Prediction of Fatigue on Rotating-Shift Workers

Anh Tuan Tran

A Thesis
in
The Department of
Computer Science and Software Engineering
Gina Cody School of Engineering and Computer Science

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

July 2019

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Anh Tuan Tran**
Entitled: **Prediction of Fatigue on Rotating-Shift Workers**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

Dr. Jinqiu Yang _____ Chair
Dr. Adam Krzyzak _____ Examiner
Dr. Tien D. Bui _____ Examiner
Dr. Brigitte Jaumard _____ Supervisor
Dr. Tristan Glatard _____ Co-supervisor
Dr. Diane B. Boivin _____ Co-supervisor

Approved by _____
Chair of Department or Graduate Program Director

_____ 2019 _____ _____
Dean
Faculty of Engineering and Computer Science

Abstract

Prediction of Fatigue on Rotating-Shift Workers

Anh Tuan Tran

Rotating shifts have become prevalent in many industries, leading to a growing concern about the impact of fatigue on workers performance and safety. Thus, it is useful to develop a method to predict the fatigue of workers with rotating shifts. This thesis aims at contributing to the development of such method by building data-driven models to predict level of fatigue.

We use random forest classifier and random forest regressor to build two fatigue prediction models. A third model is built by a combination of random forest classifier and regressor. Two imbalanced datasets from different groups of workers in the same industry are used. We explore two strategies to deal with imbalanced datasets: random over-sampling and class weights. We select features with feature importance of random forest and discover that a set of 19 features, selected from 38 original features, gives best performance.

We obtain good prediction accuracy on both datasets. The combined model reaches mean absolute error of 0.93 and 0.83 on two datasets, on a 9-level scale of fatigue. In the area of high level of fatigue, which in real work is of particular interest, our model can predict with average 85% confidence that the true level falls into ± 1 range of prediction.

We conclude that fatigue can be predicted with high confidence, based on a dataset of sleep patterns, work schedules and demographic data. Future work will focus on model generalization to datasets from different industries or geographical areas; and the discovery of other sets of features that give better prediction.

Acknowledgements

First, I wish to express my gratitude to my supervisors, Drs. Brigitte Jaumard, Tristan Glatard and Diane B. Boivin, for all their guidance and support throughout my whole time as a master's student at Concordia University and as a research assistant at the Douglas Mental Health University Institute. I would also like to thank Dr. Philippe Boudreau at Douglas Mental Health University Institute for his collaboration throughout the project and his great advice on the text of this thesis. They always have been engaged and helpful in my work and ready with feedback and comments that greatly assisted me during my studies.

Second, I would like thank my wife and my two children who are the motivation for me to pursue graduate studies. I would not have had this amazing opportunity to pursue my dreams if it was not for your support, encouragement and love.

Contents

Glossary

AI Artificial Intelligence. 1, 2

AVT Auditory Vigilance Task. 9

CART Classification And Regression Tree. 32

CCOHS Canadian Centre for Occupational Health and Safety. 2

CF Chronic Fatigue. 3

CMAE Class Mean Absolute Error. 42, 43, 45, 53, 58, 61–63, 65, 68, 70, 71, 78

CP Class Precision. 43, 45, 48, 51, 53, 54, 58, 61–65, 68, 70–72, 78

CT Computerized Tomography. 1

DS1 Dataset 1. 4, 24, 25, 30, 45–48, 50–52, 54–61, 63–68, 70–72, 74

DS2 Dataset 2. 4, 18, 24, 25, 30, 44–60, 62–74

EEG Electroencephalography. 7, 9

ICAO International Civil Aviation Organization. 20

kNN k-nearest Neighbors. 4, 10

KSS Karolinska Sleepiness Scale. 4, 7–9, 15, 18, 20, 21, 25, 30, 41, 44, 45, 47–49, 51, 52, 54, 58, 60–62, 64, 65, 67, 69, 70, 75–78

MAE Mean Absolute Error. 41, 42, 45, 58, 65, 68, 70–72, 75–77

ML Machine Learning. 6, 21, 23

MWMOTE Majority Weighted Minority Oversampling TEchnique. 13

PVT Psychomotor Vigilance Test. 20, 21, 75, 77–79

RMSE Root Mean Squared Error. 41, 78

SMOTE Synthetic Minority Over-sampling Technique. 77

SVM Support Vector Machines. 4, 9, 10, 79

TPM Three Process Model. 9, 76

TSB Transportation Safety Board of Canada. 2

Chapter 1

Introduction

Machine Learning has become increasingly popular in the last decade due to the availability of massive data sources. In 1997, Deep Blue marked history with a win over chess champion Gary Kasparov, the first time a software won over human in chess. In 2016, DeepMind's AlphaGo defeated world champion Lee Sedol in the game of Go, a game considered to be much more complex than chess (Borowiec [2016]). AlphaGo owes its success to the availability of massive datasets, in addition to machine learning algorithms. And yet, under just three years, AlphaGo was thrashed by its enhanced version AlphaZero with a 0-to-100 loss.

In the field of medicine, machine learning has applications in many branches such as disease diagnosis, drug development and treatment. Artificial Intelligence (AI) applications, a branch of machine learning, are now used to analyze tests, X-Rays, CT (Computerized Tomography) scans, data entry and other tasks. For instance, a company named Atomwise[1] uses deep neural networks to predict the possible effectiveness of new medicines, without the need for costly and time-consuming physical synthesis and testing. U.K.-based startup Babylon provides *online doctor consultations and advices*[2] and claims to be able to analyze "hundreds of millions of combinations of symptoms" in real time and provide consultations.

[1] `https://www.atomwise.com/`
[2] `https://www.babylonhealth.com/`

The application of AI has been introduced into the area of human factor research. The impact of fatigue is one of the most concerned issues at workplace in modern society (Walker [2017]). Being tired at work has been claimed to be as dangerous as consuming alcohol or drugs. Verster et al. [2011] states that prolonged nocturnal driving can be as dangerous as alcohol-impaired driving. The Transportation Safety Board of Canada (TSB) in 2018 releases Watchlist 2018 in which it states that employee fatigue is a major safety hazard in all three modes of transportation, namely aviation, rail and marine. Fatigue has been found to be a risk or contributing factor in more than 90 TSB investigations since 1992, especially in a 24/7 industry where crews can work long and irregular schedules (TSB [2018]).

1.1 Background and Motivation

Fatigue, as defined by the Canadian Centre for Occupational Health and Safety (CCOHS [3]) is the state of feeling very tired, weary or sleepy resulting from insufficient sleep, prolonged mental or physical work, or extended periods of stress or anxiety. Fatigue at work has become prevalent in our modern society due to the increased level of stress and prolonged work shifts. Undetected fatigue at work can lead to degraded performance, errors, incidents and accidents in operational settings (Belenky et al. [2014]). In some types of work, such as health care or driving, fatigue can pose safety risks to the worker as well as other individuals. Early prediction of fatigue is important for mitigation of fatigue-related risks.

Individual level of alertness is regulated by biological processes in a way that it reaches optimal level during daytime schedule. In the case of rotating-shift workers, irregular work schedules cause sleep disturbances and reduced level of performance (Wright et al. [2002]; Drake et al. [2004]; Boivin et al. [2012]). Individuals performing shifts late at night or early in the morning are particularly at risk of fatigue-related incidents and accidents at work (Folkard [1997]; Wong et al. [2011]; Boivin and Boudreau [2014]).

[3] https://www.ccohs.ca/

Statistical models have long been applied in studies of fatigue in workplace (Ingre et al. [2014]). A recent study by Ho et al. [2013] used statistical models to identify factors associated with work-related fatigue among hospital workers in Taipei City. Another study by Cai et al. [2018] aimed to investigate the prevalence of fatigue and determine factors associated with fatigue in female medical personnel in 54 hospitals in Zhuhai, China. Furthermore, machine learning techniques have been applied to tackle mental health problems. In the wake of machine learning success in the last decade, numerous studies have tried to apply its techniques to fatigue prediction. For example, Wang et al. [2014] used random forest to differentiate syndrome of chronic fatigue (CF) in traditional Chinese medicine and found that random forest not only offer outstanding performance but also provide valid confidence evaluation to differentiate the CF syndrome.

1.2 Goals and Contributions

This project was initiated by Dr D. B Boivin and data was collected by Dr Boivin's team at the Centre for study and treatment of circadian rhythms at the Douglas Mental Health University Institute[4], McGill University. Following an agreement between Drs Boivin and Jaumard-Glatard from Concordia University, data were made available to the Concordia team and served as the basis for this thesis. Drs D. B Boivin and P. Boudreau supervised the collection of the data sets they provided to us. We will refer to Drs D. B Boivin and P. Boudreau as "the Douglas team" in the remainder of this thesis.

The objective of this thesis is to design and develop data-driven models that can predict workers' level of fatigue in the particular case of rotating schedules. Models take input parameters such as sleep patterns and work schedules and predict the level of fatigue quantitatively. Models must also be able to identify the factors leading to fatigue. Result interpretation is important as we need to provide convincing justifications for the prediction. As the next step in this project, it is planned to develop a software tool to help early detection of fatigue in industry with rotating work shifts. Eventually, we expect this tool to contribute to the reduction of fatigue-related risks at work.

[4]Douglas Mental Health University Institute, `http://www.douglas.qc.ca/`

We analyze two datasets acquired by the Douglas team in a study of fatigue at work. The Douglas team conducted a study on the impact of shift work on the level of fatigue. The study was conducted in two groups of workers in the same industry, but in different geographical areas. In the rest of this thesis, these two groups will be referred to as group 1 and group 2. The outcome of the study is two datasets of raw data, named DS1 and DS2 after group 1 and group 2 respectively. We then worked in collaboration with the Douglas team to define the appropriate data format to use with the models. DS1 consists of 3,084 records from a group of 26 workers. DS2 consists of 9,782 records from 50 workers. The datasets are of significant sizes in the field of medicine. In addition, data were collected over a period of 28 to 35 days so that each participant has been studied over a complete work cycle including morning, evening and night shifts. The datasets have been curated by professional experts at the Douglas Mental Health University Institute.

We approach the problem of predicting level of fatigue from the supervised classification and regression points of view. Sleepiness has been measured by a 9-point subjective scale, the Karolinska Sleepiness Scale (KSS) in this study, and can be predicted by supervised classification. Regression will also be considered as the KSS values are ordinal. There are a number of algorithms that can be considered for this problem, among which Support Vector Machine (SVM), k-nearest Neighbors (kNN) and random forest. Among existing approaches, random forest will be used as it provides simple yet efficient solutions for our supervised learning problem. Random forest is also able to justify the prediction and quantify the level of importance of each input variable.

In this thesis work, we build three machine learning models to predict fatigue level of workers with data provided by the Douglas team. The first two models take the classification and regression approaches, using random forest classifier and regressor, respectively. The third model combines random forest classifier and random forest regressor to leverage strengths of both. With the third model, we are able to predict the level of fatigue with mean absolute error of 0.83 on the nine-graded scale of fatigue (KSS) on DS2. In addition, we identified the three most influential factors leading to the level of fatigue, i.e. the length of awake time, the sleep duration in the last sleep and the time of day.

This work on prediction of fatigue on rotating-shift workers will be submitted for publication.

1.3 Thesis Plan

The thesis is organized in six chapters. Chapter 2 discusses the background in fatigue prediction on rotating-shift workers. Chapter 3 describes data collection, their attributes, and the main pre-processing steps. Chapter 4 discusses how we formulate the problem in the framework of supervised classification, regression and a combination of both. Chapter 5 details the experimental results obtained with the three modelling approaches. Finally, Chapter 6 reports our conclusions and future work.

Chapter 2

Background and Literature Review

Fatigue has been studied extensively in the field of medicine. A broad range of studies can be found: from quantitative measurements of fatigue to assessment of fatigue predictors. We first review fatigue, sleepiness and their measurements in Section 2.1. The studies of fatigue and sleepiness prediction are then reviewed in Section 2.2. In addition, there are a lot of ML algorithms that have been used in the field of medicine. In Section 2.3 supervised classification and regression algorithms are discussed. Lastly, we review techniques to deal with imbalanced datasets in Section 2.4.

2.1 KSS as Measurement of Fatigue

Sleepiness and fatigue are two interrelated, but distinct phenomena; observed in a number of psychiatric, medical and primary sleep disorders (Shen et al. [2006]). Despite their different implications in terms of diagnosis and treatment, these two terms are often used interchangeably, or merged under the more general lay term of tired (Shen et al. [2006]).

Shen et al. [2006] also gives brief descriptions of sleepiness and fatigue as

> Sleepiness is multidimensional and has many causes (multidetermined) and distinguished from fatigue by a presumed impairment of the normal arousal mechanism.

> Despite its ubiquity, no clear consensus exits as yet as to what constitutes sleepiness. Definitions of sleepiness, to date, are at best operational definitions, conceptualized so as to produce specific assessment instruments. As a result, while a number of subjective and objective measurement tools have been developed to measure sleepiness, each only captures a limited aspect of an otherwise heterogeneous entity.

and

> Fatigue is an equally complex phenomenon, its nature captured by a number of conceptualizations and definitions. Measures of fatigue have remained subjective, with a gold standard for its measurement remaining elusive. Despite a high prevalence and high degree of morbidity, fatigue has remained a relatively under appreciated symptom, from both a clinical and research point of view.

The KSS is frequently used for evaluating subjective sleepiness (Akerstedt and Gillberg [1990]). Numerous studies have shown relatively high correlation between KSS and performance measures (Akerstedt et al. [2005], Gillberg et al. [1994], Hoddes et al. [1973]). Kaida et al. [2006] concluded that KSS was closely related to waking electroencephalogram (EEG) and behavioral variables, indicating a high validity in measuring sleepiness. An EEG is a recording of the electrical signals of the brain and is used, among other things, to help diagnose epilepsy (waking EEG) and sleep disorders (sleeping EEG) (Kaiser [2007]).

KSS is a 9-graded sleepiness scale, with the following values and descriptions

- 1. Extremely alert
- 2. Very alert
- 3. Alert
- 4. Rather alert
- 5. Neither alert nor sleepy

- 6. Some signs of sleepiness
- 7. Sleepy, but no difficulty remaining awake
- 8. Sleepy, some effort to keep alert
- 9. Extremely sleepy, fighting sleep

In this thesis work, we will use KSS as the output for both classification and regression. As described Section 2.2, other measurement of fatigue were also collected during the study, among them Samn-Perelli Fatigue Scale (Gawron [2016]). This can also be used as output for prediction of fatigue.

2.2 Prediction of Fatigue

While fatigue has been the focus of numerous studies, very few of them have tried to predict it quantitatively. One of the most comprehensive studies was conducted by a group of researchers at The University of South Australia (Dorrian et al. [2011]). In this study, the group investigated fatigue in a large sample of Australian Rail Industry Employees, taking into account work hours, workload, and sleep. Ninety participants were included, from four companies, among them 85 were males and 5 were females, with the median age of 40.2 $\pm$ 8.6 years. Data acquisition process was very close to the study we worked on. Objective measurements of activities and sleep were recorded by wearable devices (Actiwatch[1]). Participants also completed the Samn-Perelli Fatigue Scale (Gawron [2016]), which served as subjective measurement of fatigue, at the beginning and end of shifts. The study did not attempt to predict the level of fatigue; instead, it focused on analyzing the distribution of fatigue measurements (Samn-Perelli) and on the relationship to fatigue predictors such as sleep loss, extended wakefulness, and longer work hours. One notable conclusion of this study is that overall, analysis should be carried out on particular levels of fatigue, not just the average. Another attempt to predict fatigue was carried out by a group of interdisciplinary researchers from National University of Singapore (Shen et al.

[1] http://www.actigraphy.com/solutions/actiwatch/actiwatch2.html

[2008]). The study approaches the problem from classification point of view, using support vector-machines (SVM) (Vapnik [1998]). Ten subjects underwent 25-hour sleep deprivation experiments with Electroencephalography (EEG) (Kaiser [2007]) monitoring. EEG data were segmented into 3-second long epochs and manually classified into 5 mental-fatigue levels, based on subjects performance on an auditory vigilance task (AVT) (Pang et al. [2005]). AVT sessions were performed once an hour during the 25-hour experimental period and the results were used to manually annotate the subject into 5 levels of mental fatigue. The experiments were strictly controlled: participants stayed in a temperature-controlled room, no stimulants were allowed and only non-strenuous physical activities were allowed. The study reports good results of 87.2% classification accuracy. It is worth noting that input data were EEG signals; sleep and work were not taken into account. Limitation of EEG-based prediction of fatigue is clear: EEG is expensive and not easily available.

Another popular model for prediction of sleepiness is the Three Process Model (TPM). The TPM models sleep propensity using time awake (the homeostatic process, called process S), the time of day (or the circadian system, the circadian process, process C) and sleep inertia function (process W). Ingre et al. [2014] extends the model with small modification to predict sleepiness level of airline crews. They collected sleep and sleepiness data from 136 aircrews in a real life situation by means of an application running on a handheld touch screen computer device (iPhone, iPod or iPad) and used the TPM to predict sleepiness with varying level of complexity of model equations and data. Inputs from three processes were used to calculate an alertness score, which in turn, is used to predict KSS level using linear regression. Multiple models were analyzed and the best performances (model 6d) were reported to produce residual error standard deviation of 1.362 .

2.3 Supervised Classification and Regression

Supervised learning is a machine learning task that learns a function to map an input to an output. The function is inferred from historical labeled data consisting a set of training examples (Mohri et al. [2012]). Inferred function can then be applied to unseen examples, called test

examples, to predict the output. Supervised learning can be broadly divided into two categories: classification and regression. In classification, the output is a set of discrete values, i.e, the test example is classified into one of the possible classes. In regression, the output is real valued. In classification, popular algorithms are k-nearest neighbors (kNN) (Altman [1992]), support vector-machines (SVM) (Vapnik [1998]) and random forest (Breiman [2001]). While these algorithms are popular for classification, all of them can also perform regression.

kNN is based on a distance function that calculates the distance between two examples in the input feature space. kNN classifies test examples by the popular votes from its k nearest neighbors. Test examples are assigned to the class most common in their k nearest neighbors. In regression, the output value is simply the average of the outputs of its k nearest neighbors. kNN has become popular due to its simplicity: one has not much to do other than selecting k and finding an appropriate distance function. Yet, kNN has a few limitations, most notably poor run-time performance and sensitivity to redundant features. The kNN algorithm calculates the distances between the test examples and every single data points in the training set; consequently it suffers from poor performance when the dataset is large. kNN is sensitive to redundant features because, in distance calculation, it treats every feature in the input feature space equally. One of the most successful applications of kNN is in recommender systems (Bobadilla et al. [2013]).

In classification, SVM algorithms separate the training dataset by constructing a hyperplane or a set of hyperplanes in the input feature space. A hyperplane is calculated so that it has largest distance to the nearest training data points of any class. It happens often in real world that data are not linearly separable, i.e., we cannot find a linear function of the input feature that separates the dataset. The input feature space can then be mapped to a higher dimensional space through a kernel function (Scholkopf and Smola [2001]). This is one of the most important advantages of SVM over other algorithms; it gives the flexibility in choosing the form of feature transformation by different kernel functions. One major disadvantage of SVM is the lack of transparency of the results (Auria and Moro [2008]). In other words, results cannot be easily explained or justified. Most successful applications of SVM are in text categorization (Sebastiani

[2002]; Joachims [1998]; Sun et al. [2002]), image classification (Tarabalka et al. [2010]; Lin et al. [2011]) and bioinformatics (Byvatov and Schneider [2003]; Bhasin and Raghava [2004]).

Random forest is a popular ensemble method for predictive models. The ensemble is built on individual predictors (in this case, decision trees) to form a more powerful model. The decision trees are built randomly from a subset of training examples and therefore are independent of each other. Once training is done, the forest can be used as classification or regression model. In classification, each predictor produces class probabilities of the test example being in each class. The probabilities are then averaged over the entire forest and the class label with highest average probability is the predicted label of the example. In regression, the output value is the average of output values produced by individual predictors. Random forest is popular for being unbiased and providing an easy interpretation of results. One drawback of random forest is the computational complexity if training data is large. Random forest has successful applications in a broad range of topics, from genetics (Goldstein et al. [2010]), remote sensing (Belgiu and Drgu [2016]), to medial diagnosis (Yang et al. [2009]).

In this thesis work, we choose random forest for both classification and regression tasks for two reasons. First, the results are explainable. It is worth noting that one of the objectives in this study is to identify and explain factors leading to fatigue. Secondly, random forest is robust to over-fitting. This is important because the datasets, as described in Chapter 3, are relatively small from the machine learning point of view. Lastly, random forest provides feature importance which is useful for interpretation of results. The drawback of random forest in this study is minimal because, as noted above, the datasets are relatively small.

2.4 Learning from Imbalanced Datasets

Imbalanced datasets refer to the datasets with very different numbers of samples in each class. In some cases, least-populated class has much smaller number of samples compared to most-populated class. From the regression point of view, it can be seen as dataset with skewed distribution of output values. Learning from imbalanced datasets is practical because in the

real world, more often than not, we have imbalanced datasets. Examples of imbalanced datasets are: i) credit card transactions where a very small fraction of them are fraudulent, the rest are normal; ii) medical images where a small fraction of them contains tumours , and iii) set of emails where, sadly, most of them are spams. This poses a difficulty for learning algorithms, as they will be biased towards the majority group Krawczyk [2016]. In this section, we review two of the most frequently used techniques to deal with imbalanced datasets: resampling and class weights.

2.4.1 Re-sampling

Re-sampling is a general term referring to the changes in distribution of output values in a dataset. It can refer to over-sampling, under-sampling or a combination of both. In over-sampling, new data points of the minority class or classes are added to the dataset. In under-sampling, data points from majority class or classes are removed from dataset. The combination, sometimes called mix sampling or mix-ratio sampling (Bae et al. [2010]), performs both under- and over-sampling.

