Team 22 [Proj C2]: Terrain Identification from Time Series Data

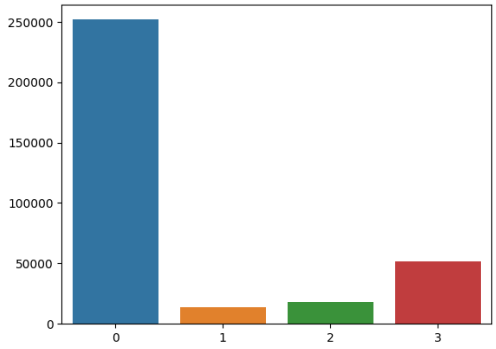
Vidit Pandya (vpandya)
Prateek Wadhwani (pwadhwa)
Saumil Shah (sashah8)

I. Methodology

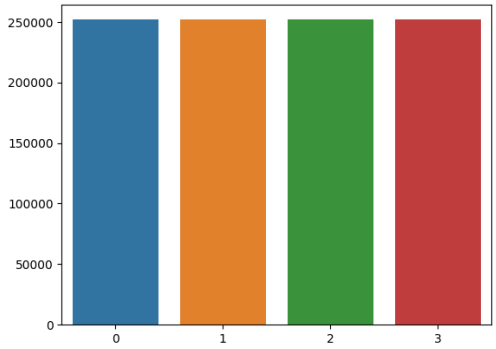
Lower limb amputations take away an individual's ability to walk, and such individuals become primarily dependent on prostheses to restore ambulation to a certain extent. Extensive research conducted for the improvement of prosthetic devices attempts to ensure that amputees can attain robust, adaptive and energy efficient walking abilities. Lower limb robotic prosthetics may benefit from contextual information of the individual's surroundings, since activities over different terrains such as walking on concrete or grass, going down or climbing up a staircase, etc. require different degrees of effort. Hence terrain identification may be used to augment the control capabilities of robotic prosthetic devices and provide enhanced safety and comfort to the user. The aim of the project is to be able to accurately identify the terrain based on the data streams from the inertial measurement unit (IMU) present on a lower limb prosthetic device.

The dataset consists of IMU data from sensors attached to the lower limbs of 6 different participants over multiple session. The IMU data consists of spatial data (xyz) from the accelerometer and the gyro sensor sampled at 40 Hz, along with the respective timestamps. The label data is provided in a separate file and is annotated with four different integer labels representing four classes of terrain along with the respective timestamps, and has a sampling rate of 10 Hz. The four terrain labels are: (0) indicating standing or walking in solid ground, (1) indicating going down the stairs, (2) indicating going up the stairs, and (3) indicating walking on grass. Based on these annotations the goal is to identify the terrain class based on the time series IMU data.

Since the provided dataset has different sampling frequencies for the spatial data and labels, the first task is to match the features with the corresponding ground truth labels. To achieve this, downsampling of feature data was done to match the timestamps of the label data. Furthermore, it was observed that there was significant imbalance between the terrain class distribution which was treated to promote generalization and reduce the chances of overfitting. Subsequently the data was split into training and validation over which multiple classical classification approaches were implemented to determine the best performing model. For model selection, RandomForest, DecisionTree, ExtraTrees, KNeighbors, GradientBoosting were compared by calculating accuracy scores, f-1 scores, and confusion matrices to identify the well performing models, and ExtraTrees was observed to achieve the highest F1 score and accuracy. To further validate the findings, stratified k-fold cross validation was performed over all the models, where again ExtraTrees had the best overall accuracy. Hence, ExtraTrees was selected as the baseline model for the scope of this project and predictions were generated for the provided test data.



(a)



(b)

Fig. 1. Treating class imbalance: (a) visual representation of class imbalance in the dataset (b) Balanced classes after treating the imbalance via SMOTE

II. Model Training and Selection

Preprocessing of the data prior to training was critical to improving the performance of the models. The preprocessed data was visually verified to ensure that the model received appropriate inputs. As the scope of the current stage of the project was to establish a logical baseline, we conducted a

thorough model selection process to ensure that the best-performing model was chosen. The following subsections provide detailed discussions of each aspect.

A. Data Processing

As previously mentioned, the dataset featured different sampling frequencies for the spatio-temporal features and their corresponding ground truth labels. To address this issue, we implemented a downsampling approach to align their frequencies. Specifically, we employed a method whereby each observation in the feature (X) dataset was matched with the label (Y) data with the closest timestamp that either preceded or followed it. To achieve this, we used the 'mergeasof' function on the time column of both datasets, setting a tolerance of 25 ms. This procedure was applied to each session file, and the resulting datasets were combined into a common dataframe for further processing.

