Alexander Bogdanowicz

Final Project Write-UP

Professor Linda Sellie

Intro to Machine Learning

Understanding Bitcoin as an Asset Class

I. Introduction

In recent years, machine learning, thanks to advances in computational power and recent strides made in the fields of deep learning and artificial neural networks has begun to enter and disrupt many facets of business and has found multidisciplinary applications across academia. The purpose of this research paper is inspired in part by these recent developments, and specifically on the historical adoption of machine learning techniques in finance, most recently that of neural networks in predictive analytics. Traditional questions posed in the realm of finance are related to predicting stock prices (regression), assessing creditor standing based on specific features (classification), and even in applying image recognition techniques to help shed light on factors such as traffic, shipping, or oil reserves (Clark and Murtaugh). However, the field of ML has also been applied to simpler problems of classifying assets - such as the S&P 500's Technology Sector - applying statistical and ML models to help shed light on potential asset qualities that transcend simple industry classification.

Among the many assets traded world wide and around the clock, cryptocurrencies have become the subject of much political as well as economic debate in both the behavior of the asset on various cryptocurrency markets and exchanges, as well as its underlying use case. While argument have been made as to their utility in easing global financial transactions or in crypto-currencies ledger based system of accountability, it is difficult not to see the asset as anything more than a speculative device given its extreme volatility and little underlying tangible value. This paper will attempt to utilize machine learning techniques to answer the question: What asset class do cryptocurrencies, and specifically Bitcoin, most ressemble?

The dataset compiled for this paper is an attempt at gathering a representative sample of different asset classes, mainly focusing on shares of companies, currencies, commodities, and debt instruments. Table A. describes the datasets in more detail. In an attempt to discover some underlying relationship between currently publicly traded assets and Bitcoin, a variety of supervised learning practices can be applied. I will focus mainly on three types of models, ARIMA (Auto-Regressive Integrated Moving Average) and GARCH (Generalized, Auto-Regressive Conditional Heteroskedasticity), Support Vector Machines, and Artificial Neural Networks, in attempting to discover whether, in the case of SVMs, models trained on specific assets can provide any predictive capacity for the direction of Bitcoin prices, or in the case of ANNs, whether a model trained on a group of assets will succeed in classifying Bitcoin into any asset bucket. In these ways, we may hope to learn more about the underlying qualities of Bitcoin.

Asset Class	Data	Features	Source	Date-Range
Stocks/Shares in Publicly Traded Companies	S&P 500 (SNP)	Price, Trade Volume	Yahoo Finance	2004-11-19 to 2019-04-18
Commodities (Gold)	SPDR Gold Shares (GLD)	Price, Trade Volume	Yahoo Finance	
Currencies (FOREX)	EUR/USD & USD/JPY	Price, Trade Volume	Dukascopy Currency Database	
Corporate Debt	iShares iBoxx Investment Grade Corporate Bond ETF (LQD)	Price, Trade Volume	Yahoo Finance	
Risk Free bonds	iShares 7-10 Year Treasury	Price, Trade Volume	Yahoo Finance	
Cryptocurrencies (BTC)	Bitcoin USD (BTC-USD)	Price, Trade Volume	Yahoo Finance	2010-07-16 to 2019-04-18

Table A. Description of Data, Sources, and Features

II. Supervised Analysis

A. ARIMA/GARCH Modeling

We begin our supervised analysis by discussing the use of ARIMA/GARCH modeling to help gain initial insights into the dataset and in order to generate new features to be used in more complicated models. The power in these models comes from their ability to process time-series data that is neither mean nor variance stationary by inducing such stationarity through innovative autoregressive and moving average techniques, to which notable contributions have been made by NYU Professor and nobel laureate Robert F. Engle, who first described the GARCH framework in his seminal 1982 paper "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation" in 1982 (Engle 1). Specifically, these models rely on the fact that time series finance data is often random-walk, meaning that a stock's position at time t, is a result of its position at time t-1. A portion of this issues is tackled in simply "normalizing" the data in calculating the logged returns, as done initially, in the data preprocessing stage. However, while the data may seem mean and variance stationary, that is, the data is time-irrelevant with constant mean and standard deviation, a look at the auto- and partial-autocorrelations of squared returns is revealing of another story, as seen in Figure 1. (Duke University, "Introduction to Arima Models"). We therefore generate Partial and Auto-Correlative plots in order to better understand the behavior of each data, and its specific ARIMA needs, as inspired and influenced by the steps taken by Shub Jain, in his set of lectures on Time Series analysis located on Medium (Jain, Shub, Medium).

