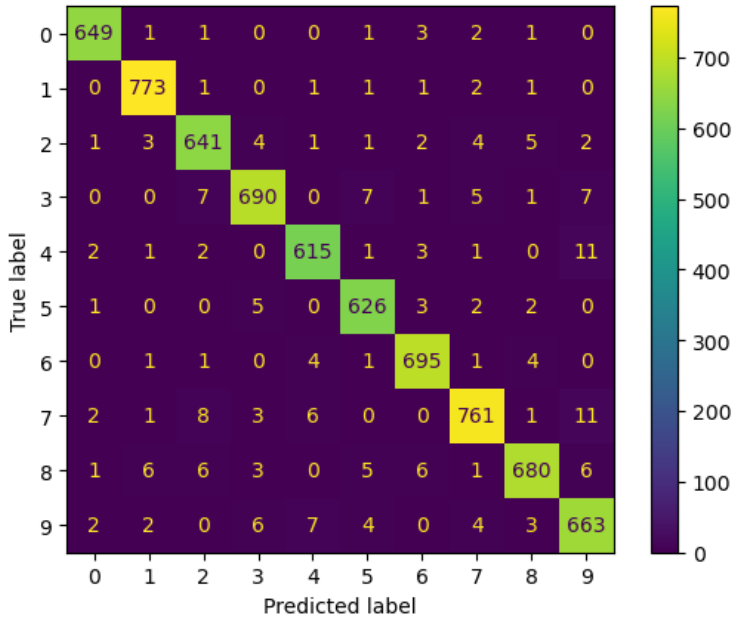
4.1.2 Modell analys

Olika modellers resultat både med standard parameter och optimala parameter.

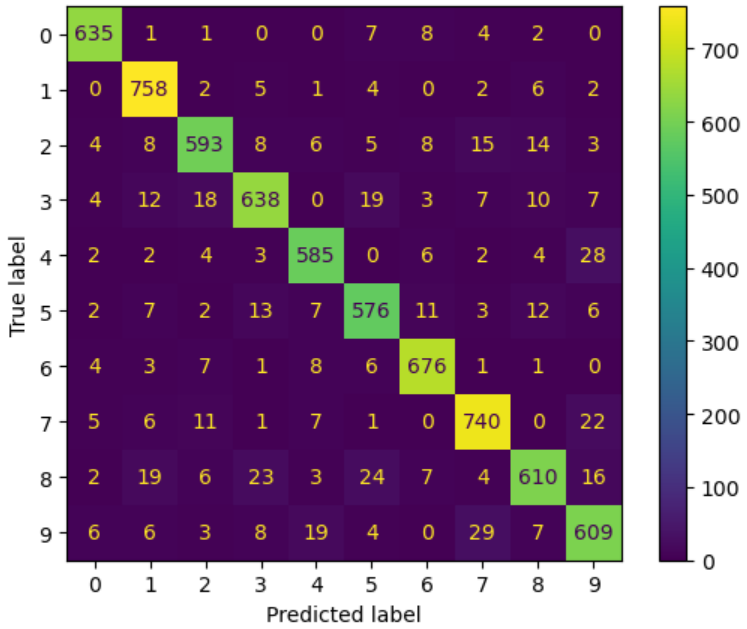
Resultat med standard parameter

RMSE för regressionsmodeller	
Enkel Linjär Regression	7.9e+9
Lasso	1.93
Ridge	1.89
Elastic Net	2.42
Accuracy Score för klassifikationsmodeller	
Logistik regression	91%
MLPClassifier	97%

Tabell 1: Root Mean Squared Error (RMSE) för de fyra valda modellerna.



