CSCI-GA.2565-001 Machine Learning: Homework 3

Due 11:59pm EST, March 29, 2022 on Gradescope

We encourage LaTeX-typeset submissions but will accept quality scans of hand-written pages. LaTeX-typeset submissions will be good practice for anyone thinking of getting into research.

1 Comparing Random Forest and NN

In this problem, we will extend our discussion from class about Random Forests versus Neural Networks. In class, it was mentioned that Random Forests do well on tabular data. Tabular here means a typical dataset where each feature has a distinct meaning (e.g. height, weight, monthly users, number of music streams). In contrast Neural Networks often perform better than Random Forests on high-dimensional signal data like images or sound recordings. We will experiment with simulated and real tabular datasets. You all will simulate/choose your own datasets and we will collect the results and discuss them in a later class.

(a) You will use two datasets. First, pick a simple regression or classification dataset from Kaggle or the UCI repository (see starter code for an example of loading a Kaggle dataset). Next, simulate a dataset $D = (X, Y)$. For simulation, for example, you could let X be multivariate normal or multivariate uniform. The distribution of each feature is not so important. For Y, let it be e.g. normal with mean equal to a non-linear function of X or Bernoulli with probability equal to sigmoid of a non-linear function of X. Please report your simulation procedure. Do the following steps for both datasets.

We choose the following dataset.

`https://www.kaggle.com/datasets/ritesaluja/bank-note-authentication-uci-data`

Our simulation procedure consists of the following, X is a multivariate normal. Bernoulli with probability equal to sigmoid of a non-linear function of X.

(b) Use the Random Forest (Regressor or Classifier) and Neural Network (also known as Multilayer Perceptron) in Scikit-learn. Fit the model with the training data and evaluate on test data. Which model gets better train accuracy. What about test accuracy? Please note any hyperparameters you make use off (e.g. max depth for Random Forest or hidden sizes for the Multilayer Perceptron).

We get the following results,

For our kaggle dataset,
Model rf
Train acc 1.0
Test acc 0.9933774834437086

Model nn
Train acc 1.0
Test acc 1.0

Our models were defined as follows,

```
rf = RF(n_estimators=100,max_depth=8)

nn = MLPClassifier(solver='lbfgs', alpha=1e-5,hidden_layer_sizes=(8,32), random_state=1)
```

For hyperparameters, The number of trees in the forest is 100. Max depth of the tree is 8. Hidden leyer sizes for the neural net are 8 and 32 in order. The regularization term apha is 1e-5.

For our generated dataset,

Model rf
Train acc 0.9667910447761194
Test acc 0.9487878787878787
Model nn
Train acc 1.0
Test acc 0.9783333333333334

Our models were defined as follows,

```
rf = RF(n_estimators=240,max_depth=15)

nn = MLPClassifier(solver='lbfgs', alpha=1e-5,hidden_layer_sizes=(50,32, 8), random_state=1)
```

For hyperparameters, The number of trees in the forest is 240. Max depth of the tree is 15. Hidden layer sizes for the neural net are 50, 32 and 8 in order. The regularization term alpha is 1e-5.

(c) Now define $\tilde{X}_A$ to be a rotated version of X with a randomly sampled rotation matrix A. You can sample A with `scipy.stats.special_ortho_group.rvs(M)` where M is the number of features. Fit the random forest and NN on $(\tilde{X}_A, Y)$ rather than (X, Y). Do not re-define Y using this new $\tilde{X}_A$, keep the original Y and only change the inputs to the model. Is the better model from the previous part still better? It may change for some datasets but not others.

1. For the Kaggle data set, we have the following results Rotated
Model rf
Train acc 1.0
Test acc 0.9867549668874173
Model nn
Train acc 1.0
Test acc 1.0

In this case, the better model nn from the previous part is better.

2. For the generated data set, We have the following results,

Rotated
Model rf
Train acc 0.9651492537313433
Test acc 0.9610606060606061
Model nn
Train acc 1.0
Test acc 0.98

In this case, the better model nn from the previous part is better.

(d) In addition to the rotation, transform X with an invertible non-linear transformation so that $\tilde{X}_{h,A} = h(\tilde{X}_A)$ for some coordinate-wise invertible non-linear function h. Fit the NN and random forest on $(\tilde{X}_{h,A}, Y)$ Which model does better? It may change for some datasets but not others.

