Regression Analysis using GraphLab Create

Krishna Sridhar (GraphLab Team)

Overview

- Intro to Regression Analysis?
- Predictive Analytics
- Linear & Logistic regression!
- Interpreting Results
- Feature Engineering
- Feature Selection
- Bag of Trix
- Demo

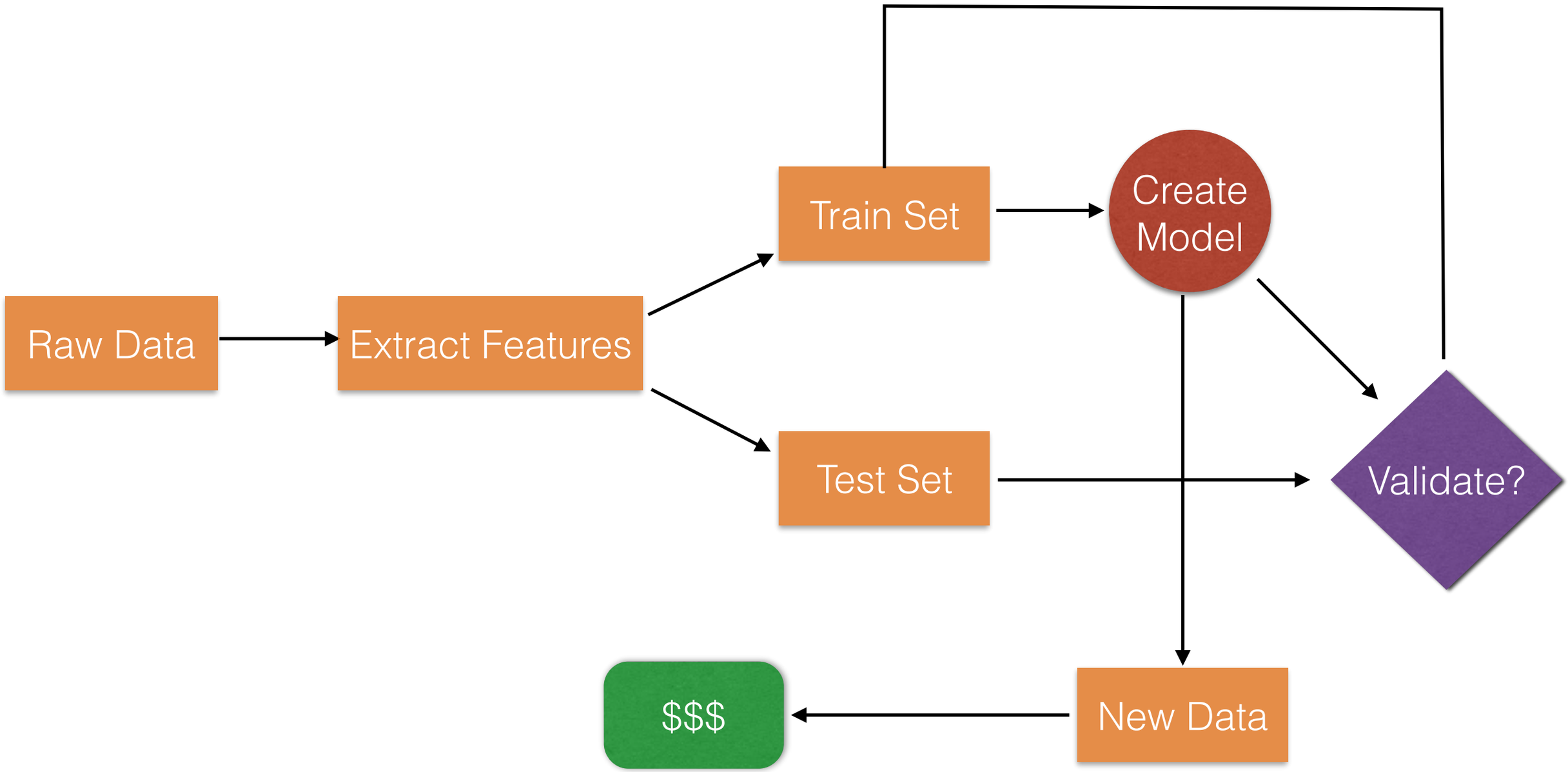
Predictive Analytics

The goal, in **predictive analytics** is to forecast or predict a target using historical data.

