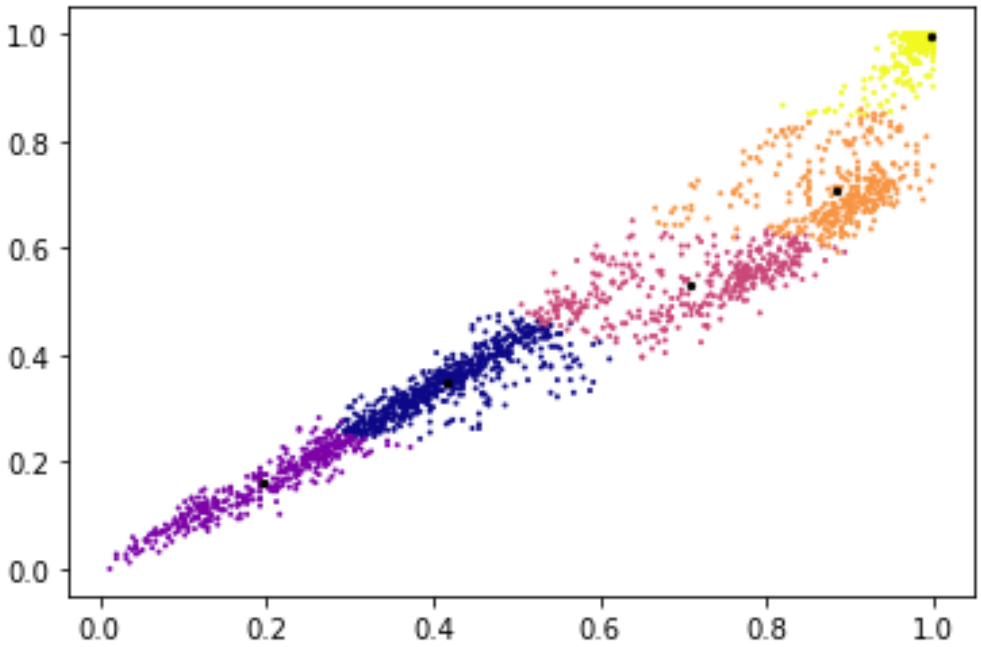
Etiquetador de Roba

Projecte Intel·ligència Artificial



Autors:

Biel González Garriga 1551813

Cristina Soler Arenys 1603542

Pau Blasco Roca 1600959

Alberto Real Quereda 1544112

Maig 2022

Matemàtiques Computacionals i Anàlisi de Dades

Resum

Els mètodes de classificació ens permeten separar en diferents classes elements d'una base de dades en un espai de característiques. Aquest projecte es centra a usar diversos mètodes de classificació en una base de dades de mostres de roba per catalogar-les segons el color, la forma o el tipus. A partir d'això s'han realitzat una sèrie de potencials millores i posteriorment una anàlisi d'aquestes per obtenir la fiabilitat i rapidesa. Posteriorment, s'ha estudiat el rendiment produït per les diferents millores per comparar els canvis duts a terme entre els mètodes originals i els mètodes que han estat modificats. Així, s'ha aconseguit avaluar si les modificacions produeixen millores significatives.

Índex

1 Introducció

Aquest projecte consisteix a resoldre un problema d'etiquetatge d'imatges, combinant els mètodes de KMeans per a l'etiquetatge del color i KNN per a l'etiquetatge de la forma. Donat un conjunt d'imatges d'un catàleg de roba desenvoluparem aquests algorismes d'aprenentatge no supervisat i supervisat respectivament, els quals permetran al classificador aprendre a etiquetar automàticament les imatges entrades per tipus de peça i per color. Per realitzar les proves, se'ns ha facilitat un dataset reduït.

Un cop programats l'estructura bàsica dels algorismes s'han realitzat un conjunt de proves per detectar les febleses d'aquests, i poder implementar millores de velocitat d'execució i recerca de les solucions, de precisió i de fiabilitat en el retorn d'imatges buscades.

El sistema que s'ha desenvolupat s'executarà sobre una imatge donada i ens retornarà una etiqueta que conté el tipus de peça de roba o el color d'aquesta. També podrem demanar que ens retorni una o més imatges amb una forma i color concret. Això es farà mitjançant l'ús de consultes simulant així un buscador real d'una botiga de roba.

1.1 Kmeans

El Kmeans[1] és un dels algoritmes d'Aprenentatge Automàtic no supervisats més usats per la seva facilitat d'implementació i per la seva simplicitat. L'objectiu base de l'algoritme és agrupar les observacions similars per descobrir patrons que a simple vista desconeixem. Per aplicar-ho emprarem un nombre fixat (K) que representarà el nombre de clústers en el dataset. Aquest mateix nombre K representa, a la vegada, el nombre de centroides que volem trobar. L'algorisme original intenta minimitzar la suma de les distàncies entre els punts i el centroide al qual pertanyen, mesura coneguda com a WCD[2].

1.2 KNN

El KNN[3] és un algoritme supervisat d'Aprenentatge Automàtic, i pot ser usat tant per classificació de dades com per regressió. En la implementació de classificació, serveix essencialment per classificar valors buscant els punts de dades més similars per proximitat, és a dir, que classifica el punt basant-se en la majoria de dades que l'envolten. La K en aquest algoritme determina la quantitat de punts veïns que tenim en compte a les proximitats dels N punts que ja es coneixen per endavant.

[1] MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability 1. University of California Press. pp. 281-297

[2] Veure apartat 6.2.3, Estadístics

[3] Altman, N. S. «An introduction to kernel and nearest-neighbor nonparametric regression». The American Statistician, 46, 3, 1992, pàg. 175–185. DOI: 10.1080/00031305.1992.10475879

2 Objectius

Objectius generals

- Realitzar un agent que ens permeti fer un etiquetatge automàtic de les imatges entrades.
- Realitzar cerques intel·ligents en llenguatge natural, simulant una botiga en línia.
- Poder realitzar consultes en un dataset flexible, sempre que aquest estigui sota unes certes restriccions de color i forma.
- Millorar els algorismes d'etiquetatge per aconseguir resultats més satisfactoris en un temps menor per facilitar així les cerques.

Objectius del Kmeans

- Trobar els agrupaments més significatius d'una mostra de punts en un espai 3D (píxels en format RGB).
- Trobar una estimació correcta i ràpida de la millor K usant diferents classificadors.

Objectius del KNN

- Facilitar la detecció de la peça de roba reduint-ne les dimensions, eliminant-ne el color i deixant-la en tonalitat de grisos.
- Optimitzar la classificació de les peces de roba redimensionant la imatge per eliminar elements innecessaris.

3 Fitxers necessaris per l'execució

Els fitxers que necessitem tenir per a poder executar el classificador:

- **KNN.py:** Conté les funcions necessàries per executar l'algorisme de classificació KNN, que ens permet obtenir les classes.
- **KMeans.py:** Conté les funcions necessàries per executar l'algorisme de classificació KMeans, que ens permet aconseguir els colors.
- **utils_data.py:** Conté les funcions necessàries per carregar, processar i visualitzar les dades.
- **utils_testing.py:** Conté les funcions per realitzar diferents tests sobre els algorismes i les millores.

A més, és necessari que les carpetes *train* i *test* estiguin al mateix directori que els fitxers de codi.

