RELATED WORK:

There is an observation in several works of the past few years that the fusion of modalities for emotion recognition tasks gets better results than using only one modality. In found studies, popular techniques are Convolutional Neural Networks (CNNs) or Transformers. Let us address CNNs first.

In the study of Sultana et al. [add link], CNNs were utilized for both audio and visual modalities in the emotion recognition process. A one-dimensional CNN architecture was employed for audio data to learn complex patterns and representations from the audio domain. On the other hand, a three-dimensional CNN framework was used to analyze video frames and extract essential facial characteristics for video data. "The fusion technique involve[d] concatenating and extending the outputs of the audio and visual models." [add link] The study's results indicate that the multimodal fusion model, combining audio and visual data for emotion recognition, achieved a respectable accuracy level of 66.90% [add link].

Another study that used CNNs for a combination of audio and visual information was conducted by Haddad et al. [add link]. CNNs were used for audio data with an AlexNet architecture and for visual data with a VGG-Face network to extract features for emotion recognition. Results showed that the fusion of all input types had an accuracy of 86.36%, demonstrating the effectiveness of the multimodal approach for emotion recognition. [add link]. The main difference to these works is the use of Transformer architecture instead of CNNs. Like Haddad et al. [add link], we used the pooling methods within our architecture and analyzed the results in Chapter 4.

However, researchers are increasingly using Transformers over CNNs, as they represent a newer approach that often yields better results. Let us now address Transformers.

Sun et al. [add link] introduce a novel approach, Multimodal Cross- and Self- Attention Network (MCSAN), to effectively fuse textual and acoustic information for accurate emotion recognition in speech. MCSAN addresses the challenge of integrating linguistic and acoustic information in Speech Emotion Recognition by employing parallel cross- and self-attention modules. Cross and self-attention are attention mechanisms in Transformer architecture that mix two different embedding sequences; they will be explained more in-depth later in the Background chapter. Licai Sun et al. [add link] achieved significantly better performance when fusing linguistic and acoustic information than using either modality alone for Speech Emotion Recognition. When comparing the performance of MCSAN with and without audio or text inputs, the model achieved higher accuracy and effectiveness when both modalities were combined, showcasing the value of multimodal fusion in improving emotion recognition. The difference to the work of Sun et al. is the use of video instead of textual information.

Huang et al. [add link] focused on model-level fusion, which utilizes the Transformer model to attend to interactions between audio and visual modalities and combine it with Long short-term memory (LSTM), which focuses on capturing temporal context information effectively. This combination allows for more effective integration of audio-visual modalities information on the model level fusion, leading to improved results in continuous emotion

recognition tasks. The difference to this work is using only Transformer's attention mechanisms, rather than combining those with LSTM. Unlike temporal pooling, other pooling methods like AVG Pool, MAX Pool, and 1D CNN were used, which will be addressed later in Chapter 4.

The research by Praveen et al. [add link] focused on enhancing facial and vocal modalities extracted from videos to predict valence and arousal in emotion recognition. The proposed model combined audio (A) and visual (V) modalities using a cross-attention method to understand how they relate to each other. This way, the fusion model achieved higher accuracy and robustness in capturing emotional cues than individual modalities alone [add link]. The study also explored self-attention and cross-attention fusion based on transformers; therefore, its results are compared with those in Chapter 4.

Overall, this work uses the known attention techniques in Transformer, self-, and cross-attention in audio-visual fusion, but it is conducted on a different dataset and using other fusion approaches.

JUST VIDEO:

Emotions are a crucial aspect of human communication, and a significant portion of emotional expression is conveyed through non-verbal cues such as facial expressions, body language, and tone of voice. Video-based emotion recognition allows for analyzing these non-verbal cues, providing a more comprehensive understanding of an individual's emotional state (cite p1). Multiple studies were conducted using video information for emotion recognition. In Gunavan et al.'s paper (cite), emotion recognition using video was implemented by selecting the Indian Spontaneous Expression Database (ISED) for training. The process involved preprocessing and feature extraction using LeNet and AlexNet neural networks. Parameter tuning was conducted to enhance performance, resulting in AlexNet achieving 93.00% recognition accuracy with the SGD optimizer. (cite 5)

Xu et al.(cite 1) implemented a video-based Emotion Recognition system using a pipeline consisting of four main modules: image processing, deep feature extraction, feature aggregation, and emotion classification. By using deep learning techniques such as CNNs, LSTM, and 3D Convolutional Networks (3D ConvNets), the system was able to effectively recognize emotions from videos by capturing both spatial and temporal information. (cite 2)This approach proved highly effective, achieving an accuracy of 48.01% by weighting the scores from different models, surpassing the baseline method by 9.2% (cite 4)

In the study of Qui et al. (cite 1), Video Emotion Recognition was implemented using a Dual Focus Attention Network (DFAN), which consisted of two attention modules, Time Series Focus and Frame Objects Focus, to be able to effectively identify the most informative visual cues in the video on both temporal and spatial dimensions. The results of the implementation of the DFAN showed state-of-the-art performance on two benchmark video emotion datasets: VideoEmotion and Ekman (cite 6). The recognition accuracy was significantly improved compared to other approaches, achieving recognition accuracies of 53.34% on the VideoEmotion dataset and 57.37% on the Ekman dataset (cite 6).

JUST AUDIO:

Unlike visual emotion recognition, audio emotion recognition is more challenging “due to the complexity of emotional expressions”(cite suganya1). However, speech emotion recognition using audio data alone can be effective and has shown promising results in various studies (cite suganya 2). In Suganya et al.’s work, speech emotion recognition was implemented using an end-to-end deep learning approach that applied a deep neural network directly to raw audio recordings of speech to learn high-level representations from the audio waveform. The CNN model with nine weight layers achieved an overall accuracy of 68.6% for IEMOCAP over four emotions and 85.62% for EmoDB over seven emotions. The results showed that the performance of the proposed model was better compared to traditional machine learning approaches and deep learning on the spectrogram of audio recordings in the same datasets (cite 1).

Zheng et al. implemented a speech emotion recognition system using an acoustic segment model (ASM) to segment speech data and a deep neural network (DNN) classifier for emotion classification. The results showed a weighted accuracy of 73.9% and an unweighted accuracy of 70.8% on the IEMOCAP dataset, outperforming state-of-the-art methods. (cite 1) Additionally, achieving accuracies of 58.5% for anger, 84.6% for neutral, 18.3% for happy, and 76.8% for sad emotions demonstrate a reasonable performance in distinguishing between different emotional states. (cite 4)

Koolagudi et al. (cite 16) note that emotions are often expressed through multiple modalities, such as facial expressions, gestures, and tone of voice. Combining information from multiple modalities (e.g., audio and visual cues) can improve the accuracy of emotion recognition systems.

