Einführung in das Maschinelle Lernen **Universität Hamburg**
WiSe 2023/24 Sören Laue

Übungsblatt 2

In dieser Übung sollen Sie sich mit polynomialer Regression und dem Konzept der Kreuzvalidierung (k-fold cross-validation) auseinandersetzen. Ziel wird es sein, für mehrere Datensätze Regressions-Modelle unterschiedlicher Komplexität (mit steigendem Grad des Polynoms) zu testen und das beste Modell zu bestimmen.
Für die Programmieraufgaben steht ein Jupyter-Notebook `polynomial_regression.ipynb` als Vorlage bereit, das für die Bearbeitung der Aufgaben erweitert werden soll. Sie können für alle Programmieraufgaben sowohl `scikit-learn` als auch `numpy` verwenden.

Aufgabe 1 – 20%

Die klassische k-fache Kreuzvalidierung kann ggf. unerwünschte Ergebnisse liefern, wenn die Folds ungünstig erstellt werden, besonders für seltene Features oder Gruppen (z.b. Kinder unter vielen erwachsenen Testpersonen). Bestimmte Varianten der Methodik lösen dieses Problem unterschiedlich, z.B. die häufig verwendeten folgenden:

- Leave-One-Out cross-validation: Benutzt so viele Folds wie es Datenpunkte gibt, wobei immer nur ein einziger Datenpunkt als Test-Fold benutzt wird
- Stratified cross-validation: Sorgt dafür, dass relevante Features (z.B. Kind / nicht Kind) gleichmäßig über die Folds verteilt sind

Erklären Sie zunächst, welches Problem in der klassischen Kreuzvalidierung entsteht und warum die beiden Varianten das Problem lösen. Welche Herausforderungen sehen Sie für die Anwendung der beiden alternativen Methoden?

Aufgabe 2 – 20%

In der Praxis wird bei polynomialer Regression selten mit Polynomen über einem Grad von 3 gearbeitet, also meistens linear, Grad 2 oder 3. Welche möglichen Gründe könnte das haben?

Aufgabe 3 – 60%

Definieren Sie eine Funktion `polynomial_regression(X,y,n)`, die für eine gegebene Datenmatrix X und die dazugehörigen Ausgabewerte y, die Parameter eines Polynoms n-ten Grades bestimmt, welches die Abhängigkeit zwischen X und y bestmöglich beschreibt. Je verwendetem Lösungsweg kann entweder ein gelerntes Modell oder die gelernten Parameter des Modells zurückgeben werden.

Implementieren Sie nun eine Funktion `kfold_crossval(X,y,n)`, der eine Datenmatrix X, die dazugehörigen Ausgabewerte y, und der Grad n des Polynoms übergeben wird. Die Methode soll für einen entsprechenden polynomialen Regressor k-fache Kreuzvalidierung auf den Daten durchführen. Für jeden der Kreuzvalidierungsblöcke soll der mittlere quadratische Fehler (mean squared error, MSE) berechnet werden. Die Funktion gibt das arithmetische Mittel aller MSE Werte zurück.

Zuletzt sollen Sie eine Funktion `find_best_poly_model` implementieren, die systematisch polynomiale Regressoren bis Grad 9 mit Kreuzvalidierung testet und den Polynomgrad zurückgibt, mit dem die beste Performance erreicht wurde.

Mit den implementierten Funktionen können Sie nun für die Datensätze `dataset0.csv`, `dataset1.csv` und `dataset2.csv` den optimalen Polynomgrad bestimmen. Trainieren Sie im Anschluss je ein Modell auf allen Trainingsdaten mit dem bestimmten, optimalen Polynomgrad. Visualisieren Sie das trainierte Modell und die Trainingsdaten und geben Sie die Visualisierung bitte mit ab (im Notebook oder PDF).

Bitte beantworten Sie die Fragen bis zum **13. November**. Ihren Python-Code, sowie Visualisierungen können Sie in Form eines Jupyter Notebooks oder PDFs hochladen.