

Mit KI am Ende weniger KI nutzen

Erstellen Sie mit KI eine dokumentierte, getestete Python-Analyse in einem lokalen VS-Code-Projekt. Danach führen Sie den gespeicherten Code ohne weitere KI-Anfrage aus. Prüfen Sie Zahlen und Abbildung, bevor Sie die Ergebnisse interpretieren.

Kernidee

Trennen Sie Analyseplanung, Software-Entwurf, Umsetzung, Ausführung und Interpretation:

Von der Fragestellung zur geprüften Analyse

Sechs Phasen · gespeicherte Artefakte · menschliche Entscheidungen



Türkis: KI-Beitrag · Orange: menschliche Entscheidung · Grau: gemeinsamer oder lokaler Schritt

Jede Phase liefert eine Datei oder ein überprüfbares Ergebnis für den nächsten Schritt. So entsteht eine wiederverwendbare Analyse.

Die Nummern entsprechen den sechs Phasen unten. Prüfen und genehmigen Sie Spezifikationen und Abnahmetests vor der Implementierung. Interpretieren Sie nur überprüfte Ergebnisse. Phase 5 lässt sich mit dem gespeicherten Code ohne weitere KI-Anfrage wiederholen.

Projektordner organisieren

Erstellen Sie zunächst vier leere Ordner: `.agents/`, `analysis/`, `data/` und `results/`. Legen Sie Daten oder einen genehmigten lokalen Zugriffsweg vor der Ausführung unter `data/` ab. Die Rohdaten bleiben schreibgeschützt. Kopieren Sie sensible Daten nicht unnötig in Prompts oder Ausgaben.

Erwartete Endstruktur

Spezifikationen und Pläne liegen unter `.agents/`, Code mit Abhängigkeiten und Tests unter `analysis/`, Eingaben unter `data/` und Ergebnisse unter `results/`:

```
project/
├── .agents/
│   ├── ANALYSIS.md
│   ├── PYTHON.md
│   ├── PLAN.md
│   └── WORKFLOW.md
├── analysis/
│   ├── <Analyse-Implementierung>
│   ├── <Abhängigkeits- oder Umgebungsdefinition>
│   └── <automatisierte Tests>
├── data/
│   └── <Ausgangsdatensatz oder -datensätze>
└── results/
    ├── <Ergebnisabbildung>
    └── <Ergebnistabelle>.csv
```

Drei Dateien steuern die Analyse:

- `ANALYSIS.md` legt fest, was herausgefunden werden soll und welche statistische Vorgehensweise dazu passt.
- `PYTHON.md` legt fest, was die Software zur Umsetzung der Analyse leisten muss.
- `PLAN.md` legt fest, wie die Implementierung erfolgt und wie ihre Korrektheit geprüft wird.

`WORKFLOW.md` beschreibt die wiederverwendbare Methode. Ergebnisnamen sind frei wählbar; eine Abbildung und ihre zugrunde liegende CSV-Tabelle gehören dazu. Die Abbildung muss die Werte dieser Tabelle zeigen. In Challenge 2 heisst der Analyseplan `CODE_PLAN.md`, um ihn vom Berichtsplan zu unterscheiden.

Workflow in sechs Phasen

Phase 1 — Fragestellung eingrenzen

Zweck: Von einem Datensatz oder Themengebiet zu einer relevanten, beantwortbaren Frage und einer statistischen Analysespezifikation gelangen.

Beispielauftrag:¹

Schlage anhand der verfügbaren Variablen eine interessante Forschungsfrage vor. Erstelle für die ausgewählte Frage eine fundierte Analysespezifikation und speichere sie unter `.agents/ANALYSIS.md`.

Für einen datenschutzfreundlichen Workflow erhält die KI eine Variablenliste, ein Datenwörterbuch, ein Schema, zulässige Wertebereiche, eine aggregierte Beschreibung oder synthetische Beispiele – nicht die sensiblen Rohdaten. Arbeitet die KI vollständig in einer genehmigten lokalen Umgebung, kann sie stattdessen den lokalen Datensatz untersuchen, ohne dass dessen Inhalte die Umgebung verlassen.