Figur 6. *Confusion matrix, visa MLPClassifiers summerad prestation med standard parameter.*



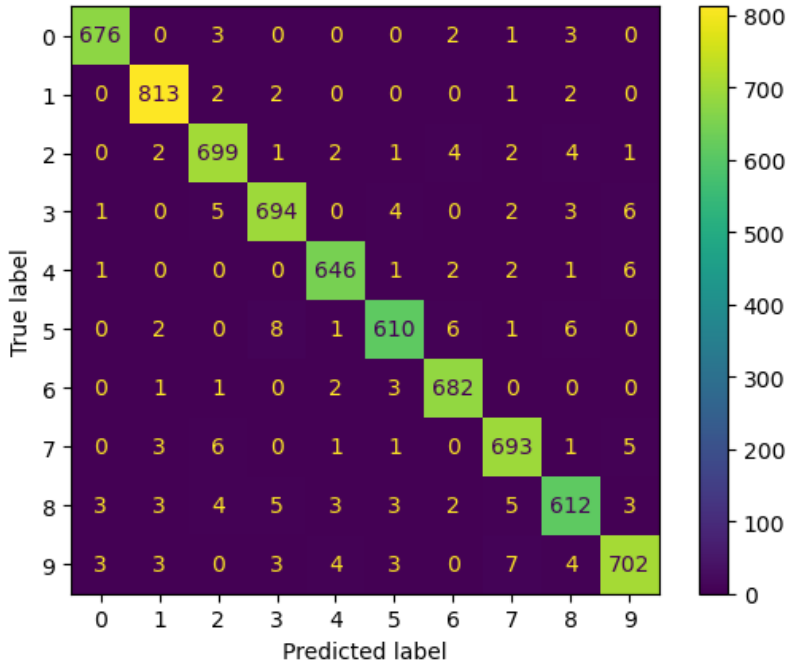
Figur 7. *Confusion matrix, visa Logistik regression summerad prestation med standard parameter.*

Resultat med optimala parameter

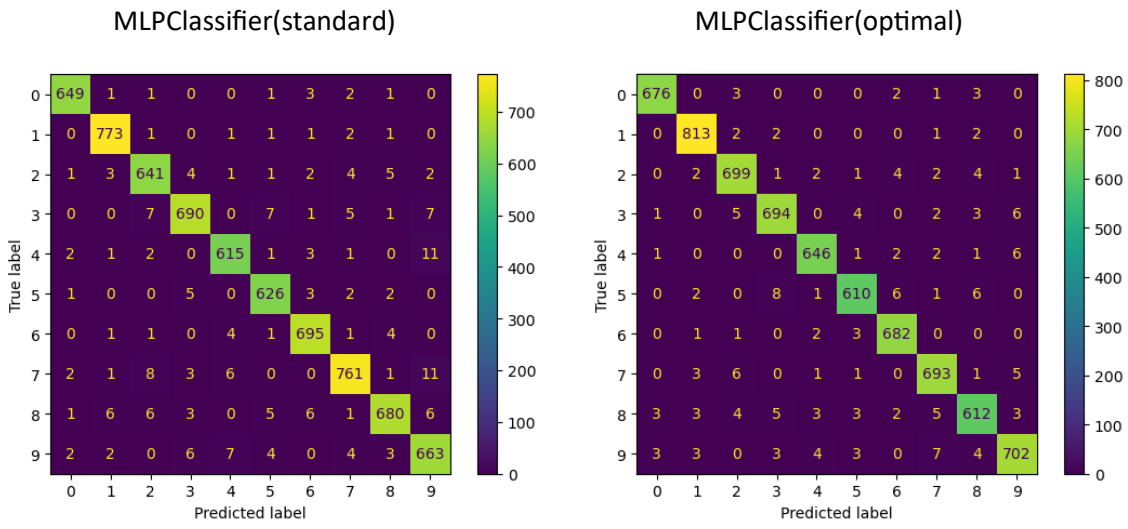
RMSE för regressionsmodeller	
Enkel Linjär Regression	9e+9
Lasso	1.94
Ridge	1.88
Elastic Net	1.97
Accuracy Score för klassifikationsmodeller	
Logistik regression	91.9%
MLPClassifier	97.5%

Välja bästa modell

Bäst modell jämförelse	
Modell	**Accuracy rate**
Ridge (Regression)	59.93%
MLPClassifier (Klassifikation)	97.53%



