

Dysgu Peirianyddol

Cyflwyniad i algorithmau dysgu peirianyddol yn R a Python

Alun Owen

B.Sc. Traethawd Blwyddyn 3

Yr Ysgol Mathemateg Caerdydd



Diolchadau

Cynnwys

1	Cyflwyniad	4
1.1	Beth yw Dysgu Peiranyddol	4
1.1.1	Dysgu dan Oruchwyliaeth	5
1.1.2	Dysgu heb Oruchwyliaeth	5
1.1.3	Darllen Pellach	5
1.2	Pam	6
1.2.1	Be sydd yna yn barod?	6
1.2.2	Pam Python ag R?	8
1.2.3	Pam Cymraeg?	8
1.3	Strwythyr	9
2	Clystyru k-cymedr	10
2.1	Cefndir	10
2.2	Sut mae Clystyru K -cymedr yn gweithio?	10
2.2.1	Y Dull	10
2.2.2	Yr Algorithm	12
2.2.3	Sut i ddarganfod y k orau?	12
2.3	Tiwtorial yn R	14
2.4	Tiwtorial yn python	18
3	Atchweliad Logistaidd	22
3.1	Cefndir	22
3.2	Sut mae atchweliad logistaidd yn gweithio?	23
3.2.1	Yr Algorithm	24
3.3	Tiwtorial yn R	26
3.4	Tiwtorial yn Python	28
4	Termau	31

Rhestr Ddarluniau

2.1	Cyn ac ar ôl clystyru k -cymedr.	10
2.2	Enghraifft o set cychwynol ar gyfer clystyru.	11
2.3	Enghraifft o blot o k yn erbyn y cyfanswm swm o sgwariau	13
2.4	Enghraifft o dendogram	13
2.5	Enghraifft o ddata da i cael ei clystyru.	19
2.6	Sut ddylsa eich graff edrych gyda 3 clystrwr.	20
2.7	Sut ddylsa eich graff edrych gyda 6 clystrwr.	21
3.1	Enghraiff o atchweliad logistaidd.	22
3.2	Enghraiff o atchweliad logistaidd gyda labelau.	23
3.3	Enghraiff o atchweliad llinol i ein data.	23

Pennod 1

Cyflwyniad

Mi fyddwn yn ysgrifennu fy mhrosiect blwyddyn ddiwethaf am dan algorithmau dysgu peirianyddol ag sut i eu defnyddio yn R ac Python. Fydd y prosiect yn cael ei ysgrifennu drwy gyfrwn Gymraeg. Mi fyddwn yn creu traethawd a gwefan i dangos y wybodaeth am yr algorithmau. Dwi am ysgrifennu am glystyru *k*-cymedr, atchweliad logistaidd ag , fydd y rhain i gyd yn bennod eu hunain.

1.1 Beth yw Dysgu Peirianyddol

Yn syml, Dysgu Peirianyddol yw rhaglennu cyfrifiaduron i optimeiddio rhyw feini prawf yn defnyddio data. Fyddem gyda model wedi'i ddiffinio o rhai paramedrau mesuradwy. Y darn dysgu o fewn y cyfrifiadur fydd i optimeiddio gyda pharch tuag at y paramedrau hyn. Gall y model fod yn un sy'n rhagfynegi neu yn disgrifiadol o'r data. Mae dysgu peirianyddol yn defnyddio ystadegaeth i adeiladu'r modelau mathemategol, mae cyfrifiadureg yn edrych i mewn mwy ar effeithiolrwydd y proses. [17]

Dydi dysgu peirianyddol ddim yn faes newydd, mae wedi bod o gwmpas ers y 50^{au} pan wnaeth Arthur Samuel o IBM creu rhaglen ar y cyfrifiadur i chwarae'r gem 'checkers'. Yn yr amser hwn cafodd y term ei eni. Yna drwy ddatblygiadau technolegol diweddar, mae'r posibilrwydd o ddysgu peirianyddol bron yn ddiddiwedd. Ers ei sefydlu yn y 50^{au}, mae wedi cymryd tan 1997 i ddatblygu rhaglen i guro'r chwaraewr gwyddbwyll orau yn y byd. Nid yn unig yw dysgu peirianyddol yn cael ei ddefnyddio i greu rhaglenni gemau, mae nawr yn cael ei ddefnyddio yn pob mathau o raglenni.

Mae sylfaeni ddysgu peirianyddol yn cael eu defnyddio yn prosesu iaith naturiol. Mae'r proseses yma yn cael ei ddefnyddio i wneud ffwythiannau fel adnabod lleferydd, creu tecst i leferydd a chyfieithiad peirianyddol i enwi ambell un.

Yn ogystal mae dysgu peirianyddol yn cael ei ddefnyddio i brosesu lluniau, mae'r defnydd yma'n cael ei weld yn aml gyda systemau nabod wynebau.

Yn y cynllun eang o bethau, mae dysgu peirianyddol yn is-set o ddeallusrwydd artiffisial. Mae deallusrwydd artiffisial wedi tyfu yn esbonyddol yn ddiweddar gydag dysgu peirianyddol. Mae deallusrwydd artiffisial yn

y newyddion drwy'r adeg oherwydd datblygiadau parhaus. Yn ddiweddar rydym wedi gweld moduron heb yrrwr, roboteg glyfar i 'chat bots'.

1.1.1 Dysgu dan Oruchwyliaeth

Gadewch barau o bwyntiau (\mathbf{x}_i, y_i) fod yn enghreifftiau a dynodwyd yn ein set ymarfer. Yn y fan hyn mae $y_i \in \alpha$ yn cael ei alw yn labeli, gall y labelau yma fod yn arwahanol neu ddi-dor. Fydd $\mathbf{x}_i \in \beta$ yn fector gyda gwerthoedd ar gyfer priodoleddau gwahanol. Mae'n bwysig i dybio pob pâr wedi'i samplu'n annibynnol a'i dosbarthu o ddosraniad dros $\alpha \times \beta$. Y nod ar gyfer dysgu dan oruchwyliaeth yw dysgu'r map o \mathbf{x} i y gan ddefnyddio'r set ymarfer. [20]

Mae'r ffurf wahanol o ddysgu dan oruchwyliaeth yn cael ei rhannu i ddau faes yn ôl sut fath o ddata sydd gennym. Os yw'r data yn arwahanol, mae gennym ddosbarthiad. Os mae'r data yn ddi-dor, mae gennym atchweliad. Mae yna lwyth o wahanol fathau o ddosbarthiadau ag atchweliadau; dyma ambell o enghreifftiau ohonyn:

Dosbarthiad:

- Coeden penderfyniadau
- Dosbarthiad naïf Bayes
- K cymydog agosaf

Atchweliad:

- Atchweliad logistaidd
- Atchweliad llinol
- Atchweliad Poisson

1.1.2 Dysgu heb Oruchwyliaeth

Gadewch $X = (x_1, \dots, x_n)$ dynodi n enghreifftiau lle mae $x_i \in \gamma$ ag $i \in \{1, \dots, n\}$. Ar gyfer dysgu heb oruchwyliaeth fyddem yn tybio fod yr enghreifftiau x_i wedi'i samplu'n annibynnol a'i dosbarthu o ddosraniad unfath ar γ . Y nod o ddysgu heb oruchwyliaeth yw amcangyfrif dwysedd o'r dosraniad ar γ . [20]

Mae'r fatha boblogaidd o ddysgu heb oruchwyliaeth yn cymryd ffurf wannach o'r syniad hyn. Dyma ambell i enghraifft ar ffurfiau gwahanol o ddysgu heb oruchwyliaeth:

- Clystyru
- Lleihad dimensiwn
- Model Markov cudd

1.1.3 Darllen Pellach

Mae dysgu dan oruchwyliaeth rannol yn cymryd priodweddau o'r ddwy ffordd o ddysgu. Y ffurf fwyaf traddodiadol o hyn yw cael data wedi'i labelu a data heb ei labelu ac yno dilyn proses hunan ddysgu.

