UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Building segment based revenue prediction for CLV model

Author:
Kateryna ZORINA

Supervisor:
Olexander ROMANKO

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2019

Declaration of Authorship

UKRAINIAN CATHOLIC UNIVERSITY

Abstract

Faculty of Applied Sciences

Master of Science

Building segment based revenue prediction for CLV model

by Kateryna ZORINA

This work presents part of customer lifetime value calculation project. Two tasks are described: dividing customers into segments and forecasting of future revenue. For both components, there are metrics to compare performance between different experiments.

Acknowledgements

I want to thank people who made this project possible: Dean McKeown (Queens University), Mark Wagner (Scotiabank), Oleksandr Romanko (Ukrainian Catholic University / UoT), Mikhail Nediak (Queens University), Oleksii Molchanovskyi (Ukrainian Catholic University). Also special gratitude to GRM and GL teams in Scotiabank, especially to Ivan Mendoza who provided support and supervision on all stages of work. Also, I want to thank Oleksandr Romanko (Ukrainian Catholic University / UoT) and Mikhail Nediak (Queens University) as they provided valuable and regular feedback of results of my work.

I want to say special thank you to Ukrainian Catholic University and Oleksii Molchanovskyi for the Data Science Master Program which was a turning point in my life. And I am very grateful to Ring Ukraine as they provided me with a scholarship which made my studying possible.

Contents

List of Figures

List of Tables

List of Abbreviations

CLV	**C**ustomer **L**ifetime **V**alue
ARIMA	**A**uto**R**egressive **I**ntegrated **M**oving **A**verage
RMSE	**R**oot **M**ean **S**quare **E**rror
KNN	**K**-**N**earest **N**eighbors
SVM	**S**upport **V**ector **M**achine
LSTM	**L**ong **S**hort-**T**ert **M**emory
DBSCAN	**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
RNN	**R**ecurrent **N**eural **N**etwork
NLP	**N**atural-**L**anguage **P**rocessing
AIC	**A**kaike **I**nformation **C**riterion

Dedicated to my mother who always supports me

Chapter 1

Introduction

Customer lifetime value is an important metric, which is used in many fields such as marketing, financial analytics, etc. This calculation is so valuable as it incorporates information about future cohort/customer behavior and profit. Obtained in this project results will go as an input to credit limit optimization strategy.

Credit limit optimization is an essential part of risk management. As some customers can pay regularly and others can have significant debt. And for every unpaid credit bank has resources to cover for it. To minimize losses, we should give an optimal limit to customers. But to keep user satisfied bank have to provide a sufficient amount of money. So it is a tradeoff between customers satisfaction and bank losses.

For now, most of the credit limit optimization strategies are built with current historical data about balance and revenue. The idea behind this work is to include into account future value (CLV). To do so, we need to calculate the amount of time that customer will continue to use our product and his future revenue.

There are many approaches to calculate CLV. But most of them work on a general level or cohort level and use average values. This calculation can be essential to understanding your business overall. But for credit limit optimization, we need to go to the user level.

In this work, we explore ways to predict CLV on customer level and remain computationally effective. A conventional technique is to group customers into segments, and we use it for understanding high-level features of each segment.

1.1 Goals

This thesis resulting from part of the project on industry partner side and there are goals for this research work:

1. Explore existing works on CLV calculation
2. Develop two components of CLV model:
   - clustering: divide customers into groups to capture user behavior
   - forecasting: predict revenue for the future period for each customer
3. Build an automated framework for CLV score calculation

1.2 Chalanges and limitations

- The only available dataset is monthly data for past year and revenue for 43 month

- Customer behavior varies through time, and it is hard to assign a specific cluster. We have to make an assumption that user behave the same in one year, which is not correct

- Revenue data is highly non-stationary which makes prediction harder. Also, many users have a lot of zero-revenue months, and it is complicated to fit any model in this scenario. This problem was one of the reasons to use customer segments

Chapter 2

Background and Related Work

2.1 CLV calculation

Many works and online materials (such as Paul D. Berger, 1998 and *Whitepaper: Customer Lifetime Value*) suggest to calculate CLV using one mathematical formula which usually is a combination of these components:

- average customer lifespan (one number for all customers or per cohort)
- retention rate
- profit margin
- discount rate
- costs

The problem with this method is that you will use only average numbers from historical data and don't use information about customer behavior. Also, this calculation is usually performed for all customers, or on cohort level when our approach works on customer level so you can obtain a more specific score.

Other works try to take into account different behavior. For example, Donkers, Verhoef, and de Jong, 2007 use two levels of the model, one - relationship-level - for calculating annual profit contribution; second - service-level - which tries to capture behavior and contribution per service. This models also use customer retention for calculation and conclude that it is not sufficient to focus only on this.

Sharad Borle, 2008 use hierarchical Bayes approach to calculate lifetime based on spending pattern. The model works with interpurchase time, purchase amount and probability of closing account.

Harsha Aeron, 2008 presented a model for revenue which incorporates different states of a customer. This model is developed for credit card customers only. Using simple business rules authors defined five states: inactive, transact, revolver, delinquent, default. Then they asses the probability to of existence in some state (using Markov Model) and transaction amount (from data). Also, they use rules for the relationship between transacting, revolving, delinquency state amount. In the end, CLV formula consists of variables and parameters that can be obtained from data.

2.2 Customers segmentation

Customer segmentation is used for many purposes such as targeting promotion, analysis, etc. As EY, 2016 stated, most banks have outdated customer segmentation techniques based on demographic factors (age, profession, geographic location). And when it comes to financial data, this information is not enough for building robust systems.

Li et al., 2011 used a k-means algorithm to cluster data among three types of variables: income, consumption amount, personal credit standing. After detecting main clusters, tree-based data mining method is used to find rules for forming obtained clusters. In result, they have a set of rules to explain customer groups and these rules are used for better understanding of related features of different customers.

As classical segmentation algorithms (such as k-means) are not always applicable to financial data due to high variation in data and a large number of outliers, many works are dedicated to alternative approaches. Such as Shashidhar HV, 2011 proposed to use a heuristic approach instead of K-means as it is more robust with outliers. Customers are segmented based on loan overdue amount and security value.

When K-Means Clustering Fails: Alternatives for Segmenting Noisy Data suggests using few alternative techniques for market segmentation. For example, GMM provides probability estimates of cluster membership, not "hard label". Also, HDBSCAN allows noisy data and does not require some clusters to be set.

Cai, Le-Khac, and Kechadi, 2016 described applying different clustering techniques for financial datasets: partitioning methods such as k-means, density-based such as DBSCAN, data stream clustering such as hierarchical agglomerative clustering. In this work, they obtained results that show that density-based clustering is not a good choice for financial data.

Micciche, Lillo, and N. Mantegna, 2005 suggested using correlation-based approach for clustering stock markets data. A minimum spanning tree is applied for correlation coefficients matrix. The resulting models can be used to validate market strategy.

Comparing Time-Series Clustering Algorithms in R Using the dtwclust Package suggest to use dynamic time warping distance to overcome some of the limitations of Euclidean distance in case of time series financial data. Also, time-series prototyping is described which is a procedure for finding centroid for each time series cluster.

2.3 Time series prediction

One of the essential parts of CLV calculation in this project is the prediction of future profit based on historical data.

