Predicting sales using time series forecasting

Presented by
Akansha Jajodia
Raaga Pranitha Kolla
Saranya Visvanathan

Team Name - APS
CMPE 257 - Project Presentation
Instructor - Prof Mahima Agumbe Suresh

Problem Statement and Motivation

Problem: Predict the sales of Walmart products across different states and different categories for 28 days given the historical data.

Motivation: There are many regression problems where time plays a major role. The time component makes the prediction more difficult and it needs to be handled in a specific way to get better predictions. In short, time series utilizes the past information and helps to make better predictions on future horizons. It provides application in many domains like Sales Prediction, Weather Prediction, yield prediction, price prediction, etc

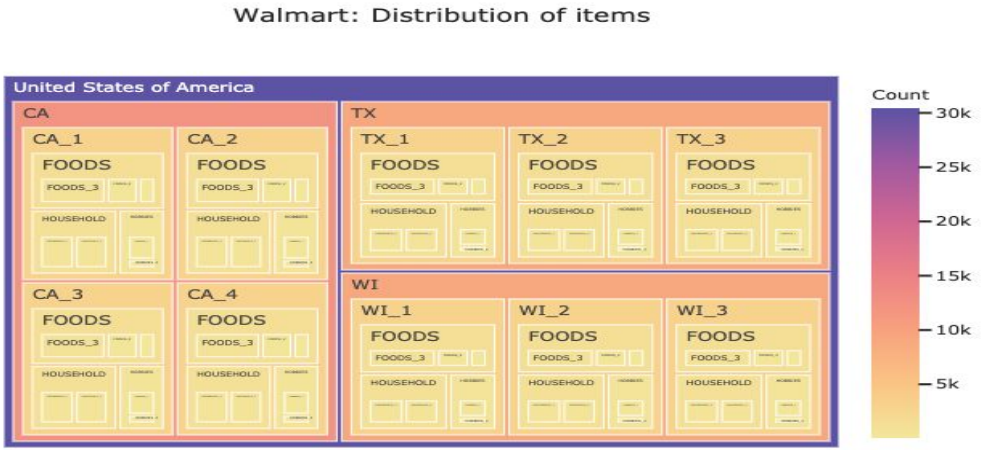
Solution: To predict the sales, we implement machine learning algorithms like ARIMA, FBProphet, LGBM.

Dataset Description

- The dataset is collected from M5 competition. It has sales data for 1941 days.
- This dataset contains details about sales of multiple items aggregating them into different departments, categories, stores and geographical areas. It is discrete and multivariate time series data.
- The calendar.csv file specifies the days where there is a special event.
- The sell_prices.csv file gives the details of sales regarding to each store and category.
- The sales_train_evaluation.csv contains information of all the sales from day_1 to day_1941