Figur 8. *Confusion matrix, visa MLPClassifiers summerad prestation med optimala parameter.*



Figur 9. *Confusion matrix, visa MLPClassifiers summerad prestation före (vänster) och efter (höger) justering av parameter.*

5 Slutsats & Diskussion

5.1 Diskussion

Undersökningen visade att MNIST_784 är en databas som innehåller en samling av handskrivna siffror (0 till 9) som är digitaliserade och i denna dataset innehåller det 70 000 data exempel i form av bilder i 28x28 pixlar. Analysen upplyste att MNIST förmodligen är en databas för klassifikationsalgoritmer, alltså för klassifikationsproblem. Det bevisades senare i resultatet att MNIST inte är gjord för att hantera regressionsalgoritmer.

För bland regressionsmodellerna som tävlade mot varandra så var det en modell som hade en tydlig exponentiellt utfall och det var enkel linjär regression som visade dålig prestations resultat att hantera MNIST databasen då högre RMSE värden innebär dålig prestationsförmåga. De andra 3 regressionsmodellerna (Lasso, Ridge, Elastic net) anpassade sig mycket bättre än linjär regression då RMSE värdena låg mycket lägre än linjär regression. Anledningen kan vara för att det finns strafftermer som begränsa hur dem ska beter sig och därmed minimeras förmågan att bete sig som linjär regression som inte har strafftermer i sin modell.

Resultatet från klassifikationsmodellerna däremot visade godtyckliga resultat vilket indikera att både MLPClassifier och logistik regression kunde anpassa sig till MNIST databas. Deras prediktiva förmåga observerades som 97% för MLPClassifier och 91% för logistik regression. Detta ger indikation att databasen är bäst hållen för klassifikationsmetoder. Även i jämförelse med den optimala regressionsmodellen så utvinner klassifikationsmodellen i prestation för denna MNIST databasen, med en träffsäkerhet på 97.53% (MLP) i jämförelse med 59.93% (Ridge).

5.2 Slutsats

Regressionsmodeller är inte lämpliga att använda eftersom MNIST 784 inte är avsett för regressionsuppgifter då målet för regressionsmodeller är att förutsäga kontinuerliga värden. MNIST 784 innehåller bilder vilket gör klassificeringsupgifter mer primärt. MNIST är därmed specifikt designat för klassifikationsuppgifter vilket innebär att databasen kategoriseras som klassifikationsproblem och klassifikationsalgoritmer rekommenderas som arbetsmetod.

Eftersom för en klassificerare, givet en MNIST-bild, så kan klassificeraren förutsäga om den representerar siffran 5 eller inte. En regressionsuppgift skulle hellre fortsätta att förutsäga det exakta pixelvärdet för en specifik pixel i en MNIST-bild.

6 Teoretiska frågor

1. Kalle delar upp sin data i ”Träning”, ”Validering” och ”Test”, vad används respektive del för?
   SVAR:
   Training – Tränings data används för att lära upp modellen och måste inkludera alla parametrar och inmatningar som modellen skall tränas upp med.

   Validation – Validerings data som är steget emellan Träning och Test, används i syfte för att justera modellens hyperparameters för att sedan observera om modellen blivit bättre prestationsförmåga med optimala parametrar. Vid detta stadie blir inte modellen upplärd, validerings data syfte träna inte modellen utan den justera inställningar så den kan prestera så bra som möjligt. Används för att välja ut bästa möjliga modellen.

Test – Test data som ska förses som simulering på verkliga världen, används för att predicera utsatser utanför modellens data som potentiellt kan inträffa i verkliga världen. Ju närmre modellens prediktering kommer till test data desto bättre blir modellens prestation och reliabilitet.

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”?
   SVAR:
   Hon har två modeller som är av regression, den sista modellen ”Random forest modell” är mindre informerad om det är regression eller klassifikation modell.

