Decision Boundaries, Modeling Considerations (Reading: [17.8])

Learning goals:

- Understand decision boundaries, multiclass classification, and regularization for log reg.
- Learn a few new techniques for diagnosing and improving models.

UC Berkeley Data 100 Summer 2019
Sam Lau

(Slides adapted from John DeNero)

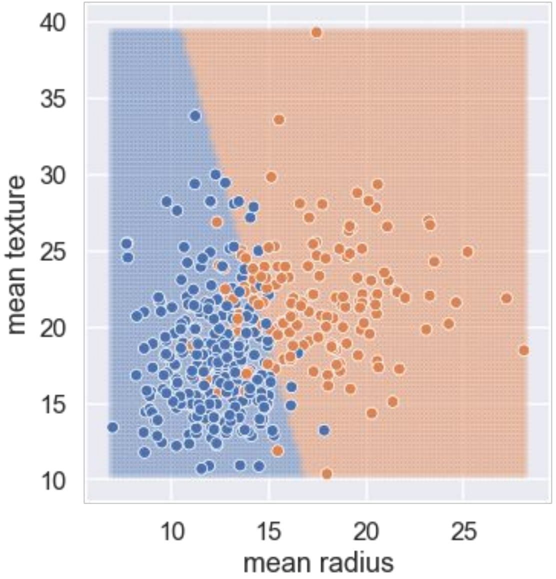
Announcements

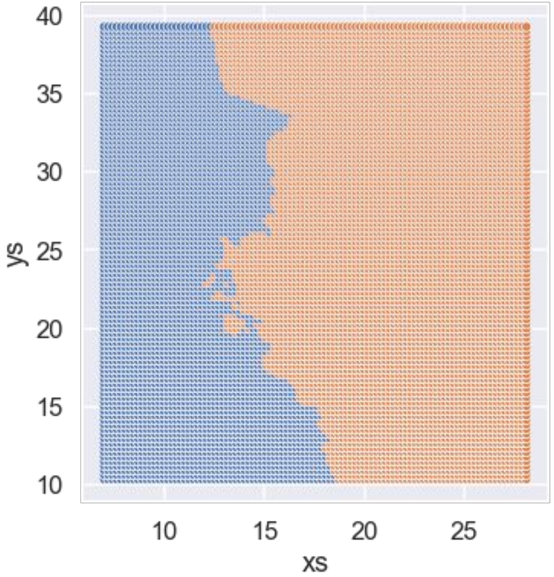
- Project 2 out
  - Due next Tuesday, Aug 5.
- Small group tutoring: tinyurl.com/d100-tutor-week6
- Manana OH today cancelled (covering Leo’s section)
- Sam will be covering Leo’s section tomorrow

Decision Boundaries

The **decision boundary** of a classifier is the set of points where the classifier changes its prediction.

Plotting in two dimensions useful for understanding model.





A complex decision boundary can indicate overfitting.

(Demo)

Linearly Separable Data

- A logistic regression model will always output a linear decision boundary. Why?
  - Use feature eng to model complex boundaries.
- If logistic regression can find a decision boundary that perfectly splits the data, the data are **linearly separable**.
  - Doesn't imply that population data are linearly separable, though!
- Without regularization, GD will never converge on linearly separable data. Why not?

(Demo)

Multiclass Classification

Multiclass Classification

- Often have >2 classes to predict. E.g:
  - Will it be rainy, sunny, or cloudy tomorrow?
  - Which book will a user buy next? (Many classes!)
- Idea: Convert to binary classification by treating each class as rainy vs not rainy, sunny vs not sunny, etc.
  - Called the **One-vs-Rest strategy**
- This estimates one probability for each class. Then, final prediction is the class with the highest probability.

Regularization for Logistic Regression

Regularization

- L2 and L1 regularization work the same way for log reg:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta} \cdot \boldsymbol{x}) \qquad \text{where } \sigma(t) = \frac{1}{1 + \exp(-t)}$$

$$\text{Let } z_i = f_{\boldsymbol{\theta}}(\boldsymbol{X_i}).$$

$$L(\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{y}) = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log z_i + (1 - y_i) \log(1 - z_i)] + \lambda \sum_{j=1}^{p} \theta_j^2$$

You Try:

Derive the BGD update rule for logistic regression with L2 regularization.

