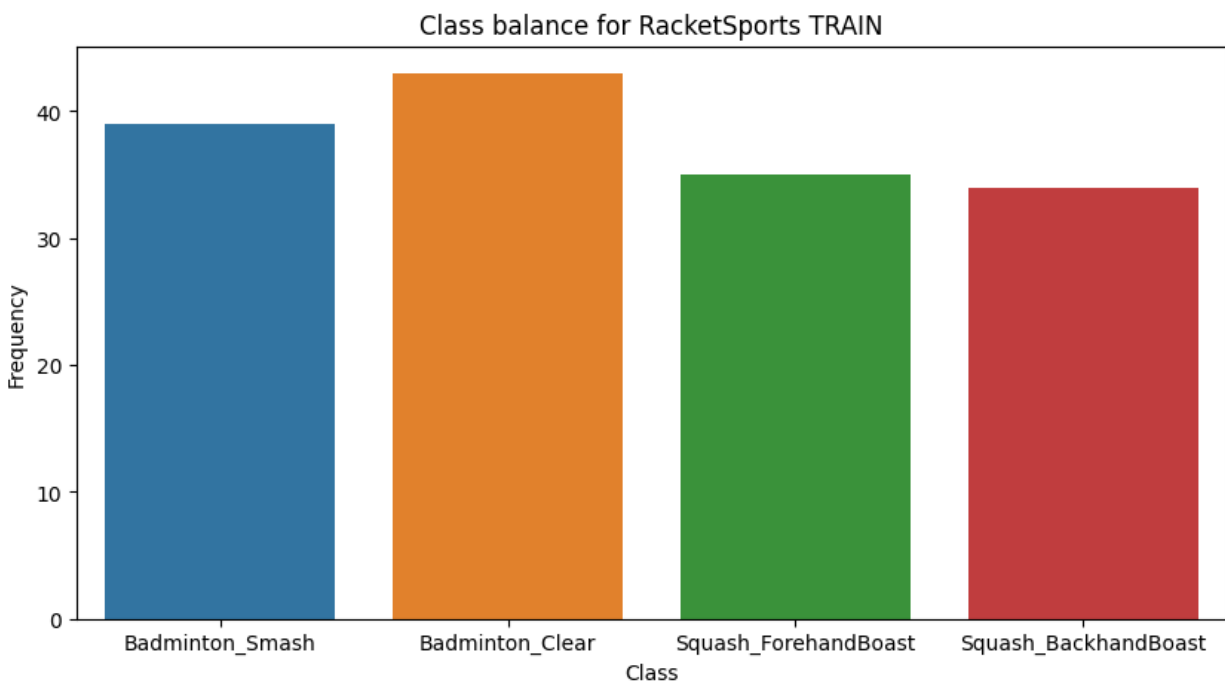


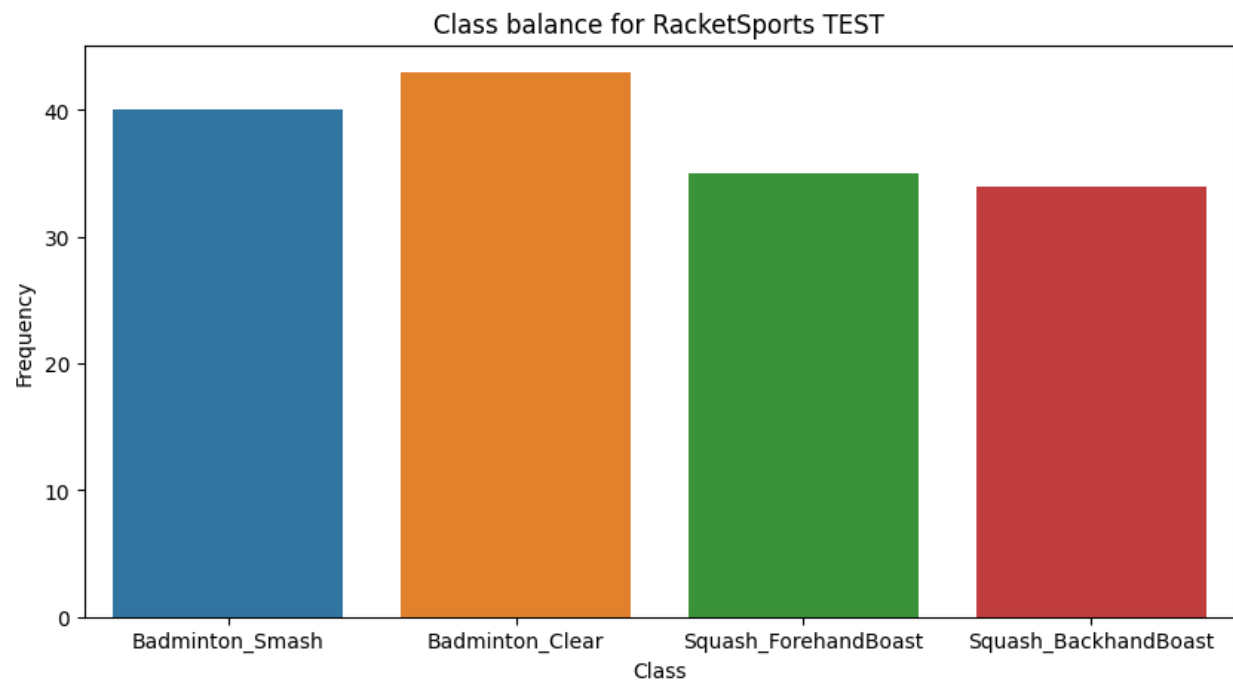
RacketSport Dataset

Distributia pe catogorii este vizibila mai jos.

Utilizez două funcții diferite - `plot_class_balance_seaborn()` și `plot_class_balance_pandas()` - pentru a afișa grafice cu numărul de exemple pentru fiecare clasă în seturile de date. Graficele sunt create atât cu ajutorul bibliotecii seaborn, cât și cu ajutorul metodei `plot.bar()` din biblioteca pandas.

Aceste grafice oferă o imagine vizuală a distribuției claselor în fiecare set de date. Ele prezintă următoarele informații. Graficele pentru RacketSports Test și Train arată echilibrul între cele 4 clase n seturile de date de testare și de antrenament.





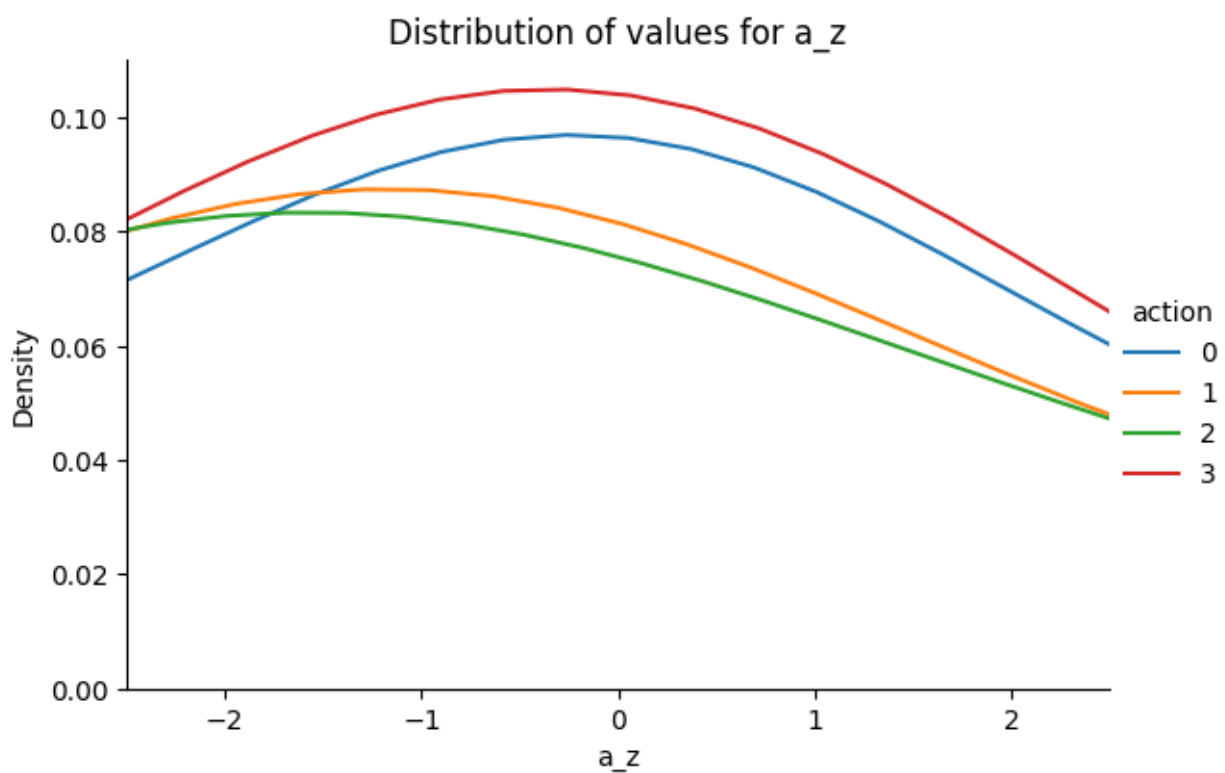
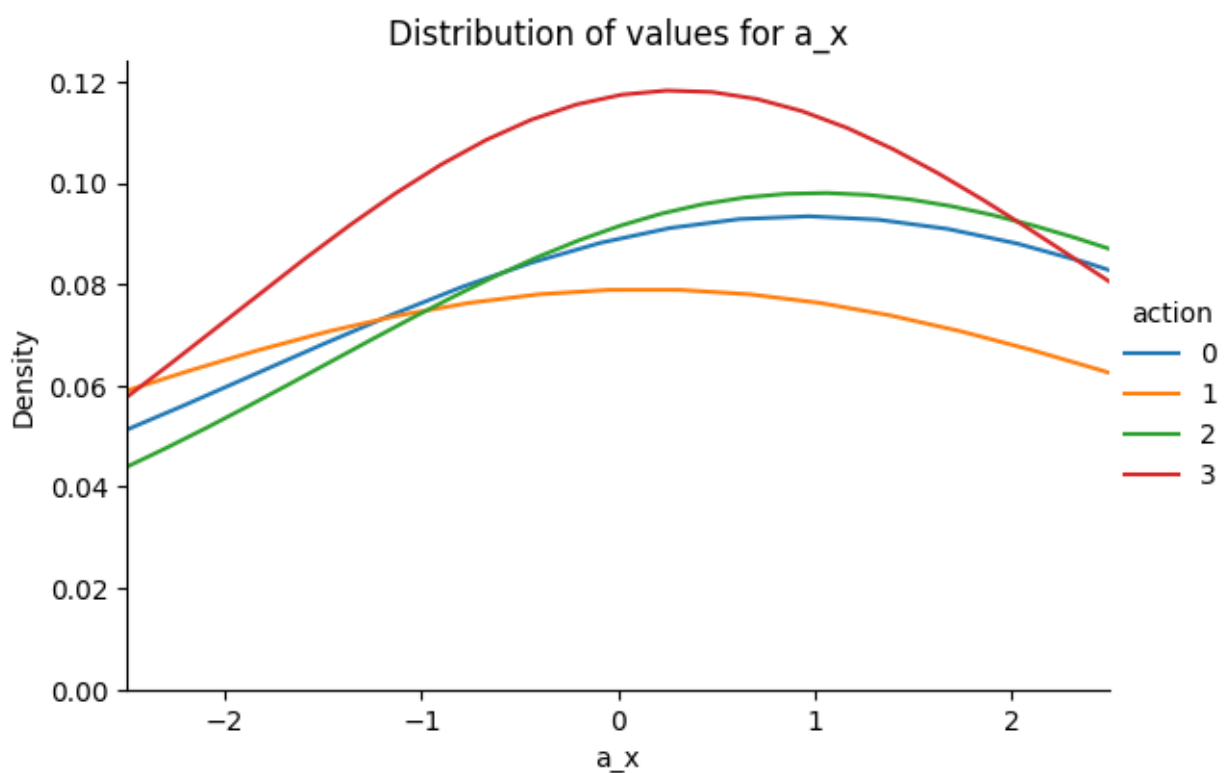
Fiecare acțiune este reprezentată de două grafice, unul pentru accelerometru și altul pentru giroscop. Pe axa orizontală a graficelor se află timpul, în timp ce axa verticală reprezintă măsurătorile de accelerare (accelerometru) și rotație (giroscop). De asemenea, graficele prezintă trei linii colorate diferite, reprezentând cele trei axe: X, Y și Z.

