Lernzettel

Einführung in Data Science: Datenvorverarbeitung, Merkmalsextraktion, Modellpipeline

Universität: Technische Universität Berlin
Kurs/Modul: Informationssysteme und Datenanalyse

Erstellungsdatum: September 19, 2025

Lernzettel: Einführung in Data Science: Datenvorverarbeitung, Merkmalsextraktion, Modellpipeline

(1) Datenvorverarbeitung. Die Datenvorverarbeitung umfasst alle Schritte, die Rohdaten in eine Form bringen, die für Lernmodelle geeignet ist. Ziel ist saubere, konsistente Eingabedaten, die gute Reproduzierbarkeit ermöglichen.

Wichtige Schritte:

- Datentypen klären und konsistent halten (numerisch, kategorial, zeitlich, textuell).
- Umgang mit fehlenden Werten (*Imputation*): Numerische Merkmale: Mittelwert oder Median; Kategoriale Merkmale: häufigster Wert.
- Duplikate entfernen und inkonsistente Einträge bereinigen.
- Merkmals-Skalierung: Standardisierung (z-Score) oder Normalisierung.

$$z = \frac{x - \mu}{\sigma}$$

- Kodierung kategorialer Merkmale: One-Hot-Encoding oder Ordinal-Encoding, je nach Modell.
- Aufteilung in Trainings-, Validierungs- und Testdaten, um Datenleckagen zu vermeiden.
- Zeitreihen- bzw. Streaming-spezifische Vorverarbeitung beachten (Offset, Verzögerung, Windowing).

Hinweise:

- Bleib konsistent bei Skalierung zwischen Trainings- und Testdaten.
- Vermeide *Data Leakage*: Informationen aus dem Testset dürfen nicht in das Training fließen.

Beispiele (Formeln)
Normalisierung (Min-Max-Skalierung) eines Merkmals x:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Typische Techniken
- Numerische Merkmale: Skalierung, Transformation (Log, Box-Cox).
- Kategoriale Merkmale: One-Hot-Encoding, Zielcodierung.
- Textdaten: Vorverarbeitung (Tokenisierung, Stoppwort-Entfernung) als Vorbereitung auf Merkmalsextraktion.
- Zeitreihen: Resampling, Rolling Statistics, Synchronisation von Features.

Beispiel-Workflow
1) Rohdaten laden und prüfen, 2) fehlende Werte imputieren, 3) kategoriale Merkmale kodieren, 4) Merkmale skalieren, 5) Daten in Train/Test spliten.

(2) Merkmalsextraktion. Merkmale (Features) sind transformierte Repräsentationen der Rohdaten, die das Lernverfahren unterstützen. Merkmalsextraktion kann explizit (Merkmalskombination) oder durch Verfahren zur Dimensionsreduktion erfolgen.

Typen von Merkmalen:

- Numerische Merkmale (continous, Integer)
- Kategoriale Merkmale (nominal/ordinal)
- Textdaten (Dokumente, Tweets)
- Bild-/Audit-/Sensormerkmale (Patches, Frequenzen)

Gängige Methoden
- One-Hot-Encoding für kategoriale Merkmale.
- Textdaten: Bag-of-Words, Term Frequency–Inverse Document Frequency (TF-IDF).
- Dimensionalitätsreduktion: Principal Component Analysis (PCA) oder Singular Value Decomposition (SVD).
Grundidee: Merkmale zentrieren, Kovarianzstruktur erfassen und Hauptkomponenten extrahieren.

$$\Sigma = \frac{1}{n-1} Z^\top Z, \quad Z \text{ standardisiert}$$

Neue Merkmale: $Y = ZW$ mit W aus den Eigenrichtungen der Kovarianz Σ. - Merkmalsauswahl: Filter-, Wrapper- und Embedded-Methoden (Mutual Information, Chi-Quadrat, Lasso). - Embeddings (fortgeschritten): Wort-Embeddings (z. B. Word2Vec, GloVe) für Text; Bild-Embeddings mittels CNN.

Beispiele (Formeln)
PCA-Transformation eines Stichprobenvektors x zu einem neuen Merkmalsvektor y:

$$y_j = x \cdot w_j, \quad j = 1, \ldots, m$$

wobei w_j die j-te Eigenrichtung ist.

Anwendungsbeispiele
- Textklassifikation mit TF-IDF + logistische Regression.
- Bildklassifikation mit Bild-Feature-Extractoren + PCA-Reduktion.
- Sensorikdaten: Zeitfenster-Features (Durchschnitt, Varianz, Max), ggf. FFT-Koeffizienten.

(3) Modellpipeline. Eine Modellpipeline verbindet Vorverarbeitung, Merkmalsextraktion und Modell in einer wiederverwendbaren Sequenz. Ziel ist eine reproduzierbare, stabile Analyse- und Lernkette.

Warum Pipelines?

- Verhindern von Datenleckagen durch klare Trennung von Training und Test.
- Konsistente Transformationen bei Validation und Deployment.
- Einfaches Hyperparameter-Tuning über geregelte Schritte.

Beispielaufbau (Pseudo-Code)

```
Pipeline([
  ('pre', ColumnTransformer([
      ('num', StandardScaler(), numeric_cols),
      ('cat', OneHotEncoder(handle_unknown='ignore'), cat_cols)
  ])),
  ('model', LogisticRegression(max_iter=1000, random_state=42))
])
_fit(X_train, y_train)
```

Wichtige Konzepte

- Cross-Validation (z. B. K-Fold) zur robusten Leistungsbeurteilung.
- Hyperparameter-Tuning (GridSearchCV, RandomizedSearchCV).
- Reproduzierbarkeit: deterministischer Zufall, feste Random Seeds.
- Metriken: Accuracy, Precision/Recall, F1, ROC-AUC; bei Regression RMSE/MAE.
- Data Leakage vermeiden: Score-Optimierung nie auf Testdaten durchführen.

(4) Beispiel-Workflow eines Data-Science-Projekts
- Datenakquise und Vorverarbeitung (Datenqualität prüfen)
- Merkmalsextraktion und ggf. Reduktion der Dimensionalität
- Aufbau einer Pipeline mit Modell
- Evaluation auf einem Hold-Out-Set oder via Cross-Validation
- Hyperparameter-Tuning und Validierung
- Deployment-Vorbereitung (Dokumentation, Reproduzierbarkeit)

Hinweis zu Modellpipelines
- Die Reihenfolge der Schritte ist entscheidend.
- Alle Transformationen sollten auf dem Training nur gelernt werden und dann auf den Test übertragen werden.
- Bei zeitlich abhängigen Daten (Streaming) ggf. inkrementelle oder Online-Modelle verwenden.