Progress Report

Completed Work:

Data Preparation (Feature Engineering, Feature Cleaning, Imputation)

Data Loading and Preprocessing:

- Implemented functions to load both static data from .csv files and dynamic data (time series data) from .parquet files.
- Dynamic Data
- Extracted statistical features from time series data, so that our machine learning model can use it.
- Standardized these statistical features and imputed the missing values with the means of them.
- Performed principle component analysis, and got 15 most significant principle components representing more than 80% variances, reducing the dimensionality, thereby mitigating redundancy and overfitting risks.
- Static Data
- Cleaned outliers and extreme values (e.g., implausible body fat rates or bone density values) to ensure values remain within reasonable boundaries. Replaced them with NaN, and performed impulation for them later.
- Deleted unrelated features, and grouped the remaining features by the age, eliminating the impact of age differences.
- Standardized features to avoid issues with differing feature scales.
- Conducted quantile-based binning to transform skewed or outlier-prone continuous features into categorized intervals.
- Imputed the NaN values with a pre-defined imputer. The imputer predicts and fills missing values by fitting a Lasso-based regression model based on other features instead of just filling with the mean. When the missing ratio is too high, or the data is sufficient, the mean will be used.
- Merged processed static features with the PCA-reduced dynamic features using `id`, resulting in a final dataset suitable for model training.

Modeling and Cross-Validation:

- Employed multiple models (LightGBM, XGBoost, CatBoost, ExtraTrees) and performed Stratified K-Fold cross-validation to obtain stable estimates of Cohen's Kappa scores.
- Optimized thresholds for mapping continuous predictions into discrete categories (0–3) by maximizing the `cohen_kappa_score`.

Hyperparameter Tuning:

- Used the Optuna library to tune hyperparameters for XGBoost, LightGBM, CatBoost, etc., to further enhance model performance.

Next Step:

Determine the final ensemble strategy

We will compare the performance of the majority voting method and the weighted average method on the validation set, especially the high and low and fluctuation of the QWK score and choose the more robust ensemble method accordingly.

Further optimize the classification threshold

Although the current classification threshold is based on the average results of cross-validation, We also plan to check the consistency of the threshold on the validation set and the test set, and try to do more fine-tuning of some key classification boundaries.

Conduct a more in-depth error analysis

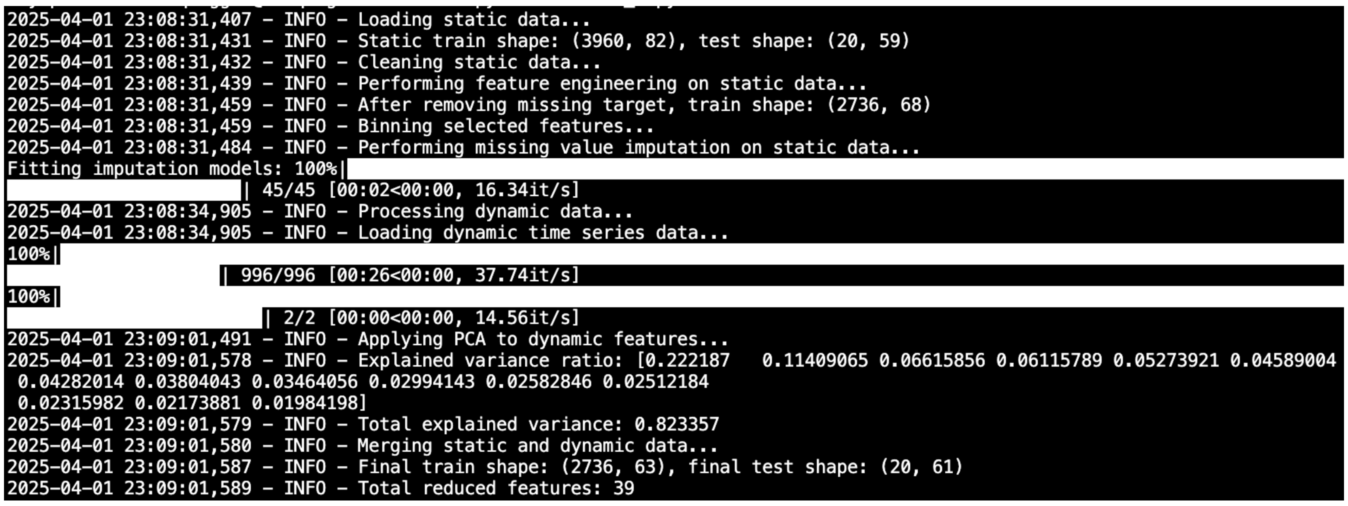
Based on the confusion matrix of the ensemble model, we will analyze which categories are most likely to be misclassified; whether there are systematic deviations in certain score segments; and combine the eigenvalues to explore whether there are boundary samples that are difficult for the model to distinguish.

