# Exercise 1: Analysing Car Sales Data with Supervised and Unsupervised Learning Models


## Abstract:

This paper explores the application of both supervised and unsupervised machine learning algorithms for used automobile market analysis. Supervised models estimate automobile prices based on previous sales data, while unsupervised techniques, notably clustering, expose patterns within the dataset(Johannes Kriebel et al, 2023) .

When predicting car prices, polynomial regression consistently performs better than linear regression, particularly when taking into account numerous features at once. This paper investigates polynomials with degree 4. Furthermore, polynomial regression is superior to Random Forest Regression and Artificial Neural Network (ANN) models in terms of capturing complex patterns.

The importance of parameter selection for optimal performance is emphasized by early stopping, hyperparameter tuning, and regularization strategies for artificial neural networks. Additionally, different segments within the car sales dataset are identified by clustering techniques as k-means and DBSCAN(Garg and Kalai, 2018).


## Introduction:

With an emphasis on both super supervised and unsupervised learning approaches, this paper uses machine learning algorithms to explain the intricacies of the used automobile market.

## 1.a: Linear Vs Polynomial Regression Model

First, let us emphasize three crucial aspects: For each feature, the following methods of regression were used: linear and polynomial: year of manufacture, engine size, and mileage. Performance measurements were then used, as indicated in

Table 1, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R2 score.

| Metric/ Feature | Year of Manufacture (Linear) | Year of Manufacture (Polynomial) | Engine Size (Linear) | Engine Size (Polynomial) | Mileage (Linear) | Mileage (Polynomial) |
|---|---|---|---|---|---|---|
| **Degree** | N/A | 4 | N/A | 4 | N/A | 4 |
| **Mean Absolute Error** | 7058.19 | 5185.47 | 10970.08 | 10970.30 | 8036.38 | 6754.36 |
| **Mean Squared Error** | 125138115.69 | 97707742.96 | 228864284.82 | 228675624.90 | 154651643.40 | 130428549.27 |
| **Root Mean Squared Error** | 11186.51 | 9884.72 | 15128.26 | 15122.02 | 12435.90 | 11420.53 |
| **R2 Score** | 0.52 | 0.63 | 0.13 | 0.14 | 0.41 | 0.50 |

**Table 1 : Results for Linear and Polynomial Regression.**

All metrics showed that polynomial regression worked better than linear regression, and Figure 1 scatter plot indicates that the best factor for predicting price is "Year of Manufacture".

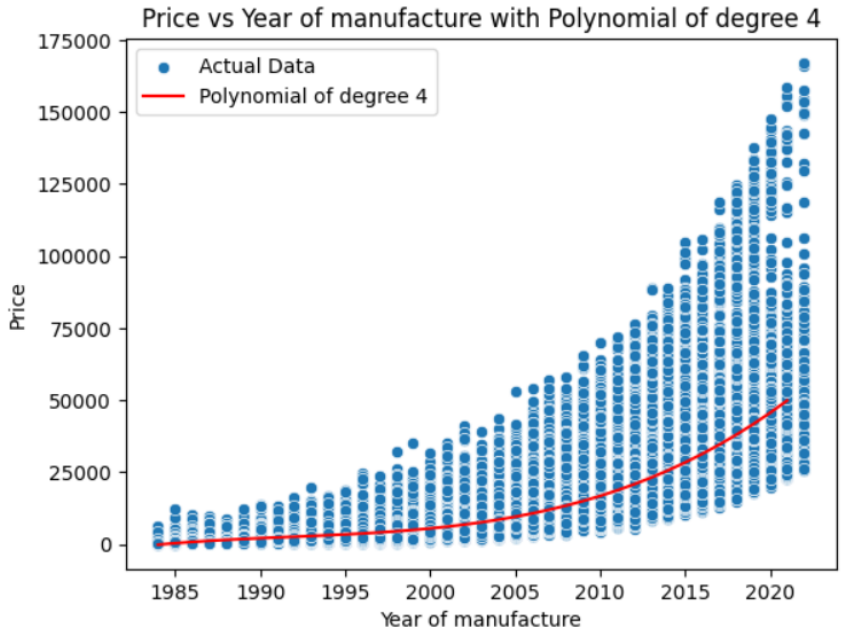



**Figure 1: Polynomial Regression (degree4) Scatter Plot.**

## 1.b: Regression models (multiple numerical variables)

| Metric/ Feature | Year of Manufacture (Polynomial) | Engine Size (Polynomial) | Mileage (Polynomial) | Regression Model (multiple numerical variables) |
|---|---|---|---|---|
| **Degree** | 4 | 4 | 4 | N/A |
| **Mean Absolute Error** | 5185.47 | 10970.30 | 6754.36 | 6167.88 |
| **Mean Squared Error** | 97707742.96 | 228675624.90 | 130428549.27 | 84405262.35 |
| **Root Mean Squared Error** | 9884.72 | 15122.02 | 11420.53 | 9187.23 |
| **R2 Score** | 0.63 | 0.14 | 0.50 | 0.68 |

**Table 2: Results for Polynomial Regression and Multiple Linear Regression Model.**

Regression models with many numerical features can be used to predict car prices with high accuracy. Selecting the appropriate parameters and monitoring model performance can help create efficient car price prediction models, as shown in Table 2.We can state that multiple Linear Regression Model consistently outperforms Polynomial Regression Models. A scatter plot is displayed in Figure 2 illustrate the correlation between the multiple linear regression model's projected and actual prices.

.

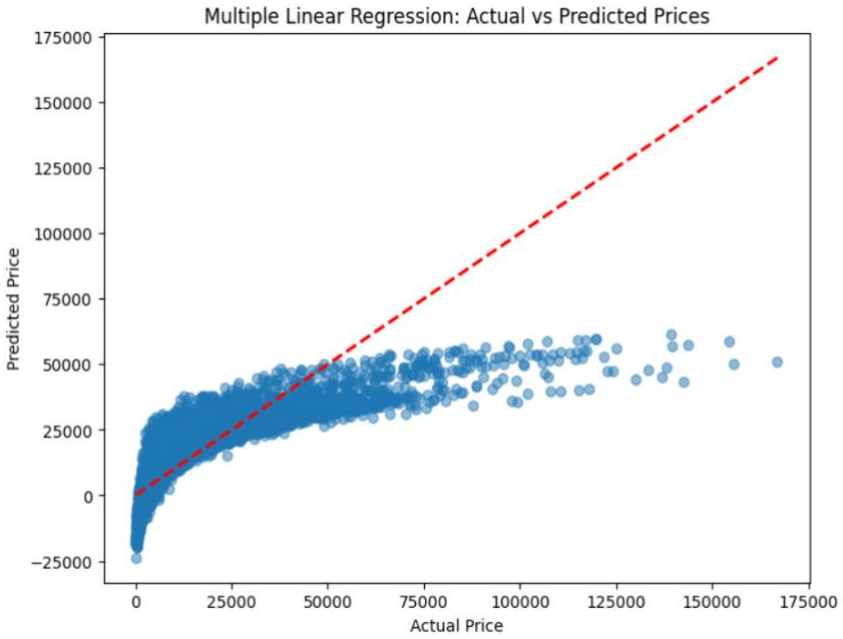



**Figure 2: Multiple Linear Regression Scatter Plot.**

## 1.c: Training a Random Forest Regressor model that uses both categorical and numerical input variables

It is advised to use a regression model that incorporates both numerical and categorical variables (such as a Random Forest Regressor model) in order to calculate performance measures as shown in Table 3

| Metric/ Feature | Year of Manufacture (Polynomial) | Engine Size (Polynomial) | Mileage (Polynomial) | Regression Model (multiple numerical variables) | Random Forest Regressor model |
|---|---|---|---|---|---|
| **Degree** | 4 | 4 | 4 | N/A | N/A |
| **Mean Absolute Error** | 5185.47 | 10970.30 | 6754.36 | 6167.88 | 332.56 |
| **Mean Squared Error** | 97707742.96 | 228675624.90 | 130428549.27 | 84405262.35 | 514920.32 |
| **Root Mean Squared Error** | 9884.72 | 15122.02 | 11420.53 | 9187.23 | 717.57 |

| | | | | | |
|---|---|---|---|---|---|
| **R2 Score** | 0.63 | 0.14 | 0.50 | 0.68 | 0.99 |

