## Agenda for today

- Final project arrangements
- Machine learning for IR
  - Informal discussion of learning: classification, ranking
  - Considerations for use in standard IR tasks
- D. Sculley. 2010. Combined Regression and Ranking. In KDD 2010: Proceedings of the 16th ACM SIGKDD International Conference on Data Mining and Knowledge Discovery
  - Paper overview by Alireza

# Final project arrangements

- Final reports
  - Write it up like a conference paper
  - Probably 4-8 pages in length
  - Due by midnight (some timezone) Monday, Dec. 3, 2012
- Final presentations
  - Tuesday, Dec. 4: Andrew, Tomer, Hamid
  - Thursday, Dec. 6: Golnar, Khoa, Masoud
  - 20 minute presentation + 5 minutes for questions
  - Send us your slides prior to your talk

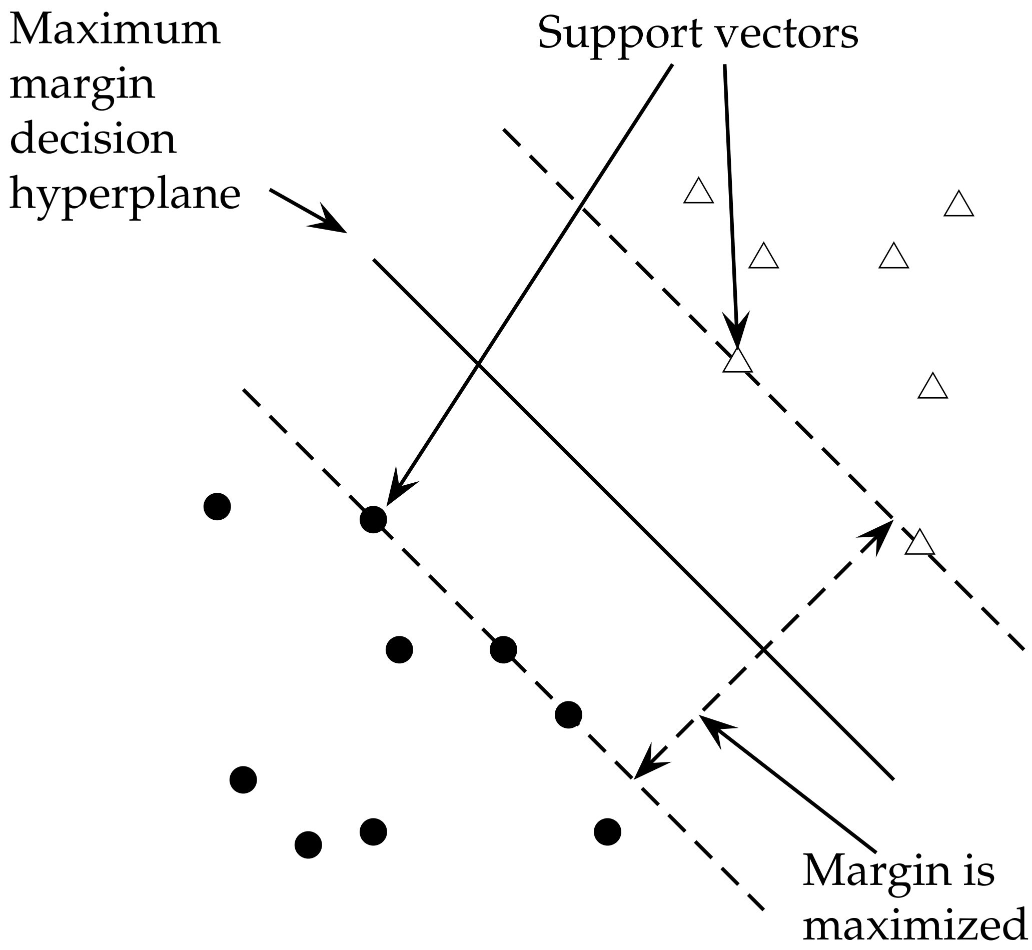
# Machine learning for IR

- Not a big ML culture in information retrieval
  - In fact, some lack of interest in supervised learning ("hill climbing")
- Far more of a history of clustering, similarity and centroids
  - Derive measures for any given pair or set of documents
  - Does not require reference labels; hence generalizes straightforwardly to novel ad hoc queries
- Yet we have seen some machine learning approaches
  - e.g., for query suggestion, reformulation
- Becoming more a part of the standard IR toolbox