Per a permetre una comprovació ràpida de la llibreria, s'ha preparat un fitxer de codi extra, que ens permet visualitzar els resultats ràpidament:

- **Menu_query.py:** Aquest menú permet cridar a les tres funcions d'anàlisi qualitativa (*retrieval* per color, forma i color+forma). D'aquesta manera, amb una interfície còmoda per a l'usuari, es poden visualitzar els resultats de les *queries*.

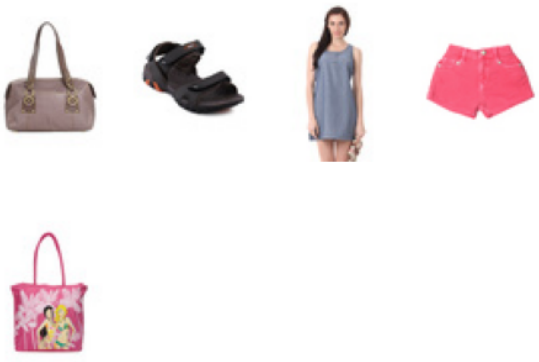
4 Funcions d'anàlisi qualitatiu

4.1 Query color

Funció que rep una sèrie d'imatges etiquetades, una pregunta (query) concreta, i un nombre natural N. Ens retorna fins a N imatges amb les etiquetes corresponents a la pregunta.

- **Entrada:** Funció que rep per entrada una llista d'imatges ja etiquetades. També li hem entrat el valor de la pregunta, és a dir el color a buscar, el nombre d'imatges a obtenir i finalment un paràmetre de visualització o no, necessari per a la funció de query general.
- **Funcionament:** Recorrem la llista de colors buscant a cada llista independent si hi ha el color demanat per l'usuari. Si és així guardem l'índex corresponent a la imatge en una llista. En finalitzar tota la llista, fem un shuffle de la llista d'índexs per treure cada vegada imatges diferents. Per poder visualitzar les imatges necessitem crear un petit dataset amb les imatges dels índexs, i amb la funció ja proporcionada "visualize retrival" podem visualitzar les n imatges demanades.
- **Sortida:** La funció ens retorna el valor dels índexs que compleixen la condició de la pregunta. La sortida és necessària per a la funció de query general. A més, es mostra a l'usuari un conjunt d'imatges amb els criteris demanants.

Exemple de funcionament: Per visualitzar que ens retornarà el query de Color hem fet diverses crides com podem veure a continuació:



(a) Query Color: Pink, nº:5l

(b) Query Color: Green, nº:3

Figura 1: Query Color

4.2 Query class

Funció que rep una sèrie d'imatges etiquetades, una pregunta (query) concreta, i un nombre natural N. Ens retorna fins a N imatges amb les etiquetes corresponents a la pregunta.

- **Entrada:**Funció que rep per entrada una llista d'imatges ja etiquetades. També li hem entrat el valor de la pregunta, és a dir la classe a buscar, el nombre d'imatges a obtenir i finalment un paràmetre de visualització o no, necessari per a la funció de query general.
- **Funcionament:** Recorrem la llista de classes buscant si la imatge pertany a la classe entrada. Si és així guardem l'índex corresponent a la imatge en una llista. En finalitzar tota la llista, fem un shuffle de la llista d'índex per treure cada vegada imatges diferents. Per poder visualitzar les imatges necessitem crear un petit dataset amb les imatges dels índexs, i amb la funció ja proporcionada "visualize retrival" podem visualitzar les n imatges demanades.
- **Sortida:** La funció ens retorna el valor dels índexs que compleixen la condició de la pregunta. La sortida és necessària per a la funció de query general. A més, es mostra a l'usuari un conjunt d'imatges amb els criteris demanants.

Exemple de funcionament: Per visualitzar que ens retornarà el query de Classes hem fet diverses crides com podem veure a continuació:

(a) Query class: Flip Flops, nº: 3 (b) Query class: Dresses, nº: 10

Figura 2: Query Classes

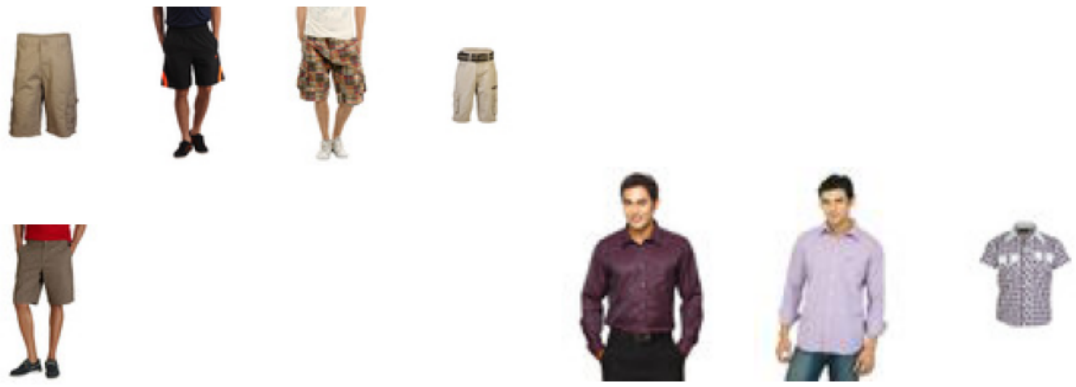
Filtre d'error: A les funcions de Query Color i Query Classes s'ha comprovat que les n imatges demanades no superi a les imatges trobades que compleixen la condició, si és així és mostra un missatge d'error i se surt de la funció. També en el menú de les funcions s'ha programat una funció que filtra les paraules entrades, per deixar-les amb el format necessari pel programa.

4.3 Query general

Funció que rep una sèrie d'imatges etiquetades, una pregunta (query) concreta, i un nombre natural N. Ens retorna fins a N imatges amb les etiquetes corresponents a la pregunta.

- **Entrada:** Funció que rep per entrada una llista d'imatges ja etiquetades. També li hem entrat els valors de la pregunta, és a dir la classe a buscar i el color, en format llista i el nombre d'imatges a obtenir.
- **Funcionament:** S'ha considerat que per filtrar millor les imatges primer es mirarà la forma d'aquestes, és a dir, cridarem al query de classes, la qual ens retornarà els índexs de les imatges que ho compleixen. A partir d'aquest generem dos subdatasets, un del dataset de les imatges i l'altre del dataset dels colors. Posteriorment, cridem al query de colors el qual se li ha proporcionat aquests subdatasets ja filtrats. Query colors ens retornarà la llista d'índex que compleixen les dues condicions. Generem un dataset amb aquests índexs i cridem la funció "visualize retrival" la qual ens mostrarà les n imatges on les dues etiquetes coincideixen.
- **Sortida:** La funció no retorna res, es mostra a l'usuari un conjunt d'imatges amb els criteris demanants.

