SAPIENZA
UNIVERSITÀ DI ROMA

Lie Detection Thesis

Facoltà di Ingegneria dell'informazione, Informatica e Statistica

Corso di Laurea Magistrale in Informatica

Candidate

Emanuele Orfanelli
ID number 1383726

Thesis Advisor

Prof. Luigi Cinque

Co-Advisors

Dr. Danilo Avola
Dr. Daniele Pannone

Academic Year 2018/2019

Lie Detection Thesis
Master's thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: emanueleorfanelli@gmail.com

Dedicated to
my Family and Friends

Acknowledgments

fill

Contents

Chapter 1

Introduction

In this chapter we give an overview of the work (Par. 1.1). We start with giving some information about how people perform at detecting lies (Par. 1.2).
We then present a taxonomy of the current state of the art (Par. 1.3) concerning lie detection with particular emphasis on computer vision.
The last section is about the structure of the rest of this work, and our contribution to it (Par 1.4).

1.1 Overview of the work

Deception detection has always been a very interesting topic since it has numerous social and psychological implications in many different fields. In the past 40 years there has been a steady increase in researchers interested in studying deceptive behavior, first from a sociological and psychological point of view, and in more recent years from a technological standpoint, aided by the advance of machine learning techniques.

Our work aims to recognize whether a person is lying or telling the truth by performing a frame by frame analysis on a database of videos, through the OpenFace framework [89], and extracting a subset of the different facial movements performed by the person in the video. After finishing the extraction, the data is submitted to analysis by using different statistical and machine learning techniques.

The dataset we are using was provided by Perez-Rosas et al. [67] and is composed by 121 videos taken from real life trials (Par. 3.2).
These 121 videos have been labeled by the authors with the help of experts to indicate the facial movement occurring in the video. Since those videos are taken from real life trials, the illumination, pose, audio and video features are not homogeneous and often substandard. Substantial work was done to eliminate the parts that are not relevant, and as result many video were trimmed or discarded.

From the remaining videos we have 58 different subjects, meaning that there are more videos of the same subject. This means that it is important to divide the training and test set to avoid just making the classifier "remember" the person. This was done by never having the same subject appear both in the training and test set. After trimming and dividing the videos, we split the data into training and test set based on the subject ID and extracted all the facial movements. Then we utilized machine learning and statistical techniques for classification, and achieved $x\%$ accuracy on the test set.

The result of this work, especially when improved by having a better and larger training dataset, can be useful in different situations, even though it is important to remember that privacy is a real concern, and with this kind of applications it should never be underestimated.
We think the result of this work can be used to aid in airport security, work interview, and many kind of social interactions. It could be eventually used by the public to review a speech of a political candidate or by the police force as an aid to interrogation. Another use can be in trials where people life are at stake and discerning a true or false testimony might be vital. We really hope this work proves to be socially useful.

1.2 People Lie Detection

It's very rare for people to be able to consistently discern between lies and truths, even though we hear lies regularly in the course of our lives. In fact most untrained people perform like chance (50%) when tasked with detecting lies [69], and considering that the ordinary person lies at least twice a day on average [55], and that in this digital age the amount of lies told daily are increasing [39] due to on-line communication making us feel more protected and confident in our deceptions, the problem of detecting lies is an important one and we think it deserves a good deal of research effort.

People's difficulty in detecting lies stem from the lack of objectivity. We are biased by so many factors and skilled deceivers can take advantage of that.
An important reason is that people generally think it is easy to spot liars, underestimating the effort it takes by having too much confidence on their own judgment and capabilities, and by believing in gut feelings instead of empirical clues [71].

In a study by Baker et al. [10] 116 participants were asked to judge 20 videos of people pleading for the safe return of missing family members, half of which were lies where the pleader was the one responsible for the disappearance of the victim. The participants provided confidence ratings, the cues they utilized to make their judgment, and their emotional response to each video.
Weirdly, emotionally intelligent people perform worse at deception detection. This is due to their greater sympathetic feelings towards others (enhanced gullibility) that can often cloud their judgment.

In [21] De Paulo et al. analyze the accuracy of deception judgments from a collection of 206 documents and 24.483 judges. They found that people can differentiate lies and truths with an accuracy of 54%, with a lie detection accuracy of 47% and a truth detection accuracy of 61%. Interestingly, their findings reveal that is easier for people to discriminate lies from the audio cues rather than the visual ones.

In another study [41] 192 students obtained an accuracy of 55.2% on lie detection, with 61% accuracy for guilty suspects and 49% for innocent ones.
Police officers and other trained officials seems to perform better than the average person at detecting lies [93], obtaining around 70% accuracy for distinguishing both lies and truths correctly, supporting the hypothesis that training and practice can make us better at spotting liars.

Lie Catchers	*Number of Lie Catchers*	*Overall Accuracy*	*Liar Spotting Accuracy*	*Truth-teller Spotting Accuracy*
Diverse (206 documents synthesized)	N/A	54%	47%	61%
Undergraduate students	192	55.2%	61.5%	49%
Police officers, CIA and FBI agents, lawyers, college students, therapists, judges, etc.	N/A	50%	N/A	N/A
Secret Service agents	N/A	70%	N/A	N/A
Police officer	37	**72%**	73%	70%
College students	20	50%	N/A	N/A

Figure 1.1. Performance on deceit detection by human observers [86].

1.3 State of the Art

How are lies detected? At the moment there are a lot of different instruments and technologies to detect lies, ranging from analyzing psychological features, to using the polygraph or thermal cameras and machine learning approaches.

In computer vision specifically, lie detection is a somewhat new field of research, and as such it's done using many different techniques, employing not only RGB cameras but also physical sensors and thermal cameras and using voice, gaze, action and body analysis tools, all of this often aided by machine learning techniques, and by combining many of these modalities together to achieve better results.

We now proceed to describe the state of the art, based on the latest researches done in the field, with focus on computer vision:

Speech

Speech is one of the many methods that can be used to recognize if a person is lying. In fact, the speech signal contains linguistic, expressive, organic and biological data [65].

One of the most used indicator of deception in various studies has been the response latency [92], since inventing a lie requires additional cognitive load as opposed to remembering the truth. The authors also notice that habitual lying makes it easier, and conversely often being spontaneous and telling the truth makes lying harder. Another indicator of lying is the speech rate, especially when it's different from the average rate of a specific person [19]. Other verbal cues like grammar usage and word frequency have been used and have achieved high accuracy in psychological researches [70].

Speech analysis can reveal changes that affect behavior, such as stress, emotion, deception etc. by analyzing the pitch and the stress level. When a stressful situation arises, the hormonal levels of the body change, and this causes an increase in blood pressure and heart rate. This in turn affects the muscle in the respiratory system, and so speech is affected [65].

In sound processing, the mel-frequency cepstrum (MFC, Fig. 1.2) is a representation of the short-term power spectrum of a sound. Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up the MFC [98].

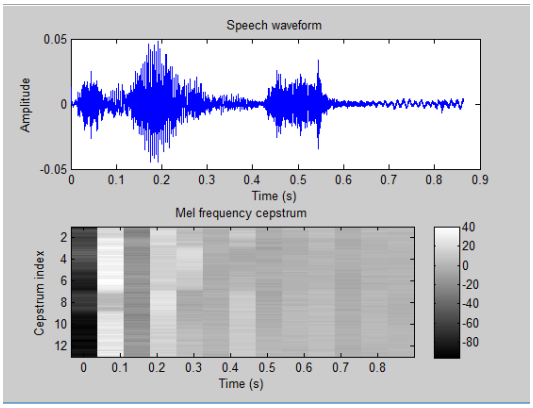


Figure 1.2. Topside is the spoken word, bottom is the MFCC derived from the word

In [61] the authors created a new database by making 40 candidates try to deceive them while telling truthful or deceptive statements for about one to two minutes each. From this experiment they extracted MFCC and pitch, so that they could process them through Matlab's Voice Box.
After acquiring the data, an SVM classifier was trained to classify new data, obtaining an accuracy of Lie and Truth detection from speech audio respectively of 88.23% and 84.52%.

In [67] [60] Perez et al. utilize real life trial data to identify deception, achieving 60-75% accuracy employing a model that extracts features from both linguistic and gesture modalities.

Eyes

Using the eyes to detect lies is one of the most studied approaches, as the eyes hold a significant amount of information regarding our thinking and emotional state [34] (Fig 1.3). We focus on the Computer Vision approaches.

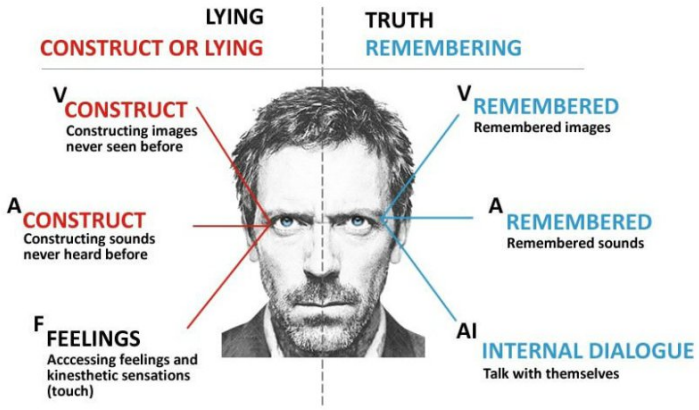


Figure 1.3. Eyes could hold a lot of informations regarding what we are thinking [1].

Moreover is possible to generate a non invasive approach while analyzing the eyes, meaning that subjects do not need to willingly participate or even know if they are being examined or not (the moral matter should be considered in another setting), and there could be no need to have big and expensive machinery, like for example the polygraph or a fMRI machine.

Cognitive load, which is set to increase while lying, is one of the most significant factor for deception detection and can be analyzed through the eyes. Important are also the blink rate, gaze aversion and pupil dilation.

In [36] the authors utilize high speed cameras to record and analyze blink count and blink duration of 50 subjects while asking 10 control questions, to see if there is a variation in them while the subject is being questioned. The authors analyzed the resulting images frame by frame and based on the facial landmarks around the eye they recognize AU45, the action unit for blinking. The results shows that both blink duration and count are increased while lying.

In another study, Leal and Vrij [50] asked 26 people, 13 liars and 13 truth tellers, to lie or tell the truth in a target period, while having a baseline from two preceding periods. The eye blinks during the target and baseline periods and directly after the target period (target offset period) were recorded.
Compared to the baseline periods, lying subjects show a decrease in eye blinks during the target period and an increase in eye blinks during the target offset period. This means that there is a decrease in eye blinks while lying and an increase just after lying. This pattern resulted very different for truth tellers, showing that there is a significant difference in eye blink behavior between truthfulness and deception.

Singh et al. in [82] show that while lying there is an increase in cognitive load and a significant decrease in eye blinks, directly followed by an increase in blink rate as soon as the cognitive demand ceases, after telling the lie. A threshold is set by the authors for this study, either at 26 blinks per minute or it is calculated personally using the average blink rate from a blink detection algorithm. Blink detection is done through MATLAB using the HAAR Cascade algorithm.

Lim et al. study eye gaze [53] to investigate the relation with lie detection. The result supports the theory that an increase in cognitive load leads to a decrement in the number of eye movements while lying.

Bhaskaran et al. measure deception by the deviation from normal behavior at critical points during an investigative interrogation [17] . For starters, a dynamic Bayesian model of the eye movement is trained during a normal conversation with each of the 40 subjects of the experiment.
The remainder of the conversation is then broken into pieces and each piece is tested against the normal behavior. The deviation from normality are observed during critical points in the interrogation and used to deduce the presence of deceit, obtaining an accuracy of 82.5%.

In [73] Proudfoot et al., using latent growth curve modeling, research how the pupil diameter changes over the course of an interaction with a deception detection system. The assumption is that anxiety changes the pupil diameter. The subjects

are presented with crime-relevant target items (possibly incriminating) and non relevant items.
The results indicate that the trends in the changes are indicative of deception during the interaction: pupil diameter is initially bigger for guilty subjects relative to innocent ones. Also the pupil diameter of the participants decreased between subsequent sections of the test, specifically the magnitude of the decrease was greater for guilty subjects who did not see incriminating items.

Neuroscience

Electroencephalogram (EEG) and functional Magnetic Resonance Imaging (fMRI) have been employed for lie detection with good results for a long time, but at the cost of invasiveness, since both methods require big machinery, a suitable environment, and a subject willing to participate.

EEG is a monitoring method that records brain activities based on its potential.

In [81] Simbolon et al, use Event Related Potentials (ERP) to measure brain response directly from thought or perception. Among the many types of signals that constitute the ERP signal, P300 (Fig. 1.4) is the most critical for lie detection, as it appears as a response to meaningful rare stimuli (called odd ball stimuli).
Eleven males of age between 20 and 27 took part in the study. The gathered data were then divided into training and test sets to produce different models. The highest accuracy of 70.83% was reached by a SVM classifier in detecting lying subjects.

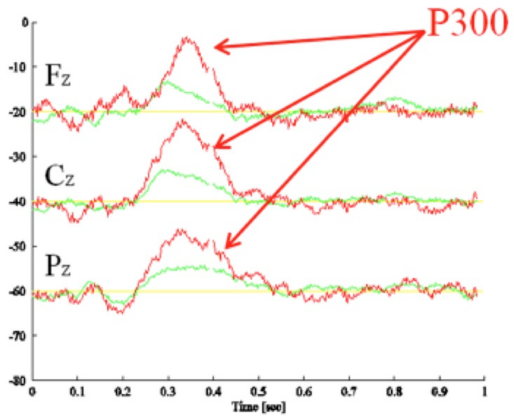


Figure 1.4. EEG of P300 waves image on channels Fz, Cz, and Pz

In [49] twenty people of ages between 22 and 24 years old were subject to a card test using an EEG machine. EEG signals were collected using electrodes attached to the subject's head. The authors used the EEG signals to identify useful frequency bands and to measure lying state based on spectral analysis, with the use of fuzzy reasoning, obtaining 89.5% detection accuracy.

Arasteh et al. [12] utilize an alternative approach to the polygraph, the P300 Guilty Knowledge Test (GTK). GTK is based on the amplitude of P300 ERP wave as an

index for the subject's recognition of concealed information. The Guilty Knowledge Test works on the assumption that among many similar unfamiliar topics, the recognition of familiar ones will be followed by a different response.
62 subjects were part of this experiment and participated in a mock crime followed by the P300 GTK. The authors used empirical mode decomposition (EMD) to extract features from the EEG signal and modeled them trough matlab. A genetic algorithm was then utilized for the feature selection and to handle the dimensionality increase of this approach. The classification accuracy of guilty and innocent subjects was 92.73%.

Another neuro-scientific approach to lie detection is functional Magnetic Resonance Imaging (fMRI). fMRI measures brain activity by detecting changes associated with blood flow (Fig. 1.5). This technique relies on the fact that cerebral blood flow and neuronal activation are coupled. When an area of the brain is in use, blood flow to that region also increases [95].

In most of the experiments with fMRI and lie detection, candidates are instructed to lie and tell the truth in specific situations, and the brain activity from these instances is compared to a baseline condition. The regions showing greater activation for lies than truth are supposed to be the most significant for deception detection. In a recent study it is shown that there is considerable agreement on the significant areas of the brain that regard lying [33].