## Binary classification

- Learn models to assign one of two labels (+1/-1)
- Map each example into multi-dimensional feature space
- Parameterize model to project from point in feature space to score
  - e.g., dot project of feature vector and weight vector
- Assign class based on score
- Many methods for learning models (Xubo's class)
  - Neural nets, multi-layer perceptron
  - Naive Bayes
  - Support Vector Machines
- Beyond classification, learning ranking and regression

# Standard SVM figure



## Standard considerations

- Most methods expect some supervision
  - Labeled examples in training data, for learning model
- Most methods involve some kind of regularization
  - To avoid overtraining and deal with outliers
- Generalize to multi-class learning
  - Brute force: combine $k$ one-vs-many classifiers
- Feature representation is the critical consideration
  - Also important for scalability to large data sets
- Often general numerical optimization methods used

## Learning for ad-hoc retrieval

- Can we treat this task as binary classification?
  - Yes: query-relevant documents versus the rest
- Where does the supervision come from?
  - We've discussed this quite a bit in this class
  - Derive from user behavior, e.g., click-through from logs
- Features must generalize to unseen or seldom seen queries
  - Vector-space model well suited for ML of weights
  - Also look at distances (between query and documents, documents and other documents in set)
  - TF-IDF, log-likelihood, etc. (alternative to hand-tuning)

# Feature engineering for text classification

- Words, word-classes, substrings of words
- Document regions (or zones), e.g., title, header, etc.
- Raw factors that would typically be used to calculate similarity
  - Cosine distance from query
  - Cosine distance from query expansion set
  - Proximity between terms
- May also use kernel function to achieve non-linear dependencies
- Rather than hand tuning some formula, let it be learned

## Scalability

- Many of the most interesting IR problems are web scale
- IR field focused on large scale problems for awhile
  - ML researchers, less so
- Lots of data often trumps better learning from the data
  - e.g., generative language models
- Good approximations and efficient algorithms important
  - Major improvements in recent years to deal with data
  - e.g., distributed learning, sampling methods, etc.

# Next

- Paper overview by Alireza
  - D. Sculley. 2010. Combined Regression and Ranking. In KDD 2010: Proceedings of the 16th ACM SIGKDD International Conference on Data Mining and Knowledge Discovery

# Combined Regression and Ranking

## Thanks to D.Sculley for sharing his slides

Alireza Bayestehtashk

November 24, 2012

# Motivation

Some applications require both ranking and regression together

- Predicting Star Ratings

# Motivation

Some applications require both ranking and regression together

- Predicting Star Ratings
- Sponsored Search Advertising
  - Overture : bid
  - Google : bid $\times$ **pCTR**

# Regression

$$\boldsymbol{\beta}_{opt} = \underset{\boldsymbol{\beta}}{\text{argmin}}\{ \underbrace{L(\boldsymbol{\beta}, D)}_{AverageLoss} + \lambda \times \underbrace{r(\boldsymbol{\beta})}_{Regularization} \}$$

where

$$L(\boldsymbol{\beta}, D) = \frac{1}{|D|} \sum_{(a, y_a, q_a) \in D} \underbrace{\ell(y_a, f(\boldsymbol{\beta}, a))}_{Loss}$$

