A Machine Learning Approach to Forecast Bitcoin Prices

Amitha Raghava-Raju
Software Engineer
Societe Generale Global Solution Center
#19, 5th cross, 3rd main, Pampa Extension
Hebbal Kempapura, Bangalore – 560024

ABSTRACT

Bitcoin is an established cryptographic digital currency whose value lays in the computational complexity rather than a physical commodity. Bitcoin is an open source software program with three aspects. (i) Peer-to-Peer network – low barrier entry; (ii) Mining – inevitable concentration of power; (iii) Software upgrades. The nodes on the network follow a decentralized consensus for establishing the value of ledger and updating the blockchain which serves as a single source of truth for all transactions. As cryptocurrencies are developing more compelling utilities, creating ever faster and safer payment systems they are shifting the "money paradigm". Bitcoins are an evolution in money and provide a unique opportunity to forecast their price unlike the existing fiat currencies. The goal of this paper is to implement, train and evaluate several machine learning models in order to predict the price of the most popular cryptocurrency – Bitcoins. The various machine learning algorithms employed are – Linear Regression, K-Nearest Neighbors, Ridge Regression, Lasso Regression, Polynomial Regression, Linear Support Vector Machine, and Kernel Support Vector Machine.

General Terms

Bitcoins, Cryptocurrency, Blockchain, Machine Learning, Regression, Forecasting.

Keywords

Decentralized Consensus, Bitcoin Prediction, Linear Regression, K-Nearest Neighbors, Ridge Regression, Lasso Regression, Polynomial Regression, Support Vector Machine, Residual Sum of Squares, Peer-to-Peer network, L1 Regularization, L2 Regularization.

1. INTRODUCTION

Bitcoin is a novel cryptocurrency that was established in 2009. Bitcoin transactions take place on the blockchain that enables the users to be anonymous. The blockchain technology is powered by a network of computers that follow a decentralized protocol through which the nodes reach a consensus and verify every bitcoin transaction that occurs. The nodes are independently and constantly solving the "hash puzzles" in order to identify and authentic transactions and append them to the block chain. This process is termed as mining and the nodes are rewarded with Bitcoins for their tedious efforts. Bitcoin is a decentralized peer-to-peer network with nodes across the globe [1].

Fiat currencies are regulated by the government and have central authority to establish the policies to achieve economic stability, growth and currency value. Bitcoins, on the other hand, by design take a decentralized approach. There is no middle man in this system. The monetary value of the Bitcoin is influenced by various factors. (i) Limited Supply – the total number of Bitcoin is limited to 21 million, this has a huge influence on the supply and demand of the Bitcoins. (2) Prestige/Credibility – Bitcoin is a "grand" financial experiment and widely recognized digital currency. (3) Media – positive news coverage leads to potential surge in Bitcoin demand. (4) Access – Bitcoin can be traded in cryptocurrency exchanges around the world. (5) Acceptance – Increase in businesses and service providers accepting Bitcoin as a payment and financial services and offering Bitcoin trading options could drive the Bitcoin demand. (6) Digital evolution – better hardware, improving computation speed, enabling safer transactions, reduced fees and accelerated transaction window increase Bitcoin demand [2].

The significant contributions of this paper are as follows. Firstly, identify the factors that influence the market price of the Bitcoin. Secondly, utilize several machine learning approaches to train the model and predict the Bitcoin price given the features of importance. Thirdly, apply the trained model to evaluate its performance on the test set and identify the machine learning algorithm that provided the optimum result intended to forecast Bitcoin prices. Lastly, improve the algorithm by tuning the performance and consider a larger subset of factors influencing the Bitcoin price.

2. RELATED WORK

Bitcoin as a cipher currency was published in 2009 as a research work under the pseudo name Satoshi Nakamoto. Since its inception and rise of the social media the current research is mostly focused on classifying the sentiment of the population in order to identify the inclination of the public towards Bitcoins.

A thesis published by KTH Royal Institute of Technology, predicts Bitcoin prices using twitter sentiment analysis system. They examined 2.27 million Bit-coin related sentiments and attributed to the fluctuations in the Bitcoin price based on the severity of the twitter feed over periods ranging from 5 minutes to 4 hours. The model provided an accuracy of 79% [3].

Studies conducted on the time series model of the Bitcoin prices with specific technical rules is the new market variable and its characteristics are considered a financial assets. It utilizes the GARCH model to explore the time series of the Bitcoin. The GARCH (Generalized Autoregressive Conditional Heteroskedasticity) process helps describe financial market fluctuations and volatile climate.

Certain studies are conducted on estimation of Bitcoin prices using categorized financial information. A linear model is built to predict the Bitcoin price based on data that is clustered into several market forces, investor interests, and global market-financial factors. The model assumes that market forces and investor interests dominant the Bitcoin price estimations when compared to global financial factors.

