Gaussian Process Report

Budjan, Haas, Klumpp, Reitze

19 Februar 2019

Einleitung

Für unser Paket zu Gausschen Prozessen hatten wir zusammengefasst zwei Aufgaben die wir meistern mussten. Erstens das schreiben zweier "regression functions" zur Vorhersage des Prozesses an anderen Punkten und zweitens die Bestimmung der richtigen Kovarianzfunktion für den Prozess.

Generelle Paketstruktur

Wir haben uns dazu entschieden, die Regressionen in R6 Klassen zu schreiben. Grund hierfür war die leichte Erstellung von schreibgeschützen Bereichen. Dies erschien sinnvoll, da die Algorithmen einige Rechenaufwendige Schritte beinhalten die nur einmal ausgeführt werden sollten (z.B. Cholesky Zerlegung) und somit beim Erstellen der Klasse ausgeführt werden mussen, wir aber den Nutzer davor schützen wollten beim Hinzufügen von Daten eine neue Berechnung der implizit genutzten Variablen in den Algorithmen zu vergessen.

Bei der Bestimmung der Kovarianzfunktion war dies hingegen nicht nötig. Hier ist unser Algorithmus als einfache Funktion implementiert. Hier haben wir Metaprogrammierung genutzt um Kovarianzfunktionen mit unterschiedlich vielen Parametern zu optimieren. Außerdem Error Handler um die Fehleranfälligkeit der Choleskyzerlegung und beim Invertieren abzufangen.

Hier was zu den Plots

Regression

Wie bereits erwähnt haben wir uns in diesem Teil für die Implementierung mit R6 Klassen entschieden. Uns erschien es als sinnvoll, dass der Nutzer einerseits einige Möglichkeiten für häufig genutzte Kovarianzfunktionen zur Verfügung hat, bei welchen er nurnoch die von ihm erwarteten Parameter eingeben muss, aber andererseits auch die Möglichkeit hat, der Klasse eine beliebige eigene Kovarianzfunktion übergeben kann. Hier muss allerdings erwähnt werden, dass für die Funktionsfähigkeit des Algorithmus eine positiv definite Kovarianzmatrix nötig ist, sodass in Wahrheit nur geeignete Funktionen eine Regression liefern können.

Als dritte Option haben wir die Möglichkeit implementiert, der Klasse eine Liste von Kovarianzfunktionen zu übergeben, sodass vor der Vorhersage durch Optimierung der Hyperparameter automatisch die am besten geeignete Kovarianzfunktion auswählt wird.

Optimierung der Hyperparameter

Die Funktion zur Optimierung der Hyperparameter - im Folgenden als Fit-Funktion bezeichnet - beruht auf den Abschnitten zur Bayessian Model Selection durch den Marginal Likelihood. Wir haben uns für diese Methode entschieden, da nach Studium der Quelle Cross-validation als das numerisch aufwändigere Verfahren erschien.

Allerdings mussten wir das Verfahren an einigen Stellen leicht anpassen um, wie bereits erwähnt, einige Probleme des Algorithmus abzufangen. Als erstes mussten wir uns mit den Eigenheiten der einzelnen Kovarianzfunktionen befassen. Bei konstanter und linearer Kovariarianz ist die ?Matrix der Ableitungen? im Allgemeinen nicht invertierbar. Daher mussten wir hier die Optimierung anpassen **?Wie funktionierte das?**. Außerdem mussten wir Sicherstellen, dass bei polynomiellen Kovarianzfunktionen die Exponenten aus

der Menge der natürlichen Zahlen gewählt werden und mussten hier somit über diskrete Parameter optimieren. Außerdem beinhaltet der Algorithmus wie bereits erwähnt das mehrfache Ausführen der Cholesky-Zerlegung innerhalb der Optimierung, welche leider nicht immer ausgeführt werden kann. Um dieses Problem zumindest einzugrenzen haben wir uns entschieden ein Error Handling einzubauen, das bei einem Fehler in einem Iterationsschritt die besten bisher berechneten Parameter zurückgibt zusammen mit einem Hinweis, dass eine weitere Berechnung nicht möglich ist.

Klassifizierung

Plots

Eigenanteil

Abschließende Bemerkungen