Under-sampling

Under-sampling is a popular technique to deal with the analysis of imbalanced datasets. It uses only a subset of majority class or classes (Liu et al. [2009]). Existing approaches to under-sampling are random, distance-based and cluster-based. Simplest method is random under-sampling where samples from majority class are sampled randomly and combined with samples from minority class to form a balanced training set.

Chyi [2003] proposes a distance-based under-sampling method where samples of the majority class are chosen based on their distances to all samples of the minority class. The choice can be made on four modes: nearest, farthest, average nearest and average farthest. Mani and Zhang [2003] also take the distance-based approach and propose four methods. *"Nearmiss-1"* selects samples from majority class with smallest average distance to *three nearest* samples of minority; *"Nearmiss-2"* selects samples from majority class with smallest average distance to

three farthest samples of minority; *"NearMiss-3"* selects a given number of the closest majority class samples for each minority class sample and *Most distant* selects the majority class samples with largest average distances to the *three closest* minority class samples.

Liu et al. [2009] took a different approach and proposed two algorithms: an *EasyEnsemble* which samples several subsets from the majority class, trains a learner using each of them, and combines the outputs of those learners; and *BalanceCascade* trains the learners sequentially, where in each step, the majority class examples that are correctly classified by the current trained learners are removed from further consideration.

Cluster-based under-sampling is a method proposed in Yen and Lee [2009]. The method first clusters the dataset into a number of clusters, then selects a suitable number of samples of the majority class from each cluster based on the ratio between minority and majority classes in that cluster.

Over-sampling

Over-sampling is another technique to deal with imbalanced datasets where new examples are added to the dataset. Similar to under-sampling, common approaches to over-sampling are random and distance-based. Simplest method is random over-sampling where samples from minority class are duplicated randomly and combined with samples from majority class to form a balanced training set.

SMOTE (Synthetic Minority Over-sampling Technique) (Chawla et al. [2002]) is one of the most popular method of over-sampling. SMOTE generates new examples of the minority class based on existing ones. The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors (Chawla et al. [2002]). Due to success of SMOTE, a number of SMOTE-based methods have been proposed such as *Borderline-SMOTE* (Han et al. [2005]) and *Kernel-based SMOTE* (Mathew et al. [2015]).

Barua et al. [2014] introduced a new method, called Majority Weighted Minority Oversampling TEchnique (MWMOTE). MWMOTE first identifies the hard-to-learn informative minority class

samples and assigns them weights according to their euclidean distance from the nearest majority class samples (Barua et al. [2014]). It then generates the synthetic samples from the weighted informative minority class samples using a clustering approach (Barua et al. [2014]).

2.4.2 Class Weights

Weighting or class weights is an algorithm-level method for dealing with imbalanced datasets (Kotsiantis et al. [2006]). The idea is to compensate for the imbalance in the training set without actually altering the class distribution (Barandela et al. [2003]). A weighted distance function is used in Barandela et al. [2003] for *kNN* algorithm, which gives higher weight to samples of majority class. In this way, distances from new test examples to the samples of majority class are much higher than to samples of minority class. This produces a tendency for the new samples to find their nearest neighbors among the minority-class examples.

Weighting can also be done with random forest. Since random forest classifier tends to be biased towards the majority class, we can give higher weights to examples of minority class Chen et al. [2004]. Class weights are used in building the forests (via building decision trees) in calculation of splitting criterion (Gini or entropy). After a decision tree is built, in each leaf node prediction is based on a weighted probability, i.e., number of samples in each class multiplied by class weight divided by weighted sum. The final prediction of the forest is the average of all individual trees in it. In this way, one can manipulate class weights to obtain desired results from random forest.

In this chapter, we discussed the problem background and solutions suggested in the literature. We discussed different machine learning algorithms that we will use to build data models for fatigue prediction. We also discussed the techniques to deal with imbalanced datasets. In the next two chapters we discuss the datasets and the implementation of models for fatigue prediction.

Chapter 3

Datasets Acquisition and Preprocessing

This chapter describes the acquisition of the datasets and how they were pre-processed. We first describe the data acquisition process and pre-processing of raw data, which was conducted by the research of the Douglas team. We also define the record format to be used in the classifier and regressor, based on the availability of KSS values. Finally, we present the data inclusion and exclusion criteria, as well as the method used to manage missing data.

3.1 Data Acquisition Process

The study was conducted on two groups of workers in the same industry but in different geographical area. In group 1, a total of 26 workers participated in the study. 18 of them were male and 8 were female. They were 30.75 years of age in average, with the youngest at 23.65 and oldest at 40.88 years of age, respectively. In group 2, 50 workers participated, among them 38 males and 12 females. They were at 32.7 years of age on average, with the youngest one at 24.37 and the oldest one at 49.13 years of age. For confidentiality reasons, results will not be provided by subgroups of age and sex.

Before the start of the study, the demographic variables presented in Table 3.1 were collected. A number of variables were removed from the list due to feature selection or impracticality.

Table 3.1 Demographic variables

No	Name	Description	Type	Note
1	age	Age of the participant	Numerical	
2	sex	M: male, F: female	Categorical	Removed by feature selection
3	month	Month of study (weighted average of study period)	Numerical	Removed due to impracticality (i.e., not available on test samples)
4	ho_chronotype	Horne-Ostberg's Morningness-Eveningness Score	Numerical	
5	ISI	Insomnia Severity Index (Smith and Wegener [2003])	Numerical	
6	ESS	Epworth Sleepiness Scale	Numerical	Removed by feature selection
7	average_sleep_hours	Average sleep hours of participant, in rest days	Numerical	
8	average_energy_drink	Average number of energy drinks per day	Numerical	Removed due to impracticality (i.e., not available on test samples)
9	average_alcohol	Average number of alcohol consumption per day	Numerical	Removed due to impracticality (i.e., not available on test samples)
10	average_cigarette	Average number of cigarettes per day	Numerical	Removed due to impracticality (i.e., not available on test samples)

Each worker was studied during a complete work cycle comprising 28 or 35 days. This work cycle was their habitual work schedule and included day, evening and night shifts alternating with rest days. Workers were requested to maintained their usual sleep-wake behaviours throughout the study period.

The data were collected using three different tools, as described below.

First, each participant was equipped with an *Actiwatch*[1], which was advised to be worn around-the-clock, to collect *objective* measurements on their sleep-wake cycles and light exposure. Actiwatches measure physical activity and light exposure as time series. A hierarchical approach, similar to that described in Patel et al. [2015], was used by the Douglas team to determine bedtimes and wake times of each participant (time in bed). Then, the algorithm described in Oakley [1997] and Kosmadopoulos et al. [2014] was used to determine objective sleep parameters during each sleep period.

Sleep parameters derived from actiwatch recordings are presented in Table 3.2.

[1] Actiwatch Spectrum, Philips/Respironics, OR, US

Table 3.2 Actiwatch variables

No	Name	Description	Type
1	AW_Bedtime	Bed time, as detected by the Actiwatch	Numerical
2	AW_Waketime	Wake time, as detected by the Actiwatch	Numerical
3	AW_TIB	Actiwatch: time in bed (time from bedtime to getting out of bed)	Numerical
4	AW_TST	Actiwatch: total sleep time (from sleep onset to final awakening-wakefulness during that interval)	Numerical
5	AW_SOL	Actiwatch: Time the participant fall asleep (decimal)	Numerical
6	AW_Snooze	Actiwatch: Time from final awakening to getting out of bed	Numerical
7	AW_SETIB	Actiwatch: Sleep efficiency based on time in bed (TST/TIB)	Numerical
8	AW_SETST	Actiwatch: Sleep efficiency based on total sleep period (TST/SP)	Numerical

Second, participants were requested to complete questionnaires, using a smartphone, approximately 5 times per day:

- Q1: wake time
- Q2: start of shift
- Q3: middle of shift
- Q4: end of shift
- Q5: bed time

On rest days, work-related questionnaires were replaced with questionnaires done at a specific time of day:

- Q2: at approximately 2 hours after wake time
- Q3: at approximately 8 hours after wake time
- Q4: at approximately 2 hours before bedtime

All questionnaires included questions about the participant's vigilance (visual analogue scale from 1 to 100), mood (visual analogue scale from 1 to 100), fatigue (Samn-Perelli Scale) and sleepiness (KSS) levels. Each questionnaire also included additional questions depending on time of administration. Q1 included sleep-related questions: bed time, wake time, time to fall asleep, total sleep time, sleep quality and sleep disturbances. Q4 included questions about the shift start and end time (if any), how many extra hours of work, how mentally and physically demanding were their tasks, the impact of weather condition, the level of effort put in the task, levels of irritability, how many nodding-offs did they have, the level of appetite, whether they had meal and whether it was a full meal. Q5 included questions about the level of irritability, number of cigarettes, number of energy drinks and number of alcoholic beverages that the participant had since last main sleep period.

Some participants of DS2 (n=50) were also requested to fill out questionnaires at meal times. In Table 3.3, a comprehensive list of variables is presented.

Table 3.3 Variables from questionnaires

No	Name	Description	Type	Note
1	form_start_time	Time stamp of the questionnaire	Date	
2	samn_perelli	Samn-Perelli Fatigue Scale	Categorical	Output
3	KSS	Karolinska Sleepiness Scale	Categorical	Output
4	VigilanceVas	Subjective vigilance	Numerical	From 0=very sleepy to 100=very alert
5	HumeurVas	Subjective mood	Numerical	From 0=very bad to 100=very good
6	AS_HeureCouche_dec	Bed time	Numerical	Time of day
7	AS_HeureLeve_dec	Wake time	Numerical	Time of day
8	AS_HeuresSommeil_dec	Number of sleep hours	Numerical	Time of day
9	AS_MinutesEndormir	Time to fall asleep (in minute)	Numerical	
10	AS_QualiteSommeilVas	Sleep quality	Numerical	From 0=very bad to 100=very good
11	conditionclimvas	Weather condition during work shift	Numerical	From 0= very bad to 100 = very good
12	nasa_exigencemental	The level of mental demand of the task	Numerical	From 0=very low to 100 = very high
13	nasa_exigencephysique	The level of physical demand of the task	Numerical	From 0=very low to 100 = very high
14	NASA_ExigenceTemporelle	The pace of the task	Numerical	From 0=very low to 100=very high
15	NASA_Performance	The level of success in accomplishing what they were asked to do	Numerical	From 0=very low to 100=very high
16	NASA_Effort	The effort to accomplish their level of performance	Numerical	From 0=very low to 100=very high
17	NASA_Frustration	The level of being insecure, discouraged, irritated, stressed and annoyed	Numerical	From 0=very low to 100=very high
18	heuresupp	Number of overtime work hours	Numerical	
19	appetit	The appetite of participant	Numerical	From 0=really low to 100=really high
20	hv_nb_cigarettes	Number of cigarettes since the last main sleep	Numerical	
21	hv_nb_boissons_energie	Number of energy drinks since the last main sleep	Numerical	
22	hv_nb_boissons_alcohol	Number of alcoholic drinks since the last main sleep	Numerical	
23	perioderepas	If they had a break to eat today	Categorical	1=Yes, 2=No
24	repasouinon	If they ate during your work shift today	Categorical	1=Yes, 2=No
25	repascomplet	If it was a full meal (as opposed to snacks)	Cagetorical	1=Yes, 2=No
26	irritabilite	The level of irritability	Numerical	From 0=very low to 100=very high

Finally, participants were asked to perform a Psychomotor Vigilance Task (PVT) on the smartphone. We used a 5-minute test validated by Lamond et al. [2008] that repeatedly presents a visual cue at pseudo-random intervals ranging from 2s to 10s. At each cue presentation, the participant simply touched the screen as quickly as possible, clear the stimulus and start the next trial. Output of a PVT session is presented in Table 3.4.

Table 3.4 PVT session variables

No	Name	Description	Type	Note
1	PVT_Derange	If the participant was disturbed in the PVT session	Categorical	Yes/No
2	meanReactionTime	Mean reaction time (in milliseconds)	Numerical	
3	meanResponseSpeed	Mean reaction speed	Numerical	
4	medianReactionTime	Median reaction time (RT, in 1/ms)	Numerical	
5	medianResponseSpeed	Median response speed	Numerical	
6	meanTop10PercentResponseSpeed	Mean of top 10 percent response speed	Numerical	
7	meanBottom10PercentResponseSpeed	Mean of bottom 10 percent response speed	Numerical	
8	minorLapses	Number of misses (RT>500ms)	Numerical	
9	totalReactions	Total number of reactions	Numerical	
10	validReactions	Total number of valid reactions	Numerical	
11	falseStarts	Number of false starts (RT <200ms)	Numerical	

As described above, different variables can be used as indicator of fatigue. They can be divided into two groups: subjective and objective variables. In the subjective group, we collected KSS, Samn-Perelli Fatigue Scale and vigilance levels. In the objective group, we collected measurements from PVT sessions.

In this thesis, we choose to use KSS for three reasons. Firstly, all of those scales have been proposed as measures of fatigue (Gander et al. [2015]). Under the ICAO definition of fatigue, sleepiness can be considered one manifestation of fatigue-related impairment (Gander et al. [2015]). With increasing time awake, sleepiness and fatigue increase and PVT response rates slow. Sleep restriction also increases sleepiness (Akerstedt and Gillberg [1990]; Kaida et al. [2006]) and postsleep fatigue ratings (Ferguson et al. [2012]), and slows PVT response speeds (Belenky et al. [2003]; Van Dongen et al. [2003]). Thus, in the laboratory, these measures reliably reflect the physiological changes that cause fatiguerelated impairment (Gander et al. [2015]). Ingre et al. [2006] also showed that subjective sleepiness measured with the KSS was

strongly related to accident risk. Secondly, in this study the participant took PVT sessions only at the beginning and end of shift. As a result, the number of records with available PVT is much smaller than those with KSS available. A very small dataset would make it very difficult to build an ML model. Thirdly, PVT is much more difficult and expensive to include in a study than subjective measures. Therefore, successful prediction models based on subjective measures may simplify future studies.

3.2 Record Definition

After the acquisition process by the Douglas team, we received initial data in the following formats:

- The daily log containing data collected through the smartphone for all participants as described in Section 3.1
- The validated sleep schedule of all participants
- The validated work schedule of all participants
- Demographic information about all participants

In order to use the data in the model, we need to incorporate the data into a single structured type of records. As we try to predict the level of fatigue based on the work scheduling and sleep-wake pattern, it is reasonable to define a record that:

- is based on the events at which KSS was taken
- has variables to reflect the previous work shifts and sleep-wake pattern
- has variables to reflect the demographic factors of the participant

We then define record structure to use in the model with variables described in Table 3.5. The calculations of cumulative variables are described in Sub-section 3.4.1. Note that all sleep

variables are calculated from the Actiwatch data, not from the self-assessment data (questionnaires).

Table 3.5: Variable list

No	Name	Description	Type
1	questionnaireno	Type of record: 1: wake up 2: start of shift 3: middle of shift (also meal time in group 2) 4: end of shift 5: sleep	Categorical
2	time_of_day	Decimal reprentation of time of day, 0.00 to 23.99	Numerical
3	day_of_year	Day of year (1-366)	Numerical
4	consecutive	Number of consecutive days that the worker had been doing the same shift type	Numerical
5	time_awake	Length of time, in hours, since awakening of the last main sleep (i.e., excluding naps)	Numerical
6	time_awake_less_nap	Length of time, in hours, since awakening of the last main sleep, less total sleep time of naps taken in between	Numerical
7	time_since_start_of_shift	Length of time since the start of work shift - 0: for start_of_shift record - actual duration: for middle_of_shift and end_of_shift records - missing: for other types of records	Numerical
8	time_in_bed	Total time in bed, from bed time to rise time	Numerical
9	sleep_time	Total sleep time	Numerical
10	sleep_onset_latency	Duration between bed_time and actual time of falling sleep	Numerical
11	total_sleep_time	Time between sleep onset to final awakening	Numerical
12	snooze	Duration between final awakening and rise time	Numerical
13	sleep_time_24h	Total time slept in the last 24 hours	Numerical
14	sleep_time_48h	Total time slept in the last 48 hours	Numerical
15	sleep_time_72h	Total time slept in the last 72 hours	Numerical
16	sleep_time_7d	Total time slept in the last 7 days	Numerical
17	sleep_time_night	Total time slept at night* in the last sleep	Numerical
18	sleep_time_night_24h	Duration of work at night** during the previous work shift	Numerical
19	sleep_time_night_48h	Total time slept at night* in the last 48 hours	Numerical
20	sleep_time_night_72h	Total time slept at night* in the last 72 hours	Numerical
21	sleep_time_night_7d	Total time slept at night* in the last 7 days	Numerical
22	work_duration	Duration of the previous work shift	Numerical
23	work_duration_24h	Cumulative duration of work time in the last 24 hours	Numerical
24	work_duration_48h	Cumulative duration of work time in the last 48 hours	Numerical
25	work_duration_72h	Cumulative duration of work time in the last 72 hours	Numerical

Table 3.5: Variable list

No	Name	Description	Type
26	work_duration_7d	Cumulative duration of work time in the last 7 days	Numerical
27	work_duration_night	Duration of the previous work shift at night**	Numerical
28	work_duration_night_24h	Cumulative duration of work time at night** in the last 24 hours	Numerical
29	work_duration_night_48h	Cumulative duration of work time at night** in the last 48 hours	Numerical
30	work_duration_night_72h	Cumulative duration of work time at night** in the last 72 hours	Numerical
31	work_duration_night_7d	Cumulative duration of work time at night** in the last 7 days	Numerical
32	time_since_end_of_shift	Length of time, in hours, since the end of the previous work shift	Numerical
33	ho_chronotype	Horne-Ostberg's Morningness-Eveningness Score	Numerical
34	ISI	Insomnia Severity Index (Smith and Wegener [2003])	Numerical
35	average_sleep_hours	Average total sleep time of the participant during rest days	Numerical
36	age	Age of the participant	Numerical
		* night, when it comes to sleep is defined as between 22:00 to 08:00 of the next day ** night, when it comes to working is defined as between 00:00 to 05:00	

3.3 Data Cleaning

There were two phases of cleaning the datasets: first, from raw data to intermediate data; second, from intermediate data to the model input. This process applied for both datasets.

In a joint effort between the Douglas and Concordia team, the "daily log" described in Section 3.1 was further cleaned to meet ML constraints. First, we discussed and validated the records by the criteria described in Table 3.6.

Table 3.6 Validation of records

No	Record type	Criteria for data exclusion	Action
1	wake_up (Q1)	timestamp does not fall into a range of ± 1 hour of bed_time of the validated sleep schedule	Remove
2	sleep (Q5)	timestamp does not fall into a range of ± 1 hour of rise_time of the validated sleep schedule	Remove
3	start_of_shift (Q2)	timestamp does not fall into a range of ± 1 hour of work in as described by the validated work schedule	Remove
4	end_of_shift (Q4)	timestamp does not fall into a range of ± 1 hour of end of work as described by the validated work schedule	Remove

In addition to the above criteria, the Douglas team, based on their field expertise, decided to bring back records that fall into one of the following categories:

- At the study start, the first wake_up questionnaire filled by each participant. They must have been removed by rule number 1 in Table 3.6 since the presumed sleep was not tracked.
- The start_of_shift questionnaires which were filled after rise_time but more than 1 hour before work_in, since there is minimal chance of "future recall bias". 23 records were brought back under this process, with 1 in DS1 and 22 in DS2.

Lastly, variables were examined for their missing values and records with those missing values were removed as described in Table 3.7

Table 3.7 Removal of missing values

No	Criteria	# records		Action	Description
		DS1	DS2		
1	KSS is missing	9	39	Remove	KSS is the label
2	time_awake_less_nap is missing	42	249	Remove	Indicates missing of sleep schedule
3	sleep_time_24h is missing	74	274	Remove	Indicates missing of sleep schedule
4	time_in_bed is missing	0	36	Remove	Indicates missing of sleep schedule
5	sleep_time_48h is missing	102	0	Remove	Unreported sleep time/bad sleep schedule

After this process, we had 2,838 records in DS1 and 9,243 records in DS2.

3.3.1 Detection and Management of Inconsistencies

In this phase, we try to detect and manage inconsistencies in the datasets. Inconsistencies may occur in two cases:

- One participant gives very different KSS values in a relatively short period of time
- A work-related event (questionnaires Q2, Q3 or Q4) was reported inside a sleep period.

In both cases, we examined all records and found out that sometime participants who forgot to fill out one questionnaire, tried to fill it out at the time of a subsequent one by "recalling" how they felt. For example, they forgot to fill mid_of_shift (Q3) and at the time of end_of_shift (Q4) they tried to fill both Q3 and Q4 by "recalling" what they felt at the time of Q3. It is worth noting that in assessing KSS, the participants were specifically asked how they felt at the time of questionnaire completion. Yet, the model will treat Q3 as an ordinary record because the timestamp is the time it was taken. When this situation occurs, records representing Q3 and Q4 have almost identical variables, yet possibly very different KSS. This clearly confuses the model and decreases its performance. As a consequence, it is safe to remove records representing events that were reported at a time they were not supposed to be.

3.3.2 Management of Missing Data and Undefined Values

Missing data can occur when one or more sleep-related events were not recorded. This mainly happened as a result of Actiwatch removal or malfunction. Indeed, without the activity data, we were not able to assess the sleep-wake state of the participant. When the participant forgot to report a sleep period, we had no way to evaluate if he/she slept or not. If it happens, the records related to those events (Q1: wake-up; Q5: bed-time) were lost. We did not try to recover these records because we have no information to guess. As mentioned in 3.4.1, there are other records that are affected by the missing of sleep events. Records close in time with the missing data

will have their sleep-related variables affected: time_in_bed, sleep_time, time_awake_less_nap, sleep_time_24h, sleep_time_48h, sleep_time_72h, sleep_time_7d. In these records, missing values are called undefined values.

The undefined values need to be filled before we can use the algorithms (random forest classifier and random forest regressor) as they do not support null values out-of-the-box. We tried to replace those missing values with either a fixed value or with the mean value of the variable. A complete list of variables and corresponding replacements of missing values is presented in Table 3.8.