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta} \cdot \boldsymbol{x}) \qquad \text{where } \sigma(t) = \frac{1}{1 + \exp(-t)}$$

Let $z_i = f_{\boldsymbol{\theta}}(\boldsymbol{X_i})$.

$$L(\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{y}) = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log z_i + (1 - y_i) \log(1 - z_i)] + \lambda \sum_{j=1}^{p} \theta_j^2$$

You Try:

Once you compute average gradient, the update rule follows.

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta} \cdot \boldsymbol{x}) \qquad \text{where } \sigma(t) = \frac{1}{1+\exp(-t)}$$

Let $z_i = f_{\boldsymbol{\theta}}(\boldsymbol{X_i})$.

$$L(\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{y}) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log z_i + (1-y_i)\log(1-z_i)\right] + \lambda \sum_{j=1}^{p} \theta_j^2$$

$$\nabla_{\boldsymbol{\theta}} L = -\frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma(\boldsymbol{X_i} \cdot \boldsymbol{\theta}))\boldsymbol{X_i} + 2\lambda\boldsymbol{\theta}$$

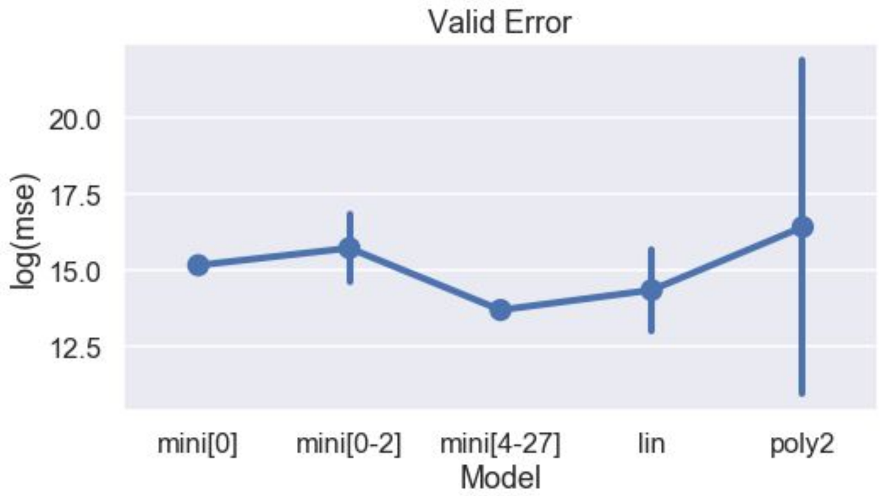
Discuss: How does changing λ influence gradient updates?