Streimikiene et al., 2018 analysed efficiency of AR, ARIMA, VAR models for tax revenue forecasting. Data was taken for 31 years. ARIMA model worked best with the lowest RMSE.

Udom, 2014 compared ARIMA, exponential smoothing (Holt-Winter) and MA approaches. Data is 108 month of sales. Results showed that ARIMA gets the best accuracy. However, data is highly seasonal.

Jocelyn Barker and Conners, 2018 suggest to use an automated approach which combines time series patterns with additional information. The model framework consists of feature extraction, a generic regression model with cross-validation, model selection, and interval prediction generation. For modeling, they use KNN, Elastic Net, Random forest model and an SVM with the radial kernel. Authors report that this methodology is better than manual forecast computed by traditional methods.

Diebold, 2017 suggest to use various approaches to treat seasonality in financial data. Important technique is described: use in the model seasonal dummies D_i. Deterministic seasonal component is $S_t = \sum_{i=1}^{s} \gamma_i D_{it}$ and γ_i's are called seasonal factors.

Siami-Namini and Namin, 2018 and Kohzadi et al., 1996 report that LSTM outperforms ARIMA model with financial. However, they train models on a rather large dataset (40 years) and do not compare with models more robust to seasonal data, like SARIMA.

Chapter 3

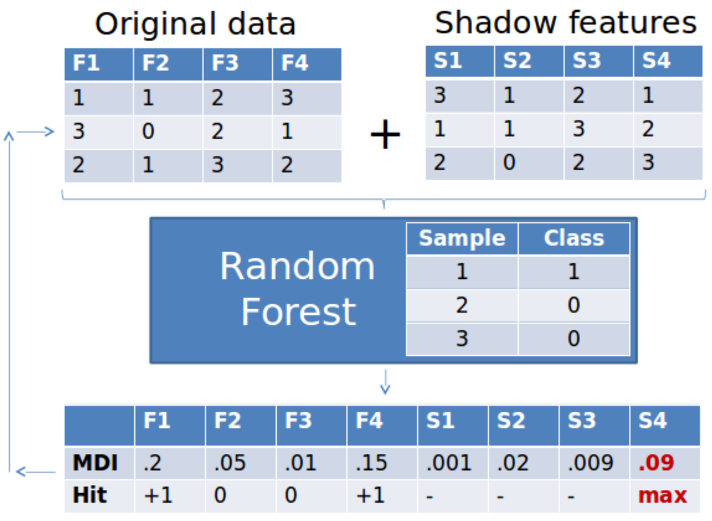
Overview of existing methods

3.1 Feature importance

As we have a lot of data, we have to use some methods to reduce the number of features before running the model. Miron B. Kursa, 2010 suggest to use Boruta algorithm for finding all relevant columns. This method is a wrapper around Random Forest.

When we use Random Forest, we can obtain importance measure for each feature which is the loss of accuracy of RF if we randomly switch values of a column between objects. After that, we can compute Z-score (divide average loss by its deviation). But this score is not directly related to the statistical significance of the feature importance.

That is where Boruta can help us. Algorithm mixes real features and randomly shuffled (shadow) features and checks if real features have higher Z-score than random (if they do happen - it called "hit") Figure 3.1. If a feature doesn't have hits - it considered not to be important.



F1	F2	F3	F4
1	1	2	3
3	0	2	1
2	1	3	2

S1	S2	S3	S4
3	1	2	1
1	1	3	2
2	0	2	3

Sample	Class
1	1
2	0
3	0

	F1	F2	F3	F4	S1	S2	S3	S4
MDI	.2	.05	.01	.15	.001	.02	.009	**.09**
Hit	+1	0	0	+1	-	-	-	**max**

FIGURE 3.1: Boruta algorithm (*Feature Selection in R with the Boruta R Package*)

3.2 Clustering

3.2.1 Kmeans

Kmeans give partitions which are reasonably efficient in the sense of within-class variance (MacQueen, 1967). Objective function 3.1 is sensitive to different scales so to avoid one of the features to skew prediction it is crucial to normalize data or scale if one feature should have more impact than other.

$$\mathbf{J} = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 \tag{3.1}$$

3.2.2 Hierarchical clustering

Hierarchical clustering is a general family of clustering algorithms that build nested clusters by merging or splitting them successively. This hierarchy of clusters is represented as a tree (or dendrogram) (*Documentation of scikit-learn package*).

At the start of the algorithm, each point is a cluster, and these clusters start to merge based on linkage criteria. In this experiments, we use "ward" criteria.

Ward's method says that the distance between two clusters, A and B, is how much the sum of squares will increase when we merge them (*Distances between Clustering, Hierarchical Clustering*):

$$\begin{aligned}\Delta(A,B) = \sum_{i\in A\cup B} \|\vec{x}_i - \vec{m}_{A\cup B}\|^2 - \sum_{i\in A} \|\vec{x}_i - \vec{m}_A\|^2 - \sum_{i\in B} \|\vec{x}_i - \vec{m}_B\|^2 = \\ = \frac{n_A n_B}{n_A + n_B} \|\vec{m}_A - \vec{m}_B\|^2\end{aligned} \tag{3.2}$$

3.2.3 Spectral Clustering

Spectral clustering does a low-dimension embedding of the affinity matrix between samples, followed by a KMeans in the low dimensional space (*Documentation of scikit-learn package*).

For cluster building similarity graph $G = (V, E)$ is used. When similarity measure s_{ij} is more than the threshold - the vertices are connected and weighted by s_{ij}. The problem of clustering can now be reformulated using the similarity graph: we want to find a partition of the graph such that the edges between different groups have very low weights (which means that points in different clusters are dissimilar from each other) and the edges within a group have high weights (which means that points within the same cluster are similar to each other) (*A Tutorial on Spectral Clustering*).

3.2.4 DBScan

To find a cluster, DBSCAN starts with an arbitrary point p and retrieves all points density-reachable from p with respect to ϵ and minimum points attribute. If p is a core point, this procedure yields a cluster. If p is a border point, no points are density-reachable from p, and DBSCAN visits the next point of the database (Ester et al., 1996).

3.2.5 Gaussian mixture

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters (*Documentation of scikit-learn package*).

This type of model is useful when data has several groups and observations from one group can be modeled with a normal distribution. Mathematically we can write it as 3.3 where for each group we have centroid μ_k, variance ϵ_k and weight π_k.

$$p(X) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{3.3}$$

To find the best parameters we need to maximize log of the likelihood function 3.4. For this purpose, the EM algorithm is used. Bishop, 2006

$$\ln p(X|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(x_n) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \tag{3.4}$$

On Figure 3.2 different cluster algorithms are shown for different data.

One more algorithm that is used in this work is the Markov Cluster Algorithm for graph clustering (*MCL - a cluster algorithm for graphs*). This algorithm works with networks and uses a simulation of stochastic flow in graphs. Two operators are used: expansion and inflation. Expansion computes random walks (sequence of random travel to a connected node) of higher length and associates new probabilities with all pairs of nodes. As long paths are more common inside cluster - probabilities of inside-cluster nodes will be larger. Inflation will demote inter-cluster walks. Iterating over these two procedures results in the separation of the graph into different segments (Figure 3.3).

