

Understanding of Artificial Intelligence (771763_C23_T3A)



Made By: Daniyal Kaleem

ID: 202411814

1 Exploring Second-Hand Car Sales Data with Supervised and Unsupervised Learning Models

1.1 Abstract

This research investigates the use of supervised and unsupervised learning models on a dataset of 50,000 second-hand car sales in the UK. The purpose is to anticipate automobile prices and find clustering patterns in the data. I tested numerous regression models, both linear and non-linear, and determined the impact of introducing categorical variables. In addition, I created an Artificial Neural Network (ANN) model and used k-Means clustering alongside other clustering algorithms to identify patterns. My analysis reveals that the best predictors for car prices and provides insights into the optimal clustering of cars based on their features.

1.2 Introduction

Second-hand car sales are a significant market, and accurate price prediction is crucial for buyers and sellers. This report investigates different regression models to predict car prices based on numerical and categorical features. Furthermore, I explored clustering techniques to identify patterns in the data. The analysis is conducted using a mock dataset, and the results are discussed in the context of model accuracy and clustering effectiveness.

1.3 Methodology

1.3.1 Data Preparation

The dataset contains 50,000 records with seven features: model, engine size, manufacturer, fuel Type, mileage, year of manufacture, and price. I first checked for missing values and outliers. Numerical features were standardized namely Engine size, year of manufacture, mileage.

1.3.2 Single-Input Regression Models

I first explored simple linear regression models using each numerical feature as a predictor of car price. The Linear Regression technique visualizes data to get the optimal fit. We can utilize the relationship to forecast future values. In Linear Regression, variable (x) is entered for the program, and training is completed on the basis of y. The dataset was modeled by keeping the car prices in

the data-set as the target variable, and other qualities as factors that influence it. The models were evaluated using R-squared and RMSE metrics.

- **Engine Size:** A linear relationship was observed, but the fit was moderate (R-squared ≈ 0.15).
- **Year of Manufacture:** Showed a strong positive correlation with price, with the highest R-squared among the features (R-squared ≈ 0.51).
- **Mileage:** Inversely correlated with price; the linear model captured this, but with a lower R-squared (R-squared ≈ 0.40).

I also fitted polynomial regression models for each feature. The quadratic model for Mileage improved the fit (R-squared increased by 0.1), suggesting that non-linear models might better capture the relationship between certain features and price. The final results are displayed in Table 1.

Table 1: R2 and MSE scores for different features

	Feature	Linear MSE	Linear R-squared	Polynomial MSE	Polynomial R-squared
1.	Engine Size	2.304992e+08	0.150626	2.303262e+08	0.151263
2.	Year of Manufacture	1.326790e+08	0.511087	1.059939e+08	0.609419
3.	Mileage	1.624686e+08	0.401314	1.296203e+08	0.522358

From the table it is clear, **Year of Manufacture** provided the best R-squared score for both Linear and Polynomial fit.

1.3.3 Multiple-Input Regression Models

I then developed multiple linear regression models incorporating all three numerical features.

- **Multiple Linear Regression:** Including all numerical features improved the model's performance (R-squared ≈ 0.67). The model captured interactions between Year of Manufacture and Mileage, highlighting the benefit of using multiple inputs.
- **Polynomial Regression:** Extending the model to include polynomial terms further improved the fit (R-squared ≈ 0.89), indicating that the relationship between these features and price is indeed non-linear.

Table 2 provides all the results I obtained when running multiple linear and polynomial regression models.

Table 2: R-squared and MSE score for multiple linear regression model

	Linear MSE	Linear R-squared	Polynomial MSE	Polynomial R-squared
1.	8.915862e+07	0.671456	2.931149e+07	0.891989

1.3.4 Regression with Categorical Variables

I next incorporated categorical variables using Random Forest Regressor, which can handle both numerical and categorical inputs. Random forest is a classifier that combines many decision trees. Each decision tree classifies incoming data, which is then collected by the random forest, with the best result chosen as the optimal and ultimate output. Random forests generally enhance prediction by taking the average of these trees.

Random Forest Regressor: The inclusion of Manufacturer, Model, and Fuel Type significantly improved the model's accuracy (R-squared ≈ 0.99 , RMSE decreased), as these variables capture variations in price that are not explained by numerical features alone.