Break!
Fill out Attendance:
http://bit.ly/at-d100

Diagnosing Models

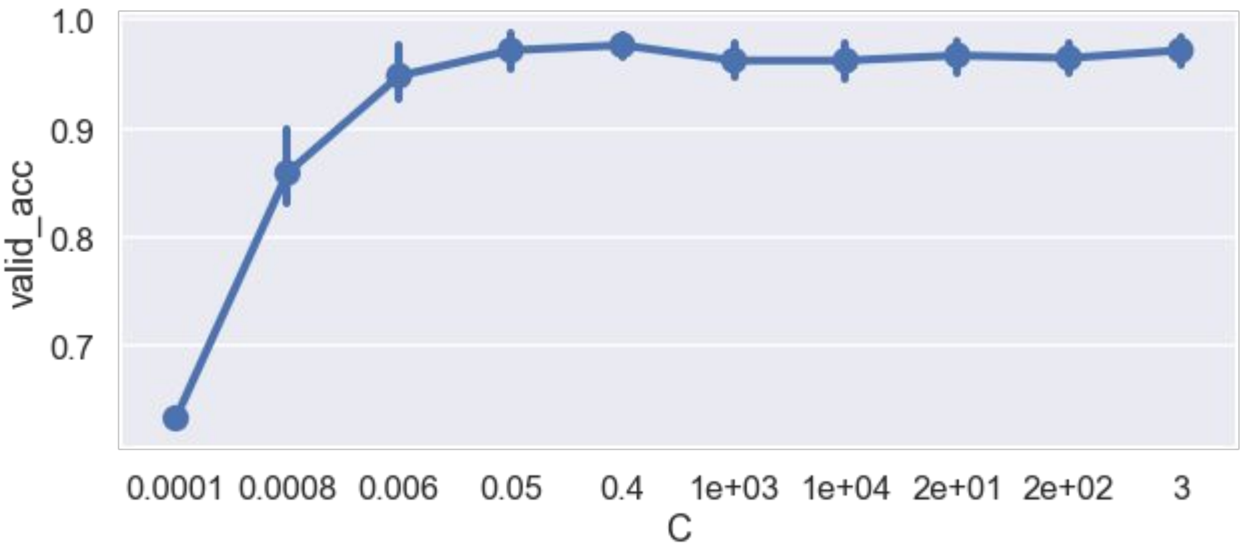
Validation Curves

- We've seen **validation curves**, which plot validation error across different models.



Validation Curves

- Similar patterns for regularization parameter λ:

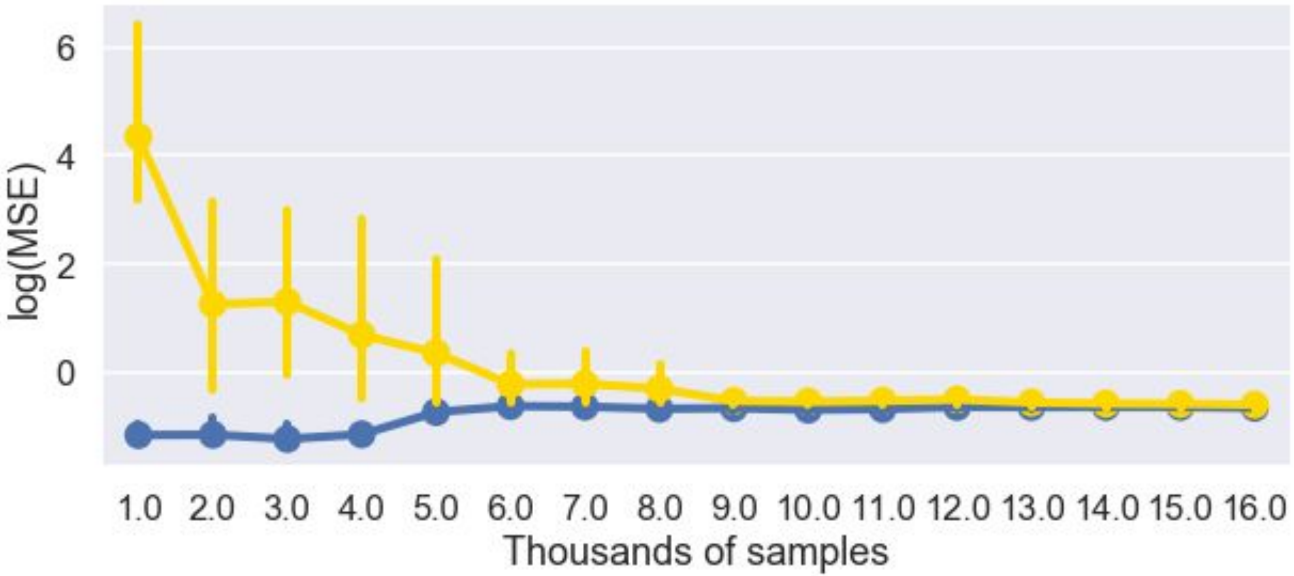


Note that accuracy is on the y-axis, not error.

Also note that in sklearn, higher values of C correspond to **lower values** of λ.