Examples

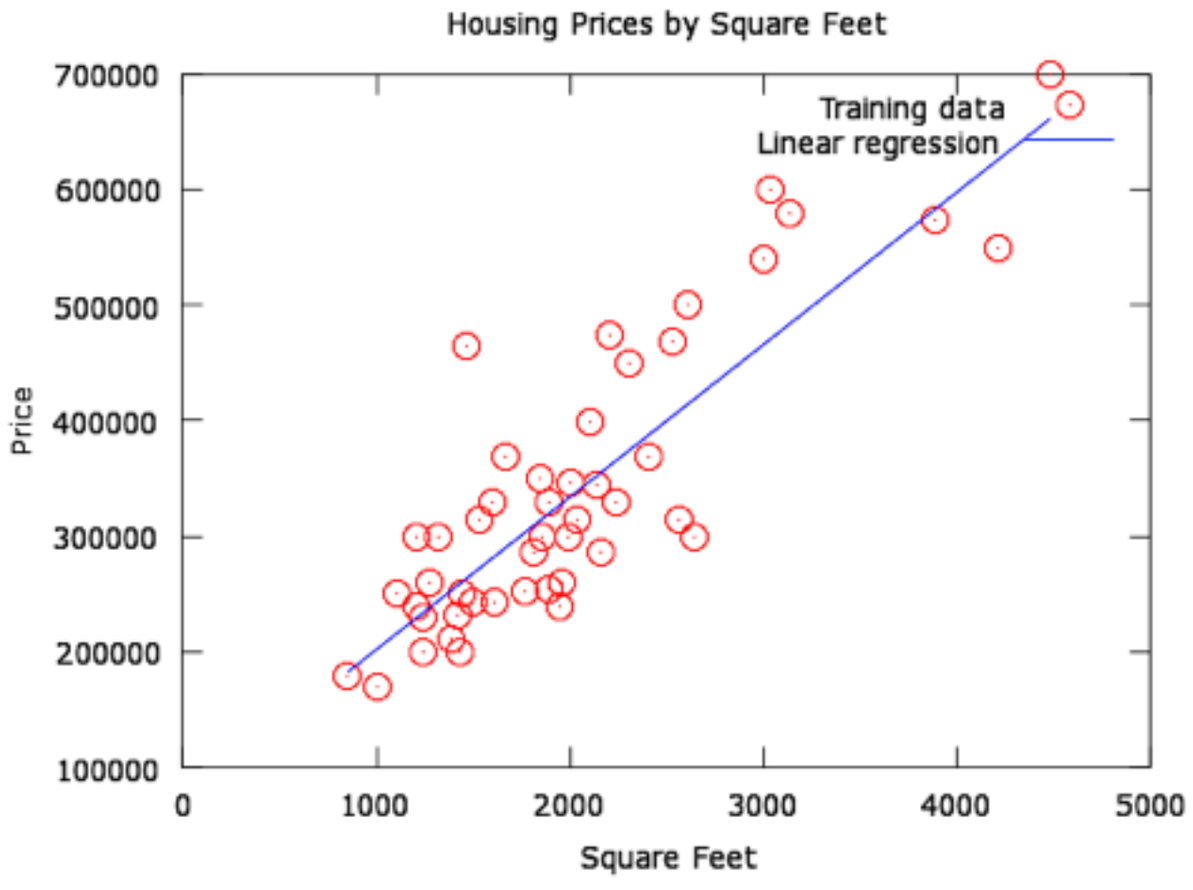
- What is the price of a house?
- Is a restaurant is good or not?
- Given a sentence, an I predict whether or not to say “That’s what she said”
- Are the Heat going to win the NBA title?

