**Dense layer model**

Load data - train and test data

reshape data to fit model – en lang vektor med 28x28 pixels

Lav all x-værdier om til fload værdier mellem 0 g 1 - både test og train data

Lav alle y-værdier om til de er one-hot encoded - både test og train data

Brug en sequential model - input layer, en række hidden layers og et output layer.

* brug Keras til at lave lagene, activation function sigmoid er en eksponentiel funktion, hvor relu er ene step funktion
* Mængden af units/neurons i vores hidden lag kan man selv vælge.

- Softmax som output layer - mapper hver af værdierne der kommer ud, efter sandsynlighed.

Til sidst kompilerer vi modellen

* Her fortæller vi den hvordan den skal måde loss – loss funktion
* Herefter bruger vi en optimizer – den skal minimere loss
* Metrics vil stort set altid være sat til metrics

For at træne vores model bruger vi model.fit og fortæller den hvor mange epoker den skal træne – hvor mange omgange.

* Her skal man passe på ikke at overtræne med for mange epoker – overfitting

**Convolutional layer model – god for billedanalyse**

Hiv dataene ind

Reshape med højde og brede af billeder samt mængden af farvekannaler

Data laves om til floads til normalisering

y-værdierne bliver lavet om til one-hot encoded

Herefter tilføjes en model nogle flere lag

* Conv2D – convolutional layers
* Et dense layer er karakteriseret ved at alle vores neuroner mellem lagene er forbundet til hinanden
* Et convolutional layer fungere ved at der er et vindu/kernal egnet til billedanalyse med højde og brede med en bestem størrelse og brede, og hver eneste af de her vinduer i det oprindelige billede transformere den som datapunkt/output i et nyt lag. Punktet bliver altså transformeret til et simplere output der giver en die om hvordan mønsteret fungerer.
  + Vi fortæller her hvor mange kernals der skal være og hvor meget output vi vil have ud af dem.
  + Her bruger vi MaxPool2D til at nedskalere det nedskalerede output fra vores convolutional layer, så vi reducere data.
  + Til sidst bruger vi flatten, så vi kan lave vores outputs om til en lang vektor – en række pixels
  + Herefter kan vi lave et sidste dense layer fro vores vektor, så den kan normalisere.
  + Til sidst bruger vi så softmax som output layer.

Til sidst kan vi bruge vores trænede model til at predicte med.

* Predict giver os predictions for en række klasses
* Predict.class() giver som prediction på en bestemt classification

**Eksamensprojekt udfordringer**

Selve eksamen

* Aflevering skal indeholde selve programmet, og en readme fil med dispositionen og selve indholdet.
* Individuel 6-7 minutters præsentation og 18-19 minutters samtale.
* Man taler dels om det man selv har lavet i projektet, men skal også kunne svare på spørgsmål om de andre dele af projektet.

Der er mange steder hvor man kan lægge sin udfordring

* Hvordan får man lavet en opgave for sig selv så man kan vise hvad man kan.
* Nogle af de nedslagspunkter som lære vil kigge efter.
* Noget man kan vise for ens egen løsning
* Faget hedder Python og Datasience og skal indeholde begge
* Problembeskrivelse for hvor den egentligeudfordring i opgaven ligger
* Komme i dybten/lægge sin indsats med en eller flere af områderne:
  + Data collection
  + Data wrangling
  + Data processing
  + Presentation

Data Collection – hvor får vi vores data fra?

* api
* webscraping
* camera, microphone
* dataset
* pdf
* sheet, csv
* user input (post to flask app, cli args, etc)

Data wrangling – hvordan skal vores data tilrettes/tilpasses

* manual overlook (eg: excel)
  + misspelled values
  + wrong columns
  + breaking up or assembling column values
* remove rows if essential data is missing
* one-hot-encoding in order for the algorithms to understand
* [normalization, standardization, rescaling](https://medium.com/@swethalakshmanan14/how-when-and-why-should-you-normalize-standardize-rescale-your-data-3f083def38ff)
  + (values scaled down to between 0 and 1),
  + (values changed to have mean = 0 and standard div = 1) respectively
* remove columns

data processing

* model fitting
* accuracy

præsentation

* filtering
* aggregation
* plotting
  + diagrams
* conclusion (ml accuracy, predictions etc)
* flask server