1.2 Pam

1.2.1 Be sydd yna yn barod?

Adnodd	Awddur	Fformat	Cynulleidfa Darged	Linc
Cwrs Cyfrifiadureg ar gyfer Mathemateg yn Python	Dr Vince Knight a Dr Geraint Palmer	Tiwtorial ar wefan	Dechreuwr mewn codio Python	[10]
Cyrsiau allgyrsiol yn Scratch, HTML, CSS ag Python	Code Club	Tiwtorialau ar y wê	Plant rhwng 9 ac 13	[4]
Adnoddau ar lawer o testynau wahanol yn cyfrifiadureg	Technocamps	Mewn ffurdd pdf	Plant yn ysgol/coleg	[6]
Cwrs cychwynol yn Python	Dr Geraint Palmer	Tiwtorialau ar y wê	Dechreuwr mewn codio Python	[11]
Gwybodath ar gydrannau pwysicaf technolegau iaith	Uned technoleg iaith Prifysgol Bangor a Cymen Cyf.	Llawlyfr ar y wê	Myfyrwyr, datblygwyr neu academyddion heb gefndir yn y maes	[7]
ap Botio i hybu rhaglennu i plant	Tinopolis	ap ar cynnyrch apple	Plant rhwng 7 ac 11	[8]
Enghreifftiau a thiwtorialau o cynnyrch technolegau iaith	Uned technoleg iaith Prifysgol Bangor a Cymen Cyf.	Ystorfa ar github	Myfyrwyr, datblygwyr neu academyddion heb gefndir yn y maes	[16]
Tiwtorialau Python	Dr Geraint Palmer a Stephanie Jones	Fideos ar Youtube	Dechreuwr yn rhaglennu gyda Python	[15]
Tiwtorialau SQL	pl/sql tiwtorial	Wefan gyda tiwtorialau	Dechreuwr yn SQL	[14]
Nodiadau agored am algebra llinol	Dr Alun Morris	Nodiadau (pdf)	Myfyrwyr prifysgol	[19]
Amrhyw o adnoddau Mathemategol yn dilyn cwricwla CBAC	Dr Gareth Evans	Wefan gyda linciau i wahanol fathau o adnoddau	Myfyrwyr rhwng 11 a 18	[5]

Mae'r tabl uchod wedi'i rhannu i adnoddau codio Cymraeg ag i adnoddau Mathemategol Cymraeg. Fel

gwelwn yn ebrwydd, mae'r nifer o adnoddau codio yn llawer fwy nag rheina o adnoddau mathemategol. Mae'r adnodd [10] yn cychwyn da i unrhyw un sydd gyda diddordeb o gychwyn codio yn Python, mae'n defnydd gwych ar gyfer myfyrwyr Mathemateg israddedig gan ei fod yn benodol i Fathemateg. Yn ogystal i'r cwrs yma mae gennym hefyd bach o gyflwyniad i Python ar y wefan sgiliau ymchwil ailhynhyrchiadwy [11].

Gwelwn yn ogystal fod yna adnoddau ar gyfer rhaglennu yn Python yn Gymraeg ar gael ar YouTube. Ar sianel Geraint Palmer mae yna gasgliad o diwtorialau Python. Mae'r tiwtorialau yn cychwyn gyda'r sylfaen o raglennu yn Python ag yn gorffen gyda mynd dros yr Algorithm Genetig sy'n mynd law yn llaw gyda fy mhrosiect.

Mae'r wefan codeclub gydag amrywiaeth eang o adnoddau i godio. Mae yna gyrsiau ar HTML, CSS, Python a Scratch. Yn ogystal mae yna brosiectau pellach sy'n gweithredu raspberry pi.

Gwelwn gydag adnodd technocamps [6], fod yna amrywiaeth eang o adnoddau yn fan hyn fyd. Mae'r wefan yn wedi'i thargedu i oedran hyn nag y wefan codeclub. Mae technocamps gyda chysiau cychwynnol fel y canlynol:

1. CS 101
2. Deallusrwydd Artiffisial
3. Greenfoot (Java)
4. Python
5. Scratch

Ar wefan technolegau iaith [7], mae yna gyflwyniad i'r darnau fwyaf sylfaenol i dechnolegau iaith. Mae lawlyfr yn cynnwys gwybodaeth am ddeallusrwydd artiffisial, dysgu dwfn a phrosesu iaith naturiol. Yn rhedeg yn gyfagos i'r wefan yma yw'r ystorfeydd ar github [16]. Yn yr ystorfeydd mae yna diwtorial ar sut i greu robot sgwrsio drwy "turing test lessons".

Yn y siop ap apple, fedrem lawrlwytho'r ap botio. Mae'n ap ar gyfer plentyn gyda diddordeb cael i mewn i rhaglennu.

Mae'r adnodd am ddim oddiwrth Alun Morris [19] yn trafod y sylfeini eu hangen i astudio algebra llinol yn mhrifysgol. Mae'r adnodd yma yn enghraifft wych o'r fath o adnoddau mae'r iaith Gymraeg angen fwy ohono. Mae yna brinder iawn ar adnoddau o'r ansawdd yma i'r lefel yma o addysg.

Mae yna gyfoeth o adnoddau mathemateg Cymraeg ar gyfer addysg o dan 18 oed, gwelwn hyn gyda'r nifer mawr o adnoddau ar wefan Dr Gareth Evans o Ysgol Creuddyn [5].

Gwelwn fod yna llawer o adnoddau ar gyfer y maes Mathemateg a Chyfrifiadureg i addysg ysgol. Mae yna ddigon o adnoddau i alluogi cenhedlaeth newydd i astudio'r sylfaen gofynnol drwy'r ysgol i astudio unrhyw un o'r ddau yn brifysgol. Unwaith fyddem yn cyrraedd lefel addysg prifysgol, mae'r prinder yn amlwg. Gan fod testunau cymhleth dysgu peiranyddol yn cynnwys Mathemateg gymhleth yn ei hun, mae rhaid cael sylfaen datblygedig o destunau Mathemateg i'w ddeall. Gallwn ddweud yr un peth am dan yr angenrheidrwydd o

sylfaen datblygedig o gyfrifiadureg. Y broblem sydd gennym yw does yna ddim yr adnoddau yw gwneud gyda Mathemateg ag rhaglennu yn R.

1.2.2 Pam Python ag R?

Fydd y tiwtorialau i ddilyn yn cael ei ddangos drwy'r ieithoedd rhaglennu Python ag R, fydd hyn oherwydd nhw yw'r ddwy iaith fwyaf poblogaidd yn dysgu peirianyddol a gwyddor data yn ôl arolwg Kaggle nol yn 2018 [1].

Rheswm arall i ddysgu defnyddio'r ieithoedd rhaglennu yma yw bod nhw'n god agored sy'n wahanol i SQL er enghraifft. Gan ei fod nhw'n god agored poblogaidd, mae'n meddwl fod nhw'n cael ei diweddarau yn aml.

Cafodd R ei greu gan ystadegwyr a gan fy mod yn creu tiwtorialau am ddysgu peirianyddol o'r ochr ystadeg-aeth, mae'n gymhelliad i ddefnyddio R. Ar gyfer dysgu Python, nol yn 2019 roedd Python yr iaith rhaglennu ail fwyaf i gael "pull requests" ar github [9], mae github y gwesteigr fwyaf o god ffynhonnell yn y byd.

Mae Python ag R yn cael ei ystyried i fod yn hawdd dysgu a deall yn gymhariaeth gydag ieithoedd rhaglennu eraill. Gwelwn hyn i fod yn wir yn y tiwtorialau i ddod.

1.2.3 Pam Cymraeg?

I gychwyn, rwyf eisiau creu gwefan sy'n cynnwys tiwtorialau Cymraeg am algorithmau dysgu peirianyddol oherwydd y prinder ohonyn. Mae yna gymaint o adnoddau ar gyfer pob mathau o algorithmau drwy Saesneg ond yn Gymraeg, does yna ddim byd o'r fath! Fel gwelwn yn y tabl o adnoddau cynnar, does yna ddim llawer o adnoddau Cymraeg sy'n cynnwys deunydd datblygedig ar gael i'r cyhoedd. O fy mhrofiad personol i, rwyf yn gwybod fod prifysgolion yng Nghymru gydag adnoddau drwy'r cyfrwng Cymraeg ar gael i'w myfyrwyr nhw yn unig. Teimlaf fod hyn yn atal y parhad o addysg ar ôl i fyfyrwyr gorffen eu hacademiau gan fod adnoddau am ddim, ddim ar gael.

Ers i'r iaith Gymraeg cael ei rhoi ar sylfaen gyfartal i Saesneg yn Gymru yn 1993 a chael gwasanaethau cyhoeddus yn ddwy iaith, mae yna wedi bod effaith domino i'r sector preifat. Mae prifysgolion yn gyrff preifat ac felly dydi ddim yn angenrheidiol iddyn nhw ddilyn [2], ond yn 2011 cafodd y coleg Cymraeg ei sefydlu i ornest a hyn. Mae'r coleg yn hybu prifysgolion Cymru i newid i fod yn ddwyieithog drwy weithio yn bartneriaeth gyda nhw. Gwelwn y datblygiad yma yn fy amser i ym mhrifysgol. Drwy Haf 2019 mi oeddwn i yn ddarn o'r prosiect i gyfieithu nodiadau darlith modiwl algebra llinol a thebygolrwydd, dangosaf hyn fod yna waith yn cael ei wneud o fewn prifysgolion i addasu'r holl waith i gyfrwng y Gymraeg. Ond yn bolisi prifysgol Gaerdydd ar wasanaethau'r iaith Gymraeg, gwelwn fod nhw wedi'i esgusodi o orfod creu adnoddau Cymraeg ar gyfer unrhyw fath o adnoddau dysgu[?]. Teimlaf fod yr Ysgol Fathemateg o'r prifysgol dim ond wedi cyfieithu digon o fodiwlau i alluogi cynnal y cwrs drwy ysgoloriaeth Coleg Cymraeg. os fyswn wedi bod eisiau gwneud fy nghwrs drwy'r cynllun yma fyswn heb gael yr ehangder o ddewis rwyf wedi cael drwy fy ngradd.

