Assignment-4 DLS Method

Submitted by: Tarun Kumar Sahu
SR No.: 23156

1. Introduction

In this report, I implemented a model to estimate run production functions in cricket matches using Python. The model is based on optimizing a loss function, which uses a Poisson log-likelihood approach to calculate the difference between actual and predicted runs. The main functions in the Python code are explained below.

2. Function Explanations

2.1 `load_and_preprocess_data(file_path)`

This function loads the cricket data from a CSV file that contains data on ODI matches from 1999 to 2011, preprocesses it by calculating the remaining runs and overs, and filters the data to include only the first innings. The function returns a data frame containing the following columns:

- `remaining_overs` - The number of overs remaining.
- `Wickets.in.Hand` - The number of wickets in hand.
- `remaining_runs` - The number of runs remaining to reach the total.

2.2 `run_production_function(Z0, L, u)`

This function implements the Duckworth-Lewis-Stern method for estimating expected runs. It calculates the expected runs based on the following formula:

$$Z(u, w) = Z_0(w) \left(1 - e^{-\frac{L(w)u}{Z_0(w)}}\right)$$

Where:

- $Z_0(w)$ is the asymptotic value of the run production function.
- $L(w)$ is the rate parameter.
- u is the number of overs remaining.

2.3 `safe_log(x)`

This utility function safely computes the logarithm, ensuring that there are no errors due to taking the log of zero by applying a small threshold. It is used within the loss function.

2.4 `loss_function(y_true, y_pred)`

This function computes the Poisson log-likelihood loss between the true (`y_true`) and predicted (`y_pred`) values. The formula used is:

$$\text{loss}(y', y) = (y' + 1) \log\left(\frac{y' + 1}{y + 1}\right) - y' + y$$

Where:

- y' is the predicted value.
- y is the true value.

The function sums this loss across all data points and returns the total.

2.5 `objective_function(params, u, w, y)`

This function defines the objective function for optimization. It takes the model parameters Z_0 and L, the overs remaining (`u`), the wickets in hand (`w`), and the true runs (`y`). It computes the predicted runs using the `run_production_function` and returns the loss.

2.6 `fit_preliminary_model(data)`

This function fits the preliminary model for each wicket (from 1 to 10 wickets in hand). It uses the optimization method `L-BFGS-B` to minimize the loss function, estimating Z_0 and L for each wicket.

2.7 `calculate_common_L(preliminary_params, data)`

This function calculates a common value for L as the weighted average of the preliminary $L(w)$ values. The weights are the number of data points available for each wicket.

2.8 `fit_final_model(data, common_L)`

This function fits the final model using the common L value obtained from the preliminary model. It estimates $Z_0(w)$ for each wicket while keeping the common L fixed.

2.9 `plot_run_production_functions(params, common_L, title)`

This function plots the run production functions for the different wickets in hand using the estimated Z_0 and L values. It generates a plot showing expected runs versus overs completed.

2.10 `calculate_normalized_loss(params, data, common_L)`

This function calculates the normalized loss across all data points using the model parameters and the standard L. It returns the average loss per data point.

3. Preliminary and Final Parameters

3.1 Preliminary Parameters (20 in total: Z0 and L for each wicket)

The preliminary model estimated the following values for Z_0 and L for each wicket:

- Wicket 1: $Z_0 = 6.50$, $L = 6.17$

- Wicket 2: $Z_0 = 19.71$, $L = 7.70$
- Wicket 3: $Z_0 = 40.38$, $L = 10.21$
- Wicket 4: $Z_0 = 67.15$, $L = 10.16$
- Wicket 5: $Z_0 = 92.31$, $L = 10.41$
- Wicket 6: $Z_0 = 122.77$, $L = 11.15$
- Wicket 7: $Z_0 = 156.31$, $L = 10.81$
- Wicket 8: $Z_0 = 194.27$, $L = 10.83$
- Wicket 9: $Z_0 = 219.05$, $L = 11.45$
- Wicket 10: $Z_0 = 274.19$, $L = 10.62$

3.2 Final Parameters (11 in total: 10 Z_0 values and 1 common L)

The final model estimated the following values for Z_0 for each wicket, using a common L value:

- Wicket 1: $Z_0 = 6.22$
- Wicket 2: $Z_0 = 18.72$
- Wicket 3: $Z_0 = 40.02$
- Wicket 4: $Z_0 = 66.10$
- Wicket 5: $Z_0 = 91.51$
- Wicket 6: $Z_0 = 125.50$
- Wicket 7: $Z_0 = 157.61$
- Wicket 8: $Z_0 = 196.21$
- Wicket 9: $Z_0 = 227.60$
- Wicket 10: $Z_0 = 273.85$
- Common $L = 10.64$

4. Poisson Log-Likelihood Loss Function

The Poisson log-likelihood loss function measures the discrepancy between the predicted and actual values. In this model, it helps capture the difference between predicted runs and actual runs. The formula:

$$\text{loss}(y', y) = (y' + 1) \log\left(\frac{y' + 1}{y + 1}\right) - y' + y$$

It is based on the logarithmic transformation of the ratio between predicted and actual values, making it sensitive to value differences. The term $\log\left(\frac{y'+1}{y+1}\right)$ ensures that small errors between predicted and actual values contribute less to the loss, while more significant discrepancies contribute more.

