ISYE 6420 Bayesian Statistics
Final Project
Mark Pearl
December 6th, 2021

Bayesian Regression to Predict Expected Punt Yards Returned for NFL Special Teams

1. Introduction

NFL football is the most popular American sport, but has only recently begun to make use of granular data for analysis and modeling. In 2015, the NFL installed its Next Gen Stats system in every stadium, which records the position, velocity, and acceleration of every player and the ball at 10Hz throughout the game. The league recently created a Kaggle competition to examine 2018-2020 special teams plays and develop new metrics and insights. The next stage in the evolution of the game is utilizing this type of data to provide insights and make better decisions during a game. This project aims to examine special teams punt plays in order to model and predict the expected yards returned on a given punt play. This includes both fair catches (i.e. the returner took a knee on the field and didn't gain or lose any yards on the play), and negative (loss of yards) or positive returns.

Provided below is an example of what the setup / formation of a punt return play in the NFL:

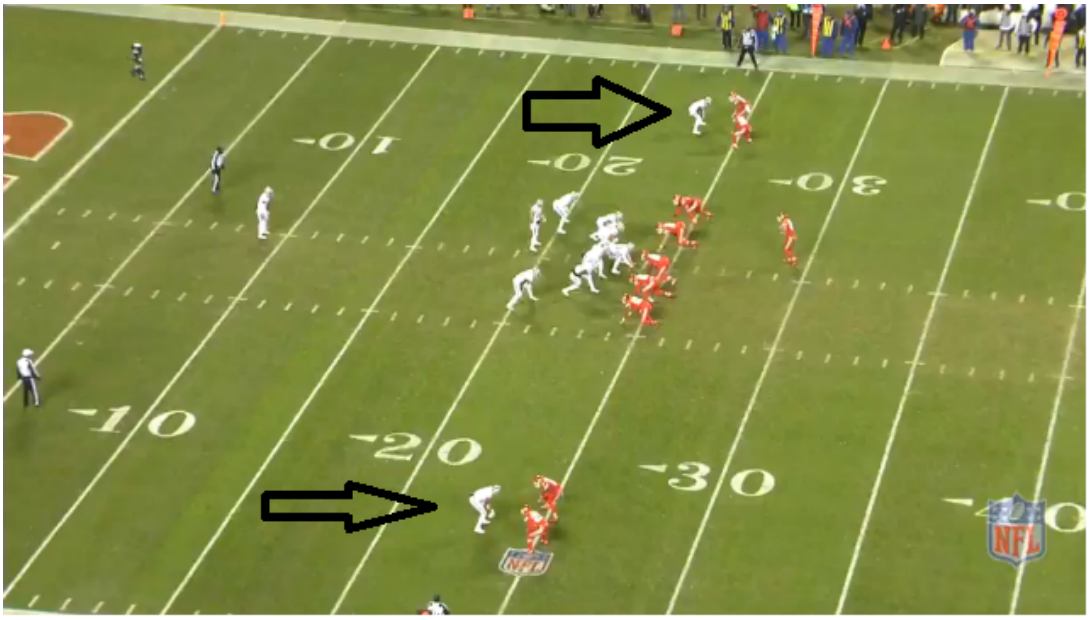


Figure 1: Example Punt Play

Please close attention to the arrows pointing to two players representing a key position on the defensive team, the gunners. These two plays pose the biggest threat to the returner being able to gain yards on the play, as they are the first 2 players the returner will come into contact with, and are typically running at high speeds.

quarter: Game quarter (numeric)
yardsToGo: Distance needed for a first down (numeric)
penaltyYards: yards gained by possessionTeam by penalty (numeric)

preSnapHomeScore: Home score prior to the play (numeric)
preSnapVisitorScore: Visiting team score prior to the play (numeric)
kickLength: Kick length in air of kickoff, field goal or punt (numeric)
s: Speed in yards/second (numeric)
a: Speed in yards/second^2 (numeric)
dis: Distance traveled from prior time point, in yards (numeric)
o: Player orientation (deg), 0 - 360 degrees (numeric)
dir: Angle of player motion (deg), 0 - 360 degrees (numeric)
hangTime: Hangtime of player's punt or kickoff attempt in seconds. Timing is taken from impact with foot to impact with the ground or a player. (numeric)
snapTime: Time from when the ball was snapped to when the kicker catches the ball
distance_returner_gunner1: Distance from returner to first gunner
distance_returner_gunner2: Distance from returner to second gunner
kickReturnYardage: Yards gained by return team if there was a return on a kickoff or punt (numeric)

We will approach this problem from the bayesian point of view, and create a bayesian regression in order to predict the target variable kickReturnYardage. This variable is not estimated as a single value, but is assumed to be drawn from a probability distribution.