- Loss : L2-norm,Huber, Logistic and Hinge
- Regularization : L1-norm, L2-norm

# Ranking

$$\boldsymbol{\beta}_{opt} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \underbrace{L(\boldsymbol{\beta}, P)}_{AverageLoss} + \underbrace{\lambda r(\boldsymbol{\beta})}_{Regularization}$$

where

$$L(\boldsymbol{\beta}, P) = \frac{1}{|P|} \sum_{((a,y_a,q),(b,y_b,q)) \in P} \ell(t(y_a - y_b), f(\boldsymbol{\beta}, a - b))$$

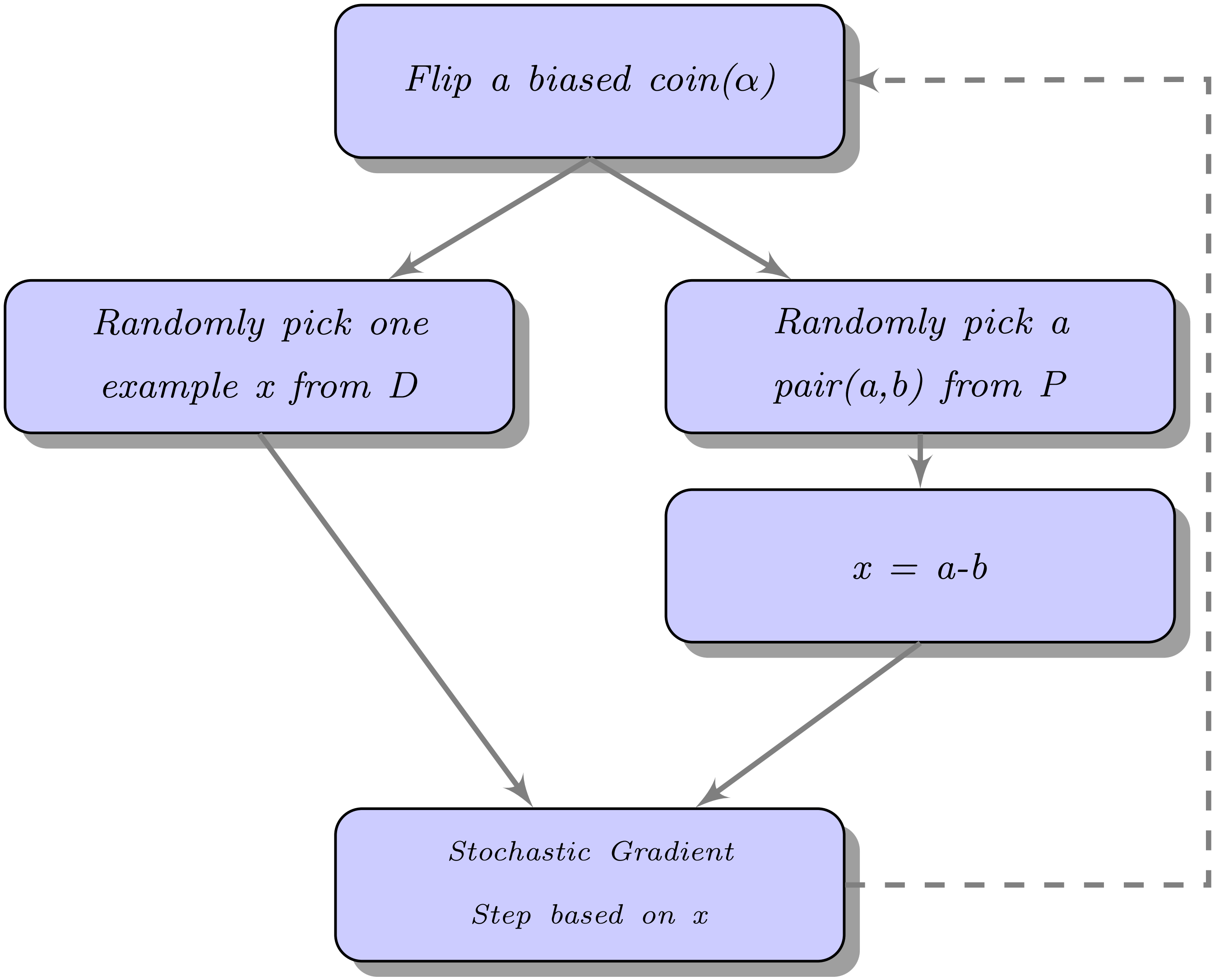
Transform function:

- Logistic : $t(y) = \frac{1+y}{2}$
- Squared :$t(y) = y$
- Hinge(SVM) :$t(y) = sign(y)$

# Combined Regression and Ranking(CRR)

$$\boldsymbol{\beta}_{opt} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \{\alpha L(\boldsymbol{\beta}, D) + (1 - \alpha) L(\boldsymbol{\beta}, P) + \lambda r(\boldsymbol{\beta})\}$$

# Solving CRR


Flip a biased coin($\alpha$)
Randomly pick one example $x$ from $D$
Randomly pick a pair(a,b) from $P$
$x$ = a-b
Stochastic Gradient Step based on $x$

# Sampling Methods

- Samping Without an Index(rejection sampling)
  - D does not fit in main memory
  - The expected number of rejected pairs per call is $O(\frac{|D|^2}{|P|})$
- Indexed Sampling
  - Fast
  - $O(log|D| + log|Y|) computations$

# Samping Without an Index

1. Read two points $((a, y_a, q_a), (b, y_b, q_b))$ from stream
2. Accept if $y_a \neq y_b$ and $q_a = q_b$, otherwise Goto (1)

# Indexed Sampling

- Q is the set of unique values q in D
- Y[q] is the set of unique y values for q in D
- P[q][y] is the set of examples $(x, \acute{y}, \acute{q}) \in D$ with $\acute{q} = q$ and $\acute{y} = y$

1. select $q$ uniformly at random from Q
2. select $y_a$ uniformly at random from $Y[q]$
3. select $y_b$ uniformly at random from $Y[q] - y_a$
4. select $(a, y_a, q)$ uniformly at random from $P[q][y_a]$
5. select $(b, y_b, q)$ uniformly at random from $P[q][y_b]$
6. return $((a, y_a, q), (b, y_b, q))$

# Stochastic gradient descent method

- Update step
  - Squared loss:

$$\beta_i \leftarrow (1 - \eta_i \lambda)\beta_{i-1} + \eta_i x(y - < \beta_{i-1}, x >)$$

  - Logistic loss:

$$\beta_i \leftarrow (1 - \eta_i \lambda)\beta_{i-1} + \eta_i x(y - \frac{1}{1 + e^{\beta_{i-1}, x>}})$$

- Learning rate:
  - $\eta_i = c$
  - $\eta_i = \frac{1}{i\lambda}$
  - $\eta_i = \frac{c}{c+i}$