Workflow


Train Set
Create Model
Raw Data
Extract Features
Test Set
Validate?
$$$
New Data

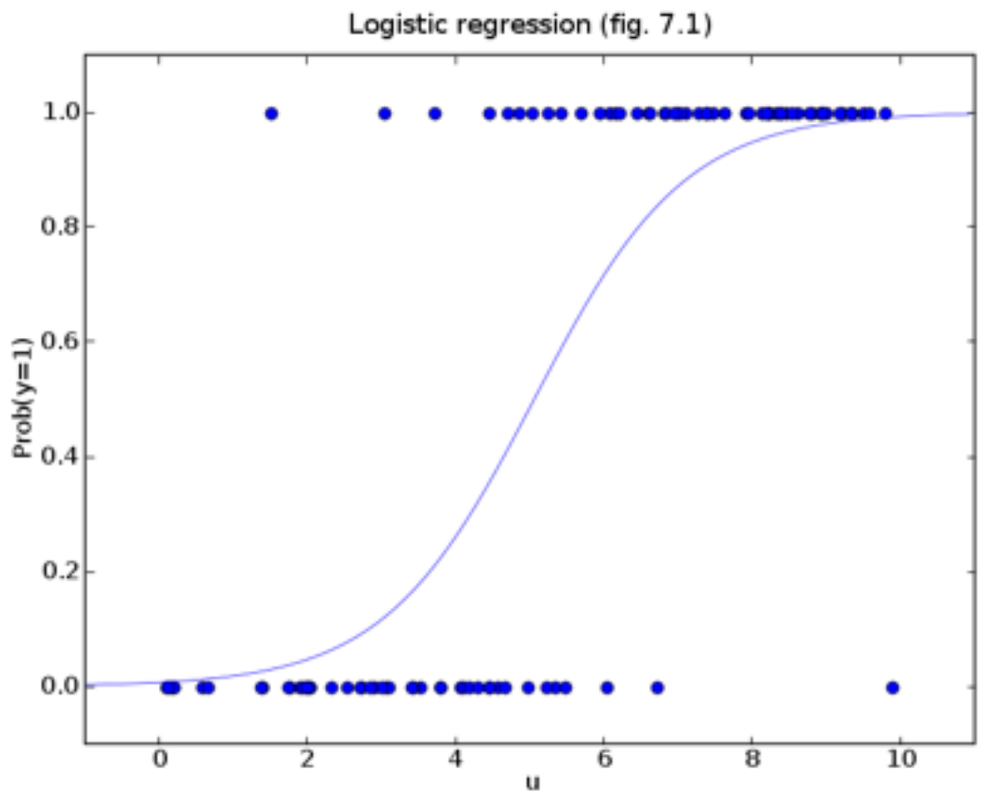
Predictive Analytics

Can I predict the price of a house given it's size?

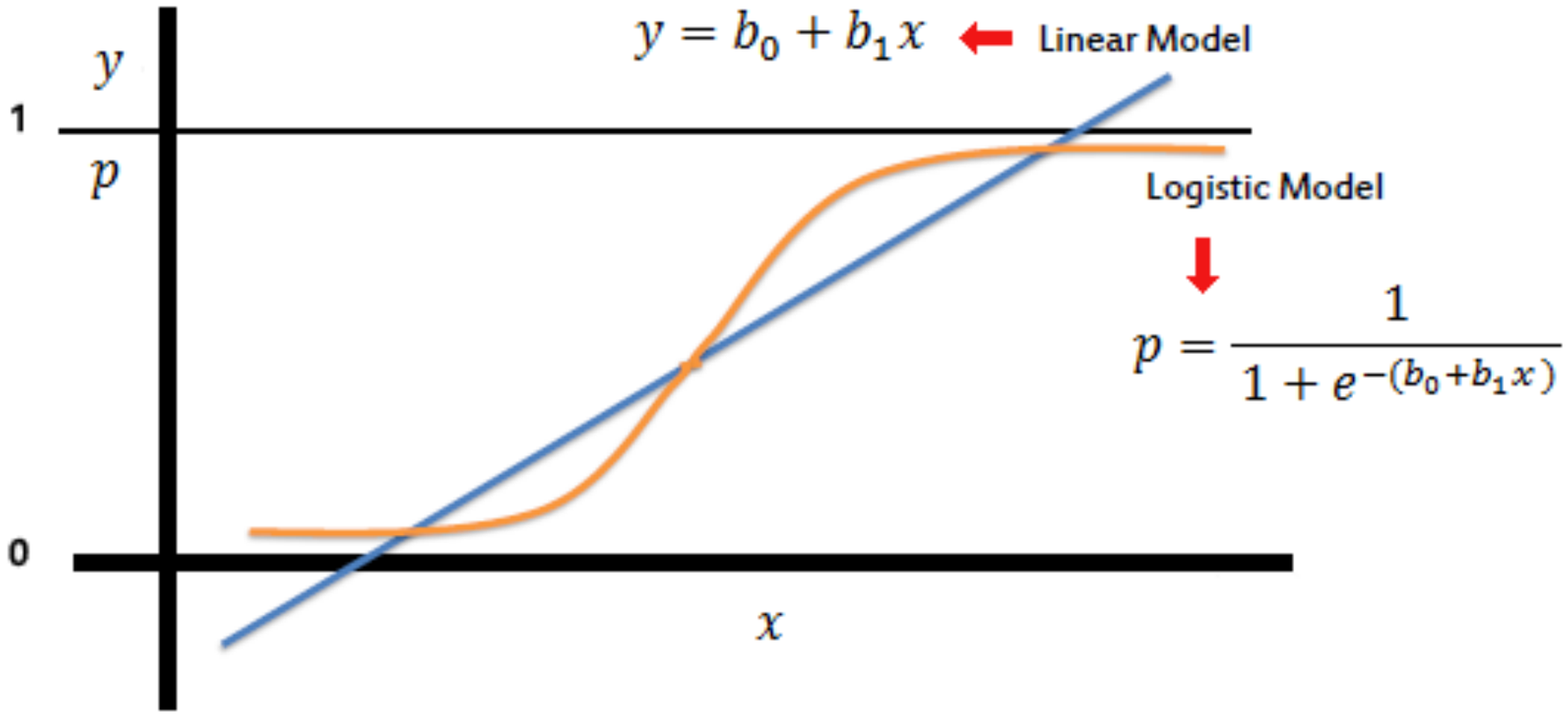


Predictive Analytics

Can I predict if the Heat are going to with the NBA title?



