Rossmann Store Sales Prediction

Group #8
Group Name: Miners
Group Members: Jeet Takwani and Hriya Maharaja

Abstract:

This project aims to predict sales for a pharmaceutical store in order to help them to be more effective in terms of their staff schedules, promotional activities and managing inventory.

Introduction of the background:

Rossmann operates over 3,000 drug stores in 7 European countries.
Reliable sales forecasts enable store managers to create effective staff schedules that increase productivity. Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality.

Problem Definition:

Through their Kaggle competition, Rossmann stores are asking to predict their daily sales for up to six weeks in advance for 1,115 of their stores located across Germany.

Data Description:

Rossmann stores have provided their historic sales data for **1,115 stores** across Germany.

The sales data is provided for each of the store starting from 1st January 2013 up to 31st July 2015 for a total of **1017209 records**.

Each record contains:

- **Store -** a unique Id for each store
- **DayOfWeek** – the day of the week on that date
- **Date** – the date for which the sales is recorded
- **Promo** – whether the store was running a promo on that day
- **Sales -** the turnover for any given day (this is what you are predicting)
- **Customers -** the number of customers on a given day
- **Open -** an indicator for whether the store was open: 0 = closed, 1 = open
- **StateHoliday -** indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
- **SchoolHoliday -** indicates if the (Store, Date) was affected by the closure of public school

Apart from this, various other attributes about the stores are also provided :

- **StoreType -** differentiates between 4 different store models: a, b, c, d
- **Assortment -** describes an assortment level: a = basic, b = extra, c = extended
- **CompetitionDistance -** distance in meters to the nearest competitor store
- **CompetitionOpenSince[Month/Year] -** gives the approximate year and month of the time the nearest competitor was opened
- **Promo -** indicates whether a store is running a promo on that day
- **Promo2 -** Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
- **Promo2Since[Year/Week] -** describes the year and calendar week when the store started participating in Promo2
- **PromoInterval -** describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug,Nov" means each round starts in February, May, August, November of any given year for that store

Given this dataset, we are supposed to predict the value of sales for each of the 1,115 stores for a 6 week duration from 1st August 2015 to 17th September 2015.

Preprocessing:

Observations for the given dataset :

- **Days when stores are closed:**

When a store is closed there will be no sale on that day. And so, that record will have no impact on predicting future sales. We therefore proceeded to remove all records where the store was closed. This leaves only the records for days on which the stores were open.

- **Attribute indicating whether store is open:**

The column indicating whether the store was open on that particular day is now redundant and no longer needed. We therefore removed that column while creating the prediction model.

- **State Holidays:**

We also noticed that the stores are always closed on state holidays, and since we removed entries for days when the stores were closed, keeping track of state holidays is no longer required. We thus removed that attribute from consideration.

- **Number of customers visiting the store at a future date:**

For predicting the sales on any future date, we will not have information regarding the number of customers that will visit the store on that day. The test data set thus does not provide the number of customers that will visit the store on any future date. If we include that attribute in the training set while building the model, it will create an inconsistency in prediction since the test dataset will not have that information. The number of customers visiting the store is not included while constructing the model.

After this we have the following attributes , Store ID, DayOfWeek, Date, Promo, School Holiday upon which to construct our model. And additional store information including store type, assortment, distance to competition and so on as noted above. We now proceed to the first attempt on building a prediction model.

Methods description

Since this is a prediction problem for a matrix dataset, our initial approach is using Linear Regression to predict the values. Since this is Linear Regression we removed the date of sale indicated in the record.

Steps:

1.) Preprocess the data as specified above
2.) Get the training dataset from the preprocessed data
3.) Find an optimal beta for each store
4.) Create a test dataset separate from the training dataset
5.) Measure accuracy of calculated model
6.) Run the model on the test dataset provided by Kaggle

Experiments design and Evaluation

Evaluation Metrics:

The evaluation metric used by Rossmann Stores is :

Root Mean Square Percentage Error (RMSPE)

$$\mathrm{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}$$

where y^i denotes the sales of a single store on a single day and y hat_i denotes the corresponding prediction. Any day and store with 0 sales is ignored in scoring.

Experiment 1:

We ran the training dataset to build a linear regression model using the stochastic gradient descent approach.