3.3 Time series prediction

3.3.1 SARIMA

Autoregressive integrated moving average processes ($ARIMA(p,d,q)$) model consists of 3 terms:

- autoregressive process of order p 3.5

- integration of order d, which means that the model is designed for d difference of original time series

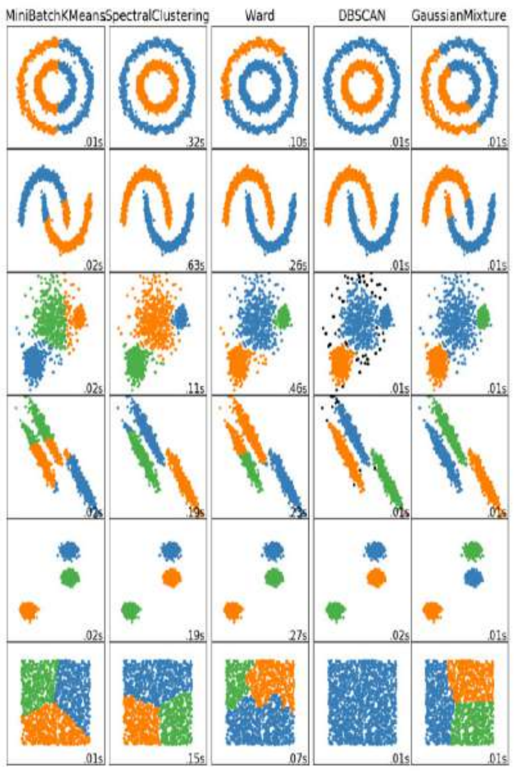


FIGURE 3.2: Visualization of difference between algorithms (*Documentation of scikit-learn package*)

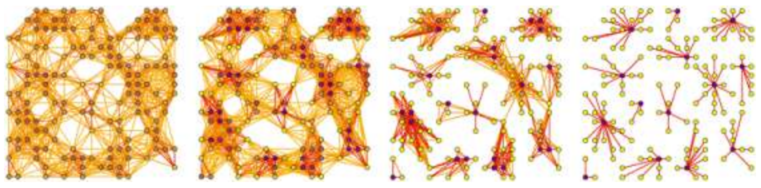


FIGURE 3.3: Visualization of MLC algorithm (*MCL - a cluster algorithm for graphs*)

- moving average process of order q 3.6
For example, formula for $ARIMA(1,1,1)$ is show in 3.7.

$$Y_t = \sum_{v=1}^{p} \alpha_v Y_{t-v} + u_t \\ (1 - \sum_{v=1}^{p} \alpha_v L^v) Y_t = u_t \tag{3.5}$$

$$Y_t = u_t + \beta_1 u_{t-1} + ... + \beta_q u_{t-q} \\ Y = (1 + \sum_{v=1}^{q} \beta_v L^v) u_t \tag{3.6}$$

$$(Y_t - Y_{t-1}) = \alpha_1 (Y_{t-1} - Y_{t-2}) + u_t + \beta_1 u_{t-1} \tag{3.7}$$

For seasonal data Box, Jenkins, and Reinsel, 2008 suggested to use seasonal AR and MA terms 3.8. In this way formula for $SARIMA(1,1,1)x(1,1,1)_{12}$ will look like 3.9.

$$\begin{aligned} Y_t &= \sum_{v=1}^{p} \alpha_v Y_{t-sv} + u_t \\ Y_t &= u_t + \beta_1 u_{t-s} + ... + \beta_q u_{t-sq} \end{aligned} \tag{3.8}$$

$$(1-\alpha_1 L)(1-\alpha_1^{(s)} L^{12})(1-L)(1-L^{12})Y_t = (1+\beta_1 L)(1+\beta_1^{(s)} L)u_t \tag{3.9}$$

To compare the quality of different models, AIC can be used. $AIC = 2k - 2ln(\hat{L})$ where k is the number of parameters and $\hat{L}$ is the maximum of the likelihood function of the model.

3.3.2 Exponential smoothing (Holt-Winters)

Exponential smoothing is a moving average when we weight the historical observations. (Brown, 1959)

$$\hat{Y}_{t+1} = \alpha(Y_t - \hat{Y}_t) + \hat{Y}_t = \alpha Y_t + (1-\alpha)\hat{Y}_t \quad with \quad \alpha \in (0,1] \tag{3.10}$$

The Holt-Winters seasonal method comprises the forecast equation and three smoothing equations — one for the level L_t, one for the trend T_t, and one for the seasonal component S_t, with corresponding smoothing parameters α, β and γ. Hyndman and Athanasopoulos, 2018

$$\begin{aligned} L_t &= \alpha(Y_t - S_{t-s}) + (1-\alpha)(L_{t-1} + T_{t-1}) \\ T_t &= \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1} \\ S_t &= \gamma(Y_t - L_t) + (1-\gamma)S_{t-s} \\ \hat{Y}_{t+h} &= L_t + T_t h + S_{t-s+h} \end{aligned} \tag{3.11}$$

3.3.3 LSTM

LSTM is a kind of RNN that captures long-term dependencies. The first time they appear in Hochreiter and Schmidhuber, 1997.

The main idea of LSTM is to use so-called memory cells (Figure 3.4) which allow to protect memory contents from the influence of irrelevant inputs (input gate unit) and on the other hand protect other units from possible influence of memory content (output gate unit).

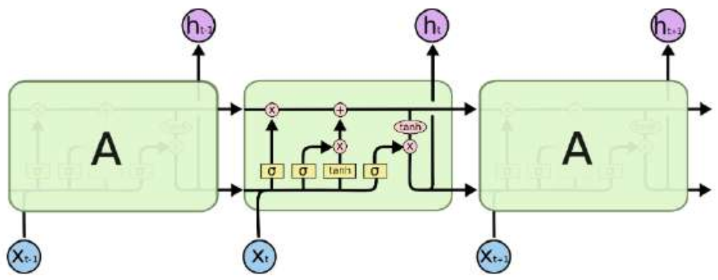


FIGURE 3.4: Module of LSTM networks (*Understanding LSTM Networks*)

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \hat{C}_t &= tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \hat{C}_t \\ o_t &= \sigma(W_o [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * thn(C_t) \end{aligned} \tag{3.12}$$

Chapter 4

Experiments

4.1 Dataset

Dataset is obtained from my industry partner, and it includes data about credit card users: general information, demography, historical revenue, balance and spending, credit history, scores. All data is taken on a fixed date (September 2017), and all historical variables are calculated for a year before that date (balance, limits). The only exception is revenue data. This data is for 43 months (31 months before September 2017 used for training model and 12 for testing, to calculate "revenue after" which is used as target variable in Boruta algorithm). For different models data is slightly different, but it is coming from the same source.

Before modeling piece there is data cleaning part, which consists of:
- removing sequential data (except time series data)
- generate dummies for categorical variables
- replace NA data (mostly with median)
- remove zero-variance and near-zero-variance columns
- trim outliers (0.99 quantiles)
- remove highly correlated features

For training, the model, a random sample of 50000 records is used to speed up the process and reduce memory usage.

4.2 Clustering

The idea behind the clustering is to group all customers into clusters/segments to investigate behavior per segment, build a forecast for different segments and calculate the probability to move from one segment to another.