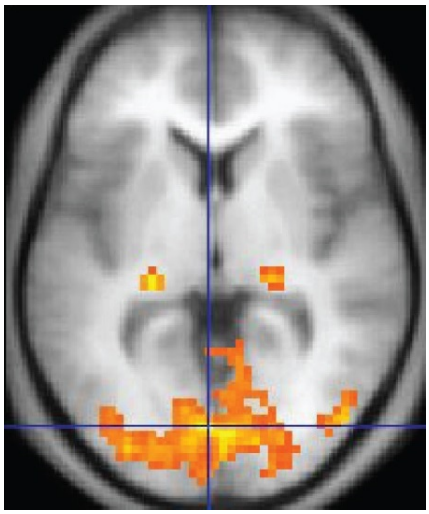


Figure 1.5. fMRI image with yellow areas showing increased activity compared with a control condition [95]

As much as fMRI is useful for lie detection, it presents some shortcomings [26] [4]: many fMRI studies are small, not replicated and done with just a few subjects; there are some contradicting results between some studies; most of the studies are not done in a contest of high stake deception, but in a controlled environment where subjects are asked to lie about some topic or event, but often without a real interest in being deceitful. Another important point is that the fMRI approach requires collaboration and expensive equipment to be carried out, so this is a very limiting

factor.

Thermal Imaging

In thermal imaging, thermal features are extracted from the face using a high definition thermal camera. The objective is to analyze what kind of differences occur when a subject responds truthfully or deceptively to particular stimuli.

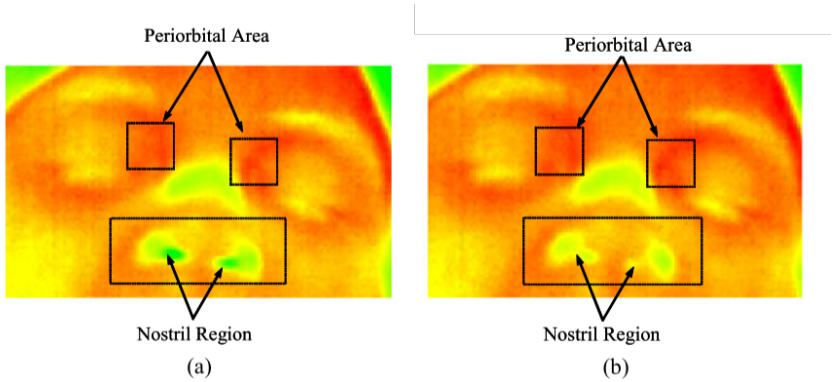


Figure 1.6. Examples of thermal images during (a) questioning and (b) answering [87]

According to a recent study [7] examining 30 subjects, the most relevant zones for deception detection, seen by utilizing thermal equipment, and located in the facial area are the forehead and periorbital regions.
In this study the subjects were registered with a thermal camera for one minute to extract the baseline features, and after that the interviewers asked a series of questions. A thermal map was created from the registrations using the Hue Saturation Value to differentiate between lies and truths.

In [75] the authors set up an experiment to collect 492 responses from 25 participants, using a deception scenario requiring the subjects to learn a story, provided by the authors, and practice their stories before hand by giving them sample questions, so that the cognitive load would increase when receiving new and unseen questions during the interview.

At the beginning of the interview four baseline questions were asked to register the initial thermal state of the subjects, and then a series of questions were asked, with answers both present and missing from the provided story. After extracting the periorbital region's thermal variation, a k-nearest neighbor classifier was used, with an 87% accuracy in predicting lies or truths.

In [87] data are gathered non-intrusively from the nostril and periorbital regions using two dimensional far infrared cameras. The study lasted for two years and covered 18 tests subjects involved in real crime cases. The temperature is extracted and converted in change in blood flow velocity, and a signature of the respiration pattern is determined in terms of the ratio of the measured maximum and minimum temperatures in the nostril area. The classification rate for this study is 88.5%.

Multimodal

After seeing all these different techniques to detect lies, it's only natural for researchers to try and fuse or aggregate some of them to see if it's possible to achieve better results [8].

In [6] Abouelenien et al. collect data from a dataset of 30 subjects to examine thermal and visual clues of deception. Their aim is to identify the regions that offer higher capability of detecting deceit.
The method employed uses the CERT (Computer Expression Recognition Toolbox) to detect facial expression and encodes them through Action Units, a way to codify almost any kind of muscle movements on the face.
To extract thermal features they create a thermal map using gray-scale and Hue Saturation Value. They also calculated normalized blinking rates and the mean head orientation angle along the entire length of the response. In addiction over 60 physiological features are extracted and stored with the use of sensors and other RGB cameras. The experimental results show that the non-contact feature fusion model outperforms traditional physiological measurements, and that the forehead region is one of the most promising areas to gather information for deception detection using thermal imaging.

In a following paper [5] Abouelenien et al. explore a multimodal deception detection approach comprised of physiological, linguistic, and thermal features on a new dataset of 149 recordings. They set out to determine which is the most discriminative region of the face to differentiate between truth and lies based on thermal imaging, and perform feature analysis using a decision tree model. The results show that the forehead could be a better indicator of deceit than the periorbital area. The physiological features did not contribute very much, while the linguistic feature played a critical role, where self-referencing and exaggeration words where big indicators of deceit. The overall accuracy of the system is 70%.

Another example of multimodal detection is found in [103] where Wu et al. develop a framework to automatically detect deception in videos of trials. They utilize three modalities: vision, audio and text. For vision, they employ various classifiers trained on low level video features to predict human micro-expressions, and to successively predict deception. Interestingly, IDT (Improved Dense Trajectory) features, often used to recognize actions in videos, are good predictors of deceptive behavior.
The authors decided to fuse the score of the classifiers on IDT and micro-expressions to boost the performance. Regarding text, the transcript of the considered videos are analyzed, but the performance increase is very marginal.
For speech, they integrated the vision side with Mel Frequency Cepstral Coefficient (MFCC) features analysis from the audio, boosting the performances significantly, reaching an AUC of 0.877.

Noje et al. [64] set up a study with ten subjects to observe the potential of head movements in lie detection. They built an application to detect head movement and position by performing a frame to frame analysis on a video stream. A correlation was made between head movement/position and the identification of lies. The

results of the study are not concluding as this data can't be utilized without being incorporated with other modalities such as voice, gaze, words, expressions et cetera.

In [67] [60] Perez et al. utilize real life trial data to identify deception, achieving 60-75% accuracy employing a model that extracts features from both linguistic and gesture modalities.

Facial Expression and Micro-Expressions

Facial Expressions are one of the main methods that we use to express our emotions, and are developed since the first months of our life to help us communicate our feelings to others. But what happens when we want to hide our emotions instead?

Facial micro-expressions are very fast (1/2 to 1/25 of a second) and involuntary expressions that come up on the human face when we are trying to suppress or hide an emotion, and are very difficult to control using just one's willpower [31].

Figure 1.7. Six basic Facial Expressions in adults and children [72].

Studying and classifying micro-expression is very valuable and has many applications, especially in psychology and forensic sciences, but it is a hard feat as the duration is very short and the intensity is low.
Micro-expressions have been studied since 1966 to recognize and distinguish real or fake emotions, initially by Haggard and Isaacs [37], and three years later by Ekman and Frisen [30].

Substantial work on Micro-Expressions has been done by Pfister, Li et al. In [68] they collaborated with psychologists to design an induced emotion suppression experiment. The data was collected with a high speed camera to be able to register the micro expressions of the subjects. A Temporal interpolation model was used to counter the shortness of the video length, while multiple kernel learning, random forest and SVM were used to classify micro expressions reaching an accuracy of 71.4%.

The lack of a big and well formed database was one of the biggest hindrance to their research. To solve this problem in [51] they reveal a new dataset, the Spontaneous Micro-expression Database (SMIC), which includes 164 micro-expression video clips taken from a group of 20 participants.
They used two high speed cameras to record the face of the subjects while they were watching a selection of videos that induced strong emotional response, and they had to try and suppress those emotions. After the video the subjects had to answer about the emotions they felt while watching it. The data were then segmented and labeled by two annotators.

A study of spontaneous micro expression spotting and recognition methods was done in [52]. A new training-free method, based on feature difference contrast for recognizing micro-expressions was presented. The features are extracted from the video using Local Binary Pattern (LBP) and Histogram of Optical Flow.
To recognize the Micro expressions, the authors performed face alignment and temporal interpolation, then trained an SVM classifier.
This micro-expression framework was tested on the SMIC and CASMEII database with very good results. After combining micro-expressions spotting and recognition they released a new micro-expression analysis system (MESR) that is able to recognize micro-expressions from spontaneous video data.

Owayjan et al. [66] designed a lie detection system using micro-expressions. At first an embedded video system is used to record the subject interview. The video stream is converted into frames, and each frame is processed in four stages: converting the images, filtering out useless features, applying geometric templates and finally extracting the measurements to detect the micro-expressions.
Results show that up to eight facial expressions can be recognized, and that lies can be discerned with high precision.

In [46] Kawulok et al. explore how to exploit fast smile intensity detectors to extract temporal features using a SVM classifier. Using exclusively a face detector, without localizing or tracking facial landmarks, they analyze the smile intensity time series. They then employ an SVM classifier to improve training from weakly labeled datasets. Then, to train the smile detectors, they use uniform local binary pattern features. This allows to detect, in real time, between spontaneous or posed expressions.

Su et al. [86] aim to test the validity of facial clues to deception detection in high-stakes situations using computer vision approaches. By using invariant 2D features from nine separate regions of the face they perform facial analysis on eye blink, eyebrow motion, wrinkle occurrence and mouth motion, integrated with a facial behavior pattern vector. Training a Random Forest to classify the patterns into deceptive or truthful, they achieved a 76.92% accuracy.

1.3.1 Action Units

Action Units (AUs) are defined as a contraction or relaxation of one or more muscles. They have been used in the Facial Action Coding System (FACS), a system

developed initially by Hjorztsjö [43] and then improved by Freisen and Ekman [29], that categorizes all the facial movements by their appearance on the face.

Using FACS it's possible to code nearly any facial expression by deconstructing them into Action Units. For example the Duchenne smile (felt smile) is a combination of AU6 (orbicularis oculi, pars lateralis) and AU12 (zygomatic mayor) as we can see from fig. 1.8).

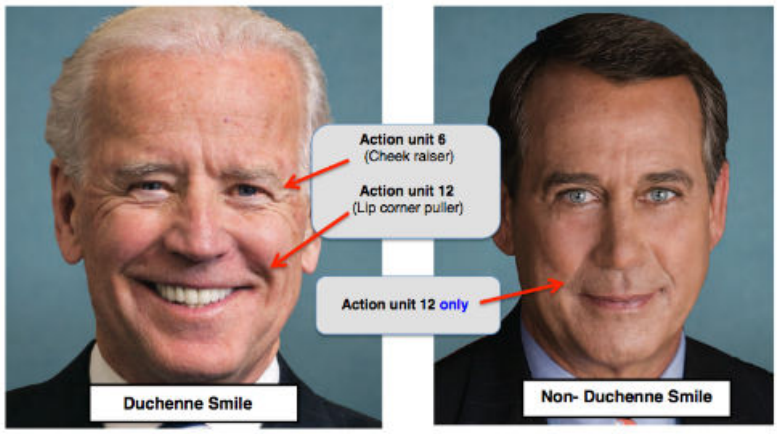


Figure 1.8. Duchenne smile (real) vs non Duchenne (fake).

In the FACS there are 44 AUs categorized with five level of intensity (A to E). Most of the AUs have an onset, peak and offset phase.
By using AUs is possible to recognize emotions based on the combination of AUs displayed (Fig. 1.9).

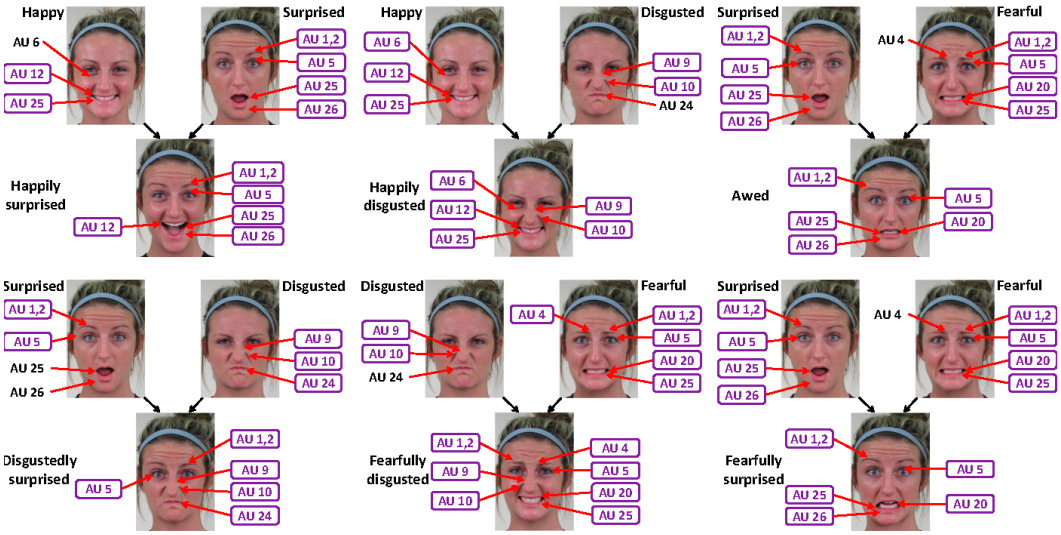


Figure 1.9. AUs of six compound facial expressions of emotion. The AUs of the basic emotions are combined to produce the compound category [28].

Substantial work on AU classification and intensity estimation has been done in [13]

by Baltrusaitis et al. (Fig. 1.10) while developing the OpenFace [15] system.
Baltrusaitis et al. developed a real-time Facial Action Unit occurrence and intensity detection system based on appearance and geometric features, using Histogram of Oriented Gradients, Shape Parameters and Landmark Locations. They achieved good results by using a median based feature normalization technique so that they could account for person specific neutral expressions (Par. 3.1).

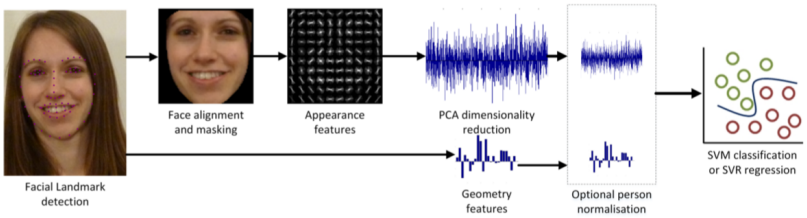


Figure 1.10. OpenFace AU detection and intensity estimation pipeline [13].

In [40] Hao et al. explore the dependencies between AUs, and propose a new AU recognition method consisting of a three layer Bayesian network where the top two layers are latent regression Bayesian networks (LRBN).
LRBN are graphical models consisting of a visible layer representing the ground truth for AUs and a latent one. LRBN is able to capture the dependencies between AUs. They tested this system on the CK+ [54], SEMAINE [59] and BP4D [105] database, demonstrating that their approach can accurately capture AU relationships.

In [24] the authors try to model three fundamental aspects of automated AU detection: spatial representation, temporal modeling, and AU correlation. They proposed an approach using a hybrid network architecture.
Spatial representation is extrapolated by using a Convolutional Neural Network (CNN). For temporal dependencies Long Short-Term Memory Neural Networks (LSTM) are stacked on top of the CNN.
The output of LSTMs and CNNs are then fused together to predict 12 AUs on a frame to frame basis. This network was then tested on the GFT and BP4D [105] dataset, gaining improvements over standard CNN.