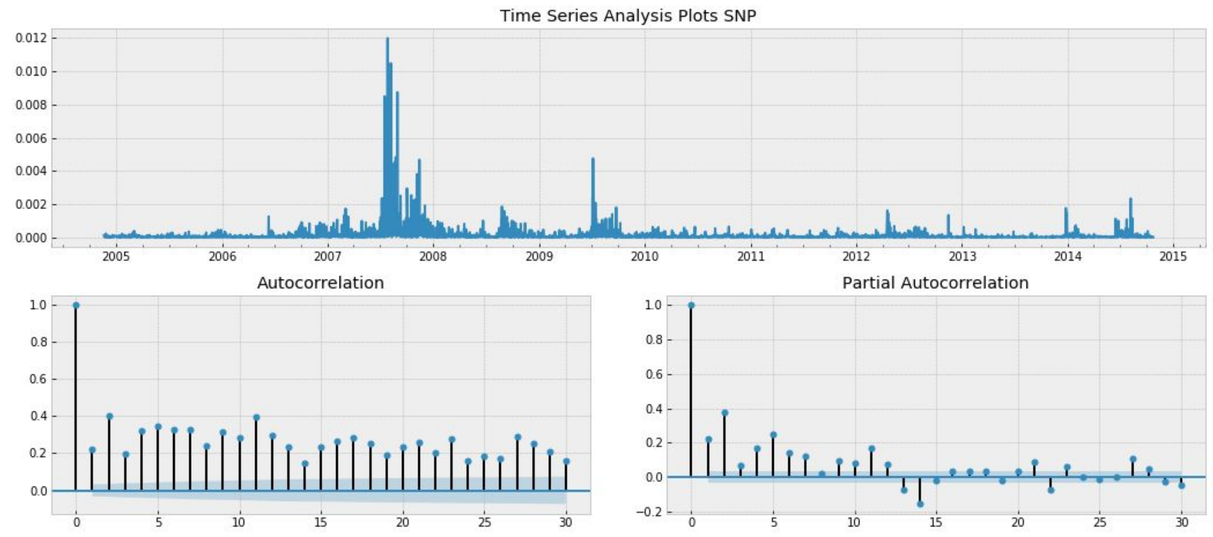


Figure 1. Time Series Partial and Auto-Correlation (Plt Design: Jain, Shub Medium Time Series Analysis)

Although attempts were made to try and generate the optimal ARIMA model parameters (i.e. # of autoregressive terms, # of differences needed for stationarity, # of lagged forecast errors) by minimizing a certain cost function, the Akaike Information Criterion, a log-likelihood measure of model fit, the time consuming nature of generating each model for different iterations of i, d, and j, and the fact that the specific time series in question were rather complex meant that a more feasible method for assessing optimal ARIMA parameters was by simply observing the Partial and Autocorrelation functions and choosing ARIMA inputs accordingly.

It is important to note that ARIMA models left to be generated automatically will often choose lags for very large p and q, leading to overfitting. The solution to this problem was in utilizing an AIC cost function to identify the best models but our method of looking at the correlation function was also effective. Both of these can be seen as forms of regularization of model parameters in an effort to prevent overfitting.

The final step was to generate ARIMA predictions for the time series, and collect the residuals. While I attempted to generate GARCH models for each asset class, the particularity of each models ARIMA inputs - as GARCH must be run on ARIMA Residuals (effectively error terms) - made GARCH outputs for certain Asset Classes unreliable and unable to classify the autocorrelation imbedded in the time series. Thus estimating variance was not a convenient option with the GARCH framework. For our Machine Learning purposes however, the residuals would provide a good amount of information regarding the variance of the model at time t, relative to its price movement.