Bitcoin pricing process using machine learning techniques have also been conducted. Multiple deep learning techniques – Recurrent Neural Network (RNN), Long short-term memory (LSTM), Bayesian Neural Network, and Autoencoders have been utilized to train the model to forecast the Bitcoin prices [4].

The current study systematically identifies factors influencing Bitcoin fluctuations and builds multiple machine learning models to enhance the Bitcoin predictive performance.

3. DATASET AND FEATURES

The Kaggle Bitcoin Data was considered for this study. This dataset was utilized to make inference from, as it was the most available and had ample number of features and adequate samples to draw conclusive inference.

3.1 Features

The Bitcoin dataset consists of 24 features. The features include date, market price, total bitcoins mined, market cap/supply, trade volume, block size, average block size, number of orphaned blocks, number of transactions per block, median confirmation time, hash rate, difficulty, miners revenue, transaction fee, cost per transaction, unique addresses, number of transactions, total transactions, number of transaction excluding 100 popular addresses, number of transactions excluding chains longer than 100, output volume, estimated transaction volume, estimated transaction volume in USD value.

Not all features provide significant information to estimate Bitcoin prices. Basic visualization of the features provides vital insight to choose features that can affect forecasting process.

3.2 Preprocessing

The dataset consists of 3000 samples, 30 samples had missing feature bitcoin trade volume. It constituted 1% of the dataset. Figure1. Plot of Kaggle bitcoin dataset with missing data. Although it was possible to discard the records with missing trade volume, it could be possible that these records could provide significant data to enhance the Bitcoin forecasting process.

Therefore previous value imputation technique was employed to treat the missing values. Figure 2 - 4 show visualization plots to identify column-wise missing data.

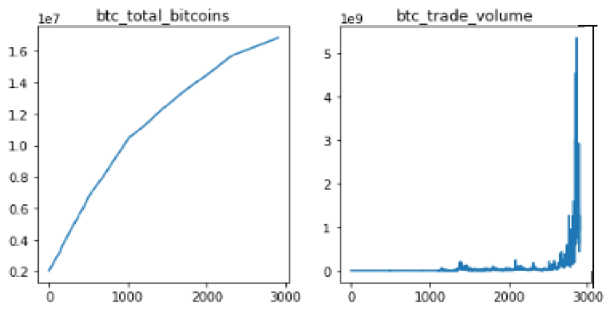


Figure 2: Total bitcoin and Trade Volume Missing Data Plot

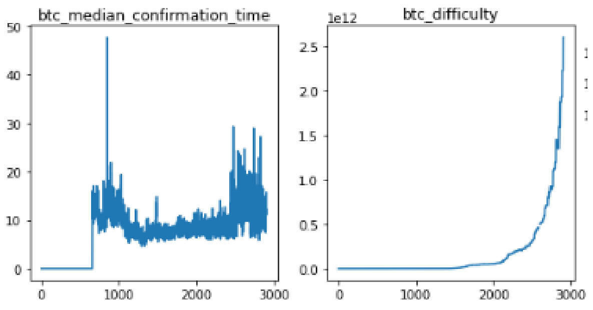
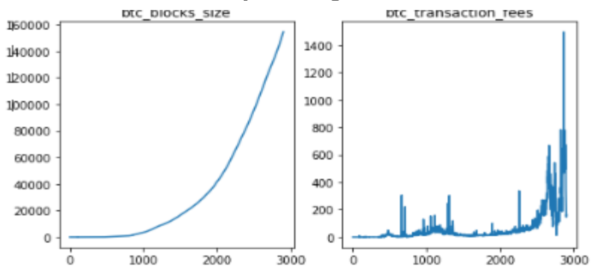


Figure 3: Median Confirmation Time and Bitcoin Difficulty Missing Data Plot



ssing

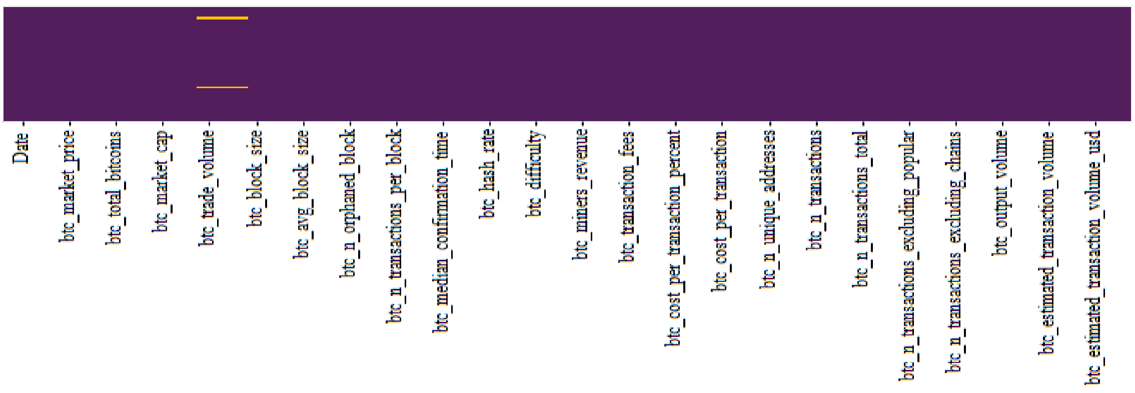


Figure 1: Kaggle bitcoin dataset with missing data

Furthermore the dataset was tested for correlation of the target variables with other predictor variables. Columns with high correlation can be removed as it otherwise could lead to overfitting the model.

