Project

Passing streaming data of stocks and predicting changes in stock data using historic data and its affecting stocks of other companies.

Introduction

Streaming the data into a limited size windows to process data to work parallel over huge data.

Then on processing data we analyze it to divide the stocks into different clusters based on the pattern of data processed in that particular window.

Now that we have the data which is already analyzed and stored into various clusters we now can analyze changes in new stock and of those whom so ever are getting affected.

And if a particular stock has a high variance will get into a new cluster and the behavior would be interdependent on the cluster in which it is present.

All of this is shown through a visualization tool which would animate the patterns of stocks data.

Stocks Data

We basically don't actually use the stocks data rather generate some random data for stocks so we generate data as a limited window.

Need to generate random data points over a window of 60 seconds and that data is parallelized in to RDDs and calculated its variance over that particular window of data points then calculated variance is normalized.

we use spark- streaming where RDDs forms a Stream (discretized stream)

An input DStream is a special DStream that connects Spark Streaming to external data sources for reading input data. For each external data source (e.g. Kafka) you need one such input DStream implementation. Once Spark Streaming is "connected" to an external data source via such input DStreams, any subsequent DStream transformations will create "normal" DStreams.

Generally for generation

np.random.randint(60)

rdd = sc.parallelize([0] * 10).map(f)

then it is set to f for values zeros,but we can generate it at once by adding the partition lines to the code `u = RandomRDDs.uniformRDD(sc, 300, 5)` then pass on to Dstreams and processed as mentioned above.

This process is done parallel for all the buffers.

Now working with the window operations where all the random data produced is supposed to be computed as a 60 second window so that there are 60 pairs of data values produced and there are 5 different windows as such, so we need pairs of data as (key, value).

So the components needed are

window(windowLength, slideInterval) which returns a new DStream which is computed on windows of DStream. Where

windowLength - time duration of window where its 60 secs

slidingInterval - The interval at which window operation is performed.

So the source RDDs that come under a window are combined and operated upon to produce RDDs of the windowed DStream.

K Means Clustering

After getting the processed data from the DStreams we apply the K Means clustering algorithm using some random centroids and finding the difference between each one and this is iterated for several times for the appropriate cluster formation. General Pseudo code is provided.

Let $X = \{x_1, x_2, x_3, \ldots\ldots, x_n\}$ be the set of data points and $V = \{v_1, v_2, \ldots\ldots, v_c\}$ be the set of centers.

Randomly select '*c*' cluster centers.

Calculate the distance between each data point and cluster centers.

Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.

Recalculate the new cluster center using:

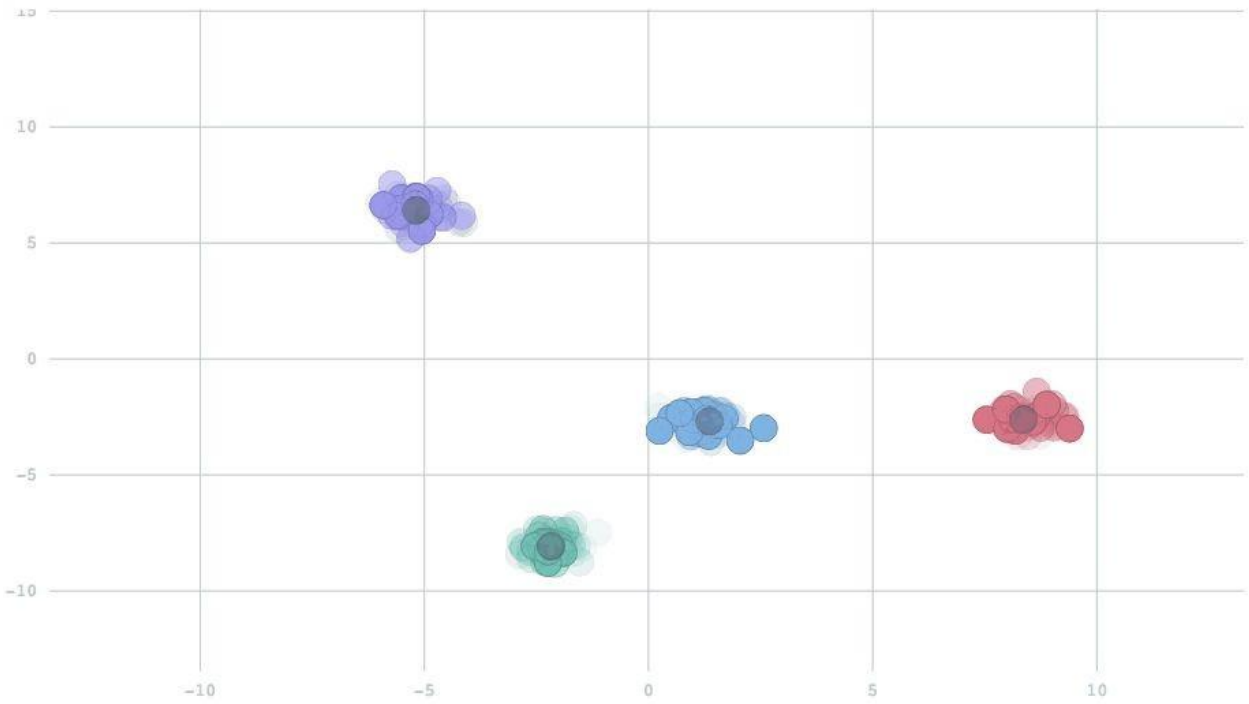
$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_i$$

Where, 'c_i' represents the number of data points in i^{th} Cluster.

Recalculate the distance between each data point and new obtained cluster centers.

Visualization

After the formation of clusters we need to show the data points using a visualization tool as in i found many tools doing so



As from the above gif we can see that any stocks changing in its variance points jumps on to other clusters, hence by making it easier to predict the other affecting stocks.

Overall Methodology

We use k-means clustering algorithm for working on clusters on the analyzing data as mentioned above. It is vector quantization which partition n observations into k clusters.

Clusters are formed based on the normalized result of variance of overall stocks for a particular window, So now when one window is processed and have stored its clusters based on normalized result we could put our new batch into some clusters which have already formed i.e. let's say we have a cluster which has stocks with just say like within 3-5 units' then new data which has same values get into that cluster.

So now we could be able to predict other stocks in same cluster as in if we have Microsoft and Google in same cluster and we find that Microsoft stocks have raised so we can tell that even Google stocks also would be raised.

We show all of those changes in the stocks using some animation which would make it easier to understand. Visualization tools which are available for spark could come in handy for the animation work for the changes in data.