Table 3.8 Replacements of missing values

Variable	Missing value replacement	Description
time_since_start_of_shift	-9999	Normal value range is 0-16. Higher value means longer time on duty. -9999 indicates 'not during a workshift'
time_since_end_of_shift	9999	Normal value range is 0-240
other variables	mean value	

3.4 Feature Engineering

Feature engineering involves two processes: feature extraction and feature selection which will be discussed in details in the next subsections.

For the feature extraction process, we worked in close collaboration with the Douglas team in an effort to incorporate their expertise in fatigue assessment into the dataset. The output of this process is a new cumulative set of features calculated based on the existing features. In feature selection, we try to remove irrelevant or redundant features in order to get a simpler model and better performance.

3.4.1 Feature Extraction

Capturing the effects of sleep-wake schedule

In this process, we created a set of new features to reflect the cumulative nature of certain process, namely sleep loss. First, in order to capture the long-term effect of sleep loss, we introduced the following variables

- *sleep_time_24h*: the total time slept in the last 24 hours
- *sleep_time_48h*: the total time slept in the last 48 hours
- *sleep_time_72h*: the total time slept in the last 72 hours
- *sleep_time_7d*: the total time slept in the last 7 days

In case of missing information to calculate these new features, for example, during the first day of the study, they were marked as undefined.

Capturing the effects of disruption of circadian system

In addition, in order to capture a possible cumulative circadian disruption effect, we introduce the following features:

- *sleep_time_night*: total sleep time at night* in the last sleep period
- *sleep_time_night_24h*: total sleep time at night* in the last 24 hours
- *sleep_time_night_48h*: total sleep time at night* in the last 48 hours
- *sleep_time_night_72h*: total sleep time at night* in the last 72 hours
- *sleep_time_night_7d*: total sleep time at night* in the last 7 days

* where *night*, when it comes to sleep, is defined as the period between 22:00 and 08:00 of the next day. In case of inadequate information to calculate these variables, for example in the first day of the study, they will be marked undefined.

Capturing the effects of prolonged cumulative work time

In order to capture the effect of work shift duration and timing, we introduce the following variables

- *work_duration_24h*: cumulative duration of work time in the last 24 hours
- *work_duration_48h*: cumulative duration of work time in the last 48 hours
- *work_duration_72h*: cumulative duration of work time in the last 72 hours
- *work_duration_7d*: cumulative duration of work time in the last 7 days
- *work_duration_night_24h*: cumulative duration of work time at night** in the last 24 hours
- *work_duration_night_48h*: cumulative duration of work time at night** in the last 48 hours
- *work_duration_night_72h*: cumulative duration of work time at night** in the last 72 hours
- *work_duration_night_7d*: cumulative duration of work time at night** in the last 7 days

** where *night*, when it comes to work time, is defined as the period between *00:00* and *05:00*.

Timestamp Extraction and Transformation

In the original data, one particularly important feature is the *record_timestamp* which is the time the event took place. It carries within a single variable multiple valuable information: the time of day, the day of the week, and the day of the year. The time of day clearly has impact on a worker's performance. One might expect that the day of year contribute to the performance as well, as it reflects the surrounding environmental conditions. Thus we extract two variables from *record_timestamp*:

- *time_of_day*: the numerical value representing time of day, ranging from 0 to 23.99 inclusively.
- *day_of_year*: the numbered day of the year, ranging from 1 to 366 inclusively.

Furthermore, we want the model to take into account the fact that these are cyclical variables, meaning that the variables repeat themselves cycles after cycles. For examples, *day_of_year* runs from 1 to 365 (366 on leap year) and goes back to 1 after. We then expect that day 1 and day 365 are actually as close as day 1 and day 2. In order to mimic the cyclical characteristic of these variables, we transform their values with cyclical functions, such as *sine* and *cosine*.

Thus we transform these variables into

$$time\_of\_day\_sine = sin(time\_of\_day * 2\pi/24)$$

$$time\_of\_day\_cosine = cos(time\_of\_day * 2\pi/24)$$

$$day\_of\_year\_sine = sin(day\_of\_year * 2\pi/366)$$

$$day\_of\_year\_cosine = cos(day\_of\_year * 2\pi/366)$$

The transformed variables are now cyclical.

3.4.2 Feature Selection

Feature selection serves two purposes: simplifying the model and enhancing its performance. First, there might be features in the datasets that are either irrelevant or redundant. Irrelevant features are those that do not provide any information to the algorithms while redundant features convey the same information as other features. It is clear that removal of such features lead to simpler models and faster training time. The results are therefore easier to interpret, since the algorithm is built with reduced feature sets and they are all relevant. It is worth noting that the aim of the models is not only to predict the level of fatigue but also to provide convincing justifications. Secondly, by removing redundant or irrelevant features we expect improvement in performance. As explained in 4.1.1, random forest builds predictors (i.e., decision trees) by selecting a feature out of a subset of feature set to split the current subset of data on. As we remove irrelevant features, we reduce the noise within the dataset and thus reduce variance. This effectively prevents over-fitting of the model and gives more generalization.

In this project, we use feature importance rankings from random forest to do feature selection. Starting with the original feature set in Table 3.5, we iteratively train random forest with the dataset and remove feature with least importance.

3.5 Overview of the Datasets

After pre-processing, DS1 and DS2 both have 38 features (i.e, input variables) and one output. The feature selection process described above results in 19 features which we will report and compare in Chapter 5. DS1 has 2,838 records and DS2 has 9,243 records. The number of records in each label (KSS value) is reported in Table 3.9.

Table 3.9 Label distribution of datasets

KSS	DS1		DS2	
	No. of records	%	No. of records	%
1	45	1.59	183	1.98
2	394	13.88	893	9.66
3	604	21.28	1,754	18.98
4	549	19.34	1,713	18.53
5	441	15.54	1,654	17.89
6	365	12.86	1,215	13.15
7	203	7.15	838	9.07
8	201	7.08	817	8.84
9	36	1.27	176	1.9
Total	**2,838**	**100**	**9,243**	**100**

Chapter 4

Fatigue Prediction as a Supervised Classification and Regression Problem

In this chapter, we present the adaptation of our problem into a classification task, a regression task and finally, a combination of both classification and regression. This chapter is organized in five sections. Section 4.1 presents decision trees and the process of building them. Section 4.2 follows with presentation of random forest. Sections 4.3 and 4.4 discuss classification and regression. In each of these sections we present a problem statement, followed by the algorithm and finally techniques to deal with imbalance datasets. In Section 4.5, we present a model by the combination of a classification and a regression model.

4.1 Decision Trees

Decision Tree is a supervised machine learning algorithm that infers decision rules by splitting the training data based on the features. A decision tree is constructed as a tree where internal

nodes represent a test on one feature of the dataset; branches form the decision path and leaf nodes contain smallest unsplittable subsets that will be used to calculate class probabilities.

In this thesis work, we use the implementation of decision tree and random forest in scikit-learn (Pedregosa et al. [2011]) which are based on Classification And Regression Tree (CART) described in Breiman [2017]. In the next sub section, we summarize the procedure to construct a decision from Breiman [2017].

4.1.1 Constructing a Decision Tree

A CART is constructed in a top-down manner. Given training vectors $x_i \in R^n, i = 1, ..., l$ and a label vector $y_i \in R^l$, a decision tree recursively partitions the space such that the samples with the same labels are grouped together. Let Q be the data set at node m. For each candidate split $\theta = (j, t_m)$ consisting of a feature j and threshold t_m, partition the data into two subsets $Q_{left}(\theta)$ and $Q_{right}(\theta)$ such that:

$$Q_{left}(\theta) = \{(x, y)|x_j \leq t_m\}$$

$$Q_{right}(\theta) = Q \setminus Q_{left}(\theta)$$

where $N_m = |Q|$ is the number of examples in training data Q at node m.

The impurity at m is computed as

$$G(Q, \theta) = \frac{|Q_{left}|}{N_m} H(Q_{left}(\theta)) + \frac{|Q_{right}|}{N_m} H(Q_{right}(\theta))$$

using an impurity function $H(Q)$ depending on whether the task is classification or regression. Impurity functions are described below.

Select the parameters that minimizes the impurity

$$\theta^* = \arg\min_{\theta} G(Q, \theta)$$

Recursively run the procedure for two subsets $Q_{left}(\theta^*)$ and $Q_{right}(\theta^*)$ until one of the stopping condition is reached:

- Maximum depth is reached
- $N_m \leq min_{samples}$
- $N_m = 1$

Prediction process

After a decision tree is built, it predicts label of a test example x with the following procedure:

1. Starting from root node, apply the criteria of the node on example x, i.e., test if $x_j \leq t_m$ where m is the current node that splits on feature j and threshold t_m.
2. If $x_j \leq t_m$, go to the left child of current node, else go to the right child.
3. Repeat two above steps until a terminal node (i.e., a leaf) is reached.
4. Calculate class probabilities, assuming m this the terminal node, N_{mc}, N_m are the number of examples of class c and total number of examples in node m respectively

$$P_c(x) = \frac{N_{mc}}{N_m} \tag{4.1}$$

Impurity function for classification task

If the target is a classification with output taking values $0, 1, ..., k-1$, for node m, representing a region R_m with N_m observations, let:

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

be the proportion of class k observations in node m.

Then the impurity measurement can either be:

- Gini:

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$

- Entropy:

$$H(X_m) = -\sum_k p_{mk} log(p_{mk})$$

where X_m is the training data in node m.

Impurity function for regression task

If the target is a continuous value, then for node m, representing a region R_m with N_m observations, common criteria to minimize as for determining locations for future splits are Mean Squared Error, which minimizes the L2 error using mean values at terminal nodes, and Mean Absolute Error, which minimizes the L1 error using median values at terminal nodes.

Mean Squared Error:

$$\overline{y}_m = \frac{1}{N_m} \sum_{i \in N_m} y_i$$
$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - \overline{y}_m)^2$$

Mean Absolute Error:

$$\overline{y}_m = \frac{1}{N_m} \sum_{i \in N_m} y_i$$
$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} |y_i - \overline{y}_m|$$

where X_m is the training data in node m.

4.1.2 Feature Importance

Feature importance in a decision tree is calculated as the decrease in node impurity, weighted by the probability of reaching that node. The probability of a node is simply the number of

samples in that node divided by total number of samples. In *scikit-learn*, the impurity function used to calculate node importance is *gini*.

Formally, the importance of node m is defined as

$$NI_m = w_m * H(X_m) - w_{left} * H(X_{m(left)}) - w_{right} * H(X_{m(right)})$$

where w_m, w_{left}, w_{right} are probabilities of nodes m, left child of m and right child of m, respectively.

Let F be the set of features of the training set. Let N be the set of all nodes and $N_f \subseteq N$ be the set of nodes that splits on feature f.

The importance of feature f can then be defined as

$$FI_f = \frac{\sum_{i \in N_f} NI_i}{\sum_{k \in N} NI_k}$$

Finally, normalized importance of feature f can be calculated as

$$NFI_f = \frac{FI_f}{\sum_{i \in F} FI_i} \tag{4.2}$$

4.2 Random Forest

Random forest is an ensemble method that can be used for both classification and regression task. Random forest builds a set of independent decision trees using various sub-samples of the training set. The sub-samples are drawn from training data set using a re-sampling technique called *bootstrapping*. With this technique, samples are randomly drawn from the original data set with replacement. Note that the sub-samples have the exact same number of examples with the original data set. The sub-samples are then used to build decision trees using a procedure presented in sub section 4.1.1.

4.2.1 Constructing a Random Forest

Constructing a random forest is simple once the decision trees are built. An algorithm for building a random forest is presented in Algorithm 1. After a forest is built, classification is made by taking averages of class probabilities over all decision trees, and class label whose averaged probability is highest will be predicted as the label of the example. Regression output is simply the average of output over all decision trees.

Algorithm 1 Random Forest for Regression or Classification (from Friedman et al. [2001])

Input:- D, N: training dataset and size; p: total number of features; B: number of trees; n_min: minimum number of samples in a node.

Output:- a set of B decision trees

1: **for** $b \leftarrow 1$ *to* B **do**

2: Draw a bootstrapped sample Z of size N from the training dataset D.

3: Grow a random-forest tree T_b from Z, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_min is reached.

i. Select a random subset, m variables from p.

ii. Pick the best variable/split-point among the m. (Using *Gini index* or other methods).

iii. Split the node into two child nodes.

4: **end for**

return $T = \{T_b : b = 1, ..., B\}$

4.2.2 Feature Importance

In a random forest, importance of feature f is averaged over all individual decision trees in the forest, as described in 4.1.2:

$$RFI_f = \frac{\sum_{t \in T} NFI_{ft}}{|T|}$$

where T is the set of trees in the random forest, NFI_{ft} is the normalized feature importance of feature f in tree t, as calculated in 4.2.

4.3 Classification

4.3.1 Problem Statement

The problem of Fatigue Prediction is written as follows: *For a given labeled training dataset with a number of features and one numerical output (KSS, ranging from 1 to 9), predict the KSS of the new examples.* This is a multi-class classification problem that we will solve using random forest classifier.

The datasets, which are discussed in details in Chapter 3, contain records with input variables and KSS value as output. KSS values are represented by integer numbers ranging from 1 to 9.

4.3.2 Random Forest Classifier

A random forest classifier is built with the algorithm described in Algorithm 1. In order to predict the label of a test example, random forest calculates the averages of class probabilities over all decision trees.

Let $T_b\,(i)$ denote the predicted output of tree T_b for sample i. Let C be the set of possible classes and $P_c^b(i)$ be the probability of sample i belongs to class $c \in C$, output by tree T_b. Prediction of a new sample x is given as:

$$\hat{C}^B\,(x) = \arg\max_c \sum_{b=1}^{B} P_c^b(x)$$

4.3.3 Data Imbalance

In classification task, popular techniques to deal with imbalanced data sets are re-sampling and class weighting as discussed in Section 2.4. Re-sampling can refer to over- or under-sampling or a combination of both. With under-sampling, examples of majority class or classes are removed from the training data. With over-sampling, new examples are added to the minority class or classes. In this thesis work, we use random over-sampling for two reasons: the datasets are

relatively small and they have multiple class labels. As seen in 3.5, the least populated class has very small number of examples, thus under-sampling would result in a very small dataset.

Class weight can also be used to deal with imbalanced datasets. One can try different sets of class weight values to search for the best one. In this work, we use a balanced class weight. Each class is given a weight which is inversely proportional to its frequency in the training data. Now as class weights are introduced, within a decision tree the calculation of class probabilities at a terminal node changes. Equation 4.1 in sub section 4.1.1 becomes

$$P_c(x) = \frac{N_{mc} w_c}{\sum_c N_{mc} w_c} \tag{4.3}$$

where w_c is the weight of class c.

4.4 Regression

4.4.1 Problem Statement

The problem of Fatigue Prediction is written, from the regression point of view, as follows: *For a given training dataset with a number of features and one numerical output (KSS, ranging from 1 to 9), predict the KSS of the new examples.* This is a regression task with that we will solve using random forest regressor.

The datasets, which are discussed in details in Chapter 3, contain records with input variables and KSS value as output. KSS values are represented by real number ranging from 1 to 9.

4.4.2 Random Forest Regressor

A random forest regressor predicts the output value simply as the averaged outputs of all trees. Adapted to our problem, we take the output of random forest regressor and round it to the

nearest integer. Using the same notation as in 4.3.2, prediction of a new sample x is given as:

$$\hat{f^B}(x) = \left[\frac{1}{B} \sum_{b=1}^{B} T_b(x) \right]$$

where $[r]$ denotes the nearest integer of a real value r.

4.4.3 Data Imbalance

With random forest regressor, we deal with imbalanced datasets in two ways: re-sampling and sample weights. As discussed in 4.3.3, we used random over-sampling with random forest regressor the same way as with random forest classifier. Unlike with random forest classifier where we can give class weight to each class, with random forest regressor we give weights to examples since there is no "class". We give each example a weight inversely proportional to its class frequency in the training set.

4.5 Combination of Classification and Regression

Random forest classifier, when trained with over-sampled training set, tends to increase the overall error rate. Random forest regressor, on the other hand, minimizes overall error rate, i.e, the root mean squared error (Chai and Draxler [2014]). Combination of a random forest classifier and regressor results in a model with low error rate and better prediction of the minority class or classes. In this work, we combine a forest classifier and a random forest regressor to form a new model. Output of the combined model is the average of output random forest classifier and real-valued output of random forest regressor, then rounded to the nearest integer.

Chapter 5

Results

In this chapter, we present the numerical results obtained on the two datasets described in Chapter 3, using random forest classifier, random forest regressor and a combination of both, as described in Chapter 4. The chapter is organized as follows. First, in Section 5.1, we discuss the different metrics that are used to evaluate the performance of the models. Results of classification and regression are presented in Sections 5.2 and 5.3, respectively. Section 5.4 presents results derived from the combination of classification and regression models. In each of these three sections, we first highlight the best results, and next present detailed analyses of the various techniques applied, i.e., weighting and resampling. Each section is concluded with a summary of the results provided by all the discussed techniques.

5.1 Performance Metrics

In a multi-class classification problem, we have a number of choices to evaluate the performance. A summary of metrics for classification can be found in Sokolova and Lapalme [2009], among them average accuracy, precision, recall and F-score. Metrics can be calculated in micro- or macro-averaging way. In micro-averaging way, the metric is averaged over all test samples while in macro-averaging, it is averaged within each class, and the class averages are then averaged. As a result, macro-averaging treats all classes equally while micro-averaging favors bigger classes.

As seen in Chapter 3, datasets are highly imbalanced as in the majority of time, people feel and rate themselves more frequently as fit than as tired. Consequently, *accuracy* is not suitable for our problem as it cannot show the performance for minority classes. In a binary classification problem, the classifier is biased toward samples of the majority class and as a result, may predict most test samples in the majority class (Kotsiantis et al. [2006]). In this way, it may obtain very high accuracy, equal to the percentage of the majority class. Obviously, it is not a good classifier as it fails to predict any samples of the minority class, which in many cases is the class of interest. It is also clear that the issue extends to multi-class classification as well. In our problem, the main objective is to predict fatigue cases, which accounts only for small portions in the context of our datasets.

In addition, KSS values are not arbitrary. In fact, they are ordinal, meaning that the classifier must predict the label as close as possible to the true value, if not exactly. For example, a true KSS value of 8 is better to be predicted as 7 (the error is 1) than as 5 (the error is 3). For this reason, precision and recall would not be sufficient as they differentiate matched to unmatched predictions only (Carterette [2009]). F-score is clearly also not appropriate because it is calculated based on precision and recall (Sokolova and Lapalme [2009]). Moreover, high KSS values of 7, 8 and 9 are particularly interesting in a work environment (Geiger-Brown et al. [2012]), as the ultimate goal is to identify the risks of fatigue in work shifts and potential factors associated with risks (Di Milia et al. [2012]).

In a regression problem, we have a number of metrics to use such as mean absolute error (MAE) and root mean squared error (RMSE). MAE, defined formally in 5.1.1, is the average of absolute differences between predictions and target values. MAE and RMSE are both suitable for regression. RMSE is more sensitive for large error and is used as objective function in Random Forest. On the other hand, MAE is more interpretable and is used as one of the metrics to evaluate our model's performance.

In our specific problem, the MAE is not enough in an imbalanced dataset since large errors in minority classes may have diminished effects to overall error. Because of these limitations to MAE, we decided to define a new set of metrics that can show:

- How close the predictions are to the true value, for each value of KSS.
- How close the true values are to the predicted value, for each value of KSS.

First we define a set of notations

T: set of test samples.

K: set of possible KSS values.

$O_k \subseteq T$, $k \in K$: sets of test samples whose true value is k.

$P_k \subseteq T$, $k \in K$: sets of test samples which are classified as k by the classifier.

$P(t)$, $t \in T$: predicted value by the classifier for test sample t

$O(t)$, $t \in T$: true value of test sample t.

We define two sets of metrics to evaluate the performance on each class and one combined metric for the overall error.

5.1.1 Mean Absolute Error

We define the mean absolute error as mean of absolute different between the predicted class and true class over the whole test set. Formally, MAE is defined as follows:

$$MAE = \frac{\sum_{x \in T} |P(x) - O(x)|}{|T|}$$

5.1.2 Class Mean Absolute Error

The CMAE metrics are used to measure the ability of a model to predict label as close as possible to true classes. Indeed, it calculates the residual for every data point, taking the absolute value of each so that negative and positive residuals do not cancel out. CMAE is then the average of

all these residuals. In other words, CMAE describes the typical magnitude of the residuals for a given class.

The analytical expression is as follows:

$$\text{CMAE}_k = \frac{\sum_{x \in O_k} |P(x) - O(x)|}{|O_k|} \qquad k \in K.$$

5.1.3 Class Precision

The CP metrics are used to measure the average mean absolute error for a predicted value. Formally, CP is defined as follows:

$$\text{CP}_k = \frac{\sum_{x \in P_k} |P(x) - O(x)|}{|P_k|} \qquad k \in K.$$

5.2 Classification Results

In this section, we present results of the classification task with two datasets and two techniques to deal with imbalance datasets. Following the discussion in Chapter 4, we used random forest implemented in *scikit-learn* Pedregosa et al. [2011] to perform classification. Hyper-parameters were searched using grid-search with 10-fold cross-validation. The parameter grid is provided in Table 5.1. In this grid, a list of possible values is provided for each parameter. The grid-search procedure iterates over all possible combinations of parameter candidates provided by the parameter grid and uses ten-fold cross-validation to evaluate the performance. With ten-fold cross-validation, it first splits the dataset into ten different groups. Then, for each of ten groups, it uses the group as the test set and the nine remaining ones as training set. After ten validations, the scores are averaged and the parameter set with lowest score is selected.

