**🔍 1. Data Verkenning (Exploratory Data Analysis - EDA)**

Doel: Inzicht krijgen in de data, kwaliteit en relaties.

* Inladen van de dataset
* Vorm (dimensies), types, eerste en laatste rijen
* Missende waarden en duplicaten
* Statistische samenvattingen (mean, std, etc.)
* Verdeling van variabelen (histogrammen, boxplots)
* Correlaties en relaties tussen variabelen (pairplots, heatmaps)
* Outliers opsporen

**🛠️ 2. Voorbewerking (Preprocessing)**

Doel: Dataset klaarmaken voor modelleren.

* Imputatie van missende waarden
* Encoderen van categorische variabelen (LabelEncoder, OneHotEncoder)
* Normalisatie of standaardisatie
* Train-test splitsing
* Eventueel feature engineering of selectie

**🧠 3. Modelselectie en Training**

Doel: Een of meerdere ML-modellen trainen.

* Keuze van model(len) (bijv. Decision Tree, Random Forest, SVM, Logistic Regression, etc.)
* Trainen van model(len) op de trainingsdata
* Hyperparameter tuning (bijv. via GridSearchCV)

**📈 4. Evaluatie van Modellen**

Doel: Prestaties beoordelen op testdata.

* Voorspellingen op testdata
* Evaluatiemetrics:
  + Regressie: MAE, MSE, R²
  + Classificatie: Accuracy, Precision, Recall, F1-score, Confusion matrix, ROC/AUC
* Visualisatie van prestaties

**🔁 5. Optimalisatie en Herhaling**

Doel: Verbeteren door iteratie.

* Feature selectie verbeteren
* Andere modellen proberen
* Balanceren van klassen (bijv. met SMOTE)
* Hyperparameters verfijnen

**💾 6. Opslaan en Hergebruik**

Doel: Model bewaren en toepassen in andere contexten.

* Opslaan van model (joblib, pickle)
* Opslaan van preprocessing stappen (bijv. Pipeline)
* Script of interface maken voor voorspellingen op nieuwe data

Top! Hier is een uitgebreide en duidelijke opsomming van de stappen bij **Exploratory Data Analysis (EDA)**, met extra aanvullingen en tips om jouw data grondig te verkennen:

**1. Data Verkenning (Exploratory Data Analysis - EDA)**

**a) Inladen van de dataset**

* Controleer of de data correct is ingeladen (bijv. juiste scheidingstekens, encoding).
* Tip: Bekijk het datatype van de ingelezen data (DataFrame, array, etc.)

**b) Vorm (dimensies), types, eerste en laatste rijen**

* shape om aantal rijen en kolommen te weten.
* dtypes om datatypes per kolom te zien.
* head() en tail() om eerste en laatste rijen te inspecteren.
* Tip: Check op rare waarden of onverwachte data.

**c) Missende waarden en duplicaten**

* isnull().sum() om missende waarden per kolom te tellen.
* Zoek naar duplicaten met duplicated().
* Tip: Visualiseer missende waarden (bijv. met missingno library).

**d) Statistische samenvattingen**

* describe() voor numerieke kolommen.
* Voor categorische kolommen: value\_counts().
* Kijk naar centrale tendentie, spreiding en verdelingen.
* Tip: Check ook de variantie en standaarddeviatie om te zien welke kolommen weinig of juist veel variatie bevatten.

**e) Verdeling van variabelen**

* Histogrammen en density plots (bijv. met matplotlib of seaborn).
* Boxplots om spreiding en outliers te identificeren.
* Voor categorische variabelen: staafdiagrammen.
* Tip: Bekijk skewness en kurtosis om afwijkende verdelingen te herkennen.

**f) Correlaties en relaties tussen variabelen**

* Correlatiematrix (Pearson/Spearman).
* Heatmap visualisatie van correlaties.
* Scatterplots en pairplots voor belangrijke variabelen.
* Tip: Kijk ook naar niet-lineaire relaties (bv. met scatterplots of plots van transformaties).

**g) Outliers opsporen**

* Boxplots per feature.
* Statistische methoden: Z-score, IQR (Interquartile Range).
* Tip: Beslis of outliers relevant zijn (datafouten of echte extreme waarden).

