Parallel Large Scale Feature Selection for Logistic Regression

Sameer Singh[1] Jeremy Kubica[2] Scott Larsen[2]
Daria Sorokina[3]

[1]Department of Computer Science
University of Massachusetts, Amherst

[2]Google Inc, Pittsburgh

[3]Department of Computer Science
Carnegie Mellon University

SIAM Data Mining, 2009

Outline

Outline

Logistic Regression

$$P(y = 1 \mid \vec{x}_i, \vec{\beta}) = \frac{e^{\vec{\beta}\cdot\vec{x}}}{1 + e^{\vec{\beta}\cdot\vec{x}}}$$

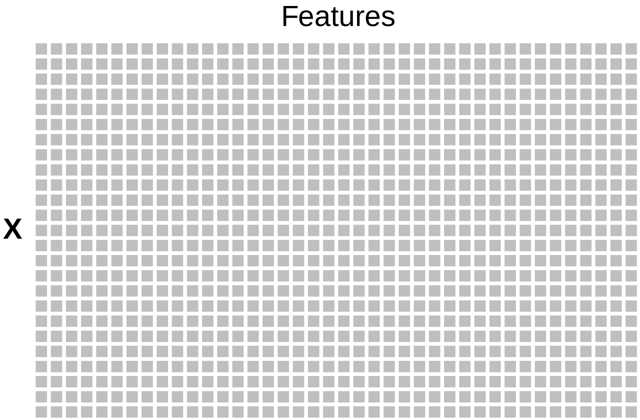
$$\vec{\beta} = \underset{\vec{\beta}}{\text{argmax}} \sum_{i=1}^{N} \Big(y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \Big)$$

Logistic Regression

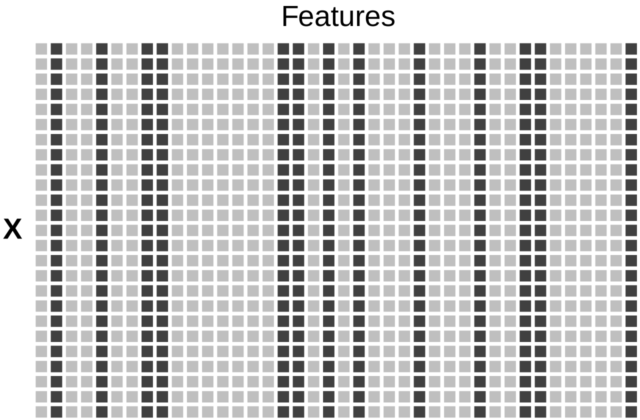
$$P(y = 1 \mid \vec{x}_i, \vec{\beta}) = \frac{e^{\vec{\beta}\cdot\vec{x}}}{1 + e^{\vec{\beta}\cdot\vec{x}}}$$

$$\vec{\beta} = \underset{\vec{\beta}}{\operatorname{argmax}} \sum_{i=1}^{N} \Big(y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \Big)$$