B. Support Vector Machines as Directional Classifiers

In an effort to further our goal of understanding Bitcoin as an asset class relative to current popularly traded assets, I decided to pursue a binary classifier, specifically a Support Vector Machine Classifier that would be trained to predict the direction of returns one-month in advance (20-days), with a unique SVM trained for each asset class, excluding bitcoin. The model parameters were inspired by

Saahil Madge's 2015 work "Predicting Stock Price Direction using Support Vector Machines" where he trained an SVM classifier on multiple time frames using Stock Price Volatility and Momentum (Madge, Saahil, Independent Work Report). In a similar fashion, I built the asset class SVM classifier to take advantage of these two data transformations, Volatility and Momentum, but also added the residuals from the previous ARIMA models smoothed over a monthly (20-day) period. Thus the SVMs were built on three main features, representing the average change in prices over 20-days, the average direction of price changes (momentum), and the deviation from the expected normal stock price (residuals). However, in addition to assessing which SVM would classify the direction of bitcoin returns most accurately, I was also curious whether the addition of the ARIMA residuals provided the SVM classifier with more information and variance in order to create a better separating hyperplane in higher-dimensional space. Therefore, the first generation of SVM models was built only on Price Volatility and Momentum.

It is also important to identify the advantages of our SVM classifier and what extra model parameters might be necessary, such as regularization parameters or model gamma. As a result of the model specification of maximizing the distance between model support vectors and the separating hyperplane, without regularization parameters, SVMs are prone to overfitting, as the model prioritizes minimizing classification error, which can cause tight margins. In training our models, we would like them to prioritize accuracy of the testing. Therefore, in training the classifier, I iterate over sets of regularization values *c*, and gamma parameters *g*, and perform 10-fold cross validation in order to find the model that performs best on the randomly initialized testing sets for specific values of C and G. An additional advantage to using SVMs is their unique ability to take advantage of kernels, which, help transform the input data to make it more linearly separable (i.e. via adding polynomial terms etc.) and ease the process of creating a separating hyperplane. Per advanced finance literature, the majority of SVMs take advantage of the Gaussian Radial Basis function, as a result of its unique geometric properties (handles diverse sets well) as explained in Rosillo, et al (Rosillo, et al, Journal of Forecasting). Therefore, I will be using the gRBF in training the SVMs. These advantages are illustrated in the plot in Figure 2. Outputs from the first generation model can be seen in Table B.

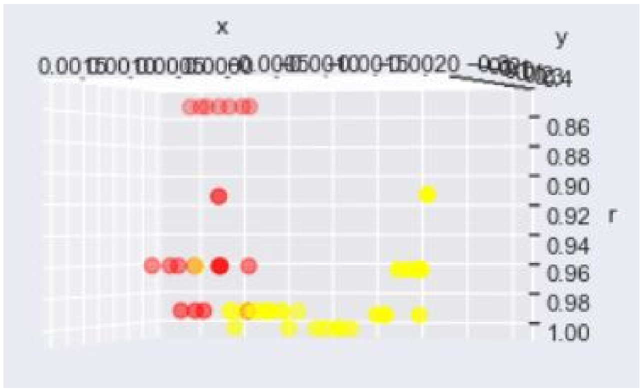


Figure 2. Plotting Volatility, Momentum and the RBF Transformation into R^3 space

	S&P 500	Gold	EUR/USD	USD/JPN	Corp_Bonds	Treas_Bonds
C_Param	10	0.001	1	0.0001	10	0.001
Gamma	10	0.01	100	100	1	0.01
Train_Accuracy	0.641176	0.530195	0.532053	0.504181	0.530341	0.527408
Test_Accuracy	0.701117	0.604457	0.579387	0.579387	0.614525	0.612813

Table B. SVM Top-Model Results from training on Volatility and Momentum

While the results do not seem impressive at first glance, it is important to put them in perspective. All classifiers were able to achieve better than guessing (>50%) accuracy in predicting stock price direction, one-month out. In finance, every percentage point counts, and this seems to be counterintuitive to the formulations of the Efficient Market Hypothesis which states that prices are always a function of new information. We see that, the SVM performed the best on the general stock market index, the S&P 500, and this is likely due to the fact of the immense role of momentum that factors into the daily/monthly price variations of the index. In any case, I proceeded to utilize the same methodology in adding the ARIMA residuals and training this time, on the three aforementioned features. The results are summarized in Table C.