Rheswm arall i wneud yn Gymraeg yw cefnogi'r prosiect o'r llywodraeth a'r wlad i gael miliwn o siaradwyr Cymraeg erbyn 2050 [12]. I gymhorthi'r prosiect yma cafodd cynllun ei rhoi allan yn hybu'r defnydd a

chynnwyrch o adnoddau technolegol Cymraeg [13], yn ogystal i greu adnoddau mae'r cynllun yn targedu y datblygiad o ddeallusrwydd artifisial drwy edrych ar brosesu iaith naturiol yn bennaf. Mae creu'r adnodd yma yn Gymraeg yn bodloni'r ddwy adran o'r cynllun, gobeithio bydd yn hybu fwy i gychwyn creu adnoddau addysg uwch yn y maes yma. Credaf fod rhyw fath o ateb cyflym i'r prinder fysa os fysa'r prifysgolion yn gadael ei adnoddau allan am ddim ar borth coleg Cymraeg, fel bod rhai prifysgolion yn yr Unol Daleithiau America lle mae yna ystorfa o gyrsiau gydag adnoddau agored ar wefan Coursera [3].

1.3 Strwythyr

Fydd y traethawd yma ymlaen yn cynnwys tair pennod ar algorithm dysgu peirianyddol. Yn pob pennod fydd yna gefndir ar beth yw'r algorithm a pryd fyddem yn ei ddefnyddio, yn ogystal fydd yna gefndir ar sut mae'r algorithm yn gweithio a beth yw'r Fathemateg tŷ ôl i'r algorithm. Yna fydd yna diwtorial o sut yw defnyddio yn R ac yna tiwtorial arall yn Python.

Ar gyfer y wefan, fydd yna dudalen cartref lle fydd yno linc i bob tiwtorialau yn y ddau Python ag R. Fydd y wefan yn cynnwys y tiwtorialau ag y wybodaeth angenrheidiol i allu gwneud yr algorithmau yn yr iaith rhaglennu o eich dewis.

Pennod 2

Clystyru k -cymedr

2.1 Cefndir

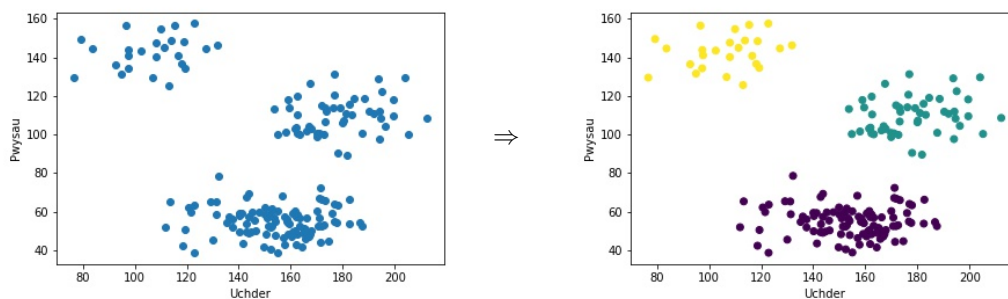
Mae clystyru k -cymedr yn ffordd o ddysgu heb oruchwyliaeth, mae'n cymryd data heb ei labelu ac yn eu sortio i mewn i k wahanol glwstwr yn yr obaith i ddarganfod rhyw strwythur doedden ddim yn gwybod yn gynharach.

I roi enghraifft gwelwch Ddarlun 2.1. Mae'r gwerthoedd ar echelin x yn cynrychioli uchder rhyw berson a'r llall yn cynrychioli pwysau'r person. Fel gwelwn yn y llun ar y chwith gallwn weld tri grŵp naturiol wedi'i ffurfio. Rydym nawr eisiau eu grwpio yn ffurf Fathemategol. Mae clystyru k -cymedr yn medru dosrannu'r tri grŵp fel gwelwn ar ochr dde'r darlun.

2.2 Sut mae Clystyru K -cymedr yn gweithio?

2.2.1 Y Dull

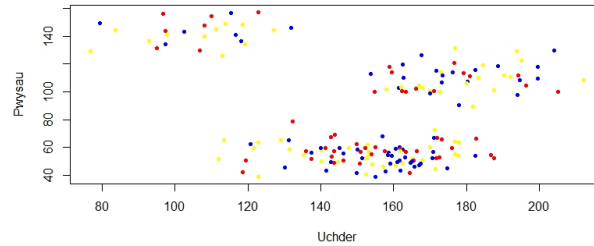
Mae clystyru k -cymedr yn syml, mae ond yn dilyn pedwar cam [21]. I wneud yn siŵr fod yn ei ffurf fwyaf cyntefig, fyddan yn defnyddio mesur pellter Ewclidaidd. Yn ogystal mae rhaid dewis k cyn cychwyn y



Darlun 2.1: Cyn ac ar ôl clystyru k -cymedr.

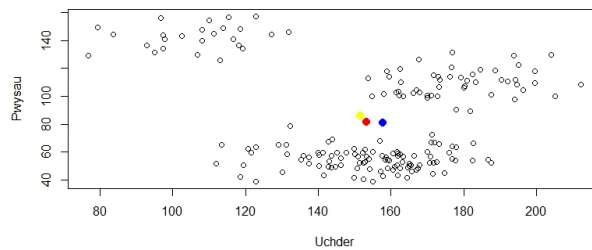
proses. Mae'n bosib optimeiddio'r dewis o k , a gwnawn drafod hyn hwyrach ymlaen. Dyma bedwar cam yr algorithm a sut maent yn edrych pan fyddwn ni'n defnyddio'r algorithm ar y data y gwelwn yn 2.1:

1. Aseinio pob elfen i un o'r k clystyrau ar hap.

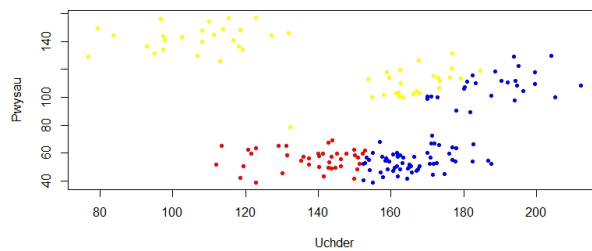


Darlun 2.2: Enghraifft o set cychwynol ar gyfer clystyru.

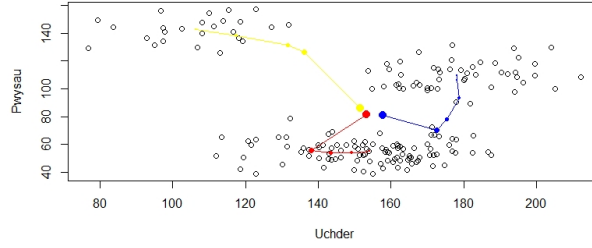
2. Cyfrifo canolbwynt (hynny yw craidd) pob clwstwr.



3. Ail-aseinio pob elfen unwaith eto i'r clwstwr gyda craidd agosaf.



4. Ailadrodd camau dau a tri tan fod y creiddiau ddim yn symud rhagor.



2.2.2 Yr Algorithm

Diffiniwn bob clwstwr rydym yn ceisio darganfod fel C_i lle bydd $i \in \{1, 2, \dots, k\}$, mae gennym hefyd n pwyntiau data x_1, x_2, \dots, x_n . Gadewch i \mathbf{c}_i bod yn bwynt sy'n graidd i clwstwr C_i . Ar gyfer y cam cyntaf angen aseinio pob \mathbf{x}_j i ryw glwstwr C_i ar hap. Yna gan ein bod yn datgelu ein bod yn delio gyda phlân Ewclidaidd, mi fyddem yn darganfod craidd pob clwstwr gan y fformiwla ganlynol:

$$\mathbf{c}_i = \frac{1}{|S_i|} \sum_{\mathbf{x}_j \in S_i} \mathbf{x}_j \quad (2.1)$$

Yn y fformiwla uchod, gwelwn fod fectorau yn cael eu symio. Fyddem yn gwneud hyn gan symio dros elfennau.¹

lle diffiniwn S_i fel y set o bwyntiau data sydd wedi'i aseinio i glwstwr C_i .

Nawr mae gan bob clwstwr craidd newydd, fedrwn aseinio pob pwynt data i'r craidd agosaf. Caiff hyn ei gwneud gan fynd drwy bob pwynt data a chyfrifo'r pellter Ewclidaidd² i bob canolbwynt. Yna fydd y pwynt priodol yn cael ei labelu gyda'r clwstwr sydd a'r pellter lleiaf o'i graidd i'r pwynt data. Hynny yw

$$\arg \min_{\mathbf{c}_i} \|\mathbf{c}_i, \mathbf{x}_j\|^2 \quad (2.2)$$

Unwaith mae'r proses wedi'i chychwyn, angen ailadrodd y darn o ddarganfod y creiddiau newydd ac ail labelu'r pwyntiau data tan fod y creiddiau ddim yn symyd rhagor.