Optimize and improve chart visualization

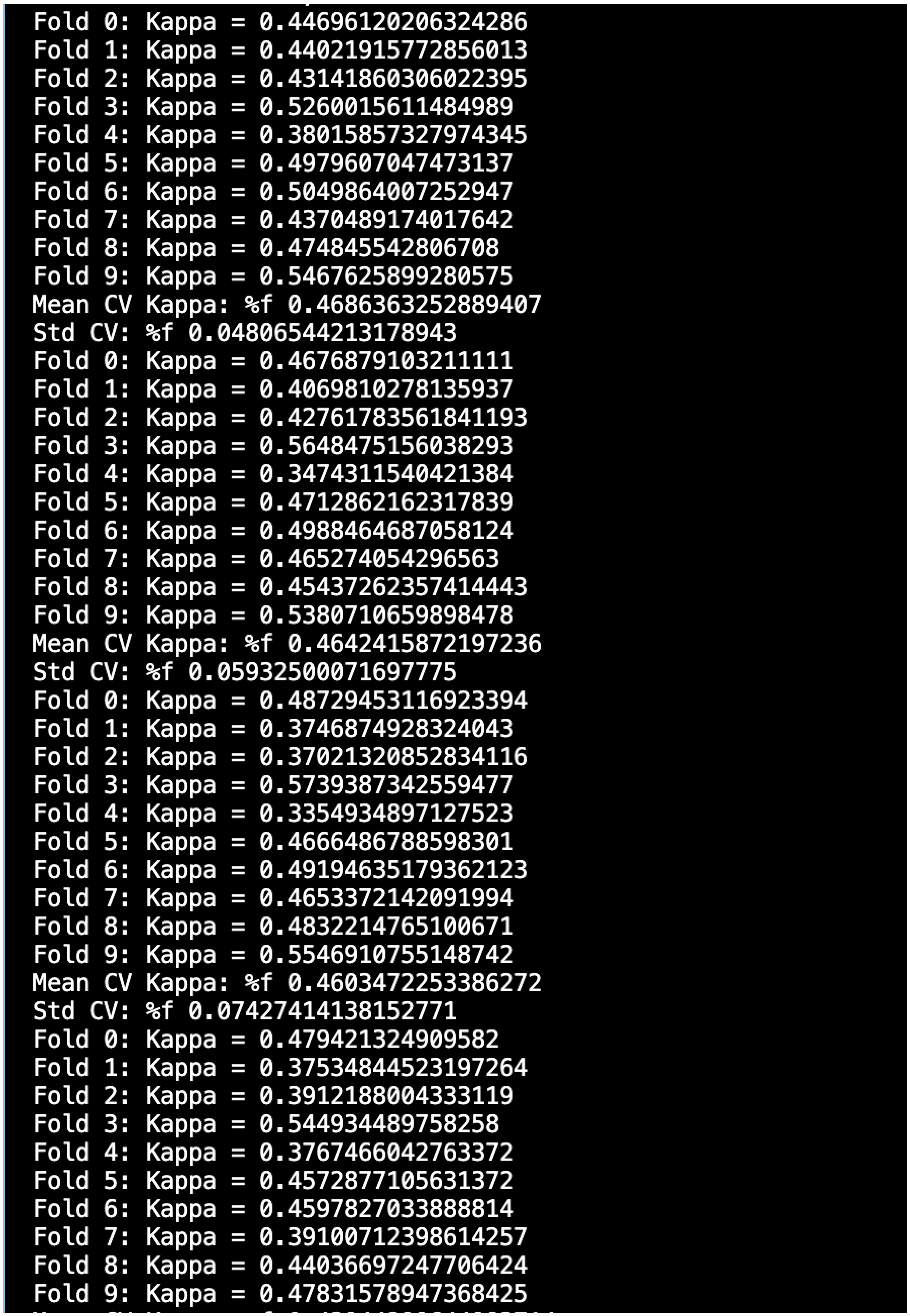
We will further adjust the performance of the chart, including: optimizing the axis range and scale; adding clearer titles, colors, labels, etc.; improving the interpretability and visual optimization of the chart in the report.

