Big data aroko metodo estatistikoak
Hotelen klustering-a

2019

Eustat
EUSKAL ESTATISTIKA ERAKUNDEA
INSTITUTO VASCO DE ESTADÍSTICA
www.eustat.es

Organismo Autónomo del
EUSKO JAURLARITZA
GOBIERNO VASCO

Lanketa:
EUSTAT
Euskal Estatistika Erakundea
Instituto Vasco de Estadística

Argitalpena:
EUSTAT
Euskal Estatistila Erakundea
Instituto Vasco de Estadística
Donostia-San Sebastián 1,
01010 Vitoria-Gasteiz

Euskal AEko Administrazioa

1. Argitalpena:
I/2019

Inprimaketa eta Koadernatzea:
Eusko Jaurlaritzako Inprimategia eta Erreprografia Zerbitzua

ISBN: 978-84-7749-491-1

Lege-Gordailua: VI-156/19

Portada irudia: Marcos Gasparutti

BIG DATA GARAIKO METODOLOGIA ESTATISTIKOAK
Hotelen klustering-a

Asier Badiola Zabala

asier.badiola@outlook.es

EUSKAL ESTATISTIKA ERAKUNDEA
INSTITUTO VASCO DE ESTADÍSTICA

Donostia-San Sebastián, 1
01010 VITORIA-GASTEIZ
Tel.: 945 01 75 00
Fax.: 945 01 75 01
E-mail: eustat@eustat.eus
www.eustat.eus

Aurkezpena

Azken garaiotan big data gizartearen arlo guztietara zabaldu da. Halaber, estatistika ofizialean, non erronka bat den bere erabilera datu iturri berri bat bezala. Estatistikako institutuetan hainbat proiektu pilotu ari dira egiten big dataren ondorioei heltzeko produkzio estatistikoaren alderdi guztietan.

Eustatek 2016an antolatu zuen Nazioarteko Estatistika Mintegiaren XXIX. edizioa "Big data for Official Statistics" izenaren pean, zeina Peter Struijsek eman baitzuen, Big Dataren programako koordinatzailea Statistics Netherlands (SN) izenekoan eta Big Dataren taldearen koordinatzailea Europar Batasuneko ESSnet (European Statistical System network) izenekoan.

Argitalpen honek ezagutarazi nahi du arlo honetan eginiko ikerketa lana Eustaten formakuntzako eta ikerkuntzako beketariko baten eskutik. Dokumentu honek bi atal ditu. Lehenbizikoan errepaso bat egiten zaie ikaskuntza automatikoko metodo batzuei big datan erabilgarriak eta bigarrenean heltzen zaio Euskal AEko hotel establezimenduen sailkapenaren ikertzeari, egina bere prezio serietatik, weba eskrapeatuz lortuak.

Vitoria-Gasteizen, 2019ko martxoan
Josu Iradi Arrieta
EUSTATeko Zuzendari Nagusia

Gaien Aurkibidea

Atarikoa

Euskal Estatistika Erakundeak (Eustat) 2017. urtean estatistika eta matematika metodologietan prestatzeko eta ikertzeko emandako bekari esker, **Machine Learning**-en inguruan egin den lanaren emaitza da Koaderno Tekniko honetan bildutakoa.

Helburu nagusia, egungo datu iturri ezberdinetatik jasotzen den informazioaren trataerarako metodologiak aztertzea eta barneratzea da, orain arte erabilitako datu eta estatistikak alde batera utzi gabe. Horren bidez, emaitza aberasgarriagoak eskaintzeko asmoarekin.

Koaderno Tekniko honetarako, turismoaren inguruko informazio gehiago izateko, web plataforma batera joan da. Enpresarekin kontaktuan jartzen saiatu ondoren eta erantzunik jasotzen ez zela ikusita, beraien orrialdeari *web scraping*-a egiten hasi zen, betiere enpresari abisua emanik. Datu horiek lortzen joan ahala (hasiera 2017 uztailean izan zelarik), Eustat-eko Establezimendu Turistiko Hartzaileen Inkestako datuekin lotzen joan ziren hotel eta pentsioak identifikatuz.

Datuen fusioa egindakoan, bekaren helburu nagusietako bat establezimendu horien sailkapen desberdinak egitea izan da, hau da, prezioen serie denboralen clustering ezberdinak. Lortutako emaitzak, *European Survey Research Association*-ek (ESRA) Bartzelonan antolatutako *BigSurv18* konferentzian aurkeztu ziren poster baten bidez.

Hori dela eta, koaderno honen egitura 2 atal nagusitan banatzen da. Alde batetik, *Machine Learning*-en inguruko teoria eta metodo ezberdinak azaltzen dira 1. atalean. Bestetik, 2. atalean hotelen clustering-en inguruan egin den lana aurkezten da, lortutako emaitzak azalduz.

Halaber, atariko hau baliatu nahi dut bekako 2 urte hauetan laguntzen ibili diren Eustateko Metodologia, Berrikuntza eta I+Gko Arloa osatzen duten guztiei: Anjeles Iztueta, Jorge Aramendi, Elena Goni, Inmaculada Gil eta Marina Ayestarán. Eskerrak eman nahi dizkiet baita Eustateko lankideei, horren giro hona sortzeagatik eta Ander Juarez bekadunari niri aguantatu izanagatik.

Gako-hitzak: machine learning, clustering.

Atala I

Teoria

Sarrera

Lehenengo atalean, **Machine Learning**-eko metodo ezberdinak azalduko dira era labur eta sinple batean. Machine Learning-en ideia nagusia, jasotako datuekin ahalik eta eredu egokienak sortzea izaten da orokorrean. Eredu horiek sortzerako orduan ordea, hainbat aukera ezberdin izaten dira, parametro ezezagun ugari agertzen dira eta horiei balio jakin batzuk eman behar izaten zaizkie. Hain zuzen ere, Machine Learning-en bidez, ordenagailuak ahalik eta era automatikoenean parametro egokienak *ikastea* bilatzen da (hortik datorkio hain zuzen ere izena).

Lanean aritu beharreko datuen arabera, 2 bloke nagusi ezberdintzen dira: **Supervised Learning** (1. kapitulua) eta **Unsupervised Learning** (2. kapitulua). Lehenengoaren kasuan, datuek *etiketa* bat izango dute eta helburua datuei *etiketa* era egokian jartzen dion eredua sortzea da. Bigarrenean aldiz, datuek ez dute inongo *etiketarik* eta beraien barne egitura lortzea da helburua.

Sequential Data-n (3. kapitulua), datu egitura berezi bat aztertzen da, ordenak garrantzia duen datuena hain zuzen ere. Datu mota hauek serie denboralekin dute zerikusia askotan eta adibide tipikoak burtsarena eta eguraldiarena dira. Bertan, **Markov-en kateak** aztertuko dira, definizio eta aplikazio batzuk aurkeztuz.

Gaur egunean gero eta indar handiagoa hartzen ari diren **Sare Neuronal Artifizial**-ek ere beraien kapitulu propioa dute (4. kapitulua). Sare horiek, Supervised Learning-eko eta Unsupervised Learning-eko ereduak sortzeko balio dute, baita beste milaka aplikazio ezberdinetarako ere. Orain dela urte dezente sortu baziren ere, konputazio prozesuak azkartu diren azken urteotan bihurtu dira bereziki erabilgarri.

Atal honetako azken kapitulua aldiz (5. kapitulua), **Ensemble Methods**-i dagokio. Metodo horiek, eredu sinple ezberdin asko sortzen dituzte gero horiek konbinatzeko eta horrela eredu sendoago bat sortzeko. Planteamendua erraza eta sinplea badirudi ere, egitura konplexuak era efizienteago batean azaltzeko oso erabilgarriak dira.

Amaitzeko, Machine Learning-ean murgiltzea nahi duen irakurleari, gaur egun R edo/eta Python programazio-lengoaiak erabiltzea gomendatzen zaio. Biak baitira software libreak, funtzio eta pakete sorta ikaragarri handia eskaintzen dute eta atzetik duten komunitate erraldoiek oso eguneratuta mantentzen dituzte 2 programazio-lengoaiak.

Kapitulua 1

Supervised Learning

Lehenengo kapitulu honetako metodoak bi talde nagusitan banatzen dira: erregresioa eta sailkapena. Bi kasu horietan, datuak *etiketaturik* egon beharko dira, lehenengo kasuan *etiketa* hori jarraitua izango da eta bigarren kasuan berriz diskretua eta finitua.

Adibide bezala, hurrengo 2 taulak daude: bi tauletako datuak berdinak dira eta bietan datuak *etiketaturik* daude (azken zutabea da *etiketa*). Hala ere, lehenengo kasuan jarraitua da (Prezioa) eta bigarrenean diskretua eta finitua (Alokatzen/Salgai).

m^2	Solairua	Kostaldean	Prezioa
110	4	Bai	400.000
80	1	Ez	190.000
150	2	Ez	330.000
...	...	...	...
70	5	Bai	390.000

Taula 1.1

m^2	Solairua	Kostaldean	Alokatzen/Salgai
110	4	Bai	Alokatzen
80	1	Ez	Salgai
150	2	Ez	Salgai
...	...	...	...
70	5	Bai	Alokatzen

Taula 1.2

Bi kasuetan bilatuko den helburua **eredu** bat sortzea izango da, datu berriak sartzerakoan *etiketa* egokia jartzeko gai dena. Tauletako adibideei begiratuz gero, lehenengo kasuan datuen aldagai ezberdinekin prezioa ondo estimatzea nahikoa da eta bigarrenean aldiz, sarrerako datuekin pisua salgai edo alokatzen dagoen aurresaten saiatzea.

Kapitulu honetan zehar, eredu horiek sortzeko metodo ezberdinak azaldu eta aztertuko dira, baita eredu horien kalitatea testatzeko erabiltzen den metodologia ere 1.3 azpi-kapituluan.

1.1 Erregresioa

Kapitulu honen sarreran esan bezala, erregresioan, datuetan 2 aldagai mota izango dira: X aldagai independenteak eta Y aldagai dependentea (sarreran *etiketa* deitu dena). Aldagai independenteko datuak **kuantitatiboak** edo **kualitatiboak** izan daitezke eta aldagai dependentea berriz **kuantitatiboa** (erregresio logistikoan, aldagai dependentea kualitatiboa da, baina erregresio logistikoa sailkapen metodo bat kontsideratzen da). Aldagai moten arabera, erregresioa egiteko modua zerbait aldatuko da, baina helburu nagusia Y aldagai dependentea X aldagai independentearen bidez adieraztea izango da, hau da:

$$Y \sim f(X, \boldsymbol{\beta})$$

Horretarako, f funtzioa eta $\boldsymbol{\beta}$ parametroa bilatu beharko dira Y ahalik eta hobekien adierazteko X-ren bitartez. Praktikan, f funtzioa zehaztea ez da lan bideragarria izaten eta horregatik oinarrizko funtzioetara jotzen da, besteak beste funtzio linealetara (ondorengo atalean aztertuko dena sakonago).

Sortutako ereduetan, beti emango da ε errore bat eta hori minimizatzen saiatu behar izaten da orokorrean. Behin f funtzioa definiturik izanez gero, honako adierazpena lortuko da:

$$Y = f(X, \boldsymbol{\beta}) + \varepsilon$$

eta egin beharreko lana, $\boldsymbol{\beta}$ parametroaren balioak lortzea da, ε errorea ahalik eta txikiena izateko.

$$\text{Errorea} = \sum_{i=1}^{n} \varepsilon_i^2 \tag{1.1}$$

1.1.1 Erregresio lineala

Erregresio linealaren bidez sortzen baldin bada eredua, Y aldagai dependentea n aldagai independenteren $(X_1, \ldots, X_n)$ konbinazio lineal bezala adierazten da:

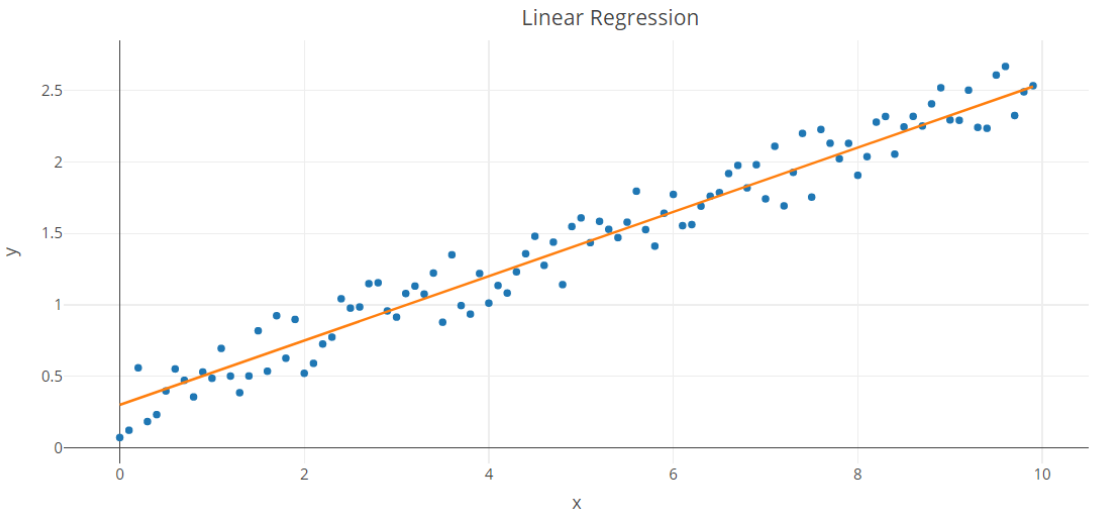
$$Y \sim \beta_0 + \sum_{i=1}^{n} X_i \beta_i \quad : \quad \beta_0, \beta_1, \ldots, \beta_n \in \mathbb{R} \tag{1.2}$$

Kapitulu honen sarrerako adibidearekin jarraituz, aldagai independenteak 3 dira: $X_1 \equiv m^2$, $X_2 \equiv$ Solairua eta $X_3 \equiv$ Kostaldean. Ondorioz, eredua sortzeko kalkulatu beharreko parametroak $\beta_0, \beta_1, \ldots, \beta_n \in \mathbb{R}$ balioak dira.

Eredu lineal hauek, sor daitezkeen eredu sinpleenetarikoenak dira, baina hala ere konputazio garaiaren aurretik garatutako metodo ugari daude eta datuak era intuitibo eta erraz batean adierazteko balio dute. Gaur egungo estatistiketan ere pisu handia dute eredu probabilistikoak sortzeko garaian.

(1.2) ekuazioko problema ebazteko ($\beta_0, \beta_1, \ldots, \beta_n \in \mathbb{R}$ balioak lortzeko) era eta metodo ezberdinak daude. Kasu gehienetan, X aldagai independenteak eta Y aldagai dependenteei baldintza batzuk betetzea eskatzen zaie, baita sortzen den erroreari ere. Koaderno honetan, ez da sakonduko erregresio linealaren atzean dauden estatistiketan, baina interesaturik dagoen irakurleak liburu asko aurki ditzake (baita hainbat eta hainbat kode eta pakete `R` bezalako programazio-lengoaietan) estatistika eta erregresio linealaren inguruan, besteak beste [10], [3] edota [5] liburuak erabili daitezke erreferentzia bezala.

(1.2) problema era egokian ebatziz gero, Irudia 1.1-ko emaitza batera hel daiteke.



Irudia 1.1: Erregresio linealeko adibide bat. x aldagai independenteak $[0, 10]$ tartean hartzen ditu balioak eta y aldagai dependenteak $[0, 3]$ tartean. Lortzen den eredua honakoa da: $y \sim 0.3 + 0.225x$. Irudian azter daitekeen bezala, sortutako ereduaren eta datuen artean errore desberdinak agertzen dira. Errore horiek (ε) sarritan datuek duten *zarataren* ondorio izaten dira eta kasuaren arabera onargarriak izango dira edo ez.

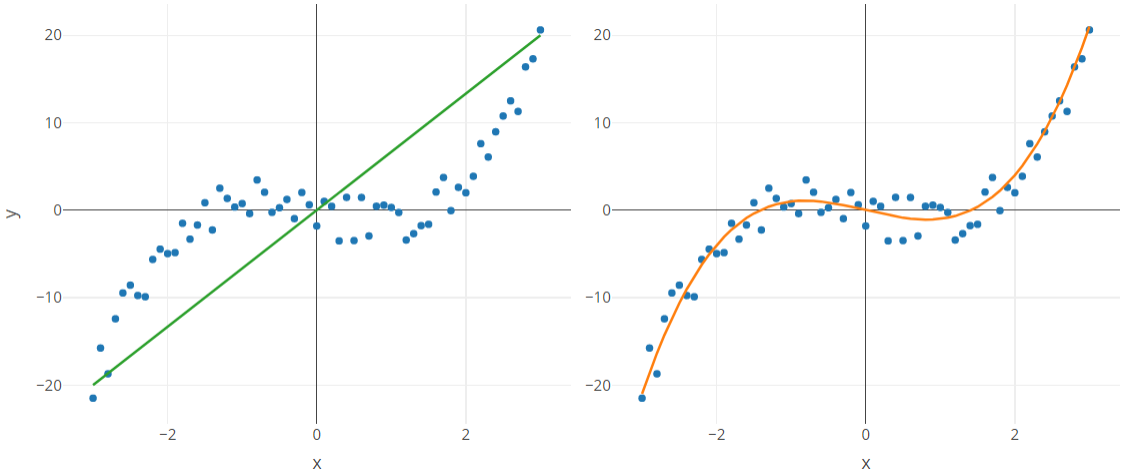
Errorea

Irudia 1.1 ikusten bada, argi ikusten da lortzen den eredua ez dela zehatza, erroreak sortzen dira. Kasu horretan, erroreak beti emango dira ezin delako datu multzoa era lineal baten adierazi. Hala ere, begi bistaz esan daiteke ereduak dezente ondo adierazten duela datu multzoaren joera. Horregatik, erregresio linealeko helburua errore hori txikitzen duen funtzio lineala lortzea izango da, hau da, (1.2) ekuazioko β_0 eta β_i egokiak aukeratzea.

Esan bezala, erregresio linealaren bidez sortzen diren ereduak, asko landutako teoria dute atzetik, baina horiek aplikatzeko, datuek hipotesi batzuk bete behar dituzte: X aldagai independenteen independentzia lineala, ε erroreak normalitatearen hipotesia onartu behar du ($\varepsilon \sim N(0, \sigma^2 I)$), etab.

Eredu lineal konplexuagoak

Azpi-kapitulu honen hasieran ikusi den (1.2) ereduarekin, Y aldagai dependentearen hurbilketa konplexuago eta hobeak egin daitezke X aldagaia eraldatuz eta metodo berdina erabiliz. Demagun Irudia 1.2-ko datu multzoa jaso dela. Argi ikusten denez lehenengo irudian, $y \sim \beta_0+\beta_1 x$ ez da eredurik onena datu multzo hori adierazteko. 3. mailako polinomio bat hartzen baldin bada berriz, $y \sim \beta_0+\beta_1 x+\beta_2 x^2+\beta_3 x^3$, ereduak datuen joera askoz egokiago adierazten du.



Irudia 1.2: Ezkerreko grafikoan, $y \sim \beta_0 + \beta_1 x$ erako eredu lineal sinple bat sortu da eta eskuineko grafikoan berriz $y \sim \beta_0+\beta_1 x+\beta_2 x^2+\beta_3 x^3$ erako eredu konplexuago bat. Azter daitekeenez, bigarren ereduak era egokiago batean azaltzen du datuen joera.

Pentsa daiteke x aldagaian berretzaileak sartzean linealtasuna galtzen ari dela, baina ez da egia, linealtasuna $\boldsymbol{\beta}$ aldagaiarekiko baita.

$$Y \sim X \cdot \boldsymbol{\beta}$$

x	x^2	x^3	y
-2.5	6.25	-15.625	-21.5
-2	4	-8	-5
-1	1	-1	0.5
...	...	...	...
2.5	6.25	15.625	21

Taula 1.3: Adibide honetan, sarrerako datu originalak zutabe grisak dira, hau da, x eta y. Hala ere, taulara beste bi aldagai gehitu dira x aldagaia eraldatuz (berreketen bidez). Era honetan, $X = x$ hasierako aldagai independentea $X' = (x, x^2, x^3)$ aldagai independenteetara aldatu da eredu konplexuago bat eratzeko asmoarekin.

X eraldatzen den arren, $\boldsymbol{\beta}$-rekiko ereduak lineala izaten jarraituko du. Hori horrela izanik, X aldagai independentean hobekien etor daitezkeen eraldaketak egin

daitezke ($\ln x, \sin x, \tan x \ldots$), betiere linealki independenteak baldin badira (ikusi Taula 1.3-ko adibidea).

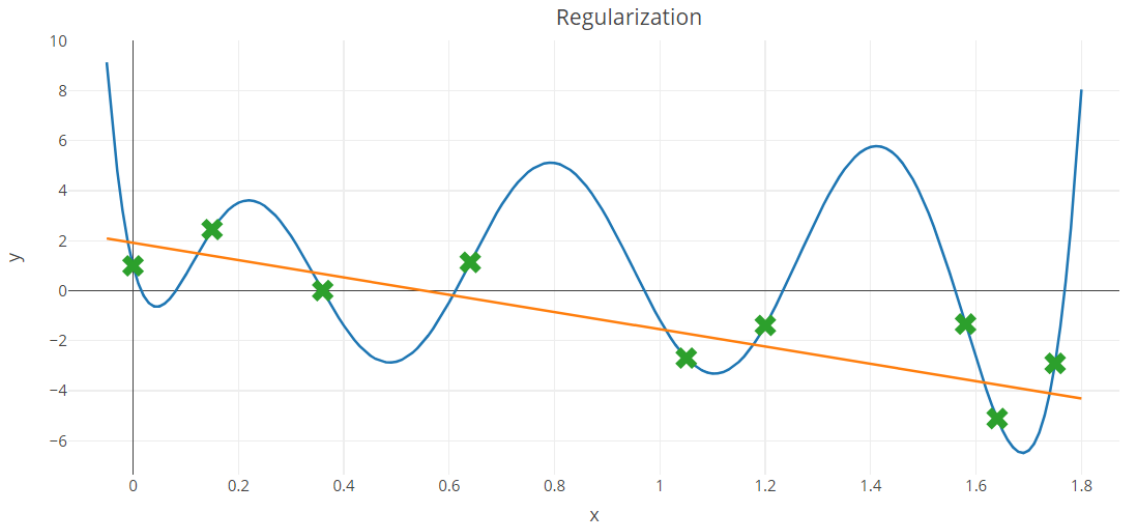
Erregularizazioa

Supervised Machine Learning-en, ez dira datu guztiak erabiltzen erregresiorako, datuen ehuneko haundi bat hartzen da eredua sortzeko eta gainontzeko datuak ereduaren *kalitatea* edo *zehaztasuna* neurtzeko erabiltzen dira (Ikusi 1.3 azpi-kapitulua). Horren ondorioz, eredua sortzerako orduan, ez da bilatzen datuak zehatz-mehatz azaltzea, baizik eta egitura orokorra barneratzea, eta hori **erregularizazioaren** bidez egin daiteke. Horretarako, metodo desberdinak daude[1], baina erabilienak, $\boldsymbol{\beta}$ parametroaren balioei mugak jartzen dizkienak dira, balio altuak penalizatuz. Adibidez, *Ridge Regression* metodoan, muga horiek distantzia euklidearraren bidez penalizatzen dira eta ondorioz, minimizatzea nahi den balioa honakoa da:

$$\sum_{i=1}^{n} \varepsilon_i^2 + \lambda \sum_{i=0}^{n} \beta_i^2 \quad : \quad \lambda > 0 \tag{1.3}$$

λ parametroa aukeratzeko, balio desberdinekin probatzen da. Probatzen den balio bakoitzeko (1.3) balioa minimizatzen duten β_i balioak aukeratzen dira eta ondoren, lortutako emaitzak eta eredua testatzen dira. Ereduak testatzeko erabiltzen diren metodoak 1.3 azpi-kapituluan azaltzen dira.

Erregularizazioaren eragina ikusteko, Irudia 1.3 azter daiteke.



Irudia 1.3: Guruzte berdeak, sarrerako 9 datuak dira eta lerro urdina 8. mailako polinomioaren bitartez lorturiko eredua. Bertan, (1.1)-n ikusitako errorea 0 baliokoa izango da, baina ez du datuen egitura era egokian azaltzen. λ parametro egokia erabiliz gero, laranjaz marraztutako eredu lineal sinplea lortu daiteke, desiragarriagoa dena.

[1]Metodo horiek zehatzago ikus daitezke [15] liburuko 3.4 atalean.

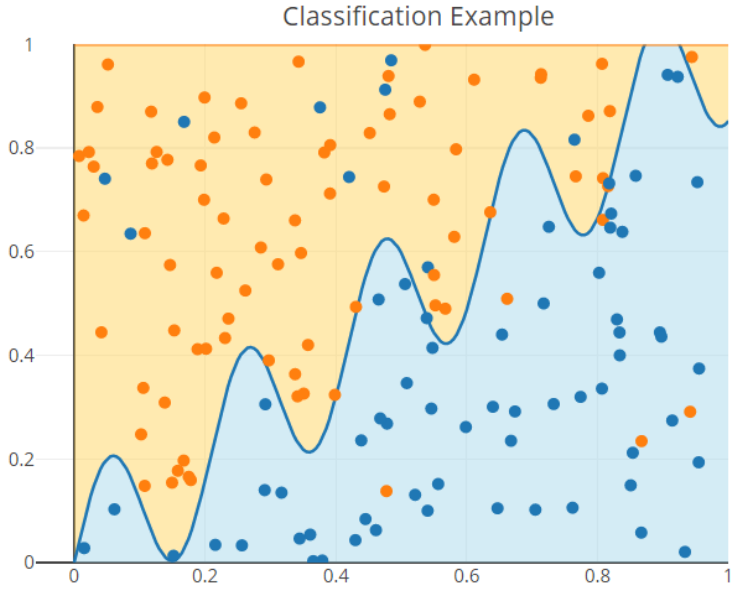
Erregularizazioaren helburua, eredua datuei gehiegi ez doitzea da (kasu horietan **overfitting**-a eman dela esaten da) eta horretarako eredu konplexuak zigortzen ditu, sinpletasunari pisu handiagoa emanez.

1.2 Sailkapen metodoak

Erregresioa aztertu ondoren, sailkapen metodoekin jarraituko da. Sarrerako datuak X espazio baten egongo dira eta datu bakoitza *klase* edo *etiketa* bakar batera esleituta egongo da. Klase horiek adierazteko, Y espazio **diskretua** edo **kualitatiboa** erabiliko da, hau da, Y espazioa K klasez osaturik badago, honelako itxura izango du:

$$Y = \{1, 2, \dots K\}$$

Helburua, f sailkapen funtzio bat lortzea izango da, X espazioko edozein elementuri klase bat esleitzen diona era egoki batean. Irudia 1.4 aztertu ezkero, bertako sarrerako datuak marraztuta dauden puntuak dira. $X = \mathbb{R}^2$ planoa da eta $Y = \{0, 1\}$ edo $Y = \{urdina, laranja\}$, hau da, 2 klase ezberdin daude. 2 klase bakarrik dauden sailkapenei, sailkapen bitar edo binario deitzen zaie eta 2 baino gehiago dutenei klase anitzeko sailkapena.