SVR:

Support Vector Regression (SVR) is a machine learning algorithm for regression problems, aiming to minimize the generalization error bound. (Basak 2007)(https://www.semanticscholar.org/paper/Support-Vector-Regression-Basak-Pal/ade375be27e7c38927d44894c7c0680b776702b3)

SVR is an extension of Support Vector Machines (SVM); therefore, it is necessary to explain SVM first.

Typically used in classification, SVM seeks to identify a hyperplane that effectively separates data points into distinct classes, maximizing the margin between the hyperplane and the nearest data points of each class while minimizing classification errors. When new data points need to be classified, the hyperplane acts as the decision boundary.(cite website https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/)

BILD 0

Figure 0.0 shows the assignment to a category divided by the hyperplane. This data is called linearly separable.

BILD 1

Sometimes, the data is more complex, and a simple line for class separation is not possible. Like in Figure 0.1, the classes can be separated using a new dimension: the z-axis. After this transformation, the data is linearly separable in three dimensions.

BILD 2

After transforming the graph back into 2D space, the hyperplane looks like a circle in Figure 0.2

In practical scenarios, when dealing with datasets containing numerous features, applying transformations involving multiple polynomial combinations of these features can result in unnecessary computational expenses. This is because operations with the higher-dimensional vectors in the transformed feature space are required to train a support vector classifier and optimize the objective function.(https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f).
The solution to this problem is the so-called "kernel trick". It is a technique that allows you to avoid the need to explicitly convert the data into a higher-dimensional space. Instead, kernel functions focus on comparing the original data observations based on their similarities.

Kernel function takes x,z as inputs vectors in the original space and returns a scalar that represents the inner product of vectors in a potentially much higher-dimensional space.
If we have data x,z el X and a map: phi : X-> Rn then k(x,z) = <phi(x),phi(z)> is a kernel function (https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f)

\text{If we have data } \mathbf{x}, \mathbf{z} \in \mathcal{X} \text{ and a map } \phi: \mathcal{X} \rightarrow \mathbb{R}^N \text{ then} is a kernel function.

k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle

BILD FORMULA

So this way, it computes these inner products without ever having to map the data points into the higher dimensions, which does not require a high computational effort. Naturally, all these calculations happen inside the kernel functions when implementing it. (cite ski kit learn). Linear Kernel is the simplest kernel form, used when the data is linearly separable. In SVR, it effectively means no transformation is needed, and the original feature space is used. The Equation looks like k(x,z)=xTz.
Other popular kernels are polynomial, radial basis function (RBF), and sigmoid. (cite paper https://dergipark.org.tr/tr/download/article-file/1047384).

In SVR, on the other hand, the hyperplane represents the regression line that minimizes the error between the predicted and actual output values while maximizing the margin. The hyperplane is determined by support vectors, which are the data points closest to the

hyperplane and significantly influence the model, while other data points have little or no impact. (cite https://medium.com/@vk.viswa/support-vector-regression-unleashing-the-power-of-non-linear-predictive-modeling-d4495836884).

So, in this case, the output is not a categorical label; it is a continuous value representing the predicted target variable for each data point. (cite website https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/)
In emotion recognition tasks, arousal and valence are essential components typically evaluated on continuous scales (Citron, Weekes, & Ferstl, 2012 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4098114/). Therefore, SVR is well-suited for such scenarios as it involves estimating a position along a continuous range, making it a more appropriate option.

BILD (https://medium.com/it-paragon/support-vector-machine-regression-cf65348b6345)

Figure 0.4 shows Linear SVR and Non-linear SVR. Each graph represents the concept of fitting an SVR model to data points marked with triangles and squares. The goal is to find the function that best fits the data while allowing for some errors within a certain margin. The following parameters are displayed:

1. **Epsilon (ε)**: the width of the margin around the predicted function. Data points within this margin are considered acceptable.
2. **Slack Variables (ξ)**: the distances by which the data points exceed the epsilon margin. The goal is to minimize the slack variables, which means minimizing the errors outside the margin.

Bild ueberschrift: **Linear SVR** and **Non-linear SVR after a transformation of the features into a higher-dimensional space via a kernel function**

In practice, Grid Search Cross-Validation (GridSearchCV) inside SVR can be used in emotion recognition. This method works through multiple combinations of parameter tunes to determine which tune gives the best performance.
Each combination of C and kernel will be tried in a defined parameter grid. The regularization parameter C determines the trade-off between achieving a low training error and a low testing error (overfitting). The kernel parameter selects the type of kernel to be used in the algorithm. Once GridSearchCV has tested all parameter combinations, it will identify and return the model with the best performance metrics. This model is then used to predict arousal or valence on the development and test datasets.
(https://scikit-learn.org/stable/modules/grid_search.html)

DECISION TREE:

Decision trees are a machine learning algorithm used for classification and regression tasks. They use the features within a dataset to progressively split it into smaller, more manageable subsets designed to minimize impurity in each resulting subset.

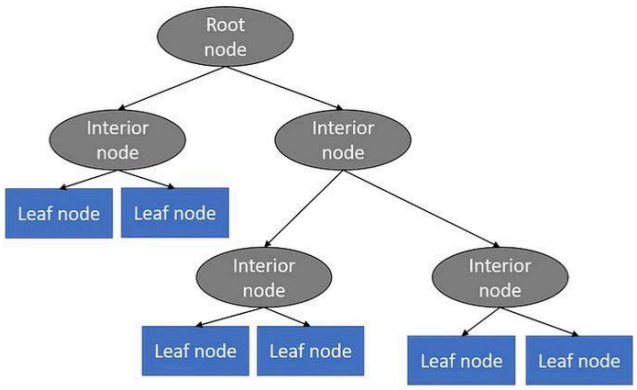
Decision Tree Classifiers deal with categorical target variables. The classifier traverses from the tree's root to a leaf node that matches the features of the new input data. The predicted category is typically the one that represents the majority of the data points in that leaf node.

Decision Tree Regressor (DTR), on the other hand, is used to predict continuous target variables.