Ref: https://mofc.unic.ac.cy/m5-competition/

Dataset Organization



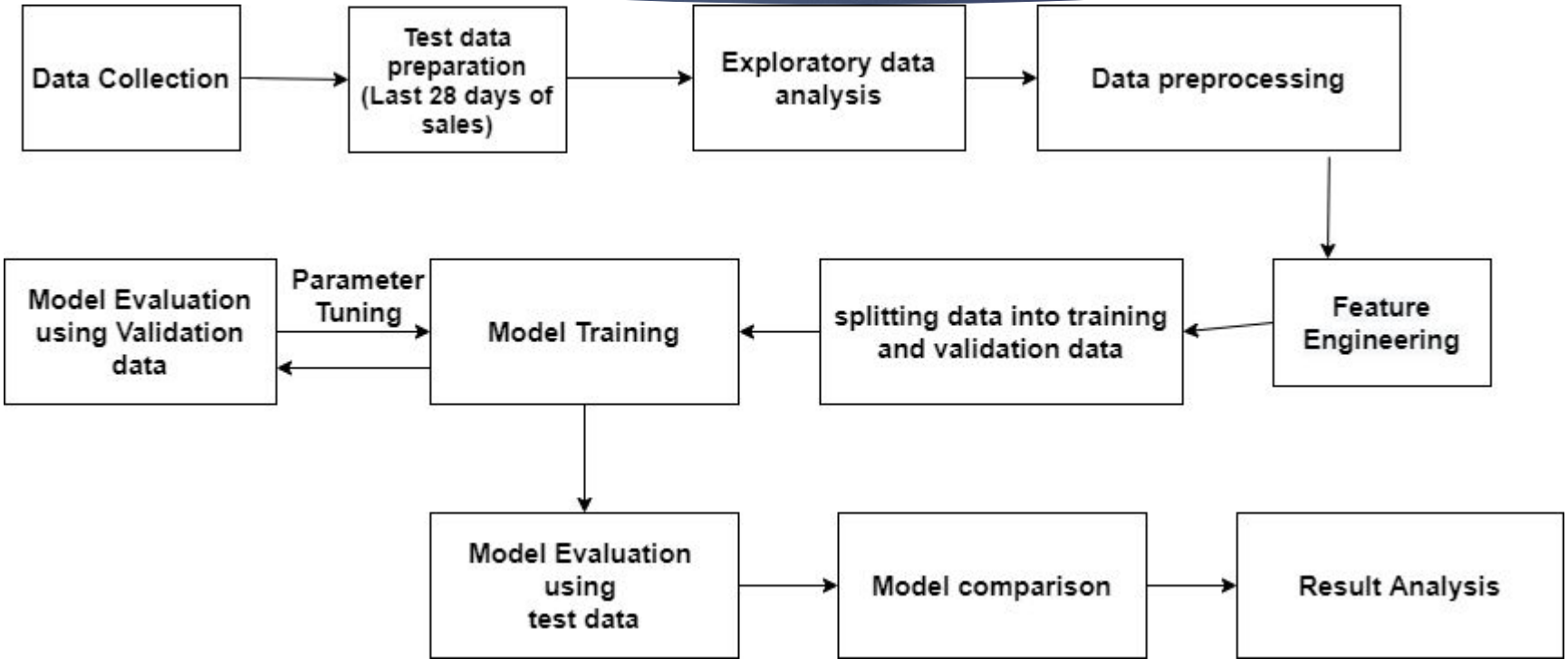
Reference: [Kaggle]

Time Series data

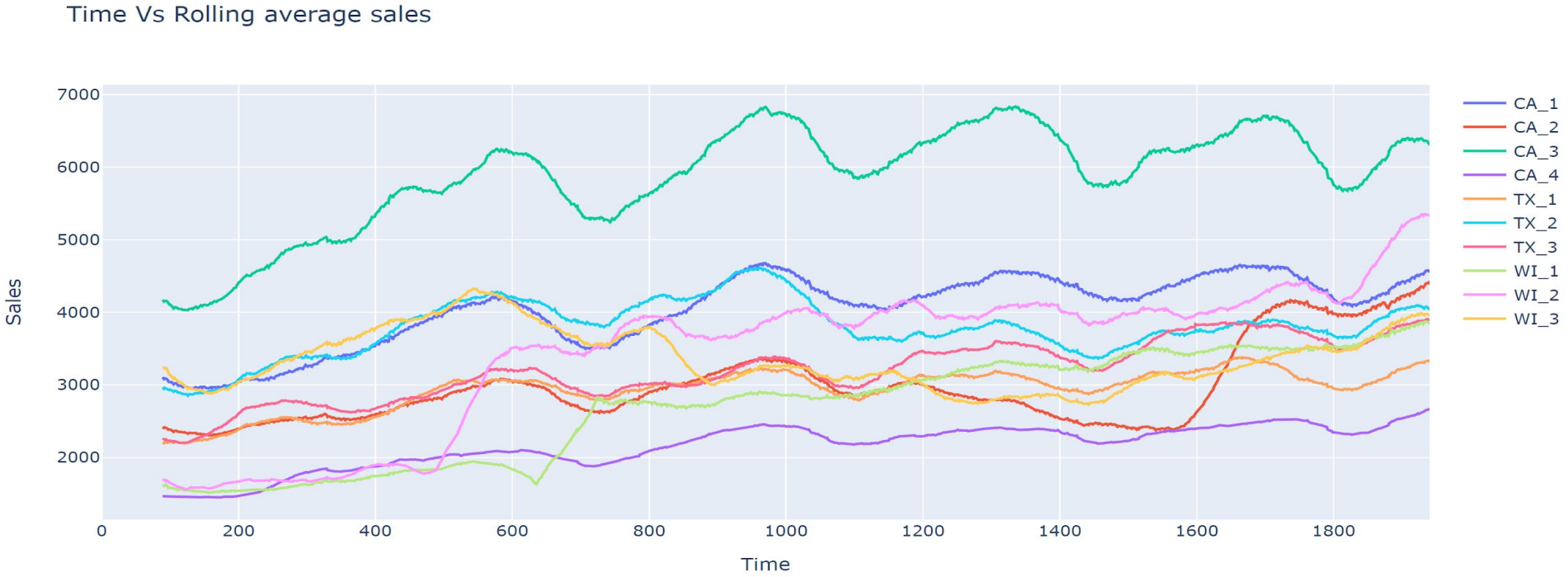
- Historical data ordered sequentially over time.
- There might be a correlation between past and future values which forms the components of time-series such as trend, seasonality, stationarity, etc.,