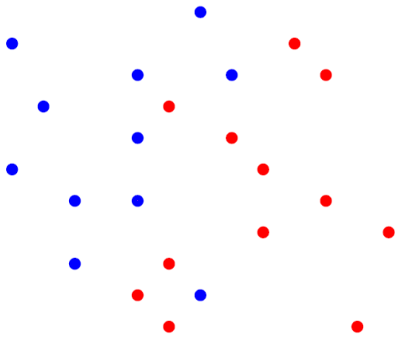
Irudia 1.4: Sailkapen metodo baten adibidea. Puntu laranjak eta urdinak, sarrerako datuak dira, bakoitza bere etiketa edo klasearekin. Lerro urdina bestalde, sailkatzeko sortu den eredua da: horren gainetik dauden puntuak klase batera esleituz eta behealdean daudenak bestera.

Irudia 1.4 agertzen den lerro urdina, lortzea nahi den f sailkapen funtzioa da. Erabiltzen den metodoaren arabera, sailkapen funtzio desberdinak lortuko dira eta ondorioz emaitzak ere desberdinak izango dira. Behin f lortzean, X espazioko puntu

bakoitza klase bati esleituta egongo da. Irudia 1.4-n, urdinez margotutako eremua, klase bati lotuta dago eta laranjaz margotutakoa besteari.

1.2.1 *k*-Nearest Neighbors Classifier

Sailkapen metodo honetan, X espazioko elementuen arteko distantzia erabiltzen da. X espazioaren sailkapena egiteko, $x \in X$ bakoitzari, *gertuen* duen edo distantzia txikienera duen elementuaren klase berdina atxikitzen zaio, Irudia 1.6-n ikusten den bezala. Kasu horretan, 1-NN sailkapena egiten da, hau da, distantzia txikienera duen puntuaren klasea bakarrik hartzen da kontutan. Garbi izan behar da distantzia laburrena beti sarrerako datuekiko bilatu behar dela.

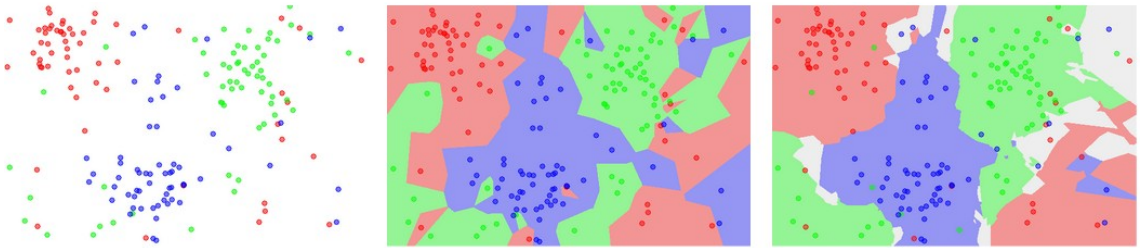


Irudia 1.5: Sarrerako datuen sailkapen bitar bat

Irudia 1.6: x elementuaren klasea, distantzia txikienera dagoen elementuaren klase berdina izango da (1-NN kasuan), kasu honetan x klase urdinean sailkatuko da.

Halere, kasu hori orokortu egin daiteke. x elementutik distantzia txikienera dagoen elementua bakarrik hartu beharrean kontutan, k elementu *gertukoenak* har daitezke. Ondoren, elementu horien artean gehien errepikatzen den klasea zein den zehaztu eta klase hori esleitu x elementuari. Kasu horri k-NN sailkapena deritzo.

k balioa aukeratzen



Irudia 1.7: Ezkerreko irudian sarrerako datu originalak agertzen dira (3 klase daudelarik), erdikoan 1-NN sailkapena egin ondoren lortzen den sailkapena eta eskuinekoan 5-NN-ren bidez lortutakoa. Eskuinean agertzen diren gune grisak, gehien errepikatzen diren 2 klase daudelako da.

Orokorrean, k-NN sailkapen metodoan, k txikia denean errorea txikiagoa izaten da eta aldiz k handia denean, sailkapen eremua *leunagoa* eta *egonkorragoa* izaten da (horren adibide da Irudia 1.7). k egokia hautatzeko, 1.3 azpi-kapituluko teknikak erabiltzen dira, eredu ezberdinak sortuz k ezberdinetarako eta ondoren emaitzak testatuz. k-ren balioaren aukeraketa aurreko ataleko erregularizazio bezala uler daiteke: k-ren balio txikietarako **overfitting**-a eman daiteke.

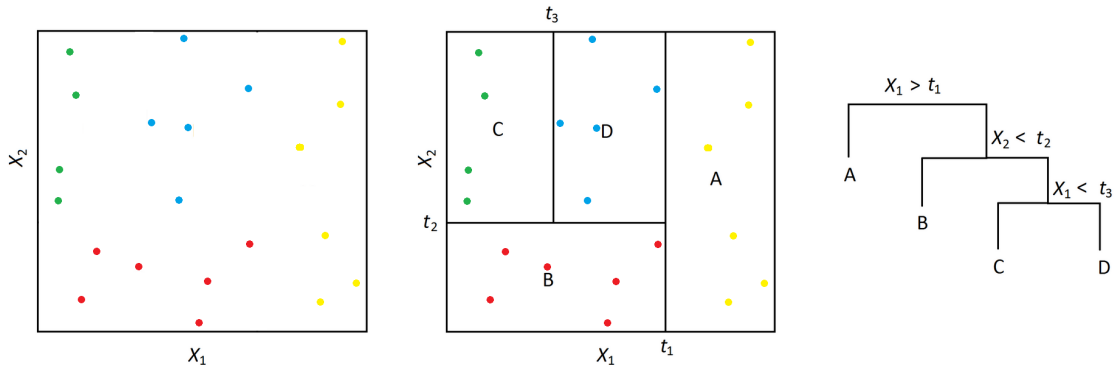
Abantailak eta Desabantailak

Aurrerago esan bezala, metodo hau ulertzeko eta inplementatzeko sinplea eta erraza da, definitu beharreko gauza bakarrenetarikoa, erabili nahi den distantzia da. Hala ere, desabantaila bezala, sarrerako datu multzoa oso handia izanez gero erabiltzen den algoritmoa kostu oso handikoa izango dela da, izan ere, k-NN metodoan puntu guztiekiko distantzia kalkulatu behar da.

1.2.2 Decision Tree Classifier

Decision Trees metodoa, **sailkapenerako** naiz **erregresiorako** erabili daitekeen metodoa da, hala ere atal honetan sailkapenerako erabiltzen den zatia ikusiko da. Metodo hau *zatiketa sisteman* oinarritzen da eta ahalik eta intuitiboen egiteko, adibide batekin hasiko da.

Irudia 1.8 aztertzen bada, 4 klaseko datu multzo bat ageri da. Bertan, 2 dimentsio daude, X_1 eta X_2, eta sailkatzeko erabili den bidea eskuineko zuhaitzean erakusten da. Lehenik, X_1 ardatzean t_1 puntua lortu eta puntu horretatik eskuinera dauden puntuak A taldean batu dira (puntu horiak). Ondoren, X_2 ardatzean t_2 puntua lortu eta puntu horretatik behera dauden puntuak B taldean sailkatu dira (puntu gorriak). Azkenik, X_1 ardatzean t_3 puntua lortu eta C eta D taldeak sortu dira (puntu berdeak eta urdinak hurrenez hurren).



Irudia 1.8: Ezkerreko irudian, sarrerako 21 datuak 4 klasetan sailkatuta daude. Eskuineko irudian, *Decision Tree* metodoa aplikatu eta gero lortzen den sailkapena agertzen da.

Abantailak [2]

- Erraza eta sinplea ulertzeko eta interpretatzeko.
- Aldagai kuantitatibo eta kualitatiboekin lan egin dezake eta ez du datuen preprozesaketa sakonik eskatzen.
- Datu askorekin lan egiteko gai den metodoa da.

Desabantailak

- *Decision tree* algoritmoak ezegonkorrak izan daitezke: sarrerako datuen aldaketa txikiak, irteerako emaitza oso ezberdinak eman ditzake.
- *Decision tree* optimo global bat lortzea, *NP-complete*[3] [11] problema bat da eta arazo konputazional horri aurre egiteko, *ensemble methods*-ak erabiltzen dira (5. kapituluan aurki daiteke informazio gehiago).
- Metodoaren sinpletasunak, kontzeptu zailagoak azaltzeko orduan arazoak ematen ditu: XOR/ALA ate logikoarekin, multiplexadoreekin...

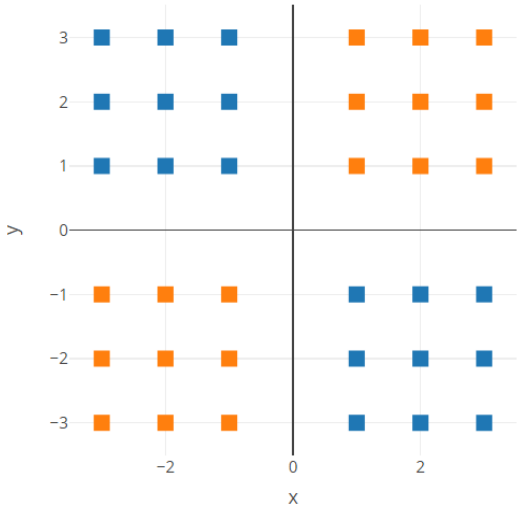
Algoritmoaren ideia

Algoritmoaren ideia, aldagaietan sailkapen errorea txikitzen duen puntua aurkitzea da. Irudia 1.8-n adibidez, algoritmoak X_1 eta X_2 aldagaietan proba desberdinak egin ondoren, sailkapen errorea minimizatzen duen puntua X_1 aldagaiko t_1 dela ondorioztatu du. Errore neurketa hori egiteko, arruntena **entropia** edo **Giniren indizea** erabiltzea da. Behin t_1 puntua zehaztutakoan, prozedura berdina jarraitzen da t_2 eta t_3 puntuak lortzeko.

[2] Abantailak eta desabantailak sakonkiago ikusteko: [27] `http://scikit-learn.org/stable/modules/tree.html`

[3] *NP-complete*-n inguruan ideia orokor bat egiteko: `http://www.geeksforgeeks.org/np-completeness-set-1/`

Algoritmo horrek ordea, zenbait kasutan huts egiten du sarrerako datuen banapenaren arabera, adibide bezala Irudia 1.9 dago. Hori konpontzeko erabiltzen den metodo bat **Pruning** metodoa da (kimatze metodoa). Metodo honen ideia nagusia, hasieran zuhaitz handi bat lortzea da eta ondoren, zuhaitz horren *nodo* edo *hostoak* ezabatzen edo mozten joatea interesekoa den zuhaitz txikiago bat lortu arte. Metodo hau *overfitting*-a saihesteko ere erabiltzen da, erregularizazio modura.



Irudia 1.9: Grafiko honetan, begi bistaz erraz sor daiteke zuhaitza, baina t_1 eta t_2 ebaketa puntuak aukeratzeko orduan, algoritmoa ez zen gai izango puntu egokia aukeratzeko, ez delako gai izango errorea minimizatzen duen puntua aukeratzeko. Kasu hauetarako, aipatutako **Pruning** metodoa erabiltzen da.

1.2.3 Support Vector Machine (SVM)

Sailkapen metodoekin amaitzeko, *support vector machine* (SVM) metodoari buruz hitz egingo da hurrengo lerroetan. SVM informatikako alorrean garatu zen 1990. hamarkadan eta gero eta ezagunago egiten joan da urteak pasa ahala. Emaitza onak ematen dituela ikusi da datu sorta ezberdinen aurrean eta askotan kontsideratu izan da erabilera errazeko sailkapen metodorik hoberenetarikoena.

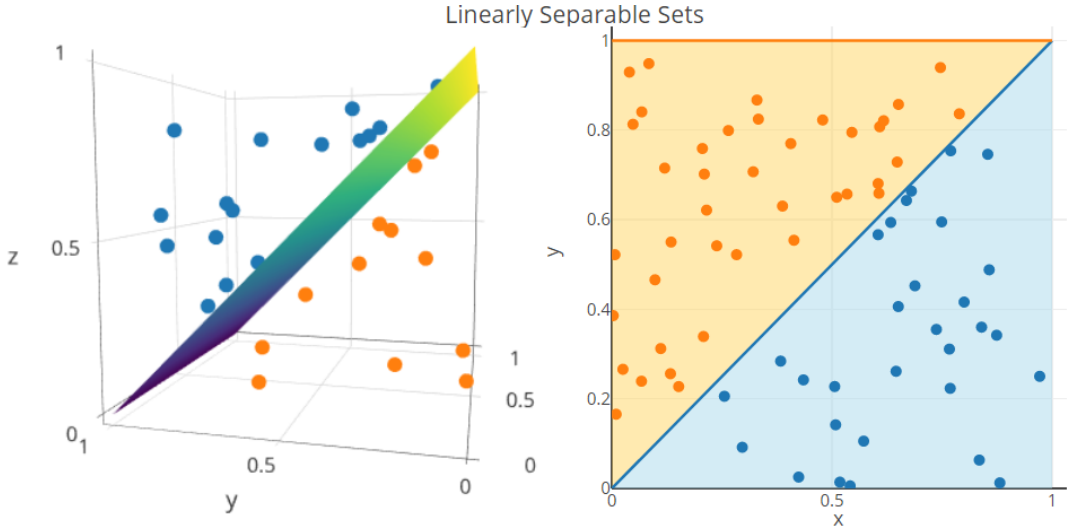
Funtzionamendua ulertzeko, lehenik eta behin hiperplanoaren definizio intuitiboa emango da eta horrekin batera, *maximum margin classifiers* algoritmoaren ideia nagusiak aztertuko dira. Behin hori definiturik, *support vector classifier* azalduko da eta azkenik SVM metodoa.

Maximum margin classifier

Algoritmo edo teknika hau erabili ahal izateko, aztergai dauden datuek sailkapen bitarra dutela eta **linealki banangarriak**[4] direla suposatuko da. Linealki banan-

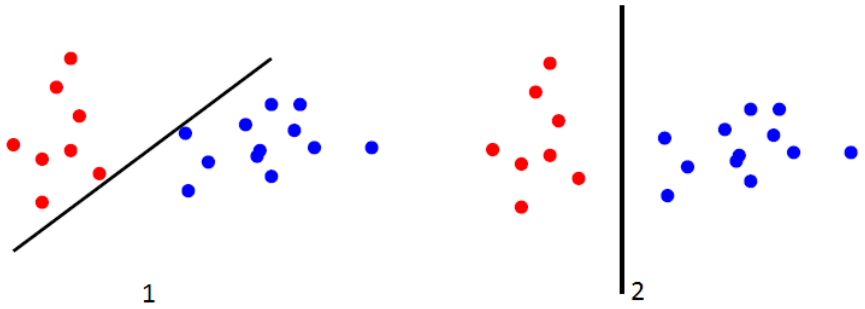
[4]SVM metodoaren atzean dauden matematikak hobeto ulertzeko eta definizio zehatzagoak izateko [17] liburuko 9. atalean aurki daitezke.

garria izateak, matematikoki, datu sortako puntuak hiperplano baten bidez banatu daitezkeela esan nahi du. 2 dimentsioko espazio baten kasuan, zuzen batekin bana daitekeela eta 3 dimentsioko espazio baten kasuan, plano batekin.



Irudia 1.10: Irudiko 2 datu sortak linealki banangarriak dira. Lehenengoan, 3 dimentsioko espazioan plano banatzaile bat lortu da eta bigarrenean, 2 dimentsioko espazioan, zuzen banatzaile bat.

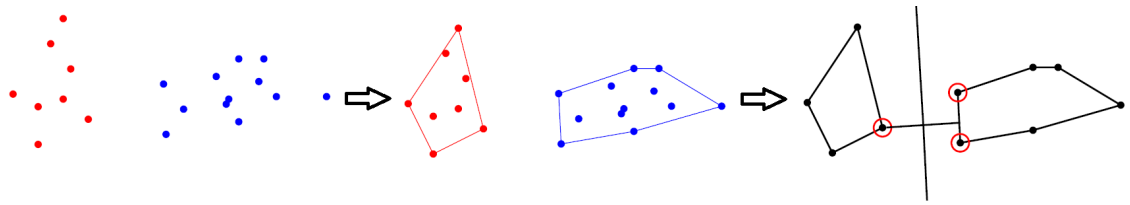
Linealki banangarriak diren datu sortetan, kasu gehienetan infinitu hiperplano existituko dira sailkapena egin ahal izateko. Adibide bezala Irudia 1.11 dago, bi zuzenak balio dute sailkapenerako, baina kasu horretan 2. sailkapenari pisu handiagoa emango zaio, *hobea* izango da nolabait.



Irudia 1.11: Linealki banangarria den datu sorta berdinaren bi sailkapen desberdin.

Maximum margin classifier-ek 2. emaitza hori bilatzen du hain zuzen ere: klase bakoitzetik **distantzia urrunera** dagoen hiperplanoa. Distantzia hori maximizatzeko erabiltzen den ideia intuitiboa Irudia 1.12-n agertzen da.

Lehenik eta behin, klase bakoitzeko puntuen multzo konbexua eratzen da eta behin eratutakoan, multzo horietatik urrutien dagoen hiperplanoa bilatzen da, hau da, bi multzo horien erdibidean dagoen hiperplanoa.



Irudia 1.12: *Maximum margin classifier*-en ideia intuitiboa.

Hiperplano hori, matematikoki lortzen da optimizazio problema baten soluzio bezala, hurrengo lerroetan idatziko dena:

- Izan bitez hurrengo sarrerako datuak $x_1, \ldots, x_n \in \mathbb{R}^p$ eta datu horiei dagokien klase bitarra $y_1, \ldots, y_n \in \{-1, 1\}$.

- Helburua $\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p = 0$ hiperplano bat lortzea da ikusi diren baldintzak betetzen dituena. Horretarako, β_i balio egokiak aukeratu beharko dira $i = 0, 1, \ldots, n$ balioetarako. Balio horiek, hurrengo optimizazio problemaren soluziotik lortzen dira:

$$\max_{\beta_0, \beta_1, \ldots, \beta_n} M$$

hurrengo bi baldintzetara loturik

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i_1} + \ldots + \beta_p x_{i_p}) \geq M, \quad \forall i$$

Support vector classifier

Support vector classifier-en ideia, *maximum margin classifier*-en ideia berdina da, hiperplano *egoki* bat aukeratzea sailkapenerako. Kasu honetan ordea, sarrerako datuek ez dute zertan linealki banangarriak izan behar, hau da, kasu orokorragoetarako balio du. Hori lortzeko, sailkapenean erroreak egon daitezkeela onartu beharko da. Halere, metodoaren funtsa berdina izaten jarraitzen du.

Kasu honetan, optimizatu beharreko problema honakoa da:

$$\max_{\beta_0,\beta_1,\ldots,\beta_n,\epsilon_1,\ldots,\epsilon_n} M$$

hurrengo baldintzetara loturik

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

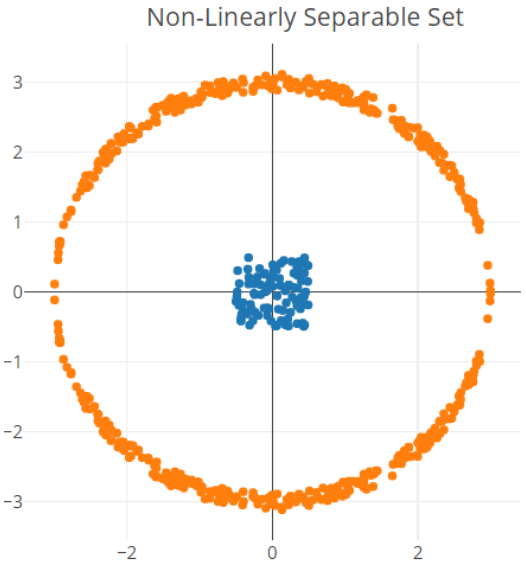
$$y_i(\beta_0 + \beta_1 x_{i_1} + \ldots + \beta_p x_{i_p}) \geq M(1 - \epsilon_i), \quad \forall i$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C$$

Ikus daitekeenez, aldaketa nagusia ϵ_i balioak agertzen direla da. Horrek, sailkapenean erroreak egotea baimentzen du eta C balioak errore hori bornatzen du (C balioa erregularizaziorako parametro bezala kontsidera daiteke).

Support vector machine

Datuak hiperplano baten bidez sailkatzea sarritan nahikoa izango da, baina orokorrean jasotako datuak ezingo dira linealki banatu, sailkapen konplexuagoak izaten dituztelako, Irudia 1.13-n ikus daitekeen bezala. Kasu horietarako, SVM erabiltzen da, *support vector classifier*-en hedapen bat. Hedapen horretarako, *kernel*-ak sartzen dira jokoan.



Irudia 1.13: Linealki banatzea ezinezkoa den datu multzoa.

Kontutan izan SVM metodoaren atzean dagoen **ideia** azalduko dela hurrengo lerroetan, honen atzean dauden matematikak eta teoria zerbait konplexua delako eta ez delako koaderno honen helburua alde teknikoetan gehiegi sartzea. Esan bezala,

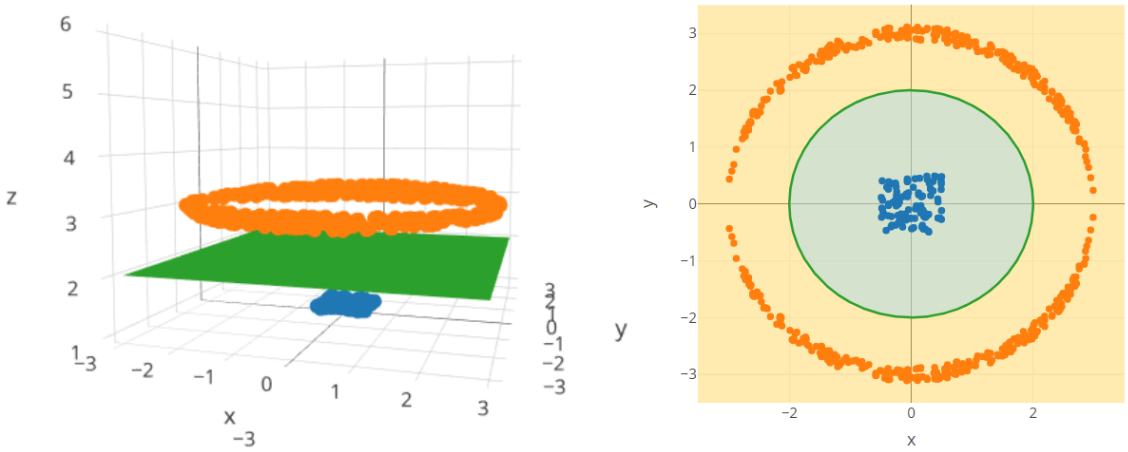
SVM-ren funtsa *kernel* desberdinak erabiltzen datza eta informazio gehiago nahi duen irakurlearentzat [17] liburuko 9. ataletik hastea gomendatzen da, atal honetan ulertzeko errazagoa den azalpen bat emango delako.

Irudia 1.13-ko adibidearekin jarraituz gero. Bertan, ezin da sailkapen egoki bat egiteko gai den hiperplano bat aurkitu, gogoratu kasu horretan hiperplanoa zuzen bat izango dela. Ondorioz, kasu horiei konponbide bat emateko datuak espazioz aldatuko dira. Horretarako, ϕ funtzio bat definituko da non $\phi : \mathbb{R}^p \longrightarrow \mathbb{R}^P$ funtzio horrek datuak dimentsio handiago batera eramango dituen. Horren adibide gisa, Irudia 1.14-ko ezkerreko grafikoa azter daiteke. Bertan, Irudia 1.13-ko $\mathbb{R}^2$ datuak $\mathbb{R}^3$ espaziora pasatzen dira hurrengo ϕ funtzioaren bidez:

$$\begin{array}{rcl} \phi: \mathbb{R}^2 & \longrightarrow & \mathbb{R}^3 \\ (x, y) & \longrightarrow & \phi(x, y) = (x, y, \sqrt{x^2 + y^2}) \end{array}$$

Datuak espazio berri horretara eraldatutakoan, hiperplano bat eratzeko aukera dago sailkapen egoki bat egiteko gai dena. Adibideko kasuan, $z = 2$ hiperplanoa hartu da. Behin hiperplanoa lortutakoan, jatorrizko espaziora bueltatzen da lortu den sailkapen eremua aztertzeko (Irudia 1.14-ko eskuineko grafikoa).

$$z = 2 \Longrightarrow 2^2 = x^2 + y^2$$



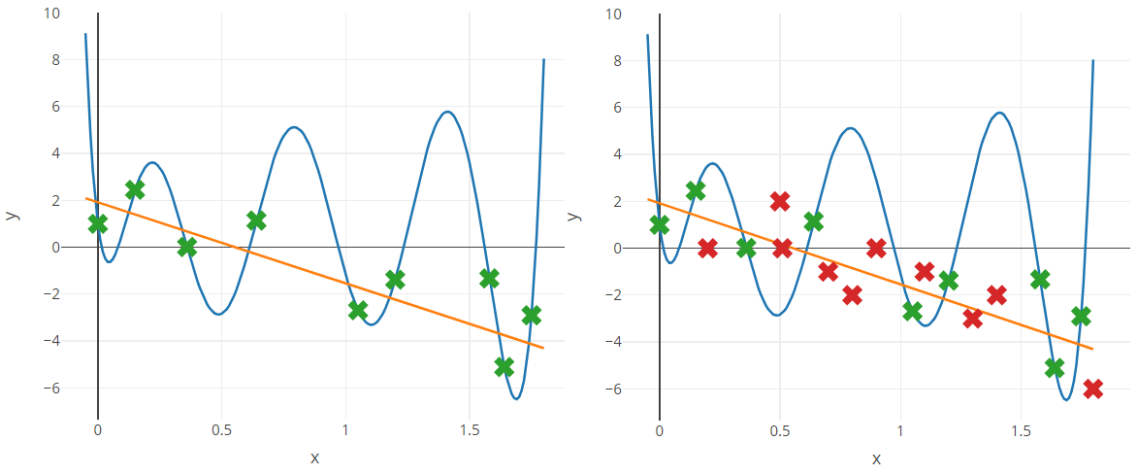
Irudia 1.14: Ezkerreko irudian, 2 dimentsiotik 3 dimentsiora pasatu dira datu originalak ϕ funtzio baten bidez eta ondoren hiperplano baten bidez sailkatu dira. Eskuineko irudian, hiperplano horren banaketaren bidez lortutako sailkapen finala agertzen da.