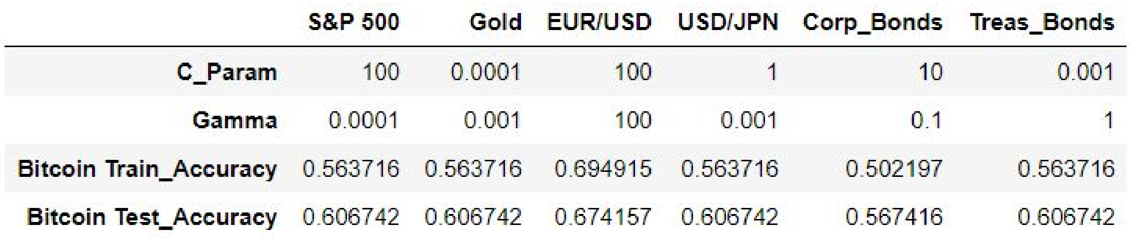
	S&P 500	Gold	EUR/USD	USD/JPN	Corp_Bonds	Treas_Bonds
C_Param	100	0.0001	100	1	10	0.001
Gamma	0.0001	0.001	100	0.001	0.1	1
Train_Accuracy	0.638278	0.529412	0.560855	0.504644	0.531124	0.527554
Test_Accuracy	0.713092	0.611732	0.598886	0.575419	0.612813	0.611732

Table C. SVM Top-Model Results from training on Volatility, Momentum, and Arima Residuals

We observe similar results to the original model trained on just two features, except that, at least for the first 3 asset classes, we seem to achieve a better Testing Accuracy, and in Finance, even a percentage point increase in accuracy can be worth the inclusion of another feature and increases in complexity. What we are most interested in however is how these two sets of asset class directional classifiers will fair when the input data is bitcoin as this will help shed light on which asset class bitcoin may share some hidden attributes with. Tables D. and E. outline the results of running each optimized SVM classifier on bitcoin volatility and momentum in the first set, and the additional inclusion of ARIMA residuals in the second set.

	S&P 500	Gold	EUR/USD	USD/JPN	Corp_Bonds	Treas_Bonds
C_Param	10	0.001	1	0.0001	10	0.001
Gamma	10	0.01	100	100	1	0.01
Bitcoin Train_Accuracy	0.567483	0.565599	0.37602	0.565599	0.474576	0.565599
Bitcoin Test_Accuracy	0.595506	0.589888	0.331461	0.589888	0.47191	0.589888

Table D. SVM Top-Model Results Testing on Bitcoin Volatility and Momentum



	S&P 500	Gold	EUR/USD	USD/JPN	Corp_Bonds	Treas_Bonds
C_Param	100	0.0001	100	1	10	0.001
Gamma	0.0001	0.001	100	0.001	0.1	1
Bitcoin Train_Accuracy	0.563716	0.563716	0.694915	0.563716	0.502197	0.563716
Bitcoin Test_Accuracy	0.606742	0.606742	0.674157	0.606742	0.567416	0.606742

Table E. SVM Top-Model Results Testing on Bitcoin Volatility, Momentum, and Arima Residuals

These results are revealing of more than just which asset class is most similar to that of bitcoin, and explains more on the efficacy of ARIMA residuals as an explanatory feature, as well as the likely inter-relation between asset classes on the basis of Momentum and Volatility. For example, we see that in reality, most of the models, 2/3rds in fact, do a decent job of predicting the direction of bitcoin returns one-month out. This is likely due to the fact that momentum and volatility behave similarly across asset classes at least in the predictive capacity (the SVMs shouldn't differ too much if we were to test each asset class on each other). However, what is more interesting is the role of ARIMA residuals, and how much this increases the predictive capacity of the EURO/USD currency value in its predictive capacity for bitcoin. Thus, all things considered, it seems that Bitcoin does not share similar volatility/momentum properties on an isolated basis to the currency pair, but instead shares a great relationship between those two factors, and the models residuals. This leads me to believe that Bitcoin may in fact house characteristics of a few different asset classes, or, as is more likely, is in fact in a category of its own. These findings warrant more investigation.

C. Neural Network for Classifying Assets

While Support Vector Machines offered interesting insights into the inter and intra-asset relationships of momentum, volatility, and ARIMA residuals, insights into what asset class best describes Bitcoin were marginal in that relationships needed to be extrapolated from the data and accuracy of each SVM. It is for this particular drawback that Neural Networks can compensate as I am relatively unconstrained when it comes to the model's feature space and the output space can be designed to categorize by asset.