1. For the Kaggle data set, we have the following results

Transformed

Model rf
Train acc 1.0
Test acc 0.9735099337748344
Model nn
Train acc 1.0
Test acc 0.9801324503311258

In this case, the better model nn from the previous part is better.

2. For the generated data set, We have the following results,
Transformed

Model rf
Train acc 0.9655970149253731
Test acc 0.9610606060606061
Model nn
Train acc 1.0
Test acc 0.953939393939394

In this case, the rf has a better test accuracy than the nn.

(e) Rotate one more time after transforming with h. Which model does better? It may change for some datasets but not others.

1. For the kaggle data set, We have the following results,
Rotated again

Model rf
Train acc 1.0
Test acc 0.9646799116997793
Model is nn
Train acc 1.0
Test acc 0.9823399558498896

In this case, the nn is slightly better.

2. For the generated data set, We have the following results,
Rotated again Model is rf
Train acc 0.9669402985074627
Test acc 0.9610606060606061
Model is nn
Train acc 1.0

```
Test acc 0.9521212121212121
```

In this case, rf has a better test accuracy.

(f) Are there takeaways given your results? Be sure to experiment with hyperparameters.

In our case, both models in general fit the data well. The neural network does sligthly better in these tasks. The random forest works well with tabular data where each feature as has a distinct meaning. The rf also does well in our case. Here, the data is all numeric and there is enough data for the NN to learn useful representations and sligthly outperform the rf. In the last case, the NN has overfitted which can be worked on by regularization. In the process of rotation, the NN does better as it is able to capture the changes and new dependenices based on hierarchical transformations.

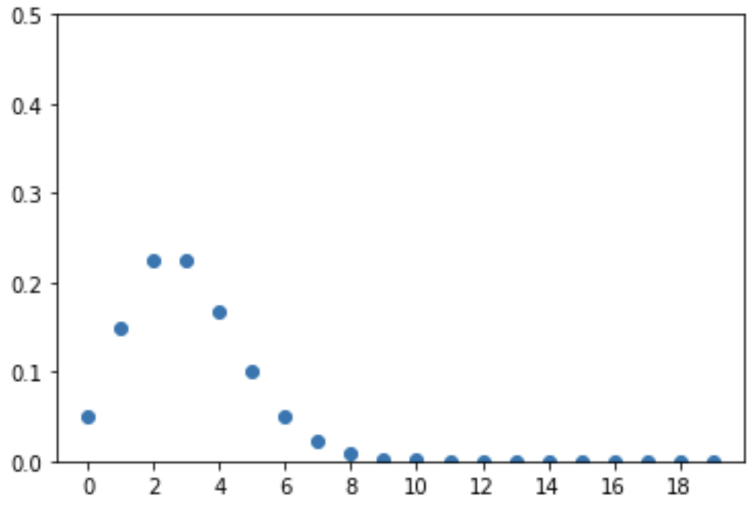


Figure 1: Poisson with rate 3 has one mode

2 Mixture of Poissons

In this problem, we will study latent variables using a two component mixture of Poissons. As part of this discussion we will study *modes*. For a continuous variable, the modes are the local maxima of the density function. For example, a Gaussian is unimodal with its mode located at its mean. For discrete distributions, modes are defined similarly. Plotted in Figure 1 is a Poisson with $\lambda = 3$, which has one mode at 3.

(a) Write down the Poisson PMF with parameter λ.
Solution:

$$p(x; \lambda) = \frac{e^{-\lambda}\lambda^x}{x!}$$

(b) Let the latent variable prior be $z_i \sim \text{Bern}(p)$ for $\theta \in [0, 1]$. Given z_i and for fixed parameters λ_0, λ_1 let the data likelihood be $x_i|z_i \sim \text{Pois}(\lambda_{z_i})$. Write down an expression for $p(x_i)$