Table 5.1 Parameter grid of random forest

Parameter	Value	Description
criterion	entropy, gini	Criteria for splitting at each node, also called impurity. The library will calculate both criteria and choose the best one.
min_samples_split	2, 5	If the number of samples in a node does not exceed this number, no splitting will be considered (i.e., the node becomes a leaf)
class_weight	None, *balanced*	Oversampling weight of classes. When set to *balanced*, each sample will be given weights inversely proportional to its class's frequency in the dataset. The *balanced* option will be experimented separately in a section.
max_depth	20, None	Maximum depth of the tree. This range allows the training process to choose between a number of possible maximum depths. This parameter is used to prevent over-fitting of the tree.
min_samples_leaf	3	Minimum number of samples in a leaf
max_leaf_nodes	250	Maximum number of leaves in a tree
bootstrap	True	Allows the bootstrapping (bagging) technique in the forest.
oob_score	True	Enables out-of-bag score, so that we can get the score after training.
n_estimators	200	Number of estimators (trees) in the forest
max_features	*auto*	Maximum number of features to consider for each node. 'auto' lets the forest use default value, which is the square root of total number of features in the training dataset.

5.2.1 Highlights

In this section, we highlight the most salient classification results for each dataset, in terms of the errors on the predicted KSS. Random forest is used to build a KSS predictive model for classification.

We present the results obtained with the 19 most relevant features selected by feature importance ranking in random forest, as described in Section 3.4. Ten runs were conducted for each dataset and performance metrics were averaged. Best results were obtained with DS2 with no class weights nor re-sampling techniques (i.e., the *original* dataset). We present results with original

dataset in this subsection, and then results with class weights and re-sampling in the next two subsections.

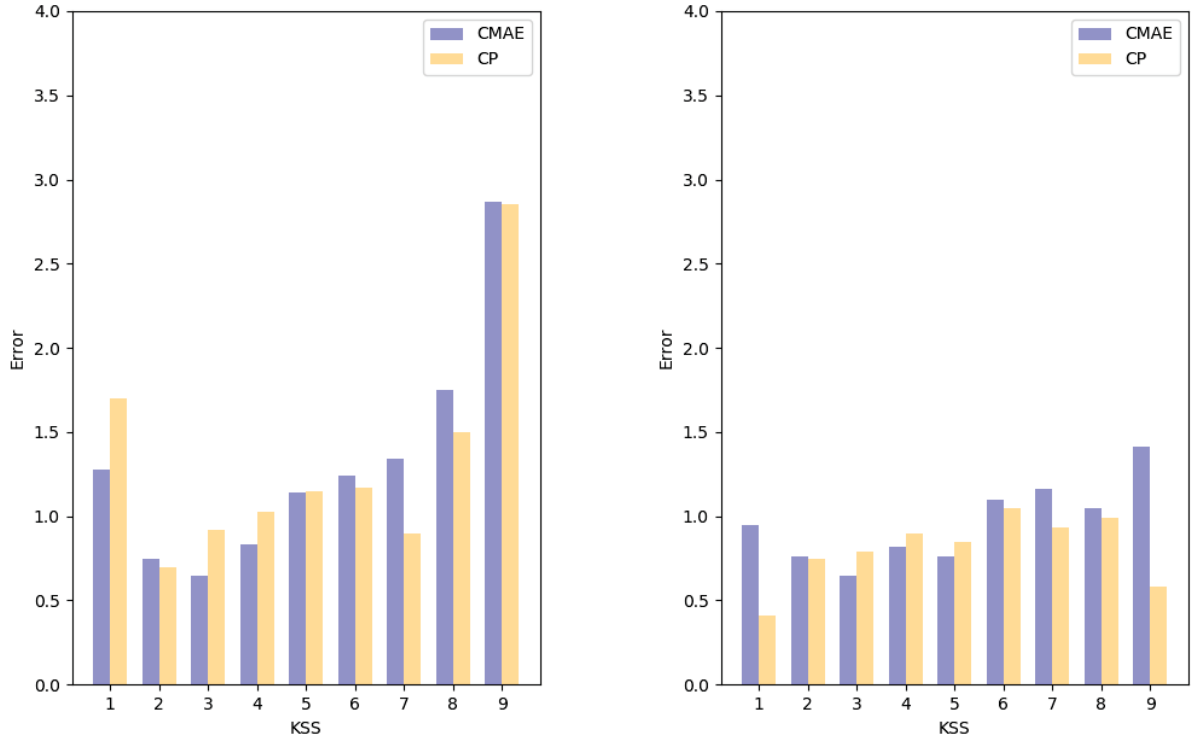
The random forest classifier performs well on both datasets with MAE of 1.02 and 0.88 on DS1 and DS2, respectively. This means that, in general, the model predicts the KSS value with an error of one unit or less. Since KSS is a subjective metric, the model presents promising overall prediction.

We summarize the results obtained with the CMAE and CP metrics, for both datasets, in Figure 5.1. In DS1, CP values were very high for classes 1 and 9. In DS2, CP values were under 1.0 for most classes except class 6 and class 8. It means the classifier had great difficulty predicting 1 and 9 correctly, most likely due to the small size of DS1. The CMAE values were of the same order of magnitude for all classes on the DS2, but high on classes 1 and 9 of DS1. On both datasets, CMAE values showed the same trend: lower values on classes 2, 3, 4 and 5; high on classes 1 and 9 (i.e., rare classes).

Table 5.2 shows comparisons of metrics by classification model on two datasets and techniques applied. The model had better metrics with DS2. With regards to the techniques applied, no single technique had clear advantage over others. Best MAE was obtained by *original* and *class weights*. Best average CMAE and average CP were obtained by and *oversampling* and *original*, respectively. At the end, when it comes to choosing which techniques to apply, it depends on which metrics are the most important ones.

Table 5.2 Metrics comparison, random forest classifier

No	Metric	DS1			DS2		
		Original	Balanced	Sampling	Original	Balanced	Sampling
1	Mean Absolute Error	1.02	1.02	1.04	**0.88**	0.88	0.90
2	Average CMAE	1.32	1.11	1.10	0.96	0.89	**0.88**
3	Average CP	1.32	1.12	1.18	**0.81**	0.91	0.96



(a) DS1

(b) DS2

Figure 5.1: CMAE and CP values by random forest classifier, original datasets

5.2.2 Detailed Performance on Original Datasets

In this subsection, results of model with original data are presented. Distributions of the true values are shown in Figures 5.2 and 5.3. In these figures, each column represents a possible predicted KSS value and indicates the percentage of samples with its corresponding true value. The data presented in each figure were normalized column-wise, i.e., numbers on each column sum to 100%.

Performance on DS1

On DS1, best parameters found by grid-search are in Table 5.3(a). The only different hyper-parameter found, compared to those with DS2 on Table 5.3(b), is *max_depth* = 12. This is expected as training size of DS1 is much smaller than that of DS2.

Table 5.3 Best parameters of random forest classifier found by grid-search

(a) DS1

No	Parameter	Value
1	criterion	entropy
2	min_samples_split	2
3	class_weight	None
4	max_depth	12
5	min_samples_leaf	2
6	n_estimators	200
7	max_features	*auto*

(b) DS2

No	Parameter	Value
1	criterion	entropy
2	min_samples_split	2
3	class_weight	None
4	max_depth	None
5	min_samples_leaf	2
6	n_estimators	200
7	max_features	*auto*

Figure 5.2 shows that the distributions of true values peak at predicted value, marked blue and gradually decreases as true value moves away from predicted value. This pattern, however, has exception on predicted values of class 9. All samples that classifier predicts as class 9 are of true value 6 and 8. The rates of true values in range of $p_k \pm 1$ are also high for most classes. 84.5% of samples predicted as class 2 have true value in range of 2 ± 1 while those numbers for class 3 and class 4 are 80.2% and 75.8% respectively. In the side of high predicted KSS values, we have 63.3% and 80.2% for classes 6 and 7. Unlike on DS2, only 58% samples with predicted KSS of 8 have true value in range of 8 ± 1 due to the fact that the classifier rarely predicts class 9. This is expected as the result of imbalanced dataset where class 9 represents only a very small fraction of 1.59%.

One of the key objectives of the study is to identify potential factors leading to the fatigue, and random forest provides a very useful tool for that purpose: the feature importance. Feature importance given by random forest is shown on Table 5.4(a). As expected, the most influential factors to level of fatigue is time awake (*time_awake_less_nap*). The second group of factors, the sleep-related features, provides significant contribution to the prediction. Other important variables include time of day (*time_of_day_sine* and *time_of_day_cosine*) and day of year (*day_of_year_sine* and *day_of_year_cosine*). It is surprising to find out that variables involving *current* work shift do not have much influence on the classifier. *time_since_start_of_shift* and *workduration* stand at 17^{th} and 15^{th} places respectively with only around 3% importance.

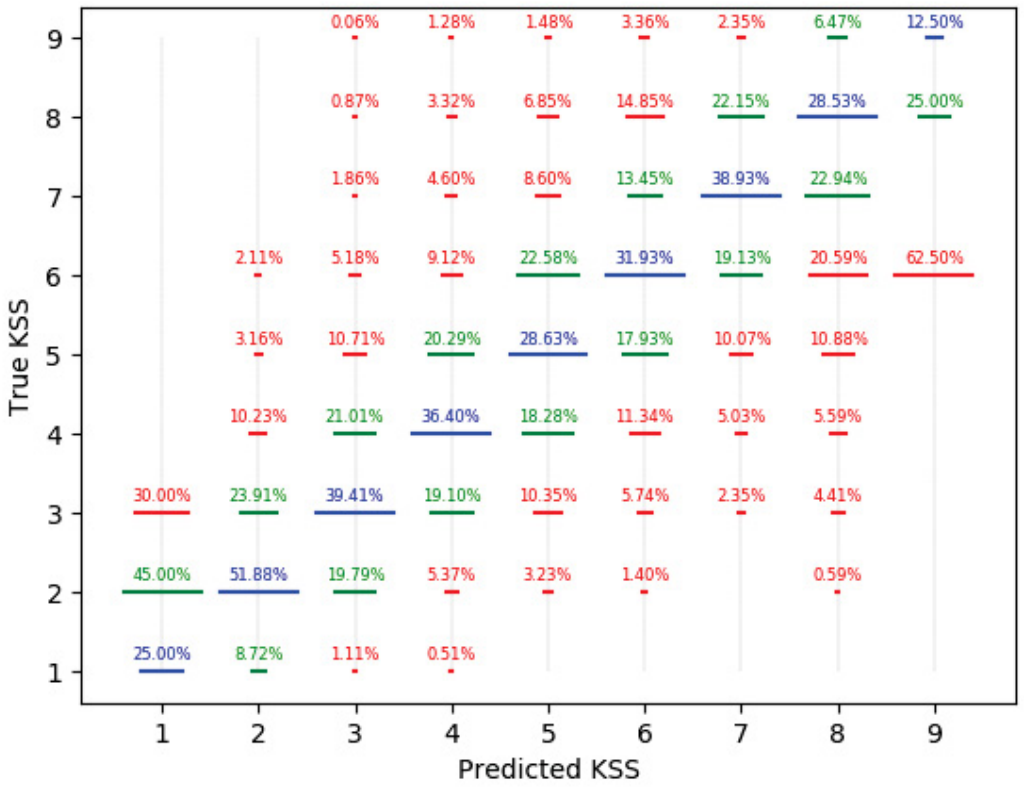


Figure 5.2: Normalized confusion matrix, original DS1

Demographic variables fairly contribute to the classification: *age* comes at 9^{th} place while *average_sleep_hours*, *ho_chronotype* and *isi* sequentially take places from 12^{th} to 14^{th}.

Performance on DS2

In DS2, the best parameters found by grid-search are reported in Table 5.3(b). After grid search, the best parameters were used to refit the random forest to perform classification on the test set, because they were found based on nine folds of the training set only.

Figure 5.3 shows the same patterns as with DS1. This confirms the good performance of the model, with average CP values below 1. For a predicted KSS value p_k, a majority of test samples has true value in range of $p_k \pm 1$. Correctly classified samples are marked with blue while samples with predicted value of range ± 1 of true value are marked with green. In the area of high predicted KSS ($p_k = 7, 8, 9$), approximately 81.6% of samples predicted at 7 have true

KSS in range 7 ± 1 while the percentages for 8 and 9 are 76.4% and 92.7% respectively. On the other side of the KSS range, we have similar results with 80% and 78.5% for predicted KSS of 4 and 5 respectively. Lowest percentage are at predicted KSS of 6, with 70.9%, which suggests that the KSS value of 6 is the border line between two areas: fresh (1 to 5) and tired (7 to 9).

For every predicted KSS value p_k, the percentage of samples with true value t_k decreases as t_k moves away from p_k. The number of samples where $|t_k - p_k| \geq 3$ are very small. In the distribution of predicted value of 9, lowest true value is 5 with only 2.4%. For a predicted value of 7 and 8, the true values can be as low as 2. This phenomenon, however, does not happen very often as seen by the thinner tail of the plot of 7, 8, and 9. In the area of low predicted KSS values (1,2 and 3), we also have very small percentages of high true values.

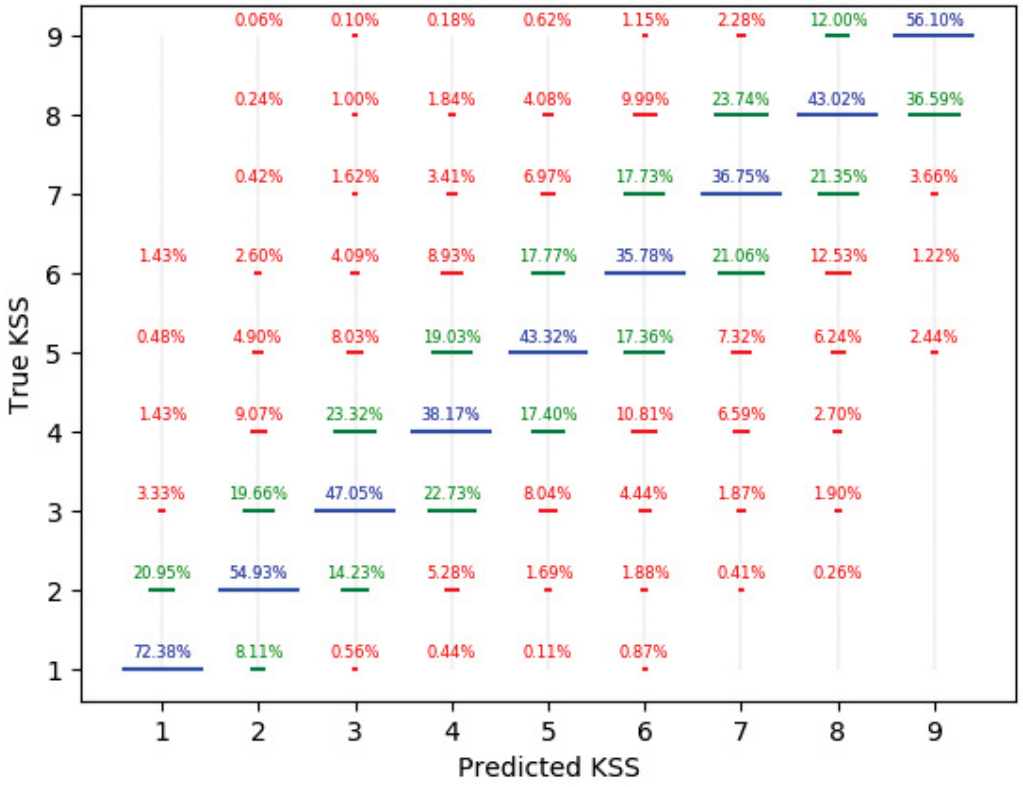


Figure 5.3: Normalized confusion matrix, original DS2

Feature importance given by random forest is shown on Table 5.4(b).

Table 5.4 Ranking of feature importance by random forest, original datasets

(a) DS1

No	Importance	Feature
1	9.19%	time_awake_less_nap
2	7.34%	time_since_end_of_shift
3	6.83%	day_of_year_sine
4	6.77%	sleep_time_72h
5	6.72%	time_of_day_sine
6	6.47%	sleep_time_24h
7	6.40%	time_in_bed
8	6.28%	sleep_time_48h
9	6.10%	age
10	6.00%	day_of_year_cosine
11	5.95%	time_of_day_cosine
12	5.58%	average_sleep_hours
13	4.26%	ho_chronotype
14	3.78%	questionnaireno
15	3.64%	isi
16	3.62%	time_since_start_of_shift
17	3.56%	work_duration
18	0.78%	nap_duration
19	0.73%	time_awake_since_last_nap

(b) DS2

No	Importance	Feature
1	9.36%	time_awake_less_nap
2	6.90%	time_since_end_of_shift
3	6.59%	time_of_day_cosine
4	6.55%	day_of_year_cosine
5	6.53%	sleep_time_72h
6	6.49%	time_in_bed
7	6.46%	day_of_year_sine
8	6.45%	sleep_time_48h
9	6.45%	time_of_day_sine
10	6.38%	sleep_time_24h
11	5.30%	age
12	4.96%	average_sleep_hours
13	4.78%	ho_chronotype
14	4.59%	isi
15	4.50%	questionnaireno
16	3.16%	work_duration
17	2.85%	time_since_start_of_shift
18	0.88%	time_awake_since_last_nap
19	0.83%	nap_duration

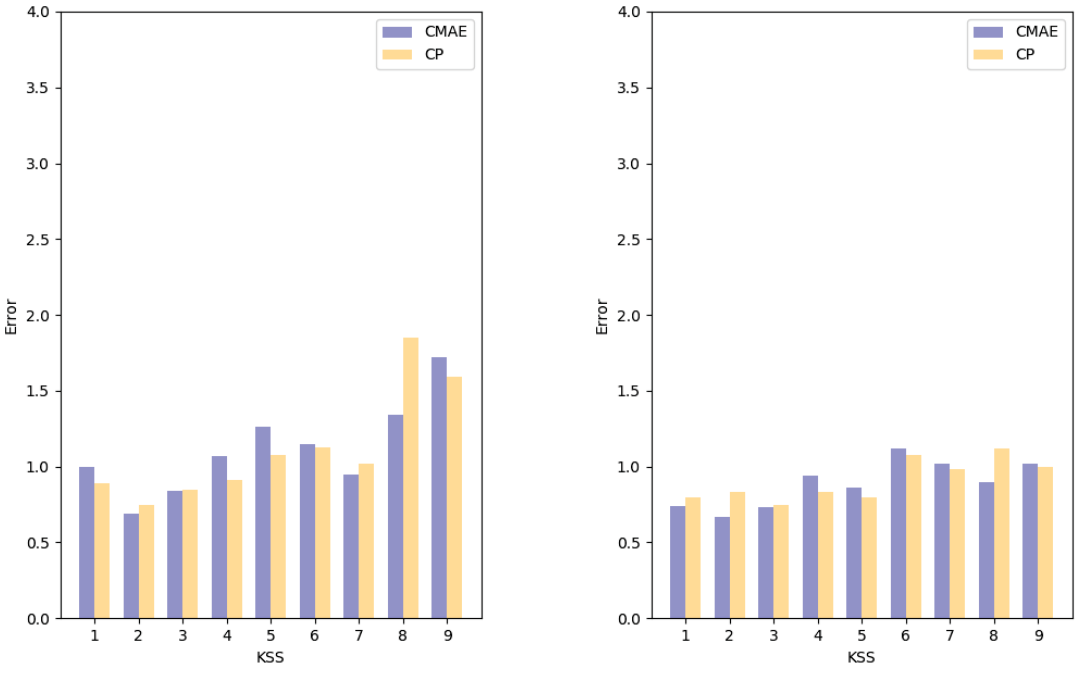
We can see the top influential factors are *time_awake_less_nap* and time of day (*time_of_day_sine* and *time_of_day_cosine*). *time_since_end_of_shift* appears at 2^{nd} place which suggests strong effect of the previous work shift to level of fatigue. Other than that, there is no significant difference in feature importance rankings compared to that of DS2.

It is worth noting that feature importance must be interpreted in presence of correlated features. In the presence of one or more groups of correlated features, feature importance of each individual feature tends to be shared with others from the same group, although not necessarily evenly. One important implication is that just because a feature has low importance does not mean that it is noisy (Genuer et al. [2010]).

5.2.3 Detailed Performance with Class Weights

In this section, the results are shown with models trained with *class_weights = balanced*, meaning that each sample in the training set is given a weight. The weights are inversely proportional to the frequency of the label on the training set. Setting class weights to *balanced* helps reduce error on minority class, particularly with imbalanced datasets. The classifier is now able to

predict all KSS values on both datasets. Both CMAE and CP metrics are more stable in DS2 than in DS1. This, in combination with results in previous section, shows that imbalance in DS1 is more severe than in DS2. Overall, both models show better performance on lower KSS values than the area of higher values. In particular, the model on DS1 has large errors in with KSS of 8 and 9 while model with DS2 performs well.



(a) DS1 (b) DS2

Figure 5.4: CMAE and CP with balanced class weights

Performance on DS1

In this dataset, class weights help prediction of rare values, class 1 and 9. In Figure 5.4(a), CP values of 8 and 9 are much higher than those of other classes. This is the expected effect of setting balanced class weights to heavily imbalanced dataset. The same effect does not happen with DS2, as samples of class 8 and class 9 are not rare.