Gogoratu, SVM metodoaren inguruan azaldutakoa era intuitibo eta errazean azaltzeko asmoarekin egin dela. Batez ere kontutan izan behar da SVM-n erabiltzen diren *kernel* desberdinek, *Support vector classifier* eta *Maximum margin classifier*-en ikusitako algoritmo eta teknikak erabilgarri egiten dituztela linealki banangarriak ez diren kasuetarako. Horien bidez, espero daitekeen bezala nahi bezain sailkapen konplexuak lor daitezke *kernel* egokiak erabili ezkero.

1.3 Ereduak testatzen

Orain arte, kapitulu honetan Supervised Machine Learning-en barnean dauden metodo ezberdinak ikusi dira ereduak sortzeko. Hala ere, aztertu den bezala eredu horiek sortzeko hainbat era ezberdin daude eta horien bidez sortutako sailkapen edo erregresio metodoek diferentzia nabarmenak izan ditzakete. Hori horrela izanik, aukera ezberdin horietatik egokiagoa edo aproposagoa zein den erabakitzeko modu bat beharko da.

Ereduak sortzerako orduan, kontutan izan behar da kasu gehienetan etorkizunean jasotzen diren datuak ondo sailkatzeko gai izan behar dutela. Adibide bezala, Irudia 1.15 kontsidera daiteke. Bertan, bi grafiko ageri dira. Gurutze berdeak, sarrerako datuak dira, ereduak sortzeko erabili direnak, eta lerro laranja eta urdina berriz sortu diren 2 eredu ezberdin. Ezkerreko grafikoan begiratuz gero, urdinez marraztutakoak datuak inongo errorerik gabe kokatzen ditu eta laranjaz marraztutakoak aldiz errore batzuk ditu. Datu berriak sartzerakoan ordea, eskuineko grafikoan gurutze gorrien bidez adierazita daudenak, eredu laranjak datuen jokaera era egokiago batean adierazten duela ikusten da, errore baxuagoa sortuz.



Irudia 1.15: Ezkerreko irudian, sarrerako datuekin (gurutze berdeak) sortu diren 2 eredu ezberdin ageri dira (laranjaz eta urdinez). Eskuineko irudian berriz, datu berriak agertzen dira (gurutze gorriak).

Askotan ordea, ezin izango da itxaron datu berriak etorri arte ereduak testatzeko eta ondorioz, eskura dauden datuak erabiltzen dira testak egiteko. Ideia nahiko sinplea da: eskura dauden datuen zati bat entrenatzeko eta bestea testatzeko erabiltzea. Hau da, datuen zati bat eredua sortzeko erabiltzea eta bestea *datu berri* bezala kontsideratuz testak egiteko erabiltzea.

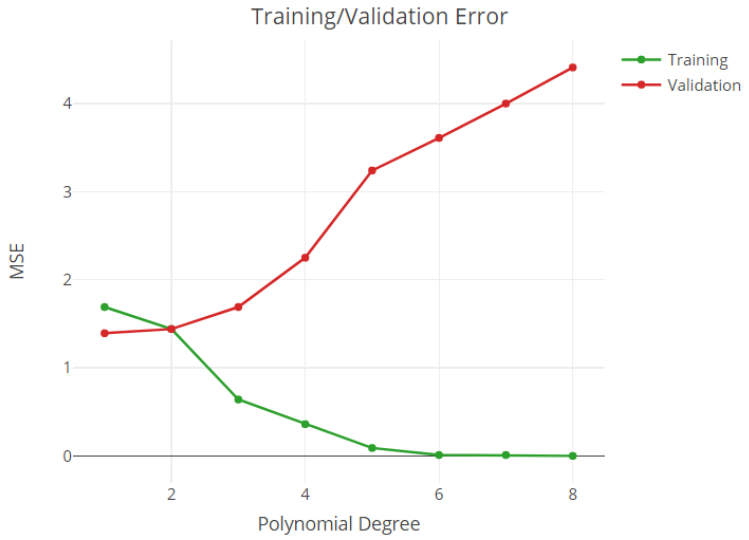
Horretarako ordea, **oso garrantzitsua da banaketa hori ahalik eta era uniformeen egitea**, ahalik eta datuen aldakortasun handiena jaso ahal izateko. Adibidez, Europan dauden jatorduen inguruko modelo bat eraikitzea nahi baldin bada, ez da egokia izango eredua sortzeko Espainiako datuak kanpoan uztea eta eredua testatzeko Espainiako datuak erabiltzea.

1.3.1 Training, validation eta test set

Ereduak testatzeko, datuak 3 zatitan banatzen dira orokorrean: *training set*, *validation set* eta *test set*. Banapen hori egiterako orduan, jarraibide orokor batzuk daude. Datu kopurua 100, 1.000 edo 100.000 ingurukoa baldin bada, datuen %70-%80-a *training set* barruan sartzen da eta geratzen den %30-%20-a *validation set* eta *test set*-en artean erdibituz. Datu kopurua handiagoa den kasuetan, 1.000.000 elementutik gorakoak adibidez, %98 inguru erabiltzen da *training set* bezala, %1-a *validation set* bezala eta %1 *test set* bezala.

Training set-aren bidez modeloa sortzen da, erregresiorako zein sailkapenerako. Kasu batzuetan ordea, eredu horiek sarrerako datuetara gehiegi doitzen dira (overfitting) eta ondorioz ez dute datu berrietarako balio. Irudia 1.15-ko eredu urdina horren adibide garbia da.

Validation set-a, kasu horiek ekiditeko erabiltzen da hain zuzen ere. Datu multzo honen bidez, eredu desberdinak testatzen dira **overfitting**-a ematen den edo ez aztertzeko. Eredua aukeratzeko, taula edo grafiko bat marrazten da *training set*-ean eta *validation set*-ean sortutako erroreekin (Irudia 1.16) eta bi erroreak ahalik eta txikienak izatea bilatzen da.



Irudia 1.16: Grafiko honetan, Irudia 1.15-eko datuekin lortutako erroreak (Bataz besteko errore koadratikoa) agertzen dira erabilitako funtzio polinomikoaren mailaren arabera. Irudia 1.15-n ikusten den bezala, 1. mailako polinomioak, 8. mailakoarekin konparatuz errore handiagoa sortzen du *training set*-ean, baina txikiagoa *validation set*-ean.

Irudia 1.16-n ikus daitekeen bezala, eredua gero eta konplexuagoa den heinean, *training set*-ean ematen den errorea gero eta txikiago da, datu horiek era zuzenean

kokatzeko gaitasuna duelako. Bestalde, *validation set*-ean ematen den errorea aztertzen bada, 3. mailatik aurrera errorea dezente handitzen dela ikusten da. Hori da hain zuzen ere overfitting-aren adierazpen grafikoa.

Esan bezala, helburua overfitting-a ekiditea eta sortutako erroreak minimizatzea denez, maila 1 edo 2 duen eredua aukeratuko zen kasu honetan. Orokorrean, *validation set*-ean errorea minimizatzen den eredua aukeratzen da. Baldintza hori betetzen duten hainbat eredu baldin badaude, *training set*-eko errore txikiagoa duena aukeratzen da.

Behin eredua aukeratutakoan, eredu horren kalitate testa egiteko *test set*-a erabiltzen da. *Validation set*-arekin alderatuz, errore finala kalkulatzeko bakarrik erabiltzen da eta ez da erabiltzen ereduaren aukeraketa egiterako orduan.

Testen inguruan informazio gehiago nahi duen irakurleak [17] liburuko 5. kapituluan aurki dezake. Bertan, **cross-validation** bezalako testen inguruko informazioa agertzen da, baita errorea neurtzeko erabiltzen diren forma ezberdinak ere.

Kapitulua 2

Unsupervised Learning

Unsupervised Learning, *etiketarik* gabeko datuen ezkutuko egitura inferitzeko Machine Learning-eko zeregin bat da. Zeregin horretarako, datuek ez dute inongo klase, kategoria edo sailkapenik eta Supervised Learning-ekin alderatuz, datuetatik inferitutako ereduak ebaluatzeko metodoak subjektiboagoak dira.

Era askotako algoritmoak eta metodoak sartzen dira talde honetan, baina orokorrean harturik bi zeregin nagusi daude *etiketaturik* gabeko datuekin: **datuak taldekatzea** eta **datuen dimentsioa murriztea**.

Lehenengoen kasuan, seguruenik ezagunena den kasua **clustering**-arena da (2.1 azpi-kapituluan ikus daitekeena), non datuak taldekatu egiten diren beraien ezaugarri komunengatik. Beste kasu ezagun bat **anomalien detekzioa** izan daiteke. Metodo horietan, helburua jasotako datuetatik *urruti* dagoen edo *arraroa* den datua edo datu sorta identifikatzea da. Koaderno honetan datuak taldekatzearekin zerikusia duten beste 2 azpi-kapitulu daude: **Matrize Faktorizazioa** (2.3 azpi-kapitulua) eta **Association Analysis** (2.4).

Bestalde, **datuen dimentsioaren murrizketaren** inguruan **Osagai Nagusien Analisian** (2.2 azpi-kapituluan) azaltzen da adibideetako bat. Datuen dimentsioa murriztearen jomuga, datuen interpretazioa handitzea eta behar ez den informazioa baztertzea da. Beste era batean esanda, datuen ulergarritasuna eta sinpletasunari ematen zaio garrantzia.

Atal honetan ikusiko den bezala, Unsupervised Learning-eko ereduak askotan Supervised Learning-eko ereduen antzera landu daitezke, aldaketa txiki batzuk eginda, nahiz eta ereduen kalitatea edo egokitasuna neurtzeko beste teknika batzuk erabili beharko diren.

2.1 Cluster

Clusterrak osatzeko garaian, esan bezala sarrerako datuak inongo *etiketarik* edo sailkapenik gabe ematen dira, eta helburua, datu horiek multzokatzea da beraien *gertutasun* edo *antzekotasunen* arabera.

Matematikoki esanda, X espazioko $x_1, \ldots, x_n$ n datuak, $K \in \mathbb{N}$ multzo desberdinetan sailkatzea izango da helburua, distantzia txikira dauden puntuak cluster

berean sartuz. Gauzak errazteko asmoarekin, aldagaiak jarraituak direla suposatuko da eta $X = \mathbb{R}^d$ espazio bektoriala dela (adibideetan $X = \mathbb{R}^2$ erabiliko da).

Eratzea nahi diren multzo kopuruaren aukeraketa, dezente subjektiboa izan daiteke eta kasuaren arabera aztertu beharreko gauza bat da, nahiz eta batzuetan lortu beharreko cluster kopurua aurretik finkatuta etortzen den (Demagun adibidez EAEn jasotako neurketa desberdinekin, herri bakoitzeko begetazio mailari 1etik 5erako nota bat ezartzea eskatzen dela). Hurrengo metodoaren barnean K cluster kopurua aukeratzeko modu bat azaltzen da.

Irudia 2.1: Ezkerreko irudian sarrerako datu multzoa ageri da eta eskuinekoan aldiz, datu multzo hori 2 clusterretan banatuta. Eskuineko irudian agertzen diren 2 gurutze beltzak, 2 cluster horien μ_1 eta μ_2 zentroideak dira.

K-Means metodoa

Clusterrak osatzeko metodo erraz, sinple eta erabilienetariko bat ***K*-Means metodoa** da. Metodo honen ideia nagusia μ_k puntu jakin batzuen (zentroide izenekoak) inguruan dauden puntuak k. taldera batzen datza eta zentroide horiek honela definitzen dira:

$$\mu_k = \frac{1}{n_k} \sum_{i=1}^{n_k} x_i \mathbb{1}_{\{c_i=k\}} \in \mathbb{R}^d \tag{2.1}$$

non n_k, k. taldeko elementu kopurua den, $\mathbb{1}$ funtzio adierazlea eta $c \in \{1, \ldots, K\}$ talde adierazlea:

$$c_i = k \Leftrightarrow x_i \text{ elementua } k. \text{ clusterrean dago} \tag{2.2}$$

Algoritmoaren helburua $\boldsymbol{c} = (c_1, \ldots, c_n)$ eta $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)$ optimoak lortzea da, puntu guztiak bere clusterretik ahalik eta *gertuen* egoteko. Matematikoki idatziz

gero, honako hau lortu nahi da:

$$\arg\min_{\boldsymbol{\mu},\boldsymbol{c}} \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{1}_{\{c_i=k\}} d(x_i, \mu_i) \tag{2.3}$$

non d erabili nahi den distantzia neurria den.

Minimizazio problema hori ordea, *NP-Hard*[1] [11] problema bat da, hau da, konputazionalki oso konplexua. Arazo horri aurre egiteko errazagoak diren beste algoritmo batzuk erabiltzen dira, **minimo lokalak** lortzen dituztenak eta erabilienetako bat **Lloyd-en algoritmoa** da. Algoritmo horretan, zentroideak iterazio bidez lortzen dira eta 2 oinarrizko pausu ditu: zentroideak birdefinitu eta taldeak birdefinitu (konbergentzia lortu arte).

Algoritmoaren ideia intuikorra ulertzeko, interneten ehunka bideo eta animazio ezberdin aurki daitezke. Adibide bezala hurrengo link-a uzten da:

`https://www.youtube.com/watch?v=5I3Ei69I40s`

Kontutan izan, orokorrean K-ren balioa gero eta handiagoa denean, orduan eta balio txikiagoa izango dela (2.3)-ren balioa. Ondorioz, K aukeratzeko era bat, K-ren balioak banaka-banaka handitzen joatea da eta (2.3)-ren balioa *gutxi* txikitzen den momentuan, azkeneko K balioarekin geratuz. Adibide bezala Irudia 2.2 hartzen baldin bada, kasu horretan 3 cluster sortzea gomendatuko zen, hain zuzen ere hortik aurrera errorean ematen den hobekuntza txikitzat jotzen delako. Metodo hau ordea gida edo orientazio moduan erabili behar da, lortu nahi diren helburuen arabera cluster gehiago edo gutxiago sor baitaitezke.

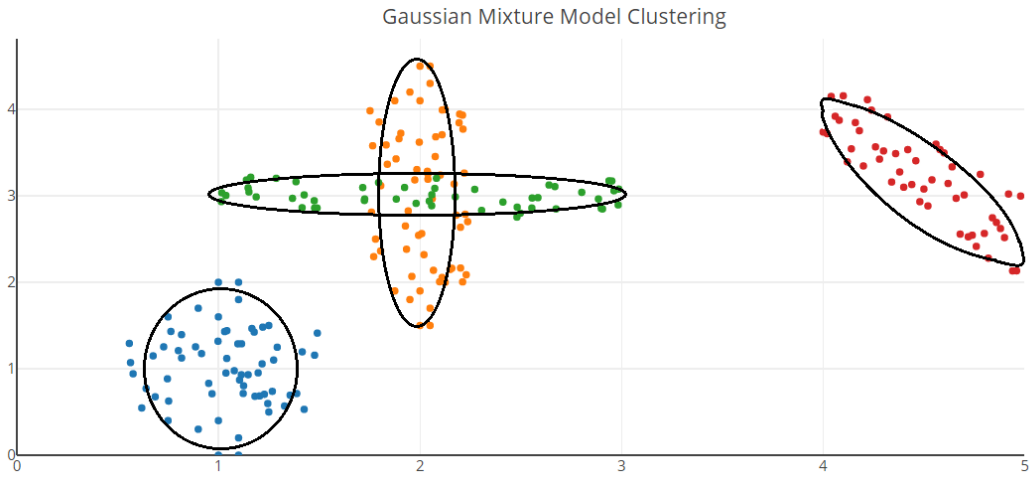
Irudia 2.2: Grafiko honetan, K cluster kopurua handitzearekin batera, trinkotasuna handitzen dela aztertzen da, hau da, cluster bakoitzeko elementuak gertuago daude beraien artean.

[1] *NP-Hard*-n inguruan ideia orokor bat egiteko:
`http://www.geeksforgeeks.org/np-completeness-set-1/`

Metodo probabilistikoak

Hortaz aparte, clusterrak sortzeko **metodo probabilistikoak** ere daude. Lortzea nahi dena berdina da, datuetatik K talde lortzea, baina datu horien inguruko suposizio batzuk egiten dira eta horrela inferentzia estatistikoa erabiltzeko aukera ematen dute.

Suposizio bezala, jasotako datuak banaketa normal ezberdinetatik datozela kontsideratzen bada (kasu hori **Gaussian Mixture Model** (GMM)[2] bezala ezagutzen da), lortzea nahiko den cluster-en banapena $N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ erakoa izango da.



Irudia 2.3: Grafiko honetan, 4 cluster daudela ikusten da eta cluster bakoitzeko puntuak banaketa normal batetik lortutakoak direla suposatzen da $N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$.

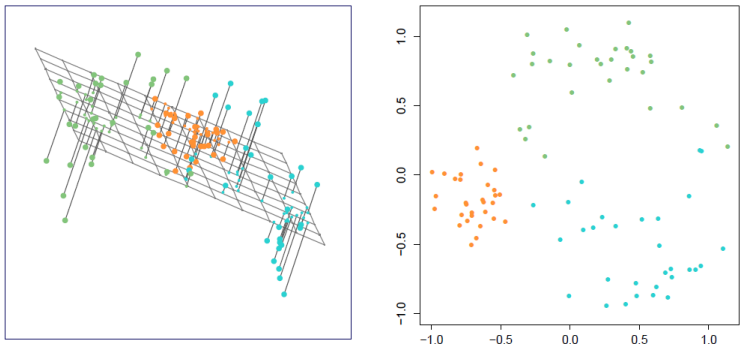
GMM-ren alde ona, esan bezala inferentzia estatistikoa erabil daitekeela da, ondorio eta egitura indartsuagoak atera ahal izateko. Hala ere, lortzen diren clusterrak forma jakin batekoak izango dira. Irudia 2.3 aztertzen bada, 2 dimentsiotan lortzen diren clusterrak elipsoide formakoak izango dira, banaketa normala hartzen delako abiapuntutzat.[3]

2.2 Osagai Nagusien Analisia

Osagai Nagusien Analisia (*Principal Component Analysis* (PCA) ingelesez), sarrerako datuen dimentsio erredukzioa egiteko erabiltzen den metodoa da batik bat. Suposatu sarrerako datuak $x_1, \ldots, x_n \in \mathbb{R}^d$ direla, d, elementu bakoitzaren aldagai kopurua izanik. Askotan, dimentsio edo aldagai horietako batzuk informazio berri oso gutxi eskaintzen dute edota beraien artean kolinealtasuna egoten da. Hori dela eta, beharrezkoak ez diren aldagaiak ezabatzea desiragarria izango da memoria aurrezteko eta konputazio denbora murrizteko.

[2]GMM-ren atzean dauden matematikak ezagutzeko [2] liburuko 9.2 atalera jo daiteke.

[3]Python programazio-lengoaiak pakete ugari ditu clusterrak era ezberdinean eratzeko [27]: `http://scikit-learn.org/stable/modules/clustering.html`



Irudia 2.4: Ezkerreko irudian, sarrerako datuak eta eskuinekoan berriz PCA aplikatu eta gero lortutako emaitza.

Irudia 2.4 begiratu ezkero, PCA algoritmoak egiten duena ikus daiteke era intuikor batean. Sarrerako datuek, 4 dimentsio dute: 3 espazialak eta bat kolorearentzat. Oraingoan, puntu horien sailkapen bat lortzea nahi da eta hori lortzeko bi aukera daude. Aukera bat, zuzenean lanean hastea da nahi den sailkapen metodoren batekin. Beste aukera bat, dimentsio erredukzioa aplikatzea da eta hori egin ondoren aplikatzea sailkapen metodoa. Eskuineko irudian ikus daitekeenez, dimentsio espazial bat ezabatu da, informazio kantitate esanguratsu bat eta hala ere sailkapena era egokian egiteko aukera dago.

PCA, Balio Propioen Deskonposizioan[4] oinarritzen da. Datuen $X \in \mathbb{R}^{n \times d}$ matrizea eratzen bada:

$$X = \begin{bmatrix} — & x_1 & — \\ & \vdots & \\ — & x_n & — \end{bmatrix}$$

XX^T matrize **erdidefinitu positiboa** izango da eta ondorioz $r \leq \min\{d, n\}$ balio eta bektore propio izango ditu.

$$\text{Balio propioak: } \lambda_1 \geq \lambda_2 \geq \ldots \lambda_r \geq 0$$
$$\text{Bektore propioak: } q_1, q_2, \ldots, q_r$$

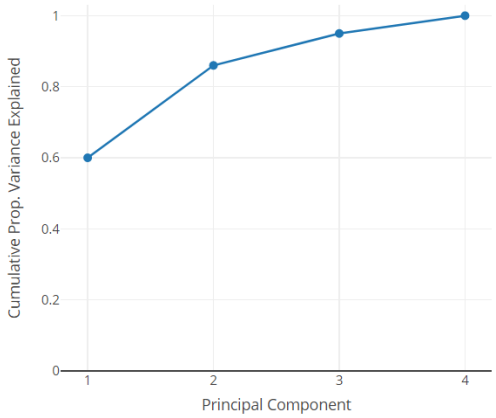
Aztertutako metodo gehientsuenetan bezala, aldagaien normalizazioak garrantzi handia du, bestela balio handiko aldagaiek besteekiko pisu handiagoa izango dute. Nola jakin daiteke ordea zenbat aldagai edo dimentsio mantendu?

Galdera horrentzako ez dago erantzun zehatz bat, baina orokorrean bariantza aztertzen da aldagai kopurua zehazteko. Ideia nagusia, datuen bariantza osotik aukeratutako aldagaiek azaltzen duten bariantza aztertzea da: azaldutako bariantza hori *handia* bada, aldagai horiekin geratzen da.[5]

Irudia 2.5-n ikus daitekeenez, datu sorta bati PCA aplikatu ondoren, aldagai edo dimentsio bakarrarekin bariantza totalaren %60-a baino gehiago azaltzen da, 2 aldagairekin %80-a baino gehiago eta 3 aldagai edo gehiagorekin %90-a baino gehiago.

[4]Balio Propioen Deskonposizioaren inguruan informazio gehiago nahi izanez gero, aljebra linealeko liburu gehientsuenetan aurki daiteke, [33] liburuko 5 atalean adibidez.

[5]Bariantzaren zehaztasun matematikoak [17] liburuko 10.2.3 atalean aurki daitezke.



Irudia 2.5: PCA-ren bidez aldagai kopuru desberdinekin azaltzen den bariantza pilatua ehunekotan.

Hori ikusita, zenbat aldagai kopuru hartu erabaki behar da eta horretarako ere ez dago arau zehatzik. Era subjektiboan egiten da kasuaren arabera eta begiratzen diren puntuak honako bi hauek dira batik bat: ehuneko bariantza *handi* bat azaltzea eta hortaz gain aldagai berriek bariantza *gutxi* handitzen dutenean. Adibideko kasuan, 2 aldagairekin geratzea nahiko arrunta litzateke.

2.3 Matrize faktorizazioa

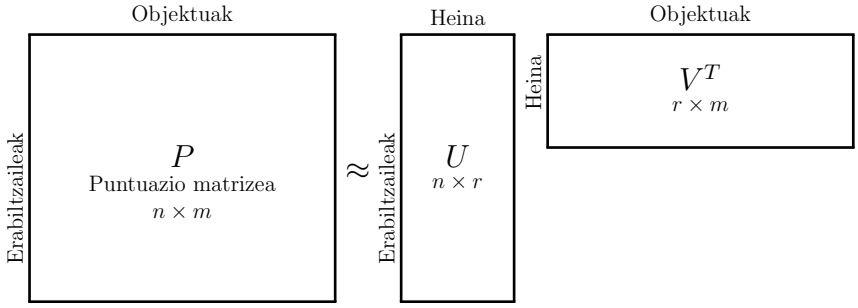
Gaur egunean, merkatuaren zati handi bat interneten aurkitzen da: pelikulak, musika, arropa, janaria eta elektronikako aparatuak erosi/alokatu daitezke besteak beste. Erosi/alokatu daitezkeen gauzen lista ordea, oso handia da eta ia ezinezkoa da dena begiratzen eta aztertzen hastea. Hori dela eta, azken urteetan bereziki garrantzitsu bilakatu dira **gomendio sistemak**.

Gomendio sistema horietan erabiltzen den metodo bat **collaborative filtering** metodoa da. Kasu hauetan, gehienetan *erabiltzaileak* egongo dira alde batetik eta *objektuak* bestetik. Lortzea nahi izaten dena, erabiltzaileek objektu desberdinei emandako puntuazioak gorde eta horietatik ikastea da. Horrela, algoritmoa objektu berriak gomendatzeko gai izango da beste erabiltzaileen puntuazioa aztertuta. Ideia nagusia honakoa da: bi pertsonek puntuazio oso berdintsua eman baldin badute hainbat objektu ebaluatzerako orduan, beste objektuetan ere antzeko puntuazioa emango dute seguruenez. Erabiltzaileen eta objektuen arteko taldekatze sistema bat bilatzen da funtsean.

Matrize faktorizazioa, **collaborative filtering** metodorako erabiltzen den algoritmo bat da hain zuzen ere.

Algoritmoaren ideia

Metodo honetan, gauzak sinplifikatuz, hasieran P matrize bat osatzen da. Matrize horren zutabeetan objektuak jartzen dira eta lerroetan erabiltzaileak eta P



Irudia 2.6: Matrize faktorizazioaren eskema

matrizeko elementuak, puntuazioak izango dira. Beste era batean esanda, P matrizeko p_{ij} elementua, i erabiltzaileak j elementuari eman dion puntuazioa izango da (1etik 5era, 1etik 10era edo dena delakoa). Matrize horretan elementu gehienak hutsik egongo dira, erabiltzaile bakoitzak objektu gutxi batzuk ebaluatzen dituelako bakarrik, eta algoritmoaren helburua hutsune horiek era egokian estimatzea izango da.

Hori lortzeko, Balio Singularretako Deskonposizioaren[6] ideia erabiltzen da (SVD siglak erabiltzen dira ingelesez). Balio Singularretako Deskonposizioaren bitartez, matrize baten balio propioak lortzen dira, matrizearen *funtsa* nolabait.

Gomendio sistemetarako ordea, ezin da zuzenean SVD deskonposizioa erabili, P matrizearen balio gehienak faltako direlako, % 99 inguru. Hori dela eta, Irudia 2.6-ko eskema bat lortzen saiatu beharko da, non $p_{ij} \approx u_i^T \cdot v_j$ izango den eta minimizatzea nahiko den errorea:

$$\sum_{(i,j)\in\kappa} (p_{ij} - u_i^T \cdot v_j)^2 \tag{2.4}$$

κ multzoan P matrizeko osagai ez-nuluak egongo dira. Hortaz gain, errorea kalkulatzeko orduan erregularizazioa kontutan hartuz gero (erregularizazioaren inguruan 1. kapituluan ageri da informazio gehiago, 1.1.1. atalean definitzen delarik), honakoa izango da minimizatu beharreko errorea:

$$\sum_{(i,j)\in\kappa} (p_{ij} - u_i^T \cdot v_j)^2 + \lambda(\|u_i\|^2 + \|v_i\|^2) \tag{2.5}$$

Errore hori minimizatzeko, bi algoritmo nagusi daude *Stochastic Gradient Descent* eta *Alternating Least Squares*[7]. Hemen, *Alternating Least Squares*-en ideia nagusiak azalduko dira.

