AYDIN ADNAN MENDERES UNIVERSITY

ENGINEERING FACULTY

COMPUTER SCIENCE ENGINEERING DEPARTMENT



Signal Processing and Machine Learning 2024-2025

Lecturer: Ahmet Çağdaş Seçkin

Students

Abdulla Yiğit

Mehmet Ali Aşkaroğlu

Ömer Ögüt

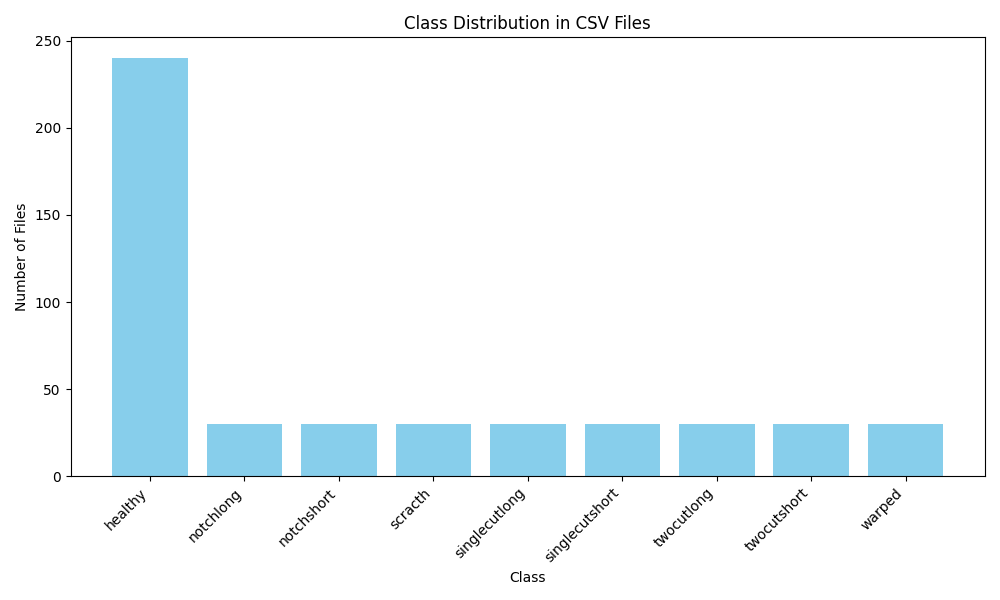
Recep Taha Ülkü

**Machine Learning Section**

**1. Dataset Overview**

**Predictive Maintanence Dataset:Input:**8 sensor data (1 sound, 3 axcelerometer, 3 gyroscope, temperature) **Output:** 9 classes  90Hz sampling,

Number of CSV files: 480



**2. Feature Extraction**

The purpose of the feature extraction code is to **extract features from sensor data** stored in CSV files and **merge them into a single dataset** for further analysis or machine learning tasks.

1. **Input**: The code reads multiple CSV files from a specified folder (input\_folder), where each file contains sensor data (e.g., speed, acceleration, gyroscope, etc.).
2. **Feature Extraction**: For each sensor column (e.g., Speed, Acceleration X, etc.), it computes the **Short-Time Fourier Transform (STFT)** to analyze the frequency content. From the STFT, it extracts statistical features like mean, variance, energy, median, max, min, skewness, kurtosis, and entropy of the magnitude spectrum.
3. **Class Label Extraction**: The class label (e.g., 'healthy' or other conditions) is extracted from the filename and added to the features.
4. **Output**: The extracted features from all files are merged into a single DataFrame and saved as a CSV file (features.csv).

ekran görüntüsü, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

(Example of extracted features)

**3. Binary Classification**

**3.1 Model Performance on Training/Validation Set**

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- | --- |
| **Random Forest** | 0.9323 | 0.9327 | 0.9323 | 0.9323 | 384 |
| **Gradient Boosting** | 0.9297 | 0.9303 | 0.9297 | 0.9297 | 384 |
| **KNN** | 0.5260 | 0.5261 | 0.5260 | 0.5260 | 384 |
| **SVM** | 0.5495 | 0.6111 | 0.5495 | 0.4769 | 384 |

**Key Observations:**

* **Random Forest** and **Gradient Boosting** performed exceptionally well, with accuracy, precision, recall, and F1-scores all above **93%**. These models are highly effective for binary classification.
* **KNN** and **SVM** performed poorly, with accuracy and F1-scores around **53%** and **55%**, respectively. SVM also showed a significant imbalance between precision and recall, indicating poor generalization.