Observation:

Beta exploded and we could not find an optimal beta

Evaluation:

For further optimization, we can study line search approach to calculate an optimal beta for each iteration, to keep the beta from exploding.

Experiment 2:

We normalized the training dataset and ran it again to find beta using stochastic gradient descent.

Observation:

Beta no longer explodes but beta fails to converge to a local optimum. Terminated the process after 10,000 iterations and noted the value of beta.

Evaluation:

As before we can probably look into implementing line search method for finding optimal beta. We ran the test set with the noted beta and submitted results to kaggle. The evaluation score was 0.25222

Experiment 3:

Ran the dataset to find beta using closed form approach.

Observation:

Beta does not explode. Noted the model for each store separately.

Evaluation:

Ran the test set provided by kaggle and submitted the result to kaggle. The evaluation score was 0.16793. Huge improvement on the previous attempt.

Experiment 4:

In order to gather as much information about a store, we proceeded with the assumption that stores that are of the same type and have the same assortment level, may be correlated. And hence we can use the records of the stores having the same type and assortment, collectively to predict their sales.

Observation:

Upon grouping stores having same type and assortment, we observed that are only 9 groups. Leaving a huge number of stores in each group.

Assort | Type	a	b	c	d
a	381	7	77	128

b	0	9	0	0
c	221	1	71	220

Thus the model for stores belonging to the same group will be similar. This leads to under fitting of the data. Combined with the fact that most attributes are binary, and day of the week has only 7 possible unique values, this leads to similar prediction values for stores belonging to the same group for most test records. The score was 0.39360, which is a lot worse than the previous experiment.

Evaluation:

Generalizing the model for multiple stores does not lead to a good prediction model. Keeping a separate model for each store is a better approach.

Experiment 5:

Ran the normalized dataset to find beta using closed form approach but took into account the month as well, along with the day of week.

Observation:

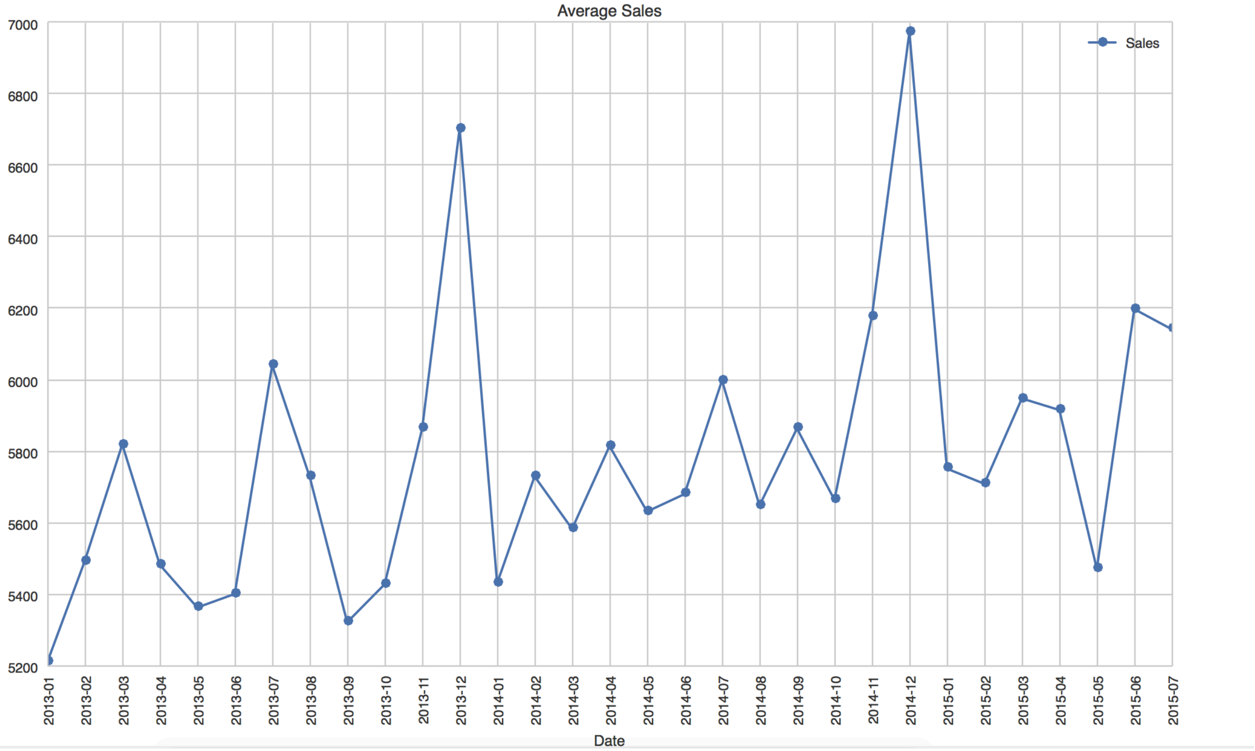
Beta does not explode. Noted the model for each store separately and re ran linear regression on the test data set with the month also taken into consideration.

