Follow these instructions exactly. You need to work in Weka's **Explorer** interface and may optionally use external visualization tools (Tableau, Power BI, or Python) for richer charts.

Use the provided Maersk_Shipping_Data_10000.csv

1. Setup & quick checks

1. Place Maersk_Shipping_Data_10000.csv in an easy-to-find folder.
2. Open **Weka → Explorer → Preprocess**.
3. Click **Open file…** and load the CSV. Confirm attributes loaded correctly (types: nominal / numeric).
4. Immediately inspect the **Attributes** list and the **Summary** panel for missing values and value ranges.

Output: dataset loaded in Weka; you can see column names like Shipment_ID, Origin_Port, Destination_Port, Blockchain_Verified, Documents_Submitted, Documentation_Errors, Customs_Clearance_Time_Hours, Transit_Time_Days, Delay_Days, Freight_Cost_USD, Cargo_Value_USD, Transaction_Trust_Score, Blockchain_Updates.

2. Data preprocessing

You need to clean and prepare the data before modeling.

1. **Remove identifier**: Use the **Remove** filter to drop Shipment_ID (not predictive).
2. **Handle missing values**: If any missing, apply **Filter → unsupervised.attribute.ReplaceMissingValues**.
3. **Convert nominal ↔ numeric if needed**:
   - Ensure Blockchain_Verified is nominal (Yes/No).
   - If you need numeric encoding for algorithms, use **NominalToBinary** or **StringToNominal** as appropriate.
4. **Create derived attributes (optional but recommended):**
   - Discretize Documentation_Errors into Low/Medium/High using **Discretize** (use 3 bins) for classification tasks.
   - Create Delay_Flag (0/1) if delay > 2 days — use **MathExpression** or derive externally and reload.
5. **Normalize/Standardize numeric attributes** for distance-sensitive models using **Normalize** or **Standardize**.

Output: clean dataset with a few derived attributes (e.g., Docs_Error_Level, Delay_Flag), no missing values, appropriate attribute types.

3. Exploratory visual analysis (use Weka & optionally external tools)

You need to visualize key relationships to form hypotheses.

1. In **Preprocess** click **Visualize All**. Inspect scatter plots (e.g., Delay_Days vs Customs_Clearance_Time_Hours; Freight_Cost_USD vs Delay_Days).
2. Use **Classify → Visualize classifier errors** after running models (later) to inspect misclassifications.
3. Export CSV or small aggregated tables for external visualization (e.g., pivot: Blockchain_Verified vs average Delay_Days) and plot in **Tableau/Power BI/Python** for charts like:
    - Bar chart: average Delay_Days by Blockchain_Verified (Yes/No)
    - Heatmap: Origin_Port × Destination_Port average delay
    - Time-series (if timestamps exist): delays over time

Output: visual evidence of patterns (e.g., verified shipments show lower average delays; higher documentation errors correlate with higher customs time).

4. ML Task 1 — Predict shipment delay (regression)

You need to predict Delay_Days (numeric) to enable proactive scheduling.

1. Open **Classify**. Select **Algorithm → functions → SMOreg** or **trees → RandomForest** (RandomForest works as regression in Weka when target is numeric).
2. Set **Test options → Cross-validation = 10 folds**.
3. Set target attribute = Delay_Days. Use predictors: Blockchain_Verified, Documentation_Errors, Customs_Clearance_Time_Hours, Transit_Time_Days, Transaction_Trust_Score, Blockchain_Updates, Cargo_Value_USD.
4. Run the model. Record **RMSE**, **MAE**, and **Correlation coefficient**.
5. Use **attribute selection** (Select attributes → InfoGain/Wrapper) to find the most predictive features.

Parameters to try: RandomForest with 100 trees; SMOreg with RBF kernel and grid-searched C.

Expected outputs & interpretation:

- A model with RMSE and MAE values; feature importance indicating which attributes (e.g., Customs_Clearance_Time_Hours, Documentation_Errors, Blockchain_Verified) most affect delays.
- Use model predictions to propose process changes (e.g., prioritize blockchain verification to reduce predicted delay).

5. ML Task 2 — Classify documentation risk (classification)

You need to classify shipments as HighRiskDocs vs LowRiskDocs based on error patterns.

1. Create a class attribute Docs_Risk by discretizing Documentation_Errors (e.g., 0→Low, 1–2→Medium, 3–4→High).
2. In **Classify**, choose **trees → J48** and **bayes → NaiveBayes** for comparison.
3. Use predictors like Documents_Submitted, Origin_Port, Carrier, Blockchain_Verified, Blockchain_Updates, Transaction_Trust_Score. Use 10-fold CV.
4. Evaluate **accuracy**, **precision**, **recall**, and **confusion matrix**. Export the decision tree for interpretation.