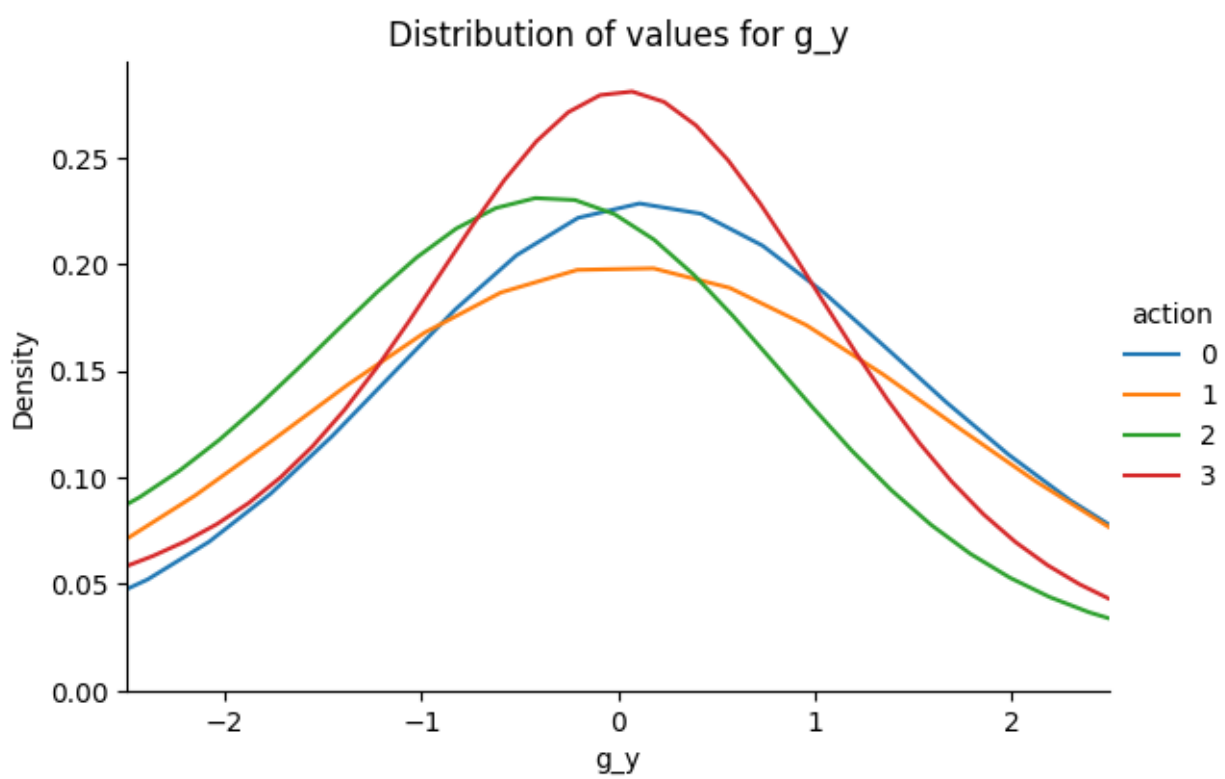
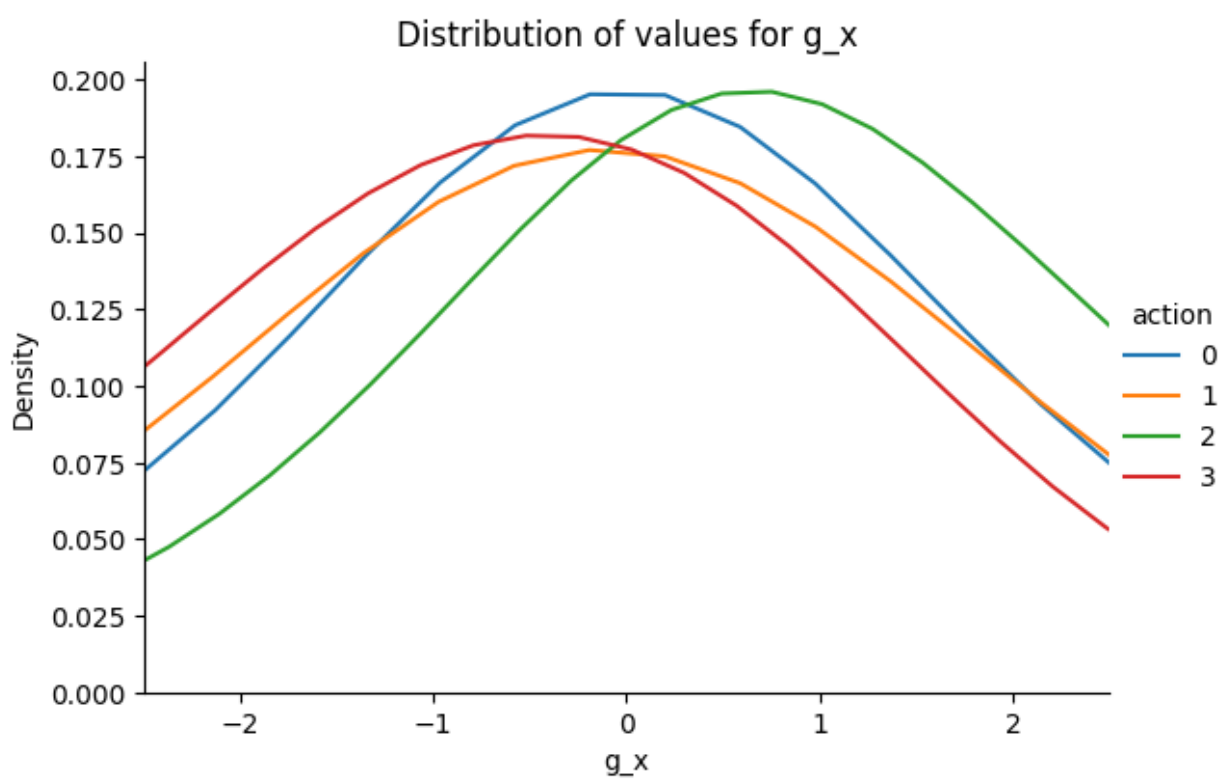
Spike-urile reprezintă variații bruște ale accelerației sau rotației. În cazul sporturilor cu rachetă, aceste spike-uri pot reprezenta momente în care jucătorul lovește mingea, schimbă direcția sau accelerează brusc. Spike-urile pot ajuta la diferențierea dintre diferite tipuri de lovituri sau mișcări, deoarece acestea pot avea profiluri diferite de accelerație și rotație și putem astfel să clasificăm informația în diferite acțiuni/label-uri precum Badminton_Clear, Badminton_Smash, Squash_BackhandBoast, Squash_ForehandBoast.

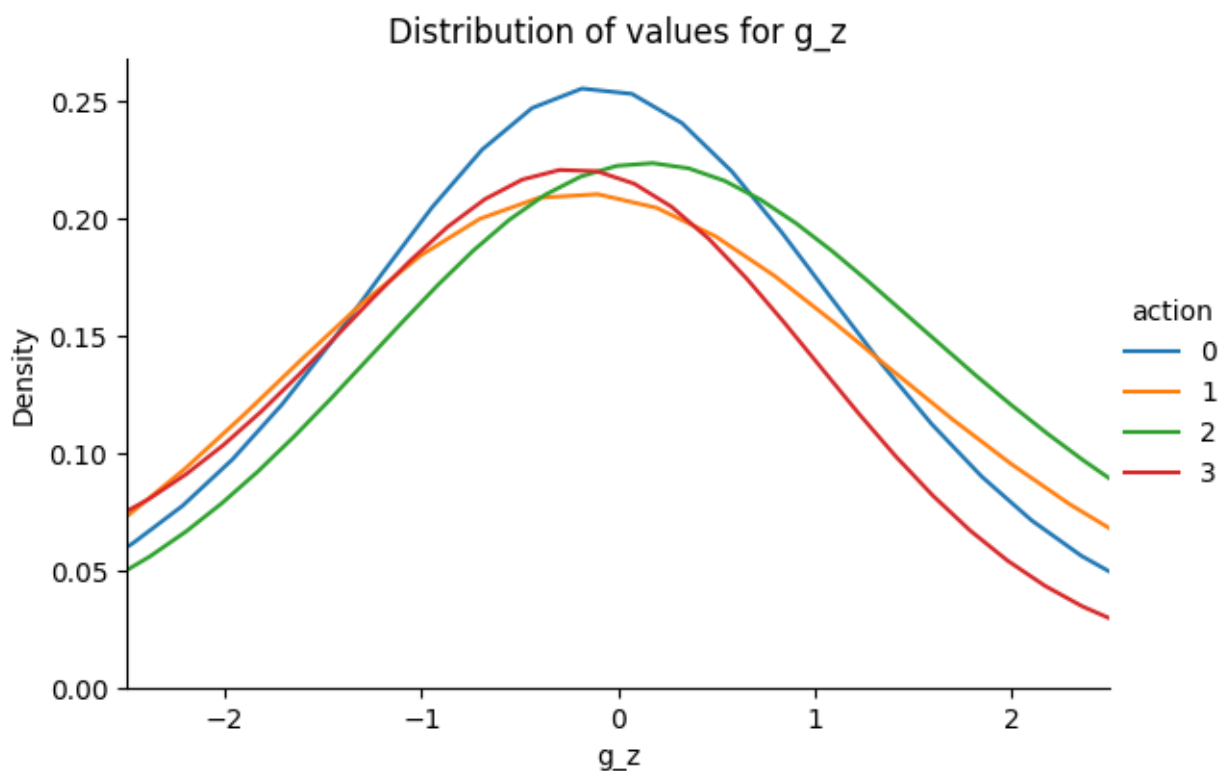
Părțile mai netede pot reprezenta perioade de mișcare constantă sau de repaus. De exemplu, un jucător care se deplasează lent sau care se pregătește pentru o lovitură. În aceste cazuri, variațiile accelerației și rotației pot fi mai mici și mai uniforme, ceea ce duce la o suprafață netedă a graficului.



Graficele de mai sus prezintă distribuția valorilor pentru fiecare axă a datelor de accelerometru și giroscop, împărțite după acțiuni. Observăm o distribuție uniformă, gaussiană.







Random Forest: Mean Accuracy: 0.8540, Std: 0.0408

XGBoost: Mean Accuracy: 0.8015, Std: 0.0754

SVM: Mean Accuracy: 0.8473, Std: 0.0691

Random Forest: Best Params: {'max_depth': 20, 'max_features': 'sqrt', 'n_estimators': 100}, Best Score: 0.8809

XGBoost: Best Params: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}, Best Score: 0.7815

SVM: Best Params: {'C': 1, 'kernel': 'rbf'}, Best Score: 0.8804

Model: Random Forest

Best Hyperparameters: {'max_depth': 20, 'max_features': 'sqrt', 'n_estimators': 100}

precision recall f1-score support

Badminton_Clear	0.69	0.81	0.74	43
Badminton_Smash	0.75	0.60	0.67	40
Squash_BackhandBoast	0.94	1.00	0.97	34
Squash_ForehandBoast	1.00	0.94	0.97	35

accuracy		0.83	152
macro avg	0.85	0.84	0.84
weighted avg	0.83	0.83	0.83

Confusion Matrix:

```
[[35 7 1 0]
 [15 24 1 0]
 [ 0 0 34 0]
 [ 1 1 0 33]]
```

Model: XGBoost

Best Hyperparameters: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}

	precision	recall	f1-score	support
Badminton_Clear	0.70	0.74	0.72	43
Badminton_Smash	0.68	0.65	0.67	40
Squash_BackhandBoast	0.94	0.94	0.94	34
Squash_ForehandBoast	0.97	0.94	0.96	35
accuracy			0.81	152
macro avg	0.82	0.82	0.82	152
weighted avg	0.81	0.81	0.81	152

Confusion Matrix:

```
[[32 9 1 1]
 [13 26 1 0]
 [ 0 2 32 0]
 [ 1 1 0 33]]
```

Model: SVM

Best Hyperparameters: {'C': 1, 'kernel': 'rbf'}

	precision	recall	f1-score	support
Badminton_Clear	0.67	0.81	0.74	43
Badminton_Smash	0.77	0.60	0.68	40
Squash_BackhandBoast	0.97	1.00	0.99	34
Squash_ForehandBoast	0.97	0.94	0.96	35
accuracy		0.83		152
macro avg	0.85	0.84	0.84	152
weighted avg	0.83	0.83	0.83	152

Confusion Matrix:

```
[[35 7 1 0]
 [15 24 0 1]
 [ 0 0 34 0]
 [ 2 0 0 33]]
```

Rezultatele prezentate mai sus reprezintă performanțele a trei modele de clasificare diferite - XGBoost, SVM și Random Forest - pe un set de date care conține informații despre diferite tipuri de lovituri în sporturile badminton și squash. Performanțele modelelor sunt evaluate în termeni de precizie, recall, f1-score și matricea de confuzie pentru fiecare categorie de lovituri.

1. Modelul Random Forest:

- Parametrii optimi găsiți: 'max_depth': 20, 'max_features': 'sqrt', 'n_estimators': 100
- Acuratețea medie: 0.83
- Performanța detaliată pentru fiecare categorie de lovituri:
 - Badminton_Clear: precizie 0.69, recall 0.81, f1-score 0.74
 - Badminton_Smash: precizie 0.75, recall 0.60, f1-score 0.67
 - Squash_BackhandBoast: precizie 0.94, recall 1.00, f1-score 0.97
 - Squash_ForehandBoast: precizie 1.00, recall 0.94, f1-score 0.97
- Matricea de confuzie indică un număr relativ mare de confuzii între Badminton_Clear și Badminton_Smash.

2. Modelul XGBoost:

- Parametrii optimi găsiți: 'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200
- Acuratețea medie: 0.81
- Performanța detaliată pentru fiecare categorie de lovituri:
 - Badminton_Clear: precizie 0.70, recall 0.74, f1-score 0.72
 - Badminton_Smash: precizie 0.68, recall 0.65, f1-score 0.67
 - Squash_BackhandBoast: precizie 0.94, recall 0.94, f1-score 0.94
 - Squash_ForehandBoast: precizie 0.97, recall 0.94, f1-score 0.96
- Matricea de confuzie arată și aici confuzii între Badminton_Clear și Badminton_Smash, dar într-o măsură mai mică decât în cazul modelului Random Forest.

