1 Introduction/Problem Statement

As one of the most famous and useful statistical models, linear regression has been studied by researchers from every possible angle. Among the various perspectives that people look at the linear regression problem, two genres stand out and have been "competed" for decades, which are Bayesian and Frequentist. In this project, I aim to build a linear regression model to predict NBA players' salaries from both the Bayesian approach and the Frequentist approach. To be more specific, I want to study the difference between the two approaches in terms of the result and the interpretations of the parameters. In the next section, I am going to give a brief introduction to the data I collected for this project and the data preprocessing I have done.

2 Data Collection and Description

The data set I used in this project is named "NBA Player Salary Dataset (2017-2018)", and is downloaded the dataset from Kaggle(link is included in the Appendix section). The data includes information of 483 NBA players. For each player, twenty-eight unique features from various aspects are provided, including player's salary, age, nationality, draft number, and many game statistics, such as minutes played (MP) by the player, value over replaced player (VORP), etc. Figure 1 shows a preview of the data

	Player	Salary	NBA_Country	NBA_DraftNumber	Age	Tm	G	MP	PER	TS%	...	TOV%	USG%	OWS	DWS	WS	WS/48	OBPM	DBPM	BPM
0	Zhou Qi	815615	China	43	22	HOU	16	87	0.6	0.303	...	18.2	19.5	-0.4	0.1	-0.2	-0.121	-10.6	0.5	-10.1
1	Zaza Pachulia	3477600	Georgia	42	33	GSW	66	937	16.8	0.608	...	19.3	17.2	1.7	1.4	3.1	0.160	-0.6	1.3	0.8
2	Zach Randolph	12307692	USA	19	36	SAC	59	1508	17.3	0.529	...	12.5	27.6	0.3	1.1	1.4	0.046	-0.6	-1.3	-1.9
3	Zach LaVine	3202217	USA	13	22	CHI	24	656	14.6	0.499	...	9.7	29.5	-0.1	0.5	0.4	0.027	-0.7	-2.0	-2.6
4	Zach Collins	3057240	USA	10	20	POR	62	979	8.2	0.487	...	15.6	15.5	-0.4	1.2	0.8	0.038	-3.7	0.9	-2.9

Figure 1: Preview of the data

Among the data entries, four features were detected with missing values, which are turnover percentage(**TOV%**), free-throw rate(**FTr**), three-point attempt rate(**3PAr**) and true shooting percentage(**TS%**). Since only two missing entries were found in the four categories, the missing values were filled with the median of each column.

The original response variable, which is **Salary**, is strongly right skewed. It indicates that most NBA players in the league are earning much less money than a few all-star players that are earning a significant amount. In order to meet the linear regression assumption, the original response was transformed to fit a normal distribution with the Python library PowerTransformer. Figure 2 shows the distribution of the dependent variable before and after the transform.

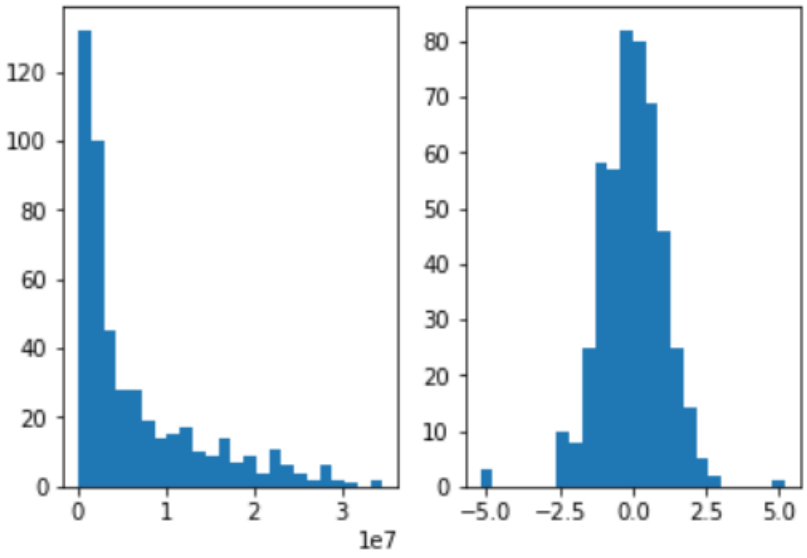

Figure 2: The left figure shows the response distribution before transform, and the right figure shows the response distribution after transform

Furthermore, I performed feature selection on the twenty-eight features as I want to avoid unimportant features that affect the predictive power of the model. After converting the two categorical features, player's nationality(**NBA_Country**) and the team(**Tm**) he plays for into one-hot vector, the correlation coefficient of each feature is calculated, and the top 9 features with the highest score were selected, which are shown in the table below.

