Bayesian statistics on Student Data

Introduction

In this project, i will be exploring student performance data collected from a higher secondary school.The data includes personal and academic characteristics of students along with final class grades.We will implement a Bayesian Linear Regression as part of this project.Throughout the project, i will be going through various machine learning process like cleaning the data, exploring it to find trends, establishing a baseline model, implementing Bayesian Linear Regression, interpreting the results, and presenting the results

Exploratory Data Analysis

I am using data on student grades collected from a Portuguese secondary (high) school. This data is from the UCI machine learning repository [1], a great collection of datasets for model testing. The data includes academic and personal characteristics of the students as well as final grades. The objective is to predict the final grade from the student information which makes this a regression task. We have a set of training data with known labels, and we want the model to learn a mapping from the features (explanatory variables) to the target (the label) in this case the final grade. It is a regression task because the final grade is a continuous value.Looking at the data, we can see that there are a total of 633 observations with 33 variables. Each row is one student with each column containing a different characteristic.We will treat the grade as continuous which makes this a regression problem.
The primary variable of interest is the grade, so let's take a look at the distribution to check for skew: The grades are close to normally distributed with a mode at 11 (the

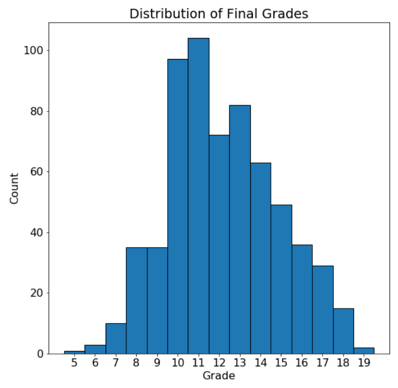
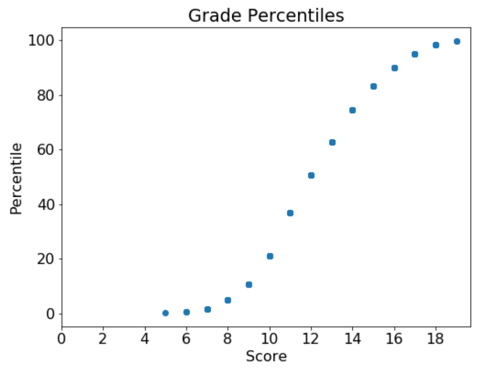
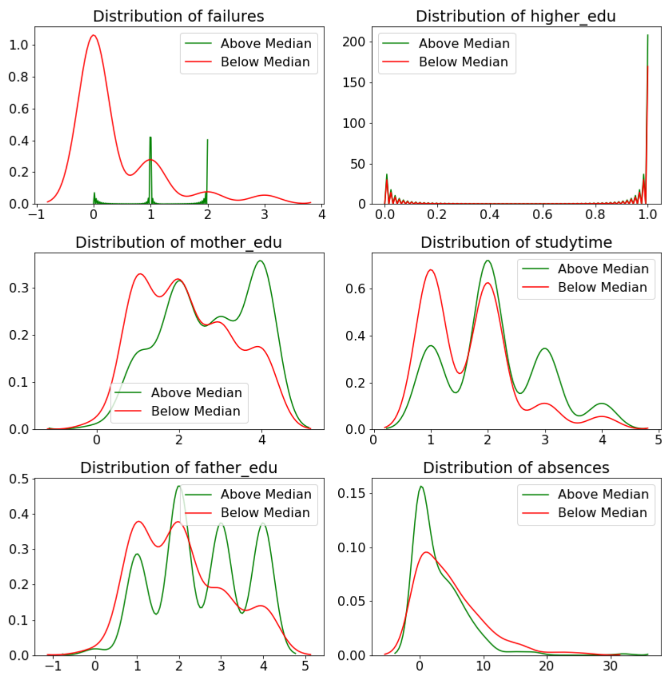
Distribution of Final Grades	**Distribution of Grade Percentiles**
	

Table 1: Distribution of Grades

grading scale in this school goes from 0–20). While the overall grades do not have a noticeable skew, it's possible that students from certain categories will have skewed grades. To look at the effect of categorical variables on the grade we can make density plots of the grade distribution colored by the value of the categorical variable.

Feature Selection

We don't expect every variable to be related to the final grade, so we need to perform feature selection (also called dimensionality reduction) to choose only the "relevant" variables. This depends on the problem, but because we will be doing linear modeling in this project, we can make distribution plots of each variable, coloring the plot by if the grade is above the median score of 12. To make these plot, we create a column in our dataframe comparing the grade to to 12 and then plot all the values in density plots.This yields the following plots: The green distributions represent students with grades at or



above the median, and the red is students below. We can see that some variables are more positively correlated with grades (such as studytime), while others are indicators of low grades, such as low *father_edu*.