(https://medium.com/@aaryanohekar277/what-is-the-difference-between-a-decision-tree-classifier-and-a-decision-tree-regressor-36641bd6559c) Since arousal and valence are typically measured on a continuous scale, indicating degrees of emotional intensity and positivity/negativity \cite{citron}, and a regressor is designed to predict such continuous outcomes, DTR seems a more suitable algorithm for the emotion recognition task. Therefore, this subchapter will focus on the decision tree regressor.

As seen in Figure 0.0, the decision tree's structure consists of nodes, branches, and leaves. The main node in a tree is known as the root node, from which the tree branches out into interior nodes and leaf nodes. Each interior node represents a decision point and splits into two child nodes, making the tree a binary tree.

BILD basic tree 0.0
https://towardsdatascience.com/machine-learning-basics-decision-tree-regression-1d73ea003fda



The growth of a regression tree involves dividing the predictor space by taking the entire set of feature variables (X1, X2, ... , Xp) and splitting it into several distinct and non-overlapping regions (R1, R2, ..., Rj)

First, recursive binary splitting is used to construct the regions R1, ..., Rj. For each potential split, the Residual Sum of Squares (RSS) is calculated to measure the effectiveness of the split as follows

$$\text{RSS} = \sum_{j=1}^{J} \sum_{i \in Rj} (y_i - \hat{y}_{Rj})^2$$

where Rj represents one of the regions created by the split, yi are the actual target values in that region, and y^Rj is the average of target values within region Rj.

The goal is to find the split with the lowest total RSS to reduce errors in the model. This means taking a feature and a specific split point where the differences between the actual and average values in each group are smallest. For any feature j and a split point s, the data is divided into two groups:

$$R_1(j, s) = \{X | X_j < s\} \text{ and } R_2(j, s) = \{X | X_j \geq s\}, \qquad (8.2)$$

The algorithm selects the split that minimizes the errors in predictions for the two resulting regions:

$$\sum_{i:\, x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\, x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2,$$

After the best split is chosen, the process is repeated for each new region created by the previous split. This continues until a stopping criterion is reached, like a minimum number of data points per region.

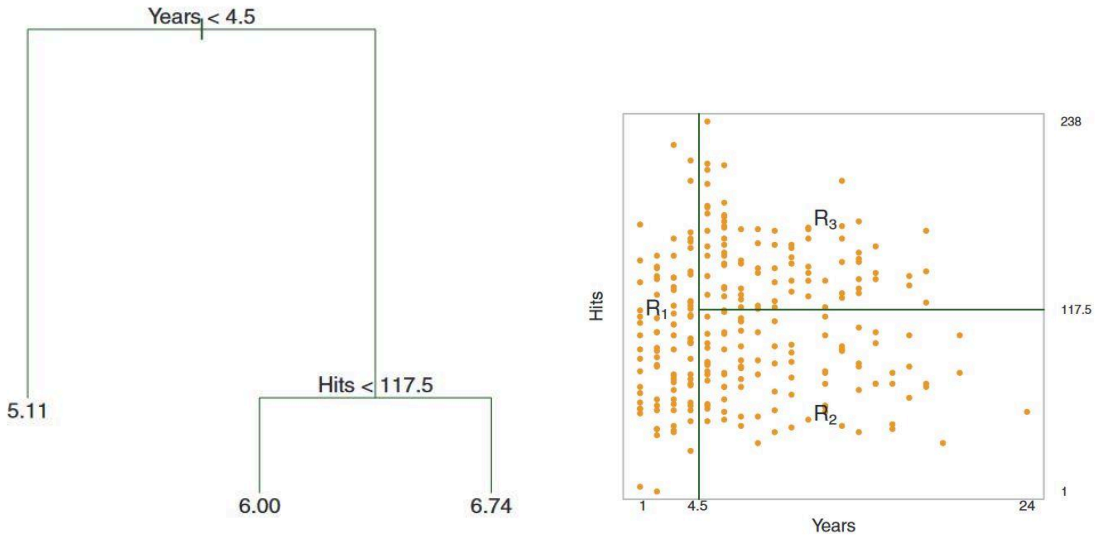


Figure 0.2 : Example of a Regression Tree

Figure 0.2 shows a regression tree for predicting the log salary of baseball players based on the number of years they have played in the major leagues and the number of hits they made in the previous year. The tree structure is displayed with nodes indicating the decision rules (e.g., "Years < 4.5") and leaves showing the average log salary for players falling into each category. This visual can help see how a decision tree makes binary splits based on feature thresholds.

Figure 0.3 : Partition of Prediction Space

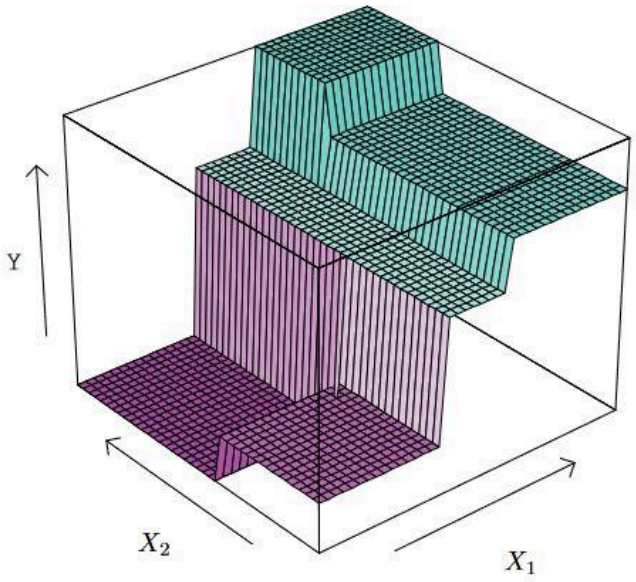
Accompanying Figure 0.2, Figure 0.3 visually represents how the predictor space is divided into different regions based on the splits defined in the regression tree. Each region (R1, R2, R3) corresponds to a combination of conditions from the tree's nodes, and the average salaries for these regions are graphically shown in a two-dimensional space of 'Years' and 'Hits.' This figure helps in understanding how continuous predictor variables are used to create distinct regions within which predictions are constant.

In typical representations of the prediction space, especially in 2D plots, the target variable that the model needs to predict is not visible. Such plots display only the input features, and showing the target variable would require a 3D representation to capture the relationship between the features and the target. Figure 0.4 provides an example of how a prediction space can be visualized along with the target variable Y.