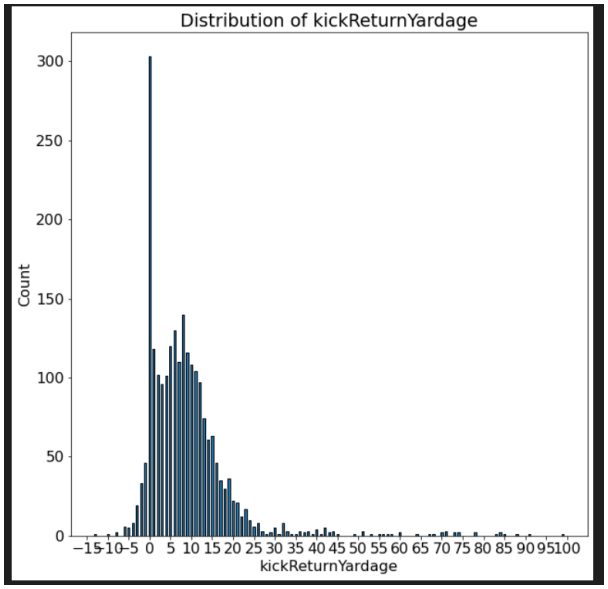


Figure 2: kickReturnYardage Histogram Plot

When taking a look at our response variable, we can see a significant number of values where the kickReturnYardage = 0. However, this is due to fair catches, and shouldn't be excluded from the model in our case because removing these observations may lead to inaccurate results.

The model for Bayesian Linear Regression with the response sampled from a normal

distribution is:

$$kickReturnYardage \sim N\left(\beta^T X,\ \sigma^2 I\right) \tag{1}$$

The goal of our model will be to determine posterior distribution for the model parameters, which will be derived from a distribution. This posterior is conditional upon the training inputs and ouputs:

$$P(\beta|\ kickReturnYardage,\ X) = \frac{P(kickReturnYardage|\ \beta, X) * P(\beta|X)}{P(y|X)} \tag{2}$$

Before implementing this derivation, we will further explore our data to determine the correlation between our response variable, and any other relationships we can derive.

2. Data Exploration

The first plot we will examine is looking at our correlation plots provided by the pandas_profiling package. This package generates a very in-depth report, going over variables distributions and summary statistics, correlations and interactions:

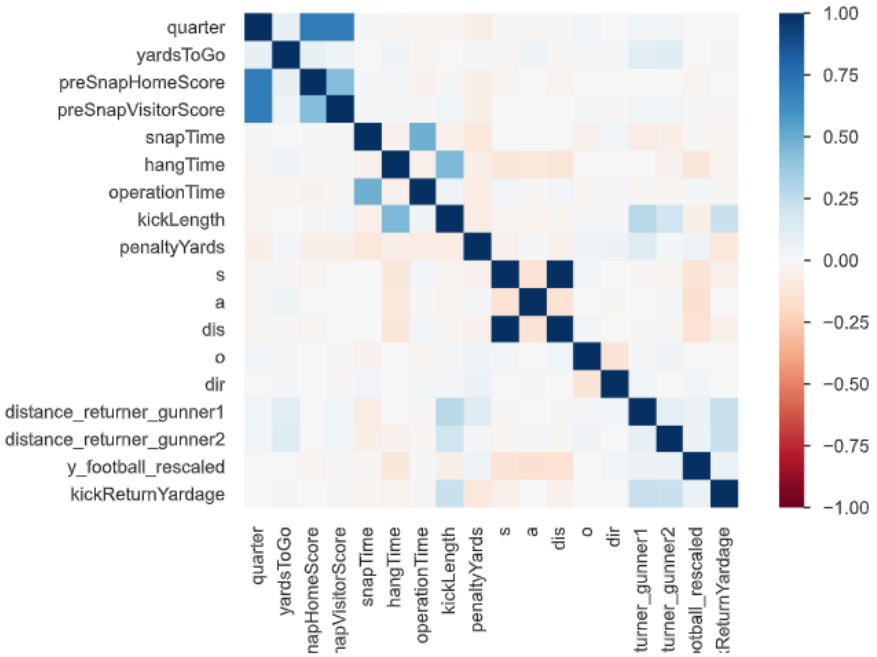


Figure 3: Spearman's Correlation Plot

From the results we can see that the kickLength and distance_returner_gunner features both correlate well with the target variable. The actual values for these correlations are as follows:

```
[('penaltyYards', -0.15063499894874638),
 ('s', -0.03224024279242861),
 ('dis', -0.03068830054110115),
 ('a', -0.02220479386384686),
 ('operationTime', -0.016331432853747475),
 ('preSnapHomeScore', -0.010146744766807445),
 ('preSnapVisitorScore', -0.0071798985522723655),
 ('hangTime', -0.0028989218521071414),
 ('quarter', -0.001059549352031851),
 ('o', 0.0009776022654363718),
 ('snapTime', 0.004766921512825578),
 ('dir', 0.006040256340115891),
 ('yardsToGo', 0.021358244641256987),
 ('y_football_rescaled', 0.050836477408908586),
 ('distance_returner_gunner2', 0.13800398916512716),
 ('distance_returner_gunner1', 0.17930198672417647),
 ('kickLength', 0.1851672036440199),
 ('kickReturnYardage', 1.0)]
```

We can also see from the above plot that multicollinearity is present between between certain features such as speed (s) and dis (distanced travelled), and we will account for these by filtering out features that don't have an absolute correlation value > 0.02.

This is a common feature selection process before conducting your training and test split. Once we derive these datasets, we can then assess how the distribution of each remaining variable is compared to the median of our response variable kickReturnYardage:

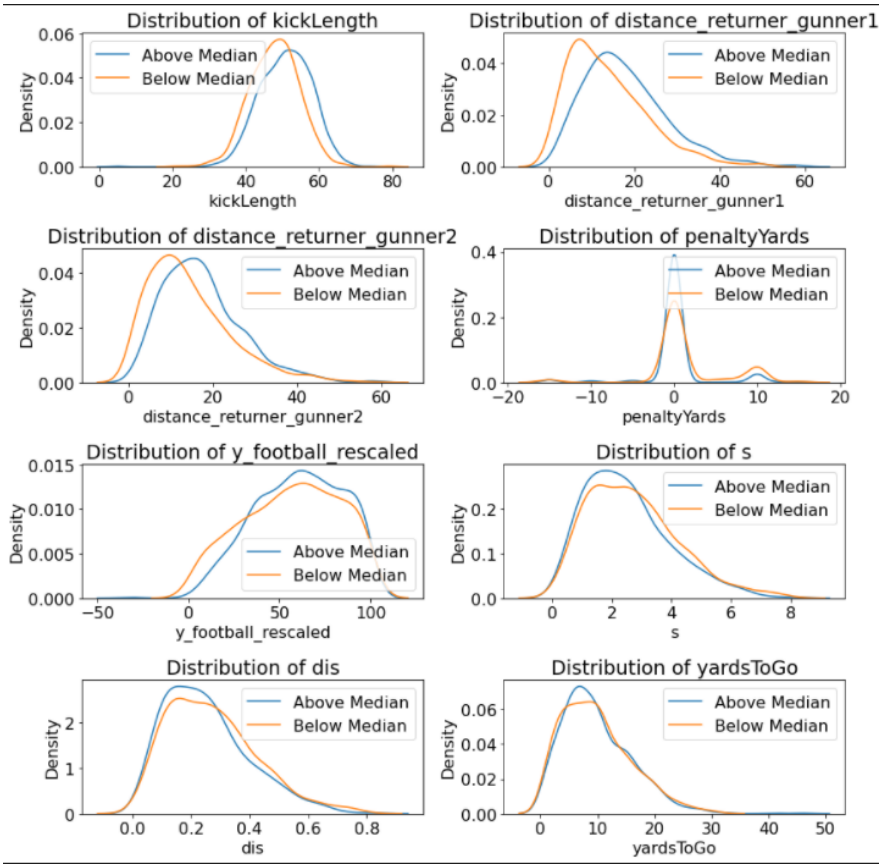


Figure 4: Distributions above / below median for each variable

We can see some very interesting observations from the above plots, such as the top 3 features: kickLength, distance_returner_gunner1 & 2. We see for these variables that the lower values of these features, have a negative impact on the kickReturnYardage, as the beginning values fall below the median. Whereas we can see a positive correlation when the values increase. This is a common situation found in football games and is more colloquially known as “out kicking your coverage”. In this situation, the punter kicked the ball far enough that the gunners could not get down the field fast enough. This allowed space between the gunners and the returner, and in turn the returner gained more return yards.

3. Model Identification & Results

As stated above, the problem formulation for our bayesian linear regression can be expressed as follows:

$$kickReturnYardage \sim N\left(\beta^T X,\ \sigma^2 I\right) \tag{3}$$

Before we conduct this modelling process, first we will train other machine learning models to get a baseline value for our validation criteria. In our case this will be MAE and RMSE.

These metrics have the following definitions:

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.

When the goal of problem is prediction, RMSE is the most widely adopted validation metric, where the model with the lowest MAE and RMSE is chosen.

In our case we chose a Linear Regression, RandomForest and GradientBoosted trees as part of the sklearn library. After running these models we got the following results:

Model Name	MAE	RMSE
Linear Regression	6.175323	10.451152
Random Forest	6.396804	10.504077
Gradient Boosted	6.108968	10.351924
baseline	6.551546	11.023795

The baseline is determined by using the median of our response variable, and using it in the MAE and RMSE calculations as a constant.

With this available we know have a good baseline to start our bayesian regression.

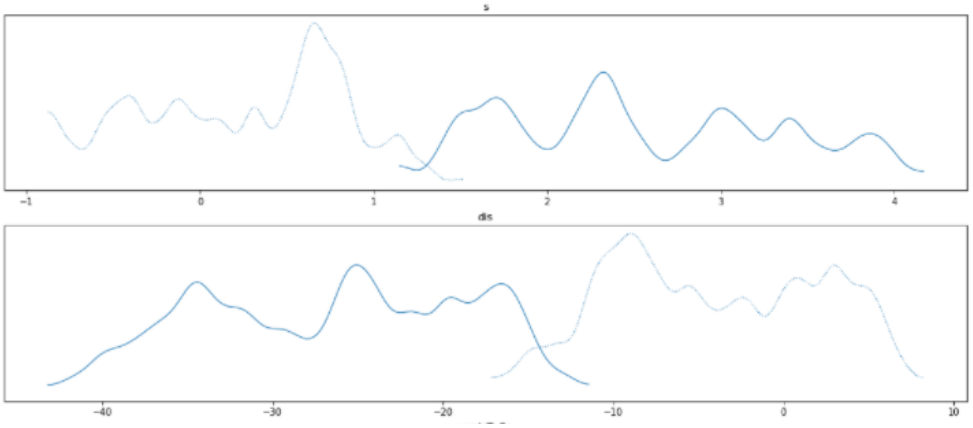
Since we're dealing with continuous values, we will use MCMC sampling method to approximate our posterior distribution for *kickReturnYardage.* We can then use the result of this to make infereces on new observations or test data. Since we don't have any domain knowledge to what our prior should be, a good non-informative prior choice would be the normal distribution.

We will use PyMC3 in order to construct our model with the code, using the normal distribution for the prior, and metropolis algorithm for the step function. Once metropolis was used for the step function, the algorithm was able to converge much quicker, whereas with default settings the model would hang.