id	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	...	d_1932	d_1933	d_1934	d_1935	d_1936	d_1937	d_1938	d_1939	d_1940	d_1941
FOODS_3_823_WI_3_evaluation	0	0	2	2	0	3	1	4	1	...	1	0	3	0	1	1	0	0	1	1
FOODS_3_824_WI_3_evaluation	0	0	0	0	0	5	0	1	1	...	0	0	0	0	0	0	1	0	1	0
FOODS_3_825_WI_3_evaluation	0	6	0	2	2	4	1	8	5	...	0	0	1	2	0	1	0	1	0	2
FOODS_3_826_WI_3_evaluation	0	0	0	0	0	0	0	0	0	...	1	1	1	4	6	0	1	1	1	0
FOODS_3_827_WI_3_evaluation	0	0	0	0	0	0	0	0	0	...	1	2	0	5	4	0	2	2	5	1

Project's ML Pipeline

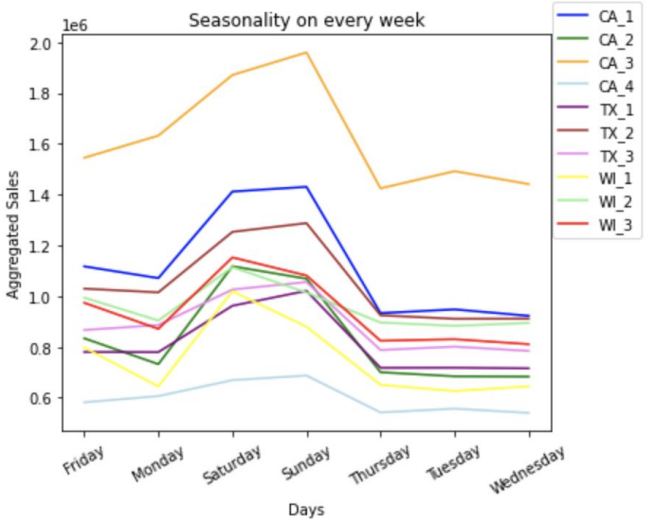


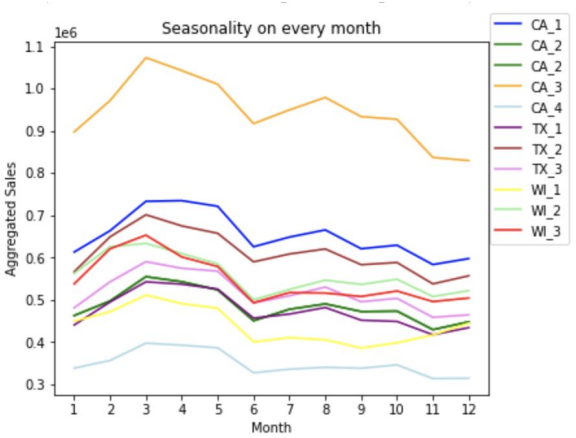
Exploratory Data Analysis - Trend



Exploratory Data Analysis - seasonality

- Seasonality: Repeating pattern of fixed frequency over a certain period of time.





ARIMA

- ARIMA stands for Auto Regressive Integrated Moving Average

  AR-Auto Regressive, **I**-Integrated, **MA**-Moving Average

- One of the effective machine learning models to predict time-series analysis.
- How does ARIMA work?

