Q1) Tree Bayesian Networks:

Dataset	Test LL
accidents	-33.1191
baudio	-44.1509
bnetflix	-60.1255
jester	-58.0062
kdd	-2.16163
msnbc	-6.53862
nltcs	-6.74804
plants	-16.4076
pumsb_star	-30.7905
tretail	-10.8304

Q2) Mixtures of Tree Bayesian Networks using EM:

Dataset	Valid LL	Best K	Mean_on_train(ts)	Std_on_train(ts)	Test LL
accidents	-20.7965	20	-24.12773231	0.002919729	-24.31936848
baudio	-16.6777	20	-22.52484139	0.008080651	-22.58026403
bnetflix	-30.3714	10	-49.64652774	0.030147172	-49.44405659
jester	-15.3482	20	-35.62738885	0.00998411	-35.45920039
kdd	-0.7184	20	-0.74188001223071	3.99268812494e-05	-0.6569698522
msnbc	-5.2948	10	-5.296847098	0.000384978	-5.297727075
nltcs	-3.32725	10	-4.545281174	0.009734972	-4.531728816
plants	-6.39344	10	-7.980804792	0.001301561	-8.101983985
pumsb_star	-12.5575	20	-19.28416176	0.007157566	-18.8957676
tretail	-4.18166	20	-4.692149771	0.000560651	-4.717063742

Q3) Mixtures of Tree Bayesian Networks using Random Forests:

A) <u>Extra Marks:</u> Another reasonable method for initializing P(i) rather than using 1/k:

Count the no of unique data samples selected in kth bag (with repetitions allowed) divide by the total of samples in the kth bag.

Dataset	Valid LL	Best K	Best R	Mean_on_train	Std_on_train	Test LL
accidents	-24.6814	2	100	-25.09239774	0.514526178	-25.5907
baudio	-20.1491	2	1000	-21.6520321	0.49321422	-21.76
bnetflix	-43.3767	20	10	-47.79066676	2.121515955	-50.3088
jester	-28.368	2	10	-32.14666193	1.194366647	-31.7237
kdd	-1.52404	2	10	-1.603135514	0.044058495	-1.59244
msnbc	-0.69129	10	1000	-0.923115595	0.306162204	-1.46855
nltcs	-0.30025	20	1000	-0.600730156	0.368426764	-0.43794
plants	-8.08775	20	100	-8.726441163	0.421342606	-9.44324
pumsb_star	-19.1287	2	100	-21.60520361	1.117738571	-21.2413
tretail	-5.02215	5	1000	-5.11392599	0.074831175	-5.06095

Which model Performs better on test data:

A) As the performance of model is dependent on the dataset, we can't rank the algorithms:

EM algorithm converges to a better value for datasets: accidents, bnetflix, kdd, plants, pumsb_star, tretail

Random Forest LL converges to a better value for datasets: baudio, jester, msnbc, nltcs

Dataset	Test LL using EM	Test LL using Random Forest
accidents	-24.3194	-25.5907
baudio	-22.5803	-21.76
bnetflix	-49.4441	-50.3088
jester	-35.4592	-31.7237

kdd	-0.656969852	-1.59244
msnbc	-5.29773	-1.46855
nltcs	-4.53173	-0.43794
plants	-8.10198	-9.44324
pumsb_star	-18.8958	-21.2413
tretail	-4.71706	-5.06095

Q4) Extra Marks:

- The two main implementation-specific details in the Gradient boosting algorithm are:
- 1)the set of H weak models 2) the method for searching for the "Optimal" weak model ht at each boosting iteration.
- For each boosting step find a first order optimal weak learner which gives the "steepest descent" in the loss at the current model predictions. (Step b in the image)
- The weak learner gives good improvement in the loss and then follows the "direction" of this weak learner to augment the current model (step b in the image)

1. Set $F_0(z)$ to uniform on the domain of z
2. For t = 1 to T
   (a) Set $w_i = 1/F_{t-1}(z_i)$
   (b) Find $h_t \in \mathcal{H}$ to maximize $\sum_i w_i h_t(z_i)$
   (c) If $\sum_i w_i h_t(z_i) \leq n$ break.
   (d) Find $\alpha_t = \arg\min_\alpha \sum_i -\log((1-\alpha)F_{t-1}(z_i) + \alpha h_t(z_i))$
   (e) Set $F_t = (1-\alpha_t)F_{t-1} + \alpha_t h_t$
3. Output the final model F_T

- As the models cannot be augmented, take a step in the direction by calculating alpha t described in step d and then setting Ft as described in Step e
- If at some stage, the current Ft cannot be improved by adding any of the weak learners as above, the algorithm terminates, and we have reached a global minimum. This can only happen if the derivative of the loss at the current model with respect to the coefficient of each weak learner is non-negative.
- In boosting Bayesian networks, a natural way of limiting the "strength" of weak learners in H is to limit the complexity of the network structure H. This can be done by bounding the number of edges in each "weak density estimator" learned during the boosting iteration.