Dataanalys

Läs in diabetes.csv. Inspektera datan med head() och använd sns.pairplot för att visualisera distributionerna. Använd ”Outcome” från dataframen som hue-parameter. Tolka resultaten.

Gör PCA på datan och ta reda på hur stor del utav den förklarade variansen som de 5 första Principal Components ger. Plotta även hela rangen utav principal components med värden på y-axeln som den kumulerade summan utav förklarad varians.

Visualisera datasetet i 2D genom att göra PCA ner till 2 komponenter. Gör en scatter plot med färgkodning given av dina ursprungliga labels.

Använd umap och/eller t-SNE för att plotta i 2D och jämför med resultaten från PCA. Ser det bättre ut?

Standardisera/normalisera all data så att den blir centrerad runt en normaldistribution. (hint: Googla normalize dataframe). Passa på att inte ha label (outcome) med i datan när detta görs då det påverkar resultatet.

Plotta de normaliserade distributionerna igen med sns.pairplot så att du ser att värdena ligger i nya intervall. Skiljer formen på dina distributioner sig åt från de ursprungliga?

DBSCAN/Parametersökning

Utgå från ditt nu standardiserade dataset. Använd DBSCAN för att försöka klustra!

DBSCAN är flexibel i den mening att man inte behöver ansätta antalet kluster. Däremot är algoritmen väldigt känslig för valet av hyperparametrar epsilon (sökradie för en punkt till grannar) och core points (hur många grannar som måste nås med sökradien epsilon för att en punkt ska tillhöra ett kluster).

Tumregel för DBSCAN är att core pointparametern ska vara i intervallet [n_dim, 2*n_dim] och defaultinställning för epsilon är 0.5 för standardiserad data. Gör en grid_search på parameterkombinationer (kan göras med t.ex. dubbla for-loopar eller grid_search från sklearn) där man ökar på respektive parameter och testar metrics.

DBSCAN ansätter i sklearn datapunkter som inte tillhör ett kluster till -1 som är brus/noise och resterande till 0, 1... n för n kluster. Testa först att mäta Silhouette Score och Davies Bouldin Index för all data. Visualisera alla resultat i en heatmap där värdet i heatmapen är antalet identifierade kluster. Spara även parametrarna för core points, epsilon och antal kluster för de 3 bästa inställningarna baserat på Silhouette Score och Davies Bouldin Index var för sig (t.ex. genom att använda 2 olika dictionaries – ett för Silhouette Score och ett för Davies Bouldin Index).

Filtrera därefter bort alla punkter som har -1 inför varje mätning av Silhouette Score och Davies Bouldin Index och gör mätningarna igen. Spara även den här gången hur stor del av datan som inte är markerad som brus (label != -1).

Fundera lite på vad dessa metrics faktiskt säger och varför det kan vara missvisande att bara kolla på Silhouette Score/DBI utan att ta hänsyn till hur stor del av datan som inte är markerad som brus (**hint:** Vad händer i extremfallet då vi använder klustrar så lite data som möjligt?)

OBS! En del metrics förutsätter att det finns fler än en label, vilket förstås är rimligt om man ska analysera klustring för olika latenta klasser.

Använd labels från din bästa klustring från DBI och ersätt dina labels i ditt reducerade dataset. (dvs – spara featurevektorer från Silhouette Score, sätt på en ny label, använd den som hue och analysera distributionerna.)

Gör samma sak med din bästa klustring baserat på Silhouette Score.

Verkar klustringen meningsfull?

PCA + DBSCAN

Gör en parametersökning där du även inkluderar reducerade dimensioner motsvarande spannet [70,90]% av förklarad varians. Förtydligande:

- Analysera resultatet av PCA för att ta reda på hur många principal components som krävs som minst (n_min) för att förklara 70% av variansen samt hur många principal components som krävs som max (n_max) för att förklara 90% av variansen.
- Inkludera rangen PCA(n_min) till PCA(n_max) i din grid_search. I varje iteration ska du alltså dimensionsreducera till det valda antalet komponenter och sedan utföra DBSCAN för en kombination av epsilon och core points.
- Utför klustringen och välj de bästa klustringarna baserat på DBI och Silhouette Score.
- Analysera resultaten i sns.pairplot för dina labels på samma sätt som i tidigare uppgift.

Värt att tänka på – eftersom dina nya ”features” (principal components) givna av PCA är linjärkombinationer av dina ursprungliga features måste detta hanteras ifall du ska analysera dina ursprungsfeatures för respektive kluster. PCA omvandlar ju datan från formen [n_samples, n_features] -> [n_samples, n_components]. Du har fortfarande på radnivå samma samples – använd labels från ditt dimensionsreducerade dataset för att sätta labels på det ursprungliga datasetet.