Gesture AI Project Report

Rodrigo Almeida

March 15, 2024

Contents

Chapter 1

Introduction

Chapter 2

Methodology

2.1 Data Pre-processing

One of the biggest challenges in machine learning is the quality of the data. The quality of the data is crucial to the performance of the model. The data pre-processing phase is a critical step in the machine learning pipeline. It involves cleaning, transforming, and preparing the data for the model.

Imbalanced Dataset

On the dataset, the number of instances for each class is not equal. The imbalanced dataset can lead to poor performance of the model.

Below are the number of instances for each class in the dataset:

- **Driving Style:**
  - EvenPaceStyle: 21,016 instances
  - AggressiveStyle: 2,759 instances
- **Road Surface Condition:**
  - SmoothCondition: 14,237 instances
  - UnevenCondition: 6,289 instances
  - FullOfHolesCondition: 3,249 instances
- **Traffic Condition:**
  - LowCongestionCondition: 17,764 instances
  - HighCongestionCondition: 3,017 instances
  - NormalCongestionCondition: 2,994 instances

For this project **Driving Style** class column was chosen due to its significant imbalance and potential for improvement. The **AggressiveStyle** class has 2,759 instances while the **EvenPaceStyle** class has 21,016 instances. The dataset is imbalanced, and the model may be biased towards the majority class. To resolve the issue of imbalanced dataset that was observed, the use of other techniques such as over-sampling and under-sampling was considered. SMOTE (Syntetic Minority Over-sampling) which is a statistical technique for increasing the number of instances in the minority class was used. SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line. It does have some limitations, such as the generation of noisy samples, but it is a widely used technique for dealing with imbalanced datasets. [1]. Below is how the **Driving Style** dataset looks like after applying SMOTE.

- **Driving Style:**
  - EvenPaceStyle: 16,808 instances
  - AggressiveStyle: 16,808 instances

With the dataset now balanced, the next step was to remove the **Unnamed: 0** column. This column was not needed for the analysis and was removed from the dataset. After having that completed it was found that there were missing values in the dataset as shown in Table 2.1.

Column	Missing Values
VehicleSpeedInstantaneous	9
EngineLoad	5
EngineCoolantTemperature	5
ManifoldAbsolutePressure	5
EngineRPM	5
MassAirFlow	5
IntakeAirTemperature	5
FuelConsumptionAverage	5

Table 2.1: Missing Values in the Dataset

Due to the small number of missing values relative to the size of the dataset, imputation seems to be a pratical approach. For the numerical columns with missing values, it can be filled with the mean or median of the column. For the categorical columns, the missing values can be filled with the mean of the column. In this case was chosen to fill the missing values with the mean of the column, as it data is normally distributed.

2.2 Data Labelling

For the categorical variables (roadSurface, traffic, drivingStyle), needs to convert these into a format that can be used for machine learning models. Since **drivingStyle** is the target variable, the focus will be on **roadSurface** and **traffic**. The continuos features, on columns such as **AltitudeVariation**, **Vehicle-SpeedInstantaneous**, **VehicleSpeedVariation**, are now standardized centered around 0(mean) and 1(standard deviation). The target variable **drivingStyle** was then encoded using the **LabelEncoder** from the **sklearn.preprocessing** module, and is now represented as 0 and 1. The categorical variables **roadSurface** and **traffic** are encoded using the **OneHotEncoder** from the **sklearn.preprocessing** module. Each column represents the presence (True/1) or absence (False/0) of a category.

2.3 Data Scaling

Feature scaling is a method used to standardize the range of independent variables or features of the data. In data processing, it is also known as data normalization and is generally performed during the data pre-processing step.

The **Standard Scaler** was used to scale the data. The Standard Scaler standardizes the features by removing the mean and scaling to unit variance. This results in a distribution with a standard deviation of 1 and a mean of 0. The formula for standard scaling is given by:

$$z = \frac{x - \mu}{\sigma} \tag{2.1}$$

Where x is the original feature value, μ is the mean of the feature, and σ is the standard deviation of the feature. This process of subtracting the mean and dividing by the standard deviation ensures that all features contribute equally to the result, a critical consideration for algorithms like Support Vector Machines (SVM), k-Nearest Neighbors (kNN), and Logistic Regression used in this project.