Feature Name	Correlation Coefficient
Minutes Played (MP)	0.567254
Win Shares (WS)	0.556820
Deffensive Win Shares (DWS)	0.520295
Draft Number	-0.504131
Offensive Win Shares (OWS)	0.504056
VORP	0.489313
Games played (G)	0.452567
Box Plus/Minus (BPM)	0.344700
Age	0.341460

Last but not least, it is necessary to check multicollinearity among the features before I start fitting the model. VIF scores are calculated for each feature. It is easy to see that there exists multicollinearity among **WS**, **OWS**,**DWS**, **MP** and **G**. After removing **WS**, **OWS**, **DWS** and **G**, the multicollinearity problem seems vanished, and we are ready to fit the model. The following table shows the VIF score of each feature before and after we removed the correlated features.

Feature Name	VIF Score Before	VIF Score After
Minutes Played (MP)	22.5	5
Win Shares (WS)	4790.7	Removed
Deffensive Win Shares (DWS)	893.8	Removed
Draft Number	3.3	3.2
Offensive Win Shares (OWS)	1872.6	Removed
VORP	10.0	2.2
Games played (G)	26.7	Removed
Box Plus/Minus (BPM)	1.6	1.5
Age	9.4	7.2

3 Bayesian Analysis vs. Frequentist Approach

In this section, I am going to describe how the Bayesian and Frequentist models were set up. The data was split up into training set(80%) and validation set(20%). As data preprocessing and feature selection have been performed in the previous sections, linear regression model was applied to the Frequentist approach on the training set and will be evaluated on the validation set. The evaluation metrics used in this project are Mean Absolute Error(MAE) and Root Mean Square Error(RMSE). For Bayesian approach, the problem is set up as follow:

$$\begin{aligned}
&\mathbf{y} \sim N(\beta^T \mathbf{X} | \sigma^2 I) \\
&\beta \propto 1 \\
&\epsilon \sim^{iid.} N(0, \sigma^2)
\end{aligned}$$

The likelihood can be expressed as

$$\begin{aligned}
p(\mathbf{y}|\mathbf{X}, \beta, \sigma^2) &\propto \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\mathbf{y}-\beta^T\mathbf{X})^2}{2\sigma^2}} \\
&= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{s_i^2}{2\sigma^2}}
\end{aligned}$$

where $s_i = y_i - (\beta_0 + \beta_1 * x_{i1} + \beta_2 * x_{i2} + \beta_3 * x_{i3} + \beta_4 * x_{i4} + \beta_5 * x_{i5})$. To sample the parameters, Python library PyMC3 was applied. In the experiment, we ran two trials with different number of samples to draw, which are 1,000 and 2,000. The starting points were computed with function **find_MAP** which applys the Broyden–Fletcher–Goldfarb–Shanno (BFGS) optimization algorithm to find the maximum of the log-posterior. The number of chains was set to 2. In terms of sampling