•

3. Modelul SVM:

- Parametrii optimi găsiți: 'C': 1, 'kernel': 'rbf'
- Acuratețea medie: 0.83
- Performanța detaliată pentru fiecare categorie de lovituri:
 - Badminton_Clear: precizie 0.67, recall 0.81, f1-score 0.74
 - Badminton_Smash: precizie 0.77, recall 0.60, f1-score 0.68
 - Squash_BackhandBoast: precizie 0.97, recall 1.00, f1-score 0.99

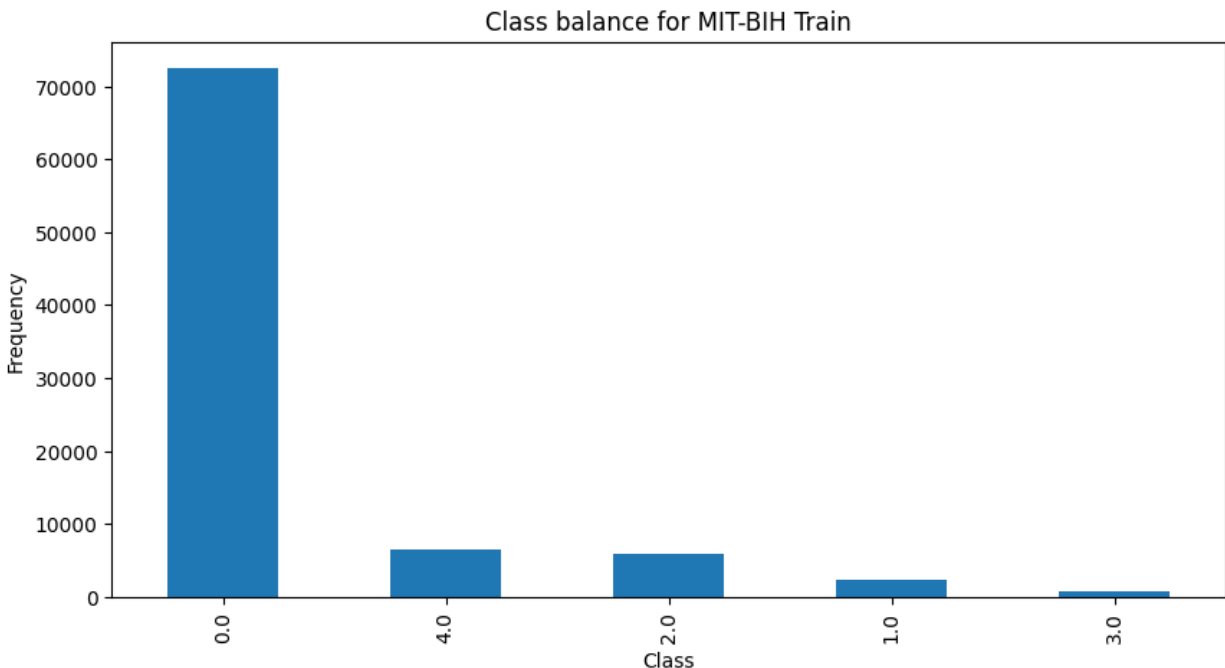
Am adaugat in format csv in arhiva, toate valorile pentru toti parametrii de antrenare folositi la fiecare model.

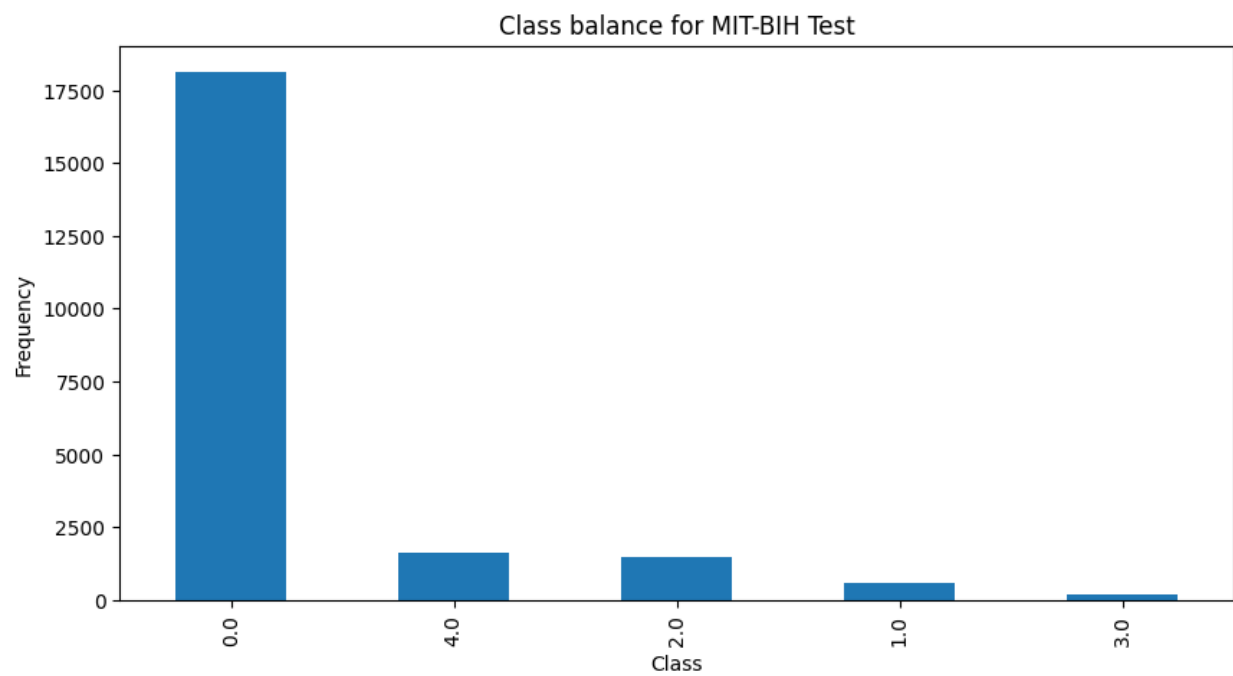
MIT-BIH/PTBDB

Distributia pe catogorii este vizibila mai jos.

Utilizez două funcții diferite - `plot_class_balance_seaborn()` și `plot_class_balance_pandas()` - pentru a afișa grafice cu numărul de exemple pentru fiecare clasă în seturile de date. Graficele sunt create atât cu ajutorul bibliotecii `seaborn`, cât și cu ajutorul metodei `plot.bar()` din biblioteca `pandas`.

Aceste grafice oferă o imagine vizuală a distribuției claselor în fiecare set de date. Ele prezintă următoarele informații. Graficele pentru MIT-BIH Test și MIT-BIH Train arată echilibrul între cele 5 clase de aritmii în seturile de date de testare și de antrenament. Aceste grafice ajuta la a intelege proporția dintre diferitele tipuri de aritmii în fiecare set de date și identificarea dezechilibrelor între clase.

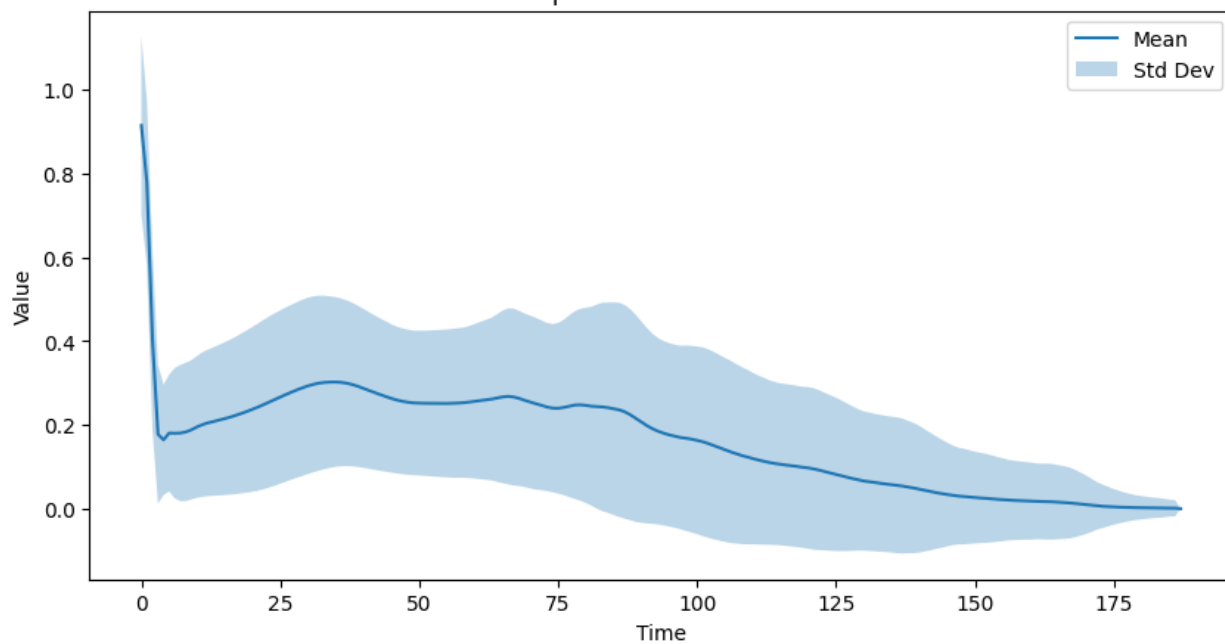




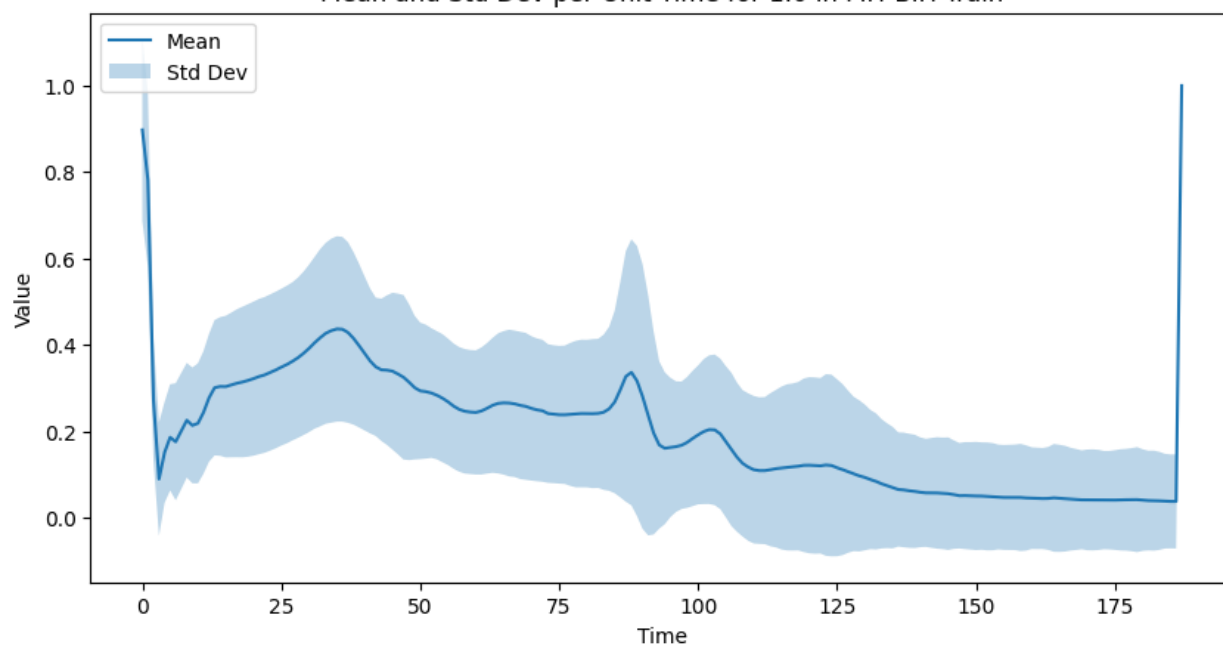
Valorile medii ale semnalelor ECG pentru fiecare clasă oferă o reprezentare generală a formei de undă pentru un tip specific de aritmie. Aceste valori medii pot fi folosite pentru a înțelege caracteristicile tipice ale formelor de undă pentru fiecare clasă și pentru a identifica diferențele între clase. De exemplu, o clasă cu o formă de undă medie distinctă de celelalte clase poate fi mai ușor de clasificat, în timp ce clase cu forme de undă similare pot necesita caracteristici suplimentare sau modele mai complexe pentru a fi diferențiate în mod corect.

Deviațiile standard oferă o măsura variației semnalelor ECG în cadrul fiecărei clase. O deviație standard mare într-o anumită zonă a formei de undă sugerează o variație semnificativă între semnalele individuale ale acelei clase, ceea ce poate indica o dificultate în distingerea acelei clase de la alte clase pe baza acelei părți a semnalului. Pe de altă parte, o deviație standard mică sugerează o variație redusă între semnalele individuale, ceea ce poate facilita clasificarea pe baza acelei părți a semnalului.