2.4 Data Analysis and Visualisation

As part of the analysis aiming to classify the driving styles, the data was visualized to understand the distribution of the features and the relationship between the features and the target variable. It was used the PCA (Principal Component Analysis) to visualize the data in 2D. The PCA is a dimensionality reduction technique that is used to reduce the dimensionality of the data to 2 or 3 dimensions so that it can be visualized. [2]

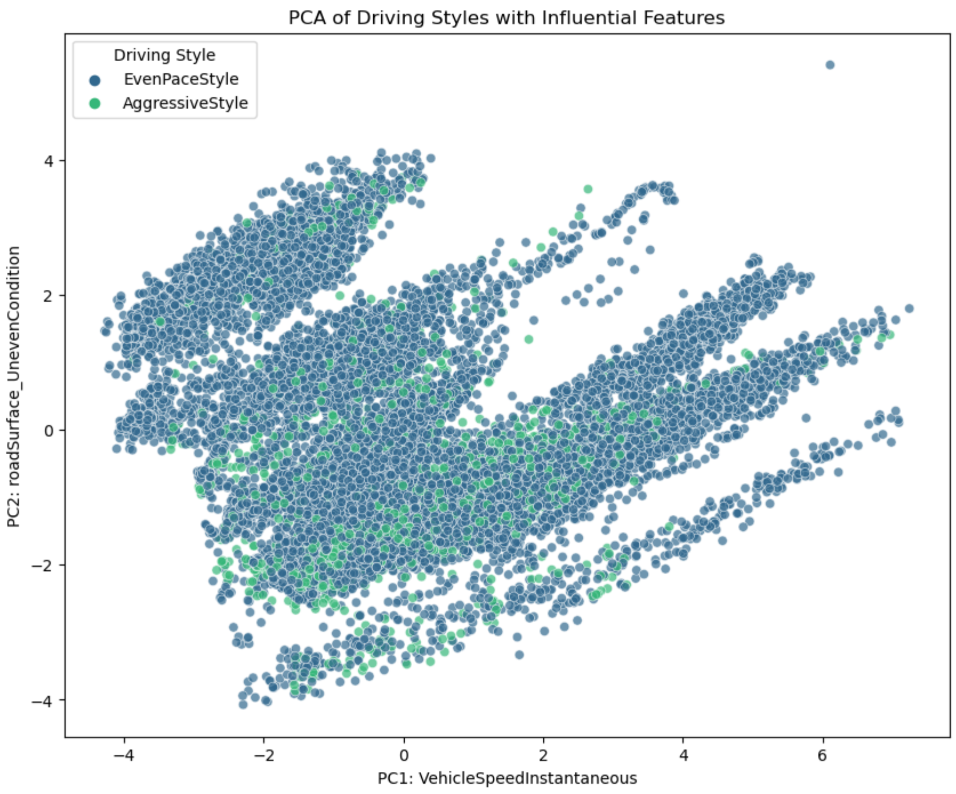

Figure 2.1: PCA Visualization of the Data

Figure 2.1 shows the PCA visualization of the data. The data is not linearly separable, and the classes are not well separated. This suggests that the data is not linearly separable and that a linear model may not be the best choice for this dataset. The axes are labelled as PC1: VehicleSpeedInstantaneous and PC2: RoadSurface_UnevenCondition, which implies that these two features contributes most to the variance in the data. VehicleSpeedInstantaneous might be defining characteristic of the driving style, while RoadSurface_UnevenCondition might also influence the driving style. Despite some level of separation, there is also a significant overlap between the classes, which indicates that not all driving style characteristics are captured by these two features, or there may be a wide range of behaviours within each style that could cause this overlap. The presence of clusters in the data suggests potential for classification models to predict the driving style based on the features.

Chapter 3

Experiments and Results

3.1 Classifier Models

3.2 Support Vector Machine Model

For the spport vector machine model, the kernel was fixed to linear considering the nature of the dataset, which suggests linear separability of driving styles. The focus was on the tuning of the hyperparameters, such as the regularization parameter C, which controls the trade off between achienving a low training error and a low testing error. A range of C values were tested to find the optimal value. The model was trained using the **GridSearchCV** method from the **sklearn.model_selection** module.

The best C parameter was identified through cross-validation, optimizing for model accuracy. The best parameters and corresponding score obtained from the grid search were accessed via `svm_grid.best_params_` and `svm_grid.best_score_`, respectively.