Die Analysespezifikation definiert:

- die untersuchte Frage
- Eingaben, Expositionen, Zielgrößen und abgeleitete Variablen
- deskriptive und inferenzstatistische Analysen
- Kovariaten, Referenzgruppen und Vergleichsrichtungen
- Annahmen, Sensitivitätsanalysen und Grenzen
- erforderliche Analyseergebnisse
- Kriterien für den Abschluss der Analyse

1. Beachten Sie, dass der Ansatz, generative KI eine Forschungsfrage entwickeln zu lassen, nutzlos ist und hier nur verwendet wird, weil der Fokus dieses Tutorials auf dem Workflow liegt. In der Realität würde die interessierende Frage aus dem Fachgebiet abgeleitet. Sie sollte der Ausgangspunkt der Datenerhebung und -analyse sein.

Phase 2 — Berechnung spezifizieren

Zweck: Die konzeptionelle statistische Analyse in eine eindeutige Software-Spezifikation übersetzen.

Beispielauftrag:

Übersetze `.agents/ANALYSIS.md` in eine vollständige Spezifikation für Python-Code. Definiere Datenvalidierung, statistische Berechnungen, Reproduzierbarkeitskontrollen, den Aufbau und Inhalt der Ausgabe-CSV und die Ergebnisabbildung. Speichere die Spezifikation unter `.agents/PYTHON.md`.

Die Software-Spezifikation definiert:

- Eingabepfade und erforderliches Datenschema
- Datenqualitätsprüfungen und Verhalten bei Fehlern
- Klarer Abbruch bei ungültigen Daten oder fehlgeschlagener statistischer Modellschätzung
- abgeleitete Variablen und genaue statistische Modelle
- Referenzniveaus, Kontrastrichtungen, Konfidenzintervalle und Zufalls-Seeds
- Verhalten auf der Kommandozeile und Abhängigkeiten
- Schema, Sortierung und Genauigkeit der Ergebnistabelle
- Inhalt und Gestaltungsanforderungen der Abbildung
- numerische, strukturelle und visuelle Prüfkriterien

Phase 3 — Meilensteine und Abnahmetests planen

Zweck: Vor der Implementierung festlegen, wie die Software gebaut wird und woran ihre Korrektheit erkannt wird.

Beispielauftrag:

Erstelle `.agents/PLAN.md` mit kleinen Implementierungsmeilensteinen. Definiere für jeden Meilenstein das erwartete Ergebnis und aussagekräftige Abnahmetests für Validierung, statistische Berechnungen, Reproduzierbarkeit und Ausgabeanforderungen.

Der Plan sollte folgende Meilensteine enthalten:

1. Daten laden und validieren.
2. Abgeleitete Variablen konstruieren.
3. Einfache Berechnungen anhand unabhängig überprüfbarer Beispiele reproduzieren.
4. Die festgelegten statistischen Modelle schätzen.
5. Ergebnistabelle zusammenstellen und validieren.
6. Abbildung aus der Ergebnistabelle erzeugen.
7. Den vollständigen Workflow ausführen und die Artefakte prüfen.

Abnahmetests sollten Folgendes abdecken:

- Zurückweisung fehlender, doppelter, fehlerhaft formatierter oder ausserhalb des Wertebereichs liegender Daten
- korrekte Vergleichsrichtungen und Schwellenwertdefinitionen
- bekannte Berechnungen anhand kleiner synthetischer Beispiele
- deterministische Ergebnisse mit festgelegtem Zufalls-Seed
- Erhalt der unveränderten Ausgangsdaten
- vorgeschriebenes CSV-Schema, Zeilenzahl, Wertebereiche und Invarianten der Konfidenzintervalle
- Erzeugung genau der erforderlichen Ergebnisartefakte
- Übereinstimmung der Werte in CSV-Datei und Abbildung
- erfolgreiche vollständige Ausführung mit einem dokumentierten Befehl

Legen Sie die Tests vor der Umsetzung fest. Prüfen Sie damit die vereinbarten Anforderungen, statt die Kriterien nachträglich an den Code anzupassen.