Exemple de funcionament: Per visualitzar que ens retornarà el query Global hem fet diverses crides com podem veure a continuació:



(a) Query class: Shorts, color: Brown nº: 5 (b) Query class: Shirts, color: Purple nº: 3

Figura 3: Query Global

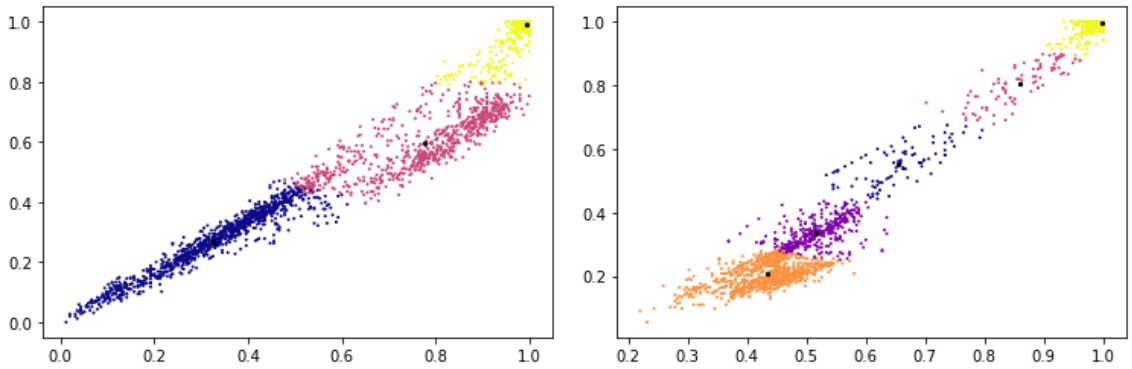
Per fer una crida als queries s'ha programat un menú de consola perquè la crida a les funcions sigui més fàcil per l'usuari.

4.4 Visualtzar Kmeans

Funció que ens mostrarà la classificació en kmeans d'una imatge concreta del dataset.

- **Entrada:** Ens caldrà proporcionar a la funció una imatge del dataset, a la qual li voldrem aplicar el Kmeans, i el nombre de clústers amb els quals començarem.
- **Funcionament:** Primerament, escalarem la imatge a un nombre entre zero i u, en comptes de zero i dos-cents cinquanta-cinc, cridarem al Kmeans i l'aplicarem sobre la imatge nova i el nombre de clústers entrat. Per poder visualitzar el plot necessitarem inicialitzar els centroides, obtenir els colors i fer el fit del Kmeans. Finalment, amb la funció *Plot Scatter* assolirem un gràfic amb els diferents colors separats i els seus respectius centroides.
- **Sortida:** La funció no retorna res com a tal, però sí que ens mostra un gràfic a la pestanya de gràfics.

Exemple de funcionament:



(a) Image 19, n_clusters:3 (b) Image 45, n_clusters:5

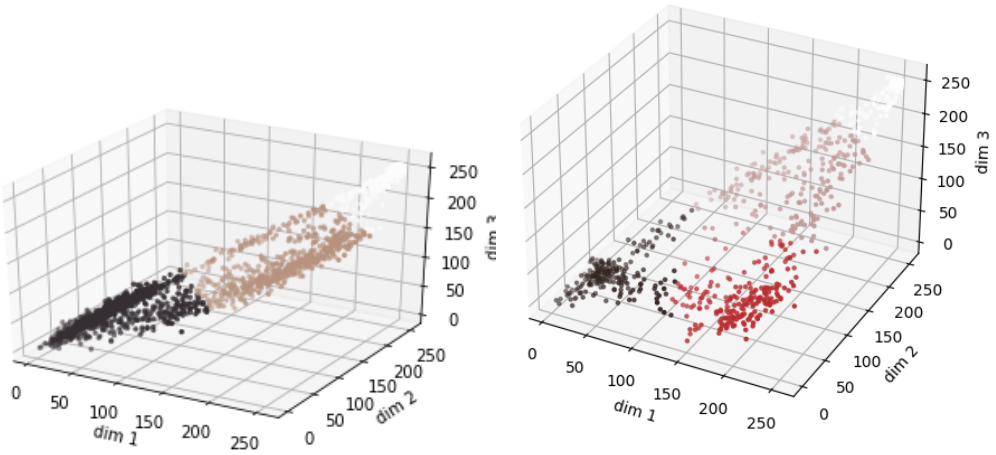
Figura 4: Visualitzar Kmeans

4.5 Plot3D

Funció que ens mostrarà la classificació en kmeans d'una imatge concreta en un plot en 3D.

- **Entrada:** Ens caldrà proporcionar a la funció una imatge del dataset, a la qual li voldrem aplicar el Kmeans, i el nombre de clústers que aplicarem al Kmeans.
- **Funcionament:** Apliquem Kmeans a la imatge amb el nombre de clústers entrat, inicialitzem centroides, labels i fem el fit del Kmeans. Per poder rebre el gràfic apliquem la funció *Plot3DCloud* ja implementada la qual li entrem el Kmeans calculat, i variables constants imposades per la funció.
- **Sortida:** La funció no retorna res, però ens mostrarà un plot 3D amb les diferents classes.

Exemple de funcionament:



(a) Image 20, n_clusters:3 (b) Una de les imatges del dataset

Figura 5: Plot3d

5 Funcions d'anàlisi quantitatiu

5.1 KNN

5.1.1 Class accuracy

Funció que calcula l'accuracy d'un conjunt de prediccions de classe, comparant-les amb les seves respectives ground_truth.

- **Entrada:** La funció rep el ground_truth, una llista de longitud l menor que zero que conté el ground_truth i també el predicted_labels, la qual conté les prediccions de classes.
- **Funcionament:** Per evitar errors de funcionament al principi comprovem que les longituds de la llista ground_truth i de predicted_labels sigui igual, si no és així mostrarà un missatge d'error i sortirà. Si les longituds són iguals, comparem cada element del ground_truth amb la de prediccions i si són diferents sumem u a un comptador d'errors. Finalment, calculem:

  $$acc = \frac{(1-e)}{l \cdot 100}$$

  on e és el comptador d'errors, i l la longitud, calcularem l'accuracy.
- **Sortida:** La funció ens retorna el valor, en tant per cent, de l'encert de les prediccions

5.2 Kmeans

5.2.1 Color accuracy

Funció que calcula l'accuracy d'un conjunt de prediccions de color, comparant-les amb les seves respectives ground_truth.

- **Entrada:** La funció rep el ground_truth, una llista de longitud l menor que zero que conté el ground_truth i també el predicted_labels, la qual conté les prediccions de colors.
- **Funcionament:** Per evitar errors de funcionament al principi comprovem que les longituds de la llista ground_truth i de predicted_labels sigui igual, si no és així mostrarà un missatge d'error i sortirà. Recorrem les dues llistes a la vegada. Primer calculem el coeficient *diff* on mirarem la longitud de la llista entre la resta dels elements únics de cada llista. Seguidament, calcularem el màxim de longitud entre la subllista de prediccions que estem mirant o la subllista del ground_truth. Finalment per calcular l'accuracy aplicarem la següent fórmula:

  $$a = (\frac{1-diff}{maxlen})100$$

- **Sortida:** La funció ens retorna el valor, en tant per cent, de l'encert de les prediccions

5.2.2 Find K distribution

Funció que estudia la distribució de la best_k per a un conjunt d'imatges. Ens permet entendre com estan distribuïts el nombre de centroides per imatge en un grup d'imatges.

- **Entrada:** A la funció li proporcionarem els labels predits i el nombre màxim de centroides del conjunt.
- **Funcionament:** Generarem un histograma de zero fins al nombre de K màxima entrada, mirarem cada label de la predicció i sumarem u a la posició que ocupa en l'histograma, calculat amb la longitud total de la subllista de labels que estem avaluant excepte un. Finalment, calculem la distribució, la qual divideix cada element de l'histograma per la longitud de la llista de predicted_labels.
- **Sortida:** Ens retornarà la distribució en tant per u de la best_k.

5.2.3 Test Convergence Speed

Funció que calcula el temps mitjà de convergència del KMeans per a un conjunt d'imatges, amb un mètode d'inicialització concret.

