Projecte 2

Etiquetador de roba

Nerea de la Torre Veguillas, 1669013

Mara Montero Jurado, 1671506

Júlia Morán Fluvià, 1667730

Adrián Prego Gallart, 1672251

Intel·ligència artificial

Curs 2023-2024

Introducció

Aquesta pràctica se centra en desenvolupar un etiquetador automàtic d'imatges de peces de vestir utilitzant els algorismes K-means i K-NN.

Es divideix en tres parts: classificació de colors amb K-means, etiquetat de formes amb K-NN i combinació de tots dos mètodes per a obtenir un etiquetatge complet. L'objectiu és aconseguir un etiquetador precís i eficient que pugui assignar etiquetes de tipus de peça i color a les imatges.

Per a la identificació dels colors predominants en les imatges mitjançant etiquetatge automàtic del color, s'utilitza l'algorisme K-means en un enfocament no supervisat. S'aborda el problema considerant només 11 colors universals bàsics i assignant a cada píxel un vector de característiques RGB. A continuació, es fan servir mètodes d'agrupació per a reconèixer els tons més destacats.

Per a l'etiquetatge automàtic de la forma, s'utilitza la supervisió amb l'algorisme K-NN per classificar les imatges en categories de peces de roba. Les imatges presenten característiques específiques i s'utilitza l'algorisme K-NN per assignar una etiqueta a cada imatge segons aquestes característiques.

S'ha decidit plantejar l'anàlisi mitjançant gràfiques dels resultats obtinguts per tal d'observar les conclusions de forma més visual. També s'inclouen instruccions específiques per a implementar de la millor forma possible aquests algorismes, com ara la determinació del valor òptim de K en K-means, la creació de funcions per a la classificació en K-NN i el desenvolupament d'un sistema de cerca utilitzant etiquetes de color i forma.

Introducció al K-Means i K-NN

El K-means és un algorisme utilitzat per identificar agrupacions de punts dins d'un conjunt de dades, permetent dividir i classificar aquestes dades en K grups diferents, cadascun amb el seu propi centre de massa. En el nostre classificador, el K-means és l'algorisme responsable d'etiquetar les imatges segons els colors que contenen, considerant els percentatges de cada color present en les imatges.

A l'hora d'executar l'algoritme, es pot fer una prèvia selecció d'opcions introduïdes amb un diccionari, anomenat "options", nosaltres afegirem diversos mètodes en dues opcions per tractar de millorar l'eficàcia del K-Means:

- Inicialització dels centroides: *km_init* . El mètode originalment implementat és "first", que assigna als centroides als primers K punts de la imatge X que siguin diferents entre ells
- Mètode per calcular la millor K: *fitting* . El mètode originalment implementat és "WCD", que és la seva distància intra-clas o *within-class-distance*

Però també podem establir un valor a *max_iter*, *verbose*, *tolerance*..

D'altra banda, el K-NN és un algorisme que determina la classe de cada punt basant-se en la classe del seu veí més proper. En el nostre sistema de classificació, el K-NN és l'algorisme utilitzat per etiquetar les imatges segons les formes que contenen, assignant una o dues etiquetes a cada imatge segons els percentatges de les diferents formes presents.

Utilitzaren el dataset *train_data* per entrenar l'algoritme i el *test_data* per classficar-lo.

Mètodes d’anàlisi implementats

Mètodes qualitatius:

Aquest anàlisi ens ajuda a provar l’eficàcia dels nostres classificadors d’una manera visual. S’ha implementat a partir de les etiquetes que retornen el KMeans i el KNN, mitjançant funcions que mostren per pantalla les imatges que contenen les etiquetes especificades en "queries" (string que conté l’objecte que volem visualitzar, el color o la peça de roba, en aquest cas).

retrieval_by_color(), *retrieval_by_shape*() i *retrieval_combined*(), són les funcions analítiques implementades:

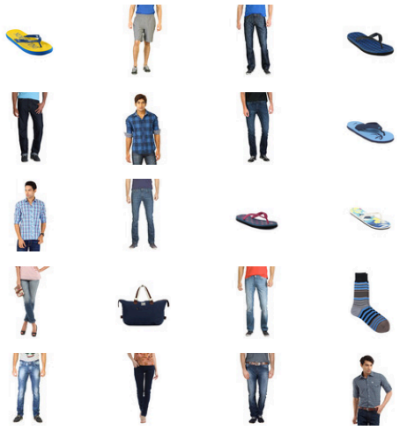
retrieval_by_color:

Aquesta funció rep un conjunt d'imatges, una llista d'etiquetes de color assignades a cadascuna d'aquestes imatges i un o més colors de consulta pels quals volem cercar les peces de roba de les imatges. L'objectiu és trobar les imatges que coincideixen amb els colors de consulta (en aquest cas 11).

Per fer-ho, la funció itera a través de totes les imatges i les seves etiquetes de classe associades. Si tots els colors de consulta es troben en les etiquetes de color d'una imatge determinada, s'afegeix a la llista d'imatges coincidents. Al final, la funció retorna aquesta llista de coincidències.

Aquesta funció pot ser millorada si afegim el paràmetre de percentatges, ja que així podem especificar la quantitat del color desitjat. En el nostre cas, l’hem pogut implementar. Per saber amb quina peça de roba té més o menys percentatge de color, hem enviat com a paràmetre el percentatge dels colors de cada imatge. Es descarten tots els colors que tenen una proporció inferior al 30% respecte a aquests colors. Un cop revisades totes les imatges, la llista d'índexs es classifica en ordre descendent segons el percentatge del color predominant de cada imatge. Finalment, es mostren les imatges ordenades segons aquesta classificació.

Per exemple, si provem les 20 primeres imatges que continguin color blau, veiem com la primera imatge conté el color blau d'una manera molt més vibrant que no l'última.



Primeres 20 imatges Blue

retrieval_by_shape:

La funció rep una llista d'imatges, una llista d'etiquetes de classe assignades a cadascuna d'aquestes imatges i una o més classes de consulta per les quals volem cercar. L'objectiu és trobar les imatges que coincideixen amb les classes de consulta (en aquest cas, les peces de roba).

Per fer-ho, la funció itera a través de totes les imatges i les seves etiquetes de classe associades. Si totes les classes de consulta es troben a les etiquetes de classe d'una imatge determinada, s'afegeix a la llista d'imatges coincidents. Al final, la funció retorna aquesta llista de coincidències.

Primeres 20 imatges flip flops

Si fem la cerca de 20 “Flip Flops”, podem observar com no hi ha cap inconvenient en la funció.

retrieval_combined:

La funció “retrieval_combined” rep una llista d'imatges, una llista d'etiquetes de color i una llista d'etiquetes de classe assignades a cada imatge, així com els colors i les classes de consulta pels quals volem cercar. L'objectiu d'aquesta funció és trobar les imatges que coincideixen tant amb els colors com amb les peces de roba.

Ho realitza combinant les funcions anteriors iterant a través de totes les imatges i les seves etiquetes de color i de forma associada. Comprova si tots els colors de consulta i totes les classes de consulta es troben a les etiquetes que acabem de fer referència. Si és així, aquesta imatge s'afegeix a la llista de coincidències. Finalment, la funció retorna aquesta llista de coincidències que contenen tant els colors com les classes de consulta especificades.

Aquesta funció és la unió de les dues anteriors, i com que les ambdues funcionen correctament, observem com aquesta també funciona bé.

Primeres 30 imatges Pink Dress

Mètodes quantitatius:

Gràcies a aquests mètodes podem analitzar i interpretar el funcionament del K-means i el K-NN.

K-MEANS

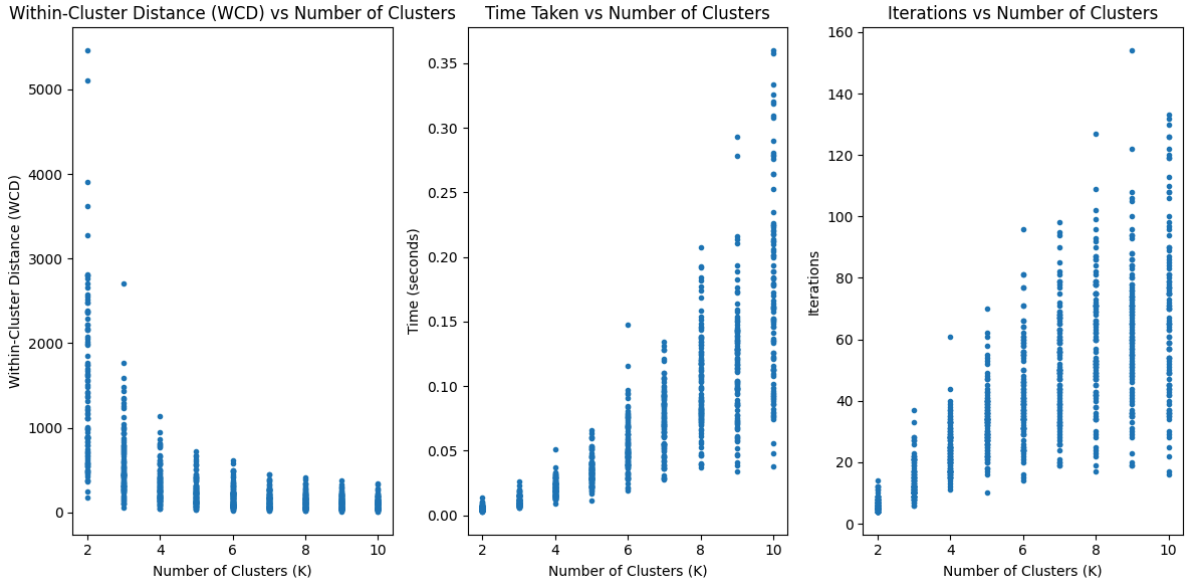
Kmeans_statistics:

La funció rep d'entrada una llista d'imatges etiquetades i un valor per a la K màxima a avaluar. L'objectiu és avaluar l'eficiència de l'algorisme K-means mitjançant la comparació de diverses mètriques com la distància dins del clúster (WCD), el temps i el nombre d'iteracions en funció del valor K.

Per a cada valor de K, la funció executa l'algoritme K-means per a cada imatge en la llista d'imatges d'entrada. Després de cada execució calcula la WCD, el nombre d'iteracions i el temps que triga l'algorisme en convergir. Aquestes dades s'afegeixen a un diccionari

d'estadístiques. Finalment, la funció visualitza aquestes estadístiques en tres gràfics diferents.

Els següents gràfics són un exemple d'output usant les opcions *km_init* = *first* i *fitting* = *WCD*:

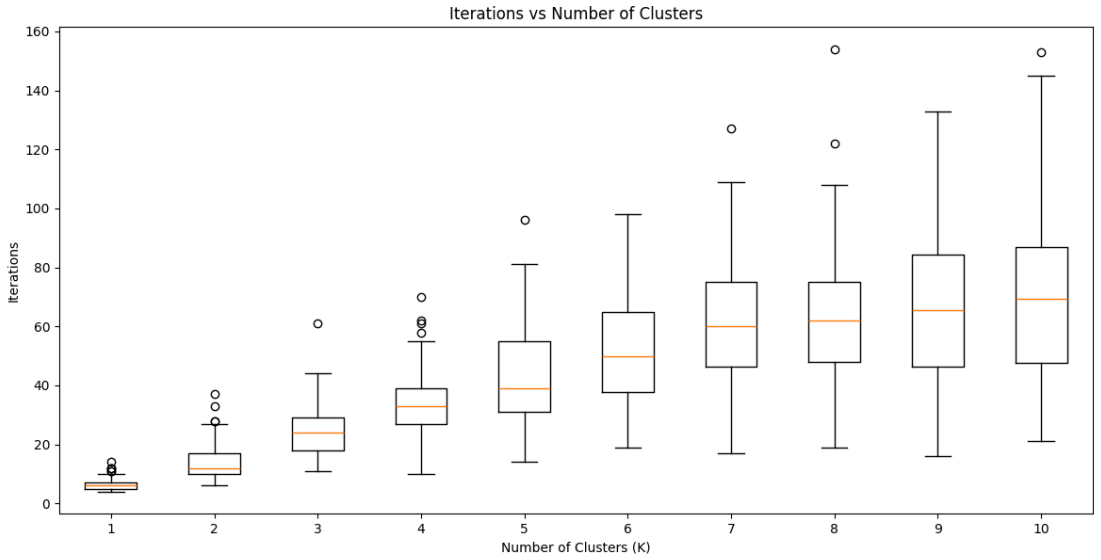


Gràfic comparatiu de K respecte a WCD | Comparació de K amb el time trigat | Gràfica de comparació de K amb les iteracions

La primera gràfica mostra com varia la distància intra-clúster (WCD) a mesura que incrementem el nombre de clústers (K). Es pot observar que a mesura que augmenta K, la WCD tendeix a disminuir. Això indica que a més clústers, els punts dins de cada clúster estan més propers entre ells, cosa que suggereix una millora en la separació dels grups de dades.

En la segona gràfica veiem com varia el temps de processament a mesura que augmentem el nombre de clústers. Es pot notar que el temps augmenta a mesura que incrementa K. Això és degut a que, amb més clústers, el model necessita més temps per convergir cap a una solució òptima.

L'última gràfica ens il·lustra com varia el nombre d'iteracions necessàries per a l'algoritme a mesura que incrementa el nombre de clústers. Es pot observar que a mesura que augmenta K, el nombre d'iteracions també ho fa, ja que més clústers requereixen més càlculs per determinar els centroids òptims i assignar els punts als clústers corresponents.



Boxplot de K respecte les iteracions

A partir d'aquest diagrama de caixes, fet amb *km_init* = *first* i *fitting* = *WCD*, veiem que la mediana es comença a estabilitzar al voltant de les 70 iteracions i que no hi ha massa outliers, el que ens diu que la majoria d'execucions de l'algorisme requereixen un nombre similar d'iteracions per convergir.

Get_color_accuracy:

La funció rep una llista d'imatges que es volen classificar, les etiquetes de referència (ground-truth) que representen els colors reals de les imatges, i el nombre màxim de clústers a considerar en l'algoritme K-means. L'objectiu és avaluar la precisió de la classificació de colors en les imatges utilitzant l'algoritme K-means.

Per calcular aquesta precisió, la funció itera a través de les imatges i aplica l'algoritme K-means a cada. Es troba la millor k per cada imatge (des d'1 fins a una k màxima) amb *find_best_K*. Després, compara els colors classificats amb les etiquetes de referència per a cada imatge. Es compta quantes d'aquestes coincidències són correctes, tenint en compte que s'eviten els colors repetits en la classificació per a cada imatge (fem la intersecció). Finalment, es calcula el percentatge de precisió de la classificació de colors com el nombre total de prediccions correctes dividit pel total de colors únics de les imatges, multiplicat per 100. La funció retorna aquest percentatge de precisió com a resultat.

La precisió del *Get_color_accuracy* és definida pel valor màxim de clústers (k_max). Podem afirmar això degut a que quan tenim una k_max baixa, així com k_max=3, tenim una accuracy de 45.08 %. D'altra banda, quan la k_max augmenta (k_max=6), l'accuracy és del 53.19 %. Aquest augment en l'accuracy ve donada a que amb un valor més alt de k_max, s'estan creant més clústers per a representar les dades. Això pot ajudar l'algorisme a capturar millor la variabilitat en els colors presents, la qual cosa condueix a una major precisió en la classificació dels colors.
Aquestes proves s'han fet amb el data *test_imgs*, *WCD i first*.

K-NN

Get_shape_accuracy:

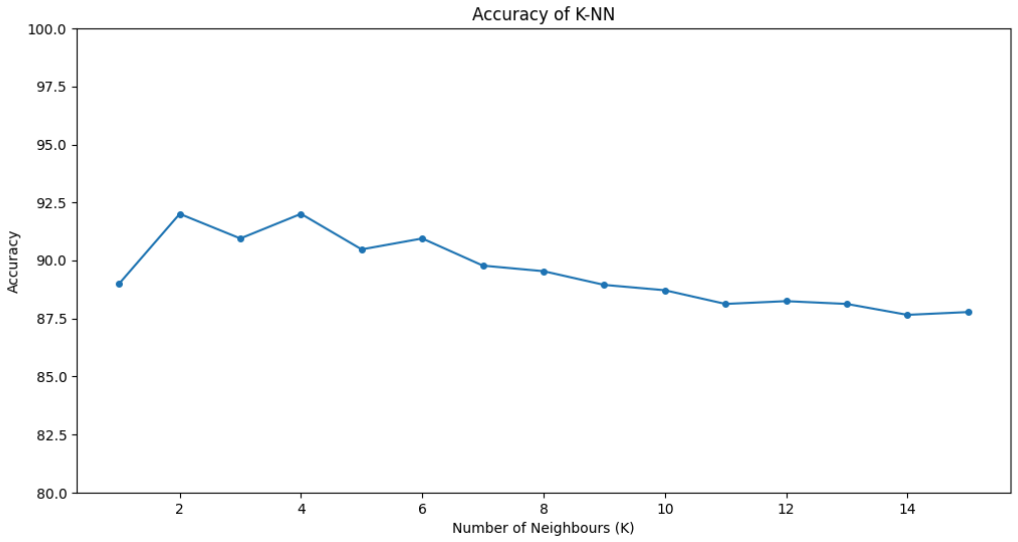
La funció rep una llista d'imatges que es volen classificar, el conjunt de dades d'entrenament utilitzat pel K-NN per a la classificació, les etiquetes corresponents al conjunt de dades d'entrenament, les etiquetes de referència (ground-truth) que representen les formes reals de les imatges i el nombre de veïns més propers a considerar en l'algoritme K-NN.

Per calcular la precisió de la classificació de formes, la funció itera a través de les prediccions del K-NN i les etiquetes de classe de referència (ground-truth). Compara les

prediccions amb les formes reals i compta quantes d'aquestes coincideixen. A continuació, calcula el percentatge de precisió, que és el nombre de prediccions correctes dividit pel total de formes, multiplicat per 100. Finalment, la funció retorna el percentatge de precisió de la classificació de formes basat en les prediccions del K-NN.

La precisió del "*Get_shape_accuracy"* és definida pel nombre de veïns (k_nn). Això ve degut a com es distribueixen les classes en l'espai de característiques i com afecta l'elecció del nombre de veïns més pròxims a la decisió final de classificació.

Quan la k_nn = 6, tenim una accuracy del 85%. En canvi, si la k_nn=3, l'accuracy és del 93,33%. La precisió és més baixa amb un nombre de veïns més alt és perquè alguns casos es poden classificar incorrectament a causa de la inclusió d'un veí addicional que pot no ser representatiu de la classe correcta. D'altra banda, amb un nombre de veïns més baix, és possible que s'obtingui una precisió més alta en considerar únicament els tres veïns més pròxims, la qual cosa pot capturar millor la classe a la qual pertany segons la forma.



Gràfica accuracy K-NN

Mètodes de millora de classificació

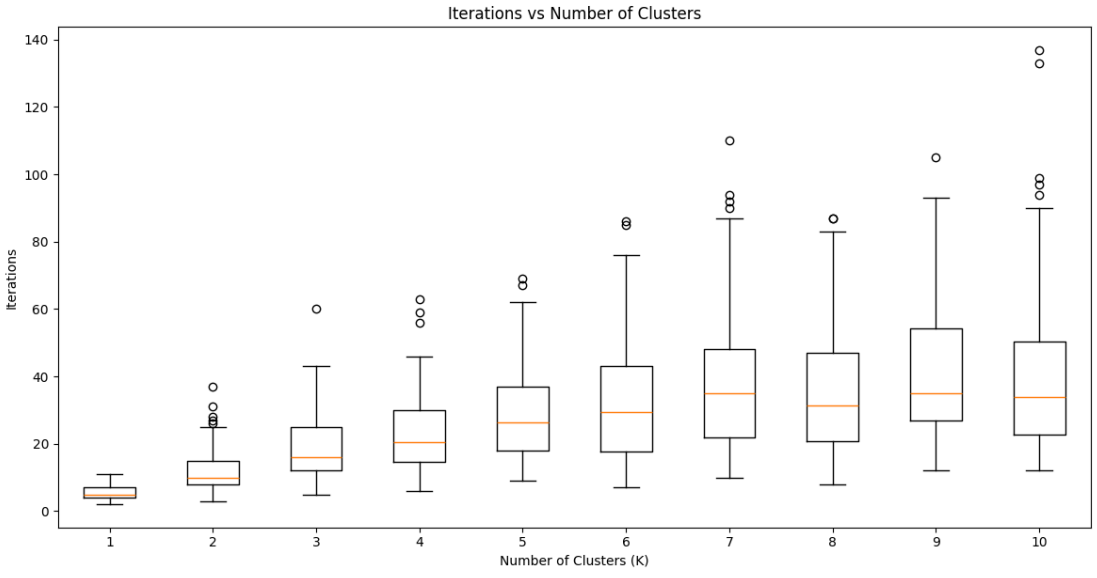
Hem implementat els 4 canvis i noves funcionalitats en els mètodes K-Means i K-NN que exposarem a continuació.

K-MEANS

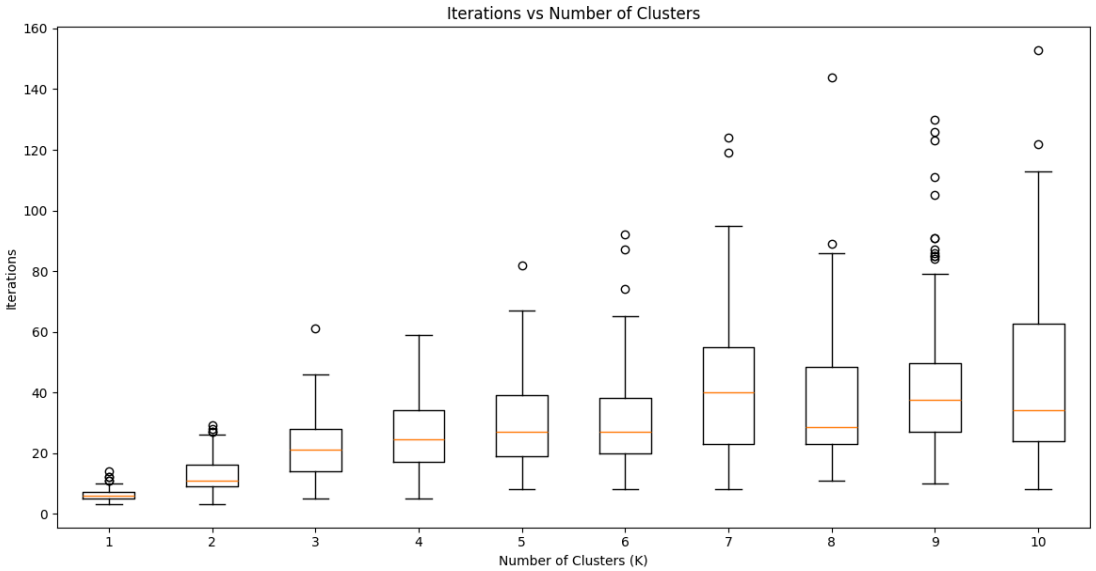
Inicialització dels centroides del Kmeans:

Dins de la funció *_init_centroids* hem afegit dos mètodes addicionals d'inicialització de centroides, a part del "first:

- "random": en aquest mètode, els centroides es seleccionen de manera aleatòria entre els punts de les dades d'entrada. Es trien aleatòriament 'K' índexs i els punts corresponents a aquests índexs es seleccionen com a centroides.
- "custom": selecciona els centroides inicials de manera equidistant al conjunt de dades self.X. Divideix el conjunt de dades en segments i escull punts d'aquests segments, assegurant-se de no repetir punts. Això es fa per garantir una distribució més uniforme dels centroides inicials a l'espai de les dades.



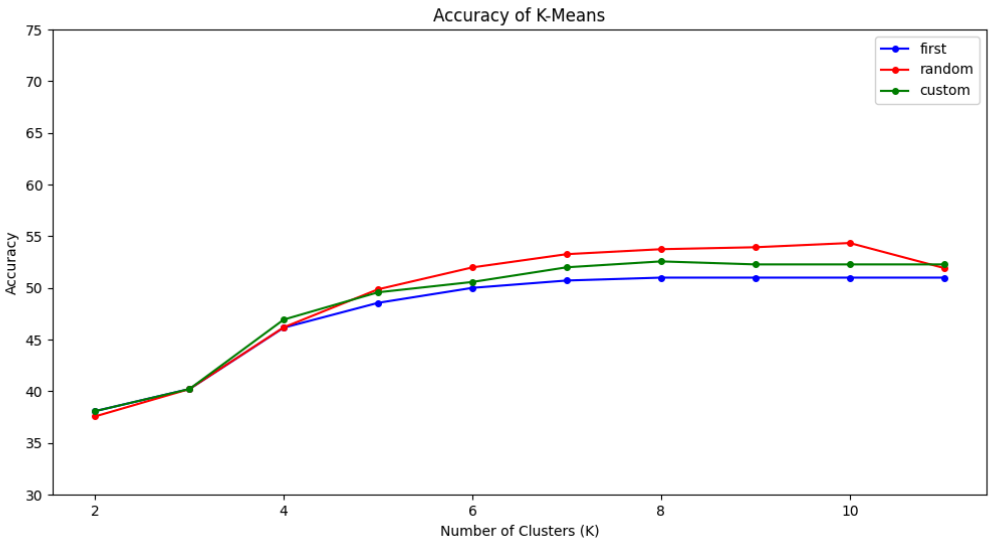
Boxplot mètode "random"



Boxplot mètode "custom"

Veiem que amb aquests dos mètodes el nombre d'iteracions és menor que amb el mètode "first" (representat a la pàgina 10). En el primer cas, la mediana s'estabilitza al voltant de les 30 iteracions i en el segon al voltant de les 40. També observem que els quartils són baixos respecte del total de dades, el que ens indica que el 50% central de casos requereixen menys iteracions i, per tant, l'algorisme convergeix més ràpidament. En el mètode "custom" observem més outliers per alguna K (per a K=9 per exemple), això ens pot dir que quan hi ha molts outliers molts valors es surten del conjunt de dades i, per tant, suposa errors en el mesurament i una major incertesa.

Per comprovar l'eficiència i veure si hi ha millores amb els nous mètodes d'inicialització implementats (random i costum), hem fet ús de la funció *get_color_accuracy*. Aquests han estat els resultats obtinguts:



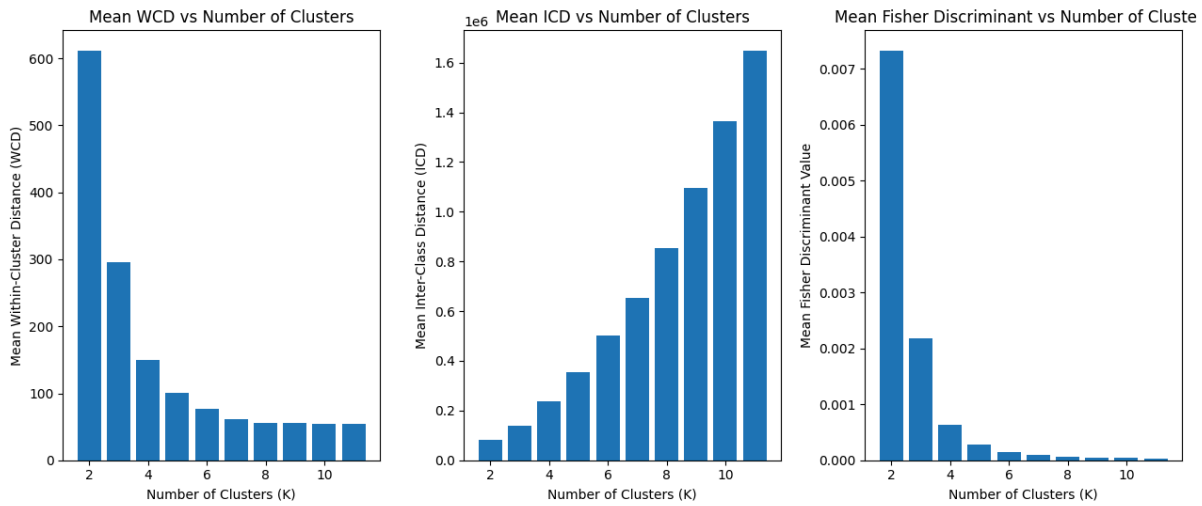
Comparació dels accuracy segon el mètode

Podem observar que a mesura que la k incrementa, l'accuracy del random i el costum es manté millor que el first. Això es pot deure a que, en el cas del random, l'aleatorietat augmenti la probabilitat que els centroides inicials estiguin dispersos per tot l'espai de dades. Pel que fa al nostre custom, podem dir que a l'hora de distribuir els centroides inicials de manera més uniforme, aquesta estratègia intenta garantir que els centroides inicials estiguin ben repartits a l'espai de dades. I finalment, el mètode first, si els primers punts no representen bé la distribució global de les dades, els centroides poden estar agrupats en una petita regió de l'espai de dades. Això pot portar a una convergència pobra, ja que els centroides inicials no cobreixen bé la variabilitat del conjunt de dades.

Una altra cosa a comentar és que sabem que a vegades el random pot no donar una bona inicialització dels centroides, ja que és aleatori, i això pot afectar a l'accuracy. En el cas k=11 podem observar que s'ha donat aquest succés.

Heurístiques per BestK:

Per a mesurar quina és la millor K hem implementat dues heurístiques més a part de la *WCD* (*Within Class Distance*): *ICD* i *FD* (*Inter Class Distance* i *Fisher Distance* respectivament).



Comparació dels K segon els 'mean' de 'WCD', 'ICD' i 'Fisher'

S'observa que la distància intra-classe i el discriminant de Fisher disminueixen ràpidament, mentre que la distància inter-classe augmenta, tal com es podria esperar. Notem que, intuïtivament, els valors òptims per a K es troben al voltant de 3 i 8. A més, veiem que la distància de Fisher determina el valor de K de manera més eficient, ja que combina els dos paràmetres més importants per agrupar els clústers: la distància inter-classe i la distància intra-classe.

Trobar una millor K:

Avaluem el rendiment del model utilitzant les tres mètriques implementades *WCD*, *ICD* i *FD*. Per fer-ho, ens podem fixar en el temps necessita l'algoritme amb cada heurística o en el nombre d'iteracions. A més a més, ens fixem si l'accuracy millora:

El temps i l'accuracy s'han fet amb 100 imatges de test, tot amb k_max=11.

Implementació	**Temps (segons)**	**Accuracy**
kmeans 100 data (first, WCD)	0.10395174503326415	50.99 %
kmeans 100 data (first, ICD)	0.10145380496978759	58.07 %
kmeans 100 data (first, Fisher)	0.10300341367721558	38.07 %
kmeans 100 data (random, WCD)	0.07108399868011475	51.4 %
kmeans 100 data (random, ICD)	0.06391222476959228	58.07 %
kmeans 100 data (random, Fisher)	0.06734518766403198	38.07 %
kmeans 100 data (custom, WCD)	0.07213806867599487	52.27 %
kmeans 100 data (custom, ICD)	0.07698936939239502	58.07 %
kmeans 100 data (custom, Fisher)	0.07420124053955078	38.07 %

Dels resultats s'observa com kmeans que obté la millor accuracy és el kmeans amb l'heurística de ICD i el que obté el resultat de manera més ràpida és Kmeans amb la inicialització random i heurística ICD.