Insights from the ARIMA/GARCH analysis and the further use of ARIMA residuals revealed certain difficulties in estimating a propert model for Corporate Bonds and the USD/JPY foreign exchange rate. Specifically, a proper GARCH model could not be fitted - at least per the aforementioned training constraints - for those assets. In addition, relative to classification, these asset classes might prove redundant given their correlations with Treasury Bonds (simple risk premium) and currency respectively. Therefore, in the interest of better classification, we remove these two classes from further consideration.

In terms of the ANNs feature space, as we are able to provide a much higher-dimensional vector as an input layer, we refrain from heavy preprocessing of the data and choose to rely on 20-days worth of Logged Returns, Normalized Trade Volumes, and residual ARIMA values. Thus the input layer will consist of a length 60 vector consisting of the aforementioned data points in consecutive order. The dependent output layer will be of length 4, and will consist of 4 possible asset classes: Stocks, Currency, Commodity, and Debt Instrument.

We run the neural network on 3 different activation functions, and provide the results for each in Table F. We can see that, as is popularly contended, the ReLU function performed the best - and most efficiently - on both the training and testing dataset with an impressive improvement over the standard Sigmoid function. While all three activation functions take advantage of properties of non-linearity, the ReLU activation function provides addition benefits that are not present in the other two, specifically, from an efficiency perspective, there is no need to perform expensive operations in calculating the next input layer (simply output max of 0 and x). Because the ReLU function outputs an absolute number, we need to transform the output space of the final layer in order to make it interpretable by our cost function, and therefore utilize the softmax output function which outputs a vector of probabilities, and therefore just as desirable (from the perspective of interpretability) as the original sigmoid output.

	S&P 500	EUR/USD	GOLD	Treas_Bond	Test Accuracy	Train Accuracy
Sigmoid	1289	1167	21	409	0.347193	0.350806
TanH	584	1520	200	582	0.424809	0.436839
ReLU	577	779	745	785	0.615385	0.664703

Table F. Raw Classification Outcomes for Each Model and Test/Train Accuracy

In regards to the complexity of the neural network, as there seems to be no specific theory on the proper set of hidden layers into the function, and as there were clear computational limits, I decided to stick to an input layer of 60 and a proportional hidden layer of the form (⅔ * input + output). From the perspective of model accuracy, I was unable to assess the benefits of more layers, however from a theoretical perspective, one seems to be sufficient for capturing the non-linearities of the dataset. From a regularization perspective, as implementing a regularization parameter for the simple neural network is rather convenient, I decided to set the Sigmoid, TanH, and ReLU functions with the following respective regularization parameters: 0.0001, 0.005, and 0.001. After a few iterations of the TanH activation function based neural network, it became apparent that the function was prone to getting stuck, and increasing the regularization parameter to the aforementioned value helped remediate this issue, as well as increasing the classification accuracy a notable amount.

It is important to highlight how well the ReLU model performed in Test Accuracy and some potential reasons for its performance. Table G. depicts the confusion matrix for ReLU classification, and sheds some light on where the model's classification strengths and weaknesses are. It is clear straight away that the model is particularly good at distinguishing the currency asset class from the other 3, at a rate of over 80% accurate. We should note that this does not come as much of a surprise as a similar phenomenon occurred while training the second generation of support vector machines with the ARIMA residual data. It may be that the value added by this feature helps to distinguish it from the other asset classes. We can also attempt to analyze the outcome from the perspective of the assets themselves. As currencies are inherently more volatile than any other asset classified by our Neural Net, it is also likely that, in addition to the variance that may be extracted from the residuals, the 20-day logged returns and volume also tell a similar story.

	S&P 500	EUR/USD	GOLD	Treas_Bond
S&P 500	0.566724	0.034660	0.284564	0.215287
EUR/USD	0.050260	0.802311	0.010738	0.039490
GOLD	0.246101	0.002567	0.562416	0.229299
Treas_Bond	0.136915	0.160462	0.142282	0.515924

Table G. ReLU Neural Network Confusion Matrix

Finally, as was the goal from the very beginning, we apply the most accurate classifier (in this case it must be ReLU as it not only beats the other models on EUR/USD, but achieves better individual asset accuracy than the other models do as a whole), to attempt to classify Bitcoin and arrive at the results featured in Table H. Of the almost 1800 sampled bitcoin feature sets of 20-day logged returns, volume traded, and ARIMA residuals, the majority were classified, in similar fashion to the results by the SVM analysis, as a currency, with another significant portion resembling treasury bonds.

	S&P 500	EUR/USD	GOLD	Treas_Bond
ReLU Classification	0.13251	0.601348	0.01909	0.247052

Table H. ReLU Neural Network Classification of Bitcoin Feature Set Results

III. Unsupervised Analysis in Finance

Although in our case, given the fact that we have been working with inherently classified data,, in many ways our goal has been to classify Bitcoin as an asset class based on its relation to other assets. In this way, the problem is one of unsupervised analysis, as based on the qualities of processed data, we would like to infer how to best classify Bitcoin. Thus our use of support vector machines, in generating a real space divided by a separating hyperplane, geometrically is akin, at least in logic, with the use of a k-means classification algorithm, which could theoretically map our feature set into real space, and analyze the relationships between different asset classes based on euclidean or cosine distance in this way. If we were to classify only companies, a k-means classifier would undoubtedly be used as a first-step screening tool to discover any, non-evident relationships between different firms across a specific sector, for example. However, due to limitations of our data set, specifically in that the diversity of the asset classes prevents determining cross-asset features, outside of trade-information (e.g. Market Cap, Price/Earnings, Debt/Equity, etc.), k-means analysis would, and has upon testing, result in no specific relationship among the different data points of each asset class.

Finance is filled with application for unsupervised analysis. For example, Principal Component Analysis, which relies fundamentally on projecting data onto the axis of greatest variance, has been applied to bond yields (e.g. yield curve), in order to discover what are called “Factor’s”, divided into level, slope, and curvature, and can be used to reproduce yield curves to a high-degree of accuracy (Redfern, David Moody’s Analytics). The power from PCA comes from its ability to reduce the dimensionality of data but at the same time retaining the variance, or what can be thought of as information. The results can then be applied to auto-regressive models utilizing less features but operating on a similar amount of information to help predict yield shifts and help financial institutions better adapt their asset portfolios.

IV. Conclusion

The results of our analysis into the characteristics of Bitcoin as an asset class may not be groundbreaking, but they certainly do pose new questions to be answered, and perhaps even offer insights into how Bitcoin across asset classes, and how different assets relate to each other. The fact that the “crypto-currency” is most similar to the EUR/USD currency in both our analysis of Support Vector Machines and Neural Networks may be the result of a variety of reasons, from the fact that the cryptocurrency may be developing currency-like use cases - though it certainly still lacks in liquidity - or perhaps the two assets, bitcoin and our currency asset class are simply both more volatile than the other classes under study, and relate to each other on that dimension alone. While it is difficult to assess, it is easy to see the power and usability of machine learning techniques from simple regression analysis and ultimately culminating in a fairly sophisticated Neural Network, in assessing non-linear relationships that cannot necessarily be extracted by any other means, besides perhaps the gift of intuition.

References

Duke University. “Introduction to ARIMA Models.” *Introduction to ARIMA Models*, people.duke.edu/~rnau/411arim.htm.

Jain, Shub. “Time Series Analysis for Financial Data VI- GARCH Model and Predicting SPX Returns.” *Medium*, Auquan, 13 Dec. 2017, medium.com/auquan/time-series-analysis-for-finance-arch-garch-models-822f87f1d755.

Madge, Saahil, and Swati Bhatt. “Predicting Stock Price Direction Using Support Vector Machines.” *Independent Work Report (Princeton)*, 2015, www.cs.princeton.edu/sites/default/files/uploads/saahil_madge.pdf.

D. d. l. F. Rafael Rosillo, Javier Giner, “Stock market simulation using support vector machines,” Journal of Forecasting, vol. 33, pp. 488–500, July 2014

Redfern, David, and Douglas McLean. “Principal Component Analysis for Yield Curve Modelling.” *Enterprise Risk Solutions*, 2014, www.moodysanalytics.com/-/media/whitepaper/2014/2014-29-08-pca-for-yield-curve-modelling.pdf.

Clark, Aaron, and Dan Murtaugh. “Satellites Are Reshaping How Traders Track Earthly Commodities.” *Bloomberg.com*, Bloomberg, 16 Dec. 2017, www.bloomberg.com/news/articles/2017-12-16/satellites-are-reshaping-how-traders-track-earthly-commodities.