From the correlation plot the following inferences can be drawn. The 'Date' feature does not contribute to the 'Bitcoin Market Price' hence can be ignored. Moreover features 'Market Cap' and 'Miners Revenue' are highly correlated with the target variable 'Bitcoin Market Price' so these columns must be removed from the dataset as it could affect the accuracy of the model and the machine learning model will overfit.

Additionally, since the column dataset has high variability it must be scaled. The min-max scaling method is applied on the dataset. Finally from the processed points, the data was randomly split into 80% training set and the remaining 20% was for testing.

4. MACHINE LEARNING MODELS

The goal of this paper is to build machine learning models to predict the price of the bitcoin given its volatile nature when compared to fiat currency. Seven models were built to train the data and thereafter evaluate the performance on the test set [5].

4.1 Linear Regression Model

Regression analysis is an important tool for modeling and analyzing data. It is a form of predictive modeling technique to investigate the relationship between dependent (target) and independent variables (predictor). The model fits a curve/line through the data points in a manner that minimizes the differences between the distances of the data points. The Regression Machine Learning Block Diagram is shown in Figure 5.

Linear Regression establishes the relationship between the target (Y) and independent variables (X) using a best fit straight line. The model calculates the best-fit line for the observed data by minimizing the sum of the squares of the vertical deviations from each data-point to the line. The model performance is evaluated using the metrics RSS (Residual Sum of Squares). The equation of a linear regression model is given in Eq. (1).

$$y' = b + w_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + \varepsilon \quad (1)$$

Where y' is the predicted label, b is the bias (y-intercept),x_1, …,x_n are the multiple input features, w_1,…,w_n are the weights of the corresponding feature inputs, ε - epsilon – error term.

The Linear Regression model is implemented in the project to fit the best line with minimum cost through the dataset in order predict the target (btc_market_price) given the input features. For the given scenario the linear regression model performs the following operations.
i. The linear regression model fits a line through the dataset which can be represented by estimated parameters.
ii. Given the estimated parameters like slope, intercept and other coefficients compute the predicted value.
iii. Compute the Prediction error – which is the difference between original and predicted value.
iv. Update the coefficients – compute the derivative, adjustment of the step-size and decrease the slope accord to hill descent algorithm.
v. Compute the magnitude of the gradient as shown in Eq. 2

$$\sqrt{\sum_{i=1}^{N}(yi - h(xi)w)^2} \quad (2)$$

Where *yi* is the original data, *h(xi)* is the estimated output and *w* is the weight of each feature.
vi. Check for convergence. If the magnitude of the gradient is less than the tolerance factor then stop the computation process and return the model and the respective coefficients.

The model searches over all possible lines and finds the smallest possible residual sum of squares.

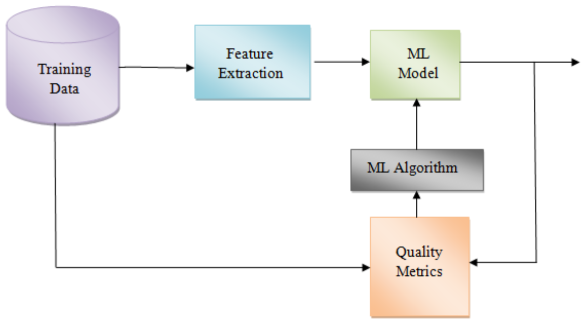


Figure 5: Regression Machine Learning Block Diagram

4.2 K-Nearest Neighbors Model

The KNN algorithm is a non-parametric and instance-based supervised learning algorithm. Non-parametric means it makes no explicit assumptions about the underlying data distribution. Instance-based learning means that the algorithm does not explicitly learn the model instead chooses to memorize the training instances which are subsequently used in the training phase. The similarity is defined by the distance metrics between the data points. Common choices for the data metrics are Euclidean, Manhattan, Chebyshev and Hamming distance. The choice of distance metrics leads to different predictive surfaces as shown in Figure 6.