$$p(x_i, z_i) = p(x_i|z_i)p(z_i)$$

$$p(x_i) = \sum^{z_i} p(x_i/z_i)p(z_i)$$

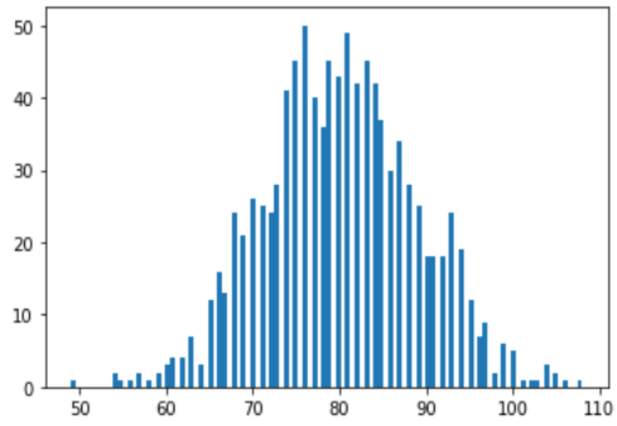
Based on Bernoulli distribution,
$p(z_i = 0) = p$ and $p(z_i = 1) = 1 - p$

$$p(x_i) = p * \frac{e^{-\lambda_0}\lambda_0^x}{x!} + (1 - p) * \frac{e^{-\lambda_1}\lambda_1^x}{x!}$$

(c) Simulate $N = 1000$ data points from a Poisson (a regular one, not our mixture) for different choices of λ. Write down what you observe. How many modes do these data distributions have?

The distribution will have one mode. The value of the mode will be proportional to the λ value.

```python
from collections import Counter
dist = generate_poisson(80)
plt.hist(dist.numpy(), bins= int(max(dist).item()))
plt.show()
a=Counter(dist.numpy())
plt.plot(list(a.keys()),list(a.values()))
plt.show()
#print(dist.numpy())
```

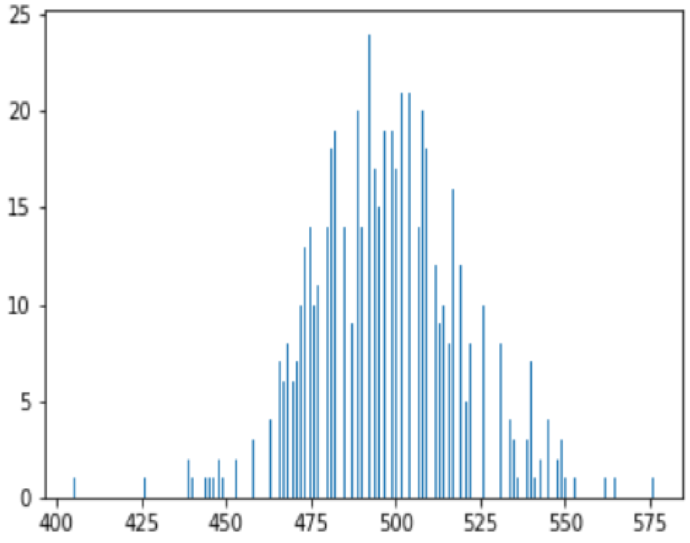


Here for the lambda value of 80, the mode is 76 empirically.

(d) Now simulate some data from our mixture of Poissons for different choices of λ_0, λ_1 (close together, far apart, small, large) and plot the data. Write down what you observe. How many modes do you see?

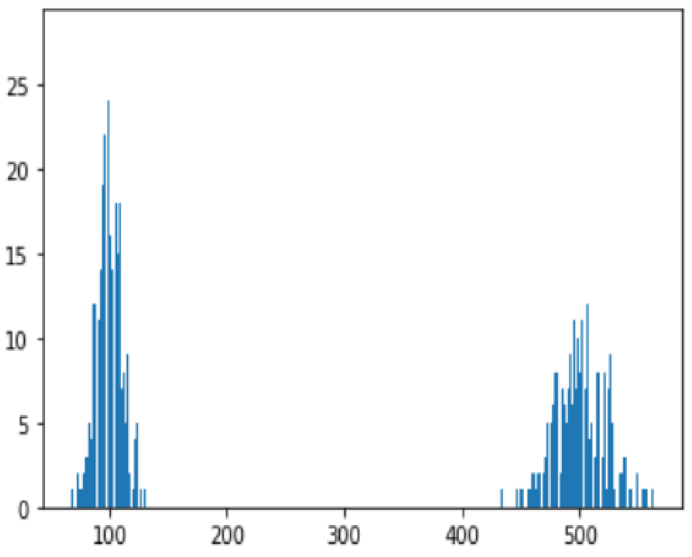
For lambda values as 500 and 495,

```python
dist = generate_mixture_poisson(500,495)
plt.hist(dist.numpy(), bins= int(max(dist).item()))
plt.show()
```



For lambda values as 100 and 500,

```
[42] dist = generate_mixture_poisson(100,500)
     plt.hist(dist.numpy(), bins= int(max(dist).item()))
     plt.show()
```



There will be two modes. Each one near the corresponding lambda value. As the lambda values are far apart, the difference in the modes is more clearly demarcated.