Phase 4 — Gemäss den Spezifikationen implementieren

Zweck: Die Implementierung und die bereits geplanten Tests erstellen.

Beispielauftrag:

Implementiere `.agents/PYTHON.md` gemäss `.agents/PLAN.md`. Lege die vollständige Anwendung unter `analysis/` ab, erstelle die geplanten automatisierten Tests und bringe sie zum Bestehen, ohne die Spezifikationen oder Abnahmekriterien abzuschwächen.

Anforderungen an die Implementierung:

- Der Code liegt unter `analysis/`.
- Die Daten werden aus `data/` gelesen und nicht verändert.
- Abhängigkeiten und Ausführungsanweisungen werden dokumentiert.
- Die automatisierten Tests setzen die Abnahmekriterien aus Phase 3 um.
- Zufallsbasierte Verfahren verwenden ausdrücklich festgelegte Seeds.
- Statistische Schätzwerte werden einmal berechnet und in einer strukturierten Tabelle gespeichert.
- Die Abbildung wird aus dieser Ergebnistabelle erzeugt und nicht aus einer separaten, möglicherweise abweichenden Berechnung.

Phase 5 — Lokal ausführen und überprüfen

Zweck: Tests und Analyse in der kontrollierten Ausführungsumgebung laufen lassen und anschliessend die erzeugten Artefakte überprüfen.

Minimaler Beispielauftrag oder lokale Anweisung:

Führe die automatisierten Tests und die Analyse lokal aus. Verändere die Ausgangsdaten nicht. Bestätige, dass `results/` die erforderliche Abbildung und CSV-Datei enthält, prüfe die Abbildung visuell und verifiziere, dass ihre Werte mit der Tabelle übereinstimmen.

Die Ausführung muss nachweisen:

- Alle automatisierten Tests bestehen.
- Der dokumentierte Analysebefehl wird erfolgreich beendet.
- Die Eingabedaten bleiben unverändert.
- Wiederholte Ausführungen mit identischen Eingaben, Seeds und Abhängigkeitsversionen erzeugen identische numerische Ergebnisse.
- Die Ergebnistabelle erfüllt ihr Schema und ihre numerischen Invarianten.
- Die Abbildung ist lesbar, vollständig und mit der Tabelle konsistent.
- `results/` enthält nur die erwarteten Analyseergebnisse.

Die Abbildung unter «Datenschutz- und Sicherheitsgrenze» zeigt den lokalen Datenfluss und die Rückkehr zur Implementierung, falls eine Prüfung fehlschlägt.

Phase 6 — Evidenz interpretieren

Zweck: Die überprüften numerischen Ergebnisse verständlich erklären und dabei das Analysedesign und seine Grenzen berücksichtigen.

Beispielauftrag:

Erkläre die Ergebnisse ausschliesslich anhand der erzeugten Abbildung, der aggregierten Ergebnistabelle und der Analysespezifikation in verständlicher Sprache. Berichte Grössenordnungen und Unsicherheit, unterscheide Zusammenhänge von Kausalität und beschreibe die wichtigsten Grenzen.

Die Interpretation sollte:

- mit der Antwort auf die ursprüngliche Frage beginnen
- wichtige Effektgrössen in aussagekräftigen Einheiten erklären
- Konfidenzintervalle erläutern, statt sich nur auf Signifikanzbezeichnungen zu stützen
- zeigen, ob eine Adjustierung den deskriptiven Vergleich wesentlich verändert
- deskriptive Ergebnisse, adjustierte Zusammenhänge und kausale Aussagen voneinander unterscheiden
- Grenzen durch Messung, Confounding, Auswahl und Generalisierbarkeit benennen
- keine individuellen Datensätze offenlegen oder zitieren

Ergebnis dieser Phase:

- eine verständliche, auf den gespeicherten Ergebnissen beruhende Erklärung
- optional eine separat gespeicherte Interpretation oder ein Bericht, wenn das Projekt dies verlangt

Die Interpretation kann mit KI-Unterstützung ausschliesslich anhand aggregierter Ergebnisse erstellt werden. Sie kann auch vollständig von einem Menschen verfasst werden; das reproduzierbare Analyseergebnis hängt nicht von einer KI-Interpretation ab.

