Motivation:

Churn rate, or the annual percentage rate at which customers stop subscribing to a service, is an important piece of information for companies to gather year after year as a measure of company performance and success, and as a predictor of future performance. Similarly, being able to predict which current customers are likely to cancel or continue their subscription to a company’s services becomes valuable for purposes such as focusing resources on retaining customers who are likely to leave. Particularly for churn prediction, the confidence of this prediction matters a lot: false positives (labeling a customer as likely to churn when they actually are not) result in wasting resources on retaining customers that did not need extra incentive to stay, and false negatives result in not putting in sufficient resources towards retaining a customer that if lost, is unlikely to return.

For our project, using customer data provided from a telecom company, we hope to predict if a customer will change providers in the near future and provide an estimate of how likely they are to either cancel services. Towards this goal, we will train a Bayesian neural network as our classifier. Specifically, we will explore the effect on model accuracy of using a model that models the aleatoric uncertainty across our dataset and compare the resultant predictions and confidence levels against models that assume a homoscedastic variance across the input feature space. The motivation behind exploring heteroscedastic variance in the input data’s feature space is the nature of real-world scenarios: the variance across any classification problem’s input feature space should not remain constant. For example, although we do not know what features are included in our dataset in order to protect the company’s proprietary information, noise could have been introduced if the dataset includes any surveys on customer satisfaction--in these cases, different customers may have different perceptions of the rating scale. Alternatively, there may be cases where information may be outdated.

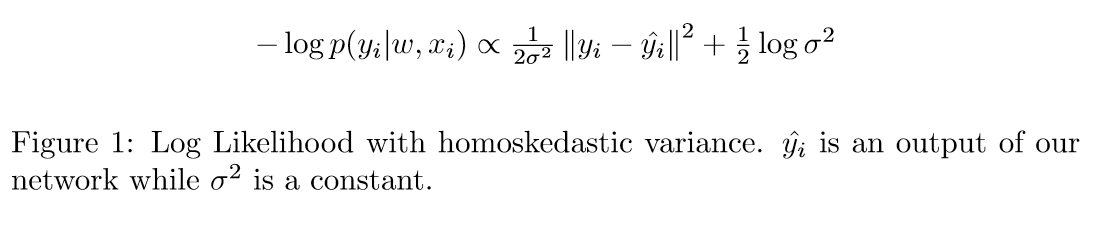
Because of the uneven noise we expect across the input features in the dataset, we predict that modeling the aleatoric variance as a function of the input will improve model performance over a Bayesian neural net that assumes constant aleatoric variance. We believe that it will also provide a more nuanced confidence of predictions that the model makes. If our model accurately model change in variance across the input feature space, the hope is that the model would be able to ask for more information, or perhaps make more educational predictions about what the level of risk associated with allocating resources towards retaining each customer is and what should be allocated given that risk.

Methods:

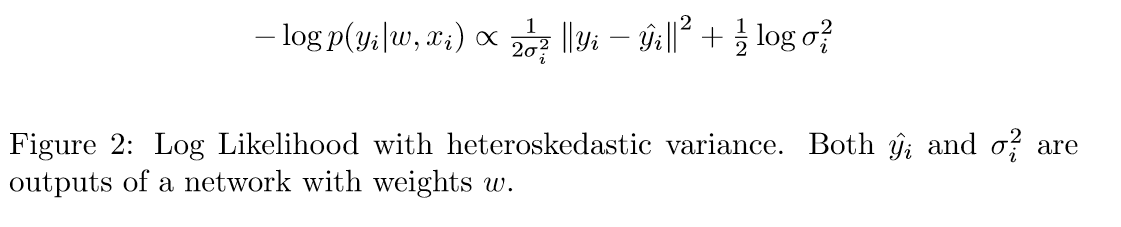
Our probabilistic model involves two different phases. First, we will encode our dataset using a variational autoencoder. Then, we will train a Bayesian neural network using the encoded data and variational inference. The variational autoencoder is used to reduce the dimensionality of the dataset and therefore shorten the required runtime of the variational inference. The implementation of the variational autoencoder will closely follow what was completed for homework 4. Variational inference will be the phase where the research group attempts to model aleatoric variance.

Each feature, xd, in our encoded space will be given a gaussian prior with a mean equal to the mean of this feature across the training set and the variance equal to the variance of this feature across the training set. (Editor’s Note: We feel uncertain if this is OK to do. We would appreciate explicit feedback. We could not think of an alternative at the time of writing).

