**From Dataset to a Finished Machine Learning Model**

1. **Data Acquisition** - Import and convert data into a pandas DataFrame:   
   This step involves loading data from a source (e.g., CSV file, database) and saving it in a pandas DataFrame. You can also import necessary libraries to work with the data.
2. **Data Cleaning** - Remove irrelevant data and format data if necessary:   
   Identify and remove irrelevant or redundant columns that are not relevant to the model. Format the data to ensure it is suitable for modeling. This may include cleaning NaN values, unifying data types, and removing duplicates, spaces, or special characters.
3. **Exploratory Data Analysis** - Examine data, especially column names:   
   After importing the data, it's important to take an initial look at the data. Review the first rows of the DataFrame to understand the data structure and column names. Analyze the data to gain a better understanding of its distribution, patterns, and relationships between variables. This can be done through visualizations and statistical analyses.
4. **Feature Engineering** - Define features and targets, consider data leakage: Determine the features (independent variables) to be used in the model and the target (dependent variable) to be predicted. Ensure that no information from the future (data leakage) is used in the training process. In this step, new features are created from existing data or existing features are transformed to better align the model with the task. This may involve creating categorical variables, scaling features, or extracting relevant information.
5. **Data Split**: The data is divided into training data, validation data, and test data. Typically, most of the data is used for training the model, a smaller portion for validation to select the model, and an independent test dataset for the final evaluation.
6. **Model Selection:** In this step, you choose the appropriate machine learning model that best suits the task. This can vary depending on the task and data, such as linear models, decision trees, neural networks, etc.
7. **Model Training:** The selected model is trained on the training data by learning the relationships between the features and the target value. This may involve tuning model parameters and performing cross-validation.
8. **Model Evaluation:** The trained model is tested on the validation data to assess its performance. Metrics such as accuracy, F1 score, RMSE (Root Mean Square Error), Mean Absolute Error, etc., are used to evaluate model performance.
9. **Model Fine-Tuning:** Based on the results of model evaluation, adjustments to model parameters can be made to improve performance. This step can be iterative.
10. **Final Model Evaluation:** Once the model is optimized, it is tested on the independent test dataset to evaluate its performance in a real-world environment and ensure it generalizes well.
11. **Model Deployment:** Finally, the trained and tested model is deployed in a production environment where it can be used for predictions on new data.
12. **Monitoring and Maintenance:** After deployment, the model is monitored and maintained to ensure it continues to make accurate predictions and meets changing requirements.

In addition to these steps, it's important to regularly review and optimize to ensure the model makes accurate predictions and operates efficiently. This may also include the use of cross-validation, hyperparameter tuning, and other advanced techniques.

**Data Pipeline:** Create a data pipeline that combines preprocessing steps like feature scaling, feature engineering, and model selection into a single step. This helps make the code more organized and maintainable. Using a pipeline allows for seamless automation and organization of these steps, improving code reusability and efficiency in model development. This is particularly important in complex projects where multiple data processing and modeling steps are required.