4.2.1 Profit drivers based segmentation

Initially, we wanted to base segments on features that impact on customer profit. These so-called profit drivers we obtain from all available data. For this purpose, Boruta algorithm is used. To assess goodness of model inside Boruta I used R2 score. For target “future average revenue” (which is average of revenue for a period after stop point - last 12 months of revenue data) R2 = 0.62 and 30 features are picked as important. After that only top N are picked for clustering, usually N is 6-8.

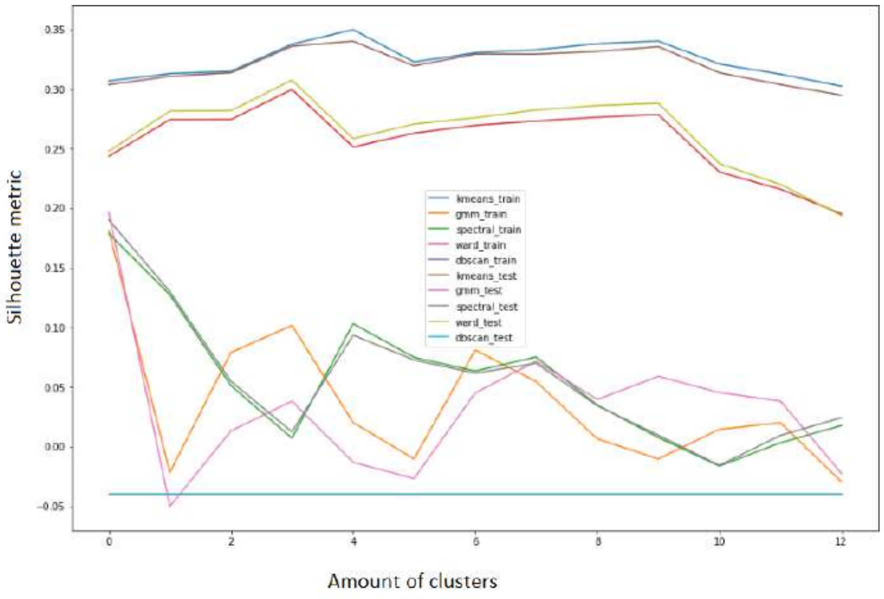
Next step is to segment customers based on profit drivers obtained on the previous step. I normalize data before passing to the algorithm. I tried both normalize evenly and give more important features bigger scale. But in the second case, one feature brings more impact and this leads to highly unbalanced clusters.

Unfortunately, there is no general theoretical solution to find the optimal number of clusters for any given data set. A simple approach is to compare the results of multiple runs with different k classes and choose the best one according to a given criterion, but we need to be careful because increasing k results in smaller error function values by definition, but also increases the risk of overfitting. *Unsupervised Learning and Data Clustering*

I used as criterion silhouette metric (Figure 4.1) and also looked at the histogram of labels to see if there are underrepresented clusters (Figure 4.2). Silhouette metric is measuring how close point is to its cluster and how far from all other clusters (4.1).

$$\begin{aligned} s(i) &= \frac{b(i) - a(i)}{maxa(i), b(i)} \\ b(i) &- minimalaveragedistancefromipointtootherclusterspoints \\ a(i) &- averagedistancefromipointtootherpointsfromsamecluster \end{aligned} \tag{4.1}$$

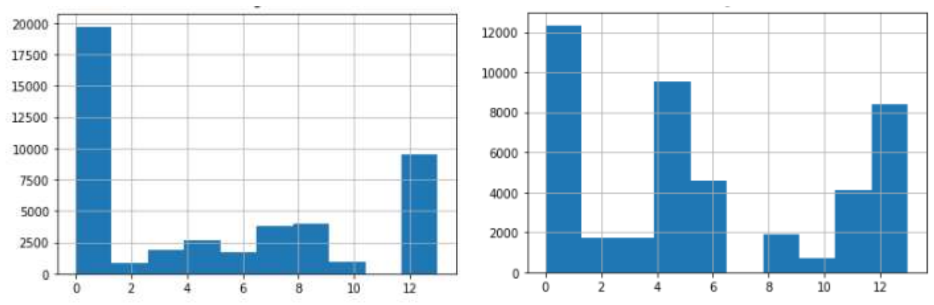
FIGURE 4.1: Silhouette metric for different clustering algorithms



As we can see on Figure 4.1, k-means has the best silhouette metric however one cluster is dominating (Figure 4.2). Balanced clusters are useful for future analysis, researching dependencies between segments.

Also, it is important to notice that the silhouette metric is not very high (0.35) and this tells us that clusters are not entirely separable. This pushed

FIGURE 4.2: Histogram of labels for kmeans(left) and ward(right), 13 clusters



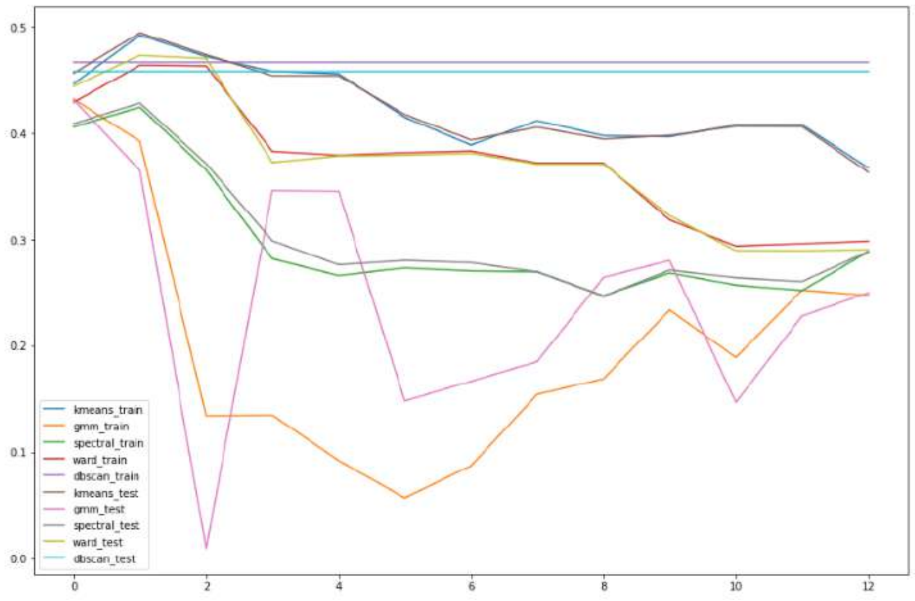
me to explore other ways to segment customers as this is a critical assumption of the model and if clusters are not good than all future models are misleading.

4.2.2 Timeseries based segmentation

Another idea was to segment on some features generated from time series, for example, mean revenue, std revenue, amount of zero-revenue month.

On Figure 4.3 we can see that silhouette metric is even higher (however this may be caused by the fact that we have fewer features: 3 instead of 8 before). But in later analysis, this clustering strategy does not show good results.