Linear vs Logistic



Linear Regression

$$y_i = \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots \beta_K x_{K,1} + \epsilon_i \quad \forall i \in \{1 \ldots n\}$$

Target Features Coefficients Noise/Error

- **Target**: Binary (classification) or continuous (regression)
- **Features**: The set of attributes that effect how a decision/prediction is made.
- **Coefficients**: A measure of how the feature is correlated with the prediction.
- **Noise:** Nobody is prefect.

Training Models

$$y_i = \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots \beta_K x_{K,1} + \epsilon_i \quad \forall i \in \{1 \ldots n\}$$

Target Features Coefficients Noise/Error

What is training?

Given some **historical data**, the process of model training tries to **find a model** which makes the **most accurate** predictions.

Training Models

$$\beta^* = \arg\min_{\beta} \|y - \beta^T x\|^2$$

Model

Historical Data

Most Accurate

What is training?

Given some **historical data**, the process of model training tries to **find a model** which makes the **most accurate** predictions.

Training Models

$$\beta^* = \arg\min_{\beta} \|y - \beta^T x\|^2$$

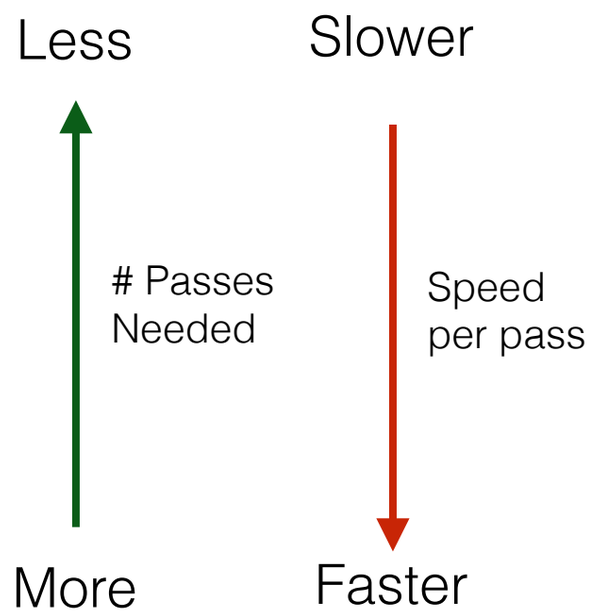
Model

Historical Data

Most Accurate

How does training happen?

- Second order methods (Newton)
- First order methods (Gradient Descent, L-BFGS)
- Stochastic Methods (SGD)



Reference: http://pages.cs.wisc.edu/~swright/nips2010/sjw-nips10.pdf

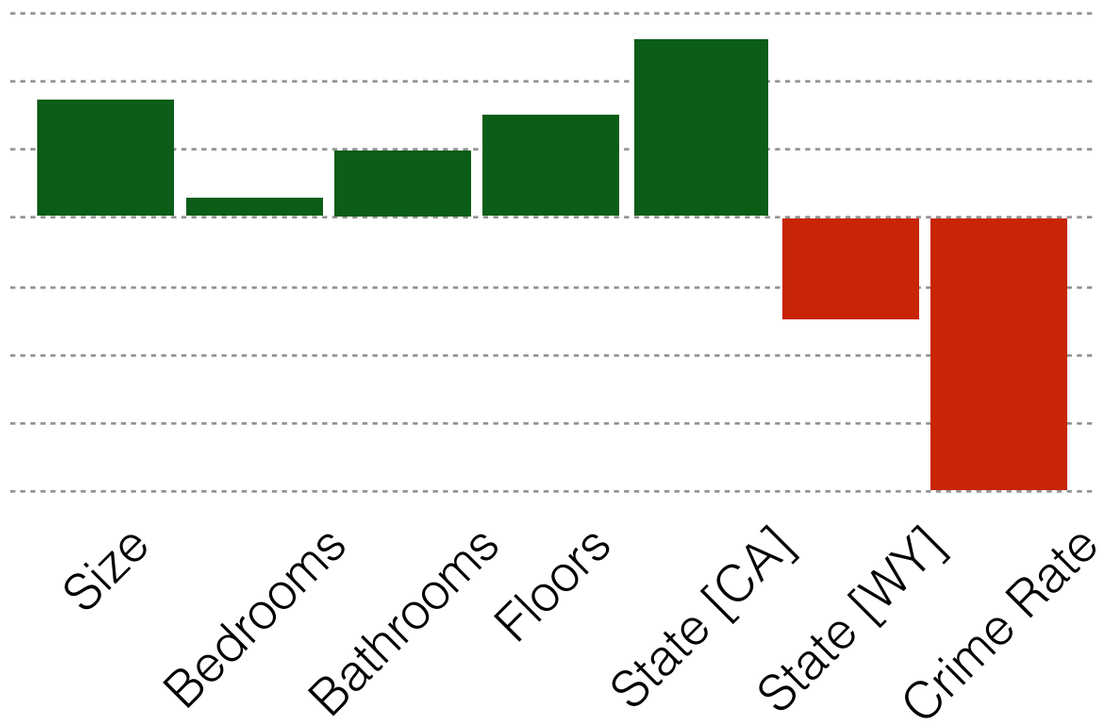
Interpreting Results

$$\beta^* = \arg\min_{\beta} \|y - \beta^T x\|^2$$

Model

Historical Data

Most Accurate



Regularization

$$\beta^* = \arg\min_{\beta} \left\| y - \beta^T x \right\|^2 + \lambda \left\| x \right\|^2$$

