NEW YORK UNIVERSITY

MASTER'S THESIS

Classifying the Quality of Movement via Motion Capture and Machine Learning

Author:
Ryan Saxe

Examiner:
Anasse Bari

Supervisor:
Dennis Shasha

A thesis submitted in fulfillment of the requirements for the degree of Master of Computer Science

in the

Courant Institute of Mathematical Sciences

December 20, 2018

Declaration of Authorship

I, Ryan Saxe, declare that this thesis titled, "Classifying the Quality of Movement via Motion Capture and Machine Learning" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date: 12/20/2018

NEW YORK UNIVERSITY

Abstract

Courant Institute of Mathematical Sciences

Master of Computer Science

Classifying the Quality of Movement via Motion Capture and Machine Learning

by Ryan Saxe

With the recent surge of Machine Vision technology and available video data, computational methods that utilize this data are becoming increasingly important. This Thesis shows that, with the proper application of Skeletal Tracking, it is possible to discern whether or not a physical task — a squat — is performed well. The Skeletal Tracking software employed is provided by Optitrack's Motion Capture client, Motive:Body. The data generated from Optitrack was used to extract features related to the proper execution of a squat. This thesis uses a variety of machine learning techniques to evalute the quality of physical performance. Support Vector Machines, Random Forests, and Decision Tree algorithms were tested with ten-fold cross validation, and compared to a baseline of Logistic Regression given the binary nature of the problem. While Regression performed at 66% accuracy, all three other algorithms performed substantially better, with Decision Trees performing best at 80%.

Acknowledgements

First and foremost, I would like to thank my supervisor, Dennis Shasha, for the guidance and support provided throughout this exploration.

I am extremely grateful for all the additional university staff that provided resources and assistance over this research exploration. Specifically, I would like to thank Ken Perlin and the members of his lab, Future Reality Lab, for allowing me access to state-of-the-art motion capture technology and assisting my utilization of said technology. I would also like to thank Anat Lubetsky for the guidance related to Physical Therapy and movement safety. Lastly, I would like to thank Leslie Greengard, David K.A. Mordecai, Samantha Kappagoda, and those affiliated with their lab, RiskEcon Lab for Decision Metrics, for their advisory role and support in relation to my understanding of movement, specifically in the field of Gait Recognition, that I pursued in the last year. This understanding was instrumental towards not only the algorithmic development of this thesis, but my passion for the problem at hand.

Outside of the university, I received assistance from two personal trainers, Dylan Ezzie and Trevor Schrier. These athletic professionals labeled the data and provided valuable consultation and feedback. I am very grateful for their help.

Last, but not least, I would also like to thank my family and friends for their much needed emotional support during this endeavor. It kept me going, and I am truly grateful for the relationships I have that made this process possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Definition

According to a large body of research (M Sluijs, Kok, and Zee, 1993, Essery et al., 2017), it is common for a patient in Physical Therapy to disregard their assigned exercises. This research suggests that the lack of proper feedback disincentivizes the patient from performing their exercises. There is a need for a system that can provide feedback on physical movement in order to encourage practice and improvement. The goal of this thesis is to show that the anatomical data from Skeletal Tracking can assess the quality of movement and therefore encourage patients to exercise.

1.2 Prior Work

Skeletal Tracking is an algorithmic approach to approximating the skeletal structure of a human, given data that represents their movement. Approaches to Skeletal Tracking span a broad range due to the varying quality and complexity of the system used to generate the required movement data. Implementations exist for simplistic single-camera setups, multi-depth-camera setups such as the Microsoft Kinect and Intel RealSense, as well as state-of-the-art motion capture systems[1] with a vast quantity of expensive cameras. This research used the latter, however the majority of the literature related to this problem utilizes the Kinect as it's a more accessible tool for research[2]. The reason we chose to use the Optitrack motion capture software rather than Microsoft's Kinect or Intel's RealSense is anatomical precision.Llorens et al., 2015 demonstrates that Optitrack achieves substantially improved accuracy on the Kinect with only two markers and two cameras and could be improved further with a more extensive setup[3]. The Microsoft Kinect is limited to the combination of an RGB camera and a depth camera. We believe that the Optitrack set up at New York University, can provide more detail for this research.

Nagymate and Kiss, 2018 provides a systematic literature review on the Optitrack, and shows a variety of research validating the impressive precision of the system. Furthermore, Skeletal Tracking algorithms that use only data from the Kinect tend to struggle with sitting positions[4]. While there is work[5] on this, it is unclear

[1] Two such systems are Optitrack and Vicon

[2] Mousavi Hondori, 2014 offers a comprehensive literature review on Microsoft Kinect applications related to Human Motion Recognition. The breadth of available literature dwarfs that of studies utilizing the expensive motion capture systems.

[3] The Optitrack setup in the Future Reality Lab at New York University uses twelve cameras and over forty markers.

[4] For a visual display of Skeletal Tracking struggling with movements such as a squat, watch this. This video is not mine, and was simply found on Youtube for this exemplary purpose.

[5] Le, Nguyen, and Nguyen, 2013 outlining an algorithm that is capable of identifying a sitting position from the data generated from Microsoft Kinect Skeletal Tracking.

how useful this would be for analyzing a squat in real-time.

Providing feedback for rehabilitation purposes increases the probability of compliance (M Sluijs, Kok, and Zee, 1993). Chang et al., 2016 devises a system using the Kinect that provides customized feedback and tracks the rehabilitation process. Su, 2013 uses Dynamic Time Warping alongside the Kinect in attempt to achieve a home rehabilitation environment and achieved 80% accuracy when compared to the feedback of a professional. Realtime feedback is crucial to optimize improved rehabilitation. This is explored in Zhao, 2016. Since the inception of the Kinect, home rehabilitation systems to accompany processes like Physical Therapy has been a desirable avenue of research. Mousavi Hondori, 2014 provides an extensive review on the clinical impact of the Kinect, and Da Gama et al., 2015 provides a more focused review on the Kinect in specific relation to motor rehabilitation.

Collections of movement data sets for Physical Therapy and rehabilitation research purposes are increasingly important in order to explore classification of movement using Skeletal Tracking and Machine Learning. Vakanski et al., 2018 and Leightley et al., 2015 are two such initiatives to develop high quality skeletal data for the purposes of Machine Learning research. Maudsley-Barton et al., 2017 is a comparative study of Machine Learning and Deep Learning techniques that utilize the Kinect for clinical motion analysis. Jun et al., 2013 used Machine Learning alongside the Kinect in order to classify the quality of a deep squat. This study appears successful, as they achieved 95% accuracy. They implemented Principal Component Analysis alongside K-Nearest Neighbor classification as their Machine Learning infrastructure. The majority of the prior work in this field uses a less involved surveillance system than Optitrack. These developments on the Kinect have lower anatomical accuracy than the work in this thesis, and hence are limited on the certainty of feedback and results they can provide. Our aim is to use the state-of-the-art Optitrack motion capture software as a stronger grounds for a Machine Learning study on movement quality classification.

1.3 Outline of the Rest of the Thesis

This thesis describes a methodology for analyzing motion capture data in order to provide feedback regarding the quality of a squat. Section 2 describes the data collection method. Furthermore, section 2 outlines inconsistencies and problems within the data in order to provide context for decisions made related to feature extraction and algorithmic design in later sections. Section 3 describes the decision process of selecting and extracting proper features in order to classify a squat as good or bad. These decisions were made upon consultation with physical therapists as well as personal trainers. Section 4 provides an analysis of Machine Learning results utilizing the data collected and features extracted described in the prior sections. This section suggests that collecting a more extensive data set could lead to a robust solution to this problem. Section 5 outlines suggestions for future work. Finally, the code relevant for this thesis is present in the appendix.

Chapter 2

Data Collection and Methodologies

2.1 Optitrack

As described in the previous section, there are many ways to go about Skeletal Tracking. For this research, we chose to utilize the Skeletal Tracking software provided in Optitrack's Motive:Body. We considered a variety of other video set-ups such as the Intel RealSense camera and Microsoft Kinect, and furthermore tested Skeletal Tracking software on the RealSense. While developing this algorithm on a simpler and cheaper system would be ideal for providing feedback to physical therapy exercises, it is a much more difficult problem as outlined in Section 1.2. Fortunately, Optitrack's Motion Capture equipment provides a more robust measurement of bone placement[1] due to the quality and quantity of cameras[2]. While this doesn't solve all the problems related to approximated skeletal structure, it helped a great deal.