FIGURE 4.3: Silhouette metric for different clustering algorithms

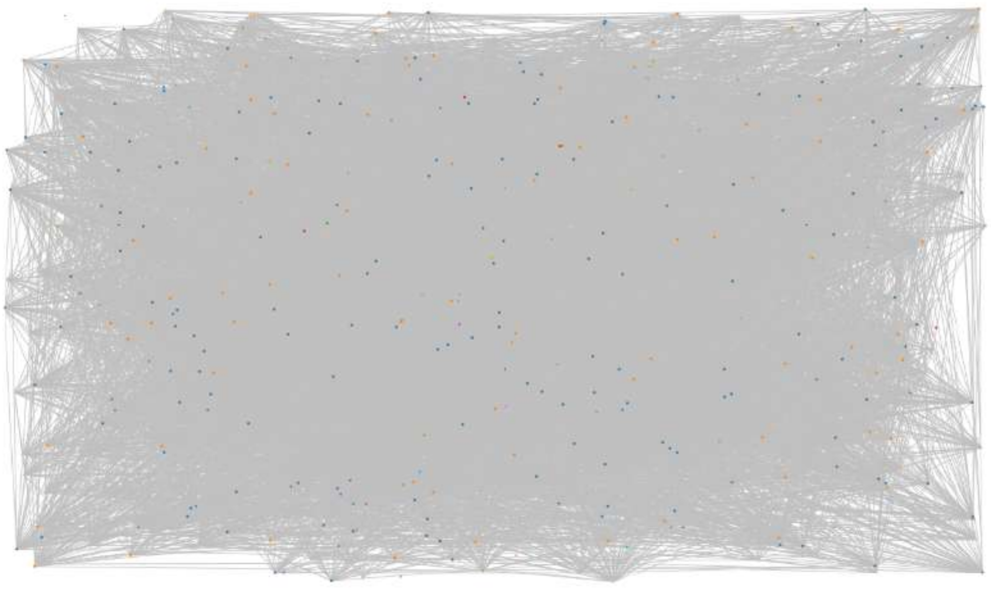


After that, I was thinking to base clustering on some time series correlation measure. As we want a similar time series to go inside one cluster and treat them similarly.

The first idea was to threshold correlation matrix and group to one cluster all of the time series that correlate more than a threshold. Grouping was very simple, the first time series is the first cluster and all highly correlated time series go to this cluster too. However even with reasonable threshold (0.6) amount of clusters was too large: 523 clusters for 5000 sample.

Other idea was to threshold correlation matrix and get the incidence matrix of a graph. After that apply Markov Cluster Algorithm to get clusters. Figure 4.4 shows the result of such manipulations. And still the problem is the same as mentioned before, amount of clusters is too large (71 cluster for 500 users, which is more than 10%) and this segmentation is not useful as it is too detailed.

FIGURE 4.4: Visualization of clustered graph



In the end, we decided to stick with k-means clustering for now and research ways to improve in later works.

4.3 Time series prediction

Initially, the idea for time series prediction was to find the best ARIMA model for each user segment (as they should have different behavior) and apply this model to predict future profit on a customer level. For this purpose, average revenue data per segment is used (sample 50000). Before applying ARIMA on the customer level, time series is logged to reduce non-stationarity and NA are replaced with 0.

4.3.1 SARIMA

SARIMA $(p, d, q \times (P, D, Q)_s$ with exogenous data of dummy months is used. For each segment average revenue best SARIMA model is found based on

FIGURE 4.5: Logged average time series and prediction with SARIMA and LSTM

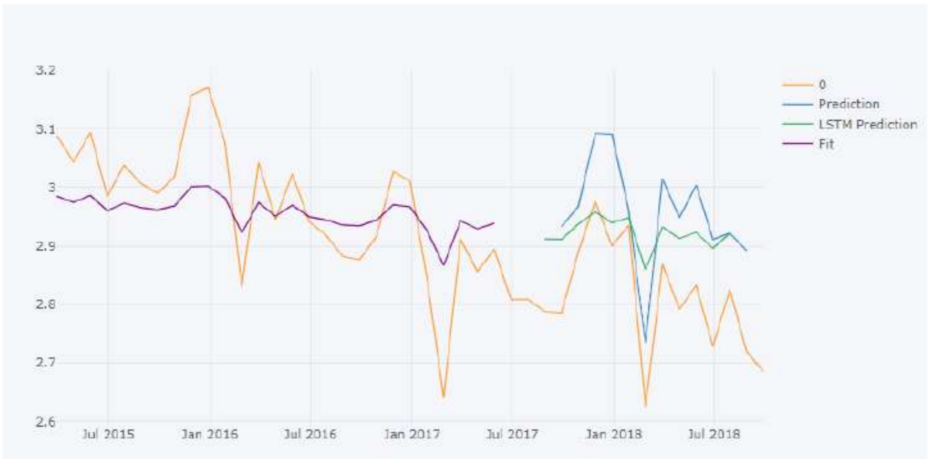
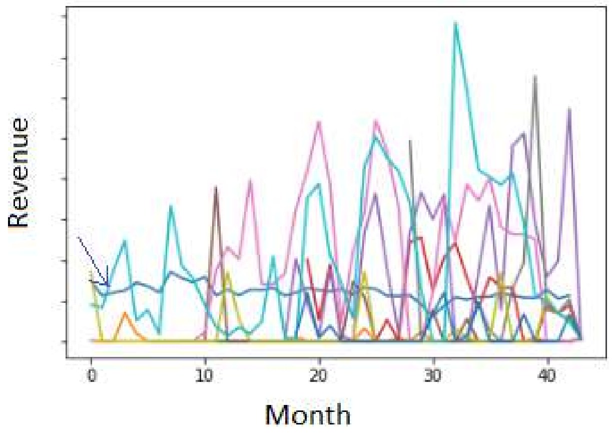


FIGURE 4.6: Average timeseries and 10 samples from one cluster



AIC. Also, best boxcox transformation parameter is defined (to treat non-stationarity). Then orders and parameters are frozen, and this model will be used for predicting user's revenue if they will fall in this cluster.

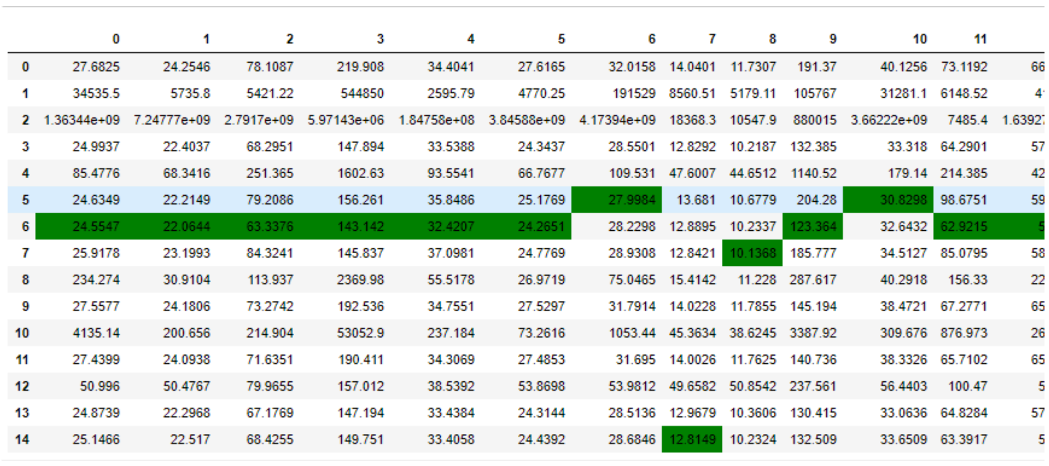
On Figure 4.5 we can see model prediction with logged average time series of one segment.