- Hasteko, V^T matrizea hasiarazten da v_j bektoreak zoriz aukeratuz $N(0, \lambda^{-1}I)$ banapen batetik.

[6]SVD deskonposizioaren inguruan informazio gehiago nahi izanez gero, aljebra linealeko liburu gehientsuenetan aurki daiteke, besteak beste [33] liburuko 6.3 atalean.

[7]Algoritmoaren inguruko informazio gehiago [20] artikuluan aurki daiteke.

- Ondoren, V^T matrizeko balioak finkatuta daudela, U matrizearen balioak eguneratzen dira (2.5)-ko balioa ahalik eta txikiena izateko.
- U eguneratuta, U finkatuta utzi eta V^T matrizea eguneratu (2.5) balioa ahalik eta txikiena egiten duena.
- 2 eta 3 pausoak errepikatu konbergentzia lortu arte.

Oharrak

Matrize faktorizazioa aplikatzeko, hurrengo puntuak kontutan izan beharko dira:

- P puntuazio matrizeak, balio gehienak nuluak izan beharko ditu.
- P matrizearen heina bere dimentsioa baino askoz txikiagoa izan beharko da.
- Metodoak emaitza onak emateko, erabiltzaileen arteko korrelazioa izatea desiratuko da, P matrizeko lerroak korrelatuta egotea.
- Algoritmoarekin lortuko den $\widetilde{P} = U \cdot V^T$ matrizeko elementu batzuk balio *arraroak* izan ditzakete: balio negatiboak edo balio altuegiak (1etik 5erako balorazioetan 6.1 emaitza lortzea adibidez). Hori konpontzeko, balorazio sistemara bornatu beharko dira lortutako emaitzak.
- *Erabiltzaileak* eta *objektuak* erabiltzeaz gain, beste faktore batzuk gehi daitezke: bilaketa historiala, herrialdea, adina, etab. Kasu horietarako, **Tensore faktorizazioa**[8] erabiltzen da.
- Irudia 2.6-n agertzen den *Heina* edo r balioa, ez da P matrizearen heina, ezin daitekeelako kalkulatu era klasikoan. Ondorioz, r balio desberdinetarako testak egiten dira (10, 50, 100, 200 adibidez) eta baliorik *desiragarrienarekin* geratu (errore/kostu-konputazional erlazio onenarekin adibidez).

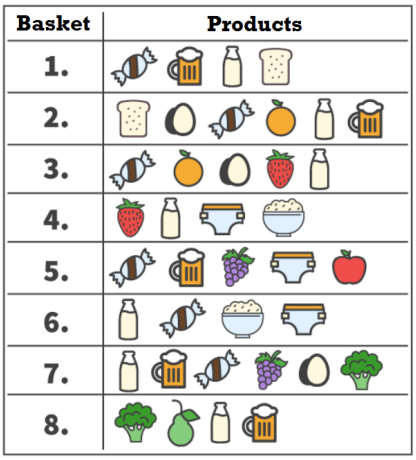
2.4 Association Analysis

Association Analysis, gehienetan dendekin lotzen den metodoa da. Helburua, erosketa zesta batean sarritan batera doazen produktuak identifikatzea da. Adibide bezala, pentsa daiteke ordenagailu eramangarri bat erosten duenak seguruenik ordenagailuarentzako maleta bat ere hartuko duela edo interesa dakiokeela. Horregatik, interesa egon daiteke erlazio horiek identifikatzea eta erosleari pack edo bilduma batean saltzen saiatzea.

Ondoren ikusiko den bezala, oinarrizko matematika erabiltzen duen metodoa da, ulerterraza eta datuen aurreprozesaketa oso urria behar duena.

Irudia 2.7 begiratzen bada, bertan erosketa zesta desberdinak agertzen dira eta batik bat, honako 2 analisi hauek egitea nahi da:

[8] [30] liburuaren 19.4.2.1 atalean aurki daiteke informazio gehiago.



Irudia 2.7: Erosketa zesta desberdinen adibide bat.

- **Association analysis:** Produktu desberdinekin osaturiko azpimultzoak aztertzen datza. Demagun Irudia 2.7 adibidean {Garagardoa, Esnea} produktuen arteko lotura aztertzea nahi dela. Horretarako, zestaren probabilitatea honela kalkulatzen da:

$$P(\{\text{Garagardoa, Esnea}\}) = \frac{\{\text{Garagardoa, Esnea}\}\text{ duten zesta kopurua}}{\text{Aztergai dauden zesta kopuru totala}} = \frac{4}{8} = 0.5$$

- **Association rules:** Produktu edo objektuen arteko korrelazioa neurtzen du nolabait. Demagun Irudia 2.7 adibidean A ={Garagardoa, Esnea} erosi ezkero, B ={Gozokiak} erosteko probabilitatea neurtzea eskatzen dela:

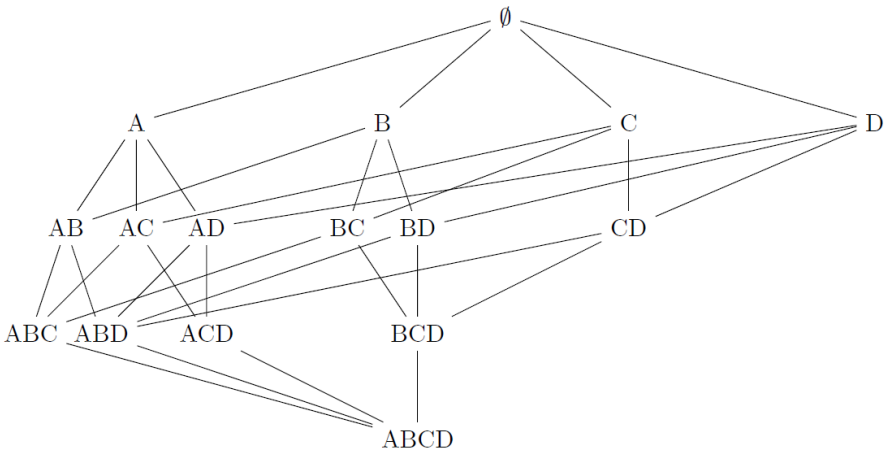
$$P(B|A) = \frac{\{\text{Garagardoa, Esnea, Gozokiak}\}\text{ duten zesta kopurua}}{\{\text{Garagardoa, Esnea}\}\text{ duten zesta kopurua}} = \frac{3}{4} = 0.75$$

  Bestalde, A ={Garagardoa, Esnea} produktuen **garrantzia** azter daiteke B ={Gozokiak} erosteko garaian:

$$L(A,B) := \frac{P(B|A)}{P(B)} = \frac{3/4}{6/8} = 1$$

  Kasu horretan, 1 horren esanahia honakoa da: probabilitate berdina dago {Goxokiak} erosteko {Garagardoa, Esnea} erosi edo ez erosi ($L(A,B) =$ 2 izango balitz adibidez, esanahia desberdina izango zen: 2 aldiz probableagoa izango zen {Goxokiak} erostea {Garagardoa, Esnea} erosi ezkero, {Garagardoa, Esnea} ez erostearekin alderatuz).

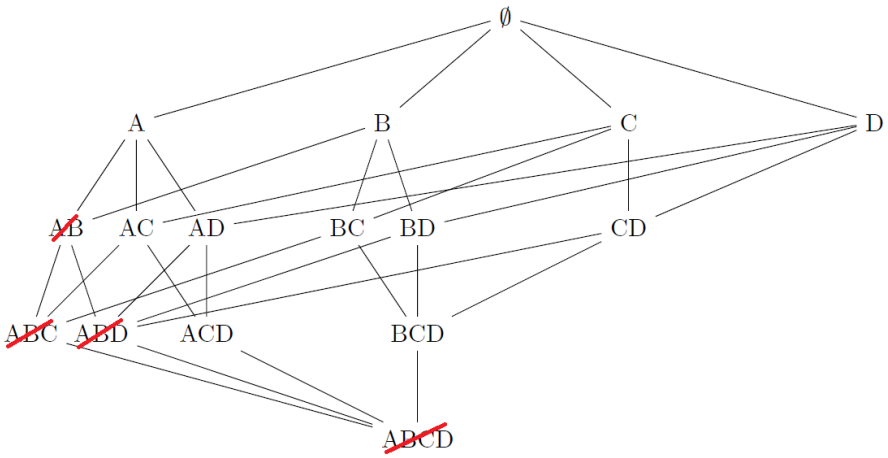
Batik bat horiek dira analisi honetarako erabiltzen diren tresnak. Hala ere, datu kopurua oso handia bada, analisi hori egitea ezinezkoa da konputazionalki. Ez da posible izango objektuen azpimultzo guztiak aztertzea eta objektu guztien arteko



Irudia 2.8: A, B, C eta D objektuekin osatutako zesta posible guztien grafoa.

erlazioa ikustea. Konponbide bezala, **Apriori Algoritmoa** erabiltzen da. Algoritmoaren funtsa, eraikitzen doan zuhaitz erraldoi hori mozten joatea da probabilitatea txikiegia denean.

Irudia 2.8-n ikusten den grafoan, zuhaitz osoa marraztu da, baina suposatu nahi dena zestaren probabilitatea $t \in [0, 1]$ balio bat baino handiagoa izatea dela eta $P(AB) < t$ dela. Hori horrela izanik, AB zesta ezabatuko da, baina ez hori bakarrik, AB zestako produktuen konbinazioz osaturiko zesta guztiak baizik, horien probabilitatea ere t baino txikiagoa izango delako.



Irudia 2.9: Apriori Algoritmoaren eskema

Horiek denak ezabatu ezkero, Irudia 2.9-ko grafoa lortuko da, hau da, hainbat zesta ezabatuko ziren aldez aurretik, hori da hain zuzen ere **Apriori Algoritmoaren** ideia[9].

[9]Gaiaren eta algoritmoaren informazio zehaztuagoa [15] liburuko 14.2 atalean aurki daiteke.

Kapitulua 3

Sequential Data

Kapitulu honetan, egitura berezi bat duten datuak aztertuko dira: **sequential data**. Datu mota horiek orokorrean ordenaturik egoten dira era espezifiko batean eta ordena horrek garrantzia handia hartzen du. Donostiako eguraldiaren adibidearekin sartuz gero, eguraldiaren neurketa ezberdinak (prezipitazioa, tenperatura, presio atmosferikoa...) denbora ezberdinetan zehar egiten dira eta lortutako datu horiek kronologikoki ordenatzen dira gehienetan. Adibide horretan, ordenak eta kronologiak garrantzia berezia hartzen du, ez baita berdina azaroan edota ekainean egitea neurketak. Aldi berean, eguraldia aurresateko garaian inportantzia nabarmena izaten du aurreko egunetan gertatukoak.

Burtsan ere, akzioen balioak kronologikoki ordenatuta egotea ia derrigorrezkoa da analisi egokiak lortzeko eta aurreikuspenak egiteko. Hori dela eta, 3. kapituluan horrelako datuekin egiten diren analisi ezberdinak aztertuko dira.

Horretarako, **Markov-en kateak** eta **Hidden Markov Model**-ak aztertuko dira. Hurrengo azpi-kapituluetan ikusiko den bezala, metodo horien bidez besteak beste sailkapen edo/eta cluster-ak egiteko aukera ematen dute. Halere, 1. eta 2. kapituluetan ez bezala, metodo hauen bidez ematen diren aukerak askoz anitzagoak eta orokorragoak dira eta horregatik ez dira Supervised Learning ezta Unsupervised Learning-en barruan sartu.

3.1 Markov-en kateak

Markov-en kateak, Andrey Markov matematikari errusiarraren omenez izendatutakoa, *memoriarik gabeko* datu sekuentziari deitzen zaio. Beste era batera esanda, datu sekuentzia horretan pausu bakoitzean eman den emaitzak, aurreko azken emaitzarekiko dependentzia bakarrik azaltzen du[1]. Hurrengo datu segida kontsideratzen bada:

$$\{x_1, x_2, \ldots, x_i, x_{i+1}, \ldots, x_n\}, \qquad x_i \in \mathbb{R}$$

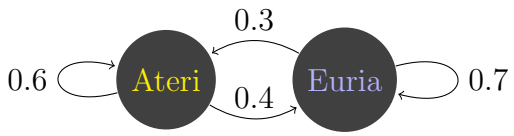
[1]Markov-en kateen definizioa orokortu egin daiteke, aurreko azken m elementuekiko dependentzia emanez. Kasu horietan, m **memoriadun Markov-en kateak** deitzen dira.

non x_{i+1} balioak x_i balioarekiko dependentzia bakarrik duen, $i \in \{1, \ldots, n-1\}$ izanik. Definizioa, orokorrean era matematiko-probabilistikoan aurkezten da:

$$P(x_{i+1}|x_i, \ldots, x_2, x_1) = P(x_{i+1}|x_i), \quad \forall i \in \{1, \ldots, n-1\}$$

Adibide bezala, eguraldiarena jarriko da. Gauzak asko sinplifikatuz, demagun gaurko eguraldia atzoko eguraldiarekiko dependentea bakarrik dela eta hurrengo probabilitateak lortu direla:

$$\begin{aligned}
&P(\text{“Bihar ateri”} \mid \text{“Gaur ateri”}) = 0.6\\
&P(\text{“Bihar ateri”} \mid \text{“Gaur euria”}) = 0.3\\
&P(\text{“Bihar euria”} \mid \text{“Gaur ateri”}) = 0.4\\
&P(\text{“Bihar euria”} \mid \text{“Gaur euria”}) = 0.7
\end{aligned}$$



Probabilitate horiekin, **trantsizio matrizea** edo **Markov matrizea** eraiki daiteke hurrengo eran:

$$M = \begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$$

Trantsizio matrizeko lerro bakoitzeko elementuen batura 1 izango da beti, probabilitateen propietateengatik eta M_{ij} elementua, i egoeratik j egoerara pasatzeko probabilitatea izango da. Aurreko kasuan, M_{12} elementua atzo eguzkia egin zuela jakinik gaur euria egiteko probabilitatea da.

Markov-en kateetan M trantsizio matrizea oso garrantzitsua da, katea guztiz definitzen duelako. Matrize horren balioak lortzeko, kasuaren arabera era desberdinetan egiten da. Batzuetan, M_{ij} elementuak aurrez definiturik egongo dira edota erabiltzaileak definitu beharko ditu (aurreko adibidean aurrez definiturik zeuden). Besteetan aldiz, estimatu egin beharko dira egiantz handieneko metodoa erabiliz sarrerako datuei.

Gainera, banaketa egonkorra (*stationary distribution* ingelesez)[2] lortzeko ezinbesteko elementua da eta hurrengo definizioak garrantzia handia hartzen du aplikazioetarako:

$$M^\infty := \lim_{n \to \infty} M^n$$

[2]Banaketa egonkorraren eta trantsizio matrizearen propietateak eta informazio gehiago [22] liburuko 1. atalean aurki daiteke.

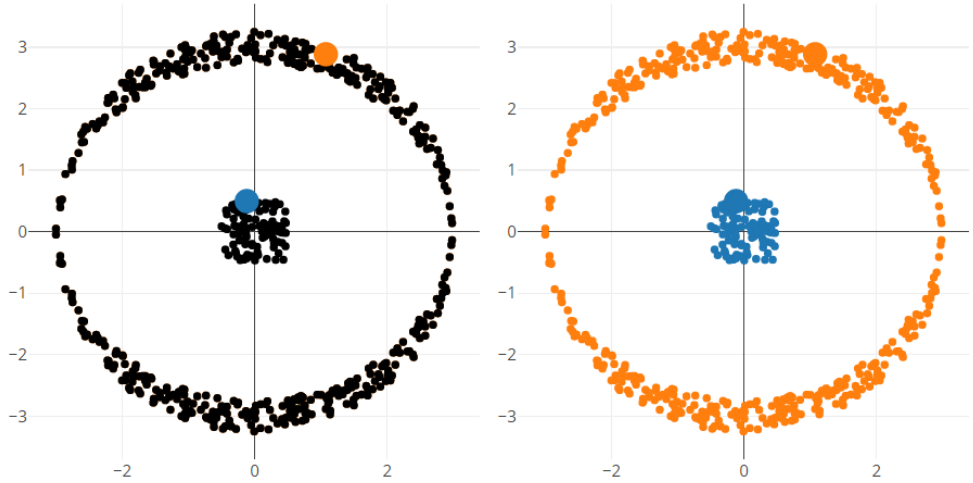
Aplikazioak

Semi-supervised classification:[3]

Irudia 3.1-n agertzen den kasuetarako balio du: sarrerako puntu guztietatik gutxi batzuk bakarrik sailkatuta daudenean. Kasu hauetan, sailkatu gabeko puntu bat hartzen da eta zorizko bide baten bidez puntu batetik bestera jauziz joaten da sailkatutako punturen batera heldu arte eta gero talde berean sailkatzen dira[4]. Puntu batetik bestera jauzi egiteko probabilitatea era askotan definitu daiteke, baina kasu hauetan distantziaren arabera definitzen da: gertuko puntuek probabilitate handiagoa izango dute urrunago daudenekin alderatuz.

Hortaz gain, hasieratik sailkatutako puntuak (Irudia 3.1-n agertzen diren puntu urdina eta laranja) **puntu xurgatzaileak** izango dira, hau da, behin bertara iritsitakoan, ezingo da bertatik irten. x_{px} puntu xurgatzailea bada:

$$P(x_{i+1} = x_{px} | x_i = x_{px}) = 1$$



Irudia 3.1: Ezkerreko irudian, sarrerako datuak agertzen dira. Bi puntu agertzen dira sailkaturik, urdinez dagoena alde batetik eta laranjaz dagoena bestetik, gainontzekoak sailkatu gabe daude. Eskuinean, Markov-en kate bat aplikatu ondoren lortutako sailkapena agertzen da.

Ranking-ak eratzen:

Beste aplikazio posible bat, ranking bat eratzea da. Horretarako, trantsizio matrize bat sortzen da era artifizial batean helburua betetzeko. Adibide bezala, esku

[3]Hurrengo link honetan animazio bidez azaltzen da metodoa:
`https://www.datasciencecentral.com/profiles/blogs/a-semi-supervised-classification-algorithm-using-markov-chain-and`

[4]Zorizko bidea egin beharrean, M^{∞} matrizearen bidez kalkulatzen da puntu bakoitzaren sailkapena. j. elementuaren sailkapena M^{∞} matrizearen j. lerroan probabilitate handiena duen elementuarena izango da.

pilotako ranking bat era daiteke. Gauzak sinple hartzeko, demagun jokalari bat bestea baino hobea dela partida irabazten badu eta amaierako emaitzak, *zenbat aldiz hobea* den adierazten duela. Hau da, A jokalariak B jokalariari irabazten badio, A jokalaria B baino hobea izango da eta amaierako emaitzaren diferentzia handia izan bada gero eta hobea.

Pilotariak	A	B	C
A	X	22 - 18	22 - 11
B	18 - 22	X	22 - 20
C	11 - 22	20 - 22	X

Taula 3.1: Ikus daitekeenez, adibide honetan A pilotariak 2 partidak irabazi ditu eta C pilotariak aldiz 2 partidak galdu.

Horrela definiturik, trantsizio matrizean B jokalaritik A jokalarira mugitzeak probabilitate handia izango du eta A jokalaritik B jokalarira mugitzeak txikia. Behin M trantsizio matrizea definiturik, M^∞ matrizea kalkulatzen da. M^∞ matrizeko zutabeetako balio guztiak berdinak izango dira eta j zutabeko balioa j. pilotariaren balorazioa izango da: gero eta handiagoa, gero eta hobea.

Pilotarien adibiderako, ebaluatzeko sistema honela egin da:

$$M_{ij} = \frac{i \text{ eta } j\text{-ren aurkako partidan } j \text{ pilotariak eginiko tantuak}}{i \text{ pilotariaren partidetan galtzaileek egindako tantu kopuru totala} + 22}, \; i \neq j$$
$$M_{ii} = 1 - \sum_{j \neq i} M_{ij}$$

Ebaluatzeko sistema horrekin, ondorengo trantsizio matrizea lortzen da:

$$M = \begin{pmatrix} \frac{22}{18+11+22} & \frac{18}{18+11+22} & \frac{11}{18+11+22} \\ \frac{22}{18+20+22} & \frac{18}{18+20+22} & \frac{20}{18+20+22} \\ \frac{22}{11+20+22} & \frac{22}{11+20+22} & \frac{9}{11+20+22} \end{pmatrix} = \begin{pmatrix} \frac{22}{51} & \frac{18}{51} & \frac{11}{51} \\ \frac{22}{60} & \frac{18}{60} & \frac{20}{60} \\ \frac{22}{53} & \frac{22}{53} & \frac{9}{53} \end{pmatrix}$$

eta horren bidez aurretik definitutako M^∞ matrizea lortzen da:

$$M^\infty = \begin{pmatrix} 0.4047482 & 0.3496906 & 0.2455612 \\ 0.4047482 & 0.3496906 & 0.2455612 \\ 0.4047482 & 0.3496906 & 0.2455612 \end{pmatrix}$$

Beraz, lortutako ranking-aren arabera, A pilotaria izan da onena eta C pilotaria txarrena.

3.2 Hidden Markov Model

Zoritxarrez, askotan ezingo dira egoerak era zuzenean aztertu datuetatik. Suposa dezagun adibidez, Titan satelitean[5] metano euria noiz ematen den aztertzea nahi dela. Demagun infragorri-teleskopio baten bidez azterketa hori egiten dela eta ikerketa batean lortutako emaitzek ondokoa diotela:

$$P(\text{“Ateri”} \mid \text{Infragorri maila} < T_0) = 0.2$$
$$P(\text{“Euria”} \mid \text{Infragorri maila} < T_0) = 0.8$$
$$P(\text{“Ateri”} \mid \text{Infragorri maila} \geq T_0) = 0.9$$
$$P(\text{“Euria”} \mid \text{Infragorri maila} \geq T_0) = 0.1$$

non T_0 balioa ikerketan lortutako balio ezagun bat den. Kasu honetan, eskura dauden tresnekin ezin daiteke egoera zehatza zein den jakin, infragorri maila bakarrik. Demagun hortaz gain, egindako beste ikerketa batzuen arabera, hurrengo probabilitateak lortu direla:

$$P(\text{“Bihar ateri”} \mid \text{“Gaur ateri”}) = 0.6$$
$$P(\text{“Bihar ateri”} \mid \text{“Gaur euria”}) = 0.3$$
$$P(\text{“Bihar euria”} \mid \text{“Gaur ateri”}) = 0.4$$
$$P(\text{“Bihar euria”} \mid \text{“Gaur euria”}) = 0.7$$

Lortu nahi dena Titan sateliteko eguraldiaren gorabeherak estimatzea eta aurresaten saiatzea izango da jasotako datuekin (infragorri mailaren neurketaren bidez):

$$\{t_1, t_2, t_3, \ldots, t_n\} \quad \text{Infragorri mailaren } n \text{ neurketa.}$$

Kasu honetan, 2 matrize izango dira, **trantsizio matrizea**:

$$A = \begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$$

eta **emisio matrizea**:

$$B = \begin{bmatrix} 0.2 & 0.8 \\ 0.9 & 0.1 \end{bmatrix}$$

eta izan bedi lehenengo egunean euria edo ateri egiteko hasierako probabilitate banaketa:

$$\pi = [0.5 \quad 0.5]$$

Jarri den adibide honetan, A, B eta π matrizeak ezaguntzat hartu dira, baina kasu gehienetan ez da horrela izango eta matrize horiek estimatu beharko dira. Hortaz gain, kasu honetan, adibideko *Hidden Markov Model*-a eredu diskretu bat da, hau da, bertako *egoerak* diskretuak dira (euria edo ateri). Eredu diskretuetan 3 estimazio problema nagusi agertzen dira:

[5] Titan satelitearen inguruko informazioa hurrengo web orrialdean aurki daiteke:
`https://www.space.com/15257-titan-saturn-largest-moon-facts-discovery-sdcmp.html`

- $\{A, B, \pi\}$ matrizeak izanda *HMM* diskretu baterako eta sarrerako $\{t_1, t_2, \ldots, t_n\}$ datuen segida, t_i bakoitzaren *egoeraren* estimazioa lortzea. Problema horri aurre egiteko, **Forward-Backward Algoritmoa** erabiltzen da.

- $\{A, B, \pi\}$ matrizeak izanda *HMM* diskretu baterako eta sarrerako $\{t_1, t_2, \ldots, t_n\}$ datuen segida, probabilitaterik handiena duen $\{s_1, s_2, \ldots, s_n\}$ *egoera segida* lortzea. Problema horretarako, **Viterbiren Algoritmoa** erabiltzen da.

- Aztertutako $\{t_1, t_2, \ldots, t_n\}$ segida emanik, $\{A, B, \pi\}$ matrizeak estimatzea. Horretarako, **egiantz handieneko** metodoa erabiltzen da.

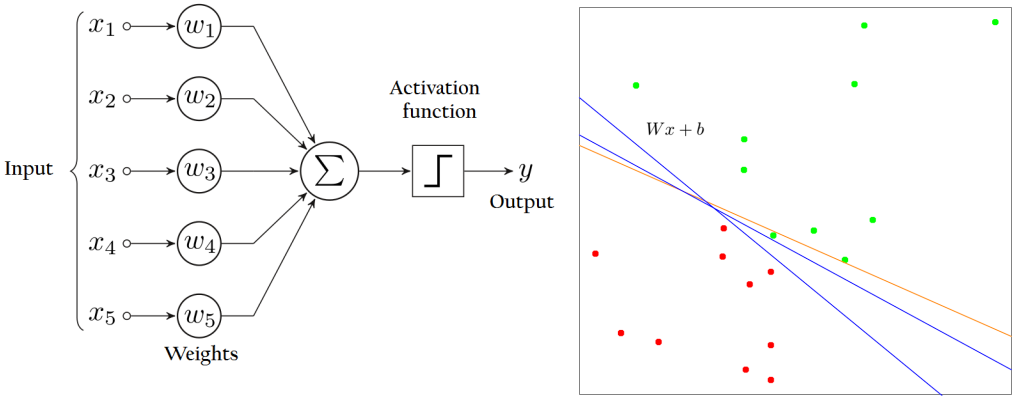
HMM-en atzean, matematika eta teoria sakon asko dago hemen aztertuko ez dena, ez delako lan honen helburua. Hala ere, interesaturik dagoen irakurlearentzat, [29] artikulua eta [4] testuliburua oso erabilgarriak izan daitezke gai honetan gehiago sakontzeko. Bertan, *HMM* jarraituak ere agertzen dira, hau da, *egoerak* diskretuak ez dituzten ereduak. Horiek, are eta konplexuagoak dira baina aplikazio interesgarriak ditu, kotxe autonomoarena besteak beste.