Learning Curves

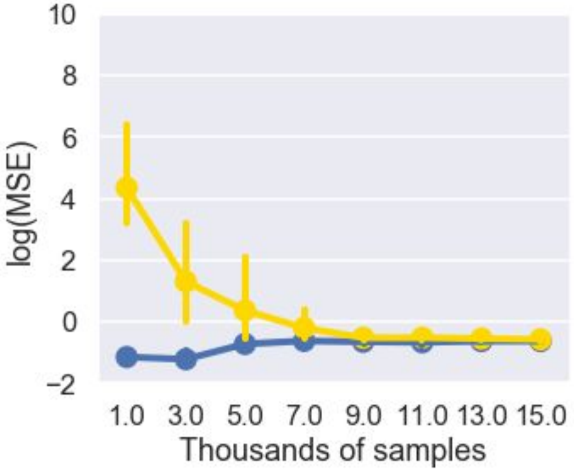
- **Learning curve**: plot training and validation error as we train on more data.
- Ideally, training and validation errors are close. Why?



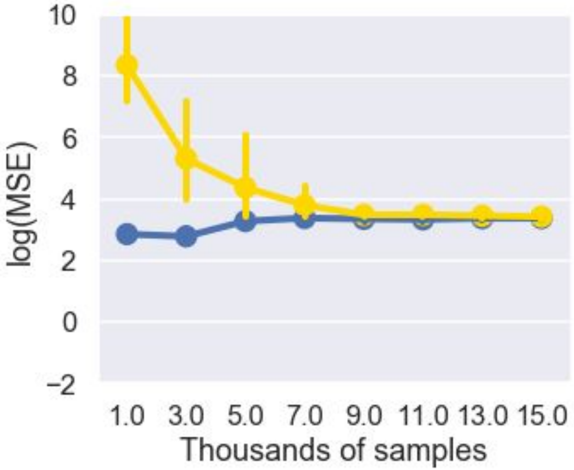
Learning Curves

- Learning curves help us decide whether model has too much variance for our sample size.

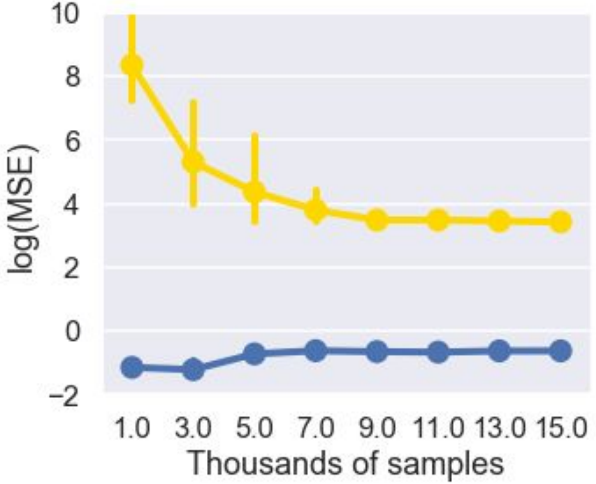
Low bias, low variance



High bias, low variance



High variance: collect more data or pick simpler model!



Improving Models

What can we tune?

We tune all of these with cross-validation.

- Feature engineering:
  - Adding polynomial features.
  - One-hot encoding.
  - Features using domain knowledge (these are usually the most helpful but also the hardest to come up with).
- Regularization
  - L2 or L1 regularization.
  - Tuning the regularization parameter λ.
- Models
  - Picking model with higher / lower complexity.

Subset Selection

- Removing features can make a better model. Why?
- Idea: remove poorly predictive features from model.
- Assume we have p features. How many combinations of features are there?
- Trying every combination is very expensive but will give the best results.
    - This algorithm is called best subset selection and is mostly used as an “ideal” algorithm for comparison.

Subset Selection

- Idea: Start with simplest model, then add features until validation error starts increasing.
- **Forward stepwise selection**:
    - Start with null model (0 features, only bias)
    - Use CV to decide next feature to add to model.
    - Repeat until validation errors start increasing.
- Fits $O(p^2)$ times instead of $O(2^p)$
- Doesn't always find the best combo but usually gets close.
- Backwards stepwise selection works in reverse.

Summary

- Decision boundaries visualize classifier's predictions
- Multiclass problems can be reduced to binary problems
- You should understand how to derive the regularized gradient descent update rule.
- Understand what validation and learning curves are useful for.
- In this class, feature engineering is the primary method of improving a model.