**Table 3 : Results for Polynomial Regression, Multiple Linear Regression and Random Forest Regressor Model.**

By using n_estimators = 10 to access the Random Forest Regressor model, we can obtain an R2 score of =0.99, or 1, indicating that the Random Forest Regressor model consistently outperformed the other models. The scatter plot used in Figure 3 illustrates the correlation between the actual and predicted prices as predicted by the Random Forest Regressor model.

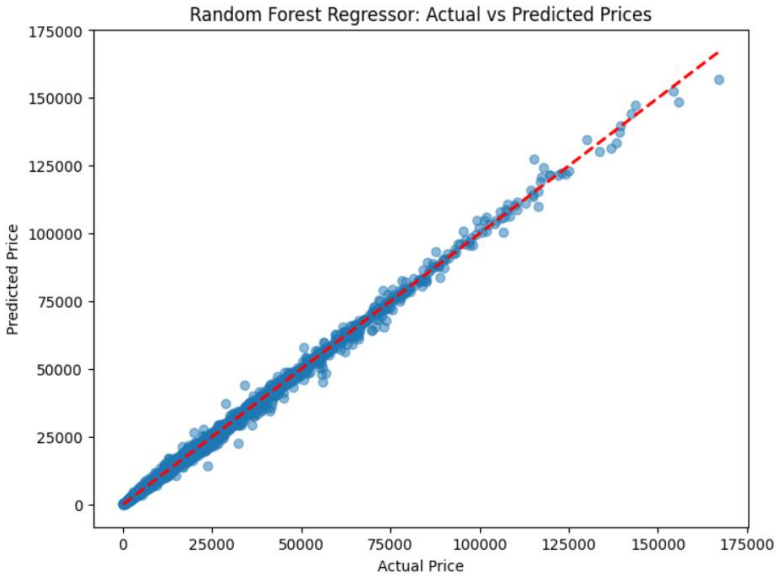



**Figure 3 : Random Forest Regressor Scatter Plot.**

## 1.d: Training a Artificial Neural Network (ANN) model.

Table 4 provides the commonly used criteria to gauge how accurate the model is at predicting car price. Designing the neural network's architectural design. There are two hidden layers in this case, with 64 hidden nodes, 10 epochs, 5 batch sizes, and 32 epochs.

| **Metric/ Feature** | **ANN Model (Epoch = 5, batch size = 32)** | **ANN Model (Epoch = 10, batch size = 32)** |
|---|---|---|
| **Mean** | 1904.07 | 1547.07 |

| | | |
|---|---|---|
| **Absolute Error** | | |
| **Mean Squared Error** | 15638973.03 | 10033925.99 |
| **Root Mean Squared Error** | 3954.61 | 3167.63 |
| **R2 Score** | 0.94 | 0.96 |

**Table 4: ANN Model with epoch value 5 and 10.**

## Hyper parameter Tuning:

Early Stopping : As demonstrated in the below experimenting with Early Stopping with the same epoch and batch size of 32 yields remarkable results when optimizing the ANN as shown in Table 5.

| **Metric/ Feature** | **ANN Model (Epoch = 5, batch size = 32)** | **ANN Model (Epoch = 10, batch size = 32)** | **ANN Model (Epoch = 5, batch size = 32, early stopping)** | **ANN Model (Epoch = 10, batch size = 32, early stopping)** |
|---|---|---|---|---|
| **Mean Absolute Error** | 1904.07 | 1547.07 | 2159.54 | 1757.58 |
| **Mean Squared Error** | 15638973.03 | 10033925.99 | 17892016.68 | 12899097.41 |
| **Root Mean Squared Error** | 3954.61 | 3167.63 | 4229.89 | 3591.53 |
| **R2Score** | 0.94 | 0.96 | 0.93 | 0.95 |