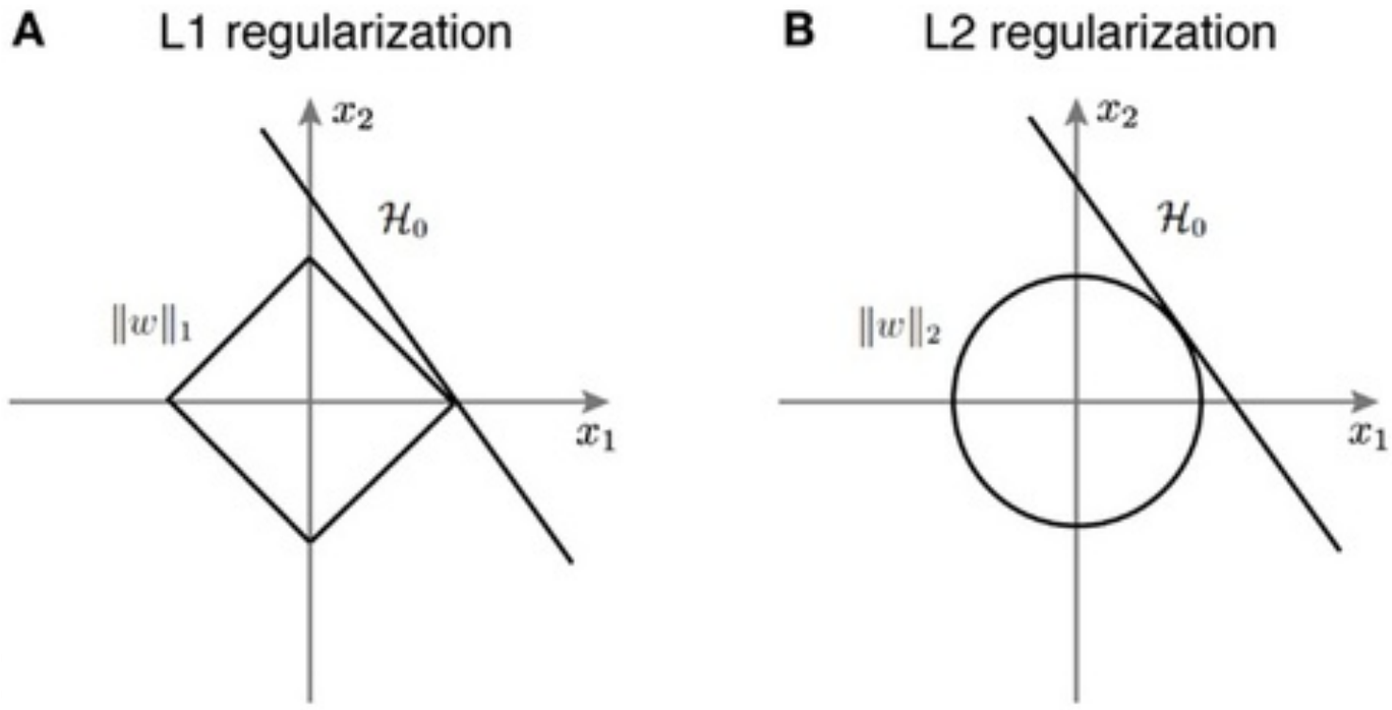
Model

Historical Data

Most Accurate

Reference: http://pages.cs.wisc.edu/~swright/nips2010/sjw-nips10.pdf

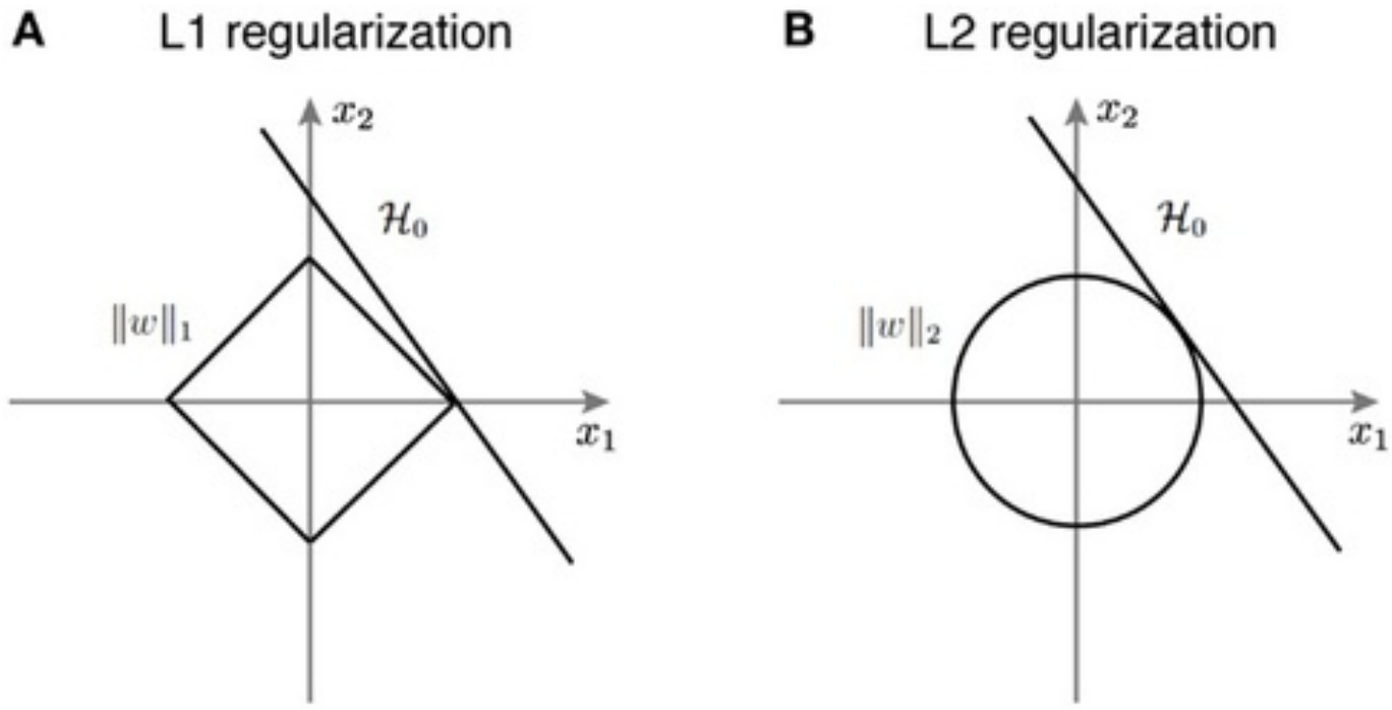
Regularization



L1 Penalty:

- Penalizes complex models.
- Creates models with lots of non-zero coefficients.
- Used for feature “selection”
- Suitable when you have more features than examples. (Overdetermined problems)

Regularization



L2 Penalty:

- Avoid overfitting model for training data.
- Stabilize the estimates especially when there's collinearity in the data.
- Shrinks coefficients towards zero.

Demo Time!

Feature Engineering

GraphLab Create supports

- Numeric Features
- Categorical features
- Dictionaries (Sparse Features)
- Lists (Dense features)