In-depth interpretation of feature importance

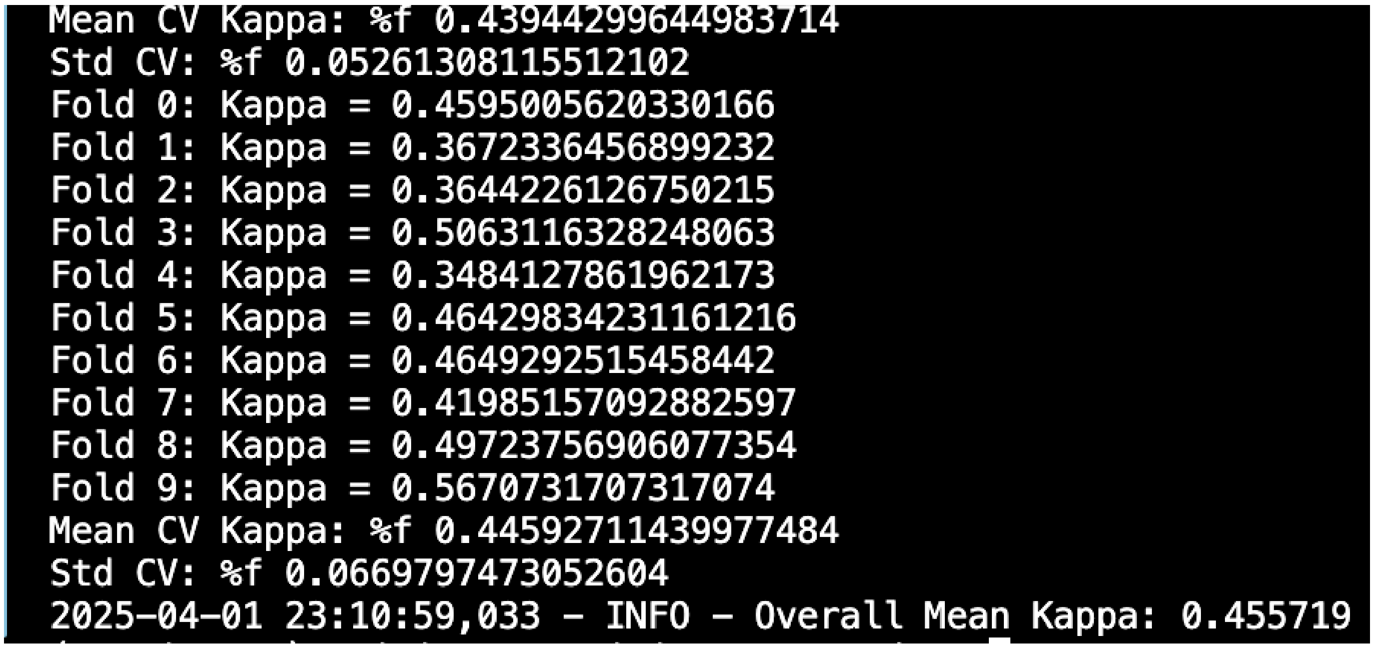
We plan to conduct a comparative analysis of the feature importance results of multiple models, focusing on: whether there are features that are considered important by multiple models; the differences in the features that different models focus on; and the inspiration for the interpretability of the results and the model's generalization ability.