   Hon kan skapa ett korsvalideringssystem för att bedöma vilken av modellerna är den mest optimala genom att tävla modellerna mot varandra. I en korsvalidering kan både regressionsmodeller och klassifikationsmodeller bedömas. För regressioner bedöms prestanda genom exempelvis RMSE, och för klassifikationer bedöms prestandamåttet genom noggrannhet, precision. Fördelarna med korsvalidering är att den ge en mer realistisk uppfattning av modellens prestanda och minska risken för överanpassning samt i korsvalidering processen så valideras även data.

3. Vad är ”regressionsproblem? Kan du ge några exempel på modeller som används och potentiella tillämpningsområden?
   SVAR:
   Regressionsproblem innebär att uppgifterna är kvantitativa, kontinuerliga variabler. Målet är att förutsäga kontinuerliga värden såsom huspriser, temperaturprognoser eller försäljningar. Exempel på modeller som använder regression är linjär regression, Lasso regression, Ridge regression.

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

   RMSE används för att predicera medelavstånd från riktiga och observerade värden. Tolkningen fås i form av felmarginal värden som bedöms, en RMSE med låg värden indikera bättre prestationsförmåga. Används för att bedöma prestandamåttet för regressions modeller.

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{n}}\sum_{1}^{\mathrm{n}}(y_i - \hat{y}_i)^2}$$

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

   Klassificeringsproblem handlar om att kategorisera data i olika klasser t.ex. identifiera om det är en 3 eller inte, om det du ser är en hund eller inte. Målet att förutsäga en diskret klass baserat på indata.
   Exempel på några modeller som används är: SVM, KNN, MLPClassifier
   Potentiella tillämpningsområden kan vara som spamfilter och bildklassificering.

   Confusion Matrix – Confusion Matrix är en tabell som används för att mäta prestationen på klassifikations modeller. I tabellen summeras prestationen hos modellen och visar antalet true positives, true negatives, false positives, och false negatives. En bra modell har högre värden i diagonalen och lägre värden utöver.

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

   K-means är en klustering algorithm. Det är en vanligt använd unsupervised maskininlärningsalgoritm för att gruppera liknande datapunkter tillsammans, även förklarad som att klustra ihop liknande datapunkter. I praktiken när den används så hitta man naturliga kluster i sin data och tolkningen kan hjälpa dig t.ex. att hitta olika grupper av individer såsom kön, ålder eller ursprung.

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

   Ordinal encoding används om det finns ordning i data. Exempel, om du har färger som röd, blå och grön så kommer ordinal encoding ge varje färg ett värde som röd =1, blå=2 och grön =0.

   One-hot encoding används om det inte finns någon ordning i data. Exempel, one-hot encoding skapa C-variabler till antalet C kategorier och kommer göra så att om en person har färgen röd så blir röd = 1 och resterande får värdet 0.
   Röd –> röd = 1, blå = 0, grön = 0
   Blå –> röd = 0, blå = 1, grön = 0
   Grön –> röd = 0, blå = 0, grön = 1

   Dummy variable encoding är en förenklad variant av one-hot encoding.
   Exempel, vi skapa C-1 variabler till antalet C kategorier vilket innebär om vi har 3 kategorier så skapa vi 2 variabler. Om kategorin är röd, blå och grön så skapas som förslag variablerna blå och grön men inte röd. Nu om en person har färgen röd så kommer dummy variable encoding ge blå = 0 och grön = 0 vilket indikera att det är en röd färg.
   Blå → blå = 1, grön = 0
   Grön → blå = 0, grön = 1

Skillnaden mellan one-hot encoding och dummy variable encoding är att one-hot encoding skapa binär variabel för varje kategori och dummy variable encoding skapar C-1 variabler för att representera C kategorier.

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?
   SVAR:
   Låt oss dela upp perspektiverna.
   Göran säger endast att data antingen är ordinal eller nominal. Ordinal innebär naturlig ordning och nominal är att det saknas inbördes ordning.