Feature rankings (Table 5.5(a)) are similar to those of DS1 without weights, as well as those of DS2 with balanced class weight. This suggests that class weighting has very little impact on

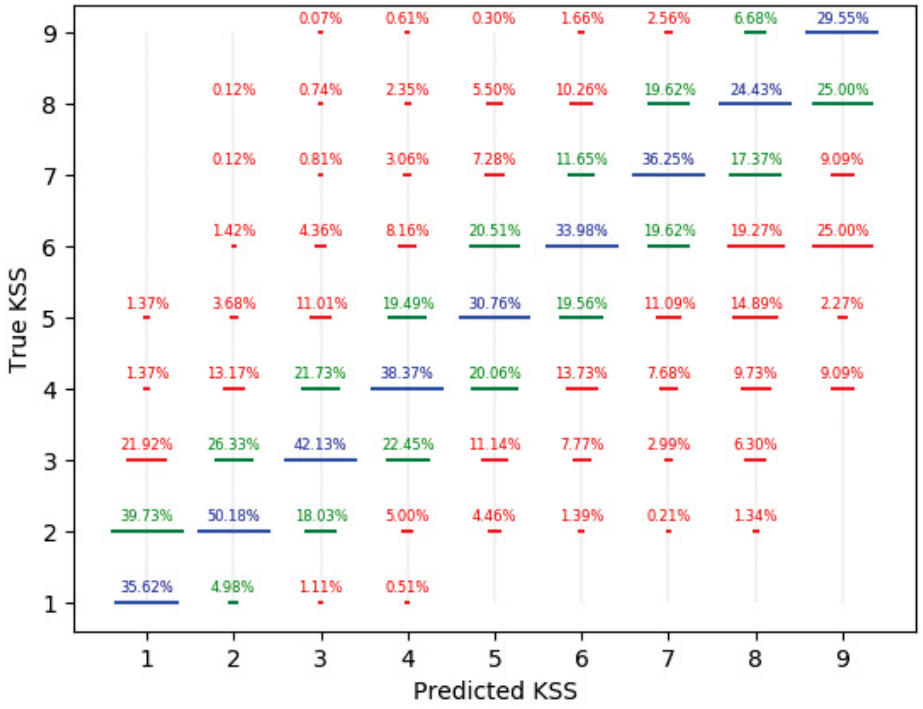


Figure 5.5: Normalized confusion matrix, DS1, with balanced class weights

feature importance of random forest.

Performance on DS2

The results are similar to those of DS2 without class weights. In all predicted KSS values, distribution of true values peak at predicted value as represented by blue segments on Figure 5.6. The rate of true values in range of $p_k \pm 1$, as represented by blue and green segments, are very good for all classes. 80.2% of samples with predicted value of 1 have actual true value in range 1 ± 1. Similar numbers are seen for classes 2, 3, 4 and 5 with minimum percentage of 80.3% (class 2) and maximum of 85.6% (class 3).

In the other side, high percentages of true values within range of $p_k \pm 1$ for all predicted $p_k = 7, 8, 9$ are seen. This is very promising, as we are trying to identify samples with high risk of fatigue, we predict 9 (very tired) when 81.2% of them actually have true value of 8 or 9. The numbers for predicted values 7 and 8 are also high, at 78.3% and 70.9% respectively.

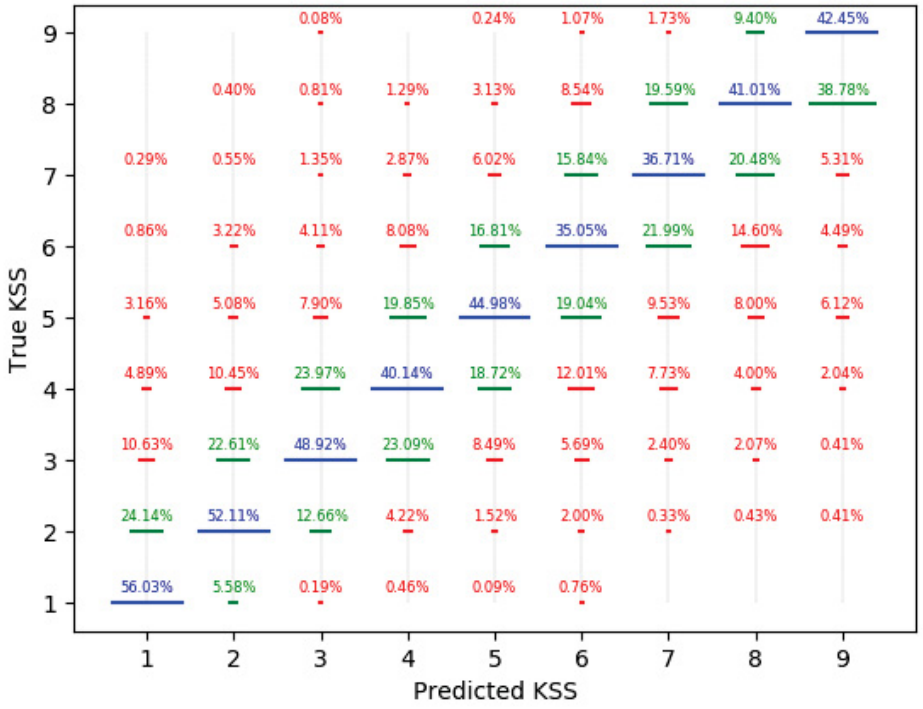


Figure 5.6: Normalized confusion matrix, DS2, with balanced class weights

Similar to results on DS2 without class weights, the distributions have thin tails which means that misclassification rate decrease rapidly as true value moves away from predicted value.

In the feature ranking (Table 5.5(b)), we see consistent rankings with those of results without class weights. In comparison to 5.4(b), individual feature may have slight changes in terms of ranking but the importance is more or less the same, with the difference of just fractions of a percent.

5.2.4 Detailed Performance with Random Over-sampling

In this section, we present the results obtained with random over-sampling. With this technique, we resample the minority classes in the training set with replacement so that all classes will have equal number of samples. CMAE and CP plots (Figure 5.7) shows the effect of generating samples to minority classes. For both datasets, the general trend is that we have lower CMAEs

Table 5.5 Ranking of feature importance by random forest classifier, with balanced class weights

(a) DS1

No	Importance	Feature
1	9.00%	time_awake_less_nap
2	7.67%	day_of_year_sine
3	7.37%	sleep_time_72h
4	7.23%	time_since_end_of_shift
5	6.80%	time_of_day_sine
6	6.63%	sleep_time_24h
7	6.30%	time_in_bed
8	6.19%	sleep_time_48h
9	5.92%	age
10	5.59%	day_of_year_cosine
11	5.43%	time_of_day_cosine
12	4.87%	average_sleep_hours
13	4.12%	questionnaireno
14	4.00%	isi
15	3.91%	time_since_start_of_shift
16	3.85%	ho_chronotype
17	3.34%	work_duration
18	1.05%	time_awake_since_last_nap
19	0.71%	nap_duration

(b) DS2

No	Importance	Feature
1	9.61%	time_awake_less_nap
2	6.81%	day_of_year_cosine
3	6.52%	day_of_year_sine
4	6.48%	time_of_day_cosine
5	6.45%	sleep_time_24h
6	6.26%	age
7	6.17%	sleep_time_72h
8	6.16%	time_in_bed
9	6.14%	sleep_time_48h
10	6.09%	time_of_day_sine
11	6.08%	time_since_end_of_shift
12	5.07%	questionnaireno
13	5.04%	average_sleep_hours
14	4.89%	ho_chronotype
15	4.79%	isi
16	2.88%	time_since_start_of_shift
17	2.83%	work_duration
18	0.90%	time_awake_since_last_nap
19	0.84%	nap_duration

and higher CPs compared to results with original datasets. The results are expected as we provide more samples to minority classes, the classifier is biased towards those classes. As a result, it predicts more samples to be in those classes and increases the class precision values.

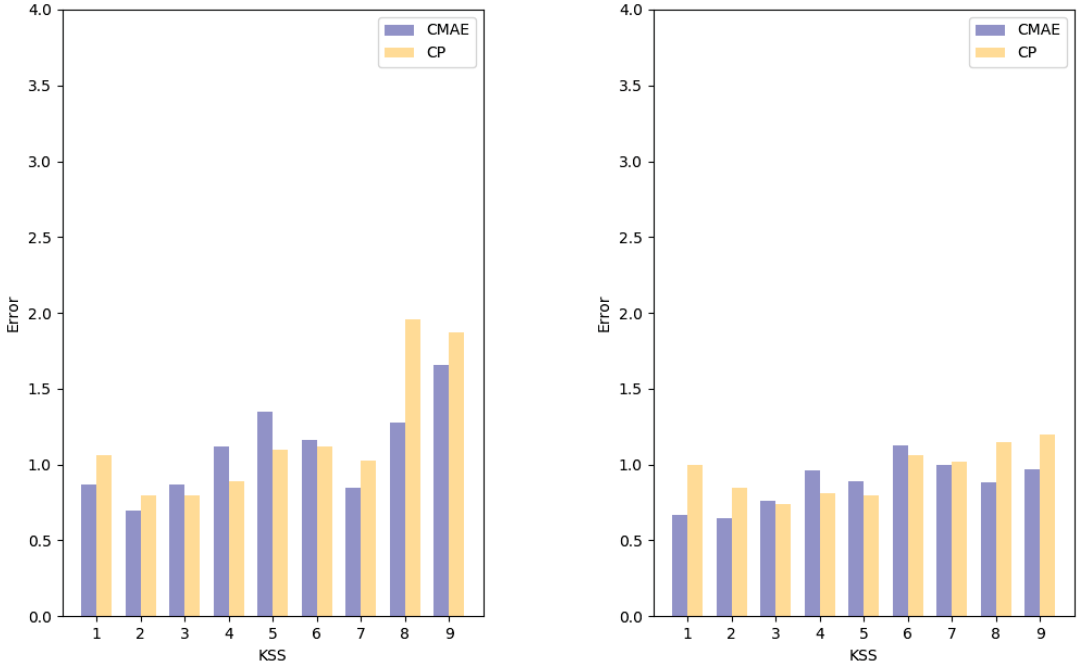
Performance on DS1

In this dataset, random oversampling helps prediction of rare values, class 1 and 9. This result, expectedly, comes with a compromise: high CP values on oversampled classes.

In the feature ranking (Table 5.4(b)), as with DS2, we see almost identical rankings to those of results with balanced class weights (Table 5.5(b)), both in terms of rankings and percentage.

Performance on DS2

The results are similar to those of DS2 presented previously. In all predicted KSS values, distribution of true values peak at predicted value as represented by blue segments on Figure 5.9, except for class 9. The rate of true values in range of $p_k \pm 1$, as represented by blue and green segments, are very similar to those with class weights. 76.6% of samples with predicted value of 1 have actual true value in range 1 ± 1. Similar numbers are seen for classes 2,3,4 and 5 with minimum percentage of 79.4% (class 2) and maximum of 85.7% (class 3). In the other side,



(a) DS1 (b) DS2

Figure 5.7: CMAE and CP with random oversampling

high percentages of true values within range of $p_k \pm 1$ for all predicted $p_k = 7, 8, 9$ are seen. Of all samples predicted 9 (very tired), 76.5% actually have true value of 8 or 9. Similar to results on DS2 with class weights, the distributions have thin tails which means that misclassification rate decrease rapidly as true value moves away from predicted value.

In the feature importance rankings (Table 5.6(b)), we can see almost identical rankings to those of results with balanced class weights (Table 5.5(b)), both in terms of rankings and percentage.

5.2.5 Concluding Remarks

In this section, we present results of the classification with two datasets and three techniques. Best results were obtained with original DS2. Results with DS2 show that the model predicts KSS values with error of ± 1 unit, with around 80% confidence. In all experiments, the classifiers consistently identify *time_awake_less_nap*, *time_of_day_dec*, *sleep_time_24h*, *time_in_bed*, and *age*

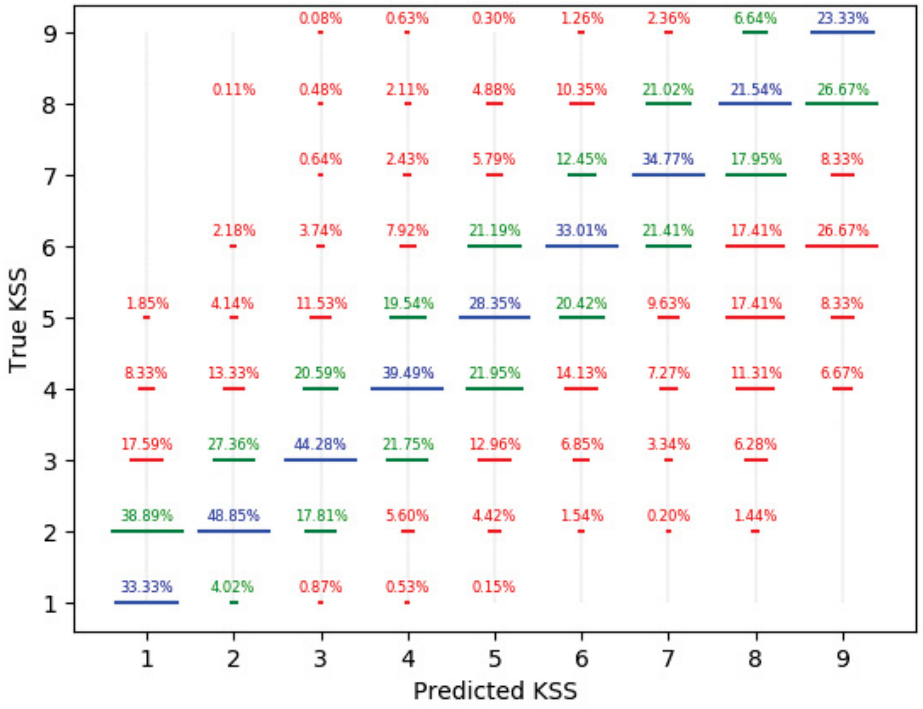


Figure 5.8: Normalized confusion matrix, DS1, with random oversampling

as the most influential features. The rankings of feature importance are robust to class weights and sampling technique, but are dataset dependent.

The results from two datasets suggest that the size of the dataset played a very important role in classification performance. Classification model trained with DS2 has better mean absolute errors as well as average CMAEs and CPs. It is worth noting that training processes for two datasets, including hyper-parameters search, are totally independent. Thus, it is expected that even better results would be obtained with larger datasets, in particular the ones with larger ratios of fatigued people.

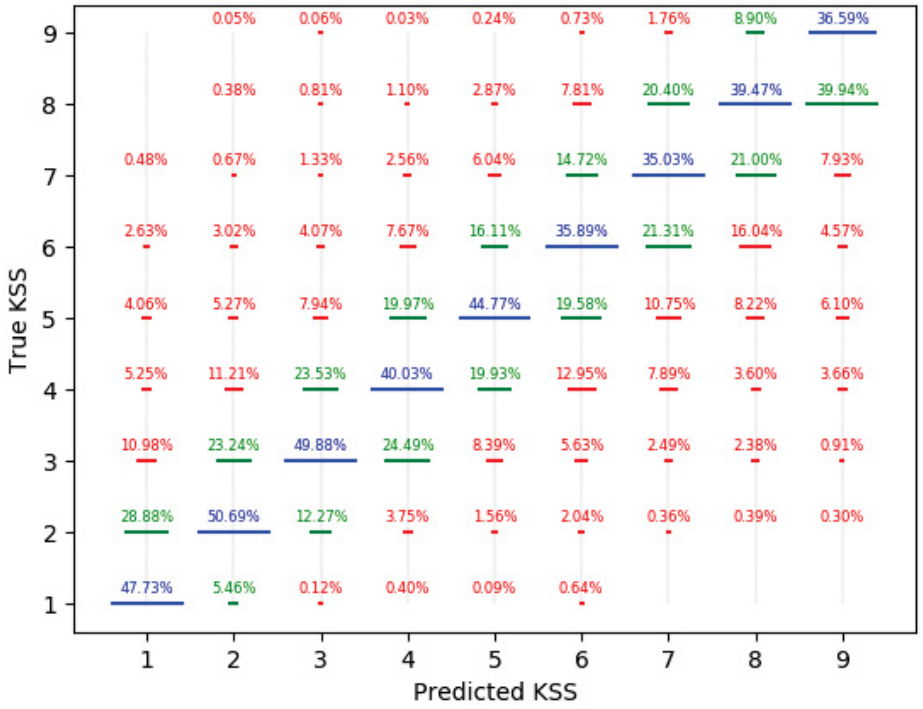


Figure 5.9: Normalized confusion matrix, DS2, with random oversampling

Table 5.6 Ranking of feature importance by random forest, with oversampling

(a) DS1

No	Importance	Feature
1	8.66%	time_awake_less_nap
2	7.93%	day_of_year_sine
3	7.72%	sleep_time_72h
4	6.98%	time_since_end_of_shift
5	6.72%	sleep_time_24h
6	6.68%	time_of_day_sine
7	6.38%	sleep_time_48h
8	6.18%	age
9	6.12%	time_in_bed
10	5.58%	time_of_day_cosine
11	5.48%	day_of_year_cosine
12	4.99%	average_sleep_hours
13	3.98%	questionnaireno
14	3.86%	ho_chronotype
15	3.74%	time_since_start_of_shift
16	3.49%	isi
17	3.48%	work_duration
18	1.15%	time_awake_since_last_nap
19	0.89%	nap_duration

(b) DS2

No	Importance	Feature
1	9.21%	time_awake_less_nap
2	6.84%	day_of_year_cosine
3	6.62%	sleep_time_24h
4	6.56%	time_of_day_cosine
5	6.52%	time_of_day_sine
6	6.36%	time_since_end_of_shift
7	6.34%	sleep_time_72h
8	6.31%	day_of_year_sine
9	6.23%	sleep_time_48h
10	5.94%	time_in_bed
11	5.75%	age
12	5.16%	questionnaireno
13	4.80%	ho_chronotype
14	4.78%	average_sleep_hours
15	4.76%	isi
16	3.01%	work_duration
17	2.83%	time_since_start_of_shift
18	1.09%	time_awake_since_last_nap
19	0.89%	nap_duration

5.3 Regression Results

In this section, we present results of the regression task with two datasets and three techniques. Following the discussion in Chapter 4, we used random forest implemented in *scikit-learn* (Pedregosa et al. [2011]) to perform regression. Outputs of random forest regressor, which are of real-valued type, are rounded to the nearest integer.

Hyper-parameters were searched the exact same way that we do in Chapter 5.2. The parameter grid is provided in Table 5.1.

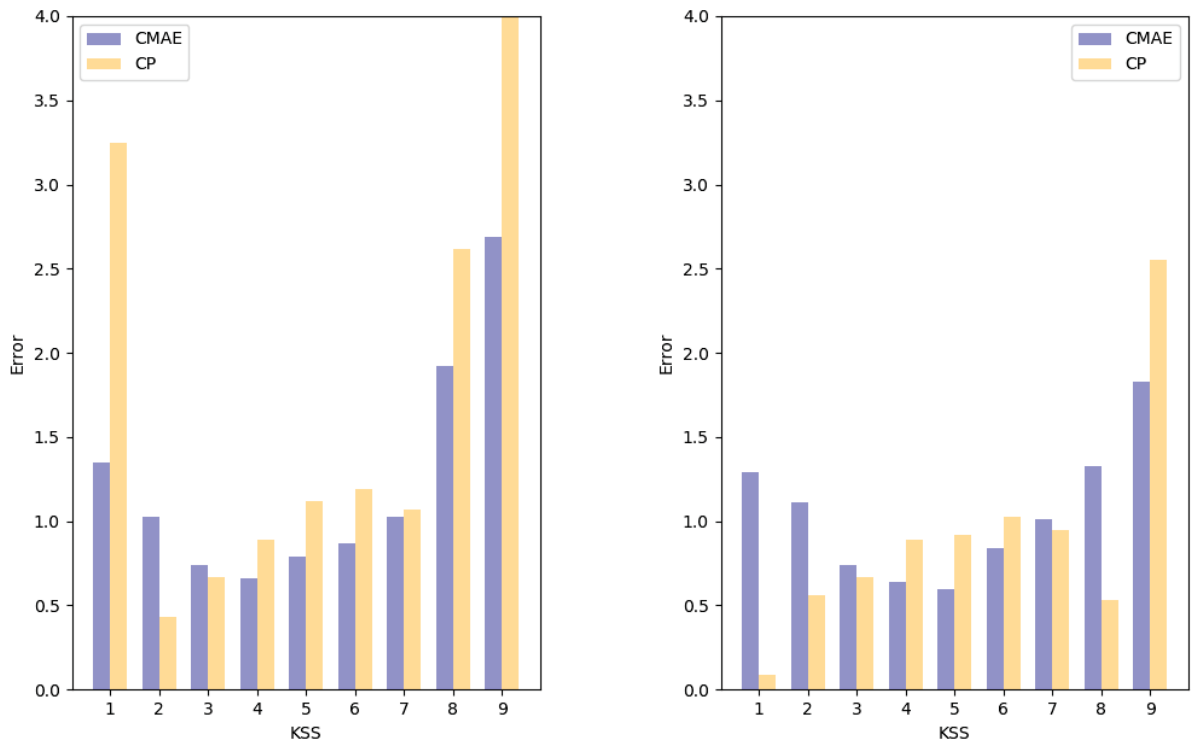
5.3.1 Highlights

In this section, we highlight the most salient results for each dataset, in terms of the errors on the predicted. Random forest regressor was used to build a KSS predictive model for regression.

We present the results obtained with the 19 most relevant features selected by feature importance ranking in random forest, as described in Section 3.4. Ten runs were conducted for each dataset and performance metrics were averaged. We present results with original datasets in this subsection, and then results with class weights and re-sampling in the next two subsections.

The random forest regressor performs well on both datasets. On DS1, the model also scores very close MAE for three scenarios: 0.93 with original and balanced class weight, and 0.96 with over-sampling. On DS2, the model scores MAE of 0.86 for all three scenarios.

In terms of CMAE and CP metrics, models trained with original datasets produce low error in the area of predicted KSS from 2 to 7 which is the area of majority classes. In other KSS values of 1, 8 and 9, the regressors produce high error on both CMAE and CP. This is expected as the regressor tends to bias towards majority classes. In particular, with DS2, the regressor cannot even predict a sample at 1 and 9 as represented by CP values of 4. We present the results in the following order: first with original datasets, followed by class weights and over-sampling.



(a) DS1 (b) DS2

Figure 5.10: CMAE and CP by random forest regressor, original datasets

Table 5.7 Metrics comparison, random forest regressor

No	Metric	DS1			DS2		
		Original	Class weights	Sampling	Original	Class weights	Sampling
1	Mean Absolute Error	0.93	0.93	0.96	**0.86**	0.86	0.86
2	Average CMAE	1.23	1.23	1.18	1.04	1.03	**0.98**
3	Average CP	1.69	1.53	1.21	0.91	0.79	**0.73**

5.3.2 Detailed Performance with the Original Datasets

In this subsection, we present results obtained with original datasets, i.e., without class weights or over-sampling.