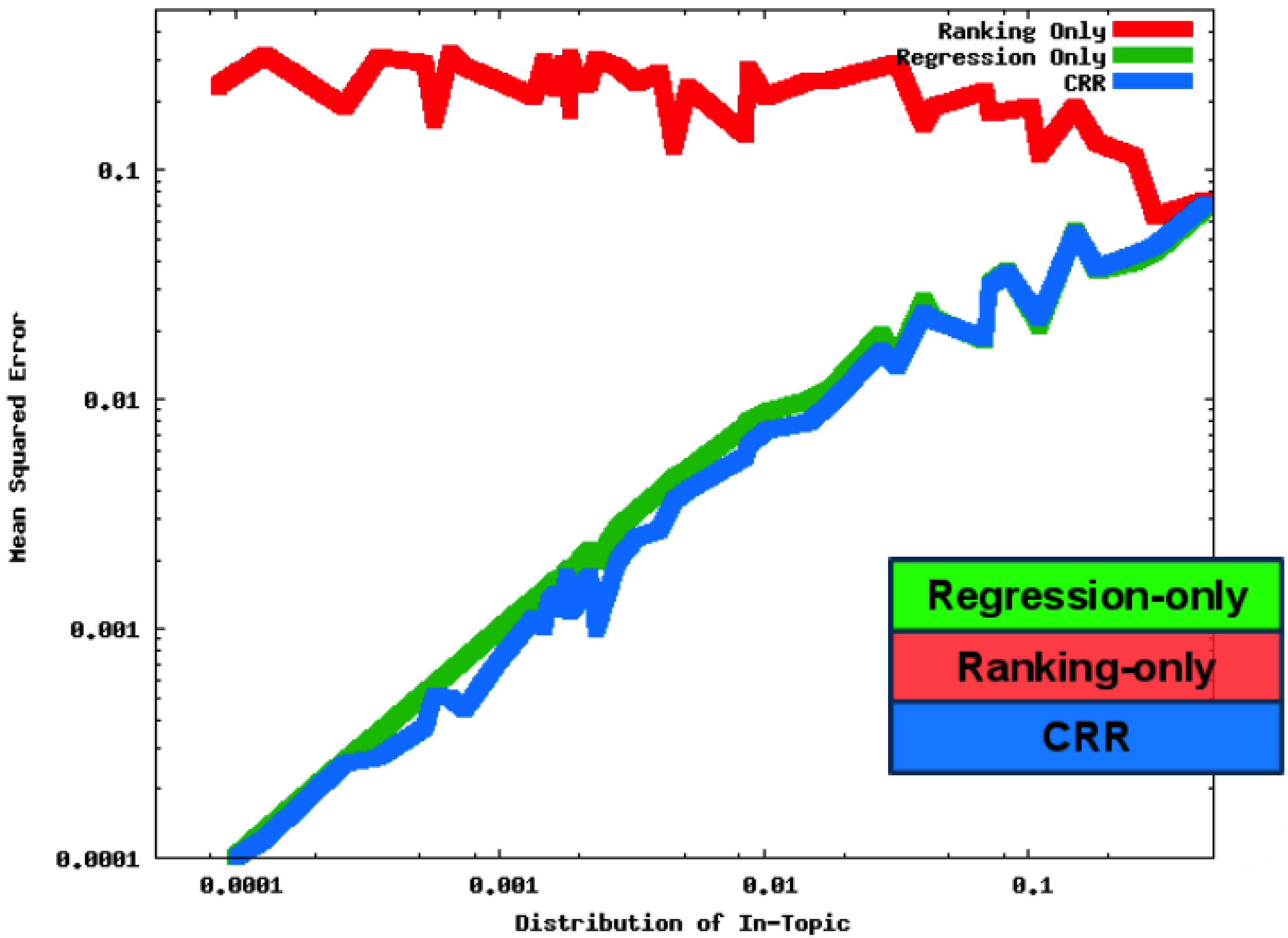
# Experimental Results

- Data sets:
  - RCV1 text classification
  - LETOR learning to rank benchmark data
  - Click prediction data for sponsored search (private)
- Comparison methods:
  - Regression-only, Ranking-only
  - Parameters tuned with cross validation on training data or on separate validation data
- Evaluation metrics:
  - Mean Squared Error (MSE)
  - AUC Loss (1 - Area Under ROC Curve)
  - Normalized Discounted