2.1.1 Skeleton

Figure 2.1 displays the specific Optitrack skeleton used for this research. It is the baseline skeleton with markers hinged at the toe, with a total of 41 markers. For the purpose of this study, we did not believe a more extensive skeleton with specific markers dedicated to fingers were necessary[3].

It should be noted that the choice of skeleton within the client does not alter the anatomical structure approximated from Skeletal Tracking. The organization of the markers enable Motive to approximate where bones are from bone markers. Figure 2.2 shows an excerpt of the data from Skeletal Tracking for bone location approximation.

2.1.2 Output

With the cameras and skeleton set up, we could generate data as described on Optitrack's Data Export Page. Given the small sample of data, we elected to export every possible feature, in order to give the widest array of possible features to extract. Figure 2.2 displays what this data looks like. Every marker from the Optitrack set up has corresponding X,Y,Z coordinates, as well as respective rotation within the physical space. However, we did not use any of the rotation variables during this study. The variables we used during feature extraction, described in detail in Section 3, are Time and (X,Y,Z) coordinates. Furthermore, we consider only the approximated location of the bones and not the positions of the bone markers.

[1]For more detail on the accuracy in regards to Joint Angles approximated via Kinect Skeletal Tracking, refer to Choppin, Lane, and Wheat, 2014

[2]The Optitrack setup for this thesis utilized twelve of their Prime 13 cameras

[3]For more information on skeletal options, please refer to the Optitrack Wiki.

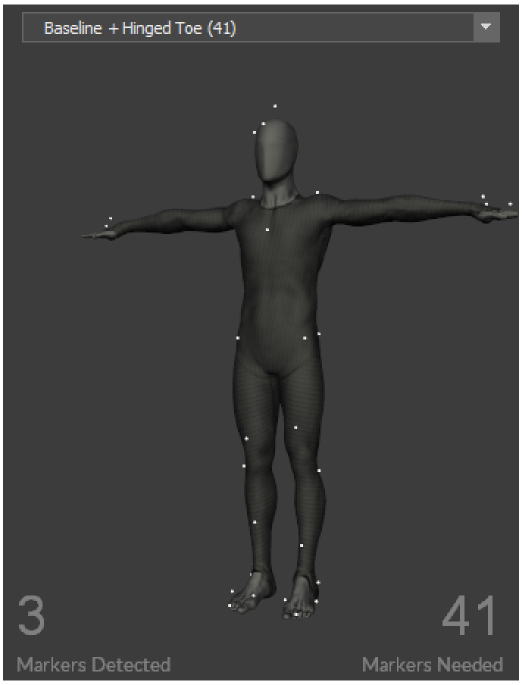

FIGURE 2.1: Optitrack Skeleton

		Bone	Bone	Bone	Bone	Bone	Bone	Bone
		Ryan:Hip	Ryan:Hip	Ryan:Hip	Ryan:Hip	Ryan:Hip	Ryan:Hip	Ryan:Hip
		1	1	1	1	1	1	1
		Rotation	Rotation	Rotation	Rotation	Position	Position	Position
Frame	Time (Seconds)	X	Y	Z	W	X	Y	Z
2350	19.583333	0.066799	0.017296	0.011963	-0.997545	0.02481	0.760111	-0.18603
2351	19.591667	0.067049	0.016981	0.011973	-0.997533	0.024698	0.760081	-0.185898
2352	19.6	0.067228	0.016668	0.01199	-0.997526	0.024589	0.760052	-0.185801
2353	19.608333	0.067477	0.016399	0.012017	-0.997514	0.024492	0.760021	-0.18571
2354	19.616667	0.067665	0.016109	0.012097	-0.997505	0.024389	0.760014	-0.185643
2355	19.625	0.067847	0.015852	0.012165	-0.997496	0.024301	0.759983	-0.185597
2356	19.633333	0.067988	0.015582	0.012303	-0.997489	0.024205	0.759982	-0.185574
2357	19.641667	0.068124	0.015286	0.012372	-0.997483	0.024131	0.759964	-0.185555
2358	19.65	0.06825	0.014956	0.012503	-0.997478	0.024043	0.759968	-0.185558
2359	19.658333	0.068372	0.014667	0.012571	-0.997473	0.02397	0.759949	-0.185565
2360	19.666667	0.068475	0.014343	0.012638	-0.99747	0.023902	0.759957	-0.185604
2361	19.675	0.068605	0.014061	0.012695	-0.997464	0.02384	0.759944	-0.185655
2362	19.683333	0.068692	0.013793	0.012792	-0.997461	0.023775	0.759961	-0.185734
2363	19.691667	0.068832	0.013511	0.012869	-0.997454	0.023723	0.759937	-0.185814
2364	19.7	0.068968	0.013292	0.01297	-0.997446	0.023672	0.759945	-0.185926
2365	19.708333	0.069174	0.013068	0.013031	-0.997434	0.023634	0.759925	-0.186065
2366	19.716667	0.0693	0.012807	0.013057	-0.997428	0.023628	0.759949	-0.186299
2367	19.725	0.06948	0.01267	0.013079	-0.997417	0.023619	0.759929	-0.186487
2368	19.733333	0.069695	0.012528	0.013137	-0.997403	0.023616	0.75995	-0.186708
2369	19.741667	0.069912	0.012423	0.013139	-0.997389	0.023617	0.759924	-0.186957
2370	19.75	0.070087	0.012329	0.013129	-0.997378	0.023633	0.75993	-0.187254
2371	19.758333	0.070336	0.012291	0.013119	-0.997361	0.023662	0.759912	-0.187572
2372	19.766667	0.070568	0.012292	0.013155	-0.997344	0.02369	0.759934	-0.187947
2373	19.775	0.070797	0.012347	0.013101	-0.997328	0.023733	0.75991	-0.188354

FIGURE 2.2: Snapshot of CSV Output.

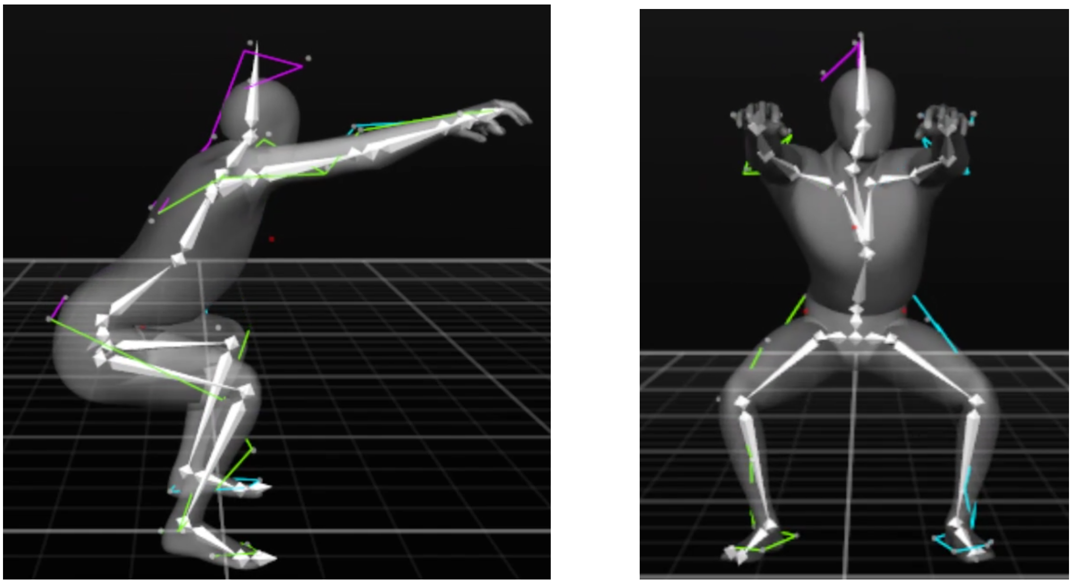

FIGURE 2.3: Optitrack data when markers are covered.