2.2.3 Sut i ddarganfod y k orau?

Mae yna wahanol ffurf i ddarganfod k , edrychwn ar ddau wahanol ffordd o wneud hyn.

Dull Penelin

Mae'r dull penelin yn cymharu'r cyfanswm o swm sgwariau o fewn y clystyrau. Unwaith gennym y cyfanswm o swm sgwariau o fewn clystyrau i bob k rydym eisiau cymharu, fyddem yn creu plot o bob k yn erbyn y

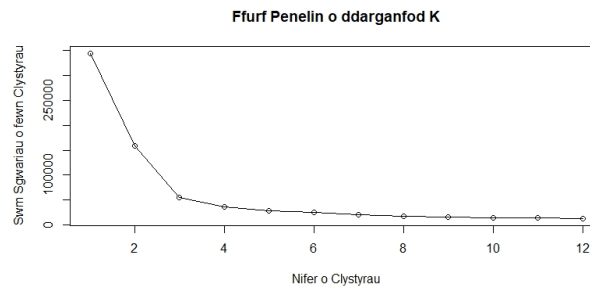
¹ $(x_1, x_2, \dots, x_n) + (y_1, y_2, \dots, y_n) = (x_1 + y_1, x_2 + y_2, \dots, x_n + y_n)$

² $\|\mathbf{p} - \mathbf{q}\| = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2}$

cyfanswm o swm sgwariau o fewn y clystyrau ar gyfer y k hynny. Fydd swm sgwariau ar gyfer clwstwr C_i yn cael ei darganfod gan symio y pellter rhwng y craidd \mathbf{c}_i a phob \mathbf{x}_j yn ei tro ag yno ei sgwario fel welwn yn y fformiwla:

$$SS_i = \sum_{j=1}^n (\mathbf{x}_j - \mathbf{c}_i)^2$$

Unwaith mae gennym y graff, allwn ei ddadansoddi.

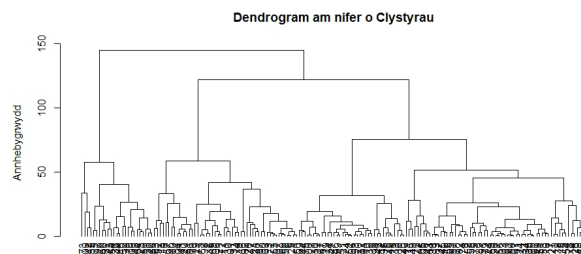


Darlun 2.3: Enghraifft o blot o k yn erbyn y cyfanswm swm o sgwariau

Yn y graff yn Darlun 2.3, gwelwn fod swm sgwariau yn fawr yn cychwyn gyda $k=1$ sydd yn gwneud synnwyr. O'r pwynt yma wedyn fydd yna newid mawr yn y swm sgwariau. Unwaith mae'r newid mawr hwn yn dod i ben fydd gennym ongl yn cael ei greu lle bydd newid k dim ond yn creu newid bach. Y pwynt yma fydd yr optimwm ar gyfer nifer k o glystyrau. Fel gwelwn yn glir yn ein henghraifft ni, mae'n glir fod $K=3$ yw dewis orau ar K .

Dendrogram

Mae dendrogram yn ffordd wahanol iawn i canfod y nifer orau k o glystyrau. Mae'n defnyddio darn o glystyru hierarchaidd i greu diagram canghennog. Mae'r echelin llorweddol yn dangos pob gwrthrych yn ein set o ddata. Mae'r echelin fertigol yn dangos mesur o annhebygrwydd. Mae Darlun 2.4 yn dangos dendrogram ar gyfer yr un data.



Darlun 2.4: Enghraifft o dendrogram

I ddadansoddi'r dendrogram mi fyddwn edrych yn bennaf ar yr echelin fertigol. Edrychwn allan am yr annhebygrwydd fwyaf rhwng cyflwyniad o gangen arall yn y goeden. Welwn ni hyn yn ein henghraifft ni ar ôl i'r drydydd clwstwr cael ei gyflwyno yn dendrogram. Mae hyn yn datganu'r un peth a'r dull penelin.

2.3 Tiwtorial yn R

Mi fyddwn yn edrych ar ddata o uchder a phwysau 175 wahanol berson. Mi allwch chi lawrlwytho y data yma o fan hyn: <https://dysgupeirianyddol.github.io/>

Yno fydd angen lawrlwytho a gosod y pecynnau `graphics`, `stats` ag `datasets` ar eich cyfrifiadur. Ffordd hawdd i wirio hyn fydd i ddefnyddio'r côd canlynol:

```
> install.packages("graphics")
> install.packages("stats")
> install.packages("datasets")
> library(graphics)
> library(stats)
> library(datasets)
```

Mae'r darn gyntaf o'r côd uchod yn gosod/diweddaru'r pecynnau angenrheidiol. Mae'r ail ddarn yn llwytho'r pecynnau i ein sesiwn ni.

Nawr mi wnawn lwytho'r data.

```
> uchderpwysau <- read.csv("C:/Users/User/Desktop/Dysgu_Peirianyddol/heightvsweight.csv")
> View(uchderpwysau)
```

Mae'r string sydd mewnbyn y ffwythiant `read.csv` yn cyfeirio at y lleoliad ar ein cyfrifiadur lle gallwn ganfod y ffeil csv priodol. Rhaid gwneud yn siŵr eich bod yn defnyddio'r lleoliad cywir i'r lleoliad o'ch ffeil chi. Ar ôl rhedeg y côd ddylai eich data edrych yn debyg i'r canlynol:

	Uchder	Pwysau
1	163.22687	100.09760
2	183.18087	110.18107
3	172.69407	99.79701
4	165.07549	51.66760
5	147.74605	59.79469
6	161.45039	103.04177
7	162.41267	58.50832
8	146.28025	50.36660
9	154.03614	47.93155
10	152.20904	59.79795

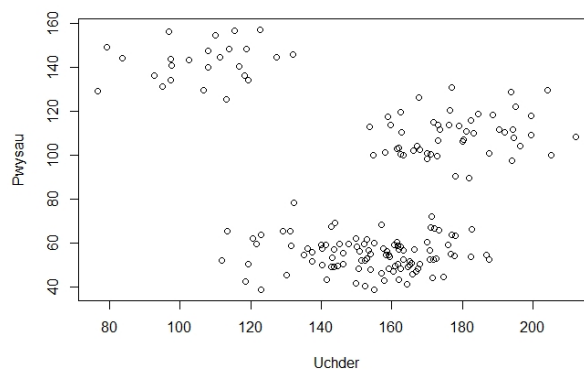
Gan fod y data hefo enwau ar gyfer y colofnau, gallwn atodi'r data i lwybr chwilio R. Bydd hyn yn gadael i ni gyfeirio at enwau colofnau'r data yn ein côd fydd yn gwneud yn lawer mwy symlach i ddeall.

```
> attach(uchderpwysau)
```

I wneud fwy o synnwyr o'r data, mi wnawn blotio'r data.

```
> plot(Uchder, Pwysau, pch = 21)
```

Sy'n rhoi:



Gwelwn fod yna 3 clwstwr clir.

Rŵan rydym yn gallu tybio fod y data yn gallu cael i rannu i dri chlwstwr gwahanol, mi wnawn ddefnyddio'r algorithm dysgu peiranyddol i'w ddehongli. Rhedwn y canlynol i redeg clystyru k -cymedr yn R. Rydym yn

defnyddio'r ymresymiad `nstart` i ddewis faint o setiau ar hap o ddata wedi'i labelu wnawn gymered. Welwn enghraifft o'r set ar hap hyn yn Darlun 1. Rydym yn neud hyn i wneud yn fwy debygol i ni ddarganfod yr uchafbwynt eang, mae hyn oherwydd mae yna gymaint o uchafbwyntiau lleol.

```
> kcyedr <- kmeans(uchderpwysau,3, nstart = 50)
```

Allwn nawr adio colofn newydd i'r data sef y clystyrau newydd mae'r algorithm wedi'i darganfod.

```
> uchderpwysau$Clwstr3 <- kcyedr$cluster
```

Gallwn weld y newid hwn gan ddefnyddio'r un côd a ddefnyddion yn gynharach.

```
> View(uchderpwysau)
```

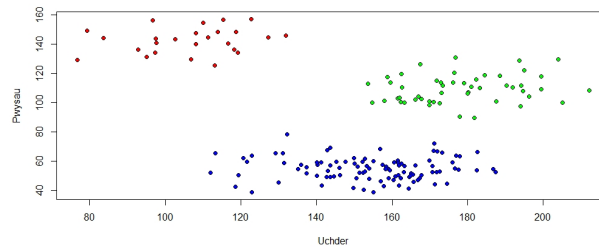
	Uchder	Pwysau	Clwstr3
1	163.22687	100.09760	1
2	183.18087	110.18107	1
3	172.69407	99.79701	1
4	165.07549	51.66760	2
5	147.74605	59.79469	2
6	161.45039	103.04177	1
7	162.41267	58.50832	2
8	146.28025	50.36660	2
9	154.03614	47.93155	2
10	152.20904	59.79795	2

Mae'n bosib fydd yr algorithm wedi labeli'r clystyrau gwahanol gyda rhifau gwahanol i'r hyn a welwch fan hyn, ddylai'r clystyrau ei hun fod yn hafal. Mae hyn oherwydd y setiau ar hap cychwynnol mae'r algorithm yn ei gymered i gychwyn.