**Table 5: ANN Model with Early Stopping..**

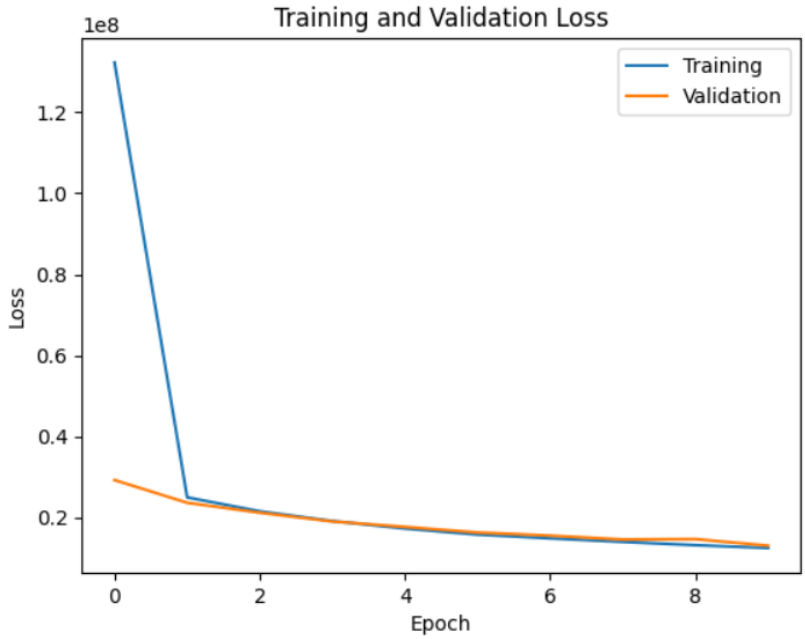



**Figure 4: ANN model graph with Early Stopping for epoch value 10.**

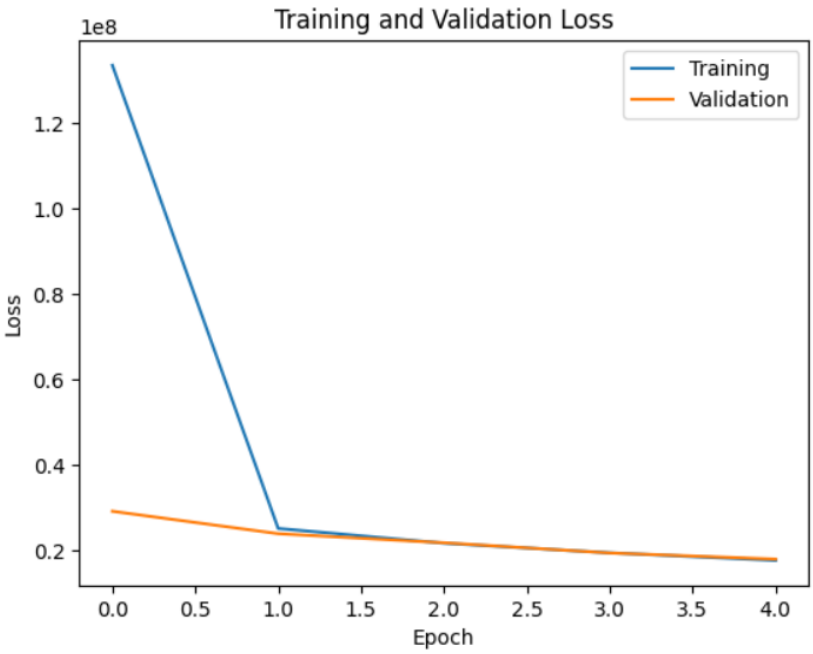



**Figure 5: ANN model graph with Early Stopping for epoch value 5.**

Dropout Layer, leaning rate and 3 hidden layers : Different learning rates were evaluated in order to determine the best setup for the ANN. Better performance metrics, as indicated in were obtained with the default rate of 0.001, dropout rate of 0.2, hidden layer of 3 with hidden nodes of 64, and epoch value of 10 as shown in Table 6.

| Metric/ Feature | ANN Model (Epoch = 5) | ANN Model (Epoch = 10) |
|---|---|---|
| **Mean Absolute Error** | 1813.36 | 1523.33 |
| **Mean Squared Error** | 13872690.50 | 7835508.87 |
| **Root Mean Squared Error** | 3724.60 | 2799.19 |
| **R2 Score** | 0.94 | 0.97 |

**Table 6: ANN Model with Dropout layer, learning rate and 3 hidden layer.**

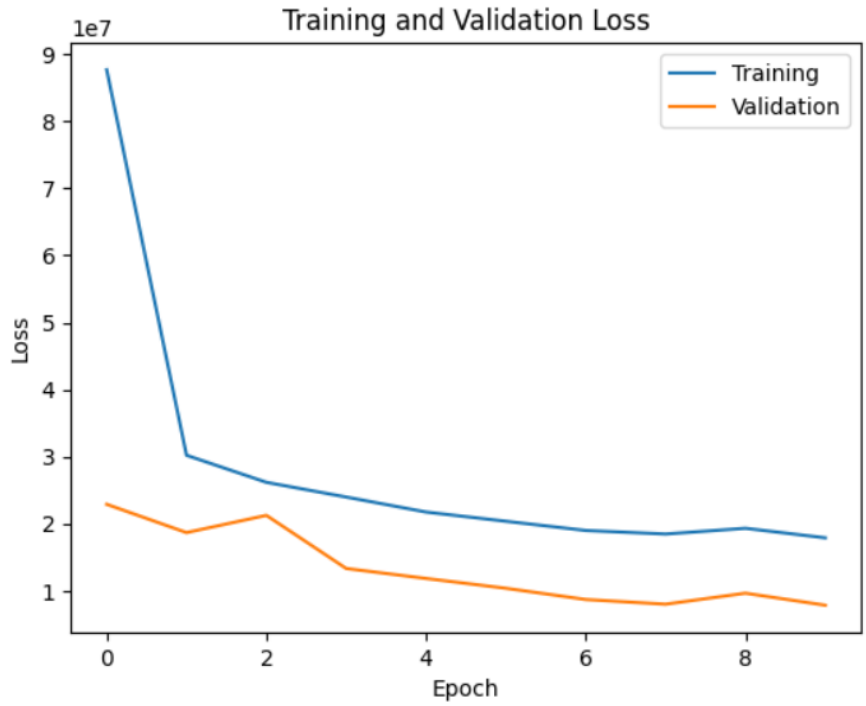



**Figure 6: ANN graph with epoch value 10.**

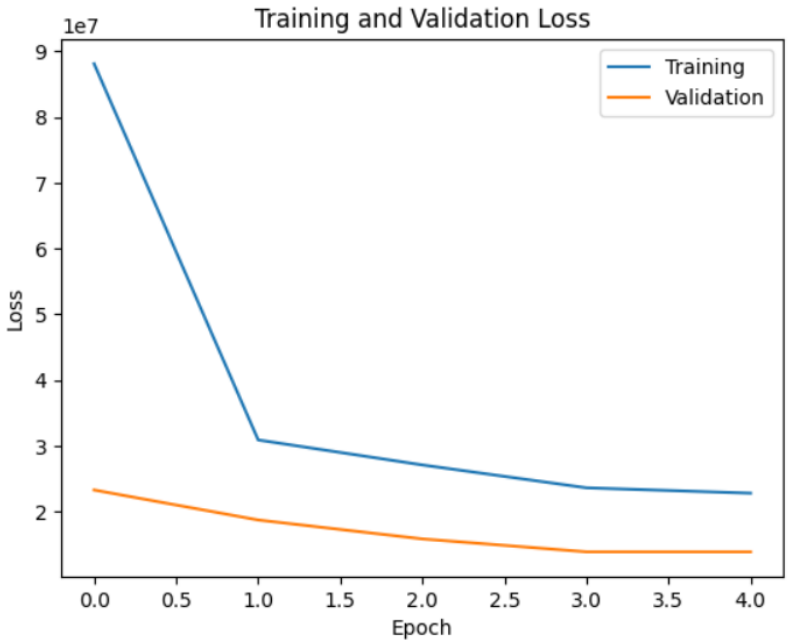



**Figure 7 : ANN graph with epoch value 5.**

## 1.e: The best model for predicting the price of a car.

The analysis's conclusion is that the model, which has the lowest MAE (332.56), the lowest MSE (514920.32) and with the lowest RMSE (717.57) is the Random Forest Regressor. Table 7 shows that the Random Forest Regressor model has the highest R2 score (0.99) among the models. A higher R2 score is indicative of a better match.

| **Metric/ Feature** | **Year of Manufacture (Linear)** | **Year of Manufacture (Polynomial = degree 4)** | **Regression Model (multiple numerical variables)** | **Random Forest Regressor model** | **ANN Model (Epoch = 10)** |
|---|---|---|---|---|---|
| **Mean Absolute Error** | 7058.19 | 5185.47 | 6167.88 | 332.56 | 1523.33 |
| **Mean Squared Error** | 125138115.69 | 97707742.96 | 84405262.35 | 514920.32 | 7835508.87 |
| **Root Mean Squared Error** | 11186.51 | 9884.72 | 9187.23 | 717.57 | 2799.19 |
| **R2 score** | 0.52 | 0.63 | 0.68 | 0.99 | 0.97 |

**Table 7 : Comparative analysis of all models.**

Figure 8 illustrates how well regression models function by comparing the model's predictions to the actual data visually.