Performance on DS1

In this dataset, the regressor do not make any prediction of 1 or 9 due to the dearth of those labels in training set. At other values, the regressor shows decent performance with approximately 70% of true values ranging in $p_k \pm 1$ for every predicted value p_k of 5,6 and 7.

Performance on DS2

Similar to results of DS2 with random forest classifier, Figure 5.12 shows that the distributions of true KSS values peak at the predicted KSS value, except for 9, as represented by the blue segments on the diagonal. For a predicted KSS value p_k, a majority of test samples has true value in range of $p_k \pm 1$. Correctly classified samples are marked with blue while samples with predicted value of range ± 1 of true value are marked with green. In the area of high predicted KSS ($p_k = 7, 8, 9$), 82.1% of samples predicted at 7 have true KSS in range 7 ± 1 while the percentages for 8 and 9 are 95% and 100% respectively. Again in this section, lowest percentage are at predicted KSS of 6, with 71.9%, which suggests that the KSS value of 6 is on the border line of fresh (1 to 5) and tired (7 to 9) area.

For every predicted KSS value p_k, the percentage of samples with true value t_k decreases as t_k moves away from p_k. The number of samples where $|t_k - p_k| \geq 3$ are very small. In the distribution of predicted value of 9, lowest true value is 8 which means the model produces very high confidence when it predicts a test sample to be 9. For a predicted value of 7 and 8, the true values can be as low as 2 and 3 but such cases are extremely rare. For example, among all samples predicted at 8, only very small number have true value of 5 (0.17%), 4 (0.33%) or 3 (0.33%). In the area of low predicted KSS values, we also have very small percentages of high true values.

Feature importance given by random forest regressor is shown on Tables 5.8(a) and 5.8(b). In both datasets, *questionnaireno* pops up at the first place. Other feature rankings are consistent with what are given by random forest classifier presented in the previous section.

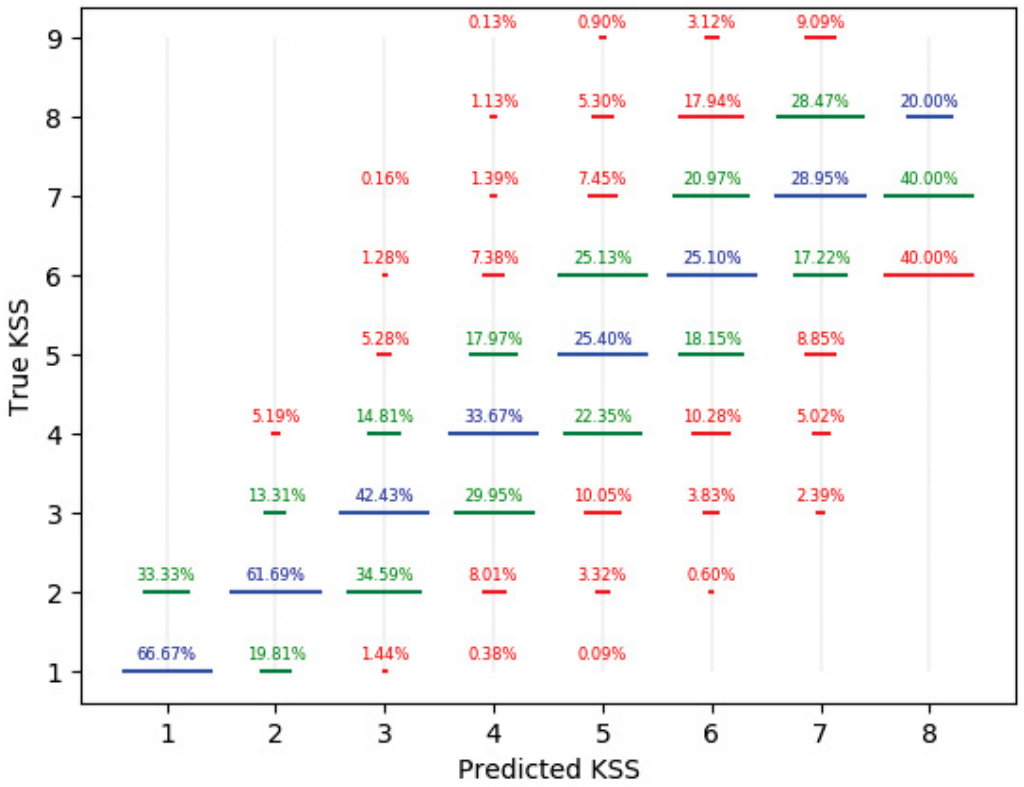


Figure 5.11: Normalized confusion matrix, original DS1 with random forests regressor

5.3.3 Detailed Performance with Class Weights

In this section, the results are shown with models trained with *class_weights* = *balanced*, meaning that each sample in the training set is given a weight. The weights are inversely proportional to the frequency of the label in the training set.

Overall, CMAE and CP values are low on predicted KSS values from 1 to 7 on both datasets as shown on Figure 5.13. In the area of minority classes, high errors are seen on both classes. Particularly, with DS1, the regressor cannot predict any sample of class 9, as represented by high CP value of 4 at predicted KSS of 9 on Figure 5.13(a).

Performance on DS1

Similar to what we have seen with random forest classifier, the class weights cause extreme high

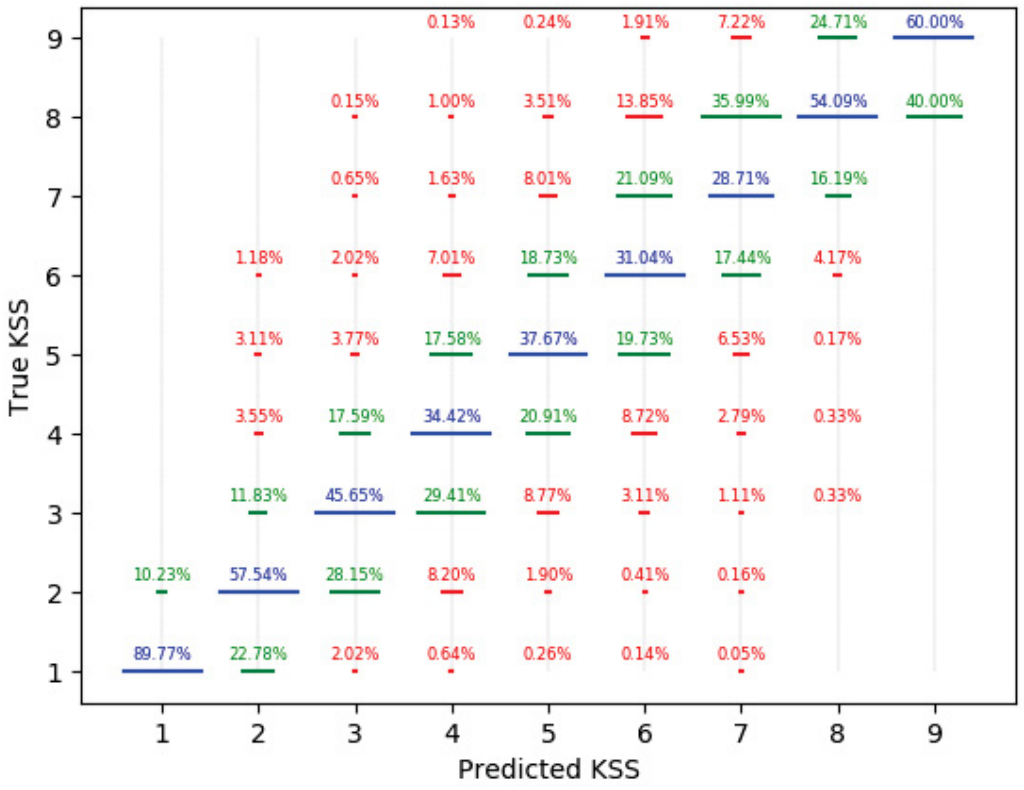


Figure 5.12: Normalized confusion matrix, original DS2 with random forest regressor

error on rare values, particularly 9. Extremely high values on both CMAE and CP of Class 9 suggest that balanced class weights might not work well with extremely imbalanced dataset. Ranking of feature importance in this dataset is almost identical to that of DS2. This suggests that most influential factors are consistent on both datasets.

Performance on DS2

The results are similar to those of DS2 without class weights, except that we get better CP values for predicted KSS of 8 and 9.

Similar to results on DS2 without class weights, the normalized confusion matrix (Figure 5.15) has thin tails which means that misclassification rate decrease rapidly as true value moves away from predicted value.

Table 5.8 Ranking of feature importance by random forest regressor, original datasets

(a) DS1

No	Importance	Feature
1	23.89%	questionnaireno
2	8.48%	age
3	8.47%	time_awake_less_nap
4	7.47%	average_sleep_hours
5	6.77%	time_of_day_sine
6	5.28%	sleep_time_24h
7	5.08%	time_since_end_of_shift
8	4.37%	sleep_time_72h
9	4.30%	day_of_year_cosine
10	4.29%	day_of_year_sine
11	4.27%	time_in_bed
12	4.11%	sleep_time_48h
13	3.22%	isi
14	2.82%	time_of_day_cosine
15	2.60%	ho_chronotype
16	1.86%	time_since_start_of_shift
17	1.58%	work_duration
18	0.61%	nap_duration
19	0.55%	time_awake_since_last_nap

(b) DS2

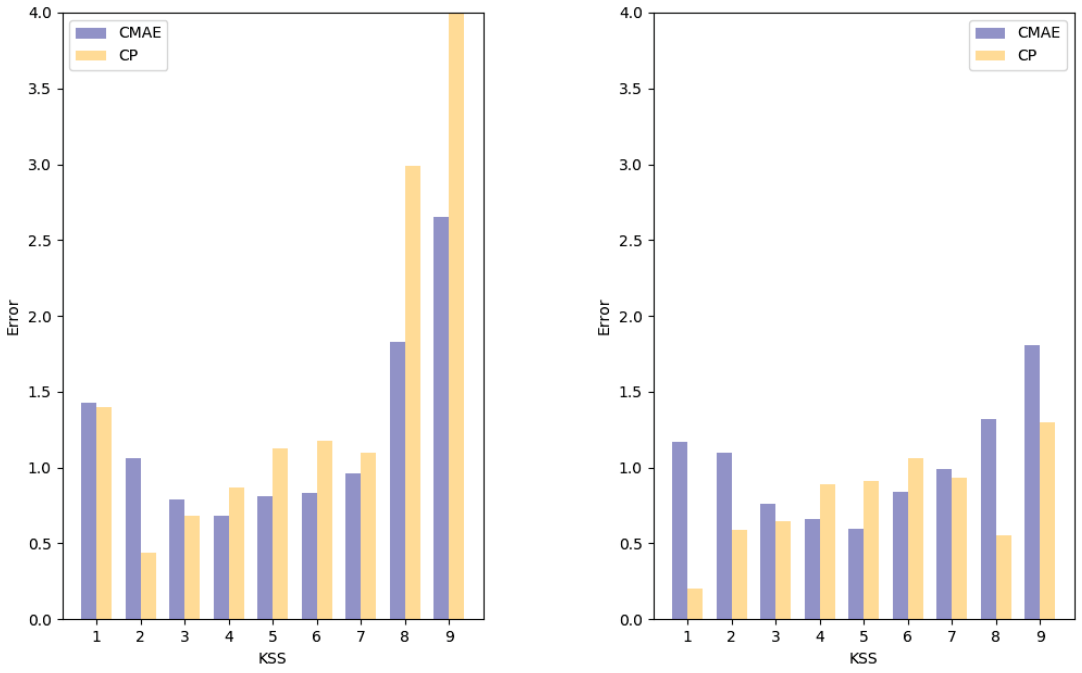
No	Importance	Feature
1	28.25%	questionnaireno
2	11.98%	time_awake_less_nap
3	6.28%	isi
4	6.06%	time_in_bed
5	4.56%	average_sleep_hours
6	4.55%	age
7	4.51%	time_of_day_sine
8	4.10%	day_of_year_cosine
9	3.96%	day_of_year_sine
10	3.92%	sleep_time_72h
11	3.82%	ho_chronotype
12	3.73%	sleep_time_24h
13	3.64%	time_of_day_cosine
14	3.44%	time_since_end_of_shift
15	3.27%	sleep_time_48h
16	1.57%	time_since_start_of_shift
17	1.54%	work_duration
18	0.48%	time_awake_since_last_nap
19	0.36%	nap_duration

In the feature ranking (Table 5.9(b)), we see consistent rankings with those of results without class weights. In comparison to 5.8(b), individual feature may have slight changes in terms of ranking but the importance is more or less the same, with the difference of just fractions of a percent.

5.3.4 Detailed Performance with Random Over-sampling

In this section, results are shown on a model train with over-sampled training data. We used random over-sampling to add samples of minor classes to training data so that we have a balanced dataset.

Overall, over-sampling has almost the same effects as class weights. As shown on Figure 5.16, on DS2, low CP values are obtained on class 8 and 9 without too much compromise on associated CMAE values. On DS1, extremely high values of CP and CMAE of class 9 suggest that over-sampling might not work well with extremely imbalanced dataset, as we have seen with class weights.



(a) DS1 (b) DS2

Figure 5.13: CMAE and CP by random forest regressor with balanced class weights

Performance on DS1

The normalized confusion matrix (Figure 5.17) shows the problematic area of predicted KSS values 8 and 9. For samples predicted as 9, true values are as low as 4 and none is 9. This confirms that over-sampling does not work well with extremely imbalanced dataset.

Performance on DS2

The results are similar to those of DS2 with class weights, except that we get better CP values of class 8 and 9.

Similar to results on DS2 without class weights, the normalized confusion matrix (Figure 5.18) has thin tails which means that misclassification rate decrease rapidly as true value moves away from predicted value.

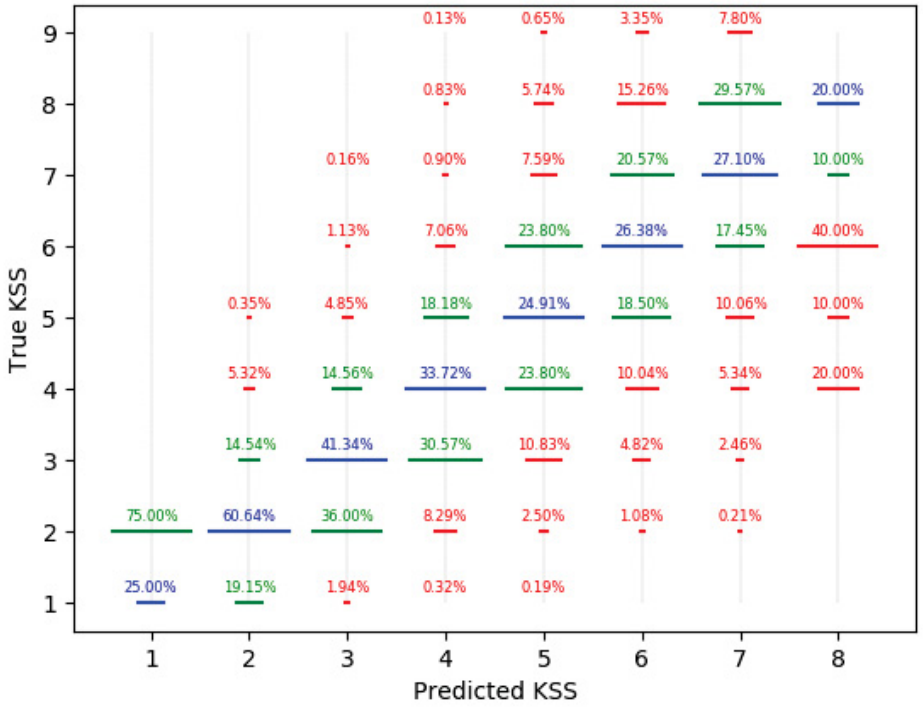


Figure 5.14: Normalized confusion matrix, DS1 with balanced class weights

5.3.5 Concluding Remarks

In this section, we present results of the regression with two datasets and three techniques. All techniques produce very close MAE. Class weights and over-sampling give much smaller CP and CMAE values in the area of high KSS (KSS = 8, 9), with DS2. However, class weights and over-sampling do not work well with DS1. These results suggest that class weights and over-sampling can be considered to deal with imbalanced dataset, but might not work well with extremely imbalanced ones.

In accordance with classification taks, these results shows that the model predicts KSS values with error of ±1 unit, with around 85% confidence. In all experiments, the classifiers consistently identify *questionnaireno*, *time_awake_less_nap*, *time_of_day_sine*, *time_of_day_cosine*, *sleep_time_24h*, *time_in_bed*, and *age* as the most influential features. The rankings of feature importance are robust to class weights and sampling technique, but are dataset dependent.

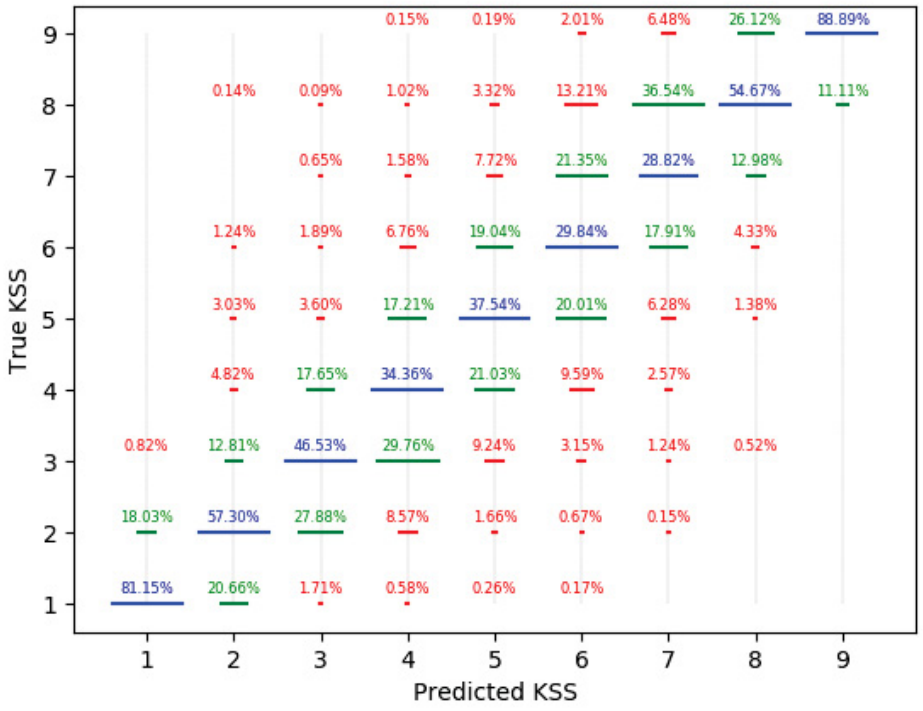


Figure 5.15: Normalized confusion matrix, DS2 with balanced class weights

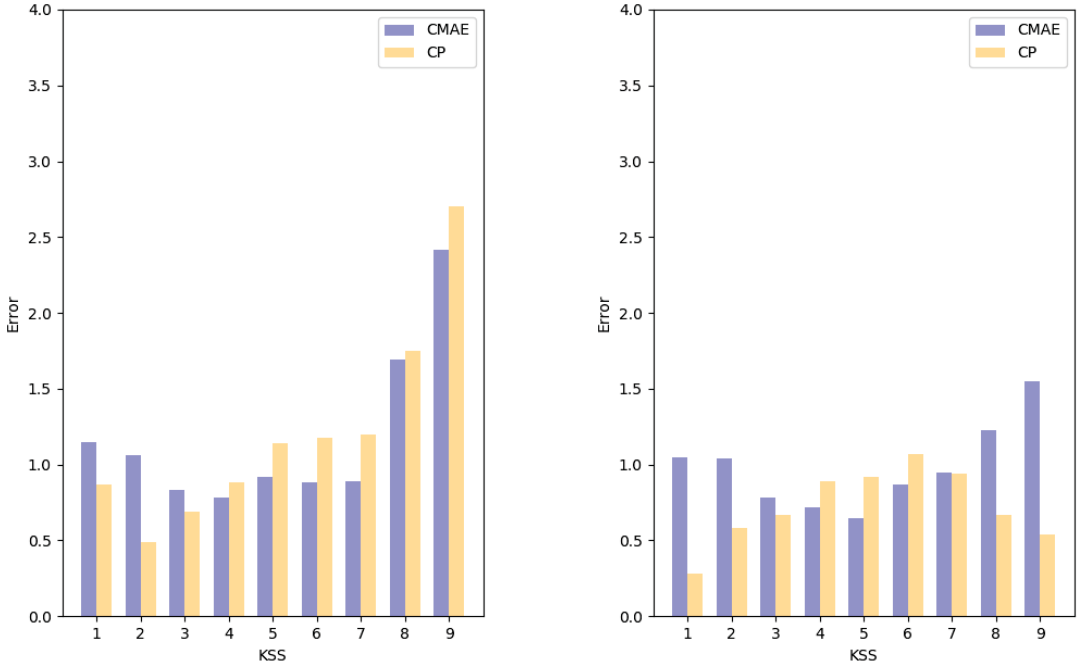
Table 5.9 Ranking of feature importance by random forests regressor with balanced class weights

(a) DS1

No	Importance	Feature
1	26.72%	questionnaireno
2	11.49%	time_awake_less_nap
3	10.07%	age
4	7.30%	sleep_time_24h
5	7.14%	time_of_day_sine
6	6.03%	average_sleep_hours
7	3.99%	sleep_time_72h
8	3.85%	day_of_year_sine
9	3.35%	isi
10	3.20%	day_of_year_cosine
11	3.04%	time_since_end_of_shift
12	2.95%	time_in_bed
13	2.78%	time_since_start_of_shift
14	2.65%	sleep_time_48h
15	1.99%	time_of_day_cosine
16	1.75%	ho_chronotype
17	1.08%	work_duration
18	0.36%	nap_duration
19	0.28%	time_awake_since_last_nap

(b) DS2

No	Importance	Feature
1	28.97%	questionnaireno
2	20.82%	time_awake_less_nap
3	6.90%	time_of_day_sine
4	5.63%	isi
5	5.45%	time_in_bed
6	4.88%	age
7	3.65%	average_sleep_hours
8	3.00%	day_of_year_cosine
9	2.98%	time_of_day_cosine
10	2.82%	sleep_time_24h
11	2.56%	ho_chronotype
12	2.55%	day_of_year_sine
13	2.54%	sleep_time_72h
14	2.44%	sleep_time_48h
15	2.05%	time_since_end_of_shift
16	1.12%	time_since_start_of_shift
17	1.03%	work_duration
18	0.37%	time_awake_since_last_nap
19	0.24%	nap_duration



(a) DS1 (b) DS2

Figure 5.16: CMAE and CP by random forest regressor with over-sampling

5.4 Combination of Regression and Classification

In this section, we present the results of a model that combines both classification and regression as discussed in 4.5. As we have seen in Sections 5.2 and 5.3, each of them has unique strengths and weaknesses. Random forest classifier, when trained with over-sampled dataset, gives satisfying predictions of rare values (KSS=9 in our case). On the other hand, random forest regressor generally gives lower mean absolute error, yet it does not predict rare values.

