ÚLOHA 6
Regularizace

Zadáno na cvičení: 8
Mezní termín: 3.12. 2020
Maximální počet bodů: 10
Nepovinná úloha

Zadání

Stáhněte si archiv *regularization.zip* ze stránky *Regularizace*. Následující soubory zkopírujte z předchozích úkolů (soubory označené ⋆ budete ještě upravovat):

- *lib/featureNormalize.m* – škálování příznaků
- *lib/gradientDescent.m* – gradientní sestup
- *lib/linReg/getLinearRegression.m* – vytvoří strukturu reprezentující algoritmus lineární regrese
- *lib/linReg/linReCost.m⋆* – cenová funkce lineární regrese včetně regularizace
- *lib/linReg/linRegPredict.m* – hypotéza lineární regrese
- *lib/linReg/normalEqn.m⋆* – normální rovnice včetně regularizace
- *lib/logReg/getLogisticRegression.m* – vytvoří strukturu reprezentující algoritmus logistické regrese
- *lib/logReg/logRegCost.m⋆* – cenová funkce lineární regrese včetně regularizace
- *lib/logReg/logRegPredict.m* – hypotéza lineární regrese
- *lib/logReg/sigmoid.m* – funkce sigmoidy

Pro konkrétní úlohy budete doplňovat následující soubory:

- *biasVarianceDemo.m* – hlavní skript první části
- *logisticRegularized.m* – hlavní skript druhé části
- *learningCurve.m* – příprava dat pro vizualizaci učící křivky
- *validationCurve.m* – příprava dat pro vizualizaci validační křivky

1 Regularizovaná regrese

1. **Doplňte regularizační člen do cenové funkce lineární a logistické regrese a do normálních rovnic.** V této části budete upravovat soubory v *lib/linReg* a *lib/logReg*.

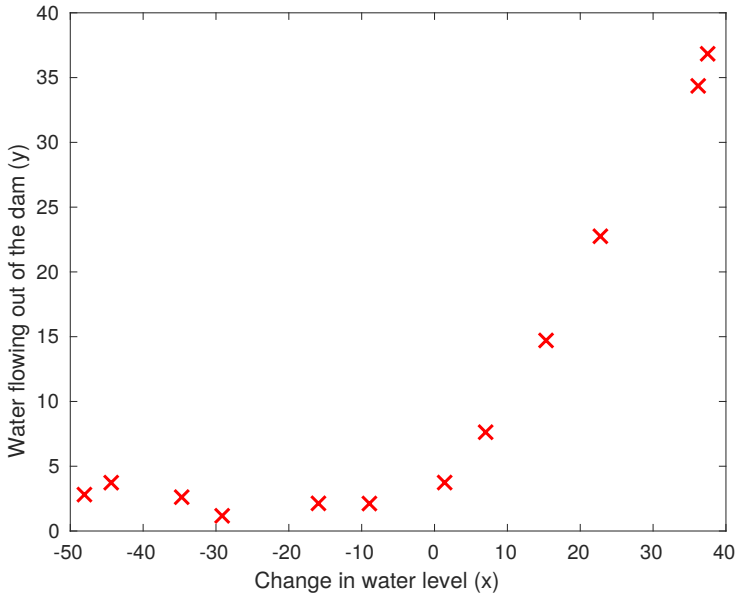
2 Přeučení a nedoučení

První úloha demonstruje problém přeučení a nedoučení na jednoduché polynomiální regresi.

Vstupní data

Máme k dispozici historické údaje o závislosti množství vody, které vyteče z přehrady na zvýšení hladiny vody.
Data jsou rozdělená na tři množiny:

1. **Trénovací data** – Využívají se k natrénování klasifikátoru.
2. **Validační data** – Využívají se k ladění hyperparametrů (alfa, lambda ...).
3. **Testovací data** – Využívají se pro vyhodnocení úspěšnosti klasifikátoru.



Obrázek 1: Vizualizace dat.

Úkoly

1. **Naprogramujte funkci *learningCurve* pro výpočet dat pro učící křivku.**

   Učící křivka je závislost chyby (například hodnoty cenové funkce) trénovacích a validačních dat na počtu **trénovacích** dat.