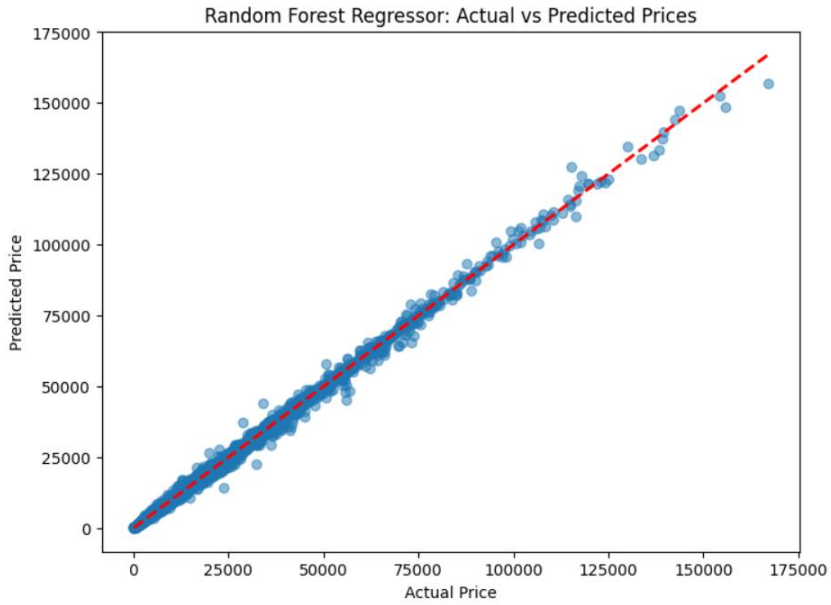



**Figure 8 : Random Forest Regression scatter plot.**

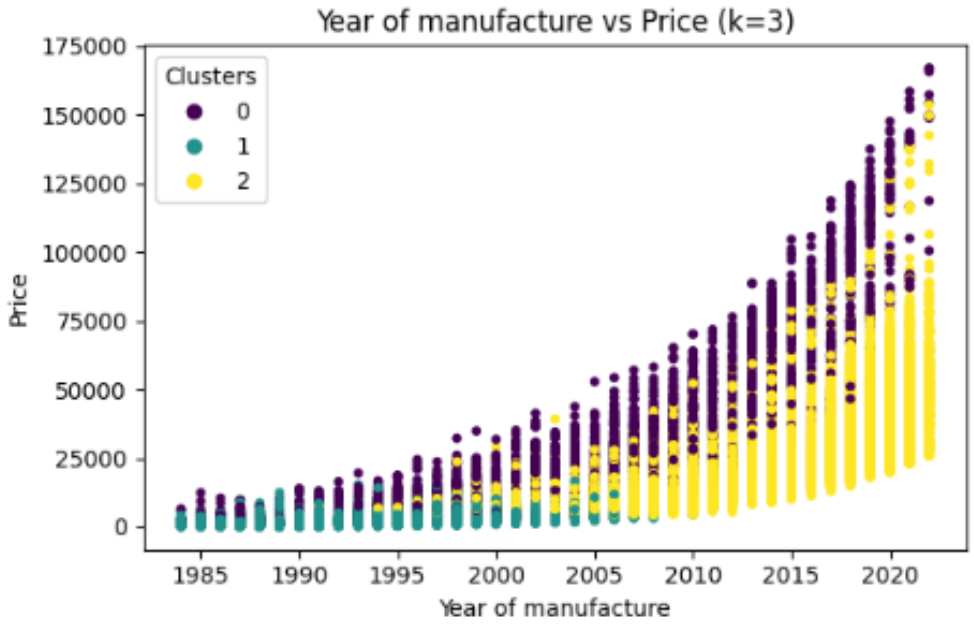
## 1.f: k-Means clustering algorithm to identify clusters in the car sales data.

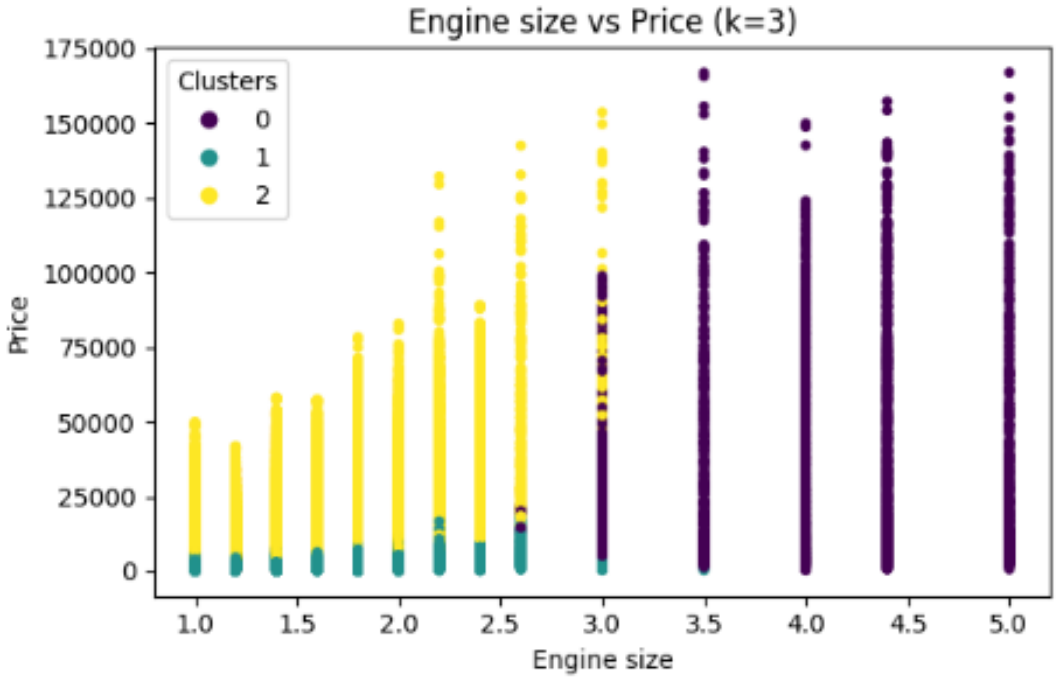
While calculating performance metrics as the goal, as shown in below Table 8. Of the given k values, k=3 has the highest Silhouette score (0.441), suggesting that it provides better separation between clusters has the best-defined clusters and the lowest Davies Bouldin index (0.801).

| Number of clusters (k) | Silhouette Score | Davies Bouldin Score |
|---|---|---|
| **K = 2** | 0.402 | 0.982 |
| **K = 3** | 0.441 | 0.801 |
| **K = 4** | 0.351 | 0.949 |
| **K = 5** | 0.288 | 1.079 |

**Table 8 : Silhouette and Davis - Bouldin Score.**

With a k value of 3, Figure 9 below illustrates the various facets of independent and dependent features.