In combination of the two models, we leverage the strengths while minimize the weaknesses of both. We trained the combined model with random over-sampled training datasets. The resulting model is good at predicting all values and keeps the mean absolute error low.

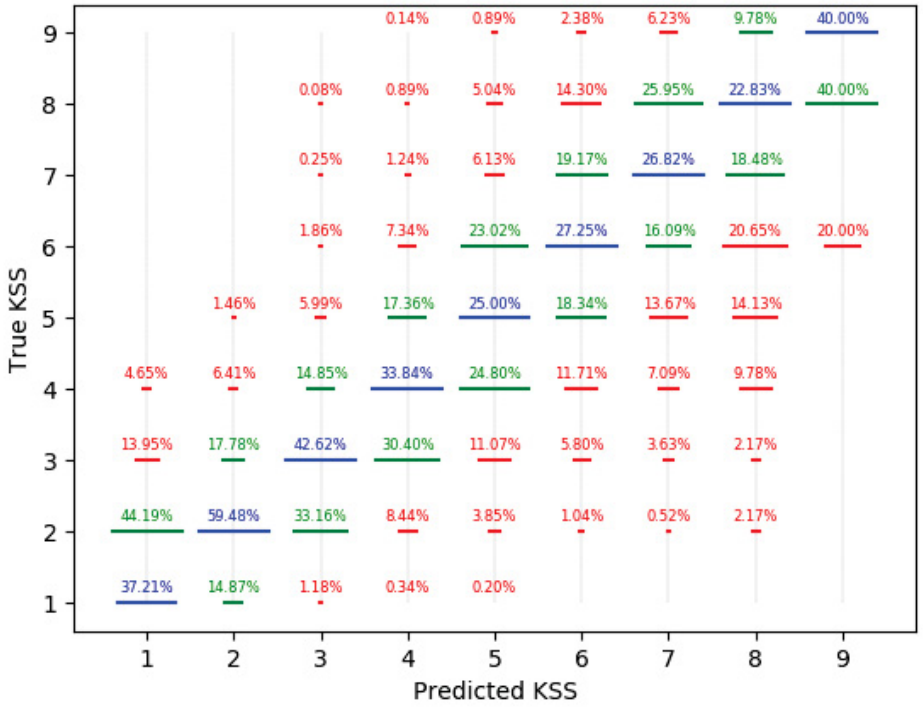


Figure 5.17: Normalized confusion matrix, DS1 with over-sampling

5.4.1 Highlights

On DS1, the combined model performs better with MAE of 0.93, compared to 0.93 of random forest regressor and 1.02 of random forest classifier. On DS2, it produces a MAE of 0.83 which is slightly better than those of individual models (0.86 and 0.88 by random forest regressor and random forest classifier, respectively). Figure 5.19 shows CMAE and CP values for the two datasets. With the exception of CP value of class 9 on DS1, the combined model performs well on both datasets. It produces low CP errors with minimal compromise on the CMAE.

5.4.2 Detailed Performance

Performance on DS1

Normalized confusion matrix of DS1 is shown on Figure 5.20. The combined model shows the same characteristics on this dataset as it does on DS2, with lower performance. The distribution

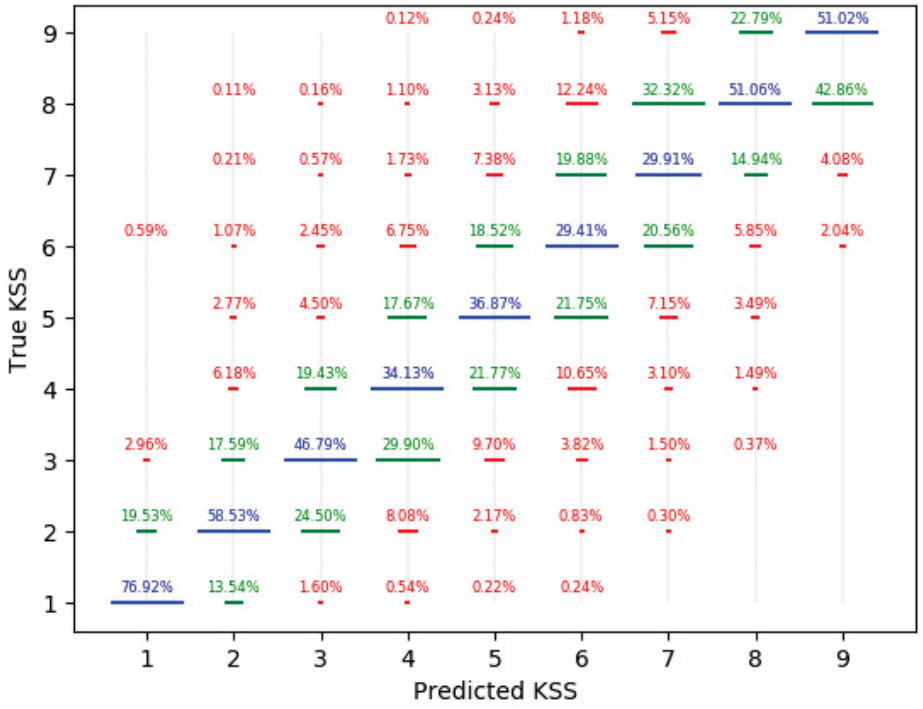
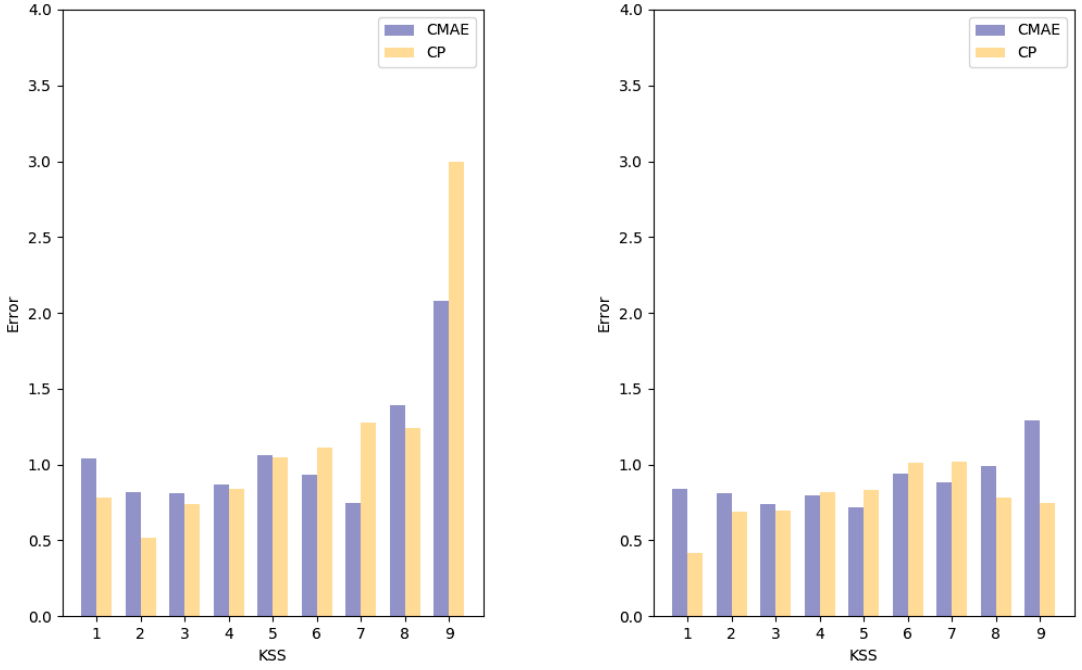


Figure 5.18: Normalized confusion matrix, DS2 with over-sampling

also peaks at predicted values, except for KSS value of class 9, but with lower percentages compared to those on DS2. The percentages of true value ranging on $p_k \pm 1$ are also lower: they range from low value of 50% at $p_k = 9$ to a high of 91.2% at $p_k = 2$, with the average of 67.4%.

Performance on DS2

The normalized confusion matrix of DS2 is shown in Figure 5.21. In all cases, distributions of true values peak at the predicted value as represented by blue diagonal line. The percentage of true values in the range $p_k \pm 1$ from predicted value p_k are high. They lie from the minimum of 72.3% (around $p_k = 6$) to maximal of 92.1% (around $p_k = 1$) with the average of 84.6%. In the area of particular interest, level of confidence around $p_k = 7, 8, 9$ are at 78.6%, 83.9% and 86.5% respectively. All distributions have thin tails which shows that the combined model makes less error when the true value moves further away from the predicted value.



(a) DS1 (b) DS2

Figure 5.19: CMAE and CP, combined model with random oversampling

5.4.3 Validation of Feature Selection

In this subsection, we compare results obtained with the original dataset of 38 features to the ones obtained with the 19-feature in DS2. We used the combination model, as it is the one providing the best results.

With 38 features the model has MAE of 0.98 compared to 0.83 with 19 features. This means that feature selection has significantly improved overall prediction performance of the model.

Furthermore, CMAE values of two datasets are shown on Figure 5.22(a) and CP values are shown on Figure 5.22(b). Overall, model with 19-feature DS2 has better CMAE values on all predicted values of KSS. On the CP metrics, we observe the same results except for class 1. The results suggest that by removing irrelevant and noisy features from the dataset, performance of model can be improved significantly (Saeys et al. [2008];Chen and Lin [2006]).

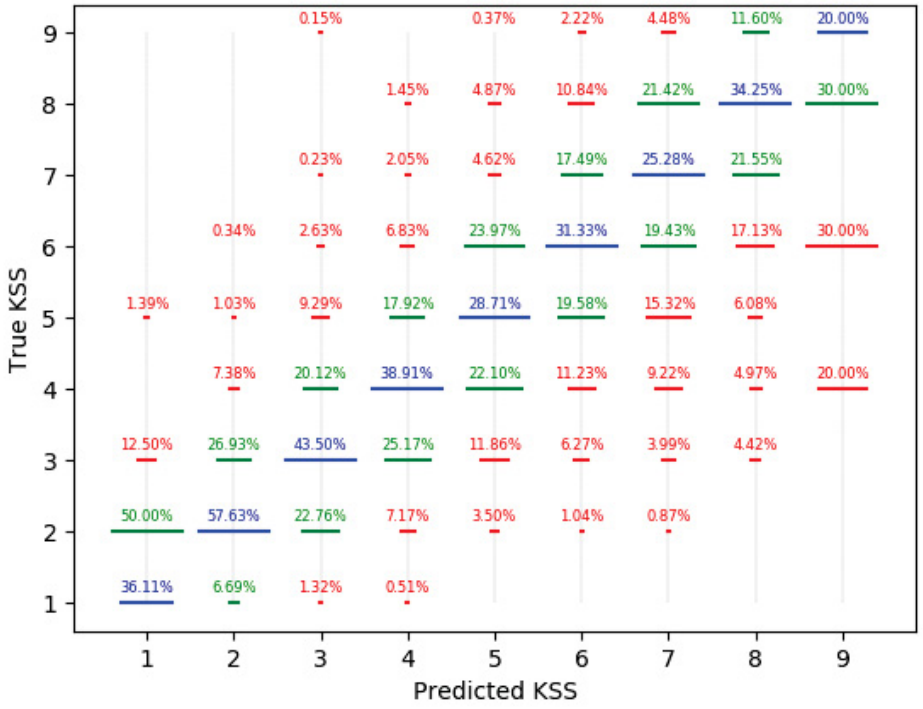


Figure 5.20: Normalized confusion matrix, DS1 with combination and random oversampling

5.4.4 Impact of the Size of the Datasets

In this subsection, we present the results on DS1 in comparison with reduced DS2. We use random under-sampling on original DS2 to reduce its size to 2,838 samples, the same as of DS1. The model to use is the combination.

With reduced DS2, the model has MAE of 0.97, which is much higher than that of original DS2 (0.83) and is very close to that of DS1 (0.93). This clearly shows the decrease in performance when the dataset is reduced. CMAE values of DS1 and reduced DS2 are shown on Figure 5.23(a) while CP values are shown on Figure 5.23(b).

It is clear from the two figures that the model produces the same effects on both datasets. It has high CMAE values on rare classes of 1, 8 and 9. On the CP metrics, the model has low

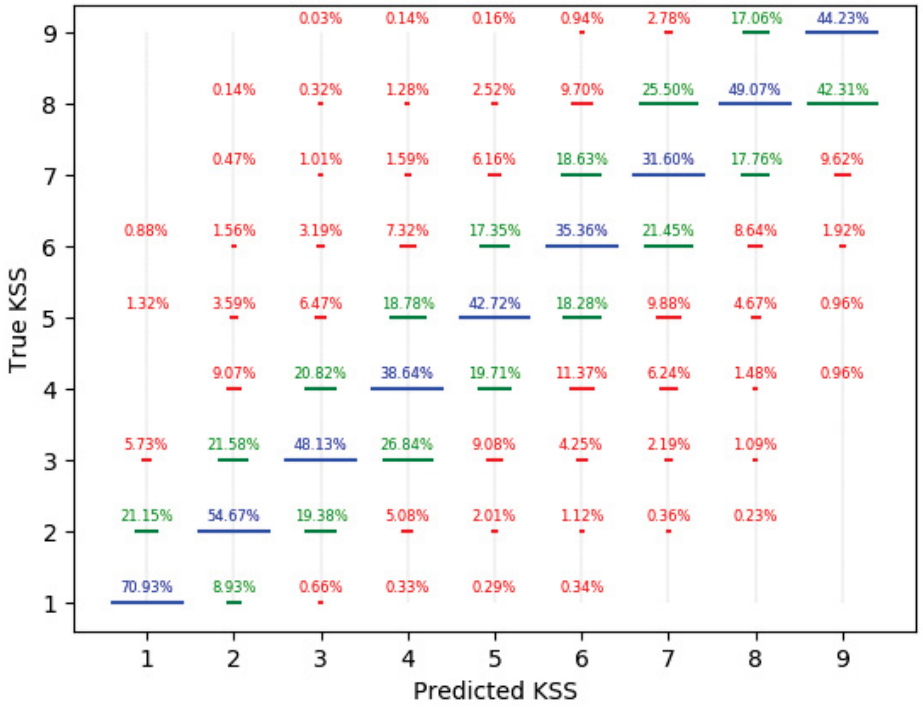


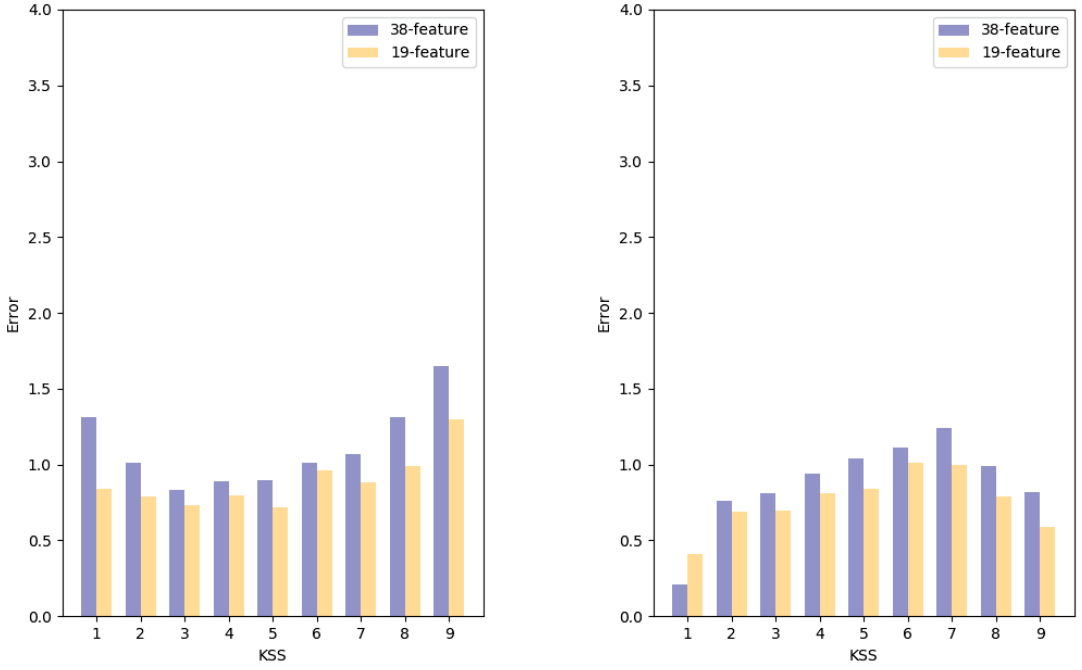
Figure 5.21: Normalized confusion matrix, DS2 with combination and random oversampling

CP values on rare classes on both datasets. The results on reduced DS2 suggest that size of datasets play crucial role to the performance of the model.

5.4.5 Concluding Remarks

In this section, we propose an ensemble of random forest classifier and regressor to leverage strengths while minimize weaknesses of them. The resulting combination has good MAE and provides satisfying prediction of sample in minority class. This results suggest that in a problem with relatively small and imbalanced dataset, one may try to combine classification and regression models to get a better performance.

Using the combination as the model, we presented the results of feature selection and comparison of DS1 and reduced DS2. Two conclusions have been drawn from these results. First, proper

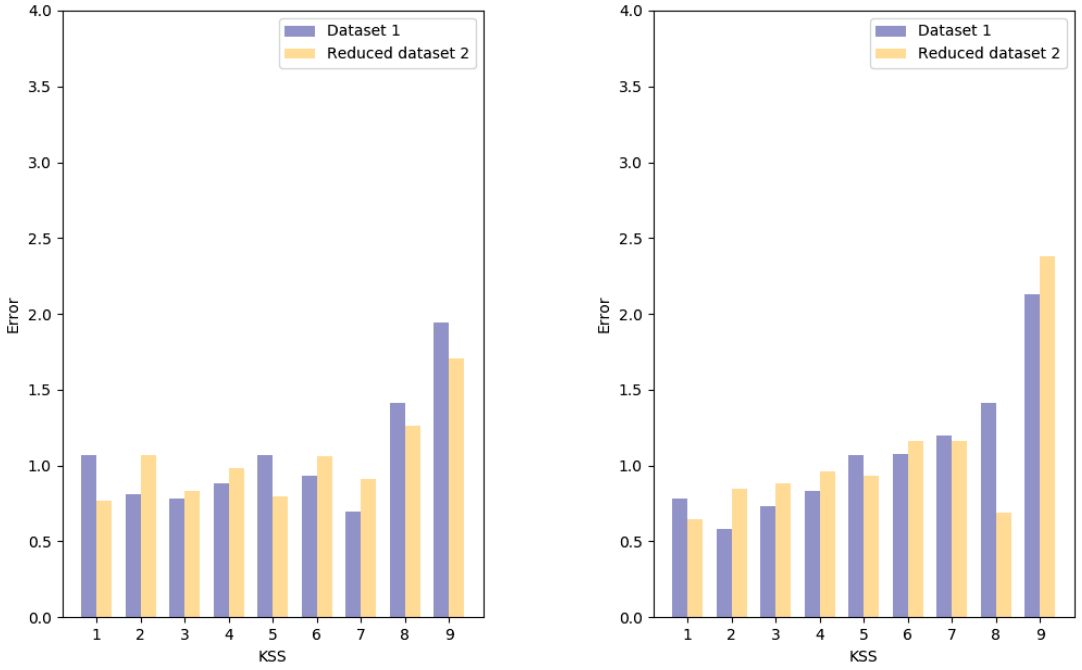


(a) CMAE

(b) CP

Figure 5.22: 19-feature and 38-feature DS2 by combination and random oversampling

selection of features can result in significant better performance. Second, the size of dataset play crucial role in the model's performance.



(a) CMAE

(b) CP

Figure 5.23: DS1 and reduced DS2 by combination and random oversampling

Chapter 6

Conclusions and Future Work

In this thesis work, three machine learning models have been built for prediction of fatigue in rotating shift workers. Fatigue was assessed as subjective levels of sleepiness based on KSS. KSS has been proposed as measure of fatigue, as in Gander et al. [2015]. KSS-based datasets are also larger as we collected them at least five times a day during the study, as opposed to PVT sessions which were taken only at the beginning and end of shifts.

Good performance of the three models suggests that KSS can be predicted using machine learning. Random forest regressor tends to give slightly better MAE than random forest classifier, regardless of the technique applied. This can be explained as random forest regressor minimizes MAE while random forest classifier relies on impurity function when they consider splits at a node. The difference in MAE between two models is not a direct result of the rounding of regression output. A quick experiment show that similar results, in terms of MAE are obtained if we do not round regressor's output to the nearest integer.

In our third model, a combination of random forest classifier and regressor showed better performance than both individual models. Two characteristics of the datasets may have contributed to this result. Firstly, the datasets are highly imbalanced. This makes the random forest regressor to favor majority classes in attempt to minimize MAE. As a result, regressor rarely predicts minority classes. Random forest classifier, however, is able to predict minority classes.

Combination of the two, therefore, is able to predict minority classes while keeping MAE low. Secondly, the output variable is ordinal. If the output was not ordinal, we would not be able to approach the problem from both classification and regression points of views and, consequently, no combination would be possible. This suggests that in a highly imbalanced dataset with ordinal output, one might expect a combination of a classifier and regressor to perform better than a single model.

