TrackTempo Inference Pipeline Review Summary

==========

1. Loading Label Encoders + Data

- Loads correctly via joblib and pandas.
- No issues with compatibility or format.

2. Applying Label Encoders (Safe Mode)

- Fallback logic for unknown categories is well-implemented.
- Prevents shape mismatches with model embeddings.

Enhancement:

- Log the count of fallback replacements per column for visibility.

3. Batching the Data

- Uses batch_races correctly with `label_col=None`.
- Supports tabular and NLP features.

Potential Risk:

- Assumes batch order matches original DataFrame.
- No ID tracking for predictions (e.g., race_id, horse_id).

4. Model Setup

- Correct use of RaceTransformer model.
- Clean load and CPU-safe.

Enhancement:

- Add `--device` argument to enable GPU support.

5. Inference Execution

- Sigmoid applied to logits to get probabilities.
- Runs without error, outputs expected shape.

Enhancement:

- Ensure batch filtering always skips empty races.

6. Prediction Storage

- Adds predicted_win_prob column to original DataFrame.
- Exports to CSV with directory auto-creation.

Potential Risk:

- Implicit reliance on batch order == df order.
- Could break if future batching introduces shuffling.

Enhancement:

- Merge predictions back using unique identifiers (e.g., race_id, horse_id).

Recommended Enhancements

==========

1. Add a tracking dictionary for (race_id, horse_id) in batch_races.
2. In inference script, store predictions in a separate DataFrame:

race_id | horse_id | predicted_win_prob

3. Merge this predictions DataFrame back into original df using:

df = df.merge(pred_df, on=["race_id", "horse_id"])

4. Log fallback encodes per column:

e.g., "Filled 23 unknowns in 'headgear' column with fallback."

5. Add a --device argument to inference CLI:

device = torch.device("cuda" if args.device == "cuda" else "cpu")

Overall Summary

==========

- All core logic is sound and functional.
- A few subtle assumptions (like DataFrame alignment) could be hardened for scale.
- Strong foundation for production-ready inference.