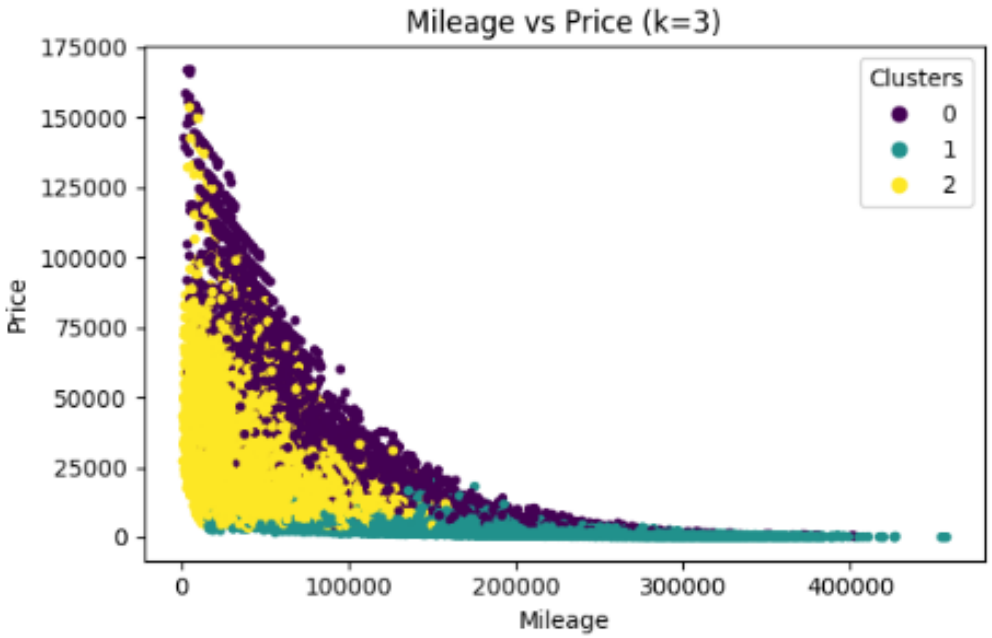


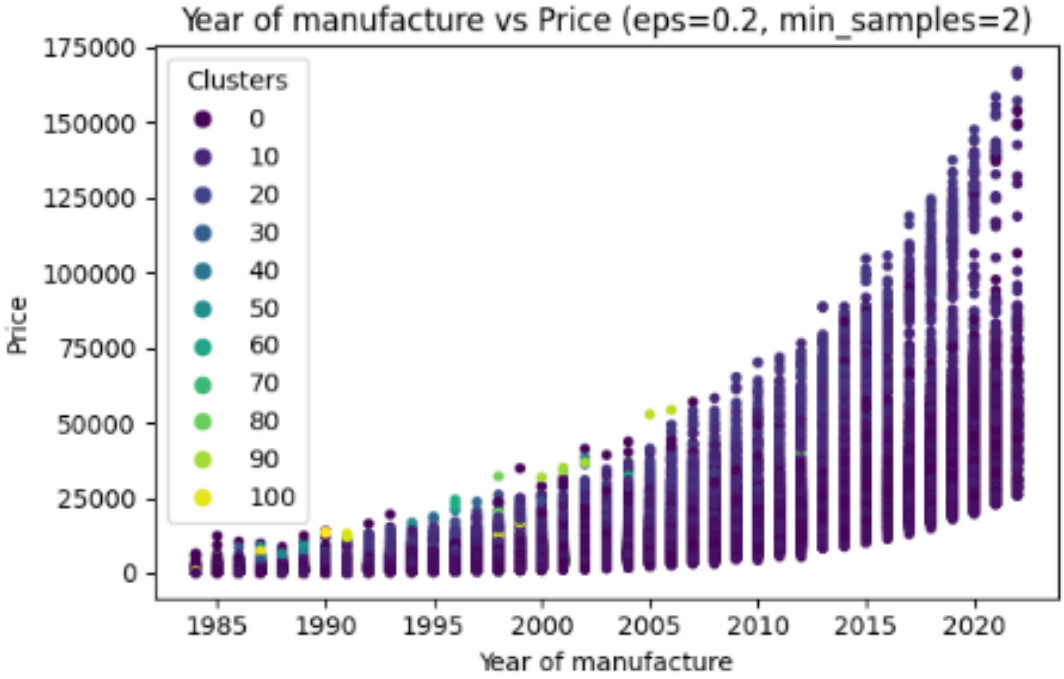

**Figure 9 : K-mean Clustering (k=3) for Car Sales Data.**

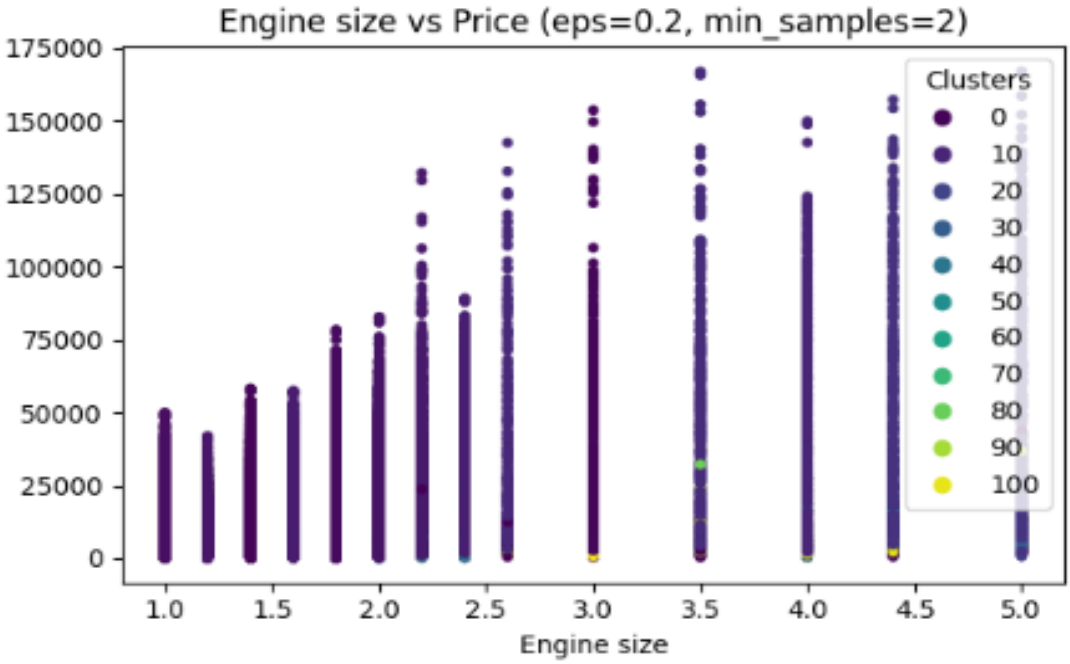
## 1.g: Compare k-Means clustering model with DBSCAN algorithm.

A clustering technique called DBSCAN, with epsilon values of 0.1, 0.2, 0.3, and min samples values of 2, 3, 4, is taken for evaluation. The clustering result with eps=0.2 and min_samples=2 seems to be the best option, as indicated in Table 9.

| epsilon values (eps) | min samples values | Silhouette Score | Davies Bouldin Score |
|---|---|---|---|
| 0.1 | 2 | -0.128 | 2.061 |
| 0.1 | 3 | -0.147 | 2.354 |
| 0.1 | 4 | -0.163 | 2.598 |
| 0.2 | 2 | -0.227 | 2.082 |
| 0.2 | 3 | -0.143 | 2.793 |
| 0.2 | 4 | -0.151 | 3.090 |
| 0.3 | 2 | 0.192 | 2.149 |
| 0.3 | 3 | 0.238 | 3.445 |
| 0.3 | 4 | 0.255 | 2.563 |

**Table 9: Silhouette and Davis - Bouldin Score for DBSCAN.**


Year of manufacture vs Price (eps=0.2, min_samples=2)
Clusters
0
10
20
30
40
50
60
70
80
90
100
Price
175000
150000
125000
100000
75000
50000
25000
0
1985
1990
1995
2000
2005
2010
2015
2020
Year of manufacture