Vom Datensatz zum fertigen Machine Learning Modell

1. **Datenerfassung - Daten importieren und in einen pandas Dataframe umwandeln:** Dieser Schritt beinhaltet das Laden der Daten aus einer Quelle (z.B. CSV-Datei, Datenbank) und das Speichern in einem pandas DataFrame. Hier kannst du auch notwendige Bibliotheken importieren, um mit den Daten zu arbeiten.
2. **Datenbereinigung - Unrelevante Daten entfernen und Daten formatieren, wenn nötig:** Identifiziere und entferne irrelevante oder redundante Spalten, die für das Modell nicht relevant sind. Formatiere die Daten, um sicherzustellen, dass sie für die Modellierung geeignet sind. Dies kann das Bereinigen von NaN-Werten, das Vereinheitlichen von Datentypen und das Entfernen von Duplikaten, Leerzeichen oder Sonderzeichen umfassen.
3. **Explorative Datenanalyse - Daten anschauen, vor allem die Säulennamen:** Nach dem Importieren der Daten ist es wichtig, einen ersten Blick auf die Daten zu werfen. Überprüfe die ersten Zeilen des DataFrames, um ein Verständnis für die Datenstruktur und die Säulennamen zu bekommen. Hier analysiert man die Daten, um ein besseres Verständnis für ihre Verteilung, Muster und Beziehungen zwischen den Variablen zu erhalten. Dies kann durch Visualisierungen und statistische Analysen erfolgen.
4. **Feature Engineering - Features und Target(s) definieren, Datenleakage beachten:** Bestimme die Features (unabhängigen Variablen), die für das Modell verwendet werden sollen, und das Target (abhängige Variable), das vorhergesagt werden soll. Achte darauf, dass keine Informationen aus der Zukunft (Data Leakage) im Trainingsprozess verwendet werden. Bei diesem Schritt werden neue Features aus den vorhandenen Daten erstellt oder bestehende Features transformiert, um das Modell besser auf die Aufgabe anzupassen. Dies kann das Erstellen von kategorialen Variablen, das Skalieren von Features oder das Extrahieren von relevanten Informationen beinhalten.
5. **Aufteilung der Daten:** Die Daten werden in Trainingsdaten, Validierungsdaten und Testdaten aufgeteilt. Typischerweise verwendet man einen Großteil der Daten für das Training des Modells, einen kleineren Teil für die Validierung zur Modellauswahl und einen unabhängigen Testdatensatz zur abschließenden Bewertung.
6. **Modellauswahl:** In diesem Schritt wählt man das geeignete Machine Learning-Modell aus, das am besten zur Aufgabe passt. Dies kann je nach Aufgabe und Daten variieren, z.B. lineare Modelle, Entscheidungsbäume, neuronale Netze usw.
7. **Modell trainieren:** Das ausgewählte Modell wird auf den Trainingsdaten trainiert, indem es die Beziehungen zwischen den Features und dem Zielwert lernt. Dies kann das Abstimmen von Modellparametern und das Durchführen von Cross-Validation einschließen.
8. **Modell bewerten: Das trainierte Modell wird auf den Validierungsdaten getestet, um seine Leistung zu bewerten. Metriken wie Genauigkeit, F1-Score, RMSE (Root Mean Square Error), Mittlerer absoluter Fehler usw. werden verwendet, um die Modellleistung zu bewerten.**
9. **Modellfeinabstimmung: Basierend auf den Ergebnissen der Modellbewertung können Anpassungen an Modellparametern vorgenommen werden, um die Leistung zu verbessern. Dieser Schritt kann iterativ durchgeführt werden.**
10. **Endgültige Modellbewertung: Nachdem das Modell optimiert ist, wird es auf dem unabhängigen Testdatensatz getestet, um seine Leistung in einer realen Umgebung zu bewerten und sicherzustellen, dass es gut generalisiert.**
11. **Modellbereitstellung: Schließlich wird das trainierte und geprüfte Modell in der Produktionsumgebung bereitgestellt, wo es für Vorhersagen auf neuen Daten verwendet werden kann.**
12. **Monitoring und Wartung: Nach der Bereitstellung wird das Modell überwacht und gewartet, um sicherzustellen, dass es weiterhin genaue Vorhersagen macht und den sich ändernden Anforderungen entspricht.**

Zusätzlich zu diesen Schritten ist es wichtig, regelmäßig zu überprüfen und zu optimieren, um sicherzustellen, dass das Modell genaue Vorhersagen macht und effizient arbeitet. Dies kann auch die Anwendung von Cross-Validation, Hyperparameter-Tuning und weiteren fortgeschrittenen Techniken umfassen.

**Datenpipeline:** Erstelle eine Datenpipeline, die Preprocessing-Schritte wie Feature Scaling, Feature Engineering und die Auswahl des Modells in einem einzigen Schritt kombiniert. Dies hilft, den Code übersichtlicher und wartbarer zu gestalten. Die Verwendung einer Pipeline ermöglicht es, diese Schritte nahtlos zu automatisieren und zu organisieren, wodurch die Wiederverwendbarkeit von Code und die Effizienz bei der Modellentwicklung verbessert werden. Dies ist besonders wichtig in komplexen Projekten, in denen mehrere Schritte der Datenverarbeitung und Modellierung durchgeführt werden müssen.