Rhedwn y côd canlynol liwio'r clystyrau newydd ar graff.

```
> plot(Uchder, Pwysau, pch = 21, bg=c("red","green","blue")[unclass(kcyedr$cluster)])
```

Sy'n rhoi:



I gymharu, nawr mi nawn rhedeg yr algorithm ar gyfer 6 clwstwr i weld y clystyrau pan fydd $k = 6$.

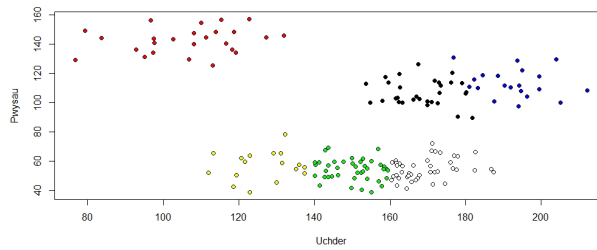
```
> kcymedr <- kmeans(uchderpwysau,6, nstart = 50)
> uchderpwysau$Clwstwr6 <- kcymedr$cluster
> View(uchderpwysau)
```

	Uchder	Pwysau	Clwstwr3	Clwstwr6
1	163.22687	100.09760	1	1
2	183.18087	110.18107	1	6
3	172.69407	99.79701	1	1
4	165.07549	51.66760	2	4
5	147.74605	59.79469	2	2
6	161.45039	103.04177	1	1
7	162.41267	58.50832	2	4
8	146.28025	50.36660	2	2
9	154.03614	47.93155	2	2
10	152.20904	59.79795	2	2

Gwelwn fod y labeli newydd wedi cael ei ychwanegu i'n tabl. Yna gan blotio graff arall, fedrem weld y 6 clwstwr yn gliriach.

```
> lliwiau <- c("red","green","blue", "yellow", "black", "white")
> plot(Uchder, Pwysau, pch = 21, bg=lliwiau[unclass(kcymedr$cluster)])
```

Sy'n rhoi:



2.4 Tiwtorial yn python

Yn y tiwtorial hwn mi wnawn edrych ar yr un data a welom yn y tiwtorial diwethaf. I gychwyn bydd rhaid llwytho'r pecynnau `pandas`, `matplotlib.pyplot` ag `sklearn.cluster` drwy redeg y côd canlynol:

```
>>> import pandas as pd
>>> import matplotlib.pyplot as plt
>>> import sklearn.cluster
```

Y rŵan mi wnawn lwytho'r data i mewn i'n gwaith gan redeg y côd:

```
>>> data = pd.read_csv('heightsweight.csv')
```

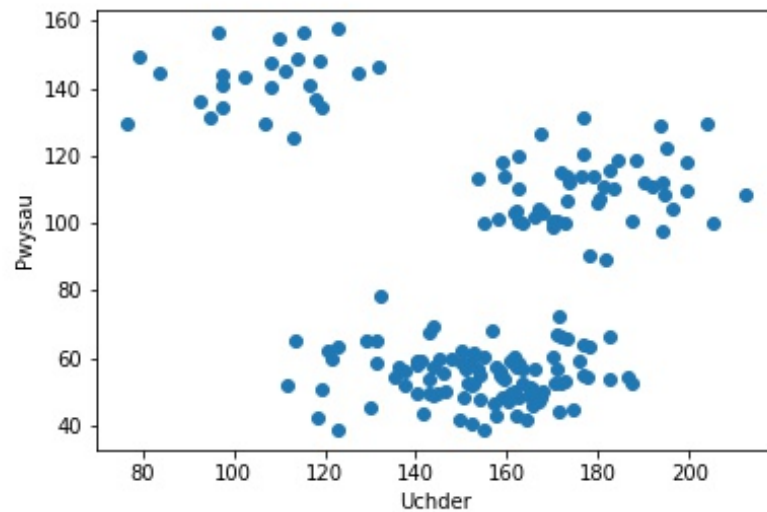
Mae'r string sydd mewnbyn y ffwythiant `pd.read_csv` yn cyfeirio at y lleoliad ar ein cyfrifiadur lle gallwn ganfod y ffeil csv priodol. Rhaid gwneud yn siŵr eich bod yn defnyddio'r lleoliad cywir i'r lleoliad o'ch ffeil chi. Unwaith fydd wedi cael ei llwytho, allwn ni gweld yn fras y data gennym ni.

```
data.head()
```

	Uchder	Pwysau
0	163.226866	100.097603
1	183.180871	110.181072
2	172.694074	99.797013
3	165.075492	51.667604
4	147.746048	59.794691

I weld y data mewn ffordd fwy gweledol, wnawn blotio graff gwasgariad o'r data.

```
>>> plt.scatter(data['Uchder'], data['Pwysau']);
>>> plt.xlabel('Uchder')
>>> plt.ylabel('Pwysau')
>>> plt.show()
```



Darlun 2.5: Enghraifft o ddata da i cael ei clystyru.

Fel gwelwn, mae'r data yn edrych fel ei fod mewn tri chlwstwr. Felly wnawn ddefnyddio'r ffurf algorithm dysgu peiranyddol i'w labelu.

```
>>> kmeans = sklearn.cluster.KMeans(n_clusters=3).fit(data)
>>> data['Cluster (k=3)'] = kmeans.predict(data)
```

Gallwn weld y newid hwn gan ddefnyddio'r un côd a ddefnyddion yn gynharach.

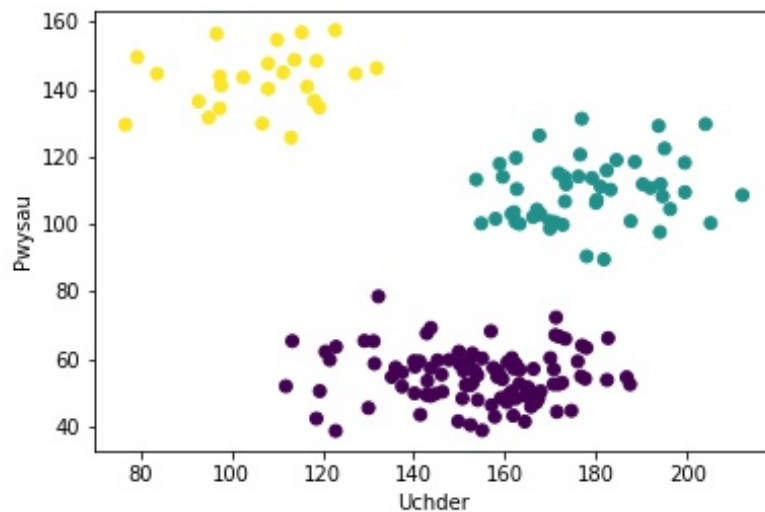
```
>>> data.head()
```

	Uchder	Pwysau	Cluster (k=3)
0	163.226866	100.097603	1
1	183.180871	110.181072	1
2	172.694074	99.797013	1
3	165.075492	51.667604	0
4	147.746048	59.794691	0

Fel y gwelwyd, mae'r data wedi'i rhoi i mewn i dri chlwstwr ac wedi'i labelu gyda rhif y clwstwr. Gan fod pob pwynt yn y data nawr gyda label, allwn ni creu'r plot eto ond gyda bob clwstwr yn lliw gwahanol.

```
>>> plt.scatter(data['Uchder'], data['Pwysau'], c=data['Cluster (k=3)']);  
>>> plt.xlabel('Uchder')  
>>> plt.ylabel('Pwysau')  
>>> plt.show()
```

Sy'n rhoi:



Darlun 2.6: Sut ddylsa eich graff edrych gyda 3 clystwr.