After a tree has been grown, it may be pruned back to prevent overfitting. Pruning involves removing subtrees from a fully grown tree to improve the tree's predictive ability on unseen data.

After the decision tree is built, predictions can be made by guiding a new data point through the tree down to the correct leaf. Once the data point reaches a leaf, the predicted value is just the average of all the training data values that also end up in that leaf.

https://www2.stat.duke.edu/~rcs46/lectures_2017/08-trees/08-tree-regression.pdf



A 3D example plot of the prediction space

Example:

Step-by-Step Example: Predicting House Prices

Suppose we have the following small dataset of house prices:

House ID	Size (sq ft)	Bedrooms	Price ($000)

1	1400	3	250
2	1600	3	270
3	1700	4	310
4	1850	3	340
5	1100	2	220

Objective: Build a decision tree to predict the house price based on size and bedrooms.

Step 1: Choose a Feature and Value to Split On

Let's consider splitting by "Size" first. We need to find the best threshold that minimizes RSS:

- **Possible splits**: 1200, 1500, 1750, 1900 sq ft (midpoints between sorted values).

Step 2: Calculate RSS for Each Split

Let's calculate RSS for the split at 1500 sq ft:

- **Region 1 (Size < 1500 sq ft)**:
  - Houses in this region: 1, 5
  - Mean Price: (250 + 220) / 2 = 235
  - RSS = (250 - 235)^2 + (220 - 235)^2 = 450
- **Region 2 (Size ≥ 1500 sq ft)**:
  - Houses in this region: 2, 3, 4
  - Mean Price: (270 + 310 + 340) / 3 ≈ 307
  - RSS = (270 - 307)^2 + (310 - 307)^2 + (340 - 307)^2 = 1503

Total RSS for split at 1500 sq ft = 450 + 1503 = 1953

Step 3: Choose the Best Split

After calculating the RSS for all possible splits, suppose the split at 1500 sq ft results in the lowest RSS compared to other splits.

Step 4: Repeat for Each Region

Now, we would take each of the resulting regions and look for the best feature and threshold to split further, following the same steps, until a stopping criterion is met (like a minimum number of houses or no further reduction in RSS).

Final Tree and Prediction

Assuming no further splits are beneficial or possible according to our stopping criteria, our simple decision tree for this example could look like:

- **Node**: Split at Size 1500 sq ft
    - **Left Child (Size < 1500 sq ft)**: Predict price = 235 (average of House 1 and 5)
    - **Right Child (Size ≥ 1500 sq ft)**: Predict price = 307 (average of House 2, 3, and 4)

For a new house to predict:

- If its size is less than 1500 sq ft, predict $235,000.
- If its size is 1500 sq ft or more, predict $307,000.

This example simplifies the decision tree process, focusing on understanding the basic steps and calculations involved. In practice, more complex decisions, more data, and additional features would be considered to refine the predictions further.

BAYESIAN RIDGE:

"All of Statistics: A Concise Course in Statistical Inference," and it's written by Larry Wasserman
https://www.stat.cmu.edu/~larry/=sml/Bayes.pdf

Bayesian Ridge Regression is a machine learning algorithm that predicts outcomes in regression tasks https://scikit-learn.org/1.0/modules/linear_model.html#ridge-regression . As the name suggests, it consists of Bayesian inference and Ridge Regression. Both terms will be explained first to understand the algorithm.

Bayesian inference is (BI) a statistical method that allows to make better predictions and decisions by using prior knowledge and new evidence. It starts with a prior distribution, the initial belief about the parameters before seeing the data. This could represent assumptions or existing knowledge about a situation.

When new data is observed, Bayesian inference doesn't just consider this new data alone. Instead, it combines the new data with the prior beliefs to form the posterior distribution. This posterior gives a new and improved understanding of the parameters after taking into account the new evidence.

The Bayesian inference procedure can be broken down into three main steps:

1. **Prior Distribution**: Defining the prior distribution that reflects the initial belief about the parameters. This can be based on previous studies, expert opinions or assumptions.
2. **Likelihood**: Considering the likelihood, which is the probability of observing the data given particular values of the parameters. This part focuses only on the new data.
3. **Posterior Distribution**: Combining the prior distribution with the likelihood using Bayes' Theorem to compute the posterior distribution. This step mathematically blends prior beliefs and the evidence from the data (likelihood) to update the understanding of the parameters.

Bayes' Theorem for probability distributions is often stated as:
Posterior ∝ Likelihood × Prior

book "Pattern Classification" by Richard O. Duda, Peter E. Hart, and David G. Stork
https://www.stat.cmu.edu/~brian/463-663/week09/Chapter%2003.pdf

where the symbol "∝" means "is proportional to". This formula means that the posterior distribution is proportional to the likelihood of the observed data multiplied by the prior distribution.

Ridge Regression (RR), also known as Tikhonov or L2 regularization, is a technique that deals with multicollinearity, which means that the predictors in a regression model are linearly dependent.
https://en.wikipedia.org/wiki/Multicollinearity
https://thaddeus-segura.com/lasso-ridge/
https://www.kaggle.com/code/avadhutvarvatkar/ridge-regression

RR is an extension of ordinary least squares (OLS) regression, a technique for finding the best-fitting line through a set of data points by minimizing the differences between the observed values and the values predicted by the line.
https://www.xlstat.com/de/loesungen/eigenschaften/ordinary-least-squares-regression-ols

Upon that, a penalty term is added to the OLS loss function to shrink the regression coefficients toward zero, which can lead to more stable and reliable estimates.
https://www.kaggle.com/code/avadhutvarvatkar/ridge-regression

The objective function for ridge regression is:

minimize||Y−Xβ||2+λ||β||2minimize||Y−Xβ||2+λ||β||2
https://www.kaggle.com/code/avadhutvarvatkar/ridge-regression

$$\text{minimize} \quad ||Y - X\beta||^2 + \lambda||\beta||^2$$

where:

- YY is the dependent variable,
- XX is the matrix of independent variables,
- ββ is the vector of coefficients to be estimated,
- λλ is the tuning parameter that controls the amount of shrinkage.