- **Entrada:** Necessitarem entrar el dataset de les imatges en color, el nombre de centroides amb el que treballarà el Kmeans i també haurem d'indicar el mode d'inicialització entre 4 opcions: first, random, custom o custom 2.
- **Funcionament:** Per aquesta funció necessitarem el mòdul de temps de python, inicialitzarem el rellotge al principi de la funció i posteriorment aplicarem el Kmeans a cada imatge del dataset, amb els centroides i el mode d'inicialització indicats. En finalitzar parem el rellotge i calculem el temps mitjà.
- **Sortida:** La funció ens retornarà el temps mitjà de convergència de les imatges.

6 Millores als mètodes de classificació

6.1 KNN

Pel que fa al KNN, al tractar-se d'un entrenament a partir d'altres imatges, s'ha pogut fer servir el mètode de distribució de dades K-Fold[4] per a estudiar com diferents proporcions d'entrenament i de testeig afectaven als resultats.

6.1.1 Reducció d'imatge

Procediment que redueix la mida d'un conjunt d'imatges per així reduir les dades a processar.

- **Entrada**: Proporcionem una lista d'imatges que conté totes aquelles imatges a les quals volem reduir la mida.
- **Funcionament**: Es processa imatge per imatge i s'agafen una quantitat reduida i continua de columnes i files d'aquesta per així reduir la mida de la imatge.
- **Sortida**: Retorna la llista d'imatges de l'inici, però reduïdes.

6.1.2 RGB a HSL

Funció que transforma les imatges de RGB a HSL (blanc i negre).

- **Entrada**: Dataset amb les imatges en RGB.
- **Funcionament**: Transforma les tres components en un unic valor hue (model HSL) i, per tant, es redueix la dimensio de les imatges.
- **Sortida**: Dataset amb les imatges en blanc i negre.

6.1.3 Reducció d'imatge i HSL

Procediment on s'apliquen les dues idees anteriors a la vegada per obtenir imatges en blanc i negre i de mida reduïda.

[4] James, G., Witten, D., Hastie, T. and Tibshirani, R., n.d. An introduction to statistical learning. p.181.

6.1.4 Resultats:

Gràcies a les gràfiques podem observar que de totes les idees proposades i implementades la millor és només aplicar la reducció d'imatge. Tot i que l'accuracy d'aquesta sigui menor a fer el canvi de RGB a HSL, l'augment de l'error no és molt significatiu. A més, la reducció del temps d'execució és, per a la reducció de les imatges, molt major que la del HSL. La combinació de les dues millores té un temps d'execució quasi idèntic al de la reducció d'imatge, però perd quasi un 1% d'encert. Per això, hem considerat que combinar els dos mètodes no és òptim.

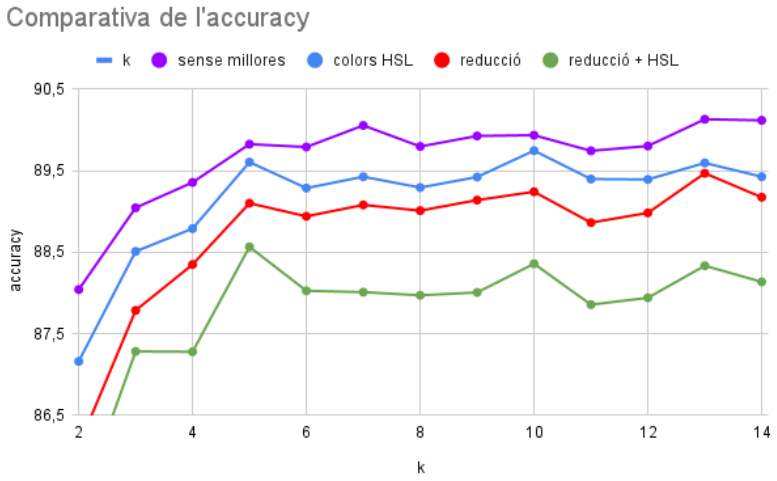


Figura 6: Comparativa de l'accuracy

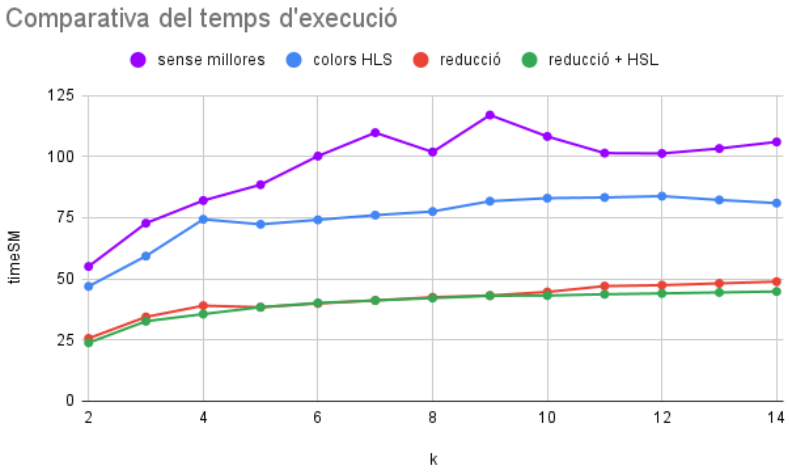


Figura 7: Comparativa del temps d'execució

A més, a partir d'aquests gràfics hem trobat que la K òptima per al K-fold, per a la nostra base de dades, és 5. Així doncs, caldria entrenar el dataset amb el 80% de les dades, i fer les proves amb el 20% restant. A partir d'aquest punt, l'encert pràcticament no millora.

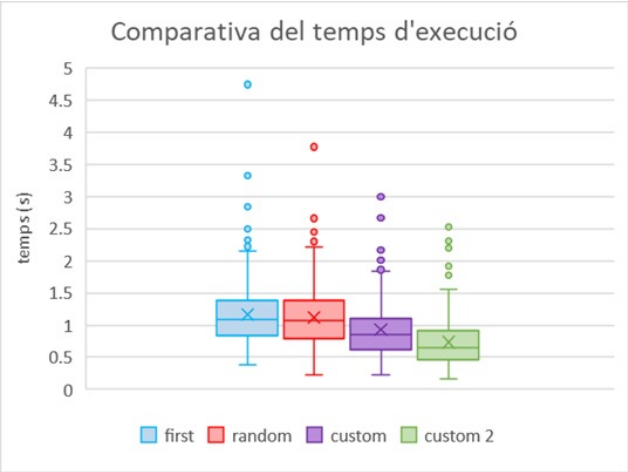
6.2 KMeans

Per a estudiar les millores a KMeans d'una forma eficient, s'ha mostrejat la base de dades repetides vegades i s'han controlat en tot moment els errors i les desviacions estàndard, per a assegurar-nos que els resultats obtinguts eren fiables.