De la Torre et al. [22] [23] tackle this problem by personalizing a generic classifier without requiring additional labels for the test subject (unsupervised). They use a transductive learning method, referred as Selective Transfer Machine (STM), that personalizes a generic classifier by attenuating people specific biases. This is done by learning a classifier and re-weighting the training samples most relevant to the subject.
The performance of STM were compared to generic classifiers and cross-domain learning methods and evaluated on CK+ [54], GEMEP-FERA, RUFACS and GFT dataset, and STM is shown to outperform the generic classifiers on all datasets.

Many approaches to automatic facial AU detection fail to account for individual difference in morphology and behavior for the target person. This is a hard problem because it's difficult for classifiers to generalize to very different subjects, and training a person-specific classifier is neither feasible (very low training data) nor

has particular theoretical interest.

1.4 My Contributions

1) This thesis has set a precedent for future research on high-stakes deception detection using facial action units. As far as we know, there is no computer vision research that has testified the validity of facial action unit as indicators of high-stakes deception. As will be seen in Chapter X, our results are promising, implying the research potential of this topic in the future. 2) The proposed method is at the forefront of analyzing facial expressions in unconstrained environments. Since the videos in our database were mostly collected from YouTube, certain uncontrollable factors add to the difficulty of their analysis.
These totally unconstrained and spontaneous videos are subject to temporal variations in illumination, head pose and facial occlusion. However, in the current literature, little research has been conducted to address these issues with regard to facial expression analysis. Instead of seeking a solution to solve them, almost all of the studies to date have excluded certain data that were not ideal for the purpose of analysis. In comparison, the proposed method seeks to address the deception detection problem, but in the presence of exactly these factors.

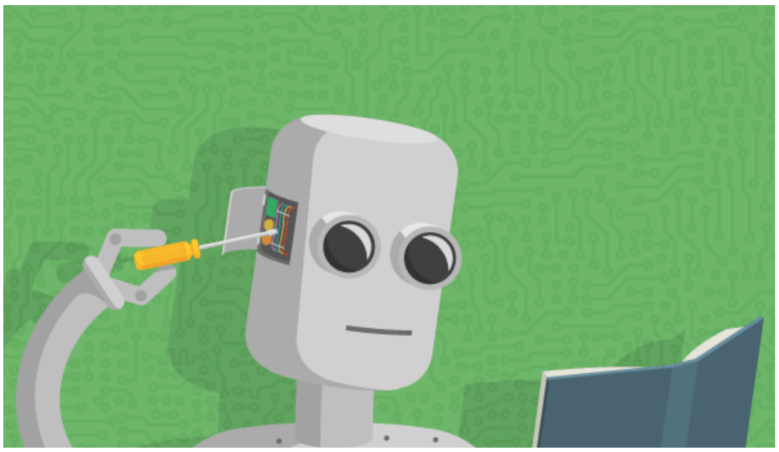
FUTUREU USE - app per telefono video -> result

Chapter 2

Architecture

In this chapter we will give a brief introduction of Machine Learning (2.1), explaining Classification (2.1.3), Regression (2.1.4), Supervised (2.1.1) and Unsupervised (2.1.2) learning, proceeding then to explain different techniques such as SVM (2.4), which is the most important technique used for this thesis, and Random Forests 2.3.

2.1 Machine Learning



Machine Learning is a subfield of Artificial Intelligence that uses statistical techniques to provide computers with the ability to progressively improve performance on different tasks using data, while not being explicitly programmed [25].

The applications for machine learning are huge and diverse, and range from character recognition to email filtering, with lots of applications in computer vision, such as image recognition and classification.

Machine learning also focuses on making prediction on data, by utilizing techniques taken from statistics mathematical optimization. This has many applications,

ranging from health care by predicting risk factors for diseases or gaining insights for prevention, to sports or politics where actual results can be predicted.

The field of machine learning is subdivided in two broad categories (Fig. 2.1), supervised learning (2.1.1) and unsupervised learning (2.1.2) based on whether the data is labeled or not.

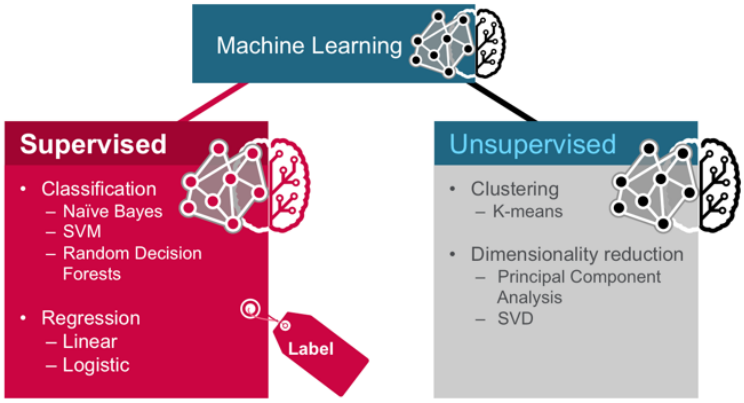


Figure 2.1. Machine Learning Subfields [58]

Another classification is based on the kind of output that the users wish to obtain, mainly Classification (2.1.3), Regression (2.1.4) or Clustering.

2.1.1 Supervised Learning

When using supervised learning we want to learn a function that maps an input to an output, based on example input-output pairs [85].
This means that we need labeled training data, consisting of a vector of input objects and an output value. The objective is to correctly classify the new data based on the previously analyzed training pairs.

The general steps to solve a supervised learning problem are (Fig. 2.2):

1. Understand what kind of training example to use.
2. Gathering a training set.
3. Model the training set to be fed as input to the algorithm by choosing which features to use and how to represent the data.
4. Choose what kind of algorithm can best train the model.
5. Run the algorithm and evaluate the resulting accuracy on the test set

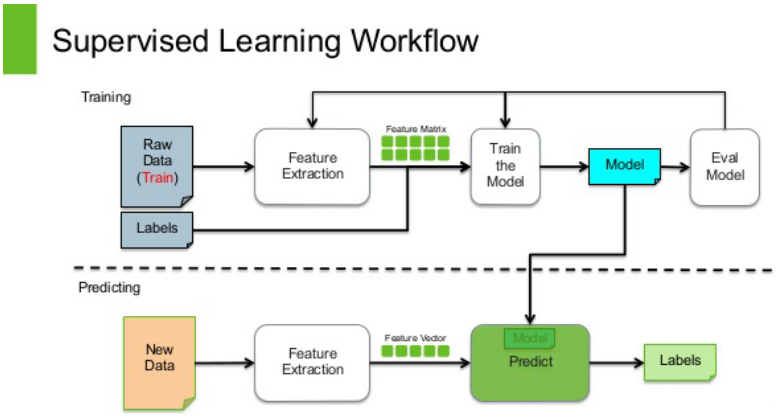


Figure 2.2. Supervised Learning Workflow [56]

There are important considerations to make when using a supervised learning approach:

- **Dimensionality of Input**:
  When the input feature vectors are very big there could be problems in learning the function, even if not all features contribute significantly to the function. This happens because the data depends on too many variables and this could cause high variance.
  To avoid this, it is important to reduce the number of features through manual removal or using feature selection algorithms. This usually improves the accuracy of the classifier.

- **Overfitting and Underfitting** [20]:
  Overfitting happens when the algorithm adapts too much on the training data and is no longer able to make accurate predictions on the test data (Fig. 2.3). This usually happens when there is an excessive number of parameters than what can be justified by the data [32].

  Underfitting is the opposite: a model is not able to correctly capture the structure of the data, for example when fitting non linear data with a linear model.
  A common way to avoid overfitting is to resample the data using different techniques, commonly k-fold cross validation or leave-one-out. Other methods include feature removal, early stopping, regularization.

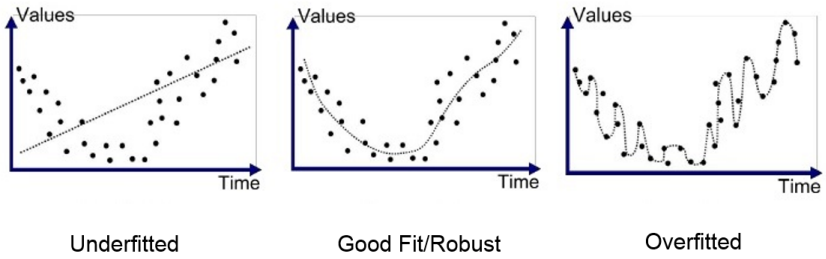


Figure 2.3. Example of overfitting and underfitting [16]

- **Bias-Variance Tradeoff** [77]:
  Suppose we have different (but equally good) training set:
  an algorithm is biased for input x if, when trained on each of these data sets, it is consistently inaccurate at predicting the correct output for x.
  A high bias indicates that the data points tend to be very close to the mean, and to each other.

  A learning algorithm has high variance for a particular input x if it predicts output values that are correct on average but inconsistent when trained on different training sets.
  Basically, a high variance indicates that the data points are very spread out from the mean, and from one another (Fig. 2.4).

  The prediction error of a classifier is related to the sum of bias and variance, so generally there is a tradeoff between them. Low bias means that it fits the new data well, but if the bias is too low it will fit each training set differently and so result in high variance.

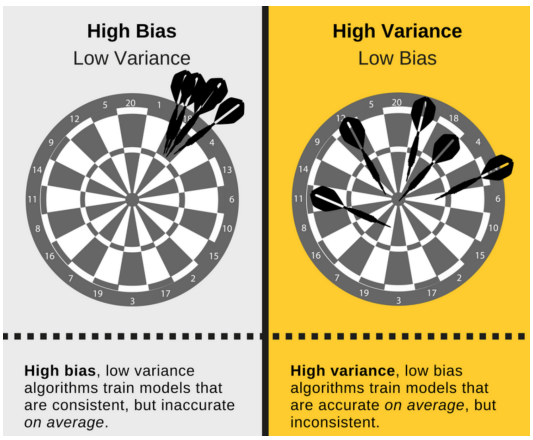


Figure 2.4. Example of Bias-Variance Tradeoff [18]

There are many algorithms used to perform supervised learning tasks, the most commonly used are:

- Linear Regression (2.1.4)

- Logistic Regression 2.2
- Naive Bayes
- Linear Discriminant Analysis
- Decision Trees
- k-Nearest Neighbor
- Neural Networks
- Support Vector Machines (2.4)

2.1.2 Unsupervised Learning

Unsupervised learning is the subfield of Machine Learning tasked with inferring a function from the analysis of unlabeled (not classified) data. Being unclassified there is also a difficulty in evaluating the accuracy of the model.
Usually items are grouped by some measure of similarity, like for example in k-means clustering (Fig. 2.5).

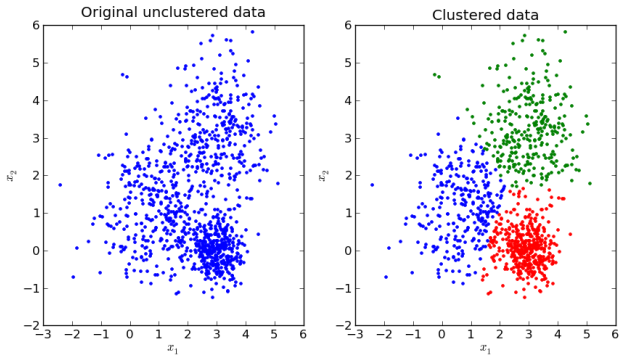


Figure 2.5. Example of Unclustered and Clustered data [45]

These are the most widely used unsupervised learning algorithms:

- Clustering
  - k-means
  - mixture models
  - hierarchical clustering
- Anomaly detection
- Neural Networks
  - Autoencoders
  - Deep Belief Nets
  - Hebbian Learning
  - Generative Adversarial Networks
  - Self-organizing map
- Expectation–maximization algorithm (EM)
- Method of moments
- Blind signal separation techniques
  - Principal component analysis,

– Independent component analysis,
– Non-negative matrix factorization
– Singular value decomposition.

2.1.3 Classification

In machine learning, classification is the problem of identifying in which of a set of categories a new observation belongs, based on a training set of data containing observations whose category is known in advance. A common example is classifying an email as spam or not [102].

Classification is considered an instance of supervised learning, based on instances where a training set is available. As for unsupervised learning, classification would be clustering since it groups data into categories based on similarity, but without knowing the label of the data.

The observations, called features or explanatory variables, take different types based on the value. They can be categorical, numerical or ordinal, or be compared by similarity between previous observations using some kind of distance function.
The observations representing the categories to be predicted are called explanatory variables (or regressors, or independent variables).
The classifier is the algorithm that implements the classification, or a function that maps input data to a category.

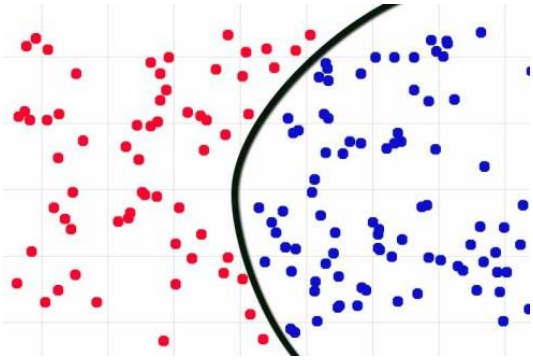


Figure 2.6. Example of data division by a Classification algorithm

Many classification algorithm, such as SVM (2.4), logistic regression 2.2, LDA (Linear Discriminant Analysis) or perceptron, can be described using a linear function assigning a score to each category c, by doing the dot product of the feature vector of an instance with a vector of weights, thus combining them. The predicted category will be the one with the highest score. This function is called linear predictor function and has this general formula:

$$score(X_i, c) = \beta_c \cdot X_i \tag{2.1}$$

where X_i is the feature vector of instance i and β_c is the vector of weights of category c. This kind of algorithms are known as linear classifiers.

2.1.4 Regression Analysis

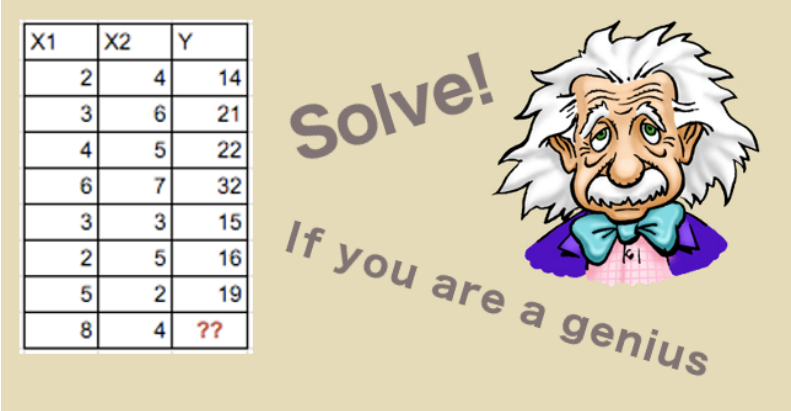


Figure 2.7. Regression Analysis can solve this! [11]

Regression analysis is used in statistics and machine learning to estimate the relationships among variables.
More specifically, it focuses on the relationship between one dependent variable and more independent variables (also called predictors), and generally is used to estimate the average value of the dependent variable when the independent variables are fixed, by calculating a regression function.

By understanding the relationship between dependent and independent variables it's also possible to perform predictions on future data based on new inputs, or on changes to the old ones [101].

In regression analysis, it is also of interest to characterize the variation of the dependent variable around the prediction of the regression function using a probability distribution.