Before proceding to evaluate the model with the balanced and pre-processed dataset, the model was trained using the original dataset to see how it would perform. The accuracy of the model was evaluated returning a score of 0.88. But then, when running the classification report, it was found that the model was biased towards the majority class as shown in Table 3.1. It is shown that the model was able to predict the majority class with a precision of 0.88 and a recall of 1.00, but the minority class was not predicted at all. This is a clear indication of the model bias towards the majority class, this is due to the imbalanced nature of the dataset.

Class	**Precision**	**Recall**	**F1-score**	**Support**
AggressiveStyle	0.00	0.00	0.00	538
EvenPaceStyle	0.89	1.00	0.94	4217
Accuracy		0.89		
Macro Avg	0.44	0.50	0.47	4755
Weighted Avg	0.79	0.89	0.83	4755

Table 3.1: Classification Report

Confusion Matrix		
Actual/Predicted	**AggressiveStyle**	**EvenPaceStyle**
AggressiveStyle	0	538
EvenPaceStyle	0	4217

Table 3.2: Confusion Matrix

To resolve the issue of imbalanced dataset that was observed in the SVM model, the use of other techniques such as over-sampling and under-sampling was considered. SMOTE (Syntetic Minority Over-sampling) which is a statistical technique for increasing the number of instances in the minority class was used. SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line. It does have some limitations, such as the generation of noisy samples, but it is a widely used technique for dealing with imbalanced datasets. [1] Table 3.3 shows that the application of SMOTE improved the models ability to predict the minority class, which was not predicted at all in the previous model. There seems to be be a trade-off; as recall increased, precision decreased. This is expected as the model is now predicting more instances of the minority class. The F1-score is more balanced, as it is now giving equal weight to both classes. This suggests that while accuracy is lower, the model is now more fair and balanced in its predictions.

Class	**Precision**	**Recall**	**F1-score**	**Support**
AggressiveStyle	0.23	0.79	0.36	538
EvenPaceStyle	0.96	0.67	0.79	4217
Accuracy	0.68			
Macro Avg	0.60	0.73	0.58	4755
Weighted Avg	0.88	0.68	0.74	4755

Table 3.3: Classification Report

Confusion Matrix		
Actual/Predicted	**AggressiveStyle**	**EvenPaceStyle**
AggressiveStyle	427	111
EvenPaceStyle	1398	2819

Table 3.4: Confusion Matrix

3.3 Linear Regression Model

Now with the knowledge about the imbalanced dataset, the new dataset processed with SMOTE was used to train the linear regression model. To find the best parameters for the model, the **GridSearchCV** method was used to find the best parameters for the model. The results from **GridSearchCV** indicated that the best parameter for **C** was 1000, which was used to train the model. The model was then evaluated using the balanced dataset and the results are shown in Table 3.5 and Table 3.6.

Class	**Precision**	**Recall**	**F1-score**	**Support**
AggressiveStyle	0.24	0.75	0.37	538
EvenPaceStyle	0.96	0.70	0.81	4217
Accuracy	0.71			
Macro Avg	0.60	0.73	0.59	4755
Weighted Avg	0.88	0.71	0.76	4755

Table 3.5: Classification Report for Logistic Regression

Confusion Matrix for Logistic Regression		
Actual/Predicted	**AggressiveStyle**	**EvenPaceStyle**
AggressiveStyle	404	134
EvenPaceStyle	1268	2949

Table 3.6: Confusion Matrix for Logistic Regression

After optimizing the model parameters and evaluating its predictive performance, it is important to understand which features contribute most significantly to the model's predictions. In logistic regression, the coefficients associated with each feature can indicate the importance and influence of that feature on the model's output.

To visualize the impact of each feature, a coefficient plot can be constructed. This plot ranks the features by their coefficients, providing clear insights into which features are most informative for predicting the target variable.