Engine size vs Price (eps=0.2, min_samples=2)
Clusters
0
10
20
30
40
50
60
70
80
90
100
175000
150000
125000
100000
75000
50000
25000
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Engine size

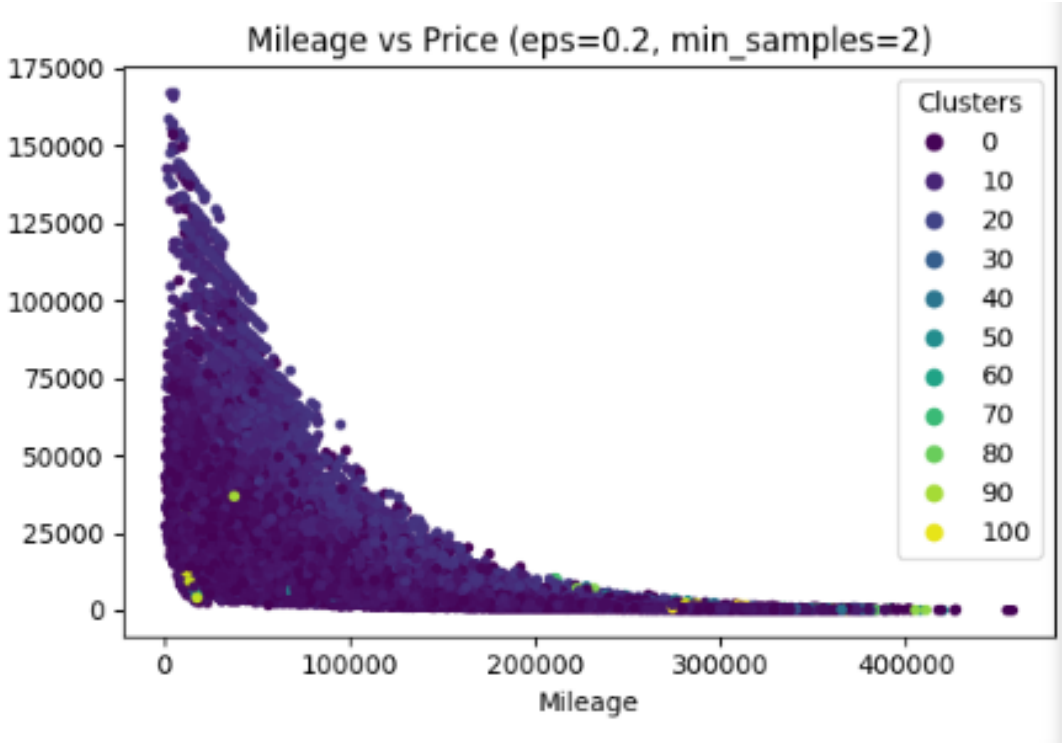



**Figure 10 :DBSCAN Clustering (eps = 0.2, min_value= 2) for Car Sales Data**

Table 8 and Table 9, can be compared to determine which option is preferable, with a higher Silhouette score and a lower Davies Bouldin index (k=3, Silhouette score=0.441, Davies Bouldin index=0.801) is best choice.

## Exercise 2: Image Recognition to Identify Handwritten Digits


## Abstract:

The usefulness of convolutional neural networks (CNNs) for classifying handwritten numbers is examined in this paper. To improve speed, dropout regularization techniques were applied to a simple model architecture. Model accuracy was further adjusted using hyperparameter adjustment, which focused on learning rate, batch size, and epochs (Shafi and Assad, 2023).


## Introduction:

A stimulating direction in deep learning research is the use of Convolutional Neural Networks (CNNs) for image recognition-based categorization of handwritten digits. The MNIST digit datasets were used to train the CNN, which then used complex patterns to correctly categorize digits. The design of recognition systems is becoming more and more dependent on machine learning approaches, particularly when applied to neural networks like CNNs(Jain and Sharma, 2021).

## 2.a: Architecture of the CNN model that you used (for example, the number and types of layers that you used, the activation functions that you used etc)

The following describes the architecture of the CNN model that is utilized in the given code snippet:

1. Input Layer:
    - Shape of the input layer is (28, 28, 1).
    - Description: An input layer with a single channel for grayscale images and a specified input shape for the images (28x28 pixels).

2. Convolutional Layer (Conv2D):
    - Number of filters: 32 - By selecting 32 filters, the model can learn 32 distinct feature maps, identifying a range of patterns

and characteristics seen in the input pictures. This figure achieves a compromise between the efficiency and model complexity.

- Kernel size: (3,3) - Convolutional layers often use a kernel size of (3,3) since it efficiently captures local spatial patterns without unduly raising the computational cost.
- Activation function: ReLU - This activation function is used because it is easy to use and efficient in introducing non-linearity, which enables the model to recognize intricate patterns and characteristics in the data.
- Description: (3,3) kernel size and 32 filters make up the convolutional layer. Applying ReLU activation creates non-linearity.

3. Max Pooling Layer (MaxPooling2D):
    - Pool size: (2,2) - By halving the spatial dimensions of the feature maps, a (2,2) pool size effectively downsamples the data and lowers computing complexity without sacrificing significant feature.
    - Description: To downsample the feature maps and lower computational cost, use a max pooling layer with a pool size of (2,2).

4. Flatten Layer:
    - Description: In order to prepare the data for input into the fully connected layers, flatten the layer and turn the 2D feature maps into a 1D vector.

5. Dense Layers:
    - Number of neurons: 128 (first dense layer), 10 (output layer) - Selecting 128 neurons gives the model enough power to understand complex correlations between features by learning high-level abstractions from the flattened feature maps.

- Activation function: ReLU (first dense layer), Softmax (output layer) - To add non-linearity and help the model learn intricate patterns, ReLU activation is employed in the first dense layer. In multi-class classification tasks, softmax activation is employed in the output layer to enable the model to provide class probabilities for every input image by converting raw scores into probabilities.
- Description: The output is flattened and then two dense layers are added. Ten neurons with softmax activation for multi-class classification (matching to the number of classes in the MNIST dataset) are found in the output layer, whereas 128 neurons with ReLU activation are found in the first dense layer.

All in all, these decisions are grounded in empirical data and standard procedures when developing CNN networks for image classification applications. In order to efficiently learn and generalize from the input data, they seek to achieve a balance between model complexity and performance by utilizing the expressive potential of convolutional and dense layers.

## 2.b: The regularisation methods that you used in your CNN model

Two regularization techniques are employed in the CNN model that is provided: dropout and explicit learning rate specification in the optimizer:

1. Dropout:
    - Usage: A 0.2 dropout rate is used after the first Dense layer.
    - Justification: In order to avoid overfitting in neural networks, dropout is a regularization technique that is frequently employed. During training, it removes (sets to zero) a random portion of the input units. This promotes the network to learn more resilient and generalizable characteristics and helps prevent the network from becoming overly dependent on any one neuron. Twenty percent of the neurons in the dense layer will randomly drop out during each training epoch when dropout is applied to the output of the first dense layer in this model, with a dropout rate of 0.2..

2. Explicit Learning Rate Specification:
   - Usage: A learning rate of 0.001 is used while instantiating the Adam optimizer.
   - Justification: Learning rate is a hyperparameter that regulates the size of the step used in optimization. For neural networks to be trained effectively, the right learning rate must be set. The Adam optimizer is applied in this model, with a learning rate of 0.001. During training, the adaptive learning rate optimization algorithm Adam automatically modifies the learning rate. We can fine-tune the learning rate based on the features of the dataset and the model architecture and have more control over the optimization process by explicitly defining the learning rate.