Evaluation:

Ran the test set provided by kaggle and submitted the result to kaggle.
The score was 0.17277, not an improvement .

We now change our approach, and analyze the dataset using time series models.

The average sales on each day for the given period is:



From the graph, we can observe that there is not enough data to identify a trend, but there are clear seasonal cycles that are peaking in December.

Our first approach is to try the Auto Regressive Model for time series data.

Experiment 6:

Ran the dataset through a AutoRegressive model of order 3.

Evaluation:

The evaluation score is 0.30351,which is not an improvement over the best score. That is probably due to error propagation since we take daily sales for the AR model.

Experiment 7:

Ran the dataset through a AutoRegressive model of order 7.

Observation:

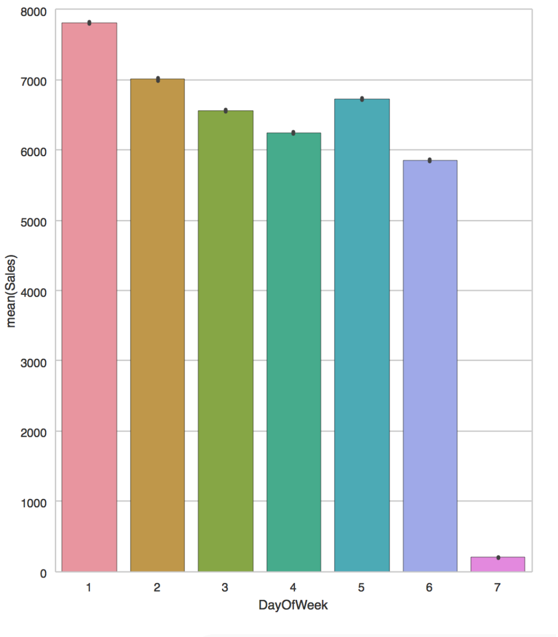
Since we increase the lag values considered, the error propagation is more.
The more lag values we consider, the more error will propagate.

Evaluation:

Ran the test set provided by kaggle and submitted the result to kaggle, the score was 0.34621, which is worse than the score of AR(3).

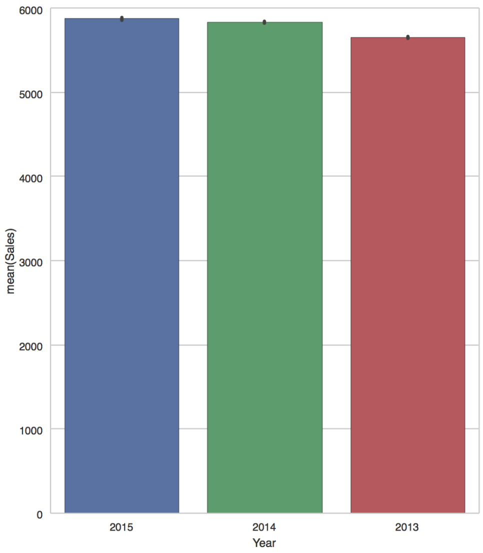
We analyzed the data further to gain insights:

This is the average distribution over a week:

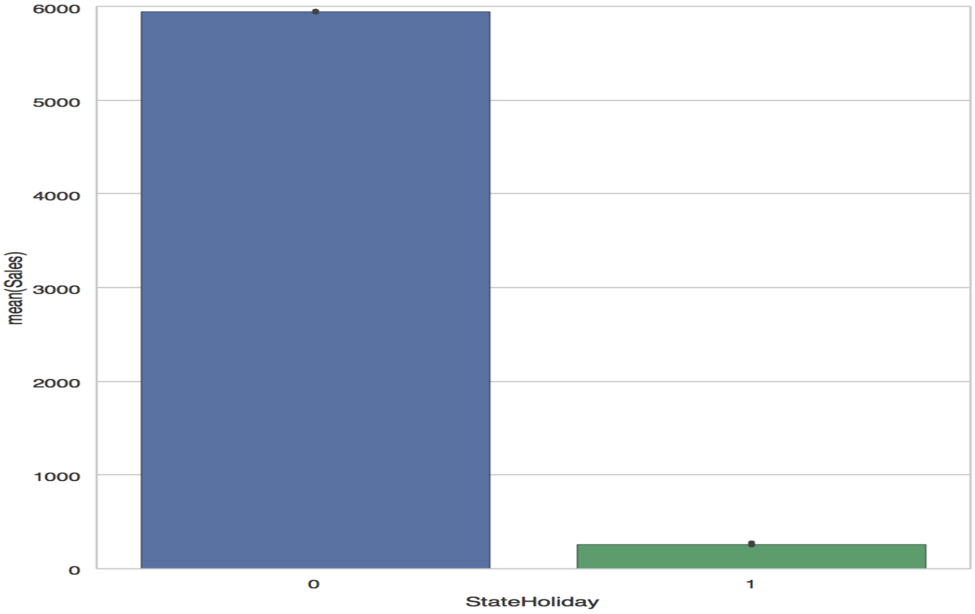


Sales on Sunday are very less since they are mostly closed on Sundays.

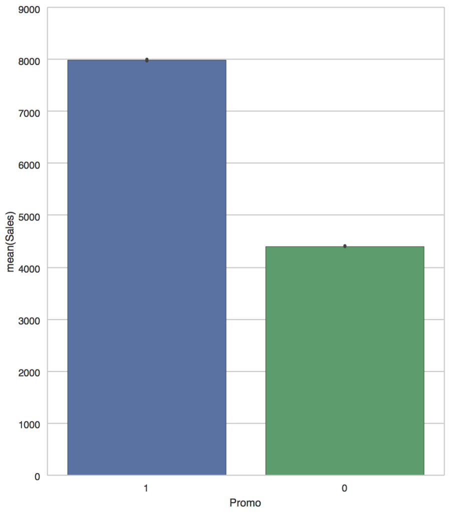
Sales do not vary much based on year:



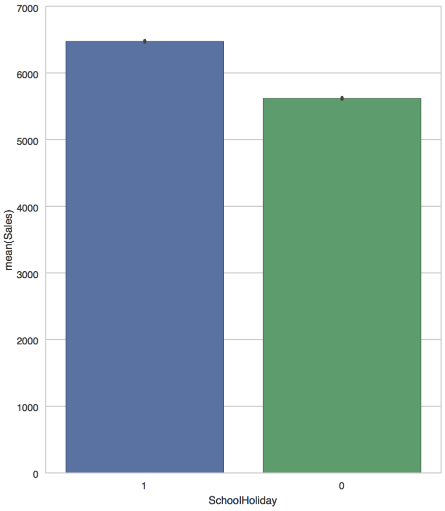
Since the stores are mostly closed on state holidays, sales are very low when it is a state holiday,



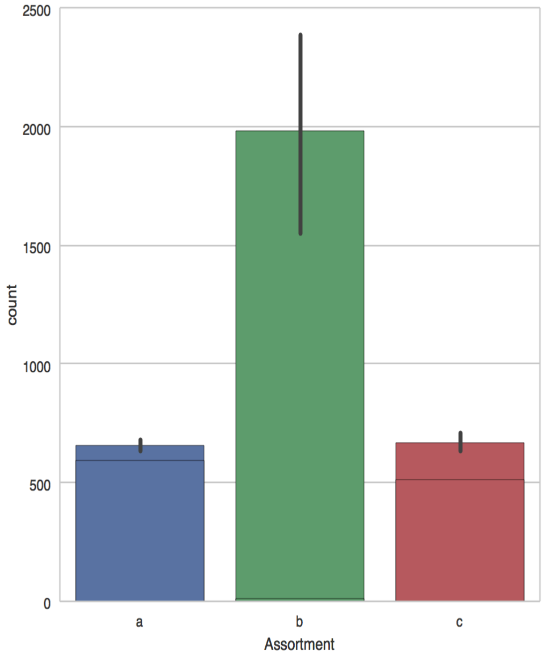
Distribution based on promo:



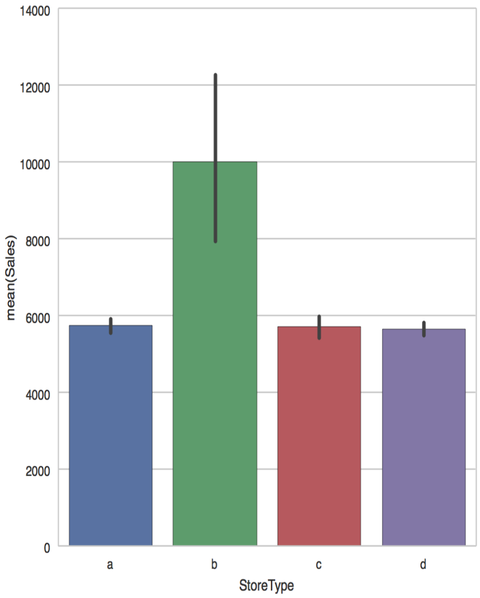
Based on school holiday:



Assortment:



Type:



Based on this information, we re ran the AR model, but also took into account the day of week, whether it was a school holiday or not, and whether there was a promo on that day or not. There is also distribution of sales based on Assortment level, and type of store but making a model based on groups leads to over fitting of data, as we saw in the previous experiment (4). Moreover, since we are building a model for each store, these properties will be implicitly included in the model.

Experiment 8:

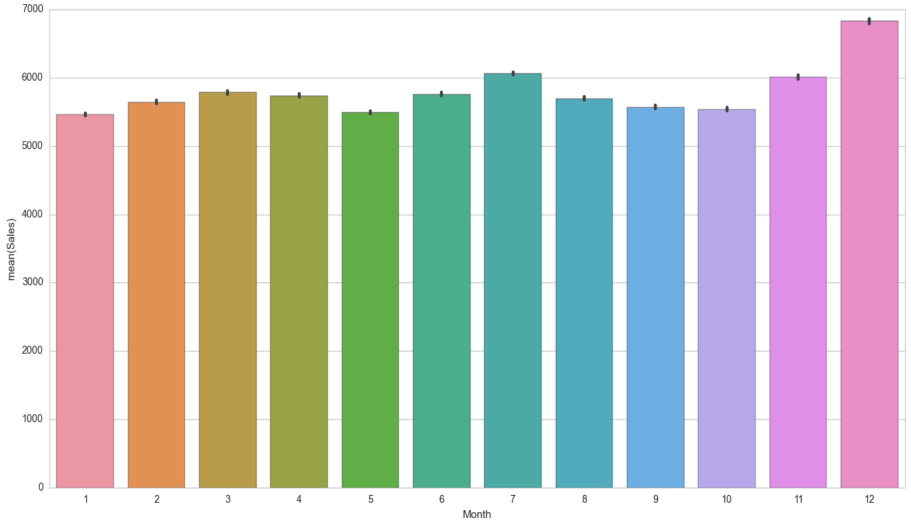
Observation:

Beta does not explode. Noted the model for each store separately and re ran linear regression on the test data set with the month also taken into consideration.

Evaluation:

Ran the test set provided by kaggle and submitted the result to kaggle.

This is the distribution over months with sales peaking in December:



Experiment 9:

Ran the normalized dataset to find beta using closed form approach but took into account the month as well, since there are clear seasonal trends based on the month.

Observation:

Beta does not explode. Noted the model for each store separately and re ran linear regression on the test data set with the month also taken into consideration.

Evaluation:

Ran the test set provided by kaggle and submitted the result to kaggle. The error score was 0.43413, which is again not an improvement.