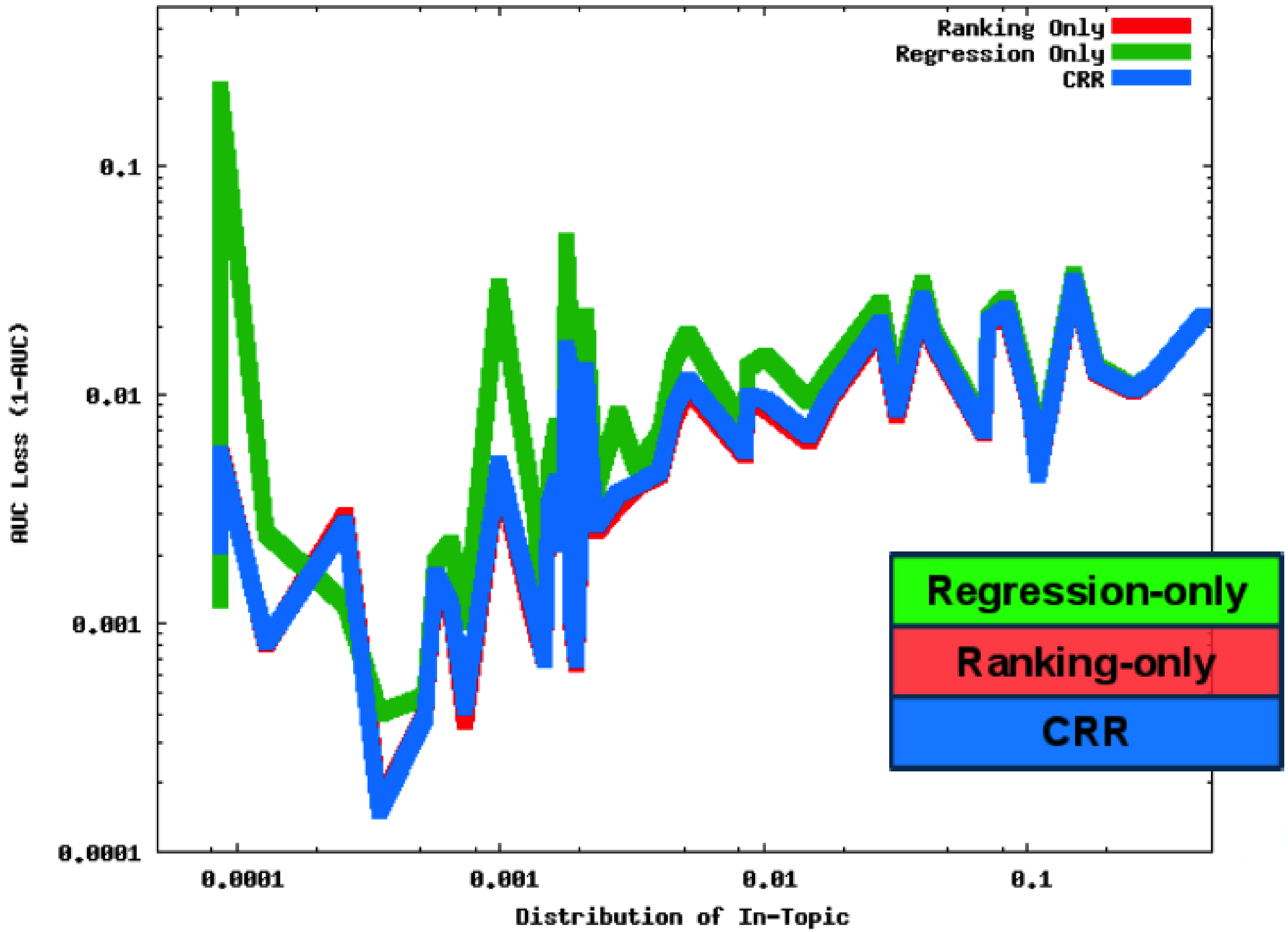
# RCV1

- Tested 40 per-topic tasks(from the 103 topics)
- 780k training examples
- 23k test examples
- 50k sparse features
- Some topics contain extreme minority class distributions
- Used logistic loss on $\{0, 1\}$ targets