2.2 Data Labeling

Two personal trainers, Dylan Ezzie and Trevor Schrier, were asked to label the videos generated[4]. The personal trainers were told to label a task that was performed well as 1, and a task that was performed poorly as -1. If they were unable to discern how well a task was performed, they were to label that video 0. Furthermore, they were instructed to do their best to be aware of the issue presented in section 2.3.1.

There are a total of 42 squats. Dylan labeled ten squats as 0, while Trevor only labeled two squats as such. Dylan agreed with the 0 labels given by Trevor. After discarding the data points that the experts either could not discern or on which they disgreed, the dataset had 22 squats. These 22 squats are the data used in the computational parts of this study. It should be noted that this sample is incredibly small for the computational methods this thesis employs[5]. However, the agreement between experts implies the signal of these 22 data points is very strong. My advisor, Dennis Shasha, suggested that 22 clearly categorized data points might be sufficient to derive a good model.

2.3 Problems and Difficulties

Several problems arose during data collection that influenced both the manner and extent to which we approached the algorithm and features. Corrections will be discussed in more depth in section 5.

2.3.1 Marker Detection

Figure 2.3 displays a subject performing a squat. The purple, blue, and green lines represent the inference made by the device software to determine the skeletal structure connected to the markers. Notice that the markers on the hip and in front of the chest are red rather than white. This occurs when the camera infrastructure cannot detect those markers. The skeletal tracking software, at this time, attempts to calibrate and can make a mistake. This happened consistently enough that we could not disregard the related data.

According to Ken Perlin, the Director of the Future Reality Lab — the lab at New York University that has this technology — A potential solution to this flaw in our data collection method is to add additional cameras at hip level. Should there be a continuation on this work, this should be considered during the set up of Optitrack.

2.3.2 Quantity and Diversity

Unfortunately, it was not possible to generate data from a diverse population of subjects. Access to the equipment was limited, and hence only one subject performed the tasks in the presented data. While this is not ideal, it is not an issue towards demonstrating that an algorithm can deduce the difference between a good and bad squat. However, this does limit the extrapolations that can be made regarding the analysis of features that determine decisions during the classification process. With a higher quality and larger quantity of data, these computational methods can provide clear insight and better results. Data collection should be the main goal in regards to improvement for a continuation of this study.

[4]Both Dylan and Trevor labeled videos of squats and lunges, however we never got to developing a similar algorithm on lunges, and hence these videos are out of the scope of this thesis. If you're curious about lunges and the labels, you can find this information in the Github repository.

[5]In order to achieve a robust version of the algorithm represented here, the data set must be improved. This was not feasible within the scope of this thesis, and is addressed in section 5

Chapter 3

Feature Extraction

The data set generated for the purpose of this research is not large. Given that the goal is classification, we must choose a small set of features in order to accommodate the size of the data set. In order to properly create these features, professionals in the field of Physical Therapy and Personal Training were consulted. This section will describe a variety of features that can be extracted from Skeletal Tracking data that is relevant to a wide array of movement. However, given that this thesis focuses on the squat, the features illustrated are discussed in this context.

Joint angles are a common metric to measure from Skeletal Tracking data when considering any physiological analysis as a variety of features can be extracted from them (Le, Nguyen, and Nguyen, 2013). Let a, b, c correspond to the ankle, knee, and hip accordingly. Then, the following yields the joint angle of the knee, where a subject standing straight up would have a $knee_{angle}$ of 180 degrees[1]:

$$ab = \frac{a - b}{\|a - b\|}$$

$$cb = \frac{c - b}{\|c - b\|}$$

$$knee_{angle} = acos(ab \cdot cb)$$

For the performance of a squat, the knee angles are the only crucial joint angles, however there are plenty of additional relevant joint angles. Measuring the angle of the ankle can assist in detecting whether or not the subject performing the squat has ankle dorsiflection[2] or plantar flexion[3].

The features in this section mainly utilize the knee joint angle, however Section 3.2 discusses the postural alignment of the hips and shoulders, which does not use joint angles.

3.1 Climax

While every joint angle has a time series representation in this data, not every point in time is equal. Analyses on squat performance and injury suggest that the climactic point of a squat, when the subject is lowest to the ground, is the most important to determine quality of performance and potential of injury (Almosnino, Kingston, and Graham, 2013, Schoenfeld, 2010, McLaughlin, Lardner, and Dillman, 1978, Lorenzetti et al., 2018).

[1] If standing straight, for some reason, should be represented with a zero-degree-angle, then replace cb with bc

[2] Ankle dorsiflexion is defined as upward flexing of the ankle such that one's toes raise.

[3] Ankle plantar flexion is defined as downward flexing of the ankle such that one's toes point downward, often leading to lifting the heals off the ground, which is poor form.

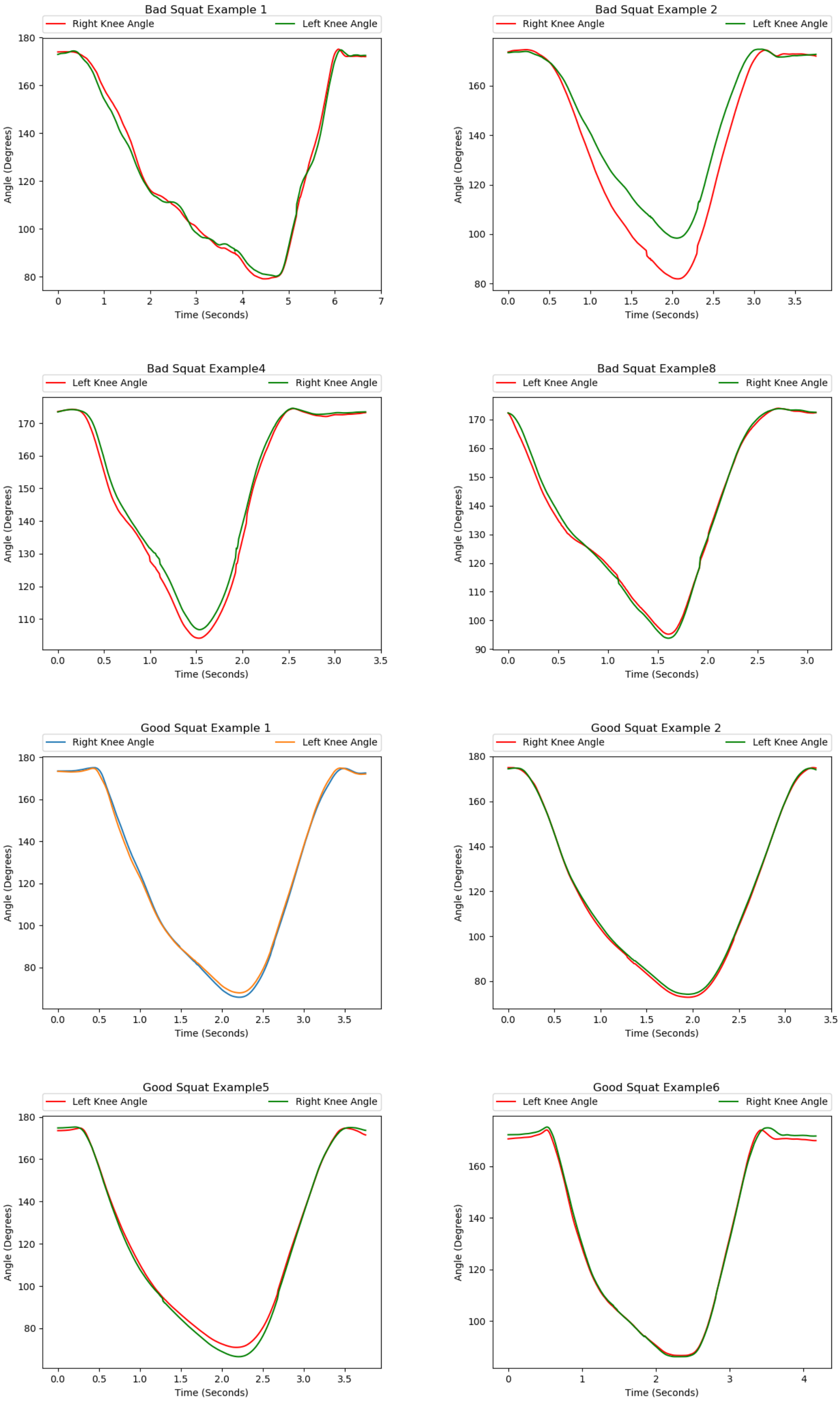

FIGURE 3.1: Knee Angles Over Time.

