Carlos Fernandez-Granda

Research focus: Machine learning for high-dimensional signal processing

Problems of interest: Denoising, segmentation, and classification of images, video and sensor data

Applications: Medicine, scientific imaging, climate

NYU COURANT INSTITUTE OF MATHEMATICAL SCIENCES

NYU DATA SCIENCE

ML models for regression

Motivation

Data-driven sub-grid parameterization

Estimate *missing term* in climate model from available coarse-scale quantities

Motivation

Data-driven sub-grid parameterization

Estimate *missing term* in climate model from available coarse-scale quantities

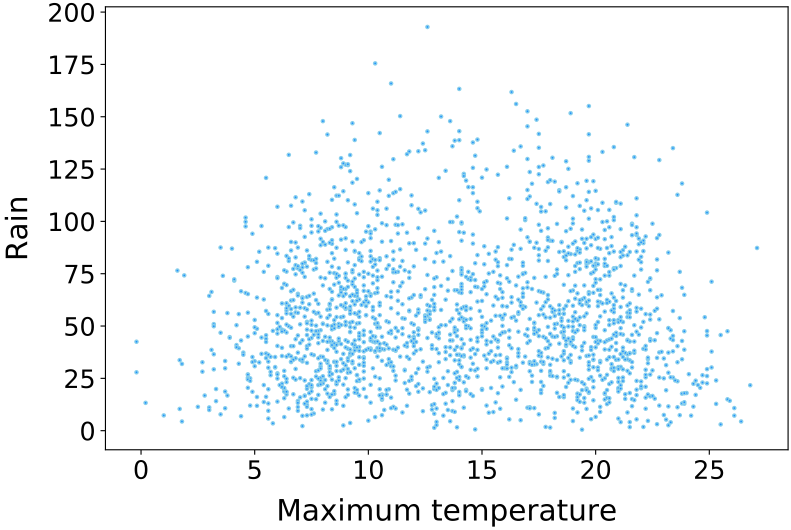
This is a regression problem!

Regression

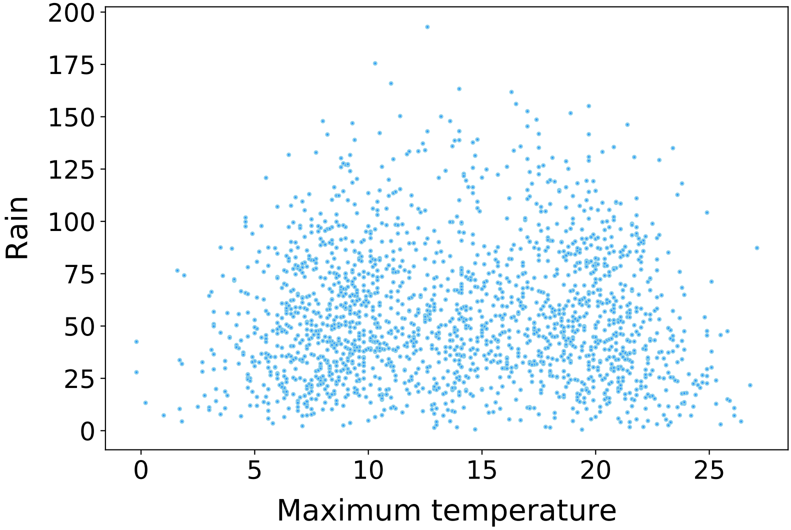
Goal: Estimate response (or dependent variable)

Data: Several observed variables, known as features (or covariates, or independent variables)

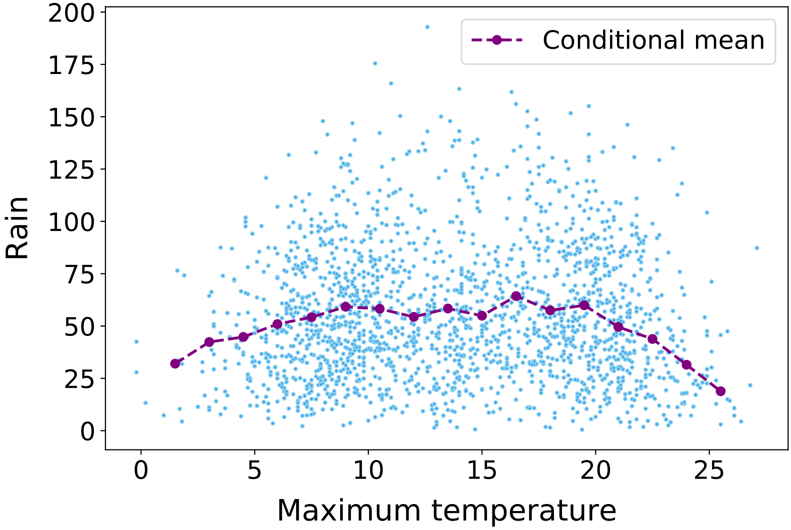
Toy regression problem



Optimal estimate in mean squared error?