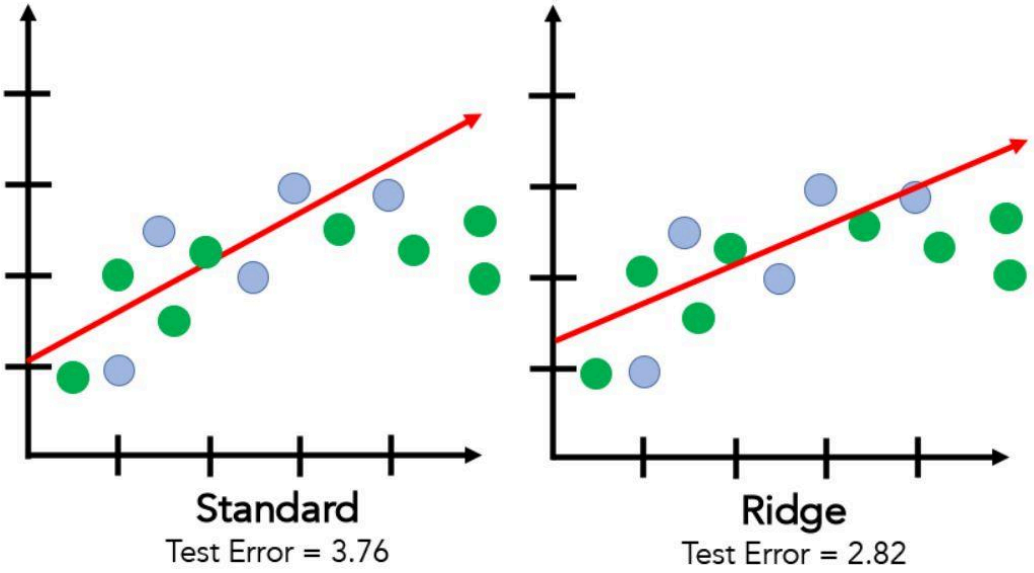


Fig 0.0 : an example of standard linear regression compared to ridge regression. Blue represents training points and green represents test points.
https://thaddeus-segura.com/lasso-ridge/

Figure 0.0 shows that by introducing the penalty term and changing the slope of the line, ridge regression does not fit the training data as closely as the standard regression line anymore, but it highly improves the prediction on test data, therefore the test error is lower.

Altogether, Bayesian Ridge Regression (BRR) is a variation of the Ridge Regression with a difference of applying probability to the regression problem. Instead of finding the single "best" value of the model parameters, BRR considers a range of possible models and weights them according to their probability given the existing data.

https://towardsdatascience.com/introduction-to-bayesian-linear-regression-e66e60791ea7

It combines prior beliefs about the values (the priors) with the evidence provided by the data (the likelihood) to form an updated belief (the posterior). It is obvious that it is the Bayes' Theorem, which was already explained above.

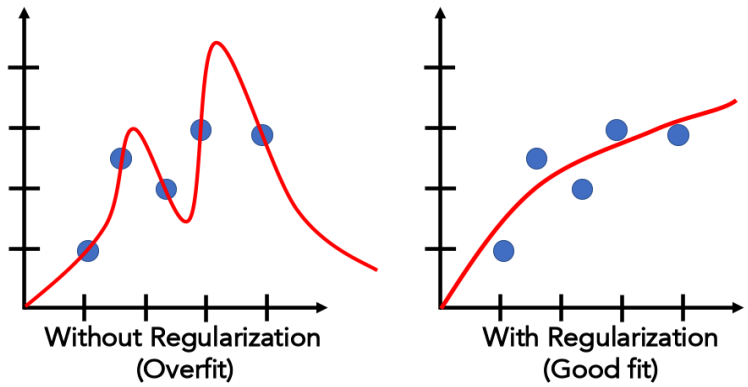
The key point is that in Bayesian Ridge Regression, the parameters are not fixed values but distributions. This allows to account for the uncertainty in the predictions: BRR not only determines the placement of the line but also the level of confidence in different segments of the line.

The main idea of BRR is that the model learns and adapts as it sees more data, becoming less wrong over time.

BILD

https://scikit-learn.org/1.0/modules/linear_model.html#ridge-regression

Figure 0.1 shows different estimates for the weights of the features in a Bayesian Ridge Regression model. The green line represents the estimates given by the BRR, which are generally more steady than the standard OLS regression, shown in blue. The yellow line represents the actual weights. Since Bayesian Ridge Regression avoids swinging to extremes, as seen in the OLS estimates, it helps make the model more reliable and a “good fit,” as shown in Figure 0.2 too.



https://www.kaggle.com/code/thaddeussegura/enough-to-be-dangerous-lasso-and-ridge-regression

https://towardsdatascience.com/introduction-to-bayesian-linear-regression-e66e60791ea7

Bayesian Ridge Regression also seems suitable for the emotion recognition task when predicting arousal and valence values since it naturally includes regularization, which helps to prevent overfitting, as stated earlier. Emotion recognition data is often considered uncertain https://www.sciencedirect.com/science/article/pii/S0262885624000040 and Bayesian inference is good at handling such uncertainty by providing a distribution of possible outcomes, not just a single prediction, as also stated before. The data can also often contain extreme values https://www.mdpi.com/1424-8220/20/21/6367 and BRR is, in this case, beneficial since it is more robust to outliers (footnote: data points that differ significantly from other observations https://en.wikipedia.org/wiki/Outlier) as it was brought in an example in Figure 0.1

Datasets:

AFEW:
https://ibug.doc.ic.ac.uk/resources/afew-va-database/ and paper
https://ibug.doc.ic.ac.uk/media/uploads/documents/afew-va.pdf

SEWA:
https://arxiv.org/abs/1901.02839
https://arxiv.org/pdf/1901.02839.pdf

Model Evaluation Metrics: Discuss the metrics used for evaluating the performance of the SVR models, such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). Explain their significance in quantifying the average error between the predicted and actual values, which provides insight into the model's accuracy.

AFEW: https://ibug.doc.ic.ac.uk/media/uploads/documents/afew-va.pdf

One of the datasets used in this thesis's experiments is AFEW-VA. It consists of 600 videos extracted from feature films. These videos show various facial expressions captured under challenging indoor and outdoor conditions, including complex, cluttered backgrounds, poor illumination, large out-of-plane head rotations, scale variations, and occlusions. AFEW-VA includes annotations for arousal and valence levels for each frame of the video clips. In total, more than 30,000 frames were annotated with valence and arousal levels in the range of -10 to 10.