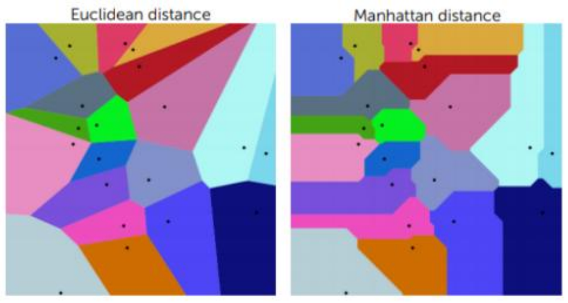


Figure 6: Predictive surface based on distance metrics

The distance metric chosen for the project is the Minkowski distance. It is the generalization of the Euclidean distance and Manhattan distance as shown in Eq (3).

$$d(x,y) = (\sum_{i=1}^{n} |X_i - Y_i|^p)^{\frac{1}{p}} \quad (3)$$

Where d is the similarity metrics, x and y are variables. When p is set to 1 the distance metrics is transformed to Manhattan distance and p=2 is equivalent to the Euclidean distance. The Scaled Euclidean distance is a variant of the Euclidean Distance metrics. The weight on each input is applied to emphasize the importance of the feature. Eq (4) represents the Scaled Euclidean Distance where $a_1, \dots, a_n$ are the weights on each input.

$$d(x,x') = \sqrt{a_1(x_1 - x'_1)^2 + \cdots + a_n(x_n - x'_n)^2} \quad (4)$$

For a given positive integer K (nearest neighbors), observations x and metrics d, the KNN algorithm performs the following:
i. Initializes the Distance **Dis2Knn** parameter by sorting the first k data records in the dataset based on the query bitcoin record.
ii. For the remaining observations, the distance d' difference between the observation and query is computed.
- If d' < **Dis2Knn**, recompute the distance metrics, remove the furthest bitcoins, shift the queue and insert new nearest neighbors.
iii. Return K most similar bitcoins. The prediction is obtained by taking the average over all the estimated outputs.

4.3 Ridge Regression

Ridge Regression is a remedial measure taken to alleviate multicollinearity amongst regression predictor variables in a model. Often predictor variables used in a regression are highly correlated. Ridge Regression performs L2 regularization, which adds a penalty equal to the sum of the square of the magnitude of the coefficients.

As model complexity increases, the bias decreases while the variance increases this leads to model overfitting. Bias is the amount by which the expected model prediction differs from the true value. Variance is the amount by which the target function changes while it is being trained on the data. Alternatively it is the models flexibity to tune itself with the data points in the dataset. When a model is highly specific to the training set it is Overfit. Ridge Regression automatically balances between the Bias and Variance which is required to achieve a good predictive performance. Overfitting is a generic issue with complex models and it can be detected since the estimated parameters magnitude becomes very large.

Ridge Regression attempt to balance between (i) Best predictive function fit through the data and (ii) The model complexity. The quality metrics or the total cost is a combination of the measure of fit and the measure of the coefficient magnitude. Measure of fit is the RSS (Residual Sum of Squares) – sum of the square of difference between the actual and observed data points as shown in Eq 5. Measure of magnitude of coefficients is the L2 norm - sum of squares of the coefficients Eq 6. The Total cost of Ridge Regression is given in Eq 7.

Measure of fit $RSS(w) = \sum_{i=1}^{N}(yi - h(xi) * w)^2$ (5)

Measure of coefficient magnitude

$$||w||_2^2 = w_0^2 + w_1^2 + \dots + w_n^2 \quad (6)$$

$$\text{Total Cost} = RSS(w) + \boldsymbol{\lambda}\, ||w||_2^2 \quad (7)$$

Where $\boldsymbol{\lambda}$ is a tuning parameter chosen to balance the fit and coefficient magnitude. In general when $\boldsymbol{\lambda}$ is small the coefficient magnitude is large and when $\boldsymbol{\lambda}$ is infinite the coefficients tend to zero. Figure 7. Shows the Ridge Coefficients as a function of the regularization.

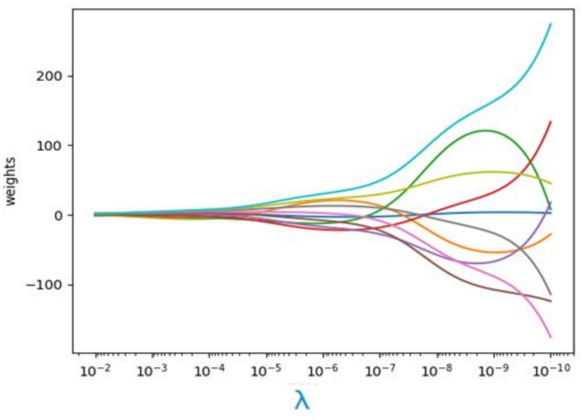


Figure 7: Ridge Coefficient Path

The Ridge Regression algorithm performs the following steps:
i. Initialize the coefficients to zero (w=0) at t=1.
ii. While the Residual sum of squares is not within the threshold limit (ε), compute the estimated coefficient at the next iteration as a function of the total cost of ridge regression on the training set.
ii. Use the validation set to select the tuning parameter $\boldsymbol{\lambda}^*$ such that the estimated coefficients $w_{\boldsymbol{\lambda}^*}$ minimizes the error on the dataset.
iii. Approximate generalization error of $w_{\boldsymbol{\lambda}^*}$, using the test set.