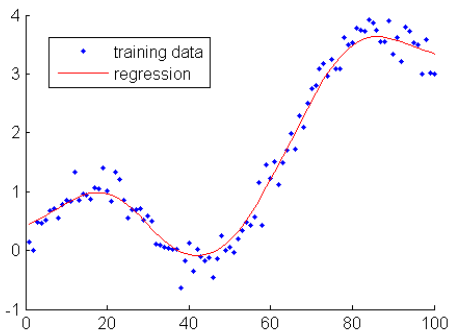


Figure 2.8. Example of Regression Analysis

Regression Model

A general regression model uses the following variables and parameters:

The unknown parameters, β, that represents either a scalar or a vector.
The independent variables, X.
The dependent variable, Y.

A regression model relates Y to a function of X and β.

$$Y \approx f(X, \beta) \tag{2.2}$$

This is usually formalized as

$$E(Y|X) = f(X, \beta) \tag{2.3}$$

If β is of length k and the number of observed data points is enough ($N > k$), then it's possible to estimate a unique value for β that best fits the data.

If this is the case, regression analysis provides the means to find a solution for unknown parameters of β to, for example, apply the method of least squares.

To apply regression the data must abide by some assumptions, generally:

- The sample is representative of the population for the inference prediction.
- The error is a random variable with a mean of zero.
- The independent variables are measured with no error.
- The independent variables (predictors) are linearly independent.
- The errors are uncorrelated.
- The variance of the error is constant across observations.

2.1.5 Linear Regression

Linear regression is the most basic case of Regression Analysis, and it is a useful tool for predicting a quantitative response. Also, most of the newer approaches to regression are often a generalization or extension of linear regression.

Linear regression is a linear approach to modeling the relationship between a dependent variable and one or more independent variables. The simplest case of linear regression (LR) is when there is only one independent variable, and is called simple linear regression [96] and has this form:

$$y_i \approx \beta_0 + \beta_1 x_i + \varepsilon_i \tag{2.4}$$

where ε_i is the error for the ith observation. The error is a catch-all for what we miss with this simple model, since it's very probable that the true relationship is not linear, and that there may be other variables that influence y [35].

The point of LR is to estimate the β coefficients to make predictions. To do so, we utilize of the training dataset to produce estimates $\widehat{\beta_0}$ and $\widehat{\beta_1}$ for the model coefficients. We can then make predictions by computing:

$$\widehat{y_i} \approx \widehat{\beta_0} + \widehat{\beta_1} x_i \tag{2.5}$$

$e_i = y_i - \widehat{y_i}$ is the difference between the true value and the prediction for an observation, and it is called residual (Fig. 2.9).

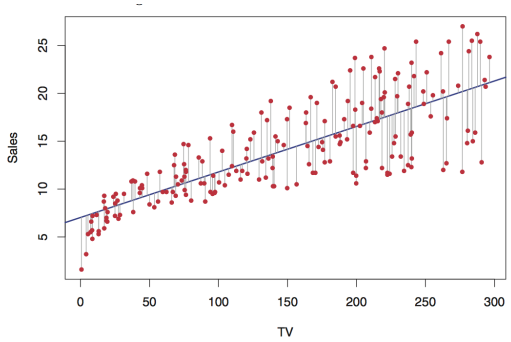


Figure 2.9. The fit is found by minimizing the sum of squared errors. Each gray line segment represents an error, and the prediction makes a compromise by averaging their square [35]

The most common method for estimation is called least squares. This method obtains parameter estimates that minimize the sum of squared residuals (SSR):

$$SSR = \sum_{i=1}^{n} e_i^2 \tag{2.6}$$

The minimizers are

$$\widehat{\beta_1} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} \tag{2.7}$$

$$\widehat{\beta_0} = \bar{y} - \widehat{\beta_1}\bar{x} \tag{2.8}$$

where $\bar{x}$ and $\bar{y}$ are the mean of x and y.

If we assume that the error term has constant variance, the estimate of the variance of the error is given by:

$$\widehat{\sigma}_\varepsilon^2 = \frac{SSR}{n-2} \tag{2.9}$$

And is called mean square error (MSE) of the regression. The denominator is the sample size reduced by the number of model parameters estimated from the same data, $(n-p)$ for p regressors or $(n-p-1)$ if an intercept is used [84]. In the case of simple linear regression $p = 1$, so the denominator is $n-2$.

We can then estimate the standard errors for the parameters, that tell us the average amount that the estimate differs from the actual value.

$$\widehat{\sigma}_{\beta_1} = \widehat{\sigma}_\varepsilon \sqrt{\frac{1}{\sum(x_i - \bar{x})^2}} \tag{2.10}$$

$$\widehat{\sigma}_{\beta_0} = \widehat{\sigma}_{\varepsilon}\sqrt{\frac{1}{n} + \frac{\bar{x}^2}{\sum(x_i - \bar{x})^2}} \tag{2.11}$$

Standard errors can be used to compute confidence intervals. A 95% confidence interval is defined as a range of values such that with 95% probability, the range will contain the true unknown value of the parameter.

The general multiple regression model, where there are p independent variables, follows this formula:

$$y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i \tag{2.12}$$

where x_{ij} is the i-th observation on the j-th independent variable.

2.2 Logistic Regression

In regression analysis, logistic regression is a method for estimating the parameters of a logistic model. A logistic model is one where the log-odds of the probability of an event is a linear combination of independent or predictor variables.
The two possible dependent variable values are often labeled as "0" and "1", "true" or "false", "pass" or "fail" etc, and they represent binary outcomes (Fig. 2.10).
It's possible to generalize the binary logistic regression model to more than two levels (outcomes) of the dependent variable [97].

The binary logistic model works by estimating the probability of a binary response based on one or more predictor variables.
With logistic regression is possible to say how much that the presence of a risk factor increases the odds of a given outcome.
The model itself simply calculates the probability of output in terms of input, and it is not specifically a classifier, but it can be used to classify data by choosing a threshold value, for example if the probability is $\geq 50\%$ the class is "1" otherwise is "0".

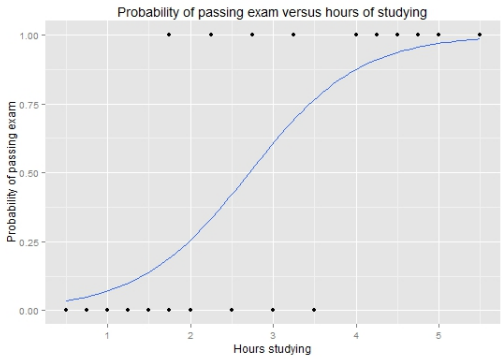


Figure 2.10. Graph of a logistic regression curve showing probability of passing an exam versus hours studying [97]

Logistic Function

The logistic function to estimate the probabilities is a sigmoid function (exemplified in fig. 2.11).
A sigmoid function takes an input from the real numbers and outputs a value between 0 and 1.

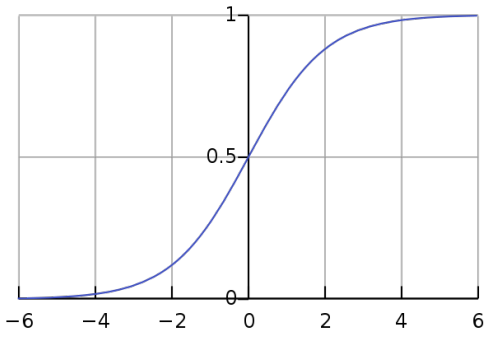


Figure 2.11. Example of a Sigmoid shaped function

We want to model the relationship between $p(X) = Pr(Y = 1|X)$ and X. To model this relationship we use the a logistic function:

$$p(X) = \frac{e^{\beta_0} + e^{\beta_1 X}}{1 + e^{\beta_0} + e^{\beta_1 X}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}} \tag{2.13}$$

Equation 2.13 can be manipulated into:

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0} + e^{\beta_1 X} \tag{2.14}$$

$\frac{p(X)}{1-p(X)}$ is called odds, and can take any value between 0 and ∞.

By doing the logarithm of 2.14 we get:

$$\log \frac{p(X)}{1 - p(X)} = \beta_0 + \beta_1 X \tag{2.15}$$

The coefficients β_0 and β_1 in equation 2.13 are unknown, and must be estimated based on the available training data.

To fit the model of equation 2.15, the maximum likelihood method is commonly used.

Intuitively we want to estimates β_0 and β_1 such that the predicted probability $\hat{p}(x_i)$ for a specific case corresponds as closely as possible to the observed class for that case.

This can be achieved by using the likelihood function:

$$\ell(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'})) \tag{2.16}$$

The estimates $\hat{\beta}_0$ and $\hat{\beta}_1$ are chosen to maximize this likelihood function.

Maximum likelihood is a very general approach that is used to fit many of the non-linear models. For example in linear regression (Ch. 2.1.5), the least squares approach is a special case of maximum likelihood.

Once the coefficient have been estimated is pretty simple to compute the probability:

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0} + e^{\hat{\beta}_1 X}}{1 + e^{\hat{\beta}_0} + e^{\hat{\beta}_1 X}} \tag{2.17}$$

2.3 Random Forest

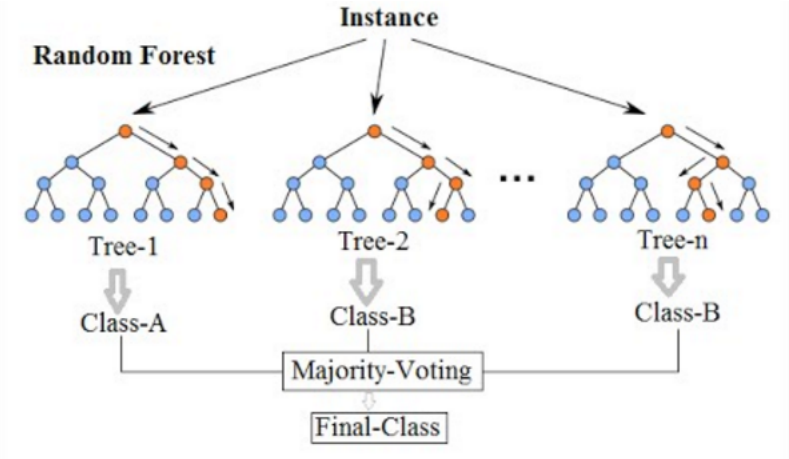


Figure 2.12. Example of Random Forest Classification [48]

Random forests are an ensemble learning method used mainly for classification and regression. They work by building a lot of decision trees while in the training phase, and then outputting the class that is the mode of the classes for classification, or the mean prediction of the individual trees for regression [100].

Random Forest (RF) is preferred to decision trees because decision trees suffer from high variance. By using tree bagging, which is a commonly used technique to reduce variance, we can correct decision trees' habit of overfitting to their training set [42].

Decision Trees

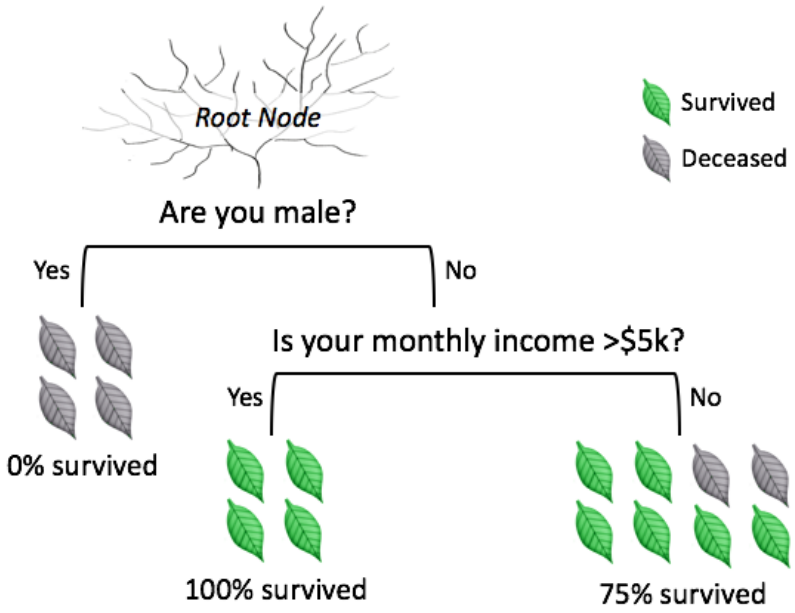


Figure 2.13. Example of a Decision Tree [63]

Decision trees are a popular method for various machine learning tasks and can be used for either classification or regression. The aim of decision trees is to create a model that predicts the value of a target variable based on several input variables. Each node of the tree corresponds to one of the input variables. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf [94].

A decision tree is a simple representation for classifying examples. The arcs coming from a node labeled with an input feature are labeled with each of the possible values of the target or output feature or the arc leads to a subordinate decision node on a different input feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes.

Decision trees, however, are often inaccurate. Specifically, by growing trees very deep, they tend to learn highly irregular patterns by overfitting to the training set (i.e. have low bias, but very high variance).

Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance.
This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model [42].

Tree Bagging

Given a set of n independent observations $X_1, \ldots, X_n$, each with variance σ^2, the variance of the mean $\overline{X}$ of the observations is given by σ^2/n.
This means that averaging a set of observations reduces the variance. So to reduce the variance we can take many training sets from the population, build a prediction model using each training set, and average the resulting predictions.

Basically we could calculate $\hat{f}^1(x), \hat{f}^2(x), \ldots, \hat{f}^B(x)$ using B separate training sets, and average them in order to obtain a single model with low-variance [35].

This solution is not practical since we normally lack more than one training set.
Instead we utilize *bagging*: we can take repeated samples from the training set in order to generate B different bootstrapped training sets. We then train our method on the b*th* bootstrapped training set, and then average the resulting predictions.

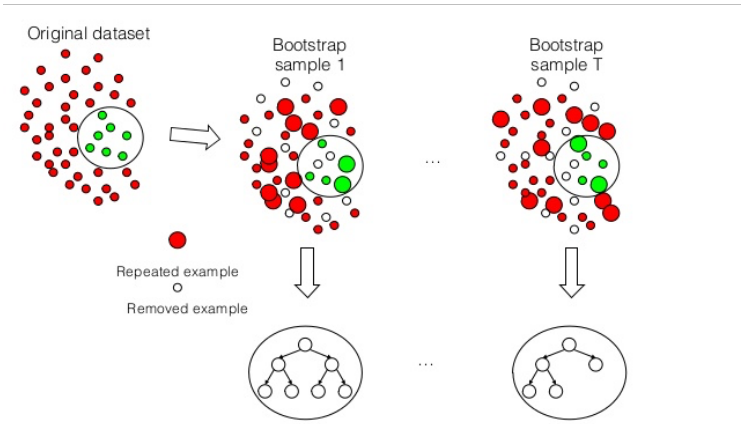


Figure 2.14. Bagging Example [2]

To apply bagging to regression trees specifically, we simply construct B regression trees using B bootstrapped training sets, and average the resulting predictions. These trees are grown deep, and are not pruned. Hence each individual tree has high variance, but low bias. Averaging these B trees reduces the variance [35].

If we are using Random Forests for classification (like in this thesis), the simplest approach is the following:
Given test observation, we record the class predicted by each of the B trees, and take a majority vote: the final prediction is the most frequent class among the B predictions.

More formally, given a training set $X = x_1, \ldots, x_n$ with responses $Y = y_1, \ldots, y_n$, bagging B times selects a random sample with replacement of the training set and fits trees to these samples [100]:

For $b = 1, \ldots, B$:

1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification or regression tree f_b on X_b, Y_b.

After training, predictions for unseen samples x' can be made by averaging the predictions from all the individual regression trees on x':

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b(x') \tag{2.18}$$

or by using the majority vote in case of classification.