As seen in the diagram in Figure 0.0, the distribution of valence and arousal values in the dataset shows a wide range of values, which means there is a variability of emotional expressions in the videos. The distribution of the annotations in the valence and arousal can also be seen on a two-dimensional scatter plot in Figure 0.1

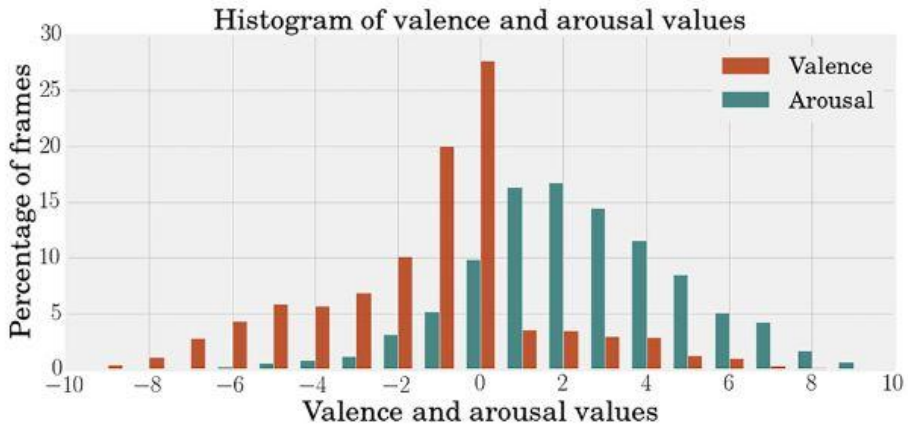


Figure: Distribution of arousal and valence values in AFEW-VA dataset.https://ibug.doc.ic.ac.uk/media/uploads/documents/afew-va.pdf

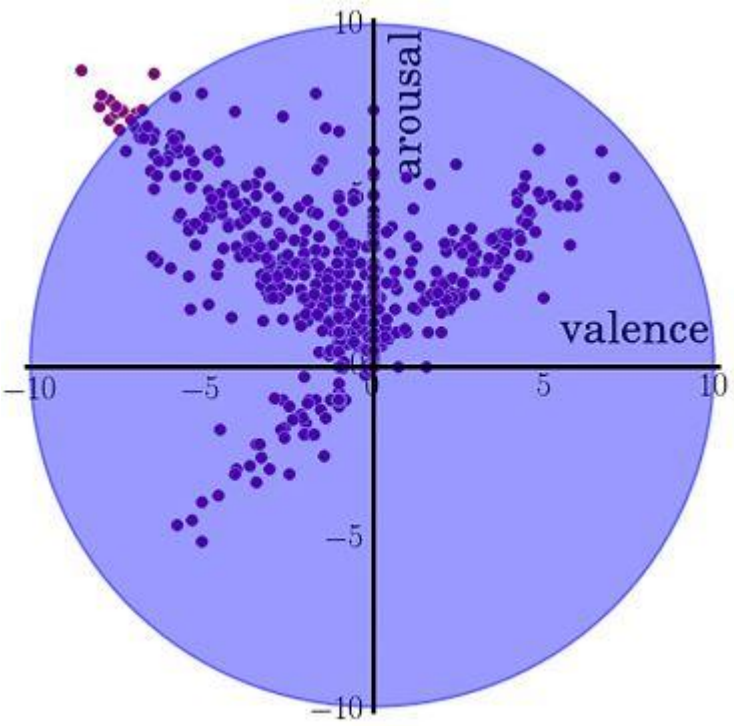


Figure: Distribution of valence and arousal values in AFEW-VA. 2D plot.
https://ibug.doc.ic.ac.uk/media/uploads/documents/afew-va.pdf

The AFEW-VA dataset addresses the limitations of existing affective datasets by providing detailed annotations for valence and arousal in challenging, real-world conditions in order to understand human emotions in more naturalistic settings.

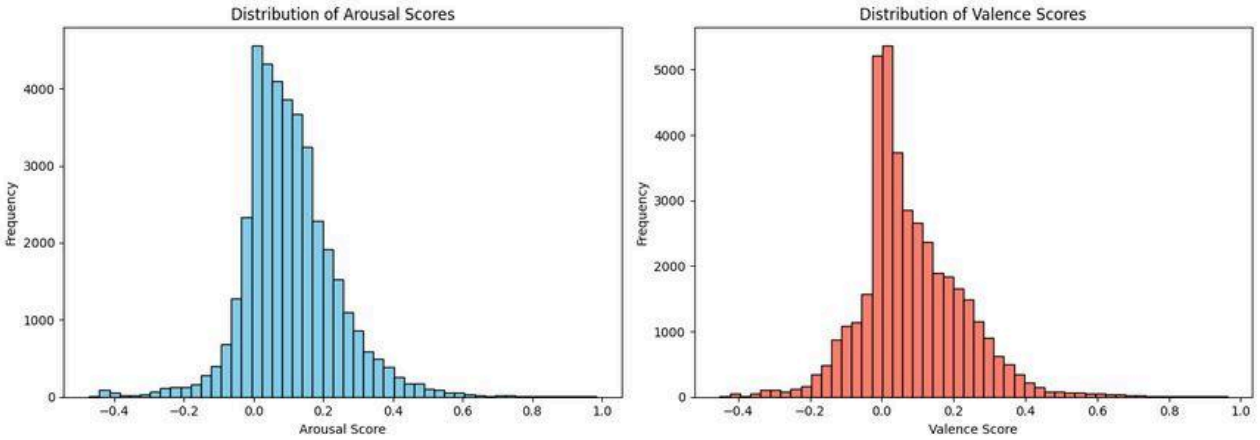
https://ibug.doc.ic.ac.uk/media/uploads/documents/afew-va.pdf