Figure 3.1 displays the time series representation of knee angles for a couple different trials. The squats that were labeled bad appear to have more variation, but not necessarily a drastically different knee angle. However, there is a lot more consistency in the angle at the climax of good squats represented in the data. We still utilized both left and right knee angle at climax as features, which can be seen in the feature vector table (3.1).

3.2 Symmetry

Dynamic Time Warping is a similarity metric employed for comparing temporal sequences[4]. Given that performing a physical task happens over time, Dynamic Time Warping can be used to compare joint angles over time. This is done by finding the warping path, which aligns both temporal sequences and minimizes the cost of this alignment.

$$D(i,j) = min\{D(i-1,j), D(i,j-1), D(i-1,j-1)\} + cost(x_i, y_j)$$

Where $cost(x_i, y_j)$ is the cost of aligning those parts of their respective sequences. Given this matrix, the DTW algorithm finds the cheapest path using dynamic programming by starting at the end and selecting, through a recursive process, the cheapest path that ends at either $D(N-1,M), D(N,M-1), D(N-1,M-1)$. The cost of this path is hence the distance between the temporal sequences in question.

We did not feel Dynamic Time Warping was necessary, as there was no need to compare sequences across different trials. However, should this research move further into more complicated features and methods, implementing Dynamic Time Warping makes sense. This has even been used for approaching similar problems regarding rehabilitation and exercise (ÅwitoÅski, JosiÅski, and Wojciechowski, 2018, Su, 2013). For now, the following simple symmetry score was used:

$$\frac{\sum_{t=0}^{n} \mid L_t - R_t \mid}{t}$$

This equation represents the average difference between the left knee angle and the right knee angle. If the result is zero, then the subject is performing a squat with very good form, as it is a symmetrical movement McLaughlin, Lardner, and Dillman, 1978[5]. Notice that a large symmetry score corresponds to a squat of poor form. This can be seen in table 3.1.

3.3 Alignment

Myer, 2014 illustrates a specific type of squat, the back squat, and the importance of alignment of a variety of parts of the body, including the knees, in order to prevent injury. Alignment is a concept extrapolated from symmetry. We consider only alignment of symmetrical anatomical parts, such as the left and the right hip. The goal of an alignment metric is to determine if some aspects of the skeleton in question are lacking proper posture (e.g. one hip sitting lower than the other). For this metric, we consider hip alignment and shoulder alignment. An individual hip or shoulder

[4]It should be noted that Dynamic Time Warping (DTW) can also be used as a classification technique as seen in (Switonski, Michalczuk, Josinski, Polanski, Wojciechowski, 2012)

[5]This specific resource is about squats with weights, however it was still useful for understanding the movement.

Label	Left Knee at Climax	Right Knee at Climax	Left Knee Max Jerk	Right Knee Max Jerk	Knee Symmetry Score	Hip Alignment Score	Shoulder Alignment Score
-1	63.1724365773	54.5690971556	4.94266880482	4.76826632781	2.39729800573	1.33787030007	2.65686059678
-1	80.5207062211	80.2577435872	3.46854307899	1.08390081566	1.94470738041	1.86621027198	3.79804039085
-1	29.7726564376	26.5151656975	0.534682302243	1.26605585371	1.98141662674	5.66975746155	10.5902633667
-1	57.8779027306	57.58720772	3.55277903731	2.94574308868	1.24613827597	5.50944733103	9.74559207554
-1	104.130029577	106.716813911	3.29580178602	4.27624726711	2.06818134541	1.30171478891	2.32025880174
-1	70.8643332503	71.0484907227	3.57171615333	1.63536163239	1.17395372985	1.58967545337	3.25466851966
1	73.6264025999	74.4930466749	1.46559533921	1.63317495951	1.73580156678	11.0365237753	20.1625893294
1	76.148555193	76.272845687	1.56499435711	2.0933349038	1.86263143955	11.0871075725	19.9904500609
1	67.8342165877	65.7318906675	0.493008623096	0.696508546562	1.2927694281	11.7120402018	21.2529174805
1	74.2264406144	72.878749014	0.639243022977	1.03836377241	0.897157243042	11.4085315669	20.7984212693
1	65.5788306279	64.517000204	0.676594197103	0.751442875429	1.09451528007	1.59106793271	2.84873178566
1	71.038009781	66.6325658496	0.963293350358	2.47597857853	1.87076064599	1.7641031901	3.2723719279
1	71.6007710585	67.2440885663	0.651431169851	1.90653298811	2.73257418567	1.56548601612	2.18983211838
1	71.5333086949	67.3557096028	0.478146886897	1.4110919396	2.43559882895	1.74758019772	2.24384244134
1	86.7258431389	86.2729713331	0.375450777423	0.618851608117	1.03257416202	1.10629488678	1.08469553853
-1	86.3709519234	88.0439464185	0.98197008532	1.23438905076	1.65052267202	1.25413110877	1.7530001486
-1	93.6267620985	92.9909171745	0.621490183353	1.45819364252	0.900036540654	1.52339108785	1.59489008586
-1	70.790383766	61.6979981554	1.61342155745	2.63101488892	3.01193459789	2.50772018433	4.14155502319
-1	95.2433705231	93.8451520022	1.70336997121	2.97725113856	1.24696249882	1.21075241187	1.92966337703
-1	98.4106308769	81.9207095487	2.13926167577	3.80496402876	8.42393447736	1.37473449707	1.7249525706
-1	90.5469375592	88.2199507259	1.30860342515	2.30897879868	1.32200740433	1.14024824557	1.75849385499
-1	98.4708416617	100.482999866	0.465001068613	0.293514008889	4.38892425675	1.87926991781	3.32749226888

â

TABLE 3.1: The Feature Vectors of the data.

cannot move to the side without also moving up/down or forward/backward. Because of this, it is only necessary to consider placement along the sagittal plane[6] for an alignment metric. Legaye and Duval-BeaupÃšre, 2005 displays the relationship of spinal alignment with the hip and pelvis along the sagittal plane, which suggests analysis along this plane is potentially useful for our alignment metric.

We quantified this by disregarding the axis with the largest delta, and measuring the sum of the difference of other axes over time, normalized to the time-length of the trial. The reason for this is that the axis with the largest delta will be the axis that separates the right and left hip (e.g. Z-axis for the left of 2.3, and X-axis for the right of 2.3). This axis does not exist on the sagittal plane, and hence is disregarded. For future research, it is possible to reduce the noise of this metric by computing a rotation on the skeletal data to ensure that each subject is facing the same direction.

3.4 Jerk

Jerk is the derivative of the acceleration. Measuring jolting movements can be useful for identifying unsafe execution of a physical task, or even identifying movement disorders, as a symptom of many are non-fluid movements. We decided to consider only the maximum value for jerk, rather than the full time series. The reason for this is that the feature vector became too complex for our small sample size, and hence a simpler feature representing Jerk was necessary. Figure 3.2 displays the jerk over time of the knee joint angle for both good and bad squats. You can see that the poor squats have a much larger range for their maximum jerk.

[6]The sagittal plane is the anatomical plane that divides the body into left and right sides.