However, time series behavior inside the cluster is not always captured by average. (Figure 4.6 blue line is average per cluster, other - sample 10 time series from same cluster).

Average RMSE on logged average time series is 0.15, however, on all customers data RMSE is much higher. Also, one model performs better on all customers, which is not the behavior that was expected as the idea of clustering was to capture customers with similar behavior and pick the best model for this cluster.

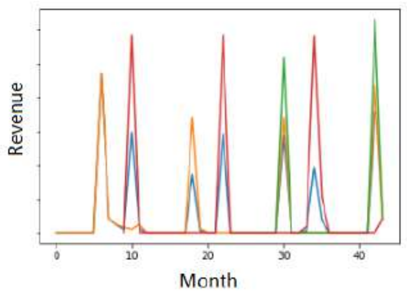
Figure 4.7 shows RMSE of all models on different clusters. For each of the 15 models (row 0 - model picked for cluster 0 based on average revenue), average RMSE on all customers from each segment is calculated. Minimum

FIGURE 4.7: RMSE of revenue for different models on customers from different clusters



	0	1	2	3	4	5	6	7	8	9	10	11	
0	27.6825	24.2546	78.1087	219.908	34.4041	27.6165	32.0158	14.0401	11.7307	191.37	40.1256	73.1192	66
1	34535.5	5735.8	5421.22	544850	2595.79	4770.25	191529	8560.51	5179.11	105767	31281.1	6148.52	4
2	1.36344e+09	7.24777e+09	2.7917e+09	5.97143e+06	1.84758e+08	3.84588e+09	4.17394e+09	18368.3	10547.9	880015	3.66222e+09	7485.4	1.63927
3	24.9937	22.4037	68.2951	147.894	33.5388	24.3437	28.5501	12.8292	10.2187	132.385	33.318	64.2901	57
4	85.4776	68.3416	251.365	1602.63	93.5541	66.7677	109.531	47.6007	44.6512	1140.52	179.14	214.385	42
5	24.6349	22.2149	79.2086	156.261	35.8486	25.1769	27.9984	13.681	10.6779	204.28	30.8298	98.6751	59
6	24.5547	22.0644	63.3376	143.142	32.4207	24.2651	28.2298	12.8895	10.2337	123.364	32.6432	62.9215	5
7	25.9178	23.1993	84.3241	145.837	37.0981	24.7769	28.9308	12.8421	10.1368	185.777	34.5127	85.0795	58
8	234.274	30.9104	113.937	2369.98	55.5178	26.9719	75.0465	15.4142	11.228	287.617	40.2918	156.33	22
9	27.5577	24.1806	73.2742	192.536	34.7551	27.5297	31.7914	14.0228	11.7855	145.194	38.4721	67.2771	65
10	4135.14	200.656	214.904	53052.9	237.184	73.2616	1053.44	45.3634	38.6245	3387.92	309.676	876.973	26
11	27.4399	24.0938	71.6351	190.411	34.3069	27.4853	31.695	14.0026	11.7625	140.736	38.3326	65.7102	65
12	50.996	50.4767	79.9655	157.012	38.5392	53.8698	53.9812	49.6582	50.8542	237.561	56.4403	100.47	5
13	24.8739	22.2968	67.1769	147.194	33.4384	24.3144	28.5136	12.9679	10.3606	130.415	33.0636	64.8284	57
14	25.1466	22.517	68.4255	149.751	33.4058	24.4392	28.6846	12.8149	10.2324	132.509	33.6509	63.3917	5

FIGURE 4.8: Average timeseries and 10 samples from one cluster



errors per segment are green. And we can see, that model for cluster 6 is good for almost all clusters when the expected behavior was to see minimum errors on diagonal.

Avg RMSE = 50.3 (with 15 clusters) - if count all customers with the best model. Avg RMSE > 1000 if count customers with the corresponding model.

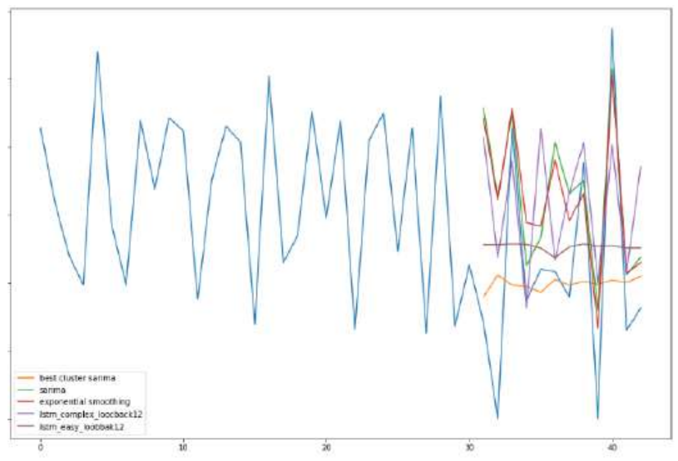
Then we tried to base clustering not on profit drivers but on attributes of time series such as mean, std, amount of 0-revenue months. With this approach behavior inside clusters became more similar (Figure 4.8):

However, the error was not better as still; we base picking of a model on average time series of the cluster. And on an example from Figure 4.8 we can see, that individual time series have one spike per year when average captures all of them. Avg RMSE for this clustering approach = 127.56 (with 5 clusters)

4.3.2 Exponential smoothing (Holt-Winters)

We also wanted to explore other possibilities of time series forecasting. However other methods are not suitable to train it on average per cluster and then apply to all of the time series from a cluster. So if we decide to use one of the following techniques - we don't need clustering and transition parts at all.

FIGURE 4.9: Actual time series and prediction by different models



I applied exponential smoothing on customer level and got Avg RMSE = 90.53 which is better than the error in case we use trained ARIMA per cluster, but still worse in case we use one best ARIMA model.

4.3.3 LSTM

To use the LSTM model for time series first step is to prepare the dataset. LSTM can predict the next step (word/number) based on inputs. So to train it with time series data, we have to break our sequence into a series of inputs and targets. Amount of observations in the input is usually called lookback parameter.

I tried different lookback parameters for building the dataset, 12 was one of the best (makes sense as we have seasonality period 12).

With simple model (< 10 dimensionalities of the output space) error is too high and for more complex model (100 dimensionality of the output space) I achieved Avg RMSE = ˜110 but running time is 1 hour for 100 customers and as we work with millions of record - this method is not appropriate.

Average errors show the whole picture, but it is interesting to look at some particular time series and model forecasts. On Figure 4.9 and Figure 4.10 there is prediction of different models for randomly picked revenue time series. On first image we can see, that exponential smoothing is the most accurate one. Also, SARIMA and complex LSTM are accurate, but SARIMA for the cluster that user belongs to is not good. On the second picture, it is the case when a user has only one purchase per year (probably some standard fee). SARIMA, exponential smoothing, and complex LSTM did a good job.

On Table 4.1 we can see a comparison of different models on two random sampled time series and average values on the whole dataset.