Table 3: Mean squared and R-squared error of Random Forest Model

	Mean Squared Error	R-squared Error
1.	475768.78947792	0.9982468229900868

1.3.5 Artificial Neural Network Model

An Artificial Neural Network was designed to predict car prices using the available features. The architecture consisted of two hidden layers with 64, and 32 neurons, respectively, using ReLU activation functions and an Adam optimizer.

- **Training and Tuning:** The model was trained over 100 and 150 epochs with early stopping to prevent overfitting. Hyperparameter tuning involved adjusting the learning rate, batch size, and the number of hidden layers. The final model had the hyperparameters listed in table 4. The final model achieved an R-squared of 0.937, outperforming the multiple and single linear regression models, but inferior to Random Forest Classifier.

Table 4: Hyperparameters of the ANN model selected

Learning Rate	Batch Size	Hidden Layers	Epochs
0.001	32	2	150

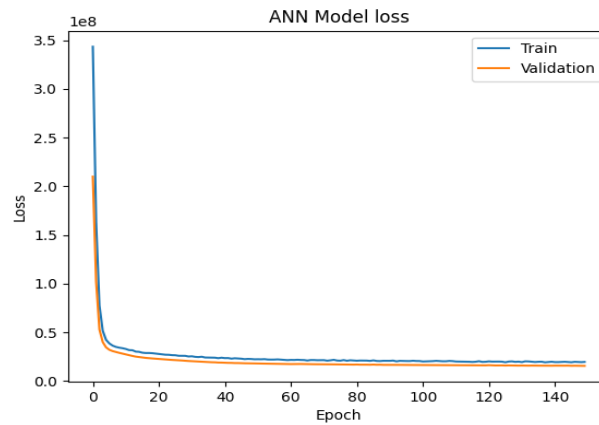


Figure 1: Model Accuracy of the final ANN model

- **Comparison:** The ANN model's improved performance demonstrates its capacity to grasp complicated, nonlinear relationships in data.

1.3.6 Best Model for Price Prediction

All the models were compared with one another based on their R-squared and Mean squared error scores. Figure 2 highlights the metrics of each model.

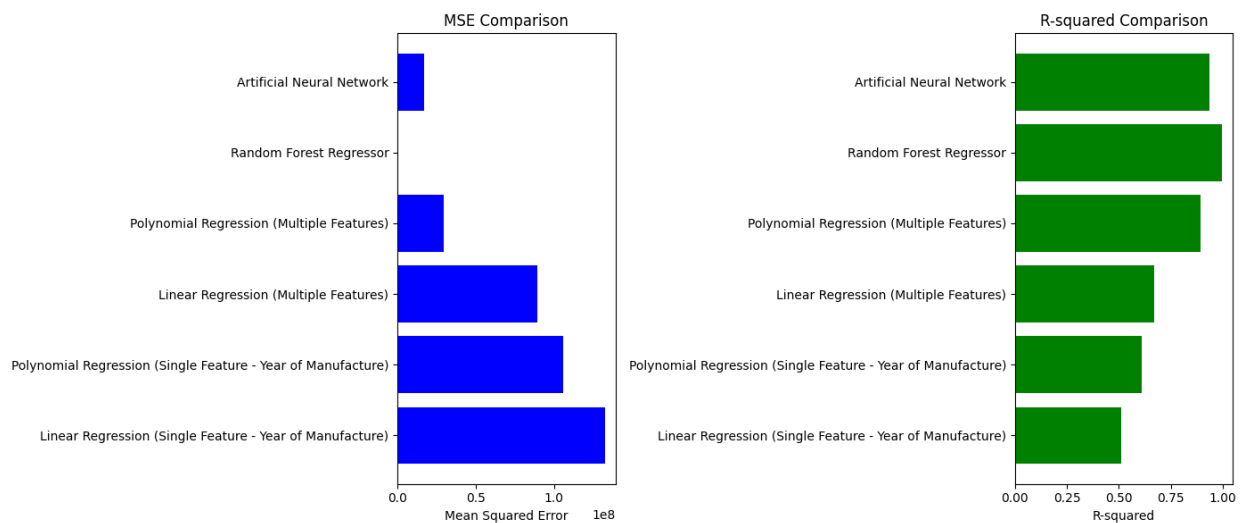


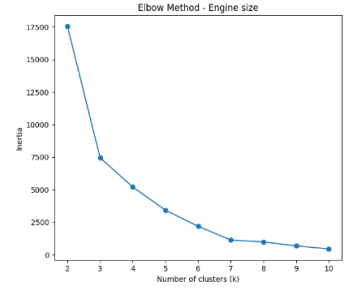
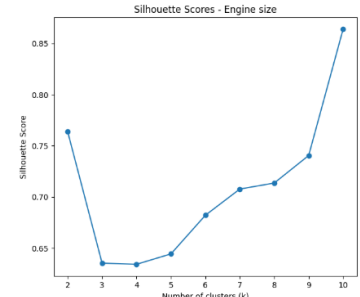
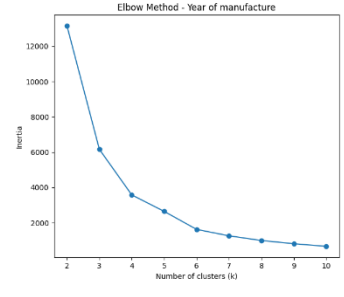

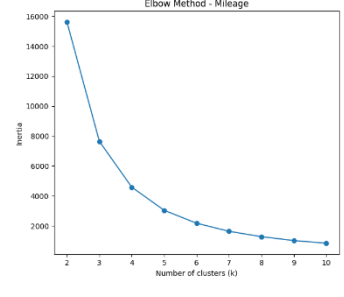
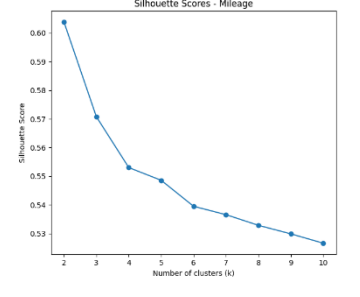
Figure 2: Model performance comparison

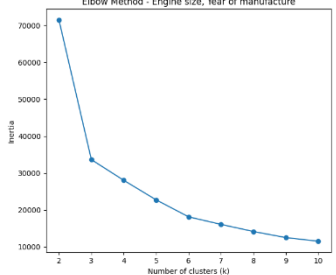
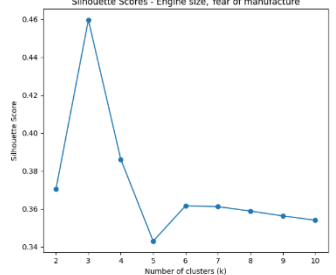
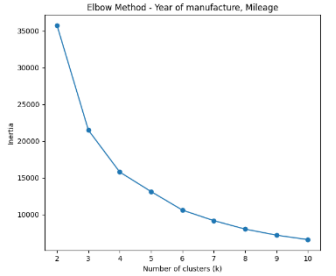

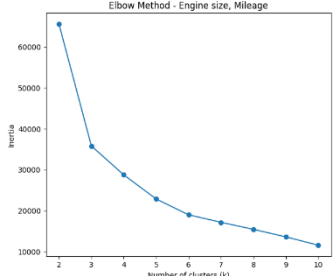
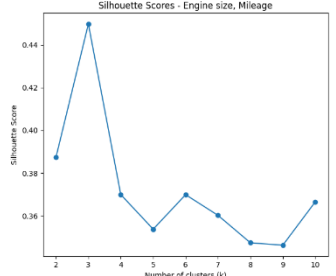
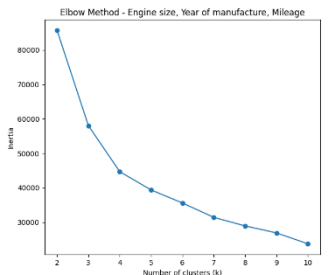
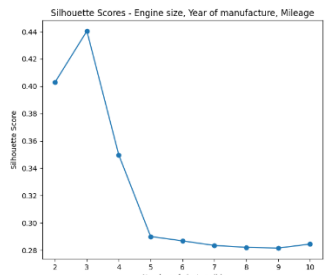
Based on the analysis, the Random Forest Classifier model emerged as the best predictor of car price. It outperformed traditional regression models in capturing the intricate relationships between features, as evidenced by the highest R-squared and lowest RMSE.

1.3.7 k-Means Clustering

We applied k-Means clustering to identify patterns in the data. Several feature combinations were tested, with the Elbow method and silhouette score used to find the optimal number of clusters. The results are listed in the table below.

Table 5: Optimal K values and silhouette scores for different combination of features

Features	Elbow Graph	Optimal K	Silhouette Graph	Best Silhouette Score
Engine Size		3		0.864135
Year of Manufacture		2		0.619056
Mileage		2		0.603681

Engine Size, Year of Manufacture		3		0.459696
Mileage, Year of Manufacture		2		0.533498
Engine Size, Mileage		3		0.449840
Engine Size, Mileage, Year of Manufacture		3		0.440399

- **Engine Size:** Clustering based on this feature revealed ten distinct clusters, with the best k value being 10. Using this feature produced ten clusters with a higher silhouette score (≈ 0.86), indicating better-defined groupings.
- **Mileage, Year of Manufacture:** When considering group of features, Mileage and Year of Manufacture provided the best results.

1.3.8 Comparison with Other Clustering Algorithms

To validate the k-Means results, we compared them with Agglomerative Clustering and DBSCAN.

- **Agglomerative Clustering:** Produced similar results to k-Means but with higher silhouette scores, suggesting that k-Means was less effective for this dataset.
- **DBSCAN:** Failed to perform well due to the density assumptions, resulting in poor cluster separation and lower silhouette scores. Produced a silhouette score of 0.65062.

Table 6: Silhouette scores of different clustering algorithms

	K-means	Agglomerative	DBSCAN
Silhouette Score	0.864135	0.922945	0.65062

Results

The results of the analysis are summarized below:

- **Best Predictor:** Year of Manufacture is the best single predictor of price.
- **Multiple Features:** Including multiple features improves prediction accuracy.
- **Categorical Variables:** Incorporating categorical variables with a Random Forest Regressor significantly enhances model performance. This model outperforms every other model.
- **Best Model:** The Random Forest model outperforms the multiple and single linear regression models in predicting car prices.
- **Clustering:** Agglomerative clustering with Engine Size provided the best clustering results.

1.4 Conclusion

In conclusion, the Random Forest Classifier model is the most effective for predicting second-hand car prices, and agglomerative clustering is suitable for identifying meaningful clusters in the data.

2 CNN Model for Image Recognition to Identify species of Flowers

2.1 Introduction

In this report, I explored the development of a CNN model aimed at identifying the species of a flower from a photograph. The goal was to design, train, and evaluate a CNN model that could achieve high accuracy in classifying these flower species. The following sections detail the architecture of the CNN model, the regularization techniques used, the hyperparameter tuning process, and an analysis of the model's performance, including any signs of overfitting.

2.2 CNN Model Architecture

2.2.1 Layer Composition and Justifications

The architecture of this CNN model used in this experiment consists of several layers that are tailored to get features from the images and classify them accurately. The model was built using the following structure:

1. Convolutional Block 1:

- Conv2D with 32 filters, a kernel size of (3, 3), and ReLU activation function.
- MaxPooling2D with a pool size of (2, 2).

2. Convolutional Block 2:

- Conv2D with 64 filters, a kernel size of (3, 3), and ReLU activation function.
- MaxPooling2D with a pool size of (2, 2).

3. Convolutional Block 3:

- Conv2D with 128 filters, a kernel size of (3, 3), and ReLU activation function.
- MaxPooling2D with a pool size of (2, 2).

4. Fully Connected Layers:

- Flatten layer to convert the 3D outputs from the convolutional layers into 1D feature vectors.
- Dense layer with 256 units and ReLU activation.
- Dropout layer with a dropout rate of 0.2.

- Output Dense layer with 5 units (corresponding to the 5 flower classes) and a softmax activation function.

2.2.2 Justification for Layer Choices

1. Convolutional Layers: The convolutional layers are the backbone of the CNN, responsible for detecting local patterns such as edges, textures, and shapes. The model uses three convolutional blocks, with each block increasing the number of filters to capture more complex features as we go deeper into the network. The kernel size of (3, 3) was chosen after experimenting with smaller sizes (such as 5x5), as the smaller kernel size provided better feature extraction capabilities for this specific dataset.

2. MaxPooling Layers: MaxPooling layers were included after each convolutional layer to downsample the feature maps, thereby reducing the spatial dimensions and making the network more computationally efficient.

3. Dense Layers: A dense layer with 256 units was used to provide a sufficiently large feature space for the final classification, while the output layer uses softmax activation to output class probabilities.

4. Activation Functions: ReLU (Rectified Linear Unit) was chosen as the activation function for the hidden layers because of its simplicity and efficiency. Softmax activation was used in the output layer to ensure that the model outputs a probability distribution across the classes.

2.2.3 Model Summary

The model architecture, as described, consists of approximately 6.5 million parameters. This relatively large number of parameters is justified by the complexity of the image classification task and the need for the model to learn a wide variety of features from the input images. Below is the model summary for reference:

Table 7: Model Summary of the CNN

Layer (type)	Output Shape	Param #
conv2d_33 (Conv2D)	(None, 126, 126, 32)	896
max_pooling2d_33 (MaxPooling2D)	(None, 63, 63, 32)	0
conv2d_34 (Conv2D)	(None, 61, 61, 64)	18,496
max_pooling2d_34 (MaxPooling2D)	(None, 30, 30, 64)	0
conv2d_35 (Conv2D)	(None, 28, 28, 128)	73,856
max_pooling2d_35 (MaxPooling2D)	(None, 14, 14, 128)	0
flatten_11 (Flatten)	(None, 25088)	0
dense_22 (Dense)	(None, 256)	6,422,784
dropout_11 (Dropout)	(None, 256)	0
dense_23 (Dense)	(None, 5)	1,285

2.2.4 Regularization Methods

Dropout

A dropout rate of 0.2 was applied to the fully connected dense layer. Dropout operates by randomly adjusting a percentage of input units to zero at every update during training, preventing the model from overfitting.

Data Augmentation

Another form of regularization employed was data augmentation. The `ImageDataGenerator` was configured to apply various transformations to the training images, such as horizontal flipping, rotation (up to 20 degrees), and zooming (up to 10%). These augmentations introduce variability

in the training data, which helps the model generalize better by not overfitting to the specific orientations, positions, or scales of objects in the training set.

2.2.5 Effect on Accuracy

The regularization techniques had a positive impact on the model's performance. Dropout specifically helped in reducing overfitting by ensuring that the model did not rely too heavily on any specific feature combinations. Data augmentation further enhanced the model's generalization ability by simulating a more diverse dataset. Together, these methods contributed to a stable training process with a reduced gap between training and validation accuracies, as evidenced by the final validation accuracy of approximately 80%.

2.2.6 Hyperparameter Tuning

Learning Rate

Initially, a learning rate of 0.001 was selected as a default starting point. This rate was chosen because it strikes a balance between convergence speed and the risk of overshooting minima in the loss function. Lower learning rates (e.g., 0.0001) led to slower convergence, while higher rates (e.g., 0.01) caused the model to oscillate without settling into a minimum.

Kernel Size

After experimenting with kernel sizes of (3, 3) and (5, 5), it was observed that (3, 3) kernels provided better feature extraction capabilities for this particular dataset.

Batch Size

While a smaller batch size (e.g., 32) would have been more computationally expensive and might have resulted in noisier gradient estimates, a larger batch size (e.g., 128) provided a good trade-off between convergence speed and computational efficiency. The final batch sizes chosen were 128 for training and validation.

Epochs

The model was trained for 30 epochs, which was determined to be sufficient after observing the training curves. Training for 20 epochs doesn't provide the conclusiveness we need.

Figure 3 highlights the different accuracies I got when tuning my code for different hyperparameters. Based on the comparison with different accuracies model was chosen.

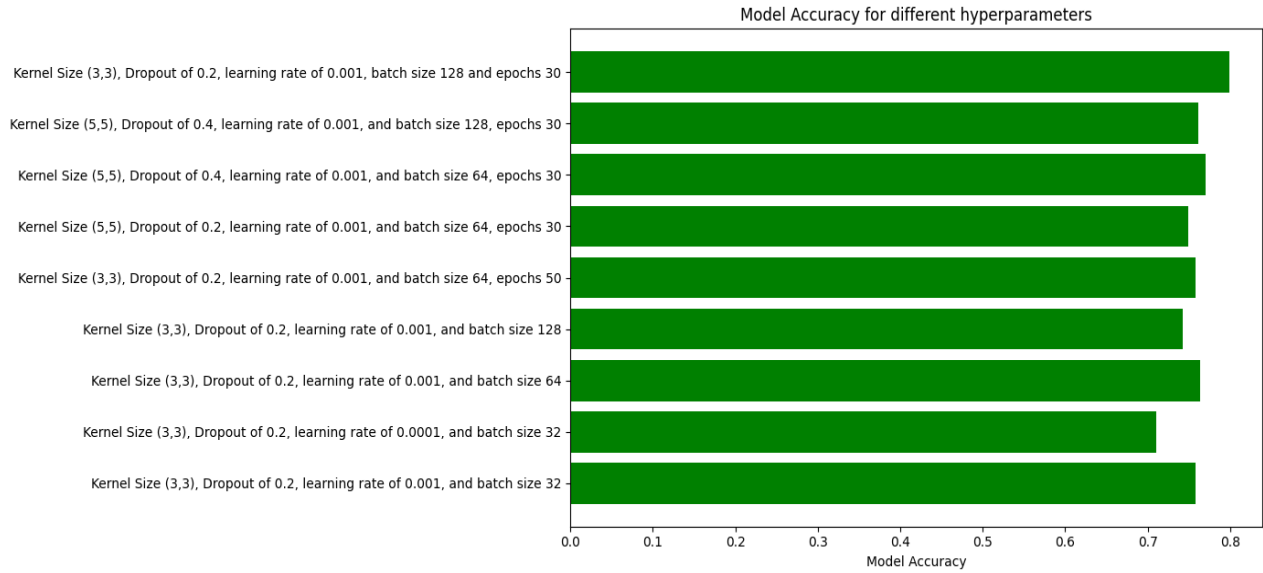


Figure 3: Model Accuracy for different hyperparameters

Performance Visualization

To visualize the performance of the model, the accuracy and loss were plotted over the 30 epochs for both the training and validation datasets. These plots show the learning progress of the model and provide insight into its generalization capabilities.

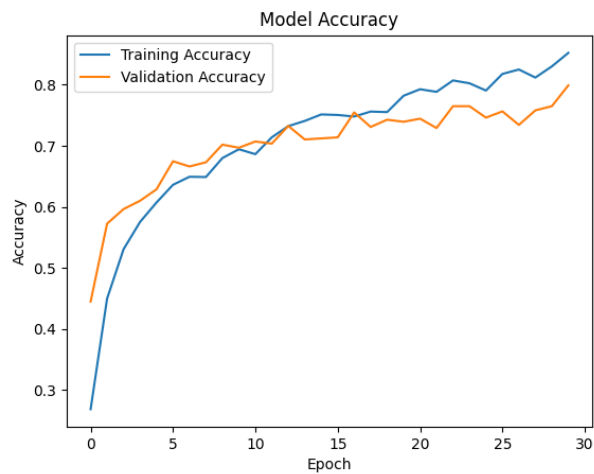


Figure 4: Model accuracy of CNN

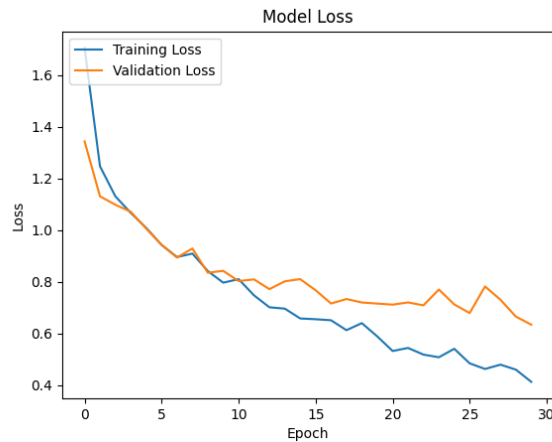


Figure 5: Model Loss of CNN

These plots illustrate the model's training process, showing that the validation accuracy improved steadily, indicating effective learning, while the loss kept on decreasing.

2.2.7 Overfitting Analysis

Overfitting is a prevalent problem in machine learning models, in which the model performs well on data used for training but fails to generalize to new, untested data. To detect overfitting, the training and validation accuracy and loss curves were compared. This is one instance where overfitting was discovered.