method, Metropolis-Hastings and No U-Turns Sampler(NUTS) were applied to draw the posterior samples. One reason to try two sampling methods is that I would like to see the performance of both methods as Metropolis method was taught in class but NUTS method is recommended by the library developer. The codes to build the models are shown below, and the results of three models will be shown in the next section

```
# Fit the Frequentist model
def evaluate_model(model, x_test, y_test):
    predictions = model.predict(x_test)
    mae = np.mean(abs(predictions - y_test))
    rmse = np.sqrt(np.mean((predictions - y_test) ** 2))
    return (mae, rmse)

final_df = df3[['Response','VORP', 'MP','NBA_DraftNumber', 'Age', 'BPM']]
from sklearn.model_selection import train_test_split
X = final_df[['VORP', 'MP','NBA_DraftNumber', 'Age', 'BPM']]
y = final_df[['Response']]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
from sklearn.linear_model import LinearRegression
frequentist = LinearRegression()
frequentist.fit(X_train, y_train)

#Nuts draw 2000 samples
with pm.Model() as normal_model:
    family = pm.glm.families.Normal()
    pm.GLM.from_formula(formula, data = bayesian_data, family = family)
    start = pm.find_MAP()
    step = pm.NUTS()
    normal_trace = pm.sample(draws=2000, step=step,
                             start=start,chains = 2, tune = 1000, progressbar=True)
# Nuts draw 1000 samples
with pm.Model() as normal_model:
    family = pm.glm.families.Normal()
    pm.GLM.from_formula(formula, data = bayesian_data, family = family)
    start = pm.find_MAP()
    step = pm.NUTS()
    normal_trace = pm.sample(draws=1000, step=step,
                             start=start,chains = 2, tune = 1000, progressbar=True)
#Metropolis
with pm.Model() as normal_model:
    family = pm.glm.families.Normal()
    pm.GLM.from_formula(formula, data = bayesian_data, family = family)
    step = pm.Metropolis()
    normal_trace = pm.sample(draws=2000, step=step, chains = 2, tune = 2000,
                             progressbar=True)
```

4 Result and Discussion

In this section, I am going to compare the results of the four models, and discuss the interpretation of the results and further improvements. The parameters' posterior distributions sampled from Metropolis are less smooth than those sampled from NUTS, and the effective size is also lower. Due to the page limit, I will illustrate the trace plot and the posterior distributions of parameters of the model with 1000 samples draw in Figure 3 and 4. The other trace plots and distribution plots along with summary statistics are included in the Appendix section.