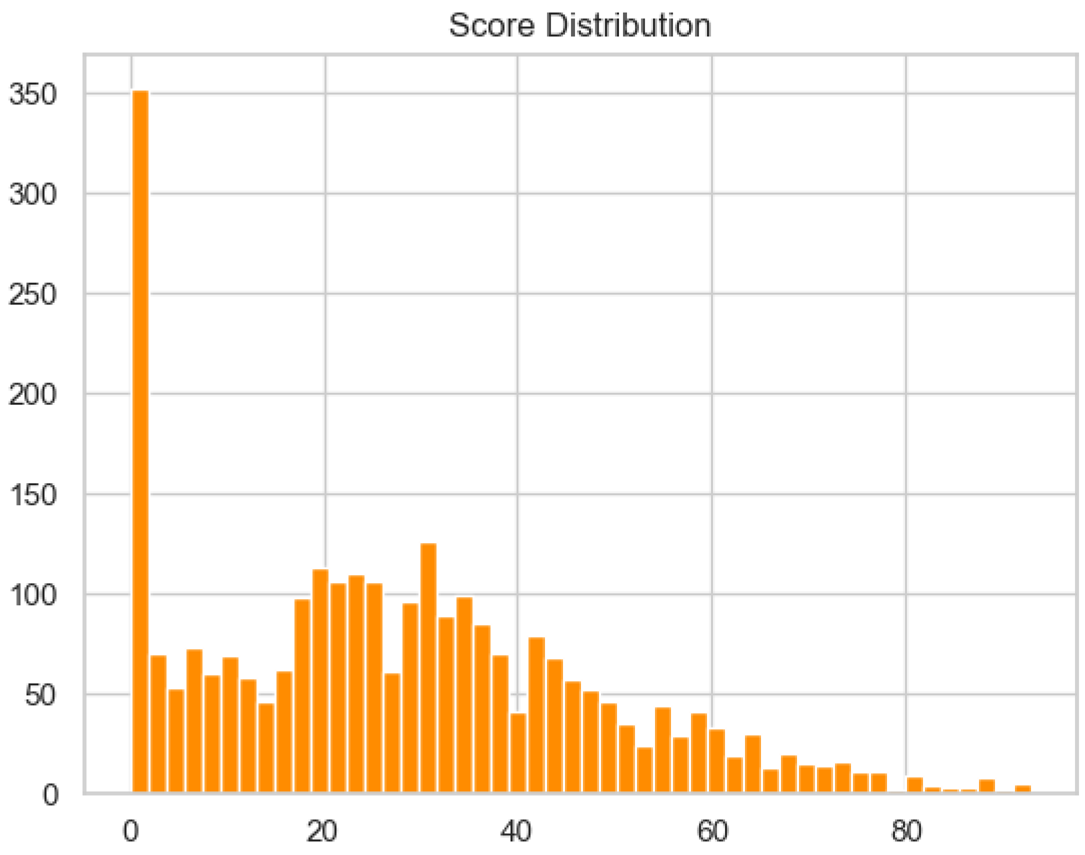
```
2025-04-01 23:08:31,407 - INFO - Loading static data...
2025-04-01 23:08:31,431 - INFO - Static train shape: (3960, 82), test shape: (20, 59)
2025-04-01 23:08:31,432 - INFO - Cleaning static data...
2025-04-01 23:08:31,439 - INFO - Performing feature engineering on static data...
2025-04-01 23:08:31,459 - INFO - After removing missing target, train shape: (2736, 68)
2025-04-01 23:08:31,459 - INFO - Binning selected features...
2025-04-01 23:08:31,484 - INFO - Performing missing value imputation on static data...
Fitting imputation models: 100%|████████████████████████████████████████
██████████████████████| 45/45 [00:02<00:00, 16.34it/s]
2025-04-01 23:08:34,905 - INFO - Processing dynamic data...
2025-04-01 23:08:34,905 - INFO - Loading dynamic time series data...
100%|
████████████████████| 996/996 [00:26<00:00, 37.74it/s]
100%|
████████████████████████| 2/2 [00:00<00:00, 14.56it/s]
2025-04-01 23:09:01,491 - INFO - Applying PCA to dynamic features...
2025-04-01 23:09:01,578 - INFO - Explained variance ratio: [0.222187   0.11409065 0.06615856 0.06115789 0.05273921 0.04589004
 0.04282014 0.03804043 0.03464056 0.02994143 0.02582846 0.02512184
 0.02315982 0.02173881 0.01984198]
2025-04-01 23:09:01,579 - INFO - Total explained variance: 0.823357
2025-04-01 23:09:01,580 - INFO - Merging static and dynamic data...
2025-04-01 23:09:01,587 - INFO - Final train shape: (2736, 63), final test shape: (20, 61)
2025-04-01 23:09:01,589 - INFO - Total reduced features: 39
```



```
Fold 0: Kappa = 0.44696120206324286
Fold 1: Kappa = 0.44021915772856013
Fold 2: Kappa = 0.43141860306022395
Fold 3: Kappa = 0.5260015611484989
Fold 4: Kappa = 0.38015857327974345
Fold 5: Kappa = 0.4979607047473137
Fold 6: Kappa = 0.5049864007252947
Fold 7: Kappa = 0.4370489174017642
Fold 8: Kappa = 0.474845542806708