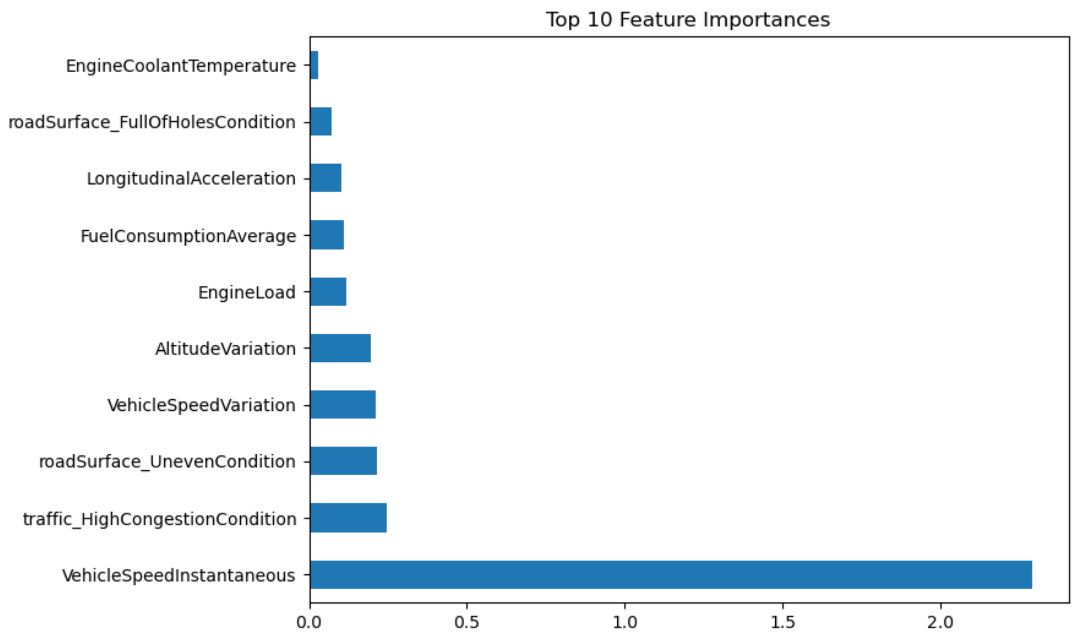

Figure 3.1: Top 10 Feature Importances in the Logistic Regression Model

This coefficient plot shows the relative importance of features in predicting driving styles. Features with larger absolute values of coefficients have a greater impact on the model's decision-making process. It is a good way to understand which features are most important for the model's predictions. The top 10 features are shown in Figure 3.1.

3.4 K-Nearest Neighbors Model

The k-Nearest Neighbors (kNN) model was trained using the balanced dataset and the results are shown in Table 3.7 and Table 3.8. The kNN model was able to predict the minority class with a precision of 0.59 and a recall of 0.81. The model was able to predict the majority class with a precision of 0.97 and a recall of 0.93. The accuracy of the model was 0.92, which is higher than the previous models. The F1-score is also higher than the previous models, suggesting that the model is more balanced in its predictions.

Class	**Precision**	**Recall**	**F1-score**	**Support**
AggressiveStyle	0.59	0.81	0.68	538
EvenPaceStyle	0.97	0.93	0.95	4217
Accuracy	0.92			
Macro Avg	0.78	0.87	0.82	4755
Weighted Avg	0.93	0.92	0.92	4755

Table 3.7: Classification Report for k-Nearest Neighbors (kNN)

Confusion Matrix for k-Nearest Neighbors (kNN)		
Actual/Predicted	**AggressiveStyle**	**EvenPaceStyle**
AggressiveStyle	436	102
EvenPaceStyle	299	3918

Table 3.8: Confusion Matrix for k-Nearest Neighbors (kNN)

3.5 Results

The Support Vector Machine (SVM) model, initially trained on the imbalanced dataset, improved after the dataset was balanced using SMOTE, achieving 0.88 accuracy. Linear Regression, opmized with GridSearchCV, achieved 0.71 accuracy with feature importance analysis as shown in Figure 3.1. The k-Nearest Neighbors (kNN) model outperformed, achieving 0.92 accuracy, effectively predicting both classes. Results demonstrate that the kNN model is the best model for this dataset, with the highest accuracy and F1-score. Linear regression and SVM models are not as effective in predicting the minority class.

Chapter 4

Conclusion

Bibliography

[1] Alberto Fernández et al. "SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary". In: *Journal of artificial intelligence research* 61 (2018), pp. 863–905.

[2] Tomasz Niedoba. "Multi-parameter data visualization by means of principal component analysis (PCA) in qualitative evaluation of various coal types". In: *physicochemical problems of Mineral processing* 50.2 (2014), pp. 575–589.