ARIMA -Data Preprocessing and Feature Engineering

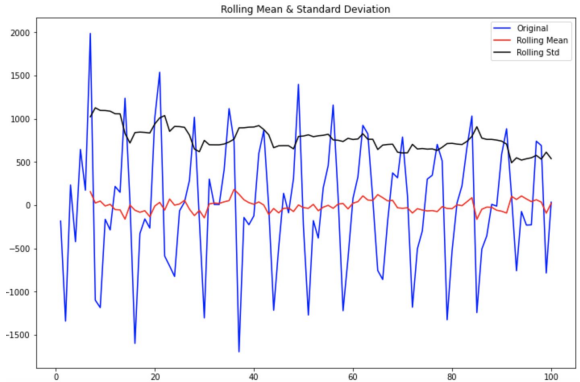
- To predict the sales , first the data is aggregated into total sales per store, per category and per department.
- Then, this dataframe is merged with calender.csv to capture the importance of event days in the sales.

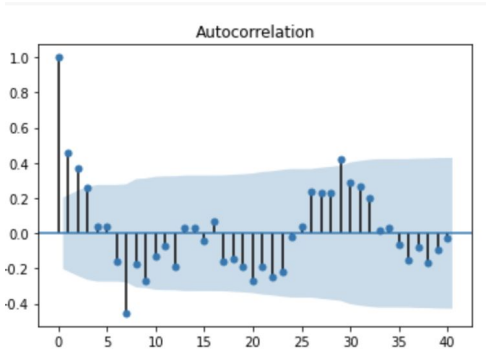
index	CA_1	CA_2	CA_3	CA_4	TX_1	TX_2	TX_3	WI_1	WI_2	WI_3	d
d_1	4337	3494	4739	1625	2556	3852	3030	2704	2256	4038	d_1
d_2	4155	3046	4827	1777	2687	3937	3006	2194	1922	4198	d_2
d_3	2816	2121	3785	1386	1822	2731	2225	1562	2018	3317	d_3
d_4	3051	2324	4232	1440	2258	2954	2169	1251	2522	3211	d_4
d_5	2630	1942	3817	1536	1694	2492	1726	2	1175	2132	d_5
d_6	3276	2288	4369	1389	2734	3439	2833	2049	2244	4590	d_6
d_7	3450	2629	4703	1469	1691	2588	1947	2815	2232	4486	d_7
d_8	5437	3729	5456	1988	2820	3772	2848	3248	2643	5991	d_8
d_9	4340	2957	5581	1818	2887	3657	2832	1674	2140	4850	d_9
d_10	3157	2218	4912	1535	2174	2932	2213	1355	1836	3240	d_10
d_11	2995	2123	4447	1368	1607	2628	1989	1335	1526	3053	d_11
d_12	2710	1901	4544	1195	2149	2669	1922	1410	1846	3312	d_12
d_13	2928	2436	4406	1434	1895	2515	1699	1349	1785	3309	d_13
d_14	3078	2584	4380	1312	2256	2883	2063	1946	2015	3928	d_14
d_15	4316	3455	5187	1652	2944	4000	3137	2634	2442	5066	d_15
d_16	4354	3563	5780	1933	3432	4779	3514	2012	2336	4677	d_16
d_17	2757	1801	3303	1183	1921	2679	2119	1522	1612	2907	d_17
d_18	2430	1797	3758	1511	2217	2927	2316	1363	1980	3771	d_18
d_19	2272	1741	3641	1190	1962	2606	2038	1300	1628	3065	d_19
d_20	2009	1614	3099	1113	1821	2449	2142	1459	1678	2934	d_20
d_21	3015	2214	3320	1476	1951	2615	2145	1872	1813	3300	d_21
d_22	4553	3080	4694	1543	2571	3705	2874	2596	2014	4059	d_22
d_23	3966	3087	4794	1618	2810	3845	3134	1415	1592	3022	d_23