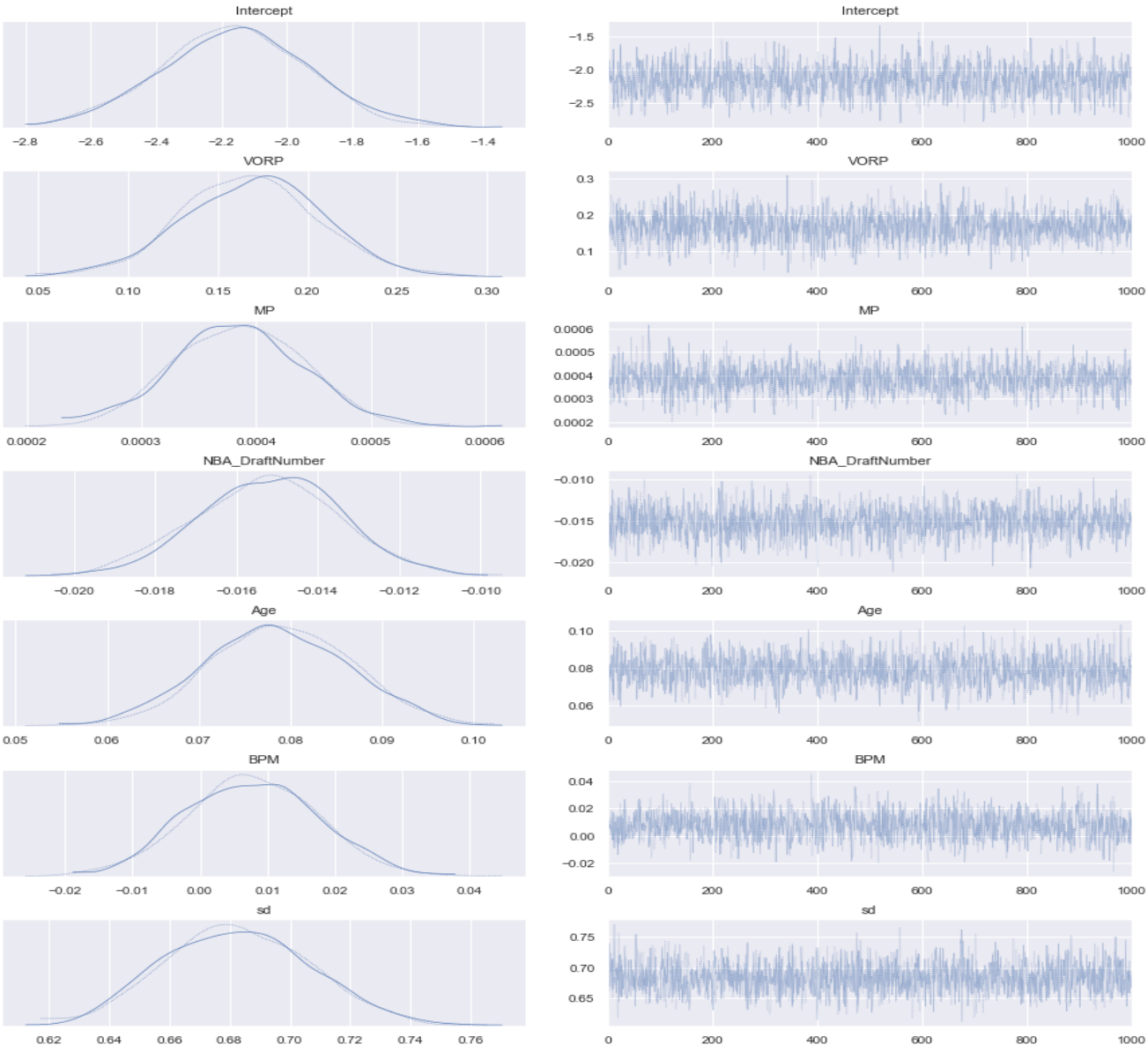

Figure 3: Trace plots of 1000 sampling

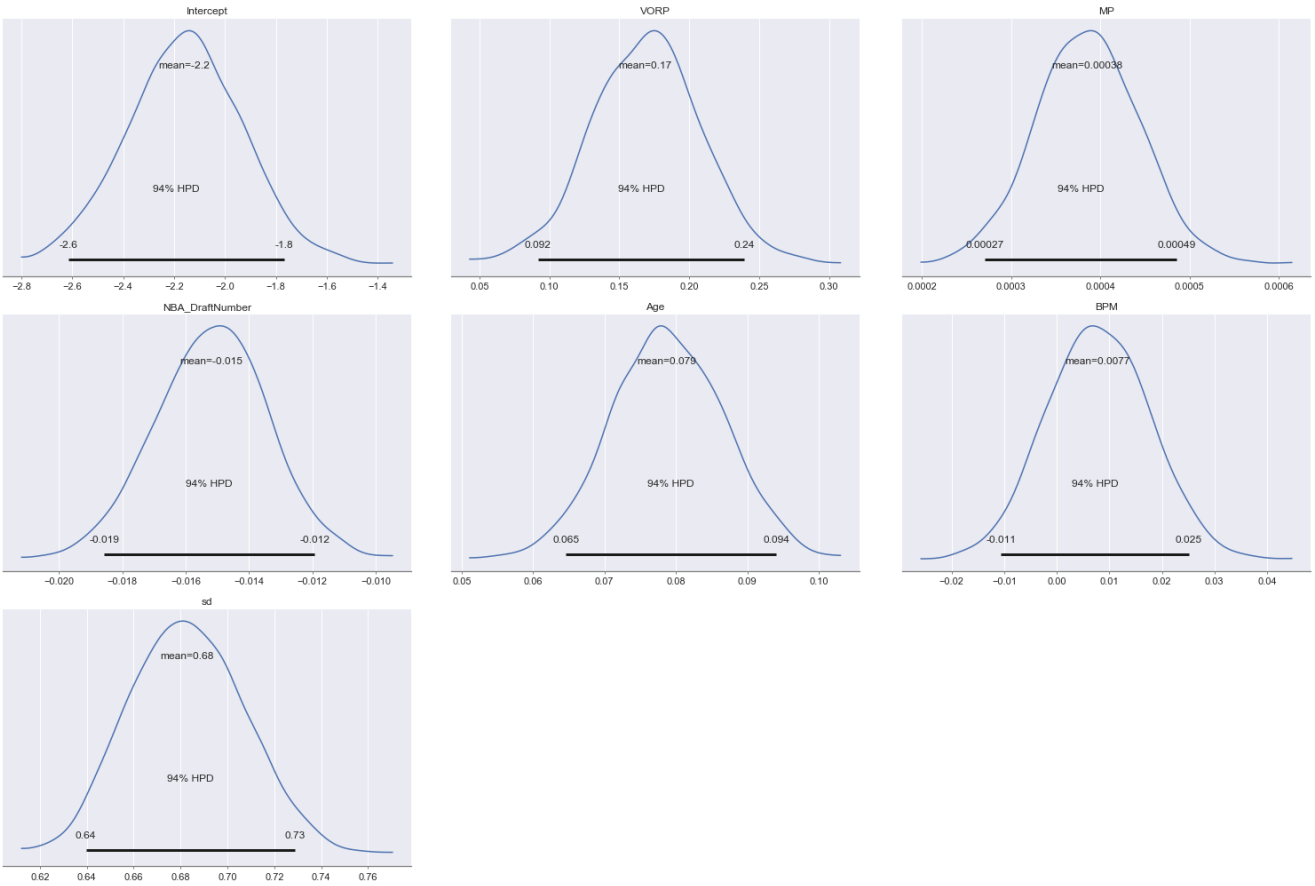

Figure 4: Posterior distributions of the parameters

The equations of the three models are shown as follow:

$$\begin{aligned}
\text{Metropolis}_{2000} : \text{Salary} = &-2.1066 + 0.1685 * \text{VORP} + 0.0004 * \text{MP} - 0.0153 * \text{DraftNumber} \\
&+ 0.0766 * \text{Age} + 0.0074 * \text{BPM} \\
\text{NUTS}_{1000} : \text{Salary} = &-2.1598 + 0.1687 * \text{VORP} + 0.0004 * \text{MP} - 0.0152 * \text{DraftNumber} \\
&+ 0.0787 * \text{Age} + 0.0077 * \text{BPM} \\
\text{NUTS}_{2000} : \text{Salary} = &-2.1598 + 0.1694 * \text{VORP} + 0.0004 * \text{MP} - 0.0152 * \text{DraftNumber} \\
&+ 0.0786 * \text{Age} + 0.0077 * \text{BPM} \\
\text{Frequentist} : \text{Salary} = &-2.1587 + 0.1703 * \text{VORP} + 0.0004 * \text{MP} - 0.0152 * \text{DraftNumber} \\
&+ 0.0786 * \text{Age} + 0.0075 * \text{BPM}
\end{aligned}$$

And the evaluation scores are show in Figure 5: If we test the three models with a virtual player

	Mean Abs. Error	Root Mean Square Error
Frequentist	0.611399	0.930005
NUTS_1000	1.134446	1.494323
NUTS_2000	1.134190	1.494059
Metro_2000	1.133371	1.493248

Figure 5: Evaluation scores of three models

who's statistics are all at 75 percentile of the population, the predicted salaries of the player from the three models are

	NUTS_1000	NUTS_2000	Metropolis_2000	Frequentist
Salary	$11984473	$11984473	$11939148	$11964463

Discussion: By looking at the evaluation scores of the three models, we can observe that the MAE and RMSE of the two Bayesian models are both slightly higher than those of the Frequentist model. However, the predicted salaries of the four model are just slightly different. Furthermore, the estimated parameters are also very similar. However, the interpretation of the parameters are very different between Bayesian and Frequentist, where Bayesian believes that the estimated parameter follow a certain distribution where Frequentist treats the parameter as a point estimate. Although the Bayesian models show higher errors, the error can be easily reduced by setting a weekly informative prior or more precise prior information. Furthermore, with Bayesian approach, the statistical inferences of the parameters can be interpreted clearly. This course is the first course I have ever taken on Bayesian statistics and I really appreciate the professor gives me the chance to beware the beauty and power of Bayesian. I am more than willing to keep learning knowledge of Bayesian statistics and seek chances to apply the skills I have learned in my future career.

5 Appendix

data source = https://www.kaggle.com/aishjun/nba-salaries-prediction-in-20172018-season
The Jupyter Notebook I worked on is appended to this section