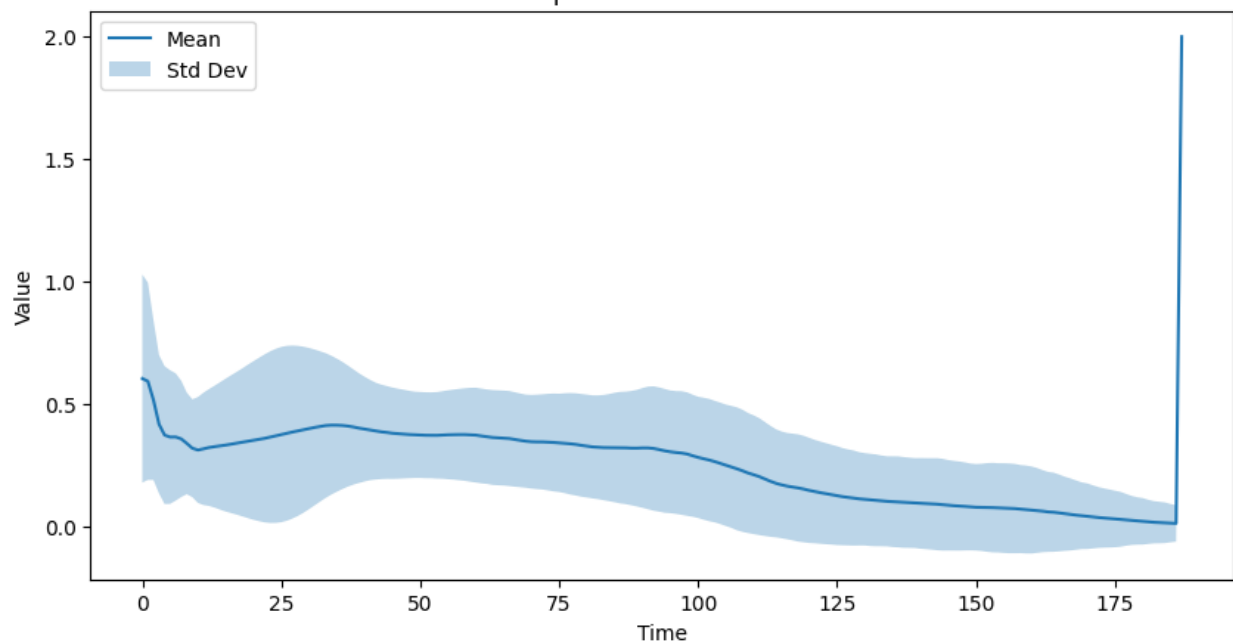
Mean and Std Dev per Unit Time for 0.0 in MIT-BIH Train



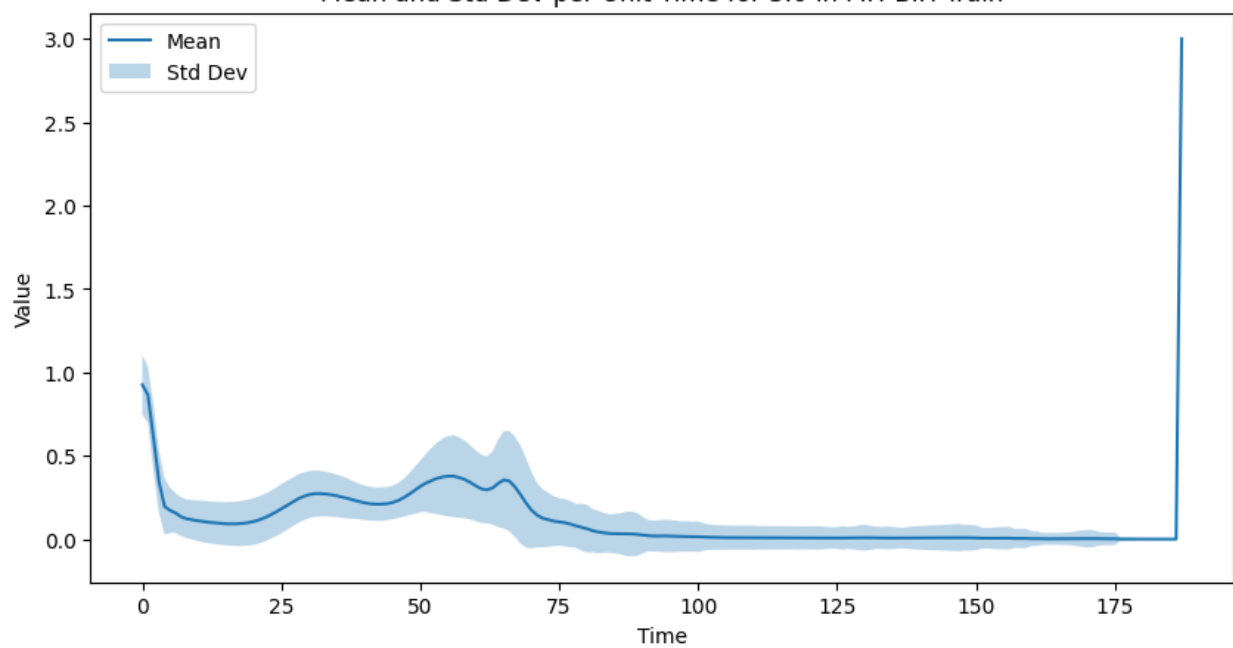
Mean and Std Dev per Unit Time for 1.0 in MIT-BIH Train

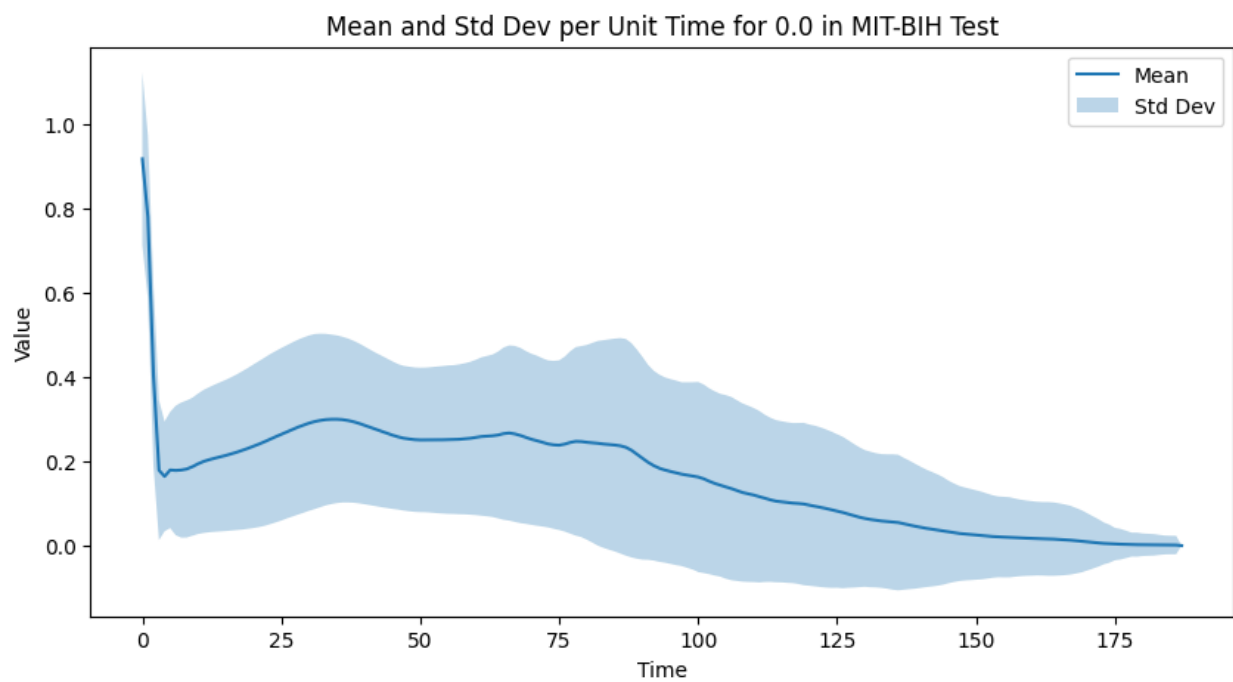
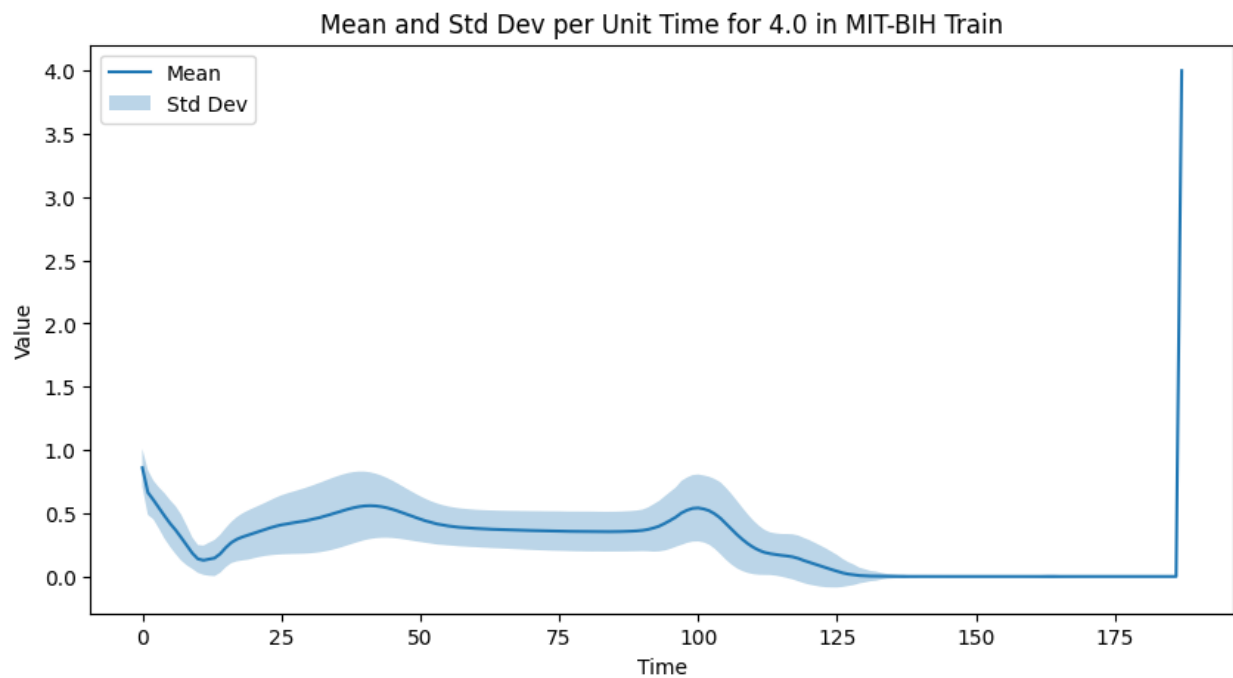