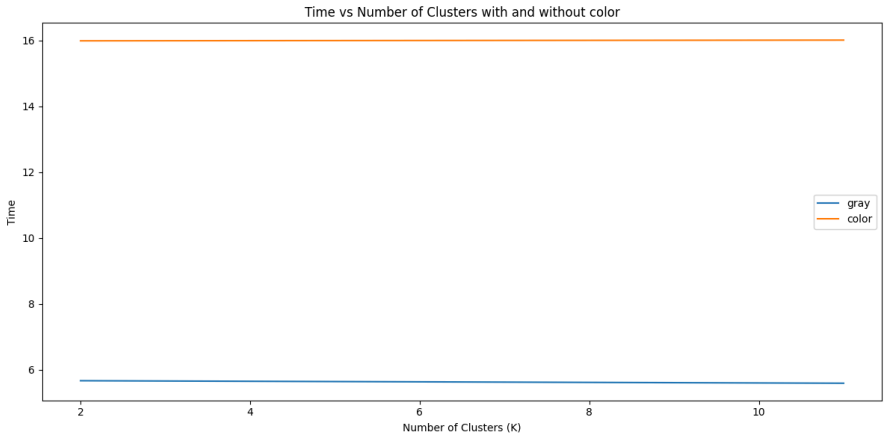
K-NN

Seguim utilitzant el dataset *train_data* per entrenar l'algoritme i el *test_data* per classficar-lo.

Obtenir una millor precisió per la forma (Mètode Grisos):

Amb l'objectiu de trobar la manera òptima de classificar per formes, hem vist que convertir les imatges que estan en format RGB a una representació més simple en escala de grisos és molt bona aplicació. Això ens permet analitzar les imatges focalitzant-nos més en les seves formes i característiques estructurals, ja que eliminem la informació sobre els colors que pot ser menys rellevant per a la nostra tasca d'anàlisi.

Podem observar com en passar les imatges a escala de grisos, el temps és d'execució del K-NN és molt més baix que si mantenim les imatges a color.



Gràphic de comparació dels K segons el temps amb color i en gris

Per comprovar si hi ha una millora a l'hora de classificar les imatges, hem comprovat quin és l'accuracy amb color i sense, i podem veure que hi ha un petit augment del percentatge (amb k=6):

- 90.95 % de class accuracy amb les imatges de color
- 91.4 % de class accuracy amb les imatges en blanc i negre

Aquesta diferència de percentatge pot ser deguda a que estem eliminant el color, una característica que és addicional i no rellevant per classificar la forma, donant lloc a un major contrast i nitidesa, cosa que fa que les imatges siguin més distingibles i més fàcils de classificar per l'algoritme.

Per tant, podem concloure que:

1. El mètode amb millor accuracy és el de grisos.
2. Hi ha una gran millora en el temps del K-NN amb grisos
3. Hi ha diferència en l'accuracy entre el rgb i el de grisos però és molt petita

Anàlisis de la millor k

Per validar el valor de la K i trobar el valor òptim, utilitzem la funció de *Get_shape_accuracy* pels diferents valors de k:

K	Accuracy	K	Accuracy
1	91.01 %	6	90.95 %
2	91.51 %	7	89.78 %
3	92.01 %	8	89.54 %
4	92.2 %	9	88.95 %
5	90.48 %	10	88.72 %
		11	88.13 %

Amb aquests resultats podem veure que tenim un accuracy major amb k = 4; encara que hi ha els valors anteriors com el 2 o el 3 que també són alts, la tendència que observem és que a partir de k=4 l'accuracy disminueix, per tant, deduïm que són valors menys òptims de k.

Conclusions finals

Hem arribat a les següents conclusions després d'aplicar les millores prèviament explicades als algoritmes i fer les seves anàlisis tant qualitatiu com quantitatiu:

Pel K-Means…

1. Hem pogut observar que la qualitat de les classes (clústers) depèn de com inicialitzem els centroides. Així, hem demostrat que és millor una distribució aleatòria o equidistant que no pas agafar el k primers.
2. Podem concloure que la millor heurística que podem utilitzares l'ICD perquè és, per exemple, amb el qual tenim un major accuracy en els colors de les imatges.
3. Dels mètodes implementats, el més lent és el kmeans que inicialitza els centroïdes amb l'opció first i l'heurística Fisher, té sentit que sigui aquesta perquè el seu error és més gran en utilitzar el WCD i ICD, per la qual cosa pot arrossegar els seus errors.
4. Dels mètodes implementats, el més eficient (ràpid i bons resultats) és amb inicialització first i amb l'heurística ICD, veiem que el accuracy és el mateix, però si és veritat que triga menys temps que si utilitzem altres (random o custom).

Pel K-NN…

1. Millor passar a grisos, ja que com hem vist a la gràfica, triga unes 8 vegades més si utilitzem colors.
2. També hem observat que l'accuracy millora una mica amb el canvi a grisos.
3. Els mètodes amb K = 4 donen una millor precisió que la resta, perquè el seu accuracy és major.