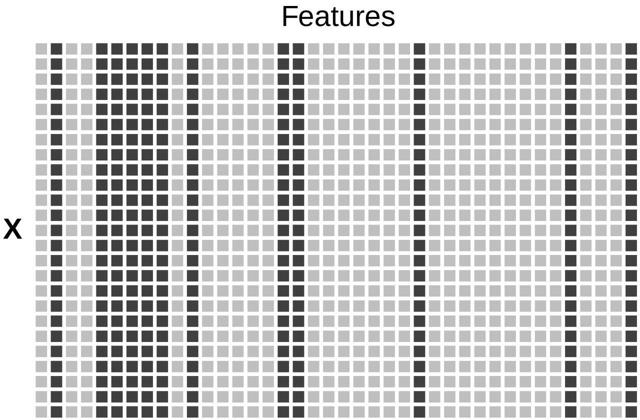
Feature Selection



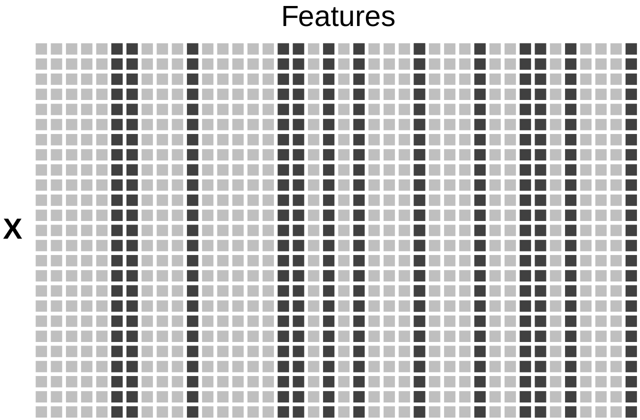
Feature Selection



Feature Selection



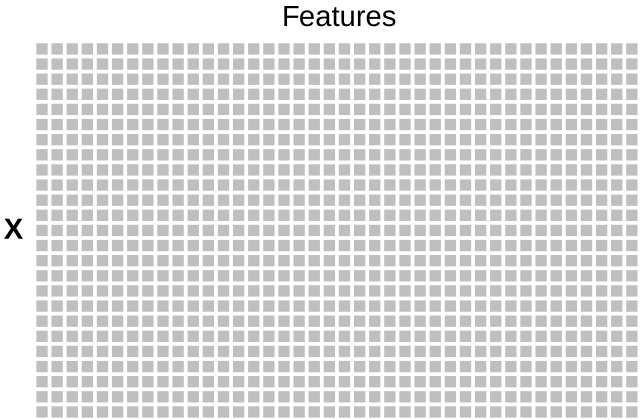
Feature Selection



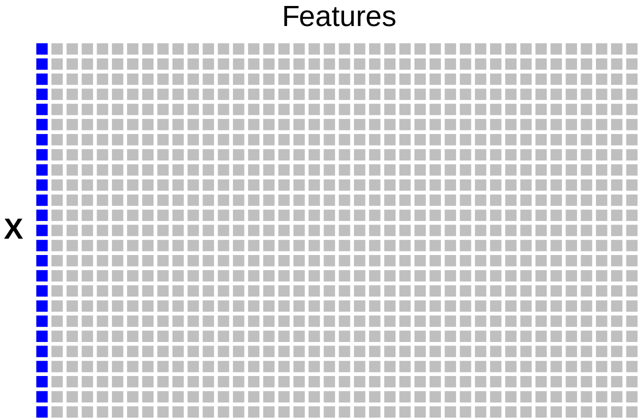
Feature Selection

For D features, train the model $O(2^D)$ times

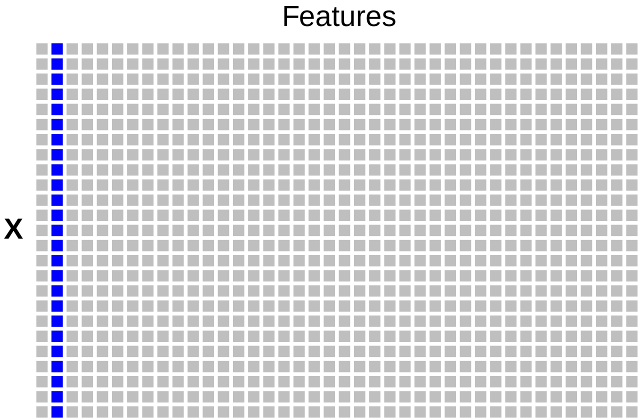
Forward Feature Selection



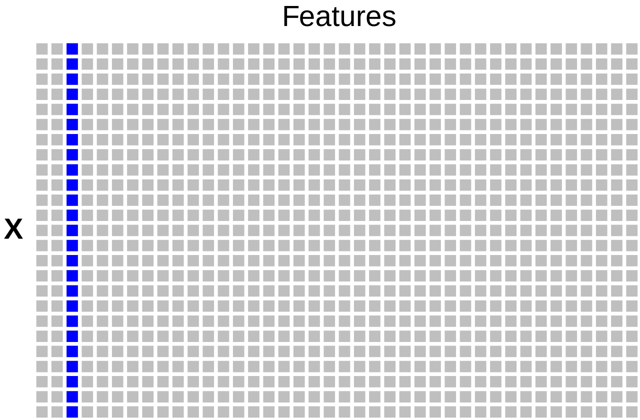
Forward Feature Selection



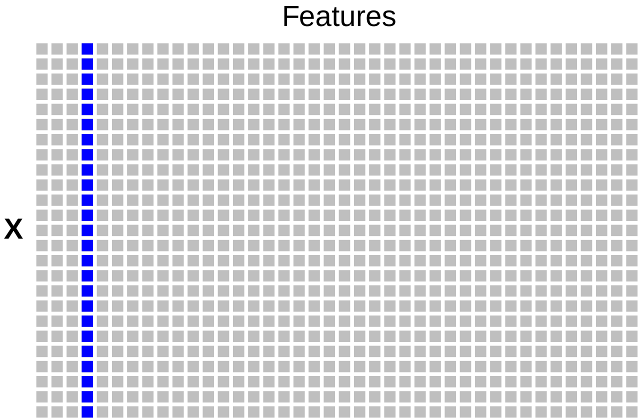
Forward Feature Selection



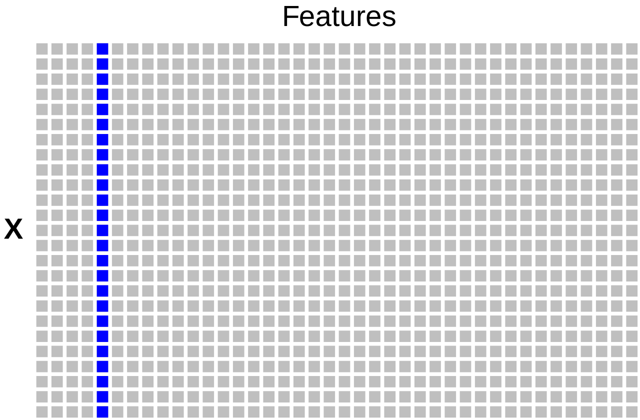
Forward Feature Selection



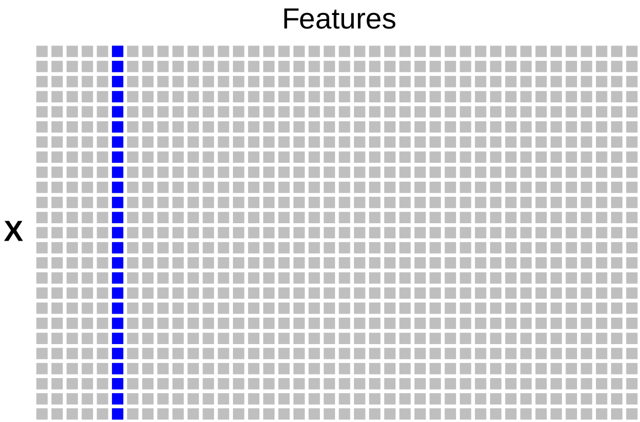
Forward Feature Selection



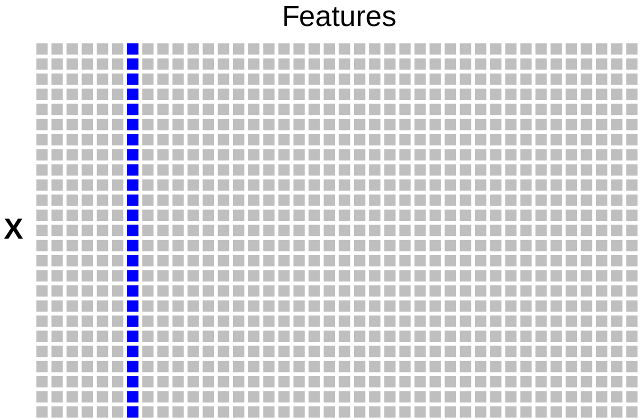
Forward Feature Selection



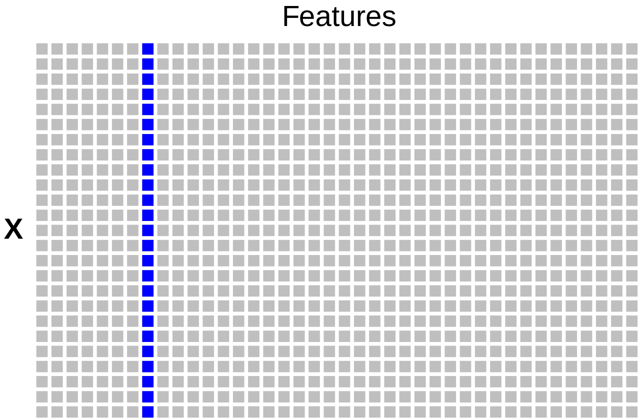
Forward Feature Selection



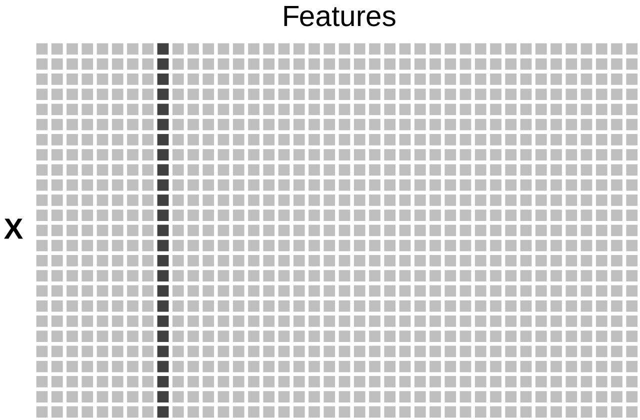
Forward Feature Selection



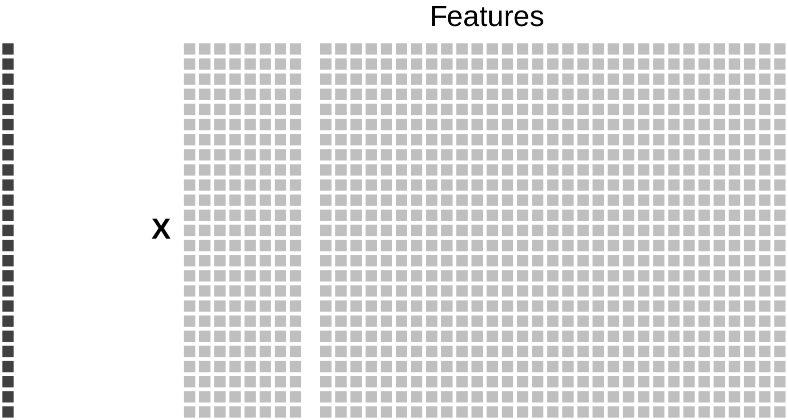
Forward Feature Selection



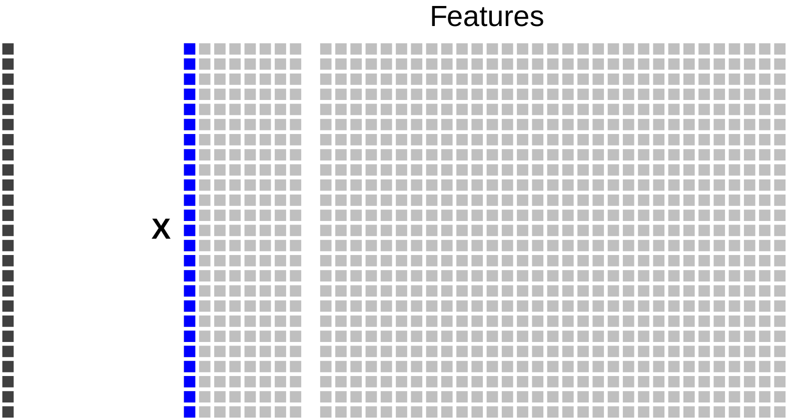
Forward Feature Selection



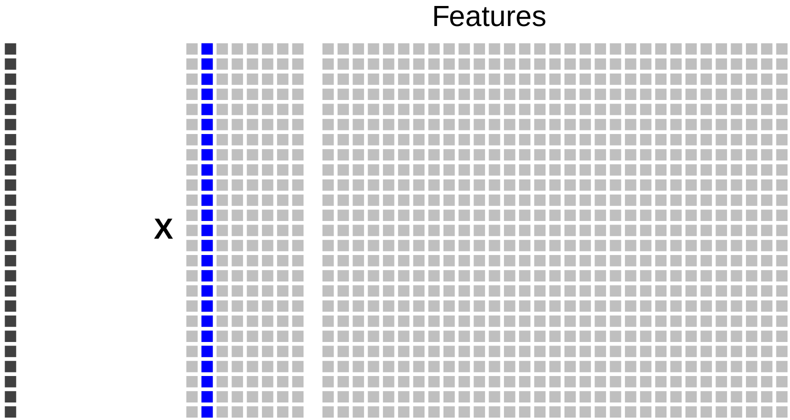
