Homework 2 report

Luka Utješinović, 63210495

I. Problem 1

Multinomial Logistic regression (Softmax regression) - Given that we have a categorical target variable y with m outcomes, in this model we assume that $y_i|X_i,\beta \sim Categorical(Softmax(u_1,\dots,u_{m-1},0))$, where:

$$u_k = \beta_k^T x_i + \beta_{k0}, \forall k \in \{1,\dots m-1\}$$

$$Softmax(u_1,\dots,u_m) = [\frac{e^{u_1}}{\sum_{k=1}^{m-1} e^{u_k}+1}, \dots \frac{e^{u_{m-1}}}{\sum_{k=1}^{m-1} e^{u_k}+1}, \frac{1}{\sum_{k=1}^{m-1} e^{u_k}+1}]$$

(we set the m-th category as a refference). Our model has $(m-1)\times(N+1)$ parameters, where N is the dimensionality of data ($+1$ comes from the bias). If we set $\beta_{mk} = 0, \forall k \in \{1,\dots N\}$ (i.e. m-th category is set as a reference) we can write the negative log-likelihood of the model as:

$$l(\beta|X,Y) = -\sum_{i=1}^{M}\sum_{j=1}^{m} ln(\frac{e^{\beta_j^T x_i+\beta_{j0}}}{\sum_{k=1}^{m} e^{\beta_k^T x_i+\beta_{k0}}})I(y_i = j)$$

where $I(y_i = j)$ is the indicator of y_i being equal to j, and M is the number of observations. For this homework, we used the **L-BFGS** algorithm for optimization. By taking a simple 80-20 train-test split of the dataset, our model achieved accuracy of $\approx$ **75 %**.

Ordinal Logistic regression - Now our target variable y is ordinal and has m levels. The assumption of this model is that $y_i|X_i,\beta \sim Categorical(p_1,\dots,p_m)$, where $p_i = F(t_i - (\beta^T x_i + \beta_0)) - F(t_{i-1} - (\beta^T x_i + \beta_0))$, where F is the CDF of zero mean and unit variance logistic distribution, and $t_1 = 0 < t_1 < \dots < t_{m-1}$ are the thresholds of the model. For convenience, $t_0 = -\infty, t_m = +\infty$. The negative log-likelihood of this model can be written as:

$$l(\beta|X,Y) = -\sum_{i=1}^{M}\sum_{j=1}^{m} ln(F(t_j(\beta^T x_i + \beta_0)) - F(t_{j-1} - (\beta^T x_i + \beta_0)))I(y_i = j)$$

We used the so called stick-breaking parametrizations of the thresholds for our model, which makes optimization much more convenient. Namely $t_1 = 0, t_k = \sum_{i=1}^{k-1}\Delta_i, \forall k \in \{2,\dots,m-1\}$ with constraint $\Delta_i > 0 \forall i \in \{1,\dots,m-1\}$. Using this parametrization, we reduce the number of parameters of our model to $N+m-1$. Target variable of the dataset we were given is not ordinal, and it does not make much to model our data like this, so we did not test this model.

II. Problem 2

First let us talk about preprocessing. Following features are categorical: **Movement**, **Competition**, **PlayerType**, **Transition**, **TwoLegged**. To feed these features to our model we used one-hot encoding. To avoid multicollinearity, we generate $m-1$ one-hot encoded vectors for a categorical variable with m different values (features **Transition** and **TwoLegged** are binary i.e. these features are already one-hot encoded). By performing one-hot encoding, we effectively set one value as a reference for each of our categorical features. So, for **Movement** reference is **dribble or cut**, for **Competition** reference is **EURO** and for **Player type** reference is **C**. Since regression models are sensitive to the scale of input features, we standardized the dataset. However, standardizing one-hot encoded vectors does not make sense, so we only performed standardization for numerical features, which were **Angle** and **Distance**. Just as a reference, standardization for column c is performed as $\hat{X}_c = \frac{X_c - \overline{X}_c}{\sigma X_c}$, where $\overline{X}_c$ is the sample mean and σX_c is the sample standard deviation. Lastly, our target category had 6 values and, the values were encoded to numbers by the following map: $\{$**above head**, **layup**, **other**, **hook shot**, **dunk**, **tip-in**$\}$ → [0, 1, 2, 3, 4, 5]. For the MLR model, tip-in category was set as a reference.

Now let us discuss model interpretation. We took 1000 bootstrap samples from the original dataset, and trained 1000 different MLR models. Each model had $5 \times (12+1) = 65$ parameters (after preprocessing, for each non-reference category we have 12 features, and in addition we have a bias for each category). We saved values obtained for each parameter, and then we generated 95% confidence intervals for every parameter (using empirical quantiles at 0.025 and 0.975). Before we show the results, we will address one more technical detail. Unconstrainted L-BFGS algorithm took $\approx$ 100 seconds to train a single MLR model (even after fully optimizing the negative log-likelihood implementation using **NumPy**, refer to the code), which would mean that to train $\approx$ 1000 models we would need $\approx$ 27.8 hours, which was unacceptable. Instead, we limited L-BFGS to 20 iterations only. This is value was empirically chosen because we noticed that after $\approx$ 20 iterations, the negative log-likelihood decreases only slightly, and the training got reduced to $\approx$ 12 seconds per model. For a few pairs of bootstrapped samples and corresponding models, we performed a simple 80-20 accuraccy test, and the average accuracy was $\approx$ 72%, which suggests that we did not loose much predictive power while having a dramatic increase in runtime, which was a compromise we were willing to take.