	CA_1	CA_2	CA_3	CA_4	TX_1	TX_2	TX_3	WI_1	WI_2	WI_3	d	date	wm_yr_wk	weekday	wday	month	year	event_
0	4337	3494	4739	1625	2556	3852	3030	2704	2256	4038	d_1	2011-01-29	11101	Saturday	1	1	2011	
1	4155	3046	4827	1777	2687	3937	3006	2194	1922	4198	d_2	2011-01-30	11101	Sunday	2	1	2011	
2	2816	2121	3785	1386	1822	2731	2225	1562	2018	3317	d_3	2011-01-31	11101	Monday	3	1	2011	
3	3051	2324	4232	1440	2258	2954	2169	1251	2522	3211	d_4	2011-02-01	11101	Tuesday	4	2	2011	
4	2630	1942	3817	1536	1694	2492	1726	2	1175	2132	d_5	2011-02-02	11101	Wednesday	5	2	2011	

ARIMA -Determining parameters

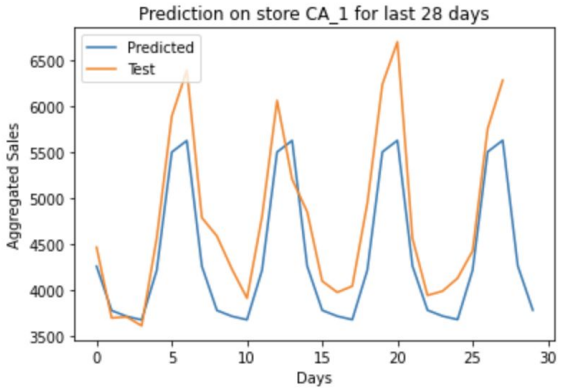
- Finding parameters for p,q, and d
- Final selected parameters order=(1,1,1) with seasonality 7





SARIMA (Evaluation)

```
##Using SARIMAX
import statsmodels.api as sm
mod = sm.tsa.statespace.SARIMAX(train, order=(1,1,1), seasonal_order=(1,1,1,7))
results=mod.fit()
print(results.summary())
```



- ARIMA model with above parameters is fitted on the train dataset which contains 1885 days of the sales, then validated on next 28 days and tested on the last 28 days.
- Sample Arima prediction on last 28 days sales on a store.

LGBM (Light Gradient Boosted Machines)

- Open source gradient boosting framework
- It uses boosting technique of ensemble learning.
  Boosting: A set of models learn sequentially based on the feedback from the previous models, thereby provides strong overall prediction.
- It works with trees. By default, it uses GBDT (Gradient Boosted Decision trees) [GBDT - uses boosting technique and gradient descent optimization function]
- Referred as “light” because of its high speed and ability to handle large scale of data.

LGBM - Data Preprocessing

- Make column data types appropriate. (Default: object datatype for all)
- Handle null values (no missing values)
- Drop unnecessary/repeated attributes.
- Convert categorical attributes into numerical attributes.
  Categorical attributes: An attribute which has a set of values which is repeated over the column.
- Merge multiple dataframes into single dataframe.

LGBM - Data Preprocessing

Reshape dataframe by converting multiple time series into a single time series.

```
print(df_sales.shape)
df_sales[["item_id","d","sales"]].head(10)
```

```
(58327370, 8)
```

	item_id	d	sales
0	HOBBIES_1_001	d_1	0
1	HOBBIES_1_002	d_1	0
2	HOBBIES_1_003	d_1	0
3	HOBBIES_1_004	d_1	0
4	HOBBIES_1_005	d_1	0
5	HOBBIES_1_006	d_1	0
6	HOBBIES_1_007	d_1	0
7	HOBBIES_1_008	d_1	12
8	HOBBIES_1_009	d_1	2
9	HOBBIES_1_010	d_1	0

LGBM - Feature Engineering

- **Lag features (n = shift range)**
  The past values are known as lags. For a time t, lags are t-1, t-2 and so on.Assuming past sales as future sales by shifting values of a column to n positions, where n can be any integer within row size.

- **Rolling statistics (mean, median, sum, max, min) based features**
  Taking the mean of the values of the sliding window of n size.

Optimal value of n is obtained from ACF (Auto correlation plot).

LGBM - Train test split

- Splitting data into training and validation data in time series is different.
- No random shuffling of data, that would disrupt the ordering in the data.
- Data is splitted based on location. Last 28 days of information is splitted as validation data.