All things considered, these regularization techniques aid in reducing overfitting and enhance the CNN model's generalization capabilities. While explicit specification of the learning rate allows for more control over the optimization process and eventually results in a more robust and accurate model, dropout adds randomness during training. The CNN model's regularization techniques, Dropout and explicit learning rate specification, can affect the accuracy of the output in different ways.

I employed the Dropout layer as a regularization technique in the given Convolutional Neural Network (CNN) model to avoid overfitting. During training, dropout makes a random fraction rate of input units equal to zero at each update, which strengthens the model and enhances its performance with unknown data. Since a dropout rate of 0.2 is being employed in this instance, 20% of the input units will be arbitrarily dropped during training.

Depending on the dropout rate and model complexity, dropout can have a positive or negative impact on accuracy. The accuracy on the training set may initially decline as the dropout rate rises, but the accuracy on the validation set will rise instead, suggesting that the model is less overfitted and more capable of generalizing to new data. But an excessive dropout rate could prevent the model from learning enough features, which would result in underfitting and decreased accuracy. To discover the ideal value that strikes a compromise between bias and variation, it is crucial to fine-tune the dropout rate.

Test accuracy, a useful indicator of how well a model works with unknown data, shows that the application of dropout in this particular CNN model has improved the network's generalization. According to the test accuracy shown in Table 11, is 98.57%, demonstrating that the model has picked up pertinent traits and is capable of producing precise predictions when faced with new data.

| **Metric** | **CNN (No Regularizer with epoch = 5)** | **CNN (Dropout Regularizer & Learning Rate with epoch = 5)** |
|---|---|---|
| **Training Accuracy** | 0.9831 | 0.9796 |
| **Testing Accuracy** | 0.9855 | 0.9857 |

**Table 11: Performance metric of CNN with Regularization and no regularization.**

## 2.c: Hyperparameter tuning that undertook to optimise the model.

With a dropout regulation of 20%, a learning rate of 0.001, and an epoch of 5, as indicated Table 11, in this model may function very well, but since it is reaching 98% accuracy, additional tuning of the model is not necessary. However, a number of hyperparameters, including the number of convolutional layers, filters, activation functions, batch size, learning rate, number of epochs, and number of filters, can be adjusted to further optimize the system's performance.

## 2.d: Any evidence of overfitting in the models?

We may compare the accuracy and loss during training between the training and validation phases of the model to see if there was any indication of overfitting. The model may be overfitting to the training data if the training accuracy is significantly greater than the validation accuracy or if the training loss is significantly lower than the validation loss.

The training and validation accuracy plots in Figure 10 and Figure 11 of the present model demonstrate that, although the difference is not very great, the training

accuracy is marginally greater than the validation accuracy. This implies that there isn't a noticeable overfitting of the model to the training set.

Similarly, we can see that the training loss is marginally smaller than the validation loss from the training and validation loss plots displayed in Figure 10 and Figure 11. However, this difference is not very noticeable. This implies that there isn't a noticeable overfitting of the model to the training set.

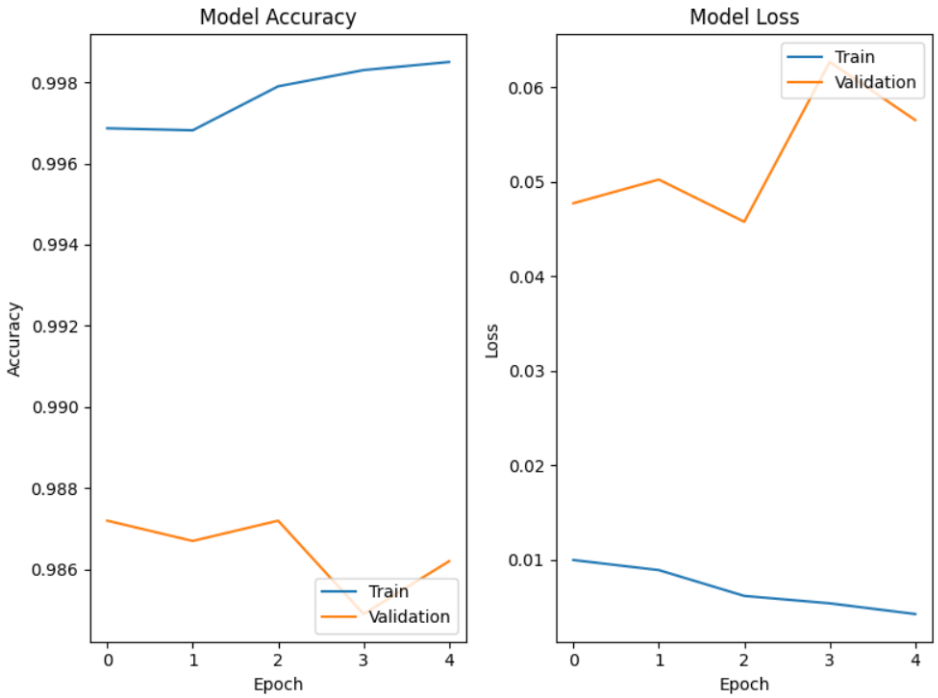



**Figure 10: Training and validation accuracy of CNN model.**

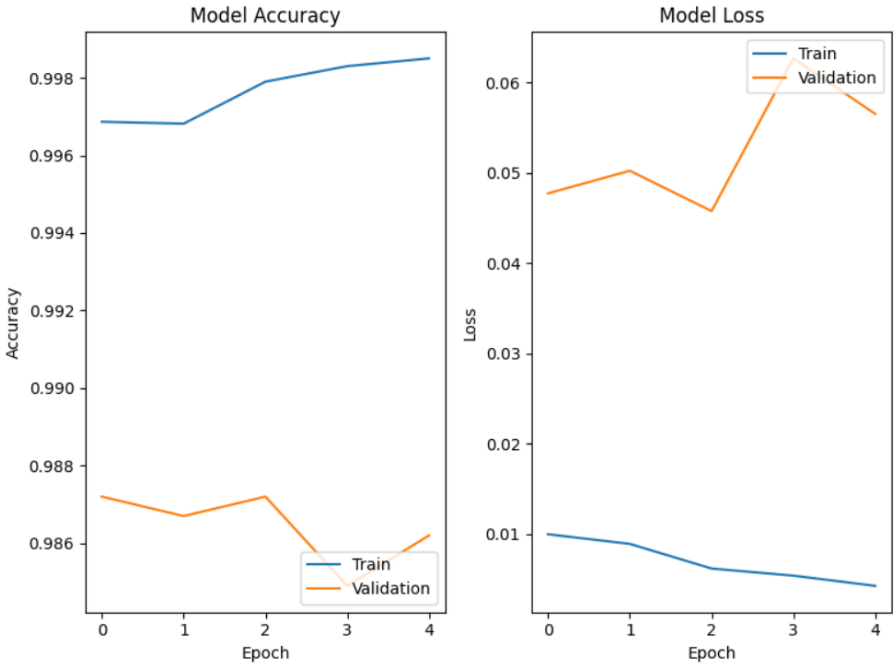



**Figure 11: Training and validation accuracy of CNN model with regularization.**

# Exercise 3: A Literature Review on Ethical Applications of AI


## Abstract:

In recent years, the rapid advancement of artificial intelligence (AI) technologies has sparked significant interest and debate surrounding its ethical implications and societal impact. As AI systems become increasingly integrated into various aspects of our lives, ranging from healthcare and finance to transportation and education, the need to ensure their ethical use and deployment has become paramount.