(e) Suppose the true data comes from a mixture of Poissons. Is it guaranteed that there a single poisson distribution model that has 0 KL with $p_{true}(x_i)$?

There is no guarantee that we can find a single Poisson distribution that fits two distributions, in case of the mixture of Poissons,we can have possibly multiple modes from different Poisson models. A mixture of Poissons is not necessarily a Poisson.

(f) Come up with a distribution without latent variables over the same support as the Poisson that also has two modes.

The support of the Poisson distribution is natural numbers starting from zero. Consider the distribution as follows,

For $x \in N_0$
$p(X = x) = 0.5$ if $x = 9$
$p(X = x) = 0.5$ if $x = 999$
$p(X = x) = 0$ else

This is a bimodal distrbution.

(g) The mixture of poissons and your distribution from the previous part both have two modes. Which was easier to specify? Which is more practical to use

The previous distribution is easier to specify. The mixture of Poissions is more practical to use. Mixture models can be relevant naturally for various tasks such as biological measurements etc. The Possion distribution and the Poission process have a theoretical basis and corresponding practical applications.

3 Building a Latent Variable Model

In this question, you will build a latent variable model based on certain assumptions about the data. Consider you have data of N patients that were treated with a new drug to control their blood cholesterol (BC) level. For each patient in the dataset, you have a single scalar change in BC and no other data; call this $\{x_i\}_{i\leq N}$. A biologist collaborator tells you that there potentially exist K patient characteristics that affect how BC changes due to the drug. Since these are unmeasured, you build a model with latent variables $z_{i,k}$ that indicate the presence of characteristic k for patient i.

Let the characteristic vector for patient i be $\mathbf{z}_i = [z_{i,1}, \cdots, z_{i,K}]$ such that $z_{i,k} \in \{0,1\}$. Assume $(x_i, \mathbf{z}_i) \perp\!\!\!\perp (x_j, \mathbf{z}_j)$ for $i \neq j$.

(a) Choose a prior distribution for the characteristic vectors $\mathbf{z}_i \sim p(\mathbf{z})$ and describe why you chose it e.g. what assumptions did you use.

Our prior distribution is as follows,

$$p(z) = \prod_{k=1}^{K} Bern(\theta_k)$$

We have independent Beroulli distributions for for the latent variable vector z. For each z_k

$$z_k = Bern(\theta_k)$$

(b) The biologist tells you that given a collection of patient characteristics, the observed BC is sampled by

$$v_{i,k} \sim \mathcal{N}(\mu_k, 1), \qquad \epsilon_i \sim \mathcal{N}(0,1), \quad x_i = \sum_{k=1}^{K} z_{i,k} v_{i,k} + \epsilon_i$$

Write down the the likelihood for $p(x_i \mid \mathbf{z}_i)$ explicitly.

We have,

$$x_i = \sum_{k=1}^{K} z_{i,k} v_{i,k} + \epsilon_i$$

This is a sum of normally distributed random variables. Given z we get,

$$E[x_i] = \sum_{k=1}^{K} z_{i,k} E[v_{i,k}] + E[\epsilon_i]$$

$$E[x_i] = \sum_{k=1}^{K} z_{i,k} u_k$$

$$Var(x_i) = \sum_{k=1}^{K} z_{i,k}^2 Var(v_{i,k}) + Var(\epsilon_i)$$

$$Var(x_i) = \sum_{k=1}^{K} z_{i,k}^2 + 1$$

The distribution $p(x_i/z_i)$ will be a gaussian distribution with,

$$mean = \sum_{k=1}^{K} z_{i,k} u_k$$

$$variance = \sum_{k=1}^{K} z_{i,k}^2 + 1$$

$$p(x_i \mid \mathbf{z}_i) \sim \mathcal{N}\left(\sum_{k=1}^{K}(z_{i,k} \cdot \mu_k), \left(\sum_{k=1}^{K} z_{i,k}^2\right) + 1\right)$$

(c) Under your choices of prior and the likelihood above, write down the implied marginal distribution $p(x_i) = \mathbb{E}_{p(\mathbf{z})} p(x_i \mid \mathbf{z})$ in terms of known quantities like $\{\mu_k\}_{k \le K}$.