Ergebnisse nach Phase

Phase	Dauerhaftes Artefakt oder überprüftes Ergebnis
Ausgangslage	Leere Ordner <code>.agents/</code> , <code>analysis/</code> , <code>data/</code> und <code>results/</code>
1. Fragestellung	<code>.agents/ANALYSIS.md</code>
2. Berechnung	<code>.agents/PYTHON.md</code>
3. Planung und Tests	<code>.agents/PLAN.md</code> mit Meilensteinen und Abnahmetests
4. Implementierung	Ausführbarer Code, Abhängigkeitsdefinition und automatisierte Tests unter <code>analysis/</code>
5. Ausführung	Überprüfte Abbildung und zugrunde liegende CSV-Datei unter <code>results/</code>
6. Interpretation	Verständliche Erklärung auf Grundlage der überprüften aggregierten Ergebnisse

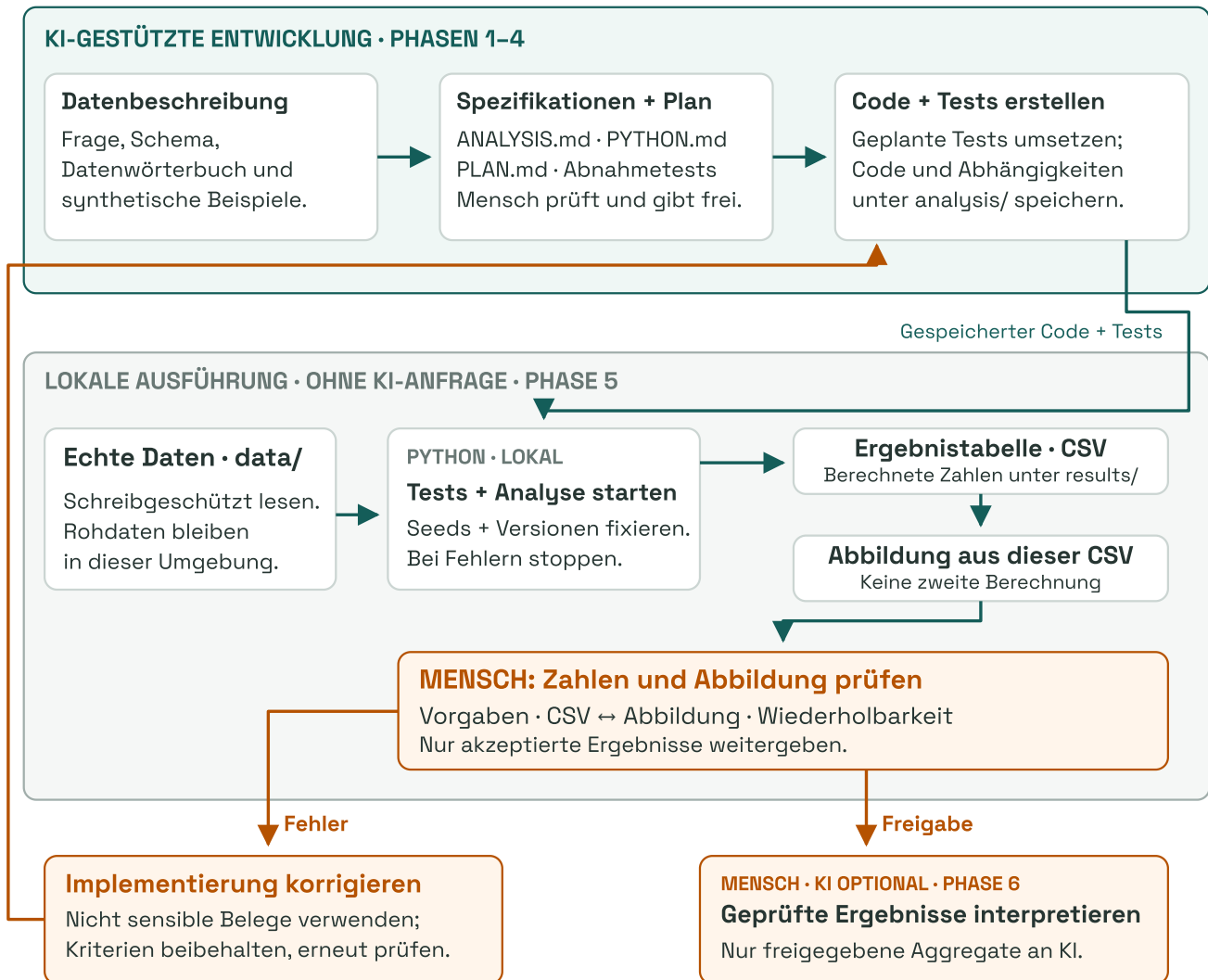
Weiterverwenden

Eine andere Person soll Spezifikationen prüfen, Abhängigkeiten installieren, Tests und Programm ausführen und Ergebnisse reproduzieren können. Die Interpretation lässt sich danach überarbeiten, ohne die Analyse neu auszuführen.

Datenschutz- und Sicherheitsgrenze

Mit KI entwickeln, lokal ohne KI ausführen

Variante ohne KI-Zugriff auf sensible Rohdaten



Rohdaten bleiben lokal. Auch Aggregate vor einer Weitergabe prüfen.

Türkis: KI-Beitrag · Orange: menschliche Entscheidung · Grau: lokale Ausführung

Die Abbildung zeigt die Variante ohne KI-Zugriff auf sensible Rohdaten. Die Ergebnisabbildung entsteht aus der CSV-Tabelle. Bei Fehlern korrigieren Sie die betroffene Implementierung und prüfen erneut; schwächen Sie die vereinbarten Abnahmekriterien nicht ab.

Datenschutz- und Sicherheitsgrenze (Fortsetzung)

Um sensible Daten nicht an die KI weiterzugeben:

1. Stellen Sie der KI ein Schema, ein Datenwörterbuch, synthetische Beispiele und Analyseanforderungen statt des sensiblen Datensatzes bereit.
2. Erzeugen Sie Spezifikationen, Pläne, Code und Tests anhand dieser nicht sensiblen Beschreibung.
3. Übertragen Sie die erzeugte Anwendung in eine kontrollierte lokale Umgebung.
4. Führen Sie sie dort mit den echten Daten aus.
5. Geben Sie für eine mögliche KI-Interpretation nur geprüfte aggregierte Ergebnisse weiter.

Eine lokale Datei bleibt nicht automatisch privat, wenn ein KI-Agent sie liest. Umgebung, Konnektor, Aufbewahrungsrichtlinie und institutionelle Vorgaben müssen bekannt sein, bevor Zugriff erteilt wird. Der Workflow bietet eine Möglichkeit, einen solchen Zugriff zu vermeiden; er garantiert Datenschutz nicht von selbst.

Sensible Werte dürfen nicht in Protokollen, Fehlermeldungen, Abbildungen, Testdaten oder erzeugter Dokumentation erscheinen. Verwenden Sie für Berechnungs- und Validierungstests nach Möglichkeit synthetische Testdaten.

Wiederholbarkeit prüfen

Die Prüfungen stehen in den Phasen 2–5. Halten Sie Testerwartungen möglichst unabhängig vom Code. Gleiche Eingaben, Seeds und Abhängigkeiten sollen gleiche numerische Ergebnisse liefern. Ein wiederholbares Ergebnis kann dennoch falsch sein; prüfen Sie Berechnung und Bedeutung getrennt.

Kurzform

Verwenden Sie KI, um aus einer klar definierten Frage getestete lokale Software zu erstellen. Führen Sie diese Software anschliessend reproduzierbar mit geschützten Daten aus und interpretieren Sie nur die überprüften aggregierten Ergebnisse.

Spezifizieren → Planen und testen → Implementieren → Lokal ausführen → Interpretieren