Conditional mean



Are we done?

Why don't we just compute the conditional mean?

Are we done?

Why don't we just compute the conditional mean?

Assume we have 5 features with 100 possible values each

How many conditional averages do we need to estimate?

Are we done?

Why don't we just compute the conditional mean?

Assume we have 5 features with 100 possible values each

How many conditional averages do we need to estimate? 10^{10}!

Are we done?

Why don't we just compute the conditional mean?

Assume we have 5 features with 100 possible values each

How many conditional averages do we need to estimate? 10^{10}!

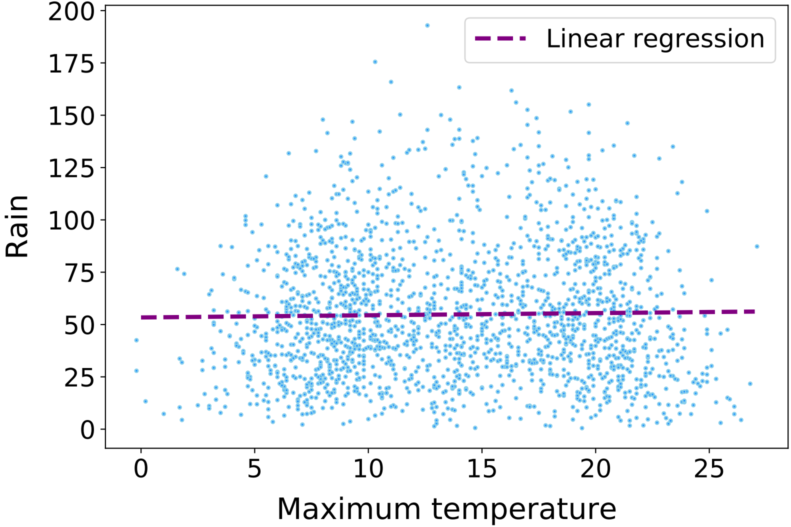
This is known as the curse of dimensionality

Linear regression

Assumption: Relationship between response and features is linear

Gradient of the regression function is constant

Estimating rain from temperature



Nonlinear regression

- *Handcrafted* nonlinear features + linear model

Nonlinear regression

- *Handcrafted* nonlinear features + linear model
- Kernel methods

Nonlinear regression

- *Handcrafted* nonlinear features + linear model
- Kernel methods
- Neural networks

Nonlinear regression

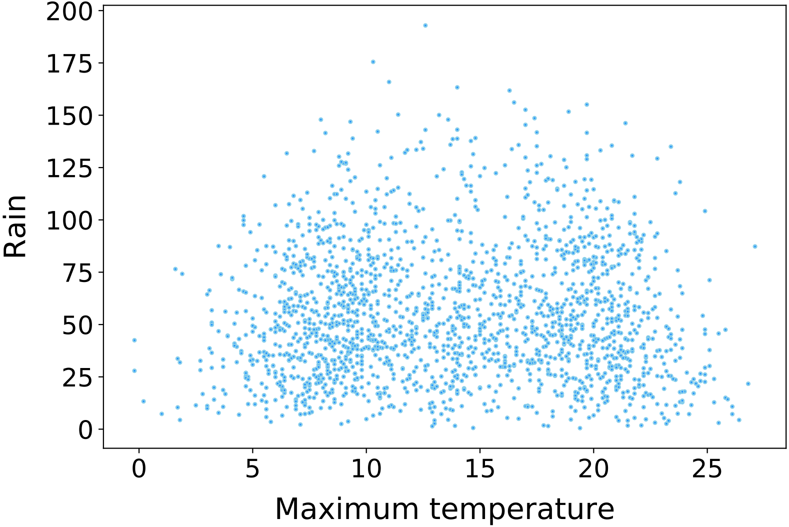
- *Handcrafted* nonlinear features + linear model
- Kernel methods
- Neural networks
- Tree-based methods