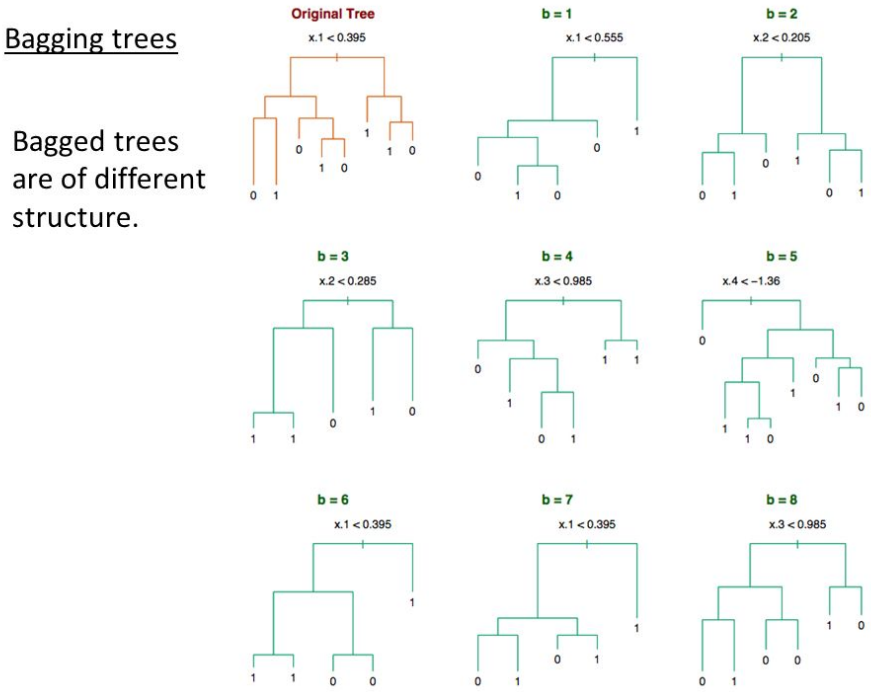


Figure 2.15. Bagging leads to different decision trees [3]

Additionally, an estimate of the uncertainty of the prediction can be made as the standard deviation of the predictions from all the individual regression trees on x':

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B} (f_b(x') - \hat{f})^2}{B - 1}} \tag{2.19}$$

The number of samples/trees, B, is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. An optimal number of trees B can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample x_i, using only the trees that did not have x_i in their bootstrap sample.

The training and test error tend to level off after some number of trees have been fit [35].

From Bagging to Random Forests

Random Forests provide an improvement over bagged decision trees by decorrelating the trees. As in bagging, the decision trees are built on a bootstrapped training set. The difference is that every time a split is considered, a random sample of m predictors is chosen as split candidates from the total p predictors. The split can only use one of those m predictors. $\sqrt{m}$ predictors are taken each split, typically with $m \equiv \sqrt{p}$ [35].

Practically, at each split in the tree, the algorithm can only consider a subset of the predictors (Fig. 2.16). This works because if there is a very strong predictor, that predictor would be used in the majority of the trees, and then all the bagged trees would look alike and would be highly correlated, thus leading to a small reduction in variance.

By forcing each split to consider only m predictors, on average $(p - m)/p$ of splits will not take into account a strong predictor. This, in turn, will decorrelate the trees.

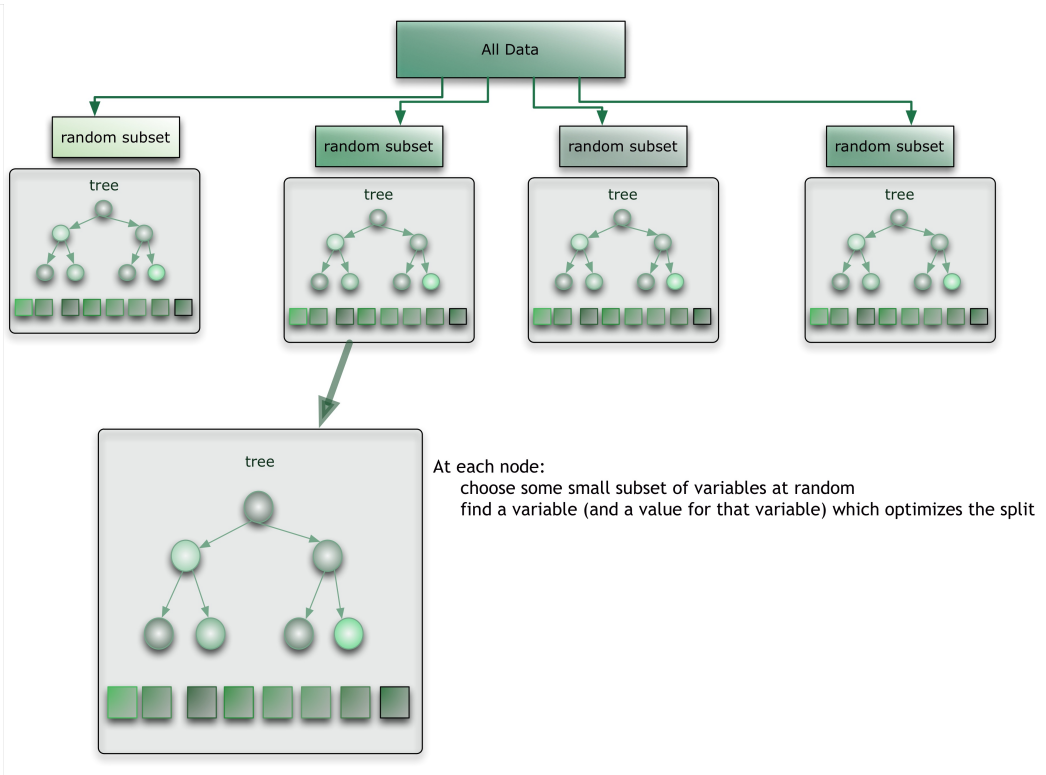


Figure 2.16. Random Forest overview

Out of bag Error

There is a very direct way to estimate the test error of a bagged model:
On average, we use 2/3 of the observations to fit a bagged tree. The remaining 1/3 that are not utilized are called the out-of-bag (OOB) observations.

We can predict the response for the i*th* observation using each of the trees in which that observation was OOB. This will generate approximately $B/3$ predictions for the i*th* observation. We can then average these predictions (or use majority voting for classification).

We can repeat the process for all the n observations and get an average classification error.
The resulting OOB error is a valid estimate of the test error for the bagged model, since the response for each observation is predicted using only the trees that were not fit using that observation [35].

Variable Importance

Bagging has the advantage of giving increased accuracy over using a single tree, at the cost of interpretability of the data. Using bagging we can no longer have easily interpreted diagrams.

With Random Forest we can instead rank the importance of each predictor:
given a dataset $D_n = \{X_i, Y_i\}_{i=1}^n$ we fit a RF to the data. While fitting the data, the out-of-bag error is averaged over the forest.

To get the importance of the j*th* feature, we permute it's values over the training data and recompute the OOB error.

The importance is given by doing the average of the difference between the OOB error before and after this permutation. Then the score is normalized by the standard deviation of the differences.

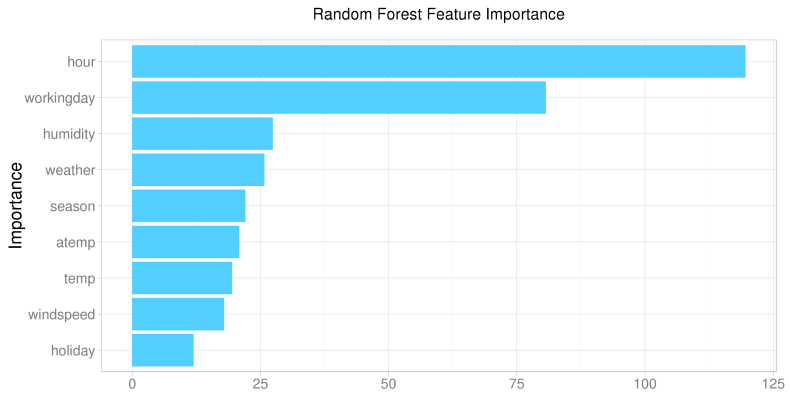


Figure 2.17. Example of variable importance with Random Forest [38].

2.4 SVM

Support Vector Machines (SVM) are a supervised machine learning algorithm used for both classification and regression.

The main idea is to find the optimal hyperplane for linearly separable data, and then extend this idea to data that are not linearly separable by mapping this data in a new space using a kernel function.

2.4.1 What is a hyperplane?

In p-dimension, a hyperplane is a flat affine subspace of $p-1$ dimension. For example, in two dimensions it's a line, in three dimensions it's a plane (Fig. 2.18), and so on.

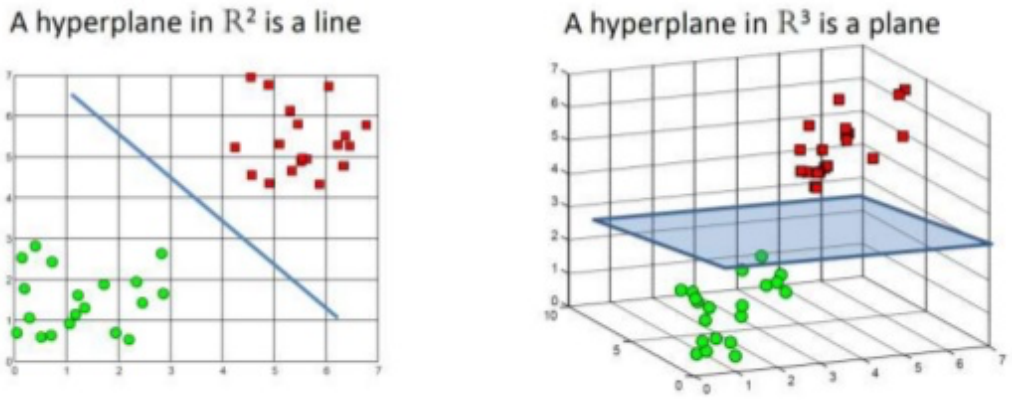


Figure 2.18. Two and three dimensional space hyperplanes [80]

The definition of an hyperplane for two dimensions is:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0 \tag{2.20}$$

Where any $X = (X_1, X_2)^T$ for which equation 2.20 holds is a point on the hyperplane.

And it can be extended to an hyperplane of p-dimensions:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0 \tag{2.21}$$

If X does not satisfy equation 2.21, but

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p > 0 \mid < 0 \tag{2.22}$$

we know that X lies to one side or the other of the hyperplane. So a hyperplane divides a space in two halves [35].

It's possible to build a hyperplane that can separate training data based on their class labels, for example by assigning $\{-1, 1\}$ if the result lies on one side or the other of the hyperplane.

2.4.2 Maximal Margin Classifier

If the data can be separated by a hyperplane, it means that there can be an infinite number of separating hyperplanes.
To build a classifier it's necessary to chose one between those infinite hyperplanes, such that the classification results are as accurate as possible.

The distance between the hyperplane and the nearest data point from either side is known as the margin.
Generally the choice is the *maximal margin hyperplane*, meaning the hyperplane that *maximizes* the margins (has the farthest minimum distance) for the data we are classifying.

In fact the distance of a data point from the hyperplane is a measure of our confidence that the observation was classified correctly [35].

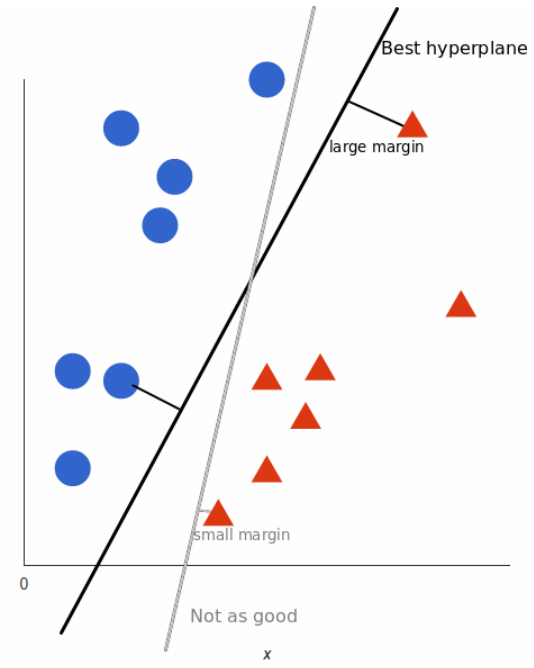


Figure 2.19. Different hyperplanes have different margins [83]

The idea is that a classifier with a large margin on the training set will have a large margin on the test set, so it will classify it correctly [35].

2.4.3 Support Vectors

Support vectors are the data points that lie closest to the hyperplane (Fig. 2.20), they are also the most difficult data points to classify and have direct influence on the location of the hyperplane. In fact, moving the support vectors would change the hyperplane's location.

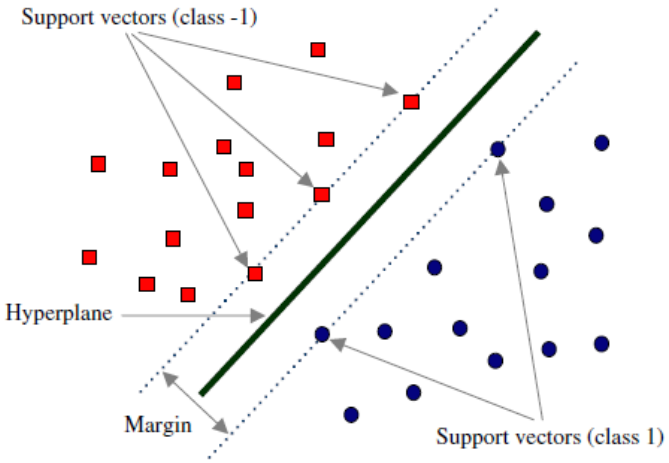


Figure 2.20. Example of Support Vectors

So the objective is to choose a hyperplane with the largest possible margin between it and the support vectors, since the larger is the margin, the lower the generalization error of the classifier [35].

2.4.4 Constructing the Maximal Margin Classifier

Finding the maximal margin hyperplane based on a set of n training observations $x_1, \dots x_n \in \mathbb{R}^p$ and with class labels $y_1 \dots y_n \in \{1, -1\}$, translates to an optimization problem:

Maximize the margin M:

$$\beta_0, \beta_1, \beta_2, \dots, \beta_p, M \tag{2.23}$$

subject to

$$\sum_{j=1}^{p} \beta_j^2 = 1 \tag{2.24}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) \geq M \quad \forall i = 1, \dots, n \tag{2.25}$$

Equations 2.24 and 2.25 ensure that each observation is on the correct side of the hyperplane, and at least at distance M from the hyperplane.
So M represents the margin of our hyperplane, and the optimization problem chooses $\beta_0, \beta_1, \beta_2, \beta_3, \dots, \beta_p$ to maximize M and find the maximal margin hyperplane [35].

2.4.5 Support Vector Classifier

Unfortunately this hyperplane does not necessarily exists (2.21), but we can extend this concept to find a hyperplane that almost separates the classes using a soft margin. This is really what an SVM does.