**h) Univariate en bivariate analyses**

* Univariate: Verdeling van 1 variabele.
* Bivariate: Verband tussen 2 variabelen, bijvoorbeeld t.o.v. de targetvariabele.
* Tip: Gebruik crosstabs en grouped boxplots bij categorische variabelen.

**i) Time series (indien van toepassing)**

* Plot tijdreeksen.
* Controleer op trends, seizoensinvloeden, en missing periods.

**j) Data kwaliteit en consistentie checks**

* Zoek naar inconsistenties (bv. negatieve leeftijden, verkeerde datums).
* Controleer op logische afhankelijkheden tussen variabelen.

**Tips voor het exploreren van je data:**

* Begin breed en zoom daarna in op interessante of problematische delen.
* Gebruik visualisaties vooral om snel patronen en issues te zien.
* Documenteer bevindingen, zodat je ze later kunt meenemen in preprocessing.
* Werk iteratief: na EDA pas je preprocessing aan, en ga eventueel terug om je inzichten te toetsen.
* Combineer statistiek met visualisatie voor een completer beeld.
* Gebruik Jupyter Notebooks voor interactieve exploratie.

Wil je nog tips over handige libraries/tools voor EDA in Python, of voorbeelden van visualisaties?

Goede vraag! Automatiseren van EDA met zo min mogelijk gebruikersinput is superhandig, maar soms heb je juist input nodig om goede keuzes te maken. Hier is een overzicht van hoe je dat slim aanpakt:

**Automatiseren van EDA — zo min mogelijk gebruikersinput**

**1. Data-informatie automatisch ophalen**

* Laat het script automatisch:
  + Dimensies, types, missing values, duplicaten berekenen.
  + Statistische samenvattingen maken.
  + Basisvisualisaties zoals histogrammen, boxplots, en correlatiematrices genereren.
* Maak gebruik van bestaande libraries zoals **pandas-profiling**, **sweetviz**, of **dataprep** die uitgebreide rapporten genereren zonder input.

**2. Standaardregels instellen**

* Definieer vaste criteria voor outlier detectie (bijv. IQR methode met vaste factor).
* Automatische detectie van categorische vs numerieke kolommen (op basis van datatype en aantal unieke waarden).
* Voor missende waarden: automatische keuzes maken, bv. kolommen met > X% missing values markeren.
* Data splitsen (train/test) op vaste ratio (bv. 80/20) zonder input.

**3. Automatische feature type detectie**

* Gebruik heuristieken om te bepalen wat de doelvariabele is (bv. kolom met ‘target’ in naam).
* Automatisch bepalen of het een classificatie- of regressieprobleem is (categorisch vs numeriek target).

**4. Logging en rapportage**

* Laat het systeem rapporten en plots opslaan met duidelijke namen.
* Eventueel een samenvattend tekstbestand met bevindingen (bijv. welke kolommen veel missing hebben).

**Wanneer wel om gebruikersinput vragen?**

**a) Onzekerheid of ambiguïteit**

* Bijvoorbeeld: Welk type probleem is het? (classificatie of regressie)
* Wat is de targetvariabele?
* Moeten bepaalde kolommen uitgesloten worden? (bv. ID’s of tijdstempels)

**b) Beslissingen die impact hebben op model**

* Hoe omgaan met missende waarden? Imputatie of verwijderen?
* Welk outlierbeleid moet worden toegepast?
* Welke features wil je handmatig toevoegen of verwijderen?

**c) Extra context en domeinkennis**

* Soms heeft de gebruiker inzicht in de betekenis van features, die automatische detectie niet kan vatten.
* Bij tijdreeksen of geospatiale data kan specifieke kennis belangrijk zijn.

**d) Interactie voor interpretatie**

* Vragen stellen na het tonen van plots/rapporten, bv. “Wil je deze kolom verder onderzoeken/verwijderen?”
* Keuze bieden om automatische preprocessing aan te passen.

**Praktische tip: hybride aanpak**

* Begin met een volledig automatische EDA-run met default instellingen.
* Laat de gebruiker daarna optioneel ingrijpen of keuzes maken via een overzichtelijk menu of config-file.
* Zo minimaliseer je input maar behoud je controle.

Wil je dat ik help met een template of voorbeeldcode die zo’n geautomatiseerde EDA pipeline start, inclusief plekken waar je input kan vragen?