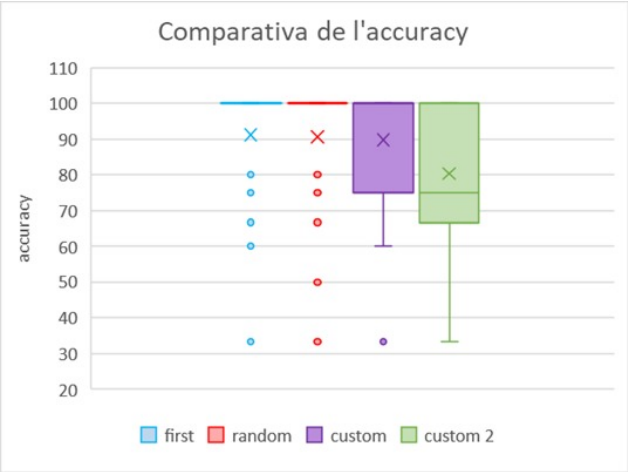
6.2.1 Inicialització de centroides

Creació de diferents mètodes d'inicialització dels punts per mirar si hi ha cap inicialització que millori la velocitat de convergència de l'algoritme.

- **Entrada**: No rep res per argument
- **Funcionament**: Inicialitza els centroides de maneres diferents segons l'opció seleccionada a `Kmeans.options['km_init']` hi ha disponibles tres més a part de la bàsica:
  - **Random**: Inicialització aleatòria dels punts.
  - **Custom**: Inicialització dels punts seguint una recta.
  - **Custom 2**: Inicialització dels punts seguint una el·lipse.
- **Sortida**: No retorna cap valor
- **Resultats**: A partir dels grafics podem deduir que la millor inicialització es la custom. Aquesta redueix el temps i l'accuracy tot i ser una mica variable la mediana es manté semblant al first i al random.



(a) Comparativa dels temps d'execució

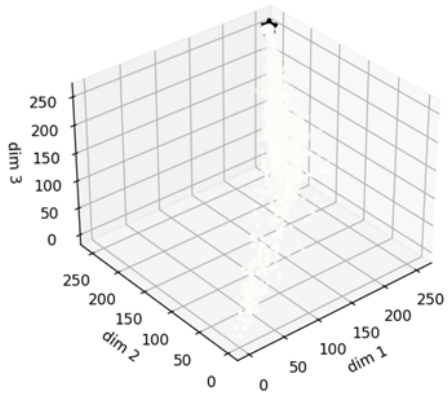


(b) Comparativa de l'accuracy

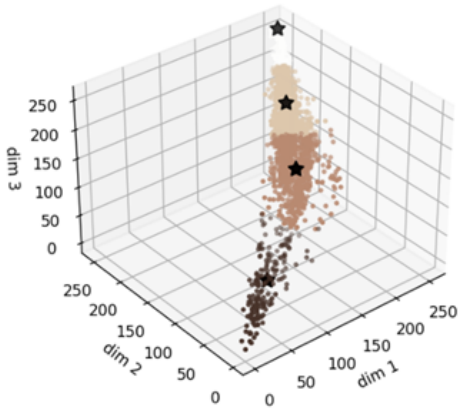
Malgrat que algunes medianes estiguin al 100%, la creu, que marca la mitjana, ens serveix com a punt de referència més que suficient per a comparar els diferents mètodes.

Gràcies a la funció plot3D podem veure com queden les diferents inicialitzacions dels centroides en un espai tridimensional. Hem representat, per a cada tipus d'inicialització, dues imatges abans de cridar a la funció *fit()*, i els seus respectius resultats.

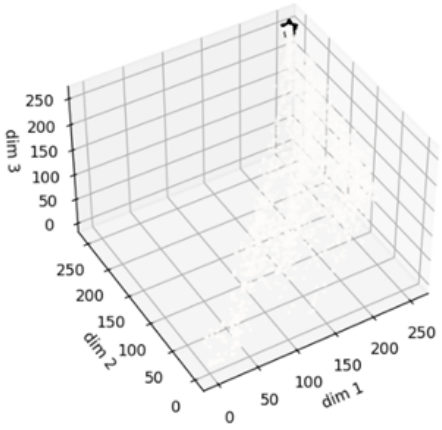
First:



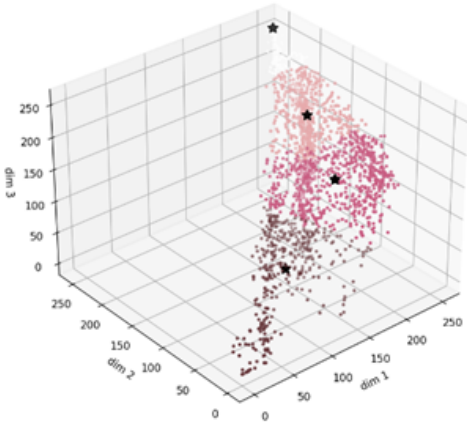
(a) Image 89, n_clusters:4, abans fit



(b) Image 89, n_clusters:4, després fit



(c) Image 687, n_clusters:4, després fit

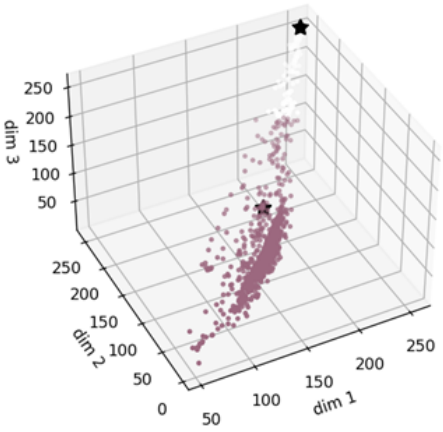


(d) Image 687, n_clusters:4, després fit

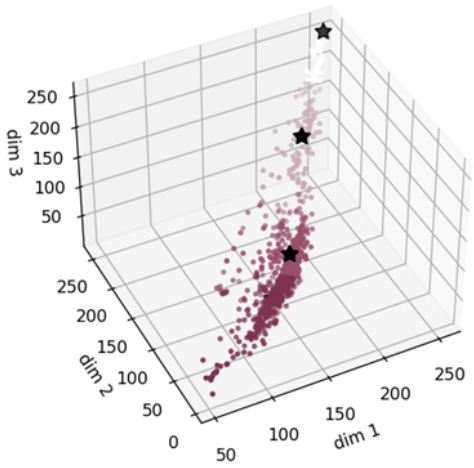
Figura 9: Exemples amb inicialització first

En el cas del *first* tenim un problema evident: el color blanc. Al ser el fons de les imatges de color blanc, els centroides queden aglomerats en una punta. Això fa que el procés de convergència, com hem vist a la figura (a) de la secció 6.2, sigui major.

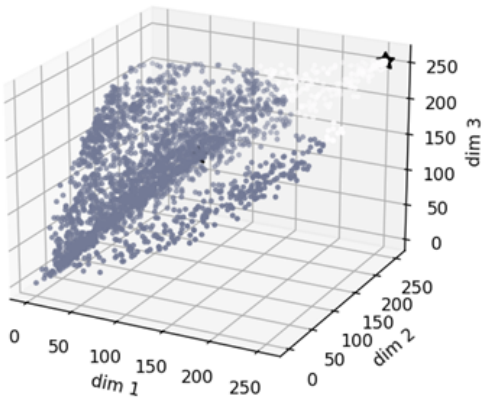
Random:



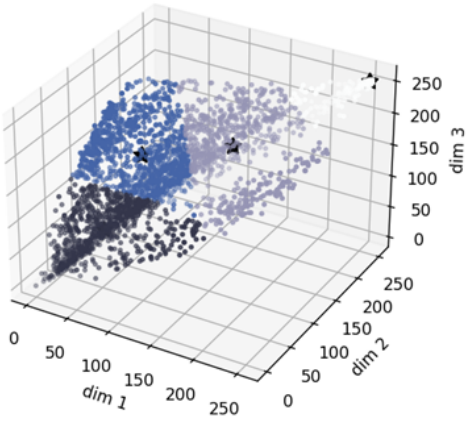
(a) Image 1626, n_clusters:4, abans fit



(b) Image 1626, n_clusters:4, després fit



(c) Image 2620, n_clusters:4, abans fit

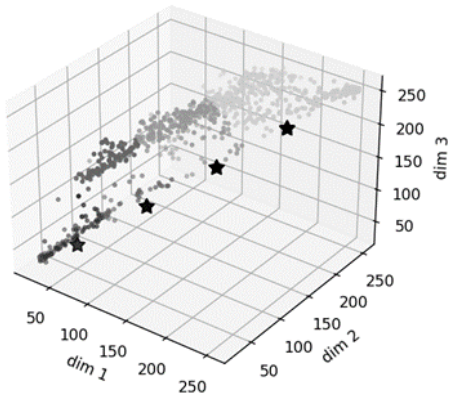


(d) Image 2620, n_clusters:4, després fit

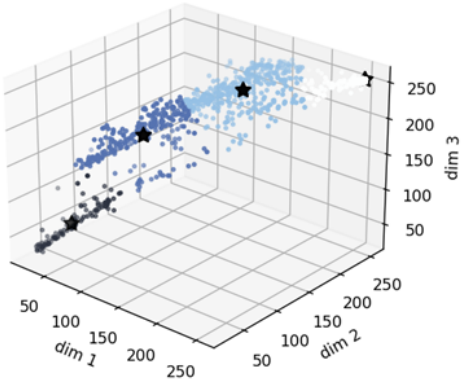
Figura 10: Exemples amb inicialització random

Pel que fa al mètode *random*, ens dona resultats millors que el first en la majoria dels casos. Així i tot, inicialitzacions desfavorables (en les que els punts queden "mal repartits") fan pujar considerablement el temps de convergència.

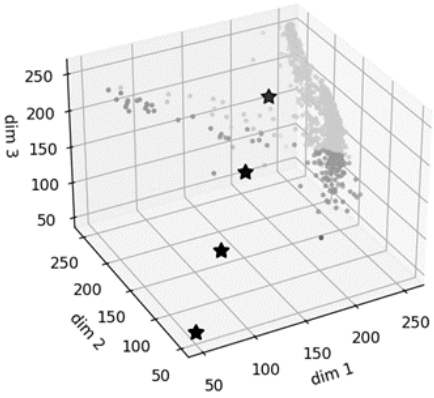
Custom:



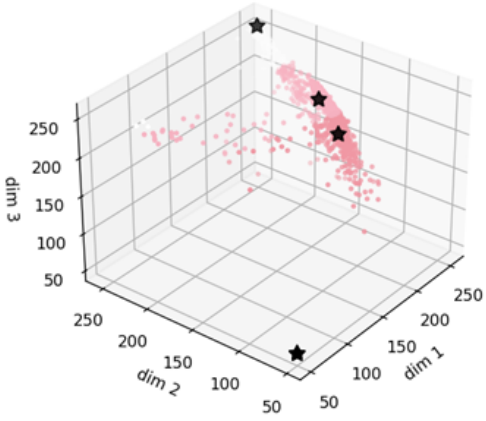
(a) Image 211, n_clusters:4, abans fit



(b) Image 211, n_clusters:4, després fit



(c) Image 2255, n_clusters:4, abans fit

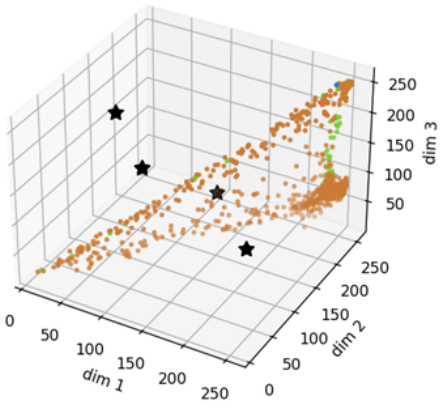


(d) Image 2255, n_clusters:4, després fit

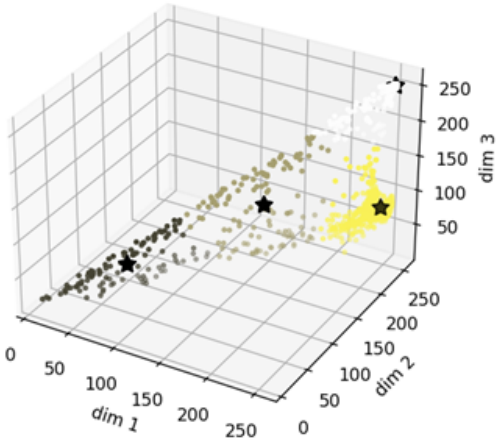
Figura 11: Exemples amb inicialització custom

El mètode *custom* és el que ens dona millors resultats pel que fa a la relació encert-temps. A més, veiem clarament com, en molts casos, els píxels ja es troben prop la recta (t,t,t) amb $t \in [0, 255]$; cosa que porta a una convergència més ràpida. Cal anar amb compte, ja que aquesta distribució dels píxels a l'espai RGB es podria deure al tipus de compressió de les imatges (en el nostre cas, .jpg).

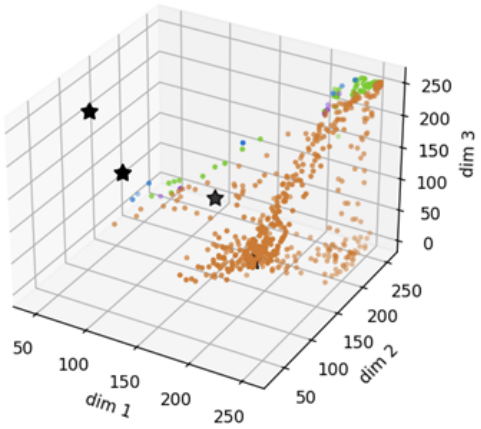
Custom 2:



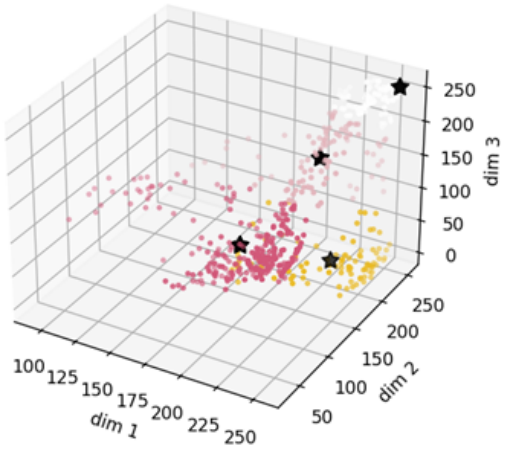
(a) Image 1086, n_clusters:4, abans fit



(b) Image 1086, n_clusters:4, després fit



(c) Image 1284, n_clusters:4, abans fit



(d) Image 1284, n_clusters:4, després fit

Figura 12: Exemples amb inicialització custom2

El mètode Custom2, tot i tenir una convergència més ràpida, condueix a resultats considerablement pitjors (quasi un 10% menys d'encert) que els altres tres mètodes. Això pot ser causat per diferents factors, alguns d'ells el tipus de compressió de la imatge o l'angle de desfasament amb el qual estan repartits els punts. Evidentment, també podria ser que el mètode simplement no fos òptim per a imatges RGB.