Expected outputs & interpretation:

- Decision rules showing combinations (e.g., Blockchain_Verified = No AND Documents_Submitted > 13 → High Risk).
- Use these rules to recommend targeted checks (e.g., mandatory verification or extra doc checks for No-verified shipments).

6. ML Task 3 — Cluster shipments (k-Means)

You need to segment shipments into groups for targeted optimization.

1. Go to **Cluster**. Choose **SimpleKMeans**. Set **k = 3** initially (Low-risk, Medium-risk, High-risk).
2. Use attributes: Delay_Days, Freight_Cost_USD, Documentation_Errors, Transaction_Trust_Score, Customs_Clearance_Time_Hours. Normalize first.
3. Run clustering and inspect cluster centroids and sizes.
4. Visualize clusters using **Visualize** in Preprocess (color by assigned cluster).

Expected outputs & interpretation:

- Centroids describing typical cluster profiles (e.g., Cluster 0: low delay, low cost, high trust — ideal; Cluster 2: high delay, high errors, low trust — problem group).
- Use clusters to design interventions (fast-track Cluster 2 for audits; automate Cluster 0 processes).

7. Optimization task — translate ML outputs to actionable optimization

You need to convert model insights into optimization actions and test their simulated impact.

1. **Define objectives** (choose at least two): minimize Delay_Days, minimize Freight_Cost_USD, minimize Documentation_Errors.
2. **Simple rule-based optimization (in-class exercise):**
   - Use regression model coefficients or decision rules to define interventions. Example rule: *if Blockchain_Verified=No and Documentation_Errors≥2, require pre-clearance*.
   - Simulate applying the rule on the dataset: create a copy of the dataset with Blockchain_Verified set to Yes for selected rows or Documentation_Errors reduced by 1 for flagged rows. Re-run regression to compare predicted Delay_Days and Freight_Cost_USD.
3. **Multi-objective thinking:** show trade-offs — improving verification (costly) vs reduced delays (savings). Compute net effect: saved delay-days × cost-per-day saved − verification cost.
4. **Optional advanced approach (external tool):** export model results to Excel or Python and run a simple linear/integer program (e.g., choose N shipments to upgrade verification under budget constraint to maximize delay reduction).

Expected outputs & interpretation:

- A before/after table showing predicted average delay and freight cost and estimated ROI for chosen interventions.
- A short recommendation list: e.g., scale blockchain verification for top 20% high-risk lanes; introduce document-precheck for shipments with docs>13.

8. Visualization & reporting

You need to create visuals that communicate results.

1. In Weka: export decision tree PNG, cluster assignments, and scatter plots.
2. In Tableau/Power BI/Python (optional): create dashboard with:
   - KPI tiles: avg Delay_Days, avg Freight_Cost_USD, % Blockchain_Verified.
   - Map or matrix: Origin_Port → Destination_Port heatmap of average delay.
   - Bar chart: average delay by Blockchain_Verified.
   - Decision tree flowchart and cluster centroid table.
3. Prepare a 4-slide summary: Problem → Methods (models used) → Key findings (visuals) → Recommendations & optimization plan.

Expected outputs: visuals that clearly show impact of blockchain verification and documentation quality on delays and costs.

9. Deliverables (what you must submit)

You need to submit the following files and items:

1. **Weka ARFF/CSV** of the final cleaned dataset + any derived attributes.
2. **Model outputs**:
   - Regression model (.model or text output) and performance metrics.
   - Classification model (J48 tree) with confusion matrix.
   - Clustering results with cluster centroids.
3. **Simulated optimization results**: before-and-after summary table and ROI estimate.
4. **Visuals**: PNGs or a short Tableau/Power BI dashboard (or exported images).
5. **One-page summary**: concise recommendations (which lanes/processes to prioritize, expected benefit).

10. Success criteria & grading rubric (suggested)

- Data preprocessing correctness (15%).
- Model quality & evaluation (30%). — report RMSE/MAE, accuracy, precision/recall.
- Practical optimization simulation & ROI thinking (30%).
- Visualizations & clarity of recommendations (15%).
- Submission completeness & reproducibility (5%).

Quick tips & parameter suggestions

- Use **10-fold CV** consistently for model evaluation.
- Try **RandomForest (100 trees)** and **J48 (default pruning)** as baseline strong models.
- For k-Means, test **k=2..6** and choose by silhouette or by interpretability.

Follow these steps and record your observations. After finishing, be ready to present your dashboard and interpret how specific optimizations (e.g., increasing blockchain verification, introducing pre-document checks) would reduce delays and costs in Maersk-like operations.