Forward Feature Selection



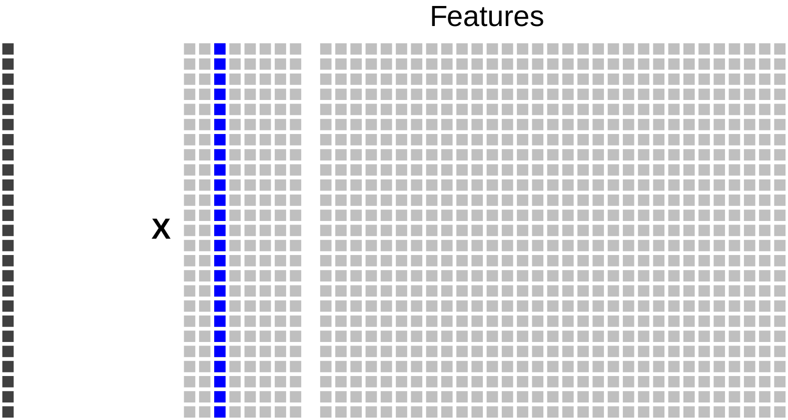
Forward Feature Selection



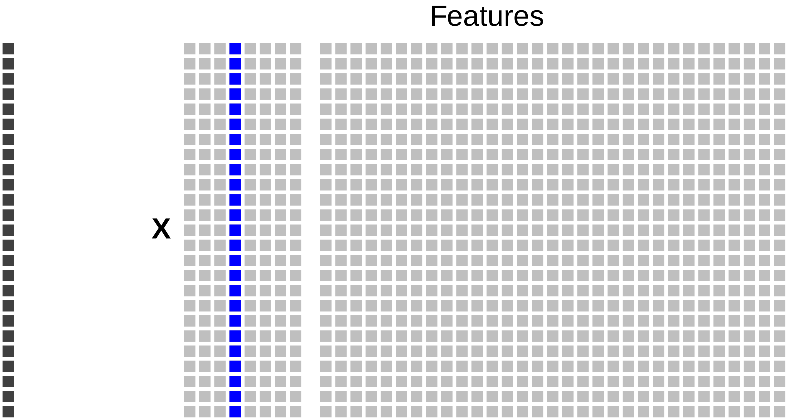
Forward Feature Selection



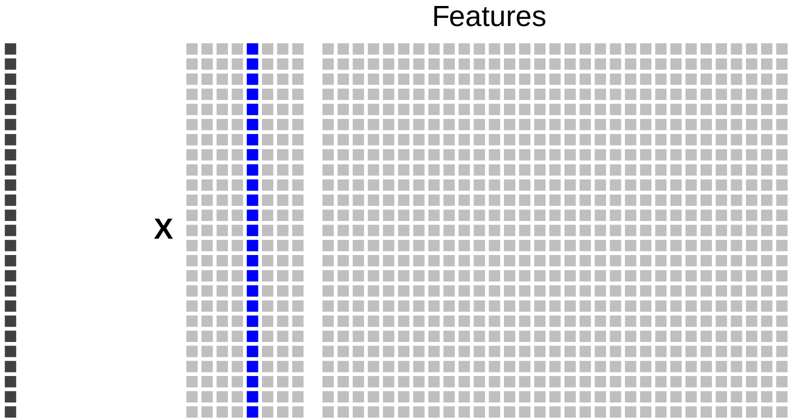
Forward Feature Selection



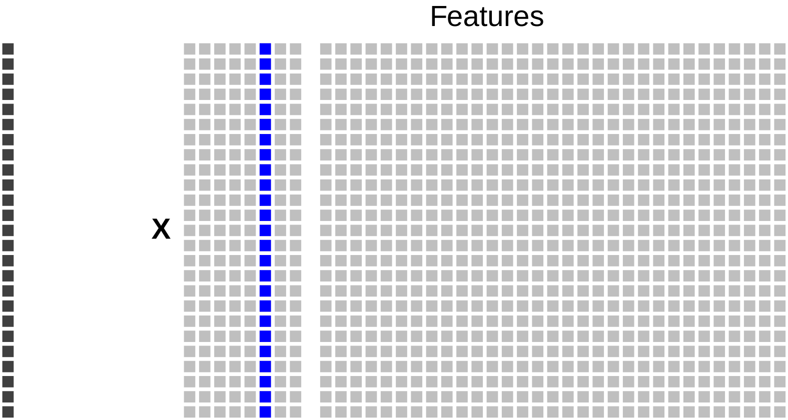
Forward Feature Selection



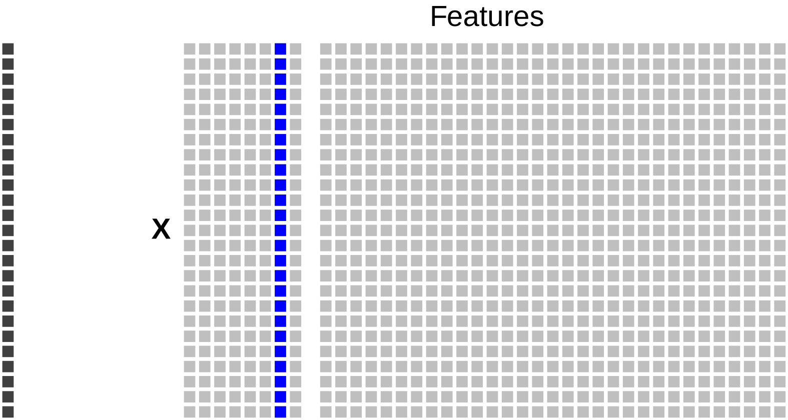
Forward Feature Selection



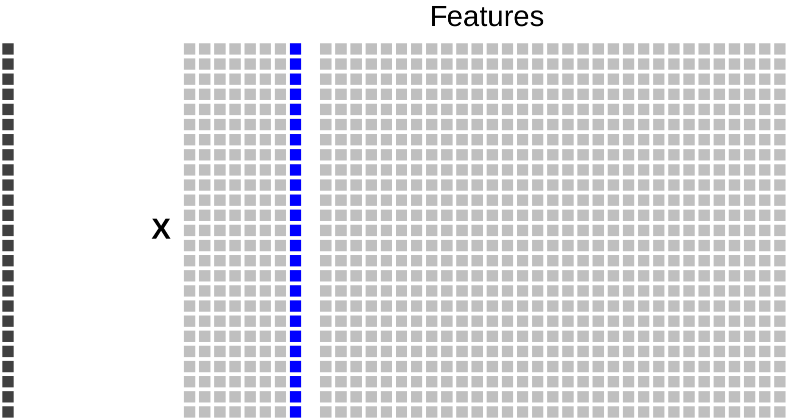
Forward Feature Selection



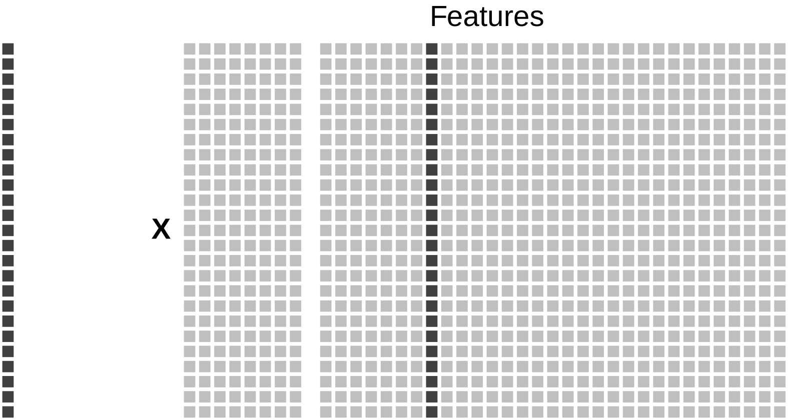
Forward Feature Selection



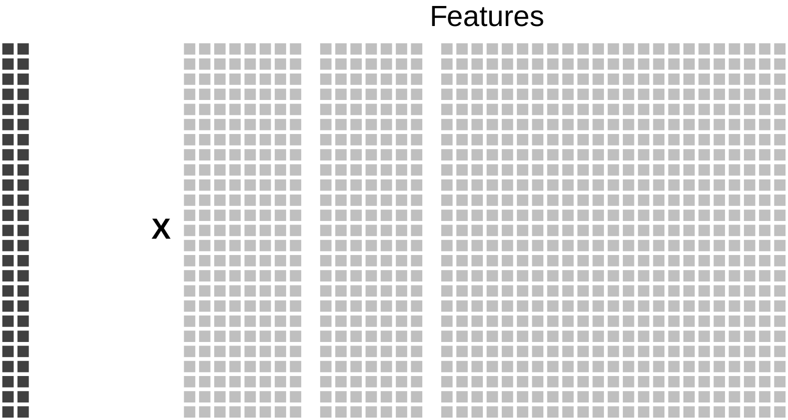
Forward Feature Selection



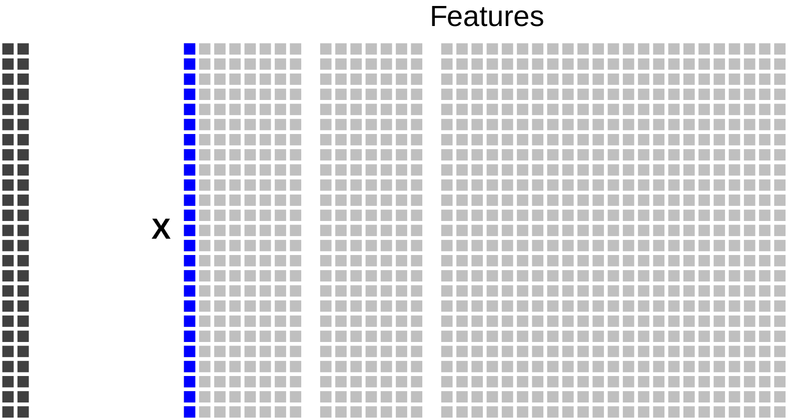
Forward Feature Selection



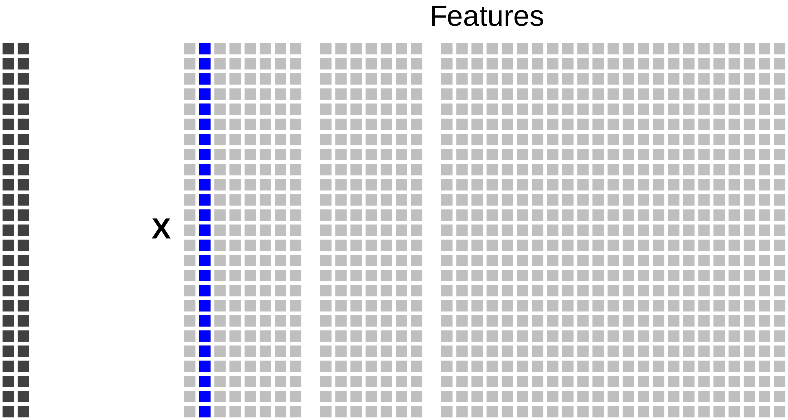
Forward Feature Selection



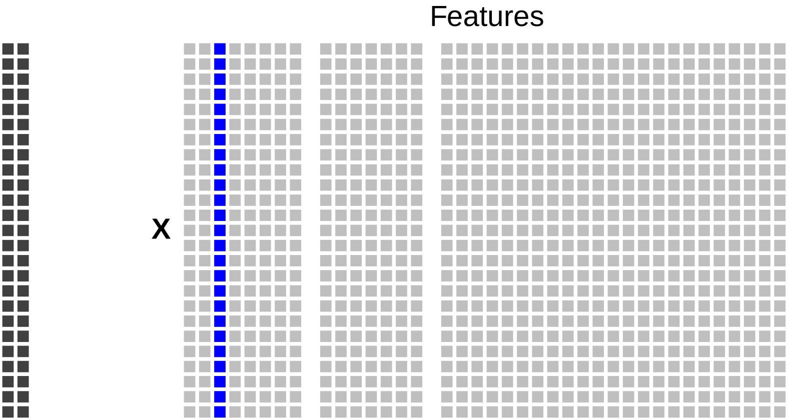
Forward Feature Selection



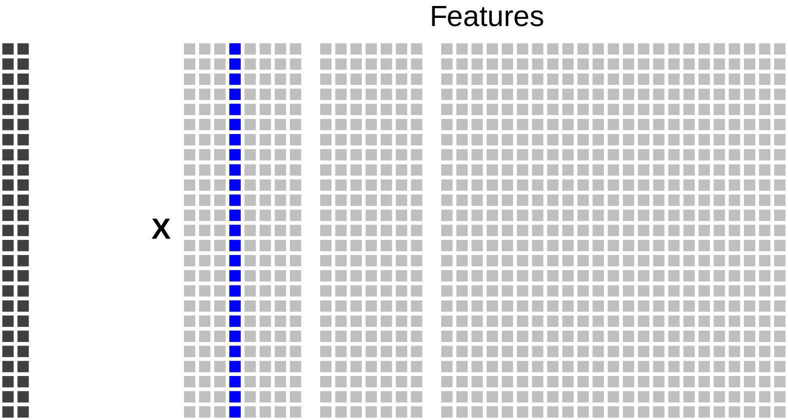
Forward Feature Selection



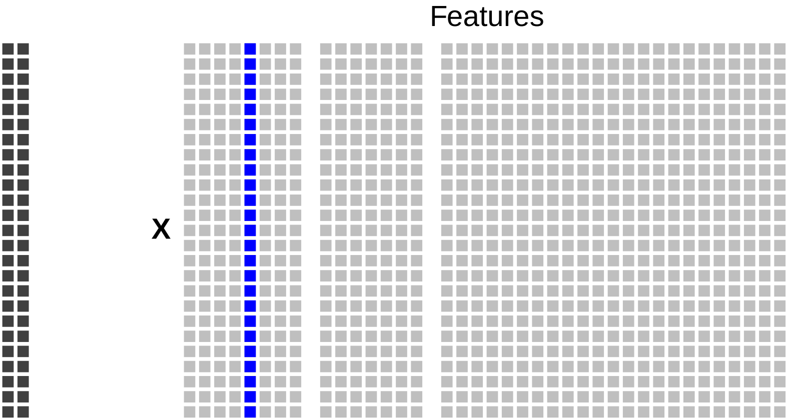
Forward Feature Selection



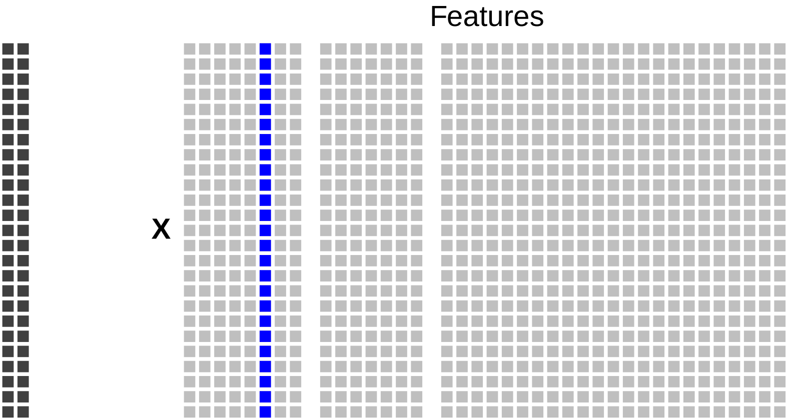
Forward Feature Selection



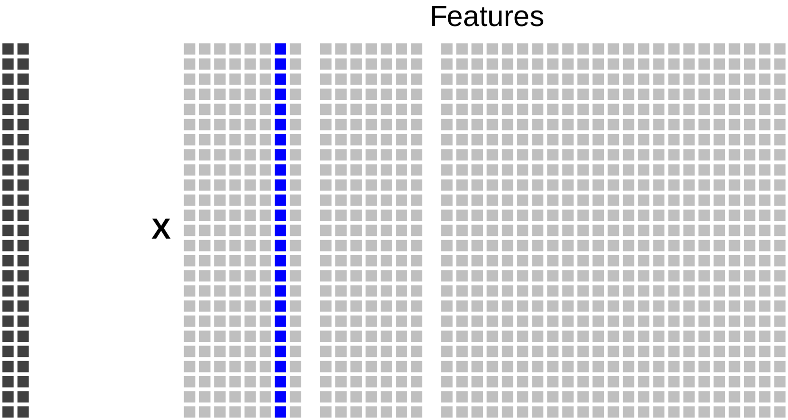
