Great — here is a **clear, structured, submission-ready explanation** of the FICO quantization code you implemented. This is written the way you would explain it in a **project report, case study, or viva**.

📘 Mortgage Credit Rating Construction using Optimal FICO Quantization

(Log-Likelihood & Dynamic Programming Approach)

---

1. Objective

The objective of this task is to transform continuous **FICO credit scores** into a fixed number of **categorical credit ratings** that best summarize the default behavior of borrowers. These ratings are required because the downstream machine learning architecture accepts **categorical inputs only**.

To achieve this, we design a **general, data-driven quantization method** that automatically determines the optimal FICO bucket boundaries such that borrowers within each bucket exhibit similar default risk.

---

2. Conceptual Approach

Each borrower either defaults or does not default. Therefore, defaults follow a **Bernoulli distribution**.

If we group borrowers into buckets, each bucket has:

- (n_i): total borrowers
- (k_i): defaults
- (p_i = k_i / n_i): probability of default

The quality of a bucket is measured using the **Bernoulli log-likelihood**:

[
L_i = k_i \log(p_i) + (n_i - k_i)\log(1 - p_i)
]

The goal is to find bucket boundaries that **maximize the total log-likelihood across all buckets**. This ensures that borrowers grouped together have statistically similar default behavior.

Because the problem requires optimal ordered splitting, we solve it using **dynamic programming**, which guarantees a **globally optimal** set of bucket boundaries.

---

3. Step-by-Step Explanation of the Code

Step 1: Data Preparation and Sorting

```
df = pd.read_csv(...)
df = df.sort_values("fico_score")
```

Borrowers are sorted by FICO score to preserve the natural risk order. This ensures all generated buckets are **contiguous FICO intervals**.

Step 2: Prefix Sums for Fast Computation

```
cum_k = np.cumsum(y)
```

Prefix sums allow us to efficiently compute the number of defaults in any FICO interval in constant time. This significantly improves performance.

Step 3: Log-Likelihood Function

```
def log_likelihood(i, j):
```

This function computes the Bernoulli log-likelihood of defaults between two FICO indices. Laplace smoothing is applied to prevent numerical issues when all values are 0 or 1.

This function becomes the **core objective metric** for bucket quality.

Step 4: Likelihood Matrix Construction

LL[i, j] = log_likelihood(i, j)

A matrix is precomputed where each entry stores the log-likelihood of every possible FICO interval. This avoids recomputing values repeatedly in dynamic programming.

Step 5: Dynamic Programming for Optimal Binning

dp[k][j] = max(dp[k-1][i-1] + LL[i][j])

This recursion determines the optimal way to split the first `j` borrowers into `k` buckets.

Dynamic programming is used because:

- The problem has **optimal substructure**
- Exhaustive search would be computationally infeasible
- DP guarantees a **globally optimal solution**

Backtracking then extracts the optimal bucket boundaries.

Step 6: Rating Map Construction

rating_df = DataFrame([...])

Each bucket is summarized as:

- FICO lower bound
- FICO upper bound
- Total borrowers
- Number of defaults
- Probability of default

This forms the **credit rating map** used by the mortgage portfolio.

Step 7: Rating Assignment

```
df["Rating"] = df["fico_score"].apply(assign_rating)
```

Each borrower's FICO score is converted into a **categorical credit rating** using the rating map. This creates the input variable required for Charlie's categorical ML architecture.

Step 8: Visualization

Three plots are generated:

1. **PD vs Rating**
   Shows monotonic increase in default risk.
2. **FICO vs Default Distribution**
   Confirms that lower FICO scores correspond to higher default concentration.
3. **FICO Histogram with Bucket Boundaries**
   Visually validates the optimal bucket splits.

These plots provide both **model validation and business interpretability**.

---

4. Interpretation of the Output

The resulting rating map shows a **smooth, monotonic increase in PD as FICO decreases**, confirming:

- FICO is a strong predictor of mortgage default risk
- The quantization successfully preserved the underlying risk structure
- Each bucket represents a statistically meaningful credit grade

These ratings are now suitable as categorical variables for future machine learning models.

---

5. Business Significance

This system enables the bank to:

- Convert raw FICO into standardized **credit grades**
- Assign each mortgage a **probability of default**
- Support **categorical ML architectures**
- Perform **risk segmentation and policy design**
- Improve **loss provisioning and capital planning**

This approach reflects the foundation of **real-world mortgage scorecards**.

---

6. Conclusion

A dynamic programming–based optimal quantization framework was developed to discretize FICO scores into credit ratings by maximizing the Bernoulli log-likelihood of defaults. The resulting rating map produces monotonic risk buckets, strong default separation, and interpretable PD estimates. This provides a robust, future-proof categorical encoding of FICO scores for mortgage credit risk modeling.

If you want, I can also give you:

✔ A **one-page executive summary**
✔ A **short viva/interview explanation**
✔ A **flowchart of the algorithm**
✔ Or a **ready-to-submit PDF report structure**

Just tell me your submission format.