Experiment 10:

Ran the ARIMA model with parameter estimates of (7,0,2)

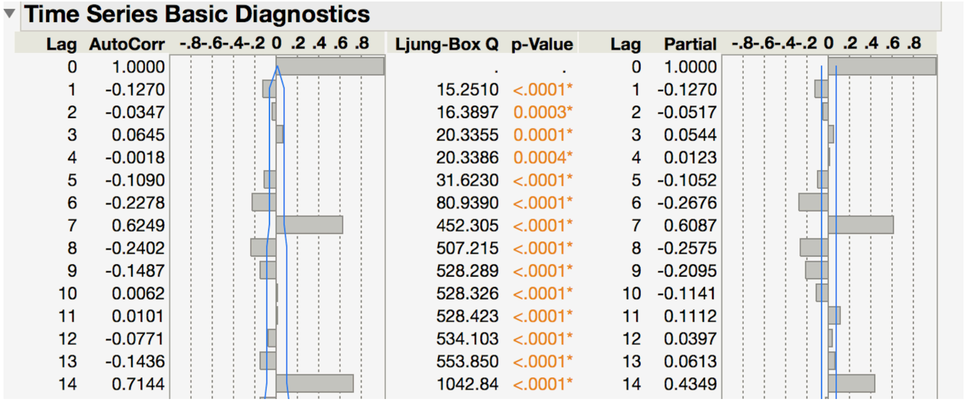
Evaluation:

The score was 0.22554, which is better than the simple AR models, but not an improvement over the best score obtained by closed form linear regression. Probably since the data contains seasonal cycles, Seasonal ARIMA may provide a better prediction model for this dataset.

Progress Discussion and Conclusion:

So far the best prediction model was provided by closed form linear regression. LR outperforms the ARMA model. Most likely this is because in our time series model, error is propagating a lot because time granularity is per day. Further work can be done for improvement in predictions:

- Use line space search to get an optimal beta in stochastic gradient descent



Time Series Basic Diagnostics

Lag	AutoCorr	-.8-.6-.4-.2 0 .2 .4 .6 .8	Ljung-Box Q	p-Value	Lag	Partial	-.8-.6-.4-.2 0 .2 .4 .6 .8
0	1.0000		.	.	0	1.0000	
1	-0.1270		15.2510	<.0001*	1	-0.1270	
2	-0.0347		16.3897	0.0003*	2	-0.0517	
3	0.0645		20.3355	0.0001*	3	0.0544	
4	-0.0018		20.3386	0.0004*	4	0.0123	
5	-0.1090		31.6230	<.0001*	5	-0.1052	
6	-0.2278		80.9390	<.0001*	6	-0.2676	
7	0.6249		452.305	<.0001*	7	0.6087	
8	-0.2402		507.215	<.0001*	8	-0.2575	
9	-0.1487		528.289	<.0001*	9	-0.2095	
10	0.0062		528.326	<.0001*	10	-0.1141	
11	0.0101		528.423	<.0001*	11	0.1112	
12	-0.0771		534.103	<.0001*	12	0.0397	
13	-0.1436		553.850	<.0001*	13	0.0613	
14	0.7144		1042.84	<.0001*	14	0.4349	

- As we can see from the ACF and PACF of a typical store, there are clear spikes at 1,7,14 indicating a 7 month seasonal pattern. We can therefore probably find appropriate model parameters of p,d,q using autocorrelation or partial autocorrelation functions, for the ARIMA(p,d,q) model.

- Since the time frame for the data is too narrow, we cannot detect any evident trend in the data, however since seasonal cycles are clearly seen in the graph, one possible approach would be to use SARIMA to make accurate predictions
- To reduce the error propagation in time series, roll up the time granularity for month, and make monthly predictions, and then redistribute the predicted value by noting the probability distribution over a typical month based on day, date, holiday or promo.
- In order to eliminate the effect of seasons, we can apply 1st or 2nd order of differencing on the data to obtain a more stable time series.

References:

1.) http://www.ccs.neu.edu/home/yzsun/classes/2015Fall_CS6220/Slides/03Matrix_Data_Prediction.pdf
2.) http://cs229.stanford.edu/notes/cs229-notes1.pdf
3.) Data Mining: Concepts and Techniques, Jiawei Han, Micheline Kamber, and Jian Pei. 3rd edition, Morgan Kaufmann, 2011
4.) https://www.kaggle.com/c/rossmann-store-sales
5.) http://userwww.sfsu.edu/efc/classes/biol710/timeseries/timeseries1.htm
6.) http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc4.htm
7.) http://people.duke.edu/~rnau/411arim3.htm

Task distribution form:

Task	People
1. Collecting and preprocessing data	Jeet, Hriya
2. Implementing Linear Regression	Jeet, Hriya
3. Generating and Analyzing graphs	Jeet, Hriya
4. Implementing Time Series Models	Jeet, Hriya
5. Report	Jeet, Hriya