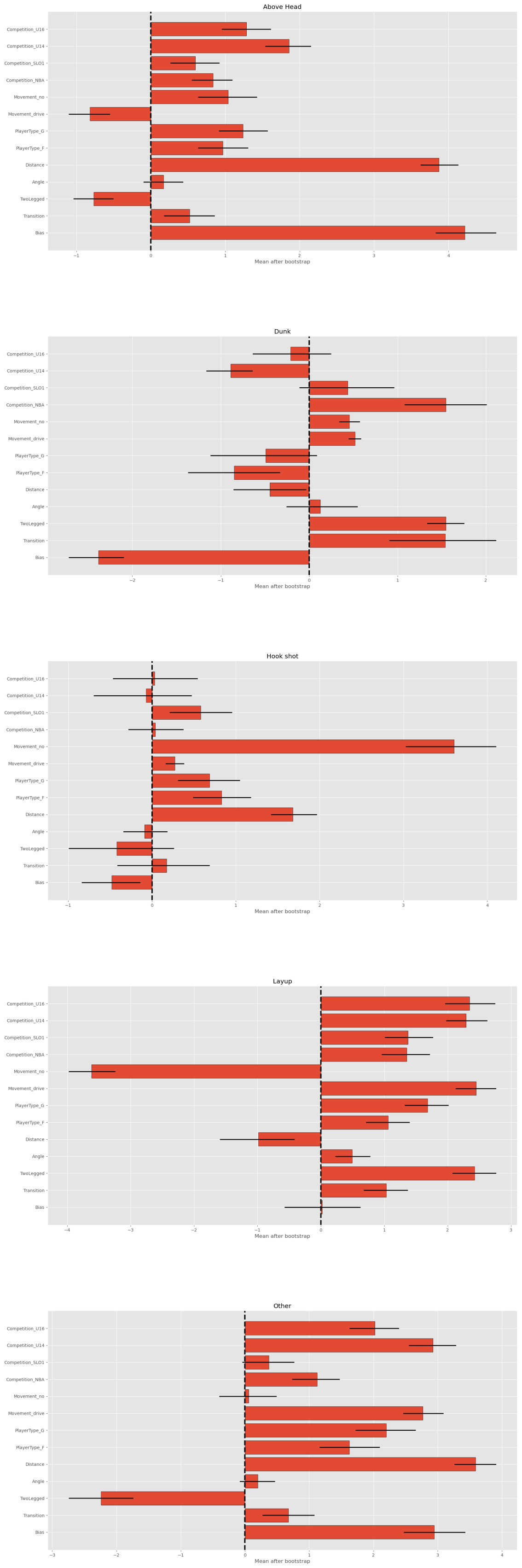

Fig. 1. Results obtained after the bootstrap. Orange bar represents the mean value of a parameter after bootstrapping, and the horizontal black line represents a 95 % bootstrap confidence interval for that parameter. Bias stands for the intercept.

We can see that for every category we have coefficients with high absolute value, with varying uncertainty. Starting with the above head category, we can see that higher distance from the hoop with little uncertainty increases the probability of a shot being above head, which is natural. For dunk, we can see that there is a lot of uncertainty for distance and angle, however it seems that dunks are lees likely to occur in junior league U 14. Assuming U 14 stands for under 14, this is also reasonable since we would expect that a player of that age cannot yet dunk. With some uncertainty, no movement increases the chances of hook shots, while the opposite holds for a layup shot with less uncertainty.

III. Problem 3

As we stated before, OLR model has much fewer parameters than the MLR model, so we would expect that training a more complex model on a smaller dataset cannot produce good results, since we have many parameters that we have to fit. During the creation of our dataset, we should also incorporate the ordering assumption of OLR to get better results than MLR. Having ordinal values that are not equdistant should favor the OLR model. Following this intuition, we constructed the following dataset: 1100 rows in total, out of which 100 are used for training the models and 1000 for testing the models. There are 4 categories, and for each category we have 275 observations. To generate each category we used a normal distributions with unit variance and different means, while making sure that the means are not equidistant. Namely:

- Category 0 $\sim \mathcal{N}(0,1)$
- Category 1 $\sim \mathcal{N}(1,1)$
- Category 2 $\sim \mathcal{N}(5,2)$
- Category 3 $\sim \mathcal{N}(12,1)$

Results are shown on the figure below:

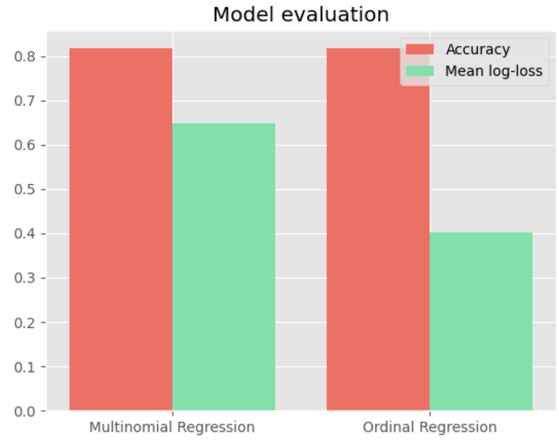

Fig. 2. Even though the acurracies of the models are pretty much the same, mean log-loss of the OLR model is significantly smaller, which means that the ORL model is on average much more confident in it's predictions than the MLR model on this dataset.