Feature Engineering

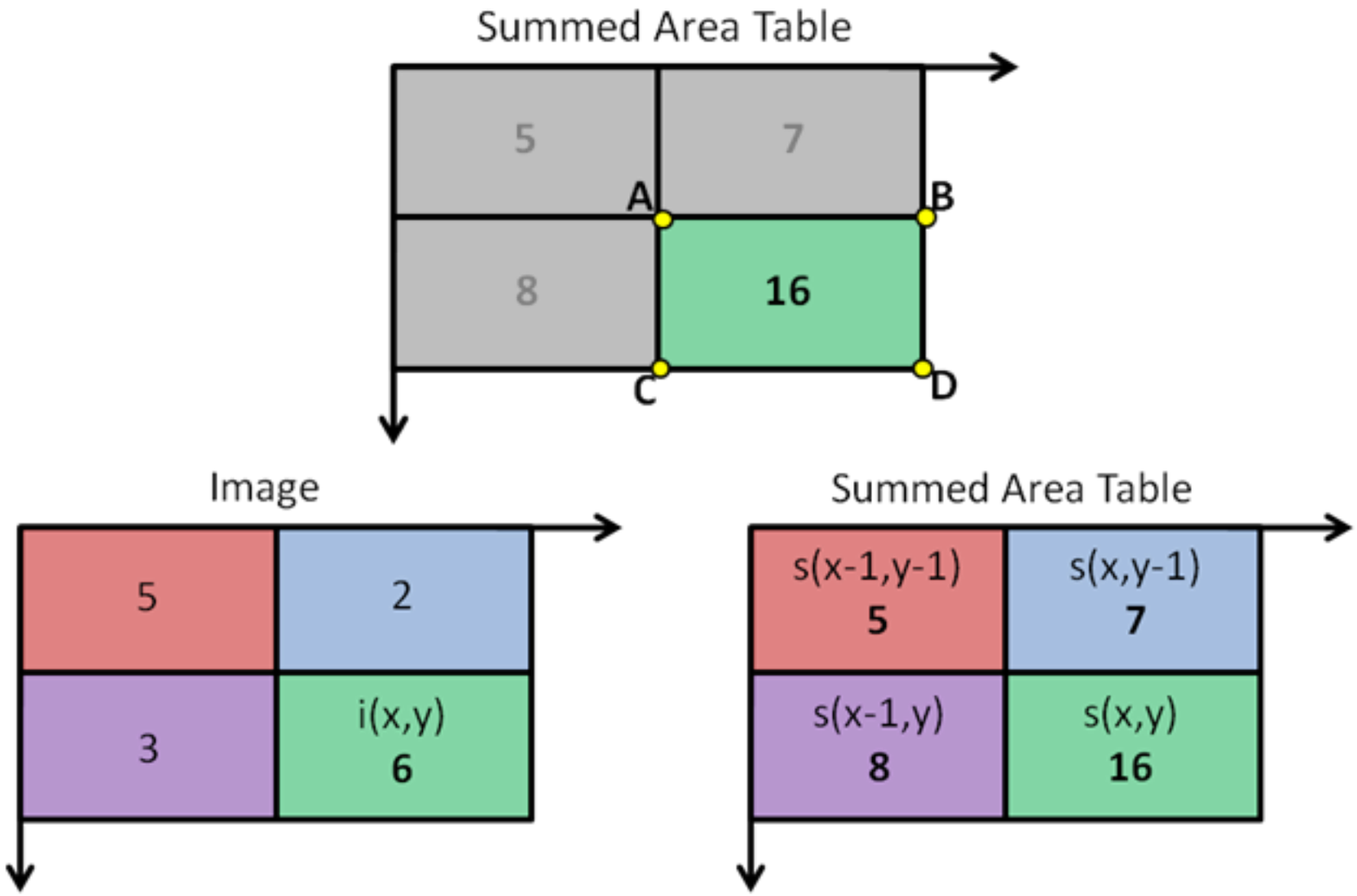
Categorical Features

- Eg: California (CA), Washington (WA), and Wisconsin (WI)