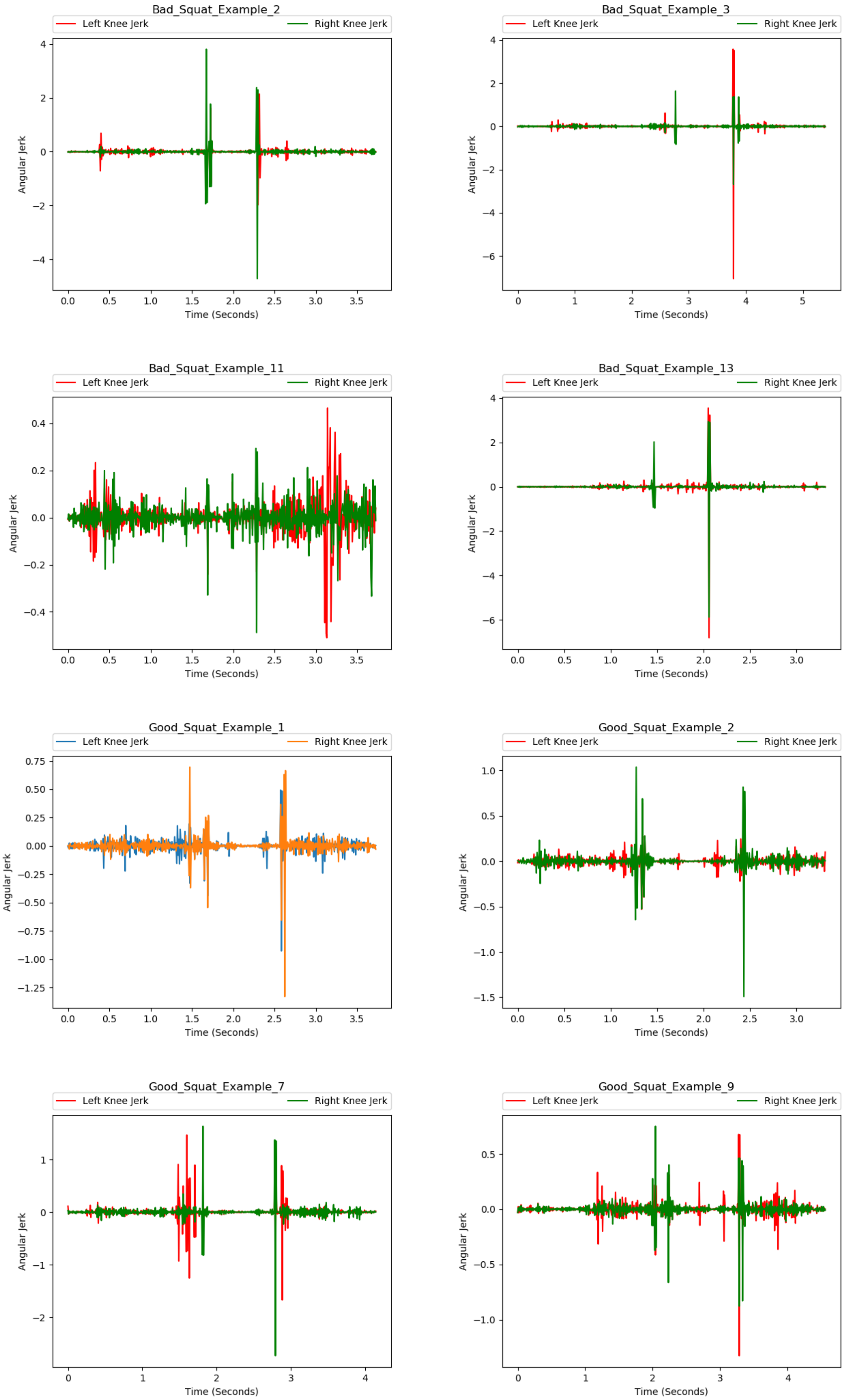

FIGURE 3.2: Jerk of Knee Angles Over Time.

Chapter 4

Algorithm and Results

4.1 Algorithm Overview

This section describes the Machine Learning algorithms utilized to analyze the data described in Section 2. As previously stated, the results should be taken as suggestive only, because the quantity of data present is quite small. Among other issues, the small size of the data removes the ability to employ hyper-parameter tuning for these algorithms. So our experiments use the baseline hyperparameter values in the Sci-kit Learn Python package.[1].

4.1.1 Logistic Regression

Logistic Regression (Yu, Huang, and Lin, 2011[2]) is a popular technique for binary classification. let x be a data point and $y \in \{-1, 1\}$ the corresponding label. Logistic Regression uses the sigmoid function[3] to model this conditional probability as such:

$$P(y|x) = \frac{1}{1 - e^{-yw^T x}}$$

Where $w \in \mathbb{R}^n$ is the weight vector corresponding to the features in your data. This weight vector is computed by minimizing the negative log likelihood of the conditional probability:

$$w = argmin_w - log \prod_{i=1}^{n} \frac{1}{1 - e^{-y_i w^T x_i}}$$

The problem in this thesis is a binary classification problem, but squat assessment has granularity. Squat performance is not just good or bad as can be seen by the literature on the kinematics of a squat (Schoenfeld, 2010). The architecture of Logistic Regression is appealing. The model learns a simple logit function for separating the data, and is hence capable of expressing confidence based on the distance of any data point to this separating function. We will use the results of logistic regression as a baseline for the other algorithms.

[1]For specifics in regards to the code, refer to Appendix A.

[2]This is the Logistic Regression reference of choice for the Scikit-Learn package, which is what we used for computation.

[3]$f(x) = \frac{1}{1-e^{-x}}$

4.1.2 Support Vector Machine (SVM)

A Support Vector Machine (Cortes and Vapnik, 1995) is a supervised learning algorithm. The objective of this algorithm is to discover a hyperplane of lower dimensionality that can separate data-points by the largest margin in order to classify them.

Let our data points be of the form (x_i, y_i) where x_i is the feature vector, and y_i is the binary classification associated with x_i. Assuming that the data in question is linearly separable[4], the following must be true for the calculated support vector w where b is a bias involved for representation of the hyperplane.

$$(w \cdot x_i + b)y_i \geq 1 \forall i$$

However, the assumption of linear separability is too strong. To solve this problem the margins found can be softened by introducing a hinge-loss function. This generates a hyperplane that minimizes misclassification in the training data.

$$min_{w,b} \sum_j max\{0, 1 - (w \cdot x_i + b)y_i\}$$

The last aspect of SVMs is called a kernel, and is a large contributor to why SVMs are so robust. A kernel k is a similarity metric with the following restriction:

Let $k : A \times A \rightarrow$, then $\exists f : A \rightarrow B$ such that $\forall x, y \in A \quad k(x, y) =< f(x), f(y) >$

We implemented this algorithm with an Radial Base Function (RBF) kernel[5]. Not only are RBF kernels the most widely used default kernel for SVMs (Campbell, 2001), but they are stationary — $k(x, y) = k(x + c, y + c)$. This is ideal for the space of motion capture data because it is important that the same squat performed in a different spot in the room is treated equally. Furthermore, we can deduce the confidence in prediction, similar to Logistic Regression, by measuring the distance from the learned hyperplane. We compare the confidence scores with that of Logistic Regression in Section 4.2.

4.1.3 Classification Tree

Classification and Regression Trees[6] (Breiman et al., 1984) are popular machine learning models. Given that the data labels in our data set are discrete, we explore the use of a classification tree rather than a regression tree. A classification tree attempts to build a binary tree of feature thresholds learned by minimizing impurity. The two impurity metrics considered in this thesis are gini (G) or entropy[7] (E). Let p_{mk} be the be the proportion of data points with the class k in node m, then

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

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

[4] Data is considered linearly separable if there exists some hyperplane that divides the data such that the binary classifier is capable of perfect classification.

[5] For more detail on kernel methods, and the RBF kernel, please refer to Campbell, 2001, as the content is out of the scope of this thesis.

[6] The equations in this section are taken from Sci-kit Learn's CART page

[7] Minimizing entropy is also the equivalent of maximizing information gain.

These are learned via searching the space, partitioning feature thresholds, and recursively minimizing the summation of the impurity of both the left and right side of the tree. Let R be a node of the tree. We want to select the feature partition $\theta = (j, v)$ such that $x_j \leq v \rightarrow LEFT$ and $x_j > v \rightarrow RIGHT$ minimizes the following:

$$R(\theta) = \frac{n_{left}}{N_R} H(R_{left}(\theta)) + \frac{n_{right}}{N_R} H(R_{right}(\theta))$$

Where H is the impurity metric, N_R are the number of data points considered in node R. This is then performed recursively down the left and right side of the tree until maximum depth is reached or $N_R = 1$.