LGBM - Building model

lgb.Dataset({data}) creates dataset as required by lgbm model.

Set lgbm parameters

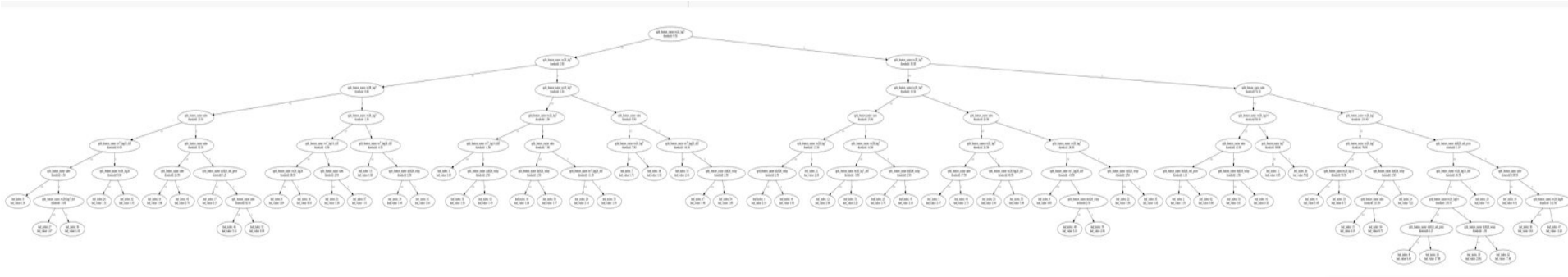
lgb.train({data}) trains the model

Use lgb.predict({data}) to predict sales

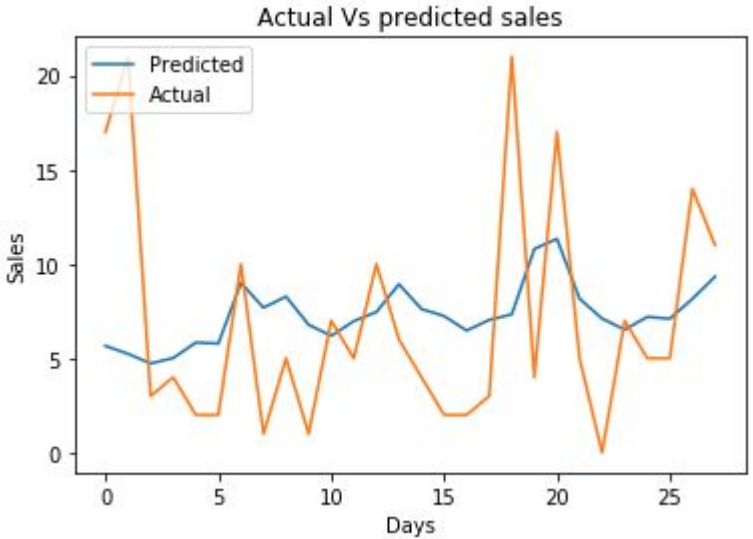
Ein: 2.61

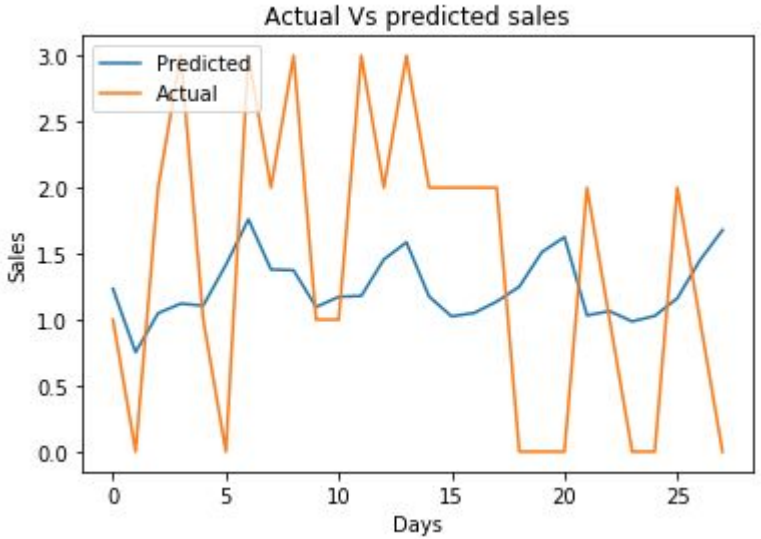
```
"""
Model training with hyperparameters
"""
import lightgbm as lgb
def train_lgbm(xtrain, xval, ytrain, yval):
    #create dataset as per lgb model expects
    train_data = lgb.Dataset(xtrain, label=ytrain)
    val_data = lgb.Dataset(xval, label=yval)

    #lgbm Hyperparameters
    params = { "seed":200, #random_state
               "learning_rate": 0.07,
               "num_leaves": 64,
               "force_row_wise": True,
               "colsample_bytree": 0.85, #to help with overfitting,
               "metric":"rmse",
               "objective": "regression"}#uses L2 regularization

    evals_result = {}
    #model training
    lgbmodel = lgb.train(params, train_data,
                         num_boost_round=800,
                         valid_sets = [val_data],
                         evals_result=evals_result,