**3.2 Model Performance on Test Set**

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **Prediction Time (seconds)** |
| --- | --- | --- | --- | --- | --- |
| **Random Forest** | 0.85 | 0.86 | 0.85 | 0.85 | 0.0050 |
| **Gradient Boosting** | 0.85 | 0.86 | 0.85 | 0.85 | 0.0020 |
| **KNN** | 0.56 | 0.56 | 0.56 | 0.56 | 0.0209 |
| **SVM** | 0.58 | 0.64 | 0.58 | 0.54 | 0.0060 |

**Key Observations:**

* **Random Forest** and **Gradient Boosting** maintained strong performance on the test set, with accuracy and F1-scores of **85%**. These models are robust and generalize well to unseen data.
* **KNN** and **SVM** continued to perform poorly on the test set, with accuracy and F1-scores below **60%**. SVM showed a significant drop in precision for class 0 (72% precision, 27% recall), indicating poor performance for one of the classes.
* **Prediction Times**: Gradient Boosting was the fastest (0.0020 seconds), while KNN was the slowest (0.0209 seconds).

**4. Multi-class Classification**

**4.1 Model Performance on Training/Validation Set**

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- | --- |
| **Random Forest** | 0.7240 | 0.7272 | 0.7240 | 0.7232 | 192 |
| **Gradient Boosting** | 0.7552 | 0.7549 | 0.7552 | 0.7530 | 192 |
| **KNN** | 0.1510 | 0.1389 | 0.1510 | 0.1431 | 192 |
| **SVM** | 0.1562 | 0.1854 | 0.1562 | 0.1454 | 192 |

**Key Observations:**

* **Gradient Boosting** outperformed **Random Forest** in multi-class classification, with an accuracy of **75.52%** compared to **72.40%**. Both models showed balanced precision, recall, and F1-scores.
* **KNN** and **SVM** performed very poorly, with accuracy and F1-scores below **16%**. These models are not suitable for this multi-class classification task.

**4.2 Model Performance on Test Set**

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **Prediction Time (seconds)** |
| --- | --- | --- | --- | --- | --- |
| **Random Forest** | 0.79 | 0.81 | 0.79 | 0.79 | 0.0060 |
| **Gradient Boosting** | 0.69 | 0.72 | 0.69 | 0.68 | 0.0030 |
| **KNN** | 0.29 | 0.30 | 0.29 | 0.27 | 0.0090 |
| **SVM** | 0.21 | 0.22 | 0.21 | 0.18 | 0.0040 |

**Key Observations:**

* **Random Forest** performed the best on the test set, with an accuracy of **79%** and balanced precision, recall, and F1-scores. It generalized well to unseen data.
* **Gradient Boosting** showed a slight drop in performance on the test set, with an accuracy of **69%**. This indicates some overfitting to the training data.
* **KNN** and **SVM** continued to perform poorly, with accuracy and F1-scores below **30%**. These models are not effective for this task.
* **Prediction Times**: Gradient Boosting was the fastest (0.0030 seconds), while KNN was the slowest (0.0090 seconds).

**5. Overall Summary**

**5.1 Binary Classification**

* **Best Model**: **Random Forest** and **Gradient Boosting** are the top performers, with high accuracy (85%) and balanced metrics on the test set.
* **Worst Model**: **KNN** and **SVM** performed poorly, with accuracy below 60%. These models are not recommended for this task.
* **Prediction Speed**: Gradient Boosting is the fastest, while KNN is the slowest.

**5.2 Multi-class Classification**

* **Best Model**: **Random Forest** is the best-performing model, with an accuracy of 79% on the test set. It generalizes well and maintains balanced metrics.
* **Worst Model**: **KNN** and **SVM** are not suitable for this task, with accuracy below 30%.
* **Prediction Speed**: Gradient Boosting is the fastest, while KNN is the slowest.

**5.3 Recommendations**

* For **binary classification**, use **Random Forest** or **Gradient Boosting** for optimal performance.
* For **multi-class classification**, use **Random Forest** for the best balance of accuracy and generalization.
* Avoid using **KNN** and **SVM** for both tasks, as they consistently underperform.

**6. Additional Insights**

* **Overfitting**: Gradient Boosting showed slight overfitting in the multi-class task, as its performance dropped from 75.52% (training) to 69% (test). Regularization or hyperparameter tuning could help.
* **Class Imbalance**: In binary classification, SVM showed a significant imbalance in precision and recall for class 0 (72% precision, 27% recall). This indicates poor handling of class imbalance.
* **Scalability**: Gradient Boosting and Random Forest are scalable and efficient, with fast prediction times. KNN is computationally expensive and not suitable for larger datasets.