Fold 9: Kappa = 0.5467625899280575
Mean CV Kappa: %f 0.4686363252889407
Std CV: %f 0.04806544213178943
Fold 0: Kappa = 0.4676879103211111
Fold 1: Kappa = 0.4069810278135937
Fold 2: Kappa = 0.42761783561841193
Fold 3: Kappa = 0.5648475156038293
Fold 4: Kappa = 0.3474311540421384
Fold 5: Kappa = 0.4712862162317839
Fold 6: Kappa = 0.4988464687058124
Fold 7: Kappa = 0.465274054296563
Fold 8: Kappa = 0.45437262357414443
Fold 9: Kappa = 0.5380710659898478
Mean CV Kappa: %f 0.4642415872197236
Std CV: %f 0.05932500071697775
Fold 0: Kappa = 0.48729453116923394
Fold 1: Kappa = 0.3746874928324043
Fold 2: Kappa = 0.37021320852834116
Fold 3: Kappa = 0.5739387342559477
Fold 4: Kappa = 0.3354934897127523
Fold 5: Kappa = 0.4666486788598301
Fold 6: Kappa = 0.49194635179362123
Fold 7: Kappa = 0.4653372142091994
Fold 8: Kappa = 0.4832214765100671
Fold 9: Kappa = 0.5546910755148742
Mean CV Kappa: %f 0.4603472253386272
Std CV: %f 0.07427414138152771
Fold 0: Kappa = 0.479421324909582
Fold 1: Kappa = 0.37534844523197264
Fold 2: Kappa = 0.3912188004333119
Fold 3: Kappa = 0.544934489758258
Fold 4: Kappa = 0.3767466042763372
Fold 5: Kappa = 0.4572877105631372
Fold 6: Kappa = 0.4597827033888814
Fold 7: Kappa = 0.39100712398614257
Fold 8: Kappa = 0.44036697247706424
Fold 9: Kappa = 0.47831578947368425
```



```
Mean CV Kappa: %f 0.43944299644983714
Std CV: %f 0.05261308115512102
Fold 0: Kappa = 0.4595005620330166
Fold 1: Kappa = 0.3672336456899232
Fold 2: Kappa = 0.3644226126750215
Fold 3: Kappa = 0.5063116328248063
Fold 4: Kappa = 0.3484127861962173
Fold 5: Kappa = 0.46429834231161216
Fold 6: Kappa = 0.4649292515458442
Fold 7: Kappa = 0.41985157092882597
Fold 8: Kappa = 0.49723756906077354
Fold 9: Kappa = 0.5670731707317074
Mean CV Kappa: %f 0.44592711439977484
Std CV: %f 0.0669797473052604
2025-04-01 23:10:59,033 - INFO - Overall Mean Kappa: 0.455719
```


Score Distribution