Establish Baselines Metrics

For regression, a good naive baseline is simply to guess the median value of the target for every observation in the test data. In our problem, the median is 12, so let's assess the accuracy of a model that naively predicts 12 for every student on the test set. We will use 2 metrics to evaluate predictions:
Mean Absolute Error (MAE): The average of the absolute value of the differences between the predictions and true values.
Root Mean Squared Error (RMSE): The square root of the average of the squared differences between the predictions and true values.
The mean absolute error is easily interpretable, as it represents how far off we are on average from the correct value. The root mean squared error penalizes larger errors more heavily and is commonly used in regression tasks. Either metric may be appropriate depending on the situation [2] and we will use both for comparison.
When we predict 12 for every example on the test set we get these results:

$$Median \quad Baseline \quad MAE : 2.1761$$

$$Median \quad Baseline \quad RMSE : 2.6777$$

Linear Regression is the simplest machine learning technique, and does not perform well on complex, non-linear problems with lots of features, but it has the benefit of being easily explained. We can extract the prediction formula from the linear regression using the trained model. Following is the formula:

Grade $= 9.19 + -1.32 \times failures + 1.86 \times higher\_edu + 0.26 \times mother\_edu + 0.58 \times studytime + 0.03 \times father\_edu + -0.07 \times absences ------(1)$

Bayesian Linear Regression

Let's briefly recap Frequentist and Bayesian linear regression. The Frequentist view of linear regression assumes data is generated from the following model:

$$\mathbf{y} = \beta^{\mathbf{T}}\mathbf{X} + \epsilon$$

Where the response, y, is generated from the model parameters, β, times the input matrix, X, plus error due to random sampling noise or latent variables. In the ordinary least squares (OLS) method, the model parameters, β, are calculated by finding the parameters which minimize the sum of squared errors on the training data. The output from OLS is single point estimates for the "best" model parameters given the training data. These parameters can then be used to make predictions for new data points.
In contrast, Bayesian Linear Regression assumes the responses are sampled from a

probability distribution such as the normal (Gaussian) distribution:

$$y \sim N(\beta^T X, \sigma^2)$$

The mean of the Gaussian is the product of the parameters, β and the inputs, X, and the standard deviation is σ. In Bayesian Models, not only is the response assumed to be sampled from a distribution, but so are the parameters. The objective is to determine the posterior probability distribution for the model parameters given the inputs, X, and outputs, y:

$$\mathbf{P}(\beta|\mathbf{y},\mathbf{X}) = \frac{\mathbf{P}(\mathbf{y}|\beta,\mathbf{X}) \times \mathbf{P}(\beta|\mathbf{X})}{\mathbf{P}(\mathbf{y}|\mathbf{X})}$$

The posterior is equal to the likelihood of the data times the prior for the model parameters divided by a normalization constant. If we have some domain knowledge, we can use it to assign priors for the model parameters, or we can use non-informative priors: distributions with large standard deviations that do not assume anything about the variable. Using a non-informative prior means we "let the data speak." A common prior choice is to use a normal distribution for β and a half-cauchy distribution for σ .
In practice, calculating the exact posterior distribution is computationally intractable for continuous values and so we turn to sampling methods such as Markov Chain Monte Carlo (MCMC) to draw samples from the posterior in order to approximate the posterior. Monte Carlo refers to the general technique of drawing random samples, and Markov Chain means the next sample drawn is based only on the previous sample value. The concept is that as we draw more samples, the approximation of the posterior will eventually converge on the true posterior distribution for the model parameters.The end result of Bayesian Linear Modeling is not a single estimate for the model parameters, but a distribution that we can use to make inferences about new observations. This distribution allows us to demonstrate our uncertainty in the model and is one of the benefits of Bayesian Modeling methods. As the number of data points increases, the uncertainty should decrease, showing a higher level of certainty in our estimates.

Implementing Bayesian Linear Modeling in Python

The best library for probabilistic programming and Bayesian Inference in Python is currently PyMC3. It includes numerous utilities for constructing Bayesian Models and using MCMC methods to infer the model parameters. We will be using the Generalized Linear Models (GLM) module of PyMC3, in particular, the $GLM.from\_formula$ function which makes constructing Bayesian Linear Models extremely simple.
There are only two steps we need to do to perform Bayesian Linear Regression with this module:
1) Build a formula relating the features to the target and decide on a prior distribution for the data likelihood

2) Sample from the parameter posterior distribution using MCMC
Instead of having to define probability distributions for each of the model parameters separately, we pass in an R-style formula relating the features (input) to the target (output). Here is the formula relating the grade to the student characteristics:

Grade $\sim$ failures + $higher\_edu + mother\_edu + studytime + father\_edu + absences$
Once the GLM model is built, we sample from the posterior using a MCMC algorithm. If we do not specify which method, PyMC3 will automatically choose the best for us. In the code , I let PyMC3 choose the sampler and specify the number of samples, 2000, the number of chains, 2, and the number of tuning steps, 500.
The sampler runs for a few minutes and our results are stored in *normal_trace*. This contains all the samples for every one of the model parameters (except the tuning samples which are discarded)[3].The trace is essentially our model because it contains all the information we need to perform inference. To get an idea of what Bayesian Linear Regression does, we can examine the trace using built-in functions in PyMC3.
A traceplot shows the posterior distribution for the model parameters on the left and the progression of the samples drawn in the trace for the variable on the right. The two colors represent the two difference chains sampled. Traceplots for all samples are available in the Appendix section. Here we can see that our model parameters are not point estimates but distributions. The mean of each distribution can be taken as the most likely estimate, but we also use the entire range of values to show we are uncertain about the true values. Now we get Linear Formula from Bayesian Inference using Mean of Parameters as :
$Grade = 9.18 * Intercept + -1.31 * failures + 1.87 * higher\_edu + 0.27 * mother\_edu + 0.58 * studytime + 0.03 * father\_edu + -0.07 * absences + 0.82 * sd\_log + 2.28 * sd ------(2)$

Predictions

When it comes to predicting, the Bayesian model can be used to estimate distributions. We remember that the model for Bayesian Linear Regression is:

$$y \sim N(\beta^T X, \sigma^2)$$

Where β is the coefficient matrix (model parameters), X is the data matrix, and σ is the standard deviation. If we want to make a prediction for a new data point, we can find a normal distribution of estimated outputs by multiplying the model parameters by our data point to find the mean and using the standard deviation from the model parameters.
In this case, we will take the mean of each model parameter from the trace to serve as the best estimate of the parameter. If we take the mean of the parameters in the trace, then the distribution for a prediction becomes:

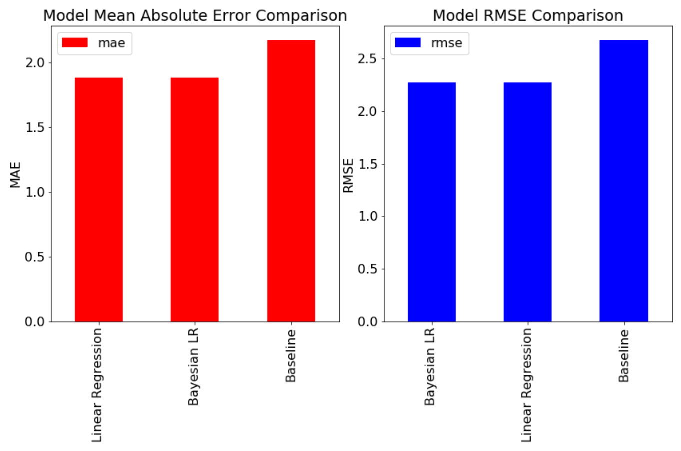
$Grade \sim N(9.18 * Intercept - 1.31 * failures + 1.87 * higher\_edu + 0.27 * mother\_edu + 0.58 * studytime + 0.03 * father\_edu - 0.07 * absences, 2.28^2)$

For a new data point, we substitute in the value of the variables and construct the probability density function for the grade. As an example, here is an observation from the test set along with the probability density function For the new covariates absences = 1, failures = 0, $father\_edu$= $1higher\_edu = 1, mother\_edu = 4, studytime = 3$ we get the grade predicted as :

$$\begin{aligned} Average \quad Estimate &= 13.8039 \\ 95\% \quad Credible \quad set \quad &= [10.0634, 17.5984] \end{aligned}$$

Comparison to Standard Machine Learning Models

To calculate the MAE and RMSE metrics, we need to make a single point estimate for all the data points in the test set. We can make a "most likely" prediction using the means value from the estimated distributed. The resulting metrics, along with those of the benchmarks, are shown below: Bayesian Linear Regression achieves nearly the same



performance as the best standard models! However, the main benefits of Bayesian Linear Modeling are not in the accuracy, but in the interpretability and the quantification of our uncertainty.

Conclusions

In this project, we walked through the complete machine learning process used to solve a data science problem. We started with exploratory data analysis, moved to establishing a baseline, tried out several different models, implemented our model of choice, interpreted the results, and used the model to make new predictions.