Kapitulua 4

Sare Neuronal Artifizialak

4.1 Sarrera historikoa

Sare neuronal artifizialak, 1940. hamarkadan hasi ziren aztertzen eta lan ezberdinak aurkezten. Horiek, kalkulu aritmetiko sinpleak egiteko gai ziren, baina mugapen handiekin. Aurrerapen garrantzitsuak 50. eta 60. hamarkadan etorri ziren, *Perceptron*-aren algoritmoarekin (neurona baten funtzioa imitatzen saiatzen dena), sare neuronal artifizialen aitzindaria. Algoritmo honen berritasun nabaria, parametro egokiak bere kabuz aukeratzeko gai zela izan zen. Irudia 4.1-n ikus daitekeen bezala, sarrerako datuak klase bitar batean daudela suposatuko da (irudiko kasuan berdea edo gorria) eta helburua puntu horiek hiperplano baten bidez sailkatzea dela. Horretarako, algoritmoa $W = (w_1, \ldots, w_n)$ pisuak *zuzentzen* joaten da iterazio bakoitzeko helburua lortzen duen arte (posible den kasuetan).



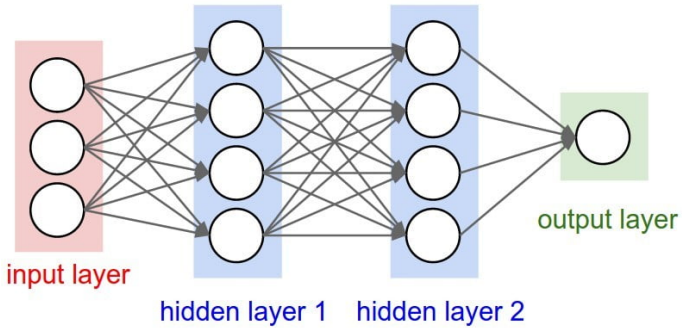
Irudia 4.1: Ezkerrean, **Perceptron**-aren eskema agertzen da eta eskuinean berriz, algoritmoaren prozesua hiperplano egokia aukeratzen duen arte.

Aurrerapen horiekin batera ordea, 60. eta 70. hamarkadetan alde batera utzi ziren sare neuronal artifizialen ikerketak, batik bat urte horietako ordenagailuen gaitasuna urria zelako eta ez zegoelako sareak entrenatzeko algoritmo egokirik. Hori dena ordea, 1986. urtetik aurrera aldatu zen, **backpropagation** edo atzerako hedapenaren algoritmoarekin. Algoritmo horrek, sare neuronalei *ikasteko* edo pisu

egokiak aukeratzeko gaitasuna ematen dio era automatikoan. Ordutik hona, hazkunde eta garapen oso handia izan du (eta izaten ari da) sare neuronalen arloak, gaitasun ia mugagabeak dituela erakusten ari baita.

4.2 Ideia nagusiak

Sare neuronal artifizialen ideia, gure burmuinean gertatzen diren prozesuak imitatzea edo emulatzea da, eskema eraikitzeko orduan batik bat. Era sinplean azaltzeko, Irudia 4.2 begira daiteke. Bertan, zirkuluak neuronak adierazteko erabiltzen dira eta geziak aldiz, beraien arteko loturentzako. Kasu sinple horretan, neurona bakoitzak bere ezkerrean dauden neuronenganako dependentzia osoa adierazten du, horien arabera aktibatuko dira edo ez.



Irudia 4.2: Sare neuronal artifizial baten oinarrizko eskema.

Demagun, Irudia 4.2-en oinarritzen jarraituz, honako kasua aztertu behar dela: azken 50 urteetan Donostian saltzen jarri diren pisuen datuak:

- Salmenta prezioa
- Metro karratu kopurua (m^2)
- Pisuaren urte kopurua
- Pisua 2 hilabetetan saldu den edo ez adierazten duen aldagai dikotomiko bat

Suposa dezagun, helburua lehenengo 3 aldagaiekin pisua 2 hilabetetan saldu den edo ez adierazteko gai den sarea sortzea nahi dela (1. kapituluan aztertutako sailkapen metodo bat bezala ikus daiteke).

Sare neuronalaren prozesuak honako pauso hauek ditu:

- $X \in \mathbb{R}^{1\times 3}$ sarrerako datuak (**input layer**-a: prezioa, m^2-ak eta urte kopurua) hurrengo geruzara (**hidden layer 1**-era) pasatu matrize biderketa baten bidez:

$$X \cdot M_1 = H_1' \text{ non } M_1 \in \mathbb{R}^{3\times 4} \text{ eta } H_1' \in \mathbb{R}^{1\times 4}$$

- Behin H_1' lortutakoan, **aktibazio funtzio**[1] bat aplikatzen zaio H_1'-ko zenbakiak 0 edo 1en bilakatzeko (neuronak aktibatzeko edo aktibatu gabe uzteko):

$$f(H_1') = H_1 \text{ non } H_1 \in \mathbb{R}^{1\times 4}$$

- H_1 **hidden layer 1**-etik hurrengo geruzara pasatu (**hidden layer 2**-ra) matrize biderketa baten bidez:

$$H_1 \cdot M_2 = H_2' \text{ non } M_2 \in \mathbb{R}^{4\times 4} \text{ eta } H_2' \in \mathbb{R}^{1\times 4}$$

- Behin H_2' lortutakoan, **aktibazio funtzio** bat aplikatzen zaio H_2'-ko zenbakiak 0 edo 1en bilakatzeko (neuronak aktibatzeko edo aktibatu gabe uzteko):

$$g(H_2') = H_2 \text{ non } H_2 \in \mathbb{R}^{1\times 4}$$

- H_2 **hidden layer 2**-tik amaierako geruzara pasatu (**output layer**-era) matrize biderketa baten bidez:

$$H_2 \cdot M_3 = y' \text{ non } M_3 \in \mathbb{R}^{4\times 1} \text{ eta } y' \in \mathbb{R}$$

- Azkenik y' lortutakoan, **aktibazio funtzio** bat aplikatzen zaio y' zenbakia 0 edo 1en bilakatzeko (pisua 2 hilabetetan saldu da edo ez):

$$h(y') = y \text{ non } y \in \{0, 1\}$$

4.3 Ikasketa prozesua

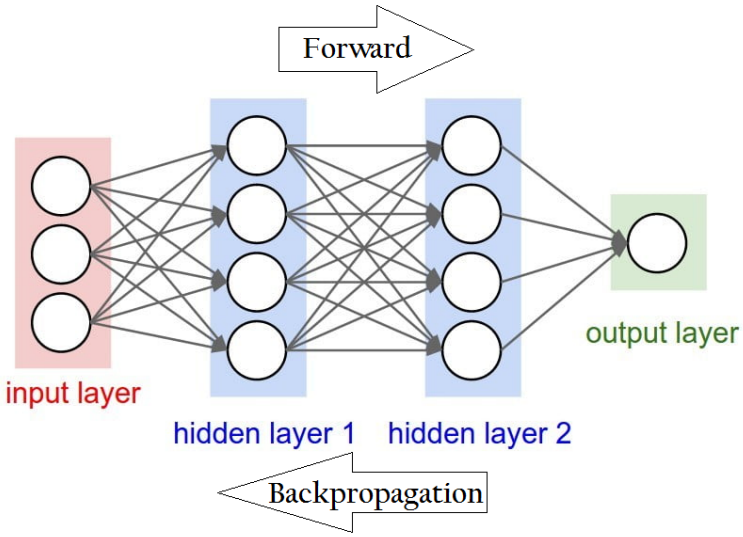
Azaldu berri den prozesuan M_1, M_2 eta M_3 matrizeak agertu dira, baina ez da adierazi nondik lortu diren matrize horien balioak. Sarearen helburu nagusia matrize horien balioak lortzea da hain zuzen ere, matrize horien balio egokiak lortuz gero irteerako emaitza ona edo txarra izango delako. Atal honetako sarreran agertu den bezala, M_i matrize horien balioak era egokian aukeratzeko erabiltzen den metodoa edo algoritmoa **backpropagation** edo atzerako hedapena da. Bertan, deribatuak erabiltzen dira errorea minimizatzeko.

Hemen ez dira azalduko metodoaren atzean dauden matematikak eta xehetasunak, ez delako koaderno honen helburua, nahiz eta interesaturik dagoen irakurleak sare neuronal artifizialen inguruko ia edozein liburutan aurki dezake, besteak beste [13, 21] liburuetan. Hortaz gain, Andrej Karpathy-k, Tesla enpresako AI saileko zuzendariak besteak beste, bideo oso interesgarri bat du interneten **backpropagation** metodoa azaltzen:

[1] Aktibazio funtzio erabilienak: sigmoide funtzioa, arku tangente funtzioa eta ReLU funtzioak dira. `https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0`

`https://www.youtube.com/watch?v=59Hbtz7XgjM`

Ondorioz, sare neuronalen ikasketa prozesua Irudia 4.3-eko prozesua behin eta berriro errepikatuz egiten da. Alde batetik, sarrerako datuak sartzen dira eta M_1, M_2, M_3 matrizeekin irteerako emaitza lortzen da. Lortutako emaitza ondo baldin badago, ez dira M_i matrizeen balioak aldatzen eta lortutako emaitza okerra baldin bada, M_i matrizeen balioak *zuzentzen* dira **backpropagation** erabiliz.



Irudia 4.3: Sare neuronal artifizial baten oinarrizko eskema.

4.4 Abantailak eta Desabantailak

Sare neuronal artifizialek garrantzia handia hartu dute azken urteetan, izan ere momentu oro aplikazio berriak azaltzen baitira eta oraindik ez zaiolako mugarik aurkitu. Supervised Learning-eko erregresio zein sailkapen ereduak sortzeko gai dira, baita Unsupervised Learning-en ikusitako datuen dimentsioa murrizteko ere (autoencoder deritzona). Bestalde, metodo gehientsuenentzat oso konplexuak diren lanak egiteko gai dira: irudien sailkapena, soinuen identifikazioa, itzulpengintza...

Hala ere, desabantaila nabari batzuk ere badituzte. Alde batetik, teoria gutxi garatu da hauen inguruan, lortutako aplikazio eta emaitza gehienak proba desberdinen bidez eman dira. Ez dago teoriarik geruza kopurua aukeratzeko ezta neurona kopurua ere ez, aurki daitekeen bakarra gomendioak dira, beste kasu batzuetan funtzionatu izan duelako. Bestalde, lortzen diren ereduak *kutxa beltzak* direla esan izan da, ereduaren nondik norakoak jakitea kasu askotan ezinezkoa delako. Beste era batean esanda, emaitza onak eman ditzakeen arren, gehienetan ezingo da jakin nola egiten dituen sailkapenak ezta estimazioak.

Kapitulua 5

Ensemble methods

Ensemble methods-en helburua, eredu anitz ezberdinak sortzea da eta eredu horien konbinazioaren bitartez hobeak diren emaitzak lortzeko. *Decision trees*-en kasuan adibidez, askotan erabiltzen dira metodo hauek emaitzak hobetzeko. *Decision trees*-en algoritmoa orokorrean erraza eta azkarra da eta propietate hori aprobetxatuz, zuhaitz desberdin asko sortzen dira. Horien arteko konbinazioa erabiliz eredu hobeak eta sendoagoak sortzeko[1].

Oro har, 2 familiatan bereizten dira metodo horiek:

- *Averaging methods* edo *Bagging methods*-en, estimatzaile desberdinak eratzen dira era independentean lehenik eta beraien arteko batez bestekoa kalkulatu ondoren. Batez bestekoa egiterakoan, lortutako estimatzaile berria hobea izango da orokorrean, alborapena txikitzen delako.

- *Boosting methods*-en aldiz, estimatzaileak era sekuentzial baten eraikitzen dira, pauso bakoitzean estimatzaile berriaren alborapena hobetzen saiatzen delarik.

5.1 Bagging methods

Kapitulu honetako sarreran esan bezala, metodo honetan, sarrerako datuekin hainbat eredu sortzen dira lehenik eta horien batez besteko eredua sortzen da ondoren, propietate hobeak izango dituenak. Ondoren, algoritmoaren ideia nagusiak azalduko dira:

- Lehenik eta behin n eta M balioak aukeratzen dira. M balioa, sortuko diren lagin kopurua izango da eta n, eredu horiek sortzeko erabiliko diren elementu kopurua.

- Hurrengo pausua laginak sortzea da. M lagin sortuko dira eta lagin bakoitzerako n elementu aukeratuko dira sarrerako datuetatik. n elementu horiek zoriz aukeratuko dira eta lagin bakoitzerako ezberdinak izango dira.

[1] *Netflix Competition*-en [7] adibidez , *collaborative filtering* algoritmo indartsu bat egiteko, *ensemble methods* erabili zuten.

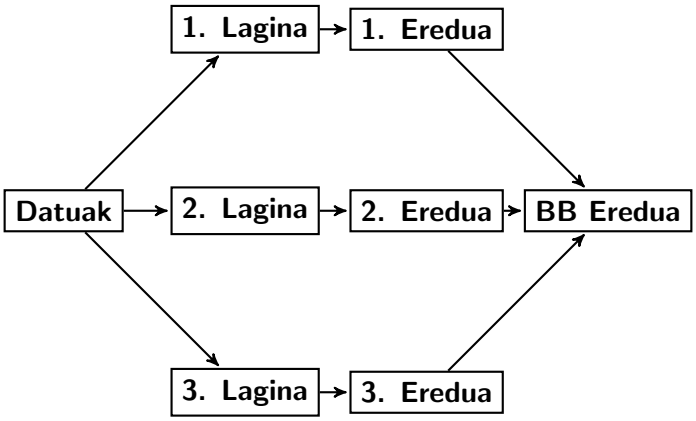
- Behin M laginak sortu direnean, lagin bakoitzerako eredu bat sortzen da erabili nahi den metodoarekin (erregresio edo sailkapenekoa).

- Azkenik, eredu finala aurreko ereduen batez bestekoa izango da. Sailkapeneko kasuan, batez bestekoa izan beharrean, puntu baten klasea, sortutako ereduetan emaitza positibo gehien dituena izango da.

Metodo hau erabiliz lortzen diren laginek ordea, sarritan korrelazioa izango dute beraien artean eta horrek amaierako ereduaren kalitatean eragin negatiboa izango du. Korrelazio hori kentzeko edo gutxitzeko, *Random Trees* metodoa erabiltzen da.

Random Trees

Ikusitako algoritmoarekiko duen aldaketa, aldagaien aukeraketan datza. Sarrerako datuak, k aldagai badituzte, lagina aukeratzeko orduan laginaren d aldagai bakarrik aukeratzen dira zoriz. Normalean, $d \approx \sqrt{k}$ hartzen da. Horrek, laginen arteko korrelazioa jaisten du eta ondorioz emaitza hobeak lortzen dira.



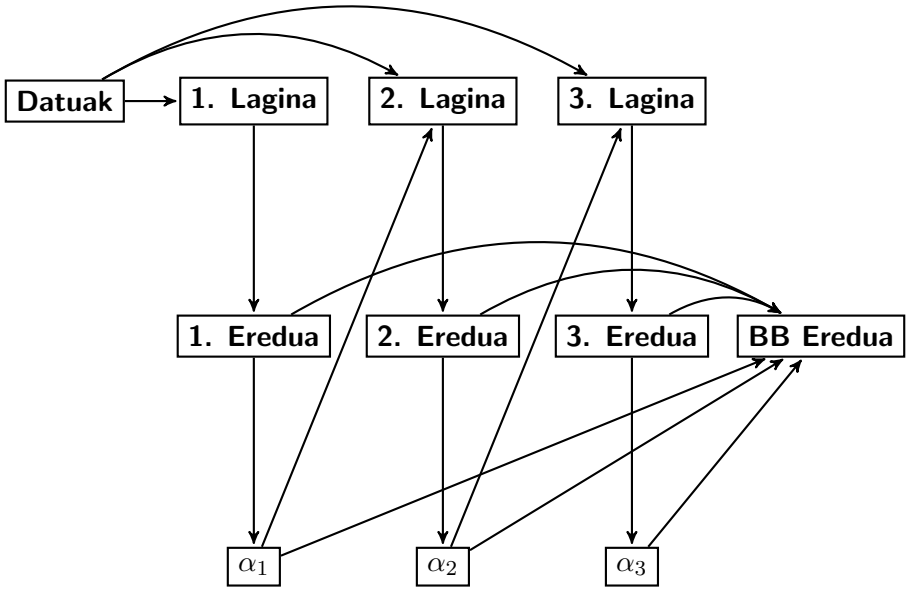
Irudia 5.1: Bagging metodoaren eskema: sarrerako datuetatik tamainu berdineko laginak hartzen dira, lagin horien ereduak sortu eta azkenik eredu horien batez besteko eredu bat sortzen da.

5.2 Boosting methods

Metodo honen funtsa, eredu sinpleetatik abiatuz, era sekuentzial batean eredu hori hobetzen joatea da. Sailkapen metodoentzako bakarrik balio du. Hobetze hori, ereduak gaizki sailkatutako datuei pisu handiago ematen egiten da. Hurrengo lerroetan, *AdaBoost* (Adaptive Boosting) algoritmoaren ideia nagusiak azaltzen dira[2]:

[2]Hurrengo bideoan, AdaBoost algoritmoaren funtzionamendua azaltzen da grafikoki: `https://www.coursera.org/learn/ml-classification/lecture/um0cX/example-of-adaboost-in-action`

- *Bagging*-arekin bezala, lehenik eta behin n eta M balioak aukeratzen dira. M balioa, sortuko diren lagin kopurua izango da eta n, eredu horiek sortzeko erabiliko diren elementu kopurua.
- Ondoren, n tamainako zorizko lagin bakun bat ateratzen da. Pauso hau, lehengo aldiz bakarrik egiten da, hurrengo pausuetan, lagina ateratzeko orduan lagin horretako elementu batzuk probabilitate handiagoa izango dutelako emango zaien pisuaren arabera.
- Lagin horri, sailkapen metodoarekin eredu bat esleituko zaio.
- Eredu horretan, gaizki sailkatutako elementuak zenbatuko dira eta elementu horiei pisu handiagoa emango zaie, probabilitate handiagoa izan dezaten hurrengo laginean agertzeko.
- Beste n tamainako lagin bat ateratzen da, lortu den banaketa probabilistiko berriarekin eta aurreko prozesu berdina egiten da M aldiz.
- Azkenik, prozesua M aldiz errepikatu ostean, lortutako M eredu horien arteko batez besteko haztatuaren bidez (algoritmoan lortutako pisuak erabiliz), nahi den eredu finala lortuko da.



Irudia 5.2: AdaBoost metodoaren eskema. α_i balioak, ereduetatik lortzen diren pisuari dagokie, hurrengo lagina eratzeko eta eredu finalerako balioko dutenak.

Algoritmo honek, edozein sailkapen metodorako balio du eta esan bezala, horren emaitzak hobetzen eta errorea txikitzen laguntzen du. Hala ere, oro har metodo sinpleei aplikatu izan zaie, *Decision trees* modukoak, kostu konputazional baxua dutelako.

Atala II

Datuekin lanean

Sarrera

Atal honetan, serie denboralen clustering-aren inguruan egindako lana eta lortutako emaitzak azalduko dira, beka honen helburu nagusia izan delarik. Horretarako, EAEko hotel eta pentsioen eguneko prezioak jaso dira web plataforma batetik. Datuak web scraping metodo ezberdinak erabiliz lortu dira eta web orrialdearen egitura ezagutzen joan den ahala, datuen kalitatea ere hobea izan da.

Eguneko prezioak finkatzeko, egun bakoitzeko 120 kontsulta egin dira plataforman. Web scraping-eko programak, EAEko hotel eta pentsio bakoitzeko hurrengo 120 egunetako prezioak jasotzen ditu. Adibide bezala, 2018ko urtarrilaren 1ean, programak urtarrilaren 1etik maiatzaren 1erako egun guztietako prezioak jasotzeko eskaera egin zuen hotel eta pentsio bakoitzeko. Hori dela eta, kasurik onenean hotel eta pentsio bakoitzak urteko egun bakoitzerako 120 prezio izango ditu.

Egoera hori ordea ia ezinezkoa da, bidean oztopo asko baitaude. Alde batetik, ostatu guztiek ez dute web bidez erreserba egiteko aukera ematen. Lan honetarako adibidez, lortutako ostatuen kobertura %75 ingurukoa izan da. Bestalde, arazo desberdinak gerta daitezke datuak jasotzen dituen programan (ezusteko erroreren bat, web orrialdearen egituran aldaketak, ostatua beteta/itxita egotea...) edota plataformako orrialdean bertan.

Behin 120 eskaera horiek egin ondoren, egun bakoitzeko prezio adierazgarria aukeratzeko prozesua dator. Azterketa hori, EHUko ikerketa lan moduan utzi da eta bitartean, lan honetarako prezio guztien mediana hartzea erabaki da, zentro-neurri bat delako eta balio arraroak kanpoan uzten dituelako.

Jasotako datu horiek, Eustat-eko Establezimendu Turistiko Hartzaileen Inkestarekin fusionatu dira, hotel/pentsioen informazio gehigarria batzeko datuetara.

Aurrepauso horien ondoren, datuekin lan egiteko `R` [28] programazio-lengoaia erabiltzea erabaki da. Arrazoi nagusiak, doako programazio-lengoaia dela, *Machine Learning* munduan azken urteetan indar handia hartu duela (Python programazio-lengoaiarekin batera) eta estatistikara bideratuta dagoela.

Lan honen prozesua 3 zati garrantzitsutan banatu da:

- Outlier-ak serie denboraletan
- Inputazioa serie denboraletan
- Serie denboralen Clustering-a

Emaitzak irudikatzeko erabilitako `R`-ko pakete garrantzitsuenak berriz, `shiny` [6], `plotly` [32] eta `leaflet` [8] izan dira.

Kapitulua 6

Outlier-ak serie denboraletan

Analisi desberdinak egiterako orduan, logikoa den bezala emaitzek datuekiko dependentzia oso handia izaten dute. Hori dela eta, jasotako datuak gaizki jaso baldin badira edo erroreak baldin badituzte, analisietatik ateratako emaitzak eta ondorioak fidagarritasun urrikoak izan daitezke. Gure kasua ez da salbuespen bat eta horregatik gure serie denboraletako balio arraroak eta outlier-ak detektatzeko metodo desberdinak testatu ditugu eta gure nahietara egokitzen den algoritmo bat sortu.

Hasieran, `tsoutliers` [9] eta `outliers` [19] paketeak erabili eta probatu ziren, baina berehala ikusi zen ez zirela erabilgarriak gure kasurako (ez behintzat era zuzenean). Pakete horiek eskaintzen duten funtzioen bidez, serie denboralak bananbanan aztertzen dira eta ondorioz, egun berezietako prezio altuak outlier bezala detektatzen dituzte. `tsoutliers`-en, egutegi efektuak zehaztu daitezke (astebukaerak, jai-egunak, aste santua...), baina hala ere horrekin ez da beti nahikoa izaten, ekitaldi bereziak egoten baitira.

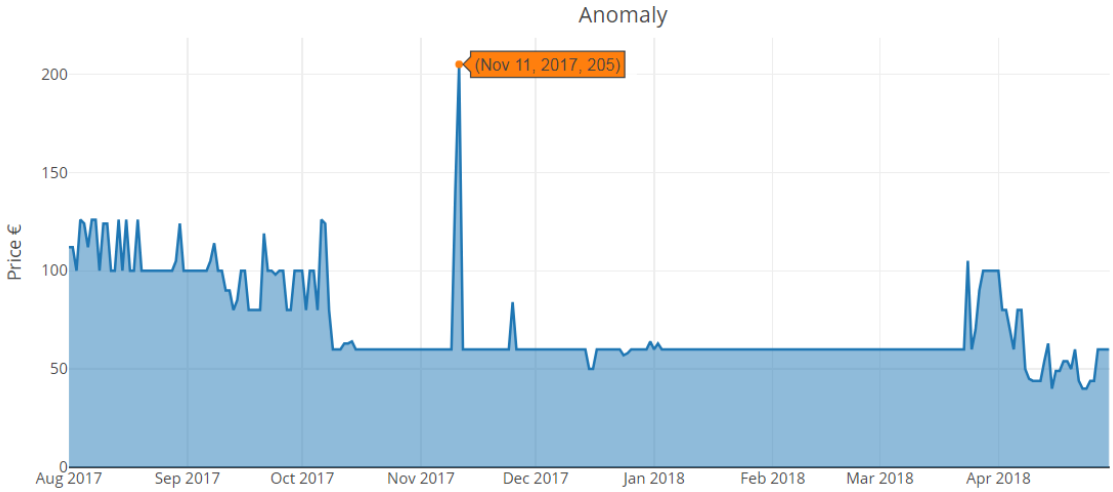
Irudia 6.1 aztertzen baldin bada, azaroak 11an balio arraro bat dagoela ikus daiteke. Egutegiari dagokionez larunbata izan zen eta EAEn ez zen zubi berezirik izan. Hala ere, Donostiako pentsio eta hotel gehienetan patroi berdina ematen denez Donostian izan zen ekitaldi berezi bat bilatu zen, kasu honetan *Behobia-San Sebastián* erdi-maratoia.

Ondorio bezala, gure serie denboralak beraien artean konparatzeko beharra zegoela ikusi zen. Anomalia edo outlier izatearen diferentzia, balio arraroa beste hotel/pentsioetan errepikatzen den edo ez delako.

6.1 Outlier-ak detektatzeko algoritmoa eraikitzen

Atal honen sarreran esan bezala, outlier-ak detektatzeko ostatu ezberdinen arteko konparaketa egitea beharrezkoa dela kontsideratu da. Ideia nagusia serie denboraletako balio arraroak detektatzea da aurrenekoz eta outlier-a edo anomalia bat den erabakitzea ondoren serie denboralen arteko konparaketaren bidez.

Azterketa hori aurrera eramateko, prozesua 4 zatitan banatu da nagusiki:



Irudia 6.1: Donostiako pentsio baten prezioen serie denborala. Azaroak 11an anomalia bat dagoela ikus daiteke astebukaera horretan *Behobia-San Sebastián* erdi-maratoia izan zelako.

Serie denboralak hileka bildu

Etorkizunean, iristen doazen datuak era automatikoan depuratzea nahi denez, hileka aztertzea erabaki da serie denboraletako outlier-ak. Gainera, outlier-ak aztertzeko orokorrean ez da erabilgarria hilabete ezberdinetako datuak konparatzea, urteko sasoi bakoitzak joera oso desberdinak dituelako.