## Article 1: Introduction

In this paper, a regulatory sandbox is used for ECG classification in a healthcare case, where the authors aim to study fairness within AI, particularly bringing regulation and theoretical knowledge closer. (RANJBAR et al, 2023). Regulatory sandbox implemented on AI-enabled ECG classifier to evaluate fairness.

## Article 1: Key Conclusion

**Regulatory Sandbox**: Utilized to evaluate AI fairness and bias within the confines of existing regulations, like GDPR.

**Fairness & Bias**: Explores the dynamic nature of fairness in regulations and identifies four main sources of bias in healthcare AI.

**Countermeasures**: Suggests measures like data control and monitoring systems to mitigate bias and ensure regulatory compliance.

**Future Research**: Highlights the need for further research on metrics and outcomes for AI fairness in line with regulatory compliance.

## Article 2: Introduction

The aim of the paper is to provide a comprehensive overview of fairness and bias in AI, discussing their sources, impacts, and mitigation strategies. It

emphasizes the importance of interdisciplinary collaboration to develop fair and unbiased AI systems that are ethical and responsible. (Ferrara, 2023).

## Article 2: key Conclusion

**Sources of Bias**: Identifies data, algorithmic, and user biases as the main sources of unfair outcomes in AI systems.

**Societal Impact**: Highlights the reinforcement of existing inequalities and harmful stereotypes due to biased AI, affecting areas like healthcare and criminal justice.

**Mitigation Strategies**: Suggests methods like data augmentation and counterfactual fairness to address biases, emphasizing the need for diverse datasets and unbiased data collection.

**Ethical Considerations**: Underlines the importance of transparency and oversight in AI systems to address privacy concerns and ensure ethical use.

## Article 3: Introduction

Examining the ethical issues related to AI development and application, the study focuses on the difficulties in comprehending AI decision-making processes and their consequences for accountability and fairness (Huriye, 2023).

## Article 3: Key Conclusion

**Bias and Discrimination**: It draws attention to the possibility that AI systems will reinforce societal prejudices, which might result in unfair treatment and discrimination, especially with regard to facial recognition technology.

**Privacy Concerns**: The study highlights how crucial effective data handling is to preventing data breaches and privacy violations, particularly in light of AI's capacity to gather enormous volumes of personal data.

**Accountability and Transparency**: The study calls for improved transparency in AI decision-making processes to ensure accountability and address the challenges posed by complex and autonomous AI systems.

## Three successes in how AI can be used in an ethical manner:

Artificial Intelligence (AI) can be used ethically in various ways to benefit society while minimizing harm. Here are three examples of successful ethical applications of AI:

1. Healthcare Improvement:

AI has significantly contributed to advancements in healthcare by enhancing diagnostic accuracy, personalizing treatment plans, and managing healthcare resources more efficiently. For instance, AI algorithms can analyze complex medical data much faster than human capabilities, leading to quicker diagnoses of diseases such as cancer or heart disease.

2. Environmental Sustainability:

AI can play a crucial role in environmental conservation efforts by optimizing energy use, reducing waste, and monitoring deforestation and wildlife. AI technologies are used to predict weather patterns more accurately, which can improve the management of energy resources like wind and solar power, leading to more efficient use of renewable energy sources.

3. Education Enhancement:

In the educational sector, AI can personalize learning experiences according to individual student needs, adapting educational content and pacing accordingly. AI systems can analyze the performance and learning habits of students to tailor educational materials and identify areas where students might struggle, providing additional resources or alternative methods of instruction.

## Three gaps or challenges that are faced in the ethics of AI based on the Articles Reviewed

There are several gaps and challenges faced in the ethics of AI, including:

1. The Transparency Gap:

AI decisions are not always intelligible to humans, which can lead to a lack of trust in the technology and its outcomes. This transparency gap can be particularly problematic in high-stakes applications like healthcare, finance, or

law enforcement, where the lack of understanding can have severe consequences.

2. The Accountability Gap:

Establishing distinct lines of accountability is more important as AI systems grow more independent and make decisions that affect our daily lives. When an autonomous car malfunctions or when medical diagnoses are made and decisions concerning treatments or drugs are implemented, who should be held responsible? This also holds true for the judicial system, where ambiguity and confusion can result from the accountability gap.

3. The Responsibility Gap:

The term "responsibility gap" describes a concern that AI technologies could make it harder or impossible to hold people morally accountable for unfortunate events. This disparity might show up as disparities in guilt, public and moral accountability, and active responsibility, among other manifestations.

## Three suggestions to bridge these gaps

To bridge the gaps in AI ethics, several strategies can be employed:

1. Transparency and Explainability:

Implementing transparent and interpretable AI algorithms can help bridge the transparency gap. This can be achieved by using techniques such as model interpretability, feature attribution, and model-agnostic explanations.

2. Accountability and Governance:

Establishing clear lines of accountability and governance structures can help address the accountability gap. This can involve defining accountability goals, such as compliance, reporting, oversight, and enforcement, and ensuring that these goals are aligned with the sociotechnical structure of AI systems.

Meaningful Human Control and Professional Responsibility:

Facilitating significant human oversight of AI systems can aid in closing the accountability gap. Establishing normative standards for human control may

entail doing things like making sure AI systems are developed and implemented in a way that permits human monitoring and intervention.

## References:


Debener, J., Heinke, V. and Kriebel, J. (2023) 'Detecting insurance fraud using supervised and unsupervised machine learning', *Journal of Risk and Insurance*, 90(3), pp. 743–768. Available at: https://doi.org/10.1111/jori.12427.

Fairness And Bias in Artificial Intelligence: A Brief Survey of Sources, Impacts, And Mitigation Strategies (2023). arXiv (Cornell University). Available at: https://doi.org/10.48550/arxiv.2304.07683.

'Fairness in Artificial Intelligence: Regulatory Sanbox Evaluation of Bias Prevention for ECG Classification' (2023) in. *Medical Informatics Europe*, pp. 488–489. Available at: https://doi.org/10.3233/SHTI230184.

Garg, V.K. and Kalai, A. (2018) 'Supervising Unsupervised Learning'.

Jain, H. and Sharma, N. (2021) 'Handwritten Digit Recognition Using CNN', in D. Goyal et al. (eds) *Proceedings of the Second International Conference on Information Management and Machine Intelligence*. Singapore: Springer, pp. 631–637. Available at: https://doi.org/10.1007/978-981-15-9689-6_69.

Shafi, S. and Assad, A. (2023) 'Exploring the Relationship Between Learning Rate, Batch Size, and Epochs in Deep Learning: An Experimental Study', in M. Thakur et al. (eds) *Soft Computing for Problem Solving*. Singapore: Springer Nature, pp. 201–209. Available at: https://doi.org/10.1007/978-981-19-6525-8_16.

The Ethics of Artificial Intelligence: Examining the Ethical Considerations Surrounding the Development and Use of AI (2023). Available at: https://doi.org/10.58425/ajt.v2i1.142.