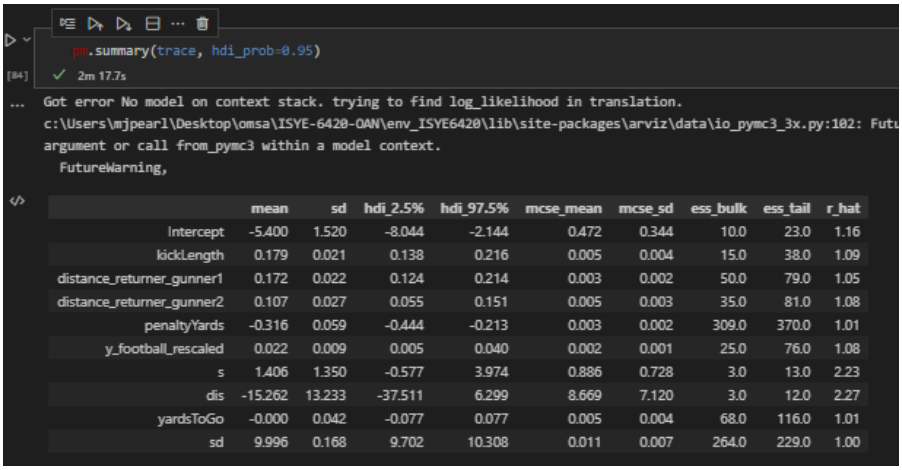
```
#Create formula used by GLM formulation for Bayesian Regression
Model
```

```
formula = 'kickReturnYardage ~ ' + ' + '.join(['%s' % variable for
 variable in X_train.columns[1:]])
formula

# Context for the model
with pm.Model() as normal_model:

    # The prior for the model parameters will be assumed to be
normal
    family = pm.glm.families.Normal()

    # Creating the model requires a formula and data (and
optionally a family)
    pm.GLM.from_formula(formula, data = X_train, family = family)

    # Perform mcmc sampling inference for the posterior
    # Have a choice of samplers
    step = pm.Metropolis()
    trace = pm.sample(draws=2000,step=step, chains = 2, tune =
500,cores=4)
```

After analyzing the trace variable, we can produce several useful plots to analyze. The first one is the traceplot, this shows the sampling conducted by MCMC for each of the two chains, and plots the posterior distribution and the samples drawn for each variable. We can see that our model did not converge for two variables s (speed) and dis:



This can be further supported by the summary plot as they have an r_hat value > 1.4 in our summary output:



```
pm.summary(trace, hdi_prob=0.95)
```

2m 17.7s

```
Got error No model on context stack. trying to find log_likelihood in translation.
c:\Users\mjpearl\Desktop\omsa\ISYE-6420-OAN\env_ISYE6420\lib\site-packages\arviz\data\io_pymc3_3x.py:102: Futu
argument or call from_pymc3 within a model context.
  FutureWarning,
```

	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
Intercept	-5.400	1.520	-8.044	-2.144	0.472	0.344	10.0	23.0	1.16
kickLength	0.179	0.021	0.138	0.216	0.005	0.004	15.0	38.0	1.09
distance_returner_gunner1	0.172	0.022	0.124	0.214	0.003	0.002	50.0	79.0	1.05
distance_returner_gunner2	0.107	0.027	0.055	0.151	0.005	0.003	35.0	81.0	1.08
penaltyYards	-0.316	0.059	-0.444	-0.213	0.003	0.002	309.0	370.0	1.01
y_football_rescaled	0.022	0.009	0.005	0.040	0.002	0.001	25.0	76.0	1.08
s	1.406	1.350	-0.577	3.974	0.886	0.728	3.0	13.0	2.23
dis	-15.262	13.233	-37.511	6.299	8.669	7.120	3.0	12.0	2.27
yardsToGo	-0.000	0.042	-0.077	0.077	0.005	0.004	68.0	116.0	1.01
sd	9.996	0.168	9.702	10.308	0.011	0.007	264.0	229.0	1.00

You can also see when looking at the hpd credible set, that the left tail corresponds to a negative value, and the right tail a positive value, so the model is not able to interpret if either of these variable have a positive or negative impact on the target variable:

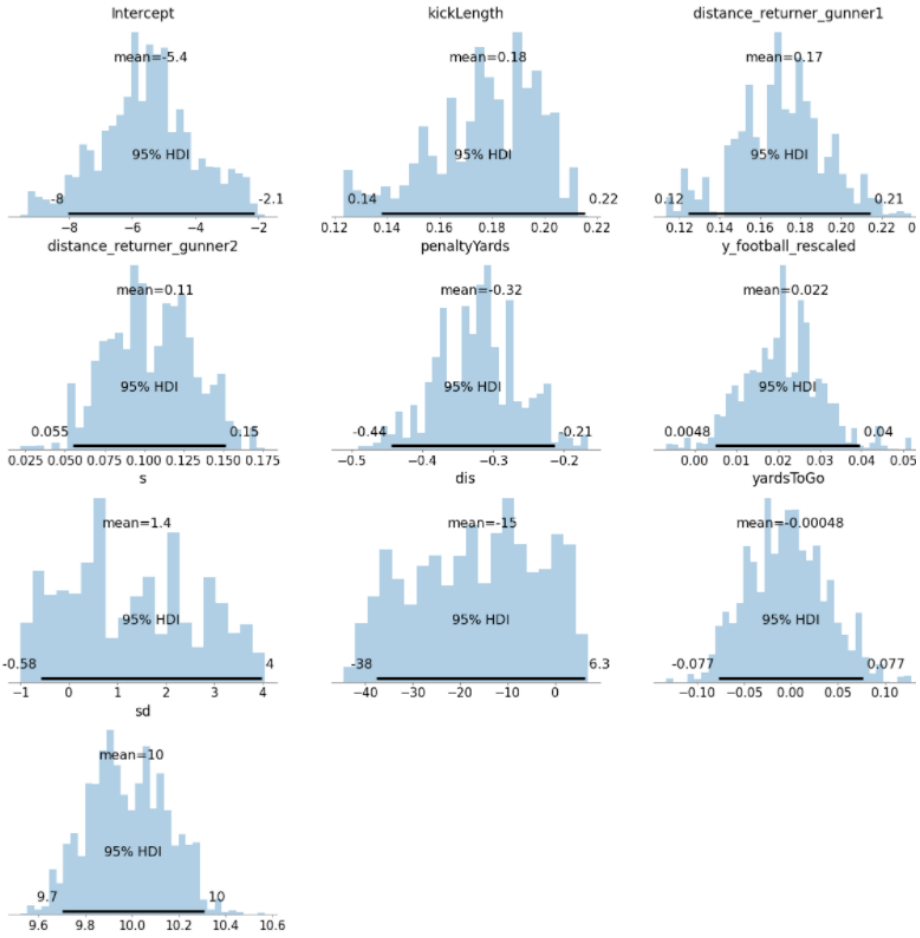
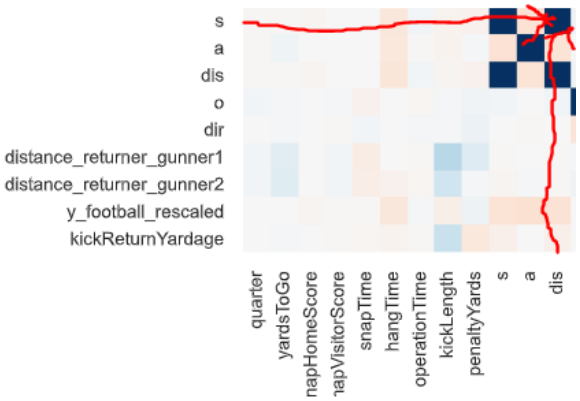


Figure 5: Posterior Plots / Variable with HPD Credible Set 95%

When reverting back to our correlation plot, we can see that this is likely due to these variables being perfectly correlated and is definitely causing the issue of convergence:



However, when we look at the modelling output, we can see that our bayesian model performs very well and wins the 2nd prize when compared against our other models, as the

Gradient Boosted tree still performed a bit better.

Model Name	MAE	RMSE
Linear Regression	6.175323	10.451152
Random Forest	6.396804	10.504077
Gradient Boosted	6.108968	10.351924
baseline	6.551546	11.023795
Bayesian LR	6.1504	10.444441

4. Conclusion / Discussion

Testing this experiment found interesting insights into NFL punt data. The initial hypothesis that distance between the gunners and the returner when the ball is caught is an important factor in predicting return success was validated in our Punt Factors Analysis model and the traceplots. This can provide very useful insights to special teams coaches, when doing analysis on a given play, and provide insights into potential formations or strategies to maximize return yardage on a given play.