Outlier minimoak ezabatu

Behin datuak hileka multzokatuta daudenean, hurrengo pausua outlier minimoak ezabatzea da. Outlier minimoak ezabatzeko erabili den ideia oso sinplea da: hileka, hotel/pentsio bakoitzeko prezioei test bat pasatu eta test horren arabera balioa ezabatzeko erabakia hartu. Detekziorako erabili den testa Grubbs-en [14] metodoan oinarritzen da, `outliers` paketean aurkitzen dena. Metodo honen bidez outlier-ak multzoka aztertzen dira eta denbora faktorea ezabatzen da, hau da, berdin du prezio horiek zein egunetan eman diren.

Outlier minimoen kasuan, maximoekin ez bezala ez da patroirik ematen, hau da, ez dira prezio baxu guztiak egun gutxi batzuetan pilatzen. Izan ere, orokorrean prezio baxuak altuak baino arruntagoak dira, beste era batean esanda, prezioek orokorrean egonkortasun bat izaten dute hilabetean zehar eta egonkortasun hori gehienetan prezio maximoekin hausten da.

Ez da aipatu gabe utzi behar Grubbs-en testak estatistiko batzuk itzultzen dituela eta horiekin loturik p-balio bat. Lanetako bat p-balio hori zehaztea da, outliera zer den erabakitzeko. Horretarako, begiz aztertu dira outlier minimoak lehen 5 hilabeteetarako eta hobekien sailkatzen dituen p-balioarekin geratu gara.

Azkenik, algoritmoak detektaturiko balioak ezabatu dira eta ezabatu ondoren, ostatu bakoitzari bere hileko prezio minimoa esleitu zaio egun horietan.

Prezioak eskalaz aldatzen

Minimoen azterketarekin amaitu ondoren, maximoena dator eta atal honen sarreran esan bezala, lan hori egiteko ostatuen arteko konparaketa egitea komeni da. Konparaketarako, erabiltzen den eskala oso garrantzitsua da, ezin baita luxuzko hotel bat barrualdeko izar bateko hotel batekin konparatu, ez behintzat prezio absolututan. Horregatik, hurrengo eskala aldaketa egin da prezioetan:

$$z_i = \frac{x_i}{\min(x)} \cdot 100 \tag{6.1}$$

non

- x: prezioen serie denborala (hileka)
- x_i: i. eguneko prezioa
- z_i: i. eguneko prezio berreskalatua

Eskala aldaketa horren bitartez, hotel/pentsio batek hilabetean zehar ehunekotan izandako igoera neurtzea nahi da. Adibide bezala, hurrengo bektorea berreskalatuko da:

$$(70, 105, 70, 140, 210) \longrightarrow (100, 150, 100, 200, 300)$$

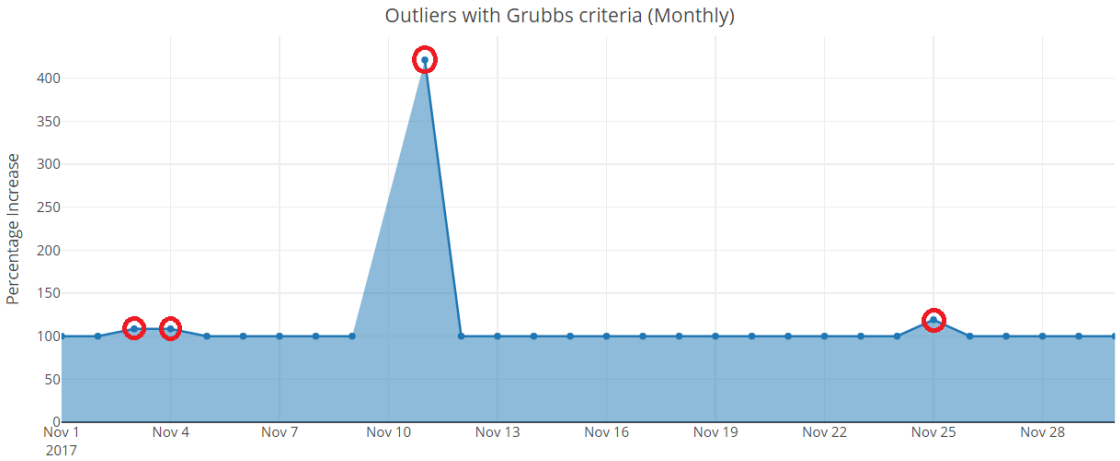
100 balioa dutenak, hilabeteko prezio minimoa adierazten dute eta beste balioek igoera ehunekotan, hau da, 200 balioak prezioa bikoiztu dela adierazten du (minimoarekin alderatuz) eta 150 balioak prezio minimoa 1.5-ekin biderkatu dela ($70 \cdot 1.5 = 105$). Eskala aldaketa honen bidez prezio aldaketak konparatzea lortu da.

Outlier maximoak ezabatu

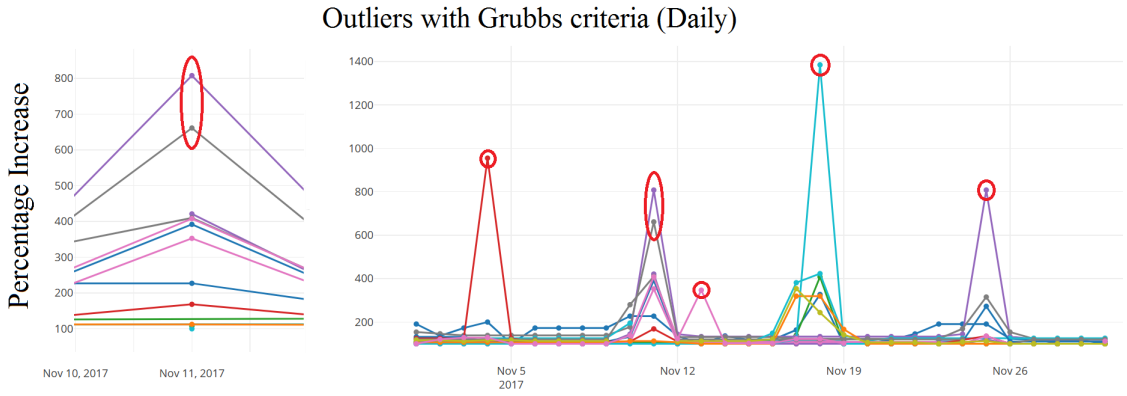
Datuen lehen aurreprozesamendua eginik, outlier maximoak detektatzeko 2 azterketa egiten dira: **hileko azterketa** eta **eguneko azterketa**. **Hileko azterketa** minimoen kasuan bezala egiten da, hotel bakoitzaren outlier-ak detektatzen dira hileka Grubbs metodoaren bitartez. Adibide bezala, Irudia 6.2-n kasu erreal bat agertzen da. Bertan, Grubbs-testa pasa ondoren outlier bezala sailkatzen dituen 4 puntu ageri dira. Grafikoa ikusita, azaroak 3, 4 eta 25 ez dirudite inondik inola ere ez outlier-ak, baina kasu berezi honetan, gainontzeko prezio guztiak konstanteak direnez, horrela sailkatzen ditu.

Lehen pausu horren ondoren, bigarren pausua **eguneko azterketa** da. Bigarren pausu honetan ere Grubbs-en testa pasatzen da, baina egun bakoitzeko pasatzen da eta beste ostatuekin konparatuz. Azaroko adibidearekin jarraitzen baldin bada, kasu horretan 30 aldiz pasatzen da Grubbs-en testa, hileko egun bakoitzeko. Egun bakoitzean, hotel desberdinen prezioak kontsideratzen dira ((6.1) formularen bidez berreskalatutako prezioak) eta multzo horren gainean aplikatzen da testa. Azaroko emaitzekin lortutako outlier-ak Irudia 6.3-n agertzen dira.

Kontutan izan behar da, eguneko azterketa egiterakoan hotel eta pentsioak lurralde historikoaren arabera multzokatu direla. Era horretan, Gasteizen azaroaren



Irudia 6.2: Grafiko honetan, azaroko hilabetean hotel bati Grubbs-en testa aplikatu ondoren lortutako 4 outlier-ak agertzen dira gorriz borobildurik.



Irudia 6.3: Azaroko egun bakoitzeko outlier-ak agertzen dira gorriz borobildurik. Ezkerreko grafikoan azaroak 11ko eguna agertzen da, egun horretan testa pasa ondoren outlier kontsideratu diren 2 puntuak borobilduta daudelarik.

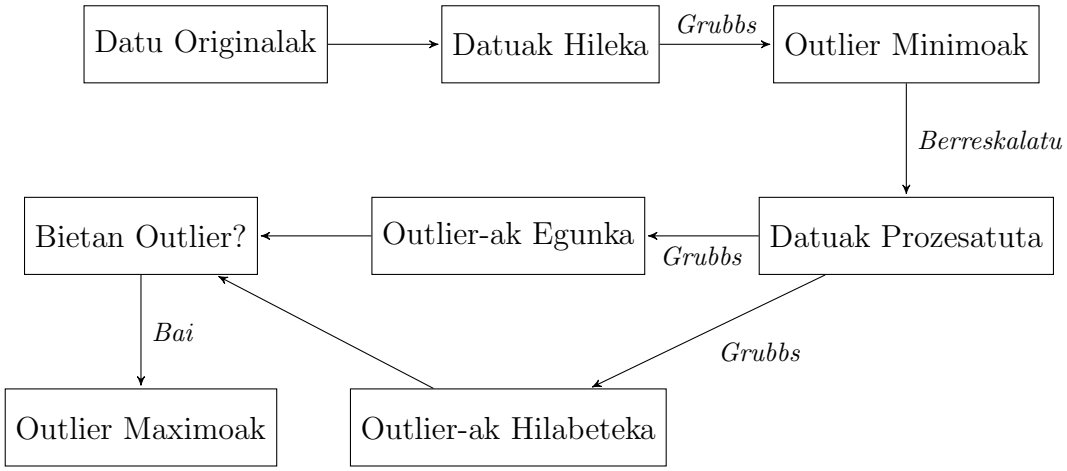
11an eman zitekeen balio arraro bat Donostiako balio arraroekin konparatzea ekiditen da, hiriburu eta lurralde historiko bakoitzak joera ezberdina duelako.

Bi azterketa edo pausu horien ondoren, hurrengo erabakia hartzen da:

- Balioren bat 2 azterketetan outlier-a baldin bada $\Longrightarrow$ **OUTLIER**
- Kontrako kasuan $\Longrightarrow$ **EZ DA OUTLIER**

Adibideko kasuan, ez da outlier kontsideratuko balio bat ere ez, izan ere ikus daitekeen bezala azaroak 11n igoera handiak eman ziren ostatu askotan eta ondorioz ez dira 2 baldintzak bete. Minimoen kasuan bezala, Grubbs-en testerako p-balioa finkatzeko begiz outlier-ak identifikatu dira 5 hilabetetarako eta hobekien sailkatzen dituen balioa aukeratu da. Ezabatutako outlier-ei, hilabeteko prezioaren 3. koartila esleitu zaie.

Laburbiltzeko, hurrengo diagraman algoritmoan emandako pausuak agertzen dira:



Kapitulua 7

Inputazioa serie denboraletan

Serie denboraletan, arrazoi desberdinak direla medio, sarritan balio galduak agertzen dira. Balio galdu horiek, batzuetan interesgarriak izango dira bere baitan, informazio gehigarri bat eskaintzen dutelako, baina besteetan aldiz arazo bat izango da eta balio horiek berreskuratzea nahiko da.

Gure kasuan, balio galduek esanahi desberdina izan dezakete: hotela itxita egotea, hotela beteta egotea, datua jasotzerako orduan arazoak edo erroreak izatea... Itxitako kasuak erraz identifika ditzakegu, inkesta bidez heltzen zaigulako informazio hori. Betetako kasuetan edota arazo teknikoak izandakoetan, konponbide posible bat datuak inputatzea izaten da.

Serie denboralen inputazioari dagokionez, `R`-k [28] `imputeTS` [26] paketea eskaintzen du. Bertan, hainbat metodo agertzen dira beharren arabera bata edo bestea erabiltzeko. Gure kasu partikularrean, datuek aldizkakotasun nabaria dute, hau da, orokorrean prezioak beti igotzen dira ostiral eta larunbatetan. Kasu horietarako, aldizkakotasun nabaria duten kasuetarako, `imputeTS`-ko dokumentazioan hurrengo funtzioak erabiltzea gomendatzen dira, emaitza hoberenak ematen dituztelako gehientsuenetan: `na.seasplit`, `na.seadec` eta `na.kalman`.

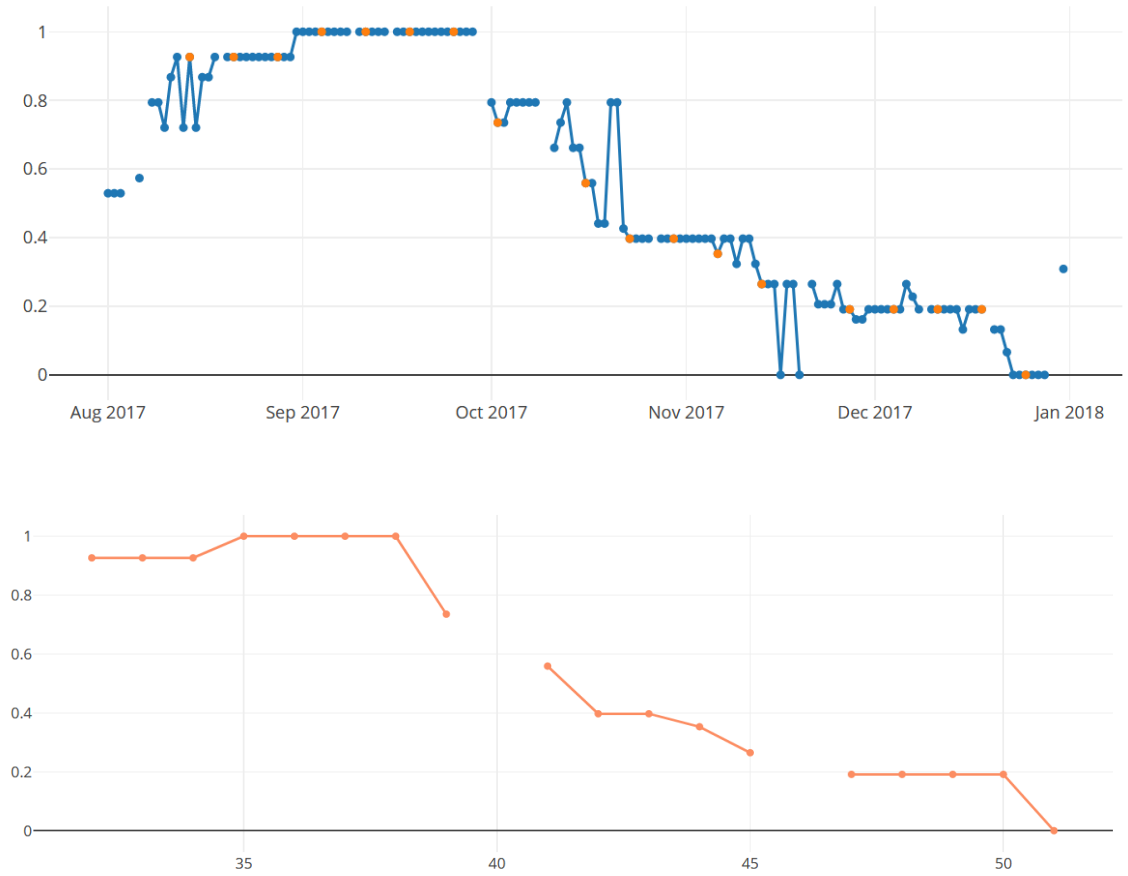
7.1 na.seasplit

Funtzio honetan, hainbat pauso daude inputazio prozesuan. Funtzioari aurresan egin behar zaio sartutako serie denboralek duten aldizkakotasuna. Gure kasuan astebeteko aldizkakotasuna dugu, beraz, funtzioaren input-a honakoa izan beharko da:

```
ts(x, frequency = 7)
```

non `x` gure sarrerako datuak diren, hau da, hotel bateko prezioak. Hasteko, funtzioak serie denboral originaletik, 7 serie denboral aterako ditu, asteko egun bakoitzeko bat (ikusi Irudia 7.1 adibide modura)

Behin gure serie denborala 7 serie denboraletan banatutakoan, horietako bakoitzari inputazio metodo bat aplikatzen zaio (`imputeTS` paketearen barruan hainbat



Irudia 7.1: Lehenengo irudia, balio galduak dituen serie denboral baten adibide bat da. Puntu laranjak, astelehenak dira. Beheko irudian asteleheneko datuak bakarrik ageri dira, urteko astearen arabera.

aukera daude: interpolazio bidez, ARIMA bidez, Basic Structural Model bidez, media bidez...).

7.2 na.seadec

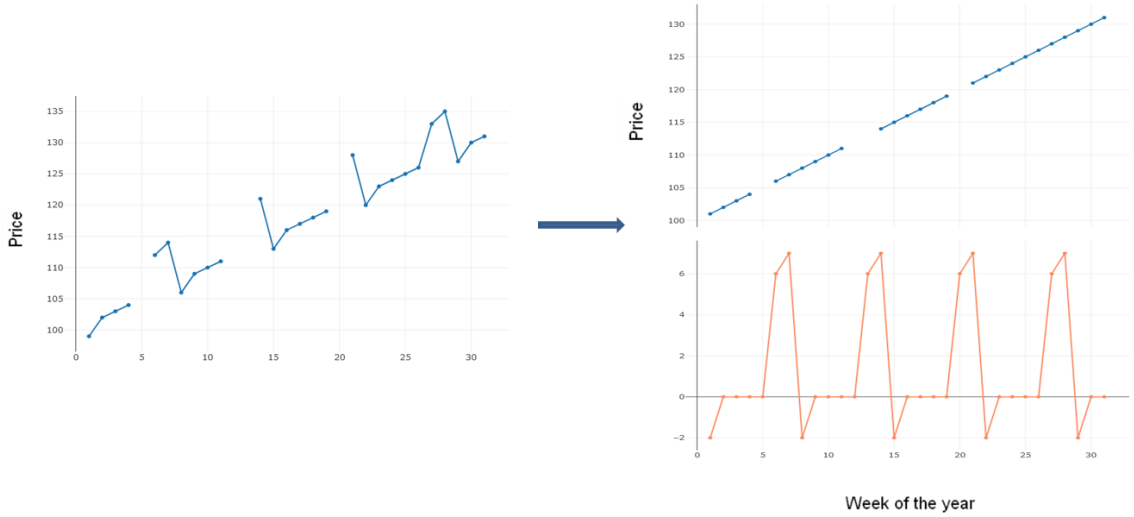
Metodo honetan ere, hainbat pausu diferente daude eta metodoaren abantaila aprobetxatzeko, aldizkakotasuna adierazi behar zaio. Gure kasuan:

```
ts(x, frequency = 7)
```

Hasteko, seriea era sinple baten inputatzen da interpolazio lineala erabiliz `na.interpolation` funtzioarekin. Hutsuneak bete ondoren, seriearen aldizkakotasuna kentzen zaio serie denboralari `stl`[1] funtzioaren bidez. Behin aldizkakotasuna ateratakoan, geratzen zaiguna inputatzen da nahi den metodo bat erabiliz (`imputeTS` paketearen barruan hainbat aukera daude: interpolazio bidez, ARIMA bidez, Basic

[1] `stl` funtzioa `R`-n barneratuta dagoen funtzio bat da. Serie denboralen deskonposaketa egiten du: *trend*, *seasonal part* and *remainder*.

Structural Model bidez, media bidez...), eta azkenik inputatu ondoren, aldizkakotasuna gehitzen zaio serieari.



Irudia 7.2: Ezkerreko irudian, inputatu nahi den serie denborala agertzen da. Eskuinean berriz, serie denboral horri `na.interpolation` funtzioa aplikatuta eta aldizkakotasuna kenduta ageri da.

7.3 na.kalman

Funtzio honek, modelizazio bidez inputatzen ditu balio galduak. Serie denboralaren eredu desberdinak sar diezazkiogu beharren arabera, baina funtzio honen barnean 2 modelizazio daude barneraturik: `auto.arima`[2] eta `StructTS`[3]. Lehenengoa `forecast` [16] paketearen barruan dago definiturik eta bigarrena `R` lengoaiak du bere barnean.

Demagun, gauzak sinplifikatzeko hurrengo eredua lortu dugula:

$$y_t = a \cdot y_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim N(0, \sigma)$$

hau da, gure serie denboraleko puntu bakoitzak bere aurreko puntuaren dependentzia bakarrik duela. Hori horrela izanik, `na.kalman` funtzioak modelo hori marrazten du **Kalman-en iragazkia**[4] erabiliz. Funtsean, Kalman-en iragazkiak y_t balioak aukeratzeko orduan, ε_t zarata *egokia* aukeratzeko gai da.

[2] ARIMA ereduei buruz gehiago jakitea nahi izanez gero, [31] liburuko 3. atalean agertzen dira metodoaren nondik norakoak.

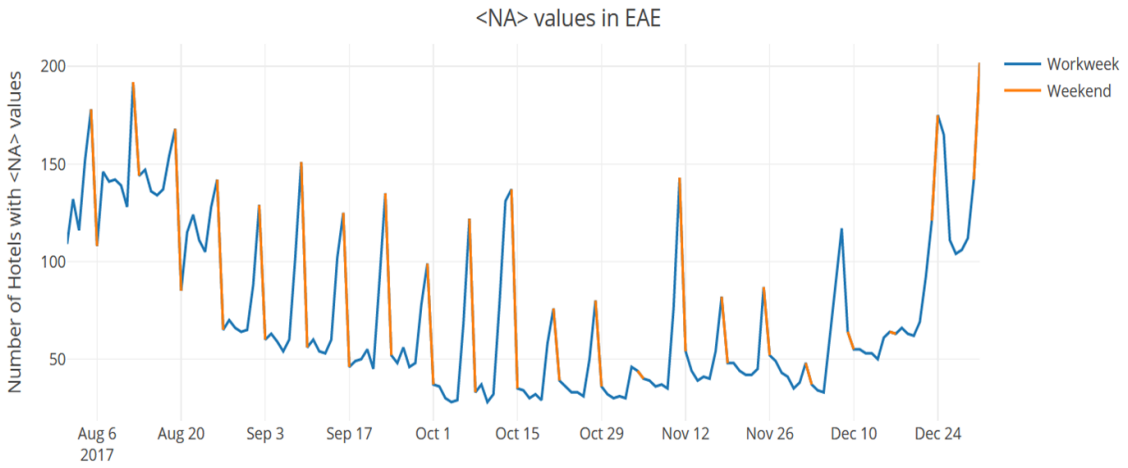
[3] *Basic Structural Model* (BSM) eta *State-Space* ereduen inguruko informazio gehiago [31] libruko 6. atalean aurki daiteke.

[4] [31] liburuaren 6.2 atalean aurki daitezke **Kalman-en iragazkiaren** nondik norakoak.

7.4 Inputatzeko Funtzioaren Aukeraketa

Behin `imputeTS`-ko dokumentuak gomendatzen dituen funtzioak aztertu eta ulertu ondoren, horietako bat aukeratu behar da gure beharrei hobekien egokitzen dena. Horretarako, gure serie denboral *osoak* hartu ditugu, hau da, 2017ko abuztutik 2017ko abendura bitartean 5 balio edo gutxiago falta dituzten serie denboralak (150 inguru) eta horietan testatu ditugu metodoak.

Testatzeko, serie denboral horiei 15-20 puntu inguru kendu dizkiegu zoriz, astebukaerei pisu handiagoa emanez (astebukaeretan balio galdu gehiago daudelako, ikusi Irudia 7.3) eta ondoren, metodo desberdinen bitartez inputatu ditugu. Sortutako errorea neurtzeko batez besteko errore koadratikoa erabili da.



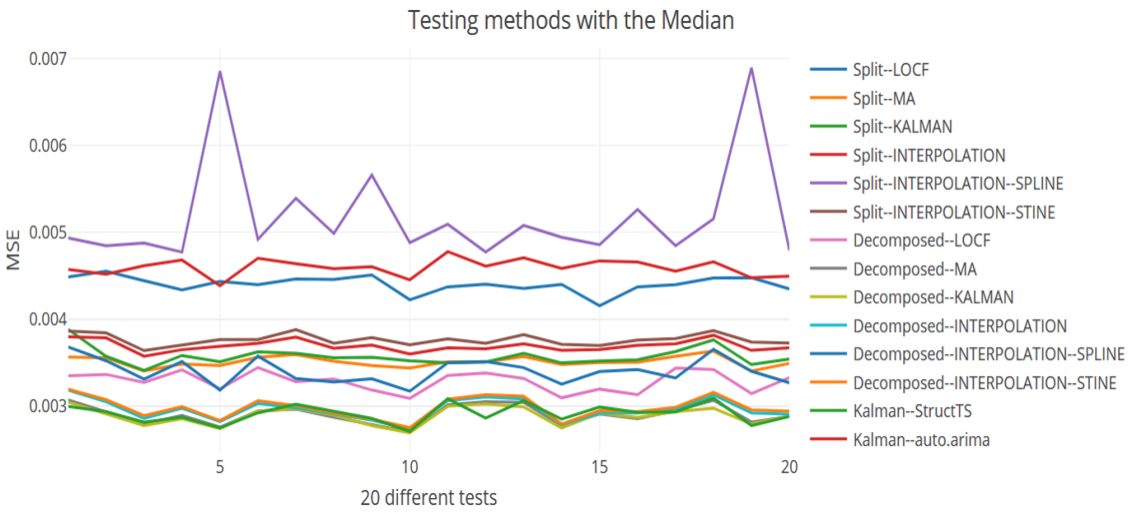
Irudia 7.3: Irudi honetan ikus daitekeenez, 2017ko abuztutik 2017ko abendura bitarteko balio galduak astebukaeretan pilatzen dira.

Test hori 20 aldiz errepikatu ondoren, Irudia 7.4 lortu da. Berdina egin da web-etik jasotako prezio minimoekin[5] (aldizkakotasun nabariagoa erakusten dutela jakinik) eta emaitzak Irudia 7.6-n ikus daitezke.