As mentioned above, the main focus of the project is to model heteroscedastic variance as opposed to homoscedastic variance. We can incorporate these different types of variances by adjusting how we calculate our log likelihood. We know the negative log likelihood is proportional to the following equation given in figure 1.



To incorporate a heteroskedastic variance, a new equation can be formulated that uses a variance dependent on the BNN weights *w*, and input *x*.



As this new equation is still proportional to the log likelihood, it can be substituted for any optimization problem involving the log likelihood. Specifically, it can be used to incorporate aleatoric uncertainty into the elbo function traditionally used for variational inference.



By incorporating the equation in figure two into the ELBO function, we hypothesize the BNN will optimize for a heteroscedastic variance that better captures the aleatoric uncertainty in the data than if the equation in figure one was incorporated into the elbo function, which would optimize for a homoscedastic variance. In order to test if the BNN does this successfully, our main baseline will be another BNN with identical architecture. The only difference between the two will be that the baseline BNN will instead utilize the likelihood in figure one that has homoscedastic variance as opposed to the likelihood in figure two.

Additionally, the dataset being used was originally made public for a data science competition. As part of the competition, the performance of multiple baseline models on the dataset, including the company’s existing solution, was made public. Both BNN models will also be compared to these models in order to validate variational inference against other machine learning techniques.

We will use the [Orange Telecom dataset](https://www.kdd.org/kdd-cup/view/kdd-cup-2009/Data) from the [2009 Knowledge Discovery in Data Competition](http://www.mtome.com/Publications/CiML/CiML-v3-book.pdf). For this competition, a French telecom company, Orange, provided customer data with 230 features and a label for whether or not each customer will churn. The dataset has 100,000 entries with imbalanced classes--the vast majority of instances belong to the “does not churn” class. In order to protect the privacy of its customers and its proprietary information, the company removed all variable names, encoded all categorical features, and multiplied all continuous features by a scalar value. Although we do not know what any of the features in our dataset represent, this dataset is ideal to explore the effect of modeling aleatoric uncertainty, as it is large enough that issues arising from epistemic uncertainty should be minimal.

The baseline models mentioned in the report on the Knowledge Discovery in Data Competition are evaluated using AUC, so we will use this as a primary metric as well. AUC is a good metric for this task as it is an imbalanced classification task. We will make a bar plot of AUC for each model. This will allow us to compare the performance of our models, and understand whether accounting for heteroscedastic variance in our dataset results in more accurate prediction.

In order to get an approximate evaluation for how accurate the class probabilities generated from our model are, we will group our predictions by ranges of probabilities assigned by our model, and then count the number of instances from each group for which our model made the correct classification.

We will be using pytorch to implement our program. The variational inference will be written by ourselves, the variational autoencoder will be borrowed from previous work.

**Timeline:**

~~10/21 Additional literature review, confirm dataset~~

~~10/28 Noise modeling research~~

11/04 Noise modeling implementation w/ simple model

11/11 DNN prototyping

11/13 - Checkpoint 2 Due

11/18 DNN prototyping p2

11/25 Buffer

12/02 Final Presentations

12/18 Paper

Looking Ahead / Potential Roadblocks:

The Changelog and the Motivation should be relatively trivial sections to write up; we anticipate that the computing resource we use is not going to affect the performance nor training time of our models, so the decision will be made right before we start implementing a simple model (by 11/04). The motivation we expect will stay consistent with our current motivation for the project, and it will be trivial to go into more detail about our hypothesis and performance evaluation methods.

Methods and Experiments are likely to require the most time between now and checkpoint 2. We currently have a relatively good understanding of what model we are planning to use and how to implement it, but I would not be surprised if we ran into roadblocks while doing additional reading on the topic and methods we are attempting to implement such as finding flaws in our logic, such as finding that the models we are interested in unviable or finding better methods and models to do our experiments in. The best thing we can do to avoid this is to start our implementation early to get a baseline estimate of the usability of our models and continue doing readings on the topic so that if we do find ourselves needing to switch our implementation to a different model, we are prepared to do so having understood what it takes to add heteroscedasticity into the new models and successfully compare and contrast the model with and without heteroscedasticity.

We have kept track of the papers and resources we have looked at so far to get to this checkpoint, and so the References section should be similarly trivial to get through.