In case of insufficient data to form a separate validation set then performs K-Folds Cross-Validation. The training set is divided into blocks and each block is treated as a validation set during each iteration. The training blocks are used to estimate the coefficients while the error is computed on the validation block. The average error across all validation set is computed.

4.4 LASSO Regression

LASSO is an acronym for Least Absolute Shrinkage and Selection Operator. It is an alternative to the least square estimate to avoid overfitting in the presence of a large number of independent variables [6].
Large coefficients are significant since it emphasize features that could be good predictors of the outcome. Lasso regression performs L1 regularization that adds the penalty equivalent to the absolute value of the magnitude of the coefficients. L1 regularization leads to sparse solutions. The total cost of LASSO Regression is the contribution from the measures of fit and L1 penalty as formulated in Eq 8.

$$\text{Total Cost} = RSS(w) + \boldsymbol{\lambda}\, \left|\left|w\right|\right|_1 \quad (8)$$

Where $RSS(w)$ is the measure of fit of the model, $\left|\left|w\right|\right|_1$ is the L1 penalty and is the sum of the absolute values of the coefficients, $\boldsymbol{\lambda}$ is the tuning factor that controls the strength of the penalty. When $\boldsymbol{\lambda} = 0$: then it is a simple linear regression model. When $\boldsymbol{\lambda} = \infty$: then all coefficients are zero. When $\boldsymbol{\lambda}$ is in between, LASSO Regression attempts fit a linear model on the dataset and shrinks the coefficients.

The LASSO procedure encourages simple and sparse solutions. This causes some coefficients to be shrunk to zero and is able to perform feature selection. As $\boldsymbol{\lambda}$ value increases, more coefficients will be set to zero and they can be discarded and only features with significant magnitude can be taken into consideration. Figure 8. Shows the LASSO Coefficients as a function of the regularization.

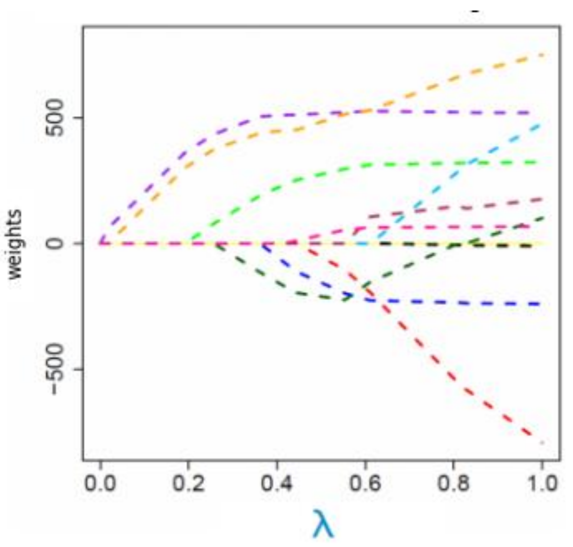


Figure 8. LASSO Coefficient Path

The Ridge regression shrinks all coefficients towards zero; the lasso tends to give a set of zero coefficients and leads to sparse solutions.

4.5 Polynomial Regression

Polynomial regression is a form regression analysis in which the maximum power of the independent variable is more than one. The polynomial regression model for a single predictor X is given in Eq 9.

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_h X^h + \epsilon \qquad (9)$$

Where h is the degree of the polynomial. When h = 2 it is a quadratic model, h = 3 is a cubic model. These models allow non-linear relationship between Y and X, but are still considered linear regression since the regression coefficients are linear $\beta_0, \ldots, \beta_h$. Since the model consists of powers of a single feature it is not possible to hold the other values still while focusing on one coefficient.

The cost of implementing a polynomial regression model is equivalent to the Residual Sum of Squares which is the square of the sum of difference between the original and predicted values. For a Dataset consisting of D features and N observations, then the total complexity in computing RSS is given in Eq 10.

$$Total\ Complexity = O(ND^2 + D^3) \qquad (10)$$

The polynomial Regression model performs the following operations:
i. The dataset is split three-ways into training set, validation set and test set.
ii. Select the degree of the polynomial in a wide range (1, 15).
iii. Train the polynomial regression model, by providing the current polynomial degree, input feature, target feature.
iv. Compute the Residual Sum of Squares on the validation set.
Return the polynomial degree that had the lowest cost (RSS) on the validation set. Consider the chosen degree from the validation set to assess the performance on the test set.
A polynomial Regression model should adhere to the hierarchy principle, which says that if your model includes X^h and X^h is shown to be a statistically significant predictor of Y, then the model should also include each X^j for all j<h, whether or not the coefficients for these lower-order terms are significant.