Forward Feature Selection



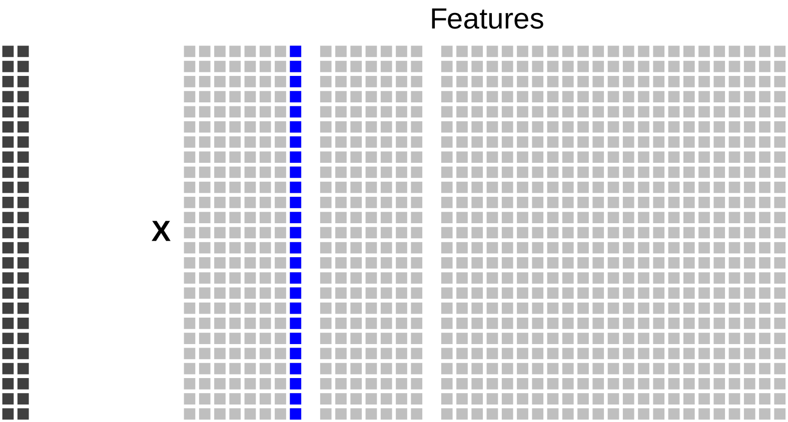
Forward Feature Selection



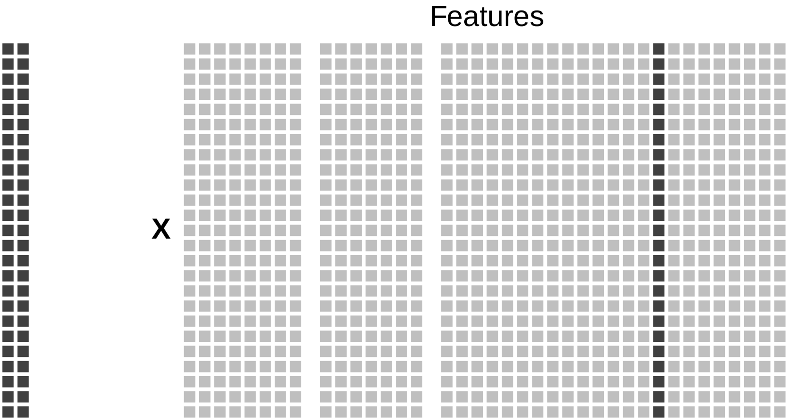
Forward Feature Selection



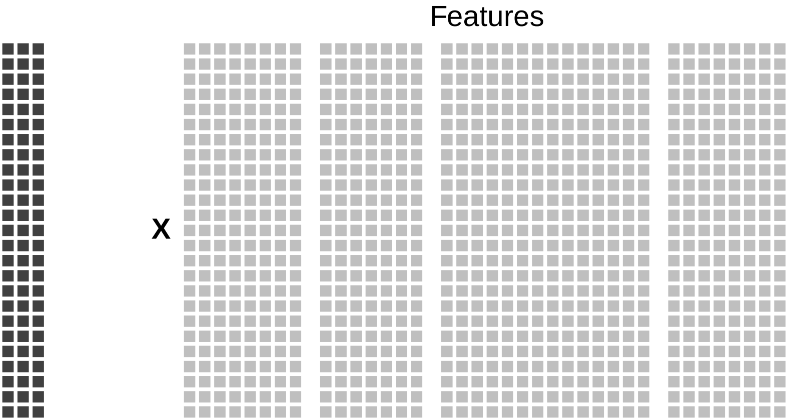
Forward Feature Selection



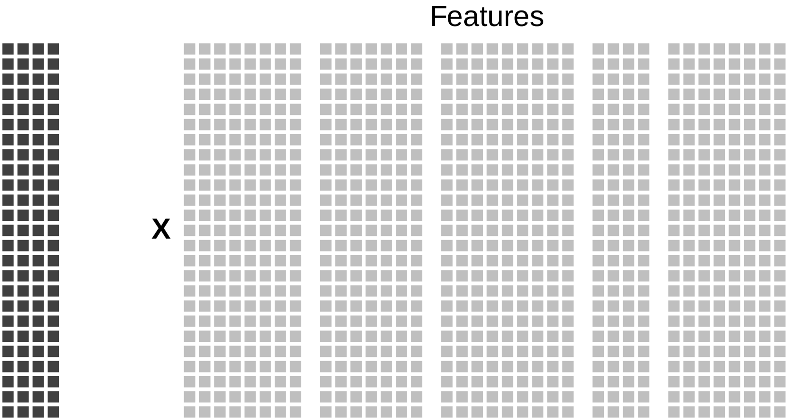
Forward Feature Selection



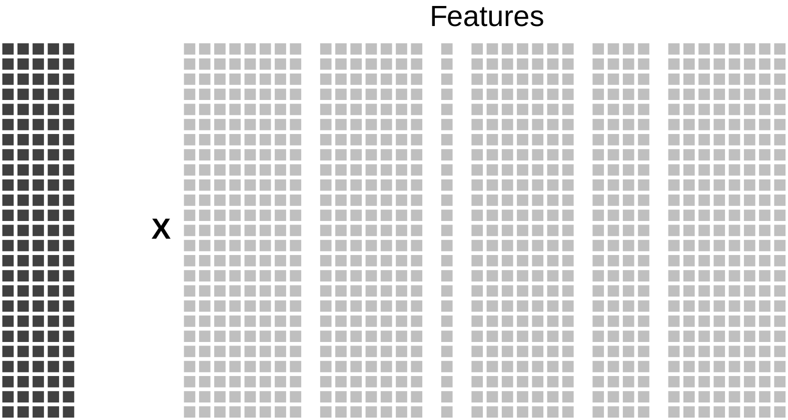
Forward Feature Selection



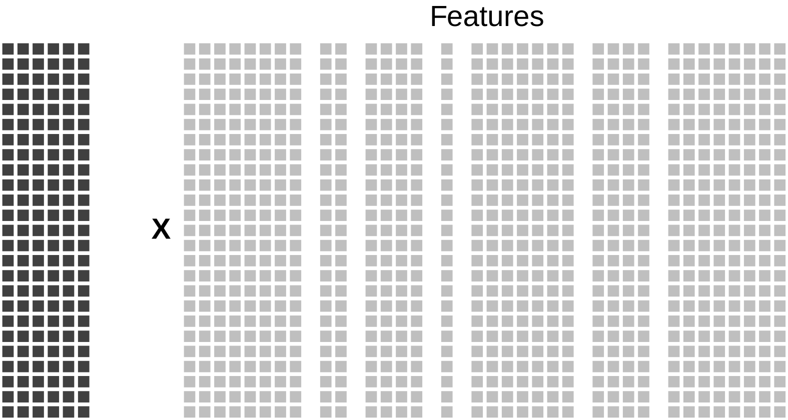
Forward Feature Selection



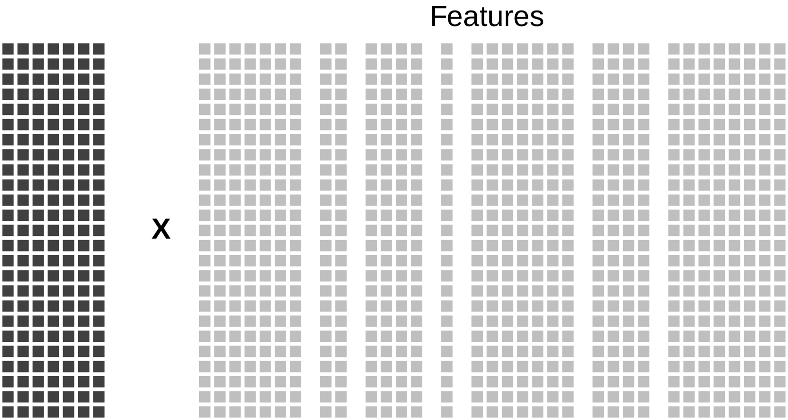
Forward Feature Selection



Forward Feature Selection



Forward Feature Selection

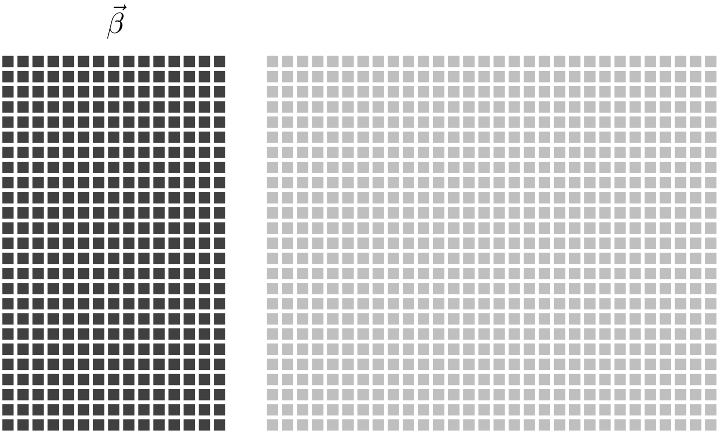


Forward Feature Selection

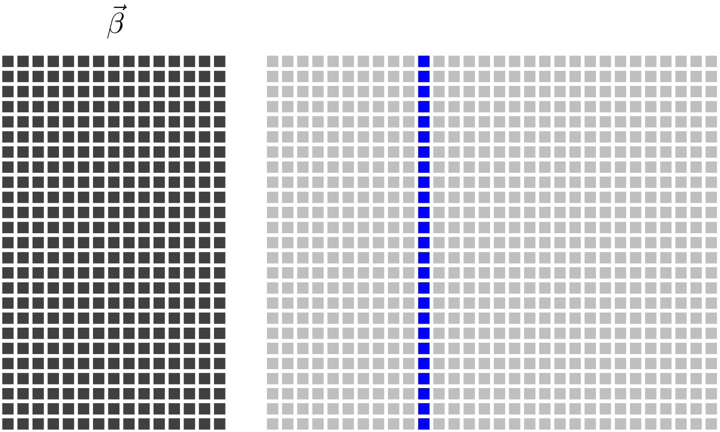
For D features, train the model $O(D^2)$ times

Outline

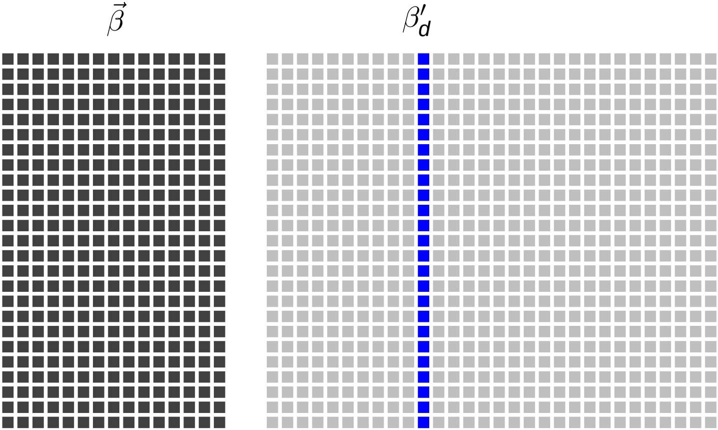
Single Feature Optimization



Single Feature Optimization



Single Feature Optimization



Newton's Method

$$p_{id} = \frac{e^{\vec{\beta}\cdot\vec{x}_i + x'_{id}\beta'_d}}{1 + e^{\vec{\beta}\cdot\vec{x}_i + x'_{id}\beta'_d}}$$