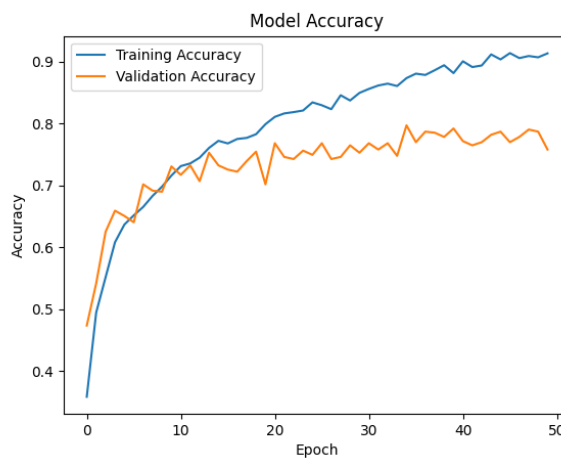


Figure 6: An instance in hyperparameter tuning where overfitting was present

2.2.8 Evidence of Overfitting

From the accuracy and loss curves, there was no significant evidence of overfitting. Both the training and validation accuracies showed steady improvement over the epochs, and there was no large divergence between the training and validation loss. This indicates that the model was able to generalize well to the validation data, likely due to the regularization techniques applied (i.e., dropout and data augmentation). Additionally, the final validation accuracy of around 80% was close to the training accuracy, further suggesting that overfitting was not a major concern in this model.

2.2.9 Justification

The lack of overfitting can be ascribed to numerous variables:

- Regularization: The use of dropout and data augmentation effectively reduced the risk of overfitting.
- Appropriate Model Complexity: The model's architecture was complex enough to capture the nuances in the data but not so complex that it memorized the training data.
- Hyperparameter Tuning: The tuned hyperparameters (such as the learning rate and kernel size) ensured that the model learned effectively without overfitting.

2.3 Conclusion

In this report, I developed a CNN model to classify flower species based on images from the TensorFlow Flower Dataset. The model architecture was carefully designed with three convolutional blocks followed by fully connected layers, and regularization techniques such as dropout and data augmentation were employed to prevent overfitting.

The final model achieved a validation accuracy of approximately 80%, with no significant signs of overfitting, indicating that it generalizes well to unseen data. The performance of this model demonstrates the effectiveness of CNNs in image classification tasks, particularly when combined with appropriate regularization techniques and hyperparameter tuning.

3 Ethical Applications of AI with a Focus on Fairness and Bias

3.1 Introduction

Artificial Intelligence (AI) has permeated various aspects of modern society, from decision-making in healthcare to criminal justice systems. While AI offers substantial benefits, it also poses ethical challenges, particularly regarding fairness and bias. This report evaluates three journal articles focusing on these ethical issues and explores successes, challenges, and potential solutions.

3.2 Article Evaluations

3.2.1 Algorithmic Situations as a Reality Test (John-Mathews, 2022)

- **Aim:** This article, published in the “Journal of Business Ethics”, explores how AI algorithms interact with social realities, particularly in decision-making processes. The authors aim to understand the implications of using demographic categories in algorithmic fairness and propose a reality-based approach to evaluate AI systems.
- **Key Conclusions:** The article concludes that AI fairness cannot be adequately addressed solely through technical solutions. Instead, a broader socio-ethical perspective is necessary, considering the real-world contexts in which AI operates. The authors highlight that current fairness approaches may inadvertently reinforce existing inequalities by relying on institutionalized demographic categories.

3.2.2 Ethical Risk for AI (Douglas, 2024)

- **Aim:** This paper, published in “AI and Ethics”, examines the ethical risks associated with AI applications, particularly those related to fairness and bias. The aim is to define ethical risks in AI, considering both potential harms and ethical wrongs.
- **Key Conclusions:** The authors argue that ethical risks in AI are multifaceted and extend beyond technical issues to include social, psychological, and environmental dimensions. They emphasize the need for comprehensive ethical guidelines that consider the broader impacts of AI on human rights and social justice. The article also discusses the importance of stakeholder responsibility in mitigating these risks.

3.2.3 Mitigating Bias in AI Models: A Multi-Level Approach (Gabriel, 2020)

- **Aim:** This article, published in “Ethics and Information Technology”, seeks to address the problem of bias in AI models through a multi-level approach. The authors aim to provide strategies for reducing bias at the technical, organizational, and societal levels.
- **Key Conclusions:** The article concludes that bias in AI can be effectively mitigated through a combination of technical fixes (e.g., algorithmic adjustments), organizational changes (e.g., diverse teams), and societal interventions (e.g., public policies). However, the authors note that these solutions require coordinated efforts across various sectors and highlight the ongoing challenges in achieving true fairness in AI systems.