Random Forest

A Random Forest (Statistics and Breiman, 2001) is an ensemble method of the Decision Trees described above. Rather than train one tree for classification, train a collection of trees by utilizing a subset of the data. Then, the classification result is the majority agreement of terminal nodes from each classification tree in the Random Forest ensemble. It is important to consider the results of an ensemble method for this study because it reduces the problem of over-fitting, which is a concern for small data sets. This doesn't necessarily mean increased performance, but more so that the results from a Random Forest are more trustworthy than that of a single Decision Tree

We tested both Entropy and Gini Impurity as Impurity metrics. While both resulted in near-identical trees when trained on the whole dataset, Gini Impurity performed better by 6% when the results were analyzed via ten-fold cross validation.

4.2 Results

We implemented and ran each of the algorithms described above on the data set with ten-fold cross validation. Given the sample size, we ran cross validation 100 times and averaged the results in order to achieve the best possible characterization of quality. Table 4.1 displays the resulting accuracy from each Machine Learning algorithm. Every algorithm performed better than Logistic Regression, which was the baseline. The Decision Tree algorithm performed the best, and Figure 4.1 illustrates the feature partitions in the tree trained on the entire data set[8].

This tree does not utilize the knee symmetry metric nor the alignment metric. It is possible that knee symmetry is so strongly correlated with the individual joint angles that this feature was redundant. Alignment may not have provided a good metric for splitting any node on the tree due to noise. Jerk provided the most information. The initial split on the maximum jerk of the left knee demonstrates that almost 90% of the poor-form squats had a jerk over 1.6. Furthermore, none of the good-form squats had such a Jerk. The next applications of this research should focus on more jarring movements for this classification problem. They are less noisy and there is a strong correlation between this feature and the proper prediction.

[8]The accuracy results do not correspond to this tree. The tree in Figure 4.1 is the tree should we train on our entire dataset. Accuracy was measured through running 10-fold cross validation using subsets of the data in order to separate testing and training data.

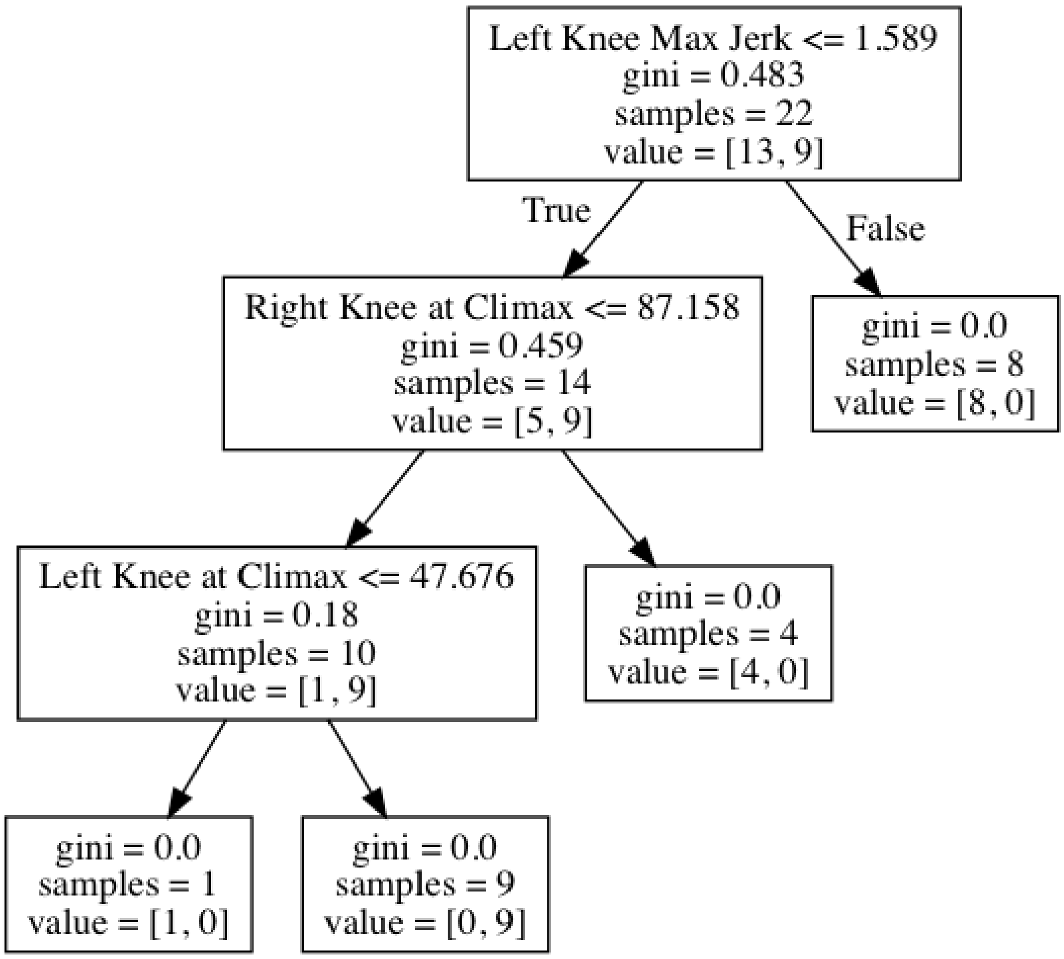

FIGURE 4.1: The resulting decision tree.

Accuracy Results	
Algorithm	Accuracy
Logistic Regression	66.67%
SVM	78.33%
CART	79.42%
Random Forest	73.97%

TABLE 4.1: Algorithm Results

Outside of the Decision Tree results, we measured the confidence in prediction from Logistic Regression and the Support Vector Machine. The average probabilistic confidence in correct classification was 84% for Logistic Regression and 69% for the SVM. The average confidence in incorrect classification was 69% for Logistic Regression and 69% for the SVM. This suggests that the Logistic Regression model is substantially better than the SVM model. The fact that the average confidence of the SVM model is the same for both correctly and incorrectly classified data points is concerning. A successful model should have less confidence surrounding the data they mis-classify, which is strongly the case for Logistic Regression and not the case for the SVM.

Chapter 5

Conclusion

5.1 Discussion

This thesis demonstrated that it is possible to implement Machine Learning to classify quality of movement via motion capture data. The results are sufficiently encouraging to warrant further exploration. The analysis of algorithmic performance combined with prediction confidence suggest that Decision Trees are the algorithm that is most promising. Results from the Support Vector Machine showed less promise, as the confidence in mis-classified points is concerning. Logistic Regression had the worst performance overall, but when confidence is considered, it's performance was reasonable. Decision Trees not only performed best, but the breakdown of feature partition of the tree suggests that this algorithm is capable of yielding sensible and transparent predictions. This transparency is especially valuable for developing a feedback system, which is crucial for improved rehabilitation and clinical purposes.

5.2 Future Work

Although several Machine Learning algorithms showed promise, in order to improve the performance of these algorithms in future research, issues of data quantity and quality must be addressed. It is difficult to trust Machine Learning results without a vast amount of data, and this was not possible in the scope of this thesis. For continued development on the Optitrack, the following should be true for the next generated data set:

- Additional cameras set at hip level in order to maximize marker detection.
- A minimum of a couple hundred data points.
- A minimum of thirty subjects for diversity.

Unfortunately, generating a data set that fits those specs is not easy. Using this system, there are limitations to the number of subjects that can take part. While it is feasible, a more fruitful pursuit of future work is mapping this infrastructure over to a more scalable system like the Microsoft Kinect or even normal video data. We illustrated why we utilized Optitrack for this thesis in Section 1.2, however the importance of data quantity is now so apparent that future applications of this study should utilize a simpler surveillance system. Cao et al., 2016 demonstrates that it is possible to perform real-time skeletal tracking on video-data. The skeleton extracted from their method is not anatomically correct[1]. Should a comparable skeletal tracking algorithm have anatomical precision and work on two-dimensional video data,

[1]The Github repository displays a video of their algorithms results. It is clear that the skeletal model does not have a spine, which is crucial if this is to be used for clinical purposes.

it would be feasible to scrape videos from Instagram and Youtube of Physical Therapy exercises and generate an extensive data set for clinical movement analysis. This is a difficult problem, but is a worthwhile pursuit. It is the most scalable solution and easiest for a patient to implement in their own home.