4.6 Linear Support Vector Machine

Support Vector Machine (SVM) is a supervised Machine learning algorithm. Support Vectors are the data points that lie closest to the decision surface or the hyperplane. This data has a direct bearing on the optimum location of the decision surface. SVMs maximize the margin around the separating hyperplane. The decision function is fully specified by a subset of the training samples. [9]
Input to the SVM is the set of training pair samples, and the Output is a set of weights **w** for each feature whose linear combination predicts the target value **y**. Key point is in SVM optimization of the margin is employed to reduce the number of weights that are non-zero to just a few that correspond to the important features that matter in separating hyperplane. The non-zero weights correspond to support vectors since they "support" the separating hyperplane.
One of the most important ideas in Support Vector Machines is presenting the solution by using a small subset of the training subset to give enormous computational advantage. Using the epsilon intensive loss function the global minimum can be ensured and the optimization of reliable generalization bound can be obtained. Figure 9. Detailed picture of the epsilon band with slack variables and selected data points.

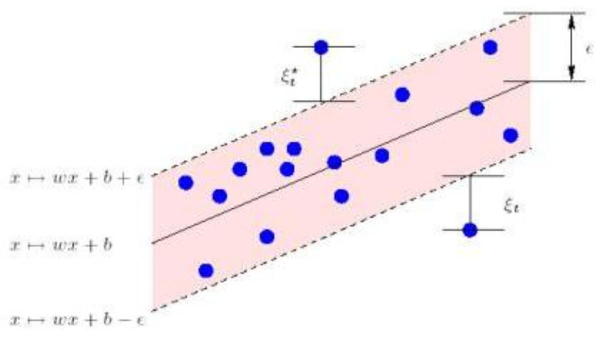

ξ_i^*
ε
$x \mapsto wx + b + \varepsilon$
ξ_i
$x \mapsto wx + b$
$x \mapsto wx + b - \varepsilon$

Figure 9. Epsilon bandwidth and selected data

In SVM regression, the input space **x** is first mapped onto a m-dimensional feature space using a fixed nonlinear mapping; thereafter the linear model is constructed in the feature space. Using the mathematical notation, the linear model is given in Eq 10.

$$f(x, w) = \sum_{j=i}^{m} w_j\, g_j(x) + b \qquad (10)$$

Where $f(x, w)$ is the features space, $g_j(x)$ denotes a set of non-linear transformations, b is the "bias" term.
The Linear SVM algorithm is implemented by setting the Kernel parameter to linear; moreover it is implemented as liblinear that gives the model more flexibility in terms of choice of penalties and loss functions that enhances the models scalability. The parameters chosen for the model are:
i. Kernel – It is set to "linear" to perform linear regression.
ii. C – It is the L2 penalty parameter. As the parameter magnitude increases, the regularization is reduced.
iii. Epsilon – It is a loss function and the value of this parameter depends on the scale of the target value.

4.7 Kernel Support Vector Machine

The idea of a Support Vector Machine for non-linear regression is to build a mapping $x \to \varphi(x)$ from the original m dimensional feature space x to a new feature space x'.In the new space x' the relationship between the new feature vector $\varphi(x)$ and target y is considered linear [8]. By building a proper mapping, the nonlinear relationship can be approximated.

The SVM performs regression by using the ε-insensitive loss and attempts to minimize the model complexity $||w||^2$ - the L2 norm. The SVM regression is formulated to minimize the loss function given in Eq 11.

$$\frac{1}{2}||w||^2 + C\sum_{i=1}^{N}(\varepsilon_i + \varepsilon_i^*) \qquad (11)$$

Where C, ε are the SVM meta-parameters and $||w||^2$ is the L2 regularization parameter. The optimization problem can be addressed by the Kernel function as shown in Eq 12.

$$f(x) = \sum_{i=1}^{n_{sv}}(\alpha_i - \alpha_i^*)\, K(x_i, x) \qquad (12)$$

Where n_{sv} is the number of Support Vectors (SVs) and $K(x_i, x)$ is the Kernel function.

Commonly used kernels include radial basis function kernel $K(x,y) = exp\,(-||x-y||/2\sigma^2)$ and the sigmoid function kernel $K(x,y) = tanh\,(kx \cdot y - \delta)$, where $\sigma,\ k,$ and δ are kernel parameters than can be tuned [9]. The Kernel employed in the Bitcoin estimation project is the Radial Basis Function (RBF) Kernel.

The RBF kernel on samples represented as feature vectors in input space can be recognized as the square of the Euclidean distance between the feature vectors. The value of the RBF Kernel decreases with distance and ranges between 0 and 1. When employing SVM with Radial Basis Function to train a dataset two parameters must be considered.

i. C – determines the tradeoff between the model complexity and deviation tolerance ε . If C is large, then the objective reduces to identifying the deviations larger than ε that can be tolerated in the optimization formulation.

ii. gamma – It defines the extent of influence a training example has on the model. If gamma is large, then the radius of area of influence of the support vectors only includes support vectors and C penalty factor cannot prevent overfitting. The model is prone to overfitting and is in a low bias / high variance state. When gamma is very small, the model is too constrained to capture the complexity and hence fails to identify the data shape. It behaves like linear model with a set of hyperplanes separating the data points.

5. RESULTS

Brief summary of the results generated by applying the above methods to the historical bitcoin dataset is presented in this section. The dataset was split into two portions, 80% of the dataset was used for training while the rest 20% was used for testing.

5.1 Linear Regression Model

The Linear regression model was the first model trained to predict the Bitcoin market price given the input features. The model performed well and produced good accuracy. The Linear regression model has no hyperparameters to tune the algorithms performance. The values of the coefficients and cost of the linear regression model is provided in Table 1.

Table 1. Linear Regression Model Coefficients and Accuracy

Model Information	Values		
Intercept	-0.00440126		
Coefficients	0.008128	0.010289	1.194470
	-0.012411	0.007715	0.028568
	0.012052	0.066543	0.320372
	0.051178	0.006806	0.187696
	0.052609	0.093790	-1.305774
	-0.069781	-0.054105	-0.033170
	-0.026893	0.593628	
Training RSS	97.41%		
Test RSS	97.99%		
Mean Square Error	0.000329838203834		

5.2 K-Nearest Neighbors Model

The K-Nearest Neighbors Model is used in the project. A range of alpha values are employed in the algorithm to test the performance. Furthermore Grid Search Logic is introduced for hyperparameter tuning. Additionally, cross-validation is done to verify the Residual Sum of Squares on the Training and Test data set. The KNN model parameters and accuracy is listed in Table 2.

Table 2. K-NN Model Parameters and Accuracy

Model Information	Values
Alpha range	1,2,3,4,5,6,7,8,9,10
Distance Metrics	Minkowski
Leaf Size	30
Cross Validation Folds	5
Grid Search Best Score	- 1.66906363028
Grid Search Best Estimator	10
Training RSS	98.77%
Test RSS	98.81%
Mean Square Error	0.000213874440035

5.3 Ridge Regression Model

The Ridge Regression algorithm trained the model to penalize for overfitting. It introduced an additional L2 regularization term to trade off between bias and variance. The Grid Search logic was used to perform parameter tuning on the model. Moreover the cross-validation technique was used to validate the r-squares values of the training and test dataset. The Ridge Regression model details and results are given in Table 3.

Table 3. Ridge Regression Model Metrics and Accuracy

Model Information	Values		
Alpha range	0.0001, 0.001, 0.01, 0.1, 0, 1, 10, 100		
Cross Validation Folds	5		
Grid Search Best Score	- 186.016540236		
Grid Search Best Estimator	100.0		
Intercept	-0.0043451		
Coefficients	-1.81e-02	-1.81e-02	4.63e-02
	2.54e-03	-1.4e-02	2.71e-03
	4.36e-04	9.41e-02	9.75e-02
	4.38e-02	8.89e-05	1.09e-01
	3.26e-02	7.67e-03	4.8e-02
	2.97e-03	5.28e-03	-2.24e-03
	-2.47e-03	9.16e-02	
Number of non-zero features	20		
Training RSS	74.6%		
Test RSS	11.5%		
Mean Square Error	0.0037794297444		

5.4 LASSO Regression Model

The Lasso Regression algorithm introduces the L1 regularization term that plays a vital role in feature selection. The Grid Search logic is applied in this model for hyperparameter tuning in order to achieve optimum performance. A maximum of 100000 iterations are computed using the LASSO regression model. The model performed well with high accuracy. The 5-folds cross validation set is used to evaluate the quality metrics of the training and test datasets. The Lasso Regression evaluation metrics and results are tabulated in Table 4.

5.5 Polynomial Regression Model

While a polynomial regression algorithm with degree set to 2 was used to train the model, the model achieved 100% accuracy. This high accuracy rate could be an indication of the model overfitting. Hence a Ridge polynomial algorithm was applied to the model to prevent overfitting and achieve a better predictive estimate of the Bitcoin market price. The polynomial Regression accuracy data is presented in Table 5.

5.6 Linear Support Vector Machine

The meta-parameters chosen for Linear SVM model are C, gamma and epsilon ε. Grid Search Logic is implemented in order to access and retrieve the optimum values of the hyper parameters. Moreover 5-fold cross validation set is used to

Table 4. LASSO Regression Model Metrics and Results

Model Information	Values		
Alpha range	0.0001, 0.001, 0.01, 0.1, 0, 1, 10, 100, 1000		
Cross Validation Folds	5		
Grid Search Best Estimator	100.0		
Grid Search Best Estimator	0.0001		
Intercept	--0.00398762		
Coefficients	-0.003	0	0
	0	0.0016	0
	0	0.0283	0.289
	0	0	0.165
	0	0	-0.038
	0	-0.0009	0
	0	0.6527	
Number of non-zero features	8		
Training RSS	97.3%		
Test RSS	97.7%		
Non-zero weight Features	btc_estimated_transaction_volume_usd	0.653	
	btc_difficulty	0.289	
	btc_cost_per_transaction	0.165	
	btc_n_transactions_total	-0.038	
	btc_hash_rate	0.028	
	btc_total_bitcoins	-0.003	
	btc_n_orphaned_blocks	0.002	
	btc_n_transactions_excluding_chains_longer_than_100	-0.001	
Mean Square Error	0.000356525918013		

Table 5. Polynomial Regression Model Accuracy

Model Information	Values
Polynomial Regression Metrics	
Training RSS	100%
Test RSS	100%
Ridge Polynomial Metrics	
Training RSS	99.7%
Test RSS	99.6%

verify the cost and quality metrics of the model. The obtained hyperparameters are used in the algorithm to train the model. The range of meta-parameters and the obtained results from the Linear SVM model are enlisted in Table 6.