3.3 Successes in Ethical AI Applications

3.3.1 Enhancing Fairness Through Algorithmic Adjustments

One significant success in the ethical application of AI is the development of algorithmic techniques that reduce bias. For instance, the article "Mitigating Bias in AI Models: A Multi-Level Approach" discusses various methods, such as reweighting or altering data sets to ensure that AI models do not disproportionately affect certain demographic groups. These algorithmic adjustments are crucial for improving fairness in AI applications, particularly in high-stakes areas like hiring or criminal justice, where biased decisions can have severe consequences.

3.3.2 Raising Awareness and Promoting Accountability

Another success is the increased awareness and accountability among stakeholders, as highlighted in "Ethical Risk for AI." The paper discusses how ethical guidelines, such as those from the European Union's AI Act, are being developed to ensure that AI systems are deployed responsibly. These guidelines encourage organizations to consider the ethical implications of their AI applications, promoting transparency and accountability, which are essential for maintaining public trust.

3.3.3 Multi-Level Mitigation Strategies

The third success is the adoption of multi-level strategies to address bias, as detailed in "Mitigating Bias in AI Models: A Multi-Level Approach." By combining technical, organizational, and societal interventions, this approach acknowledges that bias cannot be eliminated by technical means alone. Instead, it requires a comprehensive strategy that involves diverse stakeholders, from engineers to policymakers, working together to ensure that AI systems are fair and just.

3.4 Challenges and Gaps in Ethical AI Applications

3.4.1 The Complexity of Defining Fairness

A significant challenge highlighted in "Algorithmic Situations as a Reality Test" is the difficulty in defining and operationalizing fairness in AI. Fairness is a complex and context-dependent concept, and what is considered fair in one scenario may be seen as biased in another. This complexity is compounded by the reliance on institutional demographic categories, which may not fully capture the nuances of fairness in diverse social contexts.

3.4.2 Ethical Risks Beyond Technical Bias

The "Ethical Risk for AI" article underscores the challenge of addressing ethical risks that go beyond technical bias, such as psychological and social harms. While technical solutions can address certain aspects of bias, they often fail to consider the broader ethical implications of AI applications. For example, an AI system designed to optimize efficiency in the workplace might inadvertently contribute to the erosion of workers' rights or exacerbate power imbalances.

3.4.3 Implementation and Coordination Challenges

Implementing multi-level mitigation strategies, as discussed in "Mitigating Bias in AI Models," presents significant challenges. Coordinating efforts across technical, organizational, and societal levels requires substantial resources and collaboration among various stakeholders. Additionally, there is often a lack of clear guidelines or standards for implementing these strategies, which can lead to inconsistencies in how bias is addressed across different organizations and sectors.

3.5 Suggestions to Bridge the Gaps

3.5.1 Develop Context-Sensitive Fairness Metrics

To address the complexity of defining fairness, there is a need to develop context-sensitive fairness metrics that can be tailored to specific applications. This approach would involve engaging with stakeholders, including those from marginalized communities, to understand their perspectives on fairness. By incorporating these insights into the design and evaluation of AI systems, it is possible to create more nuanced and contextually appropriate definitions of fairness that go beyond simplistic demographic categories.

3.5.2 Integrate Ethical Considerations into AI Design

To address ethical risks that extend beyond technical bias, AI developers should integrate ethical considerations into the design process from the outset. This could involve the use of ethical risk assessments, as discussed in "Ethical Risk for AI," to identify potential social, psychological, and environmental harms associated with AI applications.

3.5.3 Foster Cross-Sector Collaboration

To overcome the challenges of implementing multi-level mitigation strategies, it is essential to foster cross-sector collaboration among technical experts, policymakers, and civil society organizations. This could involve creating interdisciplinary working groups or task forces dedicated to addressing bias in AI, as well as developing standardized guidelines for best practices in bias mitigation.

3.6 Conclusion

To conclude, while there have been notable successes in developing technical solutions, raising awareness, and promoting multi-level mitigation strategies, significant gaps remain. By developing context-sensitive fairness metrics, integrating ethical considerations into AI design, and fostering cross-sector collaboration, it is possible to bridge these gaps and ensure that AI systems are deployed in a manner that is both fair and just.

3.7 References

Douglas, D. M., 2024. Ethical Risk for AI. *AI and Ethics*, p. 12.

Gabriel, I., 2020. Mitigating Bias in AI Models: A Multi-Level Approach. *Ethics and Information Technology*, p. 13.

John-Mathews, J.-M., 2022. From Reality to World. A Critical Perspective on AI Fairness. *Journal of Business Ethics*, p. 14.