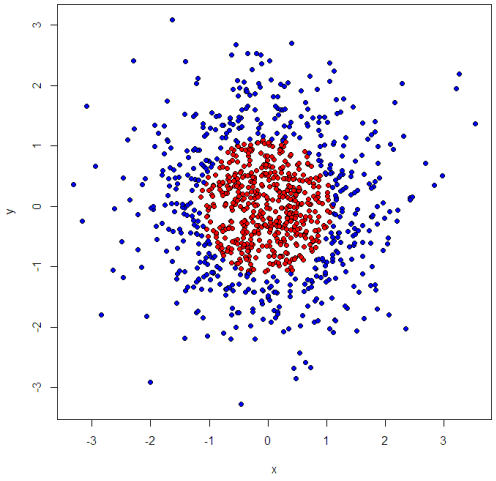


Figure 2.21. Example of Linearly non separable data [74]

Even if a separating hyperplane did exists, we might not want to use it: such a classifier would necessarily classify all of the training data perfectly, and that would lead to sensitivity to changes in data points (fig. 2.22). Also, being so sensitive could mean that it's overfitting the data.

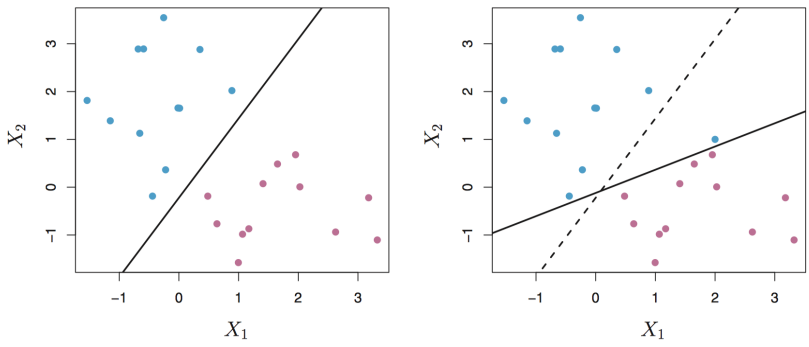


Figure 2.22. Adding a single data point (from left to right) can change the maximal margin hyperplane very significantly [35]

Instead we consider a classifier that does not separate the data perfectly, but is less prone to change for individual data points and can better classify *most* of the observations.

This classifier is called *soft margin classifier* or *support vector classifier* (Fig. 2.23). In a support vector classifier (SVC) a small subset of the observations are on the wrong side of the margin, or even on the wrong side of the separating hyperplane.

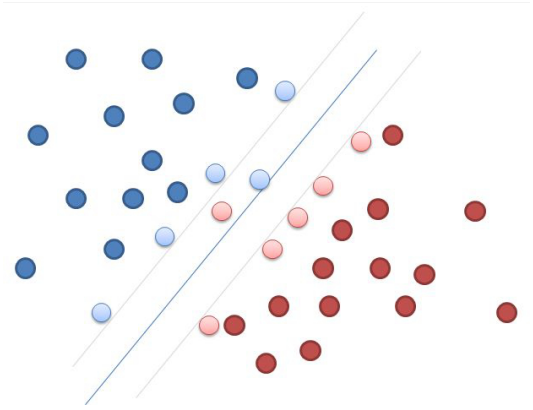


Figure 2.23. Soft margin classifier. Some data points are allowed to be misclassified to find a larger margin [76]

The optimization problem we are solving now is the following:

Maximize the margin M:

$$\beta_0, \beta_1, \beta_2, \ldots, \beta_p, \epsilon_1, \ldots, \epsilon_n, M \tag{2.26}$$

subject to

$$\sum_{j=1}^{p} \beta_j^2 = 1 \tag{2.27}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \quad \epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C \tag{2.28}$$

With $C \geq 0$, and $\epsilon_1, \ldots, \epsilon_n$ being slack variables to allow data points to be on the wrong side of the margin.

When we solve equations 2.26-2.28 we can classify x^* by determining where it lies on the hyperplane:

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \cdots + \beta_p x_p^* \tag{2.29}$$

Based on the value of ϵ_i we can make some assumptions:

- If $\epsilon_i = 0$ the ith observation is on the correct side of the margin.
- If $\epsilon_i > 0$ the ith observation is on the wrong side of the margin.
- If $\epsilon_i > 1$ the ith observation is on the wrong side of the hyperplane.

C determines the quantity and quality of the violations of the margin and hyperplane that is tolerable by our classifier, and is called *cost*.

- If $C = 0$ then we cannot violate the margin, and $\epsilon_1 = \cdots = \epsilon_n = 0$.
- If $C > 0$ then $\leq C$ observations ca be on the wrong side of the hyperplane.

When C is small the margin is small and we have a classifier that is very fit to the data, with low bias and high variance.
When C is big the margin is larger and the classifier is less fit to the data and has high bias and low variance.

The interesting property of this optimization problem is that only the support vectors (observations that are on the margin or that violate the margin) affect the hyperplane, and thus the classifier obtained.
All the observations that lie on the correct side of the margin do not affect the support vector classifier.

2.4.6 Kernels

Sometimes data is not linearly separable (Fig. 2.21). When that happens, Support Vector Machines work by enlarging the feature space using specific function called kernels.

The solution to equations 2.26-2.28 involves only the inner products of the observations, and not the observations themselves [35].
The inner product of two observations x_i , x_i' is given by:

$$\langle x_i, x_i' \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{2.30}$$

the linear SVC can be described as:

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle \tag{2.31}$$

To estimate β_0 and $\alpha_1 \ldots, \alpha_n$ all that is required are the $\binom{n}{2}$ inner products $\langle x_i, x_i' \rangle$ between all pairs of training observations [35].

Every time we compute the inner product (2.30) in equation 2.31 we could replace it with a generalization:

$$K(x_i, x_i') \tag{2.32}$$

Where K is a kernel function. A kernel function is used to quantify the similarity between two observations.
For example a linear kernel is:

$$K(x_i, x_i') = \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{2.33}$$

or for a polynomial kernel of degree d:

$$K(x_i, x_i') = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d \tag{2.34}$$

or a radial kernel:

$$K(x_i, x_i') = \exp(-\gamma \sum_{j=1}^{p} (x_{ij}, x_{i'j})^2), \quad \gamma \geq 0 \tag{2.35}$$

There are many more kernel functions that can be used based on the shape of data.

The advantage of using a kernel is first of all computational: by using a kernel we just need to compute $K(x_i, x_i')$ for all $\binom{n}{2}$ distinct pairs (i, i').

Graphically, starting with a two dimensional non linearly separable data, this is what happens:

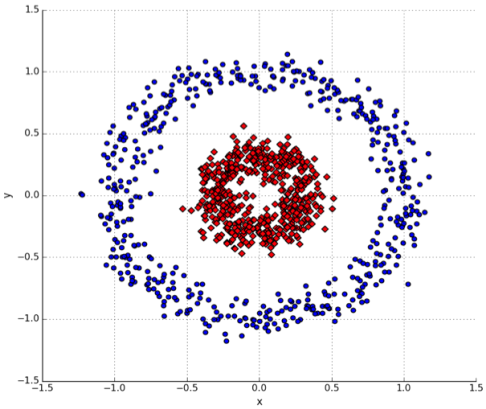


Figure 2.24. Linearly non separable data [88].

We map the data from input space X into a transformed feature space H, by using a non-linear function $\phi : X \to H$

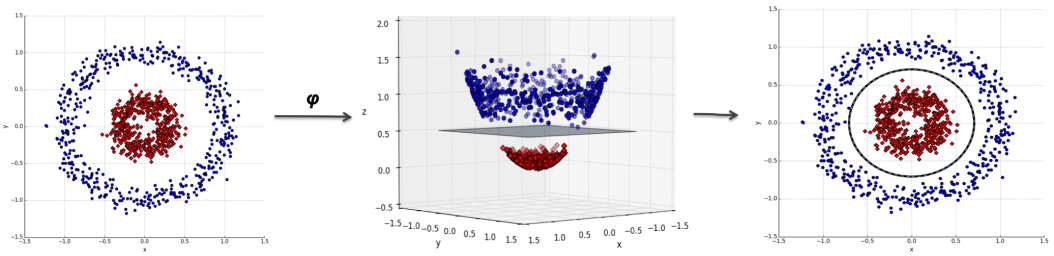


Figure 2.25. Transformation of the data in a feature space where the instances from the two classes may be linearly separable [88].

Chapter 3

Experiments

In this section I will explain what I've done and the stack used for this thesis: I will start by reviewing the OpenFace tool (3.1), the database (3.2) utilized and all the experiments and techniques used, such as.....

3.1 OpenFace

The OpenFace [89] toolkit is a tool for machine learning and computer vision researchers created ba Baltrusaitis et al. to perform facial landmark detection, head pose estimation, action unit recognition and eye gaze estimation.

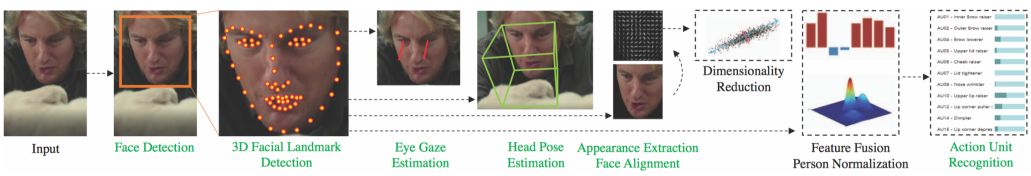


Figure 3.1. OpenFace Pipeline [89]

3.1.1 Landmark Detection

Figure 3.2. Example of different facial landmarks detected with OpenFace in different conditions and viewing angles. [89].

The first step in Action Unit identification is to detect the facial landmarks. To accomplish this a local detector called Convolutional Experts Network (CEN) 3.3 is utilized.
CEN has the advantage of aggregating a neural architecture and patch experts (local detectors, they evaluate the probability of a landmark being aligned at a particular pixel location). CEN can learn different patch experts and adapt to diverse appearance models without explicit attribute labeling.

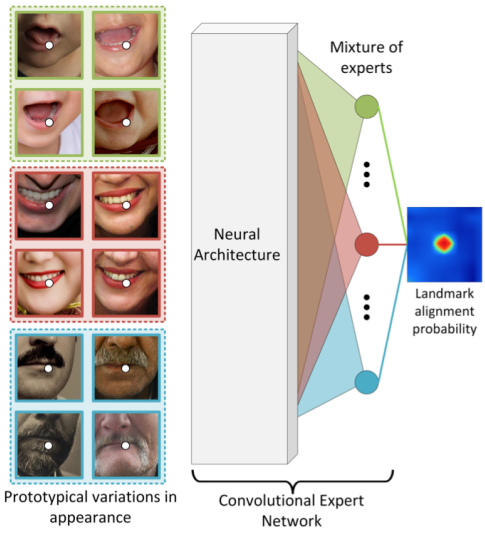


Figure 3.3. Facial landmarks naturally cluster around appearance prototypes (facial hair, expressions, make-up etc). To model such appearance variations a Convolutional Experts Network (CEN) is used to bring together the advantages of neural architectures and mixtures of patch experts to model landmark alignment probability [104].

OpenFace uses a Convolutional Experts Constrained Local Model (CE-CLM) [104], which is a Constrained Local Model (CLM) that uses CEN as a local detector.

A Constrained Local Model is class of methods for locating sets of points (constrained by a statistical shape model) on a target image [**clm_cootes**]. Generally the procedure is as follows:

- Sample a region from the image around the current estimate, projecting it into a reference frame.
- For each point, generate a "response image" giving a cost for having the point at each pixel.
- Search for a combination of points which optimizes the total cost, by manipulating the shape model parameters.

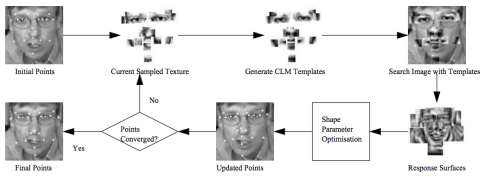


Figure 3.4. Overview of CLM [**clm_cootes**].

CLM are used to model the appearance of each facial landmark individually by using local detectors and a shape model for constrained optimization.

The CE-CLM is divided in two fundamental parts: response map computation using CEN, and shape parameter update.
In the first step, the landmarks alignment is computed separately from the other landmarks.
In the second phase, all landmarks are considered together and for misaligned landmarks and irregular shapes their position is penalized, using a Point Distribution Model (PDM).

CEN

In the first step, the objective is to generate a response map to localize the individual landmarks. This is achieved by assessing the landmark alignment probability at specific pixel locations.
CEN takes in input a Region of Interest (ROI) around the currently estimated position of a landmark, and outputs a response map that calculates the landmark alignment probability at each pixel location (Fig. 3.5).

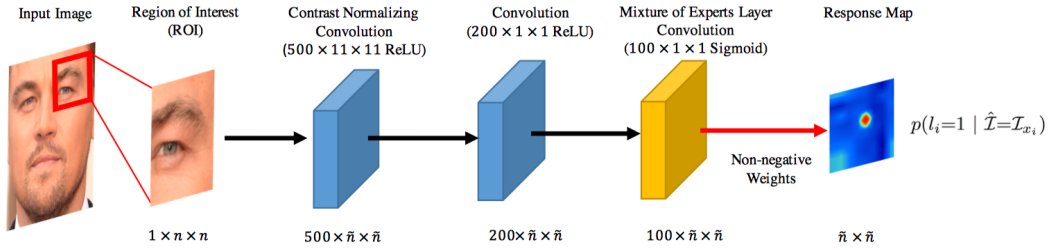


Figure 3.5. Overview of the Convolutional Experts Network model. The output response map is a non-negative and non-linear combination of neurons in ME-layer using a sigmoid activation [104].

In order to do all this, the ROI is initially passed in a Contrast Normalizing Convolution layer to perform z-score normalization and to calculate the correlation between input and kernel. The output is then convolved in another layer of ReLU neurons (convolution is an operation on two functions to produce a third function that expresses how the shape of one is modified by the other).

The last neural layer before the response map is the Mixture of Expert Layer (ME-layer), and it can model the alignment probability using a combination of patch experts (local detectors) that are able to represent different landmarks appearance prototypes by outputting individual votes on alignment through a sigmoid function. The response maps from all the local detectors are then combined in the last layer, giving the final alignment probability.

Point Distribution Model

The Point Distribution Model is a used to represent the mean geometry of a shape and some statistical modes of geometric variation inferred from a training set of shapes [99].

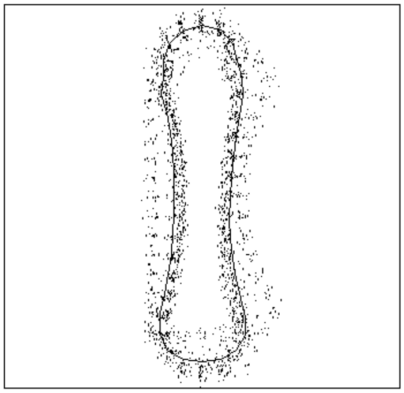


Figure 3.6. PDM of a metacarpal. Dots mark the possible position of landmarks, and the line denote the mean shape [44].

Point distribution models rely on landmark points. The general PDM works this

way:

1. A set of training images are manually landmarked to sufficiently approximate the geometry of the original shapes.
2. This k landmarks are aligned in two dimensions resulting in $\mathbf{X} = (x_1, y_1, \ldots, x_k, y_k)$
3. The shape outlines are reduced to sequences of k landmarks, so that any given training shape can be defined by the vector $\mathbf{X} \in \mathbb{R}^{2k}$.
4. The matrix of the top d eigenvectors is given as $\mathbf{P} \in \mathbb{R}^{2k \times d}$, and each eigenvector describes a principal mode of variation along the set.
5. A linear combination of the eigenvectors is used to define a new shape $\mathbf{X}'$, mathematically defined as:
$$\mathbf{X}' = \overline{\mathbf{X}} + \mathbf{Pb} \tag{3.1}$$
where $\overline{\mathbf{X}}$ is defined as the mean shape across all training images, and $\mathbf{b}$ is a vector of scaling values for each principal component.
6. By modifying the variable $\mathbf{b}$ an infinite number of shapes can be defined. $\mathbf{b}$ shouldn't generally be modified more than $\pm 3\sigma$ [99].