$$\beta'_d = \operatorname*{argmax}_{\beta'_d} \sum_{i=1}^{N} \Big(y_i \ln p_{id} + (1 - y_i) \ln(1 - p_{id}) \Big)$$

Newton's Method

$$
\begin{aligned}
p_{id} &= \frac{e^{\vec{\beta}\cdot\vec{x}_i + x'_{id}\beta'_d}}{1 + e^{\vec{\beta}\cdot\vec{x}_i + x'_{id}\beta'_d}} \\
\beta'_d &= \underset{\beta'_d}{\operatorname{argmax}} \sum_{i=1}^{N} \Big(y_i \ln p_{id} + (1 - y_i) \ln(1 - p_{id}) \Big)
\end{aligned}
$$

Newton's Method

$$p_{id} = \frac{e^{\vec{\beta}\cdot\vec{x}_i + x'_{id}\beta'_d}}{1 + e^{\vec{\beta}\cdot\vec{x}_i + x'_{id}\beta'_d}}$$

$$\frac{\partial L}{\partial \beta'_d} = \sum_{i=1}^{N} x'_{id}(y_i - p_{id})$$

$$\frac{\partial^2 L}{\partial \beta'^2_d} = -\sum_{i=1}^{N} p_{id}(1 - p_{id})x'^2_{id}$$

Histogram Approximation

- As N grows, Newton's method slows down considerably
- B bins, based on predicted probability of *base* model
  - using only $\vec{\beta}$ and $\vec{x}$
- Newton's method dependent on B instead of N
  - $N >> B$

Map Reduce implementation

- Map: Parallel over *records*
  - **Input:** Base features $\vec{x}_i$, class y_i, new features $\vec{x}_i'$