                         early_stopping_rounds=25, verbose_eval=5)

    #prediction on training and validation data
    trainpred = lgbmodel.predict(xtrain)
    valpred = lgbmodel.predict(xval)

    return lgbmodel, evals_result, trainpred, valpred
```

LGBM Results



LGBM Results - contd





FBProphet

- Open source model from Facebook
- Used for time series modelling
- Robust to missing data
- Input format is ds, y

  Ds- date format

  Y - feature to predict

PreProcessing

```
sales_eval = sales_eval.drop(['item_id','dept_id','cat_id','store_id','state_id'], axis=1)
valid_cols = ['d_'+str(1913+i) for i in range(1,29)]
sales_eval.columns = sales_eval.columns.tolist()[:-28]+valid_cols
sales_eval.columns
```

```
1  item1 = item1.drop('id').T.reset_index().merge(calendar[['d','date']], left_on='index', right_on='d', how='left').drop(['index','d'], axis=1)
```

```
1  item1.head()
```

1 to 5 of 5 entries

index	0	date
1936	0	2016-05-18
1937	3	2016-05-19
1938	3	2016-05-20
1939	0	2016-05-21
1940	1	2016-05-22

Model

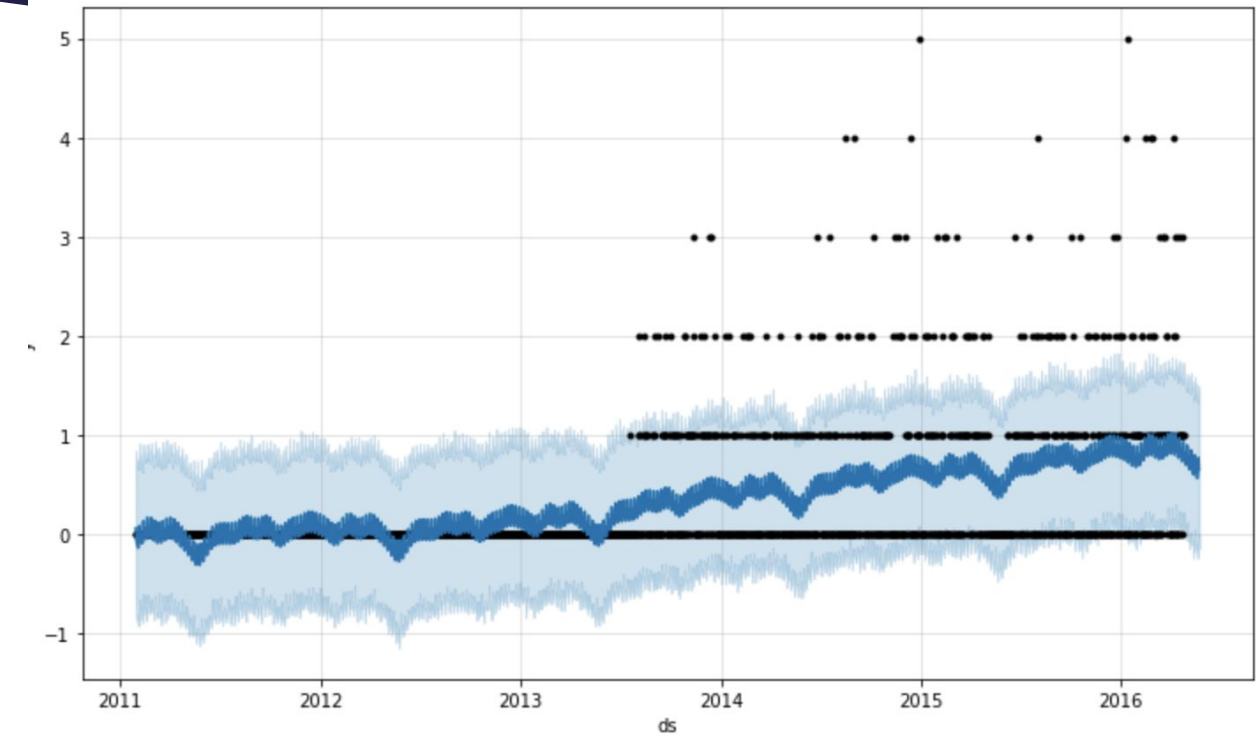
```
ph = Prophet()
ph.fit(train_item)
forecast = ph.predict(item[['ds']])
```

```
1  forecast[['ds', 'yhat','trend','weekly','yearly']]
```