6.2.2 Reducció d'imatge

Procediment que redueix la mida d'un conjunt d'imatges per així reduir les dades a processar.

- **Entrada**: Llista d'imatges.
- **Funcionament**: S'agafa imatge per imatge i s'extreu una quantitat reduïda i continua de columnes i files, aquesta serà la nova imatge i serà de dimensions menors que l'anterior.
- **Sortida**: Llista d'imatges reduïdes.
- **Resultats**: Veiem que, de les diferents reduccions d'imatge, la que dona millors resultats d'accuracy i de temps és la reducció del trenta-nou per cent. Podem observar que aquesta manté una accuracy molt propera al del ground truth (fins i tot superant-la) i redueix molt el temps d'execució de l'algoritme. La resta, tot i reduir el temps, també redueixen molt l'accuracy i, per tant, ja no surt tant a compte fer servir aquestes reduccions.

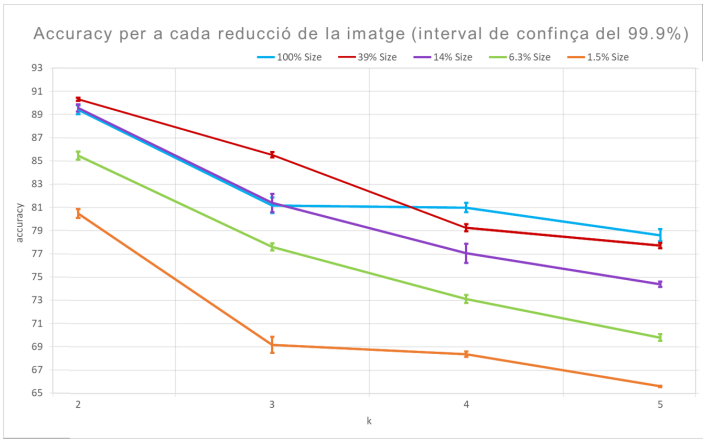


Figura 13: Accuracy amb les reduccions

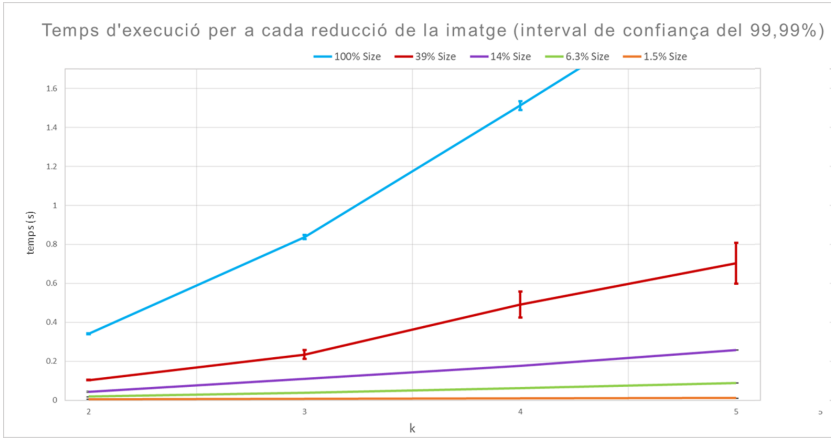


Figura 14: Temps d'execució amb les reduccions

6.2.3 Estadístics

WCD És una mesura de la distància intraclass.

- **Entrada**: No rep res per argument
- **Funcionament**: Calcula la distància entre tots els punts de la classe amb el centroide corresponent amb la fórmula:

$$WCD = \frac{1}{N} \sum_{x \in X} distancia(x, C_x)^2$$

  Els valors més petits indiquen una millor classificació.

- **Sortida**: No retorna cap valor

ECD És una mesura de la distància interclass.

- **Entrada**: No rep res per argument
- **Funcionament**: Calcula la distància entre tots els centroides existents en aquell moment a partir de la fórmula:

$$EDC = \frac{\sum_{i=0}^{k} d(i)}{\sum_{k-1}^{i=0} i}$$

  on $d(i)$ és la distància entre dos centroides. Els valors més grans són indicadors d'una millor classificació.

- **Sortida**: No retorna cap valor

WCD i ECD Mesura que combina els valors del WCD i ECD

- **Entrada**: No rep res per argument
- **Funcionament**: Retorna el quocient dels WCD i ECD seguint la formula:
  $\frac{\sum WCD}{EDC}$ Els valors més petits indiquen una millor classificació.
- **Sortida**: No retorna cap valor

WCD i ECD v2 Mesura que combina els valors del WCD i ECD ponderant-los.

- **Entrada**: No rep res per argument
- **Funcionament**: Retorna el quocient ponderat dels WCD i ECD seguint la formula:

$$\frac{\sum WCD}{EDC \cdot K^2 \cdot \sqrt{K}}$$

  on K és nombre de clústers. Els valors més petits indiquen una millor classificació.

- **Sortida**: No retorna cap valor

Silhouette Mesura la coherència dels punts d'un clúster.

- **Entrada**: No rep res per argument
- **Funcionament**: Es calcula el silohuette de tots els punts per veure com és de semblant aquest punt a la resta del seu clúster, comparant-lo també amb els altres clústers. Es segueix la fórmula:

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

  on $b(i)$ és la distància del centroide a un punt d'un altre clúster i $a(i)$ és la distància mitja del centroide a un punt del seu propi clúster. Els valors es troben entre -1 i 1, sent els més propers a 1 indicadors d'una millor classificació.

- **Sortida**: No retorna cap valor

Resultats: Realitzem un estudi que ens permet de manera general veure com distribueix les fotos cada mètode, considerant que la distribució òptima és la del ground truth. Això és dut a terme estudiant la quantitat de clústers òptima de cada foto (*find_ best_ K*). Posteriorment, es realitza un histograma amb aquestes dades per així poder comparar tots els estadístics. A partir d'aquí veiem que el que ens dona resultats més propers al ground truth és l'ECD i, per tant, dels cinc que tenim serà el que escollirem com a millor. Així i tot, cal mencionar que el Silhouette ha produït prou bons resultats.