In the OpenFace framework Point Distribution Model [78] models the location of facial feature points in the image using non-rigid shape and rigid global transformation parameters, and is utilized in the second phase of the algorithm.

The application of PDM has two objectives:

- They are employed to control the landmark locations.
- They are used to regularize the shapes in the CE-CLM framework by using Non-Uniform Regularized Landmark Mean Shift (NU-RLMS) [14].

This has an effect in the final detected landmarks, where the irregular shapes are imposed a penalty.

3.1.2 Action Unit Detection

Action Unit (AU) detection plays a fundamental role in our work. We now describe the process used to detect the AUs, starting with an overview of the utilized databases for AU detection.

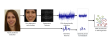


Figure 3.7. AU detection and intensity pipeline [13].

Datasets

There are three main dataset used for training the Action Unit detection system: DISFA [57], BP4D-Spontaneous [105] and SEMAINE [59]. These three datasets

consist of video of people subject to emotion inducing tasks.

The BP4D database of spontaneous facial expressions includes videos of 41 participants (23 women and 18 men, 21 for training and 20 for validating). The age ranged from 18 to 29; 11 were Asian, 6 African-American, 4 Hispanic, and 20 Euro-American.
Emotion inducing techniques were used to elicit an emotional response. Frame-level ground-truth for facial actions was obtained by using the Facial Action Coding System annotations, performed by trained professionals.
Each participant in the database is associated with 8 tasks. For each task, there are both 3D and 2D videos and the metadata include annotations for 11 AUs for occurrence and 5 AUs for intensities.

The FACS annotated SEMAINE subset contains recordings of 31 subjects (15 for training and 16 for validation). It consists of one minute long recordings, leading to 93k frames labeled for 5 AU occurrences.

DISFA (Denver Intensity of Spontaneuos Facial Action) Database is a non posed facial expression database for automatic action unit detection. It contains videos of 27 participants (12 female, 15 males; 14 used for training and 13 for validation). It includes 4 minute-long videos of spontaneous facial expression, resulting in 130k frames annotated for 12 AUs (Fig. 3.8), comprehensive of AUs intensity on a 0 to 5 scale.

AU#	FACS Name
1	Inner Brow Raiser
2	Outer Brow Raiser
4	Brow Lowerer
5	Upper Lid Raiser
6	Cheek Raiser
9	Nose Wrinkler
12	Lip Corner Puller
15	Lip Corner Depressor
17	Chin Raiser & Mentalis
20	Lip Stretcher
25	Lip Part
26	Jaw Drop

Figure 3.8. AUs coded in the DISFA database [27].

All of the datasets have three AUs in common (2, 12, and 17).
SEMAINE and DISFA share AUs 2, 12, 17, 25.
BP4D and DISFA share AUs 1, 2, 4, 6, 12, 15, 17. This allows for cross-dataset training.

Other dataset included in the study are:

- CK+ [54]:
- FERA 2011 [91]:
- FERA 2015 [90]:
- AVEC 2011 [79]:

Feature Extraction

There are two types of features that are used: appearance and geometry ones. To extract those features it's required to track certain landmarks on the face, and then continue this process by performing face alignment.

Face Tracking Face tacking is done by utilizing Constrained Local Neural Field (CLNF) facial landmark detector and tracker, backed up by a structural SVM for facial detection [14]. CLNF is a specific case of Constrained Local Model (CLM), that differs from the original by utilizing more advanced local detectors and a different optimization function.

The Constrained Local Model can be described by the following parameters:
$p = [s, \mathbf{R}, \mathbf{p}, \mathbf{t}]$.

- s is the scale factor.
- $\mathbf{R}$ is the object rotation.
- $\mathbf{t}$ is the 2D translation.
- $\mathbf{p}$ describes the shape of a vector of non rigid variations.

These parameters can be modified to compute different versions of the model. The resulting point distribution model is:

$$x_i = s \cdot \mathbf{R}(\overline{x_i} + \phi_i \mathbf{p}) + \mathbf{t} \tag{3.2}$$

Where x_i is the location of the ith feature point in an image, $\overline{x_i}$ is the mean of the ith element of the PDM, and ϕ_i is the ith eigenvector that describes the variation of the feature point.

In the Constrained Local Model we use the parameters from the face detection to estimate the maximum a posteriori probability p of the face model.

Alignment and Masking For the extracted face to be correctly analyzed, there needs to be a mapping to a common reference frame, and the changes resulting from scaling and in plane rotation need to be removed.

In order to achieve this, a similarity transform from the currently detected landmarks to a representation of frontal landmarks from a neutral expression (projection of mean shape from a 3D PDM) is utilized. The similarity transform is done with Procrustes superimposition that minimizes the mean square error between aligned pixels [14].

The result is a 112×112 pixel image of the face with 45 pixel interpupillary distance (Fig 3.9).

To reduce the weight of significant facial expressions (mouth opening, brow raises etc.) on the similarity transform, only the most stable facial landmarks must be used.
In order to determine those points, the most stable CLNF detected landmarks on the CK+ dataset [54] are examined. CK+ is a dataset containing videos of people performing diverse facial expressions, mostly starting from a neutral pose, while the head is still.

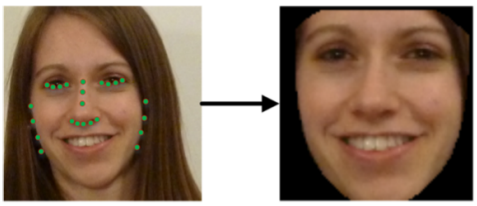


Figure 3.9. Stable points used for alignment to a common reference frame, followed by masking. [13].

Masking, performed using a convex hull surrounding the feature points, is done to remove non face relevant information from the images.

Appearance Features After the face is aligned to the 112×112 image it's time to extract the appearance features. To get the appearance features, 12×12 block of 31 dimensional Histogram of Oriented Gradients (HOG) are extracted, giving a 4464 dimensional vector characterizing the face. The implementation of HoG utilized comes from dlib [47].

Once the feature vector is obtained, the next step is to reduce it's dimensionality. In order to do that Principal Component Analysis (PCA) is applied. To generalize the dimensionality reduction, the training was performed on the FERA 2015 [90], CK+ [54], DISFA [57], AVEC 2011 [79] and FERA 2011 [91] datasets. By using PCA while sub-sampling from peak and neutral expressions and keeping 95% of explained variability, the feature vector becomes of 1379 dimensions.

Geometry Features The geometry features are obtained trough the CE-CLM models, and consist of non-rigid shape parameters and landmarks locations. They resulted in a 227 dimensional vector representing geometry features.
Summed to the appearance features, this leads to a 1606 dimensional vector that defines the appearance of the face.

Neutral Expression Extraction To extract some of the facial expression is very important to have a neutral starting expression. There are personal differences on how we appear while in a neutral resting state, such as people looking more smily or

frowny then others [62]. To address this issue there needs to be a person specific calibration, done by adjusting for neutral expressions.
This is done by computing the median value of the face descriptors in a video, leading to a neutral expression descriptor. This works assuming that in a video most of the frames will contain a neutral expression, this should hold true especially for real life situations where most of the time the interactions are performed with a neutral expression [9].
The median face descriptor is subtracted from the feature descriptor, giving normalized features. To help with the ease of computing the median, a histogram is kept for each element in the feature vector.

Classification and Regression Action Units detection is performed through Support Vector Machines (SVM), and Action Unit intensity using Support Vector Regression (SVR). In both cases a linear kernel is utilized, as more complex kernels had no effect on performances and were quite slower.
Since the AU occurrences are not balanced by nature, it's highly important to balance the training data. This was done by under-sampling the negative AU samples from training data, leading to an equal number of positive and negative samples.

3.2 Real Life Trial DataBase

This section is about the database we used to perform our experiments: it comes from the work done for the paper "Deception Detection using Real-life Trial Data" [67].

Figure 3.10. Examples of images from the dataset videos. [67].

The dataset is gathered from real-life trial videos available on YouTube and other public websites. The dataset also contains statements made by exonerees after exoneration, and a few statements from defendants during crime-related TV episodes.

The first step to collecting the dataset was to identify public multimedia sources where the recordings of the trials were available, and deceptive and truthful behavior could be observed and verified.
The videos are of trial recordings where the defendant or witness in the video could be clearly identified, the face is visible enough during most of the clip duration, and the visual quality should be good enough to accurately see the facial expressions

(Fig. 3.10).
There are three outcomes for the trials that were considered to label the videos as deceptive or truthful: guilty, non-guilty, and exoneration.
For the guilty verdicts the deceptive clips are taken from the defendant in the trial, while the truthful clips are gathered from the witnesses. There are also instances where the deceptive videos are of suspects denying a committed crime, and truthful ones are from the same person answering questions that where verified by the police as truthful.

In regards to the witnesses, when the testimony is verified by a police officer they are labeled as true. Testimonies that help the guilty party are labeled as false. Exoneration (reversal of the sentence) testimonies are regarded as truthful.

The dataset consists of 121 videos, 61 of which are deceptive and 60 truthful. The average length of the videos is 28.0 seconds. The average video length for deceptive videos is 27.7 seconds, while the one for truthful videos is 28.3 seconds. The data consists of 56 total subject, 21 unique females and 35 unique males, with ages between 16 and 60 years.

3.3 GLM

3.4 LDA

3.5 QDA

3.6 SVM

3.7 Correlations

?

Chapter 4

Results

Results obtained

Chapter 5

Conclusions

Conclusion

Bibliography

[1] URL: https://www.learning-mind.com/eye-movements-when-lying/ (p. 6).

[2] URL: https://www.slideshare.net/mlvlc/l4-ensembles-of-decision-trrees (p. 34).

[3] URL: https://slideplayer.com/slide/10391576/ (p. 35).

[4] Sean A. Spence. "Playing Devil's advocate: The case against fMRI lie detection". In: 13 (Feb. 2008), pp. 11–25 (p. 9).

[5] M. Abouelenien et al. "Detecting Deceptive Behavior via Integration of Discriminative Features From Multiple Modalities". In: *IEEE Transactions on Information Forensics and Security* 12.5 (May 2017) (p. 11).

[6] Mohamed Abouelenien, Rada Mihalcea, and Mihai Burzo. "Analyzing Thermal and Visual Clues of Deception for a Non-Contact Deception Detection Approach". In: *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*. 2016. URL: http://doi.acm.org/10.1145/2910674.2910682 (p. 11).

[7] Mohamed Abouelenien, Rada Mihalcea, and Mihai Burzo. "Trimodal Analysis of Deceptive Behavior". In: *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*. 2015. URL: http://doi.acm.org/10.1145/2823465.2823470 (p. 10).

[8] Mohamed Abouelenien et al. "Deception Detection Using a Multimodal Approach". In: *Proceedings of the 16th International Conference on Multimodal Interaction*. 2014. URL: http://doi.acm.org/10.1145/2663204.2663229 (p. 11).

[9] S. Afzal and P. Robinson. "Natural affect data — Collection amp; annotation in a learning context". In: *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*. Sept. 2009. DOI: 10.1109/ACII.2009.5349537 (p. 53).

[10] Baker Alysha, ten Brinke Leanne, and Porter Stephen. "Will get fooled again: Emotionally intelligent people are easily duped by high-stakes deceivers". In: *Legal and Criminological Psychology* 18.2 (), pp. 300–313. DOI: 10.1111/j.2044-8333.2012.02054.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.2044-8333.2012.02054.x (p. 3).

[11] Tarek Amr. *Predict the Future with Regression Analysis*. URL: https://medium.com/@gr33ndata/learn-regressions-analysis-23b789bf2c36 (p. 25).

[12] A. Arasteh, M. H. Moradi, and A. Janghorbani. "A Novel Method Based on Empirical Mode Decomposition for P300-Based Detection of Deception". In: *IEEE Transactions on Information Forensics and Security* 11.11 (Nov. 2016) (p. 8).

[13] T. Baltrušaitis, M. Mahmoud, and P. Robinson. "Cross-dataset learning and person-specific normalisation for automatic Action Unit detection". In: *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*. Vol. 06. May 2015, pp. 1–6 (pp. 14, 15, 49, 52).

[14] T. Baltrusaitis, P. Robinson, and L. Morency. "Constrained Local Neural Fields for Robust Facial Landmark Detection in the Wild". In: *2013 IEEE International Conference on Computer Vision Workshops*. Dec. 2013 (pp. 49, 51).

[15] T. Baltrušaitis, P. Robinson, and L. P. Morency. "OpenFace: An open source facial behavior analysis toolkit". In: *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Mar. 2016, pp. 1–10 (p. 15).

[16] Anup Bhande. *What is underfitting and overfitting in machine learning and how to deal with it*. URL: https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76 (p. 22).

[17] N. Bhaskaran et al. "Lie to Me: Deceit detection via online behavioral learning". In: *Face and Gesture 2011*. Mar. 2011 (p. 7).

[18] *Bias-Variance Tradeoff*. URL: https://elitedatascience.com/bias-variance-tradeoff (p. 22).

[19] Marilyn G. Boltz, Rebecca L. Dyer, and Anna R. Miller. "Jo Are You Lying to Me? Temporal Cues for Deception". In: *Journal of Language and Social Psychology* 29.4 (2010), pp. 458–466. DOI: 10.1177/0261927X10385976. URL: https://doi.org/10.1177/0261927X10385976 (p. 5).

[20] D. R. Burnham K. P.; Anderson. *Model Selection and Multimodel Inference (2nd ed.), Springer-Verlag*. 2002 (p. 21).

[21] Jr. Charles F. Bond and Bella M. DePaulo. "Accuracy of Deception Judgments". In: *Personality and Social Psychology Review* 10.3 (2006), pp. 214–234. DOI: 10.1207/s15327957pspr1003_2. URL: https://doi.org/10.1207/s15327957pspr1003_2 (p. 3).

[22] W. S. Chu, F. D. L. Torre, and J. F. Cohn. "Selective Transfer Machine for Personalized Facial Action Unit Detection". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. June 2013 (p. 15).

[23] W. S. Chu, F. D. l. Torre, and J. F. Cohn. "Selective Transfer Machine for Personalized Facial Expression Analysis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.3 (Mar. 2017). DOI: 10.1109/TPAMI.2016.2547397 (p. 15).

[24] W. S. Chu, F. De la Torre, and J. F. Cohn. "Learning Spatial and Temporal Cues for Multi-Label Facial Action Unit Detection". In: *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*. May 2017 (p. 15).

[25] Wikipedia contributors. *Machine learning — Wikipedia, The Free Encyclopedia*. [Online; accessed 16-July-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=849817385 (p. 19).

[26] Langleben Daniel D. "Detection of deception with fMRI: Are we there yet?" In: *Legal and Criminological Psychology* 13.1 (), pp. 1–9. DOI: 10.1348/135532507X251641. URL: https://onlinelibrary.wiley.com/doi/abs/10.1348/135532507X251641 (p. 9).

[27] *Denver Intensity of Spontaneous Facial Action Database*. URL: http://www.engr.du.edu/mmahoor/DISFAContent.htm (p. 50).