5. Run Production Function Plots

Two plots were generated to visualize the run production functions for the preliminary and final models.

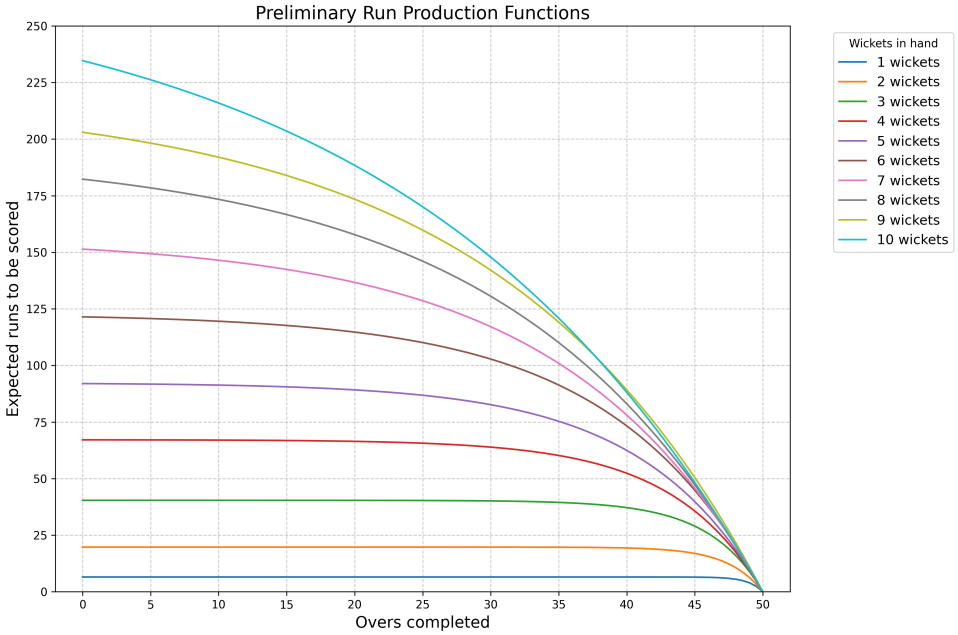

Figure 1: Preliminary Run Production Functions

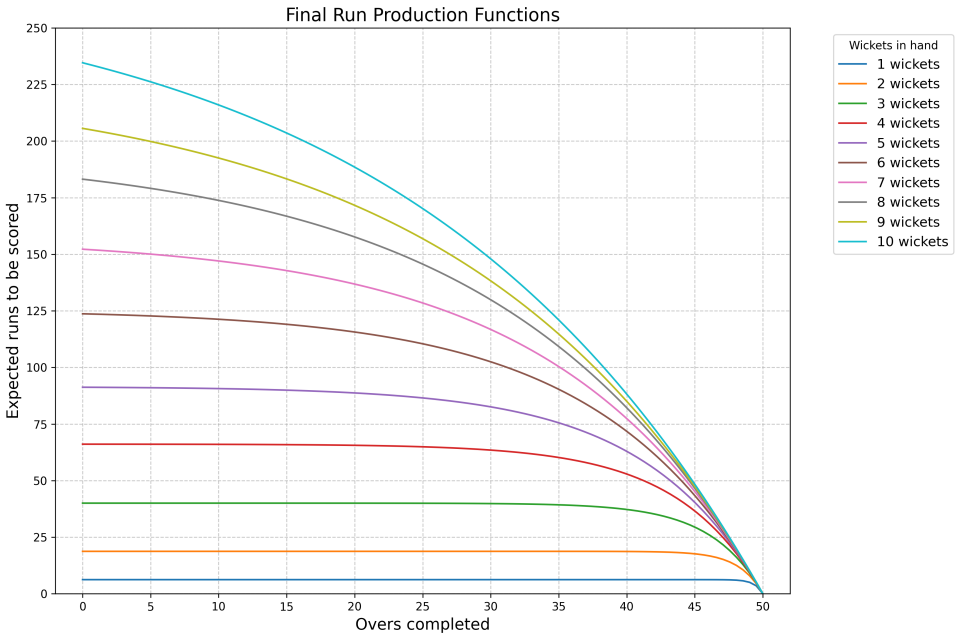

Figure 2: Final Run Production Functions

6. Conclusion

This report presented the implementation of a model to estimate run production functions in cricket using Python. The model is based on minimizing a Poisson log-likelihood loss function and uses optimization to estimate parameters for each wicket. The final model accurately estimates expected runs given the overs remaining and wickets in hand.