Inżynierka : Features extraction

https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease

To proces wyodrębniania istotnych informacji z danych wejściowych w celu ułatwienia analiy lub dalszego przetwarzania danych.

W przypadku ekstrakcji cech dokonujemy pewnej transformacji wejściowego wektora atrybutów w wektor o mniejszej wymiarowości. W efekcie generujemy nowe cechy, przekształcone w stosunku do cech wejściowych cechy.

Aby dokładniej przewidywać i rekomendować, uczenie maszynowe obejmuje ogromne zbiory danych, które wymagają znacznych zasobów do przetworzenia. Ekstrakcja cech to skuteczna metoda stosowana w celu zmniejszenia ilości potrzebnych zasobów bez utraty ważnych informacji. Ekstrakcja cech odgrywa kluczową rolę w zwiększaniu wydajności i dokładności modeli uczenia maszynowego.

Metody:

* Analiza głównych składowych (PCA) – transformacja danych wejściowych w nową przestrzeń, w której zmienne są nieskorelowane. Pozwala na redukcję wymiarowości
* Analiza dyskryminacyjna liniowa (LDA) – wyodrębnienie cech, z uwzględnieniem klas, do których należą dane wejściowe
* Metoda k – średnich (k-means) – podział danych na k grup, tak aby obserwacje w ramach jednej grupy były podobne do siebie a jednocześnie jak najbardziej odmienne od obserwacji w innych grupach
* Z wykorzystaniem sieci neuronowych – uczenie się reprezentacji cech w warstwach ukrytych
* SVD

<https://towardsdatascience.com/feature-extraction-techniques-d619b56e31be>

PCA (Analiza składowych głównych)

PCA jest jedną z najczęściej stosowanych technik redukcji wymiarowości liniowej. Korzystając z PCA, bierzemy jako dane wejściowe nasze oryginalne dane i staramy się znaleźć kombinację cech wejściowych, która najlepiej podsumowuje oryginalny rozkład danych, aby zredukować jego oryginalne wymiary. PCA jest w stanie to zrobić, maksymalizując wariancje i minimalizując błąd rekonstrukcji, patrząc na odległości w parach. W PCA nasze oryginalne dane są rzutowane na zestaw ortogonalnych osi, a każda z osi jest uszeregowana według ważności.

PCA jest algorytmem uczącym się bez nadzoru, dlatego nie dba o etykiety danych, a jedynie o zmienność. W niektórych przypadkach może to prowadzić do błędnej klasyfikacji danych.

ICA (analiza składowych niezależnych)

ICA to liniowa metoda redukcji wymiarowości, która przyjmuje jako dane wejściowe mieszankę niezależnych składników i ma na celu prawidłową identyfikację każdego z nich (usunięcie całego niepotrzebnego szumu).

LDA (Liniowa analiza dyskryminacyjna)

LDA ma na celu maksymalizację odległości między średnią każdej klasy i zminimalizowanie rozrzutu w samej klasie. LDA używa zatem w ramach klas i między klasami jako miar.

Podczas korzystania z LDA zakłada się, że dane wejściowe są zgodne z rozkładem gaussowskim, dlatego zastosowanie LDA do danych niegaussowskich może prawdopodobnie prowadzić do złych wyników klasyfikacji.

W pythonie:

<https://www.kaggle.com/competitions/icr-identify-age-related-conditions/discussion/411466>

PCA:

from sklearn.decomposition import PCA

pca = PCA(n\_components=2)

X\_reduced = pca.fit\_transform(X)

LDA:

from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis as LDA

lda = LDA(n\_components=2)

X\_lda = lda.fit\_transform(X, y)

Feature agglomeration

from sklearn.cluster import FeatureAgglomeration

agglo = FeatureAgglomeration(n\_clusters=2)

agglo.fit(X)

X\_reduced = agglo.transform(X)

ICA:

from sklearn.decomposition import FastICA

transformer = FastICA(n\_components=7, random\_state=0)

X\_transformed = transformer.fit\_transform(X)

NMF:

from sklearn.decomposition import NMF

model = NMF(n\_components=2, init='random', random\_state=0)

W = model.fit\_transform(X)

H = model. Components\_

t-sne

from sklearn.manifold import TSNE

tsne = TSNE(n\_components=2)

X\_tsne = tsne.fit\_transform(X)

Podobne do mojego tematu:

<https://www.kaggle.com/code/bhuvanchennoju/data-storytelling-auc-focus-on-strokes>

<https://www.kaggle.com/code/yasserh/heart-disease-prediction-comparing-top-ml-models>

<https://www.scirp.org/journal/paperinformation.aspx?paperid=100958>

Heatmap:

https://miroslawmamczur.pl/014-mapa-ciepla-heatmap/

Dla zmiennych kategorycznych informacja:

Dlatego często stosuje się kodowanie "one-hot" (jedno-zimne), które tworzy osobną kolumnę dla każdej unikalnej wartości w zmiennej kategorycznej. W przypadku "kolorów" zastosowanie kodowania "one-hot" spowoduje stworzenie trzech kolumn, z których każda będzie reprezentować jedną z opcji ("czerwony", "zielony", "niebieski"). Wartości w tych kolumnach będą 0 lub 1, w zależności od tego, która opcja jest obecna w danym przypadku. Ten rodzaj kodowania jest bardziej niezależny od porządku i jest często bardziej odpowiedni do analizy danych kategorycznych w kontekście algorytmów uczenia maszynowego.

Dokumentacja PCA:

<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>

Dobry przykład preprocessingu zaawansowany: https://www.kaggle.com/code/nkitgupta/advance-data-preprocessing/notebook

LDA:

<https://stackabuse.com/implementing-lda-in-python-with-scikit-learn/>

Modele:

Klasyfikator k-najbliższych sąsiadów -KNeighborsClassifier

https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn

Drzewo decyzyjne - tree.DecisionTreeClassifier

<https://www.datacamp.com/tutorial/decision-tree-classification-python>

dobre

Las losowy - RandomForestClassifier()

https://www.datacamp.com/tutorial/random-forests-classifier-python

Regresja logistyczna - LogisticRegression(solver='liblinear')

https://www.datacamp.com/tutorial/understanding-logistic-regression-python

kmeans - https://www.datacamp.com/tutorial/k-means-clustering-python

BernoulliNB()

"Gradient Boosting Classifier"