**7. Confusion Matrixes**

**Binary Classification**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Açıklama otomatik olarak oluşturuldu

**(Best Model)**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, diyagram, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Multiclass Section**

metin, ekran görüntüsü, diyagram, kare içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Açıklama otomatik olarak oluşturuldu

**(Best Model)**

metin, ekran görüntüsü, diyagram, kare içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, diyagram, kare içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Maklin Predictions With Best Models**

Sample 1: Predicted: singlecutlong Actual: singlecutlong

Sample 2: Predicted: healthy Actual: healthy

Sample 3: Predicted: healthy Actual: healthy

Sample 4: Predicted: healthy Actual: healthy

Sample 5: Predicted: healthy Actual: scracth

Sample 6: Predicted: singlecutlong Actual: singlecutlong

Sample 7: Predicted: warped Actual: healthy

Sample 8: Predicted: healthy Actual: healthy

Sample 9: Predicted: healthy Actual: healthy

Sample 10: Predicted: singlecutlong Actual: singlecutlong

**Deep Learning Section**

**CNN Model (Best DL Model)**

**1. Binary Classification Model**

* **Training Performance:**
  + The model achieved a high training accuracy of 99.74% by the 15th epoch.
  + The training loss decreased significantly from 0.7377 (Epoch 1) to 0.0623 (Epoch 15), indicating effective learning.
  + Validation accuracy improved steadily, reaching 85.42% by the 16th epoch.
  + Validation loss decreased from 0.6357 (Epoch 1) to 0.3532 (Epoch 16), showing good generalization.
* **Test Performance:**
  + Accuracy: 84.38%
  + Precision: 84.51%
  + Recall: 84.38%
  + F1-Score: 84.36%
  + Test Time: 0.2309 seconds
* **Observations:**
  + The binary model performed well, with consistent improvement in both training and validation metrics.
  + The model generalizes effectively, as indicated by the close match between training and validation accuracy.

**2. Multiclass Classification Model**

* **Training Performance:**
  + Training accuracy improved from 15.62% (Epoch 1) to 90.10% (Epoch 27).
  + Training loss decreased significantly from 2.5141 (Epoch 1) to 0.2683 (Epoch 28).
  + Validation accuracy showed slower improvement, reaching 52.08% by the 25th epoch.
  + Validation loss fluctuated, indicating potential overfitting or difficulty in generalizing to the validation set.
* **Test Performance:**
  + Accuracy: 50.00%
  + Precision: 56.90%
  + Recall: 50.00%
  + F1-Score: 50.26%
  + Test Time: 0.1647 seconds
* **Observations:**
  + The multiclass model struggled to generalize, as evidenced by the lower validation and test accuracy compared to the binary model.
  + The fluctuating validation loss and accuracy suggest potential overfitting or insufficient model complexity for the task.

**Confusion Matixes and Training/Validation Loss Graphs**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, kare, dikdörtgen içeren bir resim

Açıklama otomatik olarak oluşturuldu

çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

çizgi, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Making Predictions With CNN**

Sample Index: 1

Predicted Health Status: Healthy

Real Health Status: Healthy

Sample Index: 2

Predicted Health Status: Fault

Predicted Fault Class: singlecutlong

Real Health Status: Fault

Real Fault Class: notchlong

Sample Index: 3

Predicted Health Status: Healthy

Real Health Status: Healthy

Sample Index: 4

Predicted Health Status: Healthy

Real Health Status: Healthy

**LSTM Model**

**1. Binary LSTM Model**

* **Training Duration**: 14 epochs (out of 50 planned).
* **Average Time per Epoch**: ~3 seconds.
* **Final Training Metrics**:
  + **Loss**: 0.6832
  + **Accuracy**: 0.5339
* **Final Validation Metrics**:
  + **Validation Loss**: 0.6874
  + **Validation Accuracy**: 0.5312
* **Test Metrics**:
  + **Test Time**: 0.8486 seconds
  + **Accuracy**: 0.6250
  + **Precision**: 0.6270
  + **Recall**: 0.6250
  + **F1-Score**: 0.6235

**Observations**:

* The model showed gradual improvement in training accuracy and loss over the epochs.
* Validation accuracy peaked at 0.6458 in Epoch 10 but later dropped, indicating potential overfitting.
* Test metrics suggest the model generalizes moderately well, with an accuracy of 62.5%.

**2. Multiclass LSTM Model**

* **Training Duration**: 41 epochs (out of 50 planned).
* **Average Time per Epoch**: ~1-2 seconds.
* **Final Training Metrics**:
  + **Loss**: 1.9773
  + **Accuracy**: 0.2448
* **Final Validation Metrics**:
  + **Validation Loss**: 1.7944
  + **Validation Accuracy**: 0.3333
* **Test Metrics**:
  + **Test Time**: 0.7724 seconds
  + **Accuracy**: 0.4167
  + **Precision**: 0.3542
  + **Recall**: 0.4167
  + **F1-Score**: 0.3405

**Observations**:

* The model struggled to achieve high accuracy, with training accuracy remaining below 40% throughout training.
* Validation accuracy fluctuated but peaked at 0.4375 in Epoch 37.
* Test metrics indicate poor generalization, with an accuracy of 41.67% and low precision, recall, and F1-score.

**Confusion Matixes and Training/Validation Loss Graphs**

çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

çizgi, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**metin, ekran görüntüsü, diyagram, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

metin, ekran görüntüsü, kare, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu