Homework 2: Classification and Bias-Variance Trade-offs

Eric Tang

Introduction

This homework is about classification, bias-variance trade-offs, and uncertainty quantification. In lecture we have primarily focused on binary classifiers trained to discriminate between two classes. In multiclass classification, we discriminate between three or more classes. We encourage you to read CS181 Textbook's Chapter 3 for more information on linear classification, gradient descent, and classification in the discriminative setting. Read Chapter 2.8 for more information on the trade-offs between bias and variance.

The datasets that we will be working with relate to astronomical observations. The first dataset, found at `data/planet-obs.csv`, contains information on whether a planet was observed (as a binary variable) at given points in time. This will be used in Problem 1. The second dataset, available at `data/hr.csv`, details different kinds of stars and their measured magnitude and temperature. You will work with this data in Problem 3. As a general note, for classification problems we imagine that we have the input matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ (or perhaps they have been mapped to some basis $\mathbf{\Phi}$, without loss of generality) with outputs now "one-hot encoded." This means that if there are K output classes, rather than representing the output label y as an integer $1, 2, \ldots, K$, we represent $\mathbf{y}$ as a "one-hot" vector of length K. A "one-hot" vector is defined as having every component equal to 0 except for a single component which has value equal to 1. For example, if there are $K = 7$ classes and a particular data point belongs to class 3, then the target vector for this data point would be $\mathbf{y} = [0, 0, 1, 0, 0, 0, 0]$. We will define C_1 to be the one-hot vector for the 1st class, C_2 for the 2nd class, etc. Thus, in the previous example $\mathbf{y} = C_3$. If there are K total classes, then the set of possible labels is $\{C_1 \ldots C_K\} = \{C_k\}_{k=1}^K$. Throughout the assignment we will assume that each label $\mathbf{y} \in \{C_k\}_{k=1}^K$ unless otherwise specified. The most common exception is the case of binary classification ($K = 2$), in which case labels are the typical integers $y \in \{0, 1\}$.

In problems 1 and 3, you may use `numpy` or `scipy`, but not `scipy.optimize` or `sklearn`. Example code given is in Python 3.

Please type your solutions after the corresponding problems using this LaTeX template, and start each problem on a new page.

Please submit the **writeup PDF to the Gradescope assignment 'HW2'**. Remember to assign pages for each question. **You must include your plots in your writeup PDF.** The supplemental files will only be checked in special cases, e.g. honor code issues, etc.

Please submit your **LaTeX file and code files to the Gradescope assignment 'HW2 - Supplemental'**.

Problem 1 (Exploring Bias-Variance and Uncertainty)

In this problem, we will explore the bias and variance of a few different model classes when it comes to logistic regression and investigate two sources of predictive uncertainty.

We are currently managing a powerful telescope that is being used to monitor and gather measurements of some planet of interest. At certain times however, our telescope is unable to detect the planet at all. The data in `data/planet-obs.csv` records the observation time in the "Time" column and whether the planet was detected in the "Observed" column (with the value 1 representing that it was observed). Since it is expensive to use and maintain the telescope, we would like to build a model to help us schedule and find times when we are likely to detect the planet.

1. First split the data into 10 mini-datasets of size $N = 30$ (i.e. dataset 1 consists of the first 30 observations, dataset 2 consists of the next 30, etc. This has already been done for you). Consider the three bases $\boldsymbol{\phi}_1(t) = [1, t]$, $\boldsymbol{\phi}_2(t) = [1, t, t^2]$, and $\boldsymbol{\phi}_3(t) = [1, t, t^2, t^3, t^4, t^5]$. For each of these bases, fit a logistic regression model using $\text{sigmoid}(\mathbf{w}^\top \boldsymbol{\phi}(t))$ to each dataset by using gradient descent to minimize the negative log-likelihood. This means you will be running gradient descent 10 times for each basis, once for each dataset.

   Use the given starting values of $\mathbf{w}$ and a learning rate of $\eta = 0.001$, take 10,000 update steps for each gradient descent run, and make sure to average the gradient over the data points at each step. These parameters, while not perfect, will ensure your code runs reasonably quickly.

2. After consulting with a domain expert, we find that the probability of observing the planet is periodic as the planet revolves around its star—we are more likely to observe the planet when it is in front of its star than when it is behind it. In fact, the expert determines that observation follows the generating process $y \sim \text{Bern}(f(t))$, where $f(t) = 0.4 \times \cos(1.1t + 1) + 0.5$ for $t \in [0, 6]$ and $y \in \{0, 1\}$. Note that we, the modelers, do not usually see the true data distribution. Knowledge of the true $f(t)$ is only exposed in this problem to allow for verification of the true bias.

   Use the given code to plot the true process versus your learned models. Include your plots in your solution PDF.

   In no more than 5 sentences, explain how bias and variance reflected in the 3 types of curves on the graphs. How do the fits of the individual and mean prediction functions change? Keeping in mind that none of the model classes match the true generating process exactly, discuss the extent to which each of the bases approximates the true process.