   Julias perspektiv säger att färger är nominala men om Göran tar på sig en färg så skapas det en inbördes ordning alltså vi kan rangordna Göran och dem andra på festen med vad dem har på sig för kläder.

   Helt enkelt färg i sig saknar mening och innehåll men om färgen används till målningar eller kläder så skapas det en innebörd och då får den innehåll och mening, vilket går att rangordna.

   Detta är en tolkningsfråga och båda har rätt till en del av sanningen! Ingen har fel. Göran är mer teoretiskt och Julia är mer praktiskt tänkande.

9. Kolla följande video om Streamlit:
   https://www.youtube.com/watch?v=ggDaRzPP7A&list=PLgzaMbMPEHEx9Als3F3sKKXexWnyEKH45&index=12 Och besvara följande fråga: - Vad är Streamlit för något och vad kan det användas till?
   SVAR:
   Streamlit är en applikation för Python som gör om data kodning till interaktiv webb applikation. Den transformera python koden så att användaren kan interagera med t.ex. data visualiseringen i maskininlärningsmodellen. Så i stället för en statisk modell så får vi en interaktiv modell. Streamlit kräver inte att kunna skapa hemsidor utan behövs bara att kunna koda i python för att fungera.

7 Självutvärdering

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

   Detta var nog svåraste kursen hittills, jag förstår varför. Att förstå vad varje modell gör och parametern som spelar roll för modellen tog lång tid. Att förstå varför modellen inte fungera som den ska för databasen är upp till att ställa frågan ”vad är det för uppgift och vad för problem är denna typ av databas”.

   Jag är himla nöjd att dokumentationen till varje modell är lätt att läsa och komma tillbaka till. De flest problem fick jag utföra taktiken att kolla tillbaka mina steg och debugga eller testa annan approach. Nyckeln att hantera felmeddelande är att fortsätta kämpa.

2. Vilket betyg du anser att du skall ha och varför.
   Med arbetet som är lagd i Jupyter Notebook och mängden läsning för att lära mig svara på teorifrågorna så anse jag mig att ha ett G då VG-kriterierna kräver att inkludera Streamlit i skripten vilket jag inte har gjort då tiden är bristande.
   Men det var en trevlig utmaning att få testa olika modeller och observera hur modellen lär sig från data på olika sätt, och att databas inte är anpassad för alla modeller.

3. Något du vill lyfta fram till Antonio?
   Dina lektionsvideos är super!

   Lektions anteckningar och dokument är lätt att följa med. Speciellt under lektionsdagar så läser jag ibland lektionschatten när dagen är slut och får små inblickar här och där.
   Det hjälpte väldigt mycket att se kodnings exempel då det underlätta min förståelseförmåga så jag hade velat ha fler av dem i framtida kurser om dem involvera att förstå kodteori.

Appendix A

```
In [3]: 1 print(mnist.DESCR)
```

Author: Yann LeCun, Corinna Cortes, Christopher J.C. Burges
Source: [MNIST Website](http://yann.lecun.com/exdb/mnist/) - Date unknown
Please cite:

The MNIST database of handwritten digits with 784 features, raw data available at: http://yann.lecun.com/exdb/mnist/. It can be split in a training set of the first 60,000 examples, and a test set of 10,000 examples

It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting. The original black and white (bilevel) images from NIST were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. the images were centered in a 28x28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28x28 field.

With some classification methods (particularly template-based methods, such as SVM and K-nearest neighbors), the error rate improves when the digits are centered by bounding box rather than center of mass. If you do this kind of pre-processing, you should report it in your publications. The MNIST database was constructed from NIST's NIST originally designated SD-3 as their training set and SD-1 as their test set. However, SD-3 is much cleaner and easier to recognize than SD-1. The reason for this can be found on the fact that SD-3 was collected among Census Bureau employees, while SD-1 was collected among high-school students. Drawing sensible conclusions from learning experiments requires that the result be independent of the choice of training set and test among the complete set of samples. Therefore it was necessary to build a new database by mixing NIST's datasets.