$$p(x_i) = \sum_z p(x_i/z_i)p(z_i)$$

$$p(x_i) = \sum_z \left[\mathcal{N}\left(\sum_{k=1}^{K}(z_{i,k} \cdot \mu_k), \left(\sum_{k=1}^{K} z_{i,k}^2\right) + 1\right) \prod_{k=1}^{K} Bern(\theta_k)\right]$$

How many modes can the marginal distribution $p(x_i)$ have? This will be a summation over all possible values of z. For some K, there will be 2^K possible z vectors. For each z vector, we will have a Gaussian for x given x. Therefore, there will have 2^K modes.

(d) How would you infer $\mathbf{z}_i$? Write down one way to do it in terms of $\{x_i\}_{i \le N}$ and $\{\mu_k\}_{k \le K}$.

For x as x_i,

$$p(z/x) = p(x,z)/p(x)$$

$$p(z/x) = \frac{p(x/z)p(z)}{\sum_z p(x/z)p(z)}$$

We now have a distribution over z. Summing over z is computationally intractable for large observed data. So we use a variational inference method to calculate the posterior. We use a approximation method to solve this problem.We have a q(z;λ) that would help us approximate a posterior for $p(x_i/z_i)$. We can maximize the evidence lower bound based on a suitable form.

(e) Suppose instead of giving you exact values, the biologist tells you a range for each μ_k in $\{\mu_k\}_{k \le K}$. What prior would you places on the means and why?

We could place a uniform prior on the u values, as it will satisfy the range condition. However,as it is not a conjugate prior, the Bayesian computations can get involved.

(f) What are the trade-offs of such a latent variable model as compared to a different choice for likelihood where we let each possible value of $\mathbf{z}_i$ parameterize the likelihood with a separate mean $\mu_{\mathbf{z}_i} = \mathbb{E}[x_i|\mathbf{z}_i]$? For example, what happens when K is large?

For the model defined in part f, As K increases, for for each possible value of z_i, there will be a corresponding gaussian defined and will have some parameterization for that. The number of possible z_i values are 2^K. Therefore, our number of parameters to be learned will also be increasing exponentially. In our latent variable model, the number of parameters will increase linearly with z.

4 Bayesian parameters versus Latent Variables

(a) Consider the model $y_i \sim \mathcal{N}(w^\top x_i, \sigma^2 = 1/\lambda)$ where the inverse-variance is distributed $\lambda = 1/\sigma^2 \sim \text{Gamma}(\alpha, rate = \beta)$. Show that $p(y_i|x_i)$ follows a generalized T distribution

$$f(t) = T(t; \nu, \mu, \theta) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\nu/2)\theta\sqrt{\pi\nu}}\Big(1 + \frac{1}{\nu}\Big(\frac{t-\mu}{\theta}\Big)^2\Big)^{-\frac{\nu+1}{2}}$$

with degrees $\nu = 2\alpha$, mean $\mu = w^\top x_i$ and scale $\theta = \sqrt{\beta/\alpha}$. You may use the property $\Gamma(k) = \int_0^\infty x^{k-1}e^{-x}dx$.

Hint: Start by trying to marginalize $p(y_i, \sigma^2|x_i)$. When you get stuck, start from the T distribution results and work backwards. Solving this problem requires one integration by substitution using the above Gamma function integral identity.

For $\lambda = 1/\sigma^2$, the pdf is given as the gamma distribution as,

$$p(\lambda) = \frac{e^{\frac{-\lambda}{\beta}}\lambda^{-1+\alpha}}{\Gamma(\alpha)\beta^\alpha} \text{ ...(1)}$$

We know for the distribution of y given x,

$$p(y_i/x_i; w; \lambda) = \frac{\lambda^{1/2}}{\sqrt{2\pi}}e^{-\lambda\frac{(y_i - w^T x_i)^2}{2}} \text{ ...(2)}$$

Substituting the given in the desired students t distribution, we have to show that,

$$p(y_i/x_i) = \frac{\Gamma(\frac{\alpha+1}{2})}{\Gamma(\alpha)\sqrt{\beta/\alpha}\sqrt{2\pi\alpha}}\Big(1 + \frac{1}{2\alpha}\Big(\frac{y_i - w^T x_i}{\sqrt{\beta/\alpha}}\Big)^2\Big)^{-\frac{2\alpha+1}{2}} \text{ ... } d$$

This is our desired equation. Let this be equation d.