Category	Encoding
California	0 0
Washington	0 1
Wisconsin	1 0

Dense Vectors

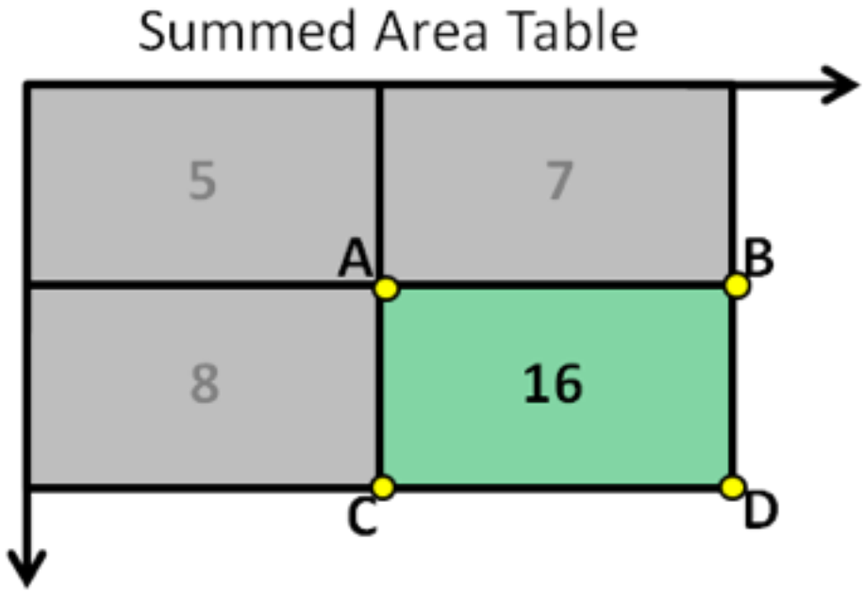
Example: Viola-Jones Face detector

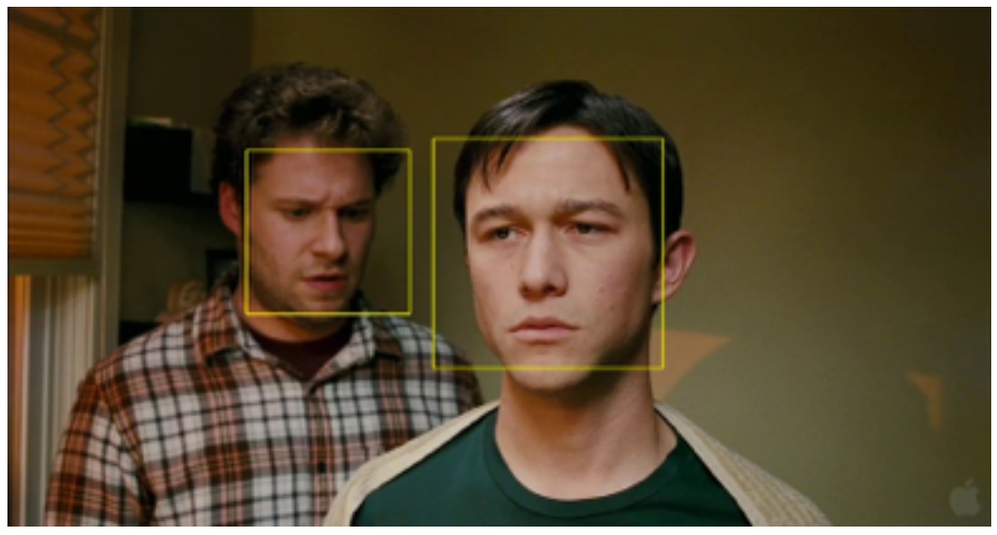


Source: http://bitsearch.blogspot.com/2012/12/overview-of-viola-jones-detector-in.html
Buzz Words: SIFT Features

Dense Vectors

Example: Viola-Jones Face detector





Encode input features as a sequence of dense vectors

Reference: Rapid object detection using a boosted cascade of simple features (2001), Viola and Jones

Sparse Features

Example: Bag of words

Doc. ID	Raw Text
1	In a bag of words, words corresponding …
2	I am a disco dancer …
3	That's what she said …

→

Doc ID.	Bag of Words
1	{'words': 2, 'in': 1 …
2	{'I': 1, 'am': 1 …
3	{'that': 1, 'what': 1 …

Encode input features as a sequence of sparse vectors

Buzz-Words: TF-IDF

Feature Engineering

- Try interaction terms.