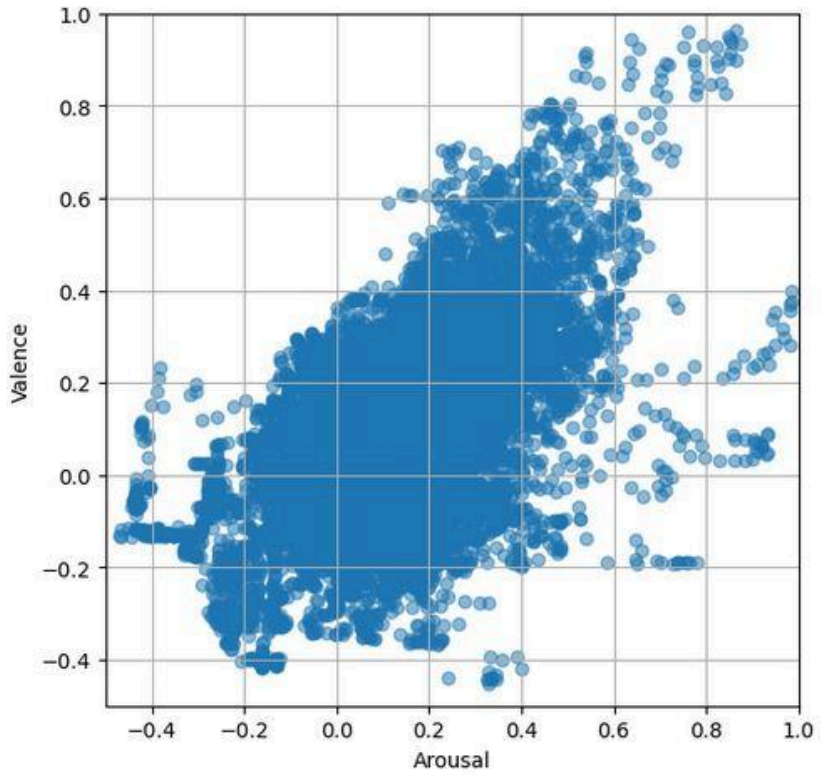
Since the dataset contains a large amount of frames, each of them already annotated with arousal and valence levels, it seems suitable for the experiments in this thesis. AFEW-VA will be used for both the baseline and the video model.

SEWA:
https://arxiv.org/pdf/1901.02839
Another dataset that was used in the experiments is SEWA. It consists of audio-visual recordings of spontaneous behavior captured in unconstrained, real-world environments using standard web cameras and microphones. Participants in the dataset were divided into pairs based on their cultural background, age, and gender. They were required to know each other personally to promote natural interactions during the recordings. The people on the recordings are from different age groups, genders, and cultural backgrounds. Kossaifi et al. state that audio-visual data consists of 398 people from six cultures (British, German, Hungarian, Greek, Serbian, and Chinese), 50% female and aged 18 to 65.
The recordings are annotated with facial action units, facial landmarks, vocal and verbal cues, as well as continuously valued emotion dimensions such as valence and arousal, which are crucial for emotion recognition tasks in this thesis.

Figure 0.3 illustrates the distribution of arousal and valence, showcasing the range of emotional variability captured in the recordings.

Figure 0.4 displays the same arousal and valence values but on a 2D scatter plot. It is visible that arousal ranges approximately from -0.4 to 1.0, and valence ranges approximately from -0.4 to 0.8





The SEWA dataset addresses the limitations of existing databases by offering rich annotations in terms of multiple behavioral cues, demographic variability, and task diversity.https://arxiv.org/pdf/1901.02839

It seems suitable for emotion recognition task in this thesis too, because it consists of not only visual, but also audio recordings, which will be used in the fusion later on. Also, all the

recordings are annotated with arousal and valence values, which are also needed in the experiments. SEWA dataset will be used for baseline, video and audio model, since it consists audio recordings, and of course the fusion.

Since the AFEW-VA and SEWA datasets lack the features necessary for predicting arousal and valence, these must be extracted beforehand.

Video feature extraction was executed using an EfficientNet-B1 model, which was pre-trained on ten emotion recognition datasets. This initial training gave the model a basic understanding of subtle emotional details in video frames. For each frame, the model extracted a set of 256 deep embeddings. These embeddings were then used as inputs for further processing by emotion recognition models that will be presented in Chapter 4.

Figure 0.0 illustrates a visual representation of the facial feature extraction pipeline. It shows a sequence of processing steps that starts with an input image and then moves through various layers of an EfficientNet-B1 neural network architecture.

Detailed documentation of the model's training process and the datasets incorporated can be accessed \href{https://github.com/ECCV-2024-cross-multi-modal/Cross-Multi-Modal-Fusion-Approach/tree/main/emotion_recognition}{here}.

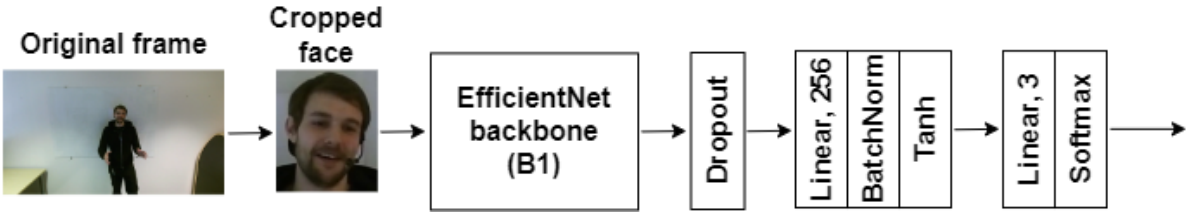


Figure 0.0: The pipeline EfficientNet-B1 of the model https://github.com/ECCV-2024-cross-multi-modal/Cross-Multi-Modal-Fusion-Approach/tree/main/emotion_recognition

For the extraction of audio features, the Wav2Vec2 model, which has been pre-trained on 960 hours of Librispeech on 16kHz sampled speech audio, was used. https://arxiv.org/pdf/2006.11477

This model features a multilayer convolutional feature encoder, from which we used the last hidden state to extract the audio features. The features were extracted using varying lengths of temporal windows—specifically, 1, 2, 3, and 4 seconds. For each length, the window was moved forward by half its size. More information about the model can be found \href{https://huggingface.co/facebook/wav2vec2-base-960h}{here}.

Afterward, the features, along with the file path of each frame and the corresponding arousal and valence values, were stored in a CSV file. In addition, SEWA's video data also included timestamps, and audio data had start and end time steps.

Evaluation Metrics:

https://d1wqtxts1xzle7.cloudfront.net/73250877/IRJET_V8I9127-libre.pdf?1634803489=&response-content-disposition=inline%3B+filename%3DIRJET_Comparative_Assessment_of_Regressi.pdf&Expires=1714369192&Signature=AjT-UGatJLXFxUUoyBghptGiFJevutuEUJL-Fek2NGN8Zr-miATWllxnGxDYuqktcAIguHPZ9ixEKiE-iOaK9Lc4uOy~My-2U9rsAHh5LamYpbkk4C5DS2K3K6HKwb0NlNmvrEn1BI5X9ComLhu9L0F5ZqQCIwkZ2VpezfegnakP8MWI92kORteMM23ClQEme~fVM1M-cxIyJcFGePkh~jKt4yIRDd6kS~Sl0M2mhTq4B24QL7yb95bXgPuaeqV5P9ZBntSlLmlf2TXhDXIunqp5NYVxJoeEGYGG7l3YAX92-Yr-LQgNTGVfzZV-VHI9WWMP2uqWHJ6PklPLYyVx2Q__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

Three metrics, mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE), are used in the experiments to evaluate the performance of regression models. Each provides a different perspective on the error between the predicted and the actual values.