2. **Naprogramujte funkci *validationCurve* pro výpočet dat pro validační křivku.**

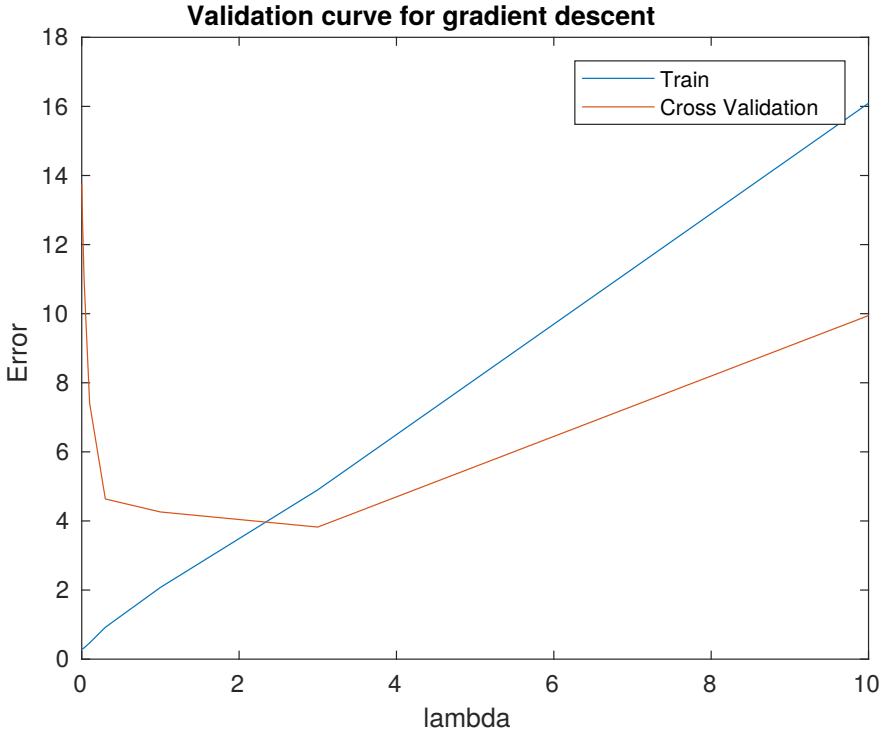
   Validační křivka je závislost chyby trénovacích a validačních dat na regularizačním koeficientu λ a slouží k nalezení optimální hodnoty λ.

3. **Vyzkoušejte si, co s učící křivkou dělá změna hodnoty λ.**

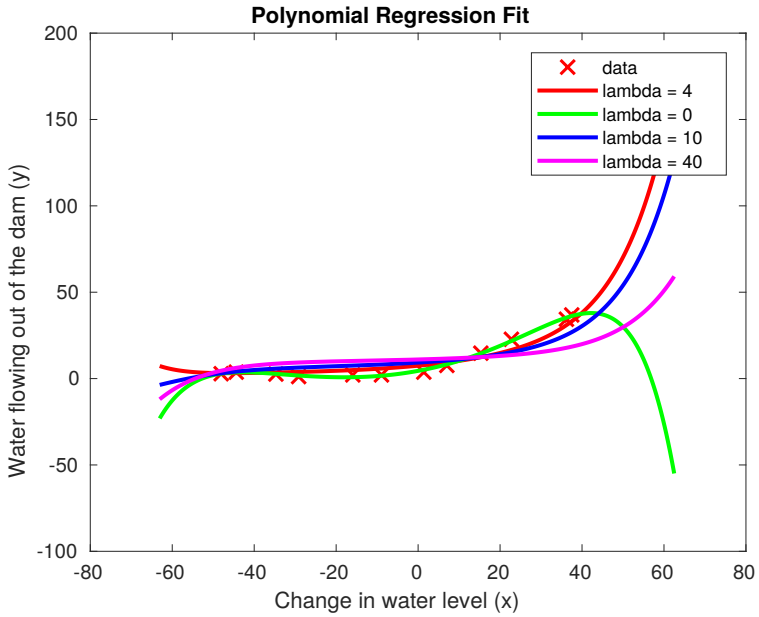
   Nemusíte se dostat k nějakému výsledku, jen se zamyslete, co vizualizace vlastně říkají o průběhu učení.

4. **Pokuste se zdůvodnit průběhy chyb v jednotlivých grafech.** Stačí krátký komentář k jednotlivým grafům.

