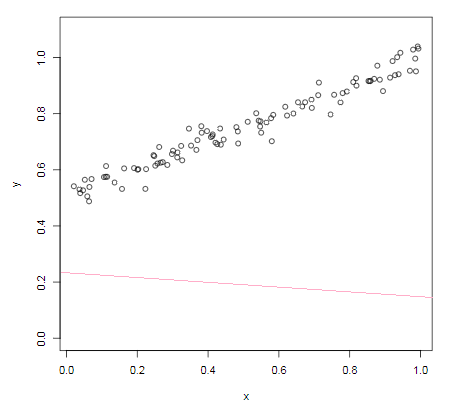
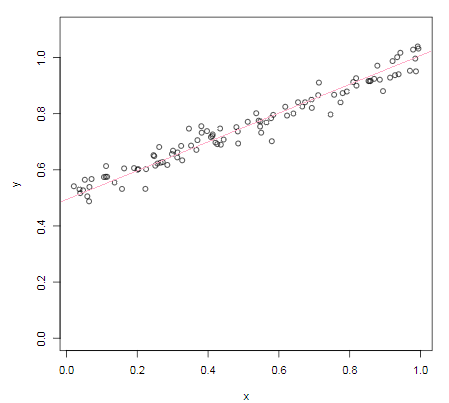
Lineær regression:

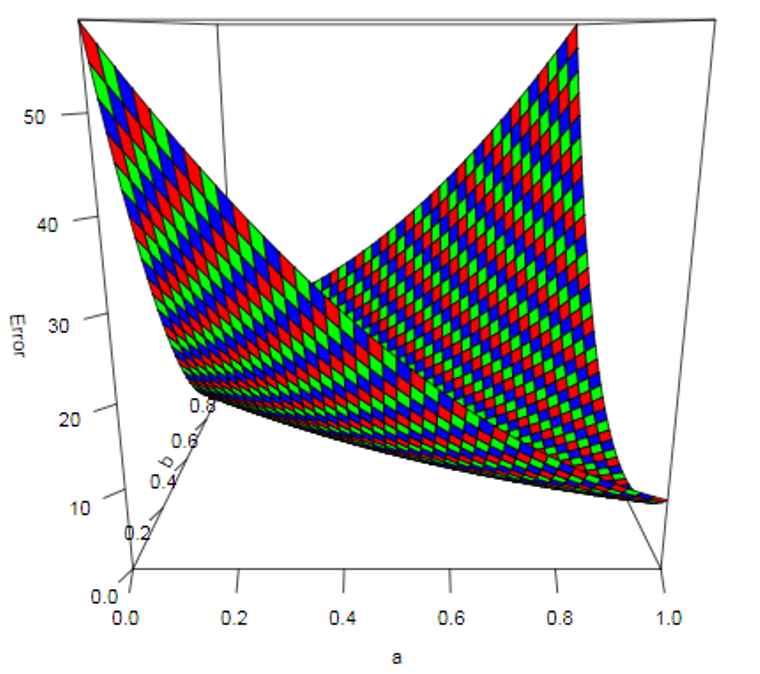
En af de mest basiske og simple supervised machine learning algoritmer er lineær regression. Lineær regression kan beskrive, som navnet hentyder, lineære data og kan tilpasse sig selv ud fra hvor godt vores regression til vores data. E.g. hvis vi har en mængde data som beskriver priser på huse i et område ud fra nogen karakteristika, i.e. mængden af toiletter, antallet af m2, afstand til det nærmest storcenter, osv. For at simplificere dette eksempel kigger vi dog kun på en variabel, antallet af m2. Hvis vi opstiller dette i et koordinatsystem med pris op ad y-aksen og m2 hen ad x-aksen. vores algoritme vil kun kunne se antallet af m2 og vil skulle lave gæt på hvor meget et normalt hus koster ud fra m2. Først vil den tage en række tilfældige tal og gætte på hvor meget husene koster, se figur 1.0.

Figur 1.0: viser data og en regression over de gæt en algoritme kunne lave. Disse gæt er tilfældige og passer derfor ikke så godt til vores data. I vores tilfælde vil prisen af huset være op ad y-aksen.

Når algoritmen har lavet et gæt, får den en fejlværdi tilbage noteret ved *θ*, som den bruger til at finde ud af hvor godt den klarede sig og vil derfor rette sig til at få den laveste fejlværdi som overhoved muligt. Se figur 1.1. Det er vigtigt at notere at fejlværdien aldrig bliver 0, da der altid i praksis kommer til at være variable som påvirker vores data til at afvige lidt

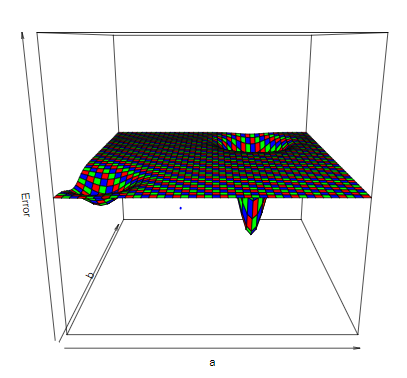
Figur 1.1: viser eksempel på lineært data hvor fejlværdien er minimal, så derfor går regressionen på grafen pænt igennem datapunkterne.

Hvordan minimere vores algoritme *θ,*så fejlværdien bliver så lille som muligt? Dette gør vi ved at plotte vores *a, b* og *θ* i et tredimensionelt koordinatsystem, med *a* hen ad x-aksen *b* hen ad y-aksen og *θ*hen ad z-aksen. Hvor *a* og *b* er *a* og *b* i en lineær funktion () og *θ*er vores fejlværdi. Dette vil danne et plan som viser hvor høj vores fejlværdi, *θ,* er når en given hældning for f(x), *a,*og skæringspunkt ved y aksen, *b,* er en vis værdi. Se figur 1.3

Figur 1.2: viser en funktion i 3 dimensioner som beskriver fejlværdien*θ*ud ad z-aksen, i en funktion af *a* og *b.*

Algoritmen bruger denne graf til at mindske sin fejlværdi. Det gøres ved at bruge sin første fejlværdi *θ,* og finde differentialet i det punkt. Man går så en vis værdi, *k*, hen af *a* og/eller *b* aksen for at minimere *θ*værdien. Dette gør vi ind til at vi har ramt den mindste *θ*værdien. Denne metode hedder ”gradient decent” og kan visualiseres ved at vi sætter en boldt på vores graf og får bolden til at falde mod bunden af grafen. Der hvor bolden står stille er hvor mindste værdien af *θ*er, der ved de bedste passende *a*og*b* i vores lineær funktion.

Der er dog et problem med gradient decent. Hvis der er et lokalt minimum, kan gradient decent ikke se forskel på det eller det globale minimum. Se figur 1.3. dette kan dog overkommes ved at måle på forskellige *θ*som vores startværdi og derved se om der er andre ’fordybninger’ i vores graf.

Figur 1.3: her ses en 3 dimensional graf for fejlværdien *θ,*ud fra *a* og *b* i en lineær funktion. Der ses et lokalt minimum et sted ved når *a* og *b*er små, mens at det globale minimum er ved en større *a* og *b* værdi.