Mean Absolute Error (MAE) is a simple metric that calculates the average of the absolute differences between the predicted and actual values. In other words, it simply says how far away the predictions are from the actual values.

It is calculated using this formula:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

MAE=n1∑i=1n∣yi−y^i∣,

where n is the number of predictions, yi are the actual observed values, y^i are the predicted values.

Figure 0.0: Visual representation of MAE using dummy data, where arousal MAE = 0.08 and valence MAE = 0.10.

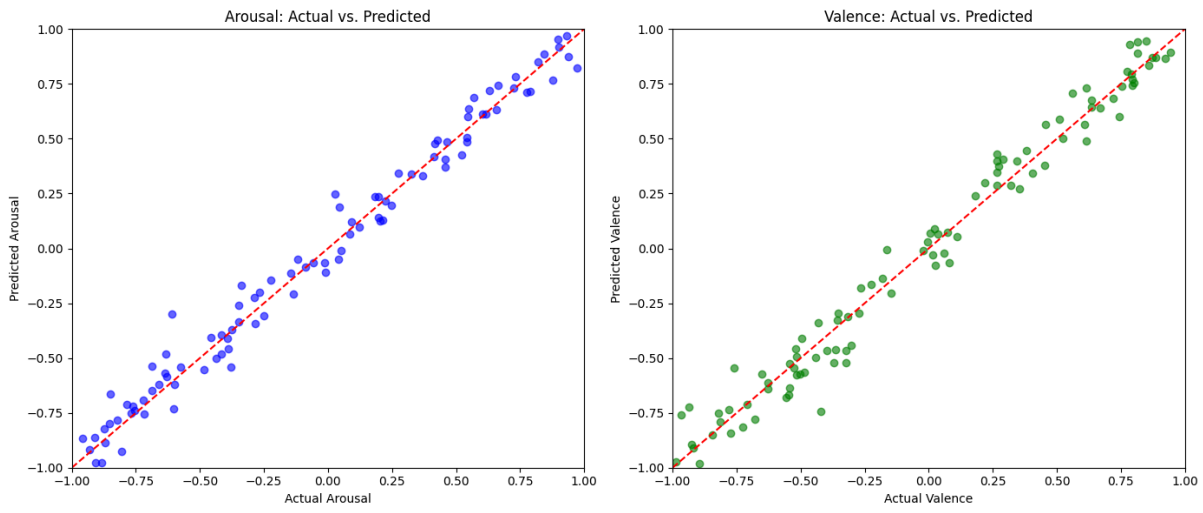


Figure 0.1 presents two plots for a visual understanding of MAE. They show the actual and predicted values for arousal and valence. The red dashed line represents where points would lie if the predictions were correct. The closer the points are to the dashed line, the better the model makes predictions.

Since this metric is easily interpretable from the raw value alone, it is useful for evaluating neural networks because it deals with many different calculations that need to be compared in the experiments. The use of absolute values of the differences also makes it more robust to outliers.

https://medium.com/@ompramod9921/loss-functions-unraveled-805d76c239fe

Mean Squared Error (MSE) is the average of the squared difference between predicted and actual values, so it is always a positive value.

This squaring of the differences has the effect of amplifying larger errors. In other words, big mistakes are even bigger when they are squared, which makes them stand out more. This helps the model correct these mistakes quickly during its learning process. Also, when the mistakes are small and close to each other, the model learns faster.

https://medium.com/@nirajan.acharya666/choosing-between-mean-squared-error-mse-and-mean-absolute-error-mae-in-regression-a-deep-dive-c16b4eeee603

MSE has the advantage of being a continuous and differentiable function, which means it is well-suited for gradient-based optimization algorithms.
https://medium.com/@ompramod9921/loss-functions-unraveled-805d76c239fe

For this reason, it is used in experiments with machine learning algorithms like SVR and Bayesian Ridge. Even though the Decision tree algorithm is not gradient-based, using MSE as a metric allows a consistent comparison between models.

The MAE metrics is determined by:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$,

where n is the number of predictions, yi are the actual observed values, y^i are the predicted values.

Root Mean Squared Error (RMSE) is the square root of MSE, which brings the error measurement back to the original units of the output variable. RMSE works by squaring the differences, summing them, dividing them by the number of observations, and finally taking the square root, which gives the standard deviation of the prediction errors. This indicates how close the line of best fit is to the set of points.

https://d1wqtxts1xzle7.cloudfront.net/73250877/IRJET_V8I9127-libre.pdf?1634803489=&response-content-disposition=inline%3B+filename%3DIRJET_Comparative_Assessment_of_Regressi.pdf&Expires=1714369192&Signature=AjT-UGatJLXFxUUoyBghptGiFJevutuEUJL-Fek2NGN8Zr-miATWllxnGxDYuqktcAIguHPZ9ixEKiE-iOaK9Lc4uOy~My-2U9rsAHh5LamYpbkk4C5DS2K3K6HKwb0NlNmvrEn1BI5X9ComLhu9L0F5ZqQCIwkZ2VpezfegnakP8MWI92kORteMM23ClQEme~fVM1M-cxIyJcFGePkh~jKt4yIRDd6kS~Sl0M2mhTq4B24QL7yb95bXgPuaeqV5P9ZBntSlLmlf2TXhDXIunqp5NYVxJoeEGYGG7l3YAX92-Yr-LQgNTGVfzZV-VHI9WWMP2uqWHJ6PkIPLYyVx2Q__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

It is similar to MAE, but RMSE is more sensitive to outliers because it gives more weight to larger errors, whereas MAE treats all errors equally.
https://medium.com/@vaibhav1403/rmse-and-mae-415470f52b58

RMSE can be calculated with this formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

In summary, we use the combination of MSE and RMSE in experiments with algorithms and combination of MAE and RMSE in neural networks to capture advantages of all metrics in used models. In all three metrics, a smaller value represents that the predictions are closer to the actual values, which means the prediction performance is better, so the goal is to find the best model with the smallest results during experiments.