Now, marginalization,

$$p(y_i/x_i) = \int p(y_i/x_i, \lambda)p(\lambda)d\lambda$$

From 1 and 2,

$$p(y_i/x_i) = \int \frac{\lambda^{1/2}}{\sqrt{2\pi}}e^{-\lambda\frac{y_i - w^T x_i}{2}} * \frac{e^{\frac{-\lambda}{\beta}}\lambda^{-1+\alpha}}{\Gamma(\alpha)\beta^\alpha}d\lambda \text{ ...3}$$

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)\beta^\alpha}\int \lambda^{-1/2+\alpha}e^{-\lambda(\frac{1}{\beta} + \frac{(y_i - w^T x_i)^2}{2})}d\lambda$$

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)\beta^\alpha}\int \lambda^{-1+(1/2+\alpha)}e^{-\lambda(\frac{1}{\beta} + \frac{(y_i - w^T x_i)^2}{2})}d\lambda$$

Comparing 3 with the form of the gamma distribution from equation 1.

We get, The integral in equation three follows the form of the gamma distribution. The gamma distribution is to be normalized. Let α_1 and β_1 be the hyper parameters for the gamma distribution in 3. We have,

$$\alpha_1 = 1/2 + \alpha$$

$$\beta_1 = \frac{1}{(\frac{1}{\beta} + \frac{(y_i - w^T x_i)^2}{2})}$$

The result of the integration will be thus $\Gamma(\alpha_1){\beta_1}^{\alpha_1}$ which entails the normalization of the gamma distribution.

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)\beta^\alpha}\Gamma(\alpha_1){\beta_1}^{\alpha_1}$$

Substituting for α_1 and β_1 This becomes,

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)\beta^\alpha}\Gamma\Big(\frac{2\alpha+1}{2}\Big)\Big(\frac{1}{\beta} + \frac{(y_i - w^T x_i)^2}{2}\Big)^{-(1/2+\alpha)}$$

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)\beta^\alpha}\Gamma\Big(\frac{2\alpha+1}{2}\Big)\Big(\frac{1}{\beta} + \frac{(y_i - w^T x_i)^2}{2}\Big)^{-(1/2+\alpha)}$$

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)\beta^\alpha}\Gamma\Big(\frac{2\alpha+1}{2}\Big)\beta^{-1/2+\alpha}\Big(1 + \frac{(y_i - w^T x_i)^2}{2\beta}\Big)^{-(1/2+\alpha)}$$

$$p(y_i/x_i) = \frac{1}{\sqrt{2\pi}\Gamma(\alpha)}\Gamma\Big(\frac{2\alpha+1}{2}\Big)\beta^{-1/2}\Big(1 + \frac{1}{2\alpha}\Big[\frac{(y_i - w^T x_i)}{\sqrt{\frac{\beta}{\alpha}}}\Big]^2\Big)^{-\frac{1+2\alpha}{2}}$$

$$p(y_i/x_i) = \frac{\Gamma\Big(\frac{2\alpha+1}{2}\Big)}{\sqrt{2\pi\alpha}\sqrt{\frac{\beta}{\alpha}}\Gamma(\alpha)}\Big(1 + \frac{1}{2\alpha}\Big[\frac{(y_i - w^T x_i)}{\sqrt{\frac{\beta}{\alpha}}}\Big]^2\Big)^{-\frac{1+2\alpha}{2}} \quad ...\ e$$

This is the equation we get. We call this equation e. We now compared the equation we get with our desired equation d.

$$p(y_i/x_i) = \frac{\Gamma(\frac{\alpha+1}{2})}{\Gamma(\alpha)\sqrt{\beta/\alpha}\sqrt{2\pi\alpha}}\Big(1 + \frac{1}{2\alpha}\Big(\frac{y_i - w^T x_i}{\sqrt{\beta/\alpha}}\Big)^2\Big)^{-\frac{2\alpha+1}{2}} \quad ...\ d$$

We see that equations e and d are the same, there $p(y_i/x_i)$ follows a generalized T distribution.

(b) We see that we can compute $p(y_i|x_i)$ in closed form. Using this form, write down the MLE objective to optimize for w for N datapoints. Do not differentiate/solve this objective.