Mean and Std Dev per Unit Time for 2.0 in MIT-BIH Train

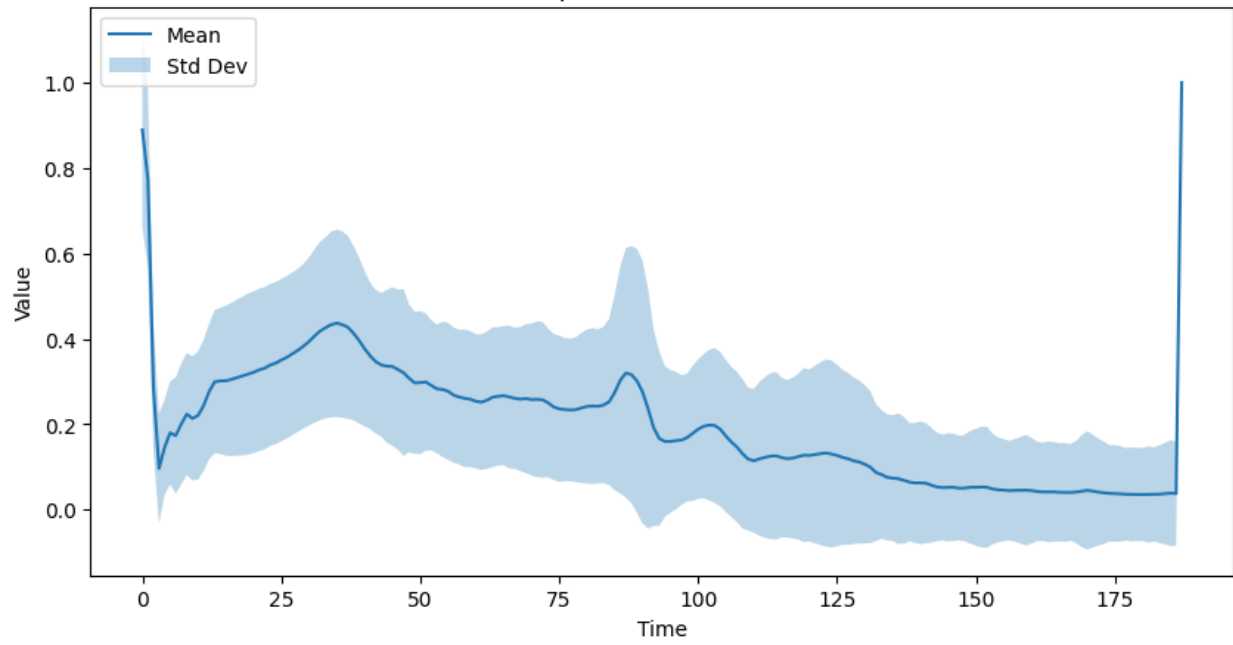


Mean and Std Dev per Unit Time for 3.0 in MIT-BIH Train

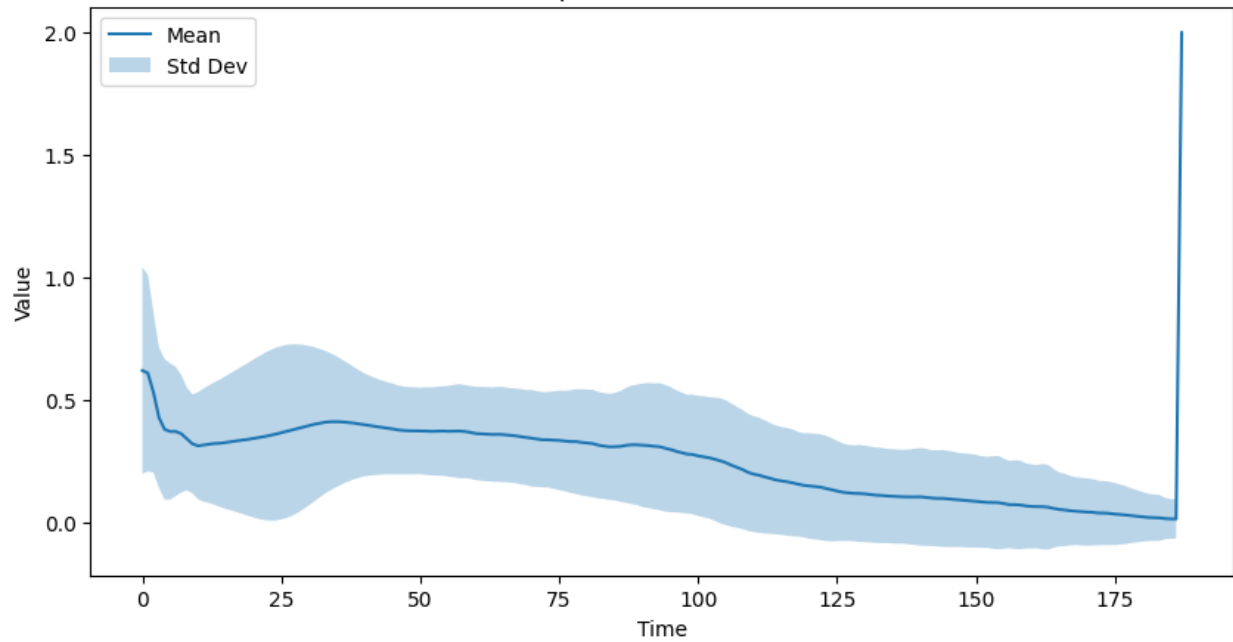




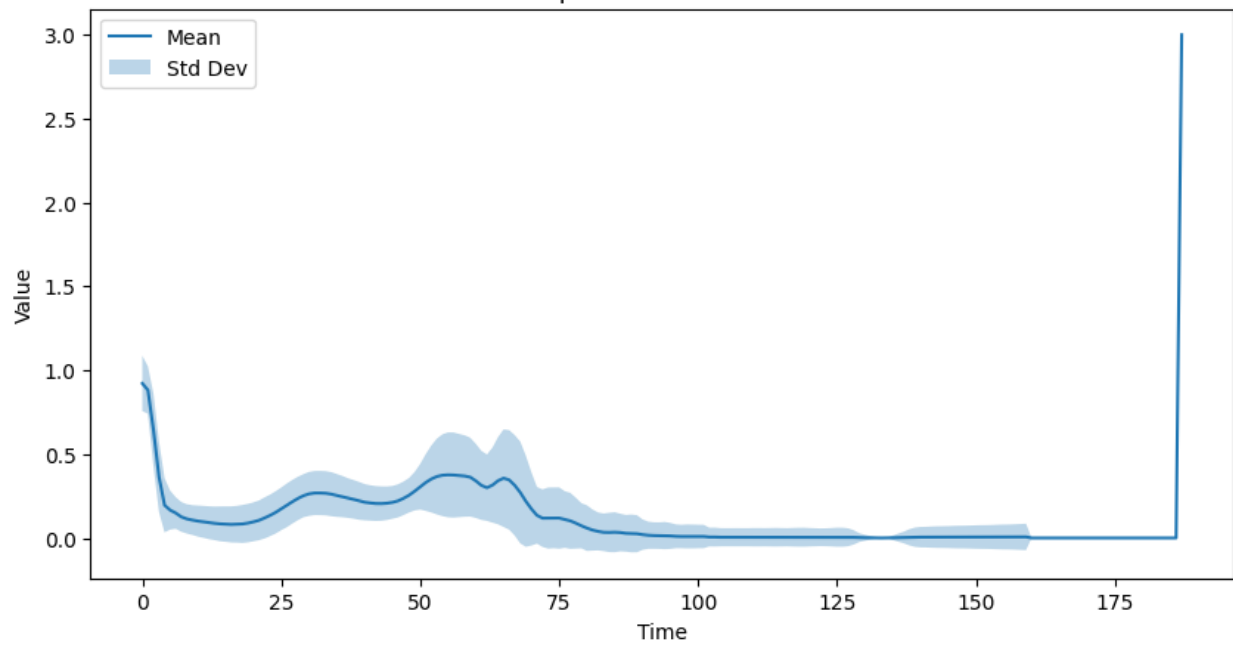
Mean and Std Dev per Unit Time for 1.0 in MIT-BIH Test

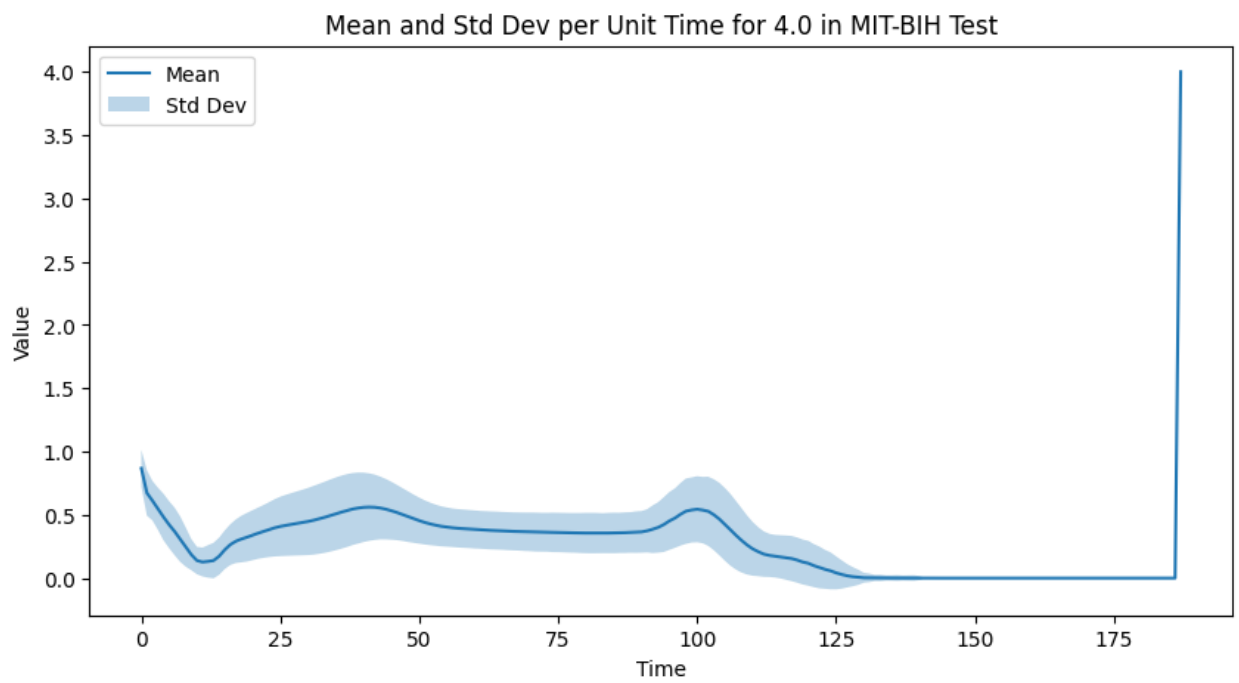


Mean and Std Dev per Unit Time for 2.0 in MIT-BIH Test



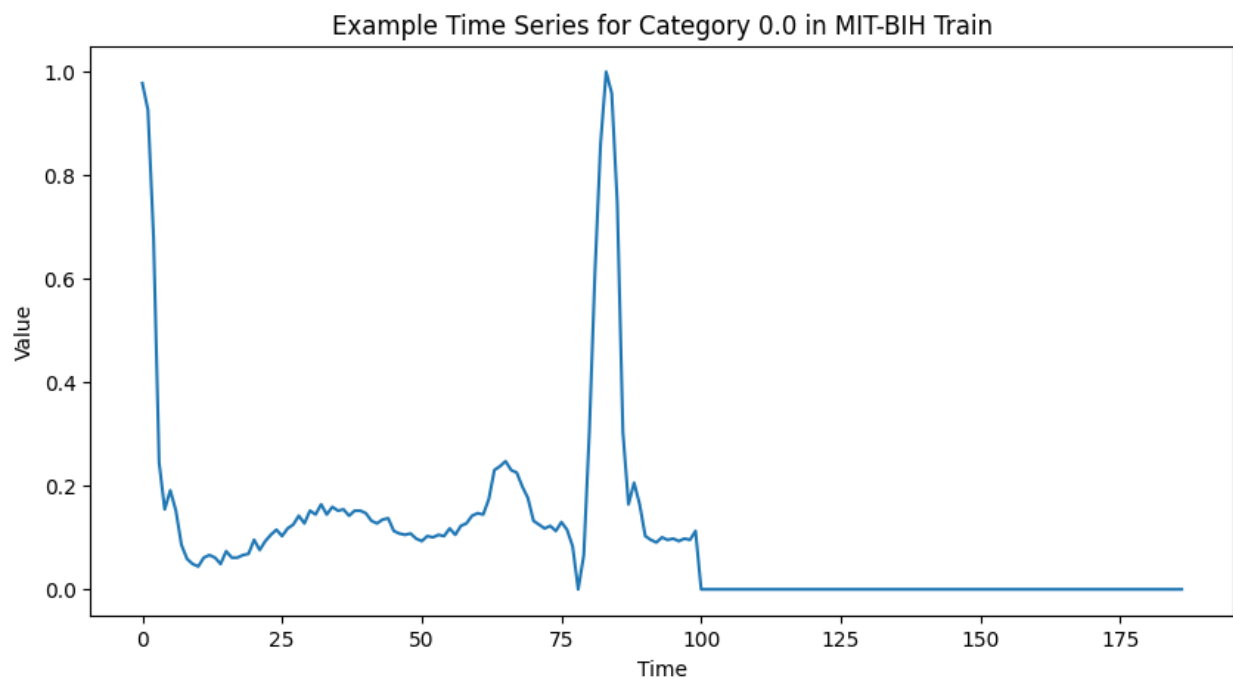
Mean and Std Dev per Unit Time for 3.0 in MIT-BIH Test



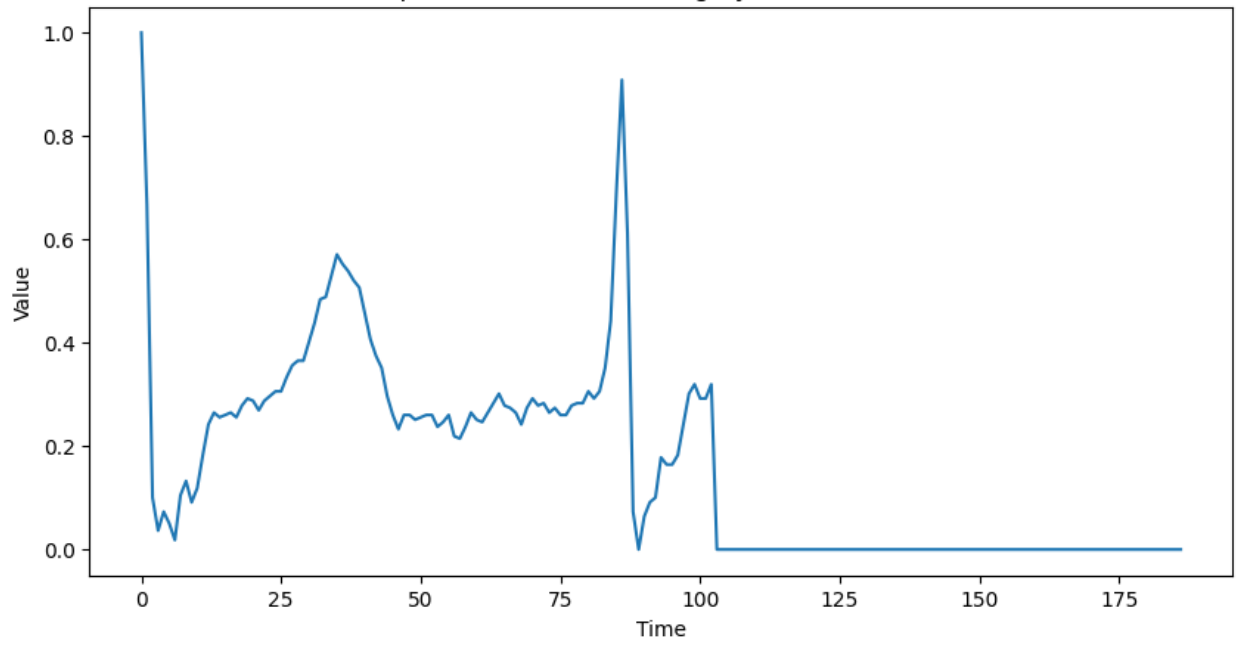


Pentru fiecare clasă unică în setul de date, funcția selectează un exemplu și creează un grafic care reprezintă semnalul ECG în funcție de timp. Aceste grafice oferă o reprezentare vizuală a formei de undă a semnalului ECG pentru fiecare clasă și set de date.