B. Model Training

To obtain the train and test datasets, we conducted a stratified train-test split, ensuring equal representation of all classes in both datasets. We used a random_state to ensure reproducibility of the results. However, upon examining the class distribution of the training dataset, we detected a significant imbalance that was skewed towards the standing (0) class label, as illustrated in Fig. 1 (a). To ensure the generalization of predictions and avoid poor performance on minority classes, it was necessary to address this imbalance.

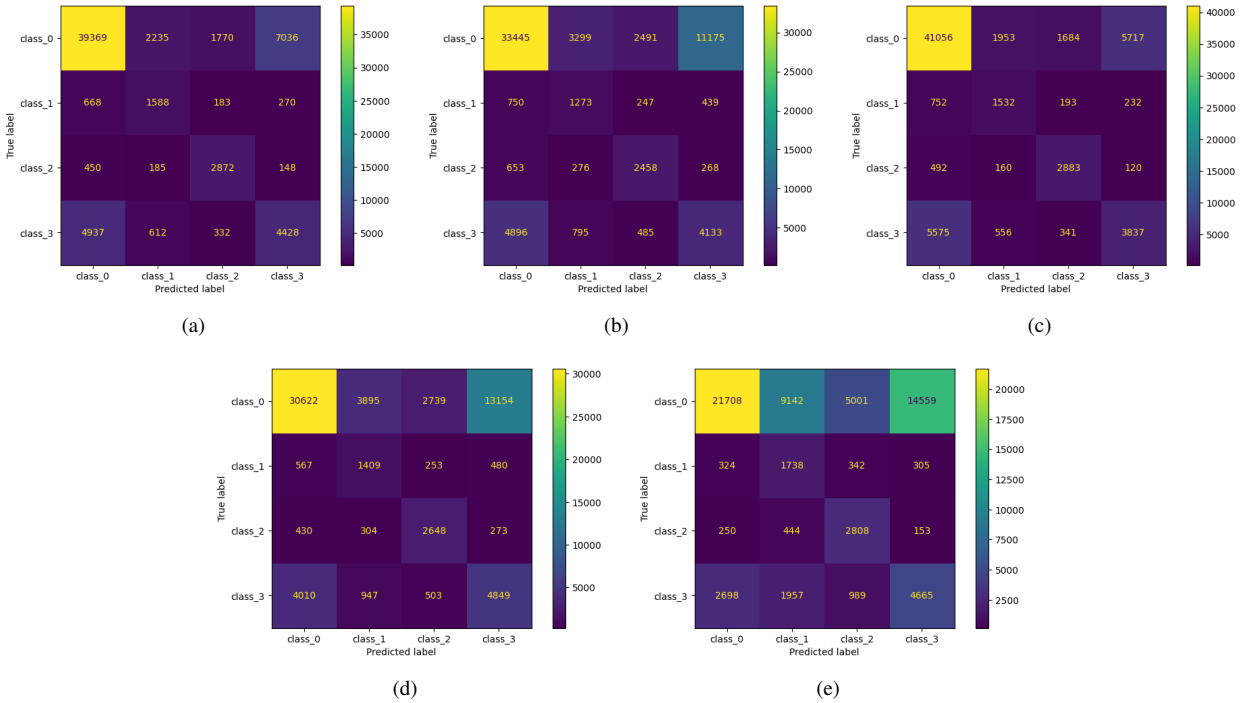
To address label imbalance in the dataset, we employ Synthetic Minority Oversampling Technique (SMOTE) [1] to oversample the minority class labels, thereby ensuring that all classes are equally represented in the dataset. SMOTE operates by randomly selecting a point from the minority class, choosing one of its k (typically 5) nearest neighbors, and generating a synthetic example that lies between the chosen point and its neighbor in the data feature space. This process can be repeated to produce as many synthetic data points as needed, and serves as a data augmentation technique that helps to prevent overfitting in the model. Consequently, each of the minority label classes is oversampled to attain a record count of 251,733, matching the record count of the standing (0) class label as illustrated in Figure 2 (b). Moreover, we do not apply the oversampling technique to the validation dataset to ensure that the accuracy of the validation dataset better reflects the results that would be obtained from expected inputs.

For the model training, we use model functions from the sklearn library and use their respective fit functions to train the model.

C. Model Selection

During the model selection process, we calculated the accuracy and macro f-1 scores for five different models: RandomForest, DecisionTree, ExtraTrees, KNeighbors, and GradientBoosting. These scores are presented in Figure 2, and their respective confusion matrices are shown in Figures 3(a)-(e). Based on the results, we observed that the ExtraTrees model [2] performed the best with the given validation dataset. It should be noted that the number of trees in the forest for ExtraTrees was not changed from the default value of 100.

Thereafter, we also performed stratified k-fold cross validation to ensure that there does not exist any bias in the results of the generated training and validation datasets and continued



(a) (b) (c)

(d) (e)

Fig. 2. Confusion matrices for the different models: (a) Random Forest (b) Decision Tree (c) Extra Trees (d) K-Nearest Neighbor (e) Gradient Boosting

to observe ExtraTrees as the best performing model with the highest accuracy scores.

