Random Forest for Digital Bermudian Option Pricing
Machine Learning Term Project, Spring 2019

Anthony Maylath; aem578

Abstract

The Longstaff Schwartz Method (LSM) (2) has emerged as the industry standard for pricing American and Bermudian style stock options. The method leverages least squared regression over a polynomial basis to estimate optimal stopping times and works well for stock options with reasonably smooth payoffs. However, polynomial regression can incur a larger error if the payoff is digital. For such payoffs I suggest a random forest prediction in lieu of polynomial regression. To test the proposed method, I show convergence for the Random Forest LSM method (LSMRF) compared to Monte Carlo Tree estimations and I compare the training and test regression errros between standard LSM and LSMRF for two digital style options. The results suggest LSMRF outperforms standard LSM for digital style options.

1. Introduction

A financial *call* option gives the holder the right to purchase a particular stock at a particular *strike* price. Likewise, a *put* option gives the holder the right to sell the stock. An *American* option allows the holder to exersise this right at any time during the term while a *European* option restricts the right to the last day of the contract only. A *Bermudian* option is in between American and Euorpean styles, allowing the holder to exercise only on specified *exercise dates*.

Since the advent of the Black Scholes formula (1), valuing a European option is now trivial and can be done in a few lines of python code. However, American options remain difficult to value. The *Longstaff Schwartz* model (LSM) (2) has emerged as the "state of the art" method to value American and Bermudian options.

At a high level, the LSM method models a stock as a geometric brownian motion and generates hypothetical paths of the stock price via Monte Carlo Simulation. After several paths are generated, LSM walks back along each path and attempts to predict the payoff at the next time step with the stock price at the current time step. The prediction is done via least squares regression and informs the option holder if he or she should exercise or hold the option longer.

In this paper, I will review the standard LSM method and introduce an extension to the method which replaces least squared with random forests. Such an extension is already an active area of research among many banks and hedge funds, but has a limited number of published resources (5), (6) [1] and almost nothing relating specificaly to digital options. I test the LSM random forests (LSMRF) method on *digital* style options which have payoffs that look like a step function. The expectation is that random forest should be able to fit a step function better than a polynomial.

2. Monte Carlo Methods for Option Pricing

2.1. Monte Carlo for European Options

Before introducing the LSM method, I will quickly review Monte Carlo for European options. There are five main factors which determine an option's price: 1) today's stock price, S_0, 2) the strike price, K, 3) the stock's volatility, σ, 4) the interest rate, r, 5) the time to maturity, T. I will assume the interest rate and volatility are constant throughout this paper. We take r and σ to be the rate of interest and volatility for one time step respectively. [2]

Notice the mean of the stock price is not an input for the option's price. Instead of the mean, we assume the stock increases at the same rate as the interest rate, r. This is a better assumption than considering the mean as stock returns are unpredictable while we can generally lend the stock out for a fixed rate.

[1] The Gramacy paper was the closest resource I could find, but even that is a bit far from the topic of this paper. There are also a few resources that use random forest to predict American option prices from exchanged listed European prices (7). However, this method completely throws out the Monte Carlo approch and may not be relavant for non-listed products.

[2] Hence, set $r = r_{annual}/S$ and $\sigma = \sigma_{annual}/\sqrt{S}$ where S is the time steps per year.

The price of a European put option is simply the expectation of the final payout discounted back to today: $Put(S_0, K, \sigma, r, T) = E[exp(-rT)max(K - S_T, 0)]$ where S_T is the stock price at the end of the contract. Assume the stock price follows a geometric brownian motion with mean r and variance σ^2. Hence the stock price at time $t \in (0, T]$ is:

$$S_t = S_0 exp\Big(\sigma\sqrt{t}Z + \Big(r - \frac{\sigma^2}{2}\Big)t\Big) \tag{1}$$

where $\sqrt{t}Z$ is $N(0, t)$ and represents a brownian motion. Later on, we will need to obtain S_t for $t \in (0, T)$, but for now we can directly evaluate S_T and plug it into $exp(-rT)max(K - S_T, 0)$. Hence, we can evaluate the price of the put by computing S_T repeadedly for different Z:

$$E[e^{-rT}max(K - S_T, 0)] \tag{2}$$

$$= e^{-rT}\sum_{i=1}^{N} max(K - S_{T,Z_i}, 0) \tag{3}$$

2.2. Longstaff Schwartz Method for American Option Pricing

Longstaff Schwartz (LSM) depends on the intermediate values, S_t, for $t \in (0, T)$. We can no longer directly obtain S_T as we did before. Recall an American option can be exercised at *any* time. Hence, we need to find the optimial exercise time when evaluating the expectation:

$$Put_A(S_0, K, \sigma, r, T) = \sup_{t^* \in [0,T]} E[e^{-rt^*}max(K - S^*_t, 0)] \tag{4}$$

To cope with the supremum, start by generating a set of S_T like we did before. However, this time, also compute in between values for $S_t,\ t \in (0, T)$. At the end of the simulation, we have N stock prices, $S_{t,i}$, for each time step $t \in [0, T]$, where i represents the path index.

For each time step and path, we have to decide whether its better to exercise now or wait. The value of waiting is known as the *Continuation Value* and can be computed by valuing an an American option just after time t:

$$C(S_{t,i}) = \sup_{t^* \in (t,T]} E[e^{-r(t^*-t)}max(K - S_{t^*,i}, 0)|S_{t,i}] \tag{5}$$

Likewise, the value of exercising at time t is simply the current payoff, $max(K - S_{t,i}, 0)$. The LSM method uses least squares regression to estimate (5).

Let's consider $S_{t,i}$ for $t = T - 1, T$. For all paths, set $P_{T,i} = max(K - S_{T,i}, 0)$. Using $S_{T-1,i}$ for $i = 1, ..., N$, regress the vector of current stock prices, S_{T-1}, against the vector of expected future payouts, $exp(-r)P_T$. Then, estimate the continuation values, $C_{T-1,i}$, with the resulting regression function. Set $P_{T-1,i} = max(C_{T-1,i}, max(K - S_{T-1,i}, 0))$, then repeat the process by estimating $P_{T-1,i}$ with $S_{T-2,i}$. Repeat the algorithm until $t = 0$, and set $Put_A(S_0, K, \sigma, r, T) = \frac{1}{N}\sum_{i=1}^{N} P_{0,i}$ to obtain the price.

At each time step, we compare the value of continuing, $C(S_{T-1,i})$, to value exercising now, $max(K - S_{T-1,i}, 0)$. If the value of continuing is larger, set $P_{T-1,i} = C(S_{T-1,i})$ as it is better to wait. Otherwise, set $P_{T-1,i} = K - S_{T-1,i}$ as it is better to exercise now. Note that if $max(K - S_{t,i}, 0) = 0$, we always wait. Hence, we exclude paths with zero immediate payoff from the regression as it is always optimal to continue with zero immediate payoff. This procedure of finding continuation values is repeated for every time step back to $t = 0$. The result will give us the best exercise time for each path. Once we hit the best exercise time on a given path, the continuation value will always be higher as we walk back to time 0.

3. Random Forest Longstaff Schwartz and Digital Options

When using Monte Carlo to evaluate the expectation in (4), there are several places where we make approximation errors. First, we only consider a finite number of paths to approximate the full distribution. Second, we only use a finite number of timesteps to represent every instant between $[0, T]$. Third, we approximate $C(S_{t,i})$ with a regression function. We could also incur error by using psudo random variables rather than *real* random variables (3). Here, we focus mainly on the error made when approximating $C(S_{t,i})$.

Typically, we use the first n terms in a orthogonal basis for the explanatory variables in the LSM least squares regression. To start, let's take the first n terms of the standard basis of polynomials as our input variables for the regression: $\{S^k_{t,i}\}^n_{k=0}$. Let $\mathbf{S}_t$ be a matrix with $[1, S_{t,i}, S^2_{t,i}, ..., S^n_{t,i}]$ in the i^{th} row. Likewise, let $\mathbf{P}_{t+1}$ be a vector with elements $exp(-r)P_{t+1,i}$ for $i = 1, ..., N$ number of paths. Perform least squares to obtain regression parameters:

$$\beta_t = (\mathbf{S}_t^T \mathbf{S}_t)^{-1} \mathbf{S}_t^T \mathbf{P}_{t+1} \tag{6}$$

Then, the error is $\langle [1, S_{t,i}, S_{t,i}^2, ..., S_{t,i}^n], \beta_t \rangle - exp(-r) P_{t+1,i}$ (with $\langle x, y \rangle$ the dot product) for path i. Figure 1 shows the the regression result for step 2 of a Bermudian Put simmulation with 10 steps and 2000 paths. The graph looks promising. The orange regression line seems capture the shape of the blue points. Figure 2 represents the initial payoff function which the regression function estimates for Bermudian puts. The payoffs are scaled by $exp(-r)$ with each timestep and alternate between $C(S_{t,i})$ and the immediate payoff, but tend to have a similar shape accross time.

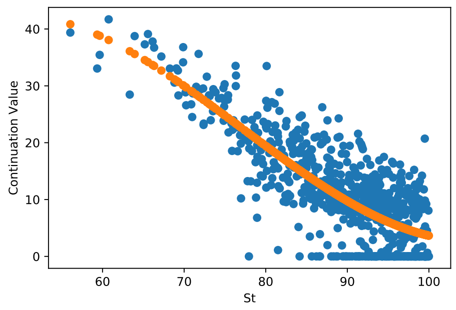

Figure 1. Regression for Vanilla Put

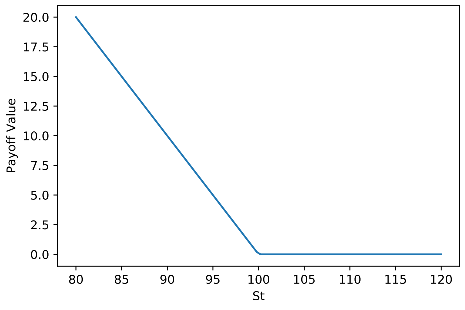

Figure 2. American Put Payoff Function

Consider a financial product, similar to an Bermudian put option, but instead of recieving $max(0, K - S_t)$, the holder gets $max(0, 1(K > S_t))$. A graph of the payoff can be found in figure 4. Payoffs with such a form are called *digital* options as the payoff value is either one or zero. Notice it is *never* optimal to continue to hold a digital put when its payoff is non-zero. We could therefore generate the continuation values while the paths are generated instead of performing the walk back regression in LSM.

Even though Monte Carlo regression is not needed to price Bermudian Digital puts we will analyze the LSM method on this product to develop intuition. Figure 3 shows the regression result for step 8 of an Bermudian digital put with 10 timesteps, 2000 paths and $K = 100$. The regression uses the first four elements of the standard basis for the training data. Clearly, the method does not perform as well as Figure 1. Note zero values for the blue dots represent paths where $K > S_t$, but $K < S_T$.

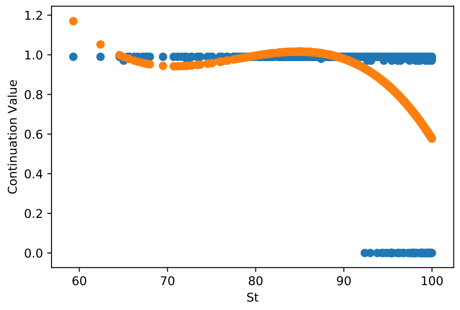

Figure 3. Regression for Digital Put

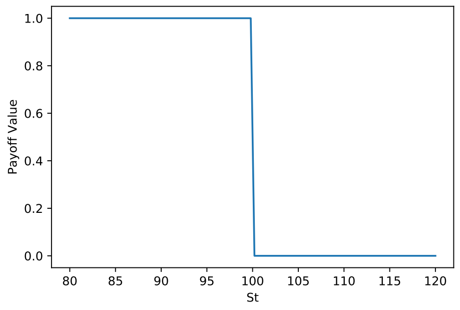

Figure 4. American Digital Put Payoff

As an alternative to our least squares approch, Figure 5 shows the estimated continuation values with a single CART decision tree of depth 4. Intuitively, a decision tree should perform better as the payoff of a digital option is an instance of a decision tree itself. The prediction values for small S_t match exactly which is much better than least squares. However, the CART tree seems to have trouble making predictions around the strike price ($K = 100$). It can be argued that the smoothness of least squares might be better in this region. However, since there are only about $50/2000 \approx 0.04$ blue labels with zero value, we want the prediction closer to 1 than zero near S_t = 100. As a hybrid approch, Figure 6 displays the fit with 10 random forest trees with maximum depth of 4 and a minimum of 10% of samples per node. For the implementation, I use sklearn RandomForestRegressor.

The random forest fit seems better as the fit near $S_t = 0$ is now somewhat smooth near the strike price as many trees average together.

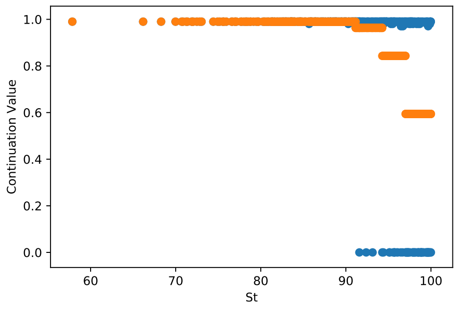

Figure 5. Decistion Tree for Digital Put

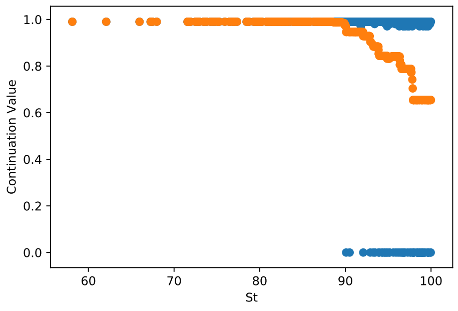

Figure 6. Random Forest Prediction for Digital Put

4. Exotic Payoffs

Since it is always optimal to exercise a digital option when the stock price is smaller than the strike price, the LSM method does not add much value. Hence, we seek a different financial product with digital payoff where the continuation values are a bit harder to obtain.

A *Weding Cake* option pays a fixed amount if the stock price is within a certain range within the strike price. The ranges occur in layers and typically decrease the further away the stock price moves from the strike price. Figure 7 shows an example of a Wedding Cake payoff with a strike price of 100. If the stock price is within 5$ of the strike, then the option pays 1$. If the stock is within 10$ but not within 5$, the option pays 0.5$. If the stock is within 15$ with further than 10$, the option pays 0.25$. Otherwise, the option pays nothing. Notice the payoff has the shape of a tiered cake typically used in weddings. If the stock price ever comes within 5$ of the strike, it is always optimal to exercise immediately. If the stock price is further than 15$ from the strike, then it is always optimal to continue. However, if the stock price is more than 5$ away from the strike, but within 15$ of the strike, it is unclear if the holder should exercise or wait. For this case, we will need the LSM mehod to compute the optimal exercise time.

A *Double Digital* option has two strike prices. If the stock is lower than the lower of the two strikes, the payoff is 2. If the stock is in between the two strikes and below the higher of the two strikes, the payoff is 1. Otherwise, the payoff is 0. Figure 8 plots the payoff of a double digital. Similar to the wedding cake payoff, if the stock is in between the two strikes, then we must compute continuation values. The analysis that follows will look at the performance of LSMRF on both the Wedding Cake and Double Digital payoffs.

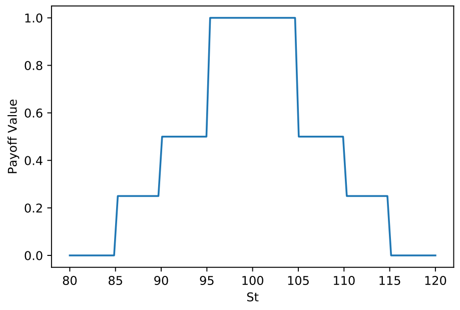

Figure 7. Wedding Cake Payoff

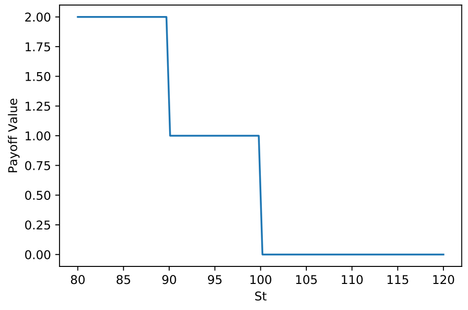

Figure 8. Double Digital Payoff

4.1. Convergence

To verify convergence of the LSMRF method, we must compare the price obtained by our method to the *true price*. As we already know that estimating the continuation values with regression introduces new errors, we will obtain the true price via Monte Carlo Tree Estimation [3] (3), which

[3]The Monte Carlo Tree approch also introduces errors when estimating continuation values. However, the new errors come

does not use a regression approximation. Such a technique obtains continuation values by spawning new paths every time the holder of the option can exercise. The method is intractable for many continuation dates as each path will spawn many new paths for each time step, making the computation exponential in the number of starting paths. Therefore, we will examine convergence with only *one* exersise date over the period of two years. This means the holder only has one day over the next two years where he or she must compute continuation values.

I compute the basline price with an average of up to 250 samples, each with 2,000,000 paths and 1,000,000 spawned paths per continuation value. See Appendix B [4] for the remaining parameters on the baseline simulations and convergence curves. Figure 9 demonstrates the convergence of the wedding cake and double digital payoffs. The number of paths goes up to a relatively small 40,000. and the methods already achieves 10^{-3} accuracy. For 10 million paths, my implementation achieves an error around 10^{-5}. This suggests LSMRF is an appropreate method to price Bermudian style wedding cake and double digital options.

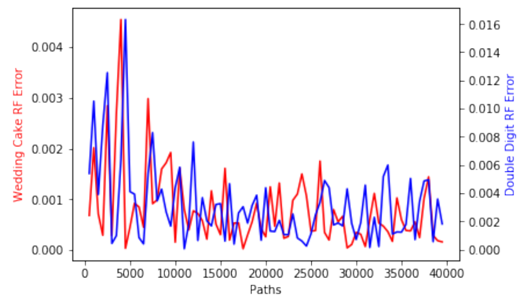

Figure 9. LSMRF Convergence for Wedding Cake and Double Digit

4.2. Comparing Random Forest to Polynomial Regression

We will compare the performance of LSMRF to the standard LSM method described in section 3. We will make a slight tweak by using the first five elements in the Laguerre Basis instead of the standard basis as the former tends to be more accurate and comes with relatively low cost. Since we are only comparing the regression piece of the algorithm, we will focus on comparing regression errors between the two models. Otherwise, the Monte Carlo sampling error may confuse the results. We will keep the

Table 1. Regression Errors at Each Exercise Date for Wedding Cake

LSM STEP	WC LSM	WC RANDOM FOREST	WC IMPROVEMENT
1	0.265676	0.264408	0.001268
2	0.285662	0.286189	-0.000527
3	0.299417	0.297026	0.002391
AVG	0.283585	0.282541	0.001044

number of exercise dates between weekly and quarterly. [5]

An important consideration when fitting the regression model is the settings of random forest parameters. Recall the random forest fit in figures 5 and 6 and how increasing the number of trees around the strike improved results. Since the projected payoffs are likely to straddle two values when the stock is near a strike price, more trees are needed to smooth out the point of singularity. In my analysis, I found 50 trees does a sufficiently good job smoothing. To control overfitting, I set the minimum samples per leaf between 0.01% and 1%. Any higher and the random forest trees tend to smooth the whole support into one average. Since the double digital payoff only has three values, I limit the maximum depth to 3. The fit for the wedding cake payoff is allowed to reach depth 5 as the payoff is slightly more complex. Appendix A contains results across different random forest parameters.

Tables 1 and 2 show the mean absolute regression errors for each LSM step for initial stock price = 94.5 and quarterly exercises. We now assume a maturity of one year. Full list of parameters can be found in the Appendix. The LSMRF and the Polynomial LSM seem to yield comprable regression errors with a slight advantage given to the polynomial regression for the double digital. The results are a bit surprising as figures 5 and 6 suggest the RF regression should be significantly better. The issue occurs because random forest cannot distingish between two payouts if the stock is near a strike price. Essentially, the regression problem is not well posed as there are several Y values for the same X. Figure 10 illistrates the point for an LSM step on the wedding cake. The polynomial fit suffers from the same issue, but is able to perform slightly better near the strike as it is more flexable.

One remedy to improve the random forest fit is to decrease the volatility and interest rate parameters. With lower volatility and rates, the stock cannot move as quickly and therefore the payoffs around a strike price become more

from Monte Carlo path estimation, which is already present in the baseline method for valuing European options. We take a significant number of paths to minimize this error.

[4]The Appendix is in a different document called, "Appendix.pdf." It can be found in the companion git repository.

[5]Exotic digital products typically have between weekly and quarterly exercise dates. More than weekly exercise dates is operationally difficult for a bank to handle.

Table 2. Regression Errors at Each Exercise Date for Double Digital

LSM Step	DD Baseline	DD RF	DD Improvement
1	0.539953	0.536331	0.003622
2	0.477726	0.483434	-0.005708
3	0.431221	0.431035	0.000186
Avg	0.48297	0.4836	-0.0006

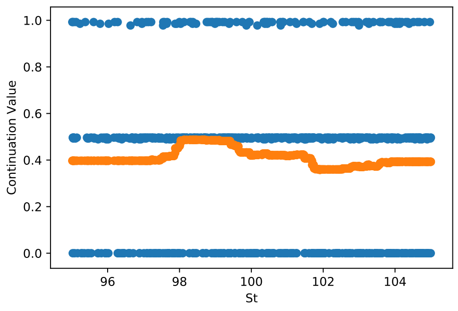

Figure 10. Wedding Cake Regression Fit

defined. However, the volatility and rates are observed in the market and generally cannot be changed by a financial engineer to improve pricing performance. As a workaround, we can increase the number exercise dates which causes the volatility and rate to be scaled down as the time steps becomes larger.

To further improve the performance of LSMRF we can move the strikes further apart. In figure 10, we see even if the stock is very far in the lowest payout range, there are still payouts at the next time step which achieves the maximum payout. If the stike was further away the stock would need to travel much further, making the scenario less likely. Consider a double digital with a lower strike of 75 and an upper strike of 120. Such a payoff is very popular among exotic derivatives desks and is usually baked into *Autocallable* payoffs. A swap dealer would buy the payoff from the client. The client would get upside gain while the dealer would get downside protection. The initial stock price would usually be between the two strikes.

Tables 3 and 4 show the average errors between the baseline method and random forest for quarterly, monthly, and weekly exercise dates respectively. We also move the strike prices further apart to highlight the impact. The LSMRF now beats the baseline LSM in across the board. As a comparison, Table 5 shows the regression errors for double digital with the same parameters but close strikes. There still seems to be an improvement when the client has more exercise dates, but it is smaller than the case with far strikes.

Table 3. Regression Errors by Timestep for Wedding Cake - Far Strikes

TimeSteps	WC LSM	WC Random Forest	Improvement WC
Quarterly	0.099282	0.098192	0.00109
Monthly	0.07695	0.073791	0.003159
Weekly	0.085752	0.082098	0.003654

Table 4. Regression Errors by Timestep for Double Digital - Far Strikes

TimeSteps	DD LSM	Random Forest DD	Improvement DD
Quarterly	0.324702	0.319951	0.004751
Monthly	0.309063	0.288816	0.020247
Weekly	0.274474	0.230078	0.044396

The results were similar for the wedding cake payoff.

Finally, we need to examine how the random forest performs on *test data*. So far, we just compared training error of the standard LSM to LSMRF. To obtain a “true” continuation value, we will use the Monte Carlo Tree Estimation from section 4.1. To keep the computation tractable, we will consider products with weekly exercise dates but only two weeks until maturity. The arrangement allows us to scale down the volatility and rate parameters but allows us to limit Monte Carlo Tree Estimation to only two continuation values. I will take the continuation value given by the Tree Estimation as the “true” value. Tables 6 and 7 show the results with both test and training error. Both the training error and the test error are lower with LSMRF compared to standard LSM. Parameters can be found in Appendix B.

5. Concluding Remarks

Bermudian style options present challanges for valuation as a methods need to compute continuation values at every exercise date to obtain a price. The LSM method has emerged as the “state of the art” tool to efficiently compute continuation values. However, the polynomial regression of LSM provides unappealing results when fitting continuation values for optons with digital payoffs.

Matching the shape of a digital payoff to the structure of trees, we suggest a random forest regressor to better predict LSM continuation values for digital options. To ensure fedelity of the LSMRF method, we demonstrate price convergence when compared to the “true” price computed with a large sample of monte carlo tree simulations. We then compared regression errors of standard LSM to LSMRF. We found LSMRF to be comprable to standard LSM when the strike prices are close together and when there are few exercise dates. However, LSMRF tends have

Table 5. Regression Errors by Timestep for Double Digital - Close Strikes

TimeSteps	DD LSM	Random Forest DD	Improvement DD
Quarterly	0.482253	0.484866	-0.002613
Monthly	0.374536	0.364623	0.009913
Weekly	0.304125	0.297083	0.007042

Table 6. Mean Absolute Errors Against the Monte Carlo Tree Method - Wedding Cake

TimeSteps	WC LSM	Random Forest WC	WC Improvement
Test	0.040996	0.04016	0.000836
Training	0.052434	0.051018	0.001416

smaller regression errors when a product has many exercise dates and when the strike prices are far apart. The analysis suggests LSMRF should be the method of choice when valuing Bermudien style digital options as the worst the LSMRF can perform is comparable to standard LSM.

Since the random forest method tends to perform well in high dimensional space, it would be interesting to extend the analysis to digital options with *multiple* underlyers. Multi-underlying options are very popular products among exotic derivative desks as they generate high fees for the seller and can provide huge returns for the buyer. Typically, payoffs have a digital form, but require *all underlying stock prices* to fall below the strike. For instance, if the underlyers are Apple, Amazon, and Facebook with a stike price of 80%, then all three stocks have to fall by 20% before the payoff triggers.

Acknowledgements

I would like to thank the Applied Research and Development team at Société Générale for suggesting the topic of study and providing feedback for my results. In particular, Hugo Para has been very helpful.

Table 7. Mean Absolute Errors Against the Monte Carlo Tree Method - Double Digital

TimeSteps	DD LSM	Random Forest DD	Improvement DD
Test	0.100562	0.091774	0.008788
Training	0.118959	0.104728	0.014231

References

[1] Fischer Black and Myron Scholes, *The Pricing of Options and Corporate Liabilities*. The Journal of Political Economy, Vol. 81, No. 3 (May - Jun., 1973), pp. 637-654

[2] Francis A. Longstaff and Eduardo S. Schwartz, *Valuing American Options by Simulation: A Simple Least-Squares Approch*. The Review of Financial Studies, Volume 14, Issue 1, January 2001, Pages 113–147

[3] Paul Glasserman, *Monte Carlo Methods in Financial Engineering*. Springer Science + Business Media, Inc. 233 Spring Street, New York, NY 10013 USA, 2003.

[4] Willian Gustafsson, *Evaluating the Longstaff-Schwartz method for pricing of American options*. Department of Mathematics, Uppsala Universiry, June, 2015

[5] Robert B. Gramacy and Michael Ludkovski, *Sequential Design for Optimal Stopping Problems*. arXiv:1309.3832 [q-fin.CP] July, 2014

[6] Bella Dubrov, *Monte Carlo Simulation with Machine Learning for Pricing American Options and Convertible Bond*. aAvailable at SSRN: https://ssrn.com/abstract=2684523 or http://dx.doi.org/10.2139/ssrn.2684523 Novembe, 2015

[7] Munira Shahir, *Predicting Call Option Prices Using Regression Models*. Mathematics Department, Carnagie Mellon Univeristy, July, 2014