Regression trees

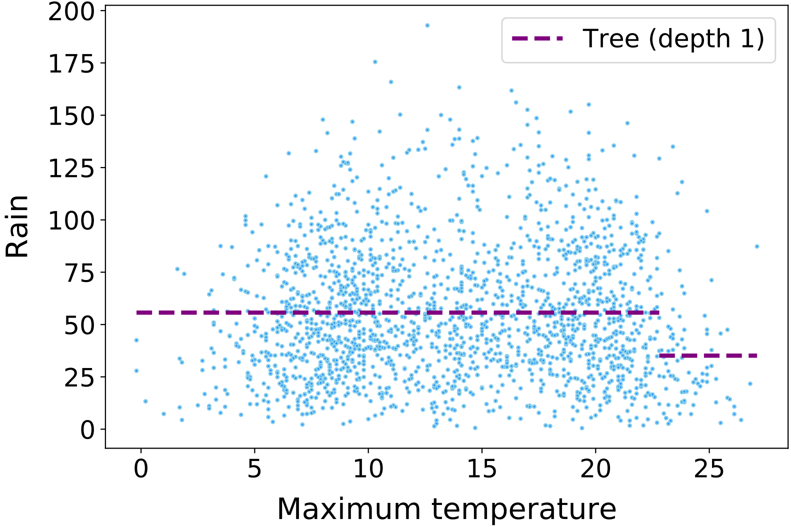
Partition feature domain recursively

Assign estimate to each set in the partition

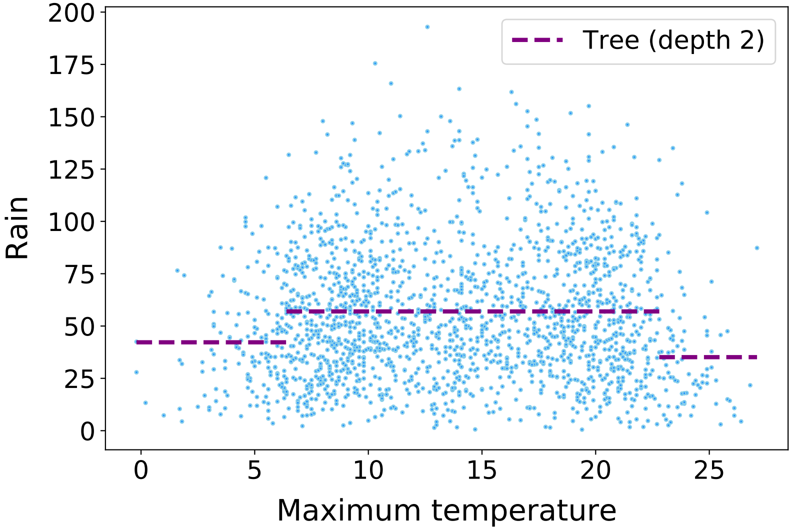
Estimating rain from temperature



Estimating rain from temperature



Estimating rain from temperature



Regression trees

Good news:

Bad news:

Regression trees

Good news: Interpretable

Bad news:

Regression trees

Good news: Interpretable

Bad news: Tend to overfit

Ensembling

General principle in machine learning:

Averaging output of different models is very helpful

Why?

Ensembling

General principle in machine learning:

Averaging output of different models is very helpful

Why? Errors approximately cancel out if models are *independent*

Naive ensembling: Bagging

Idea: Build many trees and average them

To build the trees randomly sample from training data (bootstrapping)

Naive ensembling: Bagging

Idea: Build many trees and average them

To build the trees randomly sample from training data (bootstrapping)

Problem: Tree outputs are very correlated

Random forests

Key idea: Improve bagging via randomization

Random forests

Key idea: Improve bagging via randomization

Use **random subset** of features at each node while building trees to avoid correlations

Random forests

Key idea: Improve bagging via randomization

Use **random subset** of features at each node while building trees to avoid correlations

Good news: Better generalization

Random forests

Key idea: Improve bagging via randomization

Use **random subset** of features at each node while building trees to avoid correlations

Good news: Better generalization

Bad news: Less interpretable

Gradient boosting (e.g. XGBoost)

Key idea: Choose trees sequentially to minimize cost function (e.g. mean squared error)

Gradient boosting (e.g. XGBoost)

Key idea: Choose trees sequentially to minimize cost function (e.g. mean squared error)

Regularization is often applied to avoid overfitting

Gradient boosting (e.g. XGBoost)

Key idea: Choose trees sequentially to minimize cost function (e.g. mean squared error)

Regularization is often applied to avoid overfitting

Good news: Better generalization than bagging (and often random forests)

Gradient boosting (e.g. XGBoost)

Key idea: Choose trees sequentially to minimize cost function (e.g. mean squared error)

Regularization is often applied to avoid overfitting

Good news: Better generalization than bagging (and often random forests)

Bad news: Also not very interpretable

Empirical performance

XGBoost typically outperforms other ML methods (including deep networks) for real-world problems with up to hundreds of features

Empirical performance

XGBoost typically outperforms other ML methods (including deep networks) for real-world problems with up to hundreds of features

But tree-based methods do not scale to higher-dimensional signals (images, video, audio...)