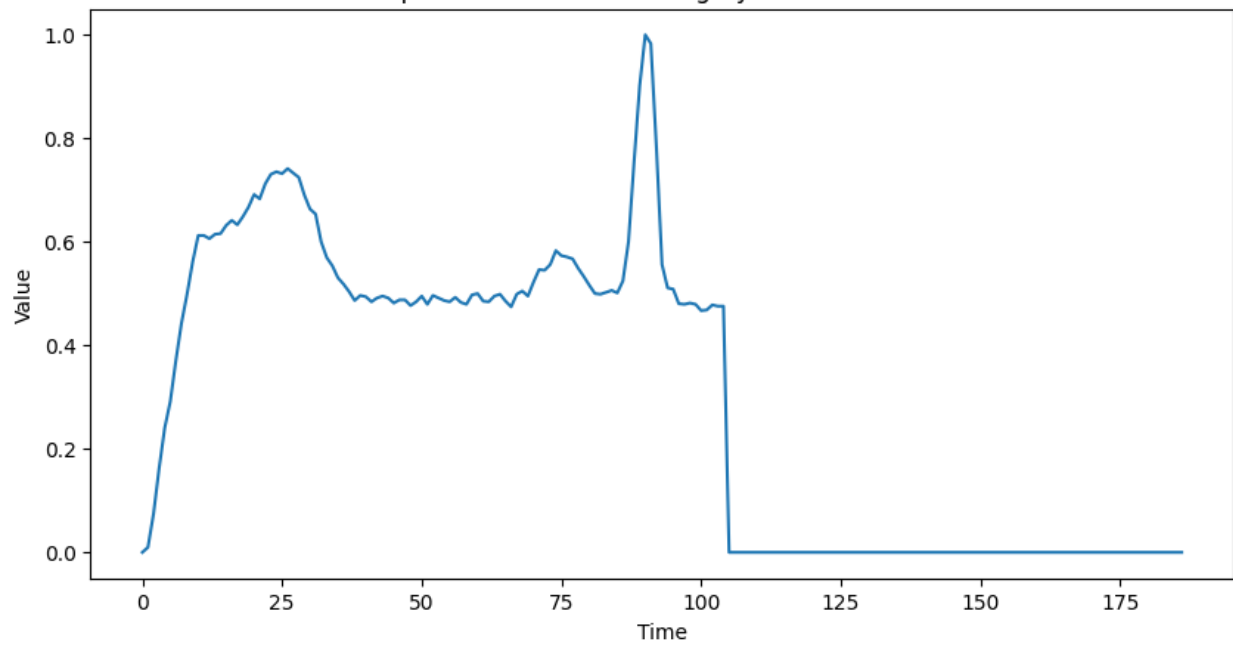
Analiza acestor grafice ajută la înțelegerea caracteristicilor semnalelor ECG pentru fiecare clasă și la identificarea diferențelor între clase. Observarea formelor de undă tipice pentru fiecare clasă și a variațiilor între exemple poate ajuta la dezvoltarea de strategii de clasificare și de extragere a caracteristicilor relevante pentru a diferenția între clase. De asemenea, aceste grafice pot oferi informații despre calitatea și coerența datelor, precum și despre eventuale probleme de prelucrare sau aliniere a semnalelor.



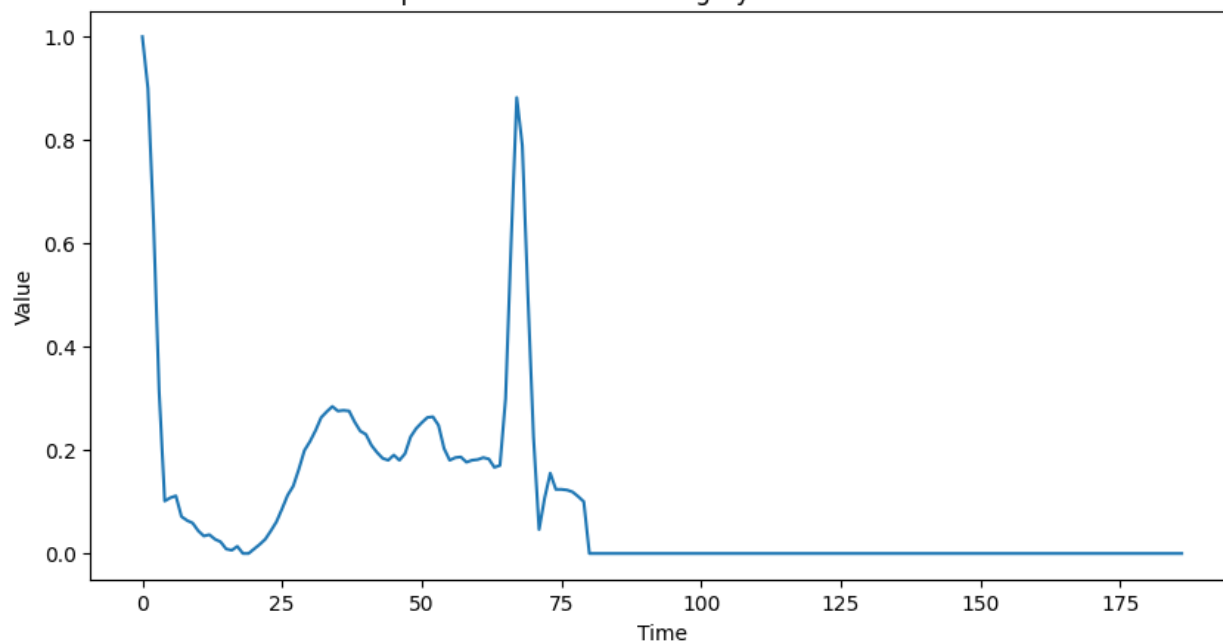
Example Time Series for Category 1.0 in MIT-BIH Train



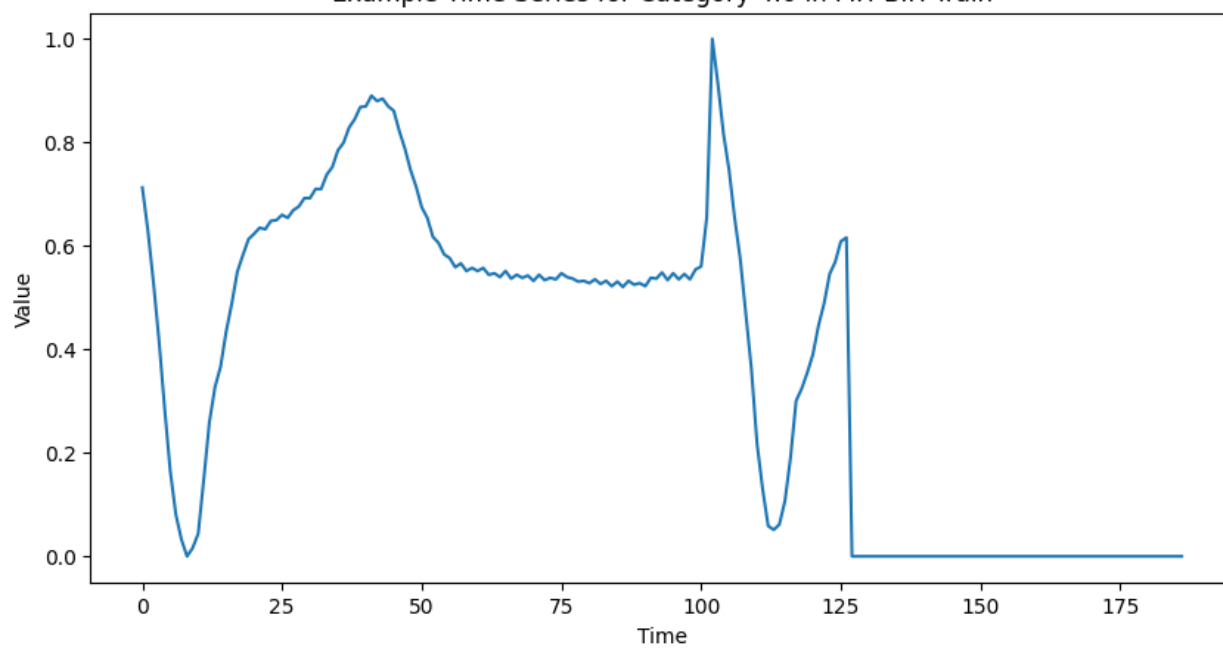
Example Time Series for Category 2.0 in MIT-BIH Train



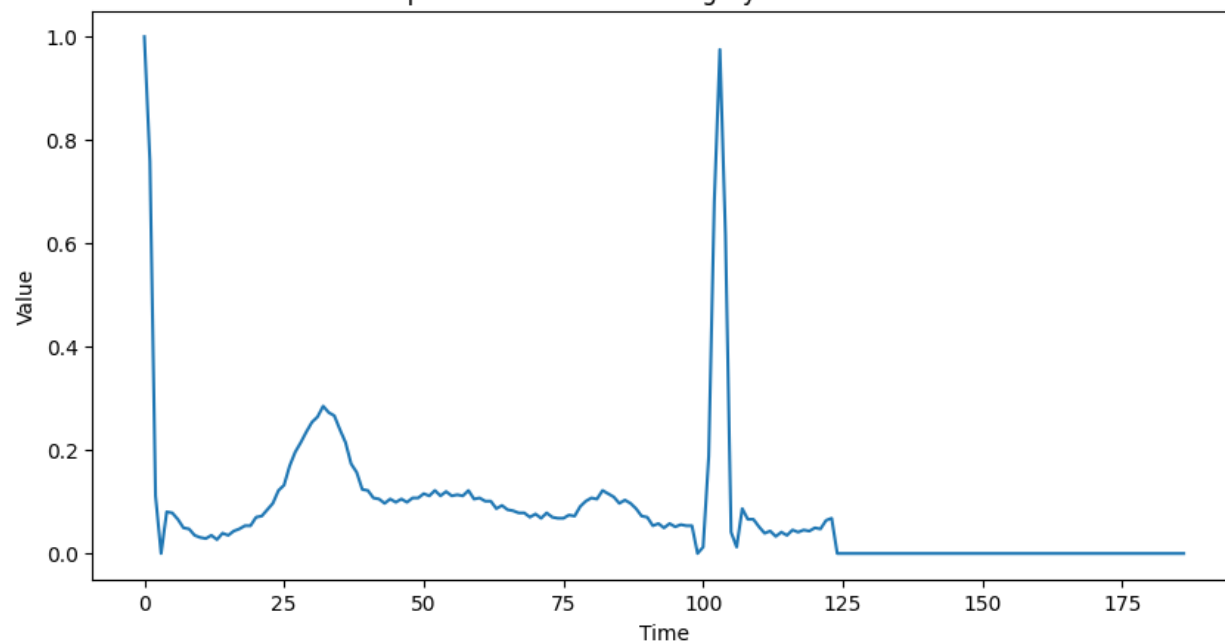
Example Time Series for Category 3.0 in MIT-BIH Train



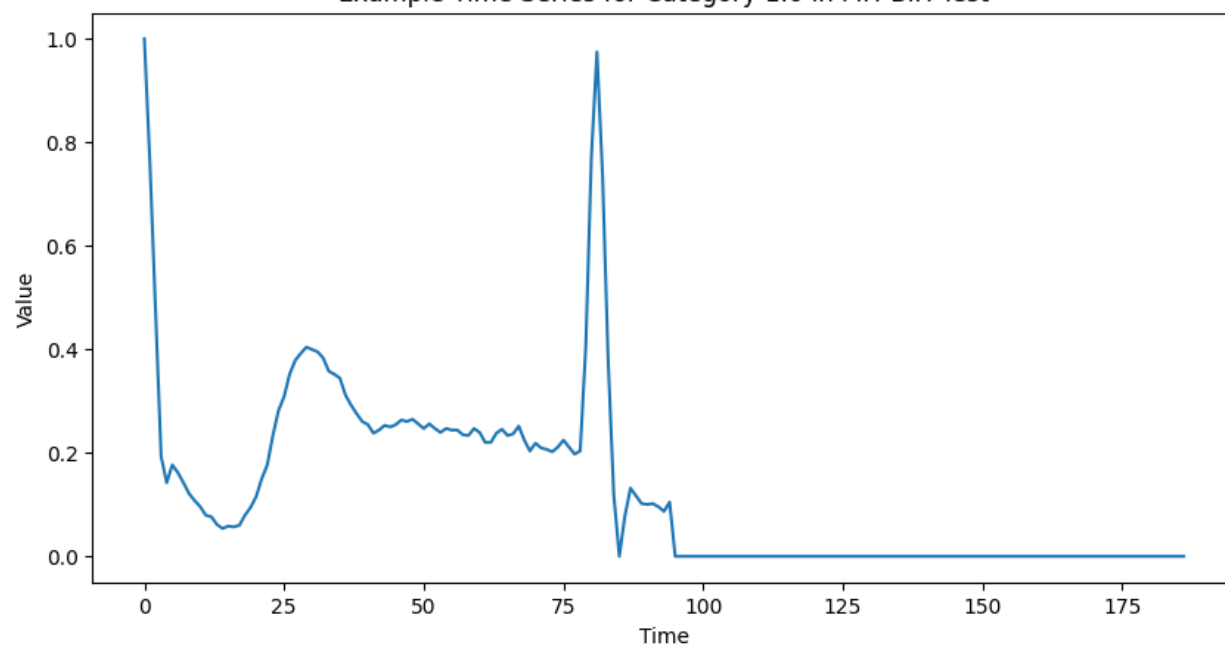
Example Time Series for Category 4.0 in MIT-BIH Train