[28] Shichuan Du, Yong Tao, and Aleix M. Martinez. "Compound facial expressions of emotion". In: *Proceedings of the National Academy of Sciences of the United States of America* (2014) (p. 14).

[29] P. Ekman and W. Friesen. "Facial Action Coding System: A Technique for the Measurement of Facial Movement". In: *Consulting Psychologists Press* (1978) (p. 14).

[30] P. Ekman and W. V. Friesen. "Nonverbal leakage and clues to deception". In: *Psychiatry* 32.1 (1969), pp. 88–106 (p. 12).

[31] Paul Ekman. *Emotions Revealed: Recognizing Faces and Feelings to Improve Communication and Emotional Life*. 2007 (p. 12).

[32] Skrondal A. Everitt B.S. "Cambridge Dictionary of Statistics". In: *Cambridge University Press* (2010) (p. 21).

[33] Martha J. Farah et al. "Functional MRI-based lie detection: scientific and societal challenges". In: *Nature Reviews Neuroscience* 15 (Jan. 2014). URL: http://dx.doi.org/10.1038/nrn3665 (p. 9).

[34] Kyosuke Fukuda. "Eye blinks: new indices for the detection of deception". In: *International Journal of Psychophysiology* 40.3 (2001), pp. 239–245 (p. 6).

[35] Trevor Hastie Gareth James Daniela Witten and Robert Tibshiran. *An Introduction to Statistical Learning*. URL: http://www-bcf.usc.edu/~gareth/ISL/index.html (pp. 26, 27, 34, 36–41, 43).

[36] S. George et al. "Eye blink count and eye blink duration analysis for deception detection". In: *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. Sept. 2017 (p. 7).

[37] E. Haggard and K. Isaacs. "Micromomentary facial expressions as indicators of ego mechanisms in psychotherapy". In: *Methods of research in psychotherapy. New York: Appleton-Century-Crofts* (1966), pp. 154–165 (p. 12).

[38] Ben Hammer. *Introducing Kaggle Scripts*. URL: https://www.kaggle.com/general/13285 (p. 37).

[39] Jeffrey Hancock. "Digital deception: When, where and how people lie online". In: (Jan. 2012), pp. 287–301 (p. 3).

[40] L. Hao et al. "Facial Action Unit Recognition Augmented by Their Dependencies". In: *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*. May 2018 (p. 15).

[41] Maria Hartwig et al. "Detecting deception in suspects: verbal cues as a function of interview strategy". In: *Psychology, Crime & Law* 17.7 (2011), pp. 643–656. DOI: 10.1080/10683160903446982. URL: https://doi.org/10.1080/10683160903446982 (p. 3).

[42] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning (2nd ed.) Springer*. 2008. URL: http://www-stat.stanford.edu/~tibs/ElemStatLearn/ (pp. 32, 33).

[43] Carl-Herman Hjortsjö. *Man's face and mimic language*. 1969 (p. 14).

[44] *Image understanding: Point distribution models*. URL: http://user.engineering.uiowa.edu/~dip/lecture/Understanding3.html (p. 48).

[45] *k-means data clustering*. URL: https://towardsdatascience.com/k-means-data-clustering-bce3335d2203 (p. 23).

[46] Michal Kawulok et al. "In Search of Truth: Analysis of Smile Intensity Dynamics to Detect Deception". In: *Advances in Artificial Intelligence - IBERAMIA 2016*. 2016 (p. 13).

[47] Davis E. King. "Dlib-ml: A Machine Learning Toolkit". In: *J. Mach. Learn. Res.* 10 (Dec. 2009). ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1577069.1755843 (p. 52).

[48] William Koehrsen. *Random Forest Simple Explanation*. URL: https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d (p. 32).

[49] Ying-Fang Lai, Mu-Yen Chen, and Hsiu-Sen Chiang. "Constructing the lie detection system with fuzzy reasoning approach". In: *Granular Computing* (Nov. 2017). URL: https://doi.org/10.1007/s41066-017-0064-3 (p. 8).

[50] Sharon Leal and Aldert Vrij. "Blinking During and After Lying". In: *Journal of Nonverbal Behavior* 32.4 (Dec. 2008), pp. 187–194. URL: https://doi.org/10.1007/s10919-008-0051-0 (p. 7).

[51] Xiaobai Li et al. "A Spontaneous Micro Facial Expression Database: Inducement, Collection and Baseline". In: *Face and Gesture (FG)*. 2013 (p. 13).

[52] Xiaobai Li et al. "Reading Hidden Emotions: Spontaneous Micro-expression Spotting and Recognition". In: *IEEE Trans. Affective Computing (TAFFC)*. 2015 (p. 13).

[53] Kai Keat Lim et al. "Lying Through the Eyes: Detecting Lies Through Eye Movements". In: *Proceedings of the 6th Workshop on Eye Gaze in Intelligent Human Machine Interaction: Gaze in Multimodal Interaction*. 2013. URL: http://doi.acm.org/10.1145/2535948.2535954 (p. 7).

[54] P. Lucey et al. “The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*. June 2010. DOI: 10.1109/CVPRW.2010.5543262 (pp. 15, 51, 52).

[55] Bella M. DePaulo et al. “Lying in Everyday Life”. In: 70 (June 1996), pp. 979–95 (p. 3).

[56] *Machine Learning 101 what is machine learning*. URL: https://hydrasky.com/network-security/machine-learning-101-what-is-machine-learning/ (p. 21).

[57] S. M. Mavadati et al. “DISFA: A Spontaneous Facial Action Intensity Database”. In: *IEEE Transactions on Affective Computing* 4 (Apr. 2013). DOI: 10.1109/T-AFFC.2013.4. URL: doi.ieeecomputersociety.org/10.1109/T-AFFC.2013.4 (pp. 49, 52).

[58] Carol McDonald. *Demystifying AI, Machine Learning and Deep Learning*. URL: https://mapr.com/blog/demystifying-ai-ml-dl/ (p. 20).

[59] G. McKeown et al. “The SEMAINE corpus of emotionally coloured character interactions”. In: *2010 IEEE International Conference on Multimedia and Expo*. July 2010. DOI: 10.1109/ICME.2010.5583006 (pp. 15, 49).

[60] Rada Mihalcea, Verónica Pérez-Rosas, and Mihai Burzo. “Automatic Detection of Deceit in Verbal Communication”. In: *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*. 2013. URL: http://doi.acm.org/10.1145/2522848.2522888 (pp. 6, 12).

[61] H. Nasri, W. Ouarda, and A. M. Alimi. “ReLiDSS: Novel lie detection system from speech signal”. In: *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*. Nov. 2016 (p. 6).

[62] Donald Neth and Aleix M. Martinez. “Emotion perception in emotionless face images suggests a norm-based representation”. In: *Journal of Vision* 9.1 (2009). DOI: 10.1167/9.1.5. URL: http://dx.doi.org/10.1167/9.1.5 (p. 53).

[63] Annalyn Ng. *Decision Trees: A Disastrous Tutorial*. URL: https://www.kdnuggets.com/2016/09/decision-trees-disastrous-overview.html (p. 33).

[64] D. I. Noje and R. Malutan. “Head movement analysis in lie detection”. In: *2015 Conference Grid, Cloud High Performance Computing in Science (ROLCG)*. Oct. 2015 (p. 11).

[65] Paola Noreña Cardona. “A COMPENDIUM OF PATTERN RECOGNITION TECHNIQUES IN FACE, SPEECH AND LIE DETECTION”. In: 24 (Nov. 2015) (p. 5).

[66] M. Owayjan et al. “The design and development of a Lie Detection System using facial micro-expressions”. In: *2012 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*. Dec. 2012 (p. 13).

[67] Verónica Pérez-Rosas et al. "Deception Detection Using Real-life Trial Data". In: *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. ICMI '15. 2015. URL: http://doi.acm.org/10.1145/2818346.2820758 (pp. 2, 6, 12, 53).

[68] Tomas Pfister et al. "Recognising Spontaneous Facial Micro-expressions". In: *International Conference on Computer Vision (ICCV)*. 2011 (p. 12).

[69] Stephen M Porter, Leanne ten Brinke, and Brendan B Wallace. "Secrets and Lies: Involuntary Leakage in Deceptive Facial Expressions as a Function of Emotional Intensity". In: 2012 (p. 3).

[70] Stephen Porter and Leanne ten Brinke. "The Truth About Lies: What Works in Detecting High-Stakes Deception?" In: 15 (Feb. 2010), pp. 57–75 (p. 5).

[71] Stephen Porter and Mary Campbell. "A. Vrij, Detecting Lies and Deceit: The Psychology of Lying and Implications forProfessional Practice". In: 7 (Sept. 1999), pp. 227–232 (p. 3).

[72] Probaway. *What action does a baby's emotion generate in its mother?* URL: https://probaway.wordpress.com/tag/dna-of-human-emotional-expression/ (p. 12).

[73] J. G. Proudfoot et al. "Deception is in the eye of the communicator: Investigating pupil diameter variations in automated deception detection interviews". In: *2015 IEEE International Conference on Intelligence and Security Informatics (ISI)*. May 2015 (p. 7).

[74] Quazi Marufur Rahman. *What is a Support Vector Machine?* URL: https://www.quora.com/What-is-a-Support-Vector-Machine (p. 41).

[75] B. Rajoub and R. Zwiggelaar. "Thermal facial analysis for deception detection". In: *IEEE Transactions on Information Forensics and Security*. 2014 (p. 10).

[76] Jan Rupnik. *Stochastic Subgradient Approach for Solving Linear Support Vector Machines*. URL: https://slideplayer.com/slide/8088877/ (p. 42).

[77] E. Bienenstock S. Geman and R. Doursat. "Neural networks and the bias/variance dilemma". In: *Neural Computation 4* (1992) (p. 22).

[78] Jason M. Saragih, Simon Lucey, and Jeffrey F. Cohn. "Deformable Model Fitting by Regularized Landmark Mean-Shift". In: *International Journal of Computer Vision* 91.2 (Jan. 2011). ISSN: 1573-1405. DOI: 10.1007/s11263-010-0380-4. URL: https://doi.org/10.1007/s11263-010-0380-4 (p. 49).

[79] Björn Schuller et al. "AVEC 2011-the First International Audio/Visual Emotion Challenge". In: *Proceedings of the 4th International Conference on Affective Computing and Intelligent Interaction - Volume Part II*. Berlin, Heidelberg: Springer-Verlag, 2011. ISBN: 978-3-642-24570-1. URL: http://dl.acm.org/citation.cfm?id=2062850.2062907 (pp. 51, 52).

[80] Ankit Sharma. *Support Vector Machines without tears*. URL: https://www.slideshare.net/ankitksharma/svm-37753690 (p. 38).

[81] A. I. Simbolon et al. “An experiment of lie detection based EEG-P300 classified by SVM algorithm”. In: *2015 International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT)*. Oct. 2015 (p. 8).

[82] B. Singh, P. Rajiv, and M. Chandra. “Lie detection using image processing”. In: *2015 International Conference on Advanced Computing and Communication Systems*. Jan. 2015 (p. 7).

[83] Bruno Stecanella. *An introduction to Support Vector Machines (SVM)*. URL: https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/ (p. 39).

[84] R.G.D Steel and J. H Torrie. *Principles and Procedures of Statistics with Special Reference to the Biological Science, McGraw Hill*. 1960 (p. 27).

[85] Peter Norvig Stuart J. Russell. *Artificial Intelligence: A Modern Approach, Third Edition - Prentice Hall*. 2010 (p. 20).

[86] Lin Su and Martin Levine. “Does “lie to me” lie to you? An evaluation of facial clues to high-stakes deception”. In: *Computer Vision and Image Understanding* 147 (2016). URL: http://www.sciencedirect.com/science/article/pii/S1077314216000345 (pp. 4, 13).

[87] S. Sumriddetchkajorn et al. “Simultaneous Analysis of Far Infrared Signals From Periorbital and Nostril Areas for Nonintrusive Lie Detection: Performance From Real Case Study”. In: *Journal of Lightwave Technology* 33.16 (Aug. 2015) (p. 10).

[88] *Support Vector Machines*. URL: http://beta.cambridgespark.com/courses/jpm/05-module.html (p. 44).

[89] T. usaitis et al. “OpenFace 2.0: Facial Behavior Analysis Toolkit”. In: *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*. May 2018 (pp. 2, 45, 46).

[90] M. F. Valstar et al. “FERA 2015 - second Facial Expression Recognition and Analysis challenge”. In: *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*. May 2015 (pp. 51, 52).

[91] M. F. Valstar et al. “The first facial expression recognition and analysis challenge”. In: *Face and Gesture 2011*. Mar. 2011. DOI: 10.1109/FG.2011.5771374 (pp. 51, 52).

[92] Bruno Verschuere et al. “The ease of lying”. In: 20 (Nov. 2010), pp. 908–11 (p. 5).

[93] Aldert Vrij et al. “Police officers ability to detect deception in high stakes situations and in repeated lie detection test”. In: 20 (Sept. 2006), pp. 741–755 (p. 3).

[94] Wikipedia contributors. *Decision tree learning — Wikipedia, The Free Encyclopedia*. [Online; accessed 4-September-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Decision_tree_learning&oldid=856803699 (p. 33).

[95] Wikipedia contributors. *Functional magnetic resonance imaging — Wikipedia, The Free Encyclopedia*. [Online; accessed 20-June-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Functional_magnetic_resonance_imaging&oldid=842791829 (p. 9).

[96] Wikipedia contributors. *Linear regression — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-August-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Linear_regression&oldid=851432903 (p. 26).

[97] Wikipedia contributors. *Logistic regression — Wikipedia, The Free Encyclopedia*. [Online; accessed 19-August-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Logistic_regression&oldid=854182949 (p. 29).

[98] Wikipedia contributors. *Mel-frequency cepstrum — Wikipedia, The Free Encyclopedia*. 2018. URL: https://en.wikipedia.org/w/index.php?title=Mel-frequency_cepstrum&oldid=835415759 (p. 5).

[99] Wikipedia contributors. *Point distribution model — Wikipedia, The Free Encyclopedia*. [Online; accessed 9-August-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Point_distribution_model&oldid=832628847 (pp. 48, 49).

[100] Wikipedia contributors. *Random forest — Wikipedia, The Free Encyclopedia*. 2018. URL: https://en.wikipedia.org/w/index.php?title=Random_forest&oldid=854167685 (pp. 32, 34).

[101] Wikipedia contributors. *Regression analysis — Wikipedia, The Free Encyclopedia*. [Online; accessed 3-August-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Regression_analysis&oldid=850022021 (p. 25).

[102] Wikipedia contributors. *Statistical classification — Wikipedia, The Free Encyclopedia*. [Online; accessed 4-September-2018]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Statistical_classification&oldid=856605904 (p. 24).

[103] Zhe Wu et al. "Deception Detection in Videos". In: *CoRR* abs/1712.04415 (2017). URL: http://arxiv.org/abs/1712.04415 (p. 11).

[104] A. Zadeh, T. Baltrušaitis, and L. Morency. "Convolutional Experts Constrained Local Model for Facial Landmark Detection". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. July 2017. DOI: 10.1109/CVPRW.2017.256 (pp. 46–48).

[105] Xing Zhang et al. "BP4D-Spontaneous: a high-resolution spontaneous 3D dynamic facial expression database". In: *Image and Vision Computing* 32.10 (2014). DOI: https://doi.org/10.1016/j.imavis.2014.06.002. URL: http://www.sciencedirect.com/science/article/pii/S0262885614001012 (pp. 15, 49).

List of Figures