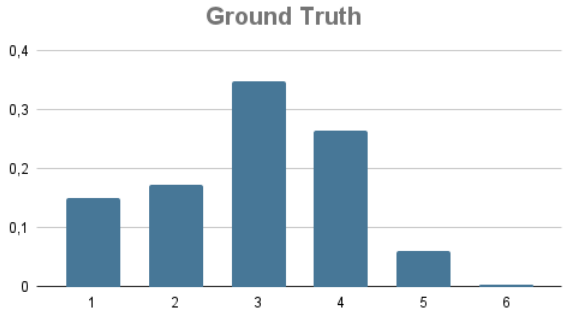


Figura 15: Distribució del GT

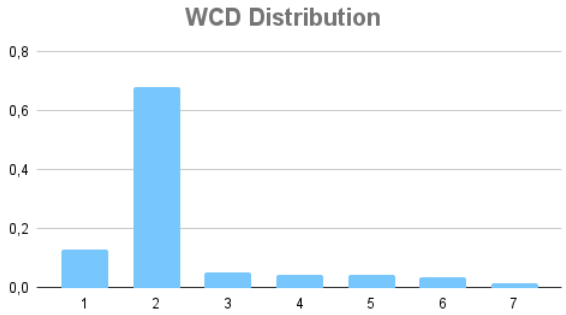


Figura 16: Distribució del WCD

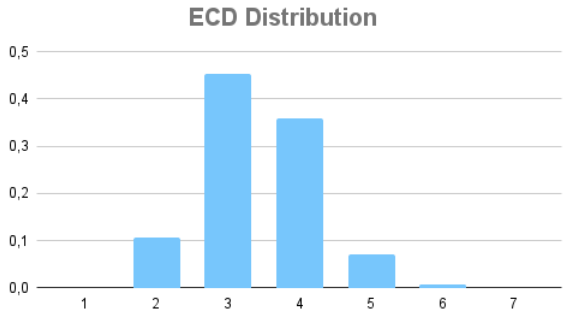


Figura 17: Distribució del ECD

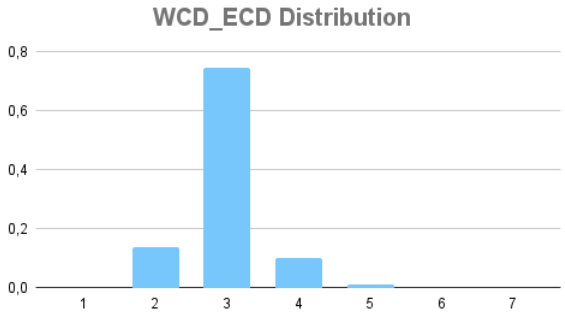


Figura 18: Distribució del WCD ECD

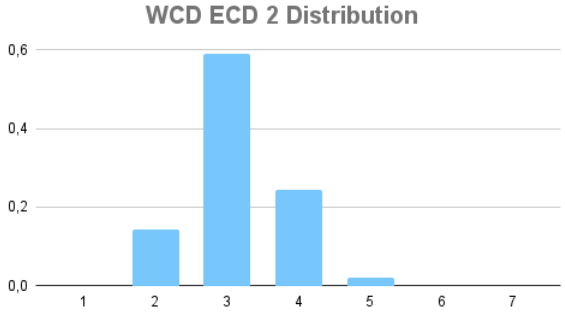


Figura 19: Distribució del WCD ECD 2

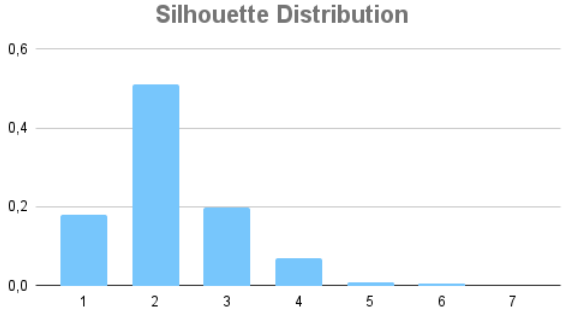


Figura 20: Distribució del Silhouette

7 Avaluació general dels resultats

Després de dur a terme diverses millores, i d'estudiar-ne la repercussió detingudament, hem arribat a les següents conclusions.

7.1 Millores a les classificacions individuals

Partint de la base de dades de la que disposem, les millores que han repercutit més positivament a l'eficàcia i velocitat de les classificacions *individuals* han estat la **reducció d'imatges**, tant per a KNN com per a KMeans (en concret al 39% de la mida original) i la inicialització **custom** pels clústers del KMeans.

7.1.1 Reducció d'imatges del 39%

La reducció d'imatges del 39% no deixa de ser un símil al raonament humà en veure imatges d'aquest estil. La vora blanca ocupa una quantitat molt considerable d'espai, que no és gens útil a l'hora d'identificar el tipus i el color de la peça de roba que hi apareix.

Aquesta reducció no només fa que el processament del color sigui 4 vegades més ràpid, sinó que millora l'encert fins a un 4% en imatges estudiades amb 2 i 3 clústers (figures 13 i 14, pàgina 20).

Pel que fa a la identificació de la classe, el processat és més del triple de ràpid, mentre que es perd menys d'un 1% d'encert. És una petita pèrdua, però a causa de la disminució de temps de processat i l'augment d'encert en la identificació del color, creiem que paga la pena aplicar-la.

7.1.2 Inicialització *custom* per al KMeans

Aquesta inicialització ha demostrat ser una millora considerable a la inicialització dels clústers per defecte, *first*. La inicialització *custom* ha aconseguit reduir el temps de processat en més d'un 20%, sense cap repercussió significativa en l'encert (figures (a) i (b), pàgina 16). Així doncs, considerem que aquest mètode és l'òptim per a tractar amb imatges similars a les de la base de dades, sempre que estiguin en format .jpg.

7.1.3 Millor mètode de parada pel *find_best_k*

Després de fer diverses comparacions entre les diferents distribucions que generaven els mètodes de parada (figures 15 a 20, pàgines 24 i 25), hem arribat a la conclusió que els mètodes que millor trobaven la k òptima són l'ECD i el Silhouette. D'entre els dos, l'ECD és el més simple de calcular, però no funciona per $k = 1$. Tot i així, creiem que és superior al Silhouette, almenys pel tipus de dades amb les que treballem.

7.2 Millores a les classificacions massives

En molts casos no estarem interessats a classificar poques imatges amb la màxima precisió possible, sinó que voldrem classificar una gran quantitat d'imatges en el menor temps possible (amb un encert prou bo, evidentment). En altres paraules, trobarem situacions en les quals no serà important classificar *poques* imatges *molt bé*, sinó que n'haurem de classificar *moltes* de forma *decent*. Per això, pel que fa a KMeans, hem estudiat els criteris de parada de *find_ best_ k()* i hem trobat quina és la **k** general (la que forçaríem a un gran nombre d'imatges) que ens assegura menys errors.

7.2.1 Comparacions de *find_ best_ k* i nombre de *k* clústers general

Després d'estudiar amb atenció les dades dels gràfics de les distribucions de les k (figures 15 a 20, pàgines 24 i 25) i les gràfiques de l'encert i el temps de processat per a cada reducció d'imatge (figures 13 i 14, pàgina 21) hem arribat a la conclusió que fer servir **3** centroides com a k general per a un estudi massiu de dades seria el que produiria millors resultats. Així i tot, en imatges en les que no haguéssim de descartar el fons (,png o fons transparent) podríem treballar amb **2** o fins i tot **1** centroide sense por a produir un error molt elevat.

7.3 Possibles consideracions i millores a la base de dades

7.3.1 Labels de color

En la gran majoria de casos, samarretes que no eren llises o que tenien algun dibuix feien baixar considerablement l'encert del classificador. Encara que es podien observar colors més "importants" que d'altres, els labels del dataset estaven ordenats per ordre alfabètic, sense donar gens d'importància a la prevalença dels colors a la imatge.

Demanar a un gran grup de persones que classifiqués les imatges per color ens solucionaria aquest problema, ja que veuríem quins colors han estat menys i més votats per cada imatge.

Aquest "percentatge" d'importància seria una millora crucial a l'hora d'avaluar el KMeans, ja que la sortida del mateix programa té un percentatge associat a cada color, indicant com de segur està que el color aparegui a la imatge.

7.3.2 Format de les imatges

Estem convençuts que un preprocessament més intens de les imatges (per part nostra o bé dut a terme prèviament), com per exemple, descartar el fons, hauria millorat considerablement l'accuracy. Així i tot, aquest preprocessament extra només ens seria útil si volguéssim més encert per part del classificador, ja que no reduiria gaire la velocitat de processament (ja retallem la imatge).