Guida codici

1. Preparazione dati
   1. **“get\_data.py”** legge i dati come da format Bing e produce *sample\_train\_features.txt* (train set), *sample\_validation\_features.txt* (validation set) e *sample\_test\_features.txt* (test set) dalla Fold1
   2. **“txt2csv\_yahoo.py”** legge i dati Yahoo! e li organizza come da formato Bing in modo da poterci far girare sopra “get\_data.py”.
2. Calcolo delle Relevance Measures
   1. **“label\_feature\_corr.py”** legge n righe del train set salvato in ***sample\_train\_features.txt*** e restituisce il file **feature\_rank.txt** compost da 4 colonne separate da tab:
      1. La prima riga riporta i nomi delle colonne “*feat\tNMI\tAG1\tKen\tSpea\n”*
      2. La prima colonna riporta la feature secondo la sintassi featX, dove X=1,2,…,F. Dove F è il numero di features totali.
      3. La seconda colonna riporta la Normalized Mutual Information tra label e feature X
      4. La terza colonna riporta la Average Group Variance tra label e feature X
      5. La quarta colonna riporta Kendall’s tau tra label e feature X
      6. La quinta colonna riporta Searman rank coefficient tra label e feature X
   2. “**lambdamart\_ndcg.py**” **”** legge I dati salvati in *sample\_train\_features.txt* (train set), *sample\_validation\_features.txt* (validation set) e *sample\_test\_features.txt* (test set) e gira un lambdamart per ogni feature (30 iterations, 100 alberi, learning rate 0.1, livelli 2). Produce su standard output una lista di NDCG medi ordinati per ordine di falutazione delle features. La lista viene copiata e incollata sul file *NDCG\_single\_feature.txt.*
3. Calcolo della Similarity
   1. **“cross\_feature\_spearmanr.py** **”** legge i dati salvati in *sample\_train\_features.txt* (train set) e per ogni coppia di feature computa lo Spearman rank coefficient. Produce il file *spear\_corr.txt* contenente F\*F Spearman coeff separati da tab.
   2. I seguenti codici non sono stati usati perché troppo pesi. **“lambdamart\_for\_similarity.py”** legge I dati salvati in *sample\_train\_features.txt* (train set), *sample\_validation\_features.txt* (validation set) e *sample\_test\_features.txt* (test set) e gira un lambdamart per ogni feature (50 iterations, 100 alberi, learning rate 0.1, livelli 2). Restituisce i file *results\_X.txt* compost da 3 colonne separate da virgola senza header: la prima colonna indica l’id della query, la seconda il predicted value del lambdamart usando la feature X e la terza la label osservata per ogni documento.

**“corrij\_ndcg.py”** legge i file *results\_X.txt* e calcola lo Spearman Rank tra le features usanto come input le prediction delle singole features

cross\_feat\_kendallb.py

cross\_feat\_nmi.py

1. Feature Selection con GAS: **“geng\_greedy.py”** legge *NDCG\_single\_feature.txt, spear\_corr.txt*, l’iperparametro c, una lista L contenente la numerosità di ciascun feature subset da produrre e la blacklist di feature da non considerare. Utilizza il valore assoluto dello Spearman Rank come misura di similarity e 1vNDCG come misura di rilevanza. Produce una lista di liste, ciascuna elemento della lista contiene il feature subset individuato da GAS di numerosità L(i), dove i è l’imo elemento di L.
2. Feature Selection con NGAS: **“my\_greedy.py”** legge *NDCG\_single\_feature.txt ,spear\_corr.txt*, una lista L contenente la numerosità di ciascun feature subset da produrre e la blacklist di feature da non considerare. Utilizza il valore assoluto dello Spearman Rank come misura di similarity e 1vNDCG come misura di rilevanza. Produce una lista di liste, ciascuna elemento della lista contiene il feature subset individuato da NGAS di numerosità L(i), dove i è l’imo elemento di L.
3. Feature Selection con NGAS-E: **“my\_greedy\_v2.py”** legge *NDCG\_single\_feature.txt, spear\_corr.txt*, un iperparametro c, una lista L contenente la numerosità di ciascun feature subset da produrre e la blacklist di feature da non considerare. Utilizza il valore assoluto dello Spearman Rank come misura di similarity e 1vNDCG come misura di rilevanza. Produce una lista di liste, ciascuna elemento della lista contiene il feature subset individuato da NGAS-E di numerosità L(i), dove i è l’imo elemento di L.
4. Feature Selection con HCAS: **“my\_hierarchical.py”** legge *NDCG\_single\_feature.txt,* una *lista di file gX.txt* in cui X indica il numero di cluster presenti nel file, una lista L contenente la numerosità di ciascun feature subset da produrre e la blacklist di feature da non considerare. Produce una lista di liste, ciascuna elemento della lista contiene il feature subset individuato da HCAS di numerosità L(i), dove i è l’imo elemento di L. I file *gX.txt* sono prodotti dallo script “**….”** Che prende in input il file *spear\_corr.txt* e usa 1-abs(*spear\_corr.txt*) per misurare la distanza tra features e costruire un clustering gerarchico e tagliare l’albero in modo da ottenere X clusters.
5. Feature Selection con MART: **“MARTforFSA.py”** prende *sample\_train\_features.txt* e gira un MART su un subset di 150,000 osservazioni. Restituisce la feature importance normalizzata per ogni feature.