# RCV1 Regression Results



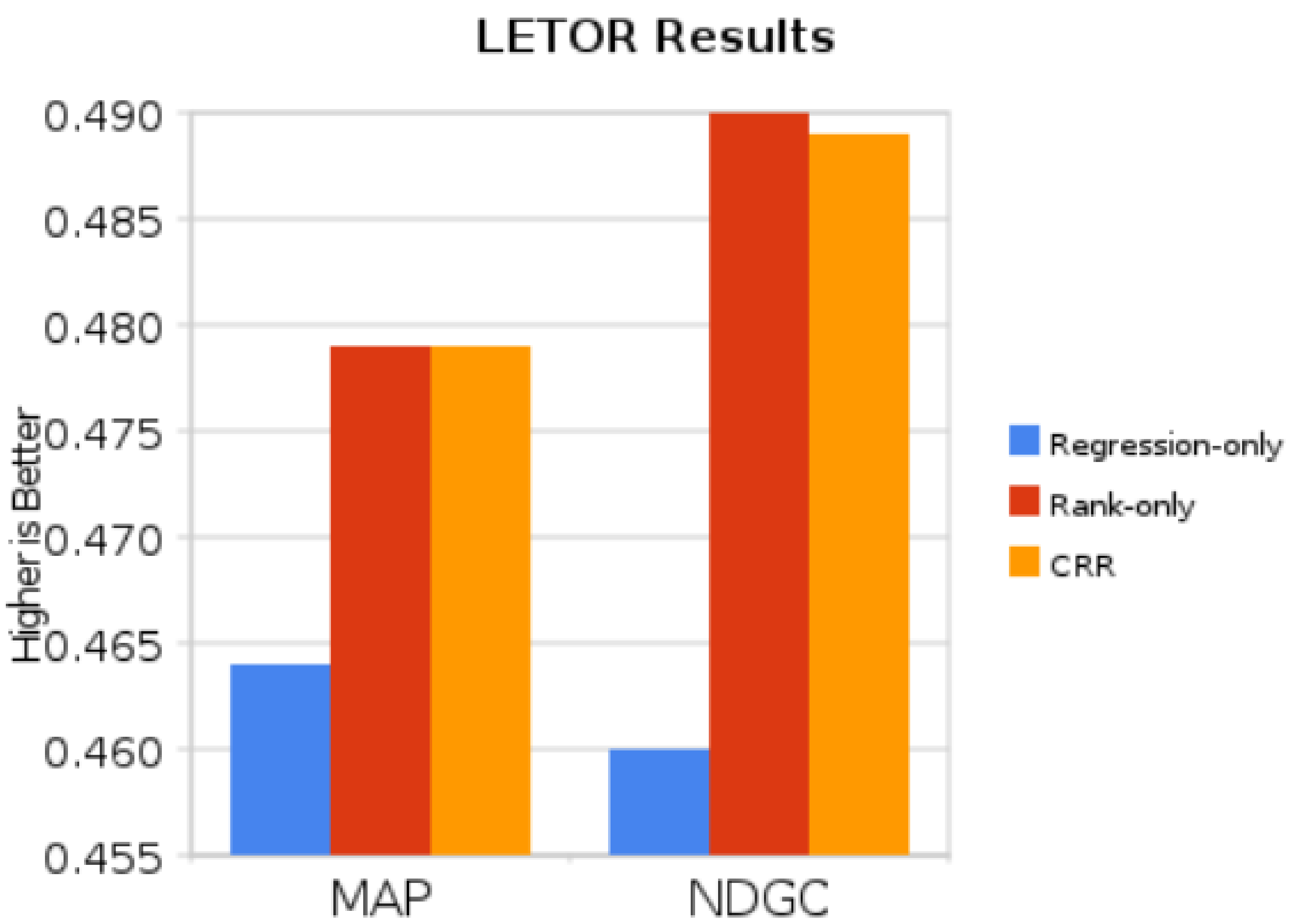
# RCV1 Ranking Results



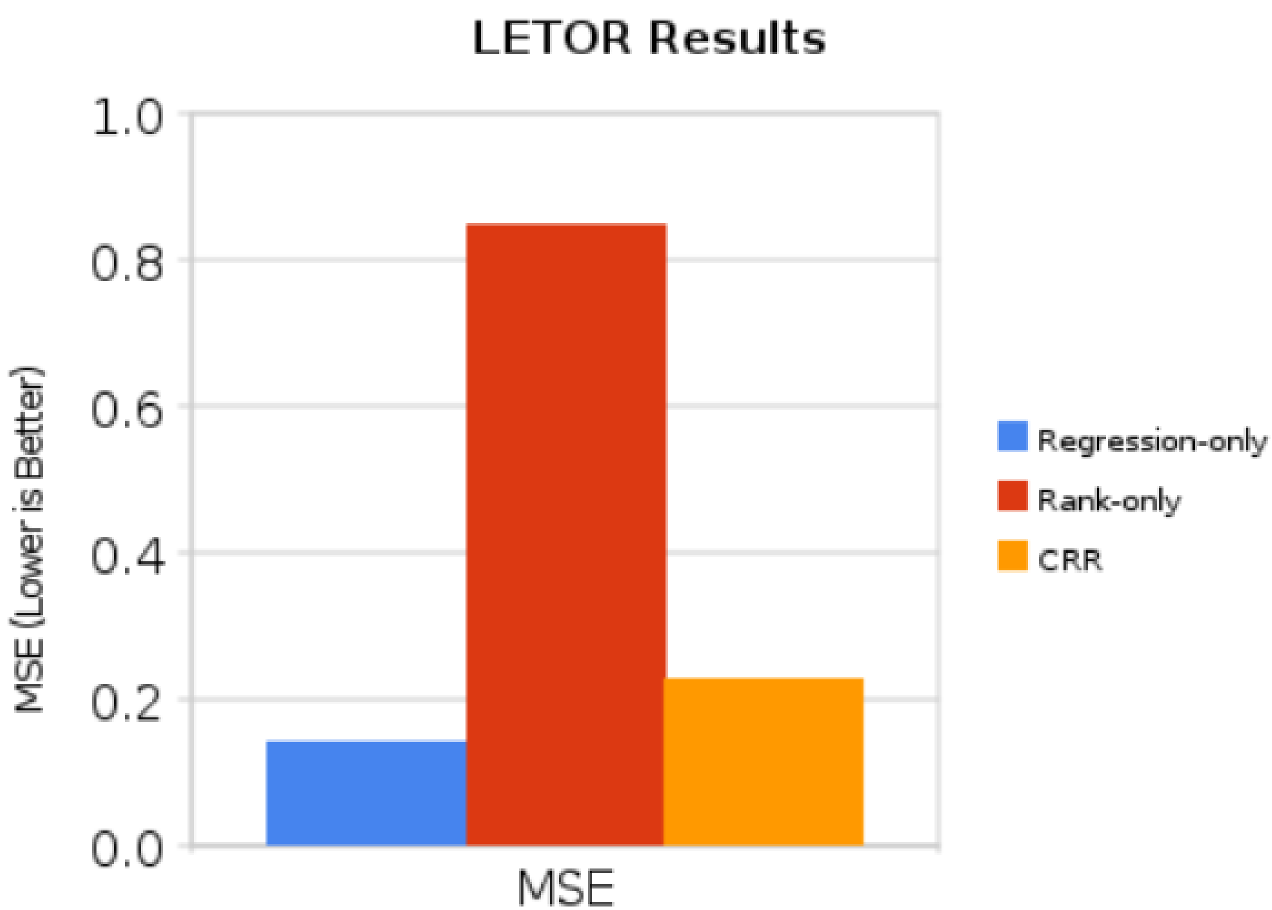
# LETOR

- Tasks with multiple relevance levels: 1,2 or 3 stars
- Squared loss

# LETOR Ranking Results



# LETOR Regression Results



# Click Prediction Data set

- Test data set of several million ads
- Labels of "clicked" and "not clicked"
- Very high dimensional feature space
- Logistic loss

# Click Prediction Results

| Method | Mean Sq. Error | AUC Loss |
|---|---|---|
| Ranking-only | 0.0935 | 0.1325 |
| Regression-only | 0.0840 | 0.1334 |
| CRR | 0.0840 | 0.1325 |

# Tradeoff Parameter