The MNIST training set is composed of 30,000 patterns from SD-3 and 30,000 patterns from SD-1. Our test set was composed of 5,000 patterns from SD-3 and 5,000 patterns from SD-1. The 60,000 pattern training set contained examples from approximately 250 writers. We made sure that the sets of writers of the training set and test set were disjoint. SD-1 contains 58,527 digit images written by 500 different writers. In contrast to SD-3, where blocks of data from each writer appeared in sequence, the data in SD-1 is scrambled. Writer identities for SD-1 is available and we used this information to unscramble the writers. We then split SD-1 in two: characters written by the first 250 writers went into our new training set. The remaining 250 writers were placed in our test set. Thus we had two sets with nearly 30,000 examples each. The new training set was completed with enough examples from SD-3, starting at pattern # 0, to make a full set of 60,000 training patterns. Similarly, the new test set was completed with SD-3 examples starting at pattern # 35,000 to make a full set with 60,000 test patterns. Only a subset of 10,000 test images (5,000 from SD-1 and 5,000 from SD-3) is available on this site. The full 60,000 sample training set is available.

Downloaded from openml.org.

Figur 2. *Beskrivningen på MNIST databasen.*

```
In [4]: 1 mnist.data
```

Out[4]:

	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	pixel10	...
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
...	...	...	...	...	...	...	...	...	...	...	...
69995	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
69996	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
69997	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
69998	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
69999	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...

70000 rows × 784 columns

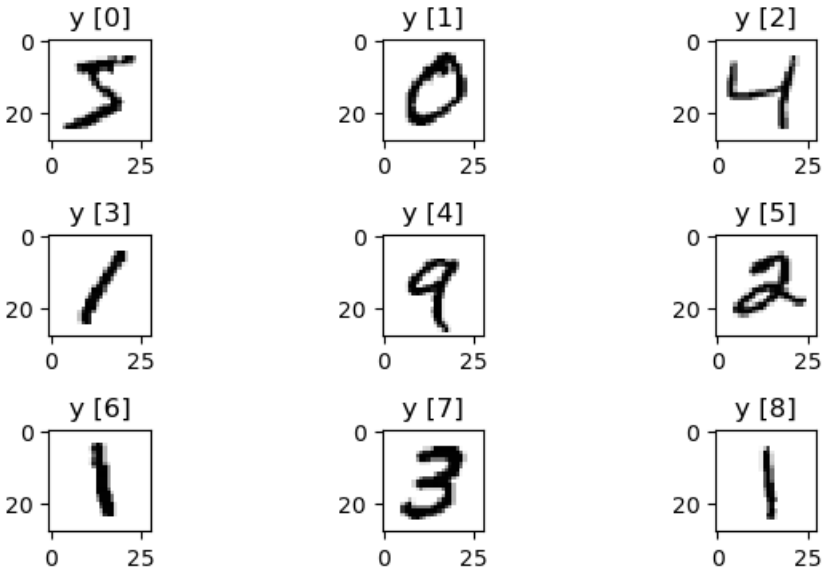
Figur 3. *MNIST databasens "Feature" data. Antalet rader och kolumner*

```
1  mnist.target
```

```
0        5
1        0
2        4
3        1
4        9
        ..
69995    2
69996    3
69997    4
69998    5
69999    6
Name: class, Length: 70000, dtype: category
Categories (10, object): ['0', '1', '2', '3', ..., '6', '7', '8', '9']
```

Figur 4. *MNIST databasens "target" data. Bl.a. datatyp och antal kategori*

```
1  image = mnist.data.to_numpy()
2
3  for i in range(9):
4      plt.subplot(431+i)
5      plt.title('y {}'.format([i]))
6      plt.imshow((image[0+i].reshape(28,28)), cmap=plt.cm.gray_r, interpolation='nearest')
7  plt.tight_layout()
8
```



Figur 5. *Pythonkod som demonstrera dem digitala handskrivna siffrorna i en bildgraf.*

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.

Scikit-learn documentation: https://scikit-learn.org/stable/