Fel y gwelwn, gweithiodd yr algorithm yn wych. Wnawn nawr trio clystyru k -cymedr gyda k yn hafal i 6.

```
>>> kmeans = sklearn.cluster.KMeans(n_clusters=6).fit(data)  
>>> data['Cluster (k=6)'] = kmeans.predict(data)
```

Sy'n rhoi:

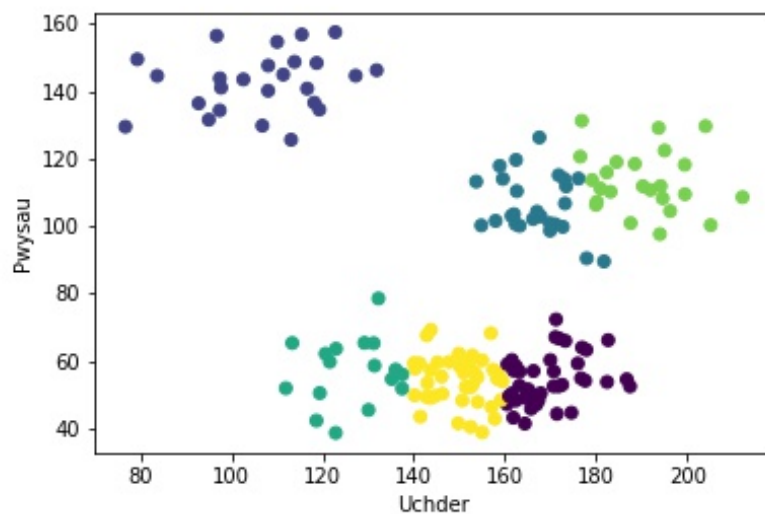
```
>>> data.head()
```

	Uchder	Pwysau	Cluster (k=3)	Cluster (k=6)
0	163.226866	100.097603	1	4
1	183.180871	110.181072	1	2
2	172.694074	99.797013	1	4
3	165.075492	51.667604	0	0
4	147.746048	59.794691	0	3

Gallwn hefyd gweld canlyniad rhoi'r data i mewn i 6 clwstwr gwahanol:

```
>>> plt.scatter(data['Uchder'], data['Pwysau'], c=data['Cluster (k=6)']);
>>> plt.xlabel('Uchder')
>>> plt.ylabel('Pwysau')
>>> plt.show()
```

Sy'n rhoi:



Darlun 2.7: Sut ddylsa eich graff edrych gyda 6 clystwr.

Dyma sut dylaf eich data edrych fel ar ôl a phrosesu drwy glystyru 6-cymedr.

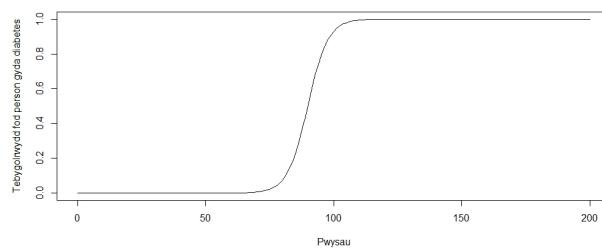
Pennod 3

Atchweliad Logistaidd

3.1 Cefndir

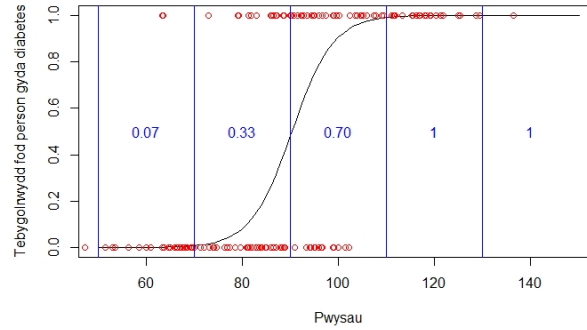
Defnyddiwn atchweliad logistaidd i fodelu'r tebygolrwydd o ddosbarthu gwrthrych i mewn i setiau deuaidd. Mae'n ddull o ddysgu dan oruchwyliaeth sy'n cael ei ddefnyddio yn aml yn academïau a diwydiannau. Gall y atchweliad cael ei ddefnyddio i weld os mae rhywun yn curo/colli, sâl/iachus neu basio/methu mewn rhyw sefyllfa benodol. Gall y syniad yma cael ei ymestyn, gall wahanol atchweliadau logistaidd cael ei rhoi yn baralel i geisio rhoi'r tebygolrwydd o liw llygaid rhyw berson er enghraifft. Mewn termau mwy cyffredinol, gall ymestyn atchweliadau logistaidd i weithio ar setiau o labeli di-deuaidd. O hyn ymlaen fyddem yn edrych ar atchweliadau gyda labeli deuaidd.

Mae'n hawdd delweddu sut fydd atchweliad logistaidd gydag un newidyn annibynnol. Gwelwn fod y model yn edrych fel y graff yn ddarlun 3.1 pan hyn yw'r sefyllfa.



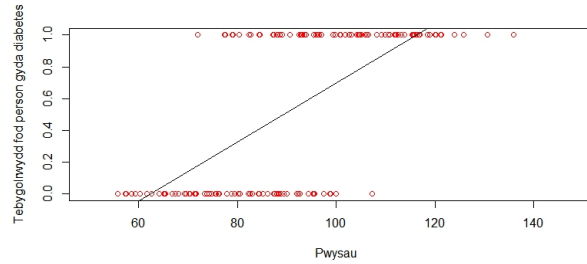
Darlun 3.1: Enghraifft o atchweliad logistaidd.

Gwelwn yn y graff nesaf fod y plot yn dangos ein data mewn ffordd rhesymol os wnawn gymharu i y cyfrannau o'r pwyntiau yn pob un o'r adrannau. Gwelwn y cyfrannau o phob adran yn glas yn ffigur 3.1.



Darlun 3.2: Enghraiff o atchweliad logistaidd gyda labelau.

Os wnawn cymharu y model logistaidd i model llinol, er fod yn bosib cael model gwell yn llinol, mae'n bwysig gwybod pa parth mae'r model angen cael cynhaliad ynddo. Gwelwn yn y plot yn ffigwr 3.1 fod dydi y model ddim yn cynnal cynhaliad ar gyfer mewnbwn llai na 60 a fwy na 120 gan fod y tebygolrwydd yn annifffiniedig (Hynny yw, dydi ddim yn bosib cael $P(\mathbf{x}) < 0$ a $P(\mathbf{x}) > 1$).



Darlun 3.3: Enghraiff o atchweliad llinol i ein data.

3.2 Sut mae atchweliad logistaidd yn gweithio?

Wnawn ddiffinio'r fector sy'n cynnwys gwybodaeth am berson j ($j \in 1, \dots, n$) gyda \mathbf{x}_j sydd hefo dimensiwn o m (hynny yw bod yna m priodweddau). Yn ogystal, wnawn ddiffinio y_j fel label deuaidd i berson j , yr hyn rydyn ni eisiau rhagfynegi. Yna gydag ein data fydd rhaid i ni wahanu'r data i mewn i ddata ymarfer ac ddata profi. Fydd hyn yn cael ei wneud ar hap. Felly fydd gennym:

Data ymarfer: \mathbf{x}_j a y_j ar gyfer $j \in \{1, \dots, k\}$ lle mae $k < n$

Data profi: \mathbf{x}_j a y_j ar gyfer $j \in \{k+1, \dots, n\}$

Nawr wnawn edrych ar y ffwythiant logistaidd, lle mae $z \in (-\infty, \infty)$:

$$f(z) = \frac{1}{1 + e^{-z}} \quad (3.1)$$

Mae'r model logistaidd yn cymryd y gwrthdro o ffurf logit, mae hyn yn cael ei ddangos yn hafaliad ??.

$$z = \alpha + \beta_1 X_1 + \dots + \beta_m X_m \quad (3.2)$$

Felly mae hafaliad 3.3 yn dangos y model cyfan.

$$P(\mathbf{x}) = P(y = 1 | x_1 \dots x_k) = \frac{1}{1 + e^{-(\alpha + \sum_{i=1}^m \beta_i x_i)}} \quad (3.3)$$

3.2.1 Yr Algorithm

[18] Fydd α a β y paramedrau fyddem yn trio amcangyfrif o wybod \mathbf{x} ac y y data ymarfer. I amcangyfrif hyn wnawn ddefnyddio'r dull amcangyfrif tebygoliaeth fwyaf. Cymerwn $\hat{\mathbf{z}}$ i fod y fector o baramedrau fyddem yn amcangyfrif. Yna mae gennym y amcangyfrif tebygoliaeth ganlynol a fyddem yn trio cael y gwerth agosaf i 1.

$$L(\hat{\mathbf{z}}) = \prod_{s \in y_i=1} p(x_i) \prod_{s \in y_i=0} (1 - p(x_i)) \quad (3.4)$$

Mae Hafaliad 3.4 yn trio uchafsymio y lluoswm o phob tebygolrwydd ag oedd yn edrych ar labeli y data ymarfer. Gan fod rhai labeli am fod yn 0 a rydym yn llusoi y rhifau yma, mae rhaid newid y dechneg ag lluoswm y tebygolrwydd $(1 - p(x_i))$ i phob i sydd gyda'r label 0 i alluogi trio darganfod y paramedrau sy'n creu yr uchafswm (yn ein sefyllfa ni fydd yr uchafswm y rhif agosaf tuag at 1).