1 to 25 of 1941 entries Filter

index	ds	yhat	trend	weekly	yearly
0	2011-01-29 00:00:00	0.06167351745378734	-0.03200154615202118	0.13540948368644817	-0.041734420080639684
1	2011-01-30 00:00:00	-0.0542318670471616	-0.03189456937861307	0.018213648411160747	-0.04055094607970927
2	2011-01-31 00:00:00	-0.09721679032810453	-0.03178759260520496	-0.02687934679204037	-0.0385498509308592
3	2011-02-01 00:00:00	-0.028850541671607423	-0.03168061583179685	0.038574293574611396	-0.03574421941442197
4	2011-02-02 00:00:00	-0.13547820789453702	-0.03157363905838874	-0.0717415063482419	-0.032163062487906374
5	2011-02-03 00:00:00	-0.1351761888983734	-0.03146666228498063	-0.07585833165651036	-0.027851194956882402
6	2011-02-04 00:00:00	-0.0719466409540354	-0.03135968551157252	-0.017718240875967458	-0.022868714566495422
7	2011-02-05 00:00:00	0.08686668798655256	-0.03125270873816441	0.13540948368659145	-0.017290086961874496
8	2011-02-06 00:00:00	-0.024134935354424812	-0.031145731964756295	0.018213648411359105	-0.011202851801027626
9	2011-02-07 00:00:00	-0.06262407784934924	-0.03103875519134818	-0.02687934679207566	-0.0047059758659254
10	2011-02-08 00:00:00	0.009734626149387533	-0.03093177841794007	0.03857429357446616	0.0020921109928614544
11	2011-02-09 00:00:00	-0.09349055587858776	-0.03082480164453196	-0.07174150634835288	0.009075752114297087
12	2011-02-10 00:00:00	-0.09045163700646222	-0.03071782487112385	-0.07585833165655913	0.016124519521220737
13	2011-02-11 00:00:00	-0.025213259931088243	-0.03061084809771574	-0.017718240875910965	0.023115829042538467

Result



Model Evaluation and comparison

Model	Eout - RMSE (Root Mean Squared Error) (Model evaluation metric)
ARIMA	0.67
FB Prophet	0.627
LGBM	2.2

Project Challenges

- Understanding time-series data, and finding ways to utilize it for better prediction results.
- Model selection and choosing evaluation metric.

Future works

- Hyperparameter tuning to improve results for all the models
- LGBM: deploying various feature engineering techniques, analyze the influence of time series components in the prediction
- FBProphet: predicting sales for each item. Plot the error measures.
- ARIMA: predicting sales for each item of each store.
- Result analysis
- Project Report completion

Conclusion

- The time series components plays a major role in prediction results, it needs to be processed accordingly.
- Different models expects different preprocessing techniques.
- While splitting the data, the order needs to be preserved, always past data needs to be used to predict future values.
- Even simple ML models work better for a smaller range of predictions. (More analysis can be done by increasing the range of predictions)

Thank You!

Any Questions?