  - Predict using the base model p_i
  - **Output:** $(x_{id}', \langle y_i, p_i \rangle)$ for every feature x_{id}' in $\vec{x}_i'$
- Reduce: Parallel over *features*
  - **Input:** $x_d', \langle y_i, p_i \rangle^n$
  - Use Newton's method to find β_d' that maximizes scoring measure
  - With or without histogram approximation
  - **Output:** Estimated coefficient β_d'

Evaluate the coefficients on test dataset to evaluate utility

Map Reduce implementation

- Map: Parallel over *records*
  - **Input:** Base features $\vec{x}_i$, class y_i, new features $\vec{x}_i'$
  - Predict using the base model p_i
  - **Output:** $(x_{id}', \langle y_i, p_i \rangle)$ for every feature x_{id}' in $\vec{x}_i'$
- Reduce: Parallel over *features*
  - **Input:** $x_d', \langle y_i, p_i \rangle^n$
  - Use Newton's method to find β_d' that maximizes scoring measure
  - With or without histogram approximation
  - **Output:** Estimated coefficient β_d'

Evaluate the coefficients on test dataset to evaluate utility

Map Reduce implementation

- Map: Parallel over *records*
  - **Input:** Base features $\vec{x}_i$, class y_i, new features $\vec{x}_i'$
  - Predict using the base model p_i
  - **Output:** $(x_{id}', \langle y_i, p_i \rangle)$ for every feature x_{id}' in $\vec{x}_i'$
- Reduce: Parallel over *features*
  - **Input:** $x_d', \langle y_i, p_i \rangle^n$
  - Use Newton's method to find β_d' that maximizes scoring measure
  - With or without histogram approximation
  - **Output:** Estimated coefficient β_d'

Evaluate the coefficients on test dataset to evaluate utility

Outline

Methods

- IRLS: Iteratively Re-weighted Least Squares
  - P. Komarek and A. Moore, *ICDM 2005*[1]
  - Fast, efficient single machine implementation of Logistic Regression
  - Retrain classifier for each candidate feature
- SFO: Single Feature Optimization
  - Use IRLS to train the "base" model
- GD: Gradient Method
  - S. Perkins and J. Theiler, *ICML 2003*
  - Ranks features according to their gradient on training data
  - Parallelize it same way as SFO