We also conducted a stratified k-fold cross-validation to ensure that the results from the generated training and validation datasets were not biased. This process confirmed that the ExtraTrees model consistently performed the best, with the highest accuracy scores. The evaluations for the discussed methods have been presented in the next section.

III. Evaluation

As mentioned previously, the models were first evaluated over a train-validation split where SMOTE has been implemented to balance the classes in the training set while the classes have not been balanced for the validation set. Table I shows the model evaluations over the metrics: accuracy, macro f1 score, precision and recall. Fig. 2 shows the confusion matrices for each model for the predictions from this test.

TABLE I
Model Metrics for train-validation split

Model	Accuracy	Macro F1	Precision	Recall
Random Forest	0.719	0.576	0.535	0.645
Decision Tree	0.615	0.471	0.439	0.551
Extra Trees	0.735	0.579	0.543	0.0.635
K Nearest	0.589	0.472	0.441	0.580
Gradient Boosting	0.719	0.576	0.535	0.645

Secondly, we used k-fold cross validation as a more robust approach to validate the model's competence. As illustrated in Table II, for all the models average accuracy and macro f1 scores were checked for 10 folds over two conditions: Firstly the scores were reported without balancing of classes for the entire dataset, secondly the scores were reported by balancing the entire dataset. It is important to note that both of the above mentioned tests were repeated by balancing the dataset using undersampling instead of oversampling (SMOTE). The reported metrics for these tests provided similar scores when compared to SMOTE class balancing.

TABLE II
Average Metrics for k-fold Cross-Validation

	Unbalanced		Balanced	
Model	**Accuracy**	**Macro F1**	**Accuracy**	**Macro F1**
Random Forest	0.792	0.531	0.902	0.900
Decision Tree	0.688	0.479	0.818	0.816
Extra Trees	0.789	0.504	0.914	0.913
K Nearest	0.769	0.516	0.882	0.875
Gradient Boosting	0.764	0.346	0.593	0.586

After selecting the ExtraTrees classifier as the baseline from the results shown in Table I and II,the accuracy, macro f1, precision and recall were observed for each class for the baseline when trained over the train-validation split. It was observed that the model performs very well for class (0): standing or walking on solid ground, but performs relatively poorly for the rest of the classes. This trend can be inferred from Fig. 2 (c) as well.

TABLE III
Class-wise metrics for ExtraTrees over Train-Validation Split

Class	Macro F-1	Precision	Recall
0	0.835	0.535	0.645
1	0.443	0.439	0.551
2	0.658	0.543	0.635
3	0.379	0.441	0.580

Lastly, the predictions for the provided test data were generated via the trained model. The class distributions from the generated predictions for each of the test subjects have been presented in Fig. 3 (a)-(d).

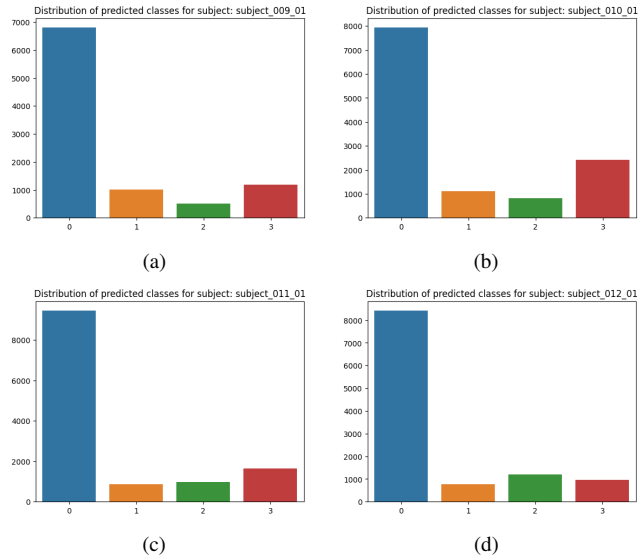


Fig. 3. Class distributions obtained from the predictions over the provided test data for (a) subject_009_01 (b) subject_010_01 (c) subject_011_01 (d) subject_012_01

Note: For selection of the model, initially the calculation of weighted F1 score was done along with model accuracy. It was observed that both the metrics show a similar trend where Extratrees outperform all other approaches. However, it was later discovered that for model testing, Macro F1 score had been used by instructor. Hence, from now on authors shall use Macro F1 scores for model evaluations.

References

[1] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. J. Artif. Int. Res. 16, 1 (January 2002), 321–357.

[2] Geurts, P., Ernst, D. Wehenkel, L. Extremely randomized trees. Mach Learn 63, 3–42 (2006). https://doi.org/10.1007/s10994-006-6226-1