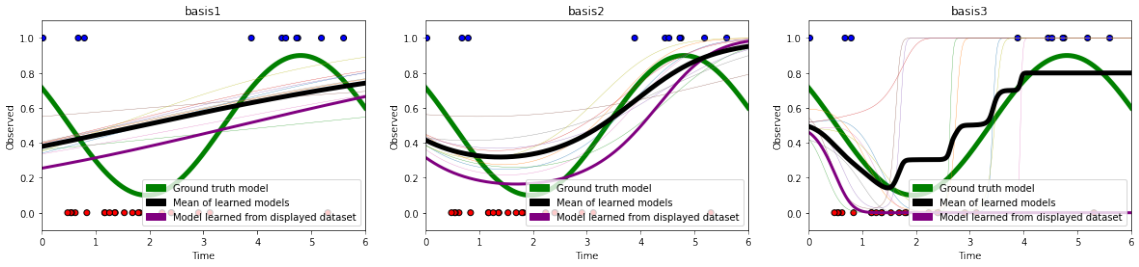
3. If we were to increase the size of each dataset drawn from $N = 30$ to a larger number, how would the bias and variance change for each basis? Why might this be the case? You may experiment with generating your own data that follows the true process and plotting the results, but this is **not** necessary. **Your response should not be longer than 5 sentences**.

4. Consider the test point $t = 0.1$. Using your models trained on basis $\boldsymbol{\phi}_3$, report the predicted probability of observation of the *first* model (the model trained on the first 30 data points). How can we interpret this probability as a measure of uncertainty? Then, compute the variance of the classification probability over your 10 models at the same point $t = 0.1$. How does this measurement capture another source of uncertainty, and how does this differ from the uncertainty represented by the classification probability?

   Repeat this process (reporting the first model's classification probability and the variance over the 10 models) for the point $t = 3.2$. At which point in time would you be more confident in detecting the planet? There's no right answer—you should consider the two different types of uncertainty and their implications when translating from model output to decision making.

1. Done! Check the supplemental code for the implementation.

2. The plots for the three respective bases are given below:



   For the first basis ϕ_1, since the linear basis is inflexible and simple, there is little to no variance (the fitted weights are somewhat oblivious to the noise in the data) but a higher bias. Since the ground truth model is very much not linear, this first basis doesn't approximate the true process very well. Moving to both the second and third bases ϕ_2 and ϕ_3, adding more polynomial terms will increase the variance of the model (the noise in the data will influence the weights more) but decrease the bias (as we see that basis 3 has the highest variance / lowest bias). Compared to ϕ_1, ϕ_2 is a better, but still not optimal, fit to the ground truth model, as it roughly follows the shape of the sinusoidal curve but fails to capture all of the noise in the ground truth model. ϕ_2 seems to be the best fit of the three bases for the ground truth model, as it not only has a decent approximation of the sinusoidal curve, but it also best captures the peaks and troughs of the true process without looking slightly jagged and rning into the concern of potentially overfitting.

3. Intuitively, increasing the size of each dataset drawn from $N = 30$ to a larger number should decrease the variance for each basis with the highest magnitude change happening in basis 3 due to it having the highest starting variance and the lowest magnitude change happening in basis 1 since it has a pretty low variance to begin with. When fitting a model, having more data should allow our constructed model to better fit the true distribution (more samples from the true distribution would intuitively imply a more certain fit). Thus, each model will have a lower variance as the model is more fit to the data and can thus be more confident in its estimate(s). Assuming the data was properly sampled, increasing the size of the dataset shouldn't influence the bias of the various models; as explained by Professor Pan, the bias of a model is more inherently tied to the complexity of the model relative to the problem as opposed to the dataset to which it is applied (thus, in our situation, we shouldn't expect bias to change too much assuming our data was correctly collected). In truly pathological examples where our data was incorrectly sampled, it is possible that bias decreases with increased sample size, but this should not be the expected scenario.

4. On $t = 0.1$, the predicted probability of observation on the first model is given by approximately 0.5199. In terms of uncertainty, this probability being close to 0.5 means that this model is quite uncertain with its prediction that the planet is observable—it essentially states that it's close to a random coin flip whether or not the planet is observable, with only a slight preference towards it being observable. You can think of this probability as the "toughness of decision" uncertainty where the model tries to convince itself to which category a datapoint belongs. The variance of the observation probability over the ten models at the same point is 0.003398. The variance measures a wholly different source of uncertainty—it measures the uncertainty of the observation probability (the uncertainty of uncertainty, if you will). You can think of this as the "quality of learning" uncertainty where it's a measurement of how much different models that were trained slightly differently agree on a particular parameter. A lower variance of the observation probability implies that the model is more confident in its observation probability and its measure of uncertainty for its hard classification. A higher variance of the observation probability, on the other hand, implies that the model is not confident in its calculated

observation probability, meaning that its measure of the uncertainty for hard classification is uncertain itself. Since the variance of this observation probability is low, the model is reasonably confident in its observation probability at $t = 0.1$.

On $t = 3.2$, the predicted probability of observation on the first model is given by approximately 0.00742, with the variance of the observation probability over the ten models at the same point is 0.245018. Since the predicted probability is very close to zero, the first model believes that at $t = 3.2$, it is very confident you will not observe the planet. However, with a high variance, this classification probability is not very reliable itself and subject to potentially significant deviation between models.

Personally, I would be more confident in detecting the planet at $t = 0.1$, as although there are a handful of models in the $t = 3.2$ case that sport a observation probability greater than that given by models generated with $t = 0.1$, the sheer variance of the models on $t = 3.2$ make it a unreliable predictor—it's hard to pin down any singular value for the classification probability. Although the average prediction probability across the ten models favors $t = 3.2$ slightly, it might not be an accurate picture of the true classification probability. In contrast, the $t = 0.1$ is very stable, and we can be reasonably confident about the observation probability at that time.

Problem 2 (Multi-class Logistic Regression and Softmax)

The objective of this problem is to generalize binary logistic regression into the more general case of three or more classes. You will use the results of this problem to implement a classifier in Problem 3.

Consider a K-class model with $K \geq 3$. Suppose we have a data set $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ with features $\{\mathbf{x}_n\}_{n=1}^N \in \mathbb{R}^d$ and one-hot encoded outputs $\{\mathbf{y}_n\}_{n=1}^N \in \mathbb{R}^K$ (see the introduction of this homework). For a K-dimensional vector $\mathbf{z} = [z_1, \ldots, z_K]^\top$, define the *softmax* function to be

$$\text{softmax}(\mathbf{z}) = \frac{1}{\sum_{i=1}^K \exp(z_i)} \begin{bmatrix} \exp(z_1) \\ \exp(z_2) \\ \vdots \\ \exp(z_K) \end{bmatrix}.$$

In other words, the softmax function is a function from $\mathbb{R}^K$ to $\mathbb{R}^K$ with k-th component of the output

$$\text{softmax}_k(\mathbf{z}) = \frac{\exp(z_k)}{\sum_{i=1}^K \exp(z_i)}.$$

We will use the shorthand notation $s_k(\mathbf{z})$ to abbreviate the above. Note that the softmax function is the general form of the sigmoid function in binary logistic regression. This means that the derivations in this problem will be very similar to what you have seen in class.

1. For $j, k \in \{1, \ldots, K\}$, show that the partial derivatives of the softmax function can be written in terms of the softmax function itself in the following form:

   $$\frac{\partial s_k(\mathbf{z})}{\partial z_j} = s_k(\mathbf{z})(\delta_{jk} - s_j(\mathbf{z}))$$

   Here, δ_{jk} denotes the *Kronecker delta* function δ_{jk}:

   $$\delta_{jk} = \begin{cases} 1 \text{ if } j = k, \\ 0 \text{ if } j \neq k. \end{cases}$$

   Since all of these partial derivatives are with respect to scalars, you can use the typical quotient rule from your univariate calculus class. It may help to consider the cases $j = k$ and $j \neq k$ separately.

   Using the answer above, find the logarithmic derivatives of softmax($\mathbf{z}$); that is, find $\dfrac{\partial \ln s_k(\mathbf{z})}{\partial z_j}$ for $j, k \in \{1, \ldots, K\}$.

Multi-class logistic regression has weights $\{\mathbf{w}_j\}_{j=1}^K \in \mathbb{R}^d$ for each class, which are often condensed into a single matrix $\mathbf{W} \in \mathbb{R}^{K \times d}$ (the j-th row of $\mathbf{W}$ corresponds to $\mathbf{w}_j$). We model the probabilities of class membership independently as

$$p(\mathbf{y}_n = C_k | \mathbf{x}_n, \mathbf{W}) = s_k(\mathbf{W}\mathbf{x}_n)$$

for $k \in \{1, \ldots, K\}$ and $n \in \{1, \ldots, N\}$. In addition, let y_{nk} denote the k-th component of the output vector $\mathbf{y}_n$.

2. Write out the negative log-likelihood of the data set, $\ell(\mathbf{W}) = -\ln p(\{\mathbf{y}_n\}_{n=1}^N | \{\mathbf{x}_n\}_{n=1}^N, \mathbf{W})$, in terms of $s_k, \mathbf{W}, \{\mathbf{x}_n\}_{n=1}^N$, and y_{nk}. You can start by noting that for a single observation $(\mathbf{x}_n, \mathbf{y}_n)$,

   $$p(\mathbf{y}_n | \mathbf{x}_n, \mathbf{W}) = \prod_{k=1}^K p(\mathbf{y}_n = C_k | \mathbf{x}_n, \mathbf{W})^{y_{nk}}$$

   because $y_{nk} = 1$ if $\mathbf{y}_n$ belongs to class C_k and $y_{nk} = 0$ otherwise. The equation above simply lets us combine all possible class memberships of $\mathbf{y}_i$ into a single expression. This is also known as the "power trick" as we express the probability as a product of terms raised to a power that is either 0 or 1.

Problem 2 (cont.)

3. Consider the weight matrix $\mathbf{W}$ and the i-th feature vector $\mathbf{x}_i$. Denote their product as $\mathbf{z}_i = \mathbf{W}\mathbf{x}_i$. Compute the derivative of $\ell(\mathbf{W})$ with respect to the j-th coordinate of the K-dimensional vector $\mathbf{z}_i$ (denoted as z_{ij}). In particular, show that

$$\frac{\partial \ell}{\partial z_{ij}} = \sum_{k=1}^{K} y_{ik}(s_j(\mathbf{z}_i) - \delta_{kj})$$

You may want to use your answer from part 1. Then, show that the above sum can be simplified:

$$\sum_{k=1}^{K} y_{ik}(s_j(\mathbf{z}_i) - \delta_{kj}) = s_j(\mathbf{z}_i) - y_{ij}.$$

It will help to consider the case $k = j$ separately again and remove the delta function from the equation.

4. Conclude that the gradient of negative log-likelihood with respect to a single weight vector $\mathbf{w}_j$ is given by the *vector*

$$\frac{\partial \ell}{\partial \mathbf{w}_j} = \sum_{n=1}^{N} (s_j(\mathbf{W}\mathbf{x}_n) - y_{nj})\mathbf{x}_n$$

for $j \in \{1, \ldots, K\}$. This can be done by using the chain rule:

$$\frac{\partial \ell}{\partial \mathbf{w}_j} = \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{\partial \ell}{\partial z_{nk}} \frac{\partial z_{nk}}{\partial \mathbf{w}_j}.$$

We found the first derivative in part 3. What do we know about the second derivative when $k \neq j$? You can start by expressing z_{nk} in terms of the vectors $\{\mathbf{w}_i\}_{i=1}^{K}$ and $\{\mathbf{x}_i\}_{i=1}^{N}$.

We can use this final expression to optimize the weights via gradient descent!

1. Consider the case where $j = k$ first. By the quotient rule, we have

$$\begin{aligned}\frac{\partial s_k(\mathbf{z})}{\partial z_k} &= \frac{(\sum_{i=1}^K \exp(z_i))\exp(z_k) - \exp(z_k)\cdot\exp(z_k)}{(\sum_{i=1}^K \exp(z_i))^2} \\ &= \frac{\exp(z_k)}{\sum_{i=1}^K \exp(z_i)} - \frac{\exp(z_k)^2}{(\sum_{i=1}^K \exp(z_i))^2} \\ &= s_k(\mathbf{z}) - (s_k(\mathbf{z}))^2 \\ &= s_k(\mathbf{z})(1 - s_k(\mathbf{z})).\end{aligned}$$

   Looking at the case when $j \neq k$, the quotient rule gives us

$$\begin{aligned}\frac{\partial s_k(\mathbf{z})}{\partial z_j} &= \frac{-\exp(z_k)\cdot\exp(z_j)}{(\sum_{i=1}^K \exp(z_i))^2} \\ &= -\frac{\exp(z_k)}{\sum_{i=1}^K \exp(z_i)} \cdot \frac{\exp(z_j)}{\sum_{i=1}^K \exp(z_i)} \\ &= -s_k(\mathbf{z})s_j(\mathbf{z}) \\ &= s_k(\mathbf{z})(0 - s_j(\mathbf{z})).\end{aligned}$$

   Combining these values together, we get our desired result that

$$\frac{\partial s_k(\mathbf{z})}{\partial z_j} = s_k(\mathbf{z})(\delta_{jk} - s_j(\mathbf{z}))$$

   where δ_{jk} denotes the Kronecker delta. Using this result, we can apply the chain rule to get

$$\frac{\partial \ln s_k(\mathbf{z})}{\partial z_j} = \frac{1}{s_k(\mathbf{z})} \cdot \frac{\partial s_k(\mathbf{z})}{\partial z_j} = \delta_{jk} - s_j(\mathbf{z}).$$

2. Given the expression

$$p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{W}) = \prod_{k=1}^K p(\mathbf{y}_n = C_k|\mathbf{x}_n, \mathbf{W})^{y_{nk}}$$

   for a single observation, assuming conditional independence between observations, the joint probability of all of the n observations is given by

$$p(\{\mathbf{y}_n\}_{n=1}^N|\{\mathbf{x}_n\}_{n=1}^N, \mathbf{W}) = \prod_{n=1}^N \prod_{k=1}^K p(\mathbf{y}_n = C_k|\mathbf{x}_n, \mathbf{W})^{y_{nk}},$$

   which, given the expression for the individual probabilities, is given by

$$p(\{\mathbf{y}_n\}_{n=1}^N|\{\mathbf{x}_n\}_{n=1}^N, \mathbf{W}) = \prod_{n=1}^N \prod_{k=1}^K (s_k(\mathbf{W}\mathbf{x}_n))^{y_{nk}}.$$

   The negative log likelihood is thus

$$\begin{aligned}l(\mathbf{W}) &= -\ln p(\{\mathbf{y}_n\}_{n=1}^N|\{\mathbf{x}_n\}_{n=1}^N, \mathbf{W}) \\ &= -\ln \prod_{n=1}^N \prod_{k=1}^K (s_k(\mathbf{W}\mathbf{x}_n))^{y_{nk}} \\ &= -\sum_{n=1}^N \sum_{k=1}^K \ln\left((s_k(\mathbf{W}\mathbf{x}_n))^{y_{nk}}\right) \\ &= -\sum_{n=1}^N \sum_{k=1}^K y_{nk} \ln(s_k(\mathbf{W}\mathbf{x}_n)).\end{aligned}$$

3. Given the substitution for $\mathbf{W}\mathbf{x}_n$, we have

$$l(\mathbf{W}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk}\ln(s_k(\mathbf{W}\mathbf{x}_n)) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk}\ln(s_k(\mathbf{z}_n)).$$

The derivative of l with respect to z_{ij} is given as

$$\begin{aligned}
\frac{\partial l}{\partial z_{ij}} &= \frac{\partial}{\partial z_{ij}}\left(-\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk}\ln(s_k(\mathbf{z}_n))\right) \\
&= \frac{\partial}{\partial z_{ij}}\left(-\sum_{k=1}^{K} y_{ik}\ln(s_k(\mathbf{z}_i))\right) \\
&= -\sum_{k=1}^{K} y_{ik}\cdot\frac{\partial}{\partial z_{ij}}\ln(s_k(\mathbf{z}_i)) \\
&= -\sum_{k=1}^{K} y_{ik}(\delta_{jk} - s_j(\mathbf{z}_i)) \\
&= \sum_{k=1}^{K} y_{ik}(s_j(\mathbf{z}_i) - \delta_{jk}),
\end{aligned}$$

applying our result from part 1 in the fourth equality, giving our desired result. Moving forward, consider the inside term of the sum. In the case when $k = j \Rightarrow \delta_{jk} = 1$. We have

$$y_{ik}(s_j(\mathbf{z}_i) - \delta_{jk}) = y_{ij}(s_j(\mathbf{z}_i) - 1) = y_{ij}s_j(\mathbf{z}_i) - y_{ij}.$$

When $k \neq j \Rightarrow \delta_{jk} = 0$, we have

$$y_{ik}(s_j(\mathbf{z}_i) - \delta_{jk}) = y_{ik}s_j(\mathbf{z}_i).$$

Thus, the above sum simplifies into

$$\begin{aligned}
\frac{\partial l}{\partial z_{ij}} &= \sum_{k=1}^{K} y_{ik}(s_j(\mathbf{z}_i) - \delta_{jk}) \\
&= \sum_{k=1}^{K} (y_{ik}s_j(\mathbf{z}_i)) - y_{ij} \\
&= s_j(\mathbf{z}_i)\sum_{k=1}^{K} (y_{ik}) - y_{ij} \\
&= s_j(\mathbf{z}_i) - y_{ij},
\end{aligned}$$

where the last inequality follows from the fact that y_i is only in one class. Thus, we have our desired expression, and we are done.

4. As suggested, let's find an expression for z_{nk} to determine the second derivative in the chain rule. We have

$$\begin{aligned}
\mathbf{z}_n &= \mathbf{W}\mathbf{x}_n \\
\Rightarrow \mathbf{z}_{nk} &= \mathbf{w}_k\mathbf{x}_n \\
\Rightarrow \frac{\partial z_{nk}}{\partial \mathbf{w}_j} &= \begin{cases} \mathbf{x}_n \text{ if } k = j \\ 0 \text{ otherwise.} \end{cases}
\end{aligned}$$

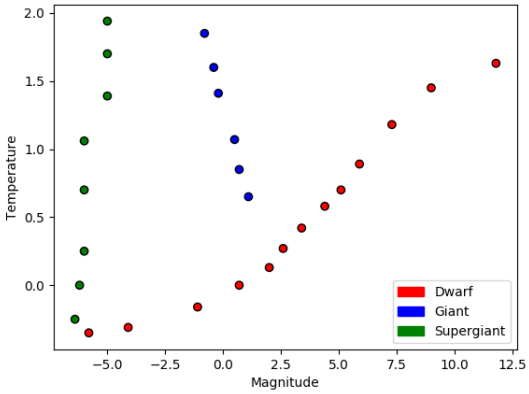
Thus, given the chain rule expression, we have

$$\begin{aligned}\frac{\partial \ell}{\partial \mathbf{w}_j} &= \sum_{n=1}^{N}\sum_{k=1}^{K}\frac{\partial \ell}{\partial z_{nk}}\frac{\partial z_{nk}}{\partial \mathbf{w}_j} \\ &= \sum_{n=1}^{N}(s_j(\mathbf{z}_n) - y_{nj})\mathbf{x}_n \\ &= \sum_{n=1}^{N}(s_j(\mathbf{W}\mathbf{x}_n) - y_{nj})\mathbf{x}_n,\end{aligned}$$

and we are done.

Problem 3 (Classifying Stars)

In this problem, you will code up three different classifiers to classify different types of stars. The file `data/hr.csv` contains data on magnitude and temperature. The data can be plotted on these two axes:



Please implement the following classifiers in the `SoftmaxRegression` and `KNNClassifier` classes:

a) **A multi-class logistic regression classifier** using the softmax activation function, which you investigated in Problem 2. In your implementation of gradient descent, **make sure to include a bias term and use L2 regularization** with regularization parameter $\lambda = 0.001$. Limit the number of iterations of gradient descent to 200,000, and set the learning rate to be $\eta = 0.001$.

b) **Another multi-class logistic regression classifier** with feature map $\phi(\mathbf{x}) = [\ln(x_1 + 10), x_2^2]^\top$, where x_1 and x_2 represent the values for magnitude and temperature, respectively.

c) **A kNN classifier** in which you classify based on the $k = 1$ and $k = 5$ nearest neighbors and the following distance function:

$$dist(star_1, star_2) = (mag_1 - mag_2)^2/9 + (temp_1 - temp_2)^2$$

where nearest neighbors are those with the smallest distances from a given point.

Note 1: When there are more than two labels, no label may have the majority of neighbors. Use the label that has the most votes among the neighbors as the choice of label.

Note 2: The grid of points for which you are making predictions should be interpreted as our test space. Thus, it is not necessary to make a test point that happens to be on top of a training point ignore itself when selecting neighbors.

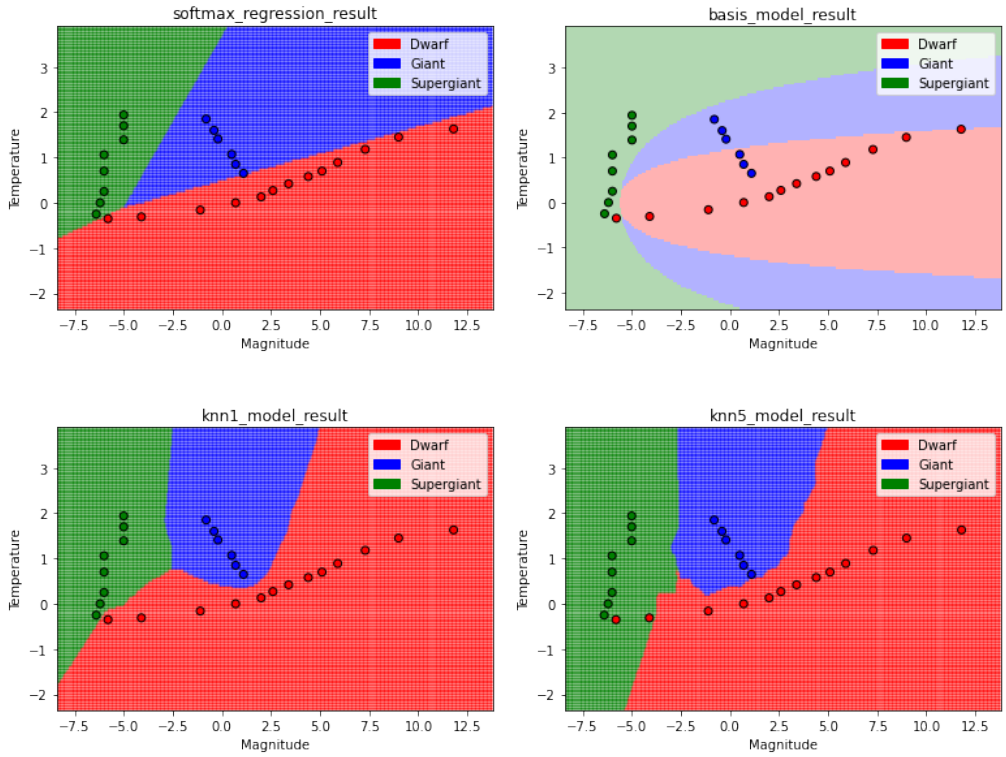
After implementing the above classifiers, complete the following exercises:

1. Plot the decision boundaries generated by each classifier for the dataset. Include them in your PDF. Identify the similarities and differences among the classifiers. What explains the differences—in particular, which aspects or properties of each model dictate the shape of its decision boundary?

2. Consider a star with Magnitude 3 and Temperature -2. To which class does each classifier assign this star? Report the classification probabilities of this star for each model.

   Interpret how each model makes its classification decision. What are the pros and cons of each interpretation? What else should we, the modelers, be aware of when making predictions on a test point "far" from our training data? **Your response should no be longer than 5 sentences.**

1. The plots of the various decision boundaries is given below:



Note that the softmax regression classifier defined a boundary that seems linear, whereas all of the other models have curved decision boundaries. This observation is just because of how the classification probability is determined—the decision boundaries define when the classification between two or more classes is the most uncertain, which, in the logistic regression framework, gives a decision boundary determined by a linear relationship. The only reason the basis model no longer follows these linear decision boundaries is because it's been translated by φ, which gives the curved, quadratic-like curves we see here. Note that although I described these curves are quadratic-like for simplicity, we can be more specific. The translation by φ gives a natural log curve (from the magnitude translation) that gets mirrored over the temperature equals 0 horizontal (from the temperature translation). On the other hand, the KNN classifiers only care about the distance to the nearest points —their decision boundaries will necessarily be more fluid. For example, in the $k = 1$ scenario, the decision boundaries are defined as the midpoint of the two nearest points. In terms of comparing their performances, note that the softmax and knn1 plots are the only plots to have what perfect (if not nearly perfect) accuracy classifying the stars. Although their decision boundaries are defined differently, knn1 and softmax seem to correctly identify all of the points in X_{stars} (though there is one Giant point in softmax that is very close to the decision boundary). In knn5 and the basis model, a handful of points were misclassified (two red points were closer to the cluster of greens in knn5 and the transformation caused the Giant datapoints to be too close to the Dwarf datapoints in the basis model).

2. The softmax, knn1, and knn5 models all classify this star as a Dwarf, while the basis model classifies this star as a Giant. The classification probabilities for these models are given by

Models				
Stars	Softmax	Basis	KNN1	KNN5
Dwarf	1.00000	0.03425432	1	1
Giant	1.3×10^{-28}	0.96455651	0	0
Supergiant	2.6×10^{-40}	0.00118917	0	0

In terms of the interpretations of each model, the pros of the logistic regression classifier is that it offers a better reflection of uncertainty regarding its classification, giving the probability that a given prediction could be in a different class than predicted. Although KNN does do this to some degree, for small values like $k = 1$ or $k = 5$, this notion of uncertainty only encapsulates the information from a handful of neighboring points as opposed to the entire classes—we do not get as much information regarding the probabilities of the point belonging to another class. On the flip side, a con of the logistic regression and a pro of the KNN classifier is interpretability—it's much easier to interpret the reasoning behind the KNN classifier's prediction (visual inspection suffices), while the interpretation of the logistic regression classifier is more mathematically involved.

As modellers making predictions on a test point far from our training data, we should be wary about the certainty our models give for predictions on these test points. Extrapolating a fitted model beyond the range of the training set can lead to seriously biased estimates if the assumed relationships determined in the training set do not hold (or have different importance) in the relevant region in which you are trying to extrapolate.

Problem 4 (Impact Question: Understanding model-assisted decision making, uncertainty in classification and model interpretation in a high-stakes situation)

Prompt: A pharmaceutical drug company is conducting a clinical drug trial for a devastating disease for which conventional treatment is often ineffective. They want to estimate the effectiveness of a new drug on patients in order to obtain FDA approval and release the drug to the market. They approach you with the results of their clinical trial conducted on 100 patients with features and labels as follows:

Features = {*age, sex, height, blood pressure, drug administered?*}
Label = {*Did the patient get cured?*}

Since testing on a larger patient population is expensive for various reasons, they are interested in developing a machine learning model that will estimate the effectiveness of the drug. They provide you with covariate values of a single unseen patient for testing model performance.

1. You fit a logistic regression model on the 100 observations from the clinical trial and obtain the following coefficients which minimize the negative log likelihood. Let us call this model A:

   $p(y = 1|\mathbf{x}) = \sigma(0.2 + 0.8 * \textit{drug administered?} - 0.012 * \textit{age} + 0.45 * \textit{sex} + 0.001 * \textit{height} - 0.007 * \textit{blood pressure})$

   Say you have a female (coded as 0) patient who is age 50, 168cm tall, blood pressure 140. What is the change in classification probability of the patient getting cured when they are administered the drug versus not (please show your work)? Use your answer to formulate an interpretation of the value of w_1 – the coefficient for *drug administered?* – for stakeholders in the drug company. The drug company wants you to answer whether or not you see evidence for the efficacy of their drug – what would you say?

2. The drug company wants to know you how confident you are that you have the right model, so you decide to sample the existing dataset with replacement to train 100 bootstrapped models. Upon testing these 100 models on the unseen patient, you find that the predictive interval of the classification probability is around ± 0.35 averaged over the bootstrapped models. The drug company is concerned and asks you to check if you can choose alternative models that are more confident in their predictions:

   (a) You first try adding some interaction terms and train this new model B on the original dataset:

   $p(y = 1|\mathbf{x}) = \sigma(-2 + 5 * \textit{drug administered?} + 2 * \textit{age} - 3.3 * \textit{sex} + 0.001 * \textit{height} + 0.2 * \textit{blood pressure} - 0.12 * \textit{age} * \textit{sex} - 0.34 * \textit{height} * \textit{sex})$

   Bootstrapping model B 100 times gives you a new predictive interval of ± 0.1 (averaged over bootstrapped models). Why might this be happening – how would you explain this reduction in uncertainty in model B?

   (b) Encouraged by the success of adding interaction terms, you add all possible combinations of interaction terms, repeat the exercise of bootstrapping 100 times and call this model C. This gives you a predictive interval of ± 0.39 averaged over the bootstrapped models. Why might this be happening – what is the source of this rise in uncertainty? What is a solution to reduce this comparatively large uncertainty, if you wished to keep all the new terms?

   (c) Which model (B or C, assuming you can reduce the predictive interval for model C) would you recommend to the drug company and why?

3. Assume that the drug company has picked model B as their final model of choice because it seems the most confident in its predictions.

   (a) Suppose that the confidence interval of your estimate of w_1 (coefficient for *drug administered*?) is ± 6. What would you recommend to a critically ill patient who is desperately seeking access to this new (and very expensive, not-covered-by-insurance) drug and why?

   (b) The drug company has strong reasons to believe that the drug is indeed effective. What can the drug company do during the clinical trial to tighten the confidence interval of w_1?

Problem 4 (cont.)

3. (Continued...)

   (c) From prior clinical trials and modeling efforts, the drug company scientists strongly believe that the real value of w_1 is either 5 or 1. For now, you take this information as ground truth. In which scenario ($w_1 = 1$ or $w_1 = 5$) is it more costly to run the drug trial and demonstrate the effectiveness of the drug? Why?
   Hint: In which case do you need your confidence intervals for the estimates of w_1 to be tighter, to demonstrate drug effectiveness?

   (d) How do you think we should make use of domain knowledge provided by our partners – that the real value of w_1 is either 5 or 1 – during model building? For example, would it be a good idea for us to simply fix the parameter w_1 to be 5 or 1?

4. Let us revisit the data collection process during the clinical trial and consider data collected from two recruitment protocols. In the first recruitment protocol, the drug company publicly advertised the development of this drug to a hospital, encouraging interested patients to sign up for the trial. Among an audience of 100, 50 patients agreed to be administered the drug and 50 choose to opt out and are observed in the study without administering the drug. In the second recruitment protocol, the company was referred 100 patients who have the disease and administered the drug to each patient randomly based on a coin flip. When you train model B on the first dataset, it has a high value of w_1 and a small confidence interval of this estimate. In contrast, when you train model B on the second dataset, you get a smaller value for w_1 with an equally small confidence interval. Which model would you trust to predict the probability of being cured from the disease given a randomly selected new patient from a Boston hospital? Why? *Hint:* What confounding factor might we have here?

1. The classification probability assuming they are administered a drug is given by

$$\begin{aligned} p(y=1|\mathbf{x}) &= \sigma(0.2 + 0.8 \cdot 1 - 0.012 \cdot 50 + 0.45 \cdot 0 + 0.001 \cdot 168 - 0.007 \cdot 140) \\ &= \sigma(-0.412) \\ &= \frac{1}{1+e^{0.412}} \\ &\approx 0.398, \end{aligned}$$

while the classification probability assuming they are not administered a drug is given by

$$\begin{aligned} p(y=1|\mathbf{x}) &= \sigma(0.2 + 0.8 \cdot 0 - 0.012 \cdot 50 + 0.45 \cdot 0 + 0.001 \cdot 168 - 0.007 \cdot 140) \\ &= \sigma(-1.212) \\ &= \frac{1}{1+e^{1.212}} \\ &\approx 0.229. \end{aligned}$$

For interpretations, without the drug, this patient is predicted to be cured with probability of approximately 22.9%, while with the drug, the patient is predicted to be cured with probability approximately 39.8%, a rather significant increase. When interpreting the value of w_1, we see that this increase is expected—w_1 is the highest coefficient by a significant margin within the model, indicating that the presence or lack of the drug is associated with the most predictive power—both compared to the categorical and quantitative factors—in predicting a patient's chances of being cured.

Since we performed an analysis on only one individual, this analysis itself does not provide efficacy of their drug, as studying this drug on a single-unit sample size will not generalize well to the population at-large. Even if we looked at the weights of the model itself from an overall perspective, the high weight associated with the presence of the drug does not necessarily imply their drug is effective—if the model is very uncertain about this weight (e.g. the confidence interval is wide), the seeming significance of the presence of this drug may just be due to pure chance.

2. (a) Adding these interaction terms necessarily adds complexity to the model and offers additional opportunities to find weights optimized to the noise in the dataset. For an potential explanation, it is possible that the initial model was somewhat underfit to the data, benefiting from the added complexity of the interaction terms, giving the correspondingly tighter predictive interval.

   (b) Just as above, adding additional terms in your model necessarily gives it more opportunity to optimize to the noise in the dataset. However, adding too many predictive terms allows for overfitting to occur, often increasing variance of the weights to balloon the predictive interval. For a potential explanation, it's possible that adding all these additional interaction terms produced an overfit model that was too sensitive to the noise in the training set. If we want to keep all of these new terms while reducing the uncertainty, we could either obtain a larger dataset, which hopefully would help reduce the variance of the overfit model, or we could introduce some regularization which again helps combat overfitting.

   (c) Even assuming we could reduce the predictive interval for C to a comparable level as B, I would recommend the drug company to use model B for its interpretability. Usually, adding additional terms offers the chance for the model to fit better to the data at the cost of interpretability. However, in this case, model C has no predictive benefit over model B, and only serves to confuse any engineer further as to the interpretability of the model and how each predictor affected the prediction probability. If we were able to somehow make the predictive interval for C to be better than that of B, I would prefer C, as even though it might be less interpretable, in the case of patient success with drugs, I would prefer a more confident estimate over interpretability just due to safety concerns.

3. (a) Since the confidence interval of our estimate covers the value 0 (and even dips into the negative), I would have to ensure that the critically ill patient understands that this model does not necessarily imply this new drug will help their chances of being cured and, potentially, might not do anything or even harm their chances of being cured. Whether or not the patient decides to take the risk of this new drug either doing nothing or actually harming them at the (assumedly) expensive cost of the treatment is ultimately a decision up to them, but personally I would not be comfortable endorsing the efficacy of this drug knowing that the confidence interval does not exclude negative values or 0 and thus could, in reality, do nothing or harm the patient's chances.

   (b) If the drug company wants to tighten the confidence interval for the weight, they should increase the sample size in the clinical trial to obtain more data, as increasing the sample size a model is trained on will typically decrease the size of the produced confidence intervals. Especially when they have strong reasons that the drug is indeed effective, increasing the sample size should not significantly increase ethical concerns (e.g. the drug actually harming the participants).

   (c) The scenario when $w_1 = 1$ is much more costly to run the drug trial. Since 1 is closer to 0 than 5, the confidence intervals for w_1 must be tighter when it is actually 1 than when it's actually 5, as we need to exclude the value 0 to be reasonably assured of drug effectiveness. In order for us to get tighter confidence intervals, that usually involves a larger sample size / a larger study, which necessarily would increase the cost.

   (d) No, it is not a good idea for us to simply fix the parameter to be 5 or 1, as even though the drug company strongly believes it to be the case, the true value of w_1 still very much be different. However, if it is backed by previous data, it still is helpful, perhaps as a prior in modelling. We might also be ethically obligated to verify this domain knowledge by reproducing these results ourselves!

4. I would trust the model trained on the second dataset. The primary distinction between the two datasets is that in the first datasets, patients would opt into the treatment, which could potentially introduce important confounding factors. For instance, it is entirely possible that patients with more severe cases of the disease were more likely to opt into the treatment (as they believe the benefits are worth the risks) while patients with less severe cases are more likely to opt out (as they believe the risks outweigh the benefits). Thus, it is entirely possible, in this scenario, that the higher value of w_1 in the first dataset is caused by it being more effective at elevating curing probability for more severe cases of the disease. Thus, the value generated in the first dataset is more prone to a degree of selection bias compared to the value generated in the second dataset. Hence, since the confidence intervals are comparable and the second collection process is more resilient against selection bias, I would prefer the second value of w_1!

Name

Eric Tang

Collaborators and Resources

Dacha Thurber, Shmu Padwa, Michal Kurek, Phevos Paschalidis

Calibration

15 hours