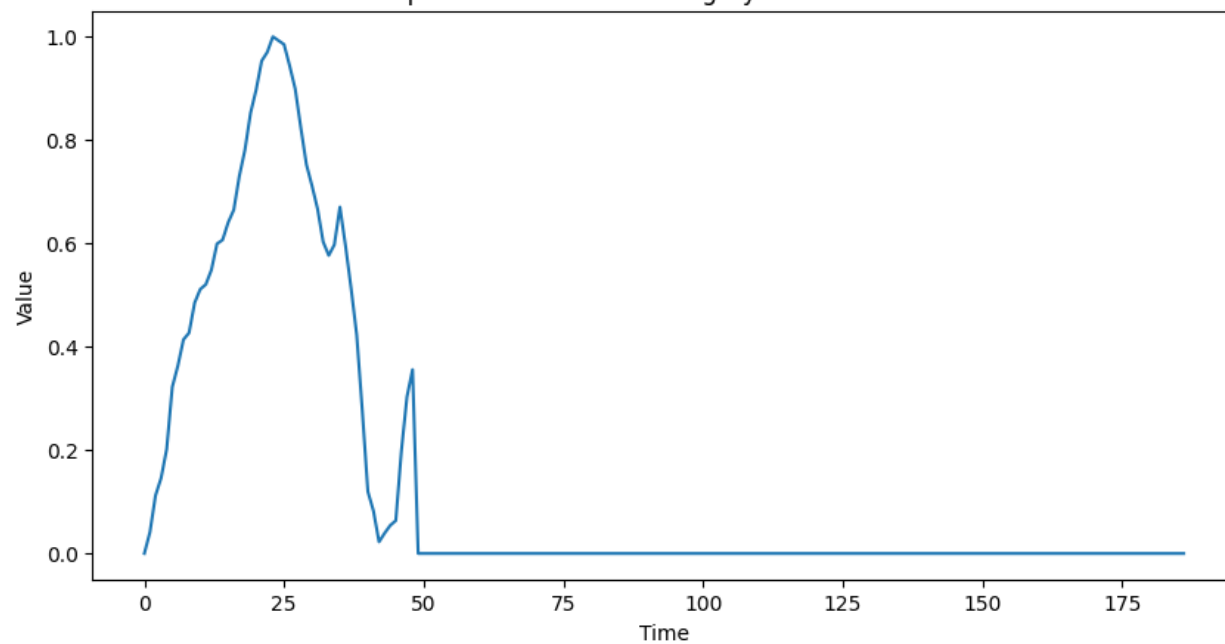
Example Time Series for Category 0.0 in MIT-BIH Test



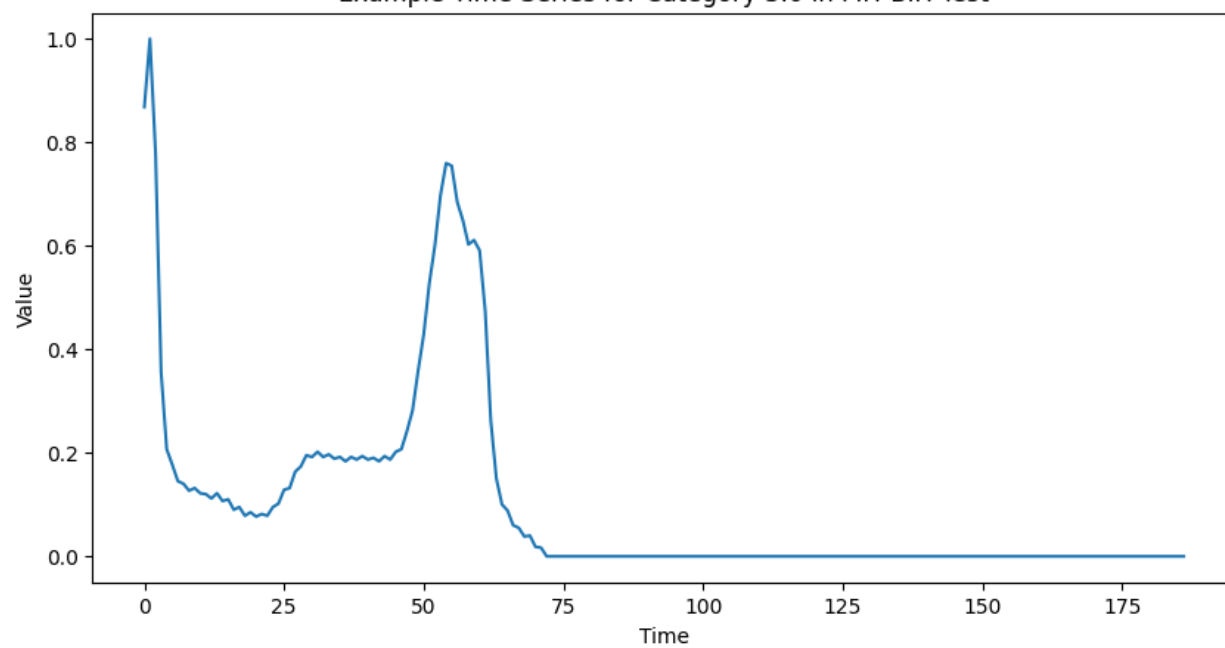
Example Time Series for Category 1.0 in MIT-BIH Test



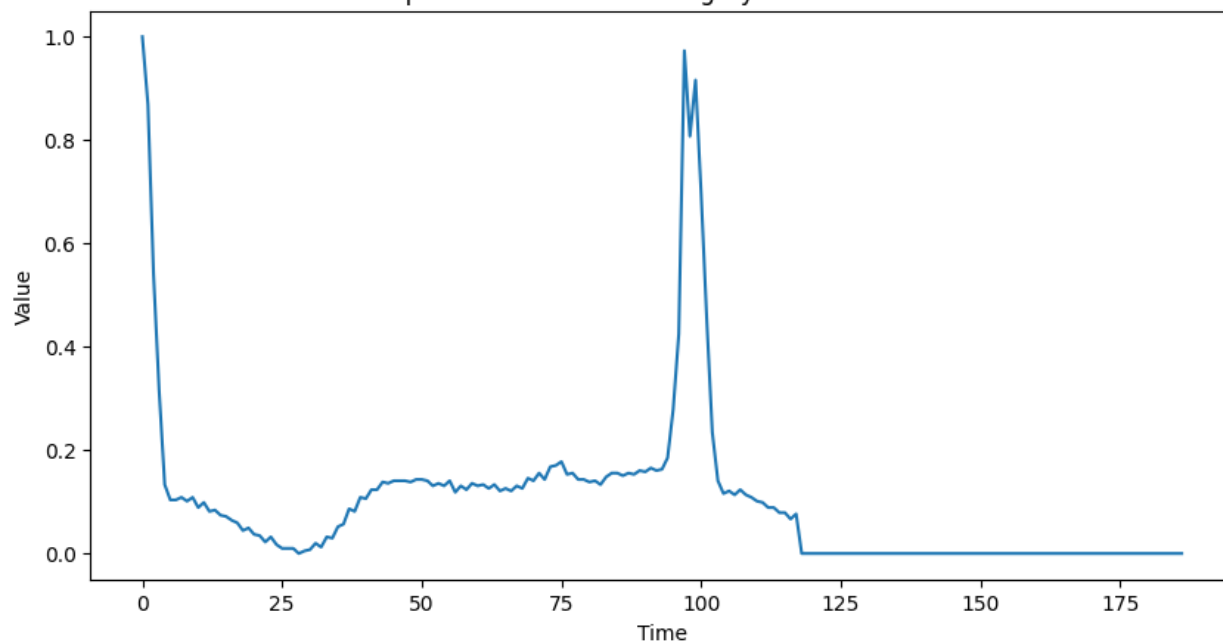
Example Time Series for Category 2.0 in MIT-BIH Test



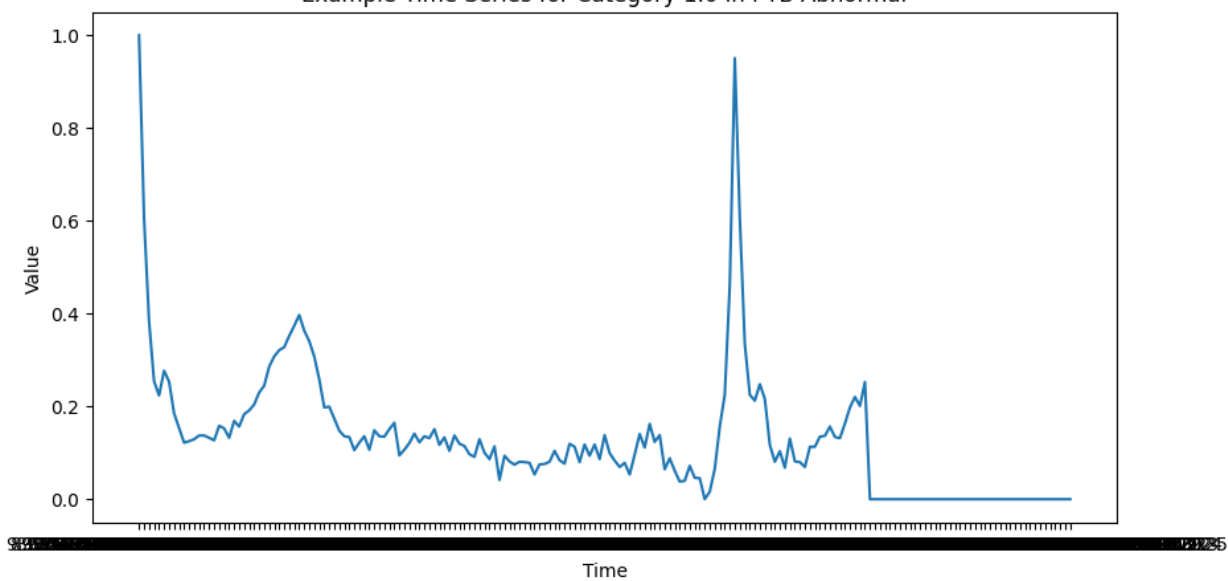
Example Time Series for Category 3.0 in MIT-BIH Test



Example Time Series for Category 4.0 in MIT-BIH Test



Example Time Series for Category 1.0 in PTB Abnormal



3.2.2

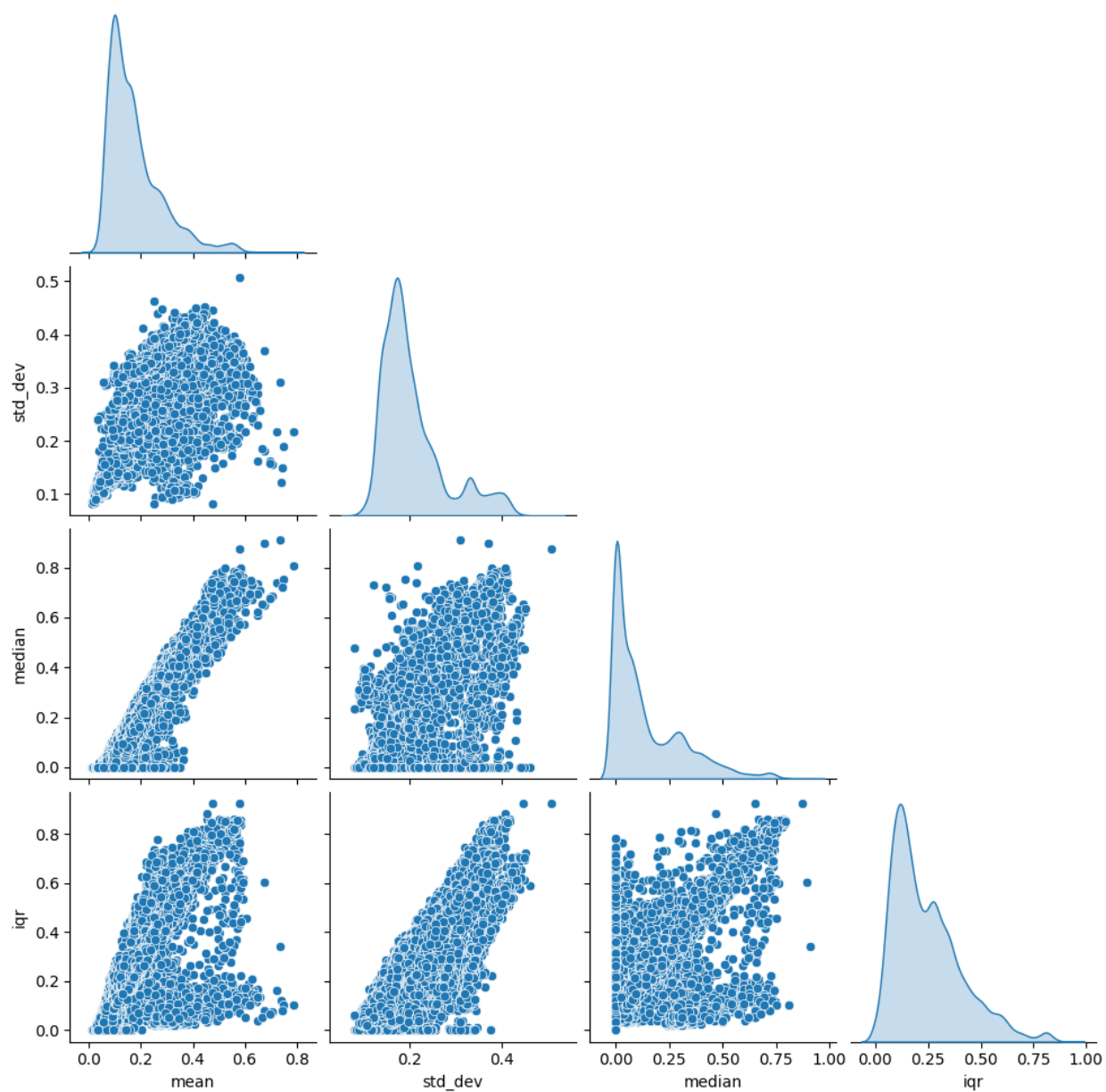
1. Definesc o funcție `extract_features` care extrage patru caracteristici (media, deviația standard, mediana și intervalul intercvartilic) pentru fiecare rând dintr-un DataFrame.
2. Definesc o funcție `sliding_window_features` care aplică o fereastră mobilă cu o lungime dată și un procent de suprapunere dat asupra unui DataFrame. Pentru fiecare fereastră, funcția extrage caracteristicile utilizând `extract_features`.
3. Definesc o funcție `plot_pairplot` care afișează un pairplot (o matrice de diagrame de dispersie) pentru caracteristicile dintr-un DataFrame.
4. Definesc o funcție `merge_with_labels` care adaugă etichetele date unui DataFrame.
5. Încarc datele din seturile de date MIT-BIH și PTB.
6. Extrag caracteristicile pentru toate seturile de date.
7. Calculez caracteristicile pe ferestrele mobile pentru toate seturile de date.
8. Combin seturile de date cu etichetele lor respective.
9. Creez pairplots pentru fiecare set de date și pentru setul de date combinat.