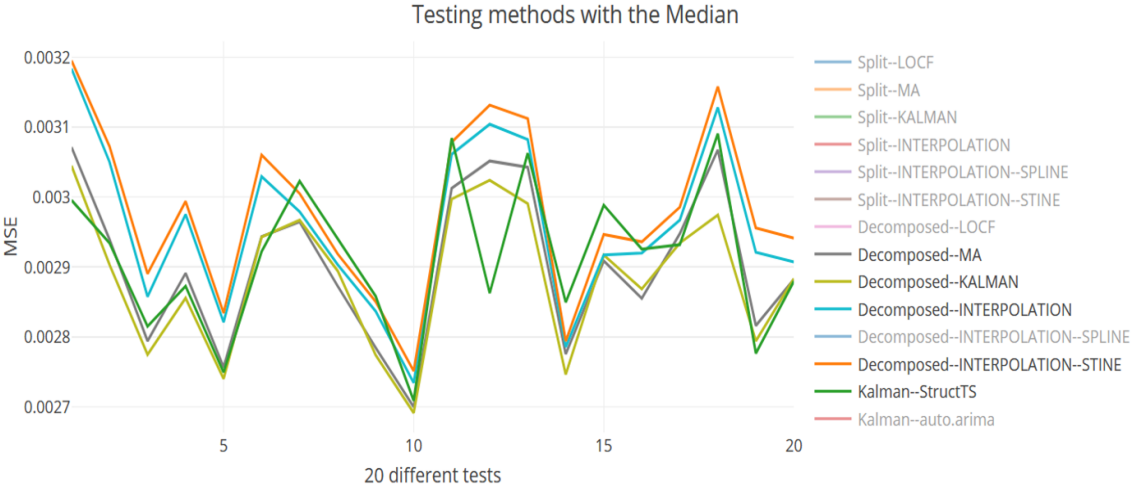
Emaitza horiekin eskuan, `na.seadec` funtzioaren barruan, `kalman` algoritmoa erabiltzea erabaki dugu, hau da:

```
na.seadec(ts(x, frequency = 7), algorithm = "kalman")
```

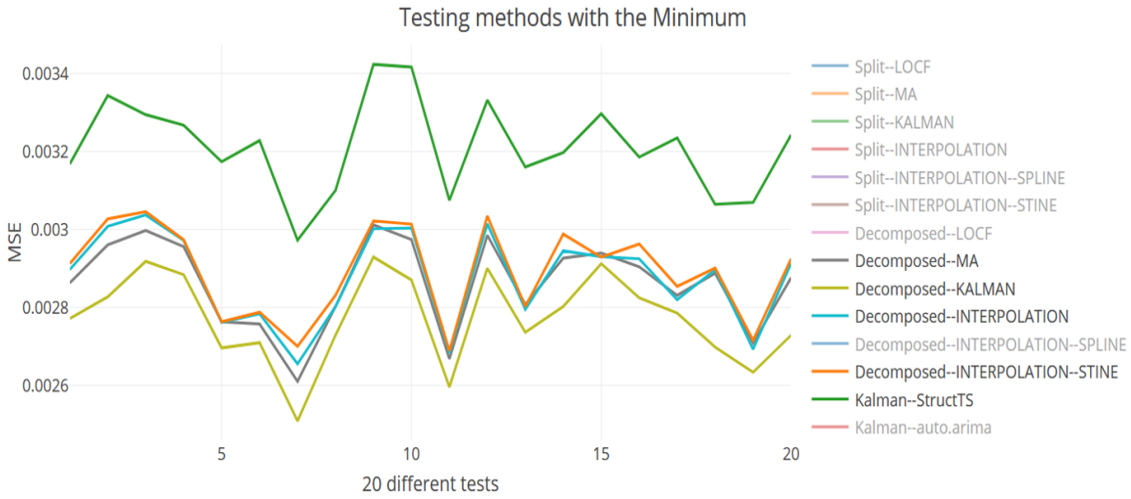
[5]Gogoratu hotelen eguneko prezioa zehazterako orduan, 120 egun ezberdinetan jasotako prezioak kontsideratu direla eta horien medianarekin egin direla azterketak. Inputazio metodoa aukeratzeko garaian, 120 prezio horien minimoak ere kontsideratu dira, aldizkakotasun handiagoa agertzen dutelako.



Irudia 7.4: Metodo desberdinekin sortutako inputazio errorea 20 test desberdinen ondoren. `Split=na.seasplit`, `Decomposed=na.seadec` eta `Kalman=na.kalman`. Beheko irudian errore gutxien sortzen duten metodoak agertzen dira.



Irudia 7.5: 20 test desberdinen egin ondoren errore txikiena eman dutenak.



Irudia 7.6: 20 test desberdinetan izandako errorea prezio minimoak erabiliz.

Kapitulua 8

Serie Denboralen Clustering-a

Beka honetako helburu nagusienetako bat, hau zen hain zuzen ere, prezioen serie denboralen clustering-a egitea. Horretarako, `R`-ko [28] hurrengo paketeak erabili dira: `TSdist` [25] eta `TSclust` [24]. Bi pakete horietan, serie denboralen arteko distantzia desberdinak ageri dira, kasuaren arabera erabiltzailearen beharrei hobekien egokitzen dena aukeratzeko. Gure analisirako, garrantzia eman zaio prezioen igoerak eta jaitsierak data berdinetan emateari, ez zelako nahi abuztuan prezio altuak zituen ostatu bat urrian prezio altuak zituen beste batekin konparatzerik, ondorioz DTW [12] (Dynamic Time Warping) bezalako distantziak ezabatu dira analisirako.

Batik bat probatu diren distantziak L_p distantziak (Manhattan eta Euklidearra) eta serie denboralen ezaugarrietatik eratorritako distantziak (korrelazio/autokorrelazio bidezko distantziak, Fourier-en koefizienteen bidezko distantziak, ARMA/ARIMA ereduetan oinarritutako distantziak...) dira. Gehientsuenekin clustering antzekoetara iritsi gara eta ondorioz, ulerterrazena den distantzia hartzea erabaki da: **distantzia euklidearra**.

8.1 Datuak preparatzen

`TSdist` eta `TSclust` paketeetan, serie denboralak era numerikoan sartu behar dira (matrize bezala, lista bezala, `data.frame` bezala, `ts` objektu bezala...), ondorioz gure hasierako `data.frame`-a eraldatu dugu lan hori egin ahal izateko. Eraldaketa hori egiteko erabili den funtzioa `reshape2` [34] paketeko `dcast` funtzioa izan da. Funtzio horren bidez, hasierako `data.frame`-etik 3 aldagai hartu dira: `IZENA`, `MEDIAN` eta `FECHA`. Behin 3 aldagaiak aukeratuta, `data.frame` berria sortu da non zutabeak `FECHA` eta hotel/pentsioen izenak diren.

Hori lortzeko erabili den kodea hurrengoa da:

```
dcast(df[,c("FECHA","IZENA","MEDIAN")],FECHA~IZENA,value.var="MEDIAN")
```

Behin taula lortutakoan, hurrengo pausua balore galduei konponbide bat ematea izan da. Irudia 8.1-n ikus daitekeen bezala, taula sortzean `NA` balio galduak lortu dira arrazoi ezberdinengatik (egun horretako prezioa ezin izan delako lortu, itxita egon delako...) eta hori arazo bat da `TSclust` eta `TSdist`-eko funtzioek serie denboral

Date	Hotel 1	Hotel 2	Hotel 3	Hotel 4	Hotel 5	Hotel 6	Hotel 7	Hotel 8	Hotel 9
2017-08-01	120	85	89	85	NA	110	NA	NA	99
2017-08-02	120	85	89	NA	NA	110	NA	NA	99
2017-08-03	120	85	89	85	NA	110	NA	NA	99
2017-08-04	120	107	89	85	NA	110	NA	NA	99
2017-08-05	120	112	89	85	NA	115	NA	NA	NA
2017-08-06	120	87	89	85	NA	121	NA	NA	99
2017-08-07	150	89	89	85	NA	121	NA	65	99
2017-08-08	120	89	89	85	NA	110	NA	65	99
2017-08-09	120	85	89	NA	NA	121	NA	65	99
2017-08-10	120	85	89	85	NA	121	NA	NA	99

Irudia 8.1: `dcast` funtzioa erabili ondoren lortutako `data.frame`-aren adibidea.

osoak behar dituztelako, hau da, balio galdurik gabekoak.

Balio galduentzako hurrengo neurriak hartu dira:

- 150 balio baino gutxiago duten (5 hilabeteko datuak baino gutxiago) hotel/-pentsioak ezabatu egin dira analisitik.
- Itxitako egunei, irekitako azken egunaren prezio berdina esleitu zaie `na.locf` funtzioaren bidez (`imputeTS` paketea erabiliz).
- Gainontzeko balio galduak, 7.4 atalean ikusitako `na.seadec` funtzioaren bidez inputatu dira, `kalman` algoritmoaren bidez.

Balio galduak inputatu ondoren, bi clustering desberdin sortu dira `TSclust` eta `TSdist` paketeen bitartez (prezio absolutuekin eta prezio normalizatuekin) eta beste clustering mota bat prezioen aldakortasuna aztertzeko.

8.2 Clustering prezio absolutuekin

Lehenengo clustering honetan, prezio absolutuen bidez taldekatu dira ostatuak. Serieak taldekatzeko erabili den distantzia, distantzia euklidearra izan da, besteak beste, erraza delako azaltzeko eta desio ziren emaitzak lortu direlako. Serieak taldekatzeko erabili den metodoa `pam` (*Partitioning Around Medoids*[1]) da, `cluster` [23] paketekoa, 2.1 ataleko K-means metodoan oinarritzen dena. K-means metodoarekin duen diferentzia nabarmena, cluster-eko zentroidea cluster-eko elementu bat izan behar dela (medoidea) da.

Hortaz aparte, cluster kopurua aukeratzeko *silhouette* metodoa erabili da (`pam` algoritmoaren bidez lortzen da), 2.1 ataleko ideian oinarrituta dagoena. Silhouette me-

[1] [18] Liburuko 2. atalean, algoritmo honen gorabeherak azaltzen dira xehetasun gehiagorekin.

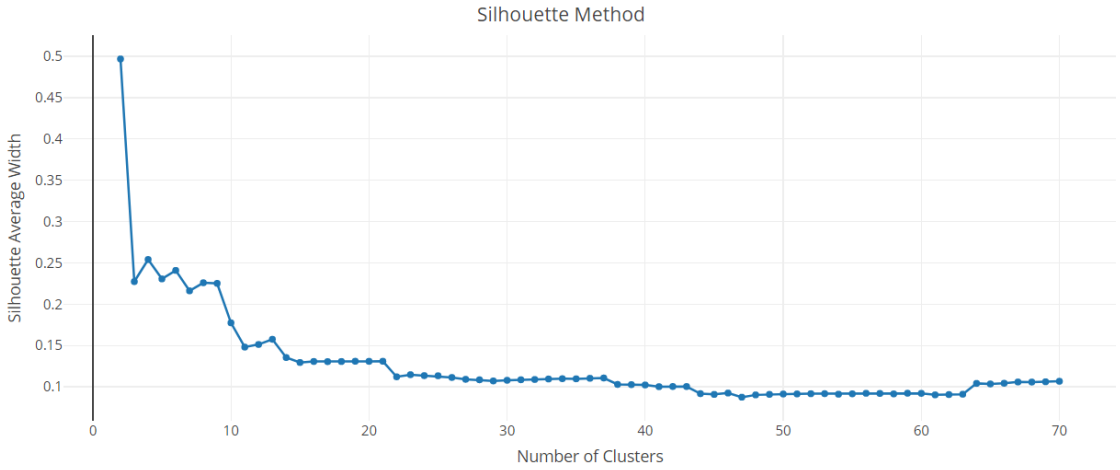
todoa, cluster-en trinkotasuna neurtzeko erabiltzen da eta i. elementuaren silhouette balioa honela definitzen da:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{8.1}$$

non

- $a(i)$: i. elementuak cluster berdinean dauden beste elementuekiko duen batezbesteko distantzia (taldekatzeko erabilitako distantzia berdina).
- $b(i)$: i. elementuak gertuen duen (eta talde berekoa ez den) cluster-eko elementu guztiekiko batezbestesko distantzia.

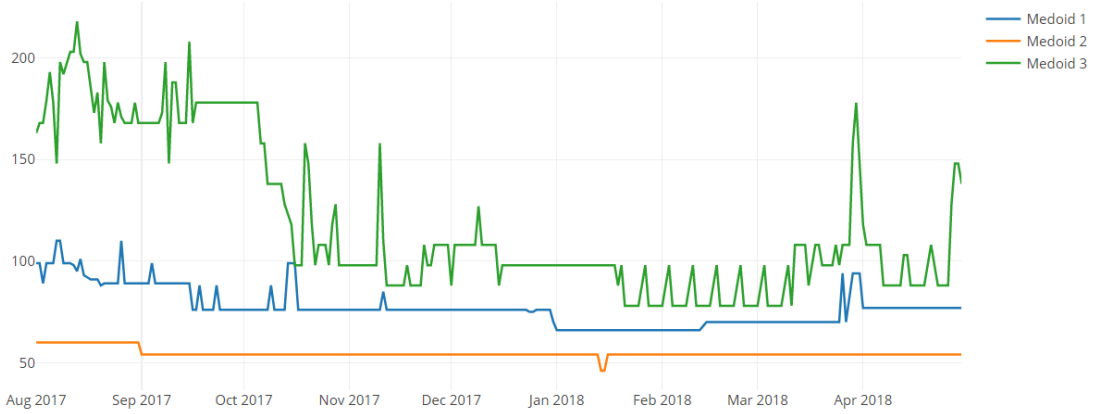
Beraz, silhouette metodoan, elementu bakoitzaren $s(i)$ balioa kalkulatzen da eta ondoren balio guztien media atera. Emaitza horiekin grafiko bat eratzen da, Irudia 8.2-n ikus daitekeen bezala.



Irudia 8.2: Clustering prezio absolutuekin lortutako Silhouette grafikoa.

Behin grafikoa marraztutakoan, cluster kopurua aukeratzeko era bat jaitsiera leuntzen hasten den puntua izaten da, kasu honetan 3 puntuan (konturatu grafikoa 2 zenbakitik hasten dela, `pam` funtzioak ez duelako talde bakar bat osatzen uzten). Kontutan hartu, silhouette metodoa irizpide bat dela, ez dela talde kopuru zehatz bat hartzera behartzen duen zerbait. Gure kasuan lehendabizi 3 taldetan banatu ziren serie denboralak (Irudia 8.3-n talde horien medoideak agertzen dira), baina informazio gutxi ateratzea lortu zenez beste bi taldekatze ezberdinekin probatu zen: 7 eta 10 cluster-ekoak hain zuzen ere (grafikoan jaitsiera zuten puntuak zirelako).

Azken bi clustering horiek aztertu ondoren, gure serie denboralak 7 taldetan multzokatzea erabaki zen, talde bakoitzetik informazio interesgarria lortzen zelako eta 10 taldetan banatzerakoan talde batzuk oso antzekoak zirelako (ia bereizi ezinak). Lortutako emaitzen medoideak Irudia 8.4-n ikus daitezke. Bertan, begi bistaz ikus daitekeen bezala talde bat besteekiko oso desberdina da, prezio oso altuak dituzten



Irudia 8.3: Clustering prezio absolutuekin lortutako medoideen grafikoa (3 taldetan multzokatzean).

hotelekin osatutakoa hain zuzen ere. Hotel oso gutxik osatzen dute talde hori eta pentsatzekoa den bezala, luxu handikoak dira.

Cluster berezi hori alde batera utziz, beste 3 aztertuko dira xehetasun gehiagorekin:

Cluster 5

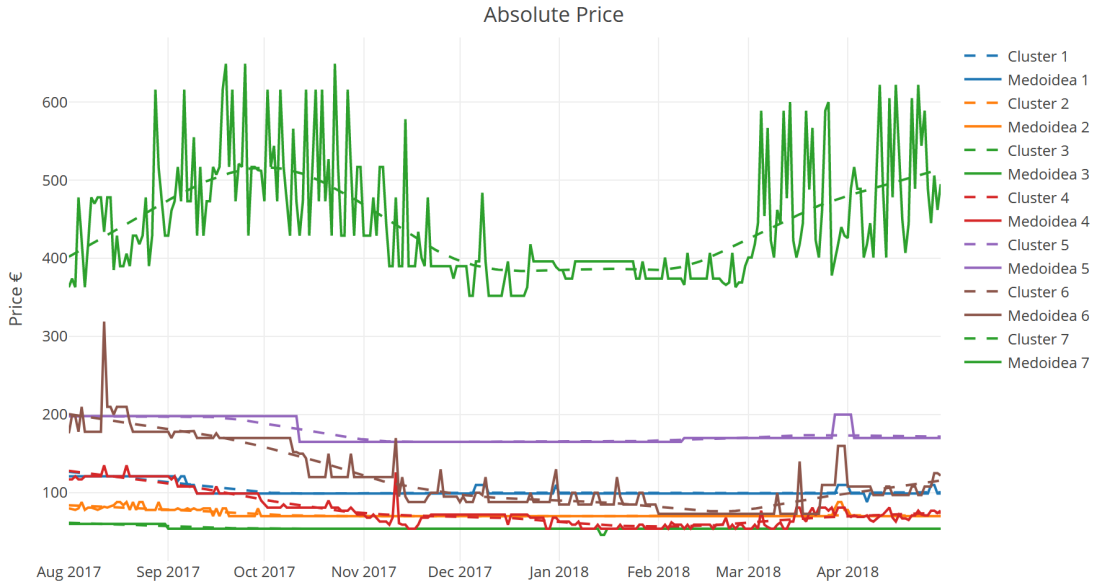
Talde honetan, hotel eta pentsio garestienak daude (7. cluster-eko luxuzko hotelak alde batera utziz gero). Gehienak 4 eta 5 izarreko hotelak dira, espero zitekeen bezala (4 eta 5 izarreko hotelen %25-a baino gehiago talde honetan aurkitzen da) eta bigarren talderik txikiena da 22 ostaturekin. Prezio aldetik, 300-180€ inguruan dabiltza. Irudia 8.5-n ikus daitezke talde honen ezaugarri batzuk.

Cluster 4

Hotel/Pentsio hauek prezio absolutuan 130-60€ inguruan dabiltza eta orokorrean prezio bariazio handikoak dira, urtaroek eta urteko sasoi desberdinek eragin handia dutelarik. 90 hotel/pentsio talde honetan daude (EAEko gelen %20 baino gehiago eskaintzen dutelarik[2]) eta Donostia eta Donostiako metropolialdean kokatzen dira gehientsuenak. Oro har pentsioak dira, 2 izarreko pentsioen %40-a talde honetan daude eta izar bakarreko pentsioen %25-a baino gehiago ere, Irudia 8.6-n ikusi daitekeen bezala. Grafiko horietatik beste datu interesgarri bat atera daiteke, Gipuzkoa eta Bizkaia barrualdean ez dagoela horrelako ostaturik.

Cluster 3

[2]Portzentaia hori kalkulatzeko orduan, datuak jaso diren 443 ostatuen gela kopuru totaletik atera da.



Irudia 8.4: Prezio absolutuekin lortutako cluster-en medoideen grafikoa (7 taldetan multzokatzean). Linea jarraituak medoideak dira eta ez-jarraituak berriz medoide horien tendentzia.

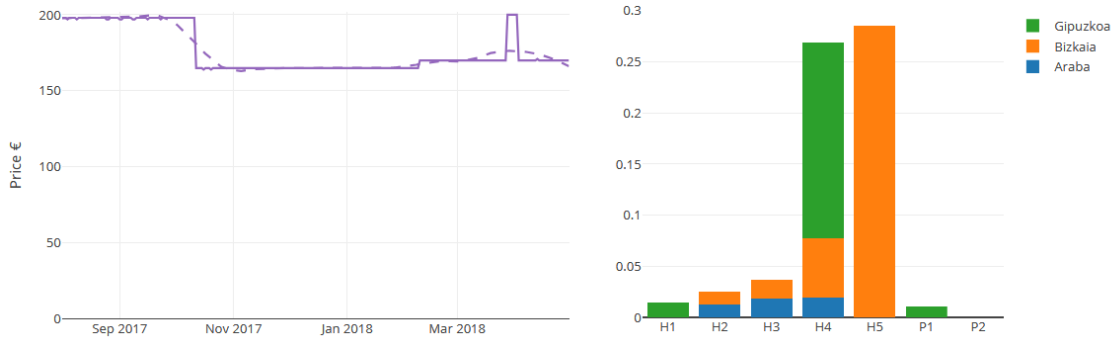
Hotel/Pentsio hauek prezio absolutuan 60-50€ inguruan dabiltza, prezio merkeena dutenak dira eta ez dute prezioa apenas aldatzen urtaroaren arabera. Merkeenak direnez, espero den bezala gehienak 3 izar baino gutxiagokoak dira eta orokorrean gela gutxi eskaintzen dituzte, hau da, tamainu txikiko ostatuak dira: 104 ostatu daude talde honetan eta gelen %15-a baino gutxiago eskaintzen dute. Errioxa Arabarrean eta Donostian ez dago ia horrelako ostaturik eta bestalde, Arabako (Gasteiz eta Errioxa Arabarra kanpoan utzita) ostatu gehienak, %60-a baino gehiago, talde honetan daude, Irudia 8.7-n ikus daitekeen bezala.

8.3 Clustering prezio normalizatuekin

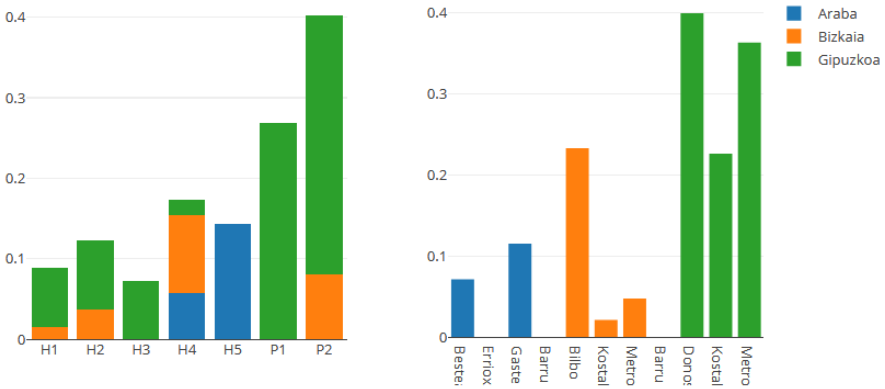
Bigarren clustering honetan, prezio normalizatuen bidez taldekatu dira ostatuak. Erabili den normalizazio metodoa *Min-Max* metodoa da, hau da, hotel bakoitzaren prezioak eskalaz aldatu dira: 0 eta 100 bitartera hain zuzen ere. Horretarako ostatu bakoitzerako erabili den normalizazio formula hurrengoa da:

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \cdot 100 \tag{8.2}$$

- x: Ostatuaren prezio-bektorea
- x_i: Ostatuaren i. eguneko prezioa
- z_i: Ostatuaren i. eguneko prezio normalizatua, $[0, 100]$ tartean egongo dena



Irudia 8.5: Ezkerreko grafikoan 5. cluster-eko medoidea eta bere tendentzia ageri dira, ikusten denez 200-170€ bitartean dabil. Eskuinean berriz ostatuen kategoriak agertzen dira portzentajetan.

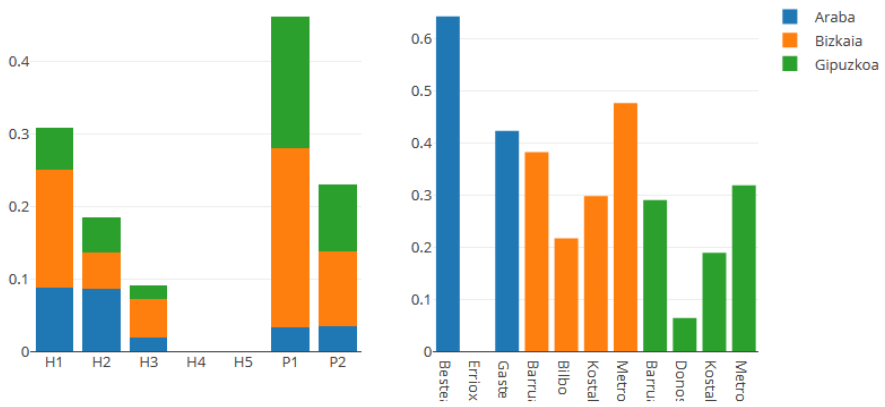


Irudia 8.6: Ezkerreko grafikoan cluster 4-ko ostatuen kategoria agertzen da portzentajetan eta eskuinekoan berriz estratoka.

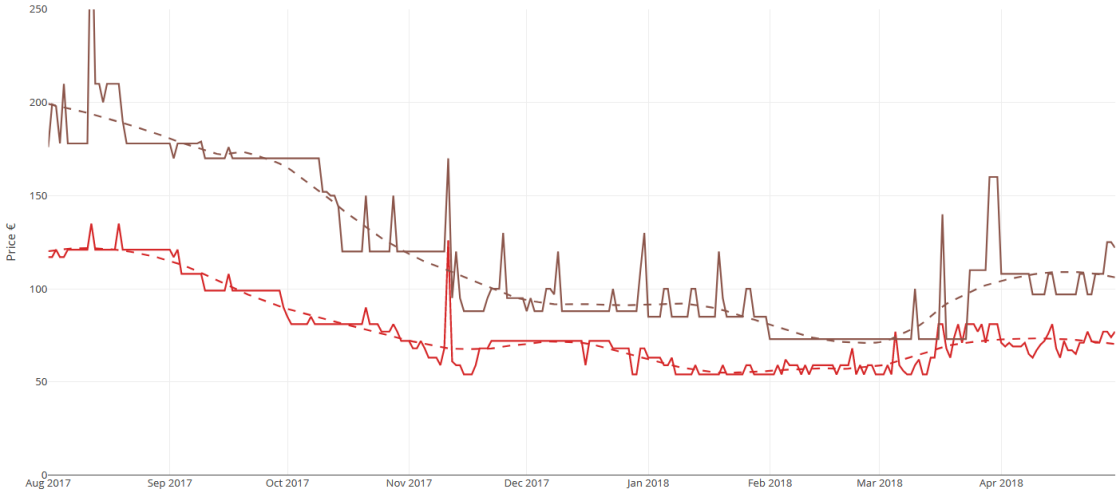
Eskala aldaketa hau, hotel/pentsio guztiak era berdin batean konparatzeko egin da, ondoren beraien tendentzia aztertzeko eta horren arabera sailkatzeko. Irudia 8.8-n adibide nahiko garbi bat dago. Bertako bi pentsioek prezio dezente ezberdinak dituzte, baina bien tendentzia oso antzekoa da. Kasu horretan, gure sailkapena egiterako orduan, bi pentsio horiek talde berdinean egotea esperoko dugu.

Datuekin lanean hastean ordea, arazoak izan ditzakegu (8.2) formula aplikatzerako orduan, izan ere $\max(x) = \min(x)$ kasuetarako indeterminazio bat dugulako. Kasu horretan, prezio konstantea duten ostatuei irteera bat eman behar zaie eta kasu honetan 100 balio konstantea esleitu zaie (3 balio ezberdinekin probatu da, 0, 50 eta 100 eta emaitza gustukoenak 100 balioarekin lortu dira). Hortaz gain, prezio aldaketa maximoa 5€-koa edo txikiagoa izan duten ostatu guztiei ere 100 balio esleitu zaie. Izan ere, 5€-ko aldaketa, aldaketa *oso txikia* dela kontsideratu da eta *Min-Max* normalizazioarekin ostatu horiek oszilazio oso handiak izaten dituzte.