Appendix A

Code

This appendix includes the relevant code for this thesis. If you would like to run the code and experiment with the data and results, please visit this Github repository[1].

A.1 Data

A.1.1 Variable Definition

The data described in the data section has a lot of additional information that was not utilized in this research. The names of the approximated bone-location variables were defined as follows:

```
LEFT_KNEE = "Ryan:LShin"
RIGHT_KNEE = "Ryan:RShin"
LEFT_HIP = "Ryan:LThigh"
RIGHT_HIP = "Ryan:RThigh"
PELVIS = "Ryan:Hip"
NECK = "Ryan:Neck"
AB = "Ryan:Ab"
CHEST = "Ryan:Chest"
HEAD = "Ryan:Head"
LEFT_SHOULDER = "Ryan:LUArm"
RIGHT_SHOULDER = "Ryan:RUArm"
LEFT_ELBOW = "Ryan:LFArm"
RIGHT_ELBOW = "Ryan:RFArm"
RIGHT_WRIST = "Ryan:RHand"
LEFT_WRIST = "Ryan:LHand"
RIGHT_ANKLE = "Ryan:RFoot"
LEFT_ANKLE = "Ryan:LFoot"
RIGHT_TOE = "Ryan:RToe"
LEFT_TOE = "Ryan:LToe"
```

A.1.2 Accessing the Data

In order to utilize the data, loading it and properly formatting a matrix is the first step. The following code does that:

```
def create_data_matrix(filename):
  #create the data matrix for the optitrack data file
  with open(filename,"r") as f:
    lines = f.readlines()
    if len(lines) == 0:
      print "File Empty"
      return None
    meta = lines[0].strip().split(",")
    columns = [col.strip().split(",") for col in lines[2:7]]
```

[1]The full link should you be reading in paper: https://github.com/ryancsaxe/Thesis

```python
    markers = columns[0][2:]
    labels = columns[1][2:]
    numbers = columns[2][2:]
    pos_bool = columns[3][2:]
    series = columns[4][2:]
    tuples = list(zip(markers,labels,pos_bool,series))
    index = pd.MultiIndex.from_tuples(tuples,names=["Marker","Label","Position","Columns"])
    matrix = np.asarray([line.strip().split(",") for line in lines[7:]])
    frame_nums = matrix.T[0]
    dt = np.diff(np.asarray(matrix.T[1],dtype=np.float32))
    t = np.append([0],np.cumsum(dt))
    data = np.asarray([row[2:] for row in matrix])
  return frame_nums,dt,t,data,index
```

Once that matrix is created, the following function accesses the X,Y,Z coordinates of a given bone (utilizing the variables described in the Variable Definition section).

```python
def get_bone_location(frame,label_name):
  #get the location in space of the bone in question
  bone = frame['Bone'][label_name]["Position"]
  bone_location = np.asarray([bone["X"],bone["Y"],bone["Z"]],dtype=np.float32)
  return bone_location
```

A.2 Feature Extraction

This section includes the code utilized to extract the features described in section 4 of this paper.

A.2.1 Joint Angles

The following takes three bone points and returns the angle related to them. If this function is passed "Left Ankle", "Left Knee", and "Left Hip", the result will be the joint angle corresponding to the Left Knee.

```python
def angle(a,b,c):
  """
  compute angle between 3 points in 3dim space

  the reason we do a -> b and then c -> b is because of the directions we care about.
  so standing straight up should be 180 degrees. If we do a -> b and b -> c, this angle
  becomes zero.
  """
  ab = (a - b) / np.linalg.norm(a - b)
  cb = (c - b) / np.linalg.norm(c - b)
  theta = np.arccos(np.clip(np.dot(ab,cb),-1,1))
  return np.degrees(theta)
```

A.2.2 Jerk

```python
def compute_jerk(series,dt):
  #compute the jerk of a timeseries
  #dt is the same dt returned from section A.1.2
  series = np.asarray(series)
  velocity = np.diff(series)
  acceleration = np.diff(velocity)
```

```
  jerk = np.diff(acceleration)
  return jerk
```

A.2.3 Climactic Angles

The climax of a squat from Skeletal Tracking data can be detected by looking at the point where the pelvis is lowest on the Y-axis. We take in the data generated from section A.1.2, compute the knee joint angles, and extract the angle when the pelvis is lowest

```
for row in data:
  frame = pd.Series(row,index=index)
  #get right side bone locations
  Rankle = utils.get_bone_location(frame,RIGHT_ANKLE)
  Rknee = utils.get_bone_location(frame,RIGHT_KNEE)
  Rhip = utils.get_bone_location(frame,RIGHT_HIP)
  #compute right knee angle
  Rknee_angle = utils.angle(Rankle,Rknee,Rhip)
  #get left side bone locations
  Lankle = utils.get_bone_location(frame,LEFT_ANKLE)
  Lknee = utils.get_bone_location(frame,LEFT_KNEE)
  Lhip = utils.get_bone_location(frame,LEFT_HIP)
  #compute left knee angle
  Lknee_angle = utils.angle(Lankle,Lknee,Lhip)
  Lknee_angles.append(Lknee_angle)
  Rknee_angles.append(Rknee_angle)
  pelvis = utils.get_bone_location(frame,PELVIS)
  hip_y_axis.append(pelvis[1])
lowest_hip_point = hip_y_axis.index(min(hip_y_axis))
Lknee_climax = Lknee_angles[lowest_hip_point]
Rknee_climax = Rknee_angles[lowest_hip_point]
```

A.2.4 Symmetry

We compute the symmetry of the knee joint angle as the average difference in left knee and right knee over the course of the squat timeseries data.

```
knee_metric = np.mean([abs(x[0] - x[1]) for x in zip(Lknee_angles,
    Rknee_angles)])
```

A.2.5 Alignment

```
def _alignment(a,b):
  #function to check alignment of joints/bones
  #the way this works is that only one axis (x,y or z) should yield
    different results
  #for example, if my shoulders are aligned and I am facing the camera,
    the z and y axis should
  # be considered, but the x axis should not.
  # however, if I am facing the side, the y axis and x axis will have
    value and the
  # z axis will not.
  #with this in mind, we look at which axis has the largest delta from
    each joint, and exclude
  # that from the computation because it can be assumed that delta is
    supposed to happen
  # then we take the sum of the delta of the other two axis for the result
  #one of the other two axes should be zero, and perfect alignment would
    have both
  # as zero.
```

```
  difference = abs(a - b)
  max_index = np.argmax(difference)
  difference[max_index] = 0
  return np.sum(difference)

for row in data:
  frame = pd.Series(row,index=index)
  Lhip = utils.get_bone_location(frame,LEFT_HIP)
  Rhip = utils.get_bone_location(frame,RIGHT_HIP)
  Lshoulder = utils.get_bone_location(frame,LEFT_SHOULDER)
  Rshoulder = utils.get_bone_location(frame,RIGHT_SHOULDER)
  shoulder_align.append(utils._alignment(Lshoulder,Rshoulder))
  hip_align.append(utils._alignment(Lhip,Rhip))
#rather than mean, we normalize these by the length of the trial
#because we are interested in alignment over time, not just the average
hip_alignment_metric = np.sum(hip_align)/float(t[-1])
shoulder_alignment_metric = np.sum(shoulder_align)/float(t[-1])
```

A.3 Algorithms

A.3.1 Machine Learning

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression

svm = SVC(gamma='auto')
forest = RandomForestClassifier()
tree = DecisionTreeClassifier()
LR = LogisticRegression()
tree_res = []
svm_res = []
forest_res = []
LR_res = []
#run cross validation multiple times to get best average results
#given the small sample size and variation.
i = 0
while i < 100:
  tree_res.append(cross_val_score(tree,X,Y,cv=10))
  svm_res.append(cross_val_score(svm,X,Y,cv=10))
  forest_res.append(cross_val_score(forest,X,Y,cv=10))
  i += 1
#unlike the other methods, cross validation of Logistic Regression
#will always yield the same results. So no need to include in the loop
LR_res.append(cross_val_score(LR,X,Y,cv=10))
#display average results
print "CART average accuracy:\t\t\t",np.mean(tree_res)
print "SVM average accuracy:\t\t\t",np.mean(svm_res)
print "Random Forest average accuracy:\t\t\t",np.mean(forest_res)
print "Logistic Regression average accuracy:\t\t\t",np.mean(LR_res)
#^these were used for table 4.1.
```

A.3.2 Regression Confidence

In order to assess the validity of Logistic Regression and SVMs, we measured the confidence of the results. This is computed by looking at the average probabilistic confidence in both correct and incorrect predictions.