Table 6. Linear SVM Hyperparameters and Accuracy

Model Information	Values
C – input range	0.01, 0.1, 1, 10
gamma – input range	0.001, 0.01, 0.1, 1
epsilon – input range	0.001, 0.01, 0.1, 1
C – best estimate	0.01
gamma – best estimate	0.001
epsilon – best estimate	0.001
Training RSS	91.2%
Test RSS	37.1%
Mean Square Error	0.00126957559829

5.7 Kernel Support Vector Machine

A Radial Basis Function SVM Kernel is used in the project to train the input features to achieve better predictive estimation of the Bitcoin Market Price. The Kernel hyperparameters such as C, gamma and epsilon ε are set to tune the algorithms performance. Similar to the Linear SVM model, the Grid search Logic is implemented on 5-fold Cross Validation set in order to achieve the optimum meta-parameters. The estimated parameters are then used to train the model and predict the results. The Kernel SVM (RBF) metrics and model accuracy is shown in Table 7.

Table 7. Kernel SVM Metaparameters and Accuracy

Model Information	Values
C – input range	0.001,0.01, 0.1, 1, 10
gamma – input range	0.01, 0.1, 1, 10
epsilon – input range	0.01, 0.1, 1, 10
C – best estimate	0.1
gamma – best estimate	0.01
epsilon – best estimate	0.01
Training RSS	95.28%
Test RSS	61.85%
Mean Square Error	0.000726112680019

6. CONCLUSION

Although the dataset has numerous features, all the features did not contribute significantly to the Bitcoin Price forecast objective. From Data Visualization models and correlation plots it was observed that 'Date' column did not add any predictive information on the target variable 'Bitcoin Market Price'. Additionally columns 'btc_market_cap' and 'btc_miners_revenue' were highly correlated with target and therefore these columns were removed from the dataset as it could affect the model accuracy and lead to overfitting. Moreover since the data columns had high variability the min-max scaling technique was applied on the dataset.

According to the model analysis, the K-Nearest Neighbors model provides the best accuracy of 98.77% of the training set and 98.81% on the test set. Additionally the KNN model has the lowest Mean Square Error and does not indicate symptoms of overfitting. Although the polynomial model gave 100% accuracy, it is highly likely that the model is overfit.

7. FUTURE WORK

The Bitcoin digital currency has evolved from a novel financial experiment to a major currency with exchanges all over the globe. The current market capitalization of Bitcoin is more than $125 billion. The blockchain technology is used to maintain a ledger of all bitcoin transaction and actively track and analyze at micro-economic level. Since Bitcoin presence is becoming dominant it is imperative to assess, predict and value this cryptocurrency. Statistical models need to be developed to forecast trading price of the Bitcoins by using data stream from the transaction network and other economical indicators in combination with Machine Learning algorithms like Convoluted Neural Networks, Recurrent Neural Network, and LSTM [10]. Additional investigation is required to identify unseen market conditions that can have a significant influence of the Bitcoin price. Work should be conducted to improve the performance of regression models by inducing additional hyper-parameters and optimizing trading strategy in correlation with the prediction. Furthermore, Wavelet coherence can be used to study the movement to digital currency alongside related factors and explore the relationship between different ciphercurrencies.

8. REFERENCES

[1] "Bitcoin Price Prediction: Will Bitcoin Crash or Rise?", Ray King, May 2018

[2] Bitcoin (BTC) price prediction, Finder, June 2018

[3] "Predicting Bitcoin price fluctuations with Twitter sentiment analysis", Evita Stenqvist, Jacob Lonno, 2017.

[4] Predicting price of Bitcoin using Machine Learning, Sean McNally, September 2016

[5] Comprehensive Guide Regression, Analytics Vidhya, August 2015.

[6] Regression Analysis, MindMajix, October 2017.

[7] Model Selection for Support Vector Machines, O. Chapelle and V. Vapnik, 1999.

[8] "Testing for Nonlinear Dependence in Daily Foreign Exchange Rates*", David A. Hseih, 1989.

[9] "A New Kernel of Support Vector Regression for Forecasting High-frequency Stock Returns", Hui Qu, Yu Zhang, 2016.

[10] "Speed of convergence to market efficiency", D. Y. Chung and K. Hrazdil, 2012