Prezioak behin normalizatuta, aurreko clustering metodo berdina jarraitu dugu distantzia euklidearraren bidez. Lortutako silhouette grafikoa Irudia 8.9-n ikus daiteke. Kasu honetan, era garbiago eta errazago batean ikusten da ebaki puntua



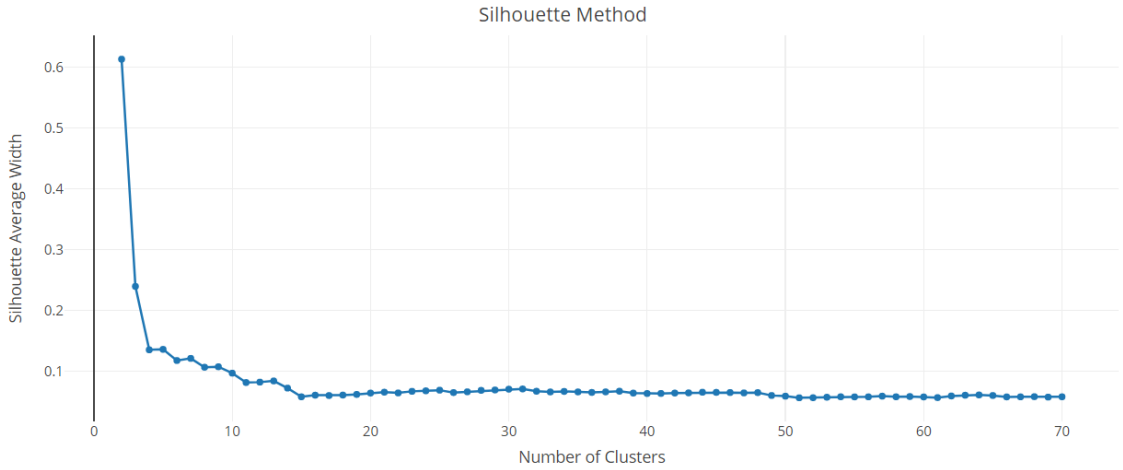
Irudia 8.7: Ezkerreko grafikoan cluster 3-ko ostatuen kategoria agertzen da portzentajetan eta eskuinekoan berriz estratoka.



Irudia 8.8: Grafiko honetan, prezio-tarte desberdinetan mugitzen diren bi pentsio ageri dira. Hala ere, argi ikus daiteke biek duten tendentzia nahiko berdintsua dela.

eta ondorioz 4 talde sortu ditugu (gogoratu lehenengo puntuak 2 cluster adierazten duela).

4 cluster horiekin lortutako emaitza Irudia 8.10-n ikus daiteke eta nahiko era errazean desberdintzen dira beraien artean. 4. taldea, aurrerago definitu dugun bezala prezio konstantea edo ia konstantea duten hotel eta pentsioak dira, barrualdean kokatzen direnak eta kategoria baxukoak. Beste taldeak aldiz, sasoiaren arabera prezioak aldatzen joaten dira. Gehien aldatzen dutenak 1. cluster-ean daude eta gutxien aldatzen dutenak 3. cluster-ean. Azken honetan, prezioaren tendentzian aldaketa gutxi edo ia ezer ere ez jasaten dutenak daude, baina 4. taldekoekin alderatuz, hauek prezio aldaketak jasaten dituzte urteko egun berezietan edo/eta astebukaeretan.



Irudia 8.9: Grafiko honetan silhouette metodo bidez lortutako emaitzak ikus daitezke. Bertan, 3 eta 4 cluster-ekin silhouette balioa asko txikitzen dela ikusten da eta bertatik aurrera hobekuntza oso txikia dela.

Cluster 1

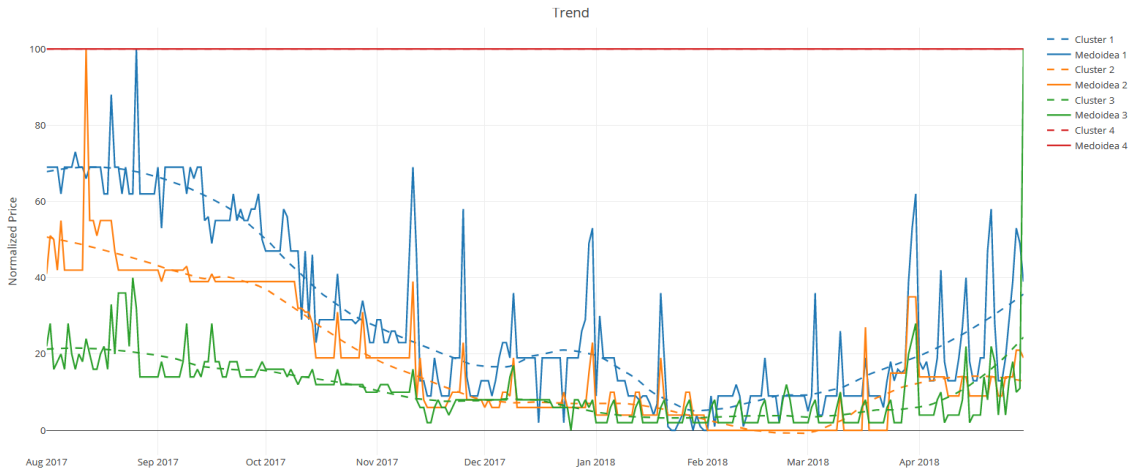
Cluster honetan aurkitzen dira prezioaren tendentzian (prezio normalizatuetan) aldaketa handienak pairatzen dituzten hotel eta pentsioak. Irudia 8.11 aztertzen bada, pentsioek eta 4 izarreko hotelek pisu handia dutela ikus daiteke talde honetan. Bestalde, Donostiako ostatuen ia %70-a era honetakoak dira. Gipuzkoa kostaldeko, Donostiako metropolialdeko eta Bilboko ostatuen ia %50-a ere talde honetan aurkitzen dira. 191 hotel/pentsio dira era honetakoak eta gela guztien %45 inguru eskaintzen dute.

Cluster 2

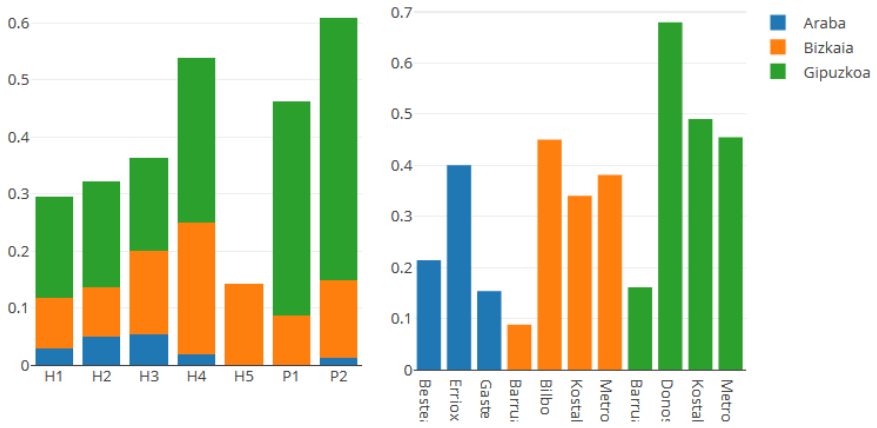
3, 4 eta batik bat 5 izarreko hotelek garrantzia berezia dute talde honetan. Estratoeri dagokionez, Gipuzkoa kostaldeko ostatuen %45-a baino gehiago hemen aurkitzen dira eta ikus daitekeenez, barrualdetik kanpo daude gehientsuenak. 130 ostatuk osatzen dute multzo hau eta gelen %35 inguru eskaintzen dute.

Cluster 3

1, 2 eta 5 izarreko hotelak dira portzentajean gehien agertzen direnak talde honetan. Ostatu hauek, prezioaren eboluzioan gorabehera gutxi jasaten dituzte, hau da, urtaroak ez du garrantzia handiegirik eskainitako prezioetan. Hala ere, oro har prezioak aldatzen dituzte egunen arabera, batzuen kasuan astean zehar eta beste batzuen kasuan egun berezietan (jai-egunak, ospakizun bereziak...). Estratoei dagokionez, Arabak (Errioxa Arabarra kanpoan utziz) eta Bizkaia barrualdeak batik bat horrelako hotel/pentsioak eskaintzen dituzte.



Irudia 8.10: Prezio normalizatuekin lortutako clustering-a. Lerro jarraituak cluster-eko medoideak dira eta lerro ez-jarraituak medoidearen tendentzia.



Irudia 8.11: Cluster 1-eko ostatuen sailkapena kategoria eta estratoka.

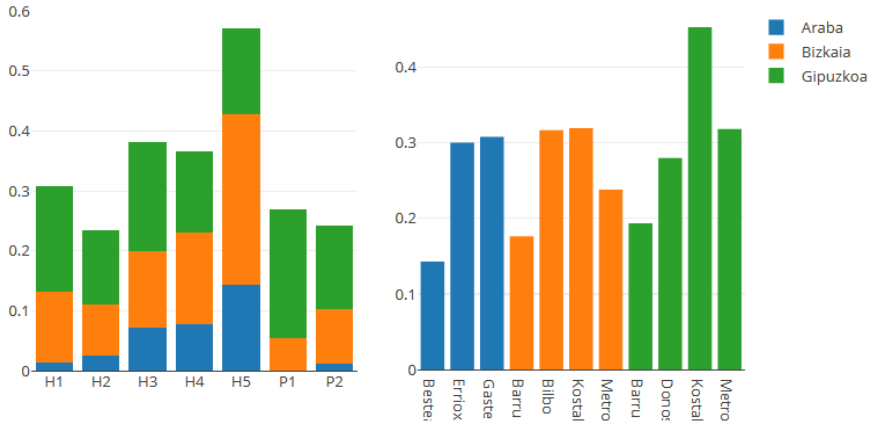
8.4 Clustering aldakortasunarekin

Egin den azken clustering mota aldakortasunean oinarrituta dago (*volatility*), prezioek egun batetik bestera dituzten aldaketak neurtzeko asmoarekin. Taldekatze honen helburu nagusia, epe motzean prezioa asko, gutxi edo ezer ere ez aldatzen duten hotel/pentsioen sailkapen bat egitea da. Horretarako, ekonomian erabiltzen den neurri baten oinarritu da azterketa: **Close-to-Close Volatility** edo **Close/Close Volatility** [1].

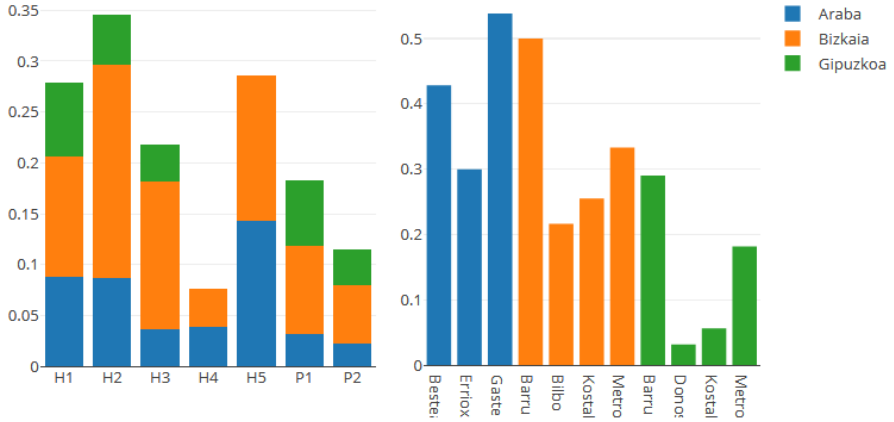
Neurri horren ideia nagusia, ondoz ondoko egunen arteko diferentzia aztertzea da logaritmoaren bidez, ondoren balio horien desbiderapen estandarra kalkulatzeko. Adibide bezala, demagun hurrengo prezio bektore bat dugula:

$$90, 90, 90, 90, 100, 105, 80, 90, 90, 90, 90, 100, 105, 80$$

Hasteko, balio horiei logaritmoa aplikatzen zaie (modu honetan, 80€ -ko ostatu



Irudia 8.12: Cluster 2-ko ostatuen sailkapena kategoria eta estratoka.



Irudia 8.13: Cluster 3-ko ostatuen sailkapena kategoria eta estratoka.

bat eta 200€ -ko beste bat eskala berdintsuagotan mugitzen dira):

$$4.5, 4.5, 4.5, 4.5, 4.605, 4.654, 4.382, 4.5, 4.5, 4.5, 4.5, 4.605, 4.654, 4.382$$

ondoren, ondoz ondoko diferentzia kalkulatzen da (kontutan hartu bektorearen luzera txikituko dela unitate batean):

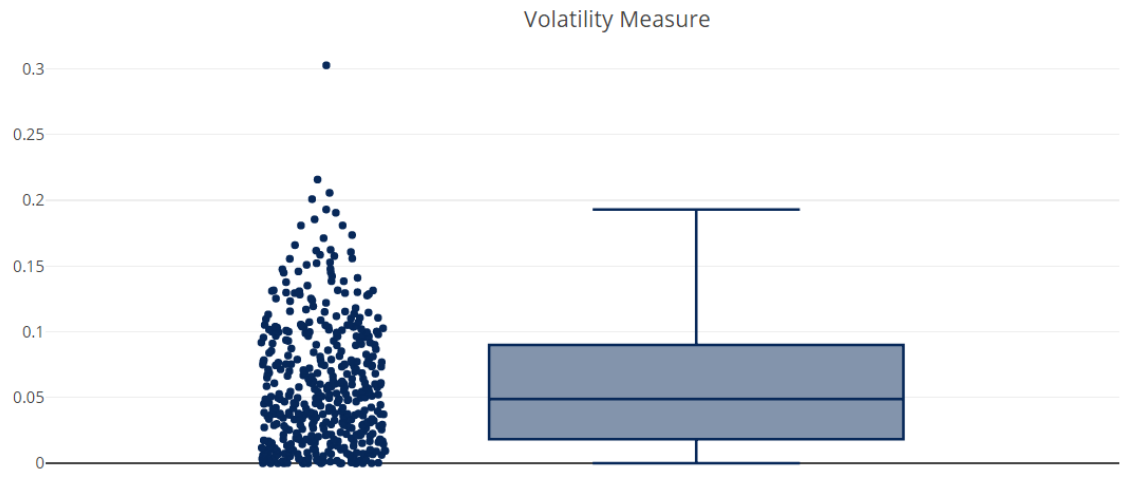
$$0, 0, 0, 0.105, 0.049, -0.272, 0.118, 0, 0, 0, 0.105, 0.049, -0.272$$

eta azkenik, desbiderapen estandarra kalkulatzen da.

Lan honetarako, ideia berdina hartu da, baina amaieran, desbiderapen estandarra kalkulatu beharrean balio absolutuen media kalkulatu da eta arrazoia hurrengoa da: sinpleagoa delako eta gure analisirako nahikoa delako. Beraz, azkenean gure analisirako erabili dugun aldakortasunaren balioa era honetan kalkulatu dugu:

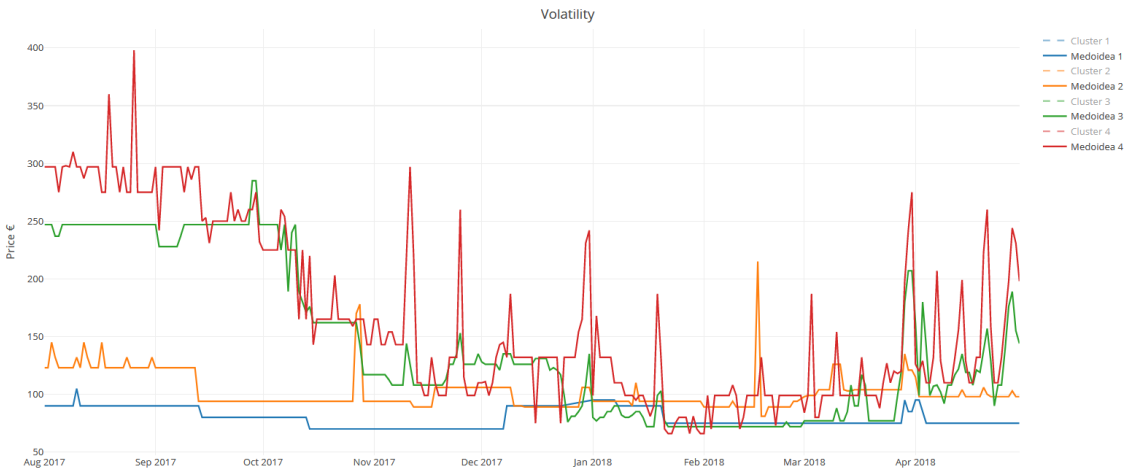
```
mean(abs(diff(log(x))))
```

non `x` prezio bektore bat izango den. Behin ostatu bakoitzari prozesu hori aplikatutakoan, balioak marraztu dira begi bistaz zerbait ikusten zen edo ez aztertzeko. Hala ere, Irudia 8.14-ko boxplot-a ikusiz ezin da ondorio nabaririk atera clustering-erako, ondorioz kuantilen bitartez eraiki dira taldeak. Probatu diren talde kopuruak 3, 4 eta 5 izan dira, amaieran 4rekin geratzeko (kuartilak).



Irudia 8.14: Hotel/pentsioei *close-to-close* algoritmoa aplikatzerakoan lortutako balioen boxplot-a.

Kasu honetan, eraiki den moduagatik ez da medoiderik behar cluster bakoitzeko, baina aurreko grafikoen estilo berdina mantentzeko asmoarekin, talde bakoitzetik ostatu bat aukeratu da taldearen ordezkari bezala (aukera desberdinak probatu dira eta bisualki egokiak direnak hartu). Emaitzak Irudia 8.15-n ikus daitezke (aldakortasun txikienetik handienera ordenatuta daude).

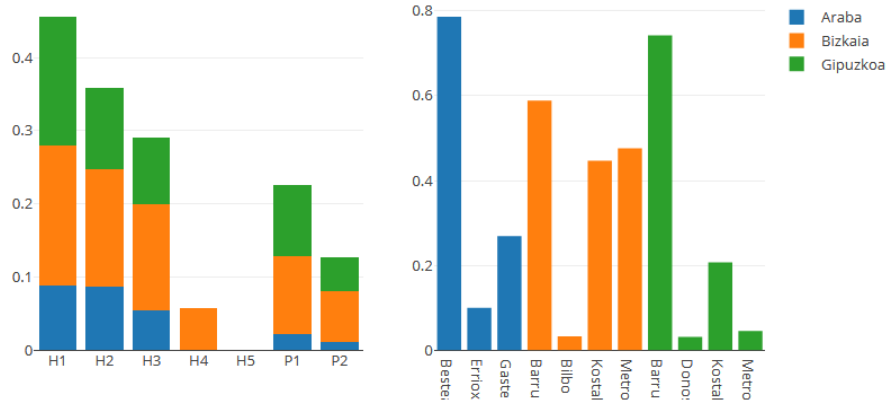


Irudia 8.15: Aldakortasunaren araberako clustering-a. Cluster-ak aldakortasun txikienetik handienera ordenaturik daude.

Orain, besteetan bezala 3 talde aztertuko dira detaile gehiagorekin:

Cluster 1

Aldakortasun gutxien dutenak aurkitzen dira hemen, hau da, prezio aldaketak oso noizean behin egiten dituztenak edo inoiz ere ez. Irudia 8.16-n ikus daitekeen bezala, gehienak 3 izar edo gutxiagoko hotel/pentsioak dira (izar bateko hotelen %40-a baino gehiago talde honetan aurkitzen da) eta barrualdean kokatzen dira batik bat. Gipuzkoa barrualdean eta Araban (Errioxa Arabarra eta Gasteiz kanpoan utziz gero) ostatuen %75-a baino gehiagok prezioetan aldakortasun oso gutxi dutela ikus daiteke. Bestalde, gelen %15-a bakarrik eskaintzen dute talde honetan, ostatu txikiak direnaren seinale.



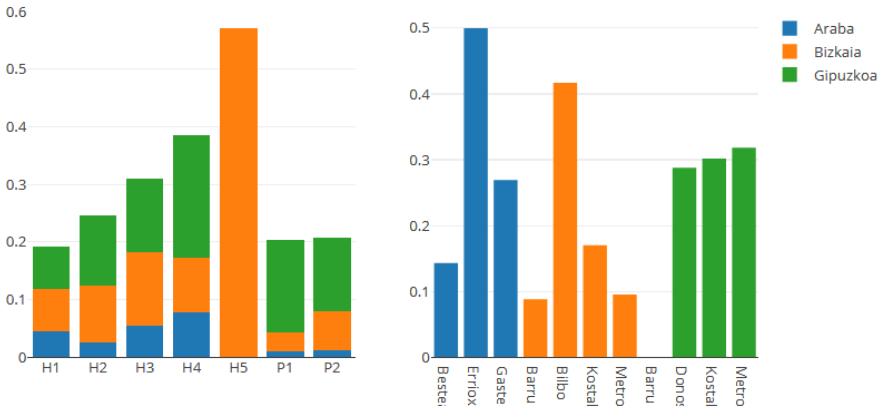
Irudia 8.16: Cluster 1-eko ostatuen sailkapena kategoria eta estratoka.

Cluster 3

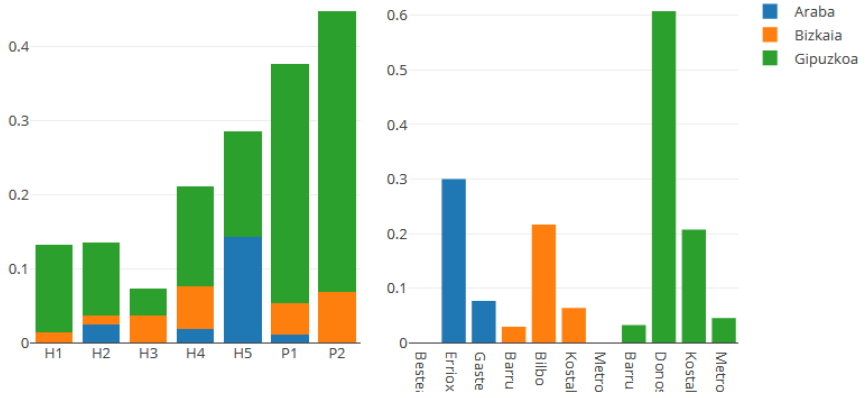
3. talde honetan, barrualdeko hotel/pentsio oso gutxi daude, Bilbon eta Errioxa Arabarrean asko pilatzen direlarik, %40-a baino gehiago (Irudia 8.17-n ikus daiteke). Bestalde, oro har kategoria altuko hotelak dira, 3, 4 eta 5 izarrekoak, Bilboko 5 izarreko hotel denak adibidez talde honetan aurkitzen dira. Ostatu hauek, prezioan aldakortasun dezente izaten dituzte, baina ez dira gehien aldatzen dituztenak.

Cluster 4

Aldakortasun handiena dutenak dira eta Irudia 8.18 ikusten baldin bada, argi geratzen da gehienak pentsioak direla eta Donostian kokatzen direla (%60-a baino gehiago). Bestalde, gelen %16-a bakarrik eskaintzen dute, beraz ostatu txikiak dira oro har.



Irudia 8.17: Cluster 3-ko ostatuen sailkapena kategoria eta estratoka.



Irudia 8.18: Cluster 4-ko ostatuen sailkapena kategoria eta estratoka.

Bibliografia

[1] Bennett, C., and Gil, M. A. Measuring historical volatility.

[2] Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] Canavos, G. *Probabilidad y Estadística*. McGraw Hill, 1994.

[4] Cappé, O., Moulines, E., and Ryden, T. *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[5] Casella, G., and Berger, R. *Statistical Inference*. Duxbury Resource Center, June 2001.

[6] Chang, W., Cheng, J., Allaire, J., Xie, Y., and McPherson, J. *shiny: Web Application Framework for R*, 2017. R package version 1.0.5.

[7] Chen, E. Winning the Netflix Prize: A Summary. `http://blog.echen.me/2011/10/24/winning-the-netflix-prize-a-summary/`.

[8] Cheng, J., Karambelkar, B., and Xie, Y. *leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library*, 2018. R package version 2.0.0.

[9] de Lacalle, J. L. *tsoutliers: Detection of Outliers in Time Series*, 2017. R package version 0.6-6.

[10] Devore, J. L. *Probability and Statistics for Engineering and the Sciences*, 8th ed. Brooks/Cole, January 2011. ISBN-13: 978-0-538-73352-6.

[11] Garey, M. R., and Johnson, D. S. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[12] Giorgino, T. Computing and visualizing dynamic time warping alignments in R: The dtw package. *Journal of Statistical Software 31*, 7 (2009), 1–24.

[13] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[14] Grubbs, F. E. Sample criteria for testing outlying observations. *Ann. Math. Statist. 21*, 1 (03 1950), 27–58.

[15] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[16] Hyndman, R. J. *forecast: Forecasting functions for time series and linear models*, 2017. R package version 8.2.

[17] James, G., Witten, D., Hastie, T., and Tibshirani, R. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.

[18] Kaufman, L., and Rousseeuw, P. *Finding Groups in Data: an introduction to cluster analysis*. Wiley, 1990.

[19] Komsta, L. *outliers: Tests for outliers*, 2011. R package version 0.14.

[20] Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer 42*, 8 (Aug 2009), 30–37.

[21] Kriesel, D. *A Brief Introduction to Neural Networks*. 2007.

[22] Levin, D. A., Peres, Y., and Wilmer, E. L. *Markov chains and mixing times*. American Mathematical Society, 2006.

[23] Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., and Hornik, K. *cluster: Cluster Analysis Basics and Extensions*, 2018. R package version 2.0.7-1 — For new features, see the 'Changelog' file (in the package source).

[24] Montero, P., and Vilar, J. A. TSclust: An R package for time series clustering. *Journal of Statistical Software 62*, 1 (2014), 1–43.

[25] Mori, U., Mendiburu, A., and Lozano, J. A. Distance measures for time series in r: The TSdist package. *R journal 8*, 2 (2016), 451–459.

[26] Moritz, S. *imputeTS: Time Series Missing Value Imputation*, 2017. R package version 2.5.

[27] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[28] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

[29] Rabiner, L. R. Readings in speech recognition. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990, ch. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pp. 267–296.

[30] Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B. *Recommender systems handbook*. Springer, New York; London, 2011.

[31] Shumway, R., and Stoffer, D. *Time Series Analysis and Its Applications: With R Examples*. Springer Texts in Statistics. Springer International Publishing, 2017.

[32] Sievert, C., Parmer, C., Hocking, T., Chamberlain, S., Ram, K., Corvellec, M., and Despouy, P. *plotly: Create Interactive Web Graphics via 'plotly.js'*, 2017. R package version 4.7.1.

[33] Strang, G. *Linear algebra and its applications*. Thomson, Brooks/Cole, Belmont, CA, 2006.

[34] Wickham, H. Reshaping data with the reshape package. *Journal of Statistical Software 21*, 12 (2007), 1–20.