The MLE objective for one data point is given as,

$$l = \log\left(\frac{\Gamma(\frac{\alpha+1}{2})}{\Gamma(\alpha)\sqrt{\beta/\alpha}\sqrt{2\pi\alpha}}\right) - \frac{2\alpha+1}{2}log\left(1 + \frac{1}{2\alpha}\frac{(y_i - w^T x_i)}{\sqrt{\frac{\beta}{\alpha}}}\right)$$

The MLE objective for N data points is given as,

$$L = \sum_{i=1}^{N}\Big[\log\left(\frac{\Gamma(\frac{\alpha+1}{2})}{\Gamma(\alpha)\sqrt{\beta/\alpha}\sqrt{2\pi\alpha}}\right) - \frac{2\alpha+1}{2}log\left(1 + \frac{1}{2\alpha}\frac{(y_i - w^T x_i)}{\sqrt{\frac{\beta}{\alpha}}}\right)\Big]$$

(c) Suppose instead that our model is $y_i \sim \mathcal{N}(w^\top x_i, \sigma^2)$ where $w \sim \mathcal{N}(0, I)$ and σ^2 is known. Use this specification of $p(y, w|x)$ to derive the predictive distribution $p(y^\star|x_1, \ldots, x_n, y_1, \ldots, y_N, x^\star)$ for a new data point $x^\star$. What are the parameters of this predictive distribution and how do you optimize them?

Given,

$$p(y/x; w) = \mathcal{N}(w^\top x_i, \sigma^2)$$
$$p(w) = \mathcal{N}(0, I)$$

We need to find the predictive distribution,

$$p(y_{N+1}/D, x_{N+1})$$

Where D is the data set, $x_1, ..x_N, y_1, ..., y_N$

We have,

$$p(y_{N+1}/D, x_{N+1}) = \int p(y_{N+1}/x_{N+1}, w)p(w/D)dw \text{ ...1}$$

Here,$p(y_{N+1}/x_{N+1}, w)$ is the Gaussian which denotes $p(y/x)$.

$p(w/D)$ is the posterior probability of w given D.

We find the posterior distribution over w as we have the likelihood and the prior.

$$p(w/D) \propto p(D/w) * p(w)$$

$p(D/w)$ is the likelihood of the data set which we know. $p(w)$ is the prior over w which we know. Both are Gaussians. This has to be normalized. This is a case of conjugate Gaussian priors. By the procedure of completing the square in the exponential and finding out the normalization coefficient, we can find the posterior over w. The posterior will be a Gaussian with the mean and co-variance given by.

$$m_p = (\sigma^2 X^T X + I)^{-1} \sigma^2 X^T y \text{ ...2}$$

$$\Sigma_p = (\sigma^2 X^T X + I)^{-1} \text{ ...3}$$

Once we have the gaussian posterior distribution over w, the integration to be performed in equation 1 corresponds to a convolution of two Gaussians, which is also a Gaussian. Comparing the forms, we find the form of the predictive distribution. The predicitve distribution is a Gaussian whose mean and variance is given by,

$$mean = {m_p}^T x_{N+1}$$

$$variance = \sigma^2 + {x_{N+1}}^T \Sigma_p x_{N+1}$$

Where, m_p and Σ_p are given by equations 2 and 3 respectively.

(d) Normally, we use the word *Bayesian* to refer to placing priors on model parameters and computing their posteriors. Is the first model with inverse-Gamma variance a Bayesian model? What is Bayesian about it and what is not Bayesian about it? Compare this to the second model.

In the first model, consider the problem of finding a posterior over λ. The posterior over λ will come from the likelihood which is a Gaussian and a prior which is a Gamma distribution. Comparing the forms, we can get a gamma posterior over λ subject to normalization.

Even if we have a posterior over λ. We have no prior information over w, therefore, it is not feasible for us to obtain a posterior over w. We can treat w as a just a parameter without a distribution, looking at λ as the random variable, we get this computation for the predictive distribution of a new y for a new x.

$$p(y_{N+1}/D, x_{N+1}) = \int p(y_{N+1}/x_{N+1}, \lambda)p(\lambda/D)d\lambda$$

In a Bayesian model, we have a prior over the parameters which enables to formulate a posterior over the parameters based on the likelihood. Here the predictive distribution is contingent on w. That we can have a posterior over λ follows a Bayesian approach. However, we cannot have a predictive distribution to quantify the uncertainty over y which is not contingent on the parameters w.