One of the most popular models for prediction of KSS is the three process model (TPM) such as implemented by Ingre et al. [2014]. In this work, they used a modification of TPM to model KSS of airline pilots. Their best model, named *6d*, was reported to produce residual errors of 1.362. By comparison, our best model (combined model) was able to produce residual errors of 1.243. Although these values were obtained on different datasets, this comparison reinforces the conclusion that our models achieve good performance.

Dataset size plays a crucial role on model performance, especially in a field such as medicine where datasets are quite small from a machine learning point of view. In our thesis work, a bigger dataset yields better results on the same model than those of smaller one; yet, when sub-sampled to the size of the smaller one, both showed similar results. Hence, we conclude that our models could provide even better results on larger datasets.

As identified by feature importance rankings, a number of features consistently appeared as most important: *time_awake_less_nap*, *time_of_day* and *sleep_time_24h*. This finding is consistent with numerous studies such as Gillberg et al. [1994], Edwards et al. [2007] and Harrison et al. [2007]. One limitation of this interpretation is that feature importance rankings do not provide the directions of correlation between a feature and output variable.

Rankings of feature importance, provided by random forest, should be interpreted with caution. In the presence of correlated features, they will share the amount of importance (Genuer et al. [2010]). In this thesis work, there are known correlated group of features: *sleep_time_24h*, *sleep_time_48h*, *sleep_time_72h* are clearly correlated. This suggests a way of interpretation: cumulative number of sleep hours is very important to sleepiness. The work to identify less

obvious correlated features can be done through medical expertise, or correlation measurements such as Pearson correlation coefficient (Benesty et al. [2009]).

Models built with reduced feature set perform better than models with original feature set. There are indeed irrelevant features in the dataset. In the presence of such features, random forest may pick them as candidates for splitting with equal probabilities as other "good" features. This leads to less predictive models as these splits do not gain enough information compared to splits by other features. Random forest, therefore, benefits from the removal of such irrelevant features. Nevertheless, this feature selection process is semi-automatic and does not guarantee optimal selection. The feature selection is based on feature importance rankings and feature with least importance is removed. The feature importance rankings from random forest, however, can be misleading in presence of correlated features. Undetected correlation of features may cause removal of important ones. This suggests that future studies on the same topic can choose to focus on the most influential factors to fatigue in their data acquisition process.

In this study, datasets are highly imbalanced and pose difficulty for both classification and regression approaches. We dealt with imbalanced dataset with two simple techniques: random oversampling and class weights. The random oversampling is the simplest and naive way to do oversampling. Although unreported in Chapter 5, we experimented with the more sophisticated technique SMOTE. The results are comparable to those with oversampling and class weights. One possible explanation for such results is that the datasets contains a noticeable level of noise which may hinder SMOTE. At the end, we decided to continue with random oversampling for its simplicity. In the future however, other re-sampling techniques could be experimented, for example the one presented in Chen et al. [2004].

In this thesis, KSS was used as output for supervised machine learning models. Yet, in the study, we have two more measurements that can be used as output variable: the subjective 7-point Samn-Perelli Fatigue Scale and objective PVT mean reaction time. The Samn-Perelli was collected exactly at the same time KSS was taken, so it would make the dataset almost the same if it was chosen as output. Although unreported in Section 5, experiments using Samn-Perelli as output showed comparable performances to those using KSS. MAE of the combination model

with Samn-Perelli was 0.62 (out of 7) compared to 0.83 (out of 9) with KSS. CMAE and CP values were also comparable. This shows that Samn-Perelli and KSS are both good candidate as output for machine learning models. PVT mean reaction time is the objective measurement of alertness and could be a good candidates for output of our models. One limitation, however, is that PVT were taken only at beginnings and ends of shift. This makes the PVT-based datasets much smaller than Samn-Perelli- or KSS-based ones. Given that the KSS-based datasets used in this thesis are already quite small, PVT-based datasets are probably not sufficient to build machine learning models. Nevertheless, we conducted a number of experiments using PVT mean reaction time as output, using random forest regression. The results showed RMSE of around 100 milliseconds (ms), out of the normal range of 200ms to 500ms. In the future, a significantly larger datasets with PVT is needed to test if PVT should be used as output for machine learning models.

In the future, the work of this thesis can be extended in a number of ways.

First, data pre-processing can be improved. A subjective measurement of fatigue is prone to mislabeling in the first place. A worker may, intentionally or unintentionally, give incorrect level of fatigue. If we see this phenomenon as label noise (i.e., the labels have been incorrectly assigned on the training data), we can try to eliminate it by identifying and removing mislabeled data (Frénay and Verleysen [2014]; Brodley and Friedl [1999]). A combination of output labels can also be considered. Single label such as Samn-Perelli, KSS and PVT mean reaction time can be noisy and a combination of them may help reduce the noise. However one problem with this approach is interpretability. The combined label is new and unknown to the community, thus it is difficult to evaluate and validate the performance of the models.

Second, other metrics to evaluate model performance can be developed. While CMAEs and CPs are useful if we need to evaluate performance of models on a specific area of fatigue level, it is also convenient to have one single performance metric. A single performance metric would make model selection as well as parameters search much convenient. An example of a single performance metric could be overall error rate weighted by each class. Other metrics on feature importance ranking, rather than percentage, could also be developed.

Third, we need to experiment the models on different datasets to test their generalization. The datasets in this work come from the same industry, are in close geographical areas and have similar demographic data.

Fourth, we can explore other sets of variables that can give comparable results. Other sets may come from feature extraction (i.e., extract more features from existing ones) or, if available, from new datasets with more features. Other output variables can also be considered, for example the Samn-Perelli Fatigue Scale (Gawron [2016]) or other objective tests such as PVT.

Fifth, other machine learning algorithms can be experimented, for examples SVM and neural networks. SVM is traditionally good for medical datasets. One limitation is that it does not provide any insights on important features, thus feature selection can be difficult. Neural networks have gain tremendous success in the last few year due to the availability of massive datasets. It is also what hinders us from using neural networks as we do not, and practically will not, have such massive datasets.

Lastly, we use these models to build a software interface to use in a real working environment. This will contribute to early detection of fatigue at work and eventually to a safer working environment. The software can also help get feedback from end-users on the performance of the models. In this way, more data become available and the models in turn can be improved with new data.

Bibliography

T. Akerstedt and M. Gillberg. Subjective and objective sleepiness in the active individual. *Int. J. Neurosci.*, 52(1-2):29–37, May 1990.

T. Akerstedt, B. Peters, A. Anund, and G. Kecklund. Impaired alertness and performance driving home from the night shift: a driving simulator study. *J Sleep Res*, 14(1):17–20, Mar 2005.

N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992. doi: 10.1080/00031305.1992.10475879. URL `https://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879`.

L. Auria and R. A. Moro. Support Vector Machines (SVM) as a Technique for Solvency Analysis. Discussion Papers of DIW Berlin 811, DIW Berlin, German Institute for Economic Research, 2008. URL `https://ideas.repec.org/p/diw/diwwpp/dp811.html`.

M. H. Bae, T. Wu, and R. Pan. Mix-ratio sampling: Classifying multiclass imbalanced mouse brain images using support vector machine. *Expert Systems with Applications*, 37(7):4955 – 4965, 2010. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2009.12.018. URL `http://www.sciencedirect.com/science/article/pii/S0957417409010641`.

R. Barandela, J. S. Sánchez, V. Garca, and E. Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, 2003.

S. Barua, M. M. Islam, X. Yao, and K. Murase. Mwmote–majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(2):405–425, Feb 2014. ISSN 1041-4347. doi: 10.1109/TKDE.2012.232.

G. Belenky, N. J. Wesensten, D. R. Thorne, M. L. Thomas, H. C. Sing, D. P. Redmond, M. B. Russo, and T. J. Balkin. Patterns of performance degradation and restoration during sleep restriction and subsequent recovery: A sleep dose-response study. *Journal of sleep research*, 12(1):1–12, 2003.

G. Belenky, A. Lamp, A. Hemp, and J. Zaslona. *Fatigue in the Workplace*, pages 243–268. Springer, 10 2014. ISBN 978-1-4614-9086-9. doi: 10.1007/978-1-4614-9087-6_18.

M. Belgiu and L. Drgu. Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, 114:24 – 31, 2016. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2016.01.011. URL `http://www.sciencedirect.com/science/article/pii/S0924271616000265`.

J. Benesty, J. Chen, Y. Huang, and I. Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

M. Bhasin and G. Raghava. Svm based method for predicting hla-drb1* 0401 binding peptides in an antigen sequence. *Bioinformatics*, 20(3):421–423, 2004.

J. Bobadilla, F. Ortega, A. Hernando, and A. Gutirrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109 – 132, 2013. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2013.03.012. URL `http://www.sciencedirect.com/science/article/pii/S0950705113001044`.

D. B. Boivin and P. Boudreau. Impacts of shift work on sleep and circadian rhythms. *Pathol. Biol.*, 62(5):292–301, Oct 2014.

D. B. Boivin, P. Boudreau, F. O. James, and N. M. Kin. Photic resetting in night-shift work: impact on nurses' sleep. *Chronobiol. Int.*, 29(5):619–628, Jun 2012.

S. Borowiec. Alphago seals 4-1 victory over go grandmaster lee sedol. *The Guardian*, 15, 2016.

L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

L. Breiman. *Classification and regression trees*. Routledge, 2017.

C. E. Brodley and M. A. Friedl. Identifying mislabeled training data. *Journal of artificial intelligence research*, 11:131–167, 1999.

E. Byvatov and G. Schneider. Support vector machine applications in bioinformatics. *Applied bioinformatics*, 2(2):67–77, 2003.

S. Cai, H. Lin, X. Hu, Y. X. Cai, K. Chen, and W. Z. Cai. High fatigue and its associations with health and work related factors among female medical personnel at 54 hospitals in Zhuhai, China. *Psychol Health Med*, 23(3):304–316, 03 2018.

B. Carterette. *Precision and Recall*, pages 2126–2127. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_5050. URL `https://doi.org/10.1007/978-0-387-39940-9_5050`.

T. Chai and R. R. Draxler. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3): 1247–1250, 2014.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

C. Chen, A. Liaw, L. Breiman, et al. Using random forest to learn imbalanced data. *University of California, Berkeley*, 110:1–12, 2004.

Y.-W. Chen and C.-J. Lin. *Combining SVMs with Various Feature Selection Strategies*, pages 315–324. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-35488-8. doi: 10.1007/978-3-540-35488-8_13. URL `https://doi.org/10.1007/978-3-540-35488-8_13`.

Y.-M. Chyi. Classification analysis techniques for skewed class distribution problems. *Department of Information Management, National Sun Yat-Sen University*, 2003.

L. Di Milia, N. L. Rogers, and T. kerstedt. Sleepiness, long distance commuting and night work as predictors of driving performance. *PLOS ONE*, 7(9):1–6, 09 2012. doi: 10.1371/journal.pone.0045856. URL `https://doi.org/10.1371/journal.pone.0045856`.

J. Dorrian, S. D. Baulk, and D. Dawson. Work hours, workload, sleep and fatigue in australian rail industry employees. *Applied Ergonomics*, 42(2):202 – 209, 2011. ISSN 0003-6870. doi: https://doi.org/10.1016/j.apergo.2010.06.009. URL `http://www.sciencedirect.com/science/article/pii/S0003687010000864`. Special Section: Ergonomics, health and working time organization.

C. L. Drake, T. Roehrs, G. Richardson, J. K. Walsh, and T. Roth. Shift work sleep disorder: prevalence and consequences beyond that of symptomatic day workers. *Sleep*, 27(8):1453–1462, Dec 2004.

B. Edwards, J. Waterhouse, and T. Reilly. The effects of circadian rhythmicity and time-awake on a simple motor task. *Chronobiology International*, 24(6):1109–1124, 2007.

S. A. Ferguson, G. M. Paech, C. Sargent, D. Darwent, D. J. Kennaway, and G. D. Roach. The influence of circadian time and sleep dose on subjective fatigue ratings. *Accident Analysis & Prevention*, 45:50–54, 2012.

S. Folkard. Black times: temporal determinants of transport safety. *Accid Anal Prev*, 29(4):417–430, Jul 1997.

B. Frénay and M. Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2014.

J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

P. H. Gander, H. M. Mulrine, M. J. van den Berg, A. A. T. Smith, T. L. Signal, L. J. Wu, and G. Belenky. Effects of sleep/wake history and circadian phase on proposed pilot fatigue safety performance indicators. *Journal of Sleep Research*, 24(1):110–119, 2015. doi: 10.1111/jsr.12197. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/jsr.12197`.

V. J. Gawron. Overview of self-reported measures of fatigue. *The International Journal of Aviation Psychology*, 26(3-4):120–131, 2016. doi: 10.1080/10508414.2017.1329627. URL `https://doi.org/10.1080/10508414.2017.1329627`.

J. Geiger-Brown, V. E. Rogers, A. M. Trinkoff, R. L. Kane, R. B. Bausell, and S. M. Scharf. Sleep, sleepiness, fatigue, and performance of 12-hour-shift nurses. *Chronobiology International*, 29(2):211–219, 2012. doi: 10.3109/07420528.2011.645752. URL `https://doi.org/10.3109/07420528.2011.645752`. PMID: 22324559.

R. Genuer, J.-M. Poggi, and C. Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225 – 2236, 2010. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2010.03.014. URL `http://www.sciencedirect.com/science/article/pii/S0167865510000954`.

M. Gillberg, G. Kecklund, and T. kerstedt. Gillberg m, kecklund g, kerstedt t. relation between performance and subjective ratings of sleepiness during a night awake. sleep 17: 236-241. *Sleep*, 17:236–41, 05 1994.

B. A. Goldstein, A. E. Hubbard, A. Cutler, and L. F. Barcellos. An application of random forests to a genome-wide association dataset: Methodological considerations & new findings. *BMC Genetics*, 11(1):49, Jun 2010. ISSN 1471-2156. doi: 10.1186/1471-2156-11-49. URL `https://doi.org/10.1186/1471-2156-11-49`.

H. Han, W.-Y. Wang, and B.-H. Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In D.-S. Huang, X.-P. Zhang, and G.-B. Huang, editors, *Advances in Intelligent Computing*, pages 878–887, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31902-3.

Y. Harrison, K. Jones, and J. Waterhouse. The influence of time awake and circadian rhythm upon performance on a frontal lobe task. *Neuropsychologia*, 45(8):1966–1972, 2007.

J. C. Ho, M. B. Lee, R. Y. Chen, C. J. Chen, W. P. Chang, C. Y. Yeh, and S. Y. Lyu. Work-related fatigue among medical personnel in Taiwan. *J. Formos. Med. Assoc.*, 112(10):608–615, Oct 2013.

E. Hoddes, V. Zarcone, H. Smythe, R. Phillips, and W. C. Dement. Quantification of sleepiness: a new approach. *Psychophysiology*, 10(4):431–436, Jul 1973.

M. Ingre, T. KERSTEDT, B. PETERS, A. ANUND, G. KECKLUND, and A. PICKLES. Subjective sleepiness and accident risk avoiding the ecological fallacy. *Journal of Sleep Research*, 15(2):142–148, 2006. doi: 10.1111/j.1365-2869.2006.00517.x. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2869.2006.00517.x`.

M. Ingre, W. Van Leeuwen, T. Klemets, C. Ullvetter, S. Hough, G. Kecklund, D. Karlsson, and T. Åkerstedt. Validating and extending the three process model of alertness in airline operations. *PloS one*, 9(10):e108679, 2014.

T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

K. Kaida, M. Takahashi, T. Akerstedt, A. Nakata, Y. Otsuka, T. Haratani, and K. Fukasawa. Validation of the Karolinska sleepiness scale against performance and EEG variables. *Clin Neurophysiol*, 117(7):1574–1581, Jul 2006.

D. A. Kaiser. What is quantitative eeg? *Journal of Neurotherapy*, 10(4):3752, Mar 2007. doi: 10.1300/j184v10n04_05. URL `http://dx.doi.org/10.1300/J184v10n04_05`.

A. Kosmadopoulos, C. Sargent, D. Darwent, X. Zhou, and G. D. Roach. Alternatives to polysomnography (psg): a validation of wrist actigraphy and a partial-psg system. *Behavior research methods*, 46(4):1032–1041, 2014.

S. Kotsiantis, D. Kanellopoulos, P. Pintelas, et al. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30(1):25–36, 2006.

B. Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, Nov 2016. ISSN 2192-6360. doi: 10.1007/s13748-016-0094-0. URL `https://doi.org/10.1007/s13748-016-0094-0`.

N. Lamond, S. M. Jay, J. Dorrian, S. A. Ferguson, G. D. Roach, and D. Dawson. The sensitivity of a palm-based psychomotor vigilance task to severe sleep loss. *Behav Res Methods*, 40(1):347–352, Feb 2008.

Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. Huang. Large-scale image classification: fast feature extraction and svm training. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1689–1696. IEEE, 2011.

X.-Y. Liu, J. Wu, and Z.-H. Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

I. Mani and I. Zhang. knn approach to unbalanced data distributions: a case study involving information extraction. In *Proceedings of workshop on learning from imbalanced datasets*, volume 126, 2003.

J. Mathew, M. Luo, C. K. Pang, and H. L. Chan. Kernel-based smote for svm classification of imbalanced datasets. In *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, pages 001127–001132, Nov 2015. doi: 10.1109/IECON.2015.7392251.

M. Mohri, A. Talwalkar, and A. Rostamizadeh. *Foundations of machine learning (adaptive computation and machine learning series)*. Mit Press Cambridge, MA, 2012.

N. Oakley. Validation with polysomnography of the sleepwatch sleep/wake scoring algorithm used by the actiwatch activity monitoring system. *Bend: Mini Mitter, Cambridge Neurotechnology*, 1997.

Y. Pang, X. Li, H. Zheng, E. P. Wilder-Smith, K. Q. Shen, and W. Zhou. An auditory vigilance task for mental fatigue detection. *Conf Proc IEEE Eng Med Biol Soc*, 5:5284–5286, 2005.

S. R. Patel, J. Weng, M. Rueschman, K. A. Dudley, J. S. Loredo, Y. Mossavar-Rahmani, M. Ramirez, A. R. Ramos, K. Reid, A. N. Seiger, et al. Reproducibility of a standardized actigraphy scoring algorithm for sleep in a us hispanic/latino population. *Sleep*, 38(9):1497–1503, 2015.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 313–325. Springer, 2008.

B. Scholkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

F. Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

J. Shen, J. Barbera, and C. M. Shapiro. Distinguishing sleepiness and fatigue: focus on definition and measurement. *Sleep Medicine Reviews*, 10(1):63 – 76, 2006. ISSN 1087-0792. doi: https://doi.org/10.1016/j.smrv.2005.05.004. URL `http://www.sciencedirect.com/science/article/pii/S1087079205000444`.

K.-Q. Shen, X.-P. Li, C.-J. Ong, S.-Y. Shao, and E. P. Wilder-Smith. Eeg-based mental fatigue measurement using multi-class support vector machines with confidence estimate. *Clinical Neurophysiology*, 119(7):1524 – 1533, 2008. ISSN 1388-2457. doi: https://doi.org/10.1016/j.clinph.2008.03.012. URL `http://www.sciencedirect.com/science/article/pii/S1388245708002034`.

M. T. Smith and S. T. Wegener. Measures of sleep: the insomnia severity index, medical outcomes study (mos) sleep scale, pittsburgh sleep diary (psd), and pittsburgh sleep quality index (psqi). *Arthritis Care & Research*, 49(S5):S184–S196, 2003.

M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009.

A. Sun, E.-P. Lim, and W.-K. Ng. Web classification using support vector machine. In *Proceedings of the 4th international workshop on Web information and data management*, pages 96–99. ACM, 2002.

Y. Tarabalka, M. Fauvel, J. Chanussot, and J. A. Benediktsson. Svm and mrf-based method for accurate classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 7(4):736–740, 2010.

TSB. Tsb watchlist 2018. http://www.tsb.gc.ca/eng/surveillance-watchlist/index.asp, 2018.

H. Van Dongen, G. Maislin, J. M. Mullington, and D. F. Dinges. The cumulative cost of additional wakefulness: dose-response effects on neurobehavioral functions and sleep physiology from chronic sleep restriction and total sleep deprivation. *Sleep*, 26(2):117–126, 2003.

V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

J. C. Verster, J. Taillard, P. Sagaspe, B. Olivier, and P. Philip. Prolonged nocturnal driving can be as dangerous as severe alcohol-impaired driving. *Journal of sleep research*, 20(4):585–588, 2011.

M. Walker. *Why we sleep: Unlocking the power of sleep and dreams*. Simon and Schuster, 2017.

H. Wang, X. Liu, B. Lv, F. Yang, and Y. Hong. Reliable multi-label learning via conformal predictor and random forest for syndrome differentiation of chronic fatigue in traditional Chinese medicine. *PLoS ONE*, 9(6):e99565, 2014.

I. S. Wong, C. B. McLeod, and P. A. Demers. Shift work trends and risk of work injury among Canadian workers. *Scand J Work Environ Health*, 37(1):54–61, Jan 2011.

K. P. Wright, J. T. Hull, C. A. Czeisler, and C. A. Czeisler. Relationship between alertness, performance, and body temperature in humans. *Am. J. Physiol. Regul. Integr. Comp. Physiol.*, 283(6):R1370–1377, Dec 2002.

F. Yang, H.-z. Wang, H. Mi, C.-d. Lin, and W.-w. Cai. Using random forest for reliable classification and cost-sensitive learning for medical diagnosis. *BMC Bioinformatics*, 10(1): S22, Jan 2009. ISSN 1471-2105. doi: 10.1186/1471-2105-10-S1-S22. URL https://doi.org/10.1186/1471-2105-10-S1-S22.

S.-J. Yen and Y.-S. Lee. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3, Part 1):5718 – 5727, 2009. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2008.06.108. URL http://www.sciencedirect.com/science/article/pii/S0957417408003527.