References

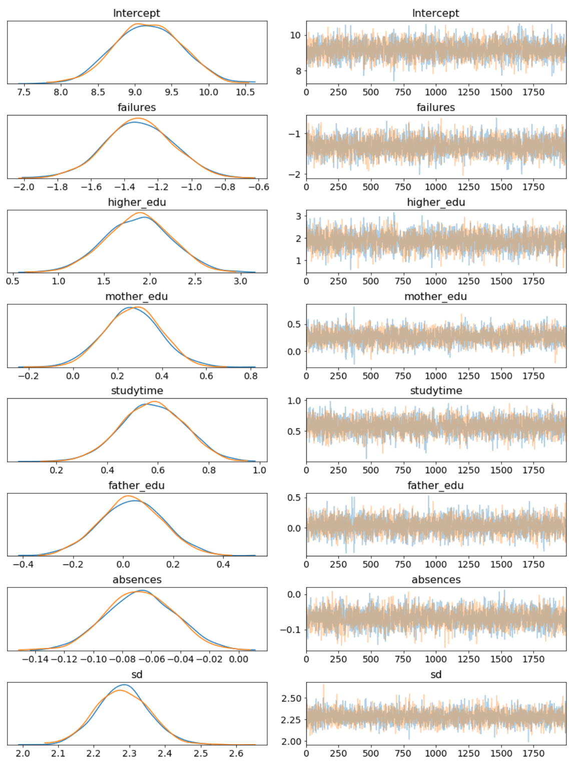
[1] https://archive.ics.uci.edu/ml/datasets/student+performance

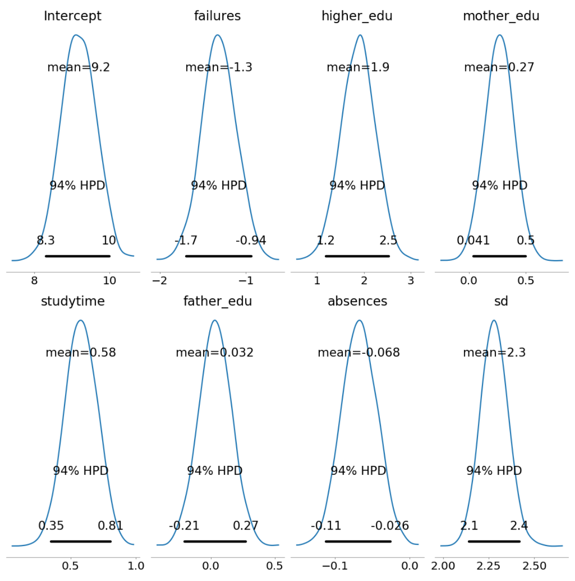
[2] https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d

[3] http://www.mit.edu/ ilkery/papers/GibbsSampling.pdf

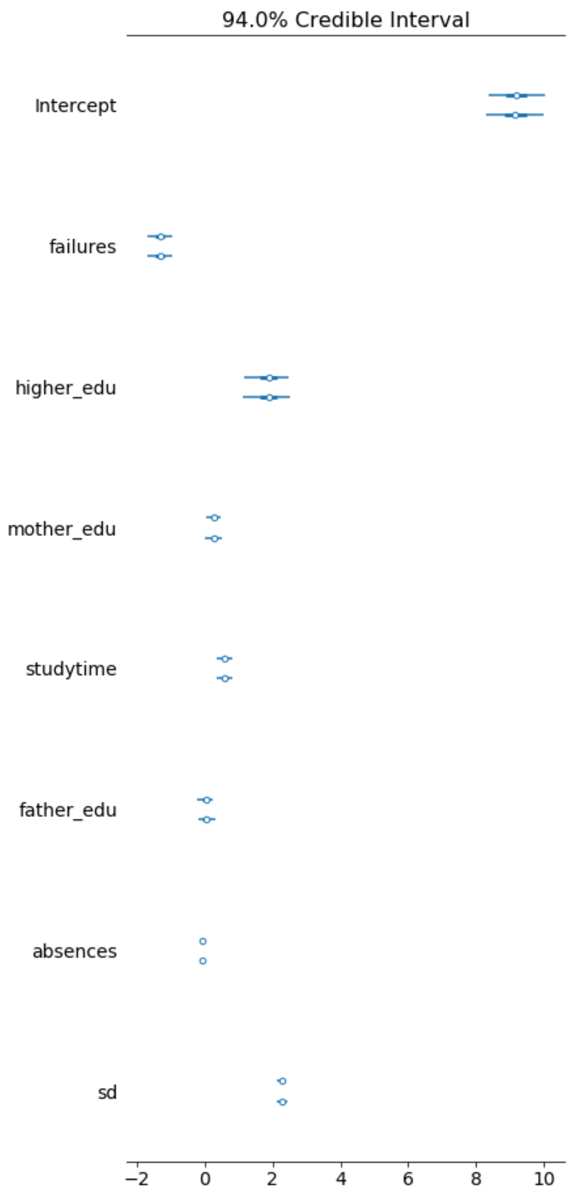
Appendix:

Posterior Plot





Forest Plot


94.0% Credible Interval
Intercept
failures
higher_edu
mother_edu
studytime
father_edu
absences
sd
−2
0
2
4
6
8
10