Sydd yn gallu cael ei symleiddio i:

$$L(\hat{\mathbf{z}}) = \prod_{i=1}^k p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}$$

Nawr fyddem yn cymryd y log o'r amcangyfrif tebygoliaeth.

$$\log L(\hat{\mathbf{z}}) = \sum_{i=1}^n y_i \log(p(x_i)) + (1 - y_i) \log(1 - p(x_i))$$

Sydd yn symleiddio i:

$$\log L(\hat{\mathbf{z}}) = \sum_{i=1}^n y_i \log \left(\frac{1}{1 + e^{-\hat{\mathbf{z}}\mathbf{x}}} \right) + (1 - y_i) \log \left(\frac{e^{-\hat{\mathbf{z}}\mathbf{x}}}{1 + e^{-\hat{\mathbf{z}}\mathbf{x}}} \right)$$

ac felly:

$$\log L(\hat{\mathbf{z}}) = \sum_{i=1}^n y_i \hat{\mathbf{z}} x_i - \log(1 + e^{\hat{\mathbf{z}} x_i})$$

Yna mae gennym y log o'r amcangyfrif tebygoliaeth. Rydym eisiau darganfod y gwerth o z lle mae'r log o'r amcangyfrif tebygoliaeth ar ei fwyaf.

$$\hat{\mathbf{z}} = \arg \max_{\mathbf{z}} \log L(\mathbf{z})$$

Does yna ddim ffordd bendant o ddatrys yr hafaliad uchod, fydd angen defnyddio algorithmau fel swm lleiaf sgwariau wedi eu hail bwysu drwy iteriadau neu ddisgyniad fwyaf fel gwelwn yn y algorithmau yn R ac Python yn y drefn honno.

Mae'r dull disgyniad fwyaf yn algorithm optimeiddiaeth trefn cyntaf i darganfod isafbwynt lleol o ffwythiant gall ei ddifferu. Mae'r algorithm yn cymryd camau yn gyfranneddol i'r graddiant yn y pwynt yno. Fydd yr algorithm am phob cam yn edrych yn debyg i hafaliad 3.5 gyda a yn rhyw bwynt a f yn y ffwythiant.

$$a_{n+1} = a_n - \nabla f \quad (3.5)$$

Ar gyfer y swm lleiaf o sgwariau wedi eu hail bwysu drwy iteriadau, mae'r algorithm yn cydgyfeirio tuag at y pwysau optimaidd ar gyfer y cyfernodau yn Fformiwla 3.2. Mae angen ail bwysu oherwydd mae'r amrywiant yn newid trwy gydol y model. Mae lle mae'r amrywiant ar ei fwyaf yn y model am dynodi lle fydd gromlin ein model.

Unwaith mae gennym amcangyfrif o'r paramedrau, mae angen darganfod pa mor dda yw ein model logistaidd. I wneud hwn byddwn yn rhoi ein data profi x_j i mewn i'r model, a cynharu'r allbwn gyda y_j . Fel allbwn cawn tebygolrwydd, rhif rhwng 0 ac 1, yna wnawn talgrynnu'r allbwn i cael label. Wedyn mae gennym ein rhagfynegiad am label pob person, yna gallwn ddarganfod cyfradd llwyddiant ein model gan:

$$1 - \sum_{j=k+1}^n \frac{(P(\mathbf{x}_j) - y_j)^2}{n - k}$$

Mae'r swm yma yn cymryd y canran o camgymeriadau rhwng y data ymarfer a'r data profi ac yna yn ei tynnu i ffwrdd o 1, darganwyd hyn y cyfradd llwyddiant. I darganfod y canran o camgymeriadau fyddwn yn tynnu y data profi oddiwrth y data ymarfer, ac yno yn sgwario yr matrices canlyniadol. Fydd y gweithrediadau yma yn gweithio yn ôl elfen. Unwaith fydd y matrices wedi'i sgwario fydd pob elfen sy'n dangos 1 yn dangos camgymeriad ac felly fydd 0 yn dangos rhagfynegiad cywir. Yna fydd y matrices yn cael ei symio ac fydd y cyfanswm yn cael ei rhannu gan y nifer o elfennau.

3.3 Tiwtorial yn R

Yn yr enghraifft hon, fyddwn yn edrych ar ddata ar 1000 o bobol, fydd y data yn cynnwys gwybodaeth ar uchder, pwysau, maint gwasg, oed, rhyw ag oes gan y person diabetes. Mae'n bosib lawrlwytho'r data oddi yma: <https://dysgupeirianyddol.github.io/>

Ar gyfer gwneud atchweliad logistaidd, mae angen y pecyn **stats** a wnawn ei lawrlwytho a'i gosod gan redeg y canlynol:

```
> install.packages("stats")
> library(stats)
```

Yna fydd rhaid lwytho'r data i mewn a'i arbed fel newidyn. Fydd rhaid neud yn siŵr fod y ffwythiant `read.csv` yn cael ei chyfeirio tuag at y lleoliad cywir o le mae eich data chi wedi'i gadw.

```
> data <- read.csv("C:/Users/User/Desktop/Dysgu_Peirianyddol/data_logistic.csv")
```

Unwaith ei fod ar ein consol, mae'n bosib gweld y data:

```
> View(data)
```

	Uchder	Pwysau	MaintGwasg	Oed	Rhyw	Diabetes
1	170.7197	87.70995	40.21594	57	Gwryw	0
2	159.2646	91.67977	39.95974	62	Gwryw	1
3	154.9078	98.35737	34.58633	29	Gwryw	0
4	168.5475	93.48007	41.47911	54	Benyw	1
5	175.8423	79.65120	34.53736	23	Benyw	0
6	169.7488	91.99920	39.35820	44	Benyw	1
7	143.8323	102.34901	42.13036	59	Benyw	1
8	157.2371	88.39940	41.19644	24	Benyw	1
9	197.6471	102.07920	36.38416	60	Gwryw	0
10	156.2165	91.67034	37.50940	27	Gwryw	0

Nawr wnawn rannu'r data fel bod 70% o'r data yw'r data ymarfer ac 30% o'r data yw'r data profi. Fyddwn yn rhannu'r data ar hap.

```
> rhifau <- c(1:1000)
> rhifauymarfer <- sample(x = rhifau, size = 700, replace = FALSE)
> rhifauprofi <- setdiff(rhifau, rhifauymarfer)
```

Mae'r côd uchod yn rhannu'r setiau gan ddefnyddio eu indecs (rhif y rhes) yn y data ac yno mae'r côd isod yn rhannu'r fectorau i mewn i setiau arwahan.

```
> ymarfer <- data[rhifauymarfer,]
> profi <- data[rhifauprofi,]
```

Nawr rydym yn barod i greu'r model logistaidd. I greu'r model fyddem yn rhedeg y côd gan ddefnyddio y ffwythiant `glm`, sydd yn fyr am "Generalized Linear Models", sydd yn golygu gall y ffwythiant cael ei ddefnyddio am lawer fwy o atchweliadau na logistaidd yn unig. Oherwydd hyn mae angen gosod yr opsiwn `family` i `binomial`. I ddilyn strwythur o'r algorithm, byddwn yn creu'r model o'r data ymarfer yn unig.

```
> atchweliad <- glm(Diabetes ~ Uchder + Pwysau + Oed + Rhyw + MaintGwasg,
+                   family = binomial,
+                   data = ymarfer)
```

Unwaith mae'r model wedi'i greu, gallwn weld eu paramedrau sydd wedi cael ei amcangyfrif:

```
> atchweliad$coefficients
(Intercept)      Uchder      Pwysau      Oed
22.858432583 -0.254347133  0.215272873  0.057360113
      RhywGwryw  MaintGwasg
-8.074052403 -0.003206262
```

Felly mae'r model sydd gennym, i dri lle degl, yn edrych fel:

$$P(\mathbf{x}) = \frac{1}{1 + e^{-22.858 + 0.254x_{\text{Uchder}} - 0.215x_{\text{Pwysau}} - 0.057x_{\text{Oed}} + 8.074x_{\text{Rhyw}} + 0.003x_{\text{MaintGwasg}}}}$$

Gan fod ein model wedi'i chwblhau, gallwn weld sut mae'n perfformio yn penderfynu os oes gan bobl y set profi diabetes ta ddim. Geith hyn ei wneud yn defnyddio'r ffwythiant `predict` a dewis yr opsiwn `type` fel "response" i gael allbwn o debygolrwydd. Heb wneud hyn, fydd yr allbwn yn cyfrifo z o hafaliad 3.2.

```
> canlyniad <- predict(object = atchweliad, newdata = profi, type = "response")
> canlyniad <- round(canlyniad, digits = 0)
> canlyniad <- unname(canlyniad)
```

Fyddem yn ogystal yn talgrynnu'r tebygolrwydd o bob person i cael dewis ar os gennym ddiabetes ta ddim. Wedyn fyddem yn tynnu i ffwrdd y rhifau o'r rhesi ar y fector o labeli. Nawr gennym y rhagfynegiad a'r canlyniadau gwreiddiol, gallwn gyfrifo'r canran o'r ddau set sy'n debyg. Gallwn gyfrifo yn y ffurf ganlynol gan fod ein setiau yn ddeuaidd:

```
> 1-(sum((test[,6]-unname(canlyniad))**2)/length(test[,6]))
0.8833333
```

Fel y gwelwn, mae ein model gydag effeithiolrwydd o 88% ar gyfer y data sydd gennym. Gallwn ni defnyddio y model rydym wedi creu i benderfynu ar os gan berson newydd ar hap diabetes neu ddim. Gwelwn hyn gan gyflwyno dyn gydag uchder o 160, pwysau 92, maint gwasg o 34 ag ugain oed yn y côd isod:

```
> unname(round(predict(object = atchweliad,
+                      newdata = data.frame(Uchder = 160,
+                      Pwysau = 92,
+                      MaintGwasg = 34,
+                      Oed = 20,
+                      Rhyw = "Gwryw"),
+                      type = "response"),
+          digits = 0))
0
```

Am y person yma gwelwn fod y model wedi rhagfynegu nad oes ganddo diabetes. Os wnawn ystyried person gyda'r un nodweddion ond yn fenyw:

```
> unname(round(predict(object = atchweliad,
+                      newdata = data.frame(Uchder = 160,
+                      Pwysau = 92,
+                      MaintGwasg = 34,
+                      Oed = 20,
+                      Rhyw = "Benyw"),
+                      type = "response"),
+          digits = 0))
1
```

Gwelwn fod y model yn rhagfynegi ei fod hi gyda diabetes.