```
def confidence(results,labels):
    wrong_certainty = 0
```

```
    right_certainty = 0
    wrong_count = 0
    right_count = 0
    #the reason for this outer loop is that
    #because of sample size, we ran the algorithms
    #many times in order to get a less variant accuracy
    #so results is a list of predictions for each time we
    #ran the algorithm we are evaluating the confidence of
    for res in results:
      for i,pred in enumerate(res):
        actual = Y[i]
        if pred[0] > pred[1]:
          val = -1
        else:
          val = 1
        if actual == val:
          right_count += 1
          if val == 1:
            right_certainty += pred[1]
          else:
            right_certainty += pred[0]
        else:
          wrong_count += 1
          if val == 1:
            wrong_certainty += pred[1]
          else:
            wrong_certainty += pred[0]
    wrong = wrong_certainty/float(wrong_count)
    right = right_certainty/float(right_count)
    return wrong,right
```

Bibliography

Almosnino, Sivan, David Kingston, and Ryan Graham (2013). "Three-Dimensional Knee Joint Moments During Performance of the Bodyweight Squat: Effects of Stance Width and Foot Rotation". In: *Journal of applied biomechanics* 29, pp. 33–43. DOI: 10.1123/jab.29.1.33.

Breiman, L. et al. (1984). *Classification and Regression Trees*. new edition **cart93**. Monterey, CA: Wadsworth and Brooks.

Campbell, C. (2001). "Radial Basis Function Networks 1". In: ed. by Robert J. Howlett, Lakhmi C. Jain, and Janusz Kacprzyk. Vienna, Austria, Austria: Physica Verlag Rudolf Liebing KG. Chap. An Introduction to Kernel Methods, pp. 155–192. ISBN: 3-7908-1367-2. URL: http://dl.acm.org/citation.cfm?id=375930.375939.

Cao, Zhe et al. (2016). "Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields". In: *CoRR* abs/1611.08050. arXiv: 1611.08050. URL: http://arxiv.org/abs/1611.08050.

Chang, R. K. Y. et al. (2016). "Kinect-based framework for motor rehabilitation". In: pp. 1–4. DOI: 10.1109/ICORAS.2016.7872606.

Choppin, Simon, Ben Lane, and Jonathan Wheat (2014). "The accuracy of the Microsoft Kinect in joint angle measurement". In: *Sports Technology* 7, pp. 1–8. DOI: 10.1080/19346182.2014.968165.

Cortes, Corinna and VN Vapnik (1995). "Support Vector Networks". In: *Machine Learning* 20, pp. 273–297. DOI: 10.1023/A:1022627411411.

Da Gama, Alana et al. (2015). "Motor Rehabilitation Using Kinect: A Systematic Review". In: *Games for Health Journal* 4, p. 150206061432001. DOI: 10.1089/g4h.2014.0047.

Essery, Rosie et al. (2017). "Predictors of adherence to home-based physical therapies: a systematic review". In: *Disability and Rehabilitation* 39.6, pp. 519–534. DOI: 10.3109/09638288.2016.1153160.

Jun, S. et al. (2013). "Automation for individualization of Kinect-based quantitative progressive exercise regimen". In: pp. 243–248. ISSN: 2161-8070. DOI: 10.1109/CoASE.2013.6654038.

Le, Thi, Minh-Quoc Nguyen, and Thanh-Mai Nguyen (2013). "Human posture recognition using human skeleton provided by Kinect". In: *2013 International Conference on Computing, Management and Telecommunications, ComManTel 2013*. DOI: 10.1109/ComManTel.2013.6482417.

Legaye, Jean and Ginette Duval-BeaupÃšre (2005). "Sagittal plane alignment of the spine and gravity A radiological and clinical evaluation". In: *Acta orthopaedica Belgica* 71, pp. 213–20.

Leightley, D. et al. (2015). "Benchmarking human motion analysis using kinect one: An open source dataset". In: pp. 1–7. DOI: 10.1109/APSIPA.2015.7415438.

Llorens, Roberto et al. (2015). "Tracking Systems for Virtual Rehabilitation: Objective Performance vs. Subjective Experience. A Practical Scenario". In: *Sensors (Basel, Switzerland)* 15, pp. 6586–606. DOI: 10.3390/s150306586.

Lorenzetti, S et al. (2018). "How to squat? Effects of various stance widths, foot placement angles and level of experience on knee, hip and trunk motion and loading".

In: *BMC Sports Science, Medicine and Rehabilitation* 10. DOI: 10.1186/s13102-018-0103-7.

M Sluijs, Emmy, Gerjo Kok, and Jouke van der Zee (1993). "Correlates of Exercise Compliance in Physical Therapy". In: *Physical therapy* 73, 771–82; discussion 783. DOI: 10.1093/ptj/73.11.771.

Maudsley-Barton, Sean et al. (2017). "A Comparative Study of the Clinical use of Motion Analysis from Kinect Skeleton Data". In:

McLaughlin, Thomas M., Thomas J. Lardner, and Charles J. Dillman (1978). "Kinetics of the Parallel Squat". In: *Research Quarterly. American Alliance for Health, Physical Education and Recreation* 49.2, pp. 175–189. DOI: 10.1080/10671315.1978.10615522.

Mousavi Hondori, Hossein (2014). "A Review on Technical and Clinical Impact of Microsoft Kinect on Physical Therapy and Rehabilitation". In: *Journal of Medical Engineering* 2014. DOI: 10.1155/2014/846514.

Myer, Gregory (2014). "The back squat: A proposed assessment of functional deficits and technical factors that limit performance". In: *Strength and conditioning journal* 36,6, pp. 4–27. DOI: 10.1519/SSC.0000000000000103].

Nagymate, Gergely and Rita Kiss (2018). "Application of OptiTrack motion capture systems in human movement analysis A systematic literature review". In: *Recent Innovations in Mechatronics* 5. DOI: 10.17667/riim.2018.1/13.

Schoenfeld, Brad (2010). "Squatting Kinematics and Kinetics and Their Application to Exercise Performance". In: *Journal of strength and conditioning research / National Strength Conditioning Association* 24, pp. 3497–506. DOI: 10.1519/JSC.0b013e3181bac2d7.

Statistics, Leo Breiman and Leo Breiman (2001). "Random Forests". In: pp. 5–32.

Su, Chuan-Jun (2013). "Personal Rehabilitation Exercise Assistant with Kinect and Dynamic Time Warping". In: *International Journal of Information and Education Technology* 3, pp. 448–454. DOI: 10.7763/IJIET.2013.V3.316.

Vakanski, Aleksandar et al. (2018). "A Data Set of Human Body Movements for Physical Rehabilitation Exercises". In: *Data* 3, p. 2. DOI: 10.3390/data3010002.

Yu, Hsiang-Fu, Fang-Lan Huang, and Chih-Jen Lin (2011). "Dual Coordinate Descent Methods for Logistic Regression and Maximum Entropy Models". In: *Mach. Learn.* 85.1-2, pp. 41–75. ISSN: 0885-6125. DOI: 10.1007/s10994-010-5221-8. URL: https://doi.org/10.1007/s10994-010-5221-8.

Zhao, W. (2016). "On automatic assessment of rehabilitation exercises with realtime feedback". In: pp. 0376–0381. ISSN: 2154-0373. DOI: 10.1109/EIT.2016.7535269.

ÅwitoÅski, Adam, Henryk JosiÅski, and Konrad Wojciechowski (2018). "Dynamic time warping in classification and selection of motion capture data". In: *Multidimensional Systems and Signal Processing*. DOI: 10.1007/s11045-018-0611-3.