  - “Male Dog” is better than “Male” or “Dog”
- Plot residuals vs Individual features.
- Transformation on features (log, binning etc.)
- Nonlinear feature generation
  - Random features for large-scale kernel machines - Rahimi & Recht
  - Count-min sketch and feature hashing (see http://hunch.net/~jl/projects/hash_reps/)

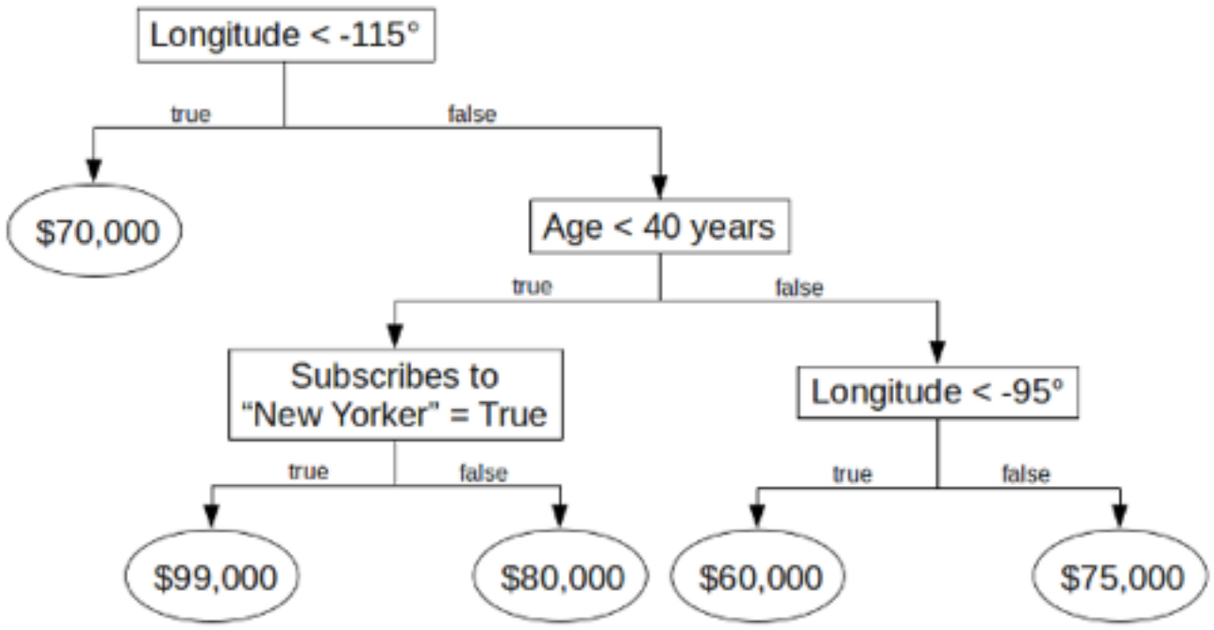
Vowpal Wabbit

Online Learning

- Learn as you go!
- Built to work with LOTS of features!
- **GraphLab Create's VW Wrapper Supports**
  - Linear, Logistic or Hinge Loss (SVM)
  - Matrix factorization and other modules using "command line args".

Reference: https://github.com/JohnLangford/vowpal_wabbit/wiki

Gradient Boosted Trees



GraphLab Create’s Wraps “EXtreme Gradient Boosting”

- Tree learning for Regression and Classification
- Linear model / LASSO

Reference: https://github.com/tqchen/xgboost

Demo Time!