3.4 Tiwtorial yn Python

Ar gyfer cynhyrchu atchweliad logistaidd yn python mae rhaid i ni ddefnyddio'r pecynnau **sklearn**, a **pandas** i trin y dada. Wnawn lwytho'r pecynnau gan redeg y côd yma:

```
>>> from sklearn.linear_model import LogisticRegression
>>> import pandas as pd
```

Nawr mae angen llwytho'r data, wnawn ddefnyddio'r un data wnaethom ddefnyddio i'r tiwtorial yn R. Cewch ei lawrlwytho o Mae'n cynnwys 1000 o cofnodion data ar fesuriadau pobl yn cynnwys uchder, pwysau, maintgwasg, oed, rhyw ag label yn dangos os gan y person diabetes neu ddim.

```
>>> data = pd.read_csv('data_logistic.csv')
```

Gallwn gweld yr data gan rhedeg:

```
>>> data.head()
```

	Uchder	Pwysau	MaintGwasg	Oed	Rhyw	Diabetes
0	170.719652	87.709946	40.215944	57	Gwryw	0
1	159.264575	91.679774	39.959742	62	Gwryw	1
2	154.907775	98.357373	34.586330	29	Gwryw	0
3	168.547460	93.480071	41.479106	54	Benyw	1
4	175.842260	79.651198	34.537361	23	Benyw	0

Gan fod ein data gyda rhyw wedi cael ei diffinio gyda'r geiriau “Gwryw” a “Benyw”, mae python yn cael trafferth yn delio gyda nhw. Felly nawn trawsnewid nhw i newidyn deuaidd (set o 1 a 0).

```
>>> data['Rhyw'] = data['Rhyw'].apply(lambda x: int(x == 'Gwryw'))
```

Nawr fydd rhaid i ni rannu'r data i ddata ymarfer ag data profi.

```
>>> ymarfer = data.sample(frac = 0.7)
>>> profi = data.drop(ymarfer.index)
```

Mae'r wybodaeth rydym angen i greu model logistaidd angen fod yn fatrics yn python, felly:

```
>>> X = ymarfer[['Uchder', 'Pwysau', 'MaintGwasg', 'Oed', 'Rhyw']].as_matrix()
>>> y = ymarfer['Diabetes'].as_matrix()
>>> X_profi = profi[['Uchder', 'Pwysau', 'MaintGwasg', 'Oed', 'Rhyw']].as_matrix()
>>> y_profi = profi['Diabetes'].as_matrix()
```

I redeg yr atchweliad logistaidd wnawn ddefnyddio'r ffwythiant yn `sklearn`. Wnawn wneud gan redeg y côd canlynol:

```
>>> clf = LogisticRegression(random_state=0).fit(X, y)
```

Gallwn edrych ar y rhyngdoriad gan

```
>>> clf.intercept_
array([ 1.70933553])
```

ac yna y paramedrau eraill:

```
>>> clf.coef_  
array([[-0.12384414,  0.14710498,  0.12995053,  0.04682347, -4.80795833]])
```

Felly dyma yw ein model i dri lle degol:

$$P(\mathbf{x}) = \frac{1}{1 + e^{-1.709 + 0.124x_{\text{Uchder}} + 0.147x_{\text{Pwysau}} - 0.047x_{\text{Oed}} + 4.808x_{\text{Rhyw}} - 0.130x_{\text{MaintGwasg}}}}$$

Nawr gallwn ni cyfrifo'r gyfradd llwyddiant o ein model ar y data profi. Gallwn ei chyfrifo yn y ffordd ganlynol oherwydd ein bod yn delio gyda data deuaid.

```
>>> 1-(sum((clf.predict(X_profi)-y_profi)**2)/len(y_profi))  
0.9166666666666666
```

Felly mae ein model logistaidd yn python yn rhoi cyfradd llwyddiant o tua 92%. Gallwn nawr ei ddefnyddio ar gyfer rhyw berson tu allan i ein data. Os oes gennym wryw gyda thaldra o 171, pwysau o 130 a maint gwasg ag oed o 40; gallwn ragfynegi os oes gan y person diabetes ta ddim.

```
>>> clf.predict([[171, 130, 40, 40, 1]])  
array([1], dtype=int64)
```

Gwelwn fod y model logistaidd yn rhagfynegu bod gan y person hwn diabetes. Nawr nawn drïo gyda person tebyg ond gyda phwysau o 90 yn lle.

```
>>> clf.predict([[171, 90, 40, 40, 1]])  
array([0], dtype=int64)
```

Gwelwn nad oes gan y person yma diabetes.

Pennod 4

Termau

Llyfryddiaeth

- [1] 2018 kaggle ml and ds survey. <https://www.kaggle.com/sudhirn17/data-science-survey-2018>. Accessed: 2020-03-24.
- [2] Addysg cymraeg yn prydain. https://www.mercator-research.eu/fileadmin/mercator/documents/regional_dossiers/welsh_in_the_uk_2nd.pdf. Accessed: 2020-03-27.
- [3] Adnoddau addysg uwch ar coursera. <https://www.coursera.org/>. Accessed: 2020-03-27.
- [4] Adnoddau code club. <https://projects.raspberrypi.org/cy-GB/codeclub>. Accessed: 2020-03-14.
- [5] Adnoddau mathemateg ar gyfer ysgolion. <https://www.mathemateg.com/>. Accessed: 2020-03-14.
- [6] Adnoddau technocamps. <https://www.technocamps.com/cy/resources>. Accessed: 2020-03-14.
- [7] Adnoddau technolegau iaith. <http://techiaith.cymru/yr-adnoddau/llawlyfr-technolegau-iaith/>. Accessed: 2020-03-14.
- [8] App botio. <https://apps.apple.com/us/app/botio/id1296278646?ls=1>. Accessed: 2020-03-14.
- [9] Canran o pull requests ar github yn wahanol ieithoedd rhaglennu. https://madnight.github.io/github/#/pull_requests/2019/4. Accessed: 2020-03-26.
- [10] Cyfrifiadureg ar gyfer mathemateg. <https://vknight.org/cfm/cy/>. Accessed: 2020-03-14.
- [11] Sgiliau ymchwil cyfrifiadurol. <https://sgiliauymchwilcyfrifiadurol.github.io/>. Accessed: 2020-03-14.
- [12] Targed o miliwn o siaradwyr cymraeg erbyn 2050. <https://gov.wales/sites/default/files/publications/2018-12/cymraeg-2050-welsh-language-strategy.pdf>. Accessed: 2020-03-24.
- [13] Technological action plan. <https://gov.wales/sites/default/files/publications/2018-12/welsh-language-technology-and-digital-media-action-plan.pdf>. Accessed: 2020-03-24.
- [14] Tiwtorial yn sql. <https://plsql-tutorial.com/cy/index.htm>. Accessed: 2020-03-26.
- [15] Videos ar sut i rhaglennu yn python. https://www.youtube.com/playlist?list=PLSkPgScy-DkFdCzwJW9X_B9IfTouojem7. Accessed: 2020-03-14.
- [16] Ystorfeydd technolegau iaith. <https://github.com/porthtechnolegauiaith>. Accessed: 2020-03-14.

- [17] Ethem Alpaydin. *Introduction to Machine Learning*. MIT Press, 2014.
- [18] K.Dietz; M.Gail; K.Krickeberg; B.Singer. *Logistic regression: a self-learning text*. New York : Springer, 1994.
- [19] Alun Morris. Algebra llinol. <https://llyfrgell.porth.ac.uk/View.aspx?id=1716~4p~QbzBunJu>. Accessed: 2020-03-14.
- [20] Alexander Zien Olivier Chapelle, Bernhard Schölkopf. *Introduction to Semi-Supervised Learning*. MIT Press, 2006.
- [21] David M. J. Tax; Ferdinand van der Heijden; Robert Duin; Dick de Ridder. Classification, parameter estimation and state estimation: An engineering approach using matlab. 2012.