FIGURE 4.10: Actual time series and prediction by different models

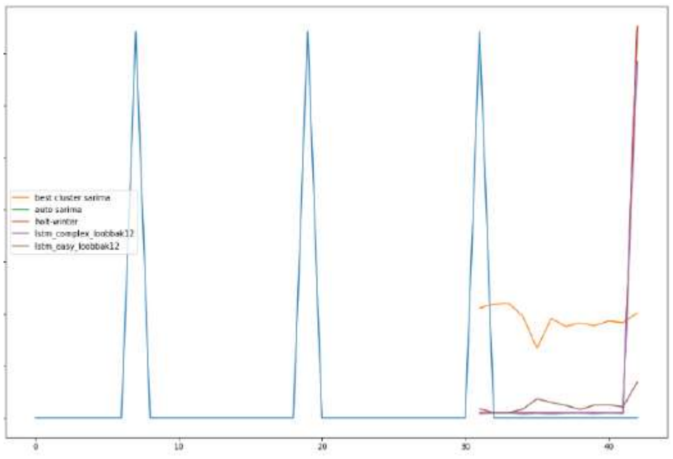


TABLE 4.1: Table of errors

Model	Random ts 1 RMSE	Random ts 2 RMSE	Avg RMSE
Best cluster SARIMA	263,5	1797,3	> 1000
Custom SARIMA	259,87	1,28	50.3
Holt-Winters	239,49	1,28	90,53
LSTM easy	271,34	369,7	> 1000
LSTM complex	267,49	2,47	˜110

4.4 CLV calculation

Calculation of lifetime value puts together all pieces of the model. It takes each user separately, predicts revenue, builds future transition (based on cluster/history).

```
Data: customer data, calculated survival lifetimes, transition
      probability
Result: CLV metric for each customer
delta = 0.91 - Discount parameter
for user in customer data do
    clv = 0
    for model in best models per cluster do
        build prediction (ARIMA), store to i row
    end
    for year in range(customer survival years) do
        yearAdd = multiply summed predictions for this year (array
          of length clusters amount) with a probability of being in a
          cluster on this year(array of length cluster amount)
        clv += delta*yearAdd
    end
end
```

For each customer, the prediction is calculated with all best models (one per cluster), and then this amount is multiplied on probability to be in this cluster. As probabilities are different for each year, time series are grouped into year sums. Standard practice in a bank is to use discount parameter for future predictions to account inflation and other instabilities.

This part can run only after all the models are trained. To reduce memory usage and speed up work all dataset is processed chunk by chunk, and the chunk size is 50000.

4.5 Architecture

On Figure 4.11 you can see a diagram which shows parts of the model, an interaction between them and a part that I was responsible for (in the orange box).

For now, the process is running on a local machine. All code is on Python 3.6; data is processed using pandas. There is a plan to move code to some cloud server and use more powerful tools for operations with data, such as Spark. But for now, due to regulatory restrictions of industry partner, this is not possible.

Model parts are trained one by one and serialized. So you can retrain some part of the model without touching others. However, some parts are dependent on other, for example, to train forecast models or transition you need to have cluster model trained.

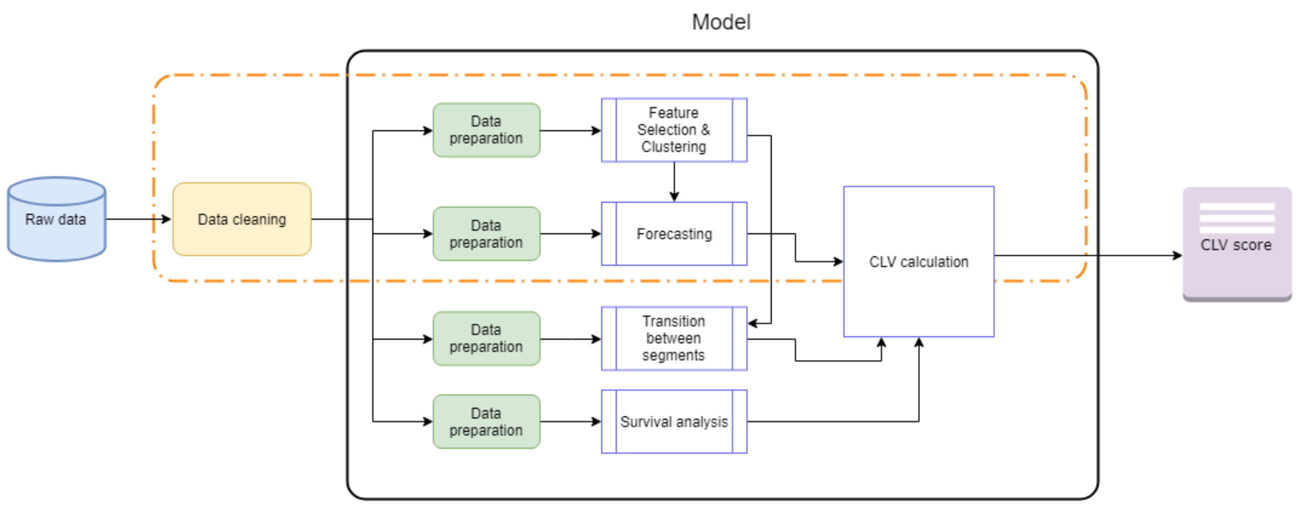


FIGURE 4.11: Diagram of project structure

Chapter 5

Discussion

5.1 Conclusions

Customer lifetime value can be used across different markets for various problems. For now, results from the designed model are tested in credit limit optimization strategy. CLV metric allows assigning correct limits depending on future user profits. In the testing stage, we noticed that CLV captures information about customers who are about to default much better than the score that was used before. If we apply this metric, we can optimize credit limits by approximately 30% (this is a percentage of users whose limits are assigned not correct according to results from CLV model, this information we got from CLA team).

The main difference of this work is including customer behavior and change of this behavior into the model. For this purpose, clusters are used. Users that spend more should be in one cluster, when users who spend less in other. And then we include information about probability to change behavior. However, for now, clusters are very volatile and not good separable. This tells us that we need to investigate this methodology further.

Another note is about historical revenue data. Sometimes we have a very short history or zero-revenue history. This is a problem to fit any model to this data. So it is important to come up with an approach to predict for this kind of users. In this work, we trained model parameters on average per cluster and then used those parameters for customer-level predictions.

As dataset is not open and specific (unique data, small history) - it is impossible to compare results with existing works. But even if we use good sense, we can see that forecasting results are not perfect. That is why some further work is required.

5.2 Future work

The system is dependant on user segments (assigned with clustering). As the idea is to catch user behavior with these clusters - it is essential for clusters to have users with the same or similar behavior. And for now, clusters are still very volatile. To improve this, we can use other data and explore techniques for feature selection.

The second idea is to incorporate into clustering both historical data of revenue and customer information. In this way, we can catch specifically the behavior of profit over time. So methods for look-alike time series can be useful. Also on this stage, we can smooth data and remove outliers. However, it is important to not corrupt data, for example, we have users who have only one spike per year and if we smooth this - we will lose information.

We can try to apply different models for different types of time series. For example, if historical data had one spike as mentioned above - we can use Holt-Winters and for other data - smooth before forecasting.

It is essential to work on the forecast prediction model as it is one of the most important parts of the CLV model. For now, we predict customer level and it is highly unstable if a customer has a short or highly volatile history. Maybe if we create more specific clusters/groups - we can build more robust predictions for this type of users. We can build initial groups based on rules and automatically cluster everything that is beyond rule.