[1] http://www.autonlab.org/autonweb/10538.html

Mushroom Dataset

Base Features	Feature Class	IRLS -LL	SFO -LL	SFO Rank	GD Rank
bias	odor	0.111	0.076	1	2
	spore-print-color	0.558	0.543	2	1
	gill-color	0.623	0.604	3	9
	stalk-surface-above	0.696	0.692	5	3
	ring-type	0.711	0.687	4	8
bias, odor	spore-print-color	0.074	0.069	1	5
	stalk-surface-above	0.098	0.090	3	3
	population	0.099	0.092	5	6
	gill-color	0.099	0.091	4	7
	stalk-color-below	0.100	0.086	2	4

Table: The negative test set log-likelihood for the top features in the Mushroom data set as selected by IRLS, the corresponding SFO scores, and rankings from SFO and the gradient method.

InternetAds Dataset

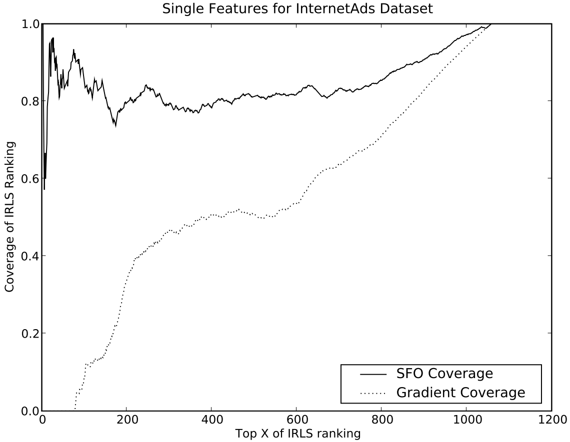


Figure: Coverage of the IRLS ranking by SFO and the Gradient method for the Internet Ads data. The features were ranked by test set log-likelihood.

RCV1

Round 1		Round 2		Round 3		Round 4		Round 5	
bias		bias econ		bias econ muni		bias econ muni defi		bias econ muni defi *shar*	
econ	283.7	muni	204.3	defi	110.2	*shar*	106.7	infl	79.5
defi	213.7	*shar*	139.3	*shar*	106.8	stat	82.1	wag	77.1
infl	190.1	coup	131.3	stat	90.2	wag	79.7	stat	76.5
gdp	182.9	obli	110.3	infl	87.2	*profit*	79.1	mood	68.6
muni	176.3	*prof*	106.1	gdp	86.6	infl	74.7	*dig*	66.0

Table: Top 5 features & estimated improvement on training set loglikelihood.

Timing Results

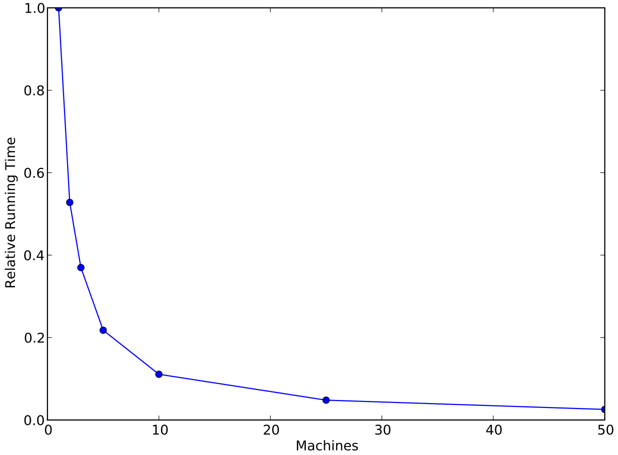


Figure: Timing (10,000,000 records / 100,000 features)

Speedup

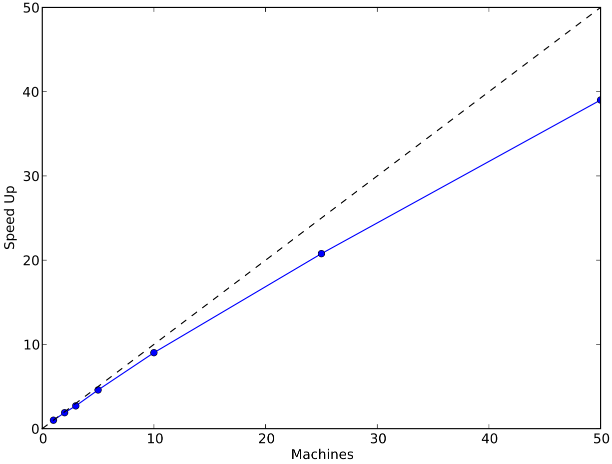


Figure: Speedup (10,000,000 records / 100,000 features)

Summary

- Introduce Single Feature Optimization (SFO)
  - approximation to Forward Feature Selection
- To scale to large datasets, utilize MapReduce for parallelism
- Histogram Approximation is used to further scalability

- Future Work:
  - Multiple Feature Optimization
    - pairs of features
  - Optimize on metrics other than LogLikelihood

Thank You

Parallel Large Scale Feature Selection for Logistic Regression

Sameer Singh[1] Jeremy Kubica[2] Scott Larsen[2]
Daria Sorokina[3]

[1]Department of Computer Science
University of Massachusetts, Amherst

[2]Google Inc, Pittsburgh

[3]Department of Computer Science
Carnegie Mellon University

SIAM Data Mining, 2009

Histogram Approximation

- For each bin b
  - Mean probability p_{id} of the bin $\hat{p}_b$
  - Total number of records in the bin N_b
  - Number of records in which $x_d = 1$, N_b^+
- Calculate p'_b using $\hat{p}_b$ and β_d

$$\frac{\partial L}{\partial \beta'_d} = \sum_{b=1}^{B} N_b^+ - p'_b \cdot N_b$$

$$\frac{\partial^L}{\partial \beta'^2_d} = -\sum_{b=1}^{B} N_b \cdot p'_b \cdot (1 - p'_b)$$

Map Reduce implementation

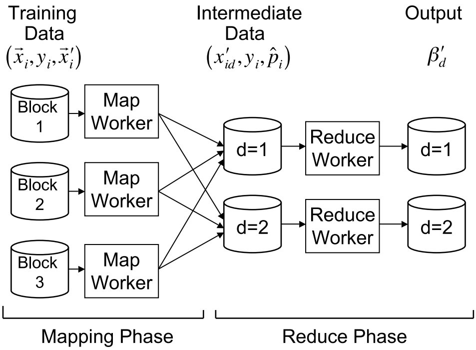


Figure: Map: operate on training data $(\vec{x}_i, y_i, \vec{x}_i')$ to produce intermediate records (y_i, p_i) for each new feature in the record $\vec{x}_i'$. Reduce: operate on intermediate records, computing coefficients for the new features β'_d.