Lisbon School of Economics & Management
Universidade de Lisboa

MASTER
FINANCE

MASTER´S FINAL WORK
DISSERTATION

A Hybrid Approach to Option Pricing Integrating Unsupervised and Supervised Machine Learning Models

---

GABRIELE LOGGIA

JULY-2024

Lisbon School of Economics & Management
Universidade de Lisboa

MASTER
FINANCE

MASTER´S FINAL WORK
DISSERTATION

A Hybrid Approach to Option Pricing Integrating Unsupervised and Supervised Machine Learning Models

---

GABRIELE LOGGIA

SUPERVISION:
JOÃO AFONSO BASTOS

JULY-2024

To my Parents and my
Brother, for being light even
in the darkest night

GLOSSARY

- Adam – Adaptive Moment Estimation
- ANN – Artificial Neural Network
- BSM – Black-Scholes-Merton
- CBOE – Chicago Board Options Exchange
- DNN – Deep Neural Network
- DTE – Days to Expiration
- FNN – Feedforward Neural Network
- HV – Historical Volatility
- IV – Implied Volatility
- K – Exercise Price or Strike Price
- MAE – Mean Absolute Error
- MAPE – Mean Absolute Percentage Error
- ML – Machine Learning
- MSE – Mean Squared Error
- NN – Neural Network
- RF – Random Forest
- RMSE – Root Mean Squared Error
- S – Current Price of the Underlying Asset
- T – Maturity Date or Expiration Date
- UMAP – Uniform Manifold Approximation and Projection

ABSTRACT

This thesis explores a novel approach to option pricing by integrating unsupervised and supervised machine learning models. The objective is to assess whether these models can outperform the traditional Black-Scholes-Merton model and to investigate the effect of clustering on prediction accuracy. The analysis uses data from the Ivy DB US database, focusing on S&P 500 options traded on the CBOE from December 30, 2019, to December 30, 2022. The methodology involves applying K-means clustering to segment the dataset, followed by training Random Forest and Deep Neural Network models on these clusters. The models' performances are then compared against non-clustered machine learning models and the BSM model. The results show that the hybrid machine learning approach improves option pricing accuracy. Specifically, the Deep Neural Network models achieve a median improvement of 39.1%, while the Random Forest models show a median improvement of 5.2%. This suggests the potential of integrating advanced clustering techniques in financial modeling for more precise option pricing.

KEYWORDS: Option Pricing, Machine Learning, K-Means Clustering, Random Forest, Deep Neural Network

JEL CODES: C45; C53; G1; G12; G17; G23

RESUMO

Esta tese explora uma abordagem inovadora para a precificação de opções, integrando modelos de *machine learnig* supervisionados e não supervisionados. O objetivo é avaliar se esses modelos podem superar o modelo tradicional de Black-Scholes-Merton e investigar o efeito da clusterização na precisão das previsões. A análise utiliza dados do banco de dados Ivy DB US, focando-se em opções do S&P 500 negociadas na CBOE de 30 de dezembro de 2019 a 30 de dezembro de 2022. A metodologia envolve a aplicação da clusterização K-means para segmentar o conjunto de dados, seguida pelo treinamento de modelos Random Forest e Deep Neural Network nesses *clusters*. O desempenho dos modelos é então comparado com os modelos de *machine learnig* não clusterizados e com o modelo BSM. Os resultados mostram que a abordagem híbrida *machine learnig* melhora a precisão da precificação de opções. Especificamente, os modelos de Deep Neural Network alcançam uma melhoria média de 39,1%, enquanto os modelos Random Forest mostram uma melhoria média de 5,2%. Isso sugere o potencial de integrar técnicas avançadas de clusterização na modelagem financeira para uma precificação de opções mais precisa.

KEYWORDS: Option Pricing, Machine Learning, K-Means Clustering, Random Forest, Deep Neural Network

JEL CODES: C45; C53; G1; G12; G17; G23

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

ACKNOWLEDGMENTS

This research brings to a close a two-year journey, which although personal would not have been possible without the help of those close to me. I take this opportunity to thank:

My parents, Giovanni e Liliana, for supporting me and giving me the opportunity to face a difficult path with the right spirit.

My brother, Michele, source of inspiration and embodiment of the family's intrinsic willpower.

All my family for their trust and encouragement.

My mentor, Paolo, for sharing his experience and knowledge with me.

My supervisor, Professor João Afonso Bastos, for helping me with the analysis of the work and for the interesting insights that made this thesis possible.

Andreia for her support, patience and brilliant conversations.

The 'Italians' and my international friends, for having found in them mates who will go beyond this experience.

All my friends around the world who despite the distance have always found a way to show their closeness.

Last, I would like to thank myself for leading this journey with courage and perseverance.

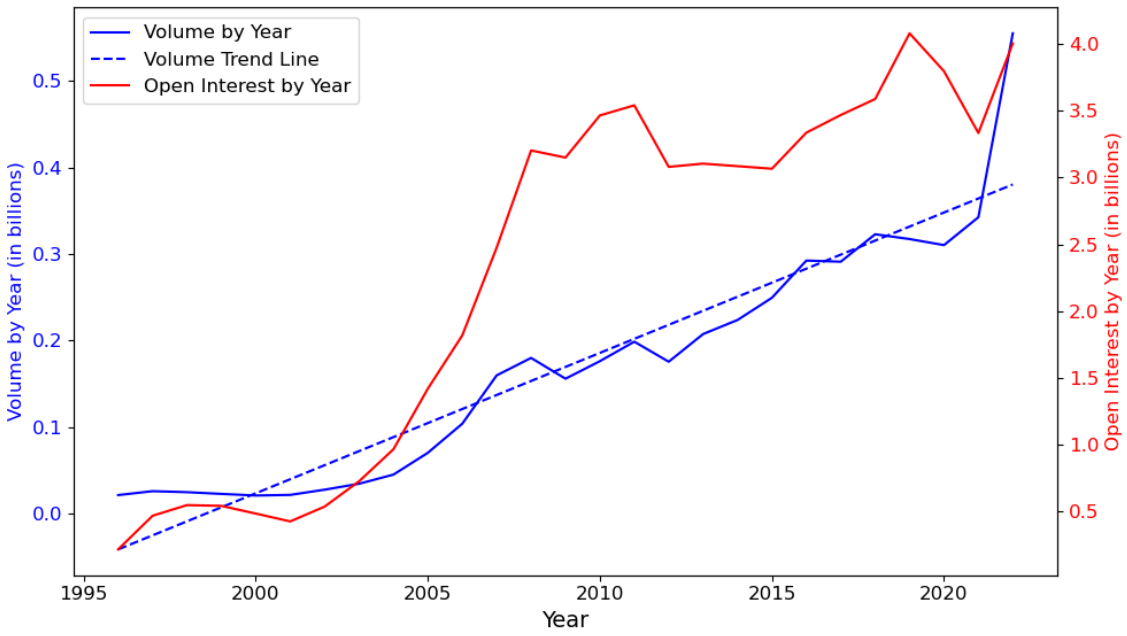
1. INTRODUCTION

The derivatives market is enormous, far larger than the stock market when measured in terms of underlying assets. The total value of assets associated with outstanding derivatives transactions vastly exceeds the global gross domestic product by several multiples (Hull, 2017).

A derivative is a financial instrument whose value depends (is derived) on the value of another variable called "underlying." Often, the underlying variables for derivatives are the prices of traded assets (e.g., options on stocks). However, derivatives could be based on almost any variable like the amount of rainfall during the summer in Portugal.

Options have become a mainstay in the derivatives market, experiencing steady growth in trading volume over the past few decades. Below we can see the growth in time of Total Volume and Open Interest by Year for Options that have the S&P 500 as underlying asset and that are traded in the Chicago Board Options Exchange (CBOE).

Figure I S&P 500 Options: Total Volume and Open Interest by Year



An option is a derivative financial instrument that gives the holder the right, but not the obligation, to buy or sell an underlying asset, (such as bonds, stocks, indices,

commodities, etc.) at a predetermined price, known as exercise price or strike price, by a specific date referred to as the maturity days.

Options are primarily classified into two categories: Plain Vanilla and Exotic. Plain Vanilla options are standard options, while exotic options are instruments with more complex features, often used to obtain specific tailor-made solutions. Plain Vanilla options are further divided into two main categories: Call and Put.

Call options give the possibility, but not the obligation, to purchase the right to buy the underlying S at the exercise price K by the expiration T. The maximum profit at expiration for a long call will be given by max $(S - K, 0)$, while for a short call it will be equal to min $(K - S, 0)$.

Put options give the possibility, but not the obligation, to purchase the right to sell the underlying S at the exercise price K by the expiration T. The maximum profit at expiration for a long put will be given by max $(K - S, 0)$, while for a short put it will be equal to min $(S - K, 0)$.

Besides the distinction between call and put, plain vanilla options also differ by the type of exercise:

• European: the buyer has the right to exercise the option only at expiration.

• American: the buyer has the right to exercise the option at any time within the expiration.

A curious anecdote told by Aristotle in the book "*The Politics Book I"* recounts what may perhaps be defined as the first use of an option. It is told that the philosopher Thales of Miletus, to demonstrate the utility of knowledge, predicted that there would be an extraordinary olive crop, and in the hearth of winter invested what little he had to secure the rights to all the olive presses in Miletus and Chios at a very low price, since no other bidder stepped forward. When the olive season came, many farmers were in search of the presses, and he was able to rent them out at the price he wanted. As we can see, the use of "options" dates back several centuries. However, despite their long history, determining the fair value of an option remains a complex question.

Notable efforts to answer this question include the work of Black and Scholes, (1973), where in an article titled *The Pricing of Options and Corporate Liabilities*, published in

the *Journal of Political Economy*, proposed a closed solution for the valuation of European options, later in the same year Merton, (1973) in *Theory of Rational Option Pricing* published in *The Bell Journal of Economics and Management Science* a modified version that gave us what we call today the famous Black-Scholes-Merton (BSM) model.

The objective of this thesis is to develop and evaluate a novel approach for option pricing using a combination of unsupervised and supervised non-parametric machine learning models. Specifically, we will employ K-means clustering to segment the datasets, followed by training Random Forest (RF) and Deep Neural Network (DNN) models on the resulting clusters. This hybrid approach aims to determine whether these machine learning algorithms can outperform the Black-Scholes-Merton (BSM) model and to study whether the inclusion of unsupervised clustering can improve the predictive accuracy of these models. Comparing the clustered machine learning and non-clustered models, and the BSM model could provide insights into the efficacy of these new methods and their potential to enhance option pricing accuracy in the evolving financial landscape.

This dissertation is structured as follows. Chapter 2 reviews the existing literature on the subject, identifying and synthesizing previous research. Chapter 3 explains the BSM model and explores the three machine learning models: K-means clustering, RF, and DNN. Chapter 4 introduces the database and explains its processing for usability by our models. Finally, Chapter 5 synthesizes the results, providing a clear comparison of the models' performances.

2. LITERATURE REVIEW

2.1. Parametric Models

A critical understanding of the Black-Scholes model's assumptions is crucial before delving into related research. This initial analysis will help us identify potential limitations and how researchers tried to develop new extensions and new formulations over the years that seek to incorporate the greater complexity of real world.

The assumptions are the no-arbitrage conditions, meaning that the market already reflect the fair price of the option and it would not be possible to achieve risk-free profits through a trading strategy. Perfectly liquid markets with the absence of transaction costs

were assumed, as well as constant short-term interest rate. The underlying asset price following a random walk with a constant expected return rate and volatility and without any dividend payment.

Previously, we mentioned that Merton, (1973) modified the original Black-Scholes formula including the possibility of dividend payments, among other alternative approaches for options pricing we can find: Local Volatility Models like Schroder, (1989) allowed volatility to vary based on the underlying asset price, better capturing the observed volatility smile. Stochastic Volatility Models like Hull and White, (1987) and Heston, (1993) where volatility itself becomes a random variable, reflecting changing market conditions. Statistical Series Expansion Models, such as Corrado and Su, (1996) which used mathematical series to approximate option prices, offering more flexibility but with greater computational complexity. Models with Jumps like Bates Model, (1996): an integration of the Merton Jump-Diffusion Model with the Heston model of stochastic volatility. This combination allows it to account for both sudden jumps in asset prices and the random fluctuation in volatility,

2.2 Non-Parametric Models

Despite the power of the most recent parametric models briefly described in the previous paragraph, they still rely on some certain economic and statistical assumptions, like no-arbitrage, and market completeness. Furthermore, these models do not offer a closed-form solution for the valuation equation, necessitating an optimization process that can be computationally costly.

To overcome these problems and have more flexibility, non-parametric models using Machine Learning (ML) have been developed. Starting from the study of Hutchinson et al., (1994) where is assessed the potential value of network pricing formulas by simulating BSM option prices and demonstrating that learning networks can effectively approximate these formulas. Other studies related to option pricing using non-parametric method are:

Amilon, (2003) that applied Neural Network (NN) on the Swedish Stock Index call options. Gradojevic et al., (2009) where it has been shown that clustering by time to maturity and moneyness can improve the performance of a Modular Neural Network. More recent contributions include the works of Gaspar et al., (2020) where it has been created a NN that receive as an input also the average put options price per company of the training set, to make the learning process for pricing American put options faster. Ivașcu, (2021) that tested several ML algorithm to price European call options who have as underlying asset the WTI crude oil future contracts. Bastos, (2024) where using conformal quantile regression applied to gradient boosting machines shows that is possible to quantify uncertainty in option price predictions.

3. MODELS DESCRIPTION

3.1 The Black-Scholes-Merton Model

As we previously introduced, Black and Scholes, (1973) provided a closed-form solution to the problem of pricing European options using the parabolic partial differential equation (PDE) presented below:

$$\frac{\partial f}{\partial t} + \frac{1}{2}\,\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} + rS\,\frac{\partial f}{\partial S} - rf = 0\,, \tag{1}$$

where:

- S represents the current price of the underlying asset;
- t denotes the time;
- σ measures the volatility of the underlying asset's price;
- r risk-free rate;

Remembering that model operates under several key assumptions:

- There are no dividends paid during the lifetime of the derivative.
- Trading of securities is continuous.
- The risk-free interest rate is constant and uniform for all durations.
- Securities are infinitely divisible with no taxes or transaction costs.
- There are no opportunities for riskless arbitrage in the market.

- The stock price follows a Geometric Brownian motion with μ and σ constant, defined as follow:

$$dS = \mu S dt + \sigma S dz\ , \tag{2}$$

where:

- μ is the expected return rate of the underlying asset;
- σ is the volatility of the underlying asset;
- dz is a Wiener process or standard Brownian motion.
- μSdt is the drift of the underlying;
- σSdz is the stochastic term.

For a European Call option with time to maturity $T > 0$ and exercise price $K > 0$, at time T, also known as "at maturity", $f(S, T) = \max(S - K, 0)$. If S tends to $+\infty$, then $f(S, t)$ will tend to $+\infty$, because the value of the option will be dominated by the stock price. If S tends to 0, then $f(S, t) \approx 0$, because it would not make sense to buy at a price $K > 0$ a stock that is worth 0.

For a European Put option with time to maturity $T > 0$ and exercise price $K > 0$, at time T, $f(S, T) = \max(K - S, 0)$. If S tends to $+\infty$, then $f(S, t) \approx 0$, because it would not make sense to sell a stock at a price $K < S$ when its value is potentially infinite. If S tends to 0, then $f(S, t) \approx Ke^{-r(T-t)}$, which is the present value of its exercise price.

For Call:

$$C(S,t) = Se^{-q(T-t)}\Phi(d_1) - Ke^{-r(T-t)}\Phi(d_2)\ , \tag{3}$$

for Put:

$$P(S,t) = Ke^{-r(T-t)}\Phi(-d_2) - Se^{-q(T-t)}\Phi(-d_1), \tag{4}$$

where:

- $C(S,t)$ is the price of the call option at time t;
- $P(S,t)$ is the price of the put option at time t;
- S is the current price of the underlying asset;

- q indicates dividends;
- K is the exercise price of the option;
- T is the expiration of the option;
- r is the risk-free interest rate;
- Φ is the normal cumulative distribution function.

With $d1$ and $d2$ respectively:

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - q + \frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}}, \tag{5}$$

$$d_2 = d_1 - \sigma\sqrt{T-t}\ . \tag{6}$$

3.2 Unsupervised and Supervised Machine Learning algorithms

Supervised learning leverages the power of labelled datasets. These datasets contain features, which are the input variables, and labels, which represent the desired output or target variable. Each data point acts as a training example, providing a clear pairing of input and its corresponding correct response. The core objective of supervised learning is to establish a mapping function based on this training data, allowing the model to predict the output for entirely new inputs. Essentially, the model learns to map unseen features to their corresponding labels. The success of a supervised learning model hinges on minimizing the error between its predictions and the true labels.

Unsupervised learning, on the other hand, tackles the challenge of unlabelled data. Here, the focus shifts from predicting specific outputs to uncovering inherent patterns and relationships within the data itself. Unsupervised learning algorithms primarily rely on clustering techniques to group similar data points together, revealing hidden structures and providing valuable insights into the underlying organization of the data. Specifically, for this work, we employed the K-Means Clustering model, Random Forest (RF) and Deep Neural Network (DNN).

3.2.1 K-Means Clustering

The K-Means algorithm partitions a dataset of N-dimensional observations into *k* clusters. Originally introduced by MacQueen, (1967) and enhanced by Lloyd, (1982) it aims to minimize the within-cluster variance, grouping similar observations together.

Notation:

- *k*: number of clusters;
- S_i: the set of points in cluster *i*;
- μ_i: the centroid of cluster *i*;
- Z: the dataset;
- z: random point of the dataset in $\mathbb{R}^d$ (d-dimensional space);
- W(S) the within-cluster variance defined as:

$$W(S) = \sum_{i=0}^{k-1} \sum_{z \in S_i} |z - \mu_i|^2 \ . \tag{7}$$

The (7) is the function to minimize, where $|z - \mu_i|$ represents the Euclidean distance between a point *z* and the centroid μ_i. To minimize it, the algorithm follows these steps and uses the k-means++ method by Arthur and Vassilvitskii, (2007), to improve the initialization process.

Steps:

- Select the optimal number of clusters *k*;
- Initialize the centroid μ_i by selecting random observation *z* from the dataset;
- Calculate the distance *D(z)* as:

$$D(z) = \min_{i \in \{0,\dots k-1\}} |z - \mu_i| \ , \tag{8}$$

then, for each point *z*, *P(z)* is computed, indicating the probability of being chosen as the next centroid:

$$P(z) = \frac{D(z)^2}{\sum_{z \in Z} D(z)^2} , \tag{9}$$

this process is repeated until *k* centroids are chosen.

The next step is to assign z to the cluster S_i such that $|z - \mu_i|$ is minimized.

At the end for each S_i is recalculated the centroid μ_i as the mean of all point assigned to the cluster:

$$\mu_i = \frac{1}{|S_i|} \sum_{z \in S_i} z \,, \tag{10}$$

where $|S_i|$ is the number of observations in the cluster.

By following these steps and using the k-means++ method for initialization, the algorithm iteratively assigns points to clusters and recomputes centroids until convergence (centroids no longer change significantly).

3.2.2 Random Forest

The Random Forest model, developed by Breiman, (2001), is a supervised learning algorithm that operates on the "divide and conquer" principle. It is based on ensemble learning, specifically aggregating various decision tree models to enhance stability and accuracy of predictions. The input data are divided into groups, and a decision tree is created for each group, the results then are aggregated to make a single prediction. It was demonstrated that growing trees in random subspaces helps tackle overfitting and improves model generalization (Ho, 1995). Random Forest is highly well known for its predictive accuracy and its ability to handle vast amounts of data in a multi-dimensional space with minimal calibration parameters required for both regression and classifications problems. Understanding Random Forests requires familiarity with two key components:

- Bagging (Bootstrap Aggregation): Bagging creates new datasets by sampling (with replacement) from the original data. Each new dataset (called a bootstrap sample) trains a separate decision tree. Finally, the predictions from all trees are averaged for regression problems or a majority vote is taken for classification problems (Breiman, 1996).
- CART (Classification and Regression Trees): The CART algorithm guides the construction of each tree, where at each node it finds the optimal split by minimizing a specific criterion. This criterion is Gini impurity for classification tasks and Mean Squared Error (MSE) for regression tasks (Breiman et al., 1984).

Each tree in a Random Forest starts with a root node containing a bootstrap sample of the training dataset. For a given node, using a variant of the CART, the split is chosen by selecting the feature and the split-point that minimize the MSE (for regression) following the formula:

$$MSE = \min_{j,t}\left(\frac{N_{left}}{N} MSE_{left} + \frac{N_{right}}{N} MSE_{right}\right), \quad (11)$$

where j is a feature, t is the threshold, N is the total number of samples at the node, and N_{left} and N_{right} are the numbers of samples in the left and right subsets formed by the split.

To evaluate the impurity within a single node the MSE_{node} is computed as follows:

$$MSE_{node} = \frac{1}{N_{node}} \sum_{(i \in node)} (y_i - \bar{y}_{node})^2, \quad (12)$$

where $\bar{y}_{node}$ node is the average of the target values in the node.

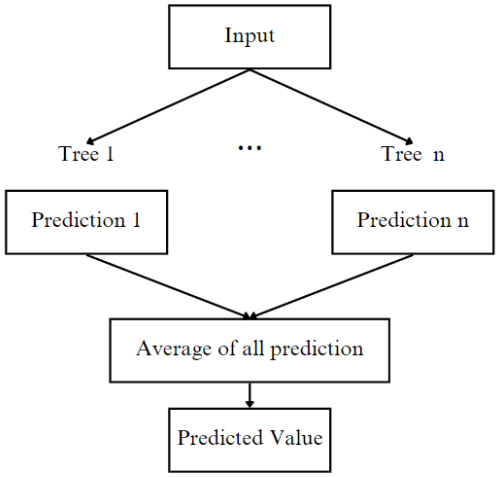
When a node can no longer be effectively split, it becomes a leaf node. In the leaf nodes, the predicted value for a new observation is determined by the average of the target values within that node.

Considering ξ as a vector of random variables, where each random variable corresponds to a choice in the construction of the tree (e.g., which indices are chosen for the bootstrap sample, and which features are selected at each split). For a given vector x (containing the information of a single observation), every tree in the forest produces a prediction; the average of these predictions is the forest prediction, and this is described by:

$$h_{forest}(x) = \frac{1}{K} \sum_{k=1}^{K} h_k(x, \xi), \quad (13)$$

where $h_k(x, \xi)$ is the prediction from the k-th tree, built with the random vector ξ_k.

Figure II Random Forest Regression



3.2.2 Deep Neural Network

Artificial Neural Networks (ANNs) are systems inspired by the biological neural networks of animal brains. The structure of the "neuron" was firstly introduced by McCulloch and Pitts (1943), then an early implementation of a NN was presented by Rosenblatt, (1958). The structure of our ANN is called Feedforward Networks, because the input layer, comprised of source nodes, feeds data forward to the output layer containing computation nodes and there are no connections carrying information back from the output layer (Haykin, 2009).

We can summarize a simple ANN structure as follows:

Neurons: Basic units of computation, receives the input, processes it and passes the output to the next layer.

Layers:

- Input Layer: first layer which has the role of receiving the unprocessed input data.

- Hidden Layers: intermediate layers composed of several neurons that process inputs from the previous layer. There can be one or multiple hidden layers. The name "Deep" derives from the number of hidden layers the ANN has.
- Output Layer: the final layer that produces the output of the network.

The way neurons process the information received is through functions called activation functions, these are divided into linear and non-linear.

Some of the most known activation functions are:

- Identity $f(x) = x$;
- Sigmoid (σ) $f(x) = \frac{1}{1+e^{-x}}$;
- Hyperbolic Tangent (tanh) $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$;
- Rectified Linear Unit (ReLu) $f(x) = \max(x, 0)$.

The universal approximation theorem states that a feedforward neural network with at least a single hidden layer and non-linear activation function containing a finite number of neurons can approximate any continuous function on a compact subset of $\mathbb{R}^n$ (Cybenko, 1989 and Hornik, 1991).

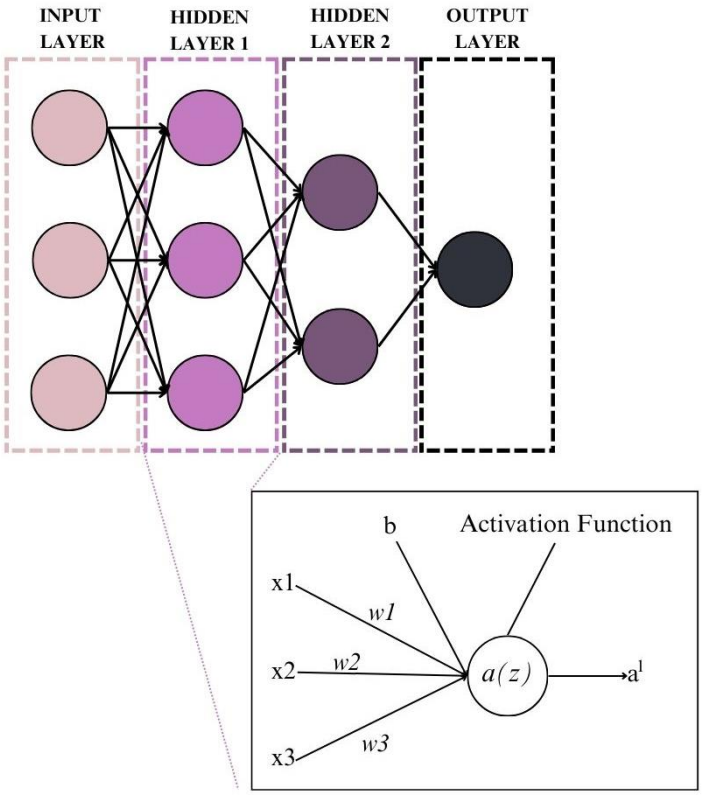
The process to train a FNN is through backpropagation concept introduced by Rumelhart et al., (1986), following we will show how a simple FNN is trained,but first let's clarify the notation we will use:

- n = input layer's number of neurons;
- h = hidden layer's number of neurons;
- m = output layer's number of neurons;
- x: Input vector of size n;
- $W^{(1)}$: Weight matrix between the input and hidden layer, of size h×n;
- $b^{(1)}$: Bias vector for the hidden layer, of size h;
- $a^{(1)}$: Activation vector for the hidden layer, of size h;
- $W^{(2)}$: Weight matrix between the hidden layer and output layer, of size m×h;
- $b^{(2)}$: Bias vector for the output layer, of size m;
- $a^{(2)}$: Activation vector for the output layer (output of the network), of size m;
- λ: regularization parameter;

- y: Target output vector of size m;
- L: Loss function;
- ⊙: Hadamard product;
- η: learning rate.

Figure III will help us to visualize the structure of a simple DNN:

Figure III DNN simple structure



Starting from the first layer the information pass from the input layer to the hidden layer, it is computed the weighted sum of the input plus a constant called bias and it's applied the activation function selected (σ):

$$z^{(1)} = W^{(1)}x + b^{(1)}\,, \tag{14}$$

$$a^{(1)} = \sigma(z^{(1)})\,. \tag{15}$$

Same process happens to pass information between the hidden layer and the output layer, where in the case of regression the activation function will be an identify function (ϕ):

$$z^{(2)} = W^{(2)}a^{(1)} + b^{(2)}, \tag{16}$$

$$a^{(2)} = \phi(z^{(2)}). \tag{17}$$

After this process is completed, the loss function is calculated (in case of regression task MSE is used), which is the difference between the network prediction and the actual target value, to prevent overfitting we also introduce the Ridge Regularization (L2) that will add a penalty term equal to the sum of the squared values of the weights: $\lambda \sum_j W_j^2$

$$L = \frac{1}{N}\sum_{i=1}^{N}(a_i - y_i)^2 + \lambda \sum_j W_j^2 . \tag{18}$$

This is where the mechanism of backpropagation comes in to try to improve the prediction power of the model. Using the chain rule of calculus, the network calculates the gradients of the loss function with respect to each weight and bias. There are two primary processes involved in this process: compute error and spread them throughout the network:

The Output Layer error:

$$\delta^{(2)} = a^{(2)} - y \ , \tag{19}$$

and the Hidden Layer Error:

$$\delta^{(1)} = \left(W^{(2)}\right)^T \delta^{(2)} \odot \ \sigma'(z^{(1)}) . \tag{20}$$

The first part of the equation $\left(W^{(2)}\right)^T \delta^{(2)}$ represents the error propagated back to the hidden layer. The second part $\sigma'(z^{(1)})$, is the derivative of the activation function at the hidden layer indicating how the activation function at this stage responds to changes in its input.

Once the layer errors are computed, the next step is to calculate the gradients, which indicate how much each parameter should be adjusted to minimize the loss:

$$\frac{\partial \mathrm{L}}{\partial \mathrm{W}^{(2)}} = \delta^{(2)}\left(a^{(1)}\right)^T + \lambda W^{(2)} ; \tag{21}$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{b}^{(2)}} = \delta^{(2)}; \tag{22}$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{W}^{(1)}} = \delta^{(1)}(x)^T + \lambda W^{(1)} ; \tag{23}$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{b}^{(1)}} = \delta^{(1)} . \tag{24}$$

With the gradients computed, we can then update the weights and biases as:

$$W^{(2)} = W^{(2)} - \eta \frac{\partial \mathrm{L}}{\partial \mathrm{W}^{(2)}} ; \tag{25}$$

$$b^{(2)} = b^{(2)} - \eta \frac{\partial \mathrm{L}}{\partial \mathrm{b}^{(2)}} ; \tag{26}$$

$$W^{(1)} = W^{(1)} - \eta \frac{\partial \mathrm{L}}{\partial \mathrm{W}^{(1)}} ; \tag{27}$$

$$b^{(1)} = b^{(1)} - \eta \frac{\partial \mathrm{L}}{\partial \mathrm{b}^{(1)}} . \tag{28}$$

Among the different optimisation methods, we used Adam optimiser introduced by Kingma et al, (2014). Adam stands for Adaptive Moment Estimation, and it combines other two methods AdaGrad and RMSProp. It works maintaining and updating the moving averages of both gradients and their squares.

The algorithm uses the following parameters:

- m_0: first moment vector;
- v_0: second moment vector;
- $t = 0$ initial timestep;
- $\beta_1 = 0.9$ and $\beta_2 = 0.99$: Exponential decay rates for the moment estimates ;
- ε: small constant to prevent division by zero;
- g_t : the gradient of the loss function at timestep t.

For each iteration t it will update the parameters:

Update the mean of gradients vector $\mathrm{m_t}$ and compute the bias-corrected first moment estimate $\widehat{m}_t$:

$$\mathrm{m_t} = \beta_1 \mathrm{m_{t-1}} + (1 - \beta_1)\mathrm{g_t} \ , \quad \widehat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{29}$$

Update of the uncentered variance of gradients vector v_t and compute the bias-corrected second moment estimate $\hat{v}_t$:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 , \quad \hat{v}_t = \frac{m_t}{1 - \beta_2^t} , \tag{30}$$

parameters update:

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\varepsilon + \sqrt{\hat{v}_t}} . \tag{31}$$

4. Data and Methodology

This research utilized data extracted from the Ivy DB US database by OptionMetrics present in the *Wharton Research Data Service*. The dataset comprises options (secid: 108105) with the S&P 500 as the underlying asset, spanning the period from January 1, 2018, to December 31, 2022.

In details the following datasets have been combined: Option Prices, Zero Coupon Yield Curve, Index Dividend Yield, Security Prices, Historical Volatility.

The Option Prices dataset contains 21,242,363 rows and has the following variables: Date showing the date of the observation. Expiration Date indicating the date when the option will expire. Cp_flag to distinguish Call and Put. Strike price. Best bid and best ask captured at 15:59 ET to better synchronize the option price with the underlying close. Volume indicating the total volume of option contracts. Open Interest lagged by one day, indicating the number of total contracts outstanding. Implied Volatility derived by numerically inverting the Black-Scholes formula, using the midpoint of the best bid and best offer closing price as the theoretical price.

The Zero-Coupon Yield Curve dataset contains 47,659 rows and has the following variables: Date showing the date of the observation. Days, showing the number of days to maturity. Rate, the continuously compounded zero-coupon interest rate. The structure is obtained enforcing the put-call parity relationship and by utilizing box spreads, a combination of bull call spread and a bear put spread, that involves to buying and selling calls and puts with the same expiration date but different strike prices.

The Index Dividend Yield dataset contains 1,259 rows and has the following variables: Date showing the date of the observation, and Dividend Yield. OptionMetrics for index options assumes that the security pays dividends continuously according to a continuously compounded dividend yield. The implied index dividend is calculated assuming a relationship of put-call parity through a linear regression model.

In this model, the difference $C-P$ between call and put option prices uses the bid price of the call and the offer price of the put, and vice versa. The regression uses three months of option data, excluding contracts with less than 15 days to expiry.

The Security Prices dataset contains 1,259 rows and has the following variables: Date showing the date of the observation. Spot close, showing the closing price for the security on this date.

The Historical Volatility dataset contains 16,367 rows and has the following variables: Date showing the date of the observation. Days, the number of days included in the calculation. Volatility, the realized volatility for different timeframes (we will use the 30 and 60 days).

Integration of the datasets was primarily conducted through date matching. However, the Zero-Coupon Yield Curve dataset required a distinct approach, being merged with the other datasets through linear interpolation based on varying short-term yield rates across different dates.

Various filters were applied to the datasets to ensure data quality and relevance. Specifically, the maturity dates for the options ranged from 14 to 252 days, covering the period from December 30, 2019, to December 30, 2022. To exclude illiquid options, a minimum volume threshold of 20 contracts per option was established. Additionally, implied volatility was capped at 0.8 for consistency. Recognizing the impact of dataset size on machine learning model performance, an equal number of observations for both call and put options were ensured for fair comparison. Consequently, 304 observations per day were selected for both call and put options datasets. As a result, we obtained two datasets: one for call options and one for put options, each containing 230,432 observations. Additionally, four more datasets were created in which Implied Volatility (IV) was replaced with 30- and 60-day Historical Volatility (referred to as 30d-HV and 60d-HV or HV30 and HV60). This change was made to avoid potential data leakage for

the machine learning models, as IV is derived from the inverse Black-Scholes-Merton (BSM) model using market prices. This may lead to losing information regarding the option's specific volatility, especially considering the volatility smile phenomenon where options that are deep in-the-money or out-of-the-money generally have higher implied volatilities than at-the-money options.

All our models employed a random split of 80/20 between the training and test sets, setting a seed to ensure reproducibility. In addition, the Deep Neural Network Model added another split of 20% of the training set for the validation set. After standardizing the variables, we ran the RF and the DNN first without applying any kind of clustering technique, where our machine learning models had price as the target and used the standard variables of the BSM model (S, K, r, σ, DTE, q) as features. The performance of these models was evaluated on the test set using various metrics, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). Then we used the Calinski-Harabasz score and the Elbow Method to decide the optimal number of clusters for the six datasets.

Figure IV Calinski-Harabasz Score

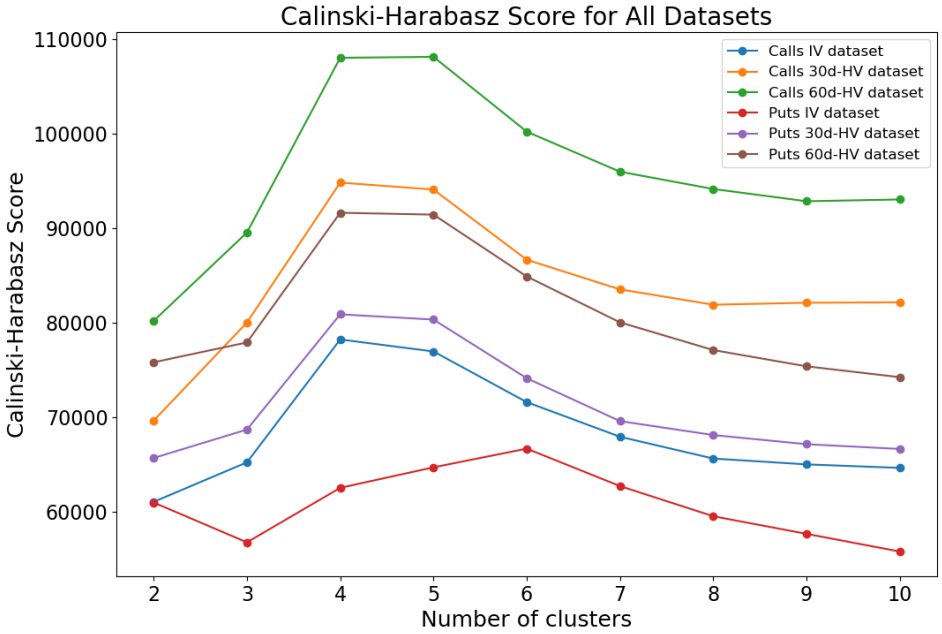
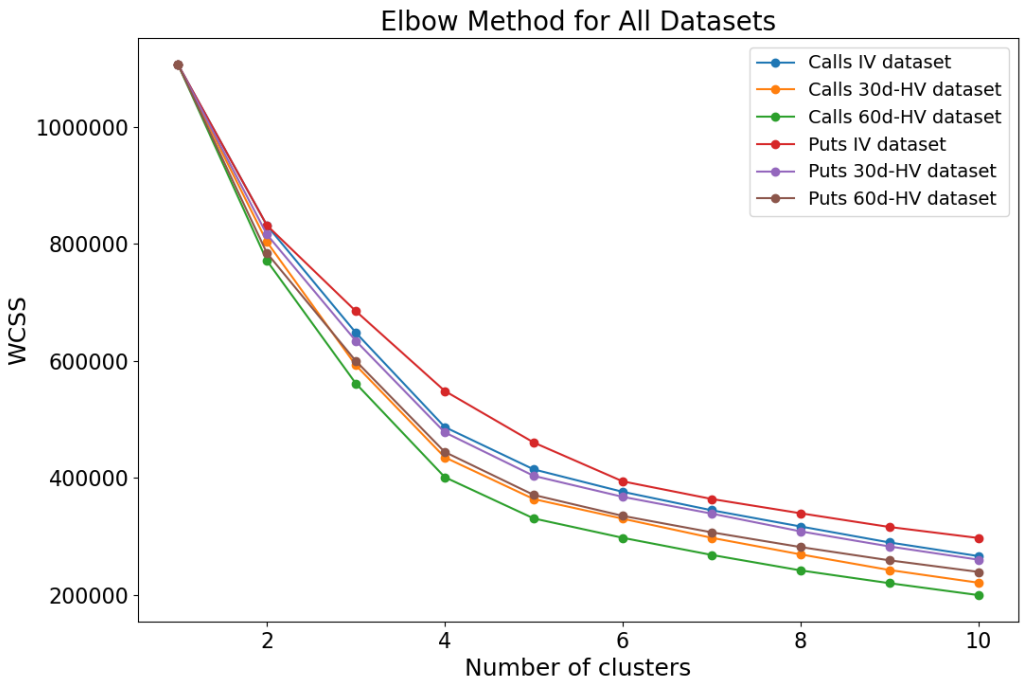


Figure V Elbow Method



For the call datasets, both methodologies found that the best number of clusters was four. For the put datasets, the two methods gave us different best numbers of clusters; specifically, the Elbow Method showed the best number of clusters to be four, while the Calinski-Harabasz score showed six as the best number of clusters for the Puts IV datasets and four for the 30d-HV and 60d-HV datasets. Therefore, we decided to proceed using four clusters for all the datasets. Subsequently, we applied K-Means clustering to the training data to group observations into clusters. We then trained a separate supervised ML model for each cluster.

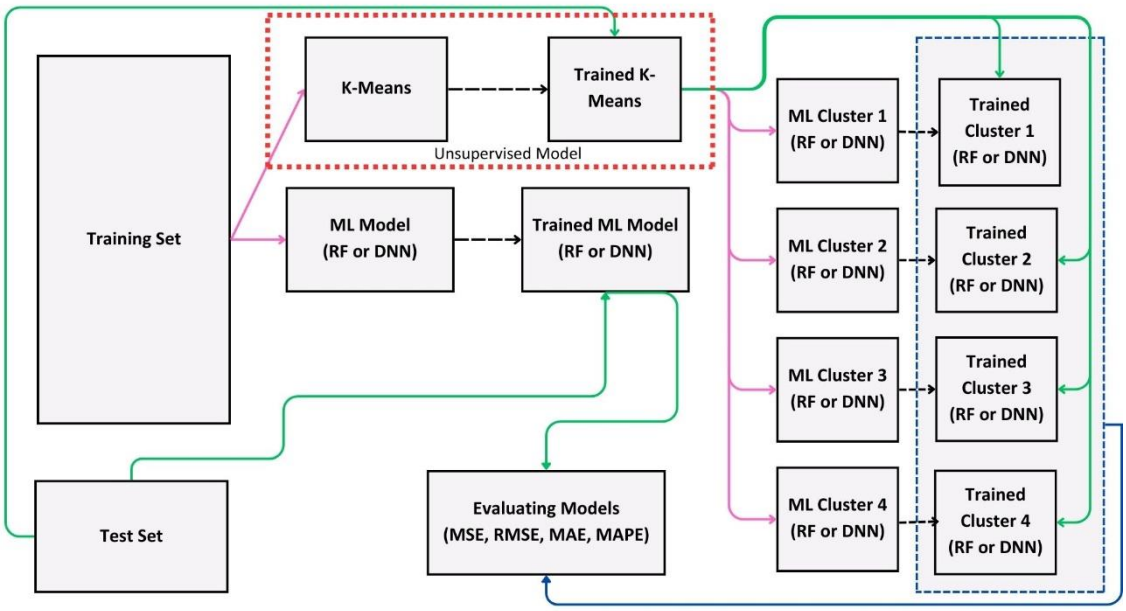
The Random Forests had a unique parameter manually selected, which specifies the number of trees used by the model to make predictions set at 100. This algorithm has the characteristic of being robust to overfitting, in simple words: the ability of the model to adapt to the details and noise of the training set too well, becoming ineffective in predicting unseen data. An elevated number of trees will then lead to reaching a plateau where the model will not improve performance, but will take longer to finish execution.

The DNN was built with an input layer to receive the features, in addition to two hidden layers, each with 64 neurons, and an output layer. To avoid overfitting, all layers except the input layer had L2 regularization with a regularization strength equal to 0.01 and early stopping set to five, which would stop the execution if the validation set did not

show improved performance for five consecutive epochs. The number of epochs for each model was set to 40, representing the number of times the learning algorithm will work through the full dataset during the training process, and the batch size was set to 32, determining how many samples are processed before the model's internal parameters are updated. In order to avoid data leakage, the performance of these models was evaluated on the test dataset by assigning each test observation to its respective cluster using a pre-trained K-Means clustering model. The corresponding cluster-specific supervised ML model was then used to make predictions. The evaluation metrics for these clustered models were also recorded. We conducted an analysis of the centroids on the original scale to understand the characteristics of each group and check the features importance range. Additionally, to determine if the clustering method has created groups that are statistically distinct based on their feature values, we performed ANOVA test. If the variance between clusters is significantly higher than the variance within clusters, the feature is considered significant.

The following images will summarize the workflow previously explained.

Figure VI Hybrid Machine Learning Workflow



The pink lines show the flow of the training data, while the green lines indicate the flow of the test data. The black dashed lines depict the model's status change from untrained to trained. Finally, the blue line represents the flow of the test data for evaluation.

5. RESULTS

After having presented in the previous chapters the theoretical foundations and detailed workings of the models and the data used, it is time to see whether RF and DNN performed better than BSM model, and if the K-means clustering technique had an impact on improving their performance. Additionally, we will go deeper into cluster centroid analysis that will provide insights into the clustering results.

5.1 Model Performance Comparison

Since these metrics can exhibit bias—such as Mean Squared Error (MSE), which is particularly sensitive to outliers due to the squaring of errors, and Mean Absolute Percentage Error (MAPE), which can be disproportionately impacted by values close to zero because it involves division by the actual values—we decided to use all the following metrics:

- Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 \quad \textbf{(32)}$$

- Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \quad \textbf{(33)}$$

- Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}| \quad \textbf{(34)}$$

- Mean Absolute Percentage error (MAPE):

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{y_i - \widehat{y_i}}{y_i}\right] * 100 \quad \textbf{(35)}$$

By employing multiple evaluation metrics, we aim to achieve a more comprehensive and clearer understanding of the model's performance, ensuring that the assessment is balanced and takes various potential biases into account.

TABLE I MODEL PERFORMANCE COMPARISON.

Model Performance Comparison Table presents: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) comparison across different machine learning models for option pricing. The models are ranked based on their MSE values. Values highlighted in green show better performance, while values highlighted in red show worse performance.

Model	MSE	RMSE	MAE	MAPE
DNN with Clustering Calls IV dataset	1.98	1.41	0.79	19.12
DNN with Clustering Puts IV dataset	3.07	1.75	0.90	11.03
DNN Calls IV dataset	5.79	2.41	0.83	35.84
DNN Puts IV dataset	6.34	2.52	0.92	20.49
DNN with Clustering Calls 30d-HV dataset	28.47	5.34	3.29	52.52
DNN with Clustering Calls 60d-HV dataset	31.88	5.65	3.46	40.96
DNN with Clustering Puts 30d-HV dataset	34.85	5.90	3.35	25.26
Random Forest with Clustering Calls 60d-HV dataset	40.75	6.38	2.78	9.60
Random Forest with Clustering Calls 30d-HV dataset	41.41	6.44	2.82	10.00
Random Forest Calls 60d-HV dataset	41.44	6.44	2.74	10.30
Random Forest Puts 60d-HV dataset	42.43	6.51	2.48	7.32
DNN with Clustering Puts 60d-HV dataset	43.17	6.57	3.78	24.50
Random Forest Calls 30d-HV dataset	45.36	6.73	2.74	10.17
DNN Puts 60d-HV dataset	47.10	6.86	3.98	43.55
DNN Puts 30d-HV dataset	47.29	6.88	3.97	42.81
Random Forest Puts 30d-HV dataset	48.39	6.96	2.45	7.36
DNN Calls 60d-HV dataset	48.99	7.00	4.34	100.29
Random Forest with Clustering Puts IV dataset	50.05	7.07	2.58	8.34
DNN Calls 30d-HV dataset	50.20	7.09	4.65	100.83
Random Forest with Clustering Puts 60d-HV dataset	52.91	7.27	2.45	7.14
Random Forest with Clustering Calls IV dataset	54.37	7.37	2.81	25.07
Random Forest Calls IV dataset	71.17	8.44	2.92	53.94
Random Forest with Clustering Puts 30d-HV dataset	71.44	8.45	2.47	7.21
Random Forest Puts IV dataset	84.84	9.21	2.70	8.35
BSM Puts 60d-HV dataset	2,138.49	46.24	24.11	67.94
BSM Puts 30d-HV dataset	2,222.80	47.15	24.14	64.11
BSM Calls 30d-HV dataset	4,341.70	65.89	35.43	293.47
BSM Calls 60d-HV dataset	4,685.00	68.45	36.25	335.87

In Table I, the cells in green represent lower values for each metric, indicating better model precision, while the cells in red highlight models with worse performance. We observe that the models which performed better overall, particularly in terms of MSE, are the Deep Neural Networks (DNNs). Specifically, the DNN with Clustering Calls IV and DNN with Clustering Puts IV datasets showed superior performance

To evaluate whether the introduction of the clustering technique enhanced model performance, we compute the difference Δ between the performance metrics of the same models with and without clustering, as presented in the following table.

TABLE II CLUSTERING EFFECT ON MODELS' PERFORMANCE

The table presents the differences (Δ) in MSE, RMSE, MAE, and MAPE. The models evaluated include Deep Neural Networks (DNN) and Random Forest (RF), both with and without clustering, across datasets for Calls and Puts with different volatility measures (IV and HV). A positive Δ indicates that clustering has improved the model's performance (cells in green), if clustering did not significantly improve the model - $0.5 < \Delta < 0.5$ (cells in yellow). Negative Δ means that clustering has worsened the model's performance (cells in red).

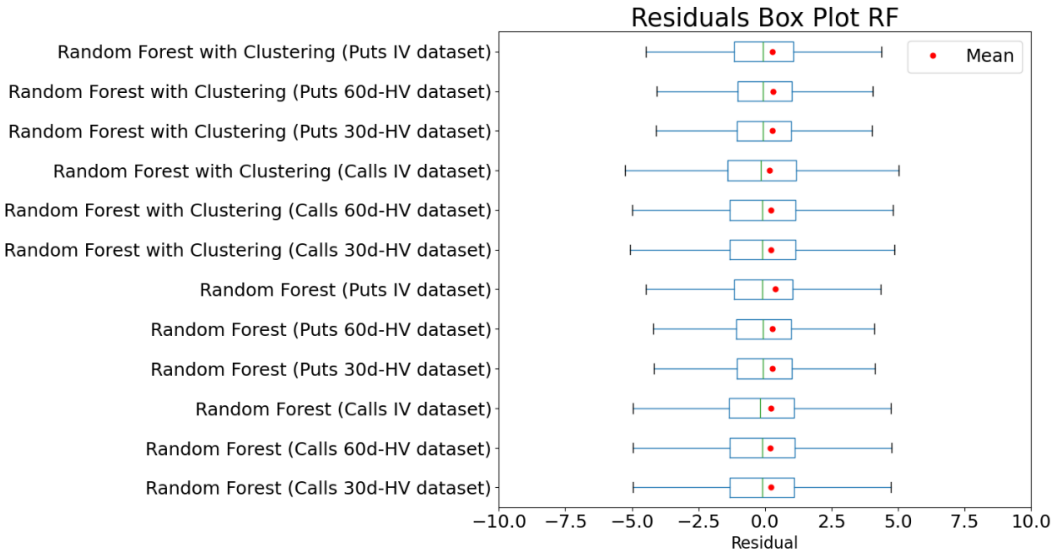
Model	Δ MSE	Δ RMSE	Δ MAE	Δ MAPE
DNN Calls IV dataset				
DNN with Clustering Calls IV dataset	3.81	1.00	0.04	16.72
DNN Calls 30d-HV dataset				
DNN with Clustering Calls 30d-HV dataset	21.73	1.75	1.35	48.31
DNN Calls 60d-HV dataset				
DNN with Clustering Calls 60d-HV dataset	17.11	1.35	0.89	59.33
DNN Puts IV dataset				
DNN with Clustering Puts IV dataset	3.27	0.77	0.02	9.46
DNN Puts 30d-HV dataset				
DNN with Clustering Puts 30d-HV dataset	12.43	0.97	0.62	17.54
DNN Puts 60d-HV dataset				
DNN with Clustering Puts 60d-HV dataset	3.93	0.29	0.20	19.05
Random Forest Calls IV dataset				
Random Forest with Clustering Calls IV dataset	16.79	1.06	0.10	28.87
Random Forest Calls 30d-HV dataset				
Random Forest with Clustering Calls 30d-HV dataset	3.95	0.30	- 0.08	0.18
Random Forest Calls 60d-HV dataset				
Random Forest with Clustering Calls 60d-HV dataset	0.69	0.05	- 0.04	0.70
Random Forest Puts IV dataset				
Random Forest with Clustering Puts IV dataset	34.79	2.14	0.12	0.01
Random Forest Puts 30d-HV dataset				
Random Forest with Clustering Puts 30d-HV dataset	- 23.04	- 1.50	- 0.02	0.15
Random Forest Puts 60d-HV dataset				
Random Forest with Clustering Puts 60d-HV dataset	- 10.48	- 0.76	0.03	0.18

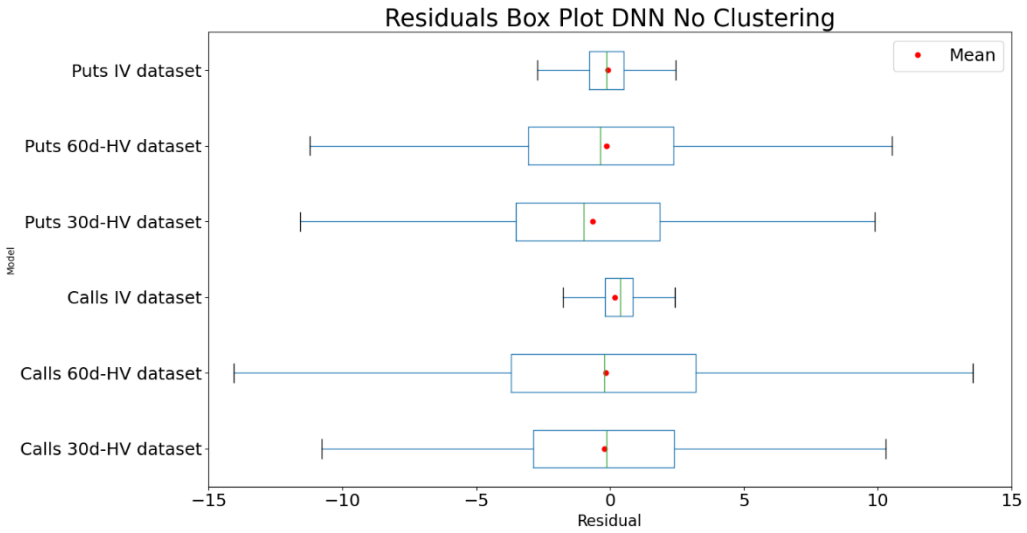
Overall K-Means clustering technique improved all the DNN models' performance. Random Forest show mixed results, where clustering enhances performance for some datasets, but decreases performance for others (RF Puts 30-and-60 days HV datasets).

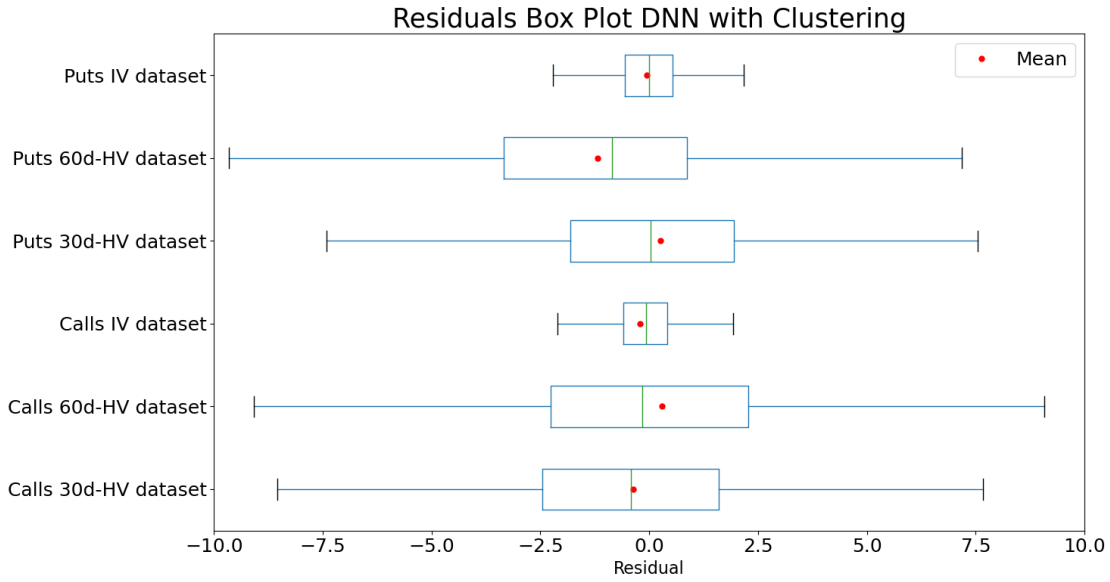
As the second step to evaluate the performance of our models we conducted a residual analysis. From Figure VII, it is evident that the ML models are performing better than

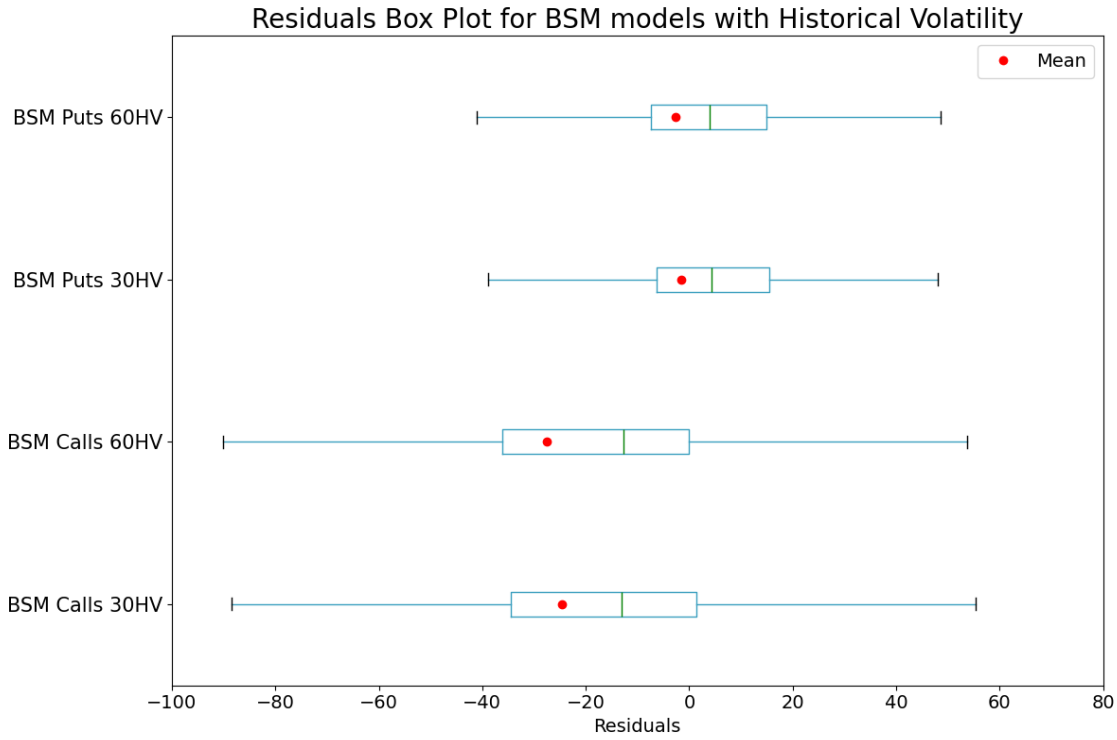
BSM. They are centered around zero, showing low variability, with a narrow interquartile range. This analysis also highlights the significant improvement in the DNN models due to the clustering method, while RFs are showing less improvement given by K-Means. Additionally, it reveals the tendency of the BSM models to underprice both call and put options, whereas the Random Forest models slightly overprice them.

Figure VII Residual Analysis







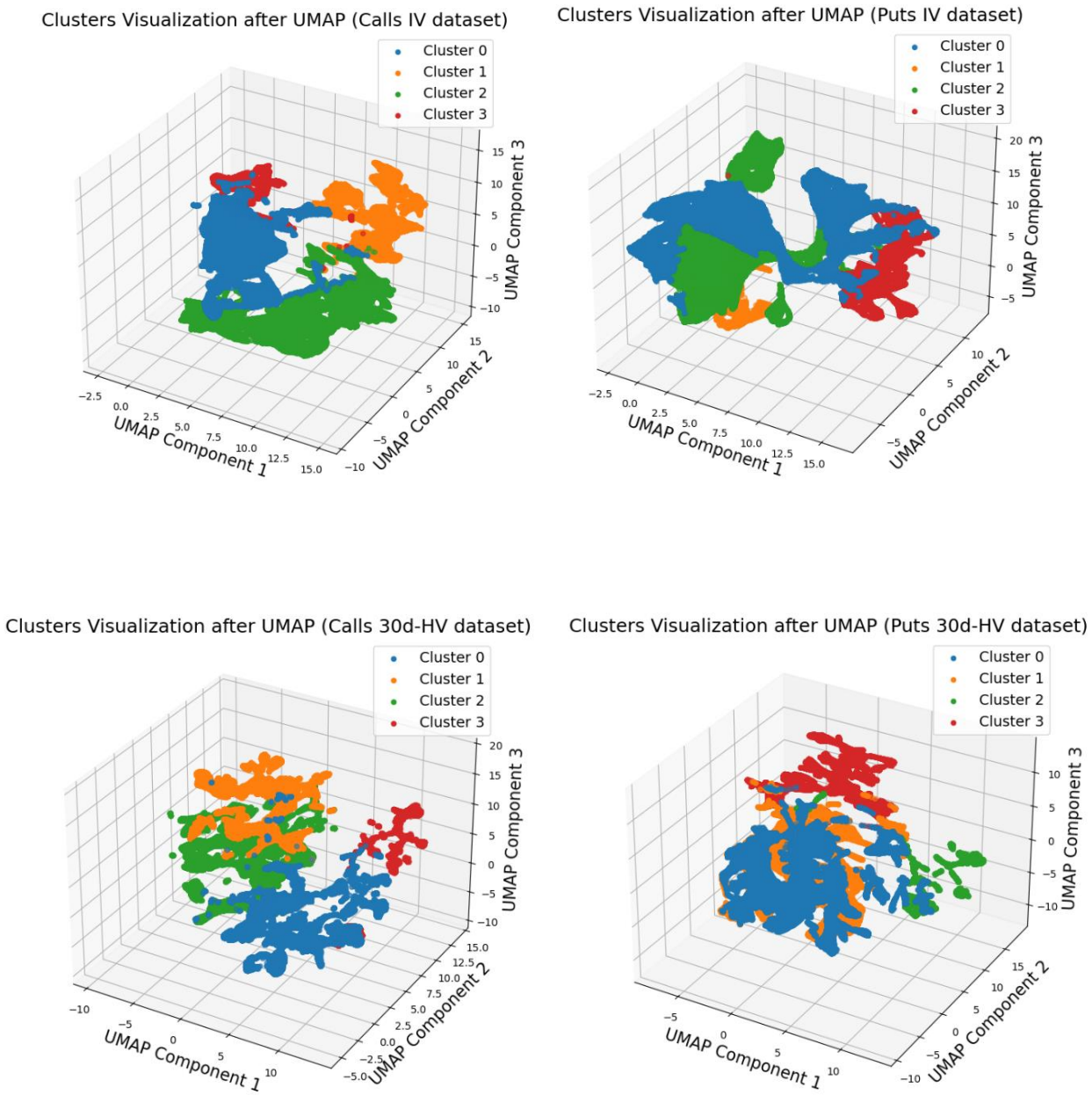


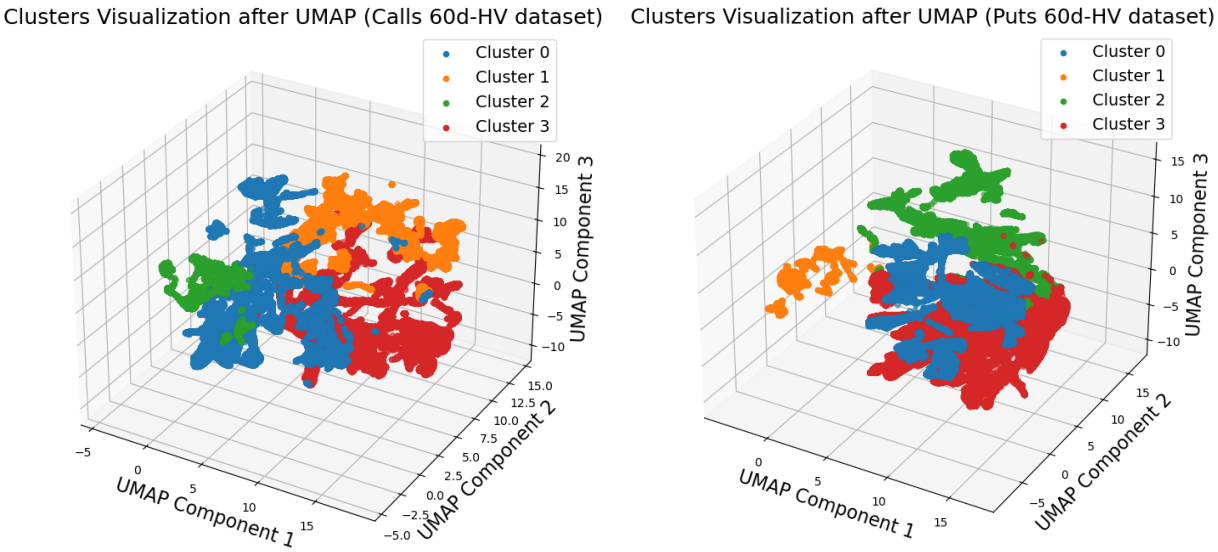
5.2 Centroid Analysis

For visual purposes in Figure VIII, we employed the Uniform Manifold Approximation and Projection (UMAP), a methodology introduced by McInnes et al., (2018). UMAP is a non-linear dimensionality reduction technique that transforms data

into a new coordinate system, reducing the number of dimensions while preserving both the local and global structure of the data as much as possible.

Figure VIII UMAP Clusters Visualization





To gain a more nuanced understanding of the clusters' characteristics, in Table III we present the results of the cluster centroid analysis.

TABLE III CENTROID ANALYSIS

The table displays key variables (K, DTE, q, r, S, IV/HV) for Calls and Puts datasets, segmented into clusters. Variables include strike price (K), days to expiration (DTE), dividend yield (q), risk-free rate (r), underlying asset price (S), and implied/historical volatility (IV/HV). Each cluster represents a group of similar option contracts. Colours indicate the relative values of each variable within the same dataset: varying across the clusters from green for higher values to red for lower values.

Calls IV dataset

Cluster	K	DTE	q	r	S	IV
0	3,560.74	0.22	0.016	0.004	3,458.74	0.19
1	4,166.19	0.26	0.012	0.028	3,916.69	0.22
2	4,550.57	0.23	0.013	0.003	4,382.94	0.15
3	2,923.22	0.29	0.003	0.006	2,783.55	0.33

Calls 30-d HV dataset

Cluster	K	DTE	q	r	S	HV30
0	3,539.22	0.22	0.016	0.004	3,454.99	0.17
1	4,172.63	0.27	0.012	0.029	3,913.92	0.25
2	4,542.28	0.23	0.013	0.003	4,379.96	0.15
3	2,965.39	0.28	0.001	0.006	2,703.67	0.64

Calls 60-d HV dataset

Cluster	K	DTE	q	r	S	HV60
0	3,542.94	0.22	0.016	0.004	3,458.63	0.18
1	4,171.67	0.27	0.012	0.029	3,912.91	0.25
2	2,973.53	0.28	0.002	0.006	2,719.62	0.66
3	4,540.88	0.23	0.013	0.003	4,378.68	0.15

Puts IV dataset

Cluster	K	DTE	q	r	S	IV
0	4,003.05	0.22	0.013	0.003	4,308.94	0.25
1	2,338.33	0.30	0.002	0.006	2,780.37	0.50
2	2,867.89	0.26	0.016	0.005	3,548.49	0.37
3	3,468.97	0.30	0.012	0.031	3,920.83	0.30

Puts 30-d HV dataset

Cluster	K	DTE	q	r	S	HV30
0	2,897.33	0.26	0.016	0.004	3,479.59	0.17
1	3,954.08	0.23	0.013	0.003	4,355.06	0.16
2	2,356.94	0.29	0.001	0.006	2,703.51	0.64
3	3,444.08	0.30	0.012	0.029	3,920.82	0.25

Puts 60-d HV dataset

Cluster	K	DTE	q	r	S	HV60
0	3,445.67	0.30	0.012	0.029	3,919.73	0.25
1	2,367.67	0.29	0.001	0.006	2,717.86	0.66
2	2,899.66	0.26	0.016	0.004	3,481.33	0.18
3	3,952.99	0.23	0.013	0.003	4,357.08	0.15

For Calls IV dataset:

- Cluster 0 has a moderate strike and underlying price, the highest dividend yield. low volatility, and short time to expiration. On average, options within this cluster might be At-the-money using the formula
- Cluster 1 has a high strike price and underlying price, moderate volatility, and the highest risk-free rate.
- Cluster 2 has the highest strike price and underlying price, with the lowest volatility and a short time to expiration.
- Cluster 3 has the lowest strike and underlying price, highest volatility and time to expiration.

For Calls 30-d HV dataset:

- Cluster 0 has a moderate strike and underlying price, highest dividend yield, low volatility, and shortest time to expiration.
- Cluster 1 has a high strike and underlying price, moderate volatility, and a slightly longer time to expiration, with the highest risk-free rate.
- Cluster 2 has the highest strike and underlying price, with the lowest volatility and a short time to expiration.
- Cluster 3 has the lowest strike and underlying price, highest volatility, and the longest time to expiration.

For Calls 60-d HV dataset:

- Cluster 0 has a moderate strike and underlying price, low volatility, and a shortest time to expiration.
- Cluster 1 has a high strike and underlying price, moderate volatility, and a slightly longer time to expiration, with the highest risk-free rate.
- Cluster 2 has the lowest strike and underlying price, highest volatility, and the longest time to expiration.
- Cluster 3 has the highest strike and underlying price, with the lowest volatility and a short time to expiration.

For Puts IV dataset:

- Cluster 0 has the highest strike and underlying price, with lowest volatility, time to expiration and risk-free rate.
- Cluster 1 has the lowest strike price and underlying price, the highest volatility, and the longest time to expiration.
- Cluster 2 has a moderate strike price and volatility, with a relatively short time to expiration.
- Cluster 3 has a high strike price and moderate volatility, with the longest time to expiration and the highest risk-free rate.

For Puts 30-d HV dataset:

- Cluster 0 presents moderate values with the highest dividend yield.
- Cluster 1 has the highest strike and underlying price, and the lowest volatility.
- Cluster 2 has the lowest strike and underlying price, and highest volatility.
- Cluster 3 has a relatively high strike and underlying price, moderate volatility, and the highest time to expiration.

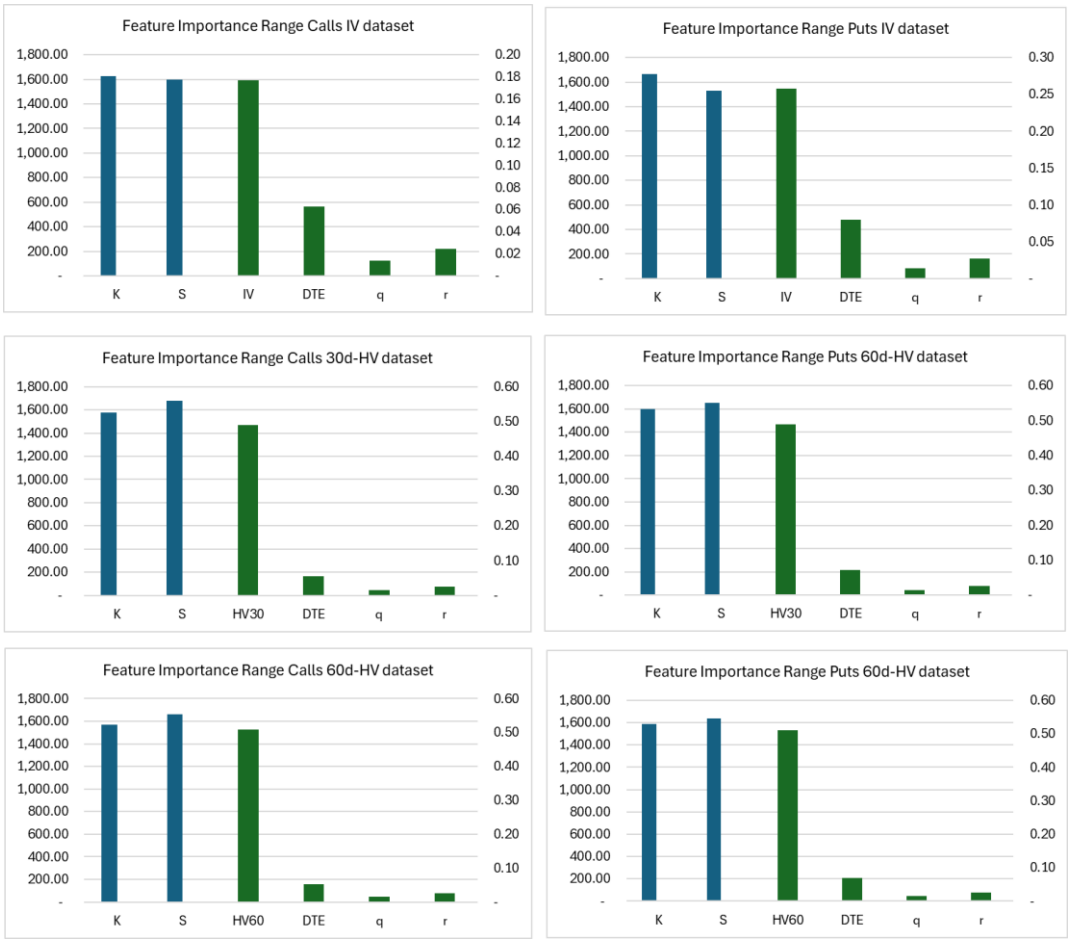
For Puts 60-d HV dataset:

- Clusters 0 present moderate values for strike and underlying price, with highest time to maturity and risk-free rate.
- Cluster 1 has the lowest strike and underlying price, and highest volatility.
- Cluster 2 presents moderate values with the highest dividend yield.
- Cluster 3 has the highest strike price and underlying asset price, but the lowest time to maturity and historical volatility.

Then to check if the K-means algorithm successfully splits the data, for each dataset and for each feature, ANOVA was performed with a 95% confidence interval. The null hypothesis H_0 is the means of the feature are equal across all clusters. A p-value greater than 0.05 suggests no significant differences. In this analysis, each feature has a p-value of 0.0000, which is substantially lower than the 0.05 threshold, thus confirming significant differences in the means of these features across the clusters.

Lastly, we check the range of each feature across cluster centroids, to understand how features varied across the clusters.

Figure IX Features Importance Range



Strike price *K* and Underlying price *S* in blue refer to the primary axis while the other variables, in green, are referring to the secondary axis. We can observe that the range depends on the nature of the variable itself, where features normally with higher value have the widest range.

6. CONCLUSION

This study explored the potential of integrating unsupervised and supervised machine learning models, specifically K-means clustering, Random Forest, and Deep Neural Networks, in option pricing. The primary objective was to assess whether these models could outperform the traditional Black-Scholes-Merton model and to investigate the impact of clustering on predictive accuracy. The findings indicate that DNN models, particularly when combined with clustering techniques, showed superior performance in option pricing compared to both non-clustered machine learning models and the BSM model, while RF even if exhibited improvement in performance compared to an identical RF without a prior clustering step, benefited less from the latter method. A critical aspect is that significant changes in the financial markets could alter the values of options, as we showed using different volatility data. This highlights the need for continuous monitoring and updating of the models to adapt to evolving market conditions. Future research could build upon these findings, trying to apply different clustering method on different dataset and to avoid data leakage issues if any, investigating methodologies for predicting implied volatility through the application of an implied volatility surface.

REFERENCES

Amilon, H. (2003). A neural network versus Black–Scholes: A comparison of pricing and hedging performances. Journal of Forecasting, 22(4), 317–335.

Aristotle, The Politics Book I.

Arthur, D., & Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (pp. 1027-1035). Society for Industrial and Applied Mathematics.

Bastos, J. A. (2024). Conformal prediction of option prices. Expert Systems With Applications, 245, 123087.

Bates, D. S. (1996). Jumps and stochastic volatility: Exchange rate processes implicit in deutsche mark options. The Review of Financial Studies, 9(1), 69–107.

Black, F & Scholes, M. (1973). The pricing of options and corporate liabilities. Journal of Political Economy.

Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123–140.

Breiman, L (2001). Random forests. Machine Learning 45(1), 5–32.

Breiman, L, Friedman, J., Olshen, R., & Stone, C. (1984). Classification and regression trees. Wadsworth Int. Group, 37(15), 237–251.

Corrado, C. J., & Su, T. (1996). Skewness and kurtosis in S&P 500 index returns implied by option prices. Journal of Financial Research, 19(2), 175–192.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems, 2(4), 303-314.

Gaspar, R. M., Lopes, S. D., & Sequeira, B. (2020). Neural network pricing of American put options. Risks, 8(3), 73.

Gradojevic, N., Gençay, R., & Kukolj, D. (2009). Option pricing with modular neural networks. IEEE Transactions on Neural Networks, 20(4), 626–637. https://doi.org/10.1109/TNN.2008.2011130.

Haykin, S. (2009). Neural networks and learning machines (3rd ed.). Pearson Education.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. The Review of Financial Studies, 6(2), 327–343.

Ho, T. K. (1995). Random decision forests. Proceedings of 3rd international conference on document analysis and recognition. Vol. 1. IEEE, pp.278–282.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. Neural Networks, 4(2), 251-257.

Hull, J. (2017) Options, Futures, and Other Derivatives 10th Edition, Pearson.

Hull, J., & White, A. (1987). The pricing of options on assets with stochastic volatilities. The Journal of Finance, 42(2), 281–300.

Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. The Journal of Finance, 49(3), 851–889. https://doi.org/10.2307/2329209.

Ivașcu, C.-F. (2021). Option pricing using machine learning. Expert Systems With Applications, 163, 113799.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980.

Lloyd, S. P. (1982). Least squares quantization in PCM. IEEE Transactions on Information Theory, 28(2), 129-137.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam & J. Neyman (Eds.), Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability (Vol. 1, pp. 281-297). University of California Press.

McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv preprint arXiv:1802.03426*. Retrieved from https://arxiv.org/abs/1802.03426.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5(4), 115–133.

Merton, R. C. (1973). Theory of Rational Option Pricing. The Bell Journal of Economics and Management Science, 4(1), 141–183. https://doi.org/10.2307/3003143.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386–408.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533–536.

Schroder, M. (1989). Computing the constant elasticity of variance option pricing formula. The Journal of Finance, 44(1), 211–219.

Wharton Research Data Services. "WRDS" wrds.wharton.upenn.edu

APPENDICES

TABLE IV DATASETS DESCRIPTIVE ANALYSIS

Calls IV Dataset

	K	IV	DTE	q	r	S	price
count	230,432	230,432	230,432	230,432	230,432	230,432	230,432
mean	4,020.82	19%	0.24	1.29%	0.87%	3,860.42	79.62
std	612.76	7%	0.21	0.41%	1.12%	551.07	145.14
min	1,000.00	7%	0.06	0.00%	0.06%	2,237.40	0.03
25%	3,550.00	14%	0.09	1.16%	0.13%	3,400.97	11.60
50%	4,095.00	18%	0.15	1.36%	0.24%	3,915.53	41.40
75%	4,480.00	22%	0.31	1.61%	1.19%	4,319.94	99.10
max	8,300.00	80%	1.00	1.90%	4.85%	4,796.56	3,651.25

Puts IV Dataset

	K	IV	DTE	q	r	S	price
count	230,432	230,432	230,432	230,432	230,432	230,432	230,432
mean	3,384.33	32%	0.26	1.29%	0.88%	3,860.42	61.72
std	722.45	14%	0.21	0.41%	1.13%	551.07	86.03
min	600.00	6%	0.06	0.00%	0.06%	2,237.40	0.03
25%	2,910.00	22%	0.10	1.16%	0.14%	3,400.97	10.10
50%	3,440.00	29%	0.17	1.36%	0.24%	3,915.53	34.30
75%	3,925.00	38%	0.35	1.61%	1.20%	4,319.94	82.60
max	7,600.00	80%	1.00	1.90%	4.85%	4,796.56	3,686.85

Calls 30d-HV Dataset

	K	HV30	DTE	q	r	S	price
count	230,432	230,432	230,432	230,432	230,432	230,432	230,432
mean	4,020.82	20.94%	0.24	1.29%	0.87%	3,860.42	79.62
std	612.76	14.47%	0.21	0.41%	1.12%	551.07	145.14
min	1,000.00	6.48%	0.06	0.00%	0.06%	2,237.40	0.03
25%	3,550.00	11.83%	0.09	1.16%	0.13%	3,400.97	11.60
50%	4,095.00	18.61%	0.15	1.36%	0.24%	3,915.53	41.40
75%	4,480.00	24.69%	0.31	1.61%	1.19%	4,319.94	99.10
max	8,300.00	97.56%	1.00	1.90%	4.85%	4,796.56	3,651.25

Puts 30d-HV Dataset

	K	HV30	DTE	q	r	S	price
count	230,432	230,432	230,432	230,432	230,432	230,432	230,432
mean	3,384.33	20.94%	0.26	1.29%	0.88%	3,860.42	61.72
std	722.45	14.47%	0.21	0.41%	1.13%	551.07	86.03
min	600.00	6.48%	0.06	0.00%	0.06%	2,237.40	0.03
25%	2,910.00	11.83%	0.10	1.16%	0.14%	3,400.97	10.10
50%	3,440.00	18.61%	0.17	1.36%	0.24%	3,915.53	34.30
75%	3,925.00	24.69%	0.35	1.61%	1.20%	4,319.94	82.60
max	7,600.00	97.56%	1.00	1.90%	4.85%	4,796.56	3,686.85

Calls 60d-HV Dataset

	K	HV60	DTE	q	r	S	price
count	230,432	230,432	230,432	230,432	230,432	230,432	230,432
mean	4,020.82	21.46%	0.24	1.29%	0.87%	3,860.42	79.62
std	612.76	13.70%	0.21	0.41%	1.12%	551.07	145.14
min	1,000.00	6.46%	0.06	0.00%	0.06%	2,237.40	0.03
25%	3,550.00	12.44%	0.09	1.16%	0.13%	3,400.97	11.60
50%	4,095.00	18.67%	0.15	1.36%	0.24%	3,915.53	41.40
75%	4,480.00	25.08%	0.31	1.61%	1.19%	4,319.94	99.10
max	8,300.00	75.46%	1.00	1.90%	4.85%	4,796.56	3,651.25

Puts 60d-HV Dataset

	K	HV60	DTE	q	r	S	price
count	230,432	230,432	230,432	230,432	230,432	230,432	230,432
mean	3,384.33	21.46%	0.26	1.29%	0.88%	3,860.42	61.72
std	722.45	13.70%	0.21	0.41%	1.13%	551.07	86.03
min	600.00	6.46%	0.06	0.00%	0.06%	2,237.40	0.03
25%	2,910.00	12.44%	0.10	1.16%	0.14%	3,400.97	10.10
50%	3,440.00	18.67%	0.17	1.36%	0.24%	3,915.53	34.30
75%	3,925.00	25.08%	0.35	1.61%	1.20%	4,319.94	82.60
max	7,600.00	75.46%	1.00	1.90%	4.85%	4,796.56	3,686.85

Figure X Call Options Data: Variable Distribution Overview

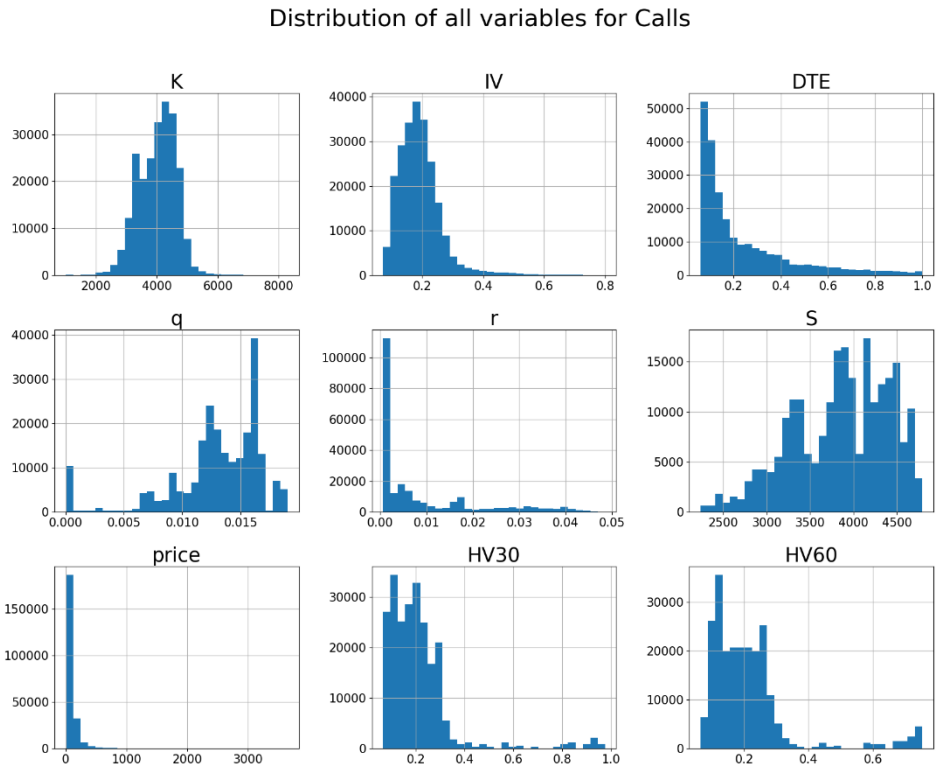
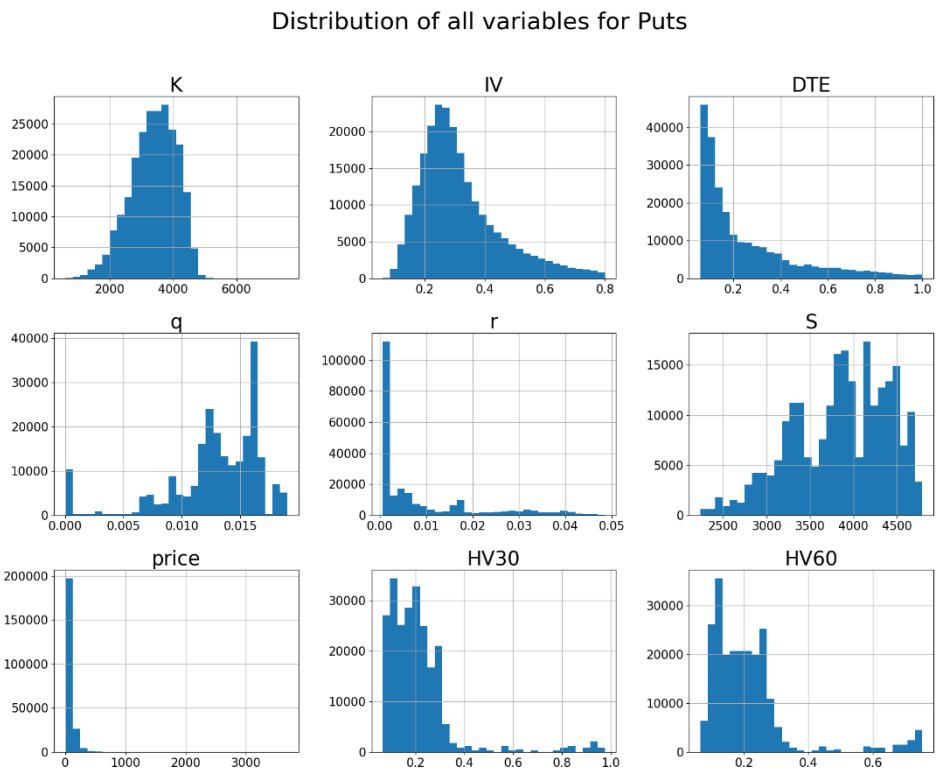


Figure XI Put Options Data: Variable Distribution Overview



DISCLAIMER

This master thesis was developed with strict adherence to the academic integrity policies and guidelines set forth by ISEG, Universidade de Lisboa. The work presented herein is the result of my own research, analysis, and writing, unless otherwise cited. In the interest of transparency, I provide the following disclosure regarding the use of artificial intelligence (AI) tools in the creation of this thesis:

I disclose that AI tools were employed during the development of this thesis as follows:

- AI-based research tools were used to assist in literature review.
- Generative AI tools were consulted for brainstorming and outlining purposes. However, all final writing, synthesis, and critical analysis are my own work. Instances where AI contributions were significant are clearly cited and acknowledged.

Nonetheless, I have ensured that the use of AI tools did not compromise the originality and integrity of my work. All sources of information, whether traditional or AI-assisted, have been appropriately cited in accordance with academic standards. The ethical use of AI in research and writing has been a guiding principle throughout the preparation of this thesis. I understand the importance of maintaining academic integrity and take full responsibility for the content and originality of this work.

Gabriele Loggia, 30-06-2024.