As available dataset is rather short - it was hard to test how precise our CLV metric is. The good idea is to find some source with a large amount of historical data for additional testing on full lifetime. In this case, we can test the model for both short and longterm users and compare losses.

Bibliography

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag. Chap. 9.2". ISBN: 0387310738.

Box, G.E.P., G.M. Jenkins, and G.C. Reinsel (2008). *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. Wiley. ISBN: 9780470272848. URL: `https://books.google.com.ua/books?id=lJnnPQAACAAJ`.

Brown, Robert Goodell (1959). "Statistical forecasting for inventory control". In: New York : McGraw-Hill. Chap. 2.2. URL: `https://catalog.hathitrust.org/Record/001125212`.

Cai, Fan, Nhien-An Le-Khac, and Tahar Kechadi (Sept. 2016). "Clustering Approaches for Financial Data Analysis: a Survey". In:

Diebold, Francis X. (2017). *Forecasting in Economics, Business, Finance and Beyond*. University of Pennsylvania.

Documentation of scikit-learn package. URL: `https://scikit-learn.org/stable/documentation.html`.

Dongen, Stijn van. *MCL - a cluster algorithm for graphs*. URL: `https://micans.org/mcl`.

Donkers, Bas, Peter C. Verhoef, and Martijn G. de Jong (June 2007). "Modeling CLV: A test of competing models in the insurance industry". English. In: *Qme-Quantitative marketing and economics* 5.2, pp. 163–190. ISSN: 1570-7156. DOI: `10.1007/s11129-006-9016-y`.

Ester, Martin et al. (1996). "A Density-Based Algorithm for Discovering Clusters". In: *Proc. of 2nd International Conference on Knowledge Discovery and*, pp. 226–231. URL: `https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf`.

EY, Global Consumer Banking Survey (2016). *How well do you know your customers?* URL: `http://www.ey.com/Publication/vwLUAssets/EY-understanding-the-key-to-customer-loyalty-report/$FILE/EY-understanding-the-key-to-customer-loyalty-report.pdf`.

Gary LaRose, SimaFore LLC. *Whitepaper: Customer Lifetime Value*. URL: `http://cdn2.hubspot.net/hub/64283/file-15131808-pdf/pdfs/simafore-whitepaper-customer-lifetime-value-modeling.pdf`.

Harsha Aeron, Tarun Bhaskar et al (2008). "A metric for customer lifetime value of credit card customers". English. In: *Palgrave Macmillan* 15.3, pp. 153–168. ISSN: 1741-2439.

Hochreiter, Sepp and Jurgen Schmidhuber (1997). "LONG SHORT-TERM MEMORY". In: *Neural Computation 9(8)*, pp. 1735–1780. URL: `https://www.bioinf.jku.at/publications/older/2604.pdf`.

Hyndman, Rob J and George Athanasopoulos (2018). "FORECASTING: PRINCIPLES AND PRACTICE". In: OTexts. Chap. 7.3. URL: https://otexts.org/fpp2/holt-winters.html.

Jocelyn Barker Amita Gajewar, Konstantin Golyaev Gagan Bansal and Matt Conners (2018). "Secure and Automated Enterprise Revenue Forecasting". In: *The Thirtieth AAAI Conference on Innovative Applications of Artificial Intelligence (IAAI-18)*, pp. 7657–7664.

Kohzadi, Nowrouz et al. (1996). "A comparison of artificial neural network and time series models for forecasting commodity prices". In: *Neurocomputing* 10.2. Financial Applications, Part I, pp. 169 –181. ISSN: 0925-2312. DOI: https://doi.org/10.1016/0925-2312(95)00020-8. URL: http://www.sciencedirect.com/science/article/pii/0925231295000208.

Li, Wei et al. (Jan. 2011). "Credit Card Customer Segmentation and Target Marketing Based on Data Mining". In: pp. 73 –76. DOI: 10.1109/CIS.2010.23.

Luxburg, Ulrike von. *A Tutorial on Spectral Clustering*. URL: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=42D7FC59D1D60B085758AFD378F4688A?doi=10.1.1.165.9323&rep=rep1&type=pdf.

MacQueen, J. (1967). "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 1, pp. 281–297. URL: https://projecteuclid.org/euclid.bsmsp/1200512992.

Micciche, Salvatore, Fabrizio Lillo, and R N. Mantegna (Sept. 2005). "Correlation Based Hierarchical Clustering in Financial Time Series". In: DOI: 10.1142/9789812701558_0037.

Miron B. Kursa, Witold R. Rudnicki (Sept. 2010). "Feature Selection with the Boruta Package". English. In: *Journal of Statistical Software* 36.

Mishra, Sanatan. *Unsupervised Learning and Data Clustering*. URL: https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a.

Olah, Christopher. *Understanding LSTM Networks*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Pathak, Manish. *Feature Selection in R with the Boruta R Package*. URL: https://www.datacamp.com/community/tutorials/feature-selection-R-boruta.

Paul D. Berger, Nada I. Nasr (1998). "CUSTOMER LIFETIME VALUE: MARKETING MODELS AND APPLICATIONS". In: *JOURNAL OF INTERACTIVE MARKETING* 12.1, pp. 17–30. DOI: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.390.2785&rep=rep1&type=pdf.

Sarda-Espinosa, Alexis. *Comparing Time-Series Clustering Algorithms in R Using the dtwclust Package*. URL: https://cran.r-project.org/web/packages/dtwclust/vignettes/dtwclust.pdf.

Shalizi, Cosma. *Distances between Clustering, Hierarchical Clustering*. URL: https://www.stat.cmu.edu/~cshalizi/350/lectures/08/lecture-08.pdf.

Sharad Borle Siddharth S. Singh, Dipak C. Jain (Jan. 2008). "Customer Lifetime Value Measurement". English. In: *MANAGEMENT SCIENCE* 54.1, pp. 100–112. ISSN: 0025-1909. DOI: 10.1287/mnsc.1070.0746.

Shashidhar HV, Subramanian Varadarajan (2011). “Customer Segmentation of Bank based on Data Mining – Security Value based Heuristic Approach as a Replacement to K-means Segmentation”. English. In: *International Journal of Computer Applications* 19.8, pp. 13–18. ISSN: 0975 – 8887.

Siami-Namini, Sima and Akbar Siami Namin (2018). “Forecasting Economics and Financial Time Series: ARIMA vs. LSTM”. In: *CoRR* abs/1803.06386. arXiv: 1803.06386. URL: http://arxiv.org/abs/1803.06386.

Streimikiene, Dalia et al. (2018). “Forecasting tax revenues using time series techniques – a case of Pakistan”. In: *Economic Research-Ekonomska Istraživanja* 31.1, pp. 722–754. DOI: 10.1080/1331677X.2018.1442236. eprint: https://doi.org/10.1080/1331677X.2018.1442236. URL: https://doi.org/10.1080/1331677X.2018.1442236.

Sukup, John. *When K-Means Clustering Fails: Alternatives for Segmenting Noisy Data*. URL: https://www.datascience.com/blog/k-means-alternatives.

Udom, Patimaporn (Jan. 2014). “A comparison study between time series model and ARIMA model for sales forecasting of distributor in plastic industry”. In: *IOSR Journal of Engineering* 4. DOI: 10.9790/3021-04213238.