În cod efectuez extragerea și analiza caracteristicilor pentru seturile de date MIT-BIH și PTB.

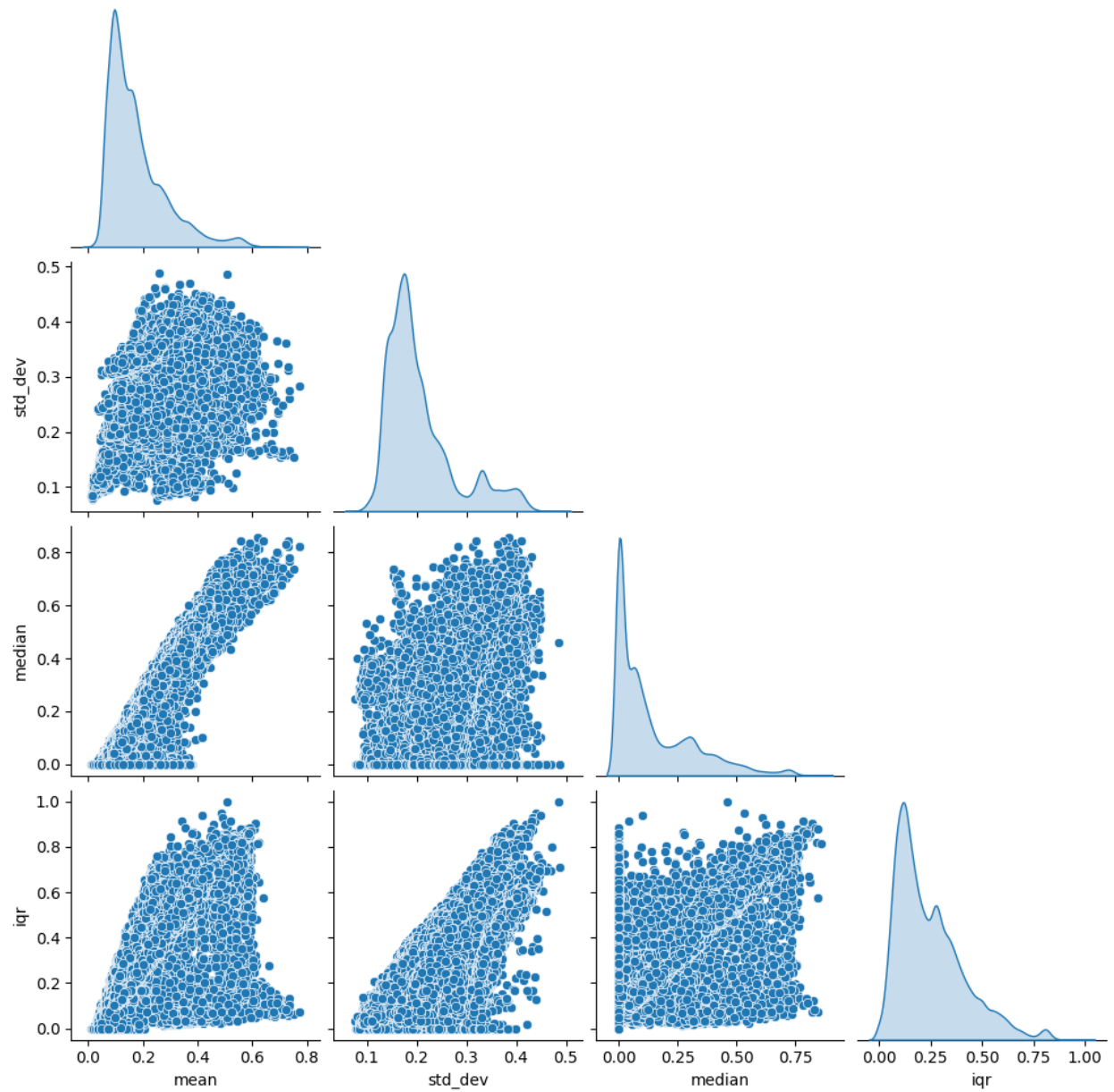
Funcțiile definite permit calcularea caracteristicilor pe ferestre mobile și afișarea pairplots pentru a vizualiza relațiile dintre caracteristicile extrase. Pairplots sunt create pentru fiecare set de date în parte, precum și pentru un set de date combinat care include etichetele pentru a permite compararea caracteristicilor între seturile de date. Acest tip de analiză poate ajuta la identificarea caracteristicilor relevante pentru clasificarea bătailor inimii și la dezvoltarea de strategii de clasificare și extragere a caracteristicilor.

De exemplu, reiese din grafice o diferență mare în IQR (Intervalul intercvartilic) și acesta ne poate ajuta la clasificarea datelor pe diferite label-uri.

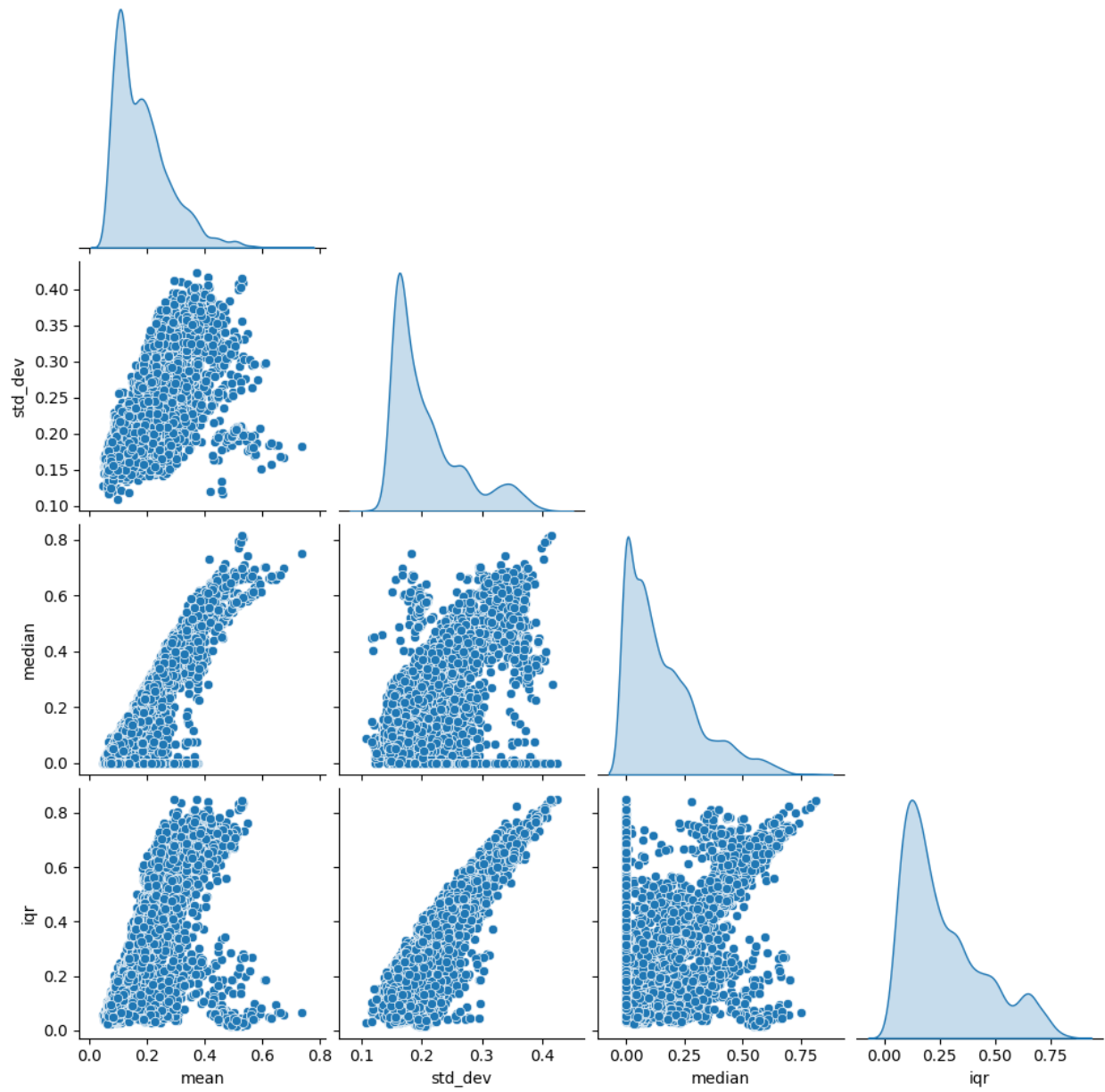
MIT-BIH Test Features



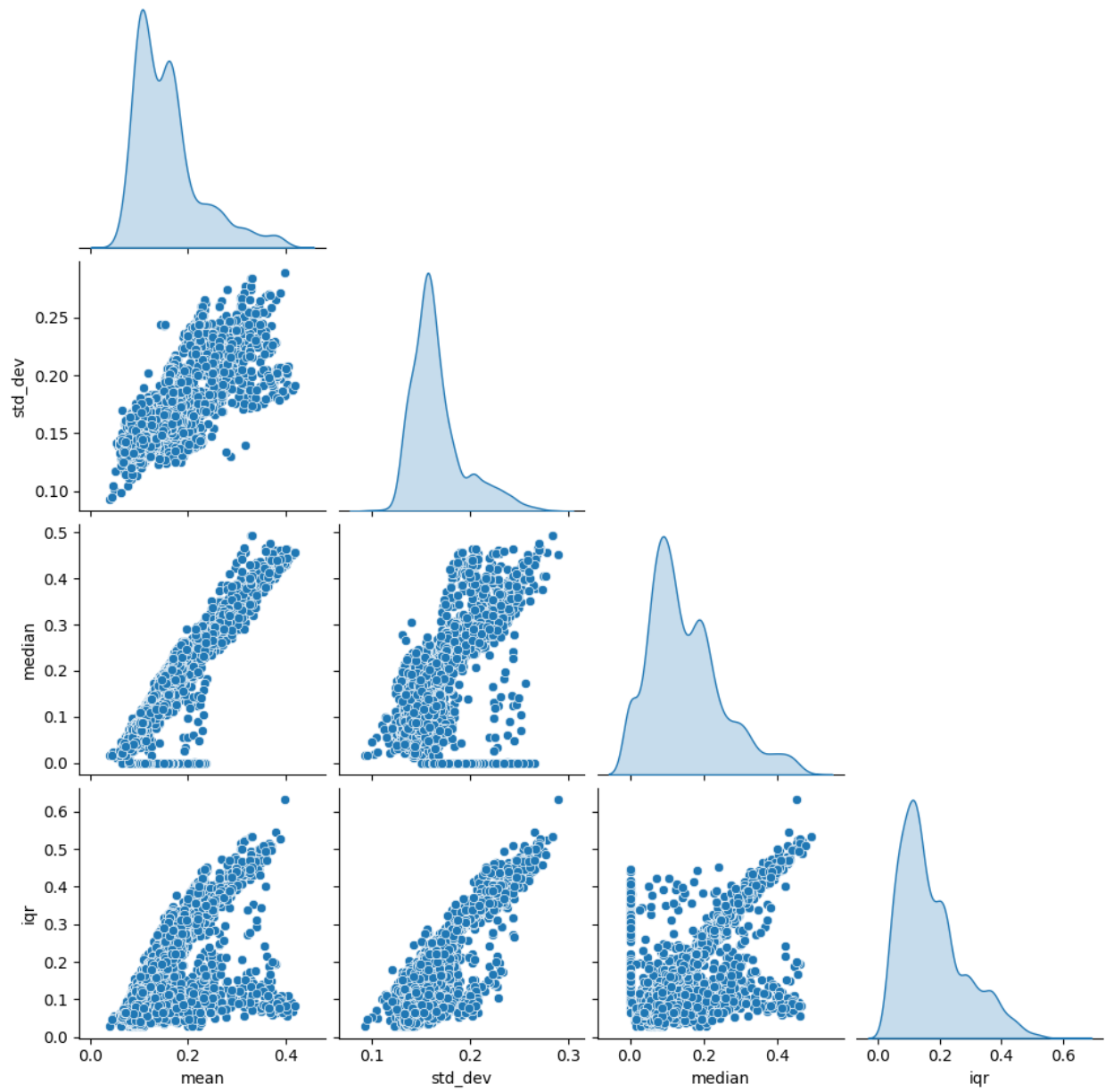
MIT-BIH Train Features



PTBDB Abnormal Features



PTBDB Normal Features



Combined Dataset Features

