Erasmus School of Economics

Master Thesis: Quantitative Finance

Single Stock Options Implied Volatility (Surface) Modeling via Machine Learning

Name student: Shubham Rathi[1]

Student ID number: 573379

Supervisor: Dr. Gustavo Freire[2]

Second Assessor: Dr. Maria Grith[3]

Date final version: June 22, 2023

[1]Student M.Sc. Quantitative Finance at Erasmus School of Economics, Rotterdam, Zuid-Holland, the Netherlands

[2]Assistant professor at Erasmus School of Economics, Rotterdam, Zuid-Holland, the Netherlands

[3]Assistant professor at Erasmus School of Economics, Rotterdam, Zuid-Holland, the Netherlands

Abstract

This research explores the potential of Machine Learning in the domain of Implied Volatility Surface Modeling for Single Stock Options by fitting various models to the Implied Volatilities of cross-section of options for a cross-section of stocks. We investigate the modeling capabilities by using the options data of ten highly liquid stocks picked from various sectors. The main finding of this study is that Machine Learning methods, especially non-linear methods like Neural Networks, show great promise for the task of modeling the Implied Volatility Surface of Single Stock Options. They significantly outperform conventional parametric models like (Ad-hoc) Black Scholes Model. Furthermore, the predictive performance reaches a new high with 2-step approach where the modeling capabilities of non-linear Machine Learning methods are combined with economic rationale of parametric models. We observe that an underlying's previous day Implied Volatility Surface and 'VIX' are most influential covariates for the prediction task across all Machine Learning models.

Keywords

Options, Single Stock Options, Implied Volatility, Implied Volatility Surface, Implied Volatility Surface Modeling, Machine Learning, Hyperparameter Tuning, Ordinary Least Squares (OLS), Elastic Net, Regularization, Generalized Linear Model, Random Forest, Boosting, Neural Networks, Predictive Accuracy, Variable Importance, Parametric Models, 2-Step Approach

Contents

1 Introduction

In this paper, we are on a quest to model Implied Volatility (henceforth, IV) and subsequently Implied Volatility Surface (henceforth, IVS) of Single Stock Options (henceforth, SSO) using novel Machine Learning (henceforth, ML) methods. During this end-to-end ML analysis, we compare the ML modeling capabilities with that of traditional parametric models like (Ad-hoc) Black-Scholes Model (Black and Scholes (1973), Dumas et al. (1998)) and also with more recent 2-step approach where the ML model's predictive abilities are combined with the 'guidance' of economic reasoning from parametric models (Almeida et al. (2022)). Furthermore, all the (econometric and ML) methods used in our research are also used to model the IVS of S&P500 (ticker: SPX) options, thereby allowing us to see if SSO IVS can be modeled with same predictive accuracy as Stock Index Options (henceforth, SIO) IVS modeling accuracy despite former having relatively lower trading activity. Finally, as a simple attempt to interpret the complex 'black-box' ML models, we seek to understand the drivers of the IVS dynamics by finding the most relevant set of covariates for IVS modeling for SSO (and SPX options).

To allow reader to have a clear structure of this research and a fluid flow of information during the study, we briefly lay down the overarching setup of this paper. First, we have Introduction where we introduce key concepts foundational to our study and also motivate the relevance and need of this research. Second, we discuss the econometric and ML tools undertaken in this research in Methodology section. We then talk about the data, some summary statistics and feature set. Finally, we present the key findings and results of our research and conclude with some final remarks.

1.1 Implied Volatility

Options are financial derivatives which gives the option holder the right to buy (call) or sell (put) the underlying instrument at a fixed price, called Strike Price (K) at (European) / before (American) a fixed point in time in future called Expiration Date (T). An option's price is influenced by the following factors - current underlying price (S_t), strike price (K), time to expiry (τ), IV (σ), risk-free interest rate (r_f) and dividends (d). IV is a forward looking estimate of the uncertainty (volatility) of the underlying asset. Conceptually, it can be understood as the market's expectation of volatility in the underlying asset until the option's expiry. Technically, it is defined as the numeric value when plugged into the Black Scholes Option pricing model, along with other pricing inputs, returns the option price that is equal to the current option price in the market. This number is generally backed out by reverse solving the Black Scholes Model (henceforth, BSM) equation using numerical methods like Newton's method (Coleman and li (1970)). Among the pricing inputs described - the current underlying price (S_t), strike price (K), time to expiry (τ) and risk-free interest rate (r_f) can be observed in the market real-time while a decent estimate of dividends to be declared can be estimated by financial analysts. Meanwhile, IV is the only input that is not readily observed and also hard to estimate because IV is dynamic i.e. it changes with time as the market prices in new information. Hence,

it is often said that trading options is synonymous with trading IV because that is what an option's price boils down to. This bring us to the first aspect of our research's motivation that **Implied Volatility (IV) is the most important input for any option pricing model**. Furthermore, options play a pivotal role in financial markets. Options are important not only for participants like option traders actively trading them but also for other participants like financial and research analysts who may use metrics like open interest etc. to gauge future expectation of market and economy. Hence, given the huge size and relevance of options market, it becomes even more important to model IV of an option as accurately as possible.

1.2 Implied Volatility Surface

For any financial instrument whose options are traded on a derivatives exchange, they have multiple options listed and traded which differ in time to expiries (τ) and strike prices (or moneyness (m)[4]). Hence, at a point in time, we have a cross-section of options being traded for an underlying. Now, each option in the underlying has its own value of IV[5]. If we generate a 3-dimensional plot where we plot IV against moneyness and time to expiry, what we get is an IVS. IVS can be seen as a summarized and compact view of implied volatilities (henceforth, IVs[6]) of the cross-section of option. IVS is of great importance empirically for multiple reason. Firstly, from a trader's or portfolio manager's point of view, its cumbersome to track prices of all the options. Such a scattered distribution of information may hinder them in reacting to news in the market swiftly, especially given that speed is crucial in financial markets as the competition takes away any exploitable opportunities quickly. Second, the price of an option alone does not convey any useful information. Instead, IV is a much more useful metric to track as it gives information on the expected deviation of the underlying's movement. IVS combines the best of both worlds by displaying the IV information in a very compact way without any loss of relevant information. It is well known fact that (Bernales and Guidolin (2014)) participants who are actively engaged in options market, for instance, portfolio managers, traders, institutional investors, market makers etc. use IVS to track and manage their options position. This brings us to the second aspect with respect to the motivation of our research that **Implied Volatility Surface (IVS) is of great empirical importance in financial markets.** Given, this importance of IVS, it is crucial to accurately model the IVS, which also subsumes the task of modeling IV of a single option in the cross-section of options for an underlying.

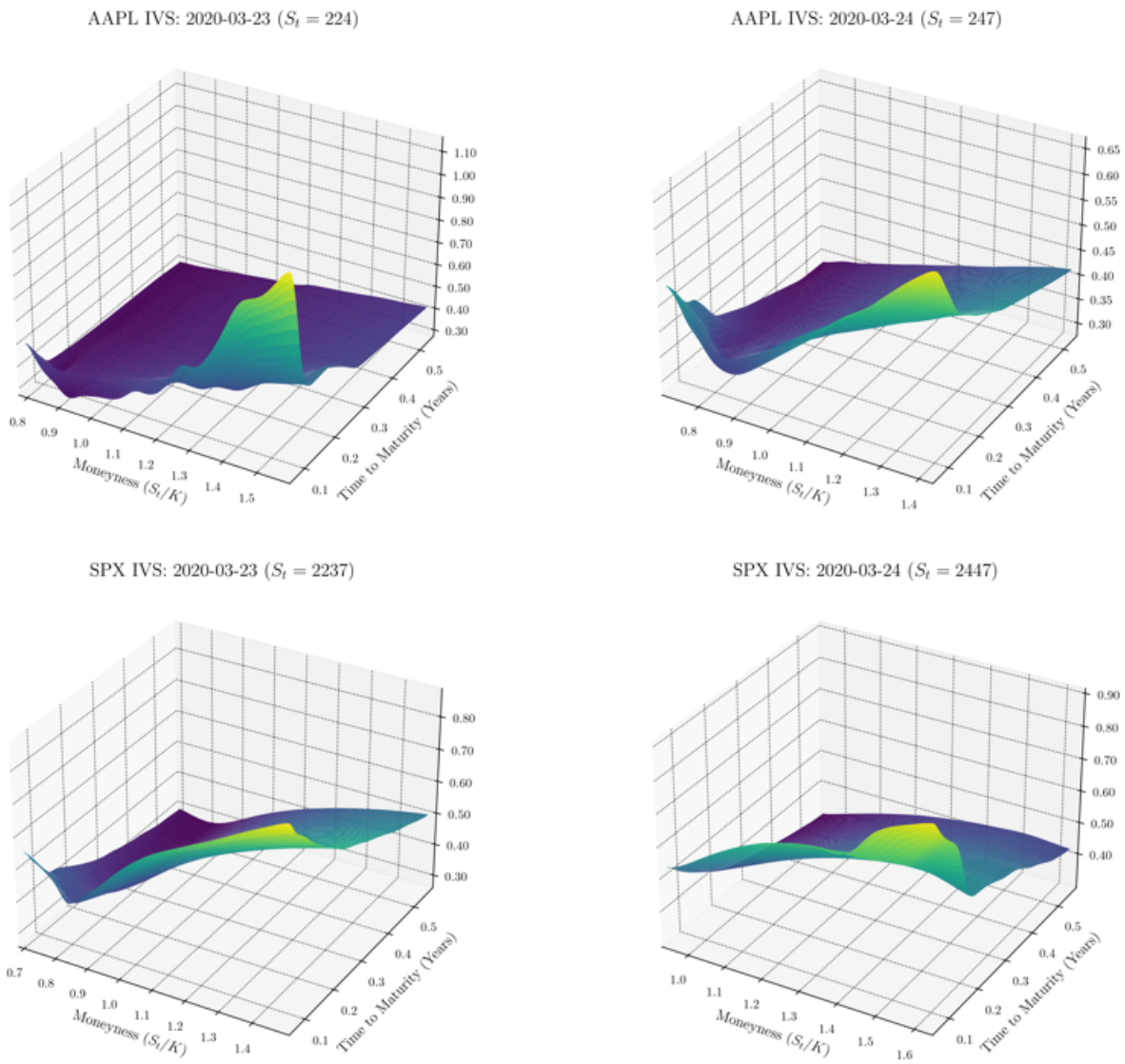
Figure 1 shows the IVS of Apple Inc. (ticker: AAPL) options and S&P500 index (ticker: SPX) options on two consecutive days during the COVID-19 crash in the financial markets. The key takeaway from the figure is that the IVS is dynamic and changes with time in addition to the fact that two different underlying generally have different IVS and potentially different (but possibly related) IVS dynamics.

[4]We define moneyness $m = S_t/K$ as done in Almeida et al. (2022), where S_t is the current underlying price and K is the strike price of option.

[5]This is because each option in the cross-section have a different market price, moneyness and time to expiration. This results in two of the inputs of an option pricing model being different (along with the option price i.e. output of an option pricing model) across the cross-section of options. Hence, the reverse engineered value of IV from an option pricing model also tends to differ across the cross-section of options.

[6]Not to confused with IVS - Implied Volatility Surface

Figure 1: IVS of Apple Inc. (AAPL) Options & S&P500 (SPX) Options on Two Consecutive Days

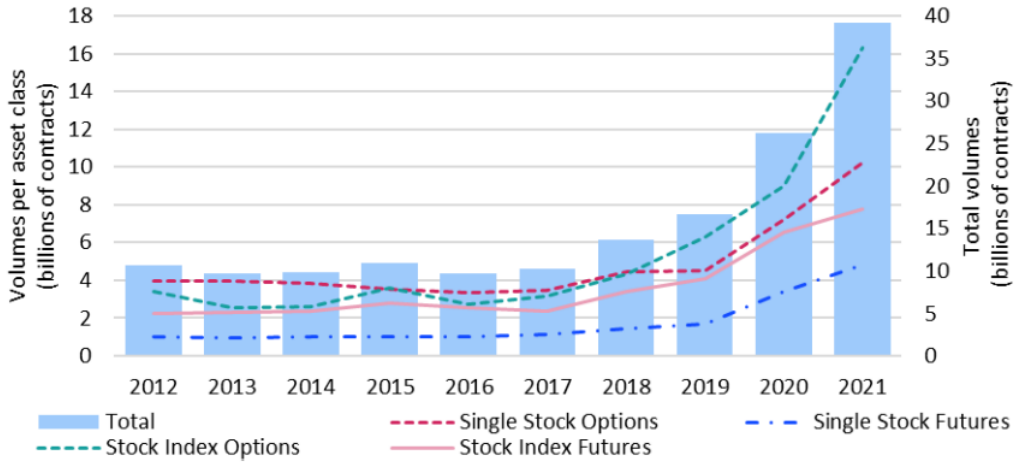


Note: The figure above shows the IVS of Apple Inc. (ticker: AAPL) options and S&P500 (ticker: SPX) options. The vertical axis in the subplots depict the value of IV for an option given its location (m, τ) in the cross-section. The IVS is being shown for two consecutive days - March 23, 2020 and March 24, 2020. March 23, 2020 was the day of biggest drop in S&P500 index during the of COVID-19 Financial Crash in 2020 which was followed by the biggest recovery in the index (during COVID-19 crash) on March 24, 2020. It can be seen that IVS is highly dynamic i.e. changes with time and that it is different for different underlyings - AAPL and SPX in this figure. Furthermore, a Gaussian Kernel smoothing algorithm has been applied to smooth out the IVS by removing distortions and spikes caused by lack of continuity in the IVS grid.

1.3 Why Single Stock Options?

SSO are a substantial category of financial derivatives. As per the World Federation of Exchanges (WFE) Derivatives Report 2021 - SSO saw a volume of 10.2 billion contracts globally (88% US), which was a 41% YoY increase from 2020. They are the second most exchange traded derivative product with 16% share (second only to stock index options which had a 26% share). SSO's importance can be attributed to their

Figure 2: Equity Derivatives Volume Evolution and Growth



Note: The figure has been sourced from World Federation of Exchanges (WFE) Derivatives Report 2021. The figure shows the volume evolution and trend over the last decade (ending 2021) for different segments falling under the exchange traded equity derivatives - SSO (our primary focus), SIO, Single Stock Futures and Stock Index Futures. It can be seen that both SSO and SIO have seen a major uptick surge in volumes traded especially since 2019. In 2021, SSO was the second most traded derivatives category in exchange traded equity derivatives and also overall exchange traded derivatives.

versatility as a derivative allowing for hedging, speculation, easy leverage etc. and hence are actively traded by market participants like market makers, asset managers, pension funds and many more for variety of intentions. This underscores the third aspect of our research's motivation that **Single Stock Options are the second most exchange traded derivative product** and hence cannot be ignored. Since, IV is the key input for option pricing and SSO being the second most exchange traded derivatives - it is paramount to model IV (and subsequently IVS) of SSO as precisely as possible. As will be covered shortly in more detail in Literature subsection that despite SSO being a substantial chunk of derivatives being traded on the exchange, there is little to no study exploring the use of ML methods to model the IVS of SSO.

Figure 2 shows the evolution of equity derivatives volumes over the last decade. It can be seen that volumes of SSO and SIO have exploded since 2019. In 2021, SSO was the second most traded derivative category in both overall exchange traded derivatives segment and exchange traded equity derivative segment[7], underscoring their importance and the urgent need to come with improved IVS models for SSO.

1.4 Why Use Machine Learning For IVS Modeling?

To appreciate the need of ML methods to model the IVS of SSO, we need to understand the flaws with currently used conventional models like BSM and how ML methods can overcome the pitfalls of conventional methods. The most prevalent option pricing model i.e. Black Scholes Model (Black and Scholes (1973)) makes an assumption that the IV of the underlying is constant across time and is same across the different moneyness and expirations of the underlying's options cross-section. On the contrary, it is a well known empirical fact

[7]Refer to the World Federation of Exchanges (WFE) Derivatives Report 2021 for more details on trading activity trends and statistics across different segments of exchange traded derivatives.

which is also backed by studies like Rubinstein (1994) that the IVs differ across the cross-section of options for a given underlying and is not static with time. Though there have been further variations and models developed after BSM (Black and Scholes (1973)) like Ad-hoc Black Scholes Model (henceforth, ADHBS) of Dumas et al. (1998), Heston model (Heston (1993)) and Carr and Wu model (Carr and Wu (2016)) etc. - and these model do take into account the changing dynamics of IVS that BSM ignores while delivering better IVS forecasts (as found in Almeida et al. (2022)) but still, these "improved" models don't deliver the desired level of accuracy that a concerned market participant would desire.

As discussed in Israel et al. (2020) that financial markets are notoriously noisy and conventional financial models built of traditional statistics may fall short in extracting the signal amidst noisy and complex data generating process (henceforth, DGP) of markets. Furthermore, given the availability of "big data" in today's world, some of the traditional statistical models may not even work. For instance, in case of Ordinary Least Squares, the model cannot produce parameters when number of observations (n) is less than the number of features (k) i.e. in language of matrices - the rank is not full ($n < k$). Furthermore, as discussed in Gu et al. (2020), given the complex nature of financial markets - it is essential to capture the "non-linear" and "interaction" effects which is simply ignored by the conventional financial models which are linear in nature and are capable of capturing only the "linear" effects of the market's complex DGP. This is where the modern ML methods come to our rescue as they are able to - capture non-linear and interaction effects inherent in a more complex DGP characterizing the financial (and options) markets, work in "big data" environment and also in situations where the number of features are (much) larger than number of observations, which can often be the case in financial problems (Israel et al. (2020)).

Therefore, in the context of IVS modeling for SSO, we describe the fourth and final aspect of our research's motivation that **ML models can help us generate superior predictive accuracy as compared to conventional parametric models** by - its ability to model non-linear and interaction effects in addition to linear effects, especially when non-linear dynamics exist in IVS (Bernales and Guidolin (2014)); deal with large dimension of data as for SSO, we have a cross-section of stocks and for each stock there are cross-section of options across moneyness and expiry horizons. The promise of ML methods for IVS modeling is underscored by Almeida et al. (2022) who see significant improvement in predictive accuracy of IVS by deploying feed forward neural networks.

To sum up the need of our research, we now know that (i) options play a significant role in financial markets and SSO are a sizable chunk of exchange traded derivatives market (second largest share) and hence pricing them accurately is crucial, (ii) IV is the most important ingredient for any option pricing model, (iii) IVS is much more practical and useful to model as they provide a richer information about underlying's distribution in a very compact and convenient manner and (iv) ML's superior modeling capabilities can allow them to model SSO's (complex) IVS (significantly) better as compared to currently used conventional parametric models in the industry.

1.5 Literature

Despite the evident necessity to model IVS of SSO using ML methods, this domain is not yet adequately researched. The fact that empirically IVS strongly contradicts the constant IV assumption of Black and Scholes (1973) is well established. As studied by Rubinstein (1994), this contradiction is backed by the existence of pronounced "skew" or "smile" displayed by the IVs of SPX options in the cross-section. Moreover, Andersen et al. (2015) establish that IVs of options change over time due to inherent highly nonlinear dynamics. There also has been research on how index options like S&P500's IVS are modeled using conventional and parametric methods (Goncalves and Guidolin (2006), Andersen et al. (2015), etc.) and (more recent) ML / non-parametric methods (Almeida et al. (2022), Garcia and Gencay (2000), Hutchinson et al. (1994)). Meanwhile, in the domain of SSO - IVS predictability is established by (Bernales and Guidolin (2014)) using the (conventional and parametric) Vector AutoRegressive (VAR) model and provides evidence for linkage between S&P500 options IVS and SSO IVS. Furthermore, Ang et al. (2012) show that past stock returns also predict SSO IVs.

Hence, the current state of literature in the domain of SSO IVS exploration shows that - (i) there is predictability in SSO IVS (established by Bernales and Guidolin (2014)), (ii) index IVS is vital for SSO IVS predictability (also established by Bernales and Guidolin (2014)) and (iii) firm characteristics can help in predicting SSO IVS (established by Ang et al. (2012)). But to my current knowledge, there is no study covering the use of (recent) ML methods to model SSO IVS. Israel et al. (2020) argue the superiority of ML in finance due to their ability to process large / high-dimensional data sets and model non-linear and interaction effects which are not possible by traditional statistical methods. Gu et al. (2020) show statistically and economically significant gains in stock return prediction exercise by ML models over conventional linear (regression) models.

Hence, this research where we apply ML methods, in the spirit of Gu et al. (2020) to model SSO IVS is interesting and relevant for both industry practitioners and academicians alike. It also stands necessary to explore the IVS of SSO using (recent and novel) ML methods given the role and weightage of SSO in the derivatives and overall financial markets.

1.6 Machine Learning Methods Used In Our Study

With an intention similar to that of Gu et al. (2020), we will be using the standard set of ML models that are part of graduate level ML textbooks encompassing from simple linear models like Ordinary Least Squares to complex models like Neural Networks. To be precise, we will be using Ordinary Least Squares, Elastic Net, Generalized Linear Model, Random Forest, Gradient Boosting and Neural Networks. These set of models represents a balanced panel of ML models for regression tasks. The motivation of selecting the above models will be discussed in more depth in Methodology. We exclude dimension reduction techniques like Principal

Component Analysis (PCA) as we will be working with both "reduced[8]" models and "full[9]" models, wherein reduced models partly serve the purpose of dimensionality reduction techniques. We also exclude Support Vector Regression (henceforth, SVR) due to the overlap of its predictive powers with abovementioned models (Gu et al. (2020)) and also for the reason that SVR are rather better suited for small-to-medium sized data sets as compared to large datasets (Joachims (2006)). For large datasets, SVR is inefficient and unstable due to kernel memory requirements scaling quadratically. We also exclude more advanced and deep learning models like LSTM, GAN etc. for the reason that the overarching goal and main focus of this research is to investigate if ML can be used to model IVS of SSO (and not to find the exact best ML model doing this task[10]). The above panel of ML models covers majority of predictive benefits that one can expect from the end-to-end breadth of ML models[11].

1.7 Main Empirical Findings

Our research is conducted by analyzing SSO of 10 highly liquid US stocks representing different sectors from 2019 to 2021 (included). Our feature set has 25 variables for each option in the cross-section of options (for all 10 stocks). We list down the following empirical observations from using ML models described previously to model SSO IVS.

ML significantly outperforms traditional parametric models like (Ad-hoc) BSM in the task of SSO IVS modeling. This is the most relevant finding of our research underscoring the great promise that ML models behold for IVS modeling task. In our study we see that ML models significantly outperform traditional parametric models like BSM and ADHBS for IVS modeling (both for SSO and SPX IVS modeling). This shows that the ML models can be deployed for IVS modeling, especially for SSO. This outperformance stems from ML models ability to process big data and capture non-linear and interaction effects (in addition to the linear effects) embodied in the complex dynamics of IVS DGP.

ML models capable of capturing both non-linear and interaction effects outperform in the task of IVS modeling. Given the complex nature of IVS dynamics and its DGP, it intuitively makes sense that "complex" ML models like Random Forests and Neural Networks are likely to outperform "simpler" models like linear Ordinary Least Squares model with respect to IVS modeling accuracy for SSO (and SPX options). This can be attributed to former's tendency to catch the complexities of DGP which is simply neglected by simple linear models.

Shallow learning outperforms deep learning. Similar to the findings of Gu et al. (2020) and Almeida et al. (2022) , we observe in our research too that the Neural Networks with one hidden layer (shallow) outperform other Neural Network with up to five hidden layers (deeper). This empirical finding

[8] By "reduced" we refer to ML models that were trained on selected set of "important" features only selected by Elastic Net.
[9] By "full" we refer to ML models that were trained on all features.
[10] Doing so is a never ending "rabbit hole" task especially with new "state-of-the-art" models coming out every now and then. Furthermore, this approach of fitting and refining models until a desired accuracy is reached is prone to overfitting and p-hacking.
[11] This also helps to conduct research within the computational constraints and abiding the Thesis timeline.

reiterates the common belief in ML community that simpler models generally tend to outperform more complex models especially in the environment of low signal-to-noise ratio (SNR) characterizing the financial (and option) markets dynamics. Furthermore, the best performing Random Forest and Gradient Boosting models contained trees of shallow depth (< 3).

Parsimonious models have a higher predictive accuracy for IVS modeling. While modeling the IVS of SSO and SPX options using ML, for each model we deployed two variants - "full" model having all the features and the "reduced" model having selected features only (dimension reduction). In both SSO and SPX IVS modeling, the reduced models produced higher predictive accuracy as compared to the full model counterparts. This was observed across all the ML methods considered in our study. This can be seen as indication that more complex models generally tend to overfit and subsequently suffer during out-of-sample prediction performance.

The 2-Step approach was the most successful model in the IVS modeling task. Almeida et al. (2022)'s 2-Step approach was also compared to the standalone ML models for Apple Inc.[12] (ticker: AAPL) to see if they bring in extra predictive power to the table. We observe that the 2-Step approach modeled IVS with highest accuracy for AAPL options (and SPX options). This finding may imply that combining economic rationale guidance from parametric model with modeling capabilities of ML (non-parametric) models can bring out synergies and generate superior predictions for financial modeling problems. The superiority manifests at least in the domain of IVS modeling for SSO (and SPX options).

SPX IVS have higher predictability than SSO IVS. All the ML models used in our study produced relatively better forecasts for the SPX IVS as compared to SSO IVS. This empirical finding can be intuitively expected because SPX options are much more liquid than SSO. This liquidity implies that more information is priced in the SPX options, allowing for better forecasts compared to SSO where the latter is generally riddled with idiosyncratic noise. Though higher mispricings in SSO can lead to potentially bigger and more profitable opportunities.

Previous day (underlying's) IVS, VIX and moneyness are the strongest predictors for SSO IVS modeling. Among 20+ predictors utilized in our study, spanning from option specific features like greeks, SPX options data, stock specific data and macroeconomic variables - Previous day (underlying's) IVS was by far the biggest predictor of IVS. This empirical finding aligns with the finding of Bollerslev (1986) stemming from high autocorrelation in volatility especially at short lags. Similar to findings of Almeida et al. (2022), macroeconomic variables didn't play a major role in IVS prediction task, possibly due to the fact that markets are forward looking and most of the relevant and expected information is already priced in the market (efficient market hypothesis) and therefore also in the underlying's previous day IVS.

[12]Since each underlying requires a separate ADHBS model to be fit to it, we only use Apple Inc. out of the 10 single stocks as it is the most liquid stock with richest option data in our study. This allows us to draw most accurate conclusions about our comparison task of 2-Step vs. standalone ML as more data implies lower standard errors and more stable and reliable results.

2 Methodology

In this section we turn to discuss the assortment of ML (econometric) techniques that we selected to undertake our SSO IVS modeling analysis[13]. In addition to talking about the models in turns briefly, we also touch upon other design choices involved in a typical end-to-end ML analysis in the succeeding subsections. We purposefully keep the description of all ML models and design choices succinct as we do not intend to reinvent the wheel and refer the reader to Gu et al. (2020) where they can find in-depth mathematical and conceptual description along with computational algorithms for each ML model used in our study.

As discussed, along the lines of Gu et al. (2020), we intend on using a mix of linear models, non-linear models and novel ML techniques. The motivation of using this mix is underscored by diversified ways of modeling covariates that is allowed by each model type for prediction purposes. This allows us to see which effects are more dominant in our IVS prediction exercise for SSO. For instance, Ordinary Least Squares (henceforth, OLS) allows for only linear effects to be modeled, Elastic Net (henceforth, ENet) allows us to see the impact of bias-variance trade-off, Generalized Linear Model (henceforth, GLM) additionally allows for non-linear effects to be modeled and (novel) ML methods like Random Forest(henceforth, RF), (Extreme) Gradient Boosting (henceforth, XGB) and Neural Networks (henceforth, NN) allows for non-linear and interaction effects to be modeled too.

We make some design choices for our ML methods while adapting Gu et al. (2020) as our baseline. First, all models need an objective function to estimate parameters using analytical or numerical (optimization) approach. All studied models share the same objective function to minimize the mean squared prediction error (henceforth, MSE). Second, we utilize regularization, a method to purposefully reduce the complexity of model (through hyperparameter tuning, where applicable) to see potential benefits of bias-variance trade off in out-of-sample (henceforth, OOS) forecasting performance by reducing overfitting tendencies of ML models (through reduced complexity of model). We discuss other design choices undertaken in our study in subsections to follow.

Similar to Gu et al. (2020), in its most general form, we describe a SSO's predicted IV (σ_i) in the cross-section of options as

$$\sigma_{i,t+1} = E_t(\sigma_{i,t+1}) + \epsilon_{i,t+1}, \tag{1}$$

where $E_t(\sigma_{i,t+1}) = f^*(z_{i,t})$, in which f^* is a flexible function of the P-dimensional predictor variables ($\vec{z}$)[14], which can be either parametric or non-parametric. Options are indexed as i[15] $= 1, ..., N_t$[16] and days by $t = 1, ..., T$. The computational goal here is to produce the expression for the functional form $f^*(.)$ using

[13] All the analysis is carried out in Python3 programming language using standard ML and scientific libraries.

[14] In the rest of the paper the vector ($\vec{z}$) is denoted by $z_{i,t}$. It will be discussed in Data section in more detail that $z_{i,t}$ consists of (i) option specific feature which is different for all options in the cross-section, (ii) stock (underlying) specific features that are same for all options in a stock's cross-section and (iii) SPX (index) specific features and Macroeconomic indicators which are same for all the options in the cross-section across all the stocks on a given day t.

[15] On a given day t, i indexes all the options in the cross-section across all the stocks.

[16] The number of options (N_t) in the total cross-section (for all stocks) on each day t can be different.

our covariates. Similar to Gu et al. (2020) our functional form $f^*(.)$ is estimated within a set of constraints. The expression of $f^*(.)$ is not option or timestamp specific i.e. independent of i and t. This constraint is imposed because we want our ML methods to produce a generalized model with generalized IVS dynamics insights that can be applied to any option at any time. Though we want to highlight that $f^*(.)$ is produced only using $z_{i,t}$ as the only input to models, implying that the ML model would not draw information from any other option or time period while estimating $f^*(.)$ for an option at a particular timestamp.

2.1 Model Training and Validation-Based Hyperparameter Tuning

Here we discuss training methodology, which is an influential step in any ML workflow. We will split our sample into the conventional train-validate-test disjoint sub-sample data sets. Our dataset is 3 year long where the first year of dataset will be used to train the model. The second year of dataset is the validation set which will be used for hyperparameter tuning[17] (where applicable). After hypertuning, we retrain the model on first 2 year of dataset i.e. combined train and validation set. Lastly, we evaluate our model on the "unseen" test set which is the data from last year of our dataset. We do not perform K-fold Cross Validation for hyperparameter tuning because we are working with time-series data which needs the temporal ordering of data to be maintained. Instead we use Walk Forward Validation (Cerqueira et al. (2017)) where we feed in more data during each iteration (fold) on a rolling basis and use the new data as validation set in current fold and the validation data from previous fold is appended to train set of current fold.

2.2 Variable Selection

Variable selection allows us to reduce the dimensionality of feature set (and subsequently the complexity of a model) by selecting the most important covariates as per the dimension reduction technique applied. This reduced dimensionality helps to build a simpler ML model, which is desirable because, generally speaking, simple and parsimonious ML models outperform complex ML models (Domingos (2012)). We implement an ENet for variable selection due to benefits mentioned in Nicolai Meinshausen and Peter Bühlmann (2010). We intend to select top five relevant variables. We discuss about ENet in more depth shortly.

2.3 Linear Model: Ordinary Least Squares

The first (ML) model we use is a linear OLS model to demonstrate the predictive power of an (unregularized) linear regression model. Due to the functional form f^* of an OLS model, only linear effects are captured and we expect it to perform the least amongst all the ML models because of - (i) potential high dimensionality of data making OLS unstable or even infeasible when number of features (k) is greater than

[17] Hyperparameters are parameters that are not learned as part of estimated functional form f^* for a ML model but guide the learning process. Some of these hyperparameters (may) control the extent of regularization. For instance, in SVR the kernel to be used dictates the kind of Support Vector Regressor we build (e.g. linear, gaussian) whereas the width of error margin around separating hyperplane (ϵ) which also is a hyperparameter controls the extent of regularization (bias-variance trade-off).

number of observations (n) i.e. $k > n$ and (ii) linear effects on its own failing to explain the complexity of DGP.

Model: The parameters of functional form f^* of OLS model are estimated linearly using the covariates as it is. Mathematically, the model is described as

$$f^*(z_{i,t};\beta) = z'_{i,t}\vec{\beta}, \tag{2}$$

where $z_{i,t}$ is the covariate vector for option i at timestamp t and $\vec{\beta}$ is the vector of estimated parameters.

Objective Function: The parameter vector $\vec{\beta}$ is estimated by solving the following objective function

$$\mathcal{L}(\beta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\left(\sigma_{i,t+1} - f\left(z_{i,t};\beta\right)\right)^2 \tag{3}$$

analytically or numerically. The objective function aims to find best-fitting linear (hyper)plane that minimizes the sum of squared vertical distances from the linear (hyper)plane.

2.4 Regularized Linear Model: Elastic Net

ENet is a regularized linear regression model to show the potential benefit of bias-variance trade-off in the out-of-sample forecasting performance. It is likely to outperform the (unregularized) OLS model. However, the modeling capabilities of ENet are restrictied to capturing linear effects only and just like OLS, ENet also ignores the non-linear and interaction effects due to the nature of its functional form f^*. We select ENet as the regularized linear model because it combines the benefits of both lasso ("l_1") and ridge ("l_2") regularizers[18].

Model: Mathematically, ENet has same model description as OLS equation 2. Despite the same model formula, the parameter vector $\vec{\beta}$ differs for both due the regularization term in ENet's objective function.

Objective Function: ENet regularizes the vanilla OLS model by adding a penalty for the coefficient vector $\vec{\beta}$ using both "l_1" and "l_2" penalty in its objective function which modifies the loss function to

$$\mathcal{L}(\beta;\cdot) = \mathcal{L}(\beta) + \phi(\beta;\cdot), \tag{4}$$

where $\mathcal{L}(\beta)$ is same as the loss function of OLS as described in equation 3 and $\phi(\beta;\cdot)$ is the regularization penalty term defined as

$$\phi(\beta;\lambda,\alpha) = \lambda(1-\alpha)\sum_{j=1}^{P}|\beta_j| + \frac{1}{2}\lambda\alpha\sum_{j=1}^{P}\beta_j^2, \tag{5}$$

where α is the mixing parameter determining the proportion of lasso (l_1) shrinkage and ridge (l_2) shrinkage,

[18] Refer to Gu et al. (2020) or any graduate level ML textbook for the advantages and disadvantages of both "lasso" and "ridge" regularizers.

λ determines the overall degree of regularization and β_j is the coefficient of j^{th} covariate. The objective function is generally solved using numerical methods like descent algorithms.

2.5 Generalized Linear Model

Next ML model in our toolkit is GLM. The version of GLM that we adopt is 2^{nd} order polynomial. This version of GLM allows us to not only model the linear effects but also is capable to incorporate the non-linear effects too. But unlike more complex ML models like RF and NN, it can't capture the interaction effects. Since the IVS has non-linear dynamics, we would expect 2^{nd} order polynomial GLM to outperform linear models. We purposefully don't include cross-interaction terms between covariates in our GLM's covariate set because we want to see how much extra predictability comes by adding non-linear effect only[19]. If the predictive gains are small then it would imply that interaction effect is the most dominant in IVS DGP.

Model: The functional form f^* of GLM is same as OLS in a sense that both are linear in nature. The differentiating element is the inclusion of 2^{nd} degree polynomial transformations of covariates ($z_{i,t}^2$) as well in the overall covariate set of GLM. Hence, the 2^{nd} order polynomial GLM takes the form

$$f^*(z_{i,t};\beta) = z'_{i,t}\vec{\beta_1} + (z_{i,t}^2)'\vec{\beta_2}, \tag{6}$$

where the nonlinear terms ($(z_{i,t}^2)'\vec{\beta_2}$) allows us to induce non-linear effect without including interaction effect between the predictors. Hence, GLM will allow us to see if non-linear effects (which linear models ignore) bring significant predictive power to the table for SSO IVS modeling.

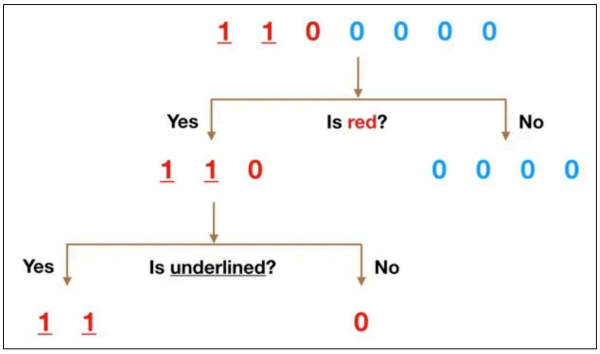
Objective Function: GLM's loss function is similar to OLS's loss function as described in equation 3 because GLM in its functional form f^* is same as OLS with only difference being that GLM's predictor set also includes the 2^{nd} order polynomial transformations of covariates ($z_{i,t}^2$). Similar to OLS, the objective function here can be solved both analytically and numerically.

2.6 Random Forests and (Gradient) Boosted Regression Trees

In the previous subsection, we explained GLM's capability in capturing the nonlinear effects of predictors. However, it doesn't model the interaction effects among them. Regression (decision) trees bypass this pitfall as it is capable of modeling both non-linear and interaction effects (in addition to linear effects). Trees are non-parametric methods, unlike traditional statistical models that are parametric in nature. Simply speaking, as the name "Decision Tree" suggests - the model tries to find similar set of observations by putting them into different buckets based on some set of criteria (decisions). The model takes such decisions multiple times sequentially and hence, forming a tree where the end nodes (leaves) contain "similar" observations (because

[19] Additionally, if we include the interaction terms then the number of covariates will grow with the order $\mathcal{O}(k!)$ where k represents the degree of polynomial. Such a large set of covariates may render the GLM unstable and even infeasible (because GLM shares same functional form as OLS and OLS becomes infeasible when number of features exceeds the number of observations i.e. rank is not full). Also, we restrict ourselves to 2^{nd} degree polynomial of GLM to avoid overfitting tendencies due to larger covariate set resulting from higher order polynomial as studied in Bishop (1995).

Figure 3: Simplified Example of a (Decision) Tree



Note: The above figure is a simplified visual of how a tree works post training (refer to Gu et al. (2020) for exact details on how a tree is trained). Each new observation goes through a sequence of decision splits until they reach a leaf node where there are no further splits. Each leaf node can be seen as a group of "similar" observations. Source: Geron (2019).

they went through same decision filters). For regression purposes, a new observation is passed through a (trained) Regression Tree where it goes through same sequential decisions until it reaches the leaf containing similar set of observations and then the predicted value for the new observation is the mean of (trained) observations in that leaf[20].

Regression (decision) trees have many benefits. First, there is no feature processing needed to train trees. Second, trees are one of the few ML models that are considered to be at "white box" model because they are very simple to understand and interpret. It is easy to see the calculations and decisions taken by a tree. But trees non-parametric nature that gives trees its many strength is also one of its biggest disadvantage. Their fully non-parametric nature implies that they are not bounded (no assumptions made about its functional form f^*) and hence, if trees are left unregularized, they can possibly grow a tree deep enough that they fit each observation separately leading to high degree of overfitting. Hence, trees when used should be appropriately regularized. In our analysis we use RF and XGB - ML models built on trees[21] as they are empirically known to outperform trees (Geurts et al. (2002)). To understand these methods we briefly describe what "Ensemble" means in the context of ML.

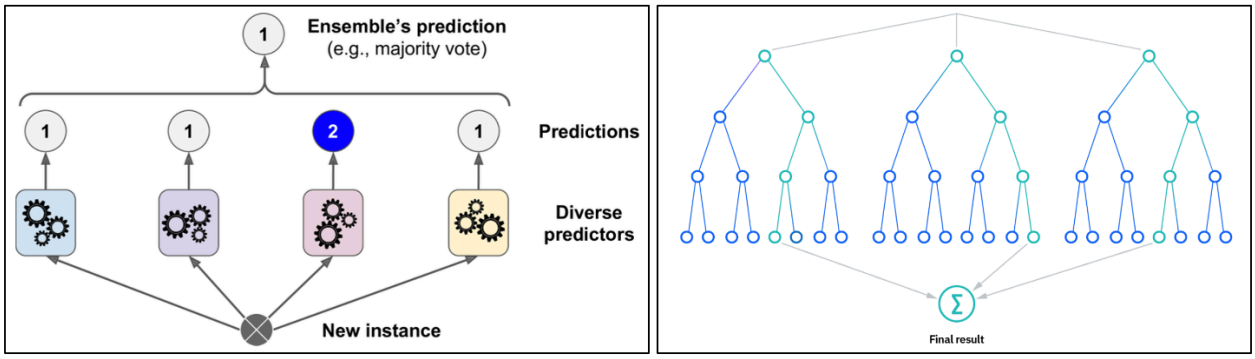
Ensemble Learning is a technique of aggregating predictions from many "weak[22]" learners (i.e. models) to arrive at single aggregated prediction. The idea of ensemble is somewhat similar to the concept of "Law of Large Numbers". As described in Geron (2019), if each of the weak learner has a predictive accuracy of 51% and we get 1000 of such weak learners then the aggregated predictive accuracy will be very close to 75% because the standard error of aggregated prediction will be much lower than prediction from a single

[20]We refer the reader to Gu et al. (2020) or any graduate level ML textbook for more details on Regression Trees regarding their functional form f^*, their objective function that guide in optimal split calculation, and other hyperaparameters (criteria to measure quality of split (e.g. gini impurity for classifiers, MSE for regressors), maximum depth of tree, minimum number of samples in a node needed for further decision based splitting, penalty on tree depth etc.) dictating the learning process of a tree. In addition to some of the hyperparameters described above, there are other methods like pruning to regularize trees.

[21]Similar to Gu et al. (2020) we will be using binary trees for simplicity reasons.

[22]More precisely, a "weak" learner is a predictor that does slightly "better" than random guessing i.e. the prediction accuracy of learner is slightly above 50%.

Figure 4: Simplified Representation of Ensemble Learning (left) and Random Forest (right)



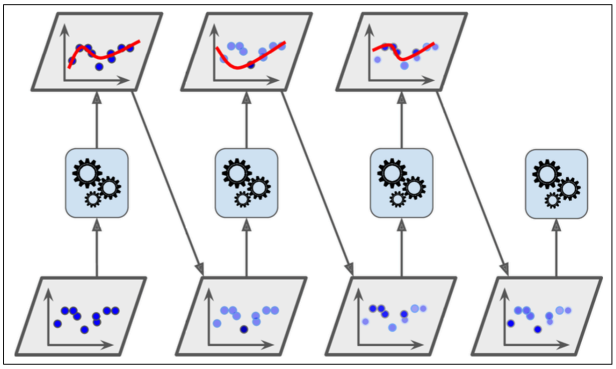
Note: The above figure shows a simplified conceptual representation of Ensemble Learning (left) and Random Forest (right). For Ensemble Learning we see that (post training) each new observation is separately predicted by "diverse" predictors in ensemble and are aggregated using an apt method (for e.g. majority voting for classification, average for regression) to arrive at a single (aggregated) prediction. For a trained RF, an unseen new observation is predicted by each (bootstrapped) tree in the forest and then the predictions from all the trees are aggregated as done for any ensemble learning method. Source: Geron (2019), IBM.

learner[23]. It is important to note that to utilize the benefit of ensemble learning, the number of learners should be large and (roughly) independent of each other (as is the requirement for Law of Large Numbers to work). Generally speaking, a prediction from ensemble of learners can outperform a single "strong" learner given the aforementioned conditions of ensemble of learners are satisfied. Bagging, boosting, stacking etc. are some of different methods to perform ensemble learning. Similar to trees, ML models based on ensemble learning techniques also need to be hypertuned and regularized in an appropriate manner.

Bagging - Random Forests (RF): Bagging stands for (B)ootstrap (agg)regation. Boostrap is technique to independently sub-sample data from given sample of data. Random Forest (RF) is a ML model that utilizes a modified version of bagging. Forest as we know (with slight abuse of definition) is a large group of trees. Now we know that if we can aggregate the prediction from ensemble of many regression trees (collectively representing a "forest") we can get a prediction which is superior than prediction from a single tree. But as described above, for ensemble to lend its predictive benefits we need the individual trees (learners) to be sufficiently diverse. To achieve this diversity (independence), we do two things. First, we induce independence amongst trees in the ensemble by differentiating them by the samples on which they are trained and this differentiation is brought in by bootstrap sampling. The idea is that if the trees are trained on (somewhat) different and independent samples, then trees train independently (due to possibly different decision splits) and subsequently produce independent predictions. Second, to further ensure the de-correlation among trees, each tree makes it decision splits based on random sub-sample of features i.e. we also bootstrap the feature set and therefore do not utilize all the features to build and train our trees and this process in RF's implementation is what modifies it from literal bagging implementation. Finally, we aggregate the predictions

[23]Standard error (σ_n) decreases with increasing sample size "n" (here, number of "weak" learners) with a factor $1/\sqrt{n}$ i.e. ($\sigma_n \propto 1/\sqrt{n}$), given the samples (learners) are independent of each other.

Figure 5: Simplified Representation of Boosting



Note: The above figure shows a simplified conceptual representation of Boosting. Here we see that (starting from left to right) how the first learner in the iteration trains and predicts on original sample and then its predictive errors are given more weight (emphasis) by the learner in next iteration. The process repeats itself until the specified number of learners in the ensemble are reached. The general underlying idea is that many "weak" learners when trained sequentially by improving the errors of previous learner collectively form into a "strong" learner as an ensemble. Source: Geron (2019).

from these many "independent" (random) trees to get a single aggregated prediction defined as

$$\hat{y} = \frac{1}{K}\sum_{k=1}^{K} f_k(z_{i,t}), \tag{7}$$

where $f_k(z_{i,t})$ represents the prediction of the k-th tree. Hence, the name Random Forest for this ML model as it is a ML model based on ensemble learning principles by aggregating the predictions from many trees which are pseudo-independent due to bootstrap sampling technique applied to both data and feature set used to train trees in a forest's ensemble. In addition RF have the tendency to prevent overfitting, hence, making it robust by generalizing to new data and generate superior predictions.

Boosting - Extreme Gradient Boosting (XGB): Boosting is an ensemble technique where a "strong" learner is built by sequentially stacking multiple "weak" learners where each weak learner corrects the predictive errors of the previous learner by giving them more weights (emphasis) in next iteration. Boosting has further sub-types but we focus on gradient boosting[24] where the current learner in the ensemble models the residual errors from previous learner. This is the core difference between boosting and bagging where the former relies of training each learner in the ensemble sequentially (dependent on previous learner) whereas the latter relies of training each member in the ensemble independently[25]. Among the models falling under the (gradient) boosting category, we select XGB (or XGBoost) due to its empirical success in ML workflows (Geron (2019)) and its optimized implementation.

[24]Due to its empirical success as also noted in Geron (2019).

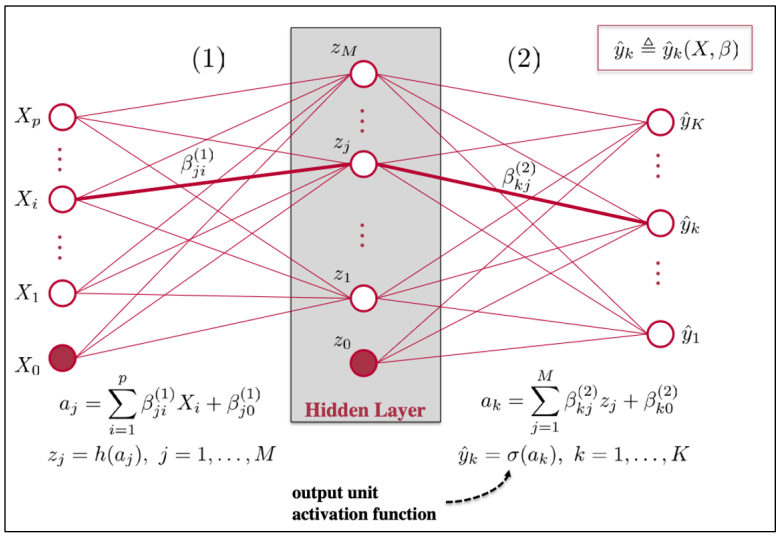
[25]Another key thing to note is that the training times of RF are generally faster than XGB because the trees in RF take benefit of "parallelization" where the trees can be trained independently of each other in parallel on different cores of CPU where has in XGB each learner has to be trained sequentially as they depend on predictive errors from previous learners.

2.7 Neural Networks

Neural Network (NN) is the final non linear ML method to be used in our research. NNs are considered as one of the most powerful tools in ML, as they can theoretically model any smooth predictive relationship and hence, are also termed as "universal approximator" (Hornik et al. (1989)). NNs are known for their ability to model complex relationships, which is attributed to their ability to stack and integrate multiple sequential layers of nonlinear predictor interactions, and hence, are often also referred to as "deep learning". Similar to RF and XGB, NN are also capable of capturing linear, non-linear and interaction effects. However, despite their modeling capabilities, NNs are considered one of the least interpretable and highly parameterized ML methods available with tendency to easily overfit, primarily due to their complexity. Similar to Gu et al. (2020) we also utilize "feed-forward" NNs.

As depicted in Figure 6, the feed forward NNs have an "input" layer, (multiple) "hidden" layers and an "output" layer. The input layer takes in the features at input nodes. Then a linear combination of the values of nodes from previous layer and a bias term is taken (with respective weights β_i of each edge connecting two different nodes at adjacent layers) which is further transformed with an activation function used at the nodes of next layer. This process is sequentially repeated from input layer to output layer while iterating through all the hidden layers which results in an output at output layer which is a non-linear combination of features at input layer intertwined with interactions between them. The activation functions induces non-linearity through its non-linear transformations and the recursive linear combinations of nodes at each layer builds

Figure 6: Complete Network Representation of a Feed Forward Neural Network



Note: The above figure shows a network representation of a feed forward neural network with a single hidden layer. The input layer takes in features (X_i) as a P-dimensional input. These features are aggregated (as a_j) at all hidden nodes (z_i) present in the hidden layer with respective weights ($\beta_{ji}^{(1)}$) along with a bias term (X_0). This aggregation (a_j) is transformed with an activation function (h^*). The same process is repeated by feeding the values of nodes of current hidden layer (z_i) and aggregating it at the next forward layer (y_i) with respective weights ($\beta_{kj}^{(2)}$) and transforming them with output unit's activation function (σ^*) to produce the final output predictions. Source: Hastie et al. (2009).

the interaction effect by creating interaction terms between different predictors at each layer.

Model: The functional form f^* of a feed forward NN can be mathematically be represented as

$$f^*(z_{i,t};\beta) = \sigma\left(\sum_j \beta_{kj}^{(\ell)} h\left(\sum_s \beta_{js}^{(\ell-1)} h\left(\dots h\left(\sum_k \beta_{jk}^{(1)} z_{i,t_k}\right)\right)\right)\right), \tag{8}$$

where σ is the output activation function, h is the activation function at hidden layers, β(s) are the weights of edges connecting nodes at adjacent layers. For any NN, we have to make several choices[26] and the large number of customization possible partly represent their complexity too. Among many available choices of activation functions, we use ReLU (Rectified Linear Units) as they are a common choice in ML community due to their empirical success. ReLU is defined as

$$\text{ReLU}(x) = max(0, x). \tag{9}$$

Furthermore, we follow Gu et al. (2020) and Masters (1993) by considering NNs with up to five hidden layers and choosing number of neurons at each hidden layer using geometric pyramid rule. The rule implies that the first (hidden) layer will have 32 neurons (nodes), second layer will have 16 neurons and so on (i.e. half the number of neurons at each subsequent layer).

Objective Function: Similar to OLS, a NN's objective function is to minimize the MSE i.e. "l_2" penalty of prediction errors. Along with Stochastic Gradient Descent (henceforth, SGD), the famous "backpropagation" algorithm is used to numerically solve the objective function which combines the principles of "chain rule" from derivative calculus and "dynamic programming". The gradients of weights are calculated backwards i.e. from output to hidden layer, propagating the predictive errors from output layer to input layer, hence, the name backpropagation.

2.8 Evaluation Metrics

To assess a model's predictive accuracy for our SSO IV(S) predictions we compute two performance metrics. One is OOS R^2 (henceforth, R^2_{oos}) in spirit of Gu et al. (2020) and Implied Volatility Root Mean Squared Error (henceforth, IVRMSE) in spirit of Almeida et al. (2022). We compute the R^2_{oos} as

$$R^2_{\text{oos}} = 1 - \frac{\sum_{(i,t)\in\mathcal{T}_3}\left(\sigma_{i,t+1} - \widehat{\sigma}_{i,t+1}\right)^2}{\sum_{(i,t)\in\mathcal{T}_3}\left(\sigma_{i,t+1} - \bar{\sigma}_{i,t+1}\right)^2}, \tag{10}$$

and IVRMSE as

$$\text{IVRMSE} = \sqrt{\frac{\sum_{(i,t)\in\mathcal{T}_3}\left(\sigma_{i,t+1} - \widehat{\sigma}_{i,t+1}\right)^2}{|\mathcal{T}_3|}}, \tag{11}$$

[26]Since NN are a big and complete topic in itself, we refer the reader to Gu et al. (2020) and Hastie et al. (2009) for the pros and cons of many design choices we make (for instance, the benefits of using ReLU as activation function).

with $\mathcal{T}_3$ denoting the observations in the testing sample and $|\mathcal{T}_3|$ denoting the number of observations in the testing sample.

To check if the OOS forecasts from a (ML) model are (statistically) significantly different from another model's OOS forecasts, we conduct Diebold and Mariano (1995) test (henceforth, DM test) on a pairwise basis for each pair of models. We borrow Gu et al. (2020)'s adaptation of DM test to uphold the weak error dependence condition for our options dataset that is potentially characterized by strong correlation in predictive errors in the cross-section. Hence, we also modify the DM test by comparing the cross-sectional average of predictive errors instead of comparing the individual forecast errors. The formula for test statistic of DM test between two different models is given as $DM = \bar{d}/\widehat{\sigma}_{\bar{d}}$ where $\bar{d}$ is computed as average of

$$d_{t+1} = \frac{1}{n_{3,t+1}} \sum_{i=1}^{n_3} \left(\left(\widehat{e}^{(1)}_{i,t+1)} \right)^2 - \left(\widehat{e}^{(2)}_{i,t+1} \right)^2 \right), \tag{12}$$

across all timestamps in our testing sample. In above $\widehat{e}^{(1)}_{i,t+1)}$ and $\widehat{e}^{(1)}_{i,t+1)}$ are the prediction errors from the models under comparison, $n_{3,t+1}$ is the number of observations (options) at timestamp $t+1$ and $\widehat{\sigma}_{\bar{d}}$ is the Newey-West standard error of d_{t+1} across all timestamps in the testing sample. As mentioned in Gu et al. (2020), this adapted version of DM test is more likely to fulfill the regularity conditions required for the DM test statistic to follow $\mathcal{N}(0,1)$ distribution asymptotically and produce reliable p-values and conclusions.

2.9 Variable Importance

The importance of interpretability for any ML model is discussed in length by Israel et al. (2020). Hence, we aim to scratch the surface in the interpretability direction by implementing a concept of "variable importance" to understand which covariates influence the IVS of SSO the most. Similar to Gu et al. (2020) we calculate variable importance of j^{th} variable (VI_j) as relative drop in the R^2_{oos} from setting all values of covariate j to zero, while keeping the remainder of model estimates fixed as it is. The intuition of variable importance is that the more the drop in R^2_{oos}, more important the covariate is.

2.10 Feature Scaling

Generally speaking, all ML models deploy a numerical optimization techniques like SGD to converge towards the solution of objective function. These objective functions are generally complex and have a non-closed form. Hence, to make the convergence towards optimal solution feasible, all ML models in our study (except OLS and RF[27]) need features to have comparable scales[28]. Hence, following Gu et al. (2020) we also apply a "min-max" scaling transformation to all the features in the feature set by scaling them between $[-1, 1]$[29]. This scaling aids in making the numerical optimization process feasible and faster.

[27] Although the output of OLS and RF do not change even if they are trained on scaled features.

[28] Refer to Hastie et al. (2009) for other benefits of feature scaling (for instance, comparability of feature coefficients importance in case of OLS etc.

[29] Scaling applied across time-series.

3 Data

In this section we discuss the dataset and features upon which our SSO IVS modeling research is conducted. For SSO, we include 10 US equities[30] as listed in table 1. To have a good representation of general economy and avoid sampling bias, we select two most liquid stocks from five biggest sectors in the US. Similar to Bernales and Guidolin (2014), we use American style options for single stocks and European style options for S&P500[31]. Our sampling frequency is daily business / trading days. The reason for selecting daily frequency is that options are generally held for "short" duration of time as compared to their underlyings (Bauer et al. (2008)). As per the Options Clearing Corporation (OCC) the average period for which an option was held was 23 days in 2021. Traders and institutional investors prefer to switch in and out of their positions quickly as they react to information in the markets and adjust their hedging and speculative positions which results in relatively short periods of option holding. In our opinion, having an IVS model which works on daily data will provide traders and institutional investors with more real-time updates of IVS dynamics and adjust their position accordingly. Any other downsampled frequency like weekly or monthly may not be of much empirical use as the options holder would not be able to react to multiple information during that duration resulting in sub-optimal trading and investment decisions.

A SSO's price (and IVS) is likely to react to information affecting the underlying, broader markets (index) and macroeconomic indicators (Bakshi et al. (2000)). Hence, we build our feature set to include the elements that absorb the above relevant information spectra described above. Our feature set contains option specific features (greeks, price, etc.), underlying specific features (returns, bid-ask spread, etc.), index specific features (SPX returns, SPX IVS, VIX[32], etc.) and macroeconomic features (US Fed rate, recession indicator, etc.).

Table 1: List of Single Stocks Considered for SSO IVS Analysis

Company Name	**Ticker**	**Sector**	**# Observations**
Apple Inc	AAPL	Technology	144,629
Microsoft Co	MSFT	Technology	119,760
Chevron Corp	CVX	Energy	66,534
Exxon Mobil	XOM	Energy	65,796
Johnson & Johnson	JNJ	Healthcare	63,953
Pfizer	PFE	Healthcare	51,974
JP Morgan	JPM	Financial	73,967
Bank of America	BAC	Financial	60,734
Boeing	BA	Industrial	143,905
General Electric	GE	Industrial	22,983

Note: The table lists the stocks undertaken for our SSO IVS analysis. For each stock we list their name, ticker, sector and number of SSO observations in the sample dataset. The number of observations in the sample are after applying some data filters (to be described shortly). The total number of observations adds up to 814,235.

[30] We focus our research to US equities only as they are generally the most liquid stocks and have extensively maintained database which is essential for reliable research.

[31] We have SPX options in our dataset for two reasons. First, Bernales and Guidolin (2014) indicate SSO IVS dependence on index (SPX) IVS, hence, we want to include them in our feature set. Second, we also model SPX IVS to compare its predictive accuracy with SSO IVS.

[32] We consider VIX as SPX (index) specific feature because VIX is computed using weekly and standard SPX options (Chicago Board Options Exchange (CBOE)).

We refer the reader to Table 7 in Appendix for complete list of 28 features used in SSO IVS analysis. All the option specific data is extracted from the OptionMetrics database under Wharton Research Data Services (WRDS), all stock and SPX specific data is obtained from Center for Research in Security Prices (CRSP (also under WRDS) and macroeconomic data is obtained from Federal Reserve Economic Data (FRED) and National Bureau of Economic Research (NBER).

On a time period basis, our sample dataset ranges from Janurary 1, 2019 to December 31, 2021, hence, 3 years of daily data. We keep the year 2019 as training set, 2020 as validation set and 2021 as our testing set. When the ML model is hypertuned, we retrain it fully on both training set (2019) and validation set (2020) to make predictions in the testing set (2021). The reason for such a split is that 2019 was a year of low volatility (average 2019 VIX: 15.3), 2020 was a year of high volatility (2020 average VIX: 29.2) and 2021 was a year with a mix of both low and high volatility periods (2021 average VIX: 22.6). Hence, during the training our ML models gets to train on both low and high volatility regimes and in the test set the models gets to predict both low and high volatility regimes. This setup allows our ML models to have both kind of volatility regimes in both train and test set, representing a true and unbiased data panel.

Similar to Bernales and Guidolin (2014), Almeida et al. (2022) and other studies on option research, we also apply some filters to our options data. We exclude options for which IV and greeks data is not available to avoid uninformative noise from the sample. To avert biases imputed by microstructure noise, we exclude options with zero volume, bid price \$0, ask $\leq$ bid, mid-price $\leq$ \$0.3 for SSO and mid-price $\leq$ \$0.375 for S&P500 options. Furthermore, we focus on options with time to expiry (τ) between 20 and 240 calendar days[33] and moneyness ($m = S_t/K$) between 0.8 and 1.2 as these options are relatively more liquid and free of noise. As done in Almeida et al. (2022), we restrict our focus to out-of-the-money (OTM) options as they are more liquid and informative than in-the-money (ITM) options.

Finally, we want to show some summary statistics of our sample SSO dataset. To give the reader an idea of distribution of options across the cross-section (i.e. moneyness (m) and time to expiry (τ)) we categorize our options data based on moneyness (m) and time to expiry (τ). Across moneyness (m) we split options into 5 disjoint intervals - Deep OTM call (DOTMC) if $m \in [0.80, 0.90)$, OTM call (OTMC) if $m \in [0.90, 0.97)$, ATM if $m \in [0.97, 1.03)$, OTM put (OTMP) if $m \in [1.03, 1.10)$ and Deep OTM put (DOTMP) if $m \in [1.10, 1.20]$. Across time to expiry (τ) we split options into 2 disjoint intervals. 'Short' term if an option has [20,60] days to expiration else 'Long' term if an option has (60,240] days to expiration.

Table 2 briefly reports the summary statistics of SSO IV data across different moneyness (m) and time to expiry (τ) categories. We can observe that the trading activity distribution (proxied by count of options) in our sample was highest for ATM options and reduced as the options became more OTM. Furthermore, we see that the ATM options had the lowest average IV across the moneyness (m) category. It is also worthy to note that our sample is slightly skewed towards long-term options. Interestingly, there is not much difference between the average IV of short-term options and long-term options within each moneyness (m) category.

[33] While calculating time to expiry (τ) we pay attention to option's settling time. We deduct one day from the τ calculation for options settling during market open (AM) as compared to market close (PM).

Table 2: Summary Statistics of SSO Implied Volatility Data

Time to expiry (τ): Short-term (20-60 days)

Moneyness (m)	**[0.80, 0.90) DOTMC**	**[0.90, 0.97) OTMC**	**[0.97, 1.03) ATM**	**[1.03, 1.10) OTMP**	**[1.10, 1.20] DOTMP**	**Total**
Count	50,817	86,091	88,465	77,805	55,944	359,122
Mean (%)	37.17	30.20	29.17	31.74	37.88	-
Std. dev. (%)	15.16	13.11	12.88	12.57	13.68	-

Time to expiry (τ): Long-term (60-240 days)

Moneyness (m)	**[0.80, 0.90) DOTMC**	**[0.90, 0.97) OTMC**	**[0.97, 1.03) ATM**	**[1.03, 1.10) OTMP**	**[1.10, 1.20] DOTMP**	**Total**
Count	85,871	101,216	102,487	90,539	75,000	455,113
Mean (%)	33.45	30.54	29.23	31.94	36.50	-
Std. dev. (%)	14.76	13.07	12.66	12.43	13.66	-
Total Count	136,688	187,307	190,952	168,344	130,944	**814,235**

Note: The above table shows summary statistics for our SSO IV data sample which ranges from Jan 1, 2019 to Dec 31, 2021. We refine the summary statistics across different moneyness (m) and time to expiry (τ) categories. For each category we show the number of options falling in that category, mean IV in that category and standard deviation of IV in that category. We also show the total count of options.

4 SSO IVS Modeling - Empirical Analysis via Machine Learning

In this section we discuss the key results from our research's investigation into ML models capability in modeling the SSO IVS. We first show the results of (important) variable selection by ENet. Then we discuss the performance of ML models for SSO (and SPX) IVS forecasts where we compare the performance of ML models relative to each other. During this comparison we also briefly talk about OOS performance difference between reduced (parsimonious) and full model. Additionally, for each ML model, we discuss the predictive accuracy difference between SPX (index) IVS and SSO IVS, lending us insights into if former have higher predictability. Furthermore, we talk about performance difference between (best) standalone ML model and Almeida et al. (2022)'s 2-Step approach for both SSO (via AAPL) and SPX IVS modeling. During all the comparisons between different models, we touch upon the statistical significance of OOS performance difference. Lastly, we report and talk about the main covariates driving the IVS dynamics.

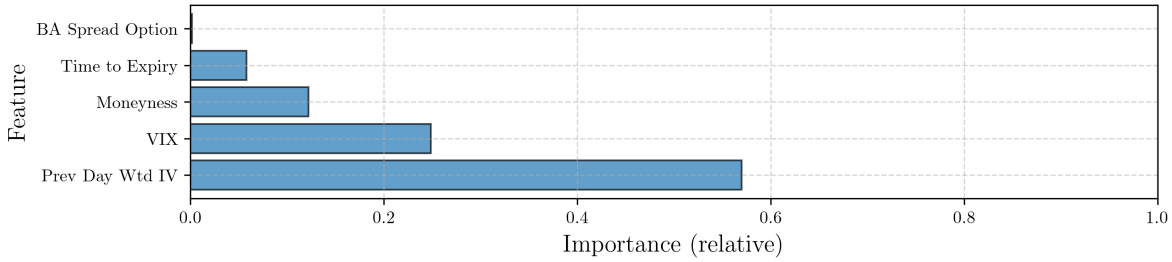
4.1 SSO IVS Modeling - Variable Selection

In Methodology section we mentioned selecting five most important variables[34] (as deemed by ENet) for the IVS modeling purposes. Figure 7 shows us the top five variables selected by ENet for the purposes of SSO IVS modelling[35]. Bigger the bar value of a variable, more importance it was given by ENet during the variable selection process. The variables (features) selected intuitively make sense. An underlying's

[34]It is important to note that the variables selected by ENet (or any other variable selection method) are not always necessarily in reality the most important variables that hold strong predictability for the DGP.

[35]Refer to Figure 16 in Appendix for results of variable selection by ENet for SPX IVS modeling

Figure 7: Variable Selection via ENet for SSO IVS Modeling



Note: The figure shows the five most important variables (from the total of 28 features) ordered by their importance as selected by ENet for SSO IVS Modeling. The length of bars indicate the relative importance of corresponding selected variables.

previous day weighted IV[36] should be a strong predictor of underlying's today's IVS (which is the basic underlying proposition of Bollerslev (1986)). VIX, Moneyness (m), Time to expiry (τ) and Bid-Ask spread of option are remaining features selected, all of which economically make sense and are (potentially) crucial covariates to model the IVS of SSO via ML. The findings of our variable selection also align with the variable importance findings of Almeida et al. (2022) where they also find features like VIX and moneyness (m) are more important covariates dictating the IVS predictability while macroeconomic features playing little-to-no role, which also are not selected by ENet in our setting.

During the prediction exercise for IVS modeling, for each ML model we predicted IVS using two versions - "reduced" model (with ENet selected features only) and "full" model (utilizing all features in the feature set). It is a common belief in ML community that simpler models are able to generalize better, are robust and deliver better OOS predictive performance as compared to complex ML models (Domingos (2012)). We see that this is the case in our study as well because the reduced (parsimonious) ML models outperformed full (complex) ML models in SSO (and SPX) IVS prediction[37]. In the analysis and results to follow, we only discuss and compare the reduced models[38].

4.2 SSO IVS Modeling - ML Models Performance

We now finally turn to answering the main and fundamental question of our research - how well suited are ML models for SSO IVS modeling and forecasting? Figure 8 shows the performance results of ML models[39] by reporting the R^2_{oos} and IVRMSE metrics for each ML model in our study. We use R^2_{oos} to draw a parallel with

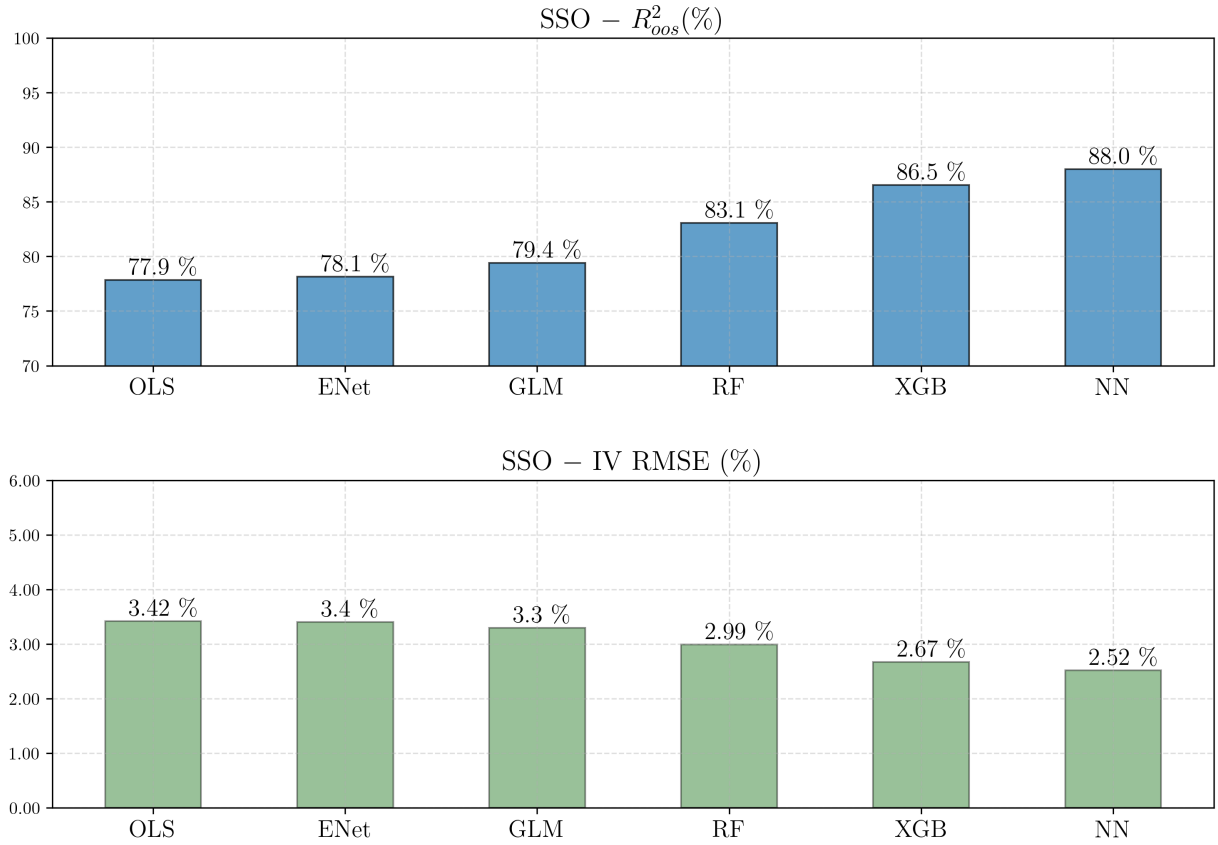
[36]The calculation is custom where we consider the IV of all the options (hence, equivalently considering the IVS) trading on previous day for the underlying and weigh them by the dollar volume amount traded in them. The intuition behind this custom calculation is that more liquid (traded) option contains relatively more information about the IV of underlying and hence should be relied upon more for getting IV(S) estimate of the underlying. This also allows us to summarize whole IVS in one number.

[37]Refer to Table 9 and 10 in Appendix for exact details on how reduced and full (ML) models compared with each other for both SSO and SPX IVS prediction exercise respectively.

[38]We do this for the sake of brevity and to maintain focus on main research topic (while not distracting ourselves too much with other finer aspects of an end-to-end ML analysis). Also, intuitively it makes sense to only compare the better versions of each ML model. Furthermore, we do not want to flood the reader with irrelevant information that may bore, astray and potentially confuse them, especially in Tables and Figures.

[39]We only show the performance of (best) hypertuned versions of each ML model. For NN this also encompasses the number of hidden layers. Refer to Table 8 in Appendix for information on the hyperparameter grid used.

Figure 8: ML Models Performance in SSO IVS Modeling



Note: The figure shows R^2_{oos} (top) and IVRMSE (bottom) metrics of ML models for SSO IVS prediction exercise. As the model's ability to capture complex relationships increases, so does their predictive performance.

Gu et al. (2020) and IVRMSE to draw a parallel with Almeida et al. (2022). There are few key observations that can be drawn from the results observed. First, compared to Gu et al. (2020) results for return prediction using ML, we see "very high" values of R^2_{oos} for our SSO IVS prediction exercise (via ML). This empirical observation reiterates the common belief in financial world that predicting volatility (with high accuracy) is more feasible than predicting returns. This is due to more independent and identical distribution (i.i.d.) nature of returns as compared to volatility, which is known to exhibit strong auto-correlation and clustering. Additionally, returns can take both positive and negative values whereas volatility values are positive and this unrestricted range of returns also make them harder to predict.

For the performance amongst ML models for SSO IVS forecasting we see that linear OLS model performs the least with highest IVRMSE of 3.42%. This least performance is expected given the fact that non-linear dynamics are inherent in IVS (Andersen et al. (2015)) which the OLS model fails to capture. The second least performing model is ENet with IVRMSE of 3.40%. The marginal performance gain of 2bps in IVRMSE from regularization seems not that "high"[40] which is possibly due to the fact that we are comparing reduced

[40]Interestingly, later in Table 3 where we perform DM test, we will see that this meagre 2bps improvement in IVRMSE is statistically significant even at 5% significance level.

models and the reduced models are in a sense already regularized. For the GLM we observe an IVRMSE of 3.30%, showing that its ability to capture non-linear effects of IVS dynamics yields a better prediction performance. For RF, which is also able to model the interaction effects, we see a further improved IVRMSE of 2.99%. For the XGBoost we obtain an IVRMSE of 2.67%, indicating that the ability to improve from weak learners helps in IVS forecasting. Finally, the best predictor is NN[41] with the lowest IVRMSE of 2.52%. In our study (and as also found by Gu et al. (2020), Almeida et al. (2022), etc.) NN successfully demonstrates its capability to model complex DGP dynamics characterizing the IVS and justifies its status as "universal approximator" (Hornik et al. (1989)).

While Figure 8 compares the ML models performance based only on metric values of IVRMSE and R^2_{oos}, we now show the results of DM test in Table 3 and see if the SSO IVS forecasts from our ML models are (statistically) significantly different. The (t)-test statistic of DM test follows $\mathcal{N}(0,1)$ distribution under the null hypothesis $(\mathcal{H}_0)$ which says that the difference in forecasts are not statistically significant. We can draw on p-value of the DM test statistic to check if the forecasts indeed are significantly different. In our DM test table a positive value implies that the model listed in the column had better forecasts as compared to the model listed in the row. For a model's forecast to be significantly better (or worse) than another model's forecast at 5% significance level, the absolute value of DM test statistic should be greater than or equal to 1.96 i.e. $|t| \geq 1.96$.

Table 3: Diebold Mariano Test - SSO IVS Forecasting

	EN	**GLM**	**RF**	**XGB**	**NN**
OLS	**3.02**	**5.70**	**13.88**	**26.93**	**30.52**
EN		**5.09**	**12.43**	**24.71**	**27.61**
GLM			**8.19**	**24.68**	**25.03**
RF				**10.63**	**13.12**
XGB					**10.81**

Note: The table lists the results of pairwise DM test between each pair of ML model comparing their performance in out-of-sample SSO IVS forecasting. We only show the upper triangular values to avoid repetition and aid in readability. The positive value of DM test statistic indicates that the model in the corresponding column outperformed the model in the corresponding row. The DM test statistic in **bold** indicate that statistic was significant at 5% significance level i.e. p-value < 0.05.

Table 3 lists down pairwise values of DM test statistic. We see that all the test statistics are significant at 5% significance level implying that each model has under/outperformed significantly with respect to every other model in our study. Another observation is that our DM test statistics have "unusually high" numbers as compared to what was observed in Gu et al. (2020) despite the modification we made to the DM test to reasonably satisfy regularity conditions[42]. Irrespective, for our study, we see that ENet's outperformance over OLS, GLM's outperformance over ENet, RF's outperformance over GLM, XGB's outperformance over RF and NN's outperformance over XGB - all of them are statistically significant. This analysis implies that Neural Network possibly is the best model to go about for IVS modelling (possibly) due its ability to catch complex dynamics of IVS DGP.

[41] With a single hidden layer only.

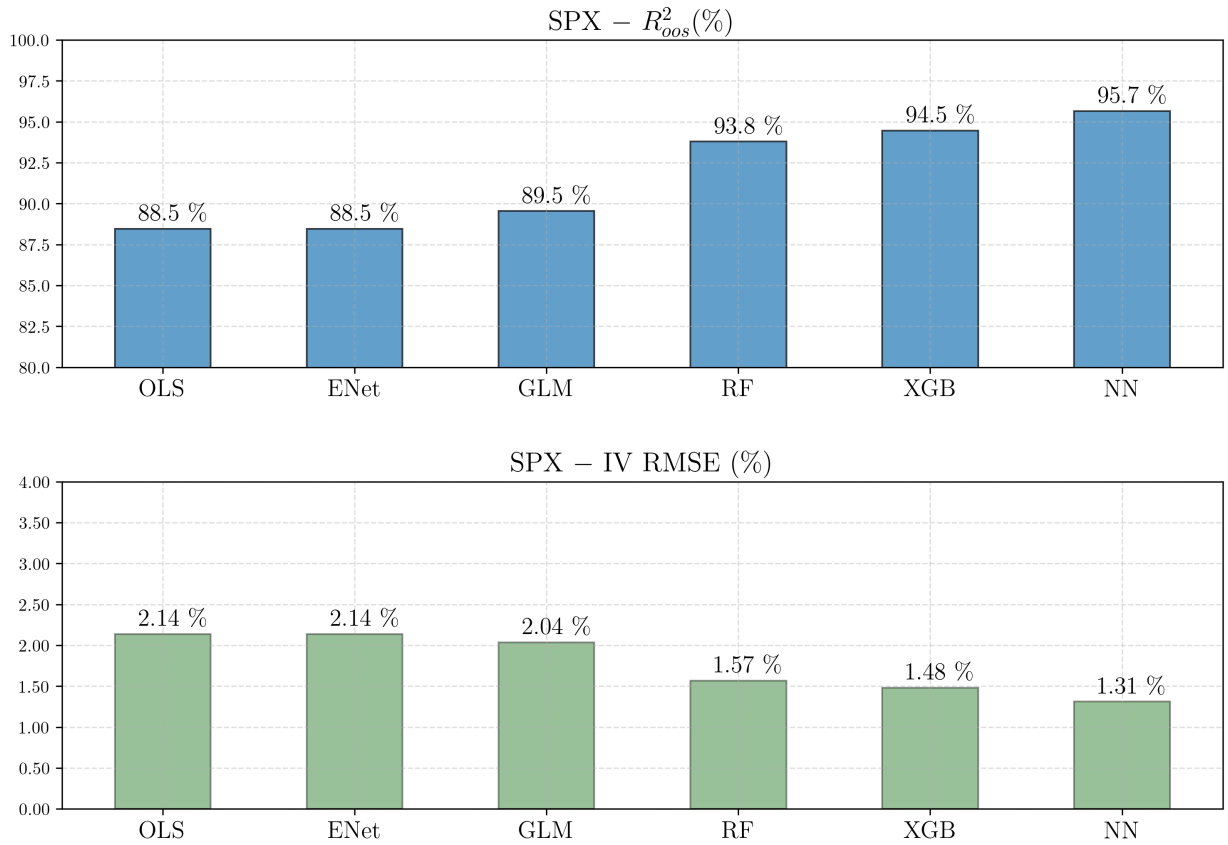
[42] Possibly this is why generally literatures related to options research omit reporting the DM test analysis.

4.2.1 Extension: SPX IVS Modeling - ML Models Performance

We now extend the discussion of ML models performance in SPX IVS modeling. By doing so we gain two mutually exclusive insights. First, how do ML models perform in SPX IVS modeling and second, if SPX (index) IVS has a higher predictability compared to SSO (single stock) IVS. Figure 9 shows us the performance of ML models for SPX options IVS forecasting in a fashion exactly similar to what we did for SSO IVS in Figure 8. The performance trend observed in SSO IVS modeling reflects as it is in SPX IVS modeling. We see that for SPX IVS forecasting as well OLS is the least performing model with 2.14% IVRMSE followed by ENet (2.14%), GLM (2.04%), RF (1.57%), XGB (1.48%) and once again NN[43] delivering the best performance at 1.31% IVRMSE.

Another important observation is that all the ML models in our study were able to produce more precise IVS forecasts for SPX options as compared to SSO with at least 1% difference in the corresponding IVRMSEs. A plausible reason for this can be that S&P500 options are one of the most liquid options traded, and more liquidity signifies more information (signal) priced in that is available for ML models to extract. This richness

Figure 9: ML Models Performance in SPX IVS Modeling



Note: The figure shows R^2_{oos} (top) and IVRMSE (bottom) metrics of ML models for SPX IVS prediction exercise. As the model's ability to capture complex relationships increases, so does their predictive performance.

[43] Again, with single hidden layer only.

of SPX options can be seen from the fact that in our sample of 3 years, we have ∼1,340,000[44] rows of SPX options data compared to ∼814,000 rows of SSO data (for 10 stocks combined). Another intuitive reason for relatively lower accuracy for SSO IVS forecasts is that stocks are also riddled with stock specific noise (in the form of company specific news) where as S&P500 is a proxy for market aggregation where the company specific noise (errors) are likely to cancel out each other on average.

Table 4: Diebold Mariano Test - SPX IVS Forecasting

	EN	**GLM**	**RF**	**XGB**	**NN**
OLS	0.08	**13.12**	**18.50**	**24.10**	**32.36**
EN		**13.23**	**18.6**	**24.23**	**32.34**
GLM			**16.35**	**24.29**	**33.34**
RF				**4.39**	**7.09**
XGB					**7.28**

Note: The table lists the results of pairwise DM test between each pair of ML model comparing their performance in out-of-sample SPX IVS forecasting. We only show the upper triangular values to avoid repetition and aid in readability. The positive value of DM test statistic indicates that the model in the corresponding column outperformed the model in the corresponding row. The DM test statistic in **bold** indicate that statistic was significant at 5% significance level i.e. p-value < 0.05.

As done for SSO, we also conduct DM test analysis for SPX IVS forecasting via our ML models. Table 4 lists down the pairwise values of DM test statistics for each pair of ML model used. The findings of DM test analysis for SSO reflects itself in SPX DM test analysis as it is. For SPX IVS forecasting too, we see that except one DM test statistic all other test statistics are significant at 5% significance level implying that each model has under/outperformed significantly with respect to every other model in our study. Again, the DM test statistics have "unusually high" numbers. Here too, we see that ENet's outperformance over OLS, GLM's outperformance over ENet, RF's outperformance over GLM, XGB's outperformance over RF and NN's outperformance over XGB - all of them are statistically significant. Combined with SSO DM test analysis, it seems that NN is the best ML model for IVS modelling, both index and single stocks.

4.3 SSO IVS Modeling - Comparison With 2-Step Approach

So far through our research it is quite evident that ML models (especially NNs) can be used to model IVS of SSO (and SPX options) with "decent" accuracy. Almeida et al. (2022) find BSM to deliver an IVRMSE of roughly 8% for SPX options IVS while we observe that by using ML methods the IVRMSE drops to 2.5% for SSO IVS and 1.3%[45] for SPX options IVS. This is a very significant improvement, especially given the significant values of DM test statistics in preceding analysis even for minor improvements in IVRMSE. Hence, it can be established that ML models are strong candidates (especially NNs) and show great promise in IVS modeling as they outperform traditional parametric models. But recently a new strand of research has emerged that combines economic rationale and ML with the hopes of generating superior modeling performance stemming from synergies formed by using economic logic to guide the ML models and somewhat

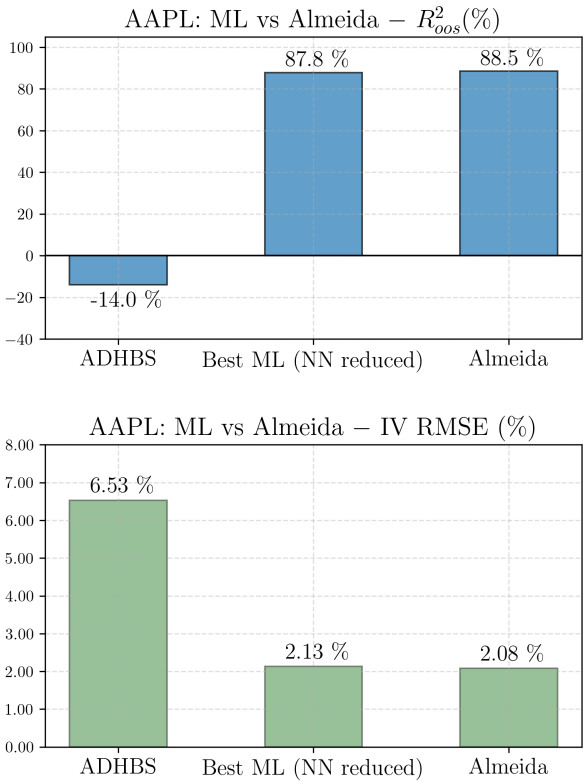
[44] Refer to Table 11 in Appendix for more details.

[45] Albeit the datasets are different. The key conveying point here is the significant performance boost observed by deploying ML for IVS modeling.

"tame" their open-ended non-parametric nature which is prone to overfitting. Almeida et al. (2022) is one such study in the field of options IVS modeling using the 2-Step approach to model the IVS of SPX options. They first use traditional IV models like BSM (Black and Scholes (1973)), ADHBS (Dumas et al. (1998)), Carr and Wu (Carr and Wu (2016)) and Heston (Heston (1993)) to model the IVS and then fitting a NN on the pricing errors of former parametric models to improve the IVS forecasts. This way the NNs are "guided" by the economic reasoning of the parametric models. Hence, it makes for an interesting case to see how the performance of standalone ML models compare to guided ML models of 2-Step approach.

On that front, we pick the best ML model we found for IVS modeling (i.e. reduced NN) and compare it with Almeida et al. (2022)'s 2-Step approach. For the sake of brevity and maintaining overarching focus of research, we carry out a simple version of Almeida et al. (2022)'s study by only using ADHBS and dropping other three parametric models. We expect this simplicity to bring no compromise to our analysis for a couple of reasons. First, as argued in Almeida et al. (2022) that applying NN to errors of BSM is equivalent to directly fitting a NN as the constant doesn't impact the optimization problem[46]. This allows us to remove

Figure 10: (AAPL) SSO IVS Forecasting Comparison - ADHBS vs. Best (Standalone) ML vs. (2-Step) Almeida



Note: The figure shows R^2_{oos} (top) and IVRMSE (bottom) metrics for Ad-Hoc Black Scholes Model (ADHBS), the best standalone ML model (reduced NN with single hidden layer) and Almeida et al. (2022)'s 2-Step approach (ADHBS + NN) for AAPL IVS forecasting (as a proxy for general SSO IVS forecasting). The figure allows for comparison among all three different styles of SSO IVS models - conventional parametric models (ADHBS), ML models (standalone) and 2-Step (ML and parametric combined).

[46]Check Equation 10 of Almeida et al. (2022) for more details.

BSM from our analysis as we already fit a NN directly. Second, ADHBS is a simple model to implement (as compared to Carr and Wu, Heston) wherein this simplicity does not come at the cost of reduced performance as Almeida et al. (2022) observe the same lowest IVRMSE post best NN fit for ADHBS, Carr and Wu, and Heston model for 1-day ahead forecasts[47]. We compare the forecasts of our best ML model with ADHBS model and 2-Step ADHBS + NN method. Finally, we want to mention that we conduct this comparison only for Apple Inc. (Ticker: AAPL) which will act as a general proxy for SSO IVS forecasting comparison[48].

Figure 10 shows us the results of IVS forecasting performance for ADHBS, best standalone ML model found during SSO IVS forecasting (i.e. reduced NN with single hidden layer) and Almeida et al. (2022)'s 2-Step (ADHBS + NN) approach. We see a negative R^2_{oos} for ADHBS which is significantly improved upon by standalone NN model which takes the R^2_{oos} to 87.8%, which further is improved by 2-Step approach to 88.5%. Correspondingly, the IVRMSE for ADHBS is 6.53%, reduced to 2.13% by standalone NN and further reduced to 2.08% by 2-Step approach. From the evaluation metrics it seems that Almeida et al. (2022)'s 2-Step approach is the best model for SSO IVS forecasting showcasing the benefits of using economic rationale to guide ML models.

We again check the significance of performance difference using DM test. For the sake of completeness and bigger picture we show the DM test analysis with all ML models, ADHBS and 2-Step approach. Table 5 shows the pairwise DM test statistics. We see that ADHBS was the least performing model for SSO IVS modeling. This performance was further improved significantly by models as specified in the order - OLS, ENet, GLM, RF, XGB, NN and Almeida et al. (2022)'s 2-Step method. The conclusion from this DM test analysis is that for SSO IVS modeling[49] - ML models are better than traditional parametric methods like ADHBS and the 2-Step approach is the best model by far, even outperforming standalone NN significantly.

Table 5: Diebold Mariano Test Incl. ADHBS and Almeida (2-Step) - AAPL (SSO) IVS Forecasting

	GLM	**EN**	**RF**	**XGB**	**NN**	**ADHBS**	**Almeida**
OLS	**5.96**	**6.53**	**7.35**	**11.95**	**12.21**	**-7.22**	**13.47**
EN		**6.47**	**7.33**	**11.92**	**12.18**	**-7.25**	**13.44**
GLM			**7.0**	**12.69**	**11.39**	**-7.34**	**12.71**
RF				**2.84**	**2.88**	**-7.65**	**3.44**
XGB					1.46	**-7.9**	**2.47**
NN						**-7.93**	**2.45**
ADHBS							**7.96**

Note: The table lists the results of pairwise DM test between each pair of models covered so far (i.e. ADHBS, all ML models and Almeida et al. (2022)'s 2-Step approach) to compare their performance in out-of-sample AAPL (SSO) IVS forecasting. We only show the upper triangular values to avoid repetition and aid in readability. The positive value of DM test statistic indicates that the model in the corresponding column outperformed the model in the corresponding row. The DM test statistic in **bold** indicate that statistic was significant at 5% significance level i.e. p-value < 0.05.

[47] Check Table 2 of Almeida et al. (2022) for exact values.

[48] Since each underlying requires a separate ADHBS model to be fit to it, we only use Apple Inc. out of the 10 single stocks to respect the thesis timeline and bounds of computational constraints. Furthermore, in our study AAPL is the most liquid stock and is most actively traded SSO by notional value and open interest. This allows us to draw most accurate conclusions about our comparison task of 2-Step vs. standalone ML as more data implies lower standard errors and more stable and reliable results.

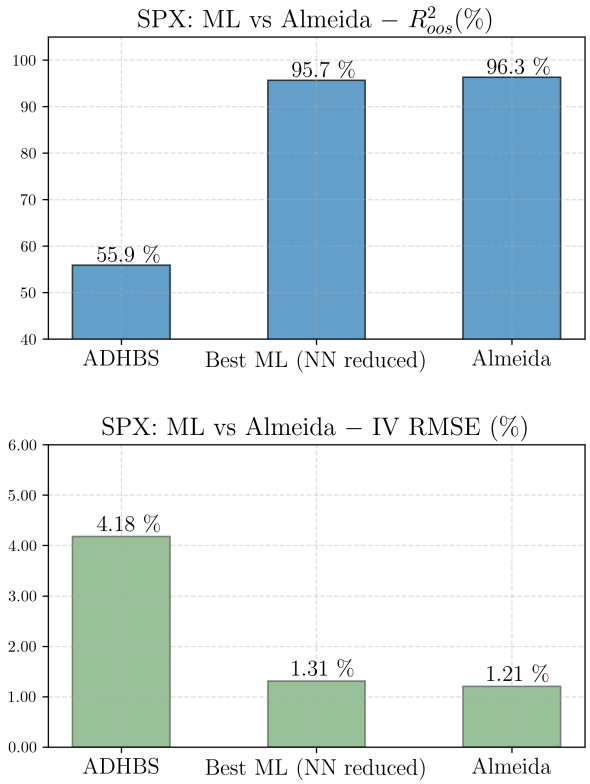
[49] Proxied by comparative analysis on Apple Inc. (AAPL)'s single stock options.

4.3.1 Extension: SPX IVS Modeling - Comparison With 2-Step Approach

We extend the comparison with Almeida et al. (2022)'s 2-Step approach for SPX options IVS forecasting to get an insight if 2-Step approach ubiquitously outperforms parametric and standalone ML models for the task of IVS forecasting for all kind of options.

As done for comparison of performance of ML models with ADHBS and 2-Step approach for SSO IVS modeling, in a similar light Figure 11 shows us the results of SPX IVS forecasting performance for ADHBS, best ML model found during our SPX analysis (i.e. reduced NN) and Almeida et al. (2022)'s 2-Step approach using ADHBS + NN. This time we see a positive R^2_{oos} for ADHBS albeit lower than the other 2 methods. This is again significantly improved upon by standalone NN which takes the R^2_{oos} to 95.7%, which further is improved by 2-Step approach to 96.3%. Correspondingly, the IVRMSE for ADHBS is 4.18%, reduced to 1.31% by standalone NN and further reduced to 1.21% by 2-Step approach. The IVRMSE of 1.21% achieved via 2-Step approach is very close to the lowest IVRMSE of 1.17% obtained by Almeida et al. (2022) in their 1-day ahead prediction[50]. Again, the evaluation metrics implies that Almeida et al. (2022)'s 2-Step approach

Figure 11: SPX IVS Forecasting Comparison - ADHBS vs. Best (Standalone) ML vs. (2-Step) Almeida



Note: The figure shows R^2_{oos} (top) and IVRMSE (bottom) metrics for Ad-Hoc Black Scholes Model (ADHBS), the best standalone ML model (reduced NN with single hidden layer) and Almeida et al. (2022)'s 2-Step approach (ADHBS + NN) for SPX IVS forecasting. The figure allows for comparison among all three different styles of SPX IVS models - conventional parametric models (ADHBS), ML models (standalone) and 2-Step (ML and parametric combined).

[50]Slight difference is expected due to the different samples covered.

is the best model for IVS modeling of index options like S&P500.

For SPX IVS modeling too we test out the significance of performance difference using DM test. Again, for the sake of completeness and bigger picture, we show a DM test with all the ML models, ADHBS and Almeida et al. (2022)'s 2-Step approach. Table 6 shows the pairwise DM test statistics. Similar to DM test analysis findings for SSO IVS modeling, for (index) SPX IVS modeling too we see that ADHBS is the least performing model. The following models saw significantly improved OOS prediction results (as specified in the order) - OLS, ENet, GLM, RF, XGB, NN and 2-Step approach. This DM test analysis shows that in addition to IVS modeling for single stocks, even for index (SPX) options IVS modeling - ML models outperform traditional parametric methods like ADHBS and the 2-Step approach (again) outperformed standalone NN significantly to be the best model for index options IVS modeling.

Table 6: Diebold Mariano Test Incl. ADHBS and Almeida (2-Step) - SPX IVS Forecasting

	GLM	**EN**	**RF**	**XGB**	**NN**	**ADHBS**	**Almeida**
OLS	0.08	**13.12**	**18.50**	**24.10**	**32.36**	**-11.88**	**37.61**
EN		**13.23**	**18.6**	**24.23**	**32.34**	**-11.89**	**37.55**
GLM			**16.35**	**24.29**	**33.34**	**-12.06**	**36.79**
RF				**4.39**	**7.09**	**-14.11**	**7.24**
XGB					**7.28**	**-7.9**	**6.61**
NN						**-13.72**	**3.79**
ADHBS							**14.23**

Note: The table lists the results of pairwise DM test between each pair of models covered so far (i.e. ADHBS, all ML models and Almeida et al. (2022)'s 2-Step approach) to compare their performance in out-of-sample SPX IVS forecasting. We only show the upper triangular values to avoid repetition and aid in readability. The positive value of DM test statistic indicates that the model in the corresponding column outperformed the model in the corresponding row. The DM test statistic in **bold** indicate that statistic was significant at 5% significance level i.e. p-value < 0.05.

The cumulative results of this research till this point indicate that ML holds strong potential and great promise in the field of IVS modeling (both single stocks and index options) where they significantly outperform traditional and parametric models like BSM and ADHBS. But we see a further boost to the performance when we combine both traditional parametric models with more recent ML models like NNs. The parametric model guided 2-Step approach is the best performing method for IVS forecasting for both SPX and SSO IVS.

4.4 SSO IVS Modeling - Covariates That Matter

So far we have seen that ML models bring significant predictive power to the table in the context of IVS modelling for SSO (and index options too). Furthermore, their predictive power gets a boost stemming from synergies when combined with a parametric model's economic rationale[51]. This sets a premise and strong case for ML to be included by financial institutions, market participants, central banks and other key players in economic system worldwide for IVS modelling and drawing valuable insights for their tasks like hedging, risk management, bringing in efficiencies to the market, economic policy decision making and other crucial

[51]Except for BSM since it predicts a constant for a whole IVS with no dynamics embodied in the model and a constant shift in the DGP has no bearing on its ability to be modeled.

tasks. Though what often times stops above concerned stakeholders to include ML methods in their financial decision making pipeline is their "black box" nature and complexity in unwinding the model's "behind the scenes" functioning. Israel et al. (2020) also discuss in detail in their paper "Can Machines Learn Finance?" that a major hold back factor for asset managers and financial institutions for staying put of complex ML models is the lack of their interpretability, which is a key criterion while explaining their products, services, strategies etc. to clients and other concerned authorities. Given this premise of importance of interpretability of a ML model, in this part of paper we aim to interpret the ML models used for IVS modeling for SSO IVS and SPX IVS.

We undertake a modest approach to uncover the interpretability of ML models in our study. Since we have used Gu et al. (2020) as baseline reference for this research, for interpretability too we follow their style by analyzing which covariates influence the predictive ability of our ML models the most. This gives us valuable insight into the set of features that are crucial for IVS modeling and their importance in dictating the IVS dynamics. It can help to act as a set of useful variables for market participants to watch out for while making decisions pertaining to IVS. For this we calculate Variable Importance (henceforth, VI) for each covariate in our model. The drop in out-of-sample predictive performance (measured by R^2_{oos}) acts as a proxy for a covariate's importance. More the drop in R^2_{oos} more is the importance of a covariate in predicting IVS dynamics. We call this method as Variable Importance Measure (henceforth, VIM)[52].

Figure 12 visually shows us the VIM for the covariates of ML models used in our study for SSO IVS modelling. Bigger the bar (hence numeric value) of a covariate, higher its importance. We measure relative importance here on a scale of 0 to 1, where 0 implies that the covariate has no role whatsoever in predicting the IVS dynamics and 1 implies that the covariate holds all the explanatory power in predicting IVS dynamics. All the bar (VI) values for a model add up to 1. One clear observation from Figure 12 is that the stock's (i.e. underlying's) previous day weighted average IV is by far the strongest predictor of IVS dynamics. Intuitively this makes sense as it is well known in financial community that volatility tends to cluster and has strong autocorrelation (especially at short lags). Hence, the last day IVS[53] acts as a strong signal to predict the underlying's today's IVS. This empirical observation is what the famous GARCH model by Bollerslev (1986) is based upon and we see it manifesting in our study too. Next up, VIX is generally the second most important covariate in predicting the IVS dynamics of single stocks. This shows that SSO IVS dynamics are partly linked and driven by SPX options IVS dynamics (which is summarized in a single number by VIX[54]). The dependence on index (SPX) IVS dynamics was also observed by Andersen et al. (2015) in their study.

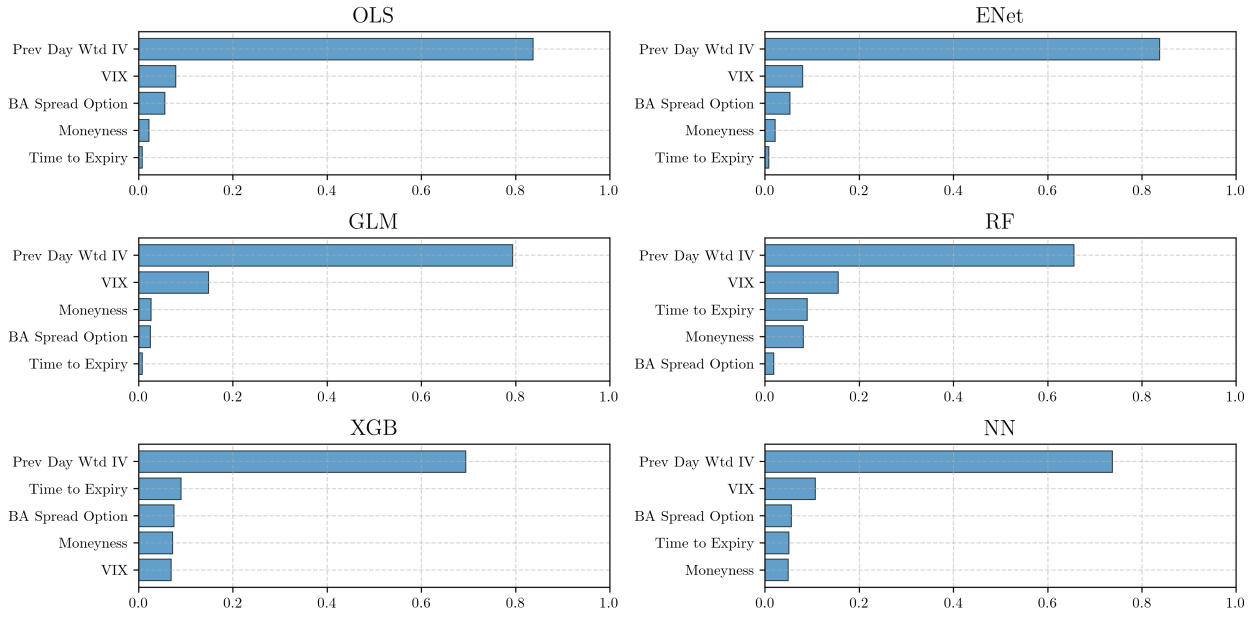
Interestingly, the previous day weighted IV plays a more important role than VIX in modelling the SSO IVS dynamics which can be explained by the fact that stocks are also driven a lot by stock specific noise

[52]For the sake of brevity we conduct the VIM analysis for reduced ML models only as they consistently outperformed full models in IVS modeling as shown in Table 9 and 10 in Appendix.

[53]Since the calculation of an underlying's previous day weighted average IV draws information from all the IV in IVS, the metric can be seen as a summarized information of IVS and hence acting as a proxy for actual IVS.

[54]This is because VIX is computed using weekly and standard SPX options Chicago Board Options Exchange (CBOE). VIX (Volatility Index) - is the measure of expected (implied) volatility of SPX index over the next 30 days. Hence, it is a "forward" looking measure of volatility giving a guesstimate of expected deviation in SPX index (or equivalently in borader markets).

Figure 12: Variable Importance Measure (VIM) - SSO IVS Modeling



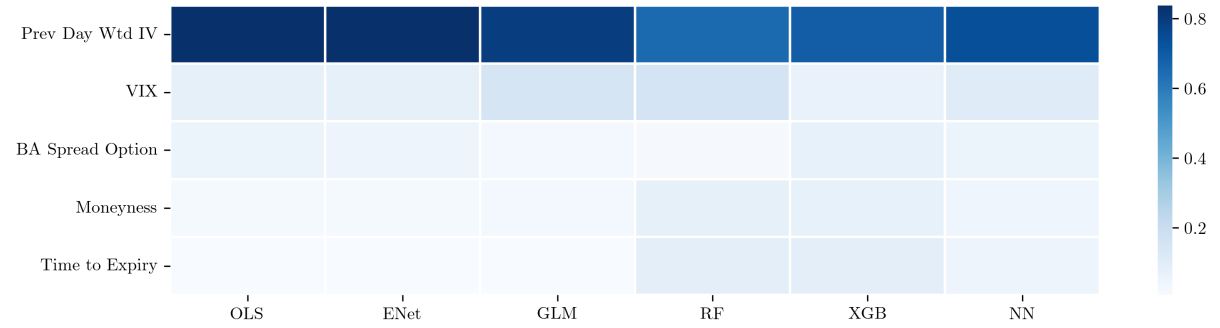
Note: The figure shows the results of Variable Importance Measure (VIM) for all the 6 models in our study used for SSO IVS modeling. The bar represents the relative importance of the corresponding covariate in predicting the SSO IVS. As a bar's value tends to 1, more the corresponding covarite's role in predicting the SSO IVS and as a bar's value tends to 0, it implies that the corresponding covariate has little-to-no role in predicting the SSO IVS. The bars for each model are ordered in their order of importance, from top bar being the most important and the bottom bar being the least important for SSO IVS modeling.

(news) in addition to aggregated macro factors reflected in SPX (and hence in VIX). After this we see that an option's bid-ask spread, moneyness (m) and time to expiry (τ) have a relatively mild influence on IVS dynamics. Generally speaking, wider bid-ask spread indicates higher volatility. For instance, all the stocks in our study had wider bid-ask spread at the height of COVID-19 crash in markets in March 2020 (characterizing highly volatile markets) compared to bid-ask spread they witness at "normal" market situations.

Finally, moneyness (m) and time to expiry's (τ) role in predicting IVS (albeit milder influence) is expected as they are the location paramters of an option in the IVS (i.e. $\sigma(m, \tau)$). Generally speaking, ATM options have higher IV as compared to OTM and ITM counterparts because ATM options have more "uncertainty" regarding an option ending in moneyness or not. With respect to time to expiry, options with more time to expiry (τ) have relatively higher IV as the underlying has more "scope" to move around and hence higher "uncertainty" about the option ending in moneyness or not.

Figure 13 shows a heatmap of VIM of ML models used for SSO IVS modelling. This heatmap is a visual representation of variable importance ranking aggregated across all models used in our study. From the figure it can be seen that generally for any ML model undertaken for the task of IVS modelling for SSO, following are the set of most important covariates to be considered in the order specified - Previous Day Weighted IV (of the underlying), VIX, Bid-Ask spread of the option, moneyness (m) and time to expiry (τ). Based on explanations of each covariate's influence in preceding paragraphs, this ranking of importance is expected.

Figure 13: Variable Importance Heatmap - SSO IVS Modeling

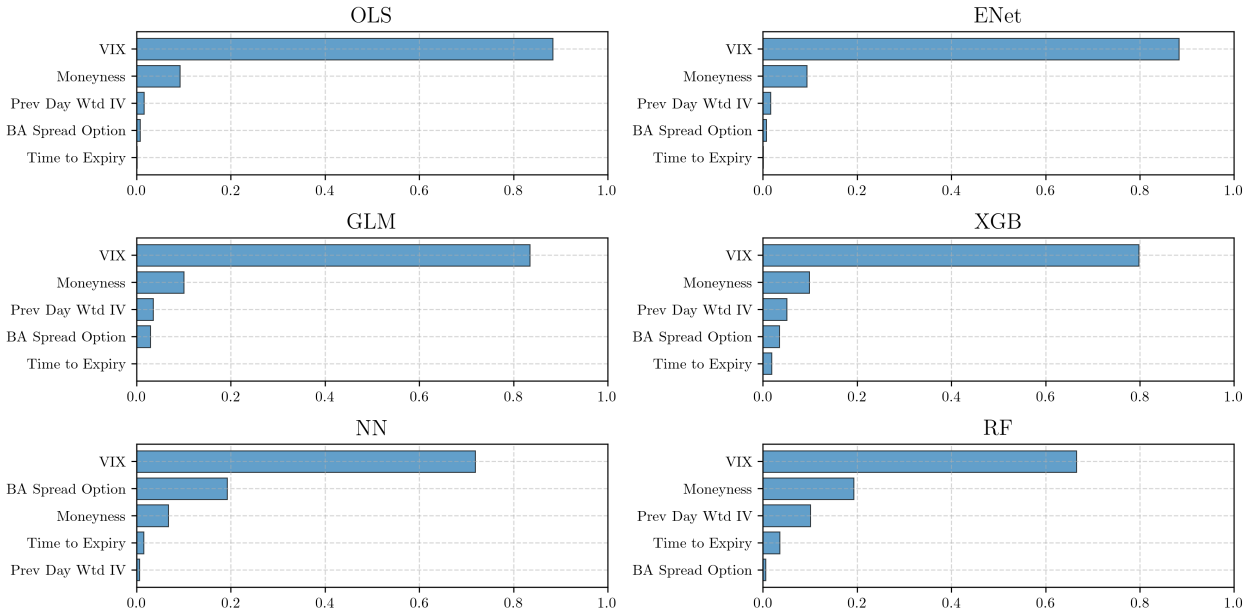


Note: The figure shows the variable importance heatmap for the covariates used in our SSO IVS modeling. The heatmap is an aggregated ranking of a covariate's importance deemed by each ML model in our study. Hence, the top most covariate in heatmap on an overall basis (across all models) was the most important covariate and the bottom most covariate in heatmap on overall basis was the least important covariate for SSO IVS modeling.

4.4.1 Extension: SPX IVS Modeling - Covariates That Matter

After finding out the relative importance of covariates for SSO IVS modeling via ML, the final (sub)task in our research is to investigate the covariates relative importance used in SPX IVS modeling. This also helps to draw one-to-one comparison and see how the importance of relevant variables differ while modeling

Figure 14: Variable Importance Measure (VIM) - SPX IVS Modeling

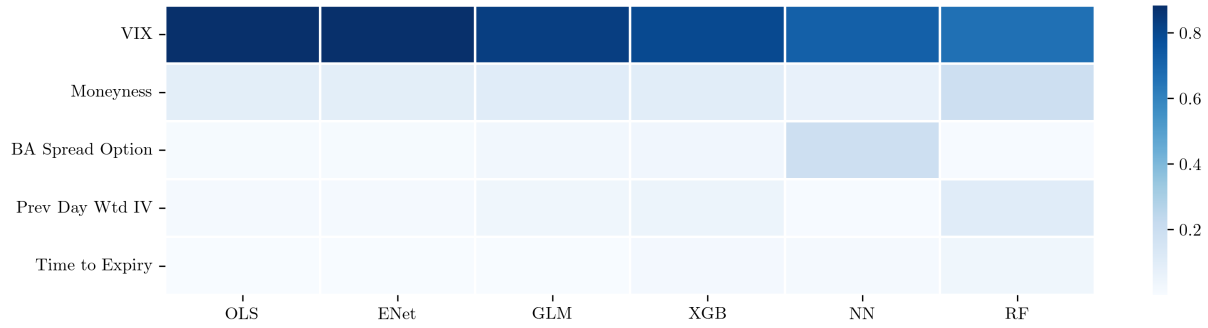


Note: The figure shows the results of Variable Importance Measure (VIM) for all the 6 models in our study used for SPX IVS modeling. The bar represents the relative importance of the corresponding covariate in predicting the SPX IVS. As a bar's value tends to 1, more the corresponding covarite's role in predicting the SPX IVS and as a bar's value tends to 0, it implies that the corresponding covariate has little-to-no role in predicting the SPX IVS. The bars for each model are ordered in their order of importance, from top bar being the most important and the bottom bar being the least important for SPX IVS modeling.

IVS of SSO as compared to modeling IVS of index options like SPX using ML. Figure 14 shows the VIM for covariates used for SPX IVS modeling. We see that VIX is the most important covariate by a big margin for this prediction task. This observation aligns with the finding of Almeida et al. (2022) as they also see VIX as the most important feature in their prediction exercise of SPX IVS. VIX as most important predictor is also intuitively expected as VIX by its calculation is designed to predict the expected (implied) volatility of S&P500 index over the next 30 days based of bid and ask quotes of SPX options (Chicago Board Options Exchange (CBOE)). Hence, the IVS of SPX options is expected to be heavily influenced by VIX. The relative importance of other relevant set of covariates is similar to what was found for relative importance analysis for SSO IVS modeling. For instance, moneyness (m) and time to expiry (τ), the location inputs for an option IV on the IVS i.e. $\sigma(m, \tau)$ here too have relatively mild influence on SPX IVS.

Finally, we also show a heatmap of VIM of covariates for SPX IVS modeling in Figure 15, as was also done during VIM heatmap analysis for SSO IVS modeling. For someone concerned with SPX IVS modeling, our analysis outlines the aggregated variable importance rankings for the covariates to be considered (in importance of given order) - VIX, moneyness (m), Bid-Ask spread of the option, Previous Day Weighted IV (of SPX options) and time to expiry (τ). Based on explanations of each individual covariate's influence in preceding paragraph for SPX IVS modeling, this ranking of relative importance is expected.

Figure 15: Variable Importance Heatmap - SPX IVS Modeling



Note: The figure shows the variable importance heatmap for the covariates used in our SPX IVS modeling. The heatmap is an aggregated ranking of a covariate's importance deemed by each ML model in our study. Hence, the top most covariate in heatmap on an overall basis (across all models) was the most important covariate and the bottom most covariate in heatmap on overall basis was the least important covariate for SPX IVS modeling.

5 Conclusion

In this study, after motivating the need to model the Implied Volatility Surface for Single Stock Options using Machine Learning methods we attempted to do the same and found that they bring significant forecasting gains to the table. All the Machine Learning models in our study significantly outperformed traditional parametric models like (Ad-Hoc) Black Scholes Model for *1*-day ahead forecasting of Implied Volatility Surface. Within the Machine Learning set of models non-linear models outperformed linear models due to former's ability to capture non-linear and interaction effects inherent in Implied Volatility Surface dynamics. Neural Networks was the best performing non-linear model for the task of Single Stock Option (and Stock Index Option) Implied Volatility Surface modeling due to their ability to model complex relationships that characterize the implied volatility dynamics. But at the same time when complexity is overdone using deep learning or more set of features, the predictive performance starts to take a hit due to potential overfitting which the complex Machine Learning models are susceptible to in low signal-to-noise ratio environments like options and financial markets.

Furthermore, for Implied Volatility Surface modeling for both Single Stock Options and Stock Index Options we find that when the predictive abilities of Machine Learning models like Neural Networks are combined with the guidance of parametric model's economic rationale, the predictive power gets a further boost. This 2-Step approach where the Neural Networks improve upon the pricing errors of parametric model was the best method to model the Implied Volatility Surface as they significantly outperformed the traditional parametric models and standalone Machine Learning models.

We also find that the index options Implied Volatility Surface are modeled with relatively higher accuracy as compared to Single Stock Options Implied Volatility Surface. This is because Machine Learning models thrive and perform well in big data settings and index options are more blessed than single stock options as the former have richer information datasets stemming from their higher trading activity. Finally, we find that a few set of covariates heavily influence the dynamics of Implied Volatility Surface for both Single Stock and Stock Index options. These covariates are underlying's previous day Implied Volatility Surface and Volatility Index (VIX).

The takeaway from our study is that Machine Learning is capable of filling the gaps in currently used parametric models for Single Stock Options Implied Volatility Surface modeling and hence should be under the radar of market participants like financial institutions and traders concerned with Implied Volatility Surface modeling. Our study is another drop in the bucket of ever expanding literature making the case for use of Machine Learning in more and more varied applications and financial problems within the financial industry.

Acknowledgement

First and foremost I would like to thank Prof. Gustavo Freire for his supervision and constant guidance during the whole thesis. Without his invaluable insights and support this research would have not reached its current level.

I would also like to thank Prof. Maria Grith for their valuable feedback as second assessor. I also extend my gratitude to Erasmus School of Economics for giving me the opportunity to work on this challenging and interesting research topic which has honed my capability to carry out research and taught me so many aspects of carrying out research that I could have not learned otherwise.

Last and surely not the least, I would like to thank my family, Rajbir, Leo, Sunny and other friends for their consistent support and words of positivity throughout the whole thesis process.

References

Almeida, C., Fan, J., Freire, G., and Tang, F. (2022). Can a machine correct option pricing models? *Journal of Business and Economic Statistics*. Publisher Copyright: © 2022 American Statistical Association.

Andersen, T., Fusari, N., and Todorov, V. (2015). The risk premia embedded in index options. *Journal of Financial Economics*, 117(3):558–584.

Ang, A., Bali, T., and Cakici, N. (2012). The joint cross section of stocks and options. *The Journal of Finance*, 69.

Bakshi, G., Cao, C., and Chen, Z. (2000). Do call prices and the underlying stock always move in the same direction? *Review of Financial Studies*, 13(3):549–584.

Bauer, R., Cosemans, M., and Eichholtz, P. (2008). Option trading and individual investor performance. *Journal of Banking Finance*, 33:731–746.

Bernales, A. and Guidolin, M. (2014). Can we forecast the implied volatility surface dynamics for cboe equity options? predictability and economic value tests. *Journal of Banking Finance*, 46:326342.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Clarendon Press.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–54.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327.

Carr, P. and Wu, L. (2016). Analyzing volatility risk and risk premium in option contracts: A new theory. *Journal of Financial Economics*, 120(1):1–20.

Cerqueira, V., Torgo, L., Smailovic, J., and Mozetic, I. (2017). A comparative study of performance estimation methods for time series forecasting. pages 529–538.

Coleman, T. and li, Y. (1970). A newton method for american option pricing. *The Journal of Computational Finance*, 5.

Diebold, F. and Mariano, R. (1995). Comparing predictive accuracy. *Journal of Business Economic Statistics*, 13(3):253–63.

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

Dumas, B., Fleming, J., and Whaley, R. E. (1998). Implied volatility functions: Empirical tests. *The Journal of Finance*, 53(6):2059–2106.

Garcia, R. and Gencay, R. (2000). Pricing and hedging derivative securities with neural networks and a homogeneity hint. *Journal of Econometrics*, 94(1-2):93–115.

Geron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*. O'Reilly Media, Sebastopol, CA, 2nd edition.

Geurts, P., Ernst, D., and Wehenkel, L. (2002). Benchmarking decision tree learners. In *European conference on principles of data mining and knowledge discovery*, pages 9–18. Springer.

Goncalves, S. and Guidolin, M. (2006). Predictable dynamics in the sp 500 index options implied volatility surface. *The Journal of Business*, 79(3):1591–1636.

Gu, S., Kelly, B., and Xiu, D. (2020). Empirical Asset Pricing via Machine Learning. *The Review of Financial Studies*, 33(5):2223–2273.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

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.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Hutchinson, J. M., Lo, A., and Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, 49(3):851–89.

Israel, R., Kelly, B., and Moskowitz, T. (2020). Can machines 'learn' finance? *SSRN Electronic Journal*.

Joachims, T. (2006). Training linear svms in linear time. volume 2006, pages 217–226.

Masters, T. (1993). *Practical Neural Network Recipes in C++*. Academic Press, San Diego, CA.

Nicolai Meinshausen and Peter Bühlmann (2010). Stability selection (with discussion and rejoinder by the authors). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

Rubinstein, M. (1994). Implied binomial trees. *Journal of Finance*, 49(3):771–818.

Appendix

A: List of Features

Table 7: List of Features Used in SSO (and SPX) IVS Modeling

Feature	**Feature Type**
Moneyness (m)	Option specific
Time to expiry (τ)	Option specific
Call indicator	Option specific
Previous day weighted IV	Option specific
Bid ask spread of option	Option specific
Dollar volume traded of option	Option specific
Greeks	Option specific
Open interest	Option specific
Underlying daily return direction indicator	Stock specific
Underlying daily return squared	Stock specific
Underlying 5-day return	Stock specific
Underlying 5-day rolling sum return squared	Stock specific
Underlying 5-day beta	Stock specific
Underlying Hi - Lo	Stock specific
Underlying bid ask spread	Stock specific
Underlying dollar traded volume	Stock specific
Underlying market cap	Stock specific
Underlying industry indicator	Stock specific
SPX daily return direction indicator	SPX specific
SPX daily return squared	SPX specific
SPX 5-day return	SPX specific
SPX 5-day rolling sum return squared	SPX specific
SPX previous day weighted IV	SPX specific
VIX	SPX specific
US Fed Funds rate	Macroeconomic
Recession indicator	Macroeconomic
Risk-free interest rate	Macroeconomic
Gold price	Macroeconomic

Note: The table lists all the features and their type used in our IVS research analysis. For SPX IVS analysis, all the stock specific features stand invalid and hence were excluded during SPX IVS Modeling. We have 8 option specific features, 10 stock specific features, 6 SPX (index) specific features and 4 macroeconomic features resulting in total 28 features (and 18 during SPX IVS analysis).

B: Hyperparameter Tuning

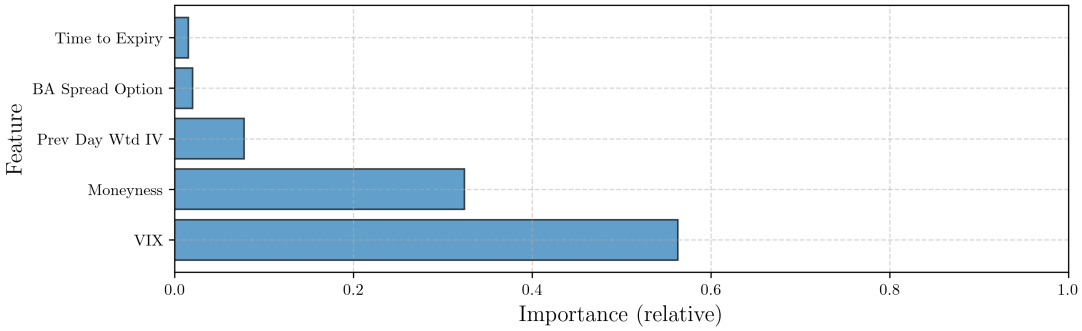
Table 8 shows the set (or range) of hyperparameter choices included for each ML model in our study. The optimal hyperparameter values fall within these settings and were utilized to train the optimal hypertuned model.

Table 8: Hyperparameter Grid Used for Tuning

ENet	RF	XGBoost	NN
$\lambda \in (10^{-4}, 10^{-1})$ $\alpha \in [0, 1]$	Depth $= 1 \sim 10$ #Trees $\in (100, 400)$ #Features in each split $\in (1, 12)$	Depth $= 1 \sim 6$ #Estimators $\in (50, 400)$ Learning rate: $\in (0.05, 0.3)$ $\alpha \in (0.05, 0.3)$	#Hidden Layers $= 1 \sim 5$ Solver: lbfgs Learning rate init: $\in \{0.001, 0.05, 0.01\}$ Learning rate: {constant, adaptive} $\lambda \in (10^{-5}, 10^{-3})$ Activation: ReLU

C: Variable Selection for SPX IVS Modeling

Figure 16: Variable Selection via ENet for SPX IVS Modeling



Note: The figure shows the five most important variables (from the total of 18 features) ordered by their importance as selected by ENet for SPX IVS Modeling. The length of bars indicate the relative importance of corresponding selected variables.

D: Full vs Reduced IVS Forecast Comparison

Table 9: Full vs Reduced Comparison for All ML Models - SSO IVS

ML Model	R^2_{oos} **Full (%)**	R^2_{oos} **Reduced (%)**	**DM Test Statistic**
OLS	77.5%	77.9%	-1.58
ENet	78.0%	78.1%	-0.14
GLM	79.0%	79.4%	**-2.25**
RF	82.8%	83.0%	-0.28
XGB	84.6%	86.5%	**-6.92**
NN	83.5%	88.0%	**-10.21**

Note: The table lists R^2_{oos} of "full" and "reduced" along with the result of DM test between them for each ML model in our study for SSO IVS forecasting. The negative value of DM test statistic indicates that the reduced version outperformed the full version, which is the case here for each ML model. The DM test statistic in **bold** indicate that statistic was significant at 5% significance level i.e. p-value < 0.05.

Table 10: Full vs Reduced Comparison for All ML Models - SPX IVS

ML Model	R^2_{oos} Full (%)	R^2_{oos} Reduced (%)	DM Test Statistic
OLS	88.3%	88.5%	-1.4
ENet	88.4%	88.5%	-0.97
GLM	84.4%	89.6%	**-5.76**
RF	93.7%	93.8%	-0.06
XGB	94.1%	94.5%	**-2.67**
NN	95.2%	95.7%	**-2.72**

Note: The table lists R^2_{oos} of "full" and "reduced" along with the result of DM test between them for each ML model in our study for SPX IVS forecasting. The negative value of DM test statistic indicates that the reduced version outperformed the full version, which is the case here for each ML model. The DM test statistic in **bold** indicate that statistic was significant at 5% significance level i.e. p-value < 0.05.

E: SPX Options Summary Statistics

Table 11: Summary Statistics of SPX Implied Volatility Data

Time to expiry (τ): Short-term (20-60 days)

Moneyness (m)	**[0.80, 0.90) DOTMC**	**[0.90, 0.97) OTMC**	**[0.97, 1.03) ATM**	**[1.03, 1.10) OTMP**	**[1.10, 1.20] DOTMP**	**Total**
Count	27,260	147,086	197,454	179,874	149,053	700,727
Mean (%)	22.60	14.86	17.00	14.86	22.56	-
Std. dev. (%)	9.21	6.56	6.79	6.92	7.38	-

Time to expiry (τ): Long-term (60-240 days)

Moneyness (m)	**[0.80, 0.90) DOTMC**	**[0.90, 0.97) OTMC**	**[0.97, 1.03) ATM**	**[1.03, 1.10) OTMP**	**[1.10, 1.20] DOTMP**	**Total**
Count	30,854	119,488	195,773	166,633	125,599	638,347
Mean (%)	20.43	15.99	17.11	15.99	23.16	-
Std. dev. (%)	9.39	7.05	7.04	7.25	8.04	-
Total Count	58,114	266,574	393,227	346,507	274,652	**1,339,074**

Note: The above table shows summary statistics for our SPX IV data sample which ranges from Jan 1, 2019 to Dec 31, 2021. We refine the summary statistics across different moneyness (m) and time to expiry (τ) categories. For each category we show the number of options falling in that category, mean IV in that category and standard deviation of IV in that category. We also show the total count of options.