Obrázek 2: učící křivka



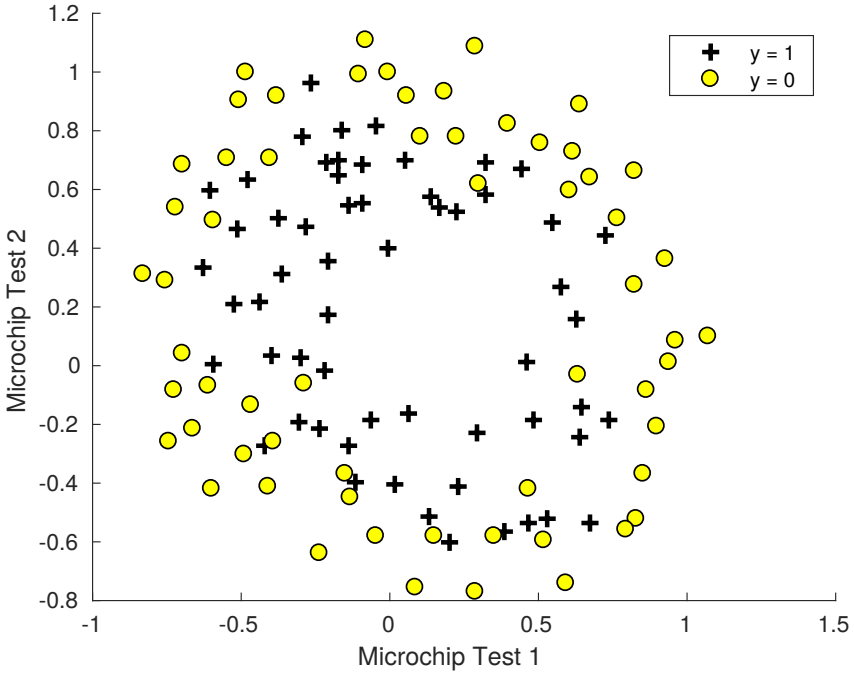
Obrázek 3: validační křivka



Obrázek 4: hypotéza pro různé hodnoty lambda

3 Polynomiální klasifikace

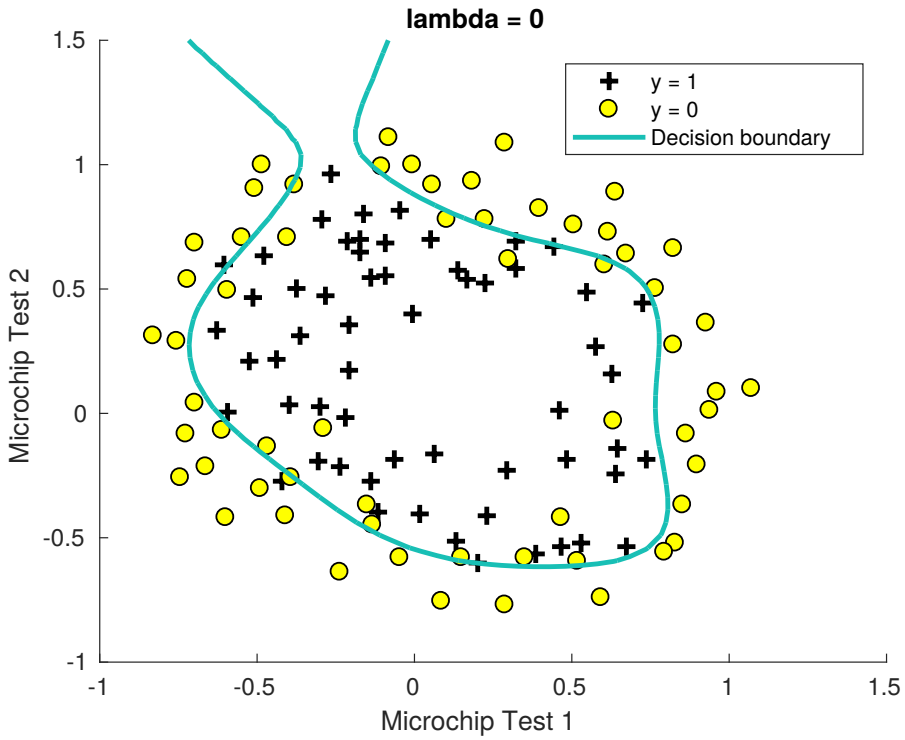
V této části pouze pustíme algoritmus regularizované logistické regrese. Testovací skript používá polynomiální logistickou regresi na klasifikaci lineárně neseparabilních dat.



Obrázek 5: vizualizace dat

Úkoly

1. **Vykreslete rozhodovací hranici pro několik různých hodnot lambda**. V této části budete upravovat script *logisticRegularized.m*.



Obrázek 6: Rozhodovací hranice bez regularizace